

Katsushi Ikeuchi
Editor

Computer Vision

A Reference Guide

Computer Vision

Katsushi Ikeuchi
Editor

Computer Vision

A Reference Guide

With 433 Figures and 16 Tables



Springer Reference

Editor

Katsushi Ikeuchi
Institute of Industrial Science
The University of Tokyo
Tokyo, Japan

ISBN 978-0-387-30771-8 ISBN 978-0-387-31439-6 (eBook)

ISBN 978-0-387-49138-7 (print and electronic bundle)

DOI 10.1007/978-0-387-31439-6

Springer New York Heidelberg Dordrecht London

Library of Congress Control Number: 2014936288

© Springer Science+Business Media New York 2014

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Computer vision is a field of computer science and engineering; its goal is to make a computer that can see its outer world and understand what is happening. As David Marr defined, computer vision is “an information processing task that constructs efficient symbolic descriptions of the world from images.” Computer vision aims to create an alternative for human visual systems on computers.

Takeo Kanade says, “computer vision looks easy, but is difficult. But, it is fun.” Computer vision looks easy because each human uses vision in daily life without any effort. Even a new-born baby uses its vision capability to recognize the mother. It is computationally difficult, however, because the original outer world is made up of three dimensional objects, while those projected on the retina or an image plane, are of only two dimensional images. This dimensional reduction from 3D to 2D occurs along the projection from the outer world to images. “Common sense” needs to be used to augment the descriptions of the original 3D world from the 2D images. Computer vision is fun, because we have to discover this common sense. This search for common sense attracts the interest of vision researchers.

The origin of computer vision can be traced back to Lawrence Roberts’ research, “Machine Perception of Three-Dimensional Solids.” Later, this line of research has been extended through Project MAC of MIT. Professor Marvin Minski, the then director of Project MAC, initially believed that computer vision could be solved as a summer project of an MIT graduate student. His original estimation was wrong, and for more than 40 years we have been investigating various aspects of computer vision.

This 40-year effort proved that computer vision is one of the fundamental sciences, and the field is rich enough for researchers to devote their entire research lives to it. This period also reveals that the field contains a wide variety of topics from low-level optics to high-level recognition problems. This richness and diversity were an important motivation for us to decide to compile a reference book on computer vision.

Lawrence Roberts’ research contains all of the essential components of the computer vision paradigm, which modern computer vision still follows: homogeneous coordinate system to define the coordinates, cross operator for edge detection, and object models represented as a combination of edges. David Marr defines his paradigm of computer vision: shape-from-x low-level vision, interpolation and fusion of such fragmental representations, 2-1/2D viewer-centered representation as the result of interpolation and fusion, and 3D object-centered representation. Roughly, this reference guide follows these paradigms, and defines the sections accordingly.

The online version of the reference guide is intended to be developed continuously, both by the updates of existing entries and by the addition of new entries. In this way, it will provide the resources to help both vision researchers and newcomers to the field be on the same page with the continuing and exciting developments in computer vision.

This reference guide has been completed through a team effort. We are most grateful for all the contributors and section editors who have made this project possible. Our special thanks go to Ms. Neha Thapa and other Springer colleagues for their assistance in the development and editing of this reference guide.

March 2014

Katsushi Ikeuchi, Editor in Chief
Yasuyuki Matsushita, Associate Editor in Chief
Rei Kawakami, Assistant Editor in Chief

Editor's Biography



Katsushi Ikeuchi is a professor at the University of Tokyo. He received his B.E. degree in mechanical engineering from Kyoto University in 1973 and Ph.D. in information engineering from the University of Tokyo in 1978. After working at the MIT Artificial Intelligence Laboratory as a postdoctoral fellow for 3 years, the MITI Electro-Technical Laboratory as a research fellow for 5 years, and the CMU Robotics Institute as a faculty member for 10 years, he joined the Institute of Industrial Science, University of Tokyo, as a professor in 1996.

His research activities span computer vision, computer graphics, robotics, and intelligent transportation systems.

In the computer vision area, he is considered as one of the founders of physics-based vision: modeling image forming processes, using physics and optics laws, and applying the inverse models in recovering shape and reflectance from observed brightness in a rigorous manner. He developed the “smoothness constraint,” a constraint to force neighboring points to have similar surface orientations. The constraint optimization method based on the smoothness constraint, later referred to as the regularization method, has evolved into one of the fundamental paradigms, commonly employed in various low-level vision algorithms. In 1992, his paper with Prof. B.K.P. Horn, “Numerical Shape from Shading with Occluding Boundaries,” was the original paper to describe the constraint-minimization algorithm with the smoothness constraint, [Ikeuchi K, Horn BKP (1981) Numerical shape from shading and occluding boundaries. *Artif Intell* 17(1–3):141–184] and was selected as one of the most influential papers to have appeared in the Artificial Intelligence Journal within the past 10 years.

Dr. Ikeuchi and his students developed a technique to automatically generate a virtual reality model by observing actual objects along the line of the physics-based vision paradigm [Sato Y, Wheeler MD, Ikeuchi K (1997) Object shape and reflectance modeling from observation. Computer graphics proceedings, SIGGRAPH97, Los Angeles, pp 379–387]. This early work, which appeared in SIGGRAPH1997, is considered one of the starting points of the area later referred to as “image-based modeling.” After returning to Japan, he and his team began to apply the image-based modeling technique to model various cultural heritage sites. This project has become to be known as the e-Heritage project. They succeeded in modeling all of the three big Buddha statues in Japan as well as the complicated stone temple, Bayon, in Angkor Ruin, to name a few [Ikeuchi K, Miyazaki D (2008) Digitally archiving cultural objects. Springer, New York]. Through these efforts, Dr. Ikeuchi received the IEICE Distinguished Achievement Award in 2012.

Dr. Ikeuchi has also been working on robot vision. In this area, he has been concentrating research on how to reduce the cost of production by using robot vision technologies. This includes how to make efficient production lines and, more importantly, how to reduce the cost of making robot programs to be used in such production lines. In the early 1980s and even today, one of the obstacles to introduce robot technologies to production lines is the so-called bin-picking problem: how to pick up one part from a stack of similar parts. Using shape-from-shading techniques, he was successful in making a robot system that could pick up a mechanical part from a stack [Horn BKP, Ikeuchi K (1984) The mechanical manipulation of randomly oriented parts. *Sci Am* 251(2):100–111].

It was evident that the next obstacle was the cost of programming after completing the bin-picking system. In the early 1990s, he began a project to make a robot program which learns motions from observing human operators’ movements [Ikeuchi K, Suehiro T (1994) Toward an assembly plan from observation, Part I: Task recognition with polyhedral objects. *IEEE Trans Robot Autom* 10(3):368–385]. He and his team demonstrated that this method, programming-by-demonstration, can be applied to handle not only assembling block-world objects, but also machine parts as well as flexible objects, such as rope knotting tasks [Takamatsu J et al (2006) Representation for knot-tying tasks. *IEEE Trans Robot* 22(1):65–78]. Along with his students, he further extended the method in the domain of whole-body motions by a humanoid robot [Nakaoka S et al (2007) Learning from observation paradigm: leg task models for enabling a biped humanoid robot to imitate human dance. *Int J Robot Res* 26(8):829–844]. They succeeded in making a dancing robot, which was capable of learning and mimicking Japanese folk dance from observation. He received several best paper awards in this line of work, including IEEE KS Fu memorial best transaction paper award and RSJ best transaction paper award.

Besides these research activities, he has also devoted his time to community service. He chaired a dozen major conferences, including 1995 IEEE-IROS (General), 1996 IEEE-CVPR (Program), 1999 IEEE-ITSC (General), 2001 IEEE-IV (General), 2003 IEEE-ICCV (Program), 2009 IEEE-ICRA (program), and 2010 IAPR-ICPR (program). His community service also includes IEEE RAS Adcom (98-04, 06-08), IEEE ITSS BOG, IEEE Fellow Committee (2010–2012), and 2nd VP of IAPR. He is an editor in chief of the *International Journal of Computer Vision*, and a fellow of IEEE, IEICE, IPSJ, and RSJ.

Through these research activities and community services, Dr. Ikeuchi received the IEEE PAMI-TC Distinguished Researcher Award (2011) and the Shiju Hou Sho (Medal of Honor with purple ribbon) from the Japanese Emperor (2012).



Yasuyuki Matsushita received his B.S., M.S., and Ph.D. in Electrical Engineering and Computer Science (EECS) from the University of Tokyo in 1998, 2000, and 2003, respectively. He joined Microsoft Research Asia in April 2003, where he is a senior researcher in the Visual Computing Group. His interests are in physics-based computer vision (photometric techniques, such as radiometric calibration, photometric stereo, shape-from-shading), computational photography, and general 3D reconstruction methodologies. He is on the editorial board of *International Journal of Computer Vision* (IJCV), *IEEE Transactions on Pattern Analysis and Machine Intelligence* (TPAMI), *The Visual Computer*, and associate editor in chief of the IPSJ Transactions on Computer Vision and Applications *Journal of Computer Vision and Applications* (CVA). He served/is serving as a program co-chair of Pacific-Rim Symposium on Image and Video Technology (PSIVT) 2010, The first joint 3DIM/3DPVT conference (3DIMPVT, now called 3DV) 2011, Asian Conference on Computer Vision (ACCV) 2012, International Conference on Computer Vision (ICCV) 2017, and a general co-chair of ACCV 2014. He has been serving as a guest associate professor at Osaka University (April 2010–), visiting associate professor at National Institute of Informatics (April 2011–) and Tohoku University (April 2012–), Japan.



Rei Kawakami is an assistant professor at The University of Tokyo, Tokyo, Japan. She received her B.S., M.S., and Ph.D. degrees in information science and technology from the University of Tokyo in 2003, 2005, and 2008, respectively. After working as a researcher at the Institute of Industrial Science, The University of Tokyo, for 3 years, at the University of California, Berkeley, for 2 years, and at Osaka University for a year, she joined the University of Tokyo as an assistant professor in 2014. Her research interests are in color constancy, spectral analysis, and physics-based computer vision. She has been serving as a program committee member and a reviewer for International Conference on Computer Vision (ICCV), International Conference on Computer Vision and Pattern Recognition (CVPR), European Conference on Computer Vision (ECCV), Asian Conference on Computer Vision (ACCV), and as a reviewer of the *International Journal of Computer Vision* (IJCV) and the *IEEE Transactions on Pattern Analysis and Machine Intelligence* (TPAMI).

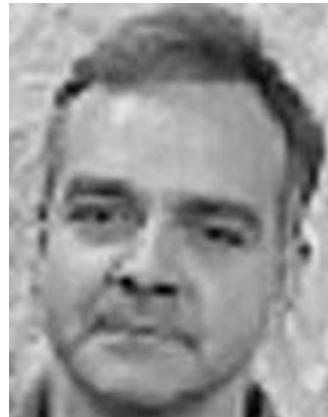
Senior Editors



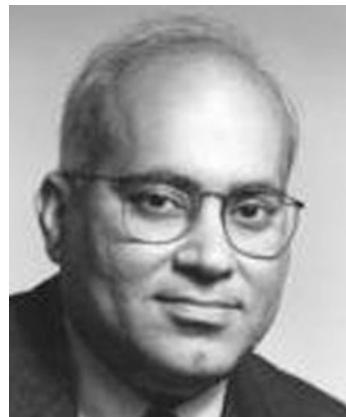
Peter N. Belhumeur Professor, Department of Computer Science, Columbia University, New York, New York, USA



Larry S. Davis Professor, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, MD, USA



Martial Hebert Professor, The Robotics Institute, School of Computer Science,
Carnegie Mellon University, Pittsburgh, PA, USA



Jitendra Malik Arthur J. Chick Professor of EECS, University of California,
Berkeley, CA, USA



Tomas Pajdla Assistant Professor, Center for Machine Perception, Department of
Cybernetics, Faculty of Electrical Engineering, Czech Technical University, Prague,
Czech Republic

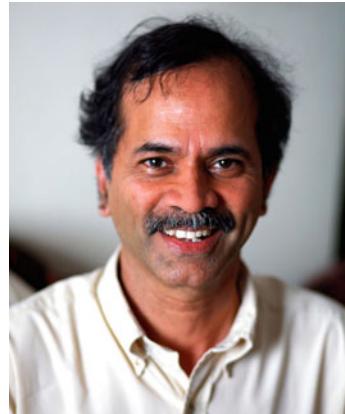


James M. Rehg Professor, School of Interactive Computing, College of Computing,
Georgia Institute of Technology, Atlanta, GA, USA



Zhengyou Zhang Research Manager/Principal Researcher, Microsoft Research,
Redmond, WA, USA

Section Editors



Rama Chellappa Minta Martin Professor of Engineering, Chair, Department of Electrical Engineering and Computer Engineering and UMIACS, University of Maryland, College Park, MD, USA



Daniel Cremers Professor for Computer Science and Mathematics, Chair for Computer Vision and Pattern Recognition, Managing Director, Department of Computer Science, Technische Universität München, Garching, Germany



Koichiro Deguchi Professor Emeritus of Tohoku University, Katahira, Aoba-ku, Sendai, Japan



Hervé Delingette Research Director, Project Asclepios, INRIA, Sophia-Antipolis, France



Andrew Fitzgibbon Principal Researcher, Microsoft Research, Cambridge, England



Luc Van Gool Professor of Computer Vision, Computer Vision Laboratory, ETH, Zürich, Switzerland



Kenichi Kanatani Professor Emeritus of Okayama University, Okayama, Japan



Kyros Kutulakos Professor, Department of Computer Science, University of Toronto, Toronto, ON, Canada



In So Kweon Professor, Robotics and Computer Vision Laboratory, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea



Sang Wook Lee Professor, Department of Media Technology, Sogang University, Seoul, Korea



Atsuto Maki Associate Professor, Computer Vision and Active Perception Laboratory (CVAP), School of Computer Science and Communication, Kungliga Tekniska Högskolan (KTH), Stockholm, Sweden



Yasuyuki Matsushita Senior Researcher, Microsoft Research, Beijing, China



Gerard G. Medioni Professor, Institute for Robotics and Intelligence Systems,
University of Southern California, Los Angeles, CA, USA



Ram Nevatia Director and Professor, Institute for Robotics and Intelligence
Systems, University of Southern California, Los Angeles, CA, USA



Ko Nishino Associate Professor, Department of Computing, College of Computing and Informatics, Drexel University, Philadelphia, PA, USA



Nikolaos Papanikolopoulos Distinguished McKnight University Professor, Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA



Shmuel Peleg Professor, School of Computer Science and Engineering, The Hebrew University of Jerusalem, Jerusalem, Israel



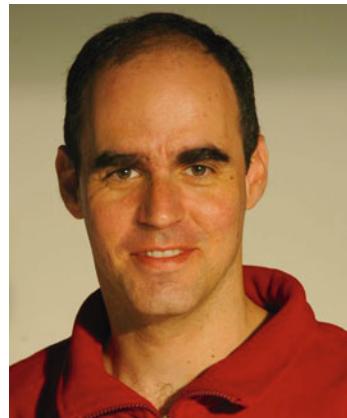
Marc Pollefeys Professor and the Head, Computer Vision and Geometry Lab (CVG) – Institute of Visual Computing, Department of Computer Science, ETH Zürich, Zürich, Switzerland



Jean Ponce Professor, Departement d'Informatique, Ecole Normale Supérieure [ENS], Paris, France



Long Quan Professor, The Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Kowloon, Hong Kong, China



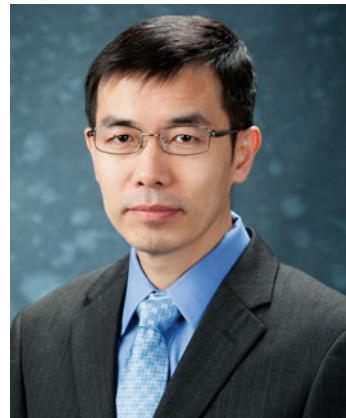
Yoav Y. Schechner Associate Professor, Department of Electrical Engineering,
Technion – Israel Institute of Technology, Haifa, Israel



Cordelia Schmid INRIA Research Director, Head of the LEAR Project,
Montbonnot, France



Jun Takamatsu Associate Professor, Graduate School of Information Science, Nara
Institute of Science and Technology (NAIST), Ikoma, Japan



Xiaou Tang Professor, Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong SAR



Song-Chun Zhu Professor, Statics Department and Computer Science Department, University of California, Los Angeles, CA, USA



Todd Zickler William and Ami Kuan Danoff Professor of Electrical Engineering and Computer Science, School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA

Contributors

Hanno Ackermann Leibniz University Hannover, Hannover, Germany

J. K. Aggarwal Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX, USA

Amit Agrawal Mitsubishi Electric Research Laboratories, Cambridge, MA, USA

Daniel C. Alexander Centre for Medical Image Computing, Department of Computer Science, University College London, London, UK

Marina Alterman Department of Electrical Engineering, Technion – Israel Institute of Technology, Haifa, Israel

Yali Amit Department of Computer Science, University of Chicago, Chicago, IL, USA

Gary A. Atkinson Machine Vision Laboratory, University of the West of England, Bristol, UK

Ruzena Bajcsy Department of Electrical and Computer Sciences, College of Engineering University of California, Berkeley, CA, USA

Simon Baker Microsoft Research, Redmond, WA, USA

Dana H. Ballard Department of Computer Sciences, University of Texas at Austin, Austin, TX, USA

Richard G. Baraniuk Department of Electrical and Computer Engineering, Rice University 2028 Duncan Hal, Houston, TX, USA

Adrian Barbu Department of Statistics, Florida State University, Tallahassee, FL, USA

Nick Barnes Australian National University, Canberra, Australia

Ronen Basri Department of Computer Science And Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel

Anup Basu Department of Computing Science, University of Alberta, Edmonton, AB, Canada

Rodrigo Benenson K.U. Leuven departement Elektrotechniek – ESAT, Centrum voor beeld- en spraakverwerking – PSI/VISICS, Heverlee, Belgium

Marcelo Bertalmío Universitat Pompeu Fabra, Barcelona, Spain

Jürgen Beyerer Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IOSB, Karlsruhe, Germany

Irving Biederman Department of Computer Science, University of Toronto, Toronto, ON, Canada

Departments of Psychology, Computer Science, and the Neuroscience Program, University of Southern California, Los Angeles, CA, USA

Tom Bishop Image Algorithms Engineer, Apple, Cupertino, CA, USA

James Bonaiuto California Institute of Technology, Pasadena, CA, USA

Yuri Boykov Department of Computer Science, University of Western Ontario, London, ON, Canada

Christoph Bregler Courant Institute, New York University, New York, NY, USA

Michael H. Brill Datacolor, Lawrenceville, NJ, USA

Matthew Brown Dept of Computer Science, University of Bath, Bath, UK

Thomas Brox Department of Computer Science, University of Freiburg, Freiburg, Germany

Frank Michael Caimi IEEE OES, Vero Beach, FL, USA

Fabio Camilli Dipartimento SBAI, “Sapienza”, Università di Roma, Rome, Italy

John N. Carter School of Electronics and Computer Science, University of Southampton, Southampton, Hampshire, UK

Vicent Caselles Universitat Pompeu Fabra, Barcelona, Spain

Shing Chow Chan Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong, China

Manmohan Chandraker NEC Labs America, Cupertino, CA, USA

Visesh Chari Institut National de Recherche en Informatique et en Automatique (INRIA), Le Chesnay Cedex, France

François Chaumette Inria, Rennes, France

Rama Chellappa Department of Electrical Engineering and Computer Engineering and UMIACS, University of Maryland, College Park, MD, USA

Liang Chen Department of Computer Science, University of Northern British Columbia, Prince George, Canada

Irene Cheng University of Alberta, Edmonton, AB, Canada

James J. Clark Department of Electrical and Computer Engineering, McGill University, Montreal, QC, Canada

Michael F. Cohen Microsoft Research, One Microsoft Way, Redmond, WA, USA

Kristin Dana Department of Electrical and Computer Engineering, Rutgers University, The State University of New Jersey, Piscataway, NJ, USA

Larry S. Davis Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, MD, USA

Fatih Demirci Department of Computer Engineering, TOBB University of Economics and Technology, Sogutozu, Ankara, Turkey

Sven J. Dickinson Department of Computer Science, University of Toronto, Toronto, ON, Canada

Leo Dorst Intelligent Systems Laboratory Amsterdam, Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands

Mark S. Drew School of Computing Science, Simon Fraser University, Vancouver, BC, Canada

Marc Ebner Institut für Mathematik und Informatik, Ernst Moritz Arndt Universität Greifswald, Greifswald, Germany

Jan-Olof Eklundh School of Computer Science and Communication, KTH - Royal Institute of Technology, Stockholm, Sweden

James H. Elde Centre for Vision Research, York University, Toronto, ON, Canada

Francisco J. Estrada Department of Computer and Mathematical Sciences, University of Toronto at Scarborough, Toronto, ON, Canada

Paolo Favaro Department of Computer Science and Applied Mathematics, Universität Bern, Switzerland

Pedro Felzenszwalb School of Engineering, Brown University, Providence, RI, USA

Robert B. Fisher School of Informatics, University of Edinburgh, Edinburgh, UK

Boris Flach Department of Cybernetics, Czech Technical University in Prague, Faculty of Electrical Engineering, Prague 6, Czech Republic

Gian Luca Foresti Department of Mathematics and Computer Science, University of Udine, Udine, Italy

Wolfgang Förstner ETH Zürich, Zürich, Switzerland

Universität Bonn, Bonn, Germany

Kazuhiro Fukui Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Japan

Yasutaka Furukawa Google Inc., Seattle, WA, USA

Juergen Gall Max Planck Institute for Intelligent Systems, Tübingen, Germany

David Gallup Google Inc., Seattle, WA, USA

Abhijeet Ghosh Institute for Creative Technologies, University of Southern California, Playa Vista, CA, USA

Michael Goesele GCC - Graphics, Capture and Massively Parallel Computing, TU Darmstadt, Darmstadt, Germany

Bastian Goldluecke Department of Computer Science, Technische Universität München, München, Germany

Gaopeng Gou Beihang University, Beijing, China

Mohit Gupta Department of Computer Science, Columbia University, New York, NY, USA

Bohyung Han Department of Computer Science and Engineering, Pohang University of Science and Technology (POSTECH), Pohang, South Korea

Richard Hartley Department of Engineering, Australian National University, ACT, Australia

Samuel W. Hasinoff Google, Inc., Mountain View, CA, USA

Nils Hasler Graphics, Vision & Video, MPI Informatik, Saarbrücken, Germany

Vaclav Hlavac Department of Cybernetics, Czech Technical University in Prague, Faculty of Electrical Engineering, Prague 6, Czech Republic

Andrew Hogue Faculty of Business and Information Technology, University of Ontario Institute of Technology, Oshawa, ON, Canada

Takahiko Horiuchi Graduate School of Advanced Integration Science, Chiba University, Inage-ku, Chiba, Japan

Berthold K. P. Horn Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, USA

Zhanyi Hu National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China

Ivo Ihrke MPI Informatik, Saarland University, Saarbrücken, Germany

Michael R. M. Jenkin Department of Computer Science and Engineering, York University, Toronto, ON, Canada

Jiaya Jia Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong, China

Zhuolin Jiang Noah's Ark Lab, Huawei Tech. Investment Co., LTD., Shatin, Hong Kong, China

Micah K. Johnson Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA

Neel Joshi Microsoft Corporation, Redmond, WA, USA

Avinash C. Kak School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA

Kenichi Kanatani Professor Emeritus of Okayama University, Okayama, Japan

Sing Bing Kang Microsoft Research, Redmond, WA, USA

Peter Karasev Schools of Electrical and Computer and Biomedical Engineering, Georgia Institute of Technology, Atlanta, GA, USA

- Jun-Sik Kim** Korea Institute of Science and Technology, Seoul, Republic of Korea
- Ron Kimmel** Department of Computer Science, Technion – Israel Institute of Technology, Haifa, Israel
- Eric Klassen** Ohio State University, Columbus, OH, USA
- Reinhard Koch** Institut für Informatik Christian-Albrechts-Universität, Kiel, Germany
- Jan J. Koenderink** Faculty of EEMSC, Delft University of Technology, Delft, The Netherlands
- The Flemish Academic Centre for Science and the Arts (VLAC), Brussels, Belgium
- Laboratory of Experimental Psychology, University of Leuven (K.U. Leuven), Leuven, Belgium
- Pushmeet Kohli** Department of Computer Science And Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel
- Ivan Kolesov** Schools of Electrical and Computer and Biomedical Engineering, Georgia Institute of Technology, Atlanta, GA, USA
- Sanjeev J. Koppal** Harvard University, Cambridge, MA, USA
- Kevin Köser** Institute for Visual Computing, ETH Zurich, Zürich, Switzerland
- Sanjiv Kumar** Google Research, New York, NY, USA
- Takio Kurita** Graduate School of Engineering, Hiroshima University, Higashi-Hiroshima, Japan
- Sebastian Kurtek** Ohio State University, Columbus, OH, USA
- Annika Lang** Seminar für Angewandte Mathematik, ETH Zürich, Zürich, Switzerland
- Michael S. Langer** School of Computer Science, McGill University, Montreal, QC, Canada
- Fabian Langguth** GCC - Graphics, Capture and Massively Parallel Computing, TU Darmstadt, Darmstadt, Germany
- Longin Jan Latecki** Department of Computer and Information Sciences, Temple University, Philadelphia, PA, USA
- Denis Laurendeau** Faculty of Science and Engineering, Department of Electrical and Computer Engineering, Laval University, QC, Canada
- Jason Lawrence** Department of Computer Science, School of Engineering and Applied Science University of Virginia, Charlottesville, VA, USA
- Svetlana Lazebnik** Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA
- Longzhuang Li** Department of Computing Science, Texas A and M University at Corpus Christi, Corpus Christi, TX, USA

Wanqing Li University of Wollongong, Wollongong, NSW, Australia

Stephen Lin Microsoft Research Asia, Beijing Sigma Center, Beijing, China

Zhe Lin Advanced Technology Labs, Adobe Systems Incorporated, San Jose, CA, USA

Tony Lindeberg School of Computer Science and Communication, KTH Royal Institute of Technology, Stockholm, Sweden

Yanxi Liu Computer Science and Engineering, Penn State University, University Park, PA, USA

Yonghuai Liu Department of Computer Science, Aberystwyth University, Ceredigion, Wales, UK

Zhi-Yong Liu Institute of Automation, Chinese Academy of Sciences, Beijing, P. R. China

Zicheng Liu Microsoft Research, Microsoft Corporation, Redmond, WA, USA

Songde Ma Ministry of Science & Technology, Beijing, China

Eisaku Maeda NTT Communication Science Laboratories, Soraku-gun, Kyoto, Japan

Michael Maire California Institute of Technology, Pasadena, CA, USA

Pascal Mamassian Laboratoire Psychologie de la Perception, Université Paris Descartes, Paris, France

Ralph R. Martin School of Computer Science and Informatics Cardiff University, Cardiff, UK

Simon Masnou Institut Camille Jordan, Université Lyon 1, Villeurbanne, France

Darko S. Matovski School of Electronics and Computer Science, University of Southampton, Southampton, Hampshire, UK

Yasuyuki Matsushita Microsoft Research, Beijing, China

Larry Matthies Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA

Stephen J. Maybank Department of Computer Science and Information Systems, Birkbeck College University of London, London, UK

Peter Meer Department of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ, USA

Christian Micheloni Department of Mathematics and Computer Science, University of Udine, Udine, Italy

Sushil Mittal Department of Statistics, Columbia University, New York, NY, USA

Daisuke Miyazaki Graduate School of Information Sciences, Hiroshima City University, Asaminami-ku, Hiroshima, Japan

Vlad I. Morariu Computer Vision Laboratory, University of Maryland, College Park, MD, USA

Joseph L. Mundy Division of Engineering, Brown University Rensselaer Polytechnic Institute, Providence, RI, USA

Hiroshi Murase Faculty of Economics and Information, Gifu Shotoku Gakuen University, Gifu, Japan

Kazuo Murota Department of Mathematical Informatics, University of Tokyo, Bunkyo-ku, Tokyo, Japan

Bernd Neumann University of Hamburg, Hamburg, Germany

Mark S. Nixon School of Electronics and Computer Science, University of Southampton, Southampton, Hampshire, UK

Takayuki Okatani Graduate School of Information Sciences, Tohoku University, Sendai-shi, Japan

Vasu Parameswaran Microsoft Corporation, Sunnyvale, CA, USA

Johnny Park School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA

Samunda Perera Canberra Research Laboratory, NICTA, Canberra, Australia

Matti Pietikäinen Department of Computer Science and Engineering, University of Oulu, Oulu, Finland

Tomaso Poggio Department of Brain and Cognitive Sciences, McGovern Institute, Massachusetts Institute of Technology, Cambridge, MA, USA

Marc Pollefeys Computer Vision and Geometry Lab (CVG) – Institute of Visual Computing, Department of Computer Science, ETH Zürich, Zürich, Switzerland

S. C. Pont Industrial Design Engineering, Delft University of Technology, Delft, The Netherlands

Emmanuel Prados INRIA Rhône-Alpes, Montbonnot, France

Jerry L. Prince Electrical and Computer Engineering, Johns Hopkins University, Baltimore, MD, USA

Srikumar Ramalingam Mitsubishi Electric Research Laboratories, Cambridge, MA, USA

Rajeev Ramanath DLP® Products, Texas Instruments Incorporated, Plano, TX, USA

Dikpal Reddy Nvidia Research, Santa Clara, CA, USA

Xiaofeng Ren Intel Science and Technology Center for Pervasive Computing, Intel Labs, Seattle, WA, USA

Xuejun Ren School of Engineering Liverpool John Moores University, Liverpool, UK

Szymon Rusinkiewicz Department of Computer Science, Princeton University, Princeton, NJ, USA

Aswin C. Sankaranarayanan ECE Department, Rice University, Houston, TX, USA

Guillermo Sapiro Electrical and Computer Engineering, Computer Science, and Biomedical Engineering, Duke University, Durham, NC, USA

Silvio Savarese Department of Electrical and Computer Engineering, University of Michigan, Ann Arbor, MI, USA

Davide Scaramuzza Artificial Intelligence Lab - Robotics and Perception Group, Department of Informatics, University of Zurich, Zurich, Switzerland

Konrad Schindler ETH Zürich, Zürich, Switzerland

David C. Schneider Image Processing Department, Fraunhofer Heinrich Hertz Institute, Berlin, Germany

William Robson Schwartz Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brazil

Guna Seetharaman Air Force Research Lab RITB, Rome, NY, USA

Chunhua Shen School of Computer Science , The University of Adelaide, Adelaide, SA, Australia

Rui Shen University of Alberta, Edmonton, AB, Canada

Ali Shokoufandeh Department of Computer Science, Drexel University Philadelphia, PA, USA

Jamie Shotton Microsoft Research Ltd, Cambridge, UK

Kaleem Siddiqi School of Computer Science, McGill University, Montreal, PQ, Canada

Leonid Sigal Disney Research, Pittsburgh, PA, USA

Manish Singh Department of Psychology, Rutgers University, Piscataway, NJ, USA

Sudipta N. Sinha Microsoft Research, Redmond, WA, USA

Arnold W. M. Smeulders Centre for Mathematics and Computer Science (CWI), University of Amsterdam, Amsterdam, The Netherlands

Intelligent Systems Lab Amsterdam, Informatics Institute University of Amsterdam, Amsterdam, The Netherlands

Cees G. M. Snoek University of Amsterdam, Amsterdam, The Netherlands

Intelligent Systems Lab Amsterdam, Informatics Institute University of Amsterdam, Amsterdam, The Netherlands

Gunnar Sparr Centre for Mathematical Sciences, Lund University, Lund, Sweden

Anuj Srivastava Florida State University, Tallahassee, FL, USA

Gaurav Srivastava School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA

Peer Stelldinger Computer Science Department, University of Hamburg, Hamburg, Germany

Peter Sturm INRIA Grenoble Rhône-Alpes, St Ismier Cedex, France

Kokichi Sugihara Graduate School of Advanced Mathematical Sciences, Meiji University, Kawasaki, Kanagawa, Japan

Min Sun Department of Electrical and Computer Engineering, University of Michigan, Ann Arbor, MI, USA

Richard Szeliski Microsoft Research, One Microsoft Way, Redmond, WA, USA

Yu-Wing Tai Department of Computer Science, Korean Advanced Institute of Science and Technology (KAIST), Yuseong-gu, Daejeon, South Korea

Tomokazu Takahashi Faculty of Economics and Information, Gifu Shotoku Gakuen University, Gifu, Japan

Birgi Tumer soy Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX, USA

Ping Tan Department of Electrical and Computer Engineering, National University of Singapore, Singapore, Singapore

Robby T. Tan Department of Information and Computing Sciences, Utrecht University, Utrecht, CH, The Netherlands

Allen Tannenbaum Schools of Electrical and Computer and Biomedical Engineering, Georgia Institute of Technology, Atlanta, GA, USA

Marshall F. Tappen University of Central Florida, Orlando, FL, USA

Federico Tombari DEIS, University of Bologna, Bologna, Italy

Shoji Tominaga Graduate School of Advanced Integration Science, Chiba University, Inage-ku, Chiba, Japan

Lorenzo Torresani Computer Science Department, Dartmouth College, Hanover, NH, USA

Tali Treibitz Department of Computer Science and Engineering, University of California, San Diego, La Jolla, CA, USA

Yanghai Tsin Corporate R&D, Qualcomm Inc., San Diego, CA, USA

Pavan Turaga Department of Electrical and Computer Engineering, Center for Automation Research University of Maryland, College Park, MD, USA

Matthew Turk Computer Science Department and Media Arts and Technology Graduate Program, University of California, Santa Barbara, CA, USA

Tinne Tuytelaars KULEUVEN Leuven, ESAT-PSI, iMinds, Leuven, Belgium

Shimon Ullman Department of Brain and Cognitive Sciences, McGovern Institute, Massachusetts Institute of Technology, Cambridge, MA, USA

Anton van den Hengel School of Computer Science, The University of Adelaide, Adelaide, SA, Australia

Pramod K. Varshney Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, NY, USA

Andrea Vedaldi Oxford University, Oxford, UK

Ashok Veeraraghavan Department of Electrical and Computer Engineering, Rice University, Houston, TX, USA

David Vernon Informatics Research Centre, University of Skövde, Skövde, Sweden

Ramanarayanan Viswanathan Department of Electrical Engineering, University of Mississippi, MS, USA

Xiaogang Wang Department of Electronic Engineering, Chinese University of Hong Kong, Shatin, Hong Kong

Isaac Weiss Center for Automation Research, University of Maryland at College Park, College Park, MD, USA

Gregory F. Welch Institute for Simulation & Training, The University of Central Florida, Orlando, FL, USA

Michael Werman The Institute of Computer Science, The Hebrew University of Jerusalem, Jerusalem, Israel

Tien-Tsin Wong Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong SAR, China

Robert J. Woodham Department of Computer Science, University of British Columbia, Vancouver, BC, Canada

John Wright Visual Computing Group, Microsoft Research Asia, Beijing, China

Ying Nian Wu Department of Statistics, UCLA, Los Angeles, CA, USA

Chenyang Xu Siemens Technology-To-Business Center, Berkeley, CA, USA

David Young School of Informatics, University of Sussex, Falmer, Brighton, UK

Guoshen Yu Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN, USA

Christopher Zach Computer Vision and Geometry Group, ETH Zürich, Zürich, Switzerland

Alexander Zelinsky CSIRO, Information Sciences, Canberra, Australia

Zhengyou Zhang Microsoft Research, Redmond, WA, USA

Bo Zheng Computer Vision Laboratory, Institute of Industrial Science, The University of Tokyo, Meguro-ku, Tokyo, Japan

Zhigang Zhu Computer Science Department, The City College of New York, New York, NY, USA

Todd Zickler School of Engineering and Applied Science, Harvard University, Cambridge, MA, USA

A

Action Prototype Trees

- ▶ Prototype-Based Methods for Human Movement Modeling

Action Recognition

- ▶ Activity Recognition
- ▶ Affordances and Action Recognition

Active Calibration

Rui Shen¹, Gaopeng Gou², Irene Cheng¹ and Anup Basu³

¹University of Alberta, Edmonton, AB, Canada

²Beihang University, Beijing, China

³Department of Computing Science, University of Alberta, Edmonton, AB, Canada

Synonyms

Active camera calibration; Pan-tilt camera calibration; Pan-tilt-zoom camera calibration; PTZ camera calibration

Related Concepts

- ▶ Camera Calibration

Definition

Active calibration is a process that determines the geometric parameters of a camera (or cameras) using the camera's controllable movements.

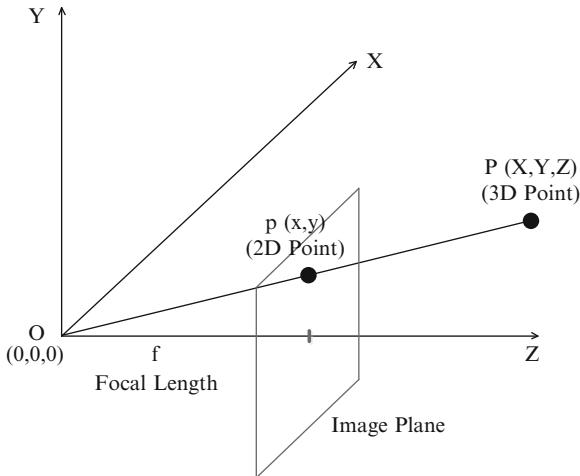
Background

Camera calibration aims to establish the best possible correspondence between the used camera model and the realized image acquisition with a given camera [12], i.e., accurately recover a camera's geometric parameters, such as focal length and image center/principal point, from the captured images. The classical calibration techniques (e.g., [10, 16]) require pre-defined patterns and static cameras and often involve solving complicated equations. Taking advantage of a camera's controllable movements (e.g., pan, tilt, and roll), active calibration techniques can automatically calibrate the camera.

Theory

The pinhole camera model is one of the most commonly used models, as shown in Fig. 1. $\mathbf{p} = (x, y)^T$ is the 2D projection of the 3D point $\mathbf{P} = (X, Y, Z)^T$ on the image plane. Using homogeneous coordinates, $\tilde{\mathbf{p}}$ and $\tilde{\mathbf{P}}$ have the following relationship:

$$\lambda \tilde{\mathbf{p}} = \mathbf{K}(\mathbf{R} \mid \mathbf{t}) \tilde{\mathbf{P}} \quad (1)$$



Active Calibration, Fig. 1 The pinhole camera model

where $\mathbf{K} = \begin{pmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{pmatrix}$ is the camera calibration matrix; λ is the depth (when \mathbf{R} is the identity matrix and \mathbf{t} is a zero-vector, $\lambda = Z$); and \mathbf{R} and \mathbf{t} are the 3×3 rotation matrix and the 3×1 translation vector, respectively. f_x and f_y are the focal lengths in pixels in the x - and y -directions, respectively. They are proportional to the focal length f shown in Fig. 1, which is normally measured in millimeters. (x_0, y_0) are the coordinates of the image center/principal point on the image plane, i.e., the intersection of the lens' optical axis and the image plane. s is the skew. Normally, the sensor element is assumed rectangular; therefore, s becomes 0. $\gamma = f_x/f_y$ is the aspect ratio. If the sensor element is quadratic, γ becomes 1. The five parameters in \mathbf{K} are called the camera intrinsic parameters; the three Euler angles in \mathbf{R} and the three offsets in \mathbf{t} are called the camera extrinsic parameters.

Figure 2 shows an active camera. When a camera rotates (no translation), the new image points can be obtained as transformations of the original image points [11]. The camera parameters can be related to the image points before and after rotation and to the angles of rotation. The equations describing these relations are simple and easy to solve. Thus, by considering different movements of the camera (pan, tilt, and roll with measurement of the angles), the intrinsic parameters of a camera can be estimated.

Basu [1] proposed an active calibration technique utilizing the edge information of a static scene.

This technique was extended in [2–4] to give more robust estimation of the intrinsic parameters. No special patterns are required, but the observed scene should have strong and stable edges. Four strategies are introduced and validated through experiments [4]. Strategies A and B only utilize pan and tilt movements; strategy C utilizes pan, tilt, and roll movements; strategy D is a special case of strategy C when the roll angle is equal to 180° . The procedure of strategy C is outlined as follows:

- Using the pan and tilt movement of the camera and a single image contour, obtain the values of f_x and f_y by solving Eqs. 2 and 3.
- Using the roll movement of the camera and a single image contour, obtain the values of δ_x and δ_y by Eqs. 4 and 5.

$$f_y = \frac{\bar{y}_t - \bar{y}(1 + \theta_t^2)}{2\theta_t} + \frac{1}{2} \sqrt{\left(\frac{\bar{y}(1 + \theta_t^2) - \bar{y}_t^2}{\theta_t}\right)^2 - 4\bar{y}^2} \quad (2)$$

$$f_x = \frac{\bar{x}_p - \bar{x}(1 + \theta_p^2)}{2\theta_p} + \frac{1}{2} \sqrt{\left(\frac{\bar{x}(1 + \theta_p^2) - \bar{x}_p^2}{\theta_p}\right)^2 - 4\bar{x}^2} \quad (3)$$

where (x_p, y_p) and (x_t, y_t) denote image coordinates after pan and tilt movements, respectively; θ_t and θ_p are the tilt angle and pan angle, respectively.

$$\begin{aligned} \delta_x(1 - \cos(\theta_r)) + \delta_y \left(-\frac{f_x}{f_y} \sin(\theta_r) \right) \\ = \bar{x}_r - \cos(\theta_r) \bar{x} - \frac{f_x}{f_y} \sin(\theta_r) \bar{y} \end{aligned} \quad (4)$$

$$\begin{aligned} \delta_x \left(-\frac{f_x}{f_y} \sin(\theta_r) \right) + \delta_y(1 - \cos(\theta_r)) \\ = \bar{y}_r - \cos(\theta_r) \bar{y} + \frac{f_x}{f_y} \sin(\theta_r) \bar{x} \end{aligned} \quad (5)$$

where (x_r, y_r) denotes image coordinates after roll movement; (δ_x, δ_y) denotes the error in the estimated principal point (e.g., taking the geometric center of the image plane as the estimated principal point); and θ_r is the roll angle.

Davis and Chen [9] introduced a new pan-tilt camera motion model, in which the pan and tilt axes are not necessarily orthogonal or aligned to the image plane. A tracked object is used to form a large virtual calibration object that covers the whole working volume. A set of pre-calibrated static cameras is needed to record the trajectory. The intrinsic parameters are recovered by minimizing the projection errors between the observed 2D data and the calculated 2D locations of the tracked object using the proposed camera model.

McLauchlan and Murray [13] applied the variable state-dimension filter to calibrating a single camera mounted on a robot by tracking the trajectories of an arbitrary number of tracked corner features and utilizing accurate knowledge of the camera rotation. The camera's intrinsic parameters are updated in real time.

Different zoom settings (focal lengths) can also be employed in active calibration. Seales and Eggert [14] calibrate a camera via a fully automated 4-stage global optimization process using a sequence of images of a known calibration target obtained at different mechanical zoom settings. Collins and Tsin [8] proposed a parametric camera model and calibration procedures for an outdoor active camera system with pan, tilt, and zoom control. Intrinsic parameters are recovered by fitting the camera model with the optic flow produced by the camera's movements. Extrinsic parameters are estimated as a pose estimation problem using sparsely

deployed landmarks. Borghese et al. [5] proposed a technique to compute camera focal lengths by zooming a single point, assuming the principal point is in a fixed and known position. Sinha and Pollefeys [15] propose a camera model that incorporates the variation of radial distortion with camera zoom. The intrinsic parameters are first computed at the lowest zoom level from a captured panorama. Then, the intrinsic and radial distortion parameters are estimated at sequentially increased zoom levels, taking into account the influence of camera zoom.

Application

Active calibration has been receiving more and more attention with the increasing use of active systems in various applications, such as object tracking, surveillance, and video conference. More generally, camera calibration is important for any application that involves relating a 2D image to the 3D world. Such applications include pose estimation, 3D motion estimation, automated assembly, close-range photogrammetry, and so on.

Open Problems

Recent research is more focused on automatic active calibration of a multi-camera system without using a predefined calibration pattern/object. Chippendale and Tobia [7] presented an autocalibration system for the estimation of extrinsic parameters of active cameras in indoor environments. One constraint of the camera deployment is that each camera must be able to observe at least one other camera to form an observation chain. The extrinsic parameters are estimated using the circular shape of the camera lenses and a predetermined moving pattern of a particular camera. The accuracy of the algorithm is mainly affected by the distance between cameras. Brückner and Denzler [6] proposed a three-step multi-camera calibration algorithm. The extrinsic parameters of each camera are first roughly estimated using a probability distribution based on the captured images. Then, each camera pair rotates and zooms in a way that maximizes image similarity, and the extrinsic parameters are reestimated based on point correspondence. A final calibration is carried out using the probabilities and the reestimated

Active Calibration, Table 1 Results of image center estimation

| Angle (strategy C) | Ground truth | | $\sigma = 0$ | | $\sigma = 5$ | | $\sigma = 10$ | |
|-----------------------|--------------|------------|--------------|------------|--------------|------------|---------------|------------|
| | δ_x | δ_y | δ_x | δ_y | δ_x | δ_y | δ_x | δ_y |
| 20° | 10 | 20 | 11 | 22 | 12 | 23 | 12 | 23 |
| 40° | 10 | 20 | 11 | 21 | 12 | 22 | 12 | 19 |
| 60° | 10 | 20 | 11 | 21 | 12 | 21 | 13 | 21 |
| 80° | 10 | 20 | 11 | 21 | 11 | 21 | 11 | 22 |
| 100° | 10 | 20 | 10 | 21 | 10 | 21 | 11 | 21 |
| 120° | 10 | 20 | 11 | 21 | 11 | 21 | 11 | 21 |
| 140° | 10 | 20 | 11 | 21 | 11 | 21 | 11 | 21 |
| 160° | 10 | 20 | 10 | 21 | 11 | 21 | 11 | 21 |
| 180° | 10 | 20 | 10 | 20 | 10 | 21 | 10 | 21 |
| Strategy D | 10 | 20 | 10 | 20 | 11 | 20 | 11 | 20 |

Active Calibration, Table 2 Results of focal length estimation

| Strategy | Ground truth | | $\sigma = 0$ | | $\sigma = 5$ | |
|------------|--------------|-------|--------------|-------|--------------|-------|
| | f_x | f_y | f_x | f_y | f_x | f_y |
| Strategy C | 400 | 600 | 403 | 602 | 396 | 603 |
| Strategy D | 400 | 600 | 401 | 601 | 403 | 599 |

extrinsic parameters. This method achieves relatively high accuracy and robustness, but one drawback is the high computational cost.

Experimental Results

Some experimental results using Strategies C and D from [4] are presented below.

Tables 1 and 2 summarize the results of computing the image center and focal lengths using simulated data, along with the ground truths. The simulated data contain an image contour consisting of 50 points. The pan and tilt angles were fixed at 3°. For the experiment on image center calculation, additive Gaussian noise with standard deviation σ of 0 (no noise), 5, and 10 pixels were added to test the robustness of the algorithms. It can be seen that strategy C performs reasonably in determining the image center even when σ is as large as 15. The results of strategy D are similar to those produced by strategy C when the roll angle is 180°.

For the experiment on focal length calculation, additive Gaussian noise with standard deviation σ of 0 (no noise) and 5 pixels were added. Strategy D is a little more accurate as the equations are obtained directly without using the estimates of f_x and f_y . The focal lengths obtained by strategy D are similar to those produced by strategy C. But when the noise

is increased, strategy D produces more reliable results than strategy C.

Strategies C and D were also tested on a real camera in an indoor environment. The estimates for δ_x and δ_y obtained by strategy C (90° roll) were 3 and 29 pixels, while the values obtained by strategy D were 2 and 30 pixels, which demonstrates the stability of the active calibration algorithms in real situations. The estimated values of f_x and f_y were 908 and 1,126, respectively.

References

- Basu A (1993) Active calibration. In: ICRA'93: proceedings of the 1993 IEEE international conference on robotics and automation, Atlanta, vol 2, pp 764–769
- Basu A (1993) Active calibration: alternative strategy and analysis. In: CVPR'93: proceedings of the 1993 IEEE computer society conference on computer vision and pattern recognition (CVPR), New York, pp 495–500
- Basu A (1995) Active calibration of cameras: theory and implementation. IEEE Trans Syst Man Cybern 25(2): 256–265
- Basu A, Ravi K (1997) Active camera calibration using pan, tilt and roll. IEEE Trans Syst Man Cybern B 27(3):559–566
- Borghese NA, Colombo FM, Alzati A (2006) Computing camera focal length by zooming a single point. Pattern Recognit 39(8):1522–1529
- Brückner M, Denzler J (2010) Active self-calibration of multi-camera systems. In: Proceedings of the 32nd DAGM conference on pattern recognition, Darmstadt, pp 31–40
- Chippendale P, Tobia F (2005) Collective calibration of active camera groups. In: AVSS'05: proceedings of the IEEE conference on advanced video and signal based surveillance, Como, pp 456–461
- Collins RT, Tsin Y (1999) Calibration of an outdoor active camera system. In: CVPR'99: proceedings of the 1999 IEEE computer society conference on computer vision and pattern recognition (CVPR), Ft. Collins, pp 528–534
- Davis J, Chen X (2003) Calibrating pan-tilt cameras in wide-area surveillance networks. In: ICCV'03: proceedings of the 9th IEEE international conference on computer vision, Nice, pp 144–149
- Horau R, Mohr R, Lorecki B (1992) Linear camera calibration. In: ICRA'92: proceedings of the IEEE international conference on robotics and automation, Nice, vol 2, pp 1539–1544
- Kanatani K (1987) Camera rotation invariance of image characteristics. Comput Vis Graph Image Process 39(3):328–354
- Klette R, Schlüns K, Koschan A (1998) Computer vision: three-dimensional data from images, 1st edn. Springer, New York/Singapore
- McLauchlan PF, Murray DW (1996) Active camera calibration for a head-eye platform using the variable state-dimension filter. IEEE Trans Pattern Anal Mach Intell 18(1):15–22

14. Seales WB, Eggert DW (1995) Active-camera calibration using iterative image feature localization. In: CAIP'95: proceedings of the 6th international conference on computer analysis of images and patterns, Prague, pp 723–728
15. Sinha SN, Pollefeys M (2006) Pan-tilt-zoom camera calibration and high-resolution mosaic generation. *Comput Vis Image Underst* 103(3):170–183
16. Tsai R (1987) A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE J Robot Autom* 3(4): 323–344

Active Camera Calibration

► [Active Calibration](#)

Active Contours

► [Numerical Methods in Curve Evolution Theory](#)

Active Sensor (Eye) Movement Control

James J. Clark
Department of Electrical and Computer Engineering,
McGill University, Montreal, QC, Canada

Synonyms

[Gaze control](#)

Related Concepts

► [Evolution of Robotic Heads](#); ► [Visual Servoing](#)

Definition

Active sensors are those whose generalized viewpoint (such as sensor aperture, position, and orientation) is under computer control. Control is done so as to improve information gathering and processing.

Background

The *generalized viewpoint* [1] of a sensor is the vector of values of the parameters that are under the control of the observer and which affect the imaging process. Most often, these parameters will be the position and orientation of the image sensor, but may also include such parameters as the focal length, aperture width, and the nodal point to image plane distance, of the camera. The definition of generalized viewpoint can be extended to include illuminant degrees of freedom, such as the illuminant position, wavelength, intensity, spatial distribution (for structured light applications), and angular distribution (e.g., collimation) [2].

Changes in observer viewpoint are used in active vision systems for a number of purposes. Some of the more important uses are:

- Tracking a moving object to keep it in the field of view of the sensor
- Searching for a specific item in the observer's environment
- Inducing scene-dependent optical flow to aid in the extraction of 3D structure of objects and scenes
- Avoiding “accidental” or nongeneric viewpoints, which can result in sensor-saturating specularities or information-hiding occlusions
- Minimizing sensor noise and maximizing novel information content
- Increasing the dynamic range of the sensor, through adjustment of parameters such as sensor sensitivity, aperture, and focus
- Mapping the observer's environment

Theory

LowLevel Camera Motion Control Systems Most robotic active vision control systems act mainly to produce either smooth pursuit motions or rapid saccadic motions. Pursuit motions cause the camera to move so as to smoothly track a moving object, maintaining the image of the target object within a small region (usually in the center) of the image frame. Saccadic motions are rapid, usually large, jumps in the position of the camera, which center the camera field of view on different parts of the scene being imaged. This type of motion is used when scanning a scene,

searching for objects or information, but can also be used to recover from a loss of tracking of an object during visual pursuit.

Much has been learned about the design of pursuit and saccadic motion control systems from the study of primate oculomotor systems. These systems have a rather complicated architecture distributed among many brain areas, the details of which are still subject to vigorous debate [3]. The high-level structure, however, is generally accepted to be that of a feedback system. A very influential model of the human oculomotor control system is that of Robinson [4], and many robotic vision control systems employ aspects of the Robinson model.

The control of an active camera system is both simple and difficult at the same time. Simplicity arises from the relatively unchanging characteristics of the load or “plant” being controlled. For most systems the moment of inertia of the camera changes only minimally over the range of motion, with slight variations arising when zoom lenses are used. The mass of the camera and associated linkages does not change. Inertial effects become more important for control of the “neck” degrees of freedom due to the changing orientation and position of the camera bodies relative to the neck. The specifications on the required velocities and control bandwidth for the neck motions are typically much less stringent than those for the camera motions, so that the inertial effects for the neck are usually neglected. The relatively simple nature of the oculomotor plant means that straightforward proportional-derivative (PD) or proportional-integral-derivative (PID) control systems are often sufficient for implementing tracking or pursuit motion. Some systems have employed more complex optimal control systems (e.g., [5]) which provide improved disturbance rejection and trajectory following accuracy compared to the simpler approaches.

There is a serious difficulty in controlling camera motion systems, however, caused by delays in the control loops. Such delays include the measurement delay due to the time needed to acquire and digitize the camera image and subsequent computations, such as feature extraction and target localization. There is also a delay or latency arising from the time needed to compute the controller output signal [6]. If these delays are not dealt with, a simple PD or PID controller can become unstable, leading to excessive vibration or shaking and loss of target tracking.

There are a number of approaches to dealing with delay. PID or PD systems can be made robust to delays simply by increasing system damping by reducing the proportional feedback gain to a sufficiently low value [7]. This results in a system that responds to changes in target position very slowly, however, and is unacceptable for most applications. For control of saccadic motion, a *sample/hold* can be used, where the position error is sampled at the time a saccade is triggered, and held in a first-order hold (integrator) [8]. In this way, the position error seen by the controller is held constant until the saccadic motion is completed. The controller is insensitive to any changes in the actual target position until the end of this *refractory* period. This stabilizes the controller, but has the drawback that if the target moves during the refractory period, the position error at the end of the refractory period can be large. In this case, another, corrective or secondary, saccadic motion may need to be triggered. For stabilization of pursuit control systems in the presence of delay, an internal positive feedback loop can be employed [4, 8]. This positive feedback compensates for delays in the negative feedback servo loop created by the time taken to acquire an image and compute the target velocity error. The positive feedback loop sends a delayed *efference copy* of the current commanded camera velocity (which is the output of the pursuit controller) back to the velocity error comparator where it is added to the measured velocity error. The positive feedback delay is set so that it arrives at the velocity error comparator at the same time as the measurement of the effect of the current control command, effectively canceling out the negative feedback and producing a new target velocity for the controller. Another delay handling technique is to use *predictive control*, such as the Smith Predictor, where the camera position and controller states are predicted for a time T in the future, where T is the controller delay, and control signals appropriate for those states are computed and applied immediately [6, 7]. Predictive methods make strong assumptions on changes in the external environment (e.g., that all objects in the scene are static or traversing known smooth trajectories). Such methods can perform poorly when these assumptions are violated.

The Next-Look Problem and Motion Planning The control of pursuit and saccadic motions are usually handled by different controllers. While pursuit or tracking behavior can be implemented using frequent small

saccade-like motions, this can produce jumpy images which may degrade subsequent processing operations. With multiple controllers, there needs to be a way for the possibly conflicting commands from the controllers to be integrated and arbitrated. The simplest approach uses the output of the pursuit control system by default, with a switch over to the output of the saccade control system whenever the position error is greater than some threshold and switching back to pursuit control when the position error drops below another (lower) threshold.

Pursuit or tracking of visual targets is just one type of motor activity. Activities such as visual search may require large shifts of camera position to be executed based on a complex analysis of the visual input. The process of determining the active vision system controller set point is often referred to as *sensor planning* [1] or the *next-look problem* [9]. The next-look problem can be interpreted as determining sensor positions which increase or maximize the information content of subsequent measurements. In a visual search task, for example, the next-look may be specified to be a location which is expected to maximally discriminate between target and distractor. One principle that has been successfully employed in next-look processes is that of entropy minimization over viewpoints. In an object recognition or visual search task, this approach takes as the next viewpoint that which is maximally informative relative to the most probable hypotheses [10]. A common approach to the next-view problem in robotic systems is to employ an *attention mechanism* to provide the location of the next view. Based on models of mammalian vision systems, attention mechanisms determine *salient* regions in visual input, which compete or interact in winner-takes-all fashion to select a single location as the target for the subsequent motion [8].

Application

In the late 1980s and early 1990s, commercial camera motion platforms lacked the performance needed by robotics researchers and manufacturers. This led many universities to construct their own platforms and develop control systems for them. These were generally binocular camera systems with pan and tilt degrees of freedom for each camera. Often, to simplify the design, a common tilt action was employed for both

cameras, and the pan actions were sometimes linked together to provided vergence and/or version motions only. Examples include the UPenn head [11], generally recognized as the first of its kind, the Harvard head [12], the KTH head [13], the TRISH head from the University of Toronto [14], the Rochester head [15], the SAGEM-GEC-Inria-Oxford head [16], the Surrey head [17], the LIFIA head [18], the LIA/AUC head [19], and the Technion head [5]. These early robotic heads generally used PD servo loops, some with delay compensation mechanisms as described above, and were capable of speeds up to 180 degrees per second. The pan axis maximum rotational velocities were usually higher than those of the tilt and vergence speeds. The axes were most often driven either by DC motors or by stepper motors.

A more recent example of a research system is the head of the iCub humanoid robot [20]. Unlike the early robotic heads, which were one-off systems limited to use in a single laboratory, this robot was developed by a consortium of European institutions and is used in many different research laboratories. It has independent pan and common tilt for two cameras as well as three neck degrees of freedom. The maximum pan speed is 180 degrees per second, and the maximum tilt speed is 160 degrees per second.

Currently, most robotic active vision systems are based on commercially available monocular pan-tilt platforms. The great majority of commercial platforms are designed for surveillance applications and are relatively slow. There are a few systems with specifications that are suitable for robotic active vision systems. Perhaps the most commonly used of these fast platforms are made by FLIR Motion Control Systems, Inc. (formerly Directed Perception). These are capable of speeds up to 120 degrees per second and can handle loads of up to 90 lbs. Commercial systems generally lack torsional motion and hence are not suitable for precise stereo vision applications.

The fastest current commercial pan/tilt units, as well as the early research platforms, only reach maximum speeds of around 200 degrees per second. This is sufficient to match the speeds of human pursuit eye movements, which top out around 100 degrees per second. However, if these speeds are compared to the maximum speed of 800 degrees per second for human saccadic motions, it can be seen that the performance of robotic active vision motion platforms still has room for improvement.

References

1. Tarabanis K, Tsai RY, Allen PK (1991) Automated sensor planning for robotic vision tasks. In: Proceedings of the 1991 IEEE conference on robotics and automation, Sacramento, pp 76–82
2. Yi S, Haralick RM, Shapiro LG (1990) Automatic sensor and light source positioning for machine vision. In: Proceedings of the computer vision and pattern recognition conference (CVPR), Atlantic City, June 1990, pp 55–59
3. Kato R, Grantyn A, Dalezios Y, Moschovakis AK (2006) The local loop of the saccadic system closes downstream of the superior colliculus. *Neuroscience* 143(1):319–337
4. Robinson DA (1968) The oculomotor control system: a review. *Proc IEEE* 56(6):1032–1049
5. Rivlin E, Rotstein H (2000) Control of a camera for active vision: foveal vision, smooth tracking and saccade. *Int J Comput Vis* 39(2):81–96
6. Brown C (1990) Gaze controls with interactions and delays. *IEEE Trans Syst Man Cybern* 20(1):518–527
7. Sharkey PM, Murray DW (1996) Delays versus performance of visually guided systems. *IEE Proc Control Theory Appl* 143(5):436–447
8. Clark JJ, Ferrier NJ (1992) Attentive visual servoing. In: Blake A, Yuille AL (eds) *An introduction to active vision*. MIT, Cambridge, pp 137–154
9. Swain MJ, Stricker MA (1993) Promising directions in active vision. *Int J Comput Vis* 11(2):109–126
10. Arbel T, Ferrie FP (1999) Viewpoint selection by navigation through entropy maps. In: Proceedings of the seventh IEEE international conference on computer vision, Kerkyra, pp 248–254
11. Krotkov E, Bajcsy R (1988) Active vision for reliable ranging: cooperating, focus, stereo, and vergence. *Int J Comput Vis* 11(2):187–203
12. Ferrier NJ, Clark JJ (1993) The Harvard binocular head. *Int J Pattern Recognit Artif Intell* 7(1):9–31
13. Pahlavan K, Eklundh J-O (1993) Heads, eyes and head-eye systems. *Int J Pattern Recognit Artif Intell* 7(1):33–49
14. Miliotis E, Jenkin M, Tsotsos J (1993) Design and performance of TRISH, a binocular robot head with torsional eye movements. *Int J Pattern Recognit Artif Intell* 7(1):51–68
15. Coombs DJ, Brown CM (1993) Real-time binocular smooth pursuit. *Int J Comput Vis* 11(2):147–164
16. Murray DW, Du F, McLauchlan PF, Reid ID, Sharkey PM, Brady M (1992) Design of stereo heads. In: Blake A, Yuille A (eds) *Active vision*. MIT, Cambridge, Massachusetts, USA, pp 155–172
17. Pretlove JRG, Parker GA (1993) The Surrey attentive robot vision system. *Int J Pattern Recognit Artif Intell* 7(1):89–107
18. Crowley JL, Bobet P, Mesrabi M (1993) Layered control of a binocular camera head. *Int J Pattern Recognit Artif Intell* 7(1):109–122
19. Christensen HI (1993) A low-cost robot camera head. *Int J Pattern Recognit Artif Intell* 7(1):69–87
20. Beira R, Lopes M, Praga M, Santos-Victor J, Bernardino A, Metta G, Becchi F, Saltaren R (2006) Design of the robot-cub (iCub) head. In: Proceedings of the 2006 IEEE international conference on robotics and automation, Orlando, Florida, USA, pp 94–100

Active Stereo Vision

Andrew Hogue¹ and Michael R. M. Jenkin²

¹Faculty of Business and Information Technology,
University of Ontario Institute of Technology,
Oshawa, ON, Canada

²Department of Computer Science and Engineering,
York University, Toronto, ON, Canada

Related Concepts

► Camera Calibration

Definition

Active stereo vision utilizes multiple cameras for 3D reconstruction, gaze control, measurement, tracking, and surveillance. Active stereo vision is to be contrasted with passive or dynamic stereo vision in that passive systems treat stereo imagery as a series of independent static images while active and dynamic systems employ temporal constraints to integrate stereo measurements over time. Active systems utilize feedback from the image streams to manipulate camera parameters, illuminants, or robotic motion controllers in real time.

Background

Stereo vision uses two or more cameras with overlapping fields of view to estimate 3D scene structure from 2D projections. Binocular stereo vision – the most common implementation – uses exactly two cameras, yet one can utilize more than two at the expense of computational speed within the same algorithmic framework.

The “passive” stereo vision problem can be described as a system of at least two cameras attached rigidly to one another with constant intrinsic calibration parameters (assumed), and the stereo pairs are considered to be temporally independent. Thus no assumptions are made, nor propagated, about camera motion within the algorithmic framework. Passive vision systems are limited to the extraction of metric information from a single set of images taken from

different locations in space (or at different times) and treat individual frames in stereo video sequences independently. Dynamic stereo vision systems are characterized by the extraction of metric information from sequences of imagery (i.e., video) and employ temporal constraints or consistency on the sequence (e.g., optical flow constraints). Thus, dynamic stereo systems place assumptions on the camera motion such as its smoothness (and small motion) between subsequent frames. Active stereo vision systems subsume both passive and dynamic stereo vision systems and are characterized by the use of robotic camera systems (e.g., stereo heads) or specially designed illuminant systems (e.g., structured light) coupled with a feedback system (see Fig. 1) for motor control. Although systems can be designed with more modest goals – object tracking, for example – the common computational goal is the construction of large-scale 3D models of extended environments.

Theory

Fundamentally, active stereo systems (see [1]) must solve three rather complex problems: (1) spatial correspondence, (2) temporal correspondence, and (3) motor/camera/illuminant control. Spatial correspondence is required in order to infer 3D depth information from the information available in camera images captured at one time instant, while temporal correspondence is necessary to integrate visual information over time. The spatial and temporal correspondences can either be treated as problems in isolation or integrated within a common framework. For example, stereo correspondence estimation can be seeded using an ongoing 3D representation using temporal coherence (e.g., [2, 3]) or considered in isolation using standard disparity estimation algorithms (see [4]).

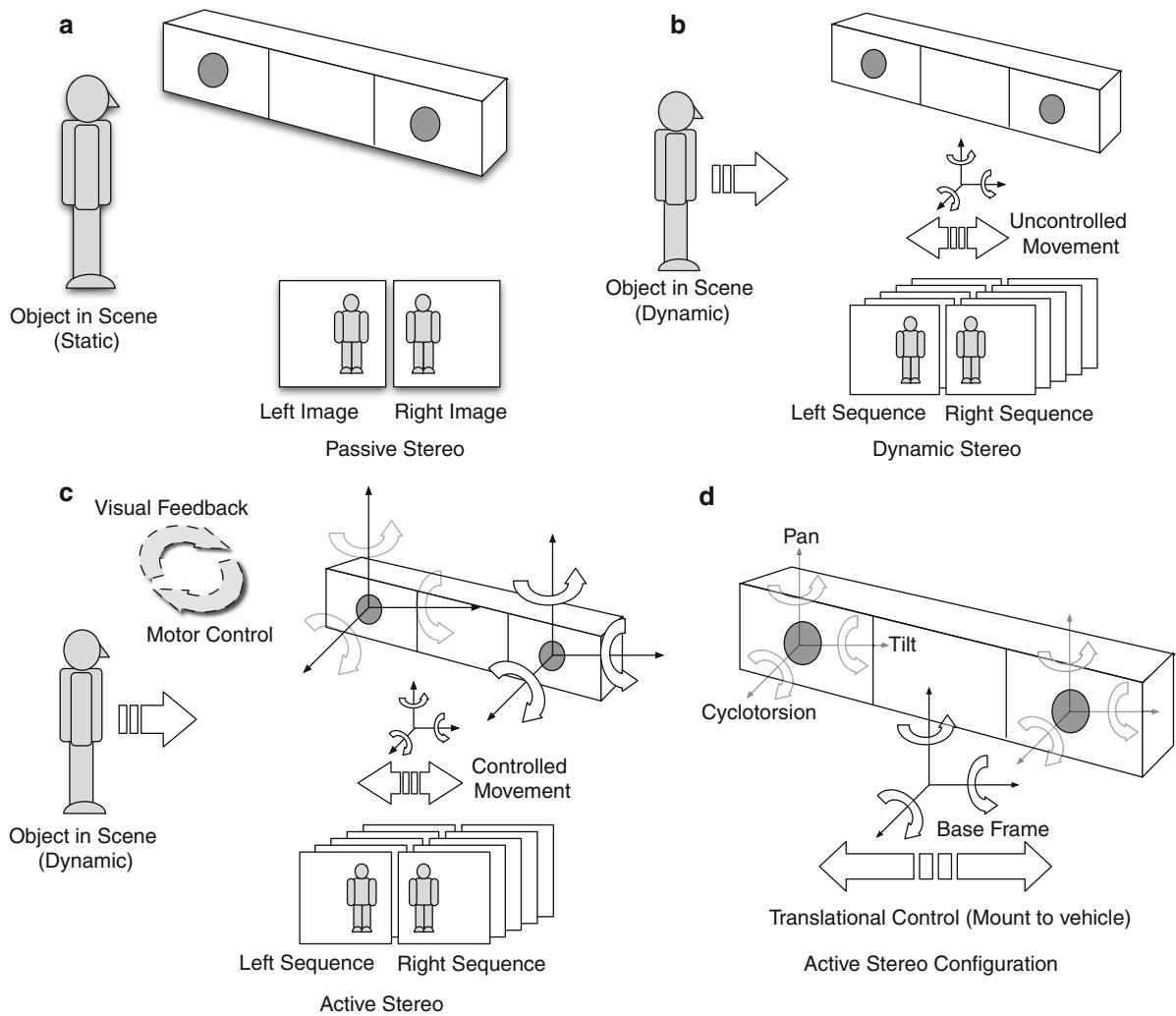
Motor or camera control systems are necessary to move (rotate and translate) the cameras so they look in the appropriate direction (i.e., within a tracking or surveillance application), change their intrinsic camera parameters (e.g., focal length or zoom), or to tune the image processing algorithm to achieving higher accuracy for a specific purpose. Solving these three problems in an active stereo system enables one to develop “intelligent” algorithms that infer ego-motion [5], autonomously control vehicles throughout the world [6], and/or reconstruct 3D models of the

environment [7, 8]. Examples of the output of such a system is shown in Fig. 2, and [9] provides an example of an active system that interleaves the vergence and focus control of the cameras with surface estimation. The system uses an adaptive self-calibration method that integrates the estimation of camera parameters with surface estimation using prior knowledge of the calibration, the motor control, and the previous estimate of the 3D surface properties. The resulting system is able to automatically fixate on salient visual targets in order to extend the surface estimation volume.

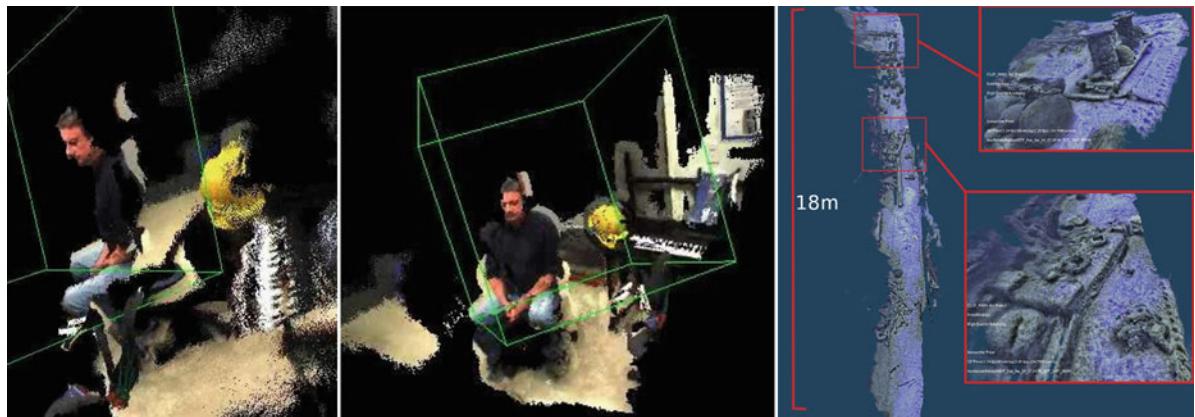
Although vision is a powerful sensing modality, it can fail. This is a critical issue for active stereo vision where data is integrated over time. The use of complementary sensors – traditionally Inertial Measurement Units (see [10]) – augment the camera hardware system with the capability to estimate the system dynamics using real-world constraints. Accelerometers, gyroscopes, and compasses can provide timely and accurate information either to assist in temporal correspondences and ego-motion estimation or as a replacement when visual information is unreliable or absent (i.e., dead reckoning).

Relation to Robotics and Mapping

A wide range of different active and dynamic stereo systems have been built (e.g., [7, 8, 11, 12]). Active systems are often built on top of mobile systems (e.g., [7]) blurring the distinction between active and dynamic systems. In robotics, active stereo vision has been used for vehicle control in order to create 2D or 3D maps of the environment. Commonly the vision system is complemented by other sensors. For instance in [13], active stereo vision is combined with sonar sensors to create 2D and 3D models of the environment. Murray and Little [14] use a trinocular stereo system to create occupancy maps of the environment for in-the-loop path planning and robot navigation. Diebel et al. [15] employ active stereo vision for simultaneous estimation of the robot location and 3D map construction, and [7], describes a vision system used for in-the-loop mapping and navigational control for an aquatic robot. Davison, in [6], was one of the first to effectively demonstrate the use of active stereo vision technology as part of the navigation loop. The system used a stereo head to selectively fixate scene features that improve the quality of the estimated map and trajectory. This involved using knowledge of the



Active Stereo Vision, Fig. 1 Different types of stereo systems



Active Stereo Vision, Fig. 2 Point cloud datasets obtained by the active stereo system described in [7]

current map of the environment to point the camera system in the direction where it should find salient features that it had seen before, move the robot to a location where the features are visible, and then searching visually to find image locations corresponding to these features.

Active Stereo Heads

The development of hardware platforms to mimic human biology has resulted in a variety of different designs and methods for controlling binocular sets of cameras. These result in what is known as “stereo heads” (see entry on *the evolution of stereo heads* in this volume). These hardware platforms all have a common set of constraints, i.e., the systems consist of two cameras (binocular) with camera intrinsics/extrinsics that may be controlled. In [16], an active stereo vision system is developed that mimics human biology that uses a bottom-up saliency map model coupled with a selective attention function to select salient image regions. If both left and right cameras estimate similar amounts of saliency in the same areas, the vergence of the cameras are controlled so that the cameras are focused on this particular landmark.

Autocalibration

A fundamental issue with active stereo vision is the need to establish and maintain calibration parameters online. Intrinsics and extrinsics are necessary to the 3D estimation process as they define the epipolar constraints which enable efficient disparity estimation algorithms [17, 18]. Each time the camera parameters are modified (e.g., vergence of the cameras, change of focus), the epipolar geometry must be re-estimated. Although kinematic modeling of motor systems provide good initial estimates of changes in camera pose, this is generally insufficiently accurate to be used by itself to update camera calibration. Thus, autocalibration becomes an important task within active stereo vision. Approaches to autocalibration are outlined in [17, 19]. In [17], the autocalibration algorithm operates on pairs of stereo images taken in sequence. A projective reconstruction for motion and structure of the scene is constructed. This is performed for each pair of stereo images individually for the same set of features (thus they must be matched in the stereo pairs as well as tracked temporally). The projective solutions can be upgraded to an affine solution (ambiguous

up to a rigid rotation/translation/scale) by noting these features should match in 3D space as well as in 2D space. A transformation can be linearly estimated that constrains the projective solution to an affine reconstruction. Once the plane at infinity is known, the affine solution may be upgraded to a metric solution. In order to achieve the desired accuracy in the intrinsics, a nonlinear minimization scheme is employed to improve the solution. If one trusts the accuracy of the camera motion control system, the extrinsics can be seeded with this information in a nonlinear optimization scheme that minimizes the reprojection error of the image matching points and their 3D triangulated counterparts. This nonlinear optimization is known as bundle adjustment [20] and is used in a variety of forms in the structure-from-motion literature (see [17, 18]).

Relation to Other Types of Stereo Systems

Since active stereo systems are characterized by the use of visual feedback to inform motor control systems (or higher-level vehicular navigational systems), they are related to a wide range of research areas and hardware systems. Mounting a stereo system to a robotic vehicle is common in the robotics literature to inform the navigation system about the presence of obstacles [21] and to provide input to mapping algorithms [22]. The use of such active systems are applicable directly to autonomous systems as they provide a high amount of controllable accuracy and dense measurements at relatively low computational cost. One significant example is the use of active stereo in the Mars Rover autonomous vehicles [12].

Estimating 3D information from stereo views is problematic due to the lack of (or ambiguous) texture in man-made environments. This can be alleviated with the use of active illumination [23]. Projecting a known pattern, rather than uniform lighting, into the scene enables the estimation of a more dense disparity field using standard stereo disparity estimation algorithms due to the added texture in textureless regions (see [24]). The illumination may be controlled actively depending on perceived scene texture, the desired range, or the ambient light intensity of the environment. The illumination may be within the visible light spectrum or in the infrared spectrum as most camera sensors are sensitive to IR light. This has the added advantage that humans in the environment are not affected by the additional illumination.

Application

Active stereo vision is characterized by the use of visual feedback in multi-camera systems to control the intrinsics and extrinsics of the cameras (or vehicular platforms). Active stereo vision systems find a wide range of application in autonomous vehicle navigation, gaze tracking, and surveillance. A host of hardware systems exist and commonly utilize two cameras for binocular stereo and motors to control the gaze/orientation of the system. Visual attentive processes (e.g., [25, 26]) may be used to determine the next viewpoint for a particular task, and dense stereo algorithms can be used for estimating 3D structure of the scene. Fundamental computational issues include autocalibration of the sensor with changes in its configuration and the development of active stereo control and reconstruction algorithms.

References

1. Vieville T (1997) *A few steps towards 3d active vision*. Springer, New York/Secaucus
2. Leung C, Appleton B, Lovell B, Sun C (2004) An energy minimisation approach to stereo-temporal dense reconstruction. In: Proceedings of the 17th international conference on pattern recognition, vol 4, Cambridge, pp 72–75
3. Min D, Yea S, Vetro A (2010) Temporally consistent stereo matching using coherence function. In: 3DTV-conference: the true vision – capture, transmission and display of 3D video (3DTV-CON), 2010, Tampere, pp 1–4
4. Scharstein D, Szeliski R (2002) A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int J Comput Vis* 47(1–3):7–42
5. Olson CF, Matthies LH, Schoppers M, Maimone MW (2001) Stereo ego-motion improvements for robust rover navigation. In: Proceedings IEEE international conference on robotics and automation, vol 2, Seoul, pp 1099–1104
6. Davison AJ (1998) Mobile robot navigation using active vision. PhD thesis, University of Oxford
7. Hogue A, Jenkin M (2006) Development of an underwater vision sensor for 3d reef mapping. In: IEEE/RSJ international conference on intelligent robots and systems, Beijing, pp 5351–5356
8. Se S, Jasobedzki P (2005) Instant scene modeler for crime scene reconstruction. In: 2005 IEEE computer society conference on computer vision and pattern recognition workshop on safety and security applications, vol III, San Diego, pp 123–123
9. Ahuja N, Abbott A (1993) Active stereo: integrating disparity, vergence, focus, aperture and calibration for surface estimation. *IEEE Trans Pattern Anal Mach Intell* 15(10): 1007–1029
10. Everett HR (1995) (ECCV) Sensors for mobile robots: theory and application. A. K. Peters, Ltd., Natick
11. Grosso E, Tistarelli M, Sandini G (1992) Active/dynamic stereo for navigation. In: Second European conference on computer vision, Santa Margherita Ligure, pp 516–525
12. Maimone MW, Leger PC, Biesiadecki JJ (2007) Overview of the mars exploration rovers autonomous mobility and vision capabilities. In: IEEE international conference on robotics and automation (ICRA) space robotics workshop, Rome
13. Wallner F, Dillman R (1995) Real-time map refinement by use of sonar and active stereo-vision. *Robot Auton Syst* 16(1):47–56. Intelligent robotics systems SIRS'94
14. Murray D, Little JJ (2000) Using real-time stereo vision for mobile robot navigation. *Auton Robot* 8(2): 161–171
15. Diebel J, Reutersward K, Thrun S, Davis J, Gupta R (2004) Simultaneous localization and mapping with active stereo vision. In: IEEE/RSJ international conference on intelligent robots and systems, vol 4, Sendai, pp 3436–3443
16. Jung BS, Choi SB, Ban SW, Lee M (2004) A biologically inspired active stereo vision system using a bottom-up saliency map model. In: Rutkowski L, Siekmann J, Tadeusiewicz R, Zadeh LA (eds) *Artificial intelligence and soft computing – ICAISC 2004*. Volume 3070 of lecture notes in computer science. Springer, Berlin/Heidelberg, pp 730–735
17. Hartley RI, Zisserman A (2000) *Multiple view geometry in computer vision*. Cambridge University Press, Cambridge
18. Faugeras O, Luong QT, Papadopoulou T (2001) *The geometry of multiple images: the laws that govern the formation of images of a scene and some of their applications*. MIT, Cambridge
19. Horaud R, Csurka G (1998) Self-calibration and euclidean reconstruction using motions of a stereo rig. In: Sixth international conference on computer vision, Bombay, pp 96–103
20. Triggs B, McLauchlan P, Hartley R, Fitzgibbon A (2000) Bundle adjustment – a modern synthesis. In: Triggs B, Zisserman A, Szeliski R (eds) *Vision algorithms: theory and practice*. Volume 1883 of lecture notes in computer science. Springer, New York, pp 298–372
21. Williamson T, Thorpe C (1999) A trinocular stereo system for highway obstacle detection. In: IEEE international conference on robotics and automation, Detroit, pp 2267–2273
22. Schleicher D, Bergasa LM, Ocaña M, Barea R, López E (2010) Real-time hierarchical stereo visual slam in large-scale environments. *Robot Auton Syst* 58:991–1002
23. Se S, Jasobedzki P, Wildes R (2007) Stereo-vision based 3d modeling of space structures. In: Proceedings of the SPIE conference on sensors and systems for space applications, vol 6555, Orlando
24. Rusinkiewicz S, Hall-Holt O, Levoy M (2002) Real-time 3d model acquisition. *ACM Trans Graph* 21(3):438–446
25. Frintrop S, Rome E, Christensen HI (2010) Computational visual attention systems and their cognitive foundations: a survey. *ACM Trans Appl Percept* 7(1):1–39
26. Tsotsos JK (2001) Motion understanding: task-directed attention and representations that link perception with action. *Int J Comput Vis* 45(3):265–280

Active Vision

► [Animat Vision](#)

Activity Analysis

► [Multi-camera Human Action Recognition](#)

Activity Recognition

Wanqing Li¹, Zicheng Liu² and Zhengyou Zhang³

¹University of Wollongong, Wollongong, NSW, Australia

²Microsoft Research, Microsoft Corporation, Redmond, WA, USA

³Microsoft Research, Redmond, WA, USA

Synonyms

[Action recognition](#)

Related Concepts

► [Gesture Recognition](#)

Definition

Activity recognition refers to the process of identifying the types of movement performed by humans over a certain period of time. It is also known as action recognition when the period of time is relatively short.

Background

The classic study on visual analysis of biological motion using moving light display (MLD) [1] has inspired tremendous interests among the computer vision researchers in the problem of recognizing human motion through visual information. The commonly used devices to capture human movement include human motion capture (MOCAP) with or without markers, multiple video camera systems, and single video camera systems. A MOCAP device

usually works under controlled environment to capture the three-dimensional (3D) joint locations or angles of human bodies; multiple camera systems provide a way to reconstruct 3D body models from multiple viewpoint images. Both MOCAP and multiple camera systems have physical limitations on their use, and single camera systems are probably more practical for many applications. The latter, however, captures least visual information and, hence, is the most challenging setting for activity recognition. In the past decade, research in activity recognition has mainly focused on single camera systems. Recently, the release of commodity depth cameras, such as Microsoft Kinect Sensors, provides a new feasible and economic way to capture simultaneously two-dimensional color information and depth information of the human movement and, hence, could potentially advance the activity recognition significantly.

Regardless of which capturing device is used, a useful activity recognition system has to be independent of anthropometric differences among the individuals who perform the activities, independent of the speed at which the activities are performed, robust against varying acquisition settings and environmental conditions (for instance, different viewpoints and illuminations), scalable to a large number of activities, and capable of recognizing activities in a continuous manner. Since a human body is usually viewed as an articulated system of the rigid links or segments connected by joints, human motion can be considered as a continuous evolution of the spatial configuration of the segments or body posture, and effective representation of the body configuration and its dynamics over time has been the central to the research of human activity recognition.

Theory

Let $O = \{o_1, o_2, \dots, o_n\}$ be a sequence of observations of the movement of a person over a period of time. The observations can be a sequence of joint angles, a sequence of color images or silhouettes, a sequence of depth maps, or a combination of them. The task of activity recognition is to label O into one of the L classes $C = \{c_1, c_2, \dots, c_L\}$. Therefore, solutions to the problem of activity recognition are often based on machine learning and pattern recognition approaches, and an activity recognition system usually involves

extracting features from the observation sequence O , learning a classifier from training samples, and classifying O using the trained classifier. However, the spatial and temporal complexity of human activities has led researchers to cast the problem from different perspectives. Specifically, the existing techniques for activity recognition can be divided into two categories based on whether the dynamics of the activities is implicitly or explicitly modeled.

In the first category [2–9], the problem of activity recognition is cast from a temporal classification problem to a static classification one by representing activities using descriptors. A descriptor is extracted from the observation sequence O , which intends to capture both spatial and temporal information of the activity and, hence, to model the dynamics of the activity implicitly. Activity recognition is achieved by a conventional classifier such as Support Vector Machines (SVM) or K-nearest neighborhood (KNN). There are three commonly used approaches to extract activity descriptors.

The first approach builds motion energy images (MEI) and motion history images (MHI), proposed by Bobick and Davis [2], by stacking a sequence of silhouettes to capture where and how the motion is performed. Activity descriptors are extracted from the MEI and MHI. For instance, seven Hu moments were extracted in [2] to serve as action descriptors and recognition was based on the Mahalanobis distance between the moment descriptors of the trained activities and the input activity.

The second approach considers a sequence of silhouettes as a spatiotemporal volume, and an activity descriptor is computed from the volume. Typical examples are the work by Yilmaz and Shah [3] which computes the differential geometric surface properties (i.e., Gaussian curvature and mean curvature); the work by Gorelick et al. [4] which extracts space-time saliency, action dynamics, and shape structure and orientation; and the work by Mokhber et al. [5] which calculates the 3D moments of the volume.

The third approach describes an activity using a set of spatiotemporal interest points (STIPs). The general concept is first to detect STIPs from the observations O which is usually a video sequence. Features are then extracted from a local volume around each STIP, and a descriptor can be formed by simply aggregating the local features together to become a bag of features or

by classifying the STIPs into a set of vocabulary (i.e., a bag of visual words) and calculating the histogram of the occurrence of the vocabulary within the observation sequence O . There are two commonly used STIP extraction techniques. One extends Harris corner detection and automatic scale selection in 2D space to 3D space and time [6] and the other is based on a pair of one-dimensional (1D) Gabor filters applied temporally and spatially [7]. Recently, another STIP detector has been proposed by decomposing an image sequence into spatial components and motion components using nonnegative matrix factorization and detecting STIPs in 2D spatial and 1D motion space using difference of Gaussian (DoG) detectors [8]. In terms of the classifier for STIP-based descriptors, besides SVM and KNN, latent topic models such as the probabilistic latent semantic analysis (pLSA) model and latent Dirichlet allocation (LDA) were used in [9]. STIP-based descriptors have a few practical advantages including being applicable to image sequences in realistic conditions, not requiring foreground/background separation or human tracking, and having the potential to deal with partial occlusions [10]. In many realistic applications, an activity may occupy only a small portion of the entire space-time volume of a video sequence. In such situations, it does not make sense to classify the entire video. Instead, one needs to locate the activity in space and time. This is commonly known as the activity detection or action detection problem. Techniques have been developed for activity detection using interest points [11].

In the second category [12–17], the proposed methods usually follow the concept that an activity is a temporal evolution of the spatial configuration of the body parts and, hence, emphasize more on the dynamics of the activities than the methods in the first category. They usually extract a sequence of feature vectors, each feature vector being extracted from a frame, or a small neighborhood, of the observation sequence O . The two commonly used approaches are temporal templates and graphical models.

The temporal-template-based approach, typically, directly represents the dynamics through exemplar sequences and adopts dynamic time warping (DTW) to compare an input sequence with the exemplar sequences. For instance, Wang and Suter [18] employed locality preserving projection (LPP) to project a sequence of silhouettes into a low-dimensional space to characterize the spatiotemporal

property of the activity and used DTW and temporal Hausdorff distance for similarity matching.

In the graphical model-based approach, both generative and discriminative models have been extensively studied for activity recognition. The most prominent generative model is the hidden Markov model (HMM), where sequences of observed features are grouped into similar configuration, i.e., states, and both the probability distribution of the observations at each state and the temporal transitional functions between these states are learned from training samples. The first work on action recognition using HMM is probably by Yamato et al. [12], where a discrete HMM is used to represent sequences over a set of vector-quantized silhouette features of tennis footage. HMM is a powerful tool to model a small number of short-term activities since a practical HMM is usually a fixed- and low-order Markov chain. Notable early extensions to overcome this drawback of the HMM are the variable-length Markov models (VLMM) and layered HMM. For details, the readers are referred to [13, 14], respectively. Recently, a more general generative graphical model, referred to as an action graph, has been established in [15], where nodes of the action graph represents salient postures that are used to characterize activities and shared by different activities, and weight between two nodes measures the transitional probability between the two postures represented by the two nodes. An activity is encoded by one or multiple paths in the action graph. Due to the sharing mechanism, the action graph can be trained and also easily expanded to new actions with a small number of training samples. In addition, the action graph does not need special nodes representing beginning and ending postures of the activities and, hence, allows continuous recognition.

The generative graphical models often rely on an assumption of statistical independence of observations to compute the joint probability of the states and the observations. This makes it hard to model the long-term contextual dependencies which is important to the recognition of activities over a long period of time. The discriminative models, such as conditional random fields (CRF), offer an effective way to model long-term dependency and compute the conditional probability that maps the observations to the motion class labels. The linear chain CRF was employed in [16] to recognize ten different human activities using features of combined shape-context and pair-wise edge

features extracted at a variety of scales on the silhouettes and 3D joint angles. The results have shown that CRFs outperform the HMM and are also robust against the variability of the test sequences with respect to the training samples. More recently, Wang and Mori [17] modeled a human action by a flexible constellation of parts conditioned on image observations using hidden conditional random fields (HCRF) and achieved highly accurate frame-based action recognition.

Despite the extensive effort and progress in activity recognition research in the past decade, continuous recognition of activities under realistic conditions, such as with viewpoint invariance and large number of activities, remains challenging.

Application

Activity recognition has many potential applications. It is one of the key enabling technologies in security and surveillance for automatic monitoring of human activities in a public space and of activities of daily living of elderly people at home. Robust understanding and interpretation of human activities also allow a natural way for humans to interact with machines. A proper modeling of the spatial configuration and dynamics of human motion would enable realistic synthesis of human motion for gaming and movie industry and help train humanoid robots in a flexible and economic way. In sports, activity recognition technology has also been used in training and in the retrieval of video sequences.

References

1. Johansson G (1973) Visual perception of biological motion and a model for its analysis. *Percept Psychophys* 14(2): 201–211
2. Bobick A, Davis J (2001) The recognition of human movement using temporal templates. *IEEE Trans Pattern Anal Mach Intell* 23(3):257–267
3. Yilmaz A, Shah M (2008) A differential geometric approach to representing the human actions. *Comput Vision Image Underst* 109(3):335–351
4. Gorelick L, Blank M, Shechtman E, Irani M, Basri R (2007) Actions as space-time shapes. *IEEE Trans Pattern Anal Mach Intell* 29(12):2247–2253
5. Mokhber A, Achard C, Milgram M (2008) Recognition of human behavior by space-time silhouette characterization. *Pattern Recogn* 29(1):81–89

6. Laptev I, Lindeberg T (2003) Space-time interest points. In: International conference on computer vision, Nice, pp 432–439
7. Dollar P, Rabaud V, Cottrell G, Belongie S (2005) Behavior recognition via sparse-temporal features. In: 2nd joint IEEE international workshop on visual surveillance and performance evaluation of tracking and surveillance, Beijing, pp 65–72
8. Wong SF, Cipolla R (2007) Extracting spatiotemporal interest points using global information. In: International conference on computer vision, Rio de Janeiro, pp 1–8
9. Niebles JC, Wang H, Fei-Fei L (2008) Unsupervised learning of human action categories using spatial-temporal words. *Int J Comput Vision* 79(3):299–318
10. Liu J, Luo J, Shah M (2009) Recognizing realistic actions from videos “in the wild”. In: International conference on computer vision and pattern recognition (CVPR), Miami, pp 1–8
11. Yu G, Goussies NA, Yuan J, Liu Z (2011) Fast action detection via discriminative random forest voting and top-K subvolume search. *IEEE Trans Multimedia* 13:507–517
12. Yamato J, Ohya J, Ishii K (1992) Recognizing human action in time-sequential images using hidden Markov model. In: International conference on computer vision and pattern recognition (CVPR), Champaign, pp 379–385
13. Galata A, Johnson N, Hogg D (2001) Learning variable-length Markov models of behaviour. *Comput Vision Image Underst* 81:398–413
14. Oliver N, Garg A, Horvits E (2004) Layered representations for learning and inferring office activity from multiple sensory channels. *Comput Vision Image Underst* 96:163–180
15. Li W, Zhang Z, Liu Z (2008) Expandable data-driven graphical modeling of human actions based on salient postures. *IEEE Trans Circuits Syst Video Technol* 18(11): 1499–1510
16. Sminchisescu C, Kanaujia A, Metaxas D (2006) Conditional models for contextual human motion recognition. *Comput Vision Image Underst* 104:210–220
17. Wang Y, Mori G (2011) Hidden part models for human action recognition: probabilistic versus max margin. *IEEE Trans Pattern Anal Mach Intell* 33(7):1310–1323
18. Wang L, Suter D (2007) Learning and matching of dynamic shape manifolds for human action recognition. *IEEE Trans Image Process* 16:1646–1661

AdaBoost

Paolo Favaro¹ and Andrea Vedaldi²

¹Department of Computer Science and Applied Mathematics, Universität Bern, Switzerland
²Oxford University, Oxford, UK

Synonyms

Adaptive boosting; Discrete adaBoost

Definition

AdaBoost is an algorithm that builds a classifier by combining additively a set of weak classifiers. The weak classifiers are incorporated sequentially one at a time so that their combination reduces the empirical exponential loss.

Background

Boosting is a procedure to combine several classifiers with weak performance into one with arbitrarily high performance [1, 2] and was originally introduced by Robert Schapire in the machine learning community [3]. AdaBoost is a popular implementation of boosting for binary classification [4]. The enthusiasm generated by boosting, and in particular by AdaBoost, in machine learning can be highlighted via a quote of Breiman [1] saying that AdaBoost with trees is the “best off-the-shelf classifier in the world.” In practice, much of the popularity of AdaBoost is due to both its performance being in the same league as support vector machines [5] and its algorithmic simplicity. In the computer vision community, AdaBoost has been made very popular by the work of Viola and Jones in face detection [6]. What attracted much of the attention was that, by using a cascade of AdaBoost-trained classifiers and the notion of integral image and Haar wavelets [6, 7], Viola and Jones were able to detect faces in real time.

The boosting framework is fairly general and several implementations have been proposed, among which are AdaBoost [4]; Real AdaBoost, LogitBoost, and GentleBoost [1]; Regularized AdaBoost [8]; or extensions to multiple classes such as AdaBoost.MH [9].

Theory

This section describes the AdaBoost algorithm as originally given by Freund and Schapire [4]. This particular variant, also known as Discrete AdaBoost [1], is summarized in Algorithm 1.

The purpose of AdaBoost is to learn a *binary classifier* that is a function $H(\mathbf{x}) = y$ that maps data $\mathbf{x} \in \mathcal{X}$ (e.g., a scrap of text, an image, or a sound wave) to its

class label $y \in \{-1, +1\}$. In AdaBoost the classifier H is obtained as the sign of an additive combination of simple classifiers $h(\mathbf{x}) \in \{-1, +1\}$, called *weak hypotheses*:

$$H(\mathbf{x}) \doteq \text{sign} \left(\sum_{t=1}^m \alpha_t h_t(\mathbf{x}) \right), \quad (1)$$

where the coefficients $\alpha_t \in \mathbb{R}$. The input to AdaBoost is a set \mathcal{H} of weak hypotheses and n data-label pairs $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$; the output is a combination H of m weak hypothesis in \mathcal{H} and coefficients $\alpha_1, \dots, \alpha_m$ that fit the data, i.e., $H(\mathbf{x}_i) = y_i$ for most $i = 1, \dots, n$.

Let us denote with H_m the classifier H with m weak hypotheses shown in Eq. (1). AdaBoost operates sequentially by adding to H_{m-1} one new weak hypothesis (h_m, α_m) . While any weak hypothesis with performance better than chance can be used, it is more common to select the weak hypothesis h_m in the set \mathcal{H} that minimizes the weighted *empirical error* $\epsilon(h; \mathbf{w})$, i.e.,

$$h_m = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \epsilon(h; \mathbf{w}),$$

where

$$\epsilon(h; \mathbf{w}) \doteq \frac{\sum_{i=1}^n w_i [y_i \neq h(\mathbf{x}_i)]}{\sum_{i=1}^n w_i}$$

Algorithm 1 Discrete AdaBoost

- 1: Initialize $F_0(\mathbf{x}) = 0$ for all $\mathbf{x} \in \mathcal{X}$.
- 2: Initialize $w_i = 1$ for all $i = 1, 2, \dots, N$.
- 3: **for** $t = 1$ to m **do**
- 4: Find the weak hypothesis $h_t \in \mathcal{H}$ that minimizes

$$\epsilon(h_t; \mathbf{w}) \propto \sum_{i=1}^N w_i [h_t(\mathbf{x}_i) \neq y_i].$$

- 5: Let

$$\alpha_t \leftarrow \frac{1}{2} \log \frac{1 - \epsilon(h_t; \mathbf{w})}{\epsilon(h_t; \mathbf{w})};$$

- 6: Update the weights

$$w_i \leftarrow w_i e^{-y_i \alpha_t h_t(\mathbf{x}_i)};$$

- 7: Update the function $F_t = F_{t-1} + \alpha_t h_t$.
 - 8: **end for**
 - 9: Return the classifier $H(\mathbf{x}) = \text{sign } F_m(\mathbf{x})$.
-

with positive data weights $\mathbf{w} = (w_1, \dots, w_n)$. Here $[y_i \neq h(\mathbf{x}_i)]$ is equal to 1 if $y_i \neq h(\mathbf{x}_i)$ and 0 otherwise. Hence, the empirical error $\epsilon(h; \mathbf{w})$ is the average number of incorrect classifications of the weak hypothesis h on the weighted training data. The selected weak hypothesis h_m is then added to the current combination H_{m-1} with coefficient

$$\alpha_m = \frac{1}{2} \log \frac{1 - \epsilon(h_m; \mathbf{w})}{\epsilon(h_m; \mathbf{w})}. \quad (2)$$

While AdaBoost minimizes the empirical error of the weak hypothesis h_m at each iteration, the weights $\mathbf{w} = (w_1, \dots, w_n)$ are chosen so that the empirical error of H_m is reduced as well. AdaBoost starts with uniform weights $\mathbf{w} = (1, \dots, 1)$ and updates them according to the rule

$$w_i \leftarrow w_i e^{-y_i \alpha_m h_m(\mathbf{x}_i)}, \quad i = 1, \dots, n. \quad (3)$$

One intuitive interpretation of this rule is that it gives more importance to examples that are incorrectly classified. A formal justification is given in the next paragraph.

AdaBoost as Stagewise Minimization Denote by $F_m(\mathbf{x}) = \sum_{t=1}^m \alpha_t h_t(\mathbf{x})$ the additive combination of the weak hypotheses, so that the classifier $H_m(\mathbf{x})$ can be written as $\text{sign } F_m(\mathbf{x})$. AdaBoost performs a stagewise minimization of the cost

$$\frac{1}{n} \sum_{i=1}^n e^{-y_i F_m(\mathbf{x}_i)}.$$

This cost is known as the *empirical exponential loss* and is a convex upper bound to the empirical classification error of H_m :

$$\epsilon(H_m) \doteq \frac{1}{n} \sum_{i=1}^n [y_i \neq H_m(\mathbf{x}_i)] \leq \frac{1}{n} \sum_{i=1}^n e^{-y_i F_m(\mathbf{x}_i)}.$$

To understand the effect of the AdaBoost update on the empirical exponential loss, let $F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \alpha_m h_m(\mathbf{x})$ be the updated additive combination at

iteration m . As the parameters of F_{m-1} are fixed, the empirical exponential loss is a function E of α_m and h_m :

$$E(\alpha_m, h_m) \doteq \frac{1}{n} \sum_{i=1}^n w_i e^{-y_i \alpha_m h_m(\mathbf{x}_i)},$$

where $w_i = e^{-y_i F_{m-1}(\mathbf{x}_i)}$.

(4)

By taking the derivative of E with respect to α_m and by setting it to zero, one obtains the optimality condition

$$\begin{aligned} 0 &= \sum_{i=1}^n w_i y_i h_m(\mathbf{x}_i) e^{-y_i h_m(\mathbf{x}_i)} e^{\alpha_m} \\ &= \sum_{i: y_i \neq h_m(\mathbf{x}_i)} w_i e^{\alpha_m} - \sum_{i: y_i = h_m(\mathbf{x}_i)} w_i e^{-\alpha_m} \end{aligned}$$

which results in the optimal coefficient given in Eq. (2):

$$\begin{aligned} \alpha_m(h_m) &= \frac{1}{2} \log \frac{\sum_{i=1}^n w_i [y_i = h_m(\mathbf{x}_i)]}{\sum_{i=1}^n w_i [y_i \neq h_m(\mathbf{x}_i)]} \\ &= \frac{1}{2} \log \frac{1 - \epsilon(h_m; \mathbf{w})}{\epsilon(h_m; \mathbf{w})}. \end{aligned}$$

By substituting this expression back in the cost (4), one obtains

$$E(\alpha_m(h_m), h_m) = 2 \sqrt{\epsilon(h_m; \mathbf{w})(1 - \epsilon(h_m; \mathbf{w}))} \sum_{i=1}^n w_i$$

which achieves its smallest value when the empirical classification error $\epsilon(h_m; \mathbf{w})$ approaches either 0, its minimum, or 1, its maximum. Notice that if the error $\epsilon(h_m; \mathbf{w}) > 1/2$, then the corresponding weight α_m is negative. In other words, when the weak hypothesis h_m makes more mistakes than correct classifications, AdaBoost automatically swaps the sign of the output label so that $\epsilon(-h_m; \mathbf{w}) < 1/2$. Finally, the weight update Eq. (3) follows from

$$\begin{aligned} w_i &\leftarrow e^{-y_i F_m(\mathbf{x}_i)} = e^{-y F_{m-1}(\mathbf{x}_i)} e^{-y_i \alpha_m h_m(\mathbf{x}_i)} \\ &= w_i e^{-y_i \alpha_m h_m(\mathbf{x}_i)}. \end{aligned}$$

Application

One of the main uses of AdaBoost is for the recognition of patterns in data. Recognition can be formulated as a binary classification problem: Find whether data points match the pattern of interest or not. In computer vision, AdaBoost was popularized by its application to object detection, where the task is not only to recognize but also to localize within an image an object of interest (e.g., a face). Most of the ideas summarized in this section were first proposed by Viola and Jones [6].

A common technique for object detection is the *sliding window* detector. This method reduces the object detection problem to the task of classifying all possible image windows (i.e., patches) to find which ones are centered around the object of interest. In practice, windows may be sampled not only at all spatial locations but also at all scales and rotations. This results in a very large number of evaluations of the classifier function for each input image. Therefore, the computational efficiency of the classifier is of paramount importance.

Classifiers computed with AdaBoost can be made very computationally efficient by using weak hypotheses that are fast to compute and by letting AdaBoost select a small set of hypotheses most useful to the given problem. For example, in the Viola-Jones face detector, a weak hypothesis is computed by thresholding the output of a linear filter that computes averages over rectangular areas of the image. These filters are known as Haar wavelets and, because of their special structure, can be computed in constant time by using the *integral image* [6].

In order to further improve the speed of a sliding window detector, AdaBoost classifiers are often combined in a *cascade* [6]. A cascade exploits the fact that the vast majority of image windows are *not* centered around the object of interest and that, furthermore, most of these negative windows are easy to recognize as such. A cascade is built by appending one AdaBoost classifier after another. Classifiers are evaluated sequentially and an image window is rejected as soon as the response of a classifier is negative. All the classifiers are tuned to almost never reject a window that matches the object of interest (i.e., high recall). However, the first classifiers in the cascade are allowed to return several false positives (i.e., low precision) in exchange for a significantly reduced evaluation cost, obtained, for instance, by limiting the number of weak

hypotheses in them. By using this scheme, the computationally costly and highly accurate classifiers are evaluated only on the most challenging cases: Windows that resemble the object of interest and that therefore either contain the object (i.e., a positive sample) or a visual structure that can be easily confused with it (i.e., a hard negative sample).

Finally, since each weak hypothesis is usually associated to an elementary feature, AdaBoost is also often used for *feature selection*. In some cases, feature selection improves the *interpretability* of the classifier. For instance, in the Viola-Jones face detector, the first few Haar wavelets selected by AdaBoost usually capture semantically meaningful anatomical structures such as the eyes and the nose.

References

1. Friedman JH, Hastie T, Tibshirani R (2000) Additive logistic regression: a statistical view of boosting. *Ann Stat* 28(2): 337–374
2. Hastie T, Tibshirani R, Friedman J (2001) *The elements of statistical learning*. Springer, New York
3. Schapire RE (1990) The strength of weak learnability. *Mach Learn* 5(2):197–227
4. Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm. In: *Proceedings of the 13th international conference on machine learning*, Bari, 148–156
5. Vapnik V (1995) *The nature of statistical learning theory*. Springer, New York
6. Viola P, Jones M (2001) Robust real-time object detection. In: *Proceedings of IEEE workshop on statistical and computational theories of vision*, Vancouver, Canada
7. Papageorgiou CP, Oren M, Poggio T (1998) A general framework for object detection. In: *International conference on computer vision*, Bombay, pp 555–562
8. Sun Y, Li J, Hager W (2004) Two new regularized adaboost algorithms. In: *Machine learning and applications*, Louisville, pp 41–48
9. Schapire RE, Singer Y (1998) Improved boosting algorithms using confidence-rated predictions. In: *Computational learning theory*, Springer, New York, pp 80–91

Adaptive Boosting

► [AdaBoost](#)

Adaptive Gains

► [von Kries Hypothesis](#)

Affine Alignment

► [Affine Registration](#)

Affine Camera

Zhengyou Zhang

Microsoft Research, Redmond, WA, USA

Synonyms

[Affine projection](#)

Related Concepts

► [Perspective Camera](#); ► [Perspective Transformation](#);
► [Weak Perspective Projection](#)

Definition

An *affine camera* is a linear mathematical model to approximate the perspective projection followed by an ideal pinhole camera.

Background

Perspective projections give accurate models for a wide range of existing cameras, especially after calibration for a limited volume of workspace. However, the relationship from a 3D point to its 2D image point is nonlinear due to a scalar factor dependent of each individual point (see entry “► [Perspective Camera](#)” for details). *Affine cameras* are introduced to make the projection model more mathematically tractable. An affine camera model is a first-order approximation obtained from Taylor expansion of the perspective camera model around a reference point. The reference point can be any point, but it may be set to the centroid of the 3D points, which results in a more accurate approximation. There are three important instances of an affine camera when the camera’s intrinsic parameters are known: orthographic, weak perspective, and paraperspective projections. The reader is

referred to entry “►Weak Perspective Projection” for a detailed description of the affine camera model and its three instances.

Affine Invariants

Michael Werman

The Institute of Computer Science, The Hebrew University of Jerusalem, Jerusalem, Israel

Concept

Images of even the same object undergo various transformations depending on changes in the camera and its settings, the lighting, and the object itself. One of the ways to handle some of these changes, for tracking, search, and understanding images, is to use a description of the object that is oblivious to some of the above-mentioned transformations. Affine invariants are commonly used for this purpose.

There is a vast literature relating to affine invariants and only a small selection will be mentioned [4, 5, 11, 14, 17].

Affine Transformation

In order to have a property of an object that is invariant to an affine transformation, affine invariants can be used.

$$x \Rightarrow Lx + t$$

is an affine transformation where $x \in R^n$ is a vector, $L \in R^{n \times n}$ a matrix, and $t \in R^n$ a vector. L is a linear transformation, and t is a translation [2].

Affine transformations are used to describe different changes that images can undergo, such as an affine transformation of the (r, g, b) color values of an object under different lighting conditions or the transformation the shape of the image of an object undergoes when the camera and object are in different relative positions. The affine transformation in these cases does not necessarily model the exact physical distortion that is seen but often is a good approximation.

In order to have a property of an object that is invariant to an affine transformation, affine invariants can be used.

Affine Invariant

Let f be a function such that $f(a, b, \dots) = f(T(a), T(b), \dots)$ for any affine transformation T and $a, b, \dots \in R^n$, f is an affine invariant.

First example Let p, q, r be real numbers then an affine transformation is of the form $x \Rightarrow \alpha x + \beta$; it is easy to check that for any three numbers p, q, r , and affine transformation parameters α, β ,

$$\frac{p - q}{p - r} = \frac{(\alpha p + \beta) - (\alpha q + \beta)}{(\alpha p + \beta) - (\alpha r + \beta)}$$

[16, 18]

Second example We can generalize the first example to any dimension, d . The ratio of the volume of any two sets is an affine invariant (constant Jacobian, $|L|$) and is proportional to the $d + 1 \times d + 1$ determinant:

$$\begin{vmatrix} p & q & \dots & r \\ 1 & 1 & \dots & 1 \end{vmatrix}$$

Algebraic curves Not only points can be used to define affine invariance; other common examples are the parameters of curves, such as the equations of lines, conics, or other algebraic curves. For degree two curves in the plane, all ellipses are affinely equivalent.

Affine differential geometry There are affine invariant analogues of arc length and curvature. In general it is possible to find affine invariants involving points and their derivatives [3, 10, 13, 15].

Other parameters of the object Fourier coefficients and moments.

An affine transformation has $d^2 + d$ parameters, so in order to have a nontrivial affine invariant, one needs a function with more than that many arguments, where the result can be computed using algebraic elimination [9]. For example, two simplices in R^d have at least $d + 2$ different points which is $d^2 + 2d$ arguments.

In a certain sense, the number of independent invariants is the #(of parameters of a configuration) $-(d^2 + d)$.

There are other properties that are affine invariant, such as incidence, parallelism, centroids, barycentric coordinates, convexity, tangency, bi-tangency, Euler number, and connectivity.

If an objects area is A , then integrating by the area element divided by A is affine invariant, for example, $\frac{1}{A} \int_{Object} g(I(x, y)) dx dy$ is affine invariant, g being any function of the pixel's color.

Affine Invariant Feature Detection

There are a number of affine invariant feature detectors that find affine invariant local features in an image [7, 17]. One of the successes in recent practice of computer vision was the SIFT, a similarity invariant feature, and its extension to an affine invariant feature, ASIFT [6, 8].

Normalization

There are normalizations that can be done to an object that remove all/some of the possible affine variation.

The center of gravity is a linear invariant, so it is possible to translate the object so that the object's center is at a fixed coordinate, thus removing the translation term, changing the problem from one of finding an affine invariant to a linear invariant.

The Whitening transform canonically transforms a set of points using an affine transformation so that the average is 0 and the covariance matrix is the identity, I .

Grassmannians

The set of labeled points modulo affine transformations are isomorphic to Grassmannians, using this one can define a geometry of affine invariant point sets and, for example, measure the distance between *affine invariant point sets* [1].

Correspondence

Another thing to notice is that usually there needs to be some correspondence of the objects in order to use these invariants, namely, the order of the arguments of the function f needs to be known, and the way to overcome this problem is summing over all permutations making this function invariant both to

permutations. Another possibility is using permutation invariant features, for example, moments, which are built from summing over all the points.

Noise

Even though the group of affine transformation may be only a subgroup of the possible transformations, for example, projective transformations for an image of planar scene, being affinely invariant maybe an overkill as most of the radical transformations never really happen. Thus, noise can be explained by affine transformations making too many things close to each other.

References

1. Begelfor E, Werman M (2006) Affine invariance revisited. In: IEEE conference on computer vision pattern recognition (CVPR), New York
2. Gallier JH (2001) Geometric methods and applications for computer science and engineering. Springer, New York
3. Gotsman C, Werman M (1993) Recognition of affine transformed planar curves by extremal geometric properties. Int J Comput Geom Appl 3(2):183–202
4. Govindu VM, Werman M (2004) On using priors in affine matching. Image Vis Comput 22(14):1157–1164
5. Hartley RI, Zisserman A (2004) Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, Cambridge/New York. ISBN:0521540518
6. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60:91–110
7. Mikolajczyk K, Tuytelaars T, Schmid C, Zisserman A, Matas J, Schaffalitzky F, Kadir T, Van Gool L (2005) A comparison of affine region detectors. Int J Comput Vis 65(1):43–72
8. Morel J-M, Yu G (2009) Asift: a new framework for fully affine invariant image comparison. SIAM J Imaging Sci 2(2):438–469
9. Olver PJ (2009) Equivalence, invariants and symmetry. Cambridge University Press, Cambridge
10. Olver PJ (2009) Moving frames and differential invariants in centroaffine geometry. Technical report
11. Petrou M, Kadyrov A (2004) Affine invariant features from the trace transform. IEEE Trans Pattern Anal Mach Intell 26:30–44
12. Rahtu E, Salo M, Heikkil J (2005) Affine invariant pattern recognition using multiscale autoconvolution. IEEE Trans Pattern Anal Mach Intell 27:908–918
13. Reid M, Szendroi B (2005) Geometry and topology. Cambridge University Press, Cambridge/New York
14. Sapiro G (2000) Geometric partial differential equations and image analysis. Cambridge University Press, Cambridge

15. Simon U (2000) Affine differential geometry [Chapter 9]. In: Handbook of differential geometry, vol 1. North-Holland, pp 905–961
16. Sprinza J, Werman M (1994) Affine point matching. Pattern Recognit Lett 15(4):337–339
17. Tuytelaars T, Mikolajczyk K (2008) Local invariant feature detectors: a survey. Found Trends Comput Graph Vis 3: 177–280
18. Werman M, Weinshall D (1995) Similarity and affine invariant distances between 2D point sets. IEEE Trans Pattern Anal Mach Intell 17(8):810–814

Affine Projection

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Synonyms

[Affine camera](#)

Related Concepts

[►Affine Camera](#); [►Perspective Camera](#); [►Perspective Transformation](#); [►Weak Perspective Projection](#)

Definition

An *affine projection* is a linear mathematical model to describe the projection performed by an affine camera. See entry “[►Affine Camera](#)” for details.

Affine Registration

Kevin Köser
Institute for Visual Computing, ETH Zurich, Zürich, Switzerland

Synonyms

[Affine alignment](#)

Related Concepts

[►Affine Camera](#); [►Rigid Registration](#)

Definition

The goal of affine registration is to find the affine transformation that best maps one data set (e.g., image, set of points) onto another.

Background

In many situations data is acquired at different times, in different coordinate systems, or from different sensors. Such data can include sparse sets of points and images both in 2D and 3D, but the concepts generalize also to higher dimensions and other primitives. *Registration* means to bring these data sets into alignment, i.e., to find the “best” transformation that maps one set of data onto another, here using an *affine transformation*. For the sake of brevity, in this entry only the registration of two data sets is discussed, although approaches exist for finding consistent transformations that align more than two sets at once (e.g., [1]). While intuitively in 1D affine transformations compensate for scale and offset, in any dimension they can represent the first-order Taylor approximation (local linearization) for nonlinear functions.

For more complicated transformations, often a first affine registration is used as an initial solution that roughly aligns the data sets, followed by some (potentially local or nonlinear) search for more complicated parameters. In contrast to rigid registration, which allows only an offset and a rotation between the data sets, the affine model allows also for the full set of linear shape changes, including (nonisotropic) scale and shear. In particular, locally this approximates perspective effects or other nonlinear warps. On the other hand, affine registration uses still a single global transformation with a set of a few global parameters, which makes it mathematically easier to handle but also less powerful as compared to general nonrigid registration of deformable or articulated objects.

Theory

Two important cases can be distinguished for registration, the purely geometrical case, where two (finite) sets of points have to be registered, or the continuous/functional case, where the similarity of two functions must be maximized by an affine transformation of the functions' domain.

The Purely Geometrical Case

Let X and Y be two sets of points from \mathbb{R}^m and \mathbb{R}^n , respectively, and without loss of generality, it is assumed that $m \geq n$. For now it is assumed that the sets are of the same size and that there exists an (unknown) affine transformation, such that each element in X is mapped to an element in Y . Then, the goal of affine registration is to find this transformation, i.e., the matrix $A \in \mathbb{R}^{n \times m}$ and the offset $t \in \mathbb{R}^n$, such as to minimize an energy:

$$E_d = \sum_{x \in X} \min_{y \in Y} d_g(Ax + t, y) \quad (1)$$

Here $d_g(a, b)$ encodes the distance between a and b . Usually the points in X or Y are not available directly but only their noisy observations (e.g., when y are observed 2D projections of known 3D points and the parameters of the affine camera [2] are sought). In that case affine registration is based on these observations, and d_g should be chosen according to the noise distribution. If the sets are not of the same size or not each point in X corresponds to a point in Y , or in case there are gross errors in the data, still E_d can be minimized, e.g., with a robust cost function d_g to obtain some alignment between the sets [3]. Intuitively, in all these cases, A and t are sought which minimize the average distance for a transformed point of X to the closest point in Y .

In case bounds on A and t are known, a simple way to find an approximate solution is to sample the (continuous) space of all possible matrices A and offsets t , but this is usually computationally intractable. If it is known which points in X correspond to which others in Y , i.e., there is a set of pairs (x_i, y_i) , the energy can be rewritten as

$$E_c = \sum_{x_i \in X} d_g(Ax_i + t, y_i) \quad (2)$$

In case $d_g(a, b) = \|(a - b)\|_2^2$, this can be solved explicitly. A contains mn unknowns, t contains n unknowns, and from each correspondence n observations can be obtained, so from counting it is clear that at least $m+1$ correspondences are required to uniquely determine a solution (assuming they are in a general configuration and not, e.g., only on a line). In case the correspondences contain gross errors (mismatches, outliers), then the solution can be obtained using robust estimation techniques such as least median of squares [4] or RANSAC [5].

In case correspondences are not known, but that the data sets are already approximately aligned, it can be assumed that the closest neighbors are in correspondence. In this case the problem can be solved by using the iterative closest points (ICP) method [6–8]. Locally, for each point the offset to the closest point in the other data set is computed, and by collecting all the local offsets, a global affine transformation is estimated that aligns these sets (in a least squares sense according to Eq. 2). This transformation is applied to recalculate the nearest neighbor correspondences and the estimation step is repeated with these. In [9] efficient implementations and practical aspects of ICP have been studied. However, since ICP relies on local neighborhoods, it cannot cope well with situations where initially close points must be moved far away.

A technique proposed for this is based on normalization (as suggested, e.g., by [10]): Here, the means of the data sets are computed individually and t is defined as the difference of the means. Then, for each data set the (unbiased) covariance is computed and the matrices A_X, A_Y are searched that bring the respective point distributions to a unit covariance matrix (whitening of covariance). The two data sets now only differ by an orthogonal matrix that may be obtained by sampling or finding other characteristics in the data (cf. also [11]). However, this normalization approach assumes that the data sets overlap fully.

In general, similar concepts can be applied as for rigid registration, however, with a slightly different set of parameters.

The Continuous/Functional Case

Let I and J be functions (images) from \mathbb{R}^n to \mathbb{R}^d , assigning some color value to each position, typically in the plane or in space. The function value will be referred to as the color hereafter, regardless of its

physical meaning. In this case the affine registration can be stated as the minimization of an energy:

$$E = \int_x d_c(I(Ax + t), J(x))dx \quad (3)$$

Here, d_c is a distance between two colors that should be chosen according to the expected measurement uncertainty of the colors. Very similar to the discrete case, a naive strategy to minimize this energy would be sampling; however, this is again computationally expensive and requires bounds on the parameters.

A possible solution is to compute local image features, such as corners, blobs, and so on (cf. to [12]), and – if possible – find correspondences among these features. There has been a huge body of work to particularly define features that can be detected reliably with affine changes of image coordinates. A comparison can be found in [13]. Given those the discrete methods of the previous section can be used to find an alignment A, t .

Afterwards, or in case A and t were approximately known from the beginning and if I and J are smooth, then the alignment can be performed by local linearization as proposed by Lucas and Kanade [14]. The assumption is that locally the image can be represented by its first-order Taylor approximation, i.e., the local color and the gradient:

$$I(x) \approx I(x_0) + \frac{\partial I}{\partial x} \underbrace{(x_0 - x)}_{\Delta x} \quad (4)$$

Consequently, given two almost aligned images, at position x_0

$$\underbrace{J(x_0) - I(x_0)}_{\Delta I_0} \approx \left. \frac{\partial I}{\partial x} \right|_{x_0} \Delta x \quad (5)$$

Stacking r of these equations on top of each other, an equation system is obtained:

$$\begin{pmatrix} \Delta I_0 \\ \Delta I_1 \\ \dots \\ \Delta I_{r-1} \end{pmatrix} \approx \begin{pmatrix} \left. \frac{\partial I}{\partial x} \right|_{x_0} \\ \left. \frac{\partial I}{\partial x} \right|_{x_1} \\ \dots \\ \left. \frac{\partial I}{\partial x} \right|_{x_{r-1}} \end{pmatrix} \Delta x \quad (6)$$

This can be solved in a least squares sense to obtain Δx .

Similar to the local image gradient with respect to position, also the partial derivatives with respect to affine transformation parameters can be computed [14]. In this case, Δx (and also ∂x) contains $n(n + 1)$ parameters (n^2 for the linear shape change and n for the offset). At least $n(n + 1)$ equations are required to uniquely solve this. Often, to increase the basin of convergence, coarse-to-fine registration is applied. This can be implemented using image pyramids in case the uncertainty lies mostly in the offset parameters or by propagation of the affine parameter uncertainty to position uncertainty in the image and appropriate smoothing [15]. After having applied the estimated update Δx on the parameters, the steps can be applied repeatedly to register the two images. In [16] Baker and Matthews compare different formulations of such iterative, gradient-based image alignment, particularly the question of how to compose and parameterize the warps across multiple iterations.

On top of transformations on the domain of the images, often the two images differ in target (e.g., the colors of corresponding positions are related by some brightness offset), in which case also parameters for the change of color need to be estimated. In case the corresponding image colors are only statistically related but no explicit transformation model between colors is known, the concept of mutual information [17] might be used, where the entropy of the joint color histogram is minimized. An overview of image-based alignment can be found in [18, 19].

Application

Affine cameras [2] approximate a real camera by an affine mapping of 3D points to 2D image coordinates. For tracking local regions through videos, it has been shown [20] that keeping track of the affine deformations of local regions can help detecting tracking failures. Such affine warps, or those implied by correspondences of affine features between different images, represent the local linearization of a potentially nonlinear image warp (e.g., of perspective effects). If the structure of this nonlinear warp is known, the affine registration can allow inferring the global warp directly [21] or provide more constraints than just using the position of a region correspondence.

In general, affine registrations often provide a reasonable solution to align mean, linear shape, and orientation of data without making the transformation too problem specific, or the affine solution can serve as a basis for further more advanced alignment. Furthermore, multiple independent or coupled local affine registrations can help in registering articulated or deformable models.

References

1. Eggert DW, Fitzgibbon AW, Fisher RB (1998) Simultaneous registration of multiple range views for use in reverse engineering of CAD models. *Comput Vis Image Underst* 69(3): 253–272
2. Mundy JL, Zisserman A (eds) (1992) Geometric invariance in computer vision. MIT, Cambridge
3. Fitzgibbon AW (2003) Robust registration of 2d and 3d point sets. *Image and Vis Computing* 21:1145–1153
4. Rousseeuw PJ (1984) Least median of squares regression. *J Am Stat Assoc* 79(388):871–880
5. Fischler M, Bolles R (1981) RANDom SAMpling Consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun ACM* 24(6):381–395
6. Besl PJ, McKay ND (1992) A method for registration of 3-d shapes. *IEEE Trans Pattern Anal Mach Intell* 14:239–256
7. Zhang Z (1994) Iterative point matching for registration of free-form curves and surfaces. *Int J Comput Vis* 13:119–152
8. Feldmar J, Ayache N (1994) Rigid and affine registration of smooth surfaces using differential properties. In: Eklundh JO (ed) European conference on computer vision (ECCV '94). Volume 801 of lecture notes in computer science. Springer, Berlin/Heidelberg, pp 396–406. doi:10.1007/BFb0028371
9. Rusinkiewicz S, Levoy M (2001) 3-D digital imaging and modeling. Proceedings Third International Conference on, Efficient variants of the ICP algorithm, 145–152. doi:10.1109/IM.2001.924423
10. Obdrzálek S, Matas J (2002) Local affine frames for image retrieval. Proceedings of the International Conference on Image and Video Retrieval, CIVR '02. Springer, London, pp 318–327
11. Ho J, Yang MH (2011) On affine registration of planar point sets using complex numbers. *Comput Vis Image Underst* 115(1):50–58
12. Tuytelaars T, Mikolajczyk K (2008) Local invariant feature detectors: A survey. *Found Trends Comput Graph Vis* 3(3):177–280
13. Mikolajczyk K, Tuytelaars T, Schmid C, Zisserman A, Matas J, Schaffalitzky F, Kadir T, van Gool L (2005) A comparison of affine region detectors. *Int J Comput Vis* 65(1–2):43–72
14. Lucas BD, Kanade T (1981) An iterative image registration technique with an application to stereo vision. Proceedings of the 7th international joint conference on Artificial intelligence, 2(6):674–679
15. Köser K, Koch R (2008) Exploiting uncertainty propagation in gradient-based image registration. *Proc Br Mach Vis Conf* 83–92
16. Baker S, Matthews I (2004) Lucas-kanade 20 years on: A unifying framework. *Int J Comput Vis* 56(1): 221–255
17. Viola PA III, WMW (1997) Alignment by maximization of mutual information. *Int J Comput Vis* 24(2):137–154
18. Zitov B, Flusser J (2003) Image registration methods: a survey. *Image Vis Comput* 21:977–1000
19. Szeliski R (2006) Image alignment and stitching: A tutorial. *Found Trends Comput Graph Comput Vis* 2
20. J Shi, Tomasi C (1994) Good features to track. *IEEE Conf Comput Vis Pattern Recognit* 593–600
21. Köser K, Koch R (2008) Differential spatial resection – pose estimation using a single local image feature. *Eur Conf 278 Comput Vis (LNCS 5302/5305)*, 312–325

Affordances

► [Affordances and Action Recognition](#)

Affordances and Action Recognition

James Bonaiuto

California Institute of Technology, Pasadena,
CA, USA

Synonyms

[Action Recognition](#); [Affordances](#)

Related Concepts

► [Gait Recognition](#); ► [Gesture Recognition](#);
► [Multi-camera Human Action Recognition](#)

Definition

Affordances are opportunities for action that are directly perceivable in an organism's environment without higher-level cognitive functions. Action recognition is the result of mapping an observed action onto an internal motor or semantic representation.

Background

Affordances are defined by Gibson [1] as opportunities for action that are directly perceivable without the need for higher-level cognitive functions such as object recognition. The concept of affordances for action has generated significant interest in the computer vision and robotics community. More recently, links between this concept and that of action recognition have been explored, suggesting that the two may share common mechanisms.

Affordances. In robotics, early use of the term affordances dealt with the extraction of features from the visual environment that signal the possibility for action. However, neural models of vision and action have suggested a more active role for affordances in monitoring the ongoing control of an action and in predicting the effects of an action on a target object [2–4].

Because the affordances available to an agent depend on its embodiment, their representation needs to be grounded in its perceptual and motor repertoire. Most approaches therefore learn affordances through a stage of motor babbling with objects, where different actions are randomly attempted on an object while recording the initial perception of the object and any effects that the action has on it.

Typically, affordance formalisms involve storing entity, action, and effects relations, allowing the agent to associate objects with possible actions to perform on them and the effects of these actions. This formulation is therefore useful for planning by allowing an agent to search for an action that will produce a desired effect. More powerful formalisms store relations between entity, action, and effect equivalence classes [5]. These allow an agent to filter incoming sensory data to extract invariant affordance cues and to generalize learned affordances to novel objects.

Action Recognition. Action recognition can take place on several hierarchical scales of organization. At the lowest level, this consists of recognizing basic actions like reaching and grasping. Recognition of these actions can be used to recognize more complex actions and sequences of actions such as putting sugar in a cup. The highest level of action recognition involves inferring the goals, intentions, or the task of the observed agent (e.g., making coffee).

The two main approaches to low- and mid-level action recognition in computer vision are model-based and template-based or holistic. Model-based action

recognition involves tracking body parts using a parametric model of the body kinematics. Body parts are typically recognized by low-level features, and then geometric constraints from the body model are imposed on them. Actions are typically represented in joint space. Template-based action recognition directly models actions using spatiotemporal features and represents actions as spatio-temporal shapes in 3 or 4d space. These features could be static features based on edges and limb shapes, dynamic features based on optical flows, or space-time volumes.

Neural models of low-level action recognition commonly use self-observation during action performance to associate the motor representation used to generate an action with the visual representation elicited by observation of it [6, 7]. An object-centered representation is hypothesized to exist for feedback-based control of actions, which causes self-observation during performance of an action and observation of another agent performing the same action to elicit similar visual representations [6, 8]. This link between affordance-activated motor representations and visual representations of observed actions is thought to ground understanding of the goals and intentions of other agents.

In order to recognize more complex actions and action sequences, the results of low-level action recognition are typically combined using some sequence parsing technique. Previous approaches have used hidden Markov models (HMMs) [9], probabilistic parsing using context-free grammars [10], and Bayesian inference on graphical models [11]. Models of action recognition at the highest level use lower level motor mechanisms in a simulation mode to infer the intention of the observed actor [12] or Bayesian inverse reinforcement learning to infer the reward function, or task, of a demonstration [13].

Theory

The close relationship between affordance and action recognition is made apparent by the related ideas that affordances play an active role in online control of an action and that the effects of an action on an object are included in the affordance representation. The same mechanisms used to monitor and control an agent’s ongoing action can therefore also be leveraged to recognize the same action performed

by another agent. Mechanisms for affordance perception and action recognition can interact through several ways. Recognition of an action performed by another agent can allow an agent to learn affordance cues or the effects of an affordance by observation. Observation of the effects of an affordance can aid in action recognition when the action itself appears ambiguous. Finally, extraction of affordance cues can similarly aid in action recognition by narrowing the space of possible actions.

Several systems use previously learned affordance information to recognize observed actions by searching for an action that is afforded by the observed object [14] or that would have produced the observed effect on the object [11, 13]. In both cases, the agent interprets the observed actions of other agents in terms of its own affordance representations. This can improve recognition accuracy when the details of the motion of the body parts during a particular is too difficult to disambiguate, but the effects of the action on the object are clearly perceivable.

Given a system for recognizing low-level body part movements, the affordances of objects can be approximately learned by observation [15]. This requires that the robot have a similar embodiment to that of the observed agent or a mechanism to map from observed body parts onto its own body. In this case, the features that indicate an affordance and the effects of the observed action can easily be recorded. If the observed action is novel and cannot be recognized at a high level, the recognized low-level subcomponents can be used to guide subsequent attempts at reproducing it by trying to replicate its effects on an object.

References

1. Gibson JJ (1966) The senses considered as perceptual systems. Houghton Mifflin, Boston
2. Fagg AH, Arbib MA (1998) Modeling parietal-premotor interactions in primate control of grasping. *Neural Netw* 11(7–8):1277–1303
3. Paletta L, Fritz G, Kintzler F, Irran J, Dorffner G (2007) Learning to perceive affordances in a framework of developmental embodied cognition. *IEEE International Conference on Development and Learning*, London, pp 110–115
4. Arbib MA (1997) From visual affordances in monkey parietal cortex to hippocampoparietal interactions underlying rat navigation. *Phil Trans R Soc Lond B* 352(1360): 1429–1436
5. Sahin E, Cakmak M, Dogar MR, Ugur E, Ucoluk G (2007) To afford or not to afford: a new formalization of

- affordances toward affordance-based robot control. *Adapt Behav* 15(4):447–472
6. Ozttop E, Arbib MA (2002) Schema design and implementation of the grasp-related mirror neuron system. *Biol Cybern* 87(2):116–140
 7. Metta G, Sandini G, Natale L, Craighero L, Fadiga L (2006) Understanding mirror neurons: a bio-robotic approach. *Interact Stud* 7(2):197–232
 8. Bonaiuto J, Rosta E, Arbib MA (2007) Extending the mirror neuron system model, I: audible actions and invisible grasps. *Biol Cybern* 96:9–38
 9. Yamato J, Ohya J, Ishii K (1992) Recognizing human action in time-sequential images using hidden Markov model. In: *Proceedings of computer vision and pattern recognition (CVPR)*, Champaign, IL, pp 379–385
 10. Bobick AF, Ivanov YA (1998) Action recognition using probabilistic parsing. In: *Proceedings of computer vision and pattern recognition (CVPR)*, Santa Barbara, pp 196–202
 11. Gupta A, Davis LS (2007) Objects in action: an approach for combining action understanding and object perception. In: *Proceedings of computer vision and pattern recognition (CVPR)*, Minneapolis, pp 1–8
 12. Ozttop E, Wolpert D, Kawato M (2005) Mental state inference using visual control parameters. *Cogn Brain Res* 22:129–151
 13. Lopes M, Melo FS, Montesano L (2007) Affordance-based imitation learning in robots. In: *IEEE/RSJ international conference on intelligent robots and systems*, San Diego, CA, pp 1015–1021
 14. Moore DJ, Essa IA, Hayes MH (1999) Exploiting human actions and object context for recognition tasks. *Proc Int Conf Comput Vis (ICCV)* 1:80–86
 15. Kjellstrom H, Romero J, Krägic D (2011) Visual object-action recognition: inferring object affordances from human demonstration. *Comput Vis Image Underst* 115(1): 81–90

Algebraic Curve

Bo Zheng

Computer Vision Laboratory, Institute of Industrial Science, The University of Tokyo, Meguro-ku, Tokyo, Japan

Synonyms

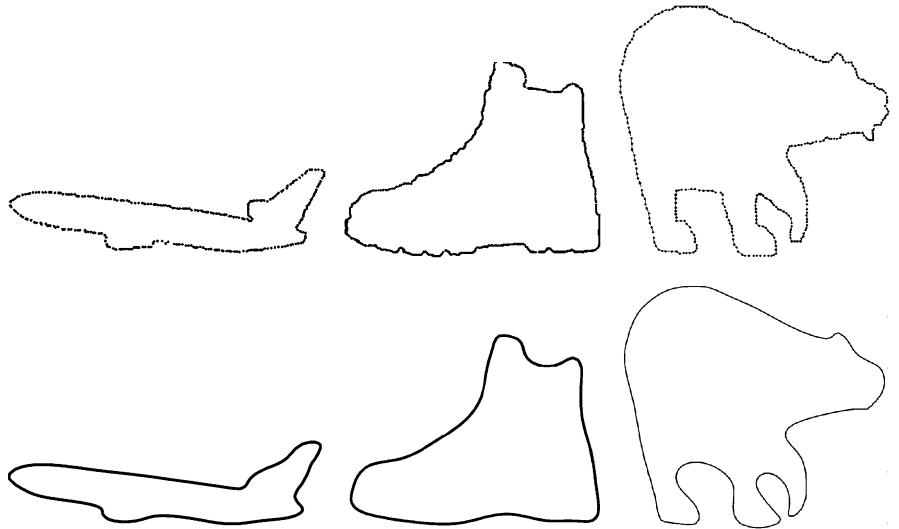
[Implicit polynomial curve](#)

Related Concepts

► [Algebraic Surface](#)

Algebraic Curve, Fig. 1

Examples of algebraic curves.
 Top row: original data sets;
 bottom row: represented
 algebraic curves

**Definition**

An *algebraic curve* is a curve determined by a 2-D *implicit polynomial* (IP) of degree n :

$$\begin{aligned} f_n(\mathbf{x}) &= \sum_{0 \leq i, j; i+j \leq n} a_{ij} x^i y^j \\ &= (\underbrace{1 \ x \ \dots \ y^n}_{\mathbf{m}^T})(\underbrace{a_{00} \ a_{10} \ \dots \ a_{0n}}_{\mathbf{a}})^T = 0, \quad (1) \end{aligned}$$

where $\mathbf{x} = (x, y)^T$ is the coordinate of a point on a curve. That is, the curve is always represented by f_n 's zero level set: $\{\mathbf{x} | f_n(\mathbf{x}) = 0\}$. The polynomial function is usually denoted by an inner product between two vectors: monomial vector \mathbf{m} and coefficient vector \mathbf{a} . For the entries in these vectors, indices $\{i, j\}$ can be arranged in different orders, such as *lexicographical order* or *inverse lexicographical order*. In addition, the *homogeneous binary polynomial* of degree r in x and y , $\sum_{i+j=r} a_{ij} x^i y^j$, is called the r -th degree form of the IP. The form of degree n is called the *leading form*. The degree of an algebraic curve is the degree of the polynomial (e.g., n). An algebraic curve of degree 2 is called a conic, degree 3 a cubic, degree 4 a quartic, and so on.

Background

In computer vision, representing 2-D data sets with algebraic curves has been studied extensively for the

past three decades. It is attractive for vision applications due to its applicability to object recognition, pose estimation, and registration. In contrast to the curves represented by other functions such as splines, Fourier, rational Gaussian, and radial basis function, algebraic curve is superior in such areas as fast fitting, few parameters, algebraic/geometric invariants, and robustness against noise and occlusion. Algebraic curve is also capable of modeling non-star shapes, open curves, curves that contain gaps, and unordered curve data. However, algebraic curve representation still suffers from some major issues such as the lack of local accuracy and global stability when representing a complex 2-D shape (see [7]). Figure 1 shows some example algebraic curves successfully used to represent closed 2-D curves.

Application and Theory

Algebraic curve representation is mainly attractive for vision applications such as fast shape registration or pose estimation [3, 6, 8, 9, 12] and recognition [2, 4–6, 9–11]. To achieve these purposes, many efforts have been made in three topics: curve fitting, algebraic/geometric invariants, and curve registration. The first is about solving the problem of accurately and stably fitting an algebraic curve to a complex shape, the second is on extracting algebraic or geometric invariants from the algebraic curve representing a shape, and the third concerns estimating

the Euclidean transformation(s) between two or more algebraic curves representing different instances of the same shape.

Curve Fitting. There have been great improvements concerning algebraic curve fitting with its increased use during the late 1980s and early 1990s [8]. Recently, new robust and consistent fitting methods like 3L fitting [1], gradient-one fitting with Rigid regression [7], and degree-adaptive fitting [13] that are suitable for vision applications have been introduced.

Algebraic/Geometric Invariants. The main advantage of the use of algebraic curves for recognition is the existence of algebraic/geometric invariants, which are functions of the polynomial coefficients that do not change after a coordinate transformation. To some major contributions, the global Euclidean invariants are found by Taubin and Cooper [9], Teral and Cooper [6], and Keren [2], which can be expressed as simple explicit functions of the IP coefficients. Wolovich et al. [11] also introduced a set of invariants from covariant conic decompositions of implicit polynomials.

Curve Registration. In prior literatures [6, 9], global shape registration is performed through a single (non-iterative) computation using the central and oriented information extracted from the polynomial coefficients of two algebraic curves. Recently, an iterative method for aligning partially matched curves that uses the distance measurement of the polynomial gradient field together with a fast polynomial transformation has been introduced [12].

References

1. Blane M, Lei ZB, Cooper DB (2000) The 3L algorithm for fitting implicit polynomial curves and surfaces to Data. *IEEE Trans Pattern Anal Mach Intell* 22(3):298–313
2. Keren D (1994) Using symbolic computation to find algebraic invariants. *IEEE Trans Pattern Anal Mach Intell* 16(11):1143–1149
3. Keren D, Cooper D, Subrahmonia J (1994) Describing complicated objects by implicit polynomials. *IEEE Trans Pattern Anal Mach Intell* 16(1):38–53
4. Oden C, Ercil A, Buke B (2003) Combining implicit polynomials and geometric features for hand recognition. *Pattern Recognit Lett* 24(13):2145–2152
5. Subrahmonia J, Cooper DB, Keren D (1996) Practical reliable bayesian recognition of 2D and 3D objects using implicit polynomials and algebraic invariants. *IEEE Trans Pattern Anal Mach Intell* 18(5):505–519
6. Tarel J, Cooper DB (2000) The complex representation of algebraic curves and its simple exploitation for pose estimation and invariant recognition. *IEEE Trans Pattern Anal Mach Intell* 22(7):663–674
7. Tasdizen T, Tarel J-P, Cooper DB (2000) Improving the stability of algebraic curves for applications. *IEEE Trans Imag Proc* 9(3):405–416
8. Taubin G (1991) Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE Trans Pattern Anal Mach Intell* 13(11):1115–1138
9. Taubin G, Cooper DB (1992) Symbolic and numerical computation for artificial intelligence, chapter 6, Computational mathematics and applications. Academic, London
10. Unel M, Wolovich WA (2000) On the construction of complete sets of geometric invariants for algebraic curves. *Adv Appl Math* 24:65–87
11. Wolovich WA, Unel M (1998) The determination of implicit polynomial canonical curves. *IEEE Trans Pattern Anal Mach Intell* 20(10):1080–1090
12. Zheng B, Ishikawa R, Oishi T, Takamatsu J, Ikeuchi K (2009) A fast registration method using IP and its application to ultrasound image registration. *IPSJ Trans Comput Vis Appl* 1:209–219
13. Zheng B, Takamatsu J, Ikeuchi K (2010) An adaptive and stable method for fitting implicit polynomial curves and surfaces. *IEEE Trans Pattern Anal Mach Intell* 32(3):561–568

Algebraic Surface

Bo Zheng

Computer Vision Laboratory, Institute of Industrial Science, The University of Tokyo, Meguro-ku, Tokyo, Japan

Synonyms

[Implicit polynomial surface](#)

Related Concepts

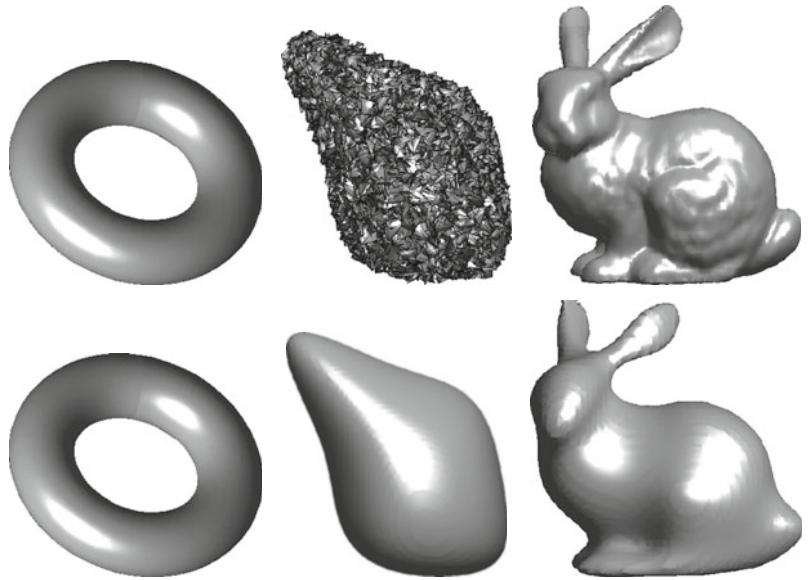
► [Algebraic Curve](#)

Definition

Similar to an algebraic curve, an *algebraic surface* is determined by a 3-D *implicit polynomial* (IP) of degree n :

Algebraic Surface, Fig. 1

Examples of algebraic surfaces. Top row: original 3-D data sets of torus, simple shape with noise, and bunny; bottom row: resulting algebraic surface fits of degree 4, 4, and 8, respectively



$$\begin{aligned}
 f_n(\mathbf{x}) &= \sum_{0 \leq i,j,k; i+j+k \leq n} a_{ijk} x^i y^j z^k \\
 &= (\underbrace{1 \ x \ y \ z \dots z^n}_{\mathbf{m}^T})(\underbrace{a_{000} \ a_{100} \ a_{010} \ a_{001} \ \dots \ a_{00n}}_{\mathbf{a}})^T \\
 &= 0,
 \end{aligned} \tag{1}$$

where $\mathbf{x} = (x, y, z)$ is a 3-D point on a surface, that is, the surface is always represented by f_n 's zero level set: $\{\mathbf{x} | f_n(\mathbf{x}) = 0\}$. The polynomial function can be denoted by an inner product of two vectors: monomial vector \mathbf{m} and coefficient vector \mathbf{a} . For the entries in these vectors, indices $\{i, j, k\}$ can be arranged in different orders, such as *lexicographical order* or *inverse lexicographical order*. In addition, the *homogeneous binary polynomial* of degree r in x , y , and z , $\sum_{i+j+k=r} a_{ijk} x^i y^j z^k$, is called the *r -th degree form* of the IP. The n -th form is also called *leading form*. The degree of algebraic surface is the degree of polynomial: n . An algebraic surface of degree 2 is called a quadric surface, degree 3 a cubic surface, degree 4 a quartic surface, and so on.

Background

In computer vision, representing 3-D surface data sets with algebraic surfaces has been also well studied. It is attractive for vision applications such as 3-D object

recognition, pose estimation, and registration. In contrast to other surfaces represented by the functions such as Splines, Fourier, Rational Gaussian, and radial basis function, algebraic surface is superior in such areas as fitting efficiency, few parameters, convenience for calculating algebraic/geometric invariants, and robustness against noise and occlusion. Algebraic surface is also capable of modeling nonstar shapes, open curves, curves that contain gaps, and unordered curve data. However, algebraic surface representation still suffers from some major issues such as the lack of accuracy and stability when representing a complex 3-D shape. Figure 1 shows some examples of algebraic surfaces representing for 3-D surface data sets.

Application and Theory

Algebraic surface representation is mainly attractive for vision applications such as 3-D object registration or pose estimation [2, 4, 8, 10, 11, 13] and recognition [3, 6, 8, 11]. To achieve those purposes, many efforts have been made in three topics: surface fitting, algebraic/geometric invariants, and 3-D object registration. The first topic faces the problem of how to fit an algebraic surface to a complex 3-D shape accurately and stably; the second topic focuses on the problem of how to extract algebraic or geometric invariants from a 3-D shape-representing polynomial; and the third topic concentrates on the task of how to estimate

the rigid transformation relationship between two algebraic surfaces representing the same object in different positions.

Surface Fitting

There have been great improvements concerning algebraic surface fitting with its increased use during the late 1980s and early 1990s [10]. Recently, new robust and consistent fitting methods such as 3L fitting [1], gradient-one fitting with Rigid regression [5, 9], and degree-adaptive fitting [14] have been proposed to make the algebraic surface representation more feasible for vision applications.

Algebraic/Geometric Invariants

The main advantage of algebraic surfaces for recognition is the existence of algebraic/geometric invariants, which are functions of the polynomial coefficients that do not change after a coordinate transformation. The algebraic/geometric invariants that are found by Taubin and Cooper [11], Teral and Cooper [8], and Keren [3] are global invariants and are expressed as simple explicit functions of the coefficients. Another set of invariants that have been mentioned by Wolovich et al. is derived from the covariant conic decompositions of implicit polynomials [12].

3-D Object Registration

In prior literatures such as [7, 11], the global shape registration is performed through single (non-iterative) computation after obtaining the central and oriented information extracted from polynomial coefficients. An iterative method in [13] is proposed by using the distance metric generated from polynomial gradient field and fast polynomial coefficient transformation.

References

1. Blane M, Lei ZB, Cooper DB (2000) The 3L algorithm for fitting implicit polynomial curves and surfaces to data. *IEEE Trans Pattern Anal Mach Intell* 22(3):298–313
2. Forsyth D, Mundy JL, Zisserman A, Coelho C, Heller A, Rothwell C (1991) Invariant descriptors for 3D object recognition and pose. *IEEE Trans Pattern Anal Mach Intell* 13(10):971–992
3. Keren D (1994) Using symbolic computation to find algebraic invariants. *IEEE Trans Pattern Anal Mach Intell* 16(11):1143–1149
4. Keren D, Cooper D, Subrahmonia J (1994) Describing complicated objects by implicit polynomials. *IEEE Trans Pattern Anal Mach Intell* 16(1):38–53
5. Sahin T, Unel M (2005) Fitting globally stabilized algebraic surfaces to range data. *Proc IEEE Conf Int Conf Comp Visi* 2:1083–1088
6. Subrahmonia J, Cooper DB, Keren D (1996) Practical reliable bayesian recognition of 2D and 3D objects using implicit polynomials and algebraic invariants. *IEEE Trans Pattern Anal Mach Intell* 18(5):505–519
7. Tarel J, Cooper DB (2000) The complex representation of algebraic curves and its simple exploitation for pose estimation and invariant recognition. *IEEE Trans Pattern Anal Mach Intell* 22(7):663–674
8. Tarel J-P, Civi H, Cooper DB (1998) Pose estimation of free-form 3D objects without point matching using algebraic surface models. In: Proceedings of IEEE Workshop Model Based 3D Image Analysis, Mumbai, pp 13–21
9. Tasdizen T, Tarel J-P, Cooper DB (2000) Improving the stability of algebraic curves for applications. *IEEE Trans Imag Process* 9(3):405–416
10. Taubin G (1991) Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE Trans Pattern Anal Mach Intell* 13(11):1115–1138
11. Taubin G, Cooper DB (1992) Symbolic and numerical computation for artificial intelligence, chapter 6. In: Donald BR, Kapur D, Mundy JL (eds) Computational Mathematics and Applications. Academic, London
12. Wolovich WA, Unel M (1998) The determination of implicit polynomial canonical curves. *IEEE Trans Pattern Anal Mach Intell* 20(10):1080–1090
13. Zheng B, Ishikawa R, Oishi T, Takamatsu J, Ikeuchi K (2009) A fast registration method using IP and its application to ultrasound image registration. *IPSJ Trans Comput Vision Appl* 1:209–219
14. Zheng B, Takamatsu J, Ikeuchi K (2010) An adaptive and stable method for fitting implicit polynomial curves and surfaces. *IEEE Trans Pattern Anal Mach Intell* 32(3): 561–568

Analytic Reflectance Functions

► Reflectance Models

Animat Vision

Dana H. Ballard

Department of Computer Sciences, University of Texas at Austin, Austin, TX, USA

Synonyms

Active vision; Purposive vision

Definition

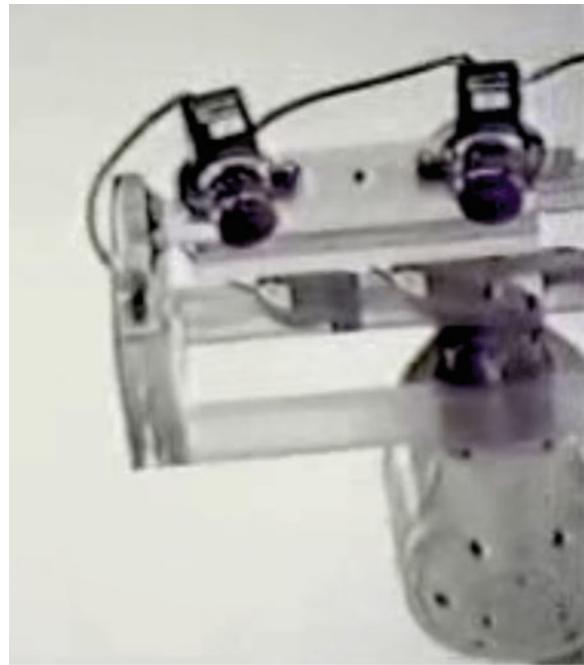
Animat vision is the computational study of the visual systems used by animals, with special attention to the binocular systems used by humans. For human vision, the goal is to show how the characteristics of the human eye movement system can be used to make the computation of needed information more efficient.

Background

The field of computer vision was given an enormous impetus by the publication of Rosenfeld's seminal book *Picture Processing by Computer* [28] in 1969, but in the 1970s the research focus shifted to human vision with the exciting new formulations of *early vision* that recognized that the human visual system devoted enormous resources to extracting cues such as binocular disparity, color, and motion from their composite representation in the initial image. Two groups were especially influential: the group at MIT headed by David Marr and Tomas Poggio [21] and the group at SRI headed by Harry Barrow and Martin Tanenbaum [8]. David Marr in particular had an enormous effect on the field and his book, *Vision*, is a classic [20].

While the early vision paradigm had a wonderful impact of defining computation in vision, by the early 1980s it was apparent that the computations defined on static images were mathematically delicate and could only be tamed with exceptional ingenuity. Thus the idea evolved that perhaps a moving camera, with known movement parameters, would help. An early effort was undertaken at MIT, but the first complete working system was built at the University of Pennsylvania by Ruzena Bajcsy who coined the term *active perception* to describe it [5]. That system was unveiled at a computer vision conference in northern Michigan run by Avi Kak and had an instantaneous acceptance among the researchers present.

Very shortly afterwards, Christopher Brown and the author built a similar system that had a significant advantage. Brown was tracking video processing pipeline computers and realized that this evolving computer architecture, when combined with a



Animat Vision, Fig. 1 The University of Rochester real-time servo-driven robotic “head” mounted on its large PUMA “body.” Similar systems were built at KTH in Stockholm, Carnegie Mellon University, MIT, the University of Pennsylvania, as well as many other places

servo-driven binocular camera system, would allow the new computations to be realized in real time. The complete system is shown in Fig. 1. Subsequently the appearance of video-rate graphics cards capable of doing real-time image operations served to spur progress. Originally driven by the needs of the computer games industry, researchers quickly realized that now a large amount of the expensive visual calculations could be done in real time. The net result is that animat/active vision moved to lower-cost mobile robotic platforms with the result that robots using a moving cameras on mobile platforms are now commonplace.

Along the mainstream path of robotic animats, Rodney Brooks at MIT, perhaps inspired by Shimon Ullman’s concept of *visual routines*, realized that the jobs that vision had to do now became preeminent and built MIT’s humanoid robot *Cog* to focus on task-based vision. The particular architecture advocated may not have caught on, but the point was made and the system has been enormously influential. Two

diverse communities – robotics and psychology – have been working on cognitive architectures for managing complex tasks that take a more integrated approach to vision and action, and both have recognized that the ultimate model architecture will have a hierarchical structure, e.g., [3, 11, 12, 16, 23]. Robotics researchers have gravitated to a three-tiered structure that models strategic, tactical, and detailed levels in complex behavior [10].

Theory

Animat vision, like its larger cousin active vision, is a paradigm with a huge number of important papers outstanding, with the consequence that it is only possible to provide the barest of outlines here. The interested reader is referred to some of the early papers [2, 6, 32]. Here we will demonstrate the impact on the calculation of early vision and introduce some more recent developments.

Consequences for Early Vision

Consider the problem of computing just one of the useful early vision representations, that of *optic flow*. Three-dimensional motion due to a moving observer induces the projection of two-dimensional motion on the retina. If the time-varying image function $f(x, y, t)$ represents only this effect, then the differential equations that represent the relationship between optic flows $(u(x, y, t), v(x, y, t))$ can be related to changes in photometric intensities $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial t}$ with the equation, captured at the sensor to the image intensity M by

$$\frac{\partial f}{\partial x}u + \frac{\partial f}{\partial y}v + \frac{\partial f}{\partial t} = 0 \quad (1)$$

The conundrum of early vision can be easily apparent: At each point in the image x, y at time t , there is a single equation in two unknowns u, v . Poggio famously characterized this as an “ill-posed problem.” A plethora of solution methods were tried, but they almost all involved integrating information across the optic array, a very delicate process. In contrast, moving the camera view point immediately solves this problem. If the motion is under the control

of the human observer, then, say for the case of horizontal motion, to a first approximation one can assume a reliable estimate for u , and of course the system reduces to a well-behaved two equations in two unknowns. This kind of simplification surfaces again and again in early vision, and many novel instances of this kind of constraint still remain to be discovered.

The Importance of Eye Fixations

Yarbus’s original work in gaze recordings [37] in the 1950s and 1960s revealed the enormous importance of gaze in revealing the underlying structure of human cognition. From this perspective, it is somewhat surprising that the first significant computational theory of vision [20] postponed the study of gaze as well as any influence of cognition of the extraction of information from the retinal array. In his “principle of least commitment,” Marr argued the case for the role of the cortex in building elaborate goal-independent descriptions of the physical world. Perhaps as a consequence, when researchers took on the task of defining the mechanisms for directing gaze deployment, these turned out to be predominantly image based [18, 19, 36]. These theories have been compelling, but have many drawbacks. They usually cannot predict the exact landing points and typically leave more than 30 % of the fixations unaccounted for.

Recent experiments show that fixations are extracting very specific information needed by the human subject’s ongoing task [14, 34]. The task context introduces enormous economies into this process that are very obvious: If a subject needs to pick up a red object, the search for that object can be limited to just red portions of the image; vast amounts of extraneous detail can be neglected. The visual information-gathering specificity of almost every portion of every task will introduce similar economies. Knowledge of task also has the promise of interpreting a substantial literature devoted to “change blindness.” Subjects fail to notice large changes between successive images or movie frames. While the exact reason for this has been the subject of controversy [22], the problem may be resolvable if one has access to the viewer’s cognitive agenda. On agenda changes are noticed and off agenda changes are not.

Theoretical Breakthroughs in Task Modeling

What experiments testing the information extracted during a fixation have lacked is a theory that accounts for the role of the cognitive processes that are controlling the subject's behaviors. What form should such a theory take? There have been several enormous theoretical advances, mostly from the fast emerging field of machine learning, that promise to have an enormous impact on quantitatively testable theories of cognition. The requirements of animat vision suggest that such a cognitive theory will have three important elements: (1) probabilistic representations, (2) the use of reward in learning, and (3) embodied cognitive architectures.

1. *Probabilistic methods.* There is rapidly increasing recognition that the brain is a probabilistic device and maintains a variety of mechanisms for calculating the statistical model of the world around it and its actions upon that world. To handle this a major new representational formalism has been developed that goes under the name *graphical models*. Originally developed by Pearl [26], such models have seen refinement as a general way to factor complex statistical interdependencies into to locally calculable quantities. The result is that the maintenance of elaborate statistical dependencies has become practical. Furthermore, Bayesian models of these interdependencies have proven their worth in characterizing many different observations in visual perceptions [35, 38].

2. *Reinforcement Learning.* A second breakthrough has been the development of model-making algorithms that are programmed by reward. It has long been appreciated that the brain must have mechanisms to learn complex behaviors and that these mechanisms must be steered by some scalar affinity signal. For the dominant effects the neurotransmitter dopamine has been implicated as the major signaling mechanism. Schultz, Glimcher, and others have made the connection between dopamine and reinforcement learning algorithms [17, 27, 33], the latter which by themselves have seen rapid development [1, 9]. Reinforcement learning algorithms are in their infancy but hold the promise of being an integral part of a comprehensive theory of animat vision learning.

3. *Embodied Cognition.* As emphasized by a number of researchers, the brain cannot be understood in isolation as so much of its structure is dictated by

the body it finds itself in and the world that the body has to survive in [7, 13, 24, 25, 29]. This has special important implications, particularly for the cognitive architectures, because the brain can be dramatically simpler than it could ever be without these encasing milieus. The reason is that the brain does not have to replicate the natural structure of the world or the special ways of interacting with it taken by the body but instead can have an internal structure that implicitly and explicitly anticipates these commitments. The brain just has to have an interface that allows successful interactions with the world, but does not have to explicitly model all the detailed consequences of the actions taken. This realization opens up a way to address the challenge of making the leap from the apparent simplicity of the observed behaviors to the complexity of the brain-body-world system that produces them and that is to see the behaving body itself as a laboratory instrument. From this vantage point, the momentary disposition of the eyes, head, and hands during the course of behavior reveals essential details about the underlying cognitive program that is making those choices.

Open problems

Although the importance of body in cognition has been stressed at least since Merleau-Ponty, until the middle of 1980s, it was only practical to study very controlled circumstances such as those made by an experimental subject seated in front of a small display.

The research program at Rochester pioneered the study of embodied, visually driven behaviors by the development of innovative laboratory equipment and techniques. With Pelz at the Rochester Institute of Technology [4], they were the first laboratory that were able to track the eyes inside a head-mounted display. This capability allowed the exploitation of another recent development: Virtual Reality(VR). It is now straightforward to render scenes in real time from a moving observer's vantage point that are extraordinarily close to the real thing. Thus a person can have the compelling illusion of being in a fictional world that at the same time is under experimental control. This capability, in turn, has allowed researchers to address many new experimental questions for the first time. For example, one can study a person's disposition

of visual resources in these virtual worlds by using the eye trackers inside the head-mounted display to manipulate the information that is available at each fixation [14, 30, 34].

Now flexible portable instrumentation can be attached to the body during the course of extended natural behaviors. Eye tracking capability that started out requiring subjects to be restrained in a bite bar has evolved to the point where portable trackers can be worn during a squash match. Head, hand, and body movements, even those of the facial muscles during expressions, can be reliably captured at high data rates during tea making, athletics, and everyday conversation. The new instrumentation opens up the possibility of acquiring large amounts of such data at millisecond time scales during these natural behaviors and thus provides access to the essential choices made in directing behavior under natural circumstances.

Obtaining such data from behavior and modeling it has led to another new question: How does one become confident that the models one builds are accurate? Answering this question has led to another new development and that is simulated human modeling. It is now possible to create models of humans that have the degrees of freedom of the skeletal system and also the capabilities of the binocular vision system [15, 31]. Thus one can build a human avatar that acts out the cognitive models obtained by fitting human data. A bonus is that one can test the models in completely new situations that were not part of the original human data gathering and observe their performance. This in turn can lead to an iterative refinement of the models and new experiments. However the most important aspect of this animat vision research avenue is the testing of the embodied cognition hypothesis with a suitably rich model. Our everyday experience and introspection as to the nature of the execution of everyday tasks has proven very misleading as to the brain's underlying representations owing to the artfulness of conscious experience.

References

1. Abeel P, Quigley M, Ng AY (2006) Using inaccurate models in reinforcement learning. In: International conference on machine learning, Pittsburgh
2. Aloimonos J, Bandopadhyay A, Weiss I (1988) Active vision. *Int J Comput Vis* 1(4):333–356
3. Arkin R (1998) Behavior based robotics. MIT, Cambridge
4. Babcock JS, Pelz JB, Peak J (2003) The wearable eye-tracker: a tool for the study of high-level visual tasks. In: Proceedings of the MSS-CCD2003
5. Bajcsy R (1988) Active perception. *Proc IEEE* 76:966–1005
6. Ballard DH (1991) Animate vision. *Artif Intell* 48(1): 57–86
7. Ballard D, Hayhoe M, Pook P (1997) Deictic codes for the embodiment of cognition. *Behav Brain Sci* 20:723–767
8. Barrow HG, Tanenbaum JM (1978) Computer vision systems. Academic, New York, pp 3–26
9. Barto AG, Mahadevan S (2004) Recent advances in hierarchical reinforcement learning. *Discret Event Dyn Syst* 13:341–379
10. Bonasso RP, Firby RJ, Gat E, Kortenkamp D, Miller DP, Slack MG (1997) Experiences with an architecture for intelligent reactive agents. *J Exp Theor Artif Intell* 9:237–256
11. Brooks RA (1986) A robust layered control system for a mobile robot. *IEEE J Robot Autom RA-2(1)*:14–23
12. Bryson JJ, Stein LA (2001) Modularity and design in reactive intelligence. In: International joint conference on artificial intelligence, Seattle
13. Clark A (1999) An embodied model of cognitive science? *Trends Cogn Sci* 3:345–351
14. Droll JA, Hayhoe MM, Triesch J, Sullivan BT (2005) Task demands control acquisition and storage of visual information. *J Exp Psychol Hum Percept Perform* 31: 1415–1438
15. Faloutsos P, van de Panne M, Terzopoulos D (2001) The virtual stuntman: dynamic characters with a repertoire of motor skills. *Comput Graph* 25:933–953
16. Firby RJ, Kahn RE, Prokowicz PN, Swain MJ (1995) An architecture for vision and action. In: Fourteenth international joint conference on artificial intelligence, Montréal, pp 72–79
17. Hayden BY, Platt ML (2007) Temporal discounting predicts risk sensitivity in rhesus macaques. *Curr Biol* 17:49–53
18. Itti L, Baldi P (2005) A principled approach to detecting surprising events in video. In: IEEE international conference on computer vision and pattern recognition (CVPR), San Diego, vol 1, pp 631–637
19. Koch C, Ullman S (1985) Shifts in selective visual attention: towards the underlying neural circuitry. *Hum Neurobiol* 4:219–227
20. Marr D (1982) Vision: a computational investigation into the human representation and processing of visual information. Freeman, San Francisco
21. Marr D, Poggio T (1979) A computational theory of human stereo vision. *Proc R Soc Lond B* 204:301–328
22. Most SB, Scholl BJ, Clifford ER, Simons DJ (2005) What you see is what you set: sustained inattentional blindness and the capture of awareness. *Psychol Rev* 112:217–242
23. Newell A (1990) Unified theories of cognition. Harvard University Press, Cambridge
24. Noe A (2005) Action in perception. MIT, Cambridge/London
25. O'Regan JK, Noe A (2001) A sensorimotor approach to vision and visual consciousness. *Behav Brain Sci* 24: 939–973
26. Pearl J (1988) Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann, San Mateo

27. Platt ML, Glimcher PW (1999) Neural correlates of decision variables in parietal cortex. *Nature* 400:233–238
28. Rosenfeld A (1969) Digital picture processing. Academic, Orland
29. Roy DK, Pentland AP (2002) Learning words from sights and sounds: a computational model. *Cogn Sci* 26:113–146
30. Shinoda H, Hayhoe MM, Shrivastava A (2001) What controls attention in natural environments? *Vis Res* 41: 3535–3546
31. Sprague N, Ballard D (2003) Multiple-goal reinforcement learning with modular sarsa(0). Technical report 798, Computer Science Department, University of Rochester
32. Terzopoulos D, Rabie TF (1997) Animat vision: active vision in artificial animals. *Videre J Comput Vis Res* 1(1): 2–19
33. Tobler PN, Fiorillo CD, Schultz W (2005) Adaptive coding of reward value by dopamine neurons. *Science* 307: 1642–1645
34. Triesch J, Ballard D, Hayhoe M, Sullivan B (2003) What you see is what you need. *J Vis* 3:86–94
35. Weiss Y, Simoncelli EP, Adelson EH (2002) Motion illusions as optimal percepts. *Nat Neurosci* 5:598–604
36. Wolfe J (1994) Guided search 2.0. a revised model of visual search. *Psychon Bull* 1:202–238
37. Yarbus AL (1967) Eye movements and vision. Plenum Press, New York
38. Yuille A, Kersten D (2006) Vision as bayesian inference: analysis by synthesis? *Trends Cogn Sci* 10:301–308

Anisotropic Diffusion

► [Diffusion Filtering](#)

Aperture Ghosting

► [Lens Flare and Lens Glare](#)

Appearance Scanning

► [Recovery of Reflectance Properties](#)

Appearance-Based Human Detection

William Robson Schwartz
Department of Computer Science, Universidade
Federal de Minas Gerais, Belo Horizonte, MG, Brazil

Synonyms

[Appearance-based pedestrian detection](#)

Related Concepts

► [Object Detection](#)

Definition

Human detection may be seen as a classification problem with two classes: human and nonhumans, in which the latter class is composed of background samples containing anything but humans. When the appearance-based human detection is employed, a large number of examples of human and nonhumans are considered to capture different poses, backgrounds, and occlusion situations through the extraction of feature descriptors so that a machine learning method can be used to classify samples as belonging to either one of the classes.

Background

Due to the large number of applications that require information regarding people's location, such as autonomous vehicles, surveillance, and robotics, finding people in images or videos presents large interest of the community. Even though widely studied in recent years [1], the human detection problem is still a challenge due to the wide variety of poses, clothing, background, and partial occlusions, which generate a large number of person's appearances.

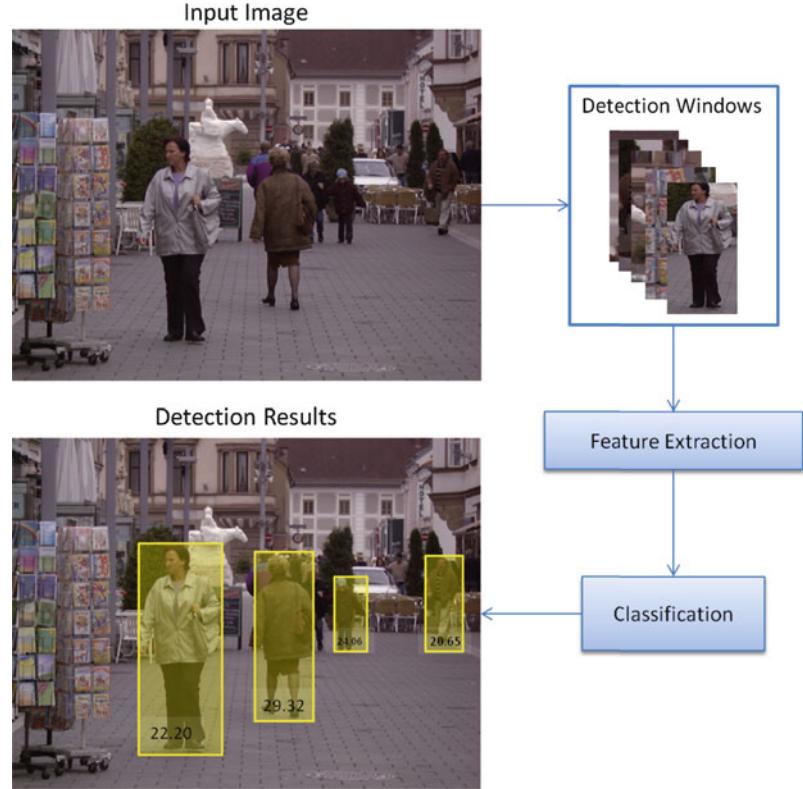
Two main approaches have been explored in the human detection literature. The first class of methods consists of a generative process that combines detected parts of the human body according to a model. The second class considers statistical analysis through the use of machine learning techniques to classify a feature vector composed of low-level feature descriptors extracted from a detection window. This approach, also referred to as appearance-based, captures the appearance information and focuses on the discrimination between human and nonhuman samples.

Theory

Appearance-based human detection presents two important aspects: feature extraction and classification. Once both aspects have been considered, the training and test (detection) steps can be performed.

Appearance-Based Human Detection

Fig. 1 Example of the human detection process. Image sample extracted from the INRIA Person dataset [2]



The feature extraction is responsible for capturing the visual information from the scene, such as the presence of strong vertical edges, homogeneous textured clothing, or color constancy in the face. Such characteristics, useful for human detection, will be extracted using low-level feature descriptors. It has been shown that the combination of these characteristics improves detection results [3]. Among the most used feature descriptors are the histograms of oriented gradients (HOG) [2], local binary patterns (LBP) [4], and Haar wavelet-based features [5].

The second relevant aspect is the choice of a machine learning method capable of classifying between humans and non-humans by giving higher importance to those descriptors that best distinguish between the two classes. Among the most employed methods are the linear discriminant analysis (LDA), neural networks (NN), support vector machines (SVM), and partial least squares (PLS).

The training step is responsible for learning parameters of the machine learning methods such that the differences between the two classes can be properly captured. For that, features are extracted from multiple

samples from both classes, and the descriptors are stored in feature vectors. It is important to emphasize that a good training set is important to assure that variations of the human appearances are captured. Each classification method presents a different way of learning the differences. For example, while SVM finds support vectors that maximize the margins between both classes, PLS will give more weight to those dimensions of the feature vector that best discriminate between the classes. In addition, it is important to note that a good training set is also important to assure that variations of the human appearances are captured.

Once the training has been performed, a sliding window is passed in the image at multiple scales to locate humans at different locations and scales. For each location, features are extracted and stored in a feature vector, which is then presented to the classifier. The output for each detection window is a value that reflects the probability or confidence in which a human is located inside the detection window. **Figure 1** illustrates the detection process of a typical appearance-based human detection method.

Application

In general, the human detection is of interest in any application that falls inside the *Looking at People* [6] (domain which focuses primarily in analyzing images and videos containing humans). For example, a human detector can be used to provide the location of each agent in a scene so that tasks such as tracking, re-identification, action, and activity recognition can be executed by a surveillance system. In addition, a human detector can be executed in the domain on autonomous navigation, where the location of the pedestrians will be used as information for path planning. Furthermore, the use of human detection systems embedded in vehicles may be very useful to assure pedestrian safety [7].

References

1. Enzweiler M, Gavrila DM (2009) Monocular pedestrian detection: survey and experiments. *IEEE Trans Pattern Anal Mach Intell* 31(12):2179–2195
2. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: *IEEE conference Computer Vision and Pattern Recognition (CVPR)*, San Diego, pp 886–893
3. Schwartz WR, Kembhavi A, Harwood D, Davis LS (2009) Human detection using partial least squares analysis. In: *IEEE International Conference on Computer Vision*, Kyoto, pp 24–31
4. Wang X, Han TX, Yan S (2009) An HOG-LBP human detector with partial occlusion handling. In: *IEEE International Conference on Computer Vision*, Kyoto, pp 32–39
5. Viola P, Jones MJ, Snow D (2005) Detecting pedestrians using patterns of motion and appearance. *Int J Comput Vis* 63(2):153–161
6. Gavrila DM (1999) The visual analysis of human movement: a survey. *Comput Vis Image Underst* 73(1):82–98
7. Gandhi T, Trivedi M (2007) Pedestrian protection systems: issues, survey, and challenges. *IEEE Trans Intell Transp Syst* 8(3):413–430

Appearance-Based Human Tracking

Bohyung Han
Department of Computer Science and Engineering,
Pohang University of Science and Technology
(POSTECH), Pohang, South Korea

Synonyms

[Human appearance modeling and tracking](#)

Related Concepts

- [Human Pose Estimation](#)

Definition

Appearance-based human tracking is a human tracking algorithm, where the measurement is based on the appearance of human such as color, texture, shape, and their combination.

Background

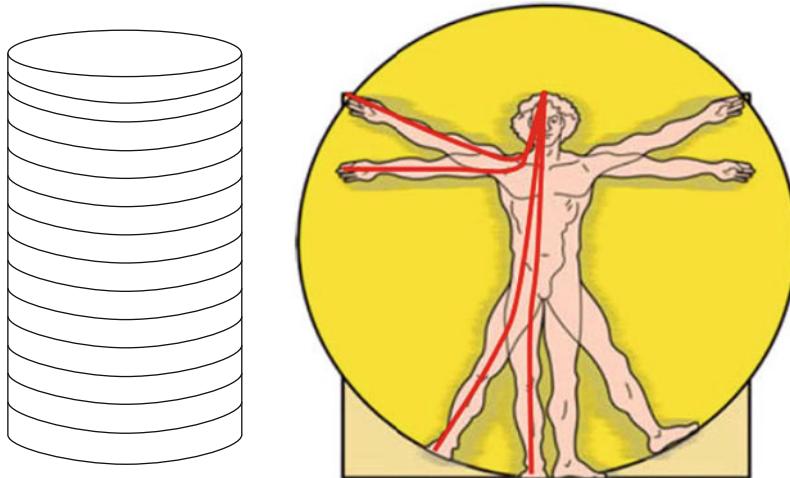
Various human tracking algorithms have been proposed so far, but the focus of each algorithm is different. Appearance-based human tracking is an algorithm to track human based on the similarity between the existing appearance model and the observation in image; the control and search algorithm in tracking can be arbitrary. Several different features have been used for human tracking including color, edge, gradient, texture, and shape, and multiple features are often integrated together for more robust observations. The appearance of human based on the features is represented by density function, histogram, template, and other descriptors. For the representation of human body, the spatial layout of the features is typically employed to model the appearance of human. The appearance models can be constructed for the entire human body, a few subregions of the human body, or the individual articulated human body parts separately, depending on the target state spaces and tracking algorithms.

Generic tracking algorithms can be employed to track a blob-based human with a simple appearance model in a low-dimensional state space. However, in case of articulated human body tracking, the dimensionality of target state space is typically very high, and the inference procedure of the complex human body configuration is generally complex; more efficient and specialized tracking algorithms are required.

Theory

Methodologies of Appearance Models

Two aspects – integrated features and mathematical representation techniques – are considered to characterize the appearance-based human tracking.



Appearance-Based Human Tracking, Fig. 1 Some examples to model spatial layout of a person in human appearance modeling. (*Left*) The appearance model is developed for each height

slice of a person. (Fig. 2 in [1]) (*Right*) The path length is the distance from the *top* of the head to a given point on the path. (Fig. 2 in [9])

Features Integrated

The appearance for human tracking is modeled with color [1–3], silhouette [4, 5], shape [6, 7], edge [4, 8], or texture.

Representation Method

Some appearance modeling techniques assume that the appearance of human body is consistent horizontally. With the assumption, human body is represented with multiple histograms based on a cylinder model as in the left subfigure in Fig. 1 [1] or density function [3]. Another method is a path-length model [9], where the spatial variations are modeled by the distance from head along the shape of the person as shown in the right subfigure in Fig. 1 and spatial-feature distribution is constructed for appearance modeling. Template is also frequently used [7, 10], and probabilistic template is integrated in [11].

Acquisition and Maintenance of Appearance Models

The appearances may be fixed throughout the sequence or adaptive to the variations of human body appearance. The initialization of the appearance can be performed based on (manual or automatic) human detection. In [12], the appearance of each body part of human is learned in an online manner based on simple features obtained in off-line process.

Tracking Control and Search

Human tracking can be classified into two types based on the description method of human body; one is blob-based tracking, and the other is articulated human body tracking. In case of blob-based tracking, tracking algorithm is simple, and the state space of the target is typically low dimensional. The algorithms in this type have no big difference from generic tracking algorithms for other objects; a major difference is that human tracking algorithms often divide target into a few sub-regions based on appearance consistency to improve measurement accuracy. However, the articulated human body tracking involves very high-dimensional state space (typically more than 20 dimension) and complicated probabilistic inference procedures; efficient tracking control and search algorithms are required to handle the challenges such as annealing [4, 11] and covariance sampling [5] in particle filter framework.

Application

Appearance-based human tracking has a lot of potential applications such as event detection and video understanding, pedestrian detection and tracking for intelligent vehicles and vision-based user interface in computer games.

References

1. Mittal A, Davis L (2003) M2tracker: a multi-view approach to segmenting and tracking people in a cluttered scene. *Int J Comput Vis* 51(3):189–203
2. Ramanan D, Forsyth D, Zisserman A (2005) Strike a pose: tracking people by finding stylized poses. In: IEEE conference on computer vision and pattern recognition (CVPR), vol 1. IEEE Computer Society, Los Alamitos, pp 271–278
3. Kim K, Davis LS (2006) Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering. In: European conference on computer vision (ECCV), Graz, vol 3, pp 98–109
4. Deutscher J, Reid I (2005) Articulated body motion capture by stochastic search. *Int J Comput Vis* 61:185–205
5. Sminchisescu C, Triggs B (2003) Estimating articulated human motion with covariance scaled sampling. *Int J Robot Res* 22(6):371–392
6. Haritaoglu I, Harwood D, David LS (2000) W4: real-time surveillance of people and their activities. *IEEE Trans Pattern Anal Mach Intell* 22:809–830
7. Lim H, Camps O, Szaier M, Morariu V (2006) Dynamic appearance modeling for human tracking. In: IEEE conference on computer vision and pattern recognition (CVPR), New York, vol 1, pp 751–757
8. Poon E, Fleet D (2002) Hybrid Monte Carlo filtering: edge-based people tracking. In: IEEE workshop on motion and video computing, Orlando, pp 151–158
9. Yoon K, Harwood D, Davis LS (2006) Appearance-based person recognition using color/path-length profile. *J Vis Commun Image Represent* 17(3):605–622
10. Cham TJ, Rehg J (1999) A multiple hypothesis approach to figure tracking. In: IEEE conference on computer vision and pattern recognition (CVPR), Fort Collins, vol 1, pp 239–245
11. Balan A, Black M (2006) An adaptive appearance model approach for model-based articulated object tracking. In: IEEE conference on computer vision and pattern recognition (CVPR), New York, vol 1, pp 758–765
12. Ramanan D, Forsyth D, Zisserman A (2007) Tracking people by learning their appearance. *IEEE Trans Pattern Anal Mach Intell* 29(1):65–81

Appearance-Based Pedestrian Detection

► [Appearance-Based Human Detection](#)

Articulated Pose Estimation

► [Human Pose Estimation](#)

Asperity Scattering

S. C. Pont

Industrial Design Engineering, Delft University of Technology, Delft, The Netherlands

Synonyms

► [Surface scattering](#); ► [Velvety reflectance](#)

Related Concepts

► [Retroreflection](#); ► [Lambertian Reflectance](#); ► [Surface Roughness](#)

Definition

The “asperities” can be of various nature, hairtips, dust, “fluff”, local high curvature spots or ridges (the term derives from scattering by powdered materials where the “asperities” are sharp edges like on broken glass).

Background

The reflectance of natural, opaque, and rough surfaces [1, 2] can be described by the Bidirectional Reflectance Distribution Function (BRDF) [6]. BRDFs that are common and well-known are those of Lambertian, perfectly diffusely scattering surfaces and of specular surfaces. Such surfaces scatter light in all directions (diffuse scattering) or primarily in the mirror direction (specular reflection). However, natural surfaces can scatter light in many other ways. Asperity scattering adds a “surface lobe” to the usual diffuse, backscatter, and specular lobes of rough surfaces. It is an important effect in many materials that are covered with a thin layer of sparse scatterers such as dust or hairs. In the common case that single scattering predominates, asperity scattering adds important contributions to the structure of the occluding contour and the edge

of the body shadow. This is the case because the BRDF is inversely proportional to the cosines of both illumination and viewing angles. The BRDF is generally low (and typically negligible), except when either the illuminating rays or visual directions graze the surface.

Because asperity scattering selectively influences the edges in the image of an object, it has a disproportionately (as judged by photometric magnitudes) large effect on (human) visual appreciation. It is a neglected but often decisive visual cue in the rendering of human skin. Its effect is to make smooth cheeks to look “velvety” or “peachy” (the appearances of both velvet and “peachy” skin are dominated by asperity scattering), that is to say, soft. This is a most important aesthetic and emotional factor that is lacking from Lambertian (looks merely dullish, paperlike), “skin type” BRDF (looks like glossy plastic), or even translucent (looks “hard”, vitreous) types of rendering.

Theory

Asperity scattering is due to scattering by a sparse “cloud cover” of the surface with essentially point scatterers. In sparse distributions of scatterers, one may assume that single scattering predominates. Then parameters of interest are the geometry of the cloud and the nature of the single scatterers. For this case, a physical, geometrical optical model was derived [3] and experimental data gathered [5].

It is also possible to fit asperity scattering characteristics in a convenient, simplified formula (note that basic physical constraints should hold, e.g., non-negativity, energy conservation, and Helmholtz reciprocity) [4]. For instance, for a surface element with unit (outward) normal n , irradiated from the direction (unit vector) i and viewed from the direction (unit vector) j , the following BRDF model

$$V(\mathbf{i}, \mathbf{j}, \mathbf{n}, a) = \frac{1}{\pi} \frac{a}{a + (\mathbf{i} \cdot \mathbf{n})(\mathbf{j} \cdot \mathbf{n})}, \quad (1)$$

describes a “surface lobe” such as one observes in black velvet cloth or peach skin. The parameter a determines the width of the lobe. (A similar behavior results if one substitutes $(\mathbf{i} \cdot \mathbf{n}) + (\mathbf{j} \cdot \mathbf{n})$ for $(\mathbf{i} \cdot \mathbf{n})(\mathbf{j} \cdot \mathbf{n})$.) The albedo is found to be

$$A_V(\mathbf{i}, \mathbf{n}, a) = \frac{2a}{(\mathbf{i} \cdot \mathbf{n})^2} \left(\mathbf{i} \cdot \mathbf{n} + a \log \frac{a}{a + \mathbf{i} \cdot \mathbf{n}} \right), \quad (2)$$

which has a lowest value

$$2a \left(1 + \log \left(\frac{a}{1+a} \right)^a \right) \approx 2a + 2 \log aa^2 + \dots \quad (3)$$

at normal incidence and rises monotonically to unity at grazing incidence. (For black velvet $a \ll 1$.) Other possibilities for simplified formulations may be found in graphics as so-called velvet shaders. However, care should be taken that many of these rendering applications do not fulfill the above mentioned basic physical constraints.

Open Problems

BRDFs of natural surfaces can probably be categorized into about a dozen different modes. Currently, only the forward, backward, diffuse, and surface scattering modes have been described by formal optical models.

Reflectance estimation from images suffers from image ambiguities. Prior knowledge on the reflectance statistics of natural materials plus formal descriptive models for the common modes of natural BRDFs can constrain this problem.

References

1. CURET: Columbia–Utrecht reflectance and texture database. <http://www.cs.columbia.edu/CAVE/curet>
2. Dana KJ, van Ginneken B, Nayar SK, Koenderink JJ (1999) Reflectance and texture of real-world surfaces, ACM Trans on Graphics, 18(1):1–34
3. Koenderink JJ, Pont SC (2003) The secret of velvety skin. Mach Vis Appl 14:260–268
4. Koenderink JJ, Pont SC (2008) Material properties for surface rendering. Int J Comput Vis Biomech 1(1):43–53

5. Lu R, Koenderink JJ, Kappers AML (1998) bidirectional reflection distribution functions) of velvet. *Appl Opt* 37(25):5974–5984
6. Nicodemus FE, Richmond JC, Hsia JJ (1977) Geometrical considerations and nomenclature for reflectance. National bureau of standard US monograph, 160

Automatic Gait Recognition

- ▶ [Gait Recognition](#)

Automatic Scale Selection

- ▶ [Scale Selection](#)

Automatic White Balance (AWB)

- ▶ [White Balance](#)

B

Backscattering

► [Retroreflection](#)

Bas-Relief Ambiguity

Manmohan Chandraker
NEC Labs America, Cupertino, CA, USA

Synonyms

[Generalized bas-relief \(GBR\) transformation](#)

Related Concepts

► [Illumination Estimation, Illuminant Estimation](#);
► [Lambertian Reflectance](#); ► [Photometric Stereo](#);
► [Shape from Shadows](#)

Definition

Members of the equivalence class of convex Lambertian surfaces that produce the same set of orthographic images under arbitrary combinations of distant point light sources are related by elements of a three-parameter subgroup of $GL(3)$, called generalized bas-relief (GBR) transformations. This inherent ambiguity in determining the three-dimensional shape of an object from shading and shadow information is called the bas-relief ambiguity.

Background

For a surface $f(x, y)$, the GBR-transformed surface is given by $\bar{f}(x, y) = \mu x + \nu y + \lambda f(x, y)$, where $\mu, \nu \in \mathbb{R}$ and $\lambda \in \mathbb{R}_{++}$. The orthographic image of an object with Lambertian reflectance, illuminated by an arbitrary set of distant point light sources, remains unchanged when the object shape is transformed by a GBR, with an inverse transformation applied on the set of light sources and a corresponding pointwise transformation on the albedos. Further, any continuous transformation that preserves the shading and shadowing configuration for a convex surface must belong to the GBR group [1].

Thus, for a Lambertian surface, any reconstruction or recognition algorithm based on shading and shadow information alone can at best enunciate the shape, albedo, or lighting up to a “bas-relief ambiguity.” The ambiguity derives its name from the corresponding low-relief sculpture technique (Italian: *basso rilievo*), which can be understood as a special case of the GBR, where $\lambda < 1$.

Theory

Under orthographic projection, each point on a surface may be represented as $[x, y, f(x, y)]^\top$, where $(x, y) \in \mathbb{R}^2$ is a point on the image plane and f is a piecewise differentiable function. The unit surface normal is given by

$$\hat{\mathbf{n}} = \frac{[-f_x, -f_y, 1]^\top}{\sqrt{f_x^2 + f_y^2 + 1}}. \quad (1)$$

A GBR transformation that maps a surface $f(x, y)$ to $\bar{f}(x, y) = \mu x + \nu y + \lambda f(x, y)$ and the corresponding inverse GBR transformation may be represented as 3×3 linear transformations:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \mu & \nu & \lambda \end{bmatrix}, \quad \mathbf{G}^{-1} = \frac{1}{\lambda} \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ -\mu & -\nu & 1 \end{bmatrix}. \quad (2)$$

Under the matrix product operation, the set of GBR transformations forms a subgroup of $GL(3)$, the group of 3×3 invertible linear transformations. The unit surface normals of the GBR-transformed surface $\bar{\mathbf{f}}$ are

$$\frac{\mathbf{G}^{-\top} \hat{\mathbf{n}}}{\|\mathbf{G}^{-\top} \hat{\mathbf{n}}\|}.$$

Shadows and Shading

The image formation equation at a point $\mathbf{p} = [x, y, f(x, y)]^\top$ on a Lambertian surface is given by

$$I(x, y) = \mathbf{n}^\top \mathbf{s} \quad (3)$$

where I is the intensity, \mathbf{n} is the product of albedo a and unit surface normal $\hat{\mathbf{n}}$, while \mathbf{s} is the light source direction, scaled by its strength. The point \mathbf{p} lies in an attached shadow if $\mathbf{s}^\top \hat{\mathbf{n}} < 0$, while it lies on a cast shadow boundary if there exists a point \mathbf{p}' on the surface, with unit normal $\hat{\mathbf{n}}'$, such that

$$\mathbf{s}^\top \hat{\mathbf{n}}' = 0, \quad \mathbf{p} - \mathbf{p}' = k\mathbf{s}, \text{ for some } k \in \mathbb{R}_{++}. \quad (4)$$

A point $\mathbf{p} = [x, y, f(x, y)]^\top$ lies in an attached shadow or on a cast shadow boundary in an image produced by the light source \mathbf{s} if and only if the point $\bar{\mathbf{p}} = \mathbf{G}\mathbf{p}$ does so in an image produced by the light source $\bar{\mathbf{s}} = \mathbf{G}\mathbf{s}$, where \mathbf{G} is a GBR transformation given by Eq. (2). Further, the image of a surface $f(x, y)$ with albedo $a(x, y)$, when illuminated by a light source \mathbf{s} , is equivalent to the image under the light source $\bar{\mathbf{s}} = \mathbf{G}\mathbf{s}$ of the GBR-transformed surface \bar{f} , with a pointwise albedo transformation given by

$$\bar{a} = \frac{a}{\lambda} \left(\frac{(\lambda f_x + \mu)^2 + (\lambda f_y + \mu)^2 + 1}{f_x^2 + f_y^2 + 1} \right)^{\frac{1}{2}}. \quad (5)$$

It follows that the set of images of a Lambertian surface-albedo pair $\{f, a\}$, under all possible combinations of distant light sources, is identical

to that of any GBR-transformed surface-albedo pair $\{\bar{f}, \bar{a}\}$ [1] (see Fig. 1). Thus, the illumination cones of surfaces related by a GBR transformation are identical [2].

Existence and Uniqueness

It is shown in [1] that any two convex, smooth surfaces with visible occluding contours that produce the same set of attached shadow boundaries must be related by a GBR transformation. Thus, the GBR transformation is the only one that preserves the set of all images of an object.

While the existence result for the bas-relief ambiguity does not explicitly require convexity of the surface, in practice, the image formation model for concave regions must account for interreflections. It has been shown that modeling diffuse interreflections uniquely determines the shape and lighting [3].

Integrability

In traditional photometric stereo, given images of a point \mathbf{p} under three or more known light sources, one may recover its surface normal $\hat{\mathbf{n}}$ using Eq. (3). However, in uncalibrated photometric stereo where the light sources are unknown, the surface normal and the light sources can be recovered only up to an arbitrary, invertible 3×3 linear transformation, since $\mathbf{n}^* = \mathbf{A}^\top \mathbf{n}$ and $\mathbf{s}^* = \mathbf{A}^{-1} \mathbf{s}$ satisfy Eq. (3) for any $\mathbf{A} \in GL(3)$.

For the recovered normal field to correspond to a surface, it must satisfy the integrability constraint [4]:

$$\frac{\partial}{\partial y} \left(\frac{n_1^*}{n_3^*} \right) = \frac{\partial}{\partial x} \left(\frac{n_2^*}{n_3^*} \right). \quad (6)$$

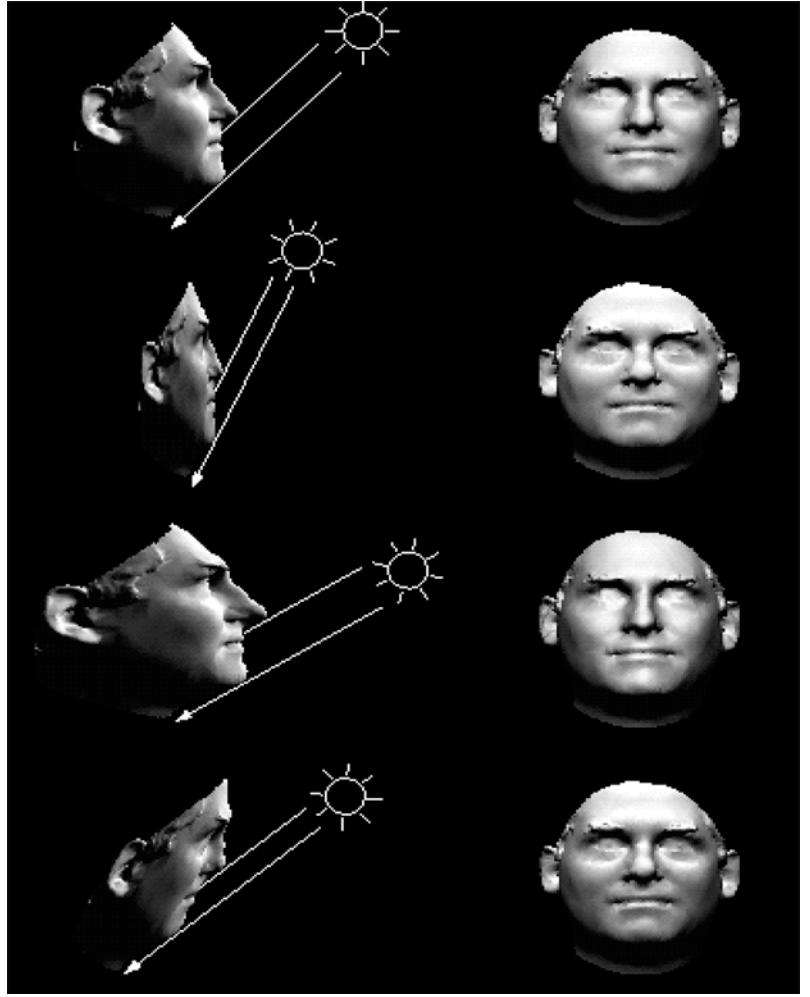
It is shown in [1] that requiring the recovered normal field to be integrable restricts \mathbf{A} to lie in the group of GBR transformations.

Generalizations

Under perspective projection, the shadows produced by an object under distant or proximal point light sources are the same as those produced by a surface transformed by a generalized perspective bas-relief (GPBR) transformation, with an inverse transformation applied on the light sources [5]. The GPBR is a three-dimensional elation [6] and, in the limiting case

Bas-Relief Ambiguity, Fig. 1

The *left column* shows various GBR transformations applied to a surface, with the corresponding inverse transformation applied to the light source direction. The *top row* is the true shape. The *right column* shows that the shading and shadows produced in an orthographic image of the surface are identical for any GBR transformation (Figure reproduced in part from [1], courtesy of the authors)



of orthographic projection, reduces to the definition of GBR in Eq. (2).

Under orthographic projection from an unknown viewpoint, there exists an ambiguity that corresponds to the group of three-dimensional affine transformations, called the Klein generalized bas-relief (KGBR) ambiguity, such that the set of images of an object is preserved under the action of a KGBR transformation on the shape, lighting, albedos, and viewpoint [7]. In the limiting case of a fixed viewpoint, the KGBR ambiguity reduces to the bas-relief ambiguity.

Application

The bas-relief ambiguity in computer vision explains psychophysical observations of similar unresolved

ambiguities in human visual perception [8]. An important consequence of the existence of the bas-relief ambiguity is that any image-based computer vision algorithm, relying on inference based solely on shading and shadow information, can only describe the object up to an arbitrary GBR transformation. Further, it has been established that for any infinitesimal motion of a surface f , there exists a motion for the GBR-transformed surface \bar{f} that produces the same motion field [1]. Thus, an infinitesimal motion does not provide additional cues for disambiguation.

Surface reconstruction up to a GBR transformation can be performed by imposing integrability in uncalibrated photometric stereo [9]. The bas-relief ambiguity may be resolved in practice by incorporating additional information, for instance, priors on albedo distribution [10, 11]. Alternatively, the presence of non-Lambertian

effects – such as specular highlights [12], a Torrance-Sparrow reflectance [13], or a spatially unvarying, isotropic, additive non-Lambertian reflectance component [14] – eliminates the GBR ambiguity.

References

1. Belhumeur P, Kriegman D, Yuille A (1999) The bas-relief ambiguity. *Int J Comput Vis* 35(1):33–44
2. Belhumeur P, Kriegman D (1998) What is the set of images of an object under all possible illumination conditions? *Int J Comput Vis* 28(3):245–260
3. Chandraker M, Kahl F, Kriegman D (2005) Reflections on the generalized bas-relief ambiguity. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR). IEEE Computer Society, San Diego, CA, pp 788–795
4. Horn B, Brooks M (1986) The variational approach to shape from shading. *Comput Vis Graph Image Process* 33: 174–208
5. Kriegman D, Belhumeur P (2001) What shadows reveal about object structure. *J Opt Soc Am A* 18(8):1804–1813
6. Coxeter H (1969) Introduction to geometry, 2nd edn. Wiley, New York
7. Yuille A, Coughlan J, Konishi S (2003) The KGBR viewpoint-lighting ambiguity. *J Opt Soc Am A* 20(1):24–31
8. Koenderink J, van Doorn A, Christon C, Lappin J (1996) Shape constancy in pictorial relief. *Perception* 25(2):151–164
9. Yuille A, Snow D (1997) Shape and albedo from multiple images using integrability. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), San Juan, pp 158–164
10. Alldrin N, Mallick S, Kriegman D (2007) Resolving the generalized bas-relief ambiguity by entropy minimization. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), Minneapolis, pp 1–8
11. Hayakawa H (1994) Photometric stereo under a light source with arbitrary motion. *J Opt Soc Am A* 11(11):3079–3089
12. Drbohlav O, Sara R (2002) Specularities reduce ambiguity of uncalibrated photometric stereo. In: Proceedings of European conference on computer vision (ECCV), Copenhagen, pp 46–60
13. Georgiades A (2003) Incorporating the Torrance-Sparrow model in uncalibrated photometric stereo. In: Proceedings of IEEE conference on computer vision (ICCV), Nice, pp 816–823
14. Tan P, Mallick S, Quan L, Kriegman D, Zickler T (2007) Isotropy, reciprocity and the generalized bas-relief ambiguity. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), Minneapolis, pp 1–8

Behavior Understanding

- Multi-camera Human Action Recognition

Bidirectional Texture Function and 3D Texture

Kristin Dana

Department of Electrical and Computer Engineering,
Rutgers University, The State University of New Jersey, Piscataway, NJ, USA

Synonyms

Mesostructure; Microgeometry; Relief texture; Solid texture; Surface roughness; Volumetric texture

Related Concepts

- Bidirectional Texture Function and 3D Texture;
- Bidirectional Feature Histogram; ► Light Field;
- Texture Recognition

Definition

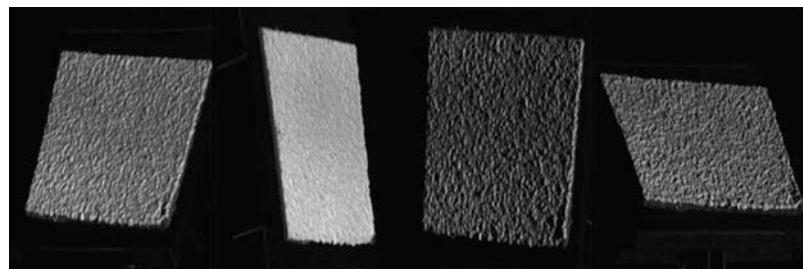
In the context of computer vision, texture often refers to a variation of image intensity or color, where the variation exhibits some type of repetition. The terms *2D texture* and *3D texture* provide a more precise definition of texture. A *2D texture* may be a color or shade variation such as a paisley print or zebra stripes. A textured surface can also exhibit geometric variations on the surface such as gravel, grass, or any rough surface. This type of texture is termed *3D texture* [1, 2]. Algorithms developed for 2D texture are generally not useful for 3D texture because appearance varies as a function of viewing and illumination direction.

The difference between 2D and 3D texture is readily apparent when considering photometric effects due to illumination direction and geometric effects due to viewing direction. Consider Fig. 1 with four images of the same surface under different surface tilt angles. The surface geometry does not change, but the illumination and viewing direction is different in each of the images. With a 2D texture model, these changes could be misinterpreted as changes in the texture class. As shown in Fig. 2, the appearance of the texture also changes significantly over the surface of a 3D

Bidirectional Texture

Function and

3D Texture, Fig. 1 Four images of the same 3D-textured surface. As the surface tilt and illumination direction varies, surface appearance changes



object. The photometry of 3D texture causes shading and shadowing that vary with illumination direction. The geometry of 3D texture causes a variation in foreshortening and occlusions along the imaged surface. Consider [Fig. 3](#) which illustrates oblique viewing of a 3D-textured surface patch. A similar oblique view of a 2D-textured surface patch gives a uniformly compressed or downsampled version of the frontal view. However, for an obliquely viewed 3D-textured surface patch, there is a non uniform resampling of the frontal view. Consequently, some texture features are compressed in the oblique view, while others expand. Computer graphics algorithms for texture mapping traditionally characterize the texture with a single image. To synthesize oblique views, these texture-mapping algorithms apply a uniform resampling which clearly cannot account for the spatially varying foreshortening and occlusions.

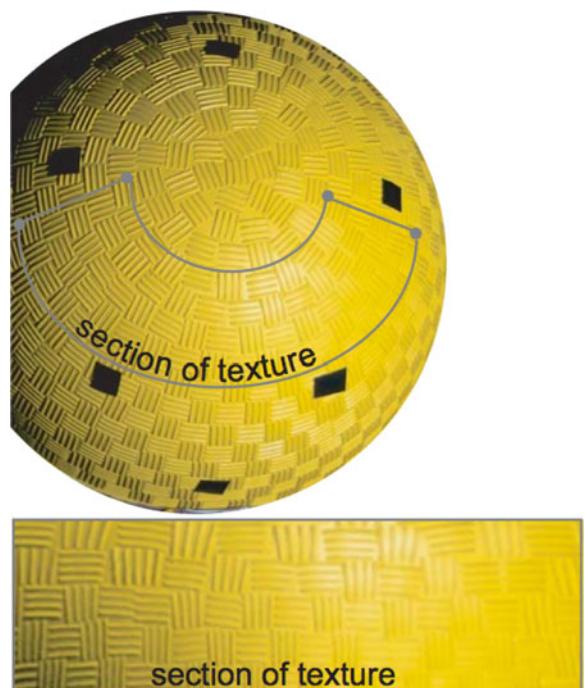
Background

Measurement of 3D texture with a bidirectional texture function (BTF) was introduced in [\[1, 2\]](#). This work created a database of 3D texture called the CUReT database (Columbia-Utrecht Reflectance and Texture database). This publicly available collection of measurements from real-world surfaces served as a starting point for subsequent work in 3D texture.

Theory

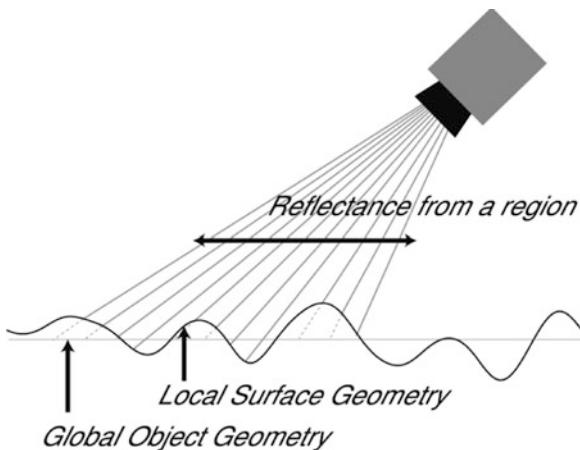
Histogram Models for 3D Texture

Numerous texture models for 2D texture have been developed since the early 1970s and are used in areas like texture mapping, texture synthesis, shape-from-texture, and texture classification and segmentation. These representations include co-occurrence matrices,



Bidirectional Texture Function and 3D Texture, Fig. 2 The local appearance of a 3D-textured object. Notice that local foreshortening, shadowing, and occlusions change across the texture because of the differences in global surface orientation and illumination direction

histograms, power spectra, Gabor filter outputs, texons, wavelets, and Markov random fields. In the late 1990s, the emphasis of texture research expanded to include 3D textures. Analytical models of intensity histograms of 3D-textured surface are developed in [\[3\]](#) and [\[4\]](#). Intensity histograms are a very basic tool to represent texture but are too simple for most computer vision tasks. A standard framework for texture representations consists of a primitive and a statistical distribution (histogram) of this primitive over space. Intensity is the simplest primitive; however, image features are better primitives to characterize the spatial



Bidirectional Texture Function and 3D Texture, Fig. 3
Geometry of 3D texture. For oblique views of a 3D-textured surface, the sampling rate of the surface depends on the local geometry

relationship of pixels. In order to account for changes with imaging parameters (view/illumination direction), either the primitive or the statistical distribution should be a function of the imaging parameters. Two methods to represent 3D texture are the bidirectional feature histogram (BFH) [5] and the 3D texton method [6]. In the 3D texton method, the primitive is a function of imaging parameters, while in the BFH method, the histogram is a function of imaging parameters. The advantage of the BFH approach is that there is no need for aligning images obtained under different imaging parameters.

Appearance-Based Representations

In computer vision, the precise surface geometry that comprises a 3D texture is often unknown. Instead, images of the 3D texture, i.e., appearance, are used to represent the textured surface. The term bidirectional texture function (BTF) is the appearance of texture as a function of viewing and illumination direction. BTF is typically captured by imaging the surface at a set of possible viewing and illumination directions. Therefore, measurements of BTF are collections of images. The term BTF was first introduced in [2, 7], and similar terms have since been introduced including BSSRDF [8] and SBRDF (spatial BRDF) [9]. SBRDF has a very similar definition to BTF, i.e., BTF is also a spatially varying BRDF.

BTF measurements can be very large because the measurements typically consist of a high-resolution image for every possible viewing and illumination direction. Dense sampling of the illumination and viewing space results in extremely large datasets to represent the surface. For example, if a 3 Mb image is captured for each of the 100 sampled viewing directions and 100 illumination directions, the resulting dataset is 30 Gb. Compact representations of the BTF are clearly important for efficient storage, rendering, and recognition. Methods for compact representations and compression of the BTF are presented in [10–13].

Geometry-Based Representations from Computer Graphics

In computer vision, image-based representations are standard because the surface geometry is typically unknown. However, in computer graphics, the precise geometry of the 3D-textured surface may be known. Rendering 3D textures using a volumetric representation of surface geometry is a common approach. Many of the rendering packages such as OpenGL and Blender use the term 3D texture to refer to volumetric texture. In this definition, the 3D texture is defined by opacity in a volume, instead of the definition here which refers to a surface height variation. In recent work [14], a volumetric representation is used for texture rendering, where the volume is a stack of semitransparent layers obtained using measured BTF data. Historically, volumetric texture methods are also referred to as solid texturing [15–18].

Application

The main applications for 3D texture representations are in recognition, synthesis, and rendering. While many applications use image-based representations, or assume a known geometric model, other applications need to capture the local geometry of 3D texture. Digital archiving of art is an example of such an application where fine-scale surface detail can enhance geometric models. Often the 3D texture is not easily captured with standard laser scanning devices. For example, in capturing the geometry of sculptures, researchers devised ways to capture high-resolution 3D texture such as tool marks [19–21]. An additional method for capturing high-resolution 3D texture

geometry uses a specialized texture camera based on the optical properties of curved mirrors [22]. For texture recognition, 3D texture methods have been used in material recognition [6, 23–25]. One of the popular recognition tasks is the recognition of materials from the CUReT database. Another real-world recognition task is the measurement and recognition of skin texture [26–28]. For texture synthesis and rendering, the main problem is to synthesize the appearance of 3D texture. Several authors have developed methods to synthesize and render 3D-textured surfaces using the BTF representation [29–35]. Other authors used an image-based approach to capture and render complex surfaces by direct photography of the full object under various illumination and viewing directions, simultaneously capturing object shape and surface light fields [36–38]. Synthesizing 3D textures via texture morphing enables creating and rendering novel 3D texture [39]. In computer graphics, 3D-textured surfaces were traditionally rendered by bump mapping [40], albeit with limited realism. In more recent years, several new methods have been developed that can efficiently render 3D texture detail. These include view-dependent displacement mapping [41], relief texture mapping [42], the polynomial texture map [43], and a Blender-based rendering method [44].

References

1. Dana KJ, van Ginneken B, Nayar SK, Koenderink JJ (1997) Reflectance and texture of real world surfaces. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Juan, Puerto Rico, pp 151–157
2. Dana KJ, van Ginneken B, Nayar SK, Koenderink JJ (1999) Reflectance and texture of real world surfaces. ACM Trans Graph 18(1):1–34
3. van Ginneken B, Koenderink JJ, Dana KJ (1999) Texture histograms as a function of irradiation and viewing direction. Int J Comput Vis 31(2–3):169–184
4. Dana KJ, Nayar SK (1998) Histogram model for 3d textures. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Santa Barbara, California, pp 618–624
5. Cula OG, Dana KJ (2004) 3D texture recognition using bidirectional feature histograms. Int J Comput Vis 59(1): 33–60
6. Leung T, Malik J (2001) Representing and recognizing the visual appearance of materials using three-dimensional textons. Int J Comput Vis 43(1):29–44
7. Dana KJ, van Ginneken B, Nayar SK, Koenderink JJ (1996) Reflectance and texture of real-world surfaces. Columbia University Technical Report CUCS-048-96
8. Jensen HW, Marschner SR, Levoy M, Hanrahan P (2001) A practical model for subsurface light transport. In: ACM SIGGRAPH, Los Angeles, California, pp 511–518
9. McAllister DK, Lastra AA, Cloward BP, Heidrich W (2002) The spatial bi-directional reflectance distribution function. In: ACM SIGGRAPH 2002 conference abstracts and applications, SIGGRAPH’02. ACM, New York, pp 265–265
10. Culha OG, Dana KJ (2001) Compact representation of bidirectional texture functions. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Kauai, Hawaii, vol 1. pp 1041–1067
11. Haindl M, Filip J (2007) Extreme compression and modeling of bidirectional texture function. IEEE Trans Pattern Anal Mach Intell 29(10):1859–1865
12. Ruiters R, Klein R (2009) Btf compression via sparse tensor decomposition. Proc EGSR Comput Graph Forum 28(4):1181–1188
13. Wu H, Dorsey J, Rushmeier H (2011) A sparse parametric mixture model for btf compression, editing and rendering. Proc Eurograp Comput Graph Forum 30(2):465–473
14. Magda S, Kriegman D (2006) Reconstruction of volumetric surface textures for real-time rendering. In: Proceedings of the Eurographics symposium on rendering, Nicosia, Cyprus, pp 19–29
15. Perlin K (1985) An image synthesizer. ACM SIGGRAPH 19(3):287–296
16. Kajiya JT, Kay TL (1989) Rendering fur with 3d textures. ACM SIGGRAPH 23(3):271–80
17. Jagannath R, Dorsey J, Rushmeier H (2004) Stereological techniques for solid textures. In: ACM SIGGRAPH 2004 Papers, SIGGRAPH ’04. ACM, New York, pp 329–335
18. Kopf J, Fu CW, Cohen-Or D, Deussen O, Lischinski D, Wong TT (2007) Solid texture synthesis from 2d exemplars. In: ACM SIGGRAPH 2007 papers, SIGGRAPH ’07. ACM, New York
19. Bernardini F, Martin IM, Rushmeier H (2001) High-quality texture reconstruction from multiple scans. IEEE Trans Vis Comput Graph 7(4):318–332
20. Bernardini F, Martin I, Mittleman J, Rushmeier H, Taubin G (2002) Building a digital model of michelangelo’s florentine pieta. IEEE Comput Graph Appl 1(22):59–67
21. Levoy M, Pulli K, Curless B, Rusinkiewicz S, Koller D, Pereira L, Ginzton M, Anderson S, Davis J, Ginsberg J, Shade J, Fulk D (2000) The digital michelangelo project: 3d scanning of large statues. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques. SIGGRAPH ’00. ACM/Addison-Wesley, New York, pp 131–144
22. Wang J, Dana KJ (2006) Relief texture from specularities. IEEE Trans Pattern Anal Mach Intell 28(3):446–457
23. Varma M, Zisserman A (2002) Classifying images of materials. In: Proceedings of the European Conference on Computer Vision (ECCV), Copenhagen, Denmark, pp 255–271
24. Culha OG, Dana KJ (2001) Recognition methods for 3d textured surfaces. Proc SPIE Conf Human Vis Electron Imaging VI 4299:209–220
25. Chantler M, Petrou M, Penirschke A, Schmidt M, McGunnigle G (2005) Classifying surface texture while

- simultaneously estimating illumination direction. *Int J Comput Vis* 62(1–2):83–96
26. Cula O, Dana K, Murphy F, Rao B (2004) Bidirectional imaging and modeling of skin texture. *IEEE Trans Biomed Eng* 51(12):2148–2159
 27. Cula OG, Dana KJ, Murphy FP, Rao BK (2005) Skin texture modeling. *Int J Comput Vis* 62(1/2):97–119
 28. Weyrich T, Matusik W, Pfister H, Bickel B, Donner C, Tu C, McAndless J, Lee J, Ngan A, Jensen HW, Gross M (2006) Analysis of human faces using a measurement-based skin reflectance model. *ACM Trans Graph* 25:1013–1024
 29. Liu X, Yu Y, Shum H (2001) Synthesizing bidirectional texture functions for real world surfaces. In: ACM SIGGRAPH, Los Angeles, pp 97–106
 30. Tong X, Zhang J, Liu L, Wang X, Guo B, Shum H (2002) Synthesis of bidirectional texture functions on arbitrary surfaces. *ACM Trans Graph* 21(3):665–672
 31. Koudelka ML, Magda S, Belhumeur PN, Kriegman DJ (2003) Acquisition, compression and synthesis of bidirectional texture functions. In: 3rd international workshop on texture analysis and synthesis (Texture 2003), Nice, France, pp 59–64
 32. Meseth J, Muller G, Klein R (2003) Preserving realism in real-time rendering of bidirectional texture functions. In: Proceedings of the openSG symposium, Darmstadt, Germany, pp 89–96
 33. Vasilescu MAO, Terzopoulos D (2004) Tensortextures: multilinear image-based rendering. In: ACM SIGGRAPH 2004 Papers, SIGGRAPH '04. ACM, New York, pp 336–342
 34. Wang J, Tong X, Chen Y, Guo B, Shum H, Snyder J (2005) Capturing and rendering geometry details for btf-mapped surfaces. *VIS Comput* 21:559–568
 35. Kautz J, Boulos S, Durand F (2007) Interactive editing and modeling of bidirectional texture functions. In: ACM SIGGRAPH 2007 papers, SIGGRAPH'07. ACM, New York
 36. Matusik W, Pfister H, Ngan A, Beardsley P, Ziegler R, McMillan L (2002) Image-based 3d photography using opacity hulls. In: Proceedings of the 29th annual conference on Computer graphics and interactive techniques. SIGGRAPH'02. ACM, New York, pp 427–437
 37. Debevec P, Hawkins T, Tchou C, Duiker H, Sarokin W, Sagar M (2002) Acquiring the reflectance field of a human face. In: ACM SIGGRAPH, San Antonio, pp 145–156
 38. Nishino K, Sato Y, Ikeuchi K (2001) Eigentexture method: Appearance compression and synthesis based on a 3d model. *IEEE Trans Pattern Anal Mach Intell* 23(11):1257–1265
 39. Matusik W, Zwicker M, Durand F (2005) Texture design using a simplicial complex of morphable textures. *ACM Trans Graph* 25(3):784–794
 40. Blinn JF (1978) Simulation of wrinkled surfaces. *ACM SIGGRAPH* 12(3):286–292
 41. Wang L, Wang X, Tong X, Lin S, Hu S, Guo B, Shum H (2003) View-dependent displacement mapping. *ACM Trans Graph* 22(3):334–339
 42. Oliveira MM, Bishop G, McAllister D (2000) Relief texture mapping. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, SIGGRAPH'00. ACM/Addison-Wesley, New York, pp 359–368
 43. Malzbender T, Gelb D, Wolters H (2001) Polynomial texture maps. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, SIGGRAPH'01. ACM, New York, pp 519–528
 44. Hatka M, Haindl M (2011) Btf rendering in blender. In: Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry, VRCAI'11. ACM, New York, pp 265–272

Blackbody Radiation

Rajeev Ramanath¹ and Mark S. Drew²

¹DLP® Products, Texas Instruments Incorporated, Plano, TX, USA

²School of Computing Science, Simon Fraser University, Vancouver, BC, Canada

Synonyms

Thermal Radiator

Related Concepts

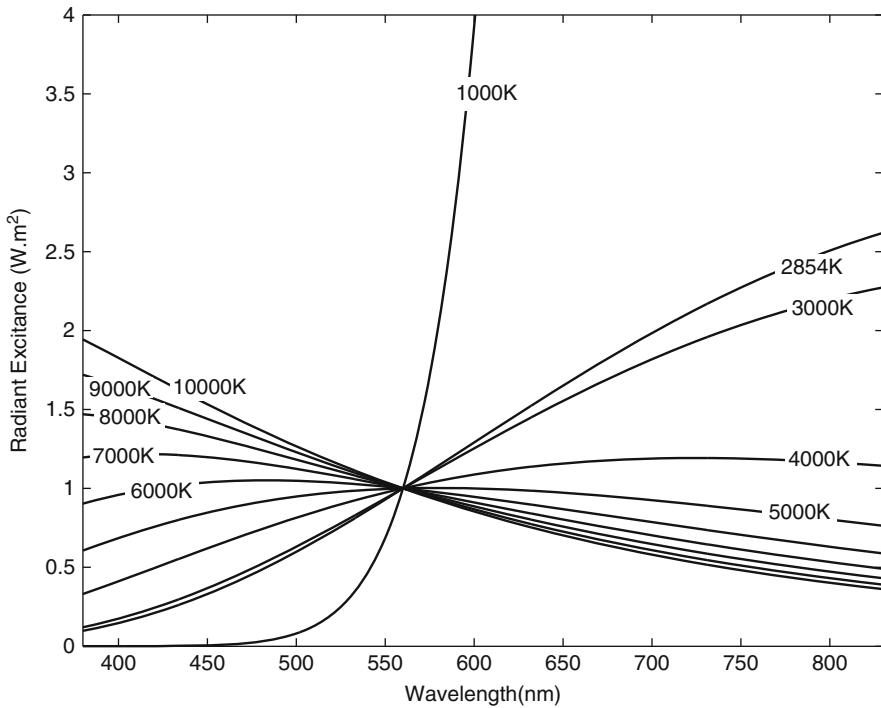
► [Planckian Locus](#)

Definition

A blackbody is an idealized object that absorbs all electromagnetic radiation incident on it. The absorbed energy is incandescently emitted with a spectral power distribution that is a function of its temperature. It is called a blackbody partly because it appears black to the human observer when it is cold, as it emits mostly infrared radiation.

Background

Blackbody radiation, in general, stood as a major challenge to the scientists in the nineteenth century as they were pushing the limits of classical physics. Several physicists studied blackbody radiators, including Lord Rayleigh, James Jeans, Josef Stefan, Gustav Kirchhoff, Ludwig Boltzmann, Wilhelm Wien, and finally, Max Planck, who arguably broke the way for quantum physics.



Blackbody Radiation, Fig. 1 Spectral power distributions of various blackbody radiators from 1,000 to 10,000 K, with all spectral distributions normalized to unity at 560 nm

Wien's approximation is used to approximately describe the spectral content of radiation from a blackbody. This approximation was first derived by Wilhelm Wien in 1893 [8]. This law was found to be accurate only for short wavelengths of emission spectra of blackbody radiators and is used today only as an approximation. Wien stated that the radiant excitance of a blackbody may be given by

$$M_e(\lambda, T) = \frac{2\pi h\nu^3}{c^2} \exp - \frac{h\nu}{kT} \quad (1)$$

$$= \frac{2\pi hc^2}{\lambda^5} \exp - \frac{hc}{kT\lambda} \quad (2)$$

The above equation holds in the specific case when $\exp - \frac{hc}{kT\lambda} \gg 1$, which typically occurs when the wavelength is short. The work by Wien was soon replaced by the findings of Max Planck, in 1901.

Another important finding regarding blackbody radiators was made by Josef Stefan in 1879 and later formalized by Ludwig Boltzmann. Known as the *Stefan-Boltzmann law*, it states that the total radiant excitance of a blackbody radiator is proportional to

the fourth power of its absolute temperature. In other words,

$$\int_{\lambda} M_e(\lambda, T) = \sigma T^4, \quad (3)$$

where $\sigma = 5.674 \times 10^{-8} W \cdot m^{-2} \cdot K^{-4}$ is known as the Stefan-Boltzmann constant.

Max Planck, in 1901, stated a more general law, known as *Planck's radiation law*. This describes the spectral distribution of radiant excitance M_e as a function of wavelength (These equations are often given in terms of frequencies instead of wavelength, as shown in this article. This is easily converted back and forth using $\nu \cdot \lambda = c$, where ν denotes frequency, λ the wavelength, and c the speed of light in vacuum.) λ and temperature T and is given by

$$M_e(\lambda, T) = \frac{c_1}{\lambda^5 \left[\exp \left(\frac{c_2}{\lambda T} \right) - 1 \right]} \quad (4)$$

where $c_1 = 2\pi hc^2 = 3.74183 \times 10^{-16} W \cdot m^{-2}$, $c_2 = h \cdot c / k = 1.4388 \times 10^{-2} m \cdot K$ (c is the speed of

light in vacuum: $2.99792458 \times 10^8 \text{ m} \cdot \text{s}^{-1}$, h is Planck's constant: $6.62606896 \times 10^{-34} \text{ J} \cdot \text{s}$, k is Boltzmann's constant: $1.38065 \times 10^{-23} \text{ J} \cdot \text{K}^{-1}$ [4], and the excitance is defined in units of $\text{W} \cdot \text{m}^{-3}$ [3, 5, 6, 9].

Figure 1 shows the spectral power distribution of various blackbody radiators from 1,000 to 10,000 K, with all spectral power distributions normalized to unity at 560 nm. As the blackbody gets hotter (T increases), one can see that the red content in the spectrum reduces and the blue content increases – an indication of the color as would be seen by the human observer. Both Wien's approximation and the Stefan-Boltzmann law can be derived from Planck's law.

Wien's displacement law may be derived by differentiating the equation for Planck's law Eq. (4) with respect to wavelength λ and equating the result to zero to find the maximum. It states that the spectral distribution of the spectral excitance of a blackbody reaches a maximum at a wavelength λ_m and that the product of the this maximum wavelength and the temperature of a blackbody is a constant, given by:

$$\lambda_m \cdot T = \frac{h \cdot c}{4.965114 k} = 2.8977685 \times 10^{-3} \quad (5)$$

where h denotes the Planck constant, c denotes the speed of light in vacuum, and k denotes the Boltzmann constant. The corresponding value of the spectral excitance is given by:

$$M_{e\lambda_m} = T^5 1.286673 \times 10^{-5} \text{ W m}^{-3} \quad (6)$$

Blackbody radiators are the select few sources of illumination that match “standard illuminant” spectral power distributions – this may be seen as valid in the case of the equivalence of standard illuminant “A” and a blackbody with a temperature 2,856 K.

References

1. 1998c CIE (1998) CIE standard illuminants for colorimetry. CIE, Vienna. Also published as ISO 10526/CIE/S006/E1999
2. CIE 15:2004 (2004) Colorimetry. CIE, Vienna
3. Longair MS (1984) Theoretical concepts in physics: an alternative view of theoretical reasoning in physics. Cambridge University Press, Cambridge

4. Mohr PJ, Taylor BN, Newell DB (2007) CODATA recommended values of the fundamental physical constants:2006. National Institute of Standards and Technology, Gaithersburg
5. Planck M (1901) On the law of distribution of energy in the normal spectrum. Annalen der Physik 4:553
6. Rybicki G, Lightman AP (1985) Radiative processes in astrophysics. Wiley-Interscience, New York
7. Wien W (1893) Sitzungsber. d. Berliner Akad, pp 55–62
8. Wien W (1896) Über die energievertheilung im emissionsspectrum eines schwarzen körpers. Annalen der Physik 58:662–669
9. Wyszecki G, Stiles WS (1982) Color science: concepts and methods, quantitative data and formulas, 2nd edn. Wiley, New York

Blackbody Radiator

► Planckian Locus

Blind Deconvolution

Tom Bishop¹ and Paolo Favaro²

¹Image Algorithms Engineer, Apple, Cupertino, CA, USA

²Department of Computer Science and Applied Mathematics, Universität Bern, Switzerland

Synonyms

Deblurring; Deconvolution; Kernel estimation; Motion deblurring; PSF estimation

Related Concepts

► Denoising; ► Image-Based Modeling; ► Inpainting

Definition

Blind image deconvolution is the problem of recovering a sharp image (such as that captured by an ideal pinhole camera) from a blurred and noisy one, without exact knowledge of how the image was blurred. The unknown blurring operation may result from camera motion, scene motion, defocus, or other optical aberrations.

Background

A correct photographic exposure requires a trade-off in exposure time and aperture setting. When illumination is poor, the photographer can choose to use a long exposure time or a large aperture. The first setting results in motion blur when the camera moves relative to objects in the scene during the exposure. The second setting results in out-of-focus blur for objects at depths away from the focal plane. Furthermore, these effects may be exacerbated by the user due to camera shake, incorrect focus settings, or other distortions such as atmospheric turbulence.

Under local approximations, these processes can be modeled as convolution operations between an ideal “sharp” image and a *kernel* or point-spread function (PSF). This (PSF) represents how a point of light in the scene would be imaged onto the camera’s sensor. In general the (PSF) may be space-varying or depth-varying, such that each point in 3D space has its own response in the kernel function. In general, the effect of the kernel is to blur and remove information from the image (see Fig. 1).

When the (PSF) is known, deconvolution algorithms can be used to remove the effect of these degradations. Deconvolution may be performed using direct (e.g., Fourier-based) or iterative (e.g., gradient descent or conjugate gradient based) algorithms. Essentially, a large linear system must be inverted to recover the sharp image, and depending on the conditioning of the matrix representing the blurring, the solution may be obtained to a greater or lesser accuracy. Observation noise also hinders exact invertibility, and for these reasons *regularization* of the solution is required. Such regularization typically imparts prior knowledge about the expected statistics of the sharp image, such as smoothness, sparseness of its gradients, or compressibility in some domain, and is typically key to obtaining well-behaved solutions.

In practice, the PSF is rarely known from calibration, and in a practical scenario it must be estimated from the blurred image itself. While many algorithms have been proposed to tackle the blind deconvolution problem with some success, a universal solution is not yet available and it is still an active area of research. The difficulty owes in part to the dimensionality of the unknowns in the problem, and its extreme ill-posedness, with the potential for multiple

local solutions arising from non-convex optimization problems. Progress has been made in both priors to describe images and blurs and better constrain the solution and estimation methods to better approximate the intractable inference problems.

Related methods also exist to recover the sharp image from multiple observations, for example, a blurred image and a sharp but noisy image, or multiple blurred images, in which case the problem is much more well posed and the solution is more readily obtained; this is also closely related to the problem of super-resolution where the sharp image is estimated at a higher resolution than the input images.

Theory

Imaging Model

The following linear spatially varying (LSV) observation model represents the formation of a general blurred image on the camera sensor:

$$g(\mathbf{x}) = \sum_{\substack{s \in \mathcal{S}_f, \\ (\mathbf{x}, s) \in \mathcal{S}_H}} h(\mathbf{x}, s)f(s) + w(\mathbf{x}), \quad \mathbf{x} \in \mathcal{S}_g, \quad (1)$$

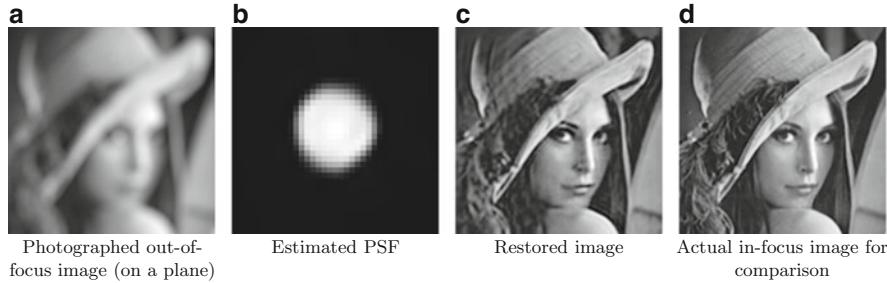
where $f(s)$ is the *true* or *sharp* image, $g(\mathbf{x})$ the observed blurred image, $w(\mathbf{x})$ is additive noise, and $h(\mathbf{x}, s)$ is the kernel function describing the blur. The position \mathbf{x} lies in the blurred image support $\mathcal{S}_g \subset \mathbb{R}^2$, while the position s lies in the true image support $\mathcal{S}_f \subset \mathbb{R}^2$. Notice that $\mathcal{S}_H \subset \mathbb{R}^4$ denotes the support of the kernel h . In the case the blur does not vary with position s in the true image and the depth is constant, the kernel may be reduced to a stationary PSF, $h(\mathbf{x}, s) = h(\mathbf{x} - s)$. Then the image model becomes a convolution:

$$g(\mathbf{x}) = [h \star f](\mathbf{x}) + w(\mathbf{x}) \quad (2)$$

With discretization and appropriate *lexicographic ordering*, or raster scanning of the images into vectors, either (1) or (2) may also be expressed in matrix-vector form:

$$\mathbf{g} = \mathbf{H}\mathbf{f} + \mathbf{w}, \quad \text{or equivalently,} \quad (3)$$

$$\mathbf{g} = \mathbf{F}\mathbf{h} + \mathbf{w}. \quad (4)$$



Blind Deconvolution, Fig. 1 Example of blind deconvolution using the method in [1] where a single image on a plane has been captured by a defocused camera. (a) Photographed out-of-focus

image (on a plane). (b) Estimated PSF. (c) Restored image. (d) Actual in-focus image for comparison

With the spatially invariant degradation model, the matrices \mathbf{F} and \mathbf{H} acquire a special structured block form, termed block Toeplitz with Toeplitz blocks (BTTB), with constant block entries on each block diagonal and constant diagonals within each block. Sometimes these matrices are approximated as circulant, implying circular convolution of the sharp image, and then they may be diagonalized using the discrete Fourier transform (DFT), enabling fast calculations to be performed.

Probabilistic Formulation

With uncertainty in the observation model (3), it is natural to estimate the most likely solution for the sharp image using a probabilistic approach. The Bayesian framework provides a unifying way to tackle such ill-posed inverse problems. Here a *likelihood* $p(g | \cdot)$ is specified from the imaging model and combined with a *prior* $p(f, h | \cdot)$ on the image and blur to be estimated, ensuring that only reasonable solutions are obtained. The resulting posterior distribution

$$p(f, h | g, \Omega) = \frac{p(g | f, h, \Omega) p(f | \Omega) p(h | \Omega)}{p(g)} \quad (5)$$

is used for inference of the unknowns, where Ω denotes *hyperparameters* of the model, such as noise variances or regularization parameters; these are usually considered known, but correctly estimating them is often critical for accurate blind deconvolution. The additive noise w is commonly assumed to be Gaussian or sometimes Poisson distributed. In the independent white Gaussian noise (WGN) case with variance σ_w , the distribution is $p_W(w) = \mathcal{N}(w | 0, \sigma_w I)$. Thus, the

likelihood of g conditioned on h, f is given by

$$\begin{aligned} p_G(g | f, h, \sigma_w) &= p_W(g - Hf) \\ &= (2\pi\sigma_w)^{-\frac{L_g}{2}} \exp\left[-\frac{1}{2\sigma_w^2} \|g - Hf\|^2\right], \end{aligned} \quad (6)$$

where L_g is the size of the vector g .

Bayesian Inference Methods

Due to the complexity of the chosen prior models, it may not be possible to obtain an exact analytic solution of Eq. (5). A common approximation is to compute a point estimate of the unknowns f and h via an optimization procedure. However, these estimates work well only with highly peaked distributions. With more uncertainty in the parameters, it is better to estimate the whole parameters *distribution*. Unfortunately, strategies that do so in the Bayesian framework are typically more computationally demanding [2–4]. Finally, as the estimated posterior distribution of the parameters must have a finite representation, further approximations or simulations must be introduced.

Maximum A Posteriori and Maximum Likelihood

The maximum a posteriori (MAP) solution is one common point estimate where it is possible to prescribe our prior knowledge about the unknowns. It is defined as the values \hat{f}, \hat{h} , and $\hat{\Omega}$ that maximize the posterior probability density:

$$\begin{aligned} &\{\hat{f}, \hat{h}, \hat{\Omega}\}_{\text{MAP}} \\ &= \underset{f, h, \Omega}{\operatorname{argmax}} p(g | f, h, \Omega) p(f | \Omega) p(h | \Omega) p(\Omega). \end{aligned} \quad (7)$$

A very related method is *maximum likelihood (ML)*, where one looks for

$$\{\hat{\mathbf{f}}, \hat{\mathbf{h}}, \hat{\boldsymbol{\Omega}}\}_{\text{ML}} = \underset{\mathbf{f}, \mathbf{h}, \boldsymbol{\Omega}}{\text{argmax}} \ p(\mathbf{g} | \mathbf{f}, \mathbf{h}, \boldsymbol{\Omega}). \quad (8)$$

Notice that the *maximum likelihood (ML)* method is essentially the *maximum a posteriori (MAP)* method where the prior distributions are uniform (uninformative). Despite this equivalence, the *maximum likelihood (ML)* is usually referred to as a non-Bayesian method. The advantage of using the *maximum a posteriori (MAP)* approach is that we can also encode the case where parameters are entirely or partly known by using degenerate distributions (Dirac deltas), that is, $p(\boldsymbol{\Omega}) = \delta(\boldsymbol{\Omega} - \boldsymbol{\Omega}_0)$. Then, the *MAP* and *ML* formulations become, respectively,

$$\begin{aligned} & \{\hat{\mathbf{f}}, \hat{\mathbf{h}}\}_{\text{MAP}} \\ &= \underset{\mathbf{f}, \mathbf{h}}{\text{argmax}} \ p(\mathbf{g} | \mathbf{f}, \mathbf{h}, \boldsymbol{\Omega}_0) p(\mathbf{f} | \boldsymbol{\Omega}_0) p(\mathbf{h} | \boldsymbol{\Omega}_0) \end{aligned} \quad (9)$$

$$\{\hat{\mathbf{f}}, \hat{\mathbf{h}}\}_{\text{ML}} = \underset{\mathbf{f}, \mathbf{h}}{\text{argmax}} \ p(\mathbf{g} | \mathbf{f}, \mathbf{h}, \boldsymbol{\Omega}_0). \quad (10)$$

The Bayesian framework can be used to describe several deconvolution methods and to emphasize their differences in terms of choice of likelihood, priors on the image, blur, and hyperparameters. Further differences can be found in how the maximization problem is solved.

A typical example of methods that can be formulated in the Bayesian framework is regularized approaches based on the L_2 norm. One such method is Tikhonov regularization, where a linear system is solved in least-squares sense by introducing an additional L_2 constraint on the unknowns. More in general, the blind deconvolution task is formulated as a constrained minimization where several regularization constraint terms are added.

A common choice is to always use a term in the form of $\|\mathbf{g} - \mathbf{H}\mathbf{f}\|^2$, called *data fidelity term*. The additional regularization terms encode the constraints on the unknowns. For example, one may want to impose smoothness of the image and the blur. To do so, a term that penalizes small variations of the image and the blur can be used. As the solution will be a trade-off between the data fidelity term and the regularization

terms, regularization parameters are used to adjust their weight.

An important example that illustrates this procedure is [5]. In that work the classical regularized image deconvolution formulation [6, 7] was extended to the blind image deconvolution (BID) case by adding regularization on the blur parameters. The problem is formulated as

$$\begin{aligned} \hat{\mathbf{f}}, \hat{\mathbf{h}} = \underset{\mathbf{f}, \mathbf{h}}{\text{argmin}} \ & \left[\|\mathbf{g} - \mathbf{H}\mathbf{f}\|_{\boldsymbol{\Omega}_w^{-1}}^2 \right. \\ & \left. + \lambda_1 \|\mathbf{L}_f \mathbf{f}\|^2 + \lambda_2 \|\mathbf{L}_h \mathbf{h}\|^2 \right], \end{aligned} \quad (11)$$

where λ_1 and λ_2 are the Lagrange multipliers for each constraint, and \mathbf{L}_f and \mathbf{L}_h are the regularization operators. To avoid oversmoothing the edges each \mathbf{L} operator is the Laplacian multiplied by space-varying weights. These weights are obtained from the local image variance as in [7–10].

Alternating Minimization or Iterated Conditional Modes

One of the main difficulties in MAP is to simultaneously recovering both \mathbf{f} and \mathbf{h} . The problem can be already observed in the image model Eq. (1), which is bilinear in both \mathbf{f} and \mathbf{h} . One common way to address this challenge is alternating minimization (AM). When applied to Eq. (11), it performs the minimization by working on one variable at a time while the others are fixed. As a result, the minimization of Eq. (11) becomes a sequence of linear problems. A related method is iterated conditional modes (ICM) proposed by Besag [11].

Minimum Mean-Squared Error

As mentioned earlier on, the MAP estimate is only a point estimate of the whole posterior probability density function (PDF). While this is not a problem when the posterior is highly peaked about the maximum, in the case of high observation noise or a broad (heavy-tailed) posterior, this estimate is likely to be unreliable (as chances of obtaining values that are different from the maximum are very likely). Indeed, in a high-dimensional Gaussian distribution, most of the probability mass lies away from the probability density peak [12].

One way to correct for this shortcoming is to use the minimum mean-squared error (MMSE) estimate.

The rationale is to find the optimal parameter values as those that minimize the expected mean-squared error between the estimates and the true values. This requires to compute the mean value of $p(\mathbf{f}, \mathbf{h}, \boldsymbol{\Omega} | \mathbf{g})$. However, computing MMSE estimates analytically is generally difficult. A more practical solution is to use sampling-based methods (see next paragraph).

Markov Chain Monte Carlo Sampling

A general technique to perform inference is to simulate the posterior distribution in Eq. (5), by drawing samples. Provided that we have obtained enough independent samples, this strategy allows us to deal with arbitrarily complex models in high-dimensional spaces, where no analytic solution is available. Markov chain Monte Carlo (MCMC) methods approximate the posterior distribution by the statistics of samples generated from a Markov chain. Widely used Markov chain Monte Carlo (MCMC) algorithms are the Metropolis-Hastings or Gibbs Samplers (see, e.g., [3, 13–15]).

The samples can then be used in Monte Carlo integration to obtain point estimates or other distribution statistics. For instance, in the BID problem the MMSE estimate of the \mathbf{f} can be readily obtained by taking the mean of the samples, $\frac{1}{n} \sum_{t=1}^n \mathbf{f}^{(t)}$.

Markov chain Monte Carlo (MCMC) can provide better solutions than AM or any other method. However, there are some limitations. First, they are very computationally intensive in comparison to the point estimate methods. Second, convergence to the posterior can be theoretically guaranteed, but in practice it can be hard to tell when this has occurred, and it may require a long time to explore the parameter space.

Marginalizing Hidden Variables

In the discussion so far the aim was to recover all the unknowns. However, in most cases one is interested in recovering only the sharp image \mathbf{f} . This leads to another approach to the BID problem where undesired unknowns are marginalized and inference is performed on the remaining variables (i.e., a subset of \mathbf{f}, \mathbf{h} , and $\boldsymbol{\Omega}$). Therefore, one can approach the BID inference problem in two steps. First, one can calculate

$$\hat{\mathbf{h}}, \hat{\boldsymbol{\Omega}} = \operatorname{argmax}_{\mathbf{h}, \boldsymbol{\Omega}} \int_f p(\mathbf{g} | \boldsymbol{\Omega}, \mathbf{f}, \mathbf{h}) p(\mathbf{f}, \mathbf{h} | \boldsymbol{\Omega}) p(\boldsymbol{\Omega}) d\mathbf{f} \quad (12)$$

and, second, one selects the sharp image

$$\hat{\mathbf{f}} \Big|_{\hat{\mathbf{h}}, \hat{\boldsymbol{\Omega}}} = \operatorname{argmax}_{\mathbf{f}} p(\mathbf{g} | \hat{\boldsymbol{\Omega}}, \mathbf{f}, \hat{\mathbf{h}}) p(\mathbf{f} | \hat{\boldsymbol{\Omega}}). \quad (13)$$

Alternatively, one can also marginalize \mathbf{h} and $\boldsymbol{\Omega}$ to obtain

$$\hat{\mathbf{f}} = \operatorname{argmax}_{\mathbf{f}} \int_{\mathbf{h}, \boldsymbol{\Omega}} p(\mathbf{g} | \boldsymbol{\Omega}, \mathbf{f}, \mathbf{h}) p(\mathbf{f}, \mathbf{h} | \boldsymbol{\Omega}) p(\boldsymbol{\Omega}) d\mathbf{h} \cdot d\boldsymbol{\Omega}. \quad (14)$$

Deconvolution Under a Gaussian Prior (MAP)

If the point-spread function (PSF) is known and assuming a Gaussian prior $p(\mathbf{f} | \boldsymbol{\Sigma}_f) = \mathcal{N}(\mathbf{f} | \mathbf{0}, \boldsymbol{\Sigma}_f)$ for \mathbf{f} , with a given covariance matrix $\boldsymbol{\Sigma}_f$, the posterior for \mathbf{f} is found as

$$p(\mathbf{f} | \mathbf{g}) \propto p(\mathbf{g} | \mathbf{f}) p(\mathbf{f} | \boldsymbol{\Sigma}_f) \quad (15)$$

$$\propto \exp\left(-\frac{1}{2} [\mathbf{f}^T (\sigma_w^{-2} \mathbf{H}^T \mathbf{H} + \boldsymbol{\Sigma}_f^{-1}) \mathbf{f} - 2\mathbf{f}^T (\sigma_w^{-2} \mathbf{H}^T \mathbf{g}) + \sigma_w^{-2} \mathbf{g}^T \mathbf{g}]\right) \quad (16)$$

which is a Gaussian

$$p(\mathbf{f} | \mathbf{g}) \propto \mathcal{N}\left(\mathbf{f} \Big| \boldsymbol{\mu}_{\hat{f}}, \boldsymbol{\Sigma}_{\hat{f}}\right) \quad (17)$$

$$\propto \exp\left(-\frac{1}{2} [\mathbf{f}^T \boldsymbol{\Sigma}_{\hat{f}}^{-1} \mathbf{f} - 2\mathbf{f}^T \boldsymbol{\Sigma}_{\hat{f}}^{-1} \boldsymbol{\mu}_{\hat{f}} + \boldsymbol{\mu}_{\hat{f}}^T \boldsymbol{\Sigma}_{\hat{f}}^{-1} \boldsymbol{\mu}_{\hat{f}}]\right). \quad (18)$$

By comparison of (16) and (18), the parameters are given as

$$\boldsymbol{\Sigma}_{\hat{f}}^{-1} = \sigma_w^{-2} \mathbf{H}^T \mathbf{H} + \boldsymbol{\Sigma}_f^{-1} \quad \boldsymbol{\mu}_{\hat{f}} = \boldsymbol{\Sigma}_{\hat{f}} (\sigma_w^{-2} \mathbf{H}^T \mathbf{g}). \quad (19)$$

The mean of this distribution, which is also the maximum, is just

$$\hat{\mathbf{f}} = \boldsymbol{\mu}_{\hat{f}} = (\mathbf{H}^T \mathbf{H} + \sigma_w^2 \boldsymbol{\Sigma}_f^{-1})^{-1} \mathbf{H}^T \mathbf{g}. \quad (20)$$

In practice, as solving the above equation involves inverting a large linear system, one employs iterative methods.

Application

Blind deconvolution methods are commonly used to restore images that have been distorted by motion blur, out-of-focus blur, and turbulence. As these methods provide an estimate of blur, other uses include digital refocusing, that is, digitally changing the focus setting of a camera after the snapshot, changing the camera bokeh, and obtaining a 3D model of the scene (from the out-of-focus blur).

Acronyms

| | |
|-------------|------------------------------|
| PSF | point-spread function |
| MC | Markov chain |
| MCMC | Markov chain Monte Carlo |
| ICM | iterated conditional modes |
| PDF | probability density function |
| ML | maximum likelihood |
| MAP | maximum a posteriori |
| AM | alternating minimization |
| BID | blind image deconvolution |
| MMSE | minimum mean-squared error |

References

1. Bishop TE, Molina R, Hopgood JR (2008) Blind restoration of blurred photographs via AR modelling and MCMC. In: IEEE international conference on image processing (ICIP), San Diego
2. Gelman A, Carlin JB, Stern HS, Rubin DB (2004) Bayesian data analysis, 2nd edn. Chapman & Hall, London
3. Neal RM (1993) Probabilistic inference using Markov chain Monte Carlo methods. Technical report CRG-TR-93-1, Department of Computer Science, University of Toronto, University of Toronto available online at <http://www.cs.toronto.edu/~radford/res-mcmc.html>
4. Jordan MI, Ghahramani Z, Jaakkola TS, Saul LK (1998) An introduction to variational methods for graphical models. Machine Learning, Kluwer Academic Publishers Hingham, MA, USA, Bari, Italy, 37(2):183–233
5. You YL, Kaveh M (1996) A regularization approach to joint blur identification and image restoration. IEEE Trans Image Process 5(3):416–428
6. Lagendijk RL, Biemond J, Boekee DE (1988) Regularized iterative image restoration with ringing reduction. IEEE Trans Acoust Speech Signal Process 36(12):1874–1887
7. Katsaggelos AK (1985) Iterative image restoration algorithms. PhD thesis, Georgia Institute of Technology, School of Electrical Engineering, Bombay, India
8. Katsaggelos AK, Biemond J, Schafer RW, Mersereau RM (1991) A regularized iterative image restoration algorithm. IEEE Trans Signal Process, Louisville, Kentucky, USA, 39(4):914–929
9. Efstratiadis SN, Katsaggelos AK (1999) Adaptive iterative image restoration with reduced computational load. Machine Learning - The Eleventh Annual Conference on computational Learning Theory archive, Kluwer Academic Publishers Hingham, MA, USA, 37(3):297–336
10. Kang MG, Katsaggelos AK (1995) General choice of the regularization functional in regularized image restoration. IEEE Trans Image Process 4(5):594–602
11. Besag J (1986) On the statistical analysis of dirty pictures. J R Stat Soc B 48(3):259–302
12. Molina R, Katsaggelos AK, Mateos J (1999) Bayesian and regularization methods for hyperparameter estimation in image restoration. IEEE Trans Image Process 8(2):231–246
13. Andrieu C, de Freitas N, Doucet A, Jordan M (2003) An introduction to MCMC for machine learning. Mach Learn 50:5–43
14. Gilks W, Richardson S, Spiegelhalter D (eds) (1995) Markov chain Monte Carlo in practice: interdisciplinary statistics. Machine Learning, Kluwer Academic Publishers Hingham, MA, USA, 37(2):183–233
15. Ó Ruanaidh JJ, Fitzgerald W (1996) Numerical Bayesian methods applied to signal processing, 1st edn. Springer series in statistics and computing. Springer, New York. ISBN:0-387-94629-2

Blur Estimation

Yu-Wing Tai

Department of Computer Science, Korean Advanced Institute of Science and Technology (KAIST), Yuseong-gu, Daejeon, South Korea

Synonyms

Blur Kernel estimation; Point spread function estimation

Related Concepts

► Defocus Blur; ► Image Enhancement and Restoration; ► Motion Blur

Definition

Blur estimation is a process to estimate the point spread function (a.k.a. blur kernel) from an image which suffered from either the motion blur or the defocus blur effects.

Background

When taking a photo with long exposure time, or with wrong focal length, the captured image will look blurry. This is because during the exposure period, the lights captured for a pixel are mixed with the lights captured for the other pixels within a local neighborhood. Such effect is modeled by the point spread function which describes how the lights are mixed during the exposure period.

In motion blur, the point spread function describes the relative motions between the camera and the scene. In defocus blur, the point spread function is related to the distance of a scene point from the focal plane of the camera. Recovering the point spread function is an important step in image deblurring in which the goal is to recover the sharp and clear image from the input blurry image. Also, the estimated point spread function can be used as a feature to evaluate the quality of a photo or to segment the in-focus regions from an image with large depth of field effects.

Theory

Representation

Mathematically, the effects of blurriness can be described by the following equation:

$$B(x, y) = \sum_{(m,n) \in \mathcal{N}(x,y)} I(x - m, y - n) k_{(x,y)}(m, n) \quad (1)$$

where B is the blurry image, I is the sharp image, $k_{(x,y)}$ is the point spread function at (x, y) , and \mathcal{N} is the local neighborhood of a pixel, respectively. In general, the point spread function $k_{(x,y)}$ is spatially varying. This means that every pixel can carry different point spread function. This happens when the scene has large depth disparity (for defocus blur) or when the scene contains a moving object (for motion blur) or when the camera exhibits rotational motion (for motion blur) during exposure period.

While the point spread function is spatially varying, the variation of the point spread function is spatially smooth. To simplify the problem, in case of the camera motion, many previous works have assumed the point spread function is spatially invariant, which reduces Eq. (1) into a convolution equation:

$$B = I \otimes k \quad (2)$$

where \otimes is the convolution operator. Recently, [1, 2] generalized Eq. (2) and proposed the projective motion blur model which uses a sequence of homographies to model the camera motion:

$$B(x, y) = \frac{1}{N} \sum_{i=1}^N I(H_i[x \ y \ 1]^t) \quad (3)$$

where H_i is the homography, $[x \ y \ 1]^t$ is the set of homogeneous coordinates of a pixel at (x, y) , and N is the number of homographies used to approximate the camera motion in discrete domain. Note that when there is image noise, an additional noise variable will be appended in Eqs. (1)–(3) under the assumption that noise is additive and is independent to the blur process.

Methodology

In order to estimate the point spread function, there are two main methods. The first method relies on hardware modification or calibration. The second method is purely based on software which typically requires assumptions on image prior or regularization to achieve reliable estimation.

If the effect of blur is caused by the camera internal setting, such as lens aberration, the point spread function can be calibrated. To calibrate the point spread function, the simplest method is to capture an image of a spotlight in a dark room. When there is no blur, the image of the captured spotlight (ideally) should occupy only one pixel. When there is blur, the image of the captured spotlight will occupy more than one pixel and the shape of the recorded spotlight is the point spread function. Similarly, for defocus blur, the focal length of the camera can be adjusted in order to obtain a set of defocus point spread function [3]. In the case of the motion blur, Ben-Ezra and Nayer [4] and Tai et al. [5] proposed a hybrid camera system which estimates the point spread function through integration of optical flows from the auxiliary high-speed camera. Yuan et al. [6] use a noisy/blurry image pair to estimate the point spread function for deblurring. Joshi et al. [7] use motion inertia sensor to measure the camera motion in 3D world.

There are many previous works targeting blur estimation using software approaches. In defocus blur, if the point spread function is a Gaussian kernel

and the edges are step edges, a defocus map can be obtained through the analysis of edge orientation and edge sharpness [8–10]. The defocus map indicates the scale of the defocus point spread function. In motion deblurring, blur estimation is usually coupled with the motion deblurring process. Fergus et al. [11] and Whyte et al. [2] proposed a multiscale variational Bayesian framework to estimate the point spread function. Jia [12] and Dai et al. [13] analyzed the edge alpha matte to obtain the marginal probability of the point spread function. Shan et al. [14] and Cho and Lee [15] used alternating optimization to iteratively refine the estimated point spread function and the deblurred image. Xu and Jia [16] proposed an edge selection method which improves the performance of blur estimation algorithms. A study on the blind motion deblurring algorithm with analysis on the point spread function can be found in [17].

Application

Blur estimation continuously receives a lot of attention in research area due to its application on deblurring. Since motion blur and defocus blur are common artifacts in imaging system, its applications range from astronomy telescope to satellite imaging, to medical imaging, and to common consumer-level camera. Besides deblurring, point spread function can also be used to evaluate image quality and to identify moving objects from a scene.

References

1. Tai YW, Tan P, Brown M (2011) Richardson-lucy deblurring for scenes under a projective motion path. *IEEE Trans PAMI* 33(8):1603–1618
2. Whyte O, Sivic J, Zisserman A, Ponce J (2010) Non-uniform deblurring for shaken images. In: *IEEE conference on computer vision pattern recognition (CVPR)*, San Francisco
3. Levin A, Fergus R, Durand F, Freeman WT (2007) Image and depth from a conventional camera with a coded aperture. *ACM Trans Graph* 26(3):70
4. Ben-Ezra M, Nayar S (2003) Motion deblurring using hybrid imaging. In: *IEEE conference on computer vision pattern recognition (CVPR)*, Madison, vol I, pp 657–664
5. Tai YW, Du H, Brown M, Lin S (2008) Image/video deblurring using a hybrid camera. In: *IEEE conference on computer vision pattern recognition (CVPR)*, Anchorage
6. Yuan L, Sun J, Quan L, Shum H (2007) Image deblurring with blurred/noisy image pairs. 26(3):1
7. Joshi N, Kang S, Zitnick L, Szeliski R (2010) Image deblurring with inertial measurement sensors. *ACM Trans Graph* 29(3):30
8. Bae S, Durand F (2007) Defocus magnification. *Computer Graphics Forum* 26(3):571–579 (Proc. of Eurographics)
9. Sun J, Sun J, Xu Z, Shum HY (2008) Image super-resolution using gradient profile prior. In: *IEEE conference on computer vision pattern recognition (CVPR)*, Anchorage
10. Joshi N, Szeliski R, Kriegman D (2008) Psf estimation using sharp edge prediction. In: *IEEE conference on computer vision pattern recognition (CVPR)*, Anchorage
11. Fergus R, Singh B, Hertzmann A, Roweis ST, Freeman WT (2006) Removing camera shake from a single photograph. *ACM Trans Graph* 25(3):787–794
12. Jia J (2007) Single image motion deblurring using transparency. In: *IEEE conference on computer vision pattern recognition (CVPR)*, Minneapolis
13. Dai S, Wu Y (2008) Motion from blur. In: *IEEE conference on computer vision pattern recognition (CVPR)*, Anchorage
14. Shan Q, Jia J, Agarwala A (2008) High-quality motion deblurring from a single image. *ACM Trans Graph* 27(3):73
15. Cho S, Lee S (2009) Fast motion deblurring. *ACM SIGGRAPH ASIA* 28(5):145
16. Xu L, Jia J (2010) Two-phase kernel estimation for robust motion deblurring. In: *European conference on computer vision (ECCV)*, Heraklion
17. Levin A, Weiss Y, Durand F, Freeman W (2009) Understanding and evaluating blind deconvolution algorithms. In: *IEEE conference on computer vision pattern recognition (CVPR)*, Miami

Blur Kernel Estimation

► Blur Estimation

Body Configuration Recovery

► Human Pose Estimation

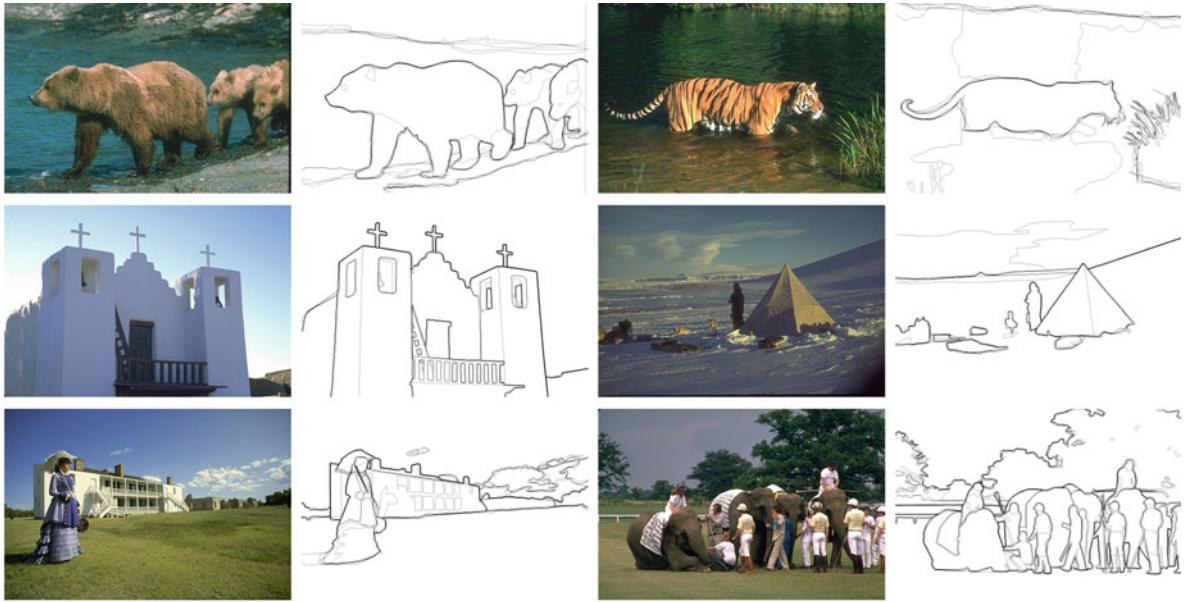
Boundary Detection

Xiaofeng Ren

Intel Science and Technology Center for Pervasive Computing, Intel Labs, Seattle, WA, USA

Synonyms

[Boundary extraction](#); [Contour detection](#)



Boundary Detection, Fig. 1 Examples of scenes in the Berkeley Segmentation Dataset [1]. Each photo is labeled by multiple human subjects, and the boundaries are shown as being

stacked together. There are variations across human subjects, but the marked boundaries are largely consistent especially for the salient ones

Related Concepts

► Edge Detection

Definition

Boundary detection is the process of detecting and localizing salient boundaries between objects in a scene.

Background

Boundary detection is closely related to, but not identical with, edge detection. Edge detection is a classical problem in computer vision which aims at finding brightness discontinuities. Edge detection is usually viewed as a low-level process of feature extraction that works under the assumption of ideal edge models (such as step and ridge edges).

In comparison, boundary detection is usually viewed as a mid-level process of finding boundaries of (and between) objects in scenes, thus having close

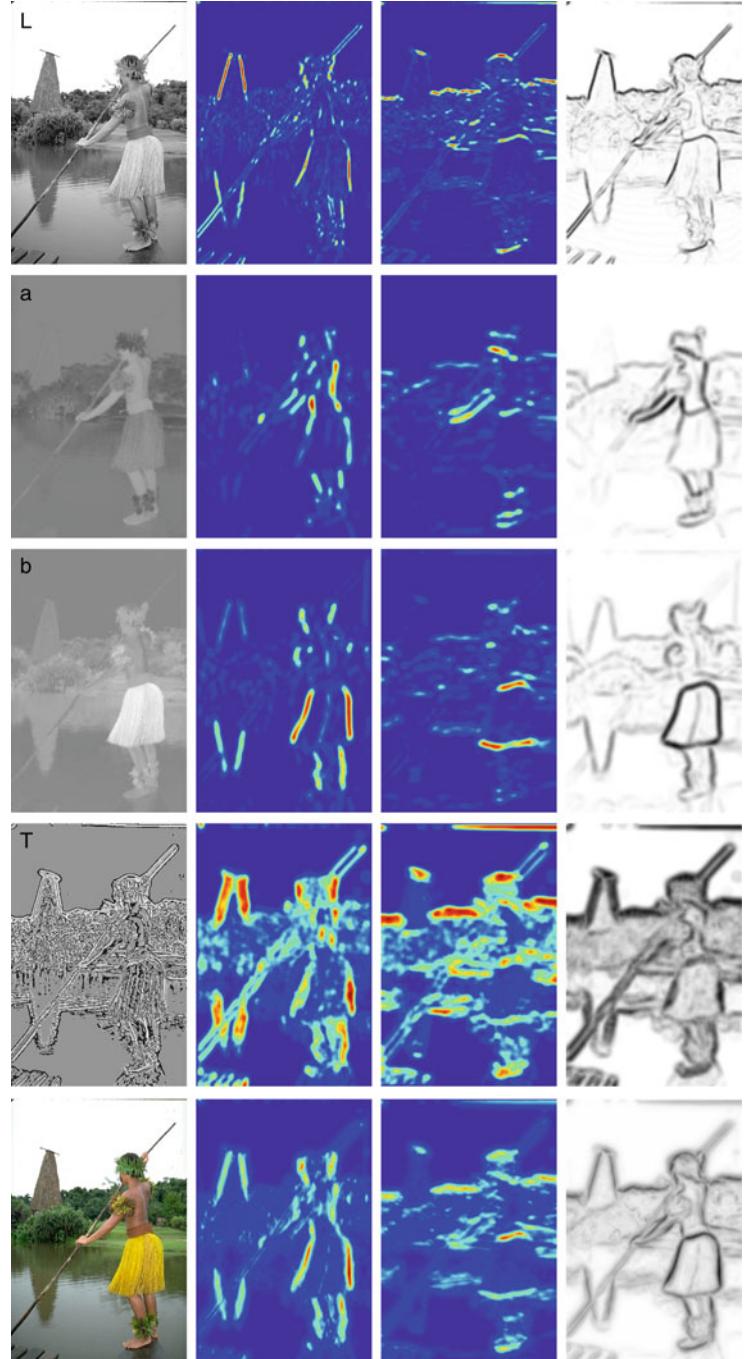
ties with both grouping/segmentation and object shape. A large-scale dataset of natural images with human-marked groundtruth boundaries, the *Berkeley Segmentation Dataset* (BSDS) [1, 2], was established in 2001 and quickly became the standard benchmark for both boundary detection and segmentation (see examples in Fig. 1). The Berkeley Segmentation Dataset helped defining the problem of boundary detection and clarifying several fundamental issues:

1. It directly addressed the complexities of real-world scenes by using a variety of photos from the Corel database.
2. It defined boundary detection as a perceptual problem by using human-marked boundaries as the groundtruth.
3. It showed that boundary detection is well defined by demonstrating that boundaries marked by human subjects are consistent.
4. It illustrated many challenges of boundary detection, including those of real-world texture, complex object appearance, and low-contrast boundaries.

By clarifying the task and establishing quantitative evaluation metrics, the Berkeley benchmark has witnessed and motivated large progresses in boundary detection in the recent years.

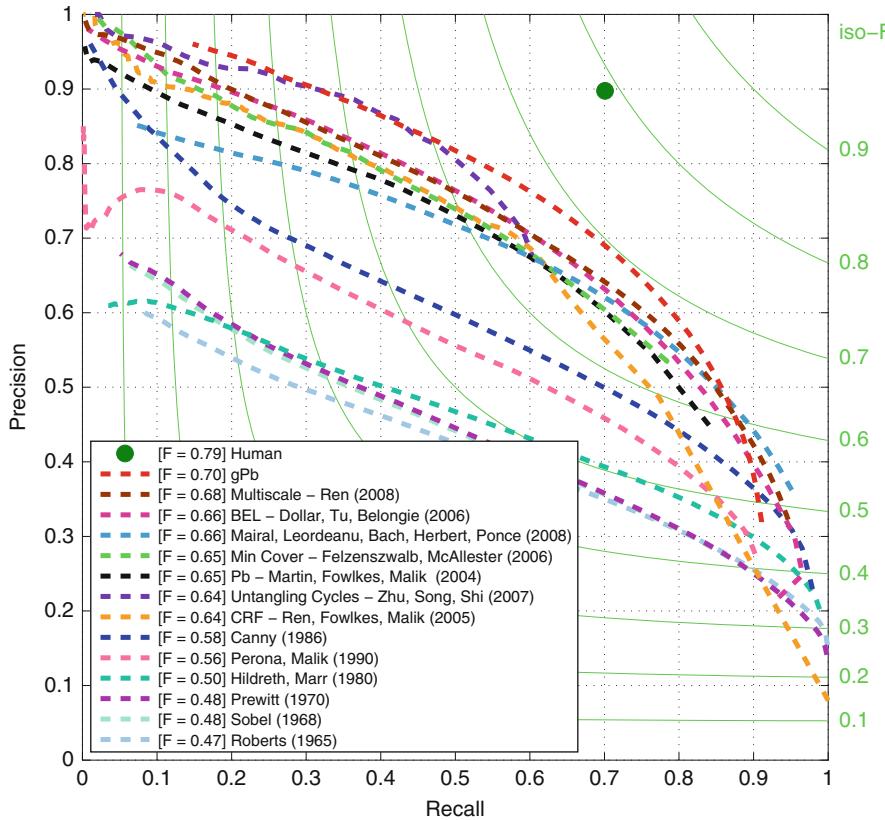
Boundary Detection, Fig. 2

Boundary detection combines multiple types of contrast (Courtesy of [9], see details there). Top-performing boundary detectors integrate together local contrast measurements from multiple channels (row 1, brightness; 2 and 3, color; and 4, texture, through textons) and multiple orientations (column 2, vertical; and 3, horizontal). *Red* color means high probability being a boundary and *blue* otherwise. The combined boundary contrast (last row, last column) is much better than any individual channel

**Local Boundary Detection**

Early approaches to edge detection used local derivative filters such as the Roberts, Sobel, or Prewitt filters [3]. More advanced solutions included that

of the zero crossing of Marr and Hildreth [4], the optimal filter design and non-maximum suppression in the Canny detector [5], and the use of quadrature filter pairs in oriented energy [6]. Scale (of the filter) is an important issue in edge detection and



Boundary Detection, Fig. 3 Precision-recall curves and F-measures of classical and modern boundary detection algorithms on the Berkeley benchmark (Courtesy of [9], also see [1]). A variety of approaches have been proposed and evaluated on the

benchmark. One can typically observe qualitative improvements in the boundary detection accuracy as the F-measure increases. State-of-the-art boundary detectors perform much better than, for instance, the Canny detector

Lindeberg proposed a mechanism for automatic scale selection [7].

The key concept in boundary detection is that of *contrast*: regions on two sides of a boundary tend to have different appearances; consequently, there tends to be a high contrast at a boundary location. To a large extent, this contrast can be captured and measured locally in an image neighborhood (e.g., a disk with a fixed radius). Local contrast can be measured in a number of ways, such as using linear filters or computing distances between histograms. To handle real-world scenes, modern boundary detectors utilize contrast information from multiple channels (including brightness, color, and texture), multiple orientations, and multiple scales (see examples in Fig. 2). Good examples of these contrast operators can be found in the *Pb* work (probability-of-boundary) of Martin et al. [8] and the *gPb* work (global probability-of-boundary) of Arbelaez et al. [9].

Given the complexities of contrast cues and the availability of labeled images, local boundary detection is often formulated as learning a binary classifier of boundary vs. non-boundary, which will produce a soft boundary “likelihood” at each pixel. Such a dense boundary map can be used directly or converted to a sparse boundary map through non-maximum suppression. A number of supervised machine learning techniques were used and tested in [8] to combine a small set of handcrafted contrast cues. Others have taken a more direct learning approach, such as using boosting trees to combine thousands of simple features over patches [10].

Global Boundary Detection

Boundaries are not local phenomena that occur independently at pixels. In fact, boundaries are defined

at the object level, and boundary pixels tend to form long, smooth contours, as evident from the examples in Fig. 1. Considerable efforts have been devoted to extracting boundaries globally, closely related to the classical problem of contour completion in perceptual organization. Algorithms for global boundary detection can be quantitatively evaluated based on their precision recall on the boundary detection task, same as for local boundary detection.

A variety of very different formulations have been proposed for global boundary detection and contour extraction, including classical works such as the Mumford-Shah functional [11]. Several recent approaches have successfully demonstrated, through benchmarking, that globalization greatly improves boundary detection accuracy over local detectors. Ren et al. [12] applied constrained Delaunay triangulation (CDT) to decompose locally detected contours into pieces and used conditional random fields (CRF) and belief propagation to integrate local contrast cues through interactions at junctions. Zhu et al. [13] computed complex eigenvectors of a normalized random walk matrix, using circular embedding, to detect topologically closed cycles. In the *gPb* work of Arbelaez et al. [9], eigenvectors of the affinity matrix were first computed, as in Normalized Cuts [14], and then the gradients of these eigenvectors were added to the local contrast cues to produce a single contrast map.

[Optional]: One related but different form of global boundary detection can be found in the case of *top-down object segmentation* [15], where the algorithm has access to the knowledge (such as shape or texture) of the objects that are in the scene. It is beyond the scope of the discussion here, as boundary detection typically refers to the *bottom-up* case where no high-level object knowledge is needed.

Application

Boundary detection is fundamentally connected to both image segmentation and object shape, and there should be no surprise that advances in boundary detection have led to many interesting applications in segmentation and object recognition.

For image segmentation, the use of *intervening contour* [16] allows one to convert any boundary map to pairwise affinities for use in the Normalized Cuts framework, and many systems (e.g., [17]) have

been using modern boundary detectors such as the *Pb* operator [8]. *Pb* is also used in [18], combined with the Watershed algorithm, to produce superpixels. Arbelaez et al. [9] proposed a hierarchical segmentation algorithm that, using the *gPb* boundary operator, produces compelling segmentation results and at the same time further improves the boundary detection accuracy.

For object recognition, boundary detectors such as *Pb* and *gPb* are often used to produce a boundary map, which is in turn used to compute shape descriptors. For instance, the work of Berg et al. [19] used *Pb* boundary maps with Geometric Blur for object and face recognition. The work of Ferrari et al. [20] used *Pb* to produce contour segments as a basis for shape matching. There are many segmentation-based approaches to recognition that also heavily rely on the quality of boundary detection (e.g., [21]).

State-of-the-art boundary detectors are sophisticated and require fairly intensive computation, which limits their applicability. There have been studies and efforts to speed up boundary detectors. In particular, the GPU-based detector of Catanzaro et al. [22] achieved a two-orders-of-magnitude improvement of speed over *gPb* without suffering any loss in boundary quality.

References

- Martin D, Fowlkes C, Malik J (2002) Berkeley segmentation dataset. <http://www.cs.berkeley.edu/projects/vision/bbsd>
- Martin D, Fowlkes C, Tal D, Malik J (2001) A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proceedings of international conference on computer vision (ICCV), Vancouver, vol 2, pp 416–423
- Faugeras O (1993) Three-dimensional computer vision: a geometric viewpoint. MIT, Cambridge
- Marr D, Hildreth E (1980) Theory of edge detection. Proc R Soc Lond B 207(1167):187–217
- Canny J (1986) A computational approach to edge detection. IEEE Trans Pattern Anal Mach Intell 8:679–698
- Morrone M, Owens R (1987) Feature detection from local energy. Pattern Recognit Lett 6:303–13
- Lindeberg T (1998) Edge detection and ridge detection with automatic scale selection. Int J Comput Vis 30: 117–156
- Martin D, Fowlkes C, Malik, J (2004) Learning to detect natural image boundaries using local brightness, color and texture cues. IEEE Trans Pattern Anal Mach Intell 26(5): 530–549

9. Arbelaez P, Maire M, Fowlkes C, Malik J (2010) Contour detection and hierarchical image segmentation. *IEEE Trans Pattern Anal Mach Intell* 33(5):898–916
10. Dollar P, Tu Z, Belongie S (2006) Supervised learning of edges and object boundaries. In: Proceedings of computer vision and pattern recognition (CVPR), New York, vol 2, pp 1964–1971
11. Mumford D, Shah J (1989) Optimal approximation by piecewise smooth functions and associated variational problems. *Commun Pure Appl Math* 42:577–685
12. Ren X, Fowlkes C, Malik J (2008) Learning probabilistic models for contour completion in natural images. *Int J Comput Vis* 77(1–3):47–64
13. Zhu Q, Song G, Shi J (2007) Untangling cycles for contour grouping. In: Proceedings of international conference on computer vision (ICCV), Vancouver, pp 1–8
14. Malik J, Belongie S, Leung T, Shi J (2001) Contour and texture analysis for image segmentation. *Int J Comput Vis* 43(1):7–27
15. Borenstein E, Ullman S (2002) Class-specific, top-down segmentation. In: Proceedings of european conference on computer vision (ECCV), Copenhagen, vol 2, pp 109–124
16. Leung T, Malik J (1998) Contour continuity in region-based image segmentation. In: Proceedings of european conference on computer vision (ECCV), Freiburg, Germany, vol 1, pp 544–559
17. Mori G, Ren X, Efros A, Malik J (2004) Recovering human body configurations: combining segmentation and recognition. In: Proceedings of computer vision and pattern recognition (CVPR), Washington, vol 2, pp 326–333
18. Hoiem D, Efros A, Hebert M (2007) Recovering occlusion boundaries from a single image. In: Proceedings of international conference on computer vision (ICCV), Rio de Janeiro
19. Berg A, Berg T, Malik J (2005) Shape matching and object recognition using low distortion correspondence. In: Proceedings of computer vision and pattern recognition (CVPR), San Diego, vol 1, pp 26–33
20. Ferrari V, Tuytelaars T, Gool LV (2006) Object detection by contour segment networks. In: Proceedings of european conference on computer vision (ECCV), Graz, pp 14–28
21. Gu C, Lim J, Arbeláez P, Malik J (2009) Recognition using regions. In: Proceedings of computer vision and pattern recognition (CVPR), Miami
22. Catanzaro B, Su BY, Sundaram N, Lee Y, Murphy M, Keutzer K (2009) Efficient, high-quality image contour detection. In: Proceedings of international conference on computer vision (ICCV), Kyoto, pp 2381–2388
23. Maire M, Arbelaez P, Fowlkes C, Malik J (2008) Using contours to detect and localize junctions in natural images. *Proceedings of computer vision and pattern recognition (CVPR)*, Anchorage, pp 1–8

Boundary Extraction

► [Boundary Detection](#)

BRDF Measurement

► [Recovery of Reflectance Properties](#)

BRDF Models

► [Reflectance Models](#)

C

Calculus of Variations

► [Variational Analysis](#)

Calibration

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Related Concepts

► [Camera Calibration](#); ► [Hand-Eye Calibration](#)

Definition

According to McGraw-Hill Encyclopedia of Science and Technology [1], calibration is the process of determining the performance parameters of an artifact, instrument, or system by comparing it with measurement standards. Adjustment may be a part of a calibration, but not necessarily. A calibration assures that a device or system will produce results which meet or exceed some defined criteria with a specified degree of confidence.

Background

In computer vision, there are multiple calibration problems. The most fundamental one is the camera

calibration, which determines the intrinsic and extrinsic parameters of a camera. It is the first step toward 3D computer vision. Other problems include hand-eye calibration, color calibration, and photometric calibration.

As stated in [1], two important measurement concepts related to calibration are precision and accuracy. Precision refers to the minimum discernible change in the parameter being measured, while accuracy refers to the actual amount of error that exists in a calibration. All measurement processes used for calibration are subject to various sources of error. It is common practice to classify them as random or systematic errors. When a measurement is repeated many times, the results will exhibit random statistical fluctuations which may or may not be significant. Systematic errors are offsets from the true value of a parameter and, if they are known, corrections are generally applied, eliminating their effect on the calibration. If they are not known, they can have an adverse effect on the accuracy of the calibration. High-accuracy calibrations are usually accompanied by an analysis of the sources of error and a statement of the uncertainty of the calibration. Uncertainty indicates how much the accuracy of a calibration could be degraded as a result of the combined errors.

References

1. Parker SP (ed) (1982) McGraw-Hill encyclopedia of science and technology, 5th edn. McGraw-Hill, New York. <http://www.answers.com/topic/calibration>

Calibration of a Non-single Viewpoint System

Peter Sturm

INRIA Grenoble Rhône-Alpes, St Ismier Cedex,
France

Synonyms

Non-central camera calibration

Related Concepts

► Camera Calibration; ► Center of Projection

Definition

A non-single viewpoint system refers to a camera for which the light rays that enter the camera and contribute to the image produced by the camera, do not pass through a single point. The analogous definition holds for *models* for non-single viewpoint systems. Hence, a non-single viewpoint camera or model does not possess a single center of projection. Nevertheless, a non-single viewpoint model (NSVM), like any other camera model such as the pinhole model, enables to project points and other geometric primitives, into the image and to back-project image points or other image primitives, to 3D. Calibration of a non-single viewpoint model consists of a process that allows to compute the parameters of the model.

Background

There exist a large variety of camera technologies (“regular” cameras, catadioptric cameras, fish-eye cameras, etc.) and camera models designed for these technologies. Often, technologies are developed in order to accommodate a desired model, such as for example, to provide a uniform spatial resolution.

Most cameras used in computer vision and other areas, can be well modeled by so-called single viewpoint, or central, camera models. These usually

model the 3D-to-2D mapping carried out by a camera, via lines of sight (or, camera rays) that all pass through a single point (the center of projection or optical center), and a mapping from these lines of sight to the image points where they hit the image plane.

Some camera types, especially some cameras having a wide field of view, but not only these, cannot be modeled very well using a single viewpoint model. This may be the case because a camera was designed to possess lines of sight that do not pass through a single point. This is, for example, the case for catadioptric cameras where the mirror surface is of conical shape: even if the camera looking at the mirror is positioned on the mirror’s axis, the lines of sight of the system do not converge to a single point, rather there exists a *viewpoint locus*. Another example is a single-lens stereo system consisting of a pinhole camera and two planar mirrors, such that the obtained images represent two perspective images acquired from two different effective viewpoints.

A camera may also be unintentionally of the non-single viewpoint type, for example, catadioptric cameras that were designed to have a single viewpoint but that due to a bad alignment of the camera and the mirror of the system, lose the single viewpoint property. Another example are fish-eye cameras; fish-eye optics are complex, and in principle, one can probably consider them as non-single viewpoint systems. However, in this and the previous example, it is not clear without further investigation of the actual system under consideration, if a single viewpoint model or an NSVM is better suited. This indeed depends on “how much” the system deviates from having a single viewpoint, how close the scene is in a typical application, how much image resolution is available, and so forth. This issue is further discussed in [1].

In the following, it is supposed that calibration is performed by acquiring one or several images of a calibration object, whose geometry is known and whose characteristic features (for simplicity, points shall be considered) can be extracted and identified in images.

Theory

There are different types of NSVM’s. One usually distinguishes parametric from non-parametric such models. For example, for non-single viewpoint



catadioptric systems, if the shape of the system's mirror is known or is known to belong to a parametric family of shapes, then the entire system can be described by few parameters: intrinsic parameters of the camera looking at the mirror, relative pose of mirror and camera, and possibly, shape parameters for the mirror. If such a parametric model is considered, calibration is, conceptually speaking, analogous to that of pinhole cameras. The main difference to calibration of pinhole cameras usually concerns the initialization process that allows to compute initial estimates of the camera parameters. Other than that, one may, in general, formulate calibration by a bundle adjustment type optimization of camera parameters, by minimizing, for example, the reprojection error, that is, a measure related to the distance between the predicted projections of points of the calibration object and those extracted in the images. Examples of parametric NSVM's are the so-called two-plane and GLC models [2–5], where lines of sight are parameterized by linear or higher-order transformations applied to points in two basis planes, other similar models where lines of sight are parameterized by linear transformations [6–9], models for pushbroom and X-slit cameras [10–13], and others [14].

A different concept consists in using non-parametric models to calibrate cameras. An example is the raxel model introduced by Grossberg and Nayar [15]. It essentially associates, to each pixel, a ray in 3D supporting the line of sight, and possibly properties such as its own radiometric response function. Importantly, one may use such a model without making any assumption about a parametric relationship between the position of pixels and the position and orientation of the associated lines of sight. Rather, one may store the coordinates of the lines of sight of all pixels, in a look-up table. Simplified versions of this model (without considering optical properties for individual pixels) have been used in several similar calibration approaches, for example, [15–18].

The principle of these approaches is thus to compute, for every camera pixel, a line of sight in 3D. To do so, at least two images of a calibration object are required. The simplest scenario considers the case where the calibration object is displaced by some known motion, between the image acquisitions. For each image, one has to estimate correspondences between camera pixels and points on the calibration

object. One way of achieving such dense correspondences is to use structured light principles, for instance, to use as calibration object a computer screen and to display a series of black and white patterns on it that encodes each pixel of the screen by a unique sequence of black and white intensities (e.g., Grey codes). Once correspondences of camera pixels and points of the calibration object (pixels of the computer screen in the above example) are known, the lines of sight can be computed by simply fitting straight 3D lines to the matched points on the calibration object. To do so, the latter must be expressed in the same 3D coordinate system, which is possible since it was assumed above that the motion of the calibration object between different acquisitions, is known. This approach was proposed independently by different researchers [15–17].

The above approach requires a minimum of two images, for different positions of the calibration object, and knowledge of the object's displacements. An extension to the case of unknown displacements was proposed in [18]. That approach requires at least three images; from matches of camera pixels and points on the calibration object, it first recovers the displacements of the object using an analysis of this scenario's multi-view geometry, and then computes lines of sight as above. Other approaches following this line are [19, 20].

The above approaches compute, for each camera pixel, an individual line of sight. If one assumes that the relation between pixels and lines of sight is particular, for instance, radially symmetric about an optical axis, then alternative solutions become possible. Such a possibility is to use a non-parametric representation of the distortion or undistortion function of a camera, that is, a function that maps viewing angles (angles between lines of sight and the optical axis) to distances in the image, between image points and the principal point or a distortion center. This can be done for both, single viewpoint and non-single viewpoint models. In the former case, it is assumed that all lines of sight pass through a single center of projection, whereas in the latter case, the model usually includes a mapping from viewing angles to the position of the intersection between lines of sight and the optical axis. Approaches of the latter type include [21, 22]. Besides making and using the assumption that the camera is radially symmetric, these calibration approaches resemble those explained above.

Application

All approaches described above, be they parametric or non-parametric, allow to perform 3D-to-2D projection and/or 2D-to-3D back projection, the latter meaning the mapping from an image point to the associated line of sight. By definition, the parametric models give analytical expressions to perform these operations. As for non-parametric ones, projection and back projection usually imply some interpolation and, possibly, a search. For instance, if a non-parametric model consists of a look-up table that gives, for each pixel, its line of sight, then back-projection of an image point with non-integer coordinates requires interpolation, whereas projection of a 3D point requires the search of the closest line(s) of sight in the look-up table and again an interpolation stage.

Other than these particular aspects, NSVM's can be used for many structure-from-motion computations completely analogously to other camera models, in particular, the pinhole model. Among the essential building blocks of structure from motion, there are pose estimation, motion estimation, and 3D point triangulation for calibrated cameras. As for pose and motion estimation (and other tasks), one usually requires two types of methods in an application: so-called minimal methods, which perform the estimation task from the minimum required number of point matches and which can be efficiently embedded in robust estimation schemes such as RANSAC, and non-linear optimization methods that refine initial estimates obtained from minimal methods. Minimal methods for pose [23–25] and motion estimation [26] with NSVM's are formulated perfectly analogously to those for the pinhole model, although their algebraic complexity is higher. All that is required by these methods from the NSVM is to compute lines of sight of interest points that are extracted and matched to another image (for motion estimation) or to a reference object (for pose estimation). As for the non-linear optimization stage, the minimization of the reprojection errors requires 3D-to-2D projections to be carried out, which, as explained above, may require search and interpolation, in which case the computation of the cost function's derivatives may have to rely on numerical differentiation. Other than that, there is no major conceptual difference compared to pose/motion estimation with pinhole cameras.

Another essential structure-from-motion task is 3D point triangulation. Here again, suboptimal methods work with lines of sight computed by the camera model for interest points in the images, and optimal methods perform the non-linear optimization of reprojection errors, where the same considerations hold as above for pose and motion estimation.

References

1. Sturm P, Ramalingam S, Tardif JP, Gasparini S, Barreto J (2011) Camera models and fundamental concepts used in geometric computer vision. *Found Trends Comput Graph Vis* 6(1–2):1–183
2. Chen NY (1979) Visually estimating workpiece pose in a robot hand using the feature points method. PhD thesis, University of Rhode Island, Kingston
3. Chen NY, Birk J, Kelley R (1980) Estimating workpiece pose using the feature points method. *IEEE Trans Autom Cont* 25(6):1027–1041
4. Martins H, Birk J, Kelley R (1981) Camera models based on data from two calibration planes. *Comput Graph Image Process* 17:173–180
5. Yu J, McMillan L (2004) General linear cameras. Proceedings of the 8th European conference on computer vision (ECCV), Prague, Czech Republic. pp 14–27
6. Pajdla T (2002) Stereo with oblique cameras. *Int J Comput Vis* 47(1–3):161–170
7. Ponce J (2009) What is a camera? Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Miami, USA
8. Seitz S, Kim J (2002) The space of all stereo images. *Int J Comput Vis* 48(1):21–38
9. Batog G, Gojac X, Ponce J (2010) Admissible linear map models of linear cameras. Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Francisco, USA
10. Gupta R, Hartley R (1997) Linear pushbroom cameras. *IEEE Trans Pattern Anal Mach Intell* 19(9):963–975
11. Pajdla T (2002) Geometry of two-slit camera. Technical Report CTU-CMP-2002-02, Center for Machine Perception, Czech Technical University, Prague
12. Zomet A, Feldman D, Peleg S, Weinshall D (2003) Mosaicing new views: the crossed-slit projection. *IEEE Trans Pattern Anal Mach Intell* 25(6):741–754
13. Feldman D, Pajdla T, Weinshall D (2003) On the epipolar geometry of the crossed-slits projection. Proceedings of the 9th IEEE international conference on computer vision, Nice, France. pp 988–995
14. Gennery D (2006) Generalized camera calibration including fish-eye lenses. *Int J Comput Vis* 68(3):239–266
15. Grossberg M, Nayar S (2005) The raxel imaging model and ray-based calibration. *Int J Comput Vis* 61(2):119–137
16. Greban K, Thorpe C, Kanade T (1988) Geometric camera calibration using systems of linear equations. Proceedings of the IEEE international conference on robotics and automation, Philadelphia, Pennsylvania, USA. pp 562–567



17. Champleboux G, Lavallée S, Sautot P, Cinquin P (1992) Accurate calibration of cameras and range imaging sensors: the NPBS method. Proceedings of the IEEE international conference on robotics and automation, Nice, France. pp 1552–1558
18. Sturm P, Ramalingam S (2004) A generic concept for camera calibration. Proceedings of the 8th European conference on computer vision (ECCV), Prague, Czech Republic. pp 1–13
19. Ramalingam S, Sturm P, Lodha S (2005) Towards complete generic camera calibration. Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Diego, USA, vol 1. pp 1093–1098
20. Dunne A, Mallon J, Whelan P (2010) Efficient generic calibration method for general cameras with single centre of projection. Comput Vis Image Underst 114(2):220–233
21. Tardif JP, Sturm P, Trudeau M, Roy S (2009) Calibration of cameras with radially symmetric distortion. IEEE Trans Pattern Anal Mach Intell 31(9):1552–1566
22. Ying X, Hu Z (2004) Distortion correction of fisheye lenses using a non-parametric imaging model. Proceedings of the Asian conference on computer vision, Jeju Island, Korea. pp 527–532
23. Chen CS, Chang WY (2004) On pose recovery for generalized visual sensors. IEEE Trans Pattern Anal Mach Intell 26(7):848–861
24. Ramalingam S, Lodha S, Sturm P (2004) A generic structure-from-motion algorithm for cross-camera scenarios. Proceedings of the 5th workshop on omnidirectional vision, camera networks and non-classical cameras, Prague, Czech Republic. pp 175–186
25. Nistér D, Stewénius H (2007) A minimal solution to the generalised 3-point pose problem. J Math Imaging Vis 27(1):67–79
26. Stewénius H, Nistér D, Oskarsson M, Åström K (2005) Solutions to minimal generalized relative pose problems. Proceedings of the 6th workshop on omnidirectional vision, camera networks and non-classical cameras, Beijing, China

Calibration of Multi-camera Setups

Jun-Sik Kim
Korea Institute of Science and Technology, Seoul,
Republic of Korea

Synonyms

Multi-camera calibration

Related Concepts

► Camera Calibration; ► Camera Parameters (Intrinsic, Extrinsic)

Definition

Calibration of multi-camera setups is a process to estimate parameters of cameras which are fixed in a setup. It usually refers to the process to find relative poses of the cameras in a single coordinate system under the assumption of known intrinsic camera parameters.

Background

Many computer vision methods including 3D reconstruction from stereo cameras utilize the multiple cameras in a system, assuming that the relative poses of cameras in a single coordinate system is already known. While the camera calibration using a planar pattern [1] simplifies calibration process for intrinsic and extrinsic parameters of each camera, estimating camera poses in *a fixed global coordinate system* is still required.

The term “multi-camera setup” includes many different camera configurations such as a stereo, inward-looking cameras, outward-looking cameras, or camera sensor networks. Because each camera setup has different viewpoint and field of view (FOV) configuration, one single calibration method is not able to deal with all the multi-camera setups. Depending on the camera configuration, different calibration approach should be considered.

Theory

A projection matrix \mathbf{P}_i of a camera i in a multi-camera setup is given as

$$\mathbf{P}_i = \mathbf{K}_i [\mathbf{R}_i \ \mathbf{T}_i]. \quad (1)$$

The matrices \mathbf{R}_i and \mathbf{T}_i represent the pose of the camera i in a *predetermined* fixed coordinate system. More specifically, the matrices express a transformation between the fixed global coordinate system and the local camera coordinate system. The goal of the multi-camera calibration is to estimate the matrices \mathbf{R}_i and \mathbf{T}_i for all i in the system. The camera matrix \mathbf{K}_i represents intrinsic parameters of the camera i , which can be assumed to be known by calibrating the

intrinsic parameters of each camera independently in advance.

Multi-camera systems can be categorized into three configurations: inward-looking cameras, outward-looking cameras and large camera networks.

Inward-looking cameras The case that all the cameras in the system have a FOV shared. A stereo camera is considered *inward-looking* because the two cameras should see the same scene.

Outward-looking cameras The case that all the cameras in the system do not share their FOV. Because no FOVs are overlapped, it is usually called *non-overlapping cameras*.

Camera networks The case that some cameras share their FOVs but there are no common FOV for all cameras. Distributed cameras are usually in this category. Most likely, nearby cameras have a common FOV, farther cameras can not see it.

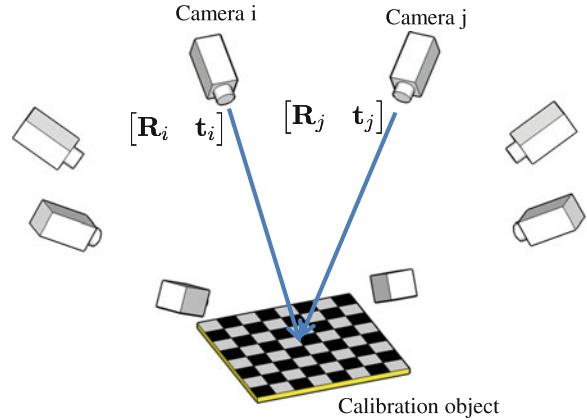
Each camera configuration has different constraints used in the multi-camera calibration, and the resulting calibration method becomes different to each other.

Inward-Looking Cameras

When all the cameras in the multi-camera setup have a common FOV, the multi-camera calibration is relatively simple. A calibration object is placed in the common FOV as shown in Fig. 1 so that each camera can see it, and the pose of the cameras with respect to the object coordinate system is estimated by using conventional pose estimation methods [2–4]. In this case, the common coordinate system of the multi-camera configuration is set to be the coordinate system of the calibration object.

For pose estimation of each camera, a planar pattern is preferable because it provides better visibility for all cameras. Note that, however, it is not limited to a planar pattern when the object visibility from every camera is ensured.

Sinha et al. [5] present an automatic calibration method using object silhouettes. In this method, epipolar geometry between cameras is estimated from dynamic silhouettes and projective structure is recovered. Following self-calibration completes the Euclidean reconstruction. This aims especially for shape-from-silhouette or visual hull reconstruction.



Calibration of Multi-camera Setsups, Fig. 1 Calibration of inward-looking cameras

Outward-Looking Cameras

When the FOVs of cameras in the system are not overlapped, it is impossible to place a calibration object which is observable from multiple cameras. The pose between cameras can be estimated by utilizing the fact that the transformation between cameras are fixed in motions called a rigidity constraint [6].

Assume that the coordinate systems of cameras i and j are transformed by a transformation \mathbf{R}_{ij} and \mathbf{t}_{ij} . When the camera i moves with a transformation $\Delta\mathbf{R}_i$ and $\Delta\mathbf{t}_i$, the motion of the camera j with a rotation $\Delta\mathbf{R}_j$ and a translation $\Delta\mathbf{t}_j$ is given as

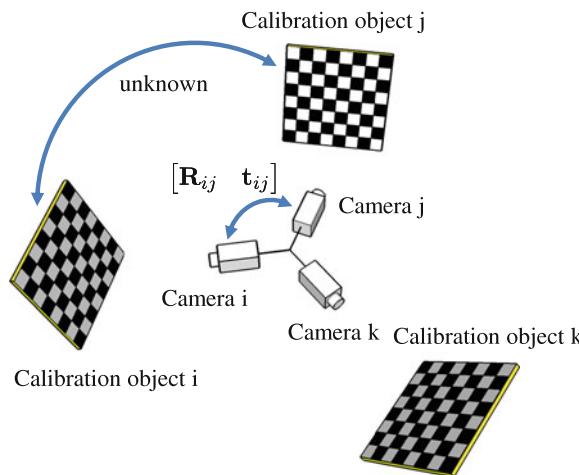
$$\begin{bmatrix} \Delta\mathbf{R}_j & \Delta\mathbf{t}_j \\ \mathbf{0}^\top & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{ij} & \mathbf{t}_{ij} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{R}_i & \Delta\mathbf{t}_i \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{ij} & \mathbf{t}_{ij} \\ \mathbf{0}^\top & 1 \end{bmatrix}^{-1}, \quad (2)$$

and this equation can be rewritten in a $\mathbf{AX} = \mathbf{XB}$ form on the Euclidean group as

$$\begin{bmatrix} \Delta\mathbf{R}_j & \Delta\mathbf{t}_j \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{ij} & \mathbf{t}_{ij} \\ \mathbf{0}^\top & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{ij} & \mathbf{t}_{ij} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{R}_i & \Delta\mathbf{t}_i \\ \mathbf{0}^\top & 1 \end{bmatrix}. \quad (3)$$

Solving the unknowns \mathbf{R}_{ij} and \mathbf{t}_{ij} on the Euclidean group is known as a *hand-eye calibration* in the robotics community [7–9]. Two or more motions of the camera rig provide enough number of constraints.

One practical problem is to estimate *motions* $\Delta\mathbf{R}$ and $\Delta\mathbf{t}$ of each camera. One stable way is to use calibration objects for each camera. First place a calibration object for each camera and take pictures while moving the camera rig. At each time frame,



Calibration of Multi-camera Setups, Fig. 2 Calibration of outward-looking cameras

the poses of each camera can be estimated by using a conventional pose estimation method. The motions of cameras are computed by calculating the difference of the poses at different time frames. Note that the geometric relation between calibration objects is not required in calculating each camera motions. The only requirement is that the calibration objects are fixed in motions. [Figure 2](#) shows the calibration objects for three non-overlapping cameras.

By solving the $\mathbf{AX} = \mathbf{XB}$ equation on the Euclidean group, the transformation between two cameras is obtained. If the multi-camera system has more than two cameras, every camera can be registered in the fixed global coordinate system by chaining the transformations pairwisely. However, the pairwise chaining of transformation does not guarantee the globally consistent registration. In addition, the transformation between cameras may be inconsistent depending on the data provided to the $\mathbf{AX} = \mathbf{XB}$ solver. Dai et al. [10] represent a rotation averaging strategy to improve the consistency of the estimated transformation, and use a global bundle adjustment [11] for final polishing. Lebraly et al. [12] more focus on an sparse implementation of the global bundle adjustment to ensure the consistency in the fixed global coordinate system.

Kumar et al. [13] present completely different approach to use a mirror so that the cameras can

observe the mirrored calibration pattern, and show successful calibration result for the ladybug camera.

Camera Networks

One general configuration is a camera network, which usually has many fixed cameras seeing in different directions. When every camera shares its FOV with any other camera in the network, relative transformations between the cameras can be estimated using calibration objects, and they are registered by chaining the transformation. In fact, this calibration process is cast to a conventional structure from motion problem. Once all the relative transformations are obtained, globally consistent localization of the cameras in the fixed coordinate system is achieved by using bundle adjustment [11]. [Figure 3](#) shows a possible placement of calibration objects in calibrating a camera network. Devarajan et al. [14] introduce a *vision graph* to describe the feature visibility between cameras, and try to optimize the graph network by belief propagation.

Baker and Aloimonos [15] present a method based on the multi-frame structure from motion algorithm, and use a rod with two LEDs at each end as a calibration object. The LEDs provides accurate and easily detectable correspondences for the precisely synchronized cameras by waiving the rod.

Svoboda et al. [16] propose a convenient and complete self-calibration method using a laser pointer including intrinsic parameter estimation. The method is based on the stratification of the transformations; at first projective reconstruction is achieved by factorization and later upgraded to Euclidean space by imposing geometric constraints such as a square pixel assumption. Their source codes are available for public use.

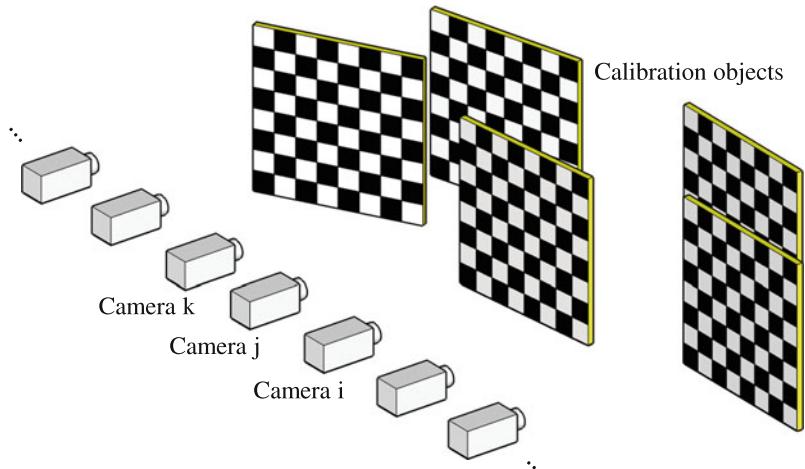
If there is no camera sharing its FOV with others, it is challenging to establish the common global coordinate system. One idea is to use a mobile robot carrying a calibration object [17]. The location of the object is estimated by the SLAM of the mobile robot. However, this is not stable enough for practical use yet.

Application

Multi-camera calibration is essential in constructing a system using multiple cameras depending on the

Calibration of Multi-camera

Setups, Fig. 3 Calibration of a camera network



applications and sensor configurations. The *inward-looking configuration* is generally used in many 3D reconstruction tasks such as stereo and visual hull reconstruction. The *outward-looking configuration* is useful to enlarge the effective FOV of the system, and especially for structure from motion applications. The most general *camera network* has diverse applications such as 3D reconstruction, surveillance, environmental monitoring and so on. Note that the camera network includes the inward-looking configuration of cameras.

References

- Zhang Z (2000) A flexible new technique for camera calibration. *IEEE Trans Pattern Anal Mach Intell* 22(11): 1330–1334
- Dementhon DF, Davis LS (1995) Model-based object pose in 25 lines of code. *Int J Comput Vis* 15:123–141
- Quan L, Lan Z (1999) Linear n-point camera pose determination. *IEEE Trans Pattern Anal Mach Intell* 21(8): 774–780
- Ababsa F, Mallem M (2004) Robust camera pose estimation using 2d fiducials tracking for real-time augmented reality systems. Proceedings of the 2004 ACM SIGGRAPH international conference on virtual reality continuum and its applications in industry. ACM, New York, pp 431–435
- Sinha S, Pollefeys M, McMillan L (2004) Camera network calibration from dynamic silhouettes. Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), vol 1. IEEE, Silver Spring, pp I–195
- Esquivel S, Woelk F, Koch R (2007) Calibration of a multi-camera rig from non-overlapping views. Proceedings of the 29th DAGM conference on pattern recognition. Springer, Berlin/New York, pp 82–91
- Andreff N, Horaud R, Espiau B (1999) On-line hand-eye calibration. Proceedings of the second international conference on 3-D digital imaging and modeling, 1999. Ottawa, Canada, IEEE, pp 430–436
- Dornaika F, Horaud R (1998) Simultaneous robot-world and hand-eye calibration. *IEEE Trans Robot Autom* 14(4): 617–622
- Park F, Martin B (1994) Robot sensor calibration: solving $ax = xb$ on the euclidean group. *IEEE Trans Robot Autom* 10(5):717–721
- Dai Y, Trampf J, Li H, Barnes N, Hartley R (2010) Rotation averaging with application to camera-rig calibration. Asian conference on computer vision. Springer, Berlin/New York, pp 335–346
- Triggs B, McLauchlan P, Hartley R, Fitzgibbon A (2000) Bundle adjustment – modern synthesis. *Vision algorithms: theory and practice*. Springer-Verlag, London, UK, pp 298–372
- Lebraly P, Royer E, Ait-Aider O, Deymier C, Dhome M (2011) Fast calibration of embedded non-overlapping cameras. IEEE international conference on robotics and automation (ICRA). Shanghai, China, IEEE, pp 221–227
- Kumar R, Ilie A, Frahm J, Pollefeys M (2008) Simple calibration of non-overlapping cameras with a mirror. IEEE conference on computer vision and pattern recognition (CVPR). Anchorage, AK, USA, IEEE, pp 1–7
- Devarajan D, Cheng Z, Radke R (2008) Calibrating distributed camera networks. *Proc IEEE* 96(10): 1625–1639
- Baker P, Aloimonos Y (2000) Complete calibration of a multi-camera network. Proceedings of the IEEE workshop on omnidirectional vision, 2000. Hilton Head Island, SC, USA, IEEE, pp 134–141
- Svoboda T, Martinec D, Pajdla T (2005) A convenient multicamera self-calibration for virtual environments. *Presence* 14(4):407–422
- Rekleitis I, Dudek G (2005) Automated calibration of a camera sensor network. IEEE/RSJ international conference on intelligent robots and systems. Edmonton, Canada, IEEE, pp 3384–3389



Calibration of Projective Cameras

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Synonyms

Lens distortion correction

Related Concepts

► Calibration; ► Camera Calibration; ► Geometric Calibration; ► Perspective Camera

Definition

Calibration of a projective camera is the process of determining an adjustment on the camera so that, after adjustment, it follows the pinhole or perspective projection model.

Background

A projective camera follows pinhole or perspective projection, which is also known as rectilinear projection because straight lines in a scene remain straight in an image. A real camera usually uses lenses with finite aperture, especially for low-end cameras (such as WebCams) or wide-angle cameras. Lens distortion also arises from imperfect lens design and manufacturing, as well as camera assembly. A line in a scene is not seen as a line in the image. A point in 3D space, its corresponding point in image, and the camera's optical center are not collinear. The linear projective equation is sometimes not sufficient, and lens distortion has to be considered or corrected beforehand.

Theory

According to [1], there are four steps in camera projection including lens distortion:

Step 1: *Rigid transformation* from world coordinate system (X_w, Y_w, Z_w) to camera one (X, Y, Z) :

$$[X \quad Y \quad Z]^T = \mathbf{R} [X_w \quad Y_w \quad Z_w]^T + \mathbf{t}$$

Step 2: *Perspective projection* from 3D camera coordinates (X, Y, Z) to *ideal* image coordinates (x, y) under pinhole camera model:

$$x = f \frac{X}{Z}, \quad y = f \frac{Y}{Z}$$

where f is the effective focal length.

Step 3: *Lens distortion*:

$$\check{x} = x + \delta_x, \quad \check{y} = y + \delta_y$$

where (\check{x}, \check{y}) are the *distorted* or *true* image coordinates and (δ_x, δ_y) are distortions applied to (x, y) . Note that the lens distortion described here is different from Tsai's treatment. Here, we go from ideal to real image coordinates, similar to [2].

Step 4: *Affine transformation* from real image coordinates (\check{x}, \check{y}) to *frame buffer* (pixel) image coordinates (\check{u}, \check{v}) :

$$\check{u} = d_x^{-1} \check{x} + u_0, \quad \check{v} = d_y^{-1} \check{y} + v_0,$$

where (u_0, v_0) are coordinates of the principal point and d_x and d_y are distances between adjacent pixels in the horizontal and vertical directions, respectively.

There are two types of distortions:

Radial distortion: It is symmetric; ideal image points are distorted along radial directions from the distortion center. This is caused by imperfect lens shape.

Decentering distortion: This is usually caused by improper lens assembly; ideal image points are distorted in both radial and tangential directions.

The reader is referred to [3–6] for more details.

The distortion can be expressed as power series in radial distance $r = \sqrt{x^2 + y^2}$:

$$\begin{aligned} \delta_x &= x(k_1 r^2 + k_2 r^4 + k_3 r^6 + \dots) \\ &\quad + [p_1(r^2 + 2x^2) + 2p_2xy](1 + p_3r^2 + \dots), \end{aligned}$$

$$\delta_y = y(k_1r^2 + k_2r^4 + k_3r^6 + \dots) + [2p_1xy + p_2(r^2 + 2y^2)](1 + p_3r^2 + \dots),$$

where k_i s are coefficients of radial distortion and p_j s are coefficients of decentering distortion.

Based on the reports in the literature [1, 2, 4], it is likely that the distortion function is totally dominated by the radial components and especially dominated by the first term. It has also been found that any more elaborated modeling not only would not help (negligible when compared with sensor quantization) but also would cause numerical instability [1, 2].

Denote the ideal pixel image coordinates by $u = x/d_x$ and $v = y/d_y$. By combining Steps 3 and 4 and if only using the first two radial distortion terms, we obtain the following relationship between (\check{u}, \check{v}) and (u, v) :

$$\check{u} = u + (u - u_0)[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2] \quad (1)$$

$$\check{v} = v + (v - v_0)[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2]. \quad (2)$$

Lens distortion parameters can be determined as an integrated part of geometric calibration [7]. This can be done by observing a known 3D target [1, 2, 6], by observing a 2D planar pattern [8], by observing a linear point pattern [9], or by moving the camera through a rigid scene [10]. The nonlinearity of the integrated projection and lens distortion model does not allow for a direct calculation of all the parameters of the camera model. Camera calibration including lens distortion can be performed by minimizing the distances between the image points and their predicted positions, i.e.,

$$\min_{\mathbf{A}, \mathbf{R}, \mathbf{t}, k_1, k_2} \sum_i \|\mathbf{m}_i - \check{\mathbf{m}}(\mathbf{A}, \mathbf{R}, \mathbf{t}, k_1, k_2, \mathbf{M}_i)\|^2 \quad (3)$$

where $\check{\mathbf{m}}(\mathbf{A}, \mathbf{R}, \mathbf{t}, k_1, k_2, \mathbf{M}_i)$ is the projection of \mathbf{M}_i onto the image according to the pinhole model, followed by distortion according to Eq. (1) and Eq. (2). The minimization is performed through an iterative approach such as using the Levenberg-Marquardt method.

An alternative approach is to perform lens distortion correction as a separate process. Invariance properties under projective transformation are exploited. One is the “plumb line” constraint [4], which is based on the

fact that a line in a scene remains a line in an image. Another is the cross-ratio constraint [2], which states that, for four collinear points with known distances between each other in 3D, their corresponding image points are collinear, and their cross-ratio remains the same. Due to lens distortion, projective invariants are not preserved, and we can use the variance to compute the distortion.

References

1. Tsai RY (1987) A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. IEEE J Robot Autom 3(4):323–344
2. Wei G, Ma S (1994) Implicit and explicit camera calibration: theory and experiments. IEEE Trans Pattern Anal Mach Intell 16(5):469–480
3. Slama CC (ed) (1980) Manual of photogrammetry, 4th edn. American Society of Photogrammetry, Falls Church
4. Brown DC (1971) Close-range camera calibration. Photogramm Eng 37(8):855–866
5. Faig W (1975) Calibration of close-range photogrammetry systems: mathematical formulation. Photogramm Eng Remote Sens 41(12):1479–1486
6. Weng J, Cohen P, Herniou M (1992) Camera calibration with distortion models and accuracy evaluation. IEEE Trans Pattern Anal Mach Intell 14(10):965–980
7. Zhang Z (2004) Camera calibration. In: Medioni G, Kang S (eds) Emerging topics in computer vision. Prentice Hall Professional Technical Reference, Upper Saddle River, pp 4–43
8. Zhang Z (2000) A flexible new technique for camera calibration. IEEE Trans Pattern Anal Mach Intell 22(11): 1330–1334
9. Zhang Z (2004) Camera calibration with one-dimensional objects. IEEE Trans Pattern Anal Mach Intell 26(7): 892–899
10. Zhang Z (1996) On the epipolar geometry between two images with lens distortion. In: International conference on pattern recognition, Vienna, Austria, vol I, pp 407–411

Calibration of Radiometric Falloff (Vignetting)

Stephen Lin

Microsoft Research Asia, Beijing Sigma Center,
Beijing, China

Synonyms

Vignetting estimation



Related Concepts

- [Irradiance](#)
- [Radiometric Calibration](#)
- [Radiance](#)
- [Vignetting](#)

Definition

Calibration of radiometric falloff is the measurement of brightness attenuation away from the image center for a given camera, lens, and camera settings.

Background

Several mechanisms may be responsible for radiometric falloff. One is the optics of the camera, which may have a smaller effective lens opening for light incident at greater off-axis angles (i.e., irradiance toward the edges of an image). Radiometric falloff also occurs naturally due to foreshortening of the lens when viewed at increasing angles from the optical axis. A third cause is mechanical in nature, where light arriving at oblique angles is partially obstructed by camera components such as the field stop or lens rim. Digital sensors may also contribute to this falloff because of angle-dependent sensitivity to light. The profile of the radiometric falloff field varies with respect to camera, lens, and camera settings such as focal length and aperture.

Many computer vision algorithms assume that the image irradiance measured at the camera sensor is equal to the scene radiance that arrives at the camera. However, this assumption often does not hold because of radiometric falloff. It is therefore important to measure or estimate the radiometric falloff, and remove its effects from images.

Theory

Radiometric falloff, or vignetting, may be modeled as a function f that represents the proportion of image brightness I at an image position (x, y) relative to that at the image center (x_0, y_0) :

$$f(x, y) = \frac{I(x, y)}{I(x_0, y_0)}. \quad (1)$$

Because of approximate radial symmetry in the optical systems of most cameras, the radiometric falloff function may alternatively be expressed in terms of image distance r from the image center:

$$f(r) = \frac{I(r)}{I(0)}, \quad (2)$$

where $r = \sqrt{x^2 + y^2}$. The purpose of radiometric falloff calibration is to recover f , so that its inverse function f^{-1} can be applied to an image i , recorded by the same camera and camera settings, to obtain an image \tilde{i} without radiometric falloff:

$$\tilde{i}(x, y) = f^{-1}(i(x, y)). \quad (3)$$

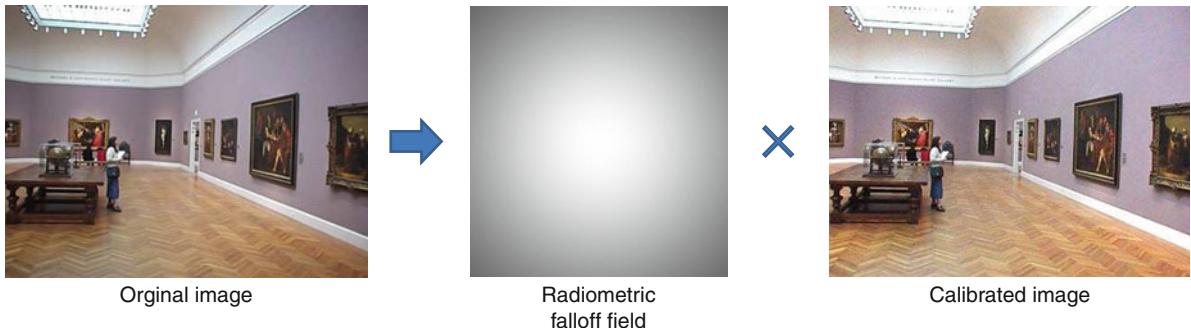
The effect of radiometric falloff calibration is illustrated in [Fig. 1](#).

Methods

A basic method for calibration of radiometric falloff is to capture a reference image consisting of a uniform radiance field [\[1–4\]](#). Since the scene itself contains no brightness variation, intensity differences in the image can be attributed solely to radiometric falloff. It must be noted that in these and other methods of falloff calibration, it is assumed that the camera response function is known.

Another approach examines image sequences with overlapping views of an arbitrary static scene [\[5–9\]](#). In overlapping image regions, corresponding points are assumed to have the same scene radiance. Differences in their intensities are therefore a result of different radiometric falloff at their respective image positions. From the positions and relative intensities among each set of corresponding points, the radiometric falloff field can be recovered without knowledge of scene content. Most of these methods are designed to recover both the radiometric falloff field and the camera response function in a joint manner [\[6–9\]](#).

The radiometric falloff field may alternatively be estimated from a single arbitrary input image. To infer the falloff field in this case, the intensity variation caused by falloff needs to be distinguished from that due to scene content. This has been done using a segmentation-based approach that identifies image regions with reliable data for falloff estimation [\[10\]](#), and by examining the effect of falloff on radial gradient distributions in the image [\[11\]](#).



Calibration of Radiometric Falloff (Vignetting), Fig. 1 Image calibrated for radiometric falloff

Application

Calibration of radiometric falloff is of importance to algorithms such as shape-from-shading and photometric stereo that infer scene properties from image irradiance values. It is also essential in applications such as image mosaicing and segmentation that require photometric consistency of the same scene point appearing in different images, or different scene points within the same image.

A measured radiometric falloff field may be used to locate the optical center of the image, since radiometric falloff generally exhibits radial symmetry. The spatial variation of light transmission may also be exploited in sensing, such as to capture high dynamic range intensity values of scene points viewed from a moving camera [12].

References

1. Sawchuk AA (1977) Real-time correction of intensity nonlinearities in imaging systems. *IEEE Trans Comput* 26(1):34–39
2. Asada N, Amano A, Baba M (1996) Photometric calibration of zoom lens systems. In: IEEE international conference on pattern recognition, Washington DC, pp 186–190
3. Kang SB, Weiss R (2000) Can we calibrate a camera using an image of a flat textureless lambertian surface? In: European conference on computer vision 2000 (ECCV), vol II. Springer, London, pp 640–653
4. Yu W (2004) Practical anti-vignetting methods for digital cameras. *IEEE Trans Consum Electron* 50:975–983
5. Jia J, Tang CK (2005) Tensor voting for image correction by global and local intensity alignment. *IEEE Trans Pattern Anal Mach Intell* 27(1):36–50
6. Litvinov A, Schechner YY (2005) Addressing radiometric nonidealities: a unified framework. In: IEEE computer

vision and pattern recognition (CVPR), IEEE Computer Society, Silver Spring, pp 52–59

7. Litvinov A, Schechner YY (2005) A radiometric framework for image mosaicing. *J Opt Soc Am A* 22:839–848
8. Goldman DB, Chen JH (2005) Vignette and exposure calibration and compensation. In: IEEE international conference on computer vision, Beijing, pp 899–906
9. Kim SJ, Pollefeys M (2008) Robust radiometric calibration and vignetting correction. *IEEE Trans Pattern Anal Mach Intell* 30(4):562–576
10. Zheng Y, Lin S, Kambhamettu C, Yu J, Kang SB (2009) Single-image vignetting correction. *IEEE Trans Pattern Anal Mach Intell* 31(12):2243–2256
11. Zheng Y, Yu J, Kang SB, Lin S, Kambhamettu C (2008) Single-image vignetting correction using radial gradient symmetry. *IEEE Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos
12. Schechner YY, Nayar SK (2003) Generalized mosaicing: high dynamic range in a wide field of view. *Int J Comput Vis* 53(3):245–267

Camera Calibration

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Related Concepts

- [Calibration](#); ► [Calibration of Projective Cameras](#);
- [Geometric Calibration](#)

Definition

Camera calibration is the process of determining certain parameters of a camera in order to fulfill desired



tasks with specified performance measures. The reader is referred to entry [►Calibration](#) for a general discussion on calibration.

Background

There are multiple camera calibration problems. The most common one, which almost becomes the synonym of camera calibration, is geometric calibration (see entry [►Geometric Calibration](#)). It consists in determining the intrinsic and extrinsic parameters of a camera.

Other camera calibration problems include:

Stereo calibration. A stereo (or stereovision) system consists of multiple cameras. Stereo calibration determines the relative geometry (rotation and translation) between cameras. The intrinsic parameters of each camera can be determined separately as in camera calibration or jointly with the relative geometry.

Photometric calibration. Photometry concerns the measurement of quantities associated with light. Photometric calibration of a camera is a process of determining a function which converts the pixel values to photometric quantities such as SI (*Système International* in French) light units. A test chart of patches with known relative luminances is usually used for photometric calibration.

Color calibration. The pixel values of a color camera depend not only on the surface reflectance but also on the illuminating source. White balance is a common color calibration task, which uses a standard test target with known reflectance to remove the influence of lighting on the scene. Another common task is to calibrate multiple seemingly identical cameras which are not due to tolerance in fabrication.

Camera Extrinsic Parameters

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Synonyms

Camera pose; Extrinsic parameters

Related Concepts

[►Camera Parameters \(Intrinsic, Extrinsic\)](#); [►Intrinsics](#)

Definition

extrinsic, short for *extrinsic parameters*, refer to the parameters not forming the essential part of a thing, which is usually a camera in computer vision. The extrinsic parameters of a camera include its pose (rotation and translation) with respect to a reference coordinate system.

See entry “[►Camera Parameters \(Intrinsic, Extrinsic\)](#)” for more details.

Camera Model

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Related Concepts

[►Camera Calibration](#); [►Camera Parameters \(Intrinsic, Extrinsic\)](#); [►Intrinsics](#); [►Perspective Camera](#)

Definition

A *camera* is a device that records lights coming from the scene and saves them in images. These images may be still photographs or moving images (videos). The *camera model* describes the mathematical relationship between the 3D coordinates of a point in the scene from which the light comes from and the 2D coordinates of its projection onto the image plane. The ideal camera model is known as the pin-hole camera model or perspective camera model, but other camera models exist such as thin and thick cameras.

Background

The term *camera* comes from the *camera obscura* (in Latin for “dark chamber”) [1]. A camera obscura is a dark room, consisting of a darkened chamber or box, into which light is admitted through a pinhole (later a

convex lens), forming an image of external objects on a surface of wall, paper, or glass. A modern camera generally consists of an enclosed hollow with an opening (aperture) at one end for light to enter and a recording or viewing surface (such as a CCD or CMOS sensor) for capturing the light on the other end. A majority of cameras have a lens positioned in front of the camera's opening to gather the incoming light and focus all or part of the image on the recording surface.

A camera may work with the light of the visible spectrum. If it records each of the red, blue, and green primary colors at each pixel, then the camera is called a color camera; if it only records the shades of black and white (the grey levels of the light intensity), the camera is called a black and white camera.

A camera may also work with the light outside of the visible spectrum, e.g., with the infrared (IR) light, and the camera is called the IR camera.

Pinhole Camera Model

A pinhole camera can be ideally modeled as the *perspective projection*. This is by far the most popularly used model in the computer vision community. The reader is referred to the entry ▶ [Perspective Camera](#) for details.

Thin and Thick Lens Camera Models

Although pinhole cameras model quite well most of the cameras we use in the computer vision community, they cannot be used physically in a real imaging system. This is for two reasons:

- An ideal pinhole, having an infinitesimal aperture, does not allow to gather enough amount of light to produce measurable image brightness (called *image irradiance*).
- Because of the wave nature of light, diffraction occurs at the edge of the pinhole and the light spread over the image [2]. As the pinhole is made smaller and smaller, a larger and larger fraction of the incoming light is deflected far from the direction of the incoming ray.

To avoid these problems, a real imaging system usually uses lenses with finite aperture. This appendix aims at having the reader know that there are other camera models available. One should choose an

appropriate model for a particular imaging device [3, Sect. 2.A.1].

For an ideal lens, which is known as the *thin lens*, all optical rays parallel to the optical axis converge to a point on the optical axis on the other side of the lens at a distance equal to the so-called *focal length* f (see Fig. 1).

The light ray through the center of the lens is undeflected; thus a thin lens produces the same projection as the pinhole. However, it gathers also a finite amount of light reflected from (or emitted by) the object (see Fig. 2). By the familiar *thin lens law*, rays from points at a distance Z are focused by the lens at a distance $-F$, and Z and $-F$ satisfy

$$\frac{1}{Z} + \frac{1}{-F} = \frac{1}{-f}, \quad (1)$$

where f is the focal length.

If we put an image plane at the distance $-F$, then points at other distances than Z are imaged as small blur circles. This can be seen by considering the cone of light rays passing through the lens with apex at the point where they are correctly focused [2]. The size of the blur circle can be determined as follows. A point at distance \hat{Z} is focused if it is imaged at a point $-\hat{F}$ from the lens (see Fig. 3), where

$$\frac{1}{\hat{Z}} + \frac{1}{-\hat{F}} = \frac{1}{-f}.$$

It gives rise to a blur circle on the image plane located at distance $-\hat{F}$. The diameter of the blur circle, e , can be computed by triangle similarity

$$\frac{e}{d} = \frac{|F - \hat{F}|}{\hat{F}},$$

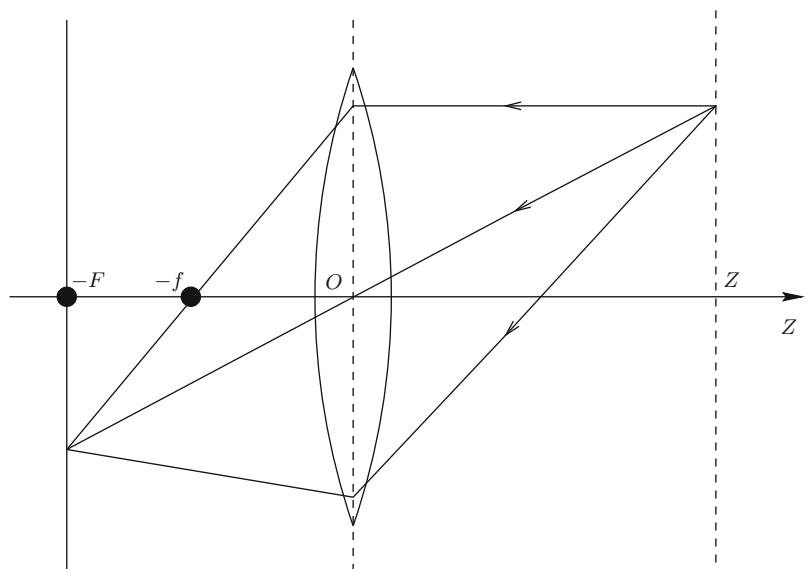
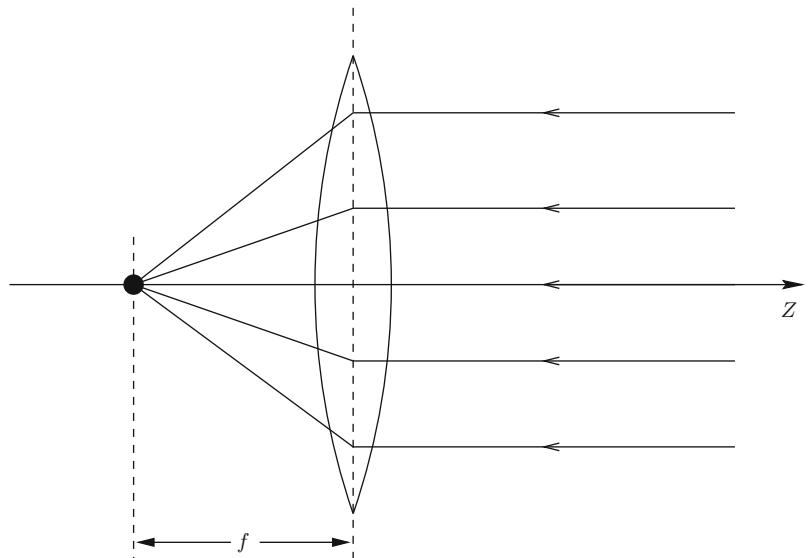
which gives

$$e = \frac{d}{\hat{F}} |F - \hat{F}| = \frac{fd}{\hat{Z}} \frac{|Z - \hat{Z}|}{Z - f},$$

where d is the diameter of the lens. If the diameter of blur circles, e , is less than the resolution of the image, then the object is well focused and its image is clean. The range of distances over which objects are focused "sufficiently well" is called the *depth of field*. It is clear

**Camera Model, Fig. 1**

Cross-sectional view of a thin lens sliced by a plane containing the optical axis. All light rays parallel to the optical axis converge to a point at a distance equal to the focal length



Camera Model, Fig. 2 A thin lens gathers light from a finite area and produces a well-focused image at a particular distance

that the larger the lens aperture d , the less the depth of field.

From (1), it is seen that for objects relatively distant from the lens (*i.e.*, $Z \gg f$), we have $F = f$. If the image plane is located at distance f from the lens, then the camera can be modeled reasonably well by the pinhole.

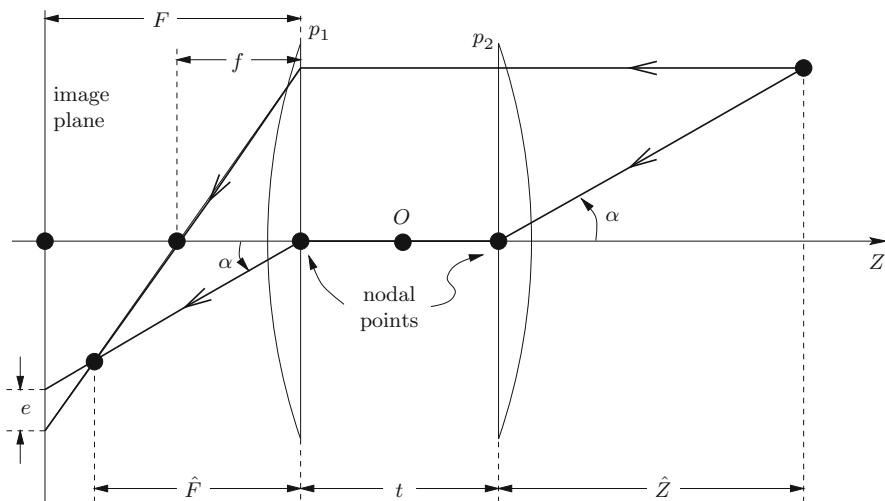
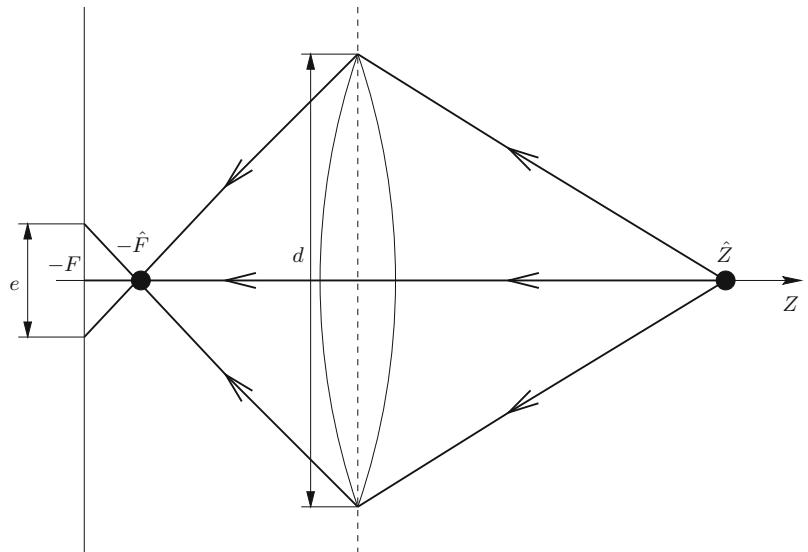
It is difficult to manufacture a perfect lens. In practice, several simple lenses are carefully assembled to make a compound lens with better properties. In an imaging device with mechanism of focus and zoom, the lenses are allowed to move. It appears difficult to

model such a device by a pinhole or thin lens. Another model, called the *thick lens*, is used by more and more researchers [4, 5].

An ideal thick lens is illustrated in Fig. 4. It is composed of two lenses, each having two opposite surfaces, one spherical and the other plane. These two planes p_1 and p_2 , called the *principal planes*, are perpendicular to the optical axis and are separated by a distance t , called the *thickness of the lens*. The principal planes intersect the optical axis at two points, called the *nodal points*. The thick lens produces the same perspective projection as the ideal thin lens,

Camera Model, Fig. 3

Focus and blur circles

**Camera Model, Fig. 4** Cross-sectional view of a thick lens sliced by a plane containing the optical axis

except for *an additional offset* equal to the lens thickness t along the optical axis. A light ray arriving at the first nodal point leaves the rear nodal point without changing direction. A thin lens can then be considered as a thick lens with $t = 0$.

It is thus clear that a thick lens can be considered as a thin lens if the object is relatively distant to the camera compared to the lens thickness (*i.e.*, $\hat{Z} \gg t$). It can be further approximated by a pinhole only when the object is well focused (*i.e.*, $F \approx \hat{F}$), and this is valid only locally.

References

1. Wikipedia (2011) History of the camera. http://en.wikipedia.org/wiki/History_of_the_camera
2. Horn BKP (1986) Robot vision. MIT, Cambridge
3. Xu G, Zhang Z (1996) Epipolar geometry in stereo, motion and object recognition. Kluwer Academic, Dordrecht
4. Pahlavan K, Uhlin T, Ekhlund JO (1993) Dynamic fixation. In: Proceedings of the 4th international conference on computer vision, Berlin, Germany. IEEE Computer Society, Los Alamitos, pp 412–419
5. Lavest J (1992) Stéréovision axiale par zoom pour la robotique. Ph.D. thesis, Université Blaise Pascal de Clermont-Ferrand, France



Camera Parameters (Internal, External)

► Camera Parameters (Intrinsic, Extrinsic)

Camera Parameters (Intrinsic, Extrinsic)

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Synonyms

Camera model; Camera parameters (internal, external)

Related Concepts

► Camera Calibration; ► Calibration of Projective Cameras; ► Camera Parameters (Intrinsic, Extrinsic); ► Depth Distortion; ► Perspective Transformation; ► Perspective Transformation

Definition

Camera parameters are the parameters used in a camera model to describe the mathematical relationship between the 3D coordinates of a point in the scene from which the light comes from and the 2D coordinates of its projection onto the image plane. The *intrinsic parameters*, also known as *internal parameters*, are the parameters intrinsic to the camera itself, such as the focal length and lens distortion. The *extrinsic parameters*, also known as *external parameters* or *camera pose*, are the parameters used to describe the transformation between the camera and its external world.

Background

In computer vision, in order to understand the environment surrounding us with a camera, we have to know first the camera parameters. Depending on the accuracy we need to achieve and on the quality of the

camera, some parameters can be neglected. For example, with a high-quality camera, the lens distortion can usually be ignored in most of the applications.

Theory

In the entry ► [Perspective Camera](#), we describe the mathematical model of a perspective camera with only a single parameter, the focal length f . The relationship between a 3D point and its image projection is described by

$$s\tilde{\mathbf{m}} = \mathbf{P}\tilde{\mathbf{M}}, \quad (1)$$

where $s = S$ is an arbitrary nonzero scalar and \mathbf{P} is a *projective projection matrix* given by

$$\mathbf{P} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Before proceeding, the reader needs to review the entry ► [Perspective Camera](#).

Extrinsic Parameters

In the projective projection matrix described in the entry ► [Perspective Camera](#), recapitulated above, we assumed that 3D points are expressed in the camera coordinate system. In practice, they can be expressed in any 3D coordinate system, which is sometimes referred as the *world coordinate system*. As shown in Fig. 1, we go from the old coordinate system centered at the optical center C (camera coordinate system) to the new coordinate system centered at point O (world coordinate system) by a rotation \mathbf{R} followed by a translation $\mathbf{t} = CO$. Then, for a single point, its coordinates expressed in the camera coordinate system, M_c , and those expressed in the world coordinate system, M_w , are related by

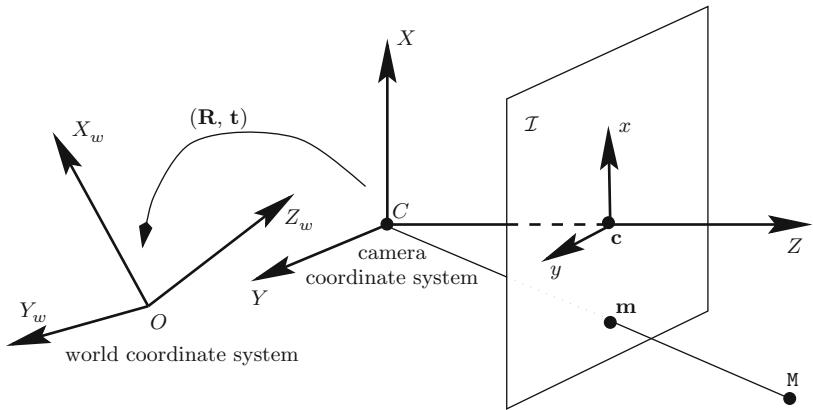
$$\mathbf{M}_c = \mathbf{RM}_w + \mathbf{t},$$

or more compactly

$$\tilde{\mathbf{M}}_c = \mathbf{D}\tilde{\mathbf{M}}_w, \quad (2)$$

Camera Parameters**(Intrinsic, Extrinsic), Fig. 1**

World coordinate system and camera extrinsic parameters



where \mathbf{D} is a Euclidean transformation of the three-dimensional space

$$\mathbf{D} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_3^T & 1 \end{bmatrix} \quad \text{with } \mathbf{0}_3 = [0, 0, 0]^T. \quad (3)$$

The matrix \mathbf{R} and the vector \mathbf{t} describe the orientation and position of the camera with respect to the new world coordinate system. They are called the *extrinsic parameters* of the camera.

From (1) and (2), we have

$$\tilde{\mathbf{m}} = \mathbf{P}\tilde{\mathbf{M}}_c = \mathbf{P}\mathbf{D}\tilde{\mathbf{M}}_w.$$

Therefore the new perspective projection matrix is given by

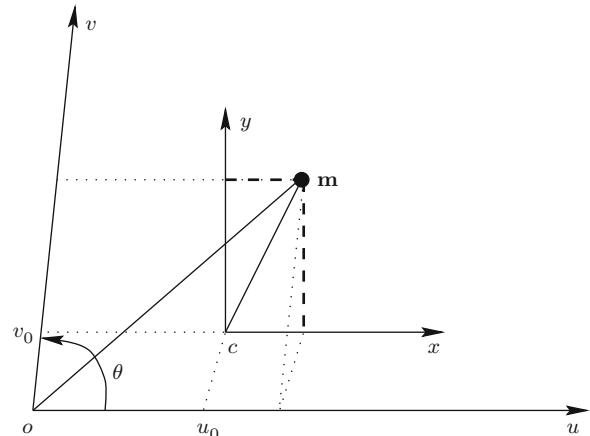
$$\mathbf{P}_{\text{new}} = \mathbf{P}\mathbf{D}. \quad (4)$$

This tells us how the perspective projection matrix \mathbf{P} changes when we change coordinate systems in the three-dimensional space: We simply multiply it on the right by the corresponding Euclidean transformation.

Intrinsic Parameters and Normalized Camera

This section considers the transformation in image coordinate systems. It is very important in practical applications because:

- We do not know the origin of the image plane in advance. It generally does not coincide with the intersection of the optical axis and the image plane.
- The units of the image coordinate axes are not necessarily equal, and they are determined by the sampling rates of the imaging devices.



Camera Parameters (Intrinsic, Extrinsic), Fig. 2 Camera intrinsic parameters

- The two axes of a real image may not form a right angle.

To handle these effects, we introduce an affine transformation.

Consider Fig. 2. The original image coordinate system \$(c, x, y)\$ is centered at the principal point \$c\$ and has the same units on both \$x\$- and \$y\$-axes. The coordinate system \$(o, u, v)\$ is the coordinate system in which we address the pixels in an image. It is usually centered at the upper left corner of the image, which is usually not the principal point \$c\$. Due to the electronics of acquisition, the pixels are usually not square. Without loss of generality, the \$u\$-axis is assumed to be parallel to the \$x\$-axis. The units along the \$u\$- and \$v\$-axes are assumed to be \$k_u\$ and \$k_v\$ with respect to the unit used in \$(c, x, y)\$. The \$u\$- and \$v\$-axes may not be exactly orthogonal, and we denote their angle by \$\theta\$. Let the coordinates of the

principal point c in (o, u, v) be $[u_0, v_0]^T$. These five parameters do not depend on the position and orientation of the cameras and are thus called the camera *intrinsic parameters*.

For a given point, let $\mathbf{m}_{\text{old}} = [x, y]^T$ be the coordinates in the original coordinate system; let $\mathbf{m}_{\text{new}} = [u, v]^T$ be the pixel coordinates in the new coordinate system. It is easy to see that

$$\tilde{\mathbf{m}}_{\text{new}} = \mathbf{H} \tilde{\mathbf{m}}_{\text{old}},$$

where

$$\mathbf{H} = \begin{bmatrix} k_u & k_u \cot \theta & u_0 \\ 0 & k_v / \sin \theta & v_0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Since, according to (1), we have

$$s \tilde{\mathbf{m}}_{\text{old}} = \mathbf{P}_{\text{old}} \tilde{\mathbf{M}},$$

we conclude that

$$s \tilde{\mathbf{m}}_{\text{new}} = \mathbf{H} \mathbf{P}_{\text{old}} \tilde{\mathbf{M}},$$

and thus

$$\mathbf{P}_{\text{new}} = \mathbf{H} \mathbf{P}_{\text{old}} = \begin{bmatrix} f k_u & f k_u \cot \theta & u_0 & 0 \\ 0 & f k_v / \sin \theta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (5)$$

Note that it depends on the products $f k_u$ and $f k_v$, which means that a change in the focal length and a change in the pixel units are indistinguishable. We thus introduce two parameters $\alpha_u = f k_u$ and $\alpha_v = f k_v$.

We will now define a special coordinate system that allows us to normalize the image coordinates [1]. This coordinate system is called the *normalized coordinate system* of the camera. In this “normalized” camera, the image plane is located at a unit distance from the optical center (*i.e.* $f = 1$). The perspective projection matrix of the normalized camera is given by

$$\mathbf{P}_N = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (6)$$

For a world point $[X, Y, Z]^T$ expressed in the camera coordinate system, its normalized coordinates are

$$\begin{aligned} \hat{x} &= \frac{X}{Z} \\ \hat{y} &= \frac{Y}{Z}. \end{aligned} \quad (7)$$

A matrix \mathbf{P} defined by (5) can be decomposed into the product of two matrices:

$$\mathbf{P}_{\text{new}} = \mathbf{A} \mathbf{P}_N, \quad (8)$$

where

$$\mathbf{A} = \begin{bmatrix} \alpha_u & \alpha_u \cot \theta & u_0 \\ 0 & \alpha_v / \sin \theta & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (9)$$

The matrix \mathbf{A} contains only the intrinsic parameters and is called *camera intrinsic matrix*. It is thus clear that the normalized image coordinates are given by

$$\begin{bmatrix} \hat{x} \\ \hat{y} \\ 1 \end{bmatrix} = \mathbf{A}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}. \quad (10)$$

Through this transformation from the available pixel image coordinates, $[u, v]^T$, to the imaginary normalized image coordinates, $[\hat{x}, \hat{y}]^T$, the projection from the space onto the normalized image does not depend on the specific cameras. This frees us from thinking about characteristics of the specific cameras and allows us to think in terms of ideal systems in stereo, motion, and object recognitions.

The General Form of Perspective Projection Matrix

The camera can be considered as a system that depends upon the intrinsic and the extrinsic parameters. There are five intrinsic parameters: the scale factors α_u and α_v , the coordinates u_0 and v_0 of the principal point, and the angle θ between the two image axes. There are six extrinsic parameters, three for the rotation and three for the translation, which define the transformation from the world coordinate system to the standard coordinate system of the camera.

Combining (4) and (8) yields the general form of the perspective projection matrix of the camera:

$$\mathbf{P} = \mathbf{A} \mathbf{P}_N \mathbf{D} = \mathbf{A} [\mathbf{R} \ \mathbf{t}]. \quad (11)$$

The projection of 3D world coordinates $\mathbf{M} = [X, Y, Z]^T$ to 2D pixel coordinates $\mathbf{m} = [u, v]^T$ is then described by

$$s\tilde{\mathbf{m}} = \mathbf{P}\tilde{\mathbf{M}}, \quad (12)$$

where s is an arbitrary scale factor. Matrix \mathbf{P} has $3 \times 4 = 12$ elements but has only 11 degrees of freedom because it is defined up to a scale factor.

Let p_{ij} be the (i, j) entry of matrix \mathbf{P} . Eliminating the scalar s in (12) yields two nonlinear equations:

$$u = \frac{p_{11}X + p_{12}Y + p_{13}Z + p_{14}}{p_{31}X + p_{32}Y + p_{33}Z + p_{34}} \quad (13)$$

$$v = \frac{p_{21}X + p_{22}Y + p_{23}Z + p_{24}}{p_{31}X + p_{32}Y + p_{33}Z + p_{34}}. \quad (14)$$

Camera *calibration* is the process of estimating the intrinsic and extrinsic parameters of a camera or the process of first estimating the matrix \mathbf{P} and then deducing the camera parameters from \mathbf{P} . A wealth of work has been carried out in this domain by researchers either in photogrammetry [2, 3] or in computer vision and robotics [4–9] (see [10] for a review). The usual method of calibration is to compute camera parameters from one or more images of an object of *known size and shape*, for example, a flat plate with a regular pattern marked on it. From (12) or (13) and (14), we have two nonlinear equations relating 2D to 3D coordinates. This implies that each pair of an identified image point and its corresponding point on the calibration object provides two constraints on the intrinsic and extrinsic parameters of the camera. The number of unknowns is 11. It can be shown that, given N points ($N \geq 6$) in general position, the camera can be calibrated. The presentation of calibration techniques is beyond the scope of this book. The interested reader is referred to the above-mentioned references.

Once the perspective projection matrix \mathbf{P} is given, we can compute the coordinates of the optical center C of the camera in the world coordinate system. We first decompose the 3×4 matrix \mathbf{P} as the concatenation of a 3×3 submatrix \mathbf{B} and a 3-vector \mathbf{b} , that is, $\mathbf{P} = [\mathbf{B} \ \mathbf{b}]$. Assume that the rank of \mathbf{B} is 3. In the entry ▶ **Perspective Camera**, we explained that, under the pinhole model, the optical center projects to $[0, 0, 0]^T$ (*i.e.* $s = 0$). Therefore, the optical center can

be obtained by solving

$$\mathbf{P}\tilde{\mathbf{C}} = \mathbf{0}, \quad \text{that is, } [\mathbf{B} \ \mathbf{b}] \begin{bmatrix} C \\ 1 \end{bmatrix} = \mathbf{0}.$$

The solution is

$$\mathbf{C} = -\mathbf{B}^{-1}\mathbf{b}. \quad (15)$$

Given matrix \mathbf{P} and an image point \mathbf{m} , we can obtain the equation of the 3-D semi-line defined by the optical center C and point \mathbf{m} . This line is called the *optical ray* defined by \mathbf{m} . Any point on it projects to the single point \mathbf{m} . We already know that C is on the optical ray. To define it, we need another point. Without loss of generality, we can choose the point D such that the scale factor $s = 1$, that is,

$$\tilde{\mathbf{m}} = [\mathbf{B} \ \mathbf{b}] \begin{bmatrix} D \\ 1 \end{bmatrix}.$$

This gives $D = \mathbf{B}^{-1}(-\mathbf{b} + \tilde{\mathbf{m}})$. A point on the optical ray is thus given by

$$\mathbf{M} = C + \lambda(D - C) = \mathbf{B}^{-1}(-\mathbf{b} + \lambda\tilde{\mathbf{m}}),$$

where λ varies from 0 to ∞ .

References

1. Faugeras O (1993) Three-dimensional computer vision: a geometric viewpoint. MIT, Cambridge
2. Brown DC (1971) Close-range camera calibration. Photogramm Eng 37(8):855–866
3. Faig W (1975) Calibration of close-range photogrammetry systems: mathematical formulation. Photogramm Eng Remote Sens 41(12):1479–1486
4. Tsai R (1986) Multiframe image point matching and 3d surface reconstruction. IEEE Trans Pattern Anal Mach Intell 5:159–174
5. Faugeras O, Toscani G (1986) The calibration problem for stereo. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Miami Beach, FL. IEEE, Los Alamitos, pp 15–20
6. Lenz R, Tsai R (1987) Techniques for calibrating of the scale factor and image center for high accuracy 3D machine vision metrology. In: International conference on robotics and automation, Raleigh, NC, pp 68–75
7. Toscani G (1987) Système de Calibration optique et perception du mouvement en vision artificielle. Ph.D. thesis, Paris-Orsay
8. Wei G, Ma S (1991) Two plane camera calibration: a unified model. In: Proceedings of the IEEE conference on



- computer vision and pattern recognition (CVPR), Hawaii, pp 133–138
9. Weng J, Cohen P, Rebibo N (1992) Motion and structure estimation from stereo image sequences. IEEE Trans RA 8(3):362–382
 10. Tsai R (1989) Synopsis of recent progress on camera calibration for 3D machine vision. In: Khatib O, Craig JJ, Lozano-Pérez T (eds) The robotics review. MIT, Cambridge, pp 147–159

Camera Pose

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Synonyms

Camera extrinsic parameters

Related Concepts

- Camera Calibration; ► Camera Parameters (Intrinsic, Extrinsic); ► Intrinsic; ► Perspective Camera

Definition

Camera pose is referred to the position and orientation of a camera with respect to a reference coordinate system, which is usually known as the world coordinate system.

Background

Determining the camera pose is usually a first step toward perceiving the surrounding environment. In structure from motion where a camera is moving through the environment, one needs to determine the successive camera poses at different instants in order to reconstruct the surrounding environment in 3D. In a multi-camera (two or more) system, one needs to determine the relative camera pose, i.e., how one camera is related to other cameras.

The reader is referred to entry “► Camera Parameters (Intrinsic, Extrinsic)” for details.

Camera Response Function

- Radiometric Response Function

Camera-Shake Blur

- Motion Blur

Catadioptric Camera

Srikumar Ramalingam
Mitsubishi Electric Research Laboratories, Cambridge, MA, USA

Synonyms

- Catoptrics; ► Dioptrics

Related Concepts

- Center of Projection; ► Field of View;
- Omnidirectional Camera

Definition

A catadioptric system is a camera configuration where both lenses and mirrors are jointly used to achieve specialized optical properties. These configurations are referred to as *catadioptric*, where “cata” comes from mirrors (reflective) and “dioptric” comes from lenses (refractive).

Background

In 1637, René Descartes observed that the refractive and reflective “ovals” (conical lenses and mirrors) have the ability to focus light into one single point on illumination from a chosen point [1]. It was reported that the same results were derived by Feynman et al. [2] and



Drucker and Locke [3]. In computer vision community, Baker and Nayar presented the complete class of single viewpoint catadioptric configurations with detailed solutions and degenerate cases [4]. Some of these results have been independently derived by Bruckstein and Richardson [5]. Survey of various catadioptric cameras, a review and details of their calibration, and 3D reconstruction algorithms can also be found in [6, 7].

Theory and Classification

The combination of mirrors and lenses provides a wide range of design possibilities leading to interesting applications in computer vision. Most catadioptric configurations have larger field of view compared to conventional pinhole cameras. Other important design goals are compactness of a sensor, a single effective viewpoint, image quality, focusing properties, or a desired projection function. The catadioptric cameras may be classified in many ways. In [6], Sturm et al. classify the catadioptric cameras in to five different types:

- Single-mirror central systems, having a single effective viewpoint
- Central systems using multiple mirrors
- Noncentral systems
- Single-lens stereo systems
- Programmable devices

In what follows, catadioptric cameras are classified into central and noncentral systems. Most of these catadioptric configurations were proposed by researchers along with specified calibration and 3D reconstruction algorithms.

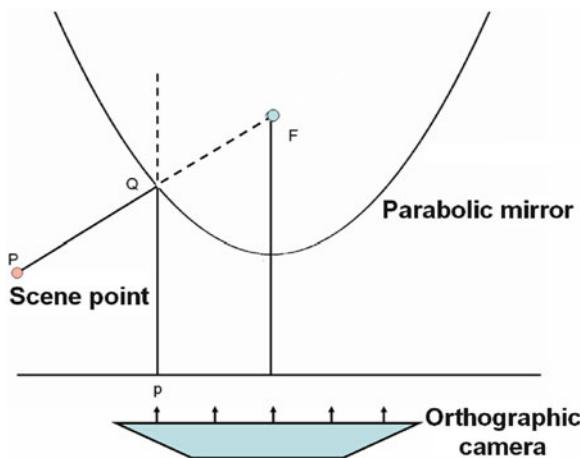
Central Catadioptric Configurations

It requires a very careful choice of the shape of the mirrors and their positioning to obtain a single effective viewpoint in catadioptric imaging. The single viewpoint design goal is important because it allows the generation of pure perspective images from the catadioptric images. Furthermore, it allows one to solve motion estimation and 3D reconstruction algorithms in the same way as perspective cameras:

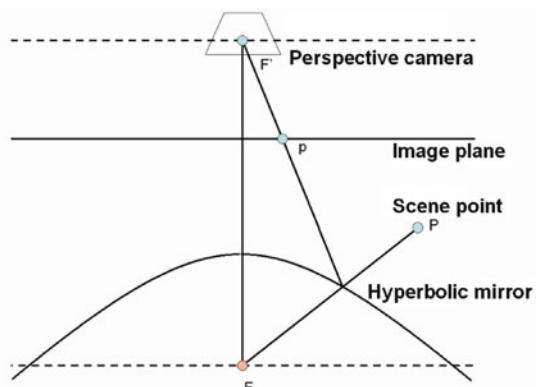
- *Planar mirror*: In [4, 8], it can be observed that by using planar mirrors along with a perspective camera, one can obtain a single viewpoint configuration.

Since planar mirrors do not increase the field of view of the system, they are not very interesting for building omnidirectional cameras. Using four planar mirrors in a pyramidal configuration along with four perspective cameras, Nalwa [9] produced an omnidirectional sensor of field of view of $360^\circ \times 50^\circ$. The optical centers of the four cameras and the angles made by the four planar faces are adjusted to obtain a single effective viewpoint for the system.

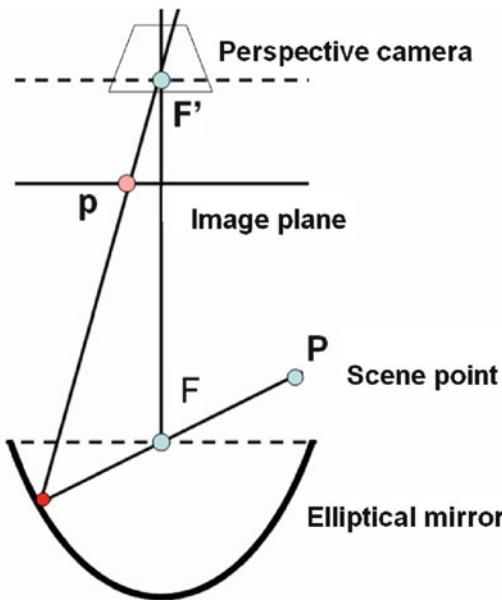
- *Conical mirrors*: By positioning the optical center of a perspective camera at the apex of a cone, one can obtain a single center configuration. Nevertheless, the only light rays reaching the camera after a reflection in the mirror are those grazing the cone. This case is thus not useful to enhance the field of view while conserving a single center of projection. However, in the work [10], it was proved that conical mirrors can be used to construct a non-degenerate single viewpoint omnidirectional cameras. The outer surface of the conical mirror forms a virtual image corresponding to the real scene behind the conical mirror. On placing the optical center of the pinhole camera at the vertex of the cone, the camera sees the world through the reflection on the outer surface of the mirror. In other words, the cone is not blocking the view. On the other hand, the cone is the view.
- *Spherical mirror*: If the optical center of a perspective camera is fixed at the center of a spherical mirror, one can obtain a single viewpoint configuration. Unfortunately, all that the perspective camera sees is its own reflection. As a result the spherical mirror produces a degenerate configuration without any advantage. Remember that by positioning the perspective camera outside the sphere, one can obtain a useful *noncentral* catadioptric camera.
- *Parabolic mirror*: Figure 1 shows a single viewpoint catadioptric system with a parabolic mirror and an orthographic camera. It is easier to study a catadioptric configuration by considering the back projection rather than the forward projection. Consider the back projection of an image point \mathbf{p} . The back-projected ray from the image pixel \mathbf{p} , starting from the optical center at infinity, is parallel to the axis of the parabolic mirror. This ray intersects and reflects from the surface of the mirror. The reflection is in accordance with the laws of reflection. This



Catadioptric Camera, Fig. 1 Parabolic mirror + orthographic camera [8]. **P** refers to the 3D scene point. **F**, the focus of the parabolic mirror, is the effective viewpoint



Catadioptric Camera, Fig. 3 Hyperbolic mirror + perspective camera [4]. **P** refers to the 3D scene point. **F** and **F'** refer to the two foci of the mirror and **p** refers to the image point. **F** is the effective viewpoint



Catadioptric Camera, Fig. 2 Elliptical mirror + perspective camera [4]. **P** refers to the 3D scene point. **F** and **F'** refer to the two foci of the mirror and **p** refers to the image point. **F** is the effective viewpoint

reflected light ray is nothing but the incoming light ray from a scene point **P** in forward projection. The incoming ray passes through the focus **F** if extended on the inside of the mirror. This point where all the incoming light rays intersect (virtually) is called the effective viewpoint.

– *Elliptical mirror:* Figure 2 shows a central catadioptric system with an elliptical mirror and a perspective camera. The optical center of the perspective camera is placed at the upper focus of the elliptical mirror. By back-projecting an image point **p**, one can observe the following. The back-projected ray, starting from the optical center at the upper focus of the elliptical mirror, intersects and reflects from the surface of the elliptical mirror. The reflected back-projected ray, or the incoming light ray, virtually passes through the lower focus of the mirror. Thus, the lower focus (**F**) is the effective viewpoint of the system.

– *Hyperbolic mirror:* In Fig. 3, a catadioptric system is shown with a hyperbolic mirror and a perspective camera. The optical center of the perspective camera is placed at the external focus of the mirror **F'**. The back-projected ray of the image point **p** starts from the optical center, which is the external focus **F'** of the mirror, of the perspective camera. Using the same argument as above, one can observe that the lower focus **F** is the effective viewpoint. The first known work to use a hyperbolic mirror along with a perspective camera at the external focus of the mirror to obtain a single effective viewpoint configuration is [11]. Later in 1995, a similar implementation was proposed in [12].

Noncentral Catadioptric Cameras

Single viewpoint configurations are extremely delicate to construct, handle, and maintain. By relaxing

this single viewpoint constraint, one can obtain greater flexibility in designing novel systems. In fact most real catadioptric cameras are geometrically noncentral, and even the few restricted central catadioptric configurations are usually noncentral in practice [13]. For example, in the case of para-catadioptric cameras, the telecentric lens is never truly orthographic and it is difficult to precisely align the mirror axis and the axis of the camera. In hyperbolic or elliptic configurations, precise positioning of the optical center of the perspective camera in one of the focal points of the hyperbolic or elliptic mirror is practically infeasible. In [14], Ramalingam et al. show that most of the practically used catadioptric configurations fall under an axial camera model where all the projection rays pass through a single line rather than a single point in space. A few noncentral catadioptric configurations are mentioned below. Analogous to the single viewpoint in central cameras, there is a *viewpoint locus* in non-central cameras. It can be defined as follows: a curve or other set of points such that all projection rays cut at least one of the points in the viewpoint locus. Usually, one tries to find the “simplest” such set of points:

- *Conical mirror*: On using a conical mirror in front of a perspective camera, one can obtain an omnidirectional sensor [15, 16]. Nevertheless this configuration does not obey the single viewpoint restriction (besides in the degenerate case of the perspective optical center being located at the cone’s vertex). If the optical center lies on the mirror axis, then the viewpoint locus is a circle in 3D, centered in the mirror axis (it can be pictured as a halo over the mirror). An alternative choice of viewpoint locus is the mirror axis. Otherwise, the viewpoint locus is more general.
- *Spherical mirror*: On using a spherical mirror along with a perspective camera, one can enhance the field of view of the imaging system [16–18]. Again this configuration does not obey the single viewpoint restriction (besides in the degenerate case of the perspective optical center being located at the sphere center).
- *Digital micro-mirror array*: Another interesting camera is the recently introduced programmable imaging device using a digital micro-mirror array [19]. A perspective camera is made to observe a scene through a programmable array of micro-mirrors. By controlling the orientations and

positions of these mirrors, one can obtain an imaging system with complete control (both in terms of geometric and radiometric properties) over the incoming light ray for every pixel. However, there are several practical issues which make it difficult to realize the full potential of such an imaging system. First, current hardware constraints prohibit the usage of more than two possible orientations for each micro-mirror. Second, arbitrary orientations of the micro-mirrors would produce a discontinuous image which is unusable for many image processing operations.

- *Oblique cameras*: An ideal example for a noncentral camera is an oblique camera. No two rays intersect in an oblique camera [20]. In addition to developing multi-view geometry for oblique cameras, Pajdla also proposed a physically realizable system which obeys oblique geometry. The practical system consists of a rotating catadioptric camera that uses a conical mirror and a telecentric optics. The viewpoint locus is equivalent to a two-dimensional surface or a set of points, where each of the projection rays passes through at least one of the points. Different catadioptric configurations come with different calibration and 3D reconstruction algorithms. Recently, there has been a lot of interest in unifying different camera models and developing generic calibration and 3D reconstruction algorithms [21–24].

Application

Due to enhanced field of view, catadioptric cameras are mainly used in surveillance, car navigation, image-based localization, and augmented reality applications.

References

1. Descartes R, Smith D (1637) The geometry of René Descartes. Dover, New York. Originally published in Discours de la Méthode
2. Feynman R, Leighton R, Sands M (1963) The feynman lectures on physics. Mainly mechanics, radiation, and heat, Addison-Wesley, Reading, vol 1
3. Drucker D, Locke P (1996) A natural classification of curves and surfaces with reflection properties. Math Mag 69(4):249–256
4. Baker S, Nayar S (1998) A theory of catadioptric image formation. In: International conference on computer vision (ICCV), Bombay, pp 35–42

5. Bruckstein A, Richardson T (2000) Omnidirectional cameras with curved surface mirrors. In: IEEE workshop on omnidirectional vision, Hilton Head Island, pp 79–84
6. Sturm P, Ramalingam S, Tardif JP, Gasparini S, Barreto J (2011) Camera models and fundamental concepts used in geometric computer vision. Found Trends Comput Graph Vis 6(1–2):1–183
7. Ramalingam S (2006) Generic imaging models: calibration and 3d reconstruction algorithms. PhD thesis, INRIA Rhone Alpes
8. Nayar S (1997) Catadioptric omnidirectional camera. In: Conference on computer vision and pattern recognition (CVPR), San Juan, pp 482–488
9. Nalwa V (1996) A true omnidirectional viewer. Technical report, Bell Laboratories, Holmdel, NJ, USA
10. Lin S, Bajcky R (2001) True single view cone mirror omnidirectional catadioptric system. In: International conference on computer vision (ICCV), Vancouver, vol 2, pp 102–107
11. Rees DW (1970) Panoramic television viewing system. US Patent 3,505,465
12. Yamazawa K, Yagi Y, Yachida M (1995) Obstacle avoidance with omnidirectional image sensor hyperomni vision. In: International conference on robotics and automation, Nagoya, pp 1062–1067
13. Micusik B, Pajdla T (2004) Autocalibration and 3d reconstruction with non-central catadioptric cameras. In: Computer vision and pattern recognition (CVPR), San Diego, pp 748–753
14. Ramalingam S, Sturm P, Lodha S (2006) Theory and calibration algorithms for axial cameras. In: ACCV, Hyderabad
15. Yagi Y, Kawato S (1990) Panoramic scene analysis with conic projection. In: International conference on robots and systems (IROS), Cincinnati
16. Bogner S (1995) Introduction to panoramic imaging. In: IEEE SMC, New York, vol 54, pp 3100–3106
17. Hong J (1991) Image based homing. In: International conference on robotics and automation, Sacramento
18. Murphy JR (1995) Application of panoramic imaging to a teleoperated lunar rover. In: IEEE SMC conference, Vancouver, pp 3117–3121
19. Nayar S, Branzoi V, Boult T (2004) Programmable imaging using a digital micromirror array. In: International conference on computer vision and pattern recognition (CVPR), Washington, DC, pp 436–443
20. Pajdla T (2002) Stereo with oblique cameras. Int J Comput Vis 47(1):161–170
21. Geyer C, Daniilidis K (2000) A unifying theory of central panoramic systems and practical implications. In: European conference on computer vision (ECCV), Dublin, pp 159–179
22. Grossberg M, Nayar S (2001) A general imaging model and a method for finding its parameters. In: International conference on computer vision (ICCV), Vancouver, vol 2, pp 108–115
23. Sturm P, Ramalingam S (2004) A generic concept for camera calibration. In: European conference on computer vision (ECCV), Prague, vol 2, pp 1–13
24. Ramalingam S, Sturm P, Lodha S (2005) Towards complete generic camera calibration. In: Conference on computer vision and pattern recognition (CVPR), San Diego

Catoptrics

► [Catadioptric Camera](#)

Center of Projection

Srikumar Ramalingam
Mitsubishi Electric Research Laboratories,
Cambridge, MA, USA

Synonyms

[Optical center; Single viewpoint](#)

Related Concepts

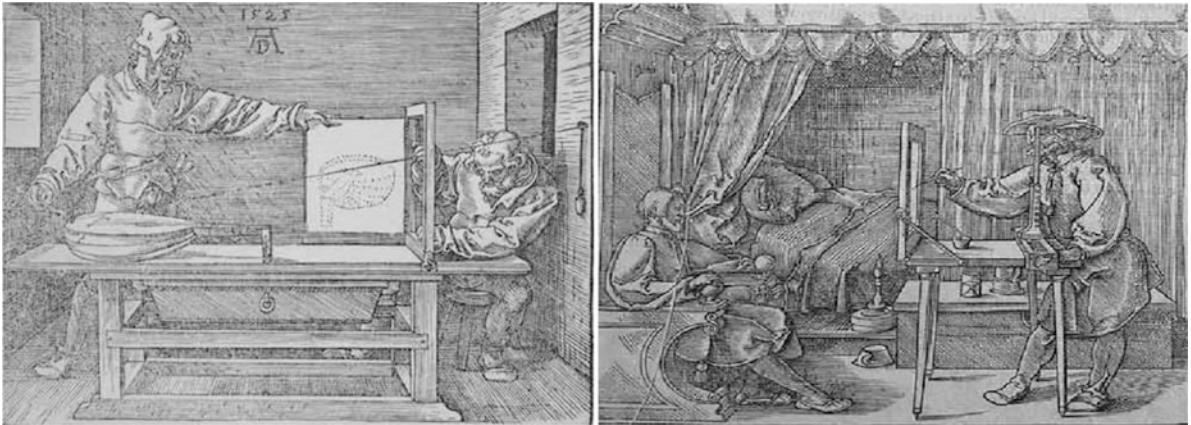
► [Field of View](#)

Definition

Center of projection is a single 3D point in space where all the light rays sampled by a conventional pinhole camera intersect.

Background

Albrecht Dürer, a German artist, published a treatise on measurement using a series of illustrations of drawing frames and perspective machines. On the left side of Fig. 1, an apparatus for drawing a lute is shown. One end of a thread is attached to a pointer and the other end to a pulley on a wall. The thread also passes through a frame in between the lute and the pulley. When the pointer is fixed at different points on the lute, the vertical and horizontal coordinates of the thread, as it passes through the frame, are marked. By meticulously marking the coordinates for each point on the lute, the perspective image of the lute is created. It is obvious to see the intuition behind this setup, i.e., its similarity to a pinhole camera. The pulley is equivalent to the single viewpoint or the center of projection, the frame replaces the image plane, and finally, the thread is nothing but the light ray emerging from the scene.



Center of Projection, Fig. 1 *Left:* One of Albrecht Dürer's perspective machines, which was used to draw a lute in the year 1525 [1]. *Right:* An artist uses the perspective principle

Though the principle is correct, the procedure is quite complicated. On the right side of Fig. 1, another perspective machine is shown. One can observe an artist squinting through a peep hole with one eye to keep a single viewpoint and tracing his sitter's features onto a glass panel. The idea is to trace the important features first and then transfer the drawing for further painting.

Theory

Pinhole Camera Model

Consider the perspective model that is shown in Fig. 2. Every 3D scene point $\mathbf{P}(X, Y, Z)$ gets projected onto the image plane to a point $\mathbf{p}(x, y)$ through the optical center \mathbf{C} . The optical axis is the perpendicular line to the image plane passing through the optical center. The center of radial symmetry in the image or principal point, i.e., the point of intersection of the optical axis and the image plane, is given by \mathbf{O} . The distance between \mathbf{C} (optical center) and the image plane is the focal length f . The optical center of the camera is the origin of the coordinate system. The image plane is parallel to the XY plane, held at a distance of f from the origin. Using the basic laws of trigonometry, one can observe the following:

$$x = \frac{fX}{Z}, \quad y = \frac{fY}{Z}$$

to accurately draw a man sitting on a chair, by looking at him through a peep hole with one eye, and tracing his features on a glass plate

Once expressed in homogeneous coordinates, the above relations transform to the following:

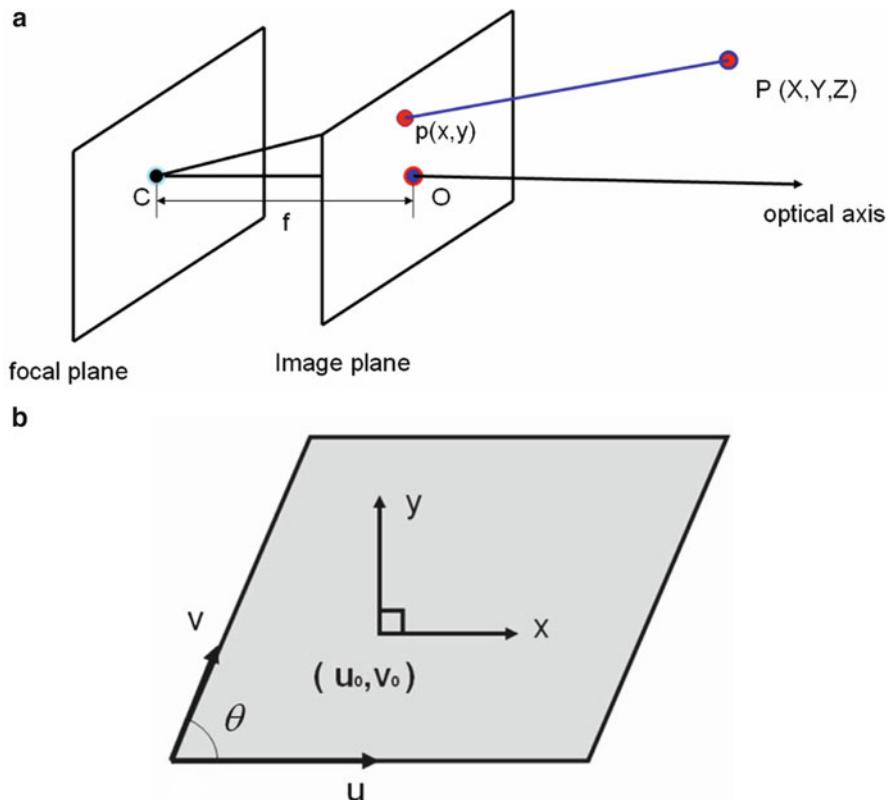
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

where the relationship \sim stands for “equal up to a scale.”

Practically available CCD cameras deviate from the perspective model. First, the principal point (u_0, v_0) does not necessarily lie on the geometrical center of the image. Second, the horizontal and vertical axes (u and v) of the image are not always perfect perpendicular. Let the angle between the two axes be θ . Finally, each pixel is not a perfect square and consequently, f_u and f_v are the two focal lengths that are measured in terms of the unit lengths along u and v directions. By incorporating these deviations in the camera model, one can obtain the following scene (X, Y, Z) to image (u, v) transformation:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} f_x & f_v \cot \theta & u_0 & 0 \\ 0 & \frac{f_v}{\sin \theta} & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

In practice, the 3D point is available in some world coordinate system that is different from the camera



Center of Projection, Fig. 2 (a) Perspective camera model. (b) The relationship between (u, v) and (x, y) is shown

coordinate system. The motion between these coordinate systems is given by (R, t) :

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} f_x & f_v \cot \theta & u_0 \\ 0 & \frac{f_v}{\sin \theta} & v_0 \\ 0 & 0 & 1 \end{bmatrix} [R \quad -Rt] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (1)$$

$$M = \begin{bmatrix} f_x & f_v \cot \theta & u_0 \\ 0 & \frac{f_v}{\sin \theta} & v_0 \\ 0 & 0 & 1 \end{bmatrix} [R \quad -Rt]$$

$$K = \begin{bmatrix} f_x & f_v \cot \theta & u_0 \\ 0 & \frac{f_v}{\sin \theta} & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

The 3×4 matrix M that projects a 3D scene point \mathbf{P} to the corresponding image point \mathbf{p} is called the projection matrix. The 3×3 matrix K that contains the internal parameters $(u_0, v_0, \theta, f_x, f_y)$ is generally referred to as the *intrinsic* matrix of a camera.

In back projection, given an image point \mathbf{p} , the goal is to find the set of 3D points that project to it. The back projection of an image point is a ray in space. One can compute this ray by identifying two points on this ray. The first point can be the optical center \mathbf{C} , since it lies on this ray. Since $MC = \mathbf{0}$, \mathbf{C} is nothing but the right nullspace of M . Second, the point M^+p , where M^+ is the pseudoinverse of M , lies on the back-projected ray because it projects to point \mathbf{p} on the image. Thus, the back projection of \mathbf{p} can be computed as follows:

$$\mathbf{P}(\lambda) = M^+ \mathbf{p} + \lambda \mathbf{C}$$

The parameter λ allows to get different points on the back-projected ray.

Caustics

In a single viewpoint imaging system, the geometry of the projection rays is given by the effective viewpoint and the direction of the projection rays. In a noncentral imaging system, *caustic*, a well-known terminology in

the optics community, can be utilized for representing the geometry of projection rays [2]. A caustic refers to the loci of viewpoints in 3D space to represent a noncentral imaging system. Concretely, the envelope of all incoming light rays that are eventually imaged is defined as the caustic. A caustic is referred to as dia-caustic for dioptric (lens-based systems) and catacaustic (mirror-based systems) for catadioptric systems. A complete study of conic catadioptric systems has been done [3]. Once the caustic is determined, each point on the caustic represents a light ray by providing its position and the direction. Position is given by the point on the caustic, and orientation is related to the concept of tangent. [Figure 3](#) shows the caustic for several noncentral imaging systems. For a single viewpoint imaging system, the caustic is a degenerate one being a single point. Simple methods exist for the computation of the caustic from the incoming light rays such as local conic approximations [4] and the so-called Jacobian method [5]. A few examples for caustics are shown in [Fig. 3](#).

Generalized and Multi-perspective Imaging Models

Many novel camera models have multiple centers of projection and they cannot be explained by a simple parametric pinhole model. In computer vision, there has been significant interest in generalizing the camera models to reuse the existing calibration and 3D reconstruction algorithms for novel cameras. In order to do this, first renounce on parametric models and adopt the following very general model: a camera acquires images consisting of pixels; each pixel captures light that travels along a ray in 3D. The camera is fully described by:

- The coordinates of these rays (given in some local coordinate system).
- The mapping between pixels and rays; this is basically a simple indexing.

The generic imaging model is shown in [Fig. 4](#). This allows to describe all above models and virtually any camera that captures light rays traveling along straight lines. The above imaging model has already been used, in more or less explicit form, in various works [3, 6–16], and is best described in [6]. There are conceptual links to other works: acquiring an image with a camera of the general model may be seen as sampling the plenoptic function [17], and a light field [18] or

lumigraph [19] may be interpreted as a single image, acquired by a camera of an appropriate design. More details of generic imaging model, their calibration, and 3D reconstruction algorithms can also be found in [20].

Taxonomy of Generic Imaging Models

Central Model: All the projection rays go through a single point, the optical center. Examples are mentioned below:

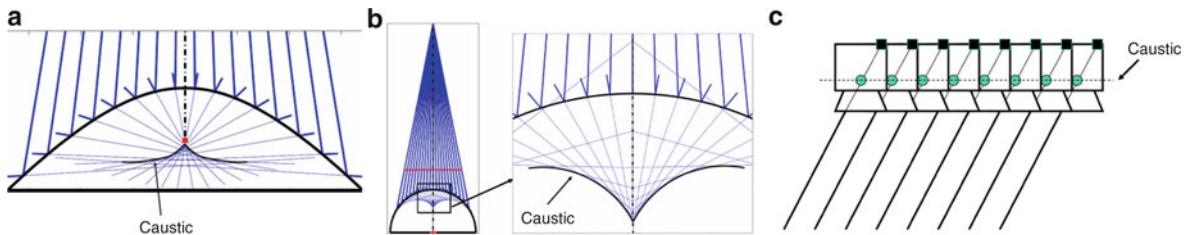
- The conventional perspective camera forms the classical example for a central camera.
- Perspective+radial or decentering distortion.
- Central catadioptric configurations using parabolic, hyperbolic, or elliptical mirrors.
- Fish-eye cameras can be considered as approximate central cameras.

Axial Model [21]: All the projection rays go through a single line in space, the *camera axis*. Examples of cameras falling into this class are:

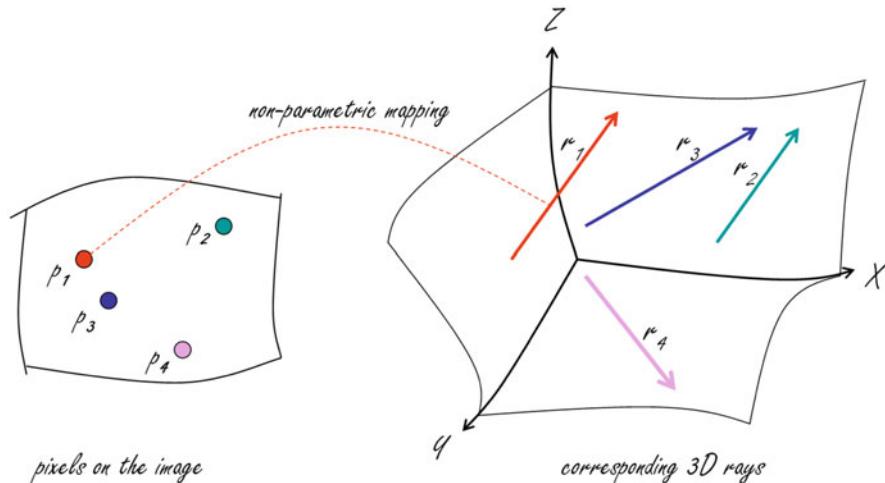
- Stereo systems consisting of 2, 3, or more central cameras with collinear optical centers.
- Noncentral catadioptric cameras of the following type: the mirror is any surface of revolution and the optical center of the central camera looking at it (can be any central camera, not only perspective) lies on its axis of revolution. It is easy to verify that in this case, all the projection rays cut the mirror's axis of revolution, i.e., the camera is an axial camera, with the mirror's axis of revolution as camera axis. Note that catadioptric cameras with a spherical mirror and a central camera looking at it are always axial ones.
- X-slit cameras [22] (also called two-slit or crossed-slit cameras), and their special case of linear push-broom cameras [23].

Noncentral Cameras: A noncentral camera may have completely arbitrary projection rays. Common examples are given below:

- Multi-camera system consisting of 3 or more cameras, all of whose optical centers are not collinear.
- Oblique camera: This is an ideal example for a non-central camera. No two rays intersect in an oblique camera [8].
- Imaging system using a micro-mirror array [24]. A perspective camera is made to observe a scene through a programmable array of micro-mirrors.



Center of Projection, Fig. 3 Caustics for several imaging systems (a) Hyperbolic catadioptric system (b) Spherical catadioptric system (c) Pushbroom camera



Center of Projection, Fig. 4 The main idea behind the generic imaging model: The relation between the image pixels (p_1, p_2, p_3, p_n) and their corresponding projection rays (r_1, r_2, r_3, r_n) is *non-parametric*

By controlling the orientations and positions of these mirrors, one can obtain an imaging system with complete control (both in terms of geometric and radiometric properties) over the incoming light ray for every pixel.

- Noncentral mosaic: An image sequence is captured by moving the optical center of a perspective camera in a circular fashion [3]. The center columns of the captured images are concatenated to create a noncentral mosaic image.
- Center strip mosaic: The optical center of the camera is moved [3]. The center columns of the captured images are concatenated to form a center strip mosaic. The resulting mosaic corresponds to a noncentral camera.

These three classes of camera models may also be defined as existence of a linear space of d dimensions that has an intersection with all the projection rays: $d = 0$ defines central, $d = 1$ axial, and $d = 2$ general noncentral cameras.

A detailed survey of various camera models, calibration, and 3D reconstruction algorithms is given in [25].

References

1. Dürer A (1525) Underweysung der Messung (Instruction in measurement). Book with more than 150 woodcuts
2. Born M, Wolf E (1965) Principles of optics. Permagon, Oxford
3. Swaminathan R, Grossberg M, Nayar S (2003) A perspective on distortions. In: Conference on computer vision and pattern recognition (CVPR), Madison, vol 2, p 594
4. Bruce JW, Giblin PJ, Gibson CG (1981) On caustics of plane curves. Am Math Mon 88:651–667
5. Burkhard D, Shealy D (1973) Flux density for ray propagation in geometrical optics. J Opt Soc Am 63(2):299–304
6. Grossberg M, Nayar S (2001) A general imaging model and a method for finding its parameters. In: International conference on computer vision (ICCV), Vancouver, vol 2, pp 108–115

7. Neumann J, Fermüller C, Aloimonos Y (2003) Polydoptic camera design and 3d motion estimation. In: Conference on computer vision and pattern recognition (CVPR), Madison, vol 2, pp 294–301
8. Pajdla T (2002) Stereo with oblique cameras. Int J Comput Vis 47(1):161–170
9. Peleg S, Ben-Ezra M, Pritch Y (2001) Omnistereo: panoramic stereo imaging. IEEE Trans Pattern Anal Mach Intell 23:279–290
10. Pless R (2003) Using many cameras as one. In: Conference on computer vision and pattern recognition (CVPR), Madison, pp 587–594
11. Seitz S (2001) The space of all stereo images. In: International conference on computer vision (ICCV), Vancouver, vol 1, pp 26–33
12. Shum H, Kalai A, Seitz S (1999) Omnivergent stereo. In: International conference on computer vision (ICCV), Corfu, pp 22–29
13. Sturm P, Ramalingam S (2004) A generic concept for camera calibration. In: European conference on computer vision (ECCV), Prague, vol 2, pp 1–13
14. Ramalingam S, Sturm P, Lodha S (2005) Towards complete generic camera calibration. In: Conference on computer vision and pattern recognition (CVPR), San Diego
15. Wexler Y, Fitzgibbon A, Zisserman A (2003) Learning epipolar geometry from image sequences. In: Conference on computer vision and pattern recognition (CVPR), Madison, vol 2, pp 209–216
16. Wood D, Finkelstein A, Hughes J, Thayer C, Salesin D (1997) Multiperspective panoramas for cell animation. In: SIGGRAPH, Los Angeles, pp 243–250
17. Adelson E, Bergen J (1991) The plenoptic function and the elements of early vision. In: Computational models of visual processing, edited by Michael S. Landy and J. Anthony Movshon, MIT Press, Cambridge, MA, pp 3–20
18. Levoy M, Hanrahan P (1996) Light field rendering. In: SIGGRAPH, New Orleans, pp 31–42
19. Gortler S, Grzeszczuk R, Szeliski R, Cohen M (1996) The Lumigraph. In: SIGGRAPH, New Orleans, pp 43–54
20. Ramalingam S (2006) Generic imaging models: calibration and 3d reconstruction algorithms. PhD Thesis, INRIA Rhone Alpes
21. Ramalingam S, Sturm P, Lodha S (2006) Theory and calibration algorithms for axial cameras. In: ACCV, Hyderabad
22. Feldman D, Pajdla T, Weinshall D (2003) On the epipolar geometry of the crossed-slits projection. In: International conference on computer vision (ICCV), Nice
23. Gupta R, Hartley R (1997) Linear pushbroom cameras. In: IEEE transactions on pattern analysis and machine intelligence (PAMI) 19(9):963–975
24. Nayar S, Branzoi V, Boult T (2004) Programmable imaging using a digital micromirror array. In: International conference on computer vision and pattern recognition (CVPR), Washington, pp 436–443
25. Sturm P, Ramalingam S, Tardif JP, Gasparini S, Barreto J (2011) Camera models and fundamental concepts used in geometric computer vision. Found Trends Comp Gr Vis 6(1–2):1–183

Change Detection

Joseph L. Mundy

Division of Engineering, Brown University
Rensselaer Polytechnic Institute, Providence,
RI, USA

Synonyms

Normalcy modeling

Definition

A determination that there are significant differences between visual scenes

Background

Change detection is a key task for computer vision algorithms. The goal is to compare two or more visual scenes and report any significant differences between the scenes. As with many vision tasks, the meaning of *significant* is application dependent. The change detection task can be rendered somewhat more concrete by considering the types of changes that are not typically of interest. Examples of changes that are usually irrelevant are:

- Camera viewpoint
- Varying illumination
- Wind-based motion
- Weather, e.g., snow and rain

The implementation of algorithms that can detect interesting changes while ignoring trivial changes such as these is a very difficult problem, and only quite limited change detection capabilities have achieved to date. It is also the case that the change detection task, when viewed broadly, overlaps the scope of many other vision tasks such as visual inspection and moving object detection.

Early attempts at change detection were based on simple strategies such as thresholding the magnitude of intensity differences between a reference image and an image that manifests change. This simple approach is only practical if the scene and imaging conditions are



closely controlled so that the only scene changes are due to events of interest. Such highly controlled conditions can be found in industrial applications where the lighting and camera pose is accurately maintained. For example, missing components in a circuit board can be detected by comparing the image intensity of a high-quality *master* board with images of boards with potentially missing components or other flaws. In this application, images are accurately registered to the master image, so that intensity differences correspond to physical differences in the boards. The detected changes correspond to missing components or additional material such as excess solder.

A more advanced change detection principle is to classify elements of a scene into categories of interest. Then two scenes can be compared as to the presence or absence of various category instances. Change detection based on this principle can be successful if the classification process is insensitive to the types of irrelevant scene variations mentioned above. This classification approach has been applied extensively to the detection of changes in multispectral aerial and satellite imagery [1]. Multispectral images typically have four or more color bands, so that each pixel is a feature vector that can be subjected to standard classifiers. Image regions are classified into types such as roads, vegetation, forest, and water. Images can then be compared to determine the change in area of each category. This approach requires pixel-level image-to-image registration, which in turn is only effective if the viewpoint change between the images is small and the spatial resolution of the images is low.

Given the complexity of scene appearance and contextual variations, the great majority of change detection algorithms have adopted a learning approach based on the observation of a number of images. In this learning paradigm, it is assumed that changes are rare so that a statistical model for *normal* image appearance can be formed without removing regions of the image that represent change.

A good example of this principle is learning the normal appearance of a road surface in a set of images that depict moving vehicles on the road. In any given image, some of the road surface is visible and contributes correct image appearance information to the model. In other images different parts of the surface may be visible. The assumption is that over a large set of training images, the frequency of vehicle appearance

at any given road surface point will be small compared to the frequency of the road appearance. This approach can also be denoted as *background modeling* because a model is being constructed for the scene background rather than the moving or changing objects, i.e., *foreground*. The approach can also be considered as developing a model for *normalcy* where changes are infrequent, i.e., abnormal.

This statistical modeling principle has received widespread acceptance with the advent of cheap computing power and the significance of video data, which provides large training samples. A classical reference is the work of Stauffer and Grimson, who implemented a moving object detection system in online video streams [2]. An important aspect of change detection in video is that illumination and atmospheric properties vary relatively slowly compared to the video frame rate. Under this condition, statistical appearance models can gradually adapt to such variations and not manifest them as change.

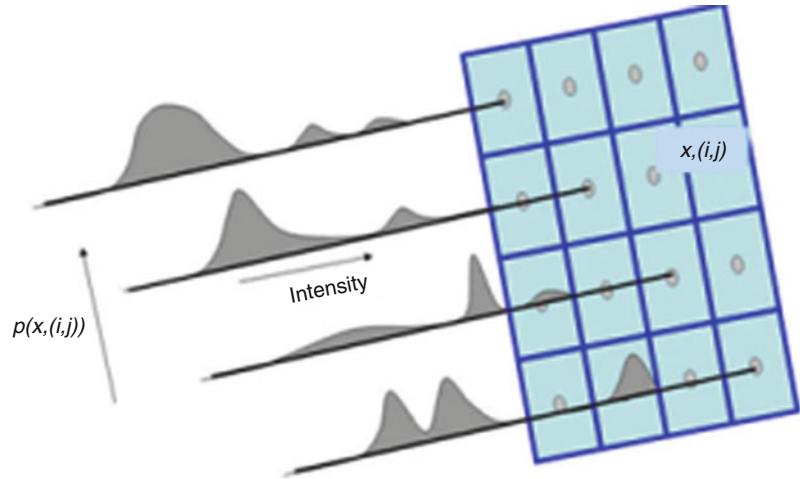
The effects of viewpoint and illumination can be overcome through the use of active 3-d range sensors such as LIDAR (Light Detection And Ranging) or laser triangulation. Scene illumination is provided by the sensor itself and therefore has known direction and spatial extent. The result is a 3-d point cloud of samples from scene surfaces and associated surface reflectance values. In this approach, change is detected by measuring the distance between two 3-d scene point sets after they have been accurately registered together. Missing points or points that are significantly far from the reference point set are considered to be change. Such change points can be grouped into a connected region to further characterize the change. An example of this strategy is the work of Girardeau-Montaut et al. [3].

Theory

The applications of change detection are so broad that a complete theoretical background is beyond the scope of an encyclopedia entry. Moreover, the success of change detection is critically dependent on accurate registration of the reference and current scenes so that change is due solely to the actual scene differences. However, the registration of scene images or 3-d data is a topic in its own right and will not be considered

Change Detection, Fig. 1

The appearance of each pixel is modeled by a Gaussian mixture distribution



here. Instead, two important statistical methods will be described that have achieved considerable success in image-based change detection: the joint probability method by Carlotto [4] and the background modeling method by Stauffer and Grimson [2].

Joint Probability Method

This approach can accommodate scenes with significant differences in illumination, and even between images taken with different sensor modalities such as visible and IR wave lengths. The method typically is applied to two images, $x_1(i, j)$ and $x_2(i, j)$, where the pixel intensity values, x , are considered to be random variables. It is assumed that the images are registered and the joint histogram, $p(x_1, x_2)$, of the intensity values at each pixel is accumulated. The expected intensity of a pixel in image 2, given the image value in image 1, is defined by

$$\tilde{x}_2(x_1) = \int x_2 p(x_2|x_1) dx_2, \quad (1)$$

where

$$p(x_2|x_1) = \frac{p(x_1, x_2)}{p(x_1)}.$$

Change is then defined as values of

$$c_{21}(i, j) = \|x_2(i, j) - \tilde{x}_2(x_1(i, j))\| \quad (2)$$

that exceed a decision threshold. The same analysis can be applied in reverse to compute

$$c_{12}(i, j) = \|x_1(i, j) - \tilde{x}_1(x_2(i, j))\|. \quad (3)$$

Gaussian Mixture Method

A second approach to modeling normal scene appearance is to associate a probability distribution with each image pixel location. This distribution accounts for normal variations in image intensity for each pixel across a set of images. It is assumed that the images are spatially registered so that a pixel in each image corresponds to the same surface element in the scene. An example of such registration is provided by a fixed video camera viewing a dynamic scene with moving objects and a stationary background.

The intensity of a given pixel will sometimes be due to background and sometimes foreground moving objects. Thus, an appropriate probability distribution for the overall appearance variation is a Gaussian mixture as shown in Fig. 1. The mixture distribution is defined by

$$p(x) = \sum_i \frac{w_i}{\sqrt{2\pi}\sigma_i} e^{-\frac{1}{2}\left(\frac{(x-\mu_i)}{\sigma_i}\right)^2}. \quad (4)$$

The distribution parameters, $\{w_i, \mu_i, \sigma_i\}$, are learned using a continuous online update algorithm where a new intensity sample, $x_{N+1}(i, j)$, is associated with an existing mixture component if it is within a few standard deviations of the component mean or a new



mixture is initiated. The update procedure for a scalar pixel intensity sample after N observations is defined by the following equations:

$$\begin{aligned} w_{N+1} &= \frac{Nw_N}{N+1} + \frac{1}{N+1} \\ \mu_{N+1} &= \frac{N\mu_N}{N+1} + \frac{x_N}{N+1} \\ \sigma_{N+1}^2 &= \frac{N\sigma_N^2}{N+1} + \frac{(x_N - \mu_N)^2}{N+1}. \end{aligned} \quad (5)$$

The update is applied if the sample, x_{N+1} , is within a specified number of standard deviations from one of the mixture component means. If a new sample is not within the capture range of one of the mixture components, then a new mixture component is added with user-specified default weight, w_i , and standard

deviation. When the limit on the number of mixture components is reached, the component with the smallest value of $\frac{w_i}{\sigma_i}$ is discarded. This component is the least informative (greatest entropy) of the mixture.

The update can be applied continuously over a video sequence, and the mixture parameters will adapt to the normal appearance of the scene. Change is detected as pixel intensities having low probability density according to the mixture distribution. It is the case that such change objects will introduce new modes into the mixture. However, these modes will typically have low weight compared to appearance mixture components corresponding to the stable scene background.

The probability of change can be computed as a Bayesian posterior based on the scene background mixture. That is,

$$P(\text{change}|x) = \frac{P(\text{change})}{P(\text{change}) + p(x|\text{background})(1 - P(\text{change}))}. \quad (6)$$

Here x is the observed intensity and $p(x|\text{background})$ is the Gaussian mixture learned from the sequence of images. The probability $P(\text{change})$ is the prior belief that the pixel exhibits change, and $P(\text{change}|x)$ is the posterior belief after observing the current image intensity value. This result is based on the assumption that image intensities are normalized to the range $[0, 1]$ and the probability density for foreground (change) intensities is uniform, i.e., $p(x|\text{change}) = 1$. Change is detected for pixels having $P(\text{change}|x)$ greater than a threshold.

that may or may not be of interest remains a significant challenge.

Experimental Results

An example of detected changes using the joint probability method is shown in Fig. 2. Note that significantly different appearance of the orchard on the right and different overall image contrast does not result in change since the differences are accumulated into the joint probability distribution. The method does not completely account for specular reflections of some of the building roofs (e.g., center and lower left) since these occur relatively infrequently. However, the cars in the roadway at the center are detected with reasonable accuracy.

An example of change detection using the Gaussian mixture model is shown in Fig. 3. In this example a sequence of aerial video frames are registered to a common ground plane and provide updates to the Gaussian mixture model at each pixel. Some false change probability can be seen at occluding boundaries and for metal building roofs that cause large variation in image intensity as the viewpoint changes. The actual changes in the scene are the moving vehicles on the roadways, which exhibit highly salient change probability values.

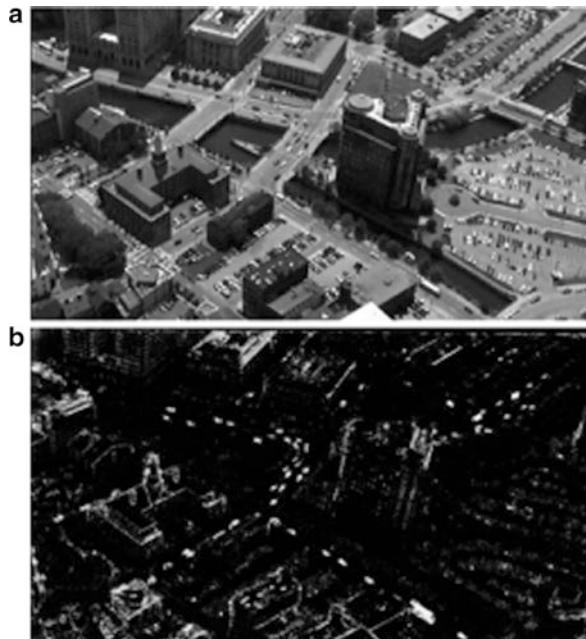
Open Problems

In spite of considerable research on change detection, the current state-of-the-art algorithms only perform well in scenes where appearance is highly consistent and viewpoint variations can be eliminated by using a fixed camera or by direct comparison of 3-d data. One possible way forward is the work of Pollard [5] where Gaussian appearance models are maintained in a full 3-d volumetric grid. This representation can account for variable viewpoint but is still limited to fixed or slowly varying illumination. Even if the viewpoint and illumination issues are overcome, accounting for the vast diversity of types of change



Change Detection, Fig. 2 Two satellite images taken at different times. (a) May 2006. (b) November 2006. (c) The resulting change map from the joint probability algorithm.

Bright intensities indicate high values of $c(i, j)$. The result is by Pollard [5] (Images copyright Digital Globe)



Change Detection, Fig. 3 An example of change probability computed using a Gaussian background mixture model. (a) A typical video frame. (b) The value of $P(\text{change}|x)$ displayed with white = 1

References

1. Congalton R, Green K (2009) Assessing the accuracy of remotely sensed data: principles and practices, 2nd edn. CRC Press, Boca Raton, FL
2. Stauffer C, Grimson W (1999) Adaptive background mixture models for real-time tracking. In: Proceedings of the international conference on computer vision and pattern recognition (CVPR), Fort Collins, Colorado, New York, vol 2, pp 246–252
3. Girardeau-Montau D, Roux M, Marc R, Thibault G (2005) Change detection on points cloud data acquired with a ground laser scanner. Remote sensing and spatial information sciences 36 (part3/W19), Enschede, The Netherland, pp 30–35
4. Carlotto MJ (2005) Detection and analysis of change in remotely sensed imagery with application to wide area

surveillance. IEEE trans on image process, New York, 2(3), pp 189–202

5. Pollard T (2009) Comprehensive 3-d change detection using volumetric appearance modeling. Brown University, Providence, RI

Chromaticity

Michael H. Brill
Datacolor, Lawrenceville, NJ, USA

Related Concepts

- [Trichromatic Theory](#)

Definition

Chromaticity is a representation of the tristimulus values of a light (see entry on ► [Trichromatic Theory](#)) by only two numbers, computed so as to suppress via ratios the absolute intensity of the light. The two numbers (called chromaticity coordinates) define a space (called chromaticity space) in which any additive mixture of two lights lies on a straight line between those two lights.

Theory

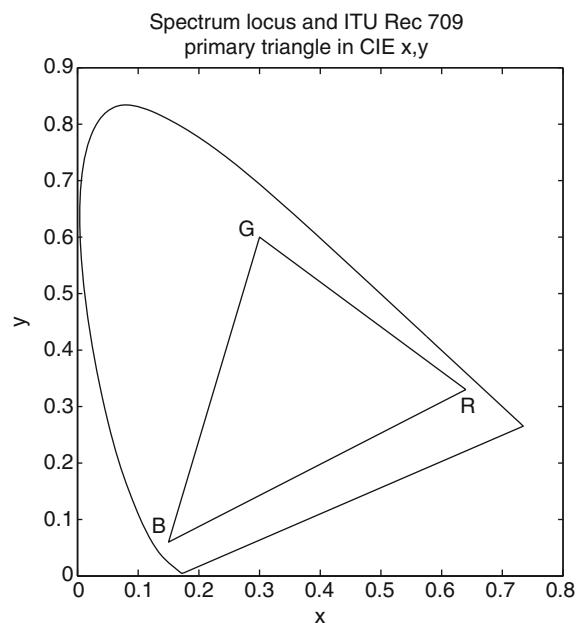
Most often, chromaticity is derived from a standardized tristimulus coordinate system representing human vision (e.g., a CIE XYZ system [1, 2]). If a light has tristimulus values X, Y, and Z, its chromaticity coordinates are conventionally defined as the ratios

$x = X/(X + Y + Z)$ and $y = Y/(X + Y + Z)$. More generally, the chromaticity coordinates could be defined as the ratios $q_1 = (aX + bY + cZ)/(pX + qY + rZ)$ and $q_2 = (dX + eY + fZ)/(pX + qY + rZ)$, where $a, b, c, d, e, f, p, q, r$ are constants that do not depend on the light. The above mapping from (X, Y, Z) to (q_1, q_2) is called a homogeneous central projection. It has the property of showing straight lines in (X, Y, Z) as straight lines in (q_1, q_2) . Hence it is obvious from looking at the chromaticity coordinates of three lights whether one of the lights could have been additively mixed from the other two. This property is useful for visualizing the color gamut (set of producible colors) of a self-luminous trichromatic display such as a cathode-ray tube (CRT) or a liquid-crystal display (LCD). The primaries (say, red, green, and blue) for such displays define three points in chromaticity space, and the triangle they generate spans the chromaticities producible by the display.

To illustrate the geometry of chromaticity, the figure below shows the 1931 CIE (x, y) diagram (Fig. 1). The horseshoe-shaped curve (from 380 nm at the left-hand end to 720 nm on the right-hand end) is the set of all monochromatic lights within the space, and is called the spectrum locus. The straight line connecting the ends of the curve is the line of purples. (There are no monochromatic purples.) Finally, the vertices of the triangle inside the spectrum locus represent the R, G, and B display primaries prescribed by ITU Recommendation 709. This set of primaries is only one example of many prescribed by standards bodies [3], and of even more sets manifested in real displays.

Various useful constructions are possible in chromaticity space. For example, the dominant wavelength of a light is defined by the point on the spectrum locus intercepted by a ray from the agreed-upon white chromaticity through the chromaticity of the light in question. Another example is the set of lights defined by the blackbody radiators (parameterized by their temperature). These lights collectively define the blackbody locus, which of course lies within the spectrum locus. The set of conventional daylights also forms a curve that is close to the blackbody locus. As another example, it is possible to transform from one tristimulus primary set to another such set using only information in the chromaticity domain [4].

Because a digital camera is a trichromatic device, one can also think of a camera's response to light as having a chromaticity. Such chromaticities are useful because they suppress spatial variations of light



Chromaticity, Fig. 1 CIE (x, y) chromaticity diagram with spectrum locus and RGB display primaries

intensity in an image (e.g., at shadow boundaries) and thereby facilitate image segmentation by object color (see entry on Band Ratios). Of course, band-ratio pairs are more general than chromaticities because the ratios can be separately defined without the mixture-on-a-straight-line constraint required for a chromaticity.

It should be noted that camera chromaticity is not at all the same as the human-vision chromaticity and must not be confused with it. In fact, two lights that have the same human-vision chromaticity can have different camera chromaticities, and vice versa. This effect is related to metamerism as noted in the entry on Trichromatic Theory. It should also be noted that, for the same reason, camera-derived values I, H, S used by computer-vision and image-processing applications are also not transformable to human perceptual attributes (e.g., intensity, hue, and saturation).

For both cameras and humans, chromaticity generalizes to a non-trichromatic system (i.e., one that has a number of different sensor types that is different from three). A sensor system with N sensor types delivers N -stimulus values for each light, the chromaticity space has $N-1$ dimensions (so as to suppress the light intensity), and the chromaticity coordinates comprise a homogeneous central projection out of the space of N -stimulus values.

References

1. Wyszecki G, Stiles WS (1982) Color science: concepts and methods, quantitative data and formulae, Chaps. 3 and 5, 2nd edn. Wiley, New York
2. Fairman HS, Brill MH, Hemmendinger H (1997) How the CIE 1931 color-matching functions were derived from Wright-Guild data. *Color Res Appl* 22:11–23
3. Poynton CA (1996) A technical introduction to digital video, Chap. 7. Wiley, New York
4. Brill MH (2008) Transformation of primaries using four chromaticity points and their matches. *Color Res Appl* 33:506–508
5. Fairchild M (2005) Color appearance models, Chap. 3, 2nd edn. Wiley, Chichester
6. MacAdam DL (1985) Color measurement: theme and variations. Springer, Berlin/New York
7. Koenderink JJ (2010) Color for the sciences, Chap. 4. MIT, Cambridge
8. Berns RS (2000) Billmeyer and Saltzman's principles of color technology, 3rd edn. Wiley, New York
9. Hunt RWG (1998) The measurement of colour, 3rd edn. Fountain Press, Kingston-Upon-Thames
10. Hunt RWG (2004) The reproduction of colour, 6th edn. Wiley, Chichester
11. Wandell BA (1995) Foundations of vision. Sinauer, Sunderland

CIE Standard Illuminant

- [Standard Illuminants](#)

Clifford Algebra

- [Geometric Algebra](#)

Clipping

- [Saturation \(Imaging\)](#)

Cluster Sampling

- [Swendsen-Wang Algorithm](#)

Coefficient Rule

- [von Kries Hypothesis](#)

Cognitive Agent

- [Cognitive System](#)

Cognitive System

David Vernon

Informatics Research Centre, University of Skövde,
Skövde, Sweden

Synonyms

- [Cognitive agent](#)

Related Concepts

- [Cognitive Vision](#)

Definition

A cognitive system is an autonomous system that can perceive its environment, learn from experience, anticipate the outcome of events, act to pursue goals, and adapt to changing circumstances.

Background

There are several scientific perspectives on the nature of cognition and on how it should be modeled. All fall under the general umbrella of cognitive science which embraces the disciplines of neuroscience, artificial intelligence, cognitive psychology, linguistics, and epistemology. Among these differing perspectives, however, there are two broad classes: the *cognitivist* approach based on symbolic information processing representational systems, and the *emergent systems* approach, encompassing connectionist systems, dynamical systems, and enactive systems, all based to a lesser or greater extent on principles of self-organization [1–4]. A third class – *hybrid systems* – attempts to combine something from each of the cognitivist and emergent paradigms. All three approaches have their origins in cybernetics [5] which in the decade from 1943 to 1953 made the first efforts to



formalize what had up to that point been purely psychological and philosophical treatments of cognition. The intention of the early cyberneticians was to create a science of mind, based on logic. Examples of the application of cybernetics to cognition include the seminal paper by McCulloch and Pitts “A logical calculus immanent in nervous activity” [6] and Ashby’s “Design for a Brain” [7].

Theory

The initial attempt in cybernetics to create a science of cognition was followed by the development of an approach referred to as *cognitivism*. The birth of the cognitivist paradigm, and its sister discipline of Artificial Intelligence, dates from a conference held at Dartmouth College, New Hampshire, in July and August 1956 and attended by people such as John McCarthy, Marvin Minsky, Allen Newell, Herbert Simon, and Claude Shannon. Cognitivism holds that cognition is achieved by computation performed on internal symbolic knowledge representations in a process whereby information about the world is abstracted by perception, and represented using some appropriate symbolic data-structure, reasoned about, and then used to plan and act in the world. The approach has also been labeled by many as the information processing or symbol manipulation approach to cognition [1, 8–10]. In most cognitivist approaches concerned with the creation of artificial cognitive systems, the symbolic representations are the descriptive product of a human designer. This is significant because it means that they can be directly accessed and interpreted by humans and that semantic knowledge can be embedded directly into and extracted directly from the system. In cognitivism, the goal of cognition is to reason symbolically about these representations in order to effect the required adaptive, anticipatory, goal-directed behavior. Typically, this approach to cognition will deploy machine learning and probabilistic modeling in an attempt to deal with the inherently uncertain, time-varying, and incomplete nature of the sensory data that is used to drive this representational framework. Significantly, in the cognitivist paradigm, the instantiation of the computational model of cognition is inconsequential: any physical platform that supports the performance of the required symbolic computations will suffice [8]. This principled separation

of operation from instantiation is referred to as functionalism.

In the emergent paradigm, cognition is the process whereby an autonomous system becomes viable and effective in its environment. It does so through a process of self-organization by which the system continually maintains its operational identity through the moderation of mutual system-environment interaction. In other words, the ultimate goal of an emergent cognitive system is to maintain its own autonomy. In achieving this, the cognitive process determines what is real and meaningful for the system: the system constructs its reality – its world and the meaning of its perceptions and actions – as a result of its operation in that world. Consequently, the system’s understanding of its world is inherently specific to the form of the system’s embodiment and is dependent on the system’s history of interactions, i.e., its experiences. This mutual-specification of the system’s reality by the system and its environment is referred to as co-determination [11] and is related to the concept of radical constructivism [12]. This process of making sense of its environmental interactions is one of the foundations of the enactive approach to cognition [13]. Cognition is also the means by which the system compensates for the immediate nature of perception, allowing it to anticipate environmental interaction that occurs over longer time scales, i.e., cognition is intrinsically linked with the ability of an agent to act prospectively: to deal with what might be, not just with what is. Many emergent approaches adhere to the principle that the primary model for cognitive learning is anticipative skill construction rather than knowledge acquisition. Thus, processes which guide action and improve the capacity to guide action form the root capacity of all intelligent systems [14].

As noted already, the emergent paradigm embraces connectionist systems, dynamical systems, and enactive systems. Connectionist systems rely on parallel processing of non-symbolic distributed activation patterns using statistical properties, rather than logical rules, to process information and achieve effective behavior [15]. In this sense, the neural network instantiations of the connectionist model are dynamical systems that capture the statistical regularities in training data [16]. Dynamical systems theory has been used to complement classical approaches in artificial intelligence [17] and it has also been deployed to model natural and artificial cognitive systems [10, 18, 19]. Although dynamical systems theory

approaches often differ from connectionist systems on several fronts, it is better perhaps to consider them complementary ways of describing cognitive systems, dynamical systems addressing macroscopic behavior at an emergent level, and connectionist systems addressing microscopic behavior at a mechanistic level [20]. Enactive systems take the emergent paradigm even further. Enaction [13, 21–23] asserts that cognition is a process whereby the issues that are important for the continued existence of a cognitive entity are brought out or enacted: co-determined by the entity and the environment in which it is embedded. Thus, enaction entails that a cognitive system operates autonomously, that it generates its own models of how the world works, and that the purpose of these models is to preserve the system's autonomy.

Considerable effort has gone into developing hybrid approaches which combine aspects of cognitivist and emergent systems. Typically, hybrid systems exploit symbolic knowledge to represent the agent's world and logical rule-based systems to reason about this knowledge in order to achieve goals and select actions, while at the same time using emergent models of perception and action to explore the world and construct this knowledge. Thus, hybrid systems still use cognitivist representations and representational invariances but they are constructed by the system itself as it interacts with and explores the world rather than through a priori specification or programming. Consequently, as with emergent systems, the agent's ability to understand the external world is dependent on its ability to interact flexibly with it, and interaction is the organizing mechanism that establishes the association between perception and action.

Cognitivism and artificial intelligence research are strongly related. In particular, Newell and Simon's "Physical Symbol System" approach to artificial intelligence [8] has been extremely influential in shaping how we think about intelligence, natural as well as computational. In their 1976 paper, two hypotheses are presented: the *Physical Symbol System Hypothesis* and the *Heuristic Search Hypothesis*. The first hypothesis is that a physical symbol system has the necessary and sufficient means for general intelligent action. This implies that any system that exhibits general intelligence is a physical symbol system and any physical symbol system of sufficient size can be configured to exhibit general intelligence. The second hypothesis states that the solutions to problems are represented as

symbol structures and that a physical-symbol system exercises its intelligence in problem-solving by search, i.e., by generating and progressively modifying symbol structures in an effective and efficient manner until it produces a solution structure. This amounts to an assertion that symbol systems solve problems by heuristic search, i.e., the successive generation of potential solution structures. The task of intelligence, then, is to avert the ever-present threat of the exponential explosion of search. Subsequently, Newell defined intelligence as the degree to which a system approximates the ideal of a knowledge-level system [24]. A knowledge-level system is one which can bring to bear *all* its knowledge onto *every* problem it attempts to solve (or, equivalently, every goal it attempts to achieve). Perfect intelligence implies complete utilization of knowledge. It brings this knowledge to bear according to the *principle of maximum rationality* which was proposed by Newell in 1982 [25] as follows: "If an agent has knowledge that one of its actions will lead to one of its goals, then the agent will select that action." Anderson [26] later offered a slightly different principle, the *principle of rationality*, sometimes referred to as rational analysis, stated as follows: "the cognitive system optimizes the adaptation of the behavior of the organism." Note that Anderson's principle considers optimality to be necessary for rationality, something that Newell's principle does not.

Cognitivist and emergent approaches are normally contrasted on the basis of the symbolic or non-symbolic nature of their computational operation and representational framework. Cognitivist systems typically use production systems to effect rule-based manipulation of symbol tokens whereas emergent systems exploit dynamical processes of self-organization in which representations are encoded in global system states. However, the distinction between cognitivist and emergent is not restricted to the issue of symbolic representation and they can be contrasted on the basis of several other characteristics such as semantic grounding, temporal constraints, inter-agent epistemology, embodiment, perception, action, anticipation, adaptation, motivation, autonomy, among others [27].

The differences between the cognitivist and the emergent paradigm can be traced to their underlying distinct philosophies [28]. Broadly speaking, cognitivism is dualist, functionalist, and positivist. It is dualist in the sense that there is a fundamental distinction between the mind (the computational processes) and



the body (the computational infrastructure and, if required, the physical structure that instantiates any physical interaction). It is functionalist in the sense that the actual instantiation and computational infrastructure is inconsequential: any instantiation that supports the symbolic processing is sufficient. It is positivist in the sense that they assert a unique and absolute empirically-accessible external reality that is apprehended by the senses and reasoned about by the cognitive processes. In contrast, emergent systems are neither dualist nor functionalist, since the system's embodiment is an intrinsic component of the cognitive process, nor positivist, since the form and meaning of the system's world is dependent in part on the system itself. The emergent paradigm, and especially the enactive approach, can trace its roots to the philosophy of phenomenology [28, 29].

A criticism often leveled at cognitivist systems is that they are relatively poor at functioning effectively outside well-defined problem domains because they tend to depend on in-built assumptions and embedded knowledge arising from design decisions. Emergent systems should in theory be much less brittle because they develop through mutual specification and co-determination with the environment. However, the ability to build artificial cognitive systems based on emergent principles is very limited at present, and cognitivist and hybrid systems currently have more advanced capabilities within a narrower application domain.

Any cognitive system is inevitably going to be complex. Nonetheless, it is also the case that it will exhibit some degree of structure. This structure is often encapsulated in what is known as a cognitive architecture [30]. Although used freely by proponents of the cognitivist, emergent, and hybrid approaches to cognitive systems, the term “cognitive architecture” originated with the seminal cognitivist work of Newell et al. [25]. Consequently, the term has a very specific meaning in this paradigm where cognitive architectures represent attempts to create unified theories of cognition [24, 31], i.e., theories that cover a broad range of cognitive issues, such as attention, memory, problem-solving, decision-making, learning, from several aspects including psychology, neuroscience, and computer science. In the cognitivist paradigm, the focus of a cognitive architecture is on the aspects of cognition that are constant over time and that are independent of the task. Since cognitive

architectures represent the fixed part of cognition, they cannot accomplish anything in their own right and need to be provided with or acquire knowledge to perform any given task. For emergent approaches to cognition, which focus on development from a primitive state to a fully cognitive state over the lifetime of the system, the architecture of the system is equivalent to its phylogenetic configuration: the initial state from which it subsequently develops through ontogenesis.

Open Problems

The study of cognitive systems is a maturing discipline with contrasting approaches. Consequently, there are several open problems. These include the role of physical embodiment, the need for development, the system's cognitive architecture, the degree of autonomy required, the issue of symbol grounding, the problem of goal specification, the ability to explain the rationale for selection actions, the problem of generating generalized concepts and transferring knowledge from one context to another, and the interdependence of perception and action. The nature of any resolution of these problems is inextricably linked to the choice of paradigm: cognitivist, emergent, or hybrid.

The role of physical embodiment in a cognitive system [32–34] depends strongly on the chosen paradigm. Due to their functionalist characteristics, cognitivist systems do not depend on physical embodiment to operate successfully but there is nothing to prevent them from being embodied if that is what the task in hand requires. Emergent systems, by definition, require embodiment since the body plays a key role in the way a cognitive system comes to understand – make sense of – its environment. If a body is required, the form of embodiment must still be specified [35]. This is significant because, in the emergent paradigm at least, the ability of two cognitive agents to communicate effectively requires them to have similar embodiments so that they have a shared history of interaction and a common epistemology.

The extent to which a cognitive system requires a capacity for development and, if so, the mechanisms by which development can take place are both open problems. In natural systems, growth is normally associated with development. However, growth in artificial systems remains a distant goal, although one whose

achievement would open up many avenues of fruitful enquiry in cognitive systems. For current state-of-the-art cognitive systems, one can define development as the process by which a system discovers for itself the models that characterize its interactions with its environment. This contrasts with learning as the process whereby the parameters of an existing model are estimated or improved. Development, then, requires a capacity for self-modification [36] and in embodied emergent systems leads to an increased repertoire of effective actions and a greater ability to anticipate the need for and outcome of future actions [27].

The capacity to develop introduces another open issue: the minimal phylogenetic configuration – the perceptual, cognitive, and motoric capabilities with which a system is endowed at “birth” – that is required to facilitate subsequent ontogenesis – development and learning through exploration and social interaction [27]. This issue is related to the specification of the system’s cognitive architecture and the necessary and sufficient conditions that must be satisfied for cognitive behavior to occur in a system. In addressing these issues, there is a trade-off between the initial phylogeny and the potential for subsequent development. This trade-off is reflected by the existence of two types of species in nature: precocial and altricial. Precocial species are those that are born with well-developed behaviors, skills, and abilities which are the direct result of their genetic make-up (i.e., their phylogenetic configuration). As a result, precocial species tend to be quite independent at birth. Altricial species, on the other hand, are born with poor or undeveloped behaviors and skills, and are highly dependent for support. However, in contrast to precocial species, they proceed to learn complex cognitive skills over their lifetime (i.e., through ontogenetic development). The precocial and the altricial effectively define a spectrum of possible configurations of phylogenetic configuration and ontogenetic potential [37]. The problem is to identify a feasible point in this spectrum that will yield a cognitive system capable of developing the skills we require of it.

Autonomy is a crucial issue for cognitive systems [38] but the degree of autonomy required is unclear. To an extent, it depends on how autonomy is defined and which paradigm of cognition is being considered. Definitions range from self-regulation and homeostasis to the ability of a system to contribute to its own persistence [39]. In the former case, self-regulation is

often cast as a form of self-control so that the systems can operate without interference from some outside agent, such as a human user. In the latter case, autonomy is the self-maintaining organizational feature of living creatures that enables them to use their own capacities to manage their interactions with the world in order to remain viable [14]. Cognitivist systems tend to adopt the former definition, emergent systems, the latter.

Broadly speaking, cognitivist systems exploit symbolic representations while emergent systems exploit sub-symbolic state-based representations, with hybrid systems using both. The manner in which cognitivist and hybrids systems ground their symbolic representations in experience is still an open issue [40], with some arguing for a bottom-up approach [41] and others for a process of learned association, where meaning is attached rather than grounded [37].

The opening definition of a cognitive system states that it can act to achieve goals. The specification of these goals poses a significant challenge due to the autonomous nature of cognitive systems. It is more easily resolved for cognitivist systems since the goals can be hard-wired into the cognitive architecture. It is less clear how goals can be specified in an emergent system since the over-arching goal here is the maintenance of the system’s autonomy. The goals of such a system reflect its intrinsic motivations and its associated value system [42]. The problem is to understand how to engineer this value system to ensure that the system is motivated to act in a way that satisfies goals which are external to the system and to decide how these goals can be communicated to the system.

Ideally, in addition to the characteristics of a cognitive system listed in the opening definition – autonomy, perception, learning, anticipation, goal-directed action, and adaptation – a cognitive computer system should also be able to say what it is doing and why it is doing it, i.e., it should be able to explain the reasons for an action [43]. This would enable the system to identify potential problems which might appear when carrying out a task and it would know when it needed new information in order to complete that task. Consequently, a cognitive system would be able to view a problem in several different ways and to look at different alternative ways of tackling it. In a sense, this is something similar to the issue discussed above about cognition involving an ability to anticipate the need for actions and their outcome. The difference in



this case is that the cognitive system is considering not just one but many possible sets of needs and outcomes. In a sense, it is adapting *before* things do not go according to plan. From this point of view, cognition also involves a sense of self-reflection.

Cognitive systems also learn from experience and adapt to changing circumstances. To do this, the system must have some capacity for generalization so that concepts can be formed from specific instances and so that knowledge and know-how can be transferred from one context to another. This capacity would allow the system to adapt to new application scenarios and to explore the hypothetical situations that arise from the self-reflection mentioned above. It is unclear at present how such generalized conceptual knowledge and know-how should be generated, represented, and incorporated into the system dynamics.

Perception and action have been demonstrated to be co-dependent in biological systems. Perceptual development depends on what actions an infant is capable of and what use objects and events afford in the light of these capabilities. This idea of the action-dependent perceptual interpretation of an object is referred to as its affordance [44]. In neuroscience, the tight relationship between action and perception is exemplified by the presence of mirror neurons, neurons that become active when an action is performed and when the action or a similar action is observed being performed by another agent. It is significant that these neurons are specific to the goal of the action and not the mechanics of carrying it out. The related Ideomotor Theory [45] asserts the existence of such a common or co-joint representational framework for perception and action. Such a framework would facilitate the inference of intention and the anticipation of an outcome of an event due to the goal-oriented nature of the action. The realization of an effective co-joint perception-action framework remains an important challenge for cognitivist and emergent approaches alike.

Although clearly there are some fundamental differences between the cognitivist and the emergent paradigms, the gap between the two shows some signs of narrowing. This is mainly due to (1) a recent movement on the part of proponents of the cognitivist paradigm to assert the fundamentally important role played by action and perception in the realization of a cognitive system [32]; (2) the move away from the view that internal symbolic representations are the only valid form of representation [2]; and (3)

the weakening of the dependence on embedded a priori knowledge and the attendant-increased reliance on machine learning and statistical frameworks both for tuning system parameters and the acquisition of new knowledge. This suggests that hybrid approaches may be the way forward, especially if a principled synthesis of cognitivist and emergent approaches is possible, such as “dynamic computationalism” [2] or “computational mechanics” [46]. Hybrid approaches appear to many to offer the best of both worlds – the adaptability of emergent systems and the advanced starting point of cognitivist systems – since the representational invariances and representational frameworks need not be learned but can be designed in and since the system populates these representational frameworks through learning and experience. However, it is uncertain that one can successfully combine what are ultimately highly incompatible underlying philosophies. Opinion is divided, with arguments both for (e.g., [2, 40, 46]) and against (e.g., [47]).

References

1. Varela FJ (1992) Whence perceptual meaning? A cartography of current ideas. In: Varela FJ, Dupuy JP (eds) Understanding origins – contemporary views on the origin of life, mind and society. Boston studies in the philosophy of science. Kluwer Academic Publishers, Dordrecht, pp 235–263
2. Clark A (2001) Mindware – an introduction to the philosophy of cognitive science. Oxford University Press, New York
3. Freeman WJ, Núñez R (1999) Restoring to cognition the forgotten primacy of action, intention and emotion. *J Conscious Stud* 6(11–12):ix–xix
4. Winograd T, Flores F (1986) Understanding computers and cognition – a new foundation for design. Addison-Wesley Publishing Company, Inc., Reading
5. Ross Ashby W (1957) An introduction to cybernetics. Chapman and Hall, London
6. McCulloch WS, Pitts W (1943) A logical calculus of ideas immanent in nervous activity. *Bull Math Biophys* 5:115–133
7. Ross Ashby W (1954) Design for a brain. Chapman and Hall, London
8. Newell A, Simon HA (1976) Computer science as empirical inquiry: symbols and search. *Commun Assoc Comput Mach* 19:113–126. Tenth Turing award lecture, ACM, 1975
9. Haugland J (ed) (1997) Mind design II: philosophy, psychology, artificial intelligence. MIT, Cambridge, MA
10. Kelso JAS (1995) Dynamic patterns – the self-organization of brain and behaviour, 3rd edn. MIT, Cambridge, MA
11. Maturana H, Varela F (1987) The tree of knowledge – the biological roots of human understanding. New Science Library, Boston & London

12. von Glaserfeld E (1995) Radical constructivism. RoutledgeFalmer, London
13. Stewart J, Gapenne O, Di Paolo EA (2011) Enaction: toward a new paradigm for cognitive science. MIT, Cambridge, MA
14. Christensen WD, Hooker CA (2000) An interactivist-constructivist approach to intelligence: self-directed anticipative learning. *Philos Psychol* 13(1):5–45
15. Medler DA (1998) A brief history of connectionism. *Neural Comput Surv* 1:61–101
16. Smolensky P (1996) Computational, dynamical, and statistical perspectives on the processing and learning problems in neural network theory. In: Smolensky P, Mozer MC, Rumelhart DE (eds) Mathematical perspectives on neural networks. Erlbaum, Mahwah, NJ, pp 1–15
17. Reiter R (2001) Knowledge in action: logical foundations for specifying and implementing dynamical systems. MIT, Cambridge, MA
18. Thelen E, Smith LB (1994) A dynamic systems approach to the development of cognition and action. MIT press/Bradford books series in cognitive psychology. MIT, Cambridge, MA
19. Port RF, van Gelder T (1995) Mind as motion – explorations in the dynamics of cognition. Bradford Books, MIT, Cambridge, MA
20. McClelland JL, Vallabha G (2006) Connectionist models of development: mechanistic dynamical models with emergent dynamical properties. In: Spencer JP, Thomas MSC, McClelland JL (eds) Toward a new grand theory of development? Connectionism and dynamic systems theory re-considered. Oxford University Press, New York
21. Maturana HR, Varela FJ (1980) Autopoiesis and cognition – the realization of the living. Boston studies on the philosophy of science. D. Reidel Publishing Company, Dordrecht
22. Varela F, Thompson E, Rosch E (1991) The embodied mind. MIT, Cambridge, MA
23. Vernon D (2010) Enaction as a conceptual framework for development in cognitive robotics. *Paladyn J Behav Robot* 1(2):89–98
24. Newell A (1990) Unified theories of cognition. Harvard University Press, Cambridge, MA
25. Newell A (1982) The knowledge level. *Artif Intell* 18(1): 87–127
26. Anderson J (1999) Cognitive architectures in rational analysis. In: van Lehn K (ed) Architectures for intelligence. Lawrence Erlbaum Associates, Hillsdale, pp 1–24
27. Vernon D, von Hofsten C, Fadiga L (2010) A roadmap for cognitive development in humanoid Robots. Volume 11 of cognitive systems monographs (COSMOS). Springer, Berlin
28. Vernon D, Furlong D (2007) Philosophical foundations of enactive AI. In: Lungarella M, Iida F, Bongard JC, Pfeifer R (eds) 50 years of AI. Volume LNAI 4850. Springer, Heidelberg, pp 53–62
29. Froese T, Ziemke T (2009) Enactive artificial intelligence: investigating the systemic organization of life and mind. *Artif Intell* 173:466–500
30. Langley P, Laird JE, Rogers S (2009) Cognitive architectures: research issues and challenges. *Cogn Syst Res* 10(2):141–160
31. Anderson JR, Bothell D, Byrne MD, Douglass S, Lebiere C, Qin Y (2004) An integrated theory of the mind. *Psychol Rev* 111(4):1036–1060
32. Anderson ML (2003) Embodied cognition: a field guide. *Artif Intell* 149(1):91–130
33. Steels L (2007) Fifty years of AI: from symbols to embodiment – and back. In: Lungarella M, Iida F, Bongard JC, Pfeifer R (eds) 50 years of AI. Volume LNAI 4850. Springer, Heidelberg, pp 18–28
34. Vernon D (2008) Cognitive vision: the case for embodied perception. *Image Vis Comput* 26(1):127–141
35. Ziemke T (2003) What's that thing called embodiment? In: Alterman R, Kirsh D (eds) Proceedings of the 25th annual conference of the cognitive science society. Lund university cognitive studies. Lawrence Erlbaum, Mahwah, pp 1134–1139
36. Weng J (2004) Developmental robotics: theory and experiments. *Int J Humanoid Robot* 1(2):199–236
37. Sloman A, Chappell J (2005) The altricial-preocial spectrum for robots. In: IJCAI '05 – 19th international joint conference on artificial intelligence, Edinburgh
38. Varela F (1979) Principles of biological autonomy. Elsevier North Holland, New York
39. Bickhard MH (2000) Autonomy, function, and representation. *Artif Intell Spec Issue Commun Cogn* 17(3–4): 111–131
40. Barsalou LW (2010) Grounded cognition: past, present, and future. *Top Cogn Sci* 2:716–724
41. Harnad S (1990) The symbol grounding problem. *Phys D* 42:335–346
42. Edelman GM (2006) Second nature: brain science and human knowledge. Yale University Press, New Haven and London
43. Brachman RJ (2002) Systems that know what they're doing. *IEEE Intell Syst* 17(6):67–71
44. Gibson JJ (1979) The ecological approach to visual perception. Houghton Mifflin, Boston
45. Stock A, Stock C (2004) A short history of ideo-motor action. *Psychol Res* 68(2–3):176–188
46. Crutchfield JP (1998) Dynamical embodiment of computation in cognitive processes. *Behav Brain Sci* 21(5):635–637
47. Christensen WD, Hooker CA (2004) Representation and the meaning of life. In: Clapin H, Staines P, Slezak P (eds) Representation in mind: new approaches to mental representation. Elsevier, Oxford, pp 41–70

Cognitive Vision

David Vernon

Informatics Research Centre, University of Skövde,
Skövde, Sweden

Related Concepts

►Cognitive System; ►Visual Cognition



Definition

Cognitive vision refers to computer vision systems that can pursue goals, adapt to unexpected changes of the visual environment, and anticipate the occurrence of objects or events.

Background

The field of cognitive vision grew from the broader area of computer vision in response to a need for vision systems that are more widely applicable, that are able adapt to novel scenes and tasks, that are robust to unexpected variations in operating conditions, and that are fast enough to deal with the timing requirements of these tasks [1]. Adaptability entails the ability to acquire knowledge about the application domain, thereby removing the need to embed all the required knowledge in the system when it is designed. Robustness allows the system to be tolerant to changes in environmental conditions so that system performance is not negatively impacted by them when carrying out a given task. Speed and the ability to pay attention to critical events are essential when providing feedback to users and devices in situations which change unexpectedly [2, 18]

While computer vision systems routinely address signal processing of sensory data and reconstruction of 3D scene geometry, cognitive vision goes beyond this by providing a capability for conceptual characterization of the scene structure and dynamics using qualitative representations. Having knowledge about the scene available in conceptual form allows the incorporation of consistency checks through the use of, e.g., logic inference engines. These checks can be applied both to the knowledge that is embedded in the system at the outset and the knowledge that the system learns for itself. Consistency checking applies across several scales of space and time, requiring cognitive vision to have an ability to operate with past, present, and future events. These consistency checks are one way in which the robustness associated with cognitive vision can be achieved [3]. Furthermore, the conceptual knowledge generated by cognitive vision can, if required, be communicated to a human user in natural language [4]. This linguistic communication is one manifestation of an autonomous system

demonstrating its understanding of the visual events in its environment [3].

Theory

Cognitive vision entails abilities to anticipate future events and to interpret a visual scene in the absence of complete information. To achieve this, a cognitive system must have the capacity to acquire new knowledge and to use it to fill in gaps that are present in what is being made immediately available by the visual sensors: to extrapolate in time and space to achieve a more robust and effective understanding of the underlying behavior of the sensed world. In the process, the system learns, anticipates, and adapts. These three characteristics of learning, anticipation, and adaptivity are the hallmarks of cognition, in general, and cognitive vision, in particular [6, 7].

A key property of cognitive vision is its capacity to exhibit a robust performance even in scenarios that were not anticipated when it was designed. The degree to which a system can deal with unexpected circumstances will vary. Systems that can adapt autonomously to arbitrary situations are unrealistic at present but it is plausible that they should be able to deal with new variants of visual form, function, and behavior, and also incremental changes in context. Ideally, a cognitive vision system should be able to recognize and adapt to novel variations in the current visual environment, generalize to new contexts and application domains, interpret and predict the behavior of agents detected in the system's environment, and communicate an understanding of the environment to other systems, including humans.

A cognitive vision system is a visually enabled cognitive system, defined in this encyclopedia as “an autonomous system that can perceive its environment, learn from experience, anticipate the outcome of events, act to pursue goals, and adapt to changing circumstances.” Since cognitive vision is principally a mode of perception, physical action – with the possible exception of the camera movements associated with active vision – usually falls outside its scope. However, speech acts may be involved when communicating an interpretation of the scene in conceptual terms through language [5]. Since cognitive vision is a particular type of cognitive system, all the issues identified in the cognitive system entry in this encyclopedia

apply equally to cognitive vision. They will not be revisited here apart from noting that there are several scientific perspectives on the nature of cognition and on how it should be modeled. Among these differing perspectives, there are two broad classes: the *cognitivist* approach based on symbolic information processing representational systems, and the *emergent systems* approach, encompassing connectionist systems, dynamical systems, and enactive systems, all based to a lesser or greater extent on principles of self-organization. A third class – *hybrid systems* – attempts to combine something from each of the cognitivist and emergent paradigms. The vast majority of cognitive vision systems adopt either a cognitivist or a hybrid approach, with matching cognitive architectures [8].

The term *visual cognition* is strikingly similar to the term *cognitive vision*. However, they are not equivalent. Visual cognition is a branch of cognitive psychology concerned with research on visual perception and cognition in humans [9, 10, 19]. It addresses several distinct areas such as object recognition, face recognition, scene understanding, visual attention (including visual search, change blindness, repetition blindness, and the control of eye movements), short-term and long-term visual memory, and visual imagery. It is also concerned with the representation and recognition of visual information currently being perceived by the senses and with reasoning about memorized visual imagery. Thus, visual cognition addresses many visual mechanisms that are relevant to cognitive vision but without necessarily treating the entire cognitive system or the realization of these mechanisms in artificial systems.

Application

Several applications of cognitive vision may be found in [11–13]. Examples include natural language description of traffic behavior [5], autonomous control of cars [14], and observation and interpretation of human activity [15, 16].

Open Problems

All of the open problems associated with cognitive systems apply equally to cognitive vision. Three which are particularly relevant are highlighted here.

The first concerns embodiment [17]. There is no universal agreement on whether or not a cognitive vision system must be embodied. Even if it is, several forms of embodiment are possible. One form is a physical robot capable of moving in space, manipulating the environment, and experiencing the physical forces associated with that manipulation. Other forms of embodiment do not involve physical contact and simply require the system to be able to change the state of its visual environment, for example, a surveillance system which can control ambient lighting. These alternative forms of embodiment are consistent with the cognitivist and hybrid paradigms of cognition but do not satisfy the requirements of the emergent approach.

Learning in cognitive vision presents several significant challenges. Since cognitive vision systems do not have all the knowledge required to carry out their tasks, they need to learn. More specifically, they need to be able to learn in an incremental, continuous, open-ended, and robust manner, with learning and recognition being interleaved, and with both improving over time. Since the learning process will normally be effected autonomously without supervision, the learning technique needs to be able to distinguish between good and bad data, otherwise bad data may corrupt the representation and cause errors to become embedded and to propagate. Furthermore, the use of learning in several domains is required, including perceptual (spatiotemporal) and conceptual (symbolic) domains, as well as in mapping between them. The mapping from perceptual to conceptual facilitates communication, categorization, and reasoning, whereas the mapping from conceptual to perceptual facilitates contextualization and embodied action. Learning may be interpreted in a restricted sense as the estimation of the parameter values that govern the behavior of models that have been designed into the system, or in a more general sense as the autonomous generation of mappings that represent completely new models.

The identification and achievement of goals in cognitive vision presents a further challenge. With cognitivist approaches, goals are specified explicitly by designers or users in terms of the required outcome of cognitive behavior. With emergent approaches, goals are more difficult to specify since cognitive behavior is an emergent consequence of the system dynamics. Consequently, they have to be specified in terms



of constraints or boundary conditions on the system configuration, either through phylogenetic configuration or ontogenetic development, or both. It is a significant challenge to understand how implicit goals can be specified and incorporated, and how externally communicated goals can be introduced to the system from its environment or from those interacting with it, e.g., through some form of conditioning, training, or communication.

References

1. Christensen HI, Nagel HH (2006) Introductory remarks. In: Christensen HI, Nagel HH (eds) Cognitive vision systems: sampling the spectrum of approaches. Volume 3948 of LNCS. Springer, Heidelberg, pp 1–4
2. Tsotsos JK (2006) Cognitive vision need attention to link sensing with recognition. In: Christensen HI, Nagel HH (eds) Cognitive vision systems: sampling the spectrum of approaches. Volume 3948 of LNCS. Springer, Heidelberg, pp 25–35
3. Nagel HH (2003) Reflections on cognitive vision systems. In: Crowley J, Piater J, Vincze M, Paletta L (eds) Proceedings of the third international conference on computer vision systems, ICVS 2003. Volume LNCS 2626. Springer, Berlin, Heidelberg, pp 34–43
4. Arens M, Ottlik A, Nagel HH (2002) Natural language texts for a cognitive vision system. In: Harmelen FV (ed) Proceedings of the 15th European conference on artificial intelligence (ECAI-2002). IOS Press, Amsterdam, pp 455–459
5. Nagel HH (2004) Steps toward a cognitive vision system. *AI Magazine* 25(2):31–50
6. Auer P et al (2005) A research roadmap of cognitive vision. ECVision: European network for research in cognitive vision systems. <http://www.ecvision.org>
7. Vernon D (2006) The space of cognitive vision. In: Christensen HI, Nagel HH (eds) Cognitive vision systems: sampling the spectrum of approaches. Volume 3948 of LNCS. Springer, Heidelberg, pp 7–24
8. Granlund G (2006) Organization of architectures for cognitive vision systems. In: Christensen HI, Nagel HH (eds) Cognitive vision systems: sampling the spectrum of approaches. Volume 3948 of LNCS. Springer, Heidelberg, pp 37–55
9. Pinker S (1984) Visual cognition: an introduction. *Cognition* 18:1–63
10. Coltheart V (ed) (2010) Tutorials in visual cognition. Macquarie monographs in cognitive science. Psychology Press, London
11. Neumann B (ed) (2005) Künstliche Intelligenz, special issue on cognitive computer vision, vol 2. Böttcher IT Verlag, Bremen
12. Christensen HI, Nagel HH (2006) Cognitive vision systems: sampling the spectrum of approaches. Volume 3948 of LNCS. Springer, Heidelberg
13. Buxton H (ed) (2008) Image vision comput, special issue on cognitive vision 26(1)
14. Dickmanns ED (2004) Dynamic vision-based intelligence. *AI Magazine* 25(2):10–29
15. Sage K, Howell J, Buxton H (2005) Recognition of action, activity and behaviour in the actIPret project. *Künstliche Intell Spec Issue Cogn Comput Vis* 19(2):30–34
16. Crowley JL (2006) Things that see: context-aware multimodal interaction. In: Christensen HI, Nagel HH (eds) Cognitive vision systems: sampling the spectrum of approaches. Volume 3948 of LNCS. Springer, Heidelberg, pp 183–198
17. Vernon D (2008) Cognitive vision: the case for embodied perception. *Image Vis Comput* 26(1):127–141
18. Tsotsos JK (2011) A computational perspective on visual attention. MIT Press, Cambridge
19. Cavanagh P (2011) Visual cognition. *Vision Res* 51(13): 1538–1551

Color Adaptation

► [von Kries Hypothesis](#)

Color Appearance Models

► [Color Spaces](#)

Color Balance

► [White Balance](#)

Color Constancy

Marc Ebner

Institut für Mathematik und Informatik, Ernst Moritz Arndt Universität Greifswald, Greifswald, Germany

Definition

Color constancy is the ability to perceive colors as approximately constant even though the light entering the eye varies with the illuminant. Color constancy also names the field of research investigating the extent of this ability, that is, the conditions under which a color is actually perceived as constant and which factors

influence color constancy. Computer scientists working in the field of color constancy try to mimic this ability in order to produce images which are independent of the illuminant, that is, color constant. Simple color-constancy algorithms, also known under the name automatic white balance, are used in digital cameras to compute a color-corrected output image. The input of a color constancy algorithm is often one image taken under an arbitrary illuminant and the output of a color constancy algorithm is frequently the image as it would appear had it been taken with a canonical illuminant such as CIE Standard Illuminant D65 or a spectrally uniform illuminant.

Background

An introduction to computational color constancy is given by Ebner [1]. Maloney [2] also reviews different algorithms for surface color perception. Color is a product of the brain [3]. When an observer perceives an object, processing starts with the retinal sensors. The sensors in the retina measure the light entering the eye. However, the light entering the eye is dependent on both the spectral characteristics of the illuminant and the reflectance properties of the object. Therefore, without any additional processing, the measured light varies with the illuminant.

In the eye, two different types of retinal sensors exist: rods and cones. The rods are used for viewing when little light is available. The cones are mostly used in bright light conditions for color vision. Three types of cones can be distinguished which absorb light primarily in the red, green, and blue parts of the spectrum. Similarly, a digital sensor often measures the incident light in three different parts of the spectrum and uses red, green, and blue sub-sensors. However, cameras with four sub-sensors, for example, red, green, blue, and cyan also exist.

Suppose that an observer views a diffusely reflecting surface which uniformly reflects the incident light. Now assume that the surface is illuminated by a candle. A candle emits more light toward the red part of the spectrum. The candle light will reach the surface where part of the light will be absorbed and the remainder will be reflected. Part of the reflected light enters the eye where it is measured. The sensitivity function of the sensors in combination with the amount of light entering the eye will determine how strongly the

sensors respond. Now consider another illuminant with a higher color temperature, for example, daylight or an electronic flash. Such an illuminant will emit more light toward the blue spectrum compared to the candle. If the same surface is viewed with an illuminant that has a high color temperature, then the sensors shift their response toward the blue part of the spectrum. Assuming a normalized response of all three sensors, then the reflected light will have a red color cast when the surface is illuminated by a candle. The measured light will appear white during daylight and it will have a bluish color cast for an illuminant with a high color temperature. The observer, however, will be able to call out the correct color of the surface independent of the illuminant. Color constancy algorithms try to mimic this ability by computing an image which is independent of the illuminant.

Theory

Let $I(\lambda, x, y)$ be the irradiance captured by either a digital sensor or by the eye at position (x, y) for wavelength λ . Let $S_i(\lambda)$ be the response function of sensor i . Then the response of the sensor $c_i(x, y)$ at position (x, y) is given by:

$$c_i(x, y) = \int S_i(\lambda) I(\lambda, x, y) d\lambda.$$

The integration is done over all wavelengths to which the sensor responds. Assuming three receptors with sensitivity in the red, green, and blue parts of the spectrum, then $i \in \{r, g, b\}$. In this case, the measurement of the sensor is a three component vector $\mathbf{c} = [c_r, c_g, c_b]$.

The irradiance I falling onto the sensor is a result of the light reflected from an object patch. Let $L(\lambda, x, y)$ be the irradiance falling onto a diffusely reflecting object patch which is imaged at position (x, y) of the sensor arrangement. Let $R(\lambda, x, y)$ be the reflectance of the imaged object patch. Thus,

$$I(\lambda, x, y) = G(x, y) R(\lambda, x, y) L(\lambda, x, y)$$

where $G(x, y)$ is a geometry factor which takes the orientation between the surface and the light source into account. For a diffusely reflecting surface, $G(x, y) = \cos \alpha$ where α is the angle between the unit



vector which points from the surface into the direction of the light source and the normal vector at the corresponding surface position. Thus, the sensor response can be written as:

$$c_i(x, y) = G(x, y) \int S_i(\lambda) R(\lambda, x, y) L(\lambda, x, y) d\lambda.$$

From this equation, it is apparent that the sensor response depends on the orientation of the patch relative to the light source (because of $G(x, y)$); it depends on the sensitivity S_i of the sensor i , the reflectance of the object patch $R(\lambda, x, y)$, and on the illuminant $L(\lambda, x, y)$. Some color constancy algorithms are based on a set of basis functions to model illuminants and reflectances. See Maloney [2] for an introduction.

Color constancy algorithms frequently assume that the sensitivity of the sensors is very narrow band. Assuming that they have the shape of a delta function, $S_i = \delta(\lambda - \lambda_i)$, it holds that:

$$\begin{aligned} c_i(x, y) &= G(x, y) \int \delta(\lambda - \lambda_i) R(\lambda, x, y) L(\lambda, x, y) d\lambda, \\ c_i(x, y) &= G(x, y) R(\lambda_i, x, y) L(\lambda_i, x, y). \end{aligned}$$

This equation is often written in the form:

$$c_i(x, y) = G(x, y) R_i(x, y) L_i(x, y)$$

where the only difference to the previous equation is that the index i is used instead of the parameter λ_i . In this treatment, sensor response, reflectance, and the illuminant is considered only for three distinct wavelengths, that is, color bands, with $i \in \{r, g, b\}$.

A color constancy algorithm tries to discount the illuminant by computing a color constant descriptor $d(\mathbf{c})$ which is independent of the illuminant $\mathbf{L}(x, y) = [L_r(x, y), L_g(x, y), L_b(x, y)]$. Geusebroek et al. [4] have derived several different descriptors which can be computed from \mathbf{c} and are invariant to some imaging conditions such as viewing direction, surface orientation, highlights, illumination direction, illumination intensity, or illumination color. Finlayson and Hordley [5] have shown how a color constant descriptor can be computed provided that the illuminant can be approximated by a black body radiator. Apart from computing a color constant descriptor, a color constancy algorithm usually tries to output an image of

the scene which would either correspond to the perception of a human photographer observing the scene or it would correspond to the image that would have resulted if a spectrally uniform illuminant or illuminant D65 had been used.

An ideal solution to the problem of color constancy would be to compute $\mathbf{R}(x, y) = [R_r(x, y), R_g(x, y), R_b(x, y)]$ from the sensor responses. It is of course clear that this problem cannot be solved without making additional assumptions, because for each position on the sensor array, one only has three measurements but there are seven unknowns (shading, reflectance, and illumination components). Note that the above model for image generation is already a simple model assuming narrow band sensor responses.

A frequently made assumption is that the illuminant varies slowly over the image while reflectance is able to change abruptly between sensor responses. Since color constancy algorithms are based on certain assumptions, they will not work correctly if the assumptions are violated. In many cases, it is possible to find images where the color constancy algorithm does not perform as intended. The goal is to develop algorithms which perform well on most everyday scenes.

Simple Algorithms

If a single illuminant illuminates the scene uniformly, that is, $L_i(x, y) = L_i$, then it suffices to estimate a three component vector $\tilde{\mathbf{L}}$ from all the measured sensor responses with $\tilde{\mathbf{L}} \approx \mathbf{L}$. Given an estimate $\tilde{\mathbf{L}}(x, y)$, a color constant descriptor can be computed by dividing the sensor response by the estimate of the illuminant.

$$\begin{aligned} \frac{c_i(x, y)}{\tilde{L}_i(x, y)} &= \frac{G(x, y) R_i(x, y) L_i(x, y)}{\tilde{L}_i(x, y)} \\ &\approx \frac{G(x, y) R_i(x, y) L_i(x, y)}{L_i(x, y)} \\ &= G(x, y) R_i(x, y) \end{aligned} \quad (1)$$

Such an output image will be a shaded reflectance image. In other words, a diagonal color transform suffices if the sensor response is very narrow band. Some color constancy algorithms, however, also use a general 3×3 matrix transform to compute a color-corrected output image.

It is also possible to transform a given image taken under one illuminant \mathbf{L} to another image taken under a different illuminant \mathbf{L}' . This can be done by multiplying each sensor response vector by a diagonal matrix whose elements are set to $k_i = \frac{L'_i}{L_i}$. The coefficients k_i are called von Kries coefficients [6]. Necessary and sufficient conditions on whether von Kries chromatic adaptation gives color constancy have been derived by West and Brill [7].

A simple algorithm to estimate the color of the illuminant is the white patch Retinex algorithm. It is a simplified version of the parallel Retinex algorithm [8]. In order to understand how this algorithm works, suppose that a white patch, that is, a uniformly reflecting patch, is contained in the imaged scene which is uniformly illuminated. Assuming a normalized sensor response, then the response of the sensors on the white patch will be an estimate of the illuminant. For the white patch, it holds that $R_i = 1$ which leads to $c_i(\text{white patch}) = GL_i$. The white patch algorithm treats each color band separately and searches for the maximum response which is assumed to be an estimate of the illuminant.

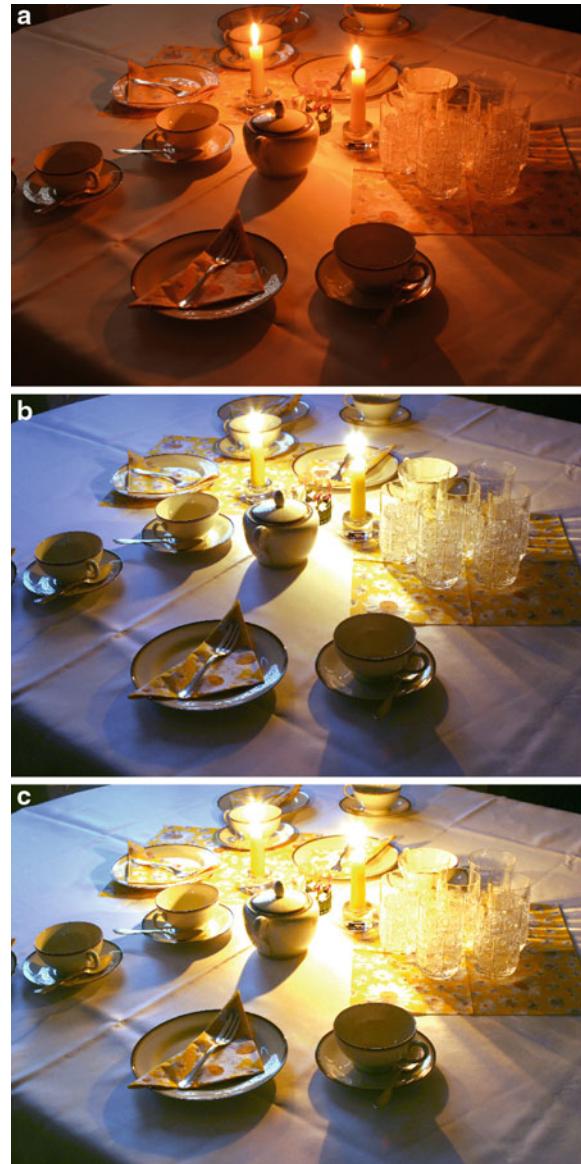
$$\tilde{L}_i = \max_{x,y} c_i(x, y)$$

Instead of locating the maximum response, one can also compute a histogram of the sensor responses and then set the estimate of the illuminant at some percentage from above. This will lead to a more robust estimate of the illuminant. Fig. 1b shows the output of the white patch Retinex algorithm for a sample image shown in Fig. 1a. The illuminant was estimated at 5% from above using a histogram approach for each color band.

Another simple algorithm is based on the gray world assumption which is due to Buchsbaum [9]. According to Buchsbaum, the world is gray on average. Let a_i be the global average of all sensor responses for color channel i where n is the number of sensors.

$$a_i = \frac{1}{n} \sum_{x,y} c_i(x, y) = \frac{1}{n} \sum_{x,y} G(x, y) R_i(x, y) L_i(x, y)$$

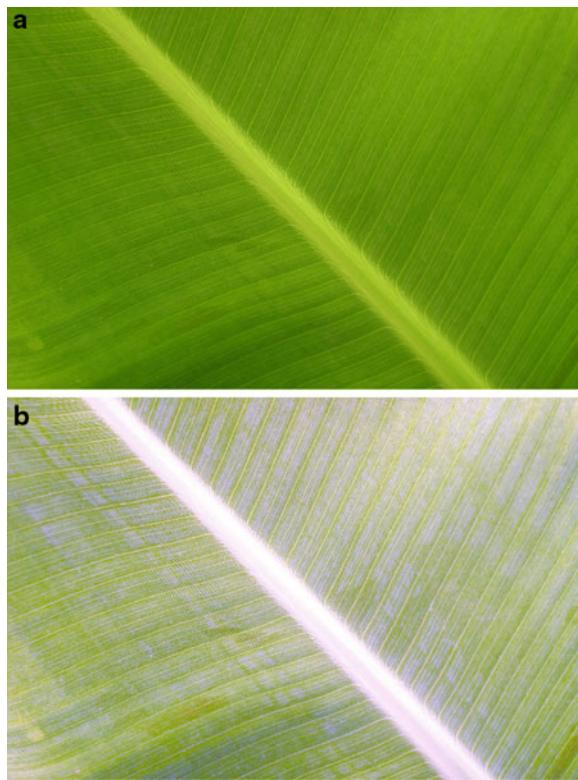
Assuming a uniform illuminant $L_i(x, y) = L_i$ and an independence between shading and reflectance, then:



Color Constancy, Fig. 1 (a) sample input image (b) results for the white patch Retinex algorithm using a histogram (c) results for the gray world assumption

$$\begin{aligned} a_i &= L_i \frac{1}{n} \sum_{x,y} G(x, y) R_i(x, y) \\ &= L_i E[G(x, y) R_i(x, y)] = L_i E[G(x, y)][R_i(x, y)] \end{aligned} \quad (2)$$

where $E[x]$ denotes the expected value of x . Suppose that a large number of differently colored objects are contained in the scene. Thus, a uniform distribution



Color Constancy, Fig. 2 (a) image of a leaf from a banana plant (b) results for the gray world assumption

of colors is assumed. This results in $E[R_i(x, y)] = \frac{1}{n} \sum_{x,y} R_i(x, y) = \frac{1}{2}$ assuming a range of $[0, 1]$ for reflectances and $E[G(x, y)] = c$ where c is a constant. The result is:

$$a_i = \frac{c}{2} L_i.$$

Hence, the illuminant is proportional to global space average color $L_i \propto a_i$ and an estimate of the illuminant can be obtained by setting:

$$\tilde{L}_i = 2a_i.$$

Using $c = 1$ assumes that all patches are frontally oriented or alternatively, that the geometry factor is subsumed into a combined reflectance and geometry factor. [Fig. 1c](#) shows the results for the gray world assumption on a sample image. [Fig. 2](#) shows the results for another image where the assumption, that a large number of different colored objects are contained in the scene, is not fulfilled. In this case, the gray world assumption will not work correctly.

Additional Color Constancy Algorithms

Other important algorithms include gamut constraint algorithms originally developed by Forsyth [10]. A gamut constraint algorithm looks at the gamut of colors contained in a sample image and then transforms this color gamut to a color gamut of a canonical image. Forsyth's algorithm operates on three color channels. Finlayson [11] has developed a variant which operates with a projected color gamut (2D gamut constraint algorithm). Van de Weijer et al. [12] have introduced the gray-edge hypothesis. While the gray-world assumption suggests that the world is gray on average, the gray-edge hypothesis suggests that image derivatives are gray on average. Brainard and Freeman [13] have addressed the problem of color constancy using Bayesian decision theory.

Uniform Versus Nonuniform Illumination

Many color constancy algorithms assume that the scene is uniformly illuminated by a single illuminant. If multiple light sources are distributed over the scene, then it is assumed that these light sources can be combined into a single average illuminant. This is possible provided that the light sources are sufficiently distant from the scene. If the illumination is uniform, then only a three component descriptor has to be estimated from the input image. However, in practice, many scenes are illuminated nonuniformly. Very often, one has several different illuminants. For instance, daylight may be falling through a window while artificial lights have already been turned on inside a building. Nonuniform illumination may also be present outside during a sunny day. Consider a family sitting in the garden under a red umbrella and a photographer taking a photograph of the family members. The family would be illuminated by light reflected from the red umbrella while the surrounding would be illuminated by direct sunlight. A digital camera usually corrects for a single illuminant. Thus, either the family members would have a red color cast to them or the background colors would not look right in the resulting image.

Algorithms have also been developed which can cope with a spatially varying illuminant. Land and McCann's Retinex algorithm [8] is a parallel algorithm for color constancy which allows for a nonuniform illumination. They only considered one-dimensional paths over the sensor array. Horn [14] extended Land's Retinex algorithm to two dimensions.

Blake [15] provided additional improvements. Barnard et al. [16] extended the 2D gamut constraint algorithm to scenes with varying illumination.

Ebner [1] showed how a grid of processing elements is able to estimate the color of the illuminant locally using the gray world assumption. Each processing element receives the measurement c_i from the sensor and computes local space average color. The processing elements are laterally connected to neighboring processing elements. Let $a_i(x, y)$ be the current estimate of local space average color for a processing element located at position (x, y) . Each processing element receives the estimate of local space average color from neighboring elements and averages the neighboring estimates to update its own estimate. A small component from the sensor measurement is then added to this estimate. Let p be a small percentage and let $N(x, y)$ be the neighborhood of the processing element at position (x, y) , then the computation of a processing element consists of the following two updates.

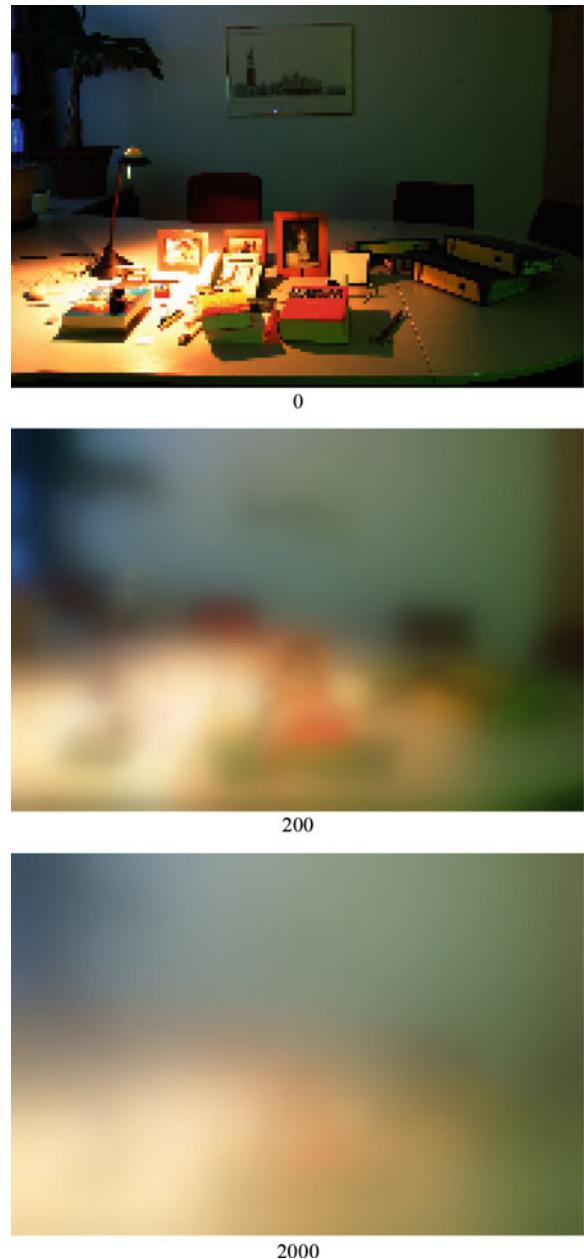
$$a_i'(x, y) = \frac{1}{|N(x, y)|} \sum_{(x', y') \in N(x, y)} a_i(x', y')$$

$$a_i(x, y) = (1 - p)a_i'(x, y) + pc_i(x, y)$$

The two updates are carried out iteratively. This process converges to local space average color which is an estimate of the illuminant, $\tilde{L}_i(x, y) = 2a_i(x, y)$. Figure 3 shows local space average color after 0, 200, and 2,000 iterations using $p = 0.0001$ given the input image shown in Fig. 4b. The extent of the averaging is determined by the parameter p . Figure 4 shows the output when local space average color is used to estimate the illuminant locally. Local average color was computed for a downsampled image (25% in each direction), the original image had 768×512 pixels. Ebner [1] suggested that a similar algorithm may be used by the brain for color perception.

Advanced Reflectance Models

The theoretical model of color image formation, that has been given above, assumed that objects diffusely reflect the incident light. This is not the case for all surfaces, for example, brushed metal or plastics. Especially for plastic objects or objects covered with gloss varnish, a more elaborate reflectance model is



Color Constancy, Fig. 3 Local space average color after 0, 200 and 2,000 iterations

more appropriate. The dichromatic reflection model [17, 18] assumes that reflectance is composed of interface reflectance, which occurs at the boundary between the object's surface and air, and body reflectance which is due to the scattering of light below the object's surface. In other words, the reflection of light from the object's surface is assumed to be partially specular



Color Constancy, Fig. 4 (a and b) input images (c and d) results using local space average color

and partially diffuse. Color constancy algorithms have also been developed using the dichromatic reflection model, for example, by Risson [19].

Application

Color constancy algorithms are ideal for color correction in digital photography. In digital photography, the goal is to obtain a color-corrected image that corresponds nicely to human perception. A printed photograph or a photograph viewed on a computer display should appear in exactly the same way that the human observer (the photographer) perceived the scene. Besides digital photography, color constancy algorithms can be applied in the context of most computer vision tasks. For many tasks, one should try to estimate object reflectances. For instance, image segmentation would not be as difficult, if object reflectance could be correctly determined. Similarly, color-based object recognition is easier if performed on reflectance information. Thus, color constancy

algorithms should often be applied as a preprocessing step. This holds especially for autonomous mobile robots equipped with a vision system because autonomous mobile robots need to operate in different environments under different illuminants.

Experimental Results and Datasets

A comparison of computational color constancy algorithms is given by Barnard et al. [20] for synthetic as well as real image data. Another detailed comparison of color constancy algorithms along with pseudocode for many algorithms is given by Ebner [1]. Data for computer vision and computational color science can be found at the Simon Fraser University, Canada (www.cs.sfu.ca/~colour/data/). A repository has also been created by the color group at the University of East Anglia, UK (www.colour-research.com). A database for spectral color science is available from the University of Eastern Finland, Finland (spectral.joensuu.fi/).

References

1. Ebner M (2007) Color constancy. Wiley, England
2. Maloney LT (1999) Physics-based approaches to modeling surface color perception. In: Gegenfurtner KR, Sharpe LT (eds) *Color vision: from genes to perception*. Cambridge University Press, Cambridge, pp 387–422
3. Zeki S (1993) *A vision of the brain*. Blackwell, Oxford
4. Geusebroek JM, van den Boomgaard R, Smeulders AWM, Geerts H (2001) Color invariance. *IEEE Trans Pattern Anal Mach Intell* 23(12):1338–1350
5. Finlayson GD, Hordley SD (2001) Color constancy at a pixel. *J Opt Soc Am A* 18(2):253–264
6. Richards W, Parks EA (1971) Model for color conversion. *J Opt Soc Am* 61(7):971–976
7. West G, Brill MH (1982) Necessary and sufficient conditions for von Kries chromatic adaptation to give color constancy. *J Math Biol* 15:249–258
8. Land EH, McCann JJ (1971) Lightness and retinex theory. *J Opt Soc Am* 61(1):1–11
9. Buchsbaum G (1980) A spatial processor model for object colour perception. *J Frankl Inst* 310(1):337–350
10. Forsyth DA (1990) A novel algorithm for color constancy. *Int J Comput Vis* 5(1):5–36
11. Finlayson GD (1996) Color in perspective. *IEEE Trans Pattern Anal Mach Intell* 18(10):1034–1038
12. van de Weijer J, Gevers T, Gijsenij A (2007) Edge-based color constancy. *IEEE Trans Image Process* 16(9): 2207–2214
13. Brainard DH, Freeman WT (1997) Bayesian color constancy. *J Opt Soc Am A* 14(7):1393–1411
14. Horn BKP (1974) Determining lightness from an image. *Comput Graph Image Process* 3:277–299
15. Blake A (1985) Boundary conditions for lightness computation in mondrian world. *Comput Vis Graph Image Process* 32:314–327
16. Barnard K, Finlayson G, Funt B (1997) Color constancy for scenes with varying illumination. *Comput Vis Image Underst* 65(2):311–321
17. Klinker GJ, Shaffer SA, Kanade T (1988) The measurement of highlights in color images. *Int J Comput Vis* 2:7–32
18. Tominaga S (1991) Surface identification using the dichromatic reflection model. *IEEE Trans Pattern Anal Mach Intell* 13(7):658–670
19. Risson VJ (2003) Determination of an illuminant of digital color image by segmentation and filtering. United States Patent Application, Pub. No. US 2003/0095704 A1
20. Barnard K, Cardei V, Funt B (2002) A comparison of computational color constancy algorithms – Part I and II. *IEEE Trans Image Process* 11(9):972–996

Color Difference

- [Color Similarity](#)

Color Discrepancy

- [Color Similarity](#)

Color Dissimilarity

- [Color Similarity](#)

Color Management

- [Gamut Mapping](#)

Color Model

Shoji Tominaga
Graduate School of Advanced Integration Science,
Chiba University, Inage-ku, Chiba, Japan

Synonyms

[Color spaces](#); [Color specification systems](#)

Definition

A color model is a mathematical model describing the way colors are specified by three-dimensional coordinates in such terms as three numerical values, three perceptual attributes, or three primary colors.

Background

The human retina contains three types of cones L, M, and S as photoreceptors, which respond best to light of long, medium, and short wavelengths, respectively, in the visible range. Since the perception of

color depends on the response of these cones, it follows in principle that visible color can be mapped into a three-dimensional space in terms of three numbers.

Trichromatic color vision theory is closely related to such three types of color sensors. The trichromatic theory by Young and Helmholtz suggests that any colors and spectra in the visible range can be visually matched using mixtures of three primary colors. These primary colors are usually red, green, and blue. Most image output devices such as television screens, computer and video displays, and image projectors create visible colors by using additive mixtures of the three primary colors.

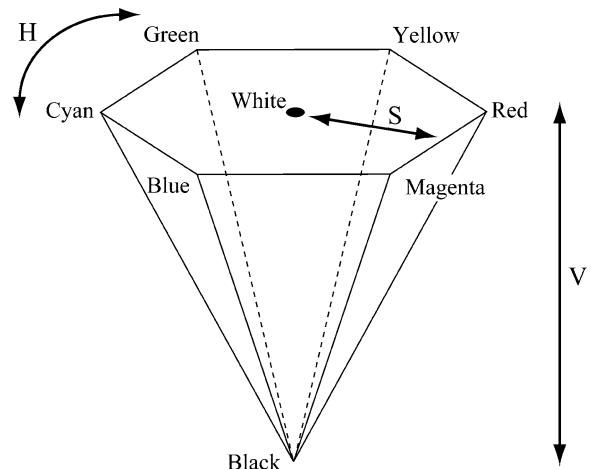
Digital cameras also capture color by using the same principle of trichromatic color vision. They usually use three different types of sensors which primarily respond to the red, green, and blue parts of the visible spectrum. Therefore, a color model mathematically describing the color coordinate system is crucial for analysis, evaluation, and the rendering of color images.

Theory

Color Model for RGB Color Signals

Most color models used in computer vision and image processing are based on the RGB primaries. This is because color images captured by digital cameras are all represented by the RGB primaries, and image output devices such as displays and projectors are based on an additive mixture of the RGB primaries. Therefore, the simplest color model (called RGB color space) uses a Cartesian coordinate system defined by (R, G, B) triples. This system, however, is not available for colorimetry, because the spectral-sensitivity curves of cameras are generally not coincident with the color-matching functions.

The RGB space is not a perceptually uniform color space. In computer graphics and image processing applications, approximately uniform spaces derived from RGB are defined in terms of the three attributes: hue (H), saturation (S), and value (V) (representing lightness). For example, the HSV model by Smith [9] was defined as a nonlinear transform of RGB given in 8 bits:



Color Model, Fig. 1 Single hexagonal color space

$$\begin{aligned} H &= \tan^{-1} \left\{ \frac{\sqrt{3}(g-b)}{(r-g)+(r-b)} \right\} \\ S &= 1 - \text{Min}/V \\ V &= \text{Max} \end{aligned} \quad (1)$$

where

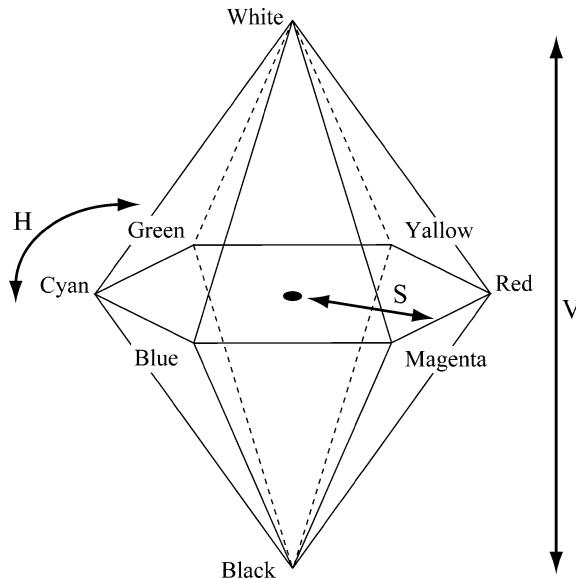
$$\begin{aligned} r &= R/255, \quad g = G/255, \quad b = B/255, \\ \text{Max} &= \max(r, g, b), \quad \text{Min} = \min(r, g, b). \end{aligned}$$

This model represents a hexagonal color space as seen in Fig. 1, where the saturation decreases monotonically as the value decreases. Figure 2 represents the improved HSV model to a double hexagonal space, which is defined as:

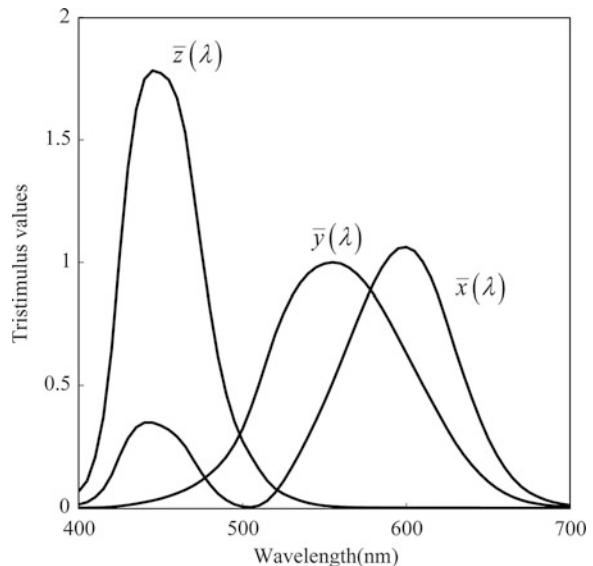
$$\begin{aligned} H &= \tan^{-1} \left\{ \frac{\sqrt{3}(g-b)}{(r-g)+(r-b)} \right\} \\ S &= \text{Max} - \text{Min} \\ V &= (R+g+b)/3 \end{aligned} \quad (2)$$

Color Model for Colorimetry

The CIE-XYZ color system for colorimetry was created by the International Commission on Illumination (CIE). This color system was derived from a series of visual experiments of color matching [1]. The tristimulus values of a color were used for representing



Color Model, Fig. 2 Double hexagonal color space



Color Model, Fig. 3 CIE 1931 2° Standard Observer color-matching functions

the amounts of three primary colors needed to match the test color. The tristimulus values depend on the observer's field of view. Therefore, the CIE defined the standard observer and a set of three color-matching functions, called $\bar{x}(\lambda)$, $\bar{y}(\lambda)$, and $\bar{z}(\lambda)$. The chromatic response of the standard observer is characterized by the three color-matching functions. Figure 3 shows the CIE 1931 2° standard observer color-matching functions.

The tristimulus values for a color signal with a spectral-power distribution $L(\lambda)$ are then given by:

$$\begin{aligned} X &= k \int_{\lambda} L(\lambda) \bar{x}(\lambda) d\lambda \\ Y &= k \int_{\lambda} L(\lambda) \bar{y}(\lambda) d\lambda \\ Z &= k \int_{\lambda} L(\lambda) \bar{z}(\lambda) d\lambda \end{aligned} \quad (3)$$

where the integration on the wavelength λ is normally calculated over the range of visible spectrum 400–700 nm, and the coefficient k is a normalizing constant. In application to photometry, the Y tristimulus value becomes the luminance of the color signal. In application to object color, the tristimulus values become:

$$\begin{aligned} X &= k \int_{\lambda} E(\lambda) S(\lambda) \bar{x}(\lambda) d\lambda \\ Y &= k \int_{\lambda} E(\lambda) S(\lambda) \bar{y}(\lambda) d\lambda \\ Z &= k \int_{\lambda} E(\lambda) S(\lambda) \bar{z}(\lambda) d\lambda \end{aligned} \quad (4)$$

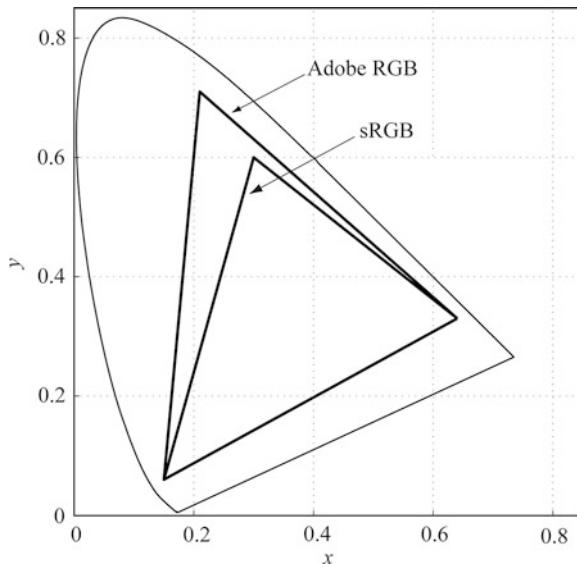
where $S(\lambda)$ is the surface-spectral reflectance of the illuminated object, and $E(\lambda)$ is the spectral-power distribution of the light source illuminating the object. In this case, the Y tristimulus value becomes the luminance factor of the object-color stimulus. The constant factor is usually chosen as:

$$k = 100 \int_{\lambda} S(\lambda) \bar{x}(\lambda) d\lambda. \quad (5)$$

The chromaticity coordinates (x, y) of a given color are defined as:

$$x = \frac{X}{X + Y + Z}, \quad y = \frac{Y}{X + Y + Z}. \quad (6)$$

Figure 4 depicts the chromaticity diagram, where all visible chromaticities are located within the horseshoe-shaped region. The outer curved boundary is the spectral locus, corresponding to monochrome light with the



Color Model, Fig. 4 CIE 1931 chromaticity diagram

most saturated color. Less saturated colors are located in the inside of the region with white at the center. The triangle of sRGB represents the gamut of a standard RGB color space proposed by HP and Microsoft in 1996 [10]. Recently, usual monitors, printers, and the Internet are based on this standard space. The triangle of Adobe RGB represents the gamut of an RGB color space developed by Adobe Systems in 1998 [11]. This color space, improving the gamut of sRGB primarily in cyan-green, is used for desktop publishing.

It should be noted that the CIE XYZ tristimulus values were not defined for color differences. Two colors with a constant difference in the tristimulus values may look very different, depending on where the two colors are located in the (x, y) chromaticity diagram.

Color Model for Uniform Color Space

Many people believe that a uniform color space is most useful for applications where perceptual errors are evaluated. The CIE 1976 L*a*b* color system (also called CIE LAB color system) was designed to approximate a perceptually uniform color space in terms of the tristimulus values XYZ. It is defined as follows:

$$L^* = 116(Y/Y_n)^{1/3} - 16$$

$$a^* = 500[(X/X_n)^{1/3} - (Y/Y_n)^{1/3}] \quad (7)$$

$$b^* = 200[(Y/Y_n)^{1/3} - (Z/Z_n)^{1/3}]$$

where (X_n, Y_n, Z_n) are the tristimulus values of the reference white point. This system provides an opponent-color space with dimension L^* for lightness, and a^* and b^* for opponent-color dimensions. $L^*=0$ and $L^*=100$ indicate black and white, respectively. The axes of a^* and b^* take the position between red and green, and the position between yellow and blue, respectively.

The two-dimensional chromaticity diagram is expressed in the orthogonal coordinate system of (a^*, b^*) . The chromaticity components can also be expressed in a cylindrical system of chroma and hue, where the chroma and hue-angle are defined respectively by:

$$C_{ab}^* = (a^{*2} + b^{*2})^{1/2} \quad (8)$$

$$h_{ab} = \arctan^{-1}(b^*/a^*)$$

The color difference between two color stimuli 1 and 2 is defined as follows:

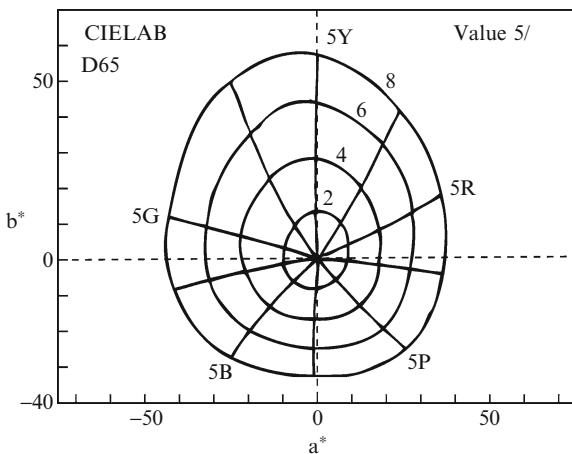
$$\Delta E_{ab}^* = [(\Delta L^*)^2 + (\Delta a^*)^2 + (\Delta b^*)^2]^{1/2}, \quad (9)$$

where $\Delta L^* = L_1^* - L_2^*$, $\Delta a^* = a_1^* - a_2^*$, and $\Delta b^* = b_1^* - b_2^*$. Figure 5 depicts the Munsell color system with value 5 on the (a^*, b^*) chromaticity coordinates under the reference condition of illuminant D_{65} . The color system approximates roughly the Munsell uniform color space.

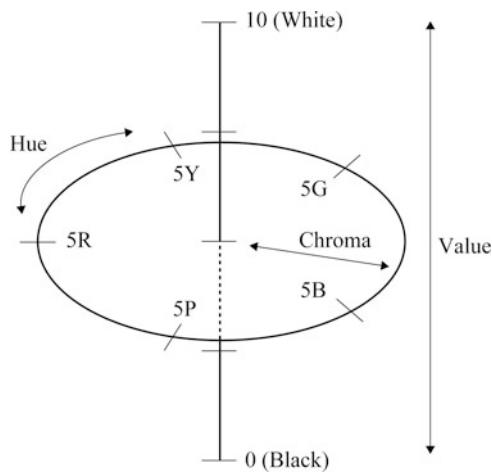
Color Model for Color Order System

Color order systems were created to order colors in terms of intuitive perceptual attributes. Most colored samples, used as a reference in many design and engineering applications, are made in equal steps of perceptual attributes.

The Munsell color order system is one of the most widely used systems in the world. This system is based on the three perceptual attributes: hue, value (representing lightness), and chroma (representing saturation), and is organized in a cylindrical coordinate system as shown in Fig. 6. Hue is arranged as a circle,



Color Model, Fig. 5 Munsell color system with Value 5 on the (a^*, b^*) chromaticity coordinates



Color Model, Fig. 6 Organization of the Munsell color system

value as the axis perpendicular to the hue circle, and chroma as the distance from the center.

The Munsell color samples are arranged in equal steps of each attribute, where each sample is described using a three-part syntax of hue chroma/value. For example, 5YR 8/4 indicates a color sample with the hue of 5YR (yellow-red), the chroma of level 4, and the value of level 8. A conversion table to reproduce the Munsell color samples in terms of the CIE tristimulus values was given by Newhall et al. [12]. The table contains the luminance factor Y and chromaticity coordinates (x , y) equivalent to the Munsell renotation system (H, V, C), under the condition of the CIE standard illuminant C. It should be noted that there is no

formula for a color notation conversion between the Munsell and CIE system, except for the table. Therefore, the conversion was carried out by interpolating the table data.

Obviously, this method is not efficient from the point of view of mass data processing by computers. Tominaga [13] developed a method for color notation conversion, between the Munsell and CIE systems by means of neural networks. In the neural network method, it is not necessary to use the special database of color samples. This is because the knowledge of conversion between the two color spaces is stored in a small set of the weighting parameters in a multilayer feedforward network.

References

- Wyszecki G, Stiles WS (1982) Color science: concepts and methods, quantitative data and formulae. Wiley, New York
- Hall R (1989) Illumination and color in computer-generated imagery. Springer, New York
- Foley JD, van Dam A, Feiner SK, Hughes JF (1990) Computer graphics: principles and practice, 2nd edn. Addison-Wesley, Reading
- Wandell BA (1995) Foundations of vision. Sinauer Associates, Sunderland
- Shevell SK (2003) The science of color, 2nd edn. Elsevier, Oxford
- Lee H-C (2005) Introduction to color imaging science. Cambridge University Press, Cambridge
- Fairchild MD (2005) Color appearance models. Wiley, Chichester
- Shanda J (2007) Colorimetry. Wiley, Hoboken
- Smith AR (1978) Color gamut transform pairs. Comput Graph 12(3):12–19
- Stokes M, Anderson M, Chandrasekar S, Motta R (1996) A standard default color space for the Internet: sRGB. Version 1.10, International Color Consortium
- Adobe Systems (2005) Adobe RGB(1998)b color image encoding. Version 2005-5, Adobe Systems
- Newhall SM, Nickerson D, Judd DB (1943) Final report of the O.S.A. subcommittee on the spacing of the munsell colors. J Opt Soc Am 33(7):385–411
- Tominaga S (1992) Color classification of natural color images. Color Res Appl 17(4):230–239
- Brainard DH (1989) Calibration of a computer controlled color monitor. Color Res Appl 14:23–34
- Pratt WK (1991) Digital image processing, 2nd edn. Wiley, New York
- Ohta N, Robertson AR (2005) Colorimetry. Wiley, Chichester
- Hunt RWG (2004) The reproduction of colour, 6th edn. Wiley, Chichester
- Green P (2010) Color management. Wiley, Chichester
- Watt A (2000) 3D computer graphics, 3rd edn. Addison Wesley, Reading



Color Similarity

Takahiko Horiuchi and Shoji Tominaga
Graduate School of Advanced Integration Science,
Chiba University, Inage-ku, Chiba, Japan

Synonyms

Color difference; Color discrepancy; Color dissimilarity

Definition

Color similarity is a measure that reflects the strength of relationship between two colors.

Background

A measure of similarity between colors is needed when one color region is matched to another region among color images, for example, a purple region is more similar to a blue region than a green region. Therefore, the similarity between two colors is an important metric in not only color science but also imaging science and technology (including computer vision). If color similarity is measured, it allows people to determine the strength of relationship in terms of numerical values; otherwise, it will be determined in nonnumerical ways.

Thus, the benefit of color similarity is its application to various techniques of image analysis, such as noise removal filters, edge detectors, object classification, and image retrieval.

Theory

Color similarity is usually represented in the range of either interval $[-1, 1]$ or interval $[0, 1]$. Let $\mathbf{x}_i = [x_{i1}, x_{i2}, x_{i3}]$ be a three-dimensional vector specifying a color feature at location i in a color image. The similarity measure $s(\mathbf{x}_i, \mathbf{x}_j)$ between two color features is a symmetric function whose value is larger when color features \mathbf{x}_i and \mathbf{x}_j are closer. The color feature is specified by three coordinates in color space (or color model). Color space is defined in many different ways

in color and image science, where color spaces such as RGB, $L^*a^*b^*$, $L^*u^*v^*$, YIQ, and HSI are often used.

On the other hand, color dissimilarity is also used for measuring the discrepancy between two colors. This quantity is usually represented in a positive range of $[0, 1]$ because the dissimilarity is measured by a “normalized distance” between two colors. A dissimilarity measure $d(\mathbf{x}_i, \mathbf{x}_j)$ is a symmetric function whose value is larger when color features \mathbf{x}_i and \mathbf{x}_j are more dissimilar. Therefore, a relationship between two measures of color similarity and color dissimilarity is given by:

$$s(\mathbf{x}_i, \mathbf{x}_j) = 1 - d(\mathbf{x}_i, \mathbf{x}_j), \quad (1)$$

where the color similarity $s(\mathbf{x}_i, \mathbf{x}_j)$ is bounded by $[0, 1]$. Note that when the color similarity is one (i.e., exactly same), the color dissimilarity is zero, and when color similarity is zero (i.e., quite different), the color dissimilarity is one. If the color similarity is defined in the range $[-1, 1]$, then:

$$s(\mathbf{x}_i, \mathbf{x}_j) = 1 - 2d(\mathbf{x}_i, \mathbf{x}_j). \quad (2)$$

Note in this case that the color dissimilarity of 1 (i.e., quite different) corresponds to the color similarity of -1 , and when the color dissimilarity of 0 (i.e., exactly same) corresponds to the color similarity of 1. In many cases, measuring dissimilarity is easier than measuring similarity. Once dissimilarity is measured, it can easily be normalized and converted to the similarity measure. Therefore, color dissimilarity measures are first described in the following.

Color Dissimilarity Measures

The most commonly used dissimilarity measure to quantify the distance between two color vectors \mathbf{x}_i and \mathbf{x}_j is the weighted Minkowski metric:

$$d(\mathbf{x}_i, \mathbf{x}_j) = c \left(\sum_{k=1}^3 \xi_k |x_{ik} - x_{jk}|^L \right)^{1/L}, \quad (3)$$

where c is the nonnegative scaling parameter for $d(\mathbf{x}_i, \mathbf{x}_j) \in [0, 1]$ and the exponent L defines the nature of the distance metric. The parameter ξ_k , for $\sum_k \xi_k = 1$, measures the weight assigned to the color channel k . Usually it is determined $\xi_k = 1/3, \forall k$ (e.g., [1, 2]). In the case $L = 1$ in (Eq. 3), the

measure is known as Manhattan distance, which is also called city-block distance, absolute value distance, and taxicab distance [3].

$$d(\mathbf{x}_i, \mathbf{x}_j) = c \sum_{k=1}^3 |x_{ik} - x_{jk}|. \quad (4)$$

The dissimilarity measure represents distance between points in a city road grid. It examines the absolute differences between a pair of colors. In the case $L = 2$ in (Eq. 3), the measure is known as Euclidean distance which is the most common use of distance.

$$d(\mathbf{x}_i, \mathbf{x}_j) = c \left(\sum_{k=1}^3 (x_{ik} - x_{jk})^2 \right)^{1/2}. \quad (5)$$

In the case $L \rightarrow \infty$ in (Eq. 3), the measure is known as Chebyshev distance which is also called Kolmogorov-Smirnov Statistic [4], chess-board distance, and maximum value distance. It examines the absolute magnitude of the differences between coordinates of a pair of objects.

$$d(\mathbf{x}_i, \mathbf{x}_j) = c \max_k |x_{ik} - x_{jk}|. \quad (6)$$

As a well-known dissimilarity metric based on the Minkowski metric, [6] color difference ΔE_{ab}^* is useful [5, 7]. ΔE_{ab}^* is the Euclidean distance between two colors in CIEL*a*b* color space as follows:

$$\begin{aligned} \Delta E_{ab}^*(\mathbf{x}_i, \mathbf{x}_j) &= \left((L_i^* - L_j^*)^2 + (a_i^* - a_j^*)^2 \right. \\ &\quad \left. + (b_i^* - b_j^*)^2 \right)^{1/2}, \end{aligned} \quad (7)$$

where $\mathbf{x}_i = [L_i^*, a_i^*, b_i^*]$ and $\mathbf{x}_j = [L_j^*, a_j^*, b_j^*]$. Please note that ΔE_{ab}^* is used as a “distance” with $c = 1$ in (Eq. 5). When $\Delta E_{ab}^* \approx 2.3$, it corresponds to a just noticeable difference (JND) of surface colors [8, 9]. Perceptual nonuniformities in the underlying CIEL*a*b* color space, however, have led to the CIE refining their 1976 definition over the years. The refined color difference ΔE_{94}^* is defined in the L*C*h color space [10]. Given a reference color $\mathbf{x}_i = [L_i^*, C_i^*, h_i^*]$ and another color $\mathbf{x}_j = [L_j^*, C_j^*, h_j^*]$,

the difference is:

$$\begin{aligned} \Delta E_{94}^*(\mathbf{x}_i, \mathbf{x}_j) &= \left(\left(\frac{L_i^* - L_j^*}{K_L} \right)^2 + \left(\frac{C_i^* - C_j^*}{1 + K_1 C^*} \right)^2 \right. \\ &\quad \left. + \left(\frac{h_i^* - h_j^*}{1 + K_2 C^*} \right)^2 \right)^{1/2} \end{aligned} \quad (8)$$

where C^* represents the geometrical average of chroma, and K_L , K_1 , and K_2 represent the weighting factors which depend on the application (in original definition, $K_L = 1$, $K_1 = 0.045$, and $K_2 = 0.015$ for graphic arts). Since the 1994 definition did not adequately resolve the perceptual uniformity issue, the CIE refined their definition, adding five corrections:

1. A hue rotation term (RT), to deal with the problematic blue region
2. Compensation for neutral colors
3. Compensation for lightness
4. Compensation for chroma
5. Compensation for hue

The definition of the CIEDE2000 color difference is explained in Ref. [11].

Color Similarity Measure

The simplest color similarity measure is an angular separation similarity. It represents a cosine angle between two color vectors, which is often called as coefficient of correlation. In this measure, similarity in orientation is expressed through the normalized inner product as:

$$s(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j / |\mathbf{x}_i| |\mathbf{x}_j| \quad (9)$$

which corresponds to the cosine of the angle between the two color vectors \mathbf{x}_i and \mathbf{x}_j . Since similar colors have almost parallel orientations, (*while significantly different colors point in different directions in a 3-D color space*) the normalized inner product can be used to quantify orientation similarity between the two color vectors.

The correlation coefficient is a standardized angular separation by centering the coordinates to their mean value. Let $\bar{x}_i = \frac{1}{p} \sum_{k=1}^p x_{ik}$ and $\bar{x}_j = \frac{1}{p} \sum_{k=1}^p x_{jk}$; the correlation coefficient between these two vectors \mathbf{x}_i and \mathbf{x}_j is given as follows:



$$s(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_{k=1}^p |x_{ik} - \bar{x}_i| |x_{jk} - \bar{x}_j|}{\left(\sum_{k=1}^p (x_{ik} - \bar{x}_i)^2\right)^{1/2} \left(\sum_{k=1}^p (x_{jk} - \bar{x}_j)^2\right)^{1/2}}. \quad (10)$$

In (Eq. 10) the color similarity between two points is shown as $p = 3$ shows. Then the correlation coefficient can be applied to color region similarity, such as textured regions by setting $p = 3 \times n$ (where n is the number of pixels).

There are many other methods to measure the similarity between two color vectors. So depending on the nature and the objective of the problem at hand, it is possible that one method is more appropriate than the other.

Application

The formulation of similarity between two color vectors is of paramount importance for the development of the vector processing techniques. These include noise removal filters, edge detectors, image zoomers, and image retrievals.

References

1. Kruskal JB (1964) Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika* 29(1):1–27. doi:10.1007/BF02289565
2. Duda RO, Hart PE (1973) Pattern classification and scene analysis. Wiley, New York
3. Krause EF (1987) Taxicab geometry: adventure in non-euclidean geometry. Dover, New York. ISBN 0-486-25202-7
4. Geman D, Geman S, Graffigne C, Dong P (1990) Boundary detection by constrained optimization. *IEEE Trans Pattern Anal Mach Intell* 12(7):609–628
5. Pauli H (1976) Proposed extension of the CIE recommendation on uniform color spaces, color difference equations, and metric color terms. *J Opt Soc Am* 66(8):866–867. doi:10.1364/JOSA.66.000866
6. CIE (2004) Colorimetry. Vienna, CIE Pb. 15:2004, ISBN 978-3-901906-33-6
7. CIE Publication (1978) Recommendations on uniform color spaces, color difference equations, psychometric color terms. 15(Suppl 2), Paris, (E.-1.3.1) 1971/(TC-1.3)
8. Sharma G (2003) Digital color imaging handbook, 1.7.2 edn. CRC, Florida. ISBN 084930900X
9. Mahy M, Eycken LV, Oosterlinck A (1994) Evaluation of uniform color spaces developed after the adoption of CIELAB and CIELUV. *Color Res Appl* 19(2): 105–121

10. CIE Publication (1995) Industrial colour-difference evaluation. Vienna, CIE 116–1995, ISBN 978-3-900734-60-2
11. CIE Publication (2001) Improvement to industrial colour-difference evaluation. Vienna, CIE 142–2001, ISBN 978-3-901906-08-4

Color Spaces

Rajeev Ramanath¹ and Mark S. Drew²

¹DLP® Products, Texas Instruments Incorporated, Plano, TX, USA

²School of Computing Science, Simon Fraser University, Vancouver, BC, Canada

Synonyms

Color appearance models; Color model

Definition

A color space describes the range of colors – the *gamut* – that an imaging device or software has to work with. Consequently, the design of these color spaces allows a user to modify images in a predefined manner based on the specific needs of the application.

Theory

Color spaces may be generally separated into those that are defined for analysis of color by color scientists (*colorimetric color spaces*) and those that are used for image/color editing.

Color Spaces for Colorimetric Analysis

Color spaces for colorimetric analysis are typically based on the human observer. Central to such color spaces is the *CIEXYZ color matching functions*, which are based on *CIERGB* color matching functions – based in turn on the LMS cone fundamentals. CIE is an abbreviation for the International Commission on Illumination (Commission Internationale de L'Eclairage) that is the body that is responsible for standardization of data in this field. However, the ability to directly measure the cone functions of an observer is only a recent development. Researchers had originally



inferred the cone functions based on psychophysical experiments measuring the RGB color matching functions. Consequently, the CIE had standardized the color matching functions long before the cone functions could be directly measured. Details of how the color matching functions were developed are well documented in the book by Wyszecki and Stiles that is arguably a cornerstone of colorimetry [27].

The RGB color matching functions, denoted $\{\bar{r}(\lambda), \bar{g}(\lambda), \bar{b}(\lambda)\}$, are shown in Fig. 1a.

Note that these color matching functions have negative excursions, due to the fact that there are some colors that reside outside the triangle formed by these three primaries – a negative excursion in the primaries' weights is the only way to represent a color outside this triangle. In order to address this problem, the CIE also published a set of color matching functions with nonnegative values, denoted $\bar{x}(\lambda), \bar{y}(\lambda), \bar{z}(\lambda)$ and shown in Fig. 1b. These are known generally as the CIEXYZ color matching functions. Note that these are now fictitious, nonphysical primaries. The equations for computing these special “tristimulus” values, in XYZ color space, are:

$$\begin{aligned} X &= k \int_{\lambda} \bar{x}(\lambda) i_r(\lambda) d\lambda \\ Y &= k \int_{\lambda} \bar{y}(\lambda) i_r(\lambda) d\lambda \\ Z &= k \int_{\lambda} \bar{z}(\lambda) i_r(\lambda) d\lambda. \end{aligned} \quad (1)$$

where $i_r(\lambda)$ denotes the spectral power distribution of the light energy incident on the retina, and k denotes a normalization factor that is set to 683 lumens/Watt in the case of absolute colorimetry and to $100 / \int_{\lambda} \bar{y}(\lambda) i_r(\lambda) d\lambda$ for relative colorimetry. In the case of relative colorimetry, this means a value of $Y = 100$ denotes the brightest color – the illuminant reflecting from a perfect reflecting diffuser.

The first set of color matching functions published by the CIE were originally empirically determined for a 2° field – the bipartite field used for matching a subtended 2° angle on the observers' retina. Following the 1931 publication, W. S. Stiles and J. M. Burch conducted experiments [25] to measure color matching functions for larger fields of view. This was combined with the findings of Speranskaya [24] into the publication by the CIE of a 10° observer in 1964 [5]. The difference between these two standard observers

is significant enough to warrant a clear specification of which observer color matching functions are used in experimental work. More specifically, the 10° observer has noticeable shifts of the color matching functions in the blue direction due to the fact that the subtense of the stimulus encompasses a larger portion of the retina and hence more S cones and also increased macular pigment absorption.

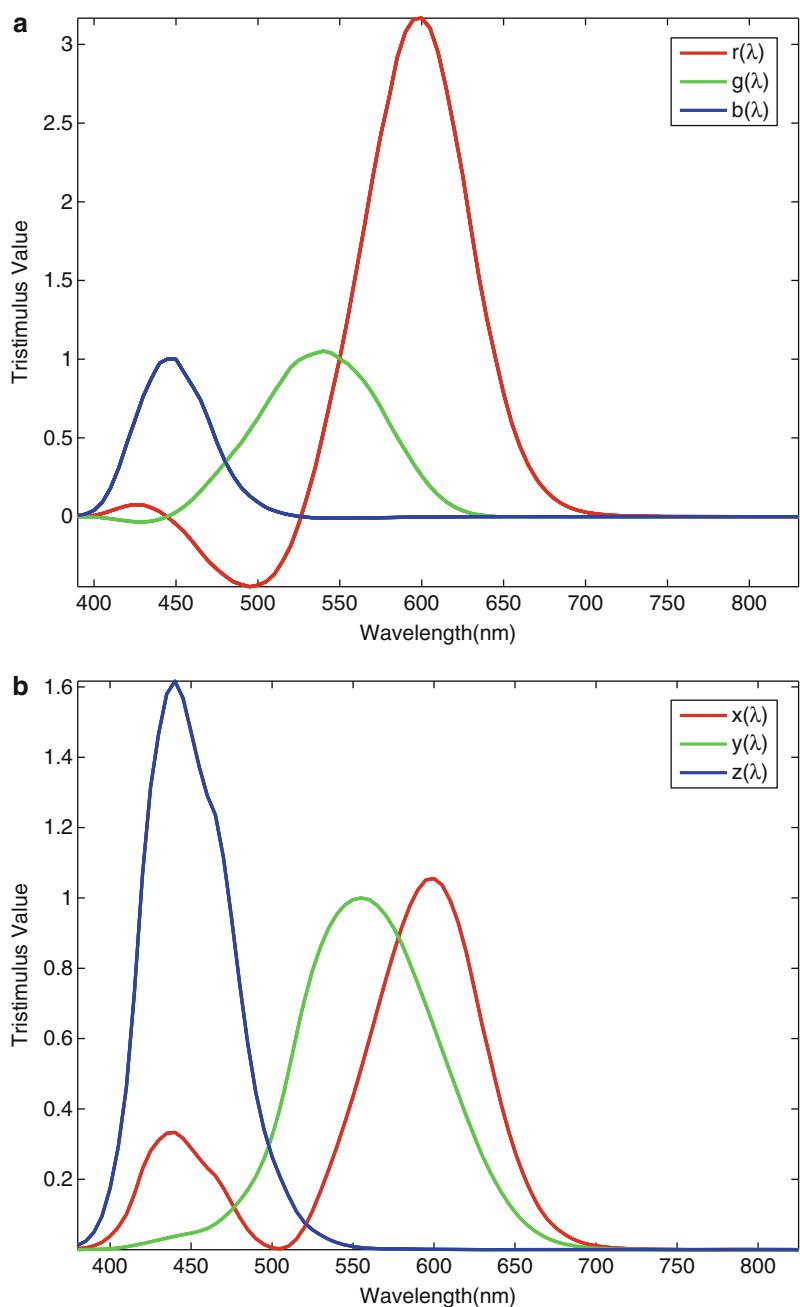
In the CIE colorimetric system, an XYZ tristimulus value uniquely specifies a color. However, a convenient 2D representation of the tristimulus values led to the projection of the tristimulus values by normalizing by the sum of the three values. These “chromaticity” values are given by:

$$\begin{aligned} x &= \frac{X}{X + Y + Z} \\ y &= \frac{Y}{X + Y + Z}. \end{aligned} \quad (2)$$

A color may be specified uniquely by its (x, y) chromaticity coordinates and its luminance Y , and is often used to describe a color since the tristimulus values are straightforward to derive from the (x, y, Y) values. The biggest advantage of the (x, y) chromaticity coordinates is that they specify a magnitude-independent hue and purity of a color. A chromaticity diagram (see Fig. 2) is typically used to specify a color using its chromaticity coordinates. In Fig. 2, the horseshoe-shaped locus denotes the locus of monochromatic stimuli visible to the standard 2° observer (the gamut of visible colors). Shorter wavelength stimuli (starting at 380 nm, eliciting a relatively strong blue response) reside in the lower left of this horseshoe shape while the longer wavelengths (ending at 830 nm, eliciting a relatively strong red response) reside on the lower right, with the top of the horseshoe curve around 520 nm (eliciting a strong green response). The line connecting the blue and red corners is referred to as the *line of purples*. Colors on this line, although on the border of the gamut, have no counterpart in monochromatic sources of light and hence have no wavelengths associated with them.

The (x, y) chromaticity diagram is perceptually nonuniform: Unit vectors in the chromaticity space do not correspond to a unit change in perception even if the luminance is kept constant. In an attempt to improve the uniformity of the chromaticity diagram, in 1976, the CIE published a uniform chromaticity scale (UCS) diagram that scaled and normalized the XYZ

Color Spaces, Fig. 1 (a) RGB color matching functions (b) XYZ color matching functions including Judd-Vos modifications

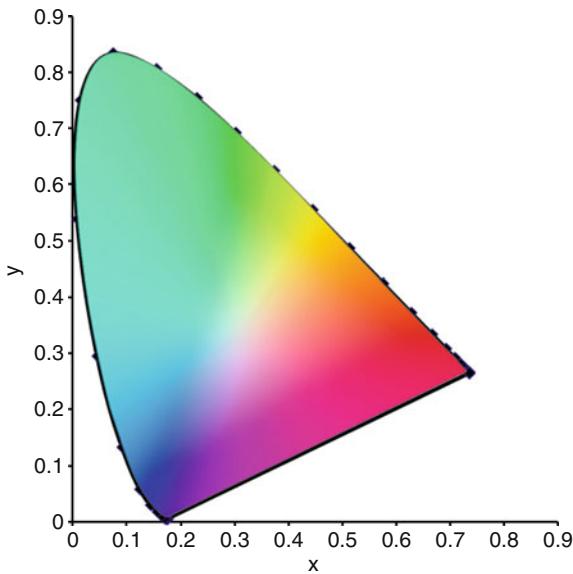


tristimulus values [6]. This chromaticity diagram is denoted by u' , v' axes, which are related to the XYZ tristimulus values by the following equations:

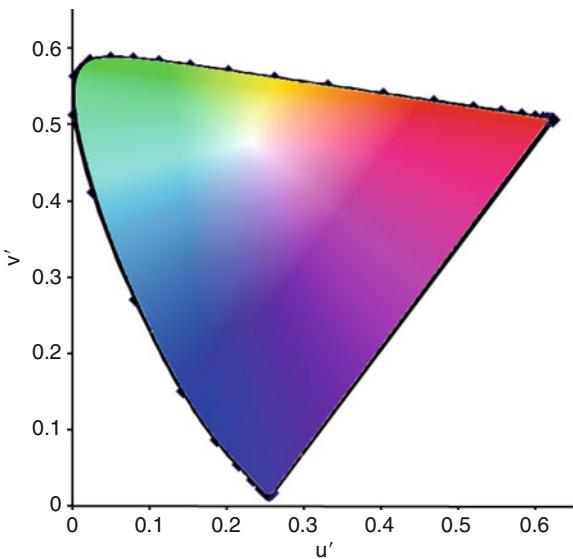
$$\begin{aligned} u' &= \frac{4X}{X + 15Y + 3Z} \\ v' &= \frac{9Y}{X + 15Y + 3Z}. \end{aligned} \quad (3)$$

Figure 3 shows the UCS. Just as in the case of Fig. 2, in this figure, the horseshoe-shaped locus represents the gamut of visible colors.

The CIEXYZ color space does not have perceptual correlates that would make it useful for common use. In an attempt to add perceptual behavior to color spaces, based on earlier works of many researchers, the CIE proposed a lightness scale along with two



Color Spaces, Fig. 2 CIE xy chromaticity diagram for a 2° observer



Color Spaces, Fig. 3 CIE UCS $u'v'$ chromaticity diagram for a 2° observer

chromatic scales. The CIELAB color space is one such space, where the axes are denoted by L^* (Lightness), a^* (redness–greenness), and b^* (yellowness–blueness). For a stimulus given by a tristimulus value X , Y , and Z , the CIELAB coordinates are given by:

$$L^* = 116 f(Y/Y_n) - 16$$

$$\begin{aligned} a^* &= 500 [f(X/X_n) - f(Y/Y_n)] \\ b^* &= 200 [f(Y/Y_n) - f(Z/Z_n)], \quad \text{where} \\ f(t) &= t^{1/3}, \quad \text{for } t > 0.008856 \\ f(t) &= 7.787 t + 16/116 \quad \text{otherwise.} \end{aligned} \quad (4)$$

In the above equations, the subscript n denotes the tristimulus values corresponding to the reference white chosen, which makes the CIELAB color space a relative color space. Given the CIELAB coordinates in a three-dimensional space, correlates of chroma and hue may be derived as follows:

$$C_{ab}^* = (a^{*2} + b^{*2})^{1/2} \quad (5)$$

$$h_{ab}^* = \tan^{-1}(b^*/a^*). \quad (6)$$

Under highly controlled viewing conditions, a CIELAB ΔE difference of 1 correlates with a single just noticeable difference in color. It is to be noted though that the CIELAB color difference measure was designed for color differences between uniform color patches in isolation.

In a similar construct, the CIE also recommended a CIELUV color space, based on the uniform chromaticity scale (UCS). This uses a subtractive shift from the reference white instead of the normalization based on division that is used in the CIELAB space. The equations to transform a tristimulus value from u^*, v^* coordinates to CIELUV are given by:

$$\begin{aligned} L^* &= 116 f(Y/Y_n) - 16 \\ u^* &= 13 L^*(u' - u'_n) \\ v^* &= 13 L^*(v' - v'_n), \quad \text{where} \\ f(t) &= t^{1/3}, \quad \text{for } t > 0.008856 \\ f(t) &= 7.787 t + 16/116 \quad \text{otherwise.} \end{aligned} \quad (7)$$

The u', v' coordinates for a tristimulus value are computed using (Eq. 3). As in the CIELAB definitions, the subscript n denotes the u', v' coordinates of the reference white being used. The descriptions of the u^* and v^* axes are similar to those in CIELAB: approximating redness–greenness and yellowness–blueness directions.

Based on these correlates, the CIE recommends that color difference measures in the two uniform-perception spaces CIELAB and CIELUV be given by the Euclidean difference between the coordinates of two color samples:

$$\begin{aligned}\Delta E^*_{ab} &= \left[(\Delta L^*)^2 + (\Delta a^*)^2 + (\Delta b^*)^2 \right]^{1/2} \\ \Delta E^*_{uv} &= \left[(\Delta L^*)^2 + (\Delta u^*)^2 + (\Delta v^*)^2 \right]^{1/2}\end{aligned}\quad (8)$$

where the differences are given between the corresponding color coordinates in the CIELAB and CIELUV spaces between the standard and test samples.

Many improvements to this basic color difference measure have been proposed and adopted over the years involving scaling the lightness, chroma, and hue differences appropriately based on the application and the dataset of samples to which the color difference measure has been adapted or improved [7]. Typically, color difference thresholds are dependent on application and thresholds for *perceptibility* judgments are significantly lower than thresholds for *acceptability* judgments. These color spaces were designed for threshold color differences and their application to supra-threshold (larger than about 5 units of ΔE) color differences is to be handled with care [17]. Many other color difference measures have been proposed and more recently, the CIE DE2000 has been adopted as a measure of color difference, again for uniform color patches under highly controlled viewing conditions, and is slowly gaining acceptance [7, 15].

These color spaces provide a powerful tool to model and quantify color stimuli and are used both in color difference modeling for color patches and, as well, have more recently been used to describe color appearance (see [19]). Models based on describing colors based on lightness, chroma, and hue are powerful in their abilities to enable communication of color stimuli, as well.

Color spaces for colorimetric analysis have the advantage that the need for communicating these colors to others can be done without much ambiguity, if the viewing conditions (reference white) are specified and well controlled. Arguably, the biggest disadvantage is that it is not straightforward to understand what a certain color coordinate in a certain color space would mean to an observer without having to use mathematical equations.

Color Spaces for Editing

Color spaces for editing are specifically designed with the following key requirements:

- Colors have perceptual correlates that are easily understood, such as hue, saturation, brightness, purity, etc.
- Colors described in these spaces are reproducible – within reason – across different media: monitors, printers, etc.
- Colors in these spaces are defined based on primaries – the axes:
 - On the display side of the world of color applications, these spaces are additive, defined by red, green, and blue primaries: Equal amounts of red and green will give yellow.
 - On the printing side, primaries are defined roughly by cyan, magenta, yellow, and black, and these spaces are subtractive: Knowing the print surface, equal densities of cyan and yellow will give green.

A Brief Note on “Gamma”

If a color system is linear, then for additive combinations of colors, a unit input of a color corresponds to a globally constant unit of the output signal; whereas for nonlinear combinations, a transfer function would determine the input–output relationship. This nonlinearity is often used in digital systems with limited available bits and in signal compression systems to take advantage of the available output signal bit depth by stretching small input codes over a larger range of output codes and compressing the larger input codes into a smaller output dynamic range. This is referred to as *gamma encoding*.

From an encoding perspective, in its simplest form, the input–output relationship is typically given by a gain-offset-gamma model, given by:

$$y = \text{round} \left[\left(2^N - 1 \right) \left(\alpha x + \beta \right)^\gamma \right] \quad (9)$$

where α denotes a scalar gain, N denotes the number of bits in a system, β is a scalar offset, γ denotes a power law with values larger than 1 (typically around 2.2), and x and y denote the normalized input and output signals respectively [1]. In encoding systems, the three channels typically have the same parameters α , β , γ , N . Display systems based on cathode ray tubes (CRTs) have an inherent response that follows the inverse relationship – large steps in input signal at the low end of the input signal cause a small change

in output whereas at the upper end of the signal range, small steps caused large output changes. It so happens that gamma encoding (using a power law of γ on the linear luminance input) the input prior to transmitting the data to a CRT display causes the display luminance to follow similar steps resulting in a net unity transfer function. This is also a useful means of encoding data to maximize bit-depth usage while reducing visibly apparent contouring on the output data and display [10, 18]. In the case of a quantized color space, for reasons of perceptual uniformity, it is preferable to establish a nonlinear relationship between color values and intensity or luminance.

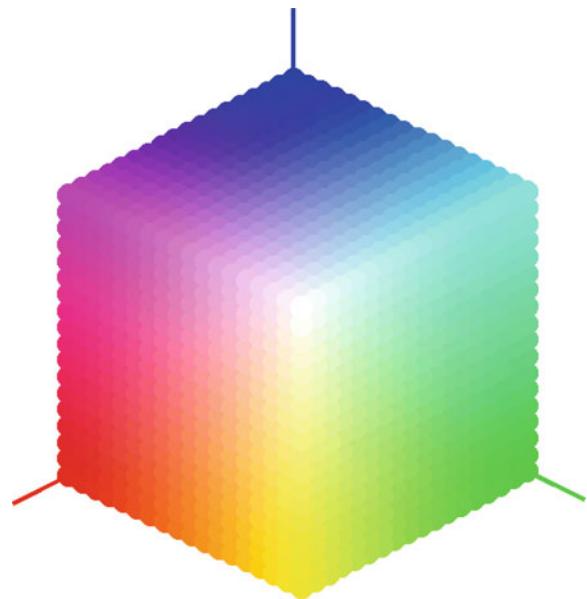
RGB Color Model

The specification of a color in the RGB color space implies a linear (after gamma is removed) or nonlinear (gamma applied) combination of the red, green, and blue primaries in varying strengths. In RGB color spaces, the manner in which colors are reproduced varies from device to device. For example, a color specified as an RGB triplet is more than likely going to look different from one display device to another when the exact same RGB triplet is provided as input, due to differences in the “color” of the primaries and also differences in gamma curves. This makes the RGB space a device-dependent color space. Specifying colors in device-dependent color spaces, although not preferred from a color-reproduction perspective, is often resorted to due to its ease in comprehension. An RGB color model can represent any color within an RGB color cube, as shown in Fig. 4. This color model is most commonly used in display applications where data is additive in nature.

For example, a full-strength yellow color is specified by (1.0, 1.0, 0.0), denoting the use of the red and green primaries at full strength and the blue primary completely turned off. In an 8-bit system, this will correspond to a code value of (255,255,0). A three-primary display with three independent color primaries (typically denoted by their CIE x,y chromaticity values along with that of white) is specified by:

$$\begin{bmatrix} x_R & x_G & x_B & x_W \\ y_R & y_G & y_B & y_W \end{bmatrix}. \quad (10)$$

An introduction to how specifications such as those in (Eq. 10) may be used to generate a transformation



Color Spaces, Fig. 4 RGB color cube commonly used in display applications

matrix from RGB to XYZ for linear RGB is given in the book [18].

Different RGB color spaces differ in their gamma values and the chromaticity coordinates in (Eq. 10). A useful compilation of various RGB color spaces may be found in a Web site hosted by Bruce Lindbloom [11].

A color mixing matrix for additive colors mixing is shown in Table 1, stating, for example, that a cyan color would be created using maximum intensity of green and blue primaries and none of the red primary.

CMY/CMYK Color Model

Printers, on the other hand, create colors using inks that are deposited on paper, in which case the manner in which they create color is called subtractive color mixing. The inks selectively absorb wavelengths of incident light and reflect the remainder. As a beam of light passes through an absorbing medium, the amount of light absorbed is proportional to the intensity of the incident light times the coefficient of absorption (at a given wavelength). This is often referred to as *Beer-Lambert-Bouguer law* and is given by:

$$A(\lambda) = \log_{10} \varepsilon(\lambda)c(\lambda)l(\lambda) \quad (11)$$

**Color Spaces, Table 1** Color mixing matrix for additive primaries

| Primary used | Color displayed | | | | | | | |
|--------------|-----------------|-------|------|------|--------|---------|-------|-------|
| | Red | Green | Blue | Cyan | Yellow | Magenta | White | Black |
| Red | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| Green | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| Blue | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |

where $\varepsilon(\lambda)$ denotes absorptivity, $c(\lambda)$ denotes the concentration, and $l(\lambda)$ denotes the path length for the beam of light. Stated differently, the higher the concentration or thickness or absorptivity of a certain absorptive material, the higher is absorption – the intensity of reflected or transmitted beam of light will be reduced [20]. The simplest model for printer inks is called the *block-dye model*, an idealized system where, for example, cyan ink absorbs all the red but none of the green or blue, with rectangular shapes for absorptivity as a function of wavelength.

In a subtractive-color setup, different thicknesses of the three primary inks may be deposited on top of each other to result in a final color to the observer. The colorant amounts required to print a stimulus designated by RGB emissions are given by $Y = 1 - X$, where $Y \in \{C, M, Y\}$ and $X \in \{R, G, B\}$, all normalized to unity. Real primary inks, however, do not correspond to these ideal functions and, hence, more sophisticated models need to include not just the spectral absorptions/reflectances of the inks, but the density (or area) of the inks and the characteristics of the media (paper) involved. The Kubelka-Munk equations describe the absorption and scattering of light as it passes through layers of ink and the substrate, for example, paper. Various extensions are used in practice that account for the shortcomings of the basic Kubelka-Munk analysis, considering issues such as nonlinearities in ink deposition, interactions between inks, etc., [8, 20]. In subtractive color mixing, the primaries are typically cyan (C), yellow (Y), and magenta (M). The color mixing matrix for subtractive color is shown in Table 2.

Much like RGB color systems, where the reference white made a difference in the appearance of a certain color, depending upon the kind of paper and inks used for printing, the rendered color can be significantly different from one printer to another.

Most printers use a “K” channel, denoting black ink, primarily because a black generated by mixing cyan, yellow, and magenta is not black enough in appearance. Additionally, to complicate matters, in order to print black, a printer would need to lay cyan,

magenta, and yellow inks on top of each other, making ink drying a cause for concern and additionally the limits of ink absorption by the substrate, for example, paper. Additionally, using 1 unit of black ink instead of 1 unit each of cyan, yellow, and magenta inks can lead to significant cost savings.

HSL/HSV Color Model

In order to make the representation of colors intuitive, colors may be ordered along three independent dimensions corresponding to the perceptual correlates of lightness, hue, and chroma. In device-dependent color spaces, there are many commonly used variants of these perceptual correlates: HSV is by far the most common. H stands for the perceptual correlate of hue; S stands for the saturation of a color, defined by the chroma of a color divided by its luminance (the more desaturated the color the closer it is to gray); and V stands for value (a perceptual correlate of lightness). This color model is commonly used in image processing and editing software. However, the HSV color model has two visualization representations, one of which is a cylinder with black at the bottom and pure full-intensity colors on the top, and the other is a representation by a cone, with black at the apex and white on the base. The equations used to convert RGB data into the HSV color space are given by:

$$V = \max \quad (12)$$

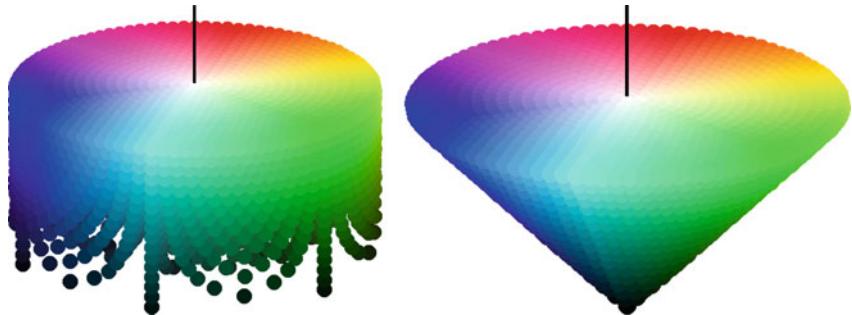
$$S = \begin{cases} 0 & \text{if } V = 0 \\ ((V - \min)/V) & \text{if } V > 0 \end{cases} \quad (13)$$

$$H = \begin{cases} 0 & \text{if } S = 0 \\ 60(G - B)/(max - min) & \text{if } (max = R \text{ and } G \geq B) \\ 60(G - B)/(max - min) + 360 & \text{if } (max = R \text{ and } G < B) \\ 60(B - R)/(max - min) + 120 & \text{if } max = G \\ 60(R - G)/(max - min) + 240 & \text{if } max = B \end{cases} \quad (14)$$

where max and min denote the maximum and minimum of the (R, G, B) triplet. These two representations are shown in Fig. 5. From the figure, it is apparent that saturation is not dependent on the

Color Spaces, Table 2 Color mixing matrix for subtractive primaries

| Primary used | Color displayed | | | | | | | |
|--------------|-----------------|-------|------|------|--------|---------|-------|-------|
| | Red | Green | Blue | Cyan | Yellow | Magenta | White | Black |
| Cyan | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| Yellow | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| Magenta | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |

Color Spaces, Fig. 5 The HSV color model represented as a cylinder and as a cone

intensity of the signal. It is, however, often useful in image processing applications to have an indicator of saturation given by a function of the intensity of the signal, resulting in a conical-shaped HSV color space (Fig. 5). When the conical representation is preferred, S is given by $(\max - \min)/(2^N - 1)$ where $2^N - 1$ denotes the largest possible value for R, G, or B. Other variants of the HSV color space also exist and are used as an intuitive link to RGB color spaces (HSB, or HLS denoting various correlates of hue, saturation, and brightness/lightness).

Other Color Spaces

Color spaces designed for editing and communication needs are typically formulated such that colors are encoded/transmitted in the color space of a reference device. Colors spaces fitting such a description include the sRGB color space, the YCC, YUV, YIQ color transmission spaces, the SWOP CMYK color space, Adobe RGB, and ProPhoto RGB, to list a few.

A popular mechanism to standardize colors across electronic devices such as printers, monitors, and the Internet is the use of the sRGB color space. Originally, this was proposed by Hewlett-Packard and Microsoft, and was later standardized by the International Electrotechnical Commission under IEC 61966-2-1 [12]. There are two primary parts to the sRGB standard: the viewing conditions and the necessary colorimetric definitions and transformations. The sRGB reference viewing environment corresponds to

conditions typical of monitor display viewing conditions and thus may not be as well suited for print material, due to the various proprietary gamut mapping algorithms in most printers that take advantage of each printer's color gamut. The colorimetric definitions provide the transforms necessary to convert between the sRGB color space and the CIEXYZ tristimulus color space as defined for a standard 2° observer. More specifically, the standard is written for a standard reference monitor that has Rec. 709 primaries and a D65 white point. An overview of the technical advantages and challenges of the sRGB color space may be found in Refs. [21, 26]. As was mentioned earlier, color spaces for video directly make use of the gamma-corrected signals, denoted R' , G' , B' , from camcorders, without any attempt to correlate to the linear signals used in color science, such as those in (Eq. 1). For still imaging as well as video, this problem can be mitigated by the use of the transform built into the sRGB standard, which includes a function for transforming from nonlinear signals I' to linear ones. On a scale of 0.0–1.0, for each of $I = R, G, B$, the following function is applied:

$$\begin{cases} I = I'/12.92, & \text{if } I' < 0.04045; \\ I = ((I' + 0.055)/1.055)^{2.4} & \text{otherwise.} \end{cases} \quad (15)$$

In the video industry, a common mode of communication is the YCbCr color space (YPbPr in the



analog domain) that converts RGB signal information into an opponent luma-chroma color space. A nonlinear transfer function is applied to linear-light R, G, B values and a weighted sum of the resulting R', G', B' values is used in the Y, Cb, and Cr signals. In the television domain, these signals have dynamic ranges (on an 8-bit scale) of 16–235 for the luma signal and 16–240 in the Cb and Cr signals. This is to allow for signal noise and potential signal processing noise, giving some head- and foot-room. The weights are different depending upon the color space that the data is being created for. For example, encoding R', G', B' signals with a 16–235 dynamic range into a color space defined by the NTSC primaries (often referred to as ITU-R BT.601), is given by:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} \quad (16)$$

whereas when using HDTV (referred to as ITU-R BT.709) primaries, is given by:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.213 & 0.715 & 0.072 \\ -0.117 & -0.394 & 0.511 \\ 0.511 & -0.464 & -0.047 \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}. \quad (17)$$

A useful reference for further details on this topic is a book by Keith Jack [14]. The Y channel typically contains most of the information in the image, as defined by spatial frequencies, and is hence sampled at much higher rates than the chroma signals. This greatly helps in the ability of the transmission system to compress luma and chroma data with low overheads when compared to luma-only systems. To aid compression formats, color images in the JPEG and JPEG2000 file formats also convert the R', G', B' information into the YCbCr color space prior to compression.

In the printing industry, a commonly specified color space is the SWOP (Specifications for Web Offset Publications) CMYK color space. The SWOP CMYK [2] is a proofing specification that has a well-established relationship between the CMYK input to a standard printer and its CIELAB values (an approximation of the perceptual coordinates of a color) and for a standardized dataset. Specifying images in the SWOP CMYK color space allows the printing house and the content creator to preview images on a common

baseline prior to printing. Most image editing software available nowadays allows the user to preview images in the SWOP CMYK color space.

Open Problems

Depending upon the application, color spaces have their individual, optimized uses. Device-independent color models like the CIEXYZ, CIELAB, CIELUV, and their derivatives are used most often to communicate color either between devices or between different color processing teams across the world. The International Color Consortium (ICC) has been extremely successful in standardizing device-independent color spaces between displays, printers, and capture devices [9, 13]. The color profiles that are stored and communicated in ICC profiles use an intermediate profile connection space (PCS) such as CIEXYZ or CIELAB. ICC profiles also store color transformation profiles to and from different color devices (say, from an input device such as a scanner to CIEXYZ, and from CIEXYZ to an output device such as a printer). For example, an sRGB ICC profile incorporates the color space transform from sRGB to the PCS and a SWOP CMYK ICC profile would incorporate the color space transform from the PCS to the CMYK output color space for a printer. Furthermore, depending upon the rendering intent (how the colors need to be represented on the output device), different transformations may be specified in the ICC profile. Interested readers are referred to more detailed discussions of this subject such as the comprehensive books by Hunt [10], BillMeyer and Saltzman [1], Kuehni [16], Fairchild [19], Sharma [23], and Green and MacDonald [8].

References

1. Berns RS (2000) Billmeyer and Saltzman's principles of color technology, 3rd edn. Wiley, New York
2. CGATS TR 001 (1995) Graphic technology – color characterization data for type 1 printing. American National Standards Institute, Washington, DC
3. Commission Internationale de l'Eclairage (1926) The basis of physical photometry, CIE proceedings 1924. Cambridge University Press, Cambridge
4. Commission Internationale de l'Eclairage (1931) Proceedings international congress on illumination. Cambridge University Press, Cambridge

5. Commission Internationale de l'Eclairage (1964) CIE proceedings (1964) Vienna session, committee report E-1.4.1. Bureau Central de la CIE, Paris
6. Commission Internationale de l'Eclairage (1986) CIE publication 15.2, colorimetry. Central Bureau CIE, Vienna
7. Commission Internationale de l'Eclairage (2004) Colorimetry, publication CIE 15:2004, 3rd edn. CIE, Vienna
8. Green P, MacDonald L (eds) (2002) Colour engineering: achieving device independent colour. Wiley, Chichester
9. <http://www.color.org>. International Color Consortium
10. Hunt RWG (2004) The reproduction of color, 6th edn. Wiley, Chichester/Hoboken
11. Information About RGB Working Spaces. <http://www.brucelindbloom.com/WorkingSpaceInfo.html>
12. International Electrotechnical Commission (1999) IEC 61966-2-1: multimedia systems and equipment – colour measurement and management – Part 2-1: colour management – default RGB colour space – sRGB. IEC, Geneva
13. International Organization for Standardization (2005) ISO 15076-1, image technology colour management – Architecture, profile format and data structure – Part 1:Based on ICC.1:2004-10, ISO, Geneva
14. Jack K (2001) Video demystified – a handbook for the digital engineer, 3rd edn. LLH Technology Publishing, Eagle Rock
15. Kuehni RG (2002) CIEDE2000, milestone or a final answer? Color Res Appl 27(2):126–127
16. Kuehni RG (2003) Color space and its divisions: color order from antiquity to the present. Wiley, Hoboken
17. Kuehni RG (2004) Color: an introduction to practice and principles, 2nd edn. Wiley, Hoboken
18. Li Z-N, Drew MS (2004) Fundamentals of multimedia. Prentice-Hall, Upper Saddle River
19. Fairchild MD (2005) Color appearance models, 3rd edn. <http://www.wiley.com/WileyCDA/WileyTitle/productCd-1119967031.html>
20. McDonald R (1997) Colour physics for industry, 2nd edn. Society of Dyers and Colourists, Bradford
21. Microsoft Corporation (2001) Colorspace interchange using srgb. <http://www.microsoft.com/whdc/device/display/color/sRGB.mspx>
22. Nassau K (1983) The physics and chemistry of color: the fifteen causes of color. Wiley, New York
23. Sharma G (ed) (2003) Digital color imaging handbook. CRC, Boca Raton
24. Speranskaya NI (1959) Determination of spectrum color coordinates for twenty-seven normal observers. Opt Spectrosc 7:424–428
25. Stiles WS, Burch JM (1959) Npl colour-matching investigation: final report (1958). Opt Acta 6:1–26
26. Stokes M, Anderson M, Chandrasekar S, Motta R (1996) A standard default color space for the internet: sRGB. <http://www.color.org/sRGB.html>
27. Wyszecki G, Stiles WS (2000) Color science: concepts and methods, quantitative data and formulae, 2nd edn. Wiley, New York

Color Specification Systems

► Color Model

Compressed Sensing

► Compressive Sensing

Compressive Sensing

Aswin C. Sankaranarayanan¹ and Richard G. Baraniuk²

¹ECE Department, Rice University, Houston, TX, USA

²Department of Electrical and Computer Engineering, Rice University 2028 Duncan Hal, Houston, TX, USA

Synonyms

Compressed sensing

Related Concepts

► Dimensionality Reduction

Definition

Compressed sensing refers to parsimonious sensing, recovery, and processing of signals under a sparse prior.

Background

The design of conventional sensors is based heavily on the Shannon-Nyquist sampling theorem which states that a signal x band limited to W Hz is determined completely by its discrete time samples provided the sampling rate is greater than $2W$ samples per second. This theorem is at the heart of modern signal processing as it enables signal processing in the discrete time or digital domain without any loss of information. However, for many applications, the Nyquist sampling rate is high as well as redundant and unnecessary. As a motivating example, in modern cameras, the high resolution of the CCD sensor reflects the large amount of data sensed to capture an image. A 10 megapixel camera, in effect, takes 10 million linear measurements

of the scene. Yet, almost immediately after capture, redundancies in the image are exploited to compress the image significantly, often by compression ratios of 100:1 for visualization and even higher for detection and classification tasks. This suggests immense wastage in the overall design of the conventional camera.

Compressive sensing (CS) refers to a sampling paradigm where additional structure on the signal is exploited to enable sub-Nyquist sampling rates. The structure most commonly associated with CS is that of signal sparsity in a transform basis. As an example, the basis behind most image compression algorithms is that images are sparse (or close to sparse) in transform bases such as wavelets and DCT. In such a scenario, a CS camera takes *under-sampled* linear measurements of the scene. Given these measurements, the image of the scene is recovered by searching for the image that is sparsest in the transform basis (wavelets or DCT) while simultaneously satisfying the measurements. This search procedure can be shown to be convex. Much of CS literature revolves around the design of linear measurement matrices, characterizing the number of measurements required and the design of image/signal recovery algorithms.

Theory

Compressive sensing [1–3] enables reconstruction of sparse signals from under-sampled linear measurements. A vector \mathbf{s} is termed K sparse if it has at most K nonzero components, or equivalently, if $\|\mathbf{s}\|_0 \leq K$, where $\|\cdot\|_0$ is the ℓ_0 norm or the number of nonzero components. Consider a signal (e.g., an image or a video) $\mathbf{x} \in \mathbb{R}^N$, which is sparse in a basis Ψ , that is, $\mathbf{s} \in \mathbb{R}^N$, defined as $\mathbf{s} = \Psi^T \mathbf{x}$, is sparse. Examples of the sparsifying basis Ψ for images include DCT and wavelets. The main problem of interest is that of sensing the signal \mathbf{x} from linear measurements. With no additional knowledge about \mathbf{x} , N linear measurements of \mathbf{x} are required to form an invertible linear system. In a conventional digital camera, an identity sensing matrix is used so that each pixel is sensed directly. For sensing reflectance fields, optimal linear sensing matrices have been designed [4].

The theory of compressive sensing shows that it is possible to reconstruct \mathbf{x} from M linear measurements even when $M \ll N$ by exploiting the sparsity

of $\mathbf{s} = \Psi^T \mathbf{x}$. Consider a measurement vector $\mathbf{y} \in \mathbb{R}^M$ obtained using an $M \times N$ measurement matrix Φ , such that

$$\mathbf{y} = \Phi \mathbf{x} + e = \Phi \Psi \mathbf{s} + e = \Theta \mathbf{s} + e, \quad (1)$$

where e is the measurement noise (see Fig. 1) and $\Theta = \Phi \Psi$. The components of the measurement vector \mathbf{y} are called the *compressive measurements* or compressive samples. For $M < N$, estimating \mathbf{x} from the linear measurements is an ill-conditioned problem. However, when \mathbf{x} is K sparse in the basis Ψ , then CS enables recovery of \mathbf{s} (or alternatively, \mathbf{x}) from $M = O(K \log(N/K))$ measurements, for certain classes of matrices Θ . The guarantees on the recovery of signals extend to the case when \mathbf{s} is not exactly sparse but compressible. A signal is termed compressible if its sorted transform coefficients decay according to power law, that is, the sorted coefficient of \mathbf{s} decay rapidly in magnitude [5].

Restricted Isometry Property (RIP)

The condition for stable recovery for both sparse and compressible signals is that the matrix Θ satisfies the following property. Given S -sparse vector \mathbf{s} , $\exists \delta_S$, $0 < \delta_S < 1$ such that

$$(1 - \delta_S) \|\mathbf{s}\|_2 \leq \|\Theta \mathbf{s}\|_2 \leq (1 + \delta) \|\mathbf{s}\|_2. \quad (2)$$

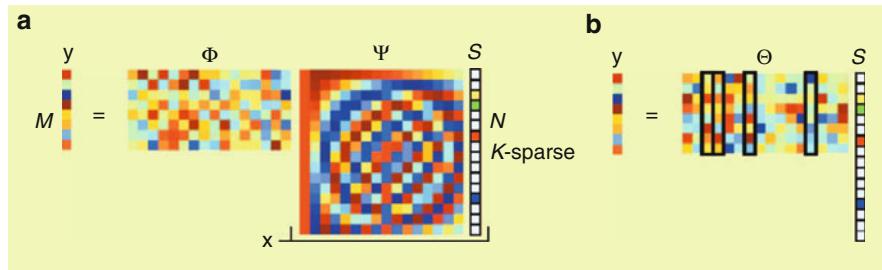
Stable recovery is guaranteed when (Eq. 2) is satisfied for $S = 2K$. This is referred to as the *Restricted Isometry Property* (RIP) [2, 7]. In particular, when Ψ is a fixed basis, it can be shown that using a randomly generated sub-Gaussian measurement matrix Φ , ensures that Θ satisfies RIP with a high probability provided $M = O(K \log(N/K))$. Typical choices for Φ (or equivalently, Θ) are matrices whose entries are independently generated using the Radamacher or the sub-Gaussian distribution.

Signal Recovery

Estimating K -sparse vectors that satisfy the measurement equation of (Eq. 1) can be formulated as the following ℓ_0 optimization problem:

$$(P0) : \min \|\mathbf{s}\|_0 \text{ s.t. } \|\mathbf{y} - \Phi \Psi \mathbf{s}\|_2 \leq \epsilon, \quad (3)$$

with ϵ being a bound for the measurement noise e in (Eq. 1). While this is a NP-hard problem in general, the equivalence between ℓ_0 and ℓ_1 norm for such



Compressive Sensing, Fig. 1 (a) Compressive sensing measurement process with a random Gaussian measurement matrix Φ and discrete cosine transform (DCT) matrix Ψ . The vector of coefficients s is sparse with $K = 4$. (b) Measurement process

systems [8] allows us to reformulate the problem as one of ℓ_1 norm minimization.

$$(P1) : \hat{s} = \arg \min \|s\|_1 \text{ s.t. } \|y - \Phi \Psi s\| \leq \epsilon. \quad (4)$$

It can be shown that, when $M = O(K \log(N/K))$, the solution to the $(P1) - \hat{s}$ – is, with overwhelming probability, the K -sparse solution to $(P0)$. In particular, the estimation error can be bounded as follows:

$$\|s - \hat{s}\|_2 \leq C_0 \|s - s_K\| / \sqrt{K} + c_1 \epsilon, \quad (5)$$

where s_K is the best K -sparse approximation of s .

There exist a wide range of algorithms that solve $(P1)$ to various approximations or reformulations [9, 10]. One class of algorithms model $(P1)$ as a convex problem and recast it as a linear program (LP) or a second order cone program (SOCP) for which there exist efficient numerical techniques. Another class of algorithms employ greedy methods [11] which can potentially incorporate other problem-specific properties such as structured supports [12].

Hardware Implementations

Compressive sensing has been successfully applied to sense various visual signals such as images [13, 14], videos [15] and light transport matrices [16]. The single-pixel camera (SPC) [13] for CS of images is designed as follows. The SPC consists of a lens that focuses the scene onto a digital micro-mirror device (DMD), which takes the place of the CCD array in

with $y = \Phi x$. There are four columns that correspond to nonzero s_i coefficients; the measurement vector y is a linear combination of these columns (Figure from [6])

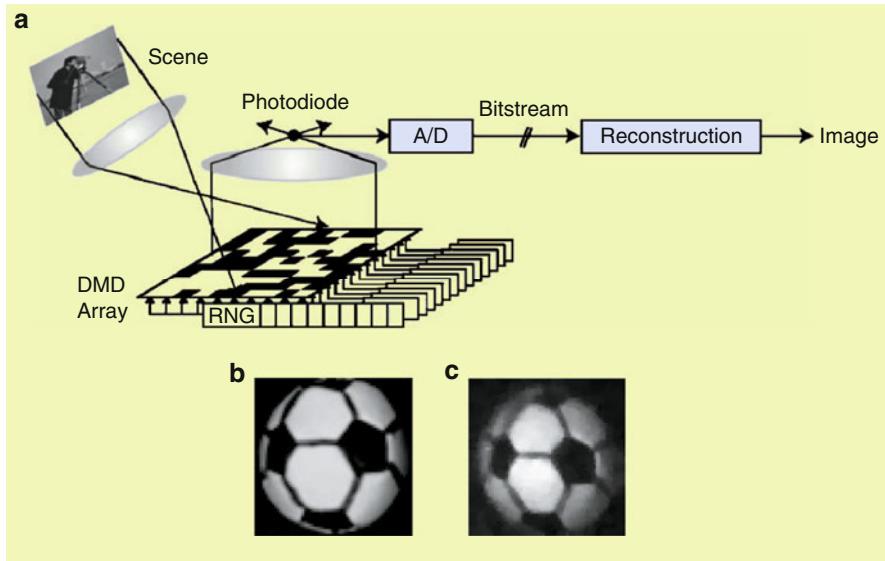
a conventional camera. Each micro-mirror can be oriented in one of two possible angles. Micro-mirrors that are in the same angle/direction are focused onto a photodetector (hence, the single pixel) using a second lens (see Fig. 2). Each measurement that is obtained corresponds to an inner product of the scene with a set of 1s and 0s corresponding to the state of the micro-mirrors. By flipping the micro-mirrors randomly according to a Bernoulli distribution, multiple compressive measurements are obtained.

The SPC is most useful when sensing is costly. One such example is in sensing at wavelengths beyond the visible spectrum, where exotic and costly materials are needed to detect light. For such applications, the cost of building a focal plane array of even VGA resolution is prohibitive. However, the SPC needs only a single photodiode tuned to the spectrum of interest, with otherwise minimal changes to the underlying architecture.

Implications for Computer Vision

Novel Sensors

Compressive sensing enables a new sensing design that measures random linear projection of a scene and subsequently reconstructs the image of the scene using these measurements. The reconstructed image is no different from one captured from a suitably placed camera, barring reconstruction artifacts. However, compressive cameras enable parsimony in sensing – which can be extremely useful in problems where sensing is costly. This could be due to storage,



Compressive Sensing, Fig. 2 (a) Single-pixel, compressive sensing camera. (b) Conventional digital camera image of a soccer ball. (c) 64×64 black-and-white image of the same ball

($N = 4,096$ pixels) recovered from $M = 1,600$ random measurements taken by the camera in (a). The images in (b) and (c) are not meant to be aligned (Figure from [6])

as in *the data deluge* problem in large sensor networks. Compressive sensing is extremely valuable in time sensitive applications such as medical imaging, sensing and modeling fluid dispersions, time varying reflectance fields, and in general, high speed imaging. Finally, sensing could be costly due to adverse affects caused by the process of sensing. In electron microscopy, there are fundamental limitations on the number of images of a thin layer of tissue that comes as the tissue is progressively destroyed in the process of imaging. In all such applications, where sensing is costly in some manner, CS offers better trade-offs over traditional linear sensing.

Tailoring Vision Algorithms for Compressive Data

In many cases, it is beneficial to work on the compressive measurements directly without reconstructing the images. As an example, in background subtraction, the silhouettes are spatially sparse and in many applications far sparser than the original images (in a transform basis). Given a static background image \mathbf{x}_s , silhouettes are recovered by identifying $(\mathbf{y} - \Phi\mathbf{x}_s)$ as compressive measurements of the silhouettes [17]. This can lead to silhouette recovery at extremely high compression ratios. An other example is in video CS for dynamic textures, where the linear dynamical

system parameters of the scene can be recovered by a suitably designed acquisition device [15].

Beyond Sparse Models

While the theory of CS relies on assumptions of sparsity or compressibility of the signal in a transform basis, it can be extended to signals that exhibit additional structures. One such idea is that of model-based CS [12] where in the signal exhibits sparsity in a space of models which encompasses models such as block sparsity, union of subspaces, and wavelet tree models. Another idea is in the use of non-sparse models such as Markov Random fields [18] for CS – with the eventual goal of using the Ising model for sensing images and background subtracted silhouettes. Such progress hints at interesting avenues for future research governed by the use of rich models in existing vision literature for the task of sensing.

Machine Learning and Signal Processing on Compressive Data

Restricted Isometry Property (see (Eq. 2)) for random matrices implies that, for K -sparse signals, distances are approximately preserved under random projections. A host of machine learning and signal processing algorithms depend on pairwise distances of points as

opposed to their exact location. For such algorithms, almost identical results can be obtained by applying them to randomly projected data as opposed to the data in the original space [19, 20]. This has tremendous advantages as the random projected data lies on a much lower dimensional space, and can be directly obtained from a compressive imager. Such ideas have been applied for detection and classification of signals in compressed domain [21].

References

1. Candès E, Romberg J, Tao T (2006) Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Trans Inf Theory* 52(2): 489–509
2. Candès E, Tao T (2006) Near optimal signal recovery from random projections: universal encoding strategies? *IEEE Trans Inf Theory* 52(12):5406–5425
3. Donoho D (2006) Compressed sensing. *IEEE Trans Inf Theory* 52(4):1289–1306
4. Schechner Y, Nayar S, Belhumeur P (2003) A theory of multiplexed illumination. International conference on computer vision, Nice, France, pp 808–815
5. Haupt J, Nowak R (2006) Signal reconstruction from noisy random projections. *IEEE Trans Inf Theory* 52(9):4036–4048
6. Baraniuk R (2007) Compressive sensing. *IEEE Signal Process Mag.* 24(4):118–120, 124
7. Davenport M, Laska J, Boufounos P, Baraniuk R (2009) A simple proof that random matrices are democratic. Technical report TREE 0906, Rice University, ECE Department
8. Donoho D (2006) For most large underdetermined systems of linear equations, the minimal ℓ_1 -norm solution is also the sparsest solution. *Comm Pure Appl Math* 59(6):797–829
9. Candès E, Tao T (2005) Decoding by linear programming. *IEEE Trans Inf Theory* 51(12):4203–4215
10. Tibshirani R (1996) Regression shrinkage and selection via the lasso. *J R Stat Soc B* 58(1):267–288
11. Needell D, Tropp J (2009) CoSaMP: iterative signal recovery from incomplete and inaccurate samples. *Appl Comput Harmon Anal* 26(3):301–321
12. Baraniuk R, Cevher V, Duarte M, Hegde C (2010) Model-based compressive sensing. *IEEE Trans Inf Theory* 56(4):1982–2001
13. Duarte M, Davenport M, Takhar D, Laska J, Sun T, Kelly K, Baraniuk R (2008) Single-pixel imaging via compressive sampling. *IEEE Signal Process Mag* 25(2):83–91
14. Wagadarikar A, John R, Willett R, Brady D (2008) Single disperser design for coded aperture snapshot spectral imaging. *Appl Opt* 47(10):B44–B51
15. Sankaranarayanan AC, Turaga P, Baraniuk R, Chellappa R (2010) Compressive acquisition of dynamic scenes. In: ECCV, Heraklion, Greece, pp 129–142
16. Peers P, Mahajan D, Lamond B, Ghosh A, Matusik W, Ramamoorthi R, Debevec P (2009) Compressive light transport sensing. *ACM Trans Graph* 28(1):1–3
17. Cevher V, Sankaranarayanan A, Duarte M, Reddy D, Baraniuk R, Chellappa R (2008) Compressive sensing for background subtraction. In: Proceedings of the European conference on computer Vision (ECCV), Marseille, France
18. Cevher V, Duarte M, Hegde C, Baraniuk R (2008) Sparse signal recovery using markov random fields. *Neural Info Proc Sys*, Vancouver, BC, Canada, pp 257–264
19. Davenport M, Boufounos P, Wakin M, Baraniuk R (2010) Signal processing with compressive measurements. *IEEE J Sel Top Signal Process* 4(2):445–460
20. Hegde C, Wakin M, Baraniuk R (2007) Random projections for manifold learning. Vancouver, BC
21. Davenport M, Duarte M, Wakin M, Laska J, Takhar D, Kelly K, Baraniuk R (2007) The smashed filter for compressive classification and target recognition, San Jose, CA

Computational Symmetry

Yanxi Liu

Computer Science and Engineering, Penn State University, University Park, PA, USA

Synonyms

[Symmetry-based X](#); [Symmetry detection](#)

Definition

Computational symmetry is a branch of research using computers to model, analyze, synthesize, and manipulate symmetries in digital forms, imagery, or otherwise [1].

Background

Symmetry is a pervasive phenomenon presenting itself in all forms and scales, from galaxies to microscopic biological structures, in nature and man-made environments. Much of one's understanding of the world is based on the perception and recognition of recurring patterns that are generalized by the mathematical concept of symmetries [2–4]. Humans and animals have an innate ability to perceive and take advantage of symmetry in everyday life [5–8], while harnessing this powerful insight for machine intelligence remains a challenging task for computer scientists.

Interested readers can find several influential symmetry-related papers below to gain a historic



perspective: the wonderful exposition on the role of symmetry in “Biological Shape and Visual Science” by Blum in 1973 [9]; in 1977, the “Description and Recognition of Curved Objects” reported by Nevatia and Binford, where bilateral symmetry of an object about different axes is examined [10]; the method of detecting angle/side regularities of closed curves and plateaus in one-dimensional patterns by Davis, Blumenthal, and Rosenfeld [11, 12]; the introduction of the term *skewed symmetry* by Takeo Kanade in 1981 [13]; the exposition on “Smoothed Local Symmetries and Their Implementation” by Brady and Asada [14]; the theory of *recognition-by-components* (RBC) proposed by Biederman in 1985 [15]; “Perceptual Grouping and the Natural Representation of Natural Form” using superquadrics as restricted *generalized cylinders* (GC) by Pentland in 1986 [16]; “Perceptual Organization and Visual Recognition” by Lowe [17], where the non-coincidental appearance of symmetry in the real world was noted; and the “Symmetry-Seeking Models for 3D Object Reconstruction” (1987) illustrated by Terzopoulos, Witkin, and Kass [18].

A computational model for symmetry is especially pertinent to computer vision, computer graphics, and machine intelligence in general because symmetry is:

- *Ubiquitous*: Both the physical and digital worlds are filled with various forms of symmetry, near-symmetry, and distorted symmetry patterns.
- *Essential*: Intelligent beings perceive and interact with the chaotic real world in the most efficient and effective manner by capturing its essential structures and substructures – the generators of symmetry, near-symmetry, distorted symmetry, and/or recurring patterns.
- *Compact*: The recognition of symmetries is the first step towards minimizing redundancy, often leading to drastic reductions in computation.
- *Aestho-physiological*: From a butterfly to an elephant, from a tea cup to a building, symmetry or deviation from it has been a time-honored principle for design (by nature or by human) that can guide machine perception, detection, recognition, and synthesis of the real world.

Figure 1 from [19] shows the statistics of published papers during the period of 1974–2009 (36 years) in several major computer vision/graphics conferences and journals. An increasing level of interests in computational symmetry, and a dominant role reflection

symmetry played in the past several decades, can be observed in both computer vision and computer graphics literature.

Theory

From the spirit of Felix Klein’s Erlangen program [20] that described geometry as the study of a space that is invariant under a given transformation group to the Gestalt principles of perception [21], symmetries and group theory play an important role in describing the geometry and appearance of an object. Informally, one may think of symmetry as expressing the notion that a figure or an object is made from multiple copies of smaller unit that are interchangeable. Mathematically, this notion is formalized by examining the effect of transformations on the object in a certain space such that its subparts map to each other.

More formally, in a metric space M , a *symmetry* $g \in G$ of a set $S \subseteq M$ is an isometry (a distance preserving transformation) that maps S to itself (an automorphism), $g(S) = S$. The transformation g keeps S invariant as a whole while permuting its parts. Symmetries G of S form a mathematical *group* $\{G, *\}$, closed under transformation composition $*$, called the *symmetry group* of S [3].

Basic Concepts

The basic definitions of symmetry and group theory can be found in most general mathematic textbooks on modern algebra. In particular the following books are recommended: *Geometry* by Coxeter [22], *Generators and relations for discrete groups* by Coxeter and Moser [3], *Symmetry* by Weyl [2], and *Tilings and Patterns* by Grünbaum and Shephard [23].

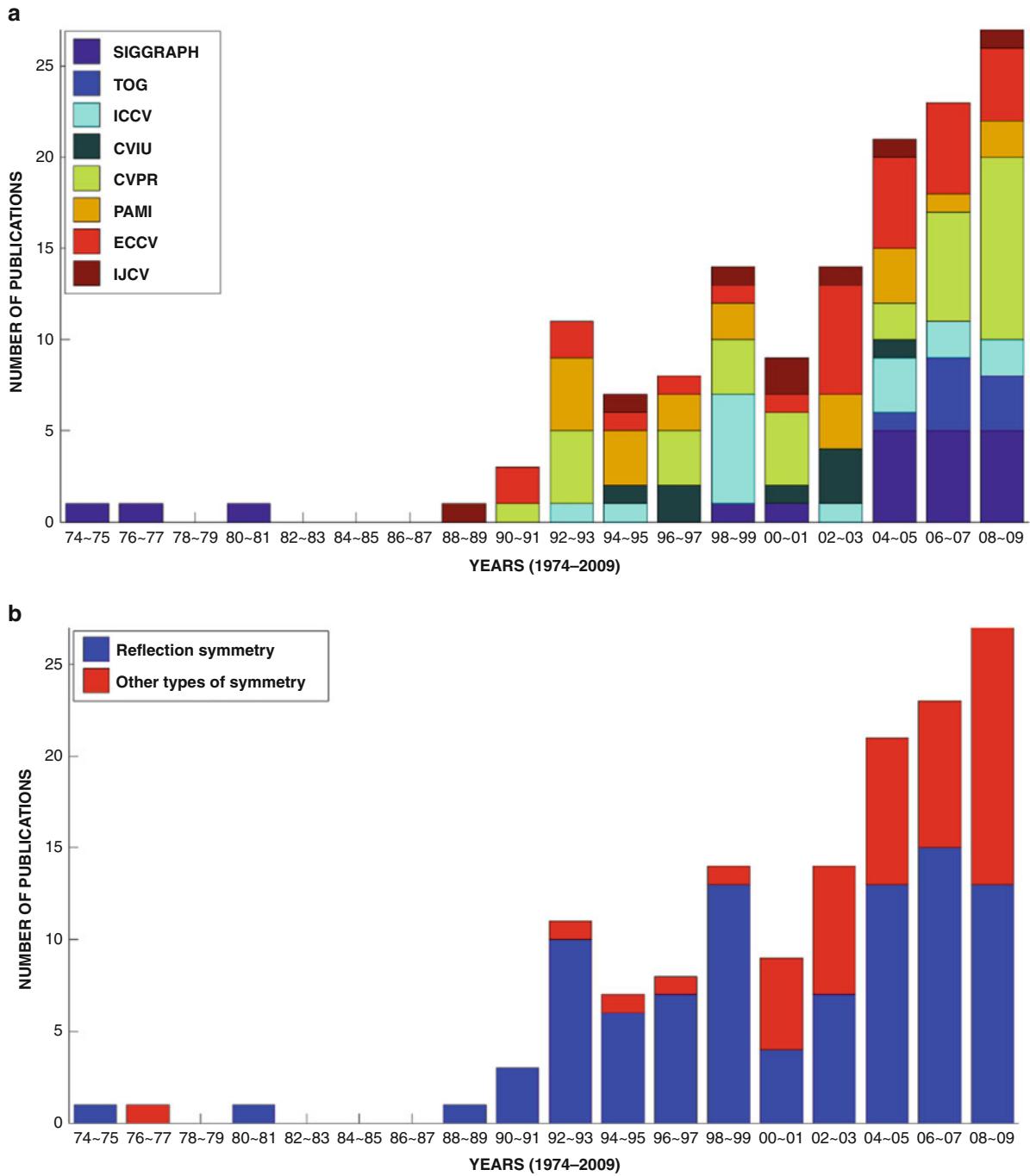
Some key concepts and computationally relevant definitions [19] are provided below:

Definition 1 Let S be a subset of R^n . Then an isometry (a distance preserving mapping) g is a symmetry of S if and only if $g(S) = S$.

Definition 2 A symmetry g for a set $S \in R^n$ is a primitive symmetry if and only if (1) $g \neq e$ where e is an identity mapping; and (2) if $g = g_1g_2$, $g_1 \neq e$, $g_2 \neq e$, then neither g_1 nor g_2 is a symmetry of S .

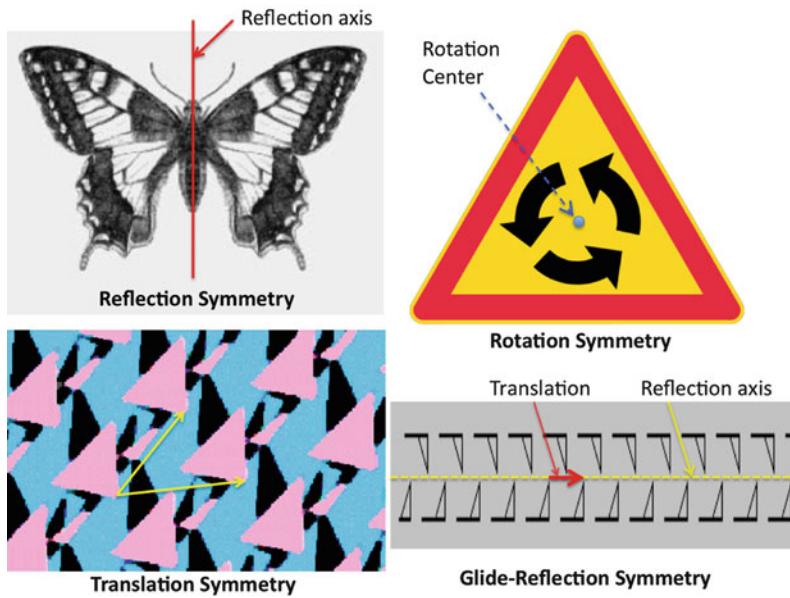
For example, in 2D Euclidean space R^2 , there are four types [2–4] of primitive symmetries $g(S) = S$.





Computational Symmetry, Fig. 1 From the publication statistics (data collected in [19]), it is obvious that (1) there is an increasing trend of interests in computational symmetry and (2) research on reflection symmetry has been dominating the field in the past several decades, though there is a recent growing awareness of the whole symmetry spectrum. A similar reflection-symmetry-dominating trend has also been

observed in the psychology literature for human perception of symmetries [8] (a) Publications on symmetry detection and applications in major computer vision and computer graphics conferences/journals (b) Dividing papers into reflection symmetry alone versus other types of symmetries (rotation, translation, and glide reflection)



Computational Symmetry, Fig. 2 An illustration of patterns with the four different primitive symmetries, respectively, in 2D Euclidean space. Reflection symmetry has the reflection axis as its point-wise invariance. Rotation symmetry has the *center* of rotation as its invariant point. An n -fold rotational symmetry with respect to a particular point (in 2D) or axis (in 3D)

means that, setwise, a rotation by an angle of $360/n$ does not change the object. Glide reflection is composed of a translation that is $1/2$ of the smallest translation symmetry t and a reflection r with respect to a reflection axis along the direction of the translation. There are no invariant points under translation and glide-reflection symmetries

They are, without loss of generality, for the four images $f(x, y)$ shown in Fig. 2:

1. *Reflection*: $f(x, y) = f(-x, y)$; its reflection axis (plane) remains invariant under the reflection.
 2. *Rotation*: $f(x, y) = f(r \cos(2\pi/n), r \sin(2\pi/n))$, $r = \sqrt{x^2 + y^2}$, n is an integer ($n = 3$ in the top right of Fig. 2), and its rotation center point (axis) remains invariant under the rotation.
 3. *Translation*: $f(x, y) = f(x + \Delta x, y + \Delta y)$, for some $\Delta x, \Delta y \in R$, no invariant points exist.
 4. *Glide Reflection*: $f(x, y) = f(x + \Delta x, -y)$, for some $\Delta x \in R$, no invariant points exist. A glide reflection g can be expressed as $g = tr$, where t is a translation and r is a reflection whose axis of reflection is along the direction of the translation.
- Note: neither t nor r alone is a symmetry of S ; thus, g is a primitive symmetry of S .

Definition 3 Let G be a nonempty set with a well-defined binary operation $*$ such that for each ordered pair $g_1, g_2 \in G$, $g_1 * g_2$ is also in G . $(G, *)$ is a **group** if and only if:

1. There exists an *identity element* $e \in G$ such that $e * g = g = g * e$ for all $g \in G$.

2. Any element g in G has an *inverse* $g^{-1} \in G$ such that $g * g^{-1} = g^{-1} * g = e$.
3. The binary operation $*$ is associative: $a * (b * c) = (a * b) * c$ for all $a, b, c \in G$.

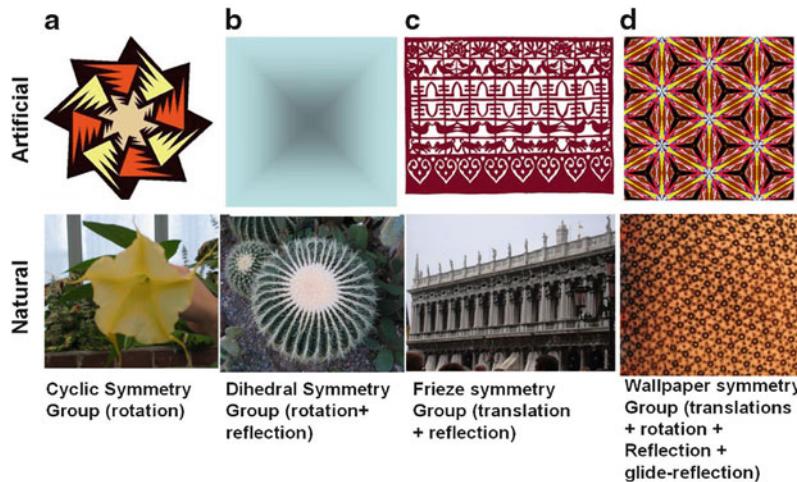
Using the composition of transformations on R^n as the binary operation $*$, one can prove that *symmetries of a subset $S \subset R^n$ form a group*, which is called the *symmetry group* of S . Figure 3 and Table 1 illustrate the four distinct types of symmetry groups in R^2 .

Definition 4 All the symmetries of R^n form the **Euclidean group** \mathcal{E} .

Proposition 5 Symmetries of a subset $S \subset R^n$ form a symmetry group G_S of S where $G_S \subset \mathcal{E}$ (see a proof in [24]).

Definition 6 A **point group** G is a symmetry group that leaves a single point fixed. In other words, $G = G_x$, here G_x is the stabilizer subgroup on x .

Definition 7 A **space group** is a group G of operations which leaves the infinitely extended, regularly repeating lattice pattern $L \subset R^n$ invariant while for all nontrivial $g \in G$, i.e., $g \neq e$, and $x \in R^n$, $g(x) \neq x$.



Computational Symmetry, Fig. 3 Sample images of symmetry categorized by their respective ideal symmetry groups: column (a) cyclic group containing rotation symmetries only, (b) dihedral group (reflection and rotation), (c) frieze group (translation plus reflection and rotation), and (d) wallpaper group

(translation, rotation, reflection, and glide-reflection). The *top row* contains synthetic patterns, while the *bottom row* shows photos of real-world scenes (*bottom right* is an image of a transverse slice of skeletal muscle magnified with a transmitter microscope 800,000 times)

Computational Symmetry, Table 1 Discrete symmetry groups in R^2 (See Fig. 3 for sample images)

| Name | Group type | Symbol | Order | Primitive symmetry | Example |
|-----------|------------|------------------------|----------|--------------------------------------|-------------------------|
| Cyclic | Point | C_n | n | Rotation | Fig. 3a |
| Dihedral | Point | D_n | $2n$ | Rotation and reflection | Fig. 3b |
| Frieze | Space | G_{frieze} | ∞ | All four 2D-primitive symmetry types | Fig. 3c |
| Wallpaper | Space | $G_{\text{wallpaper}}$ | ∞ | All four 2D-primitive symmetry types | Fig. 3d |

Crystallographic Groups

An n -dimensional periodic pattern is formed by repeating a unit pattern in equal intervals along $k \leq n$ directions. A mature mathematical theory for symmetries of periodic patterns in n -dimensional Euclidean space has been known for over a century [25–28], namely, the *crystallographic groups* [2, 22, 23]. An important mathematical discovery is the answer to the first part of the Hilbert’s 18th problem [29]: regardless of dimension n and despite an infinite number of possible instantiations of periodic patterns, the number of distinct symmetry groups for periodic patterns in any Euclidean space R^n is always *finite*. For 2D monochrome patterns, there are 7 *frieze-symmetry groups* translating along one direction (strip patterns) [22] (Fig. 3c) and 17 *wallpaper groups* covering the whole plane [23, 25] (Fig. 3d). In 3D, there are 230 different *space groups* [30] generated by three linearly independent translations (regular crystal patterns).

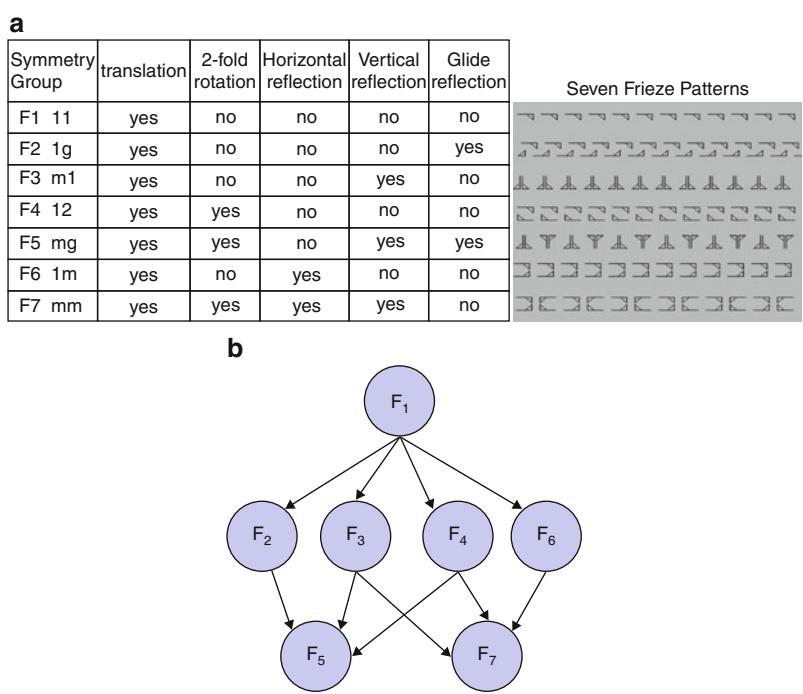
Frieze Symmetry Group

A frieze pattern is a 2D strip in the plane that is periodic along one dimension. Any frieze pattern P is associated with one of the seven unique symmetry groups (Fig. 4). These seven symmetry groups, denoted by crystallographers as $l1$, lg , ml , $l2$, mg , lm , and mm [22], are called the *frieze groups*. Without loss of generality, assuming the direction of translation symmetry of a frieze pattern is horizontal, the frieze pattern can then exhibit five different types of symmetries (Fig. 4a):

1. Horizontal translation
2. Two-fold rotation
3. Horizontal reflection (reflection axis is placed horizontally)
4. Vertical reflection
5. Horizontal glide reflection composed of a half-unit translation followed by a horizontal reflection

The primitive symmetries in each group (the inner structure of a frieze group) and the relationship

Computational Symmetry, Fig. 4 The five types of primitive symmetries (left table of (a)) and the inner (right of (a)) and inter-structures (b) of the seven frieze groups



among the seven frieze groups (inter-structure of frieze groups) are depicted in Fig. 4a, b, respectively. Each frieze pattern is associated with one of the seven possible frieze groups, subject to the combination of these five primitive symmetries in the pattern (Fig. 4a).

Wallpaper Symmetry Group

A wallpaper pattern is a 2D periodic pattern extending along two linearly independent directions [22, 31] (Fig. 5). Any wallpaper pattern is associated with one of the 17 wallpaper groups. Wallpaper group theory [23] states that all translationally symmetric patterns P_r can be generated by a pair of linearly independent, shortest (among all possible) vectors t_1, t_2 applied to a minimum tile. The *orbits* of this pair of translation symmetry generator vectors form a 2D *quadrilateral lattice*, which simultaneously defines all 2D tiles (partitions the space into its smallest generating regions by its translation subgroup), and a topological lattice structure relating all tiles (Fig. 5a).

Motifs of Wallpaper Patterns

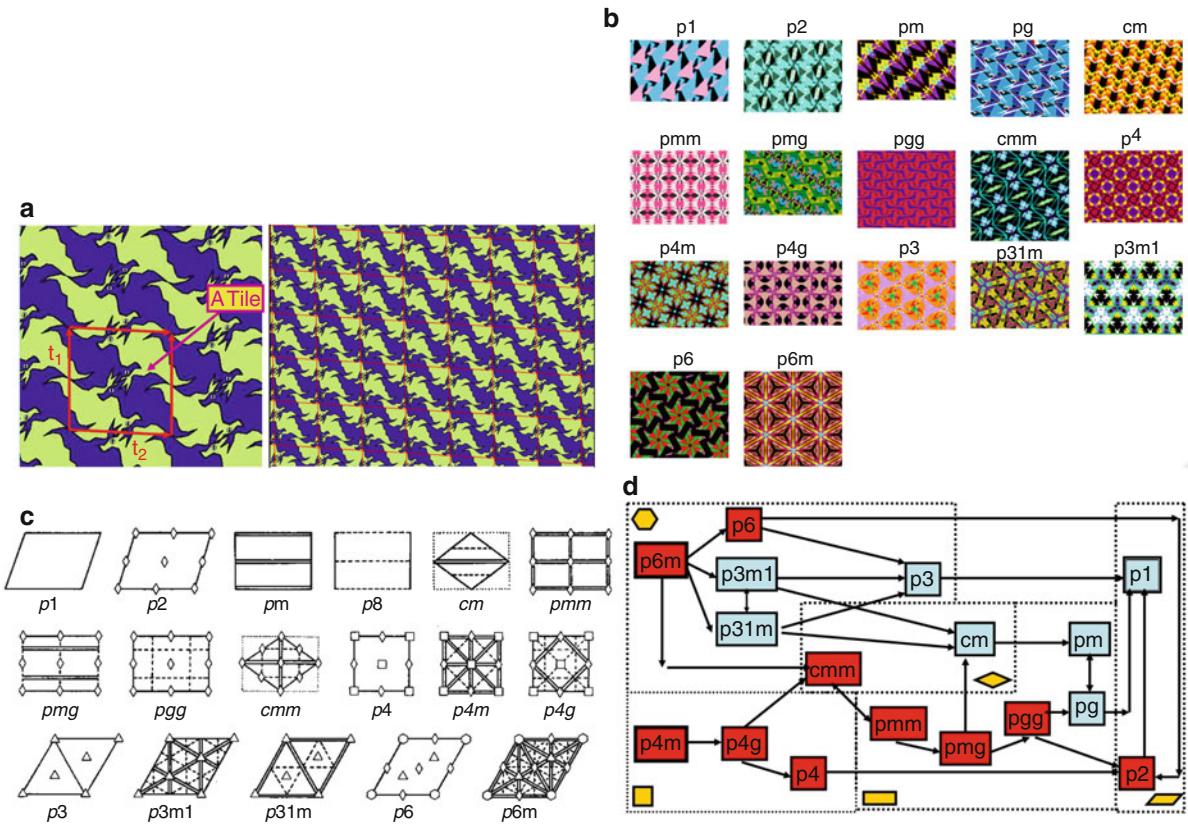
When the translational subgroup of a periodic pattern is determined, it fixes the size, shape, and orientation of the unit lattice, but leaves open the question of where the unit lattice should be anchored in the

pattern. Any parallelogram of the same size and shape carves out an equally good tile that can be used to tile the plane. However, from a perception point of view, some parallelograms produce tiles that are better descriptors of the underlying symmetry of the overall pattern than others. For example, if the whole pattern has some rotation symmetries, a tile located on the rotation centers can represent the global symmetry property of the wallpaper pattern instantly (Fig. 5a). Such *motifs*, as representative tiles of a periodic pattern, can be defined mathematically (and discovered computationally [32, 33]):

Definition 8 A motif of a wallpaper pattern is a tile that is cut out by a lattice whose lattice unit is centered on the fixed point of the largest stabilizer group.

Application

Automatic detection of 2D primitive symmetries (reflection, rotation, translation, glide reflection) from 2D images has been a standing topic in computer vision. The earliest attempt at an algorithmic treatment of bilateral reflection symmetry detection predates computer vision itself [34]. Interested readers can find a long list of published symmetry detection



Computational Symmetry, Fig. 5 (a) A *tile* and a 2D lattice (red) determined simultaneously by the generators t_1, t_2 of the translation subgroup. (b) Sample wallpaper patterns associated with the 17 distinct wallpaper groups. The inner (c) and inter-structures (d) of the 17 wallpaper groups are shown. (c) The symbols $p1, p2, pm, pg, cm, pmm, pmg, pgg, cmm, p4, p4m, p4g, p3, p31m, p3m1, p6$, and $p6m$ are crystallographers’ representations for the wallpaper groups; the diagram is

courtesy of [31]. The *diamond*, *triangle*, *square*, and *hexagon shapes* correspond to two-, three-, four-, and six-fold rotation centers. *Solid single line* and *dotted single line* and *double parallel lines* denote unit translation, glide-reflection, and reflection symmetries, respectively. (d) The subgroup hierarchy, where $A \rightarrow B$ means that B is a subgroup of A (information extracted from [22])

algorithms in [19]. The first quantitative benchmark on reflection, rotation, and translation symmetry detection algorithms can be found in [35, 36].

The first general-purpose, automatic frieze and wallpaper group classification algorithm on real images is published in [32, 33], followed by the classification of *skewed symmetry groups*, in particular wallpaper groups and rotation symmetry groups from affinely and projectively distorted images [37, 38]. More recently, aiming at capturing the translation subgroup of a wallpaper group, several algorithms [39–42] are developed for lattice detection (Fig. 5a) from various locally and/or globally distorted, unsegmented real images.

The applications of computational symmetry are diverse and cross-disciplinary [19]. Some sample

applications in computer vision/graphics include texture replacement in real photos [43], human gait analysis [33, 44], static and dynamic near-regular texture analysis, synthesis, replacement/superimposition and tracking [45–50], image “de-fencing” [51, 52], automatic geotagging [53], fabric categorization [54], architectural facade in-painting [55], multi-view 3D geometry [56], and multi-target tracking [50, 57].

New applications of computational symmetry and group theory attract attentions in urban scene and architectural image analysis, synthesis and 3D-reconstruction [53, 55, 58, 59], multi-target tracking of topology-varying spatial patterns [60], image segmentation and grouping [58, 61–63], transformation estimation [64–66], facial asymmetry as a biometric [67, 68], and an expression and gender cue [69–71].

For biomedical applications, quantified symmetry has also been used to measure symmetry of moving objects such as molecules [72–83], pathological and degenerative volumetric neuroradiology images [84–89], and the firing fields of the grid cells of rats in the context of computational neurosciences [90].

Open Problems

1. Mathematical Symmetries versus Digitized Real-World Data: Well-defined and exhaustively categorized symmetries and symmetry groups become computationally brittle on real-world, noisy, and sometimes distorted digital data.
2. Robust Symmetry Detectors: In spite of years of effort, a robust, widely applicable “symmetry detector” for various types of real-world symmetries remains elusive.
3. Benchmarking: Though an initial effort has been made to quantify and compare the outcome of multiple symmetry detection algorithms [35, 36], the field is yet to see a large-scale, systematic, quantitative evaluation and a publicly available test-image database to gauge the progress in this important, widely applicable research direction.

References

1. Liu Y (2002) Computational symmetry. In: Hargittai I, Laurent T (eds) *Symmetry 2000*. Wenner-Gren international series, vol 80. Portland, London, pp 231–245. ISBN: I 85578 149 2
2. Weyl H (1952) *Symmetry*. Princeton University Press, Princeton
3. Coxeter H, Moser W (1980) Generators and relations for discrete groups, 4th edn. Springer, New York
4. Conway J, Burgiel H, Goodman-Strauss C (2008) *The symmetries of things*. AK Peters, Wellesley
5. Thompson DW (1961) *On growth and form*. Cambridge University Press, Cambridge
6. Giurfa M, Eichmann B, Menzel R (1996) Symmetry perception in an insect. *Nature* 382(6590):458–461
7. Rodríguez I, Gumbert A, Hempel de Ibarra N, Kunze J, Giurfa M (2004) Symmetry is in the eye of the ‘beeholder’: innate preference for bilateral symmetry in flower-naïve bumblebees. *Naturwissenschaften* 91(8):374–377
8. Tyler C (ed) (1996) *Human symmetry perception and its computational analysis*. VSP, Utrecht
9. Blum H (1973) Biological shape and visual science (part i). *J Theor Biol* 38:205–287
10. Nevatia R, Binford T (1977) Description and recognition of curved objects. *Artif Intell* 8(1):77–98
11. Davis L (1977) Understanding shape: angles and sides. *IEEE Trans Comput* 26(3):236–242
12. Blumenthal AF, Davis L, Rosenfeld A (1977) Detecting natural “plateaus” in one-dimensional patterns. *IEEE Trans Comput* 26(2):178–179
13. Kanade T (1981) Recovery of the 3-dimensional shape of an object from a single view. *Artif Intell* 17:75–116
14. Brady M, Asada H (1984) Smoothed local symmetries and their implementation. *Int J Robot Res* 3(3):36–61
15. Biederman I (1985) Human image understanding: recent research and a theory. *Comput Vis Graph Image Process* 32:29–73
16. Pentland A (1986) Perceptual organization and the representation of natural form. *Artif Intell* 28:293–331
17. Lowe D (1985) *Perceptual organization and visual recognition*. Kluwer Academic, Boston
18. Terzopoulos D, Witkin A, Kass M (1987) Symmetry-seeking models and 3D object reconstruction. *Int J Comput Vis* 1:211–221
19. Liu Y, Hel-Or H, Kaplan C, Van Gool L (2010) Computational symmetry in computer vision and computer graphics: a survey. *Found Trends Comput Graph Vis* 5(1/2):1–165
20. Greenberg MJ (1993) *Euclidean and Non-Euclidean geometries: development and history*, 3rd edn. WH Freeman, New York
21. Arnheim R (2004) *Art and visual perception: a psychology of the creative eye*. University of California Press, Berkeley/London
22. Coxeter H (1980) *Introduction to geometry*, 2nd edn. Wiley, New York
23. Grünbaum B, Shephard G (1987) *Tilings and patterns*. WH Freeman, New York
24. Liu Y (1990) Symmetry groups in robotic assembly planning. PhD thesis, University of Massachusetts, Amherst
25. Fedorov E (1885) The elements of the study of figures. (Russian) (2) 21. Zapiski Imperatorskogo S. Peterburgskogo Mineralogichesko Obshchestva (Proc. S. Peterb. Mineral. Soc.), 1–289
26. Fedorov E (1891) Symmetry of finite figures. (Russian) (2) 28. Zapiski Imperatorskogo S. Peterburgskogo Mineralogichesko Obshchestva (Proc. S. Peterb. Mineral. Soc.), 1–146
27. Fedorov E (1891) Symmetry in the plane. (Russian) (2) 28. Zapiski Imperatorskogo S. Peterburgskogo Mineralogichesko Obshchestva (Proc. S. Peterb. Mineral. Soc.), 345–390
28. Bieberbach L (1910) Über die Bewegungsgruppen der n-dimensionalen Euklidischen Räume mit einem endlichen Fundamentalbereich. *Göttinger Nachrichten* 75–84
29. Milnor J (1976) Hilbert’s problem 18. In: *Proceedings of symposia in pure mathematics*, vol 28. American Mathematical Society, Providence. ISBN:0-8218-1428-1 (Browder FE, Mathematical developments arising from Hilbert problems)
30. Henry N, Lonsdale K (eds) (1969) *International tables for X-ray crystallography. Symmetry groups*, vol 1. The Kynoch, England/The International Union of Crystallography, Birmingham
31. Schattschneider D (1978) The plane symmetry groups: their recognition and notation. *Am Math Mon* 85:439–450
32. Liu Y, Collins RT (2000) A computational model for repeated pattern perception using frieze and wallpaper

- groups. In: Computer vision and pattern recognition conference (CVPR'00), Hilton Head, SC. IEEE Computer Society, pp 537–544. http://www.ri.cmu.edu/pubs/pub_3302.html
33. Liu Y, Collins R, Tsin Y (2004) A computational model for periodic pattern perception based on frieze and wallpaper groups. *IEEE Trans Pattern Anal Mach Intell* 26(3):354–371
34. Birkoff G (1932) Aesthetic measure. Harvard University Press, Cambridge
35. Chen P, Hays J, Lee S, Park M, Liu Y (2007) A quantitative evaluation of symmetry detection algorithms. Technical report PSU-CSE-07011 (also listed as technical report CMU-RI-TR-07-36), The Pennsylvania State University, State College, PA
36. Park M, Lee S, Chen P, Kashyap S, Butt A, Liu Y (2008) Performance evaluation of state-of-the-art discrete symmetry detection algorithms. In: IEEE conference on computer vision and pattern recognition (CVPR 2008), Anchorage, pp 1–8
37. Liu Y, Collins RT (2001) Skewed symmetry groups. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition (CVPR'01), Kauai, HI. IEEE Computer Society, pp 872–879. http://www.ri.cmu.edu/pubs/pub_3815.html
38. Lee S, Liu Y (2010) Skewed rotation symmetry group detection. *IEEE Trans Pattern Anal Mach Intell* 32(9):1659–1672
39. Tuytelaars T, Turina A, Van Gool L (2003) Non-combinatorial detection of regular repetitions under perspective skew. *IEEE Trans Pattern Anal Mach Intell* 25(4):418–432
40. Hays J, Leordeanu M, Efros A, Liu Y (2006) Discovering texture regularity as a higher-order correspondence problem. In: European conference on computer vision (ECCV'06), Graz, Austria
41. Park M, Liu Y, Collins R (2008) Deformed lattice detection via mean-shift belief propagation. In: Proceedings of the 10th European conference on computer vision (ECCV'08), Marseille, France
42. Park M, Brocklehurst K, Collins R, Liu Y (2009) Deformed lattice detection in real-world images using mean-shift belief propagation. *IEEE Trans Pattern Anal Mach Intell* 31(10):1804–1816
43. Tsin Y, Liu Y, Ramesh V (2001) Texture replacement in real images. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition (CVPR'01), Kauai. IEEE Computer Society, Los Alamitos, pp 539–544
44. Liu Y, Collins R, Tsin Y (2002) Gait sequence analysis using frieze patterns. In: Proceedings of the 7th European conference on computer vision (ECCV'02), a longer version can be found as CMU RI tech report 01–38 (2001), Copenhagen, Denmark
45. Liu Y, Tsin Y (2002) The promise and perils of near-regular texture. In: Texture 2002, Copenhagen, Denmark, in conjunction with European conference on computer vision (ECCV'02), pp 657–671
46. Liu Y, Lin W (2003) Deformable texture: the irregular-regular-irregular cycle. In: Texture 2003, the 3rd international workshop on texture analysis and synthesis, Nice, France, pp 65–70
47. Liu Y, Lin W, Hays J (2004) Near-regular texture analysis and manipulation. *ACM Trans Graph* 23(3):368–376
48. Liu Y, Tsin Y, Lin W (2005) The promise and perils of near-regular texture. *Int J Comput Vis* 62(1-2):145–159
49. Lin W, Liu Y (2006) Tracking dynamic near-regular textures under occlusion and rapid movements. In: Proceedings of the 9th European conference on computer vision (ECCV'06), Graz, Austria, vol 2, pp 44–55
50. Lin W, Liu Y (2007) A lattice-based mrf model for dynamic near-regular texture tracking. *IEEE Trans Pattern Anal Mach Intell* 29(5):777–792
51. Liu Y, Belkina T, Hays H, Lublinerman R (2008) Image defencing. In: IEEE computer vision and pattern recognition (CVPR 2008), Anchorage, pp 1–8
52. Park M, Brocklehurst K, Collins R, Liu Y (2010) Image defencing revisited. In: Asian conference on computer vision (ACCV'10), Queenstown. IEEE Computer Society, pp 1–13
53. Schindler G, Krishnamurthy P, Lublinerman R, Liu Y, Dellaert F (2008) Detecting and matching repeated patterns for automatic geo-tagging in urban environments. In: IEEE computer vision and pattern recognition (CVPR 2008), Anchorage, pp 1–8
54. Han J, McKenna S, Wang R (2008) Regular texture analysis as statistical model selection. In: ECCV08, Marseille
55. Korah T, Rasmussen D (2008) Analysis of building textures for reconstructing partially occluded facades. In: European conference on computer vision (ECCV08), Marseille, pp 359–372
56. Hong W, Yang AY, Ma Y (2004) On symmetry and multiple view geometry: structure, pose and calibration from a single image. *Int J Comput Vis* 60(3):241–265
57. Park M, Liu Y, Collins R (2008) Efficient mean shift belief propagation for vision tracking. In: Proceedings of computer vision and pattern recognition conference (CVPR'08), Anchorage. IEEE Computer Society
58. Park M, Brocklehurst K, Collins R, Liu Y (2010) Translation-symmetry-based perceptual grouping with applications to urban scenes. In: Asian conference on computer vision (ACCV'10), Queenstown. IEEE Computer Society, pp 1–14
59. Wu C, Frahm JM, Pollefeys M (2010) Detecting large repetitive structures with salient boundaries. In: Daniilidis K, Maragos P, Paragios N (eds) European conference on computer vision (ECCV 2010). Lecture notes in computer science, vol 6312. Springer, Berlin/Heidelberg, pp 142–155. doi:10.1007/978-3-642-15552-9_11
60. Liu J, Liu Y (2010) Multi-target tracking of time-varying spatial patterns. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition (CVPR'10), San Francisco. IEEE Computer Society, pp 1–8
61. Sun Y, Bhanu B (2009) Symmetry integrated region-based image segmentation. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition (CVPR'08). IEEE Computer Society, Anchorage, Alaska, pp 826–831
62. Yang A, Rao S, Huang K, Hong W, Ma Y (2003) Geometric segmentation of perspective images based on symmetry groups. In: Proceedings of the 10th IEEE international conference on computer vision (ICCV'03), Nice, vol 2, p 1251
63. Levinstein A, Sminchisescu C, Dickinson S (2009) Multiscale symmetric part detection and grouping. In: ICCV, Kyoto

64. Makadia A, Daniilidis K (2006) Rotation recovery from spherical images without correspondences. *IEEE Trans Pattern Anal Mach Intell* 28:1170–1175
65. Tuzel O, Subbarao R, Meer P (2005) Simultaneous multiple 3D motion estimation via mode finding on lie groups. In: Proceedings of the 10th IEEE international conference on computer vision (ICCV'05), Beijing, vol I, pp 18–25
66. Begelfor E, Werman M (2005) How to put probabilities on homographies. *IEEE Trans Pattern Anal Mach Intell* 27:1666–1670
67. Liu Y, Schmidt K, Cohn J, Weaver R (2002) Facial asymmetry quantification for expression invariant human identification. In: International conference on automatic face and gesture recognition (FG'02), Washington, DC
68. Liu Y, Schmidt K, Cohn J, Mitra S (2003) Facial asymmetry quantification for expression invariant human identification. *Comput Vis Image Underst* J 91(1/2):138–159
69. Liu Y, Palmer J (2003) A quantified study of facial asymmetry in 3D faces. In: IEEE international workshop on analysis and modeling of faces and gestures, IEEE, Nice, pp 222–229
70. Mitra S, Liu Y (2004) Local facial asymmetry for expression classification. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition (CVPR'04). IEEE Computer Society, Washington, DC, pp 889–894. http://www.ri.cmu.edu/pubs/pub_4640.html
71. Mitra S, Lazar N, Liu Y (2007) Understanding the role of facial asymmetry in human face identification. *Stat Comput* 17:57–70
72. Zabrodsky H, Avnir D (1993) Measuring symmetry in structural chemistry. In: Hargittai I (ed) Advanced molecular structure research, vol 1. JAI, Greenwich
73. Zabrodsky H, Avnir D (1995) Continuous symmetry measures, iv: chirality. *J Am Chem Soc* 117:462–473
74. Avnir D, Katzenelson O, Keinan S, Pinsky M, Pinto Y, Salomon Y, Hel-Or H (1997) The measurement of symmetry and chirality: conceptual aspects. In: Rouvray DH (ed) Concepts in chemistry. Research Studies, Somerset, pp 283–324
75. Kanis DR, Wong JS, Marks TJ, Ratner M, Zabrodsky H, Keinan S, Avnir D (1995) Continuous symmetry analysis of hyperpolarizabilities. characterization of second order nonlinear optical response of distorted benzene. *J Phys Chem* 99:11061–11066
76. Yogeve-Einot D, Avnir D (2006) The temperature-dependent optical activity of quartz: from le châtelier to chirality measures. *Tetrahedron Asymmetry* 17:2723–2725
77. Yogeve-Einot D, Avnir D (2004) Pressure and temperature effects on the degree of symmetry and chirality of the molecular building blocks of low quartz. *Acta Crystallogr B60*:163–173
78. Keinan S, Avnir D (2000) Quantitative symmetry in structure-activity correlations: the near $c2$ symmetry of inhibitor/hiv-protease complexes. *J Am Chem Soc* 122:4378–4384
79. Alvarez S, Alemany P, Casanova D, Cirera J, Llunell M, Avnir D (2005) Shape maps and polyhedral interconversion paths in transition metal chemistry. *Coord Chem Rev* 249:1693–1708
80. Pinsky M, Avnir D (1998) Continuous symmetry measures, v: the classical polyhedra. *Inorg Chem* 37:5575–5582
81. Steinberg A, Karni M, Avnir D (2006) Continuous symmetry analysis of NMR chemical shielding anisotropy. *Chem Eur J* 12:8534–8538
82. Pinto Y, Fowler P, Mitchell D, Avnir D (1998) Continuous chirality analysis of model stone-wales rearrangements in fullerenes. *J Phys Chem* 102:5776–5784
83. Keinan S, Avnir D (2001) Continuous symmetry analysis of tetrahedral/planar distortions: copper chlorides and other AB₄ species. *Inorg Chem* 40:318–323
84. Liu Y, Dellaert F (1998) A classification-based similarity metric for 3D image retrieval. In: Proceedings of computer vision and pattern recognition conference (CVPR'98), Santa Barbara. IEEE Computer Society, pp 800–807
85. Liu Y, Collins R, Rothfus W (2001) Robust midsagittal plane extraction from normal and pathological 3D neuro-radiology images. *IEEE Trans Med Imaging* 20(3):175–192
86. Liu Y, Dellaert F, Rothfus W, Moore A, Schneider J, Kanade T (2001) Classification-driven pathological neuroimage retrieval using statistical asymmetry measures. In: International conference on medical imaging computing and computer assisted intervention (MICCAI 2001), Utrecht. Springer, pp 655–665
87. Liu Y, Teverovskiy L, Carmichael O, Kikinis R, Shenton M, Carter C, Stenger V, Davis S, Aizenstein H, Becker J, Lopez O, Meltzer C (2004) Discriminative MR image feature analysis for automatic schizophrenia and alzheimer's disease classification. In: 7th international conference on medical imaging computing and computer assisted intervention (MICCAI 2004), Saint-Malo. Springer, pp 378–385
88. Liu Y, Teverovskiy L, Lopez O, Aizenstein H, Becker J, Meltzer C (2007) Discovery of “biomarkers” for Alzheimer's disease prediction from structural MR images. In: 2002 IEEE international symposium on biomedical imaging: macro to nano, Arlington, pp 1344–1347
89. Teverovskiy L, Becker J, Lopez O, Liu Y (2008) Quantified brain asymmetry for age estimation of normal and AD/MCI subjects. In: 2008 IEEE international symposium on biomedical imaging: nano to macro, Paris, pp 1509–1512
90. Chastain E, Liu Y (2007) Quantified symmetry for entorhinal spatial maps. *Neurocomputing* 70(10–12):1723–1727

Computer Vision

► Rationale for Computational Vision

Computing Architectures for Machine Perception

► High-Performance Computing in Computer Vision

Concept Languages

► Description Logics

Concurrent Mapping and Localization (CML)

- ▶ Exploration: Simultaneous Localization and Mapping (SLAM)

Conditional Random Fields

- ▶ Discriminative Random Fields

Contour Detection

- ▶ Boundary Detection

Convex Minimization

- ▶ Semidefinite Programming

Cook-Torrance BRDF

- ▶ Cook-Torrance Model

Cook-Torrance Model

Abhijeet Ghosh
Institute for Creative Technologies, University of Southern California, Playa Vista, CA, USA

Synonyms

- ▶ Cook-Torrance BRDF

Definition

Cook-Torrance model is an analytic BRDF model that describes the wavelength dependent reflectance property of a surface based on the principles of microfacet theory.

Background

Accurate descriptions of how light reflects off a surface are a fundamental prerequisite for computer vision and graphics applications. Real world materials exhibit characteristic surface reflectance, such as glossy or specular highlights, anisotropy, or off-specular reflection, which need to be modeled for such applications. The surface reflectance of a material is formalized by the notion of the Bidirectional Reflectance Distribution Function (BRDF) [1], which describes the reflected response of a surface in a certain exitant direction to illumination from a certain incident direction over a hemisphere of directions.

Analytical reflection models attempt to describe certain classes of BRDFs using a mathematical representation involving a small number of parameters. The Cook-Torrance model [2] is an analytic isotropic BRDF model that falls under the category of a physics-based model and is based on the microfacet theory of inter-reflection of light at rough surfaces. It extends the Torrance-Sparrow reflectance model, originally developed in the field of applied optics [3], for modeling wavelength dependent effects of reflection. The model predicts both the directional distribution as well as spectral composition of reflected light.

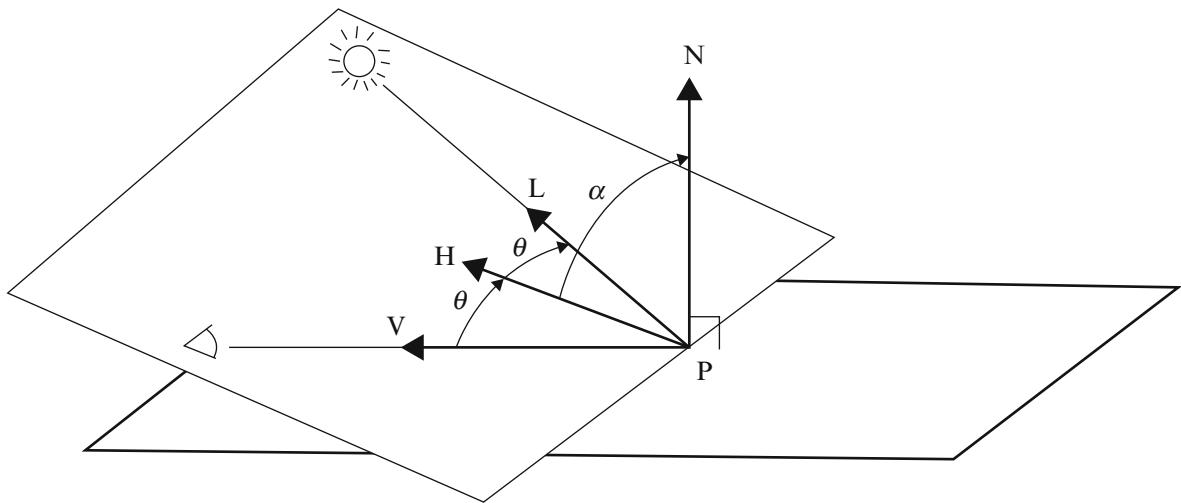
Theory

Given a light source, a surface, and an observer, the Cook-Torrance model describes the intensity and spectral composition of the reflected light reaching the observer. The geometry of reflection is shown in Fig. 1. An observer is looking at a point P on a surface. V is the unit vector in the direction of the viewer, N is the unit normal to the surface, and L is the unit vector in the direction of a specific light source. H is the normalized half-vector between V and L given as

$$H = \frac{L + V}{\|L + V\|}, \quad (1)$$

and is the unit normal of a microfacet that would reflect light specularly from L to V . α is the angle between H and N , and θ is the angle between H and V , so that $\cos \theta = V \cdot H = L \cdot H$.

The main components of the reflectance model are the directionally dependent diffuse and specular



Cook-Torrance Model, Fig. 1 The geometry of reflection [2]

reflection terms, and a directionally independent ambient term. The specular component R_s represents light that is reflected from the surface of the material. The diffuse component R_d originates from internal scattering (in which the incident light penetrates beneath the surface of the material) or from multiple surface reflections (which occur if the surface is sufficiently rough). The directional reflectance is thus given as $R = sR_s + dR_d$, where $s + d = 1$. The model also includes a directionally independent ambient term R_a to approximate the effects of global illumination.

The total intensity of the light reaching the observer is the sum of the reflected intensities from all light sources plus the reflected intensity from any ambient illumination. The basic reflectance model then becomes

$$I_r = I_{ia} R_a + \sum_l (sR_s + dR_d) I_{il} (N \cdot L_l) d\omega_{il}. \quad (2)$$

This formulation accounts for the effect of light sources with different intensities and different projected areas which may illuminate a scene. The next two sub-sections consider the directional and wavelength dependence of the reflectance model.

Directional Distribution

The ambient R_a and diffuse R_d components of the model reflect light independent of the viewing direction. However, the specular component R_s does depend on the viewing direction. The angular spread of

the specular component can be described by assuming that the surface consists of microfacets, each of which reflects specularly [3].

Only facets whose normal is in the direction H contribute to the specular component of reflection from L to V . The specular component is given as

$$R_s = \frac{F}{\pi} \frac{D \cdot G}{(N \cdot L)(N \cdot V)}, \quad (3)$$

where D is the facet slope distribution function, G is the geometric masking and shadowing attenuation term, and F accounts for Fresnel reflectance. The Fresnel term F describes the fraction of light that is reflected from each smooth microfacet. It is a function of incidence angle and wavelength of incident light and is discussed in the next section.

Microfacet Distribution

The facet slope distribution function D represents the fraction of the facets that are oriented in the direction H . Various facet slope distribution functions have been considered by Blinn [4] including the Gaussian model:

$$D = ce^{-(\alpha/m)^2}, \quad (4)$$

where c is the normalization constant.

Beckmann described a model that originated from the study of scattering of radar waves from rough surfaces [5], and is applicable to a wide range of surface conditions ranging from smooth to very rough.

For rough surfaces, the Beckmann distribution is

$$D = \frac{1}{m^2 \cos^4 \alpha} e^{-[(\tan \alpha)/m]^2} \quad (5)$$

In both the facet slope distribution functions, the spread of the specular component depends on the root mean square (rms) slope m . Small values of m signify gentle facet slopes and give a distribution that is highly directional around the specular direction, while large values of m imply steep facet slopes and give a distribution that is spread out with off-specular peak modeled by the Beckmann distribution (see Fig. 2).

For general surfaces with two or more scales of surface roughness, the slope m can be modeled by using a convex weighted combination of two or more distribution functions [6]:

$$D = \sum_j w_j D(m_j), \quad (6)$$

where m_j is the rms slope and w_j the weight of the j th distribution respectively.

Geometric Attenuation

The geometrical attenuation factor G accounts for the shadowing and masking of one facet by another and is discussed in detail in [3, 4]. The following expression is derived for G for microfacets in the shape of v-shaped grooves (see Fig. 3):

$$G = \min \left\{ 1, \frac{2(N \cdot H)(N \cdot V)}{(V \cdot H)}, \frac{2(N \cdot H)(N \cdot L)}{(V \cdot H)} \right\}. \quad (7)$$

Spectral Composition

The ambient, diffuse, and specular reflectances all depend on wavelength. R_a , R_d , and the F term of R_s may be obtained from the appropriate reflectance spectra for the material. Reflectance spectra of thousands of materials can be found in the literature [7–10]. The reflectance data are usually for illumination at normal incidence. These values are normally measured for polished surfaces and must be multiplied by $1/\pi$ to obtain the bidirectional reflectance for a rough surface. Most materials are measured at only a few wavelengths in the visible range (typically around 10–15), so that values for intermediate wavelengths must be interpolated.

The spectral energy distribution of the reflected light is found by multiplying the spectral energy distribution of the incident light by the reflectance spectrum of the surface. An example of this is shown in Fig. 4. The spectral energy distributions of the sun and a number of CIE standard illuminants are available in [11].

Fresnel Reflectance

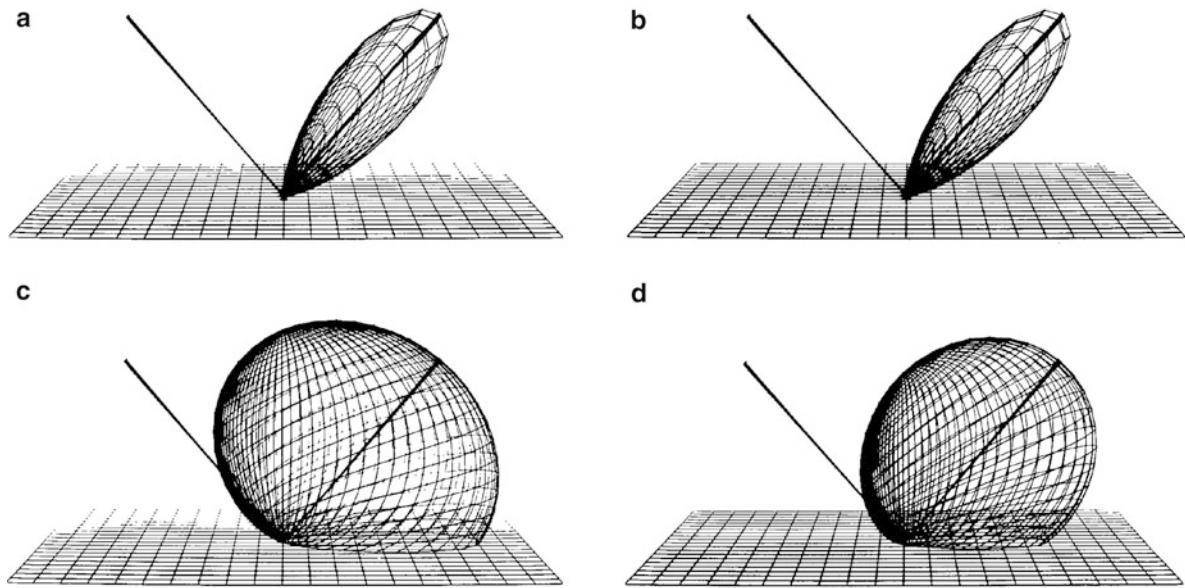
The reflectance F may be obtained theoretically from the Fresnel equation [12]. This equation expresses the reflectance of a perfectly smooth, mirrorlike surface in terms of the index of refraction n (for both metals and dielectrics) and the extinction coefficient k (for metals only) of the surface and the angle of incidence of illumination θ . The Fresnel equation for unpolarized incident light and a dielectric material ($k = 0$) is

$$F = \frac{1}{2} \frac{(g - c)^2}{(g + c)^2} \left\{ 1 + \frac{[c(g + c) - 1]^2}{[c(g - c) + 1]^2} \right\}, \quad (8)$$

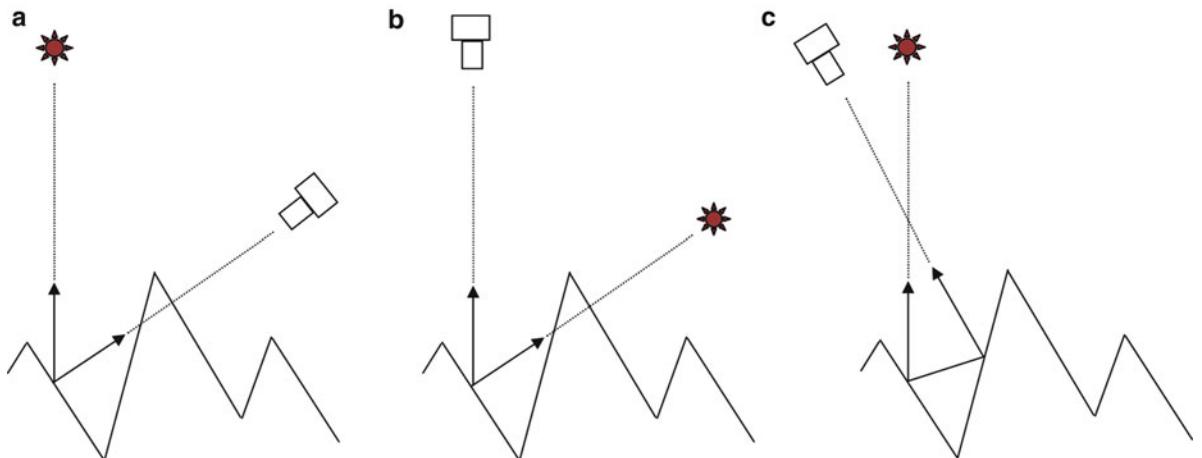
where $c = \cos \theta = V \cdot H$ and $g^2 = n^2 + c^2 - 1$. The dependence of the reflectance on wavelength and the angle of incidence implies that the color of the reflected light changes with the incidence angle, from color of the material at normal incidence to color of the illuminant at grazing incidence.

In general, both n and k vary with wavelength, but their values are frequently not known. On the other hand, experimentally measured values of the reflectance at normal incidence are frequently known. To obtain the spectral and angular variation of F , the following practical approach is adopted: If n and k are known, the Fresnel equation is used. If not, but the normal reflectance is known, the Fresnel equation is fit to the measured normal reflectance for a polished surface. For nonmetals or dielectrics, for which $k = 0$, this immediately gives an estimate of the index of refraction n . For metals, for which k is generally not 0, k is set to 0 to get an effective value for n from the normal reflectance. The angular dependence of F is then available from the Fresnel equation. The above procedure yields the correct value of F for normal incidence and a good estimate of its angular dependence, which is only weakly dependent on the extinction coefficient k .

To illustrate this procedure, consider a dielectric material ($k = 0$) at normal incidence. $\theta = 0$, so $c = 1$,



Cook-Torrance Model, Fig. 2 (a) Beckmann distribution for $m = 0.2$, (b) Gaussian distribution for $m = 0.2$, (c) Beckmann distribution for $m = 0.6$, (d) Gaussian distribution for $m = 0.6$ [2]



Cook-Torrance Model, Fig. 3 Geometric attenuation due to microfacets. (a) Masking. (b) Shadowing. (c) Inter-reflection

$g = n$ and Eq. 8 reduces to

$$F_0 = \frac{(n-1)^2}{(n+1)^2}. \quad (9)$$

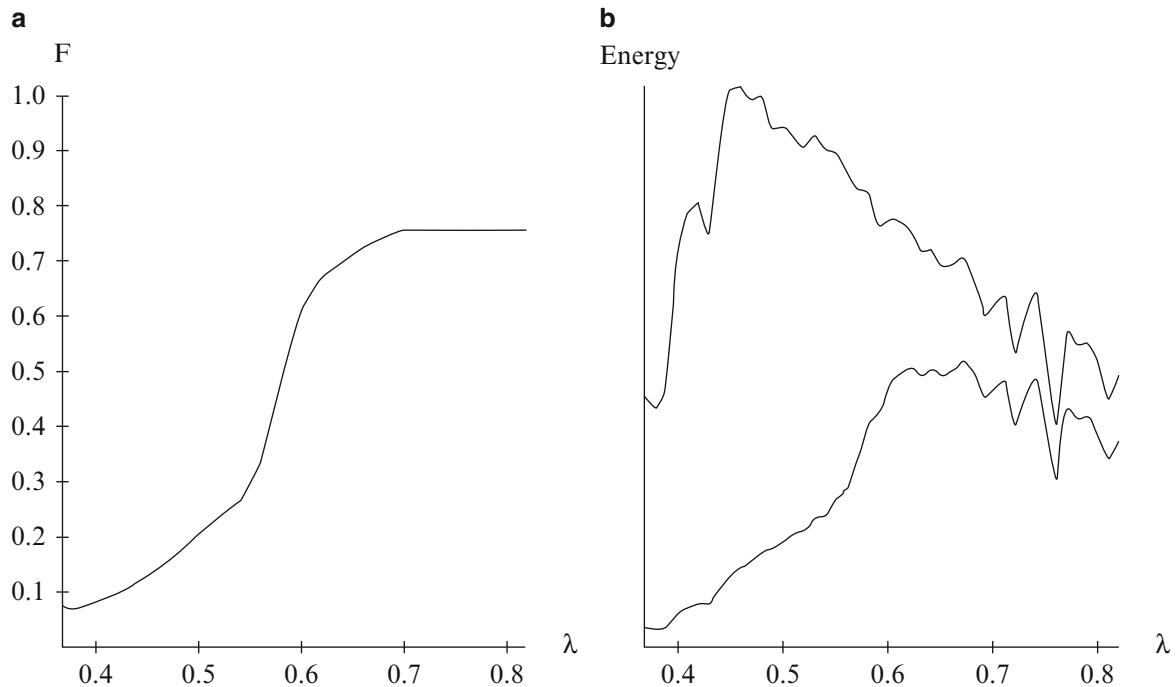
Solving for n gives the equation

$$n = \frac{1 + \sqrt{F_0}}{1 - \sqrt{F_0}}. \quad (10)$$

Values of n determined in this way can then be substituted into the original Fresnel equation to obtain the reflectance F at other angles of incidence. The procedure may then be repeated at other wavelengths to obtain the spectral and directional dependence of the reflectance.

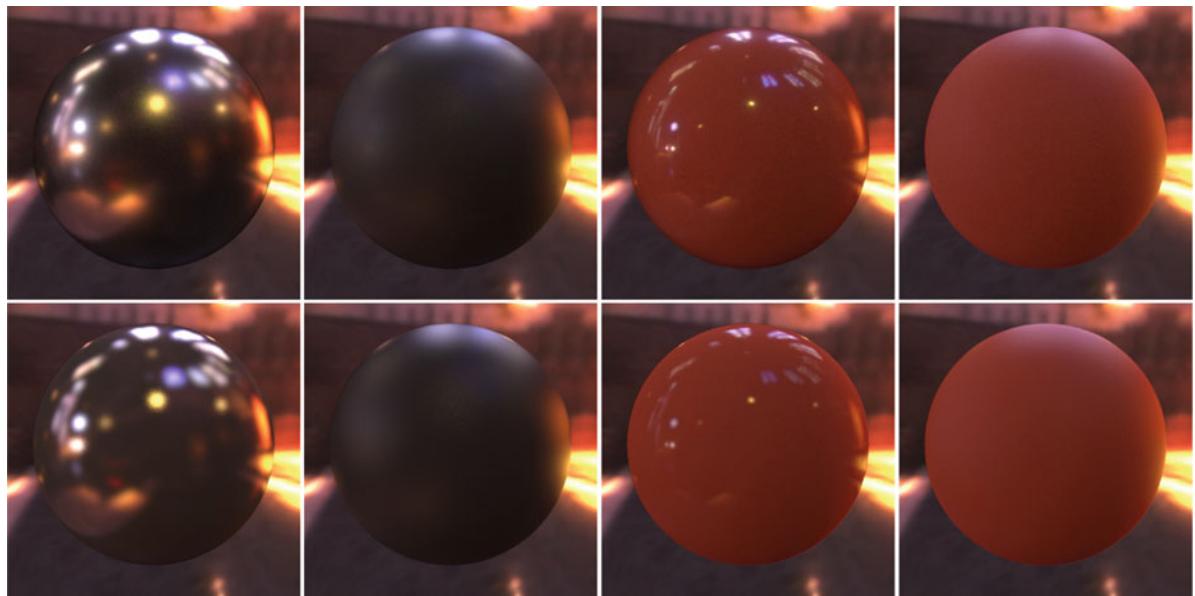
RGB Values

The laws of trichromatic color reproduction are finally used to convert the spectral energy distribution of the



Cook-Torrance Model, Fig. 4 (a) Reflectance of a copper mirror for normal incidence. Wavelength is in micrometers (b) Top curve: Spectral energy distribution of light reflected from a copper mirror illuminated by D6500 [2]

D6500. Bottom curve: Spectral energy distribution of light reflected from a copper mirror illuminated by D6500 [2]



Cook-Torrance Model, Fig. 5 Measured isotropic BRDFs [14] (top row) and their Cook-Torrance fits [15] (bottom row). Left to right: Nickel, Oxidized steel, Red plastic (specular), Dark red paint



Cook-Torrance Model, Fig. 6 Photograph-rendering pairs of Torrance-Sparrow BRDF fits for modeling spatially varying specular reflectance on faces. Left-pair: [16]. Right-pair: [17]

reflected light to the appropriate RGB values for a particular display device. Every color sensation can be uniquely described by its location in a three dimensional color space. One such color space is called the XYZ space. A point in this space is specified by three coordinates, the color's XYZ tristimulus values. Each spectral energy distribution is associated with a point in the XYZ color space and thus with tristimulus values. If two spectral energy distributions are associated with the same tristimulus values, they produce the same color sensation and are called metamers. The goal, then, is to find the proportions of RGB that produce a spectral energy distribution that is a metamer of the spectral energy distribution of the reflected light.

These proportions are determined by calculating the XYZ tristimulus values that are associated with the spectral energy distribution of the reflected light [11], and then calculating the RGB values (with a linear transform from XYZ followed by a non-linear transform based on display gamma curve) that produce a spectral energy distribution with these tristimulus values [13].

Application

Being a physics-based reflectance model, the Cook-Torrance model has been widely used in computer vision and graphics to model the appearance of real world materials. It has been shown to well approximate the reflectance of many measured isotropic BRDFs in the MERL database [14] ranging from metals, plastics, rubber and fabrics [15] (see Fig. 5).

It has also been successfully applied in computer vision to the problem of uncalibrated photometric stereo [18], and in computer graphics to model the measured surface reflectance of human skin [16, 17] (see Fig. 6).

References

- Nicodemus FE, Richmond JC, Hsia JJ, Ginsberg IW, Limperis T (1977) Geometric considerations and nomenclature for reflectance. National Bureau of Standards: NBS Monograph, vol 160
- Cook R, Torrance KE (1982) A reflection model for computer graphics. ACM Trans Graph 1(1):7–24
- Torrance KE, Sparrow EM (1967) Theory of off-specular reflection from roughened surfaces. J Opt Soc Am 57: 1104–1114
- Blinn JF (1977) Models of light reflection for computer synthesized pictures. Comput Graph 11(2):192–198
- Beckmann P, Spizzichino A (1963) The scattering of electromagnetic waves for rough surfaces. MacMillan, New York
- Porteus JO (1963) Relation between the height distribution of a rough surface and the reflectance at normal incidence. J Opt Soc Am 53(12):1394–1402
- Gubareff GG, Janssen JE, Torborg R (1960) Thermal radiation properties survey: a review of the literature. Honeywell Research Center, Minneapolis
- Purdue University (1970) Thermophysical properties of matter. Vol. 7: thermal radiative properties of metals. Plenum, New York
- Purdue University (1970) Thermophysical properties of matter. Vol. 8: thermal radiative properties of nonmetallic solids. Plenum, New York
- Purdue University (1970) Thermophysical properties of matter. Vol. 9: thermal radiative properties of coatings. Plenum, New York

11. CIE International Commission on Illumination (1970) Official recommendations of the international commission on illumination, Colorimetry (E-1.3.1), CIE 15. Bureau Central de la CIE, Paris
12. Sparrow EM, Cess RD (1978) Radiation heat transfer. McGraw-Hill, New York
13. Meyer G, Greenberg D (1980) Perceptual color spaces for computer graphics. Computer Graph 14(3):254–261
14. Matusik W, Pfister H, Brand M, McMillan L (2003) A data-driven reflectance model. ACM Trans Graph 22(3): 759–769
15. Ngan A, Durand F, Matusik W (2005) Experimental analysis of BRDF models. In: Proceeding the Sixteenth Eurographics Conf on Rendering Techniques, Aire-la-Ville, Switzerland, pp 117–126
16. Weyrich T, Matusik W, Pfister H, Bickel B, Donner C, Tu C, McAndless J, Lee J, Ngan A, Jensen HW, Gross M (2006) Analysis of human faces using a measurement-based skin reflectance model. ACM Trans Graph 25(3): 1013–1024
17. Ghosh A, Hawkins T, Peers P, Frederiksen S, Debevec PE (2008) Practical modeling and acquisition of layered facial reflectance. ACM Trans Graph 27(5)
18. Georgiades A, (2003) Incorporating the Torrance and Sparrow model of reflectance in uncalibrated photometric stereo. In: Proceedings of the IEEE international conference on computer vision, Nice, pp 816–823

Coplanarity Constraint

► [Epipolar Constraint](#)

Corner Detection

Michael Maire
California Institute of Technology, Pasadena,
CA, USA

Related Concepts

► [Edge Detection](#)

Definition

Corner detection is the process of locating points in an image whose surrounding local neighborhoods contain edges of different orientations that intersect at those points.

Background

A corner can be viewed as a special type of interest point. Interest points [1] are distinct local image regions with well-defined positions that are robust to various image deformations, such as changes in viewpoint or lighting. Many corner detection algorithms relax the strict requirement of edge intersection and, for example, instead locate centroids of windows containing high edge energy in multiple directions.

Theory

Most corner detection algorithms operate by scoring local image patches [2–6]. Nonmaximum suppression and thresholding steps can then be applied to localize corners by selecting the peak responses and retaining only those deemed sufficiently salient.

An early approach to corner detection scores patches based on their similarity to neighboring patches [7]. Comparing a patch centered on a corner to patches offset by several pixels in any direction should produce a low similarity score as the edges incident at the corner do not align once shifted. In contrast, patches in uniform regions are identical to their neighboring patches. Comparing patches using the sum of squared differences (SSD) yields score:

$$S_{u,v}(x, y) = \sum_{x_i} \sum_{y_i} [I(x_i + u, y_i + v) - I(x_i, y_i)]^2 \quad (1)$$

for comparing the patch in image I centered at pixel (x, y) to the one offset by u in the x -direction and v in the y -direction. The summation is over the pixels (x_i, y_i) belonging to the patch. An optional weighting factor can be used to decrease the importance of the outer patch region with respect to the center. Since patches centered on edge pixels which are not corners exhibit a large SSD when displaced orthogonal to the edge, but no difference when displaced along it, a robust measure of corner strength C takes the minimum over all possible displacements:

$$C(x, y) = \min_{u,v} S_{u,v}(x, y) \quad (2)$$

In the limit of small displacement, the difference in (Eq. 1) can be approximated by image derivatives:

$$\begin{aligned} I(x_i + \Delta x, y_i + \Delta y) - I(x_i, y_i) &= I_x(x_i, y_i)\Delta x \\ &\quad + I_y(x_i, y_i)\Delta y \end{aligned} \quad (3)$$

yielding:

$$S(x, y) = [\Delta x \quad \Delta y] \begin{bmatrix} \sum_{x_i, y_i} (I_x(x_i, y_i))^2 & \sum_{x_i, y_i} I_x(x_i, y_i) I_y(x_i, y_i) \\ \sum_{x_i, y_i} I_x(x_i, y_i) I_y(x_i, y_i) & \sum_{x_i, y_i} (I_y(x_i, y_i))^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \quad (4)$$

Define:

$$A(x, y) = [\nabla I \nabla I^T] \Big|_{(x, y)} \quad (5)$$

The eigenvalues λ_1 and λ_2 of $A(x, y)$ describe the behavior of the local neighborhood of point (x, y) . This region is uniform if λ_1 and λ_2 are both small, an edge if exactly one of λ_1, λ_2 is large, and a corner if both are large.

The Harris corner detector [4] translates this observation into a measure of corner strength given by:

$$C(x, y) = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \quad (6)$$

where k is a parameter. Shi and Tomasi [8] instead use:

$$C(x, y) = \min(\lambda_1, \lambda_2) \quad (7)$$

Lindeberg [5] extends the Harris detector to operate across multiple scales and adds automatic scale selection.

Wang and Brady [9] define corners in terms of curvature using the score:

$$C(x, y) = \nabla^2 I - k |\nabla I|^2 \quad (8)$$

where k is again a constant user-defined parameter (Fig. 1).

Forstner and Gulch [3] take a different approach to corner localization by finding points p whose distance to edges in their local neighborhood is minimal. Specifically,

$$p = \operatorname{argmin}_{(x', y')} \sum_{(x_i, y_i)} D((x_i, y_i), (x', y')) \quad (9)$$

where D is the distance from point (x', y') to the line passing through (x_i, y_i) with orientation orthogonal to the gradient at (x_i, y_i) . The distance is further

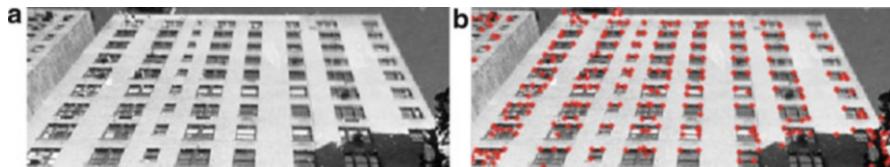
weighted by the gradient magnitude. A similar technique can be used to locate junctions with respect to discrete contour fragments [10].

Application

Interest points, including corners, have found use in a variety of computer vision applications such as image matching, object detection and tracking, and 3D reconstruction. The ability to localize interest points in different views of the same object makes matching and tracking feasible. By design, corner and other interest point detectors respond at locations with rich structure in the surrounding image neighborhood. Thus, these positions are natural choices at which to compute informative feature vectors that describe local image content. The ability to associate these descriptors with interest points further facilitates matching and detection algorithms. In general, the choice of feature descriptor to compute at corners can be application specific and may or may not be coupled with the choice of corner detection algorithm.

While a goal of corner and interest point detectors is to localize the same physical sites across different views, in practice this is only accomplished in a statistical sense. There is usually a significant chance of both missed and spurious detections. Consequently, algorithms built on these components must be robust to errors in detecting individual corners, and instead rely on average detector performance. A typical approach is to utilize a large number of corners or interest points per image for added redundancy.

It is not strictly necessary to use corners or interest points in the process of extracting feature descriptors from images. Alternatives include computing features at sampled edge points or on regions from a segmentation of the image. The sliding



Corner Detection, Fig. 1 Corners detected using the Harris operator followed by nonmaximum suppression and thresholding

window detection paradigm exhaustively scans the image, computing features over all possible image windows.

References

1. Mikolajczyk K, Schmid C (2004) Scale and affine invariant interest point detectors. In: Int J Comput Vis 60(1):63–86
2. Deriche R, Giraudon G (1993) A computational approach for corner and vertex detection. In: Int J Comput Vis 10(2):101–124
3. Forstner W, Gulch E (1987) A fast operator for detection and precise localization of distinct corners. In: ISPPR intercommission conference on fast processing of photogrammetric data, Interlaken, Switzerland pp 281–305
4. Harris C, Stephens M (1988) A combined corner and edge detector. In: Proceedings of the 4th alvey vision conference, University of Manchester, Manchester, UK
5. Lindeberg T (1998) Feature detection with automatic scale selection. In: Int J Comput Vis 30(2):79–116
6. Ruzon MA, Tomasi C (2001) Edge, junction, and corner detection using color distributions. In: IEEE Trans Pattern Analysis and Machine Intelligence, 23(11):1281–1295
7. Moravec H (1980) Obstacle avoidance and navigation in the real world by a seeing robot rover. Technical Report CMU-RI-TR-3, Carnegie-Mellon University
8. Shi J, Tomasi C (1994) Good features to track. In: International conference on computer vision pattern recognition (CVPR), Seattle, WA, pp 593–600
9. Wang H, Brady M (1995) Real-time corner detection algorithm for motion estimation. Image Vis Comput 13:695–703
10. Maire M, Arbeláez P, Fowlkes C, Malik J (2008) Using contours to detect and localize junctions in natural images. In: IEEE conference on computer vision pattern recognition, Anchorage, AK

Cross Entropy

Ying Nian Wu

Department of Statistics, UCLA, Los Angeles,
CA, USA

Definition

Cross entropy is a concept in information theory to measure the independence of two probability distributions.

Theory

For two distributions $p(x)$ and $q(x)$ defined on the same space, the cross entropy is defined as

$$\begin{aligned} H(p, q) &= E_p[-\log q(X)] \\ &= E_p[-\log p(X)] + E_p[\log(p(X)/q(X))] \\ &= H(p) + KL(p, q), \end{aligned}$$

where $H(p) = E_p[-\log p(X)]$ is the entropy of p and $KL(p, q) = E_p[\log(p(X)/q(X))]$ is the Kullback-Leibler divergence from p to q . The Kullback-Leibler divergence is also called relative entropy.

The cross entropy method is a Monte Carlo method for rare event simulation and stochastic optimization [2].

References

1. Cover TM, Thomas JA (1991) Elements of information theory. Wiley, New York
2. Rubinstein RY (1997) Optimization of computer simulation models with rare events. Eur J Oper Res 99:89–112

Curvature

Takayuki Okatani

Graduate School of Information Sciences, Tohoku
University, Sendai-shi, Japan

Synonyms

Curvedness

Related Concepts

► [Geodesics, Distance Maps, and Curve Evolution](#)



Definition

Curvature is a fundamental concept of differential geometry that represents local “curvedness” of some object such as a curve, a surface, and a Riemannian space.

Background

Dealing with the shape of an object is a fundamental issue of computer vision. It is necessary, for example, to represent the two-dimensional or three-dimensional shape of an object, to extract the object shape from various types of images, and to measure similarity between two object shapes. The application of differential geometry to these problems is getting more and more common in modern computer vision. Curvature is one of the most fundamental concept of differential geometry, and its use can be seen throughout all sorts of related problems. This entry explains basic definitions of the curvature of a plane curve and a surface in Euclidean space and summarizes their applications to a few major applications. See [8] for the definition of curvatures of a Riemannian space.

Theory and Application

A plane curve C can be specified by the coordinates $(x(s), y(s))$ of each point, where s is a monotonic function of the arc length. Algebraically, the curvature $\kappa(s)$ of C at a point $(x(s), y(s))$ is defined as

$$\kappa(s) = \frac{\dot{x}\ddot{y} - \dot{y}\ddot{x}}{(\dot{x}^2 + \dot{y}^2)^{3/2}}, \quad (1)$$

where $\dot{x} = dx/ds$, $\ddot{x} = d^2x/ds^2$, etc. Note that the denominator becomes 1 when s is the arc length itself. Let $\phi(s)$ be the orientation of the tangent to C at $(x(s), y(s))$, that is, $\tan\phi(s) = \dot{y}/\dot{x}$. If s is the arc length, the curvature can be represented as $\kappa = \dot{\phi}(s)$.

Geometrically, the absolute value of κ is equal to the reciprocal of the radius r of the circle osculating the curve C at the point, that is, $|\kappa| = 1/r$, as shown in Fig. 1a; r is called the radius of curvature. The curve C can also be locally approximated by

a second-order polynomial curve. Consider the local coordinates $X Y$ defined by the tangent and the normal vectors to C , as shown in Fig. 2b. The approximating quadratic curve will be given as $Y = \kappa X^2/2$. Note that there is freedom in the choice of the sign of the curvature; in its definition of Eq. (1), the sign depends on the parametrization of s .

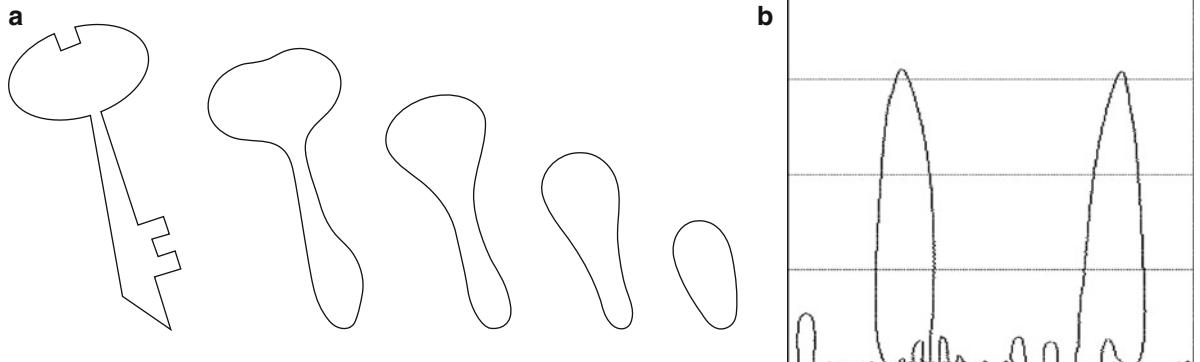
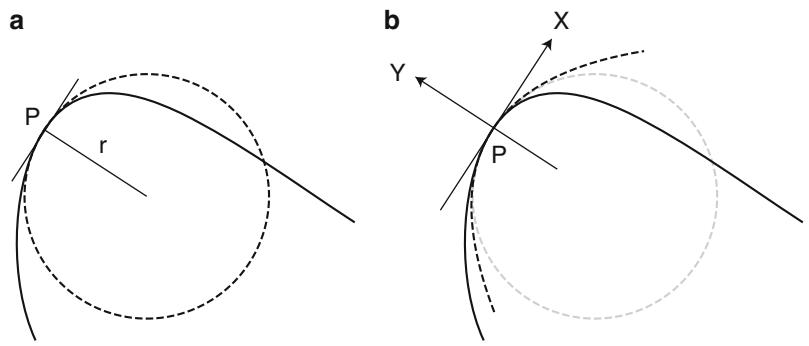
The curvature of a plane curve is effectively used to analyze and/or represent two-dimensional shapes. In [2], the curvature primal sketch is proposed, in which significant changes in curvature along a curve are detected and used for shape representation. In [6], the concept of curvature scale space (CSS) is presented, where the zero-crossings of curvature (i.e., the inflection points of the curve) are used to represent the structure of the curve at varying levels of detail. This technique has been applied to various computer vision problems, such as feature extraction, shape retrieval, and object recognition [5].

The CSS technique is summarized as follows. Let $X(s; \sigma)$ and $Y(s; \sigma)$ be the convolutions of $x(s)$ and $y(s)$ with a Gaussian kernel $g(s, \sigma)$, respectively, that is, $X(s; \sigma) = x(s) * g(s, \sigma)$ and $Y(s; \sigma) = y(s) * g(s, \sigma)$. As shown in Fig. 2a, the curve given by $(X(s; \sigma), Y(s; \sigma))$ gradually becomes smoother, as σ increases. Figure 2b shows a $s - \sigma$ plot of the trajectories of the inflection points. The horizontal axis indicates the normalized arc length s that is scaled to the range $[0-1]$.

The curvature of a plane curve serves as a basic measure of its smoothness. A smoother curve has smaller curvature at each point. Curve evolution like the one in CSS is used in many applications besides shape analysis/representation, where this property with curvature plays a central role.

The active contour model (ACM) [9] was developed to detect the contour of an object in an image by moving an elastic curve from a given initial position to nearby the object contour. This is performed by minimizing the sum of two energy terms, an external term modeling the similarity/dissimilarity to the image edges and an internal term modeling the “elasticity” of the curve. This latter term usually includes the arc length as well as the curvature of the curve to obtain a smoother curve that is more desirable in practice. ACM was later reformulated as a problem of finding local geodesics in a space with Riemannian metric computed from the image, which is known as the geodesic active contour model [4]. The same smoothing property associated with (Euclidean) curvature also plays a key role there.

Curvature, Fig. 1 (a) A plane curve (solid line) and an osculating circle (broken line). The curvature κ at P is the reciprocal of the radius r of the circle. (b) The approximating quadratic curve at P (broken line)



Curvature, Fig. 2 (a) The curvature scale space of a shape. (b) $s-\sigma$ plot of the trajectories of the inflection points of the shape

The evolution of a curve can be fully represented by specifying the normal speed of the curve, that is, the evolution speed measured along the normal vector at each point. Consider a closed curve evolving with speed equal to its curvature, where the sign of the curvature is chosen so that a circle would shrink inward. Its generalization to higher-dimensional space is known as the mean curvature flow. This curve evolution has the properties of “smoothing”; the curve evolves so that its high curvature parts are smoothed out in a finite time. It is shown that any closed curve will become a convex curve and then shrink to a point (Grayson’s theorem [7]).

The level set method (LSM) [17] is a numerical framework for computing such curve evolutions (as well as surface/manifold evolution), which has many advantages to previous methods, such as being able to handle topological changes of the curves. In LSM, a curve is represented as a zero-level curve of an auxiliary function ϕ as $\phi(x, y) = 0$. Then, the curve evolution with normal speed F is represented as $\partial\phi/\partial t = -F|\nabla\phi|$, a time-dependent evolution equation of ϕ .

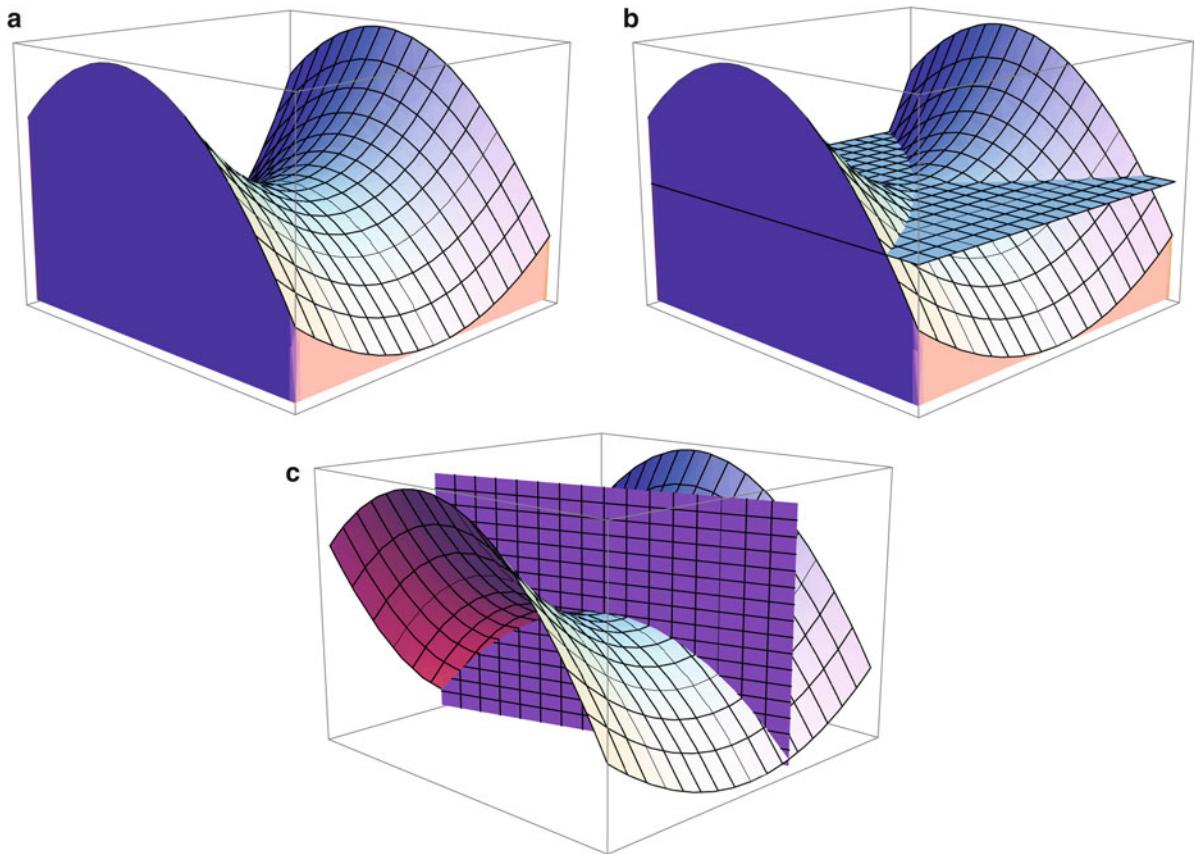
The curvature κ of each point of the level curve is computed using ϕ as

$$\kappa = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} \quad (2)$$

where ∇ is the gradient operator $\nabla = (\partial/\partial x, \partial/\partial y)$. Thus, the mean curvature flow is represented as $\partial\phi/\partial t = -\kappa|\nabla\phi|$.

In some problems, the evolution equations of an image $I(x, y)$, which are similar to those of $\phi(x, y)$ above, are considered. An example is the diffusion equation [15] represented as $\partial I/\partial t = \nabla \cdot (c(x, y)\nabla I)$, which has many applications, for example, image denoising/restoration [16] and image inpainting [3]. Although the choice of $c(x, y)$ depends on each application, it often has a term of the curvature of the level curves of $I(x, y)$, which can be also computed by Eq. (2) (When $c(x, y)$ is constant, the resulting image evolution coincides with the Gaussian blurring).

The curvature of a smooth surface S in a three-dimensional space is defined as follows. Consider the



Curvature, Fig. 3 (a) A surface S . (b) The tangent plane to S at a point P in the center. (c) A plane containing the normal vector of S at P . The intersecting curve on the plane gives the normal curvature of S at P

tangent plane to S at a point P of S (Fig. 3b). The normal vector of S at P should be perpendicular to the tangent plane. Then consider a plane containing this normal vector (Fig. 3c). The intersection of the plane with S yields a plane curve on it. The curvature of this plane curve at P , defined in the same manner as above, is called the normal curvature of S at P . The plane has a one-dimensional rotational freedom around the normal vector, and the normal curvature is defined for each of such planes.

Let the maximum and minimum values of the normal curvature at P be k_1 and k_2 . They are called the principal curvatures of S at P . Consider a local coordinate frame XYZ whose origin is located at P and Z axis coincides with the normal vector. The surface is locally approximated by a second-order polynomial surface

$$Z = \frac{1}{2} [X \quad Y] \mathbf{H} \begin{bmatrix} X \\ Y \end{bmatrix}, \quad (3)$$

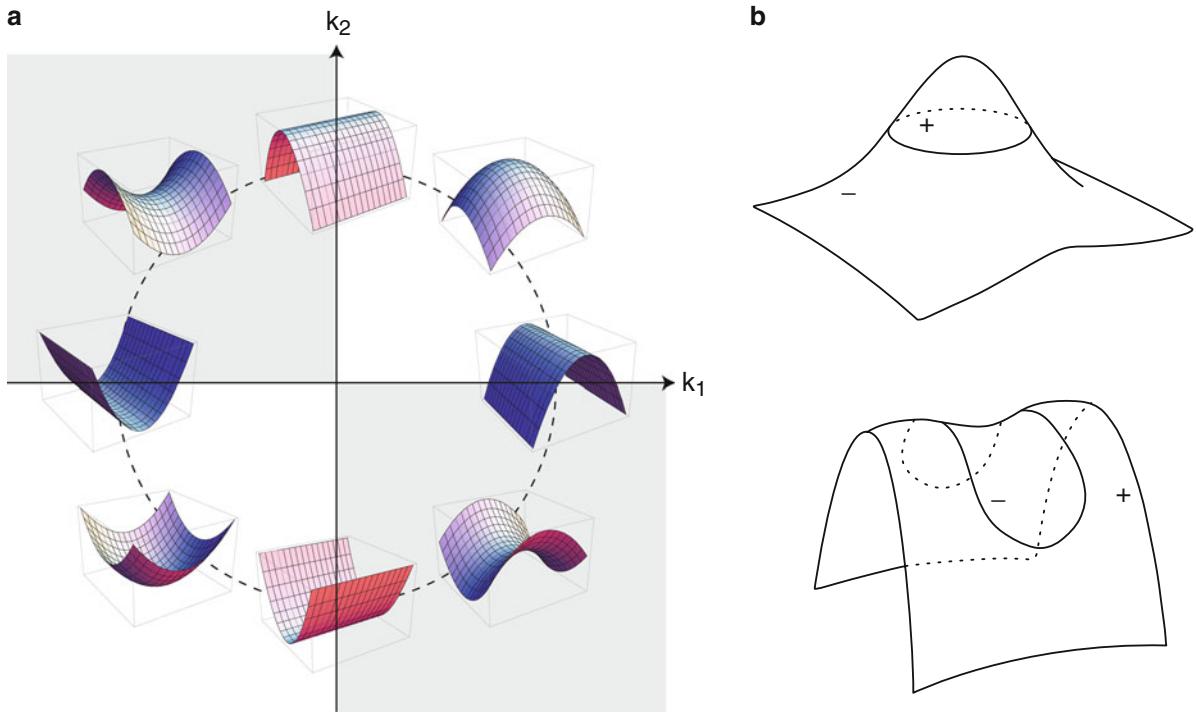
where \mathbf{H} is a 2×2 symmetric matrix. Specifying a direction in the XY plane by a two-dimensional normalized vector \mathbf{v} ($\|\mathbf{v}\|^2 = 1$), the normal curvature for the direction \mathbf{v} is given by $\mathbf{v}^\top \mathbf{H} \mathbf{v}$. The eigenvalues of \mathbf{H} are the same as the principal curvatures k_1 and k_2 , and their associated eigenvectors the corresponding directions \mathbf{v} 's. The Gaussian curvature of S at P is defined as a product of principal curvatures k_1 and k_2 ,

$$\kappa = k_1 k_2, \quad (4)$$

and the mean curvature of S at P is defined as their mean,

$$h = \frac{k_1 + k_2}{2}. \quad (5)$$

Local shapes of a surface are classified by the signs of the principal curvatures k_1 and k_2 , as shown in Fig. 4a [10]. A point at which k_1 and k_2 have the same sign, that is, the Gaussian curvature $\kappa > 0$, is called



Curvature, Fig. 4 (a) Classification of local shapes according to principal curvatures k_1 and k_2 . The first and third quadrants are where the surface point is elliptical, and the second and

an elliptical point. If k_1 and k_2 have different signs, that is, $\kappa < 0$, then the point is called a hyperbolic point. If either k_1 or k_2 vanishes, that is, $\kappa = 0$, then the point is called a parabolic point. Using this classification, a smooth surface may be segmented into finite regions depending on the sign of the curvatures; Fig. 4b shows examples of the surface segmentation based on the Gaussian curvature signs.

This classification method is used in all sorts of applications. For example, it is used in the detection of features such as corners and edges in images, where the images or their variants are regarded as the surface whose local shape is classified; see, for example, [11]. In [18], the shape of an object obtained as range data is represented based on its curvature. It is shown in [1, 13] that similar curvature-based shape representation can be computed from multiple images taken under different illumination directions but without detailed knowledge on the process of the image formation.

More advanced forms of curvatures, such as the curvature of higher-dimensional submanifolds and the Riemannian curvature tensors, are used in recent studies. In [12], several Riemannian metrics on the

fourth quadrants are where it is hyperbolic. (b) Examples of the surface segmentation based on the sign of the Gaussian curvature

space of two-dimensional shapes are studied. In [14], biases of the maximum likelihood estimates derived for the problems of estimating some geometric structure from images (e.g., the epipolar geometry) are related to the curvature of the hypersurfaces given by the geometric structure.

References

- Angelopoulou E, Wolff LB (1998) Sign of gaussian curvature from curve orientation in photometric space. IEEE Trans Pattern Anal Mach Intell 20(10):1056–1066
- Asada H, Brady M (1986) The curvature primal sketch. IEEE Trans Pattern Anal Mach Intell 8(1):2–14
- Bertalmio M, Sapiro G, Caselles V, Ballester C (2000) Image inpainting. In: Proceedings of SIGGRAPH 2000, New Orleans, pp 417–424
- Caselles V, Kimmel R, Sapiro G (1997) Geodesic active contours. Int J Comput Vis 22(1):61–79
- Mokhtarian F, Bober M (2003) Curvature scale space representation: theory, Applications, and MPEG-7 Standardization. Kluwer Academic, Dordrecht
- Mokhtarian F, Mackworth A (1992) A theory of multi-scale, curvature-based shape representation for planar curves. IEEE Trans Pattern Anal Mach Intell 14(8):789–805
- Grayson MA (1987) The heat equation shrinks embedded plane curves to round points. J Differ Geom 26(2):285–314

8. Hazewinkel M (ed) (2002) Encyclopaedia of mathematics. Springer. <http://eom.springer.de/C/c027320.htm>
9. Kass M, Witkin A, Terzopoulos D (1988) Snakes: active contour models. *Int J Comput Vis* 1:321–331
10. Koenderink JJ (1990) Solid shape. MIT, Cambridge
11. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110
12. Michor PW, Mumford D (2004) Riemannian geometries on spaces of plane curves. *J Eur Math Soc* 8:1–48
13. Okatani T, Deugchi K (1999) Computation of the sign of the gaussian curvature of a surface from multiple unknown illumination images without knowledge of the reflectance property. *Comput Vis Image Underst* 76(2):125–134
14. Okatani T, Deugchi K (2009) On bias correction for geometric parameter estimation in computer vision. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), Miami, pp 959–966
15. Perona P, Malik J (1990) Scale-space and edge detection using anisotropic diffusion. *IEEE Trans Pattern Anal Mach Intell* 12(7):629–639
16. Rudin LI, Osher S, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. *Phys D* 60:259–268
17. Sethian JA (1999) Level set methods and fast marching methods. Cambridge University Press, Cambridge
18. Vemuri BC, Mitiche A, Aggarwal JK (1986) Curvature-based representation of objects from range data. *Image Vis Comput* 4(2):107–114

Curvedness

► [Curvature](#)

Curves

► [Curves in Euclidean Three-Space](#)

Curves in Euclidean Three-Space

Jan J. Koenderink

Faculty of EEMSC, Delft University of Technology,
Delft, The Netherlands
The Flemish Academic Centre for Science and the
Arts (VLAC), Brussels, Belgium
Laboratory of Experimental Psychology, University
of Leuven (K.U. Leuven), Leuven, Belgium

Synonyms

[Curves](#); [Space curves](#)

Related Concepts

► [Curvature](#); ► [Differential Invariants](#); ► [Euclidean Geometry](#); ► [Osculating Paraboloids](#)

Definition

Space curves are one-parameter manifolds immersed in Euclidean 3D space $\mathbf{r}(s) \subset \mathbb{E}^3$, where $s \in \mathbb{R}$. One requires differentiability to whatever order is necessary, and $\|\frac{d\mathbf{r}(s)}{ds}\|^2 \neq 0$. It is convenient to require $\|\frac{d\mathbf{r}(s)}{ds}\|^2 = 1$, though this can always be achieved through a reparameterization. Such curves are known as “rectified” or “parameterized by arc-length,” and one writes $\ddot{\mathbf{r}}(s)$ for the partial derivative with respect to arc-length ($\mathbf{r}'(t)$ will be used if the parameter t is not arc-length). As discussed below, in addition one requires $\ddot{\mathbf{r}}(s) \neq 0$ for a generic space curve. The very notion of “rectifiable” is of course Euclidean. Curves in non-Euclidean spaces (affine, projective) have to be handled in appropriate ways.

Background

The classical theory of curves starts with Newton and Leibniz; it was brought in the form presented here in the course of the eighteenth and nineteenth century.

Theory

In differential geometry the “shape” of a curve is defined locally as a set of differential invariants that are algebraic combinations of derivatives $\{\mathbf{r}, \dot{\mathbf{r}}, \ddot{\mathbf{r}}, \dots\}$, and that are invariant with respect to Euclidean motions (notice that “congruencies” would assign the same shape to helices of opposite chirality). A complete set of such differential invariants, specified as a function of the parameter s (arc-length), allows one to construct the curve on the basis of this, up to arbitrary motions. Such a specification of the curve is known as its “natural equations.”

In performing coordinate-wise operations one has to refer to a fiducial frame, most conveniently an orthonormal basis. All equations will take on their simplest form in a frame that is especially fit to the curve. The classical Frenet-Serret frame is one way to

achieve this. As one moves along the curve the frame will rotate in various ways. The simplest description expresses the instantaneous motion of the frame in terms of the frame itself. The differential invariants have geometrically intuitive interpretations in such a system. This insight is due to Elie Cartan, though already implicit in the classical formulation.

These are the basic insights of the classical theory. A short derivation with appropriate geometrical interpretations follows.

The first derivative of position $\mathbf{r}(s)$ with respect to arc-length s is (by construction) a unit vector known as the *tangent* to the curve, denoted $\mathbf{t}(s) = \dot{\mathbf{r}}(s)$. Geometrically the tangent is the direction of the curve, it is the limit of the difference of two points $\mathbf{r}(s_1) - \mathbf{r}(s_2)$ (with $s_1 > s_2$) of the curve divided by the chord-length, that is, $\|\mathbf{r}(s_1) - \mathbf{r}(s_2)\|$, as the points approach each other infinitesimally. Thus the expression $\mathbf{r}(s_0) + (s - s_0)\mathbf{t}(s_0)$ is a first order approximation, the tangent line, to the curve.

Because the tangent is a unit vector, one has $\mathbf{t} \cdot \mathbf{t} = 0$. (Note the Euclidean nature of this!) Thus the second-order derivative $\ddot{\mathbf{r}}$ is orthogonal to the tangent. We write $\ddot{\mathbf{r}} = \kappa \mathbf{n}$, where the unit vector $\mathbf{n}(s)$ is the “normal” to the curve, and $\kappa(s)$ the “curvature.” Notice that the normal would be undefined in the case the tangent did not change direction. For a generic space curve we have to require $\kappa > 0$ throughout (though of course the choice of sign is arbitrary). This is different from planar curves for which a signed curvature makes sense. Thus planar curves may have points of inflection, whereas this notion makes no sense for space curves (Fig. 1).

The normal and the tangent define a plane, the so-called osculating plane of the curve. It is the limit of the plane spanned by three points of the curve as the points approach each other infinitesimally. One might say that, at least locally, the curve lies in its osculating plane (remember that “to osculate” means literally “to kiss”). The three points also define a circle (lying in the osculating plane), whose radius can be shown to be $1/\kappa$. (An easy way to show this is to write down the second order Taylor development of the curve.) Thus, locally, the curve is like a circular arc of radius $1/\kappa$ in the osculating plane, moving in the tangent direction. The curvature measures the degree to which the curve veers away from the tangent direction, into the direction of the normal. Notice that $\ddot{\mathbf{r}} \cdot \mathbf{n} = \kappa$, and that the curvature is a scalar that does not depend upon the

coordinate frame. The curvature is the first example of a differential invariant.

In Euclidean space \mathbb{E}^3 the tangent and the normal imply a third vector orthonormal to them both. It is $\mathbf{b} = \mathbf{t} \times \mathbf{n}$, known as the “binormal.” This is again a very Euclidean construction, the vector product being a Euclidean 3D concept. The orthonormal frame $\{\mathbf{t}, \mathbf{n}, \mathbf{b}\}$ is the Frenet-Serret frame. It is tightly connected to the curve and a complete basis of \mathbb{E}^3 . Thus it is perfectly suited to describe the third order derivative $\ddot{\mathbf{r}}$. The obvious move is to express \mathbf{n} in terms of the tangent and the binormal (given the fact that the normal is a unit vector this should be possible). Thus one writes $\mathbf{n} = -\kappa \mathbf{t} + \tau \mathbf{b}$, where τ is another differential invariant known as the torsion of the curve. The reason for the term $-\kappa \mathbf{t}$ is that $\mathbf{t} = \kappa \mathbf{n}$: the frame turns about the binormal with angular speed κ (Fig. 2). The third derivative itself then is $\ddot{\mathbf{r}} = -\kappa^2 \mathbf{t} + \kappa \mathbf{n} + \tau \mathbf{b}$.

The torsion (sometimes called “second curvature”) has a simple geometrical interpretation. It is the angular rate of change of the attitude of the osculating plane.

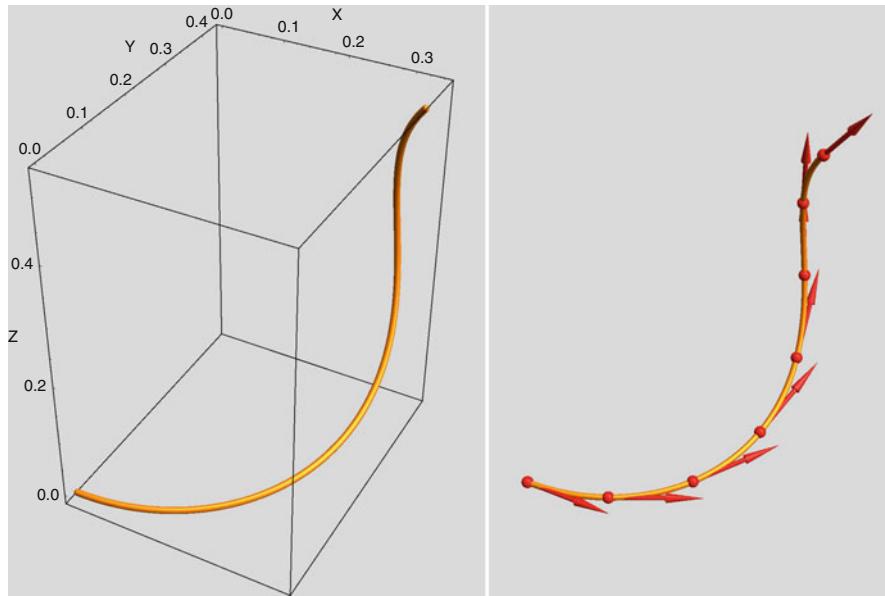
Notice that the metrical structure of \mathbb{E}^3 is used in an essential manner in all constructions thus far. The classical theory of curves cannot be used in spaces with different structures, even in homogeneous spaces such as affine or projective spaces. Of course, a theory of curves can be developed for such spaces too, but the differential invariants, although often denoted “curvature” and “torsion,” will have meanings that are completely distinct from the curvature or torsion of curves in Euclidean space. The reader should be keenly aware of this, as non-Euclidean spaces occur frequently and naturally in computer vision and image processing applications, the best known being the affine and projective 3D spaces, as well as “graph space.”

The structure found thus far can be appreciated from a straightforward Taylor expansion:

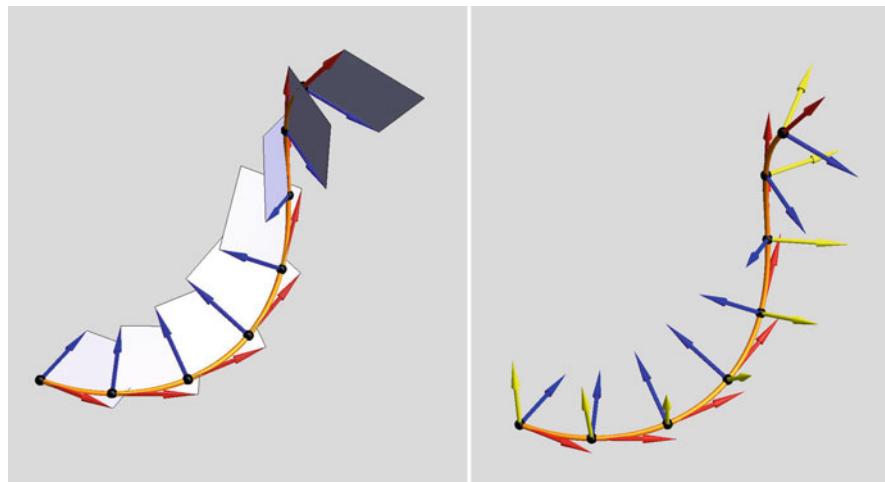
$$\mathbf{r}(s) = \mathbf{r}(0) + \dot{\mathbf{r}}s + \ddot{\mathbf{r}}\frac{s^2}{2!} + \ddot{\mathbf{r}}\frac{s^3}{3!} + O[s]^4, \quad (1)$$

which in terms of the Frenet-Serret frame is (the “canonical representation”):

$$\begin{aligned} \mathbf{r}(0) &+ \left(s - \kappa^2 \frac{s^3}{3!} + \dots \right) \mathbf{t} + \left(\kappa \frac{s^2}{2!} + \kappa \frac{s^3}{3!} + \dots \right) \mathbf{n} \\ &+ \left(\kappa \tau \frac{s^3}{3!} + \dots \right) \mathbf{b}, \end{aligned} \quad (2)$$



Curves in Euclidean Three-Space, Fig. 1 At left generic space curve. It is curved throughout and “winds” in a single sense. This curve was defined via its natural equations (see below),
 $\kappa(s) = 1 + 3s, \tau(s) = 1 + 5s, 0 < s < 1$. At right the field of tangents along the curve

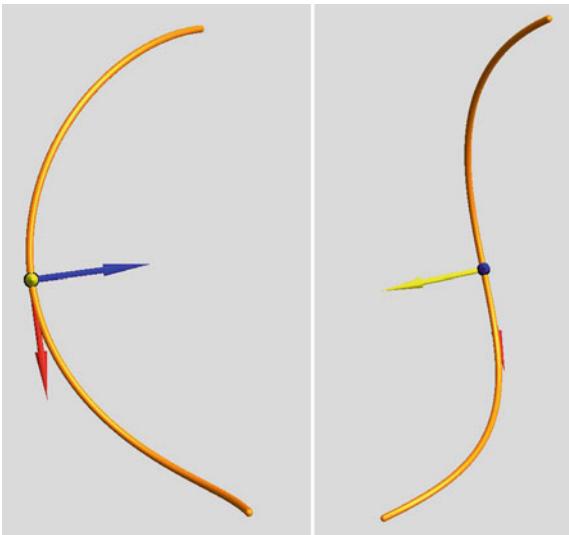


Curves in Euclidean Three-Space, Fig. 2 At left the field of osculating planes along the space curve. Notice how it rotates, revealing the curve to be a “twisted” one. At right the field of

Frenet frames along the curve. Again, notice how it rotates as it moves along the curve

from which the habitus of the curve is readily gleaned: The projection on the osculating plane is (approximately) a parabolic arc, on the normal ($\mathbf{n} \times \mathbf{b}$) plane a cusp, and on the tangential ($\mathbf{b} \times \mathbf{t}$) plane an inflection (Figs. 3 and 4).

Notice that the third order includes a term in the rate of change of curvature, not merely the torsion. The meaning of this becomes evident from another geometrical construction. Four distinct points define a sphere, and in the limit one obtains the osculating sphere at a



Curves in Euclidean Three-Space, Fig. 3 At left a view from the binormal, at right a view from the normal direction. From the binormal direction the curve shows its curvature, from the normal direction one sees an inflection

point of the curve. When the curve is twisted at the point ($\tau \neq 0$), the center of the osculating sphere is given by:

$$\mathbf{c}_{osc} = \mathbf{r}(s) + \frac{1}{\kappa(s)} \mathbf{n}(s) - \frac{\dot{\kappa}(s)}{\kappa(s)^2 \tau(s)} \mathbf{b}(s), \quad (3)$$

and its radius of curvature $\varrho_{osc}(s)$ by:

$$\varrho_{osc} = \sqrt{\varrho^2(s) + \left(\frac{\dot{\varrho}(s)}{\tau(s)} \right)^2}, \quad (4)$$

where ϱ is the radius of the osculating circle. Thus only at “vertices” of the curve ($\dot{\kappa} = 0$) is the osculating circle a great circle of the osculating sphere. The osculating sphere always cuts the osculating plane in the osculating circle though.

The geometrical structure is formulated rather elegantly by the Frenet-Serret formulas (notice the anti-symmetry of the matrix):

$$\begin{pmatrix} \dot{\mathbf{t}} \\ \dot{\mathbf{n}} \\ \dot{\mathbf{b}} \end{pmatrix} = \begin{pmatrix} 0 & +\kappa & 0 \\ -\kappa & 0 & +\tau \\ 0 & -\tau & 0 \end{pmatrix} \begin{pmatrix} \mathbf{t} \\ \mathbf{n} \\ \mathbf{b} \end{pmatrix}. \quad (5)$$

The “natural equations” simply specify $\kappa(s)$ and $\tau(s)$. Using the Frenet-Serret equations one constructs the curve by integration, specifying an arbitrary initial Frenet-Serret frame. Thus the curvature and torsion specify the curve up to arbitrary Euclidean motions.

A useful formalism that extends this is due to Darboux (Fig. 5). The “Darboux vector” is defined as $\mathbf{d} = \tau \mathbf{t} + \kappa \mathbf{b}$. Now one has $\dot{\mathbf{t}} = \mathbf{d} \times \mathbf{t}$, $\dot{\mathbf{n}} = \mathbf{d} \times \mathbf{n}$, and $\dot{\mathbf{b}} = \mathbf{d} \times \mathbf{b}$, thus the Darboux vector is the angular velocity of the “moving trihedron” $\{\mathbf{t}, \mathbf{n}, \mathbf{b}\}$. One immediately concludes that the curvature is the rate of turning about the binormal and the torsion the rate of turning about the tangent. This nicely “explains” the geometrical meaning of the differential invariants κ (the curvature) and τ (the torsion).

The Darboux formalism is by far the simplest to commit to memory. It completely sums up the structure of generic space curves in Euclidean three-space. The Darboux vector also lets one handle degenerate cases easily, for instance that of planar curves (binormal constant), or straight curves (or rather, lines; tangent constant). Finally, the Frenet-Serret equations are written in an easily remembered (because of the cyclic \mathbf{t} - \mathbf{n} - \mathbf{b} -structure) form:

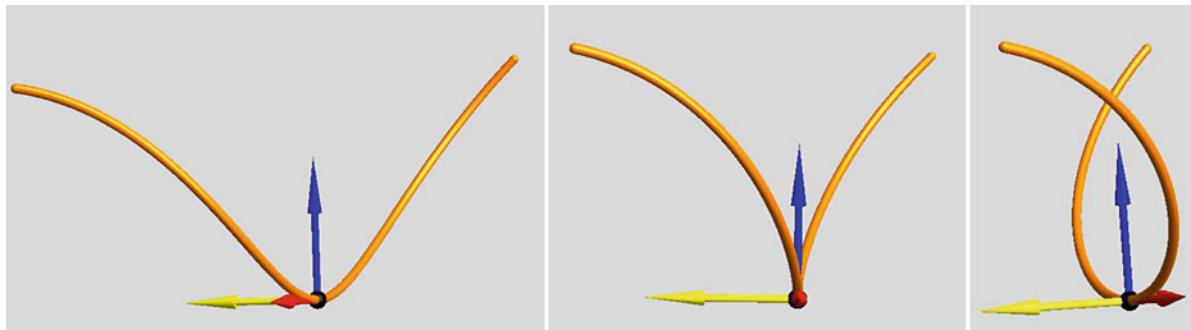
$$\dot{\mathbf{t}} = \mathbf{d} \times (\mathbf{n} \times \mathbf{b}), \quad (6)$$

$$\dot{\mathbf{n}} = \mathbf{d} \times (\mathbf{b} \times \mathbf{t}), \quad (7)$$

$$\dot{\mathbf{b}} = \mathbf{d} \times (\mathbf{t} \times \mathbf{n}). \quad (8)$$

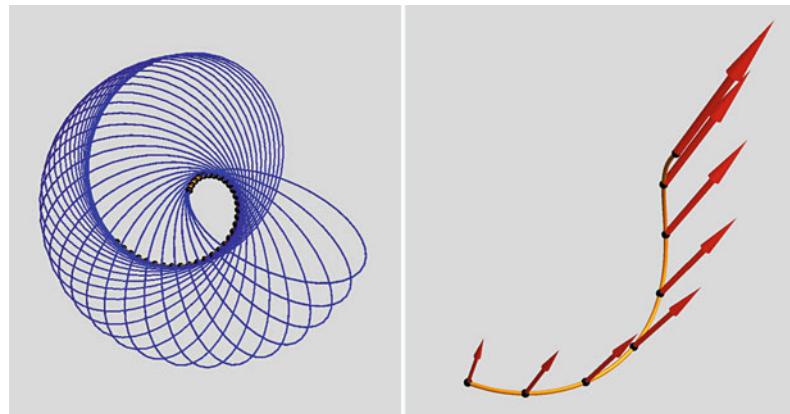
There are a number of geometrical structures related to a curve that are of occasional use. A few of these are discussed below.

So-called spherical images are spherical curves – that are curves on the surface of the unit sphere – related to a curve (Fig. 6). One naturally considers the tangent, normal, and binormal spherical images that are the curves $\mathbf{t}(s)$, $\mathbf{n}(s)$, and $\mathbf{b}(s)$ (notice that these curves are not rectified!). Notice that for straight lines the tangent spherical images degenerate to a point, whereas the other spherical images are undefined. For planar curves the tangent and normal spherical images are degenerated to arcs of great circles, whereas the binormal spherical image degenerates to a point. Special points of the spherical images, like inflections or cusps, relate to special points of the original curve. For some problems of a physical nature it is the spherical image, rather than the curve itself that is of primary

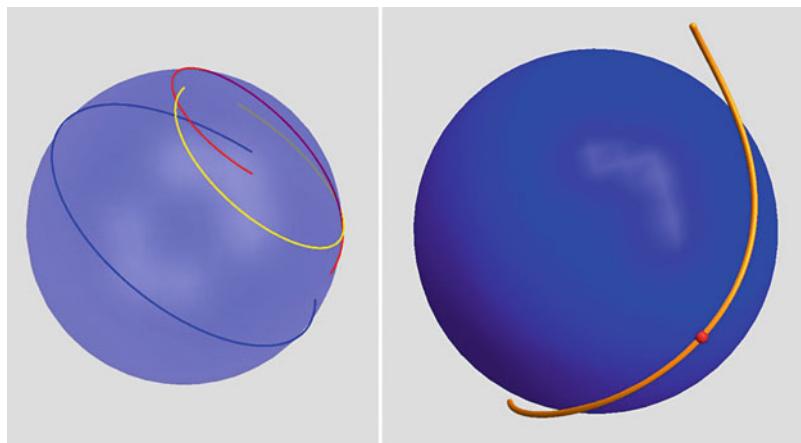


Curves in Euclidean Three-Space, Fig. 4 At center a view from the tangent direction. In this view the curve appears as a cusp. At left and right views from nearby directions

Curves in Euclidean Three-Space, Fig. 5 At left osculating circles along the curve. At right the Darboux vector field along the curve



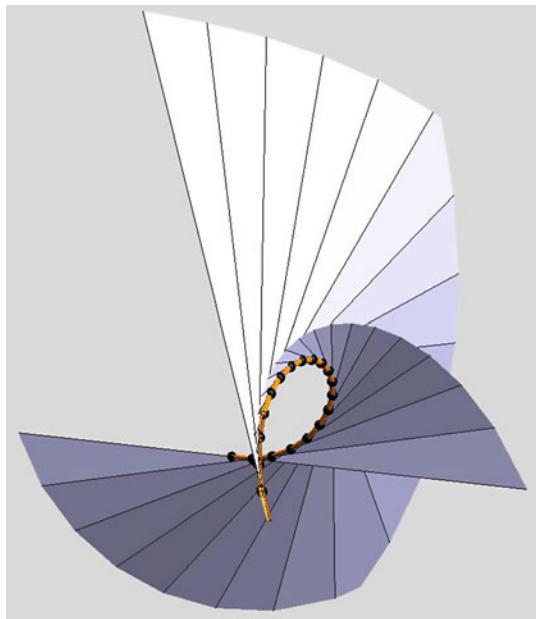
Curves in Euclidean Three-Space, Fig. 6 At left the spherical images associated with the curve (red: tangent image; blue: normal image; yellow: binormal image). At right an osculating sphere at the curve



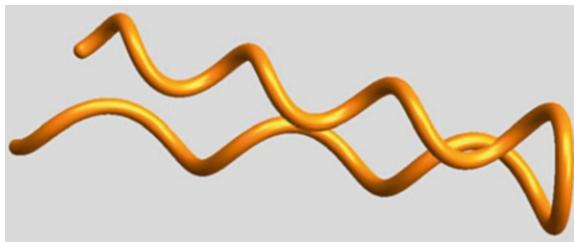
interest. An example is that of specularities on tubular surfaces like human hairs.

The surfaces described by the lines defined by the tangent, normal, binormal, and Darboux vector are also of occasional interest. These are – by construction –

ruled surfaces, though not necessarily developable ones. Their singular features, like the edge of regression in the case of developable surfaces or the line of striction in the case of skew surfaces, have useful geometrical relations to the curve. The best known



Curves in Euclidean Three-Space, Fig. 7 The surface described by the tangents to the curve is a developable surface, the curve being its edge of regression



Curves in Euclidean Three-Space, Fig. 8 Example of a torsion zero. Notice the opposite chirality of the parts of the curves at either side of the torsion zero (which is at the *right* in this picture)

example is the surface of tangent lines, which happens to be developable, with the curve itself as the edge of regression, and the surface of normal lines, which is commonly used to describe surface strips (Fig. 7).

Special points on the curve may also be studied by direct means of course. Perhaps the most obvious

instance is that of a torsion zero (Fig. 8). At a torsion zero the chirality of the curve changes. Whereas the curve generically osculates, but also pierces its osculating plane, the curve merely osculates, but fails to pierce the osculating plane, it being “deflected” by it. Such special points are often introduced by design in telephone cords, and many vines also have frequent torsion zeroes in their tendrils

Additional Problems

This entry describes the Euclidean differential geometry of space curves. In many problems the context is different from Euclidean though. Because the differential invariants introduced here are specific for the Euclidean transformation group, one needs to develop the differential geometry from scratch. Examples frequently occur in computer vision, for instance, and one often works in spaces with mere affine, or even projective structure. Spaces with even less structures are common. A common case involves “isotropic differential geometry” in “graph spaces.”

References

1. Bruce JW, Giblin PJ (1992) Curves and singularities. Cambridge University Press, Cambridge, MA
2. do Carmo MP (1976) Differential geometry of curves and surfaces. Prentice-Hall, Englewood Cliffs
3. Eisenhart LP (2004) A treatise on the differential geometry of curves and surfaces. Dover, New York
4. Gray A (1997) Modern Differential Geometry of Curves and Surfaces with Mathematica, 2nd edn. CRC, Boca Raton
5. Guggenheim H (1977) Differential geometry. Dover, New York
6. Kreyszig E (1991) Differential geometry, Chapter II. Dover Publications, New York
7. Lockwood EH (1961) Book of curves. Cambridge University Press, Cambridge, MA
8. Porteous I (2001) Geometric differentiation. Cambridge University Press, Cambridge, MA
9. Spivak M (1999) A Comprehensive introduction to differential geometry, vol 2. Publish or Perish, Houston

D

Data Augmentation

Ying Nian Wu

Department of Statistics, UCLA, Los Angeles,
CA, USA

Definition

Data augmentation is a Markov chain Monte Carlo algorithm for sampling from a Bayesian posterior distribution

Background

Data augmentation was originally developed by Tanner and Wong [10] as a stochastic counterpart of the EM algorithm [1], and it is closely related to the Gibbs sampler [2]. Thus, the basic setup of data augmentation is similar to the EM algorithm.

Theory

Let y be the observed data and z be the missing data or latent variable. Let $p(y, z|\theta)$ be the probability distribution of the complete data (y, z) , with θ being the unknown parameter. The marginal distribution of the observed data y is $p(y|\theta) = \int p(y, z|\theta)dz$. Let $p(\theta)$ be the prior distribution of θ . The goal is to draw Monte Carlo samples from the posterior distribution $p(\theta|y) \propto p(\theta)p(y|\theta)$.

The data augmentation algorithm is an iterative algorithm. It starts from an initial value θ_0 . Let (θ_t, z_t) be the values of θ and z sampled in the t -th iteration.

Then in the $(t + 1)$ -st iteration, it goes through the following two steps:

Imputation step: Sample $z_{t+1} \sim p(z|y, \theta_t)$.
Posterior step: Sample $\theta_{t+1} \sim p(\theta|y, z_{t+1})$.

$p(z|y, \theta)$ is the predictive distribution for imputing the missing data z given y and θ . $p(\theta|y, z)$ is the posterior distribution of the complete-data model. The data augmentation algorithm capitalizes on the fact that both the $p(z|y, \theta)$ in the imputation step and $p(\theta|y, z)$ in the posterior step are often easy to sample from.

In correspondence to the EM algorithm, the imputation step corresponds to the E-step, and the posterior step corresponds to the M-step. The data augmentation algorithm can also be viewed as a two-component Gibbs sampler for sampling from $p(z, \theta|y)$, but the emphasis of the data augmentation algorithm is that it augments the missing data or latent variable z to simplify the computation. In that sense, it is related to the auxiliary variable algorithm [3], the most prominent example being the Swendsen-Wang algorithm [9].

The rate of convergence of the data augmentation algorithm is determined by a quantity called Bayesian fraction of missing information [5]. It is the Bayesian version of the fraction of missing information that determines the rate of convergence of the EM algorithm [1].

Meng and van Dyk [7] observed that for the same marginal model $p(y|\theta)$, it is possible to construct a class of complete-data models $p_a(y, z|\theta)$ indexed by a working parameter a so that $\int p_a(y, z|\theta)dz = p(y|\theta)$ for all a . One may then optimize the fraction of missing information over the working parameter a in order to

devise the EM or data augmentation algorithm with the optimal rate of convergence.

Inspired by Meng and van Dyk [7], Liu, Rubin and Wu observed that when augmenting the data from y to (y, z) , it is possible to expand the parameter θ to θ, α so that the complete model becomes $p(y, z|\theta, \alpha)$, where both θ and α are identifiable in the complete-data model, but $\int p(y, z|\theta, \alpha)dz = p(y|\theta)$ so that the parameter α disappears in the observed-data model [6]. Based on this observation, they proposed a parameter-expanded EM (PX-EM) algorithm which has faster convergence rate than the original EM algorithm. Liu and Wu (2000) [4] proposed a parameter-expanded data augmentation (PX-DA) algorithm which has a faster convergence rate than the original data augmentation algorithm. Independently, Meng and van Dyk [8] proposed a similar algorithm called marginal augmentation.

Application

The data augmentation algorithm and its extensions can be used for sampling of posterior distributions from a wide range of Bayesian models. Practically for any EM algorithm for maximum likelihood estimation, there is a corresponding data augmentation algorithm for posterior sampling. In fact, the imputation step of the data augmentation algorithm can be easier to implement than the E-step of the EM algorithm because it is often easier to sample from a distribution than calculating the expectation in closed form.

References

1. Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. *J R Stat Soc B* 39:1–38
2. Geman S, Geman D (1984) Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans Pattern Anal Mach Intell* 6:721–741
3. Higdon DM (1998) Auxiliary variable methods for Markov chain Monte Carlo with applications. *J Am Stat Assoc* 93:585–595
4. Liu JS, Wu YN (1999) Parameter expansion for data augmentation. *J Am Stat Assoc* 94(448):1264–1274
5. Liu JS, Wong WH, Kong A (1994) Covariance structure of the Gibbs sampler with applications to the comparisons of estimators and augmentation schemes. *Biometrika* 81:27–40
6. Liu C, Rubin DB, Wu YN (1998) Parameter expansion to accelerate EM: the PX-EM algorithm. *Biometrika* 85(4): 755–770
7. Meng XL, van Dyk D (1997) The EM algorithm – an old folk-song sung to a fast new tune. *J R Stat Soc B* 59:511–567
8. Meng XL, van Dyk D (1999) Seeking efficient data augmentation schemes via conditional and marginal augmentation. *Biometrika* 86:301–320
9. Swendsen RH, Wang J (1987) Nonuniversal critical dynamics in Monte Carlo simulations. *Phys Rev Lett* 58:86–88
10. Tanner MA, Wong WH (1987) The calculation of posterior distributions by data augmentation. *J Am Stat Assoc* 82: 528–540

Data Fusion

Ramanarayanan Viswanathan

Department of Electrical Engineering, University of Mississippi, MS, USA

Synonyms

[Information fusion](#)

Definition

Data fusion refers to combining data from multiple sources for achieving better understanding of a phenomenon of interest. Applications abound in engineering and applied sciences, including wireless sensor networks, computer vision, and biometrics.

Background

In several fields, combining different sets of information have taken place, although a more systematic study for the fusion of data is emerging since a decade [1]. The human brain is an example of a complex system which integrates data or signals from different sensory receptors in the body. Building a machine-based system that can meaningfully integrate data from different sources for better understanding of a phenomenon of interest is the challenge faced in many fields. Since data emerges from different sensors with varying accuracy and coverage factor, benefits of data fusion include improved system reliability and/or redundancy, extended coverage, and possible shorter response time. Applications in human-machine interface area include robots in industrial automation, surveillance in military and commercial fields, battlefield management, medical diagnostics,

biometrics, satellite imaging, image understanding, computer vision, target detection and tracking, wireless sensor networks, and wireless communications.

Application

In many vision-related applications, a single sensor will not provide complete information with respect to the scene that is being sensed. Also, sensors may have different fields of views and different sensors, such as IR and optical, may have different resolution capabilities. Better results are obtained when data from sensors are combined in an appropriate manner. For imaging applications, fusion of images can be done at various levels, viz., pixel, feature, and decision. In general, the latter two entail some loss of information as the fusion is performed after extraction of information from the original images. Pixel-level fusion methods include Laplacian pyramid, discrete wavelet transforms, and support vector machines [2]. In decision/classification systems, combining features or decisions may be appropriate when sensors are geographically dispersed, thereby requiring distributed processing at sensor sites. In video surveillance applications involving tracking of objects or persons of interest, fusion of data from multiple images would be needed [3]. Similar situations arise in military applications involving tracking of maneuvering enemy aircrafts. Track-to-track fusion of target position estimates derived from individual sensor data is a possible approach [4, 5].

Detection and estimation of targets using a set of geographically dispersed multiple sensors necessitate distributed signal processing at sensor sites [6, 7]. Although individual sensors process sensed information and possibly make inference regarding the presence or absence of a target (or a phenomenon of interest), a final determination that is based on the collective information is usually made at a central site called the fusion center. It is conceivable that, in some applications, one of the sensors could be the fusion center making the final decision. Thus, the traditional signal detection (estimation) paradigm is naturally extended to the situation of distributed processing. The terminology “distributed detection,” “decentralized detection,” or “distributed decision fusion” refers to such situations. Depending on the flow of information from the sensors to the fusion center and

between the sensors, several configurations of sensor suites are possible: serial or tandem, and parallel, tree [8]. Formulation of Neyman-Pearson and Bayes optimization criteria leads to fundamental theoretical solutions in this area. Asymptotic solutions involving a very large number of sensors as well as computational approaches for obtaining optimal solutions are available [6–9]. Recently, there is interest in the estimation of parameters using processed data from multiple sensors [10].

In earlier works on distributed detection problems, the communications channels between the sensors and the fusion center were assumed ideal or error-free. However, with the pervasiveness of wireless sensor networks, the assumption of error-free links is not quite true. The inclusion of error-prone links in the distributed signal processing systems has further broadened the literature to provide practically achievable as well as theoretically possible solutions [11, 12]. Channel-aware solutions paradigm leads to several suboptimal solutions depending on the availability of knowledge of channel information. Although some similarity to traditional diversity reception in signal communication exists, the decentralized detection in wireless sensor network is distinctly different in the sense that the decisions made by the sensors regarding a phenomenon of interest need not all be identical unlike the case of diversity reception. Thus, the maximal ratio combining, which is optimal for diversity reception, is generally inferior in wireless sensor network case. Depending on the noisiness of the channel, combining based on individual sensor decisions arrived at the fusion center can outperform other suboptimal combiners based on received sensor data at the fusion center.

Cognitive radio has evolved over the last two decades. Cognitive radio now could be a simple receiver which senses and adapts to different signal modulations to a more complex modem where the radio is able to sense the non-presence of a user in a spectral frequency range, thereby adapting its operating frequency to be in that range for achieving signal connectivity [13]. The problem faced by a secondary user is the detection of the presence or absence of a primary licensed-user through spectrum sensing. The spectrum sensing is done cooperatively by several secondary users in order to opportunistically access the spectrum, when it becomes available. Presence of distributed radios, inability of some radios not being

able to sense the primary user due to shadowing or “hidden terminal” problem, limitations on communication capacity between the secondary radios and the radio acting as the fusion center, and availability of signal processing capability within the radios have set the stage for the application of decentralized detection concepts to cognitive radios [14, 15].

Biometrics is used for authentication purposes in civilian and military applications. Biometrics for person identification could be fingerprints, facial image, voice, or iris. Use of several biometrics, when appropriately combined, can lead to better results than that can be obtained through any one biometrics. Also, multiple samples of any one biometrics can be combined for achieving better results. Combining data from multimodal biometrics is a challenging problem because of the need for proper normalization of biometrics scores before combining [16, 17]. The determination of optimal combining method needs to take into account possible correlation among different biometrics. Challenges exist due to stringent requirements in surveillance applications where false accept rates and false reject rates need to be kept small. Moreover, biometrics identification systems need to be robust to the extent possible due to possible fake biometrics posed by latex fingers, face masks, etc. The fusion of biometric can be at various levels: data, features, and classifiers (decisions).

References

1. Varshney PK (1997) Mutisensor data fusion. Electron Commun Eng J 9(12):245–253
2. Zheng S, Shi W-Z, Liu J, Zhu G-X, Tian J-W (2007) Multi-source image fusion method using support value transform. IEEE Trans Image Process 16(7):1831–1839
3. Snidaro L, Niu R, Foresti GL, Varshney PK (2007) Quality-based fusion of multiple video sensors for video surveillance. IEEE Trans Syst Man Cybern Part B Cybern 37(4):1044–1051
4. Hall DL, Llinas J (1997) An introduction to multisensor data fusion. Proc IEEE 85(1):6–23
5. Chen H, Kirubarajan T, Bar-shalom Y (2003) Performance limits of track-to-track fusion versus centralized estimation: theory and application. IEEE Trans Aerosp Electron Syst 39(2):386–399
6. Viswanathan R, Varshney PK (1997) Distributed detection with multiple sensors: part I-fundamentals (invited paper). Proc IEEE 85(1):54–63
7. Blum RS, Kassam SA, Poor HV (1997) Distributed detection with multiple sensors: part II-advanced topics (invited paper). Proc IEEE 85(1):64–79
8. Dasarathy BV (1994) Decision fusion. IEEE Computer Society Press, Los Alamitos
9. Tay PW, Tsitsiklis JN, Win MZ (2008) On the subexponential decay of detection error probabilities in long tandems. IEEE Trans Inf Theory 54(10):4767–4771
10. Ribeiro A, Giannakis GB (2006) Bandwidth-constrained distributed estimation for wireless sensor networks- Part I: Gaussian case. IEEE Trans Signal Process 54(3):1131–1143
11. Chamberland J-F, Veeravalli VV (2007) Wireless sensors in distributed detection applications. IEEE Signal Process Mag 24(3):16–25
12. Chen B, Jiang R, Kasetkesam T, Varshney PK (2004) Channel aware decision fusion in wireless sensor networks. IEEE Trans Signal Process 52(12):3454–3458
13. Gandetto M, Regazzoni C (2007) Spectrum sensing: a distributed approach for cognitive terminals. IEEE J Sel Areas Commun 25(3):546–557
14. Unnikrishnan J, Veeravalli VV (2008) Cooperative sensing for primary detection in cognitive radio. IEEE J Sel Top Signal Process 2(1):18–27
15. Letaief KB, Zhang W (2009) Cooperative communications for cognitive radio networks. Proc IEEE 97(5):878–893
16. Jain AK, Chellappa R, Draper SC, Memon N, Phillips PJ, Vetro A (2007) Signal processing for biometric systems (DSP forum). IEEE Signal Process Mag 24(6):146–152
17. Basak J, Kate K, Tyagi V, Ratha N (2010) QPLC: a novel multimodal biometric score fusion method. In: Computer Vision and Pattern recognition Workshops (CVPRW), San Francisco, 2010. IEEE Computer Society Conference, pp 46–52

Deblurring

Yu-Wing Tai

Department of Computer Science, Korea Advanced Institute of Science and Technology (KAIST), Yuseong-gu, Daejeon, South Korea

Synonyms

[Deconvolution](#)

Related Concepts

► [Blind Deconvolution](#); ► [Blur Estimation](#); ► [Image Enhancement and Restoration](#); ► [Motion Blur](#)

Definition

Deblurring is a process to recover a sharp and clear image from a blurry image which suffered from either the motion blur or the defocus blur effects.

Background

When taking a photo with long exposure time, or with wrong focal length, the captured image will look blurry. This is because during the exposure period, the lights captured for a pixel are mixed with the lights captured for the other pixels within a local neighborhood. Such effect is modeled by the point spread function (a.k.a. blur kernel) which describes how the lights are mixed during the exposure period. The aim of deblurring is to reverse the blur process to recover a sharp and clear image of the scene from the captured blurring image. Deblurring, however, is a severely ill-posed problem because the number of unknowns exceeds the number of equations that can be derived from the observed data.

The problem of deblurring can be further categorized into non-blind deblurring and blind deblurring. In non-blind deblurring, the point spread function is given and is used to recover the latent image from a blurry image. In blind deblurring, the point spread function is unknown and additional process is needed to estimate point spread function. However, the process to estimate point spread function usually requires information of the latent image, which is also unknown. A typical approach in blind deblurring consists of two interdependent steps: point spread function estimation and latent image restoration. These two steps are solved alternately and iteratively, and the solution of blind deblurring is only guaranteed to converge to a local minimum.

Theory

Representation

Mathematically, the effects of blurriness can be described by the following equation:

$$B(x, y) = \sum_{(m, n) \in \mathcal{N}(x, y)} I(x - m, y - n) k_{(x, y)}(m, n) \quad (1)$$

where B is the blurry image, I is the sharp image, $k_{(x, y)}$ is the point spread function (PSF) at (x, y) , and \mathcal{N} is the local neighborhood of a pixel respectively. In general, the point spread function $k_{(x, y)}$ is spatially varying. This means that every pixel can carry different point spread function. This happens when the scene

has large depth disparity (for defocus blur) or when the scene contains a moving object (for motion blur) or when the camera exhibits rotational motion (for motion blur) during exposure period.

While the point spread function is spatially varying, the variation of the point spread function is spatially smooth. To simplify the problem, in case of the camera motion, many previous works have assumed that the point spread function is spatially invariant, which reduce (1) into a convolution equation:

$$B = I \otimes k \quad (2)$$

where \otimes is the convolution operator. Recently, [1, 2] generalized (2) and proposed the projective motion blur model which uses a sequence of homographies to model the camera motion:

$$B(x, y) = \frac{1}{N} \sum_{i=1}^N I(H_i[x \ y \ 1]^t) \quad (3)$$

where H_i is the homography, $[x \ y \ 1]$ is the homogeneous coordinate of a pixel at (x, y) , and N is the number of homography used to approximate the camera motion in discrete domain. Note that when there is image noise, an additional noise variable will be appended in (1)–(3) under the assumption that noise is addition and is independent to the blur process.

Methodology

Image deblurring is a long-standing problem. Many existing works targeting image blur due to camera shake motion has assumed a globally uniform point spread function which can be described by (2). Traditional non-blind deblurring algorithms include the well-known Richardson-Lucy algorithm [3, 4] and Wiener filter [5] (Refers to the entry on Richardson-Lucy Deconvolution and Wiener Filter for details.).

Due to poor blur kernel estimation or unrecoverable frequency loss from convolution, undesirable artifacts such as ringing and amplification of image noise can be introduced in deblurred results. The state-of-the-art approaches deal with these artifacts by including different image priors to regularize the point spread function estimation process, the latent image restoration process, or both. Representative techniques include [6–14]. For example, Chan and Wong [6] utilized total variation regularization

to help ameliorate ringing and noise artifacts. Fergus et al. [7] demonstrated how to use a variational Bayesian approach combined with gradient-domain statistics to estimate a more accurate PSF. Raskar et al. [15, 16] coded the exposure to make the PSF more suitable for deconvolution. Jia [17] demonstrated how to use an object’s alpha matte to better compute the PSF. Levin et al. [8] introduced a gradient sparsity prior to regularize results for images exhibiting defocus blur. This prior is also applicable to motion blurred images. Yuan et al. [11] proposed a multiscale approach to progressively recover blurred details while Shan et al. [12] introduced regularization based on higher-order partial derivatives to reduce image artifacts. Cho and Lee [13] proposed edge prediction step with GPU implementation to achieve almost real-time deblurring algorithm. Xu and Jia [14] further studied the scale of edges and proposed the edge selection step with the TV- L_1 deconvolution algorithm. A study on the blind motion deblurring algorithms for uniform point spread function can be found in [18].

As mentioned in [18], camera ego motion causes a spatially varying motion blur that cannot be accurately modeled with a uniform PSF. Prior work has recognized the need to handle nonuniform motion blur for camera ego motion, moving objects, and defocus blur. For example, early work by Sawchuk [19] addressed motion blur from a moving camera by first using a log-polar transform to transform the image such that the blur could be expressed as a spatially invariant PSF. The range of motion that could be addressed was limited to rotation and translation. Similarly, Shan et al. [20] handled spatially varying blur by restricting the relative motion between the camera and the scene to be a global in-plane rotation. Dai and Wu [21] used alpha matte to estimate spatially varying motion PSF.

When addressing moving objects, the input image can be segmented into multiple regions each with a constant PSF as demonstrated by Levin [22], Bardsley et al. [23], Cho et al. [24], and Li et al. [25]. Such segmented regions, however, should be small to make the constant PSF assumption valid for the spatially varying motion blur in camera shake motion. For example, Tai et al. [26] extended the hybrid camera framework used by Ben-Ezra and Nayar [27] to estimate a PSF per pixel using an auxiliary video camera.

Some recent techniques have tried to estimate the motion point spread function using the representation

in (3) for deblurring. Tai et al. [28] use coded exposure with user scribbles to estimate the point spread function through homography alignment; Li et al. [29] use video sequences and constraint the transformation from one frame to another frame following a global homography transformation and Whyte et al. [2] limited the space of parameters with only rotational motions and use the variational Bayesian framework proposed by Fergus [7] to estimate the point spread function. Joshi et al. [30] directly measure the point spread function of the camera motion using motion inertia sensor. Detailed deblurring algorithm for the representation in (3) can be found in [1].

While there have been a lot of works targeting deblurring, this problem is still an open problem. As mentioned previously, solving (1–3) is mathematically ill-posed. In addition, many common photographic effects, such as saturation, noise, nonlinear camera response function and compression artifacts, were not probably modeled in (1–3) which limits the practicality of existing deblurring algorithms. There are also studies, such as [31], on the trade-off between deblurring and denoising in high dynamic range imaging.

Application

The main application of deblurring is on image restoration and enhancement. Since motion blur and defocus blur are common artifacts in imaging system, its applications range from astronomy telescope, to satellite imaging, to medical imaging, and to common customer level camera. A method to remove motion blur/defocus blur from a captured photograph is valuable for digital photography. Not only the blurred image information can be revealed, but the photos which were taken at important moment can also be recovered.

References

- Tai YW, Tan P, Brown M (2011) Richardson-lucy deblurring for scenes under a projective motion path. *IEEE Trans Pattern Anal Mach Intell* 33(8):1603–1618
- Whyte O, Sivic J, Zisserman A, Ponce J (2010) Non-uniform deblurring for shaken images. *IEEE conference on computer vision pattern recognition (CVPR)*, San Francisco, California
- Richardson W (1972) Bayesian-based iterative method of image restoration. *J Opt Soc Am* 62(1):55–59
- Lucy L (1974) An iterative technique for the rectification of observed distributions. *Astron J* 79(6):745–754

5. Wiener N (1949) Extrapolation, interpolation, and smoothing of stationary time series. Wiley, New York
6. Chan TF, Wong CK (1998) Total variation blind deconvolution. *IEEE Trans Image Process* 7(3):370–375
7. Fergus R, Singh B, Hertzmann A, Roweis ST, Freeman WT (2006) Removing camera shake from a single photograph. *ACM Trans Graph* 25(3):787–794
8. Levin A, Fergus R, Durand F, Freeman WT (2007) Image and depth from a conventional camera with a coded aperture. *ACM Trans Graph* 26(3):70
9. Yuan L, Sun J, Quan L, Shum H (2007) Image deblurring with blurred/noisy image pairs. *ACM Trans Graph* 26(3):1
10. Chen J, Tang CK (2008) Robust dual motion deblurring. IEEE conference on computer vision pattern recognition (CVPR), Anchorage, Alaska
11. Yuan L, Sun J, Quan L, Shum HY (2008) Progressive inter-scale and intra-scale non-blind image deconvolution. *ACM Trans Graph* 27(3):74
12. Shan Q, Jia J, Agarwala A (2008) High-quality motion deblurring from a single image. *ACM Trans Graph* 27(3):73
13. Cho S, Lee S (2009) Fast motion deblurring. ACM SIGGRAPH Asia 28(5):145
14. Xu L, Jia J (2010) Two-phase kernel estimation for robust motion deblurring. European conference on computer vision (ECCV), Crete, Greece
15. Raskar R, Agrawal A, Tumblin J (2006) Coded exposure photography: motion deblurring using fluttered shutter. *ACM Trans Graph* 25(3):795–804
16. Agrawal A, Raskar R (2007) Resolving objects at higher resolution from a single motion-blurred image. IEEE conference on computer vision pattern recognition (CVPR), Minneapolis, Minnesota
17. Jia J (2007) Single image motion deblurring using transparency. IEEE conference on computer vision pattern recognition (CVPR), Minneapolis, Minnesota
18. Levin A, Weiss Y, Durand F, Freeman W (2009) Understanding and evaluating blind deconvolution algorithms. IEEE conference on computer vision pattern recognition (CVPR), Miami, Florida
19. Sawchuk AA (1974) Space-variant image restoration by coordinate transformations. *J Opt Soc Am* 64(2):138–144
20. Shan Q, Xiong W, Jia J (2007) Rotational motion deblurring of a rigid object from a single image. ICCV, Rio de Janeiro, Brazil
21. Dai S, Wu Y (2008) Motion from blur. IEEE conference on computer vision pattern recognition (CVPR), Anchorage, Alaska
22. Levin A (2006) Blind motion deblurring using image statistics. NIPS, pp 841–848
23. Bardsley J, Jefferies S, Nagy J, Plemmons R (2006) Blind iterative restoration of images with spatially-varying blur. *Opt Express* 1767–1782
24. Cho S, Matsushita Y, Lee S (2007) Removing non-uniform motion blur from images. ICCV, Rio de Janeiro, Brazil
25. Li F, Yu J, Chai J (2008) A hybrid camera for motion deblurring and depth map super-resolution. IEEE conference on computer vision pattern recognition (CVPR), Anchorage, Alaska
26. Tai YW, Du H, Brown MS, Lin S (2010) Correction of spatially varying image and video motion blur using a hybrid camera. *IEEE Trans Pattern Anal Mach Intell* 32(6):1012–1028
27. Ben-Ezra M, Nayar S (2004) Motion-based motion Deblurring. *IEEE Trans PAMI* 26(6):689–698
28. Tai Y, Kong N, Lin S, Shin S (2010) Coded exposure imaging for projective motion deblurring. IEEE conference on computer vision pattern recognition (CVPR), San Francisco, California
29. Li Y, Kang SB, Joshi N, Seitz S, Huttenlocher D (2010) Generating sharp panoramas from motion-blurred videos. IEEE conference on computer vision pattern recognition (CVPR), San Francisco, California
30. Joshi N, Kang S, Zitnick L, Szeliski R (2010) Image deblurring with inertial measurement sensors. *ACM Trans Graph* 29(3):30
31. Zhang L, Deshpande A, Chen X (2010) Denoising vs. deblurring: Hdr imaging techniques using moving cameras. IEEE conference on computer vision pattern recognition (CVPR), San Francisco, California

Deconvolution

- [Blind Deconvolution](#)
- [Deblurring](#)

Defocus Blur

Neel Joshi
Microsoft Corporation, Redmond, WA, USA

Synonyms

- [Out of focus blur](#)

Related Concepts

- [Blur Estimation](#); ► [Motion Blur](#)

Definition

Defocus blur is a loss of sharpness that occurs due to integrating light over an aperture with a nonzero area, where the light source is off of the image focal plane. The amount of blur that is visible in an image is a function of the lens aperture, the object and focal depth, and the camera pixel (or grain) size.

Background

Image blur can be described by a point spread function (PSF). A PSF models how an imaging system captures a single point in the world – it literally describes how a point spreads across an image. An entire image is then made up of a sum of the individual images of every scene point, where each point’s image is affected by the PSF associated with that point. For an image to be “in focus” means that one ideally does not want any image blur at a particular depth of the scene. Thus, the PSF should be minimal, i.e., a delta function, where each scene point should correspond only to one image point. In practice, PSFs can take on a range of shapes and sizes depending on the properties of an imaging system. When this PSF is large relative to image resolution, an image with defocus blur is captured.

The fundamental cause of defocus blur is that a camera does not point-sample a scene, but instead captures images by integrating the light over fixed areas. Defocus blur is a function of aperture size and the relative position between camera and objects in the imaged scene. For very small apertures or “pinhole”-sized apertures, the blur or PSF can be insignificant relative to the resolution or pixel size of the camera. However, in light-limited situations or when a particular photographic effect is desired, such as shallow depth of field in a portrait photo, larger apertures are used. With a large aperture, the depth of field is shallow, and the PSF for objects off the focal plane is larger. The amount of blur is depth dependent: it depends on the focal length of the lens and the focal depth, and it grows with distance from the focal plane, as illustrated in Fig. 1.

Theory

Image blur is described by a point spread function (PSF). The PSF models how an imaging system captures a single point in the world.

The most commonly used model for blur is the a linear model, where the blurred image b is represented as a convolution of a kernel k , plus noise:

$$b = i \otimes k + n, \quad (1)$$

where $n \sim \mathcal{N}(0, \sigma^2)$, which represents and additive Gaussian noise model. In this model, the blur is

assumed to be constant over the entire image, i.e., spatially invariant; however, that is often not true in practice [1, 2]. If there is depth variation in the scene, the blur will change with that depth. Similarly, in many lenses, the shape of the blur changes across the image plane, as illustrated in Fig. 1. In both of these cases, one can think of the blur kernel as being a function of image position, i.e., $k(x, y)$.

To model spatially varying blur, the spatially invariant kernel and convolution in (Eq. 1) can be replaced by a sparse re-sampling matrix that models the spatially variant blur, and the convolution process is now a matrix-vector product:

$$b = Ai + i. \quad (2)$$

Each column of A is the unraveled kernel for the pixel represented by that column. Thus, the blurred response at a pixel in the observed image is computed as a weighted sum, as governed by A , of the latent sharp image i formed into a column vector.

Representation

There are several common assumptions made on the form of a defocus blur kernel:

- The PSF is positive, i.e., all values in the kernel are nonnegative.
- The PSF is energy conserving, i.e., $\sum_i k_i = 1$.
- The PSF is symmetric – radially or along some cartesian axis.
- The PSF has a known parametric form.

The assumptions are listed in order from least to most restrictive. Positivity is a strong constraint, and the least restrictive in that does not eliminate any truly valid kernels, i.e., no true blur kernel can have negative values as blurring is a purely additive process. Another way of thinking of this is that there is no “negative” light. Similarly the second constraint is equally not restrictive in that blurring does not remove light; thus, all true blur kernels should be energy conserving. Thus, the assumptions of positivity and energy conservation are ones that can be used by virtually all PSF models. In practice, whether a particular model uses them depends on the nature of the other assumptions.

The second two assumptions are much more restrictive. Symmetry is typically used when one wants to generalize a 2D PSF from some 1D cross section [3]. Assuming a parametric form is also very restrictive as



Defocus Blur, Fig. 1 With defocus blur, the amount of blur is depth dependent; it depends on the focal length of the lens and the focal depth, and it grows with distance from the focal plane.

it assumes the entire shape of the blur kernel can be modeled by a low parameter mathematical model.

Defocus blur has two commonly used parametric models. A circular disk or “pillbox” function [4]:

$$k(x, y) = \begin{cases} 0 & \sqrt{x^2 + y^2} > r \\ \frac{1}{\pi r^2} & \sqrt{x^2 + y^2} \leq r \end{cases} \quad (3)$$

and a circularly symmetric 2D Gaussian [5]:

$$K(x, y) = C \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right), \quad (4)$$

where C is a normalization constant. In both cases, a single parameter determines the PSF – r for the pillbox and σ for the 2D Gaussian.

Application

Estimation of defocus blur constitutes an extensively researched area [4, 6–9]. Estimated blur kernels are typically used for improving image quality by reducing blur using image deblurring and deconvolution methods [1, 10–12]. Models of defocus blur are also used in depth estimation methods such as depth from defocus [11, 13, 14].

References

1. Joshi N, Szeliski R, Kriegman DJ (2008) Psf estimation using sharp edge prediction. In: Computer vision and pattern recognition 2008. IEEE conference on computer vision pattern recognition (CVPR 2008). IEEE conference on, Anchorage, pp 1–8
2. Levin A, Weiss Y, Durand F, Freeman W (2009) Understanding and evaluating blind deconvolution algorithms. In: Computer vision and pattern recognition 2009. IEEE conference on computer vision pattern recognition (CVPR 2009). IEEE conference on, Miami (Beach), IEEE Computer Society, pp 1964–1971
3. Yoon J, Shin J, Paik JK (2001) Enhancement of out-of-focus images using fusion-based psf estimation and restoration. In: VCIP, San Jose, pp 819–829
4. Cannon M (1976) Blind deconvolution of spatially invariant image blurs with phase. IEEE Trans Acoust Speech Signal Process 24(1):58–63 (see also IEEE Trans Signal Process)
5. Banham MR, Katsaggelos AK (1997) Digital image restoration. IEEE Signal Process Mag 14(2):24–41
6. Gennery DB (1973) Determination of optical transfer function by inspection of frequency-domain plot. J Opt Soc Am 63(12):1571
7. Chang MM, Tekalp AEA (1991) Blur identification using the bispectrum. IEEE Trans Signal Process 39(10):2323–2325 (see also Acoust Speech Signal Process IEEE Trans)
8. Savakis A, Trussell H (1993) Blur identification by residual spectral matching. IEEE Trans Image Process 2(2): 141–151
9. Roisman F, Philips WR, Portilla J (2004) Parametric PSF estimation via sparseness maximization in the wavelet domain. In: Truchetet F, Laligant O (eds) Wavelet applications in industrial processing II. Proceedings of the SPIE, Vol 5607, SPIE, Bellingham, pp 26–33. Volume 5607 of presented at the society of photo-optical instrumentation engineers (SPIE) conference, pp 26–33
10. Richardson WH (1972) Bayesian-based iterative method of image restoration. J Opt Soc Am (1917–1983) 62:55–59
11. Levin A, Fergus R, Durand F, Freeman WT (2007) Image and depth from a conventional camera with a coded aperture. In: SIGGRAPH '07: ACM SIGGRAPH 2007 papers, ACM, New York, p 70
12. Joshi N, Matusik W, Adelson EH, Kriegman DJ (2010) Personal photo enhancement using example images. ACM Trans Graph 29:12:1–12:15
13. Nayar S, Nakagawa Y (1990) Shape from focus: an effective approach for rough surfaces. In: International conference on robotics and automation, vol 1, Cincinnati, pp 218–225
14. Nayar S, Nakagawa Y (1994) Shape from focus. IEEE Trans Pattern Anal Mach Intell 16(8):824–831

Dehazing

► [Descattering](#)

Dehazing and Defogging

Robby T. Tan

Department of Information and Computing Sciences,
Utrecht University, Utrecht, CH, The Netherlands

Synonyms

[Visibility enhancement in bad weather](#)

Related Concepts

► [Descattering](#)

Definition

Dehazing is a process to visually improve degraded visibility caused by atmospheric conditions where the horizontal visibility on the ground level is greater than 1 km. Defogging is a similar process yet focusing on fog, which, unlike haze, is a cloud of water droplets near ground level reducing the horizontal visibility to less than 1 km. Both dehazing and defogging are part of algorithms to enhance visibility in bad weather due to light being scattered and absorbed by atmospheric particles.

Background

Poor visibility in outdoor scenes generates significant problems for many applications of computer vision. Most automatic systems for surveillance, intelligent vehicles, object recognition, etc., assume the input images have clear visibility. Unfortunately, this assumption does not always hold, particularly in bad weather. Therefore, improving the degraded visibility is practically important.

In outdoor scenes, poor visibility is caused by the substantial presence of various atmospheric particles that have significant density in the participating medium (creating haze, fog, mist, smoke, dust, etc.). Light from the atmosphere and light reflected from objects are absorbed and scattered by those particles, causing the visibility of a scene to be degraded. Physically, fog is a cloud of water droplets near ground level that reduces the horizontal visibility to less than 1 km [1]. According to [2], the particle radius of the water droplets is between 1 and $10 \mu\text{m}$ with particle concentration between 100 and 10 cm^{-3} . Unlike fog, haze is not water droplets. It is composed of dry particles (of dust or salt) so small that they cannot be felt or seen individually with the naked eye, but the aggregate reduces horizontal visibility and gives the atmosphere an opalescent appearance (a bluish or yellowish veil depending on whether the background is dark or light, respectively) [1]. In hazy scenes, the horizontal visibility on the ground level is greater than 1 km but typically less than 5 km [3]. The particle radius is between 10^{-2} and $1 \mu\text{m}$ with particle concentration between 10^3 and 10 cm^{-3} [2].

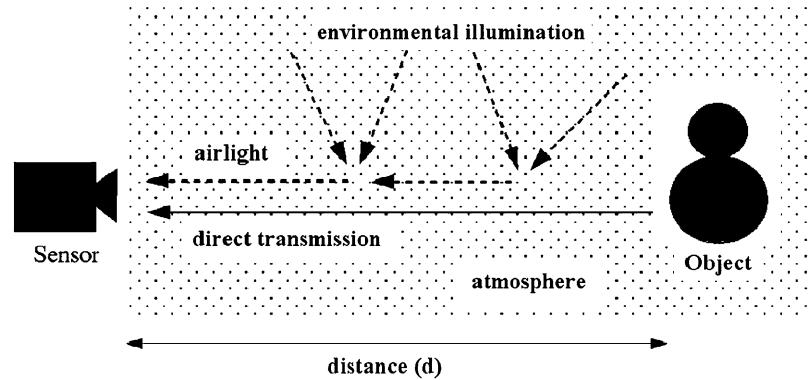
Aside from fog and haze, there is also mist. In terms of visibility, mist is similar to haze (the horizontal visibility on the ground level is greater than 1 km but less than 5 km); however, the particle type is similar to that of fog. With respect to the opalescent appearance, mist can be discriminated from haze, since it gives a grayish cast to the sky [1].

Theory

In the field of optics and atmospheric sciences, there are a number of scattering models depending on the size of the particles and the complexity of the formulas. Most of them are derived from radiative transfer, i.e., the physical phenomenon of energy transfer in the form of electromagnetic radiation, which is affected by absorption, emission, and scattering processes [4]. Most of these models are rather complex. Hence, an approximated optical model for both haze and fog is more commonly used in computer vision (e.g., [5–7]), which is based on Lambert-Beer Law [8] and Koschmieder's equation [9]:

$$\mathbf{I}(x) = \mathbf{L}_1(x)\rho(x)e^{-\beta d(x)} + \mathbf{L}_2(x)(1 - e^{-\beta d(x)}). \quad (1)$$

Dehazing and Defogging, Fig. 1 The pictorial description of the optical model in Eq. (1)



Dehazing and Defogging, Fig. 2 Left: an image plagued by atmospheric particles. Right: the result of enhancing visibility using the method introduced in Tan [16]

The first term is the direct attenuation (\mathbf{D}), and the second term is the airlight (\mathbf{A}):

$$\mathbf{D}(x) = \mathbf{L}_1(x)\rho(x)e^{-\beta d(x)} \quad (2)$$

$$\mathbf{A}(x) = \mathbf{L}_2(x)(1 - e^{-\beta d(x)}), \quad (3)$$

where \mathbf{I} is the image intensity, x is the 2D spatial location, and ρ is the reflectance of an object that appears in the image. Here \mathbf{L}_1 is the irradiance on the object, while \mathbf{L}_2 is the radiance scattered by the medium to the camera without reaching the objects. A further

assumption sometimes made is that $\mathbf{L}_1 = \mathbf{L}_2$ and both of them represent the intensity of the infinitely distant light sources and thus are assumed to be globally constant. The parameter β is the atmospheric attenuation coefficient, while d is the distance between an object in the image and the camera. Note that $\mathbf{I}, \mathbf{L}_1, \mathbf{L}_2, \rho$ in the equation are color dependent.

Equation (2) is in principle the Lambert-Beer law [8] for transparent objects. Equation (1) is basically Koschmieder's equation [9] that takes into account only single scattering towards the camera. Consequently, the model fails to capture the effects of

highly scattering media. In terms of degree of polarization (DOP), the airlight \mathbf{A} is often partially polarized. Hence, two polarization components of \mathbf{A} can be obtained using a polarizing filter attached to the camera. These components are parallel and perpendicular to the plane of incidence [6, 10, 11]. Figure 1 illustrates the model.

Problem Definition

Based on Eq.(1), the problem can be generally described as follows: Given an image or a sequence of images whose intensity is represented by $\mathbf{I}(x)$, estimate $\mathbf{R}(x)$ for every pixel, where $\mathbf{R}(x) = \mathbf{L}_1\boldsymbol{\rho}(x)$ is the scene radiance had there not been particles along the line of sight.

Possible Assumptions and Solutions

For a single image, the problem described above is ill-posed. In the literature, a few approaches trying to tackle the problem have been proposed. The first approach is based on polarizing filters (e.g., [6, 10, 11]). The main idea of this approach is to exploit two or more images of the same scene obtained by rotating a polarizing filter attached to the camera. Having at least two different intensity values of each pixel of the scene, [11] transforms the problem into a form solvable using ICA. The main assumption is that the scene is static while the filter rotates.

Another approach is based on multiple images of a scene taken in different weather conditions, i.e., with different properties of the participating medium (e.g., [5, 7]). By deriving two equations from Eq.(1), which represent the two input images, [7] can obtain the absolute values of airlight by assuming the presence of a totally dark object in the scene.

Different approaches proposed by Tan and Oakley [12], Narasimhan and Nayar [13], Hautière et al. [14], and Kopf et al. [15] are based on a single image, yet they require geometric information about the input scene. This requirement might be impractical, and thus methods in [16–20] use a single image and rely on various statistical image priors. Many methods [5, 7, 16–19] assume that β in Eq.(1) is constant for across the light spectrum. This assumption is reasonable in fog (but away from fogbows and rainbows) [2]. Figure 2 shows the result of using the method of [16].

Open Problems

Despite the development in the recent years, one problem still remains, i.e., the inaccuracy of the model in Eq.(1) with regard to multiple scattering. The model is a single scattering model and thus cannot capture significant scattering effects (e.g., blur) that can occur particularly in dense foggy scenes.

References

1. Encyclopedia-britannica. <http://www.britannica.com>
2. McCartney EJ (1976) Optics of the atmosphere: scattering by molecules and particles. Wiley, New York
3. Kumar B, De Remer D, Marshall DM (2005) An illustrated dictionary of aviation. McGraw-Hill, New York
4. Chandrasekhar S (1960) Radiative transfer. Dover, New York
5. Nayar SK, Narasimhan SG (1999) Vision in bad weather. In: The proceedings of the seventh IEEE international conference on computer vision, Kerkyra, vol 2. IEEE, pp 820–827
6. Schechner YY, Narasimhan SG, Nayar SK (2001) Instant dehazing of images using polarization. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), Kauai, vol 1. IEEE, pp 325–332
7. Narasimhan SG, Nayar SK (2003) Contrast restoration of weather degraded images. IEEE Trans Pattern Anal Mach Intell 25(6):713–724
8. Beer A (1852) Bestimmung der absorption des rothen lichts in farbigen flüssigkeiten. Ann Phys Chem 86(2):78–90
9. Koschmieder H (1924) Theorie der horizontalen sichtweite. eitr. Phys. Freien Atm 12(2):33–53
10. Shwartz S, Namer E, Schechner YY (2006) Blind haze separation. In: IEEE computer society conference on computer vision and pattern recognition (CVPR), New York, vol 2. IEEE, pp 1984–1991
11. Treibitz T, Schechner YY (2009) Polarization: beneficial for visibility enhancement? In: IEEE conference on computer vision and pattern recognition (CVPR), Miami. IEEE, pp 525–532
12. Tan K, Oakley JP (2001) Physics-based approach to color image enhancement in poor visibility conditions. JOSA A 18(10):2460–2467
13. Narasimhan SG, Nayar SK (2003) Interactive (de) weathering of an image using physical models. In: IEEE workshop on color and photometric methods in computer vision, Nice
14. Hautière N, Tarel JP, Aubert D (2007) Towards fog-free in-vehicle vision systems through contrast restoration. In: IEEE conference on computer vision and pattern recognition (CVPR), Minneapolis. IEEE, pp 1–8
15. Kopf J, Neubert B, Chen B, Cohen M, Cohen-Or D, Deussen O, Uyttendaele M, Lischinski D (2008) Deep photo: model-based photograph enhancement and viewing. ACM Trans Graph (TOG) 27:116
16. Tan RT (2008) Visibility in bad weather from a single image. IEEE conference on computer vision and pattern recognition (CVPR), Anchorage. IEEE, pp 1–8

17. Fattal R (2008) Single image dehazing. ACM Trans Graph (TOG) 27:72
18. He K, Sun J, Tang X (2009) Single image haze removal using dark channel prior. In: IEEE conference on computer vision and pattern recognition (CVPR), Miami. IEEE, pp 1956–1963
19. Tarel JP, Hautiere N (2009) Fast visibility restoration from a single color or gray level image. In: IEEE 12th international conference on computer vision, Kyoto. IEEE, pp 2201–2208
20. Kratz L, Nishino K (2009) Factorizing scene albedo and depth from a single foggy image. In: IEEE 12th international conference on computer vision, Kyoto. IEEE, pp 1701–1708

Denoising

Francisco J. Estrada

Department of Computer and Mathematical Sciences,
University of Toronto at Scarborough, Toronto,
ON, Canada

Synonyms

[Noise removal](#)

Related Concepts

► [Image Enhancement and Restoration](#)

Definition

Denoising is the process of recovering a reference signal that has been corrupted by noise. In computer vision, the reference signal is typically assumed to be the undistorted image of an object or scene, and noise is introduced as a result of the imaging process. The amount and type of noise changes from application to application. An example of a typical noisy image and the result of performing image denoising on it are shown in [Fig. 1](#).

Background

Noise is an unavoidable consequence of the imaging process. Sources of noise include measurement and quantization errors introduced during signal acquisition and processing, thermal noise from the sensor and

electronics in digital imaging systems, photographic grain in the case of film, and the physical nature of light itself.

The importance of denoising stems from the fact that noise in the input image can adversely affect the result of all subsequent visual processing. The value of image-dependent quantities such as brightness gradients, the accuracy in the localization of image features, the presence or absence of object boundaries, and even the subjective perceptual properties of the image are all affected to some degree by noise. Applications such as medical imaging, astronomy, low-light or high-speed photography, and synthetic aperture radar (SAR) imaging are typically characterized by larger amounts of noise relative to the reference signal.

Depending on the application, preprocessing the input with a denoising algorithm can improve the results obtained from further stages of visual processing. However, the denoising method should be selected with care since the denoising process is imperfect and will invariably destroy part of the information contained in the reference signal.

Theory

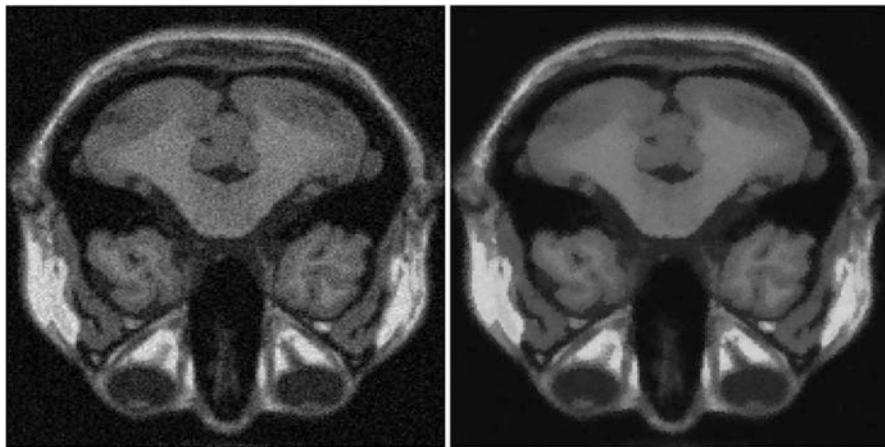
For a grayscale image, the perceived image brightness can be approximated by [6]

$$I = f(L + n_s + n_c) + n_q, \quad (1)$$

where I is the observed brightness value, $f(\cdot)$ is the sensor response function, L is the image irradiance, n_s represents the noise contribution from brightness-dependent sources, n_c represents a constant noise factor, and n_q is the quantization noise.

Estimating all the quantities in this model is too complex a problem if only I is known, so in practice the simplified relationship $I = I_r + N$ is used instead where I is the observed image brightness, I_r is reference image that the denoising process must estimate, and N is the noise component. Since there is only one known value (the observed brightness) and two unknowns, the problem is under-constrained.

Additional external constraints must be placed on the properties of I_r and N so that a solution can be computed. Typically N is assumed to be i.i.d. noise, and most denoising methods assume a zero mean



Denoising, Fig. 1 Left original, noisy MRI scan. Right result after denoising

Gaussian distribution with fixed standard deviation. Depending on the constraints placed on the properties of I_r , we can group denoising methods into a handful of major classes.

Classes of Denoising Algorithms

The first class of image denoising methods is based on averaging the value of pixels within small image neighborhoods under the assumption that the brightness away from image edges should be uniform. These algorithms perform anisotropic smoothing, so called because it has strong tendency to smooth uniform-looking regions in the image while preserving strong brightness discontinuities [13]. The original anisotropic smoothing algorithm [7], the bilateral filter [10], methods based on minimizing total variation [2], and the stochastic denoising algorithm [4] are all examples of denoising methods based on maximizing brightness uniformity across homogeneous regions.

A second class of algorithms is based on the analysis of image statistics. The underlying principle is that the statistical properties of natural images can be modelled and that given the model, denoising can be carried out by examining the statistics of a noisy image and transforming the image so that its statistics match those of learned model [11]. Common models include distributions of filter responses and pooled statistics for collections of image patches. Algorithms such as Gaussian scale mixtures [8], fields of experts [9], the

nonlocal means method [1], and the block matching algorithm [3] rely on exploiting the statistical regularities of images or small image patches.

A third class of image denoising methods specifically designed to remove salt and pepper noise is based on outlier detection. The process involves estimating a distribution of brightness values around image neighborhoods and using this distribution to identify and remove outliers. The median filter [5] is an example of this class of denoising algorithms.

Regardless of the method, the goal of denoising is to remove as much of the noise as possible while preserving the information contained in the reference signal. For this reason, denoising algorithms are often evaluated using peak signal to noise ratio (pSNR) or the structured image similarity index (SSIM) [12] on images for which the noise-free reference is known.

Application

Denoising can be a useful preprocessing step for images from domains such as medical imaging, astronomy, or synthetic aperture radar. It is also applicable to digital photography under low illumination conditions and to the restoration of archival footage and photographs. Image editing and image processing programs typically include denoising modules.

Experimental Results

The quality of the results produced by different algorithms changes from image to image and across different domains. But recent benchmarks [14] indicate that the block matching algorithm achieves better performance overall on natural images with different amounts of Gaussian noise.

References

- Buades A, Coll B, Morel J (2008) Nonlocal image and movie denoising. *Int J Comput Vision* 76:123–139
- Chambolle A (2004) An algorithm for total variation minimization and applications. *J Math Imaging Vis* 20:89–97
- Dabov K, Foi R, Katkovnik V, Egiazarian K (2006) Image denoising with block matching and 3D filtering. *SPIE Electronic Imaging* 6064A-30
- Estrada F, Fleet D, Jepson A (2009) Stochastic Image Denoising. *British Machine Vision Conference* (no printed proceedings)
- Juhola M, Katajainen J, Raita T (1991) Comparison of Algorithms for Standard Median Filtering. *IEEE T Signal Proces (TSP)* 39:204–208
- Liu C, Szeliski R, Kang SB, Zitnick L, Freeman WT (2008) Automatic estimation and removal of noise from a single image. *IEEE Trans Pattern Anal Mach Intell (PAMI)* 30(2):299–314
- Perona P, Malik J (1990) Scale-space and edge detection using anisotropic diffusion. *IEEE Trans Pattern Anal Mach Intell (PAMI)* 12(7):629–639
- Portilla J, Strela V, Wainwright M, Simoncelli E (2003) Image denoising using scale mixtures of Gaussians in the wavelet domain. *IEEE T Image Process (TIP)* 12(11): 1338–1351
- Roth S, Black M (2009) Fields of Experts. *Int J Comput Vision (IJCV)* 82(2):205–229
- Tomasi C, Manduchi R (1998) Bilateral filtering for gray and color images. *Int Conf Comput Vision* 839–846
- Torralba A, Oliva A (2003) Statistics of Natural Image Categories. *Network-Comp Neural* 391–412
- Wang Z, Bovik A, Sheikh H, Simoncelli E (2004) Image quality assessment: From error visibility to structural similarity. *IEEE T Image Process (TIP)* 13(4):600–612
- Weickert J (1998) Anisotropic Diffusion in Image Processing. Teubner-Verlag
- Image Denoising Benchmark (2010) <http://www.cs.utoronto.ca/~strider/Denoise/Benchmark>

Dense 3D Modeling

- Dense Reconstruction

Dense Reconstruction

Christopher Zach

Computer Vision and Geometry Group, ETH Zürich,
Zürich, Switzerland

Synonyms

Dense 3D modeling; Multiview stereo

Related Concepts

- Structure-from-Motion (SfM)

Definition

Dense reconstruction aims on determining the complete 3D geometry of a static environment solely from a set of provided images.

Background

Correspondences between images depicting a static scene cannot only be established for a small set of visually salient regions but can be also extended to the entire image domain (i.e., to all pixels). Under the assumption of a static environment (or equivalently, a rigidly moving scene captured by a static camera), corresponding points in images together with known camera calibration induce a 3D point. Thus, a dense set of correspondences implies a densely sampled surface in 3D. Establishing per-pixel correspondences between only two (or any small number of) narrow-baseline images is usually referred as computational stereo. *Dense reconstruction* addresses the more general problem of obtaining the full 3D geometry from a larger collection of images. In contrast to computational stereo, the returned surfaces are not 2.5D depth maps (as in stereo) but can possess arbitrary topology.

Theory

Determining the dense 3D geometry of a static scene from images can be achieved via direct or via indirect approaches:

- Direct approaches estimate the dense 3D geometry by using all available images simultaneously.
- Indirect methods estimate partial 3D models from a subset of (usually narrow-baseline) images (e.g., via computational stereo) and subsequently merge the partial models into a globally consistent geometry.

Direct Methods

Direct methods estimate the likelihood of the 3D surface being at a particular position via computation of a *photo-consistency score* incorporating all given images. Typically the 3D region of interest $\Omega \subseteq \mathbb{R}^3$ is discretized as a voxel space, and photo-consistency is computed with respect to the voxels. Since accurate visibility is not yet known, the photo-consistency score must be robust against potential occlusions. Approximate visibility of a surface patch in the images can be determined e.g., by view frustum and back-face culling, and when a coarse estimate of the 3D geometry (e.g., the corresponding visual hull) is available. By traversing the voxel space in a conservative order with respect to visibility and computing photo-consistency by only using certainly unoccluded images, one obtains the *space carving* method [1]. Sufficiently photo-inconsistent voxels are declared as empty and influence the visibility of not-yet-visited voxels in the images. Under certain assumptions, the set of occupied voxels (which is called the photo hull) contains the true 3D geometry. For objects with little texture, the space carving results are not satisfactory, and spatial smoothness assumptions need to be added. The volumetric graph cut approach [2] can be interpreted as regularized space carving. Since the state of a voxel (empty or occupied) is determined by a global optimization method, a robust photo-consistency score for each voxel is computed using static and approximate visibility information.

The knowledge of sparsely sampled surfaces, e.g., by triangulating correspondences between interest points, can be used to discretize Ω more adaptively by a general tetrahedral mesh. In such a representation, it is possible to employ a higher-resolution discretization where the true surface is expected [3, 4].

The task of determining the 3D geometry consistent with all given images can be formulated as mesh evolution approach, which is inspired, e.g., by deformable contour approaches used in image segmentation. The

evolution of the mesh is guided by external forces based on the photo-consistency and optionally on silhouette data and internal forces regularizing the mesh [5, 6]. One drawback of such methods is that the obtained result may strongly depend on the initial mesh and only local minima of the underlying energy are usually reached.

Indirect Approaches

This set of methods first computes a collection of depth maps from a suitable narrow-baseline subset of images and fuses these depth images into a consistent 3D model. Since knowing the exact visibility of 3D locations in the images is not required for photo-consistency computation in a small baseline setting, these methods do not need a coarse geometry to estimate potential occlusions. Furthermore, sophisticated computational stereo methods are available to generate high-quality depth maps ultimately yielding photo-realistic reconstructions. On the other hand, indirect methods do not utilize the available image data to the full extent, since the original images are usually not considered in the depth map fusion step.

Merging multiple depth maps can be identified as a particular instance of surface fitting from unorganized point data (e.g., [7, 8]), but it is clearly beneficial to exploit the specific structure in the input data. The mathematical formulations employed for depth map fusion can be tracked back to the problem of merging partial range scans obtained by an active sensor device, e.g., laser data or from structured light. Early approaches use an explicit polygonal representation [9], but more current methods are usually based on implicit volumetric surface representations capable of handling surfaces with arbitrary genus. It is a common aspect of all volumetric approaches that they return watertight meshes, i.e., surfaces may be hallucinated in occluded regions. In general, depth map fusion using a volumetric representation can be formalized as determining a minimizer $u : \Omega \rightarrow \{-1, 1\}$ of a functional E ,

$$E(u) = \phi(u; \{d_i\}) + \psi(u),$$

where $\Omega \subset \mathbb{R}^3$ is the region of interest (usually a 3D bounding box), $\phi(u; \{d_i\})$ is a compatibility (data

fidelity) function measuring the agreement of the solution u with the set of input depth maps $\{d_i\}$, and $\psi(\cdot)$ denotes the spatial regularization of u . The interpretation of u is that $u(x) = 1$ if the 3D location x is occupied (solid) and $u(x) = -1$ if x corresponds to empty space. Often u is allowed to attain fraction values, e.g., $u : \Omega \rightarrow [-1, 1]$. In the following, let $D_i : \Omega \rightarrow \mathbb{R}$ be the signed distance transform induced by the depth map d_i , and $F : \Omega \rightarrow \mathbb{R}^3$ is a 3D vector field with $F(x)$ corresponding to a (smoothed) surface normal induced by the sampled depth maps. $F(x) = \mathbf{0}$ if x is distant to any of the depth map surfaces. By appropriate choice of ϕ and ψ , one obtains

- *Volumetric range image integration* [10]
(with $\phi(u; \{d_i\}) = \int_{\Omega} \sum_i (u(x) - D_i(x))^2 dx$ and $\psi \equiv 0$)
- *Poisson surface reconstruction* [11]
(with $\phi(u) = \int \|F - \nabla u\|^2 dx$ and $\psi \equiv 0$)
- *Global shape fitting* [12]
(with $\phi(u) = \int F^T \nabla u dx$ and $\psi(u) = \int \|\nabla u\| dx \doteq TV(u)$)
- *TV-L¹ depth map fusion* [13]
(with $\phi(u; \{d_i\}) = \int_{\Omega} \sum_i |u(x) - D_i(x)| dx$ and $\psi(u) = TV(u)$)

The above mentioned methods use a uniformly sampled voxel space to discretize the domain Ω . Determining a minimizer u for these energies is relatively simple due to their intrinsic convexity and amounts, e.g., to per-voxel averaging [10] or solving a sparse linear system of equations [11]. The final triangular mesh can be extracted from the implicit volumetric representation as the zero-level set, e.g., via the marching cubes method [14].

Application

Dense reconstruction plays a fundamental role in fully automatic 3D content creation from image data. Its application ranges from 3D city modeling and cultural heritage preservation to obstacle avoidance in autonomous systems navigation and 3D face reconstruction for character animation. Dense reconstruction can achieve an accuracy comparable to actively measured geometry (using laser range scanners or structured light) by relying only on passive imaging sensors [15, 16]. Accurate dense scene geometry can be augmented with texture images or more general

appearance properties enabling photo-realistic rendering of virtual 3D representations.

References

1. Kutulakos K, Seitz S (2000) A theory of shape by space carving. *Int J Computer Vision*, 38(3):198–218
2. Vogiatzis G, Hernandez C, Torr P, Cipolla R (2007) Multi-view stereo via volumetric graph-cuts and occlusion robust photo-consistency. *IEEE trans pattern analysis and machine intelligence*, 29(12):2241–2246
3. Sinha S, Mordohai P, Pollefeys M (2007) Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh. *International Conf. Computer Vision*, (ICCV2007), Rio de Janeiro
4. Labatut P, Pons JP, Keriven R (2007) Efficient multi-view reconstruction of large-scale scenes using interest points, Delaunay triangulation and graph cuts, *International Conf. Computer Vision*, (ICCV2007), Rio de Janeiro
5. Faugeras O, Keriven R (1998) Variational principles, surface evolution, PDE's, level set methods, and the stereo problem. *IEEE trans image processing*, 7(3):336–344
6. Hernández-Esteban C, Schmitt F (2004) Silhouette and stereo fusion for 3d object modeling. *CVIU* 96(3):367–392
7. Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W (1992) Surface reconstruction from unorganized points. *ACM SIGGRAPH '92*. Chicago, IL, pp 71–78
8. Amenta N, Choi S, Kolluri R (2001) The power crust. In: *Proceedings of 6th ACM symposium on solid modeling*. Ann Arbor, MI, pp 249–260
9. Turk G, Levoy M (1994) Zippered polygon meshes from range images. *ACM SIGGRAPH '94*. Orlando, FL, pp 311–318
10. Curless B, Levoy M (1996) A volumetric method for building complex models from range images. *ACM SIGGRAPH '96*. New Orleans, LA, pp 303–312
11. Kazhdan M, Bolitho M, Hoppe H (2006) Poisson surface reconstruction. *Symposium on geometry processing in Proc the fourth eurographics symposium on geometry processing*, Aire-la-Ville, Switzerland, pp 61–70
12. Lempitsky V, Boykov Y (2007) Global optimization for shape fitting. *IEEE conf computer vision and pattern recognition (CVPR)*, Minneapolis, Minnesota
13. Zach C, Pock T, Bischof H (2007) A globally optimal algorithm for robust TV-L¹ range image integration. *International Conf. Computer Vision*, Rio de Janeiro
14. Lorenson W, Cline H (1987) Marching Cubes: a high resolution 3d surface construction algorithm. *ACM SIGGRAPH '87*, New York, pp 163–170
15. Seitz S, Curless B, Diebel J, Scharstein D, Szeliski R (2006) A comparison and evaluation of multi-view stereo reconstruction algorithms. *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, New York, pp 519–526
16. Strecha C, von Hansen W, Van Gool L, Fua P, Thoennessen U (2008) On benchmarking camera calibration and multi-view stereo for high resolution imagery. *IEEE Conf Computer Vision and Pattern Recognition (CVPR)*, Anchorage

Depth Distortion

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Related Concepts

► Camera Calibration; ► Perspective Camera; ► Active Stereo Vision

Definition

Depth distortion refers to the distortion on the recovered structure from motion or from stereo due to imprecision or errors in the cameras' intrinsic and extrinsic parameters.

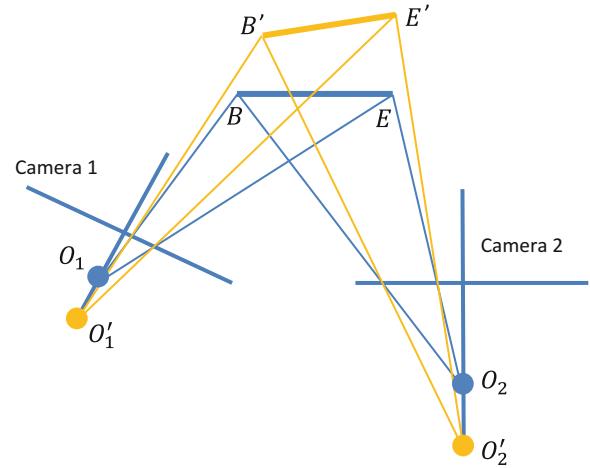
Background

In structure from motion, when a camera's intrinsic parameters are imprecise or even erroneous, the estimated motion is not accurate, and the structure inferred from motion will be a distorted version of the true structure in the environment. In stereo, when the cameras' intrinsic and extrinsic parameters are imprecise or even erroneous, the reconstructed 3D scene will be a distorted version of the true scene in the environment.

Theory

As an example, Fig. 1 illustrates a case where the focal length is misestimated. The top view of two cameras is shown. The line segment BE is the true structure in the environment. It is projected to the images according to the true optical center O_1 and O_2 , respectively. However, if the estimation of the focal length is not accurate, the reconstructed structure will be distorted. In the figure, the optical centers are estimated to be at O'_1 and O'_2 . Then, the reconstructed structure will be $B'E'$, which is clearly different from the true one BE .

In general, given N views of a scene, the 3D reconstruction of a point \mathbf{M} is a function of its image points $\{\mathbf{m}_1, \dots, \mathbf{m}_N\}$, the cameras' intrinsic parameters $\{\mathbf{c}_1, \dots, \mathbf{c}_N\}$, and the cameras' extrinsic parameters $\{\mathbf{p}_1, \dots, \mathbf{p}_N\}$. That is,



Depth Distortion, Fig. 1 Illustration of depth distortion due to imprecise focal length

$$\mathbf{M} = f(\mathbf{m}_1, \dots, \mathbf{m}_N; \mathbf{c}_1, \dots, \mathbf{c}_N; \mathbf{p}_1, \dots, \mathbf{p}_N) . \quad (1)$$

If a parameter, say \mathbf{c}_1 , is misestimated by $\Delta\mathbf{c}_1$, i.e., $\mathbf{c}'_1 = \mathbf{c}_1 + \Delta\mathbf{c}_1$, then the reconstructed 3D point will be

$$\mathbf{M}' = f(\mathbf{m}_1, \dots, \mathbf{m}_N; \mathbf{c}_1 + \Delta\mathbf{c}_1, \dots, \mathbf{c}_N; \mathbf{p}_1, \dots, \mathbf{p}_N) , \quad (2)$$

which is a distorted version of \mathbf{M} and is different from \mathbf{M} .

If the error in the parameter is small, we can use Taylor expansion and ignore the high-order terms. The first-order approximation of the error in 3D reconstruction due to $\Delta\mathbf{c}_1$ will be

$$\begin{aligned} \mathbf{M}' - \mathbf{M} &= \frac{\partial}{\partial \mathbf{c}_1} (f(\mathbf{m}_1, \dots, \mathbf{m}_N; \mathbf{c}_1 \\ &\quad + \Delta\mathbf{c}_1, \dots, \mathbf{c}_N; \mathbf{p}_1, \dots, \mathbf{p}_N)) \\ &\quad \Delta\mathbf{c}_1 \equiv \mathbf{J}_{\mathbf{c}_1} \Delta\mathbf{c}_1 , \end{aligned} \quad (3)$$

where $\mathbf{J}_{\mathbf{c}_1}$ is the Jacobian matrix of $f(\cdot)$ with respect to \mathbf{c}_1 . In practice, the exact error in the parameters is usually unknown. If the error can be modeled as a Gaussian, then the error in 3D reconstruction can also be approximated as a Gaussian. Let the error in \mathbf{c}'_1 , $\Delta\mathbf{c}_1$, be a Gaussian with mean 0 and covariance matrix $\Sigma_{\mathbf{c}_1}$, then the covariance matrix of the 3D reconstruction \mathbf{M}' is given by

$$\Sigma_{\mathbf{M}} = \mathbf{J}_{\mathbf{c}_1} \Sigma_{\mathbf{c}_1} \mathbf{J}_{\mathbf{c}_1}^T . \quad (4)$$

The reader is referred to [1–3] for details.

The impact of the error in a camera's different intrinsic parameters on 3D reconstruction is not necessarily the same in magnitude. For example, the error in the coordinates of the principal point has little effect in 3D reconstruction [4, 5]. Cheong and Peh [6] provides a more detailed discussion on depth distortion due to calibration uncertainty.

References

1. Ayache N (1991) Artificial vision for mobile robots. MIT, Cambridge
2. Zhang Z, Faugeras OD (1992) 3D dynamic scene analysis: a stereo based approach. Springer, Berlin/Heidelberg
3. Faugeras O (1993) Three-dimensional computer vision: a geometric viewpoint. MIT, Cambridge
4. Triggs B (1998) Autocalibration from planar scenes. In: Proceedings of the 5th European conference on computer vision (ECCV), Freiburg, Germany, pp 89–105
5. Zhang Z (1990) Motion analysis from a sequence of stereo frames and its applications. Ph.D. thesis, University of Paris-Sud, Orsay, Paris, France
6. Cheong LF, Peh CH (2004) Depth distortion under calibration uncertainty. Comput Vis Image Underst 93:221–244

Depth Estimation

Reinhard Koch
Institut für Informatik
Christian-Albrechts-Universität, Kiel, Germany

Synonyms

[Depth imaging](#); [Distance estimation](#); [Three dimensional estimation](#)

Related Concepts

► [Plane Sweeping](#)

Definition

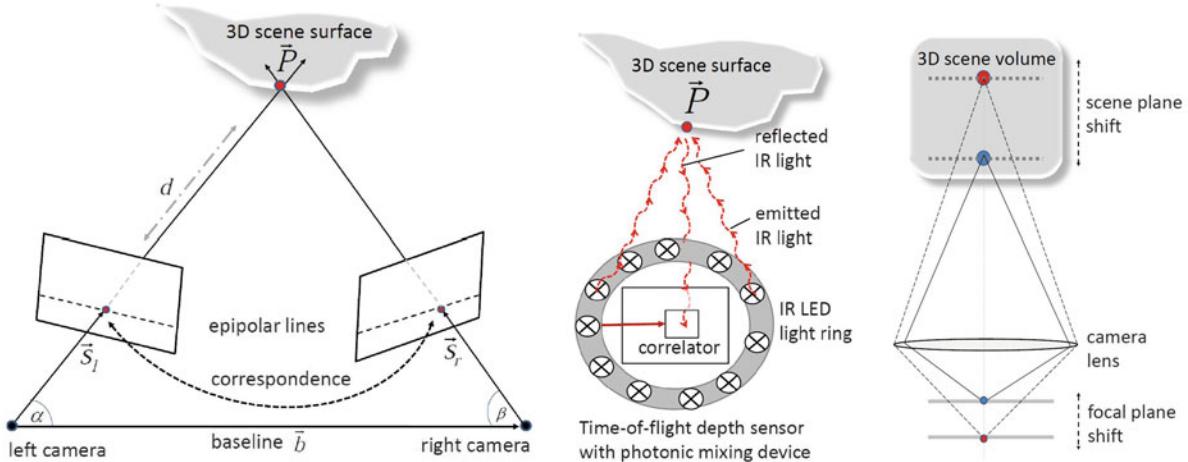
Depth estimation describes the process of measuring or estimating distances from sensor data, typically in a 2D array of depth range data. The sensors may be either optical camera configurations (stereo or

multiview stereo camera rigs), active projector-camera configurations, or active range cameras.

Background

Depth estimation is one of the fundamental computer vision tasks, as it involves the inverse problem of reconstructing the three-dimensional scene structure from two-dimensional projections. Given a 2D image of a 3D scene, the goal of depth estimation is to recover, for every image pixel, the distance from the camera center to the nearest 3D scene point along the pixel's viewing direction. The resulting 2D array of distance values is called the depth map, which is aligned with the camera coordinate system. The challenge of depth estimation is to recover, with sufficient accuracy, the depth map from the given image. Depth estimation can be accomplished in two principle ways: by triangulation from different viewpoints or by coaxial methods along the camera viewing direction. In both cases, estimation methods may be either passive, based on image data alone, or active, with additional sensor control or scene illumination as supporting data. Binocular stereo, for example, is representative for a passive triangulation method, while laser interferometry is a typical active coaxial method.

As the depth map is organized in a regular 2D image grid and contains the depth value for each pixel, it is sometimes also called 2.5D model [11]. The depth map is not a full 3D shape representation, as it contains depth to the nearest visible 3D scene elements only and does not handle regions that are occluded from the camera's view point. Extensions to layered depth images (LDI) exist that can handle multiple depth values for occluded scene elements [16], but LDIs are still restricted to the camera viewpoint and do not handle full 3D data. Full 3D surface estimation from arbitrary view points is the topic of 3D modeling of surfaces and volumetric 3D reconstruction methods, which can be obtained by merging multiple 2.5D depth maps, or by tomographic methods that actively scan a volume with multiple projections and reconstruct the 3D volume surface density. For tomographic methods, which are common in 3D medical and material science volume analysis, and for 3D modeling, refer to the related concepts. A good overview and recent bibliography on all related concepts to depth estimation can be found in [17].



Depth Estimation, Fig. 1 Depth estimation principles. *Left:* depth estimation by stereo triangulation of corresponding projections. *Center:* active time-of-flight depth estimation by phase shift correlation. *Right:* coaxial depth estimation by focal plane shift

Theory

Triangulation

Triangulation is the most common approach for depth estimation. At least two images from different view points observe the same 3D scene, and a 3D scene point is triangulated from the observed 2D projections of the scene point by ray intersection of the viewing rays; see Fig. 1 (left). The depth d is the length of the viewing ray from camera center to ray intersection. It can be computed easily from the triangulation triangle, given the length between the camera centers (base line b) and the two angles α and β between the viewing rays and the base line. This requires that the cameras are calibrated w.r.t. each other in a calibrated stereo rig configuration, and that the corresponding 2D projections of the observed 3D scene point can be related to each other. The latter causes most of the problems in depth estimation, because the 2D search of corresponding projection pairs in the 2D images is difficult.

The correspondence problem can be simplified to a one-dimensional search when the relative orientation between the cameras, the epipolar geometry, is utilized [7]. The spatial triangulation plane or epipolar plane, which is constructed by the camera base line b and the viewing ray s_l of one camera, intersects the image plane of the other camera in a line, the epipolar line. Hence, the correspondence search is

confined to this line and called disparity estimation. Disparity estimation is further simplified if the epipolar line corresponds to horizontal image scan lines. This is the case in standard stereo geometry where both cameras are aligned in identical orientation and shifted in horizontal scanline direction only. Stereo cameras with convergent configurations can be rectified to a virtual alignment by a rectifying homography [1] or with other, more general transformations [13]. A generalization to rectification is the plane sweep, where the images of multiple calibrated cameras are compared by projecting onto a common 3D reference plane. Shifting the plane throughout the depth volume allows multi-camera depth estimation [4].

Passive Correspondence Analysis

Passive correspondence estimation relies on image data alone and requires that the image contains sufficient intensity variations to obtain a unique correspondence match. Correspondences are found by evaluating a cost function of the local intensity data along the search range. There exist a large variety of cost functions and evaluation schemes, ranging from data-driven evaluation of local cost function minimization [6, 18], over semiglobal optimization along epipolar lines [5, 8], to global optimization schemes including smoothness constraints [3]. A good source for algorithms and benchmarking is the Middlebury database and their evaluation test scenario [14].

Active Correspondence Analysis

The main drawback of image-based correspondence analysis is the dependency on sufficient intensity variation in the images. One way out is to replace one of the cameras with an active video projector and to project unique patterns, which are easily identified by the other cameras, into the scene. There exist a variety of approaches. Coded light projectors project a time series of deterministic codes for unique correspondence, or a statistically distributed pattern is projected for optimal correspondence search. The patterns might be projected either in visible or in near infrared light range. Stripe projection systems generate stripes of spectrally colored light [2], or even single laser light stripes are used for easy identification in industrial depth inspection systems. One recent and very successful variant of an active depth triangulation system is the Kinect (a trademark of Microsoft) sensor device that couples an infrared pattern projector and corresponding infrared camera for fast depth estimation with a visual color camera for simultaneous color and depth capture. Such systems deliver reliable dense depth maps for many scenes, but are restricted in depth range to a few meters. Depth estimation in the longer range may be obtained by triangulating laser scanning devices that scan the scene with a rotating mirror laser reflected into the sensing camera.

Coaxial Depth Estimation

One drawback of the triangulation method is that the scene is observed from different viewpoints, and hence, occlusion might occur at object boundaries. If the baseline between the cameras is small, then fewer occlusion boundaries occur, but the depth measurement is not very accurate due to the small triangulation angle. If, on the other hand, a large baseline is chosen, then the triangulation may be accurate but the correspondence problem is more difficult due to the changing perspective. Coaxial methods do not suffer from such problems, as they measure depth from one viewpoint only. Active coaxial depth estimation has its foundations in classical interferometry systems. A coherent laser beam is sent out, reflected back at the 3D scene point, and received at the sensor. The distance the light wave has traveled causes a phase shift between emitted and returning light and hence interference. Due to the short wave length of

light, this principle is very accurate but covers only a very small unambiguous depth range. Recently, a novel class of depth sensors has emerged that either exploits the traveling time of a very short light impulse to measure the time of flight directly, or that modulates an infrared LED light source with a periodic signal amplitude and measures the signal phase shift by correlation between outgoing and reflected light. The correlation sensor is a 2D image array of photonic correlation mixers that allows to directly estimate depth in a range of meters, depending on the modulation frequency; see Fig. 1 (center). Typical depth ranges are up to 10 m. Such time-of-flight range cameras [15] are active coaxial depth estimation devices that in principle may deliver dense and accurate depth maps in real-time and can be used for depth estimation of dynamic time-varying scenes [9, 10].

Another option to estimate depth from a coaxial view is to change camera parameters and to relate the observed image changes to depth. This principle is observed by depth from focus algorithms. A series of images with different focal settings and aperture is recorded, and depth is computed from image sharpness variations [12]. This principle has been applied very successfully in depth-of-field microscopy, where a 3D volume of semitransparent objects like biological cells is scanned with shifting focal planes and very narrow depth-of-field settings; see Fig. 1 (right). Thus, a 3D stack of slices is obtained for 3D analysis, very much like 3D tomography.

References

1. Ayache N, Hansen C (1988) Rectification of images for binocular and trinocular stereovision. In: Proceedings of the 9th international conference on pattern recognition. IEEE Computer Society, Rome, Italy, pp 11–16
2. Boyer K, Kak A (1987) Color-encoded structured light for rapid active ranging. *IEEE Trans Pattern Anal Mach Intell* 9(1):14–28
3. Boykov Y, Kolmogorov V (2004) An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans Pattern Anal Mach Intell* 26(9):1124–1137
4. Collins RT (1996) A space-sweep approach to true multi-image matching. In: Proceedings of the international conference on computer vision and pattern recognition (CVPR). IEEE Computer Society, San Francisco, CA, USA, pp 358–363

5. Cox IJ, Hingorani SL, Rao SB, Maggs BM (1996) A maximum likelihood stereo algorithm. *Comput Vis Image Underst* 63:542–567
6. Gong M, Yang R, Wang L, Gong M (2007) A performance study on different cost aggregation approaches used in real-time stereo matching. *Int J Comput Vis* 75(2): 283–296
7. Hartley R, Zisserman A (2003) Multiple View Geometry in computer vision. Cambridge University Press. ISBN 0-521-54051-8
8. Hirschmueller H (2008) Stereo processing by semi-global matching and mutual information. *IEEE Trans Pattern Anal Mach Intell* 30(2):328–341
9. Kolb A, Barth E, Koch R, Larsen R (2010) Time-of-flight cameras in computer graphics. *Comput Graph Forum* 29(1):141–159
10. Kolb A, Koch R (eds) (2009) Dynamic 3D Imaging. Lecture Notes in Computer Science, vol 5742. Springer, Berlin/New York
11. Marr D (1982) Vision. A Computational Investigation into the Human Representation and Processing of Visual Information. W.H. Freeman, San Francisco
12. Nayar SK, Watanabe M, Noguchi M (1996) Real-time focus range sensor. *IEEE Trans Pattern Anal Mach Intell* 18(12):1186–1198
13. Pollefeys M, Koch R, Gool LV (1999) A simple and efficient rectification method for general motion. In: Proceedings of the 7th international conference on computer vision, vol 1, IEEE Computer Society, Korfu, Greece, pp 496–501
14. Scharstein D, Szeliski R (2002) A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int J Comput Vis* 47:7–42
15. Schwarte R, Xu Z, Heinol H-G, Olk J, Klein R, Buxbaum B, Fischer H, Schulte J (1997) New electro-optical mixing and correlating sensor: facilities and applications of the photonic mixer device (pmd). In: Proceedings of the SPIE 3100.SPIE
16. Shade J, Gortler SJ, wei He L, Szeliski R (1998) Layered depth images. In: SIGGRAPH, Proceedings of the 25th annual conference on computer graphics and interactive techniques. ACM, New York, pp 231–242
17. Szeliski R (2010) Computer vision, algorithms and applications. Springer, London/New York
18. Tombari F, Mattoccia S, Di Stefano L, Addimanda E (2008) Classification and evaluation of cost aggregation methods for stereo correspondence. In: IEEE conference on computer vision and pattern recognition, 2008 (CVPR 2008). IEEE Computer Society, Anchorage, Alaska, USA, pp 1–8

Depth from Scattering

► [Shape from Scatter](#)

Depth Imaging

► [Depth Estimation](#)

Descattering

Tali Treibitz

Department of Computer Science and Engineering,
University of California, San Diego, La Jolla,
CA, USA

Synonyms

[Dehazing](#)

Related Concepts

► [Dehazing and Defogging](#); ► [Underwater Effects](#)

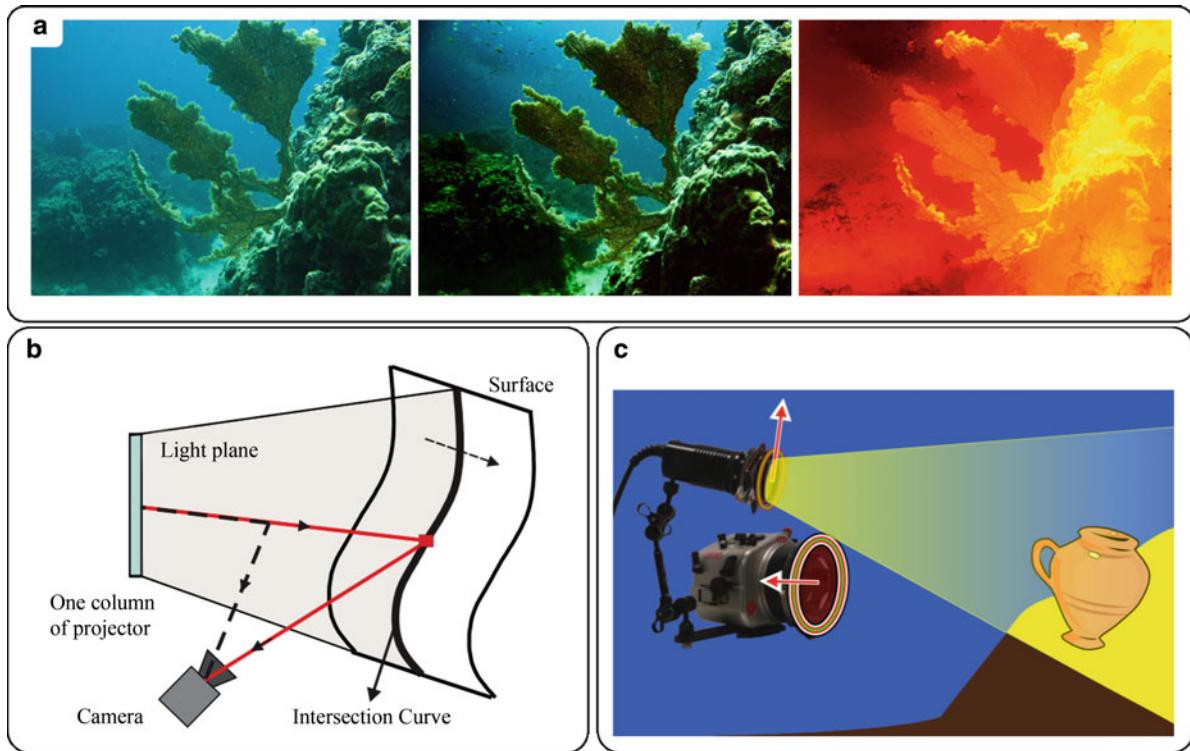
Definition

The process of descattering refers to enhancing images taken in scattering media.

Background

Imaging in scattering media poses special concerns for computer vision methods. Examples for such media include haze or bad weather, water, blood, and body tissue. Light propagating in such media is scattered and attenuated. When imaging, some light from the source is scattered back from the medium towards the camera, before ever reaching the object. This light is an additive radiance component in the image that veils the object. In air, this additive component is often termed *airlight* and in water and tissue it is termed *backscatter* (this term will be used further on). The backscatter hampers visibility by reducing contrast and signal-to-noise ratio (SNR) in the images [1]. The backscatter increases the measured radiance, which increases the photon noise, without increasing the signal value. Thus, there are methods that aim to *descatter* the image, i.e., reduce the effects of scattering. This sometimes has the effect of extending the visibility range.

Correcting for backscatter in a single image is fundamentally ill posed. When observing a single pixel, it is impossible to know what percentage of its intensity is reflected from the object and how much is contributed by the backscatter. Moreover, the



Descattering, Fig. 1 (a) Setup for descattering by polarized artificial illumination [5]. (b) Setup for the descattering by structured illumination [14]. (c) Result from single underwater image

descattering [10]. Raw image, descattered image, and range map (left to right) (Reproduced with permission from the authors)

backscatter value in a pixel is an accumulation of light scattered by the medium between the object and the camera. Thus, the accumulated backscatter increases with the object distance, which generally changes spatially in the image. This is a nonuniform effect which complicates recovery. However, the coupling of backscatter and the 3D structure of the scene also provides an opportunity since recovering either the backscatter or the distance can imply the other. After removal of the estimated backscatter, some methods also try to compensate for light attenuation. Reference [2] provides elaborate physical theory of scattering media. The reader is referred to the “Dehazing and Defogging” entry in this encyclopedia, and to [3] for vision in the atmosphere and to [4, 5] for imaging under artificial illumination.

Methods

Because of the ill-posed nature of the problem, most descattering methods use auxiliary information or

specialized hardware. Recently, some methods use a single image with prior assumptions or by applying constraints.

A known range map was used in [6]. In [7] the extremum values of the scene range are used, together with user indication of an airlight area in the image. Instead of range information, Ref. [7] relies on user indication of an area in the image that is not affected by the medium. In [3] the authors use images of the same scene under different weather conditions. Single-image descattering methods relax the need for user input by automatically estimating the airlight location and posing smoothness constraints [8, 9] or applying a dark channel prior [10, 11] (example in Fig. 1a). So far, single-image methods have been demonstrated on images taken under natural ambient illumination.

In highly scattering media, natural illumination has a limited range. Thus, several types of specialized hardware and instrumentation have been deployed. Time-gating [12] exploits the fact that the light from the object arrives to the sensor in a different time

than the scattered light, as they travel different optical path lengths. By opening the shutter for a very short period of time, the light reflected off an object can be separated from the scattered light. This requires strong laser illumination and a gated camera, with high temporal resolution. The time of arrival of the object reflectance also reveals its distance.

Rather than scanning the time of flight, various methods perform spatial scanning. Laser line scan methods use a highly collimated laser source that scans the scene. The laser source is imaged by a synchronized narrow field of view. Such systems have been shown to be very efficient, with a range of up to 6 attenuation lengths [13]. Structured light [14–17] methods capture a wide field of view with various illumination patterns. **Figure 1b** shows an example for a structured light setup.

Another approach modulates light by polarization. It has been shown that high degrees of polarization can be observed in backscatter. Therefore, polarization discrimination has been used to reject scattered light in images taken in haze, underwater, and also through skin. The light can be polarized naturally by scattering [18, 19] or be actively polarized [5, 20]. **Figure 1c** illustrates a setup of active polarized illumination. Some works assume the light reflected from the object is less polarized than the backscatter. Others assume the light from the object (often metal ones) is more polarized. When polarizing the light source, linear polarization or circular polarization can be chosen to optimize the backscatter degree of polarization in the specific medium.

Often, acoustic waves are less scattered than light in a medium and therefore can penetrate much deeper into the scattering media. However, sonic and ultrasonic images have less resolution than optical images and suffer from speckles. In photoacoustic microscopy, a pulsed laser beam is focused into biological tissue. The laser pulse creates sudden thermal expansion, generating ultrasonic waves, which are then detected to form an image. In this method, the light has to travel only half of the optical path while enabling high resolution.

Fluorescence is emission of light following excitation by light of a different wavelength. This phenomenon is used in microscopy to optically separate (using appropriate filters) the desired signal from scattered light that has the illuminant color.

The process of optical scattering is time reversible. Thus, in [21], the authors have shown that they can backpropagate the scattering by a phase-conjugate mirror.

References

- Treibitz T, Schechner YY (2009) Recovery limits in pointwise degradation. In: IEEE ICCP, San Francisco
- Kokhanovsky AA (2004) Light scattering media optics, 3rd edn. Springer, Berlin, p 200
- Narasimhan SG, Nayar SK (2002) Vision and the atmosphere. *Int J Comput Vis* 48:233–254
- Jaffe JS (1990) Computer modelling and the design of optimal underwater imaging systems. *IEEE J Ocean Eng* 15:101–111
- Treibitz T, Schechner YY (2009) Active polarization descattering. *IEEE Trans Pattern Anal Mach Intell* 31:385–399
- Oakley JP, Satherley BL (1998) Improving image quality in poor visibility conditions using a physical model for contrast degradation. *IEEE Trans Image Process* 78:167–179
- Narasimhan SG, Nayar SK (2003) Interactive (de) weathering of an image using physical models. In: IEEE workshop on color and photometric methods in computer vision, Nice
- Fattal R (2008) Single image dehazing. *ACM Trans Graph* 27:1–9
- Tan RT (2008) Visibility in bad weather from a single image. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), Anchorage
- Carlevaris-Bianco N, Mohan A, Eustice R (2010) Initial results in underwater single image dehazing. In: OCEANS, Seattle
- He K, Sun J, Tang X (2010) Single image haze removal using dark channel prior. *IEEE Trans PAMI* 33: 2341–2353
- Tan C, Sluzek A, Seet G (2005) Model of gated imaging in turbid media. *Opt Eng* 44:116002
- Giddings T, Shirron J, Tirat-Gefen A (2005) Eodes-3: An electro-optic imaging and performance prediction model. In: Proceedings of the MTS/IEEE OCEANS 2005, Washington, DC. IEEE, pp 1380–1387
- Narasimhan SG, Nayar SK, Sun B, Koppal SJ (2005) Structured light in scattering media. In: Proceedings of the IEEE ICCV, Beijing, vol 1, pp 420–427
- Jaffe J (2010) Enhanced extended range underwater imaging via structured illumination. *Opt Expr* 18(12):12328–12340
- Levoy M, Chen B, Vaish V, Horowitz M, McDowall I, Bolas M (2004) Synthetic aperture confocal imaging. *ACM Trans Graph* 23:825–834
- Nayar SK, Krishnan G, Grossberg MD, Raskar R (2006) Fast separation of direct and global components of a scene using high frequency illumination. *ACM Trans Graph* 25:935–944
- Schechner YY, Karpel N (2005) Recovery of underwater visibility and structure by polarization analysis. *IEEE J Ocean Eng* 30:570–587
- Schechner YY, Narasimhan SG, Nayar SK (2003) Polarization-based vision through haze. *Appl Opt* 42: 511–525

20. Jacques SL, Ramella-Roman J, Lee K (2002) Imaging skin pathology with polarized light. *J Biomed Opt* 7:329–340
21. Cui M, Yang C (2010) Implementation of a digital optical phase conjugation system and its application to study the robustness of turbidity suppression by phase conjugation. *Opt Express* 18(4):3444–3455

Description Logics

Bernd Neumann

University of Hamburg, Hamburg, Germany

Synonyms

[Concept languages](#); [Terminological logics](#)

Related Concepts

► [Spatiotemporal Reasoning](#)

Definition

Description Logics (DLs) is a family of logic-based knowledge representation formalisms for characterizing object classes and relationships between them.

Background

DLs have been developed to provide well-founded tools for knowledge representation and reasoning. Modern systems offer expressive concept languages with highly optimized reasoners, for example for concept subsumption, instance classification or consistency checks. DLs have gained additional importance due to OWL-DL, the standardized DL for the Semantic Web. Mature software tools exist to support the development of OWL-DL knowledge bases, notably the editor Protégé.

Theory

Knowledge representation in DLs is based on unary predicates called concepts (or concept terms), binary

predicates called roles (or role terms), and so-called individuals. A concept is interpreted in a set-theoretical semantics as a set of elements from a domain of discourse (also called universe), a role is interpreted as a set of pairs of elements from the domain, and an individual denotes a particular element of the domain. The elements in the second position of a role pair are called role fillers. Functional roles which map each first argument into at most one role filler are called features.

To construct a knowledge base, one has to fix a set of concept names, a set of role names, and a set of individuals. Names can be used to build complex concept and role terms. This is accomplished with the help of operators whose meaning is precisely defined in terms of the set-theoretical semantics.

Important concept operators supported by DLs include

- Concept union, intersection and negation
- Existential qualification, cardinality restriction
- Role axioms (e.g., reflexivity, disjointness, transitivity, inverse roles)

Definitions of concepts and roles and their relationships to each other form the TBox (terminological box), assertions about individuals in terms of unary and binary relationships form the ABox (assertional box). Typically, the TBox contains taxonomical hierarchies as background knowledge for a domain of discourse, while the ABox describes concrete facts.

Modern DLs are designed to be decidable, i.e., to allow sound and complete reasoning processes. This limits their expressivity to a subset of First Order Logic (FOL). The computational complexity of DL reasoning services has been shown to depend critically on the expressivity of the language.

Concrete domain predicates offer an interesting way to integrate quantitative data from low-level signal processing with symbolic high-level interpretations. Several DL systems allow predicates over natural and real numbers for concept definitions involving numerical restrictions.

One of the first Computer Vision applications developed with knowledge representation using a DL-like formalism was VEIL [1] which employed Loom for deductive reasoning in high-level image interpretation. Loom is a very expressive experimental system, but different from modern DLs, Loom's deductive procedures are not rigorously complete.

The usefulness of DLs for Computer Vision applications is still being explored. In principle, DLs may offer a well-founded formalism for connecting vision to higher-level knowledge, and may provide reasoning facilities for an artificial cognitive system. But there are also aspects which limit the usefulness for Computer Vision.

One problematic aspect is the limited expressivity of DLs which may prohibit intuitive concept formulations for symbolic scene interpretation. For example, in applications such as activity monitoring or situation recognition it is desirable to define compositional hierarchies where concepts describe aggregates of constituents meeting certain constraints. Unfortunately, constraints cannot be expressed except in simple cases, and powerful constraint reasoners are not provided by DL systems.

Another problematic aspect is the limited support for the scene interpretation process which can be provided by the deductive reasoning services of DLs. Scene interpretation is known to be basically equivalent to abduction or logical model construction, hence none of the usual DL reasoning services is immediately applicable.

It has been shown, however, that DLs can be extended or combined with other systems in useful ways. An interesting extension is by a rule system. While the unrestricted use of rules may jeopardize decidability, their employment for ABox reasoning (as “DL-safe rules”) can be supported without losing the advantages of a DL system.

In [2] rules are used to model the abduction process for multimedia interpretation, for example to provide deep annotations for multimedia web pages. A DL TBox represents background knowledge, and a DL ABox represents a low-level description of a multimedia document. The rules describe which higher-level hypotheses can be entertained to explain the low-level facts.

In [3] an OWL ontology of activity concepts, extended by SWRL rules (SWRL is the Semantic Web Rule Language proposed for OWL) is used as the DL kernel of a system for activity recognition. The concept definitions of the ontology are translated into compositional hierarchies representing hypothetical activities, and into rules and constraints in JAVA and JESS (the JAVA Expert System Shell). Using rule engines in parallel, the interpretation system tries to instantiate activity hypotheses based on scene

observations, this way generating scene descriptions at higher compositional levels.

Application

Applications areas of DL systems include:

- Semantic Web (e.g., logic-based information retrieval)
- Electronic Business (e.g., reasoning about service descriptions)
- Medicine, Bioinformatics (e.g., representing and managing biomedical ontologies)
- Process Engineering (e.g., formal representation of chemical processes)
- Software Engineering (e.g., representing the semantics of UML class diagrams)
- High-level Computer Vision (e.g., defining high-level concepts for scene interpretation)
- Cognitive Robotics (e.g., for representing an activity ontology)

Open Problems

The scalability of DL reasoning services is still a challenge, and new optimization techniques are developed to deal with large knowledge bases. Practical experiences, in particular with OWL-DL, have also revealed the need for more expressivity. SWRL (Semantic Web Rule Language) and its successor RIF (Rule Interchange Format) provide extensions in terms of rule expressions similar to Datalog. Computer Vision applications are mainly restricted to symbolic interpretation processes, DL system support for powerful interfaces to quantitative descriptions has yet to be developed.

References

1. Russ TA, Macgregor RM, Salemi B, Price K, Nevatia R (1996) VEIL: Combining semantic knowledge with image understanding. In: ARPA image understanding workshop, Palm Springs, pp 373–380
2. Gries O, Möller R, Nafissi A, Rosenfeld M, Sokolski K, Wessel M (2010) A probabilistic abduction engine for media interpretation. In: Alferes J, Hitzler P, Lukasiewicz Th (eds) Proceedings of the international conference on web reasoning and rule systems (RR-2010), Bressanone
3. Bohlken W, Neumann B, Hotz L, Koopmann P (2011) Ontology-based realtime activity monitoring using beam

- search. In: Crowley JL, Draper BA, Thonnat M (eds) Proc. ICVS 2011, Springer, Berlin/New York, pp 112–121
4. Brachmann RJ, Schmolze JG (1985) An overview of the KL-ONE knowledge representation system. *Cogn Sci* 9(2):171–216
 5. Brachman RJ, McGuinness DL, Patel-Schneider PF, Resnick LA, Borgida A (1991). Living with CLASSIC. When and how to use a KL-ONE-like language. In: Sowa JF (ed) Principles of semantic networks. Morgan Kaufmann, San Mateo, pp 401–456
 6. Woods WA, Schmolze JG (1992) The KL-ONE family. In: Lehmann FW (ed) Semantic networks in artificial intelligence. Pergamon, Oxford, pp 133–178
 7. Horrocks I (1998) Using an expressive description logic: FaCT or fiction? In: Proceedings of the 6th international conference on principles of knowledge representation and reasoning (KR'98), Trento, pp 636–647
 8. Haarslev V, Möller R (2001) Description of the RACER system and its applications. In: Proceedings of the 2001 description logic workshop (DL 2001), Stanford University, pp 132–141
 9. Baader F, Calvanese D, McGuinness D, Nardi D, Patel-Schneider P (2003) The description logic handbook. Cambridge University Press, Cambridge
 10. Neumann B, Möller RF (2006) On scene interpretation with description logics. In: Nagel HH, Christensen H (eds) Cognitive vision systems. Springer, Berlin/New York, pp 247–275

Dichromatic Reflection Model

Shoji Tominaga

Graduate School of Advanced Integration Science,
Chiba University, Inage-ku, Chiba, Japan

Definition

The dichromatic reflection model is a model that describes light reflected from an object's surface as a linear combination of two components. These components are the body (diffuse) reflection and the interface (specular) reflection.

Background

Reflection models are used for image analysis and object recognition in computer vision, as well as image rendering in computer graphics. Shafer proposed the dichromatic reflection model for inhomogeneous dielectric materials [1]. The first body reflection component provides the characteristic object

color, and the second interface reflection component has the same spectral composition as illumination. Thus, according to the dichromatic reflection model, all color values on a uniform object surface are described as a linear combination of two color vectors: the illumination color vector and the body color vector. Tominaga and Wandell proposed a method for testing the adequacy of the model. They show that under all illumination and viewing geometries, the spectral reflectance function is described as the weighted sum of two functions of the constant interface reflectance and the body reflectance [2]. This model is called the standard dichromatic reflection model of Type I. Moreover, Tominaga extended this model to describe surface-spectral reflectances of a variety of materials [3].

Theory

Standard Dichromatic Reflection Model

Model for Inhomogeneous Dielectrics (Type I)

The radiance of light reflected from an object's surface, $Y(\theta, \lambda)$, is a function of the wavelength λ and the geometric parameters θ , where λ ranges over a visible wavelength and θ includes the illumination direction angle, the viewing angle, and the phase angle. The dichromatic reflection model for an inhomogeneous dielectric object describes the reflected light as the sum of interface and body reflections, as shown in the following formula:

$$Y(\theta, \lambda) = c_I(\theta)L_I(\lambda) + c_B(\theta)L_B(\lambda), \quad (1)$$

where the terms $L_I(\lambda)$ and $L_B(\lambda)$ are the spectral power distributions of the interface and body reflection components, respectively. These components are unchanged as the geometric angles vary, and then the weights $c_I(\theta)$ and $c_B(\theta)$ are the geometric scale factors. Let $S_I(\lambda)$ and $S_B(\lambda)$ be the surface-spectral reflectances for the two components, and let $E(\lambda)$ be the spectral power distribution of the incident light to the surface. So we derive

$$Y(\theta, \lambda) = c_I(\theta)S_I(\lambda)E(\lambda) + c_B(\theta)S_B(\lambda)E(\lambda). \quad (2)$$

The total reflectance is defined by dividing the color signal $Y(\theta, \lambda)$ by $E(\lambda)$ as

$$S(\theta, \lambda) = c_I(\theta)S_I(\lambda) + c_B(\theta)S_B(\lambda). \quad (3)$$

The standard dichromatic reflection model incorporates the neutral interface reflection (or constant interface reflection) assumption, which means that the interface reflection component $S_I(\lambda)$ is constant and can be eliminated from Eq.(3). This is shown as follows:

$$S(\theta, \lambda) = c_I(\theta) + c_B(\theta)S_B(\lambda). \quad (4)$$

The neutral interface reflection assumption is applied to most materials, such as plastics and paints, because the interface reflection follows Fresnel's law [4] where the index of refraction is approximately constant for oil and water across the visible spectrum. For such surfaces, the specular reflection appears to have the same color as the illumination.

Extended Dichromatic Reflection Models

Model for Cloth (Type II)

There are many kinds of cloth materials such as silk, wool, rayon, polyester satin, cotton, velveteen, and velour. It is observed that the cotton and velveteen include little specular reflection from the surfaces, while the silk, wool, satin, and velour include gloss on the cloth surfaces, which show dichromatic reflection. The satin, however, is not described in Type I, because surface reflectance for such a material is dichromatic, but the specular reflection at the interface includes no illumination color. Therefore, the surface-spectral reflectance function is described as the generalized dichromatic reflection model of Type II seen below

$$S(\theta, \lambda) = c_I(\theta)S_I(\lambda) + c_B(\theta)S_B(\lambda), \quad (5)$$

where the interface reflectance component is not necessarily constant on wavelength.

Model for Metal (Type III)

Metals have quite different reflection properties from dielectric materials, because they only have interface reflection. If the surface is shiny and stainless, body reflection of the reflected light is negligibly small. So a sharp specular highlight is observed only at

the viewing angle of the mirror direction. This type of reflection follows Fresnel's law, and the surface-spectral reflectance function depends on the incidence angle of illumination. It is pointed out that the surface-spectral reflectances of some metals are described by two interface reflection components, one of which is the constant spectral reflectance [3]. Therefore, the dichromatic reflection model of Type III is defined (*for approximating the surface-spectral reflectance function of a metal*) as

$$S(\theta, \lambda) = c_{I1}(\theta)S_I(\lambda) + c_{I2}(\theta), \quad (6)$$

The right-hand side in Eq.(6) represents two interface reflection components. The first component corresponds to the specular reflection at the normal incidence, which produces the major object color of a metallic surface. The second component is constant on wavelength, which corresponds to the grazing reflection at the horizontal incidence where the spectral reflectance is whitened.

Application

The dichromatic reflection models are useful for object segmentation in image understanding and illuminant estimation in color constancy. Most reflection models in computer graphics are based on the standard dichromatic reflection model [15–19].

References

1. Shafer SA (1985) Using color to separate reflection components. *Color Res Appl* 10:210–218
2. Tominaga S, Wandell BA (1989) The standard surface reflectance model and illuminant estimation. *J Opt Soc Am A* 6:576–584
3. Tominaga S (1994) Dichromatic reflection models for a variety of materials. *Color Res Appl* 19:277–285
4. Born M, Wolf E (1983) Principles of optics. Pergamon, Oxford
5. Klinker GJ, Shafer SA, Kanade T (1988) The measurement of highlights in color images. *Int J Comput Vis* 2:7–32
6. Lee HC, Breneman EJ, Schulte C (1990) Modeling light reflection for computer color vision. *IEEE Trans PAMI* 12:402–409
7. Tominaga S, Wandell BA (1990) Component estimation of surface spectral reflectance. *J Opt Soc Am A* 7(2):312–317
8. Klinker GJ, Shafer SA, Kanade T (1990) A physical approach to color image understanding. *Int J Comput Vis* 4:7–38

9. Tominaga S (1991) Surface identification using the dichromatic reflection model. *IEEE Trans PAMI* 13(7):658–670
10. Klinker GJ (1993) A Physical approach to color image understanding. AK Peters, Wellesley
11. Novak CL, Shafer SA (1994) Method for estimating scene parameters from color images. *J Opt Soc Am A* 11:3020
12. Tominaga S (1996) Multichannel vision system for estimating surface and illumination functions. *J Opt Soc Am A* 13(11):2163–2173
13. Tominaga S (1996) Surface reflectance estimation by the dichromatic model. *Color Res Appl* 21(2):104–114
14. Tominaga S (1996) Dichromatic reflection models for rendering object surfaces. *J Imaging Sci Technol* 40(6): 549–555
15. Phong BT (1975) Illumination for computer-generated pictures. *Commun ACM* 18:311–317
16. Torrance KE, Sparrow EM (1967) Theory for off-specular reflection from roughened surfaces. *J Opt Soc Am* 57: 1105–1114
17. Blinn JF (1977) Model of light reflection for computer synthesized pictures. *Comput Graph* 11:192–198
18. Cook RL, Torrance KE (1981) A reflection model for computer graphics. *Comput Graph* 15:307–316
19. R Hall (1989) Illumination and color in computer-generated imagery. Springer, Berlin

Differential Geometry of Graph Spaces

► [Isotropic Differential Geometry in Graph Spaces](#)

Differential Geometry of Surfaces in Three-Dimensional Euclidean Space

Jan J. Koenderink

Faculty of EEMSC, Delft University of Technology,
Delft, The Netherlands

The Flemish Academic Centre for Science and the
Arts (VLAC), Brussels, Belgium

Laboratory of Experimental Psychology, University
of Leuven (K.U. Leuven), Leuven, Belgium

Synonyms

[Surfaces](#)

Related Concepts

► [Curvature](#); ► [Differential Invariants](#); ► [Euclidean Geometry](#); ► [Osculating Paraboloids](#); ► [Parametric Curve](#)

Definition

Differential geometry studies spatial entities in local (infinitesimal) neighborhoods. This approach enables one to exploit the power of (multi-)linear algebra. Geometrical entities are differential invariants, the generic example being the curvature of planar curves. The number of relevant differential invariants increases in more complicated settings like – in this entry – that of surfaces in three-dimensional Euclidean space.

Background

The differential geometry of surfaces in three-dimensional Euclidean space is often called “classical differential geometry” as its history goes back to the founding fathers of infinitesimal calculus, Newton and Leibnitz. There exists a huge literature, although novel additions are still forthcoming. This entry reviews the basics. There is ample literature for those needing to delve deeper.

Theory

Surfaces are smooth, two-parameter manifolds immersed into Euclidean three-space \mathbb{E}^3 . One requires that all partial derivatives are defined, and one often requires some form of “genericity.” For instance, many of the constructions discussed here will be undefined for planar surfaces, or spheres, and so forth.

There are many ways to represent surfaces, many of them useful in particular contexts. Perhaps the most commonly useful representation is by means of a parameterization $\mathbf{x}(u, v)$, where $\mathbf{x} \in \mathbb{E}^3$ and $\{u, v\} \in \mathbb{R}^2$. That is to say, the surface is treated as a two-parameter manifold immersed in Euclidean three-space. This representation is used here.

Unlike the case of space curves, there is no obvious way to introduce a parameterization akin to “arc length parameterization,” as this would limit the discussion to “developable surfaces,” surfaces that are metrically (“can be rolled out into”) planes. Most surfaces are intrinsically curved though; the sphere is an example. Thus the parameterization is assumed to be general, although regular, that is to say, the parameter curves

are supposed to mesh like the weave and weft of a cloth.

One objective is to describe surfaces purely in terms of their “shape,” disregarding their spatial attitude and location. This is akin to the “natural equations” of space curves, which are fully characterized by their curvature and torsion. The case of surfaces is more complicated because one cannot simply characterize them by their curvatures, but needs certain additional constraints. Unlike curves, not anything goes (see below).

The Metrical Description of Surfaces

One naturally attaches a moving frame to a surface, much like the Frenet frame in the case of curves. For a surface one has two tangents, \mathbf{x}_u and \mathbf{x}_v , and in \mathbf{E}^3 this immediately gives rise to a third vector $\mathbf{x}_u \times \mathbf{x}_v$, thus yielding a complete frame (see Figs. 1 and 2). The third vector is usually normalized as $\mathbf{n}(u, v)$, the surface normal. It is orthogonal to the tangent plane spanned by the two tangents. The metric of the tangent plane is conventionally expressed in terms of the First Principal Form

$$I(du, dv) = \mathbf{dx} \cdot \mathbf{dx} = Edu^2 + 2Fdudv + Gdv^2, \quad (1)$$

where

$$E(u, v) = \mathbf{x}_u \cdot \mathbf{x}_u, \quad (2)$$

$$F(u, v) = \mathbf{x}_u \cdot \mathbf{x}_v, \quad (3)$$

$$G(u, v) = \mathbf{x}_v \cdot \mathbf{x}_v. \quad (4)$$

Because the First Principal Form is positive definite, one necessarily has $E > 0$, $G > 0$, and $EG - F^2 > 0$. Knowing $I(u, v)$ allows one to calculate the lengths of curves on the surface, the surface area of patches of the surface, angles between directions on the surface, and so forth. Specifically, the area element is

$$dA(u, v) = \sqrt{EG - F^2} du dv = \|\mathbf{x}_u \times \mathbf{x}_v\| du dv. \quad (5)$$

The First Principal Form is independent of the representation, that is to say $I(u, v) = I'(u', v')$, although the coefficients E , F , and G are not.

The curvature of the surface has to do with the second-order derivatives \mathbf{x}_{uu} , \mathbf{x}_{uv} , and \mathbf{x}_{vv} . One conventionally introduces the Second Fundamental Form

$$\begin{aligned} II(du, dv) &= -\mathbf{dx} \cdot \mathbf{dn} = d^2\mathbf{x} \cdot \mathbf{n} \\ &= Ldu^2 + 2Mdudv + Ndv^2, \end{aligned} \quad (6)$$

where

$$L(u, v) = -\mathbf{x}_u \cdot \mathbf{n}_u = \mathbf{x}_{uu} \cdot \mathbf{n}, \quad (7)$$

$$M(u, v) = -\frac{1}{2}(\mathbf{x}_u \cdot \mathbf{n}_v + \mathbf{x}_v \cdot \mathbf{n}_u) = \mathbf{x}_{uv} \cdot \mathbf{n}, \quad (8)$$

$$N(u, v) = -\mathbf{x}_v \cdot \mathbf{n}_v = \mathbf{x}_{vv} \cdot \mathbf{n}. \quad (9)$$

The Second Principal Form, like the First Principal Form, is independent of the representation, that is, $II(u, v) = II'(u', v')$, if the normal has the same direction (otherwise II changes sign), although the coefficients L , M , and N are not.

The Second Principal Form describes the shape of the surface in the sense that the distance of the point $\mathbf{x}(u + du, v + dv)$ to the tangent plane at $\mathbf{x}(u, v)$ is (to the second order in $\{du, dv\}$) equal to $\frac{1}{2}\Pi_{u,v}(du, dv)$, the surface $\frac{1}{2}\Pi_{u,v}(du, dv)$ being the osculating paraboloid at $\{u, v\}$.

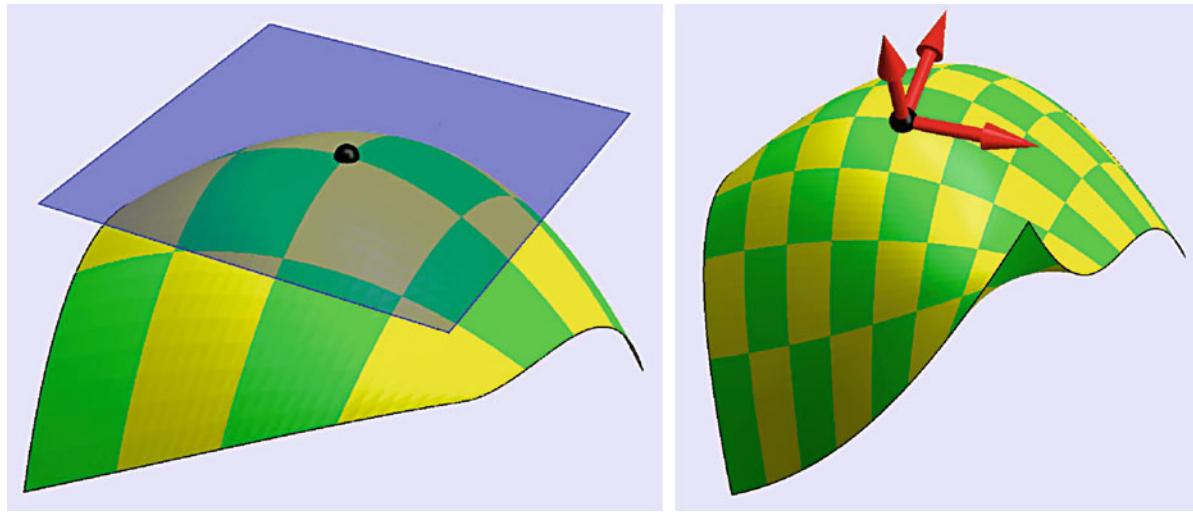
It is convenient to introduce Dupin’s indicatrix (Figs. 3–5)

$$\Pi_{u,v}(du, dv) = \pm \varepsilon^2, \quad (10)$$

with $\varepsilon \ll 1$, which, generically, is an ellipse or hyperbola. One thinks of the indicatrix as the outline of the wound that is inflicted if you take a chip off the surface with a flat knife. If the indicatrix is an ellipse, the surface is called elliptic; if it is a hyperbola, it is called hyperbolic at that point. In case $LN - M^2 = 0$, one has the degenerate parabolic case.

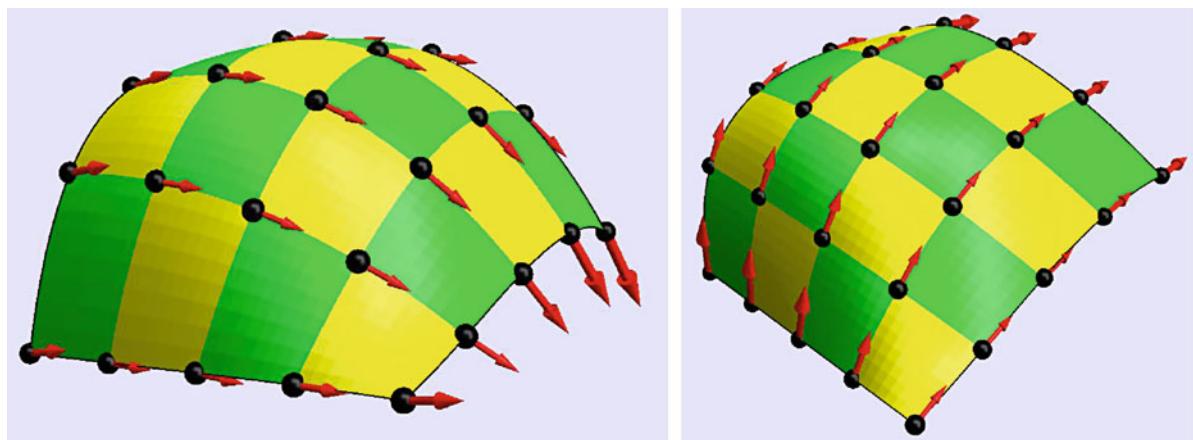
There are various ways to conceive of the curvature of surfaces. One common way is to consider the curvatures of its normal sections (Fig. 6) that are the curves of intersection with planes that contain the normal direction. The normal curvature depends on the direction $du : dv$, the normal curvature being given by

$$\kappa_n(du, dv) = \frac{II(du, dv)}{I(du, dv)} = \frac{Ldu^2 + 2Mdudv + Ndv^2}{Edu^2 + 2Fdudv + Gdv^2}. \quad (11)$$



Differential Geometry of Surfaces in Three-Dimensional Euclidean Space, Fig. 1 At left, the tangent plane at a point of a smooth surface. At right, a frame composed of the two tangents

along the parameter directions and the unit surface normal. The tangent vectors span the tangent plane



Differential Geometry of Surfaces in Three-Dimensional Euclidean Space, Fig. 2 The two fields of surface tangents \mathbf{x}_u and \mathbf{x}_v

The principal curvature varies between two extreme values, the *principal curvatures* $\kappa_{1,2}$. The principal curvatures are the solutions of the quadratic equation

$$\kappa^2 - 2H\kappa + K = 0, \quad (12)$$

where

$$H = \frac{EN + GL - 2FM}{2(EG - F^2)} = \frac{1}{2}(\kappa_1 + \kappa_2), \quad (13)$$

$$K = \kappa_1 \kappa_2 = \frac{LN - M^2}{EG - F^2}. \quad (14)$$

The differential invariant $H(u, v)$ is known as the mean curvature, and the invariant $K(u, v)$ as the Gaussian curvature. One easily shows that points with $K > 0$ are elliptic and points with $K < 0$ hyperbolic. For points with $K = 0$, one or both of the principal curvatures vanish. In the latter case, one has planar points (the osculating paraboloid being degenerated to a plane); in the former case, a cylindrical point (the osculating paraboloid being degenerated to a cylinder).

The “mean curvature” is not just the mean of the principal directions. One can easily show that the mean of the normal curvatures of arbitrary mutually

perpendicular normal sections also equals the mean curvature.

Another way to understand the Gaussian curvature is through the relation

$$K = \frac{\mathbf{n}_u \times \mathbf{n}_v \, du dv}{\mathbf{x}_u \times \mathbf{x}_v \, du dv} = \frac{d\Omega}{dA}, \quad (15)$$

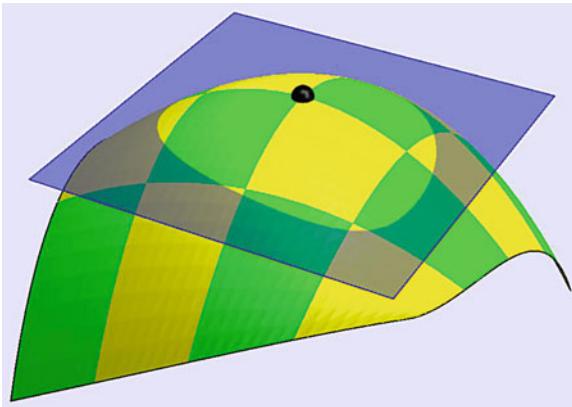
where $d\Omega$ is the solid angle subtended by the bushel of normals inside the patch of area dA . This is the direct analogon of the definition of the curvature of curves as the rate of change of direction with respect to arc length. One often introduces the Third Principal Form

$\text{III} = d\mathbf{n} \cdot d\mathbf{n}$ through the relation

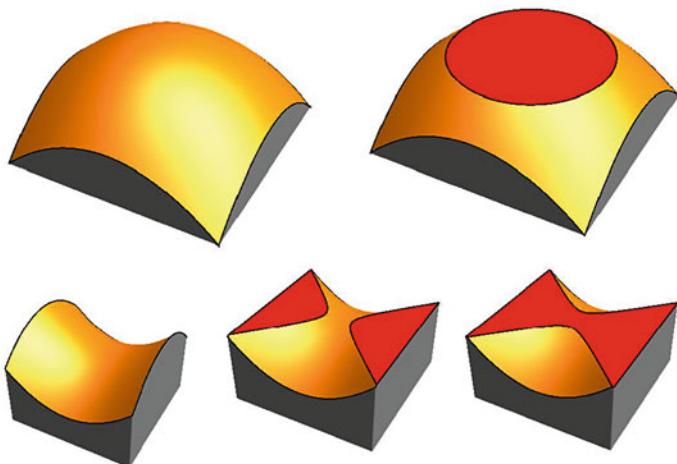
$$\text{III} - 2H \text{II} + K \text{I} = 0. \quad (16)$$

Notice that III equals the First Principal Form of the unit sphere. The mapping of the surface $\mathbf{x}(u, v)$ on the unit sphere $\mathbf{n}(u, v)$ is known as the Gauss map, or spherical image. Its area magnification is the Gaussian curvature, and the magnification in an arbitrary direction is the normal curvature in that direction. The Gauss map is a very convenient tool in a great many applications (Figs. 7–11).

The directions $du:dv$ of the principal sections are evidently special; they are known as the principal directions of curvature (see Fig. 12). One easily shows them to be mutually orthogonal. Curves for which the tangents coincide with directions of principal curvature are lines of curvature. A patch of the surface free of umbilical points (points where Dupin's indicatrix is a circle, thus where the principal directions are undefined) can be covered with a mesh of two mutually orthogonal families of lines of curvature. For such a parameterization, one has $F = M = 0$, a most “natural” representation of the surface for many purposes. In that case, the principal curvatures are $\kappa_1 = L/E$, $\kappa_2 = N/G$. An intuitive, because geometrical, characterization of the principal directions is $d\mathbf{n} \times d\mathbf{x} = 0$. In such a case, one has $d\mathbf{n} = -\kappa d\mathbf{x}$, the formula of Rodriguez. On the Gaussian sphere, conjugate directions map on mutually orthogonal directions. Euler's theorem $\kappa_n = \kappa_1 \cos^2 \varphi + \kappa_2 \sin^2 \varphi$ relates the normal curvature in the direction φ with respect to the first principal direction to the principal curvatures.

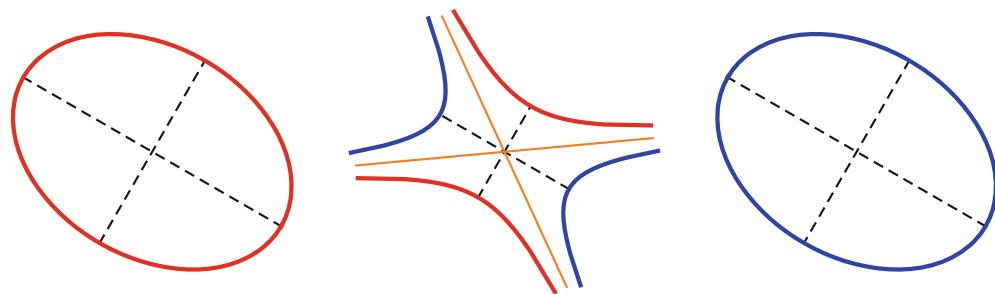


Differential Geometry of Surfaces in Three-Dimensional Euclidean Space, Fig. 3 Dupin's indicatrix can be understood as the shape of the “wound” inflicted to a surface by a cut parallel to the tangent plane. Such curves may have arbitrary shapes, but generically one obtains the cases shown in Fig. 5 for infinitesimally small wounds



Differential Geometry of Surfaces in Three-Dimensional Euclidean Space, Fig. 4

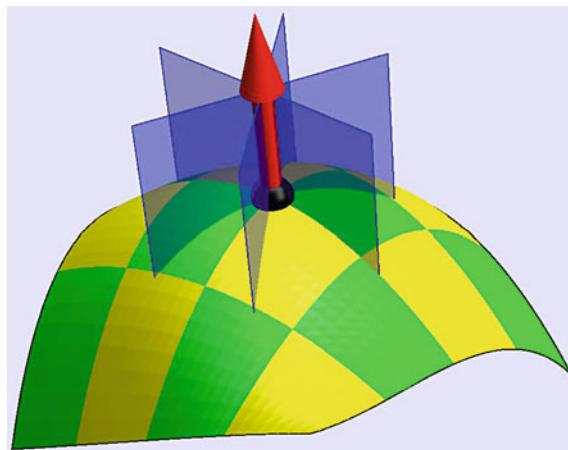
Dupin's indicatrix (the “wound”) in the case of an elliptic (top) and a hyperbolic (bottom) surface point. In the case of the elliptic point, the indicatrix has only one branch; in the case of the hyperbolic point it has two



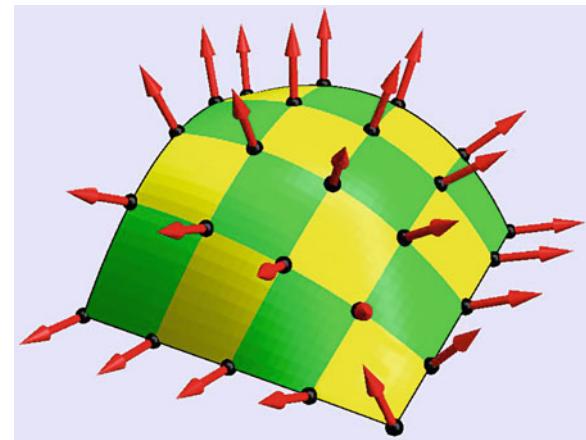
Differential Geometry of Surfaces in Three-Dimensional Euclidean Space, Fig. 5 Dupin's indicatrices for convex and concave elliptic, and for a hyperbolic surface point. The principal

directions are dashed, the asymptotic directions drawn in yellow. The red and blue curves represent the two branches of the indicatrix

D



Differential Geometry of Surfaces in Three-Dimensional Euclidean Space, Fig. 6 A series of normal sections. The normal planes contain the surface normal. They cut the surface in planar curves, the “normal sections.” The curvature of these normal sections depends on the orientation of the normal plane. The orientations that yield extremes of normal curvature are the principal directions



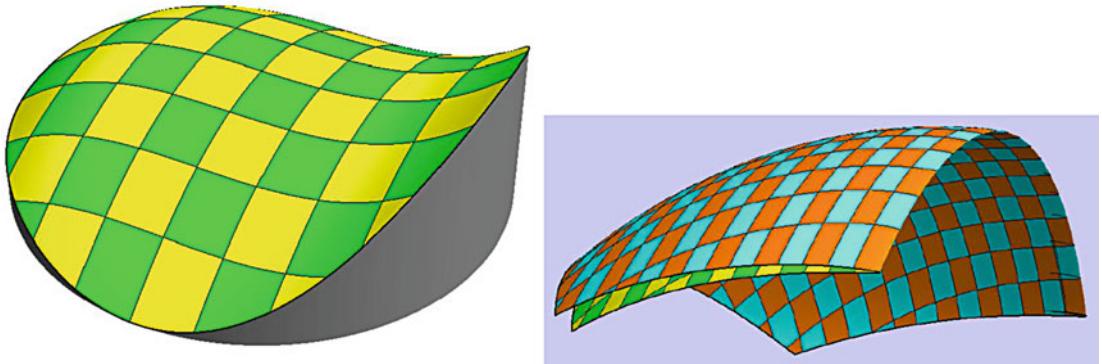
Differential Geometry of Surfaces in Three-Dimensional Euclidean Space, Fig. 7 The field of surface normals. The Gauss map, or spherical image, is obtained by moving all normals to the origin. Hence, one obtains a map from the surface to the unit sphere

Notice that it trivially applies to umbilical and planar points too.

In the case of a hyperbolic point, the principal curvatures are of opposite sign and, by Euler's theorem, there apparently exist directions for which the normal curvature vanishes. These directions are given by $\Pi(du, dv) = 0$, and are known as asymptotic directions. At generic hyperbolic points, there exist apparently two, mutually transverse, asymptotic directions. They are bisected by the principal directions. A curve whose tangents are asymptotic directions is known as an asymptotic curve. In the neighborhood of a hyperbolic point, one may cover the surface with a mesh

of two mutually transverse families of asymptotic lines. In such a case, one has $\Pi(du, dv) = M du dv$. The curvature vector \mathbf{k} of an asymptotic curve is perpendicular to the normal; typically the osculating plane is tangent to the surface. In such cases, the torsion satisfies $\tau^2 + K = 0$, the formula of Beltrami–Enneper (Fig. 13).

Directions $du:dv$ and $\delta u:\delta v$ are called mutually conjugate in case one has $d\mathbf{x} \cdot \delta \mathbf{n} = \delta \mathbf{x} \cdot d\mathbf{n} = 0$. This condition can also be written in the form $L du \delta u + M(du \delta v + \delta u dv) + N dv \delta v = 0$, which formally looks like a scalar product. At a generic point, each direction has a conjugate mate. The directions of principal curvature are mutually conjugate, whereas



Differential Geometry of Surfaces in Three-Dimensional Euclidean Space, Fig. 8 At left, the so-called “shoe surface” $\{u, v, u^3/3 - v^2/2\}$; at right, its spherical image. Notice the

fold in the spherical image. The fold is image of the parabolic curve $u = 0$. The points with parameters $\{\pm u, v\}$ have parallel normals and thus map on the same point of the unit sphere

the asymptotic directions are self-conjugate. The conjugate directions are important in computer vision because they relate the occluding contour to the viewing direction.

Like in the case of curves, one would like to attach a “moving frame” to the surface (Fig. 14). Classically, one has the Gauss equations

$$\begin{pmatrix} \mathbf{x}_{uu} \\ \mathbf{x}_{uv} \\ \mathbf{x}_{vv} \end{pmatrix} = \begin{pmatrix} \Gamma_{11}^1 & \Gamma_{11}^2 & L \\ \Gamma_{12}^1 & \Gamma_{12}^2 & M \\ \Gamma_{22}^1 & \Gamma_{22}^2 & N \end{pmatrix} \begin{pmatrix} \mathbf{x}_u \\ \mathbf{x}_v \\ \mathbf{n} \end{pmatrix}, \quad (17)$$

and the Weingarten equations

$$\begin{pmatrix} \mathbf{n}_u \\ \mathbf{n}_v \end{pmatrix} = \begin{pmatrix} \beta_1^1 & \beta_1^2 \\ \beta_2^1 & \beta_2^2 \end{pmatrix} \begin{pmatrix} \mathbf{x}_u \\ \mathbf{x}_v \end{pmatrix}, \quad (18)$$

where the Γ_{ij}^k are the Christoffel symbols of the second kind, and

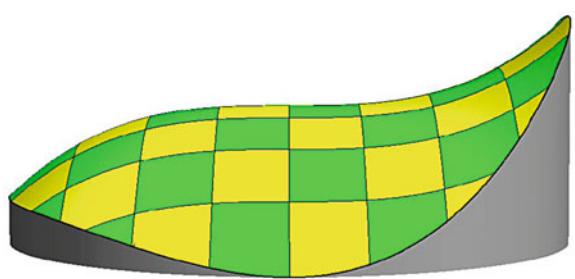
$$\beta_1^1 = \frac{MF - LG}{EG - F^2} \quad (19)$$

$$\beta_1^2 = \frac{LF - ME}{EG - F^2} \quad (20)$$

$$\beta_2^1 = \frac{NF - MG}{EG - F^2} \quad (21)$$

$$\beta_2^2 = \frac{MF - NE}{EG - F^2}. \quad (22)$$

The Christoffel symbols of the second kind depend on the coefficients of the First Principal Form and their



Differential Geometry of Surfaces in Three-Dimensional Euclidean Space, Fig. 9 As seen from a tangent to the parabolic curve, the outline of the shoe surface has an inflection. This is generic

derivatives. One has

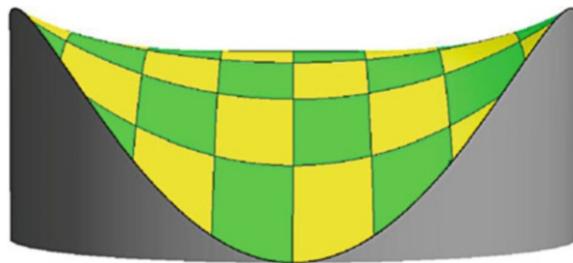
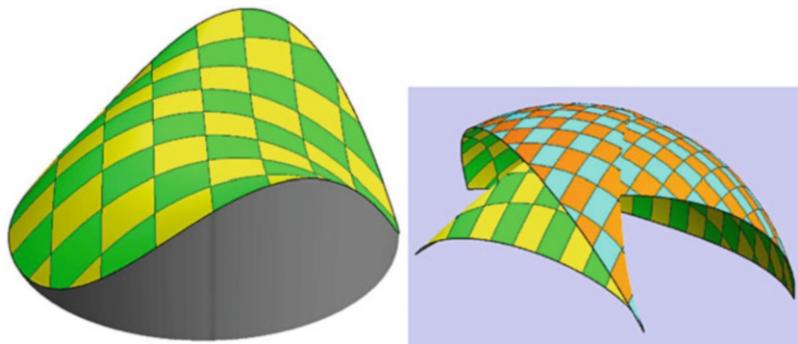
$$\begin{aligned} \Gamma_{11}^1 &= \frac{GE_u - 2FF_u + FE_v}{2(EG - F^2)} & \Gamma_{12}^1 &= \frac{GE_v - FG_u}{2(EG - F^2)} \\ \Gamma_{22}^1 &= \frac{2GF_v - GG_u - FG_v}{2(EG - F^2)} & \Gamma_{11}^2 &= \frac{EG_u - FE_v}{2(EG - F^2)} \\ \Gamma_{11}^2 &= \frac{2EF_u - EE_v + FE_u}{2(EG - F^2)} & \Gamma_{12}^2 &= \frac{EG_v - 2FF_v + FG_u}{2(EG - F^2)} \\ \Gamma_{22}^2 &= \frac{EG_v - 2FF_v + FG_u}{2(EG - F^2)}. \end{aligned} \quad (23)$$

From the Gauss equations, the geometrical interpretation of the Christoffel symbols of the second kind is obvious: the symbol Γ_{ij}^k is the k -component ($k = u, v$) of the rate of change of the tangent

Differential Geometry of**Surfaces in****Three-Dimensional****Euclidean Space, Fig. 10**

At left, the “Menn’s surface” $\{u, v, au^4 + u^2v - v^2\}$; at right, its spherical image.

Here the Gaussian image has a cusp; in the neighborhood of the cusp point, one has triples of parallel normals at distinct points of the surface

**Differential Geometry of Surfaces in Three-Dimensional Euclidean Space, Fig. 11**

At the cusp of Menn’s surface, the surface is very flat; from this viewpoint, the curvature of the outline is zero. Such points are generically isolated points on parabolic curves

\mathbf{x}_i ($i = u, v$) when transported over \mathbf{x}_j ($j = u, v$). Although the notation Γ_{ij}^k might suggest that the geometrical object Γ is a tensorial quantity, this is really not the case; its transformational properties are different.

In analogy with the case of space curves, one might guess that the “natural equations” of a surface are simply the coefficients of the First and Second Principal Forms as a function of $\{u, v\}$. However, this is not the case, the reason being the equality of mixed partial derivatives (i.e., $(\mathbf{x}_{u,v})_{uv} = (\mathbf{x}_{u,v})_{vu}$). One has to impose the additional constraints (“compatibility equations”),

$$L_v - M_u = L\Gamma_{12}^1 + M(\Gamma_{12}^2 - \Gamma_{11}^1) - N\Gamma_{11}^2, \quad (24)$$

$$M_v - N_u = L\Gamma_{22}^1 + M(\Gamma_{22}^2 - \Gamma_{12}^1) - N\Gamma_{12}^2, \quad (25)$$

the Codazzi–Mainardi equations, and

$$\begin{aligned} LN - M^2 &= F_{uv} - \frac{1}{2}E_{vv} - \frac{1}{2}G_{uu} + \\ &+ \frac{1}{EG - F^2} \left(\det \begin{vmatrix} 0 & F_v - \frac{1}{2}G_u & \frac{1}{2}G_v \\ \frac{1}{2}E_u & E & F \\ F_u - \frac{1}{2}E_v & F & G \end{vmatrix} \right. \\ &\quad \left. - \det \begin{vmatrix} 0 & \frac{1}{2}E_v & \frac{1}{2}G_u \\ \frac{1}{2}E_v & E & F \\ \frac{1}{2}G_u & F & G \end{vmatrix} \right), \quad (26) \end{aligned}$$

which, by changing $LN - M^2$ into $K(EG - F^2)$, is known as Gauss’ Theorema Egregium. It expresses the Gaussian curvature in terms of the metric and its first-order derivatives, a major result of classical differential geometry. Now we have the result that I and II, satisfying the compatibility equations, and with $E, G, EG - F^2 > 0$, determine the surface up to Euclidean movements.

With special parameterizations, these formulas may simplify greatly. For instance, in terms of a parameterization with lines of curvature, the Codazzi–Mainardi equations can be written as

$$\frac{\partial \kappa_1}{\partial v} = \frac{1}{2} \frac{E_v}{E} (\kappa_2 - \kappa_1), \quad \frac{\partial \kappa_2}{\partial u} = \frac{1}{2} \frac{G_u}{G} (\kappa_1 - \kappa_2). \quad (27)$$

An important special case in many applications is the description of the surface as a Monge patch $x\mathbf{e}_x + y\mathbf{e}_y + z(x, y)\mathbf{e}_z$, where $\mathbf{e}_{x,y,z}$ are an orthonormal Cartesian basis of \mathbb{E}^3 . One can conventionally write $p = z_x, q = z_y$ for the first- and $r = z_{xx}, s = z_{xy}, t = z_{yy}$ for the second-order partial derivatives

Differential Geometry of Surfaces in Three-Dimensional Euclidean Space

Fig. 12 At top left, a field of tangent planes along a curve: notice the rotations of the plane. At top right, the tangent planes along a curve of principal curvature. Here the tangent planes rotate about axes at right angles to the curve. At bottom, a field of tangent planes along an asymptotic curve. Here the tangent planes rotate about the tangents of the curve

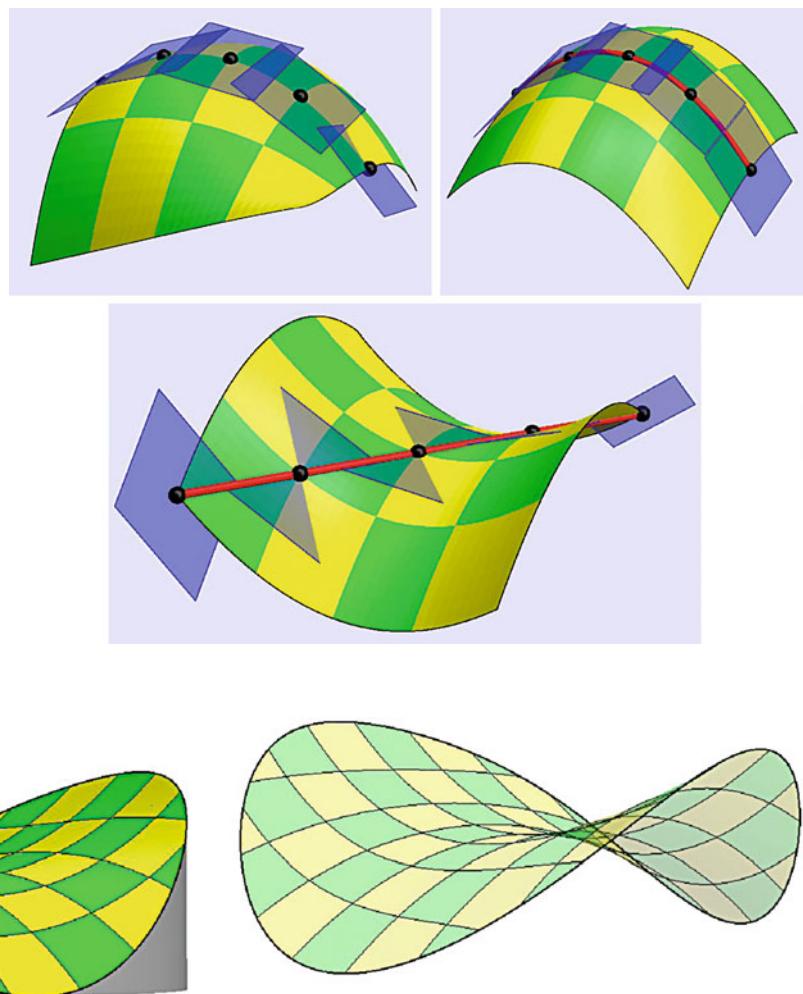


Fig. 13 At left, a view of a hyperbolic surface. Notice the singular outline; this is generic for a view along an asymptotic direction. At right, a partially transparent

hyperbolic surface which allows one to judge the singularity of the outline more easily. It is a cusp. In the opaque view, only one branch of the cusp is visible at any time

of the “height function” $z(x, y)$. Then the tangents are $\mathbf{t}_x = \mathbf{e}_x + p\mathbf{e}_z$ and $\mathbf{t}_y = \mathbf{e}_y + q\mathbf{e}_z$, the normal $\mathbf{n} = (-p\mathbf{e}_x - q\mathbf{e}_y + \mathbf{e}_z)/\sqrt{g}$, where $g = 1 + p^2 + q^2$ is the squared area element $EG - F^2$. The First Fundamental Form is

$$I(dx, dy) = (1 + p^2)dx^2 + 2pq dx dy + (1 + q^2)dy^2, \quad (28)$$

the Second Principal Form

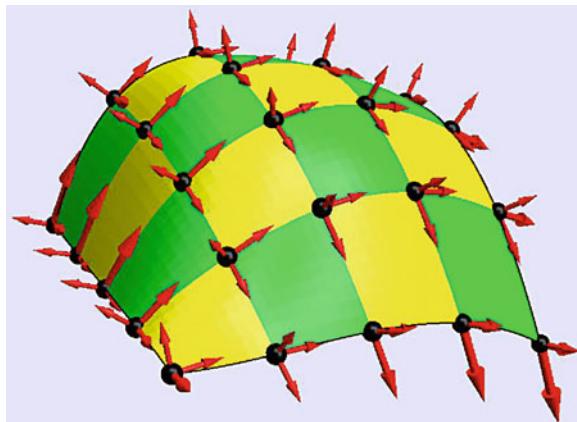
$$II(dx, dy) = \frac{r dx^2 + 2s dx dy + t dy^2}{\sqrt{g}}, \quad (29)$$

whereas the Christoffel symbols become

$$\begin{aligned} \Gamma_{11}^1 &= \frac{pr}{g}, & \Gamma_{12}^1 &= \frac{ps}{g}, & \Gamma_{22}^1 &= \frac{pt}{g}, \\ \Gamma_{11}^2 &= \frac{qr}{g}, & \Gamma_{12}^2 &= \frac{qs}{g}, & \Gamma_{22}^2 &= \frac{qt}{g}, \end{aligned} \quad (30)$$

and the coefficients of the Weingarten equations

$$\begin{aligned} \beta_1^1 &= \frac{spq - rq^2 - r}{g^{3/2}}, & \beta_2^1 &= \frac{tpq - sq^2 - s}{g^{3/2}}, \\ \beta_1^2 &= \frac{rpq - sp^2 - s}{g^{3/2}}, & \beta_2^2 &= \frac{spq - tp^2 - t}{g^{3/2}}. \end{aligned} \quad (31)$$



Differential Geometry of Surfaces in Three-Dimensional Euclidean Space, Fig. 14 The “moving frame” of a surface. The rotations and deformations of the frame as it moves over the surface are described by the equations of Gauss and Weingarten

Expression for many types of surface representations (e.g., implicit form of the type $F(x, y, z) = 0$) or special forms (e.g., surfaces of revolution) are readily found in the literature.

Of course, one often has to deal with data that are not in some convenient analytical form. This calls for methods of numerical analysis and computational geometry.

Open Problems

In this section, we described the classical differential theory of surfaces. Many settings in computer vision and image processing do not involve a Euclidean structure though. Cases include affine and projective spaces, as well as (especially important in image processing) “isotropic differential geometry” in “graph spaces.” In such cases, one needs to substitute the appropriate differential geometry. Examples abound where authors failed to take this into account and used expressions from Euclidean differential geometry inappropriately. The reader should beware.

References

1. do Carmo MP (1976) Differential geometry of curves and surfaces. Prentice-Hall, Englewood Cliffs
2. Eisenhart LP (2004) A treatise on the differential geometry of curves and surfaces. Dover, New York

3. Gray A, Abbena E, Salamon S (2006) Modern differential geometry of curves and surfaces with mathematica. CRC, Boca Raton
4. Guggenheimer H (1977) Differential geometry. Dover, New York
5. Hilbert D, Cohn-Vossen S (1952) Geometry and the imagination, 2nd edn. Chelsea, New York
6. Koenderink JJ (1990) Solid shape. MIT Press, Cambridge, MA
7. Kreyszig E (1991) Differential geometry, Chapter II. Dover, New York
8. O’Neill B (1997) Elementary differential geometry. Academic, New York
9. Spivak M (1979) Comprehensive introduction to differential geometry, 5 vols. Publish or Perish, Houston
10. Struik DJ (1988) Lectures on classical differential geometry, 2nd edn. Dover, New York

Differential Invariants

Isaac Weiss

Center for Automation Research, University of Maryland at College Park, College Park, MD, USA

Synonyms

Local invariants

Related Concepts

► [Affine Invariants](#)

Basic Concepts

Invariants are entities that do not change under the action of a transformation group, e.g., projective invariants are unchanged under projective transformations. One can distinguish between differential and algebraic invariants. Algebraic invariants involve algebraic forms such as points, lines, conics, etc., while differential invariants involve general differentiable curves and surfaces. Here we concentrate on invariants of curves, mostly in the plane. Differential invariants of surfaces also exist [18]. Also mentioned are differential invariants of space curves and of fields such as optic flow and shading.

Both differential and algebraic invariants, well developed in the mathematical literature, were introduced into computer vision in [11] in order to eliminate the search for the correct viewpoint when trying to recognize an object. Compared to algebraic invariants, the advantages of differential invariants are as follows:

- (a) Differential invariants can describe any arbitrary curve visible in the image. Algebraic curves in images are quite limited. While ellipses are common as projections of round objects, other algebraic curves are rare.
- (b) Algebraic invariants require whole curves to be visible and are thus susceptible to occlusions. Differential invariants can be extracted from any visible parts of the curve. The disadvantage is that the differential invariants are harder to extract from the image reliably.

It is useful in projective geometry to replace the normal Cartesian coordinates (x, y) of a point in the plane by a triplet

$$\mathbf{x} = (x_1, x_2, x_3) = \lambda(x, y, 1) \quad (1)$$

with an arbitrary factor λ . To these one adds points with $x_3 = 0$ which have no corresponding Cartesian coordinates but can be thought of as the points at infinity. The point $(0,0,0)$ is excluded from the space. The homogeneous coordinates are also convenient in the affine subgroup of the projective transformation group. In this case, one sets $\lambda = 1$ and $x_3 = 1$.

In these coordinates, a projective transformation (projectivity) can be written as a linear transformation of \mathbf{x} to $\tilde{\mathbf{x}}$:

$$\tilde{\mathbf{x}} = \lambda(\mathbf{x})T\mathbf{x} \quad (2)$$

with T being a constant 3×3 matrix. The factor λ can change arbitrarily from one point to another.

A shape can be described in different ways by various shape descriptors s_k . One can express a curve parametrically as $\mathbf{x}(t)$, in which case the descriptors are usually derivatives with respect to t . It can also be expressed implicitly as a function $f(\mathbf{x}, s_k) = 0$, with the descriptors s_k being constant coefficients, e.g., of a conic. They transform to \tilde{s}_k .

A relative invariant I of weight w is defined as a function of the shape descriptors s_k that transforms as

$$I(\tilde{s}_k) = J^{-w} I(s_k). \quad (3)$$

J is the Jacobian of the transformation. There are in general different weights for different J s: J of the coordinate transformation is the determinant $|T|$, and there is also J of a parameter change, i.e., $d\tilde{t}/dt$. A change can also result from multiplication of s_k homogeneously by a factor λ . This is of importance in projective homogeneous coordinates. In this case, the invariant can change as

$$I(\lambda s_k) = \lambda^d I(s_k) \quad (4)$$

with d being the *degree* of the invariant.

An invariant of weights and degree zero is *absolute*.

The Jacobians and λ can vary from one point to another, namely, they depend on \mathbf{x}, t , but they do not depend on the descriptors s_k of the shape itself.

General Properties

Among the general properties of interest here are whether there is a set of invariants that are necessary and sufficient to describe a shape. For differential invariants, the completeness theorem holds ([3], p. 144):

Theorem 1 *All differential invariants of a (transitive) transformation group in the plane are functions of the two lowest-order invariants and their derivatives.*

This is equivalent to saying that the original curve can be reconstructed from the two lowest-order-independent invariants that exist at each point, up to the relevant transformation. This is because these invariants contain all the information about the curve except for a group transformation.

This completeness makes it possible to create an invariant “signature” of a curve. For example, in the Euclidean case, all invariants can be derived from the curvature and the arc-length at each point, $\kappa(\tau)$. Thus, one can use the signature, namely, the plot of the curvature vs. the arc-length, to recognize the curve up to a Euclidean transformation. Similar quantities can be derived in the affine case as described later. In the projective case, there is no natural arc-length or curvature, but there are still two independent invariants at each curve point that can be plotted against each other. Since this signature is invariant to the viewpoint, the

curve can be recognized without a search for the correct viewpoint. This is done without having to find the point correspondence along the curve, a common and difficult problem.

Another general property is that the more general the transformation is, the more descriptors one needs to extract from the image to find invariants. The general projectivity in the plane has eight coefficients that need to be eliminated, so to obtain two invariants, one needs to extract from the image at least ten descriptors per point. The affine subgroup has only six coefficients to eliminate and the Euclidean has three. Thus, more general invariants lead to using higher-order derivatives or higher-order implicit curves.

Overview

Wilczynski's method [6, 17, 18] was the first to obtain closed form formulas for projective invariants of curves and surfaces. While interesting mathematically, it has proved difficult to implement in vision because of the high order of derivatives involved.

Cartan's "moving frame" method [3] is easily applied to transformations that have a natural arc-length, such as Euclidean or unimodular affine transformations (i.e., with $|T| = 1$). In the latter case, one can obtain the affine arc-length and affine curvature. However, it is hard to apply it to transformations (such as projectivities) which do not admit a natural arc-length parameter.

The determinants method takes advantage of the invariant properties of determinants under linear transformations and is also very useful for algebraic invariants. Here it is used to obtain the affine arc-length and curvature more simply than in Cartan's method.

All the above methods depend on the availability of derivatives of the curve $\mathbf{x}(t)$ w.r.t. the parameter t . These can be hard to extract reliably. The canonical method [4, 12] dispenses with both the derivatives and the curve parameter. The parameter is not really a part of the geometry of the curve; it is an artifact introduced for convenience. It turns out that while no method can reduce the number of descriptors that one needs to extract, the canonical method provides descriptors that are more robust and easier to extract. In a nutshell, an auxiliary implicit curve (i.e., a function $f(x, y)$ without a parameter) is fitted around a point of the data

curve. Then, the coordinate system is transformed to a canonical, or standard system, in which this auxiliary curve has a particularly simple form. All quantities in this system are invariant.

Another important issue is the connection between 2D images and 3D objects. It has long been known that (in general) the projection from a 3D shape to a single 2D image does not have invariants. There is simply not enough information in a 2D image to make up for the missing depth information by purely geometric means. The invariants discussed above are 2D to 2D (or nD to nD). However, one can find useful invariant *constraints* between invariants of 3D curves and those of their 2D projections [15]. These can identify a shape with the help of information from known models.

Finally, a method is described for obtaining accurate derivatives [13]. The common methods yield incorrect results even in the analytic noiseless case. This systematic error is analyzed and corrected.

A 1D Projective Differential Invariant

First, it is worth mentioning a well-known one-dimensional differential invariant, namely, the Schwarzian derivative [10]. Consider a particle moving along a straight line, with its position at a time t measured by a (nonhomogeneous) coordinate $x(t)$. The Schwarzian derivative S_x is defined as

$$S_x \equiv \left(\frac{x''(t)}{x'(t)} \right)' - \frac{1}{2} \left(\frac{x''(t)}{x'(t)} \right)^2, \quad (5)$$

and it is invariant under projective transformations of the line, given by $\tilde{x} = (ax + b)/(cx + d)$, with arbitrary a, b, c, d , as can be verified directly.

Furthermore, the differential equation

$$S_x = g(t) \quad (6)$$

(where $g(t)$ is given) determines the function $x(t)$ up to 1D projectivity. The Schwarzian derivative is not invariant to change of the parameter t , except by a 1D projectivity similar to that of x .

It is interesting that this invariant can be obtained as an infinitesimal limit of the well-known 1D cross ratio.

Wilczynski's Method

This method finds invariants of the transformation (Eq. 2) in stages. First it finds invariants to the linear part T of the transformation, and from these it derives invariants to λ and then to change in the curve parameter t .

Given a plane curve $\mathbf{x}(t)$, invariants to T can be obtained by solving the linear algebraic system of equations

$$\mathbf{x}''' + 3p_1\mathbf{x}'' + 3p_2\mathbf{x}' + p_3\mathbf{x} = 0 \quad (7)$$

for the three unknowns p_1, p_2, p_3 at point t . It is easy to show, by multiplying the equation through by T , that these solutions p_i are invariant to T . (In fact, p_i are expressible as determinants which fits the determinants method.) However, they are not invariant to change in the arbitrary factor $\lambda(\mathbf{x}(t))$ nor to change in the curve parameter t . From these p_i , one constructs the “semi-invariants”:

$$P_2 = p_2 - p_1^2 - p_1' \quad (8)$$

$$P_3 = p_3 - 3p_1p_2 + 2p_1^3 - p_1''. \quad (9)$$

These remain unchanged under multiplication of the coordinates by a factor $\lambda(\mathbf{x})$, but not under change of the parameter t .

The full invariants are

$$\Theta_3 = P_3 - \frac{3}{2}P_2' \quad (10)$$

$$\Theta_8 = 6\Theta_3\Theta_3'' - 7(\Theta_3')^2 - 27P_2\Theta_3^2. \quad (11)$$

Under change of the parameter t , they transform as $\tilde{\Theta}_w = (d\tilde{t}/dt)^{-w}\Theta_w$, and thus w is the weight (Eq. 3). The subscript corresponds to the weight w .

Θ_3 is the only linear invariant, and $\sqrt[3]{\Theta_3}$ can be called the projective arc-length. All other invariants can be derived from Θ_3, Θ_8 and their derivatives. This is a special case of the completeness Theorem 1. The original curve can be derived from these up to a projectivity. One can thus call the above two invariants a *complete* set of independent invariants.

These two invariants still contain the unknown weights, which vary from point to point. To eliminate

them, one can use the invariant $\Theta_{12} = 3\Theta_3\Theta_8' - 8\Theta_3'\Theta_8$. One can now define the two absolute invariants [11]:

$$I_1 = \frac{\Theta_3^8}{\Theta_8^3}, \quad I_2 = \frac{\Theta_3^4}{\Theta_{12}}. \quad (12)$$

These can be plotted against each other in an invariant plane with coordinates I_1, I_2 . One can thus obtain an invariant signature curve identifying the original curve up to a projectivity.

Since these invariants contain the eighth derivative, they are not very practical. The semi-invariant P_2 above contains the fourth derivative only. The other, P_3 , contains the fifth, but it can be replaced by

$$P_3^* = P_3 - P_2' \quad (13)$$

which again contain only four derivatives.

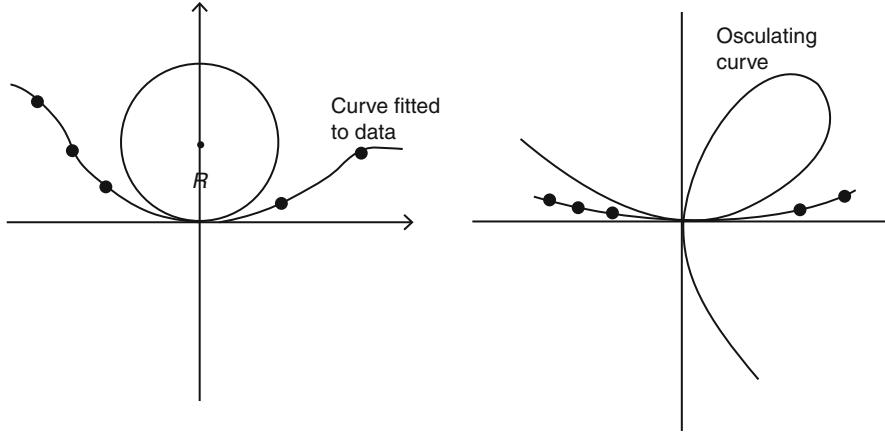
One can clearly see the burden that the curve parameter imposes on the method. The semi-invariants P_2, P_3^* are invariant to the projectivity and contain only fourth derivatives. It is the requirement of invariance to the change of parameter t that pushes the number of derivatives needed to eight. Thus, if one can get rid of the parameter, there will be fewer local quantities needed, and the robustness of the invariance will increase.

The Canonical Method

As was seen above, the need to eliminate the parameter t pushes the order of derivatives needed from four to eight. The canonical method [4, 9, 12] does not have a parameter in the first place. It can be applied quite generally, but here the application to projectivities is described.

The basic idea is to transform the given coordinate system to a “canonical,” or standard system, which is determined only by the shape itself. Since this canonical system is independent of the original system, it is invariant. All quantities defined in it are thus invariant.

An important simple example is the Euclidean invariants. To find an invariant at a given point \mathbf{x}_1 on a curve, one attaches a circle at that point so that the tangent (first derivative) of the circle coincides with that of the curve at \mathbf{x}_1 (Fig. 1). Also, the radius of the circle is set so that its second derivative is also equal



Differential Invariants, Fig. 1 Osculating circle (left) and Folium (leaf) of Descartes (right)

to the curve's at \mathbf{x}_1 . Thus, one obtains three so-called points of contact between the given curve and the circle at \mathbf{x}_1 that are infinitesimally close and linearly independent.

The next step is to move \mathbf{x}_1 to the origin and rotate the curve so that its tangent coincides with the x -axis. Now \mathbf{x}_1 and the derivative dy/dx vanish there, Fig. 1. By this, one has exhausted all the Euclidean transformations (translations and rotation) and arrived at a “canonical” Euclidean coordinate system. In this system, all quantities are Euclidean invariants as they cannot be changed by any further Euclidean transformation. In particular, the distance from \mathbf{x}_1 to the center of the circle (which now lies on the y -axis) is invariant. This is the radius of curvature which is thus proved invariant.

Although derivatives were mentioned, one does not need to actually find them. Instead, one can fit a circle, $(x - x_0)^2 + (y - y_0) = R^2$, to the curve around point \mathbf{x}_1 . The coefficients of this circle, R, x_0, y_0 , can be found without derivatives or a curve parameter. In principle, one needs only three data points to fit a circle, but in practice, one wants a wider window around \mathbf{x}_1 . It is convenient to fit a conic in this window. A conic can be expressed as a matrix of coefficients A , satisfying $\mathbf{x}' A \mathbf{x} = 0$. This is easier to fit because the conic is linear in its coefficients A . One can then find the appropriate circle from this conic algebraically. After moving to the canonical system, the new conic coefficients \hat{A} are all invariants.

One can generalize the method above to the projective case. Instead of the circle, one uses a more general “osculating curve” with more points of contact with

the given curve at \mathbf{x}_1 . To eliminate the eight projective coefficients, one needs an osculating curve with at least eight contact points. A suitable choice for projectivities is the “nodal cubic” [4] (Fig. 1). After moving to a Euclidean canonical system, this can be expressed as

$$f_{\text{osc}} = c_0 x^3 + c_1 y^3 + c_2 x y^2 + c_3 x^2 y + c_4 y^2 + x y = 0. \quad (14)$$

This curve intersects itself at the origin so it has two tangents there, one lying along the x -axis. The other tangent is called the “projective normal” [6].

The image curve is given as data pixels so one needs to fit a differentiable curve to it. Instead of fitting a conic A around \mathbf{x}_1 as before, one fits a higher-order form such as a cubic or a quartic $f(\mathbf{x}) = 0$. In principle, a cubic f will do, having nine coefficients plus the point's position on the curve. In practice, however, it was found [9] that a wide window is necessary for robustness to noise, and this requires a higher-order curve such as a quartic

$$f(x, y) = a_0 + a_1 x + \dots + a_{14} y^4 = 0. \quad (15)$$

(Not all its coefficients need be independent.)

After finding f above by fitting to the data, one finds the osculating nodal cubic (Eq. 14) from it algebraically. The goal is now to transform the coordinates so that this nodal cubic (Eq. 14) takes on the simple coefficient-free form

$$x^3 + y^3 + x y = 0 \quad (16)$$

known as a *folium of Descartes* (Fig. 1) using projective transformations.

In a nutshell, one obtains it as follows. Three of the eight projectivity parameters were already eliminated by moving to a Euclidean canonical frame, namely, moving the origin to \mathbf{x}_1 and rotating so that the x -axis is tangent to the curve there. Thus, one already has the x -axis of the canonical system. The canonical y -axis is now chosen as the other tangent of the nodal cubic, the projective normal. One now skews the system so that this projective normal becomes perpendicular to the x -axis. This will eliminate the term with c_4 in the nodal cubic (Eq. 14). Next, the coefficients c_0, c_1 are eliminated by scalings in the x and y directions. One obtains

$$\tilde{x}^3 + \tilde{y}^3 + \tilde{c}_2\tilde{x}\tilde{y}^2 + \tilde{c}_3\tilde{x}^2\tilde{y} + \tilde{x}\tilde{y} = 0. \quad (17)$$

The coefficients in this system, \tilde{c}_2 and \tilde{c}_3 , are *differential affine invariants* because one has reached an affine canonical system – all possible affine transformations (translations, rotation, skewing, scalings) have been used to eliminate all the possible affine transformation coefficients. The remaining coefficients cannot be changed by any further affine transformation so they are affine invariants.

A projective canonical system is obtained by eliminating the last two coefficients \tilde{c}_2, \tilde{c}_3 using the remaining projective transformations of tilt and slant. Finally, one transforms the original fitted curve f to this new system and obtains new coefficients \tilde{a}_i for it. Since this system is projectively invariant, these \tilde{a}_i are invariants. One can choose some suitable combinations of them as invariants I_1, I_2 .

In summary, the canonical method with implicit curves was used to eliminate the high-order derivatives and the curve parameter. Further details can be found in [9].

The Method of Determinants

The method of determinants is the main tool for deriving invariants of algebraic forms. It can also be used to derive differential affine and Euclidean invariants.

This method takes advantage of the properties of determinants under linear transformations. Many geometric entities can be cast in the form of determinants of points in homogeneous coordinates. In 1D, the

distance between points x_1, x_2 is

$$D_{12} = x_1 - x_2 = \begin{vmatrix} x_1 & 1 \\ x_2 & 1 \end{vmatrix}. \quad (18)$$

Similarly, in 2D, the area of a triangle can be written as

$$S_{123} = \frac{1}{2} \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}. \quad (19)$$

In the projective case, the coordinates can be multiplied by arbitrary factors, e.g., $\mathbf{x}_i = \lambda_i(x_i, y_i, 1)$. Thus, the projective transformation (Eq. 2) transforms a determinant by multiplying it by $|T|$ and λ_i :

$$|\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \tilde{\mathbf{x}}_3| = \lambda_1 \lambda_2 \lambda_3 |T| |\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3|. \quad (20)$$

Thus, a determinant of points in homogeneous coordinates is a *relative projective invariant* of weight 1 in $|T|$ and degree 1 in each λ_i . It is a relative affine invariant with $\lambda_i = 1$.

The above properties are used to find invariants of various algebraic forms. The main trick is to find ratios of various determinants (e.g., lengths, areas) in which all the factors λ_i as well as $|T|$ cancel out, so one gets absolute invariants. When ratios cannot do the trick, then “cross ratios,” or ratios of ratios, usually do. Beyond points, determinants of higher-order forms such as conics and cubics are also easily obtained, e.g., for two conics [10]. The duality of points and lines in the projective case makes it possible to interchange their roles in the determinants.

Explicit Differential Affine Invariants

In the canonical method, one obtains implicit (parameter-less) invariants. Here one obtains explicit differential affine invariants by using the determinants of the derivative vectors $\mathbf{x}', \mathbf{x}''$, etc., at point t . The affine case is simpler than the projective case. First, it is assumed that $\lambda = 1$ so λ does not have to be eliminated. Second, the third coordinate is now $x_3 = 1$, and it is involved only with translations. Since the derivatives eliminate any translation coefficients, it is sufficient to deal with a 2D nonhomogeneous coordinate transformation:

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = T \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}. \quad (21)$$

For this transformation, the 2D determinant $|\mathbf{x}', \mathbf{x}''|$ is a relative invariant of weight 1 in $|T|$, analogous to (Eq. 20). It can thus be used to define the *affine arc-length* [3]:

$$\tau = \int_{t_0}^t |\mathbf{x}', \mathbf{x}''|^{1/3} dt. \quad (22)$$

It is absolute with respect to $\tilde{\tau}'$ but relative with respect to T . It will now be used as an invariant parameter for all the differentiations, denoting derivatives as, e.g., \mathbf{x}_τ . Although an arc-length depends on a starting point t_0 , the derivatives with respect to it do not. Higher-order relative invariants can now be obtained as further determinants. One immediately obtains the *affine curvature*:

$$\kappa_{af} = |\mathbf{x}_{\tau\tau}, \mathbf{x}_{\tau\tau\tau}| \quad (23)$$

involving the fourth-order derivative w.r.t. t .

The affine curvature is constant along conics and only conics. It is related to the conic area, a relative invariant, $\pi \kappa_{af}^{3/2}$. Thus, one can interpret the affine curvature geometrically as the area of a conic that osculates the curve at $\mathbf{x}(\tau)$.

For unimodular affine transformations, i.e., with $|T| = 1$, all these relative invariants become absolute. They have been obtained by Cartan in the moving frame method [3].

Explicit Differential Euclidean Invariants

In this case, T is orthonormal, and one has a new invariant, namely, the differential arc-length $\mathbf{x}'^t \cdot \mathbf{x}'$. One can use it to define the integral Euclidean arc-length:

$$\tau = \int (\mathbf{x}'^t \cdot \mathbf{x}')^{1/2} dt. \quad (24)$$

Using derivatives w.r.t. it, this one can define the *Euclidean curvature*

$$\kappa = |\mathbf{x}_\tau, \mathbf{x}_{\tau\tau}| = x_\tau y_{\tau\tau} - y_\tau x_{\tau\tau}. \quad (25)$$

The Euclidean invariants are all absolute as $|T| = 1$.

Projection from 3D to 2D

The above invariants involve the transformation from 2D to 2D shapes. Similar invariants can be derived in n D spaces. In vision, one is more often interested in the projection from 3D objects to 2D images. It has been shown that there are no invariants to such transformations because one loses the depth dimension.

However, it has been shown [15, 16] that there exist invariant *constraints*, i.e., quantities in the image that constrain the corresponding 3D invariants to a subspace of their full invariant space. In particular, for given five 2D image points, the corresponding 3D invariants are constrained to lines in a 3D invariant space. These constraints were obtained by analyzing the projection properties of determinants. Similar methods have been used to obtain constraints in the differential case for the projection of 3D curves to 2D [15].

Such constraints are very useful in combination with given 3D models of the objects. These models provide additional constraints that, together with the invariant constraints, identify an object uniquely. This was applied in [16] to images of vehicles.

In [5], a version of the canonical method has been used to determine whether a space curve lies on some simple surface such as a cylinder or a cone.

Other Invariants

Hybrid invariants, combining differential and algebraic invariants, reduce the number of derivatives at the price of adding points or lines that one needs to match. They are described in [1, 2, 8, 12]. Invariants of fields are described in [7] and are applied to shape from shading in [14].

Accurate Differentiation

Differentiating an image is sensitive to noise and inaccuracies, so one normally convolves the image with a smoothing filter S . Using integration by parts, it is easy to show in theory that the n -th derivative of the image data, $f^{(n)}$, smoothed by S , can be obtained as a convolution with the n -th derivative of the filter S :

$$S \otimes f^{(n)} = S^{(n)} \otimes f. \quad (26)$$

However, the integration by parts is valid only for a *differentiable* filter S . In practice, one commonly uses the truncated Gaussian which is not differentiable at its ends. This is more than a technicality. A quick check with a simple data function such as $f = x^k$ will immediately reveal that the above equation is not valid for the truncated Gaussian filter even in this analytic, noiseless case. The error gets much worse for higher-order derivatives.

To solve the problem, one can replace the truncated Gaussian by a spline approximation of it [13]. For the n -th derivative, one can use an n -th order differentiable spline. In practice, one does not really need to calculate the splines. It is well known that the n -th order derivative of an n -th order spline interpolation of given $n + 1$ points, at the center of the spline, is nothing but the n -th order central *difference* of the given points. One can define a filter for first-order differencing as $D = (-1, 1)$. Then, a general *difference-based* differentiation filter [13] can be defined as

$$D^n \otimes S = D \otimes D \otimes D \otimes \dots \otimes S. \quad (27)$$

This new filter replaces the erroneous derivative-based filter $S^{(n)}$, and the differentiation (Eq. 26) is replaced by

$$f_s^{(n)} = D^n \otimes S \otimes f \quad (28)$$

where $f_s^{(n)}$ is a smoothed version of $f^{(n)}$. Unlike the old filter, this new filter yields accurate derivatives for polynomials up to order $n + 1$, in which case $f_s^{(n)} = f^{(n)}$. For higher-order polynomials, the derivatives will be smoothed, but there will be no errors from filter truncation.

It is easy to show that the D^n are equal to the standard central difference filters, e.g.,

$$D = (-1, 1) \quad (29)$$

$$D^2 = (1, -2, 1) \quad (30)$$

$$D^3 = (-1, 3, -3, 1). \quad (31)$$

For a pixel distance of h , the above filters are divided by h^n .

References

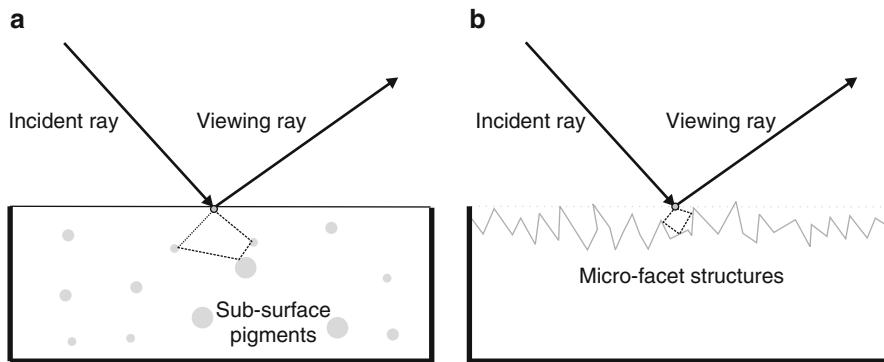
1. Bruckstein A, Rivlin E, Weiss I (1997) Scale space invariants for recognition. *Image Vis Comput* 15(5):335–344
2. Doermann D, Rivlin E, Weiss I (1996) Applying algebraic and differential invariants for logo recognition. *Mach Vis Appl* 9:73–86
3. Guggenheimer H (1963) Differential geometry. Dover, New York
4. Halphen M (1880) Sur les invariants différentiels des courbes gauches. *J Ec Polyt* 28(1)
5. Keren D, Rivlin E, Shimshoni I, Weiss I (2000) Recognizing 3D objects using tactile sensing and curve invariants. *J Math Imaging Vis* 12(1):5–23
6. Lane EP (1942) A treatise on projective differential geometry. University of Chicago Press, Chicago
7. Olver PJ (2000) Application of Lie groups to differential equations. Springer, New York
8. Pauwels EJ, Moons T, Van Gool LJ, Oosterlinck A (1995) Recognition of affine planar shapes under affine distortion. *Int J Comput Vis* 14:49–65
9. Rivlin E, Weiss I (1995) Local invariants for recognition. *IEEE Trans Pattern Anal Mach Intell (PAMI)* 17(3): 226–238
10. Springer CE (1964) Geometry and analysis of projective spaces. Freeman, San Francisco
11. Weiss I (1988) Projective invariants of shape. *Proceedings of computer vision and image processing*, Ann Arbor, pp 291–297
12. Weiss I (1993) Noise resistant invariants of curves. *IEEE Trans Pattern Anal Mach Intell* 15(9):943–948
13. Weiss I (1994) High order differentiation filters that work. *IEEE Trans Pattern Anal Mach Intell* 16(7):734–739
14. Weiss I (1994) Invariants for recovering shape from shading. In: Mundy J, Zisserman A (eds) Applications of invariance in computer vision II. Springer verlag lecture notes in computer science, vol 825. Springer, Berlin/New York, p 185
15. Weiss I (1999) Model-based recognition of 3D curves from one view. *J Math Imaging Vis* 10:1–10
16. Weiss I, Ray M (2001) Model-based recognition of 3D objects from single images. *IEEE Trans Pattern Anal Mach Intell* 23(2):116–128
17. Wilczynski EJ (1906) Projective differential geometry of curves and ruled surfaces. Teubner, Leipzig
18. Wilczynski EJ (1908) Projective differential geometry of curved surfaces (Second Memoir). *Am Math Soc Trans* 9:79

Diffuse Reflectance

Sanjeev J. Koppal
Harvard University, Cambridge, MA, USA

Synonyms

Diffuse shading; Diffusely scattered reflectance; [Lambertian reflectance](#)



Diffuse Reflectance, Fig. 1 The two diagrams depict diffuse reflection for a scene point with a vertical surface normal. In (a), color pigments under a smooth, non-specular surface scatter

the incident illumination back into the viewing direction. In (b), reflection off microfacets can also cause diffuse reflection, even if the individual microfacets exhibit specular reflection

Related Concepts

► [Lambertian Reflectance](#)

Definition

Diffuse reflection is a qualitative term to describe the relative, low-pass responses of different materials to incident illumination.

Background

Diffuse reflection implies that, without scene priors, it is generally impossible to accurately recover the incident illumination on the scene from measured intensities, even if the surface geometry, viewing direction, and object BRDF are known. A large class of materials can be described by diffuse reflectance models which, although usually low parametric, are diverse in nature and do not follow any fixed analytic form.

Theory

Ramamoorthi et al. [1] have described the diffuse reflectance as containing lower frequencies in the appropriatedualspace. Generally, this implies that a low number of bases can fully describe the diffuse reflectance function, although the exact number depends on the analytic form of the reflectance function.

Diffuse reflectances from smooth non-specular surfaces are modeled as rays of light that penetrate into the surface of a material, reflecting off of and being absorbed by flakes of color pigments, as shown in [Fig. 1a](#). Such a physical explanation describes the popular Lambertian model [2] and a variety of extensions such as through microfacets [3]. Rough specular surfaces can also exhibit diffuse reflection and these can be accounted for through microfacet modeling [4] as shown in [Fig. 1b](#).

Since diffuse reflection is a subjective term, many scene effects that exhibit the property of “blurring” the incident illumination are commonly termed as diffuse effects. These include global illumination effects in the scene such as sub-surface scattering and inter-reflections in skin and marble [5–7]. Strictly speaking, however, such effects are high parametric and global in nature and should not be incorrectly modeled as local, diffuse reflectance.

Computational Tractability

The main popularity of diffuse reflectance models is that they are low parametric and therefore have been used for inferring scene geometry (photometric stereo [8], shape from shading [9]). The high-frequency and global effects of the scene are modeled as noise in these approaches and usually ignored. Diffuse reflection has been shown as a good prior for scene recovery [10, 11]. Psychophysical experiments have also confirmed that humans can accurately estimate scene and illumination

properties [12], and much work has been done to replicate this for machine vision, tracing its roots back to retinex theory [13].

References

1. Ramamoorthi R, Hanrahan P (2001) A signal-processing framework for inverse rendering. In: SIGGRAPH, Los Angeles
2. Lambert JH (1760) Photometria sive de mensura de gratibus luminis, colorum et umbrae. Eberhard Klett, Augsburg
3. Oren M, Nayar SK (1995) Generalization of the lambertian model and implications for machine vision. Int J Comput Vis 14(3):227–251
4. Torrance K, Sparrow E (1967) Theory for off-specular reflection from roughened surfaces. J Opt Soc Am 57(9):1105–1112
5. Goesele M, Lensch HPA, Lang J, Fuchs C, Seidel H (2004) Acquisition of translucent objects. In: SIGGRAPH, Los Angeles
6. Jensen WH, Marschner RS, Levoy M, Hanrahan P (2001) A practical model for subsurface light transport. In: SIGGRAPH, Los Angeles
7. Debevec P, Hawkins T, Tchou C, Duiker H, Sarokin W, Sagar M (2000) Acquiring the reflectance field of a human face. In: SIGGRAPH, New Orleans
8. Woodham RJ (1978) Photometric stereo. MIT AI Memo, Cambridge
9. Horn B (1989) Height and gradient from shading. Int J Comput Vis. 5(1):37–75
10. Georgiades A (2003) Incorporating the torrance and sparow model of reflectance in uncalibrated photometric stereo. In: ICCV, Nice
11. van Ginneken B, Stavridi M, Koenderink JJ (1998) Diffuse and specular reflectance from rough surfaces. Appl opt 37(1):130–139
12. Adelson EH, Pentland AP (1996) The perception of shading and reflectance. Cambridge University Press, New York
13. Land EH, McCann JJ (1971) Lightness and retinex theory. J Opt Soc Am 61(1):1–11

Diffuse Shading

► [Diffuse Reflectance](#)

Diffusely Scattered Reflectance

► [Diffuse Reflectance](#)

Diffusion

► [Diffusion Filtering](#)

Diffusion Filtering

Thomas Brox

Department of Computer Science, University of Freiburg, Freiburg, Germany

Synonyms

[Anisotropic diffusion](#); [Diffusion](#); [Diffusion PDEs](#); [Nonlinear diffusion](#)

Related Concepts

► [Denoising](#)

Definition

Diffusion filtering is a smoothing technique motivated by equilibration processes in physics that allows for structure-preserving smoothing.

Background

Many problems in computer vision require smoothing or spatial aggregation of information. The most basic problem is image denoising, where noise ought to be removed from an image while preserving its relevant structures. Diffusion filtering is one among several methodologies that are capable of removing noise without blurring and dislocating image edges. The strong relationships to other filters with similar capabilities, such as wavelet shrinkage, the bilateral filter, and variational regularization, have been shown in various papers [1–3].

Theory

Diffusion filtering is motivated by the concept of diffusion in physics. Diffusion leads to equilibration of concentrations, for instance, in fluids. It is by definition a mass preserving process. Mass can move to locally change the concentration levels, but it cannot appear or disappear.

In diffusion, a concentration gradient ∇u creates a flux j as described by Fick's law:

$$j = -D \nabla u, \quad (1)$$

where D is a positive definite, symmetric diffusion matrix, which describes the magnitude and orientation preference of the flux. Combining Fick's law with the mass preservation property leads to the diffusion partial differential equation (PDE)

$$\frac{\partial u}{\partial t} = -\operatorname{div} j = \operatorname{div}(D \nabla u). \quad (2)$$

In image processing, we can interpret image intensities as concentrations and use the same diffusion PDE to smooth these image intensities. The initial condition is set by the input image

$$u(x, y, t = 0) = I(x, y). \quad (3)$$

For $t \rightarrow \infty$, diffusion yields the equilibrium state, i.e., a constant value image, where the value is the mean of the input image.

Depending on the choice of the diffusion matrix D , one can distinguish different cases:

- If D is the identity matrix, we obtain homogenous diffusion, where mass is equally distributed in all directions with a magnitude that only depends on the gradient. The corresponding PDE is also known as the heat equation, and there exists an analytic solution, which corresponds to convolution with a Gaussian kernel with variance $\sigma^2 = 2t$.
- If D is a diagonal matrix with identical entries, it degenerates to a scalar g , often called diffusivity,

and leads to isotropic diffusion. There is no orientation preference, but the magnitude of the flux may vary from point to point independent of the gradient. The diffusivity can be chosen as a decreasing function of the image gradient, for instance,

$$g(|\nabla I|) = \frac{1}{|\nabla I| + 1}, \quad (4)$$

which leads to a reduction of diffusion across edges and hence edge-preserving smoothing.

- In case of a general matrix D , the diffusion process is called anisotropic diffusion, as there is also a preference with regard to the orientation of smoothing. For instance, it is quite common to smooth along edges, but not across them. There is some confusion about terminology in the literature, as many papers, among them the original work on diffusion filtering by Perona and Malik [4], speak of anisotropic diffusion, although they use a scalar diffusivity rather than the full diffusion tensor.

Diffusion filters can further be separated into linear and nonlinear diffusion filters. In case of linear filters, the diffusion matrix D or diffusivity g does not depend on the smoothed image u but on some external quantity, usually the gradient of the input image I . This leads to PDEs that are linear in u . In case of nonlinear diffusion filters, the diffusion matrix or diffusivity depends on u and thus lead to nonlinear PDEs. Usually, nonlinear diffusion filters should be preferred as they take the enhanced image u into account when defining the areas where smoothing should be reduced, whereas linear diffusion relies on noisy measurements in the input image.

The diffusion PDEs can be solved numerically using iterative schemes. Most common is an explicit



Diffusion Filtering, Fig. 1 Left: noisy color image. Right: result after applying nonlinear isotropic diffusion

time discretization, which leads to the update equation

$$u^{k+1} = u^k + \tau \operatorname{div}(D(u^k) \nabla u^k), \quad (5)$$

where τ is a time step size, which must not be too large for the scheme to be stable. Details about discretization, stability, and many other properties of diffusion filters can be found in [5]. For fast implementations, it is also recommended to look into the works on AOS and FED schemes [6, 7].

Application

Diffusion filters are mainly used for image denoising. The corresponding PDEs, however, can be found in numerous other computer vision applications, especially in the context of variational methods. The Euler-Lagrange equation of the regularizer in variational models leads to a diffusion PDE of the general form

$$\operatorname{div}(D \nabla u) = 0, \quad (6)$$

where u in this case is the modeled function that should minimize the variational energy. Very popular is the total variation regularizer [8]

$$\int |\nabla u| dx dy \quad (7)$$

the Euler-Lagrange equation of which is

$$\operatorname{div}\left(\frac{\nabla u}{|\nabla u|}\right) = 0. \quad (8)$$

A (linear) diffusion process is implicitly also part of Laplacian eigenmaps and spectral clustering [9, 10].

Experiment

Diffusion processes can be easily extended to vector-valued images, such as color images. Figure 1 shows a noisy color image and the enhanced version after applying total variation flow

$$\frac{\partial u_k}{\partial t} = \operatorname{div}\left(\frac{\nabla u_k}{\sqrt{\sum_{l=1}^3 |\nabla u_l|^2}}\right). \quad (9)$$

References

1. Mrázek P, Weickert J, Steidl G (2005) Diffusion-inspired shrinkage functions and stability results for wavelet denoising. *Int J Comput Vis* 64(2/3):171–186
2. Didas S, Weickert J (2006) From adaptive averaging to accelerated nonlinear diffusion filtering. In: Proceedings of DAGM, Berlin. LNCS, vol 4174. Springer, Berlin/New York, pp 101–110
3. Scherzer O, Weickert J (2000) Relations between regularization and diffusion filtering. *J Math Imaging Vis* 12(1):43–63
4. Perona P, Malik J (1990) Scale space and edge detection using anisotropic diffusion. *IEEE Trans Pattern Anal Mach Intell* 12:629–639
5. Weickert J (1998) Anisotropic diffusion in image processing. Teubner, Stuttgart
6. Weickert J, ter Haar Romeny BM, Viergever MA (1998) Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Trans Image Process* 7(3):398–410
7. Grewenig S, Weickert J, Bruhn A (2010) From box filtering to fast explicit diffusion. In: Proceedings of DAGM, Darmstadt. LNCS, vol 6376. Springer, pp 533–542
8. Rudin LI, Osher S, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. *Phys D* 60:259–268
9. Shi J, Malik J (2000) Normalized cuts and image segmentation. *IEEE Trans Pattern Anal Mach Intell* 22(8):888–905
10. Coifman RR, Lafon S (2006) Diffusion maps. *Appl Comput Harmonic Anal* 21(1):5–30

Diffusion PDEs

► Diffusion Filtering

Digital Matting

► Matte Extraction

Digitization

Peer Stelldinger¹ and Longin Jan Latecki²

¹Computer Science Department, University of Hamburg, Hamburg, Germany

²Department of Computer and Information Sciences, Temple University, Philadelphia, PA, USA

Synonyms

Relation between objects and their digital images

Definition

Digitization is a mathematical model of converting continuous subsets of the plane or space (representing real objects) to digital sets in \mathbb{Z}^2 or \mathbb{Z}^3 or similar grids (representing segmented images of these objects).

Background

A fundamental task of knowledge representation and processing is to infer properties of real objects or situations given their representations. In spatial knowledge representation and, in particular, in computer vision and medical imaging, real objects are represented in a pictorial way as finite and discrete sets of pixels or voxels. The discrete sets result by a quantization process, in which real objects are approximated by discrete sets. In computer vision, this process is called sampling or digitization and is naturally realized by technical devices like computer tomography scanners, CCD cameras or document scanners. Digital images obtained by digitization are suitable to estimate the real object properties like volume and surface area. Therefore, a fundamental question addressed in spatial knowledge representation is: Which properties inferred from discrete representations of real objects correspond to properties of their originals, and under what conditions this is the case? While this problem is well-understood in the 2D case with respect to topology [1–5], it is not as simple in 3D, as shown in [7]. Only recently a first comprehensive answer to this question with respect to important topological and geometric properties of 3D objects has been presented in [8, 9].

The description of geometric and, in particular, topological features in discrete structures is based on graph theory, which is widely accepted in the computer science community. A graph is obtained when a neighborhood relation is introduced into a discrete set, e.g., a finite subset of \mathbb{Z}^2 or \mathbb{Z}^3 , where \mathbb{Z} denotes the integers. On the one hand, graph theory allows investigation into connectivity and separability of discrete sets (for a simple and natural definition of connectivity see [10, 16], for example). On the other hand, a finite graph is an elementary structure that can be easily implemented on computers. Discrete representations are analyzed by algorithms based on graph theory, and the properties

extracted are assumed to represent properties of the original objects. Since practical applications, for example in image analysis, show that this is not always the case, it is necessary to relate properties of discrete representations to the corresponding properties of the originals. Since such relations can describe and justify the algorithms on discrete graphs, their characterization contributes directly to the computational investigation of algorithms on discrete structures. This computational investigation is an important part of the research in computer science, and in particular, in computer vision (Marr [11]), where it can contribute to the development of more suitable and reliable algorithms for extracting required shape properties from discrete representations.

It is clear that no discrete representation can exhibit all features of the real original. Thus one has to accept compromises. The compromise chosen depends on the specific application and on the questions which are typical for that application. Real objects and their spatial relations can be characterized using geometric features. Therefore, any useful discrete representation should model the geometry faithfully in order to avoid false conclusions. Topology deals with the invariance of fundamental geometric features like connectivity and separability. Topological properties play an important role, since they are the most primitive object features and human visual system seems to be well-adapted to cope with topological properties.

However, one does not have any direct access to spatial properties of real objects. Therefore, real objects are represented as bounded subsets of the Euclidean space \mathbb{R}^3 , and their 2D views (projections) as bounded continuous subsets of the plane \mathbb{R}^2 . Hence, from the theoretical point of view of knowledge representation, the goal is to relate two different pictorial representations of objects in the real world: a discrete and a continuous representation.

Already two of the first books in computer vision deal with the relation between the continuous object and its digital images obtained by modeling a digitization process. Pavlidis [1] and Serra [2] proved independently in 1982 that an r -regular continuous 2D set S (the definition follows below) and the continuous analog of the digital image of S have the same shape in a topological sense. Pavlidis used 2D square grids and Serra used 2D hexagonal sampling grids.

In 3D this problem is much more complicated. In 2005 it has been shown in [7] that the connectivity properties are preserved when digitizing a 3D r -regular object with a sufficiently dense sampling grid, but the preservation of connectivity is much weaker than topology. Stelldinger and Köthe [7] also found out that topology preservation can even not be guaranteed with sampling grids of arbitrary density if one uses the straightforward voxel reconstruction, since the surface of the continuous analog of the digital image may not be a 2D manifold. The question how to guarantee topology preservation during digitization in 3D remained unsolved until 2007.

The solution was provided in [8], where the same digitization model as Pavlidis and Serra is used, also r -regular sets (but in \mathbb{R}^3) are used to model the continuous objects. As already shown in [7] the generalization of Pavlidis' straightforward reconstruction method to 3D fails since the reconstructed surface may not be a 2D manifold. For example, Fig. 1a, b shows a continuous object and its digital reconstruction whose surface is not a 2D manifold. However, it is possible to use several other reconstruction methods that all result in a 3D object with a 2D manifold surface. Moreover it is also shown in [8] that these reconstructions and the original continuous object are homeomorphic and their surfaces are close to each other.

The first reconstruction method, Majority interpolation, is a voxel-based representation on a grid with doubled resolution. It always leads to a well-composed set in the sense of Latecki [12], which implies that a lot of problems in 3D digital geometry become relatively simple.

The second method is the most simple one. It just uses balls with a certain radius instead of cubical voxels. When choosing an appropriate radius the topology of an r -regular object will not be destroyed during digitization.

The third method is a modification of the well-known Marching Cubes algorithm [13]. The original Marching Cubes algorithm does not always construct a topologically sound surface due to several ambiguous cases [14, 15]. As shown in [8, 9] most of the ambiguous cases can not occur in the digitization of an r -regular object and that the only remaining ambiguous case always occurs in an unambiguous way, which can be dealt with by a slight modification of the original Marching Cubes algorithm. Thus the generated surface is not only topologically sound, but it also has

exactly the same topology as the original object before digitization. Moreover it is shown that one can use trilinear interpolation and that one can even blend the trilinear patches smoothly into each other such that one gets smooth object surfaces with the correct topology. Each of these methods has its own advantages making the presented results applicable to many different image analysis algorithms.

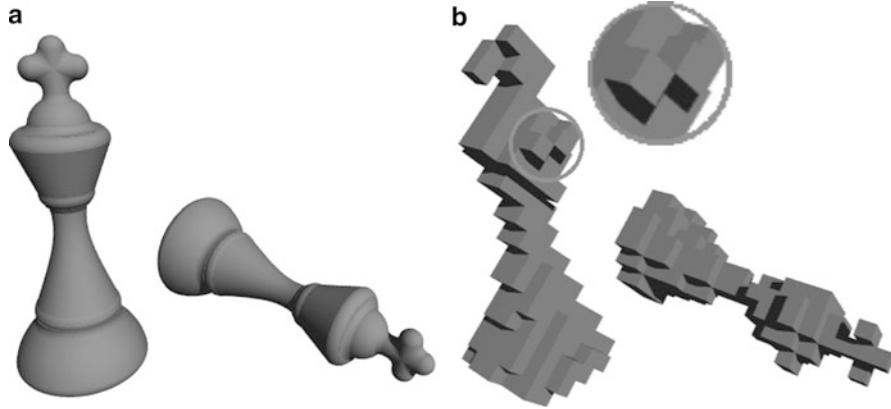
Theory

The (Euclidean) *distance* between two points x and y in \mathbb{R}^n is denoted by $d(x, y)$, and the (Hausdorff) *distance between two subsets of \mathbb{R}^n* is the maximal distance between each point of one set and the nearest point of the other. Let $A \subset \mathbb{R}^n$ and $B \subset \mathbb{R}^m$ be sets. A function $f : A \rightarrow B$ is called *homeomorphism* if it is bijective and both it and its inverse are continuous. If f is a homeomorphism, then A and B are *homeomorphic*. Let A, B be two subsets of \mathbb{R}^n (particularly, $n = 2$ or 3). Then a homeomorphism $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ such that $f(A) = B$ and $d(x, f(x)) \leq r$, for all $x \in \mathbb{R}^n$, is called an r -*homeomorphism* of A to B and A and B are r -*homeomorphic*. A *Jordan curve* is a set $J \subset \mathbb{R}^n$ which is homeomorphic to a circle. Let A be any subset of \mathbb{R}^n . The *complement* of A is denoted by A^c . All points in A are *foreground* while the points in A^c are called *background*. The *open ball* in \mathbb{R}^n of radius r and center c is the set $\mathcal{B}_r^0(c) = \{x \in \mathbb{R}^n \mid d(x, c) < r\}$, and the *closed ball* in \mathbb{R}^n of radius r and center c is the set $\bar{\mathcal{B}}_r(c) = \{x \in \mathbb{R}^n \mid d(x, c) \leq r\}$. The *boundary* of A , denoted ∂A , consists of all points $x \in \mathbb{R}^n$ with the property that if B is any open set of \mathbb{R}^n such that $x \in B$, then $B \cap A \neq \emptyset$ and $B \cap A^c \neq \emptyset$.

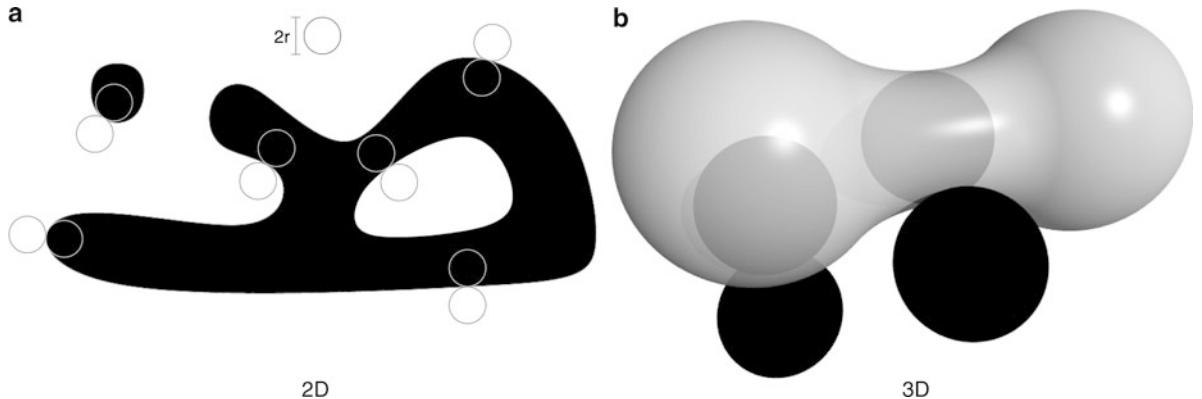
An open ball $\mathcal{B}_r^0(c)$ is *tangent* to ∂A at a point $x \in \partial A$ if $\partial A \cap \partial \mathcal{B}_r^0(c) = \{x\}$. An open ball $\mathcal{B}_r^0(c)$ is an *osculating open ball of radius r to ∂A at point $x \in \partial A$* if $\mathcal{B}_r^0(c)$ is tangent to ∂A at x and either $\mathcal{B}_r^0(c) \subseteq A^0$ or $\mathcal{B}_r^0(c) \subseteq (A^c)^0$.

Definition 1 A set $A \subset \mathbb{R}^n$ is called r -*regular* if, for each point $x \in \partial A$, there exist two osculating open balls of radius r to ∂A at x such that one lies entirely in A and the other lies entirely in A^c . Examples illustrating 2D and 3D cases are shown in Fig. 2.

Note that the boundary of a 3D r -regular set is a 2D manifold surface. Any set S which is a translated and rotated version of the set $\frac{2r'}{\sqrt{3}}\mathbb{Z}^3$ is called a *cubic*



Digitization, Fig. 1 The digital reconstruction (b) of an r -regular object (a) may not be well-composed, i.e., its surface may not be a 2D manifold as can be seen in the magnification



Digitization, Fig. 2 For each boundary point of a 2D/3D r -regular set exists an outside and an inside osculating open r -disc/ball

r' -grid and its elements are called *sampling points*. Note that the distance $d(x, p)$ from each point $x \in \mathbb{R}^3$ to the nearest sampling point $s \in S$ is at most r' . The *voxel* $\mathcal{V}_S(s)$ of a sampling point $s \in S$ is its *Voronoi region* \mathbb{R}^3 : $\mathcal{V}_S(s) = \{x \in \mathbb{R}^3 \mid d(x, s) \leq d(x, q), \forall q \in S\}$, i.e., $\mathcal{V}_S(s)$ is the set of all points of \mathbb{R}^3 which are at least as close to s as to any other point in S . In particular, note that $\mathcal{V}_S(s)$ is a cube whose vertices lie on a sphere of radius r' and center s .

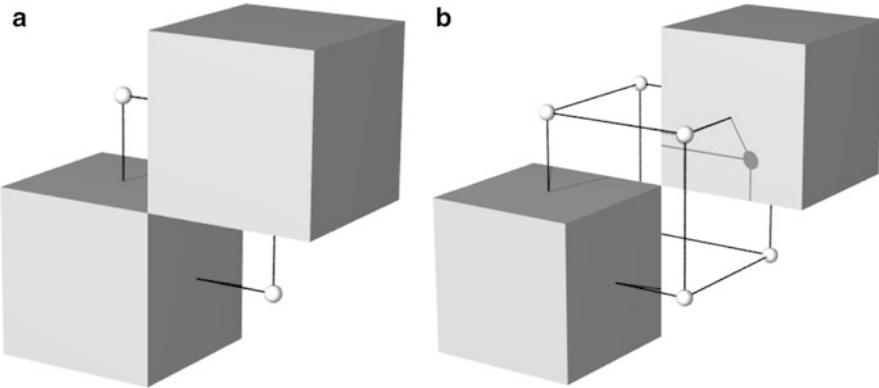
Definition 2 Let S be a cubic r' -grid, and let A be any subset of \mathbb{R}^3 . The union of all voxels with sampling points lying in A is the digital reconstruction of A with respect to S , $\hat{A} = \bigcup_{s \in (S \cap A)} \mathcal{V}_S(s)$.

This method for reconstructing the object from the set of included sampling points is the 3D generalization of the 2D *Gauss digitization* (see [16]) which

has been used by Gauss to compute the area of discs and which has also been used by Pavlidis [1] in his sampling theorem.

For any two points p and q of S , $\mathcal{V}_S(p) \cap \mathcal{V}_S(q)$ is either empty or a common vertex, edge or face of both. If $\mathcal{V}_S(p) \cap \mathcal{V}_S(q)$ is a common face, edge, or vertex, then $\mathcal{V}_S(p)$ and $\mathcal{V}_S(q)$ are *face-adjacent*, *edge-adjacent*, or *vertex-adjacent*, respectively. Two voxels $\mathcal{V}_S(p)$ and $\mathcal{V}_S(q)$ of \hat{A} are *connected in \hat{A}* if there exists a sequence $\mathcal{V}_S(s_1), \dots, \mathcal{V}_S(s_k)$, with $k \in \mathbb{Z}$ and $k > 1$, such that $s_1 = p$, $s_k = q$, and $s_i \in A$ (or equivalently, $\mathcal{V}_S(s_i) \subset \hat{A}$), for each $i \in \{1, \dots, k\}$, and $\mathcal{V}_S(s_j)$ and $\mathcal{V}_S(s_{j+1})$ are face-adjacent, for each $j \in \{1, \dots, k-1\}$. A (connected) *component* of \hat{A} is a maximal set of connected voxels in \hat{A} .

Definition 3 Let S be a cubic r' -grid, and let T be any subset of S . Then $\bigcup_{t \in T} \mathcal{V}_S(t)$ is well-composed if



Digitization, Fig. 3 (a) Critical configuration (C1). (b) Critical configuration (C2). For the sake of clarity, only the voxels of foreground or background points are shown

$\partial(\bigcup_{t \in T} \mathcal{V}_S(t))$ is a surface in \mathbb{R}^3 , or equivalently, if for every point $x \in \partial(\bigcup_{t \in T} \mathcal{V}_S(t))$, there exists a positive number r such that the intersection of $\partial(\bigcup_{t \in T} \mathcal{V}_S(t))$ and $B_r^0(x)$ is homeomorphic to the open unit disk in \mathbb{R}^2 , $\mathbb{D} = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 < 1\}$.

Well-composed digital reconstructions can be characterized by two local conditions depending only on voxels of points of S . Let s_1, \dots, s_4 be any four points of S such that $\bigcap_{i=1}^4 \mathcal{V}_S(s_i)$ is a common edge of $\mathcal{V}_S(s_1), \dots, \mathcal{V}_S(s_4)$. The set $\{\mathcal{V}_S(s_1), \dots, \mathcal{V}_S(s_4)\}$ is an instance of the *critical configuration (C1)* with respect to $\bigcup_{t \in T} \mathcal{V}_S(t)$ if two of these voxels are in $\bigcup_{t \in T} \mathcal{V}_S(t)$ and the other two are in $(\bigcup_{t \in T} \mathcal{V}_S(t))^c$, and the two voxels in $\bigcup_{t \in T} \mathcal{V}_S(t)$ (resp. $(\bigcup_{t \in T} \mathcal{V}_S(t))^c$) are edge-adjacent, as shown in Fig. 3a. Now, let s_1, \dots, s_8 be any eight points of S such that $\bigcap_{i=1}^8 \mathcal{V}_S(s_i)$ is a common vertex of $\mathcal{V}_S(s_1), \dots, \mathcal{V}_S(s_8)$. The set $\{\mathcal{V}_S(s_1), \dots, \mathcal{V}_S(s_8)\}$ is an instance of the *critical configuration (C2)* with respect to $\bigcup_{t \in T} \mathcal{V}_S(t)$ if two of these voxels are in $\bigcup_{t \in T} \mathcal{V}_S(t)$ (resp. $(\bigcup_{t \in T} \mathcal{V}_S(t))^c$) and the other six are in $(\bigcup_{t \in T} \mathcal{V}_S(t))^c$ (resp. $\bigcup_{t \in T} \mathcal{V}_S(t)$), and the two voxels in $\bigcup_{t \in T} \mathcal{V}_S(t)$ (resp. $(\bigcup_{t \in T} \mathcal{V}_S(t))^c$) are vertex-adjacent, as shown in Fig. 3b. The following theorem from [12] establishes an important equivalence between well-composedness and the (non)existence of critical configurations (C1) and (C2):

Theorem 2 ([12]) *Let S be a cubic r' -grid and let T be any subset of S . Then, $\bigcup_{t \in T} \mathcal{V}_S(t)$ is well-composed iff the set of voxels $\{\mathcal{V}(s) \mid s \in S\}$ does not contain any instance of the critical configuration (C1)*

nor any instance of the critical configuration (C2) with respect to $\bigcup_{t \in T} \mathcal{V}_S(t)$.

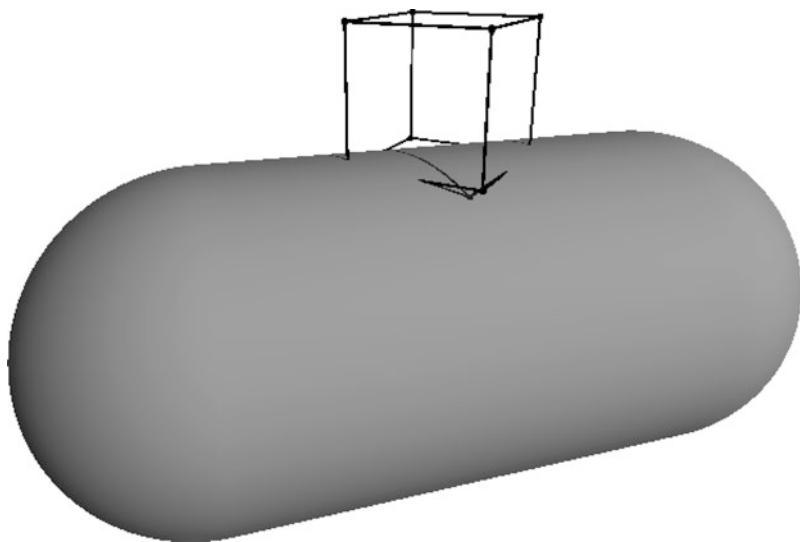
A simple consequence of the 2D digitization theorem by Pavlidis [1] is that the reconstruction of an r' -regular set is well-composed. The main difficulty of 3D digitization as compared to 2D lies in the fact that digital reconstruction \hat{A} of A with respect to S is not guaranteed to be well-composed. An example is provided in Fig. 4. Therefore, it is necessary to repair \hat{A} in order to ensure the topological equivalence between A and repaired \hat{A} . The first topology preserving repairing method has been proposed in [8], where also the following theorem is proven. It is an interesting observation that it took 25 years to obtain this 3D theorem.

Theorem 3 ([8]) *If A is an r -regular object and S is a cubic r' -grid with $2r' < r$, then the result of the topology preserving repairing of the reconstruction \hat{A} is r -homeomorphic to A .*

Application

A complete understanding of the loss of information due to the digitization process is fundamental for the justification of any computer vision application. If the relevant information is not contained in the digital image, there is no way to reconstruct it without using context knowledge. Thus, whenever one needs to have guarantees for the correct behavior of some computer vision algorithm one has to be aware of what happens during digitization. This article gives an exemplary

Digitization, Fig. 4 The surface of an object only needs to have an arbitrarily small, but nonzero curvature in order to make occurrences of the critical configuration (C1) possible in the digital reconstruction



insight to the topic, the related problems and the way to solve them.

Open Problems

The analysis of the effect of digitization to the information being extractable from an image is a challenging research area. Newest results approximate real acquisition processes and thus give direct implications for many computer vision algorithms which rely on precise information of the structures being approximated by the digital image. However, in reality the digitization process is still more complicated than the models which are used for topological or geometric sampling theorems. The goal is to derive guarantees for digitization models approximating real digitization processes.

References

- Pavlidis T (1982) Algorithms for graphics and image processing. Computer Science Press, Cambridge
- Serra J (1982) Image analysis and mathematical morphology. Academic, San Diego
- Latecki LJ, Conrad C, Gross A (1998) Preserving topology by a digitization process. *J Math Imaging Vis* 8:131–159
- Latecki LJ (1998) Discrete representation of spatial objects in computer vision. Viergever M (ed) Series on computational imaging and vision. Kluwer, Dordrecht
- Tajine M, Ronse C (2002) Topological properties of hausdorff discretization, and comparison to other discretization schemes. *Theor Comput Sci* 283(1):243–268
- Stelldinger P, Köthe U (2003) Shape preservation during digitization: tight bounds based on the morphing distance. In: Pattern recognition, DAGM 2003, Munich, pp 108–115
- Stelldinger P, Köthe U (2005) Towards a general sampling theory for shape preservation. *Image Vis Comput J* 23(2):237–248
- Stelldinger P, Latecki LJ, Siqueira M (2007) Topological equivalence between a 3D object and the reconstruction of its digital image. *IEEE Trans Pattern Anal Mach Intell* 29(1):126–140
- Stelldinger P (2008) Image digitization and its influence on shape properties in finite dimensions. DISKI 312, IOS, Amsterdam
- Kong T, Rosenfeld A (1990) If we use 4- or 8-connectedness for both the objects and the background, the Euler characteristics is not locally computable. *Pattern Recognit Lett* 11(4):231–232
- Marr D (1983) Vision. Freeman, San Francisco
- Latecki LJ (1997) 3D well-composed pictures. *Graph Models Image Process* 59(3):164–172
- Lorensen WE, Cline HE (1987) Marching cubes: a high resolution 3D surface construction algorithm. In: SIGGRAPH '87: proceedings of the 14th annual conference on computer graphics and interactive techniques. ACM, New York, pp 163–169
- Dürst MJ (1988) Letters: additional reference to “marching cubes”. *SIGGRAPH Comput Graph* 22(2): 243
- Nielson GM, Hamann B (1991) The asymptotic decider: resolving the ambiguity in marching cubes. In: Proceedings of the 2nd IEEE conference on visualization (Visualization'91), San Diego, pp 83–91
- Klette R, Rosenfeld A (2004) Digital geometry. Morgan Kaufman, Amsterdam
- Siqueira M, Latecki LJ, Tustison N, Gallier J, Gee J (2008) Topological repairing of 3D digital images. *J Math Imaging Vis* 30(3):249–274

Dimension Reduction

► Dimensionality Reduction

Dimensional Compression

► Dimensionality Reduction

Dimensional Embedding

► Dimensionality Reduction

Dimensionality Reduction

Eisaku Maeda
NTT Communication Science Laboratories,
Soraku-gun, Kyoto, Japan

Synonyms

Dimensional compression; Dimensional embedding;
Dimension reduction

Related Concepts

► Feature Selection

Definition

Dimensionality reduction is the process of reducing the dimension of the vector space spanned by feature vectors (pattern vectors). Various kinds of reduction can be achieved by defining a map from the original space into a dimensionality-reduced space.

Background

The feature space, i.e., the vector space spanned by feature vectors (pattern vectors) defined on d -dimensional space, can be transformed into a vector space of

lower-dimension $d' (< d)$ spanned by d' -dimensional feature vectors through linear or nonlinear transformation. This transformation allows feature vectors to be represented by lower-dimensional vectors, and various kinds of vector operations and statistical analysis, such as multivariate analysis, machine learning, clustering, and classification, become less expensive to perform. Moreover, it tackles the “curse of dimensionality,” the various problems created by dealing with data of high dimensionality.

Theory and Application

The most common linear approaches for dimensionality reduction are principal component analysis (PCA) and linear discriminant analysis (LDA). PCA, LDA, and related techniques have been used for pattern recognition and computer vision. It is generally considered that PCA offers reduction for information representation while LDA offers reduction for classification (Fig. 1).

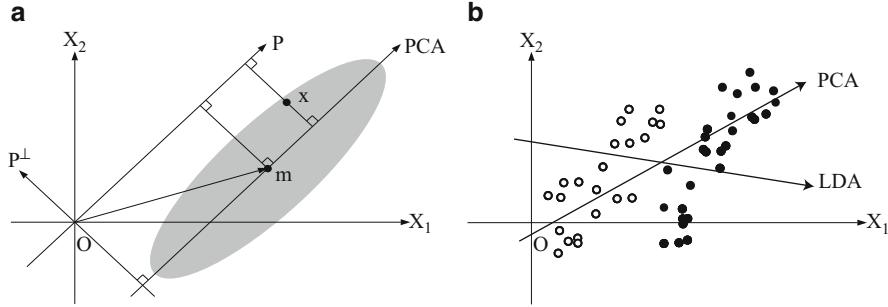
PCA finds principal axes so as to maximize the variance of the pattern vectors’ distribution in a dimensionality-reduced space. It is theoretically identical to a discrete and finite case of Karhunen-Loeve transform. In PCA, d' principal axes are obtained as eigenvectors $\Phi_1, \dots, \Phi_{d'}$, which are eigenvectors corresponding to the d' largest eigenvalues of Σ , $\lambda_1, \dots, \lambda_{d'}$. Σ is the covariance matrix of feature vectors and is defined as

$$\Sigma = \frac{1}{n} \sum_{\mathbf{x}} (\mathbf{x} - \mathbf{m})(\mathbf{x} - \mathbf{m})^T \quad (1)$$

where \mathbf{x} are d -dimensional feature vectors, n is the number of feature vectors, and \mathbf{m} is the mean of feature vectors (i.e., $\mathbf{m} = \frac{1}{n} \sum \mathbf{x}$).

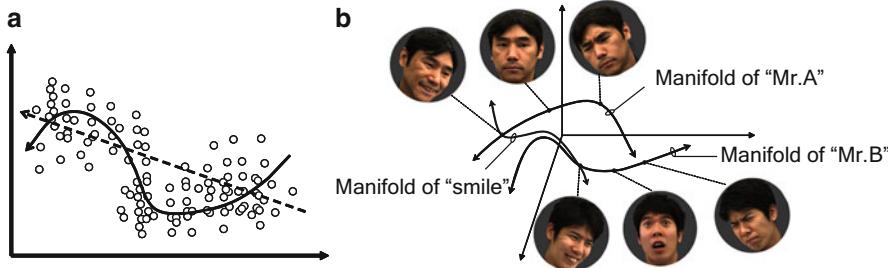
LDA realizes a linear transform into $(c - 1)$ -dimensional space so as to maximize the between-class distance, which is the distance between the pattern distribution of each class; c is the number of classes (categories). The bases of the transformed space are obtained as the eigenvectors of $\Sigma_W^{-1} \Sigma_B$, where Σ_W is within-class covariance matrix and Σ_B is the between-class covariance matrix. They are written as

$$\Sigma_W = \sum_{i=1}^c \left(\frac{1}{n} \sum_{\mathbf{x} \in \mathcal{X}_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T \right) \quad (2)$$



Dimensionality Reduction, Fig. 1 (a) Distribution of pattern vectors (gray region) and the 1st principal axis. (b) Distribution of 2-dimensional feature vectors classified as class 1

(open circle) and 2 (closed circle) and the principal axes determined by PCA and LDA



Dimensionality Reduction, Fig. 2 (a) Distribution of 2-dimensional pattern vectors and a principal axis determined by linear (solid line) and nonlinear (dotted line) dimensionality reduction. (b) Various embedded manifolds in a face space

$$\Sigma_B = \sum_{i=1}^c \frac{n_i}{n} (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T \quad (3)$$

where \mathbf{m}_i is the mean of the feature vectors that belong to i th class \mathcal{X}_i and \mathbf{m} represents the total mean of all feature vectors. As the rank of matrix Σ_B is $(c - 1)$, the dimension of the space determined by LDA is not more than $(c - 1)$.

PCA- and LDA-based techniques such as class-featuring information compression (CLAFIC) [1] and learning subspace method (LSM) [2] have been studied and widely applied to various pattern recognition tasks like character recognition and phoneme classification. More recently, they have been applied to more complicated tasks, such as face recognition [3, 4] and 3D object recognition [5].

Multidimensional scaling (MDS) is another classic technique for reconstructing a subspace to represent pattern distribution. Given only the dissimilarities (distances) between any pairs of patterns (objects, samples), MDS outputs basis vectors that minimize the distance errors in the subspace spanned by the

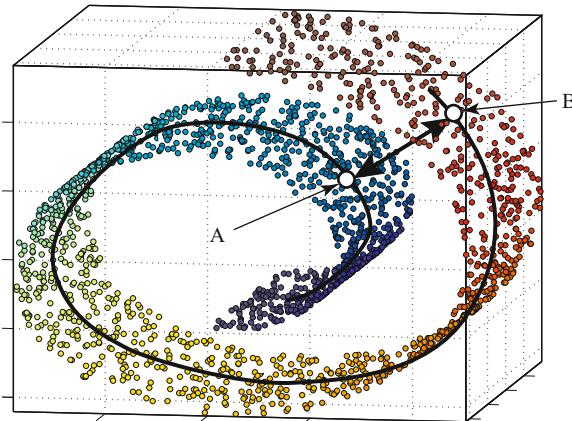
bases. Distance matrix \mathbf{D} , whose (i, j) element is the distance between \mathbf{x}_i and \mathbf{x}_j , is transformed by the Young-Householder transformation into matrix \mathbf{H} as follows:

$$\mathbf{H} = -\frac{1}{2}\mathbf{J}_n \mathbf{D} \mathbf{J}_n \quad (4)$$

$$\mathbf{J}_n = \mathbf{I}_n - \frac{1}{n}\mathbf{1}_n \quad (5)$$

where \mathbf{I}_n denotes an n -dimensional identity matrix and $\mathbf{1}_n$ denotes a square matrix, all of whose elements are 1. The principal axes minimizing the distance errors are obtained as the eigenvectors of matrix \mathbf{H} .

Nonlinear dimensionality reduction is a powerful tool for pattern recognition and computer vision despite of its higher computational cost than linear methods. Conventional linear dimensionality reduction methods have been extended to nonlinear equivalents through the kernel technique, i.e., kernel PCA [6] and kernel version of LDA [7, 8]. By replacing the inner products of feature vectors $(\mathbf{x}_i, \mathbf{x}_j)$ to $k(\mathbf{x}_i, \mathbf{x}_j)$ in the procedure of linear methods, where k is any



Dimensionality Reduction, Fig. 3 Euclidean distance between A and B (arrow) and geodesic distance (solid line) in 3-dimensional feature distribution

kernel function, various nonlinear transforms can be achieved depending on the kernel function selected and its parameters.

Many novel techniques of nonlinear dimensionality reduction have been proposed; they are collectively called manifold learning [9, 10]. Pioneering works of manifold learning are Isomap [11] and Locally Linear Embedding (LLE) [12, 13] (Fig. 2).

Isomap is a manifold learning algorithm that preserves the geodesic distance between all feature vectors. The distance is approximately obtained as the shortest path distance by tracking nearest neighbors. In Isomap, these pair-wise geodesic distances between feature vectors are applied to classic MDS (Fig. 3).

LLE, on the other hand, targets low-dimensional manifolds that preserve local geometric relationships between neighbors, so as to minimize cost function $E(\mathbf{w})$,

$$E(\mathbf{w}) = \sum_i \left(\mathbf{x}_i - \sum_j \mathbf{w}_{ij} \mathbf{x}_j \right)^2 \quad (6)$$

where \mathbf{x}_{ij} denote neighbors of \mathbf{x}_i and $v\mathbf{w}_{ij}$ denote the linear weights that satisfy $\sum_j \mathbf{w}_{ij} = 1$. Each point \mathbf{x}_i in the d -dimensional space is mapped onto point \mathbf{y}_i in the d' -dimensional space by minimizing the cost function $C(\mathbf{y})$,

$$C(\mathbf{y}) = \sum_i \left(\mathbf{y}_i - \sum_j \mathbf{w}_{ij} \mathbf{y}_j \right)^2 \quad (7)$$

where \mathbf{w}_{ij} are the weights that minimize $E(\mathbf{w})$. Isomap, LLE, and graph Laplacian [14] are variations of kernel PCA with special kernel functions [15].

Another nonlinear method is the feed-forward neural network trained to approximate the identity function [16, 17]. When the trained network has fewer hidden layer units than input and output layer units, the output signals of the hidden layer units can be considered as an encoded vector of the original input pattern vector.

References

- Watanabe S (1969) Knowing & guessing – quantitative study of inference and information. John Wiley & Sons Inc., Hoboken, NJ, USA
- Oja E (1983) Subspace methods of pattern recognition. Research Studies Press, Baldock, UK
- Turk M, Pentland A (1991) Eigenfaces for recognition. J Cogn Neurosci 3(1):71–86
- Belhumeur P, Hespanha J, Kriegman D (1997) Eigenfaces vs. fisherfaces: recognition using class specific linear projection. IEEE Trans Pattern Anal Mach Intell 19(7): 711–720
- Murase H, Nayar SK (1995) Visual learning and recognition of 3-d objects from appearance. Int J Comput Vis 14(1): 5–24
- Schölkopf B, Smola A, Müller KR (1998) Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput 10(5):1299–1319
- Mika S, Rätsch G, Weston J, Schölkopf B, Müller K (1999) Fisher discriminant analysis with kernels. In: Proceedings of IEEE neural networks for signal processing workshop IX (NNSP'99), Madison, pp 41–48
- Baudat G, Anouar F (2000) Generalized discriminant analysis using a kernel approach. Neural Comput 12:2385–2404
- Burges CJ (2005) Geometric methods for feature extraction and dimensional reduction. In: Maimon O, Rokach L (eds) Data mining and knowledge discovery handbook: a complete guide for researchers and practitioners. Springer, NY, USA
- Pless R, Souvenir R (2009) A survey of manifold learning for images. IPSJ Trans Comput Vis Appl 1:83–94
- Tenenbaum JB, de Silva V, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. Science 290(5500):2319–2323
- Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290: 2323–2326
- Saul LK, Roweis ST, Singer Y (2003) Think globally, fit locally: unsupervised learning of low dimensional manifolds. J Mach Learn Res 4:119–155

14. Belkin M, Niyogi P (2002) Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput* 15(6):1373–1396
15. Ham J, Lee DD, Mika S, Schölkopf B (2004) A kernel view of the dimensionality reduction of manifolds. In: Proceedings of the 21st international conference on machine learning (ICML'04), Banff, pp 369–376
16. DeMers D, Cottrell G (1992) Non-linear dimensionality reduction. In: Advances in Neural Information Processing Systems 5. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp 580–587
17. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507

Dioptics

► [Catadioptric Camera](#)

Discrete AdaBoost

► [AdaBoost](#)

Discriminative Random Fields

Sanjiv Kumar
Google Research, New York, NY, USA

Synonyms

[Conditional random fields](#); [Two dimensional conditional random fields](#)

Definition

Discriminative random fields (DRFs) are probabilistic discriminative graphical models for classification that allow contextual interactions among labels as well as the observed data using arbitrary discriminative classifiers.

Background

Natural image data shows significant dependencies that should be modeled appropriately to achieve good

classification of image entities such as pixels, regions, objects, or the entire image itself. Such dependencies are commonly referred to as *context* in vision. For example, due to local smoothness of natural images, neighboring pixels tend to have similar labels (except at the discontinuities). Different semantic regions in a scene follow plausible spatial configurations, for example, sky tends to occur above water or vegetation. For parts-based object recognition, different parts of an object such as handle, keypad, and front panel in a phone are related to each other through geometric constraints. In fact, the contextual interactions for object recognition are not limited to the parts of an object. These may include interactions among various objects themselves. For example, the presence of a monitor screen increases the probability of having a keyboard or a mouse nearby. The challenge is how to model these different types of context, which include complex dependencies in the observed image data as well as the labels, in a principled manner. Discriminative graphical models provide a solid platform to achieve that.

Traditional generative graphical models, that is, Markov random fields (MRFs) suffer from three main problems, which are overcome by DRFs: First, for computational tractability, the observations are assumed to be conditionally independent in MRFs. This assumption is too restrictive for many natural image analysis applications. Second, interaction among labels in MRFs arises from prior over labeling and hence do not depend on the observed data. This prohibits one from modeling data-dependent interactions in labels that are necessary for a variety of tasks. For example, while implementing local smoothness of labels in image segmentation, it may be desirable to use observed data to modulate the smoothness according to the image intensity gradients. Further, in parts-based object detection to model interactions among object parts, one needs observed data to enforce geometric constraints. DRFs allow interactions among labels based on unrestricted use of observations as necessary. This step is crucial to develop models that can incorporate interactions of different types within the same framework.

Finally, MRFs are used in a probabilistic generative framework that models the joint probability of the observed data and the corresponding labels. However, for classification purposes, one needs to estimate the posterior over labels given the observations, that is,

$P(\mathbf{x}|\mathbf{y})$, where \mathbf{y} is observed data and \mathbf{x} are corresponding labels. In a generative MRF framework, one expends efforts to model the joint distribution $p(\mathbf{x}, \mathbf{y})$, which involves implicit modeling of the observations. In a discriminative framework, one models the distribution $P(\mathbf{x}|\mathbf{y})$ directly. A major advantage of doing this is that the true underlying generative model may be quite complex even though the class posterior is simple. This means that the generative approach may spend a lot of resources on modeling the generative models which are not particularly relevant to the task of inferring the class labels. Moreover, learning the class density models may become even harder when the training data is limited. The discriminative approach saves one from making simplistic assumptions about the data.

Discriminative Random Field (DRF)

Discriminative random fields (DRFs) are discriminative graphical models based on conditional random fields (CRFs), originally proposed by Lafferty et al. [1] in the context of segmentation and labeling of 1D text sequences. CRFs are discriminative models that directly model the conditional distribution over labels, that is, $P(\mathbf{x}|\mathbf{y})$ as a Markov random field. This approach allows one to capture arbitrary dependencies between the observations without resorting to any model approximations. DRFs are 2D versions of 1D CRFs, which allow the use of arbitrary discriminative classifiers to model different types of interactions in labels and data, leading to more flexible and powerful generalization of CRFs. These were introduced in vision by Kumar and Hebert [2, 3].

Let the observed data from an input image be given by $\mathbf{y} = \{\mathbf{y}_i\}_{i \in S}$ where \mathbf{y}_i is the data from i^{th} site and $\mathbf{y}_i \in \mathcal{N}^c$. The corresponding labels at the image sites are given by $\mathbf{x} = \{x_i\}_{i \in S}$. In a binary classification problem, $x_i \in \{-1, 1\}$. It is easy to extend the formulation to multiclass labeling problems as mentioned later. The random variables \mathbf{x} and \mathbf{y} are jointly distributed, but in a discriminative framework, a conditional model $P(\mathbf{x}|\mathbf{y})$ is constructed from the observations and labels, and the marginal $p(\mathbf{y})$ is not modeled explicitly. DRFs follow the definition of CRFs given by Lafferty et al. [1].

Definition 1 CRF: Let $G = (S, E)$ be a graph such that \mathbf{x} is indexed by the vertices of G . Then (\mathbf{x}, \mathbf{y}) is

said to be a conditional random field if, when conditioned on \mathbf{y} , the random variables x_i obey the Markov property with respect to the graph: $P(x_i|\mathbf{y}, \mathbf{x}_{S-\{i\}}) = P(x_i|\mathbf{y}, \mathbf{x}_{\mathcal{N}_i})$, where $S - \{i\}$ is the set of all the nodes in the graph except the node i , \mathcal{N}_i is the set of neighbors of the node i in G , and \mathbf{x}_{Ω} represents the set of labels at the nodes in set Ω .

Thus, a DRF is a random field globally conditioned on the observations \mathbf{y} . The condition of positivity requiring $P(\mathbf{x}|\mathbf{y}) > 0, \forall \mathbf{x}$ has been assumed implicitly. Using the Hammersley-Clifford theorem [4] and assuming only up to pairwise clique potentials to be nonzero, the conditional distribution over all the labels \mathbf{x} given the observations \mathbf{y} in a DRF can be written as

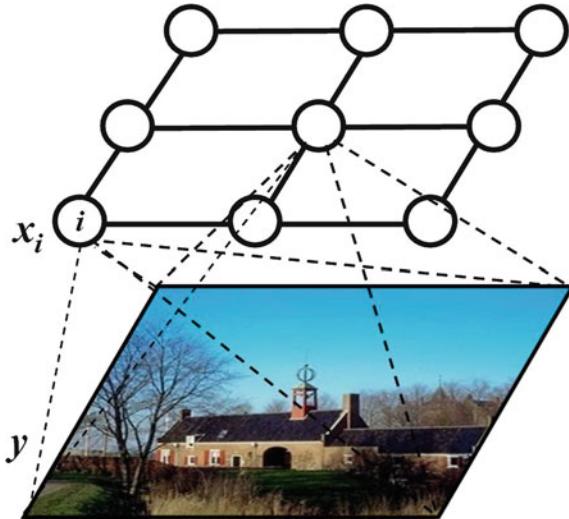
$$P(\mathbf{x}|\mathbf{y}) = \frac{1}{Z} \exp \left(\sum_{i \in S} A_i(x_i, \mathbf{y}) + \sum_{i \in S} \sum_{j \in \mathcal{N}_i} I_{ij}(x_i, x_j, \mathbf{y}) \right), \quad (1)$$

where Z is a normalizing constant known as the partition function, and $-A_i$ and $-I_{ij}$ are the unary and pairwise potentials, respectively. With a slight abuse of notation, refer A_i as the *association potential* and I_{ij} as the *interaction potential*.

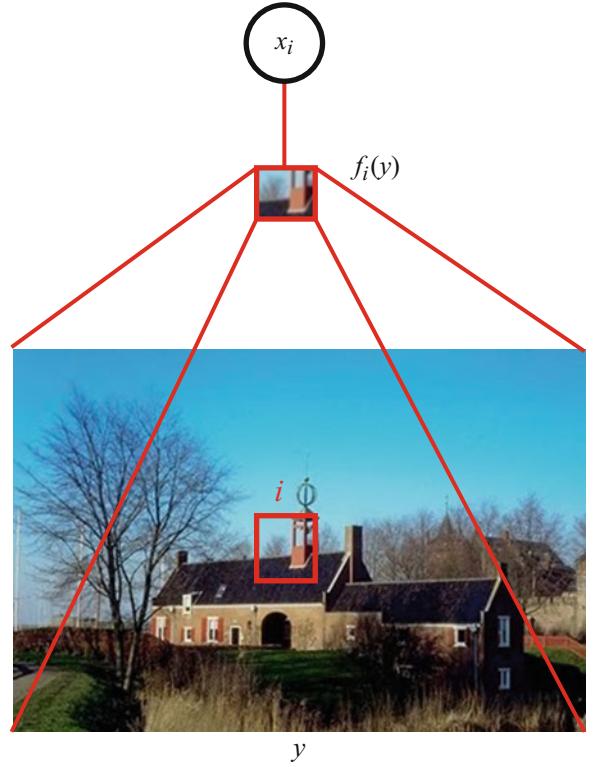
Assuming the random field given in Eq.(1) to be homogeneous, the functional forms of A_i and I_{ij} are independent of the location i . Furthermore, assuming the field to be isotropic implies that the label interactions are nondirectional. In other words, I_{ij} is independent of the relative locations of sites i and j . Thus, subsequently, one can drop the subscripts and simply use the notation A and I to denote the two potentials. In fact, the assumption of isotropy can be easily relaxed at the cost of a few additional parameters. Henceforth, consider the DRF model of the following form:

$$P(\mathbf{x}|\mathbf{y}) = \frac{1}{Z} \exp \left(\sum_{i \in S} A(x_i, \mathbf{y}) + \sum_{i \in S} \sum_{j \in \mathcal{N}_i} I(x_i, x_j, \mathbf{y}) \right). \quad (2)$$

Figure 1 illustrates a typical DRF for an example image analysis task of man-made structure detection. Given an input image \mathbf{y} shown in the bottom layer, suppose the goal is to label each image site (in this



Discriminative Random Fields, Fig. 1 An illustration of a typical DRF for an example task of man-made structure detection in natural images. The aim is to label each site, that is, each 16×16 image block whether it is a man-made structure or not. The top layer represents the labels x on all the image sites. Note that each site i can potentially use features from the whole image y unlike the traditional MRFs



Discriminative Random Fields, Fig. 2 Given a feature vector $f_i(y)$ at site i , the association potential in DRFs can be seen as a measure of how likely the site i will take label x_i , ignoring the effects of other sites in the image. Note that the feature vector $f_i(y)$ can be constructed by pooling arbitrarily complex dependencies in the observed data y

case a 16×16 image block) whether it contains a man-made structure or not. The top layer represents the labels x on all the image sites. Note that each site i can potentially use features from the whole image y unlike the traditional MRFs. In addition, DRFs allow to use image data to model interactions between two neighboring sites i and j . The following sections describe how the unary and the pairwise potentials are designed in DRFs.

Association Potential

In the DRF framework, the association potential, $A(x_i, y)$, can be seen as a measure of how likely a site i will take label x_i given image y , ignoring the effects of other sites in the image (Fig. 2). Suppose $f(\cdot)$ is a function that maps an arbitrary patch in an image to a feature vector such that $f : \mathcal{Y}_p \rightarrow \mathbb{R}^l$. Here, \mathcal{Y}_p is the set of all possible patches in all possible images. Let $\omega_i(y)$ be an arbitrary patch in the neighborhood of site i in image y from which a feature vector $f(\omega_i(y))$ is extracted. Note that the neighborhood used for the patch $\omega_i(y)$ need not be the same as the label neighborhood \mathcal{N}_i . Indeed, $\omega_i(y)$ can potentially be the whole image itself. For clarity, denote the feature vector $f(\omega_i(y))$ at each site i by $f_i(y)$.

The subscript i indicates the difference just in the feature vectors at different sites, *not* in the functional form of $f(\cdot)$. Then, $A(x_i, y)$ is modeled using a local discriminative model that outputs the association of the site i with class x_i as

$$A(x_i, y) = \log P'(x_i | f_i(y)), \quad (3)$$

where $P'(x_i | f_i(y))$ is the local class conditional at site i . This form allows one to use an arbitrary domain-specific probabilistic discriminative classifier for a given task. This can be seen as a parallel to the traditional MRF models where one can use arbitrary local generative classifier to model the unary potential. One possible choice of $P'(\cdot)$ is generalized linear models (GLM), which are used extensively in statistics to model the class posteriors. Logistic function is a commonly used link in GLMs, although other choices such

as probit link exist. Using a logistic function, the local class conditional can be written as

$$\begin{aligned} P'(x_i = 1 | \mathbf{f}_i(\mathbf{y})) &= \frac{1}{1 + e^{-(w_0 + \mathbf{w}_1^T \mathbf{f}_i(\mathbf{y}))}} \\ &= \sigma(w_0 + \mathbf{w}_1^T \mathbf{f}_i(\mathbf{y})), \end{aligned} \quad (4)$$

where $\mathbf{w} = \{w_0, \mathbf{w}_1\}$ are the model parameters. This form of $P'(\cdot)$ will yield a linear decision boundary in the feature space spanned by vectors $\mathbf{f}_i(\mathbf{y})$. To extend the logistic model to induce a nonlinear decision boundary, a transformed feature vector at each site i can be defined as $\mathbf{h}_i(\mathbf{y}) = [1, \phi_1(\mathbf{f}_i(\mathbf{y})), \dots, \phi_R(\mathbf{f}_i(\mathbf{y}))]^T$ where $\phi_k(\cdot)$ are arbitrary nonlinear functions. These functions can be seen as explicit kernel mapping of the original feature vector into a high dimensional space. The first element of the transformed vector is kept as 1 to accommodate the bias parameter w_0 . Further, since $x_i \in \{-1, 1\}$, the probability in Eq. (4) can be compactly expressed as

$$P'(x_i | \mathbf{y}) = \sigma(x_i \mathbf{w}^T \mathbf{h}_i(\mathbf{y})). \quad (5)$$

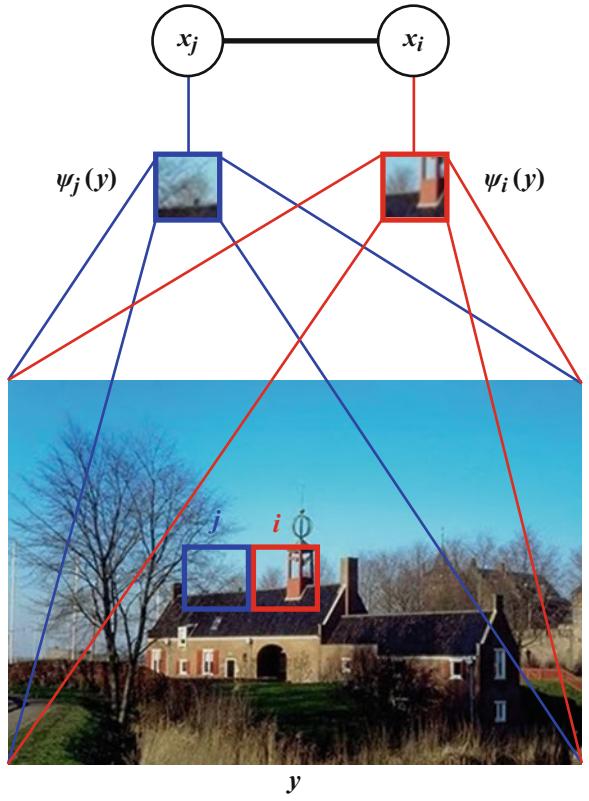
Finally, for this choice of $P'(\cdot)$, the association potential can be written as,

$$A(x_i, \mathbf{y}) = \log(\sigma(x_i \mathbf{w}^T \mathbf{h}_i(\mathbf{y}))) \quad (6)$$

This transformation ensures that the DRF is equivalent to a logistic classifier if the interaction potential in Eq. (2) is set to zero. Besides GLMs, discriminative classifiers based on SVM, neural network, and boosting have been successfully used in modeling association potential in the literature. Note that in Eq. (6), the transformed feature vector at *each* site i , that is, $\mathbf{h}_i(\mathbf{y})$ is a function of the whole set of observations \mathbf{y} . This allows one to pool arbitrarily complex dependencies in the observed data for the purpose of classification. On the contrary, the assumption of conditional independence of the data in the traditional MRF framework allows one to use data only from a particular site, that is, y_i , to design the log-density, which acts as the association potential.

Interaction Potential

In the DRF framework, the interaction potential can be seen as a measure of how the labels at neighboring sites i and j interact given the observed image \mathbf{y}



Discriminative Random Fields, Fig. 3 Given feature vectors $\psi_i(\mathbf{y})$ and $\psi_j(\mathbf{y})$ at two neighboring sites i and j , respectively, the interaction potential can be seen as a measure of how the labels at sites i and j influence each other. Note that such interaction in labels is dependent on the observed image data \mathbf{y} , unlike the traditional generative MRFs

(Fig. 3). In contrast to generative MRFs, the interaction potential in DRFs is a function of observations \mathbf{y} . Suppose $\psi(\cdot)$ is a function that maps an arbitrary patch in an image to a feature vector such that $\psi : \mathcal{Y}_p \rightarrow \mathfrak{R}^v$. Let $\Omega_i(\mathbf{y})$ be an arbitrary patch in the neighborhood of site i in image \mathbf{y} from which a feature vector $\psi(\Omega_i(\mathbf{y}))$ is extracted. Note that the neighborhood used for the patch $\Omega_i(\mathbf{y})$ need not be the same as the label neighborhood \mathcal{N}_i . For clarity, denote the feature vector $\psi(\Omega_i(\mathbf{y}))$ at site i by $\psi_i(\mathbf{y})$. Similarly, denote the feature vector at site j by $\psi_j(\mathbf{y})$. Again, to emphasize, the subscripts i and j indicate the difference just in the feature vectors at different sites, *not* in the functional form of $\psi(\cdot)$. Given the features at two different sites, the basic idea is to learn a pairwise discriminative model $P''(x_i = x_j | \psi_i(\mathbf{y}), \psi_j(\mathbf{y}))$. Note that by choosing the function ψ_i to be different from f_i , used in Eq. (4), information different from f_i

can be used to model the relations between pairs of sites.

For a pair of sites (i, j) , let $\mu_{ij}(\psi_i(y), \psi_j(y))$ be a new feature vector such that $\mu_{ij} : \Re^Y \times \Re^Y \rightarrow \Re^q$. Denoting this feature vector as $\mu_{ij}(y)$ for simplification, the interaction potential is modeled as

$$I(x_i, x_j, y) = x_i x_j \mathbf{v}^T \mu_{ij}(y), \quad (7)$$

where \mathbf{v} are the model parameters. Note that the first component of $\mu_{ij}(y)$ is fixed to be 1 to accommodate the bias parameter. There are two interesting properties of the interaction potential given in Eq. (7). First, if the association potential at each site and the interaction potentials for all the pairwise cliques except the pair (i, j) are set to zero in Eq. (2), the DRF acts as a logistic classifier which yields the probability of the site pair to have the same labels given the observed data. Of course, one can generalize the form in Eq. (7) as

$$I(x_i, x_j, y) = \log P''(x_i, x_j | \psi_i(y), \psi_j(y)), \quad (8)$$

similar to the association potential and can use arbitrary pairwise discriminative classifier to define this term. The second property of the interaction potential form given in Eq. (7) is that it generalizes the Ising model. The original Ising form is recovered if all the components of vector \mathbf{v} other than the bias parameter are set to zero in Eq. (7). A geometric interpretation of interaction potential is that it partitions the space induced by the relational features $\mu_{ij}(y)$ between the pairs that have the same labels and the ones that have different labels. Hence, Eq. (7) acts as a data-dependent discontinuity adaptive model that will moderate smoothing when the data from the two sites is “different.” The data-dependent smoothing can especially be useful to absorb the errors in modeling the association potential. Anisotropy can be easily included in the DRF model by parameterizing the interaction potentials of different directional pairwise cliques with different sets of parameters \mathbf{v} .

Parameter Learning and Inference

For 1D sequential CRFs proposed by Lafferty et al. [1], exact maximum likelihood parameter learning is feasible because the induced graph does not contain loops.

However, in typical DRFs, the underlying graph contains loops, and it is usually infeasible to exactly maximize the likelihood with respect to the parameters. Therefore, a critical issue in applying DRFs to image-based applications is the design of effective parameter learning techniques that can operate on arbitrary graphs.

Maximum Likelihood Parameter Learning

Let θ be the set of unknown DRF parameters where $\theta = \{\mathbf{w}, \mathbf{v}\}$. Given M i.i.d. labeled training images, the maximum likelihood estimates of the parameters are given by maximizing the log-likelihood $l(\theta) = \sum_{m=1}^M \log P(\mathbf{x}^m | \mathbf{y}^m, \theta)$, that is,

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{m=1}^M \left\{ \sum_{i \in S^m} \log \sigma(x_i^m \mathbf{w}^T \mathbf{h}_i(\mathbf{y}^m)) + \sum_{i \in S^m} \sum_{j \in \mathcal{N}_i} x_i^m x_j^m \mathbf{v}^T \mu_{ij}(\mathbf{y}^m) - \log Z^m \right\}, \quad (9)$$

where the partition function for the m th image is

$$Z^m = \sum_{\mathbf{x}} \exp \left\{ \sum_{i \in S^m} \log \sigma(x_i \mathbf{w}^T \mathbf{h}_i(\mathbf{y}^m)) + \sum_{i \in S^m} \sum_{j \in \mathcal{N}_i} x_i x_j \mathbf{v}^T \mu_{ij}(\mathbf{y}^m) \right\}.$$

Note that Z^m is a function of the parameters θ and the observed data \mathbf{y}^m . For learning the parameters using gradient ascent, the derivatives of the log-likelihood are

$$\frac{\partial l(\theta)}{\partial \mathbf{w}} = \frac{1}{2} \sum_m \sum_{i \in S^m} (x_i^m - \langle x_i \rangle_{\theta; \mathbf{y}^m}) \mathbf{h}_i(\mathbf{y}^m), \quad (10)$$

$$\frac{\partial l(\theta)}{\partial \mathbf{v}} = \sum_m \sum_{i \in S^m} \sum_{j \in \mathcal{N}_i} (x_i^m x_j^m - \langle x_i x_j \rangle_{\theta; \mathbf{y}^m}) \mu_{ij}(\mathbf{y}^m). \quad (11)$$

Here, $\langle \cdot \rangle_{\theta; \mathbf{y}^m}$ denotes expectation with $P(\mathbf{x} | \mathbf{y}^m, \theta)$. Ignoring $\mu_{ij}(\mathbf{y}^m)$, gradient ascent with Eq. (11) resembles the problem of learning in Boltzmann machines.

For arbitrary graphs with loops, the expectations in Eqs. (10) and (11) cannot be computed exactly due to the combinatorial size of the label space.

Sampling procedures such as Markov Chain Monte Carlo (MCMC) can be used to approximate the true expectations. Unfortunately, MCMC techniques have two main problems: a long “burn-in” period (which makes them slow) and high variance in estimates. Although several techniques have been suggested to approximate the expectations, two popular methods are described below (see [5] for other choices and a detailed comparison).

Pseudo-marginal Approximation (PMA)

It is easy to see if true marginal distributions $P_i(x_i|y, \theta)$ at each site, i , and $P_{ij}(x_i, x_j|y, \theta)$ at each pair of sites i and $j \in \mathcal{N}_i$ are known, one can compute exact expectations as

$$\begin{aligned}\langle x_i \rangle_{\theta; y} &= \sum_{x_i} x_i P_i(x_i|y, \theta) \text{ and } \langle x_i x_j \rangle_{\theta; y} \\ &= \sum_{x_i, x_j} x_i x_j P_{ij}(x_i, x_j|y, \theta).\end{aligned}$$

Since computing exact marginal distributions is in general infeasible, a standard approach is to replace the actual marginals by pseudo-marginals. For instance, one can use loopy belief propagation (BP) to get these pseudo-marginals. It has been shown in practice that for many applications, loopy BP provides good estimates of the marginals.

Saddle Point Approximation (SPA)

In saddle point approximation (SPA), one makes a discrete approximation of the expectations by directly using best estimates of labels at a given setting of parameters. This is equivalent to approximating the partition function (Z) such that the summation over all the label configurations \mathbf{x} in Z is replaced by the largest term in the sum, which occurs at the most probable label configuration. In other words, if

$$\hat{\mathbf{x}} = \arg \max_{\mathbf{x}} P(\mathbf{x}|y, \theta),$$

then according to SPA,

$$\begin{aligned}Z \approx \exp \left\{ \sum_{i \in S} \log \sigma(\hat{x}_i \mathbf{w}^T \mathbf{h}_i(y)) \right. \\ \left. + \sum_{i \in S} \sum_{j \in \mathcal{N}_i} \hat{x}_i \hat{x}_j \mathbf{v}^T \boldsymbol{\mu}_{ij}(y) \right\}.\end{aligned}$$

This leads to a very simple approximation to the expectation, that is, $\langle x_i \rangle_{\theta; y} \approx \hat{x}_i$. Further assuming a mean-field type decoupling, that is, $\langle x_i x_j \rangle_{\theta; y} = \langle x_i \rangle_{\theta; y} \langle x_j \rangle_{\theta; y}$, it also follows that $\langle x_i x_j \rangle_{\theta; y} \approx \hat{x}_i \hat{x}_j$. Readers familiar with perceptron learning rules can readily see that with such an approximation, the updates in Eq.(10) are very similar to perceptron updates.

However, this discrete approximation raises a critical question: Will the gradient ascent of the likelihood with such gradients converge? It has been shown empirically that while the approximate gradient ascent is not strictly convergent in general, it is weakly convergent in that it oscillates within a set of good parameters or converges to a good parameter with isolated large deviations. In fact, one can show that this weak-convergence behavior is tied to the empirical error of the model [5]. To pick a good parameter setting, one can use any of the popular heuristics used for perceptron learning with inseparable data. For instance, one can let the algorithm run up to some fixed number of iterations and pick the parameter setting that minimizes the empirical error. Even though lack of strict convergence can be seen as a drawback of SPA, the main advantage of these methods is very fast learning of parameters with performance similar to or better than pseudo-marginal methods.

Inference

Given a new test image y , the problem of inference is to find the optimal labels \mathbf{x} over the image sites, where optimality is defined with respect to a given cost function. Maximum a posteriori (MAP) solution is a widely used estimate that is optimal with respect to the zero-one cost function defined as

$$C(\mathbf{x}, \mathbf{x}^*) = 1 - \delta(\mathbf{x} - \mathbf{x}^*), \quad (12)$$

where \mathbf{x}^* is the true label configuration, and $\delta(\mathbf{x} - \mathbf{x}^*)$ is 1 if $\mathbf{x} = \mathbf{x}^*$, and 0 otherwise. The MAP solution is defined as

$$\hat{\mathbf{x}} = \arg \max_{\mathbf{x}} P(\mathbf{x}|y, \theta).$$

For binary classifications, the MAP estimate can be computed exactly for an undirected graph using the max-flow/min-cut type of algorithms if the probability

distribution meets certain conditions [6]. For the DRF model, since max-flow algorithms do not allow negative interaction between the sites, the data-dependent smoothing for each clique is set to be $v^T \mu_{ij}(y) = \max\{0, v^T \mu_{ij}(y)\}$, yielding an approximate MAP solution.

An alternative to the MAP solution is the maximum posterior marginal (MPM) solution which is optimal for the sitewise zero-one cost function defined as

$$C(\mathbf{x}, \mathbf{x}^*) = \sum_{i \in S} (1 - \delta(x_i - x_i^*)), \quad (13)$$

where x_i^* is the true label at the i th site. The MPM solution at each site is defined as

$$\begin{aligned} \hat{x}_i &= \arg \max_{\mathbf{x}_i} P_i(\mathbf{x}_i | \mathbf{y}, \theta), \text{ where } P_i(\mathbf{x}_i | \mathbf{y}, \theta) \\ &= \sum_{x - x_i} P(\mathbf{x} | \mathbf{y}, \theta), \end{aligned}$$

and $\mathbf{x} - x_i$ denotes all the node variables except for node i . The MPM computation requires marginalization over a large number of variables which is generally NP-hard. However, as discussed before, one can use loopy BP to obtain an estimate of the MPM solution.

Application

DRFs and related models have been applied to a wide variety of problems in computer vision to incorporate context in, for example, image denoising, scene segmentation and region classification, object recognition, and video tagging.

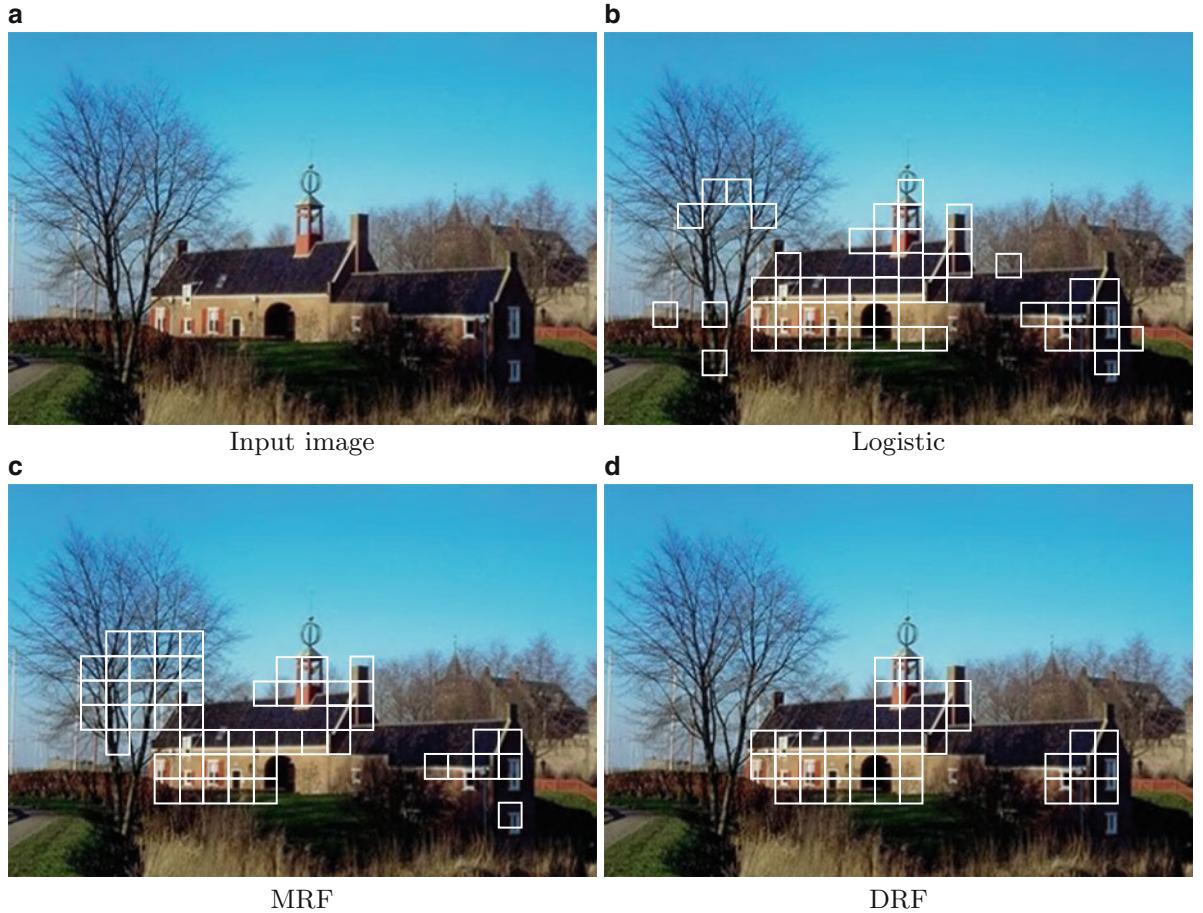
Experimental Results

Here is an application of binary DRFs to the problem of man-made structure detection in natural scenes. It is a difficult problem due to presence of significant clutter and wide variability in the appearance of man-made structure as well as the background. The training and the test set contained 108 and 129 images, respectively, from the Corel image database. Each image is divided into nonoverlapping 16×16

pixels blocks. Each block forms a site in the graph. The whole training set contained 36,269 blocks from the *nonstructured* class and 3,004 blocks from the *structured* class. Histograms over gradient orientations were used to extract heaved central-shift moment-based 14-dimensional feature vector. In the unary classifier, an explicit quadratic kernel is used to map the feature vector into a 119-dimensional space. The pairwise data vector is obtained by concatenating the unary vectors at two sites. The parameters of the DRF model were learned using the maximum likelihood framework as described before.

For an input test image given in Fig. 4a, the structure detection results from three methods are shown in Fig. 4. The blocks identified as *structured* have been shown enclosed within an artificial boundary. It can be noted that for similar detection rates, the number of false positives have significantly reduced for the DRF-based detection. Locally, different branches may yield features similar to those from the man-made structures. The logistic classifier does not enforce smoothness in labels, which led to increased isolated false positives. However, the MRF solution with Ising model simply smooths the labels without taking observations into account resulting in a smoothed false-positive region around the tree branches.

To carry out the quantitative evaluations, the detection rates and the number of false positives per image for each technique are compared. At first data, interactions are not allowed for any method by extracting features from individual sites. The comparative results for the three methods are given in Table 1 next to “MRF”, “Logistic” and “DRF”. For comparison purposes, the false-positive rate of the logistic classifier is fixed to be the same as the DRF in all the experiments. For similar false positives, the detection rates of the traditional MRF and the DRF are higher than the logistic classifier due to the label interaction. However, the higher detection rate of the DRF in comparison to the MRF indicates the gain due to the use of discriminative models in the association and interaction potentials. In the next experiment, to take advantage of the power of the DRF framework, data interaction was allowed for both the logistic classifier as well as the DRF (“Logistic” and “DRF” in Table 1). The DRF detection rate increases substantially, and the false positives decrease further indicating the importance of allowing the data interaction in addition to the label interaction.



Discriminative Random Fields, Fig. 4 Structure detection results on a test example for different methods. For similar detection rates, DRF reduces the false positives considerably. (a) Input image, (b) logistic, (c) MRF, and (d) DRF

Discriminative Random Fields, Table 1 Detection rates (DR) and false positives (FP) for the test set containing 129 images (49,536 sites). FP for logistic classifier were kept to be the same as for DRF for DR comparison. Superscript ‘ $-$ ’ indicates no neighborhood data interaction was used

| | MRF | Logistic $-$ | DRF $-$ | Logistic | DRF |
|----------------|-------|--------------|---------|----------|-------|
| DR (%) | 58.35 | 47.50 | 61.79 | 60.80 | 72.54 |
| FP (per image) | 2.44 | 2.28 | 2.28 | 1.76 | 1.76 |

Extensions and Related Work

A large number of extensions of the basic binary DRFs have been proposed. For instance, *multiclass* DRFs allow label sets consisting of more than two labels and *hierarchical DRFs* incorporate context at multiple levels [7]. For example, for parts-based object detection,

local context is the geometric relationship among parts of an object, while the relative spatial configurations of different objects (e.g., monitor, keyboard, and mouse) provides the global context. Learning in DRFs was extended to a semi-supervised paradigm by Lee et al. [8], while a Hidden-DRF model with latent variables was proposed in [9].

Open Problems

Hierarchical DRFs are of great interest since such models are quite powerful and can parse scenes at multiple levels of granularity. Robust parameter learning and inference in such models is a very challenging open problem.

References

1. Lafferty J, McCallum A, Pereira F (2001) Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the international conference on machine learning. Morgan Kaufmann Publishers, San Francisco
2. Kumar S, Hebert M (2003) Discriminative random fields: a discriminative framework for contextual interaction in classification. In: Proceedings of the IEEE international conference on computer vision (ICCV), Nice, vol 2, pp 1150–1157
3. Kumar S, Hebert M (2004) Discriminative fields for modeling spatial dependencies in natural images. In: Thrun S, Saul L, Schölkopf B (eds) Advances in neural information processing systems 16, The MIT Press, Boston
4. Hammersley JM, Clifford P Markov field on finite graph and lattices. Unpublished
5. Kumar S, August J, Hebert M (2005) Exploiting inference for approximate parameter learning in discriminative fields: an empirical study. In: Fourth international workshop on energy minimization methods in computer vision and pattern recognition (EMMCVPR), St. Augustine
6. Greig DM, Porteous BT, Seheult AH (1989) Exact maximum a posteriori estimation for binary images. J R Stat Soc 51(2):271–279
7. Kumar S, Hebert M (2005) A hierarchical field framework for unified context-based classification. In: IEEE international conference on computer vision (ICCV), Beijing
8. Lee C, Wang S, Jiao F, Schuurmans D, Greiner R (2006) Learning to model spatial dependency: semi-supervised discriminative random fields. In: Neural information processing systems conference (NIPS), Vancouver
9. Quattoni A, Collins M, Darrell T (2004) Conditional random fields for object recognition. In: Neural information processing systems conference (NIPS), Vancouver

Disocclusion

► Inpainting

Distance Estimation

► Depth Estimation

Dual Differential Geometry

► Isotropic Differential Geometry in Graph Spaces

Dynamic Optimization

► Dynamic Programming

Dynamic Programming

Boris Flach and Vaclav Hlavac

Department of Cybernetics, Czech Technical University in Prague, Faculty of Electrical Engineering, Prague 6, Czech Republic

Synonyms

Dynamic optimization

Definition

Dynamic Programming (DP) is a paradigm used in algorithms for solving optimization problems. It relies on the construction of nested subproblems such that the solution of the main problem can be obtained from the solutions of the subproblems.

Background

The paradigm was introduced by the mathematician Richard Bellman in the 1940s and applied in control theory [1].

Theory

The applicability of the paradigm relies on the following two assumptions:

Optimal substructure means that a system of nested subproblems can be constructed in such a way that the solution of the main problem can be obtained from the solutions of these subproblems.

Overlapping subproblems means that the smaller subproblems in the next level are only slightly smaller and moreover the set of these subproblems is small as well. This distinguishes DP from “divide and conquer” methods.

The method starts by solving the smallest subproblems directly. The obtained results are stored and used for solving the bigger subproblems in the next higher level. Applying this principle for each step avoids expensive recursions.

The following two algorithms illustrate applications of the DP paradigm.

Viterbi Algorithm

The task is to find the maximal value of a function F which depends on n discrete variables $x_1, x_2, \dots, x_n \in K$ in a finite domain K . The function has the form

$$\begin{aligned} F(x_1, \dots, x_n) = & f_1(x_1, x_2) + f_2(x_2, x_3) + \dots \\ & + f_{n-1}(x_{n-1}, x_n). \end{aligned} \quad (1)$$

An appropriate nested system of subproblems is defined by

$$\phi_i(k) = \max_{x_1, \dots, x_i} [f_1(x_1, x_2) + \dots + f_i(x_i, k)] \quad (2)$$

for all $i = 1, \dots, n - 1$ and all $k \in K$. It is quite obvious that the solutions for $\phi_{i+1}()$ can be obtained from those for $\phi_i()$ by

$$\phi_{i+1}(k) = \max_{k' \in K} [\phi_i(k') + f_{i+1}(k', k)]. \quad (3)$$

The solution of the simplest subproblems $\phi_1(k)$ is found simply by enumeration and similarly, the solution of the main problem is eventually obtained by solving $\max_k \phi_{n-1}(k)$.

Additional memory space is required if the task is to calculate a maximizer (x_1^*, \dots, x_n^*) of F rather than its maximal value. In such a case a “pointer”

$$p_{i+1}(k) = \operatorname{argmax}_{k' \in K} [\phi_i(k') + f_{i+1}(k', k)] \quad (4)$$

has to be stored for each $i = 1, \dots, n - 1$ and each $k \in K$. The required maximizer is found simply by backtracking in this list, that is, $x_i^* = p_{i+1}(x_{i+1}^*)$.

Floyd Warshall Algorithm

The task is to calculate the shortest paths for all pairs of vertices in a weighted graph (V, E, w) . It is assumed for simplicity that all nonexistent edges are included but have infinite length. Let $S(i, j)$, $i, j \in V$, denote the length of the shortest path connecting vertex i with vertex j . The nested system of subproblems is defined as follows: Let $S(i, j, k)$ denote the length of the shortest path connecting vertex i with vertex j which is passing only through vertices of the subset $V_k = \{1, 2, \dots, k\} \subset V$. In other words, $S(i, j, k)$ is the length of the shortest path connecting i and j in the subgraph induced by the vertex set $\{i, j\} \cup V_k$. Again, it is easy to see that $S(i, j, k+1)$ can be calculated from $S(., ., k)$. The sought-after shortest path either passes through vertices from V_k only or otherwise, is composed from a part connecting vertices i and $k+1$ and a second part connecting vertices $k+1$ and j . This leads to

$$\begin{aligned} S(i, j, k+1) = & \min \{S(i, j, k), S(i, k+1, k) \\ & + S(k+1, j, k)\}. \end{aligned} \quad (5)$$

The solutions of the smallest subproblems are obviously $S(i, j, 0) = w(i, j)$.

The algorithm has time complexity $\mathcal{O}(|V|^3)$ and space complexity $\mathcal{O}(|V|^2)$ and provides the lengths of the shortest paths between all pairs of vertices. A simple modification of the algorithm allows one to reconstruct the shortest path for each pair of vertices. The key idea is to store an additional matrix of size $|V| \times |V|$, whose elements represent the largest of the indices of intermediate vertices in the shortest path connecting vertices i and j . This matrix is initialized and updated along with matrix S . Once calculated, it allows one to reconstruct the shortest path between each pair of vertices recursively.

A more comprehensive analysis of DP and other examples can be found in [2, 3].

References

1. Bellman R (2003) Dynamic programming. Dover, Mineola
2. Denardo EV (2003) Dynamic programming: models and applications. Dover, Mineola
3. Sniedovich M (2010) Dynamic programming: foundations and principles. Taylor and Francis, Boca Raton

E

Edge Detection

James H. Elde

Centre for Vision Research, York University, Toronto,
ON, Canada

Related Concepts

► [Boundary Detection](#); ► [Scale Selection](#)

Definition

Edge detection is the process of labeling the image pixels that lie on the boundaries where abrupt intensity discontinuity occurs.

Background

The light projected from a visual scene into an eye or camera is typically piecewise smooth as a function of visual angle. Since nearby points on a surface tend to have similar attitude, reflectance, and illumination, the pixels to which these surface points project tend to have similar intensity. This rule is broken when two adjacent pixels project from points on either side of an occlusion boundary, since the points now project from different surfaces that may well have different attitude, reflectance, and illumination, and typically an abrupt change in image intensity results. Intensity edges also arise when neighboring pixels project from points on the same surface that happens to straddle a surface crease, pigment change, or shadow boundary.

Since these abrupt changes in image intensity correspond to significant physical events in the scene, the problem of reliably detecting and localizing these edges is an important and fundamental early vision problem. While detection of object boundaries (occlusion edges) is sometimes seen as the main goal, reliable detection of surface creases and reflectance edges also has clear importance for shape estimation and object recognition, and even cast shadows provide important information about relief, surface contact, and scene layout.

Since edges are sparse, edge detection is also motivated from a differential image coding viewpoint: a large fraction of the information in an image can be captured by coding just the locations, 2D orientations, and intensity changes of these edges.

The problem of edge detection dates back to the first days of computer vision: back to Roberts' thesis at least [20]. In its simplest form, the goal is to label the image pixels that lie on (or very near) a step discontinuity in the image. This definition has been generalized in a number of useful ways: to allow for possible blurring of the step discontinuity, due to shading or defocus, for example, and to require an estimate of the local 2D orientation of the edge as well as its location.

- Biological basis [9]
- History [20]
- Problem definition [1, 2, 5, 6, 13, 15, 18]

Theory and Application

Roberts based his edge detector on a simple 2 pixel \times 2 pixel discrete differential operator related to the gradient magnitude (Fig. 1). Since the operator relies

Edge Detection, Fig. 1

Edge detection example [4–6]. Top left: Original greyscale image. Top right: Edge map. Bottom left: Reconstruction of original image from brightness and contrast stored only at edge locations. Bottom right: Reconstruction including edge blur information



upon only 4 pixels, the results are highly sensitive to noise and are dominated by the fine structure of the image. Over the intervening years, most edge detection algorithms have continued to rely on a first stage of local differential filtering but have innovated in the design of smoothing kernels to increase signal-to-noise ratio, in the order of the differential operators used, and in how they are combined. This local differential filtering approach aligns well with physiological data showing that neurons early in the primate visual pathway can to some degree be approximated as stabilized low-order differential operators [9].

By far the most popular smoothing kernel has been the 2D Gaussian $G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$. Marr and Hildreth [17] proposed the use of second-order isotropic Laplacian-of-Gaussian (“Mexican hat”) filters $\nabla^2 G(x, y)$. The $\nabla^2 G$ filter produces a signed response that crosses zero precisely at the location of an (ideal) step edge, and Marr and Hildreth proposed that edges thus be identified with such zero crossings. Due to the isotropy of the $\nabla^2 G$ filter, the response is invariant to the orientation of the edge, a desirable property, since edges can occur at any orientation. Marr and Hildreth also observed that different edges occur at different scales, and so employed $\nabla^2 G$ filters over a range of scales σ . Observing that spurious zero crossings at a single scale can occur due to interference between multiple distinct edges, Marr and Hildreth proposed a *scale combination rule*: edges are deemed valid only if zero crossings at the same location and orientation are found at more than one scale.

Marr and Hildreth obtained orientation invariance by using isotropic filters and approximate scale invariance by combining information across scales. However, these invariance properties came at the price detection and localization performance. This became clear partly through the work of John Canny [2], who used a variational approach to determine an edge detector that would be (nearly) optimal precisely in terms of detection rate and localization performance, while avoiding multiple responses to the same edge. The outcome was a an edge detection filter that is well approximated by a first-derivative-of-Gaussian function. Canny used two such filters to estimate stabilized partial derivatives and hence the local gradient vector ∇G :

$$\nabla G = \begin{pmatrix} G_x \\ G_y \end{pmatrix}, \text{ where } G_x = \frac{\partial G}{\partial x} \text{ and } G_y = \frac{\partial G}{\partial y}$$

The first-derivative-of-Gaussian filter can be shown to be steerable [7]: the filter can be synthesized at any orientation from a linear combination of the two basis filters G_x and G_y . Thus, Canny’s approach amounts to computing the first derivative in the gradient direction at every pixel of the smoothed image. In this sense, the approach still enjoys the orientation invariance property emphasized by Marr and Hildreth: a rotation of the edge in the image will not change the estimated gradient magnitude. Canny’s approach, however, delivers higher signal-to-noise because the filter more closely approximates the shape of an extended edge in the tangent direction. While Marr and Hildreth’s $\nabla^2 G$

operator computes the second derivative along the edge, Canny's $\nabla^2 G$ operator locally integrates along the edge over the support of the Gaussian kernel.

While Marr and Hildreth localized edges at the zero crossings of the $\nabla^2 G$ response, Canny localized edges at maxima of the gradient magnitude, taken in the gradient direction, found using a process called *non-maximum suppression*. Since not all of the resulting maxima correspond to significant edges, a threshold on the gradient magnitude must be applied to reduce the false positive rate. Canny devised a clever heuristic technique, dubbed *thresholding with hysteresis*, that removes many false positives without unduly affecting the hit rate (proportion of correct detections). The technique exploits the fact that real weak edges are often chain-connected along a contour to stronger edges, while false positives are more likely to be isolated. The technique employs two thresholds: all edges that are above the low threshold and chain-connected to edges above the high threshold are identified as edges.

The Canny edge detector may well be the most widely used algorithm in the history of computer vision. Its early adoption is likely derived in part from the open availability of the source code, but its continuing widespread use reflects the fact that it continues to perform well in comparison to more recent algorithms.

While most versions of Canny's algorithm in use detect edges only at a single scale, in fact both Marr and Hildreth and Canny recognized the problem of scale, but dealt with it in slightly different ways. While Marr and Hildreth conjunctively combined responses across scale (zero crossings of the $\nabla^2 G$ response must be found at multiple scales at the same location and orientation to signal an edge), Canny proposed to *disjunctively* combine edges detected at different scales. This raises the *multiple response problem*: how do we know whether two extremal responses, at different scales but similar locations and orientations, signal two distinct edges in the image, or a single edge corrupted by noise? Canny proposed a *feature synthesis* method to deal with this problem, that signals the larger scale edge only if it could not be satisfactorily explained by the smaller scale responses in a local neighborhood.

Canny's proposed technique localizes edges using the smallest scale at which they are detected, an approach that Canny justified based upon his variational optimization, and this motivated the later development of *edge focusing* techniques [1] in which edges are detected at coarse scales and then tracked

through scale space to finer scales for better localization. As it turns out, Canny's one-dimensional analysis does not generalize to two dimensions: Elder and Zucker [5, 6] later showed that for an ideal step or blurred step edge corrupted by noise, localization improves monotonically with increasing scale. This result highlights the difference between theory and practice: since edges in real images are always finite in extent, often curved, and surrounded by other kinds of image structure, neither detection nor localization is likely to be optimized by maximizing scale.

Elder and Zucker pointed out that while it is difficult to model all of the real-world effects that limit the effectiveness of large filters, the small-scale problem is tractable, since performance is limited by noise that can be modeled as white, and for which parameters can be estimated. Based on this observation, they proposed a method for selecting the scale of differential operators called *local scale control*, in which scales are search from fine to coarse to determine the *minimum reliable scale*, that is the smallest scale at which the sign of the derivative measurement is statistically reliable. This approach avoids spurious responses to noise, while minimizing bias due to the finite extent of the edge, curvature, and interference from neighbouring image structure. They also demonstrated methods for subpixel localization down to roughly 1/20 of a pixel precision [5].

A different scale selection approach for edge detection was proposed at roughly the same time by Lindeberg [15]. Lindeberg's method selects the scale maximizing the response of so-called γ -normalized differential operators. This approach has the property that the scales selected for an ideal, noise-free, isolated, blurred straight edge of infinite extent will be proportional to the blur scale of the edge, and independent of the edge contrast. Thus, small scales should be selected for sharp edges, and larger scales for blurred edges. Unlike the local scale control approach of Elder and Zucker, the selection of scale is not related to the signal-to-noise ratio: in general a large-scale filter will have a smaller response to an ideal edge than a small-scale filter, even when the signal-to-noise for the two filters is the same. As a result of this bias to smaller scales, many noisy edges are detected, and these must be eliminated using some form of post hoc thresholding.

These studies also raised issues around the order of differential operators to employ. While Canny

localized edges at gradient maxima, Elder and Zucker localized edges at zero crossings of the second derivative, steered in the gradient direction. Specifically edges are localized at the boundary between two pixels where the sign of the second derivative changes from positive to negative in the gradient direction (this avoids detections of minima in the gradient magnitude). Where the same scales used for these two operators, the results would be identical. However, the Elder and Zucker approach tunes the scales of the two operators independently, based upon the signal-to-noise properties of the operators, and as a result the gradient maxima and second derivative zero crossings are decoupled. While the statistical testing method cannot be used to distinguish real and spurious maxima, it can be used to test for response sign, and hence to detect zero crossings. Thus, the use of the second derivative here is critical. Lindeberg [15] has argued for explicit calculation and testing of the sign of the third derivative, but this is equivalent to checking the sign of the second derivative on either side of the zero crossing. However, Elder and Zucker do show that explicit computation of the third derivative is useful for estimating the blur of an edge, which may be useful for discriminating different types of edges, and for recovering depth from defocus.

In their 1980s paper [17], Marr and Hildreth speculated about the possibility that the locations and gradient magnitudes at oriented zero crossings over multiple scales might constitute a complete encoding of the image, allowing, in principle, for the image to be perfectly reconstructed from the edge representation. While this may seem to be going in the wrong direction from a computer vision point of view, the question is important because it addresses whether an edge code could serve as a *complete* early visual representation, providing sufficient information for all higher-level algorithms.

Since Marr and Hildreth's original conjecture, there have been numerous theoretical and empirical studies exploring the possible completeness of an edge code. Mallat and Zhong [16] demonstrated excellent reconstruction results based upon such a code. However, since edges are represented at many scales, and at the finest scales edge density is very high, this code is highly overcomplete. More compact codes can be derived, but at the expense of noticeable artifact in the reconstruction.

Elder [4] explored an alternative reconstruction approach based upon their edge representation.

Rather than storing information from all scales at each edge point, only the location, 2-bit orientation, contrast, brightness, and blur of each edge point were stored, resulting in a far more compact code. Reconstruction is excellent as long as both the intensity and the blur information are employed in the reconstruction.

Since edge detection is often just the first computational step in a computer vision pipeline, and applications may have real-time requirements, efficiency has always been a factor in the design of edge detection algorithms. Both Marr and Hildreth [17] and Canny [2] deliberately employed filters that were $x - y$ separable, allowing 2D convolutions to be computed by composing much cheaper 1D convolutions. Rachid Deriche [3] further improved upon these efficiency by developing recursive filters based upon Canny's original design criteria, allowing very fast implementation on sequential hardware. More recent algorithms have tended to rely upon steerable filters [7] that allow oriented operators to be implemented using only a few basis functions.

While linear filtering forms the front end of most edge detection algorithms, a number of interesting non-linear techniques have been studied. In fact this has a long tradition, going back to early work that sees the problem as model fitting [10] and includes active contour ("snake") methods for fitting semi-local deformable contour models to visual data [12]. While effective in many applications, active contour methods tend to have more parameters to tune, and results are sensitive to these and to initial conditions.

Still to include:

- Scale [13]
- Color edge detection [14]
- Nonlinear filtering [11]
- Nonlocal methods [19]
- Evaluation [8, 13, 18]
- Redefinition in terms of "salient" boundaries [13, 18]

References

1. Bergholm F (1987) Edge focusing. *IEEE Trans Pattern Anal Mach Intell* 9(6):726–741
2. Canny J (1986) A computational approach to edge-detection. *IEEE Trans Pattern Anal Mach Intell* 8(6): 679–698
3. Deriche R (1987) Using Canny's criteria to derive a recursively implemented optimal edge detector. *Int J Comput Vis* 1(2):167–187



4. Elder JH (1999) Are edges incomplete? *Int J Comput Vis* 34(2–3):97–122
5. Elder JH, Zucker SW (1996) Scale space localization, blur and contour-based image coding. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos, pp 27–34
6. Elder JH, Zucker SW (1998) Local scale control for edge detection and blur estimation. *IEEE Trans Pattern Anal Mach Intell* 20(7):699–716
7. Freeman WT, Adelson EH (1991) The design and use of steerable filters. *IEEE Trans Pattern Anal Mach Intell* 13(9):891–906
8. Heath M, Sarkar S, Sanocki T, Bowyer K (1998) Comparison of edge detectors – a methodology and initial study. *Comput Vis Image Underst* 69(1):38–54
9. Hubel DH, Wiesel TN (1968) Receptive fields and functional architecture of monkey striate cortex. *J Physiol* 195:215–243
10. Hueckel MH (1971) An operator which locates edges in digitized pictures. *J Assoc Comput Mach* 18:113–125
11. Iverson LA, Zucker SW (1995) Logical/linear operators for image curves. *IEEE Trans Pattern Anal Mach Intell* 17(10):982–996
12. Kass M, Witkin A, Terzopoulos D (1987) Snakes – active contour models. *Int J Comput Vis* 1(4):321–331
13. Konishi S, Yuille AL, Coughlan JM, Zhu SC (2003) Statistical edge detection: learning and evaluating edge cues. *IEEE Trans Pattern Anal Mach Intell* 25(1):57–74
14. Lee HC, Cok DR (1991) Detecting boundaries in a vector field. *IEEE Trans Signal Process* 39(5):1181–1194
15. Lindeberg T (1998) Edge detection and ridge detection with automatic scale selection. *Int J Comput Vis* 30(2):117–154
16. Mallat S, Zhong S (1992) Characterization of signals from multiscale edges. *IEEE Trans Pattern Anal Mach Intell* 14(7):710–732
17. Marr D, Hildreth E (1980) Theory of edge-detection. *Proc R Soc Lond B* 207(1167):187–217
18. Martin DR, Fowlkes CC, Malik J (2004) Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans Pattern Anal Mach Intell* 26(5):530–549
19. Perona P, Malik J (1990) Scale-space and edge-detection using anisotropic diffusion. *IEEE Trans Pattern Anal Mach Intell* 12(7):629–639
20. Roberts L (1965) Machine perception of 3-dimensional solids. In: Tippett J (ed) Optical and electro-optical information processing. MIT, Cambridge, MA

Eigenspace Methods

Tomokazu Takahashi and Hiroshi Murase
Faculty of Economics and Information, Gifu Shotoku Gakuen University, Gifu, Japan

Synonyms

Methods of image recognition in a low-dimensional eigenspace

Related Concepts

► [Principal Component Analysis \(PCA\)](#)

Definition

The eigenspace method is an image recognition technique that achieves object recognition, object detection, and parameter estimation from images using the distances between input and gallery images in a low-dimensional eigenspace. Here, the eigenspace is constructed based on a statistical method, such as principal component analysis or Karhunen-Loëve transform, so that the variation in the appearances of target objects can be represented in a low-dimensional space efficiently. In particular, a technique called the parametric eigenspace method represents the rotation and translation of a target object or a light source as a manifold in an eigenspace. Accordingly, this method performs object recognition and parameter estimation using distances in the manifold.

Background

Appearance-based object recognition is a technique that recognizes a target object by matching between input and preregistered gallery images. One of the simplest methods to achieve this image matching calculates the distances between the pixel values of these images. However, it is difficult for many applications to apply this method due to two problems: (1) processing time needed to calculate the distance between images increases depending on the size of the images and (2) memory space needed to store the gallery images of target objects grows depending on the number of the objects.

Principal component analysis (PCA) can be used as one of the unsupervised dimensionality reduction techniques that transform a sample set of high-dimensional vectors to the set of low-dimensional vectors with minimum information loss. This technique first calculates eigenvectors that are at right angles to each other and maximizes the variances in their directions; it then constructs a low-dimensional eigenspace defined by a small number of eigenvectors. The low-dimensional vectors are obtained by projecting the high-dimensional vectors to the eigenspace. An eigenspace method constructs a low-dimensional



Eigenspace Methods, Fig. 1 Image samples in an object image set containing object images with different horizontal poses

eigenspace from preregistered gallery images of target objects and calculates the distances between input and gallery images in the low-dimensional space. Thus, the eigenspace method can reduce the processing time and memory space efficiently without degrading the recognition performance.

Theory

An image recognition procedure using an eigenspace method is divided into learning and recognition stages. The learning stage, which is performed beforehand, constructs a low-dimensional eigenspace from a large number of learning images of target objects and then projects each learning image to the eigenspace. On the other hand, the recognition stage projects an input image of a target object to the eigenspace and then recognizes the input object by matching between input and learning images in the low-dimensional space.

Learning Stage

For each learning image, the method first normalizes the image size so that the number of pixels can be N , and it represents them as

N -dimensional vectors $\mathbf{x}_1, \dots, \mathbf{x}_M$. M represents the number of learning images. Here, we assume that each target object has one learning image; therefore, M also represents the number of target objects. A matrix \mathbf{X} is constructed as follows:

$$\mathbf{X} = \left[\frac{\mathbf{x}_1 - \mathbf{c}}{\|\mathbf{x}_1 - \mathbf{c}\|} \cdots \frac{\mathbf{x}_M - \mathbf{c}}{\|\mathbf{x}_M - \mathbf{c}\|} \right], \quad (1)$$

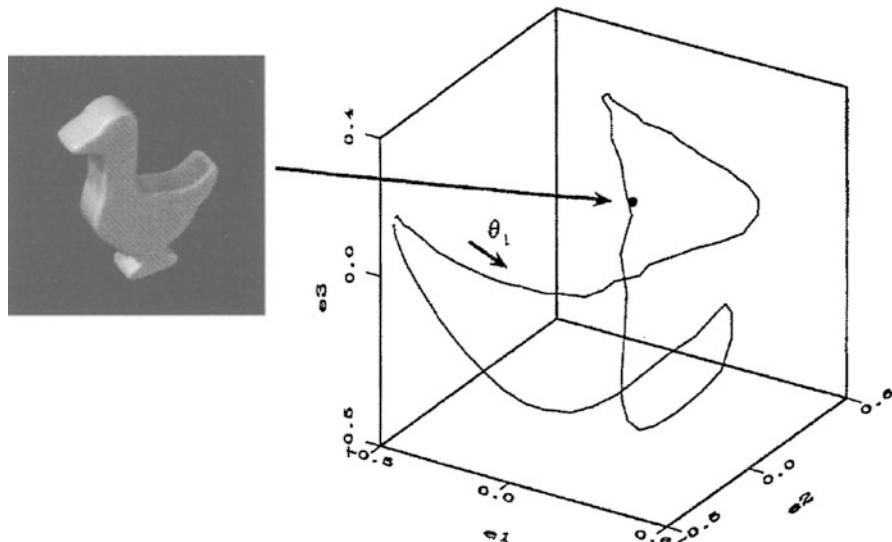
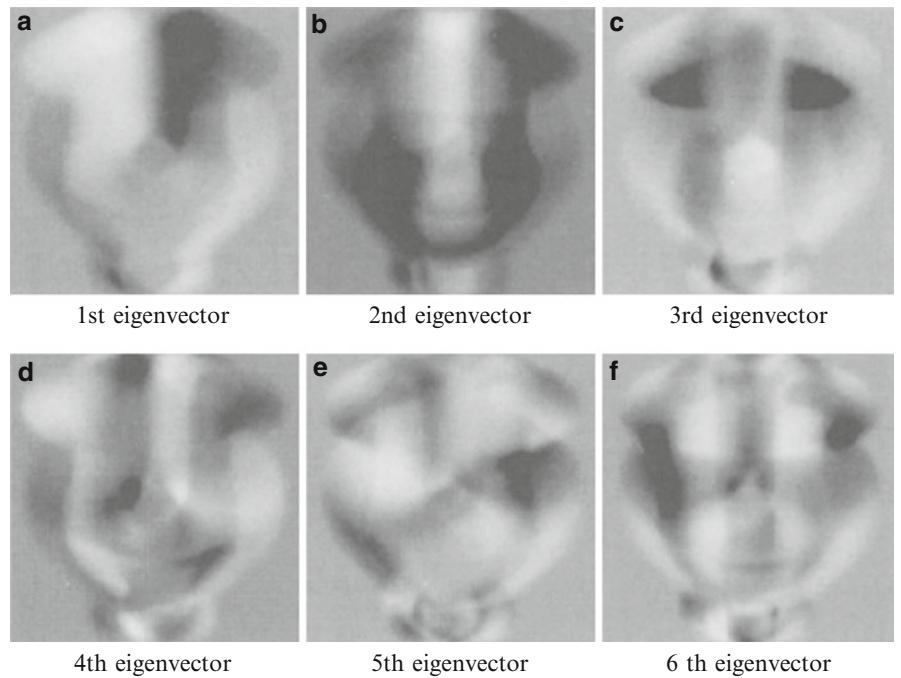
$$\mathbf{c} = \frac{1}{M} \sum_{m=1}^M \mathbf{x}_m. \quad (2)$$

Eigenvectors are obtained by solving the following eigenequation:

$$\mathbf{X} \mathbf{X}^T \mathbf{u}_i = \lambda_i \mathbf{u}_i, \quad (3)$$

where \mathbf{u}_i represents an eigenvector of $\mathbf{X} \mathbf{X}^T$ corresponding to an eigenvalue λ_i . Eigenvectors $\mathbf{u}_1, \dots, \mathbf{u}_N$ are arranged in descending order of their eigenvalues. The method constructs a low-dimensional eigenspace from $k (< N)$ eigenvectors corresponding to the k largest eigenvalues and then obtains k -dimensional vectors \mathbf{f}_m by projecting N -dimensional vectors \mathbf{x}_m to the eigenspace using the following equation:

Eigenspace Methods, Fig. 2
Eigenvectors calculated from the object image set shown in Fig. 1. (a)–(f) represent the first through sixth eigenvectors, respectively



Eigenspace Methods, Fig. 3 A manifold in the object eigenspace constructed from the eigenvectors shown in Fig. 2. Transition of object appearances according to the horizontal pose parameter θ_1 draws a closed smooth curve in the eigenspace

$$\mathbf{f}_m = [\mathbf{u}_1 \cdots \mathbf{u}_k]^T \frac{\mathbf{x}_m - \mathbf{c}}{\|\mathbf{x}_m - \mathbf{c}\|}. \quad (4)$$

Consequently, the memory space needed to store the learning images can be greatly reduced.

Recognition Stage

In the same manner as in the learning stage, the method first normalizes the image size of an input image and represents it as an N -dimensional vector \mathbf{y} . The method then obtains a k -dimensional vector \mathbf{g} by projecting \mathbf{y} to the low-dimensional eigenspace

constructed in the learning stage using the following equation:

$$\mathbf{g} = [\mathbf{u}_1 \cdots \mathbf{u}_k]^T \frac{\mathbf{y} - \mathbf{c}}{\|\mathbf{y} - \mathbf{c}\|}. \quad (5)$$

The method recognizes the input object as an object \hat{m} that minimizes the distance between the input image \mathbf{g} and the learning image f_m using the following equation:

$$\hat{m} = \arg \min_m \|\mathbf{g} - f_m\|^2. \quad (6)$$

Accordingly, the processing time needed to calculate the distance between the images can be greatly reduced.

Based on an efficient representation of human face images using PCA as proposed by Sirovich and Kirby [1], Turk and Pentland [2] proposed a method using “eigenfaces” to detect and recognize human faces in images. This method calculates a small number of eigenvectors from a large number of learning face images and measures the distances between input and learning face images in a low-dimensional eigenspace. The eigenvectors calculated from the face images are called eigenfaces. Since this method was reported, image recognition using the eigenspace method has become an active area of research.

Murase and Nayar [3] proposed a “parametric eigenspace method” for recognizing three-dimensional objects and estimating their poses simultaneously. This method constructs a manifold as a smooth curved line approximated from a point set in an eigenspace for each target object. The point set is obtained by projecting learning images with various poses to the eigenspace. Object recognition and pose estimation are achieved by calculating the distances between an input image and manifolds. In order to achieve object recognition and pose estimation simultaneously, two types of eigenspaces are constructed: a universal eigenspace from the learning images of all target objects and an object eigenspace for each target object. The method first recognizes an object in the universal eigenspace and then estimates a pose of the object in the object eigenspace. [Figure 1](#) shows samples in an object image set containing object images with different horizontal poses, and [Fig. 2](#) shows eigenvectors that were calculated from the image set. On the other hand, the

closed smooth curve shown in [Fig. 3](#) represents a manifold in the object eigenspace that was constructed from the eigenvectors. The manifold was approximated from the points that were obtained by projecting the image in the image set shown in [Fig. 1](#) to the eigenspace.

Ohba and Ikeuchi [4] proposed a method using “eigen window” to recognize partially occluded objects accurately and estimate their poses. The method extracts multiple local regions with high detectability, uniqueness, and reliability from an object region in each learning image. Eigen windows are constructed from the extracted local regions by PCA. Object recognition and pose estimation are achieved by matching between the eigen windows and local regions extracted from an input image in a similar way.

In addition to the approaches described above, there have been a number of techniques related to the eigenspace method. These include techniques of density estimation of samples in a high-dimensional space [5], nonlinear expansion of PCA [6, 7], image recognition using two-dimensional PCA [8], and image recognition using high-order tensors [9].

Application

The eigenspace method is used as a fundamental technique in arbitrary computer vision applications, such as object recognition, detection, and tracking, because the method works effectively despite its algorithm being quite simple.

References

1. Sirovich L, Kirby M (1987) Low-dimensional procedure for the characterization of human faces. *J Opt Soc Am A* 4(3):519–524
2. Turk M, Pentland A (1991) Eigenfaces for recognition. *J Cognit Neurosci* 3(1):71–86
3. Murase H, Nayar S (1995) Visual learning and recognition of 3-d objects from appearance. *Int J Comput Vis* 14:5–24
4. Ohba K, Ikeuchi K (1997) Detectability, uniqueness, and reliability of eigen window for stable verification of partially occluded objects. *IEEE Trans Pattern Anal Mach Intell* 19(9):1043–1048
5. Moghaddam B, Pentland A (1997) Probabilistic visual learning for object representation. *IEEE Trans Pattern Anal Mach Intell* 19(7):696–710
6. Schölkopf B, Smola A, Müller K (2006) Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput* 10(5):1299–1319



7. Belkin M, Niyogi P (2006) Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput* 15(6):1373–1396
8. Yang J, Zhang D, Frangi A, Yang J (2004) Two-dimensional pca: a new approach to appearance-based face representation and recognition. *IEEE Trans Pattern Anal Mach Intell* 26(1):131–137
9. Vasilescu M, Terzopoulos D (2002) Multilinear analysis of image ensembles: tensorfaces. *Proc Eur Conf Comput Vis (ECCV)* 1:447–460

Eight-Point Algorithm

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Related Concepts

- [Eight-Point Algorithm](#); ► [Epipolar Geometry](#)
- [Essential Matrix](#); ► [Fundamental Matrix](#)

Definition

The 8-point algorithm is a linear technique to estimate the essential matrix or the fundamental matrix from eight or more point correspondences.

Background

When dealing with multiple images, it is essential to determine the relative geometry between them, which is known as the epipolar geometry. Between two images of a scene, given a pair of corresponding image points ($\mathbf{m}_i, \mathbf{m}'_i$), the following epipolar constraint must be satisfied:

$$\tilde{\mathbf{m}}'_i^T \mathbf{M} \tilde{\mathbf{m}}_i = 0 , \quad (1)$$

where $\tilde{\mathbf{m}}_i = \begin{bmatrix} \mathbf{m}_i \\ 1 \end{bmatrix}$ is point \mathbf{m}_i in homogeneous coordinates. Similarly, $\tilde{\mathbf{m}}'_i$ is point \mathbf{m}'_i in homogeneous coordinates. Matrix \mathbf{M} is a 3×3 matrix. If the images are calibrated with known intrinsic camera parameters and the image points are expressed in the normalized image coordinate system, then matrix \mathbf{M} is known as

the *essential matrix*, and is usually denoted by \mathbf{E} ; otherwise, matrix \mathbf{M} is known as the *fundamental matrix* and is usually denoted by \mathbf{F} . Both essential matrix and fundamental matrix must satisfy certain properties, but the common property is that it is a rank-2 matrix, i.e., the determinant of matrix \mathbf{M} is equal to zero.

Theory

The 8-point algorithm ignores the constraints on the elements of matrix \mathbf{M} and treats them independently. Let us define a 9-D vector \mathbf{x} using the elements of matrix \mathbf{M} such that

$$\mathbf{x} = [M_{11}, M_{21}, M_{31}, M_{12}, M_{22}, M_{32}, M_{13}, M_{23}, M_{33}]^T , \quad (2)$$

where M_{ij} is the (i, j) element of matrix \mathbf{M} . Let $\mathbf{m} = [u, v]^T$. Then, the epipolar constraint (Eq. 1) can be rewritten as

$$\mathbf{a}_i^T \mathbf{x} = 0 , \quad (3)$$

where

$$\mathbf{a}_i = [u_i \tilde{\mathbf{m}}'_i^T, v_i \tilde{\mathbf{m}}'_i^T, \tilde{\mathbf{m}}'_i^T]^T . \quad (4)$$

It is clear that \mathbf{x} can only be determined up to a scale factor.

Given eight or more point correspondences, \mathbf{x} can be determined as follows. Each point correspondence yields one epipolar equation (Eq. 1). With N ($N \geq 8$) point correspondences, we can stack them together into a vector equation as follows:

$$\mathbf{A}^T \mathbf{x} = \mathbf{0} , \quad (5)$$

with $\mathbf{A} = [\mathbf{a}_1, \dots, \mathbf{a}_N]$. The solution to \mathbf{x} can be obtained by minimizing the least squares, subject to $\|\mathbf{x}\| = 1$. With Lagrange multiplier, this is equivalent to minimizing

$$\|\mathbf{A}^T \mathbf{x}\|^2 + \lambda(1 - \|\mathbf{x}\|^2) . \quad (6)$$

With simple algebra, it can be found that the solution to \mathbf{x} is the eigenvector of the 9×9 matrix $\mathbf{A} \mathbf{A}^T$ associated with the smallest eigenvalue.

When the image coordinates (u_i, v_i) are in pixels, the elements of matrix \mathbf{A} have orders of difference in value, and matrix $\mathbf{A} \mathbf{A}^T$ may not be well conditioned. One remedy is to pre-normalize the image

points, and several solutions are examined in [1]. One simplest approach is to perform a scaling and translation such that all image coordinates are within $[-1, 1]$. Compared with using directly the pixel coordinates, significant improvement in accuracy has been observed.

The matrix \mathbf{M} estimated above is obtained by ignoring its property. For example, the estimated \mathbf{M} is usually not rank-2. To obtain the closest rank-2 matrix, “closest” in terms of Frobenius norm, we perform a singular value decomposition on \mathbf{M} , i.e.,

$$\mathbf{M} = \mathbf{U}\mathbf{S}\mathbf{V}, \quad (7)$$

where $\mathbf{S} = \text{diag}(s_1, s_2, s_3)$ with $s_1 \geq s_2 \geq s_3 \geq 0$. Replacing the smallest singular value by zero, i.e., $\hat{\mathbf{S}} = \text{diag}(s_1, s_2, 0)$, then

$$\mathbf{U}\hat{\mathbf{S}}\mathbf{V} \equiv \hat{\mathbf{M}} \quad (8)$$

is the optimal rank-2 matrix.

For more details about the epipolar geometry, the reader is referred to [2, 3]. The reader is referred to [4] for a review of various methods for determining the epipolar geometry (the essential matrix and the fundamental matrix), to [5] for a study of the relationship between various optimization criteria, and to [6] for how to obtain a more robust Euclidean motion and structure estimation via the estimation of fundamental matrix.

References

1. Hartley R (1997) In defense of the eight-point algorithm. *IEEE Trans Pattern Anal Mach Intell* 19(6):580–593
2. Faugeras O, Luong QT, Papadopoulo T (2001) The geometry of multiple images. MIT, Cambridge
3. Hartley R, Zisserman A (2000) Multiple view geometry in computer vision. Cambridge University Press, Cambridge/New York
4. Zhang Z (1998) Determining the epipolar geometry and its uncertainty: a review. *Int J Comput Vis* 27(2):161–195
5. Zhang Z (1998) On the optimization criteria used in two-view motion analysis. *IEEE Trans Pattern Anal Mach Intell* 20(7):717–729
6. Zhang Z (1997) Motion and structure from two perspective views: from essential parameters to euclidean motion via fundamental matrix. *J Opt Soc Am A* 14(11):2938–2950

Ellipse Fitting

Zhi-Yong Liu

Institute of Automation, Chinese Academy of Sciences, Beijing, P. R. China

Synonyms

[Ellipse matching](#)

Definition

Fit one or more ellipses to a set of image points.

Background

Fitting geometric primitives to image data is a basic task in pattern recognition and computer vision. The fitting allows reduction and simplification of image data to a higher level with certain physical meanings. One of the most important primitive models is ellipse, which, being a projective projection of a circle, is of great importance for a variety of computer vision-related applications.

Ellipse fitting methods can be roughly divided into two categories: least square fitting and voting/clustering. Least square fitting, though usually fast to implement, requires the image data presegmented and is sensitive to outliers. On the other hand, voting techniques can detect multiple ellipses at once and exhibit some robustness against noise but suffer from a heavier computational and memory load. Furthermore, most of standard ellipse-fitting methods cannot be directly used in real-world applications involving a noisy environment. Thus, the arc finding-based techniques will also be described in the entry, with emphasis on ellipse fitting in complicated images, though, strictly speaking, they cannot be taken as a counterpart of the above two categories.

Due to space limitation, there are some other techniques that cannot be covered by the entry but with some references listed in the recommended reading, such as the moment [16] and genetic algorithm [12]-based methods.

Theory

Least Square Fitting

Least square fitting is realized by minimizing some distance measure between ellipse and image data as follows:

$$\Theta = \arg \min_{\Theta} \sum_{i=1}^N D(\Theta, \mathbf{x}_i)^2, \quad (1)$$

where $D(\Theta, \mathbf{x}_i)$ denotes some distance between pixel $\mathbf{x}_i = [x_i, y_i]^T$ and the ellipse specified by Θ .

There are two main distance measures for ellipse fitting: geometric distance and algebraic distance. In efficiency, the algebraic distance-based techniques outperform its counterpart because a direct solution of Eq. (1) is obtainable by using algebraic distance. However, the algebraic fitting suffers from the drawback of bias estimation and unclear physical interpretations on the estimated errors and fitting parameters. For instance, some algebraic fitting results are not invariant to the coordinates transformation of image data.

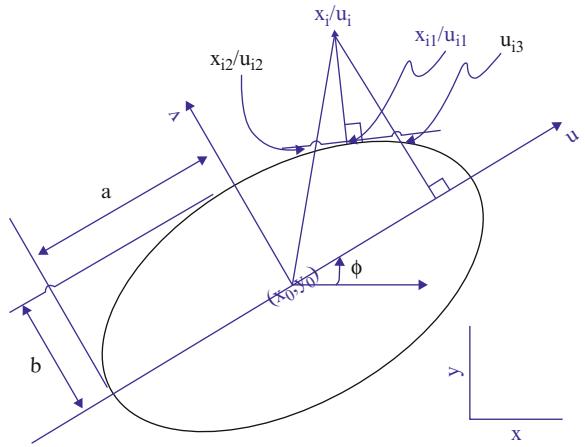
Geometric Fitting

An ellipse takes the following equation:

$$\frac{[(x - x_0) \cos \phi + (y - y_0) \sin \phi]^2}{a^2} + \frac{[(x - x_0) \sin \phi + (y - y_0) \cos \phi]^2}{b^2} = 1, \quad (2)$$

where x_0, y_0 are coordinates of center, ϕ is the orientation, and a, b are the semiaxis, as shown in Fig. 1. Geometric distance, also known as shortest or orthogonal distance, is defined by the distance between one image point and its orthogonal projection upon the ellipse, as illustrated by $D_1(\mathbf{x}_i) = \|\mathbf{x}_i - \mathbf{x}_{i1}\|$ in Fig. 1. In such a setting, the ellipse fitted becomes actually a principal curve, which is best in the sense of mean square reconstruction error minimization. However, the distance D_1 is analytically intractable. There exist some techniques to tackle the problem, unavoidably involving some iterative numerical algorithm.

Ahn et al. [1] use a Gaussian-Newton algorithm for geometric fitting, by implicitly describing the orthogonal projection point \mathbf{x}_1 . A temporary coordinate system



Ellipse Fitting, Fig. 1 An ellipse with center at (x_0, y_0) , orientation ϕ , and semiaxis a and b . The projection of point \mathbf{x}_i , or \mathbf{u}_i in coordinate system uv , upon the ellipse is the point with the shortest distance on the ellipse, denoted by \mathbf{x}_{i1} (\mathbf{u}_{i1})

uv is introduced as follows:

$$\mathbf{u} = \mathbf{R}(\mathbf{x} - \mathbf{x}_0), \quad (3)$$

where $\mathbf{R} = \begin{pmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{pmatrix}$ is a rotation matrix.

In coordinate system uv , the ellipse is transformed with center at the origin and without rotation, i.e., with the simple formulation as

$$f_1(u, v) = \frac{u^2}{a^2} + \frac{v^2}{b^2} - 1 = 0. \quad (4)$$

For a data point \mathbf{u}_i , the tangent line through its projection point \mathbf{u}_{i1} on the ellipse is perpendicular to the line connecting the two points \mathbf{u}_i and \mathbf{u}_{i1} :

$$f_2(u, v) = b^2 u(v_i - v) - a^2 v(u_i - u) = 0. \quad (5)$$

Then, a generalized Newton method is employed to find \mathbf{u}_{i1} as follows:

$$\begin{aligned} \mathbf{Q}_n \Delta \mathbf{u} &= -\mathbf{f}(\mathbf{u}_n) \\ \mathbf{u}_{n+1} &= \mathbf{u}_n + \Delta \mathbf{u} \end{aligned} \quad (6)$$

where $\mathbf{f} \triangleq (f_1, f_2)^T$, and the Jacobian matrix

$$\mathbf{Q} = \begin{pmatrix} b^2 u, & a^2 v \\ (a^2 - b^2)v + b^2 v_i, & (a^2 - b^2)u - a^2 u_i \end{pmatrix}. \quad (7)$$

It is noted that several solutions (maximum to 4) could be found by the iteration above. In order to get the right one, one can choose a proper initial value as

$$\mathbf{u}_0 = 0.5(\mathbf{u}_{i2} + \mathbf{u}_{i3}),$$

where $\mathbf{u}_{i2} = \mathbf{u}_i \frac{ab}{\sqrt{b^2 u_i^2 + a^2 v_i^2}}$ and $\mathbf{u}_{i3} = \begin{cases} (u_i, \text{sign}(v_i) \frac{b}{a} \sqrt{a^2 - u_i^2})^T & \text{if } |u_i| < a, \\ (\text{sign}(u_i)a, 0)^T & \text{else.} \end{cases}$, as illustrated in Fig. 1.

Once \mathbf{u}_1 has been found, through a backward transformation of Eq. (3), the error distance vector becomes

$$\mathbf{D}_1(\mathbf{x}_i) = \mathbf{x}_i - \mathbf{x}_{i1} = \mathbf{R}^{-1}(\mathbf{u}_i - \mathbf{u}_{i1}). \quad (8)$$

By implicitly describing the orthogonal projection \mathbf{u}_1 through f_1 and f_2 defined by Eqs. 4 and 5, the Jacobian matrix with respect to the five parameters $\theta = (x_0, y_0, a, b, \phi)^T$ is as follows:

$$\mathbf{J}_{\mathbf{x}_{i1}, \theta} = (\mathbf{R}^{-1}\mathbf{Q}^{-1}\mathbf{B})|_{\mathbf{u}=\mathbf{u}_1}, \quad (9)$$

where \mathbf{Q} is given by Eq. 7 and $\mathbf{B} = (\mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_3, \mathbf{B}_4, \mathbf{B}_5)$ with

$$\begin{aligned} \mathbf{B}_1 &= (b^2 u \cos \phi - a^2 v \sin \phi, b^2(v_i - v) \cos \phi \\ &\quad + a^2(u_i - u) \sin \phi)^T, \\ \mathbf{B}_2 &= (b^2 u \sin \phi + a^2 v \cos \phi, b^2(v_i - v) \sin \phi \\ &\quad - a^2(u_i - u) \cos \phi)^T, \\ \mathbf{B}_3 &= (a(b^2 - v^2), 2av(u_i - u))^T, \\ \mathbf{B}_4 &= (b(a^2 - u^2), -2bu(v_i - b))^T, \\ \mathbf{B}_5 &= ((a^2 - b^2)uv, (a^2 - b^2)(u^2 - v^2 - uu_i + vv_i))^T. \end{aligned}$$

Finally, based on Eqs. (8) and (9), the fitting can be accomplished by a Gaussian-Newton iteration as

$$\theta_{n+1} = \theta_n + \lambda \mathbf{J}_{\mathbf{x}_{i1}, \theta}^{-1} \mathbf{D}_1(\mathbf{x}_i), \quad (10)$$

where λ denotes a step-size.

Instead of the shortest distance, an alternative approach was proposed to use radial distance, as illustrated by $D_2(\mathbf{x}_i) = \|\mathbf{x}_i - \mathbf{x}_{i2}\|$ in Fig. 1, where \mathbf{x}_{i2} is the intersection of the ellipse and the radial

line passing through \mathbf{x}_i . Distance D_2 is analytically obtained as [9]

$$D_2(\Theta, \mathbf{x}) = \left| \frac{ab}{\sqrt{\kappa}} - 1 \right| \sqrt{(x - x_0)^2 + (y - y_0)^2}, \quad (11)$$

with $\kappa \triangleq a^2(\cos \phi(y - y_0) - \sin \phi(x - x_0))^2 + b^2(\cos \phi(x - x_0) + \sin \phi(y - y_0))^2$. However, solution of Eq. 1 can still not be directly reached. Usually, some iterative technique is still required to get the fitting results, such as stochastic gradient algorithm.

Some other developments on geometric fitting are referred to [2, 13].

Algebraic Fitting

In fact, a more commonly used distance measure for ellipse fitting is algebraic distance, which defines the distance by the second-order polynomial

$$\mathcal{D}(\Theta, \mathbf{u}) = ax^2 + bxy + cy^2 + dx + ey + f, \quad (12)$$

with the constraint

$$b^2 - 4ac < 0 \quad (13)$$

and with $\Theta = [a, b, c, d, e, f]^T$. The problem of Eq. (1) with the distance measure given by Eq. (12) has been extensively studied in the name of conic fitting. In order to fit an ellipse, though the general problem of conic fitting could be solved directly, the constraint of Eq. (13) generally makes the methods iterative. By constraining $b^2 - 4ac = -1$ and by transferring the problem to solving a generalized eigensystem, Fitzgibbon et al. [3] propose a fast and direct ellipse fitting method. Specifically, the problem is reformulated as

$$\min_{\Theta} \|\mathbf{D}\Theta\|^2 \text{ subject to } \Theta^T \mathbf{C} \Theta = 1, \quad (14)$$

where $\mathbf{D} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N]^T$ with $\mathbf{u}_i = [x^2, xy, y^2, x, y, 1]$, and the constraint matrix \mathbf{C} is defined as

$$\mathbf{C} = \begin{pmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{0} \end{pmatrix}, \quad (15)$$

where $\mathbf{0} \in \mathbb{R}^{3 \times 6}$ is a null matrix. By introducing Lagrange multipliers λ and differentiating, the conditions for the optimal Θ become

$$\begin{aligned} \mathbf{S}\Theta &= \lambda\mathbf{C}\Theta, \\ \Theta^T\mathbf{C}\Theta &= 1, \end{aligned} \quad (16)$$

where $\mathbf{S} = \mathbf{D}^T\mathbf{D}$. Thus, problem (14) can be solved by finding the eigenvector Θ_i of the generalized eigen-system defined by Eq. (16). It is also noted that there are six solution pairs (λ_i, Θ_i) . By definition, the one corresponding to the minimal positive eigenvalue λ_i is chosen as the solution because

$$\|\mathbf{D}\Theta\|^2 = \Theta^T\mathbf{D}^T\mathbf{D}\Theta = \Theta^T\mathbf{S}\Theta = \lambda\Theta^T\mathbf{C}\Theta = \lambda. \quad (17)$$

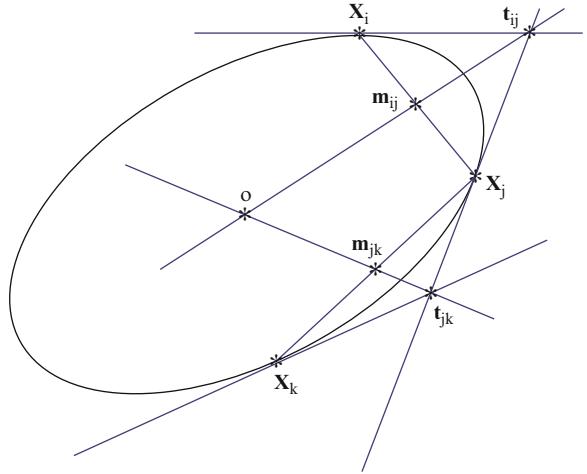
Some other developments on algebraic fitting are referred to [5, 14].

Voting-Based Techniques

Different from the least square fitting techniques which can fit only one primitive, the voting-based methods, consisting of mainly the Hough transform, can detect multiple primitives at once. Even for single ellipse fitting, the Hough transform-based techniques are more robust against outliers. On the other hand, however, Hough transform usually suffers from a larger computational and memory load, due to the fact that an ellipse involves five parameters. To alleviate the problem, there exist some modifications of the original method.

One popular way to tackle the problem is to decompose the five-dimensional space into several subspace with lower dimensions, thanks to some special geometric features of ellipse. A common decomposition method is first to locate the center of the ellipse, and then to specify the remaining three parameters. The center of an ellipse can be located in several ways, based on different geometric features of ellipse.

One geometric feature of ellipse frequently used is described as follows [18]. Let \mathbf{x}_i and \mathbf{x}_j be two points on an ellipse, and find their midpoint $\mathbf{m}_{ij} = \frac{1}{2}(\mathbf{x}_i + \mathbf{x}_j)$



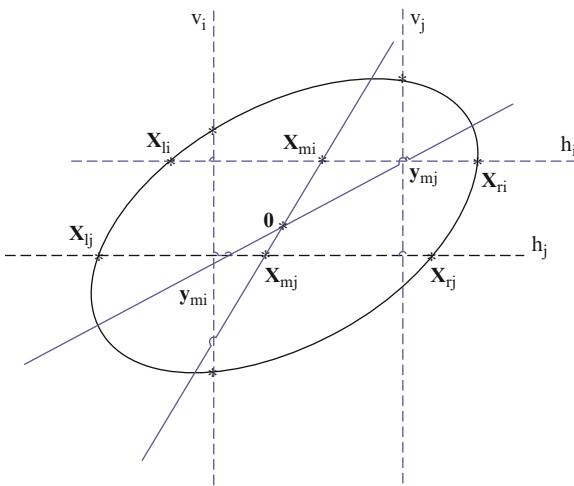
Ellipse Fitting, Fig. 2 Intersection of lines $\overline{t_{ij}m_{ij}}$ and $\overline{t_{jk}m_{jk}}$ is center of the ellipse

and the intersection point \mathbf{t}_{ij} of their tangent lines, as illustrated in Fig. 2. Then, the line connecting the two points \mathbf{m}_{ij} and \mathbf{t}_{ij} is a radial line that passes through center of the ellipse. That is, any point pairs on an ellipse with unparallel tangents can produce a radial line. Consequently, the center candidates can be located in the following way:

1. Generate the radial lines based on every point pairs in the image.
2. Define a 2-dimensional histogram on $x - y$ plane of the image, and record each radial line by incrementing the histogram bin through which the line passes.
3. Find all local maxima in the 2-dimensional histogram as the center candidates.

Another interesting feature of ellipse arises from its symmetry [6]. For two horizontal lines h_i and h_j , their intersections with an ellipse are denoted by \mathbf{x}_{li} , \mathbf{x}_{ri} and \mathbf{x}_{lj} , \mathbf{x}_{rj} , respectively, as illustrated in Fig. 3. The line $\overline{\mathbf{x}_{mi}\mathbf{x}_{mj}}$ will pass through center of the ellipse, where the midpoint $\mathbf{x}_{mi} \triangleq \frac{1}{2}(\mathbf{x}_{li} + \mathbf{x}_{ri})$ and $\mathbf{x}_{mj} \triangleq \frac{1}{2}(\mathbf{x}_{lj} + \mathbf{x}_{rj})$. Similarly, for two vertical lines v_i and v_j , the line $\overline{\mathbf{y}_{mi}\mathbf{y}_{mj}}$ also passes through center of the ellipse, where line $\overline{\mathbf{y}_{mi}\mathbf{y}_{mj}}$ is gotten in a similar way as $\overline{\mathbf{x}_{mi}\mathbf{x}_{mj}}$. Thus, intersection of the two lines $\overline{\mathbf{x}_{mi}\mathbf{x}_{mj}}$ and $\overline{\mathbf{y}_{mi}\mathbf{y}_{mj}}$ gives center of the ellipse. Based on the geometric feature, a center detection method is given as follows:

1. Fully horizontally scan the image, and find the midpoints \mathbf{m}_{hi} of every possible point pairs on each scanning line.



Ellipse Fitting, Fig. 3 Intersection of lines $\overline{x_{mi}x_{mj}}$ and $\overline{y_{mi}y_{mj}}$ is center of the ellipse

2. A 2-dimensional Hough transform is applied to find all of the lines l_h formed by the midpoints \mathbf{m}_{hi} ($i = 1, 2, \dots$).
3. All lines l_v are found in a similar way as above on vertical scanning.
4. Every intersection of l_h and l_v gives a center candidate.

Once the center candidates have been extracted, detection of the three remaining parameters can be reached through a standard voting process in an accumulator space or can be further decomposed into detection of orientation and semiaxis, respectively. A typical decomposition strategy for the three parameters is to find first axis ratio $\frac{a}{b}$ and orientation ϕ and then axis length [4]. Axis ratio and orientation satisfy the following relationship:

$$\frac{a^2}{b^2} = \frac{(x_i - x_0) \cos \phi + (y_i - y_0) \sin \phi}{(x_i - x_0) \sin \phi - (y_i - y_0) \cos \phi} \tan(\theta_i - \phi + \frac{\pi}{2}), \quad (18)$$

where θ_i denotes the angle of the tangent line through \mathbf{x}_i and x_0 , y_0 have been extracted in the previous stage. Thus, a two-dimensional plane can be built based on Eq. (18) for each point on the ellipse, and consequently the axis ratio and orientation can be estimated by finding the maxima in the plane. It is noted that, for each one center candidate, only the data points that

participate in the extraction of center are used in the above voting process.

Finally, ellipse fitting can be accomplished by specifying a or b . Taking a as an example as follows, which satisfies

$$a = \sqrt{(x_i - x_0)^2 + \gamma(y_i - y_0)^2}, \quad (19)$$

where $\gamma \triangleq \frac{a}{b}$ is gotten in the previous step. Hence, voting on a one-dimensional accumulator by the data points pertaining each candidate will retrieve the axis length of the target.

There are also some improvements of the Hough transform that are used to alleviate the computational load, such as the randomized Hough transform. Because discussion on the variants of Hough transform is out of the scope of current entry, readers are referred to some other related entries or some recommended readings such as [11].

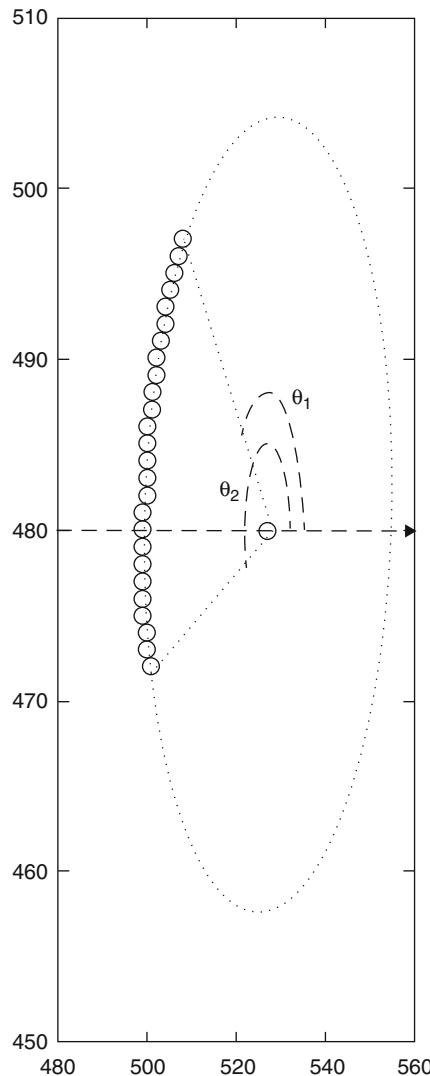
Some other developments on voting-based fitting are referred to [15, 17].

Arc Finding-Based Techniques

Arc finding-based techniques accomplish ellipse fitting by finding elliptic arcs in the edge map. Thus, they in general require edge detection as a preprocessing. Arc finding is of particular interest for ellipse fitting, thanks to the fact that any fragment/elliptic arc (longer than five pixels) of an ellipse can retrieve the whole one. This makes such techniques suitable for ellipse fitting in complicated image because it is usually easier to extract arcs than a whole ellipse from an edge map. Arc finding-based ellipse fitting generally takes the following three steps: arc finding, arc fitting and grouping, and ellipse fitting.

In order to extract ellipse, the arc found should be a *neat* curve without any branches. A simple approach to get neat curve is described as follows [9]: first, a thinning operator is applied to make the edge with one pixel width; second, delete those pixels that have more than two pixels in its 8-neighbor field; finally, a standard chain finding algorithm is used to find neat curves.

Each curve is further fitted by an ellipse, by using a geometric fitting technique mentioned above. The reason for choosing geometric fitting lies in that, first, geometric fitting provides a proper criterion

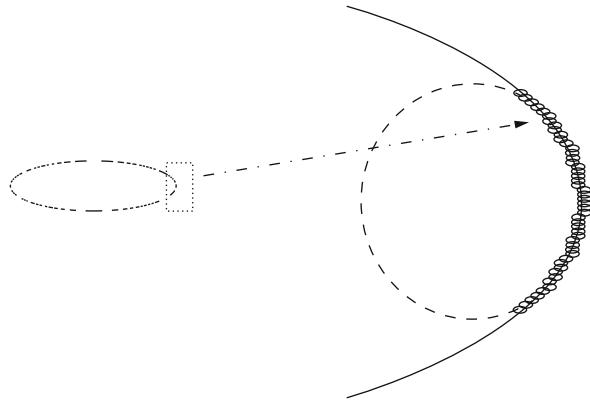


Ellipse Fitting, Fig. 4 The beginning and ending angles of an elliptic arc

(error measure) to evaluate the curve to be a qualified elliptic arc or not and, second, geometric fitting gives a more accurate result than algebraic fitting and Hough transform on a small elliptic fragment, which is frequently encountered since the neat curve finding tends to break the curves with branches into several shorter ones. Finally, each curve is characterized by a 8-dimensional vector

$$[x_0, y_0, a, b, \phi, e, \theta_1, \theta_2]^T, \quad (20)$$

where $e = \frac{1}{N} \sum_{i=1}^N \mathcal{D}(\Theta, \mathbf{x}_i)^2$ denotes the mean square fitting error and θ_1 and θ_2 denote its beginning



Ellipse Fitting, Fig. 5 Fitting results of a small elliptic fragment: in the image on right side, dash ellipse is gotten by algebraic fitting and solid ellipse by geometric fitting

and ending angles, i.e., its direction, as illustrated in Fig. 4. The curves whose fitting error is smaller than a predefined threshold are chosen as qualified elliptic arcs. The elliptic arcs belonging to one same ellipse are then grouped together according to the following two rules:

- Their positions and shapes are close to each other, by the first five parameters.
- They are mutually complementary in direction to form an entire ellipse, by the last two parameters.

During the arc grouping, the data points belonging to each single ellipse are also segmented. Consequently, each ellipse can be finally fitted by least square fitting on these segmented point sets individually.

It is noticed that the arc finding methods employ an integrated framework since its fitting technique comes from least square fitting. However, such integration seems to be unavoidable in real applications, especially in unstructured environments that usually involve heavy noises.

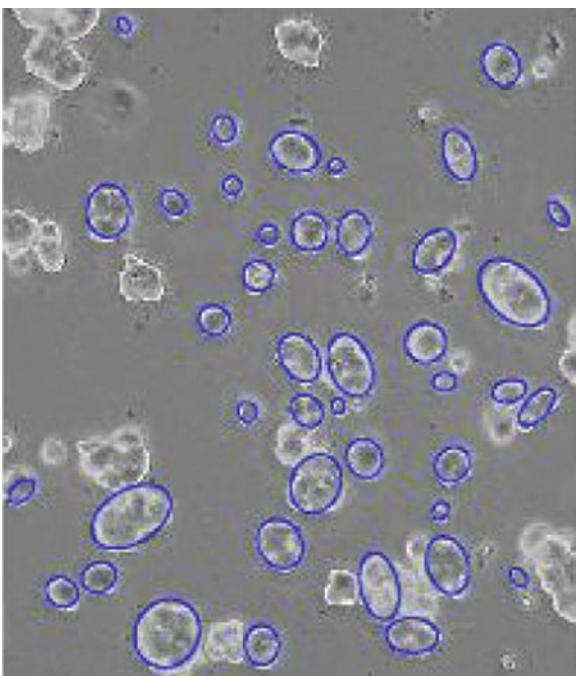
Some other developments on arc finding-based techniques are referred to [8, 10].

Open Problems

Statistical bias of algebraic fitting make the fitting results tend to shrink, especially on small fragments or on data points with heavier noises. Although there are some works on general conic fitting to remove



Ellipse Fitting, Fig. 6 Real-world application: ellipse fitting results on gear image



Ellipse Fitting, Fig. 7 Real-world application: ellipse fitting results on blood cell image

the bias [7], discussion on direct ellipse fitting is lacked. Since direct algebraic fitting seems to be the fastest algorithm for ellipse fitting, it is of theoretical and practical significance to make some progress on removing bias in the fitting process.

On the other hand, although there are some methods proposed for ellipse fitting in complicated images, it still remains a big challenge to fit ellipse in uncontrolled real-world environments.

Experimental Results

It is not intended to demonstrate all of the algorithms described above. Only two experimental results are given, with the first one to demonstrate the bias of algebraic fitting on a small elliptic arc and the second one to give one glimpse of state-of-art real-world application in noisy environments of ellipse fitting.

Image points used in the first experiments are fetched from a small elliptic fragment, as shown in Fig. 5. The dash ellipse is the fitting result of algebraic fitting, which shrinks to be smaller than the right one as denoted by the bigger solid ellipse that is resulted from geometric fitting.

Two real-world applications of ellipse fitting on holes of gear and blood cells are shown in Figs. 6 and 7, respectively.

References

1. Ahn SJ, Rauth W, Warnecke HJ (2001) Least-squares orthogonal distances fitting of circle, sphere, ellipse, hyperbola, and parabola. *Pattern Recognit* 34:2283–2302
2. Cui Y, Weng J, Reynolds H (1996) Estimation of ellipse parameters using optimal minimum variance estimator. *Pattern Recognit Lett* 17:309–316
3. Fitzgibbon AW, Pilu M, Fisher RB (1999) Direct least-squares fitting of ellipses. *IEEE Trans Pattern Anal Mach Intell* 21(5):476–480
4. Guil N, Zapata EL (1997) Lower order circle and ellipse hough transform. *Pattern Recognit* 30:1729–1744
5. Halir R, Flusser V (1998) Numerically stable direct least squares fitting of ellipses. In: WSCG'98 conference proceedings, Plzen-Bory
6. Ho CT, Chen LH (1995) A fast ellipse/circle detector using geometric symmetry. *Pattern Recognit* 28:117–124
7. Kanatani K (1994) Statistical bias of conic fitting and renormalization. *IEEE Trans Pattern Anal Mach Intell* 16(3): 320–326
8. Kim E, Haseyama V, Kitajima H (2002) Fast and robust ellipse extraction from complicated images. In: Proceedings of IEEE international conference on information technology and applications, Bathurst, NSW, Australia
9. Liu ZY, Qiao H (2009) Multiple ellipses detection in noisy environments: a hierarchical approach. *Pattern Recognit* 42:2421–2433

10. Mai F, Hung YS, Zhong H, Sze WF (2008) A hierarchical approach for fast and robust ellipse extraction. *Pattern Recognit* 8(41):2512–2524
11. McLaughlin RA (1998) Randomized hough transform: improved ellipse detection with comparison. *Pattern Recognit Lett* 19:299–305
12. Roth G, Levine MD (1994) Geometric primitive extraction using a genetic algorithm. *IEEE Trans Pattern Anal Mach Intell* 16(9):901–905
13. Spath H (1997) Orthogonal distance fitting by circles and ellipses with given data. *Comput Stat* 12:343–354
14. Taubin G (1991) Estimation of planar curves, surfaces and non-planar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE Trans Pattern Anal Mach Intell* 13(11):1115–1138
15. Tsuji S, Matsumoto F (1978) Detection of ellipses by a modified hough transformation. *IEEE Trans Comput* 25: 777–781
16. Voss K, Suesse H (1997) Invariant fitting of planar objects by primitives. *IEEE Trans Pattern Anal Mach Intell* 19(1):80–84
17. Yipa KK, Tama KS, Leung NK (1992) Modification of hough transform for circles and ellipses detection using a 2-dimensional array. *Pattern Recognit* 25:1007–1022
18. Yuen HK, Illingworth J, Kittler J (1989) Detecting partially occluded ellipses using the hough transform. *Image Vis Comput* 7:31–37

Ellipse Matching

► [Ellipse Fitting](#)

EM-Algorithm

► [Expectation Maximization Algorithm](#)

Environment Mapping

► [Image-Based Lighting](#)

Epipolar Constraint

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Synonyms

Coplanarity constraint

Related Concepts

► [Epipolar Geometry](#)

Definition

Epipolar constraint states that in stereovision with two cameras, given a point in one image, its corresponding point in the other image must lie on a line, known as the *epipolar line*. This constraint arises from the fact that the pair of corresponding image points and the optical centers of the two cameras must lie on a plane (known as *coplanarity constraint*), and the intersection of this plane with the image plane is the epipolar line.

Background

See entry ► [Epipolar Geometry](#) for details.

Epipolar Geometry

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Synonyms

Multiple view geometry; Multiview geometry

Related Concepts

► [Epipolar Constraint](#); ► [Essential Matrix](#);
► [Fundamental Matrix](#)

Definition

Epipolar geometry describes the geometric relationship between two camera systems. It is captured by a 3×3 matrix known as *essential matrix* for calibrated cameras and as *fundamental matrix* for uncalibrated cameras. It states that for a point observed in one camera, its corresponding point in the other camera must lie on a line. This is known as the *epipolar constraint*.

It reduces the search space of correspondences from two dimensions to one dimension. In motion and structure from motion, this constraint is also known as *coplanarity constraint* because the optical centers of the cameras and a pair of corresponding image points must lie in a single plane.

Background

The epipolar geometry exists between any two camera systems. Consider the case of two cameras as shown in Fig. 1. Let C and C' be the optical centers of the first and second cameras, respectively. Given a point \mathbf{m} in the first image, its corresponding point in the second image is constrained to lie on a line called the *epipolar line* of \mathbf{m} , denoted by $\mathbf{l}'_{\mathbf{m}}$. The line $\mathbf{l}'_{\mathbf{m}}$ is the intersection of the plane Π , defined by \mathbf{m} , C , and C' (known as the *epipolar plane*), with the second image plane \mathcal{I}' . This is because image point \mathbf{m} may correspond to an arbitrary point on the semi-line CM (M may be at infinity) and that the projection of CM on \mathcal{I}' is the line $\mathbf{l}'_{\mathbf{m}}$. Furthermore, one observes that all epipolar lines of the points in the first image pass through a common point \mathbf{e}' , which is called the *epipole*. Epipole \mathbf{e}' is the intersection of the line CC' with the image plane \mathcal{I}' . This can be easily understood as follows. For each point \mathbf{m}_k in the first image \mathcal{I} , its epipolar line $\mathbf{l}'_{\mathbf{m}_k}$ in \mathcal{I}' is the intersection of the plane Π^k , defined by \mathbf{m}_k , C , and C' , with image plane \mathcal{I}' . All epipolar planes Π^k thus form a pencil of planes containing the line CC' . They must intersect \mathcal{I}' at a common point, which is \mathbf{e}' . Finally, one can easily see the symmetry of the epipolar geometry. The corresponding point in the first image of each point \mathbf{m}'_k lying on $\mathbf{l}'_{\mathbf{m}_k}$ must lie on the epipolar line $\mathbf{l}_{\mathbf{m}'_k}$, which is the intersection of the same plane Π^k with the first image plane \mathcal{I} . All epipolar lines form a pencil containing the epipole \mathbf{e} , which is the intersection of the line CC' with the image plane \mathcal{I} . The symmetry leads to the following observation. If \mathbf{m} (a point in \mathcal{I}) and \mathbf{m}' (a point in \mathcal{I}') correspond to a single physical point M in space, then \mathbf{m} , \mathbf{m}' , C , and C' must lie in a single plane. This is the well-known *coplanarity constraint* in solving motion and structure from motion problems when the intrinsic parameters of the cameras are known [1].

The computational significance in matching different views is that for a point in the first image,

its correspondence in the second image must lie on the epipolar line in the second image, and then the search space for a correspondence is reduced from 2 dimensions to 1 dimension. This is called the *epipolar constraint*.

If the line linking the two optical centers is parallel to one or both of the image planes, then the epipole in one or both of the images goes to infinity, and the epipolar lines are parallel to each other. Additionally, if the line linking the two optical centers is parallel with the horizontal scanlines of the cameras, then the epipolar lines become horizontal, too. This is the assumption of many stereo algorithms which have horizontal epipolar lines.

Theory

Before proceeding further, the reader is referred to the entry [► Camera Parameters \(Intrinsic, Extrinsic\)](#) for the description of camera perspective projection matrix and intrinsic and extrinsic parameters. It is assumed that the reader is familiar with the notation used in that entry.

Assume that the second camera is brought from the position of the first camera through a rotation \mathbf{R} followed by a translation \mathbf{t} . Thus, any point (X, Y, Z) in the first camera coordinate system has coordinates (X', Y', Z') in the second camera coordinate system such that

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \mathbf{R} \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} + \mathbf{t} \quad (1)$$

where

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad \text{and} \quad \mathbf{t} = \begin{bmatrix} t_X \\ t_Y \\ t_Z \end{bmatrix}.$$

\mathbf{R} has nine components but there are only three degrees of freedom. There are six constraints on \mathbf{R} . Indeed, a rotation matrix \mathbf{R} must satisfy

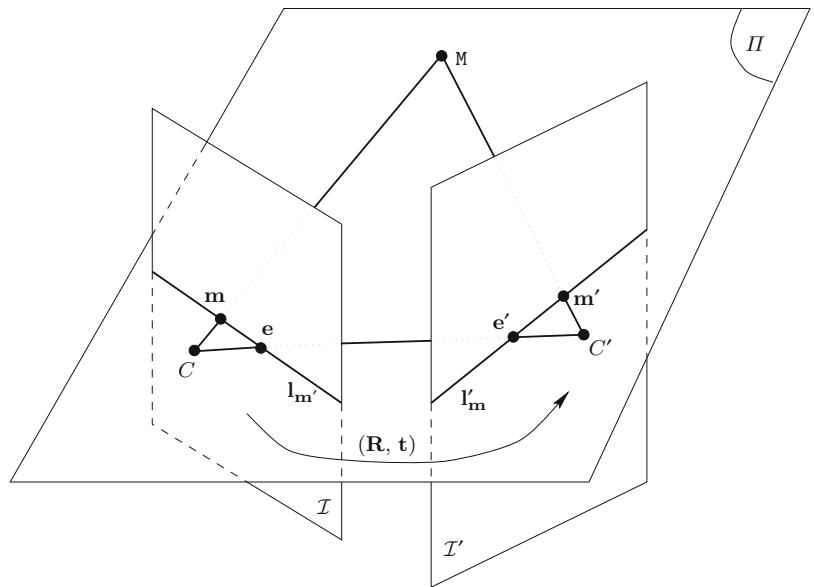
$$\mathbf{R}\mathbf{R}^T = \mathbf{I}, \quad (2)$$

and

$$\det(\mathbf{R}) = 1. \quad (3)$$

Epipolar Geometry, Fig. 1

The epipolar geometry



See, for example [2], for more details on the different representations of the rotation and its properties.

In the following, we first derive the epipolar equation with the normalized image coordinates, then extend it to include the pixel image coordinates, and finally formulate in terms of camera perspective projection matrices.

Working with Normalized Image Coordinates

The two images of a space point $\mathbf{x} = [X, Y, Z]^T$ are $[x, y, 1]^T$ and $[x', y', 1]^T$ in the first and second normalized images, respectively. They are denoted by $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$. Let $\mathbf{x}' = [X', Y', Z']^T$ be the coordinates of the same space point in the second camera coordinate system. From the pinhole model, we have

$$\begin{aligned}\tilde{\mathbf{x}} &= \mathbf{x}/Z, \\ \tilde{\mathbf{x}}' &= \mathbf{x}'/Z'.\end{aligned}$$

Eliminating the structure parameters X and X' using Eq.(1), we obtain

$$\tilde{\mathbf{x}} = \frac{1}{Z}(Z'\mathbf{R}\tilde{\mathbf{x}}' + \mathbf{t}),$$

which contains still two unknown structure parameters Z and Z' . The cross product of the above equation with vector \mathbf{t} yields

$$\mathbf{t} \times \tilde{\mathbf{x}} = \frac{Z'}{Z} \mathbf{t} \times \mathbf{R}\tilde{\mathbf{x}}'.$$

Its dot product (or inner product) with $\tilde{\mathbf{x}}$ gives

$$\tilde{\mathbf{x}}^T \mathbf{t} \times (\mathbf{R}\tilde{\mathbf{x}}') = 0. \quad (4)$$

Here, the quantity Z'/Z has been removed.

Equation (4) is very important in solving motion and structure from motion. Geometrically, it is very clear. The three vectors CC' , $C\tilde{\mathbf{x}}$, and $C'\tilde{\mathbf{x}}'$ are coplanar. When expressed in the first camera coordinate system, they are equal to \mathbf{t} , $\tilde{\mathbf{x}}$, and $\mathbf{R}\tilde{\mathbf{x}}'$, respectively. The coplanarity of the three vectors implies that their mixed product should be equal to 0, which gives Eq.(4).

Let us define a mapping $[\cdot]_x$ from a 3D vector to a 3×3 antisymmetric matrix (also called skew symmetric matrix):

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_x = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}. \quad (5)$$

It is clear that

$$[\mathbf{t}]_x = -[\mathbf{t}]_{x \times}. \quad (6)$$

Using this mapping, we can express the cross product of two vectors by the matrix multiplication of a 3×3 matrix and a three-vector: $\mathbf{t} \times \tilde{\mathbf{x}} = [\mathbf{t}]_x \tilde{\mathbf{x}}$, $\forall \tilde{\mathbf{x}}$. Equation (4) can then be rewritten as

$$\tilde{\mathbf{x}}^T \mathbf{E} \tilde{\mathbf{x}}' = 0, \quad (7)$$

where

$$\mathbf{E} = [\mathbf{t}]_{\times} \mathbf{R}. \quad (8)$$

We call this equation the *epipolar equation*.

Matrix \mathbf{E} is known under the name of the *essential matrix*. It was first proposed by Longuet-Higgins [1] for structure from motion. The essential matrix is determined completely by the rotation and translation between the two cameras. Because $[\mathbf{t}]_{\times}$ is anti-symmetric, we have $\det([\mathbf{t}]_{\times}) = 0$. Thus, we have

$$\det(\mathbf{E}) = \det([\mathbf{t}]_{\times}) \det(\mathbf{R}) = 0. \quad (9)$$

For more properties of the essential matrix, see [3, 4].

Before gaining an insight of Eq.(7), we recall how to represent a line in a plane. Any line can be described by an equation of the form

$$ax + by + c = 0. \quad (10)$$

Thus, the line can be represented by a three-vector $\mathbf{l} = [a, b, c]^T$ such that a point $\tilde{\mathbf{x}} = [x, y, 1]^T$ on it must satisfy

$$\tilde{\mathbf{x}}^T \mathbf{l} = 0. \quad (11)$$

Of course, the three-vector \mathbf{l} is only defined up to a scale factor. Multiplying \mathbf{l} by any nonzero scalar λ gives $\lambda \mathbf{l}$, which describes exactly the same line. If a line goes through two given points $\tilde{\mathbf{x}}_1$ and $\tilde{\mathbf{x}}_2$, we have

$$\tilde{\mathbf{x}}_1^T \mathbf{l} = 0 \quad \text{and} \quad \tilde{\mathbf{x}}_2^T \mathbf{l} = 0,$$

and it is easy to see that the line is represented by

$$\mathbf{l} = \tilde{\mathbf{x}}_1 \times \tilde{\mathbf{x}}_2, \quad (12)$$

that is, the cross product of the two point vectors.

For point $\tilde{\mathbf{x}}' = [x', y', 1]^T$ in the second image, its corresponding point \mathbf{x}' in space must be on the semi-line $C'\mathbf{x}'_{\infty}$ passing through $\tilde{\mathbf{x}}'$, where \mathbf{x}'_{∞} is a point at infinity. From the pinhole model, point \mathbf{x}' can be represented as

$$\mathbf{x}' = \lambda \tilde{\mathbf{x}}' = \lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}, \quad \lambda \in (0, \infty).$$

This is in fact the parametric representation of the semi-line $C'\mathbf{x}'_{\infty}$. If we express this point in the coordinate system of the first camera, we have

$$\mathbf{x} = \mathbf{R}\mathbf{x}' + \mathbf{t} = \lambda \mathbf{R}\tilde{\mathbf{x}}' + \mathbf{t}, \quad \lambda \in (0, \infty).$$

The projection of the semi-line $C'\mathbf{x}'_{\infty}$ on the first camera is still a line, denoted by $\mathbf{l}_{\mathbf{x}'}$, on which the corresponding point in the first image of point \mathbf{x} must lie. The line $\mathbf{l}_{\mathbf{x}'}$ is known as the *epipolar line* of \mathbf{x}' . The epipolar line can be defined by two points. The first point can be obtained by projecting \mathbf{x} with $\lambda = 0$, which gives $\tilde{\mathbf{e}} = \frac{1}{t_Z} \mathbf{t}$, where t_Z is the Z-component of the translation vector \mathbf{t} . This is in fact the projection of the optical center C of the second camera on the first camera and is called the *epipole* in the first image. The second point can be obtained by projecting \mathbf{x} with $\lambda = \infty$, which gives $\tilde{\mathbf{x}}_{\infty} = \frac{1}{r_3^T \tilde{\mathbf{x}}'} \mathbf{R} \tilde{\mathbf{x}}'$, where r_3 is the third row of the rotation matrix \mathbf{R} . As described in the last paragraph, the epipolar line $\mathbf{l}_{\mathbf{x}'}$ is represented by

$$\mathbf{l}_{\mathbf{x}'} = \tilde{\mathbf{e}} \times \tilde{\mathbf{x}}_{\infty} = \mathbf{t} \times \mathbf{R} \tilde{\mathbf{x}}' = \mathbf{E} \tilde{\mathbf{x}}'. \quad (13)$$

Here we have multiplied the original vector by t_Z and $r_3^T \tilde{\mathbf{x}}'$ because, as we said, a three-vector for a line is only defined up to a scalar factor.

If now we reverse the role of the two camera, we find that the epipolar geometry is symmetric for the two cameras. For a given point \mathbf{x} in the first image, its corresponding epipolar line in the second image is

$$\mathbf{l}'_{\mathbf{x}} = \mathbf{E}^T \tilde{\mathbf{x}}.$$

It is seen that the transpose of matrix \mathbf{E} , \mathbf{E}^T , defines the epipolar lines in the second image.

From the above discussion, Eq.(7) says nothing more than that point \mathbf{x} is on the epipolar line $\mathbf{l}_{\mathbf{x}'}$, that is,

$$\tilde{\mathbf{x}}^T \mathbf{l}_{\mathbf{x}'} = 0 \quad \text{with } \mathbf{l}_{\mathbf{x}'} = \mathbf{E} \tilde{\mathbf{x}}',$$

or that point $\tilde{\mathbf{x}}'$ is on the epipolar line $\mathbf{l}'_{\mathbf{x}}$, that is,

$$\mathbf{l}'_{\mathbf{x}}^T \tilde{\mathbf{x}}' = 0 \quad \text{with } \mathbf{l}'_{\mathbf{x}} = \mathbf{E}^T \tilde{\mathbf{x}}.$$

The epipoles are intersections of all epipolar lines. That is, epipoles satisfy all the epipolar line equations.

Let the normalized coordinates of the epipole in the first image be \mathbf{e} . Then, from Eq. (7), \mathbf{e} satisfies

$$\tilde{\mathbf{e}}^T \mathbf{E} \tilde{\mathbf{x}}' = 0 \quad (14)$$

regardless of \mathbf{x}' . This means,

$$\tilde{\mathbf{e}}^T \mathbf{E} = \mathbf{0}^T \quad (15)$$

at anytime. That is,

$$\tilde{\mathbf{e}}^T \mathbf{E} = \tilde{\mathbf{e}}^T [\mathbf{t}]_{\times} \mathbf{R} = \mathbf{0}^T. \quad (16)$$

Since \mathbf{R} is an orthonormal matrix, we have

$$\tilde{\mathbf{e}}^T [\mathbf{t}]_{\times} = \mathbf{0}. \quad (17)$$

The solution is

$$\tilde{\mathbf{e}} = \left[\frac{t_X}{t_Z}, \frac{t_Y}{t_Z}, 1 \right]^T. \quad (18)$$

This is exactly the projection of the optical center of the second camera onto the first image plane, as we have already explained geometrically. For the second image, we have

$$\mathbf{E} \tilde{\mathbf{e}}' = [\mathbf{t}]_{\times} \mathbf{R} \tilde{\mathbf{e}}' = \mathbf{0}. \quad (19)$$

Thus,

$$\mathbf{R} \tilde{\mathbf{e}}' = \mathbf{t}. \quad (20)$$

The position of the epipole can then be determined as

$$\tilde{\mathbf{e}}' = \frac{1}{\mathbf{r}_3' \cdot \mathbf{t}} \mathbf{R}^T \mathbf{t} = \left[\frac{\mathbf{r}_1' \cdot \mathbf{t}}{\mathbf{r}_3' \cdot \mathbf{t}}, \frac{\mathbf{r}_2' \cdot \mathbf{t}}{\mathbf{r}_3' \cdot \mathbf{t}}, 1 \right]^T, \quad (21)$$

where $\mathbf{r}_i' = [r_{1i}, r_{2i}, r_{3i}]^T$, and $i = 1, 2, 3$ are the column vectors of \mathbf{R} .

For the epipole in the first image to go to infinity, we must have

$$t_Z = 0. \quad (22)$$

This means that the translation of the camera has to be within the focal plane of the first camera. For both epipoles in the two images to go to infinity, then

$$\mathbf{r}_3' \cdot \mathbf{t} = 0. \quad (23)$$

This implies that the optical center of the first camera lies in the focal plane of the second camera. Furthermore, if we require the two focal planes to be a single one, then besides $t_Z = 0$, r_{13} and r_{23} have to be 0. Since $\|\mathbf{r}_i'\| = 1$, we have

$$r_{33} = 1. \quad (24)$$

Thus \mathbf{R} is in the form of

$$\mathbf{R} = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (25)$$

This means that the rotation can be only around the optical axis of the cameras.

Substituting Eq. (22) and (25) for (8), we have

$$\begin{aligned} \mathbf{E} &= [\mathbf{t}]_{\times} \mathbf{R} \\ &= \begin{bmatrix} 0 & 0 & t_Y \\ 0 & 0 & -t_X \\ -t_Y \cos \theta - t_X \sin \theta & -t_Y \sin \theta + t_X \cos \theta & 0 \end{bmatrix}. \end{aligned} \quad (26)$$

If we expand the above equation, it is clear that there is only linear terms of the image coordinates, rather than quadric terms in the original form. That means the epipolar lines are parallel in both images and the orientations are independent of the image points.

Working with Pixel Image Coordinates

If two points \mathbf{m} and \mathbf{m}' , expressed in pixel image coordinates in the first and second camera, are in correspondence, they must satisfy the following equation

$$\tilde{\mathbf{m}}^T \mathbf{F} \tilde{\mathbf{m}}' = 0, \quad (27)$$

where

$$\mathbf{F} = \mathbf{A}^{-T} \mathbf{E} \mathbf{A}'^{-1}, \quad (28)$$

and \mathbf{A} and \mathbf{A}' are respectively the of the first and second camera. Equation (27) is easily verified. From the pinhole camera model, the \mathbf{x} are related to the pixel coordinates \mathbf{m} by $\tilde{\mathbf{x}} = \mathbf{A}^{-1} \tilde{\mathbf{m}}$. Plunging it into Eq. (7) yields Eq. (27). This is a fundamental constraint for two pixels to be in correspondence between two images.

As with the normalized image coordinates, the above Eq.(27) can also be derived from the pinhole model. Without loss of generality, we assume

that the world coordinate system coincides with the second camera coordinate system. From the camera perspective projection model, we have

$$\begin{aligned}s\tilde{\mathbf{m}} &= \mathbf{A} [\mathbf{R} \ \mathbf{t}] \begin{bmatrix} \mathbf{M}' \\ 1 \end{bmatrix} \\ s'\tilde{\mathbf{m}}' &= \mathbf{A}' [\mathbf{I} \ \mathbf{0}] \begin{bmatrix} \mathbf{M}' \\ 1 \end{bmatrix}.\end{aligned}$$

Eliminating \mathbf{M}' , s , and s' in the above two equations, we obtain, not at all surprising, Eq. (27).

The 3×3 matrix \mathbf{F} is called the *fundamental matrix*. With this fundamental matrix, we can express the epipolar equation for two unnormalized images in the same form as for the normalized images. Since $\det(\mathbf{E}) = 0$,

$$\det(\mathbf{F}) = 0. \quad (29)$$

\mathbf{F} is of rank 2. Besides, it is only defined up to a scalar factor. If \mathbf{F} is multiplied by an arbitrary scalar, Eq. (27) still holds. Therefore, a fundamental matrix has only seven degrees of freedom. There are only seven independent parameters among the nine elements of the fundamental matrix.

We now derive the expression of the epipoles. The epipole \mathbf{e} in the first image is the projection of the optical center C' of the second camera. Since $C' = \mathbf{0}$, from the pinhole model, we have

$$s_e\tilde{\mathbf{e}} = \mathbf{A} [\mathbf{R} \ \mathbf{t}] \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \mathbf{At}, \quad (30)$$

where s_e is a scale factor. Thus, the epipole $\tilde{\mathbf{e}}$ is equal to \mathbf{At} divided by its third element. Similarly, the epipole \mathbf{e}' in the second image is the projection of the optical center C of the first camera. The optical center is determined by

$$\mathbf{A} [\mathbf{R} \ \mathbf{t}] \begin{bmatrix} C \\ 1 \end{bmatrix} = \mathbf{0},$$

which gives

$$\mathbf{C} = -\mathbf{R}^{-1}\mathbf{t}.$$

Therefore, the epipole \mathbf{e}' is given by

$$s'_e\tilde{\mathbf{e}}' = \mathbf{A}' [\mathbf{I} \ \mathbf{0}] \begin{bmatrix} C \\ 1 \end{bmatrix} = -\mathbf{A}'\mathbf{R}^{-1}\mathbf{t}, \quad (31)$$

that is, it is equal to $-\mathbf{A}'\mathbf{R}^{-1}\mathbf{t}$ divided by the third element of the vector.

Now we show, for a given point \mathbf{m}' in the second image, how to compute the corresponding epipolar line $\mathbf{l}_{\mathbf{m}'}$ in the first image. It is determined by two points. Besides the epipole \mathbf{e} , we need another point. This point can be the projection of any point $\hat{\mathbf{M}}'$ on the optical ray $\langle C', \tilde{\mathbf{m}}' \rangle$. In particular, we can choose $\hat{\mathbf{M}}'$ such that

$$\tilde{\mathbf{m}}' = \mathbf{A}' [\mathbf{I} \ \mathbf{0}] \begin{bmatrix} \hat{\mathbf{M}}' \\ 1 \end{bmatrix} = \mathbf{A}'\hat{\mathbf{M}}',$$

that is, the scale factor is equal to 1. This gives $\hat{\mathbf{M}}' = \mathbf{A}'^{-1}\tilde{\mathbf{m}}'$. The projection of this point in the first camera, $\tilde{\mathbf{m}}$, is given by

$$s_m\tilde{\mathbf{m}} = \mathbf{A} [\mathbf{R} \ \mathbf{t}] \begin{bmatrix} \hat{\mathbf{M}}' \\ 1 \end{bmatrix} = \mathbf{A}(\mathbf{R}\mathbf{A}'^{-1}\tilde{\mathbf{m}}' + \mathbf{t}),$$

where s_m is the scale factor. As already described in Eq. (10) on page 250, a line can be represented by a three-vector defined *up to a scale factor*. According to Eq. (12), the epipolar line $\mathbf{l}_{\mathbf{m}'}$ is given by

$$\begin{aligned}\mathbf{l}_{\mathbf{m}'} &= s_e s_m \tilde{\mathbf{e}} \times \tilde{\mathbf{m}} \\ &= (\mathbf{At}) \times [\mathbf{A}(\mathbf{R}\mathbf{A}'^{-1}\tilde{\mathbf{m}}' + \mathbf{t})] \\ &= (\mathbf{At}) \times (\mathbf{RA}\mathbf{A}'^{-1}\tilde{\mathbf{m}}').\end{aligned}$$

It can be shown that $(\mathbf{Ax}) \times (\mathbf{Ay}) = \det(\mathbf{A}) \mathbf{A}^{-T} (\mathbf{x} \times \mathbf{y})$ for all vectors \mathbf{x} and \mathbf{y} if matrix \mathbf{A} is invertible. Therefore, we have

$$\mathbf{l}_{\mathbf{m}'} = \mathbf{A}^{-T} [\mathbf{t} \times (\mathbf{RA}\mathbf{A}'^{-1}\tilde{\mathbf{m}}')] = \mathbf{F}\tilde{\mathbf{m}}'$$

with \mathbf{F} given by Eq. (28). Then any point \mathbf{m} on the epipolar line of \mathbf{m}' satisfies $\tilde{\mathbf{m}}^T \mathbf{F} \tilde{\mathbf{m}}' = 0$, and this is exactly Eq. (34). Therefore, we obtain geometrically the same equation.

Now we reverse the role of the two images and consider the epipolar line $\mathbf{l}'_{\mathbf{m}}$ in the second image for a given point \mathbf{m} in the first image. Line $\mathbf{l}'_{\mathbf{m}}$ goes through the epipole \mathbf{e}' . We choose the projection of a point $\hat{\mathbf{M}}'$ on the optical ray $\langle C, \tilde{\mathbf{m}} \rangle$ such that

$$\tilde{\mathbf{m}} = \mathbf{A} [\mathbf{R} \ \mathbf{t}] \begin{bmatrix} \hat{\mathbf{M}}' \\ 1 \end{bmatrix},$$

that is, the scale factor is chosen to be 1. This gives

$$\tilde{\mathbf{M}}' = (\mathbf{A}\mathbf{R})^{-1}(\tilde{\mathbf{m}} - \mathbf{At}).$$

Its projection in the second camera gives

$$\begin{aligned}s'_m \tilde{\mathbf{m}}' &= \mathbf{A}' [\mathbf{I} \ \mathbf{0}] \begin{bmatrix} \tilde{\mathbf{M}}' \\ 1 \end{bmatrix} \\ &= \mathbf{A}'(\mathbf{AR})^{-1}(\tilde{\mathbf{m}} - \mathbf{At}) \\ &= \mathbf{A}'\mathbf{R}^{-1}\mathbf{A}^{-1}\tilde{\mathbf{m}} - \mathbf{A}'\mathbf{R}^{-1}\mathbf{t}.\end{aligned}$$

The epipolar line \mathbf{l}'_m is thus represented by

$$\begin{aligned}\mathbf{l}'_m &= s'_e s'_m \tilde{\mathbf{e}}' \times \tilde{\mathbf{m}}' \\ &= -(\mathbf{A}'\mathbf{R}^{-1}\mathbf{t}) \times (\mathbf{A}'\mathbf{R}^{-1}\mathbf{A}^{-1}\tilde{\mathbf{m}}) \\ &= -(\mathbf{A}'\mathbf{R}^{-1})^{-T}(\mathbf{t} \times \mathbf{A}^{-1}\tilde{\mathbf{m}}) \\ &= -\mathbf{A}'^{-T}\mathbf{R}^T[\mathbf{t}]_{\times}\mathbf{A}^{-1}\tilde{\mathbf{m}} \\ &= \mathbf{F}^T\tilde{\mathbf{m}}.\end{aligned}$$

In the above, we have used the following properties:

- $(\mathbf{Ax}) \times (\mathbf{Ay}) = \det(\mathbf{A})\mathbf{A}^{-T}(\mathbf{x} \times \mathbf{y})$, $\forall \mathbf{x}, \mathbf{y}$ if matrix \mathbf{A} is invertible.
- $(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$ if matrices \mathbf{A} and \mathbf{B} are invertible.
- $\mathbf{R}^T = \mathbf{R}^{-1}$ if \mathbf{R} is a rotation matrix.
- $[\mathbf{t}]_{\times}^T = -[\mathbf{t}]_{\times}$ if $[\mathbf{t}]_{\times}$ is an antisymmetric matrix.

It is thus clear that if \mathbf{F} describes epipolar lines in the first image for points given in the second image, then \mathbf{F}^T describes epipolar lines in the second image for points given in the first image. The two images play a symmetric role in the epipolar geometry.

We now compute the epipoles from a different point of view. By definition, all epipolar lines in the first image go through the epipole \mathbf{e} . This implies

$$\tilde{\mathbf{e}}^T \mathbf{F} \tilde{\mathbf{m}}' = 0, \quad \forall \tilde{\mathbf{m}}',$$

or in vector equation form

$$\mathbf{F}^T \tilde{\mathbf{e}} = \mathbf{0}. \tag{32}$$

Plugging Eq. (28) into the above equation gives

$$\mathbf{A}'^{-T}\mathbf{R}^T[\mathbf{t}]_{\times}\mathbf{A}^{-1}\tilde{\mathbf{e}} = \mathbf{0}.$$

Because $[\mathbf{t}]_{\times}\mathbf{t} = \mathbf{0}$, up to a scale factor, we have $s_e \mathbf{A}^{-1}\tilde{\mathbf{e}} = \mathbf{t}$, and thus $s_e \tilde{\mathbf{e}} = \mathbf{At}$. This is exactly

Eq. (30). Similarly, for the epipole \mathbf{e}' in the second image, we have

$$\mathbf{F}\tilde{\mathbf{e}}' = \mathbf{0} \tag{33}$$

or

$$\mathbf{A}^{-T}[\mathbf{t}]_{\times}\mathbf{RA}'^{-1}\tilde{\mathbf{e}}' = \mathbf{0}.$$

This gives $s'_e \mathbf{RA}'^{-1}\tilde{\mathbf{e}}' = -\mathbf{t}$, or $s'_e \tilde{\mathbf{e}}' = -\mathbf{A}'\mathbf{R}^{-1}\mathbf{t}$. This is exactly Eq. (31).

Working with Camera Perspective Projection Matrices

In several applications, for example, in the case of calibrated stereo, the camera perspective projection matrices are given, and we want to compute the epipolar geometry. Let \mathbf{P} and \mathbf{P}' be the projection matrices of the first and second camera. Furthermore, the 3×4 matrix \mathbf{P} is decomposed as the concatenation of a 3×3 submatrix \mathbf{B} and a three-vector \mathbf{b} , that is, $\mathbf{P} = [\mathbf{B} \ \mathbf{b}]$. Similarly, $\mathbf{P}' = [\mathbf{B}' \ \mathbf{b}']$.

From the pinhole model, we have

$$\begin{aligned}s\tilde{\mathbf{m}} &= [\mathbf{B} \ \mathbf{b}] \begin{bmatrix} \tilde{\mathbf{M}}' \\ 1 \end{bmatrix} \\ s'\tilde{\mathbf{m}}' &= [\mathbf{B}' \ \mathbf{b}'] \begin{bmatrix} \tilde{\mathbf{M}}' \\ 1 \end{bmatrix}.\end{aligned}$$

Assume that \mathbf{B} and \mathbf{B}' are invertible, we can compute $\tilde{\mathbf{M}}'$ from each of the above equations:

$$\begin{aligned}\tilde{\mathbf{M}}' &= s\mathbf{B}^{-1}\tilde{\mathbf{m}} - \mathbf{B}^{-1}\mathbf{b} \\ s'\tilde{\mathbf{M}}' &= s'\mathbf{B}'^{-1}\tilde{\mathbf{m}}' - \mathbf{B}'^{-1}\mathbf{b}'.\end{aligned}$$

The right sides of the above equations must be equal, which gives

$$s\mathbf{B}^{-1}\tilde{\mathbf{m}} = s'\mathbf{B}'^{-1}\tilde{\mathbf{m}}' + \mathbf{B}^{-1}\mathbf{b} - \mathbf{B}'^{-1}\mathbf{b}'.$$

Multiplying both sides by \mathbf{B} gives

$$s\tilde{\mathbf{m}} = s'\mathbf{B}\mathbf{B}'^{-1}\tilde{\mathbf{m}}' + \mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}'.$$

Performing a cross product with $\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}'$ yields

$$s(\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}') \times \tilde{\mathbf{m}} = s'(\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}') \times \mathbf{B}\mathbf{B}'^{-1}\tilde{\mathbf{m}}'.$$

Eliminating the arbitrary scalars s and s' by multiplying $\tilde{\mathbf{m}}^T$ from the left (i.e., dot product) gives

$$\tilde{\mathbf{m}}^T \mathbf{F} \tilde{\mathbf{m}}' = 0, \quad (34)$$

where

$$\mathbf{F} = [\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}'] \times \mathbf{B}\mathbf{B}'^{-1}. \quad (35)$$

We thus obtain the epipolar equation in terms of the perspective projection matrices. Again, it is clear that the roles of \mathbf{m} and \mathbf{m}' are symmetric, and we have $\tilde{\mathbf{m}}'^T \mathbf{F}^T \tilde{\mathbf{m}} = 0$.

Now let us show how to compute the epipoles. The epipole \mathbf{e} in the first image is the projection of the optical center C' of the second camera, and the optical center C' is given by

$$C' = -\mathbf{B}'^{-1}\mathbf{b}'.$$

We thus have

$$s_e \tilde{\mathbf{e}} = \mathbf{P} \begin{bmatrix} C' \\ 1 \end{bmatrix} = \mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}', \quad (36)$$

where s_e is a scale factor. Thus, epipole \mathbf{e} is equal to $(\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}')$ divided by its third element. Similarly, the epipole in the second image, \mathbf{e}' , is equal to $(\mathbf{b}' - \mathbf{B}'\mathbf{B}^{-1}\mathbf{b})$ divided by its third element.

Next, we show how, for a given point \mathbf{m}' in the second image, to compute the corresponding epipolar line $\mathbf{l}_{\mathbf{m}'}$ in the first image. The epipolar line must go through the epipole \mathbf{e} . We thus need another point to determine it. This point can be the projection of any point $\hat{\mathbf{M}}$ on the optical ray $\langle C', \tilde{\mathbf{m}}' \rangle$. In particular, we can choose $\hat{\mathbf{M}}$ such that

$$\tilde{\mathbf{m}}' = \mathbf{P}' \begin{bmatrix} \hat{\mathbf{M}}' \\ 1 \end{bmatrix} = \mathbf{B}'\hat{\mathbf{M}}' + \mathbf{b}',$$

that is, the scale factor is equal to 1. This gives $\hat{\mathbf{M}}' = \mathbf{B}'^{-1}(\tilde{\mathbf{m}}' - \mathbf{b}')$. According to the pinhole model, the image $\hat{\mathbf{m}}$ of this point is given by

$$s_m \tilde{\mathbf{m}} = \mathbf{P} \begin{bmatrix} \hat{\mathbf{M}}' \\ 1 \end{bmatrix} = \mathbf{B}\mathbf{B}'^{-1}\tilde{\mathbf{m}}' + (\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}'),$$

where s_m is the scale factor. As already described in Eq. (10) on page 250, a line can be represented by a three-vector defined *up to a scale factor*. According to Eq. (12), the epipolar line $\mathbf{l}_{\mathbf{m}'}$ is given by

$$\begin{aligned} \mathbf{l}_{\mathbf{m}'} &= s_e s_m \tilde{\mathbf{e}} \times \tilde{\mathbf{m}} \\ &= (\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}') \times [\mathbf{B}\mathbf{B}'^{-1}\tilde{\mathbf{m}}' + (\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}')] \\ &= (\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}') \times (\mathbf{B}\mathbf{B}'^{-1}\tilde{\mathbf{m}}'), \end{aligned}$$

or

$$\mathbf{l}_{\mathbf{m}'} = \mathbf{F} \tilde{\mathbf{m}}', \quad (37)$$

where \mathbf{F} is given by Eq. (35). Then any point \mathbf{m} on the epipolar line of \mathbf{m}' satisfies $\tilde{\mathbf{m}}^T \mathbf{F} \tilde{\mathbf{m}}' = 0$, and this is exactly Eq. (34). Therefore, we obtain geometrically the same equation. Because of symmetry, for a given point \mathbf{m} in the first image, its corresponding epipolar line in the second image is represented by the vector $\mathbf{F}^T \tilde{\mathbf{m}}$.

Now, we show that if the images are calibrated, then the \mathbf{F} is reduced to the \mathbf{E} . Since the images are calibrated, the points \mathbf{m} can be expressed in normalized coordinates, that is, $\mathbf{m} = \mathbf{x}$. Without loss of generality, the world coordinate system is assumed to coincide with the second camera coordinate system. From the perspective projection model, we have the following camera projection matrices:

$$\mathbf{P} = [\mathbf{R} \ \mathbf{t}] \quad \text{and} \quad \mathbf{P}' = [\mathbf{I} \ \mathbf{0}].$$

This implies that $\mathbf{B} = \mathbf{R}$, $\mathbf{b} = \mathbf{t}$, $\mathbf{B}' = \mathbf{I}$, and $\mathbf{b}' = \mathbf{0}$. Plugging them into Eq. (35) gives $\mathbf{F} = [\mathbf{t}] \times \mathbf{R}$, which is exactly the essential matrix Eq. (8).

In the above derivation of the fundamental matrix, a camera perspective projection matrix \mathbf{P} is decomposed into a 3×3 matrix \mathbf{B} and a three-vector \mathbf{b} , and \mathbf{B} must be invertible. Later, we provide a more general derivation directly in terms of the camera projection matrices \mathbf{P} and \mathbf{P}' .

Fundamental Matrix and Epipolar Transformation

We examine the relationship between the fundamental matrix and the (i.e., the transformation of the epipoles and the epipolar lines between the two images).

For any point \mathbf{m}' in the second image, its epipolar line $\mathbf{l}_{\mathbf{m}'}$ in the first image is given by $\mathbf{l}'_{\mathbf{m}} = \mathbf{F} \tilde{\mathbf{m}}'$. It must go through the $\tilde{\mathbf{e}} = [e_1, e_2, e_3]^T$ and a point $\tilde{\mathbf{m}} = [u, v, s]^T$, that is, $\mathbf{l}_{\mathbf{m}'} = \tilde{\mathbf{e}} \times \tilde{\mathbf{m}} = \mathbf{F} \tilde{\mathbf{m}}'$. Here, we use the for the image points. Symmetrically, the epipolar line in the second image $\mathbf{l}'_{\mathbf{m}}$ of point \mathbf{m} is given by $\mathbf{l}'_{\mathbf{m}} = \mathbf{F}^T \tilde{\mathbf{m}}$ and must go through the epipole $\tilde{\mathbf{e}}' = [e'_1, e'_2, e'_3]^T$ and a point $\tilde{\mathbf{m}}' = [u', v', s']^T$, that is, $\mathbf{l}'_{\mathbf{m}} = \tilde{\mathbf{e}}' \times \tilde{\mathbf{m}}' = \mathbf{F}^T \tilde{\mathbf{m}}$. In other words, the epipole \mathbf{e}' is *on* the epipolar line $\mathbf{l}'_{\mathbf{m}}$ for any point \mathbf{m} ; that is,

$$\tilde{\mathbf{e}}'^T \mathbf{l}'_{\mathbf{m}} = \tilde{\mathbf{e}}'^T \mathbf{F}^T \tilde{\mathbf{m}} = 0 \quad \forall \mathbf{m},$$

which yields

$$\mathbf{F}\tilde{\mathbf{e}}' = \mathbf{0}. \quad (38)$$

Let \mathbf{c}_1 , \mathbf{c}_2 , and \mathbf{c}_3 be the column vectors of \mathbf{F} , and we have $e'_1\mathbf{c}_1 + e'_2\mathbf{c}_2 + e'_3\mathbf{c}_3 = \mathbf{0}$; thus the rank of \mathbf{F} is at most two. The solution to the epipole $\tilde{\mathbf{e}}'$ is given by

$$\begin{aligned} e'_1 &= F_{23}F_{12} - F_{22}F_{13} \\ e'_2 &= F_{13}F_{21} - F_{11}F_{23} \\ e'_3 &= F_{22}F_{11} - F_{21}F_{12}, \end{aligned} \quad (39)$$

up to, of course, a scale factor. Similarly, for the epipole in the first image, we have

$$\mathbf{F}^T \tilde{\mathbf{e}} = \mathbf{0}, \quad (40)$$

which gives

$$\begin{aligned} e_1 &= F_{32}F_{21} - F_{22}F_{31} \\ e_2 &= F_{31}F_{12} - F_{11}F_{32} \\ e_3 &= F_{22}F_{11} - F_{21}F_{12}, \end{aligned} \quad (41)$$

also up to a scale factor.

Now let us examine the relationship between the epipolar lines. Once the epipole is known, an epipolar line can be parameterized by its direction vector. Consider $\mathbf{l}'_{\mathbf{m}} = \tilde{\mathbf{e}}' \times \tilde{\mathbf{m}}'$; its direction vector \mathbf{u}' can be parameterized by one parameter τ' such that $\mathbf{u}' = [1, \tau', 0]^T$. A particular point on $\mathbf{l}'_{\mathbf{m}}$ is then given by $\tilde{\mathbf{m}}' = \tilde{\mathbf{e}}' + \lambda' \mathbf{u}'$, where λ' is a scalar. Its epipolar line in the first image is given by

$$\begin{aligned} \mathbf{l}_{\mathbf{m}'} &= \mathbf{F}\tilde{\mathbf{m}}' = \mathbf{F}\tilde{\mathbf{e}}' + \lambda' \mathbf{F}\mathbf{u}' = \lambda' \mathbf{F}\mathbf{u}' \\ &= \lambda' \begin{bmatrix} F_{11} + F_{12}\tau' \\ F_{21} + F_{22}\tau' \\ F_{31} + F_{32}\tau' \end{bmatrix}. \end{aligned} \quad (42)$$

This line can also be parameterized by its direction vector $\mathbf{u} = [1, \tau, 0]^T$ in the first image, which implies that

$$\begin{aligned} \mathbf{l}_{\mathbf{m}'} &\cong \tilde{\mathbf{e}} \times (\tilde{\mathbf{e}} + \lambda \mathbf{u}) = \lambda \tilde{\mathbf{e}} \times \mathbf{u} \\ &= \lambda \begin{bmatrix} -(F_{11}F_{22} - F_{21}F_{12})\tau \\ F_{11}F_{22} - F_{21}F_{12} \\ (F_{32}F_{21} - F_{22}F_{31})\tau - F_{31}F_{12} + F_{11}F_{32} \end{bmatrix}, \end{aligned} \quad (43)$$

where \cong means “equal” up to a scale factor and λ is a scalar. By requiring that Eq. (42) and Eq. (43) represent the same line, we have

$$\tau = \frac{a\tau' + b}{c\tau' + d}, \quad (44)$$

where

$$\begin{aligned} a &= F_{12} \\ b &= F_{11} \\ c &= -F_{22} \\ d &= -F_{21}. \end{aligned} \quad (45)$$

Writing in matrix form gives

$$\rho \begin{bmatrix} \tau \\ 1 \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} \tau' \\ 1 \end{bmatrix},$$

where ρ is a scale factor. This relation is known as the *homography* between τ and τ' , and we say that *there is a homography between the epipolar lines in the first image and those in the second image*. The above is, of course, only valid for the epipolar lines having the nonzero first element in the direction vector. If the first element is equal to zero, we should parameterize the direction vector as $[\tau, 1, 0]^T$, and similar results can be obtained.

At this point, we can see that the epipolar transformation is defined by the coordinates $\tilde{\mathbf{e}} = [e_1, e_2, e_3]^T$ and $\tilde{\mathbf{e}}' = [e'_1, e'_2, e'_3]^T$ of the epipoles and the four parameters a , b , c , and d of the homography between the two pencils of the epipolar lines. The coordinates of each epipole are defined up to a scale factor, and the parameters of the homography, as can be seen in Eq. (44), are also defined up to a scale factor. Thus, we have in total seven free parameters. This is exactly the number of parameters of the fundamental matrix.

If we have identified the parameters of the epipolar transformation, that is, the coordinates of the two epipoles and the coefficients of the homography, then we can construct the fundamental matrix, from Eq. (39), (41), and (45), as

$$\begin{aligned}
F_{11} &= be_3e'_3 \\
F_{12} &= ae_3e'_3 \\
F_{13} &= -(ae'_2 + be'_1)e_3 \\
F_{21} &= -de_3e'_3 \\
F_{22} &= -ce_3e'_3 \\
F_{23} &= (ce'_2 + de'_1)e_3 \\
F_{31} &= (de_2 - be_1)e'_3 \\
F_{32} &= (ce_2 - ae_1)e'_3 \\
F_{33} &= -(ce'_2 + de'_1)e_2 + (ae'_2 + be'_1)e_1.
\end{aligned} \tag{46}$$

The determinant $ad - bc$ of the homography is equal to the determinant of the first 2×2 submatrix of \mathbf{F} , $F_{11}F_{22} - F_{12}F_{21}$, which is zero when the epipoles are at infinity.

General Form of Epipolar Equation for Any Projection Model

In this section, we will derive a which does not assume any particular projection model.

Intersecting Two Optical Rays

The projections for the first and second cameras are represented respectively as

$$s'\tilde{\mathbf{m}} = \mathbf{P}'\tilde{\mathbf{M}}, \tag{47}$$

and

$$s\tilde{\mathbf{m}} = \mathbf{P}\tilde{\mathbf{M}}, \tag{48}$$

where $\tilde{\mathbf{m}}$ and $\tilde{\mathbf{m}'}$ are augmented image coordinates and $\tilde{\mathbf{M}}$ and $\tilde{\mathbf{M}'}$ are augmented space coordinates of a single point in the two camera coordinate systems. Here both projection matrices *do not include the extrinsic parameters*.

The same point in the two camera coordinate systems can be related by

$$\tilde{\mathbf{M}} = \mathbf{D}\tilde{\mathbf{M}'}, \tag{49}$$

where

$$\mathbf{D} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_3^T & 1 \end{bmatrix}$$

is the Euclidean transformation matrix compactly representing both rotation and translation. Now substituting Eq. (49) for (47), we have

$$s\tilde{\mathbf{m}} = \mathbf{P}\mathbf{D}\tilde{\mathbf{M}}'. \tag{50}$$

For an image point $\tilde{\mathbf{m}'}$, Eq. (48) actually defines an optical ray on which every space point $\tilde{\mathbf{M}'}$ projects onto the second image at $\tilde{\mathbf{m}'}$. This optical ray can be written in parametric form as

$$\tilde{\mathbf{M}'} = s'\mathbf{P}'^+ \tilde{\mathbf{m}'} + \mathbf{p}'^\perp, \tag{51}$$

where \mathbf{P}'^+ is the pseudoinverse matrix of \mathbf{P}' :

$$\mathbf{P}'^+ = \mathbf{P}'^T (\mathbf{P}'\mathbf{P}'^T)^{-1}, \tag{52}$$

and \mathbf{p}'^\perp is a four-vector that is perpendicular to all the row vectors of \mathbf{P}' , that is,

$$\mathbf{P}'\mathbf{p}'^\perp = \mathbf{0}. \tag{53}$$

There are an infinite number of matrices that satisfy $\mathbf{P}'\mathbf{P}'^+ = \mathbf{I}$. Thus, \mathbf{P}'^+ is not unique. See [5, 6] for how to derive this particular pseudoinverse matrix.

It remains to determine \mathbf{p}'^\perp . First note that such a vector does exist because the difference between the row dimension and column dimension is one, and the row vectors are generally independent of each other. Actually, one way to obtain \mathbf{p}'^\perp is

$$\mathbf{p}'^\perp = (\mathbf{I} - \mathbf{P}'^+ \mathbf{P}') \boldsymbol{\omega}, \tag{54}$$

where $\boldsymbol{\omega}$ is an arbitrary four-vector. To show that it is perpendicular to every row vector of \mathbf{P}' , we multiply \mathbf{P}' and \mathbf{p}'^\perp :

$$\mathbf{P}'(\mathbf{I} - \mathbf{P}'^+ \mathbf{P}') \boldsymbol{\omega} = (\mathbf{P}' - \mathbf{P}'\mathbf{P}'^T (\mathbf{P}'\mathbf{P}'^T)^{-1} \mathbf{P}') \boldsymbol{\omega} = \mathbf{0}$$

which is indeed a zero vector.

Actually, the following equation always stands, as long as the of the 3×4 matrix \mathbf{P}' is 3:

$$\mathbf{I} - \mathbf{P}'^+ \mathbf{P}' = \mathbf{I} - \mathbf{P}'^T (\mathbf{P}'\mathbf{P}'^T)^{-1} \mathbf{P}' = \frac{\mathbf{p}'^\perp \mathbf{p}'^{\perp T}}{\|\mathbf{p}'^\perp\|^2}. \tag{55}$$

The effect of matrix $\mathbf{I} - \mathbf{P}'^+ \mathbf{P}'$ is to transform an arbitrary vector to a vector that is perpendicular to every row vector of \mathbf{P}' . If \mathbf{P}' is of rank 3 (which is usually the case), then \mathbf{p}'^\perp is unique up to a scale factor.

Equation (51) is easily justified by projecting \mathbf{M}' onto the image using Eq. (48), which indeed gives $\tilde{\mathbf{m}'}$. If we look closely at the equation, we can find that \mathbf{p}'^\perp actually defines the optical center, which always

projects onto the origin, and $\mathbf{P}'^+ \tilde{\mathbf{m}}'$ defines the direction of the optical ray corresponding to image point $\tilde{\mathbf{m}}'$. For a particular value s' , Eq. (51) corresponds to a point on the optical ray defined by \mathbf{m}' .

Similarly, an image point $\tilde{\mathbf{m}}$ in the first image also defines an optical ray. Requiring the two rays to intersect in space means that a point $\tilde{\mathbf{M}}'$ corresponding to a particular s' in Eq. (51) must project onto the first image at $\tilde{\mathbf{m}}$. That is,

$$s\tilde{\mathbf{m}} = s'\mathbf{P}\mathbf{D}\mathbf{P}'^+ \tilde{\mathbf{m}}' + \mathbf{P}\mathbf{D}\mathbf{p}'^\perp, \quad (56)$$

where $\mathbf{P}\mathbf{D}\mathbf{p}'^\perp$ is the \mathbf{e} in the first image.

Performing a cross product with $\mathbf{P}\mathbf{D}\mathbf{p}'^\perp$ yields

$$s(\mathbf{P}\mathbf{D}\mathbf{p}'^\perp) \times \tilde{\mathbf{m}} = (\mathbf{P}\mathbf{D}\mathbf{p}'^\perp) \times (s'\mathbf{P}\mathbf{D}\mathbf{P}'^+ \tilde{\mathbf{m}}').$$

Eliminating s and s' by multiplying $\tilde{\mathbf{m}}^T$ from the left (equivalent to an inner product), we have

$$\tilde{\mathbf{m}}^T \mathbf{F} \tilde{\mathbf{m}}' = 0, \quad (57)$$

where

$$\mathbf{F} = [\mathbf{P}\mathbf{D}\mathbf{p}'^\perp]_\times \mathbf{P}\mathbf{D}\mathbf{P}'^+ \quad (58)$$

is the general form of fundamental matrix. It is evident that the roles that the two images play are symmetrical.

Note that Eq. (58) will be the essential matrix \mathbf{E} if \mathbf{P} and \mathbf{P}' do not include the intrinsic parameters, that is, if we work with normalized cameras.

We can also include all the intrinsic and extrinsic parameters in the two projection matrices \mathbf{P} and \mathbf{P}' , so that for a 3D point $\tilde{\mathbf{M}}'$ in a world coordinate system, we have

$$s\tilde{\mathbf{m}} = \mathbf{P}\tilde{\mathbf{M}}', \quad (59)$$

$$s'\tilde{\mathbf{m}}' = \mathbf{P}'\tilde{\mathbf{M}}'. \quad (60)$$

Similarly we get

$$s\tilde{\mathbf{m}} = s'\mathbf{P}\mathbf{P}'^+ \tilde{\mathbf{m}}' + \mathbf{P}\mathbf{p}'^\perp, \quad (61)$$

The same line of reasoning will lead to the general form of epipolar equation

$$\tilde{\mathbf{m}}^T \mathbf{F} \tilde{\mathbf{m}}' = 0,$$

where

$$\mathbf{F} = [\mathbf{P}\mathbf{p}'^\perp]_\times \mathbf{P}\mathbf{P}'^+. \quad (62)$$

It can also be shown that this expression is equivalent to Eq. (35) for the full perspective projection (see next subsection), but it is more general. Indeed, Eq. (35) assumes that the 3×3 matrix \mathbf{B}' is invertible, which is the case for full perspective projection but not for affine cameras, while Eq. (62) makes use of the pseudoinverse of the projection matrix, which is valid for both full perspective projection as well as affine cameras. Therefore the equation does not depend on any specific knowledge of projection model. Replacing the projection matrix in the equation by specific projection matrix for each specific projection model produces the epipolar equation for that specific projection model.

The Full Perspective Projection Case

Here we work with normalized cameras. Under the full perspective projection, the projection matrices for the two cameras are the same:

$$\mathbf{P}_p = \mathbf{P}'_p = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (63)$$

It is not difficult to obtain

$$\mathbf{P}_p^+ = \mathbf{P}_p^T,$$

and

$$\mathbf{p}'_p^\perp = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

Now substituting the above equations for Eq. (58), we have obtained the essential matrix

$$\mathbf{E}_p = [\mathbf{P}_p \mathbf{D} \mathbf{p}'_p^\perp]_\times \mathbf{P}_p \mathbf{D} \mathbf{P}_p^+ = [\mathbf{t}]_\times \mathbf{R}, \quad (64)$$

which is exactly the same as what we derived in the last section.

For the full perspective projection, we can prove that Eq. (62) is equivalent to Eq. (35). From definitions, we have

$$[\mathbf{B}' \quad \mathbf{b}'] \mathbf{p}'_p^\perp = \mathbf{0},$$

$$[\mathbf{B}' \quad \mathbf{b}'] \mathbf{P}'_p^+ = \mathbf{I}.$$

It is easy to show

$$\begin{aligned}\mathbf{P}\mathbf{p}'_p^\perp &= \lambda(\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}'), \\ \mathbf{P}'_p^+ &= \begin{bmatrix} \mathbf{B}'^{-1} - \mathbf{B}'^{-1}\mathbf{b}'\mathbf{q}^T \\ \mathbf{q}^T \end{bmatrix},\end{aligned}$$

where λ is a nonzero scalar and \mathbf{q} is a nonzero arbitrary three vector. Substituting them for Eq. (62) yields

$$\begin{aligned}\mathbf{F} &= \lambda[\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}'] \times (\mathbf{B}\mathbf{B}'^{-1} - (\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}')\mathbf{q}^T) \\ &= \lambda[\mathbf{b} - \mathbf{B}\mathbf{B}'^{-1}\mathbf{b}'] \times \mathbf{B}\mathbf{B}'^{-1},\end{aligned}\quad (65)$$

which completes the proof as \mathbf{F} is defined up to a scale factor.

The reader is referred to [5, 6] for the epipolar geometry between affine cameras and for a general expression of the fundamental matrix for both perspective and affine cameras. The reader is referred to [7, 8] for various algorithms of determining essential matrix and fundamental matrix from point correspondences. The reader is referred to [9, 10] for a general treatment of geometry across multiple cameras.

References

- Longuet-Higgins H (1981) A computer algorithm for reconstructing a scene from two projections. *Nature* 293:133–135
- Zhang Z, Faugeras OD (1992) 3D dynamic scene analysis: a stereo based approach. Springer, Berlin/Heidelberg
- Maybank S (1992) Theory of reconstruction from image motion. Springer, Berlin/New York
- Faugeras O (1993) Three-dimensional computer vision: a geometric viewpoint. MIT, Cambridge
- Xu G, Zhang Z (1996) Epipolar geometry in stereo, motion and object recognition. Kluwer Academic, Dordrecht/Boston
- Zhang Z, Xu G (1998) A unified theory of uncalibrated stereo for both perspective and affine cameras. *J Math Imaging Vis* 9:213–229
- Zhang Z (1997) Motion and structure from two perspective views: from essential parameters to euclidean motion via fundamental matrix. *J Opt Soc Am A* 14(11):2938–2950
- Zhang Z (1998) Determining the epipolar geometry and its uncertainty: a review. *Int J Comput Vis* 27(2):161–195
- Hartley R, Zisserman A (2000) Multiple view geometry in computer vision. Cambridge University Press, Cambridge/New York
- Faugeras O, Luong QT (2001) In: Papadopoulo T (ed) The geometry of multiple images. MIT, Cambridge/London

Error Concealment

Inpainting

Error-Correcting Graph Matching

Many-to-Many Graph Matching

Error-Tolerant Graph Matching

Many-to-Many Graph Matching

Essential Matrix

Zhengyou Zhang

Microsoft Research, Redmond, WA, USA

Related Concepts

►Epipolar Geometry; ►Fundamental Matrix

Definition

Essential matrix is a special 3×3 matrix which captures the geometric relationship between two calibrated cameras or between two locations of a single moving camera.

Background

See entry ►Epipolar Geometry for details.

Theory

Because the cameras are calibrated, we use the normalized image coordinates. The two images of a space point $\mathbf{X} = [X, Y, Z]^T$ are $[x, y, 1]^T$ and $[x', y', 1]^T$ in the first and second images and are denoted by $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$. Assume that the second camera is brought from the position of the first camera through a rotation \mathbf{R}



followed by a translation \mathbf{t} . From the epipolar geometry, we have the following equation

$$\tilde{\mathbf{x}}^T \mathbf{E} \tilde{\mathbf{x}}' = 0, \quad (1)$$

where

$$\mathbf{E} = [\mathbf{t}]_{\times} \mathbf{R}. \quad (2)$$

And $[\mathbf{t}]_{\times}$ is a 3×3 antisymmetric matrix defined by vector \mathbf{t} . This equation is called the *epipolar equation*, and matrix \mathbf{E} is known as the *essential matrix*.

Essential matrix has a number of properties, including:

1. $\det \mathbf{E} = 0$, because $[\mathbf{t}]_{\times}$ is an antisymmetric matrix.
2. $\mathbf{E}^T \mathbf{t} = \mathbf{0}$, because $([\mathbf{t}]_{\times} \mathbf{R})^T \mathbf{t} = \mathbf{R}^T [\mathbf{t}]_{\times}^T \mathbf{t} = -\mathbf{R}^T (\mathbf{t} \times \mathbf{t}) = \mathbf{0}$.
3. $\mathbf{E} \mathbf{E}^T = (\mathbf{t}' \mathbf{t}) \mathbf{I} - \mathbf{t} \mathbf{t}^T$, because $\mathbf{E} \mathbf{E}^T = ([\mathbf{t}]_{\times} \mathbf{R})([\mathbf{t}]_{\times} \mathbf{R})^T = [\mathbf{t}]_{\times} \mathbf{R} \mathbf{R}^T [\mathbf{t}]_{\times}^T = -[\mathbf{t}]_{\times}^2$.
4. $\|\mathbf{E}\|^2 = 2\|\mathbf{t}\|^2$. Here, $\|\mathbf{E}\|$ is the Frobenius norm of matrix \mathbf{E} , i.e., $\|\mathbf{E}\|^2 = \sum_{i,j} E_{ij}^2$.

It can be shown that a real 3×3 matrix can be decomposed into the multiplication of an antisymmetric matrix \mathbf{T} and a rotation matrix \mathbf{R} if and only if one of \mathbf{E} 's singular values is 0 while the other two are equal.

Euclidean Geometry

Gunnar Sparr

Centre for Mathematical Sciences, Lund University,
Lund, Sweden

Related Concepts

[►Algebraic Curve](#); [►Camera Calibration](#)

Definition

Euclidean geometry deals with properties of geometric configurations that are preserved under isometric (or *length preserving*) transformations. Alternatively, it may be characterized as a mathematical theory based on an axiom system (that can be traced back to Euclid) expressing, in modern terminology, incidence, order, congruence, continuity, and parallelity. Euclidean geometry is today a special case of many

geometric theories (projective, affine, and Riemannian geometries, Hilbert spaces ...).

Historical Background

Euclidean geometry has a long and glorious history (cf. [1–3]), having lived at the core of the development of science and culture since antiquity. It is today not an area of research per se, but still plays an important role in many contexts.

Euclidean geometry is one of the oldest manifestations of humans in science. The latter part of the word *geometry* originates from the Greek word *metri'a* for *measure*, and the subject developed in the antiquity as an empirical science for surveying. It was given a scientific formulation by the Greek mathematician Euclid in Alexandria, about 300 B.C. Starting from a small set of intuitively appealing *axioms*, in his monumental treatise, the *Elements*, he deduced a large number of *propositions* about geometrical figures (cf. [8]). In the *Elements*, plane geometry was presented in essentially the way it is today taught in secondary school. Also the solid geometry of three dimensions was addressed. Two other big contributors to ancient geometry, both active in the third century B.C., were Archimedes, with equations for, e.g., the circumference of the circle and the area of the sphere, and Apollonius, with investigations of conic sections.

For over 2,000 years, the attribute *Euclidean* was unnecessary, because no other kind of geometry was conceived of. Early, however, the fifth of Euclid's axioms, the *parallel axiom*, was met with challenge, and many unsuccessful efforts were made to deduce it from the other axioms. However, it took until the nineteenth-century before its independence was settled by the construction of consistent geometric models with other parallelity concepts. Some contributors to such non-Euclidean geometries were Gauss, Bolyai, and Lobachevsky.

By today's standard of rigor, the treatment of Euclid is not without objections, using some assumptions and concepts not explicitly accounted for. Beginning in the nineteenth-century, several categorical axiom systems were presented, e.g., by Hilbert in 1899 (cf. [6, 9]).

Also other geometric systems were discovered and developed, like affine geometry, beginning with Euler in the eighteenth-century, and projective geometry,

through Poncelet, von Staudt, a.o. in the nineteenth-century. While in affine geometry, parallelity plays a prominent role; in projective geometry, the concept does not exist, in that every pair of lines intersect.

In contrast to the then prevalent *synthetic* approach to geometry, based on geometric constructions, in the seventeenth century, Descartes introduced coordinate systems and founded the *analytic geometry*. Also de Moivre made significant contributions. The use of coordinates enabled the study of geometry by means of algebra and calculus. In this formalism, in a natural way, Euclidean geometry can be generalized to higher dimensions, by means of the notion of Euclidean space; see below.

From a more modern mathematical point of view, geometry in some space is the study of properties of configurations that are preserved under some group of one-to-one transformations of the space in question (cf. [1–3]). This viewpoint was first formulated by Klein 1872 in the so-called Erlanger program, cf. [6]. It has been the key to a fruitful interplay between geometry, algebra, analysis, and topology. For Euclidean geometry, the characterizing transformation group is the group of isometric transformations, reflecting, e.g., the crucial property of congruence, that two triangles are congruent if one of them can be moved rigidly onto the other one.

Theory and Applications

The topic of Euclidean geometry today has branched out in many directions. Below, we focus on a few of relevance to computer vision.

Synthetic Euclidean Geometry

The synthetic approach to Euclidean geometry (the one used by Euclid) starts from the basic concepts of *point*, *line*, and *surface*. The rules for interaction between these are described by *axioms*. Euclid used five axioms (cf. [8]):

1. Through any two points, there is exactly one line.
2. A finite line segment can be extended to a line.
3. A circle can be drawn with any center point and any radius.
4. All right angles are equal to another.
5. The parallel postulate: If a straight line falling on two straight lines make the interior angles on the

same side less than two right angles, the two straight lines, if produced indefinitely, meet on that side on which are the angles less than the two right angles.

For a rigorous exposition of Euclidean geometry, see [9], built on Hilbert's system of 20 axioms in five main groups: combination, order, parallelity, congruence, and continuity.

From the beginning, Euclidean geometry mostly dealt with squares and rectangles, right-angled triangles, trapezia, and circles. Some well-known examples of results from Euclidean geometry are the theorem of Pythagoras, the theorem stating that the sum of angles in a triangle equals two right angles, and the theorem stating that the periphery angle corresponding to an arch of a circle equals half the angle at the center.

The proofs in this tradition were constructive. Questions were raised about the possibility of generally solving geometric problems by ruler and compass. Using advances in algebra, the inherent impossibility of such constructions for some famed problems was proved, e.g., the trisectioning of an angle, the doubling of the cube, and the squaring of a circle.

Analytic Geometry: The Space \mathbf{R}^n

In the analytic geometry, as learned at school, points in the plane are represented by coordinates as (x_1, x_2) , and points in space as (x_1, x_2, x_3) . This inspires to consider n -tuples of real numbers (x_1, x_2, \dots, x_n) , which together build up \mathbf{R}^n . This space forms the scene for geometry, not only in 1, 2, and 3 dimensions, but also in higher dimensions. The elements are called points $X : (x_1, x_2, \dots, x_n)$.

The set \mathbf{R}^n can be provided with different structures:

Linear Space. Equip \mathbf{R}^n with the operations of addition, $\mathbf{x} + \mathbf{y} = (x_1, \dots, x_n) + (y_1, \dots, y_n) = (x_1 + y_1, \dots, x_n + y_n)$, and multiplication with a scalar $\lambda \mathbf{x} = (\lambda x_1, \dots, \lambda x_n)$. Then, in a way known from introductory courses in linear algebra, the notion of linear space is introduced. The elements of \mathbf{R}^n are called *vectors*. The notion of *dimension* is defined, giving \mathbf{R}^n the dimension n .

Affine Space. Combining the point and vector interpretations of \mathbf{R}^n , affine spaces are defined in a way such that, loosely speaking, it is possible to subtract points to get vectors and add a vector to a point to get another point. (On the contrary, it is not possible to add points.) For a thorough presentation,

see [1]. For affine spaces, the notation A^n will be used below.

Euclidean Space. Known from linear algebra is also the notion of *scalar product* on \mathbf{R}^n , being a function $\mathbf{x} \cdot \mathbf{y}$ such that for all vectors \mathbf{x} , \mathbf{y} , and all scalars λ :

- $\mathbf{x} \cdot \mathbf{y} = \mathbf{y} \cdot \mathbf{x}$.
- $\mathbf{x} \cdot (\lambda_1 \mathbf{y}_1 + \lambda_2 \mathbf{y}_2) = \lambda_1 \mathbf{x} \cdot \mathbf{y}_1 + \lambda_2 \mathbf{x} \cdot \mathbf{y}_2$.
- $\mathbf{x} \cdot \mathbf{x} \geq 0$ with equality if and only if $\mathbf{x} = 0$.

Given a scalar product, a *norm* on the linear space \mathbf{R}^n is defined by $\|\mathbf{x}\| = \sqrt{\mathbf{x} \cdot \mathbf{x}}$. Two vectors \mathbf{x} and \mathbf{y} are orthogonal if $\mathbf{x} \cdot \mathbf{y} = 0$. An orthonormal basis e_1, \dots, e_n is characterized by $e_i \cdot e_j = 0$ for $i \neq j$ and $= 1$ for $i = j$.

Having a scalar product, the affine space A^n can be provided with a *metric*, by which the distance between points is given by $d(X, Y) = \|\mathbf{u}\|$, where \mathbf{u} is the vector from X to Y . In particular, in an orthonormal basis, the distance between the points $X : (x_1, \dots, x_n)$ and $Y = (y_1, \dots, y_n)$ is

$$|XY| = ((x_1 - y_1)^2 + \dots + (x_n - y_n)^2)^{1/2}$$

(in agreement with the theorem of Pythagoras in the planar case).

The scalar product also makes it possible to define the *angle* θ between two vectors \mathbf{x} and \mathbf{y} as

$$\mathbf{x} \cdot \mathbf{y} = \|\mathbf{x}\| \|\mathbf{y}\| \cos \theta.$$

Provided with a scalar product for vectors, A^n becomes a Euclidean space, below denoted E^n .

Euclidean Geometry and Transformation Groups

Given a set S , consider the group $\text{Bij}(S)$ consisting of all one-to-one transformations $f : S \rightarrow S$. According to the view of the Erlanger program, to impose a geometry on S is the same as to specify a subgroup G of $\text{Bij}(S)$ and to say that two subsets A , B of S are equivalent if there is an $f \in G$ such that $fA = B$. See [1–3].

Two important such transformation groups on \mathbf{R}^n are $\text{GL}(n)$, consisting of all non-singular $n \times n$ -matrices, and $\text{O}(n)$, consisting of all orthogonal $n \times n$ -matrices.

For planar Euclidean geometry, S is the affine space A^2 and G is the group of all *isometric transformations*, i.e., transformations such that for any points

$X, Y \in \mathbf{R}^2$, $d(TX, TY) = d(X, Y)$. In fact, it can be proven that every mapping T with this property can be written as

$$T : X \rightarrow QX + b, \text{ with } Q \in \text{O}(n).$$

In particular, T is an affine map, mapping lines onto lines. Thus, it also maps triangles onto triangles, after which the isometric property guarantees congruence. In higher dimensions holds the analogous formula for T .

In Euclidean geometry, also the notion of *similarity* plays a prominent role, which could motivate the use of similarity transformations above (where Q is replaced by cQ , $c \neq 0$). Note that the isometric transformations form a subgroup of the similarity transformations.

The example of planar Euclidean geometry is so crucial that it is worthwhile to comment further on the structure of its transformation group. The formula above shows that an isometric transformation is composed by a rotation around the origin (expressed by Q), followed by a translation (by b). Furthermore, it is possible to prove that any such transformation is equal to either the identity, or a translation, or a rotation (around some point), or a reflection in some line, or a so-called glide reflection (a translation followed by a reflection in a line parallel to the translation). Also in \mathbf{R}^3 , it is possible to make an analogous characterization, where a type screw (composition of a rotation and a translation) is added. Note in particular that in this way, isometries on \mathbf{R}^2 and \mathbf{R}^3 are characterized by geometric constructions only, without use of any particular coordinate system.

On the Stratum of Euclidean, Affine and Projective Geometry

Euclidean geometry is one of the layers in a hierarchy of structures that can be imposed on \mathbf{R}^n and which are of high relevance to computer vision. With a term introduced by Faugeras [5], these form a *geometric stratum*. For more details on the respective geometries, see [1–3, 10].

Affine Geometry. For the linear space \mathbf{R}^n , a particular role is played by linear subspaces of the form $a_1x_1 + \dots + a_nx_n = 0$, called *hyperplanes* (always passing through the origin). As a linear space, its dimension is $n-1$. Analogously, in the affine space A^n , one considers *affine hyperplanes* $a_1x_1 + \dots +$

$a_n x_n = a$ (which for $a \neq 0$ do not pass through the origin). In the case $n = 2$, one talks about *lines* instead of hyperplanes.

For affine geometry on \mathbf{R}^n , the characterizing transformations are

$$T : X \rightarrow AX + b, \text{ with } A \in \mathrm{GL}(n).$$

By such transformations, affine hyperplanes are mapped onto affine hyperplanes. Two nonintersecting hyperplanes are mapped onto two nonintersecting hyperplanes. Hence, the property of parallelity is preserved and is thus a concept within affine geometry. On the contrary, distance is not preserved and is an alien concept for affine geometry. However, it is possible to compare distances along a line (e.g., saying that M is the midpoint of a line segment PQ), but it is not possible to measure and compare distances on nonparallel lines. Neither does the concept of angle live in affine geometry, in lack of a scalar product.

Also affine geometry can be built up from axioms. Compared to Euclidean geometry, one way of doing this is by replacing the parallel axiom by an axiom “for any point P and any line ℓ , not through P , there is at least one line through P which does not meet ℓ ,” plus another one, involving seven-point configurations, named after Desargues.

Projective Geometry. By a point in the so-called projective space \mathbb{P}^n is meant a line through the origin in \mathbb{R}^{n+1} . In other words, a point is represented by any of the vectors $\lambda(x_1, x_2, \dots, x_{n+1})$, $\lambda \neq 0$. This is called *homogeneous coordinates* for the point in \mathbb{P}^n . By a *projective transformation* T is meant a mapping represented by $\mathrm{GL}(n+1)$ in homogeneous coordinates, i.e., $y = \lambda T x$ for some λ .

Looking in particular at plane projective geometry, where points are represented by lines through the origin in \mathbb{R}^3 , a *projective line* is represented by a plane through the origin in \mathbb{R}^3 . Since every pair of such planes intersect in a line through the origin, i.e., a point in \mathbb{P}^2 , the notion of parallelity does not exist in projective geometry.

The projective space \mathbb{P}^n , embedded in \mathbb{R}^{n+1} , can also be visualized by means of an affine hyperplane in \mathbb{R}^{n+1} , e.g., $\Pi : x_{n+1} = 1$. Every line not parallel to Π intersects Π in a unique point. Besides these, \mathbb{P}^n contains points corresponding to lines parallel to Π .

The latter points of \mathbb{P}^n are called *points at infinity* and may be identified with directions in Π . The set of points at infinity is called the *line at infinity*. To sum up, the projective space \mathbb{P}^n can be visualized by a model formed by an n -dimensional affine space extended with points at infinity.

Perspective Transformations: Multiple View Geometry

For computer vision, *perspective transformations* play a crucial role. Specializing to three dimensions, consider two affine 3-dimensional hyperplanes Π and Π' in \mathbb{R}^4 . If C is a point outside Π and Π' , a mapping $P : \Pi \longrightarrow \Pi'$ is defined by considering the intersections of lines through C with Π and Π' . Then C is called the *focal point* of the *perspective transformation* P . In particular, one notes that if Π and Π' is nonparallel, then point at infinity of Π is mapped onto *ordinary* points of Π' and vice versa. If Π and Π' are parallel, then points at infinity are mapped onto points at infinity, and the perspective transformation is an affine transformation.

To sum up, Euclidean geometry is a special case of affine geometry, as follows from the fact that the group of isometries is a subgroup of the affine group. Affine geometry is a special case of projective geometry, since the affine transformations form a subgroup of the projective ones, characterized by leaving the line at infinite invariant.

Of particular importance to computer vision is the case of perspective transformations from three dimensions to two. These are singular, i.e., not one to one, contrary to the ones discussed above. They may be visualized by letting C tend to the plane Π' , which in the limit gives a mapping $\Pi \longrightarrow \Pi \cap \Pi'$, from three to two dimensions. Moreover, the ambient space \mathbb{R}^4 becomes superfluous and can be replaced by the three-dimensional affine space Π .

Multiple view geometry deals with the situation where a number of such two-dimensional perspective images are known of a common three-dimensional object (cf. [4, 7]). With no further information, reconstruction is possible up to projective transformations. Having more geometric structure available, e.g., Euclidean information on the image planes and focal points, more structure can be obtained for the reconstruction, ideally making it Euclidean. This situation is termed *camera calibration*; see [7].



Some Other Geometries Embracing Euclidean Geometry

Algebraic Geometry. Besides hyperplanes, crucial roles in the study of Euclidean (as well as projective and affine) geometry are played by *conics*. While hyperplanes have equations involving first-order polynomials, conics are described by second-order polynomials. In two dimensions, this leads to the analytic geometry of conic sections, representing ellipses, hyperbolas, and parabolas. Algebraic geometry is the generalization of this to higher dimensions and higher-order polynomials, where Euclidean geometry thus falls out as a special case.

Riemannian geometry. Riemann geometry lives on a so-called Riemannian manifold, which, loosely speaking, is a space constructed by deforming and patching together Euclidean spaces according to certain rules, guaranteeing, e.g., *smoothness*. Such a space enjoys notions of distance and angle but behaves in a curved, non-Euclidean manner. The simplest Riemannian manifold consists of \mathbb{R}^n with a constant scalar product, leading to classical Euclidean geometry. Riemannian geometry plays a prominent role in general relativity.

Hilbert Spaces. Considering infinite sequences (x_1, x_2, \dots) instead of finite ones, (x_1, x_2, \dots, x_n) , will lead to convergence problems whenever forming sums. Restricting oneself to sequences with $\sum_1^\infty x_i^2$ finite, a prototype of so-called Hilbert spaces is obtained. In a natural way, a scalar product is defined, leading to a distance measure, by means of which it is possible to prove, e.g., the infinite-dimensional analogue of the theorem of Pythagoras and many other theorems of Euclidean geometry.

References

1. Berger M (1987) Geometry I and II. Springer Universitext. Springer, Berlin/Heidelberg/New York/London/Paris/Tokyo
2. Coxeter HSM (1989) Introduction to geometry, 2nd edn. Wiley Classics Library, Wiley/New York
3. Eves H (1972) A survey of geometry. Allyn and Bacon, Boston
4. Faugeras O (1993) Three-dimensional computer vision. A geometric viewpoint. MIT, Cambridge/London
5. Faugeras O (1995) Stratification of three-dimensional vision: projective, affine, and metric representations. J Opt Soc Am A 12:465–484
6. Greenberg MJ (2008) Euclidean and non-Euclidean geometries: Development and history. W.H. Freeman
7. Hartley R, Zisserman A (2003) Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, Cambridge
8. Heath TL (1956) The thirteen books of Euclid's elements, 3 vols. Dover, New York
9. Hilbert D, Cohn-Vossen S (1952) Geometry and the imagination. Chelsea, New York
10. Veblen O, Young JW (1910–1918) Projective geometry I and II. Ginn and Co., Boston

Evolution of Robotic Heads

Michael R. M. Jenkin

Department of Computer Science and Engineering,
York University, Toronto, ON, Canada

Related Concepts

► Camera Calibration

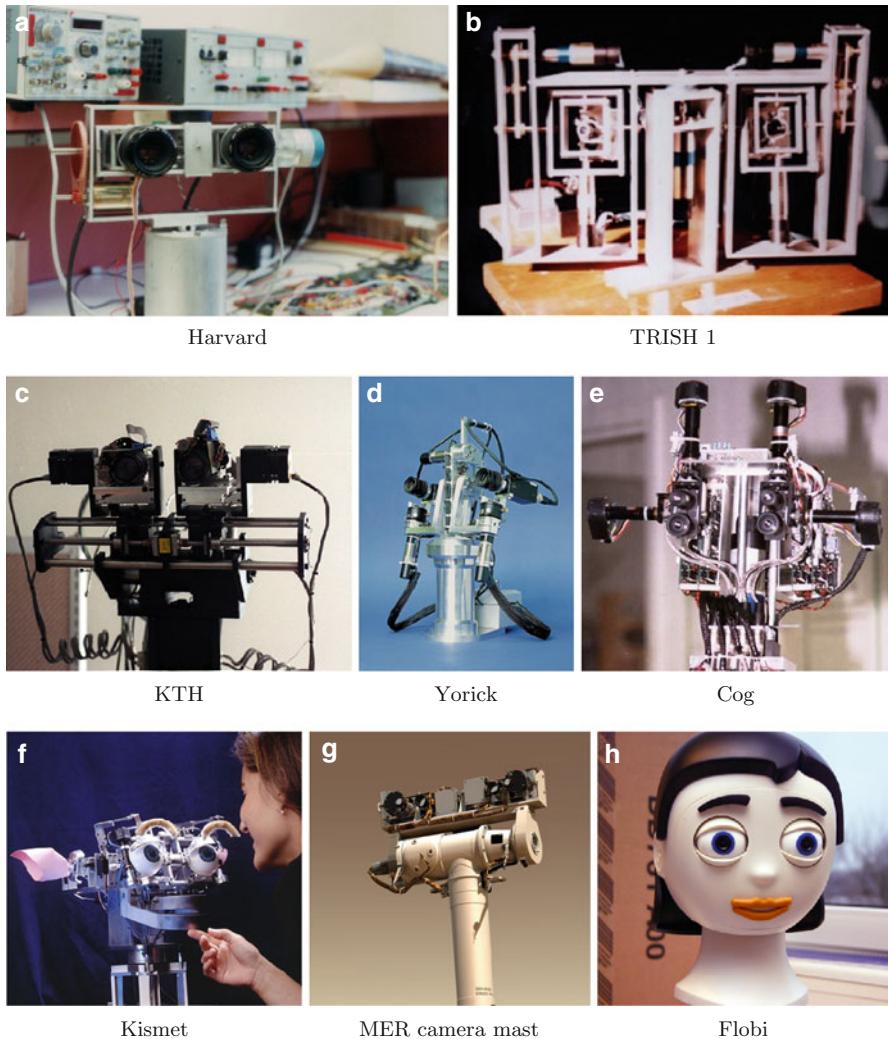
Definition

Robotic heads are actively controlled camera platforms, typically designed to mimic the head and camera (eye) motions associated with humans. Early designs were built to study the role of eye and head motion in active vision systems [1]. Later designs are also used in the study of human-robot interaction.

Background

Cameras have a finite field of view and thus must be actively controlled in order to bring out of view portions of the scene into view and to track dynamic scene events. The need for active control of camera geometry becomes even more critical when multiple cameras are involved as changes in relative camera geometry can simplify stereo image processing and can be used to bring specific scene features within the tuning range of multiple-camera scene reconstruction algorithms.

The development of single camera robotic heads can be traced back to the Stanford Cart [2]. Binocular systems began to appear in the late 1980s and early 1990s (e.g., [3–7]). By the late 1990s, head designs were beginning to be driven by research interest in human-robot interaction (e.g., [8, 9]). Current head designs are typically designed to be anthropomorphic



Evolution of Robotic Heads, Fig. 1 The evolution of robotic heads. (a) Appears with the kind permission of J. Clark (b) and (c) Appear courtesy of M. Jenkin. (d) Appears with the

kind permission of P. Sharkey. (e) Appears courtesy NAS/JPL-Caltech. (f) Appears with the kind permission of Blefeld University

(e.g., [10]) or to meet specific requirements of the sensors or the application (e.g., [11]). See Fig. 1 for examples.

Theory

Although there is a wide range of different head designs, stereo robotic heads are perhaps the most common. Such systems either have fixed relative geometry between the two cameras or the relative geometry is controllable. Individual cameras may also be equipped with controllable intrinsic

camera settings. Heads are typically mounted on pan and tilt “necks” that drive the entire head to look at different portions of the scene. Controllable parameters can include:

- *Head pan*. Pan angle introduced by a robotic neck.
- *Head tilt*. Tilt angle introduced by a robotic neck.
- *Baseline*. The displacement between the left and right cameras.
- *Fixation point*. The point at which the optical axes of the left and right cameras intersect. The fixation point can be defined in various ways including the individual pan directions of the left and right cameras and the vergence/version angles.

- *Cyclotorsion*. The roll of each camera about its optical axis.
- *Zoom/focus/aperture*. Intrinsic settings of each camera.

Being able to control the relative pose of the two cameras provides the sensor the ability to bring different portions of space into alignment so that objects appear at the same position in both cameras. As stereo image processing is typically performed only over a limited range of disparities (image differences), changing the relative geometry between the cameras leads to the camera system attending to different regions of space. The region of space that is brought into alignment is known as the horopter (see [12] for a review of the geometry).

Early designs were strongly constrained by the mass of available camera housings and especially the mass of controllable focus/zoom lenses. This is particularly evident in the early designs shown in Fig. 1. More recent designs have taken advantage of the decreased size and mass of camera systems.

Many modern robotic heads have been developed to help study aspects of human-robot interaction. Such “social robots” are designed to not only use multiple sensors to reconstruct 3D environments but they often also provide a range of actuators to simulate human facial responses.

References

1. Ballard DH (1991) Animate vision. *Artif Intell J* 48:57–86
2. Moravec HP (1983) The Stanford Cart and the CMU rover. *Proc IEEE* 71:872–884
3. Krotkov E (1989) Active computer vision by cooperative focus and stereo. Springer, New York
4. Ferrier NJ, Clark JJ (1993) The Harvard binocular head. *Int J Pattern Recognit Artif Intell* 7(1):9–32
5. Milius E, Jenkin M, Tsotsos J (1993) Design and performance of TRISH, a binocular robot head with torsional eye movements. *Int J Pattern Recognit Artif Intell* 7(1): 51–68
6. Madden BC, von Seelen U (1995) PennEyes: a binocular active vision system. Technical report MS-CIS-95-37, University of Pennsylvania
7. Sharkey PM, Murray DW, McLauchlan PF, Brooker JP (1998) Hardware development of the Yorick series of active vision systems. *Microprocess Microsyst* 21:363–375
8. Scassellati B (1998) Eye finding via face detection for a foveated, active vision system. In: Proceedings of the AAAI-98, Madison, pp 969–976
9. Breazeal C, Scassellati B (1999) How to build robots that make friends and influence people. In: IEEE/RSJ

- international conference on robots and autonomous systems, Kyongju
- 10. Lütkebohle I, Hegel F, Schulz S, Hackel M, Wrede B, Wachsmuth S, Sagerer G (2010) The Bielefeld anthropomorphic robot head “Flobi”. In: IEEE international conference on robotics and automation, Anchorage
- 11. Goldberg SB, Maimone MW, Matthies L (2002) Stereo vision and rover navigation software for planetary exploration. In: 2002 aerospace conference, Big Sky
- 12. Hansard M, Horaud R (2008) Cyclopean geometry of binocular vision. *J Opt Soc Am A* 25(9):2357–2369

Expectation Maximization Algorithm

Boris Flach and Vaclav Hlavac

Department of Cybernetics, Czech Technical University in Prague, Faculty of Electrical Engineering, Prague 6, Czech Republic

Synonyms

[EM-algorithm](#)

Related Concepts

► [Maximum Likelihood Estimation](#)

Definition

The Expectation Maximization algorithm iteratively maximizes the likelihood of a training sample with respect to unknown parameters of a probability model under the condition of missing information. The training sample is assumed to represent a set of independent realizations of a random variable defined on the underlying probability space.

Background

One of the main paradigms of statistical pattern recognition and Bayesian inference is to model the relation between the observable features $x \in \mathcal{X}$ of an object and its hidden state $y \in \mathcal{Y}$ by a joint probability measure $p(x, y)$. This probability measure is often known only up to some parameters $\theta \in \Theta$. It is thus necessary

to estimate these parameters from a training sample, which is assumed to represent a sequence of independent realizations of a random variable. If, ideally, these are realizations of pairs (x, y) , then the corresponding estimation methods are addressed as *supervised* learning. It is, however, quite common that some of those variables describing the hidden state are latent. These latent variables are never observed in the training data. Therefore, it is necessary to marginalize over them in order to estimate the unknown parameters θ . Corresponding estimation methods are known as *unsupervised* learning. Moreover, especially in computer vision, the observation x and the hidden state y both may have a complex structure. The latter can be, e.g., a segmentation, a depth map, or a similar object. Consequently, it is often not feasible to provide the complete information y for the realizations in the sample. This means to estimate the parameters in the situation of missing information. The EM algorithm is a method searching for maximum likelihood estimates of the unknown parameters under such conditions.

Theory

All the situations described in the previous section can be treated in a uniform way by assuming the training sample as a set of independent realizations of a random variable.

Let Ω be a finite sample space, \mathcal{F} be its power set, and $p_\theta: \mathcal{F} \rightarrow \mathbb{R}_+$ be a probability measure defined up to unknown parameters $\theta \in \Theta$. Let $X: \Omega \rightarrow \mathcal{X}$ be a random variable and $T = (x_1, x_2, \dots, x_n)$ be a sequence of independent realizations of X (see, e.g., [1, 2] for a formal definition of independent realizations). The maximum likelihood estimator provides estimates of the unknown parameters θ by maximizing the probability of T :

$$\theta^* = \operatorname{argmax}_{\theta} \prod_{i=1}^n p_\theta(\Omega_i), \quad (1)$$

where Ω_i denotes the pre-image $\{\omega \in \Omega \mid X(\omega) = x_i\}$. If the logarithm is taken, the task reads equivalently:

$$\theta^* = \operatorname{argmax}_{\theta} L(x_1, \dots, x_n, \theta)$$

$$= \operatorname{argmax}_{\theta} \sum_{i=1}^n \log \sum_{\omega \in \Omega_i} p_\theta(\omega). \quad (2)$$

Remark 1. It is often assumed that Ω is a Cartesian product $\Omega = \mathcal{X} \times \mathcal{Y}$ and that $X(x, y) = x$. Then the probability $p_\theta(\Omega_i) = \sum_{y \in \mathcal{Y}} p_\theta(x_i, y)$ is nothing but the marginalization over all possible y . This special case will be considered in an example below. \square

The optimization task (Eq. 2) is often complicated and hardly solvable by standard optimization methods – either because the objective function is not concave or because θ represents a set of parameters of different natures. Suppose, however, that the task of parameter estimation is feasible if complete information, i.e., a set of realizations of $\omega \in \Omega$, is available. This applies in particular if the corresponding simpler objective function $\sum_i \log p_\theta(\omega_i)$ is concave with respect to θ or if the task decomposes into simpler, independent optimization tasks with respect to individual components of a parameter collection.

The key idea of the Expectation Maximization algorithm is to exploit this circumstance and to solve the optimization task (Eq. 2) by iterating the following two feasible tasks:

1. Given a current estimate of θ , determine the missing information, i.e., $p_\theta(\omega | \Omega_i)$, for each element $x_i \in T$.
2. Given the complete information, solve the corresponding estimation task, resulting in an improved estimate of θ .

To further substantiate this idea of “iterative splitting” of task (Eq. 2), it is convenient to introduce nonnegative auxiliary variables $\alpha_i(\omega)$, $\omega \in \Omega_i$, for each element x_i of the learning sample T such that they fulfil:

$$\sum_{\omega \in \Omega_i} \alpha_i(\omega) = 1, \quad \forall i = 1, 2, \dots, n. \quad (3)$$

These variables α_i can be seen as (so far arbitrary) posterior probabilities $p(\omega | \Omega_i)$ for $\omega \in \Omega_i$, given a realization x_i . The log-likelihood of a realization x_i can be written by their use as:

$$\log p_\theta(\Omega_i) = \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\theta(\omega)$$

$$= \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\theta(\omega) - \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log \frac{p_\theta(\omega)}{p_\theta(\Omega_i)}, \quad (4)$$

where the first equality follows directly from (Eq. 3). The log-likelihood of the training sample can be therefore expressed equivalently as:

$$\begin{aligned} L(x_1, \dots, x_n, \theta) &= \sum_{i=1}^n \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\theta(\Omega_i) \\ &= \sum_{i=1}^n \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\theta(\omega) - \sum_{i=1}^n \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\theta(\omega | \Omega_i). \end{aligned} \quad (5)$$

The expression as a whole does not depend on the specific choice of the auxiliary variables α , whereas the minuend and subtrahend do. Moreover, note that the minuend is nothing but the likelihood of a sample of complete data, if the α are interpreted as the missing information, i.e., posterior probabilities for $\omega \in \Omega_i$ given the observation x_i .

Starting with some reasonable choice for the initial $\theta^{(0)}$ the likelihood is iteratively increased by alternating the following two steps. The (E)xpectation step calculates new α such that whatever new θ will be chosen subsequently, the subtrahend will not increase. The (M)aximization step relies on this and maximizes the minuend only, avoiding to deal with the subtrahend:

$$\text{E-step } \alpha_i^{(t)}(\omega) = p_{\theta^{(t)}}(\omega | \Omega_i) \quad (6)$$

$$\text{M-step } \theta^{(t+1)} = \operatorname{argmax}_\theta \sum_{i=1}^n \sum_{\omega \in \Omega_i} \alpha_i^{(t)}(\omega) \log p_\theta(\omega). \quad (7)$$

From the conceptual point of view the E-step can be seen as inference – it calculates the missing data, i.e., the posterior probabilities $p_{\theta^{(t)}}(\omega | \Omega_i)$ for each element x_i in the training sample. The M-step utilizes these posterior probabilities for a supervised learning step. The names themselves stem from a rather formal view: the E-step calculates the α and therefore

the objective function in (Eq. 7) which has the form of an expectation of $\log p_\theta(\omega)$. The computation of this objective function is sometimes considered to be a part of the E-step. The name for the M-step is obvious.

It is easy to see that the likelihood is monotonically increasing: The choice (Eq. 6) for α guarantees that the subtrahend in (Eq. 5) can only decrease whatever new θ will be chosen in the subsequent M-step. This follows from the inequality:

$$\begin{aligned} &\sum_{\omega \in \Omega_i} p_\theta(\omega | \Omega_i) \log p_{\theta'}(\omega | \Omega_i) \\ &\leq \sum_{\omega \in \Omega_i} p_\theta(\omega | \Omega_i) \log p_\theta(\omega | \Omega_i) \quad \forall \theta' \neq \theta. \end{aligned} \quad (8)$$

Since the M-step chooses the new θ so as to maximize the minuend, the likelihood can only increase (or stay constant). Another convenient way to prove monotonicity of the EM algorithm can be found in [3, 4]. These tutorials consider the EM algorithm as the maximization of a lower bound of the likelihood.

It remains unclear whether the global optimum of the likelihood is reached in a fix-point of the algorithm. Moreover, it happens quite often that the M-step is infeasible for complex models p_θ . Then a weaker form of the EM algorithm is used by choosing $\theta^{(t+1)}$ so as to guarantee an increase of the objective function of the M-step.

The derivation of the concept of the EM algorithm was given here for a discrete probability space and discrete random variables. It can be however generalized for uncountable probability spaces and random variables X with continuous probability density.

Example 1. The EM algorithm is often considered for the following special case. The sampling space Ω is a Cartesian product $\Omega = \mathcal{X} \times \mathcal{Y}$ and the random variable X simply projects onto the first component $X(x, y) = x$. The parameters $\theta \in \Theta$ of the probability $p_\theta(x, y)$ are to be estimated given a sequence of independent realizations of x . In this special case, the log-likelihood has the form:

$$L = \sum_{i=1}^n \log \sum_{y \in \mathcal{Y}} p_\theta(x_i, y). \quad (9)$$

Its decomposition (Eq. 5) is:

$$L = \sum_{i=1}^n \sum_{y \in \mathcal{Y}} \alpha_i(y) \log p_\theta(x_i, y) - \sum_{i=1}^n \sum_{y \in \mathcal{Y}} \alpha_i(y) \log p_\theta(y|x_i). \quad (10)$$

The EM algorithm itself then reads:

$$\text{E-step } \alpha_i^{(t)}(y) = p_{\theta^{(t)}}(y|x_i) \quad (11)$$

$$\text{M-step } \theta^{(t+1)} = \operatorname{argmax}_\theta \sum_{i=1}^n \sum_{y \in \mathcal{Y}} \alpha_i^{(t)}(y) \log p_\theta(x_i, y). \quad (12)$$

History and Applications

The classic paper [5] is often cited as the first one introducing the EM algorithm in its general form. It should be noted, however, that the method was introduced and analyzed substantially earlier for a broad class of pattern recognition tasks in [6] and for exponential families in [7].

A comprehensive discussion of the EM algorithm can be found in [8] and in the context of pattern recognition in [9, 10]. Standard application examples are parameter estimation for mixtures of Gaussians [8] and the Mean Shift algorithm [11]. Another important application is parameter estimation for Hidden Markov Models. This model class is extensively used for automated speech recognition. The corresponding EM algorithm is known as Baum-Welch algorithm in this context [12]. Rather complex applications of the EM algorithm arise in the context of parameter estimation for Markov Random Fields [13].

References

1. Meintrup D, Schäffler S (2005) Stochastik. Springer, Berlin
2. Papoulis A (1990) Probability and Statistics. Prentice-Hall, Englewood Cliffs
3. Minka T (1998) Expectation-maximization as lower bound maximization. Tutorial, MIT, Cambridge, MA
4. Dellaert F (2002) The expectation maximization algorithm. Technical report GIT-GVU-02-20, Georgia Institute of Technology
5. Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc Ser B 39(1):1–38
6. Schlesinger MI (1968) The interaction of learning and self-organization in pattern recognition. Kibernetika 4(2):81–88
7. Sundberg R (1974) Maximum likelihood theory for incomplete data from an exponential family. Scand J Stat 1(2): 49–58
8. McLachlan GJ, Krishnan T (1997) The EM algorithm and extensions. Wiley, New York
9. Schlesinger MI, Hlavac V (2002) Ten lectures on statistical and structural pattern recognition. Kluwer Academic Publishers, Dordrecht
10. Bishop CM (2006) Pattern recognition and machine learning. Springer, New York
11. Cheng Y (1995) Mean shift, mode seeking, and clustering. IEEE Trans Pattern Anal Mach Intell 17(8):790–799
12. Jelinek F (1998) Statistical methods for speech recognition. MIT, Cambridge, MA
13. Li SZ (2009) Markov random field modeling in image analysis. 3rd edn. Advances in pattern recognition. Springer, London

Exploration: Simultaneous Localization and Mapping (SLAM)

Samunda Perera¹, Nick Barnes² and Alexander Zelinsky³

¹Canberra Research Laboratory, NICTA, Canberra, Australia

²Australian National University, Canberra, Australia

³CSIRO, Information Sciences, Canberra, Australia

Synonyms

Concurrent mapping and localization (CML); Visual SLAM

Related Concepts

► [Structure-from-Motion \(SfM\)](#)

Definition

Exploration refers to gathering data about an environment through sensors in order to discover its structure. A fundamental technique for exploration of an unknown environment is Simultaneous Localization and Mapping (SLAM). SLAM is a technique that supports the incremental building of a 3-D map representation of an environment while also using the same incremental map to accurately localize the observer in order to minimize errors in the map building.

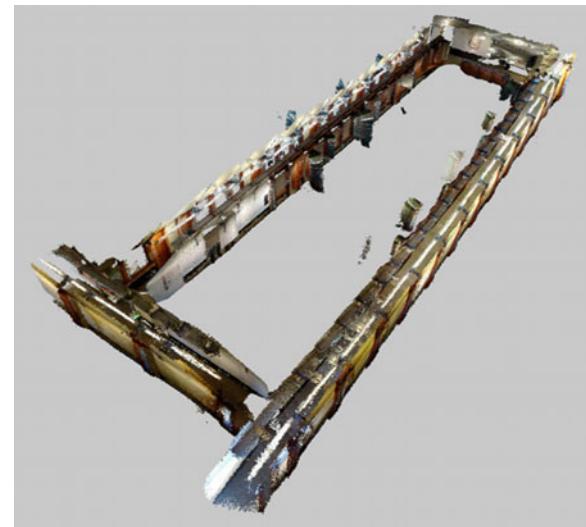
SLAM techniques can be implemented with laser range finders, monocular vision, stereo vision, and RGB-D cameras.

Background

An accurate representation of an environment is highly useful for autonomous and human assistive applications. Such a representation can be a high level minimalistic map derived from a sparse set of landmarks or a detailed dense 3-D model (e.g., Fig. 1). An accurate map can be used to perform useful tasks such as answering the question, “Where am I ?” and enables navigation and guiding tasks such as “How to get there?.” When dealing with unknown environments, the map representation must be acquired through an exploration phase. With SLAM the map can be built through a data gathering or map building phase, by either online or off-line methods. In online methods, the map is incrementally built, and location within the map is simultaneously calculated. With off-line methods the data is collected and then post-processed to yield an accurate map representation.

The key process of SLAM is an optimization algorithm. Noisy observations are made about the landmarks from different viewpoints (poses), and the task is to solve the optimization problem of estimating the position of each landmark and the set of camera viewpoints which best explain the observations. In off-line mapping this is a global optimization problem where all observations are available (e.g., Bundle Adjustment (BA) [2] applied to a full image dataset). In online mapping this is performed sequentially, using observations available up to and including the current observation. Thus, in online mapping, an incremental version of the map is available as the map is being constructed.

Two different methods are used in online mapping, namely, filtering-based and keyframe-based methods. Filtering-based methods are based on the following nature of sequential mapping. As camera observations/measurements are noisy, there is induced error in the newly identified landmarks. Later when one localizes with respect to the current map, again there is induced error in this new pose estimate due to error in observation and error in these new landmarks. So in subsequent new landmark additions to the map, there will be error in landmarks due to both measurement errors and the camera viewpoint estimation error.



Exploration: Simultaneous Localization and Mapping (SLAM), Fig. 1 A 3-D map generated by RGB-D Mapping by Henry et al. [1] (Copyright © 2012 SAGE. Reprinted by Permission of SAGE)

Due to the common error associated with camera viewpoint, estimates of the landmarks are all necessarily correlated with each other [3]. Therefore a consistent full solution to the combined localization and mapping problem requires a joint state composed of the camera pose and every landmark position, to be updated following each landmark observation and hence the name Simultaneous Localization And Mapping.

In contrast keyframe-based methods do not propagate a probability distribution over time but employ a subset of image frames, called keyframes, in the map to perform efficient global optimization of the map and keyframe poses within a practical time limit. Although non keyframes are discarded in the optimization, it permits a large number of image observations per keyframe to be efficiently processed.

Visual SLAM is possible with both monocular and stereo cameras, and these systems are termed monoSLAM and stereoSLAM, respectively. In addition, RGB-D Mapping [1] which generates a dense 3-D map with RGB-D cameras has recently been introduced. It should be noted that, despite the computational effort, the use of cameras provides several unique advantages over other sensor types such as laser rangefinders and sonars. They are inexpensive, low power, compact, and capture scene information up to very large distances.

Theory

A complete SLAM system performs following operations in a cycle. First a set of new landmarks are identified in the environment, initialized (*new landmark/feature initialization*), and inserted into the map. Upon moving to a new location, *observations* are made and matched with the map landmarks, and this is referred to as *data association*. Here, *relocalization* methods are used to recover from possible tracking failures. Next, a prior estimate of the observer motion is obtained (*motion estimation*). Then the estimates of landmark positions and sensor pose that best explains the data are sought (*optimization*). If reobserving a previously mapped portion of the environment, care should be taken to ensure correct closure of the trajectory loop and consistency of the map (*loop closure*). [Algorithm 1](#) illustrates this process (Note that as online mapping is an incremental process, [Algorithm 1](#) assumes for generality some initial map is available. This may be the result of a single observation and new landmark initialization from the starting position).

```

Input: Initial set of landmarks and initial state of observer
Output: SLAM Map and observer state at each time instant
repeat
    (1) Observation
    (2) Data association
    if tracking failure detected then
        (3) Relocalization
        (4) Motion estimation
        (5) Optimization
        if loop closure detected then
            (6) Loop closure correction
        (7) New landmark initialization

```

Algorithm 1: SLAM operation

An example situation is illustrated by [Fig. 2](#). At time k , the true state of the observer is given by \mathbf{x}_k . The observer makes observations $\mathbf{z}_{k,i}, \mathbf{z}_{k,j}, \mathbf{z}_{k,p}$ of static landmarks whose true states are given by $\mathbf{m}_i, \mathbf{m}_j$, and \mathbf{m}_p , respectively. In the landmark selection process, it also discovers a new landmark whose true state is given by \mathbf{m}_q . Based on the corresponding measurement $\mathbf{z}_{k,q}$, the state of this new landmark is then initialized and inserted into the map. At time $k+1$, the observer has moved to a state \mathbf{x}_{k+1} under the control input \mathbf{u}_{k+1} . The observer has been able to reobserve two existing landmarks in the map, namely, \mathbf{m}_p and \mathbf{m}_q . Both the corresponding measurements

$\mathbf{z}_{k+1,p}, \mathbf{z}_{k+1,q}$ and control input \mathbf{u}_{k+1} can be utilized to obtain a prior estimate of the incremental motion of the observer between time k and $k+1$.

At time $k+n$, the observer has travelled along a long trajectory loop and is revisiting a previously mapped part of the environment, namely, landmarks \mathbf{m}_i and \mathbf{m}_j . However, as can be seen from [Fig. 2](#), the estimated state of the observer might not show a closed loop. Loop closure methods detect this situation and correct the state of the observer and the states of landmarks in the map.

The following subsections details the key stages of the SLAM system operation as given in [Algorithm 1](#).

Observation

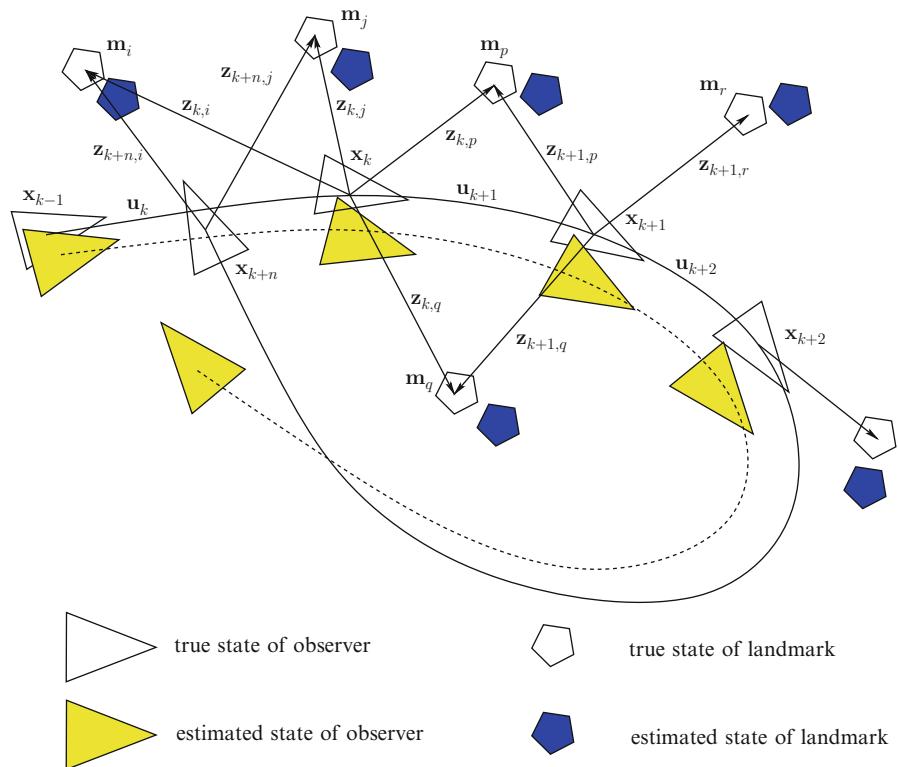
A new image frame is fetched from the camera and features (e.g., interest points, lines) which can be potential landmarks are extracted.

Data Association

Data association refers to finding correspondences between observations and landmarks. There are a number of methods in this regard. The active search/covariance-driven gating approach projects individual covariance predictions of landmarks into the observation image and limits the observation area based on suitable Mahalanobis distance. The Joint Compatibility Branch and Bound (JCBB) method [4] uses the fact that observation prediction errors are correlated with each other and hence calculates the joint probability of any set of correspondences. Active matching [5], on the other hand, considers the joint distribution and uses information theory to guide which landmarks to measure and when to measure them.

Relocalization

In case of rapid (unmodelled) motion, blur, and occlusion, camera pose tracking can fail and will result in subsequent corruption of the map. Relocalization refers to automatically detecting and recovering from such tracking failures to preserve the integrity of the map. Tracking failure can be detected by considering the percentage of unsuccessful observations and large uncertainty in the camera pose. The pose of the camera is recovered by establishing correspondence between the image and the map and solving the resulting perspective pose estimation problem [6].



Exploration: Simultaneous Localization and Mapping (SLAM), Fig. 2 SLAM problem

Motion Estimation

The observer's movement can be predicted based on odometry information or by making smooth motion assumptions (e.g., a constant velocity motion model). This information stands as a prior estimate of the observer state and covariance which is used later in the optimization stage. An example of such observer-state covariance computation is given by Eq. 5.

Optimization: Filtering-Based SLAM

Probabilistically the SLAM problem requires the computation of the joint posterior density of the landmark locations and observer (camera) state at time k , given the available observations and control inputs up to and including time k , together with the initial state of the observer [7–9].

This probability distribution can be stated as $P(\mathbf{x}_k, \mathbf{m} | \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0)$ where the quantities are defined as below (see Fig. 2).

\mathbf{x}_k = observer state vector

\mathbf{m} = vector describing the set of all landmark locations, i.e., landmark states \mathbf{m}_i

$\mathbf{Z}_{0:k}$ = set of all landmark observations $\mathbf{z}_{i,k}$

$\mathbf{U}_{0:k}$ = history of control inputs \mathbf{u}_k

Given an estimate for the problem at time $k-1$ (i.e., $P(\mathbf{x}_{k-1}, \mathbf{m} | \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1})$) a recursive solution to the problem can be obtained by employing a state transition model and an observation model which characterize the effect of the control input and observation, respectively.

Motion model: $P(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_k)$

Observation model: $P(\mathbf{z}_k | \mathbf{x}_k, \mathbf{m})$

Therefore, the SLAM algorithm can now be expressed as a standard two-step recursive prediction (time update) correction (measurement update) form as below:

Time Update

$$P(\mathbf{x}_k, \mathbf{m} | \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0) = \int P(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_k) \\ \times P(\mathbf{x}_{k-1}, \mathbf{m} | \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}, \mathbf{x}_0) d\mathbf{x}_{k-1} \quad (1)$$

Measurement Update

$$P(\mathbf{x}_k, \mathbf{m} | \mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) = \frac{P(\mathbf{z}_k | \mathbf{x}_k, \mathbf{m}) P(\mathbf{x}_k, \mathbf{m} | \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0)}{P(\mathbf{z}_k | \mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})} \quad (2)$$

Two main solution methods to the probabilistic SLAM problem include Extended Kalman Filter (EKF)-based SLAM [3] and Rao-Blackwellized Filter (RBF)-based SLAM [10, 11].

Extended Kalman Filter (EKF)-Based SLAM

As noted, the solution to the SLAM problem requires a joint state composed of the observer state and the state of every landmark, to be updated following each landmark observation. This motivates to use the Kalman filter which provides a recursive solution to the discrete data linear filtering problem. When the motion model and measurement errors/noise are independent white Gaussian, the Kalman filter computes an optimal gain which minimizes mean-square error of the posterior estimate of the system. The Gaussian distribution is completely characterized by its mean and covariance. Here correlations between observer state and different landmarks can be properly represented in the total system covariance matrix of the Kalman filter. As the system would generally be nonlinear, the Extended Kalman Filter (EKF) is used.

Propagation of Uncertainty A fundamental problem in estimating covariance matrices is how to propagate uncertainty through a function. Consider a nonlinear function f which acts on the vector \mathbf{a} of input variables to produce the output vector \mathbf{b} :

$$\mathbf{b} = f(\mathbf{a}) \quad (3)$$

Under affine approximation, the error covariance of the output vector $P_{\mathbf{b}\mathbf{b}}$ is obtained in terms of the error covariance of the input vector $P_{\mathbf{a}\mathbf{a}}$ by Eq. 4 (The covariance between two arbitrary vectors \mathbf{a} and \mathbf{b} is defined as $P_{\mathbf{ab}} = E[(\mathbf{a} - E[\mathbf{a}])(\mathbf{b} - E[\mathbf{b}])^T]$):

$$P_{\mathbf{b}\mathbf{b}} = \frac{\partial f}{\partial \mathbf{a}} P_{\mathbf{a}\mathbf{a}} \frac{\partial f^T}{\partial \mathbf{a}} \quad (4)$$

State Prediction The general form of the EKF prediction steps can be customized for SLAM to yield faster

computation. Note that there is no need to perform a time update for landmark states $\mathbf{m}_i : i = 1 \dots n$ if the landmarks are stationary. Hence, only the observer's state, \mathbf{x}_k , is predicted ahead. The new error covariance of the observer's predicted state should also be computed. In addition, though there is no change in error covariance of landmarks, there will be change in error covariance between each landmark and observer state (since observer state changed). Given below are the new time update equations for the covariance items that require an update.

Notation:

$\{\cdot\}^-$ = time update (prior) estimate

\mathbf{n} = observer motion model noise

$P_{\mathbf{nn}}$ = observer motion model noise covariance

f_v = observer state transfer function $\mathbf{x}_{k+1} = f_v(\mathbf{x}_k, \mathbf{n})$
such that $\mathbf{x}_{k+1}^- = f_v(\mathbf{x}_k, \mathbf{0})$

$$P_{\mathbf{x}_{k+1} \mathbf{x}_{k+1}}^- = \frac{\partial f_v}{\partial \mathbf{x}_k} P_{\mathbf{x}_k \mathbf{x}_k} \frac{\partial f_v^T}{\partial \mathbf{x}_k} + \frac{\partial f_v}{\partial \mathbf{n}} P_{\mathbf{nn}} \frac{\partial f_v^T}{\partial \mathbf{n}} \quad (5)$$

$$P_{\mathbf{x}_{k+1} \mathbf{m}_i}^- = \frac{\partial f_v}{\partial \mathbf{x}_k} P_{\mathbf{x}_k \mathbf{m}_i} \quad (6)$$

The computational complexity of the EKF-SLAM solution grows quadratically with the number of landmarks. This is due to the calculations involved in the EKF update steps.

Rao-Blackwellized Filter (RBF)-Based SLAM

Rao-Blackwellized filter based SLAM methods include FastSLAM [10] and FastSLAM 2 [11] algorithms. The algorithms utilize a key point in SLAM, i.e., given the observer path/trajectory (if the observer path is assumed correct), different landmark measurements are independent of each other. So a particle filter is used to estimate the observer's path and for each particle a map is maintained. Since landmarks within this map are independent/uncorrelated, they can be represented with separate low-dimensional EKFs. Therefore, it has linear complexity rather than the quadratic complexity of EKF-SLAM.

Landmark Representation As noted, probabilistic SLAM (EKF/RBF SLAM) requires the uncertainty of landmarks to be modelled. A landmark's uncertainty

modelling depends upon the parameterization used to represent its state. The simplest way to parametrize a landmark is by the common three-dimensional Cartesian coordinates $\mathbf{m}_i^{car} = (X_i, Y_i, Z_i)^T$. This is applicable in stereo camera SLAM systems where the landmark state can be obtained by triangulation, and the uncertainty follows the common Gaussian assumption. However, in single camera SLAM systems, in which it is not possible to estimate a landmark's state from a single measurement, there is large uncertainty in the depth direction which cannot be modelled as Gaussian. Therefore, alternative parameterizations are proposed to alleviate this issue.

Civera et al. [12] represent a feature using inverse depth parameterization as

$$\mathbf{m}_i^{idp} = (x_i, y_i, z_i, \theta_i, \phi_i, \rho_i)^T, \quad (7)$$

where x_i, y_i, z_i represent the camera position from which the feature was first observed, θ_i, ϕ_i represent the azimuth and elevation of the corresponding ray, and ρ_i represents the inverse depth to the landmark along this ray. The relationship of \mathbf{m}_i^{idp} to \mathbf{m}_i^{car} is as follows:

$$\mathbf{m}_i^{car} = \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} + \frac{1}{\rho_i} \begin{pmatrix} \cos \phi_i \sin \theta_i \\ -\sin \phi_i \\ \cos \phi_i \cos \theta_i \end{pmatrix}. \quad (8)$$

Marzorati et al. [13] use the homogenous coordinates

$$\mathbf{m}_i^{isp} = (x_i, y_i, z_i, \omega_i)^T, \quad (9)$$

which relates with \mathbf{m}_i^{car} as

$$\mathbf{m}_i^{car} = \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} = \frac{1}{\omega_i} \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \quad (10)$$

where ω_i is the inverse scale.

Although the number of parameters per landmark has increased from three of the minimal representation (\mathbf{m}_i^{car}), \mathbf{m}_i^{idp} and \mathbf{m}_i^{isp} are able to represent the landmark uncertainty as Gaussian and make the measurement equation more linear.

Optimization: Keyframe-Based SLAM

Keyframe-based SLAM splits camera pose tracking and mapping into separate processes. The accuracy



Exploration: Simultaneous Localization and Mapping (SLAM), Fig. 3 Wearable SLAM system for augmented reality by Castle et al. [18] – (1) handheld display with camera mounted on the back (2) active camera capable of pan and tilt (Reprinted from [18], with permission from Elsevier)

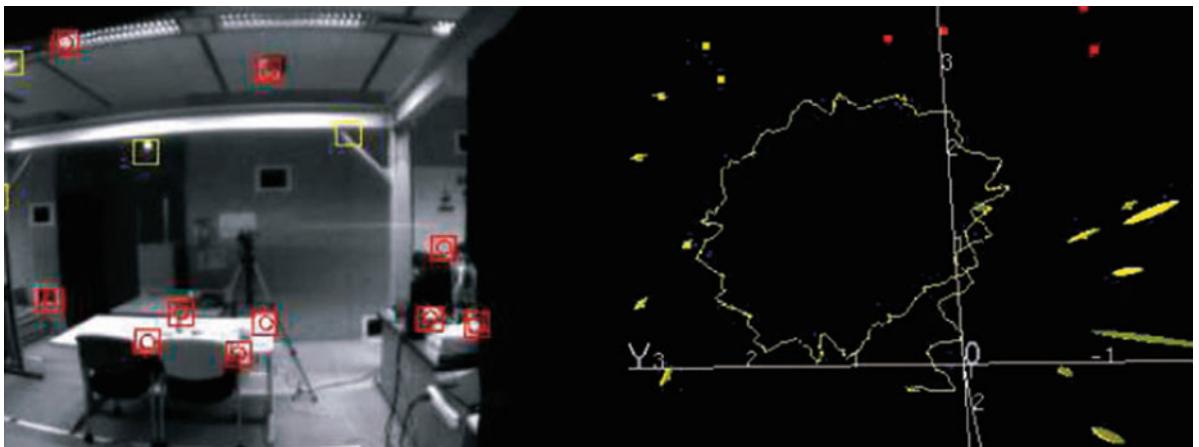
of camera pose estimation is increased by using measurements from large numbers of landmarks per image frame. The map is usually estimated by performing bundle adjustment/optimization over only a set of frames (keyframes) since measurements from nearby image frames provides redundant and therefore less information for a given computational budget [14]. Here a new keyframe can be selected and inserted to the map based on a sliding window of most recent camera poses or by ensuring a minimum distance to the pose of the nearest keyframe already in the map.

Loop Closure

Ideally when a SLAM system reobserves a previously mapped region of an environment, it should correctly recognize the corresponding landmarks in the map. However, particularly in long trajectory loops, landmark states in two such regions of the map can be incompatible given their uncertainty estimates. Therefore, loop closure methods are utilized to align such regions and make the map globally coherent. These include image-to-image matching [15], image-to-map matching [16], and hybrid methods [17].

New Landmark Initialization

Significant and distinguishable parts of the environment are selected as landmarks/features. In visual SLAM, these typically include interest points and edges/lines in the observed image. Upon selecting suitable landmarks, their position states and covariances are initialized and inserted into the map. Adding a new



Exploration: Simultaneous Localization and Mapping (SLAM), Fig. 4 A snapshot from MonoSLAM [20] as a humanoid robot walks in a circular trajectory. Here rectangular patches on the camera image (*left*) denote landmarks which are

tracked interest points. In the 3-D map (*right*), yellow trace is the estimated robot trajectory and ellipsoids show landmark location uncertainties (Copyright © 2007 IEEE. All rights reserved. Reprinted, with permission, from [20])

landmark \mathbf{m}_{new} to the map is a function of observer pose \mathbf{x}_k and new landmark observation \mathbf{z} which is given by $\mathbf{m}_{new} = f_{new}(\mathbf{x}_k, \mathbf{z})$. To add a new landmark, one needs to know the error covariance of the new landmark with other landmarks \mathbf{m}_{oth} and the observer \mathbf{x}_k . This can be calculated using the system uncertainty at that time as below:

$$P_{\mathbf{m}_{new}\mathbf{m}_{new}} = \frac{\partial f_{new}}{\partial \mathbf{x}_k} P_{\mathbf{x}_k\mathbf{x}_k} \frac{\partial f_{new}}{\partial \mathbf{x}_k}^T + \frac{\partial f_{new}}{\partial \mathbf{z}} R \frac{\partial f_{new}}{\partial \mathbf{z}}^T \quad (11)$$

$$P_{\mathbf{x}_k\mathbf{m}_{new}} = P_{\mathbf{x}_k\mathbf{x}_k} \frac{\partial f_{new}}{\partial \mathbf{x}_k}^T \quad (12)$$

$$P_{\mathbf{m}_{oth}\mathbf{m}_{new}} = P_{\mathbf{m}_{oth}\mathbf{x}_k} \frac{\partial f_{new}}{\partial \mathbf{x}_k}^T \quad (13)$$

Note R denotes the error covariance of the observation \mathbf{z} .

Application

Visual SLAM approaches have been successfully used in large scale outdoor environment mappings, and due to the human-like visual sensing of the camera, it has found applications in augmented reality applications [18] (see Fig. 3).

Open Problems

Despite recent work [19], the performance of visual SLAM is not sufficiently robust in dynamic environments where multiple moving objects are present in the scene. This is similar to the motion segmentation problem in computer vision. In addition, mapping with RGB-D cameras [1] is an emerging field with the recent introduction of affordable RGB-D cameras.

Experimental Results

Figure 4 gives a snapshot of one of the first real-time MonoSLAM system in operation [20].

References

- Henry P, Krainin M, Herbst E, Ren X, Fox D (2012) RGB-D mapping: using Kinect-style depth cameras for dense 3D modeling of indoor environments. Intern J Robot Res 31(5):647–663
- Hartley RI, Zisserman A (2004) Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, Cambridge. ISBN: 0521540518
- Smith R, Self M, Cheeseman P (1990) Autonomous robot vehicles. In: Estimating uncertain spatial relationships in robotics. Springer, New York, pp 167–193
- Neira J, Tardos JD (2001) Data association in stochastic mapping using the joint compatibility test. IEEE Trans Robot Autom 17(6):890–897

5. Chli M, Davison AJ (2008) Active matching. In: Proceedings of 10th European conference on computer vision (ECCV'08), Marseille
6. Williams B, Klein G, Reid I (2007) Real-time SLAM relocalisation. In: Proceedings of IEEE 11th international conference on computer vision ICCV 2007, Rio de Janeiro, pp 1–8
7. Durrant-Whyte H, Bailey T (2006) Simultaneous localization and mapping: part i. *IEEE Robot Autom Mag* 13(2):99–110
8. Bailey T, Durrant-Whyte H (2006) Simultaneous localization and mapping (SLAM): part ii. *IEEE Robot Autom Mag* 13(3):108–117
9. Thrun S, Burgard W, Fox D (2005) Probabilistic robotics. In: Intelligent robotics and autonomous agents. MIT, Cambridge
10. Montemerlo M, Thrun S, Koller D, Wegbreit B (2002) FastSLAM: a factored solution to the simultaneous localization and mapping problem. In: Proceedings of the AAAI national conference on artificial intelligence, Edmonton. AAAI, pp 593–598
11. Montemerlo M, Thrun S, Koller D, Wegbreit B (2003) FastSLAM 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In: Proceeding of the international conference on artificial intelligence (IJCAI), Acapulco, pp 1151–1156
12. Civera J, Davison AJ, Montiel J (2008) Inverse depth parametrization for monocular SLAM. *IEEE Trans Robot* 24(5):932–945
13. Marzorati D, Matteucci M, Migliore D, Sorrenti DG (2009) On the use of inverse scaling in monocular SLAM. In: Proceedings of IEEE international conference on robotics and automation ICRA'09, Kobe, pp 2030–2036
14. Strasdat H, Montiel JMM, Davison AJ (2010) Real-time monocular SLAM: why filter? In: Proceedings of IEEE international robotics and automation (ICRA) conference, Anchorage, pp 2657–2664
15. Cummins M, Newman P (2008) FAB-MAP: probabilistic localization and mapping in the space of appearance. *Intern J Robot Res* 27(6):647–665
16. Williams B, Cummins M, Neira J, Newman P, Reid I, Tardos J (2008) An image-to-map loop closing method for monocular SLAM. In: Proceedings of IEEE/RSJ international conference on intelligent robots and systems IROS 2008, Nice, pp 2053–2059
17. Eade E, Drummond T (2008) Unified loop closing and recovery for real time monocular SLAM. In: BMVC 2008, Leeds
18. Castle RO, Klein G, Murray DW (2010) Combining monoSLAM with object recognition for scene augmentation using a wearable camera. *J Image Vis Comput* 28(11): 1548–1556
19. Migliore D, Rigamonti R, Marzorati D, Matteucci M, Sorrenti DG (2009) Use a single camera for simultaneous localization and mapping with mobile object tracking in dynamic environments. In: Proceedings of international workshop on safe navigation in open and dynamic environments application to autonomous vehicles, Kobe
20. Davison AJ, Reid ID, Molton ND, Stasse O (2007) MonoSLAM: real-time single camera SLAM. *IEEE Trans Pattern Anal Mach Intell* 29(6):1052–1067

Extended Gaussian Image (EGI)

Sing Bing Kang¹ and Berthold K. P. Horn²

¹Microsoft Research, Redmond, WA, USA

²Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, USA

Synonyms

Surface orientation histogram (discrete version of EGI)

Related Concepts

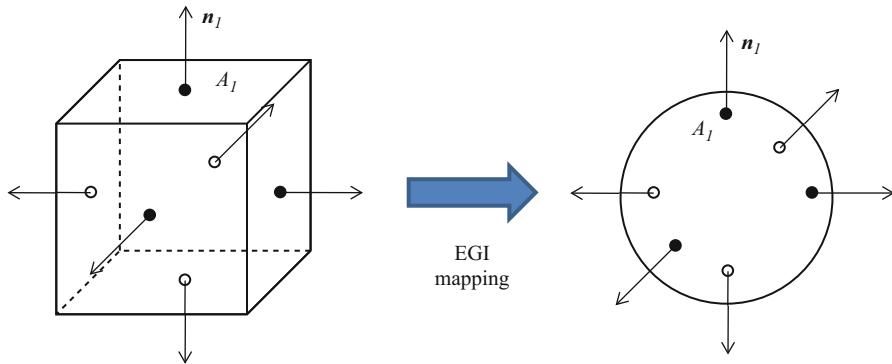
► [Extended Gaussian Image \(EGI\)](#)

Definition

The extended Gaussian image (EGI) of a 3-D object is a function defined on a unit sphere. The value of the EGI at a point on the unit sphere is the inverse of the curvature at the corresponding point on the object – where the corresponding point is the one that has the same surface orientation. In the case of a polyhedral object, the value on the sphere is zero except for impulses at points on the sphere corresponding to faces of the polyhedron. The “size” or volume of each impulse is equal to the area of the corresponding face. The mapping from a 3-D polyhedron to the EGI is illustrated in Fig. 1. Figure 2 shows the mapping for a 3-D piecewise smooth object.

Background

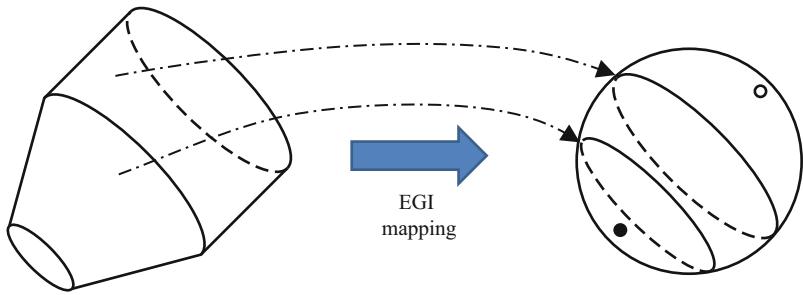
Object recognition and determination of object pose (orientation and translation) are two fundamental, interlinked tasks in 3-D computer vision and robotics. The shape of an object can be measured directly using, for example, a rangefinder or binocular stereo techniques or indirectly using, say, photometric stereo. A representation for that shape is needed that makes recognition and attitude determination relatively easy. One requirement is that the representation transforms in a simple way when the object is rotated.



Extended Gaussian Image (EGI), Fig. 1 The EGI of a polyhedron. *Left:* cube with surface normals. *Right:* arrows on the unit sphere represent impulses corresponding to faces of the cube

Extended Gaussian Image (EGI), Fig. 2

The EGI of a piecewise smooth object. *Left:* piecewise smooth object. *Right:* the two flat ends map to two impulses while the conical surface maps to a small circle, and the cylindrical surface maps to a large circle on the sphere



Theory

EGIs can be used to represent both smoothly curved objects as well as polyhedra. For ease of explanation, the convex polyhedra are discussed first. Minkowski [11] showed in 1897 that a convex polyhedron is fully specified (up to translation) by the areas and orientations of its faces (see also [14] and [10]). The area and orientation of faces can be represented conveniently by point masses (i.e., impulses of mass density) on a sphere. The extended Gaussian image of a polyhedron is obtained by placing a mass equal to the area of each face at a point on the sphere with the same surface orientation as the face.

More specifically, the Gaussian image of an object is obtained by mapping from a point on the surface of the object to the point on the unit sphere that has the same surface normal. In the case of a convex object, this mapping is invertible – only one point on the object's surface corresponds to a point on the unit sphere. This mapping extends in a natural way to curves and surface patches. The Gaussian curvature is the limit of the ratio of the (signed) area of a patch on the unit sphere to the area of the corresponding patch

on the surface of the object, as the size of the patches becomes smaller and smaller. The Gaussian curvature is everywhere nonnegative in the case of a convex object. The extended Gaussian image (EGI) associates a value with a point on the unit sphere equal to the inverse of the Gaussian curvature at the corresponding point on the object.

A.D. Alexandrov showed that a smoothly curved convex object is fully specified by curvature given as a function of surface orientation [1]. So the EGI representation is unique for both smoothly curved convex objects and convex polyhedra. Note, however, that neither of the proofs is constructive and thus they do not provide a basis for reconstructing an object from its EGI. Importantly, reconstruction is *not* required for recognition and orientation determination.

The discrete and continuous versions of the EGI can be related as follows: An object with planar faces can be thought of as the limit of an object with patches of spherical surfaces as the radius of the curvature of the patches get larger and larger. A spherical patch of radius R contributes a constant value R^2 (its Gaussian curvature) to an area of the unit sphere corresponding to all of the surface orientations at points in

that patch. As the radius tends to infinity, the value increases, at the same time that the area on the sphere shrinks. So the EGI of a polyhedron consists of a set of impulses, each of which has “volume” equal to the area of the corresponding face. It is convenient to treat these impulses as weights (or arrows of varying length) placed on the unit sphere. This is the discrete EGI. Conversely, the continuous EGI can be viewed as the limit of point mass density on the sphere as an approximation of a smoothly curved surface using a tessellation consisting of planar facets is made finer and finer.

The EGI has the following properties:

- *Translation invariance.* The EGI is not affected by object translation.
- *Rotation tracking.* Rotation of the object induces an equal rotation of the EGI since the unit surface normals rotate with the object.
- *Total mass equals total surface area.* Follows directly from the definition.
- *Center of mass at origin.* If the object is closed and bounded, the center of mass of the EGI is at the center of the Gaussian sphere.
- *Uniqueness.* There is only one closed convex object that corresponds to a particular EGI [1, 15].

If the object is not convex, more than one point on the object will contribute to a given point on the Gaussian sphere. One way of extending the definition of the EGI in the non-convex case is to use the sum of the inverses of the absolute values of the Gaussian curvature of all points on the surface that have the same orientation. While still useful in recognition and orientation determination, multiple non-convex objects may have the same EGI. For example, the EGI of all tori with the same surface area ($4\pi^2 R\rho$) and axis orientation is $2R\rho \sec \eta$, while there is but one *convex* object with that EGI. (Here R and ρ are the major axis and minor axis, respectively, while η is the angle between the surface normal and the plane perpendicular to the axis of the torus.)

In generating an EGI from an object, explicit information on shapes of faces and their adjacency relationships is not kept. Interestingly, a convex polyhedron can be recovered from its EGI [6, 9] as well as face adjacency and edge length information [12]. For a much more detailed treatment on EGIs, see [4]. The EGI has also been examined more broadly in the context of orientation-based representations [8].

The original version of the EGI is translation invariant. While it allows orientation to be extracted without regard to translation, a separate step is required to compute the translation. A variant of the EGI, the Complex EGI [7], uses complex mass to represent area (magnitude) and distance (phase). This decouples translation from rotation, allowing rotation to be determined in the usual fashion using the magnitudes while phase is subsequently used to compute translation. A support-function-based representation described in [13] encodes descriptors that are both local (tangent plane) and global (position and orientation of tangent plane).

Other extensions of the EGI address another inherent feature of not explicitly coding structural information. For example, the CSG-EESI (Constructive Solid Geometry-Enhanced Extended Spherical Image) [17] has two levels. The higher level contains the CSG tree which describes who the various subparts form the body, while the lower level describes the sub-parts using enhanced spherical images similar to that of [15]. Another tree-like representation is that of Hierarchical Extended Gaussian Image (HEGI) [18]. An HEGI description can be constructed as a tree where each leaf node corresponds to an Extended Gaussian Image (EGI) description of a convex part resulting from the recursive convex hull decomposition of an object. The COSMOS system described in [3] uses support functions based on Gaussian curvature and mean curvedness; connectivity information is maintained as a list.

Application

The EGI has been used for recognition and bin-picking of topmost objects (e.g., [2, 5, 6, 19]). It has also been used for symmetry detection [16] (for both reflectional and rotational symmetries).

In practice, the continuous EGI of a smoothly curved object is numerically approximated. The surface of the object may be divided into many patches, each of which is approximated by a planar facet of known area and orientation. The unit sphere may also be tessellated into cells, in each of which a total is accumulated of the areas of facets that have orientations falling within the range of orientations of that cell. Regular and semiregular tessellations of the sphere can be obtained by projecting regular

polyhedra (Platonic solids) or semiregular polyhedra (Archimedean solids). Unfortunately there are only 5 regular solids, with the icosahedron having 20 facets, which is not a fine enough tessellation. There are 13 semiregular solids, but again there are relatively few faces in these tessellations, the truncated icosahedron, for example, having only 32 faces. Finer, but less regular, tessellations can be obtained by subdividing faces of the regular and semiregular tessellations.

In typical practical applications, the object is only partially observed. Assuming the object is not occluded by other objects, the EGI of an observed convex object has information on at most one hemisphere. For a non-convex object, there is the issue of self-occlusion, so that the weight distribution for a set of tracked surface normal may change with object orientation. The most obvious solution is that of brute force search through the space of possible discrete orientations (e.g., as was done in [7]), but this is computationally expensive. Ikeuchi [6] reduces the search space by computing the ratio of the surface area to the projected area and constraining the freedom of rotation (since this ratio is independent of planar rotation for a given line of sight).

References

- Alexandrov AD (1942) Existence and uniqueness of a convex surface with a given integral curvature. *C R (Doklady Acad Sci URSS (NS))* 35(5):131–134
- Brou P (1983) Finding objects in depth maps. PhD thesis, Department of Electrical Engineering and Computer Science, MIT
- Dorai C, Jain AK (1997) COSMOS – a representation scheme for 3D free-form objects. *PAMI* 19(10):1115–1130
- Horn BKP (1984) Extended Gaussian images. *Proc IEEE* 72(12):1671–1686
- Horn BKP, Ikeuchi K (1984) The mechanical manipulation of randomly oriented parts (picking parts out of a bin of parts). *Sci Am* 251(2):100–111
- Ikeuchi K (1981) Recognition of 3-D objects using the extended Gaussian image. In: *IJCAI*, Vancouver, pp 595–608
- Kang SB, Ikeuchi K (1993) The complex EGI: a new representation for 3-D pose determination. *PAMI* 15(7): 707–721
- Liang P, Taubes CH (1994) Orientation-based differential geometric representations for computer vision applications. *PAMI* 16(3):249–258
- Little JJ (1983) An iterative method for reconstructing convex polyhedra from extended Gaussian images. In: *Proceedings of the national conference on artificial intelligence*, Washington, DC, pp 247–254
- Lyusternik LA (1963) Convex figures and polyhedra. Dover, New York
- Minkowski H (1897) Allgemeine Lehrsätze über die konvexen Polyeder. In: *Nachrichten von der Königlichen Gesellschaft der Wissenschaften, mathematisch-physikalische Klasse*, Göttingen, pp 198–219
- Moni S (1990) A closed-form solution for the reconstruction of a convex polyhedron from its extended Gaussian image. In: *ICPR*, Atlantic City, NJ, vol 1, pp 223–226
- Nalwa VS (1989) Representing oriented piecewise C^2 surfaces. *IJCV* 3(2):131–153
- Pogorelov AV (1956) Differential geometry. Noordhoff, Groningen
- Smith DA (1979) Using enhanced spherical images. Technical report, MIT AI Lab Memo No. 530
- Sun C, Sherrah J (1997) 3D symmetry detection using the extended Gaussian image. *PAMI* 19(2):164–168
- Xie SE, Calvert TW (1988) CSG-EESI: a new solid representation scheme and a conversion expert system. *PAMI* 10(2):221–234
- Xu JZ, Suk M, Ranka S (1996) Hierarchical EGI: a new method for object representation. In: *3rd international conference on signal processing*, Beijing, vol 2
- Yang HS, Kak AC (1986) Determination of the identification, position and orientation of the topmost object in a pile. *CVGIP* 36(2/3):229–255

Extrinsic Parameters

► Camera Extrinsic Parameters

F

Face Identification

Xiaogang Wang

Department of Electronic Engineering, Chinese University of Hong Kong, Shatin, Hong Kong

Related Concepts

► [Face Identification](#)

Definition

Face identification is to automatically identify a person by computers based on a query face image. In order to determine the identity of the query face image, the face images of all the registered persons in the database are compared against the query face image and are re-ranked based on the similarities.

Background

Face identification is a powerful technology with wide applications to biometrics, surveillance, law enforcement, human-computer interaction, and image and video search. It is often confused with another research topic called face verification. Face verification is to validate a claimed identity based on the query image. It compares the query image against the face images whose identity is claimed and decides to either accept or reject the claimed identity. Face identification involves one-to-many matches, while face verification

involves one-to-one matches. As the number of registered persons in the database increases, both the accuracy and the efficiency of face identification decreases. However, the performance of the face verification is not much affected by the size of the database.

Face identification is one of the most important biometric technologies. Compared with other biometric attributes, face images are easy to be captured in a nonintrusive way. Face identification has higher compatibility with other systems. As the fast growth of digital cameras as well as images and videos on the Internet is fast, face identification has become one of the hottest research topics in computer vision.

Face identification has some major challenges, such as large intrapersonal variations, small interpersonal variations, and the small sample size problem, to be solved. Face images of the same person may appear very differently because of variations of poses, illuminations, aging, occlusions, makeups, hair styles, and expressions. On the other hand, if the database is large, some faces of different persons may be quite similar. It is critical for face identification algorithms to effectively extract the interpersonal variations and depress the intrapersonal variations. In face identification, the visual features to be compared are usually in very high-dimensional spaces, while each person registered in the database only has a few face images for training. Therefore, it is easy for the classifier to overfit the training set. This is called the small sample size problem. Many developed face identification algorithms are to address the three challenges motioned above. Detailed surveys of face identification methods can be found in [1, 2].

Theory and Applications

The face identification pipeline is shown in Fig. 1. Given an input image, the face region is first located by a face detection algorithm [3]. Then the landmarks on the face are automatically located by a face alignment algorithm [4] through matching the face region with a predefined face graph. An example of a face graph model is shown in Fig. 2. Face alignment is critically important for depressing the variations caused by poses and expressions. It is an important pre-step for extracting visual features, especially local features. At the preprocessing step, both geometric rectification and photometric rectification are applied. For example, given the face alignment result, face images are usually cropped and transformed such that the two eye centers and the mouth center of all the face images are at fixed positions. In order to reduce the effect of lighting variations, the face region can be normalized by histogram equalization or gamma correction. In some approaches [5], a difference-of-Gaussian (DoG) filter was used to remove both high-frequency variations caused by noise and low-frequency illumination variations.

Feature Extraction

There are four types of features used for face identification: geometric features, holistic features, local features, and semantic features. Geometric features measure the sizes of facial components and the distances between components. They were used in some early face recognition systems developed in 1980s and 1990s. Because of the lack of robustness and discriminative power, they are not widely used by current face identification approaches. Holistic features and local features are most popularly used for face identification. Holistic features are simply high-dimensional vectors by concatenating all the pixels within the face regions. Local features are extracted by characterizing the texture patterns within local regions. The local regions can be defined by a regular grid or the neighborhoods around detected landmarks. Compared with holistic features, extracting local features requires higher-resolution face images. Gabor responses [6] and local binary patterns (LBP) [7] are two types of local features most widely used.

Gabor responses are obtained by the convolution of local face regions with a set of Gabor kernels. Gabor kernels are characterized as localized,

orientation selective, and frequency selective, and they are similar to the receptive field profiles in cortical simple cells. A Gabor kernel is the product of a Gaussian envelope and a plane wave:

$$\Psi_k(\mathbf{x}) = \frac{\|\mathbf{k}\|}{\sigma^2} \cdot e^{-\frac{\|\mathbf{k}\|^2 \|\mathbf{x}\|^2}{2\sigma^2}} \cdot \left[e^{i\mathbf{k} \cdot \mathbf{x}} - e^{-\frac{\sigma^2}{2}} \right]. \quad (1)$$

$\mathbf{x} = (x, y)$ is the variable in the spatial domain and k is the frequency vector, which determines the scale and the orientation of the Gabor kernel,

$$k = k_s e^{i\phi_d}, \quad (2)$$

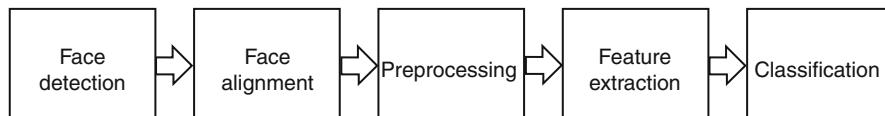
where $k_s = \frac{k_{\max}}{f_s}$, $f = 2, s = 0, \dots, L - 1$ and $\phi_d = (\pi \cdot d)/8$, for $d = 0, \dots, M - 1$. There are totally $L \times M$ Gabor kernels at L different scales along M different orientations. The number of oscillations under the Gaussian envelope function is determined by $\delta = 2\pi$. The term $\exp(-\sigma^2)$ is subtracted to make the kernel DC-free, and thus, the Gabor responses are insensitive to illumination. Figure 3 shows the real parts of Gabor kernels at 5 scales along 8 orientations.

Given an image $I(\mathbf{x})$, the Gabor response at location x_0 is computed by convolution:

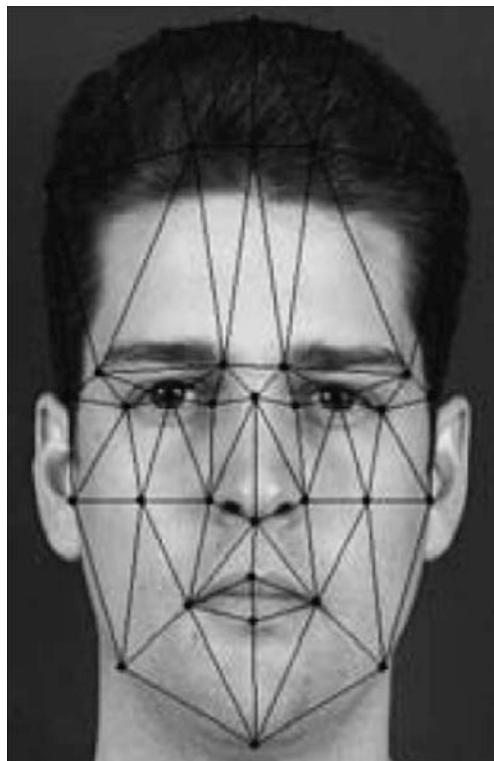
$$(\Psi_k * I)(x_0) = \int \Psi_k(x_0 - \mathbf{x}) I(\mathbf{x}) d^2(\mathbf{x}). \quad (3)$$

Since the phases of Gabor responses changes drastically with translation, usually only the magnitudes of the responses at landmarks or on a grid are used as local features for face identification. Gabor responses are robust to illumination variations and can tolerate the misalignment caused by variations of poses and expression to some extent.

LBP is a powerful texture descriptor introduced by Ojala et al. [8]. As shown in Fig. 4a, it defines the neighborhood of a pixel i by uniformly sampling P pixels along the circle centered at i with radius R . The pixels in the neighborhood are assigned with binary numbers by thresholding against the value of pixel i , as shown in Fig. 4b. These binary numbers are converted to a decimal number, which indicates the local binary pattern of pixel i . A local binary pattern is called uniform if it contains at most two bitwise transitions from 0 to 1 or vice versa when the binary string is considered circular. For example, 00011110 is a uniform pattern



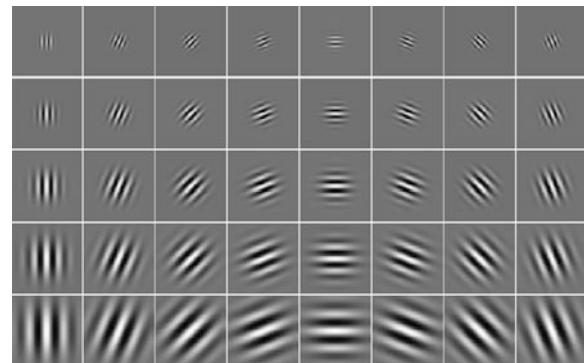
Face Identification, Fig. 1 Face identification pipeline



Face Identification, Fig. 2 Face graph model

and 00010100 is not. It is observed that uniform patterns appear much more frequently than nonuniform patterns. Both accuracy and computational efficiency of face identification can be improved if only uniform patterns are kept and all the nonuniform patterns are mapped to a single label.

Local binary patterns are treated as words of a codebook. The face region is divided into local regions by a grid as shown in Fig. 4c. The histograms of uniform patterns inside the local regions are used as features for face recognition. LBP characterizes a very large set of edges and has high discrimination power. If the values of centered pixels and their neighborhoods are under the same monotonous transformation, local binary patterns do not change. Therefore, LBP

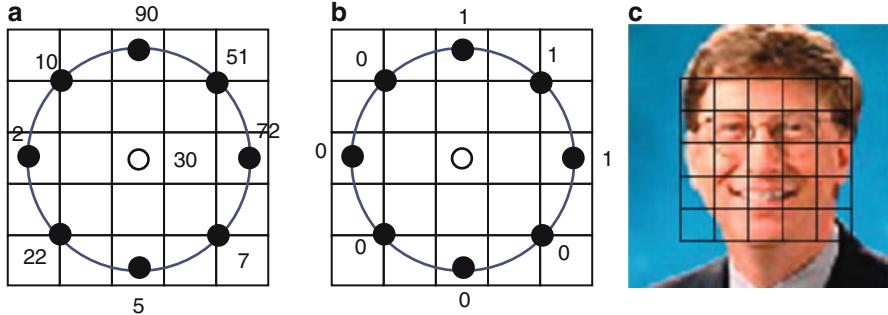


Face Identification, Fig. 3 Real parts of Gabor kernels at 5 scales and 8 orientations

is robust to lighting variations. Since the histograms within local regions are used as features, they are robust to misalignment and pose variations.

The idea of LBP has been extended to develop other types of local descriptors [9, 10] for face identification. For example, a learning-based descriptor was proposed in [11]. At each pixel, a low-level feature vector is extracted by sampling its neighboring pixels in the ring-based pattern as shown in Fig. 5. $r \times 8$ pixels are sampled at even intervals on the ring of r . After sampling, the feature vector is normalized into unit length. Then an encoding method is applied to encode the normalized feature vectors into discrete codes using the PCA tree. Histograms of codes within local regions are used as local features. Since the descriptor is unsupervised learned from the training set instead of being handcrafted, the learned codes are more uniformly distributed and the resulting code histogram can achieve better discriminative power and robustness trade-off than LBP.

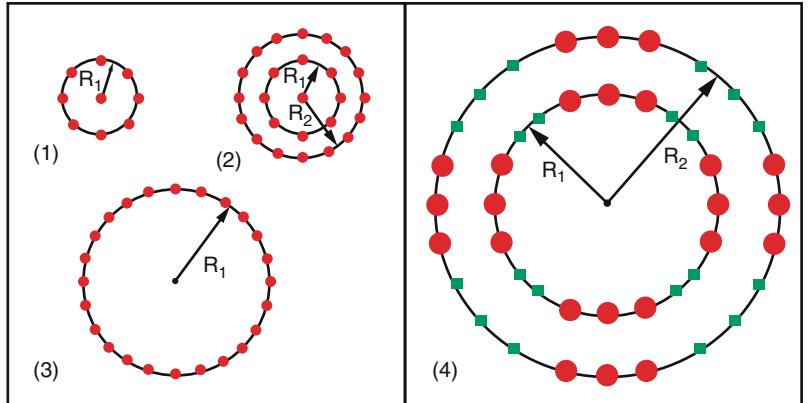
Semantic features were proposed for face identification in very recent years. Two types of semantic features, “attribute” and “simile,” were proposed in [12]. To extract the attribute features, a set of binary classifiers are trained to recognize the presence or absence of describable aspects of visual appearance (e.g., “does the face has double chin?” and “does the face have high



Face Identification, Fig. 4 LBP operator. (a): the neighborhood of a pixel. (b): the local binary pattern is labeled as $11100000 = 224$. (c) The face region is divided into local regions

Face Identification, Fig. 5

Different sampling methods used in [11] for the learning-based descriptor. (1) $R_1 = 1$, with center; $R_1 = 1$, $R_2 = 2$, with center; (3) $R_1 = 3$, no center; and (4) $R_1 = 4$, $R_2 = 7$, no center (The sampling dots on the green square-labeled arcs are omitted for better visuality)



cheekbones?”). The output of each classifier is a score between 1 and -1 . The raw outputs of attribute classifiers are concatenated to form the feature vector for face identification. The simile features of a face image are the similarities between this face image and a set of reference people. Semantic features are high-level visual features. They are inspired by the process of face identification by human beings, since human beings often describe a face using some attribute terms or using its similarities with some known people. Semantic features are more insensitive to the variations of poses, illuminations, and expressions.

The resolution of face images is an important factor affecting feature extraction. Local features usually require higher resolutions than holistic features. As the development of digital cameras, extraction of features from very high resolution images [13] is an important topic to be explored.

Classification

The extracted features are usually in very high-dimensional spaces and include large intrapersonal

variations and noise. It is inaccurate and inefficient to directly use them for face identification. They are typically projected into low-dimensional spaces or fed into trained classifiers in order to compare two face images. PCA [14] is widely used for dimension reduction. In face identification, intrapersonal variations are the main factor deteriorating the identification accuracy and may be larger than interpersonal variations. However, they cannot be depressed by PCA. Therefore, although PCA can improve the identification efficiency, it cannot effectively improve the accuracy.

Bayesian face recognition [15] effectively models of the intra-personal variations using the intrapersonal subspace, which is spanned by the eigenvectors $\{e_k\}$ and eigenvalues $\{\lambda_k\}$ of matrix,

$$C = \sum_{\ell(x_i)=\ell(x_j)} (x_i - x_j)(x_i - x_j)^T, \quad (4)$$

where x_i is the feature vector and $\ell(x_i)$ is its identity. The distance between two feature vectors x and x' is computed as

$$d(\mathbf{x}, \mathbf{x}') = \sum y_k^2 / \lambda_k, \quad (5)$$

where y_k is the projection of $\mathbf{x} - \mathbf{x}'$ on eigenvector \mathbf{e}_k . Since the intrapersonal variations concentrate on the first few eigenvectors with the largest eigenvalues λ_k , they are effectively depressed by weighting the inverse of λ_i . Therefore, Bayesian face recognition can effectively improve the accuracy by reducing the intrapersonal variations.

LDA [16] is another widely used subspace method. LDA tries to find the subspace that best discriminates different face classes by maximizing the between-class scatter matrix S_b , while minimizing the within-class scatter matrix S_w in the projective subspace. Suppose there are L persons in the training set and X_c is the set of feature vectors of person c . S_w and S_b are defined as

$$S_w = \sum_{c=1}^L \sum_{x_i \in X_c} (\mathbf{x}_i - \mathbf{m}_c)(\mathbf{x}_i - \mathbf{m}_c)^T, \quad (6)$$

$$S_b = \sum_{c=1}^L n_c (\mathbf{m}_c - \mathbf{m})(\mathbf{m}_c - \mathbf{m})^T, \quad (7)$$

where \mathbf{m}_c is the mean of feature vectors in X_c , n_c is the number of feature vectors in X_c , and \mathbf{m} is the mean of all the feature vectors in the training set. LDA subspace is spanned by a set of vectors W satisfying

$$W = \arg \max \left| \frac{W^T S_b W}{W^T S_w W} \right|. \quad (8)$$

Feature vectors are first projected into the LDA subspace before being used to compare the distance between two face images. W can be computed as the eigenvectors of $S_w^{-1} S_b$. However, because of the small sample size problem, S_w^{-1} is often singular and it is easy for W to overfit the training set. Many improvements of LDA, such as dual-space LDA [17], null-space LDA [18], and direct LDA [19], have been made to address this problem.

In [20, 21], the three popular subspace methods, PCA, Bayesian face recognition, and LDA, were unified under one framework. Based on this framework, a unified subspace analysis method was developed using PCA, Bayes, and LDA as three steps. First PCA is used to reduce noise and data dimensionality. Then within the PCA subspace, Bayesian analysis is used to learn the intrapersonal subspace and to depress the intrapersonal variations. In the last step of LDA,

interpersonal variations are extracted and the noise is further reduced. A 3D parameter space is constructed using the three subspace dimensions as axes. The choice of the three parameters greatly affects the performance of face identification. The original PCA, LDA, and Bayes methods only occupy some local lines or areas in the 3D parameter space and therefore cannot achieve the best performance. Searching through this parameter space, an optimal subspace for face identification is obtained.

When reducing the data dimensionality by PCA to solve the small sample size problem, some discriminative features have to be discarded in order to train a stable LDA classifier. A random sampling subspace method [22, 23] was proposed to effectively solve the small sample size problem and keep nearly all the discriminative information in the meanwhile by integrating multiple classifiers. Many works [24] have shown that integrating an ensemble of classifiers can effectively improve the performance of face identification. The multiple classifiers can be constructed based on different features, within different local face regions, using different training examples or even under different parameter settings.

PCA and LDA are linear subspace methods and assume a Euclidean structure of the face space. Manifold learning [25] assumes that face images reside on a manifold structure and obtains a face subspace using locality preserving projections. It is used to reduce the variations of poses, expressions, and illuminations.

Besides subspace methods, other classifiers such as SVM [26] and boosting [27] were also used for face identification. In recent years, sparse representation [28] was applied to face identification and drew a lot of attentions. Using ℓ^1 regularization, it effectively handles errors due to occlusion and corruption by exploiting the fact that these errors are often sparse. LDA was extended to non-parametric discriminant analysis assuming that the distribution of face data is non-Gaussian [29].

Face Identification Based on Other Types of Data

Most of the face identification methods assumed that the queries are 2D face photos. However, other types of face data, such as 3D data, video sequences, infrared images, and sketch drawings, can also be used as queries in some specific application scenarios. Compared to

extensive research work on 2D face photos, relatively less work has been on these topics.

3D face identification [30] measures geometry of rigid features on the face. Compared with 2D face recognition, it is more robust to the variations of illuminations, makeups, and poses. Its major technological limitation is the acquisition of 3D face data, which requires range cameras. With the development and popularity of 3D cameras, the research on 3D face identification is drawing more and more attentions.

Compared with single images, video sequences provide extra dynamic information and multiple face instances and therefore can effectively improve both the robustness and accuracy of face identification [31, 32]. The head pose may vary significantly in a video sequence, and therefore video-based face identification requires robust face tracking and pose estimation. Faces in videos are usually at lower resolutions with blurring effects. These factors add extra challenges for video-based face identification.

In order to be robust to lighting variations, infrared imaging was used for face identification in recent years [33, 34]. Infrared face images were used as queries. Infrared cameras can capture clear images at nighttime and in hazy conditions. An external illumination source is not required since the face emits thermal energy.

In law enforcement, automatically identifying suspects from the police mug-shot database can help the police narrow down potential suspects quickly. However, in some cases, the photo image of a suspect for query is not available. The best substitute is often a sketch drawing based on the recollection of an eyewitness. Therefore, face identification using sketch drawings as queries become an interesting research topic. Photos and sketches are in different modalities and have significant differences on shape and texture. It requires matching sketches with photos in the database and is more challenging. It also involves the unknown psychological mechanism of sketch generation. A few works [35, 36] have been done on face sketch recognition. However, this problem is far from being well solved yet.

Benchmark Databases

In the past decades, many databases have been built as benchmarks to evaluate the performance of face

identification and face verification. Among existing databases, FRGC [37] and Multi-PIE [38] are in larger scales and are widely used. Face images in most existing databases were collected in controlled environments. In recent years, there has been an increasing interest in studying face recognition in uncontrolled environments. It was shown that the performance of many existing face recognition approaches dropped significantly in such uncontrolled environments. The LFW [39] database collected 13,233 face images of 1,040 from the Internet and these face images were collected in uncontrolled environments. LFW has drawn a lot of attentions in the field face recognition in recent years. Besides the 2D face photo images, some databases include other types of face data. For examples, FRGC includes both 2D and 3D face data. XM2VTS [40] includes face video sequences for the evaluation of face identification using videos as queries. The CUFS database [36] includes face photos and face sketch drawings of 606 subjects and is used for the research of face sketch recognition.

References

- Li SZ, Jain A (eds) (2004) *Handbook of face recognition*. Springer, New York
- Zhao W (2003) Face recognition: a literature survey. *ACM Comput Surv* 35:399–458
- Viola P, Jones MJ (2004) Robust real-time face detection. *Int J Comput Vis* 57:137–154
- Cootes TF, Taylor CJ (2004) Statistical models of appearance for computer vision. Technical report, imaging science and biomedical engineering, University of Manchester
- Hua G, Akbarzadeh A (2009) A robust elastic and partial matching metric for face recognition. In: Proceedings of the IEEE international conference on computer vision, Kyoto
- Wiskott L, Fellous J, Kruger N, Malsburg C (1997) Face recognition by elastic bunch graph. *IEEE Trans PAMI* 19:775–779
- Ahonen T, Hadid A, Pietikainen M (2006) Face description with local binary patterns: application to face recognition. *IEEE Trans PAMI* 28:2037–2041
- Ojala T, Pietikainen M, Maenpaa T (2002) Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans PAMI* 24:971–987
- Zhang W, Shan S, Gao W, Chen X, Zhang H (2005) Local gabor binary pattern histogram sequence (LGBPHS): a novel non-statistical model for face representation and recognition. In: Proceedings of the IEEE international conference on computer vision, Beijing
- Tan X, Triggs B (2010) Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE Trans Image Process* 19(6):1635–1650

11. Cao Z, Yin Q, Tang X, Sun J (2010) Face recognition with learning-based descriptor. In: Proceedings of the IEEE international conference on computer vision and pattern recognition (CVPR), Providence
12. Kumar N, Berg AC, Belhumeur P (2009) Attribute and simile classifiers for face verification. In: Proceedings of the IEEE international conference on computer vision, Kyoto
13. Lin D, Tang X (2006) Recognizing high resolution faces: from macrocosm to microcosm. In: Proceedings of the European conference on computer vision (ECCV), Graz
14. Turk M, Pentland A (1991) Face recognition using eigenfaces. *J Cognit Neurosci* 3:71–86
15. Moghaddam B, Jebara T, Pentland A (2000) Bayesian face recognition. *Pattern Recognit* 33:1771–1782
16. Belhumeur P, Hespanda J, Kriegman D (1997) Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *IEEE Trans PAMI* 19:711–720
17. Wang X, Tang X (2004) Dual-space linear discriminant analysis for face recognition. In: Proceedings of the IEEE international conference on computer vision and pattern recognition (CVPR), Washington, DC
18. Chen L, Liao H, Ko M, Lin J, Yu G (2000) A new lda-based face recognition system which can solve the small sample size problem. *J Pattern Recognit* 33:1713–1726
19. Yu H, Yang J (2001) A direct lda algorithm for high-dimensional data – with application to face recognition. *Pattern Recognit* 34:2067–2070
20. Wang X, Tang X (2003) Unified subspace analysis for face recognition. In: Proceedings of the IEEE international conference on computer vision, Nice
21. Wang X, Tang X (2004) A unified framework for subspace face recognition. *IEEE Trans PAMI* 26:1222–1228
22. Wang X, Tang X (2004) Random sampling lda for face recognition. In: Proceedings of the IEEE international conference on computer vision and pattern recognition (CVPR), Washington, DC
23. Wang X, Tang X (2006) Random sampling for subspace face recognition. *Int J Comput Vis* 70:91–104
24. Su Y, Shan S, Chen X, Gao W (2007) Hierarchical ensemble of global and local classifiers for face recognition. In: Proceedings of the IEEE international conference on computer vision, Rio de Janeiro
25. He X, Yan S, Hu Y, Niyogi P, Zhang H (2005) Face recognition using laplacianfaces. *IEEE Trans PAMI* 27:328–340
26. Heisele B, Ho P, Poggio T (2001) Face recognition with support vector machines: Global versus component-based approach. In: Proceedings of the IEEE international conference on computer vision, Vancouver
27. Wang X, Zhang C, Zhang Z (2009) Boosted multi-task learning for face verification with applications to web images and videos search. In: Proceedings of the IEEE international conference on computer vision and pattern recognition (CVPR), Miami
28. Wright J, Yang AY, Ganesh A, Sastry SS, Ma Y (2009) Robust face recognition via sparse representation. *IEEE Trans PAMI* 31:210–227
29. Li Z, Lin D, Tang X (2009) Nonparametric discriminant analysis for face recognition. *IEEE Trans PAMI* 31:755–761
30. Flynn P, Chang K, Bowyer K (2006) A survey of approaches and challenges in 3d and multi-modal 3d+2d face recognition. *Comput Vis Image Underst* 101:1–5
31. Tang X, Li Z (2004) Frame synchronization and multi-level subspace analysis for video-based face recognition. In: Proceedings of the IEEE international conference on computer vision and pattern recognition (CVPR), Washington, DC
32. Lee K, Ho J, Yang M, Kriegman D (2003) Video-based face recognition using probabilistic appearance manifolds. In: Proceedings of the IEEE international conference on computer vision and pattern recognition (CVPR), Madison
33. Li S, Chu R, Liao S, Zhang L (2007) Illumination invariant face recognition using near-infrared images. *IEEE Trans PAMI* 29:627–639
34. Lin D, Tang X (2006) Inter-modality face recognition. In: Proceedings of the European conference on computer vision (ECCV), Graz
35. Tang X, Wang X (2003) Face sketch synthesis and recognition. In: Proceedings of the IEEE international conference on computer vision, Nice
36. Wang X, Tang X (2009) Face photo-sketch synthesis and recognition. *IEEE Trans PAMI* 31(11):1955–1967
37. Phillips, PJ, Flynn PJ, Bowyer KW (2005) Overview of the face recognition grand challenge. In: Proceedings of the IEEE international conference on computer vision and pattern recognition (CVPR), San Diego
38. Gross R, Matthews I, Cohn J, Kanade T, Baker S (2008) Multi-pie. *Image Vis Comput* 26:15–26
39. Huang GB, Manu R, Berg T, Learned-Miller E (2007) Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, University of Massachusetts, Amherst
40. Messer K, Matas J, Kittler J, Luettin J, Maitre G (1999) Xm2vtsdb: the extended m2vts database. In: Proceedings of the IEEE international conference on audio and video-based biometric person authentication, Washington, DC

Face Modeling

Zicheng Liu¹ and Zhengyou Zhang²

¹Microsoft Research, Microsoft Corporation, Redmond, WA, USA

²Microsoft Research, Redmond, WA, USA

Synonyms

[Face reconstruction](#); [Three dimensional face modeling](#)

Related Concepts

► [Structure-from-Motion \(SfM\)](#)

Definition

Face modeling usually refers to the problem of recovering 3D face geometries from one or more images,

though the recovery of lighting and skin reflectance is sometimes considered as face modeling as well.

Background

Because face is a special type of object which people are all familiar with, there has been tremendous interests among researchers in the problem of 3D face reconstruction. The most reliable and accurate way to obtain face geometries is by using active sensors such as laser scanners and structured light systems. So far, laser scanners are the most commonly used and most accurate active sensors. Structured light systems are becoming more popular because they are capable of capturing continuous motions. Some structured light systems use visible light sources, while others use invisible light sources such as infrared lights. Visible light systems give better signals, but they are intrusive.

The disadvantage of active sensors is that they are usually expensive and not widely available. Recently, Microsoft released a depth camera, called Kinect, which uses active sensors. The device works with both game consoles and PCs, and has popularized the use of depth cameras in many applications including face modeling.

Given that cameras are everywhere, it is not surprising that researchers have been fascinated by the problem of 3D face reconstruction from images. This can be thought of as a special case of the classical computer vision problem of 3D structure recovery from images. One could directly apply a generic 3D reconstruction technique to face images. It usually does not work very well because face skins are usually smooth making it difficult to find accurate matchings across images. Since all human faces are similar in terms of their rough shape and topology, a lot of the research has been devoted to developing techniques that leverage the prior knowledge on faces. For example, one can start from a generic face mesh and try to adjust the vertex positions of the mesh to fit the image observations. To reduce the number of degrees of freedoms involved in the fitting process, people have proposed to use a linear space of face geometries to constrain the parameter space. One can use a set of pre-captured or hand-designed face geometries as the examples. Any face is assumed to be a linear combination of the example faces.

Theory

Face modeling techniques can be divided into two categories based on whether the illumination effects are modeled or not. The methods that belong to the first category do not model illumination effects [1–4]. They have origins from structure from motion or stereovision. The methods in the second category takes illumination into account [5–8]. In fact, they leverage the shading information for the geometry reconstruction. These techniques can be traced to shape from shading. For a more detailed and systematic descriptions of face modeling techniques and applications, the readers are referred to the book [20].

Regardless of which method is used, the first question is how to represent a face. As mentioned earlier, linear space representation has been shown to be an effective way to constrain the parameter space. There are two different ways to construct a linear space representation.

The first, called a morphable model, was proposed by Blanz and Vetter in their seminar paper [5]. They used a set of existing face meshes obtained by laser scanners. These meshes must be aligned so that there is a correspondence between the vertices of different meshes.

Let $\mathbf{v}_i = (X_i, Y_i, Z_i)^T$, $i = 1, \dots, n$, denote the vertices of a face mesh. Its geometry is represented by a vector

$$\mathcal{S} = (\mathbf{v}_1^T, \dots, \mathbf{v}_n^T)^T = (X_1, Y_1, Z_1, \dots, X_n, Y_n, Z_n)^T. \quad (1)$$

Suppose there are $m + 1$ face meshes which are obtained by using laser scanner or some other means. Let \mathcal{S}^j denote the geometry of the j th mesh, $j = 1, \dots, m + 1$. These faces generate a linear space of face geometries:

$$F = \left\{ \sum_{j=1}^{m+1} \alpha_j \mathcal{S}^j : \sum_{j=1}^{m+1} \alpha_j = 1 \right\}. \quad (2)$$

Let \mathcal{S}^0 denote the average face geometry, that is, $\mathcal{S}^0 = \frac{1}{m+1} \sum_{j=1}^{m+1} \mathcal{S}^j$. Denote $\delta \mathcal{S}^j = \mathcal{S}^j - \mathcal{S}^0$, $j = 1, \dots, m + 1$. Note that these vectors are linearly dependent. Principal component analysis can be performed on the vectors $\delta \mathcal{S}^1, \dots, \delta \mathcal{S}^{m+1}$. Let $\sigma_1^2, \dots, \sigma_m^2$ denote the eigenvalues with $\sigma_1^2 \geq \sigma_2^2 \geq \dots \geq \sigma_m^2$. Let

$\mathcal{M}^1, \dots, \mathcal{M}^m$ denote the corresponding eigenvectors. Then any face $S \in F$ can be represented as the average face S^0 plus a linear combination of the eigenvectors, that is,

$$S = S^0 + \sum_{j=1}^m c_j \mathcal{M}^j. \quad (3)$$

c_j are the geometry coefficients. The prior probability for geometry coefficients c_1, \dots, c_m is given by

$$Pr(c_1, \dots, c_m) = e^{-\frac{1}{2} \sum_{j=1}^m \frac{c_j^2}{\sigma_j^2}}. \quad (4)$$

The second approach is to manually design the average mesh S^0 and a set of deformation vectors which act as δS^j as proposed in [2, 9]. Each deformation vector corresponds to an intuitive way of deforming the face. For face modeling purpose, the deformation vectors are used in almost the same way as the eigenvectors. The only difference is that for eigenvectors, there is a prior probability for the geometry coefficients (see (Eq. 4)). For deformation vectors, one can predefined a valid range for each model coefficient.

Given the face representation as shown in (Eq. 3), the problem of face reconstruction becomes searching for the model coefficients c_j . As mentioned earlier, the method of solving for the model coefficients can be divided into two categories depending on whether they model illumination effects or not.

The methods that belong to the first category typically assume there are two or more input images corresponding to different views of a face. If the camera motions corresponding to the views are not known, one can use structure-from-motion techniques to estimate the camera motion and obtain a set of 3D points, which are usually quite sparse and noisy. After that, one can solve for the model coefficients and the head pose by minimizing the total distances of the 3D points to the model. Numerically, one can solve it through an Iterative Closest Point (ICP) procedure. For details, the readers are referred to [2].

If the camera motions corresponding to the views are known such as in a stereo rig, one can perform dense stereo matching and obtain a depth map of the face. One such system was developed by Chen and Medioni [10]. The obtained depth map is usually quite noisy and has spikes. One could use a face model to fit the obtained depth map or inject a face model representation in the stereo reconstruction process in a way

similar to the model-based bundle adjustment formulation in [11], but the camera motions are no longer variables anymore in this case.

Another approach is to use two orthogonal views [12–14]: one frontal view and one side view. The frontal view provides the information relative to the horizontal and vertical axis, while the side view provides depth information. Since the number of feature points that can be detected on the two views is usually quite small, one could use a linear space face representation to fit the detected feature points and obtain a complete face geometry.

The methods that belong to the second category only require a single view. Since the publication of Blanz and Vetter's paper [5] which assumed a single light source, researchers have extended their technique to handle more general lighting conditions. These techniques leverage shading information and they typically recover the shape, reflectance, and lighting simultaneously. Most techniques assume the face skin is Lambertian. Similar to linear space representation for face geometry, the diffuse reflectance components (also called albedo or texture) can also be represented as a linear combination of example face albedos. In this way, the unknowns for the reflectance are the albedo (also called texture) coefficients. One effective tool to model the lighting is the spherical harmonics representation which was proposed by Ramamoorthi and Hanrahan [15] and Basri and Jacobs [16]. The basic idea is that the irradiance can be well approximated by using a linear combination of a small number of spherical harmonic basis functions. They showed that with nine spherical harmonic basis functions, the average approximation error is no more than 1%. The coefficients of the spherical harmonic basis are called lighting coefficients. Given the pose, geometry model coefficients, albedo coefficients, and lighting coefficients, one can synthesize an image of the face. The objective is to solve for these parameters so that the synthesized image matches the input image. The optimization problem can be solved through an iterative procedure. For details, the readers are referred to [8].

Application

Face modeling has many applications. The techniques allow people to create personalized avatars which can be used in chatting rooms, e-mails, greeting cards,

and games. Many human-machine dialog systems use realistic-looking avatars as visual representation of the computer agent that interacts with the human user. Face models are useful for 3D head pose tracking and facial expression tracking. In teleconferencing, face modeling techniques can be used for eye-gaze correction to improve video conferencing experience [17]. Face modeling techniques are useful for face recognition to handle pose variations [18] and lighting variations [19].

References

1. Fua P, Miccio C (1998) From regular images to animated heads: a least squares approach. In: European conference on computer vision (ECCV), Freiburg, pp 188–202
2. Zhang Z, Liu Z, Adler D, Cohen M, Hanson E, shan Y (2004) Robust and rapid generation of animated faces from video images: a model-based modeling approach. *Int J Comput Vision* 58(2):93–120
3. Dimitrijevic M, Ilic S, Fua P (2004) Accurate face models from uncalibrated and ill-lit video sequences. In: Computer vision and pattern recognition (CVPR), II, Washington, DC, pp 1034–1041
4. Amberg B, Blake A, Fitzgibbon A, Romdhani S, Vetter T (2007) Reconstructing high quality face-surfaces using model based stereo. In: International conference on computer vision, Rio de Janeiro, Brazil
5. Blanz V, Vetter T (1999) A morphable model for the synthesis of 3d faces. In: Proceedings of the Annual conference series on computer graphics, SIGGRAPH, Los Angeles, pp 187–194
6. Zhang L, Wang S, Samaras D (2005) Face synthesis and recognition from a single image under arbitrary unknown lighting using a spherical harmonic basis morphable model. In: Computer vision and pattern recognition II (CVPR), San Diego, pp 209–216
7. Lee J, Moghaddam B, Pfister H, Machiraju R (2005) A bilinear illumination model for robust face recognition. In: International conference on computer vision, II, Beijing, pp 1177–1184
8. Wang Y, Zhang L, Liu Z, Hua G, Wen Z, Zhang Z, Samaras D (2009) Face re-lighting from a single image under arbitrary unknown lighting conditions. *IEEE Trans Pattern Recognit Mach Intell* 31(11):1968–1984
9. Liu Z, Zhang Z, Jacobs C, Cohen M (2000) Rapid modeling of animated faces from video. In: Proceedings of visual 2000, Lyon, pp 58–67
10. Chen Q, Medioni G (2001) Building 3-d human face models from two photographs. *J VLSI Signal Process* 27(1–2): 127–140
11. Shan Y, Liu Z, Zhang Z (2001) Model-based boundle adjustment with application to face modeling. In: International conference on computer vision, Vancouver, pp 644–651
12. Akimoto T, Suenaga Y, Wallace RS (1993) Automatic 3d facial models. *IEEE Comput Graph Appl* 13(5):16–22
13. Ip HH, Yin L (1996) Constructing a 3d individualized head model from two orthogonal views. *Vis Comput* 12:254–266
14. Dariush B, Kang SB, Waters K (1998) Spatiotemporal analysis of face profiles: detection, segmentation, and registration. In: Proceeding of the 3rd international conference on automatic face and gesture recognition, IEEE, pp 248–253
15. Ramamoorthi R, Hanrahan P (2001) A signal-processing framework for inverse rendering. In: SIGGRAPH, Los Angeles, pp 117–128
16. Basri R, Jacobs D (2003) Lambertian reflectance and linear subspaces. *Pattern Anal Mach Intell* 25(2):218–233
17. Yang R, Zhang Z (2002) Eye gaze correction with stereovision for video tele-conferencing. In: Proceedings of the 7th European conference on computer vision (ECCV), Copenhagen, vol II, pp 479–494
18. Romdhani S, Blanz V, Vetter T (2002) Face identification by fitting a 3d morphable model using linear shape and texture error functions. In: European conference on computer vision (ECCV'2002), IV, Copenhagen, pp 3–19
19. Qing L, Shan S, Gao W, Du B (2005) Face recognition under generic illumination based on harmonic relighting. *Int J Pattern Recognit Artif Intell* 19(4):513–531
20. Zicheng L, Zhengyou Z (2011) Face geometry and appearance modeling: concepts and applications. Cambridge University Press

Face Reconstruction

► Face Modeling

Factorization

Hanno Ackermann

Leibniz University Hannover, Hannover, Germany

Synonyms

[Structure-from-motion \(SfM\)](#); [Three dimensional reconstruction](#)

Definition

Given arbitrary many images and 3D-points, the so-called *factorization* algorithm [1] is a noniterative technique for simultaneously estimating 3D-structure along with orientations and positions of the cameras which observed the 3D-scene or object. It is based on the factorization of a matrix consisting of all measurements.

The original algorithm requires a particular affine camera. It was later extended to more general affine cameras and even to the full projective camera model. Other variants can handle lines, triangles, or ellipses instead of 2D-point correspondences. It can be further extended to nonrigid scenes or objects, not merely rigid ones. It can also minimize geometric error criteria instead of algebraic errors and even allow unknown entries in the measurement matrix.

Background

Let the 3×3 upper triangular matrix K_i denote the calibration of the i th camera, the 3×3 rotation matrix R_i its orientation and the 3-vector t_i its position, and X_j^H the j th 3D-feature in homogeneous coordinates. The perspective projection x_{ij} of X_j^H by the i th camera can be modeled by

$$x_{ij} \simeq K_i [R_i | t_i] X_j^H. \quad (1)$$

The symbol \simeq implies that every scalar multiple of the observed coordinates x_{ij} is a solution to this equation. Estimating all the camera parameters and the vectors X_j^H from the measurements x_{ij} at the same time is one of the oldest and most important problems in computer vision. The vectors X_j usually represent 3D-points in homogeneous coordinates, but they can also express lines, triangles, ellipses, or other features.

These days, the most widely used algorithm for structure-from-motion estimation, *bundle adjustment*, iteratively minimizes the reprojection error (The Euclidean distance between the projections of the estimated 3D-points X_j^H into the images and the measured 2D-points x_{ij}). It is regarded as the most accurate method, but it requires a good initialization and can handle only rigid scenes or rigid objects [2]. In this sense, the factorization algorithm is more flexible and better suited to problems that do not demand so high accuracy.

Theory

If the nonhomogeneous 3D-vectors X_j are projected into the images by an affine camera (Any pinhole camera is reasonably approximated by an affine camera if the object is very far away from the camera. This

implies that the depth variation within the scene is neglectable small compared with the distance between camera and scene), (Eq. 1) can be simplified to

$$x_{ij} = K_i [R_i | t_i] X_j^H = \underbrace{K_i R_i}_{P_i} X_j + K_i t_i. \quad (2)$$

As opposed to the case of a projective camera in (Eq. 1), matrix K_i is 2×3 in (Eq. 2).

The vectors x_{ij} are collected into a measurement matrix for all m images and the n 3D-features

$$\begin{aligned} W &= \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{bmatrix} = \underbrace{\begin{bmatrix} P_1 \\ \vdots \\ P_m \end{bmatrix}}_P \underbrace{\begin{bmatrix} X_1 & \cdots & X_n \end{bmatrix}}_X \\ &+ \underbrace{\begin{bmatrix} K_1 t_1 \\ \vdots \\ K_m t_m \end{bmatrix}}_t [1 \quad \cdots \quad 1]. \end{aligned} \quad (3)$$

Since matrices P and X both have rank 3 in general (Matrix P only has three columns and X only three rows; cf. (Eq. 2).) and the matrix given by the product of t and $[1 \quad \cdots \quad 1]$ has rank 1, W cannot have more than rank 4, because each column of W is a linear combination of the columns of P and t .

The world coordinate system can be arbitrarily defined, so let us place it in such a way that the centroid of all X_i is at the coordinate origin, i.e., $\sum_{j=1}^n X_j = 0$. Then, the translation $K_i t_i$ is the centroid of all the observations of the i th image:

$$K_i t_i = \frac{1}{n} \sum_{j=1}^n x_{ij}, \quad (4)$$

Hence, if $K_i t_i$ is subtracted from x_{ij} to define

$$x'_{ij} = x_{ij} - K_i t_i, \quad (5)$$

the resulting measurement matrix is written as $W' = PX$, which has rank 3. The matrix W' can be factorized using the singular value decomposition in the form

$$W' = U \Sigma V^\top = U (\Sigma V^\top). \quad (6)$$

The original factorization algorithm in [1] decomposes W' in the form $(U\Sigma^{1/2})(\Sigma^{1/2}V^\top)$, but the decomposition in (Eq. 6) is preferred for numerical reasons and simplicity of notation. Due to the rank-3 constraint, only the first three singular values on the diagonal of the matrix Σ are nonzero in the absence of noise. It therefore suffices to take only the corresponding three vectors in U and V and truncate Σ accordingly. The matrices U and (ΣV^\top) define distorted camera parameters and a distorted 3D-structure, respectively, since singular value decomposition does not impose that each row-tuple of U are rows of a rotation matrix. Therefore, the computed camera parameters and the reconstructed 3D-structure are affinely distorted and need to be corrected in a subsequent step.

Since any nonsingular 3×3 matrix A may be inserted into the decomposition of W' ,

$$W' = UAA^{-1}(\Sigma V^\top), \quad (7)$$

without altering W' , matrix A can be determined as second step of the factorization algorithm. Denote by $I_{3 \times 3}$ the 3×3 identity matrix. As in the case of self-calibrating a perspective camera by means of the dual absolute quadric, matrix A is determined from (Eq. 2) such that the two rows U_i of U corresponding to image i equal $K_i R_i$

$$1 = [H_{11}^{11} \quad H_{22}^{11} \quad H_{33}^{11} \quad 2H_{12}^{11} \quad 2H_{13}^{11} \quad 2H_{23}^{11}] t_6, \quad (12)$$

$$1 = [H_{11}^{22} \quad H_{22}^{22} \quad H_{33}^{22} \quad 2H_{12}^{22} \quad 2H_{13}^{22} \quad 2H_{23}^{22}] t_6, \quad (13)$$

$$0 = [H_{11}^{12} \quad H_{22}^{12} \quad H_{33}^{12} \quad (H_{12}^{12} + H_{21}^{12}) \quad (H_{13}^{12} + H_{31}^{12}) \quad (H_{23}^{12} + H_{32}^{12})] t_6. \quad (14)$$

Here, H_{ab} denotes the (a, b) th entry of matrix H . For simplicity, the subscript i denoting the image is omitted.

After estimating t_6 from the three equations (Eq. 12)–(Eq. 14) per image using normal equations, the correcting matrix A can be determined by eigenvalue decomposition $T = UDU^\top$ as $A = UD^{1/2}$. The matrix of camera parameters P can be taken as $P = UA$ and the 3D-structure as $X = A^{-1}\Sigma V^\top$.

Extensions

The original factorization algorithm was extended to more sophisticated affine camera models. For the projective camera model, noniterative [3] and

$$\begin{aligned} K_i R_i &= U_i A \\ (K_i R_i)(K_i R_i)^\top &= (U_i A)(U_i A)^\top \\ \underbrace{K_i R_i R_i^\top}_{I_{3 \times 3}} \underbrace{K_i^\top}_{T} &= U_i \underbrace{A A^\top}_{T} U_i^\top \\ K_i K_i^\top &= U_i T U_i^\top. \end{aligned} \quad (8)$$

For the simplest affine camera, the *orthographic* camera, the matrices K_i consist of the first two rows of a 3×3 identity matrix, i.e., $K_i K_i^\top$ equals the 2×2 identity matrix for all images. Hence, if u_{i1}^\top and u_{i2}^\top are the first and the second rows of U_i , respectively, the following three equations are obtained:

$$1 = u_{i1}^\top T u_{i1}, \quad (9)$$

$$1 = u_{i2}^\top T u_{i2}, \quad (10)$$

$$0 = u_{i1}^\top T u_{i2} = u_{i2}^\top T u_{i1}. \quad (11)$$

Equations 9–Eq. 11 constitute three linear equations in the six unknown entries of the symmetric matrix T per image. If we define $t_6 = [T_{11} \quad T_{22} \quad T_{33} \quad T_{12} \quad T_{13} \quad T_{23}]^\top$ where T_{ab} denotes the entry of T in the a th row and the b th column, and $H^{11} = u_{i1} u_{i1}^\top$, $H^{22} = u_{i2} u_{i2}^\top$, $H^{12} = u_{i1} u_{i2}^\top$, (Eq. 9)–(Eq. 11) can be written as

iterative [4, 5] extensions exist. The latter alternate matrix decomposition with refinement of the *projective depths* λ_{ij} , scalar variables are chosen so that

$$\lambda_{ij} x_{ij} = K_i [R_i \mid t_i] X_j^H \quad (15)$$

holds true under perspective projection. In [6], a projective extension is proposed which minimizes the reprojection error to obtain projectively distorted estimates of shape and motion parameters. The reprojection error is a geometric error measure instead of the algebraic error which is minimized by factorization schemes. There also exist algorithms which can handle missing observations, i.e., not all observations x_{ij} need to be known.

A generalization of the affine factorization algorithm to the *kinematic chain* model was proposed in [7]. An algorithm for scenes or objects which generally deform nonrigidly was introduced in [8]. Using prior knowledge on the shape and pose variety of human bodies, a projective factorization algorithm was proposed in [9] which can also handle missing observations.

Open Problems

Many unsolved problems remain in all variants of the factorization scheme. One of the most prominent difficulties stems from the requirement that all observations of the measurement matrix must be known. For the affine factorization scheme, the matrix decomposition step can be solved by an alternating projection method [10], yet such algorithms turn out to be unstable for the projective camera model and can therefore handle only small amounts of unknown observations. More robust algorithms combine different error metrics [11, 12] or use strong prior knowledge [9].

For iterative extensions, the problem of initialization reappears. If arbitrarily initialized, convergence to a reasonable local minimum is not guaranteed, so the resulting reconstructions can be strongly distorted.

Algorithms on nonrigid scenes or objects can handle small nonrigid deformations, unless restrictive prior knowledge is imposed. For the projective camera model without further prior knowledge, no successful 3D-reconstruction methods have been proposed which can reconstruct nonrigidly deforming scenes unless special devices, such as stereo cameras, are used or partially rigid scenes are assumed.

Lastly, the decomposition of matrix T is not possible if some eigenvalues are negative.

References

1. Tomasi C, Kanade T (1992) Shape and motion from image streams under orthography: a factorization method. *Int J Comput Vis* 9(2):137–154
2. Frahm J, Fite-Georgel P, Gallup D, Johnson T, Raguram R, Wu C, Jen Y, Dunn E, Clipp B, Lazebnik S, Pollefeys M (2010) Building rome on a cloudless day. In: Proceedings of the 11th European conference on computer vision (ECCV), Heraklion, Crete, Greece, pp 368–381
3. Sturm PF, Triggs B (1996) A factorization based algorithm for multi-image projective structure and motion. In: The proceedings of the fourth European conference on computer vision (ECCV), vol 2. Springer, London, pp 709–720
4. Heyden A, Berthilsson R, Sparr G (1999) An iterative factorization method for projective structure and motion from image sequences. *Int J Comput Vis* 17(13):981–991
5. Mahamud S, Hebert M (2000) Iterative projective reconstruction from multiple views. In: The proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), vol 2. Hilton Head, SC, USA, pp 430–437
6. Hung YS, Tang WK (2006) Projective reconstruction from multiple views with minimization of 2d reprojection error. *Int J Comput Vis* 66(3):305–317
7. Yan J, Pollefeys M (2008) A factorization-based approach for articulated nonrigid shape, motion and kinematic chain recovery from video. *IEEE Pattern Anal Mach Intell (PAMI)* 30(5):865–877
8. Bregler C, Hertzmann A, Biermann H (2000) Recovering non-rigid 3d shape from image streams. In: IEEE computer vision and pattern recognition (CVPR), Hilton Head, pp 690–696
9. Hasler N, Ackermann H, Rosenhahn B, Thormählen T, Seidel H (2010) Multilinear pose and body shape estimation of dressed subjects from image sets. In: IEEE conference on computer vision and pattern recognition (CVPR), San Francisco
10. Wold H (1966) Estimation of principal components and related models by iterative least squares. In: Krishnaiah PR (ed) Multivariate analysis. Academic, New York, p p 391–420
11. Martinec D, Pajdla T (2002) Structure from many perspective images with occlusions. In 7th European conference on computer vision (ECCV), Copenhagen, pp 542–544
12. Ackermann H, Rosenhahn B (2010) A linear solution to 1-dimensional subspace fitting under incomplete data. In: Asian conference on computer vision (ACCV) Queenstown, NZ

Feature Reduction

► Feature Selection

Feature Selection

Rama Chellappa¹ and Pavan Turaga²

¹Department of Electrical Engineering and Computer Engineering and UMIACS, University of Maryland, College Park, MD, USA

²Department of Electrical and Computer Engineering, Center for Automation Research University of Maryland, College Park, MD, USA

Synonyms

Dimensionality reduction; Feature reduction; Variable selection

Definition

Feature selection refers to a set of techniques for automatically extracting important features from raw observations, often in a task-dependent manner.

Background

Since early works in pattern analysis [1], there has been interest in extracting parsimonious and meaningful features from raw data for tasks such as recognition, compression, etc. While features based on domain knowledge prove useful, automatic methods for feature selection that can find optimal transformations of raw data are of particular interest. Toward this, linear and nonlinear methods for feature selection have been suggested. Invertible transforms are of particular interest when one is interested in obtaining the original signal, or its approximation, from features. Task specific criteria used for feature selection range from minimizing the reconstruction error to maximizing the separability between classes for classification tasks.

Theory

The basic intuition behind feature selection methods is to extract lower-dimensional information from high-dimensional data such as images, videos, etc., in a domain-dependent or task-specific manner. This is approached by restricting the complexity of the feature selection operator, such as linear or nonlinear feature selection, and then formulating a task-dependent criterion function that measures the quality of the obtained features. Then, the problem is cast as optimizing the criterion function over the set of admissible feature selection operators.

Given a set of data points denoted by $\mathcal{X} = \{x_1, x_2, \dots, x_k\}$ where $x_i \in \mathbb{R}^n$, optionally along with lower-dimensional attributes or labels $l_i = \mathcal{L}(x_i)$ associated to each x_i , a set of admissible operators Φ , where for any $\phi \in \Phi$, $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^d$, is a mapping from the raw data space to a feature space of dimension $d < n$, and a task-specific criterion function $\mathcal{E}(\mathcal{X}, \mathcal{L}, \phi)$, the goal of feature selection is to find a mapping $\phi \in \Phi$ such that the criterion function \mathcal{E} is optimized. In computer vision, the x'_i 's typically are

images or other statistics of images; the labels l'_i 's can be discrete-valued such as names of objects and identities of people, etc., or continuous-valued such as the pose of an object, the age of a person, etc.

Linear methods restrict ϕ to the set of linear transforms, which can be represented as a matrix multiplication given by $y = \phi(x) = \mathbf{W}^T x$, where \mathbf{W} is a $n \times d$ matrix. Examples of linear feature selection methods include principal component analysis (PCA) [1], independent component analysis (ICA) [2], Fisher's linear discriminant analysis (FLDA) [3], support vector machines (SVM) [4], partial least squares (PLS) [5], boosting [6], etc. Nonlinear extensions of linear techniques are commonly achieved by means of the kernel method [7].

Examples

Principal Component Analysis: In PCA, the goal is to obtain linear projections that allow optimal reconstruction of data as measured in terms of the reconstruction error. Thus, given a set of data points \mathcal{X} , the goal is to obtain a $n \times d$ orthonormal matrix \mathbf{W} , such that $\mathcal{E}(\mathcal{X}, \mathbf{W}) = \sum_{i=1}^k \|x_i - \mu - \mathbf{W}\mathbf{W}^T x_i\|^2$, is minimized, where μ is the mean of the set \mathcal{X} . For this case, the optimal \mathbf{W} has columns which are the top d eigenvectors of the data covariance matrix [8, 9].

Fisher's Discriminant Analysis: In FLDA, along with a set of points $\{x_i\}$, one is given a set of discrete class labels $l_i \in \{1, 2, \dots, c\}$. The goal is to obtain linear projections of the data so that separation between classes is increased with respect to the spread within each class. This is measured in terms of within-class and between-class scatter matrices. Denote $D_i = \{x \in \mathcal{X} | \mathcal{L}(x) = i\}$, the subset of points belonging to the i th class, and m and m_i as the mean of entire set \mathcal{X} and the i th class, respectively. The withinclass scatter is defined as $S_W = \sum_{i=1}^c S_i$, where $S_i = \sum_{x \in D_i} (x - m_i)(x - m_i)^T$, and the between-class scatter is defined as $S_B = \sum_{i=1}^c n_i(m_i - m)(m_i - m)^T$, where n_i is the cardinality of D_i . Then, the criterion function to be maximized is $\mathcal{E}(\mathcal{X}, \mathcal{L}, \mathbf{W}) = \frac{|\mathbf{W}^T S_B \mathbf{W}|}{|\mathbf{W}^T S_W \mathbf{W}|}$. For this case, the optimal \mathbf{W} has columns which are the top d generalized eigenvectors of $S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i$ [9].

Projection Pursuit: Projection pursuit is a broad class of techniques for exploratory data analysis

where the goal is to find projections of data along interesting directions. This is typically stated as searching for directions of non-Gaussianity. Criterion functions to measure non-Gaussianity include entropy and higher-order moments such as the kurtosis. For the case of entropy, the Gaussian distribution has the largest entropy among the class of zero mean and unit variance densities. Thus, to maximize non-Gaussianity, the criterion function to be maximized is $E[f(\mathbf{W}^T \mathbf{x}) \log f(\mathbf{W}^T \mathbf{x})]$ (the negative entropy), where f is the estimated probability density function of the projections [10].

Application

Feature selection has found very wide applications in face recognition [11–13]. Kernelized versions of PCA, LDA [14], and SVMs [15] have been successful in face recognition. Similar applications have been proposed in numerous object recognition tasks. Recently, the method of partial least squares has found successful application in human detection and face recognition [16, 17].

Open Problems and Recent Trends

Feature-Selection for Non-Euclidean Manifolds: The techniques for feature selection described above are implicitly designed to operate on Euclidean spaces. The definitions of various entities used – L_2 distortion measure, the mean and covariances – are valid only for Euclidean spaces. In computer vision, there are several applications where the data space is a non-Euclidean manifold embedded in a larger ambient Euclidean space. Examples of analytical manifolds that appear in vision include the hypersphere, Hilbert sphere, Grassmann manifold, covariance matrices, etc. Applications of these manifolds in vision include shape analysis [18], human detection and tracking [19], and activity analysis [20]. Often, it is possible to extend feature-selection concepts to manifolds by taking recourse to Riemannian geometry. The significant modification lies in redefining the notions of distances in terms of manifold geodesics and statistical quantities such as means and covariances in terms of intrinsic statistics [21, 22]. For example, the counterpart of PCA for manifolds is called principal geodesic analysis (PGA) [23].

However, these procedures are often approximations, and it is very hard in general to quantify their degree of accuracy.

Further, in many cases, the geometry of the manifold is not known analytically, but one has access to several samples from the manifold. In this setting, one first takes recourse to manifold learning techniques such as LLE [24], Isomaps [25], Laplacian eigenmaps [26], etc. Manifold learning techniques aim to find an embedding of the manifold to a Euclidean space. Since any such embedding results in distortions to the geometric properties of the manifold, different algorithms aim to preserve different properties of the original manifold. Once an acceptable embedding into a Euclidean space is found, feature selection can proceed using one of the existing algorithms mentioned before. However, the number and density of samples available on the manifold strongly affects the quality of features obtained.

Feature Selection and Sensing: There has been recent interest in designing sensors and cameras that can directly sense the required task-dependent features instead of sensing first and then performing feature selection. By directly sensing features, one can potentially reduce the load on the sensor without loss of performance in the given application. This is achieved in practice by modifying camera elements, such as by introducing a programmable micro mirror array [27] or optical masks [28], in a way that the required transformations of the raw data are directly sensed. However, different designs are needed to preserve different properties of the images. In parallel, there has been recent interest in sensing “universal” features by random projections. The Johnson-Lindenstrauss lemma is used as motivation for using random projections directly for inference tasks [29]. Cameras to sense random projections of data have been designed using micro-mirror arrays [30]. Though these are application agnostic, the dimensionality required to preserve the universality of these features keeps increasing with the number of data points.

References

- Hotelling H (1933) Analysis of a complex of statistical variables into principal components. *J Educ Psychol* 24:417–441
- Hyvärinen A, Oja E (2000) Independent component analysis: algorithms and applications. *Neural Netw* 13:411–430

3. Fisher RA (1936) The use of multiple measurements in taxonomic problems. *Ann Eugen* 7:179–188
4. Cortes C, Vapnik V (1995) Support-vector networks. *Mach Learn* 20(3):273–297
5. Wold H (1985) Partial least squares. In: Kotz S, Johnson NL (eds) *Encyclopedia of statistical sciences*. Wiley, New York, pp 581–591
6. Freund Y, Schapire RE (1995) A decision-theoretic generalization of on-line learning and an application to boosting. In: Proceedings of the Second European Conference on Computational Learning Theory. Barcelona, Spain, pp 23–37
7. Shawe-Taylor J, Cristianini N (2004) Kernel methods for pattern analysis. Cambridge University Press, New York
8. Jolliffe IT (1986) Principal component analysis. Springer, New York
9. Duda RO, Hart PE, Stork DG (2001) Pattern classification, 2nd edn. Wiley, New York
10. Jones MC, Sibson R (1987) What is projection pursuit? *J R Stat Soc* 150(1):1–37
11. Turk M, Pentland A (1991) Eigenfaces for recognition. *J Cognit Neurosci* 3(1):71–86
12. Etemad K, Chellappa R (1997) Discriminant analysis for recognition of human face images. *J Opt Soc Am A* 14:1724–1733
13. Bartlett MS, Movellan JR, Sejnowski TJ (2002) Face recognition by independent component analysis. *IEEE Trans Neural Netw* 13:1450–1464
14. Yang MH (2002) Kernel Eigenfaces vs. Kernel Fisherfaces: face recognition using Kernel methods. In: Proceedings of the fifth IEEE international conference on automatic face and gesture recognition, Washington DC, p 215
15. Guo G, Li SZ, Chan K (2000) Face recognition by support vector machines. In: Proceedings of the fourth IEEE international conference on automatic face and gesture recognition, Grenoble, p 196
16. Schwartz WR, Kembhavi A, Harwood D, Davis LS (2009) Human detection using partial least squares analysis. In: International conference on computer vision, Kyoto
17. Schwartz WR, Guo H, Davis LS (2010) A robust and scalable approach to face identification. In: European Conference on Computer Vision (ECCV), Crete
18. Srivastava A, Joshi SH, Mio W, Liu X (2005) Statistical shape analysis: clustering, learning, and testing. *IEEE Trans Pattern Anal Mach Intell* 27:590–602
19. Tuzel O, Porikli F, Meer P (2008) Pedestrian detection via classification on Riemannian manifolds. *IEEE Trans Pattern Anal Mach Intell* 30:1713–1727
20. Veeraraghavan A, Srivastava A, Roy Chowdhury AK, Chellappa R (2009) Rate-invariant recognition of humans and their activities. *IEEE Trans Image Process* 18:1326–1339
21. Bhattacharya R, Patrangenaru V (2003) Large sample theory of intrinsic and extrinsic sample means on manifolds-I. *Ann Stat* 31(1):1–29
22. Pennec X (2006) Intrinsic statistics on Riemannian manifolds: basic tools for geometric measurements. *J Math Imaging Vis* 25:127–154
23. Fletcher PT, Lu C, Pizer SM, Joshi S (2004) Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE Trans Med Imaging* 23:995–1005
24. Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. *Science* 290:2323–2326
25. Tenenbaum JB, de Silva V, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. *Science* 290:2319–2323
26. Belkin M, Niyogi P (2003) Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput* 15(6):1373–1396
27. Nayar SK, Branzoi V, Boult TE (2006) Programmable imaging: towards a flexible camera. *Int J Comput Vis* 70:7–22
28. Veeraraghavan A, Raskar R, Agrawal A, Mohan A, Tumblin J (2007) Dappled photography: mask enhanced cameras for heterodyned light fields and coded aperture refocusing. *ACM Trans Graph* 26:69
29. Johnson WB, Lindenstrauss J (1984) Extensions of Lipschitz mappings into a Hilbert space. In: Conference on modern analysis and probability. American Mathematical Society, Providence, RI, pp 189–206
30. Duarte MF, Davenport MA, Takhar D, Laska JN, Sun T, Kelly KF, Baraniuk RG (2008) Single-pixel imaging via compressive sampling. *IEEE Signal Process Mag* 25:83–91
31. Candes EJ, Tao T (2006) Near-optimal signal recovery from random projections: universal encoding strategies? *IEEE Trans Inf Theory* 52(12):5406–5425

Field of View

Srikumar Ramalingam

Mitsubishi Electric Research Laboratories,
Cambridge, MA, USA

Synonyms

Field of vision

Related Concepts

► [Center of Projection](#)

Definition

Field of view refers to the angular volume of 3D space sampled by the light rays of a camera.

Background

The pinhole camera is one of the most successful mathematical models in the field of computer vision, image

processing, and graphics. People naturally accepted this imaging model because of its extreme simplicity and its closeness to an image perceived by the human visual system. In a pinhole camera, the projection rays from scene points (light rays) intersect at a single point (optical center). Typically, these conventional cameras have a very small field of view, around 50°. Omnidirectional cameras have a larger field of view and have been extremely useful in several applications like videoconferencing, augmented reality, surveillance, and large-scale 3D reconstruction. These cameras can be constructed in a simple manner, for they can be made from conventional cameras by using additional lenses or mirrors. For example, Fig. 1a shows an E8 Nikon Coolpix camera appended with a fish-eye lens having a field of view of 183° × 360°. Another possibility is to use mirrors in addition to lenses to increase the field of view. These configurations are referred to as *catadioptric*, where “cata” comes from mirrors (reflective) and “dioptric” comes from lenses (refractive). Figure 1b, c show two catadioptric configurations with hyperbolic and parabolic mirrors, respectively.

Theory and Camera Models

Fish-Eye Camera Model

Fish-eye lenses have a short focal length and a very large field of view (cf. Fig. 2a). However, when the field of view is greater than 180°, the concept of focal length is not defined. For example, the focal length is not defined for the E8 fish-eye lens of Nikon which has a field of view of 183° × 360°. Several works have used fish-eye lenses for creating omnidirectional images [1–3]. Geometrically, omnidirectional cameras can be either single viewpoint or noncentral. Single viewpoint configurations are preferred to noncentral systems because they permit the generation of geometrically correct perspective images from the image(s) captured by the camera. In addition, most theories and algorithms developed for conventional cameras hold good for single-center omnidirectional cameras. In theory, fish-eye lenses do not provide a single viewpoint imaging system [4]. The projection rays pass through a small disk in space rather than a single point. Nevertheless, in practice, it is usually a good assumption to consider these cameras as single viewpoint cameras [5]. Perspective images synthesized

from fish-eye images are visibly very accurate without any distortions. Several distortion functions can be used to model fish-eye and central catadioptric images [6, 7]. Some of them are mentioned below.

– *Stereographic projection*: Several radially symmetric models [8] were used for fish-eye images. One of them is the stereographic projection. This model gives a relation between θ , the angle made by a scene point, the optical center and the optical axis, and the distance r between the associated image point and the distortion center

$$r = k \tan \frac{\theta}{2}$$

where k is the only parameter to be estimated.

– *Equidistant projection*:

$$r = k\theta$$

– *Equisolid angle projection*:

$$r = k \sin \frac{\theta}{2}$$

– *Sine law projection*:

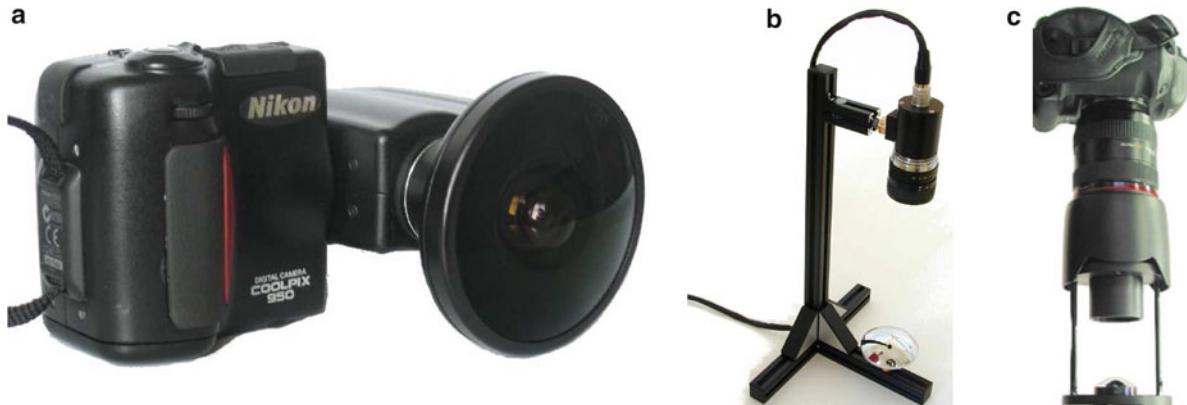
$$r = k \sin \theta$$

On fitting the Nikon FC-E8 fish-eye lens with the four radially symmetric models (stereographic, equidistant, equisolid angle, and sine law), it was found that the stereographic projection gave the lowest error [9]. The error is the Euclidean distance between the original image pixels and the projected image pixels using the models.

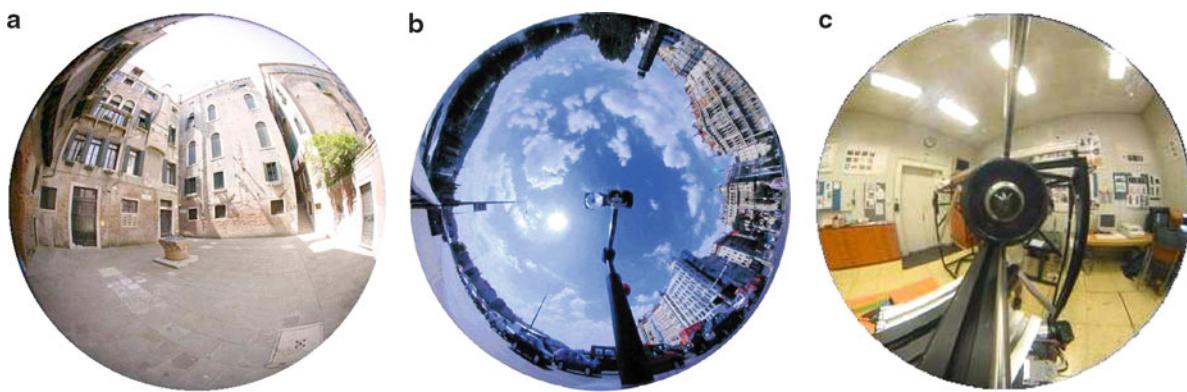
– *Combined stereographic and equisolid angle model*: In [9], Bakstein and Pajdla followed a model-fitting approach to identify the right projection model for fish-eye cameras. Their model is a combination of stereographic and equisolid angle models. The following relation was obtained with four parameters:

$$r = a \tan \frac{\theta}{b} + c \sin \frac{\theta}{d}$$

On the whole, they used 13 camera parameters: six external motion parameters (R, t), one aspect



Field of View, Fig. 1 (a) Fisheye camera (b) Catadioptric configuration using a hyperbolic mirror and a pinhole camera. (c) Catadioptric configuration using a parabolic mirror and an orthographic camera



Field of View, Fig. 2 Omnidirectional images captured by three different cameras are shown: (a) Image of a fish-eye camera (b) image of a catadioptric camera using a hyperbolic mirror

and a pinhole camera. (c) image of a catadioptric camera using a parabolic mirror and an orthographic camera. Courtesy of Branislav Micusik and Tomas Pajdla

ratio (β), two parameters for the principal point (u_0, v_0) , and the four parameters of the above projection model (a, b, c, d) .

- *Polynomial lens distortion model:* Most distortion corrections assume the knowledge of the distortion center. Let r_d refer to the distance of an image point from the distortion center. The distance of the same image point in the undistorted image is given by

$$r_u = r_d(1 + k_1 r_d^2 + k_2 r_d^4 + \dots)$$

where k_1 and k_2 are distortion coefficients [10].

- *Field of view (FOV):* The distortion function is given by

$$r_u = \frac{\tan(r_d w)}{2 \tan \frac{w}{2}}$$

The above distortion correction function is based on a single parameter w . It is a good idea to correct the distortion using the polynomial model followed by the field of view model [6].

- *Division model (DM):* The distortion correction function is given by

$$r_u = \frac{r_d}{(1 + k_1 r_d^2 + k_2 r_d^4 + \dots)}$$

where the k_i are the distortion coefficients.

Catadioptric Camera Model

Vision researchers have been interested in catadioptric cameras [4, 11–17] because they allow numerous possibilities in constructing omnidirectional cameras. The possibilities arise from the differences in size, shape,

orientation, and positioning of the mirrors with respect to the camera. Please refer to the encyclopedia entry on catadioptric camera for more details.

Application

- *Larger field of view:* Fig. 2 shows images captured by three different omnidirectional cameras. The first image is captured by a fish-eye camera, the second is captured by a catadioptric system constructed using a hyperbolic mirror and a perspective camera, and finally, the third is captured by another catadioptric camera constructed using a parabolic mirror and an orthographic camera. One can make the following observation from the omnidirectional images. A very large scene, which usually requires several pinhole images, can be captured in a single omnidirectional image although of course at a lower resolution.
- *Stable motion estimation:* Motion estimation is a challenging problem for pinhole images, especially when a larger number of images are involved. On the other hand, omnidirectional cameras stabilize the motion estimation and improves its accuracy [18–20]. In the case of small rigid motions, two different motions can yield nearly identical motion fields for classical perspective cameras. However, this is impossible in the case of omnidirectional cameras. By improving the stability of motion estimation, omnidirectional cameras also contribute to a stable 3D reconstruction.

A detailed survey of various camera models, calibration, and 3D reconstruction algorithms is given in [21, 22].

References

1. Miyamoto K (1964) Fish eye lens. J Opt Soc Am 54(8):1060–1061
2. Slater J (1932) Photography with the whole sky lens. Am Photogr
3. Wood R (1902) Fish-eye view, and vision under water. Philos Mag 12:159–162
4. Nalwa V (1996) A true omnidirectional viewer. Technical report, Bell Laboratories, Holmdel
5. Ying X, Hu Z (2004) Can we consider central catadioptric cameras and fisheye cameras within a unified imaging model. European conference on computer vision (ECCV), Prague, pp 442–455
6. Devernay F, Faugeras O (2001) Straight lines have to be straight. Mach Vis Appl 13:14–24
7. Claus D, Fitzgibbon A (2005) A rational function lens distortion model for general cameras. International conference on computer vision, vol 1, pp 213–219
8. Fleck M (1995) The wrong imaging model. Technical Report TR 95-01, University of Iowa
9. Bakstein H, Pajdla T (2002) Panoramic mosaicing with a 180 field of view lens. IEEE workshop on omnidirectional vision, Copenhagen
10. Brown D (1971) Close-range camera calibration. Photogr Eng 37(8):855–866
11. Bogner S (1995) Introduction to panoramic imaging. IEEE SMC, vol 54, pp 3100–3106
12. Charles J, Reeves R, Schur C (1987) How to build and use an all-sky camera. Astron Mag
13. Hong J (1991) Image based homing. International conference on robotics and automation
14. Murphy JR (1995) Application of panoramic imaging to a teleoperated lunar rover. IEEE SMC conference, pp 3117–3121
15. Rees DW (1970) Panoramic television viewing system. United States Patent (3,505,465)
16. Yagi Y, Kawato S (1990) Panoramic scene analysis with conic projection. International conference on robots and systems (IROS)
17. Yamazawa K, Yagi Y, Yachida M (1995) Obstacle avoidance with omnidirectional image sensor hyperomni vision. International conference on robotics and automation, pp 1062–1067
18. Brodsky T, Fernmuller C, Aloimonos Y (1996) Directions of motion fields are hardly ever ambiguous. European conference on computer vision (ECCV), vol 2, 110–128
19. Koch O, Teller S (2007) Wide-area egomotion estimation from known 3d structure. IEEE conference on computer vision pattern recognition (CVPR)
20. Ramalingam S, Bouaziz S, Sturm P, Brand M (2010) Skyline2gps: geo-localization in urban canyons using omni-skylines. IROS
21. Sturm P, Ramalingam S, Tardif JP, Gasparini S, Barreto J (2011) Camera models and fundamental concepts used in geometric computer vision. Found Trends Comput Graph Vis 6(1–2):1–183
22. Ramalingam S (2006) Generic imaging models: calibration and 3d reconstruction algorithms. PhD Thesis, INRIA Rhone Alpes

Field of Vision

- Field of View

Filling In

- Inpainting

Fisher-Rao Metric

Stephen J. Maybank

Department of Computer Science and Information Systems, Birkbeck College University of London, London, UK

Synonyms

Rao Metric

Related Concepts

►Fisher-Rao Metric; ►Maximum Likelihood Estimation

Definition

The Fisher-Rao metric is a particular Riemannian metric defined on a parameterized family of conditional probability density functions (pdfs). If two conditional pdfs are near to each other under the Fisher-Rao metric, then the square of the distance between them is approximated by twice the average value of the log likelihood ratio of the conditional pdfs.

Background

Suppose that a parameterized family of conditional pdfs is given and it is required to find the parameter value corresponding to the conditional pdf that best fits a given set of data. It is useful to have a distance function defined on pairs of conditional pdfs, such that if a given conditional pdf is a close fit to the data, then all the conditional pdfs near to it are also close fits to the data. Any such distance function should be independent of the choice of parameterization of the family of conditional pdfs and independent of the choice of parameterization of the data. The Fisher-Rao metric is the only known Riemannian metric which yields a distance function with both the required independence properties [1].

Theory

Let X be an open subset of a Euclidean space \mathbb{R}^n and let T be an open subset of a Euclidean space \mathbb{R}^d . Let x ,

θ be vectors in X and T , respectively, and let $p(x|\theta)$ be a probability density function defined for x in X and conditional on θ in T . The set T is a parameter space for the family of conditional pdfs $\theta \mapsto p(x|\theta)$. Let θ_i for $1 \leq i \leq d$ be the components of θ . With these choices of parameterization for X and T , the Fisher-Rao metric on T is defined by the following family of $d \times d$ matrices:

$$J_{ij}(\theta) = - \int_X \left(\frac{\partial^2}{\partial \theta_i \partial \theta_j} \ln p(x|\theta) \right) p(x|\theta) dx, \\ 1 \leq i, j \leq d, \theta \in T. \quad (1)$$

The same matrix $J(\theta)$ is also defined by the formulae:

$$J_{ij}(\theta) = \int_X \left(\frac{\partial}{\partial \theta_i} \ln p(x|\theta) \right) \left(\frac{\partial}{\partial \theta_j} \ln p(x|\theta) \right) p(x|\theta) dx, \\ 1 \leq i, j \leq d, \theta \in T. \quad (2)$$

Rao notes in [6] that $J(\theta)$ defines a Riemannian metric on T . For information about (Eq. 1) and (Eq. 2), see [1], Sect. 2.3.

Let $\theta + \Delta\theta$ be a point in T near to θ , and let $\text{dist}(\theta, \theta + \Delta\theta)$ be the length of the shortest path in T from θ to $\theta + \Delta\theta$, as measured using the Fisher-Rao metric. This shortest path is called the geodesic between θ and $\theta + \Delta\theta$. It can be shown that:

$$\Delta\theta^\top J(\theta)\Delta\theta = \text{dist}(\theta, \theta + \Delta\theta)^2 + O(\|\Delta\theta\|^3). \quad (3)$$

The Fisher-Rao metric is also defined when X is a finite set. Let X have n elements and let θ_i be the probability of the i th element of X . The θ_i for $1 \leq i \leq n$ are not independent because they sum to 1. Let θ be the vector formed from the probabilities θ_i for $1 \leq i \leq n-1$. The parameter space T is the open subset of \mathbb{R}^{n-1} consisting of vectors θ with components θ_i such that:

$$0 < \theta_i < 1, \quad 1 \leq i \leq n-1, \\ 0 < \sum_{i=1}^{n-1} \theta_i < 1.$$

The Fisher-Rao metric is defined on T by the following $(n-1) \times (n-1)$ matrix:

$$\begin{aligned} J_{ij}(\theta) &= \theta_n^{-1}, \quad 1 \leq i, j \leq n-1, i \neq j, \\ J_{ii}(\theta) &= \theta_n^{-1} + \theta_i^{-1}, \quad 1 \leq i \leq n-1. \end{aligned}$$

Let θ and $\theta + \Delta\theta$ be nearby points in T . Then the approximation (Eq. 3) to the square of the distance between the pdfs $p(x|\theta)$ and $p(x|\theta + \Delta\theta)$ is:

$$\sum_{i=1}^n \theta_i^{-1} (\Delta\theta_i)^2.$$

Let $D(\theta(1)\|\theta(2))$ be the Kullback-Liebler divergence between the conditional pdfs $p(x|\theta(1))$ and $p(x|\theta(2))$ [1]. In the continuous case, in which X is an open subset of \mathbb{R}^n , the Kullback-Leibler divergence is given by:

$$D(\theta(1)\|\theta(2)) = \int_X \ln \left(\frac{p(x|\theta(1))}{p(x|\theta(2))} \right) p(x|\theta(1)) dx. \quad (4)$$

On setting $\theta = \theta(1)$, $\theta + \Delta\theta = \theta(2)$, it follows from (Eq. 4) that:

$$\Delta\theta^\top J(\theta)\Delta\theta = \frac{1}{2} D(\theta\|\theta + \Delta\theta) + O(\|\Delta\theta\|^3). \quad (5)$$

An equation similar to (Eq. 5) holds when X is a finite set.

The matrix $J(\theta)$ used to define the Fisher-Rao metric appears in the theory of maximum likelihood estimation [2]. Suppose that N points $x(i)$ for $1 \leq i \leq N$ are sampled independently from X using the conditional pdf $p(x|\theta)$. Let $\hat{\theta}$ be the maximum likelihood estimate of θ :

$$\hat{\theta} = \operatorname{argmax}_{\phi} \phi \mapsto \prod_{i=1}^N p(x(i)|\phi).$$

If N is large, then the distribution of $\theta - \hat{\theta}$ is closely approximated by a Gaussian distribution with expected value 0 and covariance:

$$N^{-1} J(\theta)^{-1}.$$

Let N be any fixed positive integer, let $\phi \equiv \phi(x(1), \dots, x(N))$ be any unbiased estimator of θ , and let C be the covariance of ϕ . Then the matrix:

$$C - N^{-1} J(\theta)^{-1}$$

is positive semi-definite. The matrix $N^{-1} J(\theta)^{-1}$ is known as the Cramér-Rao lower bound for C [1].

In the continuous case, in which X is an open subset of \mathbb{R}^n , it is rare to find a closed form expression for the Fisher-Rao metric. An example of a closed form expression is provided by the family of Gaussian densities for which $X = \mathbb{R}$. Let $\theta = (\mu, t)$, such that μ, t are points in \mathbb{R} with $t > 0$. The parameter space T is the upper half plane in \mathbb{R}^2 . Let $p(x|\theta)$ be the Gaussian pdf for x in \mathbb{R} with expected value μ and standard deviation $t/\sqrt{2}$. In this case, the scaled Fisher-Rao metric, $(1/2)J(\theta)$, coincides with the Poincaré metric on T :

$$\frac{1}{2} J(\theta) = \frac{1}{t^2} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \theta \in T.$$

It is shown in [3] that under certain conditions the Fisher-Rao metric can be closely approximated by a simpler metric.

Application

The Fisher-Rao metric provides a theoretical basis for the Hough transform which is used to detect geometrical structures such as lines and circles. The size and shape of the Hough transform accumulators and the number of accumulators can be calculated using the Fisher-Rao metric [3]. Peter and Rangarajan [5] describe planar shapes using weighted sums of Gaussian pdfs. The associated Fisher-Rao metric is used to define geodesics in the parameter manifold for the shape pdfs. Each segment of a geodesic specifies a continuous family of shapes which interpolate between the two shapes represented by the end points of the segment. An algorithm to find geodesics is described in [4].

References

- Amari S-I (1985) Differential-geometric methods in statistics. Lecture notes in statistics, vol. 28. Springer, New York
- Fisher RA (1922) On the mathematical foundations of theoretical statistics. Phil Trans R Soc London Ser A 222:309–368
- Maybank SJ (2004) Detection of image structures using Fisher Information and the Rao metric. IEEE Trans Pattern Anal Mach Intell 26(12):49–62
- Mio W, Liu X (2006) Landmark representations of shapes and Fisher-Rao geometry. In: Proceedings of the IEEE

- international conference on image processing, Atlanta. IEEE, pp 2113–2116
5. Peter A, Rangarajan A (2009) Information geometry for landmark shape analysis: unifying shape representation and deformation. *IEEE Trans Pattern Anal Mach Intell* 31(2):337–350
 6. Rao C (1945) Information and accuracy attainable in the estimation of statistical parameters. *Bull Calcutta Math Soc* 37:81–91

Fisheye Camera

► [Omnidirectional Camera](#)

Fisheye Lens

Peter Sturm
INRIA Grenoble Rhône-Alpes, St Ismier Cedex,
France

Related Concepts

► [Omnidirectional Vision](#)

Definition

A fisheye lens is a lens giving a field of view of about 180° or larger.

Background

The terms fisheye camera and fisheye view seem to have been introduced by Wood in 1906 [1]. Wood was interested in the way a fish perceives objects outside the water. Besides studying the problem theoretically, he also built a camera that mimics the fisheye view. To do so, he immersed a pinhole camera in a casing filled with water and that had a glass plate as one of its faces, through which the camera could acquire images of the outside world. One basic observation Wood made is that since the camera looks from a denser medium (water) into a lighter one (air), its effective field of view is larger than its native one, due to the refraction happening at the interface between the media. This effect is related to the so-called Snell's window. In particular, when looking from water into air and supposing that

the water surface is still, the entire hemisphere above the water can be seen within a circular cone-shaped field of view with an opening angle of about 96°.

In the 1920s, Bond and Hill independently invented purely glass-based fisheye lenses that, like Wood's water-based camera, gave hemispherical fields of view [2, 3]. Many improvements were made subsequently, for instance, on chromatic aberrations; see, for example, [4–6]. Fisheye lenses can achieve larger than hemispheric fields of view, for example, Martin reports a design with a 310° field of view [6]. An example of a fisheye lens and an image acquired using one is given in Fig. 1.

Theory

Since fisheye cameras capture an entire hemisphere or more in a single image, the image is bound to show strong distortions. These cameras can thus not be modeled using a pinhole model and even classical polynomial distortion models are insufficient. Better suited models have been proposed [4, 7, 8], as follows. Let ϕ be the angle between the optical axis and an incoming light ray and θ the angle between the optical axis and the ray leaving the lens towards the image plane. Then, we can make the following definitions:

$$\text{Perspective projection: } \theta = \phi$$

$$\text{Stereographic projection: } \tan \theta = k \tan \frac{\phi}{2}$$

$$\text{Equidistant projection: } \tan \theta = k\phi$$

$$\text{Equi-solid angle projection: } \tan \theta = k \sin \frac{\phi}{2}$$

$$\text{Sine-law projection: } \tan \theta = k \sin \phi,$$

where k is a free parameter.

It is sometimes useful to express these projection models with respect to the distance r of an image point from the principal point:

$$\text{Perspective projection: } r = m \tan \phi$$

$$\text{Stereographic projection: } r = m \tan \frac{\phi}{2}$$

$$\text{Equidistant projection: } r = m\phi$$

$$\text{Equi-solid angle projection: } r = m \sin \frac{\phi}{2}$$

$$\text{Sine-law projection: } r = m \sin \phi,$$



Fisheye Lens, Fig. 1 Two images of a fisheye conversion lens and an image taken with a fisheye lens. The image is necessarily heavily distorted since it “contains” an entire hemispheric field of view

where m is a free parameter, proportional to the camera’s focal length.

Various other models for fisheye lenses and other omnidirectional cameras have been proposed in the literature, for instance [5, 9–12]. These and other models are described in [13], which also provides references to calibration methods. It seems that most fisheye lenses are designed to approach the equidistant model. In practice, an accurate calibration of a fisheye camera may require to add a classical polynomial distortion model “on top” of a specific fisheye projection model.

Application

Among the first applications of fisheye lenses were meteorology, via the study of cloud formations, and forest management, via the assessment of leaf coverage via fisheye images of forest canopies. Other applications are the same as those of other omni-directional cameras, where a wide field of view is beneficial, for instance in mobile robotics or video surveillance.

References

- 1. Wood R (1906) Fish-eye views, and vision under water. *Philos Mag* 6(12):159–162
- 2. Bond W (1922) A wide angle lens for cloud recording. *Philos Mag* 44(263):999–1001
- 3. Hill R (1924) A lens for whole sky photographs. *Q J R Meteorol Soc* 50(211):227–235
- 4. Miyamoto K (1964) Fish eye lens. *J Opt Soc Am* 54(8):1060–1061
- 5. Kumler J, Bauer M (2000) Fish-eye lens designs and their relative performance. In: Proceedings of SPIE conference on current developments in lens design and optical systems engineering, San Diego, USA, pp 360–369
- 6. Martin C (2004) Design issues of a hyper-field fisheye lens. In: Sasián J, Koschel R, Manhart P, Juergens R (eds) Proceedings of SPIE conference on novel optical systems design and optimization VII, Bellingham, WA, vol 5524, pp 84–92
- 7. Beck C (1925) Apparatus to photograph the whole sky. *J Sci Instrum* 2(4):135–139
- 8. Fleck M (1995) Perspective projection: The wrong imaging model. Technical Report TR 95–01, Department of Computer Science, University of Iowa, Iowa City, IA 52242, USA
- 9. Herbert T (1986) Calibration of fisheye lenses by inversion of area projections. *Appl Opt* 25(12):1875–1876
- 10. Gennery D (2006) Generalized camera calibration including fish-eye lenses. *Int J Comput Vis* 68(3):239–266
- 11. Bakstein H, Pajdla T (2002) Panoramic mosaicing with a 180° field of view lens. In: Proceedings of the workshop on omnidirectional vision, Copenhagen, Denmark, pp 60–68
- 12. Devernay F, Faugeras O (2001) Straight lines have to be straight. *Mach Vis Appl* 13(1):14–24
- 13. Sturm P, Ramalingam S, Tardif JP, Gasparini S, Barreto J (2011) Camera models and fundamental concepts used in geometric computer vision. *Found Trends Comput Graph Vis* 6(1–2):1–183
- 14. Wikipedia (2011) Fisheye lens. http://en.wikipedia.org/wiki/Fisheye_lens. Accessed 3 Aug 2011

Fluorescent Lighting

Stephen Lin

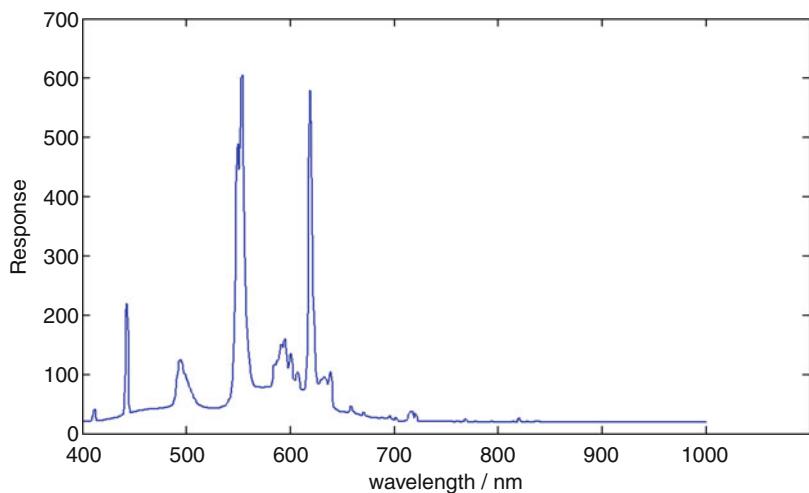
Microsoft Research Asia, Beijing Sigma Center,
Beijing, China

Definition

Fluorescent lighting is the illumination of a scene by the fluorescence of phosphor in a gas-discharge lamp.

Fluorescent Lighting, Fig. 1

A typical color spectrum of fluorescent light



Background

In fluorescent lighting, electrical energy is converted into radiant energy by a physical process in the fluorescent lamp. The lamp consists of a sealed glass tube containing mercury vapor, an inert gas (such as argon) at low pressure, and a phosphor coating on the inside surface. After ionizing the gas, electrical current flows through the gas between electrodes at the ends of the tube. The current excites the mercury atoms, which then emit ultraviolet light. Ultraviolet (UV) light is not visible to the human eye, but is used to cause fluorescence of the phosphor, which absorbs the UV radiation and produces light in the visible range of the color spectrum. Additional details on fluorescent lamp operation can be found in [3].

Fluorescent light has certain properties that distinguish it from other forms of illumination. These include a color spectrum with sharp peaks that correspond to the chemical composition of the phosphor, and flicker at twice the frequency of the alternating current. Some computer vision algorithms are designed to take advantage of these properties for particular purposes.

Application

The color spectrum of fluorescent light contains peaks as exemplified in Fig. 1. These peaks originate from energy emission of the phosphor, and lie at specific wavelengths according to the phosphor composition. The correspondence of spectrum peaks to particular

wavelengths has been utilized in computer vision for calibration of multispectral sensing devices based on dispersive optics [1]. In [1], the spectrum peaks are also used to identify the presence of fluorescent lighting in a scene, as the characteristic peaks are detectable even in light reflected from object surfaces.

Fluorescent lighting also exhibits a regular, high-frequency flicker due to the cycles of electrical current flow in fluorescent lamps. This stroboscopic effect is used in the Mova ContourTM system for markerless motion tracking of human faces [2, 4]. The face is covered with a phosphorescent makeup and then illuminated by the flicker of a blacklight fluorescent lamp, which uses a phosphor that converts the shortwave UV radiation of the mercury vapor to a long-wave UV light that stimulates the makeup. At intervals of the flicker when the fluorescent lighting is off, emission from the phosphorescent makeup is recorded by multiple cameras. The random patterns of makeup formed by a rough applicator sponge are tracked and also triangulated to form 3D models.

References

1. Du H, Tong X, Cao X, Lin S (2009) A prism-based system for multispectral video acquisition. In: Proceedings of the international conference on computer vision. IEEE, Kyoto, Japan
2. Geller T (2008) Overcoming the uncanny valley. IEEE Comput Graph Appl 28(4):11–17
3. Henkenius M (1991) How it works: fluorescent lamp. Pop Mech 10:59–60
4. King BA, Paulson LD (2007) Motion capture moves into new realms. IEEE Comput 40(9):13–16

Focal Length

Peter Sturm
INRIA Grenoble Rhône-Alpes, St Ismier Cedex,
France

Synonyms

[Principal distance](#)

Related Concepts

[►Center of Projection](#); [►Camera Calibration](#); [►Image Plane](#); [►Optical Axis](#); [►Pinhole Camera Model](#)

Definition

The focal length has different, related, meanings. In optics, the focal length of a lens or optical system is the distance from the center to the point on the optical axis where a bundle of incoming rays parallel to the optical axis get focused to. In geometric image formation models such as the pinhole model, the focal length usually represents the distance between the center of projection and the image plane.

Background

The concept of focal length stems from the area of optics. The focal length of a lens is generally defined as the distance from the center of the lens to the point where a set of incoming parallel light rays that are parallel to the optical axis get focused. It is thus related to the magnification operated by an optical system.

In computer vision, the image formation process carried out by the optics and electronics of a camera is usually modeled by simple geometric models.

In the following, the thin lens model and the pinhole model are discussed. More general background is given, for example, in [1–3].

Theory

Thin lens model. Consider first a simple “physical” model for a lens, the thin lens model, for the case of a

lens whose two outer surfaces are convex and spherical. For simplicity, we assume here that the two spheres have the same radius R . Let n be the index of refraction of the lens material and n_0 that of the surrounding medium (for a vacuum $n_0 = 1$, for air $n_0 \approx 1.0008$).

Incoming rays get bent by the lens, due to the successive refractions in the two surfaces of the lens. For lenses with spherical surfaces, even incoming rays that are parallel to the optical axis do not converge to the same point on the optical axis after these refractions. Rather, they hit the optical axis in a segment; this is known as spherical aberration; see [Fig. 1](#).

Let us now make two common approximations. First, under the thin lens approximation, one assumes that both refractions happen at the same point, on the lens’ plane instead of on the two spherical surfaces. Second, we make the so-called paraxial approximation by assuming that along the path of the refracted light ray, all angles it forms with the optical axis and the normals of the spherical surfaces are small (leading, e.g., to the approximation $\sin \alpha \approx \alpha$ for such angles). Under these two simplifying assumptions, all incoming rays that are parallel to the optical axis are focused by the lens to a point on the optical axis that is at the following distance from the lens center:

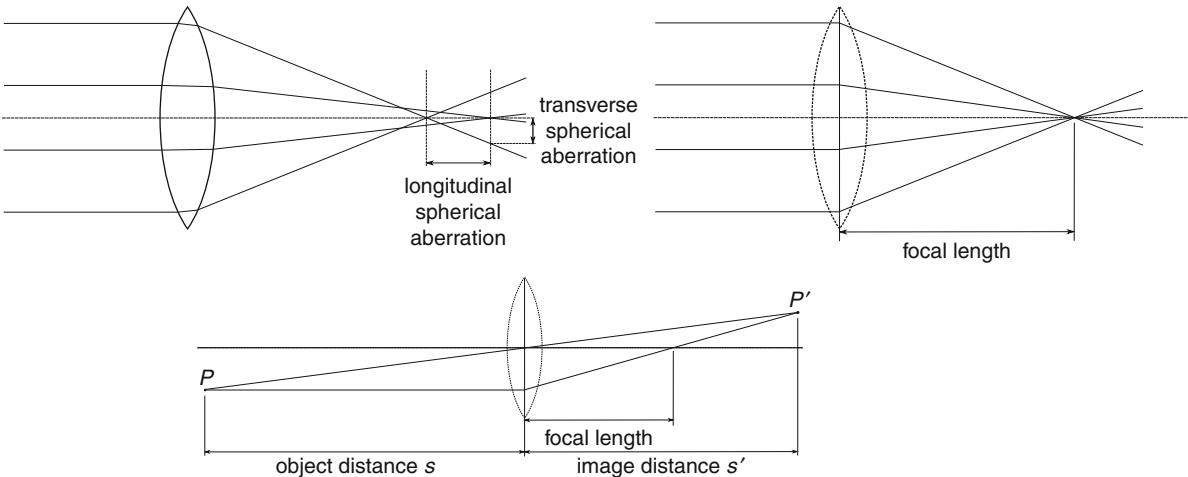
$$f = \frac{R n_0}{2(n - n_0)} .$$

This distance is the focal length of the lens, under the paraxial thin lens approximation. This formula can be generalized to lenses with two spherical surfaces of different radii, in the form of the so-called lensmaker’s equation [4, 5]. Lenses with other bounding shapes than spherical ones exist of course and can be studied similarly [2].

Under the above assumptions, if one wishes to take a sharp picture of a distant object, one would put the image plane at a distance of f from the lens. However, if the object is at a close distance s from the lens, then the light rays emerging from it converge at a distance s' from the lens that is different from f and that is given by [2, 6]:

$$\frac{1}{s} + \frac{1}{s'} = \frac{1}{f} .$$

This classical relationship can be easily derived by considering similar triangles and the following two light rays emitted from a point P at a distance s from the lens; cf. the lower part of [Fig. 1](#). The ray parallel



Focal Length, Fig. 1 *Upper left:* spherical aberration of a lens with two spherical surfaces. *Upper right:* focal length in the paraxial thin lens approximation. *Bottom:* image of an object point P at finite distance from the lens, under the paraxial thin lens model

to the optical axis (supposing here it enters the aperture) intersects the optical axis at a distance f from the lens, after being bent by it, as explained above. The ray going through the lens center does not get bent under the thin lens assumption. The two rays thus converge in a point P' at a distance of s' from the lens plane. To get a sharp picture, one would thus put the image plane at distance s' .

Pinhole model. In the pinhole model and other camera models, the focal length is defined differently than in optics, as the distance between the center of projection and the image plane. As seen above, this is in general different from the focal length of optical systems, even in the very simple case of the paraxial thin lens model thereof. The two definitions coincide however if a camera is focused to infinity and if the center of the thin lens model is considered as center of projection in the pinhole model.

The focal length of the pinhole model is often expressed in the nonmetric unit “number of pixels” (true focal length divided by the density of pixels). For real cameras, the focal length is usually given in millimeters. An often used convention is to characterize a camera by the focal length that an equivalent 35-mm-format camera would have: the focal length of a lens that, if used with a 35-mm-format image area, would have the same field of view as the camera under consideration.

Application

The focal length of the pinhole or other camera models typically used in computer vision is part of the camera’s intrinsic parameters, which can be computed by camera calibration. Camera calibration is an important requirement in most applications where geometric information about the scene or the camera movement is to be determined from images.

The difference between the meaning of the pinhole model’s focal length and that of a true optical system made of lenses (the pinhole model is lensless) is stressed again here. A consequence of what is explained above is that when calibrating a camera using the pinhole model, that model’s focal length is affected by both a change in focus and zoom of the actual camera [7].

References

1. Wikipedia (2011) Focal length. http://en.wikipedia.org/wiki/Focal_length. Accessed 3 August 2011
2. Hecht E (2001) Optics. 4th edn. Addison, Wesley
3. Geissler P (2000) Imaging optics. In: Jähne B, Haußecker H (eds) Computer vision and applications. Academic, San Diego, pp 53–84
4. HyperPhysics (2011) Light and vision. <http://hyperphysics.phy-astr.gsu.edu/hbase/ligcon.html#c1>. Accessed 4 August 2011
5. Wikipedia (2011) Lens (optics). [http://en.wikipedia.org/wiki/Lens_\(optics\)](http://en.wikipedia.org/wiki/Lens_(optics)). Accessed 4 August 2011

6. Wikipedia (2011) Thin lens. http://en.wikipedia.org/wiki/Thin_lens. Accessed 4 August 2011
7. Willson R, Shafer S (1993) Modeling and calibration of zoom lenses. In: Gruen A, Huang T (eds) Camera calibration and orientation determination. Springer, Verlag, pp 137–161

Focus Bracketing

► [Multi-focus Images](#)

Foreground-background Assignment

► [Occlusion Detection](#)

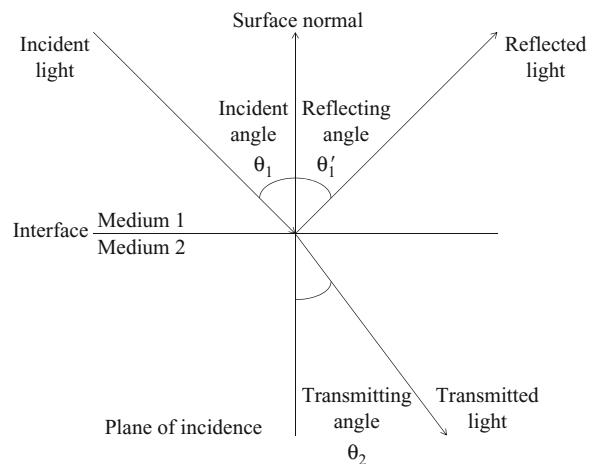
Fresnel Conditions

► [Fresnel Equations](#)

Fresnel Equations

Daisuke Miyazaki

Graduate School of Information Sciences, Hiroshima City University, Asaminami-ku, Hiroshima, Japan



Fresnel Equations, Fig. 1 Reflection, refraction, and transmission at the interface of two materials. The surface is assumed to be optically smooth

Background

Based on the theory of electromagnetism, the Fresnel equations express the reflection and transmission coefficients of light that hits an interface between two media. This entry introduces amplitude reflectivity, amplitude transmissivity, intensity reflectivity, and intensity transmissivity.

Theory

Figure 1 illustrates a light ray that hits the interface between two materials, the refractive indices of which are denoted by n_1 and n_2 , respectively. Part of the light is reflected from the interface, while another part penetrates the surface and refracts as it enters the second material. The plane including the surface normal and the incident light ray is called the plane of incidence (POI). The incident light, the reflected light, and the transmitted light are denoted as the subscripts i , r , and t , respectively.

Being an electromagnetic wave, light carries an oscillating electric field. The oscillating field (called E-vector) has amplitude components that are parallel or perpendicular to the POI. These components are denoted by p and s , respectively. Here, p is associated with the term “parallel,” while s is associated with the word “senkrecht,” which means “perpendicular” in German. The incidence, reflection, and transmission

Synonyms

Fresnel Conditions; Fresnel’s law; Light Transmission and Reflection Coefficients

Related Concepts

► [Polarized Light in Computer Vision](#); ► [Polarizer](#); ► [Polarization](#)

Definition

The four Fresnel equations express the reflection and transmission coefficients of light components whose electric-field vector is either parallel or perpendicular to the plane of incidence.

angles are defined as θ_1 , θ'_1 , and θ_2 , respectively, as illustrated in Fig. 1.

For optically smooth objects, the incidence and reflection angles are equal, $\theta_1 = \theta'_1$, while θ_1 and θ_2 are related by Snell's law (cf. Sect. 1.5.1 in the 5th edition of Born and Wolf [1], Sect. 4.4.1 in the 4th edition of Hecht [2]):

$$n_1 \sin \theta_1 = n_2 \sin \theta_2. \quad (1)$$

The ratio of the amplitude of the reflected light to that of the incident light is called reflection coefficient (or, amplitude reflectivity), r . The ratio of the amplitude of the transmitted light to that of the incident light is called transmission coefficient (or, amplitude transmissivity), t . These coefficients are generally different for the two E-vector components. The coefficients for the p -component and s -component are derived from the theory of optics as (cf. Sect. 1.5.2 in the 5th edition of Born and Wolf [1], Sect. 4.6.2 in the 4th edition of Hecht [2]):

$$r_p = \frac{\tan(\theta_1 - \theta_2)}{\tan(\theta_1 + \theta_2)} \quad (2)$$

$$r_s = -\frac{\sin(\theta_1 - \theta_2)}{\sin(\theta_1 + \theta_2)} \quad (3)$$

$$t_p = \frac{2 \sin \theta_2 \cos \theta_1}{\sin(\theta_1 + \theta_2) \cos(\theta_1 - \theta_2)} \quad (4)$$

$$t_s = \frac{2 \sin \theta_2 \cos \theta_1}{\sin(\theta_1 + \theta_2)}. \quad (5)$$

These coefficients are plotted in Fig. 2. These equations are called Fresnel equations.

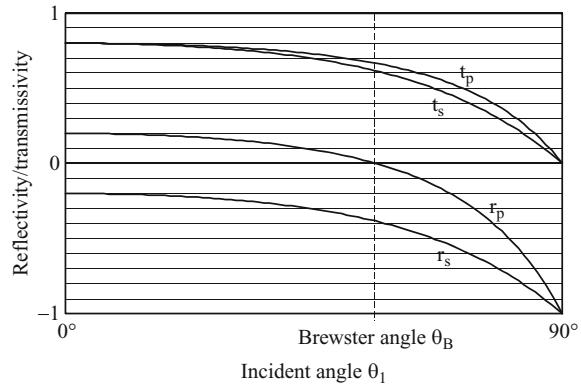
The intensity reflectivity of the p -component R_p and that of the s -component R_s , and the intensity transmissivity of the p -component T_p and that of the s -component T_s are (cf. Sect. 1.5.3 in the 5th edition of Born and Wolf [1]):

$$R_p = \frac{\tan^2(\theta_1 - \theta_2)}{\tan^2(\theta_1 + \theta_2)} \quad (6)$$

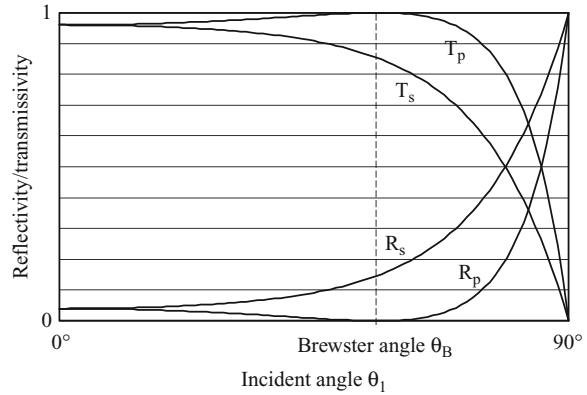
$$R_s = \frac{\sin^2(\theta_1 - \theta_2)}{\sin^2(\theta_1 + \theta_2)} \quad (7)$$

$$T_p = \frac{\sin 2\theta_1 \sin 2\theta_2}{\sin^2(\theta_1 + \theta_2) \cos^2(\theta_1 - \theta_2)} \quad (8)$$

$$T_s = \frac{\sin 2\theta_1 \sin 2\theta_2}{\sin^2(\theta_1 + \theta_2)}. \quad (9)$$



Fresnel Equations, Fig. 2 Amplitude reflectivity of the p -component r_p and s -component r_s . Amplitude transmissivity of the p -component t_p and s -component t_s . The plots correspond to the case where $n_2/n_1 = 1.5$



Fresnel Equations, Fig. 3 Intensity reflectivity of the p -component R_p and s -component R_s . Intensity transmissivity of the p -component T_p and s -component T_s . The plots correspond to the case where $n_2/n_1 = 1.5$

They are plotted in Fig. 3.

From the above equations, $R_p = 0$ can be obtained at a special incidence angle. This angle is referred to as the Brewster angle, θ_B . The Brewster angle is obtained by substituting $\theta_1 + \theta_2 = \pi/2$ (namely, $R_p = 0$) into (Eq. 1), yielding (cf. Sect. 1.5.3 in the 5th edition of Born and Wolf [1]):

$$\tan \theta_B = \frac{n_2}{n_1}. \quad (10)$$

From (Eq. 1) to (Eq. 6)–(Eq. 9), and defining $\theta = \theta_1$, $n = n_2/n_1$, the following can be derived (cf. Appendix A.6 in Miyazaki [3]):

$$R_p = \frac{1 + n^2 - (n^2 + 1/n^2) \sin^2 \theta - 2 \cos \theta \sqrt{n^2 - \sin^2 \theta}}{1 + n^2 - (n^2 + 1/n^2) \sin^2 \theta + 2 \cos \theta \sqrt{n^2 - \sin^2 \theta}} \quad (11)$$

$$R_s = \frac{1 + n^2 - 2 \sin^2 \theta - 2 \cos \theta \sqrt{n^2 - \sin^2 \theta}}{1 + n^2 - 2 \sin^2 \theta + 2 \cos \theta \sqrt{n^2 - \sin^2 \theta}} \quad (12)$$

$$T_p = \frac{4 \cos \theta \sqrt{n^2 - \sin^2 \theta}}{1 + n^2 - (n^2 + 1/n^2) \sin^2 \theta + 2 \cos \theta \sqrt{n^2 - \sin^2 \theta}} \quad (13)$$

$$T_s = \frac{4 \cos \theta \sqrt{n^2 - \sin^2 \theta}}{1 + n^2 - 2 \sin^2 \theta + 2 \cos \theta \sqrt{n^2 - \sin^2 \theta}}. \quad (14)$$

Application

Fresnel equations are mainly used to model and analyze light transmission and specular reflection. Intensity transmissivity ((Eq. 8) and (Eq. 9)) is also used for analyzing diffuse reflection and thermal radiation, since they are caused by radiation from beneath the object surface.

References

1. Born M, Wolf E (1974) Principles of optics. Pergamon, New York
2. Hecht E (2002) Optics. Pearson, San Francisco
3. Miyazaki D (2005) Shape estimation of transparent objects by using polarization analyses. PhD thesis, The University of Tokyo

Fresnel's Law

► [Fresnel Equations](#)

Fundamental Matrix

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Related Concepts

► [Epipolar Geometry](#); ► [Essential Matrix](#)

Definition

Fundamental matrix is a special 3×3 matrix which captures the geometric relationship between two cameras or between two locations of a single moving camera.

Background

See entry ► [Epipolar Geometry](#) for details.

Theory

If two points \mathbf{m} and \mathbf{m}' , expressed in pixel image coordinates in the first and second camera, are in correspondence, they must satisfy the following equation

$$\tilde{\mathbf{m}}^T \mathbf{F} \tilde{\mathbf{m}}' = 0, \quad (1)$$

where

$$\mathbf{F} = \mathbf{A}^{-T} \mathbf{E} \mathbf{A}'^{-1}, \quad (2)$$

$$\mathbf{E} = [\mathbf{t}]_{\times} \mathbf{R}, \quad (3)$$

\mathbf{A} and \mathbf{A}' are respectively the intrinsic matrix of the first and second camera, and (\mathbf{R}, \mathbf{t}) is the rigid transformation between the first and second camera. This is a fundamental constraint for two pixels to be in correspondence between two images. The 3×3 matrix \mathbf{F} is called the *fundamental matrix*, and the 3×3 matrix \mathbf{E} is known as the *essential matrix* (see entry ► [Essential Matrix](#)).

As can be seen in Eq. (2), the fundamental matrix and the essential matrix are related. If the cameras are calibrated, i.e., if \mathbf{A} and \mathbf{A}' are known, we can use the normalized image coordinates, and the fundamental matrix becomes the essential matrix.

Because $\det \mathbf{E} = 0$, we have $\det \mathbf{F} = 0$. Thus, the fundamental matrix is singular (rank 2). From Eq. (1), it is also clear that \mathbf{F} is defined up to a scale factor, which depends on the translation magnitude that

cannot be determined from image alone. Therefore, a fundamental matrix only has 7 degrees of freedom. Given one pair of corresponding image points, we have one constraint on \mathbf{F} as expressed by Eq.(1). We thus need at least seven or more point correspondences in order to determine the fundamental matrix between two images. The reader is referred to [1]

for various algorithms of determining the fundamental matrix from point correspondences.

References

1. Zhang Z (1998) Determining the epipolar geometry and its uncertainty: a review. Int J Comput Vis 27(2):161–195

G

Gait Analysis

► [Gait Recognition](#)

Gait Biometrics

► [Gait Recognition](#)

Gait Recognition

Darko S. Matovski, Mark S. Nixon and John N. Carter
School of Electronics and Computer Science,
University of Southampton, Southampton,
Hampshire, UK

Synonyms

Automatic gait recognition; Gait analysis; Gait biometrics

Related Concepts

► [Face Identification](#); ► [Optical Flow](#); ► [Principal Component Analysis \(PCA\)](#)

Definition

The way a person walks (or runs) combined with their posture is known as gait. Recognizing individuals by their particular gait using automated vision-based algorithms is known as gait recognition.

Background

Gait has few important advantages over other forms of biometric identification. It can be acquired at a distance when other biometrics are obscured or the resolution is insufficient. It does not require subject cooperation and can be acquired in a noninvasive manner. It is easy to observe and hard to disguise as walking is necessary for human mobility. Gait can be acquired from a single still image or from a temporal sequence of images (e.g., a video).

Shakespeare made several references to the individuality of gait, e.g., in *The Tempest* [Act 4 Scene 1], Cares observes “*High’st Queen of state, Great Juno comes; I know her by her gait*” and in Henry IV Part II [Act 2, Scene 3], “*To seem like him: so that, in speech, in gait, in diet, in affections of delight, in military rules, humors of blood, he was the mark and glass, copy and book.*”

The aim of medical research has been to classify the components of gait for the treatment of pathologically abnormal patients. Murray et al. [17] created standard movement patterns for pathologically normal people. Those patterns were then used to identify pathologically abnormal patients.

The biomechanics literature makes observations concerning identity: “A given person will perform his or her walking pattern in a fairly repeatable and characteristic way, sufficiently unique that it is possible to recognize a person at a distance by their gait” [27].

Psychophysiological studies such as [5, 11] have shown that humans can recognize friends and the sex of a person solely by their gait with 70–80 % accuracy. These and similar studies have inspired the use of gait as a biometric trait.

Recently, there has been a rapid growth in the number of surveillance systems, aimed to improve safety and security. These systems are yet to include recognition capabilities, and gait recognition could be a most suitable choice. The primary aim of surveillance videos is to monitor people. However, the video data can be of a low quality (poor resolution, time lapse, etc.), and the subject can try to conceal the more conventional biometrics. Nevertheless, such video can provide sufficient data for gait recognition technology, and there is already research in using gait biometrics as a forensic tool [4]. Gait recognition could be employed at a border crossing or any high-throughput environment. Gait contains very rich information and is considered to be unique. Studies have shown that gait can also be used to reveal a person's identity, gender, emotional state, etc.

Recognition by gait is one of the newest biometrics since its development only started when computer memory and processing speed became sufficient to process sequences of image data with reasonable performance. The potential for gait recognition is great, and hence there is a vast interest in computer vision research in extracting gait features.

Theory

A gait recognition system primarily consists of a computer vision system. A gait signature is created by extracting images of a walking subject which is then compared to the signatures of known subjects. [Figure 1](#) shows an example of some of the basic steps in a gait recognition system.

Step 1: Data can be acquired using a single or multiple cameras. If data is acquired using a single camera, recognition can be performed using a 2D gait signature such as the Gait Energy Image (GEI – shown in step 4). However, if multiple but synchronized cameras are used, the number of possibilities is greater. Examples of the usage of multiple-synchronized cameras include:

- Producing a 3D gait model and using it for recognition.
- 3D information can be used to improve recognition for a 2D approach by producing a non-normalized version of a 2D signature.
- Achieving a view-invariant recognition. A gait signature from any view can be re-created using 3D data which can be mapped to a signature

acquired in an outdoor environment where typically only a single and nonoptimal view is available.

Step 2: An example of preprocessing step is background subtraction or background segmentation. The subject can be acquired easily and reliably by using chroma-keying if there are clear color difference between the subject and the background. Background subtraction can measure the naturally occurring scene behind the walking subject using one of the plethora of computer vision techniques.

Step 3: As human gait is periodic, a gait sequence (sample) can consist of multiple gait cycles. Identifying the most suitable cycle can lead to better recognition rates. Signal processing techniques can be applied to the foreground signal (sum of foreground pixels) in the case of binary image.

Step 4: There are number of approaches to produce a gait signature, some of which are described later. A baseline gait signature was proposed in [22]. An example of a signature is shown in step 4.

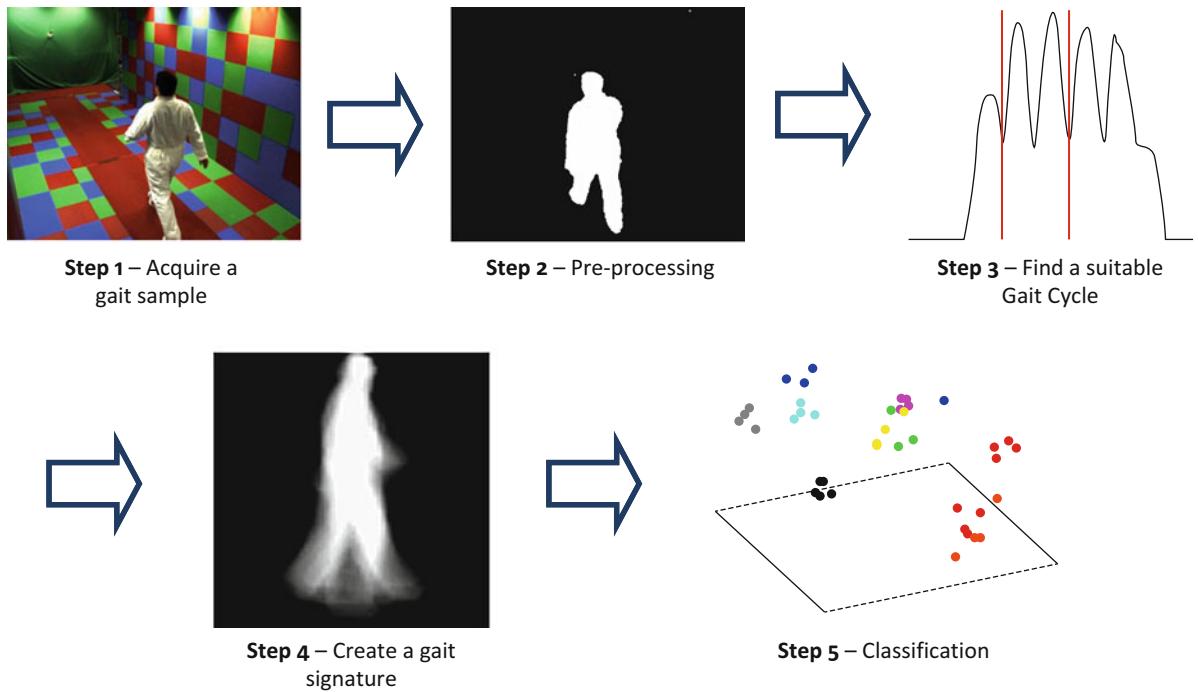
Step 5: A gait signature can be used directly within a classifier. Alternatively, features can be extracted from a signature, and those features can be used for classification. Again, there is a selection of classification techniques; in the simplest case, a classifier such as k -nearest neighbor (using Euclidian distance) can be used.

Databases

A database can be collected for various purposes. Primary concerns include uniqueness and practicality. A database should contain enough subjects to allow for an estimate of inter- and intra-subject variation. The current databases contain smaller number of subjects compared to databases used to evaluate performance of other biometrics (e.g., face, fingerprint). However, there are databases that include covariate factors and application potential. Some of the most well-known databases together with some of their characteristics are shown in [Table 1](#).

Approaches to Gait Recognition

The approaches to gait recognition can be divided in two main groups: *model-based* and *model-free*



Gait Recognition, Fig. 1 General steps of a gait recognition system

Gait Recognition, Table 1 Details of some of the well-known gait databases

| Name | Subjects | Sequences | Covariates | Viewpoints | Indoor(I)/outdoor(O) |
|-------------------------|----------|-----------|------------|------------|----------------------|
| HumanID (USF) | 122 | 1,870 | Y | 2 | O |
| SOTON 2002 | 114 | >2,500 | Y | 2 | I/O |
| CMU MoBo | 100 | 600 | Y | 6 | I (treadmill) |
| MIT 2002 | 24 | 194 | Y | 1 | I |
| UMD 2002 | 44 | 176 | N | 1 | O |
| CASIA 2006 | 124 | 1,240 | Y | 11 | I |
| SOTON multimodal [21]*1 | >300 | >5,000 | Y | 12 | I |
| Osaka University | 1,035 | 2,070 | N | 2 | I |

(see Table 2). Model-based approaches use the human body structure, and model-free methods use the whole motion pattern of the human body. Which approach is adopted depends on the acquisition conditions. Model-free (appearance-based) approaches use the input images directly to produce a gait signature without fitting a model. These approaches can perform recognition at lower resolutions which makes them suitable for outdoor applications, where a subject can be at a large distance from the camera. Model-based approaches typically require higher resolution images of a subject to be able to fit the model accurately.

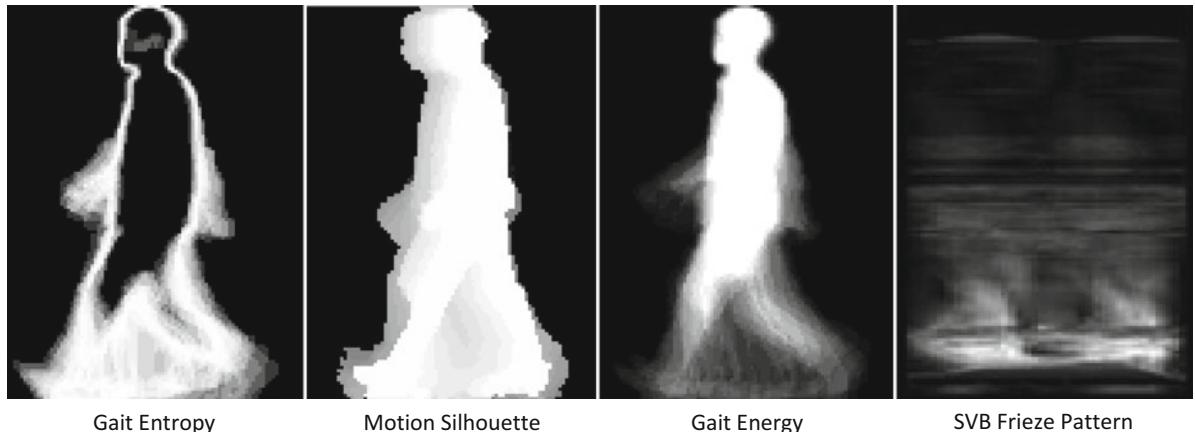
The table is taken from [18, 19]. Example papers for all of the approaches can be found in the original sources.

Model-Free Approaches

The model-free approaches derive the human silhouette by separating the moving object from the background. The subject can then be recognized by measurements that reflect shape and/or movement. The simplest approach is to simply form an average of the silhouettes over a complete gait cycle [15]. The approach is called the Gait Energy Image (GEI), and it is shown in Fig. 2. Motion Silhouette Image (MSI) is a similar representation to the GEI. The value of each pixel is computed as a function of motion of that pixel in the temporal dimension over all silhouettes that are part of a single gait cycle. Both the GEI and MSI are easy to compute, but they are vulnerable to appearance

Gait Recognition, Table 2 Approaches to gait recognition

| Model-free analysis | | Model-based analysis | |
|--|--|---|---|
| Moving shape | Shape + motion | Structural | Modeled |
| Unwrapped silhouette; silhouette similarity; relational statistics; self-similarity; key frame analysis; frieze patterns; area; symmetry; point distribution models; key poses | Eigenspace sequences; hidden Markov model; average silhouette; moments; ellipsoidal fits; kinematic features; gait style and content | Stride parameters; human parameters; joint trajectories | Articulated model; dual oscillator; linked feature trajectories |

**Gait Recognition, Fig. 2** Examples of model-free gait signatures

changes of the human silhouette. Frieze pattern represents the information contained in a gait sequence by horizontal and vertical projections of the silhouettes. Its extension, SVB Frieze patterns, use key frame subtraction in order to mitigate the effects of appearance changes on the silhouette (see Fig. 2). The Gait Entropy Image (GEI) is another example of a compact gait representation (signature). GEI is computed by calculating the Shannon entropy for each pixel. Shannon entropy measures the uncertainty associated with a random variable.

The gait signatures for the approaches shown in Fig. 2 are usually used directly for classification. There are additional ways of extracting gait signatures without using a model. Some examples are described below:

- Little and Boyd [14] derive a dense optical flow for each image sequence. Scale-independent scalar features of each flow, based on moments of the moving point, characterize the spatial distribution of the flow. The periodic structure of these sequences of scalars is analyzed. The scalar sequences for an

image sequence have the same fundamental period but differ in phase, which is used as a feature for recognition of individuals by the shape of their motion.

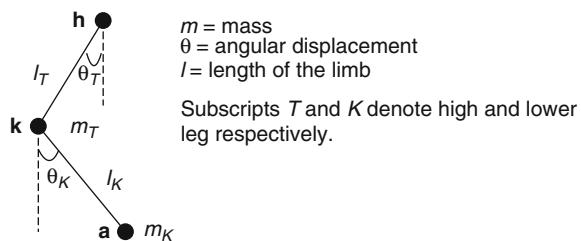
- BenAbdelkader et al. [2] use background modeling to track the subject for a number of frames and extract a sequence of segmented images of the person. A self-similarity plot is computed via correlation of each pair of images in this sequence. For recognition, PCA (principal component analysis) is used to reduce the dimensionality of the plots. A k-nearest neighbor rule is used on the reduced space for classification. Another silhouette-based gait recognition technique using PCA has been proposed by Liang et al. [13]. Eigenspace transformation based on principal component analysis (PCA) is applied to time-varying distance signals derived from a sequence of silhouette images to reduce the dimensionality of the input feature space. Supervised pattern classification techniques are performed in the lower-dimensional eigenspace for recognition.

- Hayfron-Acquah et al. [8] proposes a method for automatic gait recognition based on analyzing the symmetry of human motion. The Generalised Symmetry Operator is used to locate features according to their symmetrical properties rather than relying on the borders of a shape. The symmetry operator is used on the optical flow image to produce a gait signature. For purposes of classification, the similarity differences between the Fourier descriptions of the gait signatures are calculated using Euclidean distance.
- Gait is a temporal sequence and can be modeled using hidden Markov models (HMM). The statistical nature makes the model relatively robust. The postures that an individual adopts are regarded as states of the HMM and are typical to that individual and provide means of discrimination [24].
- Kale et al. [10] use two different image features to directly train a HMM: the width of the outer contour of a binary silhouette and the entire binary silhouette itself.

Model-Based Approaches

The advantages of the previous approaches (silhouette or features derived from it) are speed and simplicity. However, model-based approaches are better at handling occlusion, noise, scale, and rotation. Model-based approaches require a high resolution therefore not very suitable for outdoor surveillance.

Model-based approaches incorporate knowledge of the shape and dynamics of the human body into the extraction process. These approaches extract features that fit a physical model of the human body. A gait model consists of shapes of various body parts and how those shapes move relative to each other (motion model). The shape model for a human subject can use ellipse to describe the head and the torso, quadrilaterals to describe the limbs, and rectangles to describe the feet. Alternatively, arbitrary shapes could be used to describe the edges of the body parts. The motion model describes the dynamics of the motion of the different body parts. Using a model ensures that only image data corresponding to allowable human shape and motion is extracted, reducing the effect of noise. The models can be 2- or 3-dimensional. Most of the current models are 2-dimensional, but deliver good results on databases of more than 100 subjects.



Gait Recognition, Fig. 3 Example of a gait model – the dynamically coupled pendulum model [28]

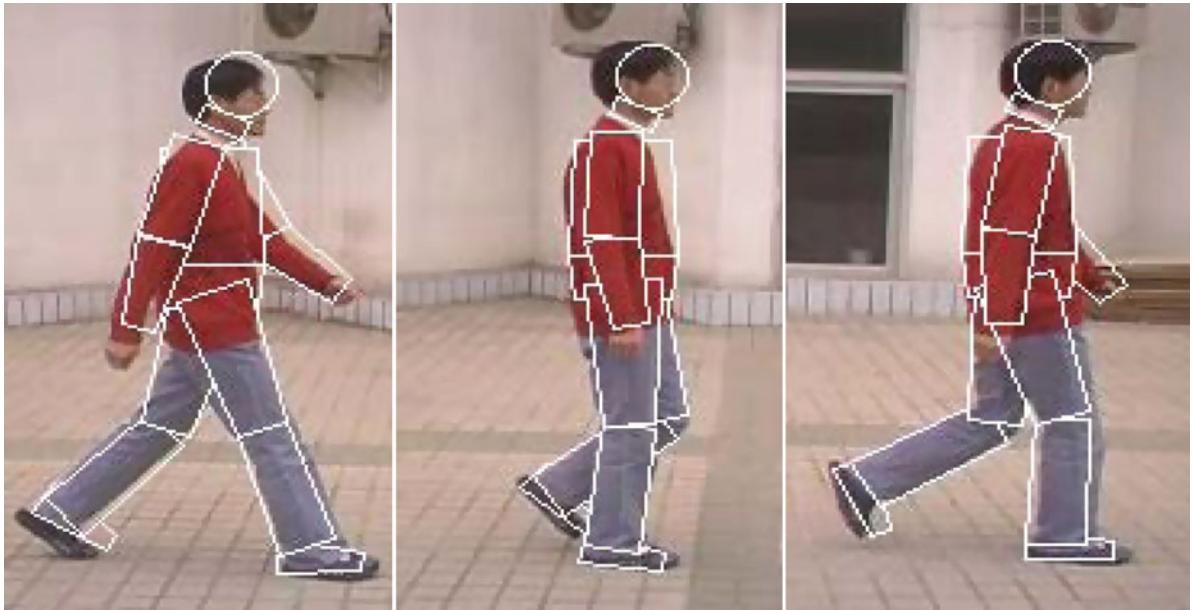
Some examples of model-based approaches are described below:

- Yam et al. [28] have used pendular motion and the understanding of biomechanics of human locomotion to develop two models: a bilateral symmetric and analytical model (employs the concept of forced couple oscillator). See Fig. 3. The gait signature is the phase-weighted magnitude of the Fourier description of both the thigh and knee rotation.
- Bouchrika and Nixon [3] have proposed a new approach to extract human joints. Spatial model templates for human motion are derived from the analysis of gait data collected from manual labeling. Motion templates describing the motion of the joints are parameterized using the elliptic Fourier descriptors

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} a_0 \\ b_0 \end{bmatrix} + \begin{bmatrix} \cos(\alpha) - \sin(\alpha) \\ \sin(\alpha) \cos(\alpha) \end{bmatrix} \begin{bmatrix} X(t) * S_x \\ Y(t) * S_y \end{bmatrix}$$

where α is the rotation angle, s_x and s_y are the scaling factors across the horizontal and vertical axes, respectively, and $X(t)$ and $Y(t)$ are the Fourier summation. Hough transform is used in the feature extraction process.

- Wang et al. [26] have proposed an algorithm based upon the fusion of static and dynamic body information. The static body information is in a form of a compact representation obtained by Procrustes shape analysis. The dynamic information is obtained by a model-based approach which tracks the subject and recovers joint-angle trajectories of lower limbs. A fusion at the decision level is used to improve recognition results. Figure 1 shows an example of the results obtained.



Gait Recognition, Fig. 4 Example of results [26]

Gait Recognition, Table 3 Progression of gait recognition systems

| Time period | No of subjects | Source | Recognition rate | Notes |
|---------------|----------------|-------------------------|------------------|--|
| 1990s | ~10 | USC | 95.2 | |
| 2000s | ~120 | HiD, CASIA, Southampton | 75–99 % | Recognition rate depends on covariates |
| Recent (2010) | >300 | Southampton multimodal | 95–100 % | Includes time-dependent covariates |
| Recent (2010) | >1000 | Osaka University | 90 % | No covariates |

There have been moves towards developing 3D gait models. Examples of work in this fields are [7, 25]. Guoying et al. [7] use video sequences from multiple cameras to construct 3D human models. The motion is tracked by applying a local optimization algorithm. The length of key segments is extracted as static parameters, and the motion trajectories of lower limbs are used as dynamic features. Linear time normalization is used for matching and recognition. Three-dimensional approaches are robust to changes in viewpoint and have a great potential. However at present, experiments only on small databases are possible mainly due to high computational requirements.

Gait is dependent on large number of parameters (joint angles and body segment size) which leads to complex models with many free parameters. Finding the best fit model for a subject leads to searching a high-dimensional parameter space. Therefore, there is a trade off between the accuracy of the model

(complexity) and computational cost. The models are often simplified based on certain acceptable assumptions, e.g., a system could assume constant walking speed. However, as computing power increases, the problems arising of high complexity can be mitigated.

Experimental Results

The current state of the art achieves very high recognition rates (close to 100 %) on relatively large databases (>300 subjects) when the training and test data are recorded under similar conditions. An example of progression in performance over time is shown in Table 2. However, recognition rate can drop with change of clothing, shoes, walking surface, and pose. Many current studies focus on solving these problems. Recent major achievements in gait recognition are

Gait Recognition, Table 4 Some experiments comprising the HumanID gait challenge problem

| Experiment | Probe | # of subjects | Difference |
|------------|---|---------------|---------------------|
| A | Different camera view than gallery | 122 | View |
| B | Subjects wore different shoes | 54 | Shoe |
| C | Different camera view and different shoes | 54 | Shoe, view |
| D | Subjects walked on a different surface | 121 | Surface |
| E | Different shoes and different walking surface | 60 | Surface, shoes |
| F | Different walking surface and different camera view | 121 | Surface, view |
| G | Different walking surface, different shoes, and different camera view | 60 | Surface, shoe, view |

described in [16, 20]. Matovski et al. [16] have shown that elapsed time does not affect gait recognition and that gait can be used as a reliable biometric over time and at a distance. The world's largest gait database of more than 1,000 people has been constructed to enable statistically reliable performance evaluation of gait recognition performance [20] (Table 3).

The HumanID gait challenge problem [22] was set up to outline a baseline algorithm for gait recognition and propose a number of difficult experiments for the existing gait matchers. The gallery set consists of 122 subjects walking on a grass surface recorded by a single camera.

Table 4 shows the differences of the probe set compared to the gallery set for each of the challenge experiments.

The results in Fig. 5 show the progress in gait recognition over a period of 2 years for the experiments shown in Table 4.

Gait recognition has been used as evidence for conviction in some criminal cases. A man in Bolton (UK) was convicted based on his distinctive gait. A CCTV footage of the burglar captured near the crime scene was compared to a video captured at the police station by a podiatrist specializing in gait analysis. In 2004, a perpetrator robbed a bank in Denmark. The Institute of Forensic Medicine in Copenhagen was contacted by the police to perform gait analysis, as they thought the perpetrator had a unique gait. The institute instructed the police to establish a covert recording of the suspect from the same angles as the surveillance recordings for comparison. The gait analysis revealed several characteristic matches between the perpetrator and the suspect. For example, both the perpetrator (to the left) and the suspect showed inverted left ankle (white arrow) during left leg's stance phase and markedly outward rotated feet (see Fig. 6). The suspect was convicted of robbery, and the court found that gait analysis is a very valuable tool [12].

One system named the Biometric Tunnel [23] has led to the first live demonstration of gait as a biometric and could indicate a possible route for future deployment of the technology. The left side of Fig. 7 depicts the system. It consists of a simple corridor with 12 synchronized and fixed cameras. The subjects are asked to walk through the middle, and the lighting and background are controlled to facilitate analysis. The right side of Fig. 7 shows the details of the arrangement. The system is designed with a high-throughput environment in mind.

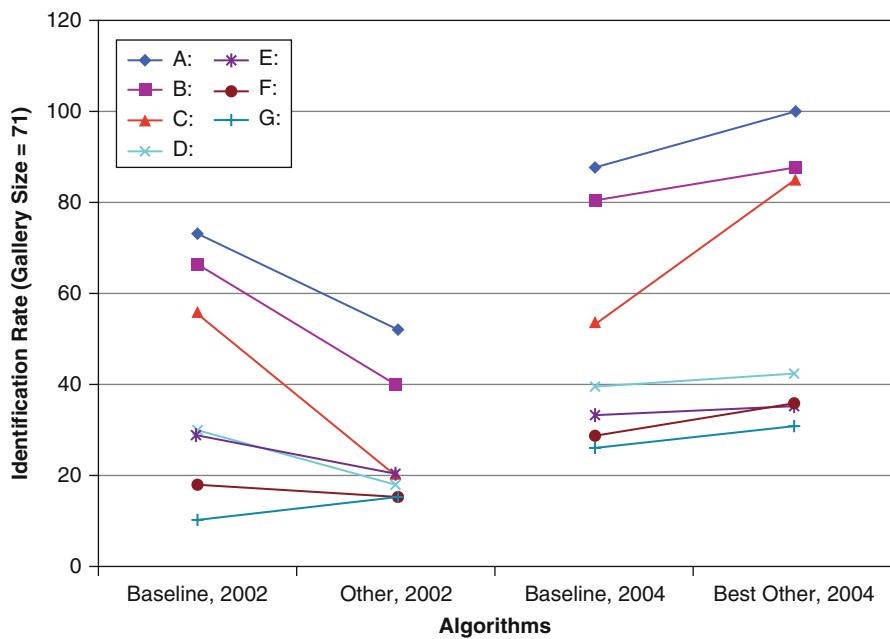
Application

Gait research is currently at an evaluation stage rather than an application stage. However, the potential for gait recognition is great. The complete unobtrusiveness without any subject cooperation or contact for data acquisition makes gait particularly attractive for identification purposes. It could be used in applications including forensics, security, immigration, and surveillance.

Many surveillance systems capture only a low-resolution video at varying lighting conditions, and gait recognition might be the only plausible choice for automatic recognition. A bank robber may wear a mask so you cannot see his face, wear gloves so you cannot get fingerprints, and wear a hat so you cannot get DNA evidence – but they have to walk or run into the bank, and they could be identified from their gait.

Open Problems

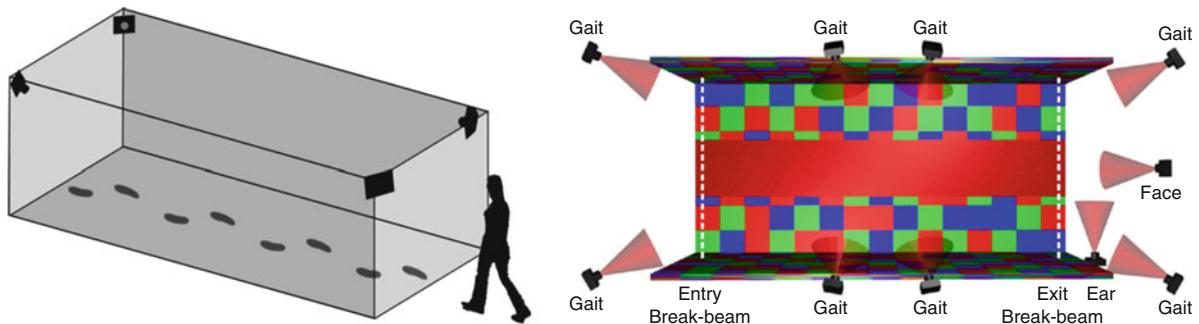
Although a large number of gait recognition algorithms have been reported, it is important to note that gait biometrics is still in its infancy. The majority of



Gait Recognition, Fig. 5 The progress from the baseline over 2 years for the various experiments shown in Table 4 [22]



Gait Recognition, Fig. 6
Bank robbery



Gait Recognition, Fig. 7 The biometric tunnel

studies achieve good recognition rates on gallery and probe acquired in similar conditions. However, it is very challenging to extract gait features that are invariant to change in appearance as well as to conditions that affect a person's gait. Examples of things that can change and negatively affect the effectiveness of current gait algorithms are change of clothing, shoe type, carrying a load, and injuries/medical conditions. Clothing for instance can change the observed pattern of motion and make it difficult to accurately locate joint position. Furthermore, there are certain factors that are related to the environment and not the subjects themselves that can cause difficulties for current gait matchers. Examples of environmental confounding factors are camera viewing angle, background, and illumination.

Recently, studies have reported progress in solving some of the issues outlined above. A study by Goffredo et al. [6] describes a self-calibrating view-invariant gait recognition algorithm. Hossain et al. [9] have developed a clothing invariant gait matcher. Aqmar et al. [1] are the most recent approach focussed on speed variation.

Currently, gait recognition can deliver very high recognition rates in a constrained environment and if certain factors are controlled. A move towards developing algorithms invariant to change over time is needed. Furthermore, additional work is required to translate the research to outside environment and to explore how scalable it is. Attempts so far suggest that developing highly reliable gait-based human identification system in a real-world application is, and will continue to be, very challenging. In the short term, some of the challenges associated with gait recognition can be addressed by fusing gait with other biometrics [29].

References

1. Aqmar MR, Shinoda K, Furui S (2010) Robust gait recognition against speed variation. In: Proceedings of 20th IEEE international conference on pattern recognition (ICPR), Istanbul, Turkey
2. Benabdelkader C, Cutler R, Davis L (2002) Motion-based recognition of people in eigengait space. In: Proceedings of IEEE international conference automatic face and gesture recognition (AFGR). IEEE, Piscataway, pp 267–272
3. Bouchrika I, Nixon M (2008) Exploratory factor analysis of gait recognition. In: 8th IEEE international conference on automatic face and gesture recognition (AFGR), Amsterdam, The Netherlands
4. Bouchrika I, Goffredo M, Carter JN, Nixon MS (2010) On using gait in forensic biometrics. Journal of Forensic Sciences 56(4):882–889 July 2011
5. Cutting JE, Kozlowski LT (1977) Recognizing friends by their walk: gait perception without familiarity cues. Bull Psychon Soc 9(5):353–356
6. Goffredo M, Bouchrika I, Carter JN, Nixon MS (2010) Self-calibrating view-invariant gait biometrics. IEEE Trans Syst Man Cybern Part B Cybern 40(4):997–1008
7. Guoying Z, Guoyi L, Hua L, Pietikainen M (2006) 3D gait recognition using multiple cameras. In: Proceedings of 7th IEEE international conference automatic face and gesture recognition (AFGR), Southampton, pp 529–534
8. Hayfron-Acquah J, Nixon M, Carter J (2003) Automatic gait recognition by symmetry analysis. Pattern Recognit Lett 24(13):2175–2183
9. Hossain MA, Makihara Y, Junqui W, Yagi Y (2010) Clothing-invariant gait identification using part-based clothing categorization and adaptive weight control. Pattern Recognit 43(6):2281–2291
10. Kale A, Sundaresan A, Rajagopalan A, Cuntoor N, Roy-Chowdhury A, Kruger V, Chellappa R (2004) Identification of humans using gait. IEEE Trans Image Process 13(9):1163–1173
11. Kozlowski LT, Cutting JE (1977) Recognizing the sex of a walker from a dynamic point-light display. Percept Psychophys 21(6):575–580
12. Larsen PK, Simonsen EB, Lynnerup N (2008) Gait analysis in forensic medicine. J Forensic Sci 53(5): 1149–1153

13. Liang W, Tieniu T, Huazhong N, Weiming H (2003) Silhouette analysis-based gait recognition for human identification. *IEEE Trans Pattern Anal Mach Intell* 25(12): 1505–1518
14. Little J, Boyd J (1998) Recognizing people by their gait: the shape of motion. *Videre: J Comput Vis Res* 1(2): 1–32
15. Liu Z, Sarkar S (2004) Simplest representation yet for gait recognition: averaged silhouette. In: Proceedings of 17th IEEE international conference pattern recognition (ICPR), Cambridge, UK, pp 211–214
16. Matovski DS, Nixon MS, Mahmoodi S, Carter JN (2010) The effect of time on the performance of gait biometrics. In: 4th IEEE international conference biometrics: theory, applications and systems (BTAS), Washington, DC, USA
17. Murray M, Drought A, Kory R (1964) Walking patterns of normal men. *J Bone Jt Surg* 46(2):335
18. Nixon MS, Carter JN (2006) Automatic recognition by gait. *Proc IEEE* 94(11):2013–2024
19. Nixon MS, Tan TN, Chellappa R (2005) Human identification based on gait. Springer, New York
20. Okumura M, Iwama H, Makihara Y, Yagi Y (2010) Performance evaluation of vision-based gait recognition using a very large-scale gait database. In: 4th IEEE international conference biometrics: theory, applications and systems (BTAS), Washington, DC
21. Samangooei S, Bustard J, Nixon MS, Carter JN (2010) On acquisition and analysis of a dataset comprising of gait, ear and semantic data. In: Bhanu B, Govindaraju V (eds) Multi-biometrics for human identification. Cambridge University Press, Cambridge, (in press)
22. Sarkar S, Phillips PJ, Zongyi L, Isidro Robledo V, Grother P, Bowyer KW (2005) The HumanID gait challenge problem: data sets, performance, and analysis. *IEEE Trans Pattern Anal Mach Intell* 27(2):162–177
23. Seely R, Samangooei S, Lee M, Carter J, Nixon M (2008) University of Southampton multi-biometric tunnel and introducing a novel 3d gait dataset. In: Proceedings of 2nd IEEE international conference biometrics: theory, applications and systems (BTAS), Washington, DC
24. Sundaresan A, Roychowdhury A, Chellappa R (2003) A hidden markov model based framework for recognition of humans from gait sequences. In: Proceedings of IEEE international conference image processing (ICIP), Barcelona, pp 93–96
25. Urtasun R, Fua P (2004) 3D tracking for gait characterization and recognition. In: Proceedings of 6th IEEE international conference automatic face and gesture recognition (AFGR), Seoul, pp 17–22
26. Wang L, Ning H, Tan T, Hu W (2004) Fusion of static and dynamic body biometrics for gait recognition. *IEEE Trans Circuits Syst Video Technol* 14(2):149–158
27. Winter DA (2009) Biomechanics and motor control of human movement. Wiley, Ottawa
28. Yam C, Nixon M, Carter J (2004) Automated person recognition by walking and running via model-based approaches. *Pattern Recognit* 37(5):1057–1072
29. Zhou X, Bhanu B (2008) Feature fusion of side face and gait for video-based human identification. *Pattern Recognit* 41(3):778–795

Gamut Mapping

Rajeev Ramanath¹ and Mark S. Drew²

¹DLP® Products, Texas Instruments Incorporated, Plano, TX, USA

²School of Computing Science, Simon Fraser University, Vancouver, BC, Canada

Synonyms

[Color management](#)

Related Concepts

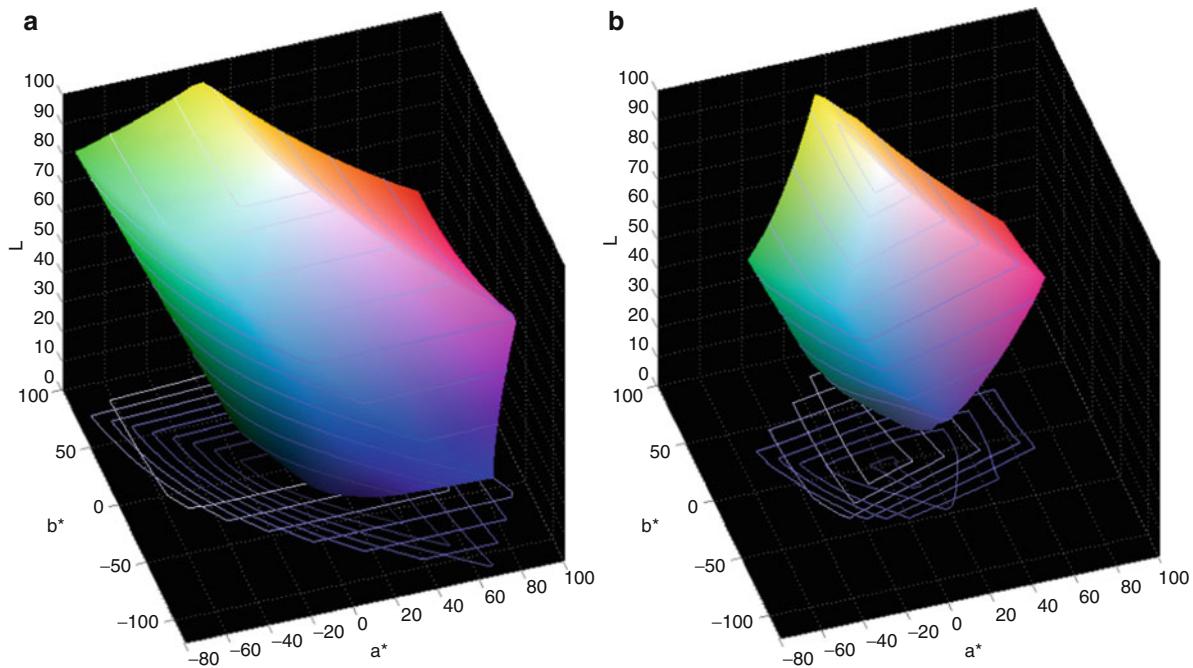
► [Gamut Mapping](#)

Definition

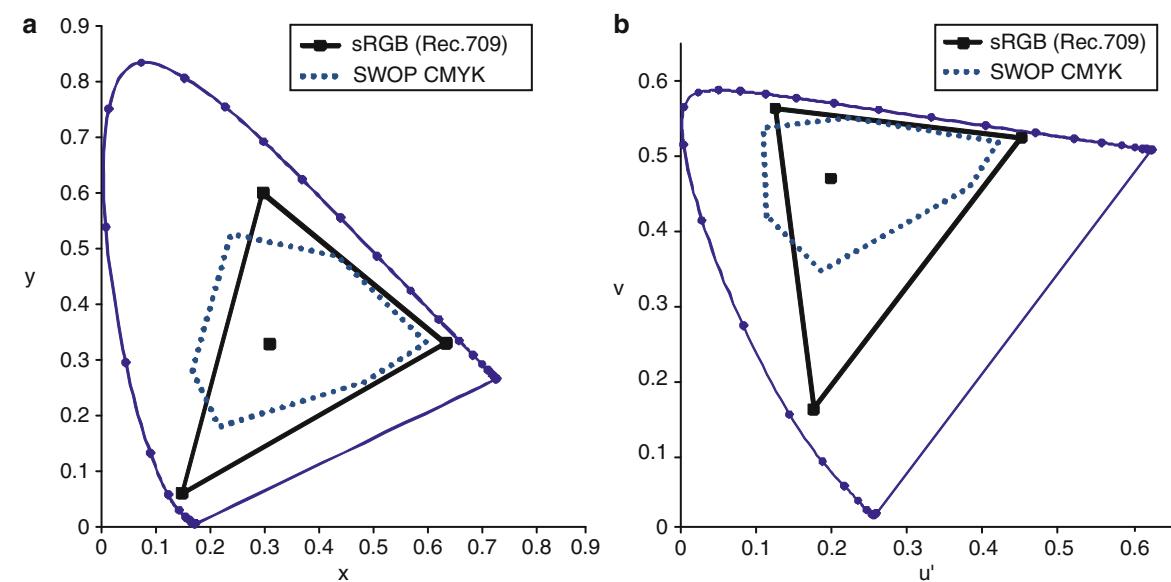
Gamut Mapping refers to the process of translating colors in one device's color space to that of another. This process is performed on colors in images and video so as to create a rendition of a source image (typically in a capture device's color space) in an output device's color space while meeting several rendering intents: absolute and relative colorimetric fidelity, perceptual accuracy, and the problem of saturation – each of which trades off one color property at the expense of another.

Background

Different media – cameras, printers, displays – have different achievable gamuts, depending on the manner in which color is either captured or reproduced. This typically means that one medium can have colors that may not be reproducible on another. The color gamut of a device may be displayed as a volume of achievable colors, and typically this is shown in a CIELAB or CIELUV (see Fig. 1), or as a projection on the CIE xy or u“v” chromaticity diagram (see Fig. 2). The three-dimensional representation is far more informative than the two-dimensional projection as it captures the nuances of the color space, specifically around the luminance of the primaries and the associated black



Gamut Mapping, Fig. 1 Three-dimensional renderings of color gamuts: (a) device with sRGB color space, (b) device with SWOP color space



Gamut Mapping, Fig. 2 Two-dimensional renderings of color gamuts of a device with sRGB color space and one with SWOP color space in (a) CIE xy chromaticity diagram, (b) CIE u“v” chromaticity diagram

and white levels of the medium. The two-dimensional projection requires multiple luminance slices to be plotted for it to be comparably informative.

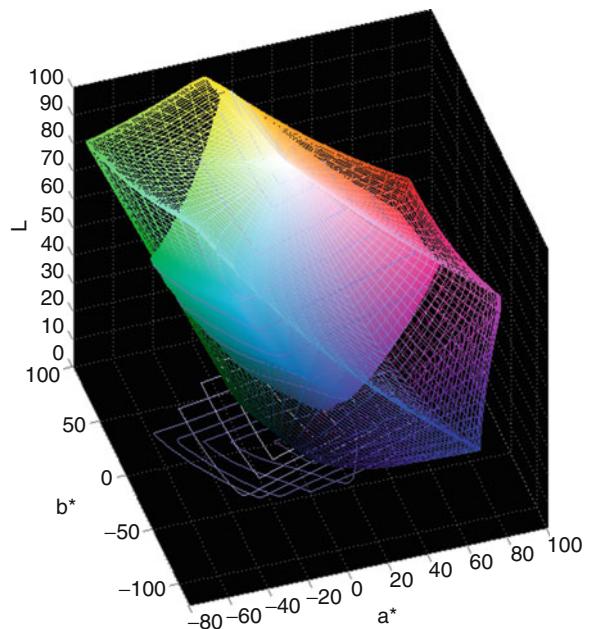
Theory

The challenge of gamut mapping may be described by means of an example of a case where an image stored within an sRGB color space needs to be reproduced in a print medium with SWOP colors: Fig. 3 shows two color gamuts – the sRGB color gamut (wireframe) almost enclosing a SWOP color space (solid). It is interesting to note that there are regions of the color space that are represented by the sRGB color space but are not represented by the SWOP color space and vice versa. In performing such a mapping, the following challenges arise:

- How should black and white be mapped?
- Should colors in the intersection of the two gamut volumes be reproduced as such, or should they be compressed?
- How should input colors that are outside the output color space be reproduced?
- What must be done with colors that can be created with the output color space but cannot be represented in the input color space?

The objective of gamut mapping algorithms is to translate colors in the input color space to achievable colors in the output color space so as to meet certain key criteria that are referred to as rendering intents (per ICC guidelines [3]):

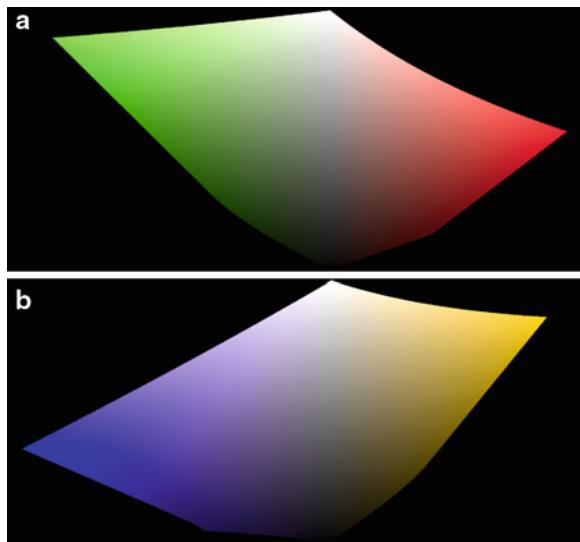
- *ICC-absolute colorimetric intent*: “Chromatically adapted tristimulus values of in-gamut colors are unchanged.” This intent preserves the relationship among in-gamut colors at the expense of out-of-gamut colors while maintaining accuracy. Again, nothing is specifically stated about the mapping of out-of-gamut colors.
- *Media-relative colorimetric intent*: “The use of media-relative colorimetry enables colour reproductions to be defined which maintain highlight detail, while keeping the medium white, even when the original and reproduction media differ in colour.” This intent also preserves the relationship between in-gamut colors at the expense of out-of-gamut colors while maintaining accuracy. Nothing is specifically stated about the mapping of out-of-gamut colors.



Gamut Mapping, Fig. 3 Three-dimensional renderings of color gamuts of an sRGB color space (wireframe) and a SWOP color space (solid)

- *Perceptual intent*: “The exact gamut mapping of the perceptual intent is vendor specific and is useful for general reproduction of pictorial images, typically includes tone scale adjustments to map the dynamic range of one medium to that of another, and gamut warping to deal with gamut mismatches.” This objective of this intent is clear from the definition, and typically involves proprietary algorithms to perform the gamut mapping for general reproduction of images, particularly pictorial or photographic-type images.
- *Saturation intent*: “The exact gamut mapping of the saturation intent is vendor specific and involves compromises such as trading off preservation of hue in order to preserve the vividness of pure colours.” This objective of this intent is also clear from the definition, and also typically involves proprietary algorithms to perform the gamut mapping for images and video that contains charts and diagrams.

Most gamut mapping algorithms are performed in either the device space or in a perceptual space: the choice of which space to use is highly application and preference dependent. For the sake of simplicity, in this



Gamut Mapping, Fig. 4 A set of slices of the color gamut of an sRGB color space (in a CIELAB representation) showing the (a) red–green slice and (b) a yellow–blue slice

article, we will restrict ourselves to a CIELAB space along with its perceptual correlates: CIELAB Lightness, Chroma, and Hue. However it should be noted that gamut mapping algorithms could be defined in a variety of spaces, making use of each space’s general perceptual correlates of hue, chroma/colorfulness, and lightness/brightness.

Typical color gamuts are shown in Fig. 3 and particular Hue slices for red–green, and yellow–blue are shown in Fig. 4. That is, these slices each show a vertical slice through a color solid, each at a particular position around the surrounding circle we consider to be Hue – i.e., is a color red, is it green, etc. The Chroma increases from zero in the center of the solid to a maximum value on the outside of the solid, and Lightness (the correlate of brightness) goes from minimum (black) at the bottom to maximum (white) at the top. To ease illustration, Hue slices will be abstracted using figures like those in Fig. 5 composed of highly simplified shapes (triangles in this case) – real-world color gamuts can be far more complicated and not included in these illustrations. It is to be noted that the point $L_{M\text{-out}}$ is often referred to as the *cusp* for that Hue slice.

Multiple variations of different scenarios of Hue slices are shown in Fig. 6 to illustrate the variety of problems one might encounter with gamut mapping even when the shape of the gamut is simplified using

triangles. These four cases show the input gamut to be larger (in the Lightness scale) than the output gamut, but it is straightforward to envision the opposite set of cases (swapping input and output color spaces) to be possible as well for different gamut pairs. It is also straightforward to envision gamut pairs for which the Lightness scales match. Clearly, even for one gamut pair, one chosen approach for “mapping” the gamuts may work for most Hue slices, but might fail for a different Hue slice.

In general, gamut mapping algorithms may be classified based on which of four approaches is taken:

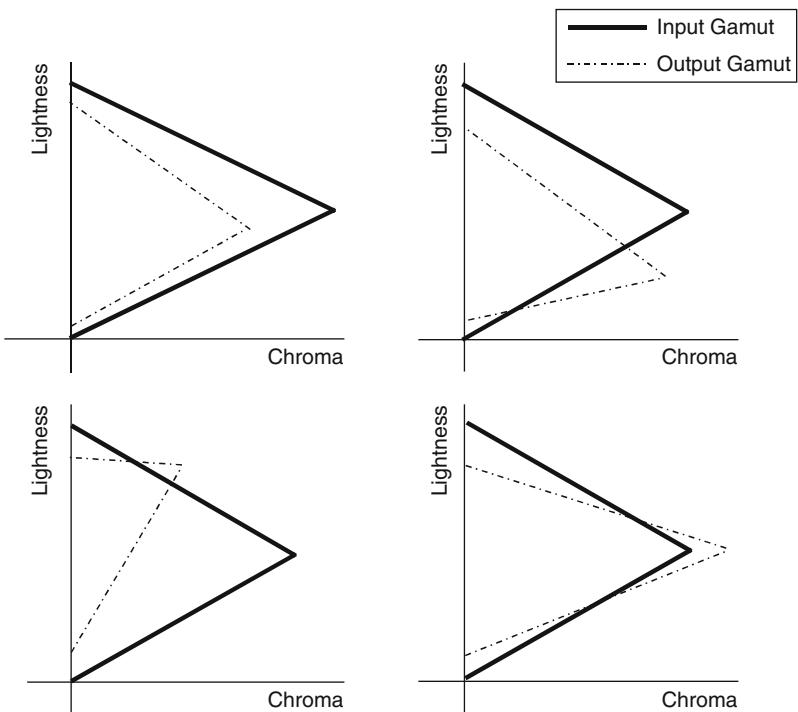
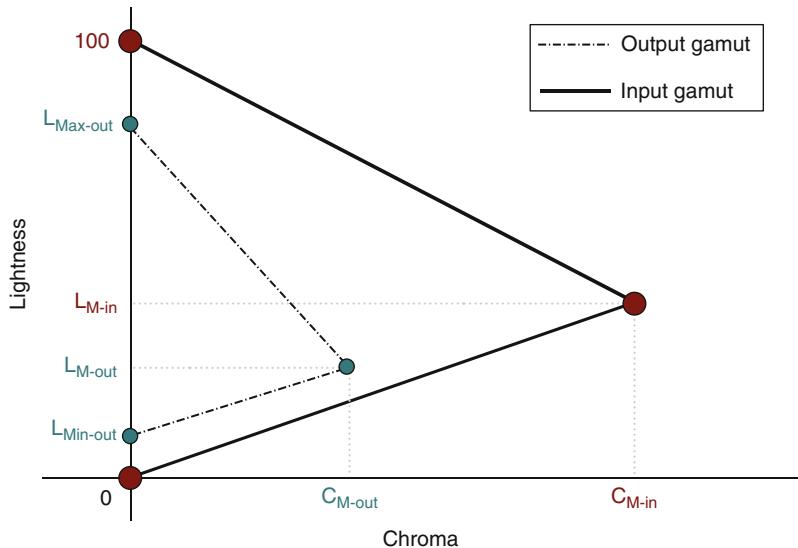
- Gamut Clipping Algorithms
- Gamut Compression/Expansion Algorithms
- Spatial Gamut Mapping Algorithms
- Memory-Color-Aware Gamut Mapping Algorithms

Gamut Clipping

Gamut clipping approaches may be described as those that address those colors that are outside the gamut of the output color space. Some of the most common approaches include those that are *nearest neighbor* and those that are *Hue-preserving*. Nearest-neighbor approaches map the color in the input gamut’s color space to a color in the output gamut’s color space that is nearest as defined by one of the following criteria:

- Nearest color in the output color space along a vector of constant Lightness: Mapping along the vector of constant Lightness tends to address the mapping issue as a simple one-dimensional problem of changing only the Chroma of a color. This is shown as approach (a) in Fig. 7, with the mapping occurring along the dashed blue lines.
- Nearest color in the output color space along a vector to the point in the Hue slice with a Lightness of 50. Mapping along the vector pointing to a Lightness of 50 has the advantage that both Lightness and Chroma are adjusted, which helps maintain contrast between colors. This is shown as approach (b) in Fig. 7. This approach is sometimes further divided as a mapping to a different Lightness value for colors above and below the *cusp* of the input gamut, shown as approach (c) in Fig. 7.
- Nearest color in the output color space along a vector to the point in the Hue slice with a Lightness of the *cusp* ($L_{M\text{-out}}$ in Fig. 5): Mapping to the cusp has the advantage of maintaining the general trend of

Gamut Mapping, Fig. 5 A highly simplified Hue slice of color gamut of a gamut pair showing the min-max and most chromatic points in the two gamuts

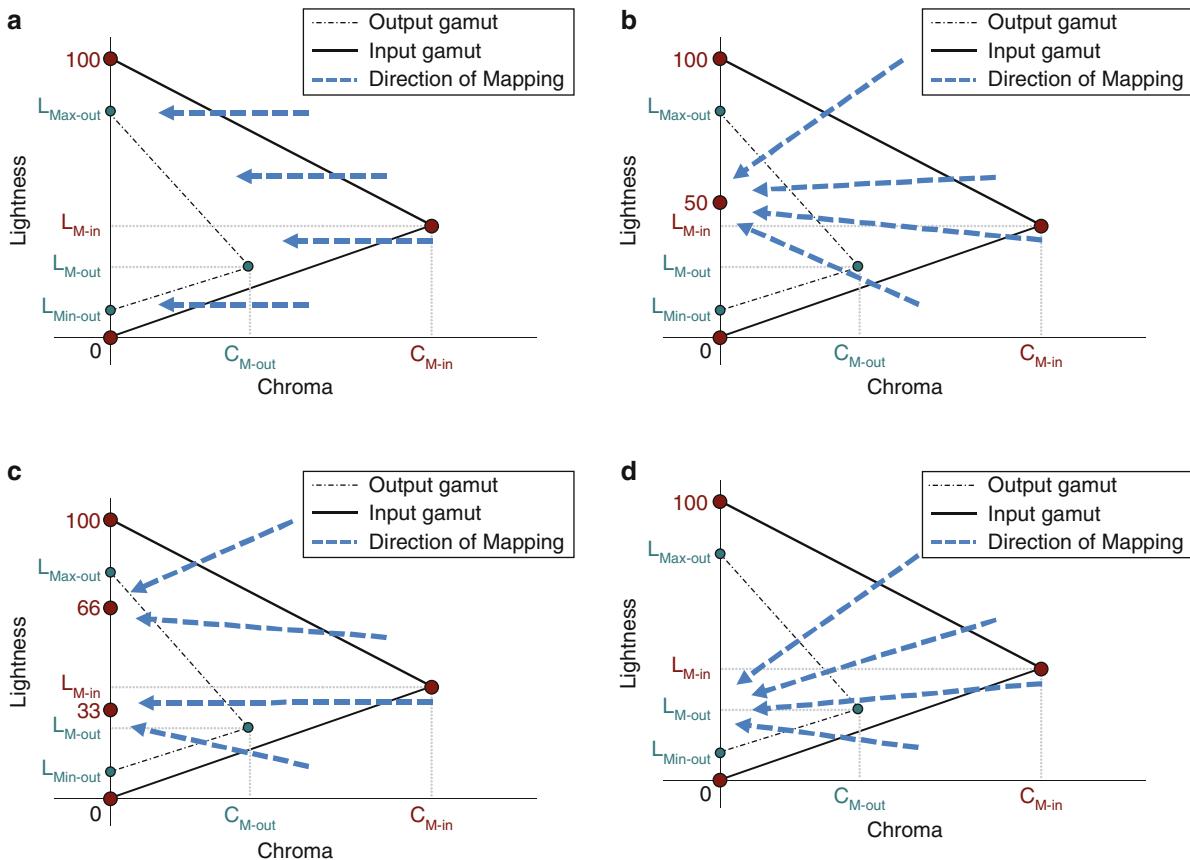


Gamut Mapping, Fig. 6
Four variations of Hue slices showing challenges that gamut mapping algorithms have to address. The complement of these approaches also exist wherein the input and output gamuts are reversed in these pictures

the Lightness in the input and output color spaces. This is shown as approach (d) in Fig. 7.

- Nearest color in the output color space based on a color difference measure: If one were to use the CIELAB difference measure ΔE_{ab}^* , which is approximately perceptually uniform, a color

would be mapped to an output color that is perceptually closest to the input color as defined by the ΔE_{ab}^* measure. A slightly different variant of this approach would be to use a different color difference measure, such as ΔE_{94} , or ΔE_{00}^* (CIEDE2000), or some other preferred color



Gamut Mapping, Fig. 7 Different directions for mapping colors showing the (a) mapping along lines of constant Lightness, (b) mapping along lines of fixed Lightness, (c) mapping along

lines of varying lightness with a simple example of two different Lightnesses, (d) mapping along the vectors to the output gamut's cusp

difference measure. It is to be noted that this can result in shifts in the Hue descriptor for a color.

Gamut Compression/Expansion

One of the challenges that arises with gamut clipping is that this invariably results in loss of detail in images when multiple input colors map to the same output color, especially when large differences in color gamuts are to be overcome. If this behavior is not desired, gamut compression (or alternatively expansion) is typically performed. The approaches for gamut compression/expansion may be classified as belonging to one of three classes:

- *Lightness compression/expansion* approaches operate only along the Lightness/luminance axis when the largest difference is along this axis of the color space. The resulting Chroma (after Lightness

mapping) could either be clipped or get compressed/expanded appropriately.

- *Chroma compression/expansion* approaches operate only along the Chroma axis. The resulting Lightness (after Chroma mapping) could either be clipped or get compressed/expanded appropriately.
- *Lightness and chroma compression* approaches operate on both Lightness and Chroma at the same time by mapping colors along various directions, as for those shown in Fig. 7a-d.

In all these approaches, independent of which dimension of the color space is compressed/expanded, or which direction of compression/expansion is chosen, the choice of the input-output relationship impacts the outcome significantly. Figure 8 shows various input-output relationships for the two parameters – Lightness and Chroma – that may be used for the

above approaches. All these approaches are typically performed on a hue-by-hue basis.

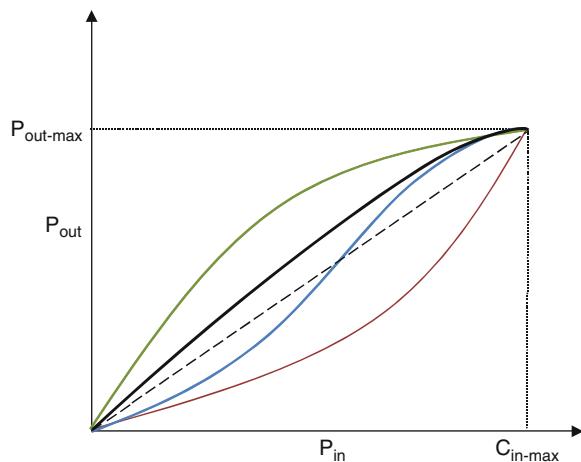
Spatial Gamut Mapping Algorithms

The gamut mapping approaches discussed thus far consider individual colors in an image in isolation, aside from their relationship in the color space. This disregards key factors that are known to impact the appearance of colors to human observers, such as spatial frequency of the image/color, surround colors, and other color appearance phenomena that can greatly change the perception of a color. Discounting these factors typically results in loss of detail in images or a degradation in the perception of the image – although specific colors may have been “accurately” mapped. A generic class of approaches called spatial gamut mapping algorithms take the spatial relationships of colors into account when mapping colors. A simple,

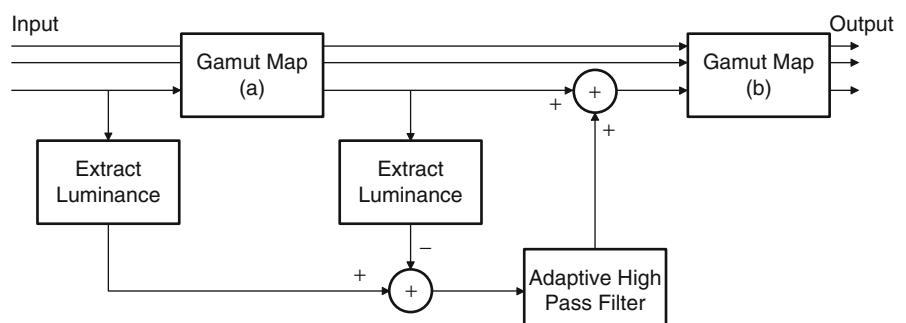
yet robust, framework proposed by Balasubramanian et al. [1] is shown in Fig. 9 where the two gamut mappers (a) and (b) are tuned for the specific application. The objective of the spatial high-pass filter is to allow for Lightness variations in the signal when high-frequency content in the image is encountered. This is based on the observation that humans possess the most visual acuity in the brightness dimension of color, and so high spatial frequency content is more perceived along the Lightness axis.

Memory-Color-Aware Gamut Mapping Algorithms

Although this class of algorithms may be combined with the three other classes of gamut mapping solutions, it is important to address this issue separately as this is at the core of high-quality gamut mapping algorithms. One of the most challenging tasks of a gamut mapping algorithm arises from the fact that its performance is typically evaluated by a human observer. Human observers have visual systems that, apart from being highly complex in terms of the appearance phenomena that determine their behavior, have preferences: the sky has to be a certain shade of blue, human skin tone rendition has to be a “certain” desired color, the color of green grass has to be a very specific shade of green, etc. This requires gamut mapping algorithms to comprehend these colors, deemed *memory colors*: colors that the human observer “knows” to be a certain shade, especially in the context in which they are presented. Approaches that attempt to maintain the rendition of memory colors tend to be proprietary to the vendors. These approaches tend to be mostly probabilistic in nature, taking into account a very large database of memory colors as rendered in the output media and deemed suitable by the solution providers.



Gamut Mapping, Fig. 8 Different input–output relationships for gamut compression/expansion, represented for parameter P which denotes Lightness and/or Chroma



Gamut Mapping, Fig. 9 A generic framework for spatial gamut mappers

Open Problems

The rendering intents as specified by the ICC say little about how the mapping needs to be performed [5]. Open questions like the following still remain unanswered:

- *What must be done when there is a combination of the rendering intents?* This is a relatively common problem when a pictorial image containing the logo of a corporation needs to be gamut mapped – the logo needs to be accurate while the image may need to be perceptually mapped.
- *What attributes of the appearance of colors must be used?* Colors may be defined by their Lightness, Chroma, and Hue; or by their brightness, colorfulness, and Hue. The choice of which triplet needs to be used will help determine the color space that is to be chosen for gamut mapping.
- *What color space is best suited to perform gamut mapping?* In the case of colorimetric intents, it is relatively clear that the tristimulus values need to be maintained/adapted, and the chromatic-adaptation transform is specified to be the linear Bradford model. Even if one were to know clearly which appearance attributes to use, for each set of appearance attributes, and application, different color spaces lend themselves differently. More importantly, different color spaces provide different “predictors” for these appearance attributes and the gamut mapping algorithm needs to rely on the accuracy of these predictors.

Interested readers are referred to works by various authors in the general field of color imaging and reproduction that address not just the immediate needs of gamut mapping algorithms but the more important aspect of their interaction with the larger image and color processing chain in typical capture, display, and print systems [2, 4, 6].

References

1. Balasubramanian R, deQueiroz R, Eschbach R, Wu W (2000) Gamut mapping to preserve spatial luminance variations. *J Imaging Sci Technol*, 45(Part 5): 436–443
2. Hunt RWG (2004) Reproduction of colour, 6th edn. Wiley, Chichester
3. International Color Consortium® (2004) Specification ICC.1:2004-10 (Profile Version 4.2.0.0) Image technology

- colour management – Architecture, profile format, and data structure. International Color Consortium, Reston, VA
4. Morovic J (2008) Color gamut mapping. Wiley, Chichester
 5. Morovic J, Luo MR (2001) The fundamentals of gamut mapping: a survey. *J Imaging Sci Technol* 45(3):283–290
 6. Sharma G (eds) (2002) Digital color imaging handbook. CRC, Hoboken

Gaze Control

- [Active Sensor \(Eye\) Movement Control](#)

Generalized Bas-Relief (GBR) Transformation

- [Bas-Relief Ambiguity](#)

Generic Object Recognition

Svetlana Lazebnik

Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA

Synonyms

- [Object class recognition \(categorization\)](#)

Related Concepts

- [Object Class Recognition \(Categorization\)](#) ► [Object Detection](#)

Definition

Generic object recognition is the problem of identifying the category membership of an object contained in a photograph. This term is largely synonymous with object class recognition (categorization), though it places a greater emphasis on approaches aimed at recognizing a broad range of natural categories, as opposed to object instances or specialized categories (e.g., faces).

Background

Historically, much of the effort in the field of object recognition has focused on identifying particular object instances or members of a few select categories, such as faces, cars, and pedestrians. However, in the last few years, the community has started moving towards the goal of building *general-purpose* recognition systems whose ability to identify everyday categories matches that of humans. This is a big challenge: according to cognitive scientists, human beings can identify around 3,000 entry-level categories and 30,000 visual categories overall [1], and the number of categories distinguishable with domain expertise may be on the order of 10^5 [2].

Theory

For a summary of the major strategies for modeling general visual categories, refer to the entry on object class recognition (categorization). Apart from the representational issues discussed in that entry, computational scalability issues are central to generic object recognition (see, e.g., [3]). To achieve scalable and accurate recognition in the many-class setting, it is important to have a good model of the semantic, visual, and structural relationships between the different categories. In particular, organizing classes into visual or semantic hierarchies [4, 5] allows the complexity of recognition to scale sublinearly in the number of classes, makes the system less prone to confusion between unrelated classes, and enables it to perform cross-category generalization (e.g., an unfamiliar type of a car may still be recognized as a vehicle). Another promising strategy for cross-category generalization is the learning of attributes [6]. For example, the system may not have been trained to recognize a tapir, but it may still be able to infer that it is a four-legged herbivorous mammal. More generally, *transfer learning*, or the sharing of knowledge between related categories, is likely to be a key component of an effective generic recognition system, enabling it to acquire new models from very few images (see, e.g., [7]).

Progress in generic object recognition crucially depends on the availability of diverse and large-scale datasets. Fortunately, beginning with the Caltech-101 dataset [8], the field has seen a rapid succession of datasets containing ever increasing numbers of images

and categories. Notable current datasets include Caltech-256 [9], LabelMe [10], Tiny Images [11], the PASCAL Visual Object Classes Challenge [12], and ImageNet [13]. In particular, ImageNet contains over 11 million images and over 15,000 concepts organized in a hierarchy, making it a good testbed for new general-purpose recognition algorithms.

References

1. Biederman I, Mezzanotte R, Rabinowitz J (1982) Scene perception: Detecting and judging object undergoing relational violations. *Cogn Psychol* 14:143–147
2. Perona P (2009) Visual recognition circa 2008. In: Dickinson S, Leonardis A, Schiele B, Tarr M (eds) *Object categorization: computer and human vision perspectives*. Cambridge University Press, Cambridge/New York, pp 55–68
3. Deng J, Berg AC, Li K, Li FF (2010) What does classifying more than 10,000 image categories tell us? European conference on computer vision (ECCV), Heraklion
4. Marszałek M, Schmid C (2008) Constructing category hierarchies for visual recognition. European conference on computer vision (ECCV), Marseille, pp 479–491
5. Marszałek M, Schmid C (2007) Semantic hierarchies for visual object recognition. IEEE conference on computer vision and pattern recognition (CVPR), Minneapolis, pp 1–7
6. Farhadi A, Endres I, Hoiem D, Forsyth D (2009) Describing objects by their attributes. IEEE conference on computer vision and pattern recognition (CVPR), Miami, pp 1778–1785
7. Fei-Fei L, Fergus R, Perona P (2006) One-shot learning of object categories. *IEEE Trans Pattern Anal Mach Intell* 28(4):594–611
8. Fei-Fei L, Fergus R, Perona P (2004) Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. IEEE CVPR workshop on generative-model based vision. http://www.vision.caltech.edu/Image_Datasets/Caltech101
9. Griffin G, Holub AD, Perona P (2006) The Caltech-256. Caltech technical report. http://www.vision.caltech.edu/Image_Datasets/Caltech256/
10. Russell B, Torralba A, Murphy K, Freeman WT (2008) LabelMe: a database and web-based tool for image annotation. *Int J Comput Vis* 77(1–3):157–173
11. Torralba A, Fergus R, Freeman WT (2008) 80 million tiny images: a large dataset for non-parametric object and scene recognition. *IEEE Trans Pattern Anal Mach Intell* 30(11):1958–1970
12. Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A (2010) The Pascal visual object classes (VOC) challenge. *Int J Comput Vis* 88(2):303–338. <http://pascallin.ecs.soton.ac.uk/challenges/VOC/>
13. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) ImageNet: a large-scale hierarchical image database. IEEE conference on computer vision and pattern recognition (CVPR), Miami, pp 248–255

Generically Describable Situation

► [Situation Graph Trees](#)

Geodesic Active Contour

► [Geodesics, Distance Maps, and Curve Evolution](#)

Geodesics, Distance Maps, and Curve Evolution

Ron Kimmel

Department of Computer Science, Technion – Israel Institute of Technology, Haifa, Israel

Synonyms

[Geodesic active contour](#); [Segmentation](#); [Variational methods](#)

Related Concepts

► [Edge Detection](#); ► [Semantic Image Segmentation](#)

Definition

Geodesics are locally the shortest paths in some space. Equivalently, geodesics are curves for which small perturbation increases their length according to some measure. A minimal geodesic corresponds to the shortest geodesic connecting two points. In computer vision and pattern recognition, the way distance is measured and the resulting geodesics define the specific application, see [1] for a pedagogical introduction to the field of numerical geometry of images.

Background

Edge detectors can be defined from a variational perspective where an edge is a curve along which the image gradient aligns with the curve's normal [2–6].

These types of edges are geodesics in the sense of integrating the inner product between the curve's normal and the image gradient and by Green's Theorem can be shown to be the Marr-Hildreth edge detectors [7], also known as LZC or Laplacian zero crossings. In fact, the more sophisticated Canny-Haralick [8, 9] edge detector can be viewed in a similar geometric-variational manner [3].

The first to formulate computer vision problems from a variational axiomatic view point was probably Horn [10]. For example, shape from shading, the problem of shape reconstruction from an intensity image, can be casted into a minimal geodesic computation problem. In a similar fashion, the image domain can be separated into meaningful regions by treating the image either as a set of pixels, each tries to place itself in the right segment, or as contours that attempt to locate themselves along the boundaries between objects. These kinds of contours live somewhere between histogram segmentation and edge detection and could be referred to as *edge integrators*. The *geodesic active contour* (GAC) model was first presented in [11]; see also [12] for a link between vector quantization and the GAC model. It is a computational framework to find local geodesics in a domain weighted by some edge indicator function. Its first implementation was via the Osher-Sethian level set framework [13]. Minimal geodesics in that metric were shown to be the solution of an eikonal equation [14] that could be efficiently solved by the *fast marching method* [15, 16]; see also its extension to surfaces [17].

Eikonal solvers compute distance maps in a specific metric. Shape morphology, which defines algebra of shapes via Minkowski operations on sets of points, can also be related to distance maps. For an intuitive explanation, one can think of adding two shapes as the process of adding two numbers A and B . The result can be thought of as adding n times the number $\frac{1}{n}B$ to A . As $n \rightarrow \infty$, addition can be considered as a differential process. In a similar manner, erosion and dilation operations in mathematical morphology of shapes have a differential interpretations. Its continuous formulation was put forward in a number of publications [18–20] relating to level sets and eikonal solvers.

The *geodesic active contour* model evolves a given contour into a local geodesic via a heat flow in the space of weighted contours. Other heat flows evolve

the image level sets themselves towards geodesics in Euclidean, affine, or perspective geometries. For example, the curvature scale space [21, 22] and the affine scale space [23, 24] can simplify the image structure in a topology preserving manner, where topology refers to the connectivity of the image level sets.

Theory

Consider the *geodesic active contour* functional

$$E(C) = \int g(C(s))ds, \quad (1)$$

where g is some edge indicator function, e.g., $g(x, y) = (|\nabla I| + \epsilon)^{-1}$, and $C(s) = \{x(s), y(s)\}$ is an arclength parametrized planar contour, for which the first variation is given by

$$\frac{\delta E}{\delta C} = -(\kappa g - \langle \nabla g, \mathbf{n} \rangle) \mathbf{n}, \quad (2)$$

where \mathbf{n} is the curve's normal and κ its curvature. One could either find geodesics under that metric by a gradient descent process $C_t = -\frac{\delta E}{\delta C}$, that in a level set formulation reads

$$\phi_t = \operatorname{div} \left(g \frac{\nabla \phi}{|\nabla \phi|} \right) |\nabla \phi|.$$

Here, ϕ is an implicit representation of the curve C . The way to find minimal geodesics under the same metric would be by solving the eikonal equation $|\nabla u| = g$ with $u(p) = 0$ at some source point p in the image domain, and then extract the minimal geodesic by gradient descent flow $C_t = -\nabla u$ from some target point q to the source point p .

Interesting Measures

In the shape from shading problem, the surface $z(x, y)$ is reconstructed from the shading image $I(x, y) = \langle N, l \rangle$, where N is the surface normal and l is the light source direction. For $l = \hat{z}$, one needs to solve for z in the eikonal equation

$$|\nabla z| = \sqrt{\frac{1 - I^2}{I^2}}$$

The Marr-Hildreth edge detector is given by the extremal curves of

$$E(C) = \int \langle n, \nabla I \rangle ds,$$

for which the first variation vanishes for $\Delta I = 0$, while the Canny-Haralick edge detector integrates the extremal curves of

$$E(C) = \int \langle n, \nabla I \rangle ds - \iint_{\Omega_C} \kappa(I) |\nabla I| dx dy.$$

In this case the first variation w.r.t. C (provided by Euler-Lagrange equation) vanishes for $I_{\xi\xi} = 0$. Equivalently, edges are found where the second derivative of the image along the image gradient direction $\xi = \frac{\nabla I}{|\nabla I|}$ is equal to zero.

Minimizing the Euclidean arclength $s = \int |C_p| dp$ leads to the curvature flow, $C_t = C_{ss}$ or equivalently $C_t = \kappa \mathbf{n}$, that for an image simultaneous evolution of all its level sets reads

$$I_t = \operatorname{div} \left(\frac{\nabla I}{|\nabla I|} \right) |\nabla I|.$$

The Equi-Affine arclength $v = \int |\det(C_p, C_{pp})|^{1/3} dp$ leads to the affine heat flow $C_t = C_{vv}$, or equivalently $C_t = \kappa^{1/3} \mathbf{n}$, that for all image level sets simultaneous evolution is given by

$$I_t = \left(\operatorname{div} \left(\frac{\nabla I}{|\nabla I|} \right) \right)^{1/3} |\nabla I|.$$

Application

Total variation [25] methods in image processing, edge detection and integration, segmentation, structure reconstruction SFS [26, 27], and shape morphology (algebra of shapes) are all applications of computing geodesics and geodesic distances. Each application has its own distance measure and computational preferred flavor.

Open Problems

The theoretical machinery and concepts that motivated the usage of differential geometry and geodesics were adopted to the field of shape matching, processing, and analysis. Further links to metric geometry in general and open problems in this field can be found in [28].

References

1. Kimmel R (2003) Numerical geometry of images: theory, algorithms and applications. Springer, Boston. ISBN:0-387-95562-3
2. Vasilevskiy A, Siddiqi K (2002) Flux maximizing geometric flows. *IEEE Trans Pattern Anal Mach Intell* 24(12): 1565–1578
3. Kimmel R, Bruckstein AM (2003) On regularized Laplacian zero crossings and other optimal edge integrators. *Int J Comput Vis* 53(3):225–243
4. Kimmel R (2003) Fast edge integration. In: Geometric level set methods in imaging, vision, and graphics. Springer, New York, pp 59–77
5. Desolneux A, Moisan L, Morel JM (2003) Variational snake theory. In: Osher S, Paragios N (eds) Geometric level set methods in imaging, vision and graphics. Springer, New York, pp 79–99
6. Fua P, Leclerc YG (1990) Model driven edge detection. *Mach Vis Appl* 3:45–56
7. Marr D, Hildreth E (1980) Theory of edge detection. *Proc R Soc Lond B* 207:187–217
8. Haralick R (1984) Digital step edges from zero crossing of second directional derivatives. *IEEE Trans Pattern Anal Mach Intell* 6(1):58–68
9. Canny J (1986) A computational approach to edge detection. *IEEE Trans Pattern Anal Mach Intell* 8(6):679–698
10. Horn BKP (1986) Robot vision. MIT press. McGraw-Hill Higher Education, New York
11. Caselles V, Kimmel R, Sapiro G (1997) Geodesic active contours. *Int J Comput Vis* 22(1):61–79
12. Chan T, Vese L (1999) An active contour model without edges. In: Scale-space theories in computer vision. Springer, Berlin, pp 141–151
13. Osher SJ, Sethian JA (1988) Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations. *J Comput Phys* 79:12–49
14. Cohen LD, Kimmel R (1997) Global minimum for active contours models: a minimal path approach. *Int J Comput Vis* 24(1):57–78
15. Tsitsiklis JN (1995) Efficient algorithms for globally optimal trajectories. *IEEE Trans Autom Control* 40(9): 1528–1538
16. Sethian JA (1996) A marching level set method for monotonically advancing fronts. *Proc Natl Acad Sci* 93(4): 1591–1595
17. Kimmel R, Sethian JA (1998) Computing geodesic paths on manifolds. *Proc Natl Acad Sci USA* 95(15):8431–8435
18. Brockett RW, Maragos P (1992) Evolution equations for continuous-scale morphology. In: Proceedings IEEE international conference on acoustics, speech, and signal processing, San Francisco, pp 1–4
19. Sapiro G, Kimmel R, Shaked D, Kimia B, Bruckstein AM (1993) Implementing continuous-scale morphology via curve evolution. *Pattern Recognit* 26(9):1363–1372
20. Kimmel R, Sethian JA (2001) Optimal algorithm for shape from shading and path planning. *J Math Imaging Vis* 14(3):237–244
21. Gage M, Hamilton RS (1986) The heat equation shrinking convex plane curves. *J Diff Geom* 23:69–96
22. Grayson MA (1987) The heat equation shrinks embedded plane curves to round points. *J Diff Geom* 26:285–314
23. Alvarez L, Guichard F, Lions PL, Morel JM (1993) Axioms and fundamental equations of image processing. *Arch Ration Mech* 123:199–257
24. Sapiro G, Tannenbaum A (1993) Affine invariant scale-space. *Int J Comput Vis* 11(1):25–44
25. Rudin L, Osher S, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. *Physica D* 60:259–268
26. Bruckstein AM (1988) On shape from shading. *Comput Vis Graph Image Process* 44:139–154
27. Horn BKP, Brooks MJ (eds) (1989) Shape from shading. MIT, Cambridge
28. Bronstein A, Bronstein M, Kimmel R (2008) Numerical geometry of non-rigid shapes. Springer, New York

Geometric Algebra

Leo Dorst

Intelligent Systems Laboratory Amsterdam,
Informatics Institute, University of Amsterdam,
Amsterdam, The Netherlands

Synonyms

[Clifford algebra](#)

Definition

Geometric algebra is an algebra based on a geometric product of vectors in an inner product space (a.k.a. the Clifford product). It naturally extends common geometrical techniques from linear algebra, to process subspaces and operations between them in a direct, coordinate-free manner. It permits division by subspaces (notably vectors), which leads to much more compact expression of common operations and algorithms in linear algebra, and more direct solutions to equations. Geometric algebra includes quaternions, in

a real construction as a ratio of vectors, and extends them to encode rigid body motions. Geometric algebra also extends vector calculus to permit direct differentiation with respect to geometrical quantities and operators, thus allowing classically scalar optimization techniques to be transferred directly to geometric settings.

Background

Geometric algebra (GA) (or Clifford algebra) and its predecessor Grassmann algebra date from the 1870s and 1840s and are therefore older than linear algebra (LA). They formed the first formalization of how to compute with linear subspaces. Simplification of the ideas by Gibbs and others with the intention to construct a compact algebraic system for 3D computations in engineering around 1900 then led to the neglect of these more general frameworks. Now that parts of their structure have effectively been patched onto LA in an *ad hoc* manner with ideas like homogeneous coordinates and quaternions, it may pay to reconsider the use of GA to unify geometric computations. Hestenes was the first to realize this in the 1980s [4], initially for physics. In the 1990s, his ideas percolated into robotics, computer vision, and computer graphics.

Theory

Geometric algebra assumes a vector space V^n with inner product, denoted “.” (aka dot product) between vectors. GA views this dot product as the symmetric part of an underlying *geometric product* denoted “” (a half space, or juxtaposition); the other, anti-symmetric part being the outer product, denoted “ \wedge ”, signifying the *span* of vectors. The axioms of the geometric product are surprisingly few: it is bilinear, associative, distributive over vector addition, scalars commute, and the geometric product of a vector \mathbf{v} with itself is a scalar equal to $\mathbf{v} \cdot \mathbf{v}$.

From this, it follows that for a (non-null) vector \mathbf{v} , the geometric product is invertible with $\mathbf{v}^{-1} = \mathbf{v}/(\mathbf{v} \cdot \mathbf{v})$. This makes division by a vector a permissible operation. That in turn simplifies many geometric expressions. For instance, a reflection of a vector in a plane with normal \mathbf{n} at the origin can be

rewritten in a much simpler form than the classical linear algebra expression:

$$\mathbf{x} \mapsto \mathbf{x} - 2 \frac{\mathbf{x} \cdot \mathbf{n}}{\mathbf{n} \cdot \mathbf{n}} \mathbf{n} = \mathbf{x} - 2 \left(\frac{\mathbf{x}\mathbf{n} + \mathbf{n}\mathbf{x}}{2} \right) \mathbf{n}^{-1} = -\mathbf{n}\mathbf{x}\mathbf{n}^{-1}.$$

Planar reflection is an orthogonal transformation, and concatenating such reflections leads to the general expression of an orthogonal transformation as being characterized as a product of invertible vectors – an algebraic and computationally practical consequence of the famous Cartan-Dieudonné theorem. In particular, a rotation in 3D may be represented as the product of two reflections, i.e., an element $\mathbf{m}\mathbf{n}$ applied in a “sandwiching” manner to a vector:

$$\mathbf{x} \mapsto (\mathbf{m}\mathbf{n})\mathbf{x}(\mathbf{m}\mathbf{n})^{-1}.$$

Taking the normal vectors to be normalized to unity, this element $(\mathbf{m}\mathbf{n})$ has the same algebraic properties as a *unit quaternion*. Quaternions in GA are therefore simply products (or ratios) of real vectors denoting double planar reflections. Parametrizing this product of vectors by their angle $\mathbf{I}\phi/2$ (i.e., $\phi/2$ from \mathbf{n} to \mathbf{m} to produce a rotation over ϕ in that sense), we may write

$$\mathbf{m}\mathbf{n} = \mathbf{n}\cdot\mathbf{m} - \mathbf{n}\wedge\mathbf{m} = \cos(\phi/2) - \mathbf{I}\sin(\phi/2) = e^{-\mathbf{I}\phi/2},$$

where \mathbf{I} is the unit “bivector” $\mathbf{n} \wedge \mathbf{m}$ of the plane span (\mathbf{n}, \mathbf{m}) , with an orientation determined by the order of the spanning vectors. One may verify (by the axioms of the geometric product) that $\mathbf{I}^2 = -1$, so that the “complex” nature of the quaternions is merely a consequence of the axioms of the geometric product of a real vector space. The final expression as an exponent then follows from the usual Taylor series definition of the exponential, extended to GA elements.

This parametrization of an orthogonal transformation as the exponent of a bivector uses the minimal $\frac{1}{2}n(n-1)$ parameters in an n -D vector space. In applications estimating an orthogonal transformation from data, this is more convenient than the usual over-parametrization as an $n \times n$ matrix M , needing to impose orthogonality as the $\frac{1}{2}n(n+1)$ constraints $M^T M = I$.

An element that can be made as the product of unit vectors, or expressed as the exponent of a bivector, is called a *rotor* in GA. In multiplying operators,

this rotor representation is more efficient than matrices (a generalization of the well-known advantage of unit quaternions over rotation matrices). For applying the operator to vectors only, matrices are faster; but rotors can be applied, without modification, to *any* geometric element. The reason behind this is that an orthogonal transformation represented by a rotor V trivially preserves the geometric product in a covariant manner:

$$V(AB)V^{-1} = (VAV^{-1})(VBV^{-1}).$$

As long as we make any construction in GA from a linear combination of geometric products, and as long as we can represent the transformations of interest as rotors, we have a *structure-preserving* implementation of our geometry. As we will see below, this may require choosing a clever embedding relationship between the space in which our geometric elements reside, and the geometric algebra that represents them.

As one multiplies elements of geometric algebra, elements of various *grades* appear. These can be decomposed on a basis consisting of *blades*, which are elements factorizable by the outer product. Such elements represent subspaces of a dimension equal to their grade. For instance, starting from a basis $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ for the vector space \mathbb{R}^3 , one obtains eight basis blades:

$$\left\{ \begin{array}{lll} \underbrace{1,}_{\text{scalar basis}} & \underbrace{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3,}_{\text{vector basis}} & \underbrace{\mathbf{e}_2 \wedge \mathbf{e}_3, \mathbf{e}_3 \wedge \mathbf{e}_1, \mathbf{e}_1 \wedge \mathbf{e}_3,}_{\text{2-blade basis}} \\ & & \\ & \underbrace{\mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3}_{\text{3-blade basis}} \end{array} \right\}.$$

Geometrically, these can be interpreted as the point at the origin, three vector directions, three 2-directions of planar elements, and a volume element. A geometric algebra of an n -dimensional vector space has a basis of 2^n elements (like its Grassmann algebra). The high dimensionality of a geometric algebra is not an argument against its use, since the possibility to generate all structure-preserving interactions between such elements automatically at compile time offsets much run time complexity [3]; also, the geometrically meaningful elements tend to be obtained by

multiplication of basis elements, and those are sparse on this basis. (Incidentally, such multiplicative generation of meaningful elements may be taken as a distinction between geometric algebra and Clifford algebra, since the latter permits arbitrary addition in its theoretical construction.) In fact, the “extra” elements are mostly geometric objects one would make anyway in classical LA, but now integrated in one consistent framework.

Scalar and vector *calculus* is naturally extended to GA elements. This makes it possible to optimize orientation estimations by differentiating a well-chosen cost criterion with respect to rotors, and demanding that the result be zero. Such capabilities are beginning to provide powerful coordinate-free geometrical optimization methods (see, e.g., [10]), in which specific parametrizations can be chosen conveniently afterward.

The Conformal Model

In linear algebra, homogeneous coordinates are an embedding trick that permits us to use the linear transformations of an $(n + 1)$ -dimensional space to represent *projective transformations* of an n -dimensional space computationally. In GA, there is an embedding which allows computing with *conformal transformations* in an n -dimensional space using the rotors (i.e., orthogonal transformations) from the algebra of an $(n + 2)$ -dimensional space [1]. This space has an inner product with signature $(n + 1, 1)$, i.e., one can provide it with an orthonormal basis of which $n + 1$ vectors square to $+1$, and 1 vector squares to -1 . A more commonly used basis is a “null basis” involving the vector n_∞ representing the point at infinity, and n_o , an arbitrarily chosen finite point (which can serve as an origin, similarly to the extra dimensions in homogeneous coordinates). The products of the two non-Euclidean dimensions obey $n_o^2 = n_\infty^2 = 0$, and $n_o \cdot n_\infty = -1$. This, combined with a well chosen embedding of points, permits the space to be an isometric image of Euclidean space (which homogeneous coordinates are not). The resulting geometric algebra is often referred to as CGA (for conformal geometric algebra).

The embedding of Euclidean space into CGA is that a point P at Euclidean location \mathbf{p} relative to n_o

is represented by a vector $p \in \mathbb{R}^{n+1,1}$ given by

$$p = \alpha(n_o + \mathbf{p} + \frac{1}{2}\mathbf{p}^2 n_\infty), \quad \text{with } \alpha \in \mathbb{R}$$

(which may be viewed as an extension of the homogeneous coordinate representation $\alpha(n_o + \mathbf{p})$). Note that p is a null vector: $p^2 = 0$. A simple computation now shows that the squared Euclidean distance between two normalized points is effectively the inner product between their representative vectors:

$$-\frac{1}{2}\|\mathbf{p} - \mathbf{q}\|^2 = \left(\frac{p}{-n_\infty \cdot p}\right) \cdot \left(\frac{q}{-n_\infty \cdot p}\right).$$

The factor $-n_\infty \cdot p$ is the homogeneous weight α of the point representation, so dividing by it provides a normalization. It follows that orthogonal transformations of $\mathbb{R}^{n+1,1}$ which preserve the point at infinity n_∞ preserve Euclidean distances, and hence represent rigid body motions of \mathbb{R}^n . Their rotors (sometimes called *motors*) form a more general form of unit quaternion which can represent rotations around a general axis, combined with a shift along that axis. When n_∞ is not preserved, a rotor of the algebra represents a general conformal transformation.

As to the blades in CGA, a general vector of the space $\mathbb{R}^{4,1}$ (to be specific) represents a Euclidean 3D sphere in a dual manner (with a hyperplane as a special case). A point is merely a dual sphere of zero radius. A 4-blade (i.e., the outer product of four points) represents the oriented sphere through those points, unless one of them is n_∞ , in which case it represents an oriented plane. Similarly, 3-blades are oriented circles or lines, and 2-blades are oriented point pairs. Other useful geometric elements obtain an algebraic definition, such as tangent vectors, tangent planes, and purely directional elements. In CGA, therefore, classical primitives already used effectively in the specification of geometrical algorithms finally obtain an algebraic expression that permits automatic generation of their transformation properties under the rotors representing the operators of interest. The generating capabilities of CGA are quite powerful: e.g., $\mathbf{P} = (\mathbf{X} \cdot \mathbf{A})/\mathbf{A}$ represents the projection of a blade \mathbf{X} onto a blade \mathbf{A} ; if \mathbf{X} is a line and \mathbf{A} is a sphere, then \mathbf{P} is a great circle. Such geometrical nuggets may be found in tutorial texts like [3].

Application

While on paper GA may look more involved than LA, its structure-preserving properties make for cleaner software for geometrical applications; the algebra playing the role – at compiler time if required – of generating the transformation methods for the various geometric data objects automatically [3]. The formalism presumably allows for the expression of all desired geometry, and helpfully hampers the construction of elements without geometric meaning (if certain simple construction rules are followed).

The novel representation of rigid body motions and the associated geometric primitives has already permitted compact treatment of such geometrical computer vision tasks as camera calibration [6, 10], pose estimation [8], and pose interpolation [2]. Image transformations in space and value can be given their own geometric algebra [5] and signal processing [9]. Applications to color processing or quaternionic Fourier transforms are so far less compelling.

Open Problems

Currently, the GA equivalents of a suite of techniques from applied linear algebra are being uncovered and turned into practical tools. These include eigenproblems, singular value decomposition, spectral analysis, calculus (Lagrange optimization) and statistics (geometric noise characterization [7]), and more. Since linear algebra and vector calculus are contained within geometric algebra, one can always defer to known techniques, but the employment of the full suite of GA techniques in sample problems is showing promise.

Moreover, other useful GA models are being identified: for 3D projective geometry, the geometric algebra of 3D lines involving the space $\mathbb{R}^{3,3}$ can represent projective transformations as orthogonal transformations, and hence perform them through structure-preserving rotor computations.

References

- Anglès P (1980) Construction de revêtements du group conform d'un espace vectoriel muni d'une "métrique" de type (p,q). Annales de l'Institut Henri Poincaré Sect A XXXIII:33–51

2. Dorst L, Valkenburg RJ (2011) Square root and logarithm of rotors in 3D conformal geometric algebra using polar decomposition. In: Dorst L, Lasenby J (eds) Guide to geometric in practice. Springer, New York, pp 81–104
3. Dorst L, Fontijne D, Mann S (2009) Geometric algebra for computer science: an object-oriented approach to geometry. Morgan Kaufman, San Francisco
4. Hestenes D, Sobczyk G (1984) Clifford algebra to geometric calculus. Reidel, Dordrecht/Boston
5. Koenderink JJ (2002) A generic framework for image geometry. In: Lasenby J, Dorst L, Doran C (eds) Applications of geometric algebra in computer science and engineering. Birkhäuser, Boston, pp 319–332
6. Lasenby J, Stevenson A (2000) Using geometric algebra for optical motion capture. In: Bayro-Corrochano E, Sobczyk G (eds) Applied clifford algebras in computer science and engineering. Birkhäuser, Boston
7. Perwass C (2009) Geometric algebra with applications in engineering. Springer, Berlin
8. Rosenhahn B, Sommer G (2005) Pose estimation in conformal geometric algebra. part i: The stratification of mathematical spaces; part ii: Real-time pose estimation using extended feature concepts. *J Math Imaging Vis (JMIV)* 22:47–70
9. Sommer G, Zang D (2007) Parity symmetry in multidimensional signals. *Commun pure Appl Anal* 6(3): 829–852
10. Valkenburg RJ, Dorst L (2011) Estimating motors from a variety of geometric data in 3D conformal geometric algebra. In: Dorst L, Lasenby J (eds) Guide to geometric in practice. Springer, London, pp 25–46

Geometric Calibration

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Related Concepts

► [Calibration](#); ► [Calibration of Projective Cameras](#);
► [Camera Calibration](#)

Definition

Geometric calibration is the process of determining the geometric property of a camera such as its intrinsic and extrinsic parameters. It is often referred to as simply *camera calibration* in computer vision. The reader is referred to entry ► [Camera Calibration](#) for a discussion on other camera calibration tasks.

Background

Much work has been done, starting in the photogrammetry community (see [1, 2] to cite a few) and more recently in computer vision ([3–10] to cite a few). According to the dimension of the calibration objects, we can classify those techniques roughly into four categories.

3D reference object-based calibration. Camera calibration is performed by observing a calibration object whose geometry in 3-D space is known with very good precision. Calibration can be done very efficiently [11]. The calibration object usually consists of two or three planes orthogonal to each other. Sometimes, a plane undergoing a precisely known translation is also used [5], which equivalently provides 3D reference points. This approach requires an expensive calibration apparatus and an elaborate setup.

2D plane-based calibration. Techniques in this category requires to observe a planar pattern shown at a few different orientations [12, 13]. Different from Tsai's technique [5], the knowledge of the plane motion is not necessary. Because almost anyone can make such a calibration pattern by himself/herself, the setup is easier for camera calibration.

1D line-based calibration. Calibration objects used in this category are composed of a set of collinear points [14, 15]. As will be shown, a camera can be calibrated by observing a moving line around a fixed point, such as a string of balls hanging from the ceiling.

Self-calibration. Techniques in this category do not use any calibration object and can be considered as 0D approach because only image point correspondences are required. Just by moving a camera in a static scene, the rigidity of the scene provides in general two constraints [9, 16] on the cameras' internal parameters from one camera displacement by using image information alone. Therefore, if images are taken by the same camera with fixed internal parameters, correspondences between three images are sufficient to recover both the internal and external parameters which allow us to reconstruct 3-D structure up to a similarity [17, 18]. Although no calibration objects are necessary, a large number of parameters need to be

estimated, resulting in a much harder mathematical problem.

Other techniques exist: vanishing points for orthogonal directions [19, 20] and calibration from pure rotation [21, 22].

Theory

What follows is an excerpt of the review on camera calibration [23]. The reader is referred to [23] for a complete review of various geometric calibration techniques.

Geometric calibration depends on which camera model is used. The reader is referred to entry ▶ [Camera Model](#) for a presentation of various camera models. In the sequel, we use the perspective camera model.

Camera Calibration with 3D Objects

The traditional way to calibrate a camera is to use a 3D reference object such as those shown in Fig. 1. In Fig. 1a, the calibration apparatus used at INRIA [11] is shown, which consists of two orthogonal planes, on each a checker pattern is printed. A 3D coordinate system is attached to this apparatus, and the coordinates of the checker corners are known very accurately in this coordinate system. A similar calibration apparatus is a cube with checker patterns painted in each face, so in general three faces will be visible to the camera. Figure 1b illustrates the device used in Tsai's technique [5], which only uses one plane with checker pattern, but the plane needs to be displaced at least once with known motion. This is equivalent to knowing the 3D coordinates of the checker corners.

According to the perspective camera model, the relationship between the 3D point \mathbf{M} and its image projection \mathbf{m} is given by

$$\tilde{\mathbf{m}} = \underbrace{\mathbf{A}[\mathbf{R} \quad \mathbf{t}]}_{\mathbf{P}} \tilde{\mathbf{M}} \equiv \mathbf{P}\tilde{\mathbf{M}}, \quad (1)$$

$$\text{with } \mathbf{A} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2)$$

$$\text{and } \mathbf{P} = \mathbf{A}[\mathbf{R} \quad \mathbf{t}] \quad (3)$$

where s is an arbitrary scale factor; (\mathbf{R}, \mathbf{t}) , called the extrinsic parameters, is the rotation and translation which relates the world coordinate system to the camera coordinate system; and \mathbf{A} is called the camera intrinsic matrix, with (u_0, v_0) the coordinates of the principal point, α and β the scale factors in image u and v axes, and γ the parameter describing the skew of the two image axes. The 3×4 matrix \mathbf{P} is called the camera projection matrix, which mixes both intrinsic and extrinsic parameters. Matrix \mathbf{P} is defined up to a scale factor, so it only has 11 free parameters.

Given one 2D-3D correspondence $(\mathbf{m}_i, \mathbf{M}_i)$, we can write down two equations based on Eq. (1). There are in total 11 unknowns (5 intrinsic parameters and 6 extrinsic parameters). If we have 6 or points that do not lie on a single plane, a unique solution is available.

A popular technique in this category consists of four steps [11]:

1. Detect the corners of the checker pattern in each image.
 2. Estimate the camera projection matrix \mathbf{P} using linear least squares.
 3. Recover intrinsic and extrinsic parameters \mathbf{A} , \mathbf{R} , and \mathbf{t} from \mathbf{P} .
 4. Refine \mathbf{A} , \mathbf{R} , and \mathbf{t} through a nonlinear optimization.
- Note that it is also possible to first refine \mathbf{P} through a nonlinear optimization and then determine \mathbf{A} , \mathbf{R} , and \mathbf{t} from the refined \mathbf{P} .

It is worth noting that using corners is not the only possibility. We can avoid corner detection by working directly in the image. In [24], calibration is realized by maximizing the gradients around a set of control points that define the calibration object. Figure 2 illustrates the control points used in that work.

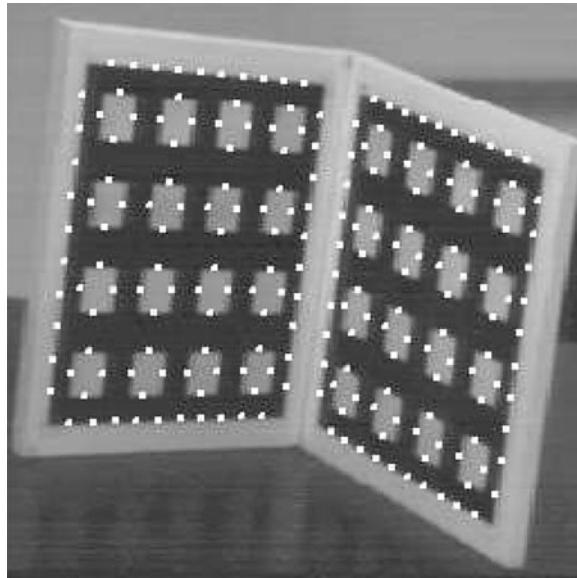
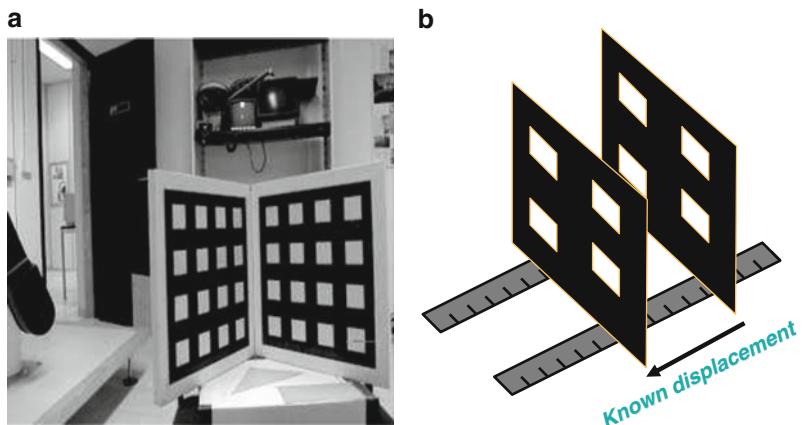
Camera Calibration with 2D Objects: Plane-Based Technique

This technique only requires the camera to observe a planar pattern from a few different orientations [12, 13]. An example is shown in Fig. 3.

Given an image of the model plane, a homography between the known model plane and the image can be estimated. Let us denote it by $\mathbf{H} = [\mathbf{h}_1 \quad \mathbf{h}_2 \quad \mathbf{h}_3]$. Without loss of generality, we assume the model plane is on $Z = 0$ of the world coordinate system. This yields

$$[\mathbf{h}_1 \quad \mathbf{h}_2 \quad \mathbf{h}_3] = \lambda \mathbf{A} [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}],$$

Geometric Calibration, Fig. 1 3D apparatus for calibrating cameras



Geometric Calibration, Fig. 2 Control points used in a gradient-based calibration technique

where λ is an arbitrary scalar. The reader is referred to [12, 25] for the derivation. Using the knowledge that r_1 and r_2 are orthonormal, we have

$$\mathbf{h}_1^T \mathbf{A}^{-T} \mathbf{A}^{-1} \mathbf{h}_2 = 0 \quad (4)$$

$$\mathbf{h}_1^T \mathbf{A}^{-T} \mathbf{A}^{-1} \mathbf{h}_1 = \mathbf{h}_2^T \mathbf{A}^{-T} \mathbf{A}^{-1} \mathbf{h}_2 . \quad (5)$$

These are the two basic constraints on the intrinsic parameters, given one homography. Because a homography has 8 degrees of freedom and there are 6 extrinsic parameters (3 for rotation and 3 for translation),

we can only obtain 2 constraints on the intrinsic parameters.

If the camera is shown with the model plane at N general orientations, we have $3N$ constraints on five intrinsic parameters. With $N \geq 3$, a unique solution is available. There is a degenerate configuration in this technique when planes are parallel to each other. See [15] for a more detailed description.

A recommended calibration procedure is as follows:

1. Print a pattern and attach it to a planar surface.
2. Take a few images of the model plane under different orientations by moving either the plane or the camera.
3. Detect the feature points in the images.
4. Estimate the five intrinsic parameters and all the extrinsic parameters using the closed-form solution.
5. Estimate the coefficients of the radial distortion.
6. Refine all parameters, including lens distortion parameters, through maximum likelihood estimation.

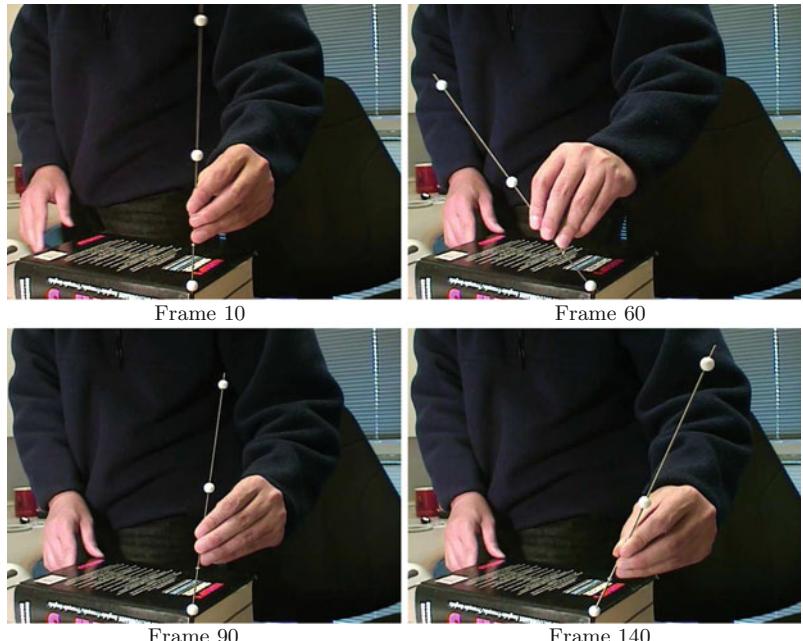
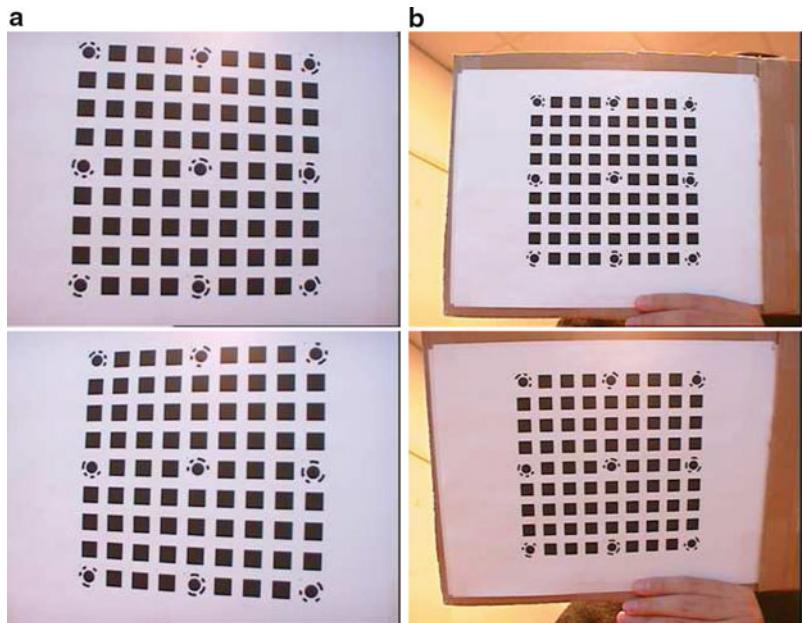
Solving Camera Calibration with 1D Objects

A 1D object consisting of three or more points on a line. An example is shown in Fig. 4.

As discussed in [14, 26], calibration is impossible with a free moving 1D calibration object, no matter how many points on the object. It is, however, possible if the 1D object moves around a fixed point. In the sequel, let the fixed point be point \mathbf{A} , and \mathbf{a} is the corresponding image point. We need three parameters, which are unknown, to specify the coordinates of \mathbf{A} in the camera coordinate system, while image point \mathbf{a} provides two scalar equations according to (Eq. 1).

Geometric

Calibration, Fig. 3 Two sets of images taken at different distances to the calibration pattern. Each set contains five images. On the *left*, three images from the set taken at a close distance are shown. On the *right*, three images from the set taken at a larger distance are shown

**Geometric**

Calibration, Fig. 4 Sample images of a 1D object used for camera calibration

Two Points with Known Distance

They could be the endpoints of a stick, and we move the stick around the endpoint that is fixed. Let \mathbf{B} be the free endpoint and \mathbf{b} , its corresponding image point. For each observation, we need two parameters to define the orientation of the line \mathbf{AB} and therefore the position of \mathbf{B} because the distance between \mathbf{A} and \mathbf{B} is

known. Given N observations of the stick, we have five intrinsic parameters, three parameters for \mathbf{A} and $2N$ parameters for the free endpoint positions to estimate, that is, the total number of unknowns is $8 + 2N$. However, each observation of \mathbf{b} provides two equations, so together with \mathbf{a} we only have in total $2 + 2N$ equations. Camera calibration is thus impossible.

Three Collinear Points with Known Distances

By adding an additional point, say C , the number of unknowns for the point positions still remains the same, that is, $8 + 2N$. For each observation, b provides two equations, but c only provides one additional equation because of the collinearity of a , b , and c . Thus, the total number of equations is $2 + 3N$ for N observations. By counting the numbers, we see that if we have six or more observations, we should be able to solve camera calibration, and this is the calibration technique as developed in [14, 26].

Four or More Collinear Points with Known Distances

Again, the number of unknowns and the number of independent equations remain the same because of invariance of cross ratios. This said, the more collinear points we have, the more accurate camera calibration will be in practice because data redundancy can combat the noise in image data.

Self-calibration

Self-calibration is also called auto-calibration. Techniques in this category do not require any particular calibration object. They can be considered as 0D approach because only image point correspondences are required. Just by moving a camera in a static scene, the rigidity of the scene provides in general two constraints [9, 16, 17] on the cameras' internal parameters from one camera displacement by using image information alone. Absolute conic is an essential concept in understanding these constraints. Therefore, if images are taken by the same camera with fixed internal parameters, correspondences between three images are sufficient to recover both the internal and external parameters which allow us to reconstruct 3-D structure up to a similarity [17, 18]. Although no calibration objects are necessary, a large number of parameters need to be estimated, resulting in a much harder mathematical problem. The reader is referred to two books [27, 28] which provide an excellent recount of those techniques.

Discussions

Although many calibration techniques exist, no single calibration technique is the best for all. It really depends on the situation a user needs to deal with. Following are my few recommendations:

- Calibration with apparatus vs. self-calibration. Whenever possible, if we can pre-calibrate a camera, we should do it with a calibration apparatus. Self-calibration cannot usually achieve an accuracy comparable with that of pre-calibration because self-calibration needs to estimate a large number of parameters, resulting in a much harder mathematical problem. When pre-calibration is impossible (e.g., scene reconstruction from an old movie), self-calibration is the only choice.
- Partial vs. full self-calibration. Partial self-calibration refers to the case where only a subset of camera intrinsic parameters are to be calibrated. Along the same line as the previous recommendation, whenever possible, partial self-calibration is preferred because the number of parameters to be estimated is smaller. Take an example of 3D reconstruction with a camera with variable focal length. It is preferable to pre-calibrate the pixel aspect ratio and the pixel skewness.
- Calibration with 3D vs. 2D apparatus. Highest accuracy can usually be obtained by using a 3D apparatus, so it should be used when accuracy is indispensable and when it is affordable to make and use a 3D apparatus. From the feedback I received from computer vision researchers and practitioners around the world in the last couple of years, calibration with a 2D apparatus seems to be the best choice in most situations because of its ease of use and good accuracy.
- Calibration with 1D apparatus. This technique is relatively new, and it is hard for the moment to predict how popular it will be. It, however, should be useful especially for calibration of a camera network. To calibrate the relative geometry between multiple cameras as well as their intrinsic parameters, it is necessary for all involving cameras to simultaneously observe a number of points. It is hardly possible to achieve this with 3D or 2D calibration apparatus if one camera is mounted in the front of a room while another in the back. An exception is when those apparatus are made transparent; then the cost would be much higher. This is not a problem for 1D objects. We can, for example, use a string of balls hanging from the ceiling.

In this entry, we have only considered the linear projective projection. With real cameras, lens distortion sometimes has to be considered. The reader is referred to the entry ► [Calibration of Projective Cameras](#).

References

1. Brown DC (1971) Close-range camera calibration. *Photogramm Eng* 37(8):855–866
2. Faig W (1975) Calibration of close-range photogrammetry systems: mathematical formulation. *Photogramm Eng Remote Sens* 41(12):1479–1486
3. Gennery D (1979) Stereo-camera calibration. In: Proceedings of the 10th image understanding workshop, Los Angeles, pp 101–108
4. Ganapathy S (1984) Decomposition of transformation matrices for robot vision. *Pattern Recognit Lett* 2:401–412
5. Tsai RY (1987) A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE J Robot Autom* 3(4): 323–344
6. Faugeras O, Toscani G (1986) The calibration problem for stereo. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Miami Beach, FL. IEEE, Washington, DC, pp 15–20
7. Weng J, Cohen P, Herniou M (1992) Camera calibration with distortion models and accuracy evaluation. *IEEE Trans Pattern Anal Mach Intell* 14(10):965–980
8. Wei G, Ma S (1993) A complete two-plane camera calibration method and experimental comparisons. In: Proceedings of the fourth international conference on computer vision, Berlin, pp 439–446
9. Maybank SJ, Faugeras OD (1992) A theory of self-calibration of a moving camera. *Int J Comput Vis* 8(2): 123–152
10. Faugeras O, Luong T, Maybank S (1992) Camera self-calibration: theory and experiments. In: Sandini G (ed) Proceedings of the 2nd European conference on computer vision (ECCV). Lecture notes in computer science, vol 588, Santa Margherita Ligure, Italy. Springer, Berlin, pp 321–334
11. Faugeras O (1993) Three-dimensional computer vision: a geometric viewpoint. MIT, Cambridge
12. Zhang Z (2000) A flexible new technique for camera calibration. *IEEE Trans Pattern Anal Mach Intell* 22(11): 1330–1334
13. Sturm P, Maybank S (1999) On plane-based camera calibration: a general algorithm, singularities, applications. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Fort Collins, Colorado. IEEE Computer Society, Los Alamitos, pp 432–437
14. Zhang Z (2002) Camera calibration with one-dimensional objects. In: Proceedings of the European conference on computer vision (ECCV'02), vol IV, Copenhagen, Denmark, pp 161–174
15. Zhang Z (2004) Camera calibration with one-dimensional objects. *IEEE Trans Pattern Anal Mach Intell* 26(7): 892–899
16. Luong QT (1992) Matrice Fondamentale et Calibration Visuelle sur l’Environnement-Vers une plus grande autonomie des systèmes robotiques. Ph.D. thesis, Université de Paris-Sud, Centre d’Orsay
17. Luong QT, Faugeras O (1997) Self-calibration of a moving camera from point correspondences and fundamental matrices. *Int J Comput Vis* 22(3):261–289
18. Hartley RI (1994) An algorithm for self calibration from several views. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Seattle, WA. IEEE, Los Alamitos, pp 908–912
19. Caprile B, Torre V (1990) Using vanishing points for camera calibration. *Int J Comput Vis* 4(2):127–140
20. Liebowitz D, Zisserman A (1998) Metric rectification for perspective images of planes. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Santa Barbara, California. IEEE Computer Society, Los Alamitos, pp 482–488
21. Hartley R (1994) Self-calibration from multiple views with a rotating camera. In: Eklundh JO (ed) Proceedings of the 3rd European conference on computer vision. Lecture notes in computer science, Stockholm, Sweden, vol 800–801. Springer, New York, pp 471–478
22. Stein G (1995) Accurate internal camera calibration using rotation, with analysis of sources of error. In: Proceedings of the fifth international conference on computer vision, Cambridge, MA, pp 230–236
23. Zhang Z (2004) Camera calibration. In: Medioni G, Kang S (eds) Emerging topics in computer vision. Prentice Hall Professional Technical Reference, Upper Saddle River, pp 4–43
24. Robert L (1995) Camera calibration without feature extraction. *Comput Vis Graph Image Process* 63(2):314–325. Also INRIA technical report 2204
25. Zhang Z (1998) A flexible new technique for camera calibration. Technical Report MSR-TR-98-71, Microsoft Research available together with the software at <http://research.microsoft.com/~zhang/Calib/>
26. Zhang Z (2001) Camera calibration with one-dimensional objects. Technical report MSR-TR-2001-120, Microsoft Research
27. Hartley R, Zisserman A (2000) Multiple view geometry in computer vision. Cambridge University Press, Cambridge
28. Faugeras O, Luong QT (2001) In: Papadopoulos T (ed) The geometry of multiple images. MIT, Cambridge/London

Geometric Fusion

► [Three dimensional View Integration](#)

Geons

Sven J. Dickinson¹ and Irving Biederman^{1,2}

¹Department of Computer Science, University of Toronto, Toronto, ON, Canada

²Departments of Psychology, Computer Science, and the Neuroscience Program, University of Southern California, Los Angeles, CA, USA

Synonyms

[Recognition-by-components \(RBC\) theory](#)

Related Concepts

► Object Class Recognition (Categorization)

Definition

Geons are a set of less than 50 qualitative 2-D or 3-D part classes derived from permuting a set of four dichotomous and trichotomous properties of a generalized cylinder (GC). The values of these properties are nonaccidental in that they can be resolved from a general viewpoint, e.g., whether the axis of a cylinder is straight or curved. Geons were originally introduced by Biederman [9, 10] as the foundation for his recognition-by-components (RBC) theory for human shape perception, whereby object-centered models are represented as concatenations of geons, and object recognition from a 2-D image proceeds by matching recovered parts, typically segmented at regions of matched concavity, and their relations to object models.

superquadric ellipsoids, geons exploited symmetry to reduce the complexity of a generalized cylinder. However, while the superquadric ellipsoid was a metric shape representation, the geon was a qualitative shape categorization. Thus, when a superquadric ellipsoid was recovered from an image, the recovered parameters defined a specific shape (a generative model), whereas when a geon was recovered from an image, it defined a symbolic part class (non-generative category) with only coarse (rather than exact) metric specification.

The appeal of the geon was twofold: (1) its properties were based on the sorts of judgments that humans are very good at, e.g., judging whether a line was straight or curved rather than estimating its exact curvature; and (2) each geon class afforded a high degree of within-class shape deformation, offering great potential for shape categorization and invariance over orientation in depth. Given extensive experiments with humans and primates that lent strong support of his RBC theory, the computer vision community quickly set out to develop computational models for geon recovery from 2-D images.

Background

The concept of modeling an object as a composition of generalized cylinders dates back to Binford [18], who spawned a generation of object-recognition systems based on generalized cylinders, e.g., [20, 44, 45, 51, 54]. Generalized cylinders suffered from unbounded complexity, for arbitrarily complex functions could be used to define the axis, cross section, and sweep functions. As a result, it became popular to restrict the complexity of these functions, e.g., straight axis, constant or linear sweep, rotationally symmetric cross section, in order to facilitate their overconstrained recovery from sparse image data.

In the mid-1980s, two alternative restrictions on generalized cylinders emerged from the computer vision and human vision communities, respectively. Pentland [46] introduced the superquadric ellipsoid to the computer vision community – a 3-D, symmetry-based part representation that afforded a large degree of descriptive power with a small number of parameters. Around the same time, Biederman [9, 10] introduced geons to the human vision community as part of his recognition-by-components (RBC) theory. Like

Theory

Geons define a partitioning of a subspace of the generalized cylinders. Like generalized cylinders, each geon is defined by its axis function, its cross-section function, and its sweep function. Biederman noted that humans are (1) much better at distinguishing between straight and curved lines than they are at estimating curvature; (2) much better at distinguishing parallelism from nonparallel symmetry than they are at estimating the angle between two causally related lines; and (3) good at distinguishing between various types of vertices produced by a cotermination of contours, such as a fork from an arrow from a L-junction. Drawing on these properties of the human visual system, Biederman mapped the spaces of the three generalized cylinder parameters to dichotomous and trichotomous values (Fig. 1):

- *Axis shape*: the axis takes on two possible values: straight or curved.
- *Cross-section shape*: the cross-section shape takes on two possible values: straight-edged or curved-edged.

Geons, Fig. 1 The space of approximately 50 geons is defined by permuting the dichotomous and trichotomous properties of a restricted space of generalized cylinders

The set of geons is generated by variations in the production function for generalized cylinders that produce viewpoint-invariant (= nonaccidental) shape differences

1. Cross Section: Straight vs. Curved



2. Axis: Straight vs. Curved



3. Size of Cross Section:
Constant (parallel sides) vs. Expand vs. Expand & Contract vs Contract & Expand



4. Termination of Geon when Nonparallel: Truncated vs. Pointed vs. Rounded



© Irving Biederman

- *Sweep function*: the cross-section sweep function takes on four possible values: constant, monotonically increasing (or decreasing), monotonically increasing and then decreasing, or monotonically decreasing and then increasing.
- *Termination*: given a nonconstant sweep function, the termination of a geon could be truncated, end in a point (projects into an L-vertex), or end as a curved surface.

Originally, Biederman [10] posited cross-section symmetry as another attribute (with three possible values: rotationally symmetric, possessing an axis of reflective symmetry, or asymmetry) but that attribute was dropped as experiments showed that people assume symmetrical cross-sections, even when the cross-section is asymmetrical (as with an airplane wing).

Permuting the possible values of these four functions defines a space of $2 \times 2 \times 4 \times 3 = 48$ 3-D geons, as illustrated in Fig. 1. Adding 2D geons, e.g., circle, quadrilateral, and triangle, and subtracting the eight instances of constant sweep (2 axis shape \times 2 cross-section shape \times 2 point and curved terminations) when the sweep function is constant brings the total to about 50.

Related Work

Hummel et al. [37, 38] first proposed a connectionist model for recovering geons from line drawings

that achieved invariance to viewpoint. In the computer vision community, Bergevin and Levine were the first to propose a computational model for geon recovery and geon-based recognition [4–8]. Dickinson et al. [27–29] introduced a hybrid object representation combining 3-D object-centered volumetric parts and 2-D viewer-centered aspects modeling the parts. While the framework was applicable to any vocabulary of volumetric parts, it was demonstrated on a qualitative shape vocabulary very similar to geons. Many geon-based frameworks followed, including probabilistic approaches [39], logic-based approaches [32], parametric geon recovery from range data [26, 48, 53], deformable contour-based approaches [47], deformable volume-based approaches [25], and active vision approaches [24, 31]. See [23] for a panel discussion on the strengths and weakness of geons and the challenges that lie ahead.

Open Problems

Geons have tremendous potential as a part representation in support of object categorization. They are qualitative and can support a high degree of within-class deformation, they (like generalized cylinders) map to the natural part structure of objects (when such elongated part structure exists), they are viewpoint-invariant 3-D parts that support object-centered 3-D models (which, in turn, better support scaling to large

databases), and there is psychophysical support for them (the human is still, by far, the best example of an object categorization system). Despite these advantages, geons declined as a subject of study in the computer vision community in the late 1990s, in part due to the advent of appearance-based recognition and a general movement away from shape features.

The main reason for their decline was not necessarily a shortcoming of the representation, i.e., geons, but rather the community's inability to extract qualitative shape from real images of real objects. Except for those approaches operating on range images, the work reviewed above operated on either line drawings or uncluttered scenes containing simple, textureless objects. The key assumption made by these systems was that a salient contour in the image maps one-to-one to a salient surface discontinuity (or occluding contour) on a geon. Unfortunately, in a real scene, objects contain texture, shadows, reflectance contours, and structural "noise" (surface discontinuities that are not salient with respect to the geon class), all of which introduce unwanted contours. Moreover, images of contours (both good and bad) may be broken or noisy, requiring complex perceptual grouping and multiscale analysis to restore and capture the salient shape of the contours. Yet despite these conditions, humans and primates have absolutely no trouble distinguishing (or abstracting) those contours that mark orientation and depth discontinuities – the critical contours for geon extraction – from contours reflecting variations in surface texture, color, lighting, shadows, etc.

As discussed in Dickinson [22], the recognition community's gradual movement from shape toward appearance, coupled with the community's interest in engineering practical systems, drew attention away from basic research on shape modeling in support of object categorization. However, the community is once again realizing that over the set of exemplars belonging to an object category, shape is far more invariant than appearance. As a result, shape-based object categorization systems (mainly using contours) are beginning to reemerge, e.g., [33]. But a return to local contour-based features is not sufficient, as local shape features are still too exemplar-specific. Rather, such features must be perceptually grouped and abstracted to form more generic shape structures that offer the within-class deformation invariance required for effective categorization. Geons offer a powerful shape abstraction with great categorization potential, but only when more progress has been made on the mid-level challenges

of perceptual grouping and intermediate-level shape abstraction. Some early work along these lines has started to appear [50].

Experimental Results: Computer Vision

[Figure 2](#) illustrates three examples of geon recovery systems in the computer vision community. In [Fig. 2a](#), the system of Bergevin and Levine [7] recovers geons from line drawings. In [Fig. 2b](#), the system of Dickinson et al. [24] recovers geon-like volumetric parts from real images of simple objects, as does the system of Pilu and Fisher [47], as shown in [Fig. 2c](#).

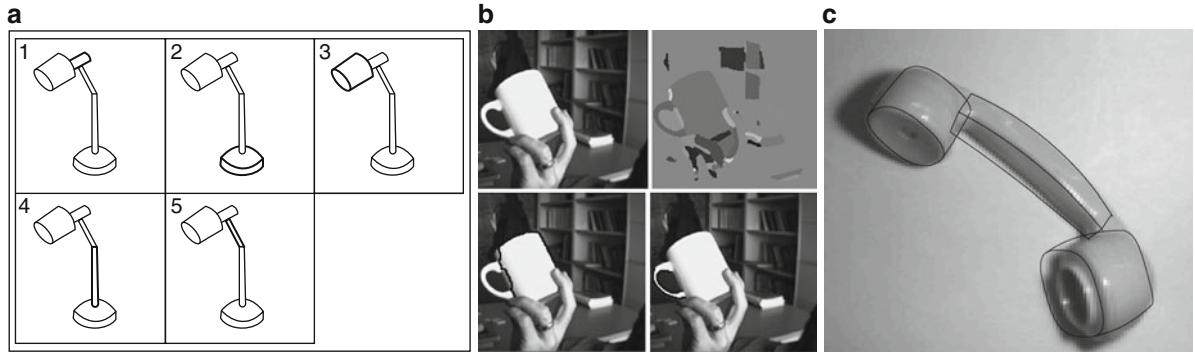
Experimental Results: Human Vision

There is now substantial neural and behavioral evidence for the representation of objects as an arrangement of geons, as specified by the recognition-by-components theory. This evidence can be summarized in terms of six independent assumptions. Any one (or several) of these assumptions can be made independent of RBC but, to date, RBC is the only theory from which all six derive.

The representation of an object is largely edge-based – specifically, those edges specifying orientation and depth discontinuities – rather than surface-based (i.e., color, texture).

Reaction times (RTs) and error rates for naming briefly presented images of objects are as fast for line drawings as they are for full, color photography [16]. This is also true of verification in which the observer verifies whether a name ("chair"), provided prior to an image of an object, matches the object. The equivalence in performance for identifying line drawings and photography is evident even when the objects have a diagnostic color/textured, such as a fish, fork, or banana, as opposed to objects with nondiagnostic surface properties, e.g., a chair or a lamp, which can be any color or texture.

The equivalence of photography and line drawings is also witnessed in fMRI activity where the adaptation (i.e., the reduction) of the BOLD signal that is evident with a repetition of a stimulus, fMRI-a, is the same when the images have the same format, i.e., identical photographs or line drawings, as when they have different formats, one a photograph and the other a line drawing [34]. This invariance to surface properties is



Geons, Fig. 2 Three examples of geon recovery in the computer vision community: (a) decomposing a line drawing of a lamp into its constituent geon parts, with bold contours indicating parts: 1 - rear shade, 2 - base, 3 - front shade, 4 - lower neck, and 5 - upper neck (Bergevin and Levine [7]); (b) from a region segmentation (*upper right*) of the image of an occluded

cup (*upper left*), the two recovered constituent qualitative volumetric parts (with matched contours highlighted in black) are shown in *lower left* (body cylinder) and *lower right* (handle bent cylinder) (Dickinson et al. [24]); and (c) decomposing a phone into its constituent geons parts (Pilu and Fisher [47])

also seen in the response of many single neurons in object-sensitive areas in the macaque [41]. In fMRI, the processing of surface properties, color and texture, activates different cortical areas than those activated when processing shape [21].

There are few transformations to appearance as dramatic as rendering a line drawing from a photograph yet the readily achieved invariance to this transformation poses a major challenge to appearance-based theories of object recognition.

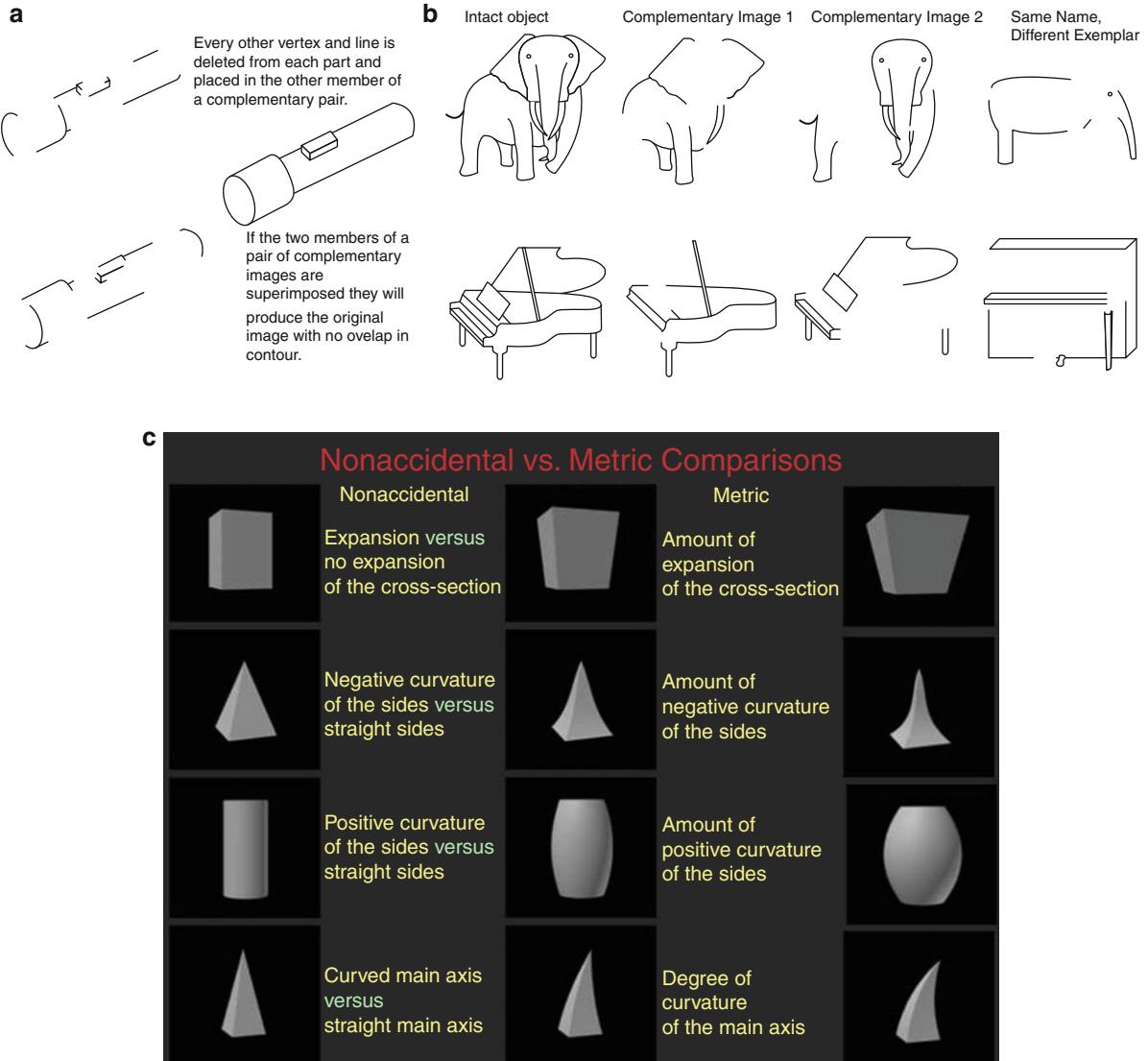
Objects are represented by parts rather than local features, templates, or concepts.

Object priming is the facilitation that ensues as a consequence of a prior perception of an object. It can be readily evidenced by a reduction in RTs and error rates in the naming of brief, masked presentations of objects and has been documented over a 14-month period from the first to second presentation of the images. (The reduction in the magnitude of the BOLD response to a repeated stimulus, termed fMRI adaptation, is generally attributed to more efficient coding and is interpreted as a neural correlate of priming.) Almost all of this priming is visual (i.e., perceptual) rather than lexical (easier access to the name itself) in that an object with the same name but a substantially different shape, e.g., a grand piano followed by an upright piano, evidences almost no facilitation.

Studies with complementary, contour-deleted line drawings document that all the priming can be

attributed to the repetition of the parts (in their appropriate relations) as opposed to local features, i.e., the specific lines and vertices in the image [14]. Thus, if every other vertex and line from each geon is deleted from one image of an object and the deleted contour composes the other member of a complementary pair, as in the two images of a flashlight on the left side of Fig. 3a (so if the two were superimposed they would comprise an intact image with no overlap of contour), the degree of priming between members of a complementary pair – which depict the same parts though with different local contours – is equal to the priming between identical images. This implies that none of the priming can be attributable to the local contours (i.e., the local lines and vertices). Presumably, the local contours are required to activate a representation of the part, but once that part (in its appropriate relations) is activated there is no contribution of the initial local image features.

Instead of deletion of local features, if the deletion is of half the parts of a complex object, as shown in Fig. 3b, then there is no visual priming between members of a complementary pair. Thus the priming is completely dependent on the overlap in the parts in the two images. These effects on behavioral priming have their exact counterpart in fMRI-a. Here, local feature complements show the same reduction in the BOLD response as when the identical images are repeated, suggesting equivalent representations, but repetition of part complements show a complete loss of adaptation



Geons, Fig. 3 Psychophysical evidence in support of Geons: (a) members of a local contour-deleted complementary pair, which have the same parts but different local features, prime each other as much as they do themselves; priming is not attributable to local contours; (b) there is no visual priming between members of a complementary pair when they have no parts in common, as between the images of the second and

third columns [14]; and (c) equal image differences between nonaccidental (between center and left columns) and metric properties (between center and right columns). Geons are distinguished by nonaccidental properties. Discrimination is much faster and more accurate for differences in nonaccidental than metric properties

thus indicating that there is no overlap in visual representations when the images are composed of different parts, even though they are of the same subordinate concept, e.g., both grand pianos [35].

Evidence against a template representation derives from studies of the priming of depth-rotated stimuli.

As long as the same parts can be readily extracted in two different images of the same object, recognition or matching of a rotated object will be achieved with virtually no cost. However, if because of self-occlusion some parts disappear and other parts emerge, then priming is reduced or object matching is impaired [15].

Single cell recordings in the inferior temporal lobe (IT) of the macaque, the area generally accepted to mediate object recognition, generally fire as strongly to one or two of the parts of an object as they do to the complete object [40].

Parts are distinguished by nonaccidental properties (NAPs) and only coarsely by metric properties (MPs).

Values of various dimensions of geons can be regarded as singular or nonsingular. A singular value, such as 0 curvature (i.e., a straight contour), retains that value as the object is rotated in depth. A nonsingular value, such as a nonzero value of curvature (i.e., a curved contour), can vary with the orientation in depth of that contour. In addition to curvature, parallelism of two contours can have a singular value of zero convergence (or divergence) or a nonzero value. Two or three contours that coterminate can be regarded as a singular value of zero separation between their terminations, forming vertices, such as Ls, arrows, or forks. This framework can define NAP differences as the difference between singular and nonsingular values as, e.g., a difference between a curved and a straight contour produced by the parallel sides of the cylinder on the left in the third row of Fig. 3c and the middle barrel. Metric differences are differences in non-singular values, such as two contours with unequal nonzero curvatures, as with the slightly curved and more curved barrels in the third row.

The aforementioned invariance to rotation in depth holds only if the objects that are to be discriminated differ in NAPs [13, 15]. Objects differing only in metric properties incur high costs when they are encountered at a different orientation in depth. At equal orientations, the discrimination of two shapes as being same or different is markedly easier if the shapes differ in NAPs than MPs [17]. Cells in the IT region of the macaque modulate (i.e., vary their firing rate) much more to a change in a NAP compared to an MP [41, 52]. Even pigeons show greater sensitivity to differences in NAPs than MPs [1]. In these comparisons of the sensitivity of NAPs and MPs, the physical differences are equated according to a model of V1 [42], the first stage of cortical shape coding.

Dimensions of generalized cylinders (GCs) are independently coded and have psychophysical and neural reality.

The set of geons is generated by combinations of the values of the independent dimensions shown in

Fig. 1. (In addition, as noted previously, there can be coarse variation in the metric of these geons, such as their aspect ratio or degree of axis curvature.) Are simple object parts actually coded by independent combinations of these dimensions (vs. just being nondimensionalized variations in shape templates)? One measure of independent coding of perceptual dimensions is whether human observers can selectively attend to one dimension without any cost from variations in another, to-be-ignored, dimension. For example, the speed and accuracy of discriminating different shapes is unaffected by whether the colors of those shapes are held constant or varied. It might seem plausible that shape could be attended while ignoring a surface feature such as color. Would efficient selective attention be manifested when observers are attending to one shape dimension, say axis curvature, while ignoring variations in another shape dimension, say aspect ratio. The answer is clearly yes [43]. Moreover, a multidimensional analysis of the firing of a population of IT cells to a set of stimuli similar to that depicted in Fig. 3c shows that 95% of the variance of the spike rates can be modeled in terms of independent coding of the GC dimensions [40].

Low sensitivity for discriminating complex, irregular shapes (= texture?) compared to simple shapes but high sensitivity for distinguishing regular from irregular.

Geons are simple and regular. What about complex, highly irregular objects, such as a bush or a crumpled sweater? It would be highly unlikely that people are employing geons for the precise representation of such objects. Interestingly, the evidence is that people do not represent such variation in any detail beyond the fact that the shapes are irregular and some simple nonaccidental characterizations, e.g., whether the surfaces are round or pointed. This is also true of IT cells [2]. Essentially, objects with irregular parts are treated as texture, rather than shape.

There is a more general point to be made here. GCs (and geons) were criticized for their unwieldiness for modeling objects such as bushes. But this is confusing a graphics system, in which the goal is to achieve an exact replica of the image, with a biological recognition system designed to do basic- and subordinate-level classification in which irrelevant variation is best ignored.

Objects are represented by a structural description that specifies simple parts and relations.

Geons are the representation of the parts of an object, but objects are typically composed of more than one part. In the same manner that people are sensitive to the order of phonemes, so “rough” and “fur” have the same phonemes but in different order, people are sensitive to the arrangement of parts of an object, so they can say, e.g., that a vertical cylinder is attached end-to-middle and perpendicular to the top of a larger horizontal brick. That geons and their relations may be coded independently is documented by a remarkable patient with a left inferior temporal lesion who had no problem distinguishing objects differing in their geons but could not distinguish objects that differed in the relations among the same geons [3]. Recent neuroimaging studies show that such relations are specified explicitly at the same cortical locus, the lateral occipital complex, that object shape is specified [36].

References

1. Lazareva OF, abd Wasserman EA, Biederman I (2008) Pigeons and humans are more sensitive to nonaccidental than to metric changes in visual objects. *Behav Processes* 77:199–209
2. Kayaert G, Biederman I, Vogels R (2005) Representation of regular and irregular shapes in macaque inferotemporal cortex. *Cereb Cortex* 15:1308–1321
3. Behrmann M, Peterson MA, Moscovitch M, Suzuki S (2006) Integrative agnosia: deficit in encoding relations between parts. *J Exp Psychol Human Percept Perf* 32(5):1169–1184
4. Bergevin R, Levine MD (1988) Recognition of 3-D objects in 2-D line drawings: an approach based on geons. Technical report TR-CIM-88, Center for intelligent machines, McGill University, Nov 1988
5. Bergevin R, Levine MD (1989) Generic object recognition: building coarse 3D descriptions from line drawings. In: Proceedings, IEEE workshop on interpretation of 3D scenes, Austin, pp 68–74
6. Bergevin R, Levine MD (1992) Extraction of line drawing features for object recognition. *Pattern Recognit* 25(3): 319–334
7. Bergevin R, Levine MD (1992) Part decomposition of objects from single view line drawings. *CVGIP: Image Underst* 55(1):73–83
8. Bergevin R, Levine MD (1993) Generic object recognition: building and matching coarse 3D descriptions from line drawings. *IEEE Trans Pattern Anal Mach Intell* 15:19–36
9. Biederman I (1985) Human image understanding: recent research and a theory. *Comput Vis Graph Image Process* 32:29–73
10. Biederman I (1987) Recognition-by-components: a theory of human image understanding. *Psychol Rev* 94:115–147
11. Biederman I (1990) Higher level vision. In: An invitation to cognitive science: visual cognition and action, vol 2. MIT, Cambridge, MA, pp 41–72
12. Biederman I (2000) Recognizing depth-rotated objects: a review of recent research and theory. *Spat Vis* 13:241–253
13. Biederman I, Bar M (1999) One-shot viewpoint invariance in matching novel objects. *Vis Res* 39:2885–2899
14. Biederman I, Cooper EE (1991) Priming contour-deleted images: evidence for intermediate representations in visual object recognition. *Cognit Psychol* 23:393–419
15. Biederman I, Gerhardstein PC (1993) Recognizing depth-rotated objects: evidence and conditions for 3D viewpoint invariance. *J Exp Psychol Hum Percept Perform* 19: 1162–1182
16. Biederman I, Ju G (1988) Surface vs. edge-based determinants of visual recognition. *Cognit Psychol* 20:38–64
17. Biederman, I, Yue, X, Davidoff J (2009) Representation of shape in individuals from a culture with minimal exposure to regular simple artifacts: sensitivity to nonaccidental vs. metric properties. *Psychol Sci* 20:1437–1442
18. Binford TO (1971) Visual perception by computer. In: Proceedings, IEEE conference on systems and control, Miami
19. Borges D, Fisher R (1997) Class-based recognition of 3d objects represented by volumetric primitives. *Image Vis Comput* 15(8):655–664
20. Brooks R (1983) Model-based 3-D interpretations of 2-D images. *IEEE Trans Pattern Anal Mach Intell* 5(2): 140–150
21. Cant JS, Goodale MA (2009) Asymmetric interference between the perception of shape and the perception of surface properties. *J Vis* 9(13):11–20
22. Dickinson S (2009) The evolution of object categorization and the challenge of image abstraction. In: Dickinson S, Leonardis A, Schiele B, Tarr M (eds) Object categorization: computer and human vision perspectives. Cambridge University Press, New York, pp 1–37
23. Dickinson S, Bergevin R, Biederman I, Eklundh J-O, Jain A, Munck-Fairwood R, Pentland A (1997) Panel report: the potential of geons for generic 3-D object recognition. *Image Vis Comput* 15(4):277–292
24. Dickinson S, Christensen H, Tsotsos J, Olofsson G (1997) Active object recognition integrating attention and viewpoint control. *Comput Vis Image Underst* 67(3): 239–260
25. Dickinson S, Metaxas D (1994) Integrating qualitative and quantitative shape recovery. *Int J Comput Vis* 13(3): 1–20
26. Dickinson S, Metaxas D, Pentland A (1997) The role of model-based segmentation in the recovery of volumetric parts from range data. *IEEE Trans Pattern Anal Mach Intell* 19(3):259–267
27. Dickinson S, Pentland A, Rosenfeld A (1990) A representation for qualitative 3-D object recognition integrating object-centered and viewer-centered models. In: Leibovic K (ed) Vision: a convergence of disciplines. Springer, New York, pp 398–421
28. Dickinson S, Pentland A, Rosenfeld A (1992) From volumes to views: an approach to 3-D object recognition. *CVGIP: Image Underst* 55(2):130–154

29. Dickinson S, Pentland A, Rosenfeld A (1992) 3-D shape recovery using distributed aspect matching. *IEEE Trans Pattern Anal Mach Intell* 14(2):174–198
30. Du L, Munck-Fairwood R (1993) A formal definition and framework for generic object recognition. In: Proceedings, 8th Scandinavian conference on image analysis, University of Tromsø, Norway
31. Eklundh J-O, Olofsson G (1992) Geon-based recognition in an active vision system. In: *ESPRIT-BRA 3038, vision as process*. Springer-Verlag ESPRIT Series
32. Fairwood R (1991) Recognition of generic components using logic-program relations of image contours. *Image Vis Comput* 9(2):113–122
33. Ferrari V, Jurie F, Schmid C (2010) From images to shape models for object detection. *Int J Comput Vis* 87(3):284–303
34. Grill-Spector K, Kourtzi Z, Kanwisher N (2001) The lateral occipital complex and its role in object recognition. *Vis Res* 41(10–11):1409–1422
35. Hayworth KJ, Biederman I (2006) Neural evidence for intermediate representations in object recognition. *Vis Res* 46:4024–4031
36. Hayworth KJ, Lescroart MD, Biederman I (2011) Visual relation encoding in anterior LOC. *J Exp Psychol Human Percept Perf* 37(4):1032–1050
37. Hummel JE, Biederman I (1992) Dynamic binding in a neural network for shape recognition. *Psychol Rev* 99:480–517
38. Hummel JE, Biederman I, Gerhardstein P, Hilton H (1988) From edges to geons: a connectionist approach. In: *Proceedings, connectionist summer school*, Carnegie Mellon University, pp 462–471
39. Jacot-Descombes A, Pun T (1992) A probabilistic approach to 3-D inference of geons from a 2-D view. In: *Proceedings, SPIE applications of artificial intelligence X: machine vision and robotics*, Orlando pp 579–588
40. Kayaert G, Biederman I, Op de Beeck H, Vogels R (2005) Tuning for shape dimensions in macaque inferior temporal cortex. *Eur J Neurosci* 22:212–224
41. Kayaert G, Biederman I, Vogels R (2003) Shape tuning in macaque inferior temporal cortex. *J Neurosci* 23:3016–3027
42. Lades M, Vorbruggen JC, Buhmann J, Lange J, von der Malsburg C, Wurtz RP, Konen W (1993) Distortion invariant object recognition in the dynamic link architecture. *IEEE Trans Comput* 42:300–311
43. Lescroart MD, Biederman I, Yue X, Davidoff J (2010) A cross-cultural study of the representation of shape: sensitivity to underlying generalized-cone dimensions. *Vis Cogn* 18(1):50–66
44. Marr D, Nishihara H (1978) Representation and recognition of the spatial organization of three-dimensional shapes. *R Soc Lond B* 200:269–294
45. Nevatia R, Binford TO (1977) Description and recognition of curved objects. *Artif Intell* 8:77–98
46. Pentland A (1986) Perceptual organization and the representation of natural form. *Artif Intell* 28:293–331
47. Pilu M, Fisher RB (1996) Recognition of geons by parametric deformable contour models. In: *Proceedings, European conference on computer vision (ECCV)*, LNCS, Springer, Cambridge, UK, April 1996, pp 71–82
48. Raja N, Jain A (1992) Recognizing geons from superquadrics fitted to range data. *Image Vis Comput* 10(3):179–190
49. Raja N, Jain A (1994) Obtaining generic parts from range images using a multi-view representation. *CVGIP: Image Underst* 60(1):44–64
50. Sala P, Dickinson S (2010) Contour grouping and abstraction using simple part models. In: *Proceedings, European conference on computer vision (ECCV)* Crete, Greece, Sept 2010
51. Ulupinar F, Nevatia R (1993) Perception of 3-D surfaces from 2-D contours. *IEEE Trans Pattern Anal Mach Intell* 15:3–18
52. Vogels R, Biederman I, Bar M, Lorincz A (2001) Inferior temporal neurons show greater sensitivity to nonaccidental than metric differences. *J Cogn Neurosci* 13:444–453
53. Wu K, Levine MD (1997) 3-D shape approximation using parametric geons. *Image Vis Comput* 15(2):143–158
54. Zerroug M, Nevatia R (1996) Volumetric descriptions from a single intensity image. *Int J Comput Vis* 20(1/2):11–42

Gesture Recognition

Matthew Turk

Computer Science Department and Media Arts and Technology Graduate Program, University of California, Santa Barbara, CA, USA

Synonyms

[Human motion classification](#)

Definition

Vision-based gesture recognition is the process of recognizing meaningful human movements from image sequences that contain information useful in human-human interaction or human-computer interaction. This is distinguished from other forms of gesture recognition based on input from a computer mouse, pen or stylus, sensor-based gloves, touch screens, etc.

Background

Automatic image-based gesture recognition is an area of computer vision motivated by a range of application areas, including the analysis of

human-human communication, sign language interpretation, human-robot interaction, multimodal human-computer interaction, and gaming. Human gesture has a long history of interdisciplinary study by psychologists, linguists, anthropologists, and others in the context of human communication [9], exploring the role of gesture in face-to-face conversation, universal and cultural aspects of gesture, the influence of gesture in human evolution and in child development, and other topics, going back at least to the work of Charles Darwin with *The Expression of the Emotions in Man and Animals* (1872). Research in computer vision-based gesture recognition began primarily in the 1990s as computers began to be capable of supporting real-time (or *interactive time*, fast enough to support human interaction) processing and recognition of video streams.

Several gesture taxonomies or categorizations have been developed by different researchers that underscore the breadth of the problem in general. Cadoz [4] described three functional roles of human gesture: semiotic (gestures to communicate meaningful information), ergodic (gestures to manipulate the environment), and epistemic (gestures to discover the environment through tactile experience). Most work in automated gesture recognition is concerned with the first role (semiotic gestures), whereas the area of human activity analysis tends to focus on the latter two. Kendon [11] described a gesture continuum, defining five types of gestures: gesticulation, language-like gestures, pantomimes, emblems, and sign languages. Each of these has a varying association with verbal speech, language properties, spontaneity, and social regulation, indicating that human gesture is indeed a complex phenomenon. Gesticulation, defined as spontaneous, speech-associated gesture, makes up a large portion of human gesture and is further characterized by McNeill [14] into four types:

- Iconic – representational gestures depicting some feature of the object, action, or event being described
- Metaphoric – gestures that represent a common metaphor, rather than the object or event directly
- Beat – small, formless gestures, often associated with word emphasis
- Deictic – pointing gestures that refer to people, objects, or events in space or time

These gesture types modify the content of accompanying speech and often help to disambiguate speech, similar to

the role of spoken intonation. Cassell et al. [5] described early research in conversational agents that models the relationship between speech and gesture and generates interactive dialogs between three-dimensional animated characters that gesture as they speak.

Vision-Based Gesture Recognition

Vision-based gesture recognition must detect human movements from image sequences, ideally in real time and independent of the specific user, the imaging condition, the camera viewpoint, clothing and other confusing factors, and the significant variation in how people gesture. Aspects of a gesture that may be critical to its interpretation include spatial information (where the gesture occurs and/or refers to), pathic information (the path a gesture takes), symbolic information (sign(s) made during a gesture), and affective information (the emotional quality of a gesture, which may be related to the speed and magnitude of a gestural act, as well as to facial expression).

A gesture may be considered as a continuous set of movements or as a sequence of discrete poses or postures. Gestures are inherently dynamic and time varying, while postures are specific – and static – configurations; recognizing specific configurations (such as making a “victory sign”) should properly be referred to as *posture recognition*. Analyzing movement (such as dance or general behaviors in a social situation) is generally referred to as *activity analysis* or *activity recognition* [1].

Unless the gestures are constrained to a particular point in time (e.g., with a “push to gesture” functionality), it is necessary to determine when a dynamic gesture begins and ends. This temporal segmentation/detection of gesture is a challenging problem, particularly in less constrained environments where several kinds of spontaneous gestures are possible amidst other non-gestural movement. While temporal detection and segmentation of gestures may be attempted as a first step, other approaches combine spatial (or spatiotemporal) segmentation with recognition [2, 12].

A typical approach to human gesture recognition involves detecting and tracking component body parts, such as hands, arms, head, torso, legs, and feet, based on an articulated body model, and subsequently classifying the movement into one of a set of known

gestures (e.g., [19]). The output of the tracking stage is a time-varying sequence of parameters describing (2D or 3D) positions, velocities, and angles of the relevant body parts and features, possibly including a representation of uncertainty that indicates limitations of the sensor and the algorithms. An alternative is to take a view-based approach, which computes parameters directly from image motion, generally bypassing human body modeling (e.g., [6, 7]).

Hand gestures have received particular attention in gesture recognition, as hands provide the opportunity for a wide range of meaningful gestures, as evidenced by the rich history of human sign languages such as American Sign Language (ASL) (e.g., [18, 20]), and may be convenient for quickly and naturally conveying information in vision-based interfaces (e.g., [3, 8]). Video-only approaches have had limited success, however, due to the complexities of highly articulated hands and skin-on-skin occlusions.

Recently, there has been a significant amount of work in gesture recognition from depth imagery or combinations of video (RGB) and depth data, largely driven by the availability of the Microsoft Kinect sensor (and SDK/toolkit) and the use of body modeling, tracking, and gesture recognition in consumer applications using the Kinect [16]. Other companies are developing new technologies for gesture recognition (e.g., [17] and [13]), as well as for spatial operating environments that leverage tracking and gesture technologies (e.g., [15]).

Open Problems

Gesture recognition is a broadly defined set of problems and challenges, for which there are some domain-specific solutions that are adequate for commercial use; however, the general problems are largely unsolved. At the low level, there are limitations to any choice of sensor type, and work remains to be done on integrating data from multiple sensors. There is no agreement on how to best represent the sensed spatial and temporal information and its relationship to human movement. Temporal segmentation of natural dynamic gestures is unlikely to be solved without a deep understanding of the gesture semantics – i.e., the high-level context in which the gestures take place. Despite the recent impact of depth sensors on this area, the field

is still wide open for solutions that can provide precise and robust gesture recognition in a wide range of environments.

Research in vision-based gesture recognition can be stimulated by the creation and sharing of thorough, annotated data sets that capture a wide range of spontaneous gestures and imaging conditions and by apples-to-apples comparisons such as the recent ChaLearn Gesture Challenges [10].

References

1. Aggarwal JK, Ryoo MS (2011) Human activity analysis: a review. *ACM Comput Surv* 43(3):1–43. ACM
2. Alon J, Athitsos V, Yuan Q, Sclaroff S (2009) A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE Trans Pattern Anal Mach Intell* 31:1685–1699
3. Bretzner L, Laptev I, Lindeberg T (2002) Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering. In: *IEEE conference on automatic face and gesture recognition*. IEEE Computer Society, Los Alamitos
4. Cadoz C (1994) *Les réalités virtuelles*. Dominos/Flammarion, Paris
5. Cassell J, Steedman M, Badler N, Pelachaud C, Stone M, Douville B, Prevost S, Achorn B (1994) Modeling the interaction between speech and gesture. In: *Proceedings of the sixteenth conference of the cognitive science Society*. Lawrence Erlbaum Associates, Hillsdale
6. Cutler R, Turk M (1998) View-based interpretation of real-time optical flow for gesture recognition. In: *Proceedings of the 1998 IEEE conference on automatic face and gesture recognition*, Nara, Japan, 14–16 Apr
7. Darrell TJ, Penland AP (1993) Space-time gestures. In: *IEEE conference on vision and pattern recognition (CVPR)*, New York, NY
8. Freeman WT, Tanaka K, Ohta J, Kyuma K (1996) Computer vision for computer games. In: *2nd international conference on automatic face and gesture recognition*, Killington, VT, USA, pp 100–105
9. <http://www.gesturestudies.com/>
10. <http://gesture.chalearn.org/>
11. Kendon A (1972) Some relationships between body motion and speech. In: Siegman AW, Pope B (eds) *Studies in dyadic communication*. Pergamon Press, New York
12. Kim D, Song J, Kim D (2007) Simultaneous gesture segmentation and recognition based on forward spotting accumulative HMMs. *Pattern Recognit* 40:2012–2026
13. Leap Motion. <http://www.leapmotion.com/>
14. McNeill D (1992) *Hand and mind: what gestures reveal about thought*. University of Chicago Press, Chicago
15. Oblong Industries. <http://www.oblong.com/>
16. Shotton J, Fitzgibbon A, Cook M, Sharp T, Finocchio M, Moore R, Kipman A, Blake A (2011) Real-time human pose

- recognition in parts from a single depth image. In: IEEE conference on computer vision pattern recognition (CVPR). IEEE, Piscataway
17. SoftKinetic. <http://www.softkinetic.com/SoftKinetic.aspx>
 18. Starner T, Weaver J, Pentland A (1998) Real-time American sign language recognition using desk and wearable computer-based video. *IEEE Trans Pattern Anal Mach Intell* 20(12):1371–1375
 19. Turk M (2001) In: Stanney K (ed) *Handbook of virtual environment technology*. Lawrence Erlbaum Associates, Inc.
 20. Vogler C, Goldenstein S (2008) Toward computational understanding of sign language. *Technol Disabil* 20(2): 109–119

Gradient Vector Flow

Chenyang Xu¹ and Jerry L. Prince²

¹Siemens Technology-To-Business Center, Berkeley, CA, USA

²Electrical and Computer Engineering, Johns Hopkins University, Baltimore, MD, USA

Synonyms

GVF

Related Concepts

►Edge Detection

Definition

Gradient vector flow is the vector field that is produced by a process that smooths and diffuses an input vector field and is usually used to create a vector field that points to object edges from a distance.

Background

Finding objects or homogeneous regions in images is a process known as image segmentation. In many applications, the locations of object edges can be estimated using local operators that yield a new image called an edge map. The edge map can then be used to guide a deformable model, sometimes called an active

contour or a snake, so that it passes through the edge map in a smooth way, therefore defining the object itself.

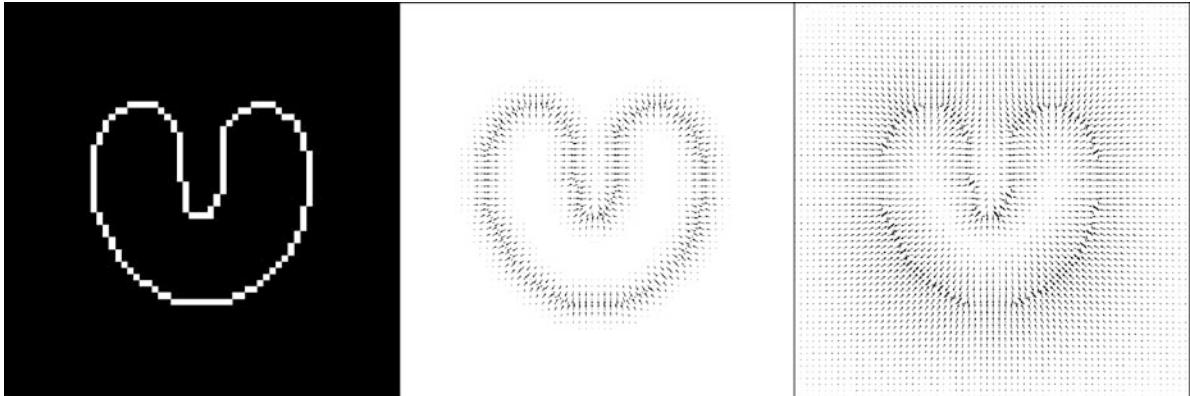
A common way to encourage a deformable model to move toward the edge map is to take the spatial gradient of the edge map, yielding a vector field. Since the edge map has its highest intensities directly on the edge and drops to zero away from the edge, these gradient vectors provide directions for the active contour to move. When the gradient vectors are zero, the active contour will not move, and this is the correct behavior when the contour rests on the peak of the edge map itself. However, because the edge itself is defined by local operators, these gradient vectors will also be zero far away from the edge, and therefore the active contour will not move toward the edge when initialized far away from the edge.

Gradient vector flow (GVF) is the process that spatially extends the edge map gradient vectors, yielding a new vector field that contains information about the location of object edges throughout the entire image domain. GVF is defined as a diffusion process operating on the components of the input vector field. It is designed to balance the fidelity of the original vector field, so it is not changed too much, with a regularization that is intended to produce a smooth field on its output.

Although GVF was designed originally for the purpose of segmenting objects using active contours attracted to edges, it has been since adapted and used for many alternative purposes. Some newer purposes including defining continuous medial axis representation [1], extracting scale-invariant image features [2], regularizing image anisotropic diffusion algorithms [3], finding the centers of ribbon-like objects [4], and much more.

Theory

The theory of GVF was originally described in [5]. Let $f(x, y)$ be an edge map defined on the image domain. For uniformity of results, it is important to restrict the intensities to lie between 0 and 1, and by convention $f(x, y)$ takes on larger values (close to 1) on the object edges. The gradient vector flow (GVF) field is given by the vector field $\mathbf{v}(x, y) = [u(x, y), v(x, y)]$ that minimizes the energy functional



Gradient Vector Flow, Fig. 1 An edge map (*left*) describes the boundary of an object. The gradient of the (blurred) edge map (*center*) points toward the boundary but is very local.

The gradient vector flow (GVF) field (*right*) also points toward the boundary but has a much larger capture range

$$\mathcal{E} = \iint_{\mathbb{R}^2} |\nabla f|^2 |\mathbf{v} - \nabla f|^2 + \mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) dx dy. \quad (1)$$

In this equation, subscripts denote partial derivatives, and the gradient of the edge map is given by the vector field $\nabla f = (f_x, f_y)$. [Figure 1](#) shows an edge map, the gradient of the (slightly blurred) edge map, and the GVF field generated by minimizing \mathcal{E} .

Equation (1) is a variational formulation that has both a data term and a regularization term. The first term in the integrand is the data term. It encourages the solution \mathbf{v} to closely agree with the gradients of the edge map since that will make $\mathbf{v} - \nabla f$ small. However, this only needs to happen when the edge map gradients are large since $\mathbf{v} - \nabla f$ is multiplied by the square of the length of these gradients. The second term in the integrand is a regularization term. It encourages the spatial variations in the components of the solution to be small by penalizing the sum of all the partial derivatives of \mathbf{v} . As is customary in these types of variational formulations, there is a regularization parameter $\mu > 0$ that must be specified by the user in order to trade off the influence of each of the two terms. If μ is large, for example, then the resulting field will be very smooth and may not agree as well with the underlying edge gradients.

Theoretical Solution. Finding $\mathbf{v}(x, y)$ to minimize (Eq. 1) requires the use of calculus of variations since $\mathbf{v}(x, y)$ is a function, not a variable. Accordingly, the Euler equations, which provide the necessary

conditions for \mathbf{v} to be a solution, can be found by calculus of variations, yielding

$$\mu \nabla^2 u - (u - f_x)(f_x^2 + f_y^2) = 0, \quad (2a)$$

$$\mu \nabla^2 v - (v - f_y)(f_x^2 + f_y^2) = 0, \quad (2b)$$

where ∇^2 is the Laplacian operator. It is instructive to examine the form of the equations in (Eq. 2). Each is a partial differential equation that the components u and v of \mathbf{v} must satisfy. If the magnitude of the edge gradient is small, then the solution of each equation is guided entirely by Laplace's equation, for example, $\nabla^2 u = 0$, which will produce a smooth scalar field entirely dependent on its boundary conditions. But the boundary conditions are effectively provided by the locations in the image where the magnitude of the edge gradient is large, where the solution is driven to agree more with the edge gradients.

Computational Solutions. There are several ways to compute GVF. First, the energy function \mathcal{E} itself (Eq. 1) can be directly discretized and minimized, for example, by gradient descent. Second, the partial differential equations in (Eq. 2) can be discretized and solved numerically. The original GVF paper used such an iterative approach, while later papers introduced considerably faster implementation such as an octree-based method [6] and a multi-grid method [7].

Extensions and Advances. GVF is easily extended to higher dimensions. The energy function is readily

written in a vector form as

$$\mathcal{E} = \int_{\mathbb{R}^n} \mu |\nabla \mathbf{v}|^2 + |\nabla f|^2 |\mathbf{v} - \nabla f|^2 d\mathbf{x}, \quad (3)$$

which can be solved by gradient descent or by finding and solving its Euler equation. [Figure 2](#) shows an illustration of a three-dimensional GVF field on the edge map of a simple object (see [8]).

The data and regularization terms in the integrand of the GVF functional can also be modified. A modification described in [9], called *generalized gradient vector flow* (GGVF), defines two scalar functions and reformulates the energy as

$$\mathcal{E} = \int_{\mathbb{R}^n} g(|\nabla f|) |\nabla \mathbf{v}|^2 + h(|\nabla f|) |\mathbf{v} - \nabla f|^2 d\mathbf{x}. \quad (4)$$

While the choices $g(|\nabla f|) = \mu$ and $h(|\nabla f|) = |\nabla f|^2$ reduce GGVF to GVF, the alternative choices $g(|\nabla f|) = \exp\{-|\nabla f|/K\}$ and $h(|\nabla f|) =$

$1 - g(|\nabla f|)$, for K a user-selected constant, can improve the tradeoff between the data term and its regularization in some applications.

The variational formulation of GVF has also been modified in *motion GVF* (MGVF) to incorporate object motion in an image sequence [10]. Whereas the diffusion of GVF vectors from a conventional edge map acts in an isotropic manner, the formulation of MGVF incorporates the expected object motion between image frames.

An alternative to GVF called vector field convolution (VFC) provides many of the advantages of GVF, has superior noise robustness, and can be computed faster [11]. The VFC field \mathbf{v}_{VFC} is defined as the convolution of the edge map f with a vector field kernel \mathbf{k} :

$$\mathbf{v}_{\text{VFC}}(x, y) = f(x, y) * \mathbf{k}(x, y), \quad (5)$$

where

$$\mathbf{k}(x, y) = \begin{cases} m(x, y) \left(\frac{-x}{\sqrt{x^2 + y^2}}, \frac{-y}{\sqrt{x^2 + y^2}} \right) & (x, y) \neq (0, 0) \\ (0, 0) & \text{otherwise} \end{cases} \quad (6)$$

The vector field kernel \mathbf{k} has vectors that always point toward the origin, but their magnitudes, determined in detail by the function m , decrease to zero with increasing distance from the origin.

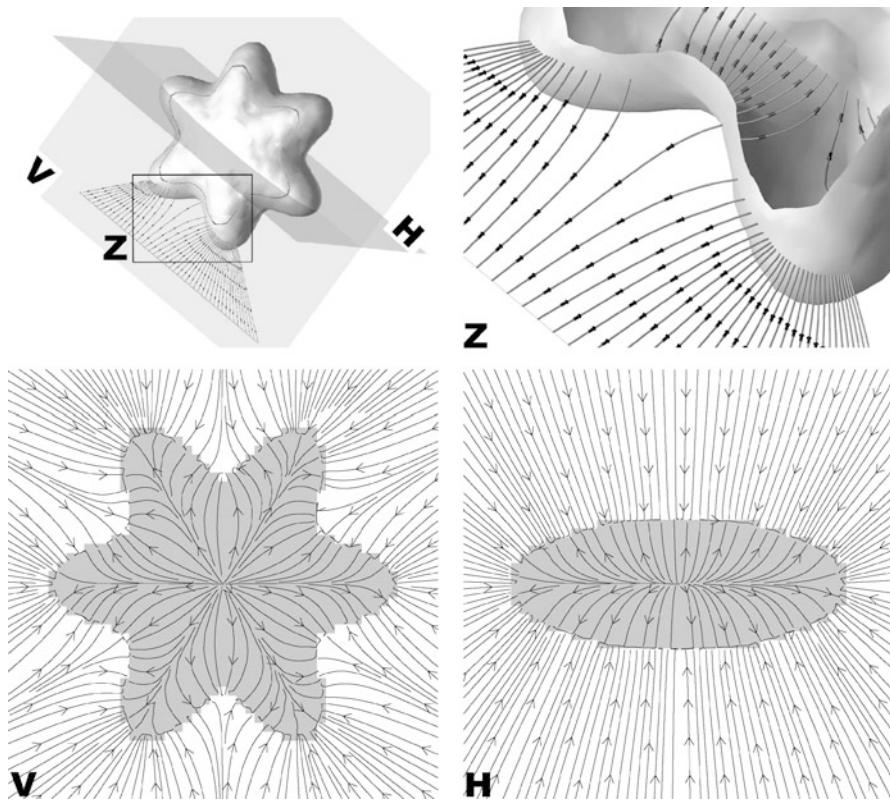
The beauty of VFC is that it can be computed very rapidly using a fast Fourier transform (FFT), a multiplication, and an inverse FFT. The capture range can be large and is explicitly given by the radius R of the vector field kernel. A possible drawback of VFC is that weak edges might be overwhelmed by strong edges, but that problem can be alleviated by the use of a hybrid method that switches to conventional forces when the snake gets close to the boundary.

Properties. GVF has characteristics that have made it useful in many diverse applications. It has already been noted that its primary original purpose was to extend a local edge field throughout the image domain, far away from the actual edge in many cases. This property has been described as an extension of the *capture range* of the external force of an active contour model. It is also capable of moving active contours into concave

regions of an object's boundary. These two properties are illustrated in [Fig. 3](#).

Previous forces that had been used as external forces (based on the edge map gradients and simply related variants) required pressure forces in order to move boundaries from large distances and into concave regions. Pressure forces, also called balloon forces, provide continuous force on the boundary in one direction (outward or inward) and tend to have the effect of pushing through weak boundaries. GVF can often replace pressure forces and yield better performance in such situations.

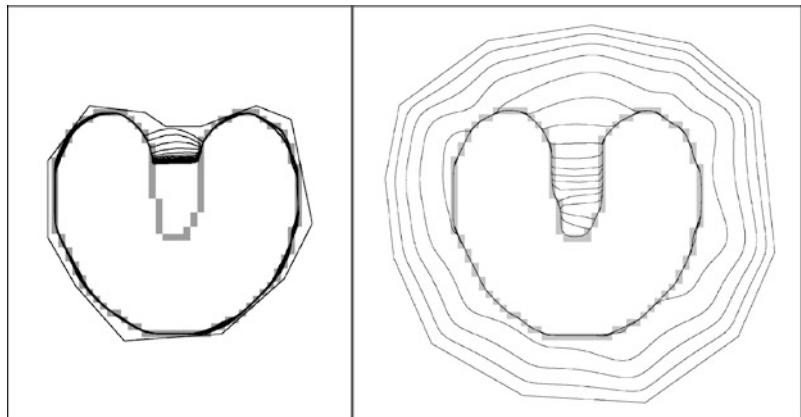
Because the diffusion process is inherent in the GVF solution, vectors that point in opposite directions tend to compete as they meet at a central location, thereby defining a type of geometric feature that is related to the boundary configuration but not directly evident from the edge map. For example, *perceptual edges* are gaps in the edge map which tend to be connected visually by human perception [12]. GVF helps to connect them by diffusing opposing edge gradient



Gradient Vector Flow, Fig. 2 The object shown in the *top left* is used as an edge map to generate a three-dimensional GVF field. Vectors and streamlines of the GVF field are shown in the (*Z*) zoomed region, (*V*) vertical plane, and (*H*) horizontal plane

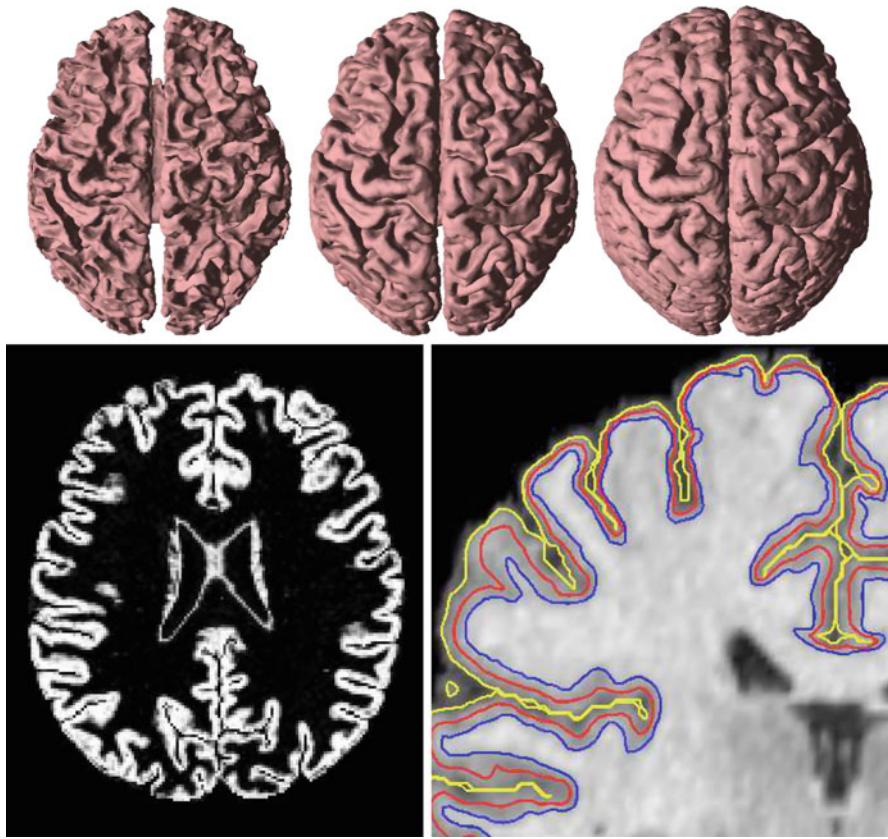
Gradient Vector Flow, Fig. 3

An active contour with traditional external forces (*left*) must be initialized very close to the boundary, and it still will not converge to the true boundary in concave regions. An active contour using GVF external forces (*right*) can be initialized farther away, and it will converge all the way to the true boundary, even in concave regions



vectors across the gap, and even though there is no actual edge map, active contour will converge to the perceptual edge because the GVF vectors drive them there (see [13]). This property carries over when there are so-called *weak edges* identified by regions of edge maps having lower values.

GVF vectors also meet in opposition at central locations of objects, thereby defining a type of medialness. This property has been exploited as an alternative definition of the skeleton of objects [1] and also as a way to initialize deformable models within objects such that convergence to the boundary is more likely.



Gradient Vector Flow, Fig. 4 The inner, central, and outer surfaces of the human brain cortex (*top*) are found sequentially using GVF forces in three geometric deformable models. The central surface uses the gray matter membership function

(*bottom left*) as an edge map itself, which draws the central surface to the central layer of the cortical gray matter. The positions of the three surfaces are shown as nested surfaces in a coronal cutaway (*bottom right*)

Application

The most fundamental application of GVF is as an external force in a deformable model. A typical application considers an image $I(\mathbf{x})$ with an object delineated by intensity from its background. Thus, a suitable edge map $f(\mathbf{x})$ could be defined by

$$f(\mathbf{x}) = \frac{|\nabla(I(\mathbf{x}) * G_\sigma(\mathbf{x}))|}{\max_{\mathbf{x}} |\nabla(I(\mathbf{x}) * G_\sigma(\mathbf{x}))|}, \quad (7)$$

where G_σ is a Gaussian blurring kernel with standard deviation σ and $*$ is convolution. This definition is applicable in any dimension and yields an edge map that falls in the range $[0, 1]$. Gaussian blurring is used primarily so that a meaningful gradient vector can always be computed, but σ is generally kept fairly

small so that true edge positions are not overly distorted. Given this edge map, the GVF vector field $\mathbf{v}(\mathbf{x})$ can be computed by solving (Eq. 2).

The deformable model itself can be implemented in a variety of ways including parametric models such as the original snake [12] or active surfaces and implicit models including geometric deformable models [14]. In the case of parametric deformable models, the GVF vector field \mathbf{v} can be used directly as the external forces in the model. If the deformable model is defined by the evolution of the (two-dimensional) active contour $\mathbf{X}(s, t)$, then a simple parametric active contour evolution equation can be written as

$$\gamma \mathbf{X}_t = \alpha \mathbf{X}_{ss} - \mathbf{v}(\mathbf{X}). \quad (8)$$

Here, the subscripts indicate partial derivatives, and γ and α are user-selected constants.

In the case of geometric deformable models, then the GVF vector field \mathbf{v} is first projected against the normal direction of the implicit wavefront, which defines an additional speed function. Accordingly, then the evolution of the signed distance function $\phi_t(\mathbf{x})$ defining a simple geometric deformable contour can be written as

$$\gamma \phi_t = [\alpha \kappa - \mathbf{v} \cdot \frac{\nabla \phi}{|\nabla \phi|}] |\nabla \phi|, \quad (9)$$

where κ is the curvature of the contour and α is a user-selected constant.

A more sophisticated deformable model formulation that combines the geodesic active contour flow with GVF forces was proposed in [15]. This paper also shows how to apply the Additive Operator Splitting schema [16] for rapid computation of this segmentation method. The uniqueness and existence of this combined model were proven in [17].

GVF has been used to find both inner, central, and central cortical surfaces in the analysis of brain images [4], as shown in Fig. 4. The process first finds the inner surface using a three-dimensional geometric deformable model with conventional forces. Then the central surface is found by exploiting the central tendency property of GVF. In particular, the cortical membership function of the human brain cortex, derived using a fuzzy classifier, is used to compute GVF as if itself were a thick edge map. The computed GVF vectors point toward the center of the cortex and can then be used as external forces to drive the inner surface to the central surface. Finally, another geometric deformable model with conventional forces is used to drive the central surface to a position on the outer surface of the cortex.

References

1. Hassouna MS, Farag AY (2009) Variational curve skeletons using gradient vector flow. *IEEE Trans Pattern Anal Mach Intell* 31(12):2257–2274
2. Engel D, Curio C (2008) Scale-invariant medial features based on gradient vector flow fields. In: 19th IEEE international conference on pattern recognition (ICPR), Tampa, pp 1–4
3. Yu H, Chua CS (2006) GVF-based anisotropic diffusion models. *IEEE Trans Image Process* 15(6):1517–1524
4. Han X, Pham DL, Tosun D, Rettmann ME, Xu C, Prince JL et al (2004) CRUISE: cortical reconstruction using implicit surface evolution. *NeuroImage* 23(3):997–1012
5. Xu C, Prince JL (1998) Snakes, shapes, and gradient vector flow. *IEEE Trans Image Process* 7(3):359–369
6. Esteban CH, Schmitt F (2004) Silhouette and stereo fusion for 3D object modeling. *Comput Vis Image Underst* 96(3):367–392
7. Han X, Xu C, Prince JL (2007) Fast numerical scheme for gradient vector flow computation using a multigrid method. *IET Image Process* 1(1):48–55
8. Xu C, Han X, Prince JL (2008) Gradient vector flow deformable models. In: Bankman I (ed) *Handbook of medical image processing and analysis*, 2nd edn. Academic, Burlington, MA, USA, pp 181–194
9. Xu C, Prince JL (1998) Generalized gradient vector flow external forces for active contours. *Signal Process* 71(2):131–139
10. Ray N, Acton ST (2004) Motion gradient vector flow: an external force for tracking rolling leukocytes with shape and size constrained active contours. *IEEE Trans Med Imaging* 23(12):1466–1478
11. Li B, Acton ST (2007) Active contour external force using vector field convolution for image segmentation. *IEEE Trans Image Process* 16(8):2096–2106
12. Kass M, Witkin A, Terzopoulos D (1988) Snakes: active contour models. *Int J Comput Vis* 1:321–331
13. Xu C, Prince JL (2012) Active contours, deformable models, and gradient vector flow. Online document: GVF online resource including code download. <http://www.iac.ece.jhu.edu/static/gvf/>
14. Xu C, Yezzi A, Prince JL (2000) On the relationship between parametric and geometric active contours and its applications. In: 34th Asilomar conference on signals, systems and computers, Pacific Grove, vol 1, pp 483–489
15. Paragios N, Mellina-Gottardo O, Ramesh V (2004) Gradient vector flow fast geometric active contours. *IEEE Trans Pattern Anal Mach Intell* 26(3):402–407
16. Goldenberg R, Kimmel R, Rivlin E, Rudzsky M (2001) Fast geodesic active contours. *IEEE Trans Image Process* 10(10):1467–1475
17. Guillot L, Bergounioux M (2009) Existence and uniqueness results for the gradient vector flow and geodesic active contours mixed model. *Commun Pure Appl Anal* 8(4):1333–1349

GVF

► Gradient Vector Flow

H

Hand-Eye Calibration

Songde Ma¹ and Zhanyi Hu²

¹Ministry of Science & Technology, Beijing, China
²National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China

Synonyms

Robot-camera calibration; Tracker-camera calibration

Related Concepts

► Camera Calibration

Definition

The hand-eye calibration problem first appeared and got its name from the robotics community, where a camera (eye) was mounted on the gripper (hand) of a robot. The camera was calibrated using a calibration pattern. Then the problem of identifying the unknown transformation from the camera to the hand coordinate system is known as the hand-eye calibration.

Background

There is a strong need for an accurate hand-eye calibration. The reasons are twofold: (i) to map sensor-centered measurements into the robot-world coordinate and (ii) to allow for an accurate prediction

of the pose of the sensor on the basis of the arm motion – in fact these are often complementary aspects of the same problem [1].

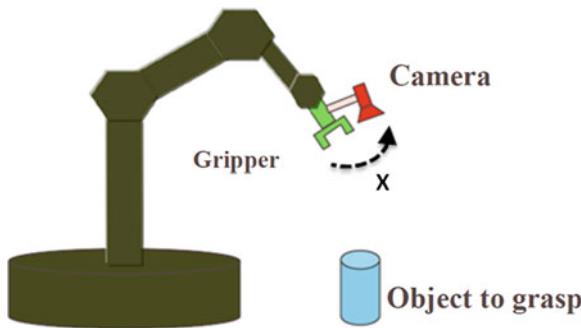
Theory

A typical hand-eye structure is shown in Fig. 1, where a camera is rigidly mounted on the gripper of the robot. In Fig. 1, \mathbf{X} , a 4×4 matrix, denotes the Euclidean transformation between the hand coordinate system and the camera coordinate system, i.e., the hand-eye calibration. Roughly speaking, the hand-eye calibration methods can be divided into two categories: one is to decompose the matrix \mathbf{X} into its rotational and translational parts then optimize the rotation at first, followed by an optimization for the translational part. The other is to optimize the rotation and translation simultaneously.

The standard approach to hand-eye calibration relies on (i) a known reference object and (ii) the possibility to reliably track points on this reference object in order to obtain corresponding points between pairs of images. As shown in Fig. 2, \mathbf{A} , \mathbf{B} , \mathbf{C} , and \mathbf{D} are Euclidean transformation matrices between different coordinate systems, which can be represented by a 4×4 homogeneous matrix as

$$\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^t & 1 \end{bmatrix} \quad (1)$$

where \mathbf{R} is a 3×3 rotation matrix and \mathbf{t} is a 3×1 vector. Matrix \mathbf{A} denotes the transformation between the first and second position of the robot hand (also known as tool center point motion, TCP motion), which could be



Hand-Eye Calibration, Fig. 1 A camera (eye) mounted at the gripper (hand) of a robot

read out from the robot controller. Since the camera is calibrated beforehand, the transformation matrices \mathbf{C} and \mathbf{D} are assumed known, then the motion of the camera, \mathbf{B} , can be expressed as

$$\mathbf{B} = \mathbf{CD}^{-1} \quad (2)$$

From this we have the basic equation of the hand-eye calibration as

$$\mathbf{AX} = \mathbf{XB} \quad (3)$$

A number of approaches have been proposed for the determination of the hand-eye calibration matrix \mathbf{X} from Eq. (3). Here is a brief historical development of the approaches.

1980s

Early solutions decoupled the rotational part of \mathbf{X} from the translational one, yielding some simple, fast, but error-prone formulations, since rotation estimation errors could propagate to the translational part. Shiu et al. [2] used angle-axis representation and least-squares fitting to calculate the rotation then the translation. Tsai et al. [3] gave a closed-form solution using a more efficient linear algorithm. The number of unknowns in their method is unchanged no matter how many measurements are available. Wang [4] compared [2] and [3] with real data and showed that [3] is slight better than [2].

1990s

Zhuang et al. [5] extended and simplified part of the results in [2] by reformulating the solutions for the rotational part of the homogeneous transform equation

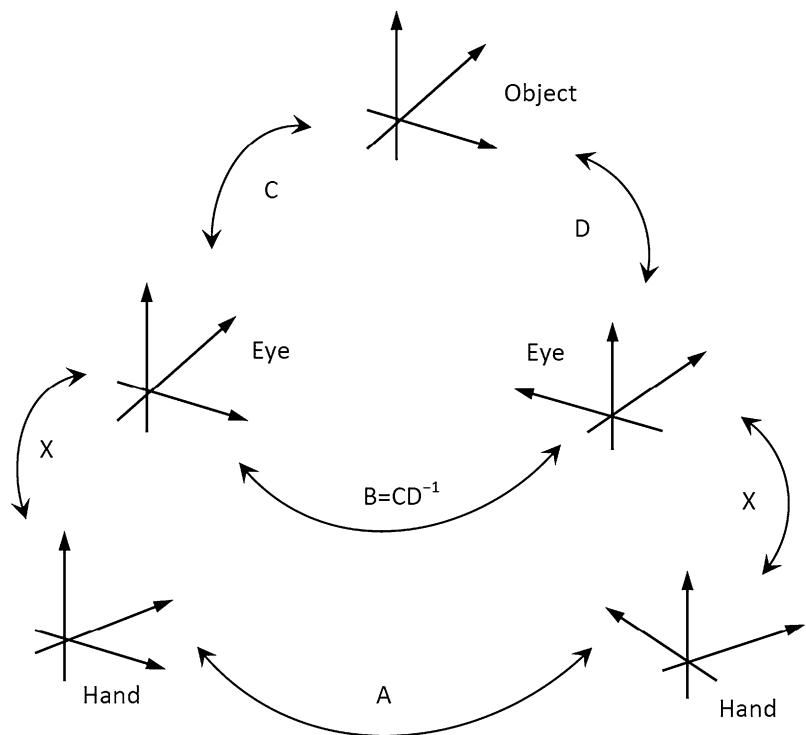
as a quaternion equation. Chou et al. [6] presented another quaternion-based approach where a closed-form solution is obtained using the generalized inverse method with singular value decomposition (SVD). Based on the concept of *screw motion*, Chen [7] did not decouple rotational and translational terms for the first time. Zhuang et al. [8] applied nonlinear optimization directly for \mathbf{X} estimation by minimizing a similar expression to the Frobenius norms of homogeneous matrices of transformation errors. Park et al. [9] performed nonlinear optimization in the same way but again under the decomposed formulation. Lu et al. [10] introduced the eight-space formulation based on quaternions and derived a closed-form least-squares solution using Schur decomposition and Gaussian elimination. Horaud et al. [11] solved simultaneously for rotation and translation using Levenberg–Marquardt technique. Ma [12] gave a linear approach for camera self-calibration and head-eye calibration by controlling the camera platform to undertake at least three orthogonal motions. It is the first approach to combine both camera self-calibration and hand-eye calibration based on active vision concept. Wei et al. [13] nonlinearly minimized algebraic distances and performed a fully automatic hand-eye and camera calibration. Daniilidis [14] introduced the dual quaternions – an algebraic representation of the screw theory to describe motions, which makes it possible to find a fast SVD-based joint solution for rotation and translation.

2000s

Bayro-Corrochano et al. [15] gave an SVD-based linear solution of the coupled problem by the use of motors within the geometric algebra framework. Andreff et al. [16] combined structure from motion with known robot motions for the calibration. They did not enforce the nonlinear orthogonality constraint by increasing the dimensionality of the rotational part and managed to formulate the problem as a single homogeneous linear system. Fassi et al. [17] investigated the standard equation using a geometrical approach and gave some new properties of the equation. Fassi et al. highlighted the reason of over-constrained system when multiple instances of the equation are to be solved simultaneously. Schmidt et al. [18] presented a calibration approach which, in contrast to the standard method, does not require a calibration pattern for determining camera position and orientation.

Hand-Eye Calibration, Fig. 2

Standard approach of hand-eye calibration



Application

Hand-eye calibration is useful in many industrial applications, for instance, grasping objects, visual servoing, robot navigation, et al. For example, in robot-assisted endoscopic surgery [19], the hand-eye transformation has to be estimated every time when the camera head is mounted anew on the endoscope optics, which is done before each operation because it has to be sterilized. Therefore, an automatic and robust hand-eye calibration algorithm is both desirable and welcome.

References

- Strobl KH, Hirzinger G (2006) Optimal hand-eye calibration. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, Beijing, China, pp 4647–4653
- Shiu YC, Ahmad S (1989) Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form $AX=XB$. IEEE Trans Robot Autom 5(1):16–29
- Tsai RY, Lenz RK (1989) A new technique for fully autonomous and efficient 3D robotics hand/eye calibration. IEEE Trans Robot Autom 5(3):345–358
- Wang CC (1992) Extrinsic calibration of a vision sensor mounted on a robot. IEEE Trans Robot Autom 2(8): 161–175
- Zhuang H, Roth ZS (1991) Comments on ‘calibration of wrist-mounted robotic sensors by solving homogenous transform equations of the form $AX = XB$ ’. IEEE Trans Robot Autom 7(6):877–878
- Chou JCK, Kamel M (1991) Finding the position and orientation of a sensor on a robot manipulator using quaternions. Int J Robot Res 10(3):240–254
- Chen HH (1991) A screw motion approach to uniqueness analysis of head-eye geometry. In: Proceedings of the IEEE conference of computer vision and pattern recognition (CVPR), Hawaii, USA, pp 145–151
- Zhuang H, Shiu YC (1993) A noise-tolerant algorithm for robotic hand-eye calibration with or without sensor orientation measurement. IEEE Trans Syst Man Cybern 23(4):1168–1175
- Park FC, Martin BJ (1994) Robot sensor calibration: solving $AX=XB$ on the euclidean group. IEEE Trans Robot Autom 10(5):717–721
- Lu YC, Chou JC (1995) Eight-space quaternion approach for robotic hand-eye calibration. In: Proceedings of the IEEE international conference on systems, man and cybernetics, Vancouver, BC, Canada, pp 3316–3321
- Horaud R, Dornaika F (1995) Hand-eye calibration. Int J Robot Res 14(3):195–210
- Ma SD (1996) A self-calibration technique for active vision systems. IEEE Trans Robot Autom 12(1):114–120
- Wei GQ, Arbter K, Hirzinger G (1998) Active self-calibration of robotic eyes and hand-eye relationships with

- model identification. *IEEE Trans Robot Autom* 14(1): 158–166
14. Daniilidis K (1999) Hand-Eye Calibration Using Dual Quaternions. *International Journal of Robotics Research* 18(3) 286–298
 15. Bayro-Corrochano E, Daniilidis K, Sommer G (2000) Motor algebra for 3D kinematics: the case of the hand-eye calibration. *J Math Imaging Vis* 13(2):79–100
 16. Andreff N, Horaud R, Espiau B (2001) Robot hand-eye calibration using structure-from-motion. *Int J Robot Res* 20(3):228–248
 17. Fassi I, Legnani G (2005) Hand to sensor calibration: a geometrical interpretation of the matrix equation $AX = XB$. *J Robot Syst* 22(9):497–506
 18. Schmidt J, Vogt F, Niemann H (2005) Calibration-free hand-eye calibration: a structure-from-motion approach. *Lect Notes Comput Sci* 3663:67–74
 19. Schmidt J, Vogt F, Niemann H (2003) Robust hand-eye calibration of an endoscopic surgery robot using dual quaternions. *Lect Notes Comput Sci* 2781(DAGM 2003):548–556

Hazard Detection

► [Obstacle Detection](#)

Heterogeneous Parallel Computing

► [High-Performance Computing in Computer Vision](#)

High-Performance Computing in Computer Vision

Guna Seetharaman

Air Force Research Lab RIEA, Rome, NY, USA

Synonyms

[Computing architectures for machine perception](#); [Heterogeneous parallel computing](#)

Definition

The central objective of a computer vision task is to perceive visual data and develop a response. Various processes take place in the *data to decision* chain. High-performance computing refers to

the capacity to achieve a computational task at the required fidelity with minimal resources including time and endurance. Recent explosive growth in video cameras and the resulting ubiquity of visual data broaden the scope of high-performance computer vision well beyond robotics and automation. It is difficult to crisply separate the processing and perception mechanisms underlying this chain. Inspired by neural information processing in the visual system, they are often broadly grouped as low-, intermediate-, and higher-level vision. Low-level vision entails very simple computations applied at each pixel and its immediate neighborhood. Examples include edge detection, texture, order statistics, and optical flow. Intermediate level analysis is concerned with local and regional consistencies including coherence. Examples include contour tracing, connected component labeling, and Hough transforms. These accumulative methods do not fully capture certain evidence-driven assimilative processes, where one or more salient features detected in the scene steer the evidence gathering low-level primitives over the rest of the image. Under special circumstances, the low-level and intermediate-level vision are coupled in an iterative style, giving rise to a recursive model of perceptive processing. Examples include optimal placement of edges, deformable templates, subjective contours, and extraction plus grouping of gestalts. Higher-level processes are defined and analyzed from a broader perspective, facilitating a net perception sufficient to achieve detection, recognition, localization, association, prediction, etc. They drive complex applications such as super resolution, segmentation, reconstruction, object recognition, content-based image retrieval, and tracking etc.

Background

Abstract model of computation is comprised of an arithmetic and logic unit (ALU) to perform a primitive operation, an access mechanism to fetch or modify the operands without imposing limits on the size of the data-set or program, ability to concurrently maintain multiple independent intermediate results impacting the course of execution, and the richness of its instruction set. Parallel computing, pipelined processing, and application specific hardware including heterogeneous and reconfigurable processors have all

been demonstrated as a means to achieve high performance. We will focus our discussion on approaches to achieve high performance in computer vision.

Distinct types of parallel computing models are used to characterize the overall computation required for computer vision: data-level parallelism, cooperating concurrent processes, and competing concurrent threads with and without mutual pre-emption. Computations for low-level vision would follow data parallelism and are efficiently implemented using specialized designs including reconfigurable and pipelined processors to perform vector, block, and stream operations. Modern general purpose graphic processing units (GPUs) and stream and vector processing CPUs are good at this. Intermediate vision processes are efficiently performed using concurrent/multi-threaded processing with multi-core and multi-threaded processors and sufficient cache memory. Higher-level processes [1–3] are well suited for computing clusters consisting of super-scalar processors with node-level capabilities to efficiently process vector and regular gridded data. Most clusters employ message passing mechanisms to move data between processors; interconnection topologies such as bi-orthogonal linear buses, meshes, tori, trees, and hypercubes have all been tried in practice. The need for heterogeneous parallel computing for vision and all aspects of artificial intelligence was envisioned early on [4]. Low-level vision algorithms were developed to take advantage of early parallel processing architectures [5]. Parallel computing algorithms at all levels of the vision hierarchy have been adapted to suit prevailing architectures [6] including the modern CMOS image sensors [7]. Modern day desktop computers make a powerful combination of heterogenous processor, multi-core, and multi-processor systems more accessible and affordable. This has further broadened high-performance computing and shifted the focus on software methods, tools, and versatility of application scenarios.

Images acquired through physical cameras suffer a loss of depth. Computational methods aimed at recovering depth information involve iterative optimization of certain regularized functions which are compute-intensive. They are highly uniform, if not identical, across the entire image, making them a candidates for parallel processing. Vision algorithms are often designed to follow a divide-and-conquer approach. For example, the image is partitioned into four quadrants, each processed independently and the results grouped

by a merging strategy. Each quadrant will in turn be recursively split into sub quadrants for further processing, until it is not necessary to split further. A problem specific preprocessing or postprocessing step is used at the split and merge stages, respectively, and each partition can be processed independently in parallelized fashion. The computation is said to follow an adaptive quad-tree control structure. A seminal algorithm in computer graphics also known as the area-subdivision (Warnock) algorithm [18] and a widely known quad-tree-based image segmentation algorithm [19] follow this structure. Specific aspects of each computation will determine the type of data-flow: either top-down or bottom-up. Certain complex visual tasks involve bidirectional inferencing where one or more salient objects present in the scene steer the way the rest of the scene is processed. Such tasks will sift up and down the quad-tree in cyclical fashion. In general, they will include intra-block computations and adjacent-block data access to apply spatial operators. The interdependence between incremental computations is captured by a hybrid topological structure, a pyramid, which combines a quad-tree with a two-dimensional mesh. Basic computations such as convolution are performed as a systolic or vector operation across the mesh, using single instruction multiple data (SIMD) primitives [8]. They can also be achieved by carefully (de)composed sequence of partial sums and efficiently implemented (on a standard von Neumann machine) as pipelined primitives exploiting the order in which the pixels will be visited. Piecewise affine computations can be used to model generalized geometric deformations and motion fields [20, 21] with efficient implementation using triangular patches. Integral histogram is a powerful emerging low-level processing method for speeding up multi-resolution sliding window-based searching and other computer vision tasks using block-level pipelined computations. Multi-core parallel implementations of integral histograms [9] can significantly scale up computation to large image and video collections.

Classic space filling curves such as Hilbert curves, Peano scans, and Morton curves have been used as a locality preserving pixel organization [10, 11] and an alternative to the standard raster scan which scans the image left to right and top to bottom visiting each pixel exactly once. The spatially compact nature of space filling scans tend to cohere highly correlated pixels as maximal subsequences, producing a significant improvement on aggregative operators such as the run-length coding and compressive sensing.

A recent paper on using spatially coherent data structures for representing matrix equations combines the advantages of a binary-tree control structure and the spatially compact nature of Peano scans to achieve cache-oblivious computations [12] and algorithms on sequential machines. Z-tree is built by inserting a Hilbert curve-scanned sequence of pixel at the terminal level of a binary tree and then building the tree upward. Such a tree represents very large images as a collection of tiles, permitting mesh-algorithms at any given level of the Z-tree and SIMD processing over two or more aligned tiles. Spatio-temporal visualization of gigapixel-sized images and videos can be extended to incorporate such tiled representations [13–15, 22].

Our understanding of human visual perception suggests multiple competing chains of evidence accumulation and incremental inferencing neural processes. The basic unit of such inferences are known as gestalts [16]. They cohere and coalesce into a candidate interpretation, which are indirectly influenced by context and cues about the scene. They account for subjective processes associated with observer expectations. Evidence gathered in one location across an image may at times add to, concur with, be neutral to, or compete against the interpretation arising from elsewhere within the image. A consistent interpretation is achieved through a constraint propagation and resolution method known as relaxation labeling and belief propagation techniques [17]. Such methods have been studied thoroughly in the perception of line drawings of a complex scene of trihedral blocks. Animals that blend with the background such as the well-known dalmatian dog camouflaged against a background of rocks are sometimes used to illustrate the inherent difficulties of separating foreground (the dog) from background based on local contrast or edge structures only. Any cue about the existence of a dog immediately contributes to the coalescing of an otherwise disconnected set of patches into a meaningful grouping in the shape of dog. From a computational standpoint, two approaches have been proposed each with significant success [1]: model-driven computations including hypothesize and verify methods and rule-based methods. The former is built around procedural programs and models that evoke a model driven sequence of tasks and operations. In the latter, a collection of associations between physical scene/object situations and resulting observations across images are combined effectively using rule-based computations to achieve

one of more stable interpretations. These methods lend themselves to parallel computations. Light-weighted threads with carefully orchestrated spatial- and feature-space-based interdependencies will be required to efficiently implement these processes. Modern multi-core heterogeneous processors are more suitable for this.

From a computer engineering perspective, the granularity of the parallelism is of importance for constructing high-throughput dedicated machines. However, from a computer science and computing architecture perspective, one will see that parallel computation for computer vision requires multiple architectures across scale and level of information processing. Topics not covered but deserving of further exploration is focal plane and light-field processing using optical parallel computing and quantum computing for vision algorithms. There is a diversity in the intrinsic parallelism that is seen at each stage of the data-to-decision chain.

References

1. Cantoni V, Levialdi S, Zavidovique B (2011) 3C vision: cues, context and channels. Elsevier, Amsterdam
2. Scientific American (1998) Science's vision: the mechanics of sight.
3. Scientific American (2008) Special report on PERCEPTION: 105 mind-bending illusions
4. Reddy R (1985) Super chips for artificial intelligence. IEEE Int Conf on Solid-State Circuits XXVIII:54–55
5. Bader DA, JáJá J (1996) Parallel algorithms for image histogramming and connected components with an experimental study. *J Parallel Distrib Comput* 35(2):173–190
6. Gross T, O'Hallaron D (1998) iWarp: anatomy of a parallel computing system. MIT, Cambridge
7. Gamal A, Eltoukhy H (2005) CMOS image sensors. *IEEE Circuits Devices Mag* 21(3):6–20
8. Seetharaman G (1995) A simplified design strategy for mapping image processing algorithms on a simd torus. *J Theor Comput Sci* 140(2):319–331
9. Bellens P, Palaniappan K, Badia RM, Seetharaman G, Labarta J (2011) Parallel implementation of the integral histogram. Lecture notes in computer science (ACIVS), vol 6915. Springer, Berlin/New York, pp 586–598
10. Seetharaman G, Zavidovique B (1997) Image processing in a tree of peano coded images. In: Proceedings computer architectures for machine perception (CAMP '97), IEEE computer society, Cambridge, MA, pp 229–234
11. Asano T, Ranjan D, Roos T, Welzl E, Widmayer P (1997) Space-filling curves and their use in the design of geometric data structures. *Theor Comput Sci* 181(1):3–15
12. Heinecke A, Bader M (2009) Towards many-core implementation of LU-decomposition decomposition using peano curves. In: Proceedings of the combined workshop on

- unconventional high performance computing plus memory access workshop. Association for Computing Machinery, New York, pp 21–30
13. Hasler AF, Palaniappan K, Manyin M, Dodge J (1994) A high performance interactive image spreadsheet (IISS). *Comput Phys* 8(4):325–342
 14. Palaniappan K, Fraser J (2001) Multiresolution tiling for interactive viewing of large datasets. In: 17th international AMS conference on interactive information and processing systems (IIPS) for meteorology, oceanography and hydrology, American meteorological society, Albuquerque, pp 338–342
 15. Ponto K, Doerr K, Kuester F (2010) Giga-stack: a method for visualizing giga-pixel layered imagery on massively tiled displays. *Future Gener Comput Systems* 26(5):693–700
 16. Rock I, Palmer S (1990) The legacy of gestalt psychology. *Sci Am* 263(6): 84–90
 17. Grauer-Gray S, Kambhamettu C, Palaniappan K (2008) GPU implementation of belief propagation using CUDA for cloud tracking and reconstruction. In: 5th IAPR workshop on pattern recognition in remote sensing (ICPR), Tampa, FL, pp 1–4
 18. Warnock J.E. (1969) A hidden surface algorithm for computer generated halftone pictures. Ph.D. dissertation. The University of Utah. AAI6919002
 19. Horowitz S.L., Theodosios Pavlidis (1976) Picture segmentation by a tree traversal algorithm. *J. ACM* 23(2): 368–388
 20. Seetharaman G, Gasperas G, Palaniappan K (2000) A piecewise affine model for image registration in nonrigid motion analysis. *IEEE Int. Conf. Image Processing*, 1:561–564
 21. Zhou L, Kambhamettu C, Goldgof D, Palaniappan K, Hasler A. F, (2001) Tracking non-rigid motion and structure from 2D satellite cloud images without correspondences. *IEEE Trans. Pattern Analysis and Machine intelligence*, 23(11):1330–1336
 22. Fraser J, Haridas A, Seetharaman G, Rao R, Palaniappan K (2013) KOLAM: A cross-platform architecture for scalable visualization and tracking in wide-area motion imagery. *Proc. SPIE Conf. Geospatial InfoFusion III (Defense, Security and Sensing: Sensor Data and Information Exploitation)*, 8747:87470N

Histogram

Ying Nian Wu

Department of Statistics, UCLA, Los Angeles,
CA, USA

Definition

Histogram is a graphical display of the distribution of observed values of some random variable.

Theory and Applications

Histogram was introduced by Karl Pearson. In a histogram, the range of the observed values is divided into a number of bins of equal length. A rectangle is erected on top of each bin so that the area of the rectangle equals the frequency that the observed values fall into this bin. A histogram is a more detailed summary of a distribution than mean and variance. It can be considered a nonparametric estimate of the probability density function of the random variable.

In image analysis and computer vision, histograms are often obtained by spatial pooling and they serve as image features. For a texture image, histograms of responses from a bank of filters are pooled over the image domain, and these histograms serve as features that characterize the texture pattern. Histograms can also be pooled within local windows as local texture features. Heeger and Bergen [3] proposed an algorithm for texture synthesis by matching histograms of filter responses. Zhu, Wu, and Mumford [7] proposed a Markov random field model for stochastic textures. The model is the maximum entropy distribution that matches observed marginal histograms of filter responses. The spatially pooled histograms discard the position information. This is appropriate for characterizing texture patterns, which are spatially stationary.

The local orientation histograms of image intensity gradients are key components of two of the most successful image features, namely, SIFT (scale-invariant feature transform) [5] and HoG (histogram of oriented gradients) [2]. Such histograms are pooled within local cells, where each pixel contributes a weighted vote to the histogram bin that corresponds to the orientation of the intensity gradient at this pixel. The weight can be the magnitude of the gradient or some nonlinear transformation of it. Such histograms are very informative descriptions of local image patches, and they are partially invariant to local geometric distortions or shape deformations, because they are spatially pooled within local cells where information about the exact positions of the gradients is discarded.

A more general version of histogram is also used in the bag-of-words method for image classification [1]. The basic idea is to obtain a codebook of “words” by quantizing local image features via clustering. The histogram is in the form of the frequencies of the occurrences of the codewords. The histogram can be pooled over the entire image. One can also divide the

image into subregions and pool the histograms within these subregions [4]. Such histograms can then be used for image classification, usually by SVM with histogram intersection kernel [4, 6]. Again, because the spatial pooling of the histograms discards the exact position information, such histograms are invariant to shape deformations.

References

1. Csurka G, Dance C, Fan L, Willamowski J, Bray C (2004) Visual categorization with bags of keypoints. In: Workshop of ECCV, Prague
2. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE conference on computer vision pattern recognition (CVPR), San Diego
3. Heeger DJ, Bergen JR (1995) Pyramid-based texture analysis/synthesis. In: SIGGRAPH'95, Los Angeles, pp 229–238
4. Lazebnik S, Schmid C, Ponce J (2006) Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: IEEE conference on computer vision pattern recognition (CVPR), New York
5. Lowe DG (1999) Object recognition from local scale-invariant features. In: ICCV, Kerkyra
6. Maji S, Berg AC, Malik J (2008) Classification using intersection kernel support vector machines is efficient. In: IEEE conference on computer vision pattern recognition (CVPR), Anchorage
7. Zhu SC, Wu YN, Mumford DB (1997) Filter, random field, and maximum entropy (FRAME): towards a unified theory for texture modeling. Int J Comput Vis 27:107–126

Human Appearance Modeling and Tracking

► Appearance-Based Human Tracking

Human Motion Classification

► Gesture Recognition

Human Pose Estimation

Leonid Sigal
Disney Research, Pittsburgh, PA, USA

Synonyms

Articulated pose estimation; Body configuration recovery

Related Concepts

► Human Pose Estimation

Definition

Human pose estimation is the process of estimating the configuration of the body (pose) from a single, typically monocular, image.

Background

Human pose estimation is one of the key problems in computer vision that has been studied for well over 15 years. The reason for its importance is the abundance of applications that can benefit from such a technology. For example, human pose estimation allows for higher-level reasoning in the context of human-computer interaction and activity recognition; it is also one of the basic building blocks for marker-less motion capture (MoCap) technology. MoCap technology is useful for applications ranging from character animation to clinical analysis of gait pathologies.

Despite many years of research, however, pose estimation remains a very difficult and still largely unsolved problem. Among the most significant challenges are the following: (1) variability of human visual appearance in images, (2) variability in lighting conditions, (3) variability in human physique, (4) partial occlusions due to self-articulation and layering of objects in the scene, (5) complexity of human skeletal structure, (6) high dimensionality of the pose, and (7) the loss of 3D information that results from observing the pose from 2D planar image projections. To date, there is no approach that can produce satisfactory results in general, unconstrained settings while dealing with all of the aforementioned challenges.

Theory and Application

Human pose estimation is typically formulated probabilistically to account for ambiguities that may exist in the inference (though there are notable exceptions, e.g., [11]). In such cases, one is interested in estimating the posterior distribution, $p(\mathbf{x}|\mathbf{z})$, where \mathbf{x} is the pose of

the body and \mathbf{z} is a feature set derived from the image. The key modeling choices that affect the inference are:

- The representation of the pose – \mathbf{x}
- The nature and encoding of image features – \mathbf{z}
- The inference framework required to estimate the posterior – $p(\mathbf{x}|\mathbf{z})$

Next, the primary lines of research in pose estimation with respect to these modeling choices are reviewed. It is worth noting that these three modeling choices are not always independent. For example, some inference frameworks are specifically designed to utilize a given representation of the pose.

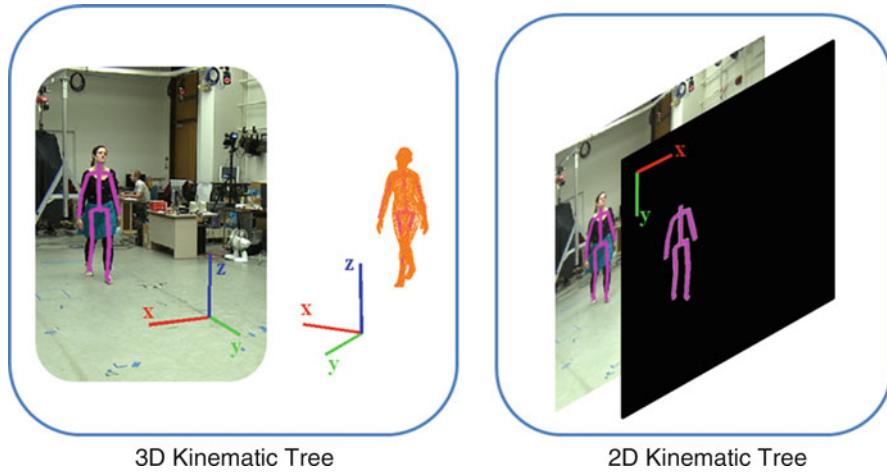
Representation: The configuration of the human body can be represented in a variety of ways. The most direct and common representation is obtained by parameterizing the body as a *kinematic tree* (see Fig. 1), $\mathbf{x} = \{\tau, \theta_\tau, \theta_1, \theta_2, \dots, \theta_N\}$, where the pose is encoded using position of the root segment (to keep the kinematic tree as short as possible, the pelvis is typically used as the root segment), τ ; orientation of the root segment in the world, θ_τ and a set of relative joint angles, $\{\theta_i\}_{i=1}^N$, that represent the orientations of body parts with respect to their parents along the tree (e.g., the orientation of the thigh with respect to the pelvis, shin with respect to the thigh).

Kinematic tree representation can be obtained for 2D, 2.5D, and 3D body models. In 3D, $\tau \in \mathbb{R}^3$ and $\theta_\tau \in SO(3)$; $\theta_i \in SO(3)$ for spherical joints (e.g., neck), $\theta_i \in \mathbb{R}^2$ for saddle joints (e.g., wrist), and $\theta_i \in \mathbb{R}^1$ for hinge joints (e.g., knee) and represents the pose of the body in the world. Note, the actual representation of the rotations in 3D is beyond the scope of this entry. In 2D, $\tau \in \mathbb{R}^2$ and $\theta_\tau \in \mathbb{R}^1$; $\theta_i \in \mathbb{R}^1$ corresponds to pose of the *cardboard* person in the image plane. 2.5D representations are the least common and are extensions of the 2D representation such that the pose, \mathbf{x} , is augmented with (typically discrete) variables encoding the relative depth (layering) of body parts with respect to one another in the 2D *cardboard* model. In all cases, be it in 2D or 3D, this representation results in a high-dimensional pose vector, \mathbf{x} , in $\mathbb{R}^{30} - \mathbb{R}^{70}$, depending on the fidelity and exact parameterization of the skeleton and joints. Alternatively, one can parameterize the pose of the body by 2D or 3D locations of the major joints [6]. For example, $\mathbf{x} = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_N\}$, where \mathbf{p}_i is the joint location in the world, $\mathbf{p}_i \in \mathbb{R}^3$, or, in an image, $\mathbf{p}_i \in \mathbb{R}^2$. This latter representation is less common, however, because it is not invariant to the morphology (body segment lengths) of a given individual.

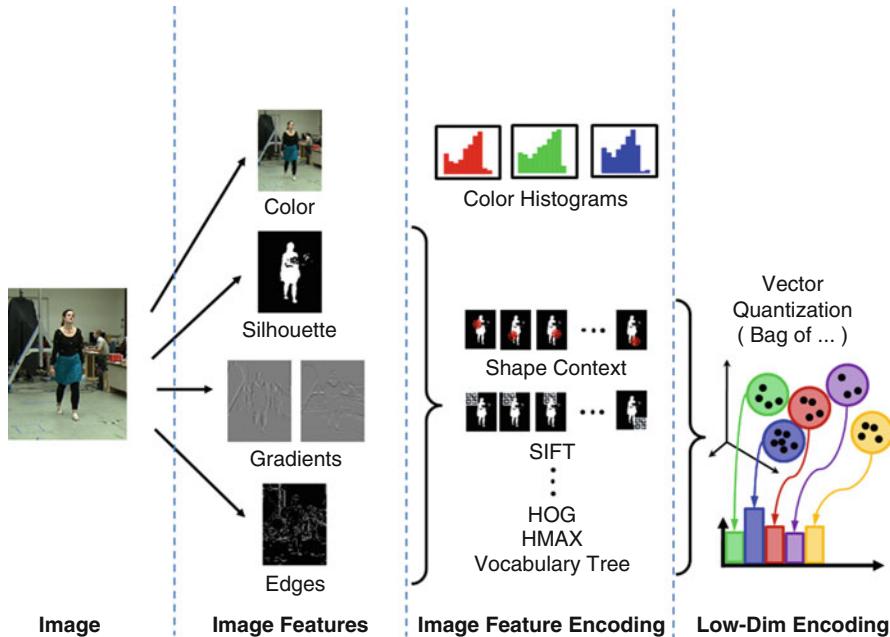
A typical alternative to the kinematic tree models is to model the body as a set of parts, $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_M\}$, each with its own position and orientation in space, $\mathbf{x}_i = \{\tau_i, \theta_i\}$, that are connected by a set of statistical or physical constraints that enforce skeletal (and sometimes image) consistency. Because part-based parameterization is redundant, it results in an even higher-dimensional representation. However, it does so in a way that makes it efficient to infer the pose, as will be discussed in a later section. Methods that utilize such a parameterization are often called *part-based*. As in kinematic tree models, the parts can be defined in 2D [2, 4, 7, 8, 11, 16] or in 3D [21], with 2D parameterizations being significantly more common. In 2D, each part's representation is often augmented with an additional variable, s_i , that accounts for uniform scaling of the body part in the image, i.e., $\mathbf{x}_i = \{\tau_i, \theta_i, s_i\}$ with $\tau_i \in \mathbb{R}^2$, $\theta_i \in \mathbb{R}^1$, and $s_i \in \mathbb{R}^1$.

Image features: Performance of any pose estimation approach depends substantially on the observations, or image features, that are chosen to represent salient parts of the image with respect to the human pose. A related and equally important issue is one of how these features are encoded. In addition to using different encodings, some approaches propose to reduce the dimensionality of the resulting feature vectors through vector quantization or *bag-of-words* representations. These coarser representations simplify feature matching but at the expense of losing spatial structure in the image. Common features and encoding methods are illustrated in Fig. 2.

Over the years, many features have been proposed by various authors. The most common features include image silhouettes [1], for effectively separating the person from background in static scenes; color [16], for modeling unoccluded skin or clothing; edges [16], for modeling external and internal contours of the body; and gradients [5], for modeling the texture over the body parts. Less common features include shading and focus [14]. To reduce dimensionality and increase robustness to noise, these raw features are often encapsulated in image descriptors, such as shape context [1, 2, 6], SIFT [6], and histogram of oriented gradients [5]. Alternatively, hierarchical multilevel image encodings can be used, such as HMAX [12], spatial pyramids [12], and vocabulary trees [12]. The effectiveness of different feature types on pose estimation has been studied in the context of several inference architectures; see [2] and [12] for discussions and quantitative analyses.



Human Pose Estimation, Fig. 1 *Skeleton representation:* Illustration of the 3D and 2D kinematic tree skeleton representation on the *left* and *right*, respectively



Human Pose Estimation, Fig. 2 *Image features:* Illustration of common image features and encoding methods used in the literature

Inference (regression models): Characterizing the posterior distribution, $p(\mathbf{x}|\mathbf{z})$, can be done in a number of ways. Perhaps the most intuitive way is to define a parametric [1, 6, 12] or nonparametric [15, 18, 22, 25] form for the conditional distribution $p(\mathbf{x}|\mathbf{z})$ and learn the parameters of that distribution from a set of training exemplars. This class of models is more widely known as *discriminative methods*, and they have been shown

to be very effective for pose estimation. Such methods directly learn $p(\mathbf{x}|\mathbf{z})$ from a labeled dataset of poses and corresponding images, $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{z}_i)\}_{i=1}^N$, which can be produced artificially using computer graphics software packages (e.g., Poser) [1, 12, 18]. The inference takes a form of probabilistic regression. Once a regression function is learned, a scanning window approach is typically used at test time to detect a

portion of the image (bounding box) where the person resides; $p(\mathbf{x}|\mathbf{z})$ is then used to characterize the configuration of the person in that target window.

The simplest method in this category is the one of linear regression [1], where the body configuration, \mathbf{x} , is assumed to be a linear combination of the image features, \mathbf{z} , with additive Gaussian noise,

$$\mathbf{x} = A[\mathbf{z} - \mu_z] + \mu_x + \nu; \quad \nu \sim \mathcal{N}(0, \Sigma);$$

$\mu_x = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i$, and $\mu_z = \frac{1}{N} \sum_{i=1}^N \mathbf{z}_i$ are means computed over the training samples to center the data. Alternatively, this can be written as

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(A[\mathbf{z} - \mu_z] + \mu_x, \Sigma). \quad (1)$$

The regression coefficients, A , can be learned easily from paired training samples, $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{z}_i)\}_{i=1}^N$, using the least squares formulation (see [1] for details).

Parametric vs. nonparametric: Parametric discriminative methods [1, 6, 12] are appealing because the model representation is fixed with respect to the size of the training dataset \mathcal{D} . However, simple parametric models, such as linear regression [1] or relevance vector machine [1], are unable to deal with complex non-linear relationships between image features and poses. Nonparametric methods, such as nearest neighbor regression [18] or kernel regression [18], are able to model arbitrary complex relationships between input features and output poses. The disadvantage of these nonparametric methods is that the model and inference complexity are both functions of the training set size. For example, in kernel regression,

$$p(\mathbf{x}|\mathbf{z}) = \sum_{i=1}^N \mathcal{K}_x(\mathbf{x}, \mathbf{x}_i) \frac{\mathcal{K}_z(\mathbf{z}, \mathbf{z}_i)}{\sum_{k=1}^N \mathcal{K}_z(\mathbf{z}, \mathbf{z}_k)}, \quad (2)$$

where $\mathcal{K}_x(\cdot, \cdot)$ and $\mathcal{K}_z(\cdot, \cdot)$ are kernel functions measuring the similarity of the arguments (e.g., Gaussian kernels) and the inference complexity is $O(N)$ (where N is the size of the training dataset). More sophisticated nonparametric methods, such as Gaussian Process Latent Variable Models (GPLVMs), can have even higher complexity; GPLVMs have $O(N^3)$ learning and $O(N^2)$ inference complexity. In practice, nonparametric methods tend to perform better but are slower.

Dealing with ambiguities: If one assumes that $p(\mathbf{x}|\mathbf{z})$ is unimodal [1], conditional expectation can be used to characterize the plausible configuration of the person in an image given the learned model. For example, for linear regression in Eq. (1),

$E[\mathbf{x}|\mathbf{z}] = A[\mathbf{z} - \mu_z] + \mu_x$; for kernel regression in Eq. (2),

$$E[\mathbf{x}|\mathbf{z}] = \sum_{i=1}^N \mathbf{x}_i \frac{\mathcal{K}_z(\mathbf{z}, \mathbf{z}_i)}{\sum_{k=1}^N \mathcal{K}_z(\mathbf{z}, \mathbf{z}_k)}. \quad (3)$$

In practice, however, most features under standard imaging conditions are ambiguous, resulting in multimodal distributions. Ambiguities naturally arise in image projections, where multiple poses can result in similar, if not identical, image features (e.g., front- and back-facing poses yield nearly identical silhouette features). To account for these ambiguities, parametric mixture models were introduced in the form of Mixture of Experts [6, 12]. Nonparametric alternatives, such as Local Gaussian Process Latent Variable Models (LGPLVM) [25], cluster the data into convex local sets and make unimodal predictions within each cluster, or search for prominent modes in $p(\mathbf{x}|\mathbf{z})$ [15, 22].

Learning: Obtaining the large datasets that are required for learning discriminative models that can generalize across motions and imaging conditions is challenging. Synthetic datasets often do not exhibit the imaging characteristics present in real images, and real fully labeled datasets are scarce. Furthermore, even if large datasets could be obtained, learning from vast amounts of data is not a trivial task [6]. To address this issue, two solutions were introduced: (1) learning from small datasets by discovering an intermediate low-dimensional latent space for regularization [15, 22] and (2) learning in semisupervised settings, where a relatively small dataset of paired samples is accompanied by a large amount of unlabeled data [12, 15, 22].

Limitations: Despite popularity and lots of successes, discriminative methods do have limitations. First, they are only capable of recovering a relative 3D configuration of the body and not its position in 3D space. The reason for this is practical, as reasoning about position in 3D space would require prohibitively large training datasets that span the entire 3D volume of the space visible from the camera. Second, their performance tends to degrade as the distributions of test and training data start to diverge; in other words, generalization

remains one of the key issues. Lastly, learning discriminative models efficiently from large datasets that cover wide range of realistic activities and postures remains a challenging task.

Inference (generative): Alternatively, one can take a generative approach and express the desired posterior, $p(\mathbf{x}|\mathbf{z})$, as a product of a likelihood and a prior:

$$p(\mathbf{x}|\mathbf{z}) \propto \underbrace{p(\mathbf{z}|\mathbf{x})}_{\text{likelihood}} \underbrace{p(\mathbf{x})}_{\text{prior}}. \quad (4)$$

Characterizing this high-dimensional posterior distribution is typically hard; hence, most approaches rely on a posteriori (MAP) solutions that look for the most probable configurations that are both typical (have high *prior* probability) and can explain the image data well (have high *likelihood*):

$$\mathbf{x}_{MAP} = \arg \max p(\mathbf{x}|\mathbf{z}). \quad (5)$$

Searching for such configurations, however, in the high-dimensional (40+) articulation space is very challenging and most approaches frequently get stuck in local optima. Global hierarchical search methods, such as Annealed Particle Filter [10], have shown some promising results for simple skeletal configurations, where body is mostly upright and when observations from multiple cameras are available. For the more general articulations and monocular observations that are often the focus of pose estimation algorithms, this class of methods has not been very successful to date.

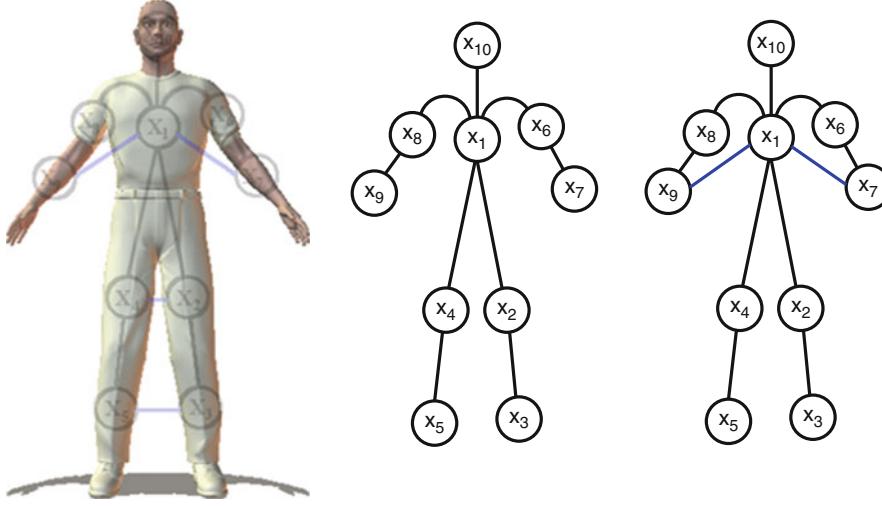
Inference (part-based models): To battle the inference complexity of generative models, part-based models have been introduced. These methods originate in the object recognition community with formulation of Fischler and Elschlager (1973) and assume that a body can be represented as an assembly of parts that are connected by constraints imposed by the joints within the skeletal structure (and, sometimes, by the image constraints imposed by projections onto an image plane that account for occlusions). This formulation reduces the inference complexity because likely body part locations can be searched for independently, only considering the nearby body parts that constrain them, which significantly prunes the total search space.

Among the earliest successes along this line of research is the work of Lee and Cohen [13]. Their

approach focused on obtaining proposal maps for the locations of individual joints within an image. These proposal maps were obtained based on a number of features that were computed densely over the image. For example, face detection was used to obtain hypotheses for the location of the head; head-shoulder contour matching, obtained using a deformable contour model and gradient descent, was used as evidence for shoulder joint locations; elliptical skin regions, obtained using skin-color segmentation, were used to determine the locations of the lower arms and lower legs. In addition, second-derivative (ridge) observations were used as evidence for other limbs of the body. Given proposals for the different joints, weighted by the confidence of corresponding detectors, a data-driven Markov Chain Monte Carlo (MCMC) approach was used to recover 3D configurations of the skeleton. This inference relied on direct inverse kinematics (IK) obtained from 2D proposal maps. To further improve the results, a kinematic jump proposal process was also introduced. The kinematic jump proposal process involves flipping a body part or a set of parts (i.e., the head, a hand, or an entire arm) in the depth direction around its pivotal joint.

Other part-based approaches try to assemble regions of an image into body parts and successively construct those parts into a body. Prime examples of such methods are introduced by Mori et al. [14] and Ren et al. [17]. In [14] superpixels were first assembled into body parts based on the evaluation of low-level image cues, including contour, shape, shading, and focus. The part proposals were then pruned and assembled together using length, body part adjacency, and clothing symmetry. A similar approach was taken in [17], but line segments were used instead of assembling superpixels. Parallel lines were assembled into candidate parts using a set of predefined rules, and the candidate parts were in turn assembled into the body with a set of joint, scale, appearance, and orientation consistency constraints. Unlike [14], the search for the most probable body configurations was formulated as a solution to an Integer Quadratic Programming (IQP) problem.

The most traditional and successful approach, however, is to represent the body using a Markov Random Field (MRF) with body parts corresponding to the nodes and constraints between parts encoded by potential functions that account for physical and statistical dependencies (see Fig. 3). Formally, the posterior,



Human Pose Estimation, Fig. 3 *Pictorial structures model:* Illustrated is the depiction of the 10-part tree-structured pictorial structures model (middle) and a non-tree-structured (loopy)

$p(\mathbf{x}|\mathbf{z})$, can be expressed as

$$\begin{aligned} p(\mathbf{x}|\mathbf{z}) &\propto p(\mathbf{z}|\mathbf{x})p(\mathbf{x}) \\ &= p(\mathbf{z}|\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_M\})p(\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_M\}) \\ &\approx \underbrace{\prod_{i=1}^M p(\mathbf{z}|\mathbf{x}_i)}_{likelihood} \underbrace{p(\mathbf{x}_1) \prod_{(i,j) \in E} p(\mathbf{x}_i, \mathbf{x}_j)}_{prior}. \end{aligned} \quad (6)$$

In this case, pose estimation takes the form of inference in a general MRF network. The inference can be solved efficiently using message-passing algorithms, such as Belief Propagation (BP). BP consists of two distinct phases: (1) a set of message-passing iterations are executed to propagate consistent part estimates within a graph, and (2) marginal posterior distributions are estimated for every body part [2, 8, 16]. A typical formulation looks at the configuration of the body in the 2D image plane and assumes discretization of the pose for each individual part, e.g., $\mathbf{x}_i = \{\tau_i, \theta_i, s_i\}$, where $\tau_i \in \mathbb{R}^2$ is the location and $\theta_i \in \mathbb{R}^1$ and $s_i \in \mathbb{R}^1$ are orientation and scale of the part i (represented as a rectangular patch) in the image plane. As a result, the inference is over a set of discrete part configurations $\mathbf{l}_i \in \mathbb{Z}$ (for part i), where \mathbb{Z} is the enumeration of poses for a part in an image (\mathbf{l}_i is a discrete version of \mathbf{x}_i). With an additional assumption of pairwise potentials that account for kinematic constraints,

pictorial structures model (right). In the non-tree-structured model, additional constraints encoding occlusions are illustrated in blue

the model forms a tree-structured graph known as the *tree-structured pictorial structures* (PS) model. An approximate inference with continuous variables is also possible [20, 21].

Inference in the tree-structured PS model first proceeds by sending recursively defined messages of the form:

$$m_{i \rightarrow j}(\mathbf{l}_j) = \sum_{\mathbf{l}_i} p(\mathbf{l}_i, \mathbf{l}_j) p(\mathbf{z}|\mathbf{l}_i) \prod_{k \in A(i) \setminus j} m_{k \rightarrow i}(\mathbf{l}_i), \quad (7)$$

where $m_{i \rightarrow j}$ is the message from part i to part j , with $p(\mathbf{l}_i, \mathbf{l}_j)$ measuring the compatibility of poses for the two parts and $p(\mathbf{z}|\mathbf{l}_i)$ the likelihood, and $A(i) \setminus j$ is the set of parts in the graph adjacent to i except for j . Compatibility, $p(\mathbf{l}_i, \mathbf{l}_j)$, is often measured by the physical consistency of two parts at the joint, or by their statistical (e.g., angular) co-occurrence with respect to one another. In a tree-structured PS graph, these messages are sent from the outermost extremities inward and then back outward.

Once all of the message updates are complete, the marginal posteriors for all of the parts can be estimated as

$$p(\mathbf{l}_i|\mathbf{z}) \propto p(\mathbf{z}|\mathbf{l}_i) \prod_{j \in A(i)} m_{j \rightarrow i}(\mathbf{l}_i). \quad (8)$$

Similarly, the most likely configuration can be obtained as a MAP estimate:

$$\mathbf{l}_{i,MAP} = \arg \max_{\mathbf{l}_i} p(\mathbf{l}_i | \mathbf{z}). \quad (9)$$

One of the key benefits of the pictorial structures (PS) paradigm is its simplicity and efficiency. In PS exact inference is possible in the time linear to the number of discrete configurations a given part can assume. Because of this property, recent implementations [2] can handle the pixel-dense configurations of parts that result in millions of potential discrete states for each body part. The linear complexity comes from the observation that a generally complex non-Gaussian prior over neighboring parts, $p(\mathbf{x}_i, \mathbf{x}_j)$, can be expressed as a Gaussian prior over the transformed locations corresponding to joints, mainly $p(\mathbf{x}_i, \mathbf{x}_j) = \mathcal{N}(T_{ij}(\mathbf{x}_i); T_{ji}(\mathbf{x}_j), \Sigma_{ij})$. This is done by defining a transformation, $T_{ij}(\mathbf{x}_i)$, that maps a common joint between parts i and j , defined in the part i 's coordinate frame, to its location in the image space. Similarly, $T_{ji}(\mathbf{x}_j)$ defines the transformation from the same common joint defined in the j 's coordinate frame to the location in the image plane. This transformation allows the inference to use an efficient solution that involves convolution (see [8] for more details).

Performance: Recently, it has been shown that the effectiveness of a PS model is closely tied to the quality of the part likelihoods [2]. Discriminatively trained models [16] and more complex appearance models [2] tend to outperform models defined by hand [8]. Methods that learn likelihood cascades, corresponding to better and better features tuned to a particular image, have also been explored for both superior speed and performance [16]. Most recent discriminative formulation of PS model allows joint learning of part appearances and model structure [26] using structural Support Vector Machine (SVM).

Speed: Cascades of part detectors serve not only to improve performance but also to speed up the inference (e.g., [23]). Fast likelihoods can be used to prune away large parts of the search space before applying more complex and computationally expensive likelihood models. Other approaches to speed up performance include data-driven methods (e.g., data-driven Belief Propagation). These methods look for the parts

in an image first and then assemble a small set of the part candidates into the body (akin to the methods of [14, 17]). The problem with such approaches is that any occluded parts are missed altogether because they cannot be detected by the initial part detectors. Inference can also be sped up by using progressive search refinement methods [9]. For example, some methods use upper body detectors to restrict the search to promising parts of the image instead of searching the whole image.

Non-tree-structured extensions: Although tree-structured PS models are computationally efficient and exact, they generally are not sufficient to model all the necessary constraints imposed by the body. More complex relationships among the parts that fall outside of the realm of these models include nonpenetration constraints and occlusion constraints [20]. Incorporating such relationships into the model adds loops corresponding to long-range dependencies between body parts. These loops complicate inference because (1) no optimal solutions can be found efficiently (message-passing algorithms, like BP, are not guaranteed to converge in loopy graphs) and (2) even approximate inference is typically computationally expensive. Despite these challenges, it has been argued that adding such constraints is necessary to improve performance [4]. To alleviate some of the inference complexities with these non-tree-structured models, a number of competing methods have been introduced. Early attempts used sampling techniques from the tree-structured posterior as proposals for evaluation of a more complex non-tree-structured model [7, 8]. To obtain optimality guarantees, branch-and-bound search was recently proposed by Tian et al. [24], with the tree-structured solutions as a lower bound on the more complex loopy model energy.

Open Problems

Despite much progress in the field, pose estimation remains a challenging and still largely unsolved task. Progress has been made in estimating the configurations of mostly unoccluded and isolated subjects. Open problems include dealing with multiple, potentially interacting people (e.g., [7]) and tolerance to unexpected occlusions. Future research is

also likely to expand on the types of postures and imaging conditions that the current algorithms can handle.

To date, most successful pose estimation approaches have been bottom-up. This observation applies to both discriminative approaches and part-based approaches. However, it seems shortsighted to assume that the general pose estimation problem can be solved purely in a bottom-up fashion. Top-down information may be useful for enforcing global pose consistency, and a combination of top-down and bottom-up inference is likely to lead to success faster. The recent success of combining bottom-up part-based models with 3D top-down priors [3] is encouraging and should be built upon to produce models that can deal with more complex postures and motions. Earlier attempts at building hierarchical models [27] may also be worth revisiting with the newfound insights.

Finally, there is significant evidence suggesting that successfully estimating pose independently at every frame is a very ill-posed problem. Spatio-temporal models that aggregate information over time [3] are emerging as a way to regularize performance obtained in individual frames and smooth out the noise in the estimates. Leveraging all sources of generic prior knowledge, such as spatial layout of the body and temporal consistency of poses, and rich image observation models is critical in advancing the state of the art.

References

1. Agarwal A, Triggs B (2006) Recovering 3d human pose from monocular images. *IEEE Trans Pattern Anal Mach Intell* 28(1):44–58
2. Andriluka M, Roth S, Schiele B (2009) Pictorial structures revisited: people detection and articulated pose estimation. In: IEEE conference on computer vision and pattern recognition (CVPR), Miami
3. Andriluka M, Roth S, Schiele B (2010) Monocular 3d pose estimation and tracking by detection. In: IEEE conference on computer vision and pattern recognition (CVPR), San Francisco
4. Bergtholdt M, Kappes J, Schmidt S, Schnorr C (2010) A study of parts-based object class detection using complete graphs. *Int J Comput Vis* 87(1–2):93–117
5. Bo L, Sminchisescu C (2010) Twin gaussian processes for structured prediction. *Int J Comput Vis* 87(1–2):28–52
6. Bo L, Sminchisescu C, Kanaujia A, Metaxas D (2008) Fast algorithms for large scale conditional 3d prediction. In: IEEE conference on computer vision and pattern recognition (CVPR), Anchorage
7. Eichner M, Ferrari V (2010) We are family: joint pose estimation of multiple persons. In: European conference on computer vision (ECCV), Heraklion
8. Felzenszwalb PF, Huttenlocher DP (2005) Pictorial structures for object recognition. *Int J Comput Vis* 61(1):55–79
9. Ferrari V, Marn-Jimnez MJ, Zisserman A (2008) Progressive search space reduction for human pose estimation. In: IEEE conference on computer vision and pattern recognition (CVPR), Anchorage
10. Gall J, Rosenhahn B, Brox T, Seidel H-P (2010) Optimization and filtering for human motion capture. *Int J Comput Vis* 87(1–2):75–92
11. Jiang H (2009) Human pose estimation using consistent max-covering. In: IEEE international conference on computer vision, Kyoto
12. Kanaujia A, Sminchisescu C, Metaxas D (2007) Semi-supervised hierarchical models for 3d human pose reconstruction. In: IEEE conference on computer vision and pattern recognition (CVPR), Minneapolis
13. Lee MW, Cohen I (2004) Proposal maps driven mcmc for estimating human body pose in static images. In: IEEE conference on computer vision and pattern recognition (CVPR), Washington, DC
14. Mori G, Ren X, Efros A, Malik J (2004) Recovering human body configurations: Combining segmentation and recognition. In: IEEE conference on computer vision and pattern recognition (CVPR), Washington, DC
15. Navaratnam R, Fitzgibbon A, Cipolla R (2007) The joint manifold model for semi-supervised multi-valued regression. In: IEEE international conference on computer vision, Rio de Janeiro
16. Ramanan D (2006) Learning to parse images of articulated bodies. In: Neural information and processing systems, Vancouver
17. Ren X, Berg AC, Malik J (2005) Recovering human body configurations using pair-wise constraints between parts. In: International conference on computer vision, Beijing
18. Shakhnarovich G, Viola P, Darrell T (2003) Fast pose estimation with parameter sensitive hashing. In: International conference on computer vision, Nice
19. Sigal L, Balan A, Black MJ (2007) Combined discriminative and generative articulated pose and non-rigid shape estimation. In: Neural information and processing systems, Vancouver
20. Sigal L, Black MJ (2006) Measure locally, reason globally: occlusion-sensitive articulated pose estimation. In: IEEE conference on computer vision and pattern recognition (CVPR), New York
21. Sigal L, Isard M, Sigelman BH, Black MJ (2003) Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In: Advances in neural information processing systems, Vancouver
22. Sigal L, Memisevic R, Fleet DJ (2009) Shared kernel information embedding for discriminative inference. In: IEEE conference on computer vision and pattern recognition, Miami

23. Singh VK, Nevatia R, Huang C (2010) Efficient inference with multiple heterogeneous part detectors for human pose estimation. In: European conference on computer vision (ECCV), Heraklion, pp 314–327
24. Tian T-P, Sclaroff S (2010) Fast globally optimal 2d human detection with loopy graph models. In: IEEE conference on computer vision and pattern recognition (CVPR), San Francisco
25. Urtasun R, Darrell T (2008) Sparse probabilistic regression for activity-independent human pose inference. In: IEEE conference on computer vision and pattern recognition (CVPR), Anchorage
26. Yang Y, Ramanan D (2011) Articulated pose estimation with flexible mixture-of-parts. In: IEEE conference on computer vision and pattern recognition (CVPR), Colorado Springs
27. Zhang J, Luo J, Collins R, Liu Y (2006) Body localization in still images using hierarchical models and hybrid search. In: IEEE conference on computer vision and pattern recognition (CVPR), New York

ICP

► [Iterative Closest Point \(ICP\)](#)

Illumination Estimation, Illuminant Estimation

Stephen Lin
Microsoft Research Asia, Beijing Sigma Center,
Beijing, China

Related Concepts

► [Color Constancy](#); ► [Incident Light Measurement](#)

Definition

The purpose of illumination estimation is to determine the direction, intensity, and/or color of the lighting in a scene. In contrast to direct measurement of lighting, the illumination information is inferred from cues within the scene, without use of a special probe or color calibration chart.

Background

The appearance of objects and scenes can vary considerably with respect to illumination conditions. In [1], differences in face appearance due to lighting were found to be greater than those due to identity. Since such appearance variations can affect the performance of certain computer vision algorithms,

much research has focused on illumination estimation, so that lighting can be accounted for in image understanding.

To simplify inference, methods for illumination estimation typically assume that the illumination originates from distant light sources. With this assumption, the illumination can be considered to be uniform across the scene, such that only a single lighting condition needs to be estimated. Most techniques perform this estimation on a single input image, as this allows for wider applicability.

Methods

Different image cues have been utilized to estimate illumination. Several techniques categorized by cue are described in the following.

Shading

Many methods for illumination estimation are based on an analysis of shading over the surface of an object. They typically utilize the relationship between shading and lighting described by the Lambertian reflectance model:

$$I(x) = \rho(x)N(x) \cdot L$$

where x indexes the shaded image pixels, I denotes image intensity (shading), ρ is the albedo, $N(x)$ is the surface normal, and L is the light vector that encodes the direction and magnitude of illumination. To solve for L , shading-based techniques for illumination estimation generally assume the surface of interest to have a uniform albedo and a known geometry. If the absolute albedo value is unknown, then L can be estimated up to an unknown scale factor.

While some methods focus on recovering only the direction of a single illuminant [2, 3], most address the more common scenario of multiple illumination sources. Hougen and Ahuja [4] solve a set of linear equations to determine light intensities from a set of sampled directions. Yang and Yuille [5] use image intensities and known surface normals at occluding boundaries to constrain illuminant directions. Ramamoorthi and Hanrahan [6] compute a low-frequency illumination distribution from a deconvolution of reflectance and lighting. Zhang and Yang [7] estimate lighting directions from critical points which have surface normals perpendicular to an illuminant direction. Based on this, Wang and Samaras [8] segmented images into regions of uniform lighting and then performed estimation by recursive least-squares fitting of the Lambertian reflectance model to these regions.

Illumination may alternatively be estimated by uncalibrated photometric stereo [9], without the need for known albedos and surface normals. This approach requires a set of images taken under different lighting conditions as input.

Cast Shadows

Several techniques analyze cast shadows for illumination estimation. For an object of known shape, the shadows that it casts provide constraints on the lighting directions and their corresponding intensities. Sato et al. [10–13] formulated these constraints as a system of equations in terms of observed brightness values within shadows and a set of sampled lighting directions at which source intensities are to be solved. These methods require a single input image for objects that cast shadows onto a uniform-colored surface; two images are needed to cancel out the effects of color variation for surfaces with texture. This approach was extended by Okabe et al. [14] to a lighting representation of Haar wavelets. Kim and Hong [15] later proposed a single-image method that handles surface texture by incorporating regularization and some user-specified information.

Specular Reflections

Some methods consider specular reflections in estimating illumination. From the locations of specular reflections on an object of known shape, these techniques compute the corresponding light source

directions according to the mirror reflection property. This approach was used by Nishino et al. [16] to obtain an initial approximation of the illumination distribution, which is then refined using a more sophisticated model of reflectance. Illumination cues from specular reflections are combined with those from shading and shadows by Li et al. [17] to minimize the effects of scene texture on lighting estimation. Without needing explicit object shape recovery, Nishino et al. [18] and Wang et al. [19] proposed to estimate lighting from specular reflections on human eyes, which are highly reflective and have a similar shape from person to person.

Color

Much research focuses on estimating the color of illumination, rather than the directional distribution. Early methods solve for light color based on assumed properties of the imaged scene, such as the average color being achromatic [20] or that the scene contains a maximally reflective white patch [21]. More recent techniques are guided by more detailed knowledge about illumination and surface colors in the natural world. A statistical model of lights and surfaces from training data is used by Brainard and Freeman [22] and Finlayson et al. [23] to obtain a solution. Knowledge about illuminations and surfaces is instead used by Forsyth [24] and Finlayson et al. [25] to constrain the range of illuminant colors that could possibly result in the observed image. A comprehensive review of illumination color estimation methods is provided in [26].

Application

Illumination estimation has been employed in various applications based on appearance modeling. In [18], lighting estimates from eye reflections are used for robust face recognition under varying illumination conditions. Estimates of scene illumination have also been used to realistically composite virtual objects into an image in an illumination-consistent manner [27, 28]. In digital cameras, methods for estimating illumination color are incorporated into automatic white balance algorithms. Recent methods for estimating light color, however, have been found to be inadequate in improving color-based object recognition [29].

References

1. Moses Y, Adini Y, Ullman S (1994) Face recognition: the problem of compensating for changes in illumination direction. In: Proceedings of European conference on computer vision (ECCV). Springer, Heidelberg/Berlin, pp 286–296
2. Zheng Q, Chellappa R (1991) Estimation of illuminant direction, albedo, and shape from shading. *IEEE Trans Pattern Anal Mach Intell* 13:680–702
3. Samaras D, Metaxas D (1999) Coupled lighting direction and shape estimation from single images. In: Proceedings of the international conference on computer vision. IEEE Computer Society, Washington, DC, pp 868–874
4. Hougen DR, Ahuja N (1993) Estimation of the light source distribution and its use in integrated shape recovery from stereo and shading. In: Proceedings of the international conference on computer vision. IEEE Computer Society, Washington, DC, pp 148–155
5. Yang Y, Yuille AL (1991) Sources from shading. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Computer Society, Washington, DC, pp 534–539
6. Ramamoorthi R, Hanrahan P (2001) A signal-processing framework for inverse rendering. In: Proceedings of ACM SIGGRAPH. ACM, New York, pp 117–128
7. Zhang Y, Yang YH (2001) Multiple illuminant direction detection with application to image synthesis. *IEEE Trans Pattern Anal Mach Intell* 23:915–920
8. Wang Y, Samaras D (2002) Estimation of multiple illuminants from a single image of arbitrary known geometry. In: Proceedings of European conference on computer vision (ECCV). Lecture notes in computer science, vol 2352. Springer, Berlin/Heidelberg, pp 272–288
9. Basri R, Jacobs D, Kemelmacher I (2007) Photometric stereo with general, unknown lighting. *Int J Comput Vis* 72:239–257
10. Sato I, Sato Y, Ikeuchi K (1999) Illumination distribution from brightness in shadows: adaptive estimation of illumination distribution with unknown reflectance properties in shadow regions. In: Proceedings of the international conference on computer vision. IEEE Computer Society, Washington, DC, pp 875–883
11. Sato I, Sato Y, Ikeuchi K (1999) Illumination distribution from shadows. In: Proceeding of the IEEE Conference on computer vision and pattern recognition (CVPR). IEEE Computer Society, Washington, DC, pp 306–312
12. Sato I, Sato Y, Ikeuchi K (2001) Stability issues in recovering illumination distribution from brightness in shadows. *Proc IEEE Conf Comput Vis Pattern Recognit (CVPR)* II:400–407
13. Sato I, Sato Y, Ikeuchi K (2003) Illumination from shadows. *IEEE Trans Pattern Anal Mach Intell* 25:290–300
14. Okabe T, Sato I, Sato Y (2004) Spherical harmonics vs. haar wavelets: basis for recovering illumination from cast shadows. *Proc IEEE Conf Comput Vis Pattern Recognit (CVPR)* I:50–57
15. Kim T, Hong K (2005) A practical single image based approach for estimating illumination distribution from shadows. In: Proceedings of the international conference on computer vision. IEEE Computer Society, Washington, DC, pp 266–271
16. Nishino K, Zhang Z, Ikeuchi K (2001) Determining reflectance parameters and illumination distribution from sparse set of images for viewdependent image synthesis. In: Proceedings of the international conference on computer vision. IEEE Computer Society, Washington, DC, pp 599–606
17. Li Y, Lin S, Lu H, Shum HY (2003) Multiple-cue illumination estimation in textured scenes. In: Proceedings of the international conference on computer vision. IEEE Computer Society, Washington, DC, pp 1366–1373
18. Nishino K, Belhumeur P, Nayar S (2005) Using eye reflections for face recognition under varying illumination. *Proc Int Conf Comput Vis* I:519–526
19. Wang H, Lin S, Liu X, Kang SB (2005) Separating reflections in human iris images for illumination estimation. In: Proceedings of the international conference on computer vision. IEEE Computer Society, Washington, DC, pp 1691–1698
20. Buchsbaum G (1980) A spatial processor model for object colour perception. *J Franklin Inst* 310:1–26
21. Land EH (1977) The retinex theory of color vision. *Sci Am* 237:108–128
22. Brainard DH, Freeman WT (1997) Bayesian color constancy. *J Opt Soc Am A* 14:1393–1411
23. Finlayson GD, Hordley S, Hubel PM (2001) Color by correlation: a simple, unifying framework for color constancy. *IEEE Trans Pattern Anal Mach Intell* 23:1209–1221
24. Forsyth DA (1990) A novel algorithm for colour constancy. *Int J Comput Vis* 5:3–36
25. Finlayson GD, Hordley S, Tastl I (2006) Gamut constrained illuminant estimation. *Int J Comput Vis* 67:93–109
26. Ebner M (2007) Color constancy. Wiley, Chichester
27. Sato I, Sato Y, Ikeuchi K (1999) Acquiring a radiance distribution to superimpose virtual objects onto a real scene. *IEEE Trans Vis Comput Graph* 5:1–12
28. Lalonde JF, Efros AA, Narasimhan SG (2009) Estimating natural illumination from a single outdoor image. In: Proceedings of the international conference on computer Vision. IEEE Computer Society, Washington, DC
29. Funt B, Barnard K, Martin L (1998) Is colour constancy good enough? In: Proceedings of the European conference on computer vision (ECCV). Springer, London, pp 445–459

Image Alignment

► Image Registration

Image Decompositions

Marshall F. Tappen

University of Central Florida, Orlando, FL, USA

Definition

An image decomposition is the result of a mathematical transformation of an image into a new set of images

that represent different aspects of the input image or scene pictured in that image. The original image can typically be reconstructed from these new images.

Background

While images are primarily stored as an array of pixel values, an image can be represented in a number of different ways. For instance, an image can be easily transformed into two images, one containing the high-frequency variation in the input image and a second containing the low-frequency variation. This process decomposes the input image into two images, each of which expresses different information about the original image.

This process is useful when further processing will treat these two images differently. If the decomposition is chosen correctly, the image is decomposed into a set of images that can each be processed uniformly. Thus, the decomposition facilitates adaptive processing of the content of an image.

Theory

Image decompositions can be roughly divided into two different types of decompositions, image-based decompositions and intrinsic image decompositions. Image-based decompositions represent the image itself using new images, while the intrinsic image decompositions reflect the content of the scene pictured in the image itself.

Image-Based Decompositions

Similar to the background example above, many image-based decompositions focus on representing multi-scale frequency content in the scene. The Gaussian pyramid is one of the most basic decompositions representing multi-scale content. The decomposition consists of a set of images of progressively smaller resolution, with each image being one level of the pyramid. Each level is created by filtering the image at the level below, then downsampling the result. This creates a multi-resolution set of images.

Depending on the application, the usefulness of the Gaussian pyramid may be limited because each level contains redundant information. This can be eliminated by modifying the pyramid creation process

to create a Laplacian pyramid [1]. In the Laplacian pyramid, the input image is progressively downsampled. The image at level i in the Laplacian pyramid is computed by taking the difference between the i th level of the Gaussian pyramid and the upsampled version of level $i + 1$, which has been downsampled from the i th level of the Gaussian pyramid. Effectively, each level of the Laplacian pyramid expresses the image information at a particular scale. [Figure 1](#) shows an example of the Laplacian decomposition of an image.

In [2], Simoncelli et al. extended this decomposition process to also separate orientation into different images, creating the steerable pyramid decomposition. Similar decompositions can also be generated by using a different process to separate the images. In [3], the bilateral filter is used to generate a two-image decomposition.

These decompositions are also connected to other image transformations, particularly wavelets. The connections are discussed in [2].

Intrinsic Image Decompositions

While image-based decompositions are focused on the pixel values themselves, intrinsic image decompositions create images that are based on the content of the scene. The intrinsic image decomposition is based on the intrinsic image approach for representing scene characteristics. In this approach, each intrinsic characteristic of the scene is represented by a distinct image. In the intrinsic image decomposition, these images are chosen to both represent intrinsic characteristics and image content.

In [4], Weiss uses video data to separate an image into illumination, or shading, and albedo components. In this decomposition, an input image pixel at location n , $I(n)$, is equal to the product of a shading image and an albedo image, or $I(n) = S(n) \times A(n)$. In [5] and [6], Tappen et al. show how the intrinsic image decomposition can be computed from a single image. [Figure 2](#) shows an example of an intrinsic image decomposition for the image on the left.

Application

Image decompositions are frequently used to generate images that processed separately. In [7], Portilla et al. use the steerable pyramid to denoise images. Heeger and Bergen showed that texture can be generated by

Image Decompositions, Fig. 1

These images are a Laplacian pyramid created from the well-known *Lena* image. Each image captures the variation at a specific scale



Input Image

Shading Image

Albedo Image

Image Decompositions, Fig. 2 An example of an intrinsic image decomposition. The image on the *left* is decomposed into shading and albedo components

forcing the marginal histograms of the levels of a steerable pyramid to match those of a pyramid generated from a reference image [8]. More complete measures of statistical similarity are used in [9], leading to improved synthesis results.

As mentioned earlier, the bilateral filter is used in [3] to separate the image into large- and fine-scale variations to combine images taken under different illumination. In [10], Bousseau et al. describe how user input can improve intrinsic image decompositions and demonstrate how they can be applied for graphics applications.

References

1. Burt P, Adelson E (1983) The Laplacian pyramid as a compact image code. *IEEE Trans Commun* 31(4):532–540
2. Simoncelli EP, Freeman WT, Adelson EH, Heeger DJ (1992) Shiftable multiscale transforms. *IEEE Trans Inf Theory* 38(2):587–607
3. Eisemann E, Durand F (2004) Flash photography enhancement via intrinsic relighting. *ACM Trans Graph* 23:673–678
4. Weiss Y (2001) Deriving intrinsic images from image sequences. In: IEEE international conference on computer vision, Vancouver, vol 2, p 68
5. Tappen MF, Freeman WT, Adelson EH (2005) Recovering intrinsic images from a single image. *IEEE Trans Pattern Anal Mach Intell* 27(9):1459–1472

6. Tappen MF, Adelson EH, Freeman WT (2006) Estimating intrinsic component images using non-linear regression. In: The proceedings of the 2006 IEEE computer society conference on computer vision and pattern recognition (CVPR), New York, vol 2, pp 1992–1999
7. Portilla J, Strela V, Wainwright M, Simoncelli E (2003) Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE Trans Image Process* 12(11): 1338–1351
8. Heeger DJ, Bergen JR (1995) Pyramid-based texture analysis/synthesis. In: Proceedings of ACM SIGGRAPH 1995. ACM, New York, pp 229–238
9. Portilla J, Simoncelli EP (2000) A parametric texture model based on joint statistics of complex wavelet coefficients. *Int J Comput Vis* 40(1):49–70
10. Bousseau A, Paris S, Durand F (2009) User assisted intrinsic images. *ACM Trans Graph (Proceedings of SIGGRAPH Asia 2009)* 28(5)

Image Enhancement and Restoration

Guoshen Yu¹ and Guillermo Sapiro²

¹Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN, USA

²Electrical and Computer Engineering, Computer Science, and Biomedical Engineering, Duke University, Durham, NC, USA

Synonyms

Image inverse problems

Related Concepts

► Denoising; ► Image-Based Modeling; ► Inpainting

Definition

Image enhancement and restoration is a procedure that attempts to improve the image quality by removing the degradation while preserving the underlying image characteristics.

Background

Image quality is often deteriorated during acquisition, compression, and transmission. Typical degradations include image blur introduced by lens out-of-focus,

resolution downgrade due to acquisition equipment pixel limitation, noise spots introduced at high ISO, and JPEG block artifact, as illustrated in Fig. 1. Image enhancement and restoration is a procedure that attempts to improve the image quality by removing the degradation while preserving the underlying image characteristics. For some specific degradations as mentioned above, image enhancement and restoration is also known as deblurring, super-resolution zooming, denoising, and deblocking. While jointly addressed here and in most of the literature, *restoration* often refers to the case where one attempts to mathematically invert the degradation (e.g., invert the blurring filter), and *enhancement* refers to the improvement of the overall image quality without explicit mathematical inversion of the degradation process.

Theory

The problems of image enhancement and restoration are ill posed since they amount to recovering some image information that has been eliminated during the degradation. Solving these problems must therefore rely on some prior knowledge of the image, or in mathematical terms *image models*, to regularize the solution. Mathematically, let f denote an ideal image, U a linear degradation operator, w an additive noise, and

$$y = Uf + w \quad (1)$$

the degraded (observed) image. While this model does not cover all possible degradation scenarios, it is very popular and useful, and serves to illustrate the underlying image enhancement and restoration key concepts. Modern image enhancement and restoration estimates the underlying image f from the degraded observation y by, for example, minimizing a functional of the form

$$\hat{f} = \arg \min_h (\|y - Uh\|^2 + \varphi(h)), \quad (2)$$

where the first term ensures that the restored image \hat{f} and the degraded image y agree with the image degradation (Eq. 1), and the second term $\varphi(h)$ regularizes the solution via a certain image model. The technology of image enhancement and restoration thus has been developed hand in hand with a better understanding of image modeling.

Image Enhancement and Restoration, Fig. 1 From left to right. Ideal image, image degraded by out-of-focus, resolution downgrade, noise spots, and JPEG block artifact

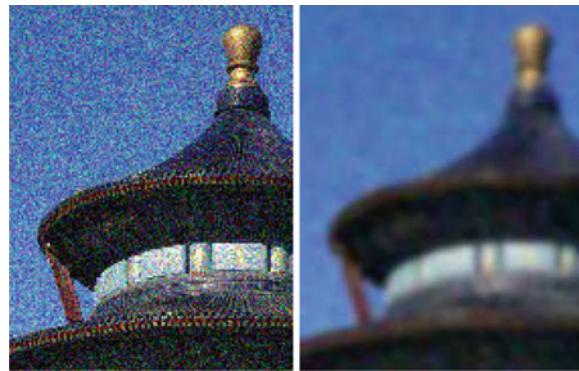


Image Enhancement and Restoration, Fig. 2 Left: noisy image. Right: image denoised by Gaussian smoothing

The most classic image model that dated from the 1960s assumes that image content is uniformly smooth [10]. This model results in a number of well-known image enhancement and restoration algorithms, including Gaussian smoothing for denoising, bicubic interpolation for zooming, and Wiener filter for deblurring [12]. All these algorithms are implemented with *linear* filtering uniformly applied over the image, typical isotropic local filters smoothing out the image. While the uniformly smooth assumption holds on regular image regions such as sky or a blackboard, that typically dominates a natural image, it is obviously oversimplified on other important types of image transition structures, such as contours, that are smooth along one direction but not the other, and textures that are oscillatory patterns. As shown in Fig. 2, although image noise is attenuated, image contours become blurred at the end of restoration when this simple uniformly smooth model is assumed.

Anisotropic image models attempting to address this problem came into the scene in the early 1990s (with some works dating to the 1960s as well, by Gabor). As opposed to the uniformly smooth

assumption, the anisotropic models assume that an image is piecewise smooth, in other words, smooth inside each sub-region, and that at a contour or boundary of the regions where the image intensity sharply changes, the smoothness holds only along the contour direction but not in the perpendicular direction. These models give clearly a better image description and have been elegantly formulated in some partial differential equation frameworks such as anisotropic diffusion [7, 15] and total variation [16]. The resulting algorithms implement *nonlinear* filtering adaptive to the image content, uniformly smoothing inside each image sub-region, and smoothing only along the contour direction on the region boundaries. Therefore, image contours are better preserved.

Since the boom of wavelets in the early 1990s, multi-resolution harmonic analysis has lead to considerable efforts and improvements on image modeling and restoration [4, 12]. Wavelet analysis models an image from multiple resolutions; at each resolution, translating local wavelet atoms oscillating at the corresponding scale are used. The wavelet response is typically high on image transition structures, such as contours and textures, and negligible on regular regions. As a result, it does not only implement *non-linear* adaptive filtering, but also reveals the important concept of “sparse modeling”: wavelet analysis represents an image with only a few large wavelet coefficients that absorb most of the image energy, while the majority of wavelet coefficients quickly decay to zero. The wavelet’s sparsity as well as its performance in image enhancement and restoration have been later improved by geometric adaptive harmonic analysis such as curvelets [5] that include local directional atoms to catch the image contours.

In order to further promote the resulting sparsity relative to prefixed harmonic analysis dictionaries (Dictionary here means an ensemble of harmonic analysis atoms), such as wavelets or curvelets, *sparsifying learned dictionaries*, i.e., dictionaries that are learned

Image Enhancement and Restoration, Fig. 3 Image enhancement and restoration examples. (a) and (b) Super-resolution zooming: Low-resolution and zoomed images. (c) and (d) Deblurring: blurred and deblurred images (Figures reproduced from [17])

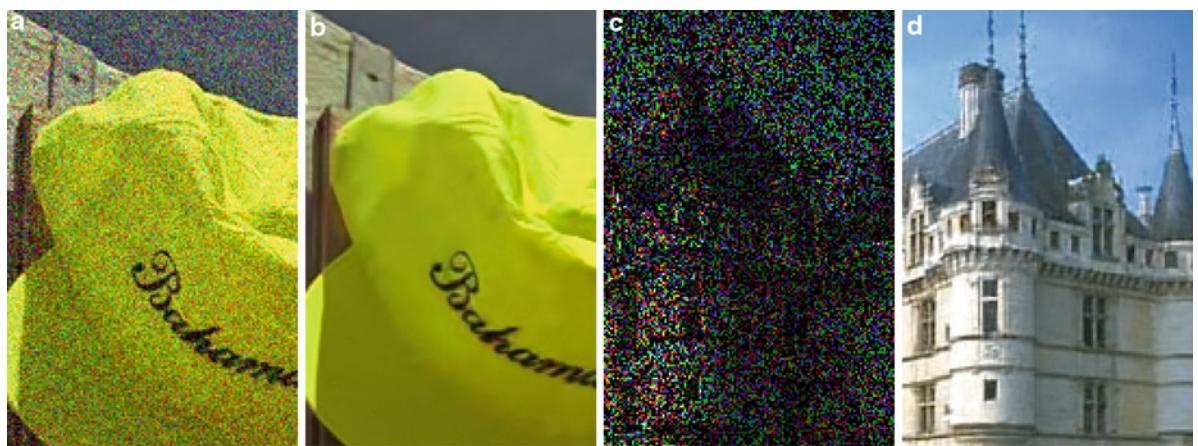
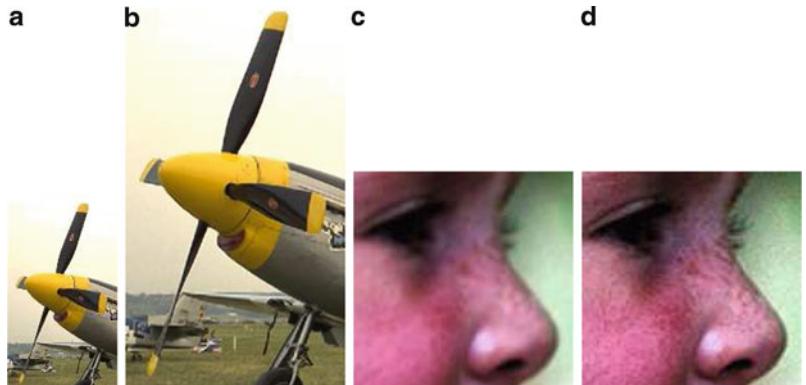


Image Enhancement and Restoration, Fig. 4 Image enhancement and restoration examples. (a) and (b) Denoising: Noisy and denoised images. (c) and (d) Inpainting: image with 80 % random missing pixels and restore image. (This

problem is related to the task of reconstructing a color image from undersampled color channels, as present in most low/mid-end digital cameras.) The results are obtained following the technique in [17]

from images of interest to yield sparse representations for that class of data, have emerged [1, 13], leading to further improved image enhancement and restoration performance [11].

Non-local image modeling is based on the observation that images typically contain repetitive local patterns (self-similarity). Since the pioneering work of the nonlocal means denoising algorithm [3] in 2005 (see also [2, 14]), non-local modeling has been extensively studied in image enhancement and restoration [8, 9].

Gaussian mixture models, a statistical model widely applied in machine learning, have been shown particularly effective for image enhancement and restoration [17]. The models assume that local image patches follow a mixture of Gaussian distributions. The resulting *piecewise linear* algorithm is not only extremely

fast, but also reveals some connections to sparse modeling and non-local modeling.

State-of-the-art image enhancement and restoration results are obtained with algorithms derived from the last three image models, namely, sparse modeling with learned dictionaries, non-local modeling, and Gaussian mixture models. Figure 3 illustrates some examples.

Open Problems

For image enhancement and restoration problems such as removing Gaussian white noise from an image and filling small holes at random positions in an image, it seems that the current performance is already acceptable, as illustrated in Fig. 4, and has arguably

reached a quality boundary uneasy to go beyond. For other more difficult problems such as deblurring and zooming, although substantial visual quality improvement has been achieved with respect to classic algorithms such as Wiener filter and bicubic interpolation, objective performance improvement is relatively limited despite considerable efforts that have been devoted. Theoretical performance bounds of image enhancement and restoration remains to be understood. The recent very exciting compressive sensing theory [6] reveals the performance bounds of the sparse modeling approaches given some random degradation operations, but is inapplicable to typical degradations such as blurring and subsampling and to the most successful learned dictionaries. The extension of these results to more realistic image degradation scenarios and image models is among the current challenges of image restoration and enhancement.

References

1. Aharon M, Elad M, Bruckstein A (2006) K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans Signal Process* 54(11):4311
2. Awate SP, Whitaker RT (2005) Higher-order image statistics for unsupervised, information-theoretic, adaptive, image filtering. In: Proceedings of conference on computer vision and pattern recognition (CVPR), vol 2, San Diego, pp 44–51
3. Buades A, Coll B, Morel JM (2006) A review of image denoising algorithms, with a new one. *Multiscale Modeling Simul* 4(2):490–530
4. Burt P, Adelson E (1983) The Laplacian pyramid as a compact image code. *IEEE Trans Commun* 31(4):532–540
5. Candes EJ, Donoho DL (2004) New tight frames of curvelets and optimal representations of objects with C^2 singularities. *Commun Pure Appl Math* 56:219–266
6. Candès EJ, Tao T (2006) Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Trans Inf Theory* 52(12):5406–5425
7. Catté F, Lions PL, Morel JM, Coll T (1992) Image selective smoothing and edge detection by nonlinear diffusion. *SIAM J Numer Anal* 29(1):182–193
8. Dabov K, Foi A, Katkovnik V, Egiazarian K (2007) Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Trans Image Process* 16(8):2080–2095
9. Gilboa G, Osher S (2008) Nonlocal operators with applications to image processing. *Multiscale Modeling Simul* 7(3):1005–1028
10. Lindenbaum M, Fischer M, Bruckstein A (1994) On Gabor's contribution to image enhancement. *Pattern Recognit* 27(1):1–8
11. Mairal J, Elad M, Sapiro G (2007) Sparse representation for color image restoration. *IEEE Trans Image Process* 17(1):53–69
12. Mallat S (2009) A wavelet tour of signal processing: the sparse way, 3rd edn. Academic, Burlington
13. Olshausen BA, Field DJ (1996) Natural image statistics and efficient coding*. *Netw Comput Neural Syst* 7(2):333–339
14. Ordentlich E, Seroussi G, Verdú S, Weinberger M, Weissman T (2003) A discrete universal denoiser and its application to binary images. In: Proceedings of international conference on image processing, vol 1, Barcelona
15. Perona P, Malik J (1990) Scale-space and edge detection using anisotropic diffusion. *IEEE Trans Pattern Anal Mach Intell* 12(7):629–639
16. Rudin LI, Osher S, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. *Phys D* 60(1–4): 259–268
17. Yu G, Sapiro G, Mallat S (2010) Solving inverse problems with piecewise linear estimators: from Gaussian mixture models to structured sparsity. *Image Processing, IEEE Transactions* 2481–2499

Image Inverse Problems

► Image Enhancement and Restoration

Image Mosaicing

► Image Stitching

Image Plane

Peter Sturm

INRIA Grenoble Rhône-Alpes, St Ismier Cedex,
France

Synonyms

Retina

Related Concepts

► Pinhole Camera Model

Definition

The image plane is the planar surface on which the image is generated in an image formation process or a model thereof.

Background

In most cameras, the photosensitive elements are arranged on a planar support. In image formation models, the image plane is the (mathematical) plane where the image is formed and within which pixels or film are supposed to be located.

There exist cameras where the photosensitive area is not flat. For instance, in most early panoramic image acquisition systems that proceeded by scanning a scene with a rotating slit camera, the film was wrapped onto the inside of a cylindrical surface [1, 2]. In that case, one may still devise an equivalent theoretical image formation model that has a planar image support surface.

References

1. McBride B (2011) A timeline of panoramic cameras. <http://www.panoramicphoto.com/timeline.htm> Accessed 3 August 2011
2. Benosman R, Kang S (2001) A brief historical perspective on panorama. In: Benosman R, Kang S (eds): Panoramic vision: sensors, theory, and applications. Springer, Verlag, pp 5–20

Image Registration

Daniel C. Alexander

Centre for Medical Image Computing, Department of Computer Science, University College London, London, UK

Synonyms

[Image alignment](#)

Definition

Image registration aligns corresponding features of images via spatial transformations.

Background

Computer vision or image processing systems often need to align multiple images of the same or similar

scenes. In medical imaging, for example, radiologists routinely compare images of a patient acquired at different times to monitor changes. The intensity difference between two images highlights such changes but only if the corresponding features are in the same location. However, patients' positions in imaging devices vary between visits, so raw images never have perfect alignment. Image registration transforms or warps one image so that the important objects and regions are in the same position as in the other image. The difference image then reveals intrinsic physical changes. [Figure 1](#) illustrates the idea. The problem becomes more challenging when the images come from different devices (inter-modality registration) or from different subjects (intersubject registration).

The same problem arises in nonmedical imaging applications. Surveillance systems, for example, often need to look for differences between images at different times, for example, to subtract the background and highlight activity in a scene viewed by a security camera. Fixed cameras can wobble in the wind and produce misaligned images that require registration before the difference image provides a meaningful result. Stitching images together to create panoramas [1–3] also requires image registration to align the overlapping parts of the images being stitched together; [Fig. 2](#) illustrates this application. Similarly, super-resolution techniques [4] align multiple images of the same scene and infer subpixel detail.

Theory

The process of automatic image registration involves optimizing a cost function, which expresses the similarity of the two images, with respect to the parameters of a transformation of one of the images. Mathematically, the optimization problem is

$$\{T^*, g^*\} = \underset{\{T, g\}}{\operatorname{argmin}}(f(I_1, T(g(I_2)))), \quad (1)$$

where I_1 is the target image, which is fixed; I_2 is the source image, which the transformations T and g act upon; T is a spatial transformation, or warp; and g affects only the image intensity at each pixel position; the optimization seeks the transformations T^* and g^* that minimize the cost function f .

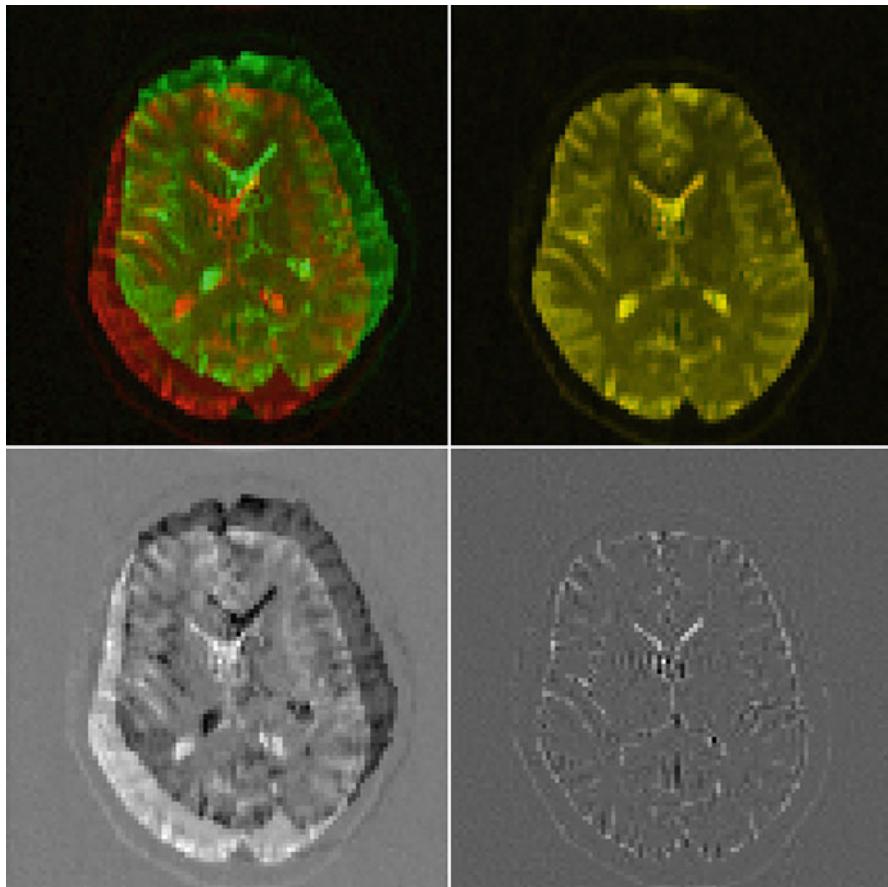


Image Registration, Fig. 1 Intrasubject brain image registration. *Top left*: overlaid images of the same brain from different acquisitions, one in red and one in green; *bottom left*:

difference image of the two unaligned images; *top right*: overlaid images after registration via a rigid transformation; *bottom right*: difference image after alignment

The process decomposes into four key components, which the following subsections discuss one by one. The literature contains many review papers, for example, [5–8], that discuss each component in more detail.

Dense features are typically pixel-by-pixel image intensities. Each feature may be a single scalar intensity or may have multiple components, as in multispectral images. Dense feature sets may not include every pixel in the image and often exclude pixels that lie outside the salient region of the image, such as the brain region in Fig. 1.

Features

Various image features can drive the registration process. Broadly, the feature set is either sparse or dense.

Sparse feature sets consist of geometric features identified in the image through some preprocessing step. These features might be salient points identified by a user or by an automatic detector; SIFT features [9] or variants thereof are a common choice. Features may also be more complex geometric objects, such as salient lines, curves, surfaces, or regions.

Cost Function

The cost function provides a measurement of similarity between two images. The definition of similarity depends on the set of features.

For sparse feature sets, the cost function typically uses a measure of distance between matched features in the two images. For example, if the feature set is a list of salient points in each image, the Euclidean distance between each corresponding pair of points

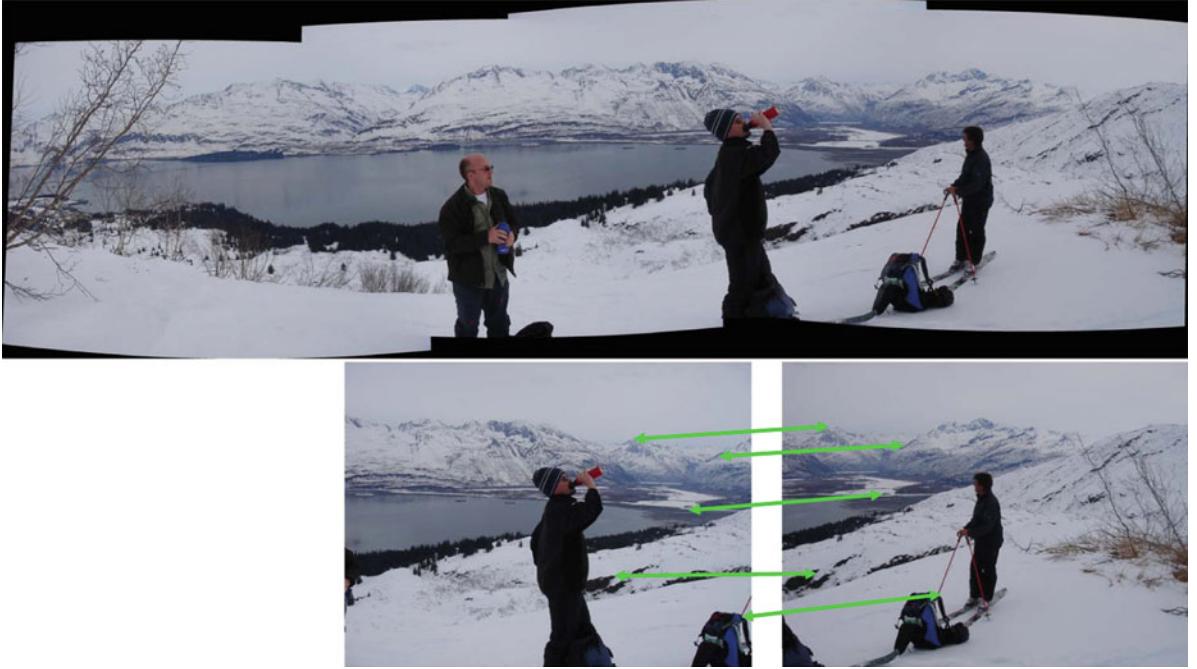


Image Registration, Fig. 2 Image registration for stitching. The panoramic image at the *top* comes from stitching together various images including the two at the *bottom*. Image

registration provides the spatial transformation that associates corresponding salient points in the two images, such as those marked by the *green arrows*

provides a measure of similarity. This requires a preceding step to establish correspondence between pairs of points, in a similar way to various other computer vision tasks, such as stereo matching.

Registration based on dense features, that is, pixel intensities, typically uses statistical measures of similarity between pixel intensities in corresponding locations. The most direct measure of similarity uses the average intensity difference

$$f_1(I_1, I_2) = - \sum_{x \in X} |I_1(x) - I_2(x)|, \quad (2)$$

where X is the salient set of pixels. Equation 2 uses the L_1 norm, but other norms are equally possible.

Direct intensity comparisons, as in f_1 , assume that the pixel intensity at corresponding locations is the same subject to some noise perturbation. However, that assumption often does not hold. For example, differences in intensity scale between images arise frequently. Where such intensity differences are likely, similarity measures based on the correlation of pixel

intensities between the two images are more appropriate. For example, the normalized cross correlation

$$f_2(I_1, I_2) = \frac{\sum_{x \in X} (I_1(x) - \bar{I}_1)(I_2(x) - \bar{I}_2)}{\sqrt{\sum_{x \in X} (I_1(x) - \bar{I}_1)^2 \sum_{x \in X} (I_2(x) - \bar{I}_2)^2}}, \quad (3)$$

where \bar{I} is the mean intensity of image I over region X .

The mapping between intensities of corresponding pixels is sometimes more complex than a simple scale change. It may be nonlinear and non-monotonic. For example, in inter-modality medical image registration, two images of the same object may have the same regional structure but different regional contrast: image 1 has higher intensity than image 2 in some regions, vice versa in others, and intensity correlation at alignment remains low. Entropy-based similarity measures [10–13] provide a useful alternative. A common choice is the normalized mutual information

$$f_3(I_1, I_2) = \frac{H(I_1) + H(I_2)}{H(I_1, I_2)}, \quad (4)$$

where

$$H(I) = - \sum_{x \in X} p(I(x)) \log p(I(x)) \quad (5)$$

is the entropy of image I , with p the distribution of image intensities in I , and

$$H(I_1, I_2) = - \sum_{x \in X} p(I_1(x), I_2(x)) \log p(I_1(x), I_2(x)) \quad (6)$$

is the joint entropy of images I_1 and I_2 , with p now the joint intensity distribution. The cost function f_3 is minimum when one image predicts the other most parsimoniously, that is, when the intensity mapping from I_1 to I_2 requires the least information to describe. Mutual-information-based cost functions prove remarkably effective and robust. In practice, they are often preferred to direct comparison or correlation-based cost functions, such as f_1 and f_2 , even for intrasubject intra-modality registration.

Transformation Model

A variety of models are available for the spatial transformation, T . Simple transformations, such as rigid, affine, or polynomial transformations, are global in the sense that even well-separated pixels undergo highly correlated displacements. More complex models, such as spline [14], radial-basis-function [15], elastic [16], or fluid [17] transformations, can have more local properties, so that the displacement of one pixel under the transformation correlates only with that of proximal pixels.

In some applications, simple global transformations are sufficient. For example, in brain imaging, rigid transformations are often sufficient to align two images from the same subject. Since a rigid skull encases the brain, it deforms very little between image acquisitions. The registration needs to correct only for the difference in position and orientation of the subject in the imaging device. Higher-order global transformations, such as full affine or polynomial transformations, can improve alignment significantly even when the physical transformation is rigid, because they can capture artifactual distortions introduced by the image device. Image stitching often uses a homography, which is a global transformation that accounts for changes in perspective.

Local transformations can capture more subtle changes between images. They are essential, for example, for detecting and quantifying local atrophy (shrinkage) of brain tissue that occurs over time in various neurological conditions [18]. In general, in medical imaging, local transformations are usually necessary for good alignment in intersubject image registration, where local variations in size and shape of organs and body structure arise.

In practice, the intensity transformation, g in Eq. 1, is often the identity. However, g becomes important in images that contain more complex information at each pixel than single or multiple scalar values. For example, vector or tensor images are common in remote sensing and medical imaging. In such images, each pixel has an associated orientation. Nontrivial g is essential to ensure that local orientations remain consistent with the image structure through the spatial transformation; see, for example, [19, 20].

Optimization

The wide range of optimization algorithms available today, from simple line search or gradient descent to stochastic and genetic optimization procedures, provides many candidates for driving the minimization of the cost function that solves the registration problem. The choice of optimization procedure depends on the feature set. Registration via sparse feature matching often relies on algorithms like RANSAC [21], which are robust to errors in point correspondence, whereas most image registration algorithms with dense features use some form of gradient descent. The cost function is almost always non-convex, and an effective optimization procedure for reliable image registration cannot ignore local minima. Even for simple rigid transformations, local minima often arise and reliable rigid registration with gradient descent requires repeated runs with multiple starting points [22]. The optimization problem tends to become harder the more complex the transformation model. In local registration, the optimization has a much larger number of parameters, so takes longer, and repeated runs can be impractical. Hierarchical approaches, which start with a simple global registration to get a good starting point and gradually add parameters and reoptimize, are common to obtain a good local registration. Multi-resolution strategies, which start with low-resolution images and gradually increase resolution, also help.

Application

The medical imaging community is a large consumer and developer of image registration techniques. Intra-subject image registration enables fusion of information in images from different devices. Intrasubject registration also enables tracking of changes over time during development or disease. In drug trials, for example, imaging offers the potential to observe the effects of a prospective treatment and establish its efficacy noninvasively; image registration is essential for monitoring such effects.

Another major application is spatial normalization for group studies, which study the variation in the size, shape, and internal organization of a particular organ or object. A common application is human brain mapping where morphological variability is well studied in a range of conditions. Intersubject image registration ensures that a collection of similar images are in the same spatial frame of reference so that studies of variation are meaningful. This spatial normalization allows, for example, medical imaging researchers to characterize differences in organ size, shape, and structure between different populations, such as normal healthy adults and patients with a certain condition.

Image registration for image stitching enables day-to-day image processing for digital camera users, as standard packages like Photoshop include such operations. Google Maps is a large-scale application of the same technology.

Many implementations of image registration are freely available. Tried and tested global registration software includes the FLIRT package [22] and Nifty Reg [23]. The popular b-spline registration algorithm [14] has implementations with some variations in FNIRT [24] and Nifty Reg [25], which also offers a GPU implementation. The DARTEL package [26] is designed specifically for spatial normalization of large brain image ensembles. The recent ANTS package [27] combines several state-of-the-art ideas and performs well in a head-to-head evaluation with other standard packages [28].

Open Problems

Consistency remains an open problem in image registration. Basic algorithms do not ensure that the transformation from registering images A to B is

the perfect inverse of that from registering images B to A. Symmetry constraints on the cost function to ensure binary consistency are straightforward to enforce. However, the problem becomes more complex as the number of images to align increases: ensuring consistency of A to B to C with C to A is more challenging. Groupwise registration, as in [26], goes some way towards ameliorating this problem.

Topological differences or changes present a further open challenge. Most transformation models do not accommodate differences in topology naturally between images. In fact, significant effort has gone into developing diffeomorphic transformation models that cannot fold or tear. However, topological differences arise frequently. In intersubject medical image registration, for example, it is not uncommon for an anatomical feature in one person to be entirely missing in another. The same problem can arise even intra-subject, say, before and after surgery to remove a tumor.

References

- Szeliski R, Shum HY (1997) Creating full view panoramic image mosaics and environment maps. In: Proceedings of the 24th annual conference on computer graphics and interactive techniques. SIGGRAPH '97, New York, NY, USA. ACM/Addison-Wesley, pp 251–258
- Szeliski R (2006) Image alignment and stitching: a tutorial. Found Trends Comput Graph Vis 2:1–104
- Brown M, Lowe D (2007) Automatic panoramic image stitching using invariant features. Int J Comput Vis 74: 59–73. doi:10.1007/s11263-006-0002-3
- Irani M, Peleg S (1991) Improving resolution by image registration. CVGIP 53(3):231–239
- Maintz J, Viergever MA (1998) A survey of medical image registration. Med Image Anal 2(1):1–36
- Lester H, Arridge SR (1999) A survey of hierarchical non-linear medical image registration. Pattern Recognit 32(1):129–149
- Hajnal JV, Hill DLG, Hawkes DJ (2001) Medical image registration. CRC Press, Boca Raton. ISBN 0-8493-0064-9
- Zitov B, Flusser J (2003) Image registration methods: a survey. Image Vis Comput 21(11):977–1000
- Lowe DG (2004) Distinctive Image Features from Scale-Invariant Keypoints. Int J Comput Vision 60(2): 91–110
- Studholme C, Hill DLG, Hawkes DJ (1995) Multiresolution voxel similarity measures for mr-pet registration. In: Proceedings of the information processing in medical imaging, Kluwer, pp 287–298
- Collignon A, Maes F, Delaere D, Vandermeulen D, Suetens P, Marchal G (1995) Automated multi-modality image registration based on information theory. In: Proceedings of the information processing in medical imaging, Kluwer, pp 263–274

12. Viola P, Wells WM (1995) Alignment by maximization of mutual information. In: Proceedings of the international conference on computer vision, IEEE, pp 16–23
13. Pluim JPW, Maintz JBA, Viergever MA (2003) Mutual information based registration of medical images: a survey. *IEEE Trans Med Imaging* 22:986–1004
14. Rueckert D, Sonoda LI, Hayes C, Hill DLG, Leach MO, Hawkes DJ (1999) Non-rigid registration using free-form deformations: application to breast mr images. *IEEE Trans Med Imaging* 18:712–721
15. Fornefelt M, Rohr K, Stiehl HS (2001) Radial basis functions with compact support for elastic registration of medical images. *Image Vis Comput* 19:87–96
16. Bajcsy R, Kovacic S (1989) Multiresolution elastic matching. *Comput Vis Graph Image Process* 46(1):1–21
17. Crum WR, Scahill RI, Fox NC (2001) Automated hippocampal segmentation by regional fluid registration of serial MRI: validation and application in alzheimer's disease. *NeuroImage* 13(5):847–855
18. Scahill RI, Schott JM, Stevens JM, Rossor MN, Fox NC (2002) Mapping the evolution of regional atrophy in alzheimer's disease: unbiased analysis of fluid-registered serial mri. *Proc Natl Acad Sci* 99:4703–4707
19. Alexander DC, Pierpaoli C, Basser PJ, Gee JC (2001) Spatial transformations of diffusion tensor magnetic resonance images. *IEEE Trans Med Imaging* 20:1131–1139
20. Zhang H, Yushkevich PA, Alexander DC, Gee JC (2006) Deformable registration of diffusion tensor mr images with explicit orientation optimization. *Med Image Anal* 10: 764–785
21. Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun ACM* 24:381–395
22. Jenkinson M, Smith S (2001) A global optimisation method for robust affine registration of brain images. *Med Image Anal* 5(2):143–156
23. Ourselin S, Roche A, Subsol G, Pennec X, Ayache N (2001) Reconstructing a 3d structure from serial histological sections. *Image Vis Comput* 19:25–31
24. Andersson J, Smith S, Jenkinson M (2008) FNIRT: FMRIB's non-linear image registration tool. In: 14th annual meeting of the organization for human brain mapping, OHBM
25. Modat M, Ridgway GR, Taylor ZA, Lehmann M, Barnes J, Fox NC, Hawkes DJ, Ourselin S (2009) Fast free-form deformation using graphics processing units. *Comput Methods Programs Biomed* 98(3):278–284
26. Ashburner J (2007) A fast diffeomorphic image registration algorithm. *NeuroImage* 38(1):95–113
27. Avants B, Epstein C, Grossman M, Gee J (2008) Symmetric diffeomorphic image registration with cross-correlation: evaluating automated labeling of elderly and neurodegenerative brain. *Med Image Anal* 12(1):26–41. Special issue on the third international workshop on biomedical image registration – WBIR 2006
28. Klein A, Andersson J, Ardekani BA, Ashburner J, Avants B, Chiang MC, Christensen GE, Collins DL, Gee J, Hellier P, Song JH, Jenkinson M, Lepage C, Rueckert D, Thompson P, Vercauteren T, Woods RP, Mann JJ, Parsey RV (2009) Evaluation of 14 nonlinear deformation algorithms applied to human brain mri registration. *NeuroImage* 46(3):786–802

Image Stitching

Matthew Brown

Dept of Computer Science, University of Bath,
Bath, UK

Synonyms

[Image Mosaicing](#); [Panoramic stitching](#)

Related Concepts

► [Environment Mapping](#)

Definition

Image stitching is the process of combining multiple overlapping images to generate a new image with a larger field of view than the originals.

Background

Image stitching can enhance the capabilities of an ordinary camera, enabling the capture of larger field-of-view, higher-resolution images. A popular example is the construction of panorama images by seamlessly combining several images of a scene taken from the same point (This process is known as panoramic stitching, which refers to the special case of image stitching for rotational motion). By capturing images with variable exposure settings, it can also be used to generate images with a higher dynamic range than the originals.

Stitching techniques were originally used in photogrammetry to produce maps from aerial and satellite images. Early techniques involved manual specification of matching images and control points (correspondences) between them [1]. Later methods used automated image alignment [2, 3] and interactive viewers to visualize the results [4]. Modern stitching pipelines offer fully automated operation [3], seam selection [5], and photometric, as well as geometric alignment [6].

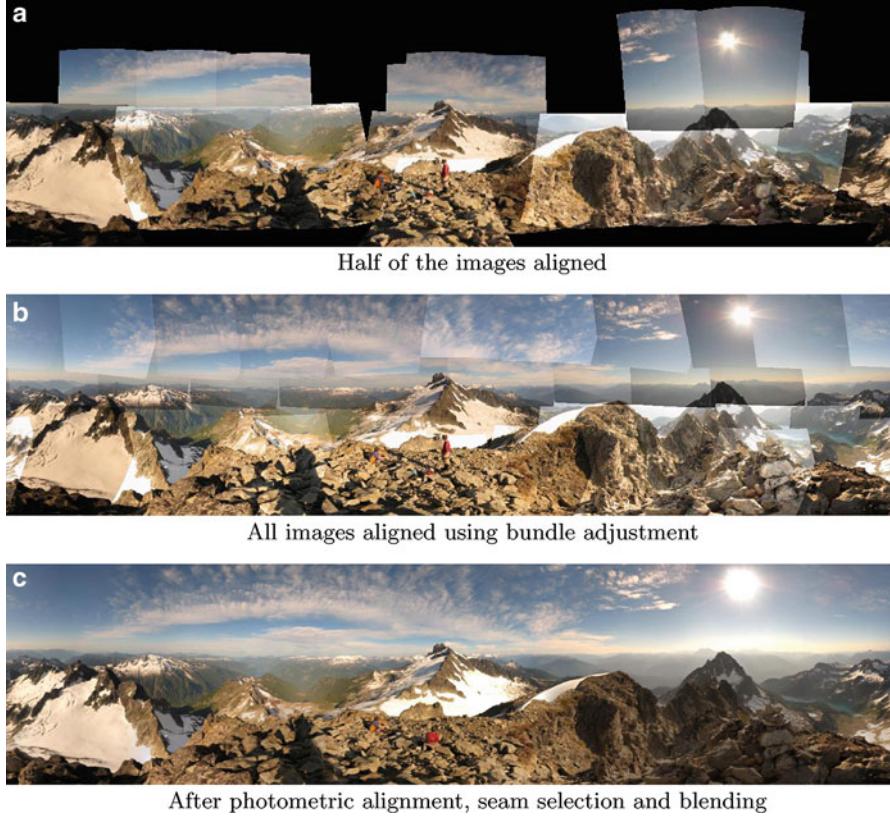


Image Stitching, Fig. 1 Panoramic stitching. Images are first geometrically aligned (using a rotational motion model in this case). Photometric alignment is used to compensate for brightness variations between the images, and the final panorama is

A typical pipeline for image stitching consists of the following stages (see Fig. 1):

1. Estimating two-frame motion and discovering overlaps between the images
2. Global alignment (e.g., using bundle adjustment [7])
3. Photometric alignment and seam selection/deghosting
4. Rendering the final panorama with blending and/or tone mapping

rendered using seam selection and pyramid blending. (a) Half of the images aligned. (b) All images aligned using bundle adjustment. (c) After photometric alignment, seam selection and

be rectilinear, the image coordinates are related by a homography

$$\tilde{\mathbf{u}}_2 = \mathbf{H}_{12}\tilde{\mathbf{u}}_1 , \quad (1)$$

where $\tilde{\mathbf{u}}_1, \tilde{\mathbf{u}}_2$ are the homogeneous coordinates in image 1 and 2 and \mathbf{H}_{12} is a 3×3 matrix that encodes the relative camera positions. For example, in the rotational case, \mathbf{H}_{12} is given by

$$\mathbf{H}_{12} = \mathbf{K}_2 \mathbf{R}_2 \mathbf{R}_1^T \mathbf{K}_1^{-1} , \quad (2)$$

Theory and Applications

Image stitching is possible when a one-to-one mapping exists between the source image coordinates. Two commonly occurring examples are: (1) a camera rotating about its optical center and (2) cameras viewing a planar scene. If the cameras are assumed to

where $\mathbf{R}_1, \mathbf{R}_2$ are the rotation matrices of cameras 1 and 2 and $\mathbf{K}_1, \mathbf{K}_2$ contain the intrinsic parameters.

A typical image stitching approach begins by robustly estimating \mathbf{H}_{12} from correspondences of local image features [8]. A standard method is to use the RANSAC algorithm [9] to sample the space of transformation hypotheses, for all images with a

sufficiently large number of feature matches. One can then reason about the adjacency relationships and recognize panoramas by making a match/no-match decision for each pair and finding connected components in the resulting graph of image matches [3].

After pairwise alignment, gaps and inconsistencies can still exist. Bundle adjustment [7] can be used to minimize projection errors between feature matches in all images and generate globally consistent results. Best results are achieved by parameterizing in terms of the intrinsic and extrinsic parameters of the cameras (e.g., rotation, focal length, radial distortion) [2]. Direct methods [10] (using all of the pixel data instead of only feature points) may optionally be used for accurate final registration.

Once the images are geometrically aligned, the remaining task is to render a seamless output view. The appropriate render surface may depend on the images being aligned: rectilinear renderings (preserving straight lines) might be best for stitching planar surfaces such as whiteboards, spherical or cylindrical render surfaces are popular for wide-angle panoramas. Multiperspective renderings can be used to preserve important geometric properties in the output [11].

Ideally, one can capture or estimate irradiance values per pixel, and given perfect alignment, these would be equal in all images overlapping a given ray. In practice, however, several sources of error contribute toward differences in the recorded radiances. Some common examples are parallax due to motion of the camera center, errors or unmodeled parameters in the camera pose estimate, and moving objects in the scene. Several algorithms have been developed to eliminate the visual seams that result. The best approaches find seam lines which minimize differences between image intensities or radiances [12], and smoothly interpolate between images using pyramid blending [13] or gradient domain fusion [5]. The final results can be tone mapped for display.

An example of an automated capture system capable of stitching gigapixel panoramas with feature-based alignment, seam selection, and dynamic tone mapping is given in [14].

References

- Slama CC (ed) (1980) Manual of photogrammetry, 4th edn. American Society of Photogrammetry, Falls Church, Virginia

- Szeliski R, Shum H (1997) Creating full view panoramic image mosaics and environment maps. *Comput Graph (SIGGRAPH'97)* 31(Annual Conference Series):251–258
- Brown M, Lowe D (2007) Automatic panoramic image stitching using invariant features. *Int J Comput Vis* 74(1):59–73
- Chen S (1995) QuickTime VR – an image-based approach to virtual environment navigation. In: ACM transactions on graphics (SIGGRAPH'95), vol 29, pp 29–38
- Agarwala A, Dontcheva M, Agarwala M, Drucker S, Colburn A, Curless B, Salesin D, Cohen M (2004) Interactive digital photomontage. In: ACM transactions on graphics (SIGGRAPH'04)
- Eden A, Uyttendaele M, Szeliski R (2006) Seamless image stitching of scenes with large motions and exposure differences. In: IEEE computer society conference on computer vision and pattern recognition (CVPR'06)
- Triggs W, McLauchlan P, Hartley R, Fitzgibbon A (1999) Bundle adjustment: a modern synthesis. In: Vision algorithms: theory and practice, number 1883 in LNCS. Springer, Corfu, September 1999, pp 298–373
- Szeliski R (2010) Computer vision: algorithms and applications, Chapter 9. Springer, <http://www.springer.com/computer/image+processing/book/978-1-84882-934-3>
- Fischler M, Bolles R (1981) Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun ACM* 24: 381–395
- Irani M, Anandan P (1999) About direct methods. In: Triggs B, Zisserman A, Szeliski R (eds) Vision algorithms: theory and practice, number 1883 in LNCS. Springer, Corfu, September 1999, pp 267–277
- Zelnik-Manor L, Peters G, Perona P (2005) Squaring the circle in panoramas. In: Tenth IEEE international conference on computer vision (ICCV'05), Beijing, pp 1292–1299
- Davis J (1998) Mosaics of scenes with moving objects. In: IEEE computer society conference on computer vision and pattern recognition (CVPR'98), pp 354–360
- Burt P, Adelson E (1983) A multiresolution spline with application to image mosaics. *ACM Trans Graph* 2(4): 217–236
- Kopf J, Uyttendaele M, Deussen O, Cohen M (2007) Capturing and viewing gigapixel images. In: ACM transactions on graphics (SIGGRAPH'07), vol 26

Image-Based Lighting

Tien-Tsin Wong

Department of Computer Science and Engineering,
The Chinese University of Hong Kong, Hong Kong
SAR, China

Synonyms

Environment mapping; Reflection mapping

Related Concepts

► Plenoptic Function

Definition

Image-based lighting [1, 2] is a rendering technique to compute the reflection from a 3D object lit in a distant environment, represented as an image, typically in the form of a cubemap (Fig. 1).

Background

Due to the computational expense of global illumination (e.g., radiosity and Monte Carlo ray tracing), most real-time graphics systems are depth-buffering based and only support local illumination. This hurts the realism of the rendered images. Environment mapping [1, 3] is proposed to simulate the reflection of the surrounding environment on an object surface (Fig. 1). The enclosing environment is assumed to be infinitely far away (distant environment) because all surface points on the object are assumed to be lit by the same environment. Due to the high computational expense, early implementations of environment mapping account only for the light contribution along the mirror reflection direction. Hence, most surfaces rendered by the environment mapping are over-shiny.

Image-based lighting can be regarded as a more comprehensive realization of the environment mapping, by accounting not only the light contribution along the mirror reflection but also the whole enclosing sphere. The surface reflectance property (bidirectional reflectance distribution function, BRDF) is also considered so as to render not only shiny or glossy surfaces but also most kinds of surfaces. Moreover, the environment maps are usually captured as high dynamic range (HDR) images to further increase the photorealism. Note that the image-based lighting remains work even low-dynamic range (LDR) environment maps are used instead.

Theory

Extending from accounting only the mirror reflection direction to the whole enclosing sphere drastically increases the computational expense. Hence, the

challenge is how to efficiently compute the following integration for each surface point:

$$I(x, s) = \int_{\Omega} L_{\text{in}}(\omega) \rho(x, \omega, u) v(x, \omega) (\omega \cdot n) d\omega \quad (1)$$

where x is the current surface point of interest; Ω is the surrounding environment (the distant environment map); ω is the incoming light direction; u is the viewing direction from x towards the eye; I is the reflected light; $L_{\text{in}}(\omega)$ is the incoming light contribution along direction ω , in other words, a point in the environment map; v is the visibility function; ρ is the BRDF; and n is the surface normal at x . Note that L_{in} , ρ , and v are spherical functions.

One way to evaluate the above integration is to approximate the environment map by a much smaller number (say m) of point light sources. The position and color of the point lights are obtained by importance sampling of the environment map [4]. In other words, the above integration is approximated by a summation of light contribution of m point lights. The rendered image can simply be generated by adding m images, each rendered by illuminating the scene with a point light source.

By adopting the image-based relighting techniques [5], the above integration can be evaluated more efficiently. The idea is to first encode the spherical function with basis functions. This effectively converts a huge spherical function (table) into a coefficient vector s_i as follows. Since the basis functions B_i are known, they need not be stored:

$$S(\omega) \approx \sum_i^k s_i B_i(\omega) \quad (2)$$

where S is a spherical function and k is the total number of basis functions, which is much smaller than the number of entries in the original spherical table.

By embedding $\rho(x, \omega, u) v(x, \omega) (\omega \cdot n)$ into a spherical function C , both C and L_{in} can be encoded with the same basis and stored as two coefficient vectors, c_i and l_i , respectively. If the selected basis functions are orthonormal, the above integration (Eq. 1) can be simply evaluated as a dot product between two coefficient vectors [1],

$$I \approx \sum_i^k c_i l_i \quad (3)$$

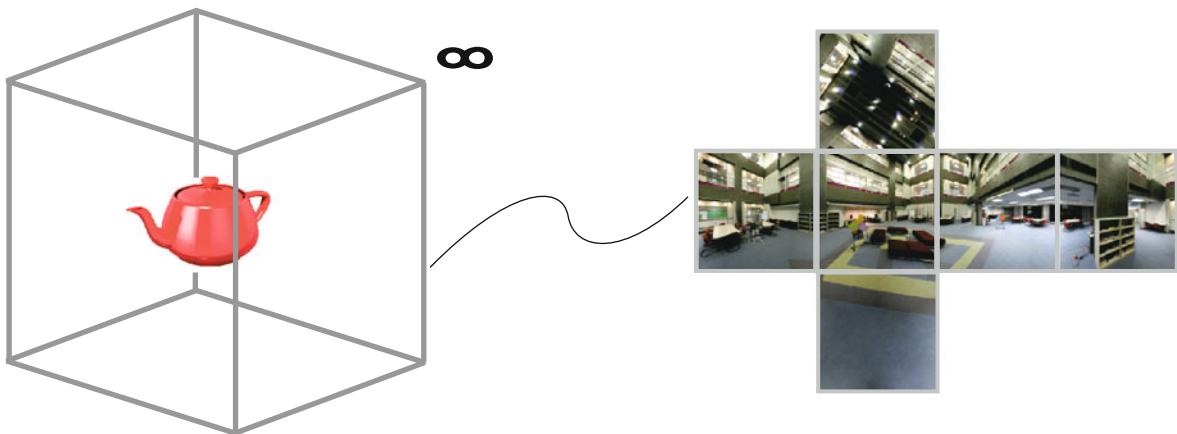


Image-Based Lighting, Fig. 1 Image-based lighting assumes the object being rendered is enclosed by an image-based environment positioned infinitely far away, or equivalently, the

object is infinitesimally small. The environment map is typically represented as a cubemap

Basis Functions

The key to efficient image-based lighting is to select an appropriate basis for representing the spherical functions. Various bases have been proposed for image-based relighting [5]. They are directly applicable to image-based lighting.

A pioneer work is proposed by Nimeroff et al. [6]. They efficiently relit the scene under various natural illumination (overcast or clear skylight). The illumination function is decomposed into a linear combination of *steerable functions*.

Principal component analysis is naturally a potential choice for basis function [7]. Singular value decomposition can be used to extract a set of eigenimages from the input reference images. The desired image can then be synthesized by a linear combination of these basis images given a set of coefficients [8, 9] if all surfaces are Lambertian.

Earlier works do not consider the spherical nature of the illumination computation. Wong et al. [5, 10] chose the *spherical harmonic basis*, which is commonly used for compressing BRDF. Pleasant rendering results are obtained with 16–25 basis functions. However, spherical harmonic is also well known in over-smoothing the high-frequency signal (e.g., shadow) in the original spherical function, leading to low-frequency results.

To achieve all-frequency rendering, Haar *wavelet basis* is proposed [11]. It may introduce visual artifact when the distant environment contains a dominant but

small-size spot. The cause of such artifact is due to the digitization of the spherical function and the limited reconstruction involving only important wavelet coefficients.

Spherical radial basis function (SRBF) [12–14] is another approach to capture all-frequency signal. The local support nature of SRBF allows its implementation to be very efficient and simple. The multiscale spherical radial basis function [15] avoids the visual artifact of the Haar wavelet basis while remains able to achieve all-frequency rendering.

Application

Image-based lighting can be applied to produce realistic rendering for both off-line movie production or real-time computer games. The parallel nature of image-based lighting (all surface points have to evaluate Eq. 1 independently) facilitates its real-time realization on modern graphics processing unit (GPU).

References

1. Cabral B, Max N, Springmeyer R (1987) Bidirectional reflection functions from surface bump maps. In: Proceedings of the 14th annual conference on computer graphics and interactive techniques. ACM, New York, pp 273–281

2. Debevec P (2002) Image-based lighting. *IEEE Comput Graphics Appl* 22(2):26–34
3. Blinn JF, Newell ME (1976) Texture and reflection in computer generated images. *Commun ACM* 19(10):542–547
4. Agarwal S, Ramamoorthi R, Belongie S, Jensen HW (2003) Structured importance sampling of environment maps. *ACM Trans Graphics* 22(3):605–612
5. Wong TT, Heng PA, Or SH, Ng WY (1997) Image-based rendering with controllable illumination. In: Proceedings of the 8th Eurographics workshop on rendering (Rendering techniques'97), St. Etienne, pp 13–22
6. Nimeroff JS, Simoncelli E, Dorsey J (1994) Efficient rendering of naturally illuminated environments. In: Fifth Eurographics workshop on rendering, Darmstadt, Germany, pp 359–373
7. Ho PM, Wong TT, Choy KH, Leung CS (2003) PCA-based compression for image-based relighting. In: Proceedings of IEEE international conference on multimedia and expo 2003 (ICME 2003), vol I, Baltimore, Maryland, USA, pp 473–476
8. Belhumeur PN, Kriegman DJ (1996) What is the set of images of an object under all possible lighting conditions. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), San Francisco
9. Zhang Z (1998) Modeling geometric structure and illumination variation of a scene from real images. In: Proceedings of the international conference on computer vision (ICCV'98), Bombay, India
10. Wong TT, Fu CW, Heng PA, Leung CS (2002) The plenoptic illumination function. *IEEE Trans Multimedia* 4(3):361–371
11. Ng R, Ramamoorthi R, Hanrahan P (2003) All-frequency shadows using non-linear wavelet lighting approximation. *ACM Trans Graphics* 22(3):376–381
12. Wong TT, Leung CS, Choy KH (2005) Lighting precomputation using the relighting map. In: Engel W (ed) *ShaderX³: advanced rendering with DirectX and OpenGL*. Charles Rivers Media, Hingham, pp 379–392
13. Leung CS, Wong TT, Lam PM, Choy KH (2006) An RBF-based image compression method for image-based rendering. *IEEE Trans Image Process* 15(4):1031–1041
14. Tsai YT, Shih ZC (2006) All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. In: Proceedings of ACM SIGGRAPH 2006, Boston, pp 967–976
15. Lam PM, Ho TY, Leung CS, Wong TT (2010) All-frequency lighting with multiscale spherical radial basis functions. *IEEE Trans Visual Comput Graphics* 16(1):43–56

Image-Based Modeling

Ping Tan
 Department of Electrical and Computer Engineering,
 National University of Singapore, Singapore,
 Singapore

Synonyms

[Three dimensional Modeling from Images](#)

Definition

Image-based modeling refers to the process of using two-dimensional images to create three-dimensional models. These models often consist of a geometric shape and a texture map defined over this shape.

Background

Three dimensional models are mathematical representations of three dimensional objects or scenes. These models are useful for various applications such as simulation, robotics, virtual reality, and digital entertainment. Automatically creating these models has been an important research topic since the early days of computer vision.

Broadly speaking, there are two schools of image-based modeling methods. One group employs range sensor for shape modeling, while the other group uses pure images with binocular or multi-view stereo. One of the origins of the first group is [1], which developed a technique to automatically generate a virtual three dimensional model by observing actual objects along the line of physics-based paradigm. Recently, this direction of research has been accelerated by the development of handy range sensor, such as the Microsoft Kinect and other consumer depth cameras. A representative work from the other group of pure image-based method is [2], which introduced an interactive method to model architectural scenes by fitting geometric primitives to the input images. This direction of research is later generalized to model more general objects. Typically, a cloud of 3D points is first recovered from the input images. A detailed and precise shape representation is then derived from these points, and a texture map is created to represent the color of each point on the shape. This process is illustrated in [Figure 1](#).

The process of obtaining 3D points from input images is known as 3D reconstruction. It is a well-studied problem, and most of the relevant theoretic results are summarized in this handbook [4]. There are also a number of well-established 3D reconstruction software systems such as [5–7]. Though the reconstructed 3D points can be used directly for certain measurements, most of applications require polygonal meshes, NURBS surfaces, or solid shape models. 3D modeling is the process of creating these shape models

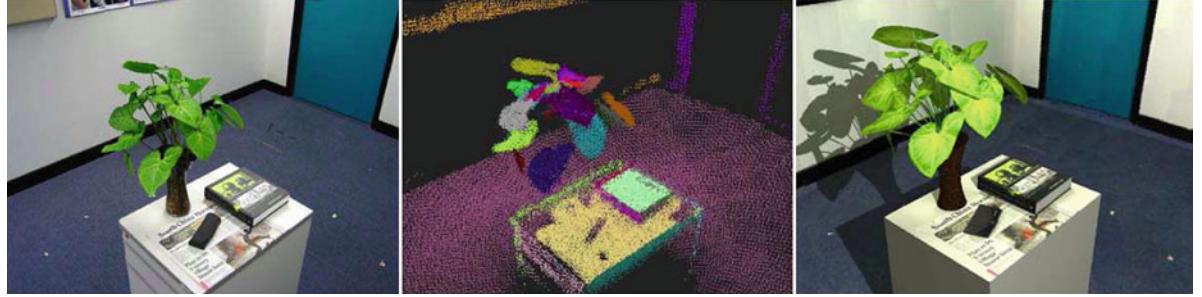


Image-Based Modeling, Fig. 1 A typical pipeline of image-based modeling. In the *left* is one of the input image. In the *middle* is the set of reconstructed 3D points. In the *right* is a rendering of the recovered 3D model of the scene (These pictures are from [3])

from 3D points and 2D images. This modeling process can be automatic or interactive. The prior knowledge of the shape to be modeled often plays an important role in this process. A comprehensive review of recent modeling techniques can be found in [8].

In image-based modeling, the appearance of a shape is often modeled by specifying a color for each point on the shape. These colors can be stored in a so-called texture map. Essentially, this representation assumes the surface is Lambertian and its Bidirectional Reflectance Distribution Function (BRDF) is a constant, that is, the color. In comparison, appearance modeling dedicates to model both the shape and precise surface reflectance properties. More details of appearance modeling can be found in [9].

Theory

In this short chapter, we only focus on the process of creating shape models from 3D points and 2D images. One way to generate these shape models is to obtain a minimal surface automatically from the input data. The minimal surface is a surface that minimizes a functional of the following form

$$\int \int w ds.$$

Here, ds is the infinitesimal surface element and w is the consistency of the surface according to the input 3D points and 2D images. This consistency can be simply the Euclidean distance between the surface and the set of 3D points. Given a set of points \mathcal{P} , Zhao et al. [10] defined w as $d(\mathbf{x}, \mathcal{P})$ which is the smallest distance

between a surface point \mathbf{x} and other points in the set \mathcal{P} . This functional is then minimized by the level set method [11]. Faugeras and Keriven [12] defined w as a function of both the surface position \mathbf{x} and its normal direction \mathbf{n} to facilitate the surface modeling. Lhuillier and Quan [13] further incorporated image reprojection errors and silhouettes in this function w .

The minimal surface-based approach works well when the points are dense and the surface is smooth. However, it has difficulties to model discontinuous surfaces such as hair fibers (linear structure), clothes (open surface patches), tree branches (tree and fractal structure), or buildings (regular axis aligned boxes). There are many existing methods which are designed to exploit the prior knowledge of a shape to facilitate modeling. Wei et al. [14] modeled hair by “growing” 3D smooth curves guided by 3D points and images. Bhat et al. [15] used videos to obtain the parameters of a cloth simulation system. Tan et al. [16] recovered some basic branch elements from the 3D points and used them to generate a fractal branch structure. Xiao et al. [17, 18] identified building facades and repetitive structures on these facades to model streets. Furukawa et al. [19, 20] assumed the scene consists of mutual orthogonal planes at different depth to model buildings. Though generating good results, these methods are limited to model the type of surface that matches their underlying prior shape assumption. A general modeling method is still missing to handle all these different data in a unified framework.

Application

Image-based modeling can be applied in autonomous robotics to generate a three-dimensional map of their

environment for path/action planning. It can also be used in industry vision for product quality inspection. The three-dimensional models can also be applied in digital entertainment such as games and movies. Recently, Google Earth and Microsoft Virtual Earth start to provide 3D map services, which can be a very good test bed of large-scale image-based modeling techniques.

References

1. Sato Y, Wheeler MD, Ikeuchi K (1997) Object shape and reflectance modeling from observation. Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '97, New York, NY, USA, ACM Press/Addison-Wesley Publishing Co, pp 379–387
2. Debevec PE, Taylor CJ, Malik J (1996) Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '96, New York, NY, USA, ACM, pp 11–20
3. Quan L, Wang J, Tan P, Yuan L (2007) Image-based modeling by joint segmentation. *Int J Comput Vis* 75:135–150
4. Hartley R, Zisserman A (2003) Multiple view geometry in computer vision, 2 edn. Cambridge University Press, New York
5. Lhuillier M, Quan L (2005) A quasi-dense approach to surface reconstruction from uncalibrated images. *IEEE Trans Pattern Anal Mach Intell* 27:418–433
6. Furukawa Y, Ponce J (2010) Accurate, dense, and robust multiview stereopsis. *IEEE Trans Pattern Anal Mach Intell* 32:1362–1376
7. Snavely N, Seitz SM, Szeliski R (2006) Photo tourism: exploring photo collections in 3d. *ACM Trans Graph* 25: 835–846
8. Quan L (2010) Image-based modeling, 1 edn. Springer, New York
9. Weyrich T, Lawrence J, Lensch HPA, Rusinkiewicz S, Zickler T (2009) Principles of appearance acquisition and representation. *Found Trends Comput Graph Vis* 4:75–191
10. Zhao HK, Osher S, Fedkiw R (2001) Fast surface reconstruction using the level set method. In: Proceedings of the IEEE workshop on variational and level set methods (VLSM'01). VLSM '01, Washington, DC, USA. IEEE Computer Society, p 194
11. Osher S, Sethian JA (1988) Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations. *J Comput Phys* 79:12–49
12. Faugeras OD, Keriven R (1998) Complete dense stereovision using level set methods. In: Proceedings of the 5th European conference on computer vision-volume i – volume i (ECCV '98). London, UK. Springer, pp 379–393
13. Lhuillier M, Quan L (2003) Surface reconstruction by integrating 3d and 2d data of multiple views. In: Proceedings of the 9th IEEE international conference on computer vision – volume 2. ICCV '03, Washington, DC, USA. IEEE Computer Society, p 1313
14. Wei Y, Ofek E, Quan L, Shum HY (2005) Modeling hair from multiple views. *ACM Trans Graph* 24:816–820
15. Bhat KS, Twigg CD, Hodgins JK, Khosla PK, Popović Z, Seitz SM (2003) Estimating cloth simulation parameters from video. In: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on computer animation. SCA '03, Aire-la-Ville, Switzerland. Eurographics Association, pp 37–51
16. Tan P, Zeng G, Wang J, Kang SB, Quan L (2007) Image-based tree modeling. *ACM Trans Graph* 26
17. Xiao J, Fang T, Tan P, Zhao P, Ofek E, Quan L (2008) Image-based facade modeling. *ACM Trans Graph* 27: 161:1–161:10
18. Xiao J, Fang T, Zhao P, Lhuillier M, Quan L (2009) Image-based street-side city modeling. *ACM Trans Graph* 28:114: 1–114:12
19. Furukawa Y, Curless B, Seitz S, Szeliski R (2009) Manhattan-world stereo. In: Proceedings of the IEEE international conference on computer vision and pattern recognition (CVPR '09). IEEE Computer Society
20. Furukawa Y, Curless B, Seitz S, Szeliski R (2009) Reconstructing building interiors from images. In: Proceedings of the IEEE international conference on computer vision. ICCV '09. IEEE Computer Society

Image-Based Rendering

Shing Chow Chan

Department of Electrical and Electronic Engineering,
The University of Hong Kong, Hong Kong, China

Synonyms

[Image-based rendering \(IBR\)](#)

Related Concepts

► [Light Field](#); ► [Lumigraph](#); ► [Plenoptic Function](#)

Definition

Image-based rendering (IBR) refers to a collection of techniques and representations that allows 3D scenes and objects to be visualized and manipulated in a realistic way without full 3D model reconstruction.

Background

One of the primary goals in computer graphics is photorealistic rendering. Motivated by the difficulties in achieving full photorealism with conventional 3D and model-based graphics, image-based rendering which works directly with real images has proposed as an alternative approach to reduce the rendering and capturing complexity. Depending on how the images are being taken and the auxiliary information, such as depths, etc., required, a number of image-based representations supporting different viewing freedom and functionalities are available. These range from the familiar two-dimension (2D) panoramas to more sophisticated representations such as the four-dimension (4D) light fields [9], lumigraphs [8], and variants, which are special cases of the radiance received at every viewing position, visual angle, wavelength, and time, called the plenoptic function.

The rendering of novel views can therefore be viewed as the reconstruction of the plenoptic function from its samples. Image-based representations are usually densely sampled high-dimensional data with large data sizes, but their samples are highly correlated. Because of the multidimensional nature of image-based representations and scene geometry, much research has been devoted to the efficient capturing, sampling, rendering, and compression of IBR.

Theory

Representation

In IBR, new views of scenes are reconstructed from a collection of densely sampled images or videos. Examples include the well-known panoramas [5], light fields [9], lumigraph [8], layered depth images [13], concentric mosaics (CM) [14], etc. Figure summarizes the concept of CM and light field (see the sections on light field, lumigraph, and plenoptic function for more illustration). The reconstruction problem (i.e., rendering) is treated as a multidimensional sampling problem, where new views are generated from densely sampled images and depth maps instead of building accurate 3D model of the scenes.

Depending on the functionality required, there is a spectrum of IBR as shown in Fig. . They differ from each other in the amount of geometry information of the scenes/objects being used. At one end of the

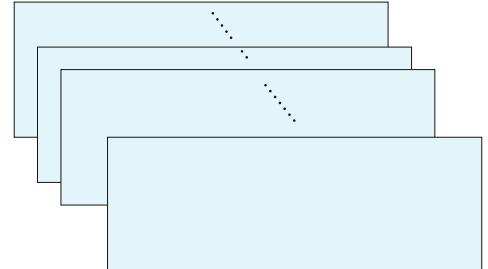
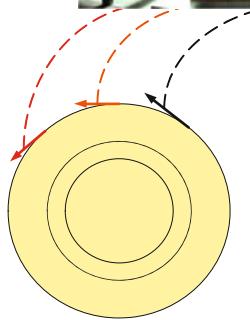
spectrum, like traditional texture mapping, very accurate geometric models of the scenes and objects say generated by animation techniques is used, but only a few images are required to generate the textures. Given the 3D models and the lighting conditions, novel views can be rendered using conventional graphic techniques. Moreover, interactive rendering with movable objects and light sources can be supported using advanced graphic hardware.

At the other extreme, light field or lumigraph rendering relies on dense sampling (by capturing more image/videos) with no or very little geometry information for rendering without recovering the exact 3D models. An important advantage of the latter is its superior image quality, compared with 3D model building for complicated real world scenes. Another important advantage is that it requires much less computational resources for rendering regardless of the scene complexity because most of the quantities involved are precomputed or recorded. This has attracted considerable attention in the computer graphic community in developing fast and efficient rendering algorithms for real-time relighting and soft-shadow generation [2, 12, 19, 22].

Broadly speaking, image-based representations can be classified according to the geometry information used into three main categories: (1) representations with no geometry, (2) representations with implicit geometry, and (3) representations with explicit geometry. 2-D panoramas, McMillan and Bishop's plenoptic modeling [11], and 3D concentric mosaics and light fields/lumigraph belong to the first category, and they can be viewed as the direct interpolation of the plenoptic function. Layere-based, object-based representations [4], pop-up light [16] using depth maps fall into the second. Finally, conventional 3D computer graphic models and other more sophisticated representations [7, 21, 22] belong to the last category. Although these representations also sample the plenoptic function, further processing of the plenoptic function has been performed to infer the scene geometry or surface property such as bidirectional reflectance distribution function (BRDF) of objects. Such image-based modeling approach has emerged as a more promising approach to enrich the photorealism and user interactivity of IBR. Moreover, since 3D models of the scenes are unavailable, conventional image-based representations are limited to the change of viewpoints and sometimes limited amount of relighting. Recently, it was found

Concentric mosaic

By constraining camera motion to planar concentric circles, concentric mosaic can be created by compositing slit images taken at different locations of each circle.



Light field

Using this 2D array of images, light field is possible to render different views of the object or scene at different viewing angles.

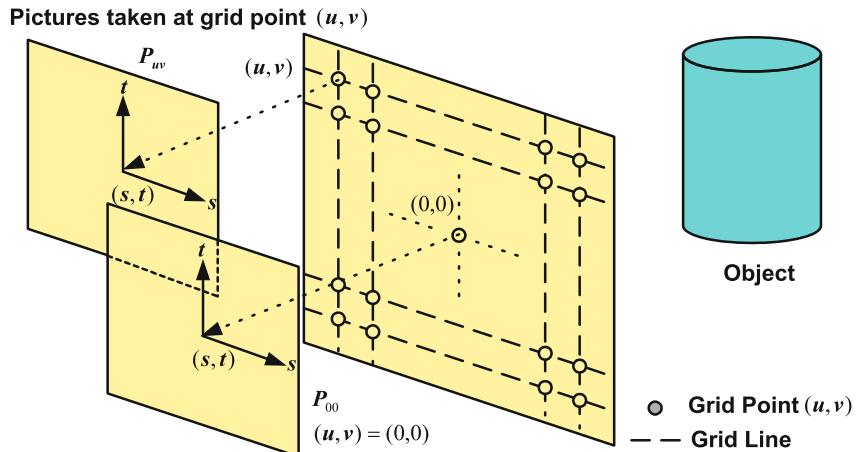


Image-Based Rendering, Fig. 1 Concentric mosaic and light field [3]

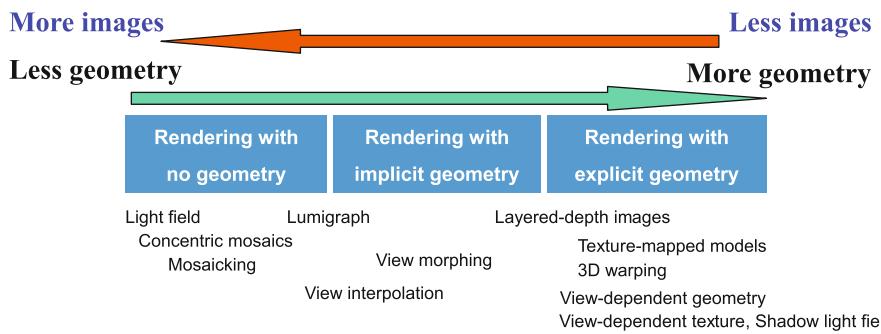


Image-Based Rendering, Fig. 2 Spectrum of IBR representations

that real-time relighting and soft-shadow computation are feasible using the IBR concepts and the associated 3D models using precomputed radiance transfer (PRT) [19] and precomputed shadow fields [22].

Earlier image-based representations are usually static, and their extension usually requires multiple camera arrays. Much research has been devoted to the capturing, compression, transmission, and processing

of these dynamic representations. For a review as of 2007, see [17].

Rendering

Rendering refers to the process of rendering of new views from the images and possibly other auxiliary information captured in the representations. For early image-based representations which do not employ any geometry information, rendering can be done simply by image blending as in panoramas [5] and ray-space interpolation in light field [9]. In ray-space interpolation, each ray that corresponds to a target screen pixel is mapped to nearby sampled rays. Figure 4a shows the example renderings of a simplified light field using ray-space interpolation [17]. For more sophisticated representations which use more geometry information such as layered depth images [13], surface light field [21], and pop-up light field [16], graphics hardware has been exploited to accelerate the rendering process. The geometry information can either be implicit that relies on positional correspondences or explicit in the form of depth along known lines-of-sight or 3D coordinates. Representations of the former usually involve weakly calibrated cameras and rely on image correspondences to render new views, say by triangulating two reference images into patches according to the correspondences as in joint view triangulation (JVT) [10]. These include view interpolation, view morphing, JVT, and transfer methods with fundamental matrices and trifocal tensors. Representations employing explicit geometry include sprites, relief textures, layered depth images (LDIs), view-dependent texture, surface light field, pop-up light field, shadow light field, etc.

In general, the rendering methods can be broadly classified into three groups [17]: (1) point-based, (2) layer-based, and (3) monolithic.

Point-Based Rendering works on 3D point clouds or point correspondences, and typically each point is rendered independently. Points are mapped to the target screen through forward mapping and variants. For the 3D point X in Fig. 3, the mapping can be written as

$$X = C_r + \rho_r P_r x_r = C_t + \rho_t P_t x_t \quad (1)$$

where x_t and x_r are homogeneous coordinates of the projection of X on target screen and reference images,

respectively. C and P are camera center and projection matrix, respectively, and ρ is a scale factor. Since C_t , P_t , and the focus length f_t are known for the target view, ρ_t can be computed using the depth of X . Given x_r and ρ_r , one can compute the exact position of x_t on the target screen and transfer the color accordingly. Gaps or holes may exist due to magnification and disocclusion, and splatting techniques have been proposed to alleviate this problem. The painter's algorithm is frequently used to avoid the problem of the mapping of multiple pixels from the reference view to the same pixel in the target view.

Layered Techniques usually discretize the scene into a collection of planar layers with each layer consisting of a 3D plane with texture and optionally a transparency map. The layers can be thought of as a continuous set of polygonal models, which is amenable to conventional texture mapping and view-dependent texture mapping. Usually, each layer is rendered using either point-based or polygon meshes as in monolithic rendering techniques before being composed in the back-to-front order using the painter's algorithm to produce the final view. Layer-based rendering is also easier to implement using graphic processing unit (GPU). Since the rendering of IBR requires very low complexity, it is even possible to perform the calculation using CPU by working on individual layer or object [4].

Monolithic Rendering usually represents the geometry as continuous polygon meshes with textures, which can be readily rendered using graphics hardware. The 3D model normally consists of vertices, normals of vertices, faces, and texture mapping coordinates. The data can be stored in a variety of data formats. The most popular formats are .obj, .3ds, .max, .stl, .ply, .wrl, .dxf, etc.

Relighting, shadow generation, and interactivity have played an increasingly important role in 3D interactive rendering. The most popular algorithms are shadow mapping, shadow volume, ray tracing, precomputed radiance transfer, precomputed shadow field, etc. Some of them have better rendering quality, while others are more efficient for real-time rendering. Thanks to the development of GPU, basic lighting, and shading algorithms like shadow mapping and shadow volume have been realized on the fly. Modern GPUs can even offer programmable rendering pipelines for customized rendering effects and "shader" is a set of software instructions running on

Image-Based Rendering, Fig. 3 Forward mapping

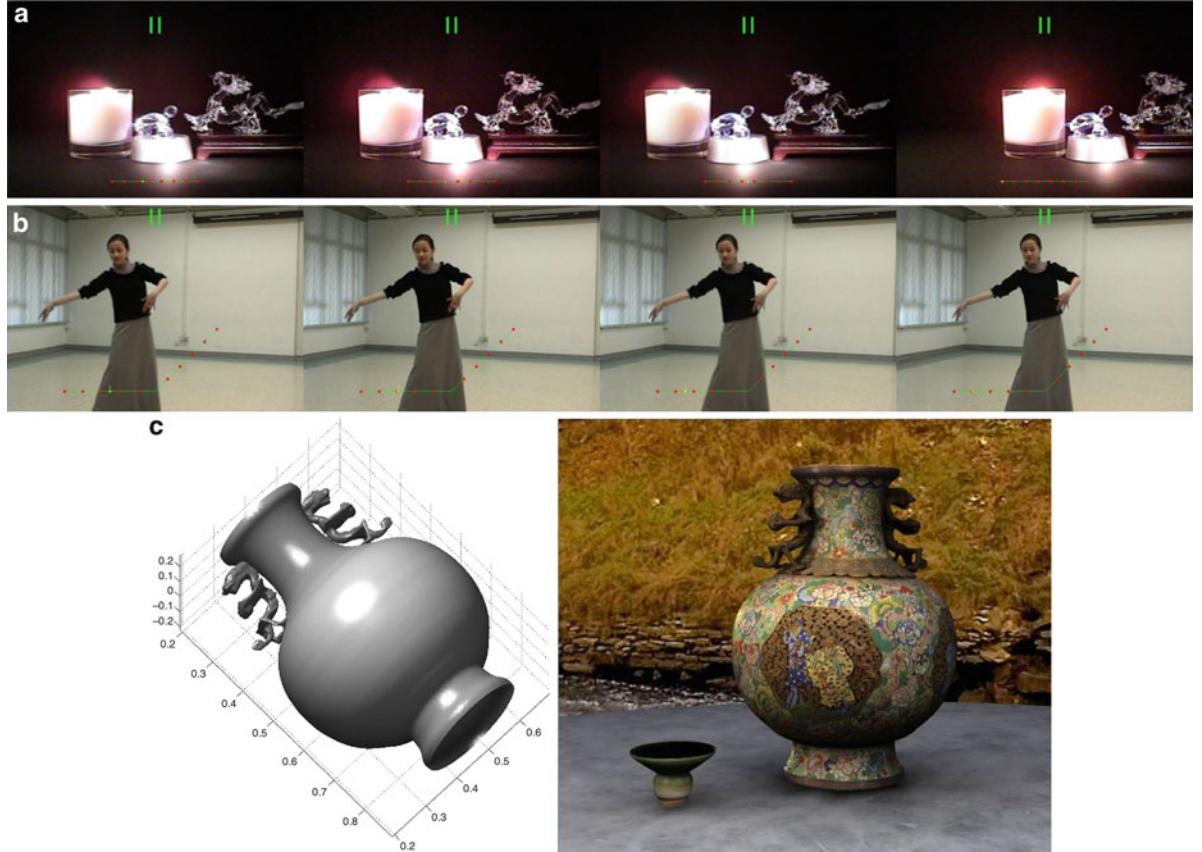
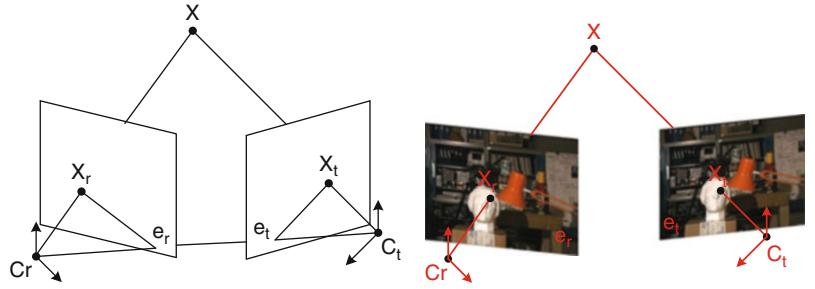


Image-Based Rendering, Fig. 4 Example renderings using (a) ray-space interpolation [17], (b) forward mapping in layered representation (with two layers – dancer and background) [4], (c)

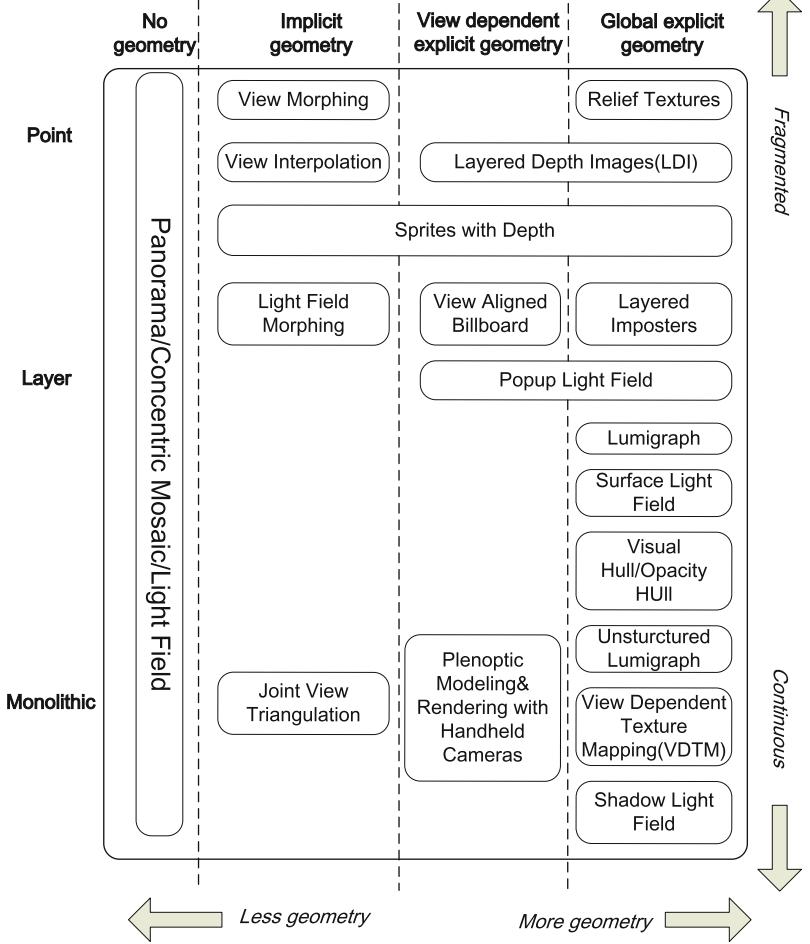
monolithic rendering using 3D polygonal mesh (*left*) estimated by multiview stereo and real-time rendering with shadow light field technique on GPU [23]

these GPUs to control the pipelines. Using shader programming, high-quality shadow rendering algorithms like precomputed shadow field can be done in real time. Figure 4 shows example renderings of the three techniques, and Fig. 5 summarizes the types of representations and rendering in IBR called the geometry-rendering matrix.

Compression

In general, there are two approaches to reduce the data size of image-based representations. The first one is to reduce their dimensionality, often by limiting viewpoints or sacrificing some realism. Panoramas and concentric mosaics are such examples. The second approach is to exploit the high correlation

**Image-Based
Rendering, Fig. 5**
Geometry-rendering matrix



(i.e., redundancy) within the representation using waveform coding or model-based techniques. The scene geometry may be used explicitly or implicitly. The second approach can be further classified into four broad categories: pixel-based methods, disparity compensation/prediction (DCP) methods, model-based/model-aided methods, and object-based approach.

In pixel-based methods, the correlation between adjacent image pixels is exploited using conventional techniques such as vector quantization and transform coding. In the DCP methods, scene geometry is utilized implicitly by exploiting the disparity of image pixels, resulting in better compression performance. (Disparity refers to the relative displacement of pixels in images taken at adjacent physical locations.) Model-based/model-aided approaches recover the geometry of the objects or scene in coding the

observed images. The models and other information such as prediction residuals or view-dependent texture maps are then encoded. In the object-based approach, the representations are segmented into IBR objects, each with its image sequences, depth maps, and other relevant information such as shape information. The main advantage is that it helps to reduce the rendering artifacts and hence the required sampling rate. For additional references, see the section on light fields.

Unlike conventional video coding, higher dimensional image-based representations such as 3D concentric mosaics (CMs) require random access at the line level, whereas the 4D light field and lumigraph require random access at the pixel level. It is usually time-consuming to retrieve and decode a single line or pixel from the compressed which is of variable length due to entropy coding. This is referred to as the “random access problem” of IBR and is usually tackled

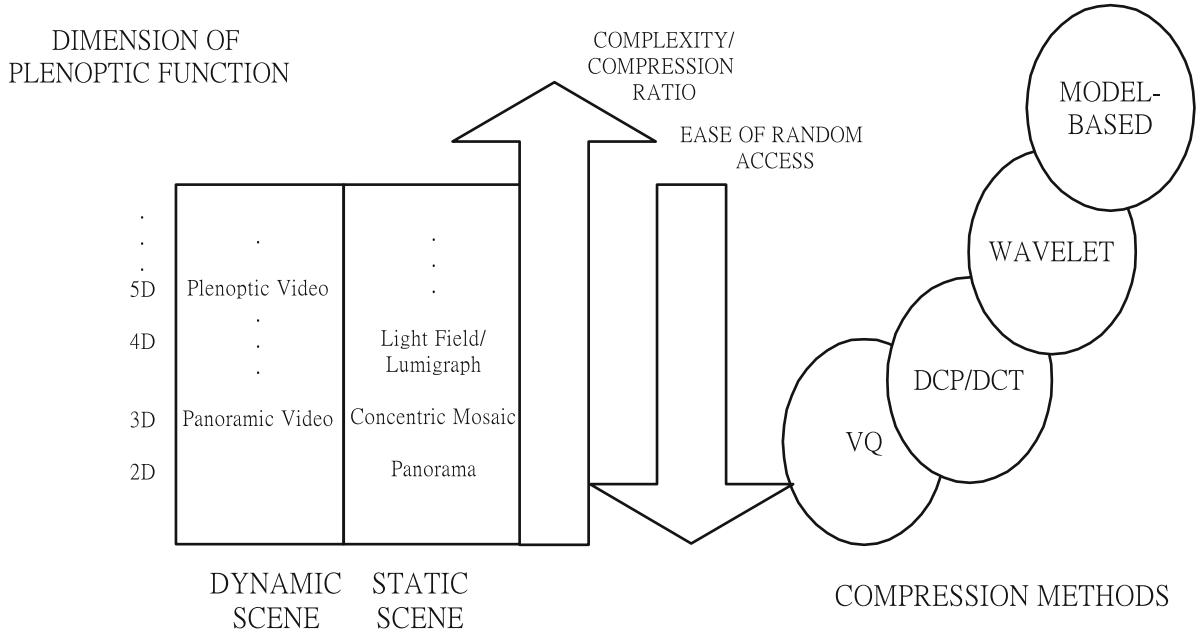


Image-Based Rendering, Fig. 6 Comparison of different image-based representation and compression methods in terms of their complexity. The ease of random access increases as the

dimension of plenoptic function decreases, while the complexity and potential for compression both increase with the dimension. *DCP* disparity compensation/prediction, *VQ* vector quantization

by grouping the compressed data of several basic units for rendering (such as lines in CMs or image blocks in light fields) together and employ pointers to locate them efficiently. Moreover, interdependence of decoding resulting from DCP should be reduced to avoid decoding excessively unnecessary intermediate data. This is also required for selective transmission or decoding of the compressed representations due to their large bandwidth and storage requirement. A simple comparison of difficult image-based representations and compression methods in terms of their complexities, compression ratios, and ease of random access is shown in Fig. 6. For more information, see [15, 17] and references in the light field section.

Application

The potential for photorealistic visualization and simplicity in rendering of IBR has tremendous appeal. They have already found applications in architectural modeling [6], cultural heritage preservation [23], virtual tour, and digital museum [18], multiview TV [3, 4], etc. Other potential applications include digital edutainment, E-commerce and photorealistic

modeling, and real-time rendering in computer graphics and mobile devices. Another emerging application is view synthesis in 3D and multiview videos and display.

Open Problems

Though there has been substantial progress in capturing, representing, rendering, and modeling of scenes, the ability to handle general complex scenes remains challenging for IBR. A substantial amount of work is still required to ensure robustness in handling reflection translucency, highlights, depth estimation, capturing complexity, object manipulation, etc. Since IBR uses images for rendering, interacting with IBR representations remains challenging. Recent approaches have focused on using advanced computer vision techniques, such as stereo/multiview vision and photometric stereo, and depth sensing devices to extract more geometry information from the scene so as to enhance the functionalities of IBR representations. While there has been considerable progress in relighting and interactive rendering of individual real static objects, such

operations are still difficult for real and complicated scenes. For dynamic scenes, the huge amount of data and vast amount of viewpoints to be provided present one of the major challenges to IBR. Advanced algorithms for processing and manipulation of the high-dimensional representation to achieve such function as object extraction, model completion, scene inpainting, etc., are all major challenges to be addressed. Finally, the efficient transmission, compression, and display of dynamic IBR and models are also urgent issues awaiting for satisfactory solution in order for IBR to establish itself as an essential media for communication and presentation.

References

1. Adelson EH, Bergen JR (1991) The plenoptic function and the elements of early vision. In: Landy M, Movshon JA (eds) Computational models of visual processing. MIT, Cambridge, MA
2. Agrawala M, Ramamoorthi R, Heirich A, Moll L (2000) Efficient image-based methods for rendering soft shadows. In: Proceedings of ACM SIGGRAPH, New Orleans, 372–384
3. Chan SC, Shum HY, Ng KT (2007) Image-based rendering and synthesis: technological advances and challenges. IEEE Signal Process Mag 24(6):22–33
4. Chan SC, Gan ZF, Ng KT, Ho KL, Shum HY (2009) An object-based approach to image/video-based synthesis and processing for 3-D and multiview televisions. IEEE Trans. Circuits Syst Video Technol 19(6):821–831
5. Chen SE (1995) QuickTime VR – an image-based approach to virtual environment navigation. In: Proceedings of ACM SIGGRAPH, Los Angeles, 29–38
6. Debevec PE, Taylor CJ, Malik J (1996) Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. In: Proceedings of ACM SIGGRAPH, New Orleans, 11–20
7. Debevec PE, Yu Y, Borshukov G (1998) Efficient view-dependent image-based rendering with projective texture-mapping. In: Eurographics workshop on rendering, Vienna, 150–116
8. Gortler SJ, Grzeszczuk R, Szeliski R, Cohen MF (1996) The lumigraph. In: Proceedings of ACM SIGGRAPH, New Orleans, 43–54
9. Levoy M, Hanrahan P (1996) Light field rendering. In: Proceedings of ACM SIGGRAPH, New Orleans, 31–42
10. Lhuillier M, Quan L (2003) Image-based rendering by joint view triangulation. IEEE Trans Circuits Syst Video Technol 13(11):1051–1063
11. McMillan L, Bishop G (1995) Plenoptic modeling: an image-based rendering system. In: Proceedings of ACM SIGGRAPH, Los Angeles, 39–46
12. Ng R, Ramamoorthi R, Hanrahan P (2004) Triple product wavelet integrals for all-frequency relighting. In: Proceedings of ACM SIGGRAPH, Los Angeles, 477–487
13. Shade J, Gortler S, He LW, Szeliski R (1998) Layered depth images. In: Proceedings of ACM SIGGRAPH, Orlando, 231–242
14. Shum HY, He LW (1999) Rendering with concentric mosaics. In: Proceedings of ACM SIGGRAPH, Los Angeles, 299–306
15. Shum HY, Kang SB, Chan SC (2003) Survey of image-based representations and compression techniques. IEEE Trans Circuits Syst Video Technol 13(11):1020–1037
16. Shum HY, Sun J, Yamazaki S, Li Y, Tang CK (2004) Pop-up light field: an interactive image-based modeling and rendering system. ACM Trans Graph 23(2):143–162
17. Shum HY, Chan SC, Kang SB (2007) Image-based rendering. Springer, New York
18. Snavely N, Simon I, Goesele M, Szeliski R, Seitz SM (2010) Scene reconstruction and visualization from community photo collections. Proc IEEE Internet Vis 98(8): 1370–1390
19. Sloan P, Kautz J, Snyder J (2002) Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environment. In: Proceedings of ACM SIGGRAPH, San Antonio, 527–536
20. Szeliski R, Shum HY (1997) Creating full view panoramic image mosaics and environment maps. In: Proceedings of ACM SIGGRAPH, Los Angeles, 251–258
21. Wood DN, Azuma DI, Aldinger K, Curless B, Duchamp T, Salesin DH, Stuetzle W (2000) Surface light fields for 3D photography. In: Proceedings of ACM SIGGRAPH, New Orleans, 287–296
22. Zhou K, Hu Y, Lin S, Guo B, Shum HY (2005) Precomputed shadow fields for dynamic scenes. In: Proceedings of ACM SIGGRAPH, Los Angeles, 1196–1201
23. Zu ZY, Ng KT, Chan SC, Shum HY (2010) Image-based rendering of ancient Chinese artifacts for multi-viewdisplays – a multi-camera approach. In: Proceedings of 2010 IEEE international symposium on circuits and systems (ISCAS), Paris, 3252–3255

Image-Based Rendering (IBR)

- [Image-Based Rendering](#)
- [Light Field](#)
- [Plenoptic Function](#)

Implicit Polynomial Curve

- [Algebraic Curve](#)

Implicit Polynomial Surface

- [Algebraic Surface](#)

Incident Light Measurement

Stephen Lin
Microsoft Research Asia, Beijing Sigma Center,
Beijing, China

Related Concepts

► [Illumination Estimation, Illuminant Estimation](#)

Definition

Incident light measurement is the recording of incoming illumination at a given scene point or in a given scene.

Background

The distribution and intensity of light incident upon a surface point or in a scene affects the amount of reflected radiance to the camera and more generally the appearance of objects. Knowledge of the incident light can aid in shape recovery through photometric analysis techniques such as shape-from-shading and photometric stereo, or may be used in reducing appearance variation caused by lighting. Various methods have been used for measurement of incident light. Different from algorithms for illumination estimation, incident light measurement does not infer lighting from indirect scene cues such as shading, but rather obtains direct observations of the light sources.

Methods for incident light measurement typically introduce a probe or a sensor into the scene to view the incoming radiance. In general, the probes are mirrored spheres that allow for precise readings of light from a broad range of incident directions. Besides lighting distribution, the color of incident illumination may be measured using a color calibration target such as a white reference standard. Unlike illumination estimation methods, light measurement with such devices often allows for accurate recovery of both direct illumination from light sources and more subtle indirect illumination from reflected light within the scene.

Some techniques are intended to measure incident light at a certain scene point. Excluding the effects of light occluders, these methods equivalently measure

far lighting that originates from distant light sources and is considered to be uniform throughout the scene. Other methods are more general in that their measurements also determine the location and brightness of near light sources, whose illumination varies within the scene. Such methods utilize triangulation, usually from two or more probes or sensors placed in the scene, to locate the positions of local light sources.

Methods

Several methods for incident light measurement are described in the following.

Spherical Probes

To measure distant illumination or the light incident at a given scene point, a common approach is to place a mirrored spherical probe at the scene point. From the reflections on the sphere, the corresponding directions of the incident light are computed from the known surface orientation of each sphere point and the mirror reflection property, which states that the incident angle of light is equal to the reflected light angle. Incident lighting environments of various scenes were measured in this manner by Debevec [1]. High dynamic range imaging was used to obtain accurate measurements of relative light source brightness.

To recover spatially variant incident lighting due to local light sources, Powell et al. [2] used three mirrored spheres at known relative positions to triangulate light source locations. For triangulation, correspondences need to be computed among the mirrored reflections of the spheres. Illumination color is also measured from the color of diffuse reflections on the spheres. Zhou and Kambhamettu [3] also employed triangulation, but instead computed correspondences in a stereo image pair of a single sphere. Shifts in specular reflections as seen from the two stereo viewpoints indicate the distance of light sources. Here, the spheres also exhibit diffuse reflection, which provides information on light intensities. Using this setup, they later proposed a method [4] based on ray tracing and convex hull computation to measure a more general light source model [5].

Hemispherical Imaging

An alternative to lighting probes is to directly place sensors within the scene. Drettakis et al. [6] employed

image mosaicing of several snapshots captured within the scene to form a panoramic image of the incident lighting. Sato et al. [7] instead used a pair of omnidirectional cameras, each outfitted with a fish-eye lens. Correspondences in the omnidirectional images are computed with an omnidirectional stereo algorithm to obtain a 3D model of the incident lighting, and high dynamic range imaging is used to measure the intensity of radiance.

Color Calibration Target

The incident light color may be measured by inserting a white reference standard into the scene. Deviations from white of the reflected light indicate the color of illumination. This approach to measuring incident lighting color is described by Barnard et al. [8] for their construction of an image dataset for computational color constancy. Directional variations in illumination color may be measured by imaging the white reference standard at different orientations.

Application

Incident light measurement is often employed for augmented reality [1, 7], to ensure that inserted virtual objects exhibit an appearance consistent with the scene's illumination environment. Measurements of real-world lighting have also been utilized in computer graphics applications to give rendered objects a more natural appearance. Applications of light color measurement include evaluation of color constancy algorithms [8] and spectral reflectance recovery using multiple illumination colors [9].

References

1. Debevec P (1998) Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. In: Proceedings of ACM SIGGRAPH. ACM, New York, pp 189–198
 2. Powell MW, Sarkar S, Goldgof D (2001) A simple strategy for calibrating the geometry of light sources. IEEE Trans Pattern Anal Mach Intell 23:1022–1027
 3. Zhou W, Kambhamettu C (2002) Estimation of illuminant direction and intensity of multiple light sources. In: Proceedings of European conference on computer vision (ECCV). Lecture notes in computer science, vol 2353. Springer, Berlin/Heidelberg, pp 206–220
 4. Zhou W, Kambhamettu C (2008) A unified framework for scene illuminant estimation. Image Vis Comput 26:415–429
 5. Langer MS, Zucker SW (1997) What is a light source? In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Computer Society, Washington, DC, pp 172–178
 6. Drettakis G, Robert L, Bougnoux S (1997) Interactive common illumination for computer augmented reality. In: Proceedings of the Eurographics workshop on rendering. Springer, London, pp 45–57
 7. Sato I, Sato Y, Ikeuchi K (1999) Acquiring a radiance distribution to superimpose virtual objects onto a real scene. IEEE Trans Vis Comput Graph 5:1–12
 8. Barnard K, Martin L, Funt B, Coath A (2002) A data set for colour research. Color Res Appl 27:147–151
 9. Han S, Sato I, Okabe T, Sato Y (2010) Fast spectral reflectance recovery using dlp projector. In: Proceedings of Asian conference on computer vision. Springer, Berlin
-

Inexact Matching

► Many-to-Many Graph Matching

Information Fusion

► Data Fusion

Inherent Optical Properties

► Underwater Effects

Inpainting

Marcelo Bertalmío¹, Vicent Caselles¹, Simon Masnou² and Guillermo Sapiro³

¹Universitat Pompeu Fabra, Barcelona, Spain

²Institut Camille Jordan, Université Lyon 1, Villeurbanne, France

³Electrical and Computer Engineering, Computer Science, and Biomedical Engineering, Duke University, Durham, NC, USA

Synonyms

Disocclusion; Error concealment; Filling in

This contribution is dedicated to the memory of Vicent Caselles, outstanding researcher, exceptional friend.

Definition

Given an image and a region Ω inside it, the inpainting problem consists in modifying the image values of the pixels in Ω so that this region does not stand out with respect to its surroundings. The purpose of inpainting might be to restore damaged portions of an image (e.g., an old photograph where folds and scratches have left image gaps) or to remove unwanted elements present in the image (e.g., a microphone appearing in a film frame). See Fig. 1. The region Ω is always given by the user, so the localization of Ω is not part of the inpainting problem. Almost all inpainting algorithms treat Ω as a hard constraint, whereas some methods allow some relaxing of the boundaries of Ω .

This definition, given for a single-image problem, extends naturally to the multi-image case; therefore, this entry covers both image and video inpainting. What is not however considered in this text is *surface* inpainting (e.g., how to fill holes in 3D scans), although this problem has been addressed in the literature.

Background

The term *inpainting* comes from art restoration, where it is also called *retouching*. Medieval artwork started to be restored as early as the Renaissance, the motives being often as much to bring medieval pictures “up to date” as to fill in any gaps. The need to retouch the image in an unobtrusive way extended naturally from paintings to photography and film. The purposes remained the same: to revert deterioration (e.g., scratches and dust spots in film) or to add or remove elements (e.g., the infamous “airbrushing” of political enemies in Stalin era USSR). In the digital domain, the inpainting problem first appeared under the name “error concealment” in telecommunications, where the need was to fill in image blocks that had been lost during data transmission. One of the first works to address automatic inpainting in a general setting dubbed it “image disocclusion” since it treated the image gap as an occluding object that had to be removed, and the image underneath would be the restoration result. Popular terms used to denote inpainting algorithms are also “image completion” and “image fill-in.”

Application

The extensive literature on digital image inpainting may be roughly grouped into three categories: patch-based, sparse, and PDEs/variational methods.

From Texture Synthesis to Patch-Based Inpainting

Efros and Leung [14] proposed a method that, although initially intended for texture synthesis, has proven most effective for the inpainting problem. The image gap is filled in recursively, inwards from the gap boundary: each “empty” pixel P at the boundary is filled with the value of the pixel Q (lying outside the image gap, that is, Q is a pixel with valid information) such that the neighborhood $\Psi(Q)$ of Q (a square patch centered in Q) is most similar to the (available) neighborhood $\Psi(P)$ of P . Formally, this can be expressed as an optimization problem:

$$\begin{aligned} \text{Output}(P) &= \text{Value}(Q), \quad P \in \Omega, \quad Q \notin \Omega, \\ Q &= \arg \min d(\Psi(P), \Psi(Q)), \end{aligned} \quad (1)$$

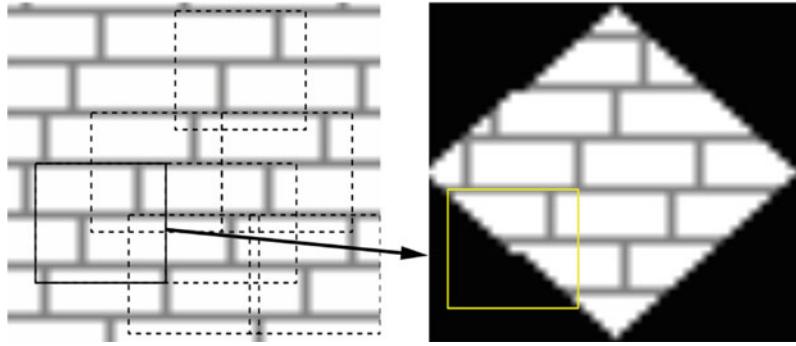
where $d(\Psi(P), \Psi(Q))$ is the sum of squared differences (SSD) among the patches $\Psi(P)$ and $\Psi(Q)$ (considering only available pixels):

$$d(\Psi_1, \Psi_2) = \sum_i \sum_j |\Psi_1(i, j) - \Psi_2(i, j)|^2, \quad (2)$$

and the indices i, j span the extent of the patches (e.g., if Ψ is an 11×11 patch, then $0 \leq i, j \leq 10$). Once P is filled in, the algorithm marches on to the next pixel at the boundary of the gap, never going back to P (whose value is, therefore, not altered again). See Fig. 2 for an overview of the algorithm and Fig. 3 for an example of the outputs it can achieve. The results are really impressive for a wide range of images. The main shortcomings of this algorithm are its computational cost, the selection of the neighborhood size (which in the original paper is a global user-selected parameter but which should change locally, depending on image content), the filling order (which may create unconnected boundaries for some objects), and the fact that it cannot deal well with image perspective (it was intended to synthesize frontal textures; hence, neighborhoods are compared always with the same size and orientation). Also, results are poor if the image gap is very large and



Inpainting, Fig. 1 The inpainting problem. *Left*: original image. *Middle*: inpainting mask Ω , in black. *Right*: an inpainting result (Figure taken from [20])



Inpainting, Fig. 2 Efros and Leung's algorithm overview (figure taken from [14]). Given a sample texture image (*left*), a new image is being synthesized one pixel at a time (*right*). To synthesize a pixel, the algorithm first finds all neighborhoods in

the sample image (boxes on the *left*) that are similar to the pixels neighborhood (box on the *right*) and then randomly chooses one neighborhood and takes its *center* to be the newly synthesized pixel

disperse (e.g., an image where 80 % of the pixels have been lost due to random *salt and pepper noise*).

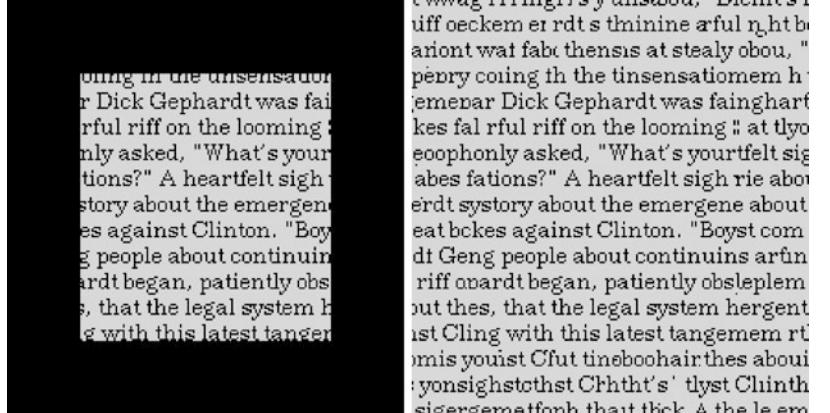
Criminisi et al. [12] improved on this work in two aspects. Firstly, they changed the filling order from the original “onion-peel” fashion to a priority scheme where empty pixels at the edge of an image object have higher priority than empty pixels on flat regions. Thus, they are able to correctly inpaint straight object boundaries which could have otherwise ended up disconnected with the original formulation. See Fig. 4. Secondly, they copy entire patches instead of single pixels, so this method is considerably faster. Several shortcomings remain, though, like the inability to deal with perspective and the need to manually select the neighborhood size (here, there are two sizes to set, one for the patch to compare with and another for the patch to copy from). Also, objects with curved boundaries may not be inpainted correctly.

Ashikhmin [2] contributed as well to improve on the original method of Efros and Leung [14]. With the idea of reducing the computational cost of the

procedure, he proposed to look for the best candidate Q to copy its value to the empty pixel P not searching the whole image but only searching among the candidates of the neighbors of P which have already been inpainted. See Fig. 5. The speedup achieved with this simple technique is considerable, and also there is a very positive effect regarding the visual quality of the output. Other methods reduce the search space and computational cost involved in the candidate patch search by organizing image patches in tree structures, reducing the dimensionality of the patches with techniques like principal component analysis (PCA), or using randomized approaches.

While most image inpainting methods attempt to be fully automatic (aside from the manual setting of some parameters), there are user-assisted methods that provide remarkable results with just a little input from the user. In the work by Sun et al. [27], the user must specify curves in the unknown region, curves corresponding to relevant object boundaries. Patch synthesis is performed along these curves inside the image gap,

Inpainting, Fig. 3 *Left:* original image, inpainting mask Ω in black. *Right:* inpainting result obtained with Efros and Leung's algorithm, images taken from their paper [14]



Inpainting, Fig. 4 *Left:* original image. *Right:* inpainting result obtained with the algorithm of Criminisi et al. [12], images taken from their paper



by copying from patches that lie on the segments of these curves which are outside the gap, in the “known” region. Once these curves are completed, in a process which the authors call *structure propagation*, the remaining empty pixels are inpainted using a technique like the one by Ashikhmin [2] with priorities as in Criminisi et al. [12]. Barnes et al. [5] accelerate this method and make it interactive, by employing randomized searches and combining into one step the structure propagation and texture synthesis processes of Sun et al. [27].

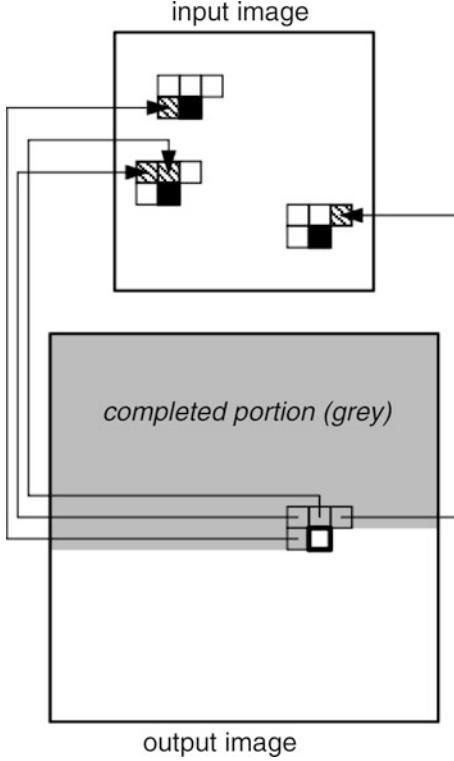
The Role of Sparsity

After the introduction of patch-based methods for texture synthesis by Efros and Leung [14], and image inpainting by Criminisi et al. [12], it became clear

that the patches of an image provide a good dictionary to express other parts of the image. This idea has been successfully applied to other areas of image processing, for example, denoising and segmentation.

More general sparse image representations using dictionaries have proven their efficiency in the context of inpainting. For instance, using overcomplete dictionaries adapted to the representation of image geometry and texture, Elad et al. [15] proposed an image decomposition model with sparse coefficients for the geometry and texture components of the image and showed that the model can be easily adapted for image inpainting. A further description of this model follows.

Let u be an image represented as a vector in \mathbb{R}^N . Let the matrices D_g and D_t of sizes $N \times k_g$ and



Inpainting, Fig. 5 Ashikhmin’s texture synthesis method (figure taken from [2]). Each pixel in the current L-shaped neighborhood generates a shifted candidate pixel (*black*) according to its original position (*hatched*) in the input texture. The best pixel is chosen among these candidates only. Several different pixels in the current neighborhood can generate the same candidate

$N \times k_t$ represent two dictionaries adapted to geometry and texture, respectively. If $\alpha_g \in \mathbb{R}^{k_g}$ and $\alpha_t \in \mathbb{R}^{k_t}$ represent the geometry and texture coefficients, then $u = D_g \alpha_g + D_t \alpha_t$ represents the image decomposition using the dictionaries collected in D_g and D_t . A sparse image representation is obtained by minimizing

$$\min_{(\alpha_g, \alpha_t): u = D_g \alpha_g + D_t \alpha_t} \|\alpha_g\|_p + \|\alpha_t\|_p, \quad (3)$$

where $p = 0, 1$. Although the case $p = 0$ represents the sparseness measure (i.e., the number of nonzero coordinates), it leads to a nonconvex optimization problem whose minimization is more complex. The case $p = 1$ yields a convex and tractable optimization problem leading also to sparseness. Introducing the constraint by penalization (thus, in practice, relaxing it) and regularizing the geometric part of the

decomposition with a total variation semi-norm penalization, Elad et al. [15] propose the variational model:

$$\begin{aligned} \min_{(\alpha_g, \alpha_t)} & \|\alpha_g\|_1 + \|\alpha_t\|_1 + \lambda \|u - D_g \alpha_g - D_t \alpha_t\|_2^2 \\ & + \gamma TV(D_g \alpha_g), \end{aligned} \quad (4)$$

where TV denotes the total variation, $\lambda, \gamma > 0$. This model can be easily adapted to a model for image inpainting. Observe that $u - D_g \alpha_g - D_t \alpha_t$ can be interpreted as the noise component of the image and λ is a penalization parameter that depends inversely on the noise power. Then the inpainting mask can be interpreted as a region where the noise is very large (infinite). Thus, if $M = 0$ and $= 1$ identify the inpainting mask and the known part of the image, respectively, then the extension of (4) to inpainting can be written as

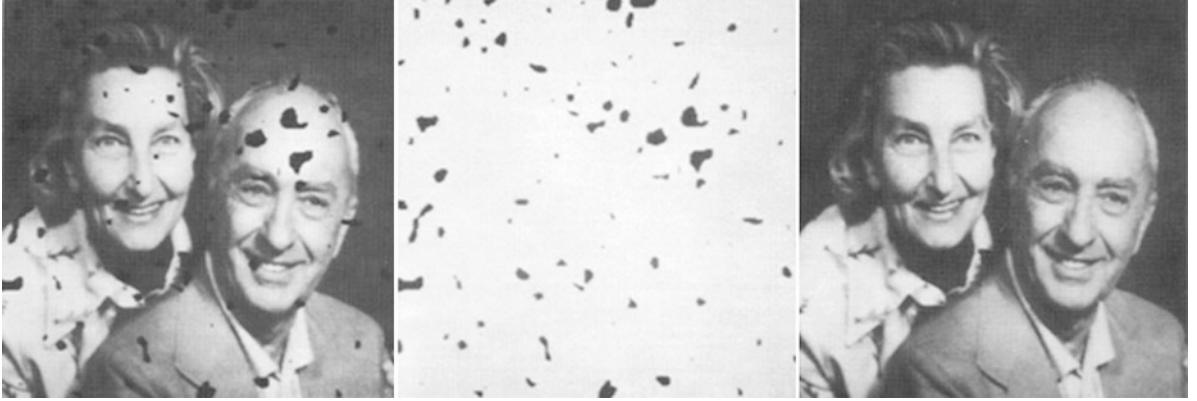
$$\begin{aligned} \min_{(\alpha_g, \alpha_t)} & \|\alpha_g\|_1 + \|\alpha_t\|_1 + \lambda \|M(u - D_g \alpha_g - D_t \alpha_t)\|_2^2 \\ & + \gamma TV(D_g \alpha_g). \end{aligned} \quad (5)$$

Writing the energy in (5) using $u_g := D_g u$, $u_t := D_t u$ as unknown variables, it can be observed that $\alpha_g = D_g^+ u_g + r_g$, $\alpha_t = D_t^+ u_t + r_t$, where D_g^+ , D_t^+ denote the corresponding pseudoinverse matrices and r_g, r_t are in the null spaces of D_g and D_t , respectively. Assuming for simplicity, as in Elad et al. [15], that $r_g = 0, r_t = 0$, the model (5) can be written as

$$\begin{aligned} \min_{(\alpha_g, \alpha_t)} & \|D_g^+ u_g\|_1 + \|D_t^+ u_t\|_1 \\ & + \lambda \|M(u - u_g - u_t)\|_2^2 + \gamma TV(u_g). \end{aligned} \quad (6)$$

This simplified model is justified in Elad et al. [15] by several reasons: it is an upper bound for (5), it is easier to solve, it provides good results, it has a Bayesian interpretation, and it is equivalent to (5) if D_g and D_t are non-singular or when using the ℓ^2 norm in place of the ℓ^1 norm. The model has nice features since it permits to use adapted dictionaries for geometry and texture and treats the inpainting as missing samples, and the sparsity model is included with ℓ^1 norms that are easy to solve.

This framework has been adapted to the use of dictionaries of patches and has been extended in several directions like image denoising, filling in missing pixels (Aharon et al. [1]), color image denoising, demosaicing, and inpainting of small holes (Mairal et al. [21]



Inpainting, Fig. 6 An inpainting experiment taken from Ogden et al. [24]. The method uses a Gaussian pyramid and a series of linear interpolations, downsampling, and upsampling

and further extended to deal with multiscale dictionaries and to cover the case of video sequences in Mairal et al. [22]. To give a brief review of this model, some notation is required. Image patches are squares of size $n = \sqrt{n} \times \sqrt{n}$. Let D be a dictionary of patches represented by a matrix of size $n \times k$, where the elements of the dictionary are the columns of D . If $\alpha \in \mathbb{R}^k$ is a vector of coefficients, then $D\alpha$ represents the patch obtained by linear combination of the columns of D . Given an image $v(i, j)$, $i, j \in \{1, \dots, N\}$, the purpose is to find a dictionary \hat{D} , an image \hat{u} , and coefficients $\hat{\alpha} = \{\hat{\alpha}_{i,j} \in \mathbb{R}^k : i, j \in \{1, \dots, N\}\}$ which minimize the energy

$$\begin{aligned} \min_{(\alpha, D, u)} & \lambda \|v - u\|_2 + \sum_{i,j=1}^N \mu_{i,j} \|\alpha_{i,j}\|_0 \\ & + \sum_{i,j=1}^N \|D\alpha_{i,j} - R_{i,j}u\|_2, \end{aligned} \quad (7)$$

where $R_{i,j}u$ denotes the patch of u centered at (i, j) (dismissing boundary effects), and $\mu_{i,j}$ are positive weights. The solution of the nonconvex problem (7) is obtained using an alternate minimization: a sparse coding step where one computes $\alpha_{i,j}$ knowing the dictionary D for all i, j , a dictionary update using a sequence of one rank approximation problem to update each column of D (Aharon et al. [1]), and a final reconstruction step given by the solution of

$$\min_u \lambda \|v - u\|_2 + \sum_{i,j=1}^N \|\hat{D}\alpha_{i,j} - R_{i,j}u\|_2. \quad (8)$$

Again, the inpainting problem can be considered as a case of nonhomogeneous noise. Defining for each pixel (i, j) a coefficient $\beta_{i,j}$ inversely proportional to the noise variance, a value of $\beta_{i,j} = 0$ may be taken for each pixel in the inpainting mask. Then the inpainting problem can be formulated as

$$\begin{aligned} \min_{(\alpha, D, u)} & \lambda \|\beta \otimes (v - u)\|_2 + \sum_{i,j=1}^N \mu_{i,j} \|\alpha_{i,j}\|_0 \\ & + \sum_{i,j=1}^N \|(R_{i,j}\beta) \otimes (D\alpha_{i,j} - R_{i,j}u)\|_2, \end{aligned} \quad (9)$$

where $\beta = (\beta_{i,j})_{i,j=1}^N$ and \otimes denotes the elementwise multiplication between two vectors.

With suitable adaptations, this model has been applied to inpainting (of relatively small holes), to interpolation from sparse irregular samples and super-resolution, to image denoising, to demosaicing of color images, and to video denoising and inpainting, obtaining excellent results; see Mairal et al. [22].

PDEs and Variational Approaches

All the methods mentioned so far are based on the same principle: a missing/corrupted part of an image



Inpainting, Fig. 7 Amodal completion: the visual system automatically completes the broken edge in the *left* figure. The *middle* figure illustrates that, here, no global symmetry process is involved: in both figures, the same edge is synthesized. In such simple situation, the interpolated curve can be modeled as Euler's elastica, that is, a curve with clamped points and tangents at its extremities and with minimal oscillations

can be well synthetized by suitably sampling and copying uncorrupted patches (taken either from the image itself or built from a dictionary). A very different point of view underlies many contributions involving either a variational principle, through a minimization process, or a (non necessarily variational) partial differential equation (PDE).

An early interpolation method that applies for inpainting is due to Ogden et al. [24]. Starting from an initial image, a Gaussian filtering is built by iterated convolution and subsampling. Then, a given inpainting domain can be filled in by successive linear interpolations, downsampling, and upsampling at different levels of the Gaussian pyramid. The efficiency of such approach is illustrated in Fig. 6.

Masnou and Morel proposed in [23] to interpolate a gray-valued image by extending its isophotes (the lines of constant intensity) in the inpainting domain. This approach is very much in the spirit of early works by Kanizsa, Ullman, Horn, Mumford, and Nitzberg to model the ability of the visual system to complete edges in an occlusion or visual illusion context. This is illustrated in Fig. 7. The general completion process involves complicated phenomena that cannot be easily and univocally modeled. However, experimental results show that, in simple occlusion situations, it is reasonable to argue that the brain extrapolates broken edges using elastica-type curves, that is, curves that join two given points with prescribed tangents at these points, a total length lower than a given L , and minimize the Euler elastica energy $\int |\kappa(s)|^2 ds$, with s the curve arc length and κ the curvature.

The model by Masnou and Morel [23] generalizes this principle to the isophotes of a gray-valued image. More precisely, denoting $\tilde{\Omega}$ a domain slightly larger than Ω , it is proposed in [23] to extrapolate the isophotes of an image u , known outside Ω and valued

in $[m, M]$, by a collection of curves $\{\gamma_t\}_{t \in [m, M]}$ with no mutual crossings, that coincide with the isophotes of u on $\tilde{\Omega} \setminus \Omega$ and that minimize the energy

$$\int_m^M \int_{\gamma_t} (\alpha + \beta |\kappa_{\gamma_t}|^p) ds dt. \quad (10)$$

Here α, β are two context-dependent parameters. This energy penalizes a generalized Euler's elastica energy, with curvature to the power $p > 1$ instead of 2, of all extrapolation curves γ_t , $t \in [m, M]$.

An inpainting algorithm, based on the minimization of (10) in the case $p = 1$, is proposed by Masnou and Morel in [23]. A globally minimal solution is computed using a dynamic programming approach that reduces the algorithmical complexity. The algorithm handles only simply connected domains, that is, those with no holes. In order to deal with color images, RGB images are turned into a luma/chrominance representation, for example, YCrCb, or Lab, and each channel is processed independently. The reconstruction process is illustrated in Fig. 8.

The word *inpainting*, in the image processing context, has been coined first by Bertalmío, Sapiro, Caselles, and Ballester in [6], where a PDE model is proposed in the very spirit of real paintings restoration. More precisely, u being a gray-valued image to be inpainted in Ω , a time-stepping method for the transport-like equation

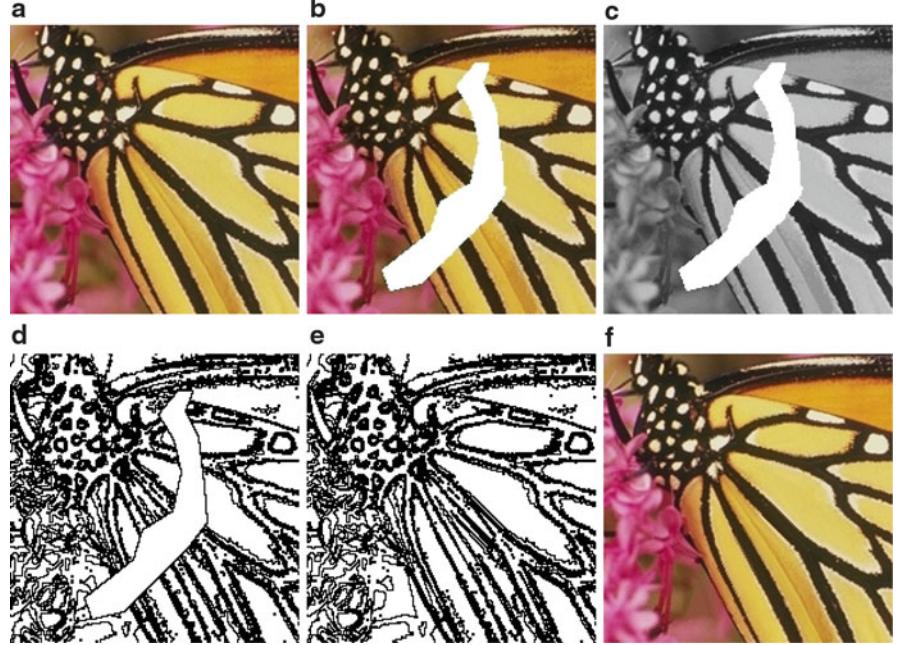
$$\begin{aligned} u_t &= \nabla^\perp u \cdot \nabla \Delta u \quad \text{in } \Omega, \\ u &\quad \text{given in } \Omega^c \end{aligned} \quad (11)$$

is combined with anisotropic diffusion steps that are interleaved for stabilization, using the following diffusion model:

$$u_t = \varphi_\epsilon(x) |\nabla u| \nabla \cdot \frac{\nabla u}{|\nabla u|}, \quad (12)$$

where φ_ϵ is a smooth cutoff function that forces the equation to act only in Ω , and $\nabla \cdot (\nabla u / |\nabla u|)$ is the curvature along isophotes. This diffusion equation, which has been widely used for denoising an image while preserving its edges, compensates any shock possibly created by the transport-like equation. What is the meaning of Eq. (11)? Following Bertalmío et al. [6], Δu is a measure of image smoothness, and stationary points for the equation are images for which Δu

Inpainting, Fig. 8 (a) is the original image and (b) the image with occlusions in white. The luminance channel is shown in figure (c). A few isophotes are drawn in figure (d), and their reconstruction by the algorithm of Masnou and Morel [23] is given in figure (e). Applying the same method to the luminance, hue, and saturation channels yields the final result of figure (f)



is constant along the isophotes induced by the vector field $\nabla^\perp u$. Equation (11) is not explicitly a transport equation for Δu , but, in the equivalent form,

$$u_t = -\nabla^\perp \Delta u \cdot \nabla u, \quad (13)$$

it is a transport equation for u being convected by the field $\nabla^\perp \Delta u$. Following Bornemann and März [9], this field is in the direction of the level lines of Δu , which are related to the Marr-Hildreth edges. Indeed, the zero crossings of (a convoluted version of) Δu are the classical characterization of edges in the celebrated model of Marr and Hildreth. In other words, as in the real paintings restoration, the approach of Bertalmío et al. [6] consists in conveying the image intensities along the direction of the edges, from the boundary of the inpainting domain Ω toward the interior. The efficiency of such approach is illustrated in Fig. 9. From a numerical viewpoint, the transport equation and the anisotropic diffusion can be implemented with classical finite difference schemes. For color images, the coupled system can be applied independently to each channel of any classical luma/chrominance representation. There is no restriction on the topology of the inpainting domain.

Another perspective on this model is provided by Bertalmío, Bertozzi, and Sapiro in [7], where connections with the classical Navier-Stokes equation

of fluid dynamics are shown. Indeed, the steady-state equation of Bertalmío et al. [6],

$$\nabla^\perp u \cdot \nabla \Delta u = 0,$$

is exactly the equation satisfied by steady-state inviscid flows in the two-dimensional incompressible Navier-Stokes model. Although the anisotropic diffusion equation (12) is not the exact counterpart of the viscous diffusion term used in the Navier-Stokes model for incompressible and Newtonian flows, a lot of the numerical knowledge on fluid mechanics seems to be adaptable to design stable and efficient schemes for inpainting. Results in this direction are shown in Bertalmío et al. [7].

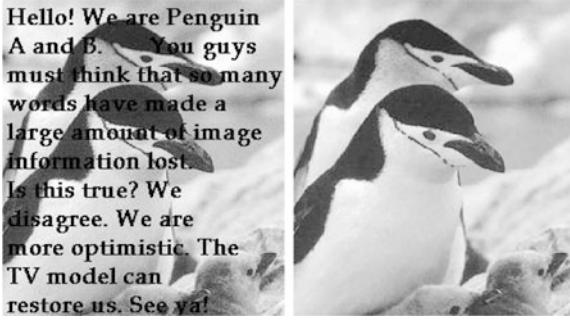
Chan and Shen propose in [10] a denoising/inpainting first-order model based on the joint minimization of a quadratic fidelity term outside Ω and a total variation criterion in Ω , that is, the joint energy

$$\int_A |\nabla u| dx + \frac{\lambda}{2} \int_\Omega |u - u_0|^2 dx,$$

with $A \supset \Omega$ the image domain and λ a Lagrange multiplier. The existence of solutions to this problem follows easily from the properties of functions of bounded variation. As for the implementation, Chan and Shen look for critical points of the energy



Inpainting, Fig. 9 An experiment taken from Bertalmío et al. [6]. *Left*: original image. *Middle*: a user-defined mask. *Right*: the result with the algorithm of [6]



Inpainting, Fig. 10 An experiment taken from Chan and Shen [10]. *Left*: original image. *Right*: after denoising and removal of text

using a Gauss-Jacobi iteration scheme for the linear system associated to an approximation of the Euler-Lagrange equation by finite differences. More recent approaches to the minimization of total variation with subpixel accuracy should nowadays be preferred. From the phenomenological point of view, the model of Chan and Shen [10] yields inpainting candidates with the smallest possible isophotes. It is therefore more suitable for thin or sparse domains. An illustration of the model's performances is given in Fig. 10.

Turning back to the criterion (10), a similar penalization on $\tilde{\Omega}$ of both the length and the curvature of all isophotes of an image u yields two equivalent forms, in the case where u is smooth enough (see Masnou and Morel [23]):

$$\int_{-\infty}^{+\infty} \int_{\{u=t\} \cap \tilde{\Omega}} (\alpha + \beta |\kappa|^p) ds dt = \int_{\tilde{\Omega}} |\nabla u| \left(\alpha + \beta \left| \nabla \cdot \frac{\nabla u}{|\nabla u|} \right|^p \right) dx. \quad (14)$$

There have been various contributions to the numerical approximation of critical points for this criterion. A fourth-order time-stepping method is proposed by Chan et al. in [11] based on the approximation of the Euler-Lagrange equation, for the case $p = 2$, using upwind finite differences and a min-mod formula for estimating the curvature. Such high-order evolution method suffers from well-known stability and convergence issues that are difficult to handle.

A model, slightly different from (14), is tackled by Ballester et al. in [4] using a relaxation approach. The key idea is to replace the second-order term $\nabla \cdot \frac{\nabla u}{|\nabla u|}$ with a first-order term, depending on an auxiliary variable. More precisely, Ballester et al. study in [4] the minimization of

$$\int_{\tilde{\Omega}} |\nabla \cdot \theta|^p (a + b |\nabla G * u|) dx + \alpha \int_{\tilde{\Omega}} (|\nabla u| - \theta \cdot \nabla u) dx,$$

under the constraint that θ is a vector field with sub-unit modulus and prescribed normal component on the boundary of $\tilde{\Omega}$, and u takes values in the same range as in Ω^c . Clearly, θ plays the role of $\nabla u / |\nabla u|$, but the new criterion is much less singular. As for G , it is a regularizing kernel introduced for technical reasons in order to ensure the existence of a minimizing couple (u, θ) .

The main difference between the new relaxed criterion and (14), besides singularity, is the term $\int_{\tilde{\Omega}} |\nabla \cdot \theta|^p$ which is more restrictive, despite the relaxation, than $\int_{\tilde{\Omega}} |\nabla u| \left| \nabla \cdot \frac{\nabla u}{|\nabla u|} \right|^p dx$. However, the new model has a nice property: a gradient descent with respect to (u, θ) can be easily computed and yields two coupled second-order equations whose numerical approximation is standard. Results obtained with this model are shown in Fig. 11.

The Mumford-Shah-Euler model by Esedoglu and Shen [16] is also variational. It combines the celebrated Mumford-Shah segmentation model for images and the Euler's elastica model for curves. Being u_0 the original image defined on a domain A , and $\Omega \subset A$ the inpainting domain, Esedoglu and Shen propose to find a piecewise weakly smooth function u , that is a function with integrable squared gradient out of a discontinuity set $K \subset A$, that minimizes the criterion

$$\int_{A \setminus \Omega} \lambda \|u - u_0\|^2 dx + \int_{A \setminus K} \gamma |\nabla u|^2 dx + \int_K (\alpha + \beta \kappa^2) ds,$$

where $\alpha, \beta, \gamma, \lambda$ are positive parameters. The resulting image is not only reconstructed in the inpainting domain Ω , but also segmented all over A since the original image is not imposed as a hard constraint.

Two numerical approaches to the minimization of this model are discussed in Esedoglu and Shen [16]: first, a level set approach based on the representation of K as the zero-level set of a sequence of smooth functions that concentrate, and the explicit derivation, using finite differences, of the Euler-Lagrange equations associated with the criterion; second, a Γ -convergence approach based on a result originally conjectured by De Giorgi and recently proved by Röger and Schätzle in dimensions 2,3. In both cases, the final system of discrete equations is of order four, facing again difficult issues of convergence and stability.

More recently, following the work of Grzibovskis and Heintz on the Willmore flow, Esedoglu et al. [17] have addressed the numerical flow associated with the Mumford-Shah-Euler model using a promising convolution/thresholding method that is much easier to handle than the previous approaches.

Tschumperlé proposes in [28] an efficient second-order anisotropic diffusion model for multivalued image regularization and inpainting. Given a \mathbb{R}^N -valued image u known outside Ω , and starting from

an initial rough inpainting obtained by straightforward advection of boundary values, the pixels in the inpainting domain are iteratively updated according to a finite difference approximation to the equations

$$\frac{\partial u_i}{\partial t} = \text{trace}(T \nabla^2 u_i), \quad i \in \{1, \dots, N\}.$$

Here, T is the tensor field defined as

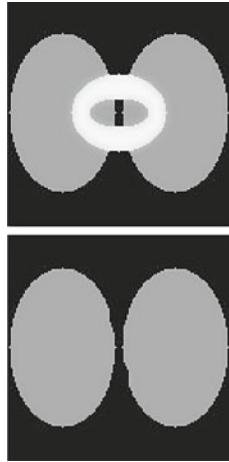
$$T = \frac{1}{(1 + \lambda_{\min} + \lambda_{\max})^{\alpha_1}} v_{\min} \otimes v_{\min} \\ + \frac{1}{(1 + \lambda_{\min} + \lambda_{\max})^{\alpha_2}} v_{\max} \otimes v_{\max},$$

with $0 < \alpha_1 \ll \alpha_2$, and $\lambda_{\min}, \lambda_{\max}, v_{\min}, v_{\max}$ are the eigenvalues and eigenvectors, respectively, of $G_\sigma * \sum_{i=1}^N \nabla u_i \otimes \nabla u_i$, being G_σ a smoothing kernel and $\sum_{i=1}^N \nabla u_i \otimes \nabla u_i$ the classical structure tensor, which is known for representing well the local geometry of u . Figure 12 reproduces an experiment taken from Tschumperlé [28].

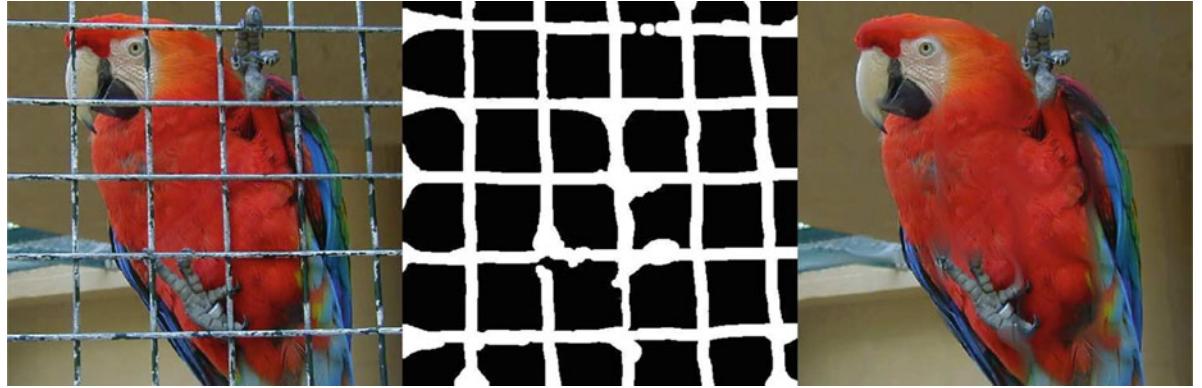
The approach of Auroux and Masmoudi in [3] uses the PDE techniques that have been developed for the inverse conductivity problem in the context of crack detection. The link with inpainting is the following: missing edges are modeled as cracks, and the image is assumed to be smooth out of these cracks. Given a crack, two inpainting candidates can be obtained as the solutions of the Laplace equation with Neumann condition along the crack and either a Dirichlet or a Neumann condition on the domain's boundary. The optimal cracks are those for which the two candidates are the most similar in quadratic norm, and they can be found through topological analysis, that is, they correspond to the set of points where putting a crack mostly decreases the quadratic difference. Both the localization of the cracks and the associated piecewise smooth inpainting solutions can be found using fast and simple finite difference schemes.

Finally, Bornemann and März propose in [9] a first-order model to advect the image information along the integral curves of a coherence vector field that extends in Ω the dominant directions of the image gradient. This coherence field is explicitly defined, at every point, as the normalized eigenvector to the minimal eigenvalue of a smoothed structure tensor whose computation carefully avoids boundary biases in the vicinity of $\partial\Omega$. Denoting c the coherence field,

Inpainting, Fig. 11 Two inpainting results obtained with the model proposed by Ballester et al. [4]. Observe in particular how curved edges are restored



Il Castello di Miramare e i suoi parco sorgono sul promontorio roccioso d'origine calcica di Grignano, per volon dell' arciduca Ferdinando Massimiliano d'Austria (1832-fratello minore dell'impero austriaco Francesco Giuseppe) Progettato nel 1856 da Carl terminato nell'aspetto esterno nel 1860. La sistemazione e la decorazione interna, o e Julius Hofmann, furono ultimati dopo la partenza di Massimiliano per il Messico nel 1864. Nell'867 l'imperatore del Messico, Maximiliano, venne fucilato a Querétaro.



Inpainting, Fig. 12 An inpainting experiment (the *middle* image is the mask defined by the user) taken from Tschumperlé [28]

Bornemann and März show that the equation $c \cdot \nabla u = 0$ with Dirichlet boundary constraint can be obtained as the vanishing viscosity limit of an efficient fast-marching scheme: the pixels in Ω are synthesized one at a time, according to their distance to the boundary. The new value at a pixel p is a linear combination of both known and previously generated values in a neighborhood of p . The key ingredient of the method is the explicit definition of the linear weights according to the coherence field c . Although the Bornemann-März model requires a careful tune of four parameters, it is much faster than the PDE approaches mentioned so far and performs very well, as illustrated in Fig. 13.

Combining and Extending PDEs and Patch Models

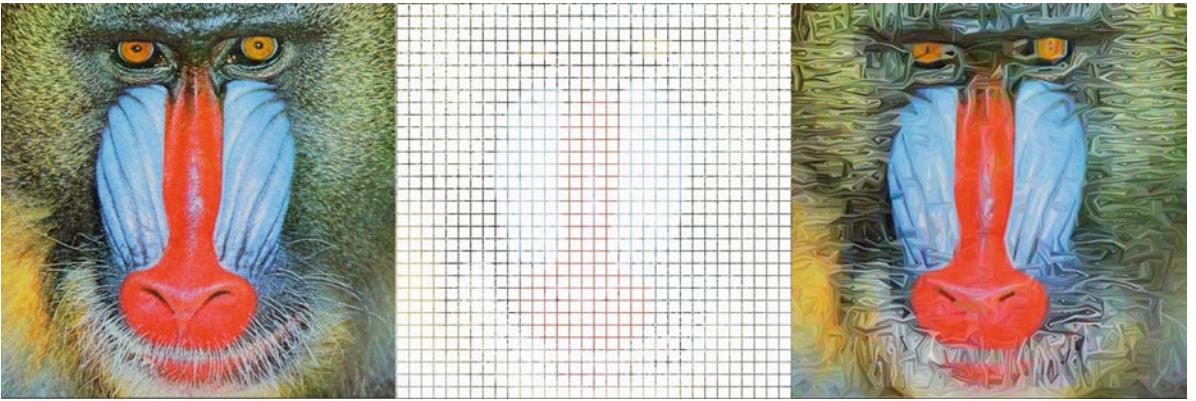
In general, most PDE/variational methods that have been presented so far perform well for inpainting either thin or sparsely distributed domains. However,

there is a common drawback to all these methods: they are unable to restore texture properly, and this is particularly visible on large inpainting domains, for instance, in the inpainting result of Fig. 12 where the diffusion method is not able to recover the parrot's texture. On the other hand, patch-based methods are not able to handle sparse inpainting domains like in Fig. 14, where no valid squared patch can be found that does not reduce to a point. On the contrary, most PDE/variational methods remain applicable in such situation, like in Fig. 14 where the model proposed by Masnou and Morel [23] yields the inpainting result. Obviously, some geometric information can be recovered, but no texture.

There have been several attempts to explicitly combine PDEs and patch-based methods in order to handle properly both texture and geometric structures. The contribution of Criminisi et al. [12] was mentioned



Inpainting, Fig. 13 An inpainting experiment taken from Bornemann and März [9], with a reported computation time of 0.4 s



Inpainting, Fig. 14 A picture of a mandrill, the same picture after removal of 15×15 squares (more than 87 % of the pixels are removed), and the reconstruction with the method introduced

by Masnou and Morel [23] using only the one-pixel-wide information at the squares' boundaries

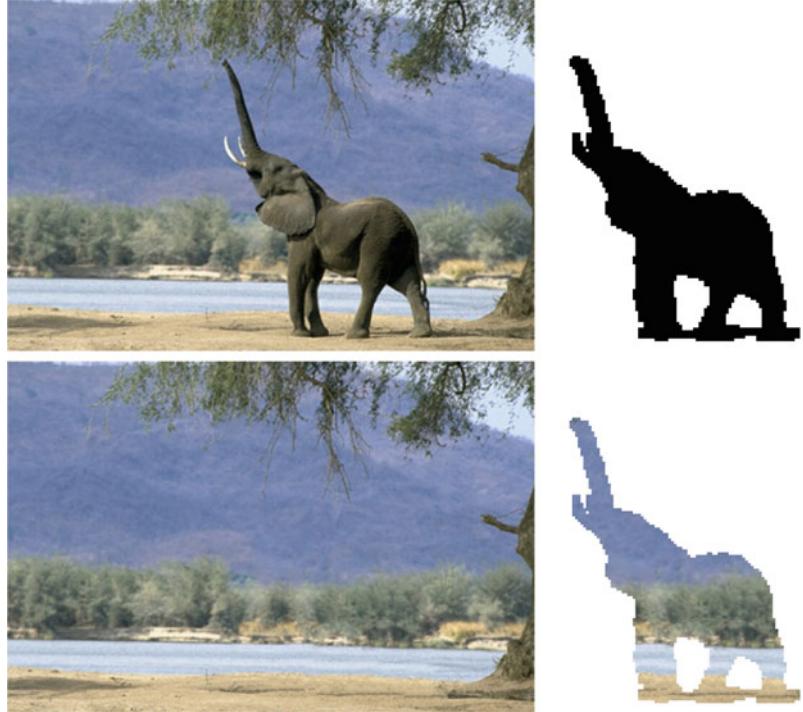
already. The work of Bertalmío et al. [8] uses an additive decomposition of the image to be inpainted into a geometric component that contains all edges information, and a texture component. Then the texture image is restored using the Efros and Leung's algorithm of [14], while the geometric image is inpainted following the method proposed in Bertalmío et al. [6] (several subsequent works have proposed other methods for the individual reconstruction of each component). The final image is obtained by addition of the restored texture and geometric components. In a few situations where the additive decomposition makes sense, this approach does indeed improve the result and extends the applications domain of inpainting.

In Komodakis and Tziritas [20], the authors combine variational and patch-based strategies by defining an inpainting energy over a graph whose nodes are the centers of patches over the image. The inpainting energy has two terms, one being a texture synthesis term and the other measuring the similarity of the overlapping area of two neighboring patches (centered on

nodes which are neighbors in the graph). By minimizing this energy with belief propagation, a label is assigned to each node, which amounts to copying the patch corresponding to the label over the position of the node. The results are very good on a variety of different images (e.g., Fig. 1), and the method is fast. Some potential issues are the following: there is no assurance that the iterative process converges to a global minimum, and visual artifacts may appear since the method uses a fixed grid and entire patches are copied for each pixel of the mask.

The work by Drori et al. in [13] does not involve any explicit geometry/texture decomposition, but the search for similar neighborhoods is guided by a prior rough estimate of the inpainted values using a multi-scale sampling and convolution strategy, in the very spirit of Ogden et al. [24]. In addition, in contrast with many patch-based methods, the dictionary of valid patches is enriched using rotations, rescalings, and reflections. An example extracted from Drori et al. [13] is shown in Fig. 15.

Inpainting, Fig. 15 An experiment from Drori et al. [13] illustrating the proposed multiscale diffusion/patch-based inpainting method. The *upper-left* image is the original, the *upper-right* image contains the mask defined by the user, the *bottom-left* image is the result, and the *bottom-right* image shows what has been synthesized in place of the elephant



Beyond Single-Image Inpainting

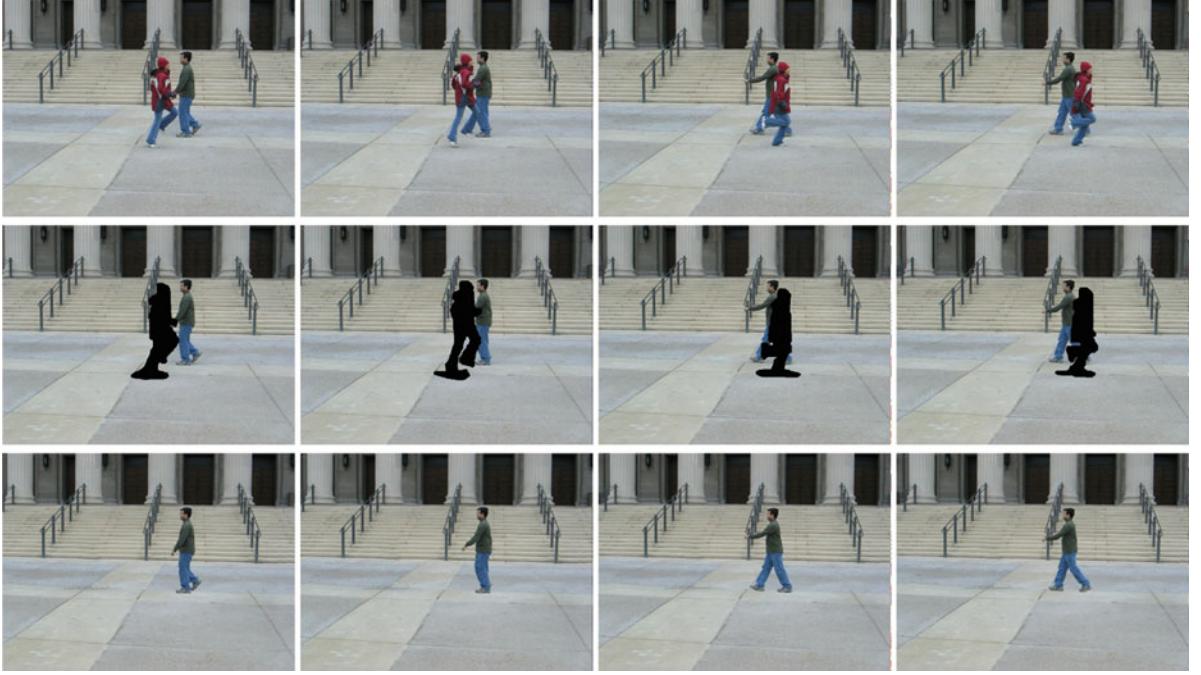
All the methods mentioned above involve just a single image. For the multi-image case, there are two possible scenarios: video inpainting and inpainting a single image using information from several images.

Basic methods for video inpainting for data transmission (where the problem is known as “error concealment” and involves restoring missing image blocks) and for film restoration applications (dealing with image gaps produced by dust, scratches, or the abrasion of the material) assume that the missing data changes location in correlative frames and therefore use motion estimation to copy information along pixel trajectories. A particular difficulty in video inpainting for film restoration is that, for good visual quality of the outputs, the detection of the gap and its filling in are to be tackled jointly and in a way which is robust to noise, usually employing probabilistic models in a Bayesian framework; see, for example, the book by Kokaram [19].

Wexler et al. [29] propose a video inpainting algorithm that extends to space-time the technique of Efros and Leung [14] and combines it with the idea of coherence among neighbors developed by Ashikhmin [2]. First, for each empty pixel P , they consider a space-time cube centered in P , compare

it with all possible cubes in the video, find the most similar, and keep its center pixel Q , which will be the correspondent of P . For each cube the information considered and compared is not only color but also motion vectors. Then, instead of copying the value of Q to P , they copy to P the average of all the values of the shifted correspondents of the neighbors of P : for instance, if R is at the *right* of P , and S is the correspondent of R , then the pixel to the *left* of S will be involved in the average to fill in P . This is based on the idea by Ashikhmin [2], see Fig. 5. The shortcomings of this video inpainting method are that the results present significant blur (due to the averaging), it seems to be limited only to static-camera scenarios (probably due to the simple motion estimation procedure involved) and periodic motion without change of scale, and the computational cost is quite high (due to the comparison of 3D blocks).

Shiratori et al. [26] perform video inpainting by firstly inpainting the motion field with a patch-based technique like that of Efros and Leung [14] and then propagating the colors along the (inpainted) motion trajectories. The method assumes that motion information is sufficient to fill in holes in videos, which is not always the case (e.g., with a static hole over a



Inpainting, Fig. 16 Top row: some frames from a video. Middle row: inpainting mask Ω in black. Bottom row: video inpainting results obtained with the algorithm of Patwardhan et al. [25]

static region). The results present some blurring, due to the bilinear interpolation in the color propagation step.

Patwardhan et al. [25] propose a video inpainting method consisting of three steps. In the first step they decompose the video sequence into binary motion layers (foreground and background), which are used to build three image *mosaics* (a mosaic is the equivalent of a panorama image created by stitching together several images): one mosaic for the foreground, another for the background, and a third for the motion information. The other two steps of the algorithm perform inpainting, first from the foreground and then from the background: these inpainting processes are aided and sped up by using the mosaics computed in the first step. See Fig. 16 for some results. The algorithm is limited to sequences where the camera motion is approximately parallel to the image plane and foreground objects move in a repetitive fashion and do not change size: these restrictions are imposed so that a patch-synthesis algorithm like that of Efros and Leung [14] can be used.

Hays and Efros [18] perform inpainting of a single image using information from a database with several millions of photographs. They use a scene descriptor

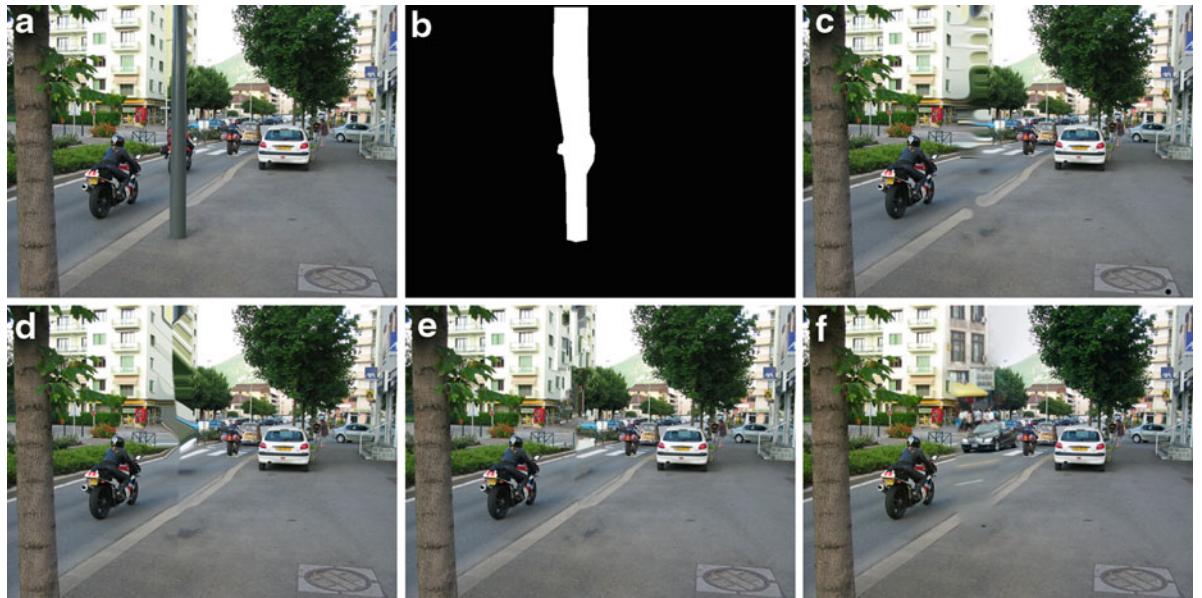
to reduce the search space from two million to two hundred images, those images from the database which are *semantically* closer to the image the user wants to inpaint. Using template matching, they align the 200 best matching scenes to the local image around the region to inpaint. Then they composite each matching scene into the target image using seam finding and image blending. Several outputs are generated so the user may select among them, and the results can be outstanding; see Fig. 17. The main shortcoming of this method is that it relies on managing and operating a huge image database. When the algorithm fails, it can be due to a lack of good scene matches (if the target image is atypical), or because of *semantic violations* (e.g., failure to recognize people, hence copying only part of them), or in the case of uniformly textured backgrounds (where this algorithm might not find the precise same texture in another picture of the database).

Open Problems

Inpainting is a very challenging problem, and it is far from being solved; see Fig. 18. Patch-based methods



Inpainting, Fig. 17 *Left:* original image. *Middle:* inpainting mask Ω , in white. *Right:* inpainting result obtained with the method by Hays and Efros [18], images taken from their paper



Inpainting, Fig. 18 An example where no inpainting method seems to work. (a) Original image, from the database provided by Hays and Efros [18]. (b) In white, the mask to be inpainted, which is not the initial mask proposed by Hayes and Efros but derives from the fuzzy mask actually used by their algorithm.

(c) Result courtesy of D. Tschumperlé using the algorithm from [28]. (d) Result courtesy of T. März and F. Bornemann using the algorithm from [9]. (e) Result using a variant of the algorithm from Criminisi et al. [12]. (f) Result from Hays and Efros [18]

work best in general, although for some applications (e.g., very spread, sparsely distributed gap Ω) geometry-based methods might be better suited. And when the image gap lies on a singular location, with surroundings that cannot be found anywhere else, then all patch-based methods give poor results, regardless if they consider or not geometry. For video inpainting the situation is worse; the existing algorithms are few and with very constraining limitations on camera and object motion. Because video inpainting is very relevant in cinema postproduction, in order to replace

the current typical labor intensive systems, important developments are expected in the near future.

References

1. Aharon M, Elad M, Bruckstein A (2006) K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans Signal Process* 54(11):4311
2. Ashikhmin M (2001) Synthesizing natural textures. In: Proceedings of the ACM symposium on interactive 3D graphics, Chapel Hill. ACM, pp 217–226

3. Auroux D, Masmoudi M (2006) A one-shot inpainting algorithm based on the topological asymptotic analysis. *Comput Appl Math* 25:1–17
4. Ballester C, Bertalmío M, Caselles V, Sapiro G, Verdera J (2001) Filling-in by joint interpolation of vector fields and gray levels. *IEEE Trans Image Process* 10(8):1200–1211
5. Barnes C, Shechtman E, Finkelstein A, Goldman DB (2009) Patchmatch: a randomized correspondence algorithm for structural image editing. *ACM Trans Graph* 28(3):2
6. Bertalmío M, Sapiro G, Caselles V, Ballester C (2000) Image inpainting. In: *Proceedings of SIGGRAPH'00*, New Orleans, USA, pp 417–424
7. Bertalmío M, Bertozzi A, Sapiro G (2001) Navier-Stokes, fluid dynamics, and image and video inpainting. In: *Proceedings of the IEEE international conference on computer vision and pattern recognition (CVPR)*, Hawaï
8. Bertalmío M, Vese L, Sapiro G, Osher S (2003) Simultaneous structure and texture image inpainting. *IEEE Trans Image Process* 12(8):882–889
9. Bornemann F, März T (2007) Fast image inpainting based on coherence transport. *J Math Imaging Vis* 28(3):259–278
10. Chan TF, Shen J (2001) Mathematical models for local deterministic inpaintings. *SIAM J Appl Math* 62(3):1019–1043
11. Chan TF, Kang SH, Shen J (2002) Euler's elastica and curvature based inpainting. *SIAM J Appl Math* 63(2):564–592
12. Criminisi A, Pérez P, Toyama K (2004) Region filling and object removal by exemplar-based inpainting. *IEEE Trans Image Process* 13(9):1200–1212
13. Drori I, Cohen-Or D, Yeshurun H (2003) Fragment-based image completion. In: *Proceedings of SIGGRAPH'03*, vol 22(3), pp 303–312
14. Efros AA, Leung TK (1999) Texture synthesis by non-parametric sampling. In: *Proceedings of the international conference on computer vision*, Kerkyra, vol 2, pp 1033
15. Elad M, Starck JL, Querre P, Donoho DL (2005) Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA). *Appl Comput Harmon Anal* 19(3):340–358
16. Esedoglu S, Shen J (2002) Digital image inpainting by the Mumford-Shah-Euler image model. *Eur J Appl Math* 13:353–370
17. Esedoglu S, Ruuth S, Tsai R (2008) Threshold dynamics for high order geometric motions. *Interfaces Free Boundaries* 10(3):263–282
18. Hays J, Efros AA (2008) Scene completion using millions of photographs. *Commun ACM* 51(10):87–94
19. Kokaram AC (1998) Motion picture restoration: digital algorithms for artefact suppression in degraded motion picture film and video. Springer, London
20. Komodakis N, Tziritas G (2007) Image completion using efficient belief propagation via priority scheduling and dynamic pruning. *IEEE Trans Image Process* 16(11):2649
21. Mairal J, Elad M, Sapiro G (2008) Sparse representation for color image restoration. *IEEE Trans Image Process* 17(1):53
22. Mairal J, Sapiro G, Elad M (2008) Learning multiscale sparse representations for image and video restoration. *SIAM Multiscale Model Simul* 7(1):214–241
23. Masnou S, Morel J.-M. (1998) Level lines based disocclusion. In: *5th IEEE international conference on image processing*, Chicago, IL, Oct 4–7
24. Ogden JM, Adelson EH, Bergen JR, Burt PJ (1985) Pyramid-based computer graphics. *RCA Eng* 30(5):4–15
25. Patwardhan KA, Sapiro G, Bertalmío M (2007) Video inpainting under constrained camera motion. *IEEE Trans Image Process* 16(2):545–553
26. Shiratori T, Matsushita Y, Tang X, Kang SB (2006) Video completion by motion field transfer. In: *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR)*, New York, vol 1
27. Sun J, Yuan L, Jia J, Shum HY (2005) Image completion with structure propagation. In: *ACM SIGGRAPH 2005 papers*. ACM, New York, p 868
28. Tschumperlé D (2006) Fast anisotropic smoothing of multi-valued images using curvature-preserving PDE's. *Int J Comput Vis* 68(1):65–82
29. Wexler Y, Shechtman E, Irani M (2004) Space-time video completion. In: *IEEE computer society conference on computer vision and pattern recognition (CVPR)*, Washington, DC, vol 1

Interactive Segmentation

Yuri Boykov

Department of Computer Science, University of Western Ontario, London, ON, Canada

Synonyms

Labeling; Object extraction; Partitioning; Segmentation; Semiautomatic; User-assisted; User-guided

Related Concepts

► [Dynamic Programming](#)

Definition

Interactive image segmentation is a (near) real-time mechanism for accurately marking/labeling an object of interest based on visual user interface (VUI) specifying seeds, rough delineation, partial labeling, bounding box, or other constraints. Semiautomatic interactive segmentation methods incorporate various generic image cues and/or object-specific feature detectors in order to facilitate acceptable results with minimum user efforts.

Background

The most basic object extraction techniques like *thresholding* (Fig. 1) and *region growing* are based on simple but very fast heuristics. The spectrum of applications for such techniques is limited as they are prone to many problems, most notably *leaking* as in Fig. 2. Despite significant problems, thresholding and region growing are widely known due to their simplicity and speed. For example, they could be easily run on personal computers available 15–20 years ago. More recent generations of commodity PCs allow much more robust segmentation techniques, which rely on optimization of some segmentation cost function, or an energy. An energy functional should define an explicit measure of goodness for evaluating any specific segmentation result. The main goal of optimization is to find the best segmentation with respect to the specified criteria. In the context of interactive segmentation, the energy can embed some *soft* and *hard* constraints specified by the user.

Discrete Segmentation Functionals

Many discrete optimization methods for interactive segmentation are based on classical combinatorial optimization techniques: *dynamic programming* (DP) or *s/t graph cuts*. In general, these approaches are guaranteed to find the exact global minimum solution in finite (low-order polynomial) number of steps. There are no numerical convergence issues (e.g., oscillations), and they work in near real time even on a single CPU. For efficiency, these methods are often implemented using the simplest 4-neighbor grids. In theory, this basic approach may generate some discrete metrification artifacts, but they are rarely observable on real images. Increasing the neighborhood size (e.g., to eight neighbors) adequately addresses the problem [8, 9].

Graph-path segmentation models are designed for 2D image segmentation. *Intelligent scissors* [2], also known as *live wire* in the medical imaging community [3], requires user to place seeds on the desired object boundary; see Fig. 3. The algorithm connects these seeds by computing the shortest path on a graph where edges (or nodes) are image weighted according to local contrast (intensity gradient). Such weighting

makes paths “stick” to image boundaries. The shortest paths from each new seed to all other image pixels can be pre-computed in $O(n \log n)$ time. Then, an optimal path from any mouse position to the seed can be previewed in real time.

This method evaluates segmentation boundary as a path between two seeds (see Fig. 4a) using energy functionals like

$$E(\mathcal{P}_{s,t}) = \sum_{\{p,q\} \in \mathcal{P}_{s,t}} w_{pq} \quad (1)$$

where $\mathcal{P}_{s,t}$ is a set of adjacent edges from source seed s to terminal seed t and edge weights $w_{pq} \geq 0$ are segmentation boundary costs based on some local measure of intensity contrast across edge $\{p,q\}$. One example of weights w_{pq} is

$$w_{pq} \propto \frac{1}{1 + |\nabla I \cdot n_{pq}|^2} \cdot \|p - q\|$$

where ∇I is image gradient, vector n_{pq} is a normal to edge $\{p,q\}$, and $\|p - q\|$ is the geometric length of edge $\{p,q\}$. Factor $\|p - q\|$ differentiates diagonal edges from horizontal and vertical edges on grids with higher connectivity, which can reduce the grid bias.

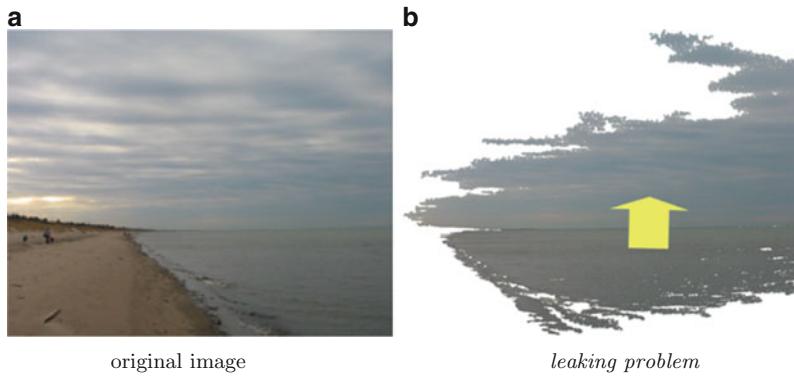
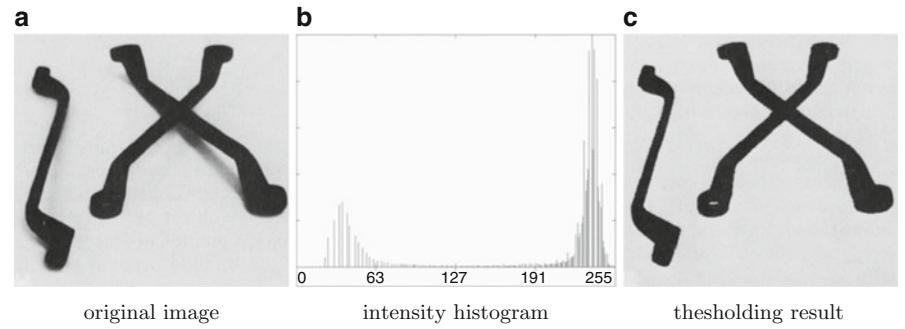
There is a number of other interactive segmentation methods based on efficient DP-based optimization algorithms. For example, the methods in [10, 11] compute globally optimal cycles (closed contours), minimizing ratios of different measures of segment’s boundary and region. For example, [11] can find a segment with the largest average contrast on its boundary. (Ratio of some cumulative contrast measure and the boundary length.) Optimization of ratio functionals evaluating boundary’s curvature was addressed in [12]. *Graph-cut segmentation models*: Boykov and Jolly [5] and Boykov and Funka-Lea [6] proposed an object extraction functional for N-dimensional images that evaluates boundary and region properties of segments as

$$\begin{aligned} E(x|\theta) = & - \sum_{p:x_p=0} \ln \Pr(I_p|\theta_0) - \sum_{p:x_p=1} \ln \Pr(I_p|\theta_1) \\ & + \sum_{\{p,q\} \in \mathcal{N}} w_{pq} \cdot [x_p \neq x_q] \end{aligned} \quad (2)$$

where $[\cdot]$ are *Iverson* brackets, variables x_p are binary object/background labels at pixels p , parameters

Interactive Segmentation, Fig. 1

Image thresholding segments a subset of pixels with intensities in a certain range, for example, $\{p : I_p < T\}$ in (c). Some threshold T works only if there is no overlap between the object and background intensities. The images above are from [1]



Interactive Segmentation, Fig. 2 Region growing is a greedy heuristic often associated with the *leaking* problem. Segment S is initialized by some *seed* in the object of interest (lake). Adjacent pixels q are iteratively added to S as long as some

“growing” criteria are met, for example, $\|I_q - I_p\| < T$ for some neighbor $p \in S$. A single low-contrast spot on the object boundary (horizon) will make the lake *leak* into the sky (b)

$\theta = \{\theta_0, \theta_1\}$ define object and background intensity distributions, and edge weights w_{pq} are a cost of discontinuity between a pair of neighboring pixels. For example,

$$w_{pq} \propto \exp\left(-\frac{|\nabla I \cdot e_{pq}|^2}{\sigma^2}\right) \cdot \frac{1}{\|p - q\|}$$

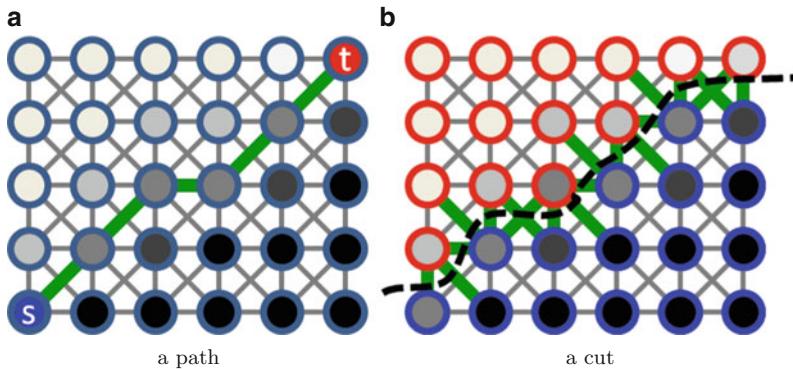
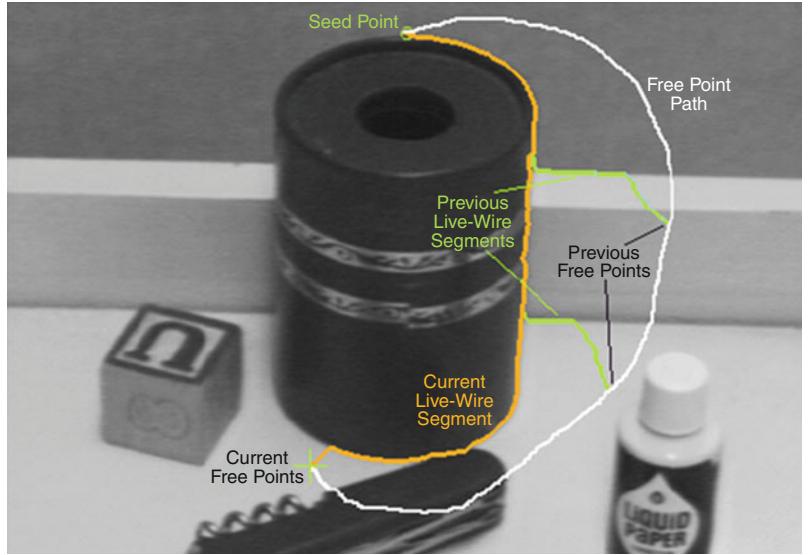
where e_{pq} is a unit vector collinear to edge $\{p, q\}$ and σ is a parameter controlling sensitivity to intensity contrast that is often set according to the level of noise in the image. Similarly to energy (1), the last term in (2) evaluates the image-weighted length of the segmentation boundary. In contrast to edge weights in (1), the weights above are based on intensity contrast along the edge $\{p, q\}$; see Fig. 4b. Normalization by edge length $\|p - q\|$ is required for grids with higher connectivity, reducing the grid bias [13].

The first two terms in (2) evaluate how well pixel intensities inside the object and background segments fit the corresponding distributions. In general, image intensity/color distributions could be extended by more sophisticated features and appearance models, for example, texture. The appearance models could be estimated from seeds or from prior data. The *grab-cut* method in [7] also uses an iterative EM-style scheme for additionally optimizing functional $E(x|\theta)$ in (2) with respect to parameters θ . In this case, sufficiently good initial appearance models can be often estimated from a user-placed box around the object. The graph-cut segmentation model also extends to video; for example, see the *snap-cut* method [14].

Functional (2) can be globally minimized over binary variables x by low-order polynomial algorithms from combinatorial optimization [15] that are fast even on a single CPU. Also, there are efficient

Interactive

Segmentation, Fig. 3
Intelligent scissors or live-wire methods connect seeds placed on object boundaries. Optimal segmentation boundary (orange curve) can be previewed for any mouse position in real time



Interactive Segmentation, Fig. 4 Segmentation on graphs. The path-based methods [2–4] represent segmentation boundary as a sequence of adjacent (green) edges (a). A path could connect two seeds or form a closed cycle. The graph-cut methods [5–7]

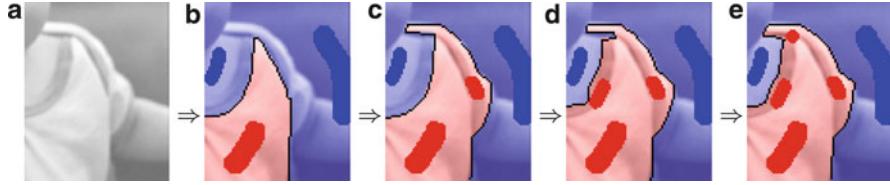
assign to pixels different labels, for example, *red* and *blue* in (b). Any such labeling implicitly defines a segmentation boundary, which is a *cut*, as a collection of (green) edges between differently labeled pixels

techniques [5, 6] for integrating interactive hard constraints (seeds) as in Fig. 5. Instead of segmentation energy (2), graph-cut framework can also use various ratio functionals [16].

Segmentation energy (2) works for N objects (labels). In general, its optimization is NP-hard for $N > 2$. An approximate solution with a factor of 2 optimality guarantee can be found via α -expansion optimization algorithm [17]. Interestingly, imposing some additional geometric constraints between object boundaries (e.g., inclusion, exclusion, minimum margin) may lead to exact polynomial optimization algorithms [18–20].

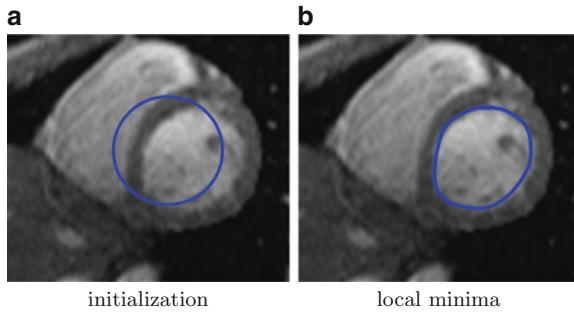
Continuous Segmentation Functionals

Many popular interactive segmentation methods use continuous representation of segments where boundaries are contours in \mathcal{R}^2 or surfaces in \mathcal{R}^3 . Such methods use either physics-based or geometric functionals to evaluate such continuous segments. Traditionally, variational calculus and different forms of *gradient descent* were used to converge to a local minima from a given initial contour; see Fig. 6. This motivates the general term, *active contours*, commonly used for such methods. Recent convex formulations for standard continuous regularization functionals [21, 22]



Interactive Segmentation, Fig. 5 Interactive editing of segments via hard constraints (seeds) based on graph cuts [5, 6]. A fragment of an original photo is shown in (a). Initial seeds and segmentation are shown in (b). The results in (c)–(e) illustrate

changes in optimal segmentation as new hard constraints are successively added. The computation time for consecutive corrections in (c)–(e) is marginal compared to time for initial results in (b)



Interactive Segmentation, Fig. 6 Snakes and other active contour methods are initialized by rough delineation of the desired object (a). Minimization of an energy associated with the contour leads to a local minimum (b) with better alignment to image boundaries

and development of continuous max-flow techniques [23, 24] now also allow good quality approximations of the global minima.

Physics-based segmentation models: Snakes [25, 26], balloons [27], spline snakes [28], and other methods explicitly represent object boundaries as an elastic band or a balloon. The band is normally assigned an internal energy (elasticity and stiffness) and a potential energy with respect to some external field of predefined *image forces* attracting the band to image boundaries, that is, locations with large intensity gradients. A user can also place seeds defining additional attraction or repulsion potentials.

Geometry-based segmentation models: Note that two visually identical *snakes* appearing in the same image position may have different internal elastic energies. In many cases this may contradict a natural assumption that a segmentation result can be evaluated only by its visible appearance (In some video applications the goal is to track specific points on a moving

segment, e.g. muscles of a beating heart. Physics-based (e.g. elastic) segmentation energy is well motivated in such cases). Based on this criticism of the physics-based approach, [29, 30] proposed *geodesic active contour* model evaluating contour C on a (bounded) domain Ω via geometric functionals like

$$E(C) = \int_{int(C)} f_1(p) dp + \int_{\Omega/int(C)} f_0(p) dp + \int_{\partial C} g(s) ds \quad (3)$$

which is similar to discrete model (2). The first two integrals in (3) are over the interior and exterior regions of C , and the third integral is the Riemannian length of C under metric g . Image-based *density* function g , for example,

$$g(p) = \frac{1}{1 + ||\nabla I(p)||^2}, \quad \forall p \in \Omega$$

shortens the length of contour C if it follows image boundaries where the density is small. The geometric length term in (3) is a continuous analogue of (1) and the spatial smoothness term in (2); see [13].

Scalar functions f_1 and f_0 on Ω are interior and exterior potentials based on some known appearance models for the object and background regions. For example, one can use $f_i(p) = -\ln \Pr(I_p|\theta_i)$ as in (2). Similarly, these potentials could also enforce user-placed hard constraints (seeds).

Geometric contours C can be represented as *level sets* of some scalar embedding function $u : \Omega \rightarrow \mathbb{R}^1$, for example, $C = \{p \in \Omega : u(p) = const\}$. This approach avoids some numerical issues, for example, the need for *reparameterization*, often associated with explicit representation of contour points needed for

physics-based bands. The level-set framework does not pose any topological constraints on contours and yields easy-to-implement gradient descent equations for geometric energies like (3). More recently, geometric energies like (3) are addressed with various continuous convex formulations [24, 31, 32] that are shown to converge to a good approximation of the global minimum.

The continuous geometric models are very closely related to discrete segmentation energies in the graph-cut framework [8, 13]. One theoretical advantage of the continuous formulations is absence of the grid bias. Continuous numerical schemes guarantee certain convergence rate, but some stopping threshold often needs to be chosen. Current fast implementations of continuous optimization methods, for example, [32], require GPU acceleration.

Distance-Based Segmentation Methods

Many interactive segmentation techniques optimize objective functions that are only indirectly related to the visual appearance of the segments and their boundaries. In particular, a large number of methods compute optimal (image-weighted) distance functions computed from seeds. For example, *fast-marching method* [33] extracts the boundary reached at time T by a front expanding with an image-weighted speed. This can be seen as a generalization of *region growing*. These ideas were extended in [34] where the segments are Voronoi cells w.r.t. geodesic distance $d(p, s)$ from the object and background seeds $s \in S_O \cup S_B$

$$x_p^* = \arg \min_{l \in \{O, B\}} \min_{s \in S_l} d(p, s)$$

Their image-based metric is based on gradients of the appearance models likelihoods instead of intensity gradients. Distance transforms can also work as the unary potentials in the segmentation models (2) and (3); for example, see [35].

Instead of the standard *min-sum* geodesic distance $d(p, s)$, many segmentation methods use other measures to compute Voronoi cells from the seeds. For example, *fuzzy-connectedness* methods [36, 37] compute Voronoi cells with respect to some *max-min* affinity measure. *Random walker* [38] outputs Voronoi

cells for probabilistic distance function $d(p, s)$, measuring the expected time of arrival for a random walk from p to s . *Watershed* method, for example, [39], connect points to seeds using *water-drop paths* instead of geodesics. *Power watershed* algorithm [40] unifies the ideas of *watershed* and *random walker*.

Some Open Problems

Energy functionals like (2) and (3) represent only the most standard ideas for evaluating segments. Accurate evaluation of the higher-order geometric properties of the boundary, for example, curvature [12, 41], remains a difficult optimization issue. Shape priors for globally optimal segmentation [42, 43] as well as enforcement of topological constraints [44, 45] are largely open problems.

References

1. Gonzalez RC, Woods RE (2007) Digital image processing, 3rd edn. Prentice Hall, Harlow
2. Mortensen EN, Barrett WA (1998) Interactive segmentation with intelligent scissors. Graph. Models Image Process 60:349–384
3. Falcão AX, Udupa JK, Samarasekera S, Sharma S (1998) User-steered image segmentation paradigms: live wire and live lane. Graph Models Image Process 60:233–260
4. Jermyn IH, Ishikawa H (1999) Globally optimal regions and boundaries. In: International conference on computer vision, Kerkyra, vol II, pp 904–910
5. Boykov Y, Jolly MP (2001) Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In: International conference on computer vision, Vancouver, vol II, pp 105–112
6. Boykov Y, Funka-Lea G (2006) Graph cuts and efficient N-D image segmentation. Int J Comput Vis 70(2):109–131
7. Rother C, Kolmogorov V, Blake A (2004) Grabcut – interactive foreground extraction using iterated graph cuts. ACM Trans Graph 23:307–331
8. Kolmogorov V, Boykov Y (2005) What metrics can be approximated by geo-cuts, or global optimization of length/area and flux. In: International conference on computer vision, Beijing
9. Boykov Y, Kolmogorov V, Cremers D, Delong A (2006) An integral solution to surface evolution PDEs via geo-cuts. In: European conference on computer vision (ECCV), Graz, Austria
10. Cox IJ, Rao SB, Zhong Y (1996) Ratio regions: a technique for image segmentation. In: International conference on pattern recognition, Vienna, vol II, pp 557–564

11. Jermyn IH, Ishikawa H (2001) Globally optimal regions and boundaries as minimum ratio weight cycles. *PAMI* 23(10):1075–1088
12. Schoenemann T, Cremers D (2007) Introducing curvature into globally optimal image segmentation: minimum ratio cycles on product graphs. In: International conference on computer vision (ICCV), Rio de Janeiro
13. Boykov Y, Kolmogorov V (2003) Computing geodesics and minimal surfaces via graph cuts. In: International conference on computer vision, Nice, vol I, pp 26–33
14. Bai X, Wang J, Simons D, Sapiro G (2009) Video Snap-Cut: Robust video object cutout using localized classifiers. In: ACM transactions on graphics (SIGGRAPH), Yokohama
15. Boykov Y, Kolmogorov V (2004) An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans Pattern Anal Mach Intell* 26(9):1124–1137
16. Kolmogorov V, Boykov Y, Rother C (2007) Applications of parametric maxflow in computer vision. In: International conference on computer vision (ICCV), Rio de Janeiro
17. Boykov Y, Veksler O, Zabih R (2001) Fast approximate energy minimization via graph cuts. *IEEE Trans Pattern Anal Mach Intell* 23(11):1222–1239
18. Li K, Wu X, Chen DZ, Sonka M (2006) Optimal surface segmentation in volumetric images-a graph-theoretic approach. *IEEE Trans Pattern Anal Pattern Recognit (PAMI)* 28(1):119–134
19. Delong A, Boykov Y (2009) Globally optimal segmentation of multi-region objects. In: International conference on computer vision (ICCV), Kyoto
20. Felzenszwalb PF, Veksler O (2010) Tiered scene labeling with dynamic programming. In: IEEE conference on computer vision and pattern recognition (CVPR), San Francisco
21. Chan T, Esedoglu S, Nikolova M (2006) Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM J Appl Math* 66(5):1632–1648
22. Pock T, Chambolle A, Cremers D, Bischof H (2009) A convex relaxation approach for computing minimal partitions. In: IEEE conference on computer vision and pattern recognition (CVPR), Miami
23. Appleton B, Talbot H (2006) Globally minimal surfaces by continuous maximal flows. *IEEE Trans Pattern Anal Pattern Recognit* 28(1):106–118
24. Yuan J, Bae E, Tai XC (2010) A study on continuous max-flow and min-cut approaches. In: IEEE conference on computer vision and pattern recognition (CVPR), San Francisco
25. Kass M, Witkin A, Terzopoulos D (1988) Snakes: active contour models. *Int J Comput Vis* 1(4):321–331
26. Amini AA, Weymouth TE, Jain RC (1990) Using dynamic programming for solving variational problems in vision. *IEEE Trans Pattern Anal Mach Intell* 12(9):855–867
27. Cohen LD, Cohen I (1993) Finite element methods for active contour models and balloons for 2-d and 3-d images. *IEEE Trans Pattern Anal Mach Intell* 15(11):1131–1147
28. Isard M, Blake A (1998) Active contours. Springer, London
29. Caselles V, Kimmel R, Sapiro G (1997) Geodesic active contours. *Int J Comput Vis* 22(1):61–79
30. Yezzi A Jr, Kichenassamy S, Kumar A, Olver P, Tannenbaum A (1997) A geometric snake model for segmentation of medical imagery. *IEEE Trans Med Imaging* 16(2):199–209
31. Unger M, Pock T, Cremers D, Bischof H (2008) Tvsseg - interactive total variation based image segmentation. In: British machine vision conference (BMVC), Leeds, UK
32. Santner J, Pock T, Bischof H (2010) Interactive multi-label segmentation. In: Asian conference on computer vision (ACCV), Queenstown
33. Malladi R, Sethian J (1998) A real-time algorithm for medical shape recovery. In: International conference on computer vision (ICCV), Bombay, pp 304–310
34. Bai X, Sapiro G (2007) A geodesic framework for fast interactive image and video segmentation and matting. In: IEEE international conference on computer vision (ICCV), Rio de Janeiro
35. Criminisi A, Sharp T, Blake A (2008) Geos: geodesic image segmentation. In: European conference on computer vision (ECCV), Marseille
36. Udupa JK, Samarasekera S (1996) Fuzzy connectedness and object definition: theory, algorithms, and applications in image segmentation. *Graph Models Image Process* 58(3):246–261
37. Herman GT, Carvalho BM (2001) Multiseeded segmentation using fuzzy connectedness. *IEEE Trans Pattern Anal Mach Intell (PAMI)* 23(5):460–474
38. Grady L (2006) Random walks for image segmentation. *IEEE Trans Pattern Anal Pattern Recognit (PAMI)* 28(11):1768–1783
39. Cousty J, Bertrand G, Najman L, Couplie M (2009) Watershed cuts: minimum spanning forests and the drop of water principle. *IEEE Trans Pattern Anal Mach Intell (PAMI)* 31(8):1362–1374
40. Couplie C, Grady L, Najman L, Talbot H (2011). Power watersheds: a unifying graph based optimization framework. *IEEE transactions on Pattern Analysis and Machine Intelligence (PAMI)* 33(7):1384–1399
41. Williams DJ, Shah M (1992) A fast algorithm for active contours and curvature estimation. *Comput Vis Graph Image Process* 55(1):14–26
42. Veksler O (2008) Star shape prior for graph-cut image segmentation. In: European conference on computer vision (ECCV), Marseille
43. Felzenszwalb P, Veksler O (2010) Tiered scene labelling with dynamic programming. In: IEEE conference on computer vision and pattern recognition (CVPR), San Francisco
44. Vicente S, Kolmogorov V, Rother C (2008) Graph cut based image segmentation with connectivity priors. In: IEEE conference on computer vision and pattern recognition (CVPR), Anchorage
45. Nowozin S, Lampert CH (2009) Global connectivity potentials for random field models. In: IEEE conference on computer vision and pattern recognition (CVPR), Miami

Interface Reflection

► Specularity, Specular Reflectance

Interpretation of Line Drawings

► [Line Drawing Labeling](#)

Interreflections

Michael S. Langer
School of Computer Science, McGill University,
Montreal, QC, Canada

Synonyms

[Mutual illumination](#)

Related Concepts

► [Bas-Relief Ambiguity](#); ► [Diffuse Reflectance](#);
► [Radiance](#)

Definition

Interreflections are reflections of light from one surface to another surface.

Background

Surfaces are illuminated not just by light sources but also by each other. These interreflections can provide a significant component of surface illumination, especially in concavities or enclosures. Numerical methods for computing interreflections were developed in the early twentieth century to solve problems in heat transfer such as in furnace design. The methods were developed further by the computer graphics community in the 1980s to render global illumination for scenes with Lambertian surfaces and later for scenes with specular components [7, 8].

Theory

Interreflections can be described mathematically in several equivalent ways. One way is to write the

reflected light as a sum of the light that is due to the illumination that arrives at a surface directly from the light source, plus the light that arrives from other surfaces in the scene via interreflections. Suppose the scene is composed of Lambertian surfaces with albedo $\rho(\mathbf{x})$ varying across surfaces. Let $L_s(\mathbf{x})$ be the component of \mathbf{x} 's outgoing radiance that is due to direct illumination from the source. Then the total radiance $L(\mathbf{x})$ leaving \mathbf{x} is the sum of $L_s(\mathbf{x})$ and the radiance that is due to interreflections:

$$L(\mathbf{x}) = L_s(\mathbf{x}) + \rho(\mathbf{x}) \int_S L(\mathbf{y}) K(\mathbf{x}, \mathbf{y}) d\mathbf{y}. \quad (1)$$

Here the integral is taken over all surface points $\mathbf{y} \in \mathcal{S}$ in the scene, and the function $K(\mathbf{x}, \mathbf{y})$ is a symmetric weighting function that depends on the surface normals at \mathbf{x} and \mathbf{y} and on the distances between \mathbf{x} and \mathbf{y} . $K(\mathbf{x}, \mathbf{y})$ is zero if the \mathbf{x} and \mathbf{y} are not visible to each other.

It is common to approximate Eq.(1) by using a polygonal mesh surface whose facets have constant radiance:

$$\mathbf{r} = \mathbf{r}_s + \mathbf{P}\mathbf{K}\mathbf{r} \quad (2)$$

where \mathbf{r} and \mathbf{r}_s are the vectors of total and direct radiance, respectively, \mathbf{P} is a diagonal matrix of albedos, and \mathbf{K} is called the *form factor* matrix. The above equations can be generalized to non-Lambertian surfaces as well [7, 8].

A second approach is to consider the eigenfunctions of \mathbf{K} which are radiance functions that are invariant to interreflections [9, 11]. These eigenfunctions are concentrated in surface concavities and at points of contact between surfaces [10].

A third approach is to use ray tracing to follow the light emitted from the source through successive reflections or bounces in the scene. The n th reflection serves as the source for the $n + 1$ st reflection, and the sum of all reflections gives an infinite series. For any scene geometry and reflectance, it is possible to construct a linear operator that can be applied iteratively to decompose the interreflections into their n bounce components [15]. Understanding the various bounces is especially important for making finite approximations. For example, a two-bounce model has been used to model how surface microfacets can account for non-Lambertian reflection [14] and how color bleeding occurs between surfaces in a concavity [5].

Application

Standard shape from shading and photometric stereo methods consider only the direct illumination component [6]. When interreflections are present, these methods produce erroneous results [4]. It is possible to extend these methods to account for interreflections by first ignoring interreflections to obtain an approximate solution and then iteratively updating the solution to account for interreflections. This idea has been applied to photometric stereo [12] and to shape from shading for the special case that the surface is an unfolded book in a photocopier [16].

The above applications assume the light sources are known. But what if they are unknown? An important fact that applies in this situation is the *bas-relief ambiguity* [1]. When a Lambertian surface is illuminated by direct illumination only, there exists a family of scenes (shape, albedo, lighting) that all produce the same image. With interreflections present, the bas-relief ambiguity no longer exists [3]. One can estimate the surface shape and reflectance similarly to above, namely, by applying a photometric stereo method that is designed for unknown lighting [17] and then iteratively updating the shape and reflectance to account for interreflections [3].

Interreflections also arise in projector-camera systems. An image that is projected on a concave screen will suffer from interreflections that will lower the image contrast. This contrast reduction can be compensated for, to some extent, by modifying the originally projected pattern [2]. A related example which involves active illumination of a 3D scene is to obtain a small number of images of the scene by illuminating it with a set of high-frequency projection patterns such as checkerboards [13]. The interreflection components of the scene will have relatively low spatial frequency and will be similar in the images. This property allows one to decompose the image of a fully illuminated scene into its direct component and its interreflection component. Unlike most methods for interreflections which assume Lambertian scenes, this method allows for non-Lambertian surfaces and other forms of reflections such as volume scattering.

References

- Belhumeur PN, Kriegman DJ, Yuille AL (1999) The bas-relief ambiguity. *Int J Comput Vis* 35(1):33–44

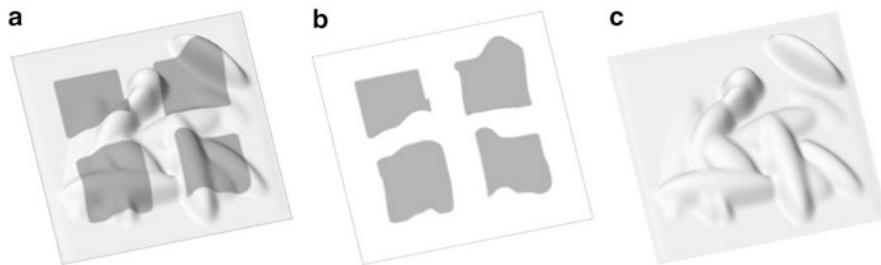
- Bimber O, Grundhofer A, Zeidler T, Danch D, Kapakos P (2006) Compensating indirect scattering for immersive and semi-immersive projection displays. In: Virtual reality conference 2006, Alexandria, pp 151–158, 25–29
- Chandraker MK, Kahl F, Kriegman DJ (2005) Reflections on the generalized bas-relief ambiguity. In: CVPR '05: proceedings of the 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR), Washington, DC, USA. IEEE Computer Society, pp 788–795
- Forsyth D, Zisserman A (1991) Reflections on shading. *IEEE Trans Pattern Anal Mach Intell* 13:671–679
- Funt BV, Drew MS, Ho J (1991) Color constancy from mutual reflection. *Int J Comput Vis* 6(1):5–24
- Horn BKP (1977) Understanding image intensities. *Artif Intell* 8(2):201–231
- Immel DS, Cohen MF, Greenberg DP (1986) A radiosity method for non-diffuse environments. *ACM Trans Graph* 20(4):133–142
- Kajiya JT (1986) The rendering equation. *ACM Trans Graph* 20(4):143–150
- Koenderink JJ, van Doorn AJ (1983) Geometrical modes as a general method to treat diffuse interreflections in radiometry. *J Opt Soc Am* 73(6):843–850
- Langer MS (1999) When shadows become interreflections. *Int J Comput Vis* 34(2/3):1–12
- Moon P (1940) On interreflections. *J Opt Soc Am* 30: 195–205
- Nayar SK, Ikeuchi K, Kanade T (1991) Shape from interreflections. *Int J Comput Vis* 6:173–195
- Nayar SK, Krishnan A, Grossberg MD, Raskar R (2006) Fast separation of direct and global components of a scene using high frequency illumination. *ACM Trans Graph* 25:935–944
- Nayar SK, Oren M (1995) Generalization of the Lambertian model and implications for machine vision. *Int J Comput Vis* 14(3):227–251
- Seitz SM, Matsushita Y, Kutulakos KN (2005) A theory of inverse light transport. In: ICCV '05: proceedings of the tenth IEEE international conference on computer vision, Washington, DC, USA. IEEE Computer Society, pp 1440–1447
- Wada T, Ukida H, Matsuyama T (1997) Shape from shading with interreflections under a proximal light-source: distortion-free copying of an unfolded book. *Int J Comput Vis* 24(2):125–135
- Yuille AL, Snow D, Epstein R, Belhumeur PN (1999) Determining generative models of objects under varying illumination: shape and albedo from multiple images using svd and integrability. *Int J Comput Vis* 35(3):203–222

Intrinsic Images

Marshall F. Tappen
University of Central Florida, Orlando, FL, USA

Related Concepts

- ▶ [Image Decompositions](#)



Intrinsic Images, Fig. 1 These images show an example of an intrinsic image decomposition. In this decomposition, the intrinsic images can be how the image in (a) can be decomposed into the albedo and shading images shown in (b) and (c), respectively

Definition

A set of images used to represent characteristics of a scene pictured in an image, with each image representing one particular characteristic of the scene.

Background

Vision systems have been categorized into low- and high-level processing, with high-level processing taking an object-centered approach [1]. In this categorization, the role of low-level processing is to extract basic characteristics at all locations in the image. These characteristics are then used to find objects.

Intrinsic images are a method for representing the low-level characteristics extracted from images. In the intrinsic image representation, proposed by Barrow and Tenenbaum in [2], one image represents each of the characteristics being used in the system. The value of each pixel represents the value of the characteristic at each point in the scene.

The types of characteristics that are conveniently expressed as intrinsic images include the illumination of each point in the scene, the motion at each point, the orientation of each point, the albedo, and the distance from the camera.

Application

Starting with [3], the term intrinsic images have also been used to refer to an image decomposition that decomposes an observed image into intrinsic images that can be recombined to recreate the observed image. The most common decomposition is, into images, representing the shading, or illumination, and albedo of each point. [Figure 1](#) shows an example of how the

image in [Fig. 1a](#) into shading and albedo images. Mathematically, the decomposition is modeled as

$$O_p = I_p \times R_p$$

where O_p is the the value of the observed image at pixel p , I is the illumination image, and R is the reflectance image.

In [3], Weiss recovers these intrinsic images from a sequence of images where the illumination varies in the scene. In [4] and [5], Tappen et al. use color and gray-scale features to estimate the decomposition from a single image.

Besides image decompositions, [6] proposes using intrinsic images that represent properties like occlusion boundaries and object depth.

References

1. Szeliski R (1990) Bayesian modeling of uncertainty in low-level vision. *Int J Comput Vis* 5(3):271–301
2. Barrow HG, Tenenbaum JM (1978) Recovering intrinsic scene characteristics from images. In: Hanson A, Riseman E (eds) Computer vision systems. Academic, New York, pp 3–26
3. Weiss Y (2001) Deriving intrinsic images from image sequences. In: The proceedings of the IEEE international conference on computer vision, Vancouver, pp 68–75
4. Tappen MF, Freeman WT, Adelson EH (2005) Recovering intrinsic images from a single image. *IEEE Trans Pattern Anal Mach Intell* 27(9):1459–1472
5. Tappen MF, Adelson EH, Freeman WT (2006) Estimating intrinsic component images using non-linear regression. In: The Proceedings of the 2006 IEEE computer society conference on computer vision and pattern recognition (CVPR), vol 2. IEEE Computer Society, Los Alamitos, pp 1992–1999
6. Hoiem D, Efros AA, Hebert M (2008) Closing the loop on scene interpretation. in: Proceedings the IEEE conference on computer vision and pattern recognition (CVPR). IEEE, Piscataway

Intrinsic Parameters

► Ininsics

Intrinsics

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Synonyms

Intrinsic parameters

Related Concepts

► Camera Parameters (Intrinsic, Extrinsic)

Definition

Intrinsics, short for *intrinsic parameters*, refer to the parameters belonging to the essential nature of a thing, which is usually a camera in computer vision. The intrinsic parameters of a camera include its focal length, the aspect ratio of a pixel, the coordinates of its principal point, and the lens distortion parameters.

See entry “► Camera Parameters” for more details.

Inverse Compositional Algorithm

Simon Baker
Microsoft Research, Redmond, WA, USA

Definition

The inverse compositional algorithm is a reformulation of the classic Lucas-Kanade algorithm to make the steepest-descent images and Hessian constant.

Background: Lucas-Kanade

The goal of the Lucas-Kanade algorithm is to minimize the sum of squared error between a template image $T(\mathbf{x})$ and a warped input image $I(\mathbf{x})$:

$$\sum_{\mathbf{x}} [T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x}; \mathbf{p}))]^2, \quad (1)$$

where $\mathbf{x} = (x, y)^T$ are the pixel coordinates, $\mathbf{W}(\mathbf{x}; \mathbf{p})$ is a parameterized set of warps, and $\mathbf{p} = (p_1, \dots, p_n)^T$ is a vector of parameters. The Lucas-Kanade algorithm assumes that a current estimate of \mathbf{p} is known and then iteratively solves for increments to the parameters $\Delta\mathbf{p}$, i.e., approximately minimize

$$\sum_{\mathbf{x}} [T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x}; \mathbf{p} + \Delta\mathbf{p}))]^2, \quad (2)$$

with respect to $\Delta\mathbf{p}$ and update the parameters

$$\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}. \quad (3)$$

Equation (2) is linearized by performing a first-order Taylor expansion:

$$\sum_{\mathbf{x}} \left[T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x}; \mathbf{p})) - \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta \mathbf{p} \right]^2. \quad (4)$$

In this expression, $\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right)$ is the *gradient* of image I and $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ is the *Jacobian* of the warp. Equation (4) has a closed-form solution as follows. The partial derivative of the expression in Eq.(4) with respect to $\Delta\mathbf{p}$ is

$$-2 \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^T \left[T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x}; \mathbf{p})) - \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta \mathbf{p} \right]. \quad (5)$$

Then denote

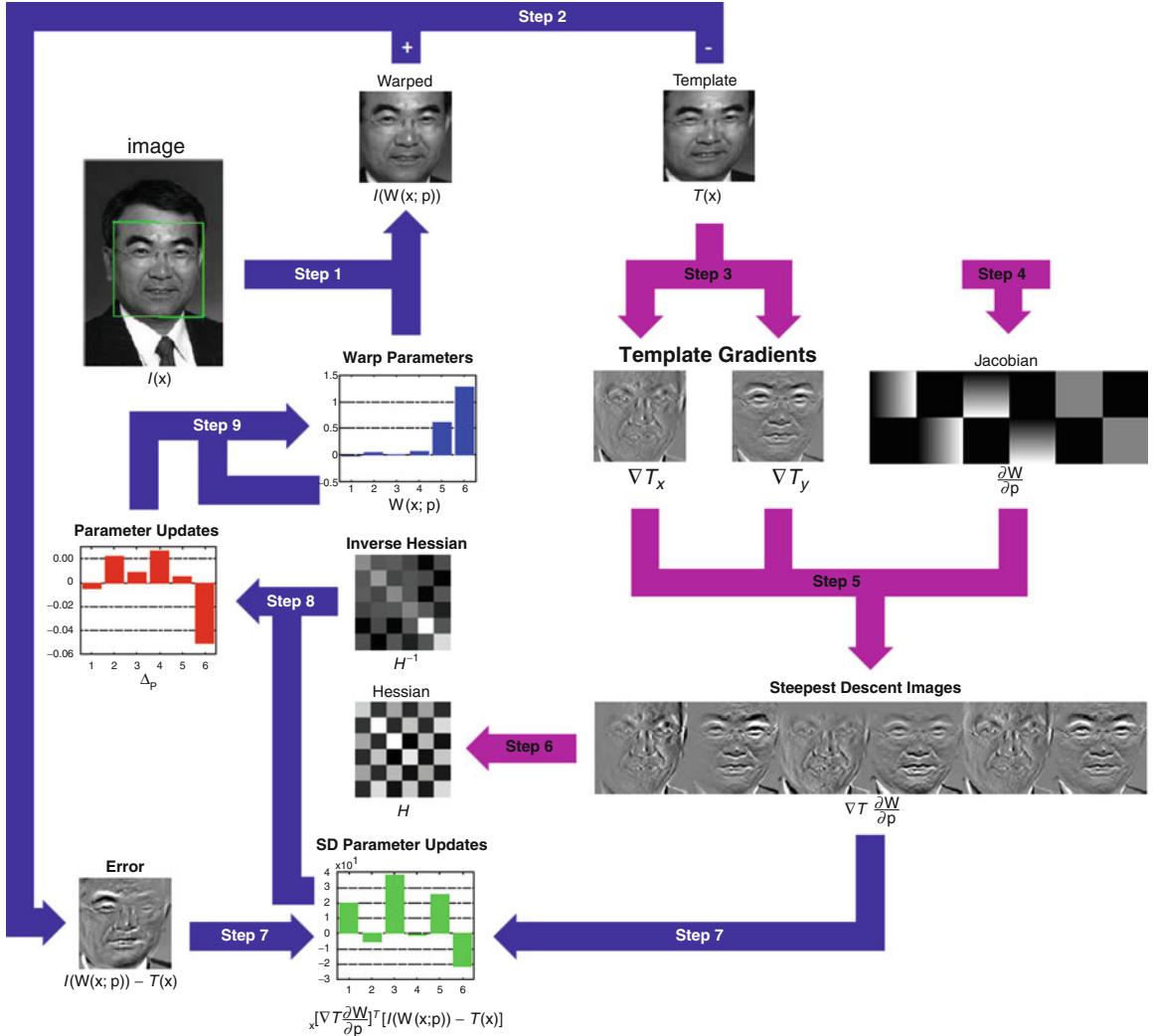
$$\mathbf{SD}_{lk}(\mathbf{x}) = \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}, \quad (6)$$

the *steepest-descent* images. Setting the expression in Eq.(5) to equal zero and solving give

$$\Delta \mathbf{p} = H_{lk}^{-1} \sum_{\mathbf{x}} \mathbf{SD}_{lk}^T(\mathbf{x}) E(\mathbf{x}) \quad (7)$$

where H_{lk} is the $n \times n$ (Gauss-Newton approximation to the) *Hessian* matrix

$$H_{lk} = \sum_{\mathbf{x}} \mathbf{SD}_{lk}^T(\mathbf{x}) \mathbf{SD}_{lk}(\mathbf{x}) \quad (8)$$



Inverse Compositional Algorithm, Fig. 1 A schematic overview of the inverse compositional algorithm. Steps 3–6 (*light-color arrows*) are performed once as a precomputation. The main algorithm simply consists of iterating image warping

(Step 1), image differencing (Step 2), image dot products (Step 7), multiplication with the inverse of the Hessian (Step 8), and the update to the warp (Step 9). All of these steps can be performed efficiently

and

$$E(\mathbf{x}) = T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x}; \mathbf{p})) \quad (9)$$

is the *error image*. The Lucas-Kanade algorithm consists of iteratively applying Eqs. (7) and (3). Because the gradient ∇I must be evaluated at $\mathbf{W}(\mathbf{x}; \mathbf{p})$ and the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ at \mathbf{p} , they both depend on \mathbf{p} . Both the steepest-descent images and the Hessian must therefore be recomputed in every iteration [1, 2].

The Inverse Compositional Algorithm

Baker and Matthews [3] proposed the inverse compositional algorithm as a way of reformulating image alignment so that the steepest descent images and Hessian are constant. Although the goal of the inverse compositional algorithm is the same as the Lucas-Kanade algorithm (e.g., minimizing Eq. (1)), the inverse compositional algorithm iteratively minimizes

$$\sum_{\mathbf{x}} [T(\mathbf{W}(\mathbf{x}; \Delta \mathbf{p})) - I(\mathbf{W}(\mathbf{x}; \mathbf{p}))]^2, \quad (10)$$

with respect to $\Delta\mathbf{p}$ and then updates the warp

$$\mathbf{W}(\mathbf{x}; \mathbf{p}) \leftarrow \mathbf{W}(\mathbf{x}; \mathbf{p}) \circ \mathbf{W}(\mathbf{x}; \Delta\mathbf{p})^{-1}. \quad (11)$$

The expression

$$\mathbf{W}(\mathbf{x}; \mathbf{p}) \circ \mathbf{W}(\mathbf{x}; \Delta\mathbf{p}) \equiv \mathbf{W}(\mathbf{W}(\mathbf{x}; \Delta\mathbf{p}); \mathbf{p}) \quad (12)$$

is the composition of 2 warps, and $\mathbf{W}(\mathbf{x}; \Delta\mathbf{p})^{-1}$ is the inverse of $\mathbf{W}(\mathbf{x}; \Delta\mathbf{p})$. The inverse compositional algorithm iterates Eq. (10) and (11) and can be shown to be equivalent to the Lucas-Kanade algorithm to first order in $\Delta\mathbf{p}$ [3].

Performing a first-order Taylor expansion on Eq. (10) gives

$$\sum_{\mathbf{x}} \left[T(\mathbf{W}(\mathbf{x}; \mathbf{0})) + \nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p} - I(\mathbf{W}(\mathbf{x}; \mathbf{p})) \right]^2. \quad (13)$$

Assuming that $\mathbf{W}(\mathbf{x}; \mathbf{0})$ is the identity warp, the minimum of this expression is

$$\Delta\mathbf{p} = -H_{ic}^{-1} \sum_{\mathbf{x}} \mathbf{SD}_{ic}^T(\mathbf{x}) E(\mathbf{x}), \quad (14)$$

where $\mathbf{SD}_{ic}^T(\mathbf{x})$ are the steepest-descent images with I replaced by T :

$$\mathbf{SD}_{ic}(\mathbf{x}) = \nabla T \frac{\partial \mathbf{W}}{\partial \mathbf{p}}, \quad (15)$$

H_{ic} is the Hessian matrix computed using the new steepest-descent images:

$$H_{ic} = \sum_{\mathbf{x}} \mathbf{SD}_{ic}^T(\mathbf{x}) \mathbf{SD}_{ic}(\mathbf{x}), \quad (16)$$

and the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ is evaluated at $(\mathbf{x}; \mathbf{0})$. Since there is nothing in either the steepest-descent images or the Hessian that depends on \mathbf{p} , they can both be precomputed. The inverse composition algorithm is illustrated in Fig. 1.

Application

The inverse compositional algorithm can be used almost anywhere the Lucas-Kanade can be. In can be

applied to anything from simple translational motion to dense optical flow. Perhaps the most significant application is its use to speed-up the fitting or tracking of active appearance models [4, 5].

References

1. Hager G, Belhumeur P (1998) Efficient region tracking with parametric models of geometry and illumination. IEEE Trans Pattern Anal Mach Intell 20(10):1025–1039
2. Shum HY, Szeliski R (2000) Construction of panoramic image mosaics with global and local alignment. Int J Comput Vis 16(1):63–84
3. Baker S, Matthews I (2004) Lucas-Kanade 20 years on: a unifying framework. Int J Comput Vision 56(3):221–255
4. Cootes T, Edwards G, Taylor C (2001) Active appearance models. IEEE Trans Pattern Anal Mach Intell 23(6):681–685
5. Matthews I, Baker S (2004) Active appearance models revisited. Int J Comput Vis 60(2):135–164

IP Camera

► Pan-Tilt-Zoom (PTZ) Camera

Irradiance

Fabian Langguth and Michael Goesele
GCC - Graphics, Capture and Massively Parallel Computing, TU Darmstadt, Darmstadt, Germany

Related Concepts

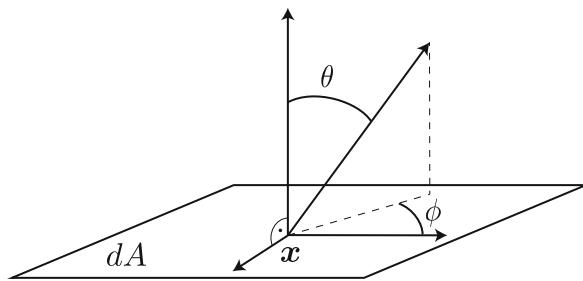
► Radiance

Definition

Irradiance E is defined as the incident power of electromagnetic radiation on a surface per unit surface area. It is expressed in watt per square meter ($\text{W} \cdot \text{m}^{-2}$).

Background

Irradiance is a concept from radiometry, the science of measuring radiant energy transfer [1]. The equivalent



Irradiance, Fig. 1 Geometric setting

concept in photometry is illuminance, with the key difference being that illuminance is adjusted to account for the varying sensitivity of the human eye to different wavelengths of light.

Theory

The irradiance at a surface point x is proportional to the radiance $L(x, \theta, \phi)$ arriving at x from direction (θ, ϕ) with a geometric foreshortening factor $\cos \theta$. Taking into account the whole hemisphere above the surface point, the irradiance is the integral over all incoming directions

$$E(x) = \int_{\theta, \phi} L(x, \theta, \phi) \cos \theta \, d\theta d\phi. \quad (1)$$

θ denotes the angle between the surface normal and the incident direction (θ, ϕ) (see Fig. 1).

Application

For a camera with an optical lens and an aperture, the image irradiance at a camera sensor is proportional to the radiance L emitted from a small scene patch in the form that

$$E = L \frac{\pi}{4} \left(\frac{d}{f} \right)^2 \cos \alpha^4 \quad (2)$$

where d is the aperture and f the focal length of the lens. α is the angle between the direction to the observed patch and the principal ray of the camera. For wide-angle lenses, the influence of α often results in a reduction of an image's brightness at the corners

compared to the image center. This effect is also called vignetting.

The pixel values of digital images are directly related to the irradiance at the sensor of the camera via the camera's response curve [2, 3]. Many computer vision techniques such as photometric stereo use this fact to recover information about the scene from the irradiance. Early works in this field include [4] and [5].

References

1. Dutré P, Bala K, Bekaert P (2006) Advanced global illumination. AK Peters, Wellesley
- 2.Debevec PE, Malik J (1997) Recovering high dynamic range radiance maps from photographs. In: Proceedings of the 24th annual conference on computer graphics and interactive techniques, SIGGRAPH '97, Los Angeles. ACM/Addison-Wesley, Los Angeles, pp 369–378
3. Robertson MA, Borman S, Stevenson RL (2003) Estimation-theoretic approach to dynamic range enhancement using multiple exposures. *J Electron Imaging* 12(2):219
4. Bruss AR (1981) The image irradiance equation: its solution and application (AITR-623). Artificial Intelligence Laboratory, Massachusetts Institute of Technology
5. Woodham R (1980) Photometric method for determining surface orientation from multiple images. *Opt Eng* 19(1):139–144

Isotropic Differential Geometry in Graph Spaces

Jan J. Koenderink

Faculty of EEMSC, Delft University of Technology,
Delft, The Netherlands
The Flemish Academic Centre for Science and the
Arts (VLAC), Brussels, Belgium
Laboratory of Experimental Psychology, University
of Leuven (K.U. Leuven), Leuven, Belgium

Synonyms

Differential Geometry of Graph Spaces; Dual Differential Geometry

Related Concepts

- Curvature; ► Curves in Euclidean Three-Space;
- Isotropic Differential Geometry in Graph Spaces

Definition

In many settings the conventional Euclidean differential geometry is not appropriate. A common case involves “graphs,” an instance being images where the carrier may be modeled as the Euclidean plane, but the intensity domain is incommensurate. Isotropic differential geometry allows one to deal with such cases.

Background

Isotropic differential geometry became highly developed during the first half of the twentieth century, mainly in German-speaking countries. The bulk of the literature is still in German.

Theory

There are frequent cases in computer vision and image processing in which the Euclidean \mathbb{E}^3 setting from classical differential geometry is not appropriate. A simple example is an image, which may be thought of as the Euclidean plane \mathbb{E}^2 , augmented with some “intensity domain.” Intensities are nonnegative quantities that somehow reflect photon-catches in, e.g., CCD devices. Usually the physical dimension is unclear and considered irrelevant to the problem. Then the structure of the intensity domain is most appropriately modeled by the affine line \mathbb{A}^1 , by considering the logarithm of the intensity modulo some arbitrary constant. But the $\mathbb{E}^2 \times \mathbb{A}^1$ -space is quite unlike \mathbb{E}^3 as becomes evident when one considers Euclidean rotations about some axis in the image plane. Such rotations make no sense because photon catches and lengths are incommensurable physical quantities. The correct way to proceed is to consider “image space” to be a fiber bundle with base space \mathbb{E}^2 and fibers \mathbb{A}^1 . Permissible transformations do not “mix” fibers, and Euclidean rotations about axes in the image plane are not among them.

This situation is typical in many contexts. The simplest example is perhaps a graph $y = f(x)$, where x and y are incommensurable physical quantities. Although the graph is evidently a curve in the xy -plane, it makes no sense to compute its Euclidean curvature as the result will depend on irrelevant transformations of the y -domain (Fig. 1).

A formal way to deal with such problems is to treat the y -axis as an isotropic dimension. Then the metric in the plane is essentially the separation in the x -dimension, the y -separation being treated as isotropic, i.e., nil. Thus the distance of points $\{x_1, y_1\}$ and $\{x_2, y_2\}$ is taken to be $x_2 - x_1$. Notice that this implies that points $\{x, y_1\}$ and $\{x, y_2\}$ with $y_1 \neq y_2$ are at mutually zero distance, *yet different*. One denotes such points “parallel” and assigns them the “special distance” $y_2 - y_1$. Only parallel points have a special distance, generic points only a proper distance. The group of “isotropic motions”:

$$x' = x + t_x, \quad (1)$$

$$y' = \alpha x + y + t_y, \quad (2)$$

conserves proper distance and, in the case of parallel points, special distance. This group is fit to replace the group of Euclidean movements:

$$x' = x \cos \alpha - y \sin \alpha + t_x, \quad (3)$$

$$y' = x \sin \alpha + y \cos \alpha + t_y, \quad (4)$$

and indeed has a somewhat similar (with important differences!) structure.

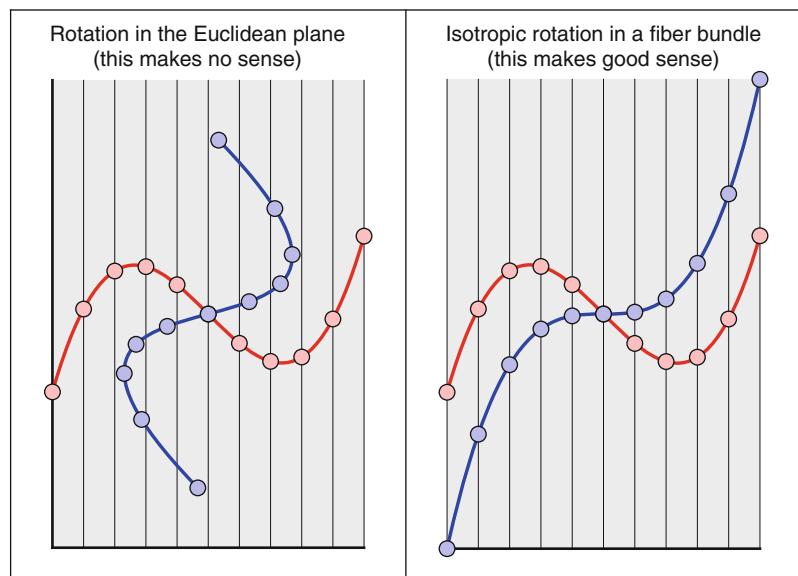
One obtains this group if the xy -plane is interpreted as the dual number plane. Dual numbers are complex numbers $z = x + \varepsilon y$, where the imaginary unit ε is defined as a nontrivial (i.e., not equal zero) solution of the quadratic equation $\varepsilon^2 = 0$. Thus $\varepsilon \neq 0$ whereas $\varepsilon^2 = 0$, from which one derives that neither $\varepsilon > 0$, nor $\varepsilon < 0$. Thus one is forced to use intuitionistic logic, for instance dropping the law of the excluded middle. A concrete representation is by way of matrices:

$$z = x + \varepsilon y = \begin{pmatrix} x & y \\ 0 & x \end{pmatrix}, \quad (5)$$

then addition and multiplication of dual numbers may be done by matrix algebra, similar to the conventional complex numbers $x + iy$ with imaginary unit $i^2 = -1$,

Isotropic Differential Geometry in Graph Spaces

Fig. 1 At left a graph in red, and the graph after a Euclidean rotation (in blue). This obviously makes no sense at all: the blue curve is not even a graph anymore! This is not a fiber bundle. At right the red graph has been subjected to an isotropic rotation. This makes perfect sense, one obtains another graph. Notice that the points move up and down along the fibers of the fiber space, they never leave their fiber, fibers “do not mix”



which are modeled through matrix algebra with the matrices:

$$z = x + iy = \begin{pmatrix} x & -y \\ y & x \end{pmatrix}. \quad (6)$$

However, although perhaps less scary, these matrix models are an unnecessary pain in hand calculations.

A linear transformation $z' = az + b$ with $a = 1 + \varepsilon\alpha$ and $b = t_x + \varepsilon t_y$ becomes:

$$\begin{aligned} z' &= (1 + \varepsilon\alpha)(x + iy) + (t_x + \varepsilon t_y) \\ &= (x + t_x) + \varepsilon(\alpha x + y + t_y), \end{aligned} \quad (7)$$

(using $\varepsilon^2 = 0$), i.e., exactly the transformation given above. Apparently the dual imaginary unit is an “infinitesimal.” Indeed, the full Taylor expansion of a function F about x is:

$$F(x + \varepsilon h) = F(x) + \varepsilon h F'(x). \quad (8)$$

Specifically, one has:

$$\sin \varepsilon\xi = \varepsilon\xi, \quad (9)$$

$$\cos \varepsilon\xi = 1, \quad (10)$$

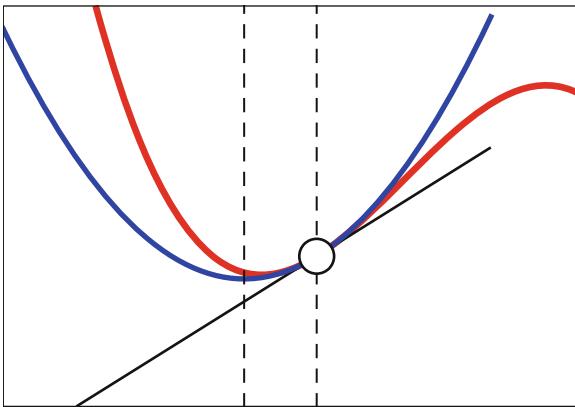
$$e^{\varepsilon\xi} = 1 + \varepsilon\xi, \quad (11)$$

thus trigonometry becomes really convenient. The polar representation of a dual number becomes:

$$z = x + \varepsilon y = x(1 + \varepsilon \frac{y}{x}) = |z|e^{\varepsilon \arg z}. \quad (12)$$

The dual angle is $\arctan y/x = y/x$ (notice that an isotropic angle equals its tangent!), thus the angle measure is parabolic instead of elliptic. Angles do not repeat with period 2π as in the Euclidean plane, but run between $\pm\infty$. Rotating the point 1 (that is $1 + \varepsilon 0$) about the origin over an angle α yields $1 + \varepsilon \alpha$, thus the line $x = 1$ is (part of) a unit circle. This brings one back to the original construction, and the rotations do not “mix” the x and y dimensions in a way that would be nonsense from the perspective of physics.

The group of proper motions (translations and rotations) of the dual plane leads to a differential geometry of curves that differs from that of the Euclidean plane. Consider the curve $z(x) = x + \varepsilon y(x)$. It is evidently parameterized by arc length, for $|z_x|^2 = 1$. The tangent is $t(x) = z_x = 1 + \varepsilon y_x(x)$, and is a unit vector, for $|t(x)| = 1$. The unit normal is ε , for $t_x(x) = \varepsilon y_{xx}(x)$ with (special) length $y_{xx}(x)$. Thus the normal is constant along the curve and useless for the purposes of differential geometry. The slope of the tangent is well defined though, the tangent subtends an angle $y_x(x)$ with the x -axis. The derivative of this angle with arc-length is $y_{xx}(x)$, thus one concludes that the curve has



Isotropic Differential Geometry in Graph Spaces, Fig. 2
Consider the local Taylor expansion of the *red curve* at the point indicated by the *white dot*. The first order is the drawn *black line*, the second-order approximation is the *blue parabola*. The center of this “circle of the second kind” is indicated by the leftmost *dotted line*. The other dotted line is the normal direction

curvature $\kappa(x) = y_{xx}(x)$. Notice that it is a much simpler expression than one has in the Euclidean plane, which is:

$$k(x) = \frac{y_{xx}(x)}{(1 + y_x(x)^2)^{3/2}}. \quad (13)$$

As expected, the Euclidean and the dual curvatures agree to first order, and for very shallow curves (infinitesimally near the x -axis) the Euclidean curvature degenerates to the dual curvature.

A curve:

$$z(x) = \varepsilon \frac{(x - c)^2}{2R}, \quad (14)$$

has curvature $1/R$, thus a radius of curvature R and is centered on $x = c$. It is evidently a circle in some sense, though different from the circle encountered above. One denotes $x = \pm 1$ a unit circle of the first kind, with center at the origin, $\varepsilon x^2/2$ a unit circle of the second kind, centered at the origin. The local second-order Taylor expansion of a curve is illustrated in Fig. 2. It is a parabola with isotropic axis, thus a “circle of the second kind.” It is the osculating circle to the curve in the isotropic geometry. The radius of the osculating circle is evidently the reciprocal of the second derivative, thus a curve $x + \varepsilon y(x)$ has curvature $y_{xx}(x)$ as argued above.

The differential geometry of curves and surfaces in a fiber bundle $\mathbb{E}^2 \times \mathbb{A}^1$ can be handled in a similar manner. All expressions are much simpler than those in Euclidean differential geometry, which is a very useful property, apart from the advantage that they make sense for a change. (Inappropriate applications of expressions taken from Euclidean differential geometry occur very frequently in computer vision and image processing. Although they certainly yield numerical results, they strictly speaking make no sense.) Thus the mean curvature of a surface $\{x, y, z(x, y)\}$ in Monge form becomes $2H = z_{xx} + z_{yy}$, the Gaussian curvature $K = z_{xx}z_{yy} - z_{xy}^2$, and so forth. Like in the planar case discussed above, the normal is constant, and thus useless. One uses the spatial attitude of the tangent plane instead. Any point of the surface may be mapped on the unit sphere of the second kind $z(x, y) = (x^2 + y^2)/2$ through parallelity of tangent planes. Even more conveniently, one notices that the xy -plane $\{x, y, 0\}$ is the stereographic projection of this sphere, thus conformal, but most remarkably – because different from the Euclidean case – also isometric. Thus the Gauss map becomes

$$\begin{aligned} \{x, y, z(x, y)\} \in \mathbb{E}^2 \times \mathbb{A}^1 &\mapsto \{-z_x(x, y), \\ &-z_y(x, y)\} \in \mathbb{R}^2, \end{aligned} \quad (15)$$

a map that is familiar in computer vision as “gradient space.” Gradient space is often used by way of a “linear approximation,” but it is really the exact Gauss map (or “spherical image”) in terms of the appropriate differential geometry.

The geometry of single isotropic space is well understood, although almost all of the literature is in German. The paper by Pottmann is the only reference in English on the general (space) setting, the book by Yaglom (translated into English from Russian) is an excellent introduction to the geometry of the dual plane.

Open Problems

This section introduced a very simple setting. In general one may have to deal with a graph over a curved surface. The paper by Pottmann gives some leads as how to handle such more general cases.

References

1. Koenderink JJ, van Doorn AJ (2002) Image processing done right. In: Proceedings of the European conference on computer vision (ECCV 2002). Lecture notes in computer science, vol 2350. Springer, Heidelberg, pp 158–172
2. Pottmann H, Opitz K (1994) Curvature analysis and visualization for functions defined on Euclidean spaces or surfaces, Comput Aided Geom Des 11: 655–674
3. Yaglom IM (1979) A simple non-Euclidean geometry and its physical basis: an elementary account of Galilean geometry and the Galilean principle of relativity. (trans: Shenitzer A). Springer, New York (translated from Russian)

Iterative Closest Point (ICP)

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Synonyms

ICP

Definition

Iterative closest point (ICP) is a popular algorithm employed to register two sets of curves, two sets of surfaces, or two clouds of points.

Background

The ICP technique was proposed independently by Besl and McKay [1] and Zhang [2] in two different contexts. Besl and McKay [1] developed the ICP algorithm to register partially sensed data from rigid objects with an ideal geometric model, prior to shape inspection. So this is a subset-set matching problem because each sensed point has a correspondence in the ideal model. Zhang [2] developed the ICP algorithm in the context of autonomous vehicle navigation in rugged terrain based on vision. His algorithm is used to register a sequence of sensed data in order to build a complete model of the scene and to plan a free path for navigation. So this is a subset-subset matching problem because a fraction of data in one set does not have any correspondence in the other

set. To address this issue, Zhang's ICP algorithm has integrated a statistical method based on the distance distribution to deal with outliers, occlusion, appearance, and disappearance. However, both algorithms share the same idea: iteratively match points in one set to the closest points in another set and refine the transformation between the two sets, with the goal of minimizing the distance between the two sets of point clouds.

Theory

The ICP algorithm is very simple and can be summarized as follows:

- *Input*: two point sets, initial estimation of the transformation
- *Output*: optimal transformation between the two point sets
- *Procedure*: Iterate the following steps:
 - (i) Apply the current estimate of the transformation to the first set of points.
 - (ii) Find the closest point in the second set for each point in the first transformed point set.
 - (iii) Update the point matches by discarding outliers.
 - (iv) Compute the transformation using the updated point matches, until convergence of the estimated transformation.

Here are a few comments on this general algorithm:

- Depending on the nature of the point sets, various pose estimation techniques described in the earlier sections can be used to compute the transformation between the two sets.
- The step of finding the closest point to a given point is generally the most time-expensive one. However, this step can be easily parallelized.
- Many data structures can be used to accelerate the finding of the closest point. They include k-D tree and octree.
- Instead of using all points from the first set, a selected subset of points (such as high curvature points) can be used to speed up the process, with only moderate sacrifice of the final accuracy.
- The above algorithm is not symmetric. Let point \hat{p}'_i in the second set be the closest point to a point p_i in the first set. In the other direction, point p_i is, in general, not necessarily the closest point to \hat{p}'_i . In

order to make the algorithm symmetric, we can find the closest point in the first transformed point set for each point in the second set and add these point matches to the overall set of matches. Better results can then be obtained at the expense of additional computational cost.

- When the ICP algorithm is applied to register curves or surfaces, they need to be sampled. The final accuracy depends on the density of sampling. The denser the sampling is, the higher the registration quality will be, but the more the computation will be required.

For more detailed and extensive discussions on ICP, the interested reader is referred to Sects. 7 and 8 of Zhang's paper [2].

There are several variants to the ICP algorithm. A useful variation is to substitute the point-to-point distance with point-to-plane distance [3]. The point-to-plane distance allows one surface to slide tangentially along the other surface, making it less likely get stuck in local minima. Consider a point p_i in the first set. Let point \hat{p}'_i in the second set be its closest point. Let the surface normal at

point p_i be n_i (a unit vector). Then, the point-to-plane distance measure is given by

$$d_i = n_i^T (\hat{p}'_i - p_i).$$

Surface normals can be precomputed to save computation.

References

1. Besl PJ, McKay ND (1992) A method for registration of 3-d shapes. IEEE Trans Pattern Anal Mach Intell 14(2):239–256
2. Zhang Z (1994) Iterative point matching for registration of free-form curves and surfaces. Int J Comput Vis 13(2):119–152. Also Research Report No.1658, INRIA Sophia-Antipolis, 1992
3. Chen Y, Medioni G (1992) Object modelling by registration of multiple range images. Image Vis Comput 10(3):145–155

Ives Transform

► [von Kries Hypothesis](#)

K

Kalman Filter

Gregory F. Welch

Institute for Simulation & Training, The University of Central Florida, Orlando, FL, USA

Synonyms

Kalman-Bucy filter; KF

Related Concepts

► Sensor Fusion

Definition

The Kalman filter is a set of mathematical equations that provides an efficient computational (recursive) means to estimate the state of a process, in a way that minimizes the mean of the squared error. The filter is very powerful in several aspects: it supports estimations of past, present, and even future states, and it can do so even when the precise nature of the modeled system is unknown.

Background

In 1960, Rudolf E. Kalman published his famous paper describing a recursive solution to the discrete-data linear filtering problem [1]. Since that time, due in large part to advances in digital computing, the Kalman filter has been the subject of extensive research and

application, particularly in the area of autonomous or assisted navigation. The goal of the filter is to produce evolving optimal estimates of a modeled process from noisy measurements of the process.

Theory

The Kalman filter addresses the general problem of trying to estimate the state $x \in \mathbb{R}^n$ of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \quad (1)$$

at time step k , with a measurement $z \in \mathbb{R}^m$ that is

$$z_k = Hx_k + v_k. \quad (2)$$

The random variables w_k and v_k represent the *process noise* and *measurement noise*, respectively. They are assumed to be independent of each other, white, and with normal probability distributions

$$p(w) \sim N(0, Q), \text{ and} \quad (3)$$

$$p(v) \sim N(0, R). \quad (4)$$

The $n \times n$ matrix A in the difference Eq. (1) relates the state x at the previous time step $k - 1$ to the state x at the current step k , in the absence of either a driving function or process noise. The $n \times l$ matrix B relates an optional control input $u \in \mathbb{R}^l$ to the state x . The $m \times n$ matrix H in the measurement Eq. (2) relates the state to the measurement z_k .

One usually does not know the true form of the process (1) and associated noise parameter (3) nor the true measurement model (2) and associated noise parameter (4), but in practice one can often arrive at useful models via analytical formulations and laboratory-based measurements.

Using the process and measurement models (1)–(4), and real (noisy) measurements \hat{z}_k at each time step k , the *Kalman filter* is used to recursively estimate the first two statistical moments of the process: the mean \hat{x}_k and the error covariance P_k .

The filter is typically implemented in two steps, a *time update* step and a *measurement update* step, as follows:

Time update:

$$\begin{aligned}\hat{x}_k^- &= A\hat{x}_{k-1} + Bu_{k-1} \\ P_k^- &= AP_{k-1}A^\top + Q\end{aligned}$$

Measurement update:

$$\begin{aligned}K &= P_k^- H^\top (HP_k^- H^\top + R)^{-1} \\ \hat{x}_k &= \hat{x}_k^- + K(\hat{z}_k - H\hat{x}_k^-) \\ P_k &= (I - KH)P_k^-\end{aligned}$$

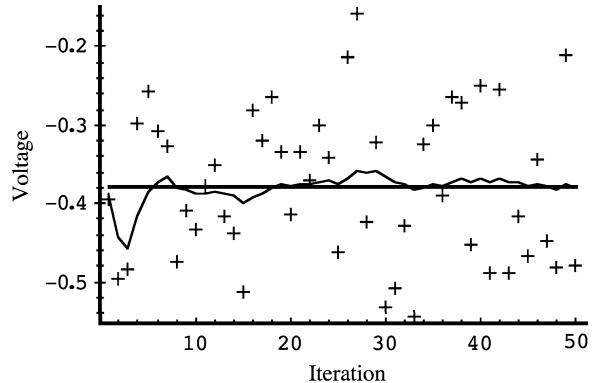
Repeatedly applying these steps recursively estimates the process mean \hat{x}_k and the error covariance P_k . Because the measurements can vary in form and timing, the filter is often characterized as a tool for *sensor fusion*.

The *Kalman filter* is optimal in that the $n \times m$ *Kalman gain* matrix K minimizes the trace of a posteriori error covariance P_k .

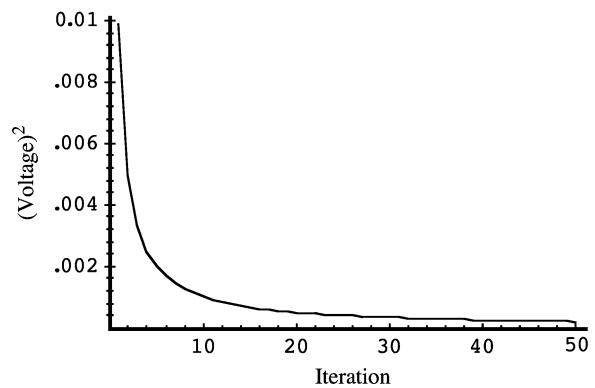
An accessible high-level introduction to the general idea of the Kalman filter can be found in Chap. 1 of [2]. A more complete introduction can be found in [3] and in [4] which also contains some interesting historical narrative. More extensive references include [2, 5–9].

Application

Despite the fact that employed process models rarely match the corresponding true systems, and the noise models rarely exhibit the characteristics required for optimality (zero mean, normally distributed, and independence over space and time), the Kalman filter remains popular – perhaps due to its relative simplicity and robustness. It continues to be used widely in diverse application areas such as electronics, robotics, localization, navigation, and even economics. In computer vision, variations of the Kalman filter



Kalman Filter, Fig. 1 The true random constant x_k (solid line), the noisy measurements \hat{z}_k (cross marks), and the filter estimate \hat{x}_k



Kalman Filter, Fig. 2 The error covariance P_k . After 50 iterations, the covariance has settled to a relatively small 0.0002 volts²

are typically used to estimate structure, motion, and camera parameters. Early examples include [10–13]. Both the OpenCV software project [14] and the Matlab numerical computing environment [15] include Kalman filter functions.

Experimental Results

A relatively simple example of using the Kalman filter to estimate a scalar random constant is given in [3], with complete details for the structure of the filter, the parameters, the initial conditions, and various results. The example presumes access to noisy measurements of a voltage that is corrupted by a 0.1 volt RMS white measurement noise. Referring back to Eqs. (1) and (2), the value to be estimated is presumed constant so $A = 1$, there is no control input so $u = 0$

(and B is irrelevant), and the noisy measurements are of the state (the voltage) directly so $H = 1$. For a true voltage of $x = -0.37727$, $Q = 1 \times 10^{-5}$, and $R = (0.1)^2 = 0.01$, plots for the true voltage x_k , noisy measurements, and estimated voltage \hat{x}_k are shown in Fig. 1; and the error covariance P_k is shown in Fig. 2.

References

1. Kalman RE (1960) A new approach to linear filtering and prediction problems. *Trans ASME J Basic Eng* 82 (Series D):35–45
2. Maybeck PS (1979) Stochastic models, estimation and control, vol 1. Volume 141 of mathematics in science and engineering. Academic, New York
3. Welch G, Bishop G (1995) An introduction to the Kalman filter. Technical report TR95-041, Department of Computer Science, University of North Carolina at Chapel Hill
4. Sorenson HW (1970) Least-squares estimation: from gauss to kalman. *IEEE Spectr* 7:63–68
5. Brown RG, Hwang PYC (1996) Introduction to random signals and applied Kalman filtering: with MATLAB exercises and solutions, 3rd edn. Wiley, New York
6. Gelb A (1974) Applied optimal estimation. MIT, Cambridge
7. Grewal MS, Andrews AP (2001) Kalman filtering theory and practice using MATLAB, 2nd edn. Information and system sciences series. Wiley, New York
8. Jacobs O (1993) Introduction to control theory, 2nd edn. Oxford University Press, Oxford/New York
9. Lewis FL (1986) Optimal estimation: with an introduction to stochastic control theory. Wiley, New York
10. Stuller J, Krishnamurthy G (1983) Kalman filter formulation of low-level television image motion estimation. *Comput Vis Graph Image Process* 21(2):169–204
11. Matthies L, Kanade T, Szeliski R (1989) Kalman filter-based algorithms for estimating depth from image sequences. *Int J Comput Vis* 3(3):209–238
12. van Pabst JV, Krekel PFC (1993) Multisensor data fusion of points, line segments, and surface segments in 3D space. *Proc. SPIE, Sensor Fusion VI*, 190 (August 20, 1993), pp 190–201
13. Azarbayejani A, Pentland A (1995) Recursive estimation of motion, structure, and focal length. *IEEE Trans Pattern Anal Mach Intell* 17(6):562–575
14. Bradski G (2000) The OpenCV Library. Dr. Dobb's Journal of Software Tools. <http://opencv.willowgarage.com/wiki/CiteOpenCV>
15. MATLAB (2012) Version 8.0.0.273 (R2012b). The MathWorks Inc. Natick, Massachusetts. <http://www.mathworks.com/products/matlab/>

Kalman-Bucy Filter

- [Kalman Filter](#)

Karhunen–Loèeve Transform (KLT)

- [Principal Component Analysis \(PCA\)](#)

Kernel Estimation

- [Blind Deconvolution](#)

KF

- [Kalman Filter](#)

Kinematic Chain Motion Models

- [Kinematic Motion Models](#)

Kinematic Motion Models

Christoph Bregler

Courant Institute, New York University, New York, NY, USA

Synonyms

[Kinematic chain motion models](#)

Definition

Kinematic motion models are mathematical models that describe the motion of objects without consideration of forces.

Background

Although kinematics is in general more broadly defined, in computer vision, the term kinematic motion model is usually used synonymously with kinematic chain motion models, a term that comes from the field of Robotics. Such a model defines a set of rigid objects (called links) that are connected with joints. The motion of the links is constraint by the degrees of freedom of the joints. For instance, a link can only

rotate relative to another link around a joint axis. These models are most commonly used to describe human and animal skeletal models or robotic manipulators. The motion constraints can be used for robust visual tracking of skeletal configurations in single-view or multi-view video. Other kinematic models include special cases like one single rigid object, or more general motion models like deformable models, often called nonrigid models. A kinematic motion model does not consider mass distributions and forces that influence the motion. This is described by so-called Dynamical Models.

Theory

A kinematic chain is a sequence of rigid links l_i that are connected by joints at location j_i (Fig. 1). There are different types of joints (Fig. 2). The simplest is a “revolute joint,” which has one axis of rotation α_i . Other possible joint types are “prismatic joints” (sliding along an axis) and joints with 2 or 3 axes of rotations (sometimes called “ball joints” or “spherical joints”). The configuration of the kinematic chain is defined by the relative joint angles between links. For instance, Fig. 1a shows a chain model with all angles set to 0 and another configuration (Fig. 1b, c) with different angle values. The first link in the chain is called the base link l_0 . To represent a human skeleton, the base link is usually the hip and several chains originating from l_0 define spine, head, arms, and legs. Connecting several chains this way leads to a kinematic tree, but for simplicity only chains are discussed. The base joint l_0 is either fixed or can move with any arbitrary translation T_0 and rotation R_0 . The configuration $\theta = [R_0, T_0, \alpha_1, \alpha_k]$ of all $k + 6$ degrees of freedom (local joint angles and l_0 translation and orientation) is often called the pose.

Forward Kinematics

Mathematically the kinematic model (M) can be defined in terms of how points P_i on a specific link in the rest-pose are moved to points Q_i by a new kinematic configuration θ (Fig. 1).

$$Q_i = M(\theta, P_i) \quad (1)$$

A very important aspect of kinematic chain models is, that a rotation of a link l_a affects all link motions

further down the chain l_b with $a > b$. For example, a rotation of a shoulder joint affects the global elbow position and rotation of the wrist link. But a local rotation of the elbow will not affect links further up. Starting with the scenario of changing the joint angle α_k of the last link l_k only (Fig. 1b), the motion of a point P_i on l_k to Q_i can be calculated by the following translation and rotation using homogeneous coordinates for P_i and $Q_i = [x, y, z, 1]^T$:

$$\begin{aligned} Q_i &= \begin{bmatrix} \mathbf{R}_k & (j_k - R_k \cdot j_k) \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ P_i &= G_k \cdot P_i \end{aligned} \quad (2)$$

The rotation matrix R_n can be parameterized using Euler Angles or the exponential map of a twist [1]. The exponential map of a twist leads to simpler computation of derivatives in tracking equations (as described below):

$$G_k(\alpha) = \exp \left(\begin{bmatrix} 0 & -\omega_z & \omega_y & v_1 \\ \omega_z & 0 & -\omega_x & v_2 \\ -\omega_y & \omega_x & 0 & v_3 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \alpha \right)$$

with twist $\xi_k = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$

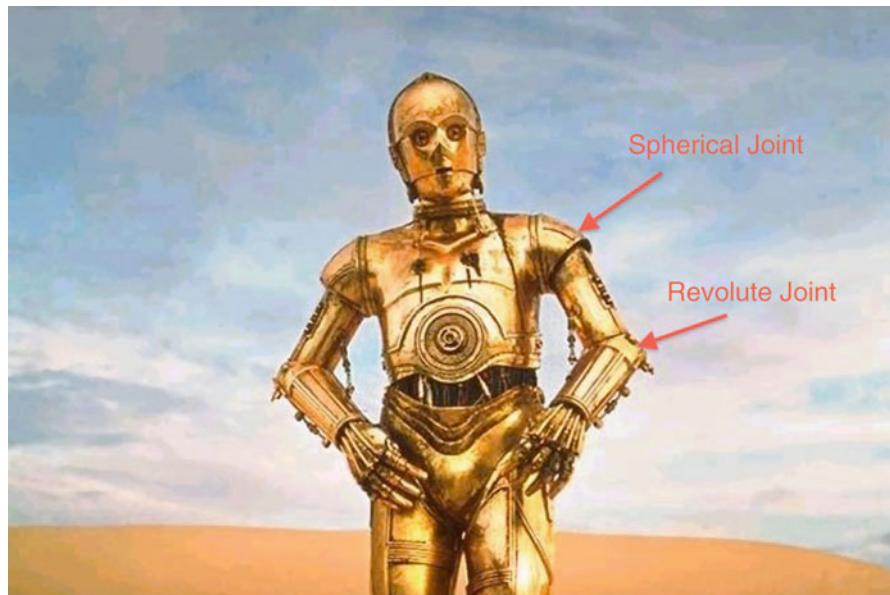
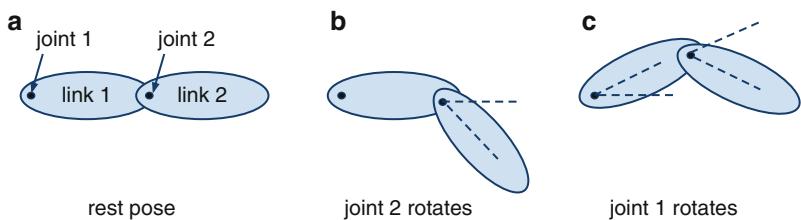
(3)

For a single axis joint (1 DOF), the twist ξ_k defines the axis at joint j_k (with $\omega_x^2 + \omega_y^2 + \omega_z^2 = 1$). α is the angle around the axis. The twist is constant, and α is the only varying parameter. For ball joints, it is modeled as a sequence on single axis joints around the same location j_k . Alternatively it can be modeled as letting all directional components for the twist vary.

Rotations of links further up the chain can be incorporated link by link. For instance, all points that are affected by a rotation around link location l_{k-1} are defined by G_{k-1} , including the points Q_i that have been moved by G_k already (Fig. 1c)

$$Q'_i = G_{k-1}(\alpha_{k-1}) \cdot Q_i = G_{k-1}(\alpha_{k-1}) \cdot G_k(\alpha_k) \cdot P_i \quad (4)$$

Kinematic Motion Models, Fig. 1 TODO example of a kinematic chain representation. (a) Rest pose. (b) Joint 2 rotates. (c) Joint 1 rotates



Kinematic Motion Models, Fig. 2 TODO different kinematic joint types

Working all the way backwards to the base link l_0 results in a product of G_0 to G_k

$$\begin{aligned} Q_i'' &= G_0 \cdot G_1 \cdot \dots \cdot G_k \cdot P_i = M_k(R_0, T_0, \alpha_1, \dots, \alpha_k) \cdot \\ P_i &= M(\theta) \cdot P_i \end{aligned} \quad (5)$$

The position and motion M_i of links further up the chain ($i < k$) can be modeled in the same way, as the product of G_0 to G_i . Using for G_0 to G_k the exponential map of twist representation, this equation is referred to as the “product of exponential map” representation [1]. An alternative representation is the more traditional Denavit-Hartenberg representation [2], which does not allow such an easy construction of the motion model or simple computation of derivatives and linearizations (more advantages of the product of exponential map representation are discussed in [1]).

Application

Kinematic Tracking

As mentioned earlier, kinematic chain models are commonly used in computer vision to track the articulated poses of humans. If visual markers are identified with high accuracy, either using a motion capture system or patterns that are easy to track, estimating the pose $\theta = [R_0, T_0, \alpha_1, \dots, \alpha_k]$ can be done with an error minimization technique. For instance, a marker-based motion capture system estimates 3D reconstructions of body markers \hat{Q}_i . The function $M(\theta)$ in Eq. (5) can be easily linearized [1] and used in estimating the 3D pose θ with the Newton-Raphson method or any other optimization technique. If 2D points are located in an image, the same can be applied to video tracking in single or multi-view footage using a camera model that relates 3D points to 2D projections. The $M(\theta)$ model can even be incorporated in optical flow models, and

a direct estimation of θ is possible from spatiotemporal image gradients [3]. More recent techniques do not assume 2D point tracks and use probabilistic techniques. For example, features like silhouettes, edges, color features, or patches can be estimated. The M function is then used to evaluate a pose hypothesis θ with an additional mapping of M to the feature space [4–11].

Most recently alternatives to kinematic motion models that use different data structures have been also applied to the domain of human pose estimation [12–16].

Model Estimation

Estimating the geometry of the kinematic chain (the point locations P_i in the rest pose, or other geometric shape representations) and all joint axes ξ_k in the rest pose can also be done with optimization techniques. This requires collecting a sequence of different poses that ideally go through the entire “range of motion” of a person or articulated object, a common procedure for motion capture systems, called “subject calibration.” The same can be also done for computer vision systems. Once a sufficient number of frames with different pose configurations are collected, jointly the poses θ_t for each frame t and the common model parameters ξ_k and P_i (that are constant over time) are estimated. Some example techniques are described in [17, 18].

Animation

Kinematic chain models are also used extensively in computer graphics and computer animation. Animating a character efficiently requires locally controlling the joint angles in the same way as previously outlined. All major 3D graphics packages support skeletal models that are based on kinematic chain motion models.

References

1. Murray R, Li Z, Sastry S (1994) A mathematical introduction to robotic manipulation. CRC, Boca Raton
2. Denavit J, Hartenberg R (1955) A kinematic notation for lower-pair mechanisms based on matrices. Trans ASME J Appl Mech 22(1):215–221
3. Bregler C, Malik J, Pullen K (2004) Twist based acquisition and tracking of animal and human kinematics. Int J Comput Vis 56(3):179–194
4. Hogg D (1983) Model-based vision: a program to see a walking person. Image Vis Comput 1(1):5–20
5. Rohr K (1994) Towards model-based recognition of human movements in image sequences. CVGIP-Image Underst 59(1):94–115
6. Gavrila D, Davis L (1996) 3-D model-based tracking of humans in action: a multi-view approach. In: 1996 IEEE computer society conference on computer vision and pattern recognition. San Francisco, pp 73–80
7. Rehg J, Kanade T (2005) Model-based tracking of self-occluding articulated objects. In: Proceedings of the fifth international conference on computer vision. IEEE, Cambridge, MA, pp 612–617
8. Duetscher J, Blake A, Reid I (2000) Articulated body motion capture by annealed particle filtering. In: IEEE conference on computer vision pattern recognition (CVPR). IEEE computer society, Miami, FL, p 2126
9. Sidenbladh H, Black M, Fleet D (2000) Stochastic tracking of 3D human figures using 2D image motion. In: European conference on Computer Vision 2000 (ECCV), Dublin, pp 702–718
10. Kakadiaris I, Metaxas D (2002) Model-based estimation of 3D human motion. IEEE Trans Pattern Anal Mach Intell 22(12):1453–1459
11. Sigal L, Bhatia S, Roth S, Black M, Isard M (2004) Tracking loose-limbed people. In: Proceedings. IEEE conference Computer Vision and Pattern Recognition (CVPR), Washington, DC, Vol.1, pp I-421-I-428
12. Ramanan D, Forsyth D, Zisserman A (2005) Strike a pose: tracking people by finding stylized poses. In: Proc. IEEE conference Computer Vision and Pattern Recognition (CVPR), San Diego, CA, vol. 1, pp 271–278
13. Felzenszwalb P, Huttenlocher D (2005) Pictorial structures for object recognition. Int J Comput Vis 61(1):55–79
14. Mori G, Malik J (2002) Estimating human body configurations using shape context matching. European conference on Computer Vision (ECCV), Copenhagen, pp 150–180
15. Shakhnarovich G, Viola P, Darrell T (2003) Fast pose estimation with parameter-sensitive hashing. 9th International conference on computer vision (ICCV), Nice France, Vol. 2 750–757
16. Agarwal A, Triggs B (2005) Recovering 3D human pose from monocular images. IEEE Trans Pattern Anal Mach Intell 28(1):44–58
17. Lu T, O’Connor J (1999) Bone position estimation from skin marker co-ordinates using global optimisation with joint constraints. J Biomech 32(2):129–134
18. Reinbold J, Schutte J, Fregly B, Koh B, Haftka R, George A, Mitchell K (2005) Determination of patient-specific multi-joint kinematic models through two-level optimization. J Biomech 38(3):621–626

L

Labeling

► [Interactive Segmentation](#)

Lambertian Model

► [Lambertian Reflectance](#)

Lambertian Reflectance

Sanjeev J. Koppal
Harvard University, Cambridge, MA, USA

Synonyms

[Lambertian model](#); [Lambert's law](#)

Related Concepts

► [Diffuse Reflectance](#); ► [Photometric Stereo](#)

Definition

Lambertian reflectance is a scene property that distributes the energy from any incident illumination into all viewing directions equally.

Background

Unlike the general, four-dimensional Bidirectional Reflectance Distribution Function (BRDF),

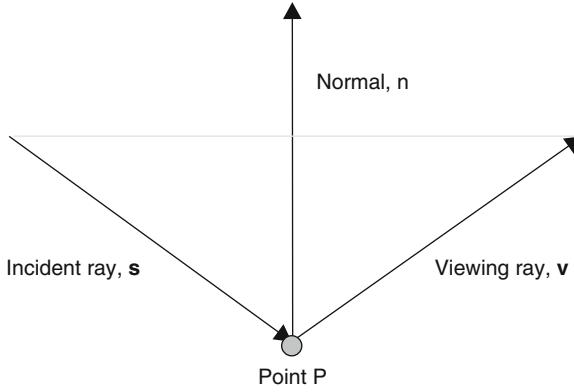
Lambertian reflectance has no dependency on viewing direction and is therefore a two-dimensional function. However, this reduction in complexity comes at a cost, since Lambertian reflectance does not account for cast shadows or specularities. This reflectance model is widely used due to both its computational tractability and its fidelity to many scenes, especially when the captured image data is of low resolution. Finally, the Lambertian model is an example of diffuse reflectance, which means it acts as a low-pass filter to incident illumination [1]. An artificial chemical called Spectralon is the real-world material with the closest appearance to the Lambertian ideal.

Theory

A scene point is said to exhibit Lambertian reflectance [2, 3] when the measured intensity E can be expressed in terms of the surface normal \mathbf{n} and incident illumination direction \mathbf{s} as

$$E = I\rho \max(\mathbf{n} \cdot \mathbf{s}, 0) \quad (1)$$

where I is the intensity of the light (which may be attenuated by falloff) and ρ is the scene point albedo (which is the ratio of the reflected and incident energies). Note that since the surface normal and illumination directions are unit vectors, Lambertian reflectance is a clamped cosine function, and this allows the model to account for attached shadows. In some applications, the raw dot product $\mathbf{n} \cdot \mathbf{s}$ is used as the reflectance model, and a nonnegativity constraint for irradiance is applied separately.



Lambertian Reflectance, Fig. 1 Reflection occurs when the incident ray, the surface normal, and the reflected lie in the same plane and, in addition, the angle between the incident ray and the normal is identical to that between the reflected ray and the normal

Extensions

A variety of extensions to Lambertian reflection exist to remove its most glaring deficiencies. The most common of these is a “diffuse + specular” extension [4],

$$E = I\rho_d \max(\mathbf{n} \cdot \mathbf{s}, 0) + I\rho_s \delta(\mathbf{v} - \mathbf{s} - 2\|\mathbf{n} \cdot \mathbf{s}\|\mathbf{n}) \quad (2)$$

where ρ_d and ρ_s are the diffuse and specular albedos of the scene and the δ function denotes when the viewing, incident, and surface normal vectors are aligned according to the Law of Reflection, as illustrated in Fig. 1.

Oren and Nayar [5] have generalized the Lambertian model to include microfacets, which model surface roughness by a gaussian distribution of the orientations of microstructure within any scene point. This extended model explains the visual appearance of rough diffuse Lambertian surfaces (such as the moon) and analytically reduces to the Lambertian model for a smooth surface whose microfacet orientation distribution is a delta function.

Application

The popularity of Lambertian models is due to its linearity. Consider a scene whose surface points are indexed by $i = 1, 2, \dots, k$. Without loss of generality, let the maximum value of the image intensity I and

albedo ρ both be in unity. If the viewing position is fixed and if t images are taken of the scene under different lighting directions, indexed by $j = 1, 2, \dots, t$, the images create a matrix whose analytic form is just a row-wise stacking of Eq. (2) as

$$\begin{pmatrix} E_{11} & E_{12} & \dots & E_{1k} \\ E_{21} & E_{22} & \dots & E_{2k} \\ \vdots & & & \\ E_{t1} & E_{t2} & \dots & E_{tk} \end{pmatrix} = \begin{pmatrix} s_1(x) & s_1(y) & s_1(z) \\ s_2(x) & s_2(y) & s_2(z) \\ \vdots & & \\ s_t(x) & s_t(y) & s_t(z) \\ n_1(x) & n_2(x) & \dots & n_k(x) \\ n_1(y) & n_2(y) & \dots & n_k(y) \\ n_1(z) & n_2(z) & \dots & n_k(z) \end{pmatrix} \quad (3)$$

which we can rewrite in matrix notation as $\mathbf{E} = \mathbf{S.N}$. Therefore, images taken of the scene can be linearly separated into a surface normal matrix and a light-source direction matrix. This separation allows many applications such as:

Explanations of Ambiguities: From the above equation, it is clear that many combinations of normals and light-source directions can create the same image data, since $\mathbf{E} = \mathbf{S.Q.Q}^{-1}.\mathbf{N}$, for any invertible Q . Belhumeur et al. [6] have shown that this ambiguity follows a closed form under orthographic viewing and distant illumination.

Scene Reconstruction: In classical photometric stereo [7], three or more images are taken of the scene with known illumination. Since the matrix \mathbf{S} is known, the surface normal matrix can be obtained by $\mathbf{N} = (\mathbf{S}^{-1}.\mathbf{E})$. Extensions to these methods have also been presented, where the illumination is unknown. In these cases, the matrix \mathbf{E} is decomposed using well-known linear methods such as QR decomposition or SVD under constraints such as smooth motion for the light source [8] or low rank [9]. Finally, we note that most extensions to the Lambertian model lose their linearity, and therefore, photometric stereo methods that model interreflections [10], microfacets [5], scattering [11], or near lighting [12] either apply nonlinear optimizations or require some form of calibration.

Subspace methods: Since the rows and columns of \mathbf{S} and \mathbf{N} lie in \mathcal{R}_3 , the intensity matrix \mathbf{E} is of at most rank 3. Belhumeur and Kriegman [13] further explain that, under fixed viewing and varying illumination, the set of images under Lambertian reflectance lies in a conical subspace. Such an argument underlines the success of dimensionality reduction methods, such



as PCA, in representing large collections of images of objects, such as faces. It also explains the existence of good recognition algorithms for scenes under varying illumination.

Relighting scenes: Data-driven methods for relighting under distant illumination exploit the additivity of light [14]. For the Lambertian case, such relighting can be modeled as creating new illumination directions. For example, consider the addition of images created by two sets of illumination directions S_1 and S_2 . The resultant data can be described as $E = S_1.N + S_2.N = S_3.N$, where $S_3 = S_1 + S_2$; that is, the relit Lambertian scene can be modeled as being illuminated by a third, new set of illumination directions.

References

1. Ramamoorthi R, Hanrahan P (2001) A signal-processing framework for inverse rendering. In: SIGGRAPH, Los Angeles
2. Lambert JH (1760) Photometria sive de mensura de gratibus luminis, colorum et umbrae. Eberhard Klett, Ausberg
3. Horn BKP (1986) Robot vision. MIT, Cambridge
4. Klinker G, Shafer S, Kanade T (1990) A physical approach to color image understanding. Int J Comput Vis 4(1):7–38
5. Oren M, Nayar SK (1995) Generalization of the lambertian model and implications for machine vision. Int J Comput Vis 14(3):227–251
6. Belhumeur P, Kriegman D, Yuille A (1999) The bas-relief ambiguity. Int J Comput Vis 35(1):33
7. Woodham RJ (1978) Photometric stereo. MIT AI Memo, Cambridge
8. Hayakawa H (1994) Photometric stereo under a light-source with arbitrary motion. J Opt Soc Am 11(11):3079
9. Basri R, Jacobs DW (2001) Photometric stereo with general, unknown lighting. In: IEEE conference on computer vision pattern recognition (CVPR), Kauai
10. Nayar SK, Ikeuchi K, Kanade T (1990) Shape from inter-reflections. Int J Comput Vis 6(3):173–195
11. Narasimhan SG, Nayar SK, Sun B, Koppal SJ (2005) Structured light in scattering media. In: ICCV, Beijing
12. Clark JJ (2006) Photometric stereo with nearby planar distributed illuminants. In: CRV, Quebec City
13. Belhumeur P, Kriegman D (1998) What is the set of images of an object under all possible illumination conditions? Int J Comput Vis 28(3):245
14. Unger J, Wenger A, Hawkins T, Gardner A, Debevec P (2003) Capturing and rendering with incident light fields. In: EGSR, Leuven

Lambert's Law

- [Lambertian Reflectance](#)

Learning from Partial Labels

- [Weakly Supervised Learning](#)

Lens Distortion Correction

- [Calibration of Projective Cameras](#)

Lens Distortion, Radial Distortion

Visesh Chari¹ and Ashok Veeraraghavan²

¹Institut National de Recherche en Informatique et en Automatique (INRIA), Le Chesnay Cedex, France

²Department of Electrical and Computer Engineering, Rice University, Houston, TX, USA

Related Concepts

- [Camera Calibration](#); ► [Lens Distortion, Radial Distortion](#)

Definition

Lens distortion is a form of optical aberration that causes lenses to deviate from rectilinear projection. Commonly, it is caused by defects in lens design and manufacturing.

Background

In many cameras, optical errors or the construction of the lens by nature itself does not allow for rectilinear projection (projection that maps 3D lines onto 2D lines). The deviation from rectilinear projection is termed lens distortion.

In most common lens systems designed for rectilinear projection, this occurs because of optical errors and is primarily radial in nature. This phenomenon, also called radial distortion, is of two types: barrel distortion ([Fig. 1](#)) and pincushion distortion ([Fig. 2](#)). In barrel distortion, image magnification increases non-linearly with distance from the optical axis resulting in lines “bulging outward.” Pincushion distortion



Lens Distortion, Radial Distortion, Fig. 1 Effects of barrel distortions respectively on lines (Distortions in these images have been exaggerated for visualization)



Lens Distortion, Radial Distortion, Fig. 2 Effects of pincushion distortions respectively on lines (Distortions in these images have been exaggerated for visualization)



Lens Distortion, Radial Distortion, Fig. 3 *Left:* Image taken using a fish-eye lens (Image courtesy Srikumar Ramalingam). *Right:* Image taken off a set of reflecting spheres (Image courtesy Yuichi Taguchi). Both methods of image capture produce distortions

has the opposite effect (lines bend radially inward), caused by decrease in image magnification with distance from the axis. Radial distortion is corrected by calibrating the camera using a planar “checkerboard” pattern.

The second, less damaging cause of distortion is the misalignment of optical elements in a compound lens. This is called tangential distortion or decentering distortion.

Some wide-angle lenses, like fish-eye lenses, have distortion as a product of the construction of the lens (sometimes in addition to radial distortion). Fish-eye lenses capture images that appear to be barrel distorted (Fig. 3).

Other Aberrations

Other common aberrations in lenses include chromatic aberration (wavelength-specific distortion) and spherical aberration (imperfect focusing of light rays incident at lens periphery).

Theory

The most common representation of radial and tangential distortion (for correction) is Brown’s distortion model [1, 2, 6], which represents the distorted image coordinate as



$$\mathbf{r} = \begin{pmatrix} r_x \\ r_y \end{pmatrix} = \mathbf{x}_u - \mathbf{cc}, \quad r = |\mathbf{r}| \quad (1)$$

$$\begin{aligned} \mathbf{x}_d = & (1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6) \mathbf{x}_u \\ & + \begin{pmatrix} 2\kappa_4 r_x r_y + \kappa_5 (r^2 + 2r_x^2) \\ \kappa_5 (r^2 + 2r_y^2) + 2\kappa_4 r_x r_y \end{pmatrix} \end{aligned} \quad (2)$$

where \mathbf{x}_u , \mathbf{x}_d are the undistorted and distorted point coordinates, $\kappa_{(1,2,3,4,5)}$ the distortion coefficients (first three radial and the rest tangential), and \mathbf{cc} the principal point.

Once estimated, images can be corrected for distortion by warping by a function that maps colors from the distorted point coordinates to the undistorted point coordinates.

Application

Radial distortion correction is generally applied as a preprocessing step in algorithms like panoramic stitching, structure from motion, and other algorithms that rely explicitly on the pinhole model of projection for their correct execution.

References

1. Brown DC (1966) Decentering distortion of lenses. Photogr Eng 32(3):444–462
2. Brown DC (1971) Close-range camera calibration. Photogr Eng 37(8):855–866
3. Byrod M, Brown M, Åström K. Minimal solutions for panoramic stitching with radial distortion
4. Distortion. <http://en.wikipedia.org/wiki/Distortion>
5. Fisheye lens. http://en.wikipedia.org/wiki/Fisheye_lens
6. Fryer JG, Brown DC (1986) Lens distortion for close-range photogrammetry. Photogr Eng Remote Sens 52(1): 51–58
7. Hartley RI, Kang SB (2007) Parameter-free radial distortion correction with center of distortion estimation. IEEE Trans Pattern Anal Mach Intell 29(8)
8. Kopf J, Uyttendaele M, Deussen O, Cohen MF Capturing and viewing gigapixel images

Lens Flare

► [Lens Flare and Lens Glare](#)

Lens Flare and Lens Glare

Dikpal Reddy¹ and Ashok Veeraraghavan²

¹Nvidia Research, Santa Clara, CA, USA

²Department of Electrical and Computer Engineering, Rice University, Houston, TX, USA

Synonyms

[Aperture ghosting](#); [Lens flare](#); [Veiling glare](#)

Related Concepts

► [Light Field](#)

Definition

Lens glare or lens flare is a global light transport phenomenon which occurs due to reflections and scattering of light in the optical elements and body of the camera and leads to decrease in the contrast of the image.

Background

Glare is a phenomenon caused due to bright light source (either narrow or extended) in or near the field of view of the camera. It is typically caused when stray light reaches the sensor and could be either due to reflections at the air-glass interface of the lens or due to scattering in the lens. The effect of glare is a decrease in contrast of the image and thereby its dynamic range and is often considered undesirable. Sometimes glare is used artistically in graphics or outdoor filming to convey a sense of realism. A simple way to decrease glare is to use lens hood to prevent stray light from entering the lens and by using antireflective coating on lens which decreases the reflections at the lens surface.

Glare can also be mitigated by first estimating it and then removing it from the image. This can be done in a passive fashion post capture or in an active fashion where the sensed image is modified for better glare estimation and removal.

Theory

While there is some confusion in the literature regarding the classification and terminology of different kinds of lens glare, glare can be broadly classified into



Lens Flare and Lens Glare, Fig. 1 Different kind of lens glare in the image. Aperture ghosting can be seen as hexagonal parasitic images symmetrically opposite the light source. Lens flare is the fogging seen around the light source. Veiling glare is the cause for decrease in contrast on the person's jacket and the wall behind (Figure courtesy [1])

three kinds as shown in Fig. 1. Two of these, aperture ghosting and lens flare, are caused by reflections at the air-glass interface of the lens. The third, veiling glare, is caused by the scattering of light in the lens.

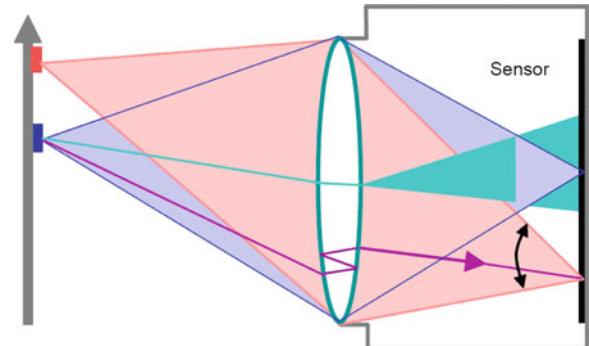
Aperture ghosting is caused due to Fresnel reflection at the air-glass interface resulting in a number of aperture shaped reflections in the image as shown in Fig. 1. These aperture-shaped reflections or ghosts occur in the image at the symmetrically opposite end of the light source. If there are n lens surfaces, the number of ghosts are $n(n - 1)/2$ [2].

Lens flare appears as fogging of a large image area as shown in Fig. 1 and is prominent when the aperture is large and the camera has a wide field of view. The effect is that region surrounding the bright light source is also bright due to reflections in the lens system as shown in Fig. 2.

Veiling glare is caused by scattering of light in the lens since lens acts as a weak diffuser as shown in Fig. 2. This results in a foggy appearance in the entire image with a distinct loss in contrast as shown in Fig. 1. Veiling glare is caused due to bright scene and not necessarily due to bright light sources close to the field of view.

In practice it is hard to distinguish and separate lens flare from veiling glare. Superior optics helps in mitigating glare by reducing the reflections at the lens surface.

Measuring glare: There are various ways of measuring glare. The standard way of measuring glare is to



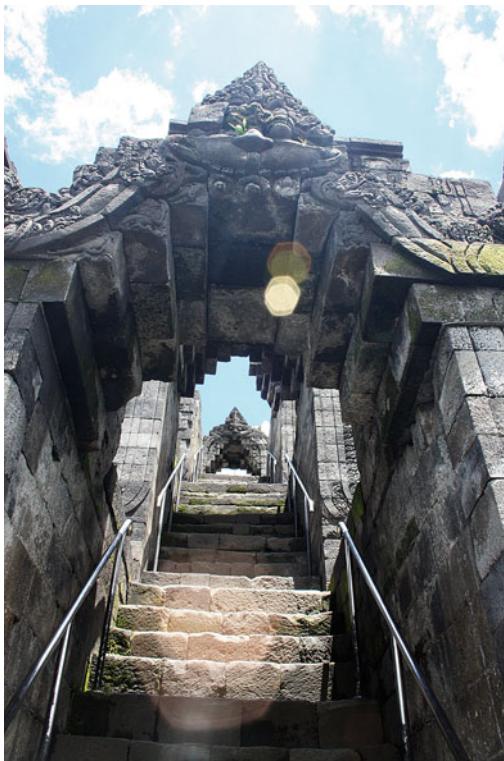
Lens Flare and Lens Glare, Fig. 2 The red and blue spot are focused on the sensor. The purple ray from the blue light source is undergoing reflection, which results in either aperture ghosting and/or lens flare. The cyan ray is scattering due to weak diffusion caused by the lens resulting in veiling glare (Figure courtesy [1])

photograph a white board with a black target in the center [3, 4]. The ratio of the luminance of the black target to the luminance in the white region is defined as veiling glare index (VGI) by the ISO standard 9358 (year 1994).

Glare spread function (GSF) is another measure defined by ISO standard to quantify glare. It is defined as the glare intensity as function of distance from the center of an image of a centered point light source. If the glare intensity is invariant to the location of the point light source in the image, then GSF can be represented by a convolution operation like the point spread function (PSF).

Traditional methods of removing glare involve assuming a shift invariant GSF, estimating it, and performing deconvolution. This has been used in removing veiling glare in X-ray imaging [5]. The methods using deconvolution are passive where the glare is estimated and removed after capture. Active methods to remove glare have also been proposed, and they involve modifying the imaging architecture for better glare estimation and removal.

Talvala et al. [6] estimate veiling glare by taking multiple images of a scene by shifting a mask. This method allows the glare to be estimated even when the glare is significantly dominant. Other active approach involves estimating aperture ghosting and lens flare due to point light source by inserting a mask near the sensor and capturing a light field [1]. This method treats glare as high-frequency phenomenon in ray space as opposed to image space and defines a 4D glare ray spread function (GRSF) that



Lens Flare and Lens Glare, Fig. 3 An image of the stairs with distinct aperture ghosting. The glare in the image enhances the sense of ascending (Photocredit: Wikipedia and Gunawan Kartapranata)

characterizes how much an incident ray on lens contributes to outgoing rays. Please refer the above entries for technical details.

Application

Glare is used in photography and videography to enhance the scene by conveying a sense of realism as shown in Fig. 3.

Glare estimation and removal is an important step in HDR imaging since the presence of glare in captured images reduces the dynamic range. Glare removal techniques in HDR are described in [7]. In multi-exposure HDR imaging, it has been used in [8]. Glare removal techniques are also used in X-ray imaging [5, 9]. They are also used in astronomical imaging where scattering effects cause a degradation in the image. Glare removal has been used to remove glare caused by water droplets sticking to the lens [10].

Since aperture ghosting is symmetrically opposite to the light source, techniques have been proposed to estimate the light source location and the optical center of the camera using this fact [11].

References

- Raskar R, Agrawal A, Wilson CA, Veeraraghavan A (2008) Glare aware photography: 4d ray sampling for reducing glare effects of camera lenses. ACM Trans. Graph. 27(56):1–56:10
- Ray SF (2002) Applied photographic optics: lenses and optical systems for photography, film, video, electronic and digital imaging. Taylor & Francis US
- Kuwabara G (1953) On the flare of lenses. J Opt Soc Am 43(1):53–53
- Matsuda S, Nitoh T (1972) Flare as applied to photographic lenses. Appl Opt 11(8):1850–1856
- Seibert JA, Nalcioglu O, Roeck W (1985) Removal of image intensifier veiling glare by mathematical deconvolution techniques. Med Phys 12(3):281–288
- Talvala EV, Adams A, Horowitz M, Levoy M (2007) Veiling glare in high dynamic range imaging. ACM Trans Graph (TOG), 26(3):37
- Reinhard E, Heidrich W, Debevec P, Pattanaik S, Ward G, Myszkowski K (2010) High dynamic range imaging: acquisition, display, and image-based lighting. Morgan Kaufman, Burlington
- McCann JJ, Rizzi A (2007) Veiling glare: the dynamic range limit of hdr images. In: Proceedings-SPIE the international society for optical engineering. SPIE 649213, vol 6492, San Diego
- Faulkner K, Kotre C, Louka M (1989) Veiling glare deconvolution of images produced by x-ray image intensifiers. In: Third international conference on image processing and its applications, Coventry, IET. pp 669–673
- Hara T, Saito H, Kanade T (2009) Removal of glare caused by water droplets. In: Conference for visual media production, London, pp 144–151
- Koreban F, Schechner Y (2009) Geometry by deflaring. In: IEEE international conference on computational photography (ICCP), San Francisco, pp 1–8

Light Field

Shing Chow Chan

Department of Electrical and Electronic Engineering,
The University of Hong Kong, Hong Kong, China

Synonyms

Image-based rendering (IBR)

Related Concepts

► Lumigraph; ► Plenoptic Function; ► Radiance

Definition

The light field is a function that describes the amount of light in radiance along light rays traveling in every direction through every point in empty space.

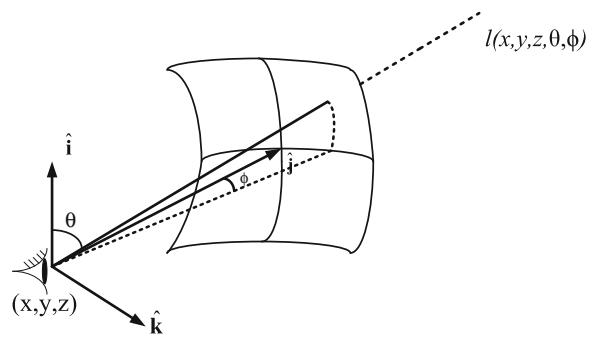
Background

The term light field was coined in a paper by A. Gershun [6] for studying surface illumination with artificial lightings. A similar concept was introduced to the computer graphics community as the light field in [8] and lumigraph in [7]. The motivation is to render new views or images of objects or scenes from densely sampled images previously taken to avoid building or capturing complicated 3D models. Light field or lumigraph rendering is a special representation of image-based rendering (IBR), and they require either no geometry [8] or limited geometry in terms of depth maps [7]. The light field and lumigraph are also four dimension (4D) simplification of the plenoptic function for static scenes.

Theory

In geometric optics, light propagation is modeled as rays which travel in straight lines in free space. The amount of light along a ray is measured in radiance which is the power transmitted per unit area perpendicular to the direction of travel, per unit solid angle. It is denoted by l in watts (W) per steradian (sr) per meter squared (m^2). The function that describes the radiance along light rays traveling in every direction through every point is called the plenoptic function.

A ray in free space can be parameterized by three coordinates, x , y , and z and the elevation (θ) and azimuth (ϕ) angles as shown in Fig. 1. Therefore, for a static scene, the radiance of all the light rays observed is a 5D function $l(x, y, z, \theta, \phi)$, which is a special case of the seven-dimensional plenoptic function which also includes time and wavelengths. Here, it is assumed



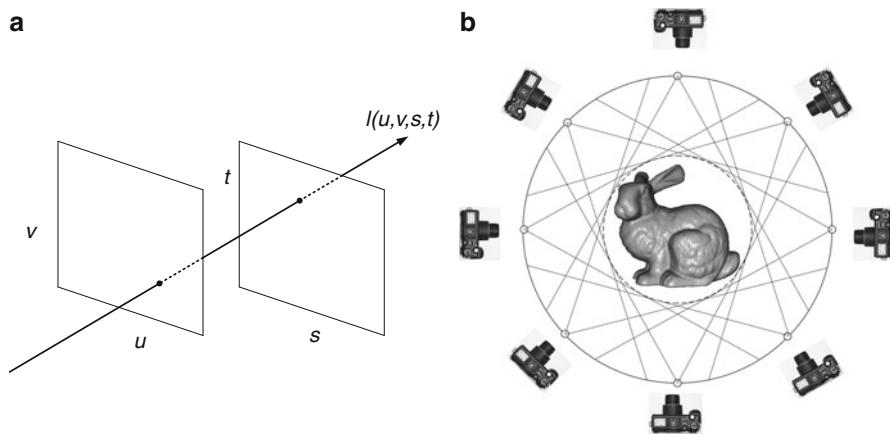
Light Field, Fig. 1 Light field describes the amount of light in radiance along light rays traveling in every direction through every point in empty space

that only the three color components, instead of all the wavelengths, are of interest, and hence the dimension of wavelengths in the plenoptic function is dropped.

If $l(x, y, z, \theta, \phi)$ can be captured by an imaging device, it is possible to select the required rays from $l(x, y, z, \theta, \phi)$ to obtain a projected image at any location (x, y, z, θ, ϕ) as if a picture is taken at this location. Very often, it is the exterior of an object which is of interest, and therefore the viewing locations can be restricted to the convex hull of the object. In this case, the plenoptic function can be captured by a digital camera, which captures the rays at the viewing locations. Since the radiance along rays in free space remain constant, this one-dimensional redundancy in the plenoptic function allows us to reduce it to the 4D light field or lumigraph. The light field concept can be similarly extended to time-varying or dynamic scenes, which results in a 5D plenoptic function.

Parameterization

The collection of light rays in a 4D static light field can be parameterized in a number of ways as shown in Fig. 2. A commonly used parameterization is the two-plane parameterization, where a light ray in the light field is parameterized as its intersections or coordinates with two parallel planes, which are denoted by (u, v) and (s, t) . Usually, one of the plane, say the (u, v) plane, is chosen as the camera plane where images are to be taken. All the rays passing through a given point in the (u, v) plane will then correspond to an image taken by a camera at that location, which can be parameterized by the image coordinate (s, t) . Therefore, (s, t) is usually called the image plane, or vice versa.



Light Field, Fig. 2 (a) 2-plane and (b) spherical parameterization of light fields

Both the light field in [8] and the lumigraph [7] use the parallel plane parameterization due to its simplicity. Other possible parameterizations include intersections with surface of a sphere or the intersections between a plane/sphere with a convex surface. These parameterizations can be further simplified by restricting the camera locations to line, line segments, circle, circular arc, etc. to reduce hardware complexity of the capturing system. This gives rise to a wide range of image-based representations specified by different camera geometry. See [19] for a review up to 2007 and a quick summary in the image-based rendering and plenoptic function sections.

For time-varying or dynamic scenes, similar parameterization can be employed. This can be done by an array of video cameras or specially design capturing devices. It can be shown from the sampling analysis of light field [3] that geometry information can be employed to reduce the sampling rate of light field. Therefore, recent research has focused on the estimation of geometry of objects using stereo or multiview vision techniques, special depth sensing devices and lighting techniques as in photometric stereo. This has been shown to improve significantly the rendering quality as illustrated in lumigraph, pop-up light [18] and object-based plenoptic videos, etc. If complete 3D model can be reconstructed, interactive relighting is also possible [24].

Creating/Capturing Light Fields

Light fields can be created by rendering 3D models using computer graphic techniques or captured

naturally using imaging devices such as cameras. The former can be used to record densely spaced images of a very complicated scene rendered off-line by computer graphic techniques. New novel views can then be rendered in real-time using the recorded image-based representations. In [21], real-time relighting of panoramas has also been demonstrated. For real and static scenes, it can be captured by placing a digital still camera at the desired locations of the image-based representations using a robot arm or at general locations using unstructured lumigraph rendering [2]. The light field can also be captured using specially designed cameras called the plenoptic cameras [1], where an optical element, such as a lenticular lens array [1, 14] or a coded aperture [10, 22], is placed in front of a sensor array to map rays from different directions at a location to neighboring pixels in the sensor array. In other words, the recorded images consist of an array of macro-pixels; each contains a set of neighboring pixels recording rays from a given set of directions. By integrating appropriately these 4D samples in a light field, one can approximate the view that would be captured by a camera having a finite aperture. In the handheld light field camera [14], an array of microlens is placed in front of the sensors of a handheld digital camera, and photographs can be refocused after they are taken, a process called synthetic aperture photography. In principle, a light field video can be obtained in a similar manner.

To achieve a larger disparity, multiple camera systems are frequently used, especially in dynamic scenes. Much research effort has been devoted to the construction of 2D camera arrays (see the section of

plenoptic function for more references in capturing systems). To simplify the capturing hardware, light field captured on line segments and circular arc have also been reported.

Sampling

An important problem in capturing, display, and rendering light fields is “how many samples of the plenoptic function and how much geometrical and textural information are needed to generate a continuous representation of the plenoptic function?” This problem was first studied for the light fields in [3] using the concept of Fourier transform, which provides valuable insight in the design of IBR systems and 3D display.

Assuming (u, v) and (s, t) are respectively the image and camera planes as in the lumigraph [7, 19]. Let $l(u, v, s, t)$ represent the continuous light field, $p(u, v, s, t)$ the pattern in sampling the light field, and $r(u, v, s, t)$ the combined filtering and interpolating low-pass filter. The output image after reconstruction is then given by

$$i(u, v, s, t) = r(u, v, s, t) * [l(u, v, s, t)p(u, v, s, t)], \quad (1)$$

where $*$ represents the convolution operation. Taking the Fourier transform of (Eq. 1), one gets

$$I(u, v, s, t) = R(u, v, s, t)[L(u, v, s, t) * P(u, v, s, t)], \quad (2)$$

where I , R , L , and P denote, respectively, the Fourier transforms of $i(u, v, s, t)$, $r(u, v, s, t)$, $l(u, v, s, t)$, and $p(u, v, s, t)$. It is important to study how the spectrum $L(u, v, s, t)$ varies with the geometry of the scene, from which the minimum sampling density and the reconstruction filter $r(u, v, s, t)$ can be determined. For Lambertian surfaces, the radiance received at the camera position (s, t) is given by

$$l(u, v, s, t) = l\left(u - \frac{fs}{z(u, v, s, t)}, v - \frac{ft}{z(u, v, s, t)}, 0, 0\right) \quad (3)$$

with Fourier transform $L(\Omega_u, \Omega_v, \Omega_s, \Omega_t) = \int_{\mathbf{x} \in R^4} l(u, v, s, t) e^{-j\mathbf{\Omega}^T \mathbf{x}} d\mathbf{x}$ where $\mathbf{x} = (u, v, s, t)$ and $\mathbf{\Omega} = (\Omega_u, \Omega_v, \Omega_s, \Omega_t)$ and f is the focal length. For rectangular sampling, lattice, $p(u, v, s, t) =$

$\sum_{n_1, n_2, n_3, n_4 \in Z} \delta(u - n_1 \Delta u) \delta(v - n_2 \Delta v) \delta(s - n_3 \Delta s) \delta(t - n_4 \Delta t)$ where $\delta(\cdot)$ is the Dirac delta function and Δu , Δv , Δs , and Δt are the corresponding sampling intervals. The sampled light field $l_S(u, v, s, t)$ is

$$l_S(u, v, s, t) = l(u, v, s, t) \sum_{m_1, m_2, m_3, m_4 \in Z} \delta(u - n_1 \Delta u) \times \delta(v - n_2 \Delta v) \delta(s - n_3 \Delta s) \delta(t - n_4 \Delta t), \quad (4)$$

and its Fourier transform is

$$L_S(\mathbf{\Omega}) = \sum_{m_1, m_2, m_3, m_4 \in Z} L\left(\Omega_u - \frac{2\pi m_1}{\Delta u}, \Omega_v - \frac{2\pi m_2}{\Delta v}, \Omega_s - \frac{2\pi m_3}{\Delta s}, \Omega_t - \frac{2\pi m_4}{\Delta t}\right). \quad (5)$$

It can be seen that $L_S(\mathbf{\Omega})$ consists of replicas of the original spectrum, which are shifted to the 4D grid points $(\frac{2\pi m_1}{\Delta u}, \frac{2\pi m_2}{\Delta v}, \frac{2\pi m_3}{\Delta s}, \frac{2\pi m_4}{\Delta t})$. For scene with a constant depth z_0 , it is noted from (Eq. 3) that $l(u, v, s, t)$ is a shifted version of the image $l(u, v, 0, 0) = l'(u, v)$ at $(s, t) = (0, 0)$ by an amount $(-fs/z_0, -ft/z_0, 0, 0)$. Consequently, $L(\mathbf{\Omega})$ is simplified to

$$L(\mathbf{\Omega}) = 4\pi^2 L'(\Omega_u, \Omega_v) \delta\left(\frac{f}{z_0} \Omega_u, \Omega_s\right) \delta\left(\frac{f}{z_0} \Omega_v, \Omega_t\right), \quad (6)$$

where $L'(\Omega_u, \Omega_v)$ is the 2D Fourier transform of $l'(u, v)$. Equation (Eq. 6) suggests that the projection of $L(\mathbf{\Omega})$ on the (Ω_v, Ω_t) (Ω_u, Ω_s) plane lies on a straight line passing through the origin with a slope of $-z_0/f$ (Fig. 3a). After sampling, the spectrum will be shifted to grid points $(\frac{2\pi m_1}{\Delta u}, \frac{2\pi m_2}{\Delta v}, \frac{2\pi m_3}{\Delta s}, \frac{2\pi m_4}{\Delta t})$ (Fig. 3b). If z_{\min} and z_{\max} denote the minimum and maximum depth values of the scene, then the spectrum will be bounded by the two lines with slopes $-z_{\min}/f$ and $-z_{\max}/f$, respectively, as shown in Fig. 3c. From Fig. 3b, the minimum interval, by which the replicas of spectral support can be packed without overlapping or aliasing, is $2\pi K_{\Omega_v} f (\frac{1}{z_{\min}} - \frac{1}{z_{\max}})$, where K_{Ω_v} is the maximum frequency of the light field in v plane, which depends on the maximum frequency of texture variations B_v and the resolutions of the sampling camera $1/\Delta v$ and the rendering camera $1/\delta v$.

as $K_{\Omega_v} = \min(B_v, 1/(2\Delta v), 1/(2\delta v))$. The maximum camera spacing is thus $\Delta t_{\max} = 1/(2\pi K_{\Omega_v} f(\frac{1}{z_{\min}} - \frac{1}{z_{\max}}))$. To reduce the sampling rate and avoid dense sampling, the value of K_{Ω_v} can be decreased by reducing B_v and $1/(2\Delta v)$, through pre-filtering the light field images, to the desired rendering resolution. Similar results apply to the (Ω_u, Ω_s) plane. A more thorough discussion on the effect of geometry on this minimum sampling density or rate can be found in [3, 19]. It was found that the sampling rate can be reduced by decomposition of the light fields into depth layers.

Ideally, if the depth map can be computed from stereo or multiview matching algorithm, then the number of images required can be significantly reduced. Despite considerable advances in depth estimation algorithm, finding correspondences in multiple images is still fundamentally limited by textureless region, reflections, and large variations of the imaging content itself. In pop-up light field [18], a sparse light field is modeled using a set of coherent layers with each layer being a collection of corresponding planar regions in the light field images with the help of an easy to use user interface. The number of layers, segmentation, and matting are done with the help of user interface so that the user can supply the information needed for refining the above processes. Each coherent layer can then be rendered free of aliasing by itself or with other background layers.

The effects of occlusion, and lighting and reflection on light field spectrum can be found in [4, 16].

Compression

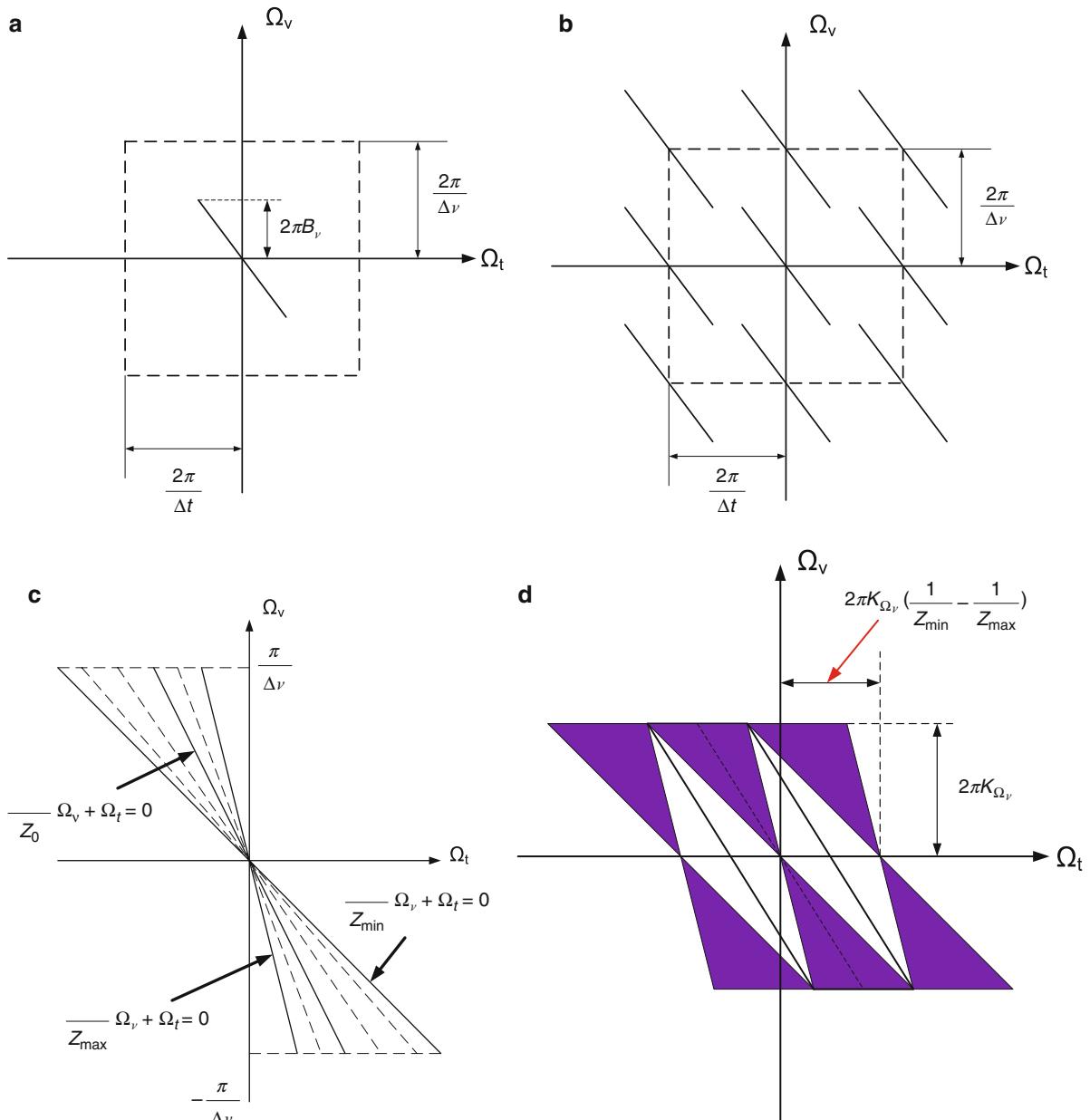
Since adjacent light field images appear to be shifted relative to each other, there is considerable redundancy in the 4D data set, which can be compressed to save storage and transmission bandwidth. Earlier approaches on light field or lumigraph compression were mostly based on conventional pixel-based methods such as vector quantization (VQ) [8, 20], discrete cosine transform (DCT)-based coding, and wavelet coding. Subsequently, disparity compensation prediction (DCP) [11], model-based/model-aided, and object-based methods were proposed to improve the compression ratio and rendering quality. In DCP, the light field images is divided into I- (intra) and P- (predicted) pictures, and the P-pictures can be predicted by disparity compensation from the nearest encoded

I-pictures, which are evenly distributed. In [23], a multiple reference frame (MRF) prediction coding method was proposed for lumigraph compression. Anchor frames (A frames) which are similar to the I-pictures are used as reference for predicting the remaining P-images, and a two-level index table is incorporated into the bit stream for quick access to individual picture and macroblocks. This helps to simplify the “random access problem” to improve rendering speed (see the image-based rendering section for more details and references). A caching stream is also incorporated to speed up the rendering.

Model-based coding makes use of the scene geometry to convert the images from a spherical light field to view-dependent texture maps, which are more amenable to coding using a wavelet-based set partitioning in hierarchical trees (SPIHTs) 4D codec. Model-aided predictive coding makes use of geometry information to morph and predict new views from already encoded images, and the prediction residuals are encoded using DCT-based coding. Experimental results showed that the model-aided approach is more robust to variations in the geometric models [12]. In object-based light field compression [15], the light field images are segmented into image-based objects, or IBR objects in short, each with its image sequences, depth maps, and other relevant information such as shape information. They are compressed using an MPEG-4-like compression algorithm so as to provide content-based functionalities such as scalability of contents, error resilience, and interactivity with individual IBR objects. Rendering scene with depth layers or IBR objects also reduce the artifacts due to depth discontinuities. For more information of IBR and light field compression, see also the survey [5, 19].

Application

Like the plenoptic function, light field serves an important concept for describing visual information in our world. Its sampling analysis also serves as a basic for designing light field cameras and automultiscopic displays for capturing and displaying such high-dimensional function, respectively. Light fields have also found many other applications in computational photography, microscope [9], glare reduction in camera [17], 3DTV [13], etc.



Light Field, Fig. 3 Spectral support of light field: (a) spectrum of a light field with constant depth on the (v, t) plane, (b) spectrum after sampling, and (c) spectrum of a light field signals with

depth bounded by z_{\min} and z_{\max} . The mean depth is z_0 ; (d) Optimal reconstruction filter (*black parallelogram* in the middle) and optimal spacing of spectrum without aliasing

Open Problems

Since the light field is a special case of the plenoptic function, most of their open problems are similar. For instance, the efficient capturing and processing of light fields are important problems in visual computing and vision research. A detailed analysis of the

light field for complicated scene is difficult because the function itself may not even be bandlimited. A general analysis involving all these components is again difficult. Moreover, how to achieve high-quality rendering and display light fields with a wide range of viewpoints in large-scale environmental modeling remains open.



References

1. Adelson EH, Wang JYA (1992) Single lens stereo with plenoptic camera. *IEEE Trans Pattern Anal Mach Intell* 14(2):99–106
2. Buehler C, Bosse ML, McMillan L, Gortler S, Cohen M (2001) Unstructured lumigraph rendering. Proceeding of ACM SIGGRAPH, Los Angeles, 425–432
3. Chai JX, Tong X, Chan SC, Shum HY (2000) Plenoptic sampling. In: Proceedings of ACM SIGGRAPH, New Orleans, 307–318
4. Durand F, Holzschuch N, Soler C, Chan E, Sillion FX (2005) A frequency analysis of light transport. In: Proceedings of ACM SIGGRAPH, Los Angeles, 1115–1126
5. Flierl M, Girod B (2007) Multiview video compression: exploiting inter-image similarities. *IEEE Signal Process Mag* 24(6):66–76
6. Gershun A (1939) The light field. Moscow, 1936. Translated by Moon P, Timoshenko G in *J Math Phys* XVIII:51–151
7. Gortler SJ, Grzeszczuk R, Szeliski R, Cohen MF (1996) The lumigraph. In: Proceedings of ACM SIGGRAPH, New Orleans, 43–54
8. Levoy M, Hanrahan P (1996) Light field rendering. In: Proceedings of ACM SIGGRAPH, New Orleans, 31–42
9. Levoy M, Ng R, Adams A, Footer M, Horowitz M (2006) Light field microscopy. In: Proceedings of ACM SIGGRAPH, Boston, 924–934
10. Liang CK, Lin TH, Wong BY, Liu C, Chen HH (2008) Programmable aperture photography: multiplexed light field acquisition. *ACM Trans Graph* 27(3):55:1–55:10
11. Magnor M, Girod B (2000) Data compression for light-field rendering. *IEEE Trans Circuits Syst Video Technol* 10(3):338–343
12. Magnor M, Ramanathan P, Girod B (2003) Multi-view coding for image-based rendering using 3-D scene geometry. *IEEE Trans Circuits Syst Video Technol* 13(11):1092–1106
13. Matusik W, Pfister H (2004) 3D TV: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes. *ACM Trans Graph* 23(3):814–824
14. Ng R, Levoy M, Bredif M, Duval G, Harowitz M, Hanrahan P (2005) Light field photography with a hand-held plenoptic camera. Stanford University Computer Science Tech Report CSTR 2005–02
15. Ng KT, Wu Q, Chan SC, Shum HY (2010) Object-based coding for plenoptic videos. *IEEE Trans Circuits Syst Video Technol* 20(4):548–562
16. Ramamoorthi R, Mahajan D, Belhumeur P (2007) A first-order analysis of lighting, shading, and shadow. *ACM Trans Graph* 26(1):2:1–2:21
17. Raskar R, Agrawal A, Wilson C, Veeraraghavan A (2008) Glare aware photography: 4D ray sampling for reducing glare effects of camera lenses. *ACM Trans Graph* 27(3):56:1–56:10
18. Shum HY, Sun J, Yamazaki S, Li Y, Tang CK (2004) Pop-up light field: an interactive image-based modeling and rendering system. *ACM Trans Graph* 23(2):143–162
19. Shum HY, Chan SC, Kang SB (2007) Image-based rendering. Springer, New York
20. Tong X, Gray RM (2003) Interactive rendering from compressed light fields. *IEEE Trans Circuits Syst Video Technol* 13(11):1080–1091
21. Wong T, Fu C, Heng P, Leung C (2002) The plenoptic illumination function. *IEEE Trans Multimed* 4(3):361–371
22. Veeraraghavan A, Raskar R, Agrawal A, Mohan A, Tumblin J (2007) Dappled photography: mask enhanced cameras for heterodyned light fields and coded aperture refocusing. *ACM Trans Graph* 26(3):69:1–69:12
23. Zhang C, Li J (2000) Compression of Lumigraph with multiple reference frame (MRF) prediction and just-in-time rendering. In: Proceedings of IEEE data compression conference, Snowbird, 254–263
24. Zhou K, Hu Y, Lin S, Guo B, Shum HY (2005) Precomputed shadow fields for dynamic scenes. In: Proceedings of ACM SIGGRAPH, Los Angeles, 1196–1201

Light Transmission and Reflection Coefficients

► [Fresnel Equations](#)

Line Drawing Labeling

Kokichi Sugihara

Graduate School of Advanced Mathematical Sciences,
Meiji University, Kawasaki, Kanagawa, Japan

Synonyms

[Interpretation of line drawings](#); [Object recognition](#); [Picture understanding](#)

Definition

Line drawing labeling is an assignment of labels to lines in a line drawing according to the interpretation of the line drawing as a three-dimensional scene.

Background

Machine interpretation of a line drawing as a three-dimensional scene is one of the fundamental problems in computer vision. However, a line drawing is a collection of line segments in the plane and has no direct information about three-dimensional structures. Line drawing labeling is an effective method to find candidates of interpretations as three-dimensional scenes. Hence, this method is used as the first step

for interpreting a line drawing. Once the labeling is obtained, the associated candidate of interpretation is analyzed in details to judge whether it is a correct interpretation.

Theory

The most basic version of line drawing labeling is an interpretation of a line drawing as a polyhedral scene viewed from a general angle. This framework is based on the next four assumptions.

Assumption 1. A scene is a collection of opaque polyhedrons.

Assumption 2. The view point is in general position in the sense that no accidental alignment happens in the picture plane (for example, nonparallel edges do not fall into parallel lines, or noncollinear edges do not fall into collinear lines).

Assumption 3. Every vertex of a polyhedron is incident to exactly three faces.

Assumption 4. Visible edges only are drawn in the line drawing.

Under these assumptions, lines in the line drawing are classified into three categories.

1. Convex line. An image of an edge forming a ridge whose two side faces are both visible; this line is represented by “+” label.
2. Concave line. An image of an edge forming a valley; this line is represented by “–” label.
3. Silhouette line. An image of an edge forming a ridge, one of whose side faces is invisible; this line is represented by an arrow label so that the associated faces are to the right of the arrow.

Then, Huffman [1] showed that the allowed combinations of labels around junctions are restricted to only 12 types shown in Fig. 1. Clowes [2] found essentially the same property with different notations.

An assignment of the three labels, one label to each line, is called a *consistent labeling* if the resulting combinations around the vertices all belong to Fig. 1. A line drawing is called *labelable* if it admits a consistent labeling. Then, the next property holds.

Property 1. If a line drawing does not admit a consistent labeling, it cannot represent a three-dimensional scene under Assumptions 1–4.

For example, the line drawing in Fig. 2a does not admit a consistent labeling, and hence it cannot represent a polyhedral scene. On the other hand, a consistent labeling sometimes corresponds to a correct interpretation as in Fig. 2b, but sometimes does not correspond to a correct interpretation as in Fig. 2c.

To find a consistent labeling is in general a hard problem; actually it was proved to be NP-hard by Kirousis and Papadimitriou [3]. On the other hand, for many line drawings observed in ordinary scenes, consistent labeling can be found efficiently by a constraint propagation method (Mackworth [4]) and a relaxation method (Rosenfeld et al. [5]).

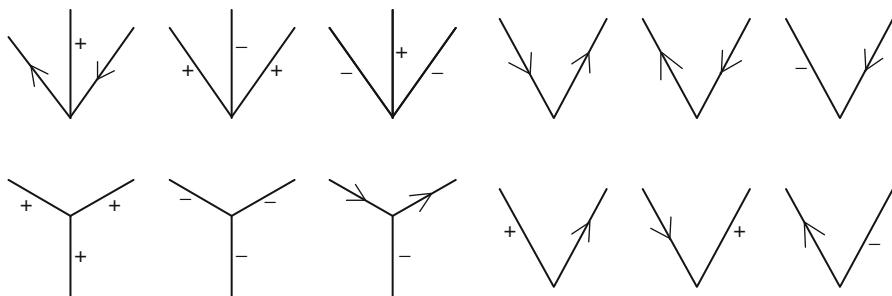
The list of allowed combinations of labels in Fig. 1 is called a *junction dictionary*; this is because a line drawing is a language to convey solid information, and the list in Fig. 1 can be considered as a dictionary to understand this language.

The junction dictionary in Fig. 1 is obtained under Assumptions 1–4. If some of the assumptions are changed, different sets of allowed combinations are obtained. Waltz [6] constructed a junction dictionary for line drawings with shadow lines and cracks. Sugihara [7] constructed a dictionary for engineering line drawings in which hidden lines are represented by broken lines. Kanade [8] constructed a dictionary for origami world in which the objects are made of thin sheets such as paper. Sugihara [9] considered a junction dictionary for range image in which the line categories are known from range data, and hence the dictionary is used for predicting missing edges.

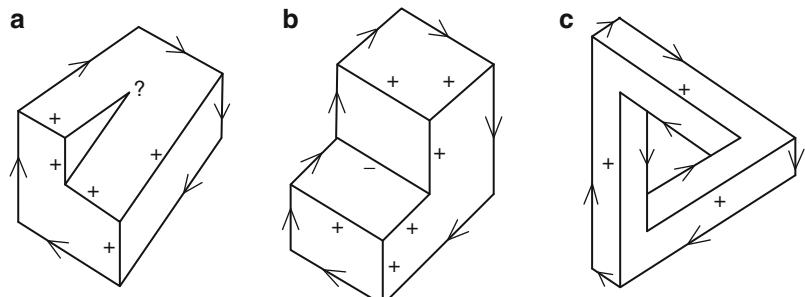
Open Problems

It is known to be NP-hard to find a consistent labeling in general. It is natural to ask what subclasses of pictures can admit polynomial-time algorithms, but it is still open.

Figure 1 is the junction dictionary for the simplest world, defined by Assumptions 1–4. If we change the assumptions, the associated junction dictionary also changes. It is open to construct the junction dictionaries for other worlds and to evaluate their performance. Important worlds include, to mention a few, the world of the three-view drawings used in engineering, and the world with curved surface objects.



Line Drawing Labeling, Fig. 1 Allowed combinations of labels around junctions



Line Drawing Labeling, Fig. 2 Labelable and unlabelable line drawings

Application

A consistent labeling gives a candidate of interpretation of a line drawing as a polyhedral scene. Usually, the set of all consistent labelings is small, and hence the line drawing labeling can effectively prune the possible candidates of interpretation for more detailed analysis. Thus, it can be a first step for machine interpretation of line drawings. A method for judging whether a consistent labeling corresponds to a correct interpretation was established by Sugihara [10].

6. Waltz D (1975) Understanding line drawings of scenes with shadows. In: Winston PH (ed) The psychology of computer vision, McGraw-Hill, New York, pp 19–91
7. Sugihara K (1978) Picture language for skeletal polyhedra. Comput Graph Image Process 8:382–405
8. Kanade T (1980) A theory of Origami world. Artif Intell 13:279–311
9. Sugihara K (1979) Range-data analysis guided by a junction dictionary. Artif Intell 12:41–69
10. Sugihara K (1986) Machine interpretation of line drawings. MIT, Cambridge

References

1. Huffman DA (1971) Impossible objects as nonsense sentences. In: Meltzer B, Michie D (eds) Machine intelligence 6. Edinburgh University Press, Edinburgh, pp 295–323
2. Clowes MB (1971) On seeing things. Artif Intell 2:79–116
3. Kirousis LM, Papadimitriou CH (1988) The complexity of recognizing polyhedral scenes. J Comput Syst Sci 37:14–38
4. Mackworth AK (1977) Consistency in networks of relations. Artif Intell 8:99–118
5. Rosenfeld A, Hummel RA, Zucker SW (1976) Scene labeling by relaxation operations. IEEE Trans Syst Man Cybern SMC-6:420–433

Linear Programming

Kazuo Murota

Department of Mathematical Informatics, University of Tokyo, Bunkyo-ku, Tokyo, Japan

Synonyms

LP

Definition

A *linear programming problem* (also termed *linear program*) is an optimization problem to minimize or

maximize a linear objective function subject to linear equality/inequality constraints. *Linear programming*, often abbreviated as *LP*, is a methodology initiated by G. Dantzig, J. von Neumann, L. V. Kantorovich, and others in the 1940s. [1–6, 10, 11, 15, 17] It includes:

- Modeling techniques to formulate real-world problems into linear programs,
- Theory about the mathematical structure of linear programs,
- Algorithms for numerically solving linear programs.

Standard Form

For theoretical treatment and software development, it is convenient to use a specific form to describe linear programs. Often adopted for use is:

$$\begin{aligned} \text{Minimize } & c^\top x \\ \text{subject to } & Ax = b \\ & x \geq \mathbf{0}, \end{aligned} \quad (1)$$

called the *standard form*, although the terminology varies in the literature. It should be clear that:

- An instance of the optimization problem is specified by a matrix A and vectors b and c . It is assumed that A is an $m \times n$ matrix, b is an m -dimensional vector, and c is an n -dimensional vector.
- Symbol \top means the transpose, and $c^\top x$ denotes the inner product of c and x , that is, $c^\top x = \sum_{j=1}^n c_j x_j$.
- The inequality between vectors means component-wise inequality; $x \geq \mathbf{0}$ for $x = (x_1, \dots, x_n)$ means $x_j \geq 0$ for $j = 1, \dots, n$.
- The variable x , an n -dimensional vector, is the variable for optimization. It is required to find an *optimal solution* x that minimizes the *objective function* $c^\top x$ under the *equality constraint* $Ax = b$ and the *inequality constraint* $x \geq \mathbf{0}$.

Any linear program can be put in the standard form by introducing auxiliary variables, changing maximization to minimization, etc. The transformation to the standard form, however, is not unique.

Example 1 A linear program of the form

$$\text{Minimize } c^\top x \text{ subject to } Ax \geq b, x \geq \mathbf{0} \quad (2)$$

can be put in the standard form (1) by rewriting $Ax \geq b$ to $Ax - s = b$, $s \geq \mathbf{0}$ with a vector $s = (s_1, \dots, s_m)$

of slack variables s_1, \dots, s_m . Conversely, a linear program in the standard form (1) can be recast into the form of (2) by rewriting $Ax = b$ to $Ax \geq b$, $-Ax \geq -b$. It is mentioned that (2) is often called the canonical form. ■

Example 2 A linear program of the form

$$\text{Maximize } c^\top x \text{ subject to } Ax \leq b \quad (3)$$

can be put in the standard form (1) as follows. First, $Ax \leq b$ is rewritten as $Ax + s = b$, $s \geq \mathbf{0}$. Then, the vector x of free variables, which are free from sign-constraint, is expressed as $x = y - z$ with $y \geq \mathbf{0}$ and $z \geq \mathbf{0}$. Finally, maximization is turned into minimization by changing the sign of the objective function. The resulting linear program is

$$\begin{aligned} \text{Minimize } & -c^\top y + c^\top z \\ \text{subject to } & Ay - Az + s = b, \quad y, z, s \geq \mathbf{0}, \end{aligned}$$

which is in the standard form (1) with A , b , c replaced by

$$\begin{aligned} \tilde{A} &= [A \quad -A \quad I], \quad \tilde{b} = b, \\ \tilde{c} &= [-c^\top \quad c^\top \quad \mathbf{0}^\top]^\top. \end{aligned} \quad (4)$$

Given a linear program, say, in the standard form (1), a vector x is called a *feasible point* or a *feasible solution* if it satisfies all the constraints. In the literature of optimization, “solution” is often used as a synonym of “point,” and therefore, “solution x ” may or may not be a “solution” to the given optimization problem, the latter being termed an *optimal solution*. By definition, an optimal solution minimizes the objective function among all feasible solutions (points). A linear program is said to be *feasible* if it has a feasible solution; otherwise, it is *infeasible*. The set of feasible solutions is called the *feasible region*. For instance, $S = \{x \mid Ax = b, x \geq \mathbf{0}\}$ is the feasible region of the problem (1).

A linear program is said to be *unbounded* if the objective function can become indefinitely small toward $-\infty$; otherwise, it is *bounded*. It is a fundamental fact in linear programming that a bounded problem has an optimal solution, which means that there exists a feasible solution $x^* \in S$ such that



$c^\top x^* = \inf\{c^\top x \mid x \in S\}$, where S denotes the feasible region. Note that this is not the case in nonlinear optimization; for example, no x^* exists such that $\exp(-x^*) = \inf\{\exp(-x) \mid x \geq 0\}$.

Basic Solution

In the standard form (1), it can be assumed, without loss of generality, that the $m \times n$ matrix A has independent rows. This implies that $m \leq n$ and $\text{rank } A = m$. Note $b \in \mathbf{R}^m$, $c \in \mathbf{R}^n$, and $x \in \mathbf{R}^n$, where \mathbf{R} denotes the set of real numbers.

Let B be the set of column numbers that corresponds to a basis of the column vectors of A and N be the set of remaining column numbers; $B \cap N = \emptyset$ and $B \cup N = \{1, \dots, n\}$. The $m \times m$ submatrix of A corresponding to B is denoted as A_B and, similarly, the $m \times (n-m)$ submatrix corresponding to N is A_N . If $B = \{1, \dots, m\}$ and $N = \{m+1, \dots, n\}$, for example, then $A = [A_B \ A_N]$. Accordingly, the vector x is partitioned into two parts: $x = (x_B, x_N)$ with $x_B = (x_i \mid i \in B) \in \mathbf{R}^m$ and $x_N = (x_j \mid j \in N) \in \mathbf{R}^{n-m}$. Variables x_i contained in x_B are *basic variables*, and those in x_N are *nonbasic variables*.

The equality constraint $Ax = b$ can be rewritten as

$$x_B + A_B^{-1} A_N x_N = A_B^{-1} b, \quad (5)$$

which is satisfied by

$$\bar{x} = (\bar{x}_B, \bar{x}_N) = (A_B^{-1} b, \mathbf{0}). \quad (6)$$

A solution \bar{x} induced from a basis B in this manner is called a *basic solution*. The basic solution \bar{x} is feasible if

$$A_B^{-1} b \geq \mathbf{0}. \quad (7)$$

A *basic feasible solution* means a basic solution that is feasible. A basis B is called a *feasible basis* if it satisfies the condition (7). Two different bases can possibly determine the same basic solution (6). This is called *degeneracy*.

The following theorem reveals a combinatorial nature of linear programs. Note that, for a given linear program, there exist only finitely many bases, at most $\binom{n}{m} = n!/(m!(n-m)!)$ in number.

Theorem 1 *For a linear program in (1), the following hold:*

- (1) *A basic feasible solution exists if it has a feasible solution.*
- (2) *A basic optimal solution exists if it has an optimal solution.*

With reference to a basis B , x and c are represented as $x = (x_B, x_N)$ and $c = (c_B, c_N)$. The basic solution corresponding to B is given by (6). It follows from (5) that

$$x_B = A_B^{-1} b - A_B^{-1} A_N x_N. \quad (8)$$

Then

$$\begin{aligned} f &= c^\top x = c_B^\top x_B + c_N^\top x_N \\ &= c_B^\top A_B^{-1} b + (c_N^\top - c_N^\top A_B^{-1} A_N) x_N = \bar{f} + \bar{c}_N^\top x_N, \end{aligned} \quad (9)$$

which is an expression of the objective function as a function in x_N , where

$$\bar{f} = c_B^\top A_B^{-1} b = c_B^\top \bar{x}_B = c^\top \bar{x}, \quad (10)$$

$$\bar{c}_N^\top = c_N^\top - c_N^\top A_B^{-1} A_N. \quad (11)$$

Expressions (8) and (9) are reformulations of the equality constraint and the objective function, respectively, in terms of the nonbasic variable x_N at a basic solution. They are often referred to as the *basic forms*.

With the notation

$$\bar{a}_{ij} = (A_B^{-1} A_N)_{ij}, \quad \bar{b}_i = (A_B^{-1} b)_i \quad (i \in B, j \in N) \quad (12)$$

for the coefficients in (8), and $\bar{b} = (\bar{b}_i \mid i \in B) \in \mathbf{R}^m$, the condition (7) for the feasibility of basic solution \bar{x} can be rewritten as

$$\bar{b}_i \geq 0 \quad (i \in B) \quad [\text{i.e., } \bar{b} \geq \mathbf{0}]. \quad (13)$$

The modified cost vector $\bar{c} = (\bar{c}_B, \bar{c}_N) = (\mathbf{0}, \bar{c}_N) \in \mathbf{R}^n$ is defined using \bar{c}_N in (11).

Theorem 2 *Let B be a feasible basis for a linear program (1). Then the basic solution \bar{x} corresponding to B is an optimal solution if*

$$\bar{c}_j \geq 0 \quad (j \in N) \quad [\text{i.e., } \bar{c}_N \geq \mathbf{0}]. \quad (14)$$

Proof Since $x_N \geq \mathbf{0}$ for any feasible solution, it follows from $\bar{c}_N \geq \mathbf{0}$ and (9) that $f = \bar{f} + \bar{c}_N^\top x_N \geq \bar{f}$. \square

A feasible basis B that satisfies the condition (14) is called an *optimal basis*. Theorem 2 states that an optimal basis induces an optimal solution; note, however, that, in case of degeneracy, not every basis associated with an optimal basic solution meets the condition (14).

Suppose that B is a feasible basis that is not optimal. As the optimality condition (14) fails at B , the set $J = \{j \in N \mid \bar{c}_j < 0\}$ is nonempty. Choose a nonbasic variable x_{j^*} with $j^* \in J$ and consider increasing x_{j^*} from zero to a positive value, while fixing other nonbasic variables to zero. It is expected that the objective function decreases since \bar{c}_{j^*} is negative.

As for the equality constraint, (8) is written componentwise as

$$x_i = \bar{b}_i - \sum_{j \in N} \bar{a}_{ij} x_j \quad (i \in B). \quad (15)$$

In increasing x_{j^*} , the basic variables x_i ($i \in B$) can be determined by (15) to maintain the equality constraint.

For the inequality constraint, define $I = \{i \in B \mid \bar{a}_{ij^*} > 0\}$. To maintain $x_i \geq 0$ for $i \in I$, the value of x_{j^*} must be chosen as $x_{j^*} \leq \bar{b}_i / \bar{a}_{ij^*}$, whereas any nonnegative value of x_{j^*} results in $x_i \geq 0$ for $i \notin I$. Therefore, if $I = \emptyset$, the given linear program is unbounded. Hence, suppose $I \neq \emptyset$. With

$$\xi = \min_i \{\bar{b}_i / \bar{a}_{ij^*} \mid \bar{a}_{ij^*} > 0\} \quad (\geq 0), \quad (16)$$

a new point \hat{x} is defined by

$$\begin{aligned} \hat{x}_{j^*} &= \xi, & \hat{x}_i &= \bar{b}_i - \bar{a}_{ij^*} \xi & (i \in B), \\ \hat{x}_j &= 0 & (j \in N \setminus \{j^*\}), \end{aligned}$$

which is a basic feasible solution associated with another basis

$$\hat{B} = (B \setminus \{i^*\}) \cup \{j^*\},$$

where i^* designates the index i that attains the minimum on the right-hand side of (16). This basis \hat{B} is a neighbor to B in the sense that $|\hat{B} \setminus B| = 1$. It is remarked that $\xi = 0$ means $B \neq \hat{B}$ and $x = \hat{x}$ (degeneracy).

The objective function is nonincreasing, since $\bar{c}_{j^*} < 0$, $\xi \geq 0$, and

$$f(\hat{x}) - f(x) = c^\top (\hat{x} - x) = \bar{c}_{j^*} \xi.$$

It is strictly decreasing if $\xi > 0$ (i.e., in a nondegenerate case).

This is the fundamental idea of the simplex method, which repeats updating bases as above until it finds an optimal basis. So long as no degeneracy occurs, the objective function decreases monotonically, and the minimum value is reached in finite steps. In case of degeneracy, however, it can happen that distinct bases corresponding to the same basic solution are visited infinitely many times – a phenomenon called *cycling*. A judicious choice of j^* and i^* avoids cycling.

Duality

For a linear program in the standard form (1) described by an $m \times n$ matrix A , an m -dimensional vector b , and an n -dimensional vector c , another linear program may be considered:

$$\begin{array}{ll} \text{Maximize} & b^\top y \\ \text{subject to} & A^\top y \leq c. \end{array}$$

This is called the *dual problem*; in contrast, the original problem (1) is referred to as the *primal problem*. The pair of linear programs read as

| | |
|---|---|
| Problem P Minimize $c^\top x$ subject to $Ax = b$ $x \geq \mathbf{0}$ | Problem D Maximize $b^\top y$ subject to $A^\top y \leq c$ |
|---|---|

(17)

Since any linear program can be put in the standard form, the dual of the standard form equivalent to Problem D, which is called the dual of Problem D, may be conceived. Problem D is equivalent to the standard form (1) with A , b , and c replaced by

$$\begin{aligned} \tilde{A} &= [A^\top \quad -A^\top \quad I], & \tilde{b} &= c, \\ \tilde{c} &= [-b^\top \quad b^\top \quad \mathbf{0}^\top]^\top; \end{aligned}$$

see (4) in Example 2 and replace $A \rightarrow A^\top$, $b \rightarrow c$ and $c \rightarrow b$. The dual problem of Problem D is



$$\text{Maximize } c^\top \tilde{y} \text{ subject to } \begin{bmatrix} A \\ -A \\ I \end{bmatrix} \tilde{y} \leq \begin{bmatrix} -b \\ b \\ \mathbf{0} \end{bmatrix},$$

which can be rewritten with $\tilde{x} = -\tilde{y}$ as

$$\text{Minimize } c^\top \tilde{x} \text{ subject to } A\tilde{x} = b, \tilde{x} \geq \mathbf{0}.$$

This coincides with Problem P. In this sense, the linear program dual to the dual problem coincides with the primal problem.

Example 3 For the canonical form, the dual pair is given as

| | |
|--|---|
| $\text{Minimize } c^\top x$ $\text{subject to } Ax \geq b$ $x \geq \mathbf{0}$ | $\text{Maximize } b^\top y$ $\text{subject to } A^\top y \leq c$ $y \geq \mathbf{0}.$ |
|--|---|

■

The *linear programming duality* is stated in the following theorem, where the feasible regions of Problems P and D in (17) are denoted as

$$\begin{aligned} P &= \{x \in \mathbf{R}^n \mid Ax = b, x \geq \mathbf{0}\}, \\ D &= \{y \in \mathbf{R}^m \mid A^\top y \leq c\}. \end{aligned} \quad (18)$$

Theorem 3 For the dual pair of linear programs in (17), the following hold.

- (1) [Weak duality] $c^\top x \geq b^\top y$ for any $x \in P$ and $y \in D$.
- (2) [Strong duality] If $P \neq \emptyset$ or $D \neq \emptyset$, then (By convention, $\inf_{x \in P} = +\infty$ if $P = \emptyset$, and $\sup_{y \in D} = -\infty$ if $D = \emptyset$)

$$\inf\{c^\top x \mid x \in P\} = \sup\{b^\top y \mid y \in D\}. \quad (19)$$

This common value is finite if and only if both P and D are nonempty, and in that case, the infimum and the supremum are attained by some $x \in P$ and $y \in D$, respectively.

Proof (1) For $x \in P$ and $y \in D$, it holds that $y^\top b = y^\top Ax \leq c^\top x$. The essence of this theorem lies in (2), which can be derived from a fundamental fact known as Farkas' lemma. □

An optimality criterion in terms of complementarity can be derived from the duality theorem above. For any $x \in P$ and $y \in D$, it holds that

$$\begin{aligned} c^\top x - b^\top y &= c^\top x - y^\top Ax = (c - A^\top y)^\top x \\ &= \sum_{j=1}^n (c - A^\top y)_j x_j \geq 0. \end{aligned} \quad (20)$$

Here, the inequality “ ≥ 0 ” turns into equality “ $= 0$ ” if and only if the *complementarity condition*:

$$x_j = 0 \text{ or } (A^\top y)_j = c_j \quad \text{for each } j = 1, \dots, n \quad (21)$$

is satisfied. Hence, (21) is sufficient for the optimality of x and y . The necessity follows from Theorem 3(2). Hence, follows the next theorem.

Theorem 4 Assume $x \in P$ and $y \in D$. Then x is optimal for Problem P, and y is optimal for Problem D if and only if they satisfy the complementarity condition (21). ■

Another implication of Theorem 3 is that solving Problem P is equivalent to finding (x, y) that satisfies a system of linear equalities/inequalities:

$$Ax = b, \quad x \geq \mathbf{0}, \quad c^\top x \leq b^\top y, \quad A^\top y \leq c. \quad (22)$$

Thus, the duality theorem transforms the optimization problem to a feasibility problem.

Integrality

Linear programming acquires combinatorial flavors through integrality [15, 16]. A linear program described by integer data (an integer matrix A and integer vectors b and c) may or may not have an integer optimal solution. A major interest in this context is under which condition an integer optimal solution is guaranteed.

An integer matrix is *totally unimodular* if every minor (subdeterminant) is equal to 1, -1, or 0. Each entry of a totally unimodular matrix is 1, -1, or 0.

A typical example of a totally unimodular matrix is the incidence matrix of a (directed) graph. Let $G = (V, E)$ be a directed graph with vertex set V and edge

set E , where no self-loops exist. The *incidence matrix* of G is a matrix such that the row-set is indexed by V , the column-set by E , and

$$(v, e)\text{-entry} = \begin{cases} +1 & (v \text{ is the initial vertex of edge } e) \\ -1 & (v \text{ is the terminal vertex of edge } e) \\ 0 & (\text{otherwise}), \end{cases}$$

where $v \in V$ and $e \in E$.

The following theorem relates the total unimodularity of the coefficient matrix to the integrality of optimal solutions of linear programs. The set of integers is denoted as \mathbf{Z} .

Theorem 5 *Let A be a totally unimodular matrix.*

- (1) *If b is integral, Problem P in (17) has an integral optimal solution $x \in \mathbf{Z}^n$, as long as it has an optimal solution.*
- (2) *If c is integral, Problem D in (17) has an integral optimal solution $y \in \mathbf{Z}^m$, as long as it has an optimal solution.*

Algorithms

Algorithms for solving linear programs are divided into three categories: the *simplex method* [1, 2, 15], the *interior-point method* [8, 9, 12–14, 18, 19], and the *ellipsoid method* [7]. The simplex method, proposed by G. Dantzig, 1947, has long been the standard algorithm. The interior-point method, initiated by N. Karmarkar, 1984, is getting more and more important. The ellipsoid method, due to L. G. Khachiyan, 1979, is of theoretical importance but of little practical value; it is the first polynomial-time algorithm and has deep implications in the theory of combinatorial optimization.

Open Problems

The simplex method, with some pivoting rule, is known to be a finite algorithm, but not a polynomial-time algorithm. A major open problem is to find, if possible, a pivoting rule that renders the simplex method a polynomial-time algorithm.

The ellipsoid method is a polynomial-time algorithm, but it is weakly polynomial in that the running time depends on the size of the input numerical data.

Another major open problem is to design a strongly polynomial-time algorithm, which, by definition, runs in time polynomial in the number of constraints and the number of variables.

References

1. Chvátal V (1983) Linear programming. W.H. Freeman and Company, New York
2. Dantzig GB (1963) Linear programming and extensions. Princeton University Press, Princeton
3. Dantzig GB, Thapa MN (1997) Linear programming 1: introduction. Springer, New York
4. Dantzig GB, Thapa MN (2003) Linear programming 2: theory and extensions. Springer, New York
5. Dorfman R, Samuelson PA, Solow RM (1987) Linear programming and economic analysis (New edition). Dover Publications, New York
6. Ferris MC, Mangasarian OL, Wright SJ (2008) Linear programming with MATLAB. Society for Industrial and Applied Mathematics, Philadelphia
7. Grötschel M, Lovász L, Schrijver A (1993) Geometric algorithms and combinatorial optimization, 2nd edn. Springer, Berlin/New York
8. den Hertog D (1994) Interior point approach to linear, quadratic and convex programming: algorithms and complexity. Kluwer Academic Publishers, Dordrecht/Boston
9. Kojima M, Megiddo N, Noma T, Yoshise A (1991) A unified approach to interior point algorithms for linear complementarity problems. Lecture notes in computer science, vol 538. Springer, Berlin/New York
10. Lasserre J-B (2009) Linear and integer programming vs. linear integration and counting: a duality viewpoint. Springer, New York/London
11. Nemhauser GL, Rinnooy Kan AHG, Todd MJ (1989) Optimization. Handbooks in operations research and management science, vol 1. Elsevier Science Publishers, New York
12. Nesterov Y, Nemirovskii A (1994) Interior point polynomial algorithms in convex programming. Society for Industrial and Applied Mathematics, Philadelphia
13. Renegar J (1987) A mathematical view of interior-point methods in convex optimization. Society for Industrial and Applied Mathematics, Philadelphia
14. Roos C, Terlaky T, Vial J-P (2006) Interior point methods for linear optimization, 2nd edn. Springer, New York
15. Schrijver A (1986) Theory of linear and integer programming. Wiley, Chichester/New York
16. Schrijver A (2003) Combinatorial optimization – polyhedra and efficiency. Springer, Berlin/New York
17. Vanderbei RJ (2008) Linear programming: foundations and extensions, 3rd edn. Springer, New York
18. Wright S (1997) Primal-dual interior-point methods. Society for Industrial and Applied Mathematics, Philadelphia
19. Ye Y (1997) Interior point algorithms: theory and analysis. Wiley, New York



Linguistic Techniques for Event Recognition

Vlad I. Morariu

Computer Vision Laboratory, University of Maryland,
College Park, MD, USA

Definition

Linguistic techniques for event recognition use elements of language such as vocabulary, grammar, expressions, syntax, and semantics to represent and recognize events.

Background

Visual event recognition systems use linguistic techniques to describe complex events by the composition of primitive sub-events according certain rules (spatial, temporal, logical, etc.). This allows systems to represent a large set of events while modeling a much smaller set of primitive events by low-level processing. Representational and computational properties of these systems depend on the types of allowable relationships between sub-events. Some systems also aim for representations that are closer to natural language to facilitate user interaction, i.e., to describe events, make queries, or interpret results.

Theory and Applications

Hidden Markov Models (HMMs) [10] are statistical Markov models that decompose temporal sequences into discrete time steps, each with an associated unobserved discrete state variable and an observed continuous or discrete variable. Each state has a probability distribution over the output variable and another over the transition to a new state in the next time step. These models have been successful in the field of speech recognition and have also been widely used for gesture [7] and event recognition [6, 16]. More general Dynamic Bayesian Networks (DBNs) [3] have also been applied to event recognition (e.g., [8]) and some extensions such as Propagation Nets (P-Nets) model

duration of temporal subintervals [13] for recognition of complex events.

Logical languages such as First-Order Logic have been used to represent qualitative spatial and temporal constraints, with temporal constraints often expressed using Allen's temporal primitives [1]. Allen and Ferguson [2] proposed a framework for using these temporal primitives to create a representation motivated by natural language that is able to reason about complex patterns such as external events, simultaneous activities, and mutually exclusive activities for the purpose of planning and natural language understanding systems. Siskind's formal language of *event logic* [14], which also uses Allen's temporal primitives, reasons about simple spatial verbs observed from animated images. To deal with the computational complexity with Allen's temporal primitives, Pinhanez and Bobick propose Past-Now-Future Propagation (PNF) [9], which constructs a three-valued PNF constraint network designed to answer only the question of whether an action is currently happening or not. Nevatia et al. [5] also use Allen's interval logic for relating event intervals to each other using the Event Recognition Language (ERL). ERL hierarchically represents single and multiple threads of composite events and is intended to simplify user interaction through its similarity to existing programming languages. Tran and Davis more recently [15] combined logical and probabilistic event reasoning via Markov Logic Networks (MLNs), which relax hard rules in First-Order Logic and perform probabilistic inference using Markov networks.

Grammars, which have been used in analyzing static images (e.g., picture languages [11]), have also been used for event recognition. Ivanov and Bobick [4] propose a system that recognizes events using a stochastic context-free grammar (SCFG) parsing mechanism which allows for long range temporal constraints, uncertain low-level detections, as well as concurrency of events. Their approach combines the statistical detection of primitives using uncertain low-level detectors, with long-term structural interpretation by stochastic context-free parsing. Ryoo and Aggarwal [12] combine Bayes Nets at the lowest layer to infer pose, HMMs that the middle layer to extract gesture symbols, and context-free grammars (CFGs) combined with Allen's interval logic to extract composite actions and interactions from primitive (or atomic) actions. Their system was applied to the recognition of two-person interactions.

References

1. Allen JF (1984) Towards a general theory of action and time. *Artif Intell* 23(2):123–154
2. Allen JF, Ferguson G (1994) Actions and Events in Interval Temporal Logic. *J Log Compu* 4(5):531–579
3. Ghahramani Z (1998) Learning dynamic bayesian networks. In: Adaptive processing of sequences and data structures, volume 1387 of lecture notes in computer science. Springer, Berlin/New York, pp 168–197
4. Ivanov YA, Bobick AF (2000) Recognition of visual activities and interactions by stochastic parsing. *IEEE Trans Pattern Anal Mach Intell* 22(8):852–872
5. Nevatia R, Zhao T, Hongeng S (2003) Hierarchical language-based representation of events in video streams. In: Conference on Computer Vision and Pattern Recognition Workshop (CVPRW), vol 4, Madison, Wisconsin, pp 39–39, June 2003
6. Oliver NM, Rosario B, Pentland AP (2000) A bayesian computer vision system for modeling human interactions. *IEEE Trans Pattern Anal Mach Intell* 22(8):831–843
7. Ong SCW, Ranganath S (2005) Automatic sign language analysis: a survey and the future beyond lexical meaning. *IEEE Trans Pattern Anal Mach Intell* 27(6):873–891
8. Park S, JK Aggarwal (2004) A hierarchical bayesian network for event recognition of human actions and interactions. *ACM Multimed Syst J* 10:164–179
9. Pinhanez C, Bobick A (1997) Human action detection using pnf propagation of temporal constraints. In: IEEE conference on computer vision pattern recognition (CVPR), San Juan, pp 898–904
10. Rabiner LR (1989) A tutorial on hidden markov models and selected applications in speech recognition. *Proc IEEE* 77(2):257–286
11. Rosenfeld A, Siromoney R (1993) Picture languages—a survey. *Lang Des* 1(3):229–245
12. Ryoo MS, Aggarwal JK (2006) Recognition of composite human activities through context-free grammar based representation. In: IEEE conference on computer vision pattern recognition (CVPR), New York
13. Shi Y, Bobick A, Essa I (2006) Learning temporal sequence model from partially labeled data. In: IEEE conference on computer vision pattern recognition (CVPR), New York
14. Siskind JM (1994) Grounding language in perception. *Artif Intell Rev* 8:371–391
15. Tran SD, Davis LS (2008) Event modeling and recognition using markov logic networks In: European conference on computer vision (ECCV), Marseille
16. Yamato J, Ohya J, Ishii K (1992) Recognizing human action in time-sequential images using hidden markov model. In: IEEE conference on computer vision pattern recognition (CVPR), Champaign, pp 379–385, June 1992

Local Invariants

► Differential Invariants

LP

► Linear Programming

Lumigraph

Michael F. Cohen and Richard Szeliski
Microsoft Research, One Microsoft Way, Redmond, WA, USA

Synonyms

Light Field

Related Concepts

► Image-Based Rendering

Definition

A Lumigraph (also known as light field [1]; see <http://graphics.stanford.edu/papers/light/> for a good explanation of the history, similarities, and differences between a light field and a Lumigraph.) is a four-dimensional representation of all the light rays passing through an empty volume of space [2]. It can be used to synthesize a novel image from any viewpoint within that space. The Lumigraph often refers to a light field augmented with a partial 3D model of the scene, which is used during the rendering process.

Background

If one considers a volume of empty space, the radiance of all the rays passing through this space can be organized in five dimensions, the three dimensions of space plus two dimensions for direction. This is known as the 5D *plenoptic function* of Adelson and Bergen [3, 4]. However, since radiance is constant along a ray in empty space, the space of all unique rays drops to four dimensions. This four-dimensional set of rays has come to be known as a light field or Lumigraph. An idealized image consists of a two-dimensional array

of pixels, each of which represents the radiance along single ray passing through a single point in space or pinhole. Thus, since a Lumigraph contains all the rays in a given space, this information can depict any image from any vantage point in the space represented by the Lumigraph. More generally, the image created by any lens and aperture embedded in the Lumigraph space can also be reconstructed. Finally, images not realizable by traditional lenses, such as multiperspective panoramas [5], can also be rendered.

These observations led to the development of the Lumigraph. Capturing a Lumigraph and then, after-the-fact, being able to reconstruct any image from any vantage point enable applications for depicting the appearance of a scene or object. Often, the empty space captured in a Lumigraph is all or a portion of the space surrounding the convex hull of an object. This allows the examination of the object from any vantage point. Alternatively, the Lumigraph may capture the space of views through a finite window. This provides the ability to move the viewpoint back and forth to depict any parallax due to varying depths in the scene, or to treat the window as a finite aperture with which to vary the plane of focus or the depth of field.

The interesting technical problems for Lumigraphs include how to:

- Capture the desired subset of rays
- Organize and store the Lumigraph
- Interpolate missing rays from discretely sampled rays
- Efficiently retrieve the visual information to enable real-time image reconstruction from any viewpoint

In the following sections, techniques developed in the original Lumigraph paper [2] are discussed and then generalized to work in the unstructured Lumigraphs of Buehler et al. [6].

Application

Lumigraph Capture

Unfortunately, it is impossible to instantaneously capture a dense 4D continuous field of rays in some volume (See the discussion at the end of this entry on lenslet arrays as one alternative for capturing discrete subsets of the 4D field.). Instead, the space of rays is sampled with a camera or cameras. Each camera image (assuming a pinhole model) captures a discretized 2D slice of the 4D space. Each pixel represents

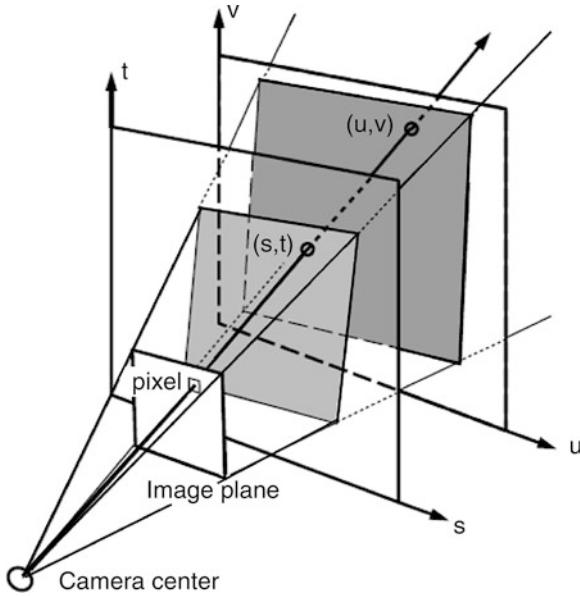
some ray. In the original Lumigraph paper, an object was placed on a special stage with fiduciary marks. A handheld camera attached to a computer was pointed at the object and moved around it to capture a set of images from various vantage points. The fiduciary points allowed the camera's pose to be calculated in real time to provide feedback to the user about what regions of the surrounding sphere had been well sampled. In the concurrent light field paper, the camera was placed on a gantry from which the camera could be carefully maneuvered and the pose could be read off mechanically.

Lumigraph Organization

Each captured image contains an array of pixels representing the radiance values for a subset of rays passing through the aperture of the camera. The set of all rays from all captured images form the discrete information from which to generate any other novel view. Likewise, constructing the novel view involves finding, for each pixel (ray) in the novel view, one or more nearby rays from the captured set from which to reconstruct the desired pixel. To make the reconstruction efficient, the input set of rays may be reorganized to aid the ray lookup process. The original Lumigraph and light field papers used two-plane reparameterizations. The subsequent unstructured Lumigraph [6] work skipped this step, instead rendering directly from the original images. The two-plane parameterization used in the original papers is discussed next.

Consider two parallel planes. It is easiest to start with aligned rectangular sections of each plane, as shown in Fig. 1. The first plane is parameterized using s and t and the other using u and v . Each ray that intersects the two rectangles is associated with a 4D coordinate (s, t, u, v) . Since the rays are directional, the rays are assumed to carry light through the (u, v) plane before passing through the (s, t) plane. Consider a single point (s_0, t_0) on the (s, t) plane and all the rays passing through that point. These rays represent an image one could get by placing the camera's center at (s_0, t_0) oriented toward the (u, v) plane. The (u, v) parameterization characterizes the image *rectified* to the (u, v) plane.

By placing the camera successively at all points on the (s, t) rectangle, a 4D Lumigraph is captured. Of course, any image with rays that pass through the two planes, such as from a camera placed behind the (s, t)



Lumigraph, Fig. 1 The Lumigraph: a ray is represented by its 4D two-plane parameters (s, t) and (u, v) (From [1], The Lumigraph, SIGGRAPH'96, (c) ACM)

plane, can contribute to filling in the Lumigraph. In Fig. 1, a pixel's 4D coordinate (s, t, u, v) is given by its intersection with the two planes. The original Lumigraph and light field papers resampled the rays from such more general input images into the two-plane parameterization.

Completing (Resampling) the Lumigraph

In any practical system, the Lumigraph is a collection of only a finite number of 4D samples. For example, given N^2 images with R^2 resolution, there are at most $N^2 R^2$ pixels representing unique rays in the Lumigraph. One could, for example, carefully place a camera at each $N \times N$ grid point on the (s, t) plane oriented precisely to cover the extent of the (u, v) plane. Unfortunately, even this simple configuration is very difficult to achieve. Instead, one can gather a much less organized set of images and *resample* the pixels into the 4D (s, t, u, v) coordinate system. Typically, the (s, t) plane is more coarsely discretized than the (u, v) plane, as the object being imaged is near the (u, v) plane. The assumption is that the object texture changes more rapidly across its surface than the directional changes in radiance of a single point on the surface.

Each pixel in an input image represents some ray (s, t, u, v) such as the ray shown in Fig. 2a. The question then becomes finding the *closest* discrete grid point to assign the color of this pixel to in the discretized Lumigraph (Fig. 2d–f). The simplest method to determine closest is to choose the closest discrete (s, t) and (u, v) grid points to ray's continuous coordinates. The original Lumigraph paper modifies this simple scheme to leverage depth information in the scene. A rough estimate of depth is obtained from the visual hull of the object being imaged, which is computed from the silhouettes of the object pulled from the blue background stage.

Consider the ray (s, u) in a two-dimensional Lumigraph (Fig. 3). The closest grid point to this ray is (s_{i+1}, u_p) . However, gridpoints (s_{i+1}, u_{p-1}) and (s_i, u_{p+1}) are likely to contain colors closer to the true color of the input ray since these grid points represent rays that intersect the object near the intersection with the ray. Suppose one knows the depth value z at which the ray first intersects the object. For nearby discrete s_i and s_{i+1} values, the corresponding u' and u'' values are computed for rays that intersect the same geometric location,

$$u' = u + (s - s_i)z/(1 - z) \quad (1)$$

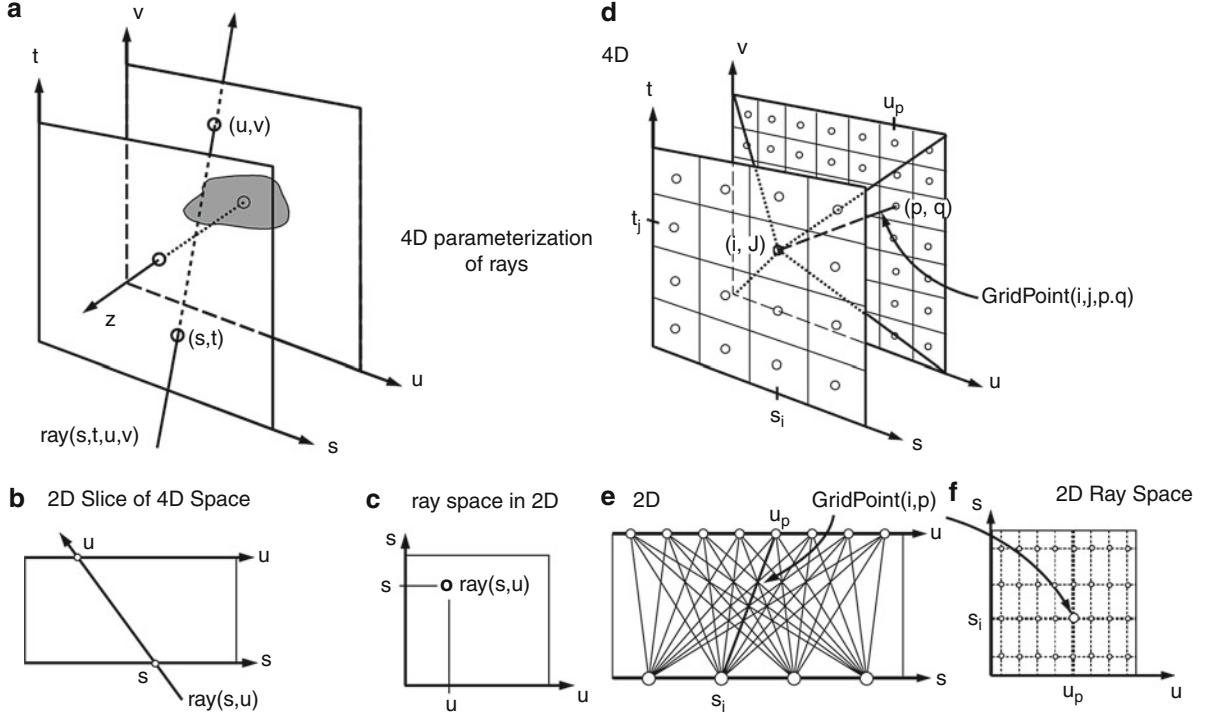
$$u'' = u + (s - s_{i+1})z/(1 - z). \quad (2)$$

As you can see, neither u' nor u'' is closest to the discrete point u_p . This observation is used to define a depth-corrected quadrilinear interpolant for *splatting* input rays into the Lumigraph. The details can be found in Gortler et al. [2].

Finally, even given a large number of input cameras, there are bound to be many discrete samples in the Lumigraph left unfilled. Gortler et al. describe a *push-pull* multiresolution method to fill out the Lumigraph by averaging up the pyramid (pulling) and then pushing values down the pyramid while never overwriting values filled during the *pull*. Figure 4 shows a 2D analog of this process.

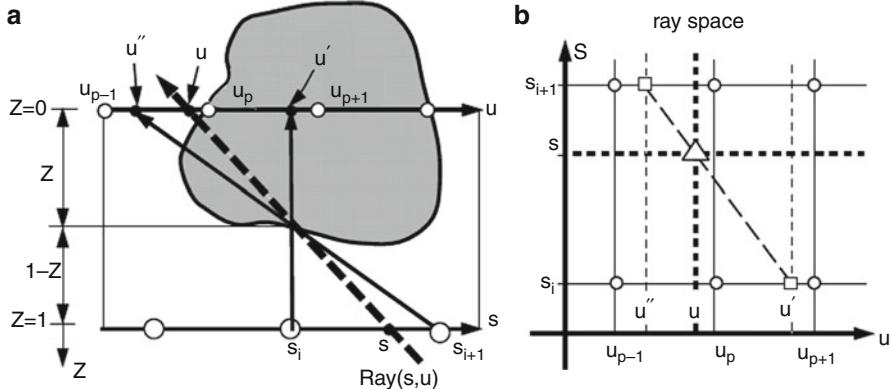
Reconstruction and Display

Rendering a novel view from a Lumigraph operates essentially in reverse of resampling. Each pixel in a desired image corresponds to a 4D ray. If this ray intersects the (s, t) and (u, v) planes, its 4D coordinate is



Lumigraph, Fig. 2 Parameterization and discretization of the Lumigraph: (a) a ray is parameterized by its 4D two-plane parameters (s, t) and (u, v) . The object being imaged is near the (u, v) plane with the distance to any point on the object surface

from the plane given as z ; (b) a 2D slice of the 4D space for illustration purposes; (c) a 2D representation of the “ray space”; (d–f) discretizing the Lumigraph (From [1], The Lumigraph, SIGGRAPH’96, (c) ACM)



Lumigraph, Fig. 3 Determining the closest grid point (From [1], The Lumigraph, SIGGRAPH’96, (c) ACM)

determined by these intersections. The color values stored at the *closest* 4D gridpoints of the Lumigraph are retrieved and interpolated. Again, the depth correction discussed in the creation of the Lumigraph can be used in the definition of *closeness* and thus

the interpolation coefficients. Gortler et al. [2] relate this interpolation to the definition of skewed 4D basis functions for reconstruction and show differing artifacts resulting from different basis function choices. Figure 5 shows reconstructions of two nearby

Lumigraph, Fig. 4 Results of a 2D analog to the 4D push-pull algorithm. *Left* figure shows lines sampled in 2D (analogous to images in 4D). *Right* shows result of push-pull fill-in (From [1], The Lumigraph, SIGGRAPH'96, (c) ACM)



Lumigraph, Fig. 5 A stereo pair of reconstructed images of a Lumigraph of a stuffed lion (Cross-eyed arrangement) (From [1], The Lumigraph, SIGGRAPH'96, (c) ACM)



views (a stereo pair) of a fuzzy lion captured with a handheld camera.

Unstructured Lumigraphs

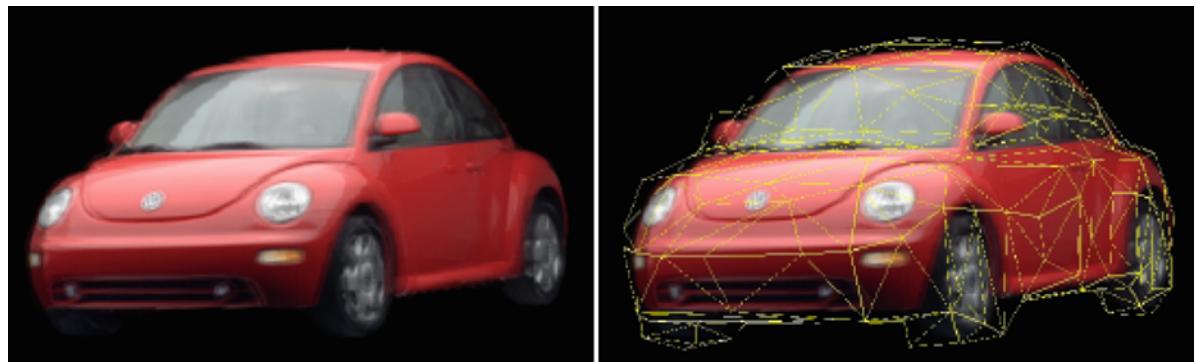
Five years after the original Lumigraph paper, Buehler et al. [6] developed a means to render directly from the original input images without first resampling the pixels/rays into a regular 4D structure. This *unstructured Lumigraph* relies on an approximate 3D tessellated proxy as a means to measure the distance between an input ray and a desired output ray in the image being generated. The distance measure combines epipole consistency (does the desired ray pass near the center of a source camera?), angular deviation (do two rays that meet at the same point on the proxy differ in angle?), continuity (do two rays intersect the proxy at nearby points?), and resolution (do the two rays integrate similar sized areas on the proxy?).

Their paper describes a real-time rendering algorithm that uses graphics hardware to compute barycentric weight contributions from each input camera.

The weights are computed for the union of each tessellated triangle in the proxy with each triangle in a tessellated output image plane. Figure 6 shows a reconstructed image and the tessellated proxy used in weight calculations.

Open Problems

In recent years, new cameras have been developed that contain an array of lenslets that can capture a small light field on a single chip [7]. Such devices support after-capture refocusing, as well as limited viewpoint motions [8]. Applications to microscopy also show great promise [9]. Large arrays of cameras have also been used to enhance both spatial and temporal resolution and to refocus with large virtual apertures to see through occluders [10, 11]. It remains an open problem to find ways to effectively capture large Lumigraphs. Finally, adding a temporal dimension presents significant new issues to overcome.



Lumigraph, Fig. 6 An unstructured Lumigraph reconstruction of a VW bug and its proxy geometry (From [6], Unstructured Lumigraph Rendering, SIGGRAPH 2001 (c) ACM)

References

- Levoy M, Hanrahan P (1996) Light field rendering. In: ACM SIGGRAPH 1996 conference proceedings, ACM SIGGRAPH, New Orleans, pp 31–42
- Gortler SJ, Grzeszczuk R, Szeliski R, Cohen MF (1996) The Lumigraph. In: ACM SIGGRAPH 1996 conference proceedings, ACM SIGGRAPH, New Orleans, pp 43–54
- Adelson EH, Bergen J (1991) The plenoptic function and the elements of early vision. In: Computational models of visual processing, MIT, Cambridge, MA, pp 3–20
- McMillan L, Bishop G (1995) Plenoptic modeling: an image-based rendering system. In: ACM SIGGRAPH 1995 conference proceedings, Los Angeles, pp 39–46
- Rademacher P, Bishop G (1998) Multiple-center-of-projection images. In: ACM SIGGRAPH 1998 conference proceedings, ACM SIGGRAPH, Orlando, pp 199–206
- Buehler C, Bosse M, McMillan L, Gortler SJ, Cohen MF (2001) Unstructured Lumigraph rendering. In: Fiume E (ed) ACM SIGGRAPH 2001 conference proceedings, ACM/ACM SIGGRAPH, Los Angeles, pp 425–432
- Ng R, Levoy M, Brédif M, Duval G, Horowitz M, Hanrahan P (2005) Light field photography with a hand-held plenoptic camera. Technical Report CSTR 2005-02, Stanford University
- Ng R (2005) Fourier slice photography. ACM Trans Graph 24(3):735–744
- Levoy M, Ng R, Adams A, Footer M, Horowitz M (2006) Light field microscopy. ACM Trans Graph 25(3):924–934
- Wilburn B, Joshi N, Vaish V, Talvala EV, Antunez E et al. (2005) High performance imaging using large camera arrays. ACM Trans Graph 24(3):765–776
- Vaish V, Szeliski R, Zitnick CL, Kang SB, Levoy M (2006) Reconstructing occluded surfaces using synthetic apertures: shape from focus vs. shape from stereo. IEEE computer society conference on computer vision and pattern recognition (CVPR'2006), New York, vol 3, pp 2331–2338

M

Machine Recognition of Objects

Tomaso Poggio and Shimon Ullman
Department of Brain and Cognitive Sciences,
McGovern Institute, Massachusetts Institute of
Technology, Cambridge, MA, USA

Related Concepts

► [Object Class Recognition \(Categorization\)](#); ► [Visual Cortex Models for Object Recognition](#)

Definition

Machine recognition of objects is the task of locating and recognizing a given object in an image and consists of the following steps: object detection, feature extraction, and recognition.

Background

Early computer vision recognition schemes focused primarily on the recognition of rigid three-dimensional (3D) objects, such as machine parts, tools, and cars. This is a challenging problem because the same object can have markedly different appearances when viewed from different directions. It proved possible to deal successfully with this difficulty by using detailed 3D models of the viewed objects, which were compared with the projected 2D image (e.g., [14, 18, 33]). Over the last decade or so, computational models have made significant progress in the task of recognizing natural

object categories under realistic, relatively unconstrained viewing conditions. Within object recognition, it is common to distinguish two main tasks: identification, for instance, recognizing a specific face among other faces, and categorization, for example, recognizing a car among other object classes. We will discuss both of these tasks below and use “recognition” to include both.

The qualitative improvement in the performance of recognition models can be attributed to three main components. The first is the use of extensive learning in constructing recognition models. In this framework, rather than specifying a particular model, the scheme starts with a large family of possible models and uses observed examples to guide the construction of a specific model which is best suited to the observed data. The second component was the development of new forms of object representation for the purpose of categorization, based on both computational considerations and guidelines from known properties of the visual cortex. These two components, representation and learning, are interrelated: initially, the class representation provides a family of plausible models, and effective learning methods are then used to construct a particular model for a novel class such as “dog” or “airplane” based on observed examples. The third component was the use of new statistical learning techniques, such as regularization classifiers (SVM and others) and Bayesian inference (such as graphical models). We next discuss each of these advances in more detail.

Learning instead of design. A conceptual advance that facilitated recent progress in object recognition was the idea of learning the solution to a specific

classification problem from examples, rather than focusing on the classifier design. This was a marked departure from the dominant practices at the time: instead of an expert program with a predetermined set of logical rules, the appropriate model was learned and selected from a possibly infinite set of models, based on a set of examples. The techniques used in the 1990s originated in the area of supervised learning, where image examples are provided together with the appropriate class labels (e.g., “face” or “non-face”). A comprehensive theory of the foundations of supervised learning has been developed, with roots in functional analysis and probability theory [6, 26, 27, 36]. The formal analysis of learning continues to evolve and to contribute to our understanding of the role of learning in visual recognition.

New image representations. A recognition scheme typically extracts during learning a set of measurements or “features” and uses them to construct new object representations. Objects are then classified and recognized based on their feature representation. Feature selection and object representation are crucial, because they facilitate the identification of elements that are shared by objects in the same class and support discrimination between similar objects and categories. Different types of visual features have been used in computational models in the past, ranging from simple local-image patterns such as wavelets, edges, blobs, or local-edge combinations to abstract three-dimensional shape primitives, such as cylinders [21], spheres, cubes, and the like [4].

A common aspect of most past recognition schemes is that they use a fixed small generic set of feature types to represent all objects and classes. In contrast, recent recognition schemes use pictorial features extracted from examples, such as object fragments or patches, together with their spatial arrangement [1, 3, 19, 30]. Unlike generic parts, these schemes use a large set of features, extracted from different classes of objects. The use of large feature sets is also connected to an interesting new trend in signal processing, related to “over-complete” representations. Instead of representing a signal in terms of a traditional complete representation, such as Fourier components, one uses a redundant basis (such as the combination of several complete bases).

Representations using such features have been used successfully in recent computer vision recognition

systems for two reasons. First, these representations can be learned and used efficiently; second, they proved to capture effectively the broad range of variability in appearance within a visual class.

An additional comment is appropriate. The representations described above are view based, as opposed to object-centered models. A representation based on image appearance can include not only 2D image properties but also 3D aspects such as local depth variations or 3D curvature.

New statistical learning methods. Over the last few years, the mathematics of learning has become the “lingua franca” of large areas of computer science and, in particular, of computer vision. As we discussed, the use of a learning framework enabled a qualitative jump in object recognition. Whereas the initial techniques used to construct useful classification models from data were quite simple, there are now more efficient algorithms originally introduced in the area of learning in the 1990s such as regularization algorithms (also called kernel machines), which include SVM [35, 36] and boosting [12]. By now, the area of learning has grown to include, in addition to discriminative algorithms, probabilistic approaches with the goal of providing full probability distributions as solutions to object recognition tasks. These techniques are mostly Bayesian and range from graphical models [13, 15] to hierarchical Bayesian models [16, 17]. At the same time, the focus of research is shifting from supervised to unsupervised and semisupervised learning problems, using techniques such as manifold learning [2]. Semisupervised problems, in which the training set consists of a large number of unlabeled examples and a small number of labeled ones, are gaining attention.

Application

A number of early schemes, mainly focusing on the class of human faces, obtained significant improvement over previous methods [5, 29, 31, 32, 38]. The techniques have evolved to reach practical applications, as evidenced by their use in current digital cameras. The more recent versions of these computational schemes have started to deal successfully with an increasing range of complex object

categories such as pedestrians, cars, motorcycles, airplanes, horses, and the like, in unconstrained natural scenes, to deal with a broad range of objects within each class (e.g., [1, 8, 19, 22–24, 30, 34, 39]). The algorithms that were refined over the last few years can deal successfully with a large number of different object classes, in complex and highly cluttered scenes. They are being applied to databases of hundreds [9] and even thousands of object classes [7]. Yearly competitions in computer-based recognition, such as the Pascal challenge [25, 28], witness continuous improvement in the range of classes and in scene complexity successfully handled by automatic object categorization algorithms [10, 11, 37].

References

1. Agarwal S, Roth D (2002) Learning a sparse representation for object recognition. In: Proceedings of the 7th ECCV, Copenhagen, pp 113–130
2. Belkin M, Niyogi P (2004) Semi-supervised learning on Riemannian manifolds. *Mach Learn J* 56:209–239
3. Belongie S, Malik J, Puzicha J (2002) Shape matching and object recognition using shape contexts. *IEEE PAMI* 24(4):509–522
4. Biederman I (1985) Human image understanding: recent research and a theory. *Comput Vis Graph Image Process* 32:29–73
5. Brunelli R, Poggio T (1993) Face recognition: features versus templates. *IEEE Trans PAMI* 15(10): 1042–1052
6. Cucker F, Smale S (2002) On the mathematical foundations of learning. *Bull Am Math Soc* 39:1–50
7. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) ImageNet: a large-scale hierarchical image database. In: IEEE computer vision and pattern recognition (CVPR), Miami
8. Fei-Fei L, Fergus R, Perona P (2003) A Bayesian approach to unsupervised one-shot learning of object categories. In: Proceedings of the ICCV, Wisconsin
9. Fei-Fei L, Fergus R, Perona P (2004) Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In: IEEE conference on computer vision pattern recognition (CVPR 2004), workshop on generative-model based vision, Washington, DC
10. Felzenszwalb D, McAllester D, Ramanan A (2008) Discriminatively trained, multiscale, deformable part model. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), Anchorage
11. Felzenszwalb PF, Girshick RB, McAllester D, Ramanan D (2009) Object detection with discriminatively trained part based models. *IEEE Trans Pattern Anal Mach Intell* 32:1627–1645
12. Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. *J Comput Syst Sci* 55(1):119–139
13. Geman S (2005) On the formulation of a composition machine. Technical report, Division of Applied Mathematics, Brown University
14. Grimson WEL (1990) Object recognition by computer. MIT, Cambridge
15. Jordan I (2004) Graphical models. *Stat Sci (Special issue on Bayesian Stat)* 19:140–155
16. Kemp C, Tenenbaum JB (2008) The discovery of structural form. *Proc Natl Acad Sci* 105(31):10687–10692
17. Lee TS, Mumford D (2003) Hierarchical Bayesian inference in the visual cortex. *J Opt Soc Am A Opt Image Sci Vis* 20(7):1434–1448.
18. Lowe DG (1987) Three-dimensional object recognition from single two-dimensional images. *J Artif Intell* 31: 355–395
19. Lowe D (2004) Distinctive image features from scale-invariant key-points. *Int J Comput Vis* 60(2): 91–110
20. Marr D (1982) Vision: a computational investigation into the human representation and visual information. W.H. Freeman, New York
21. Marr D, Nishihara HK (1978) Representation and recognition of the spatial organisation of three-dimensional shapes. *Proc R Soc Lond B* 200:269–294
22. Mohan A, Papageorgiou C, Poggio T (2001) Example-based object detection in images by components. *IEEE Trans Pattern Anal Mach Intell* 23:349–361
23. Papageorgiou C, Evgeniou T, Mukherjee S, Poggio T (1998) A trainable pedestrian detection system. In: IEEE international conference on intelligent vehicles, Stuttgart, vol 1, pp 241–246
24. Papageorgiou C, Oren M, Poggio T (1998) A general framework for object detection. In: Proceedings of the international conference on computer vision, Bombay, 4–7 Jan 1998
25. Pascal website. <http://pascallin.ecs.soton.ac.uk/challenges/voc/>
26. Poggio T, Smale S (2003) The mathematics of learning: dealing with data. *Notices AMS* 50:537–544
27. Poggio T, Rifkin R, Mukherjee S, Niyogi P (2004) General conditions for predictivity in learning theory. *Nature* 428:419–422
28. Ponce J, Berg TL, Everingham M, Forsyth DA, Hebert M, Lazebnik S, Marszałek M, Schmid C, Russell BC, Torralba A, Williams CKI, Zhang J, Zisserman A (2007) In: Ponce J, Hebert M, Schmid C, Zisserman A (eds) *Toward category-level object recognition*. Lecture notes in computer science. Springer, Berlin
29. Rowley H, Baluja S, Kanade T (1998) Neural network-based face detection. *IEEE Trans Pattern Anal Mach Intell (PAMI)* 20(1):23–38
30. Sali E, Ullman S (1999) Combining class-specific fragments for object classification. In: Proceedings of the 10th British machine vision conference, Nottingham, vol 1, pp 203–213
31. Sung K, Poggio T (1998) Example-based learning for view-based human face detection. *IEEE Trans Pattern Anal Mach Intell* 20(1):39–51

32. Turk M, Pentland A (1991) Eigen faces for recognition. *J Cognit Neurosci* 3(1):71–86
33. Ullman S, Basri R (1991) Recognition by linear combination of models. *IEEE PAMI* 13(10):992–1006
34. Ullman S, Vidal-Naquet M, Sali E (2002) Visual features of intermediate complexity and their use in classification. *Nature Neurosci* 5(7):1–6
35. Vapnik N (1995) The nature of statistical learning theory. Springer, New York
36. Vapnik N (1998) Statistical learning theory. Wiley, New York
37. Vedaldi A, Gulshan V, Varma M, Zisserman A (2009) Multiple Kernels for object detection. In: Proceedings of the international conference on computer vision, Kyoto
38. Viola P, Jones M (2001) Robust real-time object detection. *Int J Comput Vis* 56:151–177
39. Zhang J, Zisserman A (2006) Dataset issues in object recognition. In: Ponce J, Hebert M, Schmid C, Zisserman A (eds) Toward category-level object recognition. Springer, Berlin, pp 29–48

Many-to-Many Graph Matching

Fatih Demirci¹, Ali Shokoufandeh² and Sven J. Dickinson³

¹Department of Computer Engineering, TOBB University of Economics and Technology, Sogutozu, Ankara, Turkey

²Department of Computer Science, Drexel University, Philadelphia, PA, USA

³Department of Computer Science, University of Toronto, Toronto, ON, Canada

Synonyms

Error-correcting graph matching; Error-tolerant graph matching; Inexact matching; Transportation problem

Definition

When objects exhibit large within-class variation and/or when image features are under- or over-segmented, the image features extracted from two exemplars belonging to the same category may no longer be in one-to-one correspondence but, in general, many-to-many correspondence. If the features are structured, i.e., captured in a graph, then computing the correct correspondence can be formulated as a many-to-many graph matching problem.

Background

The matching of image features to object models is typically formulated as a one-to-one assignment problem, based on the assumption that for every salient image feature belonging to the object to be matched, e.g., SIFT feature, image patch, contour fragment, there exists a single corresponding feature on the model (and vice versa). While the one-to-one correspondence assumption has been prevalent in the object recognition community throughout its entire evolution, including the paradigms of graph matching [9], alignment [13], geometric invariants [11], local appearance [14], and a recent return to local contour-based features [8], one-to-one feature correspondence is a highly restrictive assumption that breaks down as within-class variation increases and as the segmentation and extraction of more abstract image features suffer from over- or under-segmentation [7]. In the more general case, feature correspondence is not one-to-one, but rather *many-to-many*. If a feature set is described by a graph, with nodes representing features and edges capturing pairwise relations between features, computing the correct many-to-many feature correspondence can be formulated as *many-to-many graph matching*.

Consider two simple examples, shown in Fig. 1. In Fig. 1a, a set of multiscale blobs and ridges have been extracted from two exemplars (humans) belonging to the same category. In the top image, the straight arm yields a single elongated ridge, while in the bottom image, the bent arm yields two smaller and coterminating elongated ridges. In this case, simple object articulation (a form of within-class variation) has led to a violation of the one-to-one correspondence assumption. Instead, the correspondence is clearly two-to-one; enforcing one-to-one correspondence will lead to an incorrect matching of the entire arm to either the upper or lower arm, e.g., the red highlighted features. In Fig. 1b, two region segmentations of two exemplars belonging to the same class yield a set of region correspondences that are rarely one-to-one, but more typically many-to-many. Once again, enforcing a one-to-one feature correspondence will ensure an incorrect matching, and will miss the correct correspondence.

The problem of computing a one-to-one correspondence between a model feature graph and a cluttered image graph can be formulated as a largest isomorphic subgraph problem, whose complexity is NP-hard.



Many-to-Many Graph Matching, Fig. 1 Two graph matching problems in computer vision for which assuming a one-to-one feature correspondence will lead to incorrect correspondences, and which can only be solved if formulated as a many-to-many graph-matching problem. In (a), a multiscale blob and ridges decomposition [17] of the two humans yields a single ridge for the extended arm (*top*) and two coterminating ridges for the bent arm (*bottom*). In this example, articulation has violated the one-to-one feature correspondence assumption; if a

one-to-one correspondence is enforced for the arm, e.g., the red highlighted features, it will be incorrect. In this case, the correspondence should be two-to-one (or more generally, many-to-many). In (b), two different cup exemplars (*top row*) have been region segmented (*top row*), yielding regions that are rarely in one-to-one correspondence (due to within-class variation or region over- and/or under-segmentation). Once again, the correct correspondence is not one-to-one, but rather many-to-many

The complexity of the many-to-many matching problem is even more prohibitive, for the space of possible correspondences is greater (any subset of features in the image may match any subset of features on the model). The intractable complexity of the many-to-many matching problem can only be reduced by exploiting the types of regularities suggested by the perceptual grouping community, such as proximity, continuity, conservation of mass, etc. In what follows, a formal statement of the problem is introduced, and a number of approaches to its solution is reviewed.

graphs. In an attribute-weighted graph $G = (V, E)$, let $\mathbb{L}(v)$ denote the set of attributes associated with $v \in V$. Given a subset $U \subset V$, let $\mathbb{L}(U) = \cup_{u \in U} \mathbb{L}(u)$. For a set $U \subset V$, let $G|_U$ denote the subgraph of G induced on the vertices in U , and let $w(u, v)$ denote the weight of an edge $(u, v) \in E$. Finally, let $\mathbb{P}(G)$ denote the power-set 2^V for the vertex set of G . A *many-to-many mapping* between two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is a mapping among power-sets $\mathbb{P}(G_1)$ and $\mathbb{P}(G_2)$ and can be characterized as a function:

$$\mathcal{M} : (\mathbb{P}(G_1) \times \mathbb{P}(G_2)) \rightarrow \{0, 1\}. \quad (1)$$

Theory

The main objective of the many-to-many graph matching problem is to establish a minimum cost mapping between the vertices of two attributed, edge-weighted

For two sets, $U \in \mathbb{P}(G_1)$ and $V \in \mathbb{P}(G_2)$, there will be a cost $C(\mathbb{L}(U), \mathbb{L}(V))$ associated with mapping the labels in set $\mathbb{L}(U)$ to those in $\mathbb{L}(V)$. An example of a common cost function is the edit-distance between the labels in sets $\mathbb{L}(U)$ and $\mathbb{L}(V)$. Let $\mathcal{S}(G_1|_U, G_2|_V)$ denote

the structural distance between induced subgraphs $G_1|_U$ and $G_2|_V$. Observe that every mapping \mathcal{M} has a natural representation as a matrix, with $\mathcal{M}_{U,V} = 1$ if the sets $U \in \mathbb{P}(G_1)$ and $V \in \mathbb{P}(G_2)$ are mapped to each other under \mathcal{M} , and $\mathcal{M}_{U,V} = 0$ otherwise. Combining these two cost functions will result in the cost function $C(\mathcal{M})$ associated with the mapping \mathcal{M} :

$$\begin{aligned} C(\mathcal{M}) = & \sum_{U \in \mathbb{P}(G_1), V \in \mathbb{P}(G_2)} \mathcal{M}_{U,V} \\ & \times C(\mathbb{L}(U), \mathbb{L}(V)) \times \mathcal{S}(G_1|_U, G_2|_V). \end{aligned} \quad (2)$$

In defining an optimal many-to-many matching between two attributed graphs, G_1 and G_2 , a many-to-many mapping \mathcal{M}^* of minimum cost $C(\mathcal{M}^*)$ subject to specific requirements on the structure or cardinality of \mathcal{M}^* will be obtained. For example, to prevent a trivial solution that sets $\mathcal{M}_{U,V} = 0$, for all U and V , one can require a matching such that its cardinality, i.e., $\sum_{U,V} \mathcal{M}_{U,V}$, exceeds a threshold while minimizing $C(\mathcal{M})$. Other functions, such as maximizing the number of vertices from V_1 and V_2 that participate in \mathcal{M} , can be used to evaluate the quality of the mapping. Note that cost functions $C(\mathbb{L}(U), \mathbb{L}(V))$ and $\mathcal{S}(G_1|_U, G_2|_V)$ may be used to enforce constraints such as consistency of mapped labels, limits of feasible label mappings, or allowed structural mapping of induced graphs $G_1|_U$ and $G_2|_V$ by imposing arbitrary large values or by being ill-defined.

The above description of the many-to-many matching results in an intractable computational problem. First, due to the exponential size of power-sets $\mathbb{P}(V_1)$ and $\mathbb{P}(V_2)$ in terms of number of vertices in G_1 and G_2 , the size of the search space for the many-to-many matching problem is exponential. Even simplifying the problem to one-to-one mappings, by replacing the power-sets $\mathbb{P}(V_1)$ and $\mathbb{P}(V_2)$ with sets V_1 and V_2 , respectively, will result in the multidimensional matching problem that is known to be NP-complete for arbitrary labeled graphs.

Related Work

Many-to-many graph matching has been studied extensively in a variety of contexts, including graph edit distance [2, 16], spectral methods [4, 18], optimization problems [20], metric embedding [6], abstract models [10], and grammars [1, 21]. The classical

formulation of graph edit distance introduces a set of graph edit operations, such as insertion, deletion, merging, splitting, and substitution of nodes and edges. Given a set of graph edit operations and a cost function, the objective is to find the lowest cost sequence of graph edit operations that transform one graph into the other. The edit distance between two graphs critically depends on the costs of the underlying edit operations; typically, lower costs are assigned to the most frequent edit operations. A number of approaches have addressed the problem of defining an appropriate cost, e.g., [3].

Many-to-many graph matching has also been studied in the context of spectral methods by examining the spectral properties of graph adjacency matrices. In [4], the authors present an approach based on renormalization projections of vertices into a common eigensubspace of two graphs. Instead of finding the overall similarity of two graphs from the positions of vertex projections, this approach uses an agglomerative hierarchical clustering technique to produce many-to-many vertex correspondences.

Another spectral method is due to [18, 19], which constructs a low-dimensional “signature” of a directed graph’s “shape” from the magnitudes of the eigenvalues of the graph’s adjacency matrix. The eigenvalues are invariant to the reordering of a graph’s branches and are shown to be robust under minor structural perturbation of the graph. This vector can be used for both structural indexing and for matching in the presence of noise and occlusion. If two signatures (vectors) are close, their corresponding (sub)graphs, possibly having different cardinalities, are in many-to-many correspondence.

Recently, the approach presented in [20] formulates the many-to-many graph matching problem as a discrete optimization problem. The algorithm starts by extending the optimization problem for one-to-one matching to the case of many-to-one matching. The algorithm then obtains many-to-many vertex correspondences through two many-to-one mappings. Since this formulation of the many-to-many matching requires the solution of a hard optimization problem, the authors propose an approximate algorithm based on a continuous relaxation of the combinatorial problem.

The concept of a low-distortion graph embedding has been used to obtain many-to-many vertex correspondences [6]. Specifically, low-distortion graph

embedding is employed to transform the problem of many-to-many graph matching to a many-to-many point matching problem in a geometric space. This transformation maps nodes to points and edge weights to interpoint distances, not only simplifying the original graph representation (by removing the edges), but also retaining important local and global graph structure; moreover, the transformation is robust under perturbation. Representing two graphs as sets of points reduces the many-to-many graph matching problem to that of many-to-many point matching in the geometric space, for which a number of efficient distribution-based similarity measures are available. The authors use the Earth Mover’s Distance [15] algorithm to find such correspondences and show that the resulting many-to-many point matching realizes the desired many-to-many matching between the vertices of the input graphs.

A number of researchers, e.g., [10, 12] and [5], have explored many-to-many graph matching in the context of model-based abstraction from images. The work presented in [10] starts by forming a region adjacency graph from each image. The approach then searches the space of pairwise region groupings in each graph, forming a lattice. Each input image yields a lattice such that its bottom node represents the original region adjacency graph and its top node represents the silhouette of the object. The framework defines a common abstraction as a set of nodes, one per lattice, such that for a pair of nodes, their corresponding graphs are isomorphic. The lowest common abstraction (LCA) is defined as the common abstraction whose underlying graph has the maximum number of nodes. Thus, the resulting LCA carries the most informative abstraction common to each image. Although effective, this technique does not find a match between two graphs whose common abstraction does not exist.

The two algorithms presented in [12] and [5] use the many-to-many graph matching technique of [6] for automatically constructing an abstract model from examples. The work in [12] computes the multi-scale ridge/blob decomposition (AND-OR) graph for each input image and obtains the many-to-many node correspondences between each pair of graphs, yielding a matching matrix. By exploring this matrix, the algorithm first finds features that match one-to-one across many pairs of input images. The many-to-many matchings between these features are then analyzed to obtain the decompositional relations among them.

The extracted features and their relations are used to construct the final abstract model.

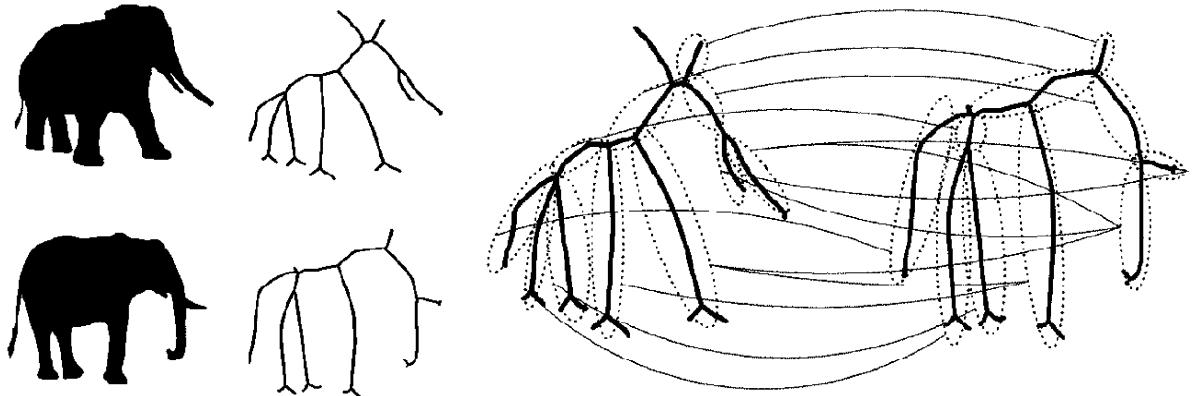
After obtaining many-to-many node correspondences based on [6], the algorithm in [5] computes the abstracted medial axis graph by first computing the averages of the corresponding pairs of subgraphs to yield the nodes in the abstracted graph, and then defining the overall topology of the resulting abstract parts to yield the relations. Each matching pair of subgraphs corresponds to a single node in the abstracted graph, and two abstracted nodes are connected by an edge if the corresponding subgraphs are adjacent in the original graphs. This procedure forms the basis of an iterative framework in which pairs of similar medial axis graphs are clustered and abstracted, yielding a set of abstract medial axis graph class prototypes.

In the domain of grammars, objects are represented as variable hierarchical structures. Each part in this representation can be defined in terms of other parts, allowing an object to be modeled by its coarse-to-fine appearance. Overall, grammar-based models including AND-OR graphs support structural variability. To represent intra-category variation and to account for many-to-many correspondence, the grammar creates a large number of configurations from a small vocabulary set. To scale to a large number of object categories, the AND-OR graph, learning, and inference algorithms are defined recursively. Some examples of this type of approach include [1, 21].

Experimental Results

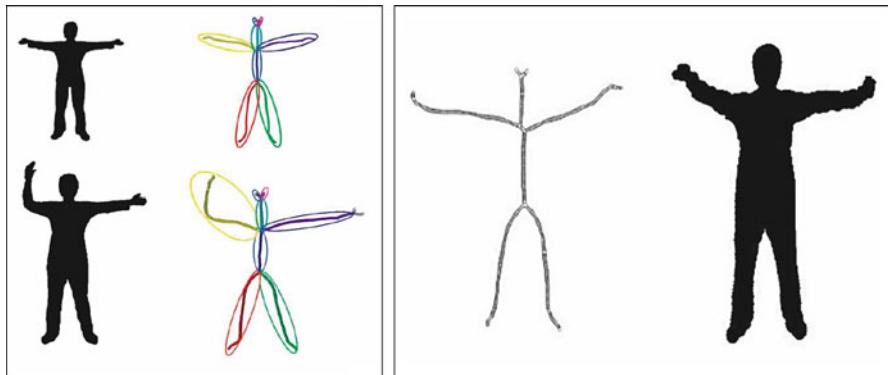
In this section, some example results from some of the many-to-many matching approaches described in the Related Work section are illustrated. After representing silhouettes as skeleton graphs in Fig. 2, the algorithm proposed in [6] obtains many-to-many node correspondences through metric embedding, as discussed earlier. Based on the many-to-many correspondences of this algorithm, Fig. 3 demonstrates an example for the abstract shape created by the approach presented in [5]. The left part presents input silhouettes, their skeleton graphs, and many-to-many correspondences. The right part presents the abstract skeleton graph and its shape reconstructed from this graph.

Graph edit distance is another important class of many-to-many graph matching algorithms. Figure 4



Many-to-Many Graph Matching, Fig. 2 Example many-to-many correspondences computed by [6]. After representing two silhouettes as skeleton graphs, the graphs are embedded into geometric spaces of the same dimensionality. The embedded

points are then matched using the Earth Mover's Distance algorithm. The *right part* illustrates the many-to-many correspondences between the vertices of the input graphs. Each *dashed ellipsoid* represents a set of vertices from the original graph



Many-to-Many Graph Matching, Fig. 3 A shape abstraction example of [5] based on many-to-many correspondences obtained by [6]. The *left image* shows input silhouettes and their skeleton graphs in which the same color is used to show the

corresponding parts. Using these correspondences, the abstract skeleton graph and its silhouette are created as shown on the *right*



Many-to-Many Graph Matching, Fig. 4 Graph edit distance algorithms compute many-to-many correspondences of a pair of graphs by finding the lowest cost sequence of graph edit operations needed to transform one graph into another. In the example,

same colors indicate the matching skeleton parts, while gray colors show spliced or contracted edges (The example is taken from Ref. [16])

shows the result of matching the skeleton graphs for two input shapes using the graph edit distance algorithm described in [16]. Same colors indicate the matching skeleton parts while gray colors show spliced or contracted edges. Observe that the many-to-many correspondences are intuitive in these figures.

References

1. Bunke H (1982) Attributed graph grammars and their application to schematic diagram interpretation. *IEEE Trans Pattern Anal Mach Intell* 4:574–582
2. Bunke H (1997) On a relation between graph edit distance and maximum common subgraph. *Pattern Recognit Lett* 18(8):689–694
3. Bunke H, Shearer K (1998) A graph distance metric based on the maximal common subgraph. *Pattern Recognit Lett* 19:255–259
4. Caelli T, Kosinov S (2004) An eigenspace projection clustering method for inexact graph matching. *IEEE Trans Pattern Anal Mach Intell* 26:515–519
5. Demirci F, Shokoufandeh A, Dickinson S (2009) Skeletal shape abstraction from examples. *IEEE Trans Pattern Anal Mach Intell* 31:944–952
6. Demirci F, Shokoufandeh A, Keselman Y, Bretzner L, Dickinson S (2006) Object recognition as many-to-many feature matching. *Int J Comput Vis* 69(2):203–222
7. Dickinson S (2009) The evolution of object categorization and the challenge of image abstraction. In: Dickinson S, Leonardis A, Schiele B, Tarr M (eds) *Object categorization: computer and human vision perspectives*. Cambridge University Press, New York, pp 1–37
8. Ferrari V, Jurie F, Schmid C (2010) From images to shape models for object detection. *Int J Comput Vis* 87(3): 284–303
9. Fischler MA, Eschlagier RA (1973) The representation and matching of pictorial structures. *IEEE Trans Comput* 22(1):67–92
10. Keselman Y, Dickinson S (2005) Generic model abstraction from examples. *IEEE Trans Pattern Anal Mach Intell* 27(7):1141–1156
11. Lamdan Y, Schwartz J, Wolfson H (1990) Affine invariant model-based object recognition. *IEEE Trans Rob Autom* 6(5):578–589
12. Levinstein A, Sminchisescu C, Dickinson S (2005) Learning hierarchical shape models from examples. In: Proceedings of the EMMCVPR, St. Augustine. Springer, Berlin, pp 251–267
13. Lowe D (1985) Perceptual organization and visual recognition. Academic, Norwell
14. Lowe D (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110
15. Rubner Y, Tomasi C, Guibas LJ (2000) The earth mover's distance as a metric for image retrieval. *Int J Comput Vis* 40(2):99–121
16. Sebastian T, Klein P, Kimia B (2004) Recognition of shapes by editing their shock graphs. *IEEE Trans Pattern Anal Mach Intell* 26:550–571
17. Shokoufandeh A, Bretzner L, Macrini D, Demirci MF, Jönsson C, Dickinson S (2006) The representation and matching of categorical shape. *Comput Vis Image Underst* 103(2):139–154
18. Shokoufandeh A, Macrini D, Dickinson S, Siddiqi K, Zucker SW (2005) Indexing hierarchical structures using graph spectra. *IEEE Trans Pattern Anal Mach Intell* 27(7):1125–1140
19. Siddiqi K, Shokoufandeh A, Dickinson S, Zucker S (1999) Shock graphs and shape matching. *Int J Comput Vis* 30: 1–24
20. Zaslavskiy M, Bach F, Vert J (2010) Many-to-many graph matching: a continuous relaxation approach. Lecture Notes in Computer Science, <http://arxiv.org/abs/1004.4965>, DBLP, <http://dblp.uni-trier.de/6323:5> 15–530
21. Zhu S, Mumford D (2006) A stochastic grammar of images. *Found Trends Comput Graph Vis* 2:259–362

Matte Extraction

Jiaya Jia

Department of Computer Science and Engineering,
The Chinese University of Hong Kong, Shatin, N.T.,
Hong Kong, China

Synonyms

Digital matting; Pulling a matte

Definition

An alpha matte has the same size as the input image. It contains respective weights to linearly blend latent foreground and background colors for each pixel to form the observed color. Estimating the alpha matte together with the foreground color image is generally referred to as matte extraction or digital matting.

Background

Classifying each pixel in an input image to either foreground or background is called *binary segmentation*, which is a fundamental computer vision problem. Digital matting relaxes the hard separation assumption

and takes ubiquitous foreground and background color blending in image formation, which happens along almost all object boundaries, into consideration. Results from matte extraction can be used to generate a new composite.

Color blending in natural images has a variety of causes, such as color interpolation during image production and light photons received by the camera sensor containing both background and foreground color for some pixels. Without additional information, digital matting is an ill-posed problem with many unknowns. So generally, either multiple frames are taken or a certain amount of user interaction is involved to sample foreground and background color in image and video matting.

Theory

In the digital matting framework, separating the background image B and foreground image F with respect to an alpha matte α from an input natural image I is expressed as

$$I = \alpha F + (1 - \alpha)B. \quad (1)$$

If $\alpha(x, y) = 1$, the pixel with coordinate (x, y) is definitely in the foreground. $\alpha(x, y)$ being 0 defines an absolutely background pixel. $\alpha(x, y)$ can also be in between 0 and 1, indicating a certain level of color mixing. Digital matting aims to estimate α and F (sometimes also B) from I . Existing methods follow one of the following lines.

Blue Screen Matting

Blue screen matting [1], which is widely employed in movie and commercial production, needs to set up a controlled environment and uses a single or multiple constant-color backgrounds (as shown in Fig. 1). The blue screen matting problem is directly solvable. Its triangular matting technique, which captures images with two backgrounds containing different shades of the backing color, is particularly noteworthy because a closed-form solution exists. This technique can produce very accurate matting results usable as ground truth data. Blue screen matting can be applied in a frame-by-frame fashion to video foreground object extraction.

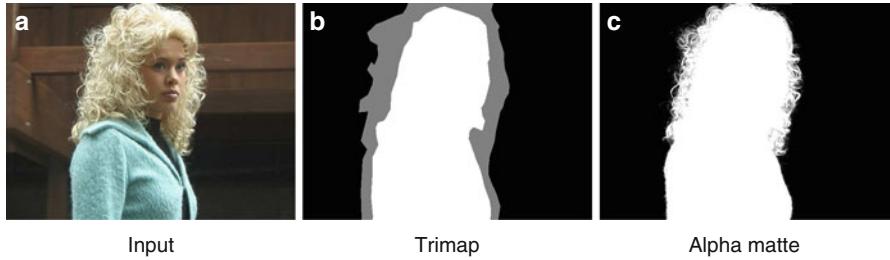
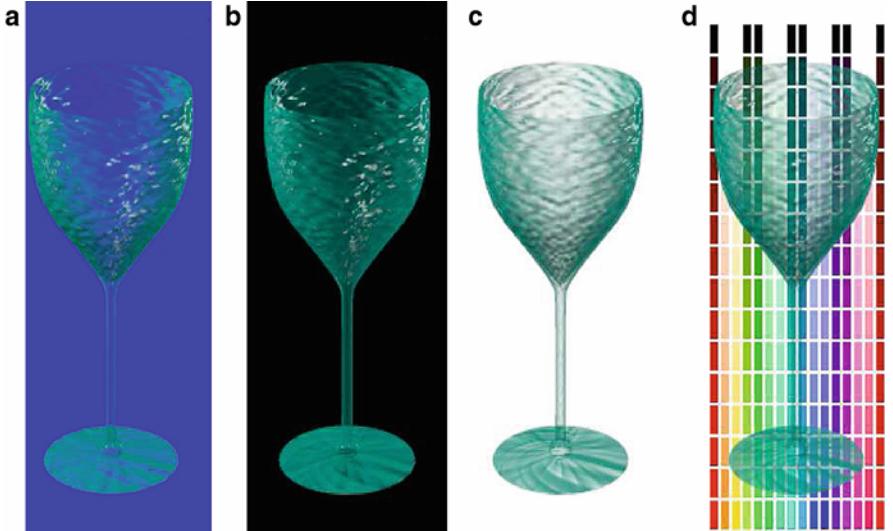
Natural Image Matting

In natural image matting, background B is generally unknown and possibly contains complex structures. In this case, simultaneously estimating α , F , and B becomes an ill-posed problem. Several methods [3–5] need user input of additional segmentation information to constrain it. *Trimap* is a popular format that partitions the image into three regions, i.e., “definitely foreground” (DF for short), “definitely background” (DB for short), and “unknown region”, as shown in Fig. 2b. In DF and DB, α is set to 1 and 0, respectively. Digital matting only estimates α , together with the foreground and background color, in the unknown region by gathering color information from DF and DB.

There have been several methods proposed to sample color from DF and DB. In knockout [6], F is computed as the weighted average of foreground color along the perimeter of the DF region. B is computed similarly but with a final refinement step. Ruzon and Tomasi [3] sample F and B from local windows and then parameterize them as a mixture of unoriented Gaussians. Alpha values are computed by maximizing a function that interpolates the mean and variance of the Gaussians. Bayesian matting [4] gathers color samples from DF and DB using sliding windows and fits them with oriented Gaussian distributions. A *maximum a posteriori* (MAP) estimation of α , F , and B is applied. The final α values are chosen from the foreground and background color pairs that maximize the probability. Global Poisson matting [5] contributes a gradient-domain alpha matte estimation. When the condition of locally smooth color change in DF and DB is violated, user interaction is involved to improve the matting result with the supply of a group of filters.

The quality of results of these methods partly depends on how accurate the trimap is since color is sampled and the alpha matte is estimated within windows. Many later approaches instead require the user to only draw several foreground and background scribbles to coarsely indicate DF and DB and leave all unspecified pixels in the unknown region. This scheme simplifies user interaction but provides looser constraints for digital matting, as shown in Fig. 3. Representative work that can robustly solve for alpha mattes based on it includes (1) the iterative-optimization method [2], which samples color from user-drawn scribbles, builds the Markov Random Field (MRF), and solves for segmentation and matte

Matte Extraction, Fig. 1
 Blue screen matting [1]. (a) Object against known constant *blue*. (b) Object against constant *black*. (c) Pulled foreground. (d) New composite



Matte Extraction, Fig. 2 A trimap matting example. (a) Input image. (b) The user-provided trimap where definitely foreground and background are in *white* and *black*, respectively. The *gray*

pixels are unknown ones. (c) Alpha matte estimate by global Poisson matting

extraction using belief propagation, and (2) closed-form matting [7] that introduces a color line model and based on it derives a quadratic cost function only involving α and a matting Laplacian, enabling linear optimization. In addition, Rhemann et al. [8] extract high-resolution mattes by trimap segmentation and by employing gradient preserving alpha priors. The soft-scissor method of Wang et al. [9] can achieve real-time matting along with user painting the foreground boundary.

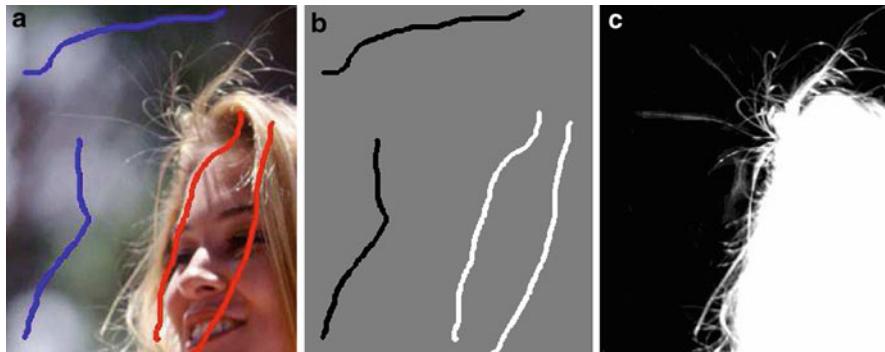
There are also automatic image matting methods. A soft color segmentation method was proposed in [26], where a global objective function is modeled by global and local parameters. These parameters are alternately optimized until convergence. It can be applied to matting without intensive user interaction. Spectral matting [10] is a single image approach. It shows that the smallest eigenvectors of

the matting Laplacian span individual matting components, making their estimation equivalent to finding linear transformation of the eigenvectors. Flash matting [11] captures a pair of flash/no-flash images and assumes only the foreground region is lit by flash. This method can automatically extract foreground in a joint Bayesian matting framework.

Albeit some inevitable limitations as described in respective papers, all the above techniques advance image matting from different aspects. Other recommended readings also include [12–14].

Video Matting

Digital matting was extended to videos in various ways. Typical video matting methods deal with foreground regions with hair, trees, or smoke, where color blending exists for a large amount of pixels. To preserve temporal coherence among frames, with



Matte Extraction, Fig. 3 Matting with scribbles. (a) Input image with user-drawn scribbles. (b) The initial trimap. (c) Alpha matte result of Wang and Cohen [2]

the input of a monocular video, Bayesian video matting [15] propagates manually specified trimaps from keyframes to other frames using optical flow and suggests completing background using mosaic construction. This method is improved in [16] by incorporating stronger prior terms. The geodesic matting method [17] introduces temporal neighbors for each pixel and infers foreground and background scribbles in the video based on user input in only sparse frames. With the setup of special devices or systems, defocus matting [18] and the camera-array method [19] make use of multiple cameras that take pictures with different focus settings and with the existence of parallax respectively to profit video matting.

The other set of methods [20, 21] adopt video matting in a refinement step to improve foreground boundary estimation after video cutout where unknown regions are generally narrowbands around the boundaries.

Application

Digital matting exploits pixel-wise color blending and is an indispensable technique for high-quality object extraction from images and videos. The matte estimate together with the computed foreground color can then be used to form a new composite with another background image. Simple composition applies linear color blending again based on (Eq. 1), while several other approaches, include drag-and-drop pasting [22], context-sensitive blending [23], and compositional matting [24], explore the structure relationship between the source and target images and combine color blending with other schemes.

Matte extraction and the corresponding composite construction are fundamental tools for image/video editing and finds many applications in computer graphics and vision. Movie and commercial production relies on it to naturally insert objects into a virtual or real scene. Digital matting can possibly be combined with other decomposition, recognition, and tracking techniques to further improve the performance and expand the usability.

Experimental Results

Rhemann et al. [25] established a digital matting evaluation website containing data classified into high, strong, medium, and little transparency groups. Training data are also provided. This website contains updated experimental results of many approaches.

References

- Smith AR, Blinn JF (1996) Blue screen matting. In: Proceedings of the ACM SIGGRAPH, New Orleans, 259–268
- Wang J, Cohen MF (2005) An iterative optimization approach for unified image segmentation and matting. In: Proceedings of the ICCV, Beijing, 936–943
- Ruzon MA, Tomasi C (2000) Alpha estimation in natural images. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), San Diego, 18–25
- Chuang YY, Curless B, Salesin DH, Szeliski R (2001) A bayesian approach to digital matting. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), Kauai, vol II, 264–271
- Sun J, Jia J, Tang CK, Shum HY (2004) Poisson matting. ACM Trans Graph 23(3):315–321

6. Berman A, Vlahos P, Dadourian A (2000) Comprehensive method for removing from an image the background surrounding a selected object. US Patent 6,134,345
7. Levin A, Lischinski D, Weiss Y (2006) A closed form solution to natural image matting. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), New York, 61–68
8. Rhemann C, Rother C, Rav-Acha A, Sharp T (2008) High resolution matting via interactive trimap segmentation. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), Anchorage
9. Wang J, Agrawala M, Cohen MF (2007) Soft scissors: an interactive tool for realtime high quality matting. In: Proceedings of the ACM SIGGRAPH, San Diego
10. Levin A, Rav-Acha A, Lischinski D (2008) Spectral matting. *IEEE Trans Pattern Anal Mach Intell* 30: 1699–1712
11. Sun J, Li Y, Kang SB, Shum HY (2006) Flash matting. In: Proceedings of the SIGGRAPH, Boston, 772–778
12. Grady L, Schiwietz T, Aharon S, Westermann R (2005) Random walks for interactive alpha-matting. In: Proceedings of the VIIP, 423–429
13. Wang J, Cohen MF (2007) Image and video matting: a survey. *Found Trends Comput Graph Vis* 3:97–175
14. Rhemann C, Rother C, Kohli P, Gelautz M (2010) A spatially varying psf-based prior for alpha matting. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), San Francisco, 2149–2156
15. Chuang YY, Agarwala A, Curless B, Salesin DH, Szeliski R (2002) Video matting of complex scenes. In: Proceedings of the SIGGRAPH, San Antonio, 243–248
16. Apostoloff N, Fitzgibbon A (2004) Bayesian video matting using learnt image priors. *Comput Vis Pattern Recognit* 1:407–414
17. Bai X, Sapiro G (2009) Geodesic matting: a framework for fast interactive image and video segmentation and matting. *Int J Comput Vis* 82:113–132
18. McGuire M, Matusik W, Pfister H, Hughes JF, Durand F (2005) Defocus video matting. In: Proceedings of the ACM SIGGRAPH, Los Angeles, 567–576
19. Joshi N, Matusik W, Avidan S (2006) Natural video matting using camera arrays. In: Proceedings of the SIGGRAPH, Boston, 779–786
20. Li Y, Sun J, Shum HY (2005) Video object cut and paste. In: Proceedings of the ACM SIGGRAPH, Los Angeles, 595–600
21. Wang J, Bhat P, Colburn RA, Agrawala M, Cohen MF (2005) Interactive video cutout. In: Proceedings of the ACM SIGGRAPH, Los Angeles, 585–594
22. Jia J, Sun J, Tang CK, Shum HY (2006) Drag-and-drop pasting. In: Proceedings of the ACM SIGGRAPH, Boston, 631–637
23. Lalonde JF, Hoiem D, Efros AA, Rother C, Winn J, Criminisi A (2007) Photo clip art. In: Proceedings of the ACM SIGGRAPH, San Diego
24. Wang J, Cohen MF (2007) Simultaneous matting and compositing. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), Minneapolis
25. Rhemann C, Rother C, Wang J, Gelautz M, Kohli P, Rott P (2009) A perceptually motivated online benchmark for image matting. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), Miami
26. Tai Y-W, Jia J, Tang C-K (2007) Soft color segmentation and its applications. *IEEE Trans. Pattern Anal. Mach. Intell.* 29, 9 (September 2007), 1520–1537

Maximum Likelihood Estimation

Thomas Brox

Department of Computer Science, University of Freiburg, Freiburg, Germany

Synonyms

[Maximum likelihood estimator](#)

Definition

Maximum likelihood estimation seeks to estimate model parameters that best explain some given, independent measurements according to a noise model.

Background

Many problems in computer vision can be formulated as finding the parameters of a predefined model given measurements or training examples.

For example in image segmentation one may want to describe a region by a simple region model, e.g., by a constant intensity value μ . There are many measurements, namely all the pixel intensities in the region. Assuming that these pixel intensities are independently generated from the constant intensity model according to a Gaussian distribution, the goal is to find the most likely parameter μ given these measurements. In this simple example, the optimal parameter μ is the mean of all intensities.

There are many more similar problems in computer vision, for instance, in the scope of optical flow estimation, camera calibration, image denoising, or pattern recognition. In the special case of a Gaussian noise model, maximum likelihood estimation comes down to a least squares approach.

Maximum likelihood estimation is often criticized because it ignores a-priori information, which can be interpreted as assuming a uniform prior density on the parameter space. This becomes especially problematic

when the model is described by many parameters and there are relatively few measurements. In cases where good a-priori assumptions can be made, maximum likelihood estimation should be replaced by maximum a-posteriori estimation, which takes the prior density into account.

Theory

Given a probabilistic model that is described by a parameter vector $\mathbf{w} \in \mathbb{R}^D$ and given N independent measurements $\mathbf{x}_n \in \mathbb{R}^K$, $N \geq D$, one aims at maximizing the likelihood

$$p(\mathbf{x}_1, \dots, \mathbf{x}_N | \mathbf{w}) = \prod_{n=1}^N p(\mathbf{x}_n | \mathbf{w}). \quad (1)$$

For numerical reasons, rather than maximizing this probability, it is common to maximize its logarithm, the so-called log-likelihood:

$$\begin{aligned} \mathbf{w}^* &= \underset{\mathbf{w}}{\operatorname{argmax}} \log p(\mathbf{x}_1, \dots, \mathbf{x}_N | \mathbf{w}) \\ &= \underset{\mathbf{w}}{\operatorname{argmax}} \sum_{n=1}^N \log p(\mathbf{x}_n | \mathbf{w}). \end{aligned} \quad (2)$$

Application

Applying this to a simple regression problem, where a line is to be fitted to a couple of points, one has the constraints

$$w_1 x_{1,n} + w_2 = x_{2,n}, \quad n = 1, \dots, N. \quad (3)$$

Assuming a Gaussian distribution with constant covariance yields

$$\sum_{n=1}^N \log p(\mathbf{x}_n | \mathbf{w}) \propto \sum_{n=1}^N (w_1 x_{1,n} + w_2 - x_{2,n})^2. \quad (4)$$

The connection to least squares estimation can be seen immediately, but one could as well assume a Laplace distribution, which is more robust to outliers among the

measurements and would lead to

$$\sum_{n=1}^N \log p(\mathbf{x}_n | \mathbf{w}) \propto \sum_{n=1}^N |w_1 x_{1,n} + w_2 - x_{2,n}|. \quad (5)$$

A necessary condition for a maximum of this expression is that the gradient with respect to the parameter vector must vanish:

$$\begin{aligned} \frac{\partial}{\partial w_1} \sum_{n=1}^N |w_1 x_{1,n} + w_2 - x_{2,n}| &= 0 \\ \frac{\partial}{\partial w_2} \sum_{n=1}^N |w_1 x_{1,n} + w_2 - x_{2,n}| &= 0 \end{aligned} \quad (6)$$

leading to the nonlinear system

$$\begin{aligned} \sum_{n=1}^N \frac{1}{2} \frac{(w_1 x_{1,n} + w_2 - x_{2,n}) x_{1,n}}{|w_1 x_{1,n} + w_2 - x_{2,n}|} &= 0 \\ \sum_{n=1}^N \frac{1}{2} \frac{(w_1 x_{1,n} + w_2 - x_{2,n})}{|w_1 x_{1,n} + w_2 - x_{2,n}|} &= 0, \end{aligned} \quad (7)$$

which can be solved by iteratively keeping the denominators fixed, solving the resulting linear system and updating the denominator. Gaussian distributions lead to linear systems that can be solved directly. More details and examples on maximum likelihood estimation can be found in [1, 2].

References

1. Duda RO, Stork DG, Hart PE (2000) Pattern classification, 2nd edn. Wiley, New York
2. Bishop CM (2006) Pattern recognition and machine learning. Springer

Maximum Likelihood Estimator

► [Maximum Likelihood Estimation](#)

Mesostructure

► [Bidirectional Texture Function and 3D Texture](#)

Methods of Image Recognition in a Low-Dimensional Eigenspace

- ▶ [Eigenspace Methods](#)

Microgeometry

- ▶ [Bidirectional Texture Function and 3D Texture](#)

Micro Scale Structure

- ▶ [Surface Roughness](#)

Mirrorlike Reflection

- ▶ [Specularity, Specular Reflectance](#)

Mirrors

Jürgen Beyerer
 Fraunhofer Institute of Optronics,
 System Technologies and Image Exploitation IOSB,
 Karlsruhe, Germany

Definition

A mirror is an optical device used for beam-forming or imaging based on the directional reflection of electromagnetic radiation.

Background

Computer vision applications apply mirrors in a twofold manner: for optical imaging and for illumination purposes. Furthermore, mirrors themselves could be test objects in visual inspection systems. This leads, with regard to the 3D shape of the mirror, to the shape-from-specular-reflection problem, and in the context of visual inspection systems to deflectometry.

Theory and Application

Mirrors consist of a smooth substrate with a metal coating (e.g., Au, Ag, Al) and/or dielectric layers. In the case of a surface mirror, the reflection takes place at a metal coating on the front side that has to be protected against scratches. The main advantage of a surface mirror is the lack of beam displacement due to the glass substrate. Alternatively, the backside of a glass substrate can be coated with a metal layer and with an additional protection against humidity and mechanical damage. Backside mirrors are usually more robust than surface mirrors, but lack their optical characteristics mentioned above.

The physical effect leading to the reflection of electromagnetic waves on metal surfaces can be simply described as “short circuit” of the electrical field.

Dielectric mirrors are composed of multiple thin layers of dielectric materials. They exhibit very high reflectance values, whereas the reflectance depends on wavelength, incident angle, and polarization. Advanced multilayer structure designs can be used to obtain certain functionality [10]:

- A broader reflection bandwidth
- A combination of desirable reflectivity values in different wavelength ranges
- Special polarization properties (for non-normal incidence, thin-film polarizers, polarizing beam splitters)
- Non-polarizing beam splitters
- Edge filters, e.g., long-pass filters, high-pass filters, band-pass filters
- Tailored chromatic dispersion properties

Such mirrors are especially used in laser applications.

Furthermore, thin metal layers allow semitransparent mirrors to be realized for coaxial illumination.

The electromagnetic theory of light is fundamental for the physical understanding of specular reflections [3]. Thereby, the law of reflection describes the geometric aspects, and the Fresnel equations the reflection coefficients, i.e., the radiometric behavior.

The law of reflection states the relationship of the incident s_i and reflected s_r light rays with the normal of the specular surface \mathbf{n} :

$$\mathbf{s}_i \times \mathbf{n} = \mathbf{s}_r \times \mathbf{n}, \quad (1)$$

with $\|\mathbf{s}_i\| = \|\mathbf{s}_r\| = \|\mathbf{n}\| = 1$.

Equation 1 leads, with $\|\mathbf{s}_i \times \mathbf{n}\| = \sin \theta_i = \sin \theta_r = \|\mathbf{s}_r \times \mathbf{n}\|$, directly to the following two conditions:

- The angle of the incident ray equals that of the reflected ray ($\theta_i = \theta_r$).
- The incident and reflected ray are coplanar with the surface normal.

In computer graphics and ray-tracing the law of reflection is often used in the form of a Householder transformation:

$$\mathbf{s}_r = \mathbf{H} \mathbf{s}_i \quad \text{with} \quad \mathbf{H} := \mathbf{I} - 2\mathbf{n}\mathbf{n}^T, \quad (2)$$

with the identity matrix \mathbf{I} .

The bidirectional reflectance distribution function (BRDF; $\rho(\theta_i, \varphi_i; \theta_r, \varphi_r)$) describes the reflectance characteristics of a surface, i.e., the ratio of incident and reflected radiance in dependency of incident and observation angles, Nicodemus et al. [9]. According to Horn and Sjoberg [5] the BRDF for an ideal mirror is according to Horn and Sjoberg [5] (c.f., Fig. 1):

$$\begin{aligned} \rho(\theta_i, \varphi_i; \theta_r, \varphi_r) &= \frac{dL_r}{L_i \cos \theta_i d\Omega_i} \\ &= 2 \delta(\sin^2 \theta_r - \sin^2 \theta_i) \delta(\varphi_r - ((\varphi_i + \pi) \bmod 2\pi)). \end{aligned} \quad (3)$$

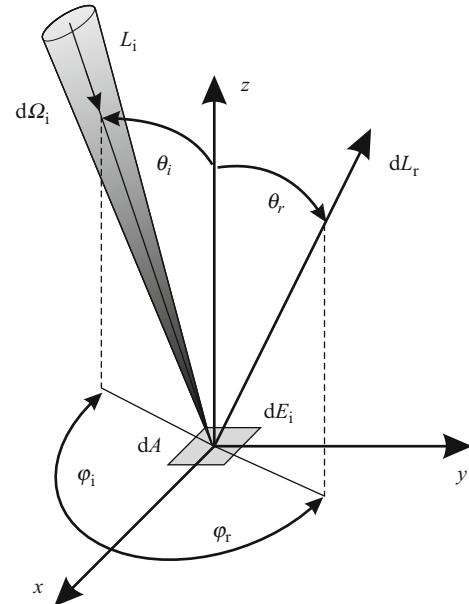
In general, the amount of reflected light intensity depends on the wavelength, incident angle, and polarization of the incident light and on the characteristics of the surface itself, e.g., refraction index, shape and roughness.

The Fresnel equations describe the relationship between reflected intensity, incident angle, and polarization state of the incident electromagnetic wave of a smooth surface. These formulas are applicable in the two cases of dielectric and strongly absorbing materials (metals), and establish the theoretical basis for the creation of polarized light with mirrors.

In Fig. 2, the reflectance of some metals is plotted against the wavelength. Most metals have a strong reflectance in the infrared spectrum. For laser applications, mirrors with gold coatings are often sufficient.

With dielectric films even higher reflectance values can be achieved.

Furthermore, the reflectance depends on the surface quality. The dependency on surface roughness σ (root-mean-squared roughness), wavelength λ , and the reflectances R_σ for rough and R for ideal smooth



Mirrors, Fig. 1 Geometry of reflection and the BRDF, thereby $d\Omega_i$ denotes an infinitesimal solid angle of the incident radiation, L_i , L_r the incident and reflected radiance, and dE_i the irradiance on the surface element dA

surfaces can be stated as [2]:

$$R_\sigma = R \exp \left[- \left(\frac{4\pi\sigma \cos \theta_i}{\lambda} \right)^2 \right]. \quad (4)$$

The roughness requirements in the far-infrared spectrum are lower than in the visible range. With large incident angles θ_i and surfaces with very small roughness, mirrors applicable even for X-radiation applications can be manufactured.

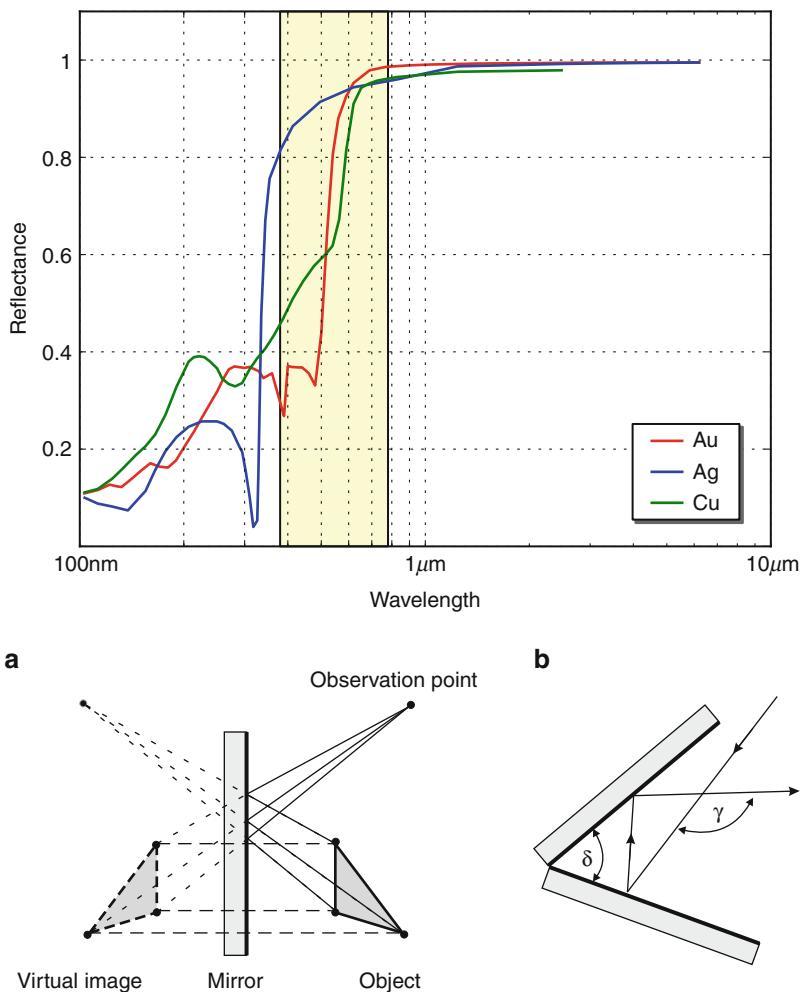
The most familiar type of mirror is the plane mirror, which has a flat surface. This mirror is mostly used for beam deflection purposes. In Fig. 3a, the geometry of reflection on a plane mirror is shown.

Thereby the image of an object is virtual with magnification equal to one, upright, right-left inverted, without aberrations, and symmetric to the mirror plane.

Figure 3b shows two plane mirrors in an angular mirror setup, whereby $\gamma = 2\delta$. A special case is a triple mirror with three pairwise orthogonal planes ($\delta = 90^\circ$), which is used as a retroreflecting element.

Curved mirrors are also used, such as spherical, ellipsoid, paraboloid, or conical mirrors. Figure 4

Mirrors, Fig. 2 Reflectance vs. wavelength curves for gold (Au), silver (Ag), and copper (Cu) at normal incidence [1]



Mirrors, Fig. 3 Plane (a) and angular mirror (b)

shows convex and concave mirrors for optical imaging. The focal distance of a spherical mirror with radius r is given by:

$$f = \frac{r}{2}. \quad (5)$$

The mirror equation:

$$\frac{2}{r} = \frac{1}{s} + \frac{1}{s'} \quad (6)$$

describes the relationship between object and image distances (s, s') with the mirror radius r .

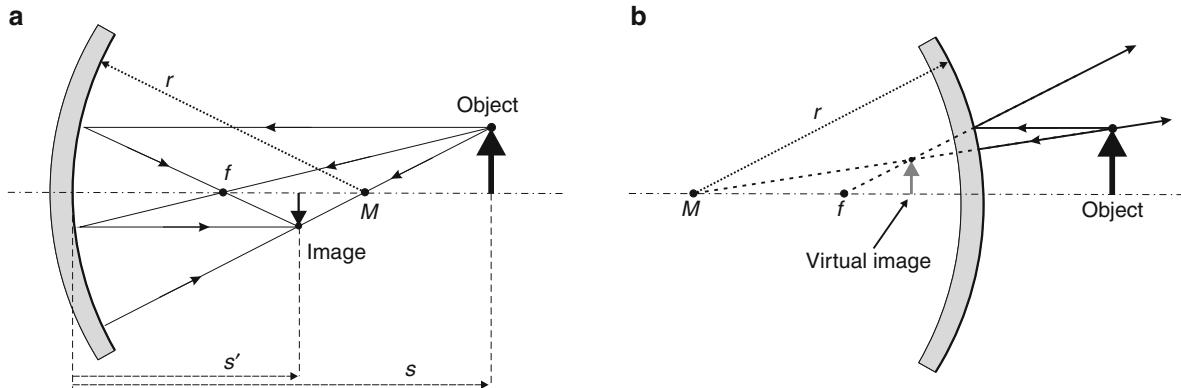
A big advantage of mirrors above lenses is the lack of aberrations, but with the disadvantage of higher centering and adjustment requirements.

Torrance and Sparrow [13] and Phong [11] have, among many others, introduced surface models which can be used to describe specular reflections. Modeling

of mirrors or partially reflecting surfaces is of ongoing interest for computer graphics applications.

Open Problems

The main principle for the visual inspection of mirrors is to use a highly controllable environment, where a screen presenting a well-designed pattern is observed via the specular reflecting surface. Knowing that pattern, it is possible to inspect the surface qualitatively and – at least with certain additional knowledge – to reconstruct the surface quantitatively. This reconstruction problem is ill-posed in a mathematical sense, and several regularization approaches have been proposed. The reconstruction of large and complex formed mirrors is still a challenge in the field of computer vision [6–8, 14, 15].



Mirrors, Fig. 4 Optical imaging: concave (a) and convex (b) mirror

Although the reconstruction problem is ill-posed, humans can usually estimate the shape of mirrors quite well, c.f., Fleming et al. [4]. The visual perception of mirror-like objects is an ongoing research effort.

Another area of ongoing research is the development of mirrors for the extreme ultraviolet (EUV) spectral range, used in EUV lithography tools. These mirrors can be standard Mo/Si mirrors or multilayer setups [12].

References

1. Bass M (ed) (2009) Handbook of optics, vol I–V. McGraw-Hill Professional, New York
2. Beckmann P, Spizzichino A (1963) The scattering of electromagnetic waves from rough surfaces. Pergamon Press, London
3. Born M, Wolf E (1999) Principles of optics: electromagnetic theory of propagation, interference and diffraction of light. Cambridge University Press, Cambridge, MA
4. Fleming RW, Torralba A, Adelson EH (2004) Specular reflections and the perception of shape. *J Vis* 4: 798–820
5. Horn BKP, Sjoberg RW (1979) Calculating the reflectance map. *Appl Opt* 18(11):1770–1779
6. Ihrke I, Kutulakos KN, Lensch HPA, Magnor M, Heidrich W (2008) State of the art in transparent and specular object reconstruction. In: Theoharis T, Dutre P (eds) STAR - State of the Art Report, EUROGRAPHICS 2008, Crete, Greece. The Eurographics Association, pp 87–108
7. Ikeuchi K (1981) Determining surface orientations of specular surfaces by using the photometric stereo method. *IEEE Trans Pattern Anal Mach Intell* 3(6):661–669
8. Nayar SK, Ikeuchi K, Kanade T (1991) Surface reflection: physical and geometrical perspectives. *IEEE Trans Pattern Anal Mach Intell* 13(7):611–634
9. Nicodemus FE, Richmond JC, Hsia JJ, Ginsberg IW, Limperis T (1977) Geometrical considerations and nomenclature for reflectance. Final report national bureau of standards, Washington, DC. Inst. for Basic Standards
10. Paschotta R (2008) Encyclopedia of laser physics and technology. Wiley-VCH, Berlin
11. Phong BT (1975) Illumination for Computer Generated Pictures. *Commun ACM* 18(6):311–317
12. Soer WA, Gawlitz P, van Herpen MMJW, Jak MJJ, Braun S, Muys P, Banine VY (2009) Extreme ultraviolet multilayer mirror with near-zero IR reflectance. *Opt Lett* 34(23): 3680–3682
13. Torrance KE, Sparrow EM (1967) Theory for off-specular reflection from roughened surfaces. *J Opt Soc America* 57(9):1105–1114
14. Wang Z, Inokuchi S (1993) Determining shape of specular surfaces. In: The 8th scandinavian conference on image analysis, Tromso, pp 1187–1194
15. Werling S, Mai M, Heizmann M, Beyerer J (2009) Inspection of specular and partially specular surfaces. *Metrology Meas Syst* 16(3):415–431

Mobile Observers

Jan-Olof Eklundh

School of Computer Science and Communication,
KTH - Royal Institute of Technology, Stockholm,
Sweden

Definition

A mobile (visual) observer is an agent or a system that perceives its environment using vision. In computer vision this typically is a mobile device such as a robot carrying one or more cameras.

Background

Gibson [1] claimed that a mobile observer is a prerequisite for natural vision. He discriminated between *ambient* or *ambulatory vision*, when the observer can move its head or body, and *snapshot* or *aperture vision* in cases when one or several images are recorded momentarily at certain fixation points. All those aspects are treated in computer vision, although current trends are on processing static images in the spirit of *snapshot* or *aperture vision*. Computer vision researchers began to study *visual motion* in the 1970s, when it became possible to connect video cameras to computers. This work did not really concern mobile observers, but such existed even earlier, when cameras were used as input devices to robots, e.g., in the work on “Shakey” [2]. Nowadays, mobile observers most often occur in the context of mobile robots, but recent developments on *wearable vision* have widened the interest in the topic. *Ambient vision* is what you have for instance in the case of *pan-tilt heads*, which are used in a large range of applications.

Theory

There have been attempts to find the notions of active and mobile observers theoretically. In biological vision Gibson’s work is of a landmark nature, but there are many other proposals as well, e.g., relating to functionalism [3]. In computer vision the problem has been considered from the point of view of active vs. passive vision [4–6]. In [7–9] the theoretical aspects are more directly addressed. However, even with these attempts, one can hardly say that there exists any complete theory for a mobile observer.

Problems and Applications

The mobile observer obtains a stream or sequence of images as input rather than single images. This provides rich information about the environment as well as of the movements of the observer. However, observer motion also implies that there is image motion in almost every point in the sequence. In a static world, observer motion creates essentially all the variations over time in the images, i.e., those that are due to change of viewpoint and not, e.g., in illumination. If there are things in the environment that also move, the two types of motion are confounded in the images.

A mobile observer can derive (static) scene geometry through *structure-from-motion* algorithms. Moreover, *ego-motion*, i.e., the motion of the observer, can be estimated. Generally such methods assume a static background that is prominent in the field of view. Independently moving objects can then also be detected, and under certain conditions their motion can be estimated. *Ego-motion estimation* obviously plays an important role here. There are many types of algorithms for this, e.g., based on *optical flow*, monocular or binocular *feature tracking*, or *image stabilization*. There also exist algorithms for using *omnidirectional* or *composite cameras*, which highlights the fact that effects of ego-motion are manifested in a wide field of view. For instance, small rotations of an observer moving straight ahead can be estimated from peripheral flow, something that is useful in driving and in guiding of autonomous robots.

A mobile observer can be active or passive. In the first case, it purposively guides its motion and/or the way it directs its gaze on the basis of tasks it is involved in and as a reaction to what it observes. *Gaze control* and *fixation* in dynamic situations have been studied extensively in the field of *active vision*. In some cases these mechanisms have been used to control observer motion, e.g., for exploring a scene or an object and to facilitate recognition. Then *viewpoint planning* becomes an issue. However, a more general case is when the observer motion is only loosely dependent of what is seen, except for possible control of gaze. For instance, a mobile observer can induce depth cues through parallax by (small) camera motions that are not pure rotations. Another example is given by a robot moving from one point to another while observing an object along its path, analogously to a person riding in a car. Many applications contain elements of both active and passive observations, for instance in robot navigation including *obstacle avoidance* and *mapping* (as in *SLAM*), *hand-eye control* in grasping and manipulation, and in general for an ambulant observer, such as those studied in the context of *wearable* or *egocentric vision*.

Open Problems

The study of mobile observers from a computational perspective involves a broad range of problems traditionally addressed in computer vision. However,

there are certain issues that become central. For instance, the *correspondence problem* is ubiquitous. In applications such as those described above, the tight connection between *perception and action* is apparent. Visual sensing involving motor control raises problems on time criticality and real-time computations [9]. Other problems arise because the mobile observer continuously samples the visual world. Meaningful behavior based on the huge amounts of information requires methods for *attention* and *visual search*. In all, although some of the problems encountered in the study of mobile observers largely overlap those generally treated in computer vision, there are others that are specific to this area.

References

1. Gibson JJ (1979) The ecological approach to visual perception. Houghton Mifflin, Boston
2. <http://www.ai.sri.com/shakey/>
3. O'Regan JK, Noe A (2001) A sensorimotor account of vision and visual consciousness. *Behav Brain Sci* 24:939–1031
4. Bajcsy R (1985) Active perception vs. passive perception. In: Proceedings of the 3rd IEEE workshop on computer vision, Bellaire. IEEE CS Press, pp 55–59
5. Aloimonos Y, Weiss I, Bandyopadhyay A (1987) Active vision. In: Proceedings of the 1st ICCV, London. IEEE CS Press, pp 35–54
6. Ballard DH (1991) Animate vision. *Artif Intell* 48:57–86
7. Tsotsos JK (1992) On the relative complexity of active vs. passive visual search. *Int J Comput Vis* 7:127–141
8. Bennett BM, Hoffman DD, Prakash C (1989) Observer mechanics: a formal theory of perception. Academic, San Diego
9. Soatto S (2011) Steps toward a theory of visual information. CoRR. MIT, Cambridge

MoCap

- Motion Capture

Model-Based Object Recognition

Min Sun and Silvio Savarese
Department of Electrical and Computer Engineering,
University of Michigan, Ann Arbor, MI, USA

Synonyms

[Object models](#); [Object parameterizations](#); [Object representations](#); [Visual patterns](#)

Related Concepts

- [Human Pose Estimation](#); ► [Object Class Recognition \(Categorization\)](#); ► [Object Detection](#)

Definition

Model-based object recognition addresses the problem of recognizing objects from images by means of a suitable mathematical model that is used to describe the object.

Background

In model-based object recognition, an object model is typically defined so as to capture object's geometrical and appearance properties at the appropriate level of specificity. For instance, an object model can be designed to recognize a generic "face" as opposed to "someone's face" or vice versa. In the former case, which is often referred to as the object categorization problem, the main challenge is to design models that are capable of retaining key visual properties for representing an object category, such as a "face," at the appropriate level of abstraction. Such models can be then used to recognize novel object instances from a query image. Moreover, a model must be able to generalize across variations in the object's visual characteristics due to viewpoint and illumination changes as well as due to occlusions or deformations. Meeting all of these desiderata can be extremely challenging. This makes object recognition an open, yet key, problem in computer vision.

Object Models for Recognition

The design of an object model must reflect its ability to (i) capture geometrical and appearance characteristics of the object at the appropriate level of specificity and (ii) generalize across variations in viewpoint, illumination, occlusions, and deformations. The complexity of the representation can be reduced by making assumptions on the type of object specificity or the degree of viewpoint, occlusions, and deformation variability. Ultimately, the strategy in designing an object model will depend on the relevant application scenario.

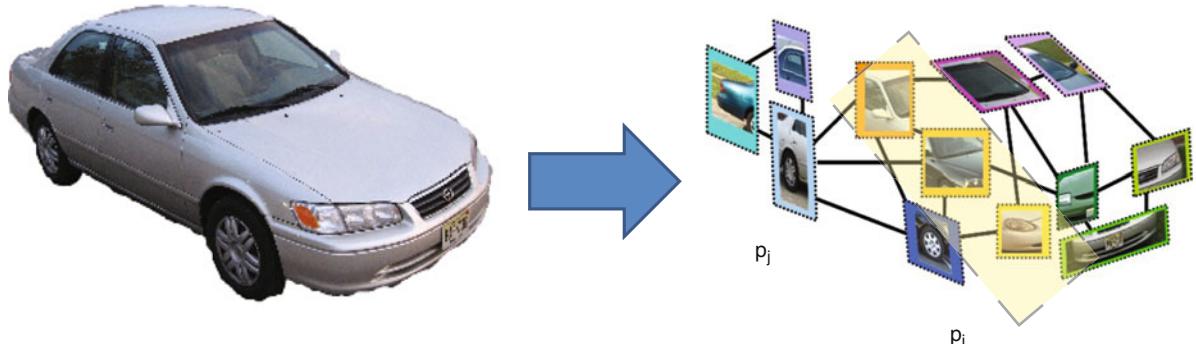
Object models that are designed to recognize objects at the highest level of specificity – e.g., “my face” as opposed to “a face” – are often referred to as single-instance object models. These models are capable of recognizing a specific object instance while guaranteeing the ability to handle occlusions and a large degree of viewpoint variability. Research on 3D object recognition, from early contributions [1–9] to the most recent ones [10–13], follows these assumptions. Since single-instance object models do not need to accommodate any intra-class variations, they often consist of a rigid collection of visual features associated to a number of 2D or 3D templates. In recognition, by matching features of the query image with those associated to the models, it is possible to identify the object of interest and determine its 3D pose with respect to a common reference system. This matching process is usually subject to a geometrical validation phase that helps verify that the appearance, and geometric properties of the query object are consistent with the estimated pose transformation between observation and object model. While critical for ensuring sufficient discrimination power for recognizing single-instance objects as well as for enabling large viewpoint variability, tight geometrical constraints become inadequate when shape and appearance intra-class variability must be accounted for.

Object models that are designed to recognize objects at a lower level of specificity – e.g., “a face” as opposed to “my face” – are often referred to as categorical object models. The ability to generalize across instances in the same category is critical and is typically achieved by characterizing the object as a collection of features whose appearance and geometrical properties tend to systematically occur in the category of interest. For instance, if the goal is to recognize a car, appearance properties such as the “color of the body” are not adequate to help obtain the right level of generalization (abstraction), whereas the orientation of edges associated to a wheel can capture more general appearance cues across instances. Appearance properties are typically captured by image descriptors such as [10, 14] associated to interest points that are detected at different locations and scales of the image. A popular design choice is to describe the object appearance by histograms of vector-quantized descriptors [15–17]. The ability of image descriptors such as [10] to be invariant to affine illumination transformations makes the appearance models robust

to variability in illumination conditions. Geometrical properties are captured by retaining the spatial organization of features in the image and include simple characterizations based on the 2D location of either feature points or aggregation of features (e.g., edges, parts, fragments) with respect to a given object reference point [18–22]. Object models constructed upon constellation of parts such as [18–20] are suitable to accommodate object variations due to occlusions and simple 2D planar geometrical deformations (isometries or affinities). Suitable machine learning and probabilistic inference techniques such as expectation maximization (EM) [23], latent SVM (LSVM), [54] Markov random field (MRF) [24, 25], conditional random field (CRF) [26], generalized Hough voting [27], and RANdom SAmple Consensus (RANSAC) [28] are used to automatically select appearance and geometrical properties so as to reach the appropriate level of generalization and discrimination power.

Most of the object models for object categorization mitigate the complexity of the representation by assuming that objects are viewed from a limited number of poses and learn an object model that is specialized to identify the object from a specific viewpoint. These are often referred to as view-dependent object models. If similar views in the training set are available, the recognition problem is reduced to match the new query object to one, or a mixture, of the learnt view-dependent object models [29, 30]. The drawback of view-dependent object models is that (i) they can accommodate very limited viewpoint variability – mostly changes in scale or 2D rotation transformations – and (ii) different poses of the same object category result in completely independent models, where neither features or parts are shared across views. Because each single-view models are independent, these methods are often costly to train and prone to false alarms, if several views need to be encoded.

Object models that can accommodate both large viewpoint changes and large intra-class variability (low degree of specificity) overcome the above limitations by introducing a representation that seeks to effectively captures the intrinsic three-dimensional nature of the object category. These models are typically divided into two types: 2-1/2D layout models and 3D layout models [33]. In the 2-1/2D layout models [31, 32, 34], object diagnostic elements (features, parts, contours) are connected across views to form an unique and coherent 2-1/2D model for the



Model-Based Object Recognition, Fig. 1 Example of 2-1/2D layout models as introduced in [31] and generalized in [32]. *Left panel:* An image of an object category of interest. *Right panel:* In the 2-1/2D layout model, object parts are connected to form a graph structure. Each node P_i captures diagnostic

appearance of the object part which is assumed to be locally planar. Each edge describes an homographic transformation that captures the viewpoint transformation between parts. The homographic transformation is illustrated by showing that some parts are slanted with respect to others

object category (Fig. 1). Relationships between features or parts capture the way that such elements are transformed as the viewpoint changes. These methods share some key ideas with pioneering works in 3D object recognition [1–6, 8, 9] as well as with the theory of aspect graphs [7, 35]. In the 3D layout models [36–41], object elements are organized in a common 3D reference frame and form a compact 3D representation of the object category. Such 3D structures of features (parts, edges) can give rise, for instance, to either a 3D generalization of 2D pictorial structures or constellation models or to hybrid models where features (parts or edges) lie on top of 3D object reconstructions or CAD volumes.

Open Problems

Although object recognition has been a core problem in computer vision for more than four decades and several powerful models have been proposed, state-of-the-art methods are still far from the level of accuracy, efficiency, and robustness that the human visual system achieves in recognizing, detecting, and categorizing objects from images. Recently, several new paradigms have been explored to address the above limitations. One major effort involves large-scale object recognition. With the introduction of ultra-large-scale datasets such as the ImageNet [42] – a collection of millions of images organized into a hierarchical ontology of thousands of categories – it is now possible to evaluate

methods for object categorization that seek to (i) efficiently process these many images and categories and (ii) understand objects at different level of specificity; this is also referred to as to the fine-grain categorization problem [43–45]. Another major effort is related to the introduction of a recent paradigm whereby objects are modeled and recognized by means of their attributes. As pioneered by [46–48], visual attributes such as “it is metallic”; “it has wheels” can be used to obtain more effective and descriptive characterizations of object categories (i.e., a car or a truck). This has the benefit of (i) making the “boundaries” between different categories more fluid than in traditional parameterizations, (ii) enabling more powerful methods for fine-grained categorization [44], and (iii) providing critical building blocks for transferring visual properties across categories (transfer learning, one shot learning) [46, 48].

Other important problems for future work include the ability to (i) overcome the traditional paradigm whereby objects are identified as just bounding boxes in images but rather provide a richer characterization in terms of their accurate outlines or segments, 3D properties (pose or 3D shape) [36, 41], as well as attributes; (ii) find a common ground between bottom-up representations (from pixels to features), akin to recent developments on convolutional neural networks [49, 50], and top-down models as recently advocated in [51] and (iii) describe the interplay between objects and their components at different levels of semantic resolution [52, 53].

References

1. Binford T (1971) Visual perception by computer. IEEE conference on systems and control
2. Marr D (1978) Representing visual information. Computer vision systems
3. Palmer S, Rosch E, Chase P (1981) Canonical perspective and the perception of objects. *Atten Perform* 9:135–151
4. Tarr M, Pinker S (1989) Mental rotation and orientation-dependence in shape recognition. *Cogn Psychol* 21: 233–282
5. Poggio T, Edelman S (1990) A neural network that learns to recognize three-dimensional objects. *Nature* 343:263–266
6. Ullman S, Basri R (1991) Recognition by linear combinations of models. *TPAMI* 13:992–1006
7. Koenderink J, Doorn AV (1979) The internal representation of solid shape with respect to vision. *Biol Cybern* 32: 211–216
8. Huttenlocher DP, Ullman S (1987) Object recognition using alignment. In: *ICCV*
9. Lowe D, Binford T (1985) The recovery of three-dimensional structure from image curves. *TPAMI* 7: 320–326
10. Lowe DG (1999) Object recognition from local scale-invariant features. In: *ICCV*
11. Rothganger F, Lazebnik S, Schmid C, Ponce J (2003) 3D object modeling and recognition using affine-invariant patches and multi-view spatial constraints. In: IEEE conference on computer vision pattern recognition (CVPR)
12. Brown M, Lowe DG (2005) Unsupervised 3D object recognition and reconstruction in unordered datasets. In: *3DIM*
13. Ferrari V, Tuytelaars T, Gool L (2006) Simultaneous object recognition and segmentation from single or multiple model views. *IJCV* 67:159–188
14. Mikolajczyk K, Schmid C (2002) An affine invariant interest point detector. In: European conference on computer vision (ECCV)
15. Dance C, Willamowski J, Fan L, Bray C, Csurka G (2004) Visual categorization with bags of keypoints. In: European conference on computer vision (ECCV) international workshop on statistical learning in computer vision, Prague
16. Grauman K, Darrell T (2005) The pyramid match kernel: Discriminative classification with sets of image features. In: *ICCV*
17. Fei-Fei L, Fergus R, Perona P (2004) Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In: IEEE conference on computer vision pattern recognition (CVPR)
18. Fergus R, Perona P, Zisserman A (2003) Object class recognition by unsupervised scale-invariant learning. In: IEEE conference on computer vision pattern recognition (CVPR)
19. Felzenszwalb PF, Huttenlocher DP (2005) Pictorial structures for object recognition. *IJCV* 61(1):55–79
20. Leibe B, Leonardis A, Schiele B (2004) Combined object categorization and segmentation with an implicit shape model. In: European conference on computer vision (ECCV) workshop on statistical learning in computer vision
21. Lazebnik S, Schmid C, Ponce J (2006) Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: IEEE conference on computer vision pattern recognition (CVPR)
22. Savarese S, Winn J, Criminisi A (2006) Discriminative object class models of appearance and shape by correlatons. In: IEEE conference on computer vision pattern recognition (CVPR)
23. Dempster A, Laird N, Rubin D (1977) Maximum likelihood from incomplete data via the em algorithm. *J R Stat Soc* 39:1–38
24. Koller D, Friedman N (2009) Probabilistic graphical models: principles and techniques. MIT
25. Wainwright MJ, Jordan MI (2008) Graphical models, exponential families, and variational inference. *Found Trends Mach Learn* 1:1–305
26. Lafferty J, McCallum A, Pereira F (2001) Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: *ICML*
27. Ballard D (1981) Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognit* 13:111–122
28. Fischler MA, Bolles RC (1981) Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun ACM* 24:381–395
29. Schneiderman H, Kanade T (2000) A statistical approach to 3D object detection applied to faces and cars. In: IEEE conference on computer vision pattern recognition (CVPR)
30. Weber M, Einhaeuser W, Welling M, Perona P (2000) Viewpoint-invariant learning and detection of human heads. In: International conference on automatic face and gesture recognition
31. Savarese S, Fei-Fei L (2007) 3D generic object categorization, localization and pose estimation. In: *ICCV*
32. Su H, Sun M, Fei-Fei L, Savarese S (2009) Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In: *ICCV*
33. Hoiem D, Savarese S (2011) Representations and techniques for 3D object recognition and scene interpretation. In: *Synthesis lecture on artificial intelligence and machine learning*. Morgan Claypool, San Rafael
34. Thomas A, Ferrari V, Leibe B, Tuytelaars T, Schiele B, Goo LV (2006) Towards multi-view object class detection. In: IEEE conference on computer vision pattern recognition (CVPR)
35. Bowyer K, Dyer CR (1990) Aspect graphs: An introduction and survey of recent results. *Int J Imaging Syst Technol* 2:315–328
36. Sun M, Bradski G, Xu BX, Savarese S (2010) Depth-encoded hough voting for joint object detection and shape recovery. In: European conference on computer vision (ECCV)
37. Hoiem D, Rother C, Winn J (2007) 3D layoutcrf for multi-view object class recognition and segmentation. In: IEEE conference on computer vision pattern recognition (CVPR)
38. Liebelt J, Schmid C (2010) Multi-view object class detection with a 3D geometric model. In: IEEE conference on computer vision pattern recognition (CVPR)
39. Pepik B, Stark M, Gehler P, Schiele B (2012) Teaching 3D geometry to deformable part models. In: IEEE conference on computer vision pattern recognition (CVPR)
40. Arie-Nachimson M, Basri R (2009) Constructing implicit 3D shape models for pose estimation. In: *ICCV*

41. Xiang Y, Savarese S (2012) Estimating the aspect layout of object categories. In: IEEE conference on computer vision pattern recognition (CVPR)
42. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) Imagenet: a large-scale hierarchical image database. In: IEEE conference on computer vision pattern recognition (CVPR)
43. Yao B, Bradski G, Fei-Fei L (2012) A codebook-free and annotation-free approach for fine-grained image categorization. In: IEEE conference on computer vision pattern recognition (CVPR)
44. Duan K, Parikh D, Crandall D, Grauman K (2012) Discovering localized attributes for fine-grained recognition. In: IEEE conference on computer vision pattern recognition (CVPR)
45. Perona P (2010) Visions of a visipedia. Proc IEEE 98: 1526–1534
46. Farhadi A, Endres I, Hoiem D, Forsyth D (2009) Describing objects by their attributes. In: IEEE conference on computer vision pattern recognition (CVPR), Miami
47. Ferrari V, Zisserman A (2007) Learning visual attributes. In: NIPS
48. Lampert CH, Nickisch H, Harmeling S (2009) Learning to detect unseen object classes by between-class attribute transfer. In: IEEE conference on computer vision pattern recognition (CVPR)
49. Lee H, Grosse R, Ranganath R, Ng AY (2009) Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: ICML
50. Yann LeCun FJH, Bottou L (2004) Learning methods for generic object recognition with invariance to pose and lighting. In: IEEE conference on computer vision pattern recognition (CVPR)
51. Arbelaez P, Maire M, Fowlkes C, Malik J (2011) Contour detection and hierarchical image segmentation. IEEE Trans Pattern Anal Mach Intell 33(5):898–916
52. Zhu L, Chen Y, Yuille A (2006) Unsupervised learning of a probabilistic grammar for object detection and parsing. In: NIPS
53. Todorovic S, Ahuja N (2008) Unsupervised category modeling, recognition, and segmentation in images. IEEE Trans Pattern Anal Mach Intell 30(12):2158–2174
54. Felzenszwalb P, Girshick R, McAllester D, Ramanan D (2010) Object detection with discriminatively trained part-based models. TPAMI 32:1627–1645

Monte Carlo Annealing

► [Simulated Annealing](#)

Morphology, Form Analysis

► [Statistical Shape Analysis](#)

Motion Blur

Neel Joshi

Microsoft Corporation, Redmond, WA, USA

Synonyms

[Camera-shake blur](#); [Object motion blur](#)

Related Concepts

► [Blur Estimation](#); ► [Defocus Blur](#)

Definition

Motion blur is due to motion of scene objects or the camera while the camera shutter is open, thus causing scene points to be imaged over a large area of camera sensor or film. The motion blur is a projection of the motion path of the moving objects onto the image plane. The motion path of a point can be due to translation and rotation of the camera or scene objects in three dimensions. There can be different paths for different parts of the scene, and in light-limited situations, when using long exposures, these paths can be quite large, resulting in very large blurs.

Background

Image blur can be described by a point spread function (PSF). A PSF models how an imaging system captures a single point in the world – it literally describes how a point “spreads” across an image. An entire image is then made up of a sum of the individual images of every scene point, where each point’s image is affected by the PSF associated with that point. For an image to be “sharp” means that one ideally does not want any image blur. Thus, the PSF should be minimal, i.e., a delta function, where each scene point should correspond only to one image point. In practice, PSFs can take on a range of shapes and sizes depending on the properties of an imaging system. When this PSF is



Motion Blur, Fig. 1 With motion blur, the amount of blur depends on the relative motion between the camera and the scene objects; it depends on the focal length and of the lens and the scene depth and motion trajectories. An example of

camera motion blur is shown in the *middle*, where the blur kernel is drawn for each corner of the image (From Joshi et al. [3]). An example of object motion blur is shown on the *right* (From Jia [11])

large relative due to camera or scene motion and relative to the image resolution and pixel size, an image with motion blur is captured.

The fundamental cause of motion blur is that a camera does not sample light from a single moment in time, but instead captures images by integrating the light over an exposure window. The relative motion between camera and scene objects is the primary factor in motion blur, as illustrated in Fig. 1. The path of motion during exposure affects the PSF and thus the blur shape and size. Properties such as exposure duration, lens focal length, and pixel size play an additional role.

by a sparse re-sampling matrix that models the spatially variant blur, and the convolution process is now a matrix-vector product:

$$\mathbf{b} = \mathbf{A}\mathbf{i} + \mathbf{n}. \quad (2)$$

Each column of \mathbf{A} is the unraveled kernel for the pixel represented by that column. Thus, the blurred response at a pixel in the observed image is computed as a weighted sum, as governed by \mathbf{A} , of the latent sharp image \mathbf{i} formed into a column vector.

M

Theory

Image blur is described by a point spread function (PSF). The PSF models how an imaging system captures a single point in the world.

The most commonly used model for blur is the linear model, where the blurred image \mathbf{b} is represented as a convolution of a kernel \mathbf{k} , plus noise:

$$\mathbf{b} = \mathbf{i} \otimes \mathbf{k} + \mathbf{n}, \quad (1)$$

where $\mathbf{n} \sim \mathcal{N}(0, \sigma^2)$, which represents an additive Gaussian noise model. In this model, the blur is assumed to be constant over the entire image, i.e., spatially invariant; however, that is often not true in practice [1, 2]. If there is depth variation in the scene, the motion blur can change with that depth due to parallax. In these cases, one can think of the blur kernel, \mathbf{k} , as being a function of image position, i.e., $\mathbf{k}(x, y)$.

To model spatially varying blur, the spatially invariant kernel and convolution in (Eq. 1) can be replaced

Representation

To model motion blur, first, let us consider the image a camera captures during its exposure window. The intensity of light from a scene point (X_t, Y_t, Z_t) at an instantaneous time t is captured on the image plane at a location (u_t, v_t) , which is a function of the camera projection matrix P_t . In homogenous coordinates, this can be written as

$$(u_t, v_t, 1)^T = P_t(X_t, Y_t, Z_t, 1)^T. \quad (3)$$

If there is camera motion, P_t varies with time as a function of camera rotation and translation causing points in the scene to project to different locations at each time. If there is scene motion, (X_t, Y_t, Z_t) also varies with time, which also affects where the points project on the image plane. The integration of these projected observations creates a blurred image, and the projected trajectory of each point on the image plane is that point's point spread function (PSF). The camera projection matrix can be decomposed as

$$P_t = K \Pi E_t, \quad (4)$$

where K is the intrinsic matrix, Π is the canonical perspective projection matrix, and E_t is the time-dependent extrinsic matrix that is composed of the camera rotation R_t and translation T_t . In the case of image blur, it is not necessary to consider the absolute motion of the camera, only the relative motion and its effect on the image. This can be modeled by considering the planar homography that maps the initial projection of points at $t = 0$ to any other time t [3], i.e., the reference coordinate frame is coincident with the frame at time $t = 0$:

$$H_t(d) = \left[K(R_t + \frac{1}{d}T_t N^T)K^{-1} \right] \quad (5)$$

$$(u_t, v_t, 1)^T = H_t(d)(u_0, v_0, 1)^T, \quad (6)$$

for a particular depth d , where N is the unit vector that is orthogonal to the image plane.

If the scene is not moving, given an image I at time $t = 0$, the pixel value of any subsequent image is

$$I_t(u_t, v_t) = I(H_t(d)(u_0, v_0, 1)^T). \quad (7)$$

This image warp can be rewritten in matrix form as

$$\mathbf{I}_t = A_t(d)\mathbf{I}, \quad (8)$$

where \mathbf{I}_t and \mathbf{I} are column-vectorized images and $A_t(d)$ is a sparse re-sampling matrix that implements the image warping and re-sampling due to the homography. Each row of $A_t(d)$ contains the weights to compute the value at pixel (u_t, v_t) as the interpolation of the point $(u_0, v_0, 1)^T = H_t(d)^{-1}(u_t, v_t, 1)^T$. Thus, an alternative formulation for image blur is the integration of applying these homographies over time [3]:

$$\mathbf{B} = \int_0^s [A_t(d)\mathbf{I} dt]. \quad (9)$$

This leads to the spatially variant blur matrix in (Eq. 2):

$$A(d) = \int_0^s A_t(d)dt. \quad (10)$$

For camera motion blur, A is a function of depth. If there is scene motion, the full model can be extended to handle the time-varying mapping of scene points, (X_t, Y_t, Z_t) , to the image plane.

Application

Estimation of camera motion blur [1, 3–9] and estimation of object motion blur [10–13] are extensively researched areas.

Estimated blur kernels are typically used for improving image quality by reducing blur using image deblurring and deconvolution methods [1, 10, 14, 15]. There are also methods that reduce motion blur or make the blur more easily removable but changing how images are captured [13, 16].

References

- Joshi N, Szeliski R, Kriegman DJ (2008) Psf estimation using sharp edge prediction. In: Computer vision and pattern recognition, 2008 (CVPR 2008). IEEE conference on, Anchorage, pp 1–8
- Levin A, Weiss Y, Durand F, Freeman W (2009) Understanding and evaluating blind deconvolution algorithms. In: Computer vision and pattern recognition, 2009 (CVPR 2009). IEEE conference on, Miami (Beach), IEEE Computer Society, pp 1964–1971
- Joshi N, Kang SB, Zitnick CL, Szeliski R (2010) Image deblurring using inertial measurement sensors. ACM Trans Graph 29:30:1–30:9
- Bascle B, Blake A, Zisserman A (1996) Motion deblurring and super-resolution from an image sequence. In: ECCV '96: Proceedings of the 4th European conference on computer vision-vol II, Springer, London, pp 573–582
- Fergus R, Singh B, Hertzmann A, Roweis ST, Freeman WT (2006) Removing camera shake from a single photograph. ACM Trans Graph 25:787–794
- Yuan L, Sun J, Quan L, Shum HY (2007) Image deblurring with blurred/noisy image pairs. In: SIGGRAPH '07: ACM SIGGRAPH 2007 papers, ACM, New York, pp 1
- Ben-Ezra M, Nayar S (2004) Motion-based motion deblurring. IEEE Trans Pattern Anal Mach Intell 26(6): 689–698
- Tai YW, Du H, Brown MS, Lin S (2008) Image/video deblurring using a hybrid camera. In: Computer vision and pattern recognition, 2008 (CVPR 2008). IEEE conference on, Anchorage, pp 1–8
- Park SY, Park ES, Kim HI (2008) Image deblurring using vibration information from 3-axis accelerometer. J Inst Electron Eng Korea. SC, Syst control 45(3):1–11
- Levin A (2006) Blind motion deblurring using image statistics. In: Weiss Y et al (eds) Advances in neural information processing systems. MIT, Cambridge
- Jia J (2007) Single image motion deblurring using transparency. Computer vision and pattern recognition, 2007 (CVPR '07). IEEE conference on, Minneapolis, pp 1–8
- Qi Shan WX, Jia J (2007) Rotational motion deblurring of a rigid object from a single image. In: ICCV '07, Rio de Janeiro

13. Levin A, Sand P, Cho TS, Durand F, Freeman WT (2008) Motion-invariant photography. *ACM Trans Graph* 27:71:1–71:9
14. Richardson WH (1972) Bayesian-based iterative method of image restoration. *J Opt Soc Am (1917–1983)* 62:55–59
15. Levin A, Fergus R, Durand F, Freeman WT (2007) Image and depth from a conventional camera with a coded aperture. In: *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, ACM, New York, p 70
16. Raskar R, Agrawal A, Tumblin J (2006) Coded exposure photography: motion deblurring using fluttered shutter. *ACM Trans Graph* 25:795–804

Motion Capture

Nils Hasler
 Graphics, Vision & Video, MPI Informatik,
 Saarbrücken, Germany

Synonyms

[MoCap](#); Motion capturing; Motion tracking; Performance capture

Related Concepts

► [Kinematic Motion Models](#); ► [Multiview Stereo](#)

Definition

Motion capture is the process of recording the motion of a subject, processing it on a computer, and mapping it onto a virtual character.

Background

Parameterizing human motion is not just of academic interest, e.g., for studying the musculoskeletal system of humans, but has many applications in the industry. Historically, the first motion capture systems have been developed in the 1970s and 1980s to perform gait analysis in clinical settings. Today, however, sports sciences and the entertainment industry make heavy use of the technology as well. The setups are also not constrained to human gait analysis any longer. Instead, full-body motion of several humans, animals, stage

props, and virtual cameras can be processed in real time.

When capturing motion, it is commonly assumed that the body or object can be decomposed into rigidly moving parts connected by joints. That way, the pose of a human, animal, or mechanical stage prop can be parameterized by a small set of joint angles organized in a hierarchical tree, the skeleton. This hierarchy can be inferred given trajectories of markers attached to the body. Yet, since this step is computationally expensive, the skeleton is normally supplied and scaled to the size of the actor beforehand.

Classification

Motion capture systems can be categorized by their use of sensors. Optical systems use cameras operating in the visible or infrared spectrum, whereas nonoptical systems are based on various other modalities.

Nonoptical Systems

Various nonoptical motion capture systems have been proposed using different sensors. They are grouped here because compared to optical motion capture, they occupy a marginal position. Yet, all approaches discussed here, have in common that they solve the main disadvantage of optical systems, the sensitivity to occlusion.

Mechanical tracking systems measure the angles of the joints mechanically, i.e., by attaching goniometers to the joints of the subject. Estimating the pose given the joint angles is straightforward, but several problems exist with the approach. The mechanical alignment of the goniometers with the body joints can be difficult, especially for joints with more than one degree of freedom, e.g., the shoulder, the devices are cumbersome, and limb lengths have to be measured very accurately for every subject to prevent drift.

Magnetic fields can be used to estimate the orientation of a magnetic sensor relative to the source of the field. By modulating magnetic coils in the vicinity of the capture volume and measuring the field at different points in time, position and orientation of the sensor can be inferred. The approach has the advantage that it does not suffer from occlusion because the human body is transparent to magnetic fields. However, magnetic fields attenuate rapidly over distance and are sensitive to interference with electrical equipment. The latter is a severe shortcoming as motion

capture systems are frequently used in conjunction with other equipment such as motion picture cameras, computers, or stage lights.

Inertial sensors measure acceleration and angular velocity of the limbs they are attached to. Aside from the acceleration of the body, accelerometers measure the gravitational acceleration. After compensating for gravitation, integrating measurements over time yields the pose of the subject. This approach is, like the previous methods, invariant to occlusion, but the numerical integration of measurements leads to drift in position and orientation. It can, however, be used effectively in conjunction with a drift-free method to compensate for drift and bridge occlusions with the acceleration data.

Optical Systems

The most common systems today are optical, i.e., they use one or more calibrated camera(s) to estimate the pose of the subject. Most commercial systems today require the actor to wear markers to simplify the tracking. One of the first marker-based systems used pieces of paper that glow under ultraviolet illumination. Nowadays, either small retroreflective balls or light-emitting diodes (LEDs) operating in the visible or infrared spectrum are used.

Passive systems are normally equipped with infrared lights located in rings around the cameras. This setup evokes a distinct dot for each retroreflective marker in the captured video frames. To further improve image quality, infrared filters in front of the cameras help to reduce spurious highlights in the visible spectrum. There are two main drawbacks of passive marker-based systems. Since all markers look alike, their trajectories can easily be confused, and all optical systems have in common that they suffer from occlusions.

Active markers, e.g., pulsed LEDs, have the advantage that markers can be identified by the blinking pattern. That way, confusing marker trajectories becomes impossible. Additionally, active marker systems are not restricted to studio environments because the blinking patterns can be distinguished effectively from interference introduced by sunlight. One disadvantage of active markers is that they have to be powered. Carrying a battery pack is not a big burden for an actor, but some stage props such as arrows cannot easily be fitted with batteries.

Markerless systems are subject of intense scientific research. However, first commercial systems are available as well. Yet, it is still unclear, which of the proposed approaches will prove to be the most effective in the long run. Most vision-based approaches today use a combination of edge features, silhouette constraints, texture analysis, and feature tracking or optical flow. Pose optimization is performed using gradient descent, particle filters, simulated annealing, belief propagation, or a combination of these methods. The main advantage of markerless systems is that the amount of preparation of the subject is minimal. Common drawbacks of current systems are that they lack robustness or restrict the setting in other ways than by adding markers, e.g., it is assumed that the background is static. Markerless systems also tend to be computationally expensive. The most advanced systems today are able to capture a single actor in real time, compared to five actors with a marker-based system.

Theory

Generally, the different motion capture systems require different workflows to set up the system and then to fit a skeleton to the captured data. Since optical systems are the most common, in the following, the procedure for this particular pipeline is outlined.

As a first step, all camera-based systems today need to be calibrated. That is, the relative positions of the cameras, their orientations, and their internal parameters (distortion) have to be estimated. This is commonly achieved by either placing a three-dimensional calibration object with known dimensions inside the capture volume or by covering the volume with a calibration wand, a typically one-dimensional object with known dimensions and marker positions. The advantage of wand-based calibration is that it is easier to cover the entire capture volume using only a small object. Covering the capture volume as exhaustively as possible is important to ensure accuracy of the calibration. Calibration objects, in contrast, tend to be as large as possible for a similar reason. In either case, the parameters of the cameras are extracted using a variation of Structure from Motion.

When using marker-based systems, the next step is to extract marker positions in the video frames. Normally, centroids and extents of the markers are extracted. The extents can be used as a measure of

quality of the marker. For example, distant markers tend to be smaller and should be considered less reliable. For large systems, with dozens of cameras running at high framerates ($>[100]\text{Hz}$) and high resolution ($>[4]\text{Megapixel}$), doing this processing close to the cameras is essential because the required bandwidth for transmitting the entire frames to a central processing unit would be prohibitively expensive.

Subsequently, the 2D dots can be combined into 3D markers using the calibration matrices of the cameras. For passive markers, this step is ambiguous. So additional heuristics have to be taken into account. For example, dots can be tracked in 2D or 3D to propagate the identity of a marker, established in a previous frame to the current frame. 3D markers should also be consistent with as many of the 2D dots as possible. Finally, skeleton fitting can be posed as a nonlinear minimization problem

$$\operatorname{argmin}_{\xi} \sum_{i=1}^M (m_i - s_i(\xi))^2, \quad (1)$$

where m_i is an estimated 3D marker position and $s_i(\xi)$ is the corresponding marker attached to the skeleton, as a function of the pose parameters ξ . This problem can be solved in many different ways. Commonly, gradient descent, Gauss-Newton, or Levenberg-Marquardt optimization is used. These approaches make use of the tracking assumption, i.e., the solution of the previous frame is used as a starting point for the current frame. Other methods, like particle filters or simulated annealing, are less likely to lose track by getting stuck in a local minimum but are computationally more expensive.

Alternatively, the 3D marker reconstruction step can be skipped. Instead, the nonlinear optimization is performed in image space of the cameras. That is, the distances between the projections of the skeleton's markers and the 2D markers are minimized rather than their distance in world space.

Markerless systems have to solve a very similar optimization problem. Only their method of acquiring correspondences between 2D image features and the tracked object is more sophisticated. Frequently, different characteristics are combined. Common features include edges, silhouettes, texture statistics, and corners.

Application

Historically, the first applications of motion analysis can be found in biomedicine. Gait analysis is used to assess pathological conditions of patients and to plan treatment such as orthopedic surgery and for follow-up monitoring. Similarly the methodology is applied in sports science, where the motion of athletes is optimized or monitored and automatically evaluated during endurance exercises.

More recently, motion analysis and motion capture have been applied in the entertainment industry. In its earliest form this involved a process called rotoscoping, i.e., an artist traced the outline of an actor in every frame of a reference video to create 2D animations of a subject. Nowadays, marker-based systems are frequently used to capture motions for use in both computer games and motion picture productions because creating animations by hand with the fidelity required in this industry is a very time-consuming task. Most commercial marker-based systems are still limited to controlled studio environments. Although recently, systems using active markers have been proposed that can be used outdoors or in onset conditions.

Open Problems

Although the main challenges of marker-based motion capture are generally considered solved, there is still room for improvement. Directors ask for ever more actors to be tracked simultaneously in real time; systems that work in outdoor settings are still only available using active markers or do not support real-time feedback. Retroreflective markers cannot be used outdoors because interference with other light sources prevents detecting markers reliably. Vision-oriented systems may be able to solve these issues and in most cases make less assumptions about the scene. Many approaches use no markers and significantly fewer cameras. In some cases, even moving backgrounds can be handled, or cameras are not assumed to be static. However, vision-based systems tend to be computationally expensive and less robust. Overall, combining the general capabilities of vision systems with the speed and robustness of marker-based approaches would significantly advance the state-of-the-art.

References

1. Sutherland DH (2002) The evolution of clinical gait analysis: part II kinematics. *Gait Posture* 16(2):159–179
2. Zhou H, Hu H (2008) Human motion tracking for rehabilitation – a survey. *Biomed Signal Process Control* 3:1–18
3. Moeslund TB, Hilton A, Krüger V (2006) A survey of advances in vision-based human motion capture and analysis. *Comput Vis Image Underst* 104(2):90–126
4. Menache A (1999) Understanding motion capture for computer animation and video games. Morgan Kaufmann, San Diego

Motion Capturing

► Motion Capture

Motion Deblurring

► Blind Deconvolution

Motion Tracking

► Motion Capture

Multi-baseline Stereo

David Gallup
Google Inc., Seattle, WA, USA

Definition

Multi-baseline stereo is any number of techniques for computing depth maps from several, typically many, photographs of a scene with known camera parameters.

Background

The goal of any stereo algorithm is to reconstruct the 3D surface geometry of a scene from multiple

photographs. Multi-baseline stereo can be seen as a generalization of binocular stereo, and it is one instance of a broader class of multi-view stereo algorithms. The classic binocular stereo problem focuses on using two views of a scene (the minimal case), whereas multi-baseline stereo uses more than two and typically many more views of the scene. More views not only provide a better signal to noise ratio but also eliminate most repetitive structure errors and offer new ways to handle occlusions.

Another type of multi-view stereo is volumetric stereo, which explicitly models the scene's surface in a volume, and is sometimes called object-based. Multi-baseline stereo on the other hand is image-based, and seeks to reconstruct the scene by assigning depth values to the pixels of one or more of the input images. Often this leads to better sampling of the input data and greater memory efficiency. The disadvantage is that special care must be taken at depth discontinuities.

Theory

Multi-baseline stereo shares the same theoretical concepts as other stereo problems. The inputs consist of a set of n input images with camera parameters. Camera parameters define a projection function which projects a 3D point into a pixel in the image. The goal is to compute a depth map for one or more of the input images. For simplicity we will focus on computing a depth map for a single view called the reference view.

Computing a depth map from images can be viewed as a maximum a posteriori estimation problem. Given images I_1, \dots, I_n , compute the depth map Z that maximizes $P(Z|I_1, \dots, I_n) = P(I_1, \dots, I_n|Z)P(Z)$. The likelihood $P(I_1, \dots, I_n|Z)$ describes how well the depth map fits the input data, and the prior $P(Z)$ describes desired properties of the depth map such as smoothness. This formulation can be expressed as an energy function:

$$\begin{aligned} E(Z) = & \sum_{p \in Z} E_{\text{data}}(Z(p)) \\ & + \sum_{(p,q) \in \mathcal{N}} E_{\text{smooth}}(Z(p), Z(q)). \end{aligned} \quad (1)$$

The data term E_{data} measures how well the depth value $Z(p)$ matches the input images. These *matching*

scores can be computed by comparing the intensity value in the reference view $I_{\text{ref}}(p)$ to the intensity values of the projections of $Z(p)$ in the matching views, $I_k(\text{proj}_k(Z(p)))$. Some views may be occluded, meaning that $Z(p)$ may not be visible from that view and $I_k(\text{proj}_k(Z(p)))$ will not match $I_{\text{ref}}(p)$. Occlusion handling aims to remove the influence of these occluded views and is a critical part of stereo. The smoothness term E_{smooth} penalizes variations between neighboring depth values (given by the set \mathcal{N}).

Matching Scores

Consider a known 3D point X on the surface of the scene as shown in Fig. 1. Let x_0, \dots, x_k be the projections of X into each image. The image intensities at these points should be *photo-consistent* (have the same appearance) since they are all images of the same point on the surface. This will not necessarily be true for 3D points off the surface of the scene. Brightness constancy is the assumption that the intensity of light does not change from viewpoint to viewpoint, a property of Lambertian surfaces. This can be measured by taking the absolute difference or squared difference between the intensity value in the reference view and the intensity value in matching view. Typically a single pixel does not carry enough information to make an informative matching score, and so often the differences between patches of nearby pixels are also incorporated into the matching score. This yields the sum of absolute differences (SAD) and sum of squared differences (SSD) scores.

The brightness constancy assumption can be violated for many reasons, for example, different exposure settings between images and specular surfaces that reflect light at different intensities depending on the angle. To account for some of these changes, patches can be normalized by removing the mean intensity of the patch and scaling the values so that variance is 1. This yields the normalized cross-correlation score (NCC). Let M and N be two rectangular image patches of size $w \times h$. These patches can be flattened to form vectors \mathbf{m} and \mathbf{n} of size $w \cdot h$. The NCC score is then

$$NCC(M, N) = \frac{(\mathbf{m} - \bar{\mathbf{m}}) \cdot (\mathbf{n} - \bar{\mathbf{n}})}{\text{var}(\mathbf{m})\text{var}(\mathbf{n})}, \quad (2)$$

where $\bar{\mathbf{m}}$ is the mean of \mathbf{m} and $\text{var}(\mathbf{m})$ is its variance.

When patches originate from highly slanted surfaces, they may need to be corrected before being correlated. The surface's tangent plane can be defined given the point's surface normal, and images can be aligned by projecting them onto this plane. This transformation can be expressed as a homography. This adds two additional angle parameters per pixel to the problem. One method to account for the surface normal is to compute the matching score as the best score over all surface normals [1]. A much faster alternative is to consider only a small number of likely candidates [2]. Another is to iteratively estimate a depth map, using surface normals given by the depth map from the previous iteration.

Summing the SAD, SSD, or NCC scores from multiple views reduces the influence of noise as well as disambiguates mismatches due to repetitive structures since it is less likely that a mismatch will occur in all views simultaneously [3].

Occlusion Handling

Summing scores from multiple views treats all images equally. In fact some images may be occluded, meaning a different surface is seen from that viewpoint, and the matching score is arbitrarily bad. Handling these cases is important for stereo, and multi-baseline stereo has advantages in this regard. One method based on robust statistics is to assume that at least k views are unoccluded and to discard the rest. The sum of the best k views can be obtained by sorting and summing. Another method is to assume an object will be occluded from one side and not another. Assuming cameras are arranged in a line, the unoccluded half-set will be either the left or right half-set of cameras [4]. See Fig. 2.

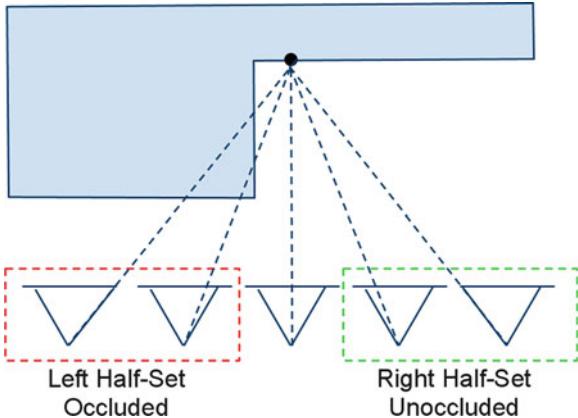
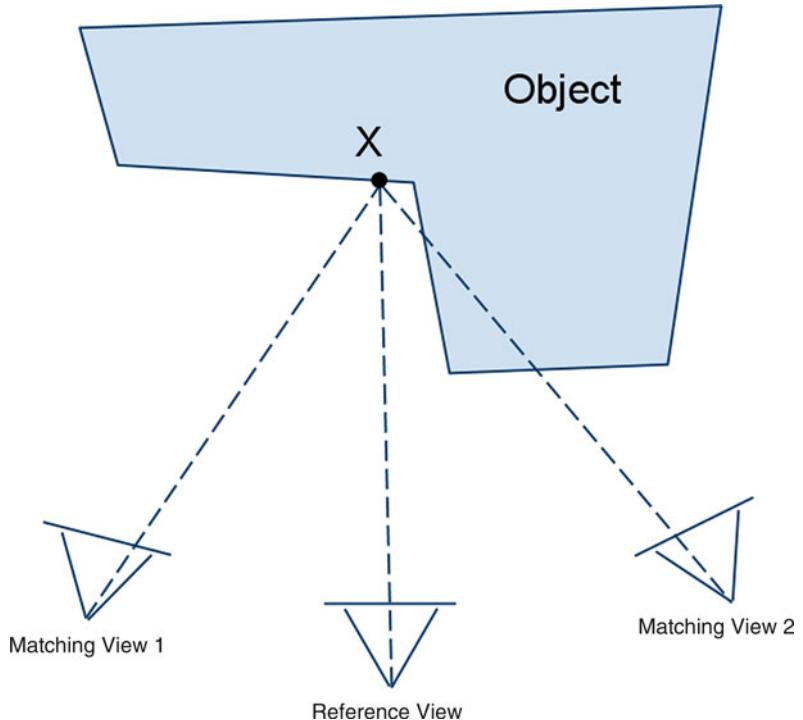
Other methods seek to detect occlusions explicitly by using the reconstruction itself to identify occluded views. This is a chicken and egg problem: the occlusions must be known to reconstruct the scene, and the reconstruction of the scene must be known to detect the occlusions. Some methods model occlusions probabilistically [5], and others start with a safe configuration and update the reconstruction conservatively [6].

Optimization

Optimizing Eq. 1 is difficult due to the non-convex data and smoothness terms. Some methods ignore the

Multi-baseline

Stereo, Fig. 1 A surface point X is projected into the images. Matching view 2 is occluded and should not contribute to the matching score



Multi-baseline Stereo, Fig. 2 Simple occlusion geometry. Typically, either the left half-set or the right half-set will be free of occlusion

smoothness term, and each pixel is optimized independently by exhaustive search. The Z function is discretized into points along viewing rays that project to individual pixels in the matching views. Testing all points for each pixel can be done efficiently using graphics hardware which is well suited for this type of *embarrassingly parallel* computation [12]. Methods

that do use the smoothness term obtain better results, and if the benefits of normalization (NCC) are not needed, patches need not be used and single pixels can be used. The ambiguity of single pixel matching is resolved because the smoothness term regularizes the solution. The optimization problem is *NP-hard*, but there are effective approximation algorithms based on graph cuts [8] and belief propagation [9].

Depth Resolution

The resolution of stereo as a depth sensor depends on the distance of the surface from the cameras. The depth resolution is the distance between pixels in a matching view projected onto a viewing ray in the reference view. In the binocular case, the resolution is

$$\Delta z = \frac{z^2}{bf}, \quad (3)$$

where Δz is the resolution, z is the distance to the reference view, b is the baseline or distance between camera centers, and f is the focal length of the cameras. Depth measurement uncertainty depends on the matching uncertainty which describes how precisely the two observed patterns can be registered and depends on

factors like texture and image noise. Depth measurement uncertainty is proportional to depth resolution. With enough views, multi-baseline stereo has the advantage that the baseline can be treated as a variable rather than a constant. This gives greater control over the depth resolution and computation time of the stereo algorithm [10].

Application

Multi-baseline stereo is applied to many 3D reconstruction problems such as 3D city modeling [11], view synthesis [12], and digital archiving. Multi-baseline stereo is often used in real-time applications where high-quality depth can be computed from a large number of views without the need for computationally intensive optimization techniques. This is especially true for video applications, where the camera moves in a fairly linear manner, and so more general multi-view stereo techniques are unnecessary.

References

1. Zabulis X, Daniilidis K (2004) Multi-camera reconstruction based on surface normal estimation and best viewpoint selection. In: Proceedings of the 2nd international symposium on 3D data processing, visualization and transmission (3DPVT), pp 733–740
2. Gallup D, Frahm J-M, Mordohai P, Yang Q, Pollefeys M (2004) Real-time plane-sweeping stereo with multiple sweeping directions. In: IEEE conference computer vision and pattern recognition (CVPR), Minneapolis
3. Okutomi M, Kanade T (2002) A multiple-baseline stereo. In: IEEE Trans Pattern Anal Mach Intell, 15(4): 353–363
4. Kang S, Szeliski R, Chai J (2001) Handling occlusions in dense multi-view stereo. In: IEEE conference computer vision and pattern recognition (CVPR), Kauai
5. Hernandez C, Vogiatzis G, Cipolla R (2001) Probabilistic visibility for multi-view stereo. In: IEEE conference computer vision and pattern recognition (CVPR), Minneapolis
6. Kolmogorov V, Zabih R (2002) Multi-camera scene reconstruction via graph cuts. In: European conference on computer vision 2002 (ECCV), Copenhagen
7. Yang R, Pollefeys M (2003) Multi-resolution real-time stereo on commodity graphics hardware. In: IEEE conference on computer vision and pattern recognition (CVPR), Madison
8. Boykov Y, Veksler O, Zabih R (2001) Fast approximate energy minimization via graph cuts. In: IEEE Trans Pattern Anal Mach Intell, 23(11): 1222–1239
9. Felzenszwalb P, Huttenlocher D (2001) Efficient belief propagation for early vision. In: IEEE conference on computer vision and pattern recognition (CVPR), Kauai
10. Gallup D, Frahm J-M, Mordohai P, Pollefeys M (2008) Variable baseline/resolution stereo. In: IEEE conference on computer vision and pattern recognition (CVPR), Anchorage
11. Pollefeys M, Nister D, Frahm J-M, Akbarzadeh A, Mordohai P, Clipp B, Engels C, Gallup D, Kim S-J, Merrell P, Salmi C, Sinha S, Talton B, Wang L, Yang Q, Stewenius H, Yang R, Welch G, Towles H (2008) Detailed real-time urban 3D reconstruction from video. Int J Comput Vision, 78(2–3): 143–167
12. Yang R, Pollefeys M, Yang H, Welch G (2003) A unified approach to real-time, multi-resolution, multi-baseline 2D view synthesis and 3D depth estimation using commodity graphics hardware. Int J Image Graph, 4(4): 627–651

Multi-camera Calibration

► [Calibration of Multi-camera Setups](#)

Multi-camera Human Action Recognition

Gaurav Srivastava¹, Johnny Park¹, Avinash C. Kak¹, Birgi Tumeroy² and J. K. Aggarwal³

¹School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA

²Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX, USA

³Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX, USA

Synonyms

[Activity analysis](#); [Behavior understanding](#)

Related Concepts

► [Gesture Recognition](#); ► [Human Pose Estimation](#)

Definition

Multi-camera human action recognition deals with using multiple cameras to capture several views of humans engaged in various activities and then combining the information gleaned from the cameras for the classification of those activities.

Background

Research on human activity recognition gathered momentum in the mid- to late 1990s; much early work is summarized in a review by Aggarwal and Cai [1]. There emerged two dominant approaches during this period: (1) state-space modeling of human actions [2, 3]; and (2) template matching [4, 5]. The focus during that early phase of this research was primarily on recognizing human activities on the basis of the images collected by a single camera. While this is still an active research area in computer vision (see Aggarwal and Ryoo [6] for a survey), it unfortunately suffers from several serious shortcomings, many of them owing to the limitations inherent to images that are recorded from just one viewpoint. Human activities, in general, are much too complex in 3D to be described by cues extracted from single-viewpoint 2D projections. While it is true that the human eye (even just a single eye) can do a wonderful job of categorizing human activities, trying to replicate that in a computer would be far too ambitious a research project for a long time to come. It is not yet fully understood how the human brain fills in the information that it cannot see directly in order to recognize objects and movements despite severe occlusion and noise. While it is a proper exercise in humility to be awed by the capabilities of the human brain, it is nonetheless good to keep in mind that even a human can be fooled in its perception of an activity when the perception is limited to a single viewpoint. Magicians frequently take advantage of such limitations of human perception in order to produce their magical effects.

In addition to the problems caused by the fact that a single camera provides only single-viewpoint 2D projections of the scene, other reasons for the more recent interest in multi-camera approaches to activity recognition stem from the current global interest in wide-area surveillance, on the one hand, and in the design of intelligent environments for the living spaces of the future on the other. In both of these applications, the goal is to characterize a human activity as it is evolving with time and as it is occupying space that may not be limited to the coverage provided by a single camera. Consider a habitat of the future for the aged and the infirm where you may wish to use a network of cameras that silently watch for any undesirable human behavior, such as someone suddenly collapsing on the floor or tripping over a piece of furniture.

To recognize such human activities, the camera system would need to analyze the sensed data over a period of time and, even more particularly, over some a span of physical space. Multi-camera imagery would obviously lend itself much better to the sort of data analysis that would need to be carried out for the required inferences.

With a view to gaining insight into the various aspects of multi-camera human action recognition, in the remainder of this chapter, section “Theory” describes the theoretical details of the different types of approaches that have been proposed by researchers to accomplish multi-camera action recognition. Owing to this chapter’s intent of providing accessibility to the general readers, significant details about the algorithms are not discussed; rather, only the distinguishing highlights of the different approaches are presented. Section “Application” presents a discussion on some of the application areas that will benefit from multi-camera action recognition as compared to the single camera modality. Since human action recognition is a complex and challenging problem, there are quite a few open problems that need to be addressed before this research becomes useful for mainstream society. Section “Open Problems” enumerates some of these open problems related to human action recognition in general and also those that are specific to the area of multi-camera action recognition. Finally, in section “Experimental Results,” this chapter is concluded with a performance comparison between the different multi-camera approaches on a common benchmark dataset.

Theory

The first step in human action recognition involves creating a library of models for different human actions to be recognized. An action model characterizes the unique motion patterns associated with an action. These models can be created from a temporal sequence of either 2D images or 3D reconstructions obtained by combining multiple camera images of a human actor. Researchers have proposed different approaches to creating the action models. For example, an action model may comprise of a set of local motion features extracted from the spatio-temporal volume of the action image sequence. Another example of an action model is a set of exemplar human poses represented

either as 2D silhouettes or 3D reconstructions. Yet another example is a set of spatiotemporal trajectories of different human body parts. Once the action models are created, new instances of human actions can be recognized. Given a test image sequence that contains an unknown human action, the same technique used for creating the action models is applied to the test sequence to generate its action representation. This action representation is then compared against each action model in the library. Finally, the test sequence is assigned the label of the most similar action model. In terms of the underlying operating principle, there is no difference between human action recognition and any other type of object recognition: A test object whose category label is to be ascertained is assigned the label of the closest matching model whose category is known.

A common assumption in single camera action recognition is that the test action and the model actions have been captured from identical or very similar camera viewpoints. The quality of match between the test action and the model actions is a function of the conformity between their camera views. As Souvenir and Babbs [7] point out, it would be impractical for any human motion analysis system to impose the constraint that humans engaged in an activity are facing the same direction relative to the camera view at all times. In order to alleviate this problem, several methods have been developed which use multiple cameras in the training and/or testing phases of action recognition. These methods provide viewpoint invariance, i.e., the ability to perform matching between a pair of action observations even if they have been acquired from different camera views. A good overview of multiple camera action recognition approaches can be found in [8]. Based on the distinct fundamental ideas of these methods, they have been categorized into the following three classes:

1. Multi-view geometry-based methods
2. View-invariant representation-based methods
3. Exhaustive search methods

Multi-view Geometry-Based Methods

The multi-view geometry-based approaches utilize the epipolar geometric constraints between multiple cameras for human action recognition. The intuition is that if two actors perform the same action, then assuming the same temporal rate of action execution, their

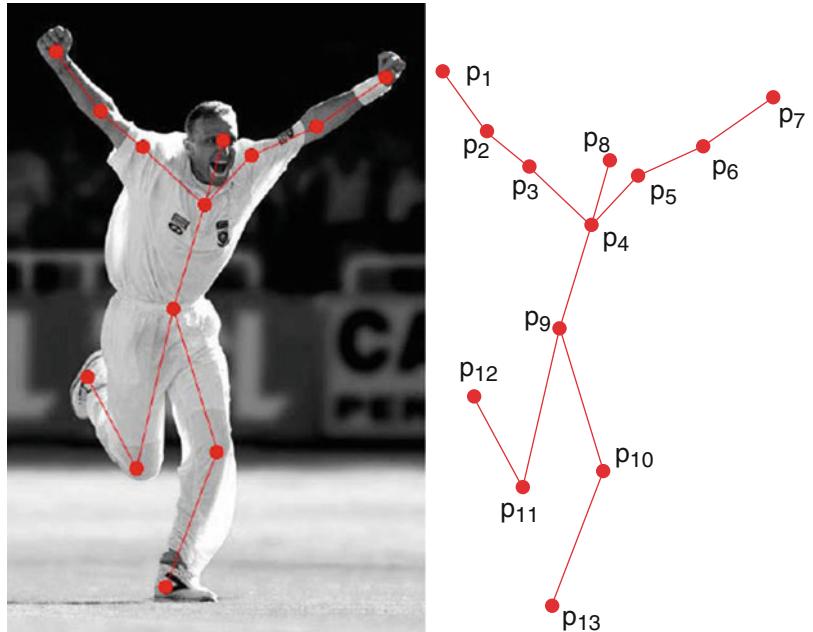
postures at all corresponding time instants are related by epipolar geometric relationships. Such relationships between the two views are applied to point correspondences where the points are generally chosen as the anatomical landmarks on the human body, e.g., head, shoulders, hands, and feet (see Fig. 1). Action recognition is performed by measuring the similarity between the postures of a test action sequence and a model action sequence at every time instant. The similarity measure can be expressed in terms of the point correspondences and a matrix F known as the fundamental matrix that is computed using epipolar geometry (Fig. 2). Given at least eight pairs of point correspondences (x_i, x'_i) , the fundamental matrix F satisfies the relation $x_i^T F x'_i = 0$, $i = 1, \dots, n \geq 8$. In practical settings, the point correspondences between the action representations in two different views will generally not be precise, and hence, the quantity $x_i^T F x'_i$ will not be exactly zero. Nevertheless, the residual $\sum_i |x_i^T F x'_i|^2$ can be used as the matching cost or the similarity measure. If the matching cost is below a certain threshold, then the point correspondences come from the same action represented in the different views. Generally, the test action postures and the action models are derived from different persons with different body sizes and proportions; therefore, certain anthropometric constraints may also need to be imposed to normalize the landmark points to a common coordinate frame before the epipolar geometric constraints can be applied. Such and other similar constraints on the point correspondences have been used for matching of different action instances in [9, 11–13].

View-Invariant Representation-Based Methods

In addition to the multi-view geometry-based approaches, there are other interesting methods that accomplish viewpoint invariant action recognition based on machine learning techniques. It is a conceivable scenario that a discriminative model of an action is available for one camera viewpoint (source view), and the action recognition needs to be performed in another view (target view) for which such a model is not available. This issue has been addressed in [14], where a transfer learning approach is used along with examples of corresponding observations of actions from both views in order to learn how the appearance of an action changes with the change of viewpoint.

Multi-camera Human Action Recognition, Fig. 1

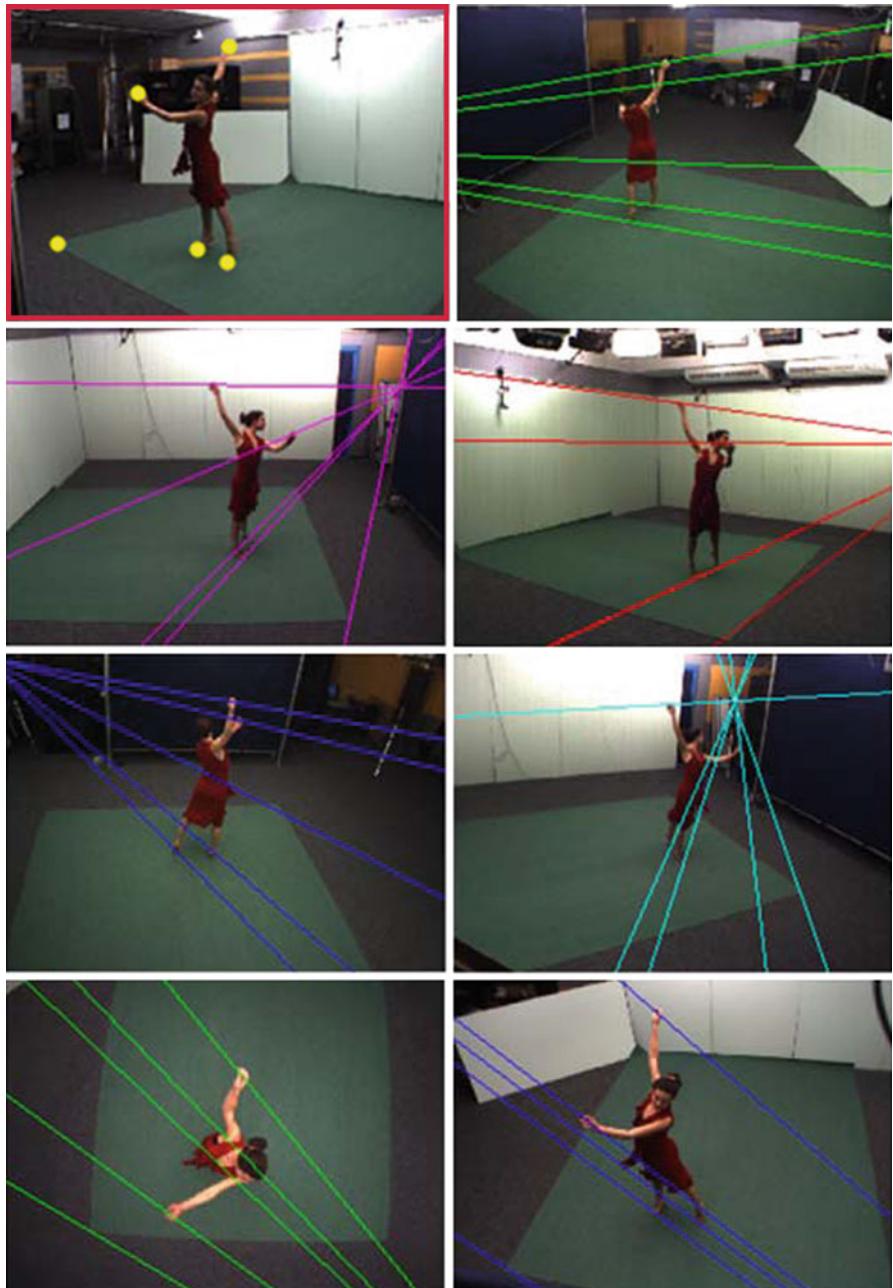
The posture of a human actor at a specific time instant. It is represented as a set of anatomical landmarks (Gritai et al. [9], ©2004 IEEE)



It is worth noting that such learning can be performed with one set of actions and then applied to a new unknown action for which the transfer model is not explicitly built. Transfer learning is applied by splitting the source domain feature space using the action descriptors and the action labels to produce certain split-based features. These split-based features are considered transferable between the source view and the target view and therefore are used to construct an action-discriminative split of the target domain feature space. For a test action sequence in the target view, its action descriptors are extracted, and action recognition is performed by a nearest neighbor matching with the action descriptors of the model actions which were transferred from the source view.

In [7], the variation in the appearance of an action with viewpoint changes is estimated by learning low-dimensional representations of the actions using manifold learning. The action descriptor used is the \mathcal{R} transform surface which is a temporal extension of the well-known Radon transform (Fig. 3). In the figure, the horizontal axes correspond to time t and the polar angle θ used in Radon transform computations. The \mathcal{R} transform surface is a high-dimensional data that lies on a non-linear manifold, and hence it can be embedded into a lower dimensional space. In this low-dimensional space, learning how the data varies as

a function of the viewpoint provides a representation which allows to avoid storing action examples from all possible viewpoints. Action recognition is performed by obtaining a similarity measure between two \mathcal{R} transform surfaces S_1 and S_2 , such as the L^2 distance $\|S_1 - S_2\|$. Another interesting approach has been described in [15], where the temporal self-similarity between the frames of an action sequence was shown to be highly stable with a changing viewpoint (Fig. 4). Specifically, for the same action sequence recorded from very different views, the so-called temporal self-similarity matrices (SSMs) corresponding to the different views were shown to be very similar. This observation was consistent even when different image features were used for computing the self-similarity matrix. By constructing histogram-based descriptors from the elements of the SSM (as described in [15], Sect. IV), action recognition can be performed by applying classifiers like nearest-neighbor or support vector machine on these descriptors. Recently, Kusakunniran et al. [16] proposed a view transformation model based on support vector regression for solving the multi-view gait recognition problem. This view transformation model uses local regions of interest (ROIs) in one view to predict the motion information of the corresponding regions in a different view. In order to perform gait recognition, the gait features from the different actors'

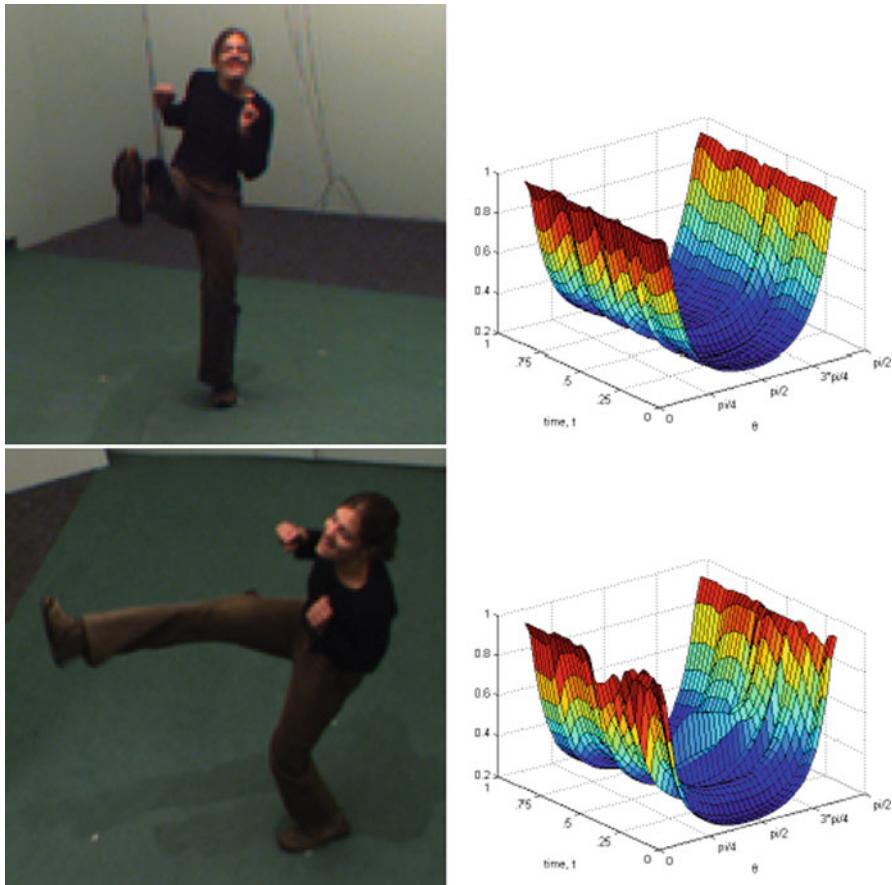


Multi-camera Human Action Recognition, Fig. 2 Landmark points (yellow) in one view. The other views show the epipolar lines which contain the points corresponding to the landmark points (Sinha and Pollefeys [10], ©2009 Springer)

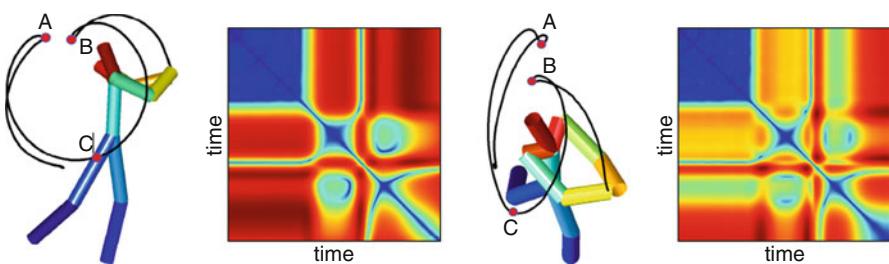
gait sequences and possibly different view angles are first normalized to a common viewing angle using the view transformation model followed by a similarity measure calculation between the normalized gait features using the Euclidean metric.

Exhaustive Search Methods

The third popular approach in the multi-camera human action recognition is to perform an exhaustive search in the space of multi-view action poses to find the best match for the test action poses. Such an exhaustive



Multi-camera Human Action Recognition, Fig. 3 Kicking action from two different view points and their corresponding \mathcal{R} transform surface action descriptors. They are quite similar despite view point variation (Souvenir and Babbs [7], ©2008 IEEE)



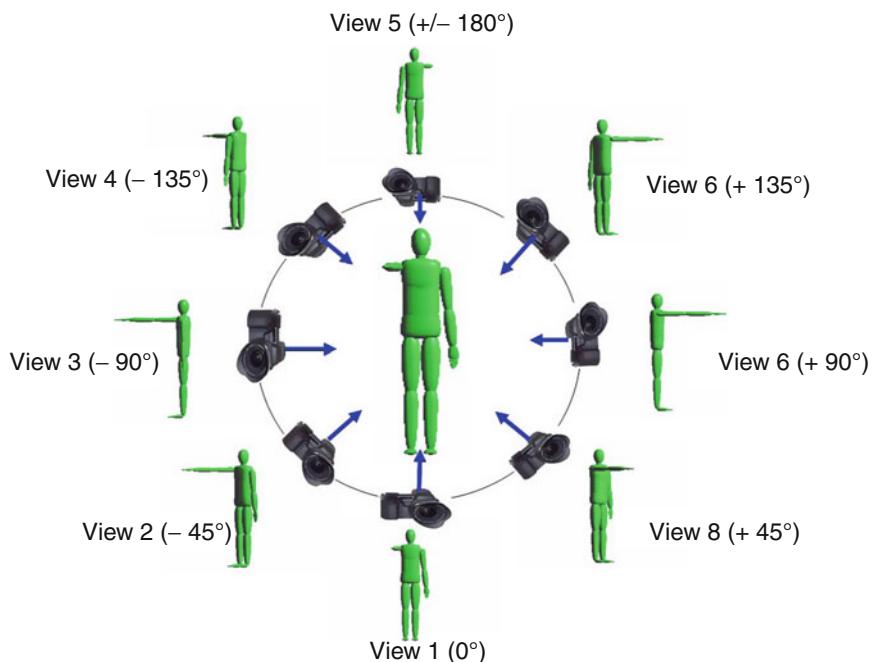
Multi-camera Human Action Recognition, Fig. 4 A golf swing action seen from two different views and their corresponding temporal self-similarity matrices (Junejo et al. [15], ©2011 IEEE)

search can be performed in 2D or 3D. During the training time, a set of multiple fixed cameras installed around the actor is used to record the multi-view sequences of his or her actions (Fig. 5). In the 2D exhaustive search approach, an observation is

recorded for the unknown action and matched against each recorded view from the training session. The matching can be performed using the shape features derived from the 2D silhouettes, the motion features obtained from the optical flow, or histograms of local

Multi-camera Human Action Recognition, Fig. 5

Acquiring simultaneous multiview action sequences for training (Ahmad and Lee [17], ©2006 IEEE)



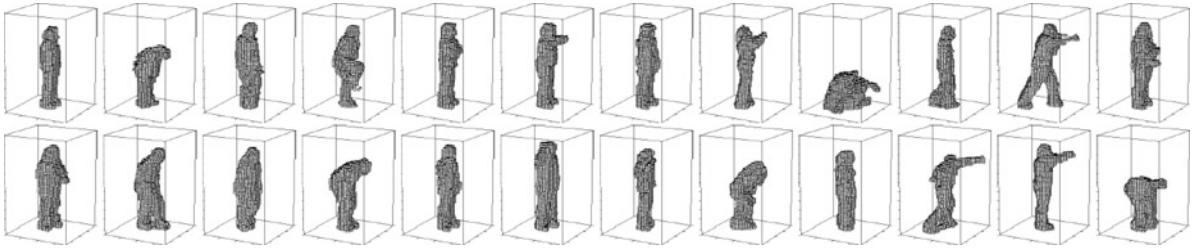
spatiotemporal cuboid features. In order to accomplish action recognition, this 2D search is performed at every time instant of the action sequence, and the model action resulting in the smallest feature distance over all the time instants is used to label the unknown action. A limitation of the 2D approach is that the same spatial configuration of cameras has to be used during the training and testing sessions. Generally, during the test time, the observations are recorded from a single view, but more than one view can also be used with the appropriate changes in the matching algorithm. The 2D approach has been used by several research groups [17–19].

The 3D exhaustive search approach provides more flexibility with regard to the placement of the cameras in the monitoring environment. Different spatial configurations of cameras can be used during the training and testing sessions. Here, instead of storing all the discrete views of the action during the training time, they are combined to produce an action model based on 3D reconstruction (Fig. 6). The key advantage of such a strategy is that there is no restriction on what camera view(s) can be used to record the test action sequence. If the camera parameters of the arbitrary camera view used during the testing session are known, then the model 3D action representations can be projected into that 2D view for matching with the observation. Some examples of such an approach are [20, 21].

Application

Vision-based human activity recognition has diverse applications. Generally, any practical application related to monitoring human activities requires that the monitoring can be done over an extended physical space beyond the viewing area of a single camera. It may also be required to monitor an event from several viewpoints simultaneously so as to obtain richer descriptions of complex human activities. Such requirements necessitate the use of multiple cameras for capturing the events. In this section, some of the application areas are briefly discussed where multi-camera human activity recognition or multi-camera event detection is currently being used or has strong potential for use in the near future.

1. *Wide area surveillance* – Facilities like government buildings, military installations, airports, subways, power plants, dams, and so on require round-the-clock surveillance. Some examples of the general human-related activities that need to be monitored are wide-area perimeter breaches by intruders, a person approaching the doors after hours, leaving of a suspicious unattended object by a person, extended loitering around the facility perimeter, and persons tampering with the facility security systems. In such scenarios, it is necessary to study not only the isolated activities of the persons but also



Multi-camera Human Action Recognition, Fig. 6 An action model consisting of 3D exemplars that were selected based on 11 types of actions by 10 human actors [20]. Given a test sequence

of 2D action silhouettes, action recognition involves finding the best matching exemplar sequence and the best matching 2D projections (©2007 IEEE)

- the patterns of their interactions with their surroundings, such as who are other persons they interact with, are they carrying any objects, how long have they been present or are they loitering around the security systems.
2. *Monitoring the elderly and children* – Assisted living homes and day care centers require constant monitoring of the activities of the elderly and children to avoid incidents such as an elderly person tripping over a furniture and falling down or a child playing too close to hazardous areas like a swimming pool or an electrical appliance. Commercial applications are available for day care centers that allow the installation of multiple wireless surveillance cameras in play areas, resting areas, or dining areas, where the video and audio feed from the cameras can be transmitted to a central computer connected to the Internet. Parents can thus check on the safety of their children at any time by viewing over the Internet.

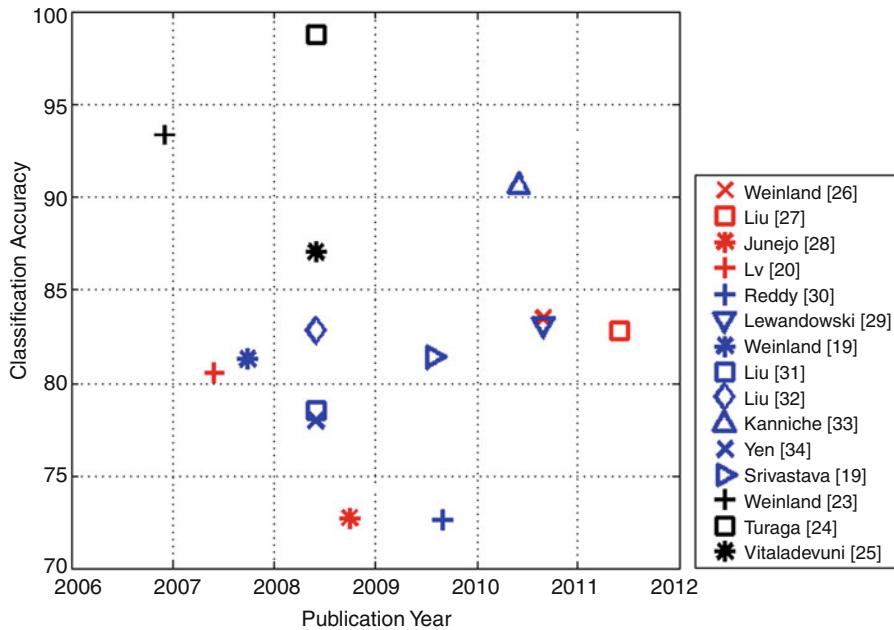
3. *Three-dimensional human motion analysis for sports and medicine* – A 3D motion-based system can be used to perform a marker-based or markerless capture of the human motion while performing ordinary activities like walking or bending or sports activities like a golf swing, swimming strokes, tennis serves, gymnastics, and so on. In the field of medicine, such human motion data is used to characterize the dynamics of activities, discover the fundamental principles that govern movement, and understand the causes of movement disorders. In the area of sports, the motion data of the athlete can be analyzed by the coach for recommending changes in the dynamics to achieve more energy-efficient and agile performance, or it may be compared with the 3D reference motion of another expert athlete.

4. *Enhanced sports viewing experience* – Multiple cameras are routinely used in sports like football, cricket, and soccer to capture many views of a dynamic event for providing an enriched fly through experience of the scene to the viewers or for use by the referees when it is difficult to make judgments based on a single camera view.

Open Problems

In the last 15 years, substantial progress has been made in the field of human action recognition. The accomplishments have primarily been in the area of single-person action recognition in an uncluttered background and recognizing a set of simple activities like walking, jumping, waving hands, punching, and so on. This is evidenced by the popularity of datasets like KTH, Weizmann, and IXMAS for benchmarking the action recognition algorithms (please see [8], Sect. 6 for description of these datasets). These datasets capture simple actions performed by a single human, with a clean background, and negligible variations in spatial scale of the person or temporal speed of execution. The challenge for the research community, in most simple words, is to extend the current recognition algorithms to work on datasets comprising of video sequences captured in unconstrained settings such as the Hollywood movie dataset [22] or the YouTube video dataset [23].

Specifically in the area of multi-camera action recognition, an open problem is to obtain nonrigid point correspondences on the human body that are needed for applying the geometric constraints. The main underlying difficulty is the reliable detection and tracking of human body parts in an unconstrained visual setting. Similarly, a limitation on the view



Multi-camera Human Action Recognition, Fig. 7 Action recognition performance of different algorithms on IXMAS dataset

invariant representations like self-similarity matrices [15] and \mathcal{R} transform surface [7] is that they are constructed from the temporal variation of the features; hence, they need the full video sequences to be available offline. Real-time applications like video surveillance and human-computer interaction will benefit from the development of view-invariant features that can be computed online.

Experimental Results

This chapter is concluded with a chronological summarization of the different algorithms' action recognition performances on a benchmark dataset named the INRIA Xmas Motion Acquisition Sequences (IXMAS) [24].

The IXMAS dataset is the most commonly used dataset for evaluating the multi-camera action recognition algorithms. It contains 13 daily-life actions, such as *check watch*, *cross arms*, and *scratch head*. Each action is performed three times by 11 actors. The actions are captured using five calibrated and synchronized cameras. The actors are free to perform the actions in any orientation, making this a fairly challenging dataset.

Figure 7 presents the reported average recognition accuracies of several recent works on multi-camera action recognition. It is important to note that not all the approaches use the same evaluation methodologies, and hence, it is difficult to compare them merely based on these values.

For easier comparison, these works are categorized into three groups based on their evaluation methodologies: (1) the methods, which use 3D representations in the recognition stage [24–26], (2) the methods, which report camera-specific recognition accuracies [21, 27–29], and (3) the methods, which incorporate results from multiple cameras (e.g., using simple voting) [19, 20, 30–35]. These groups are distinguished by black, red, and blue markers in **Fig. 7**, respectively.

References

1. Aggarwal JK, Cai Q (1999) Human motion analysis: a review. *Comput Vis Image Underst* 73(3):428–440
2. Bobick AF, Wilson AD (1995) A state-based technique for the summarization and recognition of gesture. In: Proceedings of the fifth international conference on computer vision, ICCV '95, Washington, DC. IEEE Computer Society, pp 382–389
3. Brand M, Oliver N, Pentland A (1997) Coupled hidden markov models for complex action recognition. In: IEEE

- computer society conference on computer vision and pattern recognition (CVPR), Washington, 1997, p 994
- 4. Polana R, Nelson R (1994) Low level recognition of human motion (or how to get your man without finding his body parts). In: Proceedings of the IEEE workshop on motion of non-rigid and articulated objects, Austin, TX, USA, 1994, pp 77–82
 - 5. Bobick A, Davis J (1996) Real-time recognition of activity using temporal templates. In: WACV '96., proceedings 3rd IEEE workshop on applications of computer vision, Sarasota, FL, USA, 1996, pp 39–42
 - 6. Aggarwal JK, Ryoo MS (2011) Human activity analysis: a review. *ACM Comput Surv* 43(3):1–43
 - 7. Souvenir R, Babbs J (2008) Learning the viewpoint manifold for action recognition. In: IEEE conference on computer vision and pattern recognition (CVPR), Anchorage, pp 1–7
 - 8. Weinland D, Ronfard R, Boyer E (2010) A survey of vision-based methods for action representation, segmentation and recognition. *Comput Vis Image Underst* 115(2): 224–241
 - 9. Gritai A, Sheikh Y, Shah M (2004) On the use of anthropometry in the invariant analysis of human actions. In: Proceedings of the 17th international conference on pattern recognition, Cambridge, 2004. ICPR 2004, vol 2, pp 923–926
 - 10. Sinha SN, Pollefeys M (2009) Camera network calibration and synchronization from silhouettes in archived video. *Int J Comput Vis* 87(3):266–283
 - 11. Syeda-Mahmood T, Vasilescu A, Sethi S (2002) Recognizing action events from multiple viewpoints. In: Proceedings of IEEE workshop on detection and recognition of events in Video, Vancouver, BC, Canada, 2001, pp 64–72
 - 12. Rao C, Yilmaz A, Shah M (2002) View-invariant representation and recognition of actions. *Int J Comput Vis* 50(2):203–226
 - 13. Parameswaran V, Chellappa R (2003) View invariants for human action recognition. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), Madison, 2003, vol 2, pp 613–19
 - 14. Farhadi A, Tabrizi MK (2008) Learning to recognize activities from the wrong view point. In: Forsyth D, Torr P, Zisserman A (eds) Computer vision—ECCV, 2008. Springer, Berlin/Heidelberg, pp 154–166
 - 15. Junejo IN, Dexter E, Laptev I, Pérez P (2011) View-independent action recognition from temporal self-similarities. *IEEE Trans Pattern Anal Mach Intell* 33(1):172–85
 - 16. Kusakunniran W, Wu Q, Zhang J, Li H (2010) Support vector regression for multi-view gait recognition based on local motion feature selection. In: IEEE conference on computer vision and pattern recognition (CVPR), San Francisco, pp 974–981
 - 17. Ahmad M, Lee SW (2006) HMM-based human action recognition using multiview image sequences. In: 18th international conference on pattern recognition, Hong Kong, 2006. ICPR 2006, vol 1, pp 263–266
 - 18. Ogale A, Karapurkar A (2007) View-invariant modeling and recognition of human actions using grammars. In: Proceedings of workshop on dynamic vision, Beijing, China, pp 115–126
 - 19. Srivastava G, Iwaki H, Park J, Kak AC (2009) Distributed and lightweight multi-camera human activity classification. In: 2009 third ACM/IEEE international conference on distributed smart cameras (ICDSC), Stanford, CA, USA, pp 1–8
 - 20. Weinland D, Boyer E, Ronfard R (2007) Action recognition from arbitrary views using 3D exemplars. In: Computer vision, Rio de Janeiro, 2007. ICCV 2007, IEEE 11th International Conference, pp 1–7
 - 21. Lv F, Nevatia R (2007) Single view human action recognition using key pose matching and viterbi path searching. In: IEEE conference on computer vision and pattern recognition, Minneapolis, 2007 (CVPR'07), pp 1–8
 - 22. Laptev I, Marszalek M, Schmid C, Rozenfeld B (2008) Learning realistic human actions from movies. In: IEEE computer society conference on computer vision and pattern recognition (CVPR), Anchorage, pp 1–8
 - 23. Liu J, Luo J, Shah M (2009) Recognizing realistic actions from videos in the wild. In: IEEE conference on computer vision and pattern recognition (CVPR), San Francisco, 1996–2003
 - 24. Weinland D, Ronfard R, Boyer E (2006) Free viewpoint action recognition using motion history volumes. *Comput Vis Image Underst* 104(2–3):249–257
 - 25. Turaga P, Veeraghavan A, Chellappa R (2008) Statistical analysis on stiefel and grassmann manifolds with applications in computer vision. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Anchorage
 - 26. Vitaladevuni SN, Kelloggumpu V, Davis LS (2008) Action recognition using ballistic dynamics. In: IEEE conference on computer vision and pattern recognition (CVPR), Anchorage
 - 27. Weinland D, Ozuysal M, Fua P (2010) Making action recognition robust to occlusions and viewpoint changes. In: Proceedings of the 11th European conference on computer vision (ECCV), Heraklion. Lecture notes in computer science
 - 28. Liu J, Shah M, Kuipers B, Savarese S (2011) Cross-view action recognition via view knowledge transfer. In: IEEE conference on computer vision and pattern recognition (CVPR), Colorado Springs
 - 29. Junejo I, Dexter E, Laptev I, Perez P (2008) Cross-view action recognition from temporal self-similarities. In: Proceedings of the 10th european conference on computer vision (ECCV), Marseille. ECCV'08
 - 30. Lewandowski M, Makris D, Nebel JC (2010) View and style-independent action manifolds for human activity recognition. In: Proceedings of the 11th European conference on computer vision: part VI (ECCV'10). Springer, Berlin/Heidelberg, pp 547–560
 - 31. Reddy K, Liu J, Shah M (2009) Incremental action recognition using feature-tree. In: Computer vision, 2009 IEEE 12th international conference, Kyoto, pp 1010–1017
 - 32. Liu J, Ali S, Shah M (2008) Recognizing human actions using multiple features. In: IEEE conference on computer vision and pattern recognition (CVPR), Anchorage
 - 33. Liu J, Shah M (2008) Learning human actions via information maximization. In: IEEE conference on computer vision and pattern recognition (CVPR), Anchorage

34. Kaaniche MB, Bremond F (2010) Gesture recognition by learning local motion signatures. In: IEEE conference on computer vision and pattern recognition (CVPR), San Francisco, pp 2745–2752
35. Yan P, Khan S, Shah M (2008) Learning 4d action feature models for arbitrary view action recognition. In: IEEE conference on computer vision and pattern recognition (CVPR), Anchorage

Multi-focus Images

Amit Agrawal
Mitsubishi Electric Research Laboratories,
Cambridge, MA, USA

Synonyms

Focus bracketing

Definition

Multi-focus images are a set of images of the same scene focused at different depths in the scene.

Background

Conventional imaging systems have a finite depth of field (DOF), which depends on the aperture size and the focal length of the lens. The DOF is the depth range within which the scene points appear sharp in the captured image. For scene points within the DOF, the size of the defocus blur is smaller than the minimum acceptable circle of confusion. Often, a single photo is unable to capture the entire scene in sharp focus.

DOF can be increased by decreasing the aperture size (increasing the F-number). However, reducing the aperture size decreases the light throughput, resulting in a dark and noisy image. Multi-focus images offer a solution to increase the DOF without decreasing light throughput. By capturing different photos focused at different depths in the scene and combining them, the entire scene can be brought into focus. This capture procedure is also called “focus bracketing.” This is similar to “exposure bracketing,” where

images are taken under different exposures and combined to obtain a high dynamic range image. However, a disadvantage with focus bracketing is that the entire scene has to remain static during the capture process.

Theory

For a thin lens with focal length f and lens to sensor plane distance u , the plane of focus is at a distance v from the lens, where

$$\frac{1}{v} = \frac{1}{f} - \frac{1}{u}. \quad (1)$$

Let c be the size of the acceptable circle of confusion and A be the aperture diameter. Then, f-number $N = f/A$. The DOF is then spanned between D_n and D_f , where

$$D_n = \frac{vf^2}{f^2 + Nc(v-f)}, \quad (2)$$

$$D_f = \frac{vf^2}{f^2 - Nc(v-f)}. \quad (3)$$

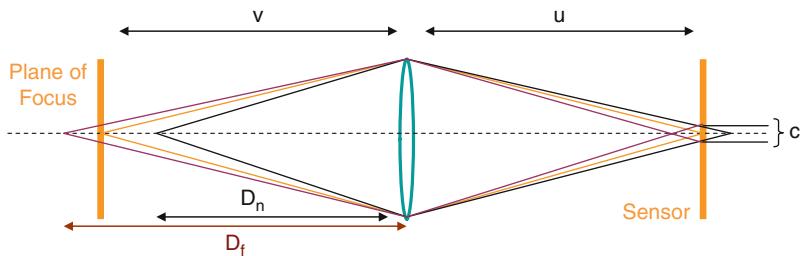
As shown in Fig. 1 the DOF region in front of the plane of focus is not equal to the DOF region behind it. Multi-focus images can be obtained by capturing several images, each under a different focus setting of the lens (change of u). Note that other camera parameters such as aperture size, exposure time, and zoom can also be modified along with the focus setting of the lens during the capture.

Minimizing the Capture Time

Since multi-focus images require the entire scene to remain static during the capture process, it is important to decrease the overall capture time. In [1], the problem of imaging a scene with a given depth of field at a given exposure level in the shortest amount of time was considered. The criteria for optimal capture sequence were derived. Since the light throughput is quadratic with respect to the aperture size but linear with respect to the exposure time, increasing the aperture size is more beneficial to reduce the overall capture time. Intuitively, one should use a large aperture and sweep the focus such that all scene points are sharp in at least one of the captured image.

Multi-focus Images, Fig. 1

DOF for a thin lens with focal length f lies between D_f and D_n



Combining Multi-focus Images

Multi-focus images can be combined to generate an image with larger DOF than any of the individual source images. This is also known as focus stacking and is especially useful in macro photography and optical microscopy. The resultant image is equivalent to the photo captured using a small aperture but will be significantly less noisy. Haeberli [2] showed how to combine multi-focus images by choosing each pixel intensity from the image where it appears to be the sharpest. The sharpness measure can be defined using local variance or local image gradients. Recent approaches have used an energy minimization framework to combine multi-focus images using fast techniques such as graph-cuts [3].

Depth from Defocus and Focus

An additional advantage of multi-focus images is that they can be used to estimate the scene depths, since the defocus blur of a scene point is related to its depth. Depth from defocus is an active area of research in computer vision. Depth from defocus techniques model the relationship between depth and defocus blur using a parametric function and use it to estimate depth from several defocused images. See [4] for a review on such techniques. On the other hand, depth from focus [5, 6] approaches use a focus measure to identify the focus setting for each pixel, which is converted to the metric depth via calibration.

Extended Depth of Field

Focus bracketing is an easy and practical solution to increase the DOF of imaging systems without any hardware modifications. The underlying problem of extending depth of field has several other interesting solutions such as (a) aperture modification [7, 8], (b) light-field based digital refocusing [7, 9–12], (c) lens modifications [13–15], and (d) sensor motion [16].

Application

Extending the DOF has applications in consumer and sports photography, as well as scientific imaging. Multi-focus images offer a viable solution if the scene is changing slowly compared to the capture time required for focus bracketing.

Open Problems

Applying multi-focus images to a dynamic scene is an open and interesting problem.

References

- Hasinoff S, Kutulakos KN (2008) Light-efficient photography. In: Proceeding of the 10th european conference on computer vision (ECCV), Marseille, pp 45–59
- Haeberli P (1994) A multifocus method for controlling depth of field. <http://www.sgi.com/grafica/depth/index.html>
- Agarwala A, Dontcheva M, Agrawala M, Drucker S, Colburn A, Curless B, Salesin D, Cohen M (2004) Interactive digital photomontage. ACM Trans Gr 23(3):294–302
- Chaudhuri S, Rajagopalan A (1999) Depth from defocus: a real aperture imaging approach. Springer, New York
- Grossmann P (1987) Depth from focus. Pattern Recognit Lett 5:63–69
- Nayar S (1992) Shape from focus system. In: Proceedings of the conference on computer vision and pattern recognition (CVPR), Orlando, pp 302–308
- Veeraraghavan A, Raskar R, Agrawal A, Mohan A, Tumblin J (2007) Dappled photography: Mask enhanced cameras for heterodyned light fields and coded aperture refocusing. ACM Trans Gr 26(3):69:1–69:12
- Levin A, Fergus R, Durand F, Freeman WT (2007) Image and depth from a conventional camera with a coded aperture. ACM Trans Gr 26(3):70
- Ng R, Levoy M, Brifkdfid M, Duval G, Horowitz M, Hanrahan P (2005) Light field photography with a hand-held plenoptic camera. Technical report, Stanford University
- Ng R (2005) Fourier slice photography. ACM Trans Gr 24:735–744

11. Georgiev T, Zheng C, Nayar S, Curless B, Salasin D, Intwala C (2006) Spatio-angular resolution trade-offs in integral photography. In: Eurographics symposium on rendering, Nicosia, 263–272
12. Liang CK, Lin TH, Wong BY, Liu C, Chen H (2008) Programmable aperture photography: multiplexed light field acquisition. ACM Trans Gr 27(3):55:1–55:10
13. Green P, Sun W, Matusik W, Durand F (2007) Multi-aperture photography. ACM Trans Gr 26(3):68:1–68:7
14. Dowski ER, Cathey W (1995) Extended depth of field through wavefront coding. Appl Opt 34(11):1859–1866
15. Levin A, Hasinoff SW, Green P, Durand F, Freeman WT (2009) 4d frequency analysis of computational cameras for depth of field extension. ACM Trans Gr 28(3):97:1–97:14
16. Nagahara H, Kuthirummal S, Zhou C, Nayar S (2008) Flexible depth of field photography. In: Proceeding of the european conference of computer vision (ECCV), Marseille

Multimedia Retrieval

► Video Retrieval

Multiple Similarity Method

► Subspace Methods

Multiple View Geometry

► Epipolar Geometry

Multiple View Stereo

► Multiview Stereo

Multiplexed Illumination

Marina Alterman

Department of Electrical Engineering, Technion – Israel Institute of Technology, Haifa, Israel

Synonyms

Multiplexed sensing

Definition

In multiplexed illumination, multiple light sources are used simultaneously in different measurements of intensity data arrays. Then, the intensity under individual sources is derived by computational demultiplexing. This scheme enhances the results: it increases the signal-to-noise ratio of intensity data arrays, without increasing acquisition resources such as time. It also improves dynamic range.

Background

Measuring a set of variables is a common task. For example, in computer vision and graphics, there is a need to acquire multiple images under various lighting conditions; in spectroscopy, there is a need to measure several wavelength bands; in tomography, measurements are taken at a set of different directions; in microscopy, there is a set of focal planes or a set of measurements under several fluorescence excitation wavelength bands. Usually these variables are measured sequentially.

The measurements are subjected to noise, which may yield a low signal-to-noise ratio (SNR). In many cases, the SNR cannot be improved simply by increasing the illumination of individual sources or the exposure time. Simultaneously combining signals corresponding to multiple variables into a single multiplexed measurement may be more efficient. This way, in some cases, the total intensity of multiplexed signals increases relative to the noise. The acquired multiplexed measurements are demultiplexed by a computer, yielding an array of intensity values with a higher SNR.

Theory

Basic Multiplexing

Often, sensors seek measurement of a vector of observable intensity variables \mathbf{i} . Generally, the acquired raw measurements form a vector \mathbf{a} of length N_{measure} . This raw vector is related to \mathbf{i} by

$$\mathbf{a} = \mathbf{Wi} + \boldsymbol{\eta}, \quad (1)$$

where η is a vector of measurement noise (the noise is uncorrelated in different measurements). Here \mathbf{W} is a weighting matrix, referred to as a *multiplexing code*. One case is $\mathbf{W} = \mathbf{I}$, where \mathbf{I} is the identity matrix. This special case is referred to as *trivial* sensing: only a single component of \mathbf{i} is acquired at a time. More generally, multiple components of \mathbf{i} can be simultaneously summed up (multiplexed) and acquired in each raw measurement. The components of \mathbf{i} included in the m th measurement are determined by the m th row of \mathbf{W} . After the measurements are taken, the vector \mathbf{i} of intensities corresponding to the individual source can be decoded from the vector of measurements \mathbf{a} , using

$$\hat{\mathbf{i}} = \mathbf{W}^{-1} \mathbf{a} \quad (2)$$

(when \mathbf{W} is invertible) or by an estimator such as least squares.

A simple case in which $N_{\text{sources}} = 3$ is depicted in Fig. 1. There, two sources are used simultaneously per acquired measurement. For general number of sources, the vectors \mathbf{i} and \mathbf{a} are related by a linear superposition as in Eq. (1). The elements $w_{m,s}$ of \mathbf{W} represent [13] the normalized radiance of source s in measurement m . If $w_{m,s} = 0$, then source s is turned off completely at measurement m ; if $w_{m,s} = 1$, then this source irradiates the object, at the source's maximum power. Generally, $0 \leq w_{m,s} \leq 1$.

The mean squared error (MSE) [5] of $\hat{\mathbf{i}}$ is

$$\text{MSE}_{\mathbf{i}} = \frac{\sigma^2}{N_{\text{sources}}} \text{tr} \left[(\mathbf{W}^T \mathbf{W})^{-1} \right], \quad (3)$$

where tr is a trace operation and σ^2 is the variance of η . Based on Eqs. (2) and (3), \mathbf{i} can be reconstructed, with a potentially higher SNR [5, 11, 13] than \mathbf{i} which is *trivially* sensed using \mathbf{I} . The *gain* of using multiplexing (termed *multiplex advantage*) is defined [5] as

$$\text{GAIN}_{\mathbf{i}} = \sqrt{\sigma^2 / \text{MSE}_{\mathbf{i}}}. \quad (4)$$

Most related studies have aimed to maximize the SNR of the recovered images $\hat{\mathbf{i}}$. Thus, Refs. [5, 9, 11, 13] sought a multiplexing matrix that minimizes Eq. (3):

$$\hat{\mathbf{W}}_{\mathbf{i}} = \arg \min_{\mathbf{W}} \text{MSE}_{\mathbf{i}}. \quad (5)$$

An optimal multiplexing code should yield the highest SNR of the demultiplexed values. When the signal dependency of noise is not considered, the

optimal multiplexing codes are based on Hadamard matrices [5].

According to the affine noise model [11], the detector noise variance is composed of two components, signal dependent and signal independent. The gray-level variance of the signal-independent noise is denoted by κ_{gray}^2 . Considering a diffuse object, each light source yields a similar object radiance. Therefore, each source yields a similar level of noise. If C sources are activated in their maximum power, the total noise variance of the measured gray level is [11, 14]

$$\sigma^2 = \kappa_{\text{gray}}^2 + C\mu^2, \quad (6)$$

where μ^2 is the photon noise variance, caused by object irradiance from a single source activated at its maximum power. If photon noise dominates the noise ($C\mu^2 \gg \kappa_{\text{gray}}^2$), Hadamard multiplexing codes degrade the decoded images. The reason is that simultaneous sources increase the image intensity, which in turn increases the photon noise. Therefore, there is a need for generalization of the multiplexing model to obtain new and improved multiplexing codes.

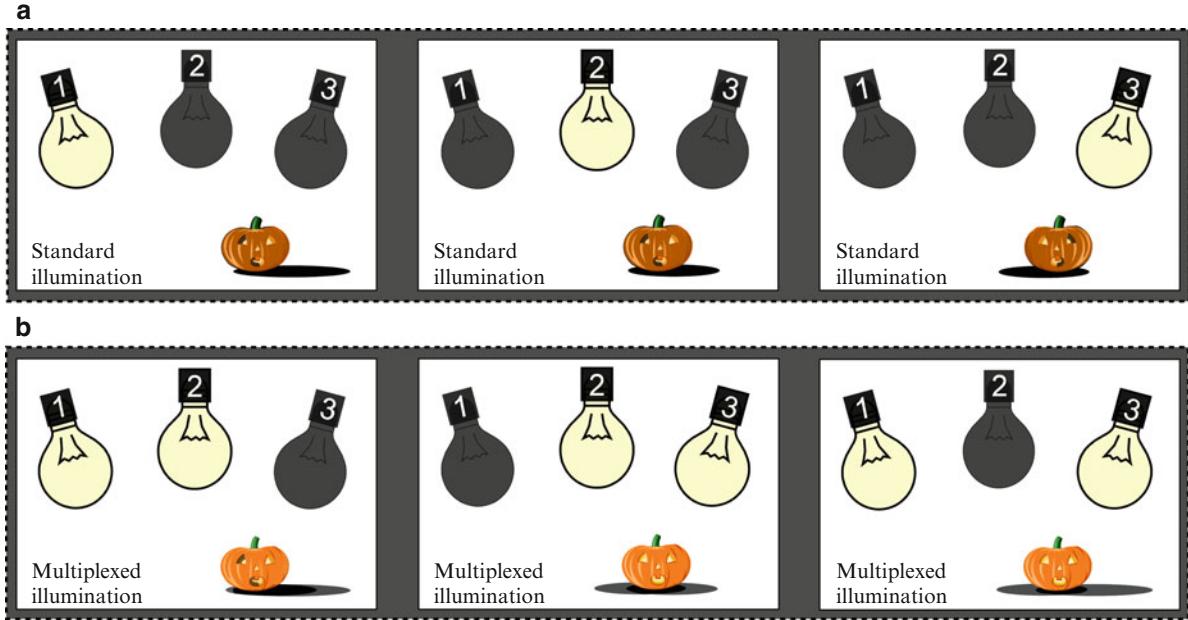
Generalized Multiplexing

References [11, 12] derive optimal multiplexing codes considering photon noise and saturation. Saturation occurs when the total illumination radiance exceeds a certain threshold. If all light sources yield a similar object radiance, then this threshold C_{sat} is expressed in units of light sources ($C = C_{\text{sat}}$). It is preferable to exploit the maximum radiance for every measurement. Thus, to account for saturation, the constraint

$$\sum_{s=1}^{N_{\text{sources}}} w_{m,s} = C_{\text{sat}} \quad (7)$$

is added to the optimization problem in Eq. (5). By scanning a range of C_{sat} values in Eq. (7) and using $C = C_{\text{sat}}$ in Eqs. (3) and (6), the value that yields the maximum gain Eq. (4) is found. This accounts both for saturation and photon noise.

However, is a demultiplexed array \mathbf{i} the true goal of a vision system? Often not. Usually, the recovered intensity or reflectance array \mathbf{i} is by itself an input to further analysis. For example, a multispectral imager



Multiplexed Illumination, Fig. 1 Three light sources illumination [13]. (a) Standard (trivial) illumination: single light source is active in each measurement. (b) Multiplexed illumination: two light sources are active in each measurement

may recover a scene's spectral datacube, and multiplexing is helpful in this. But the resulting multispectral datacube itself is of little interest *per se*: usually (e.g., in remote sensing) the user is interested in the underlying spatial distribution of *materials* or objects that created the spectral data. This is formulated as a mixing model

$$\mathbf{i} = \mathbf{X} \mathbf{c}, \quad (8)$$

where \mathbf{X} is a *mixing* matrix. The end product of interest in this example is *not* demultiplexed intensities or spectral reflectance (\mathbf{i}) but a distribution of materials (\mathbf{c}). Similarly, in multispectral imaging of fluorescing specimen, intensities (\mathbf{i}) are just a means to obtain information about molecular distributions in the specimen (\mathbf{c}). Recovery of \mathbf{c} based on \mathbf{i} is called *unmixing*.

Reference [2] showed that unmixing can (and should) be fully integrated when optimizing the multiplexing codes. Otherwise, the true underlying variables of interest may be *harmed* by multiplexing. Let multiple sources be active in each measurement (intentional multiplexing). Using Eqs. (1) and (8), \mathbf{c} can be estimated, for example, by using weighted least squares

$$\hat{\mathbf{c}} = [(\mathbf{W}\mathbf{X})^T \Sigma_{\text{noise}}^{-1} (\mathbf{W}\mathbf{X})]^{-1} (\mathbf{W}\mathbf{X})^T \Sigma_{\text{noise}}^{-1} \mathbf{a}, \quad (9)$$

where Σ_{noise} is the covariance matrix of the raw measurement noise η . Then, the MSE of \mathbf{c} is

$$\text{MSE}_{\mathbf{c}} = \frac{1}{N_{\text{materials}}} \text{tr} \left\{ [(\mathbf{W}\mathbf{X})^T \Sigma_{\text{noise}}^{-1} (\mathbf{W}\mathbf{X})]^{-1} \right\}, \quad (10)$$

where $N_{\text{materials}}$ is the number of materials to unmix. To multiplex measurements in a way that optimally recovers \mathbf{c} (*multiplexed unmixing*), a multiplexing matrix \mathbf{W} that minimizes the MSE of \mathbf{c} Eq. (10) is sought,

$$\hat{\mathbf{W}}_{\mathbf{c}} = \arg \min_{\mathbf{W}} \text{MSE}_{\mathbf{c}}. \quad (11)$$

Application

Multiplexed illumination is used in multispectral imaging (array of spectral bands) [3, 9], spectroscopy [4], and lighting (reflection from an array of light sources) [8, 11, 13]. It also has analogue formulations in coded apertures (array of spatial positions or viewpoints) and coded shuttering [1, 6, 7, 10] (array of spatiotemporal pixel values).

References

1. Agrawal A, Raskar R (2009) Optimal single image capture for motion deblurring. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), Miami
2. Alterman M, Schechner YY, Weiss A (2010) Multiplexed fluorescence unmixing. In: IEEE ICCP, Cambridge
3. Ben-Ezra M, Wang J, Wilburn B, Li X, Ma L (2008) An LED-only BRDF measurement device. In: IEEE conference on computer vision pattern recognition (CVPR), Anchorage
4. Cull EC, Gehm ME, Brady DJ, Hsieh CR, Momtahan O, Adibi A (2007) Dispersion multiplexing with broadband filtering for miniature spectrometers. *Appl Opt* 46:365–374
5. Harwit M, Sloane NJA (1979) Hadamard transform optics. Academic, New York
6. Levoy M, Chen B, Vaish V, Horowitz M, McDowall I, Bolas M (2004) Synthetic aperture confocal imaging. *ACM Trans Graph* 23:825–834
7. Liang CK, Lin TH, Wong BY, Liu C, Chen HH (2008) Programmable aperture photography: multiplexed light field acquisition. In: ACM TOG. ACM Trans Graph, New York, pp 1–10
8. Narasimhan S, Koppal S, Yamazaki S (2008) Temporal dithering of illumination for fast active vision. In: Proceedings of the European conference on computer vision (ECCV), Marseille, pp 830–844
9. Park J, Lee M, Grossberg MD, Nayar SK (2007) Multispectral imaging using multiplexed illumination. In: Proceedings of the IEEE ICCV, Rio de Janeiro
10. Raskar R, Agrawal A, Tumblin J (2006) Coded exposure photography: motion deblurring using fluttered shutter. *ACM Trans Graph* 25:795–804
11. Ratner N, Schechner YY (2007) Illumination multiplexing within fundamental limits. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), Minneapolis
12. Ratner N, Schechner YY, Goldberg F (2007) Optimal multiplexed sensing: bounds, conditions and a graph theory link. *Opt Express* 15:17072–17092
13. Schechner YY, Nayar SK, Belhumeur PN (2007) Multiplexing for optimal lighting. *IEEE Trans PAMI* 29:1339–1354
14. Wuttig A (2005) Optimal transformations for optical multiplex measurements in the presence of photon noise. *Appl Opt* 44:2710–2719

Multiplexed Sensing

► [Multiplexed Illumination](#)

Multisensor Data Fusion

► [Sensor Fusion](#)

Multiview Geometry

► [Epipolar Geometry](#)

Multiview Stereo

Sudipta N. Sinha

Microsoft Research, Redmond, WA, USA

Synonyms

[Multiple view stereo](#)

Related Concepts

► [Dense Reconstruction](#); ► [Multi-baseline Stereo](#)

Definition

Multiview stereo refers to the task of reconstructing a 3D shape from calibrated overlapping images captured from different viewpoints. Various representations of 3D shape can be used. For example, dense 3D point cloud or surface mesh representations are common in applications that synthesize a new photorealistic image of the scene using computer graphic rendering techniques. The topics of multiview stereo and multi-baseline stereo matching share key concepts related to the recovery of dense 2D pixel correspondences in multiple images.

Background

Reconstructing 3D geometry from images (often also called *3D photography*) involves using cameras or optical sensors (and optionally illumination) to acquire the 3D shape and appearance of objects and scenes in the real world. Existing methods can be broadly divided into two categories – *active* and *passive*

methods. Active methods usually require additional special light sources, whereas passive methods work with natural lighting. Active methods often use special-purpose sensors such as laser range scanners and depth sensors (Kinect, time-of-flight cameras) and can often capture high-quality 3D models. However, developing passive methods for ordinary cameras is also important because of their greater ease, flexibility, and wider applicability.

Multiview stereo is a popular and well-studied passive 3D reconstruction technique. It is based on the principle that dense pixel correspondences in multiple calibrated images captured from different viewpoints makes it possible to estimate the 3D shape of the scene via triangulation, which involves intersecting rays backprojected from the corresponding pixels. Multiview stereo therefore requires the camera calibration parameters to be known. These parameters can either be precomputed offline or can be computed from the image sequence using structure from motion algorithms. Figure 1 shows an example of a 3D reconstruction obtained using multiview stereo.

The main challenge in multiview stereo lies in computing precise, dense pixel correspondence between images. Difficulties arise due to ambiguities in matching pixels in two images of the same scene. Multiview stereo works best when surfaces are textured and *Lambertian*, i.e., when the local appearance of a surface patch does not depend on the viewing angle. Glossy or specular surfaces are non-Lambertian and are more difficult to handle compared to diffuse surfaces. Occlusions can complicate the situation even further, especially in the *wide baseline* setting where the camera viewpoints are farther from each other compared to the *narrow baseline* setting, where the effect of occlusions is less pronounced.

For reconstructing a static scene with multiview stereo, a single camera can be used to capture images from multiple viewpoints over time. Sometimes, images are captured with a static camera with the object placed on a rotating turntable [1]. These techniques can also be applied to dynamic scenes, provided multiview video is captured from a calibrated, synchronized multi-camera rig. The Virtualized Reality project at CMU [2] was the first to demonstrate multiview 3D reconstruction of dynamic events within a large scene.

Theory

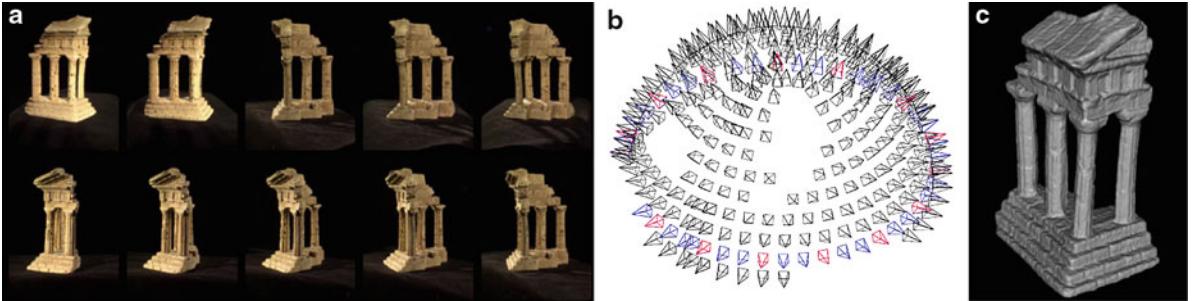
A taxonomy was recently introduced to broadly categorize multiview stereo approaches [3]. It proposed studying various methods based on the following properties: 3D shape representations, the photoconsistency measure, visibility handling, shape priors, and the reconstruction algorithm. A recent tutorial [4] and an earlier survey [5] are also excellent sources of information on this topic.

3D Shape Representation

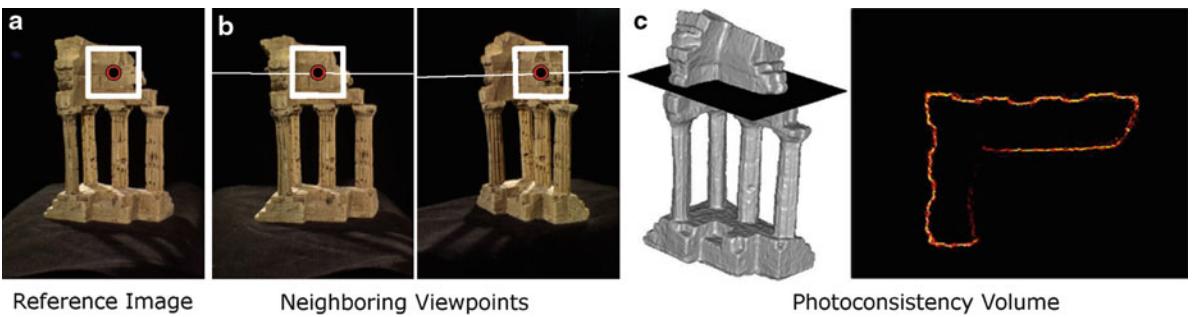
Many multiview stereo methods represent the 3D scene as a set of depth maps, one for each calibrated viewpoint, and recover the 3D shape that is most consistent with all the depth maps [6–8]. Other methods use explicit surface-based representations. Polygonal meshes are used both as an internal representation in some methods [1] and as the representation for the final 3D shape. Volumetric representations such as a uniform 3D voxel grid where each voxel is labeled as occupied or empty are common as well for their flexibility in approximating a wide variety of shapes [9–11]. Other volumetric representations such as adaptive tessellation of 3D space into tetrahedral cells can be more compact [12, 13]. Finally, patch-based representations are also possible [14, 15]. Recent multiview stereo methods represent scenes using a finite set of locally planar, oriented 3D patches, referred to as *surfels*. No connectivity information is stored as in a surface mesh. While surface meshes and volumetric representations are advantageous for reconstructing closed surfaces or 3D objects, depth maps or patch-based representations are often better choices for reconstructing large open scenes [7, 8, 15, 16].

Photoconsistency

It is a key ingredient in multiview stereo methods. It is a measure for the photometric similarity of the 2D projections of any 3D scene point in a set of calibrated images. For 3D points on surfaces visible in a set of cameras, the 2D projections in those images are expected to be similar or photometrically consistent. Such points are said to have high photoconsistency. On the other hand, arbitrary 3D scene points are likely to have low photoconsistency in most



Multiview Stereo, Fig. 1 Images captured with a camera rig. Calibrated cameras and the final 3d model produced by a state-of-the-art multiview stereo method rendered as a triangulated mesh



Multiview Stereo, Fig. 2 (a) A pixel in a reference image and a *square patch* around it. (b) Corresponding pixels and associated patches in the nearby images that lie on respective *epipolar lines*. In calibrated images, the search for correspondences

reduces to 1D, i.e., along the *epipolar line*. (c) A horizontal slice of the 3D cost volume is shown. It is constructed using the photoconsistency measure described in [10]

situations. Some earlier methods used the variance in pixel colors in images where a 3D point is visible, as a measure of photoconsistency. However, since comparing individual pixels can be ambiguous, a more reliable approach is to compare the similarity of image patches around the projections of a 3D point (see Fig. 2). For narrow baselines, it is sufficient to compare image-oriented square patches centered on pixels. However, by incorporating the geometry of the 3D patch, the matching windows can be correctly adjusted to account for scale differences in the images and slanted surfaces.

A pair of 2D image patches can be compared using normalized cross correlation (NCC) after resampling a $n = k \times k$ 2D grid at each patch and comparing the vector of color values denoted here as u and v , respectively:

$$\text{NCC}(u, v) = \frac{\sum_{j=0}^n (u_j - \bar{u})(v_j - \bar{v})}{\sqrt{\sum_{j=0}^n (u_j - \bar{u})^2 \cdot \sum_{j=0}^n (v_j - \bar{v})^2}} \quad (1)$$

The NCC-based similarity measure lies in the range $[-1, 1]$ where a value closer to 1 indicates that the vectors are similar in appearance. Other similarity measures such as the sum of absolute differences (SAD), the sum of squared differences (SSD), or nonparametric measures can be used to compute photoconsistency as well. Unlike NCC, SAD or SSD are affected by brightness changes in different images.

Photoconsistency Volume Computation

Many multiview stereo methods require the photoconsistency function to be evaluated densely on a 3D voxel grid for the volume containing the scene which is then referred to as the *cost volume*. Figure 2 shows an example. One simple way to construct this cost volume is to evaluate the photoconsistency of all 3D points (or voxels) using a pairwise similarity measure such as NCC and then compute the mean NCC score from multiple pairs of nearby images. There also exist direct methods to compute the cost volume [17] or approaches that are better at minimizing noise in the photoconsistency

estimates [10]. Being able to estimate the visibility of a 3D point allows one to select which images contribute to its photoconsistency measure. A coarse approximation of the shape such as the visual hull reconstruction obtained from silhouettes is often sufficient when reconstructing closed objects [1, 10, 12]. However, when a large number of images are available, occlusions can also be treated as outliers and explicit visibility reasoning is not required [10, 14].

Another general approach to construct the cost volume involves first estimating depth maps from each camera’s viewpoint using a state-of-the-art multi-baseline stereo matching algorithm and then merging them into a consistent volumetric representation. This sort of aggregation incorporates visibility information induced by the independently estimated depth maps. Intuitively, this can be thought of as a way of probabilistically carving out the 3D volume [6, 18], an idea that is closely related to the problem of fusion of range images [19].

Plane Sweep Stereo and Depth Map Fusion

A class of methods referred to as plane sweep stereo methods avoid building the cost volume on a uniform 3D grid. Instead, they sweep the scene with a set of parallel planes corresponding to the candidate depths considered for pixels in the reference image. These planes are typically fronto parallel to the reference camera, and their spacing is selected in uniform intervals with respect to their inverse depth from the camera. Neighboring images are warped on to these planes using 2D homographies, and the photoconsistency measure is computed at each pixel for each candidate depth plane. Plane-sweeping strategies are often chosen by incorporating some knowledge of scene structure and are popular for 3D reconstruction in urban scenes [7]. Their main advantage over uniform voxel grids or other 3D tessellations lies in the fact that they represent large working volumes more efficiently, which is important for reconstructing large open scenes. Plane sweep stereo can be combined with depth map fusion which can be implemented using an image space representation [8]. Artifacts in the original depth maps are reduced in the fused depth maps from which triangulated surface meshes can be directly extracted. The fusion step works better when per-pixel confidence estimates associated with the depth estimates are available.

Optimization Methods

Broadly all multiview stereo methods formulate the reconstruction task in terms of a *local* or *global* optimization problem. A local method such as space carving [9] starts with an overestimate of the 3D scene, and uses a greedy strategy to remove voxels that are not photoconsistent one at a time. Similarly, in fast plane sweep stereo [7], each pixel’s depth is computed independent of other pixels. These local methods are susceptible to noise and outliers and cannot easily reconstruct smooth geometric shapes. Part of the difficulty is that the 3D reconstruction task is inherently ill-posed since different 3D scenes can be consistent with the same set of images. By assuming that surfaces in the scene are primarily smooth, various global optimization methods seek the optimal 3D shape which maximizes both photoconsistency and smoothness. This is achieved by formulating multiview stereo as a global optimization problem with geometric regularization terms in the objective function or energy function. These are minimized either in the discrete or in the continuous setting.

Global methods for depth map estimation incorporate image-based smoothness constraints into the formulation by designing suitable energy functions that encourages neighboring pixels to take on identical or similar depth values. Such methods are often based on a 2D Markov Random Field (MRF) framework for which efficient optimization algorithms have been developed in recent years. The MRF framework can also be used for enforcing surface regularization in volumetric methods on a 3D uniform grid [10] where any binary labeling of voxels in the grid corresponds to some 3D shape. The optimal surface corresponding to the global minimum of the energy function can be efficiently computed using graph cut algorithms in many situations [1, 18]. In case of [12, 13], the minimal surface computed using graph cuts directly produces a triangulated mesh. These methods first solve a discrete optimization problem to obtain a globally consistent solution and then performs local refinement using continuous optimization to recover finer details in the structure [1, 12]. As an alternative, direct continuous optimization methods based on convex relaxations have also been developed to address discretization artifacts which can occur in graph cut-based methods [11].

Variational methods can also be used to extract an optimal smooth surface from the cost volume using

multiple iterations that progressively minimize a global objective function. In [1], this was achieved by evolving a deformable surface mesh using photoconsistency cues. However, a good initial guess for the 3D shape was required. Methods based on level sets are more flexible as the surface topology is allowed to change during the iterations. These approaches represent the surface as the zero level set of an evolving implicit function [20]. The energy function is minimized by modeling the evolution of the function using partial differential equations. These level-set methods have the advantage of producing smooth surface reconstructions but have the disadvantage of being susceptible to local minima unless a good initialization is available.

Patch-Based Multiview Stereo

Another class of popular multiview stereo approaches represent surfaces as a set of oriented patches without storing any connectivity information. This makes the representation flexible and suitable for reconstructing both closed 3D shapes and open scenes. A watertight surface mesh can be extracted using the Poisson surface reconstruction algorithm as a post-processing step. Patch-based methods are related to region-growing methods as they typically start from a few seed 3D points with known depth. The algorithm proposed in [14] starts with a sparse structure from motion 3D point cloud and iteratively optimizes the depth and normals of the oriented patches using an NCC-based photoconsistency measure. The pixels in the reference image are processed in a matching score-based priority order that ensures that confident estimates are propagated first in the surface-growing phase. This step produces semi-dense depth maps which are then merged to generate a global set of oriented 3D patches. This method was used to perform 3D reconstruction from Internet images of popular landmarks and showed that dense correspondences could be reliably computed from such large heterogenous image collections as well.

Patch-based multiview stereo (PMVS) [15] is another popular algorithm that also uses a seed and grow reconstruction strategy and consists of three important steps. First, seed patches are created from sparse 2D keypoint correspondences in neighboring

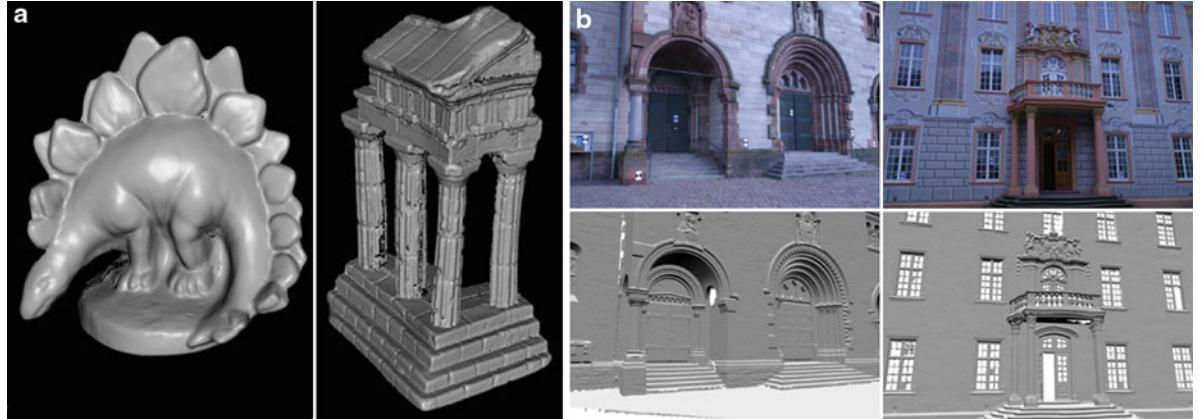
overlapping views which can be matched with high confidence. These patches are iteratively expanded using a locally planar model to generate new patches. This is followed by a patch optimization and filtering step which refines the position and orientation of the patches and then removes noisy or outlier samples based on photoconsistency and visibility constraints. A publicly available implementation of this algorithm is available as part of the PMVS library [15]. Recently, this library was also extended to support large-scale reconstructions from Internet collections [16].

Efficient Multiview Stereo on GPUs

Some multiview stereo methods strive for high-fidelity reconstructions and can be very computationally intensive especially when high-resolution images are processed [1, 14–16, 21]. The main computational bottleneck in multiview stereo lies in the photoconsistency computation or computing matching cost over many pairs of windows. Typically, this can be accelerated by orders of magnitude on massively parallel hardware and is also perfectly suitable for the data-parallelism supported on modern programmable graphics hardware (GPUs) with SIMD architectures. Many variants of multiview stereo ranging from plane sweep stereo [7], depth map fusion [8] to level-sets-based methods [22] have been successfully ported to the GPU, and one or two orders of magnitude speedup have been demonstrated.

Benchmarks

The Middlebury multiview stereo datasets shown in Fig. 3a contain ground truth 3D models created by scanning the models using a laser stripe scanner and registering the 3D mesh to the calibrated images captured with a gantry. The benchmark has recently been quite popular for evaluating multiview stereo algorithms [3]. It uses two criteria to evaluate the reconstructions – *accuracy* and *completeness*. The model’s accuracy is calculated by computing the distance between the points sampled on the reconstructed model and the nearest points on the ground truth model and reporting the distance (in mm) such that 90% of the points on the reconstructed model are within that distance from the ground truth model. Similarly, the



Multiview Stereo, Fig. 3 Multiview stereo benchmarks used for quantitative evaluation: (a) Middlebury *Dino* and *Temple* datasets from the Middlebury multiview stereo benchmark

(<http://vision.middlebury.edu/mview/>). (b) Large scenes from the outdoor multiview stereo benchmark (<http://cvlab.epfl.ch/strecha/multiview/denseMVS.html>)

completeness measure is computed for a given threshold by finding the nearest point on the reconstructed mesh for each vertex in the ground truth mesh and the percentage of points on the ground truth model that is within a distance threshold (default value = 1.25 mm) of the reconstructed model.

Another benchmarks for evaluating multiview stereo reconstruction of large scenes is also available [23]. Precise laser scanned models are provided for ground truth comparisons. Unlike Middlebury where the scanned models are quite small (only 16 cm on the longest dimension), these datasets consist of high-resolution images and much larger outdoor scenes. Two of the scenes captured in this benchmark are shown in Fig. 3b. Several multiview stereo methods have demonstrated accurate result on these datasets.

5. Slabaugh GG, Culbertson WB, Malzbender T, Stevens MR, Schafer RW (2004) Methods for volumetric reconstruction of visual scenes. *Int J Comput Vis* 57: 179–199

6. Goesele M, Curless B, Seitz SM (2006) Multi-view stereo revisited. *CVPR '06*, vol 2, pp 2402–2409. New York, USA

7. Gallup D, Frahm JM, Mordohai P, Yang Q, Pollefeys M (2007) Real-time plane-sweeping stereo with multiple sweeping directions. In: *IEEE conference on computer vision pattern recognition (CVPR)*. Minneapolis, USA

8. Merrell P, Akbarzadeh A, Wang L, Mordohai P, Frahm JM, Yang R, Nistér D, Pollefeys M (2007) Real-time visibility-based fusion of depth maps. In: *ICCV*, pp 1–8. Rio de Janeiro, Brazil

9. Kutulakos K, Seitz S (2000) A theory of shape by space carving. *Int J Comput Vis* 38(3):199–218

10. Vogiatzis G, Esteban CH, Torr PHS, Cipolla R (2007) Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency. *IEEE Trans Pattern Anal Mach Intell* 29(12):2241–2246

11. Kolev K, Klodt M, Brox T, Cremers D (2009) Continuous global optimization in multiview 3d reconstruction. *Int J Comput Vis* 84(1):80–96

12. Sinha SN, Mordohai P, Pollefeys M (2007) Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh. In: *ICCV*. Rio de Janeiro, Brazil

13. Vu HH, Keriven R, Labatut P, Pons JP (2009) Towards high-resolution large-scale multi-view stereo. In: *Conference on computer vision and pattern recognition (CVPR)*. Miami Beach, USA

14. Goesele M, Snavely N, Curless B, Hoppe H, Seitz SM (2007) Multi-view stereo for community photo collections. In: *ICCV*, pp 1–8. Rio de Janeiro, Brazil

15. Furukawa Y, Ponce J (2007) Accurate, dense, and robust multi-view stereopsis. In: *Computer vision and pattern recognition (CVPR)*, pp 1–8. Minneapolis, USA

16. Furukawa Y, Curless B, Seitz SM, Szeliski R (2010) Towards internet-scale multi-view stereo. In: *IEEE*

References

1. Hernández C (2004) Stereo and silhouette fusion for 3D object modeling from uncalibrated images under circular motion. PhD thesis, Ecole Nationale Supérieure des Télécommunications
2. Kanade T, Rander P, Narayanan PJ (1997) Virtualized reality: constructing virtual worlds from real scenes. *IEEE MultiMed* 4(1):34–47
3. Seitz SM, Curless B, Diebel J, Scharstein D, Szeliski R (2006) A comparison and evaluation of multi-view stereo reconstruction algorithms. In: *Computer vision and pattern recognition (CVPR)*, vol 1, pp 519–528. New York, USA
4. Hernandez C, Vogiatzis G, Furukawa Y (2010) CVPR Tutorial. <http://carlos-hernandez.org/cvpr2010/>

- conference on computer vision pattern recognition (CVPR). San Francisco, USA
17. Hornung A, Kobbelt L (2006) Robust and efficient photo-consistency estimation for volumetric 3d reconstruction. *ECCV'06*, vol 2, pp 179–190. Graz, Austria
 18. Hernández C, Vogiatzis G, Cipolla R (2007) Probabilistic visibility for multi-view stereo. In: IEEE conference on computer vision pattern recognition (CVPR). Minneapolis, USA
 19. Curless B, Levoy M (1996) A volumetric method for building complex models from range images. In: Proceedings of the 23rd annual conference on computer graphics and interactive techniques. *SIGGRAPH '96*, pp 303–312. New Orleans, USA
 20. Pons JP, Keriven R, Faugeras O (2007) Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *Int J Comput Vis* 72(2): 179–193
 21. Tola E, Strecha C, Fua P (2011) Efficient large-scale multi-view stereo for ultra high-resolution image sets. *J Mach Vis Appl* 23(5):903–920
 22. Labatut P, Keriven R, Pons JP (2006) Fast level set multi-view stereo on graphics hardware. In: International symposium on 3D data processing visualization and transmission. Chapel Hill, USA
 23. Strecha C, von Hansen W, Gool LJV, Fua P, Thoennessen U (2008) On benchmarking camera calibration and multi-view stereo for high resolution imagery. In: IEEE conference on computer vision pattern recognition (CVPR). Anchorage, USA

Mutual Illumination

► [Interreflections](#)

N

Network Camera

- ▶ Pan-Tilt-Zoom (PTZ) Camera

Noise Removal

- ▶ Denoising

Non-central Camera Calibration

- ▶ Calibration of a Non-single Viewpoint System

Nonlinear Diffusion

- ▶ Diffusion Filtering

Normalcy Modeling

- ▶ Change Detection

Numerical Methods in Curve Evolution Theory

Peter Karasev, Ivan Kolesov and Allen Tannenbaum
Schools of Electrical and Computer and Biomedical
Engineering, Georgia Institute of Technology,
Atlanta, GA, USA

Synonyms

Active contours; Numerical partial differential equations

Definition

Two numerical approximations of curvature-driven flows are described for use in computer vision and image processing. The level-set methodology of Osher-Sethian that dominates the field is of particular interest, and a more recent, alternative interpretation of curvature flow, based on a stochastic evolution of density functions, is also presented.

Background

Two separate methods for the numerical approximation and computer implementation of curvature-driven flows in the plane are described. The first has become, perhaps, the standard in the field: the level-set methodology of Osher and Sethian [20, 21, 24, 25]. This approach and its variants (fast marching, narrow banding, etc.) dominate the literature and practice in computer vision and image processing.

There are, however, other possibilities that may be useful when estimation is necessary. For example, when tracking in a noisy, uncertain environment, the lack of a geometric prior tends to result in undesired topological changes (e.g., merging and breaking of contours) in response to noise in the image data. There are several possibilities that may be considered for

such scenarios described in books [7, 8]. A completely stochastic method from [3, 4] also acts as a useful approximation in many circumstances. A stochastic interpretation of curve-shortening flows is formulated, bringing together the theories of curve evolution and hydrodynamical limits, and thus impacts the growing use of joint methods from probability and PDEs in image processing and computer vision.

It is emphasized that the various ways of numerically implementing curvature-driven flows in vision described above should be regarded as complementary: each approach has its own advantages depending on the application. Thus, it is not intended to compare or critique the methods in the present work; rather, they described in some detail and give the interested reader a guide to the relevant literature. Thus, this entry will have a strong tutorial flavor. Finally, derivations herein are not considered in complete generality. For most of the entry, simply consider the simplest case of Euclidean curve shortening [13–15].

Theory

Evolution Equations

This section reviews mathematical background on the basic evolution equations used in computer vision. For complete details, the interested reader is referred to books [22, 23] that also have an extensive list of references.

Consider the families of embedded closed curves $\mathcal{C} : S^1 \times (0, T) \rightarrow \mathbb{R}^2$ evolving according to functions of the curvature. Here, S^1 denotes the unit circle. (Since this discussion is informal, strong restrictions are not placed of the families of curves under consideration. See [5, 13–15] a formal mathematical treatment.) More precisely, in vision problems, the general deformation of a curve in the plane of interest is given by

$$\frac{\partial \mathcal{C}}{\partial t} = \alpha(p, t)\mathcal{T} + \hat{\beta}(p, t)\mathcal{N}, \quad (1)$$

where \mathcal{N} is the unit (inward) normal, \mathcal{T} is unit tangent, and $\alpha, \hat{\beta}$ are smooth functions. Note that positive orientation of a curve is defined such that the interior is to the left when traversing the curve. The *curvature*, *orientation*, and *length* are defined in the standard way. The normal is taken to point inward, where the inward

or outward directions are determined by the interior or equivalently by the orientation of the curve. Notice that since only shape is of interest, it is admissible to take $\alpha = 0$; changing α only changes the curve's parametrization and not its shape. Furthermore, as is typical in this area, the deformations are constrained to be determined by the local geometry of the curve, that is, $\hat{\beta}(p, t) = \beta(\kappa)$ where $\kappa(p, t)$ denotes the Gaussian curvature of the curve $\mathcal{C}(p, t)$. The following basic equations are obtained for curve evolution under curvature-driven flow:

$$\frac{\partial C}{\partial t} = \beta(\kappa)\mathcal{N}. \quad (2)$$

In the mathematics literature, a number of cases for the function β have been explored. For example, there has been a great deal of work in connection with the geometric heat equation in which $\beta(\kappa) = \kappa$. In this case, the isoperimetric ratio L^2/A (L is the length of the closed curve and A is the area enclosed by the curve) approaches 4π as the enclosed area approaches 0; in other words, the curve shrinks to a *round point* as shown in [13, 14]. In computer vision problems, the function $\beta(\kappa)$ typically takes the form

$$\beta(\kappa) = a\kappa + 1, \quad (3)$$

where $a \geq 0$. The case $a = 0$, that is, $\beta \equiv 1$ is very important; Eq. (2) reduces to an equation that has been studied in relation to problems in geometric optics [6], flame propagation [26], and shape morphology [10], as well as shape decomposition. Indeed, this is the differential equation for the Blum prairie fire model. In (3), $a\kappa$ adds a diffusive effect, while the constant part adds a wave (hyperbolic) effect, which tends to create singularities and break a shape into its constituent parts. This point is elaborated below.

Some general properties of Eq. (2) are presented, beginning with some standard notation. Let

$$\rho(p, t) := \left\| \frac{\partial \mathcal{C}}{\partial p} \right\| = [x_p^2 + y_p^2]^{1/2} \quad (4)$$

denote the length along the curve. The arc-length parameter s is then defined as

$$s(p, t) := \int_0^p \rho(\zeta, t) d\zeta. \quad (5)$$

Then it follows that

$$\mathcal{T} := \frac{\partial \mathcal{C}}{\partial s} = \frac{1}{\rho} \frac{\partial \mathcal{C}}{\partial p}, \quad (6)$$

$$\kappa := \frac{1}{\rho} \left\| \frac{\partial \mathcal{T}}{\partial p} \right\|, \quad (7)$$

$$\mathcal{N} := \frac{1}{\kappa \rho} \frac{\partial \mathcal{T}}{\partial p}, \quad (8)$$

$$L(t) := \int_0^1 \rho(p, t) dp. \quad (9)$$

Finally, let

$$\bar{\kappa}(t) := \int_0^1 |\kappa(p, t)| \rho(p, t) dp \quad (10)$$

denote the total absolute curvature.

Behavior of the classical solutions to Eq. (2) has been thoroughly analyzed and a number of useful results (from the applied point of view) have been proved [5, 17, 21, 24]; the results are summarized below. Let $\mathcal{C}(p, t)$ be a classical solution of Eq. (2) for $t \in [0, t']$ and suppose that $\kappa \beta(\kappa) \leq M$ for all $\kappa \in \mathbf{R}$ (regarding β as a function of κ). Then,

$$L(t) \leq L(0) e^{Mt}. \quad (11)$$

In the case $\beta(\kappa) = a\kappa + 1$,

$$L(t) \leq \min(L(0) + 2\pi t, L(0) e^{\frac{t}{4a}}). \quad (12)$$

Let $\mathcal{C}^{(t)}(p) := \mathcal{C}(p, t)$ be a classical solution of (2) for $t \in [0, t']$. Suppose that $\kappa \beta(\kappa) \leq M$ and $\beta_\kappa \leq 0$. Then,

$$\bar{\kappa}(t) \leq \bar{\kappa}(0). \quad (13)$$

Moreover, if $(0, t')$ is an interval on which a classical solution exists, one may also show that

$$d_H(\mathcal{C}^{(t)}, \mathcal{C}^{(0)}) \leq \sqrt{4at}, \quad (14)$$

where d_H denotes the Hausdorff metric on compact subsets of \mathbf{R}^2 . The above facts easily imply that

$$\lim_{t \rightarrow t'} \mathcal{C}^{(t)} = \mathcal{C}^* \quad (15)$$

in the Hausdorff metric, and curve \mathcal{C}^* , regarded as a mapping $S^1 \rightarrow \mathbf{R}^2$, is Hölder continuous with exponent 1/2. In vision, a major area of interest is weak

solutions of equations of type (2), whose details follow in section ‘‘Geometric Heat Equation’’.

Geometric Heat Equation

In this section, a key flow for nonlinear scale space and active geometric contours is presented. This flow is achieved by setting $\beta(\kappa) = \kappa$ in (2).

So, when $\beta(\kappa) = \kappa$, where κ is the curvature, and \mathcal{N} the inward unit normal, a plane curve evolves according to the *geometric heat equation*

$$\frac{\partial \mathcal{C}}{\partial t} = \kappa \mathcal{N}. \quad (16)$$

This equation has a number of properties which make it very useful in image processing and in particular, the basis of a nonlinear scale space for shape representation [1].

Indeed, Eq. (16) is the Euclidean curve-shortening flow in the sense that the Euclidean perimeter shrinks as quickly as possible when the curve evolves according to Eq. (16) [13–15]. Since similar argument is needed for the snake model discussed in the next section, details are presented.

Let $\mathcal{C} = \mathcal{C}(p, t)$ be a smooth family of closed curves where t parameterizes the family and p the given curve, say $0 \leq p \leq 1$. (Assume that $\mathcal{C}(0, t) = \mathcal{C}(1, t)$ and similarly for the first derivatives.) Consider the length functional

$$L(t) := \int_0^1 \left\| \frac{\partial \mathcal{C}}{\partial p} \right\| dp.$$

After differentiating and using integration by parts,

$$\begin{aligned} L'(t) &= \int_0^1 \frac{\langle \frac{\partial \mathcal{C}}{\partial p}, \frac{\partial^2 \mathcal{C}}{\partial p \partial t} \rangle}{\left\| \frac{\partial \mathcal{C}}{\partial p} \right\|} dp \\ &= - \int_0^1 \langle \frac{\partial \mathcal{C}}{\partial t}, \frac{1}{\left\| \frac{\partial \mathcal{C}}{\partial p} \right\|} \frac{\partial}{\partial p} \left[\frac{\partial \mathcal{C}}{\partial p} \right] \rangle \left\| \frac{\partial \mathcal{C}}{\partial p} \right\| dp \end{aligned}$$

(multiplying and dividing by $\left\| \frac{\partial \mathcal{C}}{\partial p} \right\|$ in the latter integral). Observing that

$$\left\| \frac{\partial \mathcal{C}}{\partial p} \right\| dp =: ds$$

is (Euclidean) arc length, and using the definition of curvature, the last integral becomes

$$-\int_0^{L(t)} \left\langle \frac{\partial \mathcal{C}}{\partial t}, \kappa \mathcal{N} \right\rangle ds$$

and

$$L'(t) = - \int_0^{L(t)} \left\langle \frac{\partial \mathcal{C}}{\partial t}, \kappa \mathcal{N} \right\rangle ds.$$

Using the Cauchy-Schwartz inequality,

$$\left\langle \frac{\partial \mathcal{C}}{\partial t}, \kappa \mathcal{N} \right\rangle \leq \|C_t\|_{L^2} \|\kappa \mathcal{N}\|_{L^2};$$

therefore, the direction in which $L(t)$ is decreasing most rapidly is given by

$$\frac{\partial C}{\partial t} = \kappa \mathcal{N}.$$

Thus, (16) is precisely a gradient flow.

A much deeper fact is that simple closed curves converge to *round points* when evolving according to (16) without developing singularities; see [13, 14]. This fact is key for geometric active contour models considered below.

Level Curve Representations

When applying curvature-driven flows, as above, to images, it is necessary to move from curves to (gray-scale) images. Thus, it is essential to be able to relate curve evolution theory to operations on 2D images. Fortunately, the powerful algorithms developed by Osher and Sethian for curve evolution allow precisely this. In this section, a sketch is given of the beautiful work of Osher and Sethian on level curve evolutions. For more details, see [20, 21, 25].

Let $\mathcal{C}(p, t) : S^1 \times (0, T) \rightarrow \mathbb{R}^2$ be a family of curves satisfying the following evolution equation:

$$\frac{\partial \mathcal{C}}{\partial t} = \beta(\kappa) \mathcal{N}. \quad (17)$$

There are a number of problems which must be resolved when implementing curve evolution equations such as (17) numerically. For example, singularities may develop. Note for $\beta \equiv 1$ in (17) that even a smooth initial curve can develop singularities. The question is how to continue the evolution after the singularities appear. A natural way is to choose

the solution which agrees with the Huygens principle [24, 26], or as Sethian observed, if the front is viewed as a burning flame, this solution is based on the principle that *once a particle is burnt, it stays burnt* [24, 26]. As indicated above that from all the *weak* solutions corresponding to (17), the one derived from the Huygens principle is *unique* and can be obtained via the entropy condition constraint.

In any numerical algorithm, the key requirements of accuracy and stability must be addressed. The numerical algorithm must approximate the evolution equation, and it must be robust. Osher and Sethian [20, 21, 24, 25] showed that a simple, Lagrangian difference approximation requires an impractically small time step in order to achieve stability. The basic problem with Lagrangian formulations is that the marker particles on the evolving curve can become unevenly distributed along the curve during the evolution.

The algorithm proposed by Osher and Sethian [20, 21, 24, 25] provides a reliable numerical solution for curve (and hypersurface) evolution. It is based on the Hamilton-Jacobi equation and viscosity theory. First, the curve is embedded in a two-dimensional surface, and then, the equations of motion are solved using a combination of straightforward discretization and numerical techniques derived from hyperbolic conservation laws [28].

The embedding step is done in the following manner: the curve $\mathcal{C}(p, t)$ is represented by the zero level set of a smooth and Lipschitz continuous function $\Phi : \mathbb{R}^2 \times [0, \tau) \rightarrow \mathbf{R}$. Assume that Φ is negative in the interior and positive in the exterior of the zero level set. The zero level set defined by

$$\{X(t) \in \mathbb{R}^2 : \Phi(X, t) = 0\}. \quad (18)$$

It is necessary to find an evolution equation of Φ such that the evolving curve $\mathcal{C}(t)$ is given by the evolving zero level $X(t)$, that is,

$$\mathcal{C}(t) \equiv X(t). \quad (19)$$

By differentiating $\Phi(X, t)$ with respect to t ,

$$\nabla \Phi(X, t) \cdot X_t + \Phi_t(X, t) = 0. \quad (20)$$

Note that at the zero level, the following relation holds:

$$\frac{\nabla \Phi}{\|\nabla \Phi\|} = -\mathcal{N}. \quad (21)$$

In this equation, the left side uses terms of the surface Φ , while the right side is related to the curve \mathcal{C} . Using Eqs. (17)–(21),

$$\Phi_t = \beta(\kappa) \|\nabla \Phi\|, \quad (22)$$

and the curve \mathcal{C} , evolving according to (17), remains equal to the zero level set of the function Φ , which evolves according to (22), throughout the evolution. Osher and Sethian [21] called this scheme an *Eulerian formulation* for front propagation because it is written in terms of a fixed coordinate system. *Moreover, notice that Eq. (22) gives the extension of curve evolution equations to gray-scale images when Φ is interpreted as intensity.*

The second step of the algorithm consists of discretizing Equation (22). If singularities cannot develop during the evolution, as in the geometric heat equation flow, a straightforward discretization can be performed [21]. If singularities can develop, as in the case of $\beta = 1$, a special discretization must be implemented. In this case, the implementation of the evolution of Φ is based on a *monotone* and *conservative* numerical algorithm derived from the theory of hyperbolic conservation laws [19, 21, 28]. For a large class of functions β of this type, this numerical scheme automatically obeys the entropy condition, that is, the condition derived from Huygens principle [28]. For velocity functions of the form $\beta = a\kappa + 1$, a combination of both methods is used [20, 21, 25].

Of particular interest for handling hyperbolic systems is the *upwinding* discretization [31]. For example, with a second-order upwinding scheme, a flux term $\frac{\partial}{\partial x} F(u) = f(u)u_x$ is discretized at a grid coordinate x_i as

$$\begin{aligned} \left. \frac{\partial}{\partial x} F(u) \right|_{x_i} &:= f^+ u_x^- + f^- u_x^+, \\ f^+ &:= \max(f(u_i), 0), \\ f^- &:= \min(f(u_i), 0), \\ u_x^- &:= \frac{3u_i - 4u_{i-1} + u_{i-2}}{2\Delta x}, \end{aligned}$$

$$u_x^+ := \frac{-u_{i+2} + 4u_{i+1} - 3u_i}{2\Delta x}.$$

Such an approximation for spatial derivatives enables sharp features in the curve to be preserved while satisfying stability conditions.

It is important to note that the discretization of the evolution equations is performed on a fixed *rectangular grid* [20, 25]. This rectangular grid can be associated with the *pixel* grid of digital images making this discretization method natural for image processing. Since the evolving curve is given by the level set of the function Φ , it is necessary to find this level set (Φ is discrete now). This is done using a very simple contour-finding algorithm described in [20, 25].

Curve Shortening as a Semilinear Diffusion Equation

In this section, stochastic algorithms for implementing curvature-driven flows following [3, 4] are described. In order to formulate a stochastic implementation of the curvature-driven flows described above, it is necessary to reexamine the evolution equations from a different point of view and to modify slightly some previous notation.

As above, let $\mathcal{C} : \mathbb{T} \times (0, T) \rightarrow \mathbb{R}^2$ be a family of plane curves, evolving according to the curve shortening or geometric heat equation [13–15]. As already described, such a family moves by curve shortening

$$\frac{\partial \mathcal{C}}{\partial t} = \frac{\partial^2 \mathcal{C}}{\partial s^2} \quad (23)$$

if and only if its curvature $\kappa(p, t)$ satisfies

$$\frac{\partial \kappa}{\partial t} = \kappa_{ss} + \kappa^3. \quad (24)$$

For a given differentiable curve f in the family \mathcal{C} , the partial derivative with respect to the arc-length variable s is defined by

$$\frac{\partial f}{\partial s} = \frac{1}{g(p, t)} \frac{\partial f}{\partial p}, \quad \text{with } g(p, t) = \left\| \frac{\partial \mathcal{C}}{\partial p} \right\|. \quad (25)$$

And g evolves by

$$\frac{\partial g}{\partial t} = -\kappa^2 g.$$

The length

$$L(t) = \int_C ds = \int_0^1 g(p, t) dp$$

evolves by

$$L'(t) = - \int_C \kappa^2 ds = - \int_0^1 \kappa(p, t)^2 g(p, t) dp. \quad (26)$$

Renormalized arc length is given by

$$\sigma(p, t) = \frac{1}{L(t)} \int_0^p ds = \frac{1}{L(t)} \int_0^p g(p', t) dp' \quad (27)$$

and evolves according to

$$\begin{aligned} \frac{\partial \sigma}{\partial t} &= -\frac{L'}{L} \sigma - \frac{1}{L} \int_0^p \kappa^2 ds \\ &= -\frac{L'}{L} \sigma - \int_0^p \kappa^2 d\sigma \\ &= - \int_0^p \kappa^2 d\sigma + \sigma \int_0^1 \kappa^2 d\sigma. \end{aligned}$$

The curves are parameterized by σ . The curvature, as a function of σ , is denoted by $\hat{\kappa}$:

$$\hat{\kappa}(\sigma(p, t), t) = \kappa(p, t).$$

Some calculus involving the chain rule shows that

$$\frac{\partial \hat{\kappa}}{\partial t} = \frac{\partial \kappa}{\partial t} - \frac{\partial \sigma}{\partial t} \hat{\kappa}_\sigma,$$

and hence,

$$\frac{\partial \hat{\kappa}}{\partial t} = \frac{1}{L^2} \hat{\kappa}_{\sigma\sigma} + \left(\int_0^\sigma \hat{\kappa}^2 d\sigma' - \sigma \int_0^1 \hat{\kappa}^2 d\sigma \right) \hat{\kappa}_\sigma + \hat{\kappa}^3.$$

Multiplying both sides by $L(t)^3$

$$\begin{aligned} L^3 \frac{\partial \hat{\kappa}}{\partial t} &= (L\hat{\kappa})_{\sigma\sigma} + \left(\int_0^\sigma (L\hat{\kappa})^2 d\sigma' - \sigma \int_0^1 (L\hat{\kappa})^2 d\sigma \right) \\ &\quad (L\hat{\kappa})_\sigma + (L\hat{\kappa})^3 \end{aligned}$$

and defining an (unnormalized) density function according to

$$K(\sigma, t) = L(t)\hat{\kappa}(\sigma, t),$$

the following time derivative is obtained:

$$\frac{\partial K}{\partial t} = L \frac{\partial \hat{\kappa}}{\partial t} + L' \hat{\kappa}.$$

By (26),

$$L'(t) = - \int_{p=0}^1 \kappa^2 ds = -L \int_0^1 \hat{\kappa}(\sigma, t)^2 d\sigma,$$

and so,

$$\begin{aligned} L^2 \frac{\partial K}{\partial t} &= L^3 \frac{\partial \hat{\kappa}}{\partial t} + K(\sigma, t) \int_0^1 K(\sigma', t)^2 d\sigma' \\ &= K_{\sigma\sigma} + \left(\int_0^\sigma K^2 d\sigma' - \sigma \int_0^1 K^2 d\sigma' \right) K_\sigma \\ &\quad + K \left(K^2 - \int_0^1 K^2 d\sigma' \right) \\ &= \frac{\partial}{\partial \sigma} \left\{ \frac{\partial K}{\partial \sigma} + \left(\int_0^\sigma K^2 d\sigma' - \sigma \int_0^1 K^2 d\sigma' \right) K \right\}. \end{aligned}$$

Let

$$b_K(\sigma, t) = \int_0^\sigma K^2 d\sigma' - \sigma \int_0^1 K^2 d\sigma'$$

be the *drift coefficient*; this equation can be written as

$$L^2 \frac{\partial K}{\partial t} = K_{\sigma\sigma} + (b_K K)_\sigma.$$

Note that the total curvature of an embedded closed curve is always 2π , so the following conservation law must hold:

$$\oint k ds = \int_0^1 K(\sigma, \tau) d\sigma = 2\pi. \quad (28)$$

This equation can be turned into an autonomous system by introducing a new time variable

$$\tau(t) = \int_0^t \frac{dt'}{L(t')^2}$$

so that $L^2 \frac{\partial}{\partial t} = \frac{\partial}{\partial \tau}$. Note that this has the effect of pushing the singularity to infinity, that is, the equation does not blow up at all. From the results of [5, 14, 15], this property means that there is no blowup as long as $\tau(t)$ is finite.

Thus,

$$\frac{\partial K}{\partial \tau} = K_{\sigma\sigma} + (b_K K)_\sigma$$

$$\frac{dL}{d\tau} = -L \int_0^1 K^2 d\sigma$$

$$\frac{dt}{d\tau} = L^2.$$

A system which admits only nonnegative densities can be obtained by setting $K = \lambda - \mu$ and letting λ and μ evolve by the following system:

$$\frac{\partial \lambda}{\partial \tau} = \lambda_{\sigma\sigma} + (b_{\lambda-\mu} \lambda)_\sigma \quad (29)$$

$$\frac{\partial \mu}{\partial \tau} = \mu_{\sigma\sigma} + (b_{\lambda-\mu} \mu)_\sigma \quad (30)$$

$$\frac{dL}{d\tau} = -L \int_0^1 (\lambda - \mu)^2 d\sigma \quad (31)$$

$$\frac{dt}{d\tau} = L^2 \quad (32)$$

in which

$$\begin{aligned} b_{\lambda-\mu}(\sigma, \tau) &= \int_0^\sigma \{ \lambda(\sigma', \tau) - \mu(\sigma', \tau) \}^2 d\sigma' \\ &\quad - \sigma \int_0^1 \{ \lambda(\sigma', \tau) - \mu(\sigma', \tau) \}^2 d\sigma' \end{aligned} \quad (33)$$

and

$$(b_{\lambda-\mu})_\sigma = (\lambda - \mu)^2 - \int_0^1 (\lambda - \mu)^2 d\sigma \quad (34)$$

are simply the average-free part of $(\lambda - \mu)^2$.

The maximum principle will keep both λ and μ positive as long as $\lambda(\cdot, t_0), \mu(\cdot, t_0) \geq 0$. For any given initial $K(\cdot, t_0)$, a suitable initial partition can be, for example,

$$\begin{aligned} \lambda(\sigma, t_0) &= \max(K(\sigma, t_0), 0), \\ \mu(\sigma, t_0) &= \max(-K(\sigma, t_0), 0) \end{aligned}$$

(which have disjoint support) or

$$\begin{aligned} \mu(\sigma, t_0) &= \mu = -\min_{0 \leq \sigma \leq 1} K(\sigma, t_0), \\ \lambda(\sigma, t_0) &= \mu + K(\sigma, t_0). \end{aligned}$$

Recovering the Curve

Equations (29)–(32) will only determine the curve itself up to rotation and translation. Therefore, given the initial curve \mathcal{C}_0 , it is necessary to couple these equations with the evolution equation

$$\frac{\partial \mathcal{C}}{\partial \tau} = (\lambda - \mu) \mathcal{N}, \quad (35)$$

where \mathcal{N} denotes the (inner) unit normal. Further, it is possible to decompose the evolving curve $\mathcal{C}(\tau)$ into translational $\mathbf{T}(\tau)$ and rotational $\mathbf{R}(\tau)$ components composed with an evolving curve $C(\tau)$ whose evolution involves no overall translation or rotation, that is,

$$\mathcal{C}(\tau) = \mathbf{R}(\tau)C(\tau) + \mathbf{T}(\tau).$$

According to this decomposition, the evolution of \mathcal{C} is given by

$$\frac{\partial \mathcal{C}}{\partial \tau} = \mathbf{R}'C + \mathbf{R} \frac{\partial C}{\partial \tau} + \mathbf{T}' = (\lambda - \mu) \mathcal{N}. \quad (36)$$

Thus, starting from an initial \mathcal{C} (and initial $\mathbf{R} = \mathbf{I}$ and $\mathbf{T} = \mathbf{0}$), it is possible to solve for \mathbf{R}' and \mathbf{T}' using Eq. (36) at two or more points on the curve in order to completely specify the evolution of \mathcal{C} .

Stochastic Implementation

A stochastic implementation can be created by the evolution of a *density* corresponding to Eqs. (29), (30). Accordingly, using parametrization σ , the quantities λ and μ are interpreted as densities.

The approximations used here are based on so-called interacting particle systems as described in [18]. Notice that because of the special parametrization, the diffusion terms of Eqs. (29) and (30) are linear. In this case, there will be two types of particles, one simulating the λ and the other the μ , and the particles interact through the drift rate.

Let $T_N = \mathbb{Z}/N\mathbb{Z}$ denote the discrete torus. The configuration of particles at time τ is given by the pair of functions $(\eta_\tau^\lambda(\cdot), \eta_\tau^\mu(\cdot)) : T_N \rightarrow \mathbb{N}^2$, and the construction is such that $(\eta_\tau^\lambda([\sigma N]), \eta_\tau^\mu([\sigma N]))$ converges to $(\lambda(\sigma, \tau), \mu(\sigma, \tau))$.

Let the *diffusion rates* $g : \mathbb{N} \rightarrow \mathbb{R}_+$ (with $g(0) = 0$) and the *drift rate* $h^\lambda, h^\mu : T_N \times \mathbb{R}_+^{N^2} \rightarrow \mathbb{R}$ be given and define the Markov generator on the particle

configuration $E_N = \mathbb{N}^{T_N} \times \mathbb{N}^{T_N}$ by

$$(\mathcal{L}^N f)(\eta^\lambda, \eta^\mu) = N^2 (\mathcal{L}_0 f)(\eta^\lambda, \eta^\mu) + N (\mathcal{L}_1 f)(\eta^\lambda, \eta^\mu), \quad f \in C_b(E_N),$$

where

$$\begin{aligned} (\mathcal{L}_0 f)(\eta^\lambda, \eta^\mu) &= \frac{1}{2} \sum_{i \in T_N} g(\eta^\lambda(i)) [f(\eta^{i,i+1,\lambda}, \eta^\mu) \\ &\quad + f(\eta^{i,i-1,\lambda}, \eta^\mu) - 2f(\eta^\lambda, \eta^\mu)] \\ &\quad + \frac{1}{2} \sum_{i \in T_N} g(\eta^\mu(i)) [f(\eta^\lambda, \eta^{i,i+1,\mu}) \\ &\quad + f(\eta^\lambda, \eta^{i,i-1,\mu}) - 2f(\eta^\lambda, \eta^\mu)] \end{aligned}$$

and

$$\begin{aligned} (\mathcal{L}_1 f)(\eta^\lambda, \eta^\mu) &= \sum_{i \in T_N} h^\lambda(i, \eta^\lambda, \eta^\mu) \\ &\quad \times \left[f(\eta^{i,i+\text{sgn}(h^\lambda(i, \eta^\lambda, \eta^\mu)), \lambda}, \eta^\mu) - f(\eta^\lambda, \eta^\mu) \right] \\ &\quad + \sum_{i \in T_N} h^\mu(i, \eta^\lambda, \eta^\mu) \left[f(\eta^\lambda, \eta^{i,i+\text{sgn}(h^\mu(i, \eta^\lambda, \eta^\mu)), \mu}) \right. \\ &\quad \left. - f(\eta^\lambda, \eta^\mu) \right], \end{aligned}$$

where, with $\star \in \{\lambda, \mu\}$,

$$\begin{aligned} \eta^{i,i\pm 1,\star}(j) &= \begin{cases} \eta^\star(j) + 1, & j = i \pm 1, \eta^\star(i) \neq 0, \\ \eta^\star(j) - 1, & j = i, \eta^\star(i) \neq 0, \\ \eta^\star(j), & \text{else} \end{cases} \\ \eta^{i,+,\star}(j) &= \begin{cases} \eta^\star(j) + 1, & j = i, \\ \eta^\star(j), & \text{else} \end{cases} \\ \eta^{i,-,\star}(j) &= \begin{cases} \eta^\star(j) - 1, & j = i, \eta^\star(i) > 0, \\ \eta^\star(j), & \text{else} \end{cases}. \end{aligned}$$

Note that \mathcal{L}_0 will be used to approximate the diffusion term of Eqs. (29), (30), while \mathcal{L}_1 will be used to approximate the drift term.

Recall that for a Poisson random variable X of parameter α , one has

$$E(X^2) = \alpha^2 + \alpha, \quad E(X) = \alpha.$$

Thus, referring to Eq. (34), in order to define the rates, set the function $B : E_N \times T_N \rightarrow \mathbb{R}_+$ as

$$\begin{aligned} B(\eta, \eta')(j) &= \frac{1}{N} \sum_{i=1}^j ([\eta(i) - \eta'(i)]^2 \\ &\quad - [\eta(i) + \eta'(i)] - c), \\ c &= \frac{1}{N} \sum_{i=0}^{N-1} ([\eta(i) - \eta'(i)]^2 \\ &\quad - [\eta(i) + \eta'(i)]). \end{aligned} \quad (37)$$

Note that if $\eta(j), \eta'(j)$ are taken as independent Poisson variables of rates $\lambda(j/N, \tau), \mu(j/N, \tau)$, respectively, then

$$E(B(\eta, \eta')(j)) = b_{\lambda(j/N, \tau) - \mu(j/N, \tau)}.$$

Following [3, 18], for (29), the rates can be taken as follows:

$$\begin{aligned} g(k) &= k, \quad h^\lambda(i, \eta, \eta') = -B(\eta, \eta')(i)\eta(i), \\ h^\mu(i, \eta, \eta') &= -B(\eta, \eta')(i)\eta'(i). \end{aligned}$$

Note that it is necessary to take the minus sign in the drift since the equation describes a forward diffusion. The rate for g is, of course, classical, and the drift rate h is similar to [18], when one takes into account the extra averaging due to the nonlocal nature of the function $b_{\lambda-\mu}$.

Conclusions

Geometric-based partial differential equations and curve evolution theory have proven to be powerful tools for a number of key topics in computer vision and image processing. In this note, the theory and practice of two numerical approaches used to implement these flows on a computer have been introduced: the first is based on level sets, and the second is based on a stochastic interpretation of certain equations as zero-range birth-death diffusion processes.

These equations are straightforward to implement in code and extend naturally to the case of Lagrangians driven by image or video data. The ease of implementation coupled with the natural ability to regularize intrinsic geometry and deal with topological changes

has contributed to the popularity of these methods. Some of the key applications include active contours for segmentation and tracking, which may be found in works such as [8, 11, 16].

References

1. Alvarez L, Guichard F, Lions PL, Morel JM (1992) Axiomes et équations fondamentales du traitement d'images. *C R Acad Sci Paris* 315:135–138
2. Alvarez L, Lions PL, Morel JM (1992) Image selective smoothing and edge detection by nonlinear diffusion. *SIAM J Numer Anal* 29:845–866
3. Arous GB, Tannenbaum A, Zeitouni O (2003) Stochastic approximations of curve shortening flows. *J Differ Equ* 195:119–142
4. Angenent S, Tannenbaum A, Yezzi A, Zeitouni O (2006) Curve shortening and interacting particle systems. In: Krim H, Yezzi A (eds) *Statistics and analysis of shapes*. Birkhäuser Boston, Boston, MA, USA, pp 303–313
5. Angenent S (1991) On the formation of singularities in the curve shortening flow. *J Differ Geom* 33: 601–633
6. Arnold VI (1989) Mathematical methods in classical mechanics. Springer, New York
7. Blake A, Yuille A (1992) Active vision. MIT, Cambridge
8. Blake A, Isard M (1998) Active contours. Springer-Verlag New York, Secaucus, NJ, USA
9. Busemann H (1958) Convex surfaces. Interscience, New York, NY, USA
10. Blum H (1973) Biological shape and visual science. *J Theor Biol* 38:205–287
11. Caselles V, Kimmel R, Sapiro G, (1997) Geodesic snakes. *Int J Comput Vis* 22:6179
12. Crandall MG, Ishii H, Lions PL (1992) Users guide to viscosity solutions of second order partial differential equations. *Bull Am Math Soc* 27:1–67
13. Gage M, Hamilton RS, (1986) The heat equation shrinking convex plane curves. *J Differ Geom* 23:69–96
14. Grayson M (1987) The heat equation shrinks embedded plane curves to round points. *J Diffe Geom* 26: 285–314
15. Grayson M, (1989) Shortening embedded curves. *Ann Math* 129:71–111
16. Kichenassamy S, Kumar A, Olver P, Tannenbaum A, Yezzi A, (1996) Conformal curvature flows: from phase transitions to active vision. *Arch Ration Mech Anal* 134:275–301
17. Kimia BB, Tannenbaum A, Zucker SW (1992) On the evolution of curves via a function of curvature, I: the classical case. *J Math Anal Appl* 163:438–458
18. Kipnis C, Landim C (1999) Scaling limits of interacting particle systems. Springer, New York
19. LeVeque RJ, (1992) *Numerical Methods for Conservation Laws*. Birkhäuser, Boston
20. Osher S, Fedkiw R (2003) Level set methods and dynamic implicit surfaces. Springer-Verlag New York, Secaucus, NJ, USA
21. Osher SJ, Sethian JA, (1988) Fronts propagation with curvature dependent speed: algorithms based on hamilton-jacobi formulations. *J Comput Phys* 79:12–49
22. B. ter Haar Romeny (ed) (1994) *Geometry driven diffusion in computer vision*. Kluwer, Holland
23. Sapiro G (2001) *Geometric partial differential equations and image analysis*. Cambridge University Press, Cambridge, UK
24. Sethian JA (1985) Curvature and the evolution of fronts. *Commun Math Phys* 101:487–499
25. Sethian JA (1999) *Level set methods and fast marching methods*, 2nd edn. Cambridge University Press, Cambridge, UK
26. Sethian JA (1982) An analysis of flame propagation. Ph. D. dissertation, University of California
27. Smoller J (1983) *Shock waves and reaction-diffusion equations*. Springer, New York
28. Sod GA (1985) *Numerical methods in fluid dynamics*. Cambridge University Press, Cambridge
29. Terzopoulos D, Szelski R (1992) Tracking with kalman snakes. In: Blake A, Zisserman A (eds) *Active vision*. MIT, Cambridge
30. Ushijima TK, Yazaki S (2000) Convergence of a crystalline algorithm for the motion of a closed convex curve by a power of curvature $V = K^\alpha$. *SIAM J Numer Anal* 37:500–522
31. LeVeque RJ (1992) *Numerical methods for conservation laws*. Birkhäuser, Boston

Numerical Partial Differential Equations

► [Numerical Methods in Curve Evolution Theory](#)

O

Object Class Recognition (Categorization)

Svetlana Lazebnik

Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA

Synonyms

Generic object recognition

Related Concepts

► [Generic Object Recognition](#); ► [Object Detection](#)

Definition

Object class recognition (categorization) is the problem of identifying the class membership (e.g., a car, a dog, a toy) of an object contained in a photograph. Categorization is to be distinguished from the recognition of particular object *instances* (e.g., my Toyota Prius, my dog Fluffy, my teddy bear).

Specific categorization tasks include *image classification* (labeling the image with one of n possible object or scene types), *image annotation* (labeling the image with all objects present), *object detection* or *localization* (finding the location and spatial extent of the object in the image), *image parsing* (segmenting and identifying every object and surface class in the image), and *image understanding* or *interpretation* (describing the content of the image in terms of its

scene type, the objects present and their relationships, and the action taking place).

Background

Object class recognition has been one of the chief intellectual endeavors of computer vision since the inception of the field in the 1960s. This problem is extremely difficult due to the large amount of variability in images from the same category, including viewpoint and lighting change, intra-class shape and texture variation, nonrigid deformations and articulations, as well as clutter and occlusion. The key challenge of achieving human-level recognition capability is designing category representations that are robust to all these sources of variability and yet are discriminative enough to distinguish between members of different categories. Another challenge is the sheer number of visual categories that can be readily recognized by humans – 10^4 – 10^5 , according to estimates made by cognitive scientists. Accurate and efficient recognition at this scale requires representations that take into account the complex visual, semantic, and structural relationships between different categories.

Theory

At present, there is no single definitive theory of visual categorization for either humans or computers. Throughout its almost 50-year history, the field of object recognition has seen a succession of proposed theories and representations focusing on different aspects of intra- and inter-category variation



and reflecting influences from dominant paradigms in closely related fields, including artificial intelligence, cognitive science, and machine learning. The following is a brief and non-exhaustive overview of some of the most influential representational strategies for tackling the categorization problem.

Early in the history of the field, object representations have emphasized shape. The “geometric era” of recognition has been dominated by 3D shape models composed of geometric primitives such as polyhedra and generalized cylinders [14]. In cognitive science, this approach was supported by the theory of recognition by components [1]. The main advantage of 3D geometric models is that they permit a principled treatment of viewpoint change. They are also expressive – most familiar shapes, at a suitable level of abstraction, can be represented with the help of a few simple geometric part types. Geometric methods have shown some success in recognizing instances of texture-free man-made objects based on edges extracted from relatively uncluttered images. However, they ultimately could not overcome several critical limitations, including the need to manually create 3D object models, the lack of good representations for intra-class shape variability, and the brittleness of model-to-image alignment in the presence of complex appearance changes due to texture, reflectance, lighting, and background clutter.

In the mid-1990s, the geometric recognition paradigm has given way to a radically different philosophy of *appearance-based methods*. The idea of such methods, e.g., *appearance manifolds* [15], is to essentially represent each image by the 1D vector of its pixel values to automatically acquire or *learn* a model of the category by sampling the “manifold” of all possible images from that category under all relevant sources of variability, and to perform recognition by computing the distances between a test image and the learned manifolds. The advantage of this strategy lies in its simplicity and its emphasis on empirical or statistical modeling of pixel-level appearance phenomena that were almost completely ignored by the geometric methods. Its drawbacks include a lack of invariance to any conditions not explicitly sampled at training time (e.g., clutter and occlusion) and a near-lack of abstraction from the image.

The above shortcomings were alleviated with the advent of *local* appearance-based models. The basic

building blocks of such models are appearance descriptors computed in the neighborhood of *local features* that are invariant to scaling, rotation, or affine deformations of the image [19]. In addition to having good repeatability and invariance properties, the small spatial extent of local features allows them to survive changes that affect significant parts of the image, such as clutter and occlusion. Even though local invariant features were initially developed for wide-baseline matching and object instance recognition, their usefulness for categorization became apparent very quickly. An important class of local appearance-based models are *bags of features* [2], which work by quantizing feature descriptors to discrete *visual words* and representing images by their visual word histograms. When combined with modern machine learning methods, bags of features can give very good performance on image classification and annotation tasks. However, because they discard all spatial information and do not attempt to associate individual features with objects or background, they are not suitable for localization.

Local features have also been successfully used to construct part-based object models. For example, *constellation models* [20] capture both the appearance and the spatial configuration of object features or parts in a generative Bayesian framework. The probability distributions over part descriptors and locations can be learned in a *weakly supervised* fashion from images where the object of interest is known to be present, but it is not known which features belong to the object. In the formulation of [20], the joint distribution of all part locations is assumed to be Gaussian, which effectively restricts the models to single aspects of rigid objects (e.g., side views of cars). In addition, the computational demands of Bayesian learning and inference in the weakly supervised setting limit the number of parts and the amount of clutter that can be tolerated.

A related important class of part-based object models are *pictorial structures* [8]. Similarly to constellation models, pictorial structures are based on a Bayesian formalism where the probability of an object configuration given an image is a product of individual part likelihoods and a prior over the part locations. Unlike constellation models, pictorial structures constrain the spatial dependencies between parts to form a tree, enabling efficient algorithms for finding the globally optimal match of a pictorial structure

model to an image. The tree structure of these models makes them especially suitable for estimating part layouts for articulated objects such as humans and animals.

When a localization task calls only for an estimate of the object's bounding box, it is common to use a *sliding window detector*. Discriminatively trained detectors using powerful features such as *histograms of oriented gradients* (HOG) [3] work well for certain classes, including faces, cars, and pedestrians. Successful detection of a wider range of highly variable, deformable classes requires a more flexible part-based representation that is yet efficient enough for sliding window search. A promising development in this direction is given by discriminatively trained hierarchical part-based models [9]. These models consist of HOG "filters" at two different levels of resolution for the whole object and its parts, and they support efficient learning and inference in the framework of *latent support vector machines*.

The methods discussed above may be capable of finding part layouts or bounding boxes, but they do not precisely segment out the recognized objects. Historically, it has been hoped that object regions could be accurately segmented by bottom-up *perceptual organization* techniques and then passed along to a recognition module. However, the hope for high-quality bottom-up segmentation has not yet been realized, and today, an emerging consensus in the recognition community holds that segmentation and categorization processes should feed into each other to jointly determine object identity and spatial extent. For example, in the *implicit shape model* approach [12], feature matches "vote" for object hypotheses using a generalized Hough transform, and the winning hypothesis is used to infer a top-down probabilistic segmentation. In turn, the segmentation helps to verify the object hypothesis by discarding the influence of background features.

A different line of attack is required for the problem of image parsing, which involves labeling every pixel in the image by its object class. In this setting, *conditional random fields* provide an appropriate framework for combining local evidence of the presence of multiple object classes with information about their contextual interaction (see, e.g., [17]). In general, the subject of context in recognition is becoming more and more important today as the community moves

towards interpreting complex scenes consisting of multiple object types [10].

By embracing statistical appearance-based modeling of intra-class variation, recognition has made great strides over the last two decades. Unfortunately, the original emphasis on representing the 3D nature of real object categories has been largely abandoned in the process. Recently, some researchers have begun an important effort to infuse explicit modeling of viewpoint relationships back into statistical category models [16].

Another topic of much current interest is constructing efficient recognition architectures for large numbers of categories. To date, the standard approach to many-class problems is still to learn separate models for all classes of interest and then to apply each model to the test image. This approach is unsatisfactory because its testing time scales linearly with the number of categories and because its model structure ignores the semantic and visual relationships between different categories. To make problems with many classes more scalable, one potential strategy is to force related classes to share features [18], and another is to organize object models into semantic or visual hierarchies (see, e.g., [13]). Yet another possible way of sharing information between related category models is through attribute-centric representations [6]. An important advantage of such representations is in allowing the recognition system to make useful inferences about previously unseen categories. For example, the system may not have been trained to recognize a tapir in a picture, but it may still be able to infer that it is a four-legged herbivorous mammal.

Application

Automated recognition systems have the potential to make a significant impact in many applications, including search and indexing of large-scale image and video collections, information access through images, human-computer interaction, environmental monitoring, documentation and preservation of natural diversity and cultural heritage, assistance to the visually impaired, personal robotics, automated driving, surveillance and security, and many others.



Open Problems

Despite the rapid progress of the field, designing recognition systems that approach human-level performance on $10^4\text{--}10^5$ categories is still an open problem. While the biological mechanisms of recognition themselves remain poorly understood, whatever representations the human brain uses give rise to some remarkable abilities, including the ability to infer the properties of never before seen categories and to learn their models from as little as one image, or the ability to construct narratives about the content of an image. Designing computer systems capable of such behavior remains a wide-open goal.

Experimental Results

Until the late 1990s, due to the limitations of digital imaging and computer technology, the evaluation of object recognition methods has tended to be mostly small-scale and anecdotal. In the last few years, the field has established a number of benchmark datasets that enable systematic measurement of progress and comparative evaluation of different algorithms. One of the earliest many-category benchmarks was Caltech-101 [7]. Despite having some widely acknowledged flaws (limited intra-class variation, centered objects, absence of clutter), this dataset had by far the largest *inter-class* diversity at the time of its release, and it has helped to catalyze much research into category-level recognition. More recent datasets include Caltech-256 [11], the PASCAL Visual Object Classes Challenge [5], and ImageNet [4]. In particular, ImageNet contains over 11 million images and over 15,000 concepts, which makes it a good platform for developing and testing new categorization algorithms.

References

1. Biederman I (1987) Recognition by components—a theory of human image understanding. *Psychol Rev* 94:115–147
2. Csurka G, Dance C, Fan L, Willamowski J, Bray C (2004) Visual categorization with bags of keypoints. In: ECCV international workshop on statistical learning in computer vision, Prague
3. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. *IEEE Conf Comput Vis Pattern Recognit (CVPR)* 1:886–893
4. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) ImageNet: a Large-Scale Hierarchical Image Database. In: IEEE conference on computer vision and pattern recognition (CVPR), Miami, pp 248–255
5. Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A (2010) The Pascal visual object classes (VOC) challenge. *Int J Comput Vis* 88(2):303–338. <http://pascallin.ecs.soton.ac.uk/challenges/VOC/>
6. Farhadi A, Endres I, Hoiem D, Forsyth D (2009) Describing objects by their attributes. In: IEEE conference on computer vision and pattern recognition (CVPR), Miami, pp 1778–1785
7. Fei-Fei L, Fergus R, Perona P (2004) Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In: IEEE computer vision pattern recognition (CVPR) workshop on generative-model based vision. http://www.vision.caltech.edu/Image_Datasets/Caltech101
8. Felzenszwalb P, Huttenlocher D (2000) Efficient matching of pictorial structures. *IEEE Conf Comput Vis Pattern Recognit (CVPR)* 2:66–73
9. Felzenszwalb P, Girshick R, McAllester D, Ramanan D (2010) Object detection with discriminatively trained part based models. *IEEE Trans Pattern Anal Mach Intell* 32(9):1627–1645
10. Galleguillos C, Belongie S (2010) Context based object categorization: a critical survey. *Comput Vis Image Underst* 114:712–722
11. Griffin G, Holub AD, Perona P (2006) The Caltech-256. Caltech technical report. http://www.vision.caltech.edu/Image_Datasets/Caltech256/
12. Leibe B, Leonardis A, Schiele B (2008) Robust object detection with interleaved categorization and segmentation. *Int J Comput Vis* 77(1–3):259–289
13. Marszałek M, Schmid C (2008) Constructing category hierarchies for visual recognition. In: European conference on computer vision (ECCV), Marseille, 479–491
14. Mundy J (2007) Object recognition in the geometric era: a retrospective. In: Ponce J, Hebert M, Shmid C, Zisserman A (eds) *Toward category-level object recognition*. Lecture notes in computer science. Springer, Berlin/New York, pp 3–29
15. Murase H, Nayar S (1995) Visual learning and recognition of 3d objects from appearance. *Int J Comput Vis* 14(1):5–24
16. Savarese S, Fei-Fei L (2010) Multi-view object categorization and pose estimation. *Studies in computational intelligence: computer vision*, vol 285/2010. Springer, Berlin/New York, pp 205–231
17. Shotton J, Winn J, Rother C, Criminisi A (2009) Texton-Boost for image understanding: multi-class object recognition and segmentation by jointly modeling appearance, shape and context. *Int J Comput Vis* 81(1):2–23
18. Torralba A, Murphy KP, Freeman WT (2007) Sharing visual features for multiclass and multiview object detection. *IEEE Trans Pattern Anal Mach Intell* 29(5):854–869
19. Tuytelaars T, Mikolajczyk K (2008) Local invariant feature detectors: a survey. *Found Trends Comput Graph Vis* 3: 177–820
20. Weber M, Welling M, Perona P (2000) Unsupervised learning of models for recognition. In: European conference on computer vision (ECCV), Dublin, pp 18–32



Object Detection

Yali Amit¹ and Pedro Felzenszwalb²

¹Department of Computer Science, University of Chicago, Chicago, IL, USA

²School of Engineering, Brown University, Providence, RI, USA

Definition

Object detection involves detecting instances of objects from a particular class in an image.

Background

The goal of object detection is to detect all instances of objects from a known class, such as people, cars, or faces in an image. Typically, only a small number of instances of the object are present in the image, but there is a very large number of possible locations and scales at which they can occur and that need to somehow be explored.

Each detection is reported with some form of *pose* information. This could be as simple as the location of the object, a location and scale, or the extent of the object defined in terms of a bounding box. In other situations, the pose information is more detailed and contains the parameters of a linear or nonlinear transformation. For example, a face detector may compute the locations of the eyes, nose, and mouth, in addition to the bounding box of the face. An example of a bicycle detection that specifies the locations of certain parts is shown in Fig. 1. The pose could also be defined by a three-dimensional transformation specifying the location of the object relative to the camera.

Object detection systems construct a model for an object class from a set of training examples. In the case of a fixed rigid object, only one example may be needed, but more generally multiple training examples are necessary to capture certain aspects of class variability.

Object detection methods fall into two major categories, *generative* [1–5] and *discriminative* [6–10]. The first consists of a probability model for the pose variability of the objects together with an appearance model: a probability model for the image appearance

conditional on a given pose, together with a model for background, that is, nonobject images. The model parameters can be estimated from training data and the decisions are based on ratios of posterior probabilities. The second typically builds a classifier that can discriminate between images (or sub-images) containing the object and those not containing the object. The parameters of the classifier are selected to minimize mistakes on the training data, often with a regularization bias to avoid overfitting.

Other distinctions among detection algorithms have to do with the *computational* tools used to scan the entire image or search over possible poses, the type of *image representation* with which the models are constructed, and what type and how much training data is required to build a model.

Theory

Images of objects from a particular class are highly variable. One source of variation is the actual imaging process. Changes in illumination and changes in camera position, as well as digitization artifacts, all produce significant variations in image appearance, even in a static scene. The second source of variation is due to the intrinsic appearance variability of objects within a class, even assuming no variation in the imaging process. For example, people have different shapes and wear a variety of clothes, while the handwritten digit seven can be written with or without a line through the middle, with different slants, stroke widths, etc. The challenge is to develop detection algorithms that are *invariant* with respect to these variations and are computationally efficient.

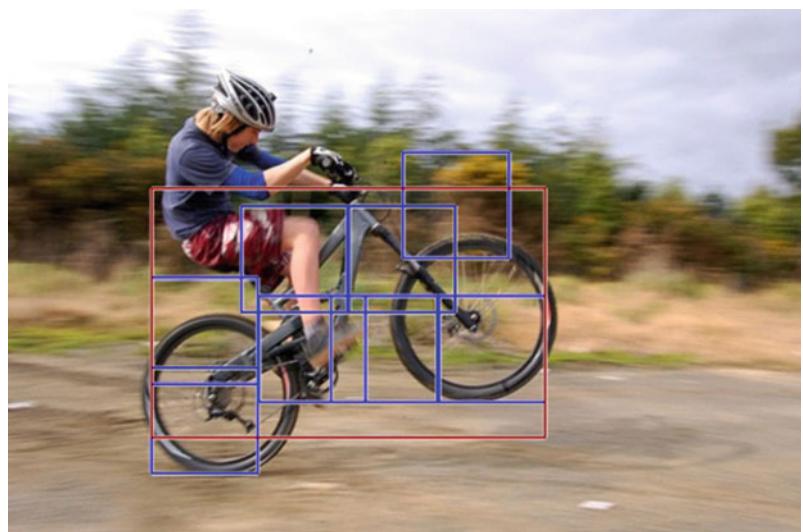
Invariance

The brute force (naive) approach to invariance assumes training data is plentiful and represents the entire range of object variability. Invariance is implicitly learned from the data while training the models.

When training data is limited, it is necessary to build invariance into the models. There are two complementary methods to achieve this. One involves computing invariant functions and features; the other involves searching over latent variables. Most algorithms contain a combination of these approaches. For example, many algorithms choose to apply local transformations



Object Detection, Fig. 1 A bicycle detection specified in terms of the locations of certain parts



to pixel intensities in such a way that the transformed values are invariant to a range of illumination conditions and small geometric variations. These local transformations lead to *features* and the array of feature values is the *feature map*. More significant transformations are often handled through explicit search over latent variables or by learning the remaining variability from training data.

Invariant Functions and Features

This method constructs functions of the data that are invariant with respect to the types of variability described above and can still distinguish between object and background images. This may prove difficult if object variability is extensive. Invariant functions that produce the same output no matter the pose and appearance of the object necessarily have less discriminative power.

There are two common types of operations leading to invariant functions. The first involves computing local features that are invariant to certain image transformations. The second operation involves computing geometric quantities that are invariant to some or all three-dimensional pose variation. For example, the cross ratio among distinguished points is a projective invariant that has been used to recognize rigid objects.

An example of a local feature, invariant to certain photometric variations and changes in illumination, is the *direction* of the image gradient, from which a variety of *edge* features can be computed. More complex features capture the appearance of small image

patches and are often computed from edge features. An example would be the histogram of gradient (HOG) features [9]. These features are usually computed at a *dense* grid of locations in the image, leading to a dense feature map.

Local pooling of features is commonly used to introduce some degree of invariance to small geometric variations. A typical example is the MAX operation [1, 11]. In this case, a quantity that is to be computed at a pixel is replaced by the maximum of the quantity in a neighborhood of the pixel. When the maximum is extended over the entire window, the result is a *bag of features* model [12], which reduces to counting the number of binary features of different types that occur within a window. In this case, all spatial information is lost, leading to models that are invariant to fairly large geometric transformations.

For computational reasons, it is often useful to sparsify the feature map by applying local decisions to find a small set of *interest points*. The assumption is that only certain features are useful (or necessary) for object detection. The approach yields *sparse* feature maps that can be processed much more efficiently. Examples of commonly used sparse features are SIFT descriptors [13], corner detectors, and edge conjunctions [1]. One drawback of sparse features is that hard decisions are being made on their presence, and if some are missed, an algorithm may fail to detect an instance of the object.

Note that it is possible to predefined a very large family of features that is never fully computed, rather,

in training an informative subset is selected that can produce the required classification for a particular object class. One example are the Haar features that compute differences of intensity averages in adjacent rectangles of varying sizes and locations [7]. Another example are geometric edge arrangements of increasing complexity.

Latent Variables

An explicit parameterization of the variability can be defined via *latent* variables that are not directly observable from the image data. These are not necessarily needed for the final report on the object detections, but their values simplify the solution of the detection problem. For example, to detect faces at a range of orientations, at each candidate region, one could decide, for *each* possible orientation, whether or not the region contains a face at that orientation. In general a set Θ defines latent parameters that could capture global illumination parameters, a linear or nonlinear map from a model domain into the image domain, or specify the locations of a finite set of object parts. The last case is common in part-based models where latent part placements are used to decide if the object might be present at a particular location in the image [10]. The set of possible latent values, Θ , can be quite large or infinite. This leads to computational challenges that have been addressed by a variety of methods including coarse-to-fine computation, dynamic programming, and geometric alignment.

Detection via Classification

Both generative and discriminative models start with an initial choice of image features and with a choice of the latent pose parameters that will be explicitly modeled. The primary differences between generative and discriminative models are in the methods of training and computation. One important distinction is that generative models do not need data from background to train the object model, whereas discriminative methods need data from both classes to learn the decision boundaries.

The most common approach to object detection reduces the problem to one of binary classification. Consider the problem of detecting objects of fixed size but varying positions in the image. Let W denote a reference window size that an instance of the object would occupy. Let L denote a grid of locations in

the image. Let X_{s+W} denote the image features in a window (sub-image) with top-left corner at $s \in L$. One can reduce the detection problem to a binary classification problem as follows. For each location $s \in L$ classify X_{s+W} into two possible classes corresponding to windows that contain an object and windows that do not contain an object. The sliding-window approach to object detection involves explicitly considering and classifying every possible window. Note that the same approach can be used to detect objects of different sizes by considering different window sizes or alternatively windows of fixed size at different levels of resolutions in an image pyramid.

Generative Models

A general framework for object detection using generative models involves modeling two distributions. A distribution $p(\theta; \eta_p)$ is defined on the possible latent pose parameters $\theta \in \Theta$. This distribution captures assumptions on which poses are more or less likely. An appearance model is defined describing the distribution of the image features in a window *conditional* on the pose, $p(X_{s+W}|object, \theta; \eta_a)$. Here, η_p and η_a are the model parameters. For example, η_a might define a *template* specifying the probability of observing certain features at each location in the detection window under a canonical choice for the object pose, while θ specifies a transformation of the template. Warping the template according to θ leads to probabilities for observing certain features at each location in X_{s+W} conditioned on this particular choice of pose parameters [1, 2, 4].

Training data with images of the object are used to estimate the parameters η_p and η_a . Note that the images do not normally come with information about the latent pose variables θ , unless annotation is provided. Estimation thus requires inference methods that handle unobserved variables, for example, the different variants of the expectation maximization algorithm [3, 4]. In some cases, a probability model for background images is estimated as well using large numbers of training examples of images not containing the object.

The basic detection algorithm then scans each candidate window in the image, computes the most likely pose under the object model, and obtains the “posterior odds,” that is, the ratio between the conditional probability of the window under the object hypothesis at the optimal pose and the conditional probability of the



window under the background hypothesis. This ratio is then compared to a threshold τ to decide if the window contains an instance of the object

$$\frac{p(X_{s+W}|\text{object}, \theta; \eta_a) p(\theta; \eta_p)}{p(X_{s+W}|\text{background})} > \tau.$$

When no background model has been trained offline, a simple *adaptive* background model can be estimated online for each window being tested. In this case, no background training data is needed [4]. Alternative background models involve subcollections of parts of the object model [14].

Discriminative Models

If no explicit latent pose variables are used, the underlying assumption is that the training data is sufficiently rich to provide a sample of the entire variation of object appearance. The discriminative approach trains a standard two-class classifier using large amounts of data from the object and background classes. Many classifier types have been used, including neural networks, SVMs, boosted decision trees, and radial basis functions.

Cascades

Because of the large size of the background population and its complexity, discriminative methods are often organized in *cascades* [7]. An initial classifier is trained to distinguish between the object and a manageable amount of background data. The classifier is designed to have no false negatives at the price of a larger number of false positives. Then a large number of background examples are evaluated and the misclassified ones are collected to form a new background data set. Once a sufficient number of such false positives is accumulated, a new classifier is trained to discriminate between the original object data and the new “harder” background data. Again this classifier is designed to have no false negatives. This process can be continued several times.

At detection time, the classifiers in the cascade are applied sequentially. Once a window is classified as background, the testing terminates with the background label. If the object label is chosen, the next classifier in the cascade is applied. Only windows that are classified as object by all classifiers in the cascade are labelled as object by the cascade. Note that having

no false negatives during training provides no guarantee that in testing instances of the objects won’t be missed.

Pose Variables

Certain discriminative models can also be implemented with latent pose parameters [10]. Assume a generic classifier defined in terms of a space of classifier functions $f(x; u)$ parameterized by u . Usual training of a discriminative model consists of solving an equation of the form

$$\min_u \sum_{i=1}^n D(y_i, f(x_i; u)) + C(u),$$

for some regularization term $C(u)$ which prevents overfitting and a loss function D measuring the distance between the classifier output $f(x_i; u)$ and the ground truth label $y_i = 1$ for object and $y_i = 0$ for background.

The minimization above can be replaced by

$$\begin{aligned} \min_u \sum_{y_i=1} \min_{\theta \in \Theta} D(1, f(\theta(x_i); u)) \\ + \sum_{y_i=0} \max_{\theta \in \Theta} D(0, f(\theta(x_i); u)) + C(u). \end{aligned}$$

Here $\theta(x)$ defines a transformation of the example x . Intuitively for a positive example, one would like there to be some transformation under which x_i is classified as object, while for a negative example, one would like it to be the case that there is no transformation under which x_i is classified as object.

Computational Methods

The basic detection process consists of scanning the image lattice and at each location s testing whether X_{s+W} is classified as object or background. This is typically done at multiple resolutions of the image pyramid to detect objects at multiple scales and is clearly a very intensive computation. There are a number of methods to make it more efficient.

Sparse Features

When sparse features are used, it is possible to focus the computation only in regions around features. The two main approaches that take advantage of this sparsity are *alignment* [15] and the *generalized Hough*

transform. Alignment uses information regarding the relative locations of the features on the object. In this case, the locations of some features determine the possible locations of the other features. Various search methods enable a quick decision on whether a sufficient number of features were found to declare object or not. The Hough transform typically uses information on the location of each feature type relative to some reference point in the object. Each detected feature votes with some weight for a set of candidate locations of the reference point. Locations with a sufficiently large sum of weighted votes determine detections. This idea can also be generalized to include identification of scale as well. The voting weights can be obtained either through discriminative training or through generative training [1].

Cascades

As mentioned above, the cascade method trains a sequence of classifiers with successively more difficult background data. Each such classifier is designed to be very computationally efficient. When the data in the window X_{s+w} is declared background by any classifier of the cascade, the decision is final and the computation proceeds to the next window. Since most background windows are rejected early in the cascade, most of the windows in the image are processed very quickly.

Coarse to Fine

The cascade method can be viewed as a coarse-to-fine decomposition of background that gradually makes finer and finer discriminations between object and background images that have significant resemblance to the object. An alternative is to create a coarse-to-fine decomposition of object poses [8]. In this case, it is possible to train classifiers that can rule out a large subset of the pose space in a single step. A general setting involves a rooted tree where the leaves correspond to individual detections and internal nodes store classifiers that quickly rule out all detections below a particular node. The idea is closely related to branch-and-bound methods [12] that use admissible lower bounds to search a space of transformations or hypotheses.

Dynamic Programming

There are a number of object detection algorithms that represent objects by a collection of parts arranged in deformable configurations or as hierarchies of such arrangements of parts of increasing complexity. When the hierarchies and the arrangements have the appropriate structure, dynamic programming methods can be used to efficiently search over the spaces of arrangements [1, 2].



Object Detection, Fig. 2

The output of a face detection algorithm



Application

Object detection methods have a wide range of applications in a variety of areas including robotics, medical image analysis, surveillance, and human-computer interaction. Current methods work reasonably well in constrained domains but are quite sensitive to clutter and occlusion.

A popular benchmark for object detection is the PASCAL VOC object detection challenge. The goal of the challenge is to detect objects from common categories such as people, cars, horses, and tables in photographs. The challenge has attracted significant attention in the computer vision community over the last few years, and the performance of the best systems has been steadily increasing by a significant amount on a yearly basis.

Face detection is a typical application of object detection algorithms. There has been significant success in deploying face detection methods in practical situations. For example, current digital cameras use face detection to decide where to focus and even detect smiles to decide when to take the picture. Figure 2 shows a typical output of a face detection algorithm.

References

1. Amit Y (2002) 2d object detection and recognition: models, algorithms and networks. MIT Press, Cambridge
2. Felzenszwalb P, Huttenlocher D (2005) Pictorial structures for object recognition. *Int J Comput Vis* 61(1):55–79
3. Fergus R, Perona P, Zisserman A (2003) Object class recognition by unsupervised scale-invariant learning. In: IEEE conference on computer vision pattern recognition (CVPR 2003), Madison
4. Amit Y, Trouvé A (2007) POP: patchwork of parts models for object recognition. *Int J Comput Vis* 75(2):267–282
5. Jin Y, Geman S (2006) Context and hierarchy in a probabilistic image model. In: IEEE conference on computer vision pattern recognition (CVPR 2006), New York
6. Rowley HA, Baluja S, Kanade T (1998) Neural network-based face detection. *IEEE Trans Pattern Anal Mach Intell* 20(1):23–38
7. Viola P, Jones MJ (2004) Robust real time face detection. *Int J Comput Vis* 57(2):137–154
8. Fleuret F, Geman D (2001) Coarse-to-fine face detection. *Int J Comput Vis* 41(1–2):85–107
9. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE conference on computer vision pattern recognition (CVPR 2005), San Diego
10. Felzenszwalb P, Girshick R, McAllester D, Ramanan D (2010) Object detection with discriminatively trained part based models. *IEEE Trans Pattern Anal Mach Intell* 32(9):1627–1645
11. Riesenhuber M, Poggio T (2000) Models of object recognition. *Nat Neurosci* 3:1199–1204. Supplement
12. Lampert C, Blaschko M, Hofmann T (2009) Efficient subwindow search: a branch and bound framework for object localization. *IEEE Trans Pattern Anal Mach Intell* 31(12):2129–2142
13. Lowe D (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110
14. Chang LB, Jin Y, Zhang W, Borenstein E, Geman S (2011) Context computation, and optimal roc performance in hierarchical models. *Int J Comput Vis* 93:117–140
15. Ullman S (1996) High-level vision. MIT Press, Cambridge

Object Extraction

► [Interactive Segmentation](#)

Object Models

► [Model-Based Object Recognition](#)

Object Motion Blur

► [Motion Blur](#)

Object Parameterizations

► [Model-Based Object Recognition](#)

Object Recognition

► [Line Drawing Labeling](#)

Object Representations

► [Model-Based Object Recognition](#)

Object Segmentation

► [Semantic Image Segmentation](#)



Obstacle Detection

Larry Matthies

Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA

Synonyms

Hazard detection

Definition

Obstacle detection is the process of using sensors, data structures, and algorithms to detect objects or terrain types that impede motion.

Background

Obstacle detection is applicable to anything that moves, including robot manipulators and manned or unmanned vehicles for land, sea, air, and space; for brevity, these are all called *vehicles* here. Obstacle detection and hazard detection are synonymous terms, but are sometimes applied in different domains; for example, obstacle detection is usually applied to ground vehicle navigation, whereas hazard detection is often applied to aircraft or spacecraft in the process of landing, as in “landing hazard detection.” Obstacle detection is a system problem that encompasses sensors that perceive the world, world models that represent the sensor data in a convenient form, mathematical models of the interaction between objects and the vehicle, and algorithms that process all of this to infer obstacle locations. Obstacle detection algorithms use the world model and the interaction model to locate obstacles, to characterize the degree to which the obstacles impede motion, and to produce an *obstacle map* for use by path planning algorithms. The complexity of obstacle detection systems varies greatly, depending on the domain of operation, the size and cost of the vehicle, and the degree of reliability required.

Because sensor data is noisy and incomplete, often it is necessary to combine many sensor measurements into the world model to reduce noise and fill

in gaps in the world model before applying obstacle detection algorithms; this makes obstacle detection related to mapping. Obstacles may completely block motion of the vehicle or just make it more difficult to move, such as having to move slowly over rough ground; this makes obstacle detection related to terrain classification.

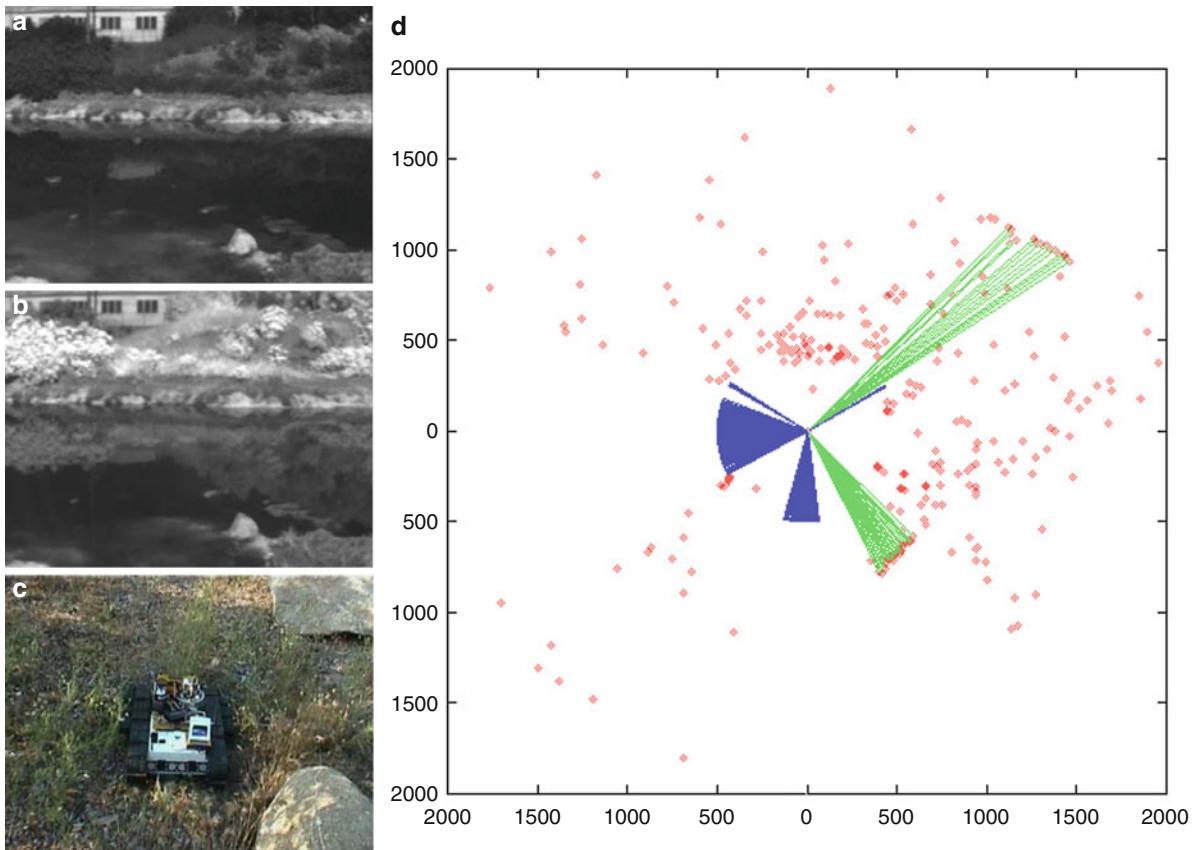
Sensors for Obstacle Detection

An excellent introduction to a broad range of sensors for obstacle detection is included in [1]. 3-D perception plays a central role in obstacle detection, because the geometry of the world is a major factor in determining what constitutes an obstacle. 3-D sensors include active ranging techniques, like lidar, radar, and sonar, and passive ranging techniques, like stereo vision and structure from motion. Appearance can also be useful for discriminating obstacles from non-obstacles, so color, texture, polarization, and radiant temperature can also be important sensor attributes. Since all 3-D sensors have limited range, appearance attributes are especially important for detecting obstacles at long range.

Often, mechanical properties like the stiffness of objects or terrain are important to obstacle detection, as well as geometric properties. This requires sensors that directly or indirectly give insight into mechanical properties of the scene. These can be proprioceptive sensors, like inertial sensors, wheel encoders, or instrumented bumpers, that sense aspects of the vehicle-world interaction as it occurs, or remote sensors that perceive the terrain ahead of the vehicle. Examples of remote sensing to predict terrain mechanical properties are using lidar [2] or multispectral imagery [3] to infer whether material is vegetation that can be pushed through by the vehicle or is something more solid that must be avoided (Fig. 1).

Environmental conditions like ambient illumination and atmospheric obscuration also impact obstacle detection sensors. For example, fog, smoke, and night operation may require active sensors or thermal infrared cameras (Fig. 2), while thick dust or heavy precipitation may require radar. Dynamic environments require sensors that measure the velocity of obstacles as well as their size and location.

Choosing appropriate sensors is a critical first step in obstacle detection, to ensure that the sensors can



Obstacle Detection, Fig. 1 Sensors for perceiving vegetation. (a) Image acquired with narrow spectral band around 650 nm, with a pond in the foreground and soil and vegetation in the background. The foliage is *darker* than the soil in this spectral band. (b) Image of the same scene acquired with a narrow spectral band around 800 nm. The foliage is *brighter* than the soil in this spectral band. Ratios of spectral bands at 650 and 800 nm can be used to distinguish growing foliage more effectively than

can be done with just visible spectrum imagery. (c) A small robot in sparse grass, with a few rocks nearby, carrying a single-axis, 360° scanning laser range finder (lidar). (d) Plot of lidar pixels (*red dots*), with sensor at (0,0). *Blue cones* are areas that were blocked by structure on the robot. Pixels that appear to be randomly distributed are measurements of grass; pixels that fall into line segments are measurements of the rocks. The *green rays* are rock pixels that were automatically detected

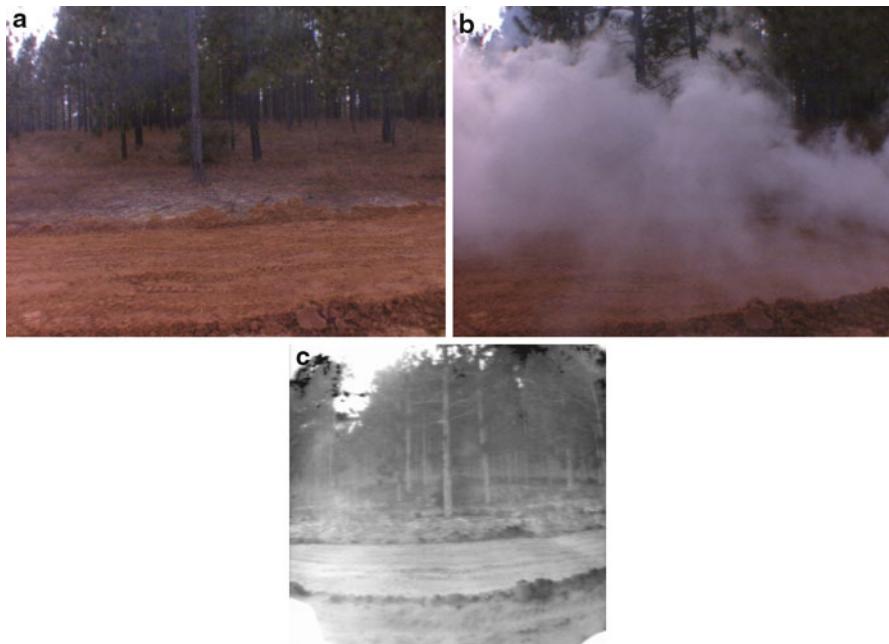
discriminate obstacles under the required conditions. This can be difficult; for example, operation in high levels of airborne dust requires use of sensors that measure range to solid objects behind the dust, instead of to the dust. It can also be very challenging to obtain appropriate sensors that fit within the size, weight, power, and cost constraints of the application.

Data Structures for Obstacle Detection

Obstacle detection algorithms can be developed to process raw sensor data or data that has been transformed into another representation. Raw data is often expressed in *sensor space* (also called *image space*

for some sensors); these are data structures that are indexed by the natural coordinate system of the sensor, such as (row, column) or (azimuth, elevation). With two-axis lidar, for example, surface normals, range discontinuities, and height above ground can be computed directly from range images provided by the sensor, and obstacle locations can be inferred from these quantities. Operating in image space preserves pixel adjacency information, which can be convenient for segmentation.

The normal alternative to working in image space is to work in a *map space*, such as a Cartesian reference frame that is either centered on the vehicle or has a fixed origin in the world. Map representations can simply accumulate raw data from many sensors or



Obstacle Detection, Fig. 2 Ability to see through smoke with thermal infrared imagery. (a) Scene viewed with a color camera. (b) Same scene with smoke. (c) Same scene viewed with

the smoke, using a mid-wave infrared camera, which is sensitive to light from 3 to 5 μm ; the sensor sees quite well through the smoke

many points in time, such as point clouds from range-imaging sensors like lidar or stereo vision, or they may be gridded or geometric primitives [4]. Elevation maps and occupancy grids are common gridded representations. Elevation maps record the height of the ground surface in each cell of a discretized map of the ground plane. Occupancy grids record a probability that each cell of the map is occupied by solid material, instead of empty; these can be defined in 2-D or 3-D [5, 6]. Geometric representations fit geometric primitives to the raw sensor data, like line segments or polygons for 2-D maps and planar surfaces or bounding boxes for 3-D maps.

Map space representations are used for several purposes. First, they are used to accumulate data over time, which is valuable to “fill in” gaps that may occur in individual frames of sensor data. Second, they facilitate noise reduction, because the accumulated data can be averaged or filtered for outliers. Third, they can effectively interpolate higher-resolution representations of the world than are available from individual frames of sensor data; an example is when occupancy grids are used to build world models from wide-angle sonar data. Finally, they can be in a coordinate system

that makes some aspects of obstacle detection easier; for example, transforming range data from image space to a Cartesian map space data structure makes it possible to use shift-invariant algorithms to detect fixed-size obstacles.

Uncertainty is inherent in sensors, therefore also in maps and in obstacle detections, so modeling that uncertainty is an important issue. One of the most common approaches to this is the use of occupancy grids to estimate the probability that map cells contain obstacles [5, 6]. Uncertainty also can drive the choice of world model representation, such as the use of polar-perspective maps, which use an inverse-range parameterization of the map to match the spatial and range resolution characteristics of stereo vision [7].

The output of the obstacle detection process is also represented in some type of world model. Some obstacle models identify only certain locations as obstacles; these can be a subset of the cells of a grid map, polygons in the ground plane, bounding boxes in 3-D, or other representations. For off-road navigation, it is common for obstacle detection to produce a *cost map*, where every cell of the map gets an associated traversal cost [8]. Especially for off-road navigation, obstacle



detection overlaps the problem of trafficability analysis, where the vehicle can pass over a given patch of terrain, but may sink, slip, or experience a rough ride in the process; in this case, the output may represent aspects of trafficability that are useful for path planning [9].

Vehicle-World Interaction Models for Obstacle Detection

Obstacle detection algorithms implicitly or explicitly use a model of how the vehicle interacts with the world to determine where obstacles exist. In the simplest case, the model may be just a binary decision based on raw sensor data, such as when single-axis scanning lidars are used for obstacle detection indoors and any object that produces a range measurement counts as an obstacle. For outdoor navigation, the size of an object affects whether or not it is considered an obstacle; for example, small bumps on the ground may be ignored, but large rocks are obstacles. This distinction depends on characteristics of the vehicle, such as wheel diameter, ground clearance, velocity, and suspension stiffness. In this context, interaction modeling can take into account just the geometry of the vehicle and world, or varying levels of fidelity in the dynamics of the interaction [10, 11].

Algorithms for Obstacle Detection

Obstacle detection algorithms use the vehicle-world interaction model to transform the sensor data and the world model into an intermediate representation that enables more efficient path planning. There are many approaches to this, which depend on the application context.

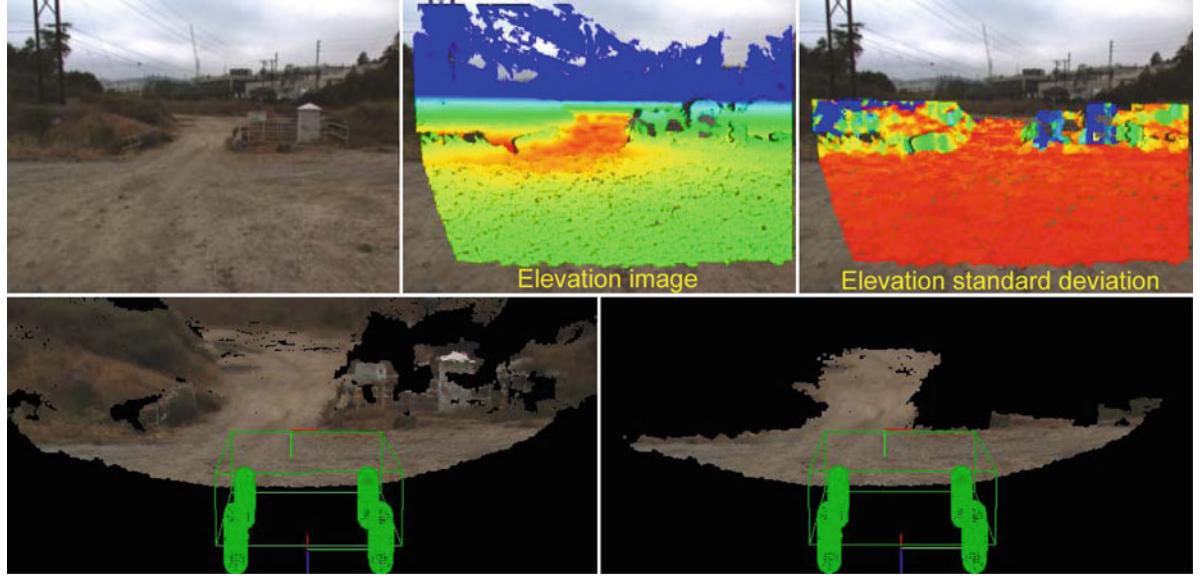
As an example, in a purely geometric approach for ground vehicles using grid-based world models like elevation maps or occupancy grids, local operators are convolved over the map to determine if there is an obstacle at each point in the map. These operators vary from being as simple as thresholding the cell probabilities in occupancy grids, to fitting planes or computing elevation variance over local patches of elevation maps, to configuration tests that tell if a geometric model of the vehicle will fit in a given configuration at a given point in the map without intersecting an object

in the world (Fig. 3) [10, 13]. The output of such algorithms is typically a cost map, with a very high cost assigned to cells the vehicle cannot enter and intermediate costs assigned to cells that can be traversed with some degree of difficulty. Similar concepts can be applied to range data in image space. Path planning algorithms then search for low-cost paths through the cost map.

In some contexts, geometry is not the only consideration: material or mechanical properties of objects also determine whether they are obstacles. Common examples are soft objects like vegetation, which vehicles can readily traverse but which may appear as obstacles to naïve interpretations of range data. In this example, heuristics can be designed based on the spatial scatter of 3-D point clouds to estimate whether a patch of range data represents a solid obstacle surface, or a cloud of thin objects most likely to be traversable vegetation, or thin objects more likely to be wires or fences that are obstacles [3, 12]. Using both geometric and appearance cues is important in this context, since color, texture, and other appearance properties of objects may be as important as their size and shape in determining whether or not they are obstacles.

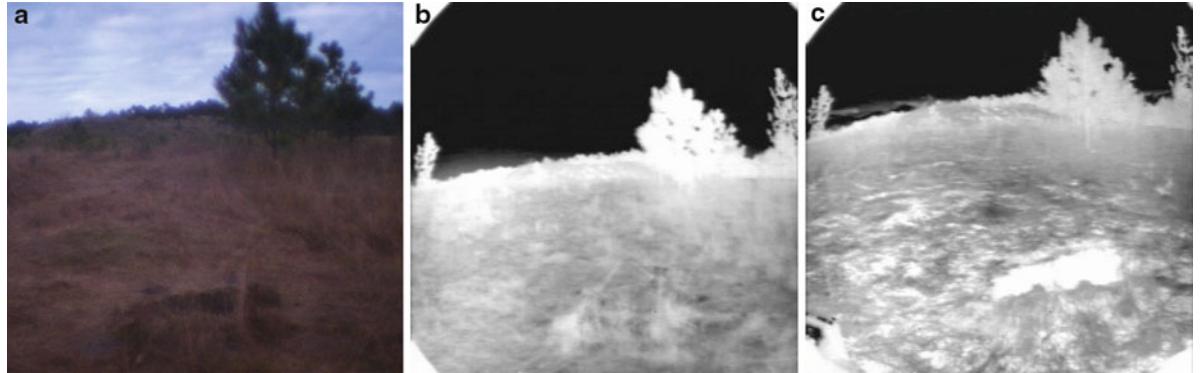
Some classes of obstacles have been detected based on more detailed, physics-based analysis of multiple sensor phenomenologies. *Negative obstacles* are any depression below the ground plane, such as potholes and ditches. These are harder to detect than *positive obstacles*, which protrude above the ground plane, because as their distance increases, negative obstacles grow smaller in the image at a faster rate than positive obstacles. At night, heat transfer phenomena tend to keep the interiors of small negative obstacles warmer than surrounding terrain; therefore, features computed from range data and thermal infrared images can help to detect them (Fig. 4) [14]. Water bodies and patches of ice or snow are another example, where distinctive effects of reflections, absorption, polarization, and temperature can create discriminable features in both range data and specific spectral channels of appearance data [15]. Different materials can have different signatures in radar, so fusion of radar with other sensors also helps to distinguish obstacles from traversable materials.

Early algorithms for obstacle detection often used manually designed procedures based on the designer's knowledge of the problem. For several reasons, there is now a trend to incorporate more automated,



Obstacle Detection, Fig. 3 Some steps in obstacle detection. Top row, left to right: one image of a stereo image pair of a scene; false color overlay of elevation data on the image, with red showing lowest elevation and blue showing highest; false color overlay of elevation standard deviation on image, as computed

from an associated grid map. Red is low standard deviation, blue is high. Bottom row, left to right: 3-D visualization of entire point cloud from stereo matching; 3-D visualization of just the ground surface as segmented from the elevation standard deviation (From [13])



Obstacle Detection, Fig. 4 Thermal phenomenology for negative obstacle detection. (a) Color image of a pothole at 5 p.m.; the pothole is the dark patch in the bottom center of the image. (b) Thermal infrared image of the same scene at the same time.

(c) Thermal infrared image of the same scene at 7 a.m.; now differential cooling of the interior of the pothole and the terrain has made the pothole noticeably warmer in the thermal image

learning-based methods. First, terrain classification algorithms have always needed training data; to reduce the human labor involved, it is desirable to automate the process of labeling the training data. Second, the great variety of objects in the world makes it impractical to manually program rules for distinguishing all obstacles. Sometimes both of these issues can be addressed by using the vehicle's own experience, by

using knowledge gained from one sensor to automatically generate training labels for data from another sensor. Third, obstacles may be easier to detect at short range than at long range, because richer or higher-resolution perception is possible at short range. All of these issues are embodied in *self-supervised, near-to-far* methods for training obstacle detectors. In this paradigm, the most detailed knowledge of obstacles



comes from proprioceptive sensing of physical contact with them (*underfoot*); this knowledge can be used to train classifiers applied to *near field* remote sensors, where terrain is within range of 3-D perception sensors. In turn, classifiers trained with the aid of 3-D perception in the near field can be used to discriminate obstacle using only appearance features, like color and texture, for terrain beyond the reach of 3-D perception (*far field*) [16].

Kinematic issues enter the obstacle detection problem when assessing the likelihood that the vehicle will collide with other moving objects. This raises the problems of segmentation, motion estimation, and behavior prediction for other objects, which are large topics in their own right. Dynamic issues arise when considering whether the vehicle may slip, sink, roll over, or experience damaging reaction forces while traveling. Assessing these factors requires appropriate vehicle-world interaction models. When these models are expensive to evaluate, instead of doing obstacle detection first over the entire world model, then searching for low-cost paths, obstacle detection may be embedded within the path planner. In this case, the path planner searches for candidate paths that obey kinematic and dynamic constraints on vehicle motion and evaluates obstacle-checking criteria within the process of evaluating the quality of each path [17]. This can be made more efficient by applying a hierarchical, triage process, where simple, low-cost obstacle detection algorithms are used first to find areas that are easily traversable, definitely not traversable, or in between, then using higher-fidelity models as needed in the gray areas. Obstacle detection is often done this way when the planning problem involves many degrees of freedom, such as with robot manipulators.

Open Problems

Some classes of obstacles are still difficult to recognize reliably, such as mud [18] and small negative obstacles. Sensor fusion and modeling context are potential approaches to these problems. Self-supervised, near-to-far training of learning-based obstacle detection methods are relatively new techniques and are under ongoing refinement. Dynamical aspects of vehicle-world interaction modeling are still being actively explored, including the interplay between such

models and efficient algorithms for obstacle detection, trafficability evaluation, path planning, and learning. Designing efficient data structures and multi-resolution world models for obstacle detection at both short and long range and over long distance travel remains an important problem. Developing sensor hardware with low size, weight, power, and cost that enables obstacle detection at long range, high speed, and in all environmental conditions will be a long-term challenge.

References

1. Everett HR (1995) Sensors for mobile robots: theory and application. A.K. Peters, Wellesley, MA
2. Matthies L, Bergh C, Castano A, Macedo J, Manduchi R (2005) Obstacle detection in foliage with ladar and radar. In: Dario P, Chatila R (eds) Robotics research: the eleventh international symposium. Springer, Berlin, pp 291–302
3. Matthies L, Kelly A, Litwin T, Tharp G (1996) Obstacle detection for unmanned ground vehicles: a progress report. In: Giralt G (ed) Robotics research: the seventh international symposium. Springer, Berlin, pp 475–486
4. Lalonde J-F, Vandapel N, Hebert M (2007) Data structures for efficient dynamic processing in 3-D. Int J Robot Res 26(8):777–796
5. Moravec H, Elfes AE (1985) High resolution maps from wide angle sonar. In: Proceedings of the IEEE international conference on robotics and automation (ICRA), St. Louis, pp 116–121
6. Wurm KM, Hornung A, Bennewitz M, Stachniss C, Burgard W (2010) OctoMap: a probabilistic, flexible, and compact 3D map representation for robotic systems. In: Proceedings of the workshop on best practice in 3D perception and modeling for mobile manipulation, Anchorage
7. Bajracharya M, Moghaddam B, Howard A, Brennan S, Matthies L (2009) A fast stereo-based system for detecting and tracking pedestrians from a moving vehicle. Int J Robot Res 29(11–12):1466–1485
8. Stentz A, Hebert M (1995) A complete navigation system for goal acquisition in unknown environments. Auton Robot 2(2):127–145
9. Iagnemma K et al (2011) Terramechanics modelling of Mars surface exploration rovers for simulation and parameter estimation. In: Proceedings of the ASME international design engineering technical conference, Washington, DC
10. Olin KE, Tseng DT (1991) Autonomous cross-country navigation: an integrated perception and planning system. IEEE Expert 6(4):16–30
11. Trease B et al (2011) Dynamic modelling and soil mechanics for path planning of the Mars Exploration Rovers. In: Proceedings of the ASME international design engineering technical conference, Washington, DC
12. Lalonde J-F, Vandapel N, Huber DF, Hebert M (2006) Natural terrain classification using three-dimensional ladar data for ground robot mobility. J Field Robot 23(10):839–861

13. Rankin AL, Huertas A, Matthies L (2009) Stereo vision based terrain mapping for off-road autonomous navigation. In: Proceedings of the SPIE symposium on unmanned systems technology XI. Orlando, Florida, USA
14. Matthies L, Rankin A (2003) Negative obstacle detection by thermal signature. In: Proceedings of the IEEE/RSJ conference on intelligent robots and systems (IROS), Las Vegas
15. Matthies L, Bellutta P, McHenry M (2003) Detecting water hazards for autonomous off-road navigation. In: Proceedings of the SPIE symposium on unmanned ground vehicles V. Orlando, Florida, USA
16. Bajracharya M, Howard A, Matthies L, Tang B, Turmon M (2009) Autonomous off-road navigation with end-to-end learning for the LAGR program. J Field Robot 26(1):3–25
17. Howard TM (2009) Adaptive model-predictive motion planning for navigation in complex environments. Ph.D. thesis, Carnegie Mellon University, CMU-RI-TR-09-32
18. Rankin A, Matthies L (2010) Passive sensor evaluation for unmanned ground vehicle mud detection. J Field Robot 27(4):473–490

Occlusion Boundaries from Motion

► [Occlusion Detection](#)

Occlusion Boundaries from Stereo Matching

► [Occlusion Detection](#)

Occlusion Detection

Rodrigo Benenson

K.U. Leuven departement Elektrotechniek – ESAT,
Centrum voor beeld- en spraakverwerking –
PSI/VISICS, Heverlee, Belgium

Synonyms

[Occlusion boundaries from motion](#); [Occlusion boundaries from stereo matching](#); [Foreground-background assignment](#)

Related Concepts

► [Edge Detection](#); ► [Object Detection](#); ► [Occlusion Handling](#)

Definition

Occlusion detection refers to the set of techniques employed to detect which areas of the images are occlusion boundaries or areas that appear occluded in views of the scene.

Background

Occlusion is one of the fundamental phenomenon that limits the information available in an image. Given a 3-dimensional scene containing non transparent objects, a projection of the scene into the image will not include information about all the surfaces in the scene. The backside of the nontransparent objects will be occluded by the frontside, and the foreground objects will occlude the background surfaces. Which areas are occluded and which ones are visible depends on the position of the camera with respect to the scene.

In natural images, occlusions are prevalent. Knowing which are the occlusion boundaries in an image helps segmenting pixels at the object's level and thus help processes such as objects detection or relative depth estimation. The occlusion areas depend on the camera position; thus, most algorithms matching two or more images are concerned with occlusion detection; since not all the surface points visible in image I_1 will be visible in image I_2 , not all pixels in I_1 will have a corresponding pixel I_2 . Due to this stereo matching, optical flow and object tracking algorithms are particularly concerned with occlusion detection.

Theory

For occlusion detection two cases should be distinguished: single image or multiple images. When a single image is available, no direct observation of occlusion boundaries can be done, and thus, the nature of the problem is quite different from the multiple images case.

Single Image

If the depth assigned to each pixel was known, then the occlusion boundaries would correspond to where the depth is discontinuous. Without any additional information but a single image, estimating the



occlusion boundaries corresponds to estimating depth discontinuities when depth information is not available. Without priors, such problem cannot be solved. A common assumption (in stereo matching and optical flow estimation literature) is that depth discontinuities generate color discontinuities. Classical edge detection algorithms can be used to find the color discontinuities and the occlusion detection problem becomes a classification problem; given the detected edges of the image, one wants to know which one corresponds to depth discontinuities and which ones correspond to texture gradient over the same surface. Such classification problem can be addressed using machine learning techniques such as the one described in [1].

Multiple Images

When multiple images are available, there are means to access partial depth information. Such information allows to have a more reliable estimate of the occlusion boundaries. Multiple images of the same scene may come from multiple cameras observing the scene (multi-view imaging, stereo imaging) or from one moving camera observing the scene at two consecutive instants.

Detecting Occlusion Areas via Pixel Matching

When multiple views of the scene are available, it is possible to do pixel-level matching between the multiple images. This is an instance of a data association problem, and it is commonly instantiated as a labeling problem in the vision community. Each pixel in image I_1 will be either visible in I_2 , and thus should be matched or it may be part of an occluded area in I_2 and thus will be rejected during the matching. This strategy is effective for stereo matching (multi-view can be reduced to a set of pair-wise matching problems), and for optical flow cases.

Simultaneously deciding which is the displacement/disparity of each pixel and which pixel is occluded or not is a difficult optimization problem (see, for instance, [2]). A common strategy to relax the problem is to focus on symmetry constraints. Non-occluded pixels are visible in both images; by definition occluded pixels are visible in only one image or none. Thus, a common strategy to detect occluded pixels consists on matching first I_1 to I_2 (disregarding occlusion issues), then matching I_2 to I_1 , and finally verifying which displacements are consistent between

the two matching results. Occluded pixels are expected to have inconsistent displacement.

This strategy has shown acceptable results for stereo matching and optical flow problems [3, 4].

Detecting Occlusion Boundaries via Flow Cues

In the case of optical flow, there are additional cues available other than simply pixel matching. Due to parallax effects or actual object motion, different objects will appear in the image with different (2-dimensional) motion vectors. Assuming object rigidity, it is then possible to delineate objects boundaries by finding areas where the optical flow is divergent [5]. Such objects boundaries estimates are also occlusion boundaries estimates.

Detecting Occlusions During Object Tracking

Previous sections describe occlusion detection at the pixel level. Occlusion detection is also a common concern when trying to track whole objects across multiple frames (i.e., trying to solve the object tracking problem). In its common formulation objects tracking is a data association problem of the same kind as pixel-level matching for depth estimation or optical flow. In its simplest form object-level occlusion detection will be done either via a threshold on the appearance similarity (if the appearance of the expected object location is too different from the object template, the object is considered occluded) or via a temporal threshold on the objects detector (if an object nearby the expected location has not been detected for too long, it is considered occluded) [6].

Application

When computing depth maps, optical flow, or objects tracks for decision making, it is important to know which areas are reliable and which ones are not. By definition occluded areas offer unreliable information (no data) and thus must be detected. Once unreliable areas are detected, the decision process is improved. In applications when the output must be complete (e.g., in computer graphics), knowing which areas are occluded, enables applying infilling/interpolation algorithms over them to provide pleasing results. Occlusion areas and occlusion boundaries are linked to

object boundaries, and as such are also an useful clue for object segmentation and detection.

References

1. Stein A (2008) Occlusion Boundaries: Low-Level Detection to High-Level Reasoning. PhD thesis, Robotics Institute, Carnegie Mellon University
2. Strecha C, Fransens R, Van Gool L (2004) A probabilistic approach to large displacement optical flow and occlusion detection. *Stat Methods Video Process* 3247:25–45
3. Sun J, Li Y, Kang S, Shum HY (2005) Symmetric stereo matching for occlusion handling. In: Proceedings of the IEEE Conference Computer Vision and Pattern Recognition (CVPR)
4. Alvarez L, Deriche R Dr, Papadopoulo T, Sánchez J (2007) Symmetrical dense optical flow estimation with occlusions detection. *Int J Comput Vis* 75(3):371–385
5. Thompson WB, Mutch KM, Berzins VA (1985) Dynamic occlusion analysis in optical flow fields. *IEEE Trans Pattern Anal Mach Intell* 7(4):374–383
6. Yilmaz A, Javed O, Shah M (2006) Object tracking: a survey. *Comput Surv* 38(4):13.1–13.45

Occlusion Handling

Rodrigo Benenson

K.U. Leuven departement Elektrotechniek – ESAT,
Centrum voor beeld- en spraakverwerking –
PSI/VISICS, Heverlee, Belgium

Related Concepts

► [Inpainting](#); ► [Occlusion Detection](#)

Definition

Occlusion handling refers to the set of techniques employed to mitigate the effects of occlusion when doing inference from image.

Background

Occlusion is a fundamental phenomenon that limits the information that can be extracted from a camera observing a scene. Either when computing depth maps, optical flow, or tracking objects, occlusion will interfere with the inference process and create blank

areas. In many application, simply knowing which areas are occluded is good enough for the desired output. In other applications, it is desired to have an infilled/interpolated output without any blank. Occlusion handling makes reference to the latter case.

Theory

How to tackle the occluded areas is an application-specific problem. The three most common tasks that have to deal explicitly with occlusion are stereo matching, optical flow, and objects tracking. Other applications, such as visual odometry, also suffer from occlusion, but they simply ignore it and consider it as input noise.

Occlusion Handling in Pixel Matching

For stereo matching and optical flow, the common strategy is to do extrapolation. Due to occlusion only a partial (non-occluded) view of the background object/surface is available. It is commonly assumed that the background surface is large, and thus covers also the occluded area. Given this assumption, the occluded area is simply infilled using extrapolations from the surrounding connecting background area.

For stereo matching, this can be done by simply infilling row per row the occluded area using the same disparity value as the lowest one (farthest from the camera) between the right and left side of the occlusion area. More sophisticated approaches try to fit local planes to infill the disparity map [1, Chap. 5].

For optical flow, the inpainting is can be done via an edge-sensitive anisotropic smoothing, that will naturally infill the blank areas using the surrounding flow [2].

Occlusion Handling in Object Tracking

In object tracking, one must distinguish the online case (only previous frames are available) and the offline case (all image frames of a video sequence are available).

For the online case, the appearance similarity is the main clue available. When no data association is found between a current track and the elements in the current frame, then the object position is extrapolated using the motion model. If too much time passes since the last successful detection, then the track is aborted. Should



the object reappear later in the video, it will be annotated with a different object identifier (identity switch) unless a life-long set of detected objects is kept, and the match between the re-appearance and the object model is strong [3]. When objects are re-detected and associated after occlusions, the track trajectory can be adjusted based on the new evidence [4].

For the offline case, a global optimization problem can be cast [5]. In such setup, the time of occlusion is trade-off with the strength of the global appearance similarity; it is then possible to track objects through longer occlusions without loosing the object identifier.

References

1. Bleyer M (2006) Segmentation-based Stereo and Motion with Occlusions. PhD thesis, Vienna University of Technology
2. Ince S, Konrad J (2008) Occlusion-aware optical flow estimation. IEEE Trans Image Process 17:1443–1451
3. Yilmaz A, Javed O, Shah M (2006) Object tracking: a survey. Comput Surv 38(4):13.1–13.45
4. Leibe B, Schindler K, Cornelis N, Van Gool L (2008) Coupled detection and tracking from static cameras and moving vehicles. PAMI 30(10):1683–1698
5. Xing J, Ai H, Lao S (2009) Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

Omnidirectional Camera

Davide Scaramuzza

Artificial Intelligence Lab - Robotics and Perception Group, Department of Informatics, University of Zurich, Zurich, Switzerland

Synonyms

Catadioptric camera; Fisheye camera; Panoramic camera; Spherical camera; Wide-angle camera

Related Concepts

- ▶ [Camera Calibration](#); [Camera Parameters \(Intrinsic, Extrinsic\)](#);
- ▶ [Epipolar Geometry](#);
- ▶ [Intrinsic Parameters](#);
- ▶ [Omnidirectional Vision](#);
- ▶ [Structure-from-Motion \(SfM\)](#)

Definition

An omnidirectional camera (from *omni*, meaning all) is a camera with a 360-degree field of view in the horizontal plane or with a visual field that covers a hemisphere or (approximately) the entire sphere.

Background

Most commercial cameras can be described as pinhole cameras, which are modeled by a perspective projection. However, there are projection systems whose geometry cannot be described using the conventional pinhole model because of the very high distortion introduced by the imaging device. Some of these systems are omnidirectional cameras.

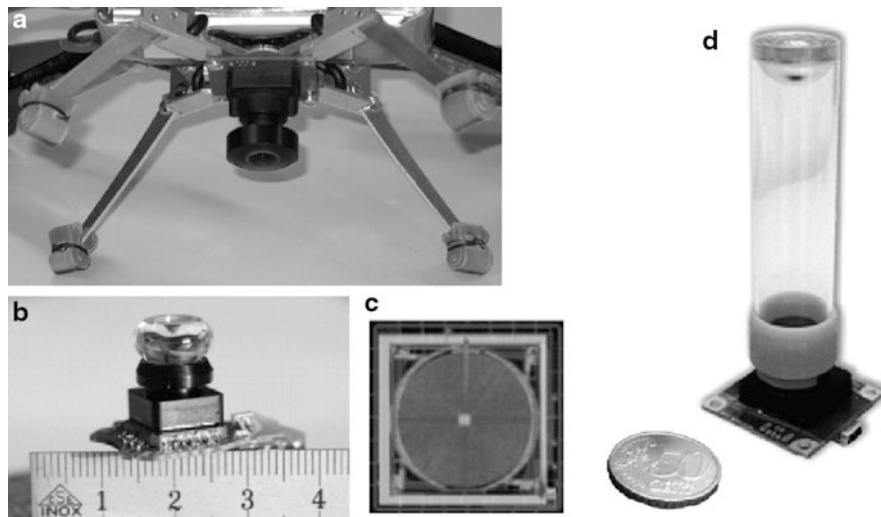
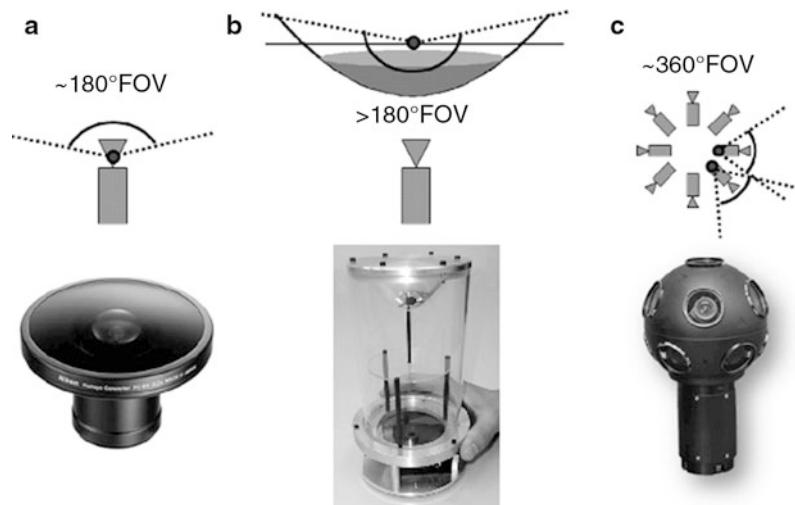
There are several ways to build an omnidirectional camera. Dioptric cameras use a combination of shaped lenses (e.g., fisheye lenses; see Fig. 1a) and can reach a field of view even bigger than 180° (i.e., slightly more than a hemisphere). Catadioptric cameras combine a standard camera with a shaped mirror – such as a parabolic, hyperbolic, or elliptical mirror – and provide 360-degree field of view in the horizontal plane and more than 100° in elevation. In Fig. 1b, you can see an example catadioptric camera using a hyperbolic mirror. Finally, polydioptric cameras use multiple cameras with overlapping field of view (Fig. 1c) and so far are the only cameras that provide a real omnidirectional (spherical) field of view (i.e., 4π steradians).

Catadioptric cameras were first introduced in robotics in 1990 by Yagi and Kawato [1], who used them for localizing robots. Fisheye cameras started to spread over only in 2000 thanks to new manufacturing techniques and precision tools that led to an increase of their field of view up to 180° and more. However, it is only since 2005 that these cameras have been miniaturized to the size of 1–2 cm and their field of view has been increased up to 190° or even more (see, for instance, Fig. 2a).

In the next sections, an overview on omnidirectional camera models and calibration will be given. For an in-depth study on omnidirectional vision, the reader is referred to [2–4] and to [5] for a more detailed survey on omnidirectional camera models.



Omnidirectional Camera, Fig. 1 (a) Dioptric camera (e.g., fisheye); (b) catadioptric camera; (c) an example polydioptric camera produced by Immersive Media



Omnidirectional Camera, Fig. 2 (a) The fisheye lens from Omnitech Robotics (www.omnitech.com) provides a field of view of 190°. This lens has a diameter of 1.7 cm. This camera has been used on the sFly autonomous helicopter at the ETH Zurich, [18]. (b) A miniature catadioptric camera built at the ETH Zurich, which is also used for autonomous flight. It uses

a spherical mirror and a transparent plastic support. The camera measures 2 cm in diameter and 8 cm in height. (c) The muFly camera built by CSEM, which is used on the muFly helicopter at the ETH Zurich. This is one of the smallest catadioptric cameras ever built. Additionally, it uses a polar CCD (d) where pixels are arranged radially

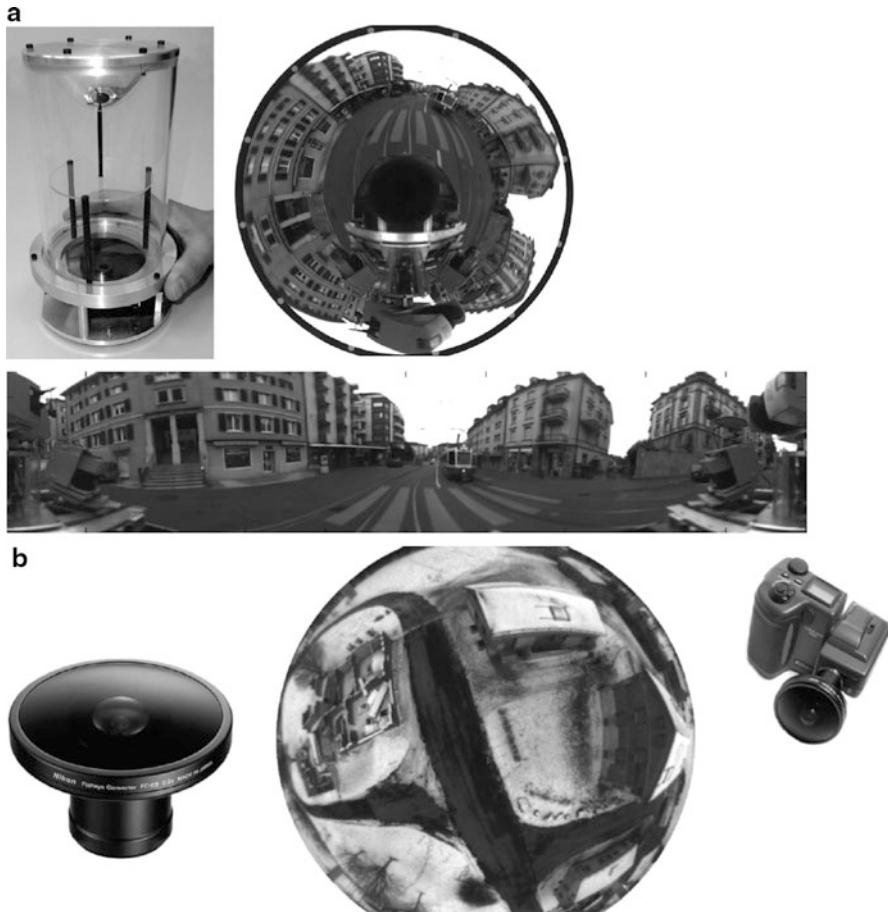
Theory

Central Omnidirectional Cameras

A vision system is said to be central when the optical rays to the viewed objects intersect in a single point in 3D called projection center or single effective viewpoint (Fig. 4). This property is called single effective viewpoint property. The perspective camera is an example of a central projection system because all

optical rays intersect in one point, that is, the camera optical center.

All modern fisheye cameras are central, and hence, they satisfy the single effective viewpoint property. Central catadioptric cameras conversely can be built only by opportunely choosing the mirror shape and the distance between the camera and the mirror. As proven by Baker and Nayar [6], the family of mirrors that satisfy the single viewpoint property is the class



Omnidirectional Camera, Fig. 3 (a) A cadioptic omnidirectional camera using a hyperbolic mirror. The image is typically unwrapped into a cylindrical panorama. The field of view

is typically 100° in elevation and 360° in azimuth. (b) Nikon fisheye lens FC-E8. This lens provides a hemispherical (180°) field of view

of rotated (swept) conic sections, that is, hyperbolic, parabolic, and elliptical mirrors. In the case of hyperbolic and elliptical mirrors, the single view point property is achieved by ensuring that the camera center (i.e., the pinhole or the center of the lens) coincides with one of the foci of the hyperbola (ellipse) (Fig. 5). In the case of parabolic mirrors, an orthographic lens must be interposed between the camera and the mirror; this makes it possible that parallel rays reflected by the parabolic mirror converge to the camera center (Fig. 5).

The reason a single effective viewpoint is so desirable is that it allows the user to generate geometrically correct perspective images from the pictures captured by the omnidirectional camera (Fig. 6). This is possible because, under the single view point constraint, every pixel in the sensed image measures

the irradiance of the light passing through the viewpoint in one particular direction. When the geometry of the omnidirectional camera is known, that is, when the camera is calibrated, one can precompute this direction for each pixel. Therefore, the irradiance value measured by each pixel can be mapped onto a plane at any distance from the viewpoint to form a planar perspective image. Additionally, the image can be mapped on to a sphere centered on the single viewpoint, that is, spherical projection (Fig. 6, bottom).

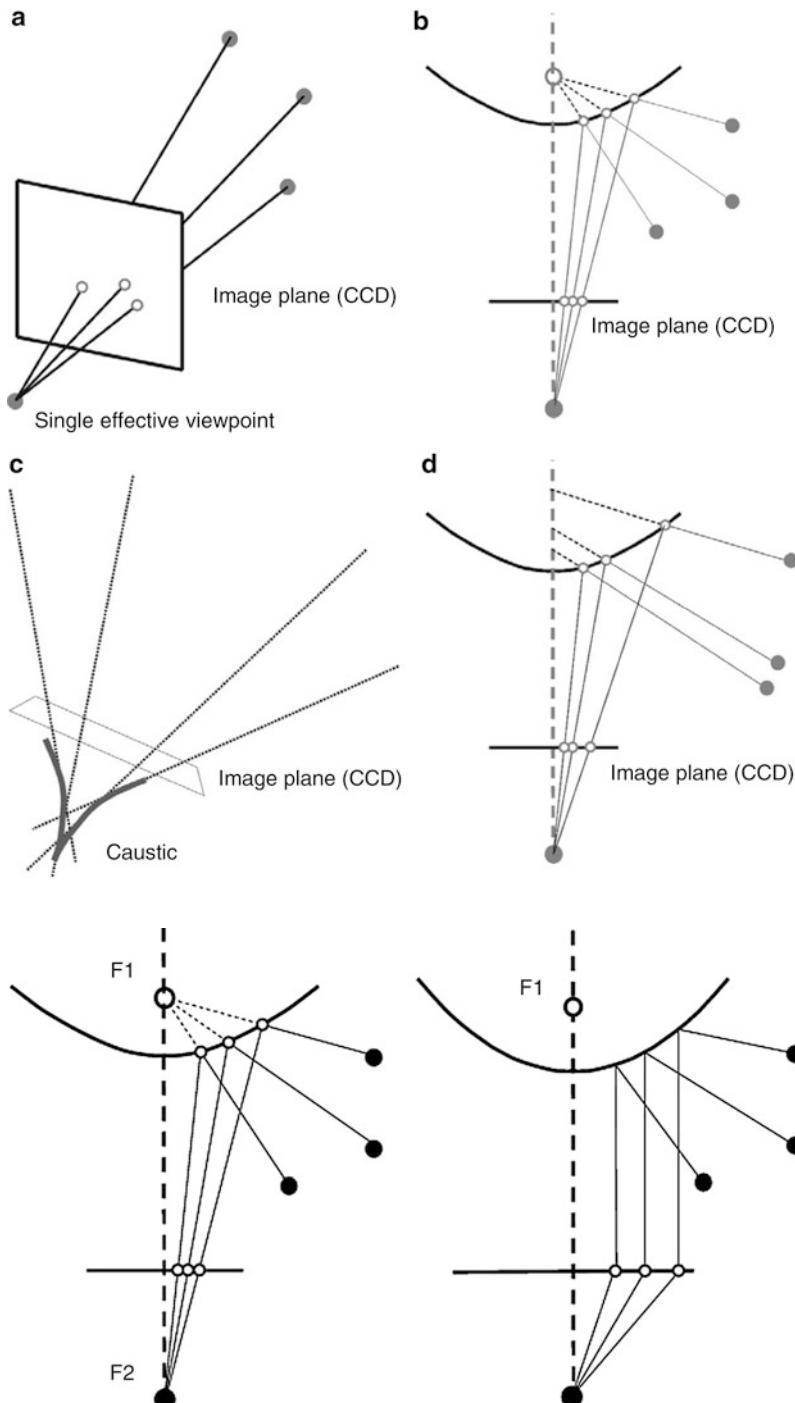
Another reason why the single view point property is so important is that it allows the user to apply the well-known theory of epipolar geometry, which is extremely important for structure from motion. Epipolar geometry holds for any central camera, both perspective and omnidirectional.



Omnidirectional

Camera, Fig. 4 (a) and (b)

Example of central cameras: perspective projection and catadioptric projection through a hyperbolic mirror. (c) and (d) Example of noncentral cameras: the envelope of the optical rays forms a caustic



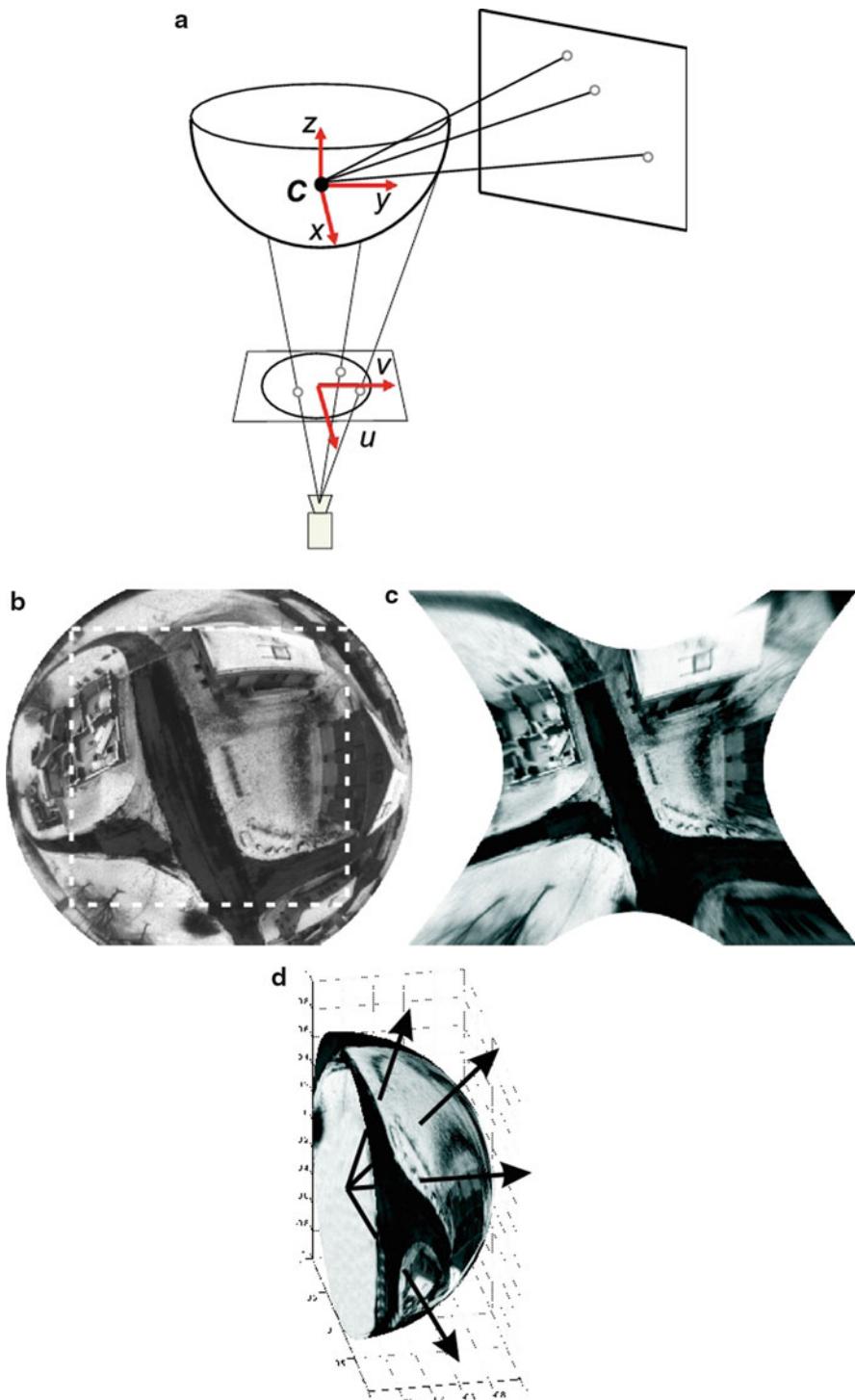
Omnidirectional

Camera, Fig. 5 Central catadioptric cameras can be built by using hyperbolic and parabolic mirrors. The parabolic mirror requires the use of an orthographic lens

Omnidirectional Camera Model and Calibration

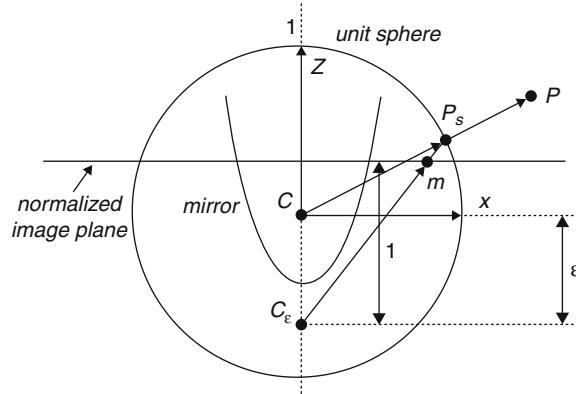
Intuitively, the model of an omnidirectional camera is a little more complicated than a standard perspective camera. The model should indeed take into account the reflection operated by the mirror in the case of a

catadioptric camera or the refraction caused by the lens in the case of a fisheye camera. Because the literature in this field is quite large, this entry reviews two different projection models that have become standards in omnidirectional vision and robotics. Additionally,



Omnidirectional Camera, Fig. 6 Central cameras allow the user to remap regions of the omnidirectional image into a perspective image. This can be done straightforwardly by intersecting the optical rays with a plane specified arbitrarily by the user

(a). For obvious reasons, we cannot project the whole omnidirectional image onto a plane but only subregions of it (b-c). Another alternative is the projection onto a sphere (d). In this case, the entire omnidirectional image can be remapped to a sphere



Omnidirectional Camera, Fig. 7 Unified projection model for central catadioptric cameras of Geyer and Daniilidis

Matlab toolboxes have been developed for these two models, which are used worldwide by both specialists and nonexperts.

The first model is known as the unified projection model for central catadioptric cameras. It was developed in 2000 by Geyer and Daniilidis [7] (later refined by Barreto and Araujo [8]), who have the merit of having proposed a model that encompasses all three types of central catadioptric cameras, that is, cameras using a hyperbolic, parabolic, or elliptical mirror. This model was developed specifically for central catadioptric cameras and is not valid for fisheye cameras. The approximation of a fisheye lens model by a catadioptric one is usually possible – however, with limited accuracy only – as investigated in [9].

Conversely, the second model unifies both central catadioptric cameras and fisheye cameras under a general model also known as Taylor model. It was developed in 2006 by Scaramuzza et al. [10, 11] and has the advantage that both catadioptric and dioptric cameras can be described through the same model, namely, a Taylor polynomial.

Unified Model for Central Catadioptric Cameras

With their landmark paper from 2000, Geyer and Daniilidis showed that every catadioptric (parabolic, hyperbolic, elliptical) and standard perspective projection is equivalent to a projective mapping from a sphere, centered in the single viewpoint, to a plane with the projection center on the perpendicular to the plane and distant ϵ from the center of the sphere. This is summarized in Fig. 7.

Omnidirectional Camera, Table 1 ϵ values for different types of mirrors

| Mirror type | ϵ |
|-------------|-----------------------------|
| Parabola | 1 |
| Hyperbola | $\frac{d}{\sqrt{d^2+4l^2}}$ |
| Ellipse | $\frac{d}{\sqrt{d^2+4l^2}}$ |
| Perspective | 0 |

The goal of this section is to find the relation between the viewing direction to the scene point and the pixel coordinates of its corresponding image point. The projection model of Geyer and Daniilidis follows a four-step process. Let $P = (x, y, z)$ be a scene point in the mirror reference frame centered in C (Fig. 7). For convenience, we assume that the axis of symmetry of the mirror is perfectly aligned with the optical axis of the camera. We also assume that the x and y axes of the camera and mirror are aligned. Therefore, the camera and mirror reference frames differ only by a translation along z .

- The first step consists in projecting the scene point onto the unit sphere; therefore,

$$P_s = \frac{P}{\|P\|} = (x_s, y_s, z_s). \quad (1)$$

- The point coordinates are then changed to a new reference frame centered in $C_\epsilon = (0, 0, -\epsilon)$; therefore,

$$P_\epsilon = (x_s, y_s, z_s + \epsilon). \quad (2)$$

Observe that ϵ ranges between 0 (planar mirror) and 1 (parabolic mirror). The correct value of ϵ can be obtained knowing the distance d between the foci of the conic and the latus rectum l as summarized in Table 1. The latus rectum of a conic section is the chord through a focus parallel to the conic section directrix.

- P_ϵ is then projected onto the normalized image plane distant 1 from C_ϵ ; therefore,

$$\tilde{m} = (x_m, y_m, 1) = \left(\frac{x_s}{z_s + \epsilon}, \frac{y_s}{z_s + \epsilon}, 1 \right) = g^{-1}(P_s). \quad (3)$$

- Finally, the point \tilde{m} is mapped to the camera image point $\tilde{p} = (u, v, 1)$ through the intrinsic parameter



matrix K ; therefore,

$$\tilde{p} = K\tilde{m}, \quad (4)$$

where K is

$$K = \begin{bmatrix} \alpha_u & \alpha_u \cot(\theta) & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (5)$$

- It is easy to show that function g^{-1} is bijective and that its inverse g is given by:

$$P_s = g(m) \propto \begin{bmatrix} x_m \\ y_m \\ 1 - \epsilon \frac{x_m^2 + y_m^2 + 1}{\epsilon + \sqrt{1 + (1 - \epsilon^2)(x_m^2 + y_m^2)}} \end{bmatrix}, \quad (6)$$

where \propto indicates that g is proportional to the quantity on the right-hand side. To obtain the normalization factor, it is sufficient to normalize $g(m)$ onto the unit sphere.

Equation (6) can be obtained by inverting (3) and imposing the constraint that P_s must lie on the unit sphere and, thus, $x_s^2 + y_s^2 + z_s^2 = 1$. From this constraint, we then get an expression for z_s as a function of ϵ , x_m , and y_m . More details can be found in [12].

Observe that Eq. (6) is the core of the projection model of central catadioptric cameras. It expresses the relation between the point m on the normalized image plane and the unit vector P_s in the mirror reference frame. Note that in the case of planar mirror, we have $\epsilon = 0$ and (6) becomes the projection equation of perspective cameras $P_s \propto (x_m, y_m, 1)$.

This model has proved to be able to describe accurately all central catadioptric cameras (parabolic, hyperbolic, and elliptical mirror) and standard perspective cameras. An extension of this model for fisheye lenses was proposed in 2004 by Ying and Hu [9]. However, the approximation of a fisheye camera through a catadioptric one works only with limited accuracy. This is mainly because, while the three types of central catadioptric cameras can be represented through an exact parametric function (parabola, hyperbola, ellipse), the projective models of fisheye lenses vary from camera to camera and depend on the lens field of view. To overcome this problem, a new unified model was proposed, which will be described in the next section.

Unified Model for Catadioptric and Fisheye Cameras

This unified model was proposed by Scaramuzza et al. in 2006 [10, 11]. The main difference with the previous model lies in the choice of the function g . To overcome the lack of knowledge of a parametric model for fisheye cameras, the authors proposed the use of a Taylor polynomial, whose coefficients and degree are found through the calibration process. Accordingly, the relation between the normalized image point $\tilde{m} = (x_m, y_m, 1)$ and the unit vector P_s in the fisheye (mirror) reference frame can be written as:

$$P_s = g(m) \propto \begin{bmatrix} x_m \\ y_m \\ a_0 + a_2 \rho^2 + \dots + a_N \rho^N \end{bmatrix}, \quad (7)$$

where $\rho = \sqrt{x_m^2 + y_m^2}$. As the reader may observe, the first-order term (i.e., $a_1 \rho$) of the polynomial is missing. This follows from the observation that the first derivative of the polynomial calculated at $\rho = 0$ must be null for both catadioptric and fisheye cameras (this is straightforward to verify for catadioptric cameras by differentiating (6)). Also observe that because of its polynomial nature, this expression can encompass catadioptric, fisheye, and perspective cameras. This can be done by opportunely choosing the degree of the polynomial. As highlighted by the authors, polynomials of order 3 or 4 are able to model very accurately all catadioptric cameras and many types of fisheye cameras available on the market. The applicability of this model to a wide range of commercial cameras is at the origin of its success.

Omnidirectional Camera Calibration

The calibration of omnidirectional cameras is similar to that for calibrating standard perspective cameras. Again, the most popular methods take advantage of planar grids that are shown by the user at different positions and orientations. For omnidirectional cameras, it is very important that the calibration images are taken all around the camera and not on a single side only. This is in order to compensate for possible misalignments between the camera and mirror.

It is worth to mention three open-source calibration toolboxes currently available for Matlab, which differ mainly for the projection model adopted and the type of calibration pattern:

- The toolbox of Mei uses checkerboard-like images and takes advantage of the projection model of Geyer and Daniilidis discussed earlier. It is particularly suitable for catadioptric cameras using hyperbolic, parabolic, folded mirrors, and spherical mirrors. Mei's toolbox can be downloaded from [13], while the theoretical details can be found in [14].
- The toolbox of Barreto uses line images instead of checkerboards. Like the previous toolbox, it also uses the projection model of Geyer and Daniilidis. It is particularly suitable for parabolic mirrors. The toolbox can be downloaded from [12], while the theoretical details can be found in [15] and [16].
- Finally, the toolbox of Scaramuzza uses checkerboard-like images. Contrary to the previous two, it takes advantage of the unified Taylor model for catadioptric and fisheye cameras developed by the same author. It works with catadioptric cameras using hyperbolic, parabolic, folded mirrors, spherical, and elliptical mirrors. Additionally, it works with a wide range of fisheye lenses available on the market – such as Nikon, Sigma, and Omnitech Robotics – with field of view up to 195°. The toolbox can be downloaded from [17], while the theoretical details can be found in [10] and [11]. Contrary to the previous two toolboxes, this toolbox features an automatic calibration process. In fact, both the center of distortion and the calibration points are detected automatically without any user intervention. This toolbox became very popular and is currently used at several companies such as NASA, Philips, Bosch, Daimler, and XSens.

Application

Thanks to the camera miniaturization, to the recent developments in optics manufacturing, and to the decreasing prices in the cameras market, catadioptric and dioptric omnidirectional cameras are being more and more used in different research fields. Miniature dioptric and catadioptric cameras are now used by the automobile industry in addition to sonars for improving safety, by providing to the driver an omnidirectional view of the surrounding environment. Miniature fisheye cameras are used in endoscopes for surgical operations or onboard microaerial vehicles for pipeline inspection as well as rescue operations. Other examples involve meteorology for sky observation.

Roboticians have also been using omnidirectional cameras with very successful results on robot localization, mapping, and aerial and ground robot navigation [18–23]. Omnidirectional vision allows the robot to recognize places more easily than with standard perspective cameras [24]. Furthermore, landmarks can be tracked in all directions and over longer periods of time, making it possible to estimate motion and build maps of the environment with better accuracy than with standard cameras; see Fig. 2 for some of examples of miniature omnidirectional cameras used on state-of-the-art micro aerial vehicles. Several companies, like Google, are using omnidirectional cameras to build photorealistic street views and three-dimensional reconstructions of cities along with texture. Two example omnidirectional images are shown in Fig. 3.

References

1. Yagi Y, Kawato S (1990) Panorama scene analysis with conic projection. In: IEEE international conference on intelligent robots and systems, workshop on towards a new frontier of applications, Ibaraki
2. Benosman R, Kang, S (2001) Panoramic vision: sensors, theory, and applications. Springer, New York
3. Daniilidis K, Klette R (2006) Imaging beyond the pinhole camera. Springer, New York
4. Scaramuzza D (2008) Omnidirectional vision: from calibration to robot motion estimation, PhD thesis n. 17635, ETH Zurich
5. Sturm P, Ramalingam S, Tardif J, Gasparini S, Barreto J (2010) Camera models and fundamental concepts used in geometric computer vision. *FI Int J Comput Vis* 35(2):175–196
6. Baker S, Nayar S, A theory of single-viewpoint catadioptric image formation. *Int J Comput Vis* 35(2):175–196
7. Geyer C, Daniilidis K (2000) A unifying theory for central panoramic systems and practical applications. In: European conference on computer vision (ECCV), Dublin
8. Barreto J (2001) Issues on the geometry of central catadioptric image formation, Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition (CVPR)
9. Ying X, Hu Z (2004) Can we consider central catadioptric cameras and fisheye cameras within a unified imaging model? In: European conference on computer vision (ECCV). Lecture notes in computer science. Springer, Berlin
10. Scaramuzza D, Martinelli A, Siegwart R (2006) A flexible technique for accurate omnidirectional camera calibration and structure from motion. In: IEEE international conference on computer vision systems, New York
11. Scaramuzza D, Martinelli A, Siegwart R (2006) A toolbox for easy calibrating omnidirectional cameras. In: IEEE international conference on intelligent robots and systems, Beijing



12. Barreto J, Omnidirectional camera calibration toolbox for matlab (ocamcalib toolbox). <http://www.isr.uc.pt/~jpbar/CatPack/pag1.htm>
13. Mei C, Omnidirectional camera calibration toolbox for matlab. <http://homepages.laas.fr/~cmei/index.php/Toolbox>
14. Mei C, Rives P(2007) Single view point omnidirectional camera calibration from planar grids. In: IEEE international conference on robotics and automation, Roma
15. Barreto J, Araujo H (2005) Geometric properties of central catadioptric line images and their application in calibration. *IEEE Trans Pattern Anal Mach Intell* 27(8): 1237–1333
16. Barreto J, Araujo H (2006) Fitting conics to paracatadioptric projection of lines. *Comput Vis Image Underst* 101(3): 151–165
17. Scaramuzza D, Omnidirectional camera calibration toolbox for matlab (ocamcalib toolbox) Google for “*ocamcalib*”. <https://sites.google.com/site/scarabotix/ocamcalib-toolbox>
18. Blosch M, Weiss S, Scaramuzza D, Siegwart R (2010) Vision based mav navigation in unknown and unstructured environments. IEEE international conference on robotics and automation, Anchorage
19. Bosse M, Rikoski R, Leonard J, Teller S (2002) Vanishing points and 3d lines from omnidirectional video. In: International conference on image processing, Rochester
20. Corke P, Strelow D, Singh S (2004) Omnidirectional visual odometry for a planetary rover. In: IEEE/RSJ international conference on intelligent robots and systems, Sendai
21. Scaramuzza D, Siegwart R (2008) Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles. *IEEE Trans Robot* 24(5): 1015–1026
22. Scaramuzza D, Fraundorfer F, Siegwart R (2009) Real-time monocular visual odometry for on-road vehicles with 1-point ransac. In: IEEE international conference on robotics and automation, Kobe
23. Tardif J, Pavlidis Y, Daniilidis K (2008) Monocular visual odometry in urban environments using an omnidirectional camera. In: IEEE/RSJ international conference on intelligent robots and systems, Nice
24. Scaramuzza D, Fraundorfer F, Pollefeys M (2010) Closing the loop in appearance-guided omnidirectional visual odometry by using vocabulary trees. *Robot Auton Syst* J. 58(6):820–827. Elsevier

Omnidirectional Vision

Peter Sturm
INRIA Grenoble Rhône-Alpes, St Ismier Cedex,
France

Related Concepts

►Field of View; ►Fisheye Lens

Definition

Omnidirectional vision consists of the theoretical and practical computer vision approaches devised for images acquired by omnidirectional cameras, that is, cameras with a very large field of view, typically hemispherical or larger.

Background

Most regular cameras have a restricted field of view, typically up to several tens of degrees. Omnidirectional cameras, on the contrary, have hemispheric or even larger, up to complete spherical, fields of view. Computer vision theories and algorithms dedicated to omnidirectional images are subsumed under the expression omnidirectional vision. Researchers working on omnidirectional vision have also proposed novel designs of omnidirectional cameras.

The interest of building omnidirectional cameras arose rather soon after the invention of photography, with the first known panoramic camera built in 1843 [1]. To be precise, this and other early panoramic cameras had a restricted vertical field of view and are thus not truly omnidirectional.

There exist several technologies to acquire omnidirectional images. The first to be used was based on rotating a regular camera about its optical center and optically or computationally stitching images together to a panoramic image. *Panoramic cameras* can be considered as a subset of omnidirectional ones, in that they usually deliver an extended field of view only in one direction. If the camera is rotated about different axes, acquired images may be stitched to form a complete omnidirectional mosaic.

Probably the second omnidirectional technology was based on multi-camera systems, where the rotating camera was replaced by several colocated cameras with overlapping fields of view. Such systems were built at least as early as in 1884 [2, 3].

The most common single-camera solutions that allow the acquisition of instantaneous omnidirectional images, that is, without having to acquire multiple images while rotating a camera, are *fisheye cameras* and *catadioptric cameras*. The latter achieve large fields of view by having one or more cameras look at a curved mirror or one or more planar or curved mirrors.

Theory

When working with images acquired by omnidirectional cameras, one usually first needs an *image formation model*. Many camera models have been proposed for omnidirectional systems. Some of them resemble classical models for radial and tangential distortion in that they are based on polynomial expressions. Others, typically models for fisheye lenses, use trigonometric functions and yet others are specific to catadioptric cameras. A recent overview of such camera models can be found in [3]. Common to all models is that they allow to project 3D points onto image points and/or perform back projection, from image points to lines of sight in 3D.

Besides purely geometrical aspects of omnidirectional image formation, other aspects and properties have been examined such as focusing mechanisms, blur, and chromatic aberrations; see, for instance, [4]. Generally speaking, these issues all tend to be more complex than with regular cameras.

The *calibration* of an omnidirectional camera is in principle not different from that of a regular camera, in that it exploits images of reference objects with known shape or special properties, for example, showing straight lines on their surface. In practice, calibrating an omnidirectional camera is usually more complex, since usual calibration objects do not fill the field of view completely, thus requiring the acquisition of more images. Also, omnidirectional images are often less sharp and are by definition less resolved, thus making feature extraction and matching more challenging.

Besides modeling and calibrating omnidirectional cameras, other fundamental works in omnidirectional vision are dedicated to image processing. The goal of these works is to adapt image processing operations, for example, for edge or interest point extraction, to the image formation model which usually includes strong distortions, in order to enhance their performance [5, 6].

Application

Once an omnidirectional camera is calibrated, one can in principle use the same algorithms for tasks such as pose estimation, motion estimation, or 3D modeling as for regular cameras and the usual model thereof (pinhole model). There exist several differences

though. For example, for motion estimation, results with omnidirectional cameras are often better than with cameras with a smaller field of view [7]. With a small field of view, a lateral translation and a lateral rotation give rise to similar optical flows, whereas in an omnidirectional image, the optical flows in the “sideways” looking parts of the image will be very different for these two types of motion. This explains why motion estimation may be expected to be more stable when performed with omnidirectional images. For pose estimation, the converse is often the case, simply because the spatial resolution is lower than with a smaller field of view, decreasing the accuracy of the estimated object pose. Further, as mentioned above, processing of omnidirectional images may have to have recourse to specific approaches, in order to handle the strong distortions present in them.

Typical applications of omnidirectional vision are those where the extended field of view is directly beneficial. Among the first applications of omnidirectional images were the study of cloud formations in meteorology and the measurement of leaf coverage from fisheye images of forest canopies. A main application field for omnidirectional vision is the navigation of mobile robots, where obstacles can be detected instantaneously in all directions and where an omnidirectional camera allows for a more stable motion estimation and a more complete path planning than a narrow field of view camera. Other obvious usages are in video surveillance and tele-presence applications.

References

1. McBride B (2011) A timeline of panoramic cameras. <http://www.panoramicphoto.com/timeline.htm>. Accessed 3 Aug 2011
2. Tissandier G (1886) La photographie en ballon. Gauthier-Villars, Paris
3. Sturm P, Ramalingam S, Tardif JP, Gasparini S, Barreto J (2011) Camera models and fundamental concepts used in geometric computer vision. Found Trends Comput Graph Vis 6(1–2):1–183
4. Baker S, Nayar S (1999) A theory of single-viewpoint catadioptric image formation. Int J Comput Vis 35(2):1–22
5. Daniilidis K, Makadia A, Bülow T (2002) Image processing in catadioptric planes: spatiotemporal derivatives and optical flow computation. In: Proceedings of the workshop on omnidirectional vision, copenhagen, Denmark, pp 3–10
6. Bülow T (2002) Multiscale image processing on the sphere. In: Proceedings of the 24th DAGM symposium. Lecture notes in computer science, Zurich, Switzerland, vol 2449, pp 609–617



7. Nelson R, Aloimonos J (1988) Finding motion parameters from spherical motion fields (or the advantages of having eyes in the back of your head). *Biol Cybern* 58(4):261–273
8. Yagi Y (1999) Omnidirectional sensing and applications. *IEICE Trans Inf Syst E82-D(3):568–579*
9. Benosman R, Kang S (eds) (2001) Panoramic vision. Springer, New York

Opacity

Ivo Ihrke
MPI Informatik, Saarland University, Saarbrücken,
Germany

Synonyms

Opaque

Related Concepts

►Polarization; ►Transparency and Translucency

Definition

Opacity describes the degree to which an object or material is not transmitting light. Sometimes the opposite is referred to as *transmittance* (mostly in physics). In computer graphics, the opposite of opacity is typically referred to as *transparency*.

Background

Most computer vision algorithms, whether designed for 3D reconstruction, object detection, object classification, etc., require the scene and the objects it consists of to be opaque. More precisely, objects are required to be diffusely opaque. Specularly reflective objects, e.g., are opaque by definition but most vision techniques cannot handle them.

A multitude of research has gone into lifting the restriction on scene opacity. When going beyond opaque objects, images are typically enhanced by an *alpha channel* – a fractional measure of opacity, or the amount by which an object obscures the background in an image.

In television and film production, the alpha channel is extensively used and usually estimated by *chroma keying* [1]. This technique uses a colored background in order to estimate the alpha channel.

Theory

The following contains a discussion of the physical (optical) principles causing the perception of opacity. Please refer to the classic textbook by Born and Wolf [2] for a detailed discussion.

Physical Processes

There is not a single physical process giving rise to what is known as opacity in computer vision. There can be a multitude of reasons for opacity to occur. The physical processes involved are those of *attenuation* and *scattering* of light upon molecular interaction with an object material, Fig. 1. A further, unrelated reason is physical *sensor integration* over the area of a single pixel: If the scene consists of many tiny objects that are somehow interacting with the surrounding light, seen against a background, sensor integration averages over the light contribution from the objects and the background. This can often be seen when observing thin structures like hair or fur, or leaves on distant trees.

Scattering describes a broad range of physical processes, both particle and wave phenomena, upon interaction of light with a material. The term itself describes deviation of light from a straight path due to inhomogeneities in the material that is being traversed. As an example, scattering can be caused by small particles in paints, droplets in clouds and water spray, soot particles in smoke, and many more. Waves can be scattered by obstacles that are small in comparison to the wavelength. A formal description of energy transfer in scattering media, of which light transport is a special case, can be found in [3].

Scattering can occur both in transmission and in reflection. The latter is usually described by the *bidirectional reflection distribution function* (BRDF), whereas the former is described by the *bidirectional transmission distribution function* (BTDF). These functions describe scattering at an interface or due to very thin transmissive structures. Sometimes, the combined effect of reflection and transmission at an interface is described by a single *bidirectional scattering*

distribution function (BSDF). This family of distribution functions can be seen as a statistical description of the scattering process: The probability that a particular photon is redirected into a given direction is modeled by the density function. A deterministic description, on the other hand, would involve quantum mechanical concepts (Fig. 1).

In contrast, for volumetric phenomena, the redistribution of light is described by the *scattering phase function*. For light traversing a volume of participating media, the major effects are out-scattering, i.e., light being deflected from its original, unobstructed path, and in-scattering, i.e., light being deflected from elsewhere onto the unobstructed path of a photon, Fig. 1. If only a single scattering event occurs, the process is called *single scattering*, otherwise it is referred to as *multiple scattering*.

The above mentioned effects all belong to the family of inelastic scattering processes where photons are essentially bounced around without being modified in wavelength. The counterpart, elastic scattering describes processes where a photon is absorbed by some atom just to be released in a different direction. This way, a change of energy levels can occur. The major effects are fluorescence and phosphorescence.

Attenuation is due to the physical process of absorption: Photons, while traversing some medium, may collide with molecules that are able to absorb their energy, Fig. 1. Usually this energy is converted to a faster motion of the molecule, producing heat as a net effect. The amount of absorption depends on the path length of the photon in the traversed medium. It is given by the

Beer-Lambert law

$$\frac{L}{L_0} = \exp^{-\int_{s_0}^{s_1} \sigma_s \circ c(s) + \sigma_a \circ c(s) ds}.$$

Here, L_0 is the incident radiance and L the radiance after attenuation. The path of the photon is described by the curve $c(s) : \mathbb{R} \mapsto \mathbb{R}^3$, σ_s is the *absorption coefficient* and σ_a the *scattering coefficient*. The scattering coefficient describes the loss of light due to out-scatter, while the absorption coefficient describes the loss due to absorption. Sometimes, the factor $\alpha = \sigma_s + \sigma_a$ is referred to as *attenuation coefficient*.

The linearized version of the Beer-Lambert law $\ln(L/L_0) = -\int_{s_0}^{s_1} \sigma_s \circ c(s) + \sigma_a \circ c(s) ds$ is the basis for emission and absorption tomography.

The *Fresnel Effect* is a natural source of attenuation and blending at interfaces of materials with different refractive index. A ray is usually split into a transmitted and a reflected part. While the geometry and the law of reflection, respective Snell's law, determine only the directions of the reflected and refracted ray, the radiance ratio can be computed with the Fresnel formulae. The transmittance and reflectance depend on the polarization state of the incident ray.

Refraction

As previously discussed, a ray of light crossing an interface is typically both reflected and transmitted. The transmitted part is usually refracted as well, see Fig. 1. This results in a different background pixel to be seen if a refractive object is moved into a ray that previously hit the background. A single opacity value for a camera pixel is thus not sufficient to explain the complex change in light transport. Rather, the opacity value must be associated with the ray in question. For opaque objects, the attenuation of the transmitted ray is very strong. They are thus perceived as not transmitting light. A special case of refraction occurs in unevenly heated media like gases or liquids or in mixtures of different gases. Here, *continuous refraction* is taking place, resulting in curved light paths. In wave optics, the *complex refractive index* is used as a mathematical tool to represent continuous refraction and attenuation in one number.

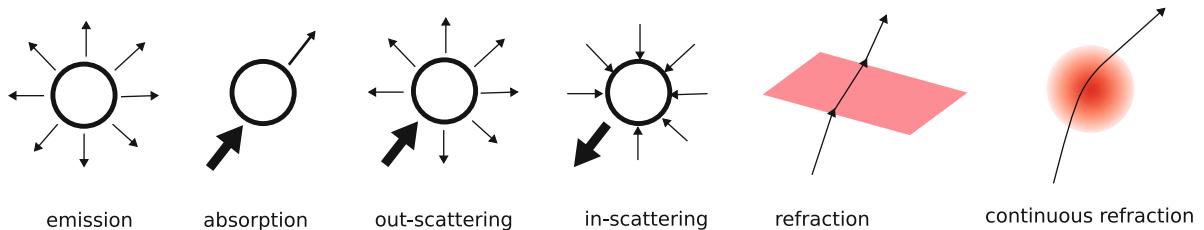
Application

Alpha Matting

Alpha matting is an extension to chroma keying discussed in the background section. Here, the more general problem where the background is arbitrary instead of consisting of a single color is considered. This problem comes in several flavors:

- Dynamic scene, static known background
- Static scene, static unknown background, i.e., natural image matting
- Dynamic scene, static unknown background
- Dynamic scene, dynamic background, i.e., video matting [4]

The corresponding process of synthetically overlaying an object image onto some background is known as *alpha blending*.



Opacity, Fig. 1 The major light transport effects in participating media (from *left to right*): emission, absorption, out-scattering, in-scattering, and two types of refraction

Apart from these two-dimensional applications, opacity has been used to three-dimensionally model transparent or translucent objects either phenomenologically as partially transparent view-dependent visual hulls [5] or for estimating the internal structure of volumetric light-emitting or absorbing objects and phenomena. The latter is usually performed using tomographic techniques [6].

Tomography

Tomography is the process of determining a function from its projections. In the tomographic sense, a projection is a line integral through some volumetric function. A collection of line integrals, taken from many different directions, can be used to compute the inner structure of volumetric phenomena. The best known application is computed tomography (CT). In computer vision, attempts to reconstruct volumetric phenomena like fire, smoke, and plasmas using tomographic techniques applied to observed opacities, even dynamically, have shown good success.

References

1. Gerald Millerson J Owens (2009) Television production. Focal Press, Amsterdam/Boston
2. Born M, Wolf E (1999) Principles of optics, 7th edn. Cambridge University Press, Cambridge, UK
3. Siegel R, Howell JR (1992) Thermal radiation heat transfer, 3rd edn. Taylor & Francis, New York, USA
4. Chuang YY, Agarwala A, Curless B, Salesin DH, Szeliski R (2002) Video matting of complex scenes. In: SIGGRAPH '02: proceedings of the 29th annual conference on computer graphics and interactive techniques, San Antonio. ACM, New York, pp 243–248
5. Matusik W, Pfister H, Ngan A, Beardsley P, Ziegler R, McMillan L (2002) Image-based 3D photography using opacity hulls. In: SIGGRAPH '02: proceedings of the 29th

annual conference on computer graphics and interactive techniques, San Antonio. ACM, New York, pp 427–437

6. Ihrke I, Magnor M (2004) Image-based tomographic reconstruction of flames. In: ACM SIGGRAPH/Eurographics symposium proceedings, symposium on computer animation, Grenoble, pp 367–375

Opaque

► Opacity

Optic Flow

► Optical Flow

Optical Axis

Peter Sturm
INRIA Grenoble Rhône-Alpes, St Ismier Cedex,
France

Synonyms

Principal axis

Related Concepts

- Center of Projection; ► Image Plane; ► Pinhole Camera Model

Definition

For rotationally symmetric imaging systems, the optical axis is the (usually unique) axis of symmetry.



Background

The usual definition of optical axis in computer vision is tied to the pinhole camera model, where it is defined as the line passing through the center of projection and that is orthogonal to the image plane. For a perfectly aligned usual optical system and where the image plane is parallel to the lenses' principal planes, the optical axis is the line passing through the lenses' centers.

Theory

A more general definition is to consider the optical axis as the (unique) axis of rotational symmetry of an imaging system, if this symmetry exists. Here, it may be advisable to examine rotational symmetry of only the optical part of the image formation process and not the entire process from 3D to the pixel domain. This is meant to exclude the way the photosensitive surface samples the incoming light.

Let us examine this with the aid of the pinhole camera model as example. Suppose that the aspect ratio equals 1, that is, pixel density is identical in vertical and horizontal directions on the image plane. Then, the projection function from the 3D scene to 2D pixel coordinates is rotationally symmetric relative to the pinhole camera's optical axis and the image plane's principal point. However, if the aspect ratio deviates from 1, the rotational symmetry is broken. If one looks at what happens before light rays hit the image plane, then the projection function is still rotationally symmetric though.

A similar example is a camera that exhibits distortions due to the image plane not being perfectly parallel to the lenses' principal planes. Again, the full mapping from 3D to 2D pixel coordinates is not rotationally symmetric here.

To consider rotational symmetry and thus the existence of an optical axis, one may thus consider the function that maps light rays coming into the camera, to rays leaving the optics towards the image plane. In that respect, rotational symmetry and the concept of optical axis are also defined for noncentral cameras, that is, cameras without a single center of projection, like some catadioptric cameras.

References

1. Hecht E (2001) Optics, 4th edn. Addison Wesley, Reading
2. Geissler P (2000) Imaging optics. In: Jähne B, Haußecker H (eds) Computer vision and applications. Academic press, San Diego, pp 53–84

Optical Center

- [Center of Projection](#)
-

Optical Flow

Thomas Brox
Department of Computer Science, University of Freiburg, Freiburg, Germany

Synonyms

- [Optic flow](#)

Related Concepts

- [Rigid Registration](#)

Definition

Optical flow is the vector field that describes the perceived motion of points in the image plane.

Background

When working with image sequences, analyzing the change of the image over time provides valuable information about the scene. The motion of points in the image, as described by the optical flow field, is an important cue in many computer vision tasks, such as tracking, motion segmentation, and structure-from-motion.

Estimating the optical flow from pairs of images is a classical computer vision task. The problem is approached by matching certain image structures, which are assumed to be more or less stable over



time, between two successive images. While earlier estimation methods could only provide sparse optical flow fields, nowadays it is common to estimate a dense optical flow field that provides a displacement vector for each pixel in the first image, describing where this pixel went in the second image.

Optical flow estimation is closely related to the problem of nonrigid registration. The main difference is that the latter problem often assumes a one-to-one mapping between two images. This assumption is not satisfied for optical flow due to occlusion.

Optical Flow Estimation

For estimating the optical flow field, a matching criterion is needed. In most cases this matching criterion is based on the pixel gray value or color, which leads to the so-called optical flow constraint:

$$I(x+u(x, y), y+v(x, y), t+1) - I(x, y, t) = 0, \quad (1)$$

where I denotes the image sequence and $w = (u, v)^\top$ the optical flow field. As this constraint is nonlinear in u and v , it is usually linearized to yield

$$I_x u + I_y v + I_t = 0, \quad (2)$$

where subscripts denote partial derivatives.

For each pixel, the optical flow constraint yields one equation. Since we have two unknowns per pixel to estimate, the problem is underconstrained without additional assumptions. This is also known as the aperture problem.

There are two solutions to this problem. One is to assume a parametric flow model in a fixed neighborhood around each pixel. Choosing the neighborhood (the aperture) large enough, one collects enough information to estimate the model parameters, which determine the flow. This methodology was initially proposed by Lucas and Kanade [1].

The other solution imposes a regularity constraint on the resulting flow field. Rather than estimating flow vectors for each pixel independently, this leads to a global estimation of the whole flow field and is expressed as an energy minimization problem. The first such model has been introduced by Horn and Schunck [2] and reads:

$$\begin{aligned} E(u, v) = & \int (I_x u + I_y v + I_t)^2 \\ & + \alpha (|\nabla u|^2 + |\nabla v|^2) dx dy. \end{aligned} \quad (3)$$

The energy is minimized via variational methods. Since it is a convex functional, the global minimizer $(u, v)^\top$ can be found with standard techniques such as gradient descent or by solving the linear system emerging from the Euler-Lagrange equations.

Numerous extensions of the basic Horn-Schunck model have been presented over the years to deal with several shortcomings of the original model. One important modification is to replace the quadratic penalizers in (Eq. 3) by nonquadratic ones [3, 4]. This can be interpreted as replacing the inherent Gaussian distribution model in (Eq. 3) by one based on robust statistics. Long-tailed distributions allow for outliers in both the optical flow constraint and the smoothness assumption, i.e., occlusion and motion discontinuities are integrated into the model. The most popular choice to date is the use of a regularized Laplace distribution, which leads to the l_1 -norm in the energy functional [5]. As the l_1 -norm is the limit between convex and non-convex penalizers, it allows for the maximum number of outliers while still leading to a convex optimization problem.

The linearization of the optical flow constraint in (Eq. 2) helps solving for the optical flow field, but it includes the assumption that the image is locally linear. In practice this is only true for very small neighborhoods with a few pixels radius and restricts the magnitude of the displacement vectors that can be estimated. In [6] it was suggested to work with the original, nonlinearized optical flow constraint and to postpone the linearization to the numerical scheme. As due to the nonlinearized optical flow constraint the energy becomes nonconvex, local minima are an issue and need to be approached by appropriate heuristics. In [5] this was shown to coincide with the ideas of earlier multiresolution approaches [1, 4]. The numerical scheme in [5] can be regarded as a Gauss-Newton method combined with a continuation strategy to avoid local minima.

Contemporary variational methods for optical flow estimation are based on the following energy functional:

$$E(u, v) = \int \Psi((I(x+u, y+v, t+1) - I(x, y, t))^2) + \alpha \Psi(|\nabla u|^2 + |\nabla v|^2) dx dy, \quad (4)$$

where $\Psi(s^2)$ denotes a robust function, which is often chosen as the regularized l_1 -norm $\Psi(s^2) = \sqrt{s^2 + \epsilon^2}$ with a reasonably small regularization constant ϵ .

Other improvements include extended matching criteria that are invariant to brightness changes [7] and orientation-specific, anisotropic, or nonlocal regularizers [8, 9]. Multigrid solvers and efficient GPU implementations have pushed even advanced optical flow estimation techniques towards real-time performance [10, 11].

Instead of these variational techniques, one can approach optical flow estimation also with combinatorial methods. The most straightforward version of this is simple block matching, which leads to unsatisfactory results though. More enhanced combinatorial methods, e.g., based on belief propagation, perform much better [12, 13].

Nonetheless, variational techniques are currently the dominating way to estimate optical flow. This is very much in contrast to the related task of disparity estimation, where combinatorial methods are much more powerful. This is mainly because disparity estimation is a search problem among an ordered set of labels, which allows for efficient, globally optimal algorithms. In contrast, the two-dimensional nature of optical flow estimation leads to combinatorial problems that are NP-hard. Without the advantage of global optimality, pure combinatorial techniques are inferior to variational techniques due to discretization artifacts and missing subpixel accuracy. Most combinatorial approaches alleviate these problems by running a variational approach as a postprocessing step. Combinatorial techniques are promising in the sense that they can potentially better deal with large motion as well as occlusion. Brox and Malik [14] is an example of combining a combinatorial search (simple block matching) with variational methods to estimate faster motion.

Application

Typical applications of optical flow are tracking, motion segmentation, and action recognition. Tracking

has again many applications in fields like structure-from-motion and surveillance. The most widely spread tracker, the KLT tracker [15], is based on the Lucas-Kanade technique described above. Trackers based on variational optical flow have been proposed as well [16, 17]. In motion segmentation, one can directly cluster the optical flow vectors, as done in the motion layer framework by Wang and Adelson [18]. A joint estimation of optical flow and corresponding motion segments has been proposed in [19, 20]. Histograms of the optical flow can be used for action recognition [21].

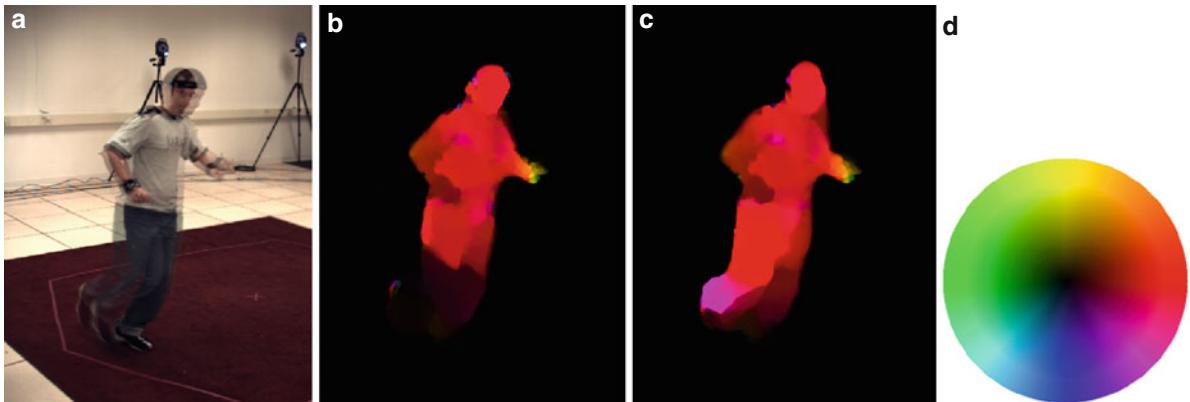
Open Problems

The main remaining problems in optical flow estimation are concerned with occlusion and precise motion boundary estimation, especially in homogenous image areas. While both are implicitly covered by state-of-the-art techniques in areas with rich structure, homogeneous image areas yield the data term to be irrelevant and the estimation results are solely driven by the smoothness assumption, which is a relatively weak prior for such situations.

Experimental Results

A quantitative comparison of a large number of optical flow estimation methods is available at [22]. However, the numbers must be treated with care. As they are computed on the basis of only eight image pairs, overfitting is an issue. More complex methods with many partially hidden parameters have a clear advantage, as more parameters can be specifically adapted to the needs of the benchmark. For this reason, the top-performing methods in the benchmark are not necessarily the best performing ones on a more diverse set of natural videos. Nevertheless, the benchmark allows to coarsely separate techniques that lack certain qualities in the lower part of the table. The computation times, even though they are not directly comparable due to the usage of different hardware, also give hints on the applicability of techniques in practice.

Measuring the quality of optical flow estimation methods is generally hard, because deriving ground truth flow in real videos comes with a very big effort. With the improved rendering quality in computer graphics, large sets of realistic, synthetic test



Optical Flow, Fig. 1 From left to right: (a) Two-frame overlay showing large motion especially at the foot of the person. (b) Optical flow estimated with [5]. The motion of the upper body part is estimated well, but the motion of the leg is missed.

(c) Optical flow estimated with large displacement optical flow [14]. The fast motion can be captured. (d) *Color code* used to represent the optical flow. For instance, red stands for motion to the right

sequences could be a possible way out of this dilemma and will ideally lead to larger benchmark datasets that better cover the space of scenes faced in optical flow applications.

Special properties of certain optical flow estimation techniques can be observed also qualitatively. Figure 1 compares a method that can explicitly deal with larger motion to one that does not have this specific capability.

8. Zimmer H, Bruhn A, Weickert J, Valgaerts L, Salgado A, Rosenhahn B, Seidel HP (2009) Complementary optic flow. In: Proceedings of the 7th international conference on energy minimization methods in computer vision and pattern recognition. Volume 5681 of LNCS. Springer, Berlin/Heidelberg/New York, pp 207–220
9. Werlberger M, Pock T, Bischof H (2010) Motion estimation with non-local total variation regularization. In: International conference on computer vision and pattern recognition, San Francisco
10. Bruhn A (2006) Variational optic flow computation: accurate modelling and efficient numerics. PhD thesis, Faculty of Mathematics and Computer Science, Saarland University, Germany
11. Zach C, Pock T, Bischof H (2007) A duality based approach for realtime TV-L1 optical flow. In: Pattern recognition – proceeding DAGM. Volume 4713 of LNCS. Springer, Heidelberg pp 214–223
12. Shekhovtsov A, Kovtun I, Hlaváč VV (2007) Efficient MRF deformation model for non-rigid image matching. In: International conference on computer vision and pattern recognition (CVPR), Minneapolis
13. Glocker B, Paragios N, Komodakis N, Tziritas G, Navab N (2008) Optical flow estimation with uncertainties through dynamic MRFs. In: International conference on computer vision and pattern recognition (CVPR), Anchorage
14. Brox T, Malik J (2011) Large displacement optical flow: descriptor matching in variational motion estimation. In: IEEE transactions on pattern analysis and machine intelligence 33(3):500–513
15. Shi J, Tomasi C (1994) Good features to track. In: International conference on computer vision and pattern recognition (CVPR), Seattle, pp 593–600
16. Sand P, Teller S (2008) Particle video: long-range motion estimation using point trajectories. Int J Comput Vis 80(1):72–91
17. Sundaram N, Brox T, Keutzer K (2010) Dense point trajectories by GPU-accelerated large displacement optical

References

1. Lucas B, Kanade T (1981) An iterative image registration technique with an application to stereo vision. In: Proceedings of the seventh international joint conference on artificial intelligence, Vancouver, pp 674–679
2. Horn B, Schunck B (1981) Determining optical flow. Artif Intell 17:185–203
3. Black MJ, Anandan P (1996) The robust estimation of multiple motions: parametric and piecewise smooth flow fields. Comput Vis Image Underst 63(1):75–104
4. Mémin E, Pérez P (1998) Dense estimation and object-based segmentation of the optical flow with robust techniques. IEEE Trans Image Process 7(5):703–719
5. Brox T, Bruhn A, Papenberg N, Weickert J (2004) High accuracy optical flow estimation based on a theory for warping. In: European conference on computer vision (ECCV). Volume 3024 of LNCS. Springer, Berlin/New York, pp 25–36
6. Alvarez L, Weickert J, Sánchez J (2000) Reliable estimation of dense optical flow fields with large displacements. Int J Comput Vis 39(1):41–56
7. Papenberg N, Bruhn A, Brox T, Didas S, Weickert J (2006) Highly accurate optic flow computation with theoretically justified warping. Int J Comput Vis 67:141–158

- flow. In: European conference on computer vision (ECCV). LNCS. Springer, Berlin/New York
18. Wang JYA, Adelson EH (1994) Representing moving images with layers. *IEEE Trans Image Process* 3(5): 625–638
 19. Weiss Y (1997) Smoothness in layers: motion segmentation using nonparametric mixture estimation. In: International conference on computer vision and pattern recognition (CVPR), San Juan, Puerto Rico, pp 520–527
 20. Cremers D, Soatto S (2005) Motion competition: a variational framework for piecewise parametric motion segmentation. *Int J Comput Vis* 62(3):249–265
 21. Efros A, Berg A, Mori G, Malik J (2003) Recognizing action at a distance. In: International conference on computer vision, Nice, pp 726–733
 22. Middlebury optical flow benchmark. vision.middlebury.edu

Optimal Estimation

Kenichi Kanatani

Professor Emeritus of Okayama University,
Okayama, Japan

Synonyms

[Optimal parameter estimation](#)

Related Concepts

► [Maximum Likelihood Estimation](#)

Definition

Optimal estimation in the computer vision context usually refers to estimating the parameters that describe the underlying problem from noisy observation. The estimation is done according to a given criterion of optimality, for which maximum likelihood is widely accepted. If Gaussian noise is assumed, it reduces to minimizing the Mahalanobis distance. If furthermore the Gaussian noise has a homogeneous and isotropic distribution, the procedure reduces to minimizing what is called the reprojection error.

Background

One of the central tasks of computer vision is the extraction of 2-D/3-D geometric information from

noisy image data. Here, the term *image data* refers to values extracted from images by image processing operations such as edge filters and interest point detectors. Image data are said to be *noisy* in the sense that image processing operations for detecting them entail uncertainty to some extent.

For optimal estimation, a statistical model of observation needs to be introduced. Let $\mathbf{x}_1, \dots, \mathbf{x}_N$ be the observed image data. The standard model is to view each datum \mathbf{x}_α as perturbed from its true value $\bar{\mathbf{x}}_\alpha$ by $\Delta\mathbf{x}_\alpha$, which is assumed to be independent Gaussian noise of mean $\mathbf{0}$ and covariance matrix $V[\mathbf{x}_\alpha]$. Then, maximum likelihood is equivalent to the minimization of the *Mahalanobis distance*

$$I = \sum_{\alpha=1}^N (\bar{\mathbf{x}}_\alpha - \mathbf{x}_\alpha, V[\mathbf{x}_\alpha]^{-1}(\bar{\mathbf{x}}_\alpha - \mathbf{x}_\alpha)), \quad (1)$$

with respect to the true values $\bar{\mathbf{x}}_\alpha$ subject to given knowledge about them. Hereafter, (\mathbf{a}, \mathbf{b}) denotes the inner product of vectors \mathbf{a} and \mathbf{b} .

If the noise is homogeneous and isotropic, in which case $V[\mathbf{x}_\alpha] = c\mathbf{I}$ for all α for some constant c and the unit matrix \mathbf{I} , the Mahalanobis distance I is equivalent to the sum of the squares of the geometric distances between the observations \mathbf{x}_α and their true values $\bar{\mathbf{x}}_\alpha$, often referred to as the *reprojection error*. That name originates from the following intuition: In inferring the 3-D structure of the scene from its projected images, maximum likelihood under homogeneous and isotropic Gaussian noise means *reprojecting* the inferred 3-D structure onto the images and minimizing the square distance between the *reprojection* of the solution and the projection of the scene. Reprojection error minimization is also referred to as *geometric fitting*.

Theory

The estimation procedure depends on the way the knowledge about true values $\bar{\mathbf{x}}_\alpha$ is represented. A typical approach is to introduce some function $\mathbf{g}(\mathbf{t}, \theta)$ to express $\bar{\mathbf{x}}_\alpha$ in a parametric form

$$\bar{\mathbf{x}}_\alpha = \mathbf{g}(\mathbf{t}_\alpha, \theta), \quad (2)$$

where \mathbf{t}_α is a control variable that specifies the identity of the α th datum and θ is an unknown parameter



that specifies the underlying structure. After (2) is substituted, the Mahalanobis distance I becomes a function of θ alone, which is then minimized with respect to θ . This is the standard approach in the traditional statistic estimation framework and also known as *regression*.

This parametric approach, however, is quite limited in computer vision applications. Often, no such knowledge as (2) is available about the true values \bar{x}_α except that they satisfy some implicit equations of the form

$$F^{(k)}(\mathbf{x}, \theta) = 0, \quad k = 1, \dots, L. \quad (3)$$

The unknown parameter θ allows one to infer the 2-D/3-D shape and motion of the objects observed in the images.

This type of estimation leads to some theoretical problems. Usually, no restriction is imposed on the true values \bar{x}_α except that they should satisfy (3). This is called the *functional model*. One could alternatively introduce some statistical model according to which the true values \bar{x}_α are sampled. Then, the model is called *structural*. This distinction is crucial when one considers limiting processes in the following sense. Traditional statistical analysis mainly focuses on the asymptotic behavior as the number of observations increases to ∞ . This is based on the reasoning that the mechanism underlying noisy observations would better reveal itself as the number of observations increases (the law of large numbers) while the number of available data is limited in practice. So, the estimation accuracy vs. the number of data is a major concern. In this light, efforts have been made to obtain a consistent estimator in the sense that the solution approaches its true value in the limit $N \rightarrow \infty$ of the number N of the data.

In computer vision applications, in contrast, one cannot *repeat* observations. One makes an inference that given a single set of images and how many times one applies image processing operations, the result is always the same because standard image processing algorithms are deterministic and no randomness is involved. This is in a stark contrast to conventional statistical problems where observations are viewed as *samples* from potentially infinitely many possibilities and could obtain, by repeating observations, different values originating from unknown, uncontrollable, or unmodeled causes, which is called *noise* as a whole.

In vision problems, the accuracy of inference deteriorates as the uncertainty of image processing operations increases. Thus, the inference accuracy vs. the uncertainty of image operations, which is called *noise* for simplicity, is a major concern. Usually, the noise is very small, often subpixel levels. In light of this observation, it has been pointed out that in image domains the *consistency* of estimators should more appropriately be defined by the behavior in the limit $\sigma \rightarrow 0$ of the noise level σ [1, 3]. The functional model suits this purpose. If the error behavior in the limit of $N \rightarrow \infty$ were to be analyzed, one needs to assume some structural model that specifies how the statistical characteristics of the data depend on N . However, it is difficult to predict the noise characteristics for different N . Image processing filters usually output a list of points or lines or their correspondences along with their confidence values, from which only those with high confidence are used. If a lot of data are to be collected, those with low confidence need to be included, but their statistical properties are hard to estimate, since such data are possibly misdetections. This is the most different aspect of image processing from laboratory experiments, in which any number of data can be collected by repeated trials.

Maximum Likelihood with Implicit Constraints

Maximum likelihood based on the functional model is to minimize (1) subject to implicit constraints in the form of (3). In statistics, maximum likelihood is criticized for its lack of consistency. In fact, estimation of the true values \bar{x}_α , called *nuisance parameters* when viewed as parameters, is not consistent as $N \rightarrow \infty$ in the maximum likelihood framework [6]. However, the lack of consistency has no realistic meaning in vision applications as explained above. On the contrary, maximum likelihood has very desirable properties in the limit $\sigma \rightarrow 0$ of the noise level σ : The solution is *consistent* in the sense that it converges to the true value as $\sigma \rightarrow 0$ and *efficient* in the sense that its covariance matrix approaches a theoretical lower bound as $\sigma \rightarrow 0$ [1, 3].

According to the experience of many vision researchers, maximum likelihood is known to produce highly accurate solutions, and no necessity is felt for

further accuracy improvement. Rather, a major concern is its computational burden because maximum likelihood usually requires complicated nonlinear optimization. The standard approach is to introduce some auxiliary parameters to express each of $\bar{\mathbf{x}}_\alpha$ explicitly in terms of $\boldsymbol{\theta}$ and the auxiliary parameters. After they are substituted back into (1), the Mahalanobis distance I becomes a function of $\boldsymbol{\theta}$ and the auxiliary parameters. Then, this joint parameter space, which usually has very high dimensions, is searched for the minimum. This approach is called *bundle adjustment*, a term originally used by photogrammetrists. This is very time consuming, in particular if one seeks a globally optimal solution by searching the entire parameter space exhaustively.

Linear Reparameterization

In many important vision applications, the problem can be reparameterized to make the functions $F^{(k)}(\mathbf{x}, \boldsymbol{\theta})$ linear in $\boldsymbol{\theta}$ (but generally nonlinear in \mathbf{x}), allowing one to write (3) as

$$(\xi^{(k)}(\mathbf{x}), \boldsymbol{\theta}) = 0, \quad k = 1, \dots, L, \quad (4)$$

where $\xi^{(k)}(\mathbf{x})$ represents a nonlinear mapping of \mathbf{x} . This formalism covers many fundamental problems of computer vision including fitting a parametric curve such as a line, an ellipse, and a polynomial curve to a noisy 2-D point sequence or a parametric surface such as a plane, an ellipsoid, and a polynomial surface to a noisy 3-D point sets and computing the fundamental matrix or the homography from noisy point correspondences over two images. For this type of problem, a popular alternative to bundle adjustment is minimization of a function of $\boldsymbol{\theta}$ alone, called the *Sampson error*. Let us abbreviate $\xi^{(k)}(\mathbf{x}_\alpha)$ to $\xi_\alpha^{(k)}$. The first-order variation of $\xi_\alpha^{(k)}$ by noise is

$$\Delta\xi_\alpha^{(k)} = \mathbf{T}_\alpha^{(k)} \Delta\mathbf{x}_\alpha, \quad \mathbf{T}_\alpha^{(k)} \equiv \left. \frac{\partial \xi^{(k)}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\bar{\mathbf{x}}_\alpha}. \quad (5)$$

Define the covariance matrices of $\xi_\alpha^{(k)}$, $k = 1, \dots, L$, by

$$\begin{aligned} V^{(kl)}[\xi_\alpha] &= E[\Delta\xi_\alpha^{(k)} \Delta\xi_\alpha^{(l)\top}] \\ &= \mathbf{T}_\alpha^{(k)} E[\Delta\mathbf{x}_\alpha \Delta\mathbf{x}_\alpha^\top] \mathbf{T}_\alpha^{(l)\top} \\ &= \mathbf{T}_\alpha^{(k)} V[\mathbf{x}_\alpha] \mathbf{T}_\alpha^{(l)\top}, \end{aligned} \quad (6)$$

where $E[\cdot]$ denotes expectation. The Sampson error that approximates the minimum of the Mahalanobis distance I subject to the constraints in (4) has the form

$$K = \sum_{\alpha=1}^N \sum_{k,l=1}^L W_\alpha^{(kl)}(\xi_\alpha^{(k)}, \boldsymbol{\theta})(\xi_\alpha^{(l)}, \boldsymbol{\theta}), \quad (7)$$

where $W_\alpha^{(kl)}$ is the (kl) element of $(V_\alpha)^{-}$. Here, V_α is the matrix whose (kl) element is

$$V_\alpha = \left((\boldsymbol{\theta}, V^{(kl)}[\xi_\alpha]\boldsymbol{\theta}) \right), \quad (8)$$

where the true data values $\bar{\mathbf{x}}_\alpha$ in the definition of $V^{(kl)}[\xi_\alpha]$ are replaced by their observations \mathbf{x}_α . The operation $(\cdot)_r^{-}$ denotes the pseudoinverse of truncated rank r , (i.e., with all eigenvalues except the largest r replaced by 0 in the spectral decomposition), and r is the rank (the number of independent equations) of (4). The name Sampson error stems from the classical ellipse fitting scheme [8].

The Sampson error (7) can be minimized by various means including the *FNS* (*fundamental numerical scheme*) [2], and the *HEIV* (*heteroscedastic errors-in-variable*) [5]. It can be shown that the exact maximum likelihood solution can be obtained by repeating Sampson error minimization, each time modifying the Sampson error so that in the end the modified Sampson error coincides with the Mahalanobis distance [4]. It turns out that in many practical applications the solution that minimizes the Sampson error coincides with the exact maximum likelihood solution up to several significant digits; usually, two or three rounds of Sampson error modification are sufficient.

It can be shown that the covariance matrix $V[\hat{\boldsymbol{\theta}}]$ of any unbiased estimator $\hat{\boldsymbol{\theta}}$ of $\boldsymbol{\theta}$ satisfies under some general conditions the inequality

$$V[\hat{\boldsymbol{\theta}}] \succ \left(\sum_{\alpha=1}^N \sum_{k,l=1}^L \bar{W}_\alpha^{(kl)} \bar{\xi}_\alpha^{(k)} \bar{\xi}_\alpha^{(l)\top} \right)_r^{-}, \quad (9)$$

where $\bar{\xi}_\alpha^{(k)}$ are the true value of $\xi_\alpha^{(k)}$ and $\bar{W}_\alpha^{(kl)}$ is the value of $W_\alpha^{(kl)}$ defined earlier evaluated for the



true values of $\xi_\alpha^{(k)}$ and θ . The symbol \succ means that the left-hand side minus the right-hand side is positive semidefinite. The right-hand side of (9) is called the *KCR (Kanatani-Cramer-Rao) lower bound* [1, 3]. It can be shown that the covariance matrix of Sampson error minimization solution coincides with this bound in the leading order in the noise level [1, 3].

Algebraic Methods

Sampson error minimization schemes such as FNS and HEIV rely on local search, and the iterations do not always converge unless started from a value sufficiently close to the solution. For accurate initialization of the iterations, various types of algebraic method have been studied. *Algebraic methods* refer to minimizing the *algebraic distance*

$$\begin{aligned} J &= \sum_{\alpha=1}^N \sum_{k=1}^L (\xi_\alpha^{(k)}, \theta)^2 \\ &= \sum_{\alpha=1}^N \sum_{k=1}^L \theta^\top \xi_\alpha^{(k)} \xi_\alpha^{(k)\top} \theta \\ &= (\theta, M \theta), \end{aligned} \quad (10)$$

where the matrix M is defined by

$$M = \sum_{\alpha=1}^N \sum_{k=1}^L \xi_\alpha^{(k)} \xi_\alpha^{(k)\top}. \quad (11)$$

Note that if $W_\alpha^{(kl)}$ in (7) is replaced by the Kronecker delta δ_{kl} , the Sampson error K coincides with the algebraic distance J . Algebraic distance minimization is also referred to as *algebraic fitting* or simply *least squares*.

The algebraic distance J is trivially minimized by $\theta = \mathbf{0}$ unless some scale normalization is imposed on θ . The standard normalization is $\|\theta\| = 1$. A more general class of normalization is the form of $(\theta, N\theta) = 1$, where N is some positive definite or semidefinite symmetric matrix. It is easily seen that the solution is given by solving the generalized eigenvalue problem

$$M\theta = \lambda N\theta, \quad (12)$$

for the smallest generalized eigenvalue λ . The choice of $N = I$ corresponds to the standard normalization. However, the solution depends on N , and it has been reported by many researchers that the choice $N = I$ leads to large statistical bias. For example, the ellipse thus fitted to a point sequence is almost always smaller than the true one.

One naturally asks: What N will maximize the accuracy of the solution? It is widely recognized that the scheme due to Taubin [9] produces a fairly accurate estimate. Recently, it has been found that if N is allowed to be nondefinite, i.e., neither positive or negative (semi)definite, the statistical bias can be eliminated up to second-order noise terms, resulting in a method called *HyperLS* with slightly better performance than the Taubin method [7].

References

- Chernov N, Lesort C (2004) Statistical efficiency of curve fitting algorithms. *Comput Stat Data Anal* 47(4):713–728
- Chojnacki W, Brooks MJ, van den Hengel A, Gawley D (2000) On the fitting of surfaces to data with covariances. *IEEE Trans Pattern Anal Mach Intell* 22(11):1294–1303
- Kanatani K (2008) Statistical optimization for geometric fitting: theoretical accuracy analysis and high order error analysis. *Int J Comput Vis* 80(2):167–188
- Kanatani K, Sugaya Y (2010) Unified computation of strict maximum likelihood for geometric fitting. *J Math Imaging Vis* 38(1):1–13
- Leedan Y, Meer P (2000) Heteroscedastic regression in computer vision: Problems with bilinear constraint. *Int J Comput Vis* 37(2):127–150
- Neyman J, Scott EL (1948) Consistent estimates based on partially consistent observations. *Econometrica* 16(1):1–32
- Rangarajan P, Kanatani K, Niituma H, Sugaya Y (2010) Hyper least squares and its applications. In: Proceedings of the 20th international conference on pattern recognition, August 2010, Istanbul, Turkey
- Sampson PD (1982) Fitting conic sections to “very scattered” data: An iterative refinement of the Bookstein algorithm. *Comput Graph Image Process* 18(1):97–108
- Taubin G (1991) Estimation of planar curves, surfaces, and non-planar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE Trans Pattern Anal Mach Intell* 13(11):1115–1138

Optimal Parameter Estimation

► [Optimal Estimation](#)



Oren-Nayar Reflectance Model

Ping Tan

Department of Electrical and Computer Engineering,
National University of Singapore, Singapore,
Singapore

Related Concepts

► Radiance; ► Reflectance Models

Definition

The Oren-Nayar reflectance model calculates the amount of reflected radiance at a surface point according to the lighting, viewing, and surface normal directions at that point. It is characterized by its high accuracy in modeling diffuse reflection for rough surfaces. It was introduced by Michael Oren and Shree K. Nayar [1] in 1994.

Background

When light arrives at surfaces, it can be reflected, refracted, scattered, or absorbed. Reflectance models are mathematical functions that describe the interactions between light and surfaces. Usually, they are functions of lighting, viewing, and surface normal directions. There are various reflectance models at different levels of precision and complexity. Most reflectance models include components describing diffuse and specular reflection, respectively.

The Oren-Nayar model is a reflectance model that accurately represents the diffuse reflection of rough surfaces. In the field of computer vision, diffuse reflection is often modeled by the Lambert's model [2] which assumes the light is uniformly reflected in all reflected directions. However, many real surfaces, especially rough surfaces such as clay or concrete, exhibit significant non-Lambertian diffuse reflection. For example, the reflection is often stronger when the viewing direction is close to the lighting direction, which is called *backscattering*. Oren and Nayar designed a more accurate model to describe diffuse reflection of rough surfaces. This model can be applied

to generate realistic computer graphic images or to accurately infer scene properties such as shape and material from observed image intensities.

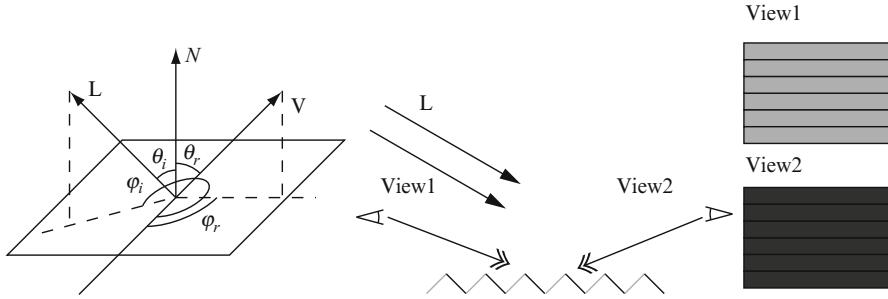
Theory

Light reflected by a surface can be roughly divided into specular and diffuse reflection. The specular reflection refers to the light reflected directly at the surface without entering it. Specular reflection typically is concentrated within a small angle in space. In comparison, diffuse reflection refers to the light that enters the surface and is distributed more uniformly in all reflected directions.

In computer vision, the Lambert's reflectance model is often used to describe diffuse reflection, which assumes that the diffuse reflection is equally distributed on a hemisphere indicating all the reflected directions. However, real surfaces, especially rough surfaces like clay and concrete, often exhibit non-Lambertian reflection. A common non-Lambertian phenomenon is the *backscattering* where more light is reflected toward the incident direction and the surface appears brighter when the viewing direction is closer to the lighting direction. For example, this phenomenon is observed in lunar photometry and is modeled by Opik [3] and Minnaert [4].

Backscattering can be well explained by studying the surface microstructures. As shown on the right of Fig. 1, when the polar angle of the incident light θ_i is large, some of the microstructures are in shadows. When the surface is viewed from the direction of the incident lighting L , all visible microstructures are illuminated and the surface appears brighter. However, when the surface is viewed from the direction V , all visible microstructures are shadowed and the surface appears darker.

Oren and Nayar adopted a microfacet surface model to derive the diffuse reflection of rough surfaces, where surfaces consist of many infinitely long V-cavities and each cavity facet is Lambertian. The reflectance model of a surface is derived by integrating the reflection of all facets. A similar microfacet model is used by the Cook-Torrance reflectance model [5] to study specular reflection, where each microfacet acts as a mirror. The Oren-Nayar model is derived in three steps in the original paper [1]. First, a reflectance model $I^{(1)}$ is derived for anisotropic surfaces consisting of V-cavities of



Oren-Nayar Reflectance Model, Fig. 1 *Left:* directions involved in the definition of the Oren-Nayar model. N is the surface normal direction, and L and V are the lighting and viewing direction, respectively. *Right:* a rough surface consists of many microfacets. When the polar angle of the incident light θ_i is

the same slope and the same direction. This model includes two components, one for direction illumination and the other for interreflection between microfacets. Second, a reflectance model $I^{(2)}$ is derived for isotropic surfaces consisting of V-cavities with the same slope but uniformly distributed directions by integrating $I^{(1)}$ over different cavity directions. Third, the final Oren-Nayar model is derived for general isotropic surfaces that consist of V-cavities whose slopes follow a Gaussian distribution with mean μ and variance σ^2 and whose directions follow a uniform distribution. The Oren-Nayar reflectance model $I^{(3)}$ can be derived by integrating $I^{(2)}$ over different cavity slopes. This model also includes a component for direction illumination

$$I_1^{(3)}(\theta_r, \theta_i, \phi_r - \phi_i; \sigma) = \frac{\rho}{\pi} E_0 \cos \theta_i \left[C_1(\sigma) + \cos(\phi_r - \phi_i) C_2(\alpha; \beta; \phi_r - \phi_i; \sigma) \tan \beta + (1 - |\cos(\phi_r - \phi_i)|) C_3(\alpha; \beta; \sigma) \tan \left(\frac{\alpha + \beta}{2} \right) \right] \quad (1)$$

and a component for microfacet interreflection

$$I_2^{(3)}(\theta_r, \theta_i, \phi_r - \phi_i; \sigma) = 0.17 \frac{\rho^2}{\pi} E_0 \cos \theta_i \frac{\sigma^2}{\sigma^2 + 0.13} \left[1 - \cos(\phi_r - \phi_i) \left(\frac{2\beta}{\pi} \right)^2 \right]. \quad (2)$$

Here, (θ_i, ϕ_i) and (θ_r, ϕ_r) are the polar and azimuth angles of the lighting and viewing directions, respectively, as illustrated on the left of Fig. 1, $\alpha =$

large, some of the microfacets are in shadow and some of them are illuminated, as illustrated by darker and brighter segments. The surface appears brighter in the view 1 than in the view 2, because most of the visible microfacets in view 1 are illuminated and those in view 2 are shadowed

$\max(\theta_r, \theta_i)$, $\beta = \min(\theta_r, \theta_i)$, E_0 indicates the illumination intensity and ρ is the albedo in the Lambert's reflectance model of microfacet. C_1 , C_2 , and C_3 are evaluated according to the following equations:

$$C_1 = 1 - 0.5 \frac{\sigma^2}{\sigma^2 + 0.33}$$

$$C_2 = \begin{cases} 0.45 \frac{\sigma^2}{\sigma^2 + 0.09} \sin \alpha & \text{if } \cos(\phi_r - \phi_i) \geq 0 \\ 0.45 \frac{\sigma^2}{\sigma^2 + 0.09} \left(\sin \alpha - \left(\frac{2\beta}{\pi} \right)^3 \right) & \text{otherwise} \end{cases}$$

$$C_3 = 0.125 \left(\frac{\sigma^2}{\sigma^2 + 0.09} \right) \left(\frac{4\alpha\beta}{\pi^2} \right)^2.$$

Experiments reported in [1] show that the Oren-Nayar reflectance model matches the measured reflectance data from rough surfaces well and represents the *backscattering*. An interesting fact about this model is that it degenerates to the Lambert's model when $\sigma^2 = 0$. In applications, to reduce computation and simplify analysis, a simplified version of the Oren-Nayar model is often desired. The term C_3 and the interreflection are found to be relatively small during simulation. Hence, a simplified model can be obtained by discarding C_3 and ignoring interreflection:

$$I(\theta_r, \theta_i, \phi_r - \phi_i; \sigma) = \frac{\rho}{\pi} E_0 \cos \theta_i (A + B \max[0, \cos(\phi_r - \phi_i)] \sin \alpha \tan \beta) \quad (3)$$

Here, $A = 1.0 - 0.5 \frac{\sigma^2}{\sigma^2 + 0.33}$ and $B = 0.45 \frac{\sigma^2}{\sigma^2 + 0.09}$.



Application

The Oren-Nayar reflectance model computes reflected radiance according to the 3D shape, viewing, and lighting configurations. Like many other reflectance models, it can be used to create computer graphic images. It is shown in [1] that the Oren-Nayar reflectance model can generate images of rough surfaces more realistically than the Lambert's model. This reflectance model was also applied for photometric stereo in [6] and generated better results than the Lambertian photometric stereo algorithm.

References

1. Oren M, Nayar SK (1994) Generalization of lambert's reflectance model. In: SIGGRAPH '94: proceedings of the 21st annual conference on computer graphics and interactive techniques. ACM, New York, pp 239–246
2. Lambert JH (1760) Photometria sive de mensure de gratibus lumi-nis, colorum umbrae. Eberhard Klett
3. Opik E (1924) Photometric measures of the moon and the moon the earth-shine. Publications de L'Observatoire Astronomical de L'Universite de Tartu 26(1):1–68
4. Minnaert M (1941) The reciprocity principle in lunar photometry. *Astrophys J* 93:403–C410
5. Cook RL, Torrance KE (1981) A reflectance model for computer graphics. In: SIGGRAPH '81: proceedings of the 8th annual conference on computer graphics and interactive techniques. ACM, New York, pp 307–316
6. Oren M, Nayar SK (1995) Generalization of the lambertian model and implications for machine vision. *Int J Comput Vis* 14(3):227–251

Osculating Paraboloids

Jan J. Koenderink

Faculty of EEMSC, Delft University of Technology,
Delft, The Netherlands
The Flemish Academic Centre for Science and the
Arts (VLAC), Brussels, Belgium
Laboratory of Experimental Psychology, University
of Leuven (K.U. Leuven), Leuven, Belgium

Synonyms

Osculating quadric; Second order approximation

Related Concepts

►Curvature; ►Differential Invariants

Definition

Osculating paraboloids commonly occur in second-order approximations where the first order is irrelevant or can be transformed away.

Background

Despite the common occurrence of osculating paraboloids in many settings, there appears to be no literature dedicated to their shape space. Although the standard taxonomy of quadrics in Euclidean space is familiar to most, what is missing is the geometrical structure of the manifold of all osculating paraboloids, inclusive a metric. Here the two-parameter case (important in many applications) is discussed in a little detail.

Theory

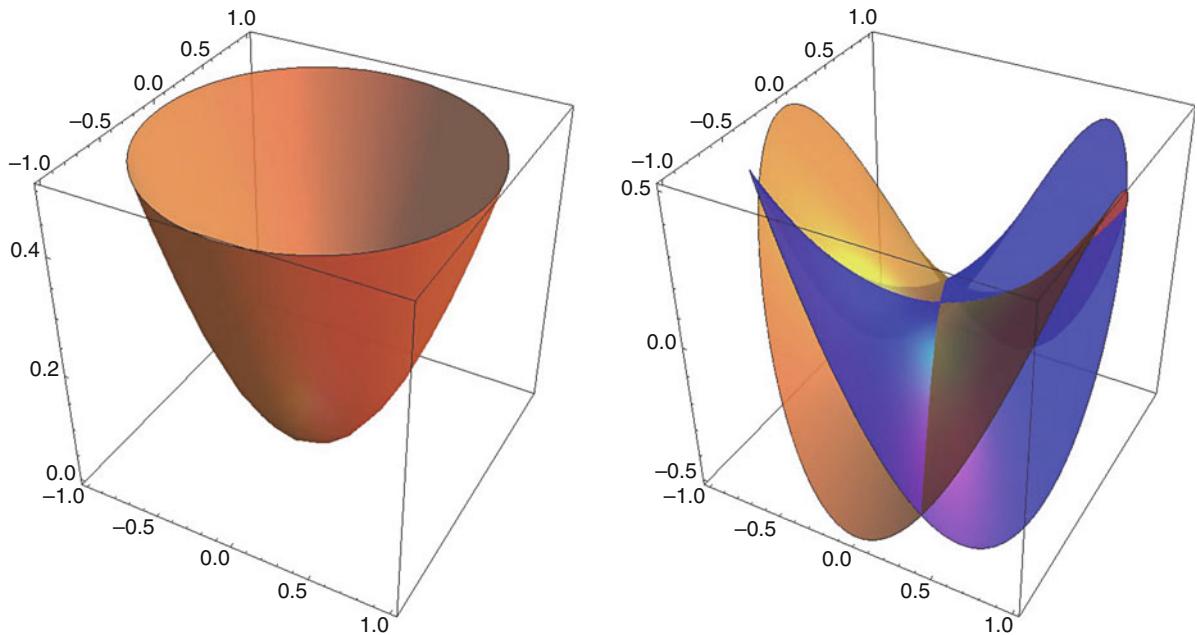
“Osculating paraboloids” are second-order Monge patches:

$$\begin{aligned} x\mathbf{e}_x + y\mathbf{e}_y + z(x, y)\mathbf{e}_z &= x\mathbf{e}_x \\ &+ y\mathbf{e}_y + \frac{1}{2}(a_{20}x^2 + 2a_{11}xy + a_{02}y^2)\mathbf{e}_z, \end{aligned} \quad (1)$$

that occur in many contexts. Apparently they inhabit a three-dimensional quadric shape space that may be parameterized by the coefficient triple $\{a_{20}, a_{11}, a_{02}\}$.

Typically the z -domain will be some physical parameter with a physical dimension different from that of the xy -domain. The appropriate setting then is not Euclidean. In most applications one treats the z -dimension as isotropic, then the space is a “singly isotropic space,” or “graph space.”

Scaling and addition in this space correspond to point-wise addition of heights, thus is well defined and makes geometrical sense. The quadric shape space is a linear space under addition and multiplications with real numbers. Thus it is of some interest to find a basis that makes intuitive, and/or pragmatic sense.



Osculating Paraboloids, Fig. 1 The basis of osculating quadrics. At *left* the isotropic paraboloid $(x^2 + y^2)/2$, at *right* the anisotropic paraboloids xy and $(x^2 - y^2)/2$, these are mutually congruent

One lead is that all such surfaces that differ only by a rotation about the z-axis (\mathbf{e}_z) are geometrically congruent and may often be grouped as a single “shape.” This leads one to consider the isotropic paraboloid:

$$\frac{x^2 + y^2}{2}, \quad (2)$$

to be “special,” since it is invariant under such rotations. Moreover, the pair:

$$xy, \quad \frac{x^2 - y^2}{2}, \quad (3)$$

is likewise “special” because they are the same under a rotation over $\frac{\pi}{4}$ (notice that all quadrics transform into themselves under rotations over π). These shapes transform into their “negatives” under rotations of $\frac{\pi}{2}$, thus, in a sense, they are their own negatives. This is not the case in general, for instance, $x^2 + y^2$ cannot be transformed into its negative (i.e., $-(x^2 + y^2)$) under any rotation.

Thus the basis of quadrics contains only two distinct shapes (Fig. 1), rather than three as one might naively

expect. Using this basis one evidently has:

$$\begin{aligned} & \frac{1}{2}(a_{20}x^2 + 2a_{11}xy + a_{02}y^2) \\ &= r \frac{x^2 - y^2}{2} + sxy + t \frac{x^2 + y^2}{2}, \end{aligned} \quad (4)$$

with:

$$r = \frac{a_{20} - a_{02}}{2}, \quad s = a_{11}, \quad t = \frac{a_{20} + a_{02}}{2}. \quad (5)$$

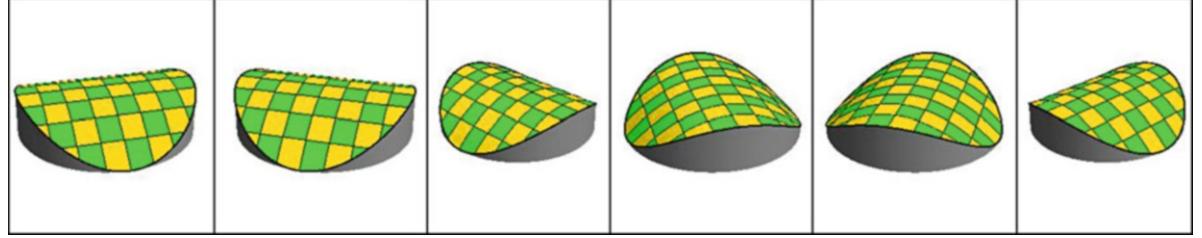
By a suitable rotation any quadric may be written in the canonical form

$$\frac{1}{2}(\kappa_1 u^2 + \kappa_2 v^2), \text{ with } \kappa_1 \geq \kappa_2. \quad (6)$$

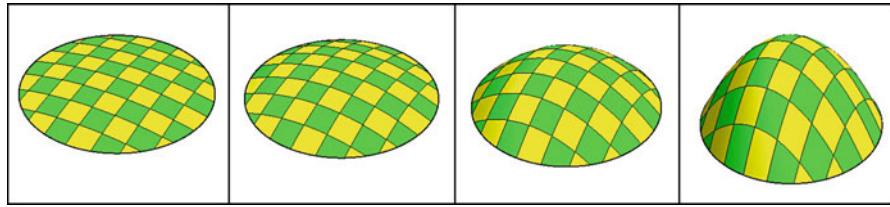
Such a canonical form is convenient because it abstracts away from the orientation about the z-axis (Fig. 2). The general quadric is rotated by angle φ with respect to the u-axis (the first principal direction), where $-\frac{\pi}{2} < \varphi < \frac{\pi}{2}$. One has:

$$a_{20} = \frac{\kappa_1 + \kappa_2}{2} + \frac{\kappa_1 - \kappa_2}{2} \cos 2\varphi, \quad (7)$$

$$a_{11} = \frac{\kappa_1 - \kappa_2}{2} \sin 2\varphi, \quad (8)$$



Osculating Paraboloids, Fig. 2 A suite of surfaces of the same Casorati curvature and shape parameter but different orientations. The orientation domain is periodic with period π



Osculating Paraboloids, Fig. 3 A scale of surfaces of the same shape, the Casorati curvature varying by factors of two

$$a_{02} = \frac{\kappa_1 + \kappa_2}{2} - \frac{\kappa_1 - \kappa_2}{2} \cos 2\varphi, \quad (9)$$

and, equivalently:

$$r = \frac{\kappa_1 - \kappa_2}{2} \cos 2\varphi, \quad (10)$$

$$s = \frac{\kappa_1 - \kappa_2}{2} \sin 2\varphi, \quad (11)$$

$$t = \frac{\kappa_1 + \kappa_2}{2}. \quad (12)$$

Introduce the Casorati curvature (see Fig. 3) as:

$$\kappa = \sqrt{\frac{\kappa_1^2 + \kappa_2^2}{2}}, \quad (13)$$

and the shape parameter (see Fig. 4) as:

$$\sigma = \arctan \frac{\kappa_1 + \kappa_2}{\kappa_1 - \kappa_2}. \quad (14)$$

These differential invariants can be written into various forms, e.g.,

$$\kappa = \sqrt{\frac{a_{20}^2 + 2a_{11}^2 + a_{02}^2}{2}} = \sqrt{r^2 + s^2 + t^2}, \quad (15)$$

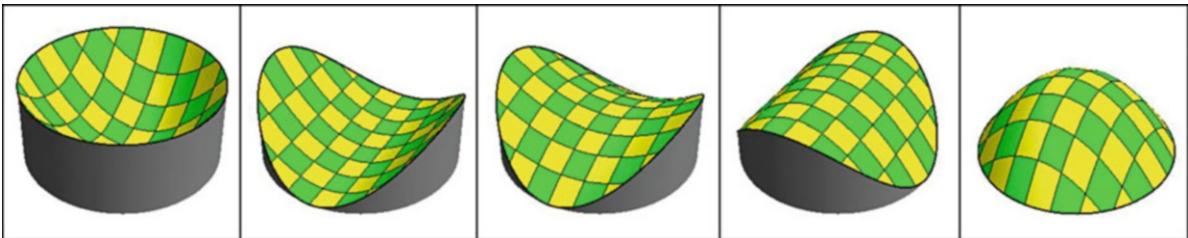
and

This can be further simplified by identifying the mean curvature $2H = \kappa_1 + \kappa_2 = a_{20} + a_{02}$, the Gaussian curvature $K = \kappa_1 \kappa_2 = a_{20} a_{02} - a_{11}^2$, and the “bending energy” $E = \kappa_1^2 + \kappa_2^2 = a_{20}^2 + 2a_{11}^2 + a_{02}^2$. (Notice that H and K should be distinguished from the invariants of the same name in Euclidean differential geometry. They correspond to the case of infinitesimal height, or to isotropic geometry.) The expression $\frac{1}{2} \sqrt{r^2 + s^2 + t^2}$ captures the anisotropy of the quadric and may well be denoted its “non-umbilicity.”

The Casorati curvature is perhaps less well known, and it may be motivated as follows. One has:

$$\kappa = \sqrt{\frac{\kappa_1^2 + \kappa_2^2}{2}}, \quad (17)$$

and thus κ vanishes only for the planar case. This is different for the mean curvature $2H = \kappa_1 + \kappa_2$, which vanishes for minimal surfaces ($t = 0$), or the Gaussian curvature $K = \kappa_1 \kappa_2$, which vanishes for cylindrical surfaces. This is often confusing to the beginner, for whom minimal surfaces and cylinders are evidently “curved.” Moreover, one easily verifies that when the “spatial average” and “standard deviation” of a function $F(x, y)$ are defined as:



Osculating Paraboloids, Fig. 4 A suite of surfaces of the same Casorati curvature, the shape parameter taking values of $+\pi/2$, $+\pi/4$, 0 , $-\pi/4$, and $-\pi/2$

$$\langle F(x, y) \rangle_\mu = \int_{\mathbb{R}^2} F(x, y) \left(\frac{e^{-\frac{x^2+y^2}{2\mu^2}}}{2\pi\mu^2} \right) dx dy, \quad (18)$$

and $[F]_\mu = \sqrt{\langle F^2 \rangle_\mu - \langle F \rangle_\mu^2}$,

one has:

$$\kappa = \frac{[\kappa_1 x^2 + \kappa_2 y^2]}{[x^2 + y^2]} \quad (19)$$

(where $\mu > 0$ has fallen out of the equation). Thus the Casorati curvature is essentially *the root mean square deviation from planarity*.

The shape parameter σ is best understood as a measure of the ratio of umbilicity (isotropy) to non-umbilicity (anisotropy) of the shape. Shapes that differ only in the sign of the shape parameters stand in the relation as a cast to its mold. The minimal shapes ($\sigma = 0$) “are their own molds” as they can be fitted into their negatives after a rotation over $\frac{\pi}{2}$.

Now consider polar coordinates in shape space, defined as:

$$\varrho = \sqrt{r^2 + s^2 + t^2} = \kappa \quad (20)$$

$$\vartheta = \arctan \frac{t}{\sqrt{r^2 + s^2}} = \sigma, \quad (21)$$

$$\phi = \arctan \frac{s}{r} = 2\varphi. \quad (22)$$

The spheres concentric with the origin are loci of constant Casorati curvature, and the origin represents the planar case. A “shape” is a right circular cone with the t-axis as its axis. The t-axis itself is such a (degenerated) cone, it represents the umbilics. The rs-plane is also such a (again, degenerated) cone, it represents the minimal surfaces, which are the surfaces of zero mean curvature. The cone with semi-top angle of $\frac{\pi}{2}$ is the locus of parabolic (cylindrical) surfaces. Each half-plane on the t-axis houses the complete zoo of shapes

up to orientation. A meridian contains all shapes of the same curvature, a latitude circle a single shape in all orientations. Thus one obtains a complete overview of all quadric shapes in a very natural parameterization (Figs. 5–8).

On a general curved surface the shape of the local osculating paraboloids will change from point to point (see Figs. 5–7). Since the change will be smooth, the surface can be mapped on a surface in shape space, a surface that may well be expected to have self-intersections and perhaps not everywhere smooth (e.g., have edges of regression or swallowtails), but will be continuous. This allows one to draw a number of useful conclusions concerning generic surfaces, e.g.:

- Planar points do not occur.
- Umbilics are isolated points.
- As a surface is deformed umbilics come and go in pairs.
- Parabolic curves occur on curves; on a closed surface they are closed curves.
- Minimal points occur on curves in hyperbolic areas; on a closed surface they are closed curves.
- Convex and concave regions are mutually isolated through hyperbolic areas.

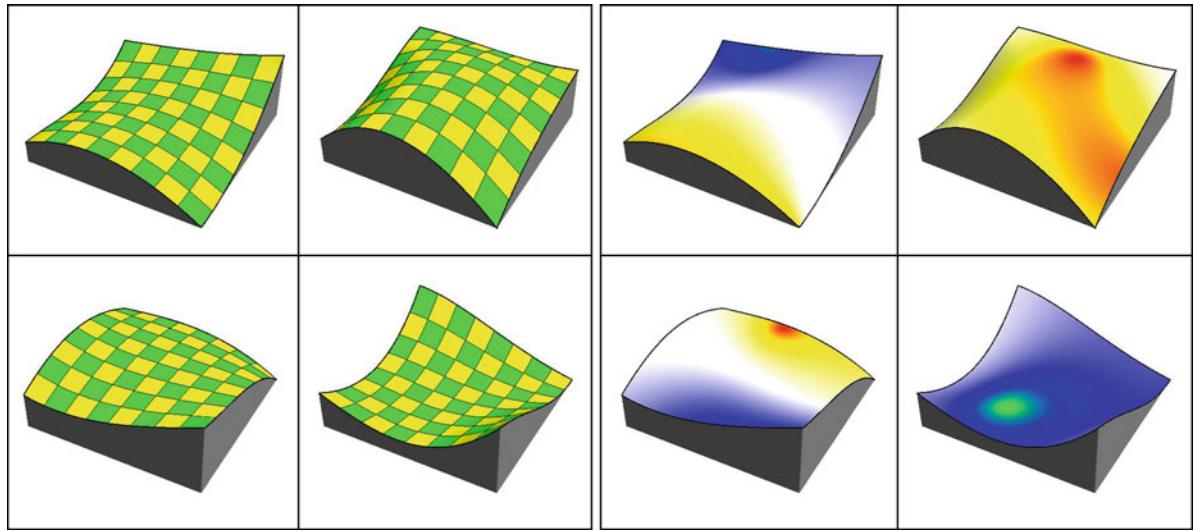
The local structure of these surfaces naturally involves the osculating cubics, which can be written as follows:

$$C(x, y) = \frac{1}{2!}(a_{20}x^2 + 2a_{11}xy + a_{02}y^2) + \frac{1}{3!}(a_{30}x^3 + 3a_{21}x^2y + 3a_{12}xy^2 + a_{03}y^3). \quad (23)$$

In an infinitesimal neighborhood of the origin one finds:

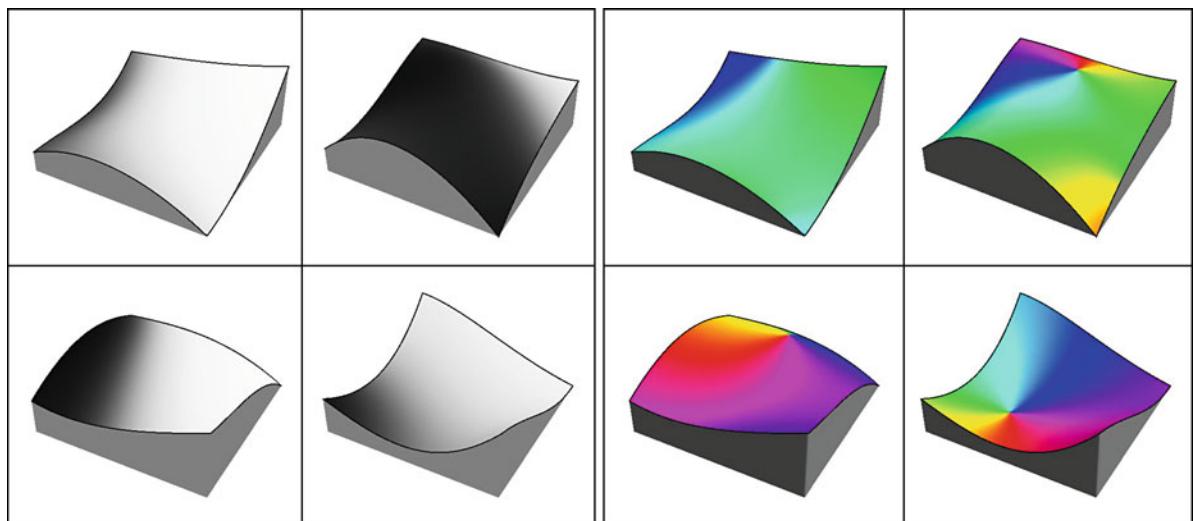
$$a'_{20}(x, y) = a_{20} + a_{30}x + a_{21}y, \quad (24)$$

$$a'_{11}(x, y) = a_{11} + a_{21}x + a_{12}y, \quad (25)$$



Osculating Paraboloids, Fig. 5 At left plots of four random surfaces. At right the surfaces have been color coded with the shape parameter. The color scale uses Hering's "opposite colors" (G: *Gegenfarben*). The umbilics are red (convex) and green

(concave), the parabolic points *yellow* (ridge) and *blue* (rut), whereas the minimal points (zero mean curvature, symmetric saddles) are *white*



Osculating Paraboloids, Fig. 6 At left the surfaces of Fig. 5 left have been shaded with the Casorati curvature. The gray scale represents a nonlinear monotonic function of the

logarithm of the curvature. At right the surfaces have been colored with the orientation of the direction of largest principal curvature. Notice the singularities at umbilical points

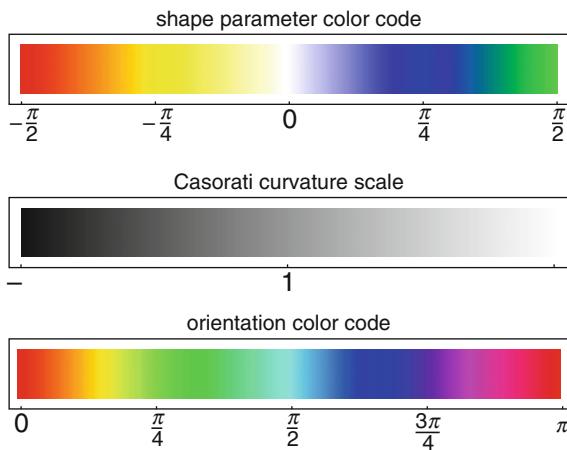
$$a'_{02}(x, y) = a_{02} + a_{12}x + a_{03}y. \quad (26)$$

Thus the quadric is perturbed by a linear combination of the two vectors in $\{r, s, t\}$ -space:

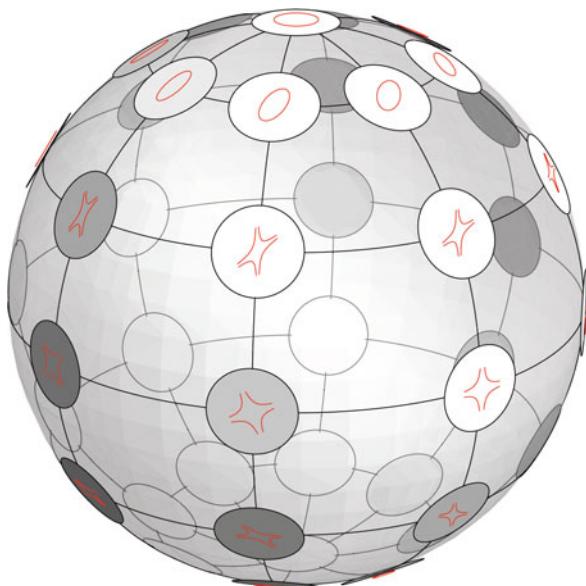
$$\frac{\partial}{\partial x}\{r, s, t\} = \frac{1}{2}\{a_{30} - a_{12}, 2a_{21}, a_{30} + a_{12}\} \quad (27)$$

$$\frac{\partial}{\partial y}\{r, s, t\} = \frac{1}{2}\{a_{21} - a_{30}, 2a_{12}, a_{03} + a_{21}\} \quad (28)$$

with weights x and y . These tangent vectors span a planar element in $\{r, s, t\}$ -space. The planar element will in general be nondegenerate, the condition being that the two tangent vectors are independent, implying the



Osculating Paraboloids, Fig. 7 Scales used to indicate the parameters of osculating quadrics; these scales are used in Figs. 5 and 6



Osculating Paraboloids, Fig. 8 A sphere of constant Casorati curvature. The shapes have been indicated by their Dupin indicatrices. The equator has all orientations of the symmetric saddle, and each meridian has all shape indices with the umbilics at the poles. Each latitude circle repeats a shape at all orientations

cubic indicatrix of Dupin to have a single branch. Thus a neighborhood of a generic surface point maps on a surface in shape space. This allows one to infer various generic properties in a most simple manner, for instance that generic umbilics will be isolated points, the locus of parabolic points will be a curve on the surface, and so forth.

The osculating quadrics shape space may also be used to measure shape differences, a likely metric being:

$$d\ell^2 = \frac{dr^2 + ds^2 + dt^2}{r^2 + s^2 + t^2}, \quad (29)$$

which is invariant with respect to rotations and homotheties about the origin. The metric can be rewritten as:

$$d\ell^2 = (d\log \kappa)^2 + d\sigma^2 + \sin^2 \sigma d\phi^2, \quad (30)$$

from which one sees that the geodesics are planar logarithmic spirals in planes through the origin. Lines through the origin and circles with the center at the origin are degenerated cases. For shapes of the same Casorati curvature the distance is simply a spherical arc length; in case the shapes have the same orientation this becomes the shape parameter, and for minimal surfaces it is the orientation difference. For shapes of the same shape parameter and orientation the distance is the logarithm of the ratio of Casorati curvatures, thus independent of the unit of length (or absolute size).

The osculating paraboloids shape space has numerous applications in very diverse contexts.

Open problem

The shape space concept described here appears to be little known and there is a decided lack of useful literature. The Casorati curvature and shape parameter appear in the literature as “curvedness” and (in a scaled version) as “shape index.”

References

1. Griffin LD (2007) The 2nd order local-image–structure solid. *IEEE Trans Pattern Anal Mach Intell* 29(8):1355–1366
2. Koenderink JJ (1990) Solid shape. MIT, Cambridge, MA
3. Koenderink JJ, van Doorn AJ (1992) Surface shape and curvature scales. *Image Vis Comput* 10(8):557–564

Osculating Quadric

- ▶ [Osculating Paraboloids](#)

Out of Focus Blur

- ▶ [Defocus Blur](#)

P

Panoramic Camera

- ▶ Omnidirectional Camera

Panoramic Image Generation

- ▶ Video Mosaicing

Panoramic Stitching

- ▶ Image Stitching

Pan-Tilt Camera Calibration

- ▶ Active Calibration

Pan-Tilt-Zoom (PTZ) Camera

Sudipta N. Sinha
Microsoft Research, Redmond, WA, USA

Synonyms

IP camera; Network camera; Surveillance camera

Related Concepts

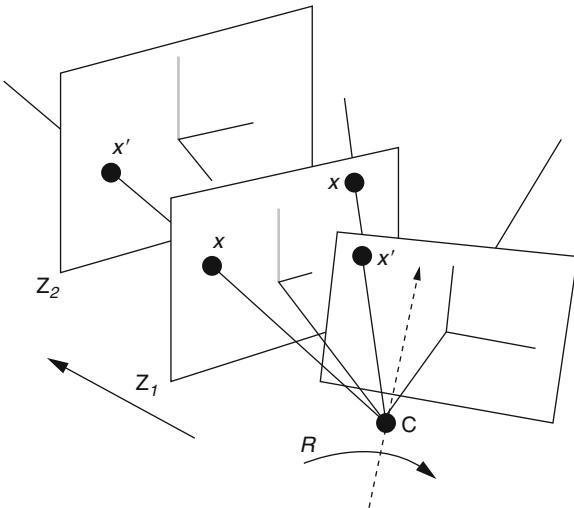
- ▶ Camera Calibration

Definition

A pan-tilt-zoom (PTZ) camera typically refers to an active camera which has some degree of pan, tilt, and zoom control. They are commonly used to monitor large areas for visual surveillance applications. The pan, tilt, and zoom controls of most off-the-shelf PTZ cameras are often programmable, enabling the camera to be remotely controlled over a network. Some variants of PTZ cameras are called *network* cameras when they come equipped with a real-time operation system which makes it possible to stream video over a network in real time. An example of such a camera is the Canon VB-C60 (see http://www.usa.canon.com/app/pdf/nvideo/VB-C60_Product_Sheet.pdf).

Background

Active PTZ cameras serve as a practical alternative to high-resolution omnidirectional cameras in wide-area surveillance systems. On one hand, flexible pan and tilt ranges provide most PTZ cameras a large *effective* field of view (FOV) similar to an omnidirectional sensor, but in addition to that, these cameras can also zoom in on a small region of interest and capture it in high resolution. However, unlike omnidirectional sensors, PTZ cameras cannot simultaneously observe the complete scene. Since activities of interest in a visual surveillance scenario typically occur in small regions within a large area, simultaneous imaging is not always required. The inability to simultaneously observe the complete scene is often compensated by deploying a network of static and PTZ cameras.



Pan-Tilt-Zoom (PTZ) Camera, Fig. 1 2D feature point correspondences under camera rotation and zoom

In order to utilize a programmable PTZ camera, many systems require the knowledge of a camera model that explains how 3D points in the world project to the image plane of a PTZ camera given a specific pan, tilt, and zoom configuration. Analyzing images from PTZ cameras may also require backprojecting a pixel on the image plane to obtain its corresponding 3D viewing ray (parameterized by the pan-tilt angles) on the camera's viewing sphere or the corresponding 3D ray in the world coordinate frame, provided the camera position is also known. For example, consider a calibrated PTZ camera monitoring a car parking area. For a calibrated camera, pixels can be mapped to precise locations on the ground plane. The model also makes it possible to actively track a person in the scene while ensuring that the PTZ camera is able to center its view on the individual. When a scene is monitored by a network of PTZ cameras, a calibrated camera model makes it possible to infer associations between objects or events detected in video captured from different viewpoints.

Conventional offline camera calibration methods such as [1] cannot easily be used to calibrate PTZ cameras since both the observed scene as well as the inter-camera baselines can be quite large. This precludes the use of conventional calibration objects which are typically too small for such large scenes [3, 4]. Also an active camera's calibration parameters must be continually estimated online by refining the pan-tilt angle and

focal length estimates using a closed-loop mechanism [2] instead of relying on a set of precomputed pan-tilt control settings computed offline. This is due to the fact that the pan-tilt controllers present in most off-the-shelf PTZ cameras can be imprecise during operation.

Theory

PTZ Camera Model: In many cases, PTZ cameras can be modeled using a simple motion model where the pan and tilt rotation axes are assumed to pass through the center of projection of the camera. However, depending on the camera's mechanical assembly and design, the pan and tilt rotation axes may need to be modeled as arbitrary axes not passing through the projection center [5]. In either case, the 2D projection of a 3D point onto the image plane can be computed using a few matrix operations. For indoor scenes, the intrinsic and extrinsic parameters of a network of pan-tilt cameras can be estimated offline by tracking a single moving LED over time within the working volume [5].

In outdoor scenes, the simple model works fine since the deviation of the rotation axes from the projection center is negligible compared to the average depth in the scene. This allows the PTZ camera to be modeled as a purely rotating and zooming camera for which self-calibration algorithms are well known [6]. A PTZ camera is then treated like a pinhole camera with a fixed projection center, but the camera intrinsics is modeled as a function of zoom and the camera's orientation obviously depends on the camera's pan and tilt settings.

Using homogeneous coordinates to represent a 3D world point and the corresponding image point denoted as \mathbf{X} and \mathbf{x} , respectively, the imaging process can then be linearly modeled as follows:

$$\mathbf{x} = \mathbf{K}_z \mathbf{R}_{pt} [\mathbf{I}] - \mathbf{C} \mathbf{X} \quad (1)$$

where \mathbf{K}_z denotes the 3×3 camera *intrinsic* matrix, \mathbf{R}_{pt} denotes the 3×3 rotation matrix, and \mathbf{C} denotes the fixed projection center of the camera in world coordinates. The subscripts z , p , and t refer to the zoom, pan, and tilt settings of the active camera. The rigid transformation from world coordinates to camera coordinates is defined by the rotation matrix \mathbf{R}_{pt}

and translation vector $\mathbf{t}_{pt} = -\mathbf{R}_{pt}\mathbf{C}$ which are often referred to as the camera's *extrinsic* parameters. \mathbf{K}_z depends on the zoom setting z and is defined in terms of five intrinsic parameters – α , s , f_z , u_z , and v_z . α is the pixel aspect ratio (an unknown constant), s is a skew parameter (typically set to 0), f_z is the focal length measured in pixel, and (u_z, v_z) is the principal point in the image plane:

$$\mathbf{K}_z = \begin{pmatrix} \alpha f_z & s & u_z \\ 0 & f_z & v_z \\ 0 & 0 & 1 \end{pmatrix} \quad (2)$$

Rotating and Zooming Cameras: Let \mathbf{x} and \mathbf{x}' be the 2D projections of a 3D point \mathbf{X} for two different PTZ configurations (see Fig. 1). Based on Eq. 1, $\mathbf{x} = \mathbf{K}[\mathbf{R} \; \mathbf{t}]\mathbf{X}$ and $\mathbf{x}' = \mathbf{K}'[\mathbf{R}' \; \mathbf{t}]\mathbf{X}$. Selecting \mathbf{C} as the world origin results in $\mathbf{t} = \mathbf{0}$ and then eliminating \mathbf{X} leads to the following:

$$\mathbf{x}' = \mathbf{K}'\mathbf{R}'\mathbf{R}^{-1}\mathbf{K}^{-1}\mathbf{x}$$

For rotation at constant zoom, the intrinsics are constant. Therefore,

$$\mathbf{x}' = \mathbf{K}\mathbf{R}_{rel}\mathbf{K}^{-1}\mathbf{x} \quad (3)$$

where $\mathbf{R}_{rel} = \mathbf{R}'\mathbf{R}^{-1}$ denotes the relative 3D rotation about \mathbf{C} between the two views and \mathbf{K} denotes the intrinsic for the fixed zoom level. Similarly for a zooming camera with fixed orientation,

$$\mathbf{x}' = \mathbf{K}'\mathbf{K}^{-1}\mathbf{x} \quad (4)$$

These 2D homographies $\mathbf{H}_{rot} = \mathbf{K}\mathbf{R}_{rel}\mathbf{K}^{-1}$ and $\mathbf{H}_{zoom} = \mathbf{K}'\mathbf{K}^{-1}$ are used for automatic calibration of PTZ cameras [7, 8].

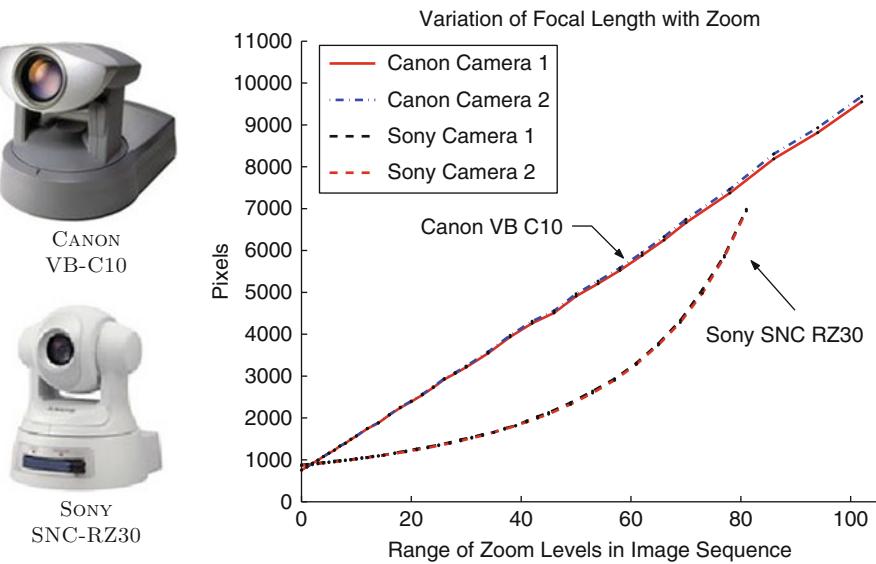
Radial Distorting Correction: PTZ cameras deviate from an ideal pinhole model due to radial distortion. This can be corrected using a standard model [9, 10]. The effect of radial distortion is more pronounced for smaller focal lengths, i.e., for lower zoom settings. The coefficients of radial distortion for a PTZ camera are therefore functions of zoom. The center of distortion is often assumed to coincide with the principal point (u_z, v_z) which is also dependent on zoom [11].

PTZ Camera Calibration: For a PTZ camera, the camera calibration process involves estimating the

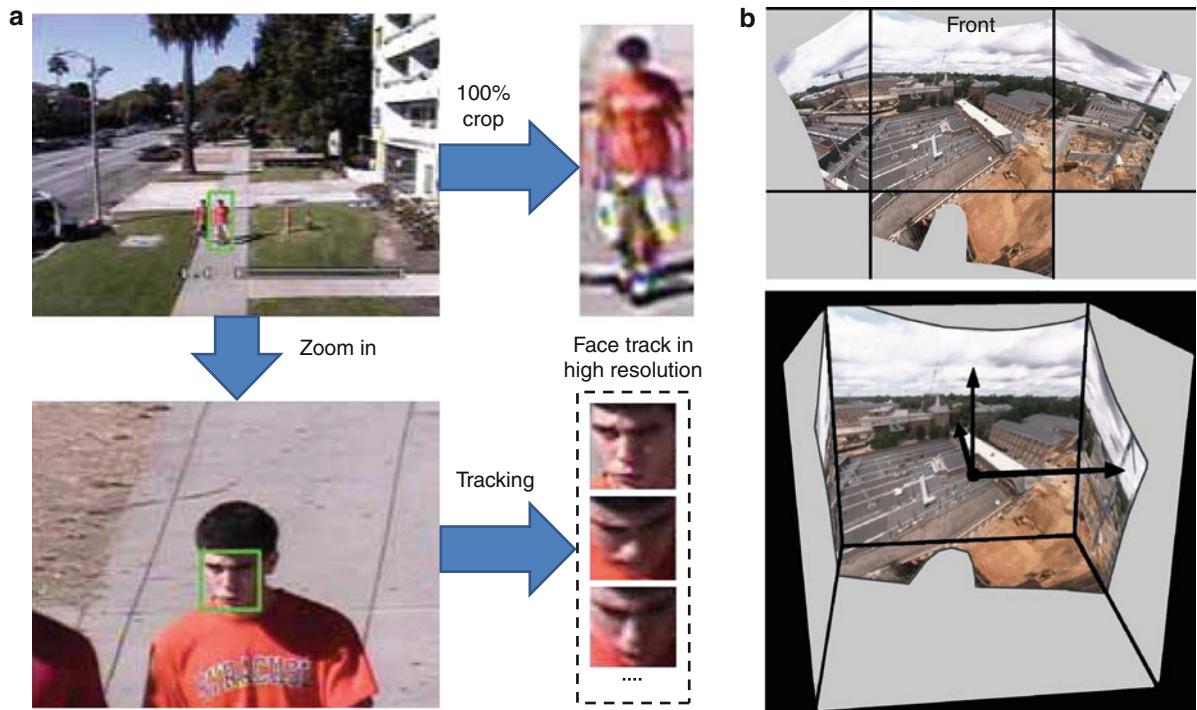
camera intrinsics \mathbf{K}_z and coefficients of radial distortion for all zoom settings within the admissible range. Some other methods only model the variation of focal length with zoom [8]. In practice, the intrinsics are computed at a discrete set of zoom values, and during online operation the intrinsics for any zoom setting can be obtained by piecewise interpolation after looking up values in a precomputed table [7, 8]. The extrinsic parameters of a PTZ camera can be computed using image observations of 3D points with known coordinates [3, 10]. Alternatively, structure from motion techniques may also be used to calibrate a PTZ camera network where a common area is visible to all the cameras. The process is similar to the calibration of conventional camera networks [12, 13] but requires handling dynamic network topologies due to changing visibility relations between active cameras in the network [14].

To maintain precise calibration of an active PTZ camera in deployment, one must also address the fact that most PTZ cameras lack precise and repeatable pan-tilt controllers. This implies that even if the motorized controllers are precalibrated, the true 3D rotation of the camera for a particular pan-tilt setting of the camera may differ from the orientation predicted based on the precalibration. Systems where a precise calibration must be maintained may require a closed-loop calibration system depending on the accuracy of the PTZ controllers [7]. For closed-loop calibration, a calibrated panoramic mosaic must be computed offline. During online operation, this image serves as a calibration reference for all video frames captured online. Every video frame can then be aligned to the calibrated mosaic using a homography computed online with a feature-based method. This makes it possible to estimate the true 3D directions on the camera's viewing sphere for all pixels in the video stream of the active camera.

In [7], both camera intrinsics and the calibrated mosaic are computed automatically using the motion of natural scene features induced by actively panning, tilting, and zooming the camera. The 2D feature correspondences are robustly computed using the 2D homography-based motion model for rotating and zooming cameras. Fig. 2 shows parameters estimated using this method for two common off-the-shelf PTZ cameras. This method makes it possible to automatically recalibrate PTZ cameras after they are deployed on site.



Pan-Tilt-Zoom (PTZ) Camera, Fig. 2 The variation of the focal length parameter with the zoom control for two common PTZ cameras. These parameters were estimated using an automatic method for PTZ camera calibration



Pan-Tilt-Zoom (PTZ) Camera, Fig. 3 (a) Examples of high-resolution face images captured by an active PTZ camera tracking system is shown. These images were acquired after pedestrians were detected in the zoomed out view of the camera.

(b) (top) A seamless panorama shown as an unwrapped cubemap generated by automatically stitching images from a rooftop PTZ surveillance camera is shown. (bottom) The calibrated cubemap is shown in the camera coordinate system



Pan-Tilt-Zoom (PTZ) Camera, Fig. 4 Closed-loop control of an active PTZ camera. (*Left-top*) Video frame captured online. (*Left-middle*) Frame generated from the calibrated panorama

based on predicted orientation. (*Left-bottom*) The aligned video frame. (*Right*) The predicted and aligned frames shown overlaid on the panorama

Application

Capturing High-Resolution Images of Humans:

Surveillance cameras monitoring large scenes cannot capture images of human faces at a sufficient resolution for identifying the individual when they are at a distance. PTZ cameras have been used to automatically zoom in on the face and capture a series of high-resolution face images once a pedestrian is detected (see [Fig. 3a](#)) [15]. This has applications in forensic video analysis. A multi-view video acquisition system utilizing a network of controllable PTZ cameras was also described in [16]. It was designed to track a person within an indoor scene while automatically adapting the pan-tilt camera controls to keep the individual centered in all the views. Such a system can be used for video conferencing [17] and event broadcasting.

Detection and Tracking: PTZ cameras are often deployed as part of a network of cameras for visual surveillance. In certain cases, static cameras provide a global view of the environment and are used primarily for detection and for actively steering the PTZ camera to regions of interest or to actively track

moving targets. Such systems find applications in traffic monitoring, tracking pedestrians, security systems, or automatic event detection. To monitor more complex environments, smart surveillance systems can use multiple calibrated PTZ cameras working in cooperation to track multiple targets [18, 19]. Accurate calibration is required to correctly match targets tracked by different PTZ cameras and for handing off a moving target from one camera to another [20].

PTZ Closed-Loop Calibration: An open-loop calibration system for PTZ cameras that rely only on precomputed calibration of the PTZ controls will tend to be inaccurate during operation due to the imprecise nature of the PTZ controllers or due to vibrations or other sources of instability. To deal with this, a closed-loop calibration system should be used which is based on a calibrated panorama constructed offline (see [Fig. 3b](#) for an example). [Figure 4](#) shows an outdoor panorama used for closed-loop control in [7]. Using feature-based image alignment, the video frames are robustly aligned to the calibrated panorama which allows pixels in the video stream to be precisely mapped to 3D directions on the camera's viewing

sphere. When extrinsic parameters are also known, this provides the ability to perform 3D scene reasoning within a PTZ camera network.

References

1. Zhang Z (1999) Flexible camera calibration by viewing a plane from unknown orientations. In: International conference on computer vision (ICCV 1999), Kerkyra, vol 1, pp 666–673
2. Wu Z, Radke R (2012) Keeping a pan-tilt-zoom camera calibrated. IEEE Trans Pattern Anal Mach Intell 99:1. PrePrints
3. Bouguet J (2000) Matlab camera calibration toolbox http://www.vision.caltech.edu/bouguetj/calib_doc/
4. Tsai R (1987) A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. Robotics and Automation, IEEE Journal of, 3(4):323–344
5. Davis J, Chen X (1999) Calibrating pan-tilt cameras in wide-area surveillance networks. In: Proceedings of the IEEE international conference on computer vision (ICCV), Kerkyra, pp 528–534
6. de Agapito L, Hartley R, Hayman E (1999) Linear self calibration of a rotating and zooming camera. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Ft. Collins, pp 15–21
7. Sinha S, Pollefeys M (2006) Pan-tilt-zoom camera calibration and high-resolution mosaic generation. Comput Vis Image Underst 103(3):170–183
8. Sankaranarayanan K, Davis JW (2010) Ptz camera modeling and panoramic view generation via focal plane mapping. In: Proceedings of the 10th Asian conference on computer vision – vol. Part II. ACCV, Queenstown, pp 580–593
9. Hartley R, Zisserman A (2005) Multiple view geometry in computer vision, vol 23. Cambridge University Press, New York
10. Collins R, Tsin Y (1999) Calibration of an outdoor active camera system. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Ft. Collins, pp 528–534
11. Willson R, Shafer S (1993) What is the center of the image? In: Proceedings of the IEEE international conference on computer vision (ICCV), Berlin, pp 670–678
12. Sinha SN, Pollefeys M, McMillan L (2004) Camera network calibration from dynamic silhouettes. CVPR 1(1):195–202
13. Svoboda T, Martinec D, Pajdla T (2005) A convenient multicamera self-calibration for virtual environments. Presence Teleoper. Virtual Environ. 14(4):407–422
14. Devarajan D, Radke RJ, Chung H (2006) Distributed metric calibration of ad hoc camera networks. ACM Trans Sens Netw 2(3):380–403
15. Dinh TB, Vo N, Medioni GG (2011) High resolution face sequences from a ptz network camera. In: Ninth IEEE international conference on automatic face and gesture recognition (FG 2011), Santa Barbara. IEEE, pp 531–538
16. Collins RT, Amidi O, Kanade T (2002) An active camera system for acquiring multi-view video. In: Proceedings of the international conference on image processing, Rochester, pp 517–520
17. Liao C, Liu Q, Kimber D, Chiu P, Foote J, Wilcox L (2003) Shared interactive video for teleconferencing. In: Proceedings of the Berkeley, CA, USA, pp 546–554
18. Everts I, Sebe N, Jones GA (2007) Cooperative object tracking with multiple ptz cameras. In: Proceedings of the 14th international conference on image analysis and processing, ICIAP '07, Modena, pp 323–330
19. Collins R, Lipton A, Fujiyoshi H, Kanade T (2001) Algorithms for cooperative multisensor surveillance. Proc IEEE 89(10):1456–1477
20. Qureshi F, Terzopoulos D (2006) Surveillance camera scheduling: a virtual vision approach. Multimed Syst 12(3):269–283
21. Sinha SN, Pollefeys M, Kim SJ (2004) High-resolution multiscale panoramic mosaics from pan-tilt-zoom cameras. In: Proceedings of the Fourth Indian conference on computer vision, graphics and image processing, Calcutta, pp 28–33
22. Stillman S, Tanawongsuwan R, Essa I (1998) A system for tracking and recognizing multiple people with multiple cameras. In: Proceedings of the second international conference on audio-vision based person authentication, Washington DC, pp 96–101
23. Starzyk W, Qureshi FZ (2011) Multi-tasking smart cameras for intelligent video surveillance systems. In: IEEE conference on advanced video and signal based surveillance, Klagenfurt, vol 1, 154–159

Pan-Tilt-Zoom Camera Calibration

► Active Calibration

Parametric Curve

Bo Zheng

Computer Vision Laboratory, Institute of Industrial Science, The University of Tokyo, Meguro-ku, Tokyo, Japan

Related Concepts

► Algebraic Curve; ► Parametric Surface; ► Splines

Definition

A *parametric curve* S in 2-dimensional Euclidean space has the following form:

$$S(\cdot) : \mathbb{R} \rightarrow \mathbb{R}^2 \\ t \mapsto (x(t), y(t)), t \in [a, b] \quad (1)$$

where t is the parameter and varies in the domain $[a, b]$. In practice, the domain $[a, b]$ is often normalized as a specific region, such as $[0, 1]$. And $x(t)$, $y(t)$ are real-valued functions continuously mapping to a 2D point on a curve.

Similarly, a parametric curve S in 3-dimensional space has the following form:

$$\begin{aligned} S(\cdot) : \mathbb{R} &\rightarrow \mathbb{R}^3 \\ t &\mapsto (x(t), y(t), z(t)), t \in [a, b] \end{aligned} \quad (2)$$

For example, an ellipse can be represented in a parametric form as: $x = a \cos t$, $y = b \sin t$, with $t \in [0, 2\pi]$, in contrast with its implicit representation: $\frac{x^2}{a^2} + \frac{y^2}{b^2} - 1 = 0$.

Background

Since parametric functions are easy to construct and analyze, parametric curves are one of the most popular representations of general curves in computer graphics and computer vision. There are many types of parametric curves because of the flexibility in choosing the underlying analytic function. Spline curves are one of the most widely used parametric curves. Some popular spline curves are cubic curves, Hermite curves, Bézier curves, and B-spline curves (see *Splines* for further details).

Compared to nonparametric representations of curves, such as the implicit representation of *algebraic curves* and other explicit representation, parametric curves are easier to control the local geometry and thus attractive for interactive curve design or representing the nonrigid deformation; parametric curves are capable of approximating complex shapes with desired smoothness; parametric curves are independent of the choice of coordinate system and lend themselves well to geometric transformations.

Application

In computer vision, parametric curves are generally used in modeling image edges, contours, and 2D object boundaries. It is shown as a powerful tool for shape fitting and manipulating. There are numerous applications, such as image segmentation, rigid/nonrigid object registration, motion estimation, object tracking,

that benefit from parametric curve representations. For example, in the Snake-based image segmentation designed by Kass et al. [6] and modified by Brigger et al. [2], splines are used to model the image contours which converge to object shapes by minimizing the energy guided by external and internal forces. For image motion estimation, Szeliski and Coughlan [11] proposed to represent the local motion flow field using multi-resolution splines. And grid-based splines, such as Thin-Plate spline (TPS) [3] and Free Form Deformation (B-spline) proposed by Sederberg [9], can be effectively applied to nonrigid image registration. Other types of parametric curves are proposed for specific vision problems. For instance, Rational Gaussian curves [5] does not require a regular grid of control points and is suitable for shape recovery. And elastic strings [8] have metrics which are invariant under reparametrizations of curves and are useful for modeling elastic objects.

References

- Bartels RH, Beatty JC, Barsky BA (1987) An introduction to splines for use in computer graphics and geometric modeling. Morgan Kaufmann Publishers Inc., San Francisco
- Brigger P, Hoeg J, Unser M (2000) B-spline snakes: a flexible tool for parametric contour detection. IEEE Trans Image Process 9(9):1484–1496
- Duchon J (1977) Splines minimizing rotation-invariant semi-norms in Sobolev spaces. In: Constructive theory of functions of several variables, lecture notes in mathematics, vol 571. Springer, Berlin, pp 85–100
- Farin G (1990) Curves and surfaces for computer aided geometric design. Academic, San Diego
- Goshtasby A (1992) Design and recovery of 2-D and 3-D shapes using rational gaussian. Int J Comput Vis 10(3): 233–256
- Kass M, Witkin A, Terzopoulos D (1988) Snakes: active contour models. Int J Comput Vis 1:321–331
- Kaufman A (1987) Efficient algorithms for 3D scan-conversion of parametric curves, surfaces, and volumes. In: Proceedings of the 14th annual conference on computer graphics and interactive techniques, Association for Computing Machinery, New York, pp 171–179
- Mio W, Bowers JC, Liu X (2009) Shape of elastic strings in euclidean space. Int J Comput Vis 82(1):96–112
- Sederberg T (1986) Free-form deformation of solid geometric models. ACM SIGGRAPH Comput Graph 20(4): 151–160
- Sederberg TW, Anderson DC, Goldman RN (1984) Implicit representation of parametric curves and surfaces. Comput Vis Graph Image Process 28(1):72–84
- Szeliski R, Coughlan J (1997) Spline-based image registration. Int J Comput Vis 22(3):199–218

Parametric Surface

Bo Zheng

Computer Vision Laboratory, Institute of Industrial Science, The University of Tokyo, Meguro-ku, Tokyo, Japan

Related Concepts

► [Algebraic Curve](#); ► [Parametric Curve](#); ► [Splines](#)

Definition

A parametric surface is a surface in the Euclidean space \mathbb{R}^3 which is defined by a parametric equation with two parameters,

$$\begin{aligned} \mathbb{S}(\cdot) : \mathbb{R}^2 &\rightarrow \mathbb{R}^3 \\ (u, v) &\mapsto (x(u, v), y(u, v), z(u, v)), \end{aligned} \quad (1)$$

where u, v are the parameters and vary within a certain 2D domain in the parametric uv -plane. $x(u, v), y(u, v), z(u, v)$ are the real-valued functions continuously mapping to the points on a surface.

For example, Bézier surface is one of the most commonly used parametric surfaces (patch) defined as

$$P(u, v) = \sum_{i=0}^m \sum_{j=0}^n B_i^m(u) B_j^n(v) \mathbf{p}_{ij}, \quad (2)$$

where \mathbf{p}_{ij} ($\in \mathbb{R}^3$) are the control points of Bézier spline surface. $B_i^n(\cdot) : \mathbb{R} \rightarrow \mathbb{R}$ are the basis functions determined by *Bernstein polynomials* of degree n (see the detail in contribution *splines*).

Parameterization is an important process of deciding and defining the parameters necessary for modeling a parametric surface. For example, through a spherical parameterization, a unit sphere can be described as

$$\begin{aligned} x &= \cos \theta \cos \varphi \\ y &= \cos \theta \sin \varphi \\ z &= \sin \theta \\ -\frac{\pi}{2} &\leq \theta < \frac{\pi}{2}, 0 \leq \varphi < \pi. \end{aligned} \quad (3)$$

Background

Parametric representation is the most general method to represent a surface, because it is capable of modeling a complex shape in a compact set of parameters and with desired smoothness. Surfaces that occur in two of the main theorems of vector calculus, Stokes' theorem and the divergence theorem, are frequently given in a parametric form. Parametric surfaces also provide a convenient way for computing the curvature and arc length of curves on the surface, surface area, differential geometric invariants such as the first and second fundamental forms, Gaussian, mean, and principal curvatures. Compared to implicit representation, parametric representation is more convenient for image rendering, due to the local points on a surface that can be fast and precisely determined. In addition, they may provide the potential for feature retrieving and surface modifying. Some typical parametric surfaces (patches) are spline-based parametric surfaces, such as bicubic patches, Bézier patches, and B-spline patches (see the contribution *splines*).

Application

Parametric surface attracts attention in computer vision, because of its convenience in handling 3D image segmentation, nonrigid registration, and recognition. For example, a classic method for image segmentation, the level set method introduced by Osher and Fedkiw [8], employs parametric curve/surface to represent image contours (level sets) for tracking shapes. The parametric curve/surface makes it easy to handle shapes that change topology, e.g., shape splits in two, develops holes, or the reverse of these operations. The snake-based image segmentation designed by Kass et al. [7] and modified by Brigger et al. [2] employs splines to model the image contours and serving for an energy minimization guided by external and internal forces. Spherical harmonics represented in spherical parameterized coordinates play a special role in a wide variety of topics including indirect lighting and in recognition of 3D shapes [5]. And grid-based parametric surface, such as Thin-Plate spline (TPS) [3] and Free-Form Deformation (B-spline) proposed by Sederberg [9], can be effectively applied for nonrigid object registration.

References

1. Bartels RH, Beatty JC, Barsky BA (1987) An introduction to splines for use in computer graphics and geometric modeling. Morgan Kaufmann, San Francisco
2. Brigger P, Hoeg J, Unser M (2000) B-spline snakes: a flexible tool for parametric contour detection. *IEEE Trans Image Proc* 9(9):1484–1496
3. Duchon J (1977) Splines minimizing rotation-invariant semi-norms in Sobolev spaces. Constructive theory of functions of several variables. *Lect Notes Math* 571:85–100
4. Farin G (1990) Curves and surfaces for computer aided geometric design. Academic, San Diego
5. Funkhouser T, Min P, Kazhdan M, Chen J, Halderman A, Dobkin D, Jacobs D (2003) A search engine for 3D models. *ACM Trans Graph* 22(1):83–105
6. George P-L, Borouchaki H, Laug P (2000) Parametric surface meshing using a combined advancing-front generalized Delaunay approach. *Int J Numer Method Eng* 49(1–2): 233–259
7. Kass M, Witkin A, Terzopoulos D (1988) Snakes: active contour models. *Int J Comput Vis* 1:321–331
8. Osher SJ, Fedkiw RP (2002) Level set methods and dynamic implicit surfaces. Springer, New York
9. Sederberg T (1986) Free-form deformation of solid geometric models. *ACM SIGGRAPH Comput Graph* 20(4):151–160

Partitioning

► Interactive Segmentation

Penumbra and Umbra

Rajeev Ramanath¹ and Mark S. Drew²

¹DLP® Products, Texas Instruments Incorporated, Plano, TX, USA

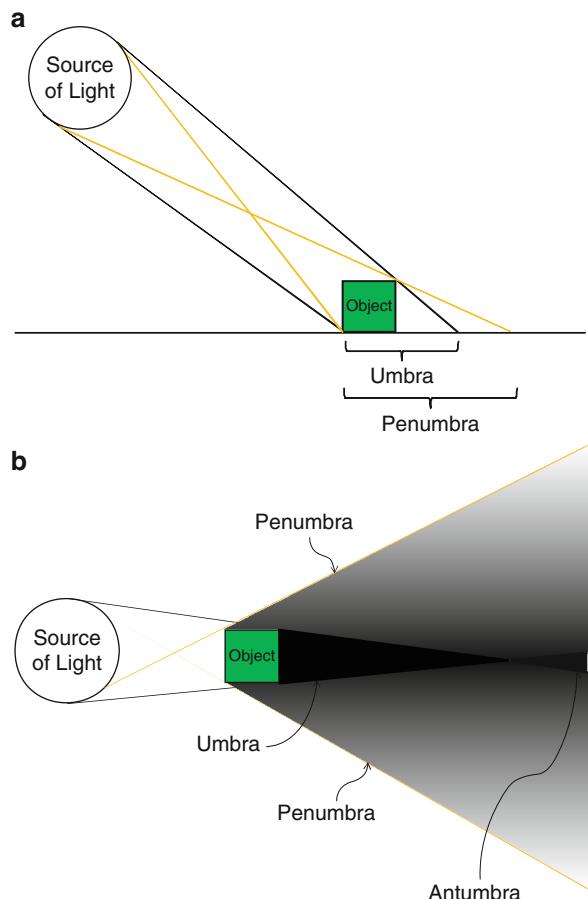
²School of Computing Science, Simon Fraser University, Vancouver, BC, Canada

Synonyms

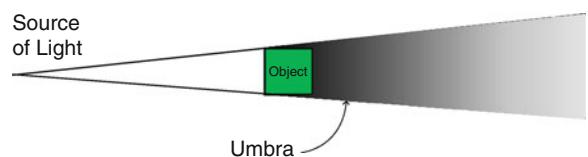
Shadow

Definition

Umbra is the Latin name for “shadow” and is typically used to refer to the shadow cast by an object when illuminated by a light source [2]. In the umbra region of the shadow, the entire light source is occluded



Penumbra and Umbra, Fig. 1 (a) Shadows cast in the case of an object on a surface and (b) shadows cast in the case of an object in free-space



Penumbra and Umbra, Fig. 2 Shadows cast in the case of an object in free space with a point source of light

by the object. The region around the umbra where only a portion of the light is obscured is called the penumbra [1].

Considering a simplistic arrangement of a light source and an object, Fig. 1a shows the regions denoted as the umbra and penumbra. As well, Fig. 1b shows the antumbra, the region where the light source actually

appears bigger than the object; an antumbra is typically formed only when the object is in free-space, as opposed to resting on a surface.

The above considerations are for an area light source. In contrast, it is straightforward to see that for a point source of light, only an umbra is cast (see Fig. 2).

In astronomical setups – similar to when objects are in free space – observers who see partial eclipses (of the sun or the moon) are located in the penumbra.

References

1. Encyclopedia Britannica (2010) <http://www.britannica.com/EBchecked/topic/450494/penumbra>. Accessed 10 Sept 2010
2. Encyclopedia Britannica (2010) <http://www.britannica.com/EBchecked/topic/613811/umbra>. Accessed 10 Sept 2010

Performance Capture

► [Motion Capture](#)

Perspective Camera

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Synonyms

[Pinhole camera](#)

Related Concepts

► [Camera Calibration](#); ► [Camera Parameters \(Intrinsic, Extrinsic\)](#); ► [Calibration of Projective Cameras](#); ► [Depth Distortion](#); ► [Perspective Transformation](#)

Definition

A *perspective camera* is a mathematical model of an ideal pinhole camera that follows perspective projection.

Background

A modern camera generally consists of an enclosed hollow with an opening (aperture) at one end for light to enter, a lens positioned in front of the opening, and a recording surface on the other end. In an ideal pinhole camera, the camera aperture is described as a point and no lenses are used to focus light. In that case, the camera can be modeled by a perspective transformation, thus also called *perspective camera*.

This model does not consider many effects of a real camera such as geometric distortions or blurring of unfocused objects caused by lenses and finite-sized apertures. Therefore, the pinhole camera model (perspective camera) can only be used as a first-order approximation of the transformation from a 3D scene to a 2D image. Its validity depends on the quality of the camera and the camera calibration process.

Lens distortion might be the major effect that the pinhole camera model does not take into account. It is sufficiently small if a high-quality camera is used and thus can be neglected. Otherwise, less distortion can be modeled through camera calibration and can be compensated for by applying suitable coordinate transformations on the image coordinates (lens distortion correction). Because of that, the pinhole camera model (perspective camera) has been the most popularly used camera model in computer vision and computer graphics to describe the relationship between a 3D scene and an image.

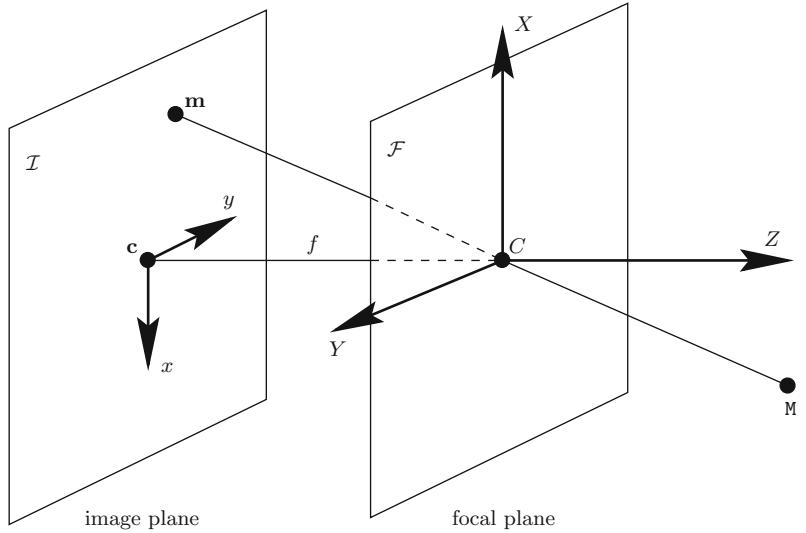
Theory

Figure 1 shows a pinhole camera model. There is a plane \mathcal{F} at a fixed distance f in front of an *image plane* \mathcal{I} . The image plane is also called the *retinal plane*. An ideal pinhole C is found in the plane \mathcal{F} . Assume that an enclosure is provided so that only light coming through the pinhole can reach the image plane. The rays of light emitted or reflected by an object pass through the pinhole and form an inverted image of that object on the image plane. Each point in the object, its corresponding image point, and the pinhole constitute a straight line. This kind of projection from 3D space to a plane is called *perspective projection*.

The geometric model of a pinhole camera thus consists of an image plane \mathcal{I} and a point C on the plane \mathcal{F} .

Perspective Camera, Fig. 1

The pinhole camera model



The point C is called the *optical center*, or the *focus*. The plane \mathcal{F} going through C and parallel to \mathcal{I} is called the *focal plane*. The distance between the optical center and the image plane is the *focal length* of the optical system. The line going through the optical center C and perpendicular to the image plane \mathcal{I} is called the *optical axis*, and it intersects \mathcal{I} at a point c , called the *principal point*. It is clear that the focal plane is also perpendicular to the optical axis. Experiences have shown that such a simple system can accurately model the geometry and optics of most of the modern Vidicon and CCD cameras [1].

Now let us derive the equations for the perspective projection. The coordinate system (c, x, y) for the image plane is defined such that the origin is at the point c (intersection of the optical axis with the image plane) and that the axes are determined by the camera scanning and sampling system. We choose the coordinate system (C, X, Y, Z) for the three-dimensional space as indicated in Fig. 1, where the origin is at the optical center and the Z -axis coincides the optical axis of the camera. The X - and Y -axes are parallel, but opposite in direction, to the image x - and y -axes. The coordinate system (C, X, Y, Z) is called the *standard coordinate system* of the camera, or simply *camera coordinate system*. From the above definition of the camera and image coordinate system, it is clear that the relationship between 2D image coordinates and 3D space coordinates can be written as

$$\frac{x}{X} = \frac{y}{Y} = \frac{f}{Z}. \quad (1)$$

It should be noted that, from the geometric viewpoint, there is no difference to replace the image plane by a virtual image plane located on the other side of the focal plane (Fig. 2). Actually this new system is what people usually use. In the new coordinate system, an image point (x, y) has 3D coordinates (x, y, f) , if the scale of the image coordinate system is the same as that of the 3D coordinate system.

Perspective Projection Matrix

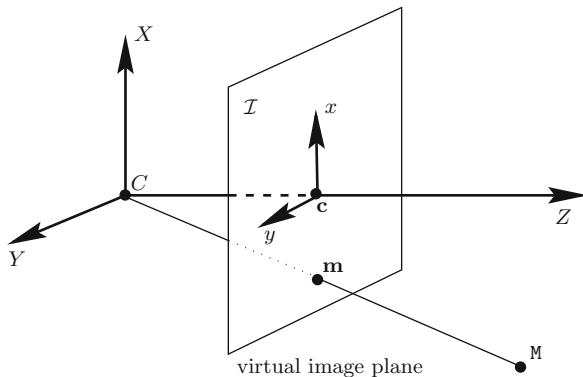
The relationship between 3D coordinates and image coordinates, Eq. (1), can be rewritten linearly as

$$\begin{bmatrix} U \\ V \\ S \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \quad (2)$$

where $x = U/S$ and $y = V/S$ if $S \neq 0$.

Given a vector $\mathbf{x} = [x, y, \dots]^T$, we use $\tilde{\mathbf{x}}$ to denote its augmented vector by adding 1 as the last element. Let \mathbf{P} be the 3×4 matrix

$$\mathbf{P} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$



Perspective Camera, Fig. 2 The pinhole camera model with a virtual image plane

which is called the camera *perspective projection matrix*. Given a 3D point $\mathbf{M} = [X, Y, Z]^T$ and its image $\mathbf{m} = [x, y]^T$, the formula (Eq. 2) can be written in matrix form as

$$s\tilde{\mathbf{m}} = \tilde{\mathbf{P}}\tilde{\mathbf{M}}, \quad (3)$$

where $s = S$ is an arbitrary nonzero scalar.

For a real image point, S should not be 0. We now make an extension to include the case $S = 0$. If $S = 0$, then $Z = 0$, i.e., the 3D point is in the focal plane of the camera, and the image coordinates x and y are not defined. For all points in the focal plane but the optical center, their corresponding points in the image plane are at infinity. For the optical center C , we have $U = V = S = 0$ (i.e., $s = 0$) since $X = Y = Z = 0$.

The reader is referred to the entry *camera parameters (Intrinsic, Extrinsic)* for description of more general form of perspective projection matrix with intrinsic and extrinsic parameters [2].

References

1. Faugeras O (1993) Three-dimensional computer vision: a geometric viewpoint. MIT, Cambridge
2. Xu G, Zhang Z (1996) Epipolar geometry in stereo, motion and object recognition. Kluwer Academic, Dordrecht/Boston

Perspective Projection

► Perspective Transformation

Perspective Transformation

Zhengyou Zhang

Microsoft Research, Redmond, WA, USA

Synonyms

Perspective camera; Perspective projection

Related Concepts

► Affine Camera; ► Projection; ► Weak Perspective Projection

Definition

A *perspective transformation*, also called *perspective projection*, is a mathematical model to describe the projection performed by a perspective camera. Under perspective projection, an object that is closer to the camera appears larger than those farther away. See entry “► Perspective Camera” for details.

Phong Model

► Phong Reflectance Model

Phong Reflectance Model

Ping Tan

Department of Electrical and Computer Engineering,
National University of Singapore, Singapore,
Singapore

Synonyms

Phong model

Related Concepts

► Radiance; ► Reflectance Models; ► Specularity, Specular Reflectance

Definition

The Phong reflectance model calculates the amount of reflected radiance at a surface point according to the lighting, viewing, and surface normal directions at that point. It is characterized by modeling specular reflection as an exponential function of a cosine function, which provides moderate accuracy in a simple formulation. It was introduced by Bui Tuong Phong in 1973 in his PhD thesis and later published in [1].

Background

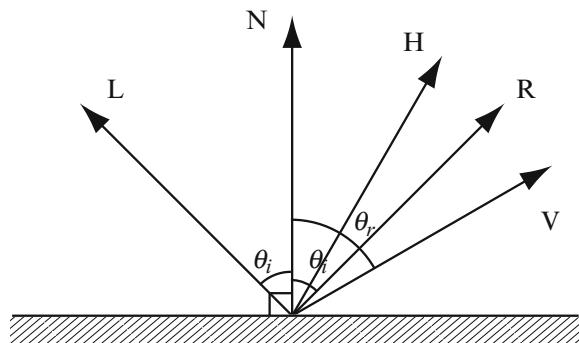
When light arrives at surfaces, it can be reflected, refracted, scattered, or absorbed. Reflectance models are mathematical functions that describe the interactions between light and surfaces. Usually, they are functions of lighting, viewing, and surface normal directions. There are various reflectance models at different levels of precision and complexity. Most reflectance models include components describing diffuse and specular reflection, respectively.

The Phong model is a reflectance model widely used for its simplicity and moderate accuracy. It represents reflected light as a linear combination of ambient, diffuse, and specular components. This model is characterized by its treatment of the specular component, which is designed based on the empirical observation that glossy surfaces have focused highlight that falls off quickly and matt surfaces have extended highlight that falls off slowly. Like many other reflectance models, this model can be applied to create computer graphic images or to infer scene properties such as shape and material from observed image intensities.

Theory

Light reflected by a surface can be roughly divided into specular and diffuse reflection. The specular reflection, or highlight, refers to the light reflected directly at the surface without entering it. Specular reflection typically is concentrated within a small angle in space. In comparison, diffuse reflection refers to the light that enters the surface and is distributed more uniformly in all reflected directions.

The observed intensity of specular reflection depends on the viewing direction. For example, an

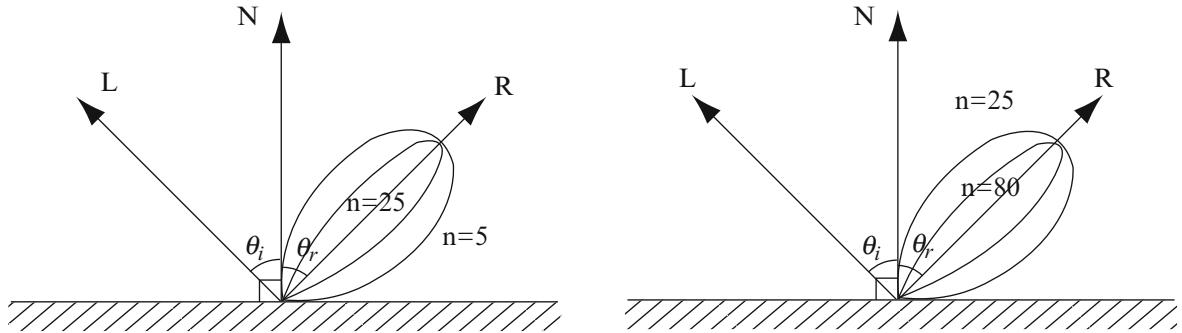


Phong Reflectance Model, Fig. 1 Directions involved in the definition of Phong model. N is the surface normal direction. L and V are the lighting and viewing direction, respectively. H is the bisector of L and V . R is L mirrored about N

ideal reflector like a mirror reflects light toward a direction R that is the incident direction L reflected about the surface normal direction N . Hence, the reflectance is zero except when the viewing direction V is coincident with R . Here, R is within the same plane as L and N , and the angle between R and N is the same as that between L and N . Mathematically, R can be computed as $R = 2N(N \cdot L) - L$. All these directions are represented by unit vectors. The geometric relations of these unit vectors are shown in Fig. 1.

Most real surfaces are not ideal reflectors. The Phong model can be used to describe their specular reflection. According to this model, the specular reflection is centered at the direction R and falls off as an exponential function of cosine of the angle between R and V . In other words, the specular reflection is proportional to $(V \cdot R)^n$. $n > 0$ is known as shininess, which determines the spread of the specular reflection. It is larger for glossy surfaces and smaller for matt surfaces. Specular reflections with different shininess are illustrated on the left of Fig. 2. When $n = \infty$, it represents reflection of ideal reflectors. It should be noticed that Phong model is designed based on empirical observations rather than physics. There are other physically based reflectance models for specular reflection such as Cook-Torrance model [2], Ward's model [3], and Ashikhmin's model [4] that are more complicated and accurate than the Phong model.

The complete Phong reflectance model represents reflected light as a linear combination of ambient, diffuse, and specular components. The ambient component is a small constant accounting for weak scattered light in the scene. The diffuse component



Phong Reflectance Model, Fig. 2 *Left:* the specular reflection of Phong model. *Right:* the specular reflection of Blinn-Phong model

is represented by the Lambert's model. Hence, the reflected radiance is computed as

$$I = k_a I_a + k_d I_d \max(N \cdot L, 0) + k_s I_s (\max(V \cdot R, 0))^n. \quad (1)$$

Here, I_a , I_d , and I_s are parameters to indicate the illumination power for the ambient, diffuse, and specular components. k_a , k_d , and k_s are scalars that determine the relative strength of these three components. In color images, these parameters are vectors with different values at different wavelengths.

Variations There is a well-known variation of the Phong model, the Blinn-Phong model, which is proposed by Jim Blinn [5]. In the Blinn-Phong model, the term $V \cdot R$ is replaced by $N \cdot H$. Hence, the reflected radiance is

$$I = k_a I_a + k_d I_d \max(N \cdot L, 0) + k_s I_s (\max(N \cdot H, 0))^{n'}, \quad (2)$$

where $H = \frac{L+V}{\|L+V\|}$ is the bisector of L and V , which is often called "halfway vector." When V is in the same plane as N and L , the angle between N and H is half of that between V and R . Hence, it is a close approximation to the original Phong model. This model is illustrated on the right of Fig. 2. The Blinn-Phong model is originally proposed to speed up the computation of the Phong model. It is the default shading model in OpenGL [6] for its efficiency. According to an experimental evaluation [7] with measured reflectance data from real surfaces, the Blinn-Phong model can represent real data more accurately than the original Phong model.

Application

The Phong reflectance model computes reflected radiance according to the 3D shape, viewing, and lighting configurations. Like many other reflectance models, it is widely used to create computer graphic images. It also provides an analytical tool for radiometric image analysis to understand scene properties such as shape and material from measured image intensities.

References

1. Phong BT (1975) Illumination for computer generated pictures. Commun ACM 18(6):311–317
2. Cook RL, Torrance KE (1981) A reflectance model for computer graphics. In: SIGGRAPH '81: proceedings of the 8th annual conference on computer graphics and interactive techniques, New York, NY, USA. ACM, pp 307–316
3. Ward GJ (1992) Measuring and modeling anisotropic reflection. In: SIGGRAPH '92: proceedings of the 19th annual conference on computer graphics and interactive techniques, New York, NY, USA, vol 26. ACM, pp 265–272
4. Ashikmin M, Premaž Š, Shirley P (2000) A microfacet-based brdf generator. In: SIGGRAPH '00: proceedings of the 27th annual conference on computer graphics and interactive techniques, New York, NY, USA. ACM/Addison-Wesley, pp 65–74
5. Blinn JF (1977) Models of light reflection for computer synthesized pictures. In: SIGGRAPH '77: proceedings of the 4th annual conference on computer graphics and interactive techniques, New York, NY, USA. ACM, pp 192–198
6. Shreiner D, Woo M, Neider J, Davis T (2005) OpenGL(R) programming guide: the official guide to learning OpenGL(R), Version 2 (5th Edition) (OpenGL). Addison-Wesley Professional, Boston
7. Ngan A, Durand F, Matusik W (2005) Experimental analysis of brdf models. In: Proceedings of the eurographics symposium on rendering, eurographics association. Association for Computing Machinery, New York, pp 117–226

Photo-Consistency

Yasutaka Furukawa
Google Inc., Seattle, WA, USA

Synonyms

Photometric consistency function

Related Concepts

► Camera Calibration; ► Multi-baseline Stereo

Definition

Photo-consistency is a scalar function that measures the visual compatibility of a 3D reconstruction with a set of calibrated images.

Background

Automated 3D reconstruction from images has been a core computer vision problem for years. Multi-view stereo (MVS) is a process of reconstructing 3D structure of an object or a scene from multiple images [1]. MVS assumes calibrated photographs, where camera calibration is often achieved by a calibration chart (e.g., checkerboard patterns) or a Structure from Motion algorithm [2].

In principle, how MVS algorithms recover 3D information is the same as how humans perceive depths with their two eyes, that is, triangulation from correspondences. Therefore, the critical first step of MVS is to establish feature correspondences across multiple input images, where a robust mechanism is necessary to evaluate the goodness of such feature correspondences, which is the role of a photo-consistency function. In a sense, an MVS reconstruction process is to carve out a 2D surface in a 3D space, where photo-consistency scores are high.

Theory and Examples

Photo-consistency $f(p, V)$ is a scalar function, which measures the visual compatibility of a given 3D

reconstruction p with a set of images V . Typically, p is a 3D point ($p \in \mathbb{R}^3$), while more sophisticated methods use an oriented point (a 3D point with a surface normal) [3, 4] or a bounded surface region such as a triangle in a polygonal mesh model [5, 6]. For the moment, the visibility information V is assumed to be given for p , where details about visibility estimation are referred to a later section.

A simple photo-consistency function at a 3D point p is defined as follows: p is projected into each visible image in V , and the similarity of image textures near their projections is computed as photo-consistency. Instead of comparing a single pixel color in each image, a set of pixel colors in each local image region is compared for robustness. More concretely, let A_{uv}^i be the $\tau \times \tau$ rectangular grid of pixel intensities centered at the image projection of p in image $I_i \in V$ (see Fig. 1). Note that u and v are indexes of the rectangular grid, and $\tau = 5, 7$ is typically used. Photo-consistency $f(p, V)$ can be defined as

$$f(p, V) = \sum_{I_i, I_j \in V} \sum_{u,v} (A_{uv}^i - A_{uv}^j)^2, \quad (1)$$

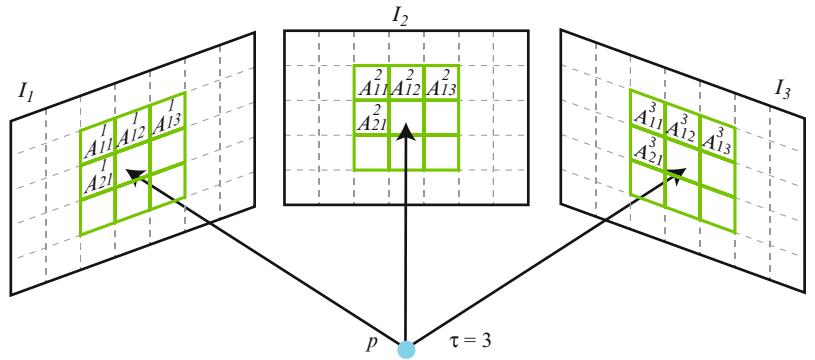
which evaluates a sum of squared differences (SSD) of intensities for every pair of images. SSD score is often used for binocular stereo problems, where a pair of images have a narrow baseline and are acquired under the same or similar lighting conditions. The main reasoning is that the SSD score is sensitive to illumination changes and non-Lambertian effects, which is often the case of MVS problems. Instead of SSD, many MVS algorithms employ the Normalized Cross Correlation measure, which has shown to be more robust:

$$f(p, V) = \frac{\sum_{u,v} (A_{uv}^i - \bar{A}_{uv}^i) (A_{uv}^j - \bar{A}_{uv}^j)}{\sqrt{\left(\sum_{u,v} (A_{uv}^i - \bar{A}_{uv}^i)^2\right) \left(\sum_{u,v} (A_{uv}^j - \bar{A}_{uv}^j)^2\right)}}. \quad (2)$$

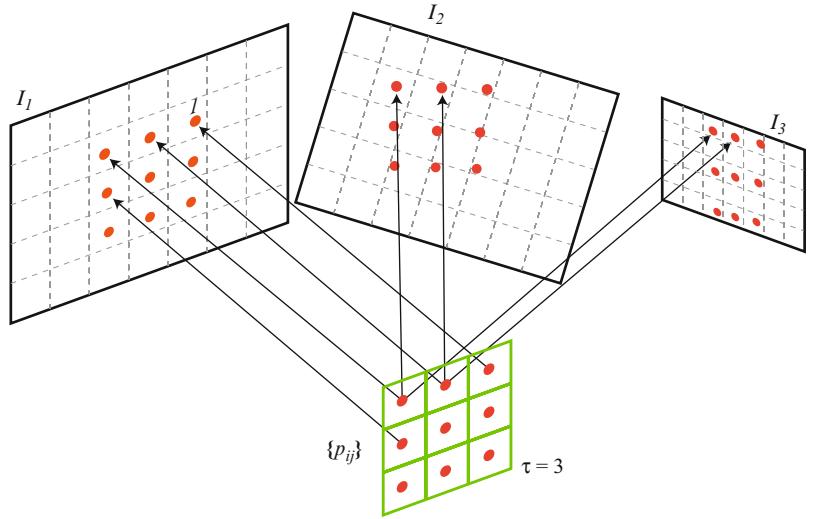
\bar{A}_{uv}^i is the average intensity of $\{A_{uv}^i\}$. Another advantage of the NCC score is that the value is guaranteed to be in a range $[-1.0, 1.0]$, where 0.6 or 0.7 is usually a good photo-consistency score, while 0.3 or 0.4 is bad. For a color image, where A_{uv}^i represents a 3D

Photo-Consistency, Fig. 1

A simple photo-consistency evaluation starts by projecting a 3D point p into each visible image and collecting pixel colors $\{A_{uv}^i\}$ in each local image region

**Photo-Consistency, Fig. 2**

More appropriate and accurate photo-consistency evaluation is to sample points in 3D, then project them into each image, which handles differences in camera rotations (I_2) and resolutions (I_3) properly



vector (red, green, blue), the formula slightly changes its form

$$f(p, V) = \sum_{I_i, I_j \in V} \frac{\sum_{u,v} (A_{uv}^i - \bar{A}_{uv}^i)^T \cdot (A_{uv}^j - \bar{A}_{uv}^j)}{\sqrt{\left(\sum_{u,v} |A_{uv}^i - \bar{A}_{uv}^i|^2\right) \left(\sum_{u,v} |A_{uv}^j - \bar{A}_{uv}^j|^2\right)}},$$

where the average intensity \bar{A}_{uv}^i is computed for each color channel independently.

This photo-consistency function works well for a simple scene, where images in V have similar resolutions, distances to the point p , and rotational parameters. For more complicated scenes, more appropriate evaluation is necessary, which is to first sample a

rectangular grid of 3D points $\{p_{ij}\}$ around p in the 3D space, project all the points into each image, and sample pixel colors at their projections (see Fig. 2). Typically, 3D points are sampled on a plane that is front parallel to one of the images in V (I_i in Fig. 2), so that image distances between the adjacent projected points are roughly one pixel. Sampled pixel colors are used in exactly the same way as before (Eq. 1)–(3) to compute photo-consistency scores. Figure 2 illustrates that this photo-consistency function takes into account camera rotation (I_2) and resolution (I_3) differences to sample colors at the correct pixel locations.

Advanced Photo-Consistency Functions

For robustness, photo-consistency functions often measure the similarity of image textures over a local region instead of at a single point. In this sense, a

natural input to the function should be a 3D point plus some spatial support. One example is an oriented point, which is a combination of a 3D location (\mathbb{R}^3) and a surface normal (\mathbb{S}^2), which essentially uses a tangent plane approximation of a surface to sample 3D points [3, 4]. Pons and Vu et al. initialize their reconstruction as a polygonal mesh model, and in the process of iterative mesh deformation, photo-consistency is evaluated on the polygonal mesh model directly [5, 6].

Visibility Estimation

Photo-consistency requires visibility information V as an input. However, it is not easy to obtain good visibility without a 3D model, simply because occlusion information is unknown. Similarly, without visibility, it becomes difficult to obtain accurate 3D reconstructions – chicken-and-egg problem. Furthermore, visibility should reflect certain photometric factors such as specular highlights – images with specular highlights should be excluded from V . Fortunately, researchers found out that accurate visibility is not necessary to obtain a successful 3D reconstruction. Instead, a “robust” photo-consistency function can be used to ignore outlier images in V , while overestimating V . There are several approaches in robustifying photo-consistency functions. One approach is to ignore images whose photo-consistency function scores are worse than a predetermined threshold, which is simple but has proven to work well [3]. Vogiatzis and Hernández et al. proposed more sophisticated approach that handles outliers systematically, which boosted their reconstruction accuracy [7].

References

1. Seitz SM, Curless B, Diebel J, Scharstein D, Szeliski R (2006) A comparison and evaluation of multi-view stereo reconstruction algorithms. CVPR 1:519–528
2. Hartley R, Zisserman A (2004) Multiple view geometry in computer vision. Cambridge University Press, Cambridge/New York
3. Furukawa Y, Ponce J (2010) Accurate, dense, and robust multi-view stereopsis. IEEE Trans Pattern Anal Mach Intell 32(8):1362–1376
4. Habbecke M, Kobbelt L (2007) A surface-growing approach to multi-view stereo reconstruction. CVPR
5. Pons JP, Keriven R, Faugeras O (2007) Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. IJCV 72(2):179–193
6. Vu HH, Keriven R, Labatut P, Pons JP (2009) Towards high-resolution large-scale multi-view stereo. In: Conference on computer vision and pattern recognition (CVPR), Miami
7. Vogiatzis G, Esteban CH, Torr PHS, Cipolla R (2007) Multi-view stereo via volumetric graph-cuts and occlusion robust photo-consistency. IEEE Trans Pattern Anal Mach Intell 29(12):2241–2246

Photogrammetry

Konrad Schindler¹ and Wolfgang Förstner^{1,2}

¹ETH Zürich, Zürich, Switzerland

²Universität Bonn, Bonn, Germany

Definition

Photogrammetry is the science and technology of obtaining information about the physical environment from images, with a focus on applications in surveying, mapping and high-precision metrology. The aim of photogrammetry is to provide automated or semi-automated procedures for these engineering tasks, with an emphasis on a specified accuracy, reliability and completeness of the extracted information.

Background

Photogrammetry is a long-established engineering discipline, which dates back to the middle of the nineteenth century, shortly after the invention of the photographic process. It has its roots in surveying, predominantly for aerial mapping of the earth’s surface, although terrestrial “close-range” photogrammetry has always been an integral part of the discipline. Traditionally photogrammetry has emphasized 3D *geometric* modeling of the environment, since in an interactive setting this implicitly encompassed the semantic interpretation of the image content. The use of geospatial imagery with the primary purpose to infer semantic object properties from radiometric intensities is often referred to as “remote sensing.” Today these two fields overlap both in terms of methodology and of applications.

Methods

The goal of photogrammetry is to extract geometric and semantic information from imagery.

In terms of *geometric* processing methods, the photogrammetric measurement process is essentially an application of structure-from-motion theory [6], mostly (but not exclusively) with calibrated camera intrinsics. In fact a large part of the theory of camera calibration and camera orientation was first developed and applied in photogrammetry, including camera orientation from 2D-to-3D correspondences [1, 5], relative camera orientation from 2D-to-2D correspondences [3], and bundle adjustment [2].

The methods for *semantic* interpretation comprise the entire armamentarium of image understanding, from early rule-based systems [14] through model-based object recognition [18] to statistical learning with modern Bayesian techniques [16]. For a recent overview, see [12]. Due to the complexity of the task, only semi-automatic methods have so far found their way into commercial software and operational production pipelines.

Relation to Computer Vision

Since the advent of digital images in the 1970s, a main goal has been to automate the photogrammetric process, and photogrammetrists have developed or adopted pattern recognition methods for tasks such as interest point extraction [15], feature matching and dense stereo reconstruction [7, 8], semantic segmentation [11], and object category detection [4]. Thus, the science of photogrammetry is increasingly converging with computer vision and image understanding. Still photogrammetry, being a practical engineering discipline, tends to put greater emphasis on a defined (usually high) accuracy, reliability, and completeness than on total automation.

Recent Developments

Since the 1990s range images captured directly with airborne and terrestrial laser scanners have gained popularity with both practitioners and researchers in geo-information, and have become a second main data source of photogrammetry [17].

Hand in hand with that development, it has become a standard routine to determine approximate or even final sensor orientations directly, rather than indirectly from observed points. The practice of observing the position and attitude of the camera during flight missions directly with a highly accurate GNSS (global navigation satellite system) receiver and IMU (inertial measurement unit) is known as *direct georeferencing*.

With the advent of high-resolution satellite sensors, spaceborne images from both optical and microwave sensors nowadays also serve as input data for the photogrammetric process. Mobile mapping systems mounted on vehicles have lead to a growing interest in large-scale photogrammetric mapping from the ground.

Application

The most important application field of photogrammetry is topographic mapping of the earth's surface at different scales. The overwhelming majority of all existing maps have been created through photogrammetric processing of airborne or spaceborne imagery. However, photogrammetry is also prominent for small-scale mapping down to single villages, mines, etc. A related endeavor has been the mapping of other planets in the solar system from images taken by spacecraft.

Non-topographic applications for a long time occupied only a small fraction of the market. They used to be subsumed under the term "close-range photogrammetry," the main application fields being industrial metrology (e.g., aircraft, ships, vehicle parts), construction, cultural heritage documentation, forensics, and the medical domain.

Although mapping remains the dominant application area, the boundary between photogrammetry and 3D computer vision is dissolving more and more. Today tasks like visual driver assistance and robot navigation, motion capture, virtual and augmented reality, object tracking, etc. are by many also considered applications of photogrammetry.

Technology transfer between academia and industry has always been well established in photogrammetry, via the International Society for Photogrammetry and Remote Sensing (ISPRS, <http://www.isprs.org>) and the long-running biennial Photogrammetric Week (<http://www.ifp.uni-stuttgart.de/publications/phowo.html>).

References

1. Abdel-Aziz YI, Karara HM (1971) Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. Proceedings of the Symposium on Close-Range Photogrammetry, Falls Church, VA, pp 1–18

2. Brown DC (1958) A solution to the general problem of multiple station analytical stereotriangulation. RCA-MTP Data Reduction Technical Report No. 43, Patrick Airforce Base, FL
3. Finsterwalder S (1897) Die geometrischen Grundlagen der Photogrammetrie. Jahresbericht der Deutschen Mathematiker-Vereinigung 6(2):1–41
4. Grabner H, Nguyen TT, Gruber B, Bischof H (2007) Online boosting-based car detection from aerial images. ISPRS J Photogramm Remote Sens 63(3):382–396
5. Grunert JA (1841) Das Pothenotische Problem in erweiterter Gestalt nebst Über seine Anwendungen in der Geodäsie. Grunerts Archiv für Mathematik und Physik 1: 238–248
6. Hartley RI, Zisserman A (2004) Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, Cambridge/New York
7. Helava UV (1978) Digital correlation on photogrammetric instruments. Photogrammetria 34(1):19–41
8. Kelly RE, McConnell PRH, Mildenberger SJ (1977) The Gestalt photomapping system. Photogramm Eng Remote Sens 42(11):1407–1417
9. Kraus K (2007) Photogrammetry: geometry from images and laser scans, 2nd edn. Walter de Gruyter, Berlin/New York
10. Luhmann T, Robson S, Kyle S, Harley I (2006) Close Range Photogrammetry: Principles, Techniques and Applications. Whittles, Dunbeath
11. Mayer H, Laptev I, Baumgartner A (1998) Multi-scale and snakes for automatic road extraction. Proceedings of the 5th European Conference on Computer Vision (ECCV), Freiburg, Germany, pp 720–733
12. Mayer H (2008) Object extraction in photogrammetric computer vision. ISPRS J Photogramm Remote Sens 63(2): 213–222
13. McGlone JC (ed.) (2013) Manual of Photogrammetry. 6th edn. American Society for Photogrammetry and Remote Sensing, Virginia
14. McKeown DM, Harvey WA, McDermott J (1985) Rule-Based interpretation of aerial imagery. IEEE Trans Pattern Anal Mach Intell 7(5):570–585
15. Paderes FC, Mikhail EM, Förstner W (1984) Rectification of single and multiple frames of satellite scanner imagery using points and edges as control. NASA Symposium in Mathematical Pattern Recognition and Image Analysis. Houston, TX, pp 309–400
16. Stoica R, Descombes X, Zerubia J (2004) A Gibbs point process for road extraction in remotely sensed images. Int J Comput Vis 57(2):121–136
17. Vosselman G, Mass H-G (eds) (2010) Airborne and terrestrial laser scanning. Whittles, Dunbeath
18. Weidner U, Förstner W (1995) Towards automatic building extraction from high-resolution digital elevation models. ISPRS J Photogramm Remote Sens 50(4): 38–49

Photometric Consistency Function

► Photo-Consistency

Photometric Invariants

Todd Zickler

School of Engineering and Applied Science, Harvard University, Cambridge, MA, USA

Related Concepts

► Image Decompositions

Definition

A photometric invariant is a function of an image, or a function of a set of images, that is discriminative with respect to some scene properties and independent of others.

Background

J. J. Gibson was the first to write extensively about computing useful “invariants” from visual observations. According to Gibson, an invariant represents the extraction of persistent scene properties despite changing environmental conditions and therefore enables the concurrent awareness of both persistence and change within a scene. While he does not use the term “photometric invariant,” he describes the concept quite well [10]:

... the illumination can change in amount, in direction, and in spectral composition. Some features of any optic array in the medium will change accordingly. There must be invariants for perceiving surfaces, their relative layout, and their relative reflectances. They are not yet known, but they almost certainly involve ratios of intensity and color among parts of the array.

The notion of a photometric invariant is different from, but related to, the mathematical definition of an invariant. In mathematics, an invariant is a property that remains unchanged when certain transformations are applied, whereas a photometric invariant is a function of an image (or set of images) that persists despite changes in certain scene properties and conditions.

Photometric invariants are also related to, but distinct from, image decompositions based on the explicit separation of an image according to scene characteristics, such as shading and lightness [11], diffuse and specular reflectance [12], or illuminant and spectral

reflectance [1]. Computing such explicit decompositions is ill-posed, whereas photometric invariants can be computed directly from image intensities, often in closed form. In fact, it is not uncommon for photometric invariants to be used as initialization for iterative image decomposition techniques.

Photometric invariants are used to discriminate among the scene properties of interest while eliminating dependence on other “distractors.” Insensitivity and discriminability are usually in conflict, and in general one cannot obtain one without sacrificing some of the other. For example, it can be shown that there is no photometric invariant computed from grayscale imagery that can exactly discriminate between surface shapes under variable lighting, because given any two images, one can always find a single surface to explain them [2]. A high-performing invariant, therefore, is one that provides a *balance* between insensitivity and discriminability and does so in a way that is appropriate for the desired visual task and operating environment. For this reason, performance of photometric invariants should be conducted empirically, preferably using imagery that is representative of the end goal.

Theory

An early discovery of a photometric invariant was made by Koenderink and van Doorn [13], who considered matte (Lambertian) surfaces under directional lighting. Under these conditions, certain stationary points of image isophotes cling to parabolic surface points and therefore provide information about surface shape that does not depend on the locations of light sources.

Since then, the community has discovered a diverse collection of photometric invariants, usually written as simple functions of intensity and color at the image projection of one or more scene points. Each of these invariants is derived by assuming models for lighting, reflectance, and spectral sensors and by doing so in the context of an imaging model that expresses the measurements in the k th channel as

$$I_k(x, L) = \int_{\Lambda} \int_{\mathbb{S}^2} c_k(\lambda) L(\lambda, \omega) f(x, \lambda, n, \omega, \omega') \max(0, \langle \omega, n \rangle) d\omega d\lambda. \quad (1)$$

Here, $\{c_k(\lambda)\}_{k=1\dots K}$ are the spectral responses of a camera with K channels; $f(x, \lambda, n, \omega, \omega')$ is the spatially varying, spectral bidirectional reflectance distribution function (BRDF) at the back projection of image point x evaluated with surface normal $n \in \mathbb{S}^2$, light direction $\omega \in \mathbb{S}^2$, and view direction $\omega' \in \mathbb{S}^2$; and $L(\lambda, \omega)$ is the (spatially uniform) spectral radiance distribution that illuminates the scene.

The photometric invariants differ in their assumptions about the sensors $\{c_k(\lambda)\}$ as well as their assumptions regarding reflectance and lighting. In all cases, reflectance and lighting models are based on factored representations that separate the variation with respect to spatial position, wavelength, and angular geometry:

$$f(x, \lambda, \theta) = \sum_{n=1}^N m_n(x, \lambda) g_n(\theta), \quad (2)$$

with $\theta \triangleq \{n, \omega, \omega'\}$ defined to simplify notation and

$$L(\lambda, \omega) = \sum_{m=1}^M e_m(\lambda) \ell_m(\omega). \quad (3)$$

The assumptions that underlie each derivation provide some intuition for when each invariant might be usefully employed. These assumptions are categorized below; the resulting invariants are listed in Tables 1 and 2; and visualizations for some of them appear in Fig. 1. Note that empirical evaluations of many of these invariants reveal them to be useful when the underlying assumptions are only approximately satisfied.

Reflectance model. Common factored models are the Lambertian model ($N = 1$, $g_1(\theta) = \text{constant}$ in Eq. 2) and the dichromatic model with a neutral interface ($N = 2$, $g_1(\theta) = \text{constant}$, $m_2(x, \lambda) = m_2(x)$). Invariants for both of these cases are listed separated in Tables 1 and 2.

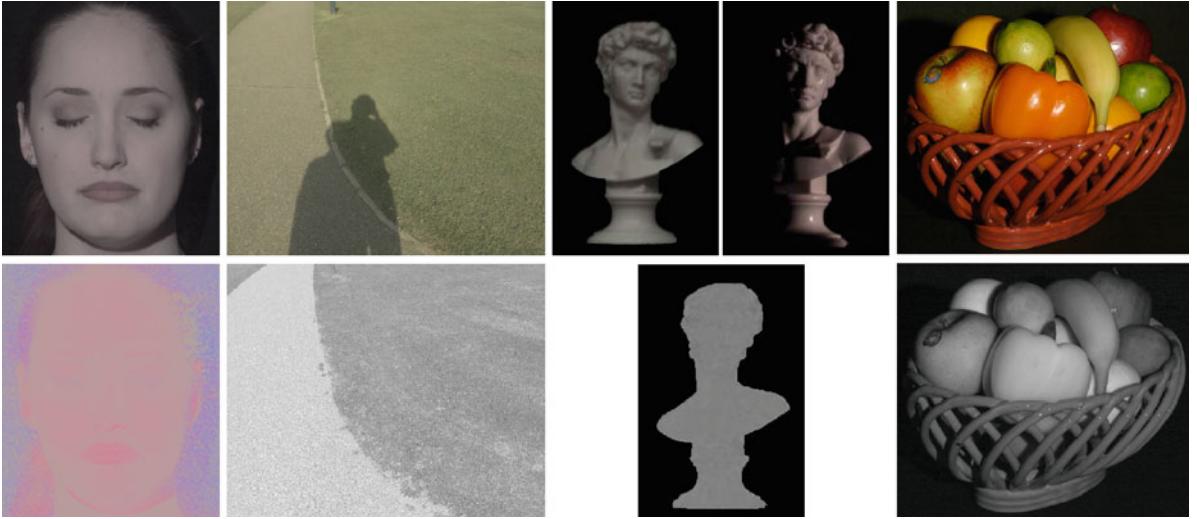
Lighting model. The most common instance of the factored lighting model has the same spectrum in every direction ($M = 1$ in Eq. 3), with the associated spectrum $e_1(\lambda)$ being either arbitrary, “even” ($e_1(\lambda) = \text{constant}$, so that $L(\lambda, \omega) = \ell(\omega)$), or Planckian and thus completely defined by its color temperature T : $e_1(\lambda; T)$.

Photometric Invariants, Table 1 Photometric invariants for surface reflectance of the form $f(x, \lambda, \theta) = m(x, \lambda)g(\theta)$, which includes the Lambertian model as a special case. Each is independent of certain scene properties and derived from assumed models of lighting and sensors

| Expression | Independent of | Comments |
|--|--|--|
| $I_1(x)$ | Geometry θ | – “Normalized RGB” |
| $I_1(x) + I_2(x) + I_3(x)$ | Intensity $\ell(\omega)$ | – Related: chromaticity, hue, saturation – Reference: <i>e.g.</i> , [9] – Lighting: $L(\omega, \lambda) = \ell(\omega)e(\lambda)$ – Sensors: general |
| $\frac{I(x_1)}{I(x_2)}$ | Geometry θ Intensity $\ell(\omega)$ Spectrum $e(\lambda)$ | – “Reflectance ratio” – x_1, x_2 must have same surface normal – References: [7, 15] – Lighting: $L(\omega, \lambda) = \ell(\omega)e(\lambda)$ – Sensors: von Kries |
| $\frac{I_1(x_1)I_2(x_2)}{I_1(x_2)I_2(x_1)}$ | Geometry θ Intensity $\ell(\omega)$ Spectrum $e(\lambda)$ | – Reference: [9] – Lighting: $L(\omega, \lambda) = \ell(\omega)e(\lambda)$ – Sensors: von Kries |
| $\alpha \log \frac{I_1}{I_3} + \beta \log \frac{I_2}{I_3}$ | Geometry θ Intensity $\ell(\omega)$ Spectrum $e(\lambda)$ | – (α_1, α_2) depend on camera sensors – Reference: [5] – Lighting: $L(\omega, \lambda) = \ell(\omega)e(\lambda)$ with $e(\lambda)$ Planckian – Sensors: narrow band |
| $\left(\frac{\partial I_2}{\partial x} I_1 - I_2 \frac{\partial I_1}{\partial x} \right) / I_1^2$ | Geometry θ Intensity $\ell(\omega)$ Spectrum $e(\lambda)$ | – Reference: [8] – Sensors: $c_1(\lambda)$ and $c_2(\lambda)$ must approximate Gaussian spectral derivatives |
| $\frac{I(x, L_1)}{I(x, L_2)}$ | Material $m(x, \lambda)$ | – “Photometric ratio” – Reference: [17] – Lighting: $L_j(\omega, \lambda) = \ell_j(\omega)e(\lambda)$ – Sensors: general |

Photometric Invariants, Table 2 Photometric invariants for surface reflectance of the form $f(x, \lambda, \theta) = m_1(x, \lambda)g_1(\theta) + m_2(x)g_2(\theta)$, which includes the dichromatic model with neutral interface as a special case. Each is independent of certain scene properties and derived from assumed models of lighting and sensors

| Expression | Independent of | Comments |
|---|---|---|
| $\text{atan} \left(\frac{\sqrt{3}(I_2 - I_3)}{(2I_1 - I_2 - I_3)} \right)$ | Geometry θ Intensity $\ell(\omega)$ | – “Hue” – Reference: [4] – Lighting: $L(\omega, \lambda) = \ell(\omega)$ – Sensors: general |
| $\alpha_1 I_1 + \alpha_2 I_2 + \alpha_3 I_3$ | Intensity $\ell(\omega)$ Component $m_2(x)g_2(\theta)$ | – “Color subspace” – $\{\alpha_i\}$ depend on $e(\lambda)$, sensors – Reference: [18] – Lighting: $L(\omega, \lambda) = \ell(\omega)e(\lambda)$; generalizes to $M > 1$ in Eq. 3 – Sensors: general |
| $\frac{I_1(L_1)I_2(L_2) - I_2(L_1)I_1(L_2)}{I_1(L_1)I_3(L_2) - I_3(L_1)I_1(L_2)}$ | Geometry θ | – “Ratio of determinants” – Generalizes to BRDF with $N > 2$ in Eq. 2 – Reference: [14] – Lighting: $L_j(\omega, \lambda) = \ell_j(\omega)e(\lambda)$ – Sensors: general |



Photometric Invariants, Fig. 1 Visualization of photometric invariants (*bottom row*) computed from one or two input images (*top row*). From left to right that based on normalized RGB, Planckian lighting [5], ratio of determinants [14], and color subspaces [18]

Color or grayscale. Invariants can be computed from multiple spectral measurements (usually three), which are denoted by $\{I_k(x)\}_{k=1,2,3}$ in Tables 1 and 2. In some cases, they are computed from a single spectral measurement (a “grayscale” image), which is denoted by $I(x)$.

Sensor model. For invariants based on color, the three sensors $\{c_k(\lambda)\}_{k=1,2,3}$ may be arbitrary, or they may be delta functions (*i.e.*, “narrow-band” sensors): $c_k(\lambda) = \delta(\lambda - \lambda_k)$. In between these two extremes are possibly overlapping sensors that nonetheless support spectral relighting by independent per-channel gain factors. Such sensors are said to support *von Kries adaptation*, and, as described in [3], they must satisfy a tensor rank constraint when integrated against all pairs of material spectral ($m_n(\cdot, \lambda)$) and lighting spectra ($e_m(\lambda)$) that could possibly exist in the operating environment.

Single point or multiple points. Invariants can be computed independently at each pixel, or they may be computed by combining measurements at distinct image points (denoted by $I(x_i)$ in the tables).

Single image or multiple images. Similarly, invariants can be computed from measurements in a single image or by combining measurements from multiple images captured under distinct lighting environments (denoted by $I(x, L_j)$ in the tables).

Application

Photometric invariants can be viewed as an alternative to learning-based approaches that attempt to model the appearance of persistent scene properties (shape, reflectance patterns, *etc.*) over all configurations of lighting, viewpoint, and other distractors. Instead of modeling this appearance variation, they “hard-code” an invariance, perhaps at the expense of discriminability. When applicable, the main advantages of invariant-based approaches are their computational efficiency and reduced requirement for training data. Whether a learning-based approach or an invariant-based approach (or a combination of the two) is more desirable depends on the visual task and the operating environment.

Multiple-point photometric invariants that are independent of geometry θ and light intensity $\ell(\omega)$ isolate surface reflectance information, and since they do not depend on light and viewing conditions, they may improve performance on object recognition and image indexing tasks [7, 15]. In some cases, performance may be further improved by using an invariant that is also independent of illuminant spectrum [9].

Invariants that are computed at a single point and are independent of geometry can also improve material-based segmentation and boundary detection by reducing the occurrence of false boundaries due to shading and specular highlights. In some cases, this

can be achieved from a single image [8, 18], and additional images can be used to handle more complex BRDFs [14].

The Planckian invariant [5] has the unique property of being independent of illuminant spectrum while also being computed at a single point. For this reason, it can be used for detecting and removing shadows in images that contain mixtures of distinct illuminant spectra [6].

Photometric invariants that are computed at a single point and are independent of material properties can be used to extract surface shape information. For diffuse surfaces, photometric ratios [17] can provide access to surface curvature information independent of surface albedo, and for surfaces described by the dichromatic model, color subspaces [18] can isolate the diffuse component and therefore improve the performance of shape-from-shading, photometric stereo, and a variety of other Lambertian-based vision algorithms.

Finally, as mentioned above, photometric invariants may be used as initialization for explicit image decompositions. The color subspace invariant [18] (and one closely related to it [16]) can be useful when decomposing an image into its diffuse and specular components, and the reflectance ratio [7, 15] is closely related to retinex-like algorithms for decomposing an image into shading and lightness [11].

References

1. Brainard D, Freeman W (1997) Bayesian color constancy. *J Opt Soc Am A* 14(7):1393–1411
2. Chen H, Belhumeur P, Jacobs D (2002) In search of illumination invariants. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)
3. Chong H, Gortler S, Zickler T (2007) The von Kries hypothesis and a basis for color constancy. In: Proceedings of the IEEE international conference on computer vision, Rio de Janeiro
4. D'Zmura M, Lennie P (1986) Mechanisms of color constancy. *J Opt Soc Am A* 3(10):1662–1672
5. Finlayson G, Hordley S (2001) Color constancy at a pixel. *J Opt Soc Am A* 18(2):253–264
6. Finlayson G, Hordley S, Drew M (2006) Removing shadows from images. In: Proceedings of the European conference on computer vision (ECCV), Graz
7. Funt B, Finlayson G (1995) Color constant color indexing. *IEEE Trans Pattern Anal Mach Intell* 17(5):522–529
8. Geusebroek J, van den Boomgaard R, Smeulders A, Geerts H (2001) Color invariance. *IEEE Trans Pattern Anal Mach Intell*, 23(12):1338–1350
9. Gevers T, Smeulders A (1997) Color based object recognition. In: Image analysis and processing. Springer, Berlin/New York, pp 319–326
10. Gibson J (1979) The ecological approach to visual perception. Houghton Mifflin, Boston
11. Horn B (1974) Determining lightness from an image. *Comput Graph Image Process* 3(4):277–299
12. Klinker G, Shafer S, Kanade T (1988) The measurement of highlights in color images. *Int J Comput Vis* 2(1):7–32
13. Koenderink J, van Doorn A (1980) Photometric invariants related to solid shape. *Optica Acta* 27:981–996
14. Narasimhan S, Ramesh V, Nayar S (2003) A class of photometric invariants: Separating material from shape and illumination. In: Proceedings of the IEEE international conference on computer vision, Nice
15. Nayar S, Bolle R (1996) Reflectance based object recognition. *Int J Comput Vis* 17(3):219–240
16. Tan R, Ikeuchi K (2005) Separating reflection components of textured surfaces using a single image. *IEEE Trans Pattern Anal Mach Intell* 27(2):178–193
17. Wolff L, Fan J (1994) Segmentation of surface curvature with a photometric invariant. *J Opt Soc Am A* 11(11): 3090–3100
18. Zickler T, Mallick S, Kriegman D, Belhumeur P (2008) Color subspaces as photometric invariants. *Int J Comput Vis* 79(1):13–30

Photometric Stereo

Ronen Basri

Department of Computer Science And Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel

Related Concepts

► [Active Stereo Vision](#); ► [Lambertian Reflectance](#)

Definition

Photometric stereo is the problem of recovering the 3-dimensional shape of a stationary scene given a collection of images of the scene taken under variable lighting conditions.

Background

The amount of light reflected by an object is a function of the incoming light, the shape of the object,

and its material properties. As a function of shape, the light reflected by each surface point depends on the inclination of the surface (the surface normal) at that point relative to the light sources. Early work in photometric stereo [1] assumes that the lighting conditions are given and that the material properties of the object are known. A collection of images of an object taken under varying lighting conditions (with both the object and camera stationary) can then be used to compute the normals to the surface of the object and subsequently its 3-dimensional shape. More recent photometric stereo methods can additionally recover the lighting conditions and material properties as part of the process.

Photometric stereo (PS) can be contrasted with shape from shading (SFS), the problem of recovering the shape of an object from a *single* image [2]. The problem of SFS is generally ill posed, and its solution typically requires complete knowledge of lighting and material properties. In addition, SFS is often cast as a partial differential equation (PDE), and so its solution relies on boundary conditions. These boundary conditions generally require knowledge of the 3D coordinates of a subset of the points on the sought shape. In contrast, by using a collection of images, PS generally leads to more robust solutions; it can often be solved also in the absence of knowledge of lighting conditions or material properties, requires no boundary conditions, and its solution is generally algebraic.

Theory

Let $I_1(x, y), I_2(x, y), \dots, I_k(x, y)$, $(x, y) \in \Omega \subset \mathbb{R}^2$, be a collection of k images depicting a stationary scene pictured by a stationary camera. Suppose further that each image I_i is taken under different lighting, which is denoted by L_i . The objective of photometric stereo is to recover the 3D shape of the scene, represented by $z(x, y)$, which depicts the depth value z at each point (x, y) .

In general terms, let $\hat{\mathbf{n}}(x, y) \in \mathcal{S}^2$ denote the normal to the surface at $(x, y, z(x, y))$ ($\|\hat{\mathbf{n}}(x, y)\| = 1$), and let $\rho(x, y)$ denote the reflectance properties of each scene point. The light reflected by a scene point $(x, y, z(x, y))$ is determined by the surface normal $\hat{\mathbf{n}}(x, y)$, the reflectance properties $\rho(x, y)$, and the lighting conditions L_i and can thus be expressed as a function $R(\hat{\mathbf{n}}, \rho, L_i)$. (Note that this expression

does not model the effects of cast shadows or inter-reflections, as these effects depend more globally on the shape of the observed object.) Each image then provides a constraint on the surface normal of the form $I_i(x, y) = R(\hat{\mathbf{n}}, \rho, L_i)$. With sufficiently many images these constraints can be used to recover the surface normal at each point, $\hat{\mathbf{n}}$, and, subsequently, the surface $z(x, y)$.

For a concrete example, due to the pioneering work of Woodham [1], consider a scene composed of a matte surface whose reflectance is described by the Lambertian model, and suppose that in each image the scene is illuminated by a single directional source. Specifically, a *directional source* (also referred to as a *point source at infinity*) is expressed by a vector $\mathbf{l}_i \in \mathbb{R}^3$ in the direction of the source whose magnitude represents the light intensity. According to the Lambertian law, the light reflected by a point $(x, y, z(x, y))$ with normal $\hat{\mathbf{n}}(x, y)$ and albedo $\rho(x, y)$ (the *albedo* of a scene point is a material property representing the fraction of incident light that is reflected by the surface at that point) is given by

$$I_i(x, y) = \rho(x, y) \mathbf{l}_i^T \hat{\mathbf{n}}(x, y), \quad (1)$$

where the superscript T denotes the transpose operator. This law is applicable as long as $\mathbf{l}_i^T \hat{\mathbf{n}}(x, y) \geq 0$; otherwise, the point is in shadow (commonly referred to as *attached shadow*). Note that for this equation, it is assumed that the camera is calibrated so that the amount of light recorded by the camera is identical to the amount of light reflected by the surface.

Suppose now that k such images I_1, \dots, I_k are obtained, in which the scene is illuminated by directional light sources $\mathbf{l}_1, \dots, \mathbf{l}_k$, respectively. Let $\mathbf{I}_i \in \mathbb{R}^p$ be a vector arrangement of the pixel intensities in $I_i(x, y)$, $\mathbf{I}_i = (I_i(x_1, y_1), \dots, I_i(x_p, y_p))^T$, where p denotes the number of discrete pixels in Ω . Let $M = [I_1, \dots, I_k]^T$ be a $k \times p$ matrix whose rows include the intensity measurements in all images. Let $L = [\mathbf{l}_1, \dots, \mathbf{l}_k]^T$ be a $k \times 3$ matrix containing all k light sources. Finally, let $S = [\rho(x_1, y_1) \hat{\mathbf{n}}(x_1, y_1), \dots, \rho(x_p, y_p) \hat{\mathbf{n}}(x_p, y_p)]$ be a $3 \times p$ matrix whose columns contain the surface normals of the scene points scaled by the corresponding albedo (with the columns of M and S organized in correspondence). M is referred to as the measurement matrix, L as the lighting matrix, and S as the shape matrix. Then the Lambertian law for all the input images can

be summarized by the following matrix equation:

$$M = LS. \quad (2)$$

Assuming that all the lighting directions and intensities are known (so L is known) and that L includes three linearly independent rows, then S can be determined by solving the linear, possibly overdetermined equation system above. In particular, if $k = 3$ and L is invertible, then

$$S = L^{-1}M. \quad (3)$$

Once S is recovered the surface normals of the scene can be recovered by normalizing each column of S , i.e., let \mathbf{s}^i denotes the i 'th column of S then

$$\hat{\mathbf{n}}(x_i, y_i) = \frac{\mathbf{s}^i}{\|\mathbf{s}^i\|} \quad (4)$$

$$\rho(x_i, y_i) = \|\mathbf{s}^i\|. \quad (5)$$

Note that the collection of surface normals of a differentiable shape uniquely determines the depth values of its bounding surface up to an additive constant.

Integrability

To further obtain an explicit recovery of the depth values $z(x, y)$, *integrability* (or *consistency*) of the recovered normals can be imposed. For integrability the surface $z(x, y)$ is assumed to be differentiable, and the normals can be expressed in terms of its derivatives. Under these conditions, it can be readily shown that the normal at a point $(x, y, z(x, y))$ is given by

$$\hat{\mathbf{n}}(x, y) = \frac{(z_x, z_y, -1)}{\sqrt{z_x^2 + z_y^2 + 1}}, \quad (6)$$

where z_x and z_y represent the two partial derivatives of z , $z_x = \partial z / \partial x$ and $z_y = \partial z / \partial y$. An estimate of these partial derivatives can be obtained from the recovered shape matrix S using the ratios $z_x = -s_1/s_3$ and $z_y = -s_2/s_3$, where $s = (s_1, s_2, s_3)^T$ is the column of S corresponding to point (x, y) . Finally, a forward discretization of the partial derivatives is used to obtain the following first-order difference equations

$$z(x + h, y) - z(x, y) = h z_x(x, y) \quad (7)$$

$$z(x, y + h) - z(x, y) = h z_y(x, y), \quad (8)$$

where h is the (known) meshsize (often $h = 1$). This is an overdetermined system of linear equations in $z(x, y)$ and can be solved to least squares up to an additive constant which can be determined, e.g., by setting $z(x_0, y_0) = 0$ for some point (x_0, y_0) .

Simultaneous Recovery of Shape and Light

The method above assumes that the intensity and direction of the light sources are known and uses this information to recover the shape and albedo of the object. However, these derivations can still be used even when the lighting conditions are unknown. The main observation is that in the absence of noise, the measurement matrix M has rank 3, and so it can be factored to recover the lighting and shape up to a certain ambiguity [3]. In addition, when $\text{rank}(M)$ exceeds 3, replacing it by its rank 3 approximation can eliminate at least some of the noise in the image.

To see that $\text{rank}(M)$ is (at most) 3, note that it is a product of the $k \times 3$ lighting matrix L with the $3 \times p$ shape matrix S (Eq. (2)). Hence, M can be factored using singular value decomposition (SVD) as follows. Let \tilde{M} be the best rank 3 approximation to M , i.e., $\hat{M} = \text{argmin}_{\tilde{M}} \|\tilde{M} - M\|_F$, where $\|\cdot\|_F$ denotes the Frobenius Norm of a matrix. \hat{M} can be readily computed using the SVD decomposition of M . Next, let $\hat{M} = U\Sigma V^T$ be the SVD decomposition of \hat{M} and define $\hat{L} = U\sqrt{\Sigma}$ and $\hat{S} = \sqrt{\Sigma}V^T$. Evidently, this decomposition is nonunique, as any 3×3 nonsingular matrix A can be used to obtain another valid decomposition, i.e.,

$$\hat{M} = (\hat{L}A^{-1})(A\hat{S}). \quad (9)$$

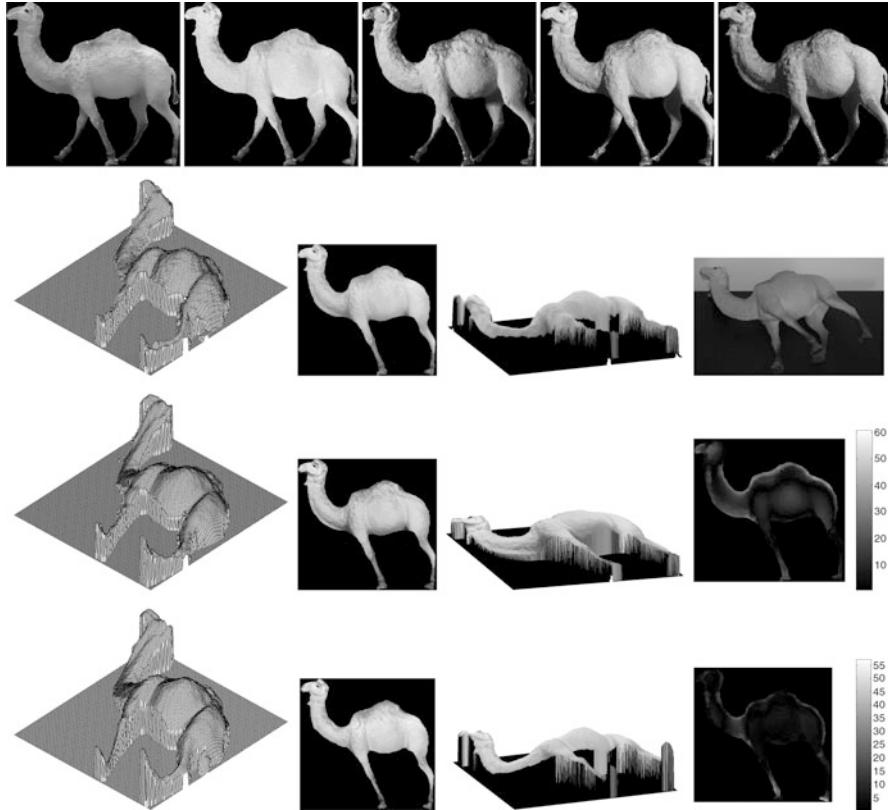
Equation (9) therefore defines a 9-parameter ambiguity – the entries of A . This ambiguity can be reduced by imposing integrability [4], as is explained below.

To impose integrability the following equation can be used:

$$z_{xy} = z_{yx}, \quad (10)$$

which holds when z is twice differentiable, $z_{xy} = \partial^2 z / (\partial x \partial y)$ and $z_{yx} = \partial^2 z / (\partial y \partial x)$. Let A denote the ambiguity matrix so that $S = A\hat{S}$. Then

$$z_x = -\frac{s_1}{s_3} = -\frac{\mathbf{a}_1^T \hat{\mathbf{s}}}{\mathbf{a}_3^T \hat{\mathbf{s}}} \quad (11)$$



Photometric Stereo, Fig. 1 *Top row:* five (of 13) images used for reconstruction. *Second row:* the shape of the camel obtained with a laser scanner (from *left* to *right*: surface, albedo, and albedo-painted surface) and an image taken from roughly the same view (*right*). *Third row:* reconstruction using the first-order

method (including shape, albedo, albedo-painted shape, and difference from the laser-scanned surface). *Bottom row:* reconstruction using the second-order method (From [5, 6], used with permission)

$$z_y = -\frac{s_2}{s_3} = -\frac{\mathbf{a}_2^T \hat{\mathbf{s}}}{\mathbf{a}_3^T \hat{\mathbf{s}}}, \quad (12)$$

where $(s_1, s_2, s_3)^T$ is a column in the unknown shape matrix S , $\hat{\mathbf{s}}$ is the corresponding column in the recovered matrix \hat{S} , and \mathbf{a}_i^T denotes the i th rows of A ($1 \leq i \leq 3$). Plugging these into (10),

$$\frac{\partial}{\partial y} \left(\frac{\mathbf{a}_1^T \hat{\mathbf{s}}}{\mathbf{a}_3^T \hat{\mathbf{s}}} \right) = \frac{\partial}{\partial x} \left(\frac{\mathbf{a}_2^T \hat{\mathbf{s}}}{\mathbf{a}_3^T \hat{\mathbf{s}}} \right), \quad (13)$$

a linear equation in the components of $A^T A$ is obtained for every column in \hat{S} :

$$\mathbf{a}_1^T \hat{\mathbf{s}}_y \mathbf{a}_3^T \hat{\mathbf{s}} - \mathbf{a}_1^T \hat{\mathbf{s}} \mathbf{a}_3^T \hat{\mathbf{s}}_y = \mathbf{a}_2^T \hat{\mathbf{s}}_x \mathbf{a}_3^T \hat{\mathbf{s}} - \mathbf{a}_2^T \hat{\mathbf{s}} \mathbf{a}_3^T \hat{\mathbf{s}}_x, \quad (14)$$

with $\hat{\mathbf{s}}_x = \partial \hat{\mathbf{s}} / \partial x$ and $\hat{\mathbf{s}}_y = \partial \hat{\mathbf{s}} / \partial y$. This (generally overdetermined) system of equations determines A up

to a *generalized bas-relief* (GBR) transformation [4], i.e., a depth estimate $\hat{z}(x, y)$ is obtained that is related to the original depth $z(x, y)$ by

$$\hat{z}(x, y) = \alpha x + \beta y + \gamma z(x, y), \quad (15)$$

with arbitrary constants α , β , and γ .

Photometric Stereo with General Lighting

The derivations above assume that each of the input images is produced by illuminating the object by a single directional source. More recently, Basri et al. [5, 6] proposed an approach to extend this to handle Lambertian objects illuminated by arbitrary combinations of directional and extended light sources. Their algorithm is based on the observation that the light reflected by Lambertian objects, as a function of the surface normal, is a low-pass filtered version of the surrounding

ambient light [7–9]. This allows one to describe the images of Lambertian objects, with great accuracy, as linear combinations of a small set (typically 4 or 9) of basis images, which are determined by the low-order spherical harmonic functions of the surface normal. This linear representation can now be exploited to construct factorization algorithms to photometric stereo under general lighting. The experimental results below show an example of a reconstruction achieved with this approach.

Application

Three-dimensional reconstruction is one of the fundamental tasks of computer vision. Photometric stereo is a reliable method for reconstruction. It is however used mostly in *laboratory conditions* that are partly controlled. This is because its required input should include a stationary scene under variable lighting conditions.

Open Problems

The majority of existing photometric stereo methods are designed to handle Lambertian objects. Initial work has been devoted to handling objects that exhibit specular reflectance, either simply by removing highlighted pixels [10] or by a detailed modeling of reflectance [11, 12]. Another challenge is to model the effects of cast shadows and interreflections (see, e.g., [13]). Finally, a challenging problem is to recover the 3D shapes of objects when the objects are moving with respect to a light source. Preliminary work in this subject can be found in [14–17].

Experimental Results

Figure 1 shows a photometric stereo reconstruction obtained with the algorithms proposed in [5, 6] and a comparison to a laser scan of the same object. In this experiment, 13 images of a camel-shaped doll were obtained under varying lighting conditions. The lighting setting was general and involved a number of sources along with reflections from surrounding objects. A factorization method was applied to recover both the lighting and the 3-dimensional shape of the

doll. Two results are shown. In the first case reflectance was approximated by a first-order harmonic approximation. This approximation is analogous to assuming that the lighting setting includes a single directional source in addition to a uniform, ambient source. In the second case reflectance was approximated by the more accurate, second-order harmonic approximation. The figure shows a subset of the input images and the laser scanned reconstruction along with the shapes and albedos produced by the photometric stereo algorithms. It can be seen that both methods managed to recover the shape of the camel correctly (the results produced by the second-order method are slightly more accurate), as can be judged by comparing the reconstructions to the shape produced by the laser scanner.

References

1. Woodham RJ (1980) Photometric method for determining surface orientation from multiple images. Opt Eng 19(1):139–144
2. Horn BKP (1975) Obtaining shape from shading information. The psychology of computer vision. McGraw-Hill, New York
3. Hayakawa H (1994) Photometric stereo under a light source with arbitrary motion. J Opt Soc Am A 11(11):3079–3089
4. Belhumeur PN, Kriegman DJ, Yuille AL (1999) The bas-relief ambiguity. Int J Comput Vis 35(1):33–44
5. Basri R, Jacobs D (2001) Photometric stereo with general, unknown lighting. In: IEEE conference on computer vision and pattern recognition (CVPR), Kauai, vol II, pp 374–381
6. Basri R, Jacobs D, Kemelmacher I (2007) Photometric stereo with general, unknown lighting. Int J Comput Vis 72(3):239–257
7. Basri R, Jacobs D (2001) Lambertian reflectance and linear subspaces. In: Proceedings of the IEEE international conference on computer vision, Vancouver, vol II, pp 383–390
8. Basri R, Jacobs D (2003) Lambertian reflectance and linear subspaces. IEEE Trans Pattern Anal Mach Intell 25(2): 218–233
9. Ramamoorthi R, Hanrahan P (2001) On the relationship between radiance and irradiance: determining the illumination from images of convex lambertian object. J Opt Soc Am A 18:2448–2459
10. Coleman EN, Jain R (1982) Obtaining 3-dimensional shape of textured and specular surfaces using four-source photometry. Comput Graph Image Process 18(4):309–328
11. Ikeuchi K (1981) Determining surface orientations of specular surfaces by using the photometric stereo method. IEEE Trans Pattern Anal Mach Intell 3(6):661–669
12. Georghiades A (2003) Incorporating the torrance and sparrow model of reflectance in uncalibrated photometric stereo. In: Proceedings of the IEEE international conference on computer vision, Nice, pp 816–823
13. Nayar SK, Ikeuchi K, Kanade T (1991) Shape from interreflections. Int J Comput Vis 6(13):173–195

14. Basri R, Frolova D (2008) A two-frame theory of motion, lighting and shape. In: IEEE conference on computer vision and pattern recognition (CVPR), Anchorage
15. Joshi N, Kriegman D (2007) Shape from varying illumination and viewpoint. In: Proceedings of the IEEE international conference on computer vision, Rio de Janeiro
16. Simakov D, Frolova D, Basri R (2003) Dense shape reconstruction of a moving object under arbitrary, unknown lighting. In: Proceedings of the IEEE international conference on computer vision, Nice, pp 1202–1207
17. Adato Y, Vasilyev Y, Zickler T, Ben-Shahar O (2010) Shape from specular flow. PAMI 32(11):2054–2070

Photon, Poisson Noise

Samuel W. Hasinoff
Google, Inc., Mountain View, CA, USA

Synonyms

[Schott noise](#); [Shot noise](#)

Related Concepts

► [Sensor Fusion](#)

Definition

Photon noise, also known as Poisson noise, is a basic form of uncertainty associated with the measurement of light, inherent to the quantized nature of light and the independence of photon detections. Its expected magnitude is signal dependent and constitutes the dominant source of image noise except in low-light conditions.

Background

Image sensors measure scene irradiance by counting the number of discrete *photons* incident on the sensor over a given time interval. In digital sensors, the photoelectric effect is used to convert photons into electrons, whereas film-based sensors rely on photosensitive chemical reactions. In both cases, the independence of random individual photon arrivals leads to *photon noise*, a signal-dependent form of uncertainty that is a property of the underlying signal itself.

In computer vision, a widespread approximation is to model image noise as signal *independent*, often using a zero-mean additive Gaussian. Though this simple model suffices for some applications, it is physically unrealistic. In real imaging systems, photon noise and other sensor-based sources of noise contribute in varying proportions at different signal levels, leading to noise which is dependent on scene brightness. Understanding photon noise and modeling it explicitly is especially important for low-level computer vision tasks treating noisy images [2, 8] and for the analysis of imaging systems that consider different exposure levels [1, 4, 10] or sensor gains [5].

Theory

Individual photon detections can be treated as independent events that follow a random temporal distribution. As a result, photon counting is a classic Poisson process, and the number of photons N measured by a given sensor element over a time interval t is described by the discrete probability distribution

$$\Pr(N = k) = \frac{e^{-\lambda t} (\lambda t)^k}{k!}, \quad (1)$$

where λ is the expected number of photons per unit time interval, which is proportional to the incident scene irradiance. This is a standard Poisson distribution with a rate parameter λt that corresponds to the expected incident photon count. The uncertainty described by this distribution is known as *photon noise*.

Because the incident photon count follows a Poisson distribution, it has the property that its variance is equal to its expectation, $E[N] = \text{Var}[N] = \lambda t$. This shows that photon noise is signal dependent and that its standard deviation grows with the square root of the signal.

In practice, photon noise is often modeled using a Gaussian distribution whose variance depends on the expected photon count [1, 2, 4, 5, 8, 10],

$$N \sim \mathcal{N}(\lambda t, \lambda t). \quad (2)$$

This approximation is typically very accurate. For small photon counts, photon noise is generally dominated by other signal-independent sources of noise, and for larger counts, the central limit theorem ensures that the Poisson distribution approaches a Gaussian.

Since photon noise is derived from the nature of the signal itself, it provides a lower bound on the uncertainty of measuring light. Even under ideal imaging conditions, free from all other sensor-based sources of noise (e.g., read noise), any measurement would still be subject to photon noise. When photon noise is the only significant source of uncertainty, as commonly occurs in bright photon-rich environments, imaging is said to be *photon limited*.

In general, the only way to reduce the effect of photon noise is to capture more signal. The ratio of signal to photon noise grows with the square root of the number of photons captured, $\sqrt{\lambda t}$. This shows that photon noise, while growing in absolute terms with signal, is relatively weaker at higher signal levels. However, in order to capture more photons, longer exposure times are required, and the number of photons captured in a single shot is limited by the full well capacity of the sensor. Note that while squeezed coherence lasers and other forms of nonclassical light can achieve amplitude noise below the photon noise limit [11], such exotic lighting configurations are typically not relevant for computer vision applications.

In digital sensors, a related source of noise that also follows a Poisson distribution is dark current noise. Dark current refers to “phantom” photon counts due thermal energy causing the sensor to release electrons at random. While photon noise is a property of the signal itself, dark current comes from the embodiment of the sensor and depends on both temperature and exposure time.

Application

Photon noise is inherent to the measurement of light, has no parameters to be calibrated, and is independent of other noise sources. As a result, the effect of photon noise on imaging can be characterized using the *radiometric response function* that relates the photon count and the expected pixel intensity [3, 6].

To handle the signal dependence caused by photon noise, a first step is to estimate the noise variance for each pixel. This can be approximated in a straightforward way by inverting the forward model for imaging noise [5, 6, 9]. For increased accuracy, several other factors can be taken into account as well: the coupling between signal and noise leads to a recursive estimation [3]; pixels near saturation have reduced variance which can lead to bias [2, 8]; and on-camera

processing such as demosaicing may introduce spatial correlation [8].

Image processing methods that explicitly incorporate more realistic signal-dependent models of noise, either calibrated [3, 5, 6] or inferred from the image [7, 8], adapt naturally to pixels of different intensities. As a result, for a variety of computer vision tasks such as denoising [3, 8] and edge detection [7], these methods can perform better than those handicapped by the assumption of scene-independent noise.

An alternative approach for handling signal-dependent noise is to transform the image using a *variable-stabilizing transformation* that amounts to applying per pixel nonlinearities that effectively reduce the signal dependence [2, 9]. Because the transformed signal approximates one with signal-independent noise, it may be processed using methods that assume a simpler noise model.

References

1. Agrawal A, Raskar R (2009) Optimal single image capture for motion deblurring. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Miami, pp 2560–2567
2. Foi A, Trimeche M, Katkovnik V, Egiazarian K (2008) Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data. *IEEE Trans Image Process* 17(10):1737–1754
3. Granados M, Adjin B, Wand M, Theobalt C, Seidel H-P, Lensch Hendrik PA (2010) Optimal HDR reconstruction with linear digital cameras. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Francisco, pp 215–222
4. Hasinoff SW, Kutulakos KN, Durand F, Freeman WT (2009) Time-constrained photography. In: Proceedings of the IEEE international conference on computer vision, Kyoto, pp 333–340
5. Hasinoff SW, Durand F, Freeman WT (2010) Noise-optimal capture for high dynamic range photography. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Francisco, pp 553–560
6. Healey GE, Kondepudy R (1994) Radiometric CCD camera calibration and noise estimation. *IEEE Trans Pattern Anal Mach Intell* 16(3):267–276
7. Hwang Y, Kim J-S, Kweon I-S (2007) Sensor noise modeling using the Skellam distribution: application to the color edge detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Minneapolis, pp 1–8
8. Liu C, Szeliski R, Kang SB, Lawrence Zitnick C, Freeman WT (2008) Automatic estimation and removal of noise from a single image. *IEEE Trans Pattern Anal Mach Intell* 30(2):299–314
9. Prucnal PR, Saleh BEA (1981) Transformation of image-signal-dependent noise into image-signal-independent noise. *Opt Lett* 6(7):316–318

10. Treibitz T, Schechner YY (2009) Polarization: beneficial for visibility enhancement? In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Miami, pp 525–532
11. Vahlbruch H, Mehmet M, Chelkowski S, Hage B, Franzen A, Lastzka N, Goßler S, Danzmann K, Schnabel R (2008) Observation of squeezed light with 10-dB quantum-noise reduction. *Phys Rev Lett* 100(3):033602

Picture Understanding

► [Line Drawing Labeling](#)

Piecewise Polynomial

► [Splines](#)

Pinhole Camera

► [Perspective Camera](#)

Pinhole Camera Model

Peter Sturm
INRIA Grenoble Rhône-Alpes, St Ismier Cedex,
France

Related Concepts

► [Affine Camera](#); ► [Camera Calibration](#); ► [Center of Projection](#); ► [Focal Length](#); ► [Image Plane](#); ► [Optical Axis](#); ► [Weak Perspective Projection](#)

Definition

The pinhole camera model is the basic camera model used in computer vision. Its name originates from the concept of pinhole camera and it models perspective projections.

Background

The pinhole model is the basic camera model used in computer vision. Its name stems from the concept of

pinhole camera [1] (also related to the *camera obscura* [2]): usually, a closed box into which a single tiny hole is made with a pin, through which light may enter and hit a photosensitive surface inside the box (cf. Fig. 1). Pinhole cameras allow to take photographs of objects, which usually requires long exposure times due to the small aperture. The principles behind pinhole cameras and the *camera obscura* have been known, at least partially, since the fourth century BC [2].

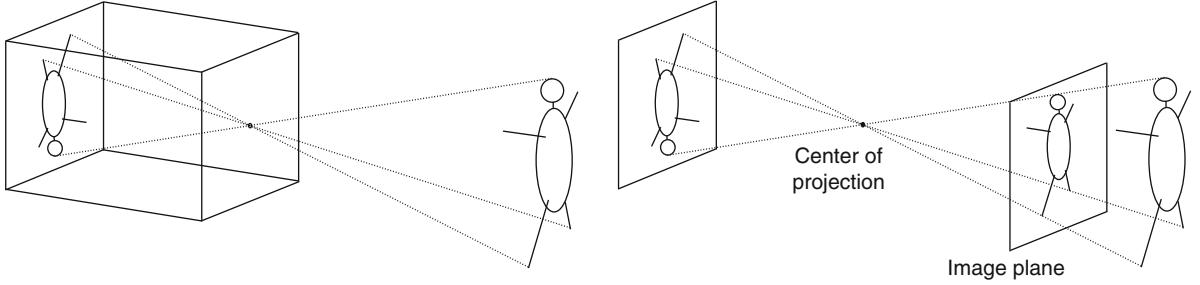
The pinhole camera model mimics the geometrical projection carried out by a pinhole camera, as follows (see also Fig. 1). The entire optics and aperture of a camera are reduced to a single point – the *optical center* or *center of projection*. The photosensitive surface is assumed to be planar and is, geometrically, represented by the so-called *image plane*. To determine where a 3D point is depicted in the image, it suffices to construct a straight line from that point and going through the optical center (one may consider this as a light ray). This line's intersection with the image plane gives the desired image point of the 3D point. A key property of this model is that points that are collinear in 3D get imaged to image points that are also collinear.

The pinhole model is simple and neglects many aspects of true cameras. For instance, apertures are finite and thus most 3D points are not imaged in a unique image point but within a finite area of the image plane. Likewise, pixels in digital cameras gather incoming light across finite areas. Thus, the pinhole model does not tell anything about blur, point spread functions, quantization, or other effects such as vignetting that occur in true cameras.

Other effects that are not modeled by it are radial or other geometric distortions, which usually occur with small focal lengths and that violate the above property of mapping collinear points in 3D to collinear points in the image. To handle such distortions, the pinhole model may be extended by adding models for radial distortion for instance, or by using different models altogether, such as becomes necessary for fisheye cameras. Despite the above limitations, the pinhole model is already a good approximation for many regular cameras.

Theory

Above, the geometrical projection mapping described by the pinhole model is described. Before translating



Pinhole Camera Model, Fig. 1 *Left:* sketch of a pinhole camera. *Right:* the two basic constituents of the pinhole camera model are the center of projection and the image plane. The true image plane shows inverse images of objects. To ease drawings, a common convention is to replace the true image plane by a

virtual one between the center of projection and the scene, at the same distance from the center as the original image plane and parallel to it. This image plane produces an identical image, but that is “correctly” oriented

this into an algebraic formulation, a few notations are introduced. The *focal length* f is the distance between the center of projection and the image plane. The line passing through the optical center and that is orthogonal to the image plane is called *optical axis*. The intersection point of the optical axis and the image plane is the *principal point*. These definitions are only based on the optical center and the image plane. When considering digital cameras, one in addition needs to take into account the layout of pixels in the image plane. The usual layout consists of a regular rectangular grid (although other arrangements, such as log-polar ones, were also experimented with [3, 4]): pixels are arranged into rows and columns. Let k_u and k_v be the column-wise and row-wise density of pixels, respectively (e.g., measured as number of pixels per millimeter). The value of k_u/k_v is also called the *aspect ratio* of a camera. Usually, the two densities are equal to one another, that is, cameras have a unit aspect ratio, but especially with video cameras, this should not be taken for granted.

The following coordinate systems are used to derive algebraic expressions for the pinhole model (Fig. 2). The *camera coordinate system* has its origin in the optical center. A usual convention is to define the Z -axis as being coincident with the camera’s optical axis and the X/Y -axes to be parallel to the columns and rows of pixels in the image plane. The *image coordinate system* has its origin in the principal point. The x -axis and y -axis are parallel to the X -axis and Y -axis, respectively, of the camera coordinate system. Typically, the camera and image coordinate systems use metric units. To represent the final digital image, one defines the *pixel coordinate system*. Its origin usually

lies in one of the image area’s corners. Let its coordinates, relative to the image coordinate system, be $(-x_0, -y_0)$. The two axes, u and v , are parallel to x and y , respectively; their unit is (number of) pixels, counted row-wise and column-wise, respectively. Note that the above choices of coordinate systems are not unique; other choices are possible, for example, for the X and Y axes and the origin of the pixel coordinate system, although care should be taken to use right-handed systems. The following equations will have to be adapted accordingly.

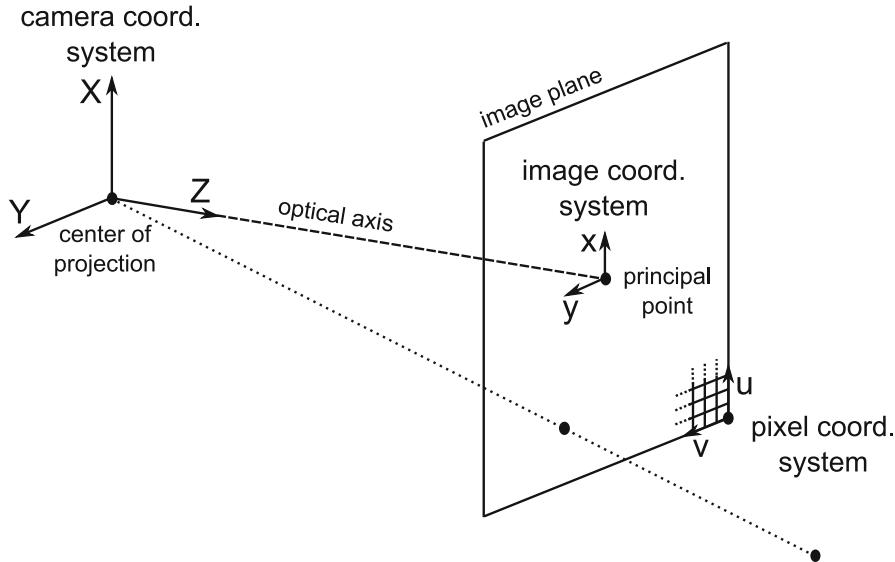
With the above definitions, the projection carried out by the pinhole camera model can be formulated as follows. Let (X, Y, Z) be the coordinates of a 3D point, expressed in the camera coordinate system. From the comparison of similar triangles, one obtains the image point’s coordinates in the image coordinate system as:

$$x = f \frac{X}{Z} \quad y = f \frac{Y}{Z}.$$

The coordinates in the pixel coordinate system are obtained by applying a translation corresponding to the shift of origin and scalings corresponding to the conversion from metric coordinates to pixel ones:

$$\begin{aligned} u &= k_u(x + x_0) = k_u f \frac{X}{Z} + k_u x_0 \\ v &= k_v(y + y_0) = k_v f \frac{Y}{Z} + k_v y_0. \end{aligned}$$

It is often useful to express these projection equations using homogeneous coordinates for the 3D and image points:



Pinhole Camera Model, Fig. 2 The components of the pinhole camera model. The distance between the center of projection and the image plane is the focal length f

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \sim \begin{pmatrix} k_u f & 0 & k_u x_0 & 0 \\ 0 & k_v f & k_v y_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \quad (1)$$

where \sim signifies equality up to scale of vectors or matrices.

It is common to replace the above “metric” entities f, k_u, k_v, x_0 , and y_0 by equivalent ones given in pixel units:

$$\alpha_u = k_u f \quad \alpha_v = k_v f \quad u_0 = k_u x_0 \quad v_0 = k_v y_0.$$

Here, α_u and α_v measure the focal length in number of pixels (column-wise and row-wise, respectively) and (u_0, v_0) are the coordinates of the principal point given in the pixel coordinate system. These four entities are also called the *intrinsic parameters* of the pinhole camera model, since they describe what happens “inside” a camera. They are often grouped together in an upper triangular so-called *calibration matrix*:

$$K = \begin{pmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{pmatrix}. \quad (2)$$

Sometimes, a fifth intrinsic parameter is added to the model – a so-called skew parameter that replaces

the zero in the first row of the calibration matrix and which allows to model pixel layouts with skewed axes or cameras with desynchronized pixel readout. With modern digital cameras, these issues can usually be neglected though.

In order to model cameras in motion or multi-camera systems, one needs to describe the position and orientation of a camera. To do so, a final coordinate system, the *world coordinate system*, is considered. This may be attached to a physical object, for instance, an object to be inspected; otherwise, it can be assumed arbitrarily, as long as it remains fixed throughout an application. Let t be the coordinates of the center of projection in the world coordinate system and let the rotation matrix R represent the camera’s orientation. Then, a 3D point is mapped from the world to the camera coordinate system, as:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} R & -Rt \\ 0^T & 1 \end{pmatrix} \begin{pmatrix} X^w \\ Y^w \\ Z^w \\ 1 \end{pmatrix}, \quad (3)$$

where as above, homogeneous coordinates are used.

Putting together (Eq. 1) and (Eq. 3), we get the complete expression of the pinhole camera model:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \sim \begin{pmatrix} k_u f & 0 & k_u x_0 & 0 \\ 0 & k_v f & k_v y_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} R & -Rt \\ 0^T & 1 \end{pmatrix} \begin{pmatrix} X^w \\ Y^w \\ Z^w \\ 1 \end{pmatrix}.$$

With the above definitions of intrinsic parameters and calibration matrix (Eq. 2), this can be written more compactly as:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \sim \underbrace{KR(\text{Id}_3 \quad -\mathbf{t})}_{P} \begin{pmatrix} X^w \\ Y^w \\ Z^w \\ 1 \end{pmatrix},$$

where Id_3 is the 3×3 identity matrix. The 3×4 matrix P is called *projection matrix* or *camera matrix*.

The projection expressed by the pinhole model is a perspective projection. Even simpler camera models exist in the form of orthographic or other affine projections.

Application

The pinhole camera model, like any other camera model [5], can be used to infer geometrical information about an imaged scene, from one or more images. Some of the most common applications are sketched below. It is usually assumed that a camera is calibrated prior to an application, that is, that its intrinsic parameters are known through a *camera calibration* procedure. Most camera calibration procedures utilize a reference object of known shape, a calibration grid. However, there also exist approaches for *self-calibration* (or autocalibration or on-line calibration) that allow to compute the intrinsic parameters directly from images of an unknown scene.

Pose estimation refers to the computation of an object's position and orientation relative to a camera, or vice-versa. Here, it is usually assumed that the object's shape is known and the camera calibrated. *Motion estimation* is the determination of the camera's motion between two or more acquisitions; this is possible even if the scene is unknown, that is, if it does not contain any reference object of known shape or type. Motion estimation usually requires some amount of image matching, that is, the determination of projections of the same scene feature in different images. *3D*

modeling is usually done from two or more images of a rigid scene and also usually requires image matching, although exceptions exist, such as shape-from-shading, photometric stereo, and interactive single-view 3D modeling.

Note that none of these applications is specific to the pinhole model, although the theory underlying them has most extensively been studied for this model, as parts of perspective multi-view geometry [6]. It is also reminded, as said above, that the pinhole model neglects many aspects of image formation and does not model non-perspective image distortions. A camera model, be it the pinhole or another one, should only be used in an application if it is sure that it is appropriate for the camera used.

References

1. Wikipedia (2011) Pinhole camera. http://en.wikipedia.org/wiki/Pinhole_camera. Accessed 3 Aug 2011
2. Wikipedia (2011) Camera obscura. http://en.wikipedia.org/wiki/Camera_obscura. Accessed 5 Aug 2011
3. Tistarelli M, Sandini G (1993) On the advantage of polar and log-polar mapping for direct estimation of time-to-impact from optical flow. IEEE Trans Pattern Anal Mach Intell 15(4):401–410
4. Pardo F, Dierickx B, Scheffer D (1997) CMOS foveated image sensor: signal scaling and small geometry effects. IEEE Trans Electron Devices 44(10):1731–1737
5. Sturm P, Ramalingam S, Tardif JP, Gasparini S, Barreto J (2011) Camera models and fundamental concepts used in geometric computer vision. Found Trends Comput Graph Vis 6(1–2):1–183
6. Hartley R, Zisserman A (2004) Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, Cambridge

Planckian Locus

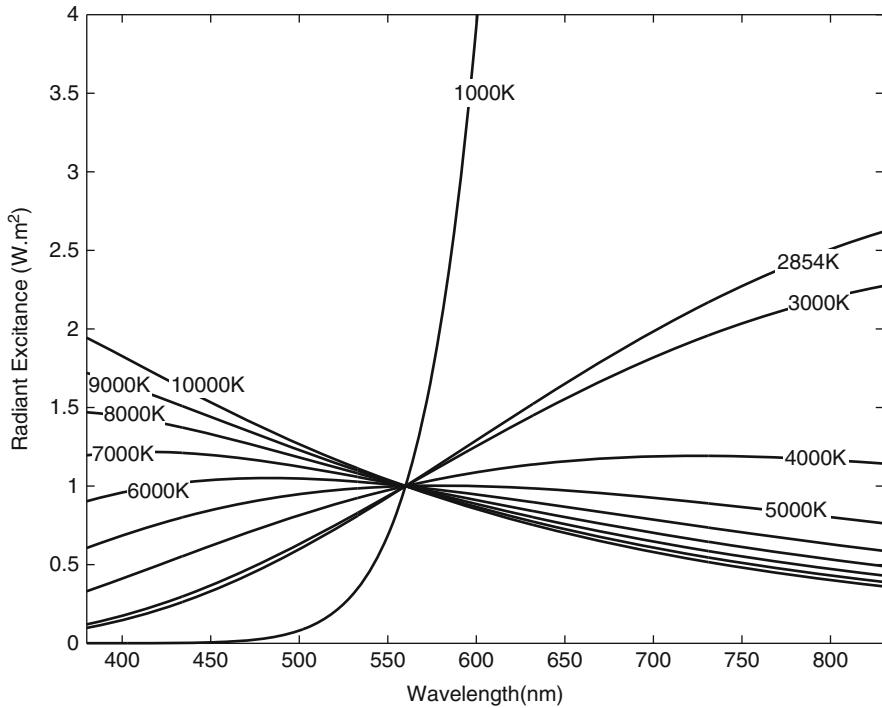
Rajeev Ramanath¹ and Mark S. Drew²

¹DLP® Products, Texas Instruments Incorporated, Plano, TX, USA

²School of Computing Science, Simon Fraser University, Vancouver, BC, Canada

Synonyms

Blackbody radiator; Thermal radiator



Planckian Locus, Fig. 1 Spectral power distributions of various blackbody radiators from 1,000 to 10,000 K, with all spectral distributions normalized to unity at 560 nm

Definition

The Planckian locus as it relates to color is the locus of points in a color space that would be followed by an incandescent blackbody radiator as its temperature changes. This locus is typically described in the CIE x, y or CIE u', v' chromaticity spaces.

Background

The CIEXYZ color space is defined by

$$\begin{aligned} X &= k \int_{\lambda} \bar{x}(\lambda) i(\lambda) r(\lambda) d\lambda \\ Y &= k \int_{\lambda} \bar{y}(\lambda) i(\lambda) r(\lambda) d\lambda \\ Z &= k \int_{\lambda} \bar{z}(\lambda) i(\lambda) r(\lambda) d\lambda, \end{aligned} \quad (1)$$

where k denotes a normalization factor that is set to 683 lumens/Watt in the case of absolute colorimetry

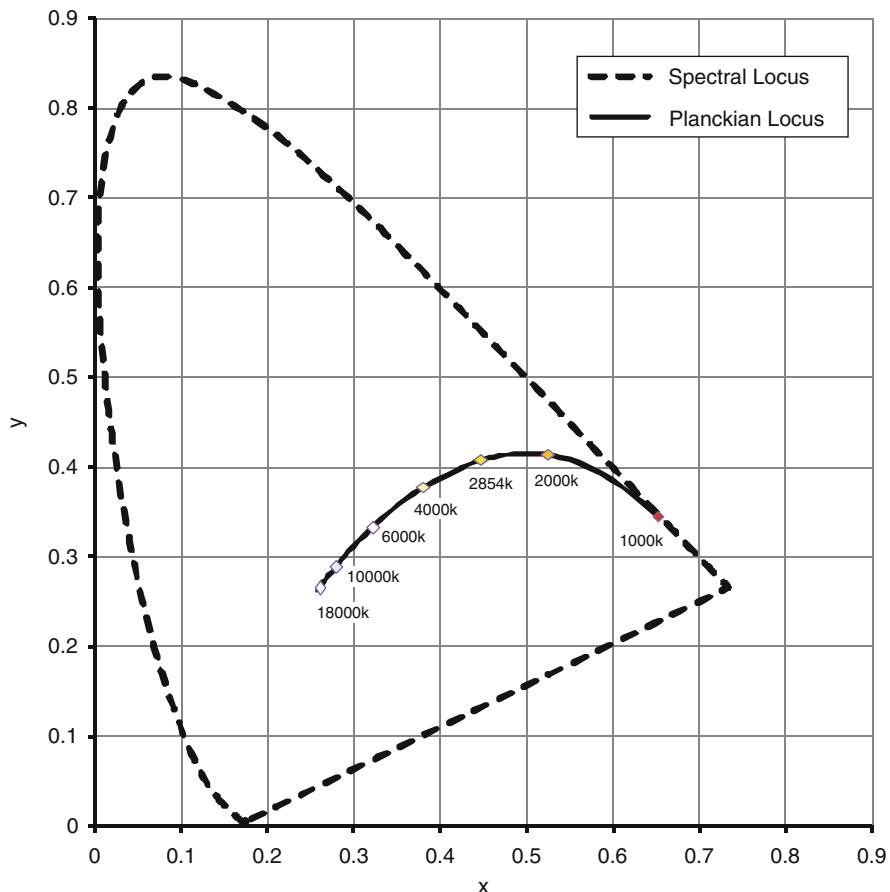
and to $100/\int_{\lambda} \bar{y}(\lambda) i(\lambda) d\lambda$ for relative colorimetry and $i(\lambda)$ denotes the spectrum of an illuminant. The functions $\bar{x}(\lambda), \bar{y}(\lambda), \bar{z}(\lambda)$ are the CIE color-matching functions.

In the case of relative colorimetry, this means that a value of $Y = 100$ denotes the brightest color – the illuminant reflecting from a perfect reflecting diffuser [1, 3].

Planck's radiation law describes the spectral distribution of radiant excitance M_e as a function of wavelength λ and temperature T and is given by Planck's Law:

$$M_e(\lambda, T) = \frac{c_1}{\lambda^5 \left[\exp\left(\frac{c_2}{\lambda T}\right) - 1 \right]} \quad (2)$$

where $c_1 = 2\pi hc^2 = 3.74183 \times 10^{-16} W.m^{-2}$, $c_2 = h.c/k = 1.4388 \times 10^{-2} m.K$ (c is the speed of light in vacuum, h is Planck's constant, k is Boltzmann's constant), and the excitance is defined in units of $W.m^{-3}$.



Planckian Locus, Fig. 2 Projection of the emission spectra of various blackbody radiators on the CIE x, y chromaticity diagram showing the locus of spectral colors along with the Planckian locus

Radiation emitted from blackbody radiators is defined by Planck's Law and is among the few radiations from sources that has its relative spectral power distribution match those of illuminants. In other words, blackbody radiators are the select few sources of illumination that match standard illuminant spectral power distributions – this may be seen as valid in the case of the equivalence of standard illuminant "A" and a blackbody with a temperature 2,856 K. Radiation from these radiators is typically seen as "white" to human observers. This makes it important to plot a locus of these points in a space such as CIEXYZ, or shown in a simpler view such as the two-dimensional space of the CIE x, y chromaticity diagram. The locus of points followed by illuminants defined by Planck's Law is called the Planckian locus. Figure 1 shows the spectral power distribution of various blackbody radiators from

1,000 to 10,000 K, with all spectral power distributions normalized to unity at 560 nm. As the blackbody gets hotter (T increases), one can see that the red content in the spectrum reduces and the blue content increases – an indication of the color as would be seen by the human observer.

The projection of the emission spectra of various blackbody radiators onto the CIEXYZ color space via Eq. (2), and then further projected onto the CIE x, y 2-D chromaticity diagram, is shown in Fig. 2, for the CIE 2-degree observer [1]. One can see that the cooler blackbody radiators (lower T) are more red in their appearance (as given by their location in the chromaticity diagram) and the hotter blackbody radiators (higher T) are more blue in their appearance – the chromaticity diagram is red at the bottom right, green at the top, and blue at the bottom left.

References

1. CIE 15:2004 (2004) Colorimetry. CIE, Vienna
2. 1998c CIE (1998) CIE standard illuminants for colorimetry. CIE, Vienna. Also published as ISO 10526/CIE/S006/E1999
3. Wyszecki G, Stiles WS (1982) Color science: concepts and methods, quantitative data and formulas, 2nd edn. Wiley, New York

Plane Sweeping

David Gallup¹ and Marc Pollefeys²

¹Google Inc., Seattle, WA, USA

²Computer Vision and Geometry Lab (CVG) – Institute of Visual Computing, Department of Computer Science, ETH Zürich, Zürich, Switzerland

Related Concepts

► [Multi-baseline Stereo](#)

Definition

Plane sweeping is a multi-view stereo algorithm notable for its efficiency, especially on modern graphics hardware (GPU).

Background

Plane sweeping addresses the stereo problem, which is to find the surface of a scene, given two or more calibrated views of the scene. Assuming that surfaces are Lambertian and that there are no occlusions, a point on the surface will have the same appearance in all views. Therefore, a conceptual solution to the stereo problem is to find the points that maximize *photoconsistency*. This can be done by searching along the viewing rays for some image. The distance to the surface along the ray is called *depth*, and the set of all depths is called a depth map. Plane sweeping is a technique for efficiently organizing this search by sweeping a plane through space. Projecting all points on a plane into a perspective camera is simply a homography transformation, which can be executed efficiently on modern graphics hardware (GPU) [8].

Binocular stereo would typically *rectify* a pair of images such that searching along a viewing ray is equivalent to looping over the pixels in a row of the other image [6]. But three or more views, in general,

cannot be rectified simultaneously. Plane sweeping avoids this problem by operating in 3D, searching along rays, one plane at a time, instead of searching image pixels. However, the proper sampling of 3D space is an issue. The image signal is sampled in 2D image space, and the sampling of the plane sweep must map to a similar sampling rate in image space. The right sampling is important for both correctness and efficiency.

Theory

The input to the plane sweep algorithm is a set of *views*. Let a view be defined as an image plus camera parameters. For simplicity, choose one view to be a *reference view*. All other images will be compared against the reference image to measure photoconsistency. The output of the algorithm will be a depth map for the reference view. Let the reference camera be identity, that is, its camera matrix is

$$\mathbf{P}_{\text{ref}} = \mathbf{K}_{\text{ref}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (1)$$

Let $\mathcal{I} = \{I_1, I_2, \dots, I_N\}$ be a set of matching views whose cameras are $\mathcal{P} = \{\mathbf{P}_1, \mathbf{P}_2, \dots, \mathbf{P}_N\}$, where

$$\mathbf{P}_i = \mathbf{K}_i [\mathbf{R}_i \quad \mathbf{t}_i]. \quad (2)$$

Let a plane π be defined as

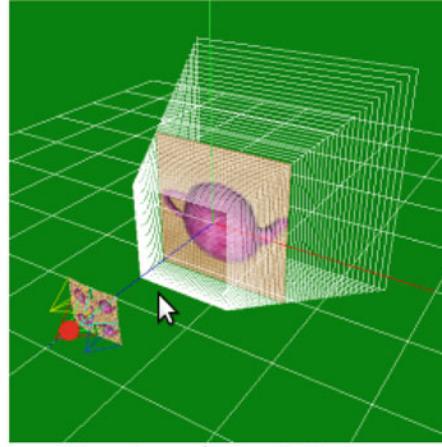
$$\pi = [\mathbf{n}_x \quad \mathbf{n}_y \quad \mathbf{n}_z \quad d]^T. \quad (3)$$

The plane can be swept through space in direction \mathbf{n} by varying d . For a given plane, photoconsistency will be evaluated on the plane at each point. This can be accomplished by projecting all images onto the plane and comparing the image values. Ultimately, it is desired to compute photoconsistency in the image to be reconstructed, and so, each image will be projected onto the plane and then into reference image. This transformation is known as a plane homography. The equation for the plane homography from I_{ref} to I_i via the plane π is

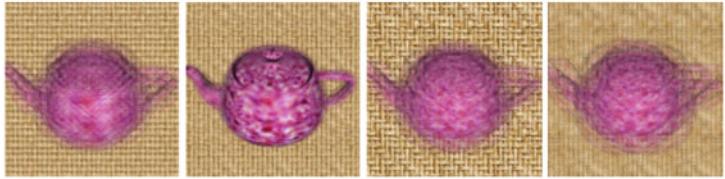
$$\mathbf{H}_i^\pi = \mathbf{K}_i \left(\mathbf{R}_i - \frac{\mathbf{t}_i \mathbf{n}^\top}{d} \right) \mathbf{K}_{\text{ref}}^{-1}. \quad (4)$$

Plane Sweeping, Fig. 1

Plane sweep of a synthetic scene. As the plane sweeps from front to back, you can see the alignment of the teapot followed by the mat behind it



Four views of a synthetic scene.



The warped images blended together at different stages of the plane sweep.

The homography \mathbf{H}_i^π maps from the reference image to the image i so that pixel (x, y) in the warped image can be computed as

$$\begin{aligned}\tilde{I}_i(x, y) &= I \left(\frac{\tilde{x}}{\tilde{w}}, \frac{\tilde{y}}{\tilde{w}} \right) \text{ where } [\tilde{x} \quad \tilde{y} \quad \tilde{w}]^\top \\ &= \mathbf{H}_i^\pi [x \quad y \quad w]^\top.\end{aligned}\quad (5)$$

The photoconsistency at any point in the reference view can be computed using any number of matching scores. Common choices include the sum of squared differences (SSD), sum of absolute differences (SAD), and normalized cross correlation (NCC). These scores are window-based because they operate over a neighborhood or correlation window. This produces a more robust matching score at the expense of some overextension artifacts [3]. As an example, the SSD can be computed as follows:

$$\begin{aligned}SSD(x, y) &= \sum_{i=1}^N \sum_{(i,j) \in \mathcal{N}} (I_{\text{ref}}(x+i, y+j) \\ &\quad - \tilde{I}_i(x+i, y+j))^2.\end{aligned}\quad (6)$$

The SSD is actually a photoconsistency *cost*, and it will be *minimized* by points near the surface. Defining the neighborhood \mathcal{N} in the space of the reference image assumes that points in the neighborhood are on the plane π . For scenes with highly slanted surfaces, better results can be achieved by sweeping the plane through space in different directions (different \mathbf{n}). Searching all directions is slow, but in some cases, like man-made scenes, a few dominant directions are sufficient [1].

Equation 5 assumes that all matching views are unoccluded, but this is often not the case, and an occluded view can produce an arbitrarily high matching cost. Because plane sweeping can use many views, occlusion handling amounts to summing over some subset of views likely to be unoccluded. In the case where the camera path is close to linear, the subset can be either the previous or the subsequent half-set of matching views. A more general solution is to assume the best 50% of scores are unoccluded [4].

While sweeping the plane through space, the plane with the best photoconsistency score is recorded per pixel. The depth for each pixel can be computed by intersecting the viewing ray with the recorded best plane. The warped images at various positions in the plane sweep are shown in Fig. 1.

Proper sampling of the 3D space during the plane sweep is important for both correctness and efficiency. Sampling too sparsely could miss the photoconsistency optimum, and sampling too densely is inefficient. During the plane sweep, the plane should move so that the motion of the warped images is no more than 1 pixel (or the desired image sampling rate). Not all pixels move at the same rate, but measuring the corner pixels of the reference image is sufficient. Since the planar warps are linear, the interior points are linear combinations of corners and are therefore bounded. Sampling can be sped up by using downsampled images, effectively making pixels larger. Varying the resolution as well as varying the set of matching images (to control the baseline) can be used to control the sampling rate to achieve an optimal balance between efficiency and precision [2].

Application

The plane sweep algorithm has been popular for several real-time and large-scale systems, where efficiency is a key concern. The Urbanscape system is able to process VGA resolution video at over 30 frames per second, allowing it to reconstruct entire cities from street-level video in a matter of hours [1, 5].

A plane sweep approach was used in [7] for real-time view synthesis which allowed virtual views of the scene to be rendered from camera positions that were not physically captured. This algorithm differs slightly from plane sweep stereo in that the most consistent color, rather than the depth, is returned.

References

1. Gallup D, Frahm J-M, Mordohai P, Qingxiong Y, Pollefey M (2007) Real-time plane-sweeping stereo with multiple sweeping directions. In: Computer vision and pattern recognition (CVPR). IEEE, Piscataway, Minneapolis, Minnesota
 2. Gallup D, Frahm J-M, Mordohai P, Pollefey M (2008) Variable baseline/resolution stereo. In: Computer vision and pattern recognition (CVPR). IEEE, Piscataway, Anchorage, Alaska
 3. Kanade T, Okutomi M (1994) A stereo matching algorithm with an adaptive window: theory and experiment. In: Pattern analysis and machine intelligence (PAMI), IEEE transaction; IEEE Trans Pattern analysis and machine intelligence, 16(9):920-932
 4. Kang SB, Szeliski R, Chai J (2001) Handling occlusions in dense multi-view stereo. In: Computer vision and pattern

recognition (CVPR). IEEE Computer Society, Los Alamitos, Kauai, Hawaii, pp I:103-110

Plenoptic Function

Shing Chow Chan

Department of Electrical and Electronic Engineering,
The University of Hong Kong, Hong Kong, China

Synonyms

Image-based rendering (IBR)

Related Concepts

► Light Field; ► Lumigraph

Definition

The plenoptic function describes the intensity of each light ray in the world as a function of visual angle, wavelength, time, and viewing position.

Background

The term plenoptic was derived from the word roots plen- (*plenus*) and opti- (*optos*), which means full/complete and eye/view, respectively. The plenoptic function was coined by Bergen and Anderson [1] to

describe the intensity of each light ray in the world as a function of visual angle, wavelength, time, and viewing position. It captures everything that can potentially be seen by an optical device and is related to previous concept of J. J. Gibson's "the structure of ambient light" and Leonardo da Vinci's "visual pyramid."

A light ray in the space can be parameterized by a position with three dimensions and a direction or visual angle in two dimensions (Fig. 1). Therefore, together with the wavelength and time dimensions, the plenoptic function is a seven dimensional (7D) function. The study of early vision is thus closely related to the sampling and processing of the plenoptic function. For instance, the derivatives of the plenoptic function with the position and time give usual information regarding the motion of objects, etc. In computer vision and video processing, the wavelength domain is simplified by sampling in the (*R, G, B*) color system. Consequently, images and videos are just two-dimensional (2D) and three-dimensional (3D) special cases or samples of the 7D plenoptic function. Based on this function, theoretically, novel views at different positions and time can be reconstructed from its samples, provided that the sample rate is sufficiently high. Because of the multidimensional nature of the plenoptic function, various such simplifications called image-based representations have been proposed to render new views from the representations with different complexity/functionalities trade-offs. This is the foundation of image-based rendering (IBR) without using geometry, and it usually requires large amount of samples.

As the plenoptic function observed is a consequence of the interaction of the light sources with objects in the scene, which further involve their geometries and surface properties, there is considerable redundancy in the plenoptic function, which can be further exploited by estimating or measuring the geometry, lightings, and surface properties of the scene to different extent. Image-based representations employing such modeling approach and auxiliary information, called image-based modeling, generally will provide improved user interaction and require fewer samples for rendering. The capturing, sampling, rendering, and processing of the plenoptic function are important areas of research in IBR and related applications such as computational photography, 3D/multiview videos and displays, etc.

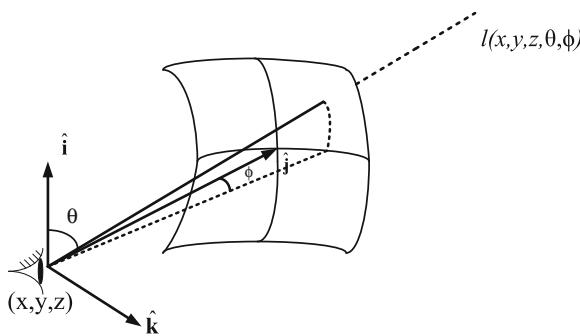
Theory

The amount or intensity of light along a ray is measured in radiance which is the power transmitted per unit area perpendicular to the direction of travel, per unit solid angle. Therefore, the plenoptic function that describes the radiance along light rays traveling in every direction through every point, denoted by l , is in watts (W) per meter squared (m^2) per steradian (sr) ($\text{W}/(\text{m}^2\text{sr})$).

The 7D plenoptic function can be parameterized using different coordinate systems, such as the familiar Cartesian, spherical, or cylindrical coordinates system. For instance, if the position V is parameterized by the Cartesian coordinates (x, y, z) while the direction is parameterized by the spherical coordinate (θ, ϕ) where θ and ϕ are the elevation and azimuth angles, respectively, as shown in (Fig. 1), the plenoptic function can be written as $l(x, y, z, \theta, \phi, \lambda, \tau)$ where λ and τ denote respectively the wavelength and time. By employing different parameterization and simplification, different image-based representations can be derived from the plenoptic function.

Representation

The conventional camera employs a lens to compress the light rays passing through an opening called the aperture and obtain a 2D image by placing an array of imaging sensors at the focal plane of the lens. For static scene, one can rotate the camera along the camera center and capture the rays at a given position V with different elevation and azimuth angles. The plenoptic function is then reduced to a panorama $l_V(\theta, \phi)$ with two dimensions. Since the sensor array is usually rectangular in shape, some rebinning or stitching of the images is necessary [5, 19]. Moreover, due to the finite field of view of cameras, the samples may be incomplete near the two poles of the sphere. This spherical set of rays can also be projected on a cube or a cylinder, which provides other convenient representation of panoramas. A panoramic video can be obtained by employing multiple closely spaced video cameras, instead of rotating a single camera, to obtain a 3D plenoptic function $l_V(\theta, \phi, \tau)$ for dynamic scenes. The close relationship between plenoptic function and image-based rendering was due to McMillan and Bishop [14] who proposed plenoptic modeling



Plenoptic Function, Fig. 1 Plenoptic function as a light energy received

using the 5D complete plenoptic function for static scene $l(x, y, z, \theta, \phi)$.

In the free space, the radiance along rays remains constant. This one-dimensional redundancy in the plenoptic function allows us to reduce it to a 4D function for static scene, which is called the light field [11] and lumigraph [8] in computer graphics. A similar concept in video communications is called the ray space [7]. The collection of light rays in a 4D static light field can be parameterized in a number of ways. A commonly used parameterization is the two-plane parameterization, where a light ray in the light field is parameterized as its intersections or coordinates with two parallel planes. These rays can be captured by taking a series of pictures on a 2D rectangular plane, which results in an array of images. Lumigraph differs from light field in that geometry in form of depth map is utilized to improve the rendering quality, which paves the way to more sophisticated representations in image-based modeling. For more information, see the section on light field and lumigraph. In [17], an outward facing camera moving on a circle was used to capture a series of densely sampled images, called concentric mosaic, of a static scene. This gives rise to a 3D representation, which can be used to render views inside the circle. A brief summary of these classical representations is given in Fig. 2. See the section on IBR for more illustration. These parameterizations can be further simplified by restricting the camera locations to line, line segments [4, 23], circle, circular arc, etc. to reduce hardware complexity of the capturing system. This gives rise to a wide range of image-based representations specified by different camera geometry [18]. For time-varying or dynamic scenes, similar parameterization can be employed. However, light rays

at the viewing locations have to be captured continuously. This can be done by an array of video cameras or specially design capturing devices.

The plenoptic function of static scene has also been extended to include the change of illumination direction as in the plenoptic illumination function [20] at the expenses of increased data samples to be recorded at different lighting directions. Since the effect of multiple light sources is additive, one can relight a panorama or light field with arbitrary lightings using such representations.

Plenoptic Cameras

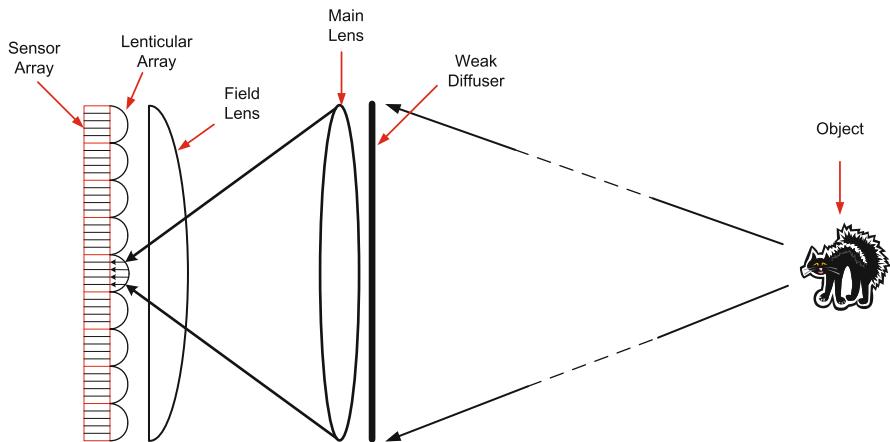
The term plenoptic camera was coined by Anderson and Wang [2]. It is a camera that captures a chunk of the optical structure of the light impinging on the lens. Basically, it records information about how the world appears from all possible viewpoints within the lens aperture. The plenoptic camera in [2] collects light with a single lens, but it uses a microlens or lenticular array at the image plane to redirect lights at certain directions at this location as collected from the lens aperture onto the sensor array (Fig. 3). Effectively, the plenoptic camera captures the cylindrical or spherical light fields over the lens aperture as macropixels onto a sensor array, depending respectively on whether a cylindrical or spherical lenticular array is used. The same principle has been used to produce autostereoscopic/multiview 3D display [10]. As a ray has to be mapped to an imaging pixel and there may be several directions to be recorded in a macropixel, the effective resolution of sensor array will be reduced. In [2], a (5×5) macropixel is employed. One can therefore expect such resolution reduction will be more severe for spherical than cylindrical lenticules.

Using the rays or light field recorded by the plenoptic camera, one can render novel views by arranging the light rays captured as in light field rendering. The images captured by a cylindrical lenticules is a 3D light field which support viewpoint changes along a horizontal line, whereas a spherical lenticules is able to capture a 4D light field as demonstrated and refined later by Ng et al. [16]. Moreover, as mentioned in [2], one can measure the parallax corresponding to these virtual displacements and derive depth estimates for objects in the scene. Since the virtual displacement, and hence the potential resolution of depth, is somewhat limited by the size of the lens aperture, it tends to be lower than using stereo cameras.

Plenoptic Function, Fig. 2

A taxonomy of plenoptic functions

| Dimension | Year | View space | Name |
|-----------|------|-----------------|--------------------------------|
| 7 | 1991 | Free | Plenoptic function |
| 5 | 1995 | Free | Plenoptic modeling |
| 4 | 1996 | Bounding box | Lightfield/Lumigraph |
| 3 | 1999 | Bounding circle | Concentric Mosaics |
| 2 | 1994 | Fixed point | Cylindrical/Spherical panorama |

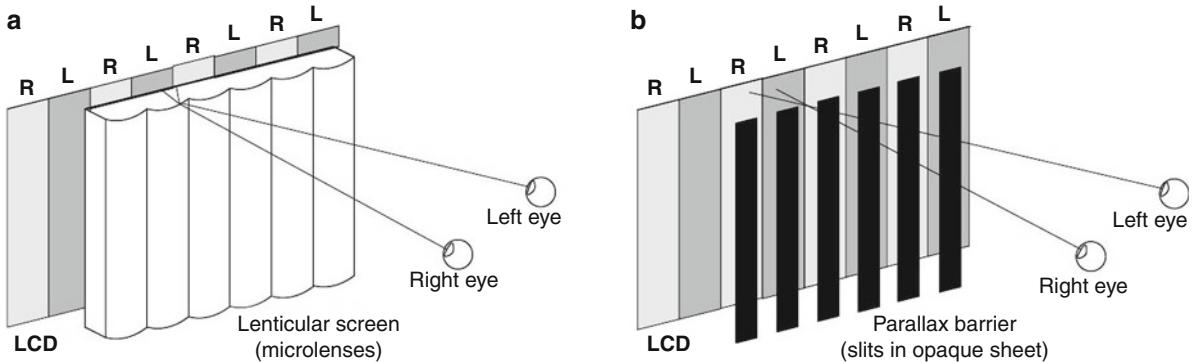


Plenoptic Function, Fig. 3 Plenoptic camera employing lenticular lens [2]

The concept of inserting optical elements or masks in the camera for 3D imaging dates back nearly a century ago, called integral photography or parallax panoramagrams and realized using a fly-eye lens array or a slit plate [9, 13]. Basically, these manipulate the 4D light field spectrum by modulation or reparameterization to make it fit into a 2D sensor array [2]. Another technique is to make use of coded aperture or shutter as an optical modulator to capture stereoscopic images [6] and more recently to preserve the high-frequency components of motion-blurred images and to provide high-dynamicrange. Another recent technique is to employ multiple capturing of the scene sequentially by using beam splitters and camera arrays where at each exposure, the image parameters such

as lighting, exposure time, focus, viewpoints, or spectral sensitivity are made different. After processing, a quality image or additional information is obtained. In [12], a coded aperture implemented using a programmable liquid crystal array, and multiple exposure was employed to capture light field using a single camera and derive the corresponding depth information.

Early systems for capturing light fields or plenoptic function for large environmental model usually involves camera arrays [4, 15, 21–23]. They are more expensive and difficult to build, as compared with a single plenoptic camera. On the other hand, due to their large baseline, it can also support a larger range of possible view points.



Plenoptic Function, Fig. 4 Multiview display employing (a) lenticular array and (b) parallax barrier

Sampling and Compression

During the capturing and display of plenoptic function, it is important to determine the number of samples required and how to obtain a satisfactory rendering quality. This problem was first studied for the light fields in [3] using the concept of Fourier transform. For Lambertian surfaces and rectangular sampling (see the section on light field for a more detailed treatment), it was found that the maximum camera spacing in the v plane is $\Delta t_{\max} = 1/(2\pi K_{\Omega_v} f(\frac{1}{z_{\min}} - \frac{1}{z_{\max}}))$ (assuming the notation of the lumigraph), where z_{\min} and z_{\max} denote respectively the minimum and maximum depth values of the scene and K_{Ω_v} is the maximum frequency of the light field in the v plane, which depends on the maximum frequency of texture variations B_v , the resolutions of the sampling camera $1/\Delta v$, and the rendering camera $1/\delta v$ as $K_{\Omega_v} = \min(B_v, 1/(2\Delta v), 1/(2\delta v))$. Similar results hold for the s plane. To reduce the sampling rate and avoid dense sampling, the value of K_{Ω_v} can be decreased by reducing B_v and $1/(2\Delta v)$, through prefiltering the light field images, to the desired rendering resolution. A more thorough discussion on the effect of geometry on this minimum sampling density or rate can be found in [18, 22]. It was found that the sampling rate can be reduced by decomposing the light fields into depth layers. Therefore, recent research has focused on the estimation of geometry of objects using stereo or multiview vision techniques, special depth sensing devices, and lighting techniques as in photometric stereo.

Since the plenoptic function is a high-dimensional and highly correlated signal, they need to be compressed for efficient storage and transmission.

Comprehensive reviews of the subjects are available at [18]. See also the sections on image-based rendering and light field.

Multiview Displays

Multiview three-dimensional displays consist of view-dependent pixels that reveal a different color according to the viewing angle and offer viewing of simple plenoptic function such as light fields with limited number of views without glasses. A view-dependent pixel can be implemented by multiplexing the image pixels of different views and rely on a lenticule arranged in an array or slits arranged in parallel (parallax barriers) to angularly separate (filter) them to create the given color at the desired viewing direction. This is illustrated in Fig. 4 for a display using the lenticular array and parallax barriers, respectively. To suppress the artifacts due to the periodic lenticules and slits in automultiscopic displays, they are slightly slanted so that pixels of a given view are made nonuniform rather than widely separated as periodic vertical lines. Prefiltering has to be applied to generate the nonuniform samples so as to avoid aliasing due to down-sampling since all the pixels from all views have to be multiplexed on the same display [10].

The bandwidth of such displays is also limited because an object with increasing depth becomes smaller, and smaller and hence it will eventually reach the minimum sampling interval and aliasing will occur. To display a light field on such multiview displays, one may also need to bandlimit the plenoptic function [24] as studied in plenoptic sampling [3].

Application

The plenoptic function serves an important concept for describing visual information in our world. Its sampling analysis also serves as a basic for designing plenoptic cameras and automultiscopic displays for capturing and displaying such high-dimensional function, respectively.

Open Problems

The efficient capturing and processing of plenoptic function has always been a problem in visual computing and vision research. Due to the multidimensional nature of the plenoptic function and its dependence on the scene geometry, the analysis is very difficult because the function itself may not even be bandlimited. Appropriate optical elements have to be used to reduce the aliasing due to sampling. This makes a general analysis very difficult, though some simple cases such as Lambertian surface, occlusion, and lighting on light field spectrum have been analyzed and found to be useful in practice. Recent advances in computational photography, microelectronics, and processing algorithms have accelerated the development of more sophisticated capturing devices and techniques to improve the rendering quality and reducing the hardware complexity. However, how to achieve high-quality renderings supporting a wide range of view points in large scale environmental modeling remains open.

References

1. Adelson EH, Bergen JR (1991) The plenoptic function and the elements of early vision. In: Landy M, Movshon JA (eds) Computational models of visual processing. MIT, Cambridge, MA
2. Adelson EH, Wang JYA (1992) Single lens stereo with plenoptic camera. *IEEE Trans Pattern Anal Mach Intell* 14(2):99–106
3. Chai JX, Tong X, Chan SC, Shum HY (2000) Plenoptic sampling. In: Proceedings of ACM SIGGRAPH, New Orleans, 307–318
4. Chan SC, Ng KT, Gan ZF, Chan KL, Shum HY (2005) The plenoptic videos. *IEEE Trans Circuits Syst Video Technol* 15(12):1650–1659
5. Chen SE (1995) QuickTime VR – an image-based approach to virtual environment navigation. In: Proceedings of ACM SIGGRAPH, Los Angeles, 29–38
6. Farid H, Simoncelli EP (1998) Range estimation by optical differentiation. *J Opt Soc Am A* 15(7):1777–1786
7. Fujii T, Kimoto T, Tanimoto M (1996) Ray space coding for 3D visual communication. In: Proceedings of Picture coding Symposium, Melbourne, 447–451
8. Gortler SJ, Grzeszczuk R, Szeliski R, Cohen MF (1996) The lumigraph. In: Proceedings of ACM SIGGRAPH, New Orleans, 43–54
9. Ives HE (1930) Parallax panoramagrams made with a large diameter lens. *J Opt Soc Am* 20(6):332–342
10. Konrad J, Halle M (2007) 3-D Displays and signal processing. *IEEE Signal Process Mag* 24(6):97–111
11. Levoy M, Hanrahan P (1996) Light field rendering. In: Proceedings of ACM SIGGRAPH, New Orleans, 31–42
12. Liang CK, Lin TH, Wong BY, Liu C, Chen HH (2008) Programmable aperture photography: multiplexed light field acquisition. *ACM Trans Graph* 27(3):55:1–55:10
13. Lippmann MG (1908) Epreuves reversibles donnant la sensation du relief. *J Phys* 7:821–825
14. McMillan L, Bishop G (1995) Plenoptic modeling: an image-based rendering system. In: Proceedings of ACM SIGGRAPH, Los Angeles, 39–46
15. Naemura T, Tago J, Harashima H (2002) Real-time video-based modeling and rendering of 3D scenes. *IEEE Comput Graph Appl* 22(2):66–73
16. Ng R, Levoy M, Bredif M, Duval G, Harowitz M, Hanrahan P (2005) Light field photography with a hand-held plenoptic camera. Stanford University Computer Science Tech Report CSTR 2005–02. <http://graphics.stanford.edu/papers/lfcamera/>
17. Shum HY, He LW (1999) Rendering with concentric mosaics. In: Proceedings of ACM SIGGRAPH, Los Angeles, 299–306
18. Shum HY, Chan SC, Kang SB (2007) Image-based rendering. Springer, New York
19. Szeliski R, Shum HY (1997) Creating full view panoramic image mosaics and environment maps. In: Proceedings of ACM SIGGRAPH, Los Angeles, 251–258
20. Wong T, Fu C, Heng P, Leung C (2002) The plenoptic illumination function. *IEEE Trans Multimed* 4(3):361–371
21. Wilburn B, Joshi N, Vaish V, Talvala E, Antunez E, Barth A, Adams A, Horowitz M, Levoy M (2005) High performance imaging using large camera arrays. *ACM Trans Graph* 24(3):765–776
22. Yang JC, Everett M, Buehler C, McMillan L (2002) A real-time distributed light field camera. In: Proceedings of Eurographics Workshop on Rendering, Pisa, 77–86
23. Zitnick CL, Kang SB, Uyttendaele M, Winder S, Szeliski R (2004) High-quality video view interpolation using a layered representation. In: Proceedings of ACM SIGGRAPH, Los Angeles, 600–608
24. Zwicker M, Vetro A, Yea S, Matusik W, Pfister H, Durand F (2007) Resampling, antialiasing, and compression in multi-view 3-D display. *IEEE Signal Process Mag* 24(6):88–96

Point Spread Function Estimation

► Blur Estimation

Polarization

Gary A. Atkinson

Machine Vision Laboratory, University of the West of England, Bristol, UK

Related Concepts

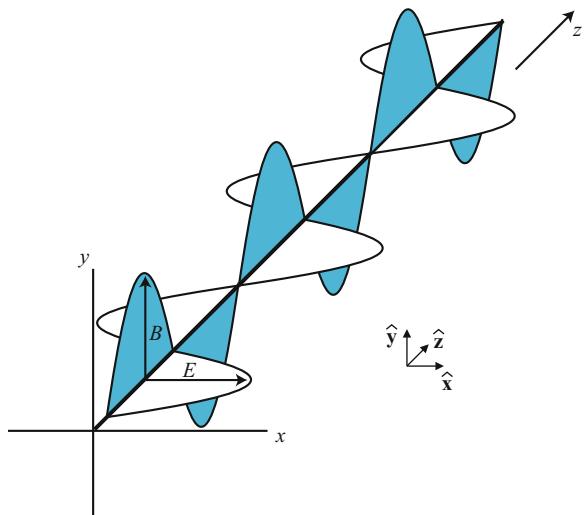
►Fresnel Equations; ►Polarized Light in Computer Vision; ►Polarizer

Definition

Polarization refers to the orientation distribution of the electromagnetic waves that constitute light rays. Light can assume a range of polarization states including unpolarized (uniformly mixed orientations), linearly polarized (fixed orientations), and circularly polarized (rotating field about the line of sight).

Background

The polarization of light is a property that relates to the directionality of the constituent electric and magnetic fields. In free space, light consists of sinusoidally oscillating electric and magnetic fields that are orthogonal to each other and to the direction of propagation. In the case of *perfectly linearly polarized light*, as shown in Fig. 1, the two types of field remain in a fixed plane through space. For *unpolarized light*, the orientations of these planes are randomized through temporal and spatial location. *Circular and elliptical polarizations* are also possible, whereby in a plane normal to the ray, the field is radial at each point with an orientation following a circular or elliptical temporal pattern. Finally, it is possible to have light consisting of combinations of polarization types. For example, *partially linearly polarized light* involves fields oscillating in all directions, but with one particular preferred direction. Polarization can be caused by a range of phenomena such as scattering, reflection, and transmission through media interfaces. In most computer vision and industrial applications, polarized light is generated and measured using polarizing filters.



Polarization, Fig. 1 A linearly polarized wave consisting of vertical magnetic fields (B) and horizontal electric fields (E)

Theory

Plane Wave Versus Photon Representation

Most texts on polarization regard light as an electromagnetic wave and explain the various phenomena via interactions of the constituent electric and magnetic fields with matter or other waves. However, in the realm of quantum optics, light consists of streams of particles known as photons. The details of this [1] are beyond the scope of this entry, although it is worthwhile noting a few basic points:

- Each photon has an associated energy that is proportional to its frequency.
- The angular momentum of each individual photon is quantized and can only take values of $-h/2\pi$ or $+h/2\pi$, where h is the *Planck Constant* [2].
- Circular polarization consists of identical photons (for monochromatic light) all with angular momentum vectors oriented either parallel to the direction of propagation (left- or \mathcal{L} -state) or antiparallel to the direction of propagation (right- or \mathcal{R} -state).
- Linear polarization (\mathcal{P} -state) can be described by a linear combination of \mathcal{L} - and \mathcal{R} -states.

For the remainder of this entry, the electromagnetic wave description of light will be employed.

Linear Polarization

In linearly polarized light, the electric and magnetic fields assume a fixed orientation through space, as

shown in Fig. 1. For the remainder of this entry, the focus will be on the electric field (similar arguments to those below can be applied to the magnetic component – see [3] for a thorough description). In the simplest case, the electric wave of linearly polarized light can be described using a standard equation of a plane wave:

$$\mathbf{E}(z, t) = \hat{\mathbf{x}} E_0 \cos(kz - \omega t) \quad (1)$$

where the coordinate axes (x, y, z) and unit vectors $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$ are defined in Fig. 1, E_0 is the amplitude of the wave, k is the wave number, ω is the angular frequency, and t is time. A detailed overview of the construction of the plane wave equation and its manipulation can be found in [4].

The wave described by (Eq. 1) is horizontally oriented. Of course, it is possible to replace the $\hat{\mathbf{x}}$ in (Eq. 1) with $\hat{\mathbf{y}}$, in which case, a vertical plane polarized wave will be represented. In general, an arbitrary case can be represented as a sum of orthogonal components:

$$\mathbf{E}(z, t) = \mathbf{E}_x(z, t) + \mathbf{E}_y(z, t) \quad (2)$$

where

$$\begin{aligned} \mathbf{E}_x(z, t) &= \hat{\mathbf{x}} E_{0x} \cos(kz - \omega t) \\ \mathbf{E}_y(z, t) &= \hat{\mathbf{y}} E_{0y} \cos(kz - \omega t + \phi) \end{aligned} \quad (3)$$

Here, E_{0x} and E_{0y} refer to the amplitude components in the x and y directions, respectively, and ϕ is the *phase difference* between the components (the absolute phases are irrelevant).

A wave is linearly polarized wherever $\phi = m\pi$, where m is an integer:

$$\begin{aligned} \mathbf{E}(z, t) &= (\hat{\mathbf{x}} E_{0x} + \hat{\mathbf{y}} E_{0y}) \cos(kz - \omega t) && \text{even } m \\ \mathbf{E}(z, t) &= (\hat{\mathbf{x}} E_{0x} - \hat{\mathbf{y}} E_{0y}) \cos(kz - \omega t) && \text{odd } m \end{aligned} \quad (4)$$

where $(\hat{\mathbf{x}} E_{0x} + \hat{\mathbf{y}} E_{0y})$ or $(\hat{\mathbf{x}} E_{0x} - \hat{\mathbf{y}} E_{0y})$ define both the amplitude and direction of the electric field (or *plane of polarization*) for each case.

Circular and Elliptical Polarization

The linear polarization case of (Eq. 4) is essentially a special case of the general electric field equation (Eq. 2). A second special case is where $\phi = \pi(m - 1/2)$, where m is, again, an integer. For the

case where $E_{0x} = E_{0y} = E_0$, (Eq. 3) can be substituted into (Eq. 2), to generate the following wave:

$$\begin{aligned} \mathbf{E}(z, t) &= E_0 (\hat{\mathbf{x}} \cos(kz - \omega t) + \hat{\mathbf{y}} \sin(kz - \omega t)) && \text{even } m \\ \mathbf{E}(z, t) &= E_0 (\hat{\mathbf{x}} \cos(kz - \omega t) - \hat{\mathbf{y}} \sin(kz - \omega t)) && \text{odd } m \end{aligned} \quad (5)$$

This case has two interesting properties. Firstly, in contrast to the linearly polarized case (Eq. 4), the scalar amplitude is fixed at E_0 . Secondly, the direction of the electric field is rotating at a fixed angular velocity in a clockwise (for even m) or anticlockwise (for odd m) direction relative to the direction of propagation. An alternative way to envisage this is that for a given point along the line of propagation of the wave, the tip of the electric field vector follows a circular motion about that point. The clockwise case is referred to in the optics literature as *right circularly polarized* light, while the anticlockwise case is called *left circularly polarized*. Note that a superposition of right and left circularly polarized light gives rise to a linearly polarized wave of amplitude $2E_0$. This is conducive to the photon angular momentum arguments discussed earlier.

For the general case, where $E_{0x} \neq E_{0y}$ and for nonintegral values of m , the tip of the electric field vector follows an elliptical path. This can be found by seeking a time- and position-independent solution to (Eq. 2) and (Eq. 3). The result [1] is that

$$\left(\frac{E_x}{E_{0x}} \right)^2 + \left(\frac{E_y}{E_{0y}} \right)^2 - 2 \left(\frac{E_x}{E_{0x}} \right) \left(\frac{E_y}{E_{0y}} \right) \cos \phi = \sin^2 \phi \quad (6)$$

This is the equation of an ellipse oriented at an angle, γ , given by

$$\tan 2\gamma = \frac{2E_{0x}E_{0y} \cos \phi}{E_{0x}^2 - E_{0y}^2} \quad (7)$$

For the simplest case, where $\gamma = 0$ the relationship reduces to

$$\frac{E_x^2}{E_{0x}^2} + \frac{E_y^2}{E_{0y}^2} = 1 \quad (8)$$

Elliptically polarized light is often referred to as the \mathcal{E} -state.

Partial Linear Polarization

Often in nature and technology, light assumes a partially polarized form. The most common of these is

partially linearly polarized light. This consists of a superposition of linearly polarized light and unpolarized light. It is often desirable to define a quantity known as the *degree of polarization*, ρ , that relates the intensity (or flux density) of the polarized component, I_p , to that of the unpolarized component, I_u :

$$\rho = \frac{I_p}{I_p + I_u} \quad (9)$$

This is often expressed as a percentage, such that light of equal components is 50% polarized. The degree of polarization can be analyzed using a polarizing filter (often referred to as an “analyzer” in optics texts). If the maximum and minimum transmitted light intensities as the polarizer is rotated are I_{\max} and I_{\min} , then $I_{\min} = I_u/2$ and $I_{\max} = I_u/2 + I_p$ so that

$$\rho = \frac{I_{\max} - I_{\min}}{I_{\max} + I_{\min}} \quad (10)$$

Alternative Representation

Stokes Vectors

In many practical applications of polarized light, the physical wave description above is of little use due to the minuscule quantities involved. The Stokes vectors aim to represent the polarization state of light in a compact form that most detectors can measure (See [5] for derivations and further details).

The general Stokes vector is defined by

$$S = [S_0, S_1, S_2, S_3]^T \quad (11)$$

The first of the parameters of the Stokes vector, S_0 , is simply the intensity of the light. The second parameter, S_1 , quantifies the tendency for vertical or horizontal polarization and is related to the difference in the corresponding vertical and horizontal components of the wave’s electric field. Where $S_1 > 0$, the light resembles a horizontal \mathcal{P} -state, while $S_1 < 0$ relates to vertical \mathcal{P} -states. For the case where $S_1 = 0$, there is no tendency either way as in circular polarized light, elliptical light at 45° and unpolarized light. S_2 is interpreted in a similar fashion, but relates to angles of 45° and -45°. Finally, S_3 relates to the handedness of the polarization ($S_3 > 0$ is right-handed, $S_3 < 0$ is left-handed and $S_3 = 0$ is neither).

Often, the Stokes vectors are normalized by S_0 so that, for example, unpolarized light has

$S = [1, 0, 0, 0]^T$; vertically linearly polarized light has $S = [1, -1, 0, 0]^T$; linearly polarized light at 45° has $S = [1, 0, 1, 0]^T$; and right-circularly polarized light has $S = [1, 0, 0, 1]^T$. Furthermore, light may consist of superpositions of different states, in which case the individual vectors are simply added together.

The Stokes vectors provide useful representations of polarization for *incoherent light*, which is abundant in nature. Computer vision and other fields may utilize the Stokes representation of the degree of polarization. In the case of partially linearly polarized light, for example, the degree of polarization is

$$\rho = \frac{\sqrt{S_1^2 + S_2^2}}{S_0} \quad (12)$$

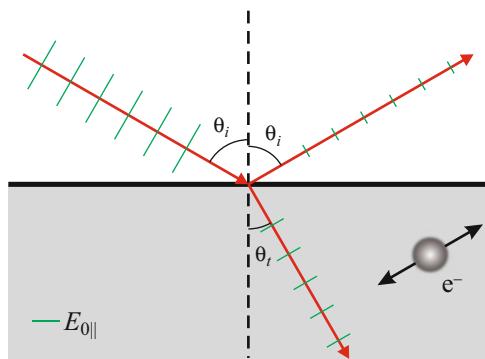
Jones Vectors

For *coherent light*, typically from lasers, an alternate representation is also possible. The Jones vector of a polarized light wave is simply given by

$$\mathbf{E} = \begin{bmatrix} E_x(t) \\ E_y(t) \end{bmatrix} \quad (13)$$

The electric fields in the Jones vectors are generally represented as a wave via the imaginary number approach [4] ($E = E_0 e^{i(kz - \omega t + \phi)}$). They are then normalized by the intensity in a similar fashion to the Stokes vectors. The results [1] are that horizontal and vertical \mathcal{P} -states are given by $[1, 0]^T$ and $[0, 1]^T$, respectively; \mathcal{P} -states at 45° are given by $(1/\sqrt{2})[1, 1]^T$ and $(1/\sqrt{2})[1, -1]^T$; and \mathcal{R} - and \mathcal{L} -states are given by $(1/\sqrt{2})[1, -i]^T$ and $(1/\sqrt{2})[1, i]^T$.

The advantage of the Jones representation (aside from its compactness) is that certain optical operations can be described sequentially in terms of matrix operations. For example, light passing through a linear vertical polarizer has its Jones vector modified by the a *Jones matrix* of $\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$. A related set of matrices for operating on Stokes vectors has also been derived. These are called *Mueller matrices* and have dimensions of 4×4 . Further details of Jones and Mueller matrices can be found in [1] and references therein.



Polarization, Fig. 2 Polarization from specular reflection

Polarized Light in Nature

Sunlight, which constitutes the overwhelming majority of natural light on Earth, is unpolarized. Despite this, linearly polarized light is abundant in nature as a result of various phenomena that convert unpolarized light to polarized light. Circularly polarized light by contrast is rare in nature and only occurs under certain ideal conditions. The two most common causes of polarization are reflection from surfaces and light scattering in the atmosphere and these are discussed below. Other causes [6] include internal reflection, rainbows, clouds, and birefringence (in materials where optical properties are anisotropic).

Reflection

Consider a light wave impinging upon a smooth dielectric surface, as shown in Fig. 2. A fraction of the light penetrates the surface and the remainder is reflected. In real surfaces, the penetrated light undergoes a series of complicated scattering interactions according to the radiative transfer equation [7]. However, the problem can be simplified by assuming that no scattering occurs apart from at the interface. In this case, the transmitted wave is refracted according to the well-known Snell's Law [1]:

$$n_i \sin \theta_i = n_t \sin \theta_t \quad (14)$$

where n_i and n_t refer to the refractive indices of air (≈ 1) and the reflecting material and θ_i and θ_t refer to the incident and transmitted angles respectively.

Consider the component of the electric field of the wave that is parallel to the plane of incidence. This component results in electron dipole oscillations in the medium at angle θ_t (see Fig. 2). These dipoles close to the interface emit the reflected wave. However, this

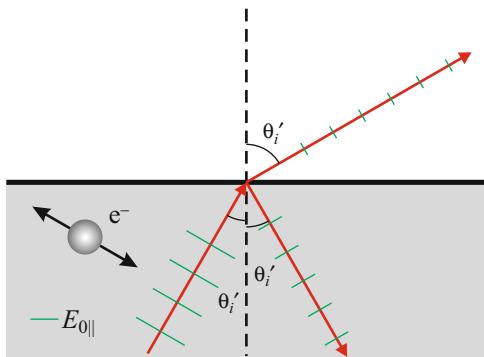
reflected wave is at a different angle to θ_t meaning that the parallel component of the reflected wave is attenuated due to foreshortening. For the component of the electric field that is perpendicular to the plane of incidence, the foreshortening effect is not present. This difference in wave attenuation caused by foreshortening gives rise to a partially linearly polarized reflection.

Note that, for the case where $\theta_i + \theta_r = 90^\circ$, foreshortening eliminates the parallel component of the reflection entirely. The angle at which this occurs is known as the *Brewster Angle*, θ_B , and it follows from (Eq. 14) that, for the case where $n_i = 1$,

$$\theta_B = \arctan n_t \quad (15)$$

Much of the light present in everyday life is not specularly reflected from surfaces, as discussed above, but diffusely reflected. That is, the light is scattered either at the surface or below the surface and both of these phenomena affect the polarization state of the reflected light. For the latter case, several efforts have been made to model the precise means by which light is scattered to form a reflection [8]. The important point, so far as polarization is concerned, is that after undergoing internal scattering, the light impinges upon the air-medium interface from within, as shown in Fig. 3. This light then undergoes a refraction as it passes the interface. A similar argument to that above for specular reflection can then be applied to establish that a foreshortening effect polarizes the diffusely reflected light also. Note however, that as the incident wave can never be perpendicular to the reflected wave, there is no angle by which the diffusely reflected light becomes completely polarized. The degree and angles of polarization can be determined from the Fresnel equations and are discussed in detail in the entry ► [Polarized Light in Computer Vision](#). Note also that polarization by refraction occurs as light from the Sun enters water, causing light near the surface of lakes, seas, and oceans to be partially polarized.

For the case of surface scattering, several attempts to model the microscopic roughness patterns have been made, such as [9] and [10]. For both specular and diffuse reflection, the random nature of microscopic surface corrugations has the effect of mixing the orientations of reflected light fields. One of the effects



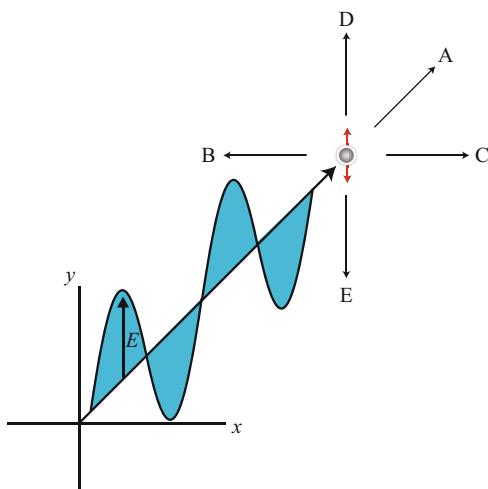
Polarization, Fig. 3 Polarization from diffuse reflection (refraction)

of this is to depolarize the reflection. As predicted by the *Umov Effect* [11], rough surfaces with lower albedo retain higher degrees of polarization than light surfaces. This is due to the fact that fewer inter-reflections take place on dark surfaces before the light waves are completely absorbed by the medium. This effect was first noted in astronomy, where the degree of polarization of an astronomical body is inversely proportional to its albedo.

Scattering

As unpolarized light is scattered by small particles (for e.g., a molecule in the atmosphere or a dust particle) it becomes partially linearly polarized. The effect can be explained in a similar fashion to that with surface reflection. Consider a light wave impinging on a particle. The particle will scatter the light in a range of directions. It turns out that, when viewed at an angle of 90° to the direction of the light wave, the scattered light is completely polarized, while the waves that are scattered close to the direction of propagation or scattered by $\approx 180^\circ$ are only slightly polarized. This phenomenon is observed on a grand scale in the Earth's atmosphere. When a clear sky is viewed at an angle of 90° from the Sun, it can be observed to be $\approx 75\%$ polarized, compared to almost no polarization at $\approx 0^\circ$ and approaching 180° . In practice, light scattered at 90° is not completely polarized due to a range of factors such as molecular anisotropies and multiple-scattering depolarization.

The phenomenon of polarization by scattering can be explained as follows. The wave shown in Fig. 4 induces an oscillating dipole in the scattering particle



Polarization, Fig. 4 Polarization from scattering

as a result of the redistribution of the electron cloud and nucleus location caused by the electric field. The dipole, in turn, generates the scattered wave. Assume that the impinging wave is linearly vertically polarized, as in Fig. 4. The new wave has maximum amplitude at an angle perpendicular to the dipole orientation and zero amplitude when parallel to the dipole due to foreshortening. In relation to Fig. 4, amplitude maxima would be found in directions *A*, *B*, and *C* (in addition to backscatter), while no scattering occurs in directions *D* or *E*. Next consider an impinging wave that is polarized horizontally. In this case, directions *A*, *D*, and *E* are the maxima, while directions *B* and *C* have no scattering. Finally, by representing an unpolarized incident wave as a linear combination of randomized linearly polarized states, it can be seen that direction *A* and backscatter remain unpolarized, directions *B* and *C* retain only the vertical polarization and directions *D* and *E* retain only the horizontal polarization. Directions intermediate between those discussed will, of course, become partially linearly polarized. For such angles with the common case of scattering by atmospheric dust or smoke, the degree of polarization is given by [3]

$$\rho = \frac{\sin^2 \beta}{1 + \cos^2 \beta} \quad (16)$$

where β is the scattering angle. This is known as Rayleigh scattering. However, complete polarization is

not typically observed due to the reasons mentioned above.

Application

Many creatures, including some insects and marine animals, have eyes that are sensitive to the polarization state of light [12]. Bees and ants, for example, use the polarization pattern in the sky or water to aid navigation [6]. Several marine creatures use similar patterns found underwater for the same purpose [13]. There is evidence [14] that certain aquatic creatures use polarization to isolate objects of interest within their field of view. It is also believed that some species communicate by reflecting light only of a certain polarization angle.

In everyday life, a common application of polarization is in sunglasses and camera filters. A pair of polarizing sunglasses with vertical transmission axis blocks out large amounts of specular glare from surfaces as Fig. 2 suggests. A correctly oriented camera lens filter can also be used to diminish glare. In addition, such a filter can increase contrast between clouds and the sky by cutting out the polarized component of sky and having less impact on light from clouds.

Liquid crystal displays (LCDs) are based on crystals placed between crossed-polarizers. Light is transmitted or absorbed in different segments, depending on which areas are subjected to a voltage. Where the voltage is applied, the crystals rotate the angle of polarization by 90°. Stress patterns of many materials can be analyzed by placing samples between crossed polarizers and measuring the transmitted intensity patterns. This is possible due to the changing birefringence properties of materials under strain. In entertainment, 3D images can be generated by time-multiplexing different polarization states onto a screen, with each state representing a different viewpoint of the object being displayed. The viewers wear polarizing spectacles that transmit each state to separate eyes. Finally, polarization is prevalent in astronomy. For example, transmission and absorption effects in strongly magnetic regions of the Sun's surface cause circular polarization, while interstellar dust is responsible for polarization by scattering over long distances.

Within the field of computer vision, polarization has been used for a range of tasks, including the

following (further details and references can be found in the separate entry on ►Polarized Light in Computer Vision):

- Specularity reduction
- Shape recovery
- Reflectance analysis
- Image enhancement
- Reduction of inter-reflections
- Separation of reflectance components
- Image segmentation

References

1. Hecht E (1998) Optics, 3rd edn. Addison Wesley, Reading
2. Tipler PA, Mosca G (2007) Physics for scientists and engineers, 6th edn. Freeman, New York
3. Born M, Wolf E (1999) Principles of optics. Electromagnetic theory of propagation, interference and diffraction of light, 7th edn. Pergamon, London
4. Main IG (1993) Vibrations and waves in physics, 3rd edn. Cambridge University Press, Cambridge
5. Shurcliff WA (1962) Polarized light: production and use. Harvard University Press, Cambridge
6. Können GP (1985) Polarized light in nature. Cambridge University Press, Cambridge
7. Chandrasekhar S (1960) Radiative transfer. Oxford University Press, New York
8. Jensen HW, Marschner SR, Levoy M, Hanrahan P (2001) A practical model for subsurface light transport. In: Proceedings of SIGGRAPH, Los Angeles, pp 511–519
9. Torrance K, Sparrow M (1967) Theory for off-specular reflection from roughened surfaces. *J Opt Soc Am* 57: 1105–1114
10. Beckmann P, Spizzichino A (1963) The scattering of electromagnetic waves from rough surfaces. Pergamon, New York
11. Umov N (1905) Chromatische depolarisation durch lichtzerstreuung. *Phys Z* 6:674–676
12. Shashar N, Cronin TW, Wolff LB, Condon MA (1998) The polarization of light in a tropical rain forest. *Biotropica* 30:275–285
13. Horváth G, Varjú D (1995) Underwater polarization patterns of skylight perceived by aquatic animals through Snell's window on the flat water surface. *Vision Res* 35: 1651–1666
14. Cronin TW, Shashar N, Caldwell RL, Marshall J, Cheroske AG, Chiou TH (2003) Polarization and its role in biological signalling. *Integr Comp Biol* 43:549–558

Polarization Filter

►Polarizer

Polarized Light in Computer Vision

Gary A. Atkinson

Machine Vision Laboratory, University of the West of England, Bristol, UK

Related Concepts

- ▶ [Fresnel Equations](#); ▶ [Polarization](#); ▶ [Polarizer](#);
- ▶ [Transparent Layers](#)

Definition

Polarization refers to the orientation distribution of the electromagnetic waves that constitute light rays. The phenomenon of light polarization has been exploited in computer vision for a range of applications including surface reconstruction, specular/diffuse separation, and image enhancement.

Background

Light consists of orthogonal electric and magnetic fields. Most natural light is unpolarized and so consists of randomly fluctuating field directions. However, a range of natural phenomena (e.g., scattering and reflection) and human inventions (e.g., polarizing filters and liquid crystal displays) cause the light to become polarized. That is, the electric and magnetic fields become confined to specific planes or get constrained in other ways. In the field of computer vision, both natural and artificially generated polarized light has been utilized for a range of applications including specularity reduction, shape recovery, reflectance analysis, image enhancement, segmentation, and separation of reflectance components.

Theory

The majority of research into polarization methods for computer vision rely on passively analyzing the polarization state of incoming light using either uncontrolled, or at least unpolarized, illumination. However,

one of the most common uses of polarization in computer vision is to remove specularities from images using a more active approach. This entry first describes this method and is followed by descriptions of techniques for polarization measurement. It then covers the relevant theory of the two most common natural causes of polarization (reflection and scattering) and their exploitation in computer vision.

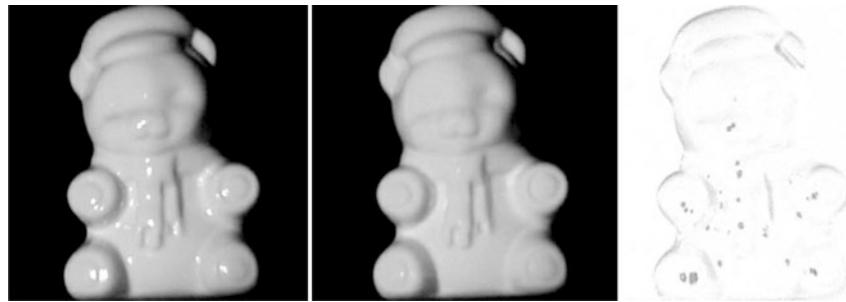
Polarization and Specularities

The images of the porcelain bear model in Fig. 1 were obtained using a camera with one linear polarizer (often referred to as an “analyzer” in optics texts) mounted on the lens and another in front of the only light source. For the first image, the two polarizers were oriented parallel to each other, while for the second image they were at 90°. The second image clearly shows a minimization of specularity. This is of great benefit to a range of computer vision methods that assume specularities are absent from images such as shape-from-shading [1] and photometric stereo [2].

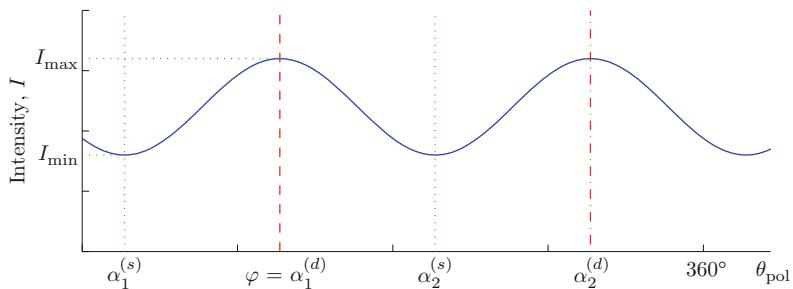
The theory behind the specularity reduction is that the polarized incoming light induces electron oscillations in the reflecting medium in only one direction (see the entry on ▶ [Polarization](#)). This means that the *specularly* reflected light, which is generated by these oscillations, has an electric field that is constrained to a single plane also. If the lens polarizer is oriented at right angles to this plane then all of this light is blocked from the camera. On the other hand, the *diffusely* reflected light is depolarized by subsurface scattering and surface roughness. This means that much of the diffusely reflected light is transmitted through the lens polarizer.

Polarization Imaging

For the cases where the incident light is unpolarized, the simplest and most common method of measuring polarization is to take a sequence of images with a rotating linear polarizer. Consider a camera’s field of view where the incoming light is partially linearly polarized with varying degrees between 0% and 100% (see the entry on ▶ [Polarization](#) for the definition of the degree of polarization). The aim is to acquire a *polarization image*: a three-channel image where each pixel has components corresponding to (1) the grayscale intensity, (2) the degree of polarization, and (3) the phase angle (not to be confused with the phase of the

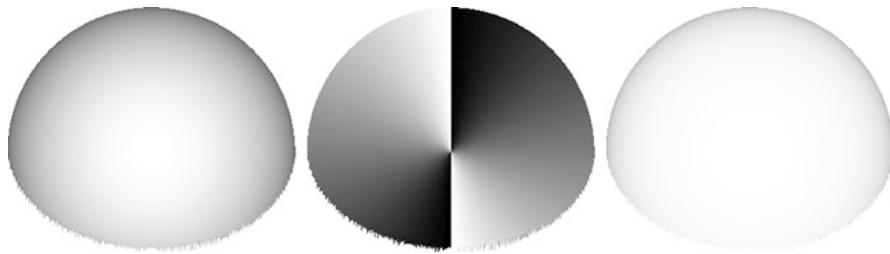


Polarized Light in Computer Vision, Fig. 1 *Left:* Uncrossed polarizers, *center:* crossed polarizers, *right:* difference image [3]. N.B. The difference image has been inverted and undergone nonlinear intensity scaling to aid clarity



Polarized Light in Computer Vision, Fig. 2 Transmitted radiance sinusoid. $\alpha_1^{(s)}$ and $\alpha_2^{(s)}$ are the two possible surface azimuth angles for a given phase angle φ , assuming that the reflection is

specular. For diffuse reflection, the possible azimuth angles are $\alpha_1^{(d)}$ and $\alpha_2^{(d)}$



Polarized Light in Computer Vision, Fig. 3 Synthetic rendering of a polarization image of a hemisphere. *Left:* intensity, *center:* phase angle, *right:* degree of polarization

electric and magnetic fields of the wave that constitute the light ray) of the linearly polarized component of the incoming light.

As the polarizer on the lens rotates, the transmitted intensity at each point is given by the *transmitted radiance sinusoid*:

$$I(\theta_{\text{pol}}, \varphi) = \frac{I_{\max} + I_{\min}}{2} + \frac{I_{\max} - I_{\min}}{2} \cos(2\theta_{\text{pol}} - 2\varphi) \quad (1)$$

where θ_{pol} is the polarizer orientation, I_{\min} and I_{\max} are the minimum and maximum observed pixel brightnesses and φ is the phase angle. The transmitted radiance sinusoid is shown graphically in Fig. 2.

The complete polarization image is then constructed by fitting the transmitted radiance sinusoid to three or more different intensity images for different polarizer angles. This assumes that the pixel brightness is proportional to the light impinging on the sensor – i.e., the camera response function is linear. For typical

imaging equipment, the intensity component is in the range [0, 255], the degree of polarization is always in the range [0, 1], and the phase is always in the range [0, 180°]. Fig. 3 shows a synthetic example of the three components of a polarization image.

A major weakness of many polarization methods is the amount of time needed to acquire the data (the actual processing of the data is often highly efficient). For the rotating polarizer method described above, three or more images are required with the polarizer at different orientations. This limits applications to static scenes. Matters were improved after the development of *polarization cameras* [4] that used liquid crystals to rapidly switch the axis of the polarizing filter. Subsequently, PLZT (polarized lead zirconium titanate) cameras [5] were designed with the aim of recovering all four components of the Stokes vectors (defined in the entry on ►Polarization) [6].

Other methods of rapid polarization image capture include placing multiple cameras with different polarizers but near-parallel optical axes [7]; using a detector array-like Bayer pattern to capture the polarization field on chip [8]; using a lightfield camera, where the aperture is divided into segments, some of which contain polarizers [9]; and employing multiple CCDs to capture all the polarization components using beam splitters [10]. The interested reader is also referred to [11] for a detailed survey.

Polarization and Reflection in Images

Perhaps the most common exploitation of polarization in computer vision is to measure the polarization state of light reflected from dielectric surfaces [12]. As described in the entry on ►Polarization, natural light undergoes partial polarization when it is reflected from surfaces. Consider a specular reflection being viewed through a rotating polarizer. Because the electric field of the reflected wave is most attenuated in the plane of reflection, the greatest transmission through the polarizer occurs when it is oriented at right angles to the plane. Minimum transmission occurs when the polarizer is parallel to the plane. Note however, that the polarizer is unable to distinguish two planes of reflection oriented 180° apart. If α is the azimuth angle of the surface (the angle of the projection of the surface normal onto the image plane), then there are two possible values for a given measurement of φ :

$$\alpha_1^{(s)} = \varphi - 90^\circ \quad \alpha_2^{(s)} = \varphi + 90^\circ \quad (2)$$

where the superscript s indicates that the reflection is specular.

As explained in [12], the Fresnel equations can be used to represent the maximum and minimum intensities passing through a rotating polarizer as

$$I_{\min} = \frac{R_{\perp}(n, \theta)}{R_{\perp}(n, \theta) + R_{\parallel}(n, \theta)} I_r \quad (3)$$

$$I_{\max} = \frac{R_{\parallel}(n, \theta)}{R_{\perp}(n, \theta) + R_{\parallel}(n, \theta)} I_r$$

Here R_{\parallel} and R_{\perp} are the parallel and perpendicular Fresnel intensity reflectivity components, respectively (see the entry on ►Fresnel Equations), n is the refractive index, θ is the angle of incidence, and I_r is the magnitude of the specularity. In an imaging situation, θ is equivalent to the zenith angle of the surface (the angle between the surface normal and the line of sight of the camera). Rewriting in terms of the degree of polarization and rearranging gives [12]

$$\rho_s(n, \theta) = \frac{2 \sin^2 \theta \cos \theta \sqrt{n^2 - \sin^2 \theta}}{n^2 - \sin^2 \theta - n^2 \sin^2 \theta + 2 \sin^4 \theta} \quad (4)$$

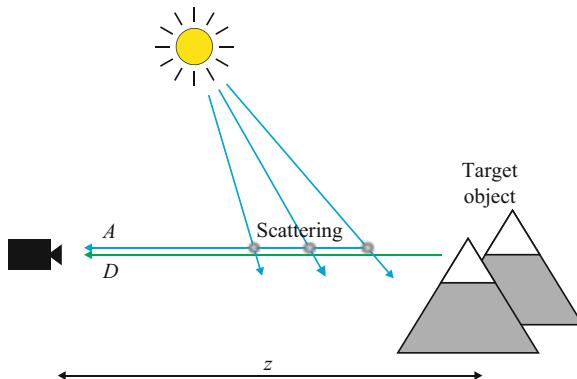
where the suffix s is appended to the degree of polarization as this assumes a purely specular reflection.

Equations 2 and Eq. 4 allow one to impose constraints on the angle of a reflecting surface for shape recovery applications using measurements taken of φ and ρ . The azimuth angle of the surface has two possibilities as shown by (Eq. 2) and Fig. 2. Apart from at the Brewster angle, where $\rho = 1$, there are also two solutions to (Eq. 4), meaning two zenith angles are also possible.

Combining the above theory with the subsurface scattering model of reflection [12, 13], the following relations for *diffuse* reflection can be derived:

$$\alpha_1^{(d)} = \varphi \quad \alpha_2^{(d)} = \varphi + 180^\circ \quad (5)$$

$$\rho_d = \frac{(n - 1/n)^2 \sin^2 \theta}{2 + 2n^2 - (n + 1/n)^2 \sin^2 \theta + 4 \cos \theta \sqrt{n^2 - \sin^2 \theta}} \quad (6)$$



Polarized Light in Computer Vision, Fig. 4 Haze in an image due to atmospheric scattering

On this occasion, there is only one solution for the zenith angle, but the degree of polarization is lower and therefore more difficult to measure. Indeed, the information contained in this degree of polarization is only useful for large zenith angles [14].

Outdoor Polarization Imaging

One of the most common causes of polarization in nature is atmospheric scattering [15]. Consider the imaging process of a distant outdoor object or scene. Light that arrives at a detector consists of a component transmitted from the object itself, D , and a component of “airlight,” A . The former of these is typically reflected sunlight while the latter is light scattered from the atmosphere, as shown in Fig. 4. The total intensity at the camera can be represented by

$$I_{\text{tot}} = A + D \quad (7)$$

It can be shown [16] that

$$A = A_{\infty} (1 - t(z)) \quad (8)$$

$$D = L_{\text{obj}} t(z) \quad (9)$$

where A_{∞} is the airlight radiance for an object at infinity, L_{obj} is the object radiance, z is the distance to the object and $t(z)$ is given by

$$t(z) = \exp \left(- \int_0^z \beta(z') dz' \right) \quad (10)$$

where β is called the *coefficient of extinction*.

As explained in the ►Polarization entry, atmospheric scattering causes polarization. The degree of polarization for the airlight component is given by

$$\rho = \frac{A_{\perp} - A_{\parallel}}{A_{\perp} + A_{\parallel}} \quad (11)$$

where A_{\perp} and A_{\parallel} are the scattered components parallel to and perpendicular to the plane of scattering, respectively. If it is assumed that the target object is emitting/reflecting unpolarized light, then the light measured through a linear polarizer is a transmitted radiance sinusoid, as in Fig. 2, but with an intensity offset equal to the light directly transmitted from the target object. Clearly, the simple task of rotating a lens-mounted polarizer to the angle that allows minimum transmission removes some of the airlight and so enhances the image of the target object. Taking images at more than one polarizer orientation allows to solve for the unknowns in (Eq. 7)–(Eq. 11) and dehaze an image more effectively [16]. A related polarization analysis can also be used for images taken in murky water [17].

Application

The above theory, and extensions thereof, has been used in a range of areas within computer vision. Perhaps the most studied is in shape recovery. This is due to the ease by which strong constraints can be placed on the surface normals from single viewpoints. Polarization therefore offers huge benefits to shape-from-shading and related methods. As explained above, the normals are not fully constrained using these techniques so additional methods are required to complete the reconstruction [14, 18]. It is also possible to model surfaces that are usually inaccessible to computer vision such as transparent surfaces [18]. As already mentioned, polarization from the atmosphere has been applied to image enhancement outdoors [16] and underwater [17].

The difference in polarizing properties between metallic and dielectric reflection [15] has been used for segmentation [12], while the difference between specular and diffuse properties has allowed the two reflection components to be separated [19]. Related work has allowed improved laser-based range finding, by minimizing the effects of inter-reflections [20]. Other work

has exploited polarization by specular reflection for separating transmitted scenes from reflected scenes in a glass window [21, 22]. Polarization has also found a range of other applications including reflectance analysis [23] and cosmetics [24].

References

1. Horn BKP, Brooks MJ (1989) Shape from shading. MIT, Cambridge
2. Woodham RJ (1980) Photometric method for determining surface orientation from multiple images. Opt Eng 19: 139–144
3. Ragheb H, Hancock ER (2002) Highlight removal using shape-from-shading. In: Proceedings of European conference on computer vision (ECCV). Springer, Berlin/New York, pp 626–641
4. Wolff LB (1997) Polarization vision: a new sensory approach to image understanding. Image Vis Comput 15:81–93
5. Shames PE, Sun PC, Fainman Y (1998) Modelling of scattering and depolarizing electro-optic devices. I. characterization of lanthanum-modified lead zirconate titanate. Appl Opt 37:3717–3725
6. Miyazaki D, Takashima N, Yoshida A, Harashima E, Ikeuchi K (2005) Polarization-based shape estimation of transparent objects by using raytracing and PLZT camera. Proc SPIE 5888:1–14
7. Hooper BA, Baxter B, Piotrowski C, Williams JZ, Dugan J (2009) An airborne imaging multispectral polarimeter. In: Proceedings of IEEE/MTS Oceans, Biloxi, Mississippi, USA
8. Gruel V, der Spiegel JV, Engheta N (2009) Advances in integrated polarization image sensors. In: Proceedings of LiSSA Workshop, Bethesda, Maryland, USA, pp 62–65
9. Horstmeyer R, Euliss G, Athale R, Levoy M (2009) Flexible multimodal camera using a light field architecture. In: Proceedings of computational photography (ICCP). IEEE, Piscataway, pp 1–8
10. Zappa CJ, Banner ML, Schultz H, Corrada-Emmanuel A, Wolff LB, Yalcin J (2008) Retrieval of short ocean wave slope using polarimetric imaging. Meas Sci Technol 19:055503
11. Tyo JS, Goldstein DL, Chenault DB, Shaw JA (2006) Review of passive imaging polarimetry for remote sensing applications. Appl Opt 45:5453–5469
12. Wolff LB, Boult TE (1991) Constraining object features using a polarisation reflectance model. IEEE Trans Pattern Anal Mach Intell 13:635–657
13. Wolff LB (1994) Diffuse-reflectance model for smooth dielectric surfaces. J Opt Soc Am A 11:2956–2968
14. Atkinson GA, Hancock ER (2007) Shape estimation using polarization and shading from two views. IEEE Trans Pattern Anal Mach Intell 29:2001–2017
15. Können GP (1985) Polarized light in nature. Cambridge University Press, Cambridge/New York
16. Schechner YY, Narashimhan SG, Nayar SK (2003) Polarization-based vision through haze. Appl Opt 42: 511–525
17. Schechner YY, Karpel N (2005) Recovery of underwater visibility and structure by polarization analysis. IEEE J Ocean Eng 30:570–587
18. Miyazaki D, Kagesawa M, Ikeuchi K (2004) Transparent surface modelling from a pair of polarization images. IEEE Trans Pattern Anal Mach Intell 26:73–82
19. Umeyama S, Godin G (2004) Separation of diffuse and specular components of surface reflection by use of polarization and statistical analysis of images. IEEE Trans Pattern Anal Mach Intell 26:639–647
20. Wallace AM, Liang B, Clark J, Trucco E (1999) Improving depth image acquisition using polarized light. Int J Comput Vis 32:87–109
21. Schechner YY, Shamir J, Kiryati N (2000) Polarization and statistical analysis of scenes containing a semireflector. J Opt Soc Am 17:276–284
22. Farid H, Adelson EH (1999) Separating reflections from images using independent components analysis. J Opt Soc Am 16:2136–2145
23. Atkinson GA, Hancock ER (2008) Two-dimensional BRDF estimation from polarisation. Comput Vis Image Underst 111:126–141
24. Lefaudoux N, Lechocinski N, Clemenceau P, Breugnot S (2009) New luster formula for the characterization of hair tresses using polarization imaging. J Cosmet Sci 60(2):153–169

Polarizer

Daisuke Miyazaki

Graduate School of Information Sciences, Hiroshima City University, Asaminami-ku, Hiroshima, Japan

Synonyms

[Polarization filter](#); [Polarizing film](#)

Related Concepts

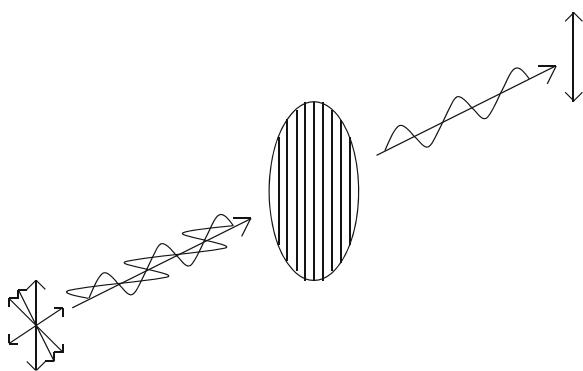
► [Polarization](#)

Definition

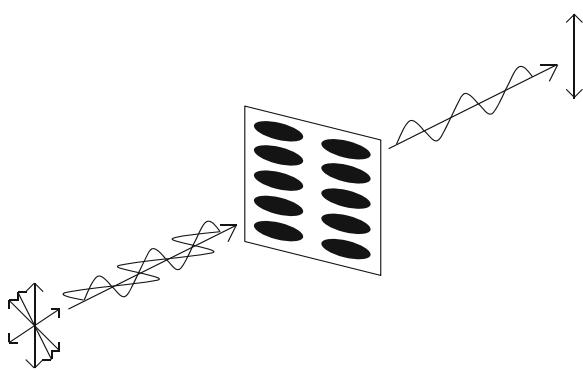
The device which polarizes the light when the light goes through it.

Background

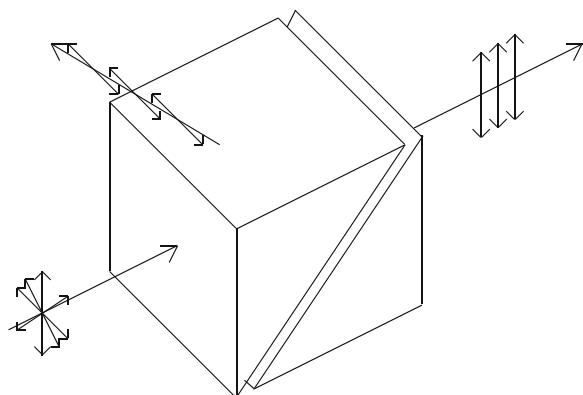
Polarizers makes the light oscillate in one direction. The polarization state of light changes if it hits any



Polarizer, Fig. 1 Illustration of linear polarizer



Polarizer, Fig. 2 Dichroic polarizer



Polarizer, Fig. 3 Polarization prism which is called Glan Taylor Polarizer

materials, and polarization of the light is useful in many application fields. Polarizers are made artificially since natural objects usually do not change the light to be always perfectly polarized.

Theory

There are two kinds of perfectly polarized light: linearly polarized light and circularly polarized light. This entry first explains linear polarizers, and then explains circular polarizers.

[Figure 1](#) is the typical illustration of linear polarizer. Linear polarizer is often expressed as a circle with some straight lines inside. The orientation of the lines expresses the orientation of oscillation of the light transmitted through the polarizer. [Figure 1](#) represents the case when the unpolarized light goes through the polarizer, and then, the electric vector field of the transmitted light oscillates only in vertical direction.

Typical polarizers are dichroic polarizers and wire-grid polarizers. The typical material used in dichroic polarizer is iodine. As is shown in [Fig. 2](#), if the long axis of iodine molecule is lining up horizontally, the light transmitted through the polarizer will be vertically polarized. In this case, the horizontal component of the light is absorbed by the iodine molecule. The same applies to the wire-grid polarizers: If the wire-grid is aligned horizontally, the transmitted light becomes vertically polarized.

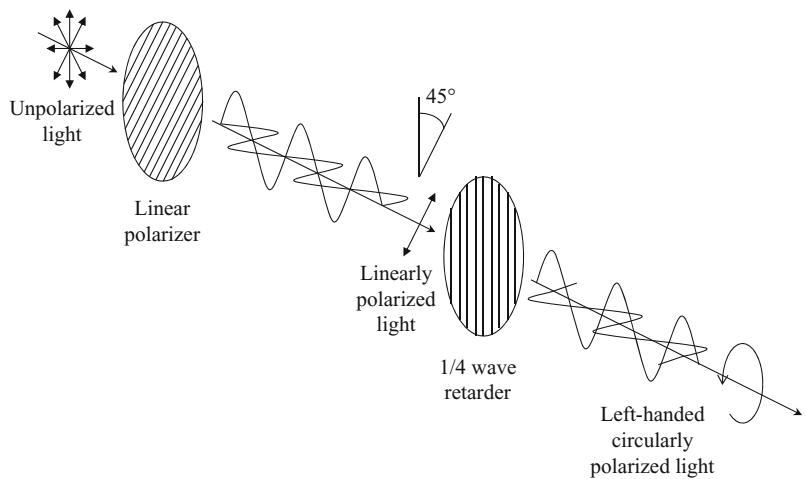
Polarizers composed of prisms are also widely used. The typical material used in these polarizers is calcite. There are many types of prism polarizers such as Glan Taylor Polarizer, Glan Thompson Polarizer, Wollaston Polarizer, Rochon Polarizer, Nicol Polarizer, etc. [Figure 3](#) is an illustration of Glan Taylor Polarizer. Two calcite prisms are separated by thin layer of the air. Calcite is a birefringent material, and the extraordinary ray is only transmitted through this polarizer.

Circularly polarized light is generated by circular polarizer. The circular polarizer is consisted of linear polarizer and 1/4 wave retarder as is shown in [Fig. 4](#). The angle between the transmission axis of linear polarizer and the fast axis of 1/4 wave retarder is set to be 45°.

Application

Liquid crystal display and liquid crystal projector use two linear polarizers in order to change the brightness of the light. In order to produce a three-dimensional view using projectors, polarized 3D glasses are often used. The viewer wears the polarized 3D glasses, and watch the screen. Two images are superimposed

Polarizer, Fig. 4 Circular polarizer



into the screen; however, each eye perceives different image. This structure makes the viewer to recognize the 3D effect.

References

1. Shurcliff WA (1962) Polarized light: production and use. Harvard University Press, Cambridge

Polarizing Film

► [Polarizer](#)

Principal Axis

► [Optical Axis](#)

Principal Component Analysis (PCA)

Takio Kurita
Graduate School of Engineering, Hiroshima
University, Higashi-Hiroshima, Japan

Synonyms

Karhunen–Loëve transform (KLT)

Definition

Principal component analysis (PCA) is a standard tool in modern data analysis and is used by almost all scientific disciplines. The goal of PCA is to identify the most meaningful basis to reexpress a given data set. It is expected that this new basis will reveal hidden structure in the data set and filter out the noise. There are so many applications such as dimensionality reduction, data compression, feature extraction, and data visualization.

Background

The modern form of PCA was formalized by Hotelling [3] who also introduced the term *principal component*. It is also known as the Karhunen–Loëve transform.

Observations are often described by several dependent variables which are, in general, intercorrelated and include noise. PCA is used to extract the important information from such observations and to reduce the noise. To achieve this goal, PCA computes a set of new orthogonal variables called *principal components* which are obtained as linear combinations of the original variables. The values of these new variables for the observations are called *factor scores*. They can be interpreted as the projections of the observations onto the principal components. PCA can be also formulated as the linear projection that minimizes the average projection cost defined as the mean squared distance between the data points and their projections [5].

Several reformulations or extensions of PCA have been proposed. PCA can be expressed as the maximum likelihood solution of a probabilistic latent variable model [6]. This reformulation is known as *probabilistic* PCA. A nonlinear generalization of PCA, which is known as kernel PCA, is also proposed by using the approach of kernel learning [8]. Sparse principal component analysis finds sparse coefficients by introducing a constraint on the norm of the coefficients [9]. This sparseness makes the interpretation of the results of PCA easier.

Theory

Let $X = [\mathbf{x}_1, \dots, \mathbf{x}_N]$ be the data set to be analyzed by PCA, where each column is a single observation described by M variables. Then the sample mean vector $\bar{\mathbf{x}}$ and the sample covariance matrix Σ can be represented by

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i \quad (1)$$

$$\Sigma = \frac{1}{N} \sum_{i=1}^N (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T = \frac{1}{N} \tilde{X} \tilde{X}^T \quad (2)$$

where the matrix \tilde{X} is defined as $\tilde{X} = [\mathbf{x}_1 - \bar{\mathbf{x}}, \dots, \mathbf{x}_N - \bar{\mathbf{x}}]$.

To extract the important information from the observations, PCA computes factor scores as linear combinations of the original variables

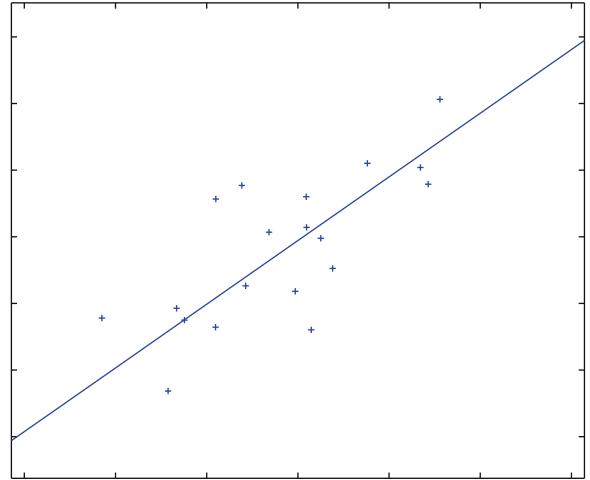
$$y_{1i} = \mathbf{a}_1^T (\mathbf{x}_i - \bar{\mathbf{x}}), \quad (i = 1, \dots, N) \quad (3)$$

where $\mathbf{a}_1 = (a_{11}, \dots, a_{M1})^T$ is a set of weights of the linear combinations. The optimum weight vector \mathbf{a}_1 is obtained so that the sample variance of the new variable is maximized under the normalization constraint $\mathbf{a}_1^T \mathbf{a}_1 = 1$. Since the sample variance is represented by

$$V(y_{11}, \dots, y_{1N}) = \mathbf{a}_1^T \Sigma \mathbf{a}_1, \quad (4)$$

the optimization problem can be defined by using Lagrange multiplier λ_1 as the maximization of the Lagrange function

$$L(\mathbf{a}_1, \lambda_1) = \mathbf{a}_1^T \Sigma \mathbf{a}_1 - \lambda_1 (\mathbf{a}_1^T \mathbf{a}_1 - 1). \quad (5)$$



Principal Component Analysis (PCA), Fig. 1 An example of principal axis for two-dimensional samples

By setting the derivative of the Lagrange function with respect to the weight vector \mathbf{a}_1 equal to zero, the solution of this problem can be obtained as a unit eigenvector of the covariance matrix Σ corresponding to the largest eigenvalue λ_1 as

$$\Sigma \mathbf{a}_1 = \lambda_1 \mathbf{a}_1. \quad (6)$$

If the vector \mathbf{a}_1 is multiplied from the left, the maximum variance is given by

$$V(y_{11}, \dots, y_{1N}) = \mathbf{a}_1^T \Sigma \mathbf{a}_1 = \lambda_1. \quad (7)$$

[Figure 1](#) shows an example of the first principal axis for the two-dimensional samples.

The second principal components

$$y_{2i} = \mathbf{a}_2^T (\mathbf{x}_i - \bar{\mathbf{x}}), \quad (i = 1, \dots, N) \quad (8)$$

can be defined as a new variable which maximizes the projected variance among all possible directions orthogonal to the first principal axis under the constraint on the normalization. The optimum coefficient vector \mathbf{a}_2 is also obtained as a unit eigenvector of the covariance matrix Σ corresponding to the second largest eigenvalue λ_2 . Similarly, additional principal components can be defined in an incremental fashion. For the general case of L -dimensional projection, the optimal linear projection of PCA



Principal Component Analysis (PCA), Fig. 2 Examples of eigenfaces computed from 200 face images

$$Y = A^T \tilde{X} \quad (9)$$

can be obtained by taking the L eigenvectors $A = [\mathbf{a}_1, \dots, \mathbf{a}_L]$ of the covariance matrix Σ corresponding to the L largest eigenvalue $\lambda_1, \dots, \lambda_L$. Then the eigenvector equation is given by

$$\Sigma A = A\Lambda, \quad (10)$$

where $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_L)$.

PCA is closely related with the singular value decomposition (SVD) of the data matrix. The data matrix \tilde{X} can be decomposed by SVD as

$$\tilde{X} = S\Delta V^T, \quad (11)$$

where S is the matrix of left singular vectors, V is the matrix of right singular vectors, and Δ is the diagonal matrix of singular values. By using this relation, the covariance matrix can be rewritten as

$$\Sigma = \frac{1}{N} \tilde{X} \tilde{X}^T = \frac{1}{N} S \Delta V^T V \Delta S^T = \frac{1}{N} S \Delta^2 S^T. \quad (12)$$

By multiplying S from the right, the eigenvector equation is obtained as

$$\Sigma S = S \frac{1}{N} \Delta^2. \quad (13)$$

This shows that the weight vectors A is equal to S and the diagonal matrix of the eigenvalues Λ is equal to $\frac{1}{N} \Delta^2$ if the number of principal components L is equal to the rank of the data matrix \tilde{X} . From these relations, the matrix of factor scores Y can be represented as

$$Y = A^T \tilde{X} = S^T S \Delta V^T = \Delta V^T. \quad (14)$$

This shows that the principal components are also obtained from the SVD of the data matrix \tilde{X} . By using this relation, the data matrix \tilde{X} can be represented as the product of the score matrix by the weight vectors A :

This means that the original data matrix \tilde{X} can be reconstructed from the factor scores.

If the number of principal components L is less than the rank of \tilde{X} , this reconstruction gives an approximation of the original data matrix. The mean squared errors between the original observations and the approximations are then given by

$$\begin{aligned} \varepsilon^2(L) &= \frac{1}{N} \sum_{i=1}^N \|(\mathbf{x}_i - \bar{\mathbf{x}}) - AA^T(\mathbf{x}_i - \bar{\mathbf{x}})\|^2 \\ &= \sum_{i=1}^{\text{rank}(\tilde{X})} \lambda_i - \sum_{i=1}^L \lambda_i \end{aligned} \quad (16)$$

which is simply the sum of the $\text{rank}(\tilde{X}) - L$ smallest eigenvalues. This means that PCA is minimizing this approximation errors by selecting the principal subspace spanned by the eigenvectors corresponding to the L largest eigenvalues.

Application

There are many applications of PCA in pattern recognition or computer vision. One of the famous applications of PCA is for face recognition [7]. Face images are decomposed into a set of characteristic feature images called “eigenfaces,” which are the eigenvectors computed by PCA to the training set of face images. Recognition is performed by projecting a new face image into the subspace spanned by the eigenfaces and then classifying the face by comparing the distances of factor scores. Figure 2 shows examples of the eigenfaces computed from 200 face images.

PCA is also used to construct 3-D object models from their appearances [4]. To obtain a low-dimensional subspace of object appearance parametrized by pose and illumination, PCA is applied to a large set of images obtained by varying pose and illumination. Then a new image is projected to the

constructed subspace and the recognition is performed on the subspace. The object's pose in the image is also determined from the position of the projection on the subspace.

References

1. Adibi H, Williams LJ (2010) Principal component analysis. Wiley Interdiscip Rev Comput Stat 2:433–459
2. Bishop CM (2006) Pattern recognition and machine learning. Springer, New York
3. Hotelling H (1933) Analysis of a complex of statistical variables into principal components. J Educ Psychol 61:417–441
4. Murase H, Nayar S (1995) Visual learning and recognition of 3-D objects from appearance. Int J Comput Vis 14:5–24
5. Pearson K (1901) On lines and planes of closest fit to systems of points in space. Lond Edinb Dublin Philos Mag J Sci Sixth Ser 2:559–572
6. Tipping ME, Bishop CM (1999) Probabilistic principal component analysis. J R Stat Soc B 61:611–622
7. Turk M, Pentland A (1991) Eigenfaces for recognition. J Cogn Neurosci 3(1):71–86
8. Schölkopf B, Smola A, Müller KR (1998) Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput 10(5):1299–1319
9. Zou H, Hastie T, Tibshirani R (2006) Sparse principal component analysis. J Comput Graph Stat 15(2):265–286

Principal Distance

►Focal length

Probabilistic Hill Climbing

►Simulated Annealing

Projection

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Related Concepts

►Affine Camera; ►Depth Distortion; ►Perspective Camera; ►Perspective Transformation; ►Weak Perspective Projection

Definition

A projection is an image of a geometric figure, with or without its appearance property, reproduced on a line, a plane, or a surface. It is usually a mapping from a higher dimensional space to a lower dimensional subspace. A camera performs a projection from a 3D scene to a 2D picture. See related concepts listed above.

Projection Matrix

►Projection Transformation

Projection Transformation

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Synonyms

Projection matrix

Related Concepts

►Affine Camera; ►Perspective Camera; ►Perspective Transformation

Definition

Projection transformation describes the transformation from a 3D scene to a 2D image.

Background

The actual projection transformation depends on the camera's projection model and is determined by the relationship between the center of projection (optical center) and the projection plane (image plane). If the center of projection is at a finite distance from the projection plane, it is a perspective transformation. If the center of projection is at infinity, the projection is

parallel, known as an orthographic projection. There are several possible transformations between the two extremes, and the reader is referred to entry “► [Affine Camera](#)” for more information.

In computer graphics a projection transformation transforms three-dimensional eye coordinates into points in three-dimensional *clip coordinates*. Besides various types of projection as mentioned above, it defines the *viewing volume*, which determines which objects or parts of objects are projected onto the screen. The viewing volume, also known as *viewing frustum*, is an six-sided enclosure defined by clipping planes: Primitives outside the viewing volume are not displayed.

Projective Reconstruction

Richard Hartley

Department of Engineering, Australian National University, ACT, Australia

Synonyms

[Projective structure and motion](#)

Related Concepts

► [Euclidean Geometry](#); ► [Exploration: Simultaneous Localization and Mapping \(SLAM\)](#)

Definition

From several images of a scene and the coordinates of corresponding points identified in the different images, it is possible to construct a three-dimensional point-cloud model of the scene and compute the camera locations. From uncalibrated images the model can be reconstructed up to an unknown projective transformation, which can be upgraded to a Euclidean model by adding or computing calibration information.

Background

Projective reconstruction refers to the computation of the structure of a scene from images taken with

uncalibrated cameras, resulting in a scene structure, and camera motion that may differ from the true geometry by an unknown 3D projective transformation.

Suppose that a set of interest points are identified and matched (or tracked) in several images. The configuration of the corresponding 3D points and the locations of the cameras that took these images are supposed unknown. The task of reconstruction is to determine the values of these unknown quantities.

Formally, assume that a set of image points $\{\mathbf{x}_{ij}\}$ are known, where \mathbf{x}_{ij} represents the image coordinates of the j -th point seen in the i -th image. It is generally not required that every point’s location be known in every image, so only a subset of all possible \mathbf{x}_{ij} are given. The structure-from-motion (SfM) problem is to determine the camera projection matrices \mathbf{P}_i and the 3D point locations \mathbf{X}_j such that the projection of the j -th point in the i th image is the measured \mathbf{x}_{ij} . Assuming a pinhole (projective) camera model, this relationship is expressed as a linear relationship:

$$\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j , \quad (1)$$

where \mathbf{P}_i is a 3×4 matrix of rank 3, \mathbf{X}_j and \mathbf{x}_{ij} are expressed in homogeneous coordinates, and the equality is intended to hold only up to an unknown scale factor λ_{ij} . More precisely, therefore, the projection equation is

$$\lambda_{ij} \mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j . \quad (2)$$

In the SfM problem, cameras \mathbf{P}_i and points \mathbf{X}_j are to be determined, given only the point correspondences.

Homogeneous Coordinates

Both 2D (image) points and 3D (world) points are most conveniently expressed in *homogeneous coordinates*. Thus, an image point \mathbf{x} is represented by a 3-vector $\mathbf{x} = (u, v, w)^\top$, known as its homogeneous representation. The relationship to the standard Euclidean (nonhomogeneous) coordinates (x, y) of the point is given by $x = u/w$ and $y = v/w$. This process of division by the final coordinate of the homogeneous vector is known as *dehomogenization*. Note that two vectors $\mathbf{x} = (u, v, w)^\top$ and $\mathbf{x}' = (u', v', w')^\top$ represent the same point in Euclidean coordinates if and only if $\mathbf{x} = k\mathbf{x}'$ for some nonzero constant k . Thus, a given point may be expressed in infinitely many different ways in homogeneous coordinates. This is analogous

with the way a given rational number has many different representations, such as $1/2 = 2/4 = 3/6 = k/2k$ for any k . One particularly convenient homogeneous representation of a point is the 3-vector with unit final coordinate: $(x, y, 1)^\top$.

Homogeneous coordinates (3-vectors) with final coefficient zero do not coincide to any real point in nonhomogeneous coordinates, since the process of dehomogenization involves division by zero. Such points are commonly known as points at infinity. The vector $(0, 0, 0)^\top$ is not considered to be a valid set of homogeneous coordinates.

In a similar way, $3D$ points are represented by homogeneous 4-vectors $\mathbf{X} = (x, y, z, t)^\top$. The main advantage of using homogeneous coordinates to represent world and image points is that Eq. (1) has a particularly simple form as a linear relationship between the homogeneous coordinates of the points.

Two homogeneous vectors differing by a constant multiplicative factor are considered to be *equivalent* representations of the same point. The set of all equivalence classes of (nonzero) homogeneous $(n + 1)$ -vectors form the *projective n -space*, \mathcal{P}^n . In studying projective reconstruction, it is conventional to consider image points to lie in projective 2-space \mathcal{P}^2 , whereas $3D$ points lie in projective 3-space \mathcal{P}^3 . This identifies the projective space \mathcal{P}^2 consisting of the (image) plane, augmented with points at infinity. Similarly, \mathcal{P}^3 consists of \mathbb{R}^3 along with a plane of points at infinity.

Ambiguity

Expressed in full generality, the solution to the reconstruction problem may only be determined up to an unknown projective transformation, applied both to points and cameras.

A projective transformation of \mathcal{P}^3 , the model for 3-space containing world points, is a mapping:

$$\mathbf{X} \mapsto \mathbf{H}\mathbf{X}$$

where \mathbf{H} is a non-singular 4×4 matrix representing a mapping between homogeneous coordinates. Using this relationship, it is easily seen that the determination of camera matrices \mathbf{P}_i and points \mathbf{X}_j cannot be unique, given only corresponding image coordinates \mathbf{x}_{ij} . Consider

$$\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j$$

$$\begin{aligned} &= (\mathbf{P}_i \mathbf{H}^{-1}) (\mathbf{H}\mathbf{X}_j) \\ &= \mathbf{P}'_i \mathbf{X}'_j . \end{aligned} \quad (3)$$

In this relationship, new points $\mathbf{X}'_j = \mathbf{H}\mathbf{X}_j$ are defined in terms of points \mathbf{X}_j and similarly new camera matrices $\mathbf{P}'_i = \mathbf{P}_i \mathbf{H}^{-1}$ in terms of the camera matrices \mathbf{P}_i . Since both $(\{\mathbf{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\mathbf{P}'_i\}, \{\mathbf{X}'_j\})$ give rise to the same projected image coordinates \mathbf{x}_{ij} , there is no way to choose between these two solutions to the reconstruction problem. In fact, there exists a complete family of solutions to the problem, corresponding to all possible choices of the matrix \mathbf{H} . All such solutions are related to each other by the application of a projective transformation and are hence called *projectively equivalent*. A particular solution, consisting of camera matrices \mathbf{P}_i and points \mathbf{X}_j satisfying Eq. (1) is known as a *projective reconstruction* of the scene, computed from the given corresponding image points.

The effect of projective ambiguity is given shown in Fig. 1.

The Projective Reconstruction Theorem

The above analysis does not rule out the possibility that other solutions to this reconstruction problem exist, not related to a particular obtained solution by any projective transformation.

However, this possibility is excluded by the projective reconstruction theorem, which essentially says that if the set of corresponding points \mathbf{x}_{ij} are sufficiently numerous (at least 8 in number), and do not lie in some degenerate configuration, then the solution to the reconstruction problem is unique up to a projective transformation.

The exact statement of the theorem requires the definition of the *fundamental matrix* which will be considered next.

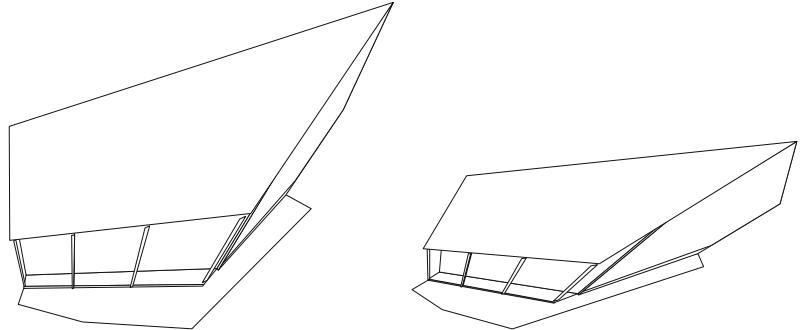
Theory and Applications

Two View Reconstruction

Consider the reconstruction problem for only two images. Rather than using a subscript, entities belonging to the second camera are distinguished by a prime. Thus, the given input to this problem consists of corresponding points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$; $i = 1, \dots, n$, where the points \mathbf{x}_i come from one image and the \mathbf{x}'_i are the corresponding points in the other.

Projective Reconstruction, Fig. 1

Projective reconstruction.
 (Top) Original image pair.
 (Bottom) Two views of a 3D projective reconstruction of the scene. The *lines* of the wireframe link the computed 3D points. The reconstruction requires no information about the camera matrices or information about the scene geometry. In a projective reconstruction, the resulting model is distorted by an arbitrary projective transformation from the true geometrically correct model (Figures derived from [15])



Let the camera matrices (unknown) be \mathbf{P} and \mathbf{P}' , and let \mathbf{X}_i be the 3D point corresponding to the image points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. The projection equations are

$$\begin{aligned}\lambda_i \mathbf{x}_i &= \mathbf{P} \mathbf{X}_i \\ \lambda'_i \mathbf{x}'_i &= \mathbf{P}' \mathbf{X}_i,\end{aligned}$$

where the scale factors λ_i and λ'_i are explicitly written (but are unknown). These equations may be written in a single system

$$\left[\begin{array}{ccc} \mathbf{P} & \mathbf{x}_i & \\ \mathbf{P}' & \mathbf{x}'_i & \end{array} \right] \left(\begin{array}{c} \mathbf{X}_i \\ -\lambda_i \\ -\lambda'_i \end{array} \right) = \mathbf{0}. \quad (4)$$

Since this equation must have a nonzero solution $(\mathbf{X}_i, -\lambda_i, -\lambda'_i)^\top$, the determinant of the matrix on the left (which shall be denoted as \mathbf{A}) must be zero. Since the point coordinates \mathbf{x}_i and \mathbf{x}'_i each appear in a single column, it follows that the determinant is a bilinear expression in $(\mathbf{x}_i, \mathbf{x}'_i)$, and hence, the equation $\det(\mathbf{A}) = 0$ can be written in the form

$$\mathbf{x}'^\top \mathbf{F} \mathbf{x}_i = 0, \quad (5)$$

where \mathbf{F} is a 3×3 matrix depending only on the two camera matrices \mathbf{P} and \mathbf{P}' . Consequently, this

equation will hold for any pair of corresponding points $(\mathbf{x}_i, \mathbf{x}'_i)$. The matrix \mathbf{F} is called the *fundamental matrix* corresponding to the camera pair $(\mathbf{P}, \mathbf{P}')$.

Closer examination of the matrix \mathbf{A} appearing in (4) reveals the exact form of the matrix \mathbf{F} . Expanding $\det(\mathbf{A})$ by cofactors down the last two columns yields the following formula:

$$F_{jk} = (-1)^{j+k} \det \left[\begin{array}{c} \mathbf{P}^{(\sim j)} \\ \mathbf{P}'^{(\sim k)} \end{array} \right], \quad (6)$$

where $\mathbf{P}^{(\sim j)}$ is the 2×4 matrix obtained by omitting the j th row of \mathbf{P} and $\mathbf{P}'^{(\sim k)}$ is similarly defined.

Another way of writing the Eq. (5) is

$$(\mathbf{x}_i \otimes \mathbf{x}'_i)^\top \mathbf{f} = 0, \quad (7)$$

where $(\mathbf{x}_i \otimes \mathbf{x}'_i)^\top$ is the vector

$$(u'_i u_i, u'_i v_i, u'_i w_i, v'_i u_i, v'_i v_i, v'_i w_i, w'_i u_i, w'_i v_i, w'_i w_i)^\top \quad (8)$$

expressed in terms of coordinates $\mathbf{x}_i = (u_i, v_i, w_i)^\top$ and $\mathbf{x}'_i = (u'_i, v'_i, w'_i)^\top$. Further, \mathbf{f} is the vector $(F_{11}, F_{12}, \dots, F_{33})^\top$ made up of the entries of the fundamental matrix \mathbf{F} .

Computing the Fundamental Matrix

Note that the Eq. (7) is a linear equation with unknowns equal to the entries of the fundamental matrix. The explicit form of the equation is given by (8). Given $n \geq 8$ point correspondences, one has a set of linear equations

$$\mathbf{A}\mathbf{f} = \mathbf{0},$$

where \mathbf{A} is an $n \times 9$ matrix, with entries determined by the coordinates of the matched image points. This set of equations is solved to find \mathbf{f} .

Since this is a set of homogeneous equations, there is a solution $\mathbf{f} = \mathbf{0}$, which is not interesting; a nonzero solution is required. With exactly 8-point correspondences, there is an exact solution to this problem. With more points, a least-squares solution is computed. This is most conveniently done by solving the problem

$$\begin{aligned} &\text{Minimize } \|\mathbf{A}\mathbf{f}\| \\ &\text{subject to } \|\mathbf{f}\| = 1, \end{aligned}$$

where the condition $\|\mathbf{f}\| = 1$ is imposed in order to obtain a unique solution (apart from sign). The solution is the eigenvector of $\mathbf{A}^\top \mathbf{A}$ corresponding to the smallest eigenvalue. Alternatively, if \mathbf{A} has singular value decomposition

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^\top,$$

then the required \mathbf{f} is the last column of \mathbf{V} (assuming that the singular values of \mathbf{D} are in descending order). Once the solution \mathbf{f} is found, the fundamental matrix \mathbf{F} is reconstituted from the entries of \mathbf{f} .

The algorithm just described is the so-called 8-point algorithm for computing the fundamental matrix [20]. In order to get good results, it is necessary to preprocess the input image coordinates, using the so-called *normalized* 8-point algorithm, which will be described later.

Projective Reconstruction Theorem

This discussion leads to the basic theorem of projective reconstruction, which states that under appropriate conditions, the reconstruction of a scene from sufficiently many point correspondences in two views is unique up to projective transformation.

Theorem 1 *Let $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$; $i = 1, \dots, n$ be point correspondences in two views and let $(\mathbf{P}, \mathbf{P}', \{\mathbf{X}_i\})$ be a pair of camera matrices, and some 3D points forming a 3D reconstruction specifically stated*

$$\begin{aligned} \lambda_i \mathbf{x}_i &= \mathbf{P} \mathbf{X}_i \\ \lambda'_i \mathbf{x}'_i &= \mathbf{P}' \mathbf{X}_i \end{aligned} \quad (9)$$

for some unknown $\lambda_i, \lambda'_i \neq 0$. Let \mathbf{H} be an invertible 4×4 matrix \mathbf{H} , and define

$$\begin{aligned} \tilde{\mathbf{P}} &= \mathbf{P} \mathbf{H}^{-1} \\ \tilde{\mathbf{P}}' &= \mathbf{P}' \mathbf{H}^{-1} \\ \tilde{\mathbf{X}}_i &= \mathbf{H} \mathbf{X}_i. \end{aligned} \quad (10)$$

Then the triple $(\tilde{\mathbf{P}}, \tilde{\mathbf{P}}', \{\tilde{\mathbf{X}}_i\})$ is also a reconstruction satisfying the Eq. (9).

Furthermore, if the set of vectors $\{\mathbf{x}_i \otimes \mathbf{x}'_i\}$ has rank 8 (spans a linear subspace of dimension 8 in \mathbb{R}^9), then any reconstruction $(\tilde{\mathbf{P}}, \tilde{\mathbf{P}}', \{\tilde{\mathbf{X}}_i\})$ satisfying (9) is related to the original reconstruction $(\mathbf{P}, \mathbf{P}', \{\mathbf{X}_i\})$ by (10) for some non-invertible matrix \mathbf{H} .

This theorem was proved in [7, 16].

Note the condition that the set of vectors $\{\mathbf{x}_i \otimes \mathbf{x}'_i\}$ has rank 8 is exactly the condition that the set of equations of the form (7) has a unique solution. If the rank of the vectors $\{\mathbf{x}_i \otimes \mathbf{x}'_i\}$ is equal to 9, then there is no solution to the Eq. (7) and the point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ cannot be a valid set of points corresponding to projections of a set of 3D points in two images.

If the vectors $\{\mathbf{x}_i \otimes \mathbf{x}'_i\}$ span a space of dimension less than 8 (for instance, if there are fewer than 8-point correspondences), then there is not a unique matrix \mathbf{F} satisfying the condition (5), and the reconstruction may not be unique up to projectivity.

Extraction of Camera Matrices

Once the fundamental matrix has been computed, it is possible to extract a pair of camera matrices directly from \mathbf{F} . The decomposition is not unique, since according to Theorem 1, there are many pairs of camera matrices $(\mathbf{P}, \mathbf{P}')$ that correspond to the same fundamental matrix \mathbf{F} . It is always possible to assume that one of the camera matrices is of the form $\mathbf{P} = [\mathbf{I} \mid \mathbf{0}]$, so the problem is simply to compute the other camera matrix \mathbf{P}' .

An algorithm to do this is as follows:

1. Compute the singular value decomposition

$$\mathbf{F} = \mathbf{U}\mathbf{D}\mathbf{V}^\top,$$

where $D \approx \text{diag}(p, q, 0)$. Note that since F should have rank 2, the last singular value should be zero, but with noise this will not exactly hold.

2. Define matrices

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} ; Z = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Define $\hat{D} = \text{diag}(p, q, r)$ for some value of r , and observe that

$$F = UDU^T = (UZU^T)(UW^T\hat{D}V^T) = SM, \quad (11)$$

where S is a skew-symmetric matrix and M is defined by this equation. The value of r may be arbitrarily chosen.

3. A pair of camera matrices corresponding to the fundamental matrix F is now

$$P = [I \mid \mathbf{0}] ; P' = [M \mid \mathbf{u}_3] \quad (12)$$

where \mathbf{u}_3 is the third column of U .

Notes

1. The vector \mathbf{u}_3 satisfies $\mathbf{u}_3^T F = \mathbf{u}_3^T S = \mathbf{0}$; it is the generator of the left null-space of F .
2. The value of r , the last diagonal entry of \hat{D} , may be chosen arbitrarily, but a good choice is to set $r = (p + q)/2$ so that M is far from singular.
3. If $r = 0$, the matrix M is singular, but has a particularly simple form, namely, $M = SF$. The corresponding camera $P' = [SF \mid \mathbf{u}_3]$ is sometimes used, but it has the property that the right-hand 3×3 block is singular, so the camera center lies at a nonfinite point.

Complete Projective Reconstruction Algorithm

It is now possible to state a complete algorithm for projective reconstruction of a scene from two images. Suppose a set of image correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$; $i = 1, \dots, n$ are given.

1. From the image correspondences, compute the fundamental matrix F linearly from Eq.(7), as described in section 3.1.
2. From F find the two camera projection matrices $P = [I \mid \mathbf{0}]$ and $P' = [M \mid \mathbf{t}]$, as in section 3.2.

3. The corresponding 3D points \mathbf{X}_i may be computed linearly as the least-squares solution to Eq.(4). This process is called *triangulation*.

The linear triangulation method via Eq.(4) does not give optimal results. A method optimal in the presence of noise is given in [13, 14].

The Normalized Eight-Point Algorithm

It was pointed out in [11] that the simple version of the 8-point algorithm given above can lead to very poor results in some circumstances, but this problem is largely alleviated by simple normalization of the image coordinates.

The issue with the 8-point algorithm for computing F is that the vector (Eq. 8) expressed in terms of image-point coordinates can contain entries of widely different magnitude. This leads to poor conditioning of the linear equations used to solve for F . In addition, the results are dependent on the particular coordinate system (origin and scale) used to express image points.

Given corresponding image points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, one may define normalized coordinates $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}'_i$ obtained from the original coordinates by the following operations:

1. Each \mathbf{x}_i is replaced by $\mathbf{x}_i - \bar{\mathbf{x}}$, where $\bar{\mathbf{x}}$ is the mean (barycenter) of all the coordinates \mathbf{x}_i . This corresponds to a shift of the coordinate origin so that the mean of the \mathbf{x}_i is at the origin.
2. The points are scaled so that their average (alternatively, their root-mean-squared) Euclidean distance from the origin is equal to $\sqrt{2}$. This is done by applying a common scaling to all the points $\mathbf{x}_i - \bar{\mathbf{x}}$. The resulting point is $\hat{\mathbf{x}}_i$.

The reason for choosing an average distance of $\sqrt{2}$ is so that the *average* point has homogeneous coordinates $(1, 1, 1)^T$.

One applies these operations to the points \mathbf{x}_i and \mathbf{x}'_i independently. Note that both normalization steps are simple affine transformations of the points. These transformations may be written as

$$\hat{\mathbf{x}}_i = T\mathbf{x}_i ; \hat{\mathbf{x}}'_i = T'\mathbf{x}'_i \quad (13)$$

where T and T' are 3×3 matrices acting on the homogeneous representations of the points.

Once this normalization has taken place, the computation of the fundamental matrix and the complete projective reconstruction may be carried out using the

normalized coordinates. The result is a fundamental matrix \hat{F} satisfying the condition

$$\hat{\mathbf{x}}_i^T \hat{F} \hat{\mathbf{x}}_i = 0 \quad (14)$$

from which by substitution using (13), one has

$$(\mathbf{x}'_i^T T'^T) \hat{F} (T \mathbf{x}_i) = 0 = \mathbf{x}'_i^T F \mathbf{x}_i .$$

From this it follows that $F = T'^T \hat{F} T$ is the fundamental matrix corresponding to the original points.

Similarly, if \hat{P} and \hat{P}' are camera matrices belonging to a reconstruction from the normalized image coordinates, then

$$\hat{\mathbf{x}}_i = \hat{P} \hat{\mathbf{X}}_i ; \quad \hat{\mathbf{x}}'_i = \hat{P}' \hat{\mathbf{X}}_i .$$

Once more, substituting for $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}'_i$, it follows that

$$\mathbf{x}_i = T^{-1} \hat{P} \hat{\mathbf{X}}_i ; \quad \mathbf{x}'_i = T'^{-1} \hat{P}' \hat{\mathbf{X}}_i$$

which implies that the reconstruction $(P, P', \{\mathbf{X}_i\})$ for the original points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ is given by

$$P = T^{-1} \hat{P} ; \quad P' = T'^{-1} \hat{P}' ; \quad \mathbf{X}_i = \hat{\mathbf{X}}_i .$$

This normalized 8-point algorithm gives markedly superior results to the unnormalized algorithm, which should never be used directly. For more details and analysis, see [11].

Three View Reconstruction

The 8-point algorithm and other methods involving the fundamental matrix are useful for reconstruction from two views.

If three images of a scene are available and point correspondences are known across all three views, then such linear methods can be extended to three-image reconstruction, using the *trifocal tensor*. This is an extension of the fundamental matrix to three views.

In this analysis of three-view reconstruction, it is convenient from a notational point of view to denote the three camera matrices as A, B, and C, instead of P_1 , P_2 , and P_3 .

Given a three-way image-point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i \leftrightarrow \mathbf{x}''_i$, the goal is to find camera matrices A, B, and C and points \mathbf{X}_i such that

$$\mathbf{x}_i = A \mathbf{X}_i ; \quad \mathbf{x}'_i = B \mathbf{X}_i ; \quad \mathbf{x}''_i = C \mathbf{X}_i . \quad (15)$$

This may be written in a form similar to (4), as follows:

$$\begin{bmatrix} A & \mathbf{x}_i \\ B & \mathbf{x}'_i \\ C & \mathbf{x}''_i \end{bmatrix} \begin{pmatrix} \mathbf{X}_i \\ -\lambda_i \\ -\lambda'_i \\ -\lambda''_i \end{pmatrix} = \mathbf{0} . \quad (16)$$

In this case, the 9×7 matrix on the left is not square. Nevertheless, since there is a solution $(\mathbf{X}_i, -\lambda_i, -\lambda'_i, -\lambda''_i)^T$, the matrix must be rank-deficient. Consequently, any 7×7 submatrix must have vanishing determinant. Each such determinant implies a trilinear relationship between the coefficients of the matching points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i \leftrightarrow \mathbf{x}''_i$.

It is not necessary to consider all possible 7×7 submatrices to obtain useful relationships. Given three camera matrices A, B, and C, one can define a triply indexed entity \mathcal{T}_i^{qr} :

$$\mathcal{T}_i^{qr} = (-1)^{i+1} \det \begin{bmatrix} A^{(i)} \\ B^{(q)} \\ C^{(r)} \end{bmatrix} . \quad (17)$$

Here, all indices range from 1 to 3. Further, $B^{(q)}$ and $C^{(r)}$ represent rows q and r of the matrices A and B, whereas $A^{(i)}$ means the matrix A with row i omitted. This results in a 4×4 matrix, whose determinant with the indicated sign is the chosen value \mathcal{T}_i^{qr} . This triply indexed set of 27 values is known as the *trifocal tensor* corresponding to the three cameras. Note that this tensor depends only on the camera matrices and not any image points.

Now, it may be shown [12, 15] that the coordinates of any matching triple $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i \leftrightarrow \mathbf{x}''_i$ satisfy trilinear relations:

$$\sum_{i,j,k,q,r=1}^3 x^i x'^j x''^k \epsilon_{jqu} \epsilon_{krv} \mathcal{T}_i^{qr} = 0_{uv} . \quad (18)$$

This relation is to be interpreted as follows:

1. The indices on the point coordinates, such as x^i , denote the i th coordinate of the homogeneous vector representing the point $\mathbf{x} = (x^1, x^2, x^3)^T$.
2. The symbol ϵ_{jqu} (and similarly ϵ_{krv}) has value 0 unless j, q , and u are all distinct; otherwise, it is +1

if jqu is an even permutation of the three indices 1, 2, and 3 and -1 if it is an odd permutation.

3. The indices u and v are free indices, and each choice of u and v leads to a different trilinear relation, for a total of 9 distinct relations. However, only 4 of these relations are linearly independent.

In the case where the first camera matrix A has the canonical form $[I \mid 0]$, the expression (Eq. 17) for the trifocal tensor may be written simply as

$$\mathcal{T}_i^{qr} = b_i^q c_4^r - b_4^q c_i^r, \quad (19)$$

where b_i^q is the element in row q and column i of B and c_i^r is defined analogously.

A complete three-view reconstruction algorithm can then be outlined as follows:

1. From point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i \leftrightarrow \mathbf{x}''_i$ for $i = 1, \dots, n$, each relation of the form (18) gives 4 linearly independent linear constraints on the entries of the trifocal tensor. From 7-point correspondences there are sufficiently many equations to compute \mathcal{T}_i^{qr} linearly.
2. As with two-view reconstruction, it is possible to determine the form of the two other camera matrix B and C from the entries of the trifocal tensor using the formula (19).
3. Finally, by triangulation from three views based on the Eq. (16), one can find the image points \mathbf{X}_i , completing the reconstruction from three views.

A few more comments.

1. In the definition (18), the first camera matrix A is treated differently from the two others (in that two rows of A appear in the determinant, but only one from B and C). There are two other similarly defined trifocal tensors in which matrices B or C are distinguished in this way.
2. Unlike with the fundamental matrix, there are relations similar to (18) that hold for line correspondences or mixed line and point correspondences. Thus, computation of the trifocal tensor and hence projective reconstruction is possible not only from point correspondences but from mixed correspondences of this type.

Minimal Configurations

The reconstruction algorithms from two or three views described in Sections “Two View Reconstruction” and “Three View Reconstruction” require 8 or 7 points,

respectively. However, it is possible to carry out reconstruction using only 7 points from 2 views, or as few as 6 points from 3 views.

From two views, the algorithm is easily explained. Given only 7-point correspondences, the set of equations $\mathbf{x}_i'^\top \mathbf{F} \mathbf{x}_i = 0$ represents a set of 7 homogeneous equations in the 9 entries of F . The solution to this equation set is a two-parameter family $F = \lambda F_1 + \mu F_2$ where F_1 and F_2 are determined by solving this system.

The condition that the fundamental matrix F must be a singular matrix gives a further equation $\det F = 0$. Since F is a 3×3 matrix, this leads to a cubic homogeneous equation in λ and μ . Solving this cubic equation gives either one or three real solutions for the ratio $\lambda : \mu$ and hence one or three solutions (determined as ever up to scale) for the fundamental matrix F . In short, from 7-point correspondences one or three possible fundamental matrices may be computed. From these possible values of F , the rest of the method described previously will lead to a projective reconstruction, in fact either a unique or three possible reconstructions.

A method for computing the projective reconstruction from three views of 6 points is described in [27].

Factorization Algorithms

The algorithms described previously for projective reconstruction work on two or three images. In many cases, one has many more images of a scene to use for reconstruction. To handle this situation, a variant of the Tomasi-Kanade factorization algorithm [30] may be used to do reconstruction from many views at once. This is the algorithm of Sturm and Triggs [29] for projective reconstruction.

As input, consider a set of image points \mathbf{x}_{ij} for $i = 1, \dots, m$ and $j = 1, \dots, n$, where \mathbf{x}_{ij} represents the image of the j th point in the i th image. It is assumed (and required) that every point should be visible in every image, so \mathbf{x}_{ij} is defined for all (i, j) .

The projection equations are of the form

$$\lambda_{ij} \mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \quad (20)$$

where the constants λ_{ij} are required scale factors, the so-called *projective depths* of the points. This set of equations may be put together in one matrix equation as follows:

$$\begin{aligned} & \begin{bmatrix} \lambda_{11}\mathbf{x}_{11} & \lambda_{12}\mathbf{x}_{12} & \dots & \lambda_{1n}\mathbf{x}_{1n} \\ \lambda_{21}\mathbf{x}_{21} & \lambda_{22}\mathbf{x}_{22} & \dots & \lambda_{2n}\mathbf{x}_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ \lambda_{m1}\mathbf{x}_{m1} & \lambda_{m2}\mathbf{x}_{m2} & \dots & \lambda_{mn}\mathbf{x}_{mn} \end{bmatrix} \\ &= \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \dots & \mathbf{X}_n \end{bmatrix}. \quad (21) \end{aligned}$$

In this equation the matrix on the left has dimension $3m \times n$, since each $\lambda_{ij}\mathbf{x}_{ij}$ is a 3-vector. This set of equations has the form

$$\Lambda \odot W = PX \quad (22)$$

where

$$\Lambda = \begin{bmatrix} \lambda_{11} & \dots & \lambda_{1n} \\ \vdots & \ddots & \vdots \\ \lambda_{m1} & \dots & \lambda_{mn} \end{bmatrix}; W = \begin{bmatrix} \mathbf{x}_{11} & \dots & \mathbf{x}_{1n} \\ \vdots & \ddots & \vdots \\ \mathbf{x}_{m1} & \dots & \mathbf{x}_{mn} \end{bmatrix} \quad (23)$$

and \odot is to be interpreted as an elementwise product, so that $\Lambda \odot W$ is the matrix on the left of (21).

From the form of (21), it is evident that the matrix on the right has rank 4, since it is the product PX of matrices of dimension $3m \times 4$ and $4 \times n$. This is a low-rank constraint on the matrix $\Lambda \odot W$ of depth-weighted image coordinates.

Unfortunately, although the matrix W of image coordinates is known, the matrix Λ of projective depths is not known. With an incorrect set of projective depths, the matrix $\Lambda \odot W$ will not have the expected rank 4. This suggests an algorithm in which the factorization and the projective depths are estimated alternately as follows:

1. Form the matrix W of homogeneous image coordinates as given in (Eq. 23), and define Λ^0 in which all $\lambda_{ij}^0 = 1$. Then carry out the following steps iteratively for $k = 0, \dots, N$:
 - (a) Find the closest rank-4 matrix W^k to $\Lambda^k \odot W$.
 - (b) Define Λ^{k+1} to be the matrix of weights λ_{ij}^{k+1} so that $\Lambda^{k+1} \odot W$ is as close as possible to W^k under Frobenius norm.
2. Compute a final factorization $W^N = PX$, to obtain P and X providing the camera matrices and point locations, respectively.

In step 1(a), the low-rank factorization is carried out by singular value decomposition. Suppose $\Lambda^k \odot W = UDV^\top$. Let \hat{D} be the matrix obtained by setting all but the four first (largest) diagonal entries of D to zero. Then set $W^k = U\hat{D}V^\top$. The number of iterations N is vaguely defined in this algorithm. The intention is to continue until “convergence,” but as will be remarked below, continuing to convergence is problematic.

Variants of the Method

It has been observed [24] that the bare projective algorithm as given above will converge to a *trivial* limit in which all the values of λ_{ij} will be zero, except for those values in 4 columns of Λ . This solution is spurious, since zero values of the projective depths are not possible for a geometrically valid reconstruction. In addition, convergence is very slow. Therefore, different variants on the algorithm have been proposed, as follows:

1. In the original paper of Sturm and Triggs [29], an initialization of the projective depths is proposed, in which projective depths are derived from two-view reconstructions.
2. A viable strategy is to carry out only a fixed small number of alternation steps, since this significantly improves the solution without encountering a trivial solution.
3. A further step of normalization of the projective depths λ_{ij} may be used [15]. Observe that if $\lambda_{ij}\mathbf{x}_{ij} = p_i\mathbf{X}_j$, for all (i, j) , then for any constants c_i and d_j ,

$$c_i d_j \lambda_{ij} \mathbf{x}_{ij} = (c_i p_i) (d_j \mathbf{X}_j). \quad (24)$$

Thus, each λ_{ij} may be replaced by $c_i d_j \lambda_{ij}$ without materially changing the factorization. Thus, one may at will multiply each i th row of Λ by c_i and the j th column by a constant d_j . In [15] it is suggested that constants c_i and d_j may be chosen so that first the rows and then the columns of Λ sum to unity. However, no analysis of this normalization procedure is given there.

4. More complex schemes for normalization schemes are given in [24] and [21, 22], for which convergence to a meaningful (local) minimum of some cost function is demonstrated.
5. Methods to accommodate missing data or outliers in projective factorization algorithms have been

proposed. Though many algorithms have addressed missing data in matrix factorization (for instance, [2–4, 18, 28]), a notable paper addressing projective factorization specifically is [5].

6. L_1 -factorization has been recognized as more robust alternative to matrix factorization; an effective method is given in [6].

Bundle Adjustment

Given measured image points \mathbf{x}_{ij} in several images, the projection equations $\lambda_{ij} \mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j$ cannot be satisfied exactly if there is any inaccuracy, or noise, in the measurements. Therefore, in finding the projection matrices \mathbf{P}_i and 3D points \mathbf{X}_j to satisfy these equations, it is appropriate to find an approximate solution. Typically, this solution will be one that minimizes some appropriate cost function representing a residual error in the solution.

Since errors arise in the measurement of the coordinates of image points, it is appropriate to seek a solution that minimizes the error with respect to the measured image coordinates. This corresponds to choosing a cost function of the form

$$C(\{\mathbf{X}_j\}, \{\mathbf{P}_i\}) = \sum_{i,j \in \mathcal{N}} d(\mathbf{x}_{ij}, \mathbf{P}_i \mathbf{X}_j)^2, \quad (25)$$

where \mathcal{N} is a set of pairs (i, j) for which \mathbf{x}_{ij} is measured. Further, $d(\mathbf{x}_{ij}, \mathbf{P}_i \mathbf{X}_j)$ represents the Euclidean distance in the two-dimensional image plane between the measured point \mathbf{x}_{ij} and the projected point $\mathbf{P}_i \mathbf{X}_j$. This is commonly referred to as the *reprojection error*. The cost is to be minimized over all choices of \mathbf{P}_i and \mathbf{X}_j . This is a nonlinear function. The choice of the squared distance means that a nonlinear least-squares cost function is to be minimized. The motivation for this choice is the observation that the solution to this least-squares problem represents the maximum likelihood (ML) solution, under the assumption that each image measurement error conforms to an isotropic Gaussian distribution, each point measurement being independent of the others.

Minimizing the cost function (25) over all choices of the variables \mathbf{P}_i and \mathbf{X}_j is known as *bundle adjustment*. Since this is a nonlinear optimization problem,

an iterative algorithm is required. The most common algorithm used to minimize this cost function is the Levenberg-Marquardt algorithm [15, 19, 23, 31]. In order to converge to the globally optimal solution, a good initial solution is necessary. Such an initial solution is found by applying any of the algorithms previously described in this chapter.

Robust Cost Functions

The cost function (25) is suitable and represents the ML solution if the measured point coordinates conform to a Gaussian distribution, and may be used if there are no gross errors (outliers) among the measured points. In most cases, this is unlikely, and a more robust cost function is to be preferred. In this case, the squared Euclidean distance function $d(\cdot, \cdot)^2$ is replaced by some other function $f(\cdot, \cdot)$ that is more tolerant of outliers, meaning that $f(\mathbf{x}, \mathbf{y})$ grows less rapidly than $d(\mathbf{x}, \mathbf{y})^2$ as the distance between the two arguments \mathbf{x} and \mathbf{y} increases. A good choice of robust cost function is the Huber cost function [15, 17]:

$$C(\{\mathbf{X}_j\}, \{\mathbf{P}_i\}) = \sum_{i,j \in \mathcal{N}} H(d(\mathbf{x}_{ij}, \mathbf{P}_i \mathbf{X}_j))^2, \quad (26)$$

where $H(x)^2$ is quadratic for $|x| < \delta$ and linear for $|x| \geq \delta$, and δ is some threshold approximately equal to the standard deviation of the measurements.

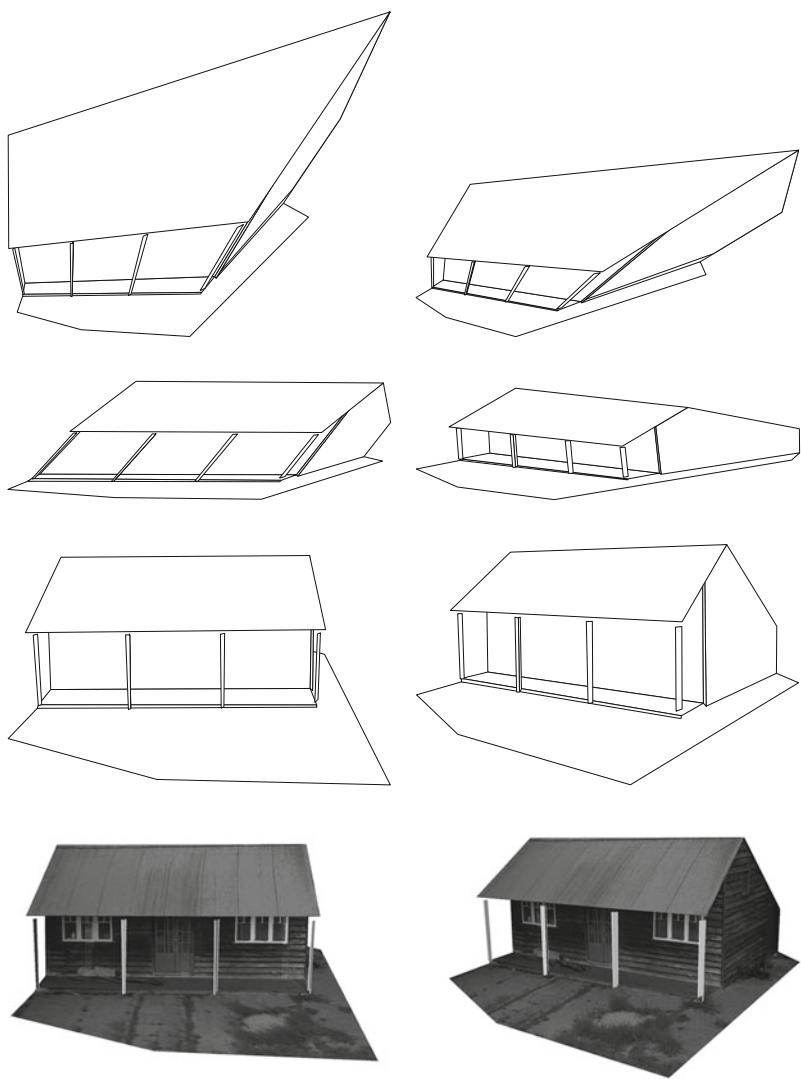
Sparse Methods

A reasonable sized reconstruction problem may involve 1,000 camera matrices \mathbf{P}_i and 100,000 points \mathbf{X}_j . Consequently, the cost function (25) depends on a large number of variables (311,000 parameters if the cameras are parametrized by 11 parameters). Since the central step in the Levenberg-Marquardt optimization process involves the solution of equations to compute the update of the parameters, this would involve solving a very large set of equations in all the variables. For a dense set of equations in 300,000 parameters, this would be almost impossible.

Fortunately, the set of equations involved in this update process is quite sparse, so the problem is tractable. To see this, note that if a single point \mathbf{X}_j is moved, then only the image points \mathbf{x}_{ij} involving this point are affected. Similarly, if some camera matrix \mathbf{P}_i is altered, then only image points \mathbf{x}_{ij} are changed. This means that each image measurement depends only

Projective Reconstruction, Fig. 2

Stratification. The projective reconstruction (top row) obtained by uncalibrated reconstruction techniques is first upgraded to an affine reconstruction (second row). In the affine reconstruction, parallel lines in the image are parallel in the reconstruction, but geometric structures are still skewed. In the final stage of the reconstruction, the true Euclidean model (third row) is computed, in which angles and dimensions are correct up to an indeterminate scale. The fourth row shows two views of the texture-mapped model (Figures derived from [15])



on the parameters of one 3D point and one camera. This sparse dependence structure for the cost function results in a special sort of sparse structure for the Jacobian matrix. Sparse solution methods may then be used to accelerate the update step and allow it to be run in reasonable time. Methods that are used for this numerical problem include the Schur complement method [15], in which the sparseness of the Jacobian is used to allow the camera updates to be computed first, followed by the point updates. The exact form of the equations is given in [15]. Alternatively, conjugate gradient methods [1] may be used; in such methods the sparseness of the equation set lends itself naturally to sparse methods.

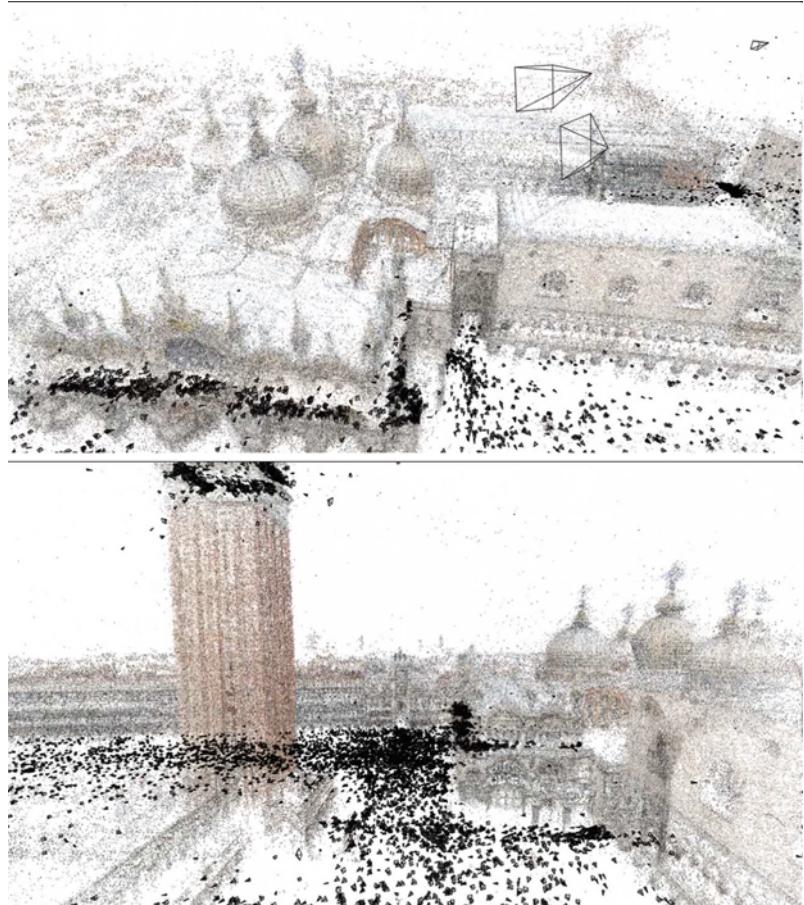
Euclidean Update

A projective reconstruction may be used as an initial step towards a geometrically correct (Euclidean) reconstruction. There are various ways in which this can be done:

1. By determining or knowing the calibration of the cameras. The camera calibration may be known a priori, or determined through the process of auto-calibration [9]. Constraints on the camera parameters, such as known focal length, or an assumption that some cameras have the same shared internal parameters, may be enforced easily during bundle adjustment. Automatic methods for auto-calibration

Projective Reconstruction, Fig. 3

Views of reconstruction of San Marco, Venice, from Flickr images. The *top* image shows San Marco Cathedral and the doge's palace. Below is shown the campanile at *left* and the palace on the *right*. Black pyramids show the position and orientation of the cameras (Figures are reproduced with thanks to Noah Snavely)



often compute an affine reconstruction first, followed by an update to a Euclidean reconstruction and full determination of the camera calibration parameters [8, 10, 25]. This process is known as stratification.

2. By the knowledge of the 3D Euclidean coordinates of some number of *ground-control points*, at least five such points are required [16].
3. If partial camera calibration is known, the full calibration and Euclidean reconstruction may be computed more simply than if no calibration information is given. A notable paper demonstrating this is [26] and more details on self-calibration given different types of partial camera calibration are given in [15].

Figure 2 illustrates the steps from projective to Euclidean reconstruction via stratification.

A large-scale reconstruction, computed from thousands of images, is shown in **Fig. 3**.

References

1. Agarwal S, Snavely N, Seitz SM, Szeliski R (2010) Bundle adjustment in the large. In: Proceedings of the 11th European conference on computer vision: Part II (ECCV'10). Springer, Berlin, pp 29–42
2. Brand M (2002) Incremental singular value decomposition of uncertain data with missing values. In: Proceedings of the 7th European conference on computer vision, part I (ECCV). Lecture notes in computer science, vol 2350, Copenhagen, Denmark. Springer, pp 707–720
3. Brandt S (2002) Closed-form solutions for affine reconstruction under missing data. In: Proceedings of the 7th European conference on computer vision, part I (ECCV). Lecture notes in computer science, vol 2350, Copenhagen, Denmark. Springer, pp 109–114
4. Buchanan AM, Fitzgibbon AW (2005) Damped Newton algorithms for matrix factorization with missing data. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Diego, vol 2, pp 316–322
5. Dai Y, Li H, He M (2010) Element-wise factorization for N-view projective reconstruction. In: Proceedings of

- the European conference on computer vision (ECCV), Heraklion
6. Eriksson A, van den Hengel A (2010) Efficient computation of robust low-rank matrix approximations in the presence of missing data using the ℓ_1 norm. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Francisco, pp 771–778
 7. Faugeras OD (1992) What can be seen in three dimensions with an uncalibrated stereo rig? In: Proceedings of the 2nd European conference on computer vision (ECCV), Santa Margherita Ligure, Italy, pp 563–578. Springer
 8. Faugeras OD (1995) Stratification of three-dimensional vision: projective, affine, and metric representation. *J Opt Soc Am A* 12:465–484
 9. Faugeras OD, Luong Q, Maybank S (1992) Camera self-calibration: theory and experiments. In: Proceedings of the 2nd European conference on computer vision (ECCV), Santa Margherita Ligure, Italy. Springer, pp 321–334
 10. Hartley RI (1994) Euclidean reconstruction from uncalibrated views. In: Mundy J, Zisserman A, Forsyth D (eds) Applications of invariance in computer vision. Lecture notes in computer science, vol 825. Springer, pp 237–256
 11. Hartley RI (1997) In defense of the eight-point algorithm. *IEEE Trans Pattern Anal Mach Intell* 19(6):580–593
 12. Hartley RI (1997) Lines and points in three views and the trifocal tensor. *Int J Comput Vis* 22(2):125–140
 13. Hartley RI, Sturm P (1994) Triangulation. In: ARPA image understanding workshop, Monterey, pp 957–966
 14. Hartley RI, Sturm P (1997) Triangulation. *Comput Vis Image Underst* 68(2):146–157
 15. Hartley RI, Zisserman A (2004) Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, Cambridge
 16. Hartley RI, Gupta R, Chang T (1992) Stereo from uncalibrated cameras. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Champaign, pp 761–764
 17. Huber PJ (1981) Robust statistics. Wiley, New York
 18. Jacobs DW (2001) Linear fitting with missing data for structure-from-motion. *Comput Vis Image Underst* 82: 57–81
 19. Levenberg K (1944) A method for the solution of certain non-linear problems in least squares. *Q Appl Math* 2:164–168
 20. Longuet-Higgins HC (1981) A computer algorithm for reconstructing a scene from two projections. *Nature* 293:133–135
 21. Mahamud S, Hebert M (2000) Iterative projective reconstruction from multiple views. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Hilton Head, pp II–430–437
 22. Mahamud S, Hebert M, Omori Y, Ponce J (2001) Provably-convergent iterative methods for projective structure from motion. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Kauai, pp 1018–1025
 23. Marquardt DW (1963) An algorithm for least-squares estimation of nonlinear parameters. *J Soc Ind Appl Math* 11:431–441
 24. Oliensis J, Hartley R (2007) Iterative extensions of the Sturm/Triggs algorithm: convergence and nonconvergence. *IEEE Trans Pattern Anal Mach Intell* 29(12): 2217–2233
 25. Pollefeys M, Van Gool L (1999) Stratified self-calibration with the modulus constraint. *IEEE Trans Pattern Anal Mach Intell* 21(8):707–724
 26. Pollefeys M, Koch R, Van Gool L (1998) Self calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In: Proceedings of the 6th international conference on computer vision, Bombay, India, pp 90–96
 27. Quan L (1995) Invariants of six points and projective reconstruction from three uncalibrated images. *IEEE Trans Pattern Anal Mach Intell* 17:34–46
 28. Shum HY, Ikeuchi K, Reddy R (1995) Principal component analysis with missing data and its application to polyhedral object modeling. *IEEE Trans Pattern Anal Mach Intell* 17(9):854–867
 29. Sturm P, Triggs W (1996) A factorization based algorithm for multi-image projective structure and motion. In: Proceedings of the 4th European conference on computer vision (ECCV), Cambridge, pp 709–720
 30. Tomasi C, Kanade T (1992) Shape and motion from image streams under orthography: a factorization approach. *Int J Comput Vis* 9(2):137–154
 31. Triggs W, McLauchlan PF, Hartley RI, Fitzgibbon A (2000) Bundle adjustment for structure from motion. In: Triggs B, Zisserman A, Szeliski R (eds) Vision algorithms: theory and practice. Springer, Berlin, pp 298–372

Projective Structure and Motion

► Projective Reconstruction

Prototype-Based Methods for Human Movement Modeling

Zhe Lin¹, Zhuolin Jiang² and Larry S. Davis³

¹Advanced Technology Labs, Adobe Systems Incorporated, San Jose, CA, USA

²Noah's Ark Lab, Huawei Tech. Investment Co., LTD., Shatin, Hong Kong, China

³Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, MD, USA

Synonyms

Action prototype trees

Definition

Human movement modeling is a static and dynamic human appearance representation with feature descriptors. Typical feature descriptors include local and holistic descriptors. Local descriptors mean sparse features extracted locally, i.e., local features [4] or local space-time features [6]; holistic descriptors means dense features extracted inside a human bounding region, i.e., shape/appearance descriptors [9], and motion descriptors [1, 3].

Prototype-based methods [2, 7, 13, 14] are a category of approaches representing original feature descriptors with a finite set of indices through feature quantization. Given a large number of descriptors extracted from training images or videos, a vector quantization (or data clustering) algorithm is used to divide the feature space into nonoverlapping cells where each cell is uniquely represented with an integer index. Given the quantization, each test feature can be mapped to an integer index (corresponding to the center of a quantization cell or cluster center) based on exact (or approximate) nearest neighbor search, and finally, recognition can be performed through simple index matching. This scheme is significantly more efficient than the direct descriptor matching schemes due to the speedup in calculating Euclidean distances in high-dimensional feature spaces.

Background

There are two categories of methods to model human appearances and movements. The first category is local feature-based approaches. In this category, usually a set of local features are extracted across all image regions or video frames, and recognition is based on matching of those features either via histogram comparisons or class voting. For example, a bag-of-features representation is computed and used as the descriptors [11, 12], or a Hough voting-based approaches [15] are used to simultaneously locate humans and recognize their movements.

The second category is holistic feature-based approaches. These approaches typically extract a single high-dimensional feature descriptor from a hypothesized bounding region and classifies it with pre-trained classifier. Typically, a sliding window approach is used over all possible sub-rectangles

across images [9] or across all space-time volumes [11, 14], and for every hypothetical bounding region, a holistic descriptor (consisting of appearance descriptor [9] and/or motion descriptor [1]) is extracted and classified into human/nonhuman (for human detection) or action IDs (for action recognition) based on a discriminative classifier trained off-line. The descriptors extracted from hypothetical human bounding region are invariant to human translation and scale variations and typically contain more information due to dense extraction; consequently, those features are much more reliable for recognition. In case of human movement understanding or activity recognition, typically, a generic human detector [9] is used to locate most probable human bounding boxes, and holistic descriptors extracted from all the frames are used for recognition.

Once feature descriptors are extracted from human images or videos, they will be fed into a classifier for recognition and categorization purposes. The classification modules are mostly based on machine learning or pattern recognition techniques as in the object recognition literature. Classifiers commonly used include NN/k-NN classifiers [2, 7], Support Vector Machine (SVM) classifiers [6], boosting-based classifiers [3, 5], dynamic time warping [13] for sequence alignment and matching, etc.

Direct descriptor matching and classification-based schemes have been common for human movement recognition. However, for large-scale action recognition problems, such a matching scheme may require tremendous amount of time for computing similarities between descriptor sequences and learning discriminative classifiers. Also, sequence matching (action recognition) involves frame alignment issues. In this regard, an efficient and effective descriptor matching scheme is needed to handle both scalability to the number of training classes and training examples and ability to handle sequence alignment.

Theory

Prototype-based approaches have been very effective in handling scalability to large training data. These approaches typically represent a human action as a sequence of basic action units [2, 7, 13, 14]. Each action frame or frame segment (multiple consecutive frames) is represented as one of the holistic action units

trained off-line or histogram of local action units (similar to bag of visual words). The action units are usually obtained by feature quantization such as k -means clustering in the feature space. The centers of quantization cells are defined as action units, and all test descriptors belong to a cell are assigned the same index (i.e., the index of the cell).

In [2], an action is represented as a set of pose primitives and n -Gram models are used for action matching. Weinland and Boyer [7] models an action as a set of minimum distances from exemplars to action frames in an exemplar-based embedding space. These action representation methods are compact and efficient but might be limited in capturing global temporal consistency between actions because they either use low-order statistics such as histograms and n -Grams or use a minimum-distance-based representation which does not enforce temporal ordering.

Relaxing temporal constraints [2] makes action representation more invariant to intra-class variation and, consequently, might be effective in recognizing small numbers of actions, but when the number of action classes is large, global temporal consistency is very important for action recognition due to small inter-class variability (i.e., increased ambiguity between actions). In fact, there have been approaches modeling the global temporal consistency.

Prototype Tree-Based Method

Recently, tree-based methods have been popular for human detection and activity recognition problems. Mikolajczyk and Uemura [8] proposes a random forest method on large number local features for action recognition. Although this method is efficient due to fast nearest neighbor search and can handle action detection problems in difficult cases, action frame time alignment issue is not explicitly addressed. Lin et al. [13] learns an action prototype tree based on hierarchical k-means clustering [10] and aligns action sequences with a fast dynamic time warping algorithm in order to compute accurate similarities between action sequences. More specifically, the cluster centers of leaf nodes (of the tree) are defined as the set of action prototypes. Distances between action prototypes are precomputed and stored in a lookup table, so the sequence alignment and matching stage is very efficient. The action prototype tree-based method learns action prototypes in joint shape and motion

space so that human movements can be described more accurately and compactly. This prototype tree model is also applied to sliding window-based framework for simultaneous action detection and recognition in [14]. The prototype tree-based motion modeling methods can also be combined with a Markov model to incorporate prototype transition priors between frames.

Application

Appearance prototype-based methods generally can be applied to all the human-related recognition problems in computer vision including appearance-based human identification, human detection, and human action detection and recognition. More broadly, they can be applied to video surveillance, human-computer interaction, virtual reality, and multimedia understanding.

Open Problems

Human movement modeling still remains challenging due to articulated nature of human bodies and varying camera viewpoints. Although there has been a significant progress on view-dependent appearance and movement modeling, view-invariant human movement modeling is still an open research topic. There has been several efforts to introduce view-invariant descriptors for human movements, but still more research is to be done to be practical in real applications.

References

1. Efros AA, Berg AC, Mori G, Malik J (2003) Recognizing action at a distance. In: Proceedings of the international conference on computer vision (ICCV), Nice, pp 726–733
2. Thrun C, Hlavac V (2008) Pose primitive based human action recognition in videos or still images. In: Proceedings of the computer vision and pattern recognition (CVPR), Anchorage
3. Fathi A, Mori G (2008) Action recognition by learning mid-level motion features. In: Proceedings of the computer vision and pattern recognition (CVPR), Anchorage
4. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60(2):91–110
5. Laptev I, Perez P (2007) Retrieving actions in movies. In: Proceedings of the international conference on computer vision (ICCV), Rio de Janeiro
6. Dollar P, Rabaud V, Cottrell G, Belongie S (2005) Behavior recognition via sparse spatio-temporal features. In: Proceedings of the VS-PETS, Beijing, pp 65–72

7. Weinland D, Boyer E (2008) Action Recognition Using Exemplar-based Embedding. In: Proceedings of computer vision and pattern recognition (CVPR), Anchorage
8. Mikolajczyk K, Uemura H (2008) Action Recognition with motion-appearance vocabulary forest. In: Proceedings of the computer vision and pattern recognition (CVPR), Anchorage
9. Dalal N, Triggs B (2005) Histograms of oriented gradients for Human detection. In: Proceedings of the computer vision and pattern recognition (CVPR), San Diego, pp 886–893
10. Nister D, Stewenius H (2006) Scalable recognition with a vocabulary tree. In: Proceedings of the computer vision and pattern recognition (CVPR), New York, NY, pp 2161–2168
11. Yuan J, Liu Z, Wu Y (2009) Discriminative Subvolume Search for Efficient Action Detection. In: Proceedings of the computer vision and pattern recognition (CVPR), Miami
12. Liu J, Luo J, Shah M (2009) Recognizing realistic actions from videos in the wild. In: Proceedings of the computer vision and pattern recognition (CVPR), Miami
13. Lin Z, Jiang Z, Davis LS (2009) Recognizing actions by shape-motion prototype trees. In: Proceedings of the international conference on computer vision (ICCV), Kyoto
14. Jiang Z, Lin Z, Davis LS (2010) A tree-based approach to integrated action localization, recognition and segmentation. In: Proceedings of the 3rd workshop on Human motion, Crete, Greece
15. Yao A, Gall J, Gool LV (2010) A hough transform-based voting framework for action recognition. In: Proceedings of the computer vision and pattern recognition (CVPR), San Francisco

PSF Estimation

- [Blind Deconvolution](#)
-

PTZ Camera Calibration

- [Active Calibration](#)
-

Pulling a Matte

- [Matte Extraction](#)
-

Purposive Vision

- [Animat Vision](#)
-

R

Radiance

Fabian Langguth and Michael Goesele
GCC - Graphics, Capture and Massively Parallel Computing, TU Darmstadt, Darmstadt, Germany

Related Concepts

► Irradiance

Definition

Radiance L is defined as the power received or emitted per unit solid angle $d\omega$ and per unit projected area $dA \cos \theta$ at a point x in direction (θ, ϕ) :

$$L(x, \theta, \phi) = \frac{d^2\Phi}{d\omega dA \cos \theta}.$$

Φ is the radiant power, which describes the total amount of energy that flows through a surface per time interval [1]. θ is the angle between the surface normal of the differential area dA and the direction under consideration (see Fig. 1). The unit of radiance is watt per steradian per square meter ($\text{W} \cdot \text{sr}^{-1} \cdot \text{m}^{-2}$).

Background

Radiance is a concept from radiometry, the science of measuring radiant energy transfer. It is an essential concept for global illumination algorithms [1].

In computer vision, it is mostly used in the context of light transport in scenes [2]. The equivalent concept in photometry is luminance, with the key difference being that luminance is adjusted to account for the varying sensitivity of the human eye to different wavelengths of light.

Theory

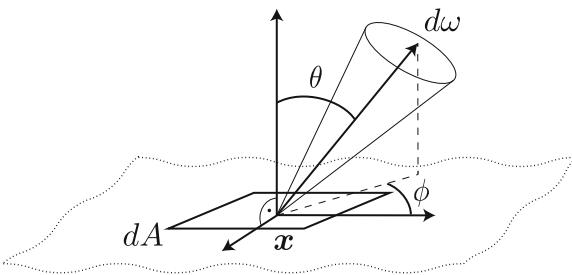
An important aspect of radiance is that it is constant along a line of sight in an empty medium. This follows from the conservation of energy in a small bundle of light rays between two differential surface patches. More formally, given two points x and y at distance $r = |x - y|$ and two small areas dx and dy located at x and y , respectively, the radiance leaving dx towards dy is written as

$$L(x \rightarrow y) = \frac{d^2\Phi}{d\omega_{x \leftarrow dy} \cos \theta_x dx}. \quad (1)$$

$d\omega_{x \leftarrow dy}$ denotes the solid angle covered by dy as seen from x . Similarly the radiance arriving at dy from dx is written as

$$L(y \leftarrow x) = \frac{d^2\Phi}{d\omega_{y \leftarrow dx} \cos \theta_y dy}. \quad (2)$$

Considering a total vacuum and no additional source of power, all energy in the system must be conserved. This means that all energy leaving from dx towards dy must arrive at dy . Therefore $d^2\Phi$ is equal in both equations, and we can derive



Radiance, Fig. 1 Geometric setting

$$L(x \rightarrow y) = \frac{d^2\Phi}{d\omega_{x \leftarrow dy} \cos \theta_x dx} \quad (3)$$

$$= \frac{d^2\Phi}{\frac{\cos \theta_y dy}{r^2} \cos \theta_x dx} = \frac{d^2\Phi}{\frac{\cos \theta_x dx}{r^2} \cos \theta_y dy} \quad (4)$$

$$= \frac{d^2\Phi}{d\omega_{y \leftarrow dx} \cos \theta_y dy} = L(y \leftarrow x). \quad (5)$$

Application

An application of the fact that radiance is constant along a ray can be seen in photography. The correct exposure is hereby determined by measuring the radiance (or more precisely the luminance) of a scene or part of a scene that should be correctly exposed. Once exposure is determined, the scene can be photographed from an arbitrary distance without changing the exposure settings of the camera.

References

1. Dutré P, Bala K, Bekaert P (2006) Advanced global illumination. AK Peters, Wellesley
2. Seitz SM, Matsushita Y, Kutulakos KN (2005) A theory of inverse light transport. In: Proceedings of the 10th IEEE international conference on computer vision (ICCV 2005), Beijing

Radiometric Calibration

Yasuyuki Matsushita
Microsoft Research, Beijing, China

Synonyms

Radiometric camera calibration

Related Concepts

► Radiance

Definition

Radiometric calibration is a process of determining radiometric response functions, which relate sensor irradiance with measured intensity values.

Background

Many computer vision algorithms rely on the assumption that image intensities are linearly related to the image irradiance recorded at the camera sensor. Since most cameras nonlinearly alter irradiance values for purposes such as dynamic range compression, this assumption generally does not hold. It is therefore important to calibrate the response function of a camera, so that the nonlinear mapping can be inverted and subsequent algorithms can assume linearity of intensity observations.

Radiometric calibration aims to estimate the response function f of a camera. The radiometric response function f maps the irradiance I that is captured at the sensor to the image intensity M that is read from the camera:

$$M = f(I). \quad (1)$$

For vision algorithms that require irradiance values I rather than measured intensity M as input, the inverse response function $g = f^{-1}$ needs to be determined so that measured intensities can be made linear with respect to irradiances. Since response functions f are monotonic and continuous, they are invertible to uniquely determine inverse response functions g .

Theory

Radiometric calibration methods require means to collect samples of the radiometric response function with some known relationship. One traditional approach is to use a special target, such as a Macbeth color chart [1], which has color patches with known reflectances. By uniformly illuminating the target, known radiances from the color patches are recorded.

The radiometric response function is then obtained by relating the sensor irradiance I with the recorded intensity values M . Nayar and Mitsunaga [13] use an optical filter with spatially varying transmittance; the variation corresponds to the radiance ratio.

To avoid using such special equipment, some methods use a set of images of a static scene from a fixed viewpoint, taken with different exposure times, so the radiance ratio is known. Known exposure times provide information about sensor irradiance ratios, which are the ratios of exposure times. In a similar manner with the approach of using a special target, by relating the sensor irradiance I with the measured intensities M , a radiometric response function is estimated. The early work of Mann and Picard [10] uses a gamma correcting function to represent response functions. With known exposure ratios, their method can successfully recover the inverse response function in the parametric form. With only approximate knowledge of relative exposure levels, Mitsunaga and Nayar [12] iteratively solve for a response function based on the assumption that it has a polynomial form. Other iterative estimation methods include that of Tsin et al [18], which estimates nonparametric responses using a statistical model of the CCD imaging process, and that of Pal et al [14], which utilizes probabilistic imaging models and prior models of response functions to compute response functions that can differ from image to image. Debevec and Malik [2] assumed a smoothness property of the response functions and estimated them in a non-parametric manner. As pointed out in [3, 5, 8], without the knowledge of exposure ratios, the estimate still has an *exponential ambiguity*. While not unique, such an estimate is still useful for many applications, such as radiometric alignment, high-dynamic range image production, and image stitching.

Several methods have been developed that use multiple exposures but do not require precise registration. Grossberg and Nayar [3] use the relationship between the intensity histograms of two scenes imaged with different exposures, because intensity histograms are relatively unaffected by small changes in the scene. Kim and Pollefeys [5] compute point correspondences between images. Mann [9] estimates response functions from a rotating and zooming camera.

Instead of using varying exposure times, some approaches use statistical or physical properties embedded in images to achieve radiometric calibration. Tsin et al's method [18] estimates nonparametric response functions using a statistical model of the

CCD imaging process. Pal et al [14] used probabilistic imaging models and weak prior models for deriving response functions to produce high-quality high dynamic range images. Matsushita and Lin [11] proposed to use the symmetric property of image noise by observing noise distributions contained in images. Takamatsu et al [16, 17] improved the noise-based method with a probabilistic intensity similarity measure, which requires a fewer number of images. Lin et al [6] and Lin and Zhang [7] proposed a method that takes only a single image as input. Their method uses edges for obtaining color or gray-scale histogram distributions, and the optimal inverse response function is determined by transforming linear distributions. Their method uses a database of response functions (DoRF) compiled by Grossberg and Nayar [4]. In a similar manner, Wilburn et al [19] use temporal color mixtures to directly sample the response function by observing motion blur in an image. More recently, Shi et al [15] show a calibration method from images taken under varying lighting conditions. In their approach, an inverse response function is determined by linearizing color profiles that are defined as a set of measured RGB values at a pixel across images.

Open Problem

In many practical situations, the input dataset is naturally restricted by the camera's capability or application scenarios. For example, it is difficult to obtain multiple images at different exposures with ordinary web cameras. A more general and robust approach for radiometric calibration is still to be investigated.

References

- Chang Y-C, Reid JF (1996) Rgb calibration for color image analysis in machine vision. *IEEE Trans Image Process* 5(10):1414–1422
- Debevec P, Malik J (1997) Recovering high dynamic range radiance maps from photographs. In: Proceedings of ACM SIGGRAPH. Association for Computing Machinery, New York, pp 369–378
- Grossberg M, Nayar S (2003) Determining the camera response from images: what is knowable? *IEEE Trans Pattern Anal Mach Intell* 25(11):1455–1467
- Grossberg M, Nayar S (2003) What is the space of camera response functions? In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos, pp 602–609
- Kim S-J, Pollefeys M (2004) Radiometric alignment of image sequences. In: Proceedings of computer vision and

- pattern recognition (CVPR). IEEE Computer Society, Los Alamitos, pp 645–651
6. Lin S, Gu J, Yamazaki S, Shum H-Y (2004) Radiometric calibration from a single image. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos, pp 938–945
 7. Lin S, Zhang L (2005) Determining the radiometric response function from a single grayscale image. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos, pp 66–73
 8. Litvinov A, Schechner YY (2005) Addressing radiometric nonidealities: a unified framework. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos, pp 52–59
 9. Mann S (2001) Comparametric imaging: estimating both the unknown response and the unknown set of exposures in a plurality of differently exposed images. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos, pp 842–849
 10. Mann S, Picard RW (1995) On being ‘undigital’ with digital cameras: extending dynamic range by combining differently exposed pictures. In: Proceedings of the IS & T, 48th annual conference. The Society for Imaging Science and Technology, Springfield, pp 422–428
 11. Matsushita Y, Lin S (2007) Radiometric calibration from noise distributions. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Piscataway, pp 1–8
 12. Mitsunaga T, Nayar S (1999) Radiometric self-calibration. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos, pp 374–380
 13. Nayar S, Mitsunaga T (2000) High dynamic range imaging: spatially varying pixel exposures. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos, pp 472–479
 14. Pal C, Szeliski R, Uyttendaele M, Jojic N (2004) Probability models for high dynamic range imaging. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos, pp 173–180
 15. Shi B, Matsushita Y, Wei Y, Xu C, Tan P (2010) Self-calibrating photometric stereo. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE, New York, pp 1118–1125
 16. Takamatsu J, Matsushita Y, Ikeuchi K (2008) Estimating camera response functions using probabilistic intensity similarity. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Piscataway, pp 1–8
 17. Takamatsu J, Matsushita Y, Ikeuchi K (2008) Estimating radiometric response functions from image noise variance. In: Proceedings of European conference on computer vision (ECCV). Springer, Berlin, pp 623–637
 18. Tsin Y, Ramesh V, Kanade T (2001) Statistical calibration of ccd imaging process. In: Proceedings of international conference on computer vision (ICCV). IEEE Computer Society, Los Alamitos, pp 480–487
 19. Wilburn B, Xu H, Matsushita Y (2008) Radiometric calibration using temporal irradiance mixtures. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Piscataway

Radiometric Camera Calibration

► [Radiometric Calibration](#)

Radiometric Response Function

Yasuyuki Matsushita

Microsoft Research, Beijing, China

Synonyms

Camera response function

Related Concepts

► [Radiance](#); ► [Radiometric Calibration](#); ► [Vignetting](#)

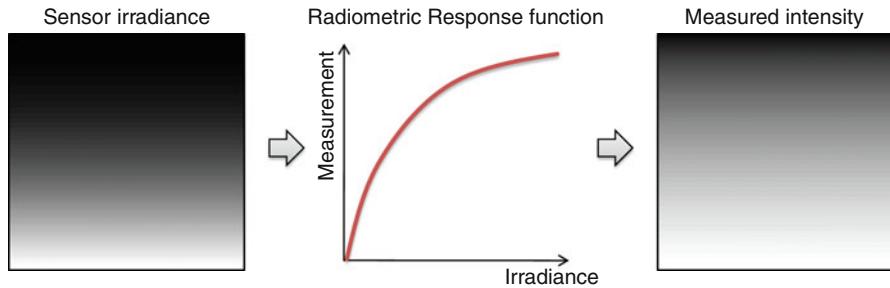
Definition

Radiometric response function is a function that transforms sensor irradiance into measured intensities that are the output from the camera.

Background

In most cameras, there exists a radiometric response function that relates sensor irradiance to measured intensity values. The radiometric response functions are typically nonlinear. This nonlinearity is intentionally designed by camera manufacturers for purposes such as compressing the dynamic range of scene brightness or to take into account the nonlinear mapping of display systems.

While many computer vision algorithms assume a linear (or affine) relationship between the sensor irradiance and the measured image intensity, the radiometric response functions are typically unknown, and these vary with camera parameter settings. Therefore, it is important to estimate the response function to linearize the measured image intensity values for vision algorithms to work. The process of determining radiometric response functions is called *radiometric*



Radiometric Response Function, Fig. 1 A radiometric response function relates the incoming sensor irradiance to measured intensity values

calibration. Once the radiometric response function is determined, the measured intensity can be linearized and transformed into sensor irradiance with a scaling ambiguity.

Theory

The radiometric response function f maps irradiance I that is captured at the sensor to the image intensity M by

$$M = f(I). \quad (1)$$

For computer vision algorithms that require irradiance values I rather than measured intensity M as input, the inverse response function $g = f^{-1}$ needs to be determined so that measured intensities can be made linear with respect to irradiances. Since response functions f are continuous and monotonic, they are invertible to determine inverse response functions (Fig. 1).

Representation

Many parametrization methods have been used to represent radiometric response functions f . To deal with the scale difference between irradiance I and measured intensity M , both are normalized in the range of $[0, 1]$ so that $f(0) = 0$ and $f(1) = 1$.

When representing a radiometric response function in a certain form, there is a trade-off between complexity and flexibility. A simpler representation makes the estimation problem more tractable at the cost of approximation accuracy. On the other hand, a more flexible representation requires an uneasy solution method. Major representations of radiometric response functions are reviewed here.

Mann and Picard [6] represent the response functions in a form of a gamma correction function as

$$M = f(I) = \alpha + \beta I^\gamma, \quad (2)$$

where α and β are offset and scale factors, and γ is the power-law parameter.

Mitsunaga and Nayar use a high-order (order of N) polynomial function as the model of inverse response functions g as

$$I = g(M) = \sum_{n=0}^N c_n M^n, \quad (3)$$

where c_n are the coefficients of the polynomial function.

Grossberg and Nayar apply principal component analysis (PCA) to a database of real-world response functions (DoRF) and show that the space of response functions can be represented by a small number of basis functions [3].

$$g = \bar{g} + \sum_{n=1}^N c_n g_n. \quad (4)$$

In the above equation, \bar{g} is the mean inverse response, and g_i is the i -th principal component of the inverse response functions.

Debevec and Malik [1] use a nonparametric form of the radiometric response functions. The nonparametric representation has great descriptive power, but it is necessary to estimate $f(I)$ (or $g(M)$) for each intensity level (e.g., 256 for 8-bit images). Therefore, the solution methods tend to become more complex. The same representation is also used by Tsin et al [11].

Existing methods can estimate radiometric response functions from a set of images taken with different known exposure times from a fixed viewpoint [1, 2, 8]. More recent methods use different cues to achieve the estimation in more general settings. These include methods using single-image edges [4, 5], image noise observations [7, 9, 10], and motion blur [12]. These estimation methods are detailed in the entry “► [Radiometric Calibration](#)”.

Application

The estimation of radiometric response functions constitutes an extensively researched area due to its fundamental importance for many computer vision algorithms, such as shape-from-shading, photometric stereo, high-dynamic-range imaging, and photo stitching. Estimated radiometric response functions are used to linearize the measured intensity values as preprocessing for these computer vision algorithms that require a linear (or affine) relationship with the irradiance.

References

1. Debevec PE, Malik J (1997) Recovering high dynamic range radiance maps from photographs. In: Proceedings of ACM SIGGRAPH. ACM SIGGRAPH, New York, pp 369–378
2. Grossberg MD, Nayar SK (2003) Determining the camera response from images: what is knowable? *IEEE Trans Pattern Anal Mach Intell (PAMI)* 25(11):1455–1467
3. Grossberg MD, Nayar SK (2004) Modeling the space of camera response functions. *IEEE Trans Pattern Anal Mach Intell (PAMI)* 26(10):1272–1282
4. Lin S, Gu J, Yamazaki S, Shum H-Y (2004) Radiometric calibration from a single image. In: Proceedings of computer vision and pattern recognition (CVPR), vol 2. IEEE Computer Society, Los Alamitos, pp 938–945
5. Lin S, Zhang L (2005) Determining the radiometric response function from a single grayscale image. In: Proceedings of computer vision and pattern recognition (CVPR), vol 2. IEEE Computer Society, Los Alamitos, pp 66–73
6. Mann S, Picard RW (1995) Being eudigitalf with digital cameras: Extending dynamic range by combining differently exposed pictures. In: Proceedings of IS & T 48th annual conference. The Society for Imaging Science and Technology, Springfield, pp 422–428
7. Matsushita Y, Lin S (2007) Radiometric calibration from noise distributions. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos
8. Mitsunaga T, Nayar SK (1999) Radiometric self-calibration. In: Proceedings of computer vision and pattern recognition (CVPR), vol 2. IEEE Computer Society, Los Alamitos, pp 374–380
9. Takamatsu J, Matsushita Y, Ikeuchi K (2008) Estimating camera response functions using probabilistic intensity similarity. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos
10. Takamatsu J, Matsushita Y, Ikeuchi K (2008) Estimating radiometric response functions from image noise variance. In: Proceedings of European conference on computer vision (ECCV). Springer, New York, pp 623–637
11. Tsin Y, Ramesh V, Kanade T (2001) Statistical calibration of ccd imaging process. In: Proceedings of international conference on computer vision (ICCV), vol 1. IEEE Computer Society, Los Alamitos, pp 480–487
12. Wilburn B, Xu H, Matsushita Y (2008) Radiometric calibration using temporal irradiance mixtures. In: Proceedings of computer vision and pattern recognition (CVPR). IEEE Computer Society, Los Alamitos

Rao Metric

► [Fisher-Rao Metric](#)

Rationale for Computational Vision

Ruzena Bajcsy

Department of Electrical and Computer Sciences,
College of Engineering University of California,
Berkeley, CA, USA

Synonyms

[Computer vision](#)

Definition

Vision is a scientific field that investigates biological systems and machines how to use light to gain information about their environments. It covers several sub-fields such as optics, perception, psychophysics, neurophysiology, information science, signal processing, cognitive science, and related subjects.

Background

Background section will not be a review of all the contributions to the field of vision from all the subfields mentioned above. Rather it shall concentrate on how different subfields try to solve the problem vision and

how this field has evolved over time due to better understanding of the problems but also due to more powerful technological tools.

Fundamentally vision utilizes the spatial and temporal information (structure) that stems from the reflection of light from the environment. The focus will be on the computational aspects of processing of this visual information, asking how visual information is represented for recognition, mobility, and manipulation, hence computational vision.

Much of the machine vision has been motivated by various applications: military medical, industrial, cultural, and commercial. The application-driven solution can be a separate article by itself. In this article the emphasis will be more on the scientific rational for computational vision than all the other reasons.

Motivation for this scientific endeavor comes from two very different sources:

1. How to model (mathematically and/or algorithmically) the biological process of vision. Can one design vision-based processes that explain visual perceptual phenomena?
2. How to design machines that will produce the desired outcome (recognition or navigation of a robot) from visual data.

The goals and the evaluation of success of these two approaches are very different.

Computational models of biological systems and machine vision is a very broad subject because it entails mammals as well as humans. It also encompasses fields such as computational neuroscience, through psychophysics up to computational cognitive modeling. In this article only some representative works will be mentioned which in no way does justice to this field. It covers modeling at different scale from neurons with prominent work of Hodgkin–Huxley model [1] and Hubel and Wiesel [2], through sensory processing and communication of Barlow [3]. Psychologists have seriously considered computational models, especially how to explain visual illusions (exemplars are Gregory [4] and Frisby [5]), mechanisms of binocular vision (see Julesz [6]), and theory of color vision in retinex theory (by Land [7]).

Coincidentally, the engineering community also was heavily engaged in developing computational models for processing data coming from cameras. The limitations of poor resolution of cameras (64 by 64 pixels), limited memory, and compute power limited the applications of this era. Hence the topics were primarily two-dimensional picture recognition,

such as digit recognition, finger print recognition, and in medical domain chromosome recognition. The methodologies used were statistical pattern recognition represented by Duda and Hart [8], signal-image processing by Rosenfeld and Kak [9], syntactic pattern recognition by K.S. Fu [10], and many others.

In early 1970s another trend emerged from the newly established Artificial Intelligence Laboratories at MIT and Stanford Universities. This trend emphasized the need for recognizing three-dimensional (3D) objects and scenes, arguing that today beings live in the 3D world; therefore the visual objects need to be represented as such. Guzman [11] was the first one who dealt with this problem using principles of projective geometry. Binford and Agin [12] and his students proposed a generalized cylindrical model (skeleton and corresponding cross-sections) as a generic model of 3D objects. It was argued that the model can represent parts and in turn the part-whole relationships can be represented as graphs.

This was more or less object-centered representation as oppose to the viewer-centered representation, proposed by Koenderink and vanDoorn [13] as aspect graphs. This debate object-centered representation vs. viewer-centered representation is still ongoing!

In meantime David Marr [14] in late 1970s and early 1980s questioned the scientific approach of engineering community as ad hoc and not anchored in scientific theory. He argued that the validation of the engineering approaches must be guided by what is known from neuroscience and psychophysics about human visual processing (this at present is the best model).

He outlined three levels of analysis:

1. The computational theory, specifying the goals of the computation
2. Representation and algorithm, giving a representation of the input and output which transforms one into the other
3. The hardware implementation, how algorithm and representation may be physically realized

Concurrently with Marr's effort in the UK, Christopher Longuet-Higgins [15] introduced cognitive science as the interdisciplinary study of how information is represented and transformed in the brain. It spans many levels of analysis, from low-level learning and decision mechanisms to high-level logic and planning. This field has blossomed with many works of modeling of perceptual and cognitive processes (see Newell and Simon [16], Hinton [17], and many others).

The 1980s benefitted from rapid advancements in hardware both in better cameras and computing power. This development enabled to perform some real-time computation and connect signal processing and perception with action. Motivated by Gibson [18] and Bajcsy [19] proposed a new research paradigm, called Active Perception. In this new framework, it has been shown that there is benefit in controlling the data acquisition and serves as way of modeling focus of attention.

Simultaneously progress has been made in various basic algorithms such as stereo reconstruction, motion detection and interpretation, multiple view reconstruction, shape from shading and photometric stereo, and shape–object–scene representation. Several textbooks cover these advancements, just to cite a few: Faugeras [20], Forsyth and Ponce [21], and Ma et al. [22]:

- Another challenge for machine perception comes from Gestalt psychology founded by Max Wertheimer [23], in the beginning of the twentieth century. The word Gestalt means a unified or meaningful whole. The laws of Gestalt theory of perceptual organization are as follows:
- The Law of Similarity
- The Law of Pragnanz
- The Law of Proximity
- The Law of Continuity
- The Law of Closure

Computationally however, they are hard to define and map into an algorithm since they deal with rather vague concepts such as proximity and similarity. From the beginning of computational vision, this problem has attracted attention, first just building some distance function (Zobrist and Thompson [24]) later looking at these laws as a guide to perceptual organization (Low [25]) and more recently as “image segmentation,” the partitioning of an image (or video streams) into sets of pixels that correspond to “objects” or parts of objects. This process is based on bottom-up cues such as similarity of pixel brightness, color, texture, and motion as well as top-down input derived from familiar object categories such as faces. Malik and his coworkers [26] aimed at developing a scientific understanding of grouping, both in the context of human perception and for computer vision, and have shown progress in this area but also its limitations.

At the onset of the twenty-first century, the abundance of images on Internet has spurred an interest in Machine Learning Technologies (MLT) as they apply to object recognition and classification. The latest premier vision conferences ICCV and ECCV have been representative examples of this research. The theoretical development related to this effort is compressed sensing; see Emmanuel Candès and Terence Tao [27]. They discovered important results on the minimum number of data needed to reconstruct an image even though the number data would be deemed insufficient by the Nyquist–Shannon criterion. Further effort in combining the compressed sensing with principal component pursuit is in [28].

In conclusion, the scientific community has seen serious progress in computational vision afforded by technological advances (cameras, computing power) but also by the community mastering much more sophisticated mathematical and computational tools than ever before. The emphasis on robustness, sharing code, and creating standard data bases where different approaches can be tested is very good.

Open problems

The basic representations of visual objects and their dynamics are still open problems. Further, the adaptive nature and flexibility of biological vision is still a dream to be accomplished by the computational vision community

References

1. Hodgkin A, Huxley A (1952) A quantitative description of membrane current and its application to conduction and excitation in nerve. *J Physiol* 117:500–544
2. Hubel DH, Wiesel TN (1962) Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *J Physiol (Lond)* 160:106–154
3. Barlow HB (1961) Possible principles underlying the transformation of sensory messages. In: Rosenblith WA (ed) *Sensory communication*. MIT, Cambridge, pp 217–234
4. Gregory R (1966, 1997) *Eye and brain: the psychology of seeing*, 1st edn. Weidenfeld and Nicolson, London (1966); 5th edn. Oxford University Press (1997)
5. Frisby JP (1979) *SEEING: illusion, brain and mind*. Oxford University Press, Oxford/New York
6. Julesz B (1971) Foundations of cyclopean perception. The University of Chicago Press, Chicago
7. Land EH (1977) The retinex theory of color vision. *Sci Am* 237:108–128

8. Duda R, Hart P, Stork D (2000) Pattern classification, 2nd edn. Wiley, New York
9. Rosenfeld A, Kak A (1982) Digital picture processing. Academic, New York
10. Fu KS (1981) Syntactic pattern recognition and applications. Prentice Hall, Upper saddle river, New Jersey
11. Guzman A (1968) Decomposition of a visual scene into three-dimensional bodies. Proceedings of AFIPS fall conference in Dec 1968, San Francisco. ACM, New York
12. Agin G, Binford T (1976) Representation and description of curved objects. IEEE Trans Comput 25(4):439–449
13. Koenderink JJ, van Doorn AJ (1979) The internal representation of solid shapes with respect to vision. Biol Cybern 32:211–216
14. Marr D (1982) Vision: a computational investigation into the human representation and processing of visual information. MIT, Cambridge
15. Longuet-Higgins HC (1979) The perception of music. Proc R Soc B 205:307–322
16. Newell A, Simon HA (1972) Human problem solving. Prentice Hall, Englewood Cliffs
17. Hinton GE, Sejnowski TJ (1983) Analyzing cooperative computation. In: Proceedings of the fifth annual conference of the Cognitive Science Society, Rochester
18. Gibson JJ (1987) The ecological approach to visual perception. Lawrence Erlbaum Associates, Hillsdale
19. Bajcsy R (1988) Active perception. Proc IEEE 76(8): 996–1006
20. Faugeras O (1993) Three-dimensional computer vision, a geometric viewpoint. MIT, Cambridge
21. Forsyth DA, Ponce J (2003) Computer vision, a modern approach. Prentice Hall, Upper Saddle River/London
22. Ma Y, Soatto S, Kosecka J, Sastry SS (2004) An invitation to 3-D vision; from images to geometric models. Springer, New York
23. Wertheimer M (2000) Laws of organization of perceptual forms. In: Yantis S (ed) Visual perception: essential readings. Key readings in cognition. Psychology Press, Philadelphia, pp 216–224
24. Zobrist AL, Thompson WB (1975) Building a distance function for Gestalt grouping. IEEE Trans Comput C-24(7): 718–728
25. Lowe DG (1985) Perceptual organization and visual recognition. Kluwer Academic, Boston
26. Malik J, Arbaleaz P, Maire M (2009) Projects: grouping and ecological statistics. http://www.eecs.berkeley.edu/Research/Projects/CS/vision/vision_group.html
27. Candès EJ, Tao T (2008) Reflections on compressed sensing. IEEE Inf Theory Soc News 58(4):20–23
28. Zhou Z, Wright J, Li X, Candès EJ, Ma Y (2010) Stable principal component pursuit. In: Proceedings of international symposium on information theory, Austin

Recognition-By-Components (RBC)

Theory

► Geons

Recovery of Reflectance Properties

Jason Lawrence

Department of Computer Science, School of Engineering and Applied Science University of Virginia, Charlottesville, VA, USA

Synonyms

[Appearance scanning](#); [BRDF measurement](#); [Reflectometry](#)

Definition

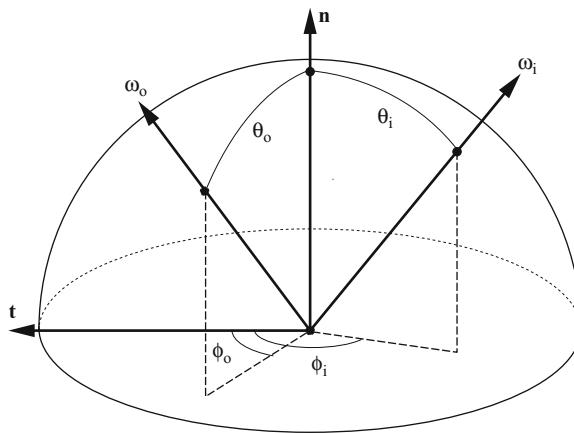
Recovering reflectance properties refers to the process of measuring and modeling the manner in which a material scatters incident light. This often involves estimating the parameters of a light scattering function based on measurements of a physical sample.

Background

Along with its 3D shape, another crucial aspect of an object's appearance is the manner in which it scatters incident light, often referred to generally as its *reflectance*. In the case of opaque objects, it may be assumed that any light which strikes the surface is either reflected back into the environment at that same incident location, possibly in different amounts along different directions, or absorbed by the material. Translucent objects require consideration of the way light scatters internally within the medium. Only metallic surfaces are technically opaque, although dielectrics may be treated as opaque at an appropriate measurement scale. Further, the reflectance of many objects varies spatially over their surface. This entry focuses on techniques for measuring and representing the optical properties of opaque surfaces.

Theory

The appearance of a homogeneous opaque surface is completely characterized by the Bidirectional Reflectance Distribution Function (BRDF) [1]. This is



Recovery of Reflectance Properties, Fig. 1 Geometry of the BRDF. The incident direction ω_i and exitant direction ω_o are defined with respect to the local surface frame defined by the surface normal \mathbf{n} and tangent \mathbf{t} . These directions are often written in terms of their respective elevation and azimuthal angles θ and ϕ

a scalar-valued function of the ratio of the differential radiance dI leaving the surface along direction ω_o with respect to the differential surface irradiance dE from light arriving along direction ω_i with wavelength λ

$$f_r(\omega_i, \omega_o, \lambda) = \frac{dI(\omega_o, \lambda)}{dE(\omega_i, \lambda)} \quad [\text{sr}^{-1}]. \quad (1)$$

This is a function of five variables (note that the directions are unit length and thus occupy a 2D domain) defined with respect to the local tangent frame attached to a location on a 2-manifold surface (Fig. 1). A very nice overview of the radiometry underlying the BRDF is available in a chapter written by Pat Hanrahan entitled “Rendering Concepts” in the textbook by Cohen and Wallace [2].

Note that Eq. (1) does not differentiate between the wavelength of the incident and exitant light and thus it cannot capture photoluminescence effects such as phosphorescence or fluorescence. Furthermore, the full dependence on wavelength is often ignored (as will be the case in this entry), in which case λ in Eq. (1) is omitted and the BRDF is a function of four variables defined within a trichromatic color space such as RGB. The dimensionality of the BRDF can be further reduced to three if the material’s reflectance is *isotropic*, meaning it is unaffected if the incident and

exitant directions are rotated together around the surface normal. This is in contrast to *anisotropic* materials which exhibit a visible *surface grain* that causes the reflectance to depend on this azimuthal component such as brushed metal, satin, silk, and velvet.

In many cases, the surface BRDF will vary from one surface location to the next (e.g., a wooden object with visible spatially varying grain densities). The spatially varying BRDF (svBRDF) is used to characterize the reflectance of these inhomogeneous surfaces, and it simply adds surface position \mathbf{x} to the angular variables in the BRDF: $S(\mathbf{x}, \omega_i, \omega_o)$. The svBRDF is thus a function of six variables since two numbers are required to specify a surface location.

Representations

A considerable amount of research has focused on developing efficient BRDF representations that apply to a wide range of materials. This includes a number of early and still widely popular phenomenological models such as the Phong [3] and related Blinn-Phong models [4], which offer reasonable approximations for plastics and smooth painted surfaces.

An alternative approach is to derive the BRDF from the laws of physics based on a hypothesized physical model of a material’s surface. These physically based models include the seminal Torrance-Sparrow [5] and Cook-Torrance models [6], both of which assume that the surface is composed of randomly oriented microscopic mirrors, or *microfacets*. These models capture important effects predicted by Fresnel equations including color shifts near highlights and an overall increased specular response near *grazing angles* (i.e., as ω_o and ω_i approach the horizon in opposite directions). The He-Torrance-Sillion-Greenberg (HTSG) model [7] was derived using a similar methodology, but considers wave-related effects such as diffraction and interference. The Oren-Nayar model [8] is also based on a microfacet theory, but one in which the individual microfacets are perfect Lambertian reflectors and are meant to reproduce the reflectance of matte materials such as dust or chalk. The Kajiya-Kay model [9] considers the way light is reflected from small fibers modeled as cylinders and is intended to model the appearance of hair and fur. In all of these cases, the resulting BRDF is an analytic expression with a handful of parameters that control the

magnitude, color, and shape of dominant lobes in the BRDF that are commonly either diffuse (largely nondirectional), specular (forward scattering), or retroreflective (backward scattering).

Another family of BRDFs was derived to fit empirical data. This includes the Lafourture [10] and Ward [11] models. These are both analytic models with parameters similar to those described above. The Lafourture model is defined as the sum of an arbitrary number of cosine lobes and can express a wide range of phenomena. The Ward BRDF is functionally more similar to the Cook-Torrance or Blinn-Phong model, but can produce elongated specular highlights that match a common form of anisotropic reflectance.

More recent research has considered nonparametric representations of the BRDF which are often constructed directly from measured data. These range from straightforward tabulated models [12] to compressed representations obtained by projecting the measurements into either a fixed basis defined over the double hemisphere (e.g., Zernike polynomials, spherical harmonics, wavelets, radial basis functions [13]) or a specialized basis estimated from the measured data itself. In the latter case, this can be achieved using common dimensionality reduction algorithms such as Principal Component Analysis (PCA) or cast as a matrix factorization problem [14, 15]. The accuracy of these representations is greatly affected by the way the BRDF is parameterized. A particularly useful parameterization was introduced by Rusinkiewicz [16] and is based on the *half-angle* and *difference angle* (inset at right). The half-angle is simply the bisector of the incident and exitant directions $\mathbf{h} = (\omega_i + \omega_o)/\|\omega_i + \omega_o\|$,

and the difference angle (θ_d, ϕ_d) is the incident direction expressed with respect to the half-angle. This parameterization has the desirable property of aligning common BRDF features to the transformed coordinate axes – including specular peaks [6], grazing-angle effects, and retroreflective peaks [8] – and as a result, only a relatively small amount of data is needed for an accurate representation.

Acquisition

The most basic requirements of a BRDF measurement system are a light source to illuminate the surface and a photodetector to record the amount of energy reflected in a particular direction. The difficulty of measuring the BRDF of a material lies in the high dimensionality of these functions and the calibration requirements of existing setups.

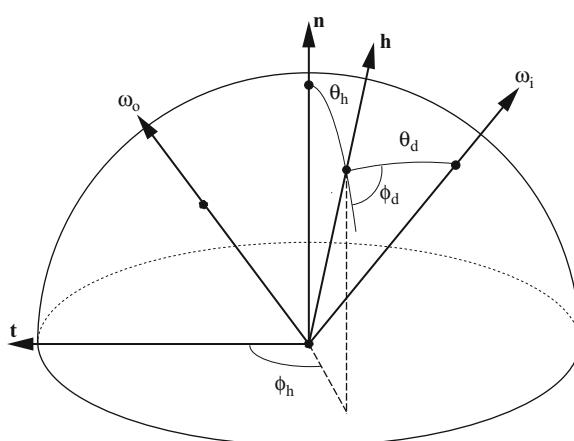
Gonioreflectometers

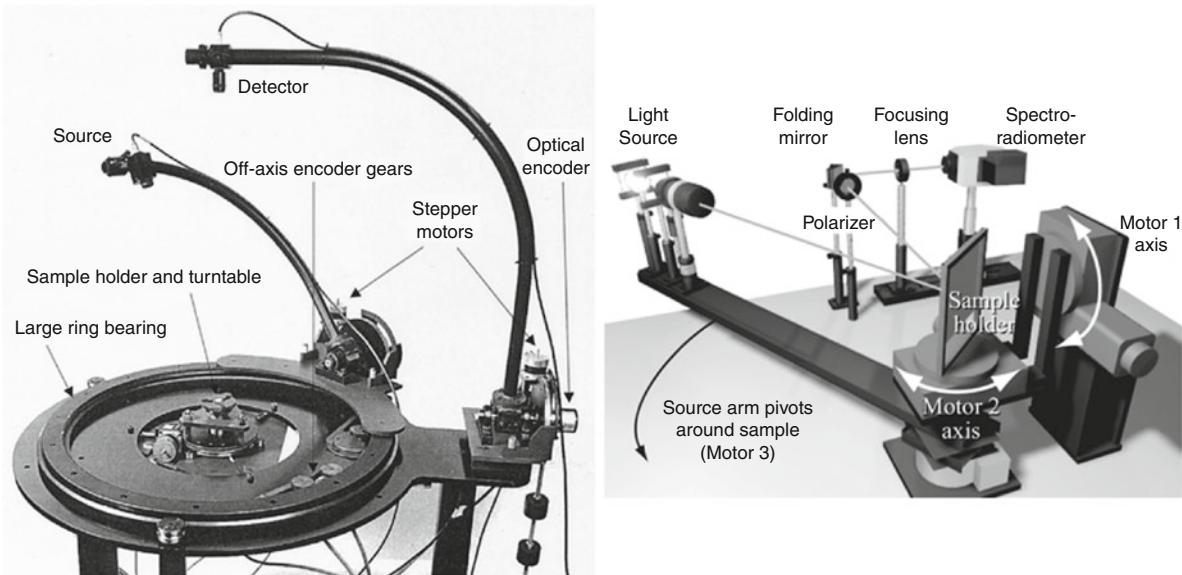
BRDFs were originally measured using a *gonioreflectometer*, a term that incorporates the Latin word for angle (*gonio*). These consist of a single photodetector and light source that can be moved to arbitrary locations on a hemisphere centered around a small planar sample (Fig. 2). Gonioreflectometers are only suitable for acquiring relatively sparse angular measurements since densely sampling the full BRDF domain would lead to infeasible acquisition times. However, they are very accurate and repeatable and are thus still used to validate newer designs and maintain reflectance standards.

In order to use the data recorded by a gonioreflectometer in a practical system, it is often necessary to estimate the regions of the BRDF that were not directly measured. This is often done by fitting the parameters of one of the analytic BRDF models described previously to the measured data. This process involves solving a nonlinear optimization problem with many variables and is often difficult and error prone in practice [19].

Camera-Based Systems

A seminal development was Greg Ward's use of a curved mirror and camera [11] (Fig. 3). Note that a single image recorded by the camera contains a densely sampled 2D slice of the BRDF at a fixed





Recovery of Reflectance Properties, Fig. 2 Two gonioreflectometer designs. *Left:* the apparatus developed by White et al. [17] has four degrees of freedom and achieves an angular resolution of approximately 0.1° and is accurate to within 0.3° . (Image reproduced from [17].) *Right:* the design of Li et al. [18]

incident direction. Recording multiple images at different light source positions allows probing the full 4D domain. This setup enables efficiently measuring a considerably larger portion of the BRDF domain than was previously possible with gonioreflectometers and led to the development of a new anisotropic BRDF model that is still in wide use today.

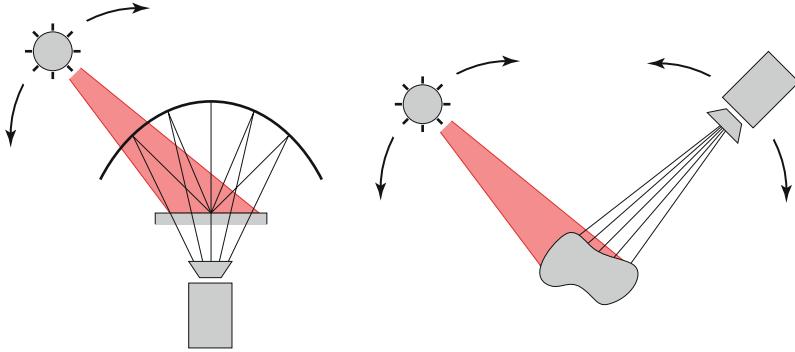
Ghosh et al. [20] describe a related setup that uses a digital projector and multiple curved mirrors to achieve a similar acquisition process, but one that does not require any moving parts, which significantly decreases acquisition time. Dana et al. [21] also explore using a mirrored dome and camera to accelerate BRDF capture. A key aspect of their design is that it allows translating the material sample within the measurement plane in order to sample the spatial dimensions of the svBRDF.

The camera-based BRDF measurement system introduced by Marschner et al. [22] was another significant step in the field, which traded Ward's curved mirror for a curved sample to achieve a similar effect (Fig. 3 right). A unique measurement of the object's BRDF is recorded at each pixel since each pixel observes a different orientation of the surface (recall that the BRDF is defined with respect to the *local*

has three degrees of freedom and is thus restricted to measuring isotropic material samples. Due to the use of a spectroradiometer, this system can measure the BRDF at 10 nm increments over the visible spectrum

surface frame). Assuming that the sample object's reflectance is homogeneous (i.e., no perceptible spatial variation), this provides a dense 2D slice of its BRDF, although the one that differs from the slices is acquired with Ward's setup. Matusik et al. [12] refined Marschner's design and measured the BRDFs of nearly one hundred isotropic spherical samples at unprecedented angular resolutions. Ngan et al. [19] extended this design further to allow measuring anisotropic samples that were formed into thin strips and wrapped around a cylinder.

Although camera-based systems can achieve a much higher angular resolution than traditional gonioreflectometers, the quality of the individual measurements is generally lower. This is due to the reduced quality of the individual photosensitive elements in a typical CCD or CMOS array, the need for more complex optical systems which can produce internal reflections, and the way wavelength is sampled using, for example, a Bayer filter [23]. However, this is beginning to change with the steady improvement in digital camera technology and the development of tunable narrowband color filters. Additionally, the high angular resolution of the data returned by these systems enables the use of non-parametric representations



Recovery of Reflectance Properties, Fig. 3 Two camera-based BRDF measurement systems. *Left:* Ward’s acquisition rig used a curved half-silvered mirror and a camera to enable

efficient acquisition of dense BRDF measurements. *Right:* Marschner’s setup exploited the same principle through the use of a curved specimen and camera

discussed previously. The advantage of these models is that they often provide a much more accurate fit to the measured data compared to an analytic model and require solving a linear optimization problem which is more stable.

In the aforementioned camera-based systems, the geometry of the target sample and its location with respect to the camera and light source must be known in order to properly interpret the recorded images as BRDF measurements. This calibration step is notoriously difficult and is often simplified by using samples with a specific known geometry: planar, spherical, or cylindrical. A recent research focus has been on developing systems for measuring the BRDF or svBRDF of samples with *arbitrary geometry*. On the one hand, this task is considerably harder since existing techniques for scanning geometry often rely on assumptions about the surface reflectance. On the other hand, all that is needed is the normal direction and, for anisotropic surfaces, the tangent direction at each image pixel as opposed to a complete 3D model.

One of the first general systems for measuring the reflectance of arbitrarily shaped spatially varying opaque isotropic objects was proposed by Lensch et al. [24]. Their approach has multiple steps. First, the geometry of the target object is acquired using either a penetrative method such as a computed tomography (CT) scanner or with a standard laser scanner after first coating the object in a diffuse powder. In a second step, the object is photographed from different viewpoints under variable point lighting (Fig. 3). Third, the 3D geometry in the first step is registered to the images in the second step using a silhouette-based alignment

algorithm. The extraction of BRDF data can proceed using the surface normal of the registered 3D model at each image pixel. Finally, they use the resulting BRDF data to estimate the parameters of a Lafontaine model [10] at each vertex in a triangle mesh of the surface. To make this fitting process more robust, they assume that the object’s reflectance can be accurately modeled with only a handful of unique *basis BRDFs* and corresponding spatial blending weights. Therefore, as opposed to storing a unique set of Lafontaine parameters at each vertex, only a set of nonnegative barycentric coordinates (unity of partition) over the set of basis BRDFs is computed instead. Computing these basis BRDFs and blending weights is cast as a clustering problem.

A related system was proposed by Goldman et al. [25]. The key difference is that they *jointly* estimate the surface normal at each pixel in a fixed camera along with the coefficients of the Ward BRDF model by solving a large non-linear optimization problem. On the one hand, this avoids having to scan the geometry in a separate step and subsequently register this geometry to the set of images, significantly simplifying the experimental setup and processing. On the other hand, this approach recovers only the normals and BRDFs for the portion of the object visible from the chosen viewpoint. This approach can be regarded as a generalization of Lambertian photometric stereo [26] since the surface BRDFs are drawn from a much larger space (i.e., the space of BRDFs expressible by the Ward model). Goldman et al. follow a similar strategy as Lensch et al. and assume that object reflectance at each pixel can be expressed as a convex combination

over a small basis of homogeneous BRDFs. They demonstrate the importance of this representation of the svBRDF for achieving a stable optimization.

More recent work has extended the basic approach of Goldman et al. to avoid relying on a parametric BRDF model. The downside to using a parametric model such as the Ward BRDF is that this imposes a particular structure on the surface reflectance. Whenever the object's actual reflectance deviates from what this model can express, errors are introduced. The system presented by Alldrin et al. [27] jointly estimates the surface normal and BRDF at each pixel in a fixed camera where the BRDF is modeled using a tabulated bivariate representation. This can significantly improve the accuracy of the resulting model in many situations.

Another related approach is due to Wang et al. [28] which focuses on measuring the reflectance of anisotropic surfaces. Their system works in two steps. First, they acquire densely sampled angular measurements of the BRDF at a small number of strategically chosen points on the object surface. Second, they record sparse angular measurements sampled densely over the object surface. The sparse measurements in the second step are used to estimate a convex combination of the densely sampled BRDFs in order to recover a model of the svBRDF that is dense in both the spatial and angular dimensions. This basic strategy of combining dense angular data at a small number of surface locations with sparse angular data sampled densely over the surface represents a compelling trade-off between acquisition time and final model quality. Wang et al. also estimate the tangent direction at each surface location as a by-product of their optimization and model the BRDF using a hybrid representation that combines a tabulated (nonparametric) normal distribution function (NDF) with analytic (parametric) expressions for the shadowing, masking, and Fresnel components [19].

A Note on Bidirectional Texture Functions

This entry has focused on methods for measuring the BRDF or svBRDF of physical objects. There is a family of related techniques that focus on the bidirectional texture function (BTF) [29]. Despite having the same domain as the svBRDF, the BTF conveys a slightly more general notion of reflectance. A BTF quantifies the amount of light that is exchanged (scattered) between pairs of angles located along the surface of a

proxy geometry that does not necessarily coincide with the actual surface. As a result, the BTF couples visibility, interreflections, and local reflectance in complex ways that are perceptible at the chosen measurement scale. Technically, the difference between BTFs and svBRDFs is a matter of the degree to which this proxy geometry deviates from the actual object surface, and indeed, no real-world surface is ever perfectly smooth or exhibits exactly the microfacet structure assumed by many BRDFs. Nevertheless, some objects are more suitable than others to be represented as an svBRDF plus a surface, whereas others (e.g., cloth and other hairy or fuzzy surfaces) do not allow resolving the geometry at a fine enough resolution to isolate the local reflectance, and the BTF is perhaps the only option. Systems for measuring BTFs [29, 30] are nearly identical to those for measuring svBRDFs. The difference is a matter of how the resulting data is interpreted.

Application

Measuring and modeling the reflectance of real-world materials is a key component of most graphics and vision systems. For example, recreating a visually rich and realistic virtual 3D world requires populating it with materials with the same variety and intricacy as those found in nature. Similarly, any vision system that attempts to infer information about the 3D structure of a natural image must reason about the way light is absorbed and reradiated by the various surfaces that compose the scene. Therefore, it is important to have efficient and accurate techniques for measuring or *scanning* the reflectance of physical samples.

Open Problems

Developing techniques for measuring and modeling BRDFs and svBRDFs is an active area of research. Much of this work focuses on expanding the set of materials that can be reliably measured to include those with anisotropic and translucent properties. Another thrust of current research focuses on the usability and operation of appearance acquisition systems themselves. Measurement systems will need to be much more efficient and easier to calibrate before they can be reliably deployed in non-laboratory conditions. Finally, extending camera-based systems to measure the spectral dimensions of BRDFs is beginning to

receive serious attention. This includes resolving the spectral profiles of the incident and exitant light in addition to photoluminescence effects caused by materials that absorb energy at one wavelength and emit it at another wavelength.

References

1. Nicodemus FE, Richmond JC, Hsia JJ (1977) Geometrical considerations and reflectance. *Natl Bur Stand, NBS MN-160*
2. Cohen MF, Wallace JR (1993) Radiosity and realistic image synthesis, 1st edn. Morgan Kaufmann Publishers, San Francisco
3. Phong BT (1975) Illumination for computer generated images. *Commun ACM* 18:311–317
4. Blinn JF (1978) Simulation of wrinkled surfaces. In: Computer graphics (SIGGRAPH '78 proceedings). ACM, New York, pp 286–292
5. Torrance KE, Sparrow EM (1967) Theory for off-specular reflection from roughened surfaces. *J Opt Soc Am* 57
6. Cook RL, Torrance KE (1981) A reflectance model for computer graphics. In: Computer graphics (SIGGRAPH 1981). ACM, New York, pp 7–24
7. He XD, Torrance KE, Sillion FX, Greenberg DP (1991) A comprehensive physical model for light reflection. In: Computer graphics (SIGGRAPH 1991). ACM, New York, pp 175–186
8. Oren M, Nayar SK (1994) Generalization of lambert's reflectance model. In: Computer graphics (SIGGRAPH 1994). ACM, New York, pp 239–246
9. Kajiya JT, Kay TL (1989) Rendering fur with three dimensional textures. In: Computer graphics (SIGGRAPH 1989). ACM, New York, pp 271–280
10. Lafourture EPF, Foo SC, Torrance KE, Greenberg DP (1997) Non-linear approximation of reflectance functions. In: Computer graphics (SIGGRAPH 1997), Los Angeles, pp 117–126
11. Ward GJ (1992) Measuring and modeling anisotropic reflection. In: Computer graphics (SIGGRAPH 1992). ACM, New York, pp 265–272
12. Matusik W, Pfister H, Brand M, McMillan L (2003) A data-driven reflectance model. *ACM Trans Graph* 22(3):759–769
13. Zickler T, Enrique S, Ramamoorthi R, Belhumeur P (2005) Reflectance sharing: image-based rendering from a sparse set of images. In: Proceedings of the eurographics symposium on rendering, eurographics association, Konstanz, pp 253–264
14. Kautz J, McCool M (1999) Interactive rendering with arbitrary BRDFs using separable approximations. In: Eurographics workshop on rendering, Granada, pp 247–260
15. Lawrence J, Rusinkiewicz S, Ramamoorthi R (2004) Efficient BRDF importance sampling using a factored representation. *ACM Trans Graph* 23(3):494–503
16. Rusinkiewicz S (1998) A new change of variables for efficient BRDF representation. In: Eurographics workshop on rendering, Vienna, pp 11–22
17. White DR, Saunders P, Bonsey SJ, van de Ven J, Edgar H (1998) Reflectometer for measuring the bidirectional reflectance of rough surfaces. *Appl Opt* 37(1):3450–3454
18. Li H, Foo SC, Torrance KE, Westin SH (2006) Automated three-axis gonioreflectometer for computer graphics applications. *Opt Eng* 45:043605
19. Ngan A, Durand F, Matusik W (2005) Experimental analysis of BRDF models. In: Proceedings of the eurographics symposium on rendering, eurographics association, Konstanz, pp 117–226
20. Ghosh A, Achutha S, Heidrich W, O'Toole M (2007) BRDF acquisition with basis illumination. In: IEEE 11th international conference on computer vision, 2007 (ICCV 2007), Rio de Janeiro, pp 1–8
21. Dana KJ, Wang J (2004) Device for convenient measurement of spatially varying bidirectional reflectance. *J Opt Soc Am A* 21(1):1–12
22. Marschner S, Westin S, Lafourture E, Torrance K, Greenberg D (1999) Image-based BRDF measurement including human skin. In: Eurographics rendering workshop 99, Granada, pp 139–152
23. Bayer BE (1976) Color imaging array. United States Patent No. 3971065
24. Lensch HPA, Kautz J, Goesele M, Heidrich W, Seidel HP (2001) Image-based reconstruction of spatially varying materials. In: Eurographics workshop on rendering, London, pp 63–70
25. Goldman DB, Curless B, Hertzmann A, Seitz SM (2005) Shape and spatially-varying BRDFs from photometric stereo. In: IEEE international conference on computer vision, Beijing
26. Woodham RJ (1980) Photometric method for determining surface orientation from multiple images. *Opt Eng* 19(1):139–144
27. Alldrin N, Zickler T, Kriegman D (2008) Photometric stereo with non-parametric and spatially-varying reflectance. In: IEEE conference on computer vision and pattern recognition (CVPR). IEEE, Piscataway
28. Wang J, Zhao S, Tong X, Snyder J, Guo B (2008) Modeling anisotropic surface reflectance with example-based microfacet synthesis. *ACM Trans Graph* 27(3):41
29. Dana K, van Ginneken B, Nayar S, Koenderink J (1999) Reflectance and texture of real-world surfaces. *ACM Trans Graph* 18(1):1–34
30. Han JY, Perlin K (2003) Measuring bidirectional texture reflectance with a kaleidoscope. *ACM Trans Graph* 22(3):741–748

Reference Plane

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Related Concepts

- Camera Calibration; ► Perspective Camera; ► Perspective Transformation; ► Projection Transformation

Definition

A *reference plane* is a plane used as a reference or a constraint to solve various computer vision problems.

Background

A reference plane, because of its known geometry, can be used in many applications:

- Plane-based camera calibration [1]
- Single-view metrology [2]
- Ground plane as a reference plane for vehicle navigation
- 3D structure recovery using a reference plane

The key observation is that an image of a plane is related to the plane in space by a homography (plane projective transformation). A point not on the plane will not be mapped by the homography, resulting the so-called plane parallax.

Theory

Projective transformation is a concept used in projective geometry to describe how a set of geometric objects maps to another set of geometric objects in *projective space*. The basic intuition behind projective space is to add extra points (points at infinity) to Euclidean space, and the geometric transformation allows to move those extra points to traditional points, and vice versa.

Homogeneous coordinates are used in projective space much as Cartesian coordinates are used in Euclidean space. A point in two dimensions is described by a 3D vector. A point in three dimension is described by a 4D vector. If the homogeneous coordinates of a given point are multiplied by a nonzero scalar, the resulting homogeneous coordinates represent the same point. That is, $\lambda \mathbf{m}$ ($\lambda \neq 0$) and \mathbf{m} represent the same point. Consider a point $\mathbf{p} = [u, v]^T$ on a plane in Euclidean space; its corresponding homogeneous coordinates are $\mathbf{m} = \lambda[u, v, 1]^T$. A point at infinity on the plane is represented by $[\alpha, \beta, 0]^T$, i.e., the last element is 0. A point at infinity in 2D space can be used to describe the direction of a line on the plane. Now consider a point $\mathbf{p} = [x, y, z]^T$ in 3D Euclidean space; its corresponding homogeneous coordinates are

$\mathbf{m} = \lambda[x, y, z, 1]^T$. A point at infinity in 3D space is represented by $[\alpha, \beta, \gamma, 0]^T$, i.e., the last element is 0.

Projective linear transformations do not preserve sizes and angles. They do preserve incidence (e.g., points on a line remain on a line after transformation; two lines intersecting with each other will intersect after transformation) and cross ratio. A projective linear transformation are also known as a *collineation* or *projectivity*. In the case of projective plane (P^2), it is also known as a *homography* or *plane projectivity*. In computer vision, homography plays an important role because any two images of the same planar surface are related by a homography.

A camera is modeled by the usual pinhole (see Fig. 1). A 2D point is denoted by $\mathbf{m} = [u, v]^T$. A 3D point is denoted by $\mathbf{M} = [X, Y, Z]^T$. We use $\tilde{\mathbf{x}}$ to denote the augmented vector by adding 1 as the last element: $\tilde{\mathbf{m}} = [u, v, 1]^T$ and $\tilde{\mathbf{M}} = [X, Y, Z, 1]^T$. The relationship between the 3D point \mathbf{M} and its image projection \mathbf{m} is given by

$$s\tilde{\mathbf{m}} = \mathbf{A}[\mathbf{R} \quad \mathbf{t}]\tilde{\mathbf{M}} \quad (1)$$

where s is a scale factor; (\mathbf{R}, \mathbf{t}) , called the extrinsic parameters, is the rotation and translation which relates the world coordinate system to the camera coordinate system; and \mathbf{A} is called the camera intrinsic matrix.

Without loss of generality, we assume the reference plane is on $Z = 0$ of the world coordinate system. Let us denote the i^{th} column of the rotation matrix \mathbf{R} by \mathbf{r}_i . From Eq. (1), we have

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}.$$

By abuse of notation, we still use \mathbf{M} to denote a point on the model plane, but $\mathbf{M} = [X, Y]^T$ since Z is always equal to 0. In turn, $\tilde{\mathbf{M}} = [X, Y, 1]^T$. Therefore, a model point \mathbf{M} and its image \mathbf{m} is related by a homography \mathbf{H} :

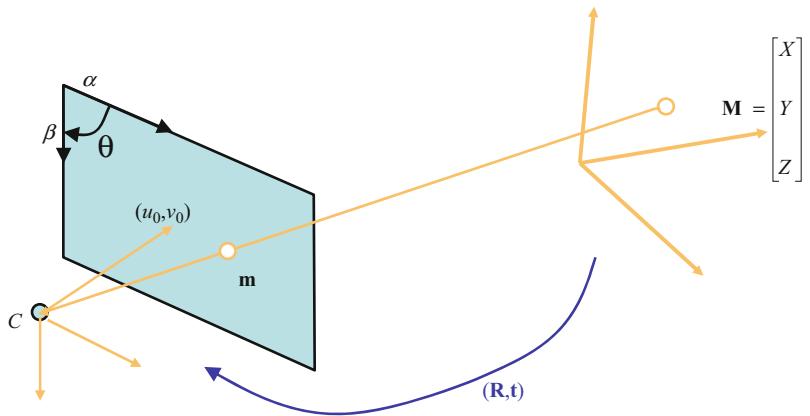
$$s\tilde{\mathbf{m}} = \mathbf{H}\tilde{\mathbf{M}} \quad \text{with} \quad \mathbf{H} = \mathbf{A}[\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}] . \quad (2)$$

As is clear, the 3×3 matrix \mathbf{H} is defined up to a scale factor.

Furthermore, two images of the same plane are related with each other also by a homography. We use

Reference Plane, Fig. 1

Pinhole camera model



the superscript ⁽¹⁾ and ⁽²⁾ to indicate the image points related to images 1 and 2, respectively. From Eq. (2), we have

$$s^{(1)}\tilde{\mathbf{m}}^{(1)} = \mathbf{H}^{(1)}\tilde{\mathbf{M}}, \quad (3)$$

$$s^{(2)}\tilde{\mathbf{m}}^{(2)} = \mathbf{H}^{(2)}\tilde{\mathbf{M}}. \quad (4)$$

It is easy to see that image point $\mathbf{m}^{(2)}$ is related to image point $\mathbf{m}^{(1)}$ by

$$s^{(2)}\tilde{\mathbf{m}}^{(2)} = \mathbf{H}^{(2)}\tilde{\mathbf{m}}^{(1)} \quad \text{with } \mathbf{H}^{(2)} = \mathbf{H}^{(2)}\mathbf{H}^{(1)-1}. \quad (5)$$

$\mathbf{H}^{(2)}$ is the homography from images 1 to 2.

For a point not on the reference, say $\mathbf{P} = [X, Y, Z, 1]^T$, its projection onto the reference plane is $\mathbf{M} = [X, Y]$. Mapping it to the image plane with the homography gives

$$s\tilde{\mathbf{m}} = \mathbf{H}\tilde{\mathbf{M}} = \mathbf{A} [r_1 \ r_2 \ t] \tilde{\mathbf{M}}.$$

The real image of point \mathbf{P} , however, is given by

$$t\tilde{\mathbf{p}} = \mathbf{A} [\mathbf{R} \ t] \tilde{\mathbf{P}} = \mathbf{A} [r_1 \ r_2 \ t] \tilde{\mathbf{M}} + \mathbf{A} r_3 Z.$$

Clearly, the plane-mapped point \mathbf{m} is not the same as the real image point \mathbf{p} . The difference is the plane parallax. The farther away the point \mathbf{P} is from the reference plane, the larger the plane parallax is. Plane parallax is an important quantity in 3D structure recovery based on a reference plane.

References

1. Zhang Z (2000) A flexible new technique for camera calibration. *IEEE Trans Pattern Anal Mach Intell* 22(11):1330–1334
2. Criminisi A (2001) Accurate visual metrology from single and multiple uncalibrated images. Springer, London/New York

Reflectance Map

Robert J. Woodham

Department of Computer Science, University of British Columbia, Vancouver, BC, Canada

Related Concepts

► Photometric Stereo; ► Radiance

Definition

A reflectance map is a function that gives scene radiance as a function of surface orientation.

Background

The amount of light reflected by a surface element in a given direction depends on its optical properties, on its microstructure, and on the spatial and spectral distribution of the light sources. For many materials, the fraction of the total irradiance reflected toward the viewer depends only on the surface orientation. Horn [1] introduced the reflectance map as a way to

specify scene radiance as a function of surface orientation, given the following simplifying assumptions:

1. The direction toward the viewer is the same at every visible point on the surface. This holds when image projection is orthographic.
2. The direction toward the light source is the same at every visible point on the surface. This holds for point light sources at infinity and for parallel (i.e., collimated) light sources.
3. Reflectance is isotropic about the surface normal. This holds when there is no inherent directional-ity in surface microstructure, making reflectance invariant to rotation of the surface about the surface normal.

A reflectance map compiles the relevant information about surface material, light source distribution, and viewing geometry into a single function. It is central to *physics-based vision* including *shape from shading* and *photometric stereo*.

Work to recover height profiles from intensity measurements originated in lunar astronomy. The height variation of points on the lunar surface is small compared to the overall distance of the moon from the earth, making image projection effectively orthographic. Direct sun illumination on the lunar surface is effectively illumination of a point light source at infinity.

As originally formulated by Horn [1], the reflectance map used the gradient to represent surface orientation. Earlier, Mackworth [2] used gradient space to express geometric constraints in the interpretation of line drawings of polyhedra. Defining the reflectance map as a function of the gradient combines radiometric and geometric constraints in a common representation, the gradient space. Subsequent variants of the reflectance map have used other representations for surface orientation including the (unit) surface normal, spherical coordinates, and stereographic coordinates.

Theory

A standard geometry is assumed. Let the visible surface be given explicitly by $z = f(x, y)$ in a left-handed Euclidean coordinate system, where the viewer is looking in the positive z direction, image projection is orthographic, illumination is parallel, and the image in the xy -axes coincides with the scene in the xy -axes.

The surface *gradient* (p, q) is defined by

$$p = \frac{\partial f(x, y)}{\partial x} \text{ and } q = \frac{\partial f(x, y)}{\partial y} \quad (1)$$

so that a surface normal vector is $[p, q, -1]$. The gradient (p, q) is one way to represent the two degrees of freedom of surface orientation.

The reflectance map, $R(p, q)$, determines scene radiance as a function of the gradient for a specific surface material, scene illumination, and viewing geometry. Further, if an ideal (calibrated) camera produces image intensity proportional to scene radiance, then the image irradiance equation is

$$E(x, y) = R(p, q) \quad (2)$$

where $E(x, y)$ is image irradiance. Equation (2) is a nonlinear, first-order partial differential equation. *Shape from shading* methods determine a surface, $z = f(x, y)$, that satisfies the image irradiance equation over some domain, Ω , including any initial conditions specified on the boundary, $\partial\Omega$, or elsewhere. Sometimes, a priori constraints on the reflectance map simplify shape analysis. Three such cases are:

1. If $R(p, q)$ is symmetric about the origin in gradient space, then it is a function of $\sqrt{p^2 + q^2}$ alone and Eq. (2) is eikonal. An eikonal image irradiance equation can often be achieved by aligning a single light source direction with the viewing direction.
2. For the special case of material in the maria of the moon and other materials for which reflectance depends only on the ratio of the incident and emergent angles, $R(p, q)$ is linear in p and q .
3. For an ideal diffuse (Lambertian) reflector, Eq. (2) becomes linear if the (unit) surface normal, rather than the gradient, is used to represent surface orientation.

Application

Generic robot vision tasks such as object recognition, pose determination, and inspection typically assume that measured brightness depends upon surface shape. When the illumination and surface material are fixed, it becomes possible to relate measured brightness directly to shape, as the reflectance map demonstrates.

Remote sensing, on the other hand, typically assumes that (multispectral) measurements define a “spectral signature” that depends upon surface material (i.e., ground cover). Not surprisingly, difficulties arise when rugged terrain and illumination change confound the measurements. Reflectance maps have been used to decouple geometric effects, associated with elevation, slope, and aspect, and from spectral effects, associated with surface material.

Determining the Reflectance Map

Reflectance maps are used in methods to determine shape and surface material from measured brightness and color. But, how are the appropriate reflectance maps determined? There are three approaches:

1. Reflectance can be modeled phenomenologically. That is, one imagines how an idealized material might reflect light and derives the expression for scene radiance accordingly. Ideal diffuse (Lambertian) reflection is one example of a phenomenological model. Phong shading, popular in computer graphics, is another. A reflectance map is obtained when the gradient (p, q) is used to represent surface orientation.
2. There is a standard nomenclature for reflectance [3]. The intrinsic reflectance properties of a surface material are specified by its *bidirectional reflectance distribution function* (BRDF). When the BRDF and light source distribution are known, the reflectance map can be derived analytically. Specific examples are given in [4].

Analytic modeling extends to taking simple phenomenological models, like ideal diffuse or ideal specular reflection, and applying them to a surface microstructure of known (or assumed) particle shape and distribution. For example, in reflectance spectroscopy, a technique in analytic chemistry, materials are ground into fine powders of known particle size and shape. Analytic models are developed to relate measured reflectance of the powders to the optical properties of the material of which they are composed.

3. Finally, a reflectance map can be measured directly using a goniometer-mounted sample or indirectly from images of a calibration object of known shape, such as a sphere. Empirical measurement has the benefit of automatically compensating for the transfer characteristics of the camera (or other sensor).

Calibration results are directly applicable to the analysis of other objects of different shape but made of the same material as the calibration object and illuminated and viewed under the same conditions. In this way, a material with any reflectance characteristic can be handled, provided that the necessary calibration can be done. In some applications, it has been useful to use paint (or other coating) to match reflectance properties between a calibration object and other objects to be analyzed.

Empirical measurement over a wide range of viewpoints and illumination conditions is both data and time intensive. This has led to approaches that are best termed semiempirical. Reflectance is assumed to take on a particular functional form, typically a linear combination of certain basis reflectance functions, and empirical measurement is used to estimate the parameters associated with the assumed functional form.

Open Problems

Image irradiance equations can be generalized to accommodate perspective projection, nearby light sources, reflectance that is not isotropic about the surface normal, and optical properties of the medium through which the radiant energy is transmitted. Increasingly, the spectral dependence of reflectance also is made explicit. This adds complexity to the associated optical equations, in the case of analytic models, and to the associated calibration/storage requirements, in the case of empirical models.

Unfortunately, with empirical models, it is difficult to use measurements acquired under one condition of illumination and viewing to predict the reflectance map for other conditions of illumination and viewing. Specifically, there is no fundamental, scale-independent distinction to be made among intrinsic optical properties (i.e., the BRDF), surface microstructure, and gross surface shape.

Any reflectance map is subject to error in the presence of cast shadows and interreflection. No purely local definition can succeed since these phenomena are inherently nonlocal. Interreflection, for example, causes changes in the local illumination owing to the secondary reflections from adjacent object points.

References

1. Horn BKP (1977) Understanding image intensities. *Artif Intell* 8:201–231
2. Mackworth AK (1973) Interpreting pictures of polyhedral scenes. *Artif Intell* 4(2):121–137
3. Nicodemus FE, Richmond JC, Hsia JJ, Ginsberg IW, Limperis T (1977) Geometrical considerations and nomenclature for reflectance. NBS monograph, vol 160. U.S. Department of Commerce, Washington DC
4. Horn BKP, Sjoberg RW (1979) Calculating the reflectance map. *Appl Opt* 18:1770–1779

Reflectance Models

Szymon Rusinkiewicz
Department of Computer Science, Princeton
University, Princeton, NJ, USA

Synonyms

Analytic reflectance functions; BRDF models

Definition

Reflectance models are analytic functions specifying the ratio of reflected radiance to irradiance at a point.

Background

Whether for recognition or image synthesis, it is frequently necessary to have a model for how much light a surface will reflect. Though a surface may reflect a different amount of light at each position, the bidirectional reflectance distribution function (BRDF) embodies the reflectance at a single point, for every possible angle of incident and exitant light. It is defined [1] as the ratio of reflected radiance in a direction ω_o to irradiance from direction ω_i :

$$f_r(\omega_i \rightarrow \omega_o) = f_r(\theta_i, \varphi_i, \theta_o, \varphi_o) = \frac{dL_o(\omega_o)}{dE_i(\omega_i)}. \quad [\text{sr}^{-1}] \quad (1)$$

Note that this equation is defined in terms of *irradiance*, which implicitly includes the famous “cosine law” of incident light. Thus, the BRDF must be multiplied by $\cos \theta_i$ in order to obtain the full variation

of reflected light as a function of the angle of incidence (for a distant light source).

Though research into photorealistic image synthesis increasingly uses dense BRDF measurements (including thousands of samples across different angles of incidence and exitance) [2], the reflectance of many surfaces is adequately predicted by simple analytic functions with few free parameters. Assuming such an analytic model makes it practical to infer the complete bidirectional reflectance from a small number of measurements, fitting the function and obtaining its parameters based on the available data [3]. This function, in turn, may be used to interpret further images.

Theory

Three general classes of reflectance models have been developed in the literature. *Physically based* models attempt to model reflectance from first principles, beginning with the solution to Maxwell’s equations on surfaces of known geometry. *Microfacet* models assume a rough surface geometry, which is not known exactly but may be characterized statistically. Finally, ad hoc, phenomenological, or *empirical* models capture some qualitative features of reflectance without necessarily striving for, or achieving, certain aspects of *physically plausible* reflectance.

Physical plausibility in reflectance typically refers to two specific properties satisfied by all actual surfaces. The first is *energy conservation*: because all incident light must be either reflected or absorbed and no light may be created during reflection, it is impossible for a surface to reflect more light than was incident on it. Mathematically, the integral of the BRDF over all outgoing directions, scaled by a cosine term to account for foreshortening, must be less than one:

$$\forall \omega_i : \int_{\Omega} f_r(\omega_i, \omega_o) \cos \theta_o d\omega_o \leq 1. \quad (2)$$

A second, more subtle, property of BRDFs is that they must be unchanged when the angles of incidence and exitance are swapped:

$$f_r(\omega_i \rightarrow \omega_o) = f_r(\omega_o \rightarrow \omega_i). \quad (3)$$

This is a condition known as *Helmholtz reciprocity* and is due to the symmetry of light transport [4]. Some systems, such as the work on Helmholtz stereopsis [5],

have relied on this property, which often expressed as camera/projector duality: in many imaging systems, it is possible to interchange the roles of camera and projector, provided that cosine terms are properly accounted for.

Lambertian Reflectance

We now turn to specific reflectance models. The simplest is just a constant:

$$f_r = \text{const.} = \rho/\pi. \quad (4)$$

(It is important to keep in mind that the BRDF is defined in terms of irradiance, which has the “incident cosine law” implicitly included.) This results in a matte or diffuse appearance and is known as ideal Lambertian reflectance. This BRDF is frequently written as a constant ρ divided by π . In this case, ρ is interpreted as the diffuse albedo: it is the fraction of light that is reflected (vs. absorbed) by the surface, and a surface with this reflectance conserves energy precisely when the albedo is less than or equal to one. Because this model is independent of the directions of incidence and exitance, it also satisfies reciprocity.

Phong and Blinn-Phong BRDFs

Another simple analytic BRDF, designed to empirically mimic the appearance of glossy (also called shiny or specular) materials, is the Phong model [6]:

$$f_r = k_s (\mathbf{r} \cdot \mathbf{v})^n, \quad (5)$$

where \mathbf{v} is the direction toward the viewer and \mathbf{r} is the mirror reflection of the light direction from the tangent plane. Note that a frequently used version of the Phong “BRDF” includes an additional $1/\cos\theta_i$ factor, which is canceled by the irradiance cosine law. The latter is therefore not a physically plausible BRDF: it does not exhibit reciprocity and does not conserve energy.

A common variant of this model is sometimes known as the Blinn-Phong model [7]:

$$f_r = k_s (\mathbf{n} \cdot \mathbf{h})^n, \quad (6)$$

though again it is often stated as a physically implausible shading model rather than a BRDF. In this equation, \mathbf{h} is the “halfway” vector, which is midway between the light direction \mathbf{l} and the viewer direction \mathbf{v} :

$$\mathbf{h} = \frac{\mathbf{l} + \mathbf{v}}{\|\mathbf{l} + \mathbf{v}\|}. \quad (7)$$

In contrast to the Lambertian BRDF, the distribution of reflected light in these models is not constant. In fact, there is a lobe centered around the direction of ideal mirror reflection for each incident angle, containing significantly more energy than the rest of the domain. This is known as the specular lobe, and its size and width (fall off) are controlled by the parameters k_s and n , respectively.

Lafortune BRDF

A popular model used for fitting analytic functions to measured BRDF data is the Lafortune model [8]:

$$f_r = (C_x l_x v_x + C_y l_y v_y + C_z l_z v_z)^n, \quad (8)$$

in which l_x , v_x , etc. represent the components of the light vector \mathbf{l} and view vector \mathbf{v} , in a coordinate system in which the surface normal is oriented along the z -axis. This model reduces to Phong by choosing $-C_x = -C_y = C_z = \sqrt[3]{k_s}$ but through suitable choice of parameters can also represent non-Lambertian diffuse reflection, off-specular reflection, anisotropy, and retro reflection. It is also common to fit a sum of multiple lobes of (Eq. 8) to measured datasets.

Ward BRDF

Another popular BRDF used in fits to measurements is the Ward model [9]:

$$f_r = k_s \frac{e^{-\tan^2\theta_h ((\cos^2\phi_h)/\alpha_x^2 + (\sin^2\phi_h)/\alpha_y^2)}}{4\pi \alpha_x \alpha_y \sqrt{\cos\theta_i \cos\theta_o}}. \quad (9)$$

Compared to the Blinn-Phong BRDF, the Ward model includes a specular peak shaped by a Gaussian function (as opposed to a power-of-cosine model) but also can model anisotropic reflection by using separate Gaussian widths α_x and α_y in two perpendicular directions.

Microfacet BRDFs

Numerous BRDFs have been derived from first principles that predict the aggregate reflectance for surfaces that at a small scale consist of tiny, mirror-reflective “microfacets” oriented in random directions. An early microfacet BRDF was originally developed in the physics community by Torrance and Sparrow [10],

introduced in graphics by Blinn [7], and later refined by Cook and Torrance [11]:

$$f_r = \frac{D G F}{\pi \cos \theta_i \cos \theta_o}. \quad (10)$$

There are three major terms in the model that describe the angular distribution of microfacets, how many are visible from each angle, and how light reflects from each facet.

The first term D in the Torrance-Sparrow model describes the density of facets facing in any possible direction:

$$D = \frac{e^{-(\tan^2 \theta_h)/m^2}}{4 m^2 \cos^4 \theta_h}, \quad (11)$$

where θ_h is the angle between the halfway vector \mathbf{h} and the surface normal \mathbf{n} . Notice that part of this term resembles a Gaussian, and this is not a coincidence: the Torrance-Sparrow model makes the assumption that the microfacet normals have a Gaussian distribution controlled by a “roughness” parameter m . The $\cos^4 \theta_h$ term occurring here is a change-of-basis term: it is included to properly normalize a probability distribution expressed in terms of the halfway vector.

The next term G in the Torrance-Sparrow model accounts for the fact that not all facets are visible from all directions, because they are hidden by the facets in front of them. It includes both “shadowing” and “masking” effects, representing occlusion from the point of view of the light and viewer, respectively:

$$G = \min \left\{ 1, \frac{2 \cos \theta_h \cos \theta_i}{\cos \theta_d}, \frac{2 \cos \theta_h \cos \theta_o}{\cos \theta_d} \right\}. \quad (12)$$

This formula is derived by considering a particular microgeometry: the microfacets are assumed to form V-shaped grooves in the surface, which are symmetric about the (macroscopic) surface normal.

Finally, the reflection from each facet is described by the Fresnel term F , which predicts that reflection increases toward grazing angles. This term arises from a solution to Maxwell’s equations on a surface:

$$F = \frac{1}{2} (F_{\perp} + F_{\parallel}) = \frac{1}{2} \left[\left(\frac{\sin(\theta_t - \theta_d)}{\sin(\theta_t + \theta_d)} \right)^2 + \left(\frac{\tan(\theta_d - \theta_t)}{\tan(\theta_d + \theta_t)} \right)^2 \right], \quad (13)$$

where θ_d is half the angle between the incident and exitant directions, $\theta_t = \sin^{-1}((\sin \theta_d)/\eta)$, η is the index of refraction of the surface, and the two terms represent the portion of reflected light polarized perpendicular and parallel to the plane of incidence. Note that the “difference angle” θ_d is the angle of incidence (and exitance) on a microfacet oriented to produce mirror reflection between the desired angles of incidence and reflection.

More recently, Ashikhmin et al. [12] generalized these types of microfacet BRDFs to allow expressing arbitrary half angle distributions. They demonstrate how to modify these BRDFs to replace the analytic distribution in Eq. 11 with alternative analytic forms or tabulated (sampled) functions that can express arbitrary patterns.

More Complex Analytic BRDFs

In addition to models for primarily specular surfaces, physically based BRDFs have been derived for rough diffuse surfaces (the Oren-Nayar model [13]) and for dusty surfaces (the Hapke/Lommel-Seeliger model, developed to model lunar reflectance [14]). They range in complexity from simple formulas that ignore many real-world effects to complex models that attempt to account for most actually observed surface phenomena (e.g., the He-Torrance-Sillion-Greenberg model [15]). While a detailed description of these models is beyond the scope of this entry, they are sometimes used in rendering or vision systems. The Oren-Nayar model, in particular, is often combined with the Torrance-Sparrow model (with the abbreviation TSON) to model surfaces with both a specular and non-Lambertian diffuse component. One drawback of these models, however, is that their additional complexity and many parameters can make it difficult or unstable to fit them to measured data.

References

- Nicodemus FE, Richmond JC, Hsia JJ, Ginsberg IW, Limperis T (1977) Geometric considerations and nomenclature for reflectance. Monograph 160. National Bureau of Standards (US)
- Matusik W, Pfister H, Brand M, McMillan L (2003) A data-driven reflectance model. ACM Trans Graph (SIGGRAPH 2003) 22(3):759–770

3. Ngan A, Durand F, Matusik W (2005) Experimental analysis of brdf models. In: Proceedings of the 15th eurographics symposium on rendering, eurographics association, pp 117–226
4. von Helmholtz H (1867) Handbuch der Physiologischen Optik. Leopold Voss, Leipzig
5. Zickler T, Belhumeur PN, Kriegman DJ (2002) Helmholtz stereopsis: exploiting reciprocity for surface reconstruction. In: ECCV '02: proceedings of the 7th european conference on computer vision-part III (ECCV). Springer, London, pp 869–884
6. Phong BT (1975) Illumination for computer generated pictures. Commun ACM 18(6):311–317
7. Blinn JF (1977) Models of light reflection for computer synthesized pictures. In: SIGGRAPH '77: proceedings of the 4th annual conference on computer graphics and interactive techniques. ACM, New York, pp 192–198
8. Lafortune EPF, Foo SC, Torrance KE, Greenberg DP (1997) Non-linear approximation of reflectance functions. In: SIGGRAPH '97: proceedings of the 24th annual conference on computer graphics and interactive techniques. ACM/Addison-Wesley, New York, pp 117–126
9. Ward G (1992) Measuring and modeling anisotropic reflection. Comput Graph 26(Annual Conference Series): 265–273
10. Torrance KE, Sparrow EM (1967) Theory for off-specular reflection from roughened surfaces. JOSA 57(9): 1105–1114
11. Cook RL, Torrance KE (1982) A reflection model for computer graphics. ACM Trans Graph 1(1):7–24
12. Ashikhmin M, Premoze S, Shirley P (2001) A microfacet-based BRDF generator. In: Computer graphics (Proceedings of ACM SIGGRAPH 2001), pp 65–74
13. Oren M, Nayar S (1994) Generalization of lambert's reflectance model. In: Proceedings of SIGGRAPH
14. Hapke B (1963) A theoretical photometric function for the lunar surface. J Geophys Res 68(15)
15. He XD, Torrance KE, Sillion FX, Greenberg DP (1991) A comprehensive physical model for light reflection. Comput Graph 25(Annual Conference Series):175–186

Reflection Mapping

- [Image-Based Lighting](#)

Reflectometry

- [Recovery of Reflectance Properties](#)

Relation Between Objects and Their Digital Images

- [Digitization](#)

Relief Texture

- [Bidirectional Texture Function and 3D Texture](#)

Retina

- [Image Plane](#)

Retinex Algorithm

- [Retinex Theory](#)

Retinex Theory

Micah K. Johnson
Computer Science and Artificial Intelligence
Laboratory, Massachusetts Institute of Technology,
Cambridge, MA, USA

Synonyms

- [Retinex algorithm](#)

Related Concepts

- [Color Constancy](#); ► [White Balance](#)

Definition

Retinex theory is a computational model for human color constancy. It defines a mechanism for computing lightness values from an image. Retinex theory proposes that the lightness values for each class of photoreceptors are derived independently and that this triplet of values correlates with perceived reflectance.

Background

The human visual system is remarkable in its ability to deal with varying illumination. Throughout the day, the visual system encounters both artificial and natural



Retinex Theory, Fig. 1 An arrangement of colored rectangles similar to “Mondrian-like” stimuli used by Land and McCann [1]

illumination, yet the appearance of the world seems stable. For example, a piece of white paper on a desk does not appear to change color when taken outside.

But anyone who has used a camera knows that the world is not as simple as it appears. Sometimes photographs captured outdoors have a blue color cast, while photographs captured indoors appear yellow. These color shifts are due to the different sources of illumination, e.g., sunlight versus an incandescent bulb. For a camera to capture the scene as it appears to the human visual system, it needs to determine, or be configured with, the color of the illumination. This setting in the camera is known as white balance.

While a camera needs to be configured for the type of illumination, the human visual system accounts for illumination changes automatically. This ability is known as color constancy. The process is fast and effortless – many people do not realize how much the measured color of an object changes under different light sources because its appearance seems stable. In a sense, the human visual system has an automatic white balance mechanism that operates without the need for a reference or knowledge of the illumination.

Land and McCann investigated color constancy under varying illumination in a series of experiments known as the “Mondrian” experiments [1].

The stimuli for the experiments consisted of a large array of rectangular colored papers that were arranged to resemble a painting by the artist Mondrian; see Fig. 1. The papers were illuminated by three projectors with filters designed to pass long waves, middle-length waves, or short waves. Focusing on one of the papers (e.g., a green paper), they adjusted the flux from each of the projectors to achieve a predetermined set of luminance values at the eye, as measured by a photometer. The subject in the study noted the color name of the paper in question. The process was repeated for other pieces of paper – first adjusting the lights to achieve the same set of luminance values, then asking the subject to note the color name. The subjects consistently reported the correct color names for the papers, despite receiving the same luminance triple at the eye from all papers. Thus, in a controlled environment, Land and McCann verified the ability of the visual system to identify correct color names under varying illumination, even when the luminance values at the eye in all cases were the same.

Theory

Retinex theory grew out of the Mondrian experiments and it defined a computational model for the color constancy of the Mondrian stimuli. Land and McCann coined the name “Retinex” to signify that, in humans, this process could involve both the retina and the cortex [1].

The main principle of Retinex theory is that images from the three classes of photoreceptors (i.e., color channels) are processed separately into three *lightness* images. The lightness values at different regions in the image correlate with the true reflectance of the region and are as independent of illumination differences as possible. Therefore, the color constancy problem is cast into a lower-dimensional problem of understanding the lightness of a single-channel image.

Retinex theory describes an algorithm for computing lightness images from intensity images. There were five main components in the original algorithm: ratio, sequential product, reset, threshold, and average [1].

Land and McCann realized the perceptual importance of edges and made the ratio of luminances across edges the fundamental unit of their algorithm. To compare areas that were not directly next to each other, they introduced the notion of a sequential product.

The sequential product accumulates edge ratios along a one-dimensional path. The initial value for the sequential product is set to one, and the sequential product can become greater than one if the path crosses a region with greater reflectance than the initial region. In this case, the sequential product is reset to one to account for the highest reflectance found along the path.

Across large areas of uniform reflectance, adjacent pixels may have slightly different luminance values, leading to ratios near, but not equal to, one. The threshold component discards these small changes to make the algorithm more robust to gradual changes of illumination across the scene.

The sequential product reveals the relative reflectance of a region with respect to the highest reflectance seen along a one-dimensional path. The Retinex algorithm considers several one-dimensional paths through the same region and averages the relative reflectances to obtain the lightness value for the region.

Through the five operations, the Retinex algorithm converts a grayscale intensity image into a lightness image. The triple of lightness values from three color channels were shown to correlate with color judgments, regardless of illumination conditions [2].

Since the original Retinex paper, there have been further studies on the roles of the different operators and modifications to these operators [3, 4]. For example, the length of the path used in the sequential product affects the ability of the algorithm to model color constancy. Short paths include little of the surrounding context, while in the limit, infinite paths result in a scaling of the image and are a poor model for human perception [5]. Later variants of the Retinex algorithm also discarded the threshold step [6], though other authors have found the threshold step to be important for separating shading and reflectance effects in images [7–9].

References

- Land EH, McCann JJ (1978) Lightness and retinex theory. *J Opt Soc Am* 61(1):1–11
- Land EH (1977) The retinex theory of color vision. *Sci Am* 237(6):108–128
- Land EH (1983) Recent advances in retinex theory and some implications for cortical computations: color vision and the natural image. *Proc Natl Acad Sci USA* 80:5163–5169
- Land EH (1986) An alternative technique for the computation of the designator in the retinex theory of color vision. *Proc Natl Acad Sci USA* 83:3078–3080
- Brainard DH, Wandell BA (1986) Analysis of the retinex theory of color vision. *J Opt Soc Am A* 3(10):1651–1661
- Funt B, Ciurea F, McCann J (2004) Retinex in MATLAB. *J Electron Imaging* 13(1):48–57
- Horn BKP (1974) Determining lightness from an image. *Comput Graph Image Process* 3:277–299
- Blake A (1985) Boundary conditions for lightness computation in Mondrian world. *Comput Vis Graph Image Process* 32:314–327
- Tappen MF, Freeman WT, Adelson EH (2005) Recovering intrinsic images from a single image. *IEEE Trans Pattern Anal Mach Intell* 27(9):1459–1472

Retroreflection

S. C. Pont

Industrial Design Engineering, Delft University of Technology, Delft, The Netherlands

Synonyms

Backscattering

Related Concepts

- Asperity Scattering; ► Lambertian Reflectance;
- Surface Roughness

Definition

Retroreflection is a type of reflectance which is characterized by a peak in the backward or illumination direction. Thus, materials that backscatter scatter light primarily in the direction from which it is illuminated.

Background

The reflectance of natural, opaque, rough surfaces can be described by the Bidirectional Reflectance Distribution Function (BRDF) [7]. BRDFs that are common and well known are those of Lambertian, perfectly diffusely scattering, and of specular surfaces. Such surfaces scatter light in all directions (diffuse scattering) or primarily in the mirror direction (specular reflection). However, natural surfaces can scatter light in many other ways. Retroreflection is just one such manner. Retroreflection can be caused by, for instance, hemispherical concavities in the surface [5, 8]. Even

if the surface is locally Lambertian, such a “thoroughly pitted surface” shows retroreflection. Another example is the triple mirror retroreflector, which is, for instance, used in reflectors for cars and bicycles (in small-scale versions) and in distance measurements (relatively large-scale versions). Many reflectors for outside applications (e.g., clothes of roadworkers, number plates) consist of a thin layer of small spherical particles with a refraction index of 2. Natural phenomena that are caused by retroreflection are “Heiligenschein” (the halo you can see around your head if you look at wet or dewy grass with the sun at a low angle behind you; for examples see [3]) and the Seeliger effect (reflective objects that are in opposition to the sun are brighter than in other positions).

Theory

The geometrical optical models with retroreflection lobes which were mentioned above result in quite complicated BRDFs. It is possible to fit retroreflection characteristics in a convenient, simplified formula (note that basic physical constraints should hold, e.g., nonnegativity, energy conservation, and Helmholtz reciprocity) [4]. For instance, the following BRDF model,

$$B(\mathbf{i}, \mathbf{j}, \mathbf{n}) = \frac{1}{\pi 2^k} \frac{(1 + \mathbf{i} \cdot \mathbf{j})^k}{(\mathbf{i} + \mathbf{j}) \cdot \mathbf{n}}, \quad (1)$$

is a perfect backscatterer in the sense that the BRDF equals $(2\pi(\mathbf{i} \cdot \mathbf{n}))^{-1}$ for coincident directions of incidence and viewing ($\mathbf{i} = \mathbf{j}$): When the normal irradiance is H_N , the irradiance becomes $H_N(\mathbf{i} \cdot \mathbf{n})$ and the backscattered radiance $H_N/2\pi$, i.e., independent of the slant of the surface. Thus, the full moon would appear as a featureless disk. It may well be the simplest analytical expression that leads to a pronounced backscatter lobe. The BRDF peaks at $\mathbf{i} = \mathbf{j}$; the parameter k determines the width of the peak. The albedo is a complicated function, for instance, for $k = 1$ the albedo is

$$\begin{aligned} A_B(\vartheta_i) &= \frac{1}{2}(1 + \cos \vartheta_i - \cos 2\vartheta_i) \\ &+ \frac{1}{4}(\cos \vartheta_i - \cos 3\vartheta_i) \log \frac{\cos \vartheta_i}{1 + \cos \vartheta_i}, \end{aligned} \quad (2)$$

where $\cos \vartheta_i = \mathbf{i} \cdot \mathbf{n}$. This albedo is $\frac{1}{2}$ for normal incidence and rises to 1 for grazing incidence. For $k > 1$ the expressions for the albedo become unwieldy, but numerical integration reveals that the albedo remains lower than one and approximately constant for a large range, then rises to one at grazing incidence. It is not obvious how to write down a backscatter BRDF with unit albedo throughout. Other possibilities for simplified formulations may be found in graphics as so-called backscatter shaders. However, care should be taken that many of these rendering applications do not fulfill the above-mentioned basic physical constraints.

Open Problems

BRDFs of natural surfaces can probably be categorized into about a dozen different modes. Currently only the forward, backward, diffuse, and surface scattering modes have been described by formal optical models.

Reflectance estimation from images suffers from image ambiguities. Prior knowledge on the reflectance statistics of natural materials plus formal descriptive models for the common modes of natural BRDFs can constrain this problem.

References

1. CURET, Columbia–Utrecht reflectance and texture database. <http://www.cs.columbia.edu/CAVE/curet>
2. Dana KJ, van Ginneken B (1977) Reflectance and texture of real-world surfaces. In: Proceedings of the IEEE computer science conference on computer vision and pattern recognition (CVPR)
3. Heiligenschein photographs, <http://www.weatherscapes.com/album.php?cat=optics&subcat=heiligenschein>
4. Koenderink JJ, Pont SC (2008) Material properties for surface rendering. Int J Comput Vis Biomech 1(1):43–53
5. Koenderink JJ, van Doorn AJ, Dana KJ, Nayar S (1999) Bidirectional reflection distribution function of thoroughly pitted surfaces. Int J Comput Vis 31(2/3):129–144
6. Lambert JH (1760) Photometria Sive de Mensure de Gradibus Luminis, Colorum et Umbræ. Eberhard Klett, Augsburg
7. Nicodemus FE, Richmond JC, Hsia JJ (1977) Geometrical considerations and nomenclature for reflectance. National Bureau of Standards, U.S. Monograph, vol 160. U.S. Department of Commerce, Washington
8. Pont SC, Koenderink JJ (2002) Bidirectional reflectance distribution function of specular surfaces with hemispherical pits. J Opt Soc Am A 19(12):2456–2466

Rigid Alignment

► [Rigid Registration](#)

Rigid Matching

► [Rigid Registration](#)

Rigid Positioning

► [Rigid Registration](#)

Rigid Registration

Yonghuai Liu¹, Ralph R. Martin², Liang Chen³,
Xuejun Ren⁴ and Longzhuang Li⁵

¹Department of Computer Science, Aberystwyth University, Ceredigion, Wales, UK

²School of Computer Science and Informatics Cardiff University, Cardiff, UK

³Department of Computer Science, University of Northern British Columbia, Prince George, Canada

⁴School of Engineering Liverpool John Moores University, Liverpool, UK

⁵Department of Computing Science, Texas A and M University at Corpus Christi, Corpus Christi, TX, USA

Synonyms

[Rigid alignment](#); [Rigid matching](#); [Rigid positioning](#); [Rigid transformation estimation](#)

Definition

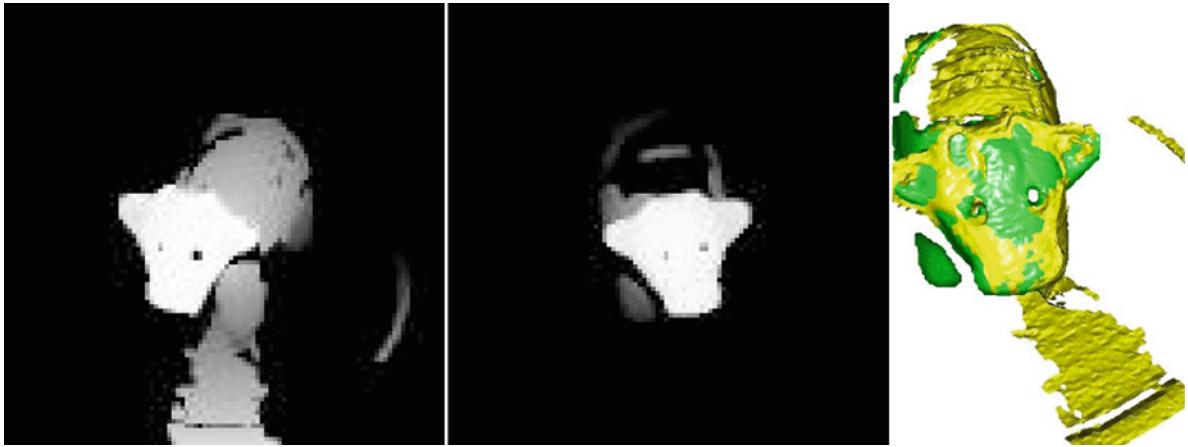
Given two copies of an object surface at different locations and orientations in space, or two parts of the surface of a single object with at least some shared overlapping area, find a translation and rigid rotation which places the objects, or corresponding parts of the object, at the same location and orientation.

This process is called rigid registration. In practice many approaches to rigid registration work by finding point-to-point correspondences between parts of the object surface in each dataset and use these to estimate the geometric transformation in either least-squares or weighted least-squares sense with closed-form solution. Often, registration algorithms also output the point-to-point correspondences which can be just as useful to many applications as the transformation itself. A correspondence is such a pair of points that while they are described in two different coordinate systems, they represent the same point on the object surface in 3D space.

Sometimes, multiple captured datasets need to be registered either simultaneously or sequentially in a pairwise manner. Sometimes, captured data need to be registered against, say, a stored CAD model of an ideal version of the object, etc. Thus, in some cases, the “reference” shape comes from the data itself, while in other cases, there is an external reference shape.

Background

Current laser scanning technologies enable the acquisition of both depth and intensity information from an object of interest in the form of range (depth) and intensity images as illustrated in [Fig. 1](#). In this case, the object of interest is a (toy) cow and is represented as structured points. In [Fig. 1](#), holes can be seen on the head and back of the cow due to inability of the scanner to capture data: its hind legs cannot be seen, while a circle and patch underneath the cow belong to the background. The occurrences and locations of such holes, occlusions, and clutter vary from one dataset to another and are unpredictable. While the range images describe the geometry of the object of interest, the intensity images describe its appearance. Since laser scanning systems (range cameras) have a limited field of view, and one part of the object may occlude another, a number of images have to be captured from different viewpoints ([Fig. 1](#)) to obtain reasonably complete coverage of the object surface; even then small gaps may remain. Each image is recorded in a local laser scanning system-centred coordinate system. To combine these images to give an overall model of the object by fusing the geometrical and optical information in these images, all images have to be placed and aligned in a single global coordinate system. This process is

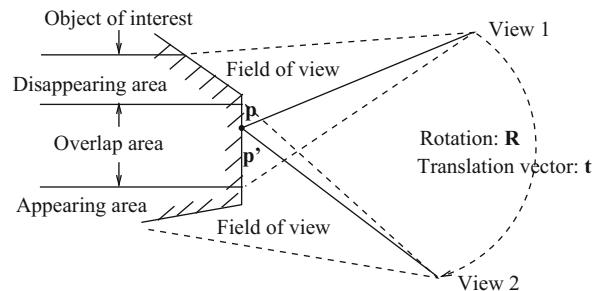


Rigid Registration, Fig. 1 Range images of a cow captured from two different viewpoints [1] are superimposed after registration. *Left:* cow45; *middle:* cow48; *right:* in the result, differently colored points belong to the two different original point sets

called registration. Shape registration has two goals: one is to establish correspondences between two overlapping shapes; the other is to estimate the underlying transformation parameters that bring one shape into the best possible alignment with the other. Fixing either of these two goals renders the other easier. However, they are in practice interwoven, complicating the shape registration process.

Suppose that the object of interest is rigid and represented using a set of points. Then interpoint Euclidean distances on the object surface are invariant with respect to changes in viewpoint from which the object is imaged. Let a general point on the object surface belong to the overlapping area between two views. Suppose it is seen in these two views as points with coordinates $(\mathbf{p}, \mathbf{p}')$. This pair is called a correspondence between the views. These coordinates are related by a rigid transformation (Fig. 2), which may be expressed using a rigid rotation matrix \mathbf{R} and a translation vector \mathbf{t} : $\mathbf{p}' = \mathbf{Rp} + \mathbf{t}$. As a rigid rotation matrix, \mathbf{R} has to satisfy two conditions: (1) unit determinant: $\det(\mathbf{R}) = 1$ and (2) orthonormality: $\mathbf{R}^T \mathbf{R} = \mathbf{I}$. \mathbf{R} and \mathbf{t} describe the relative orientation and position, respectively, of points seen in the second scene in the coordinate system of the first scene.

Automatic shape registration is a challenging problem for various reasons: (1) imaging noise: the captured data is corrupted by noise due to spatial sampling, variations in reflectance properties, depth discontinuities, mechanical control errors, signal



Rigid Registration, Fig. 2 The relationship of a correspondence $(\mathbf{p}, \mathbf{p}')$ between two views V_1 and V_2 can be described using a rigid rotation matrix \mathbf{R} and a translation vector \mathbf{t} : $\mathbf{p}' = \mathbf{Rp} + \mathbf{t}$

quantization, signal detection processes, etc. (2) occlusion: points may be absent in certain views, as one part of the object hides another; (3) limited field of view: this characteristic of scanning devices means that there is often relatively little overlap between adjacent views; (4) ambiguity between foreground and background: to facilitate data capture, a background is usually set up with different reflectance properties from the foreground to help separate background data from the foreground data. However, this is not always fully successful, resulting in foreground data being contaminated by background data; and (5) computational complexity: the amount of captured data is large, leading to ever more time-consuming computations as scanner resolution and field of view increase.

Theory

There are many approaches to registration such as feature extraction and matching [2], an optimal match of points in different shapes [3, 4], and randomized transformation search [5]. The iterative closest point (ICP) algorithm [4] was the first popular approach, and there are many variants [6–9] of it. Here a state-of-the-art approach is proposed. Assume that the two shapes to be registered are represented as two sets of unorganized points $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_{n_1}\}$ and $\mathbf{P}' = \{\mathbf{p}'_1, \mathbf{p}'_2, \dots, \mathbf{p}'_{n_2}\}$, representing two different *parts* of the same free-form shape from two different nearby viewpoints with a certain amount of overlap in 3D space (Fig. 2). Suppose that the underlying transformation parameters rotation matrix \mathbf{R} and translation vector \mathbf{t} have somehow been initialized or estimated. Then a weight w_{ij} is defined for the representation of the extent to which the point pair $(\mathbf{p}_i, \mathbf{p}'_j)$ represents a correct tentative correspondence between \mathbf{P} and \mathbf{P}' . The definition for w_{ij} must take into account imaging noise, appearance and disappearance of points, and clutter in both \mathbf{P} and \mathbf{P}' . To estimate w_{ij} , the following objective function is constructed:

$$J(\mathbf{W}) = \operatorname{argmin}_{\mathbf{W}} \sum_{j=1}^{n_2} \sum_{i=1}^{n_1} w_{ij} \|\mathbf{p}'_j - \mathbf{R}\mathbf{p}_i - \mathbf{t}\|^2 - \left(-\frac{1}{\beta} \sum_{j=1}^{n_2} \sum_{i=1}^{n_1} w_{ij} (\ln w_{ij} - 1) \right) \quad (1)$$

where $\mathbf{W} = \{w_{ij}\}$. β is a positive parameter used to describe the extent to which it is believed that the transformation parameters \mathbf{R} and \mathbf{t} are correct. At the beginning of registration, \mathbf{R} and \mathbf{t} are often quite inaccurate. In this case, all tentative correspondences $(\mathbf{p}_i, \mathbf{p}'_j)$ established can be considered equally unreliable, so β will be small, and all w_{ij} are close to a constant. As registration progresses, \mathbf{R} and \mathbf{t} become increasingly accurate, and β is asymptotically increased to penalize tentative correspondences $(\mathbf{p}_i, \mathbf{p}'_j)$ having large errors $e_{ij}^2 = \|\mathbf{p}'_j - \mathbf{R}\mathbf{p}_i - \mathbf{t}\|^2$. Only those tentative correspondences $(\mathbf{p}_i, \mathbf{p}'_j)$ with small errors now have a non-zero constant weight, and those with large errors have a weight close to 0. For all the points in one dataset without correspondents in another, the resulting tentative correspondences will have large errors and thus a weight close to 0. This is

a deterministic annealing process for global optimization of w_{ij} ; β is called the inverse temperature.

In this objective function, the first term sums the weighted registration error $e_{ij} = \|\mathbf{p}'_j - \mathbf{R}\mathbf{p}_i - \mathbf{t}\|$ for each tentative correspondence $(\mathbf{p}_i, \mathbf{p}'_j)$, while the second term is the negative entropy of these weights w_{ij} (*negative* entropy is used as this is a minimization problem). Thus, the optimization process aims to maximize the entropy (EntMax) of the weights, so that eventually, all the incorrect and correct correspondences will have the same zero and nonzero constant weights.

Differentiating the above objective function with regard to the unknown w_{ij} and setting the result to zero yields

$$w_{ij} = \exp(-\beta \|\mathbf{p}'_j - \mathbf{R}\mathbf{p}_i - \mathbf{t}\|^2). \quad (2)$$

Weights and Correspondences in the Overlapping Area

Each w_{ij} in Eq. 2 describes the extent to which $(\mathbf{p}_i, \mathbf{p}'_j)$ is correct with regard to the existing rigid transformation parameters \mathbf{R} and \mathbf{t} . However, it does not consider the situation in which in the real world, a point on the surface of an object in one view cannot correspond to more than one point in another. Let the weight a_i describe how likely it is that point \mathbf{p}_i lies in the overlapping area and the point $\mathbf{p}'_{c(i)}$ be its most likely correspondent in \mathbf{P}' . Then both a_i and $\mathbf{p}'_{c(i)}$ need to be further refined using the basic weights estimated from Eq. 2. To this end, the simplified min-sum (SMS) algorithm [10] is used. It sends messages $m_{i \rightarrow j}$ and $m'_{j \rightarrow i}$ between points \mathbf{p}_i and \mathbf{p}'_j ($i = 1, 2, \dots, n_1$; $j = 1, 2, \dots, n_2$), so that the two-way constraint of mutual correspondence is communicated and circulated among all points: (1) if \mathbf{p}_i matches \mathbf{p}'_j , then \mathbf{p}'_j should also match \mathbf{p}_i ; (2) if \mathbf{p}_i does not match \mathbf{p}'_j , then \mathbf{p}'_j should not match \mathbf{p}_i . The SMS algorithm in outline works as follows:

1. Initialize the messages $m_{i \rightarrow j} = m'_{j \rightarrow i} = w_{ij}$.
2. Update messages $m_{i \rightarrow j}$ from \mathbf{P} to \mathbf{P}' using messages $m'_{j \rightarrow i}$:
 - For each point \mathbf{p}_i , identify the point \mathbf{p}'_k in \mathbf{P}' with maximum message $m'_{k \rightarrow i}$ and also the second largest message $m'_{l \rightarrow i}$.
 - Update $m_{i \rightarrow j}$: if $j = k$, then $m_{i \rightarrow j} = w_{ij} - m'_{l \rightarrow i}$ else $m_{i \rightarrow j} = w_{ij} - m'_{k \rightarrow i}$.

3. Update messages $m'_{j \rightarrow i}$ from \mathbf{P}' to \mathbf{P} using messages $m_{i \rightarrow j}$:
 - For each point \mathbf{p}'_j , identify the point \mathbf{p}_k in \mathbf{P} with maximum message $m_{k \rightarrow j}$ and also the second largest message $m_{l \rightarrow j}$.
 - Update $m'_{j \rightarrow i}$: if $i = k$, then $m'_{j \rightarrow i} = w_{ij} - m_{l \rightarrow j}$ else $m'_{j \rightarrow i} = w_{ij} - m_{k \rightarrow j}$.
4. Estimate the weight a_i of point \mathbf{p}_i in the overlapping area with \mathbf{P}' and its most likely correspondent $\mathbf{p}'_{c(i)}$ in \mathbf{P}' : $a_i = \max_j m'_{j \rightarrow i}$ and $c(i) = \operatorname{argmax}_j m'_{j \rightarrow i}$.
5. Repeat steps 2–4 until fewer than 1% of the points \mathbf{p}_i in \mathbf{P} change correspondents $\mathbf{p}'_{c(i)}$ between two successive iterations or the maximum number (taken as 5 here) of iterations has been reached.

As long as the optimal solution to a_i and $(\mathbf{p}_i, \mathbf{p}'_{c(i)})$ is unique, it is guaranteed [10] that this message-passing scheme will find it.

Estimation of the Underlying Rigid Transformation

Once the tentative correspondences $(\mathbf{p}_i, \mathbf{p}'_{c(i)})$ and their weights a_i have been estimated, the following objective function is used to reestimate the underlying rigid transformation rotation matrix \mathbf{R} and translation vector \mathbf{t} in a weighted least-squares sense:

$$\mathbf{R} = \begin{pmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{pmatrix} \quad (4)$$

- Estimate the rotation matrix \mathbf{R} from \mathbf{q} using
- Estimate the translation vector \mathbf{t} from $(\bar{\mathbf{p}}, \bar{\mathbf{p}'})$ and \mathbf{R} as: $\mathbf{t} = \bar{\mathbf{p}'} - \mathbf{R}\bar{\mathbf{p}}$.

Summary of the Registration Algorithm

Pulling together all the ingredients described in the previous paragraphs, the registration algorithm can be summarized as follows:

Initialize \mathbf{R} to the identity matrix \mathbf{I} , rotation axis $\mathbf{h} = (0 \ 0 \ 1)^T$, rotation angle $\theta = 0.1(\text{rad})$, $\mathbf{t} = \frac{1}{n_2} \sum_{j=1}^{n_2} \mathbf{p}'_j - \frac{1}{n_1} \sum_{i=1}^{n_1} \mathbf{p}_i$, initial inverse temperature β_0 , final inverse temperature β_f , the expected relative transformation estimation error ρ as defined in the Experimental Results section, maximum number of iterations, and inverse temperature $\beta = \beta_0$

$$J(\mathbf{R}, \mathbf{t}) = \min_{\mathbf{R}, \mathbf{t}} \sum_{i=1}^{n_1} a_i \|\mathbf{p}'_{c(i)} - \mathbf{R}\mathbf{p}_i - \mathbf{t}\|^2. \quad (3)$$

This objective function can be optimized using various methods having a closed-form solution. The quaternion method [4] is one of these methods and is outlined as follows:

- Compute the weighted centroids $(\bar{\mathbf{p}}, \bar{\mathbf{p}'})$ of the tentative correspondences $(\mathbf{p}_i, \mathbf{p}'_{c(i)})$: $\bar{\mathbf{p}} = \sum_{i=1}^{n_1} a_i \mathbf{p}_i / \sum_{i=1}^{n_1} a_i$ and $\bar{\mathbf{p}'} = \sum_{i=1}^{n_1} a_i \mathbf{p}'_{c(i)} / \sum_{i=1}^{n_1} a_i$.
- Compute the weighted 3×3 covariance matrix $\mathbf{C} = \{c_{mn}\}$ of the tentative correspondences $(\mathbf{p}_i, \mathbf{p}'_{c(i)})$: $\mathbf{C} = \sum_{i=1}^{n_1} a_i (\mathbf{p}_i - \bar{\mathbf{p}})(\mathbf{p}'_{c(i)} - \bar{\mathbf{p}'})^T$.
- Compute the difference $\mathbf{D} = \{d_{mn}\}$ of matrix \mathbf{C} and its transpose \mathbf{C}^T : $\mathbf{D} = \mathbf{C} - \mathbf{C}^T$.
- Construct a new incremental 4×4 symmetric matrix \mathbf{A} using matrices \mathbf{C} , \mathbf{D} , and \mathbf{I} : $\mathbf{A} = \begin{pmatrix} \operatorname{trace}(\mathbf{C}) & \Delta^T \\ \Delta & \mathbf{C} + \mathbf{C}^T - \operatorname{trace}(\mathbf{C})\mathbf{I} \end{pmatrix}$ where $\Delta = \begin{pmatrix} d_{23} \\ d_{31} \\ d_{12} \end{pmatrix}$.
- Find the eigenvector $\mathbf{q} = (q_0 \ q_1 \ q_2 \ q_3)^T$ of matrix \mathbf{A} with the maximum eigenvalue.

Outerloop: While $(\beta_0 < \beta_f)$ do the following
Iteration number $k = 0$;

Innerloop: Do the following while the relative errors in the rotation and translation for two successive iterations are larger than a threshold ρ and while the maximum number of iterations has not been exceeded

$$k \leftarrow k + 1;$$

Estimate the weight w_{ij} of each tentative correspondence $(\mathbf{p}_i, \mathbf{p}'_j)$ between \mathbf{P} and \mathbf{P}' ;

Estimate the weight a_i of point \mathbf{p}_i and its most likely correspondent $\mathbf{p}'_{c(i)}$ in the overlapping area;

Estimate the rigid transformation parameters, the rotation matrix \mathbf{R} and translation vector \mathbf{t}

Endinnerloop

$$\beta \leftarrow \beta_r \beta$$

Endouterloop

If better estimates are available for \mathbf{R} (e.g., from a turntable) and \mathbf{t} , they should be used for initialization. In the experiments to be described later, the following values were used: $\beta_0 = n_1 n_2 / \sum_{j=1}^{n_2} \sum_{i=1}^{n_1} \|\mathbf{p}'_j - \mathbf{p}_i\|^2$, $\beta_r = 1.06$, $\beta_f = 4,000\beta_0$, and $\rho = 0.001$, and ten iterations were used at most. These values may be data dependent and thus, need to be fine-tuned for better results, even though the dependence may be weak.

The above algorithm is called the message-passing two-way (MPTW) constraint algorithm. From its development, it can be seen that it has a computational complexity of $O(n_1 n_2)$ for estimating w_{ij} , $O(n_1 n_2)$ for message passing, and $O(n_1)$ for estimating the underlying transformation. Thus, it has an overall complexity of $O(n_1 n_2)$. Experimental results produced by this method are given later in this chapter.

Application

Rigid registration finds numerous applications in digital shape reconstruction, anatomical modeling, robot navigation, object recognition, and inspection, to just name a few. These applications can be classified into three main categories according to the information sought:

- Merging information from different views for shape modeling. Rigid registration is applied in [11] to data captured by a flying laser range scanner from different viewpoints. Registration allows redundant points in overlapping areas to be removed and the remaining points to be integrated, leading to a full digital 3D model. This has been used to create a digital shape archive documenting the Bayon temple. In an example of human anatomical modeling, data from computerized tomography, video cameras, and a controllable XYZ coordinate table are registered in [12] using Image-Pro Plus software (Media-Cybernetics, Inc., www.mediacy.com) to allow modeling of the entire masticatory system.
- Using underlying transformations for localization. Different range scans from an AIS laser range finder are aligned in [13] to allow localization of a robot and avoidance of obstacles during the process of

navigation. To provide the requisite accuracy, the data are first aligned using a fast variant of the traditional iterative closest point (ICP) algorithm [4], then multiple scans are registered simultaneously for global consistency, and finally dynamic objects are detected by using additional sensors together with the information from registered frames.

- Making comparisons between an original view or CAD model and a new view for recognition and inspection. Different range images are registered in [14] that allow an ear detection rate of 99.9% and an ear identification rate of 95.4% on Collection F of the UND database. To perform recognition, ear images are detected from 2D profile images using a cascaded AdaBoost detector. The corresponding 3D ear data is then extracted from the co-registered range image and represented with local 3D features. Local features are used to construct a rejection classifier, to extract a minimal region with feature-rich data points, and finally, to compute the initial transformation for matching with the ICP algorithm. Five laser scans acquired at different stages of the construction of the steel structure of a building are registered against its 3D CAD model containing 612 objects with a total of 19,478 facets in [15] for object recognition and calculation of objects' as-built dimensions and dimensional compliance control. The ICP algorithm is adapted for the global registration. A CAD object is recognized when the surface covered by its registered as-built point cloud is larger than a threshold. For better pose estimation, the ICP variant is applied again between the matched data and model points individually for each object. Once the as-built poses of all recognized objects are calculated, they are compared to their as-designed poses in order to infer some information on the compliance of the project with respect to dimensional tolerances.

Open Problems

Automatic registration is challenging, and the following issues remain open:

- Point sampling. When a scanner is used to image an object of interest, it essentially samples the object surface. In this case, the points sampled in different views of the surface are sampled at different locations. The nature of this imaging process implies

that no exact point will be sampled from two different viewpoints. The variation in location of the sampled points is determined by the resolution of the scanner. The higher the resolution, the smaller the variation. Even though the sampling resolution can be increased, it has a limit. Consequently, no matter how good the scanner is, typically, correspondences are never correct and must have some errors.

- Robustness. Outliers are a big problem in scan data. They come from either background, other parts of the same surface, or other objects, or are artifacts of the scanning method. In practice, numbers of outliers can vary, and they can be clustered rather than spread uniformly through the data. Robust algorithms need to take such issues into account. The unpredictable proportion, distribution, and location of outliers render it difficult to evaluate the relative quality of the established correspondences. To this end, a large number of algorithms have been developed based on feature extraction and matching [2], the optimized match of points in shapes [3, 4, 6], and randomized underlying transformation search [5]. All these methods can succeed in some cases but may fail catastrophically in others. These failures have two main causes: (1) these algorithms all depend upon some parameters, many of which are data dependent, and (2) no prior knowledge is available about such things as distribution of points, occlusion, appearance and disappearance of points, imaging noise, and the magnitude of the transformation. If such knowledge is available, expected locations of overlapping and nonoverlapping points can be used to guide the registration process.
- Computational efficiency. In theory, only three correct exact correspondences are needed for the estimation of the underlying transformation. In practice, since the correspondences are usually contaminated by noise, more correspondences should be used instead to obtain a consensus. On the other hand, the more measured points there are for each shape to be registered, the more candidates that must be considered in the search for correct correspondences. Various algorithms exist for searching the space of tentative correspondences, with various computational complexities. Both the max-product algorithm [16] and the game-theoretic approach [17] have a computational complexity of

$O(n^4)$, while the simplified min-sum algorithm [10] and the brute force iterative closest point (ICP) algorithm [4] have a computational complexity of $O(n^2)$, and the variant of the ICP algorithm in [6] has a computational complexity of $O(n \log n)$. These algorithms all have greater than linear time complexity. Both the resolution and field of view of scanners have increased steadily in the last decade, and now millions of points can be captured within minutes. While such a large number of points are useful to accurately capture the geometry and fine details of shapes, it causes difficulties for registration. An algorithm with a computational complexity of $O(n^2)$ can be practically applied interactively to shapes with thousands of points, and one with a computational complexity of $O(n \log n)$ can be used with tens of thousands of points. Other cases take tens of minutes upwards to perform registration and thus cannot satisfy the requirements of real-time applications.

- Performance measurement. Various methods have been developed in the literature to assess the performance of registration algorithms: the root-mean-squared-distance (RMSD) [7, 17], mean squared error (MSE) [5, 15], the average of registration errors of reciprocal correspondences [6], or the surface interpenetration measure (SIM) [5]. However, no single parameter can always successfully capture the performance of different algorithms with different data. All these methods measure different aspects of the success of registration algorithms. A further problem arises when the underlying rigid transformation is modeled as nonrigid [18], using, e.g., thin-plate splines (TPS). In such cases, these performance measures may indicate low errors, but this does not necessarily mean that the registration is accurate. This can happen because the TPS has $n + 4$ degrees of freedom and often overfits any noisy points and outliers in the data.

Experimental Results

In this section, both synthetic data and a pair of overlapping real measurements of a toy cow (cow45, cow48) [1] are used to validate the MPTW algorithm given earlier. In this case, ground truth values of the transformation parameters are known.

Synthetic Shape Generation

The synthetic data were generated as follows. Firstly, n points $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n\}$ were randomly generated with uniform distribution within the 3D space $[5, 15] \times [5, 15] \times [5, 15]$. These points were then subjected to a rotation angle θ around a fixed unit rotation axis $\mathbf{h} = (h_x \ h_y \ h_z)^T$ randomly placed and oriented with uniform distribution within the 3D space $[1, 3] \times [1, 3] \times [1, 3]$ followed by a constant translation vector \mathbf{t} randomly generated with uniform distribution within the 3D space $[10, 20] \times [10, 20] \times [10, 20]$. Let the transformed points be $\mathbf{P}' = \{\mathbf{p}'_1, \mathbf{p}'_2, \dots, \mathbf{p}'_n\}$, where $\mathbf{p}'_i = \mathbf{R}\mathbf{p}_i + \mathbf{t}$, $\mathbf{R} = \mathbf{I} + \mathbf{H} \sin \theta + \mathbf{H}^2(1 - \cos \theta)$, and $\mathbf{H} = \begin{pmatrix} 0 & -h_z & h_y \\ h_z & 0 & -h_x \\ -h_y & h_x & 0 \end{pmatrix}$. Given this data, precise knowledge of the selected points and their correspondents $(\mathbf{p}_i, \mathbf{p}'_i)$ ($i = 1, 2, \dots, n$) is available. The underlying transformation given by \mathbf{R} and \mathbf{t} serves as ground truth for error estimation and validation of the algorithm.

In order to simulate noisy real-world data, Gaussian white noise was added to the coordinates of each point with standard deviation $\sigma_1 = 0.1$ in one series of experiments and $\sigma_2 = 0.2$ in another. In order to simulate occlusion, appearance, and disappearance of points in both \mathbf{P} and \mathbf{P}' , the last 40% points in \mathbf{P} and the first 30% points in \mathbf{P}' were removed, giving two new sets of points \mathbf{P} and \mathbf{P}' for registration with 30% overlap in 3D. (Note that this simulates a new scenario where appearing and disappearing points can appear anywhere in 3D space, instead of at only boundaries of shapes. Such different scenarios are useful to test the robustness of registration algorithms.)

Performance Measurement

The performance of the MPTW algorithm was measured in three ways:

- The percentage relative errors $e_{\mathbf{h}}$, e_{θ} , and $e_{\mathbf{t}}$ in estimated rotation axis $\hat{\mathbf{h}}$, rotation angle $\hat{\theta}$, and translation vector $\hat{\mathbf{t}}$ for the underlying transformation: $e_{\mathbf{h}} = \|\hat{\mathbf{h}} - \mathbf{h}\| \times 100\%$, $e_{\theta} = (\hat{\theta} - \theta)/\theta \times 100\%$, and $e_{\mathbf{t}} = \|\hat{\mathbf{t}} - \mathbf{t}\|/\|\mathbf{t}\| \times 100\%$, where $\hat{\theta} = \cos^{-1}((r_{11} + r_{22} + r_{33} - 1)/2)$, $\hat{\mathbf{h}} = (r_{32} - r_{23}, r_{13} - r_{31}, r_{21} - r_{12})^T / \sin \hat{\theta}$, and $\hat{\mathbf{R}}$ and $\hat{\mathbf{t}}$ are estimated rotation matrix and translation vector. The relative error e_r of the rotation vector is defined as $e_r = \|\hat{\theta}\hat{\mathbf{h}} - \theta\mathbf{h}\|/\theta \times 100\%$.

– The average e_{μ} and standard deviation e_{σ} of registration errors of the finally established reciprocal correspondences (RCs) and the χ^2 distance d : $e_{\mu} = \frac{1}{N} \sum_{i=1}^N e_i$, $e_{\sigma} = \sqrt{\frac{1}{N} \sum_{i=1}^N (e_i - e_{\mu})^2}$, and $d = \frac{1}{N^2} \sum_{j=1}^N \sum_{i=1}^N (c_{ij} - 1)^2/(c_{ij} + 1)$, where N is the number of RCs, $e_i = \|\mathbf{p}'_{c(i)} - \hat{\mathbf{R}}\mathbf{p}_i - \hat{\mathbf{t}}\|$, $c_{ij} = \min(\|\mathbf{p}_j - \mathbf{p}_i\|, \|\mathbf{p}'_{c(j)} - \mathbf{p}'_{c(i)}\|) / \max(\|\mathbf{p}_j - \mathbf{p}_i\|, \|\mathbf{p}'_{c(j)} - \mathbf{p}'_{c(i)}\|)$, and $(\mathbf{p}_i, \mathbf{p}'_{c(i)})$ and $(\mathbf{p}_j, \mathbf{p}'_{c(j)})$ are any two established RCs. The χ^2 distance measures the extent to which the established RCs are compatible in the sense of rigidity: $\|\mathbf{p}_j - \mathbf{p}_i\| = \|\mathbf{p}'_{c(j)} - \mathbf{p}'_{c(i)}\|$. An RC is a correspondence $(\mathbf{p}_i, \mathbf{p}'_{c(i)})$ that if point \mathbf{p}_i in \mathbf{P} corresponds to $\mathbf{p}'_{c(i)}$ in \mathbf{P}' , then $\mathbf{p}'_{c(i)}$ in \mathbf{P}' also corresponds to \mathbf{p}_i in \mathbf{P} . Clearly, RCs satisfy the two-way constraint and are thus (more) likely to represent correct correspondences.

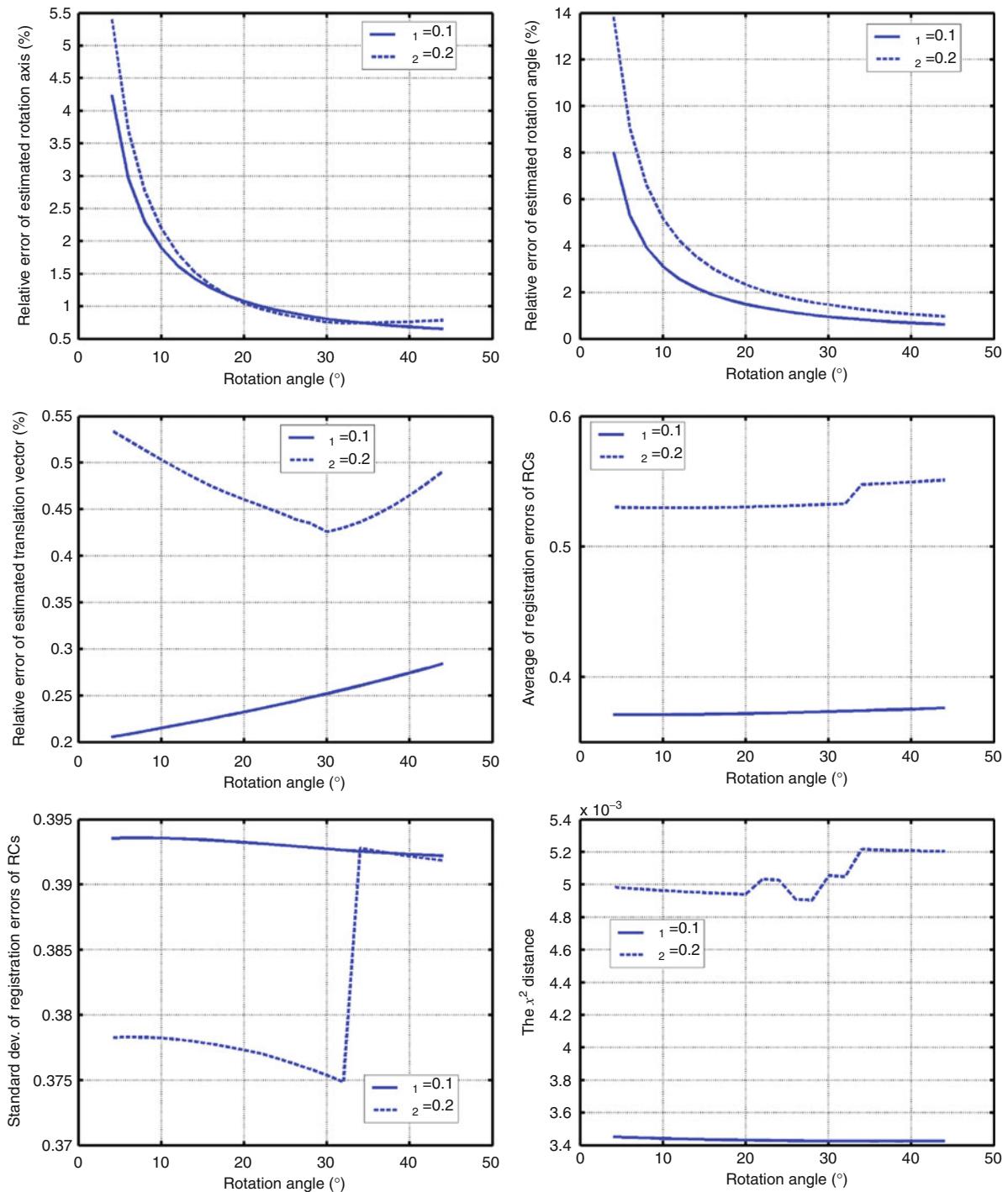
– Computational time. All the experiments were carried out on a Pentium IV 2.80GHz computer with 512M memory and implemented using the C programming language inside Microsoft Visual C++ 6.0.

While the first kind of measurements requires ground truth, the second does not. The smaller these values are, the more accurate the registration algorithm is.

Results and Analysis

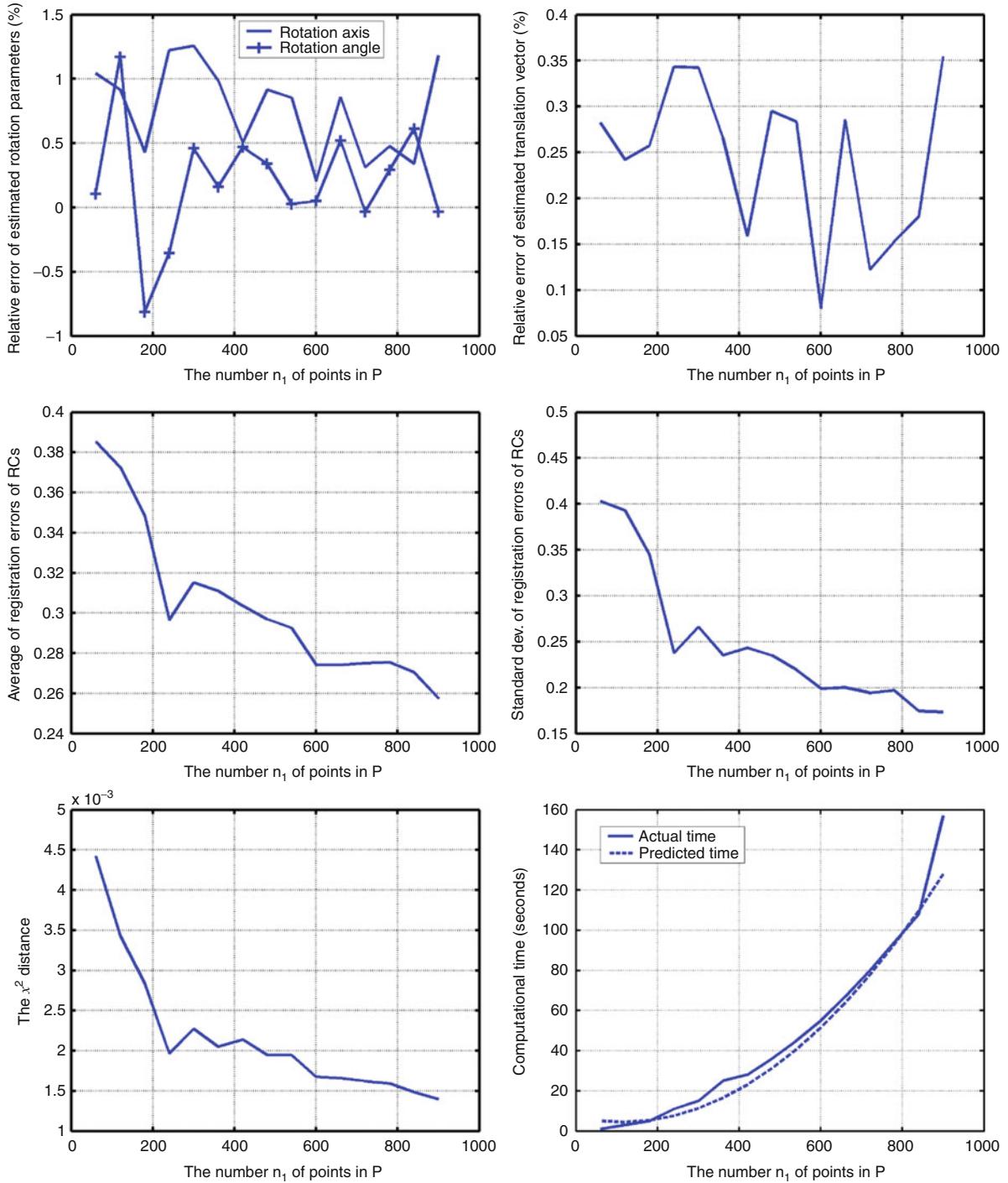
The experimental results for synthetic data are presented in Figs. 3 and 4 and Tables 1 and 2. Solid lines correspond to a lower level $\sigma_1 = 0.1$ of noise, while dashed lines correspond to a higher level $\sigma_2 = 0.2$ of noise. In Fig. 3 and Table 1, n was fixed as $n = 200$ with the rotation angle θ of the underlying transformation ranging from 4° to 44° at intervals of 2° . In Fig. 4 and Table 2, the rotation angle θ of the underlying transformation was fixed as $\theta = 25^\circ$ with the parameter n ranging from 100 to 1,500 at intervals of 100, and the data were subject to the lower level $\sigma_1 = 0.1$ of noise.

From Fig. 3 and Table 1, it can be seen that all the parameters of interest smoothly vary with the rotation angle of the underlying transformation, showing that the MPTW algorithm is very stable. When data were corrupted by stronger noise, the MPTW algorithm typically produced worse registration results, as might be expected. The estimation of the translation vector is usually more accurate than that of either the rotation axis or angle. The reason why the χ^2 distance is small



Rigid Registration, Fig. 3 Different performance parameters as a function of the rotation angle of the underlying transformation. *Top left:* rotation axis; *Top right:* rotation angle; *Middle*

left: translation vector; *Middle right:* the average of the registration errors of RCs; *Bottom left:* standard deviation of registration errors of RCs; *Bottom right:* the χ^2 distance



Rigid Registration, Fig. 4 Different performance measures as a function of the number n_1 of points in the first shape \mathbf{P} . *Top left*, rotation parameters; *top right*, translation vector; *middle*

left, average of registration errors of RCs; *middle right*, standard deviation of registration errors of RCs; *bottom left*, the χ^2 distance; *bottom right*, computational time

Rigid Registration, Table 1 Average μ and standard deviation σ of the relative estimation errors e_h , e_θ , and e_t of the rotation axis \hat{h} , rotation angle $\hat{\theta}$, and translation vector \hat{t} ; average e_μ and standard deviation e_σ of registration errors of RCs, and the χ^2 distance of the MPTW algorithm as a function of the rotation angle θ of the underlying transformation

| Noise | Meas | e_h (%) | e_θ (%) | e_t (%) | e_μ | e_σ | χ^2 Dis |
|-------|----------|-----------|----------------|-----------|---------|------------|--------------|
| 0.1 | μ | 1.30 | 1.95 | 0.24 | 0.37 | 0.39 | 0.0034 |
| | σ | 0.88 | 1.80 | 0.02 | 0.00 | 0.00 | 0.0000 |
| 0.2 | μ | 1.45 | 3.20 | 0.47 | 0.54 | 0.38 | 0.0050 |
| | σ | 1.17 | 3.14 | 0.03 | 0.01 | 0.01 | 0.0001 |

Rigid Registration, Table 2 Average μ and standard deviation σ of relative estimation errors e_h , e_θ , and e_t of rotation axis \hat{h} , rotation angle $\hat{\theta}$, and translation vector \hat{t} ; average e_μ and standard deviation e_σ of registration errors of RCs; the χ^2 distance and the computational time in seconds of the MPTW algorithm as a function of the number n_1 of points in the first shape \mathbf{P}

| Meas | e_h (%) | e_θ (%) | e_t (%) | e_μ | e_σ | χ^2 Distance | time (s) |
|----------|-----------|----------------|-----------|---------|------------|-------------------|----------|
| μ | 0.77 | 0.20 | 0.24 | 0.30 | 0.25 | 0.0022 | 48.67 |
| σ | 0.34 | 0.44 | 0.08 | 0.04 | 0.07 | 0.0008 | 43.70 |

is that the compatibility of RCs increases as a function of their Euclidean distances when they are far apart, even though they may have the same registration error.

From Fig. 4 and Table 2, it can be seen that all the parameters of interest fluctuate somewhat as a function of the number n_1 of points in \mathbf{P} . This is because the points were randomly generated each time and thus their distribution changed accordingly. The overall trend is clear and stable. The actual computational time exhibits quadratic behaviour in the number n_1 of points in \mathbf{P} , fitting a second order polynomial: $time = 0.0002n_1^2 - 0.0458n_1 + 7.0813$ s. Note that in this case, both \mathbf{P} and \mathbf{P}' have similar numbers of points. If n_1 were 10,000, it would have taken 19,549 s to register the shapes: despite its accuracy and robustness, the MPTW algorithm is unsuited to the registration of shapes with a large number of points. To make it practically useful, either the numbers n_1 and n_2 of points from \mathbf{P} and \mathbf{P}' must be reduced by sampling or fewer combinations of these points must be considered for initial weight estimation and message passing.

The cow45 and cow48 shapes have a transformation with an expected rotation angle of 30° around an unknown rotation axis and include 7,049 and 3,195 points with an average of the distances between neighboring points being 1.66 mm and 1.73 mm, respectively. The estimated rotation matrix $\hat{\mathbf{R}}$ and translation

vector $\hat{\mathbf{t}}$ are $\hat{\mathbf{R}} = \begin{pmatrix} 0.86 & -0.43 & 0.29 \\ 0.42 & 0.90 & 0.09 \\ -0.30 & 0.05 & 0.95 \end{pmatrix}$ and

$\hat{\mathbf{t}} = (389.74 \ 115.65 \ -61.49)^T$, from which the rotation axis \hat{h} and angle $\hat{\theta}$ are $\hat{h} = (-0.04 \ 0.57 \ 0.82)^T$ and 31.11°, respectively. The relative error in the estimated rotation angle is 3.70%. The number N of

RCs found is $N = 1643$, while the average and standard deviation error in the RCs are 0.80 and 0.64 mm. This is a sub-pixel accuracy that a registration algorithm can at best achieve. The χ^2 distance is 0.0006. The superimposition of the transformed cow45 and cow48 shapes is presented in Fig. 1 – the yellow color represents the transformed cow45 shape \mathbf{P} , and the green color represents the reference cow48 shape \mathbf{P}' . Even though no a priori knowledge was given about what the two point clouds represent, and they have an overlap of just $1,643 / \max(7,049, 3,195)$ points = 23.31%, they have been brought into a good alignment with a large amount of interpenetration. The disappearing cluttered background and the appearing left front leg have been correctly identified. The computational time taken for registration was 7,595 s, which is less than predicted: 9,622 s, resulting in a prediction error of 21%. This is because the relative difference in the numbers of points in \mathbf{P} and \mathbf{P}' is larger in the real data than in the synthetic data.

References

- OSU(MSU/WSU) range image database (2010) <http://sampl.ece.ohio-state.edu/data/3DDB/RID/index.htm>
- Johnson A, Hebert M (1999) Using spin images for efficient object recognition in cluttered 3D scenes. IEEE Trans Pattern Anal Mach Intell 21:433–449
- Gold S, Rangarajan A et al (1998) New algorithms for 2-D and 3-D point matching: pose estimation and correspondence. Pattern Recognit 31:1019–1031
- Besl PJ, McKay ND (1992) A method for registration of 3-D shapes. IEEE Trans Pattern Anal Mach Intell 14(2): 239–256

5. Silva L, Bellon Olga RP, Boyer KL (2005) Precision range image registration using a robust surface interpenetration measure and enhanced genetic algorithms. *IEEE Trans Pattern Anal Mach Intell* 27:762–776
6. Liu Y (2005) Automatic 3d free form shape matching using the graduated assignment algorithm. *Pattern Recognit* 38:1615–1631
7. Phillips JM, Liu R, Tomasi C (2007) Outlier robust ICP for minimizing fractional RMSD. In: Proceedings of International Conference on 3D Digital Imaging and Modeling. IEEE Computer Society, Washington DC, pp 427–434
8. Chen Y, Medioni G (1992) Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155
9. Rusinkiewicz S, Levoy M (2001) Efficient variants of the ICP algorithm proceedings of 3rd international conference on 3-D digital imaging and modeling, pp 145–152
10. Bayati M, Shah D, Sharma M (2008) Max-product for maximum weight matching: convergence, correctness, and LP duality. *IEEE Trans Inf Theory* 54(3):1241–1251
11. Banno A, Masuda T, Oishi T, Ikeuchi K (2008) Flying laser range sensor for large-scale site-modeling and its applications in Bayon digital archival project. *Int J Comput Vis* 78:207–222
12. Pileickien G, Surna A et al (2009) A new technique for the creation of a higher accuracy 3D geometrical model of the human masticatory system. *Mechanika* 78(4): 44–50
13. Surmann H, Nuchter A, Hertzberg J (2003) An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. *Robot Auton Syst* 45:181–198
14. Islam SMB, Davies R, Bennamoun M, Mian AS (2011) Efficient detection and recognition of 3D ears. *Int J Comput Vis* 95:52–73
15. Bosch F (2010) Automated recognition of 3D CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction. *J Adv Eng Inform* 24(1):107–118
16. Xiao P, Barnes N, Lieby P, Caetano T (2009) Applying sum and max product algorithms of belief propagation to 3D shape matching and registration. In: Proceedings of International Conference on Digital Image Computing: Techniques and Applications. IEEE Computer Society, Washington DC, pp 387–394
17. Albarelli A, Rodola E, Torsello A (2010) A game-theoretic approach to fine surface registration without initial motion estimation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), San Francisco, pp 430–437
18. Chui H, Rangarajan A (2003) A new point matching algorithm for non-rigid registration. *Comput Vis Image Underst* 89:114–141

Rigid Transformation Estimation

► Rigid Registration

Robot-Camera Calibration

► Hand-Eye Calibration

Robust Clustering

► Robust Methods

Robust Methods

Peter Meer¹ and Sushil Mittal²

¹Department of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ, USA

²Department of Statistics, Columbia University, New York, NY, USA

Synonyms

Robust clustering; Robust regression

Definition

The goal of robust methods in computer vision is to extract all the information necessary to solve a given task while discarding everything that is not needed. The tasks can be very simple or very complex, but in real-life applications, a robust procedure will always be required. In the end, the performance of a machine for solving a vision problem will be judged against that of human observers performing the equivalent task. Since the human visual system works in a much more sophisticated manner than the present day computer vision systems, this ultimate goal is still far from reality.

Background

The task of a robust algorithm is to derive a *model* that estimates one or more structures present in the data, each structure depending only on a part of the data. A point obeying the model is called an *inlier*, while a point not obeying it is called an *outlier*. Since

there are several inlier structures, therefore relative to one inlier structure the outliers can either be on another inlier structure (structured outliers) or can be completely unstructured. The model estimation problems can be divided into two categories – location estimation (clustering) and regression estimation.

Nonparametric method-based image segmentation is a typical example of a location estimation problem. In this class of problems, first the data points in the original space are transformed to a feature space (suitable for the particular problem), where they form *meaningful* clusters. The goal of a robust algorithm then is to find correct estimates of the cluster centers and associate observed data points to their respective clusters while rejecting the outliers. Among nonparametric clustering algorithms, kernel density estimation-based methods are used very frequently for vision applications. These methods start by computing a kernel density estimate over the entire observed data. The modes of this density correspond to the cluster centers of the data in the feature space. These modes, together with their associated points, are then recovered using techniques such as mean shift [1]. The bandwidth of the kernel is a crucial parameter which is either supplied by the user or determined from the data prior to estimating the density.

Robust regression is the second class of estimation problems. Currently, in computer vision, a maximum of four or five regression inlier structures can be estimated from the data. The structures are usually recovered iteratively, one after the other. Almost all the models in geometric vision describing 3D relations, 3D to 2D relations, or 2D to 2D relations are nonlinear. For example, the epipolar geometry between corresponding points in two frames, that is, the fundamental or essential matrix estimation, has to satisfy the constraint $\mathbf{y}_2^T \mathbf{F} \mathbf{y}_1 = 0$. Given a few point correspondences between the two frames, a 3×3 matrix \mathbf{F} has to be determined. The equation written in affine coordinates has four variables – the x, y coordinates of the matched point in the two frames. However, due to errors in the output of a point matching algorithm, the points may or may not be in correspondence.

In most of the cases, a robust computer vision estimator starts by reinterpreting the nonlinear relation (generally) in a higher-dimensional linear space. The original variables and their monomials which appear in products become the linearized variables \mathbf{x}_j . In the case of fundamental matrix estimation, the derived

linear equation has eight variables, the four original variables and the four monomials – the products terms formed by multiplying each coordinate from the first image with each coordinate from the second image. See *Epipolar Constraint* entry.

The linearized variables are much easier to estimate, and the inliers can be obtained using a robust procedure. However, the original problem almost certainly has additional constraints which also have to be satisfied. A nonlinear processing, taking into account all the constraints, leads to the final estimate. For data containing multiple inlier structures, the whole procedure is repeated for each structure until all of the structures have been recovered. A good robust algorithm should be able to reliably extract all the inlier structures (in general having different inlier noise) *without* any prior knowledge about the actual number of structures present.

The epipolar geometry example mentioned above has a simple linearized constraint. But more complex cases also exist. In camera calibration (or camera pose recovery), there are five original variables – the coordinates of a point in 3D and the corresponding coordinates in 2D. After linearization, the constraint becomes a 2×12 matrix for each point pair. In projective factorization over F frames, with all points visible in each frame, the original and the linearized variables have the same dimensions – a $3F \times (3F - 3)$ matrix. See the respective entries.

Due to linearization, the first-order approximation of the covariance matrix of the linearized variables is heteroscedastic, that is, each of the points has a different covariance even when the original variables are homoscedastic, that is, have the same variance. The heteroscedasticity of the linearized variables is not taken into account in majority of the robust algorithms. Robust estimators used in computer vision should be unbiased in the first-order approximation. This means that an estimate should converge toward its theoretical value (up to first order) when the average is taken over a large number of inliers. Heteroscedasticity plays an important role in achieving that.

The following procedure shows how to estimate each inlier structure. After the linearization, let the subscript “o” denote the theoretical value of the m -dimensional linearized variable \mathbf{x}_{io} . In the most general case, an $m \times k$ matrix, Θ , and a k -dimensional intercept, α , have to be estimated. Their theoretical values are obtained using

$$\Theta^T \mathbf{x}_{io} - \alpha = \mathbf{0}_k \quad i = 1, \dots, n . \quad (1)$$

If Eq. (1) is solved correctly, $n_1 < n$ inliers are returned together with the estimates $\hat{\Theta}$ and $\hat{\alpha}$ and the corrected points $\hat{\mathbf{x}}_i$. If required, the covariances of these points can also be computed.

Many times, the constraint $\Theta^T \Theta = \hat{\Theta}^T \hat{\Theta} = \mathbf{I}_{k \times k}$ is also used during the estimation procedure since it reduces the ambiguity of the estimate Θ only to an orthonormal gauge matrix [2]. This constraint defines a k -dimensional subspace of the n -dimensional space and thus lies on a Grassmann manifold. While exploiting group theoretic methods for processing over Grassmann manifolds is complicated, they can be used to estimate a better local optimum [15]. A nonlinear variant of mean shift has also been used to perform clustering over Grassmann manifolds [3].

Important Methods

Robust computer vision and robust statistics appeared independently at the end of 1950s and beginning of 1960s. In computer vision, the Hough transform was introduced in 1959 in a conference, and it was patented in United States in 1962. See *Hough Transform* entry.

While the rejection of outliers was practiced for hundreds of years in statistics, the first so-called robust estimator, the M-estimator, was proposed by Huber only in 1964 [4]. There are many different types of M-estimators, but each of them requires an estimate of the scale (variance) of the inlier noise beforehand. In regression problems, the estimation of scale is equivalent to the kernel bandwidth estimation in clustering. Today, in computer vision, only few M-estimator types are still used.

The least median of squares (LMedS) was introduced by Rousseeuw in 1984 and was in fashion in computer vision too in the 1990s. Today, LMedS is no longer used. While the scale of the inlier noise estimate is not needed beforehand, the LMedS can recover only a single inlier structure containing more than 50 % of the total points present in the data. The statistics of LMedS is described in the book [5], while the paper [6] also presents some of its computer vision applications. A survey of M-estimators and LMedS, including their limitations, can be found in [7], while the special issue of Computer Vision and Image Understanding [8]

gives a good overview of the robust procedures of the 1990s.

In computer vision, the M-estimators in general are employed with a nonconvex, redescending loss function, which, at least theoretically, does not always converge to the global optimum solution. The prototype of this loss function class is

$$0 \leq \rho(u) \leq 1 \quad |u| \leq 1 \quad \rho(u) = 1 \quad |u| > 1 . \quad (2)$$

It is nonnegative with $\rho(0) = 0$, even symmetric with $\rho(-u) = \rho(u)$, and nondecreasing with $|u|$. Examples of several loss functions can be found in [4, 7].

In the simplest case, given the measured points and their the m -dimensional linearized values \mathbf{x}_i , an m -dimensional vector θ and a scalar intercept α need to be estimated. The theoretical formulation is

$$\operatorname{argmin}_{\alpha, \theta} \frac{1}{n} \sum_{i=1}^n \rho \left(\frac{\theta^T \mathbf{x}_{io} - \alpha}{s} \right) , \quad (3)$$

where the user has to supply *beforehand*, the scale s of the inlier noise. The residual of \mathbf{x}_i is usually nonzero and is used to classify it either as an inlier or an outlier, depending on the value of scale and the loss function. Since $\rho(u)$ is symmetrical in the residuals, loss function in u^2 can be introduced instead. More complex M-estimators can then be represented with a squared Mahalanobis distance as the argument of $\rho(u^2)$. The $k \times k$ scale matrix \mathbf{S} is diagonal and like before has to be supplied by the user.

The estimation is performed iterative. The initial parameter estimate is obtained by ordinary or total least squares. The nonnegative weights

$$w(u_i) = \frac{1}{u_i} \frac{d\rho(u_i)}{du} \geq 0 \quad u_i = \frac{\theta^T \mathbf{x}_{io} - \alpha}{s} \quad i = 1, \dots, n \quad (4)$$

are then computed. The values of these weights decrease from the maximum at $u = 0$ to zero at $u = \pm 1$. Beyond $u = \pm 1$, the weights are zero, and the corresponding points are considered outliers. A $n \times n$ diagonal matrix of the weights is defined, and the updated model estimate is computed taking the weights for all points into consideration. In practice, the procedure converges after a few iterations.

For any M-estimator, at least three disadvantages can be enumerated. Firstly, often it is difficult to give a good estimate of the scale of the inlier noise. For example, when the estimation is performed over a video sequence, the real scale may change, and it is very difficult to guess that change online. Secondly, in every iteration, *all* the samples participate in the process, and if the initial estimate is completely wrong, then it is very hard to recover. Thirdly, the majority of M-estimators reject only a few outliers and do not work with asymmetric noise or with data having multiple inlier structures. [Figure 1](#) shows a line fitting example using M-estimator. While the estimate is good in case of symmetric outliers around the inliers, it is grossly incorrect when the outliers are little asymmetric.

Today in computer vision, almost everybody uses the *RANdom SAmple Consensus* (RANSAC), proposed by Fischler and Bolles in 1980 and published in a journal in 1981 [9]. RANSAC started to take over around 1995, and in 2006, a workshop to honor 25 years of the initial publication was held in conjunction with the conference on Computer Vision and Pattern Recognition.

Like M-estimator, the scale of inlier noise is still provided by the user *beforehand* in RANSAC, but the estimation is based on a totally different principle. Unlike M-estimator, in RANSAC, a hypothesis is drawn exclusively from an *elemental subset*. An elemental subset is the smallest number of data points required to fully estimate a model. The model being linear Eq. (1), the number of points in the elemental subset should be equal to the number of unknowns in the linearized model equation. For example, in the scalar case of the fundamental matrix, a hypothesis needs eight points because the 3×3 matrix \mathbf{F} , written as a vector, has eight independent linear unknowns. See *8-point algorithm* entry. Other hypotheses also exist, like the 7-point algorithm for fundamental matrices which takes into account the rank-two constraint of \mathbf{F} too. The method requires the satisfaction of additional, in general, nonlinear constraints leading to more complicated computations. Using the same notation as defined in Eq. (3), the simplest theoretical objective function is

$$\operatorname{argmin}_{\alpha, \boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^n \rho_{zo} \left(\frac{\boldsymbol{\theta}^T \mathbf{x}_{io} - \alpha}{s} \right). \quad (5)$$

The value of ρ_{zo} , loss function for RANSAC is

$$\rho_{zo}(u) = 0 \quad |u| \leq 1 \quad \rho_{zo}(u) = 1 \quad |u| > 1, \quad (6)$$

and it is relative easy to tune RANSAC to detect an inlier structure with constant scale.

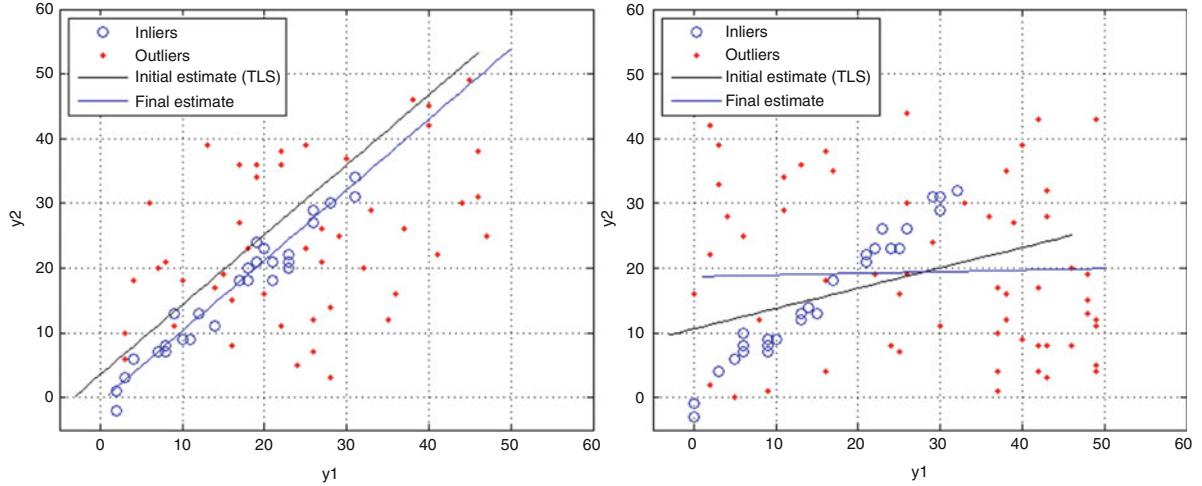
Before the start of the procedure, two parameters have to be provided to RANSAC. One is crucial – the scale of the inlier noise s , and one is approximate – the number of trials N . The value N should be several times the minimum number of trials computed for the lowest bound of the inlier/outlier ratio assumed in the problem. The number of trials increases quasi-exponentially with the size of the elemental subset. Depending on a specific problem, the number of trials can vary from a few hundreds to a few thousands.

The procedure is very simple.

- Repeat N times:
 - Choose an elemental subset by random sampling *without* replacement.
 - Find analytically, the corresponding model candidate pair $\hat{\boldsymbol{\Theta}}, \hat{\alpha}$.
 - Assuming that this model candidate is valid for the entire dataset of n points, compute the value of the robust objective function. The scale estimate s is used to separate the inliers from the outliers.
 - The model candidate yielding the smallest objective function value gives the estimate of RANSAC, the values $\hat{\boldsymbol{\Theta}}, \hat{\alpha}$.

Since the constraints in the *original* problem usually impose additional limits, the detected inliers are further processed with a generally nonlinear procedure. RANSAC is a universal flowchart valid for any differentiable or nondifferentiable objective function. For example, the non-robust least squares can also be computed this way but without closed-form analytical results.

Like any elemental subset-based estimation method, RANSAC does not guarantee to find the global optimum. If successful, RANSAC gives a good inlier/outlier separation which almost always is sufficient. This separation also depends on the complexity of the objective function. For example, the scalar objective function of the epipolar geometry may not be able to eliminate all the outliers from the region detected as inliers. Additionally it could fail when



Robust Methods, Fig. 1 Line fitting example using M-estimator. *Left:* the method works in case of symmetrical outliers. *Right:* it fails in the presence of unsymmetrical outliers

the data contains a lot of outliers, asymmetric noise, and/or multiple inlier structures. For example, Fig. 2 shows a simple line fitting example using RANSAC. Note that in both cases, the true value of the scale was provided. Also, traditional RANSAC cannot update the scale estimate in a video sequence with the value of the scale changing.

Over the past many years, several enhancements to the original RANSAC algorithm have been proposed. These enhancements fall into three main categories. The first kind deals with changes in the model verification. The second examines the way the sampling process generates hypotheses. The third category estimates from the data the scale of the inlier noise. The three types of enhancements were never attempted to improve RANSAC together.

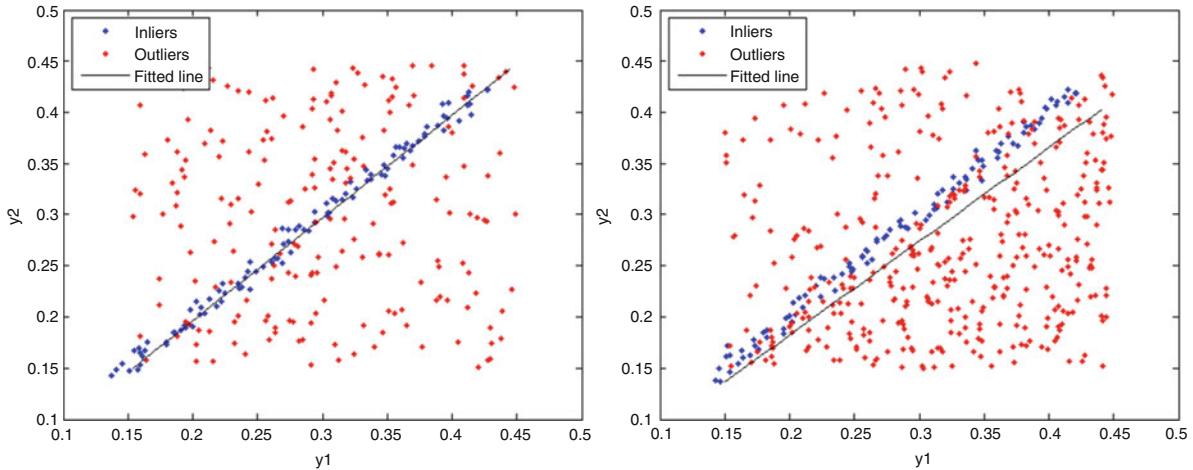
In *model verification*, Matas and Chum used the $T_{d,d}$ test to find first a small set of inliers which they then extended to a larger set. In the *bailout* test, Capel assumed that a random subset of the data has a hypergeometric distribution for the inliers and tried to find a new hypothesis with a larger number of inliers after verifying it with the current subset. The procedure *WaldSAC*, described by Chum and Matas, used Wald's theory of sequential decisions to reject hypotheses that have the likelihood ratio of points belonging to a good and bad hypothesis less than a certain threshold.

These methods work correctly only if certain assumptions are satisfied. For example, the $T_{d,d}$ test

may reject a correct hypothesis if the points in the initial subset are not all inliers, the bailout test could signal a false bailout if the lower bound on the number of inliers in the observed set of data points is not estimated correctly, and WaldSAC will be optimal only if the number of outliers are known a priori. Also, relative to the original RANSAC, for all these methods, often more parameters are required to be estimated from the data.

Two other methods were also developed that loosely belong to this category. In *preemptive* RANSAC, Nister started with a fixed number of hypotheses and used a breadth-first approach to evaluate and score all the hypotheses on a subset of the data in parallel. At each step, the hypotheses with best scores are evaluated on the new subset of data in the next iteration. As the method proceeds, the number of hypotheses decreases based on a preemption function, and the final estimate is returned by the procedure. In RANSAC for (quasi)-degenerate data (QDEGSAC) proposed by Frahm and Pollefeys, the method first detects possible degeneracies in the data and after that converges to the correct model.

Using *a priori* information introduces nonuniform sampling which can improve the quality of the generated hypotheses. In Progressive Sample Consensus (PROSAC), Chum and Matas used similarity scores of matching points to build a recurrence relation of the data points and used it to sample data points



Robust Methods, Fig. 2 Line fitting example using RANSAC. *Left:* the method works in case of fewer, symmetrical outliers. *Right:* it fails in the presence of a large number of unsymmetric outliers

nonuniformly thus converging to inliers faster. In *guided sampling*, Tordoff and Murray extended Torr and Zisserman's method of uniform sampling of the data points, called Maximum Likelihood Estimation Sample Consensus (MLESAC) [8]. They first computed matching probability of the feature points across two frames and then used these probabilities to sample the points nonuniformly. In locally optimized RANSAC (Lo-RANSAC), Chum et al. proposed generating a fixed number of *additional* hypotheses from the inliers already found by the algorithm and check against the whole dataset for improvement. Since the detected points are all inliers, there is no need to use the elemental subset for model estimation.

The reviews of all the algorithms above are given in [10]. Many of these variants are fast, and some can process data in real time. These methods are valid for the respective applications presented but will fail in general. Also, all these methods return only one inlier structure. Note that procedures which assume certain distribution for inliers or outliers will not be able to succeed in problems where the assumption may not hold.

The third category extracts, with additional processing, the *scale of inlier noise* from the data. The $k \times k$ scale matrix \mathbf{S} is diagonal and is computed element by element. Some methods order the residuals assuming a Gaussian distribution for the inliers and take the variance of the k -th sample as an estimate of the scale. A more advanced scalar estimate uses

$$s = n^{-1/5} \operatorname{med}_j |z_j - \operatorname{med}_i z_i|, \quad (7)$$

where z_i is the projection of data points on to one of the k dimensions of \mathbf{x}_i .

Note that Eq. (7) differs from median absolute deviation (MAD) only by $n^{-1/5}$ and is symmetric relative to the median. Wang and Suter [11] employed mean shift mode and mean shift valley techniques using over-smoothed kernel bandwidth to estimate the scale of the inlier noise assuming the noise to be Gaussian. In the projection-based M-Estimators (pbM), Subbarao and Meer [12] computed for each hypothesis Θ , a new scale using the estimate of Eq.(7). Using mean shift for the intercept, the pbM usually recovered the right estimate. These extinctions of RANSAC can recover three/four inlier structures but only under a quasi-symmetric outlier structures. In data with strongly asymmetric outliers, these methods will not work if there are more than one inlier structures.

The *generalized principal component analysis* (GPCA) introduced by Vidal et al. [13] is the last robust method that should be mentioned. The method can estimate an unknown number of inlier subspaces by analyzing the derivatives of polynomials obtained from data. Each of these subspaces can have different number of dimensions. The method is elegant, detects a few inlier structures in lower dimensions, but fails in the presence of unstructured outliers.

Application

In computer vision, robust methods have been applied to fundamental matrix estimation, trifocal tensor estimation, camera pose estimation and structure from motion. See [14] for references. Among all robust methods, RANSAC is topic of active research and newer enhancements to the original algorithm are proposed in all major vision conferences and journals. These papers will be judged positively only if the methods promise generalizability to large classes of problems involving images or image sequences.

Open Problems

A robust algorithm should be able to meet three fundamental goals in estimation. Firstly and most importantly, the scale of the inlier noise should be estimated automatically yet correctly, even if the outliers are strongly asymmetric. Secondly, the main estimation should retain from RANSAC the elemental subset-based hypothesize-and-test framework and from M-estimator the squared Mahalanobis distance with the diagonal scale matrix. Finally, the objective function should be chosen carefully taking into account the most general nature of the estimation problems with multiple inlier structures and small inlier to outlier ratios. Additionally, the heteroscedasticity of the linearized variables should be taken into consideration, and processing over Grassmann manifold using group theoretic methods should be an option [15].

2. Kanatani K, Morris D (2001) Gauges and gauge transformations for uncertainty description of geometric structure with indeterminacy. *IEEE Trans Inf Theory* 47:2017–2028
3. Subbarao R, Meer P (2009) Nonlinear mean shift over Riemannian manifolds. *Int J Comput Vis* 84:1–20
4. Huber PJ (1996) Robust statistical procedures, 2nd edn. SIAM, Philadelphia
5. Rousseeuw PJ, Leroy AM (1987) Robust regression and outlier detection. Wiley, New York
6. Meer P, Mintz D, Kim DY, Rosenfeld A (1991) Robust regression methods in computer vision: a review. *Int J Comput Vis* 6:59–70
7. Stewart CV (1999) Robust parameter estimation in computer vision. *SIAM Rev* 41:513–537
8. Meer P, Stewart CV, Tyler DE (eds) (2000) Robust statistical techniques in image understanding. Special issue, *Computer Vision and Image Understanding* vol 78. Academic Press San Diego
9. Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun ACM* 24(6):381–395
10. Raguram R, Frahm JM, Pollefeys M (2008) A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. In: *ECCV '08: proceedings of the 10th European conference on computer vision*, Springer, pp 500–513
11. Wang H, Suter D (2004) Robust adaptive-scale parametric model estimation for computer vision *IEEE Trans Pattern Anal Mach Intell* 26:1459–1474
12. Subbarao R, Meer P (2006) Beyond RANSAC: user independent robust regression. *Workshop on 25 years of RANSAC*, New York, NY
13. Vidal R, Ma Y, Sastry S (2005) Generalized principal component analysis (GPCA). *IEEE Trans Pattern Anal Mach Intell* 27:1945–1959
14. Hartley RI, Zisserman A (2004) Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, Cambridge
15. Mittal S, Anand S, Meer P (2012) Generalized Projection Based M-Estimator. *IEEE Trans Pattern Anal Mach Intell.*, 34:2351–2364

References

1. Comaniciu D, Meer P (2002) Mean shift: a robust approach toward feature space analysis. *IEEE Trans Pattern Anal Mach Intell* 24(5):603–619

Robust Regression

► Robust Methods

S

Saturation (Imaging)

Samuel W. Hasinoff
Google, Inc., Mountain View, CA, USA

Synonyms

Clipping

Related Concepts

► Radiometric Response Function

Definition

In imaging, saturation is a type of distortion where the recorded image is limited to some maximum value, interfering with the measurement of bright regions of the scene.

Background

The role of a sensor element is to measure incident irradiance and record that quantity as an image intensity value. However, physical constraints limit the maximum irradiance that can be measured for a given camera setting. In the absence of noise, the mapping from irradiance to image intensity is fully described by the *radiometric response function*, a monotonically

increasing function whose range is restricted by the maximum irradiance. Pixels whose intensity corresponds to this maximum are known as saturated.

Saturated pixels contain less information about the scene than other pixels. While non-saturated pixels can be related to the incident irradiance by applying the inverse of the radiometric response function, saturated pixels provide only a lower bound on irradiance. Therefore, estimating the irradiance of saturated pixels is similar to other image “hallucination” tasks such as inpainting [2].

Since many computer vision algorithms assume a linear relationship between sensor irradiance and the measured image intensity, it is important to identify saturated pixels and handle them appropriately. In practice, saturated pixels are often treated as missing values or otherwise ignored.

Theory

In the idealized noise-free case, the image intensity M of a pixel can be described as mapping the incident irradiance I according to the radiometric response function $f(\cdot)$, limited by the maximum irradiance I_{\max} ,

$$M = f(\min(I, I_{\max})). \quad (1)$$

For an irradiance of I_{\max} or higher, the image intensity will saturate at its maximum value of $M_{\max} = f(I_{\max})$. Since saturated pixels do not have unique corresponding irradiance values, they provide no direct information about incident irradiance beyond imposing a lower bound of I_{\max} .

Identifying saturated pixels is straightforward in practice, since many saturated pixels have the nominal maximum pixel value of M_{\max} . Sensor noise and other on-camera processing introduce the minor complication that saturated pixels may have values slightly less than this maximum. This effect is easily addressed, however, by using a lower, more conservative threshold to detect saturation [6, 8].

Saturation is caused by underlying physical characteristics of the sensor which limit the highest irradiance that can be measured for the given settings of the camera. In a digital sensor, where incident photoelectrons are recorded as electric charge, each sensor element can store a maximum amount of charge known as the *full well capacity*. Together with the exposure time and amplifier gain, the full well capacity imposes a limit on the maximum irradiance that can be measured before saturation. Film-based sensors are subject to saturation as well, but the mechanism limiting their photosensitivity is chemical [9].

While modern digital sensors are designed to dissipate excess charge above the full well capacity, for very bright parts of the scene, excess charge from a saturated pixel can spill over to adjacent regions. This artifact, known as *blooming*, can lead to saturation in pixels that would not otherwise be saturated.

Application

Saturation can pose problems for computer vision algorithms that assume linearity unless saturated pixels are identified and handled appropriately. For example, methods operating in the Fourier domain require special attention to saturation [1, 18], because the global nature of the transform means that even isolated saturated pixels can corrupt the whole image. The two main approaches to dealing with saturated pixels are explicitly treating them as missing values and interpolating them from surrounding pixels.

The effect of saturation should also be taken into account when estimating the parameters of sensor noise from an image [5, 11]. Pixels near saturation will demonstrate reduced variance in general, since the maximum value imposed by saturation will make their samples closer on average.

From the standpoint of photography, camera settings should be chosen to avoid saturation in

regions of interest, otherwise important detail or color information may be lost. Photographers describe saturated images as being *overexposed* or as having clipped or blown highlights. Although ill-posed, a problem of great practical interest to photographers is recovering detail in saturated regions of the scene or at least hallucinating plausible detail.

Under mild overexposure, only partial color information may be lost due to saturation. Partial saturation results from the different spectral sensitivities of each color channel, leading one channel to saturate before the others. In this setting, the main approach for restoring detail is to represent the correlation between color channels, using either global [19] or spatially varying [7, 12] color distribution models, then using this correlation to transfer information from the non-saturated color channels.

With greater overexposure, pixels become saturated in all color channels. The most common approach for restoring detail in this setting is to blindly extrapolate smooth peaks within saturated regions [7, 15, 17]. In fact, saturated regions can sometimes provide quantitative evidence about the underlying irradiance. Provided that overexposure is moderate and the scene is sufficiently smooth, the band-limitation of irradiance [1] or the resulting noise distribution [4] can be exploited to recover detail in fully saturated regions. For more severe overexposure or larger saturated regions, none of these methods are generally sufficient. In such cases, user guidance may be enlisted to help transfer plausible high-frequency detail from other sources [17].

In general, choosing the exposure setting for a photo requires balancing competing goals. While overexposure causes loss of detail in the highlights due to saturation, underexposure leads to higher relative noise. The relationship between noise and saturation defines the *dynamic range* of the sensor and determines the range of irradiances that can be captured acceptably in a single shot. When restricted to a single shot, one should generally choose the exposure setting so that the brightest region of interest falls just below the saturation point [14].

For scenes with large dynamic range, such considerations have motivated *high dynamic range imaging* methods based on capturing multiple photos with different exposure times [3], each of which saturates at a different irradiance. There is also an ongoing effort to develop new kinds of high dynamic range sensors

offering higher effective saturation levels [10]. A broad range of new designs have been proposed, including sensors that record the precise length of exposure time needed to reach saturation and sensors with a logarithm-like response. Each of these designs presents unique tradeoffs, including different noise characteristics over their operating range [10]. An orthogonal imaging approach is to use spatial multiplexing to incorporate multiple types of sensor elements, each having different sensitivities [13, 18] or sizes [16].

References

1. Abel JS, Smith JO (1991) Restoring a clipped signal. In: Proceedings of the IEEE international conference on acoustics, speech, and signal processing, Toronto, pp 1745–1748
2. Criminisi A, Pérez P, Toyama K (2003) Object removal by exemplar-based inpainting. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Madison, vol 2, pp 721–728
- 3.Debevec PE, Malik J (1997) Recovering high dynamic range radiance maps from photographs. In: Proceedings of the ACM SIGGRAPH, Los Angeles, pp 369–378
4. Foi A (2009) Clipped noisy images: heteroskedastic modeling and practical denoising. *Signal Process* 89(12): 2609–2629
5. Foi A, Trimeche M, Katkovnik V, Egiazarian K (2008) Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data. *IEEE Trans Image Process* 17(10):1737–1754
6. Granados M, Adjin B, Wand M, Theobalt C, Seidel H-P, Lensch HPA (2010) Optimal HDR reconstruction with linear digital cameras. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Francisco, pp 215–222
7. Guo D, Cheng Y, Zhuo S, Sim T (2010) Correcting overexposure in photographs. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Francisco, pp 515–521
8. Hasinoff SW, Durand F, Freeman WT (2010) Noise-optimal capture for high dynamic range photography. In: Proceedings IEEE conference on computer vision and pattern recognition (CVPR), San Francisco, pp 553–560
9. Hunt RWG (2004) The reproduction of colour, 6th edn. Wiley, Hoboken
10. Kavusi S, El Gamal A (2004) Quantitative study of high dynamic range image sensor architectures. In: Proceedings of the SPIE electronic imaging conference, San Jose, CA, vol 5301, pp 264–275
11. Liu C, Szeliski R, Kang SB, Zitnick CL, Freeman WT (2008) Automatic estimation and removal of noise from a single image. *IEEE Trans Pattern Anal Mach Intell* 30(2):299–314
12. Masood SZ, Zhu J, Tappen MF (2009) Automatic correction of saturated regions in photographs using cross-channel correlation. *Proc Pac Graph* 28:1861–1869
13. Narasimhan SG, Nayar SK (2005) Enhancing resolution along multiple imaging dimensions using assorted pixels. *IEEE Trans Pattern Anal Mach Intell* 27(4):518–530
14. Reichmann M (2003) <http://www.luminous-landscape.com/tutorials/expose-right.shtml>
15. Rempel AG, Trentacoste M, Seetzen H, Young HD, Heidrich W, Whitehead L, Ward G (2007) Ldr2hdr: on-the-fly reverse tone mapping of legacy video and photographs. *ACM Trans Graph* 26
16. Sugimoto M (2008) Digital camera. United States Patent 7,468,746, Dec 2008
17. Wang L, Wei L-Y, Zhou K, Guo B, Shum H-Y (2007) High dynamic range image hallucination. In: Proceedings of the eurographics symposium on rendering, Grenoble, pp 1–7
18. Wetzstein G, Ihrke I, Heidrich W (2010) Sensor saturation in fourier multiplexed imaging. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Francisco, pp 545–552
19. Zhang X, Brainard DH (2004) Estimation of saturated pixel values in digital color imaging. *J Opt Soc Am A* 21(12):2301–2310

Scale Selection

Tony Lindeberg

School of Computer Science and Communication,
KTH Royal Institute of Technology, Stockholm,
Sweden

Synonyms

Automatic scale selection; Scale-invariant image features and image descriptors

Related Concepts

► [Corner Detection](#); ► [Edge Detection](#)

Definition

The notion of scale selection refers to methods for estimating characteristic scales in image data and for automatically determining locally appropriate scales in a scale-space representation, so as to adapt subsequent processing to the local image structure and compute scale-invariant image features and image descriptors.

An essential aspect of the approach is that it allows for a bottom-up determination of inherent scales of features and objects without first recognizing them or

delimiting, alternatively segmenting, them from their surroundings.

Scale selection methods have also been developed from other viewpoints of performing noise suppression and exploring top-down information.

(with multi-index notation $\alpha = (\alpha_1, \dots, \alpha_N)$ for $\partial_{x^\alpha} = \partial_{x_1^{\alpha_1}} \dots \partial_{x_N^{\alpha_N}}$). Such Gaussian derivatives can be used as a basis for expressing a large number of visual modules including feature detection, feature classification, image matching, motion, shape cues, and image-based recognition [14].

Background

The concept of *scale* is essential when computing features and descriptors from image data. Real-world objects may contain different types of structures at different scales and may therefore appear in different ways depending on the scale of observation. When observing objects by a camera or an eye, there is an additional scale problem due to perspective effects, implying that distant objects will appear smaller than nearby objects. A vision system intended to operate autonomously on image data acquired from a complex environment must therefore be able to handle and be robust to such scale variations.

For a vision system that observes an unknown scene, there is usually no way to a priori know what scales are appropriate for extracting the relevant information. Hence, a multi-scale representation of the image data is essential, whereby the original signal is embedded into a *one-parameter family* of signals using scale as the parameter. Given an N -dimensional signal $f: \mathbb{R}^N \rightarrow \mathbb{R}$ and with the notation $x = (x_1, \dots, x_N) \in \mathbb{R}^N$, the *scale-space representation* [8, 10, 26] of f is defined by the convolution operation

$$L(x; t) = \int_{\xi \in \mathbb{R}^N} f(x - \xi) g(\xi; t) d\xi, \quad (1)$$

where $g: \mathbb{R}^N \times \mathbb{R}_+ \rightarrow \mathbb{R}$ denotes the *Gaussian kernel*

$$g(x; t) = \frac{1}{(2\pi t)^{N/2}} e^{-|x|^2/2t} \quad (2)$$

and the variance $t = \sigma^2$ of this kernel is referred to as the *scale parameter*. Based on this representation, *Gaussian derivatives*, or *scale-space derivatives*, at any scale t can then be computed by differentiating the scale-space representation or equivalently by convolving the original image with Gaussian derivative kernels:

$$L_{x^\alpha}(\cdot; t) = \partial_{x^\alpha} L(\cdot; t) = (\partial_{x^\alpha} g(\cdot; t)) * f(\cdot) \quad (3)$$

Theory

The notion of scale selection complements traditional *scale-space theory* by providing explicit mechanisms for generating hypotheses about interesting scales.

Scale Selection from γ -Normalized Derivatives

A particularly useful methodology for computing estimates of *characteristic scales* is by detecting *local extrema over scales* of differential expressions in terms of γ -normalized derivatives [11, 12] defined by

$$\partial_\xi = t^{\gamma/2} \partial_x. \quad (4)$$

A general and very useful property of this construction is that if two signals f and f' are related by a scaling transformation

$$f'(x') = f(x) \quad \text{with} \quad x' = s x, \quad (5)$$

and if there is a local extremum over scales at $(x_0; t_0)$ in a differential expression $\mathcal{D}_{\gamma-norm} L$ defined as a homogeneous polynomial of Gaussian derivatives computed from the scale-space representation L of the original signal f , then there will be a corresponding local extremum over scales at $(x'_0; t'_0) = (s x_0; s^2 t_0)$ in the corresponding differential expression $\mathcal{D}_{\gamma-norm} L'$ computed from the scale-space representation L' of the rescaled signal f' [11, Sect. 4.1].

This scaling result holds for all homogeneous polynomial differential expressions, including rotationally invariant differential invariants, and implies that local extrema over scales of γ -normalized derivatives are preserved under scaling transformations. Thereby, such local extrema over scales provide a theoretically well-founded way to automatically adapt the scale levels to local scale variations.

Specifically, scale-normalized scale-space derivatives of order $|\alpha| = \alpha_1 + \dots + \alpha_N$ at corresponding

points will then be related according to

$$L'_{\xi'^{\alpha}}(x'; t') = s^{|\alpha|(\gamma-1)} L_{\xi^{\alpha}}(x; t), \quad (6)$$

which means that $\gamma = 1$ implies perfect scale invariance in the sense that the γ -normalized derivatives at corresponding points will be equal. If $\gamma \neq 1$, the difference in magnitude can on the other hand be easily compensated for using the scale values of corresponding scale-adaptive image features.

These results imply that detection of image features and computation of image descriptors at scale levels equal to or proportional to the scales at which there are local extrema over scales constitutes a very general methodology for obtaining *scale-invariant image features* and *scale-invariant image descriptors*.

Indeed, it can also be axiomatically shown that the notion of γ -normalized derivatives arises by necessity, given the condition that local extrema over scales of scale-normalized derivatives should be preserved under scaling transformations [11, Appendix A.1].

Relation to Frequency Estimation

There is a conceptual similarity between this principle and local frequency estimation from peaks in the Fourier transform. For a one-dimensional sine wave

$$f(x) = \sin(\omega x), \quad (7)$$

it can be shown [11, Sect. 3] that there will be a peak in the magnitude of the m th order γ -normalized derivative at a scale

$$\sigma_{\max} = \frac{\sqrt{\gamma m}}{2\pi} \lambda \quad (8)$$

proportional to the wavelength $\lambda = 2\pi/|\omega|$ of the signal. Two conceptual differences compared to Fourier-based frequency estimation, however, are that (i) no window size is needed for computing the Fourier transform and (ii) this approach applies also to nonlinear differential expressions.

Relations to Image Statistics

It can be shown [11, Sect. 9.1] that the notion of γ -normalized derivatives corresponds to normalizing the m th order N -dimensional Gaussian derivatives to

constant L_p -norms over scale with

$$p = \frac{1}{1 + \frac{m}{N}(1 - \gamma)}, \quad (9)$$

where the perfectly scale-invariant case $\gamma = 1$ corresponds to L_1 -normalization for all orders m .

It can also be shown [11, Sect. 9.2] that the γ -normalized derivatives are *neutral* with respect to *self-similar* power spectra of the form

$$S_f(\omega) = |\omega|^{-N-2m(1-\gamma)}. \quad (10)$$

Natural images often show a qualitative behavior similar to this [5].

Scale-Space Signatures

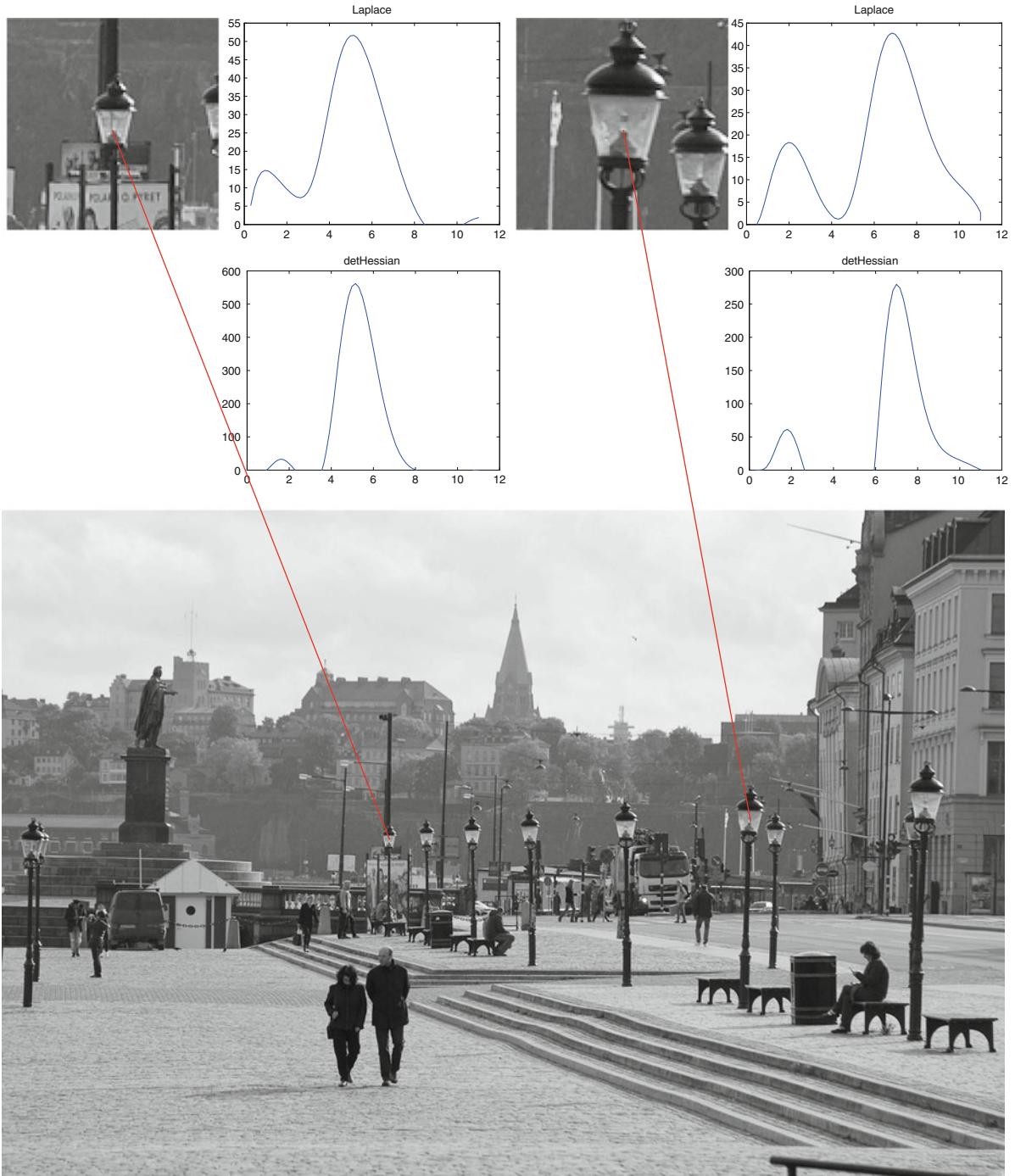
Figure 1 illustrates the basic idea, by showing the so-called *scale-space signatures* accumulated in the two-dimensional case (In the specific 2-D case, the simplifying notation $(x, y) \in \mathbb{R}^2$ is used instead of $x = (x_1, x_2) \in \mathbb{R}_2$, implying that $L_{x_1 x_1} = L_{xx}$, $L_{x_1 x_2} = L_{xy}$, $L_{x_2 x_2} = L_{yy}$, etc.) for two generally applicable differential entities for scale selection: the *scale-normalized Laplacian* [10, Sect. 13.3] [11, Sect. 5] (with $\gamma = 1$)

$$\nabla^2 L_{norm} = t(L_{xx} + L_{yy}) \quad (11)$$

and the *scale-normalized determinant of the Hessian* [10, 11] (also with $\gamma = 1$)

$$\det \mathcal{H}_{norm} L = t^2 (L_{xx} L_{yy} - L_{xy}^2). \quad (12)$$

In the scene in **Fig. 1**, there are strong perspective scaling effects due to differences in depth between similar objects in the world. These scale variations are reflected in the scale-space signatures in the respect that the local extrema over scales are assumed at finer scales for distant objects and at coarser scales for nearby objects. If one computes the ratio between the scale values in terms of a scale parameter $\sigma = \sqrt{t}$ of dimension [length], then the ratio between the scale values is in very good agreement with the ratio between the sizes of the objects in the image domain as measured by a ruler. This property illustrates one of the scale-invariant properties of the scale selection mechanism.



Scale Selection, Fig. 1 Scale-space signatures accumulated for image structures of different size in the image domain. The *upper part* of the illustration shows windows around two details from the image at the *bottom*, with corresponding scale-space signatures of the scale-normalized Laplacian $\nabla_{norm}^2 L$ and the scale-normalized determinant of the Hessian $\det \mathcal{H}_{norm} L$ accumulated at the central point in each window. As can be seen from the graphs, the local extrema over scales are assumed at coarser

scales for the larger size object than for the smaller size object. Specifically, the ratio between the scale values at which the local extrema are assumed provides an estimate of the relative amount of scaling, when measured in dimension [length] (In the graphs, the horizontal axis represents effective scale [10, pp 180–182] approximated by $\tau \approx \log_2(1 + t)$) (Reprinted from [27] with permission)

General Framework for Defining Scale-Invariant Image Descriptors

By computing an image descriptor at a scale proportional to the detection scale \hat{t} of a scale-invariant image feature or by normalizing an image patch by a corresponding scaling factor $\hat{\sigma} = \sqrt{\hat{t}}$ provides two very general scale normalization mechanisms that can be used for defining much wider classes of *scale-invariant image descriptors* [11, 15] (see the “Applications” section below for two specific examples regarding image-based recognition). The scale-invariant properties of these descriptors originate from the general scale-invariant property of local extrema over scales of differential expressions in terms of γ -normalized derivatives.

[Figure 2](#) illustrates how scale normalization can be performed in this way by rescaling the local image patches around the two details in [Fig. 1](#) using the scale values $\hat{\sigma} = \sqrt{\hat{t}}$ at which the Laplacian $\nabla_{norm}^2 L$ and the determinant of the Hessian, respectively, assumed their strongest local extrema over scales. In this sense, scale normalization from the detection scales \hat{t} constitutes a general mechanism for establishing a *common scale-invariant reference frame* with regard to scaling transformations.

It should be noted, however, that multiple extrema over scales may in general be found in the scale-space signature, as can be seen in [Figs. 1](#) and [2](#), where two significant local extrema over scales are obtained in each scale-space signature, with the coarser-scale response corresponding to the lamp as a whole and the finer-scale response corresponding to the light bulb inside. Because of this inherent multi-scale nature of real-world objects, a vision system intended to interpret images from a natural environment must in general be able to handle *multiple scale hypotheses over scales*.

Scale-Space Extrema

The notion of scale selection from scale-normalized derivatives can be complemented by *spatial selection* by detecting points in scale space that assume *local extrema with respect to both space x and scale t* . Such points are referred to as *scale-space extrema*. Specifically, detection of scale-space extrema of rotationally invariant differential invariants provides a general, effective, and robust methodology for detecting *interest points with built-in scale selection* [10, 11, 15].

Thus, given a scale-normalized differential expression $\mathcal{D}_{\gamma-norm} L$, one simultaneously obtains spatial positions \hat{x} and scale estimates \hat{t} according to

$$(\hat{x}; \hat{t}) = \operatorname{argmaxminlocal} (\mathcal{D}_{\gamma-norm} L)(x; t). \quad (13)$$

[Figure 3](#) shows the result of detecting the 50 strongest scale-space extrema of the scale-normalized Laplacian $\nabla_{norm}^2 L$ and the scale-normalized determinant of the Hessian $\det \mathcal{H}_{norm} L$ from an image that contains two objects of different sizes. Each scale-space extremum has been illustrated by a circle with the radius proportional to the detection scale $\hat{\sigma} = \sqrt{\hat{t}}$. In [Fig. 4](#), each feature has been visualized by a sphere in the 3-D scale-space volume of the 2-D image, with the radius of the sphere increasing with the detection scale. As can be seen from this illustration, the notion of scale-space extrema can effectively reveal interest points and characteristic scales of those (see the “Applications” section below for more details about scale-invariant interest point detectors). Specifically, the differences in the radii of the circles in the 2-D illustration and in the heights over the image plane in the 3-D graphics reveal the scale differences between corresponding image features from the two objects.

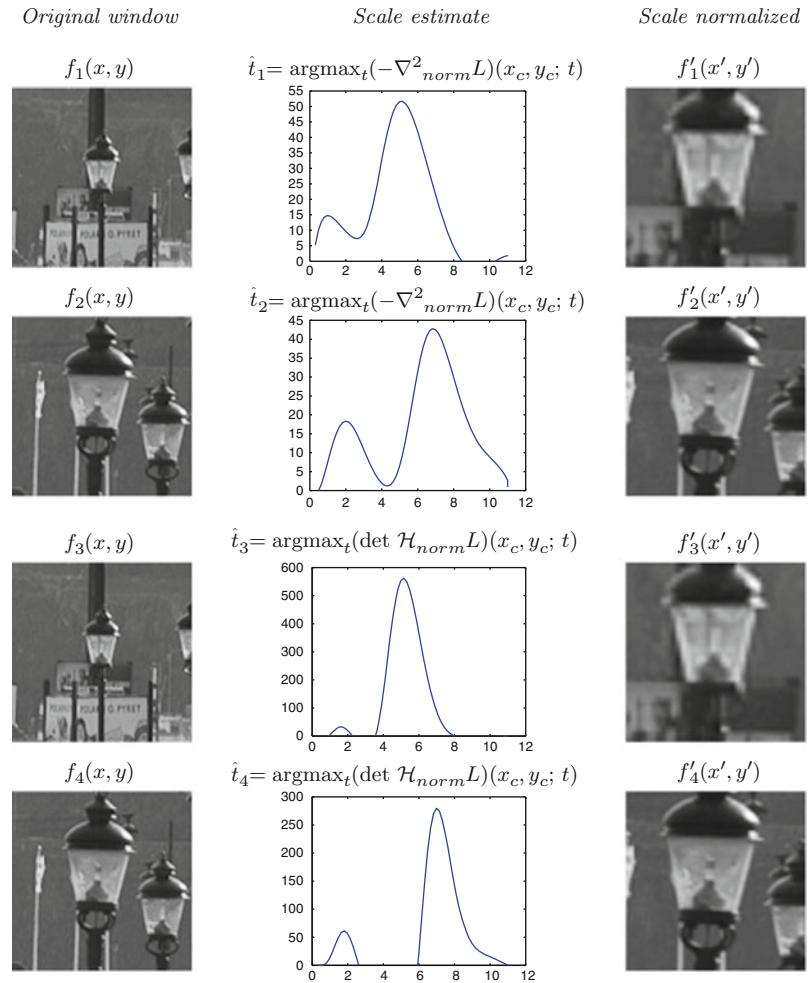
The differential operators $\nabla_{norm}^2 L$ and $\det \mathcal{H}_{norm} L$ in general both produce strong responses at the centers of blob-like structures that are either brighter or darker than their surrounding, provided that the these differential entities are computed at scale levels that roughly match the size of the corresponding image structures. For this reason, they constitute very useful differential entities for *blob detection*.

Discrete Implementation

Detection of scale-space extrema from an N -dimensional discrete image can be performed by nearest-neighbor comparisons in the $N + 1$ -dimensional scale-space volume. For a 2-D image, this implies that nearest-neighbor comparisons are performed by local comparisons with the 26 neighbors in a $3 \times 3 \times 3$ neighborhood over space and scale [10] [11, footnote 16] [20]. Scale estimates and position estimates of higher accuracy can then be obtained by fitting a parabola to the data around any scale-space extremum [17, 20].

Discrete analogues of γ -normalized derivatives can be obtained either by (i) *variance-based normalization*

Scale Selection, Fig. 2 In this illustration, local windows around two of the lamps in Fig. 1 (shown in the left column) have been rescaled by scaling factors $\hat{\sigma} = \sqrt{\hat{t}}$ obtained from the dominant response over scales of the Laplacian $\nabla_{norm}^2 L$ and the determinant of the Hessian $\det \mathcal{H}_{norm} L$ (shown in the middle column) to compute a scale-normalized window (shown in the right column) around each detail. In this way, scale selection can be used for defining a scale-normalized reference frame for subsequent computation of scale-invariant image descriptors (Reprinted from [27] with permission)



which implies that the discrete derivative approximations $\delta_{x^\alpha} L$ are multiplied by an appropriate power of the scale parameter

$$L_{\xi^\alpha}(\cdot; t) = L_{\xi_1^{\alpha_1} \dots \xi_N^{\alpha_N}}(\cdot; t) = (t^{\gamma/2})^{|\alpha|} (\delta_{x^\alpha} L)(\cdot; t) \quad (14)$$

or by using the notion of (ii) l_p -normalization [11, Appendix A.4.2]:

$$L_{\xi^\alpha}(\cdot; t) = L_{\xi_1^{\alpha_1} \dots \xi_N^{\alpha_N}}(\cdot; t) = C_\alpha (\delta_{x^\alpha} L)(\cdot; t), \quad (15)$$

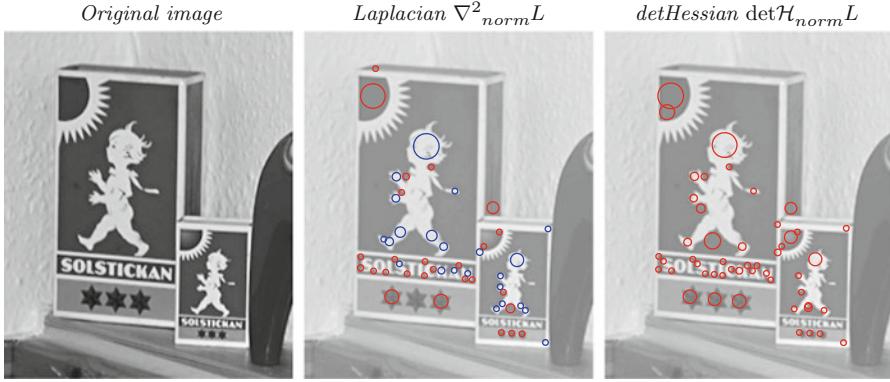
where the discrete normalization constants C_α are determined such that the l_p -norms of the scale-normalized discrete derivative approximation kernels $\delta_{x^\alpha} T$ [10, Chap. 5] are to be equal to the L_p -norms of the corresponding γ -normalized Gaussian derivative kernels $\partial_{x^\alpha} g$

$$C_\alpha \left(\sum_{n \in \mathbb{R}^N} |(\delta_{x^\alpha} T)(n; t)|^p \right)^{1/p} = (t^{\gamma/2})^{|\alpha|} \left(\int_{x \in \mathbb{R}^N} |(\partial_{x^\alpha} g)(x; t)|^p dx \right)^{1/p}. \quad (16)$$

Experiments in [17] show that the notion of l_p -normalization gives more accurate scale estimates in situations where discretization effects become important.

A particularly convenient way of implementing scale-space smoothing in this context is by convolution with the *discrete analogue of the Gaussian kernel* [10, pp 84–87]:

$$T(n; t) = e^{-t} I_n(t), \quad (17)$$



Scale Selection, Fig. 3 2-D illustration of the 50 strongest scale-space extrema of the Laplacian $\nabla^2_{norm} L$ and the determinant of the Hessian $\det \mathcal{H}_{norm} L$ computed from an image with two similar objects of different physical sizes. Each feature is

illustrated by a circle centered at the position (\hat{x}, \hat{y}) of the scale-space extremum and with the radius proportional to the detection scale $\hat{\sigma} = \sqrt{\hat{t}}$. Red circles represent scale-space maxima, while blue circles represent scale-space minima

which implies that semigroup property of the Gaussian scale space holds exactly also for the discrete scale-space kernels $T(\cdot; t_1) * T(\cdot; t_2) = T(\cdot; t_1 + t_2)$ and the *cascade smoothing property*

$$L(\cdot; t_2) = T(\cdot; t_2 - t_1) * L(\cdot; t_1), \quad (18)$$

for $t_2 \geq t_1 \geq 0$ implies that one can perform a set of incremental convolutions with kernels of smaller support instead of computing each scale level from the original signal f independently.

The notion of scale selection from scale-normalized derivatives can also be transferred to a *pyramid* representation to allow for real-time implementation on standard processors [3, 17, 20].

Alternative Approaches to Scale Selection

A number of other mechanisms for scale selection have also been developed based on ideas of

- Detecting peaks over scales in weighted entropy measures [7] or Lyapunov functionals [25]
- Minimizing normalized error measures over scales in order to compute more accurate localization estimates for coarser-scale corner features [11, Sect. 7.2] or for coarse-to-fine matching of highly noisy image data [13]
- Determining minimum reliable scales for feature detection according to an a priori determined noise suppression model [4]
- Determining optimal stopping times in nonlinear diffusion-based image restoration methods using

similarity measurements relative to the original data [22]

- Performing image segmentation from the scales at which a supervised classifier delivers class labels with the highest posterior [18, 19]

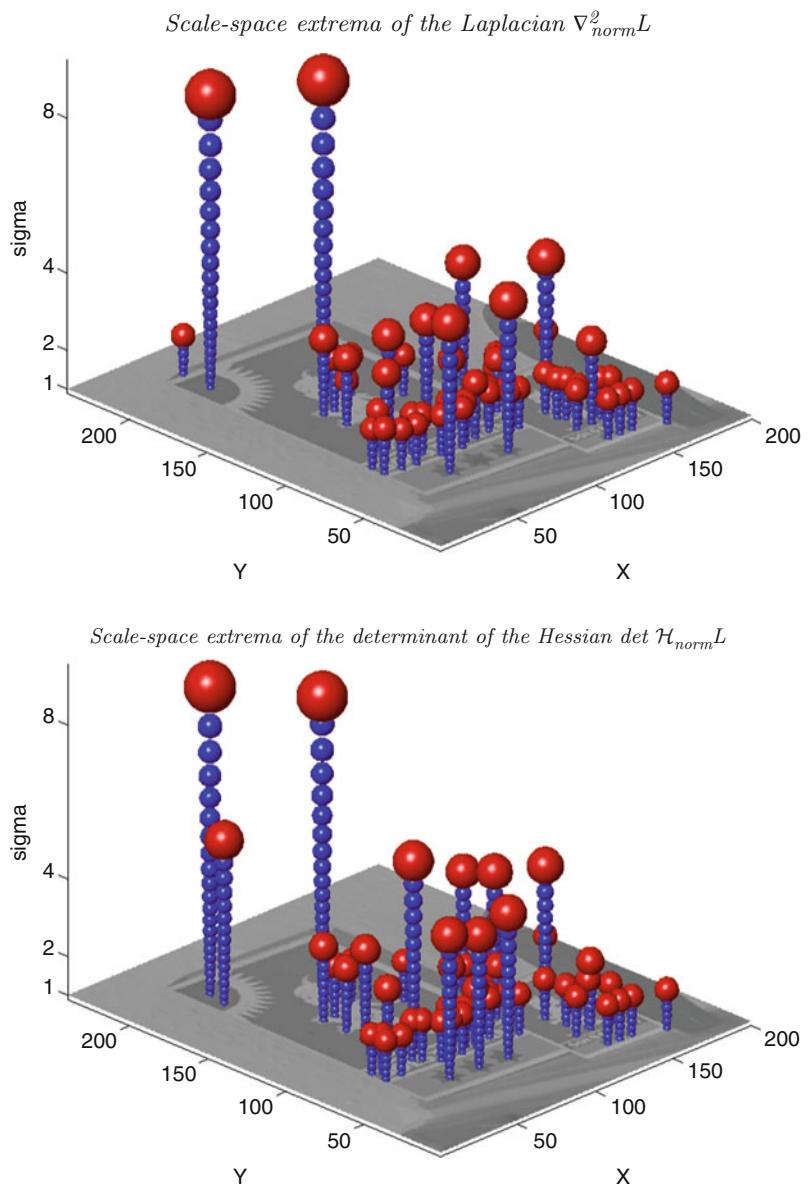
Relations Between the Different Approaches to Scale Selection

The different approaches to scale selection may have quite different properties, depending on the types of data they are applied to. For noise-free data, an adaptive noise suppression scheme optimized for suppressing high-frequency noise can be expected to not smooth the data at all, thus implying the selection of a zero scale, whereas scale selection based on local extrema over scales will always select a scale level reflecting a characteristic length in the image data.

Provided that the characteristic lengths of the relevant image features are greater than the typical characteristic lengths in the noise, scale selection based on scale-normalized derivatives will therefore lead to scale-invariant image features. Smoothing approaches that are optimized for suppressing superimposed high-frequency noise will on the other hand lead to an amount of smoothing that is primarily determined by the noise level and therefore not necessarily corresponding to scale-invariant image descriptors. In this respect, these two types of scale determination approaches can lead to fundamentally different results.

If the task is to detect fine-scale details with amplitude and/or characteristic scales comparable to the

Scale Selection, Fig. 4 3-D illustration of the 50 strongest scale-space extrema of the Laplacian $\nabla_{norm}^2 L$ and the determinant of the Hessian $\det \mathcal{H}_{norm} L$ computed from the image in Fig. 4. Here, each feature is illustrated by a red sphere centered at the position $(\hat{x}_0, \hat{y}_0; \hat{t}_0)$ of the scale-space extremum and with the radius increasing with the detection scale \hat{t}_0 . The blue spheres have been inserted to simplify visual interpretation



noise, it does, however, not seem unlikely that the two types of approaches could possibly benefit from each other.

according to a similar methodology can be found in [15] with an in-depth theoretical analysis of their scale selection properties in [16].

Application

Interest Point Detectors with Built-in Scale Selection

Below, four different interest point detectors with automatic scale selection will be presented. A more general set of scale-invariant interest point detectors defined

Blob Detection

Based on the notion of scale-space extrema, straightforward methods for blob detection can be obtained by detecting scale-space extrema of either (i) the scale-normalized Laplacian $\nabla_{norm}^2 L = t(L_{xx} + L_{yy})$ or (ii) the scale-normalized determinant of the Hessian $\det \mathcal{H}_{norm} L = t^2(L_{xx}L_{yy} - L_{xy}^2)$ [10, 11]. Specifically, using the Laplacian operator one can detect

- Bright blobs from negative scale-space minima of $\nabla_{norm}^2 L$
- Dark blobs from positive scale-space maxima of $\nabla_{norm}^2 L$

Using the determinant of the Hessian, one can on the other hand detect

- Bright blobs from positive scale-space maxima of $\det \mathcal{H}_{norm} L$ that satisfy $\nabla^2 L < 0$
- Dark blobs from positive scale-space maxima of $\det \mathcal{H}_{norm} L$ that satisfy $\nabla^2 L > 0$
- Saddle-like image features from negative scale-space minima of $\det \mathcal{H}_{norm} L$

These two blob detection approaches do both satisfy the basic scale selection property that if the scale-adaptive blob detector is applied to a two-dimensional Gaussian blob with scale value t_0 , i.e., $f(x, y) = g(x, y; t_0)$, then the select scale \hat{t} will be equal to the scale of the blob in the input data, i.e., $\hat{t} = t_0$.

In comparison, the image features obtained from the determinant of the Hessian blob detector do often have better repeatability properties under affine image deformations than Laplacian image features [11, 15, 16].

Figure 5 shows the result of applying these interest point detectors to a grey-level image. Please note how the variations in the detection scales of the blob responses reflect the perspective scaling effects in the scene.

Corner Detection

A straightforward method for scale-invariant corner detection can be obtained by detecting positive scale-space maxima and negative scale-space minima of the scale-normalized *rescaled level curve curvature* measure

$$\tilde{\kappa}(L) = t^{2\gamma} |\nabla L|^2 \kappa(L) = t^{2\gamma} \left(L_x^2 L_{yy} + L_y^2 L_{xx} - 2L_x L_y L_{xy} \right), \quad (19)$$

where $\kappa(L)$ denotes the curvature of the level curves of the Gaussian smoothed image at any scale and $\gamma = 7/8$ turns out to be a good choice [11, Sect. 6] [15]; see Fig. 6a for an illustration.

The *Harris-Laplace* operator [21] is structurally different in the respect that it uses different entities for spatial selection (the Harris measure) and scale selection ($\nabla_{norm}^2 L$); see Fig. 6b.

Edge Detection

With regard to edge detection, the evolution properties over scales of the *scale-normalized gradient magnitude*

$$|\nabla L|_{norm} = t^{\gamma/2} \sqrt{L_x^2 + L_y^2} \quad (20)$$

can be shown to reveal local characteristics of the type of edge [12, Sect. 4]. Specifically, by choosing $\gamma = 1/2$, a local maximum over scales will be assumed at a scale corresponding to the diffuseness of a one-dimensional diffuse step edge

$$\Phi(x; t_0) = \int_{u=-\infty}^x g(u; t_0) du \quad (21)$$

and may then provide cues to, e.g., focus blur, shadow edges, or rounded edges.

Ridge and Valley Detection

Let e_p and e_q denote the eigendirections of the Hessian matrix $\mathcal{H}L$ such that the mixed second-order derivative in this coordinate frame is zero $L_{pq} = 0$ and denote the eigenvalues of the Hessian matrix by L_{pp} and L_{qq} . These eigenvalues are also referred to as *principal curvatures*, and these directions are assumed to be ordered such that $L_{pp} < L_{qq}$.

Then, a differential geometric definition of the *ridges* in the image at any scale can be expressed as the set of points that satisfy [12, Sect. 5.2]

$$L_p = 0, \quad L_{pp} \leq 0, \quad |L_{pp}| \geq |L_{qq}|. \quad (22)$$

Similarly, the *valleys* at any scale can be defined from [14]

$$L_q = 0, \quad L_{qq} \geq 0, \quad |L_{qq}| \geq |L_{pp}|. \quad (23)$$

With $R_{\gamma-norm}$ denoting a scale-normalized measure of *ridge strength* (or valley strength) defined from the principal curvatures L_{pp} and L_{qq} , one can also express ridge and valley detection methods with automatic scale selection by detecting *scale-space ridges* using the definition

$$L_p = 0, \quad L_{pp} \leq 0, \quad \partial_t(R_{\gamma-norm}) = 0, \\ \partial_{tt}(R_{\gamma-norm}) \leq 0 \quad (24)$$

Scale Selection, Fig. 5

Scale-invariant interest points obtained from the 1,000 strongest scale-space extrema of the Laplacian $\nabla^2_{norm} L$ and the determinant of the Hessian $\det \mathcal{H}_{norm} L$ with the size of each circle reflecting the detection scale of the corresponding feature. Red circles represent local maxima of the operator response, while blue circles indicate local minima (Adapted from [15])



and *scale-space valleys* according to

$$\begin{aligned} L_q = 0, \quad L_{qq} \geq 0, \quad \partial_t(R_{\gamma-norm}) = 0, \\ \partial_{tt}(R_{\gamma-norm}) \leq 0. \end{aligned} \quad (25)$$

Specifically, it can be shown that for natural measures of ridge or valley strength, the choice $\gamma = 3/4$ implies

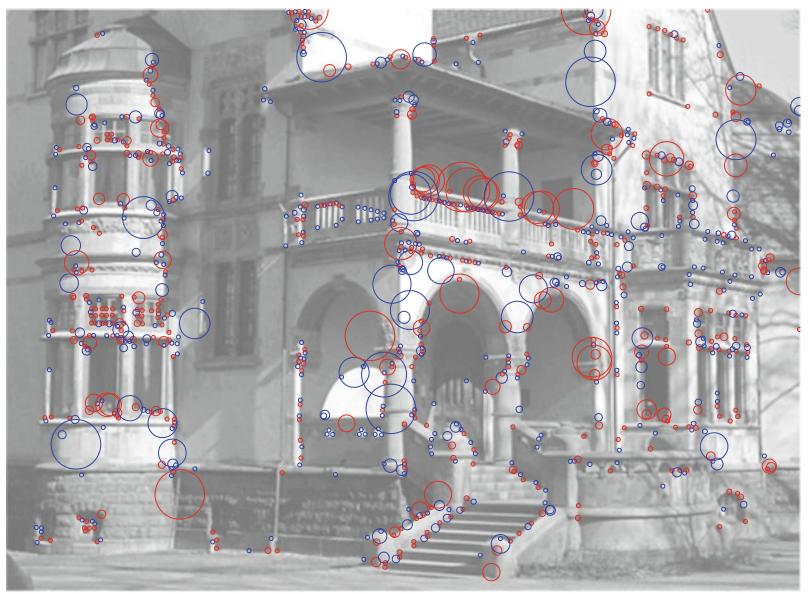
that the selected scale will reflect the width of a Gaussian ridge (or valley) [12]. For generalizations to 3-D images, see [6, 9, 24].

Feature Tracking

By adapting the scales for feature detection by a local scale selection mechanism, the resulting image features will be robust to scale changes, which means that they can be matched over substantial

Scale Selection, Fig. 6

Scale-invariant interest points obtained from the 1,000 strongest scale-space extrema of the rescaled level curve curvature $\tilde{\kappa}(L)$ and the Harris-Laplace operator. The size of each circle reflects the detection scale of the corresponding feature. For the rescaled level curve curvature operator $\tilde{\kappa}(L)$, the color of the circles show the sign of the curvature; red circles represent a local maxima of the operator response, while blue circles indicate local minima (Adapted from [15])

Scale-space extrema of the rescaled level curve curvature $\tilde{\kappa}(L)$ 

The Harris-Laplace operator



size variations [2]. Indeed, the variations over time in the characteristic scale estimates obtained during feature tracking can, if appropriately implemented, be robust enough for computing estimates of *time to collision* [17].

Image-Based Matching and Recognition

The SIFT descriptor [20] comprises a bottom-up keypoint detection stage with scale-space extrema

detection in a differences-of-Gaussians (DoG) pyramid. The scale-invariant properties of the SIFT descriptor can be explained as follows.

From the way that the DoG operator is implemented in the pyramid in [20], it follows that the normalization will be similar to the scale-normalized Laplacian. Using the fact that the scale-space representation satisfies the diffusion equation, it follows that the Laplacian operator can be approximated from the difference between two levels in the scale-space representation:

$$\begin{aligned} \frac{1}{2} \nabla^2 L(x, y; t) &= \partial_t L(x, y; t) \\ &\approx \frac{L(x, y; t + \Delta t) - L(x, y; t)}{\Delta t} \\ &= \frac{DOG(x, y; t, \Delta t)}{\Delta t}, \end{aligned} \quad (26)$$

i.e., from the difference of two Gaussian smoothed images.

With the scale levels distributed such that the ratio between successive scale levels is k when measured in terms of $\sigma = \sqrt{t}$, (i.e., $\sigma_{i+1} = k \sigma_i$ and $t_{i+1} = k^2 t_i$ which implies that $\Delta t_i = (k^2 - 1) t_i$), it follows that [15]

$$\begin{aligned} DOG(x, y; t) &= L(x, y; k^2 t) - L(x, y; t) \\ &\approx (k^2 - 1) t (\partial_t L(x, y; t)) = (k^2 - 1) t \frac{1}{2} \nabla^2 \\ L(x, y; t) &= \frac{(k^2 - 1)}{2} t \nabla^2 L(x, y; t) \\ &= \frac{(k^2 - 1)}{2} \nabla_{norm}^2 L(x, y; t). \end{aligned} \quad (27)$$

Hence, with self-similar sampling of the scale levels, the pyramid-implemented DoG interest point operator can be interpreted as an approximation of the scale-adapted Laplacian operator in Eq. (11).

In the SURF descriptor [1], local feature detection is performed by detecting local extrema over space and scale of an approximation of the determinant of the Hessian operator in terms of Haar wavelets, with the filters normalized to constant l_1 - or Frobenius norm over scales. According to Eq. (9), the γ -normalized derivative concept corresponds to normalization of the Gaussian derivative operators to unit L_p -norm over scales. Furthermore, it was shown in [17] that normalizing the filter responses to constant l_p -norm over scales gives better accuracy in a practical implementation than normalization of the discrete filters by multiplication with the scale parameter raised to a power of $m\gamma/2$, where m denotes the order of differentiation. Hence, the initial feature detection step in the SURF descriptor can be seen as an approximation of the scale-normalized determinant of the Hessian operator in Eq. (12).

The scale-invariant property of the actual image descriptors in the SIFT and SURF descriptors does in turn follow from the scale-invariant properties of

the initial feature detection step, in line with the general framework for computing scale-invariant image descriptors from scale estimates obtained from local extrema over scales of scale-normalized differential expressions, as described in the “Theory” section above.

In these ways, the notion of scale selection constitutes a general mechanism for computing scale-invariant image descriptors for image-based matching and recognition.

References

1. Bay H, Ess A, Tuytelaars T, van Gool (2008) Speeded up robust features (SURF). *Comput Vis Image Underst* 110(3):346–359
2. Bretzner L, Lindeberg T (1998) Feature tracking with automatic selection of spatial scales. *Comput Vis Image Underst* 71(3):385–392
3. Crowley J, Riff O (2003) Fast computation of scale normalised receptive fields. In: Proceedings of scale-space’03. Lecture Notes in Computer Science, vol 2695. Springer, Berlin/New York, pp 584–598
4. Elder JH, Zucker SW (1998) Local scale control for edge detection and blur estimation. *IEEE Trans Pattern Anal Mach Intell* 20(7):699–716
5. Field DJ (1987) Relations between the statistics of natural images and the response properties of cortical cells. *J Opt Soc Am* 4:2379–2394
6. Frangi AF, Niessen WJ, Hoogeveen RM, vanWalsum T, Viergever MA (2000) Model-based quantitation of 3D magnetic resonance angiographic images. *IEEE Trans Med Imaging* 18(10):946–956
7. Kadir T, Brady M (2001) Saliency, scale and image description. *Int J Comput Vis* 45(2):83–105
8. Koenderink JJ (1984) The structure of images. *Biol Cybern* 50:363–370
9. Krissian K, Malandain G, Ayache N, Vaillant R, Trouset Y (2000) Model-based detection of tubular structures in 3D images. *Comput Vis Image Underst* 80(2):130–171
10. Lindeberg T (1994) Scale-space theory in computer vision. The Springer international series in engineering and computer science. Springer, Berlin/New York
11. Lindeberg T (1998) Feature detection with automatic scale selection. *Int J Comput Vis* 30(2):77–116
12. Lindeberg T (1998) Edge detection and ridge detection with automatic scale selection. *Int J Comput Vis* 30(2):117–154
13. Lindeberg T (1998) A scale selection principle for estimating image deformations. *Image Vis Comput* 16(14):961–977
14. Lindeberg T (2008) Scale-space. In: Wah B (ed.) Encyclopedia of computer science and engineering. Wiley, pp 2495–2504. dx.doi.org/10.1002/9780470050118.ecse609. Also available from <http://www.nada.kth.se/~tony/abstracts/Lin08-EncCompSci.html>
15. Lindeberg T (2010) Interest point from scale-space features: scale-space primal sketch for differential descriptors. *International Journal of Computer Vision*

16. Lindeberg T (2013) Scale selection properties of generalized scale-space interest point detectors. *J Mathematical Imaging and Vision*, 46(2):177–210
17. Lindeberg T, Bretzner L (2003) Real-time scale selection in hybrid multi-scale representations. In: *Proceedings of Scale-Space Methods in Computer Vision*. Lecture Notes in Computer Science, vol 2695. Springer, Berlin/New York, pp 148–163
18. Loog M, Li Y, Tax D (2009) Maximum membership scale selection. In: *Multiple Classifier Systems*. Lecture Notes in Computer Science, vol 5519. Springer, Berlin, pp 468–477.
19. Li Y, Tax DMJ, Loog M (2012) Supervised scale-invariant segmentation (and detection). In: *Proc. Scale Space and Variational Methods in Computer Vision (Scale-Space'11)*, Ein Gedi, Israel. Lecture Notes in Computer Science, vol 6667. Springer, Berlin, pp 350–361
20. Lowe D (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110
21. Mikolajczyk K, Schmid C (2004) Scale and affine invariant interest point detectors. *Int J Comput Vis* 60(1):63–86
22. Mrázek P, Navara M (2003) Selection of optimal stopping time for nonlinear diffusion filtering. *Int J Comput Vis* 52(2–3):189–203
23. Negre A, Braillon C, Crowley JL, Laugier C (2008) Real-time time-to-collision from variation of intrinsic scale. *Exp Robot* 39:75–84
24. Sato Y, Nakajima S, Shiraga N, Atsumi H, Yoshida S, Koller T, Gerig G, Kikinis R (1998) 3D multi-scale line filter for segmentation and visualization of curvilinear structures in medical images. *Med Image Anal* 2(2):143–168
25. Sporrer J, Colios, CJ, Trahanias, PE (2000) Generalized scale-selection. In: *Proc. International Conference on Image Processing (ICIP'00)*, Vancouver, Canada, pp 920–923
26. Witkin AP (1983) Scale-space filtering. In: *Proceedings of 8th International Joint Conference on Artificial Intelligence*, Karlsruhe, Germany, pp 1019–1022
27. Lindeberg T (2013) Invariance of visual operations at the level of receptive fields. *PLoS ONE* 8(7): e66990. doi:10.1371/journal.pone.0066990

Scale-Invariant Image Features and Image Descriptors

► [Scale Selection](#)

Scene/Image Parsing

► [Semantic Image Segmentation](#)

Schott Noise

► [Photon, Poisson Noise](#)

Second Order Approximation

► [Osculating Paraboloids](#)

Segmentation

► [Geodesics, Distance Maps, and Curve Evolution](#)
► [Interactive Segmentation](#)

Semantic Image Segmentation

Jamie Shotton¹ and Pushmeet Kohli²

¹Microsoft Research Ltd, Cambridge, UK

²Department of Computer Science And Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel

Synonyms

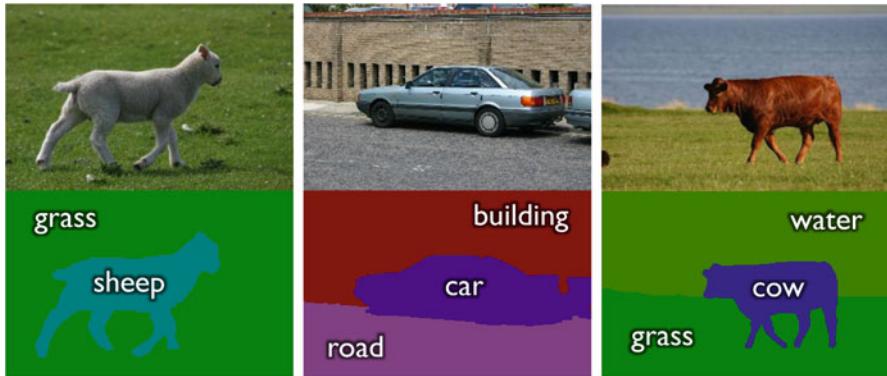
[Object segmentation](#); [Scene/image parsing](#)

Definition

Semantic image segmentation describes the task of partitioning an image into regions that delineate meaningful objects and labeling those regions with an object category label. Some example semantic segmentations are given in [Fig. 1](#). It can be seen as a generalization of figure-ground segmentation [1] where one segments a particular object, say a horse, from the background.

Background

Images typically contain multiple objects, including *things* such as people, cars, and cows and *stuff* such as grass, sky, and water. The imaging process composites the appearances of these objects, leaving (at most) an intensity edge between one object and the next. Semantic image segmentation aims to recover the image regions corresponding directly to objects, as well as labeling those regions with the relevant object category.



Semantic Image Segmentation, Fig. 1 *Top row:* three input images. *Bottom row:* the corresponding semantic segmentations where colors represent object categories

The task is usually approached as supervised or semi-supervised machine learning, using a set of training images that are manually segmented and labeled. The learning algorithm then discovers relevant image features that help discriminate regions belonging to different categories in unseen test images. Other cues such as layout and context further help resolve ambiguities; for example, a pixel that neighbors a car pixel is likely to have the same label, and a region containing a cow is likely to be above a region containing grass.

Semantic image segmentation has been of interest since at least 1989 [2], but only fairly recently [3, 4] have processor speeds started to allow the rich models for high accuracy across many object categories.

Theory

Problem Formulations In essence, the semantic segmentation task can be treated as a pixel-labeling problem. The different methods proposed for solving this problem can be categorized based on the relationships they encode between different pixels (see Fig. 2). Some methods for semantic segmentation solve the pixel-labeling problem by classifying each pixel independently [5, 6]. Another class of methods works by grouping pixels into segments (or super-pixels) and assigning a single label to each group [7]. Superpixels are computed from the image in a bottom-up fashion [8–10] and can aid computational efficiency but may lead to a final incorrect labeling.

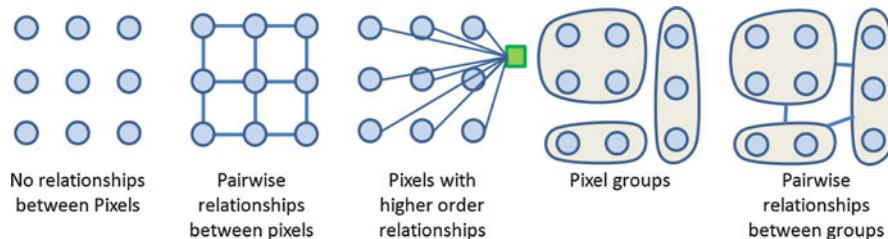
Many semantic segmentation algorithms are based on the pairwise Markov Random Field (MRF) [11]

or Conditional Random Field (CRF) models [12, 13] which enforce relationships between pairs of neighboring pixels [5]. They encourage adjacent pixels that are similar in appearance to take the same semantic label, and lead to segmentation results with smooth boundaries. A related notion of layout consistency was explored in [14].

Some methods go beyond pairwise interactions between pixels and enforce higher-order relationships between groups of (or even all) pixels in the image. For instance, they encourage groups of pixels to take the same semantic label [15], or make sure that some semantic label is taken by at least one pixel in the image [16]. These models also allow the use of top-down object detection results to prime the segmentation [6, 17]. In related work, data-driven Markov Chain Monte Carlo (DDMCMC) was used to parse images with a rich generative model [18]. A nonparametric approach to semantic segmentation was proposed in [19], where labels were transferred between nearest-neighbor images matched using SIFT flow features.

Features Used for Pixel/Super-Pixel Classification

There are many informative image cues that can be used for semantic segmentation, including intensity, color, texture, context, motion, and 3D structure. Dense interest point descriptors can be used, for example, [20–22], or light-weight region integrals of textons [5, 6], or even features based on histogram of gradients [16]. Motion-derived 3D structure was used in [23] to segment images from a video sequence. Range images or depth cameras have been used for semantic segmentation in [24].



Semantic Image Segmentation, Fig. 2 Different approaches for semantic image segmentation

Incorporating Context

Context can be incorporated in several ways: (i) *appearance* context [5] captures the notion that a sheep might typically stand on something green, (ii) *semantic* context [6, 25–27] captures the notion that a sheep might typically stand on grass, and (iii) *auto-context* [6, 28] describes a *recursive* classification procedure whereby the classification uses contextual features from a previous stage of classification, capturing the notion that this pixel might belong to sheep because a nearby pixel was classified as sheep at the previous stage.

Datasets

To a large extent, the quality of the semantic segmentation algorithm is related to the quality of the training dataset. Most techniques require fully pixel-wise labeled images which are expensive to obtain. Other approaches reduce this requirement in various ways, including multiple over-segmentations [29], co-segmenting several images at once [30], incorporating image or region labels [6, 17, 31], and exploiting probabilistic aspect models [32].

Early datasets for semantic segmentation include Corel and Sowerby [33] and the MSRC dataset [5]. Recently, more challenging dataset have been proposed that dramatically increase the variability in the images and thus bring us much closer to solving semantic segmentation as a real-world problem. LabelMe [34] used a web interface to capture a large number of image labels. The Pascal VOC Segmentation Challenge [35] runs yearly and deliberately tries to remove contextual clues in the data to foster the best object detection algorithms. The SUN 09 dataset [36] goes to the other extreme of labeling many objects per image to include as much context as possible. LabelMe, Pascal VOC, and SUN 09 for the most part only provide labels for things and not stuff,

often leaving the background a single heterogenous category.

Application

Accurately segmenting and recognizing what those objects are opens up many potential applications. Not only does it tell us what things are in an image but also where they are and how they look. This information can be used in, for example, image search, image editing, augmented reality, robot navigation, and medical image analysis.

References

- Borenstein E, Ullman S (2002) Class-specific, top-down segmentation. In: Heyden A, Sparr G, Johansen P (eds) Proceedings of the European conference on computer vision. Lecture notes in computer science (ECCV), vol 2351. Springer, Berlin, pp 109–124
- Wright W (1989) Image labelling with a neural network. In: Proceedings of the 5th Alvey vision conference, Reading
- Everingham M, Thomas B, Trosianko T (1999) Head-mounted mobility aid for low vision using scene classification techniques. Int J Virtual Real 3(4):3–12
- Konishi S, Yuille AL (2000) Statistical cues for domain specific image segmentation with performance analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Hilton Head, vol 1, pp 125–132
- Shotton J, Winn J, Rother C, Criminisi A (2009) Textron-boost for image understanding: multi-class object recognition and segmentation by jointly modeling texture, layout, and context. Int J Comput Vis 81(1):2–23
- Shotton J, Johnson M, Cipolla R (2008) Semantic textron forests for image categorization and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Anchorage
- Campbell N, Mackewn W, Thomas B, Trosianko T (1997) Interpreting image databases by region classification. Pattern Recognit 30:555–563

8. Shi J, Malik J (1997) Normalized cuts and image segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Juan, pp 731–737
9. Felzenszwalb P, Huttenlocher D (2004) Efficient graph-based image segmentation. *Int J Comput Vis* 59(2): 167–181
10. Carreira J, Sminchisescu C (2010) Constrained parametric min-cuts for automatic object segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Francisco
11. Geman S, Geman D (1984) Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans Pattern Anal Mach Intell* 6(6):721–741
12. Lafferty J, McCallum A, Pereira F (2001) Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the international conference on machine learning, Williams College, pp 282–289
13. Kumar S, Hebert M (2003) Discriminative random fields: a discriminative framework for contextual interaction in classification. In: Proceedings of the international conference on computer vision, Kerkyra, vol 2, pp 1150–1157
14. Winn J, Shotton J (2006) The layout consistent random field for recognizing and segmenting partially occluded objects. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Miami, vol 1, pp 37–44
15. Kohli P, Ladický L, Torr P (2009) Robust higher order potentials for enforcing label consistency. *Int J Comput Vis* 82:302–324
16. Ladický L, Russell C, Kohli P, Torr P (2010) Graph cut based inference with co-occurrence statistics. In: Proceedings of the European conference on computer vision (ECCV), Heraklion
17. Ladický L, Sturges P, Alahari K, Russell C, Torr P (2010) What, where and how many? Combining object detectors and CRFs. In: Proceedings of the European conference on computer vision (ECCV), Heraklion
18. Tu Z, Chen X, Yuille A, Zhu S (2003) Image parsing: unifying segmentation, detection, and recognition. In: Proceedings of the IEEE conference on computer vision, Nice, France, vol 1, pp 18–25
19. Liu C, Yuen J, Torralba A (2009) Nonparametric scene parsing: label transfer via dense scene alignment. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Miami
20. Lowe D (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110
21. Belongie S, Malik J, Puzicha J (2002) Shape matching and object recognition using shape contexts. *IEEE Trans Pattern Anal Mach Intell* 24(24):509–522
22. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Diego, vol 2, pp 886–893
23. Brostow G, Shotton J, Fauqueur J, Cipolla R (2008) Segmentation and recognition using structure from motion point clouds. In: Proceedings of the European conference on computer vision (ECCV), Marseille
24. Anguelov D, Taskar B, Chatalbashev V, Koller D, Gupta D, Ng A (2005) Discriminative learning of markov random fields for segmentation of 3D scan data. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Beijing
25. Torralba A, Murphy K, Freeman W, Rubin M (2003) Context-based vision system for place and object recognition. In: Proceedings of the conference on computer vision, Nice, France, vol 2, pp 273–280
26. Hoiem D, Efros A (2006) Objects in perspective. In: Proceedings of the conference on computer vision and pattern recognition (CVPR), New York
27. Rabinovich A, Vedaldi A, Galleguillos C, Wiewiora E, Belongie S (2007) Objects in context. In: Proceedings of the international conference on computer vision, Rio de Janeiro
28. Tu Z (2008) Auto-context and its application to high-level vision tasks. In: Proceedings of the international conference on computer vision and pattern recognition (CVPR), Anchorage
29. Russell BC, Efros AA, Sivic J, Freeman WT, Zisserman A (2006) Using multiple segmentations to discover objects and their extent in image collections. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), New York
30. Rother C, Minka T, Blake A, Kolmogorov V (2006) Cosegmentation of image pairs by histogram matching – incorporating a global constraint into MRFs. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), New York
31. Barnard K, Duygulu P, de Freitas N, Forsyth D, Blei D, Jordan MI (2003) Matching words and pictures. *J Mach Learn Res* 3:1107–1135
32. Verbeek J, Triggs B (2007) Region classification with markov field aspect models. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Minneapolis
33. He X, Zemel R, Carreira-Perpiñán M (2004) Multiscale conditional random fields for image labeling. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Washington, DC, vol 2, pp 695–702
34. Russell B, Torralba A, Murphy K, Freeman WT (2008) Labelme: a database and web-based tool for image annotation. *Int J Comput Vis* 77:157–173
35. Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A, The PASCAL VOC challenge. <http://www.pascal-network.org/challenges/VOC/>
36. Choi M, Lim J, Torralba A, Willsky A (2010) Exploiting hierarchical context on a large database of object categories. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), San Francisco

Semiautomatic

► Interactive Segmentation

Semidefinite Optimization

► Semidefinite Programming

Semidefinite Programming

Chunhua Shen and Anton van den Hengel
 School of Computer Science, The University of
 Adelaide, Adelaide, SA, Australia

Synonyms

[Convex minimization](#); [Semidefinite optimization](#)

Definition

Semidefinite programming is a subtopic of convex optimization. Convex optimization refers to minimization of a convex function subject to a set of convex constraints. Semidefinite programming involves minimization of a linear objective function over the intersection of linear constraints and the cone of positive semidefinite matrices. Clearly, semidefinite programming is a special case of convex optimization.

Background

Many computer vision problems can be formulated as convex optimization problems. The main advantage of convex optimization is that if a local minimum exists, then it is also a global minimum. In other words, the convexity guarantees to attain the global optimum if it exists.

In a semidefinite programming problem, one minimizes a linear function subject to the constraint that an affine combination of symmetric matrices is positive semidefinite. Semidefinite programming unifies a few standard problems such as linear programming, quadratic programming, and second-order cone programming. Semidefinite programming has many applications in computer vision.

Theory

Mathematically, semidefinite programming solves the following problem:

$$\begin{aligned} \min_X \langle C, X \rangle, \quad & \text{s.t. } \langle A_k, X \rangle = b_k, (k = 1, \dots, m), \\ & X \succeq 0. \end{aligned} \tag{1}$$

Here, the optimization variable $X \in \mathcal{R}^{D \times D}$ is a symmetric and positive semidefinite matrix. The operator $\langle A, B \rangle = \sum_{ij} A_{ij} B_{ij}$ calculates the inner product of two matrices (or vectors). The last constraint $X \succeq 0$ means X is positive semidefinite. Such a constraint is nonlinear and nonsmooth but convex. The following statements about a semidefinite matrix are equivalent: (1) $X \succeq 0$; (2) All eigenvalues of X are nonnegative; and (3) $\forall u \in \mathcal{R}^D, u^\top X u \geq 0$.

We can easily write the dual problem of (Eq. 1) by finding the saddle point of its Lagrangian:

$$\max_y \langle b, y \rangle, \quad \text{s.t. } \sum_{i=1}^m y_k A_k \preceq C. \tag{2}$$

Here, $y \in \mathcal{R}^m$ is the variable to optimize. The notation $A \preceq B$ means $B - A \succeq 0$. Weak duality and strong duality hold for the primal problem (Eq. 1) and dual problem (Eq. 2) under mild conditions. In particular, strong duality means that optimal values of the primal and dual problems are the same. Strong duality follows from Slater's condition for constraint qualification [1], which is stated as follows. If (Eq. 1) and (Eq. 2) both are feasible and there is a strictly interior point for either (Eq. 1) or (Eq. 2), then optimal primal and dual solutions exist and the corresponding optimal values are the same.

Semidefinite programming can be viewed as an extension of linear programming where the element-wise inequalities between vectors become linear matrix inequalities (LMIs). An LMI takes the form of $\sum_{k=0}^m y_k A_k \geq 0$, with $y = [y_0 \dots y_m]$ being a vector and matrices A_k being symmetric matrices. An LMI enforces a convex constraint on the vector y . Indeed, the interior-point methods for linear programming can be generalized to semidefinite programming. In fact, as shown by Nesterov and Nemirovsky [2], in principle interior-point methods for linear programming can be generalized to all convex optimization problems. Significantly, interior-point methods enable semidefinite programs to be solved very efficiently. It is theoretically guaranteed that the effort needed to solve a semidefinite program to a prescribed accuracy grows no faster than a polynomial of the problem size. Off-the-shelf solvers include CSDP [3], SDPT3 [4], and SeDuMi [5]. One can also use optimization modeling languages such as CVX [6] and YALMIP [7] to greatly simplify the problem modeling process.

Although interior-point methods are polynomial-time algorithms, they do not scale well. If the number of constraints m is of order $O(D^2)$ with the semidefinite matrix being $D \times D$, at each iteration, the computational time is of order $O(D^6)$. This high computational complexity makes interior-point methods impractical for large-scale problems. Recently, first-order augmented Lagrangian approaches have been proposed for solving large problems. In particular, as shown in [8], the alternating direction method is applied to solve the dual augmented Lagrangian problem of the standard semidefinite problem. This method is much more scalable than the conventional interior-point methods.

Application

Semidefinite programming has attracted extensive research interests in computer vision and machine learning due to its many applications in these fields.

Max-cut approximation and image segmentation. It is well known that semidefinite programming can be used to approximately solve some difficult combinatorial optimization problems. It is a useful tool for approximating NP-hard problems. Such an example is the Max-cut problem, which can be described as follows. Given a graph $G = (V, E)$, output a partition of the vertices V so as to maximize the number of edges crossing from one side to the other. This problem can be expressed as an integer quadratic problem:

$$\max_{\mathbf{u}} \sum_{(i,j) \in E} \frac{1 - u_i u_j}{2}, \text{ s.t. } u_i \in \{-1, 1\}, \forall i = 1, \dots, n. \quad (3)$$

This problem is NP-hard due to the discrete constraints. To obtain a semidefinite relaxation, we lift each scalar variable u_i to a higher dimension $\mathbf{u}_i \in \mathcal{R}^n$ and $\|\mathbf{u}_i\|_2 = 1$. We introduce another variable substitution $U_{ij} = \mathbf{u}_i^\top \mathbf{u}_j$ to bring (Eq. 3) into a semidefinite program:

$$\max_U \sum_{(i,j) \in E} \frac{1 - U_{ij}}{2}, \text{ s.t. } U_{ii} = 1, \forall i = 1, \dots, n; U \succeq 0. \quad (4)$$

The constraint $U \succeq 0$ is due to the fact $U_{ij} = \mathbf{u}_i^\top \mathbf{u}_j$. The optimal value of (Eq. 4) is an upper bound of the

original problem (Eq. 3). It is easy to see that (Eq. 4) is a semidefinite problem. Hence, the global solution of (Eq. 4) is guaranteed. Approximate solutions to the original problem can be computed by rounding the vector solution of (Eq. 4). The solutions of the SDP formulation suggest to cut nodes i and j if $U_{ij} = \mathbf{u}_i^\top \mathbf{u}_j$ is close to -1 . The Goemans-Williamson randomized rounding technique is to choose a uniformly random hyperplane through the origin and use it to cut vectors into separate parts. See [9] for details. Similar ideas can be applied to solve combinatorial problems of minimizing quadratic functionals in binary decision variables subject to linear constraints, which has many applications in image segmentation, perceptual grouping, and image restoration. See, for example, [10].

Maximum variance unfolding. Another application is embedding high-dimensional data (such as image data) into an underlying low-dimensional space by maximizing the variance while maintaining the data's local neighborhood. Here, the input data are assumed to be sampled from a much lower dimensional manifold that is embedded inside of a higher dimensional vector space. The primary idea of maximum variance unfolding is to create a mapping that preserves local neighborhoods at every point of the underlying manifold.

Let \mathbf{u}_i ($i = 1, \dots, n$) be the original data points in the high-dimensional manifold and \mathbf{v}_i be the *unfolded* low-dimensional data points. For $(i, j) \in E$ are neighbors, the local isometry constraints are

$$\|\mathbf{v}_i - \mathbf{v}_j\|_2^2 = \|\mathbf{u}_i - \mathbf{u}_j\|_2^2 = d_{ij}, \forall (i, j) \in E. \quad (5)$$

If we constrain the embedded data to center at the origin, we have

$$\sum_i \mathbf{v}_i = \mathbf{0}. \quad (6)$$

The variance of the embedded data can be written as $\frac{1}{n} \sum_i \mathbf{v}_i^\top \mathbf{v}_i$. So the optimization problem becomes

$$\begin{aligned} \max_{\mathbf{v}} & \sum_i \mathbf{v}_i^\top \mathbf{v}_i, \text{ s.t. } \sum_i \mathbf{v}_i = \mathbf{0}, \|\mathbf{v}_i - \mathbf{v}_j\|_2^2 \\ & = d_{ij}, \forall (i, j) \in E. \end{aligned} \quad (7)$$

This problem is a non-convex quadratic program. To formulate it into a semidefinite problem, we

introduce a new variable $V_{ij} = \mathbf{v}_i^\top \mathbf{v}_j$, the same as in the first example. We can then relax (Eq. 7) into

$$\begin{aligned} \max_V \langle V, \mathbf{I} \rangle, \text{ s.t. } \sum_{ij} V_{ij} &= 0, V_{ii} + V_{jj} - 2V_{ij} \\ &= d_{ij}, \forall (i, j) \in E; V \succeq 0. \end{aligned} \quad (8)$$

Here, \mathbf{I} is an identity matrix. Problem (Eq. 8) is a relaxation of (Eq. 7). The optimal value of (Eq. 8) is an upper bound of the one of (Eq. 7) because the feasibility set of (Eq. 8) is a superset of (Eq. 7)'s feasibility set. The advantage of this relaxation is that now (Eq. 8) can be efficiently solved using off-the-shelf semidefinite programming solvers. The embedded points \mathbf{v} can be obtained via Cholesky decomposition of the matrix V . Weinberger and Saul [11] applied maximum variance unfolding to detect low-dimensional structure in high-dimensional data sets of images.

References

1. Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge University Press, Cambridge
2. Nesterov Y, Nemirovsky A (1988) A general approach to polynomial time algorithms design for convex programming. Technical report, USSR Acad Sci, Moscow, USSR
3. Borchers B (1999) CSDP, a C library for semidefinite programming. Optim Methods Softw 11(1):613–623
4. Toh K, Todd M, Tutuncu R (1999) SDPT3—a matlab software package for semidefinite programming. Optim Methods Softw 11(1–4):545–581
5. Sturm JF (1999) Using SeDuMi 1.02, a matlab toolbox for optimization over symmetric cones. Optim Methods Softw 11(1–4):625–653
6. Grant M, Boyd S (2011) CVX: matlab software for disciplined convex programming, version 1.21. <http://cvxr.com/>
7. Löfberg J (2004) YALMIP: a toolbox for modeling and optimization in MATLAB. Proceedings of the IEEE symposium on computer-aided control system design, Taipei, Taiwan
8. Wen Z, Goldfarb D, Yin W (2009) Alternating direction augmented lagrangian methods for semidefinite programming. Math Program Comput 2(3–4):203–230
9. Goemans MX, Williamson DP (1994) .879-approximation algorithms for max cut and max 2SAT. Proceedings of the ACM symposium on theory of computing. ACM, New York, pp 422–431
10. Keuchel J, Schnörr C, Schellewald C, Cremers D (2003) Binary partitioning, perceptual grouping, and restoration with semidefinite programming. IEEE Trans. Pattern Anal Mach Intell 25(11):1364–1379
11. Weinberger KQ, Saul LK (2006) Unsupervised learning of image manifolds by semidefinite programming. Int J Comput Vision 70(1):77–90

Sensor Fusion

Pramod K. Varshney

Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, NY, USA

Synonyms

Multisensor data fusion

Related Concepts

► Data Fusion

Definition

Sensor fusion refers to systems, techniques, theory, and tools that exploit the synergy in the information acquired from multiple sensors to enhance system performance.

Background

Conventional systems used single sensors for monitoring phenomenon of interest and make inferences regarding them. Due to significant advances in sensing, networking, and computing technologies, multiple sensors are increasingly being used. This provides improved system performance, resulting in a better understanding of the phenomenon being monitored. In addition, distributed sensing improves robustness, extends spatial and temporal coverage while requiring shorter response time [1–3]. In order to optimally fuse information acquired from different distributed sensing architectures, advances in theory and algorithm design are required.

Theory

Data from multiple sensors can be combined at three possible levels. In data level fusion, raw sensor data is combined. This requires that data acquired from different sensors be commensurate and the data needs to be transported to a fusion center for centralized

processing. This approach has the potential of achieving the best possible performance at the expense of large communication requirements. For noncommensurate data, either feature level fusion or decision level fusion is employed. In feature level fusion, features are extracted from the data which are then fused. In decision level fusion, higher level decisions such as detections and estimates are obtained based on data from individual sensors. These decisions are then fused at the fusion center. In feature level and decision level fusion, data transmission requirements are lower, but the quality of fused result degrades due to data compression involved in the feature extraction and decision-making processes. Several topologies for sensor fusion such as parallel, serial, tree, and network topologies can be used. Choice of topology is often application dependent, but parallel topology is used quite commonly.

Sensor fusion is employed to solve a number of generic problems that results in improved situational awareness for the phenomenon under observation. Object and event detection using multisensor data is carried out based on distributed detection theory and decision fusion [4–6]. For conditionally independent observations, a likelihood ratio-based quantizer is employed at the sensors, and the fusion rule is based on a weighted sum of incoming quantized data. For parameter estimation or tracking problems, quantized data are fused at the fusion center [7, 8]. For tracking, a number of distributed filtering and track fusion algorithms are employed [9, 10]. When tracking multiple objects, data association is a major challenge [11, 12]. When sensor data is image and video data, data-level image fusion techniques are employed. One popular approach is to transform image data into another domain, e.g., wavelet domain, perform fusion in the transformed domain, and then transform the fused data back to the image domain [13]. Sensor fusion approaches are often problem and sensing modality dependent. New approaches are constantly being devised, such as distributed inference based on probabilistic graphical models [14] or through consensus and gossip algorithms [15, 16].

Application

There are many military and nonmilitary applications of sensor fusion. In military applications, sensor

fusion is employed for the detection, location, tracking, and identification of military entities such as aircrafts, ships, submarines, ground units, emitters, and weapons. Different sensing modalities such as radar, sonar, electro-optic imagers, infrared imagers, and electronic intelligence are employed. Nonmilitary applications are numerous and continue to increase. Wireless sensor networks [17, 18] and distributed camera networks [19, 20] are being deployed for many application domains. These include air traffic control, homeland security, medical diagnosis, smart homes and buildings, monitoring of critical infrastructures, robotics, vehicle health management, remote sensing, and environmental monitoring.

Open Problems

There are many theoretical and practical challenges to fully utilize the potential of distributed sensing and sensor fusion. These include scaling, fundamental limits on achievable performance, fusion of heterogeneous sensors, treatment of dependent data, and fusion of hard and soft data.

References

1. Varshney PK (1997) Scanning the issue. Proc IEEE 85(1): 3–5
2. Varshney PK (1997) Multisensor data fusion. Electron Commun Eng J 9(12):245–253
3. Liggins ME, Llinas J, Hall DL (2008) Handbook of multisensor data fusion: theory and practice, 2nd edn. CRC, Boca Raton
4. Varshney PK (1997) Distributed detection and data fusion. Springer, New York
5. Chen B, Tong L, Varshney PK (2006) Channel-aware distributed detection in wireless sensor networks. IEEE Signal Process Mag 23(4):16–26
6. Chamberland J-F, Veeravalli VV (2007) Wireless sensors in distributed detection applications. IEEE Signal Process Mag 24(3):16–25
7. Xiao J-J, Ribeiro A, Giannakis GB, Luo Z-Q (2006) Distributed compression estimation using wireless sensor networks. IEEE Signal Process Mag 23(4): 27–41
8. Ribeiro A, Schizas I, Roumeliotis S, Giannakis G (2010) Kalman filtering in wireless sensor networks. IEEE Control Syst Mag 30(2):66–86
9. Bar-Shalom Y, Blair WD (2000) Multitarget-multisensor tracking: applications and advances, vol III. Artech House, Boston

10. Mallick M, Krishnamurthy V, Vo B-N (2011) Integrated tracking, classification, and sensor management: theory and applications. Wiley/IEEE
11. Mahler Ronald PS (2007) Statistical multisource-multitarget information fusion. Artech House, Boston
12. Bar-Shalom Y, Daum F, J Huang J (2009) The probabilistic data association filter. IEEE Control Syst Mag 29(6): 82–100
13. Blum R, Liu Z (2005) Multi-sensor image fusion and its applications. CRC, Boca Raton
14. Çetin M, Chen L, Fisher III JW, Ihler AT, Moses RL, Wainwright MJ, Willsky AS (2006) Distributed fusion in sensor networks: a graphical models perspective. IEEE Signal Process Mag 23(4):42–55
15. Olfati-Saber R, Fax JA, Murray RM (2007) Consensus and cooperation in networked multi-agent systems. Proc IEEE 95(1):215–233
16. Dimakis AG, Kar S, Moura JMF, Rabat MG, Scaglione A (2010) Gossip algorithms for distributed signal processing. Proc IEEE 98(11):1847–1864
17. Akyldiz IF, Vuran MC (2010) Wireless sensor networks. Wiley, Chichester/Hoboken
18. Swami A, Zhao Q, Hong Y-W, Tong L (2007) Wireless sensor networks: signal processing and communications perspectives. Wiley, Chichester/England/Hoboken
19. Foresti GL, Regazzoni CS, Varshney PK (2003) Multisensor surveillance systems: the fusion perspective. Kluwer, Boston
20. Bhambhani B, Ravishankar CV, Roy-Chowdhury AK, Aghajan H, Terzopoulos D (eds) (2011) Distributed video sensor networks. Springer, London/New York

Shadow

► [Penumbra and Umbra](#)

Shape from Outlines of Projection

► [Shape from Silhouette](#)

Shape from Scatter

Mohit Gupta
Department of Computer Science, Columbia
University, New York, NY, USA

Synonyms

[Depth from Scattering](#)

Definition

Light scatters in the presence of volumetric media such as fog, smoke, mist, dust, and murky water. Volumetric scattering results in several daily life visual effects, such as the glow around the streetlights and car headlights on a foggy day and the murky appearance of underwater scenes. From a computer vision point of view, while on one hand, volumetric scattering degrades images by reducing contrast, it also provides important shape/depth cues, especially for outdoor scenes. This entry provides a summary of various techniques and algorithms for recovering shape/depth using scattering.

Background

Volumetric scattering in the atmosphere (atmospheric scattering) has been studied for over two centuries in the atmospheric optics literature. Some of the prominent sources of literature on the subject are books by Minnaert [1], Middleton [2], and McCartney [3]. In computer vision literature, Cozman and Krotkov [4] and Narasimhan, Schechner, and Nayar [5–12] were among the first to develop techniques for scene analysis under volumetric scattering. Excerpts from these papers are used in several locations in this entry. Most of the algorithms presented in these papers required capturing two or more images of the scene under different weather/imaging conditions. Note that while the models and techniques in this entry are discussed in the context of atmospheric scattering, they are valid in general for other volumetric scattering scenarios, such as underwater imaging [13, 14].

Theory

Mechanisms of Atmospheric Scattering: There are two main mechanisms for getting depth from atmospheric scattering: attenuation and airlight.

Attenuation: Light gets deflected and absorbed as it travels through a volumetric medium (e.g., haze, mist, murky water), resulting in exponential *attenuation* of the intensity. Consider a collimated beam of light traveling through a medium from point A to point B. Suppose the irradiance of the beam at point A is

$E_A(\lambda)$, where λ is the wavelength of light. The irradiance of the beam at point B after the attenuation is given as

$$E_B(\lambda) = E_A(\lambda) e^{-\int_A^B \sigma(x, \lambda) dx}, \quad (1)$$

where $\sigma(x, \lambda)$ is the attenuation coefficient of the medium for light of wavelength λ . $\sigma(x, \lambda)$ is directly proportional to the volume density $\rho(x)$ of the medium. For homogeneous media, $\sigma(x, \lambda)$ is constant with respect to the spatial location, that is, $\sigma(x, \lambda) = \sigma(\lambda)$. Thus, in the presence of homogeneous media, the attenuated intensity at point B is given as

$$E_B(\lambda) = E_A(\lambda) e^{-\sigma(\lambda)d}, \quad (2)$$

where d is the distance between points A and B.

So far in this entry, light attenuation only for a collimated light beam has been considered. For diverging beams from point light sources, there is an additional inverse-square falloff. Given a point light source at point A with a radiant intensity of $I_o(\lambda)$, the irradiance at point B is

$$E_B(\lambda) = g \frac{I_o(\lambda) e^{-\sigma(\lambda)d}}{d^2}, \quad (3)$$

where the constant g accounts for the optical parameters of the camera and conversion between radiant intensity and irradiance.

Airlight: A second mechanism causes the atmosphere to behave like a source of light. This phenomenon is called airlight [15], and it is caused by the scattering of environmental illumination by particles in the atmosphere towards the observer. The environmental illumination can have several sources, including, direct sunlight, diffuse skylight, and light reflected by the ground. Consider an observer at a distance d from a physical object. The total radiance due to airlight at the observer is given as (for a derivation, see [5])

$$L(d, \lambda) = L(\infty, \lambda) (1 - e^{-\sigma(\lambda)d}), \quad (4)$$

where $L(\infty, \lambda)$ is the radiance due to airlight for an object at infinity (e.g., horizon). While attenuation causes the scene radiance to decrease with path length, airlight increases with path length. It therefore causes the apparent brightness of a scene point to increase with depth.

Application

1. *Depth from Attenuation:* Consider the image of a scene at night. Suppose the scene has artificial light sources, for example, street lights, building windows, and car headlights. There is no airlight due to sunlight or skylight. In this setting, the dominant scattering mechanism is attenuation. The observed irradiance due to a light source of radiance intensity $I(\lambda)$ at a distance d is

$$E(d, \lambda) = g \frac{I(\lambda) e^{-\sigma(\lambda)d}}{d^2}, \quad (5)$$

where, as before, $\sigma(\lambda)$ is the attenuation coefficient of the medium and g is the parameter which accounts for the conversion between radiant intensity and irradiance. If the detector of the camera has spectral response $s(\lambda)$, the final image brightness value recorded is determined as

$$\begin{aligned} E'(d) &= \int s(\lambda) E(d, \lambda) d\lambda \\ &= \int g s(\lambda) \frac{I(\lambda) e^{-\sigma(\lambda)d}}{d^2} d\lambda. \end{aligned} \quad (6)$$

Since the spectral bandwidth of the camera is limited (visible light range when camera is black and white and even narrower spectral bands when the camera is color), it is safe to assume the attenuation coefficient $\sigma(\lambda)$ to be constant over this bandwidth.

Then

$$E'(d) = g \frac{e^{-\sigma d}}{d^2} \int s(\lambda) I(\lambda) d\lambda = g \frac{e^{-\sigma d}}{d^2} I'. \quad (7)$$

If the light source is observed under two different weather conditions, for example, dense fog and mild fog (or one condition could be clear weather with $\sigma = 0$), there are two different attenuation coefficients σ_1 and σ_2 . Taking the ratio of the two resulting image brightness values,

$$R = \frac{E'_1}{E'_2} = e^{-(\sigma_1 - \sigma_2)d}. \quad (8)$$

Using the natural log,

$$R' = \ln R = (\sigma_1 - \sigma_2)d. \quad (9)$$

This quantity is independent of the sensor gain and the radiant intensity of the source. By computing the above quantity (log of ratio) for two different light sources in the scene and taking their ratio, the relative depths of the two source locations can be computed:

$$\frac{R'_i}{R'_j} = \frac{d_i}{d_j}. \quad (10)$$

Hence, the relative depths of all sources (with unknown radiant intensities) in the scene can be computed from two images taken under unknown but different haze or fog conditions.

2. *Depth from Airlight*: Under dense fog and close-by objects or mild fog and distant objects, attenuation of object brightness is severe, and airlight is the main cause of image irradiance. Also, in the case of dense haze around noon, airlight dominates. In these scenarios, scaled scene depths can be computed from a single image [5, 9].

Let a scene point at depth d produce airlight radiance $L(d, \lambda)$. If the camera has a spectral response $s(\lambda)$, the final brightness value recorded for the scene point is

$$E'(d) = g \int s(\lambda)L(d, \lambda)d\lambda, \quad (11)$$

where, as before, g accounts for the constant of proportionality between scene radiance and image irradiance. Substituting the model of airlight from Eq. 4:

$$E'(d) = \int gs(\lambda)L(\infty, \lambda)(1 - e^{-\sigma(\lambda)d})d\lambda. \quad (12)$$

By assuming that the scattering coefficient $\sigma(\lambda)$ is more or less constant over the spectral band of the camera,

$$E'(d) = E(\infty)(1 - e^{-\sigma d}). \quad (13)$$

This equation gives a relationship between the observed airlight intensity E' and the scaled scene depth σd . $E(\infty)$ is the image intensity corresponding to a scene point at infinity and can be measured from the image if the horizon is visible (this part can be identified as the brightest region of the image). Then, the scaled depth can be computed as

$$\sigma d = \ln\left(\frac{E(\infty)}{E(\infty) - E'(d)}\right). \quad (14)$$

3. *Depth from Chromatic Decomposition*: So far in the entry, attenuation and airlight have been considered separately. At night, there can be no airlight (since there is no environmental illumination) and hence, attenuation dominates. In contrast, under dense fog or haze during daylight, the radiance from a scene point is severely attenuated and hence airlight dominates. However, in most situations, the effects of both attenuation and airlight coexist.

Narasimhan and Nayar presented a technique [6, 9] to recover depths for these situations by considering the chromatic effects of atmospheric scattering. The key idea is that the spectral composition of the irradiance at a scene point is the weighted sum of two distributions, corresponding to the direct transmission after attenuation and airlight. The weights are a function of the weather conditions. Using this model, the authors showed that it is possible to recover depths by capturing and performing simple arithmetic operations on two images of the scene under different bad weather conditions. For details, the reader is referred to Refs. [6, 9].

4. *Depth from Dehazing*: There are several techniques which explicitly remove the effects of scattering from the images, for example, using polarization [8, 10] or image priors [16–18]. While the primary goal of these *dehazing* techniques is to improve visibility in the images, as a by-product, scene depths are also recovered by using the technique outlined in the section *Depth from Airlight* on the separated haze layer (airlight).

Open Problems

A long-standing open problem has been *single image dehazing*, that is, removing the effects of atmospheric scattering using a *single image*. This would enable image and scene recovery for dynamic scenes. Recently, there have been a few techniques which have addressed this problem [16–18]. In order to account for the under-constrained nature of the problem, these techniques use a variety of scene priors [16–18].

References

- Minnaert M (1954) The nature of light and color in the open air. Dover, New York

2. Middleton WEK (1952) Vision through the atmosphere. University of Toronto Press, Toronto
3. McCartney EJ (1976) Optics of the atmosphere: scattering by molecules and particles. Wiley, New York
4. Cozman F, Krotkov E (1997) Depth from scattering. Proc IEEE conf on Computer Vision and Pattern Recognition (CVPR), San Juan. 801–806
5. Nayar SK, Narasimhan SG (1999) Vision in bad weather. Proc IEEE ICCV 2:820–827
6. Narasimhan SG, Nayar SK (2000) Chromatic framework for vision in bad weather. Proc IEEE CVPR 1: 598–605
7. Narasimhan SG, Nayar SK (2001) Removing weather effects from monochrome images. Proc IEEE CVPR 2:186–193
8. Schechner YY, Narasimhan SG, Nayar SK (2001) Instant dehazing of images using polarization. Proc IEEE CVPR 1:325–332
9. Narasimhan SG, Nayar SK (2002) Vision and the atmosphere. IJCV 48(3):233–254
10. Schechner YY, Narasimhan SG, Nayar SK (2003) Polarization-based vision through haze. Applied Optics 42(3):511–525
11. Narasimhan SG, Nayar SK (2003) Shedding light on the weather. Proc IEEE CVPR 1:665–672
12. Narasimhan SG, Nayar SK (2003) Interactive deweathering of an image using physical models. In: IEEE Workshop on Color and Photometric Methods in Computer Vision, In Conjunction with ICCV, Nice, France
13. Treibitz T, Schechner Y (2006) Instant 3descatter. Proc IEEE conf Computer Vision and Pattern Recognition (CVPR), New York, NY, pp 1861–1868
14. Schechner YY, Karpel N (2005) Recovery of underwater visibility and structure by polarization analysis. IEEE J Ocean Eng 30(3):570–587
15. Koschmieder H (1924) Theorie der horizontalen sichtweite. Beitr Phys freien Atm 12:33–53, 171–181
16. Tan RT (2008) Visibility in bad weather from a single image. Proc IEEE conf on Computer Vision and Pattern Recognition (CVPR), Anchorage, AK. 801–806
17. Fattal R (2008) Single image dehazing. ACM Trans Graphics 27(3):72:1–9
18. He K, Sun J, Tang X (2009) Single image haze removal using dark channel prior. Proc IEEE Trans Pattern Analysis and Machine Intelligence. 33(12):2342–2353

Shape from Shadows

Pascal Mamassian

Laboratoire Psychologie de la Perception, Université Paris Descartes, Paris, France

Definition

Shape from shadows is the inference of the three-dimensional shape of objects from their shadows.

Background

David Waltz first introduced to computer vision the problem of shape from shadows while attempting to segment objects that had sharp edges in a three-dimensional scene under various lighting conditions [1]. The problem was then refined to arbitrary surfaces [2]. Nowadays, shape from shadows is a cue commonly used when one is interested in inferring the three-dimensional structure of a visual scene. A benefit relative to other cues such as stereopsis or motion parallax is that it is available in a single image. Shape from shadows differs from shape from shading because in the latter the useful information is the gradient of reflected light on a curved surface, whereas shape from shadows considers that the information coming from the area inside cast shadow is uniform (there are no variations inside the shadow that are related to the scene geometry [3]). However, it is important to remember that similar to most cues to three-dimensional shape, shape from shadows is an under-constrained problem in the sense that several critical scene parameters are often unknown, in particular the light source position and the nature of the surface on which the shadow is cast.

Shadow Identification

The first serious problem in shape from shadows is to identify a region of the image as a shadow. Dark regions on a surface can be caused by the absence of illumination or by a darker surface reflectance (albedo). The cues that allow humans to distinguish variations in light intensity from variations in material properties include luminance relations across shadow borders, figural relations, 3D-shape, depth, color, texture, and motion [4]. In computer vision, very promising results have been obtained if a shading model is assumed [5].

Another dichotomy exists between self-shadows (or attached shadows, i.e., shadows cast on the object itself) and cast shadows (shadows cast on a remote surface) [6]. This differentiation can be achieved to some extent by using some invariant color properties [7]. In addition, a variety of geometrical constraints of self-shadows on a smooth surface have been derived [8, 9]. The list of shadow properties that human and artificial visual systems could use is still open [10].

Shadow Correspondence Problem

In order for shadows to be useful, appropriate parts of an object should be matched to corresponding parts of shadows. The choice of the appropriate features to solve the correspondence problem is still a topic of intense research [11]. Potential candidates include points of high curvature along the object and shadow contours, some Fourier components for textured surfaces [12], or only the low spatial frequencies if one is merely interested in inferring where an object is relative to the background [11]. The shadow correspondence problem is more complex when the receiving surface is not flat. In particular, perceptual problems are thought to arise when the receiving surface is saddle shaped [13].

Using multiple images of the scene under various illuminations, especially if the light source positions are controlled in an active way, is highly beneficial [14, 15]. Multiple images can also be useful to detect the presence of concavities in an object [16].

Open Problems

Object segmentation and recognition are believed to be easier when shadows are absent from the image [17]. Is it true, and if so, what is the best method to eliminate shadows in a scene?

The dual problem of shape from shadows is to infer the illumination from shadows [18]. Should we separately estimate the shape and the illumination from shadows or is it better to attempt to infer both simultaneously?

Moving shadows bring new challenges for object segmentation and tracking [19]. So why do human observers appear to excel at extracting the spatial layout of a scene when moving shadows are present [20]?

References

1. Waltz DL (1975) Understanding line drawings of scenes with shadows. In: Winston PH (ed) *The psychology of computer vision*. McGraw-Hill, New York, pp 19–91
2. Shafer SA, Kanade T (1983) Using shadows in finding surface orientations. *Comput Vis Graph Image Process* 22:145–176
3. Cavanagh P, Leclerc YG (1989) Shape from shadows. *J Exp Psychol* 15(1):3–27
4. Kingdom FAA (2008) Perceiving light versus material. *Vis Res* 48(20):2090–2105
5. Bell M, Freeman WT (2001) Learning local evidence for shading and reflectance. In: *Proceedings of international conference on computer vision (ICCV)*, Vancouver, BC, vol 1, pp 670–677
6. Mamassian P, Knill DC, Kersten D (1998) The perception of cast shadows. *Trends Cogn Sci* 2(8):288–295
7. Salvador E, Green P, Ebrahimi T (2001) Shadow identification and classification using invariant color models. In: *Proceedings of ICASSP 01*, Salt Lake City, UT, vol 3 (IEEE), pp 1545–1548
8. Knill DC, Mamassian P, Kersten D (1997) Geometry of shadows. *J Opt Soc Am A* 14(12):3216–3232
9. Hatzitheodorou M (1998) Shape from shadows: a Hilbert space setting. *J Complex* 14(1):63–84
10. Casati R (2004) The shadow knows: a primer on the informational structure of cast shadows. *Percept* 33(11):1385–1396
11. Mamassian P (2004) Impossible shadows and the shadow correspondence problem. *Percept* 33(11):1279–1290
12. Ramamoorthi R, Koudelka M, Belhumeur P (2005) A Fourier theory for cast shadows. *IEEE Trans Pattern Anal Mach Intell* 27:288–295
13. Norman JF, Lee YL, Phillips F, Norman HF, Jennings LR, McBride TR (2009) The perception of 3-D shape from shadows cast onto curved surfaces. *Acta Psychol* 131(1):1–11
14. Clark JJ, Wang L (2001) Active shape-from-shadows with controlled illuminant trajectories. *Int J Comput Vis* 43(3):141–166
15. Kriegman D, Belhumeur P (2001) What shadows reveal about object structure. *J Opt Soc Am A* 18:804–813
16. Savarese S, Andreetto M, Rushmeier H, Bernardini F, Perona P (2007) 3D reconstruction by shadow carving: theory and practical evaluation. *Int J Comput Vis* 71(3):305–336
17. Finlayson G, Hordley S, Lu C, Drew M (2006) On the removal of shadows from images. *IEEE Trans Pattern Anal Mach Intell* 28:59–68
18. Sato I, Sato Y, Ikeuchi K (2003) Illumination from shadows. *IEEE Trans Pattern Anal Mach Intell* 25:290–300
19. Jiang H, Drew MS (2007) Shadow resistant tracking using inertia constraints. *Pattern Recognit* 40(7):1929–1945
20. Kersten D, Mamassian P, Knill DC (1997) Moving cast shadows induce apparent motion in depth. *Percept* 26(2):171–192

Shape from Silhouette

David C. Schneider

Image Processing Department, Fraunhofer Heinrich Hertz Institute, Berlin, Germany

Synonyms

[Shape from outlines of projection; Visual hull](#)

Related Concepts

► Visual Hull

Definition

Shape from silhouette (SfS) algorithms compute the (approximate) 3D shape of an object from multiple 2D projections considering only the outline of the object in the projections. The most important class of SfS methods are Visual Hull algorithms.

Background

The problem of computing shape from outlines (silhouettes) of projections is generally underconstrained. Depending on the object's geometry and the number of available views, approximations can be computed which are sufficient for some applications. Often, the SfS output serves as initialization for appearance-based shape reconstruction methods such as volumetric reconstruction or multiview stereo.

The most general class of SfS approaches, where no assumptions about the type of objects to reconstruct are made, are the Visual Hull algorithms (e.g., [1–4]). These are treated in a dedicated article.

If the class of objects to handle is restricted, specialized model-based algorithms can solve SfS-like problems even for a single view. In this case, the 3D shape is not actually reconstructed. The algorithms rather compute a shape within the scope of the model that satisfies silhouette constraints and otherwise maximizes domain-specific priors (e.g., a likelihood with respect to a learned distribution). Examples are [5, 6], where human body pose is estimated from a single silhouette, or [7] where the shape of a head is estimated from a face profile.

References

1. Baumgart BG (1974) Geometric modeling for computer vision. PhD thesis, Stanford, Stanford, CA, USA
2. Ahuja N, Veenstra J (1989) Generating octrees from object silhouettes in orthographic views. IEEE Trans Pattern Anal Mach Intell 11(2):137–149

3. Laurentini A (1991) The visual hull: a new tool for contour-based image understanding. In: Proceedings of the 7th Scandinavian conference on image analysis. Pattern Recognition Society of Denmark, Aalborg East
4. Szeliski R (1993) Rapid octree construction from image sequences. CVGIP 58:23–32
5. Balan AO, Sigal L, Black MJ, Davis JE, Haussecker HW (2007) Detailed human shape and pose from images. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR '07). IEEE, Piscataway, Minneapolis, Minnesota, pp 1–8
6. Guan P, Weiss A, Balan AO, Black MJ (2009) Estimating human shape and pose from a single image. In: Proceedings of the IEEE 12th International Conference on Computer Vision (ICCV), Kyoto, Japan, pp 1381–1388
7. Schneider DC, Eisert P (2009) Fast nonrigid mesh registration with a data-driven deformation prior. In: Proceedings of the IEEE 12th international computer vision workshops, Kyoto, Japan, pp 304–311

Shape from Specular Reflections

► Shape from Specularities

Shape from Specularities

Silvio Savarese

Department of Electrical and Computer Engineering,
University of Michigan, Ann Arbor, MI, USA

Synonyms

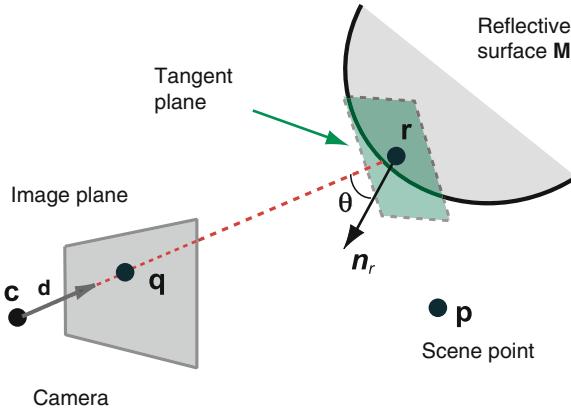
Shape from specular reflections

Definition

Specular reflections or *specularities* carry valuable information about the geometry of reflective surfaces and can be used to recover their shape.

Background

Cues such as texture and shading are often inadequate for recovering the shape of shiny reflective objects. For such objects it is not possible to observe their surfaces directly, rather only what they reflect. Yet, specular reflections present an additional cue that potentially



Shape from Specularities, Fig. 1 The basic shape from specularity problem. Given a scene point reflected off a specular surface M at \mathbf{r} and given its observation in the camera image plane at \mathbf{q} , the goal is to recover 3D location of \mathbf{r} and higher-order local parameters of M around \mathbf{r}

may be exploited for shape recovery. A curved mirror produces “distorted” images of the surrounding world. For example, the image of a straight line reflected by a curved mirror is, in general, a curve (Fig. 2). It is clear that such distortions are systematically related to the shape of the surface. Is it possible to invert this map and recover the shape of the mirror from its reflected images? The general “inverse mirror” problem is under-constrained: by opportunely manipulating the surrounding world, we may produce a great variety of images from any curved mirror surface. This inverse problem may become tractable under the assumption that some knowledge about the structure of the scene, the shape of the object, or the observer is available.

Theory

The basic shape from specularities problem is formulated as follows: Let M be a reflective surface whose shape is unknown (Fig. 1). Consider a camera observing M and let \mathbf{c} be the camera projection center. Given a scene point (or light point source) \mathbf{p} , let \mathbf{q} be the image of \mathbf{p} observed in the image plane through a specular reflection on the reflective surface at \mathbf{r} . Let \mathbf{n}_r be the unit normal to the surface at \mathbf{r} (i.e., the normal of the tangent plane of M at \mathbf{r}). The main objective of the shape from specularities problem is to recover the 3D location of \mathbf{r} in the camera reference system

given the observation \mathbf{q} . Often, it is desirable to recover higher-order local shape information around \mathbf{r} . In this case the objective is to estimate \mathbf{r} , the unit normal \mathbf{n}_r , and second- or higher-order local parameters of M at \mathbf{r} such as, for instance, directions and magnitudes of the surface principal curvature at \mathbf{r} .

If the camera is calibrated (i.e., if the internal camera parameters are known), the surface position at \mathbf{r} is completely determined by a single distance parameter s . This is easy to show: It follows from the perspective projection constraint that the point \mathbf{r} must belong to the line defined by \mathbf{c} and \mathbf{q} , resulting in the following relationship:

$$\mathbf{r} = \mathbf{c} + s \mathbf{d}, \quad (1)$$

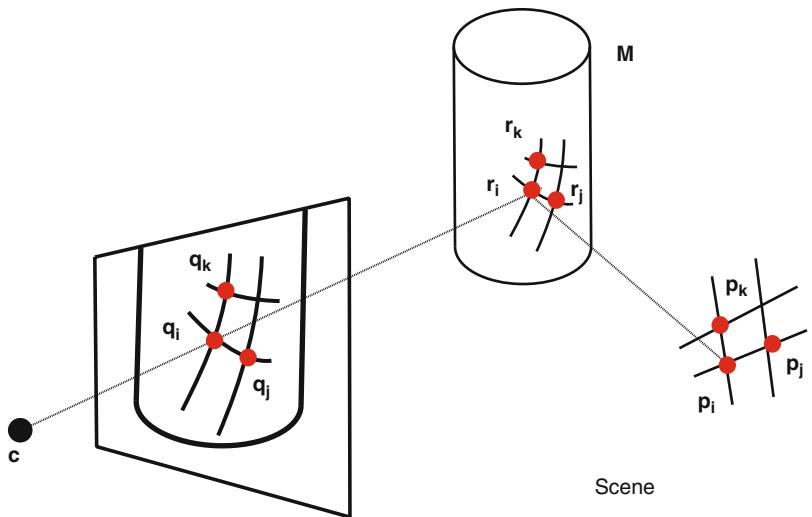
where the unit vector $\mathbf{d} = (\mathbf{q} - \mathbf{c}) / \| \mathbf{q} - \mathbf{c} \|$ is parallel to the line of sight and $s = \| \mathbf{r} - \mathbf{c} \|$ is the distance from \mathbf{c} to \mathbf{r} . It follows that \mathbf{r} is known up one degree of freedom (i.e., s). Notice that no information about the surface normal \mathbf{n}_r or other higher-order surface parameters can be obtained in this case.

Assumptions on geometrical configuration of the scene (light source) or the reflectance properties of the surface lead to further constrains for estimating s and higher-order local shape information around \mathbf{r} . Some of the most notable cases are:

- *Single calibrated scene point (light point source).* If one assumes that the location of \mathbf{p} is known in the camera reference system, it can be shown that \mathbf{r} and \mathbf{n}_r are known up to one-dimensional family of solutions parameterized by s [1, 2]. This unknown can be determined if one further assumes that \mathbf{p} is at infinity [3].
- *Extended calibrated scene (light source).* The following assumptions are made. Assume that (at least) three points $\{\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k\}$ can be identified within a neighborhood of a scene planar patch and their reflection $\{\mathbf{r}_i, \mathbf{r}_j, \mathbf{r}_k\}$ off the surface M are observable in the image plane as $\{\mathbf{q}_i, \mathbf{q}_j, \mathbf{q}_k\}$ (Fig. 2). Assume that correspondences between image points and scene points can be established (e.g., one knows that \mathbf{q}_i is the observation of reflection \mathbf{r}_i of scene point \mathbf{p}_i , for any i). Then, it can be shown that location of $\{\mathbf{r}_i, \mathbf{r}_j, \mathbf{r}_k\}$ and corresponding surface normals at $\{\mathbf{r}_i, \mathbf{r}_j, \mathbf{r}_k\}$ can be recovered as long as $\{\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k\}$ are close enough and are not collinear. Moreover, second-order information of M at $\{\mathbf{r}_i, \mathbf{r}_j, \mathbf{r}_k\}$ (i.e., directions and magnitudes of

Shape from Specularities, Fig. 2

Local shape of a reflective surface M can be recovered up second order if (at least) three points $\{\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k\}$ (along with their observations $\{\mathbf{q}_i, \mathbf{q}_j, \mathbf{q}_k\}$) can be identified within a small planar patch in the scene [1]



principal curvature) can be estimated up to one-dimensional family of solutions. Details of this analysis are presented in the theoretical work by Savarese et al. [1] where necessary and sufficient conditions for reconstructing the local shape of a specular surface from one or multiple reflected calibrated points and (or) lines are discussed.

- *Known reflectance model for the surface.* If the reflected light is modeled using a physical reflectance model such as the Torrance-Sparrow BRDF model [4], a reflected point on the surface can be in general characterized by an extended pattern which is typically indicated as a *highlight*. It can be shown that by measuring the radiance falloff of specular highlights [5], the local curvature can be estimated.

Several practical methods have been proposed for recovering the shape of specular surfaces that use a setup similar to that in Figs. 1 and 2 wherein a calibrated fixed camera observes a scene element (light source) being reflected off the surface. Methods vary depending on whether the scene is calibrated or uncalibrated, static or time varying, and point source or extended. Pioneering work by Ikeuchi [3] and, later, works by Sanderson et al. [6] utilize distant light sources which are sequentially activated so as to generate sequences of point source reflections. Halstead et al. [7] and Tarini et al. [8] use extended scene structures composed of conically shaped patterns and a sequence of striped patterns, respectively. These methods recover the complete surface of a shiny object by iteratively fitting a parametric representation of the

shape to local measurements. Zheng and Murata [9] propose a system where the object rotates while being illuminated by extended radial light sources. Savarese et al. [1] use a single calibrated grid pattern similar to that in Fig. 2 to obtain sparse local surface estimates (up to second order). Rozenfeld et al. [10] extend this analysis and demonstrate that dense reconstruction can be obtained from sparse correspondences between scene points and observations. Kutulakos and Steger [11] exploit directed ray measurements of a calibrated planar target reflected off the surface and positioned at different locations in the world reference system. Adato et al. [12] show that it is possible to relax the hypothesis of calibrated scene by analyzing the reflection of a time varying distant unknown scene observed by an orthographic camera.

Additional constraints for estimating the reflection point \mathbf{r} (along with higher-order shape information) can be obtained by having multiple observations of \mathbf{r} from different vantage points. These observations can be generated from a moving camera. Unfortunately, standard structure from motion (SFM) constraints based on epipolar geometry [13] cannot be used in this case: As the camera vantage point changes, reflected points move (flow) on the surface following the law of reflection; this violates the assumption of static 3D points which is essential in SFM methods. As shown in the pioneering studies by Koenderink and van Doorn [14], Blake [15], and Zisserman et al. [2], even when multiple observations of the same reflected point \mathbf{r} are available, \mathbf{r} and \mathbf{n}_r can still be estimated up to one-dimensional family of solutions

(whereas the local concave/convex shape ambiguity can be determined [15]). Similarly to the case of static cameras, additional constraints are required to estimate the unknown parameters. The theoretical analysis in Oren and Nayar [16] and more recently the level set formulation by Solem et al. [17] demonstrate that camera movements allow to recover 3D surface profiles if boundary conditions at object occluding contours are used. Bonfort and Sturm [18] develop a discrete multi-view approach where the surfaces reflect a calibrated 3D scene using a volumetric stereo framework [19]. Roth and Black [20] relax the assumption of calibrated scene by introducing a probabilistic formulation-based expectation-maximization that combines cues from specular and diffuse components, the former being defined as *specular flow*.

Application

Modeling the shape of reflective surfaces is valuable in numerous research and industrial applications such as digital archival (e.g., acquisition of digital models for preservation of artistic artifacts with reflective components), medicine (e.g., noninvasive inspection of organs such as the cornea of the eye [7]), and metrology of industrial parts.

References

1. Savarese S, Chen M, Perona P (2005) Local shape from mirror reflections. *Int J Comput Vision* 64(1):31–67
2. Zisserman A, Giblin P, Blake A (1989) The information available to a moving observer from specularities. *Image Video Comput* 7:38–42
3. Ikeuchi K (1981) Determining surface orientation of specular surfaces by using the photometric stereo method. *IEEE J Pattern Anal Mach Intell* 3:661–669
4. Torrance K, Sparrow E (1976) Theory for off-specular reflection from roughened surfaces. *J Opt Soc Am* 57: 1105–1114
5. Healey G, Binford T (1988) Local shape from specularity. *Comput Vision Graphics Image Process* 42:62–86
6. Nayar SK, Sanderson AC, Weiss LE, Simon DA (1990) Specular surface inspection using structured highlight and gaussian images. *T-RA* 6:208–218
7. Halstead M, Barsky B, Klein S, Mandell R (1996) Reconstructing curved surfaces from specular reflection patterns using spline surface fitting of normals. In: *ACM SIGGRAPH*, New Orleans, pp 335–342
8. Tarini M, Lenz H, Goesele M, Seidel H (2002) Shape from distortion: 3d range scanning of mirroring objects. In: *Proceedings of the SIGGRAPH, sketches and applications*. ACM, New York, p 248
9. Zheng J, Murata A (2000) Acquiring a complete 3d model from specular motion under the illumination of circular-shaped light sources. *IEEE J Pattern Anal Mach Intell* 8:913–920
10. Rozenfeld S, Shimshoni I, Lindenbaum M (2010) Dense mirroring surface recovery from 1d homographies and sparse correspondences. *IEEE Trans Pattern Anal Mach Intell* 99:325–337
11. Steger E, Kutulakos KN (2008) A theory of refractive and specular 3d shape by light-path triangulation. *Int J Comput Vision* 76(1):13–29
12. Adato Y, Vasilyev Y, Ben-Shahar O, Zickler T (2007) Towards a theory of shape from specular flow. In: *Proceedings of the IEEE international conference on computer vision (ICCV)*, Rio de Janeiro
13. Hartley RI, Zisserman A (2004) *Multiple view geometry in computer vision*, 2nd edn. Cambridge University Press, Cambridge, ISBN:0521540518
14. Koenderink J, van Doorn A (1980) Photometric invariants related to solid shape. *Optica Acta* 27:981–996
15. Blake A (1985) Specular stereo. In: *IJCAI*, Los Angeles, pp 973–976
16. Oren M, Nayar SK (1997) A theory of specular surface geometry. *Int J Comput Vision* 24(2):105–124
17. Solem J, Aanæs H, Heyden A (2003) A variational analysis of shape from specularities using sparse data. In: *International symposium on 3D data processing visualization and transmission*, Padova, Italy
18. Bonfort T, Sturm P (2003) Voxel carving for specular surfaces. In: *Proceedings of the IEEE international conference on computer vision (ICCV)*, Nice, pp 394–403
19. Kutulakos KN, Seitz SM (1999) A theory of shape by space carving. In: *Proceedings of the seventh IEEE international conference on computer vision*, Kerkyra, Greece, pp 307–313
20. Roth S, Black MJ (2006) Specular flow and the recovery of surface structure. In: *Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR)*, New York, pp 1869–1876

Shock Graph

Sven J. Dickinson¹, Ali Shokoufandeh² and
Kaleem Siddiqi³

¹Department of Computer Science, University of Toronto, Toronto, ON, Canada

²Department of Computer Science, Drexel University Philadelphia, PA, USA

³School of Computer Science, McGill University, Montreal, PQ, Canada

Related Concepts

- [Many-to-Many Graph Matching](#); ► [Object Class Recognition \(Categorization\)](#)

Definition

The shock graph is obtained from the 2D Blum medial axis by incorporating properties of the radius function along the skeleton. The direction in which the radius function increases, or equivalently, the direction of the grassfire flow, is used to order groups of skeletal points and to derive parent-child relationships. This results in a directed acyclic graph whose nodes represent skeletal points and whose edges represent adjacency relationships. A variant of this construction associates skeletal points with edges, with the nodes representing the adjacencies.

Background

When Blum conceived of the medial axis or skeleton, his goal was to use it as a means to categorize objects from their projected (2D) outlines [4]. Specifically, by associating the direction of increasing radius value along a skeletal branch, or equivalently the direction of propagation of singularities of the grassfire flow, he proposed the concept of an axis-morphology or *a-morph* by which to achieve object categorization. His basic insight was that this could lead to a decomposition that reflected the qualitative part structure of the object. As an example, ignoring their detailed boundary geometry, outlines of hands would have similar a-morphs and these would be quite distinct from those of outlines of humans, fish or other object classes. In fact, he drew upon these later examples towards the end of his classic paper [4], where he also sketched possible extensions to 3D.

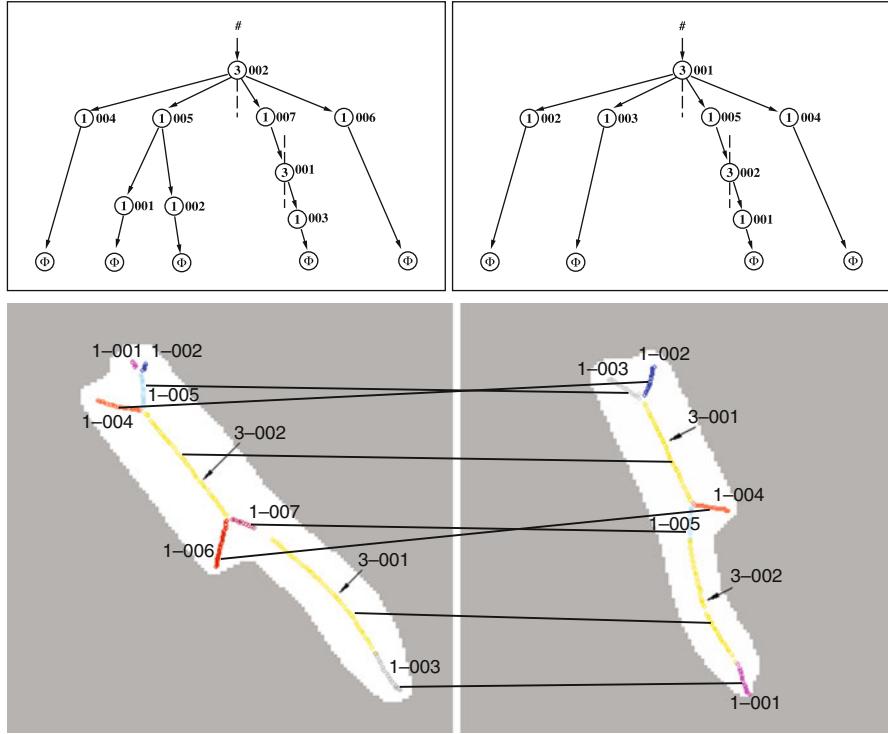
Whereas much has been written about medial or skeletal representations over the years (see [23] and also the medial axis/skeleton entry in this encyclopedia) the idea that an a-morph was essentially a directed graph which could be used for object recognition caught on only in the early 1990s. One likely reason is that it took the image analysis and computer vision communities many years to develop robust algorithms for skeleton computation. Since this time, however, a variety of successful approaches to view-based recognition using shock-graphs have been proposed and have been validated on large databases. Several of these are described in the present entry. There also

exist more recent variants of the shock graph, such as Macrini et al.'s bone graph [14], which attempt to mitigate the representational instability of the Blum medial axis. In fact, the mapping of the Blum skeleton to a graph-based representation, of which the shock graph is the most widely researched example, remains an active area of investigation.

Theory

The Blum medial axis or skeleton of a 2D outline is homotopic to it and is comprised of three types of skeletal points: endpoints of skeletal curves, interior points and branch points. The branch points are generically of degree 3, i.e., three skeletal curves are connected at a branch point. A formal classification is presented in [11]. The shock graph takes the 2D skeleton of a simple closed curve as input (one without holes) and labels each skeletal point according to whether the radius function at it is increasing monotonically (a 1-shock), is a local minimum (a 2-shock), is constant (a 3-shock) or is a local maximum (a 4-shock). Groups of adjacent 1-shocks are considered together, as are groups of 3-shocks. Given this interpretation, a directed acyclic graph is obtained by considering the skeletal points with the largest radii, which are the last to form in the grassfire flow, as the children of a dummy root node. The children are then placed, recursively, in order of decreasing radius value. This process of reversing the grassfire flow and adding 1-shock groups or 3-shock groups as children, is governed by the rules of a grammar, as shown in [24].

Rather than provide all the details of the grammar in this entry, the reader is referred to the examples in Fig. 1, which show the construction of the shock graphs of two brush shapes. The medial axis of each object is shown in the bottom row, with distinct groups of shocks being given a unique color (3-shocks are shown in yellow). In the labeling, the shock type appears first, followed by a unique identifier. The associated shock graphs are shown in the top row. It is clear that each shape is abstracted by a single root node (the 3-shock group describing the elongated portion of the brush), with its children being additional protrusions (1-shock groups). One of these protrusions has a 3-shock group as a child, which describes the handle of each brush. From this example it is evident



Shock Graph, Fig. 1 The shock graphs derived for two different views of a brush using the algorithm of Siddiqi et al. [24] are represented in the top row. The bottom row depicts the

correspondences between nodes in the shock graphs computed by the matching algorithm

that the shock graph is a formalization of Blum's α -morph, with the advantage that it lends itself to the use of graph-based methods for object categorization, as detailed below.

It is also important to point out that there is a variant of the shock graph where the representation places the skeletal points at edges of the graph, with the nodes representing connections. This variant is described in detail in [18, 19]. This representation has led to different but equally successful methods for object recognition, based on a notion of the edit-distance between two graphs. The results of using this approach are also briefly described below.

Shock Graph-Based Object Categorization

An object categorization system based on shock graphs consists of two components: (1) an indexing component, which takes an input shock graph and returns, from a large database of model shock graphs, a small

number of candidate shock graphs that might account for the input; and (2) a matching component, which takes one of the candidates and the input, and computes a similarity (or distance), along with a set of node correspondences. Under ideal conditions, the input shock graph would contain no artifacts due to noise, occlusion, or clutter, and would be isomorphic to one of the model shock graphs (provided that the input object represents one of the model objects). However, such conditions are highly unlikely, for in addition to noise, occlusion, and scene clutter, ligature-induced instabilities [1] often lead to spurious nodes/edges as well as medial branch oversegmentation. Formulating the problem as graph isomorphism, subgraph isomorphism, or even largest isomorphic subgraph will not lead to a meaningful solution, for large, or even significant isomorphisms may simply not exist between two shock graphs that represent instances of the same category. The shock graph indexing and matching problems are therefore inexact graph indexing/matching problems.

Indexing Shock Graphs

Given an input shock graph, the goal of the indexing module is to quickly (sublinearly) retrieve a small number of candidate model database shock graphs among which the input is likely included. As mentioned above, the input shock graph may be corrupted in a number of ways, precluding a simple global (based on the entire input) indexing framework. For example: (1) occlusion may remove part of the input shock graph and replace the missing part with a shock graph (or subgraph) belonging to a different object; (2) shadows or poor illumination may simply delete some portion of the input shock graph; (3) scene clutter may embed the object shock graph (or portion thereof) in a much larger “scene” shock graph; and (4) ligature-based instability may introduce spurious nodes or may overpartition other nodes in the input shock graph. These factors require a part-based indexing framework that can operate in the presence of noise, occlusion, clutter, and ligature-based instability.

One such indexing framework that is applicable to not only shock graphs but any hierarchical, graph-based representation (specifically, any directed acyclic graph-based representation) was introduced by Shokoufandeh et al. [22], originally for the purpose of shock graph indexing. The key concept behind the approach is to capture the abstract shape of a graph (or subgraph) with a low-dimensional vector, yielding an efficient indexing mechanism. Capturing the abstract shape of a graph is important so that the index is invariant to noise and minor within-class shape deformation. Indexing at the part level is important in the presence of occlusion and scene clutter. Mapping a discrete graph structure to a low-dimensional point facilitates a simple nearest-neighbor search in a geometric space for similar model parts which, in turn, can vote for those model objects that contain those parts. Those model objects receiving the largest votes represent those candidate objects passed to the shock graph matching module for a more detailed analysis.

The graph-based shape abstraction is computed at every non-leaf node, and captures the abstract “shape” of the underlying subgraph rooted at that node. Therefore, each non-leaf node (with only four shock graph node types, leaf nodes are far too uninformative/ambiguous) “votes” for those objects that share its substructure; the root of the graph would therefore vote at the object level, and would be

meaningful only if the object were unoccluded and not embedded in a larger scene. Mapping the structure of a rooted subgraph to a vector assigned to the subgraph’s root is based on a spectral analysis of the graph’s structure. The eigenvalues of a graph’s adjacency matrix (whose values are $0, 1, -1$) capture important properties of the degree distribution of the graph’s nodes. The eigenvalues can be combined to yield a low-dimensional abstraction of the graph’s shape in terms of how and where the edges are distributed throughout the graph. Moreover, such a spectral “signature,” called the *topological signature vector*, is proven to be stable under minor perturbations of graph structure due to noise. Details of the approach are found in [22], while an application of the same indexing framework to a different hierarchical graph, specifically a 3-D medial surface graph, can be found in [25].

Matching Shock Graphs

Given two shock graphs, e.g., one representing the input and one representing a model candidate, the matching component needs to return not only a similarity or distance measure that can be used to rank order the candidates, but also an explicit correspondence that defines which model nodes correspond to which input nodes. Such correspondence is necessary, for in the case of a cluttered scene, those nodes found to match a given model would be removed, and another candidate model matched to the remaining nodes. Moreover, the correspondence need not be one-to-one, for in the case of ligature-induced medial branch oversegmentation, node correspondence many be *many-to-many*.

Siddiqi et al. [24] developed a matching algorithm for shock graphs which, like the indexing framework of Shokoufandeh et al. [22] discussed above, can be applied to the matching of any directed acyclic graph structure, provided that a domain-dependent node similarity function is given. The algorithm is based on the same spectral graph theoretic abstraction that forms the heart of the indexing component described above. The algorithm formulates the matching of two graphs as finding a maximal matching in a bipartite graph over the two nodes sets (input and model). The edge weights (each spanning one input shock graph node and one model shock graph node) in the graph have two components: (1) the distance between the two nodes’ respective topological similarity vectors, defining the similarity of their underlying graph structures

(rooted at the two nodes); and (2) a node similarity function (the only domain-dependent component of the algorithm) that defines the similarity of the node attributes (for shock graphs, this encodes the geometric similarity between the two skeletal branches corresponding to the two nodes).

At first glance, the matching algorithm would seem to throw out all the important hierarchical structure in the two graphs (absent in the bipartite graph); nodes in one graph are matched to nodes in the other graph, but the edges in the two original graphs appear to play no role. However, the key contribution of the algorithm is that the hierarchical edge structure is brought back via the topological signature vector similarity term. For the bipartite matching algorithm to match two nodes (i.e., select that edge in the matching), both their geometric similarity and their topological similarity must be high. In other words, the contents of the two nodes must be similar and the subgraphs rooted at the two nodes must be similar. The algorithm iterates by computing a matching, selecting the best edge from the matching (having maximum similarity), adding it to the solution set, and recursively continuing the process on the remaining graphs (after removing the pair of matching nodes defined by the best edge). Details of the approach are found in [24], while its application to other shape matching problems is described in [21] (multiscale blob and ridge graphs), [25] (medial surface graphs), and [8, 26] (curve skeleton graphs).

The above algorithm eventually yields a one-to-one node correspondence between the two graphs. However, because the algorithm generates the node correspondence in a coarse-to-fine manner, stopping the algorithm at the level of a coarse node-to-node correspondence defines an explicit many-to-many correspondence between the nodes in the subgraphs rooted at the coarse nodes. Moreover, since the topological signature vectors are stable under small amounts of additive graph noise, similarity can remain high even though the two subgraphs may have different numbers of nodes. As the cardinalities of the two graphs' node sets begin to differ more dramatically, for example due to heavy under- or over-segmentation, the method breaks down and more powerful many-to-many graph matching must be employed.

One such method for many-to-many graph matching of medial axis-based graphs was proposed by Demirci et al. [9, 10]. Their algorithm transforms the

graphs into a finite dimension metric space in which an approximate solution to the many-to-many matching problem becomes tractable. The embedding step will result in a set of points, each representing a vertex of the original graph. Their proposed embedding has the additional property that pairwise distances between points in the target metric space closely resemble the shortest-path distances between the corresponding nodes in the graphs. Matching two graphs can then be formulated as the problem of matching their two embeddings. The many-to-many matching of the two embeddings then can be computed by solving a transportation problem using the Earth Mover's Distance algorithm [7]. The solution of this latter problem computes the mass which flows from one weighted point set to another that minimize the total transportation cost. The computed flows, in turn, define the many-to-many node correspondences between the original graphs.

The problem of matching shock graphs has also been studied in the context of edit-distance methods [18, 29]. These algorithms estimate the cost of matching as a function of edit operations, including node relabelings, additions and deletions, and edge contraction that transform one graph into another. A fundamental issue in devising algorithms based on edit-distance is the choice of cost of each operation. Torsello and Hancock [29] use the heuristic proposed by Bunke [5] for the cost associated with their edit operations. For example, the cost of relabeling elements is less than the cost of performing a deletion followed by inserting a new node with a new label. In contrast, Sebastian et al. [18] propose a multi-step heuristic to derive their edit costs. Their overall heuristic is centered around the notion of a shape cell, i.e., a collection of shapes which have identical shock graph topology. They define the cost of the deformation operation as a function of the discrepancy between matching shock attributes of shapes within a given cell. The cost associated with other edit operations is derived as the limit of the deformation cost when a shape moves to the boundary a shape cell.

Caelli and Kosinov [6] show how inexact matching can be utilized for measuring shape similarity between shock graphs. Their method establishes correspondence between sets (clusters) of vertices of two given graphs and as such can be viewed as a many-to-many matching approach. Their algorithm can be viewed as

| Instance | Distance to Class Prototype | | | | | | | | |
|----------|-----------------------------|-------|-------|-------|-------|-------|-------|-------|-------|
| | | | | | | | | | |
| | 0.02 | 2.17 | 4.48 | 3.55 | 2.96 | 0.21 | 4.58 | 14.33 | 10.01 |
| | 2.39 | 0.10 | 5.97 | 15.90 | 3.98 | 0.14 | 26.12 | 17.28 | 28.94 |
| | 10.89 | 4.72 | 2.08 | 12.24 | 3.12 | 2.15 | 19.73 | 10.11 | 12.64 |
| | 7.15 | 6.42 | 1.19 | 1.35 | 5.10 | 3.38 | 10.58 | 11.11 | 11.11 |
| | 4.08 | 7.72 | 2.98 | 1.49 | 4.26 | 4.14 | 26.60 | 13.54 | 14.21 |
| | 14.77 | 6.72 | 5.69 | 0.36 | 2.30 | 5.90 | 10.58 | 16.25 | 19.10 |
| | 7.86 | 8.90 | 5.94 | 0.74 | 1.59 | 1.10 | 10.81 | 10.39 | 16.08 |
| | 2.66 | 4.23 | 3.23 | 6.47 | 0.62 | 1.48 | 11.73 | 15.38 | 15.15 |
| | 3.18 | 5.31 | 1.25 | 4.64 | 0.60 | 1.30 | 14.18 | 17.22 | 9.08 |
| | 4.55 | 0.76 | 1.32 | 2.86 | 1.49 | 0.11 | 21.38 | 15.35 | 13.04 |
| | 6.77 | 19.46 | 22.11 | 13.27 | 8.21 | 29.50 | 0.15 | 5.12 | 5.03 |
| | 8.73 | 23.14 | 31.45 | 24.41 | 10.16 | 31.08 | 0.18 | 8.45 | 7.05 |
| | 12.46 | 19.0 | 27.40 | 14.58 | 24.26 | 17.10 | 8.85 | 7.49 | 16.93 |
| | 13.86 | 23.07 | 12.81 | 11.24 | 17.48 | 23.23 | 6.02 | 6.92 | 3.06 |
| | 15.73 | 21.28 | 14.10 | 12.46 | 19.56 | 19.21 | 9.53 | 7.12 | 5.06 |

Shock Graph, Fig. 2 Similarity between database and class prototypes computed using the algorithm of Siddiqi et al. [24]. In each row, a box is drawn around the most similar shape

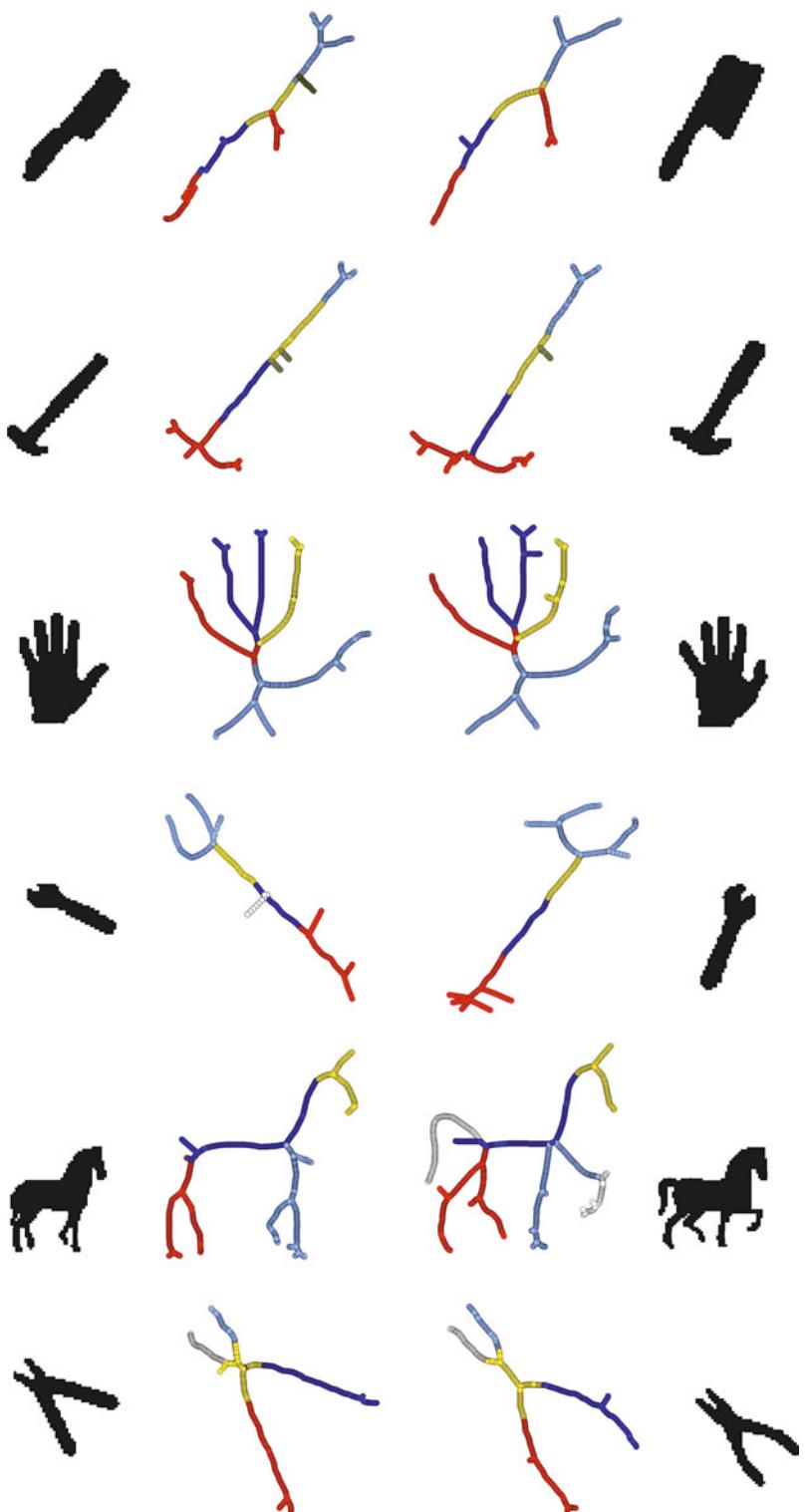
a generalization of the approach of Scott and Longuet-Higgins [17]. The actual matching is established using the renormalization of projections of vertices into the eigenspaces of graphs combined with a form of relational clustering. Similar to other inexact matching algorithms, their eigenspace renormalization projection clustering method is able to match graphs with different numbers of vertices.

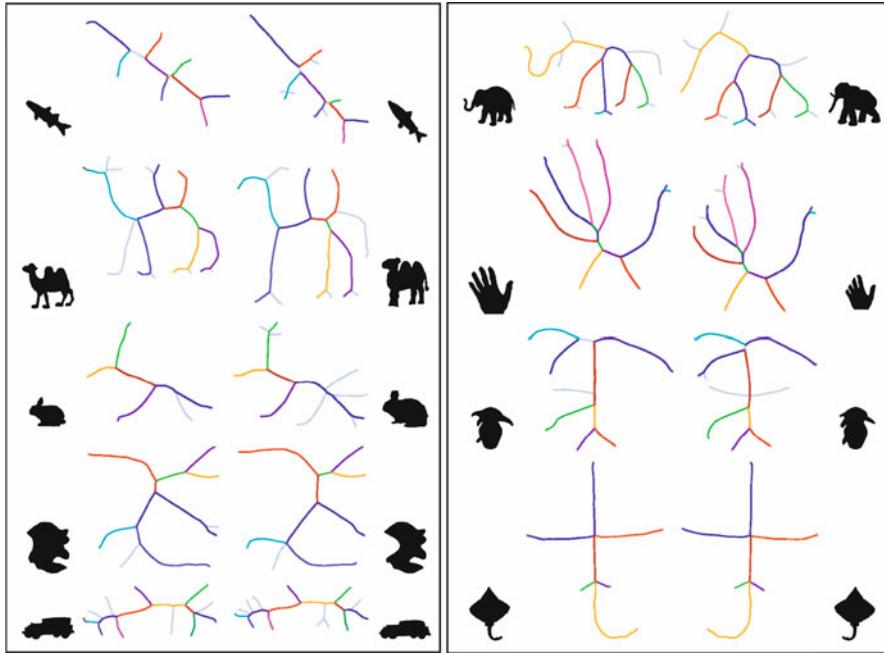
Experimental Results

This section presents some examples of shock graphs and their matchings using the approaches described above. Figure 1(top) illustrates two shock graphs, describing different views of a brush, computed by the algorithm of Siddiqi et al. [24]. The underlying shocks, along with the computed matchings between segments (nodes), are shown in Fig. 1(bottom).

Figure 2 represents the ability of the algorithm to compare objects based on their prototypical or coarse shape. Here, columns 2 through 10 denote the prototype views for each of nine object classes. The similarity between the prototypes and some of the objects in the database is reflected in the rows of this table. For each row, a box has been placed around the most similar shape. Demirci et al. [9] also evaluated the effectiveness of their matching algorithm for shape retrieval based on shock graphs from the Rutgers Tool Database [24]. Figure 3 shows some examples of the many-to-many feature matching results obtained from the algorithm for some of the objects in the Rutgers Tools Database. Finally, Fig. 4 shows the results obtained from applying the edit-distance algorithm of Sebastian et al. [18] to the matching of shock segments. Note that their edit distance algorithm will also produce a sequence of intermediate shock graphs that identify the steps of the transformation of one input shock graph to another.

Shock Graph, Fig. 3 The results of matching skeleton graphs for some pairs of shapes in the Rutgers Tools Database using the algorithm of Demirci et al. [9]. Corresponding segments are shown using the same color. Observe that correspondences are intuitive in all cases





Shock Graph, Fig. 4 The matching results for a few shock graphs produced by the edit-distance algorithm of Sebastian et al. [18]. Matching shock branches are shown using the same

color, while the gray colored edges in the shock graphs indicate that they are spliced or contracted

Open Problems

Symmetry is a powerful shape regularity that has formed the basis of many shape representations, including generalized cylinders [3], superquadrics [16], and geons [2]. Just as geons provide a qualitative and discrete shape abstraction of a generalized cylinder, shock graphs provide a discrete and qualitative shape abstraction of a medial axis. The resulting graph is ideally suited to shape categorization, for it is part-based, is stable under within-class deformation, and is stable under part articulation. However, the shock graph also faces some important challenges. First of all, it assumes that a closed contour has been recovered from an image, separating figure from background. While figure-ground segmentation remains an open research problem, it is important to note that in a categorization system, a perfect figure-ground separation may not be necessary. If a significant portion of the figure's boundary is correctly segmented, a significant portion of the resulting shock graph may be correct – enough to yield the correct candidate (among the list of returned candidates) during indexing. Still, while a shock graph does preserve

locality of representation, significant figure-ground segmentation errors can propagate through the representation, disrupting it to a degree that prevents effective indexing. A recent attempt to recover a symmetric part decomposition from a cluttered scene has been reported by Levinstein et al. [13], in which symmetric parts are detected locally (bottom-up) and then grouped to form an approximation to a medial axis.

The second challenge facing the shock graph is the ligature-based instability discussed earlier [1]. A number of approaches exist to try and regularize the medial axis through boundary smoothing, e.g., [12, 20, 27]; however, these methods do not effectively address the ligature structure. Other methods have sought to abstract the medial axis by regularizing out small internal branches, e.g., [28, 30]; however these methods don't explicitly target ligature structure. A recent promising approach to abstracting out ligature structure is proposed by Macrini et al. [14, 15], yielding a representation, called the *bone graph*, whose parts are the non-ligature medial branches that represent the salient parts and whose edges represent the “glue” (defined by the ligature branches) that binds the parts.

References

1. August J, Siddiqi K, Zucker S (1999) Ligature instabilities in the perceptual organization of shape. *CVIU* 76(3): 231–243
2. Biederman I (1985) Human image understanding: Recent research and a theory. *Comput Vis Graph Image Process* 32:29–73
3. Binford TO (1971) Visual perception by computer. In: Proceedings of the IEEE conference on systems and control, Miami
4. Harry Blum (1973) Biological shape and visual science. *J Theor Biol* 38:205–287
5. Bunke H (1997) On a relation between graph edit distance and maximum common subgraph. *Pattern Recognit Lett* 18(8):689–694
6. Caelli T, Kosinov S (2004) An eigenspace projection clustering method for inexact graph matching. *IEEE Trans Pattern Anal Mach Intell* 26(4):515–519
7. Cohen SD, Guibas LJ (1999) The earth mover's distance under transformation sets. In: Proceedings of the international conference on computer vision (ICCV), Kerkyra, pp 1076–1083
8. Cornea N, Demirci MF, Silver D, Shokoufandeh A, Dickinson S, Kantor P (2005) 3d object retrieval using many-to-many matching of curve skeletons. In: Proceedings of the international conference on shape modeling and applications (SMI), MIT, Cambridge, pp 368–373
9. Denirci MF, Shokoufandeh A, Keselman Y, Bretzner L, Dickinson S (2006) Object recognition as many-to-many feature matching. *Int J Comput Vis* 69(2):203–222
10. Demirci F, Shokoufandeh A, Dickinson S (2009) Skeletal shape abstraction from examples. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(5): 944–952
11. Giblin Peter J, Kimia Peter J (2003) On the local form and transitions of symmetry sets, medial axes, and shocks. *IJCV* 54(1–3):143–157
12. Katz Robert A, Pizer Stephen M (2003) Untangling the blum medial axis transform. *Int J Comput Vis* 55(2–3): 139–153 .
13. Levinstein A, Sminchisescu C, Dickinson S (2009) Multi-scale symmetric part detection and grouping. In: Proceedings of the international conference on computer vision (ICCV), Kyoto
14. Macrini D, Dickinson S, Fleet D, Siddiqi K (2011) Bone graphs: Medial shape parsing and abstraction. *Comput Vis Image Underst (CVIU)*, special issue on graph-based representations, Special Issue on Graph-Based Representations, 115(7):1044–1061
15. Macrini D, Dickinson S, Fleet D, Siddiqi K (2011) Object categorization using bone graphs. *Computer Vision and Image Understanding (CVIU)* 115(8):1187–1206
16. Pentland A (1986) Perceptual organization and the representation of natural form. *Artif Intell* 28:293–331
17. Scott G, Longuet-Higgins H (1991) An algorithm for associating the features of two patterns. *Proceedings of royal society of london B244:21–26*
18. Sebastian T, Klein P, Kimia B (2001) Recognition of shapes by editing shock graphs. In: Proceedings of the international conference on computer vision (ICCV), Vancouver 755–762
19. Sebastian T, Klein P, Kimia B (2004) Recognition of shapes by editing shock graphs. *IEEE Trans Pattern Anal Mach Intell* 26:550–571
20. Doron Shaked, Bruckstein Alfred M (1998) Pruning medial axes. *Comput Vis Image Underst* 69(2):156–169
21. Shokoufandeh A, Bretzner L, Macrini D, Demirci MF, Jönsson C, Dickinson S (2006) The representation and matching of categorical shape. *Comput Vis Image Underst* 103(2):139–154
22. Shokoufandeh A, Macrini D, Dickinson S, Siddiqi K, Zucker SW (2005) Indexing hierarchical structures using graph spectra. *IEEE Trans Pattern Anal Mach Intell* 27(7):1125–1140
23. Siddiqi K, Pizer Stephen M (2008) Medial representations: mathematics, algorithms and applications. Springer, Dordrecht
24. Siddiqi K, Shokoufandeh A, Dickinson S, Zucker S (1999) Shock graphs and shape matching. *Int J Comput Vis* 30: 1–24
25. Siddiqi K, Zhang J, Macrini D, Shokoufandeh A, Bioux S, Dickinson S (2008) Retrieving articulated 3-d models using medial surfaces. *Mach Vis Appl* 19(4):261–275
26. Sundar H, Silver D, Gagvani N, Dickinson S (2003) Skeleton based shape matching and retrieval. In: Proceedings of the international conference on shape modelling and applications (SMI), Seoul, pp 130–142
27. Hüseyin Tek, Kimia Benjamin B (2001) Boundary smoothing via symmetry transforms. *J Math Imaging Vis* 14(3):211–223
28. Telea A, Sminchisescu C, Dickinson S (2004) Optimal inference for hierarchical skeleton abstraction. In: Proceedings of the international conference on pattern recognition, Cambridge, UK, pp 19–22
29. Torsello A, Hancock ER (2003) Computing approximate tree edit distance using relaxation labeling. *Pattern Recognit Lett* 24(8):1089–1097
30. van Eede M, Macrini D, Telea A, Sminchisescu C, Dickinson S (2006) Canonical skeletons for shape matching. In: Proceedings of the international conference on pattern recognition, Hong Kong, pp 64–69

Shot Noise

► [Photon, Poisson Noise](#)

Simulated Annealing

Juergen Gall

Max Planck Institute for Intelligent Systems,
Tübingen, Germany

Synonyms

[Monte Carlo annealing](#); [Probabilistic hill climbing](#);
[Statistical cooling](#); [Stochastic relaxation](#)

Definition

Simulated annealing is a stochastic computational technique derived from statistical mechanics for finding near globally-minimum-cost solutions to large optimization problems [1].

Background

Many computer vision problems require the minimization of an application dependent objective function in a high-dimensional state space subject to conflicting constraints. Finding the global minimum can be an NP-complete problem since the objective function tends to have many local minima. A procedure for solving hard optimization problems should sample values of the objective function in such a way as to have a high probability of finding a near-optimal solution and should also lend itself to efficient implementation. A method which meets these criteria was introduced by Kirkpatrick et al. [2] and independently by Černý [3] in the early 1980s. They introduced the concepts of annealing in combinatorial optimization. These concepts are based on a strong analogy between the physical annealing process of solids and the problem of solving large combinatorial optimization problems.

Theory

Statistical mechanics is the study of the behavior of very large systems of interacting components, such as atoms in a fluid, in thermal equilibrium at a finite temperature. If the system is in thermal equilibrium at a given temperature T , then the probability $\pi_T(s)$ that the system is in a given state s depends upon the energy $E(s)$ of the state and follows the Boltzmann distribution:

$$\pi_T(s) = \frac{\exp\left(-\frac{E(s)}{k_B T}\right)}{\sum_{s' \in \Omega} \exp\left(-\frac{E(s')}{k_B T}\right)}, \quad (1)$$

where k_B denotes a physical constant known as the Boltzmann constant and Ω the set of all possible states.

Using a technique developed by Metropolis et al. [4], one can simulate the behavior of a system

of particles in thermal equilibrium at temperature T . Suppose that at time t the system is in state q_t . Then, a subsequent state r_{t+1} is generated by a perturbation mechanism which transforms the current state into a next state. If the energy difference, $E(r_{t+1}) - E(q_t)$, is less than or equal 0, the state r_{t+1} is accepted as current state. Otherwise, the state r_{t+1} is accepted with probability

$$p = \frac{\pi_T(r_{t+1})}{\pi_T(q_t)} = \exp\left(-\frac{E(r_{t+1}) - E(q_t)}{k_B T}\right). \quad (2)$$

Since the method accepts states that decrease the energy as well as those that increase the energy, it is the principle that avoids entrapment at a local minimum. It can be shown that as $t \rightarrow \infty$, the probability that the system is in a given state s equals $\pi_T(s)$, and thus that the distribution of states generated converges to the Boltzmann distribution [5].

In order to obtain a low-energy state of the energy function E , one must use an annealing process, where the temperature of the system is elevated, and then gradually lowered, spending enough time at each temperature to reach thermal equilibrium. Applying simulated annealing to an optimization problem in computer vision where the energy function becomes the objective function to minimize, requires two ingredients:

- An annealing schedule consisting of a starting temperature, a decreasing set of temperatures, and the amount of time to spend at each temperature;
- A perturbation mechanism that generates new states.

The annealing algorithm proposed by Kirkpatrick et al. [2] consists of running the Metropolis-Hastings algorithm [4, 6] at each temperature in the annealing schedule for the amount of time prescribed by the schedule, and selecting the final state as a near-optimal solution.

Geman and Geman [5] applied simulated annealing to image restoration and determined an annealing schedule sufficient for convergence. Specifically, for a given sequence of temperatures $\{T_t\}$ such that $T_t \rightarrow 0$ as $t \rightarrow \infty$ and $T_t \geq \frac{c}{\log(t)}$ for a large constant c , the probability that the system is in configuration s as $t \rightarrow \infty$ is equal to $\pi_0(s)$. For a finite set Ω , the optimal annealing schedule for the convergence of the generated states to the set of global minima with probability 1 was determined by Hajek [7].

Simulated annealing is not limited to discrete state spaces. It can also be applied to minimize objective functions defined on Euclidean spaces, i.e., $\Omega \subseteq \mathbb{R}^d$, where similar convergence results have been proved [8]. Further convergence results and detailed discussions on simulated annealing are given in the books [9, 10].

In the discrete case, the perturbation mechanism depends usually on the application where some examples are given in [9]. In the simplest case, a new state is randomly sampled from a local neighborhood of the previous state, e.g., by permutations, swapping, or inversions. In the continuous case, the visiting distribution can be modeled as Gaussian distribution that favors local search, the algorithm is also called Boltzmann machine, but also other distributions like the Cauchy-Lorentz distribution, known as fast simulated annealing [11], can be applied. This distribution results in frequently local searches, but can also generate a state that is very distant to the current state. Having the state q_t , a new state $r_{t+1} = q_t + \Delta s$ is generated by sampling from

$$g_t(\Delta s) = \frac{1}{(\pi T_t)^{\frac{d}{2}}} \exp\left(-\frac{\|\Delta s\|^2}{T_t}\right) \quad (\text{Boltzmann machine}), \quad (3)$$

$$g_t(\Delta s) = \frac{\Gamma(\frac{d+1}{2})}{\pi^{\frac{d+1}{2}}} \frac{T_t}{(\|\Delta s\|^2 + T_t^2)^{\frac{d+1}{2}}} \quad (\text{Cauchy machine}). \quad (4)$$

The new state is then accepted according to the probability p (2). An approach that covers Gaussian and Cauchy-Lorentz distribution as special cases is called generalized simulated annealing [12]. The generalized acceptance probability (2) reads

$$p = \left(1 + \frac{(a-1)(E(r_{t+1}) - E(q_t))}{T_t^a}\right)^{-\frac{1}{a-1}}, \quad (5)$$

the generalized visiting distribution is defined by

$$g_t^v(\Delta s) = \left(\frac{v-1}{\pi}\right)^{\frac{d}{2}} \frac{\Gamma\left(\frac{1}{v-1} + \frac{d-1}{2}\right)}{\Gamma\left(\frac{1}{v-1} - \frac{1}{2}\right)} \frac{(T_t^v)^{-\frac{d}{3-v}}}{\left(1 + (v-1)\frac{\|\Delta s\|^2}{(T_t^v)^{\frac{2}{3-v}}}\right)^{\frac{1}{v-1} + \frac{d-1}{2}}}, \quad (6)$$

and the generalized annealing schedule is given by

$$T_t^b = T_1^b \frac{2^{b-1} - 1}{(1+t)^{b-1} - 1} \quad b \in \{a, v\}. \quad (7)$$

The additional parameters (a, v) give an additional flexibility where $(1, 1)$ corresponds to the Boltzmann machine and $(1, 2)$ to the Cauchy machine. In practice, the optimal choice of (a, v) depends on the objective function and needs to be empirically determined.

For speeding up simulated annealing, the algorithm can be implemented in parallel [9]. The annealing principle is also used for other optimization methods. For instance, a deterministic annealing method has been proposed for clustering [13] or an interacting particle system with annealing properties has been proposed in [14] and applied to human motion capture [15], where at each iteration a set of particles estimates the current distribution.

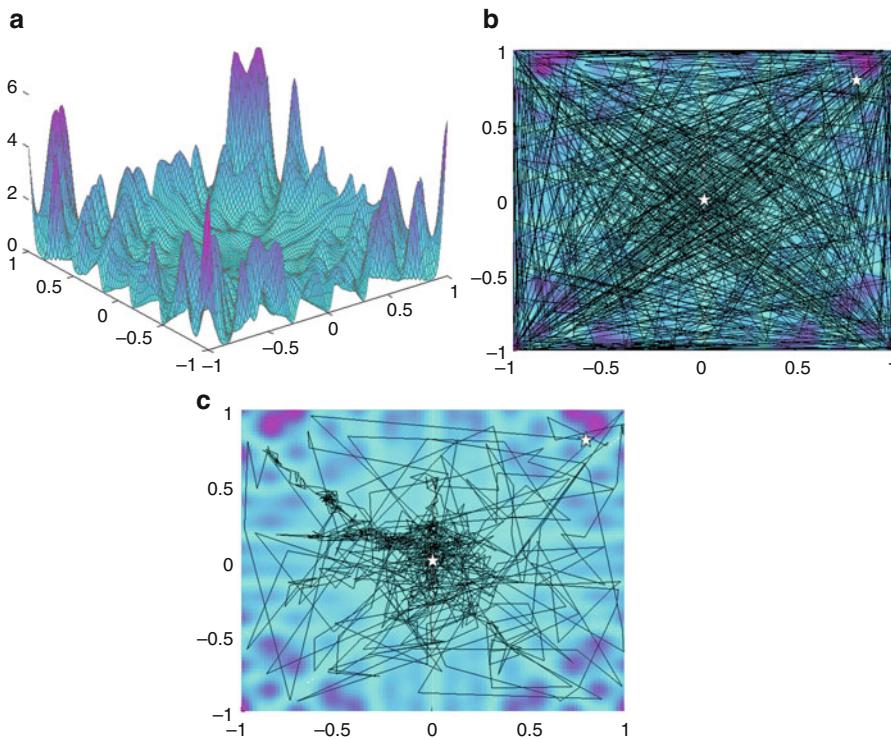
Application

Since its introduction in 1983, simulated annealing has been applied for solving complex, non-convex optimization problems in image processing and computer vision. An overview of applications in the 1980s including the work of Geman and Geman [5] is given in [9]. Nowadays, simulated annealing is still an easy-to-implement and practically useful tool for solving a wide spectrum of optimization problems, particularly for solving hard problems where no completely successful heuristics exist. For an optimal performance, however, the algorithm needs to be tailored to the problem at hand.

Experimental Results

The following example for global optimization in \mathbb{R}^2 is taken from [16]: Let

$$\begin{aligned} h(x, y) = & (x \sin(20y) + y \sin(20x))^2 \cdot \cosh \\ & (\sin(10x)x + (x \cos(10y) - y \sin(10x))^2 \cdot \cosh(\cos(20y)y) \end{aligned} \quad (8)$$



Simulated Annealing, Fig. 1 (a) Function $h(x, y)$. (b) Visited points using Boltzmann machine and initial temperature 100. The starting point, $(0.8, 0.8)$, and the final estimate are indicated

by a white star. (c) Visited points using Cauchy machine. The Cauchy machine cools down faster, focusing the search more on low energy regions

Simulated Annealing, Table 1 Average and standard deviation for the obtained estimate (\hat{x}, \hat{y}) , its function value $h(\hat{x}, \hat{y})$, and the number of required function evaluations at different initial temperatures T_1 .

| | T_1 | (\hat{x}, \hat{y}) | $h(\hat{x}, \hat{y}) \times 10^5$ | # function evaluations |
|-----------|-------|-----------------------------------|-----------------------------------|------------------------|
| Boltzmann | 1 | $(-0.02, 0.04) \pm (0.23, 0.37)$ | 1.14 ± 2.00 | $1,681.9 \pm 507.5$ |
| Cauchy | 1 | $(0.42, 0.38) \pm (0.38, 0.43)$ | 700.61 ± 3241.27 | $1,504.4 \pm 461.0$ |
| Boltzmann | 10 | $(-0.04, -0.01) \pm (0.38, 0.51)$ | 7.16 ± 12.01 | $1,740.9 \pm 573.0$ |
| Cauchy | 10 | $(0.00, -0.02) \pm (0.46, 0.39)$ | 0.20 ± 0.55 | $1,732.0 \pm 580.7$ |
| Boltzmann | 100 | $(-0.02, 0.03) \pm (0.34, 0.33)$ | 2.41 ± 4.02 | $1,733.9 \pm 560.8$ |
| Cauchy | 100 | $(0.00, 0.02) \pm (0.19, 0.24)$ | 0.50 ± 1.12 | $1,802.6 \pm 494.0$ |

be the objective function to minimize (Fig. 1). Its global minimum is 0, attained at $(x, y) = (0, 0)$.

In Table 1, results are shown for the Boltzmann and the Cauchy machine. The starting point was $(0.8, 0.8)$ and each simulation was repeated 100 times. Each simulation was stopped when the average change in value of the objective function in 1,000 iterations was less than 10^{-6} . The table contains the average and standard deviation for the obtained estimate (\hat{x}, \hat{y}) , its function value $h(\hat{x}, \hat{y})$, and the number of necessary function evaluations.

References

1. Davis L (1987) Genetic algorithms and simulated annealing. Morgan Kaufmann, San Francisco
2. Kirkpatrick S, Jr CG, Vecchi M (1983) Optimization by simulated annealing. Science 220(4598):671–680
3. Černý V (1985) Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. J Optim Theory Appl 45(1):41–51
4. Metropolis N, Rosenbluth A, Rosenbluth M, Teller A, Teller E (1953) Equations of state calculations by fast computing machines. J Chem Phys 21(6): 1087–1092

5. Geman S, Geman D (1984) Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images. *IEEE Trans Pattern Anal Mach Intell* 6:721–741
6. Hastings W (1970) Monte carlo sampling methods using markov chains and their applications. *Biometrika* 57(1): 97–109
7. Hajek B (1988) Cooling schedules for optimal annealing. *Math Oper Res* 13(2):311–329
8. Royer G (1989) A remark on simulated annealing of diffusion processes. *SIAM J Control Optim* 27(6):1403–1408
9. Aarts E, Korst J (1989) Simulated annealing and Boltzmann machines: a stochastical approach to combinatorial optimization and neural computing. Wiley, New York
10. Gidas B (1995) 7: Metropolis-type Monte Carlo simulation algorithms and simulated annealing. In: Topics in contemporary probability and its applications. CRC, Boca Raton, pp 159–232
11. Szu H, Hartley R (1987) Fast simulated annealing. *Phys Lett A* 122:157–162
12. Tsallis C, Stariolo D (1996) Generalized simulated annealing. *Physica A* 233:395–406
13. Rose K (1998) Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proc IEEE* 86(11): 2210–2239
14. Moral PD (2004) Feynman-Kac formulae. Genealogical and interacting particle systems with applications. Springer, New York
15. Gall J, Rosenhahn B, Seidel HP (2008) An Introduction to Interacting Simulated Annealing. In: Human motion – understanding, modeling, capture and animation. Springer, Dordrecht, pp 319–343
16. Robert C, Casella G (2002) Monte Carlo statistical methods. Springer, New York

Single Viewpoint

► [Center of Projection](#)

Situation Graph Trees

Christian Micheloni¹ and Gian Luca Foresti²

¹Department of Mathematics and Computer Science,
University of Udine, Udine, Italy

²Department of Mathematics and Computer Science,
University of Udine, Udine, Italy

Synonyms

[Generically describable situation](#); [Situation scheme](#)

Definition

Situation Graph Trees (SGT) provide a deterministic formalism to represent the knowledge required for human behavior modeling.

Background

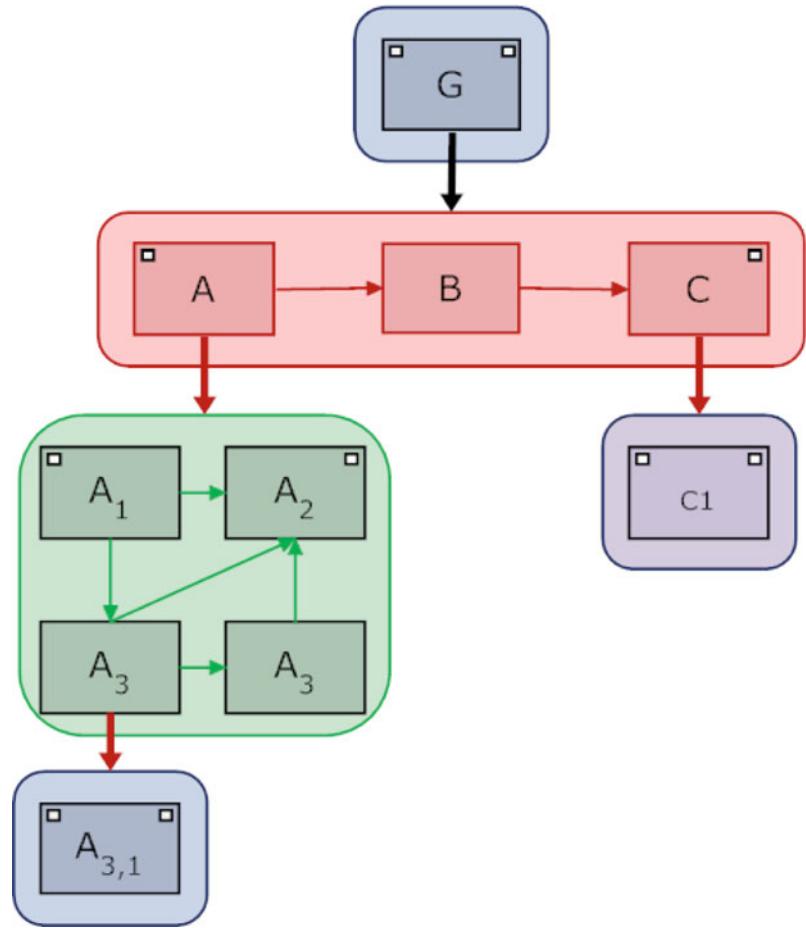
In many domains, high-level descriptions to represent the status of an environment are desirable. Describing a situation requires to conceptualize the knowledge about the possible actions of the actors involved in the environment and their possible interactions. Conceptual descriptions of different application domains like traffic analysis [1], parking lot security [2, 3], and human behavior recognition [4] are of primary importance. The conceptualization can proceed from simple descriptions (simple events) to complex descriptions (complex events). Following relations, concepts can be aggregated into more complex concepts. Hence, an event can be described as a sequence of simple events. To allow such an incremental description of the events, two main processes are required: (a) modeling of the behaviors and (b) the reasoning engine. Different approaches like Bayesian networks [5], Hidden Markov models [6], and SVM [7] have been recently introduced. However, in complex environments, such models are difficult to apply as they would require a large amount of data for training the models. In such cases, a formalism to model behaviors by means of temporal and semantic relationships or specialization of fundamental concepts can be useful. Situational Graph Trees (SGTs) represent such a modeling tool.

Theory

The Situation Graph Trees (SGTs) [4] provide a deterministic formalism to represent the knowledge necessary to describe an actor behavior. Generally, SGTs are based on the description of the situation that consists on an agent state and the possible actions that the agent can actuate in such a state. Thus, a hierarchy of possible situation is defined on temporal and conceptual terms. This means that given a recognized situation only, its possible successor is evaluated at the next time instant.

Situation Graph

Trees, Fig. 1 Example of SGT



The fundamental block of the SGT is the situation scheme that represents the knowledge of an agent for a given time instant. A situation scheme is composed by sections:

- State: describe the state of an agent in terms of predicates
- Action: describe the possible and supposed actions that an agent can do whenever one of the state predicates is satisfied

Situation schemes are connected by means of direct edges (called *prediction edges*) to define a temporal successor relationship between situation schemes. When an agent is instantiated by its predicates, a possible next situation is represented by a scheme pointed by a prediction edge originated by the current situation. If an agent keeps on staying in its current situation for more than a single time instant, *self-prediction edges*, starting from a situation and pointing to itself, are used to model such a behavior. Thus, a situation scheme can be a ring of a chain of situation schemes describing a sequence of situations. Such sequences,

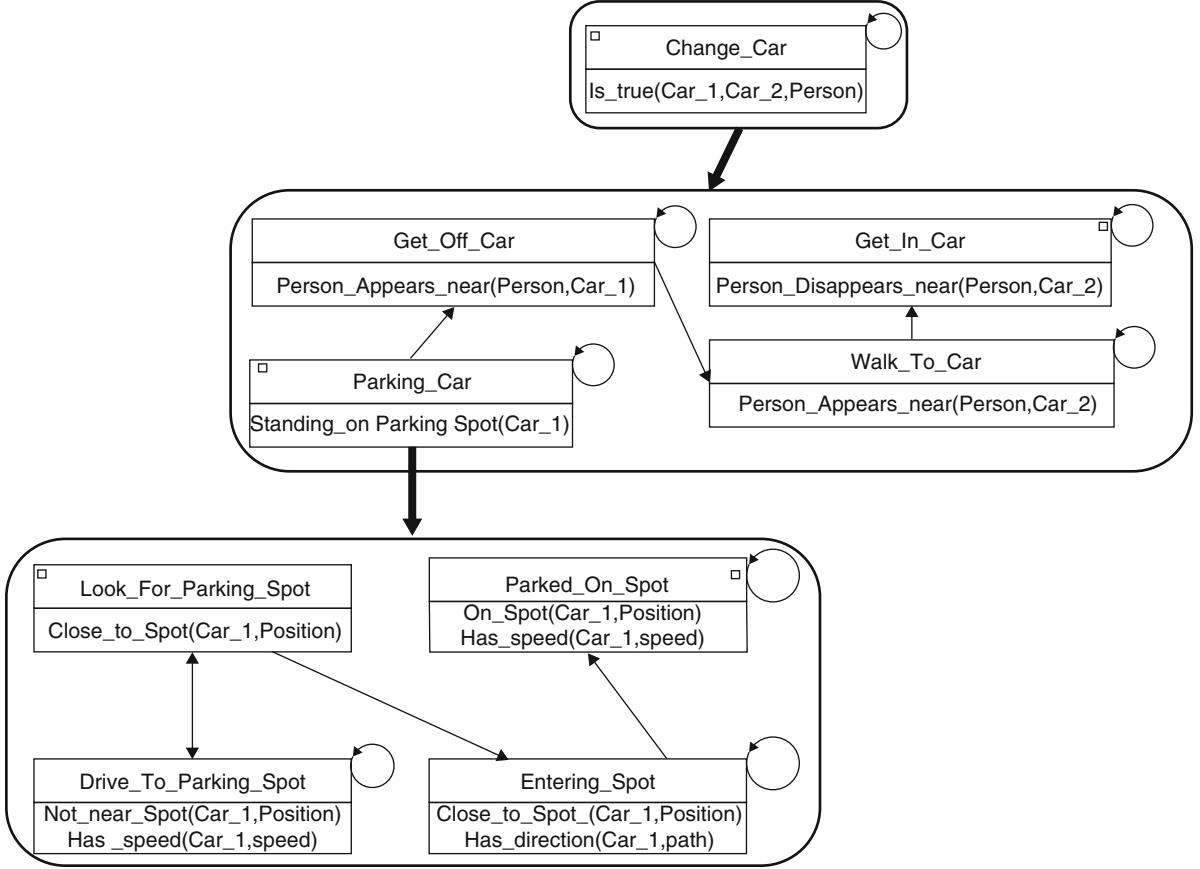
composed by situation schemes and prediction edges, are called situation graphs. A situation can be temporally or conceptually refined by particularizing its situation scheme. In such a case, a situational graph is connected to a situational scheme by means of a particularized edge. Following [4], it is possible to derive the following definitions for SGTs.

Definition 1 (SGT-Episode) *Any sequence E of situations inside a situation graph G that is a path from a start situation to a end situation is defined as SGT-Episode.*

Definition 2 (Particularized SGT-Episode) *Any SGT-Episode of a situation graph G particularizing a situation scheme s is defined as a particularizing SGT-Episode.*

Definition 3 (SGT-Event) *Given a SGT T, an event is defined as:*

- Any SGT-Episode E within the root situation graph of T.
- Given an SGT-Event E, replacing any situation S in E with a particularized SGT-Episode of S is a new SGT-Event.



Situation Graph Trees, Fig. 2 Example of SGT for an event of interest in a parking lot monitoring context

Definition 4 (Particularized SGT-Event) *Given a SGT T and a SGT-Event E , a SGT-Event \hat{E} is defined as a particularized SGT-Event of E iff \hat{E} is obtained from E by substituting any situation scheme S of E with a particularized SGT-Episode of S .*

Definition 5 (Maximal Event) *Given a SGT T , any SGT-Event E is a maximal event iff there not exists a situation scheme S in E that can be substituted with a particularized SGT-Episode of S .*

Definition 6 (Compatible Events) *Given a SGT T , E and E' are compatible events iff:*

- *Each situation scheme in E is a situation scheme in E' .*
- *For each pair (S_i, S_j) in E where S_i proceed S_j , there is the same pair in E' .*

Representation

The common representation of SGTs is given on Fig. 1. Situation schemes are represented as rectangles and situation graphs as set of rectangles with rounded corners. Start situations (end situations) are depicted with small rectangles in the upper left (right)

corner of the situation schemes. Prediction edges are thin arrows that decide the current situation and the possible next situation. Thick arrows represent particularization edges.

Application

A common application of SGTs is the representation of behaviors in video-surveillance context. In such a field, contrarily to anomaly detection algorithm, expected behaviors or behaviors of interest can be defined for all the actors/agents operating inside the monitored environment. To represent such behaviors, SGT can be powerfully exploited. As example in a parking lot, the monitoring application may be interested in detecting and recognizing a person driving into a parking spot, leaving the parked car, and driving away on board of a second car. Such an event can be described by the SGT depicted in Fig. 2 that can be further particularized in the situation schemes of the first child node.

References

1. Haag M, Nagel H (2000) Incremental recognition of traffic situations from video image sequences. *Image Vis Comput* 18(2):137–153
2. Micheloni C, Remagnino P, Eng HL, Geng J (2010) Intelligent monitoring of complex environments. *Intelligent Systems, IEEE* 25(3):12–14
3. Micheloni C, Snidaro L, Foresti G (2009) Exploiting temporal statistics for events analysis and understanding. *Image Vis Comput* 27(10):1459–1469
4. Arens M, Nagel H (2003) Behavioral knowledge representation for the understanding and creation of video sequences. In: German conference on artificial intelligence, Hamburg(GE), pp 149–163
5. Robertson N, Reid I (2006) A general method for human activity recognition in video. *Comput Vis Image Underst* 104(2–3):232–248
6. Galata A, Johnson N, Hogg D (2001) Learning variable-length markov models of behavior. *Comput Vis Image Underst* 81(3):398–413
7. Piciarelli C, Micheloni C, Foresti G (2008) Trajectory-based anomalous event detection. *IEEE Trans Circuits Syst Video Technol* 18(11):1544–1554

Situation Scheme

► [Situation Graph Trees](#)

Solid Texture

► [Bidirectional Texture Function and 3D Texture](#)

Space Curves

► [Curves in Euclidean Three-Space](#)

Sparse Coding

John Wright
Visual Computing Group, Microsoft Research Asia,
Beijing, China

Synonyms

[Sparse representation](#)

Definition

Sparse coding is the act of expressing a given input signal (e.g., image or image patch) as a linear superposition of a small set of basis signals chosen from a prespecified dictionary.

Background

At a high level, the problem of sparse coding is one of representing a given input signal as efficiently as possible:

Given an input signal $y \in \mathbb{R}^m$ (say an image or image patch) and a dictionary of basis signals $\mathbf{a}_1 \dots \mathbf{a}_n \in \mathbb{R}^m$, find a good approximation

$$y \approx x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \dots + x_n\mathbf{a}_n$$

in which most of the coefficients x_i are zero.

That is, we try to represent y as a linear combination of basis elements in which only a few of the coefficients are nonzero (i.e., the vector $\mathbf{x} = (x_1 \dots x_n)$ is *sparse*). This deceptively simple problem arises repeatedly in signal processing, modern statistics, and machine learning. The most readily apparent application is in data compression, where we can consider the sparse coefficients \mathbf{x} as a compressed representation of the signal y . However, numerous additional applications arise in signal and image acquisition, denoising, and inpainting. Sparse coding techniques have also received significant attention in the statistics literature, where sparsity is recognized as a means of regularizing high-dimensional inference – in particular, for regularizing linear regression when the number of predictors is larger than the number of observations.

These interactions make sparse coding a very vibrant area of research, with contributions from statistics, signal processing, optimization, applied mathematics, and cognitive neuroscience. Indeed, the term “sparse coding” originally comes from the neuroscience literature, where it has been observed that seeking a sparse codes for natural image patches yields Gabor-like basis functions that resemble the receptive fields in the human visual system [1]. In applied mathematics and statistics, a deep literature has developed around the question of when it is possible to solve sparse coding problems efficiently.

While the tools and problems encountered in sparse coding have precedents dating back almost a century, much of the development has been relatively recent. In computer vision, techniques from sparse coding (and related areas of sparse error correction and compressed sensing) have been employed for recognizing faces and objects, performing image upsampling, denoising, and superresolution. At the time of writing this article, sparse coding techniques are the subject of intense exploration in the vision community [2].

Theory

The model problem in sparse coding is one of searching for the *sparsest representation* of a given input signal y as a linear combination of dictionary elements:

$$\text{minimize } \|\mathbf{x}\|_0 \quad \text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{y}. \quad (1)$$

Above, the ℓ^0 -pseudonorm

$$\|\mathbf{x}\|_0 = \#\{i \mid x_i \neq 0\}$$

simply counts the number of elements in \mathbf{x} that are not zero. Although conceptually desirable, the problem (1) is computationally intractable (hard to approximate in the worst case), and so it is common practice to replace the ℓ^0 norm with a more tractable surrogate. One way to do this is to instead minimize the ℓ^1 -norm

$$\|\mathbf{x}\|_1 = \sum_i |x_i|.$$

This gives a convex optimization problem

$$\text{minimize } \|\mathbf{x}\|_1 \quad \text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{y}. \quad (2)$$

This relaxation is well motivated, in the sense that the ℓ^1 norm can be shown to be the tightest convex under-estimator of the ℓ^0 norm over the set of all vectors \mathbf{x} with $\max_i |x_i| \leq 1$. Moreover, whereas the non-convex cardinality minimization problem (1) does not admit an efficient algorithm, the convex problem (2) can be cast as a linear program and solved efficiently. For more details on available techniques for solving

the optimization problem (2), the interested reader can refer to the survey paper [3].

In certain situations, the link between (2) and (1) can be made quite a bit tighter. When the dictionary \mathbf{A} satisfies technical conditions that essentially assert that its columns are not too collinear, it can be shown that these two problems are formally equivalent: the tractable optimization (2) exactly recovers the sparse solution to (1)! For example, suppose that the columns \mathbf{a}_i of \mathbf{A} have unit ℓ^2 norm, and let

$$\mu(\mathbf{A}) \doteq \max_{i \neq j} |\langle \mathbf{a}_i, \mathbf{a}_j \rangle|. \quad (3)$$

Then, whenever $\mathbf{y} = \mathbf{A}\mathbf{x}_0$ for some \mathbf{x}_0 satisfying

$$\|\mathbf{x}_0\|_0 < \frac{1}{2}(1 + 1/\mu(\mathbf{A})), \quad (4)$$

we have that \mathbf{x}_0 is the unique optimal solution to the ℓ^1 minimization (2). That is to say, whenever there exists a sufficiently sparse solution to the system of equations $\mathbf{y} = \mathbf{A}\mathbf{x}$, this solution will be recovered by ℓ^1 norm minimization. Variants of this result have been obtained by a number of authors; the version described above is due to Donoho and Elad [4]. There is a vast literature on guarantees for ℓ^1 -minimization – in particular, a family of beautiful results on ℓ^1 -minimization with random matrices \mathbf{A} has inspired the recent development of *compressed sensing*, an approach to more efficiently acquire signals that are sparse in some known basis. For readers who are interested in learning more, one starting point is the survey paper [5].

In practice, the observation \mathbf{y} may contain noise, and so it is desirable to relax the constraint $\mathbf{A}\mathbf{x} = \mathbf{y}$. This leads to a new convex program

$$\text{minimize } \|\mathbf{x}\|_1 \quad \text{subject to} \quad \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2 \leq \varepsilon, \quad (5)$$

which is known in the signal processing literature as *basis pursuit denoising* [6]. Its Lagrangian reformulation,

$$\text{minimize } \|\mathbf{x}\|_1 + \lambda \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 / 2, \quad (6)$$

is known as the *Lasso* in statistics [7]. The optimization problems (5) and (6) are equivalent under an appropriate calibration $\varepsilon \leftrightarrow \lambda$, although no explicit expressions for the corresponding parameters are known.

The theoretical results alluded to above make ℓ^1 -minimization a very attractive approach to sparse

coding. However, it is by no means the only available algorithm. Researchers have explored a number of nonconvex objective functions which may more closely approximate the ℓ^0 norm in (1), at the expense of being hard to solve in general. Examples include the ℓ^p norms $\|\mathbf{x}\|_p = (\sum_i x_i^p)^{1/p}$ ($0 < p < 1$) and entropy-like functions such as $\sum_i \log(1 + |x_i|^2)$.

Greedy algorithms comprise another popular alternative to ℓ^1 -minimization. These algorithms construct a solution \mathbf{x} via an iterative procedure that repeatedly selects a “best” dictionary element \mathbf{a}_i to add to the representation. One prototypical example is the Orthogonal Matching Pursuit (OMP) algorithm [8] (which has been rediscovered in a number of different settings). OMP maintains an active set of indices J and a residual \mathbf{r} . Initially, J is empty, and $\mathbf{r} = \mathbf{y}$. At each step, the index $j \in \{1 \dots n\}$ that maximizes $|\langle \mathbf{a}_j, \mathbf{r} \rangle|$ is added to the active set J . The sparse coefficients \mathbf{x} are estimated via

$$\text{minimize } \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 \quad \text{subject to } \mathbf{x}(J^c) = \mathbf{0}. \quad (7)$$

The residual \mathbf{r} is updated as $\mathbf{r} = \mathbf{y} - \mathbf{A}\mathbf{x}$. This procedure is repeated until a sufficiently accurate approximation to \mathbf{y} is obtained. OMP is attractive for its simplicity and also comes with performance guarantees in some situations; the interested reader can refer to [5] and the references therein for a start.

All of the algorithms described above assume that the given observation \mathbf{y} indeed has a sparse approximation in terms of a known dictionary \mathbf{A} . In some situations, this can be guaranteed from prior understanding of the physical structure of the problem. For example, in face recognition, the dictionary \mathbf{A} can be constructed from images of training faces chosen in order to guarantee a good approximation of the test image \mathbf{y} [9]. In other situations, it may be possible to design optimal representations for classes of signals – as witnessed by the development of signal representations in signal processing and harmonic analysis over the past few decades.

However, if the signal model is not known ahead of time, or if the specific class of signals is believed to have some additional structure, an attractive alternative is to attempt to learn the dictionary \mathbf{A} itself from sample data. This leads to a problem known as *dictionary learning*, in which we observe multiple examples $\mathbf{Y} = \mathbf{y}_1 \dots \mathbf{y}_p \in \mathbb{R}^m$. The goal is to find a dictionary

$\mathbf{A} \in \mathbb{R}^{m \times n}$ of basis functions and sparse coefficient vectors $\mathbf{X} = \mathbf{x}_1 \dots \mathbf{x}_p \in \mathbb{R}^n$ such that $\mathbf{Y} \approx \mathbf{AX}$:

$$\text{minimize}_{\mathbf{A}, \mathbf{X}} \|\mathbf{X}\|_0 \quad \text{subject to } \|\mathbf{Y} - \mathbf{AX}\|_F \leq \varepsilon. \quad (8)$$

Notice that here the minimization is with respect to both \mathbf{A} and \mathbf{X} , and so even if we relax the ℓ^0 norm, the resulting optimization problem,

$$\text{minimize}_{\mathbf{A}, \mathbf{X}} \|\mathbf{X}\|_1 + \lambda \|\mathbf{Y} - \mathbf{AX}\|_F^2 / 2, \quad (9)$$

is not convex, and guaranteeing a global optimum is difficult. The key observation is that if either \mathbf{A} or \mathbf{X} is fixed, the optimization (9) becomes convex in the remaining variable. This naturally suggests an alternating directions approach:

$$\mathbf{X}_{k+1} = \arg \min_{\mathbf{X}} \|\mathbf{X}\|_1 + \lambda \|\mathbf{Y} - \mathbf{A}_k \mathbf{X}\|_F^2 / 2 \quad (10)$$

$$\mathbf{A}_{k+1} = \arg \min_{\mathbf{A}} \|\mathbf{X}_{k+1}\|_1 + \lambda \|\mathbf{Y} - \mathbf{AX}_{k+1}\|_F^2 / 2. \quad (11)$$

Each of these subproblems can be solved efficiently. It is not difficult to show that this procedure converges to some pair $(\mathbf{A}_*, \mathbf{X}_*)$. However, unlike the problem of sparse coding in a known dictionary \mathbf{A} (as discussed above), for dictionary learning there is currently no theory to explain when the algorithm will succeed. This is partially a consequence of the fact that the unknowns \mathbf{A} and \mathbf{X} enter into the equation $\mathbf{Y} = \mathbf{AX}$ in a bilinear fashion – the dictionary learning problem is difficult to analyze for the same reason that it is difficult to solve.

Nevertheless, empirical evidence suggests that there are situations in which this approach learns very effective data representations, and many researchers have used dictionary learning techniques to solve problems in image processing and vision. In fact, much of the initial excitement about sparsity in vision came from the classical paper of Olshausen and Field [1], which observed that the dictionary elements \mathbf{a}_i learned from natural images patches are similar to the receptive fields in the human visual system.

Many variants of the basic alternating directions approach have been investigated in the literature. For a more thorough history and additional references, we refer the interested reader to the survey paper [10].

Application

As alluded to above, at the time of this writing, numerous applications of sparse coding are being explored in the computer vision community. Due to their sheer number, most of these works lie beyond the scope of this article. In this section we briefly outline two examples of how the algorithms for sparse coding can be useful for solving problems in imaging and vision. The interested reader is invited to see [2] for a more thorough review.

Our first example comes from automatic face recognition [9]. In this application, the “dictionary elements” a_i are training images of subjects in the database. Several images of each subject are taken under varying illumination, stacked as vectors in \mathbb{R}^m (here $m = W \times H$ is the number of image pixels), concatenated together to form a large matrix $A \in \mathbb{R}^{m \times n}$. Given a new test image y taken under new illumination, and possibly subject to some additional corruption or occlusion, one can solve a sparse coding problem:

$$\text{minimize } \|x\|_1 + \|e\|_1 \quad \text{subject to } y = Ax + e. \quad (12)$$

Here, the sparse “error” term e allows some robustness to occlusion, while the sparse coefficients x naturally select images of the same subject to participate in the representation of y . This sparse code can be used for identifying the subject pictured in y or rejecting impostors not present in the database; see [9] for details.

Another representative example comes in image inpainting and superresolution [11, 12]. Suppose that patches y of a given input image are known to have good sparse approximations in some dictionary A , learned from a large collection of natural image patches. Suppose that some of the image pixels missing, so that rather than observing $y \in \mathbb{R}^m$, we observe only a subset $y(\Omega)$, where $\Omega \subset \{1 \dots m\}$. Then, one very natural approach to recovering the missing pixels is to solve a sparse coding problem (In [11], greedy algorithms are used, rather than ℓ^1 -minimization.)

$$\text{minimize } \|x\|_1 \quad \text{subject to } \|y(\Omega) - P_\Omega Ax\|_2^2 \leq \varepsilon^2, \quad (13)$$

where $P_\Omega \in \mathbb{R}^{|\Omega| \times m}$ is a projection matrix onto the coordinates indexed by Ω . Once the solution \hat{x} is recovered, one can estimate the missing elements via $\hat{y} = A\hat{x}$. For more examples of how sparse coding algorithms can be used in inpainting and related problems, see [11].

Open Problems

At the time of writing this article, there are numerous open problems in sparse coding, many of which are currently under vigorous attack. One question that has received significant recent attention is how to incorporate additional structure or prior knowledge into the algorithm to allow more accurate recovery of the sparse coefficients x . This leads to notions such as “group sparse coding,” in which certain subsets of coefficients are known to all be either active or inactive, simultaneously [13].

For some vision applications, explaining the good performance of sparse coding techniques and understanding their limitations remain open problems. At the same time, with so much recent development, we arguably have yet to fully realize the full power of sparsity for vision problems.

References

- Olshausen B, Field D (1997) Sparse coding with an overcomplete basis set: a strategy employed by V1? *Vis Res* 37(23):3311–3325
- Wright J, Yang A, Mairal J, Sapiro G, Huang T, Yan S (2010) Sparse representation for computer vision and pattern recognition. *Proc IEEE* 98(6):1031–1044
- Tropp J, Wright S (2010) Computational methods for sparse solution of linear inverse problems. *Proc IEEE* 98(6):948–958
- Donoho D, Elad M (2003) Optimally sparse representation in general (non-orthogonal) dictionaries via ℓ^1 -minimization. *Proc Natl Acad Sci* 100:2197–2202
- Bruckstein AM, Donoho DL, Elad M (2009) From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Rev* 51(1):34–81
- Chen S, Donoho D, Saunders M (2001) Atomic decomposition by basis pursuit. *SIAM Rev* 43(1):129–159
- Tibshirani R (1996) Regression shrinkage and selection via the Lasso. *J R Stat Soc B* 58(1):267–288
- Pati Y, Rezaifar R, Krishnaprasad P (1993) Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In: *Proceedings of*

- the Asilomar conference on signals, systems and computers, Pacific Grove, CA, vol 1, pp 40–44
9. Wright J, Yang A, Ganesh A, Sastry S, Ma Y (2009) Robust face recognition via sparse representation. *IEEE Trans Pattern Anal Mach Intell* 31(2):210–227
 10. Rubenstein R, Bruckstein A, Elad M (2010) Dictionaries for sparse representation modeling. *Proc IEEE* 98(6): 1045–1057
 11. Mairal J, Elad M, Sapiro G (2008) Sparse representations for color image restoration. *IEEE Trans Image Process* 17(1):53–69
 12. Yang J, Wright J, Huang T, Ma Y (2010) Image super-resolution via sparse representation. *IEEE Trans Image Process* 19(11):2861–2873
 13. Yuan M, Lin Y (2007) Model selection and estimation in regression with grouped variables. *J R Stat Soc B* 68(1): 49–67

Sparse Representation

► Sparse Coding

Spatiotemporal Reasoning

David Young

School of Informatics, University of Sussex, Falmer, Brighton, UK

Definition

Spatiotemporal reasoning is the use of time-varying information in image sequences, together with assumptions about the properties of objects and the environment, to make predictive inferences or to test hypotheses about the scene.

Background

Spatiotemporal reasoning plays an important rôle in theories of human and animal vision, especially those associated with the ecological psychology school of J.J. Gibson and his followers [6]. A central idea is that the optic flow field provides active organisms with the means to pick up the *affordances* of their environment (the likely possibilities for future interaction with the environment). A seminal example is the case of an object approaching to the viewer along the line

of sight and without rotation: in this case, the rate of expansion of the object's image specifies the inverse of the expected time to collision, assuming that the relative velocity remains constant.

The essential point of this example is that the time to collision may be estimated from the changing image without having any information about positions or velocities and without estimating these variables. A spatiotemporal relationship in the image (rate of expansion) is linked to a spatiotemporal property of the scene (time to collision) with no need for any geometrical reconstruction. This is particularly relevant to systems where rapid control of action in response to visual information is required, and the relationship has been proposed as part of the mechanism controlling interceptive actions in people and animals, as well as for the automatic control of vehicles [1, 7].

In computer vision, the expansion/time-to-collision relationship has been explored theoretically [8], while practical methods for measuring expansion rates have been investigated [2, 9, 11]. However, this particular relationship can best be seen as a special case of the general theory describing the optic flow induced by the relative motion of a camera and a surface, laid out by Koenderink and coworkers [4, 5]. This theory allows predictive inferences to be made without assuming line-of-sight approach or pure translation.

In parallel with such quantitative reasoning, qualitative spatiotemporal reasoning has also been investigated. Here, object attributes such as identity are assigned partly on the basis of expectations generated from the history of the image sequence. An example is the logging of vehicle movements at an airport, exploiting expectations about the spatiotemporal structure of event sequences [3, 10].

Although qualitative and quantitative spatiotemporal reasoning have very different flavors, both aim to extract practically useful information from image sequences by exploiting such properties of the scene as continuity of motion, the solidity and compactness of objects, and surface smoothness. The area is thus closely related to object tracking and robot navigation.

Theory

First-order optic flow provides an example of the use of spatiotemporal information. It is straightforward to

establish the relationship between the parameters of first-order flow models and information that may be applied to the control of action.

We assume that the optic flow in a region of an image may be approximated by a first-order model; thus,

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} + \begin{bmatrix} D + S_1 & S_2 - R \\ S_2 + R & D - S_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

where $[u_1 \ u_2]^T$ is the optic flow vector at image position $[x_1 \ x_2]^T$ and $[v_1 \ v_2]^T$ is the optic flow at the origin of image coordinates (the “zero-order” flow). D , R , S_1 , and S_2 are the parameters of the first-order component of the flow: respectively, dilation, rotation, and the two components of shear.

If we also assume that the flow is generated by a smooth surface patch in the scene, we can define some parameters which provide a partial description of the current state of relative motion between the patch and the camera. For example, the plane *immediacy* is the reciprocal of the time remaining until the camera passes through the plane tangent to the surface patch, assuming constant velocity. For a parachutist looking at a patch anywhere on a planar ground surface, it is the reciprocal of the expected time to landing. It is possible to show (using, e.g., Koenderink’s framework [4]), that plane immediacy is computable from the optic flow parameters plus the component of the camera’s angular rotation rate about the line of sight (also called the *spin*). The equation is

$$i_p = D - 3S \cos \phi$$

where $S = \sqrt{S_1^2 + S_2^2}$, $\sin \phi = (R - \omega)/S$, and ω is the spin. The relationship for direct approach, $i_p = D$, is a special case, applicable when the shear component of the flow vanishes.

Similarly, the immediacy of the plane passing through the surface patch, and normal to the line of sight, may be determined from

$$i_A = D - S \cos \phi$$

The essential quality of these quantities is that no information about the spatial layout of the scene is

required or inferred. Nonetheless, they can be applied to the practical control of movement, such as braking a vehicle to a halt at an obstacle. In general, optic flow measurements constrain the physical motion parameters to some subspace without determining those parameters exactly; this constraint may nonetheless be sufficient information to solve practical navigational or interceptive tasks.

References

1. Alenya G, Nègre A, Crowley JL (2009) Time to contact for obstacle avoidance. In: European conference on mobile robotics, Dubrovnik, Croatia, 09 2009
2. Blake A, Curwen R, Zisserman A (1993) A framework for spatiotemporal control in the tracking of visual contours. Int J Comput Vis 11(2):127–145
3. Buxton H, Walker N (1988) Query based visual analysis: spatio-temporal reasoning in computer vision. Image Vis Comput 6(4):247–254
4. Koenderink JJ (1986) Optic flow. Vis Res 26(1):161–179
5. Koenderink JJ, van Doorn AJ (1992) Second-order optic flow. J Opt Soc Am A 9(4):530–538
6. Lee DN (1980) The optic flow field: the foundation of vision. Philos Trans R Soc Lond Ser B 290(1038):169–179
7. Lee DN, Reddish PE (1981) Plummeting gannets: a paradigm of ecological optics. Nature 293:293–294
8. Maybank S (1987) Apparent area of a rigid moving body. Image Vis Comput 5(2):111–113
9. Seales W (1995) Measuring time-to-contact using active camera control. In: Hlaváč V, Šára R (eds) Computer analysis of images and patterns. Lecture Notes in Computer Science, vol 970. Springer, Berlin/Heidelberg, pp 944–949
10. Toal A, Buxton H (1992) Spatio-temporal reasoning within a traffic surveillance system. In: Sandini G (ed) Computer Vision (ECCV’92). Lecture notes in computer science, vol 588. Springer, Berlin/Heidelberg, pp 884–892
11. Tunley H, Young D (1994) First order optic flow from log-polar sampled images. In: ECCV ’94: Proceedings of the 3rd European conference on computer vision, Secaucus, NJ, USA, vol 1. Springer, New York, pp 132–137

SPDE

► [Stochastic Partial Differential Equations](#)

Specular Highlight

► [Specularity, Specular Reflectance](#)

Specularity, Specular Reflectance

Robby T. Tan

Department of Information and Computing Sciences,
Utrecht University, Utrecht, CH, The Netherlands

Synonyms

Interface reflection; MirrorLike reflection; Specular highlight

Related Concepts

- ▶ Dichromatic Reflection Model; ▶ Fresnel Equations;
- ▶ Surface Roughness

Definition

Specular reflection occurs when light is incident on a boundary interface between two different media and immediately reflects back to the medium where it comes from. Specular reflectance is the ratio of the reflected light by a boundary interface to the incident light. The visual appearance of specular reflections is known as specularity or specular highlight. To find the regions of surfaces that exhibit specular reflections is generally known as specularity detection.

Background

Reflection of light from an object is principally caused by the surface and body of the object. The former is known as specular or interface reflection and the latter is body or diffuse reflection. The specular reflection is the shiny mirrorlike reflection, which is commonly present in both man-made and natural objects. Mirrors, glass, ceramics, gold, silver, many fruits' skins, some leaves, etc. emit specular reflections. Theoretically, by considering the definition, almost all objects emit specular reflections, although the amount of the reflections varies depending on the object's optical properties, such as the surface roughness and the Fresnel reflection coefficient.

There are two main reasons why specular reflection is important in computer vision: (1) Many algorithms in computer vision assume perfect diffuse reflections and deem specular reflections to be outliers. However, in the real world, the presence of specular reflections is inevitable. Hence, incorporating the knowledge of specular reflections is essential to make the algorithms more robust. (2) Many computer vision algorithms may benefit from the information conveyed by specular reflection. This information includes the photometric and the geometric information, where the photometric information may be important for material recognition, and the geometric information may be useful for shape recognition.

Theory

Fresnel equations describe the reflection and transmission of light or electromagnetic waves at an interface between two media of differing refractive indices. According to the equations, when unpolarizing light impinges on a point of a flat and smooth surface, the reflection coefficient is $R = 0.5(R_{\perp} + R_{\parallel})$, where R_{\perp} and R_{\parallel} are the reflections of the electric field when it is perpendicular and parallel to the surface, respectively. The reflection coefficient is dependent on the angle of incoming light with respect to the surface normal and the refractive indices of the two media.

When a surface is perfectly flat and smooth (i.e., a perfect mirror), the direction of specular reflection will follow the law of reflection, which states the angle of incoming light θ_i and the angle of outgoing reflected light θ_r are the same ($\theta_i = \theta_r$). This implies that the specularly reflected light cannot be seen by an observer if the observer's position is not at the same direction as θ_r , which is true for the case of perfect mirrors. However, in many objects which are not perfect mirrors, a certain degree of specularity is still observable, even though the observer's position is slightly apart from the direction of θ_r . In other words, specular reflections do not only form a sharp line (spike) distribution of reflection but also form a lobe distribution. Therefore, there are two components of specular reflections: (1) specular spike and (2) specular lobe.

The Torrance-Sparrow reflection model [1] provides a good approximation of the specular lobe component, which is expressed as:



Specularity, Specular Reflectance, Fig. 1 (a) A dichromatic object exhibiting both diffuse and specular reflection. (b) The separated diffuse reflection component. (c) The separated specular reflection component. The results were computed using [5]

$$\rho = \frac{FG}{\cos \theta_r} \exp \left(-\frac{\alpha^2}{2\sigma^2} \right) \quad (1)$$

where F is the Fresnel reflection coefficient, G is the geometrical attenuation factor, θ_r is the angle between the viewing direction and the surface normal, α is the angle between the surface normal and the bisector of the viewing direction and the light source direction, and σ represents the surface roughness.

The Torrance-Sparrow reflection model uses geometric optics to describe the mechanism of specular reflection, which is only valid when the wavelength of light is much smaller than the roughness of the surface. According to [2], the model uses a slope distribution model to represent the profile of a surface. The surface is assumed to be a collection of planar microfacets, where their dimension is much larger than the wavelength of incident light. Each microfacet is perfectly smooth, and the orientation of each facet deviates from the mean orientation of the surface by an angle α . The model considers the masking and shadowing of microfacets by adjacent facets, where they can block light going into a facet or light reflected by it. The geometrical attenuation factor, G , is introduced to compensate the masking and shadowing effect. The surface roughness, σ , represents the spatial distribution of the lobe. The larger the value of σ , the larger the lobe distribution (implying less shiny surfaces), and vice versa. In this reflection model, the distribution of the specular reflections follows the Gaussian distribution, with mean α and standard deviation σ . Later, Cook and Torrance [3] replaced the Gaussian distribution with the Beckmann distribution function.

While the Torrance-Sparrow reflection model is able to approximately generate a mirrorlike distribution, namely, when σ is considerably small, its main drawback is that it cannot generate both the specular spike and specular lobe at the same time. To overcome this drawback, Nayar et al. [2] introduced a model unifying the Torrance-Sparrow's specular lobe and a spike specular model, and the latter is modeled as:

$$K_{ss} \delta(\theta_i - \theta_r) \delta(\phi_r) \quad (2)$$

where K_{ss} is the strength of the specular spike and (θ_r, ϕ_r) is the direction of the reflected. Nayar et al. base their analysis on the Beckmann-Spizzichino reflection model [4], which predicts the presence of both the specular lobe and spike. Unlike the Torrance-Sparrow reflection, the Beckmann-Spizzichino reflection model is based on physical optics analysis derived from the Maxwell's equations.

In comparison with diffuse reflections, in principle, specular reflections have three different properties [5]:

1. The diffuse and specular reflections have different degrees of polarization (DOP). The DOP represents the ratio of the light being polarized. For unpolarized incident light, the DOP of specular reflection is larger than that of diffuse reflection for most angles of incidence light, meaning that specular reflection is generally more polarized than diffuse reflection [6–8].
2. While recently a number of researchers (e.g., [9]) have introduced more complex models, the intensity distribution of diffuse reflections approximately follows Lambert's law [10]. In contrast, the intensity distribution of specular reflections generally follows the Torrance-Sparrow reflection model [1].

3. For optically inhomogeneous objects, the spectral power distribution (SPD) of specular reflection is determined by the object's interface spectral reflectance, which is mostly constant throughout the wavelength of visible spectrum, causing the SPD of specular reflections to be the same as the illumination's SPD [11]. In contrast, the SPD of diffuse reflection is determined by the object's body spectral reflectance. This spectral power distribution (color) independence of diffuse and specular reflections was described in the dichromatic reflection model proposed by Shafer [12].

Note that the condition that the SPD of specular reflections is the same as the illumination's SPD is called neutral interface reflection or NIR [11]. This mostly occurs for the surfaces of optically inhomogeneous objects (such as ceramics, plastics, paints); however, it does not always occur for the surfaces of optically homogeneous objects (such as gold, bronze, copper) [13].

Application

Many existing algorithms in computer vision assume perfect diffuse surfaces and deem specular reflections to be outliers. However, in the real world, the presence of specular reflections is inevitable since there are many objects that exhibit both diffuse and specular reflections. To properly acquire the diffuse-only reflections, a method to separate the two components robustly and accurately is required (e.g., [5], see Fig. 1 for an example of the separation). Once this separation has been accomplished, the specular reflection component can be advantageous since it conveys useful information of the object photometric properties (e.g., [6, 14]). Moreover, specular highlights are useful for estimating illumination colors or color constancy (e.g., [15]). Aside from the photometric properties, specular reflections can be employed to estimate the geometric properties (e.g., [16–18]).

Open Problems

Without knowing the polarizing states and only analyzing image intensities, specularity detection from a single image is still an open problem.

References

- Torrance K, Sparrow E (1966) Theory for off-specular reflection from roughened surfaces. *J Opt Soc Am* 57: 1105–1114
- Nayar S, Ikeuchi K, Kanade T (1991) Surface reflection: physical and geometrical perspectives. *IEEE Trans Pattern Anal Mach Intell* 13(7):611–634
- Cook R, Torrance K (1981) A reflectance model for computer graphics. *Comput Graph* 15:307–316
- Beckmann P, Spizzochino A (1963) The scattering of electromagnetic waves from rough surfaces. Pergamon, New York
- Tan RT, Ikeuchi K (2005) Separating reflection components of textured surfaces using a single image. *IEEE Trans Pattern Anal Mach Intell* 27(2):178–193
- Miyazaki D, Tan RT, Hara K, Ikeuchi K (2003) Polarization-based inverse rendering from a single view. In: 9th IEEE international conference on computer vision (ICCV 2003), Nice, France, pp 982–987
- Wolff LB (1990) Polarization-based material classification from specular reflection. *IEEE Trans Pattern Anal Mach Intell* 12(11):1059–1071
- Nayar SK, Fang XS, Boult T (1996) Separation of reflection components using color and polarization. *Int J Comput Vis* 21(3):163–186
- Wolff LB, Nayar S, Oren M (1998) Improved diffuse reflection models for computer vision. *Int J Comput Vis* 30(1):55–71
- Lambert JH (1760) Photometria sive de mensura de gratibus luminis., colorum et umbrae. sumptibus viduae E. Klett
- Lee H, Breneman E, Schulte C (1990) Modeling light reflection for computer color vision. *IEEE Trans Pattern Anal Mach Intell* 12(4):402–409
- Shafer S (1985) Using color to separate reflection components. *Color Res Appl* 10:210–218
- Healey G (1989) Using color for geometry-insensitive segmentation. *J Opt Soc Am A* 6(6):920–937
- Nishino K, Zhang Z, Ikeuchi K (2001) Determining reflectance parameters and illumination distribution from a sparse set of images for view-dependent image synthesis. In: 8th IEEE international conference on computer vision (ICCV 2001), Vancouver, Canada, pp 599–606
- Tan RT, Nishino K, Ikeuchi K (2004) Color constancy through inverse-intensity chromaticity space. *J Opt Soc Am A* 21:2004
- Blake A, Brelstaff G (1988) Geometry from specularities. In: 2nd IEEE international conference on computer vision (ICCV 1988), Tampa, USA, pp 394–403
- Oren M, Nayar SK (1997) A theory of specular surface geometry. *Int J Comput Vis* 24(2):105–124
- Adato Y, Vasilyev Y, Ben-Shahar O, Zickler T (2007) Toward a theory of shape from specular flow. In: 11th IEEE international conference on computer vision (ICCV 2007), Rio de Janeiro, Brazil, pp 1–8

Spherical Camera

► Omnidirectional Camera

Splines

Bo Zheng

Computer Vision Laboratory, Institute of Industrial Science, The University of Tokyo, Meguro-ku, Tokyo, Japan

Synonyms

Piecewise polynomial

Related Concepts

►Algebraic Curve; ►Parametric Curve

Definition

In mathematics, splines are piecewise continuous functions, such as polynomials, defined in successive subintervals. They are often used to represent one- or multidimensional data set (e.g., a curve or a surface) in the applications requiring interpolation, smoothing or nonrigid transformation [1]. For example, a spline curve is a piecewise collection of curve segments defined in polynomials that are connected end to end to form a single continuous curve. A curve in L -dimensional space can be simply defined by the following form:

$$S : [a, b] \rightarrow \mathbb{R}^L, \quad (1)$$

where function S takes variables from an interval $[a, b]$ and maps them to an L -dimensional real number. If the interval $[a, b]$ is divided into k ordered disjoint subintervals t_i, t_{i+1} with

$$a = t_0 \leq t_1 \leq \dots \leq t_k = b, i = 0, \dots, k - 1, \quad (2)$$

then in each subinterval $[t_i, t_{i+1}]$ there is a polynomial defined as

$$P_i : [t_i, t_{i+1}] \rightarrow \mathbb{R}^L. \quad (3)$$

Therefore,

$$S(t) = P_i(t), t_i \leq t < t_{i+1}, \quad (4)$$

where t_i is called a *knot* and vector $\mathbf{t} = (t_0, \dots, t_k)^T$ is called a *knot vector*.

Many types of splines have been developed through that $P(t)$ is defined in different types of functions. Some examples of representative splines are Bézier spline, B-spline, Nonuniform rational B-spline and Thin-plate spline which are briefly introduced in the following sections.

Bézier Spline

The polynomial for Bézier spline of degree n is

$$P(t) = \sum_{i=0}^n B_i^n(t) \mathbf{p}_i, \quad (5)$$

where \mathbf{p}_i ($\in \mathbb{R}^L$) are called the control points of a Bézier spline, and $B_i^n(t)$ are the basis functions determined by *Bernstein polynomials* of degree n as

$$B_i^n(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}. \quad (6)$$

Several Bézier spline curves can be concatenated by sharing the first and last control points. While a Bézier spline has C^1 continuity to its defined interval, at the shared control points it gets C^0 continuity. C^1 continuity can be achieved by sharing two control points at the end of two curves. An improvement that adds an adjustable weight to each basis can make it easier to control and more closely approximated to arbitrary shapes.

B-Spline

Similar to Bézier spline, the polynomial for B-spline is defined as

$$P(t) = \sum_{i=0}^n N_{i,k}(t) \mathbf{p}_i, \quad (7)$$

where \mathbf{p}_i ($\in \mathbb{R}^L$) are called the control points of a B-spline curve and the basis functions $N_{i,k}(t)$ of degree k can be derived by the recurrence equations as

$$N_{i,1}(t) = \begin{cases} 1 & (t_i \leq t < t_{i+1}) \\ 0 & (\text{otherwise}) \end{cases} \quad (8)$$

$$\begin{aligned} N_{i,k}(t) &= \frac{t - t_i}{t_{i+k-1} - t_i} N_{i,k-1}(t) \\ &\quad + \frac{t_{i+k} - t}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(t) \end{aligned} \quad (9)$$

The knots t_i for B-spline are generally with uniform, open uniform or nonuniform intervals.

Compared to Bézier splines, B-splines can have C^2 continuity at the joint control points. B-splines is more flexible and pleasing to work with, and thus it is popular in various graphics development environment.

Nonuniform Rational B-Splines (NURBS)

Since B-spline can be viewed as weighted sum of its control points and the weights $N_{i,k}$ usually have the property: $\sum_{i=0}^n N_{i,k}(t) = 1$. As weights $N_{i,k}(t)$ of B-spline depend only on the knot vector, it is useful to add to every control point one more weight w_i which can be set independently as

$$P(t) = \frac{\sum_{i=0}^n w_i N_{i,k}(t) \mathbf{p}_i}{\sum_{i=0}^n w_i N_{i,k}(t)}. \quad (10)$$

Then increasing a weight w_i makes the point more influential and attracts the curve to it. NURBS is often employed in computer-aided design systems.

Spline Surface

Not only for curve representation, spline is easy to be extended for representing a surface segment. Appropriately parameterized 2D variables are required for defining the 2D region subinterval to control a surface. For example, through a parameterized $u-v$ plane, Bézier spline can be used to represent a surface with the following form:

$$P(u, v) = \sum_{i=0}^m \sum_{j=0}^n B_i^m(u) B_j^n(v) \mathbf{p}_{ij}, \quad (11)$$

where \mathbf{p}_{ij} ($\in \mathbb{R}^3$) are the control points of Bézier spline surface.

Thin-Plate Spline (TPS)

Different to Bézier spline or B-spline in parametric form, Thin-plate spline (TPS) explicitly takes the point \mathbf{x} on a curve or a surface as the variable and maps it to a new value $f(\mathbf{x})$. A TPS function is often composed of a summation of radial basis functions and a low-order

polynomial, e.g., defined in the following form:

$$f(\mathbf{x}) = \sum_{i=1}^N w_i \phi(||\mathbf{x} - \mathbf{c}_i||) + \mathbf{v}^T \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix}, \quad (12)$$

where \mathbf{c}_i are the control points of TPS; w_i are the mapping coefficients through weighting the basis functions $\phi(\cdot)$; \mathbf{v} is the coefficient vector of the polynomial of degree 1; basis function $\phi(\cdot)$ is defined as

$$\phi(r) = \begin{cases} r^k & \text{with } k = 1, 3, 5, \dots, \\ r^k \ln(r) & \text{with } k = 2, 4, 6, \dots. \end{cases} \quad (13)$$

where k is equal to the dimension of dataset.

TPS has been widely applied for building a smooth nonrigid transformation model, by minimizing the integral bending energy [4]. It is also useful for data interpolation, since it can explicitly represent the arbitrarily spaced tabulated data set, e.g, $(x_i, y_i, f(x_i, y_i))$ in 2D case. The interpolation is smooth with derivatives of any order.

Background

The idea of presenting a curve using the connection with splines comes from the ship building industry, where they construct templates for ships by passing thin strips of wood (called “splines”). In some sub-fields of computer science, wide class of spline functions are applied to the applications where the discrete data requires interpolation and/or smoothing. Because splines are superior in terms of the following qualities: the simplicity of their construction, the ease and accuracy of control, their capacity to approximate complex shapes and the ability to design curves interactively.

Application and Theory

In addition to the usages of interpreting or smoothing for image representation, such as the work in [9], splines also play an important role in some specific computer vision applications. For example, in the Snake-based image segmentation designed by Kass et al. [5] and modified by Brigger et al. [3], splines are used to model the image contours by minimizing

the energy under the guidance of external and internal forces; For image motion estimation, Szeliski and Coughlan [8] proposed to represent the local motion flow field using multi-resolution splines; A classical linear method for estimating the TPS coefficients for image warps is proposed by Bookstein [2]; And Free Form Deformation (FFD) proposed by Sederberg [7] represents the nonrigid deformation of object using grid B-splines, which has a successful application in medical image registration [6].

References

1. Bartels RH, Beatty JC, Barsky BA (1987) An introduction to splines for use in computer graphics and geometric modeling. Morgan Kaufmann, San Francisco
2. Bookstein FL (1989) Principal warps: thin-plate splines and the decomposition of deformations. *IEEE Trans Pattern Anal Mach Intell* 11:567–585
3. Brigger P, Hoeg J, Unser M (2000) B-spline snakes: a flexible tool for parametric contour detection. *IEEE Trans Image Process* 9(9):1484–1496
4. Duchon J (1977) Splines minimizing rotation-invariant semi-norms in Sobolev spaces. *Constr Theory Funct Several Var Lect Notes Math* 571:85–100
5. Kass M, Witkin A, Terzopoulos D (1988) Snakes: active contour models. *Int J Comput Vis* 1:321–331
6. Rueckert D, Sonoda LI, Hayes C, Hill DLG, Leach MO, Hawkes DJ (1999) Nonrigid registration using free-form deformations: application to breast MR images. *IEEE Trans Med Imaging* 18:712–721
7. Sederberg T (1986) Free-form deformation of solid geometric models. *ACM SIGGRAPH Comput Graph* 20(4):151–160
8. Szeliski R, Coughlan J (1997) Spline-based image registration. *Int J Comput Vis* 22(3):199–218
9. Unser M, Aldroubi A, Eden M (1991) Fast b-spline transforms for continuous image representation and interpolation. *IEEE Trans Pattern Anal Mach Intell* 13(3):277–285

Standard Illuminants

Rajeev Ramanath¹ and Mark S. Drew²

¹DLP® Products, Texas Instruments Incorporated,
Plano, TX, USA

²School of Computing Science, Simon Fraser
University, Vancouver, BC, Canada

Synonyms

CIE standard illuminant

Related Concepts

► Chromaticity

Definition

A standard illuminant, as defined by the International Commission on Illumination (Commission Internationale de L'Eclairage, abbreviated as the CIE), denotes a specific relative distribution of energy (“spectral power distribution”) for an illuminant, over the range 300–830 nm [3].

Background

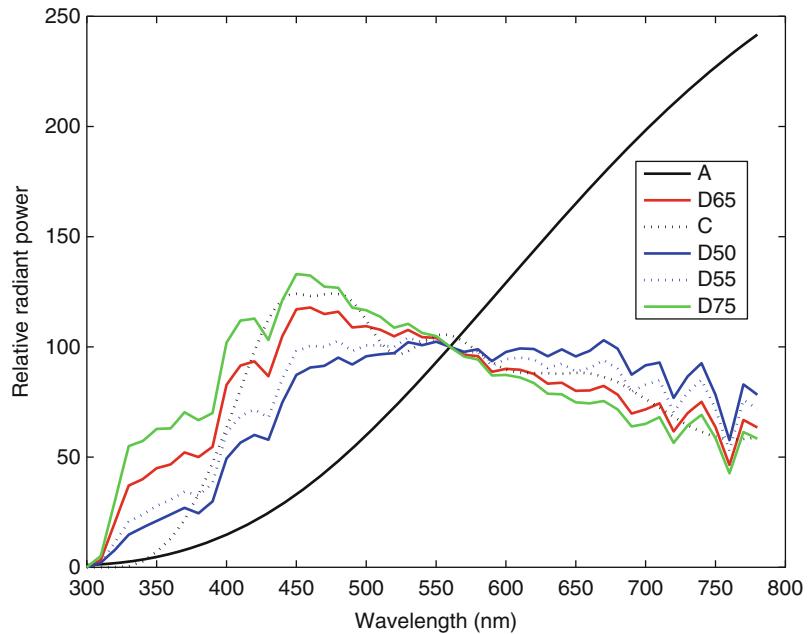
The motivation for a definition of standard illuminants arises from the need for accurate measurement of the colors of objects. It is well understood that for a Lambertian surface with a spectral reflectance function given by $r(\lambda)$, and under an illuminant given by $i(\lambda)$, the three-vector of CIE XYZ tristimulus values is given by [8]:

$$\begin{aligned} X &= k \int_{\lambda} \bar{x}(\lambda)i(\lambda)r(\lambda)d\lambda \\ Y &= k \int_{\lambda} \bar{y}(\lambda)i(\lambda)r(\lambda)d\lambda \\ Z &= k \int_{\lambda} \bar{z}(\lambda)i(\lambda)r(\lambda)d\lambda, \end{aligned} \quad (1)$$

where \bar{x} , \bar{y} , \bar{z} denote the three CIE XYZ color-matching functions (based on a standard observer) and k is a constant. Working in the XYZ color space – natively three-dimensional in nature – when a color changes it is difficult to attribute the change to a specific function of wavelength, because of the above 3-D projection: the observer color-matching functions $\{\bar{x}, \bar{y}, \bar{z}\}$, the illuminant $i(\lambda)$, or the object’s reflectance $r(\lambda)$. When the primary objective of a measurement is to quantify the object’s reflectance properties, it is therefore helpful to use a standard observer and a standard illuminant. The standardization of the observer is well documented in various CIE standards since the early 1900s, most recently in CIE 15-2004 [1].

Standard Illuminants, Fig. 1

Standard illuminants as specified by the CIE, normalized to a peak of 100 at 560 nm



Theory

In CIE-1998c (and in CIE 15-2004), the CIE defines the following standard illuminants, along with their relative spectral power distribution, in the spectral range 300–830 nm [1, 3]:

- Standard Illuminant A
- Standard Illuminant C
- Standard Illuminant D50
- Standard Illuminant D55
- Standard Illuminant D65
- Standard Illuminant D75

The spectral range 300–830 nm, wider than the range of visual perception, which is approximately 360–830 nm ([8], p.122), is used specifically to enable the evaluation of luminescent samples where the ultraviolet range becomes important. For colorimetric measurements, however, typically 380–780 nm is used.

Standard Illuminant A represents a black-body radiator with a temperature of approximately 2,856 K. Standard Illuminant C, although not used often, represents average daylight with correlated color temperature of about 6,800 K. Standard Illuminant D65 is

used to represent a phase of daylight with a correlated color temperature of approximately 6,500 K and is by far the most commonly used standard illuminant in colorimetry. Other D-illuminants are standard daylight illuminants annotated with the first two digits of their correlated color temperature; e.g., D55 denotes a standard daylight illuminant with a correlated color temperature of 5,500 K. [Figure 1](#) shows the relative spectral power distributions of the above standard illuminants.

It is to be noted that although the plots in [Fig. 1](#) show the spectra in the extended range from 300 to 780 nm (as is given in the CIE standard), data from 300–330 nm to 780–830 nm is extrapolated, but considered sufficiently accurate for colorimetric purposes. Further, the standard provides data to 5 nm increments, and should finer increments be needed the standard recommends performing linear interpolation on the spectra.

The correlated color temperature (T) of a daylight illuminant, for the purposes of interpolating standard daylights, is related to its chromaticity coordinates in the x, y chromaticity diagram by the following equations ([6], p. 111):

$$y_D = -3.0000x_D^2 + 2.870x_D - 0.275 \quad (2)$$

$$x_D = \begin{cases} \frac{-4.6070 \times 10^9}{T^3} + \frac{2.9678 \times 10^6}{T^2} + \frac{0.09911 \times 10^3}{T} + 0.244063, & 4,000 \text{ K} \leq T \leq 7,000 \text{ K} \\ \frac{-2.0064 \times 10^9}{T^3} + \frac{1.9018 \times 10^6}{T^2} + \frac{0.24748 \times 10^3}{T} + 0.237040, & 7,000 \text{ K} \leq T \leq 25,000 \text{ K}. \end{cases} \quad (3)$$

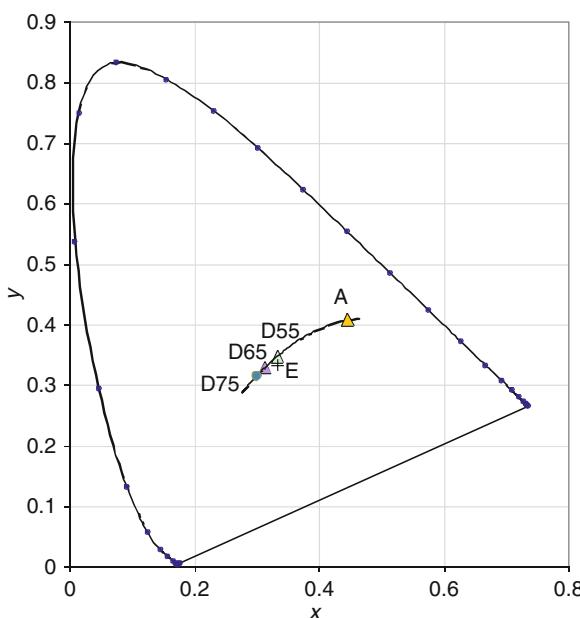
Note that several other suggestions for a best CCT have been posited [4, 7].

Figure 2 shows the locus of chromaticities of standardized daylight illuminants, specifically denoting the locations D55, D65, and also, for reference, the location of illuminant A and the equi-energy point, E ($x = y = 0.33$) (all for the CIE 1931 2° observer).

The CIE also specifies equations to compute the relative spectral power distribution of other D-illuminants as a function of their x,y chromaticity coordinates.

$$S(\lambda) = S_0(\lambda) + M_1 S_1(\lambda) + M_2 S_2(\lambda), \quad (4)$$

where $S_0(\lambda)$, $S_1(\lambda)$, and $S_2(\lambda)$ are specified in the CIE standard [1] and plotted in **Fig. 3**. These may be



Standard Illuminants, Fig. 2 CIE x, y chromaticity diagram showing illuminants D55, D65, D75, A, and E ('+', equi-energy illuminant)

considered as the mean and the following two eigenvectors of the space of daylight illuminants [5].

M_1 and M_2 are scale factors defined as a function of the chromaticity coordinate of illuminant, given by:

$$M_1 = \frac{-1.3515 - 1.7703x_D + 5.9114y_D}{0.0241 + 0.2562x_D - 0.7341y_D} \quad (5)$$

$$M_2 = \frac{0.03 - 31.4424x_D + 30.0717y_D}{0.0241 + 0.2562x_D - 0.7341y_D}. \quad (6)$$

These equations provides the relative spectral power distribution of the various D-illuminants, which are shown in **Fig. 4**. Customarily, $S(\lambda)$ is scaled such that its value at wavelength $\lambda = 560 \text{ nm}$ is 100.

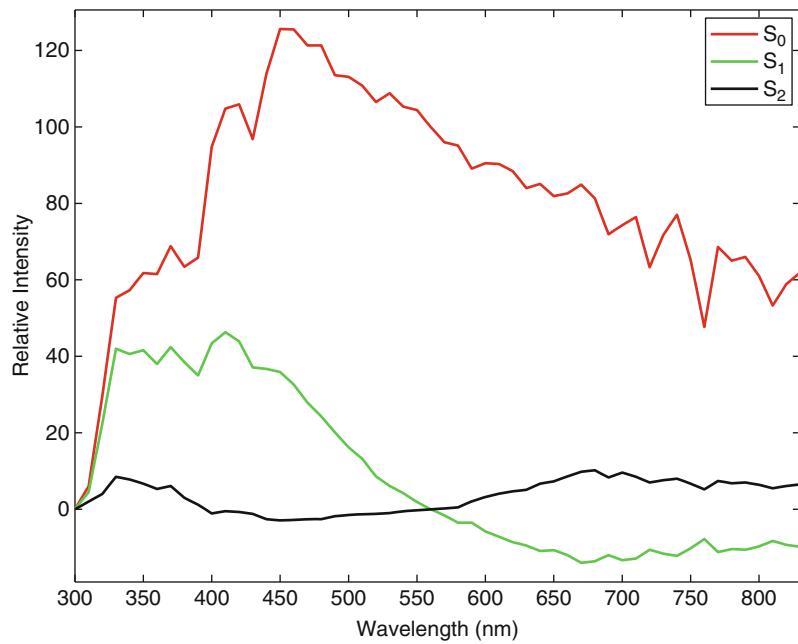
It is perhaps most important to note that standard illuminants are not the same as standard sources – which are the real-world equivalents of standard illuminants. In other words, a standard illuminant – defined by its relative spectral power – may not necessarily be realizable by a physical emitter of radiation (see **Fig. 5** for an example of a D65 source made by a fluorescent lamp and the D65 illuminant). The details of the challenges between the “theoretical” illuminant that is useful for computations and the “real” source may be found in a different CIE standard [2], and also in the book by Wyszecki and Stiles [8].

References

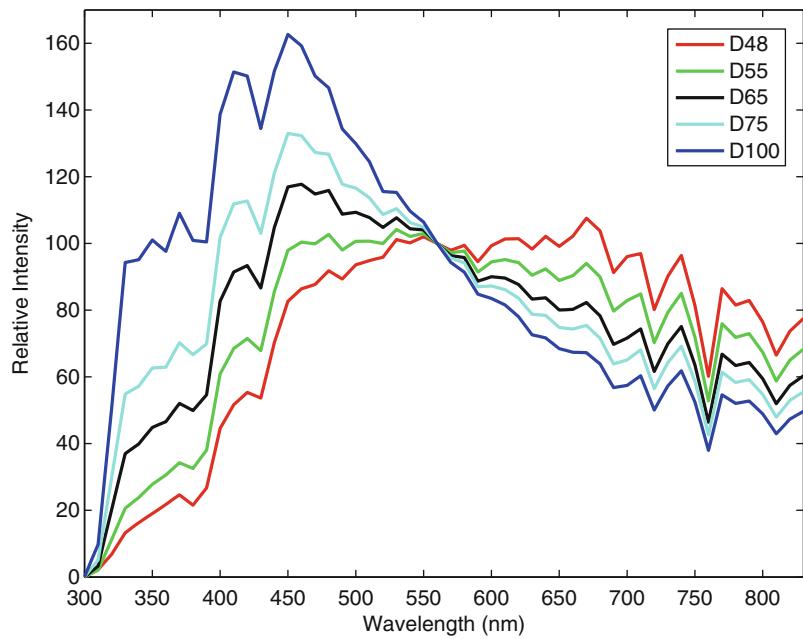
1. CIE 15:2004 (2004) Colorimetry. CIE, Vienna
2. CIE 51.2–1999 (1999) A method for assessing the quality of daylight simulators for colorimetry. CIE, Vienna
3. 1998c CIE (1998) CIE standard illuminants for colorimetry. CIE, Vienna. Also published as ISO 10526/CIE/S006/E1999
4. Hernández-Andrés J, Lee RL, Romero J (1999) Calculating correlated color temperatures across the entire gamut of daylight and skylight chromaticities. Appl Opt 38(27): 5703–5709

Standard Illuminants, Fig. 3

Component vectors (S_0 , S_1 , S_2 defined by the CIE), also considered as the mean and two additional eigenvectors of the space of daylight illuminants

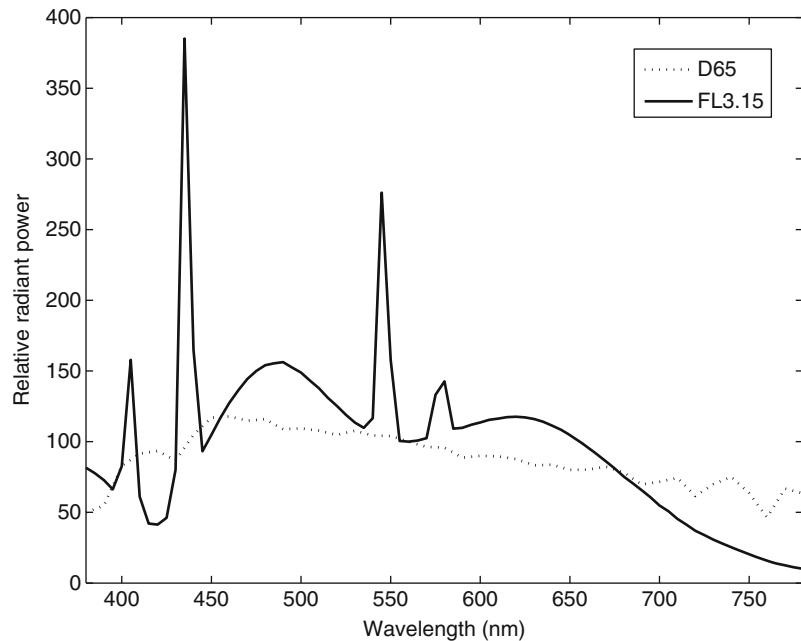
**Standard Illuminants, Fig. 4**

Daylight illuminants of different correlated color temperatures as computed from the component vectors (S_0 , S_1 , S_2 defined by the CIE)



Standard Illuminants, Fig. 5

Comparison of a standard illuminant (D65) and a standard source (FL 3.15) normalized to 100 at 560 nm



5. Judd DB, MacAdam DL, Wyszecki G, Budde HW, Condit HR, Henderson ST, Simonds JL (1964) Spectral distribution of typical daylight as a function of correlated color temperature. *J Opt Soc Am* 54:1031–1036
6. Judd DB, Wyszecki G (1975) Color in business, science and industry, 3rd edn. Wiley, New York
7. Ohno Y, Jergens M (1999) Results of the intercomparison of correlated color temperature calculation. Council for optical radiation measurements, June 16 1999. CORM subcommittee CR3 photometry. Boulder, Colorado
8. Wyszecki G, Stiles WS (1982) Color science: concepts and methods, quantitative data and formulas, 2nd edn. Wiley, New York

Statistical Cooling

► [Simulated Annealing](#)

Statistical Independence

Ying Nian Wu

Department of Statistics, UCLA, Los Angeles, CA, USA

Related concepts

► [Principal Component Analysis \(PCA\)](#)

Definition

Statistical independence is a concept in probability theory. Two events A and B are statistical independent if and only if their joint probability can be factorized into their marginal probabilities, i.e., $P(A \cap B) = P(A)P(B)$. If two events A and B are statistical independent, then the conditional probability equals the marginal probability: $P(A|B) = P(A)$ and $P(B|A) = P(B)$. The concept can be generalized to more than two events. The events A_1, \dots, A_n are independent if and only if $P(\bigcap_{i=1}^n A_i) = \prod_{i=1}^n P(A_i)$.

Theory

Two random variables X and Y are independent if and only if the events $\{X \leq x\}$ and $\{Y \leq y\}$ are independent for all x and y , that is, $F(x, y) = F_X(x)F_Y(y)$, where $F(x, y)$ is the joint cumulative distribution function and F_X and F_Y are the marginal cumulative distribution functions of X and Y , respectively. If X and Y are continuous random variables, then X and Y are independent if $f(x, y) = f_X(x)f_Y(y)$, where $f(x, y)$ is the joint probability density function and f_X and f_Y are the marginal probability density functions of X and Y , respectively. Similar results hold when both X

and Y are discrete, or one is discrete and the other is continuous.

If two random variables X and Y are independent, then their covariance $\text{Cov}(X, Y) = E(XY) - E(X)E(Y) = 0$, that is, they are uncorrelated. However, the reverse may not be true. Two uncorrelated random variables are not necessarily independent of each other. For example, if $X \sim \text{Uniform}[-1, 1]$ and $Y = X^2$, then $\text{Cov}(X, Y) = 0$, but clearly they are not independent.

The concept of independence can be generalized to more than two random variables. In probability theory, the law of large number, the central limit theorem, and concentration inequalities are usually obtained for independent random variables, although these results can be generalized to dependent random variables. In statistical learning theory, it is usually assumed that the training and testing examples are independent and identically distributed.

Application

Statistical independence is a key assumption in independent component analysis (ICA) [1], where the observed multivariate signal is assumed to be linear mixing of independent sources. A useful extension is independence is conditional independence. Two events A and B are independent given event C if $P(A \cap B|C) = P(A|C)P(B|C)$. If X , Y , and Z are discrete random variables, then X and Y are independent given Z if $P(X = x, Y = y|Z = z) = P(X = x|Z = z)P(Y = y|Z = z)$ for all x, y, z . If X , Y , and Z are continuous, then X and Y are independent given Z if $f_{X,Y|Z}(x, y|z) = f_{X|Z}(x|z)f_{Y|Z}(y|z)$, where $f_{X,Y|Z}$ is the conditional probability density function of (X, Y) given Z and $f_{X|Z}$ and $f_{Y|Z}$ are the conditional probability density functions of X given Z and Y given Z , respectively.

Conditional independence is a key assumption in Markov chains, Markov random fields, and more generally graphical models [2].

References

1. Hyvärinen A, Karhunen J, Oja E (2001) Independent component analysis. Wiley, New York
2. Cowell RG, Dawid AP, Lauritzen SL, Spiegelhalter DJ (1999) Probabilistic networks and expert systems. Springer, New York

Statistical Shape Analysis

Anuj Srivastava¹, Sebastian Kurtek²
and Eric Klassen²

¹Florida State University, Tallahassee, FL, USA

²Ohio State University, Columbus, OH, USA

Synonyms

Morphology, form analysis

Related Concepts

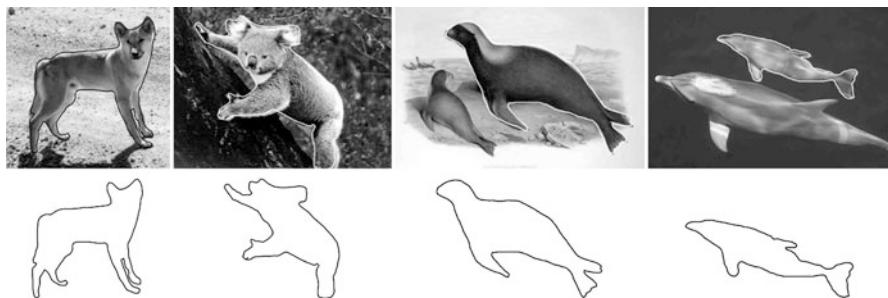
► [Activity Recognition](#); ► [Object Detection](#)

Definition

What is shape? Although the use of words *shape* or *shape analysis* is very common in computer vision, its definition is seldom made precise in a mathematical sense. According to the Oxford English Dictionary, it means “the external form or appearance of someone or something as produced by their outline.” Kendall [1] described shape as a mathematical property that remains unchanged under certain transformations such as rotation, translation, and global scaling. Shape analysis seeks to represent shapes as mathematical quantities, such as vectors or functions, that can be manipulated using appropriate rules and metrics. *Statistical* shape analysis is concerned with quantifying shape as a random quantity and developing tools for generating shape comparisons, averages, probability models, hypothesis tests, Bayesian estimates, and other statistical procedures on shape spaces.

Background

Shape is an important physical property of objects that characterizes their appearances and can play an important role in their detection, tracking, and recognition in images and videos. One usually restricts to the boundaries of objects, rather than the whole objects, for shape analysis and that leads to a shape analysis of curves (for 2D images) and surfaces (for 3D images).

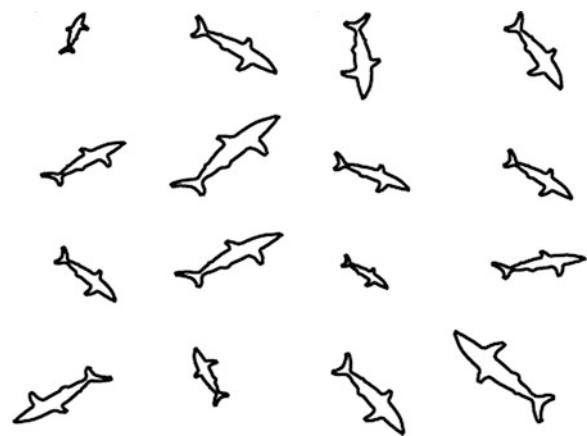


Statistical Shape Analysis, Fig. 1 Shapes of boundary curves are useful in object characterizations

Figure 1 suggests that shapes of boundaries can help characterize objects present in images. Any boundary contains only some partial information about the object since the color (texture) information inside and outside the boundary is lost. However, even with this limited information, it is often possible to broadly classify an object using shape analysis.

To understand the issues and challenges in shape analysis, one has to look at the imaging process since that is a major source of shape data. A picture can be taken from an arbitrary pose (arbitrary distance and orientation of the camera relative to the imaged object), and this introduces a random rotation, translation, and scaling of boundaries in the image plane. Therefore, any proper metric for shape analysis should be independent of the pose and scale of the boundaries. A visual inspection also confirms that any rotation, translation, or scaling of a boundary, while changing its coordinates, does not change its shape. **Figure 2** shows an example of 16 curves that differ in orientations, scales, and locations, but still represent the same shape.

In case of parameterized curves and surfaces, an additional challenge arises when it comes to invariance. Let $\beta : [0, 1] \rightarrow \mathbb{R}^2$ represent a parameterized curve, and let $\gamma : [0, 1] \rightarrow [0, 1]$ be a smooth, invertible function such that $\gamma(0) = 0$ and $\gamma(1) = 1$. Then, the composition $\tilde{\beta}(t) \equiv (\beta \circ \gamma)(t)$ represents a curve with coordinate functions that are different from those of $\beta(t)$ but have the same shape. $\tilde{\beta}$ is called a *re-parameterization* of β . **Figure 3** illustrates this issue with a simple example. It shows that the coordinate functions of the re-parameterized curve, $\tilde{\beta}_x(t)$ and $\tilde{\beta}_y(t)$, as functions of t , are different from the original coordinate functions $\beta_x(t)$ and $\beta_y(t)$. But when $\tilde{\beta}_x(t)$ is plotted versus $\tilde{\beta}_y(t)$, it traces out the same curve as that traced by $\beta_x(t)$ versus $\beta_y(t)$. This results in an

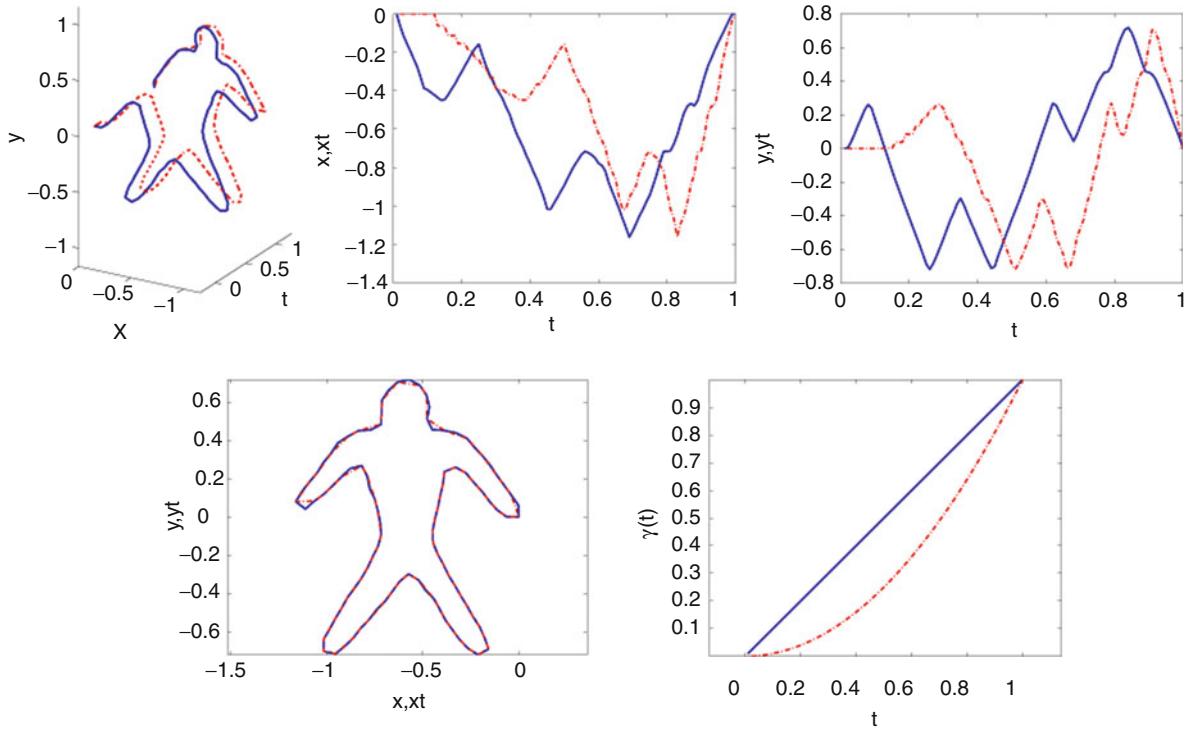


Statistical Shape Analysis, Fig. 2 Sixteen curves with different orientations, scale, and locations, but with identical shapes

additional invariance requirement in shape analysis of parameterized curves (and similarly for surfaces). That is, the shape metrics should be invariant to how the curves are parameterized.

In *statistical* shape analysis, one treats shapes as random quantities and tries to answer questions of the type:

- What is the difference between shapes of any two given objects? How can such shape differences be quantified while maintaining the desired invariances?
- What shape best represents the shapes of a given collection of objects? Another way to ask the same questions is: What is the statistical average of a given collection of shapes?
- What are the principal modes of variations in a given set of shapes?
- How can one capture the main shape variability in a population using probability models? Can random shapes be generated from such models?



Statistical Shape Analysis, Fig. 3 Re-parameterized curve has different coordinate functions but same shape as the original curve. *Left:* curves $(t, \beta_x(t), \beta_y(t))$ and $(t, \tilde{\beta}_x(t), \tilde{\beta}_y(t))$. Next

two panels: $\beta_x(t)$ and $\tilde{\beta}_x(t)$, and $\beta_y(t)$ and $\tilde{\beta}_y(t)$. *Right:* curves $(\beta_x(t), \beta_y(t))$ and $(\tilde{\beta}_x(t), \tilde{\beta}_y(t))$

- How can one use such probability models for shape classification and object recognition?

To start answering such questions, one needs precise mathematical representations of shapes and tools from algebra and geometry for analyzing them. In the literature there are numerous mathematical representations of objects that have been used for this purpose. These include representations using (unordered) point sets, (ordered) landmarks, level sets, deformable images, medial representations, and parameterized curves and surfaces. An important aspect, common to most representations, is the nonlinear geometry of a shape space. It is easy to recognize that shape is not a quantity that can be added, averaged, or grouped easily using Euclidean calculus. The desired invariance of shape to certain transformations (rigid motion, global scaling, and re-parameterization) implies that some nontraditional tools are needed.

Among the different methods used in shape analysis, the earlier methods typically represented objects by

finite sets of points, such as point sets or landmarks, but more recent methods are beginning to handle parameterized curves and surfaces directly as functions. Two of the earlier ideas based on point-set representations are summarized next.

Active Shape Models (ASM)

The active shape models approach to shape analysis was introduced by Cootes and Taylor in [2]. The simplest idea in shape analysis is to sample the boundaries at a number of points and form polygonal shapes by connecting those points with straight lines. Of course, the number and locations of these points on the objects can drastically change the resulting polygonal shapes, but this issue will be disregarded for the moment. One can organize the coordinates of these points in a form of a vector of coordinates and perform standard vector calculus. Let $x \in \mathbb{R}^{n \times 2}$ represent n ordered points selected from the boundary of an object. It is often convenient to identify points in \mathbb{R}^2 with elements of \mathbb{C} , i.e., $x^i \equiv z^i = (x^{i,1} + jx^{i,2})$, where $j = \sqrt{-1}$. Thus, in this

complex representation, a configuration of n points x is now $z \in \mathbb{C}^n$. Before analyzing the shape of z , it is “standardized” by moving its center to the origin (of the coordinate system):

$$z^i \mapsto \left(z^i - \frac{1}{n} \sum_{i=1}^n z^i \right).$$

To remove the scale variability, z is rescaled to have norm one, i.e., $z \mapsto z/\|z\|$. Then, one uses tools from standard multivariate statistics to analyze and model them. So far, the translation and the scale variability of a configuration are removed, but the rotation remains. That is, two configurations, z and a rotation of z , will have a nonzero distance between them even when they have the same shape. This problem is solved by using an additional step of rotational alignment when comparing shapes, as follows:

$$\begin{aligned} \phi^* &= \operatorname{argmin}_{\phi \in \mathbb{S}^1} \|z_1 - e^{j\phi} z_2\|^2 \\ &= \operatorname{argmin}_{\phi \in \mathbb{S}^1} (\|z_1\|^2 + \|e^{j\phi} z_2\|^2 - 2\Re(\langle z_1, e^{j\phi} z_2 \rangle)) \\ &= \operatorname{argmax}_{\phi \in \mathbb{S}^1} (\Re(e^{-j\phi} \langle z_1, z_2 \rangle)) = \theta, \\ \text{where } \langle z_1, z_2 \rangle &= r e^{j\theta}, \end{aligned} \quad (1)$$

and $\langle \cdot, \cdot \rangle$ is the standard Hermitian inner product. The distance between the two configurations is then $\|z_1 - e^{j\phi^*} z_2\| = \sqrt{2(1-r)}$. The corresponding optimal deformation from one shape to another is simply a straight line between z_1 and $e^{j\phi^*} z_2$, i.e., $\alpha_{asm}(\tau) = (1-\tau)z_1 + \tau e^{j\phi^*} z_2$ for $\tau \in [0, 1]$.

One remaining issue in this analysis is that on a closed curve which point should be selected as z^1 , the first point or the *seed*. If there are n points sampled on a curve, then there are n candidates for the seed. The solution is to select the best seed during a pairwise comparison of configurations. That is, select any point on the first configuration as the seed for the first shape and try all n points in the second configuration as candidate seeds for the second shape. Of those, select the one that results in the smallest distance from the first configuration. Figure 5 shows several examples of these deformations: one between a pair of human silhouettes, one between a pair of hands, and so on. These geodesics have been computed using $n = 200$ points on each configuration so that the resulting polygons look like smooth curves.

One can define the mean shape of several configurations z_1, z_2, \dots, z_k as the configuration that minimizes the sum of squares of distances:

$$\mu_{asm} = \operatorname{argmin}_{z \in \mathbb{C}^n} \sum_{i=1}^k \|z - e^{j\phi_i^*} z_i\|, \quad \phi_i^* = \cos^{-1}(\Re(\langle z, z_i \rangle)).$$

Figure 6 shows several illustrations of the mean computations.

Although this technique is relatively simple and fast, it has some important limitations. One limitation is that it does not preserve the scale constraints imposed on the shape representations. For instance, the intermediate shapes along the optimal deformations between any two unit length configurations do not have unit length. Similarly, the mean shape of $\{z_i\}$ is generally not of unit length. This is because no effort is made to restrict to the set of unit-length configurations, a limitation that is addressed in the next approach.

Kendall's Shape Analysis (KSA)

This approach, first laid out by Kendall [1] and advanced by several others [3], preserves desired constraints by restricting to appropriate manifolds. Once again a configuration of n points, taken from the boundary of an object, is treated as a complex vector. As earlier, the translations are removed by restricting to those elements of \mathbb{C}^n whose average is zero and the scale variability by rescaling the complex vector to have norm one. This results in a set:

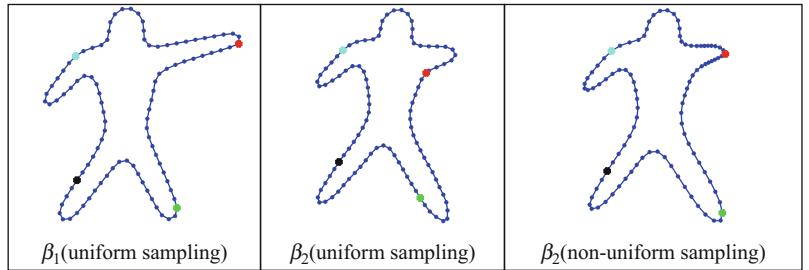
$$\mathcal{D} = \left\{ z \in \mathbb{C}^n \mid \frac{1}{n} \sum_{i=1}^n z^i = 0, \|z\| = 1 \right\}.$$

\mathcal{D} is not a vector space because $a_1 z_1 + a_2 z_2$ for $a_1, a_2 \in \mathbb{R}$ and $z_1, z_2 \in \mathcal{D}$ is typically not in \mathcal{D} , due to the unit norm constraint. However, \mathcal{D} is a unit sphere and one can utilize the geometry of a sphere to analyze points on it. Under the Euclidean metric, the shortest path between any two elements $z_1, z_2 \in \mathcal{D}$, also called a *geodesic*, is given by the great circle: $\alpha_{ksa} : [0, 1] \rightarrow \mathcal{D}$, where

$$\begin{aligned} \alpha_{ksa}(\tau) &= \frac{1}{\sin(\theta)} [\sin(\theta(1-\tau))z_1 + \sin(\tau\theta)z_2], \quad \text{and} \\ \theta &= \cos^{-1}(\Re(\langle z_1, z_2 \rangle)). \end{aligned} \quad (2)$$

Statistical Shape

Analysis, Fig. 4 Registration of points across two curves using the uniform and a convenient nonuniform sampling. Nonuniform sampling allows a better matching of features between β_1 and β_2



| | | |
|-----|--|--|
| ASM | | |
| KSA | | |
| ESA | | |
| ASM | | |
| KSA | | |
| ESA | | |
| ASM | | |
| KSA | | |
| ESA | | |

Statistical Shape Analysis, Fig. 5 Examples of geodesic paths between same shapes using ASM, KSA, and ESA

In order to compare the shapes represented by z_1 and z_2 , they need to be aligned rotationally, as was done earlier, but the shape space is defined more formally this time. Let $[z]$ be the set of all rotations of a configuration z according to:

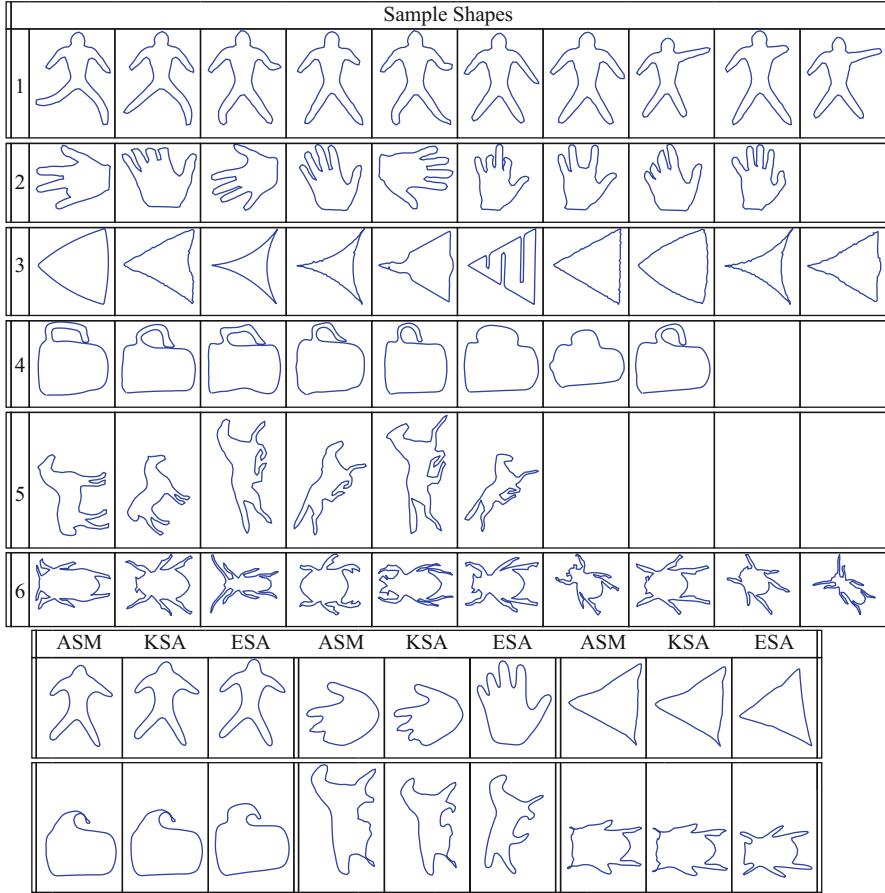
$$[z] = \{e^{j\phi} z | \phi \in \mathbb{S}^1\}.$$

One defines an equivalence relation on \mathcal{D} by setting all elements of this set as equivalent, i.e., $z_1 \sim z_2$ if there exists an angle ϕ such that $z_1 = e^{j\phi} z_2$. The set of all such equivalence classes is the quotient space $\mathcal{D}/U(1)$, where $U(1) = SO(2) = \mathbb{S}^1$ is the set of all rotations

in \mathbb{R}^2 . This space is called the *complex projective space* and is denoted by \mathbb{CP}^{n-1} . A geodesic between two elements $z_1, z_2 \in \mathbb{CP}^{n-1}$ is given by computing α_{ksa} between z_1 and $e^{j\phi^*} z_2$, where ϕ^* is the optimal rotational alignment of z_2 to z_1 . The length of the geodesic is given by θ and that quantifies the difference in shapes of the boundaries represented by z_1 and z_2 . Figure 5 shows several examples of geodesic paths between the same shapes as for the ASM examples.

Issue of Landmark Selection

Although Kendall's approach succeeds in preserving the unit-length constraints on the landmark



Statistical Shape Analysis, Fig. 6 Examples of mean shapes under three different methods

configurations, it does not address a very important practical issue: How to systematically select points on objects, say curves, to form representative point sets? This process is difficult to standardize and different selections can lead to drastically differing solutions. This issue is present in any point-based approach, including the ASM method discussed above. Some may suggest to sample a curve uniformly along its length, i.e., parameterize a curve β using arc length and sample $\{\beta(t_i) | i = 0, 1, 2, \dots, n\}$ where $t_i = i/n$. Although this provides a standardized way of sampling curves, the results are not always good since this forces a particular registration of points, i.e., the point $\beta_1(t_i)$ on the first curve is matched to the point $\beta_2(t_i)$ on the second curve, irrespective of the shapes involved. Figure 4 illustrates this point using an example. Shown in the left two panels are two curves: β_1 and β_2 , sampled uniformly along their lengths. For

$t_i = i/4$, $i = 1, 2, 3, 4$ the corresponding four points on each curve $\{\beta_1(t_i)\}$ and $\{\beta_2(t_i)\}$ are shown in the same color. While two of the four pairs seem to match well, the pairs shown in red and green fall on different parts of the body, resulting in a mismatch of features. This example shows the pitfall of using uniform sampling of curves. In fact, any predetermined sampling and preregistration of points will, in general, be problematic. A more natural solution is to treat the boundaries of objects as *continuous* curves, rather than discretize them into point sets at the outset, and find an optimal (perhaps nonuniform) sampling, such as the one shown in the rightmost panel, that better matches features across curves. This way one can develop a more comprehensive solution, including theory and algorithms, assuming continuous objects and will discretize them only at the implementation stage.

Theory

The *elastic shape analysis* (ESA) framework, for analyzing shapes represented by simple, closed, planar, curves is described here. (These ideas are also applicable, with some modifications, to curves in higher dimensions and to shape analysis of surfaces, but that is not discussed here.) The most natural way to study shapes of curves seems to be by treating them as parameterized curves. As mentioned earlier, an important aspect of this framework is that shape distances, geodesics, and statistics should be invariant to how the curves are parameterized.

Basic Challenge

To understand the basic challenge in analyzing shapes of parameterized curves, let $\|\cdot\|$ denote the \mathbb{L}^2 norm of a vector-valued function, i.e., $\|\beta\| = \int_{\mathbb{S}^1} \|\beta(t)\|^2 dt$, where the norm inside the integral is the vector 2-norm. Let $\beta_1, \beta_2 : \mathbb{S}^1 \rightarrow \mathbb{R}^2$ be two parameterized closed curves and γ be a re-parameterization function of the type used in Fig. 3. (It is more natural for the domain of parameterization to be \mathbb{S}^1 instead of $[0, 1]$ for closed curves.) The basic challenge in using the \mathbb{L}^2 norm $\|\beta_1 - \beta_2\|$ for comparing shapes of these two curves, even after a proper translation and scaling for standardization, is that $\|\beta_1 - \beta_2\| \neq \|\beta_1 \circ \gamma - \beta_2 \circ \gamma\|$ in general. That is, the distance between any two curves changes even if they are re-parameterized in the same way. This implies that the shape distance will depend on parameterizations, and this violates the requirement of invariance to parameterization. This problem necessitates a new representation and/or a new metric for analyzing shapes of curves.

Mathematical Representation

Let a parameterized closed curve be denoted as $\beta : \mathbb{S}^1 \rightarrow \mathbb{R}^2$. In order to analyze its shape, one represents β by its square-root velocity function (SRVF): $q(t) = \frac{\dot{\beta}(t)}{\sqrt{\|\dot{\beta}(t)\|}} \in \mathbb{R}^2$. The SRVF q includes both the instantaneous speed ($\|q(t)\|^2 = \|\dot{\beta}(t)\|$) and the direction ($\frac{q(t)}{\|q(t)\|} = \frac{\dot{\beta}(t)}{\|\dot{\beta}(t)\|}$) of the curve β at time t . The use of the time derivative makes the SRVF invariant to any translation of curve β . Conversely, one can reconstruct the curve β from q up to a translation. In order for the shape analysis to be invariant to scale, one rescales each curve to length one. With a slight abuse of notation, let us denote the rescaled curves by β . Since

$\int_{\mathbb{S}^1} \|\dot{\beta}(t)\| dt = 1$: $\int_{\mathbb{S}^1} \|q(t)\|^2 dt = \int_{\mathbb{S}^1} \|\dot{\beta}(t)\| dt = 1$. In other words, the \mathbb{L}^2 norm of the SRVF q is one. Additionally, if the curve β is closed, then its SRVF satisfies $\int_{\mathbb{S}^1} q(t) \|q(t)\| dt = 0$. Restricting to the curves of interest, represented by their SRVFs, the following set is obtained:

$$\mathcal{C} = \{q : \mathbb{S}^1 \rightarrow \mathbb{R}^2 \mid \int_{\mathbb{S}^1} q(t) \|q(t)\| dt = 0, \\ \int_{\mathbb{S}^1} \|q(t)\|^2 dt = 1\}.$$

\mathcal{C} is called the *preshape space* and is the set of SRVFs of all unit length, closed curves in \mathbb{R}^2 . Four shape-preserving transformations were mentioned: translation, scale, rotation, and re-parameterization. Of these, the first two have already been eliminated from the representations, but the other two remain. Curves that are within a rotation and/or a re-parameterization of each other result in different elements of \mathcal{C} despite having the same shape. The unification of such curves is performed algebraically as follows: Let $SO(2)$ be the group of 2×2 rotation matrices and Γ be the group of all re-parameterizations (they are actually positive diffeomorphisms of the unit circle \mathbb{S}^1). For a curve β , a rotation $O \in SO(2)$ and a re-parameterization $\gamma \in \Gamma$, the transformed curve is given by $O(\beta \circ \gamma)$. The SRVF of the transformed curve is given by $O(q \circ \gamma) \sqrt{\dot{\gamma}}$. In order to unify all elements in \mathcal{C} that denote the same shape, one can define equivalence classes of the type:

$$[q] = \{O(q \circ \gamma) \sqrt{\dot{\gamma}} \mid O \in SO(2), \gamma \in \Gamma\}.$$

Each such equivalence class $[q]$ is associated with a shape uniquely and vice versa. The set of all these equivalence classes is called the shape space \mathcal{S} ; mathematically, it is a quotient space of the preshape space: $\mathcal{S} = \mathcal{C}/(SO(2) \times \Gamma)$. The preshape space \mathcal{C} is a non-linear manifold because it is a subset of a unit sphere. One cannot perform calculus on this space as if it is a vector space. Operations such as addition, subtraction, and multiplication are not available on nonlinear spaces. This means that standard techniques in functional analysis for inferences on \mathcal{C} and \mathcal{S} cannot be used.

It should be noted that the mathematical representation used here, i.e., the SRVF, is not the only such representation. Younes et al. [4, 5] use a slightly different expression for an SRVF, based on an identification

of \mathbb{R}^2 with \mathbb{C} and taking complex square roots of the coordinates. The advantage of the SRVF form used in the current article is that it applies to curves in \mathbb{R}^m for any m , while the complex analysis in [4, 5] is applicable only to curves in \mathbb{R}^2 .

Elastic Riemannian Metric

Now that a mathematical representation of the shape of a curve, namely $[q]$, has been defined, how should one compare the shapes of two curves? In other words, for any two curves β_1 and β_2 , represented by their shape classes $[q_1]$ and $[q_2]$, respectively, what should be the shape metric $d_s([q_1], [q_2])$ that quantifies their shape differences? First, consider the role of a metric more closely. When one curve is deformed into another, a continuous sequence of curves, or a path in the curve space, is generated, and a natural question is how long that path is. The length of this path also quantifies the amount of deformation in going from one curve to the other. The question changes to the following: What should be the metric to measure this path length? A metric called the *elastic metric* will be used for this purpose. An elastic metric is a metric that measures the amount of bending and stretching between successive curves along the path and adds them up for the full path. Mio et al. [6] defined a family of elastic metrics depending upon how much relative weights are attached to bending and stretching. More recently, it was shown that (Joshi et al. [7], Srivastava et al. [8]) under the SRVF representation, the complicated elastic metric turns into (using a change of variables) the standard \mathbb{L}^2 metric. That is, one can alternatively compute the path lengths, or the sizes of deformations between curves, using the cumulative norms of the differences between successive curves along the paths in the SRVF space. This turns out to be much simpler and a very effective strategy for comparing shapes of curves, by finding the paths with least amounts of deformations between them, where the amount of deformation is measured by an elastic metric. Another distinct advantage of using SRVFs is that for any $q_1, q_2 \in \mathcal{C}$, $O \in SO(2)$ and $\gamma \in \Gamma$:

$$\|q_1 - q_2\| = \|O(q_1 \circ \gamma) \sqrt{\dot{\gamma}} - O(q_2 \circ \gamma) \sqrt{\dot{\gamma}}\|.$$

This means that the distance between any two curves remains the same if they are rotated and re-parameterized in the same way! This property, when

combined with an optimization step Eq. 3 (later), allows one to make shape metrics invariant to parameterizations.

Shape Comparison and Geodesics

Once a Riemannian metric on a manifold has been defined, one can compute distances between points in that manifold. For any two points, the distance between them is given by the length of the shortest path, called a *geodesic*, connecting them in that manifold. An interesting feature of this framework is that it not only provides a distance between shapes of two curves but also a geodesic path between them in \mathcal{S} . This path has the interpretation that it provides the optimal deformation of one shape into another. The geodesics are actually computed using the differential geometry of the underlying space \mathcal{S} . Consider two curves β_1 and β_2 , represented by their SRVFs q_1 and q_2 . Let $\alpha : [0, 1] \rightarrow \mathcal{C}$ be a differentiable path connecting them in \mathcal{C} . The length of this path is given by

$$L[\alpha] = \int_0^1 \langle \dot{\alpha}(\tau), \dot{\alpha}(\tau) \rangle^{1/2} d\tau,$$

where the inner product inside the integral is given by the elastic Riemannian metric. A geodesic is a path whose length cannot be minimized by locally perturbing it. It is often obtained by minimizing the cost function of the type:

$$\hat{\alpha} = \operatorname{argmin}_{\{\alpha : [0, 1] \rightarrow \mathcal{C} | \alpha(0) = q_1, \alpha(1) = q_2\}} \left(\int_0^1 \langle \dot{\alpha}(\tau), \dot{\alpha}(\tau) \rangle dt \right).$$

This cost function differs from the expression for $L[\alpha]$ in that the square root inside the integral is missing. It can be shown that a local minimizer of this cost function is also a local minimizer of $L[\alpha]$ and, hence, is a geodesic.

One technique for finding geodesics is called *path straightening*. It is an iterative technique that initializes an arbitrary path and then iteratively “straightens” it by updating it along the negative gradient of the cost function. Klassen and Srivastava [9] provide a nice analytical expression for the gradient of this cost function that results in a convenient gradient iteration. One applies these iterative updates, or straightening, until the gradient becomes negligible and the resulting path is the desired geodesic $\hat{\alpha}$. The length of this curve is denoted by $d_c(q_1, q_2) = L[\hat{\alpha}]$. This gives a geodesic

and a geodesic distance between SRVFs in \mathcal{C} , but the goal is to compute geodesic paths in \mathcal{S} . In other words, geodesic paths between the equivalence classes $[q_1]$ and $[q_2]$ are needed and not just q_1 and q_2 . It turns out that the desired geodesic is obtained by finding the shortest geodesic among all pairs $(\tilde{q}_1, \tilde{q}_2) \in ([q_1] \times [q_2])$. This search is further simplified by fixing an arbitrary element of $[q_1]$, say q_1 , and searching over all rotations and re-parameterizations of q_2 to minimize the geodesic length:

$$(O^*, \gamma^*) = \underset{O \in SO(2), \gamma \in \Gamma}{\operatorname{argmin}} d_c(q_1, O(q_2 \circ \gamma) \sqrt{\dot{\gamma}}). \quad (3)$$

The minimization over $SO(2)$ is similar to the ASM and KSA alignments earlier, but the optimization over Γ is new. This is accomplished using the dynamic programming algorithm or a gradient-type approach [8]. The resulting geodesic between q_1 and $O^*(q_2 \circ \gamma^*) \sqrt{\dot{\gamma}^*}$ is actually a geodesic between $[q_1]$ and $[q_2]$ in the shape space \mathcal{S} ; its length $d_s([q_1], [q_2]) = d_c(q_1, O^*(q_2 \circ \gamma^*) \sqrt{\dot{\gamma}^*})$ provides the geodesic distance in \mathcal{S} . Several examples of geodesic paths in the shape space \mathcal{S} are shown in Fig. 5; these geodesics are compared with the deformations/geodesics obtained by two previously described methods: ASM and KSA, for the same shapes. It is easy to see that the geodesics resulting from ESA provide more natural deformations as they are better in matching features across shapes.

The role of geodesics is preeminent in this framework because (1) its length d_s is a quantitative measure of difference between shapes of curves represented by q_1 and q_2 ; (2) this measure is invariant to rigid motion, scaling, and re-parameterization of any of the curves; (3) it also provides a full deformation (along the geodesic path) for taking one shape into another in an optimal way; and (4) the availability of geodesics leads to a further development of tools for building statistical summaries of shape classes, as described next.

Mean Shape

The first important task in statistical shape analysis is to define and compute the mean shape for a set of curves. Compared to the sample means of real-valued random variables, this task is not straightforward since the shape space \mathcal{S} is not a vector space. One cannot simply take the SRVFs of the given curves and average them point by point to get a mean shape. The notion

of a mean on a nonlinear manifold is typically established using the Karcher mean [10]. For a given set of curves $\beta_1, \beta_2, \dots, \beta_n$, represented by their SRVFs q_1, q_2, \dots, q_n , their Karcher mean is defined as the quantity that satisfies:

$$[\mu] = \underset{[q] \in \mathcal{S}}{\operatorname{argmin}} \sum_{i=1}^n d_s([q], [q_i])^2.$$

There is a gradient-based iterative algorithm for finding the minimizer of this cost function that can be found in [10–12]. Since this algorithm is based on a local search, the solution obtained is usually local and depends on the initial condition. Shown in Fig. 6 are some examples of mean shapes. The top six rows show a set of given curves and bottom rows display their means computed using the three methods discussed here: ASM, KSA, and ESA.

Shape Covariance and Principal Modes of Variation

Now that the first moment, i.e., the mean, of a set of curves has been defined, one can look for the higher moments. The role of the second centralized moment, the covariance, is especially important as (1) one can define a Gaussian distribution using just the mean and the covariance and (2) the singular value decomposition (SVD) of the covariance matrix can be used for a principal component analysis (PCA) of shape data. These two ideas are briefly summarized next, starting with the PCA.

For computing and analyzing the second and higher moments of a shape sample, the tangent space to the shape manifold \mathcal{S} at the point μ is used. This space, denoted by $T_\mu(\mathcal{S})$, is convenient because it is a vector space and one can apply more traditional methods here. First, for each given curve q_i , the vector $v_i \in T_\mu(\mathcal{S})$ is computed such that a geodesic that goes from μ to q_i in unit time has the initial velocity v_i . The function $v_i : \mathbb{S}^1 \rightarrow \mathbb{R}^2$ is also called the *shooting vector* from μ to q_i . Let \hat{K} be the sample covariance matrix of all the shooting vectors from μ to the SRVFs q_i s. For capturing the essential variability in a shape set, one can use principal component analysis (PCA) of the shooting vectors. The basic idea is to compute the SVD $\hat{K} = U \Sigma U^T$, where U is an orthogonal matrix and Σ is the diagonal matrix of singular values. Assuming that the entries along the diagonal

| | PC1 | PC2 |
|-----|-----|-----|
| ASM | | |
| KSA | | |
| ESA | | |
| ASM | | |
| KSA | | |
| ESA | | |
| ASM | | |
| KSA | | |
| ESA | | |

Statistical Shape Analysis, Fig. 7 Two principal directions of variability in shapes given in Examples 1–3 in Fig. 6

| Method | “Gaussian” Random Samples | | |
|--------|---------------------------|---------|---------|
| | Shape 1 | Shape 2 | Shape 3 |
| ASM | | | |
| KSA | | | |
| ESA | | | |

Statistical Shape Analysis, Fig. 8 Random samples from the “Gaussian”-type distributions under the three methods: ASM, KSA, and ESA, for parameters estimated from the given shapes shown in Examples 1–3 of Fig. 6

in Σ are organized in a nonincreasing order, U_1 , U_2 , etc. represent the dominant directions of variability in the data. If a singular vector U_j is used to form vectors $t\sqrt{\Sigma_{jj}}U_j$, then they represent shooting variability in the direction U_j . One can map these directions back on the shape space \mathcal{S} using an exponential map at μ . The details of this exponential map are omitted here, but it is basically the point reached on \mathcal{S} by constructing a geodesic in the shooting direction so that the length of the geodesic is the same as the length of the shooting vector. The resulting geodesics are also called the *principal geodesic paths*. Figure 7 shows the

principal geodesic paths along U_1 and U_2 , respectively, for $t = -1.5$ to $t = 1.5$. Of course, the middle points in each row are the mean shape μ .

Probabilistic Shape Models

One important use of means and covariances of shape families is in devising “Gaussian”-type probability densities on the shape space \mathcal{S} . In case of ASM this idea is straightforward since the shape representations are simple vectors, and one can define multivariate normal densities on these vector spaces. However, for KSA and ESA the shape spaces are nonlinear

manifolds and such probability densities are not easy to define. One common idea to tackle the nonlinearity of the shape space is to impose a Gaussian distribution on the tangent space $T_\mu(\mathcal{S})$ since that is a vector space. In case of ESA this space is infinite dimensional, so the Gaussian model is actually imposed on a finite-dimensional subspace, e.g., a principal subspace, of $T_\mu(\mathcal{S})$. Let $\{U_j\}$, $j = 1, 2, \dots, k$ denote the singular vectors of the sample covariance matrix as earlier. Then, one can define a random vector $v \equiv \sum_{j=1}^k f_j U_j$ where $f_j \sim N(0, \Sigma_{jj})$ and define $q = \exp_\mu(v)$, the exponential map of v from $T_\mu(\mathcal{S})$ to the shape space \mathcal{S} . The procedure defines a generative random model on the shape space and is easy to sample from. Shown in Fig. 8 are examples of random samples from \mathcal{S} using means and covariances estimated from the given data in Examples 1–3 from Fig. 6. For comparison, this figure also shows random samples from similar Gaussian models but using ASM and KSA. It is easy to observe the superiority of the results obtained using ESA; the modeling results from ASM are typically the worst of the three methods.

Open Problems

Although there has been a significant progress in shape analysis of curves, especially the planar curves, several important problems remain open. Firstly, the choice of Gaussian-type models for capturing shape variability of curves is more for convenience than data driven. It is important to explore and develop statistical models on shape manifolds that are both efficient, e.g., parametric models are more efficient, and provide a better representation of the observed variability. The second family of open problems relates to the shape analysis of surfaces. One needs to develop special mathematical representations of parameterized surfaces that will enable their shape analysis in a manner that is invariant to their parameterizations. Some preliminary ideas in that direction have been proposed in [13].

References

- Kendall DG (1984) Shape manifolds, procrustean metrics and complex projective spaces. *Bull Lond Math Soc* 16: 81–121
- Cootes TF, Taylor CJ, Cooper DH, Graham J (1995) Active shape models: their training and application. *Comput Vis Image Underst* 61:38–59

- Dryden IL, Mardia K (1998) Statistical shape analysis. Wiley, Chichester/New York
- Younes L (1998) Computable elastic distance between shapes. *SIAM J Appl Math* 58(2):565–586
- Younes L, Michor PW, Shah J, Mumford D, Lincei R (2008) A metric on shape space with explicit geodesics. *Mat E Appl* 19(1):25–57
- Mio W, Srivastava A, Joshi SH (2007) On shape of plane elastic curves. *Intl J Comput Vis* 73(3):307–324
- Joshi SH, Klassen E, Srivastava A, Jermyn IH (2007) A novel representation for riemannian analysis of elastic curves in \mathbb{R}^n . In: Proceedings of IEEE conference on computer vision pattern recognition (CVPR), Minneapolis, pp 1–7
- Srivastava A, Klassen E, Joshi SH, Jermyn IH (2010) Shape analysis of elastic curves in euclidean spaces. *IEEE Trans Pattern Anal Mach Intell* (Accepted for publication). doi:10.1109/TPAMI.2010.184
- Klassen E, Srivastava A (2006) Geodesics between 3D closed curves using path-straightening. In: Proceedings of European conference on computer vision (ECCV) Graz. Lecture notes in computer science I, pp 95–106
- Karcher H (1977) Riemannian center of mass and mollifier smoothing. *Commun Pure Appl Math* 30(5):509–541
- Le HL, Kendall DG (1993) The riemannian structure of euclidean shape spaces: a novel environment for statistics. *Ann Stat* 21(3):1225–1271
- Srivastava A, Joshi SH, Mio W, Liu X (2005) Statistical shape analysis: clustering, learning and testing. *IEEE Trans Pattern Anal Mach Intell* 27(4):590–602
- Kurtek S, Klassen E, Ding Z, Jacobson S, Jacobson J, Avision M, and Srivastava A (2011) Parameterization-invariant shape comparisons of anatomical surfaces. *IEEE Trans Med Imaging* 30(3):849–858

Stochastic Differential Equations in Infinite Dimensions

► [Stochastic Partial Differential Equations](#)

Stochastic Partial Differential Equations

Annika Lang

Seminar für Angewandte Mathematik, ETH Zürich,
Zürich, Switzerland

Synonyms

[SPDE](#); Stochastic differential equations in infinite dimensions

Definition

A stochastic partial differential equation (SPDE) is a partial differential equation (PDE) with an extra stochastic term, e.g., an Itô integral. Sometimes partial differential equations, where the differential operator or the initial condition is disturbed, are also called SPDEs, but the more common term for these equations is random PDEs.

Background

First results on SPDEs and infinite-dimensional stochastic differential equations (SDEs) appeared in the mid-1960s. Ample publications and results are due to the end 1970s and early 1980s. Here the work by Walsh [20] and Pardoux [16] should be mentioned. In the early 1990s, Da Prato and Zabczyk published their book with an infinite-dimensional approach to SPDEs driven by Wiener processes [6]. In the last years, a number of books on SPDEs were published, in particular an extension of [6] by Peszat and Zabczyk [17]. Other recent publications to be mentioned are Chow [3], Prévôt and Röckner [18], and Holden et al. [8].

The motivation to study SPDEs was driven on the one hand from the internal development of analysis and theory of stochastic processes and on the other side from applications. Random phenomena studied in natural sciences needed to be described. Especially applications in physics, chemistry, biology, control theory, nonlinear filtering, engineering, and finance pushed the development of the theory of SPDEs and are still pushing it. In recent years, the applications also inspire the design of numerical methods to “create numbers,” i.e., to simulate the equations.

Theory

Let $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t \geq 0}, \mathbb{P})$ be a filtered probability space. The filtration is supposed to be right continuous, and \mathcal{F}_0 contains all \mathbb{P} -zero sets. A *stochastic differential equation* is given by

$$dX(t) = a(t, X(t)) dt + b(t, X(t)) dM(t) \quad (1)$$

with initial condition $X(0) = X_0$ and $X : \mathbb{R}_+ \times \Omega \rightarrow \mathbb{R}^d$. The initial condition might be a random

variable. This notation is the abbreviation for the integral equation

$$\begin{aligned} X(t) = X_0 &+ \int_0^t a(s, X(s)) ds \\ &+ \int_0^t b(s, X(s)) dM(s). \end{aligned}$$

In this notation, M is a stochastic process adapted to (\mathcal{F}_t) which is, e.g., a local martingale. In many applications, the stochastic process is a Brownian motion also called Wiener process and abbreviated by B or W . Especially in recent years, other typical stochastic processes that are considered are Lévy processes. The expression

$$\int_0^t b(s, X(s)) dM(s)$$

is a stochastic integral of Itô type (cf. [5, 9, 14, 15, 19]). In order to ensure the existence of a solution to Eq. (1), the functions $a : \mathbb{R}_+ \times \mathbb{R}^d \rightarrow \mathbb{R}^d$, $b : \mathbb{R}_+ \times \mathbb{R}^d \rightarrow \mathbb{R}^d$, and the stochastic process M have to satisfy certain conditions. One possibility is that a and b are of linear growth and Lipschitz type.

One approach to derive an SPDE is to extend Eq. (1) to maps $X : \mathbb{R}_+ \times \Omega \rightarrow \mathbb{R}^d$ where d tends to infinity. Another possibility to derive SPDEs is to start with a PDE. Therefore, let A be a differential operator on a domain $D \subset \mathbb{R}^d$. Then,

$$\frac{\partial}{\partial t} u(t, x) = Au(t, x) + f(u(t, x)) + g(u(t, x))\dot{\eta}(t, x)$$

with initial condition $u(0, x) = u_0(x)$ for $x \in D$ denotes an SPDE where $\dot{\eta}$ is white noise. As in general stochastic processes are almost surely nowhere differentiable with respect to time t , this notation is used seldom. Instead of that, SPDEs are integral equations, i.e.,

$$\begin{aligned} u(t, x) = u_0(x) &+ \int_0^t (Au(s, x) + f(u(s))) ds \\ &+ \int_0^t g(u(s, x)) dM(s, x). \end{aligned}$$

One approach to solve equations of that type was introduced by Da Prato and Zabczyk in 1992 [6]. There, a more general framework is used. Let H and U be separable Hilbert spaces, e.g., $L^2(D), \mathbb{R}^d$, or Sobolev

spaces $H^\alpha(D)$, then the previous equation can be rewritten as

$$\begin{aligned} u(t) &= u_0 + \int_0^t (Au(s) + F(u(s))) ds \\ &\quad + \int_0^t G(u(s)) dM(s), \end{aligned}$$

where $F : D(F) \subset H \times \Omega \rightarrow H$, $G : D(G) \subset H \rightarrow L(U, H)$, and M is a U -valued stochastic process, e.g., a square integrable martingale like a Wiener or Lévy process. Here $D(F)$ denotes the domain of F , and $L(U, H)$ is the space of linear operators from U into H . The stochastic process u is a mapping from $\mathbb{R}_+ \times \Omega$ into the Hilbert space H . The abbreviated form of the previous equation is

$$du(t) = (Au(t) + F(u(t))) dt + G(u(t)) dM(t) \quad (2)$$

with $u(0) = u_0$. So instead of solving an SPDE, here Hilbert space-valued SDEs are solved. Existence and uniqueness of solutions might be shown, using, e.g., semigroup theory and classical SDE theory. In the theory of SPDEs, there exist three main concepts of solutions which are similar to those known from PDE theory:

1. *Strong solutions* [6]: A predictable H -valued stochastic process $u = (u(t), t \in \mathbb{R}_+)$ is a *strong solution* to Eq. (2), if:

- For all $t \geq 0$, $u(t)$ takes values in $D(A) \cap D(F) \cap D(G)$ \mathbb{P} -a.s.
- For all $t \geq 0$, it holds \mathbb{P} -a.s.

$$\begin{aligned} u(t) &= u_0 + \int_0^t (Au(s) + F(u(s))) ds \\ &\quad + \int_0^t G(u(s)) dM(s). \end{aligned}$$

2. *Weak solutions* [17]: A predictable H -valued process u is a *weak solution* to Eq. (2), if:

- $\sup_{t \in [0, T]} \mathbb{E}(\|u(t)\|_H^2) < +\infty$ for all $T \in \mathbb{R}_+$
- For all $a \in D(A^*)$, $t \geq 0$, it holds \mathbb{P} -a.s.

$$\begin{aligned} \langle a, u(t) \rangle_H &= \langle a, u_0 \rangle_H + \int_0^t (\langle A^* a, u(s) \rangle_H \\ &\quad + \langle a, F(u(s)) \rangle_H) ds \\ &\quad + \int_0^t \langle G^*(u(s)) a, dM(s) \rangle_{\mathcal{H}}. \end{aligned}$$

Here A^* denotes the adjoint operator and \mathcal{H} is the reproducing kernel Hilbert space generated by M .

3. *Mild solutions* [17]: Let A be the generator of a strongly continuous semigroup $(S(t), t \geq 0)$, then the stochastic process u is a *mild solution* to Eq. (2), if:

- $\sup_{t \in [0, T]} \mathbb{E}(\|u(t)\|_H^2) < +\infty$ for all $T \in \mathbb{R}_+$
- For all $t \geq 0$, it holds \mathbb{P} -a.s.

$$\begin{aligned} u(t) &= S(t)u_0 + \int_0^t S(t-s)F(u(s)) ds \\ &\quad + \int_0^t S(t-s)G(u(s)) dM(s). \end{aligned}$$

Furthermore, there are viscosity solutions [13]. One could also think of solutions in probability or in expectation and many other concepts.

If A is the generator of a strongly continuous semigroup, similarly to SDE theory, solutions exist under linear growth and Lipschitz conditions, but many extensions are possible. The reader is referred to the literature for explicit conditions for existence and uniqueness results. In this context, an overview can be found in [17], but the subject is still evolving such that recent research papers will give more general results.

For applications, a suitable type of solution has to be chosen. This leads to the next section where some possible applications are discussed as well as the problem that solutions for most SPDEs are not known. Therefore, numerics of SPDEs have become more and more important within the last year.

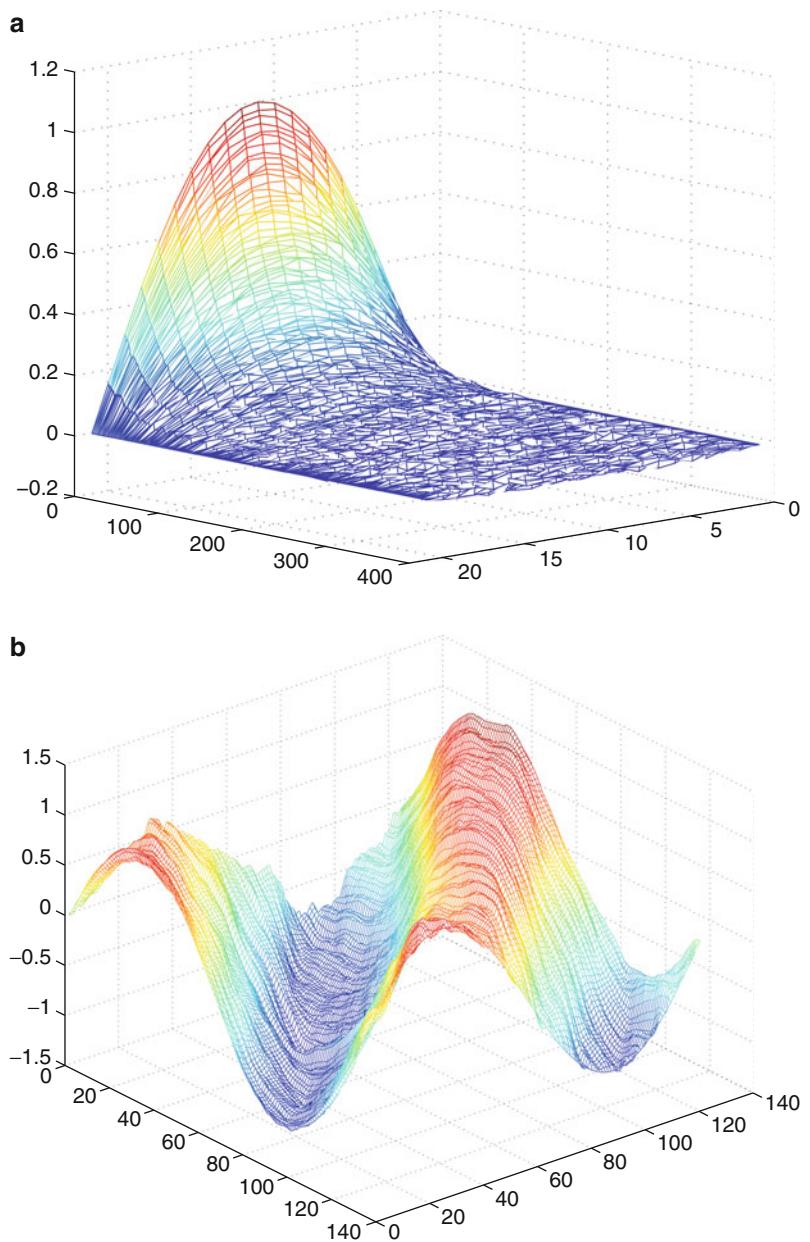
Application

SPDEs are relevant in many different applications. In engineering such as image analysis, surface analysis, and filtering, they result from addition of noise to PDEs. In finance, systems of SDEs are common that extend to infinite-dimensional problems and therefore to SPDEs. Furthermore, first applications to life and bio-sciences are done.

All these applications are interested in solutions of SPDEs, but these are not known explicitly in most cases. Therefore, numerics of SPDEs have become

Stochastic Partial Differential Equations, Fig. 1

Simulation of one path of an SPDE with additive noise on an interval. (a) Parabolic equation. (b) Hyperbolic equation



important within the last years. Methods combine SDE methods with PDE methods. Simulation methods for SDEs are especially Euler–Maruyama, Milstein, and higher-order schemes, where a good survey is given in [11], as well as Monte Carlo methods [7]. From PDE theory Galerkin methods, especially finite elements are relevant, where the reader is referred, e.g., to [4] and [2].

Open Problems

As the topic is fairly new, there are still many open questions. People are still working on existence and uniqueness theory and extending known results. Especially the simulation of SPDEs, i.e., the numerics evolve a lot in recent years, and so far, no book on the topic is available.



Stochastic Partial Differential Equations, Fig. 2 Segmentation using an SPDE and stochastic active contours. (a) initial condition. (b) Intermediate step. (c) Segmentation result

Experimental Results

To illustrate what an SPDE is and where to use it for, two figures are included. Fig. 1 shows two simulated paths, i.e., $\omega \in \Omega$ was chosen and a finite element method in space and an Euler–Maruyama scheme in time were used. On the left-hand side, the heat equation with additive noise and Dirichlet boundary conditions, i.e.,

$$du(t) = \Delta u(t) dt + dW(t)$$

on the space and time interval $[0, 1]$ with initial condition $u(0, x) = \sin(\pi x)$, is displayed, where W is a Wiener process with space correlation given by the kernel function $q(x, y) = \exp(-2|x - y|)$. On the right-hand side, a hyperbolic equation with additive noise is presented. It is given by

$$du(t) = \nabla u(t) dt + dW(t)$$

with initial condition $u(0, x) = \sin(2\pi x)$ and inflow boundary condition $u(t, 0) = -\sin(2\pi t)$. The other parameters are the same as for the heat equation. In both figures, time evolves from left to right. This and more simulation and convergence results can be found in [1].

One example for an application of SPDEs in computer vision is segmentation. This approach was suggested by Juan et al. [10]. Using level set methods and variational calculus, we obtain as one possible SPDE to be simulated

$$du(t, x) = \nabla \frac{\nabla u(t, x)}{|\nabla u(t, x)|} dt + |\nabla u(t, x)| dW(t, x),$$

where the initial condition is, e.g., a weighted distance function as in Fig. 2a. The red circles are the zero level sets. Different types of noise than the coupling with the size of the gradient can be found in [10] and [12]. An example of the segmentation of a zebra from [12] is shown in Fig. 2.

References

1. Barth A, Lang A (2012) Simulation of stochastic partial differential equations using finite element methods. *Stochastics* 84(2–3):217–231
2. Braess D (2007) Finite elements. Theory, fast solvers and applications in elasticity theory (Finite Elemente. Theorie, schnelle Löser und Anwendungen in der Elastizitätstheorie.) 4th rev and extended edn. Springer, Berlin
3. Chow P-L (2007) Stochastic partial differential equations. Applied mathematics and nonlinear science series. Chapman & Hall/CRC, Boca Raton
4. Dautray R, Lions J-L (2000) Mathematical analysis and numerical methods for science and technology. Springer, Berlin
5. Dellacherie C, Meyer P-A (1978) Probabilities and potential (Transl. from the French.) North-Holland mathematics studies, vol 29. North-Holland, Amsterdam/New York/Oxford
6. Da Prato G, Zabczyk J (1992) Stochastic equations in infinite dimensions. Encyclopedia of mathematics and its applications, vol 44. Cambridge University Press, Cambridge
7. Fishman GS (1996) Monte Carlo. Concepts, algorithms, and applications. Springer, New York
8. Holden H, Øksendal B, Ubøe J, Zhang T (2010) Stochastic partial differential equations. A modeling, white noise functional approach. Universitext, 2nd edn. Springer, New York
9. Ikeda N, Watanabe S (1989) Stochastic differential equations and diffusion processes. North-Holland

- mathematical library, 2nd edn., vol 24. North-Holland, Amsterdam/Kodansha, Tokyo
10. Juan O, Keriven R, Postelnicu G (2006) Stochastic motion and the level set method in computer vision: stochastic active contours. *Int J Comput Vis* 69(1):7–25
 11. Kloeden PE, Platen E (1992) Numerical solution of stochastic differential equations. Applications of mathematics, vol 23. Springer, Berlin
 12. Lang A (2007) Simulation of stochastic partial differential equations and stochastic active contours. PhD thesis, Universität Mannheim
 13. Lions P-L, Souganidis PE (1998) Fully nonlinear stochastic partial differential equations. *C R Acad Sci Paris Sér I Math* 326(9):1085–1092
 14. Métivier M (1982) Semimartingales: a course on stochastic processes. de Gruyter studies in mathematics, vol 2. Walter de Gruyter, Berlin/New York
 15. Øksendal B (2003) Stochastic differential equations. An introduction with applications. Universitext, 6th edn. Springer, Berlin
 16. Pardoux E (1979) Stochastic partial differential equations and filtering of diffusion processes. *Stochastics* 3:127–167
 17. Peszat S, Zabczyk J (2007) Stochastic partial differential equations with Lévy noise. An evolution equation approach. Encyclopedia of mathematics and its applications, vol 113. Cambridge University Press, Cambridge
 18. Prévôt C, Röckner M (2007) A concise course on stochastic partial differential equations. Lecture notes in mathematics, vol 1905. Springer, Berlin
 19. Protter PE (2004) Stochastic integration and differential equations. Applications of mathematics, 2nd edn., vol 21. Springer, Berlin
 20. Walsh JB (1986) An introduction to stochastic partial differential equations (École d'été de probabilités de Saint-Flour XIV - 1984.) *Lect Notes Math* 1180:265–437

Stochastic Relaxation

► Simulated Annealing

Structure-from-Motion (SfM)

► Factorization

Subpixel Estimation

Robert B. Fisher
School of Informatics, University of Edinburgh,
Edinburgh, UK

Synonyms

Superresolution

Definition

Subpixel estimation is the process of estimating the value of a geometric quantity to better than pixel resolution even though the data was originally sampled on an integer pixel quantized space.

Background

It is naively assumed that information at a scale smaller than the pixel level is lost when continuous data is sampled or quantized into pixels from, e.g., time-varying signals, images, data volumes, and space-time volumes. However, in fact, it may be possible to estimate geometric quantities to better than the original pixel accuracy. The underlying foundations of this estimation are the following:

- *Models of expected spatial variation*: Discrete structures, such as edges or lines, produce characteristic patterns of data when measured, allowing fitting of a model to the data to estimate the parameters of the structure.
- *Spatial integration during sampling*: Sensors typically integrate a continuous signal over a finite domain (space or time), leading to measurements whose values depend on the relative position of the sampling window and the original structure.
- *Point-spread function*: Knowledge of the PSF could be used, e.g., by deconvolution of a blurred signal, to estimate the position of the signal.

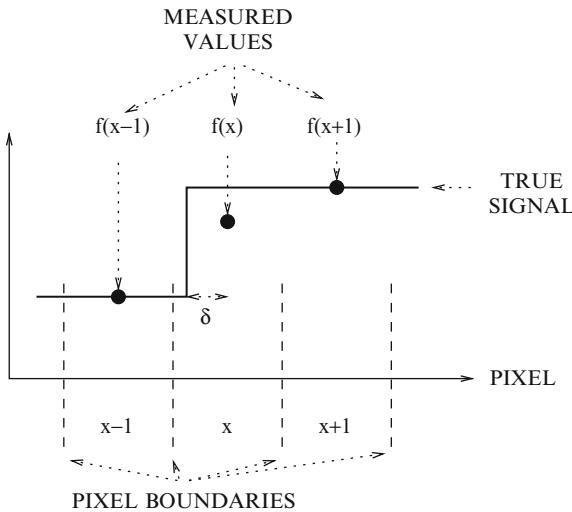
Applications commonly benefitting from subpixel estimation are (1) camera calibration and triangulation (e.g., in stereo and structured light depth estimation) and (2) image motion estimation for improved image stabilization and compression.

One of the earliest instances of subpixel edge detection in computer vision research was by MacVicar-Whelan and Binford [13] in 1981.

The accuracy of subpixel estimation depends on a number of factors, such as the image point-spread function, noise levels, and spatial frequency of the image data. A commonly quoted rule of thumb is 0.1 pixel, but lower is achievable, e.g., about 0.02 pixel is shown for stripe position detection in [1].

Theory

There are four common approaches to estimating subpixel positions:



Subpixel Estimation, Fig. 1 The values of $f(x)$ created by integrating the continuous signal over the whole pixel

1. *Interpolation:* An example is in subpixel edge position estimation, which is demonstrated here in one dimension in ideal form in Fig. 1. One can see that intensity $f(x)$ is a function of the edge's actual position within a pixel and the values at adjacent pixels. Here we assume that the pixel "position" refers to the center of the pixel. Let δ be the offset of the true edge position away from the pixel center. Then, one can model the value $f(x)$ at x in terms of the values at the neighbors, assuming a step function:

$$f(x) = \left(\frac{1}{2} + \delta\right) * f(x-1) + \left(\frac{1}{2} - \delta\right) * f(x+1)$$

from which we can solve for the subpixel edge position $x + \delta$ by:

$$\delta = \frac{2f(x) - f(x-1) - f(x+1)}{2(f(x-1) - f(x+1))}$$

Another approach is to interpolate a continuous curve (or surface) and then find the optimal position on the reconstructed curve (e.g., by using correlation for curve registration).

2. *Integration:* An example is the estimation of the center point of a circular dot, such as required for control point localization in a camera calibration scheme. The assumption is that the minor deviations from many boundary pixels can be accumulated to give a more robust estimate. Suppose that $g(x, y)$

are the gray levels of a light circle on a dark background, where (x, y) are in a neighborhood N closely centered on the circle. Assume also that the mean dark background level has been subtracted from all values. Then, the center of the dot is estimated by its gray-level center of mass:

$$\hat{x} = \frac{\sum_{(x,y) \in N} x g(x, y)}{\sum_{(x,y) \in N} g(x, y)}$$

and similarly for \hat{y} .

3. *Taylor series approximation:* An example is the subpixel feature point position estimation in the SIFT [12] operator. Given the difference of Gaussian function $D(x)$, where x represents the two spatial and one scale dimensions, the Taylor series expansion is:

$$D(x + \delta) = D(x) + \frac{\partial D(x)}{\partial x}^\top \delta + \frac{1}{2} \delta^\top \frac{\partial^2 D(x)}{\partial x^2} \delta$$

Differentiating with respect to δ and setting to 0 give the subpixel (and subscale) estimate:

$$\delta = -\frac{\partial^2 D(x)}{\partial x^2}^{-1} \frac{\partial D(x)}{\partial x}$$

4. *Phase correlation:* The key principle behind phase correlation is the assumption that the pattern of data across a whole window is more distinctive than the individual pixel values. The technique is also independent of intensity, so it can be used for multispectral or illumination-varying registration. Assume that we have two image windows f_a and f_b and their discrete Fourier transforms $F_a = \mathcal{F}(f_a)$ and $F_b = \mathcal{F}(f_b)$. Compute the cross-power spectrum as $F_a F_b^*$ (by elementwise multiplication) where $*$ is the complex conjugate, normalize elementwise by $|F_a F_a^*|$, and finally apply the inverse Fourier transform:

$$T = \mathcal{F}^{-1} \left(\frac{F_a F_b^*}{|F_a F_a^*|} \right)$$

The peak position in T is the desired offset. For subpixel alignment, the above method can be used to remove the integer component of the registration. Thereafter, one can estimate the subpixel peak position of the original registration or repeat the

process on an upsampled version of the image windows once the integer portion of the offset has been removed.

Application

Subpixel methods have been developed to analyze the following:

- *Shape parameters*: circle and other “blob” shape parameters [9], ellipse parameters for improved camera calibration [20], photometric stereo [19], superresolution [18], decomposition of mixed pixels formed by imaging two or more source types [2]
- *Feature positions*: point-like signals [10], “interest” points [12], “edge” transitions [15], “line” transitions [6]
- *Shape matching and registration*: image registration using phase analysis [7, 16] or spatial domain matching [11], motion estimation prior to image compression [17], stereo matching [8] and disparity estimation [14], feature tracking [3], optical flow [4], image and video stabilization [5]

References

1. Alexander BF, Ng KC (1991) Elimination of systematic error in subpixel accuracy centroid estimation. Opt Eng 30(9):1320–1331
2. Bovolo F, Bruzzone L, Carlin L (2010) A novel technique for subpixel image classification based on support vector machine. IEEE Trans Image Process 19(11):2983–2999
3. Brantner S, Auer T, Pinz A (1999) Real-time optical edge and corner tracking at subpixel accuracy. In: Proceedings of the 8th international conference on computer analysis of images and patterns (CAIP). Lecture notes in computer science, vol 1689/1999. Springer, Berlin/New York, pp 534–541
4. Davis CQ, Karul ZZ, Freeman DM (1995) Equivalence of subpixel motion estimators based on optical flow and block matching. In: Proceedings of the international symposium on computer vision, Coral Gables, pp 7–12
5. Erdem C, Erdem AT (2001) An illumination invariant algorithm for subpixel accuracy image stabilization and its effect on MPEG-2 video compression. Signal Process 16(9): 837–857
6. Fisher RB, Naidu DK (1996) A comparison of algorithms for subpixel peak detection. In: Sanz J (ed) Advances in image processing, multimedia and machine vision. Springer-Verlag Berlin/Heidelberg/New York, pp 385–404
7. Foroosh H, Zerubia JB, Berthod M (2002) Extension of phase correlation to subpixel registration. IEEE Trans Image Process 11(3):188–200
8. Henkel RD (1998) Fast stereovision with subpixel-precision. In: Proceedings of the sixth international conference on computer vision, Bombay, pp 1024–1028
9. Hinz S (2005) Fast and subpixel precise blob detection and attribution. In: Proceedings of the IEEE international conference on image processing (ICIP), vol III, Genoa, pp 457–460
10. Jia H, Yang J, Li X (2010) Minimum variance unbiased subpixel centroid estimation of point image limited by photon shot noise. J Opt Soc Am 27(9):2038–2045
11. Karybali IG, Psarakis EZ, Berberidis K, Evangelidis GD (2008) An efficient spatial domain technique for subpixel image registration. Signal Process 23(9):711–724
12. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60(2):91–110
13. MacVicar-Whelan PJ, Binford TO (1981) Intensity discontinuity location to subpixel precision. In: Proceedings of the international joint conferences on artificial intelligence (IJCAI), Vancouver, pp 752–754
14. Morgan GLK, Liu JG, Yan H (2010) Precise subpixel disparity measurement from very narrow baseline stereo. IEEE Trans Geosci Remote Sens 48(9):3424–3433
15. Pedersini F, Sarti A, Tubaro S (1997) Estimation and compensation of subpixel edge localization error. IEEE Trans Pattern Anal Mach Intell 19(11):1278–1284
16. Stone HS, Orchard MT, Chang E-C, Martucci SA (2001) A fast direct fourier-based algorithm for subpixel registration of images. IEEE Trans Geosci Remote Sens 39(10): 2235–2243
17. Suh JW, Jeong J (2004) Fast sub-pixel motion estimation techniques having lower computational complexity. IEEE Trans Consum Electron 50(3):968–973
18. Takeshima H, Kaneko T (2008) Image registration using subpixel-shifted images for super-resolution. In: Proceedings of the 15th IEEE international conference on image processing (ICIP), San Diego, pp 2404–2407
19. Tan P, Lin S, Quan L (2008) Subpixel photometric stereo. IEEE Trans Pattern Anal Mach Intell 30(8):1460–1471
20. Xiao Y, Fisher RB (2010) Accurate feature extraction and control point correction for camera calibration with a mono-plane target. In: Proceedings of the international conference on 3D data processing, visualization and transmission (3DPVT), Paris, electronic proceedings

Subspace Methods

Kazuhiro Fukui

Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Japan

Synonyms

Multiple similarity method

Related Concepts

► Dimensionality Reduction; ► Principal Component Analysis (PCA)

Definition

Subspace analysis in computer vision is a generic name to describe a general framework for comparison and classification of subspaces. A typical approach in subspace analysis is the subspace method (SM) that classifies an input pattern vector into several classes based on the minimum distance or angle between the input pattern vector and each class subspace, where a class subspace corresponds to the distribution of pattern vectors of the class in high-dimensional vector space.

Background

Comparison and classification of subspaces has been one of the central problems in computer vision, where an image set of an object to be classified is compactly represented by a subspace in high-dimensional vector space.

The subspace method is one of the most effective classification method in subspace analysis, which was developed by two Japanese researchers, Watanabe and Iijima around 1970, independently [1, 2]. Watanabe and Iijima named their methods the CLAFIC [3] and the multiple similarity method [4], respectively. The concept of the subspace method is derived from the observation that patterns belonging to a class form a compact cluster in high-dimensional vector space, where, for example, a $w \times h$ pixels image pattern is usually represented as a vector in $w \times h$ -dimensional vector space. The compact cluster can be represented by a subspace, which is generated by using Karhunen-Loëve (KL) expansion, also known as the principal component analysis (PCA). Note that a subspace is generated for each class, unlike the Eigenface Method [5] in which only one subspace (called eigenspace) is generated.

The SM has been known as one of the most useful methods in pattern recognition field since its algorithm

is very simple and it can handle classification of multiple classes. However, its classification performance was not sufficient for many applications in practice, because class subspaces are generated independently of each other [1]. There is no reason to assume a priori that each class subspace is the optimal linear class subspace in terms of classification performance.

To deal with this problem, the SM has been extended. Two typical extensions are the orthogonal subspace method and the learning subspace methods. The orthogonal subspace method [6] executes the SM to a set of class subspaces that are orthogonalized based on the framework proposed by Fukunaga and Koontz [7] in learning phase. The orthogonalization is known as a useful operation to boost the performance of angle-based method, such as SM, since class subspaces are usually close to each other in many classification problems.

The learning subspace methods [1, 8, 9] execute the SM to a set of class subspaces, the boundaries between which are adjusted to suppress classification errors for the learning pattern vectors. This adjustment is performed based on the following procedure. First, a learning vector x is classified by using the SM. Then, if x is wrongly classified into an incorrect class subspace L_r , which is not corresponding to the class of x , subspace L_r is slightly rotated into the direction away from x , and in contrast the correct class subspace L_c of x is slightly rotated to the direction close to x . This adjustment is repeated several times for a set of learning vectors until a minimum classification error is achieved.

Moreover, to deal with the nonlinear distribution of pattern vectors, the SM had also been extended to the kernel nonlinear SM [10, 11] by introducing a nonlinear transformation defined by kernel functions.

These extensions aim mainly to improve the classification ability. In addition to such extensions, the generalization of the SM to classification of sets of patterns is also important for many computer vision problems. In order to handle a set of multiple pattern vectors as an input, the SM has been extended to the mutual subspace method (MSM) [12]. The MSM classifies a set of input pattern vectors into several classes based on multiple canonical (principal) angles [13, 14] between the input subspace and class subspaces, where the input subspace is generated from a set of input patterns as class subspaces. The concept of the MSM is closely related to that of the canonical correlation

analysis (CCA) [13]. Actually, the cosine of the i -th smallest canonical angle corresponds to the i -th largest canonical correlation.

The MSM has achieved high performance in recognition of complicated 3D object such as face, using a set of images from image sequence or multi-view images. This success can be mainly explained by the fact that the MSM implicitly utilizes 3D shape information of objects in classification. This is because the similarity between two distributions of various view images of objects reflects the 3D shape similarity between the two objects. To boost the performance of the MSM, it has been further extended to the constrained mutual subspace method (CMSM) [15] and the whitening (or orthogonal) mutual subspace method (WSM) [16], where the relationship among class subspaces is modified to approach orthogonalization in the learning phase. These extensions have boosted the classification ability of the MSM. The MSM and its extensions also have been further extended to the kernel nonlinear SM [17–20].

Theory

Subspace Method

Assume an input vector p and k class subspaces in f -dimensional vector space. The similarity S of the pattern vector p to the i -th class is defined based on either of the length of the projected input vector \hat{p} on the i -th reference subspace [3] or the minimum angle [4] between the input vector p and the i -th class subspace as shown in Fig. 1a. The length of an input vector p is often normalized to 1.0. In this case, these two criteria coincide. In the following explanation, therefore, the angle-based similarity S defined by the following equation will be used:

$$S = \cos^2 \theta = \sum_{i=1}^{d_q} \frac{(p \cdot \phi_i)^2}{\|p\|^2}, \quad (1)$$

where d_q is the dimension of the class subspace and ϕ_i is the i -th f -dimensional orthogonal normal basis vector of the class subspace, which are obtained from applying the principal component analysis (PCA) to a set of patterns of the class. Concretely, these orthonormal basis vectors can be obtained as the eigenvectors of the correlation matrix $\sum_{i=1}^l x_i x_i^\top$ calculated from the l learning patterns $\{x\}$ of the class.

Process Flow of the SM

The whole process of the SM consists of a learning phase and a recognition phase.

In the Learning Phase All k class d_q -dimensional subspaces are generated from a set of pattern vectors of each class by using PCA.

In the Recognition Phase The similarities S of an input vector p to all the k class subspaces are calculated by using Eq. (1). Then, the input vector is classified into the class of the class subspace with highest similarity. If the highest similarity is lower than a threshold value fixed in advance, the input vector is classified into a reject class.

Mutual Subspace Method

Assume an input subspace and class subspaces in f -dimensional vector space. The similarity of the input subspace to the i -th class subspace is defined based on a minimum canonical angle θ_1 [13, 14] between the input subspace and the class subspace, as shown in Fig. 1b.

Given a d_p -dimensional subspace \mathcal{P} and a d_q -dimensional subspace \mathcal{Q} (for convenience, $d_p \leq d_q$) in the f -dimensional vector space, the canonical angles $\{0 \leq \theta_1, \dots, \theta_{d_p} \leq \frac{\pi}{2}\}$ between \mathcal{P} and \mathcal{Q} are uniquely defined as [14]

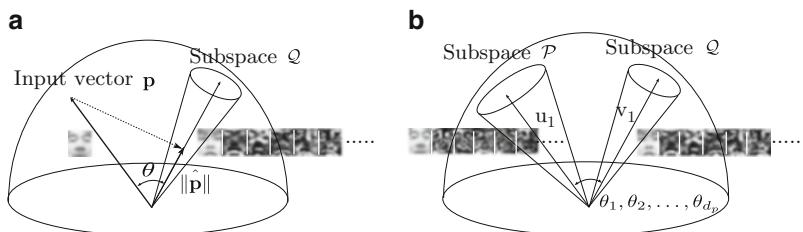
$$\cos^2 \theta_i = \max_{\substack{u_i \perp u_j (i \neq j, i, j = 1 \sim d_p) \\ v_i \perp v_j (i \neq j, i, j = 1 \sim d_p)}} \frac{(u_i \cdot v_i)^2}{\|u_i\|^2 \|v_i\|^2}, \quad (2)$$

where $u_i \in \mathcal{P}, v_i \in \mathcal{Q}, \|u_i\| \neq 0, \|v_i\| \neq 0$, (\cdot) and $\|\cdot\|$ represent an inner product and a norm, respectively.

Let Φ_i and Ψ_i denote the i -th f -dimensional orthonormal basis vectors of the subspaces \mathcal{P} and \mathcal{Q} , respectively. A practical method of finding the canonical angles is by computing the matrix $X = A^\top B$, where $A = [\Phi_1, \dots, \Phi_{d_p}]$ and $B = [\Psi_1, \dots, \Psi_{d_q}]$. Let $\{\kappa_1, \dots, \kappa_{d_p}\}$ ($\kappa_1 \geq \dots \geq \kappa_{d_p}$) be the singular values of the matrix X . The cosines of canonical angles $\{\theta_1, \dots, \theta_{d_p}\}$ can be obtained as $\{\kappa_1, \dots, \kappa_{d_p}\}$. The original MSM uses only a minimum canonical angle θ_1 to define the similarity. However, since the remaining canonical angles also have information for classification, the value, $\tilde{S} = \frac{1}{l} \sum_{i=1}^l \cos^2 \theta_i$, defined from the

Subspace Methods, Fig. 1

Conceptual illustrations of SM and MSM. (a) Subspace method (SM). (b) Mutual subspace method (MSM)



smallest t canonical angles is often used as the similarity in many computer vision problems. The value \tilde{S} reflects the structural similarity between two subspaces. The whole process of the MSM is the same as that of the SM except that an input vector is replaced by an input subspace.

Application

The subspace methods and their extensions have been applied to various problems [1, 10, 11] of computer vision due to their high general versatility and low computational cost. In particular, the extended SMs have produced remarkable results in optical character recognition (OCR), such as handwriting Chinese character recognition [2, 4], in Japanese industry.

The mutual subspace method has also been demonstrated to be extremely effective for 3D object recognition. In particular, the MSM has been known to be suitable for face recognition [15, 16, 21] because the subspace (called “illumination subspace”), which includes any face image patterns under all possible illumination conditions, can be generated from face images under more than three different illumination conditions [22]. The nonlinear extensions of the MSM have been shown to be further effective for 3D object recognition using image sequences, multi-view images [17–20].

References

- Oja E (1983) Subspace methods of pattern recognition. Research Studies Press, Letchworth
- Kuroswawa Y (2007) The engineer’s guide to the subspace method. In: ACCV 2007 workshop Subspace 2007, Tokyo, pp 1–8
- Watanabe S, Lambert PF, Kulikowski CA, Buxton JL, Walker R (1967) Evaluation and selection of variables in pattern recognition. In: Tou J (ed) Computer and information sciences. Academic, New York
- Iijima T, Genchi H, Mori K (1973) A theory of character recognition by pattern matching method. In: Proceedings of 1st international conference on pattern recognition (ICPR), pp 50–56
- Turk M, Pentland A (1991) Eigenfaces for recognition. *J Cogn Neurosci* 3:71–86
- Kittler J (1978) The subspace approach to pattern recognition. *Prog Cybern Syst Res* 3:92
- Fukunaga K, Koontz W (1970) Application of the Karhunen-Loeve expansion to feature selection and ordering. *IEEE Trans Comp* 19(4):311–318
- Kohonen T, Nemeth G, Jalanko M, Riittinen H (1979) Spectral classification of phonemes by learning subspace methods. In: Proceedings of IEEE international conference on acoustics, speech, and signal processing (ICASSP1979), Washington, DC, vol 4, pp 97–100
- Oja E, Kuusela M (1983) The ALSM algorithm – an improved subspace method of classification. *Pattern recognition* 16:421–427
- Maeda E, Murase H (1999) Multi-category classification by kernel based nonlinear subspace method. In: Proceedings of IEEE international conference on acoustics, speech, and signal processing (ICASSP1999), Phoenix, vol 2, pp 1025–1028
- Tsuda K (1999) Subspace classifier in the hilbert space. *Pattern Recogn Lett* 20:513–519
- Maeda K, Watanabe S (1985) A pattern matching method with local structure. *Trans IEICE* J68-D:345–352 (in Japanese)
- Hotelling H (1936) Relations between two sets of variates. *Biometrika* 28:321–377
- Chatelin F (1993) Eigenvalues of matrices (enlarged translation of the French publication with masson). Wiley, Chichester
- Fukui K, Yamaguchi O (2003) Face recognition using multi-viewpoint patterns for robot vision. In: 11th international symposium of robotics research (ISRR2003), Siena, pp 192–201
- Kawahara T, Nishiyama M, Kozakaya T, Yamaguchi O (2007) Face recognition based on whitening transformation of distribution of subspaces. ACCV 2007 workshops Subspace2007, Tokyo, pp 97–103
- Sakano H, Mukawa N (2000) Kernel mutual subspace method for robust facial image recognition. In: Fourth international conference on knowledge-based intelligent engineering systems & allied technologies (KES2000), Brighton, vol 1, pp 245–248
- Wolf L, Shashua A (2003) Learning over sets using kernel principal angles. *J Mach Learn Res* 4: 913–931

19. Fukui K, Stenger B, Yamaguchi O (2006) A framework for 3D object recognition using the kernel constrained mutual subspace method. In: Proceedings of Asian conference on computer vision (ACCV2006), Hyderabad, pp 315–324
20. Fukui K, Stenger B, Yamaguchi O (2007) The kernel orthogonal mutual subspace method and its application to 3D object recognition. In: Proceedings of Asian conference on computer vision (ACCV2007), Tokyo, pp 467–476
21. Yamaguchi O, Fukui K, Maeda K (1998) Face recognition using temporal image sequence. In: Proceedings of IEEE international conference on automatic face and gesture recognition (FG), Nara, pp 318–323
22. Georgiades AS, Belhumeur PN, Kriegman DJ (2001) From few to many: illumination cone models for face recognition under variable lighting and pose. *IEEE Trans Pattern Anal Mach Intell* 23:643–660

Superresolution

► [Subpixel Estimation](#)

Surface Corrugations

► [Surface Roughness](#)

Surface Orientation Histogram (Discrete Version of EGI)

► [Extended Gaussian Image \(EGI\)](#)

Surface Reconstruction

► [Three Dimensional View Integration](#)

Surface Roughness

S. C. Pont
Industrial Design Engineering, Delft University of Technology, Delft, The Netherlands

Synonyms

[Micro scale structure](#); [Surface corrugations](#); [Surface undulations](#)

Related Concepts

► [Bidirectional Texture Function and 3D Texture](#)

Definition

Surface roughness is structure on the microscale of object surfaces. The illumination of such rough surfaces causes shading, shadowing, interreflections, and occlusion effects on the microscale, resulting in 3D texture, which depends strongly on the viewing direction and on the illumination conditions.

Background

Most natural surfaces are rough on the microscale. This microscale structure can be described mathematically with exact geometrical models, with statistical surface height or attitude (slope) distributions.

Surface roughness causes 3D texture in images, which varies over objects as a function of the local viewing angle and illumination conditions. They can be described by the bidirectional texture function or BTF [3]. Thus, textures of rough objects cannot be texture-mapped, in contradistinction to flat, wallpaper-type textures. They need to be synthesized using surface models or photographed BTFs [6]. Even unresolved 3D texture in an image affects material appearance, through effects on the bidirectional reflectance distribution function (BRDF [7]).

Theory

Since the optical effects due to surface roughness are quite complicated, it is hard to formally derive models for 3D textures. Physics-based optical models can be of a geometrical optical or a statistical nature. Physically exact geometrical models are scarce, because for most surface roughness structures the shadowing and interreflections calculations are intractable. As a summary description of surface roughness, one may use such measures as the probability density of heights, the autocorrelation function of heights, and the probability density of orientations of local microfacets. For example, “bump mapping” techniques [4] in computer

graphics regard only the distribution of orientations and ignore differences in height. Indeed much of the image structure generated by 3D texture is due to the fact that surface microfacets differ in orientation and thus receive different illuminances according to Lambert's law [5]. The bidirectional texture contrast function (BTCF [9]) provides robust guesstimates of the attitude distribution. However, the height distribution is also important because it causes such important effects as vignetting, shadowing and occlusion on the microscale.

The field of surface metrology is concerned with surface roughness measurements and descriptions. Using profilometers, surface profiles of real surfaces can be measured, from which roughness parameters can be derived. Roughness parameters usually are statistical measures over lines or areas of the height profile. Many different parameters are in use and can easily be found in engineering literature and via the Internet [11]. Photometric surface metrology from single images suffers from the bas-relief ambiguity [1] and is in computer vision usually referred to as texture analysis [6].

Open Problems

3D texture provides information which is additional to shading; shading is often primarily dependent on the normal component of the illumination, while 3D texture is primarily dependent on the tangential component of the illumination. The spatial structure of 3D texture gradients, e.g., the "illuminance flow," allows inferences about shape and illumination. Formal solutions to the question how exactly shading and 3D texture combine and interact might further the field of shape from shading.

Experimental Results

Several databases of images of rough surfaces can be found on the Internet [2, 8, 10].

References

- Belhumeur PN, Kriegman DJ, Yuille AL (1999) The bas-relief ambiguity. *Int J Comput Vis* 35(1):1573–1405

- CUReT, Columbia–Utrecht reflectance and texture database. <http://www.cs.columbia.edu/CAVE/curet>
- Dana KJ, van Ginneken B (1977) Reflectance and texture of real-world surfaces. In: Proceedings IEEE computer science conference on computer vision and pattern recognition (CVPR)
- Foley JD, van Dam A, Feiner SK, Hughes JF (1990) Computer graphics, principles and practice. Addison-Wesley, Reading
- Lambert JH (1760) *Photometria Sive de Mensure de Gradibus Luminis, Colorum et Umbrae*. Eberhard Klett, Augsburg
- Mirmehdi M, Xie X, Suri J (2008) Handbook of texture analysis. Imperial College Press, London
- Nicodemus FE, Richmond JC, Hsia, JJ (1977) Geometrical considerations and nomenclature for reflectance. National bureau of standards U.S. monograph, vol 160. U.S. Department of Commerce, Washington
- PhoTex, Photometric 3D texture database. <http://www.taurusstudio.net/research/pmtexdb/index.htm>
- Pont SC, Koenderink JJ (2005) Bidirectional texture contrast function. *Int J Comput Vis* 67(1/2):17–34
- Texture Lab PhoTex, Photometric 3D texture database. <http://www.macs.hw.ac.uk/texturelab/resources/databases/Photex/thumbnails.htm>
- Wikipedia, Roughness. <http://en.wikipedia.org/wiki/Roughness>

Surface Scattering

- Asperity Scattering

Surface Undulations

- Surface Roughness

Surfaces

- Differential Geometry of Surfaces in Three-Dimensional Euclidean Space

Surveillance Camera

- Pan-Tilt-Zoom (PTZ) Camera

SW Cut

- Swendsen-Wang Cut Algorithm

Swendsen-Wang Algorithm

Adrian Barbu

Department of Statistics, Florida State University,
Tallahassee, FL, USA

Synonyms

Cluster sampling

Related Concepts

► Swendsen-Wang Cut Algorithm

Definition

The Swendsen-Wang (SW) algorithm [1] is an efficient Markov chain Monte Carlo algorithm for sampling from the Ising/Potts model:

$$\pi_{\text{Potts}}(X) = \frac{1}{Z} \exp \left\{ \sum_{\langle i,j \rangle \in E} \beta_{ij} \delta_{X_i=X_j} \right\}, \quad (1)$$

where δ_A is a Boolean function, equal to 1 if condition A is true and 0 otherwise. The Ising/Potts model is defined for a graph $G = \langle V, E \rangle$ and a labeling $X : V \rightarrow \{1, \dots, L\}$. The obtained model is called Ising model [2] when $L = 2$ and Potts model [3] when $L \geq 3$.

Most computer vision applications use $\beta_{ij} = \beta > 0$, also named the ferro-magnetic model, preferring similar colors for neighboring vertices. The Potts models and its extensions are used as prior probabilities in some Bayesian inference tasks.

Background

The SW algorithm was developed in [1] to overcome some of the limitations of the Gibbs sampler [4] in obtaining samples from the Ising/Potts model (1). If one sets $\beta_{ij} = 1/kT$ where T is a parameter called *temperature* and k is a constant, the Gibbs sampler was observed to slow down, obtaining highly correlated

consecutive samples, around a certain temperature named the critical temperature.

In contrast, consecutive samples obtained by the SW algorithm exhibit much smaller correlation at the critical temperature.

Theory

As opposed to the Gibbs sampler [4] that relabels one node at a time, the SW algorithm changes the label of a cluster of nodes in a single move.

The SW algorithm is illustrated in Fig. 1. At each step, the SW method constructs a new set $F \subset E$ of graph edges, also called the active or “on” edges. This is done by initializing $F = \emptyset$ and adding to F any edge $\langle i, j \rangle \in E$ such that $X_i = X_j$ with probability $1 - e^{-\beta_{ij}}$. A connected component of the new graph $G' = (V, F)$ is selected at random, a new label l is chosen at random among the possible labels $\{1, \dots, L\}$, and all nodes in C are relabeled to l . For more details, see Algorithm 1 below.

Alternatively, the labels of all connected components C of the graph $G' = \langle V, F \rangle$ can be flipped independently.

The SW algorithm is run for many iterations, and after a burn-in period that depends on the number of nodes and the coefficients β_{ij} , the labeling states X will follow the posterior probability Eq. (1).

In a modified version by Wolff [5], one may choose a vertex $v \in V$ and grow a connected component C starting at $C = \{v\}$ and following Bernoulli trials on edges adjacent to C that have not been visited yet. This saves some computation in the clustering step,

Algorithm 1 The Swendsen-Wang Algorithm

Given: Graph $G = \langle V, E \rangle$, $V = (V_1, \dots, V_n)$ and the Ising/Potts model from Eq. (1).

Input: Current labeling state $X = (X_1, \dots, X_n)$.

Output: New labeling state X' .

Set $F = \emptyset$

for all edges $e = \langle i, j \rangle \in E$ with $X_i = X_j$ **do**

Sample $u_{ij} \sim \text{Bernoulli}(p_{ij})$ with $p_{ij} = 1 - e^{-\beta_{ij}}$

if $u_{ij} = 1$ **then**

$F \leftarrow F \cup \{\langle i, j \rangle\}$

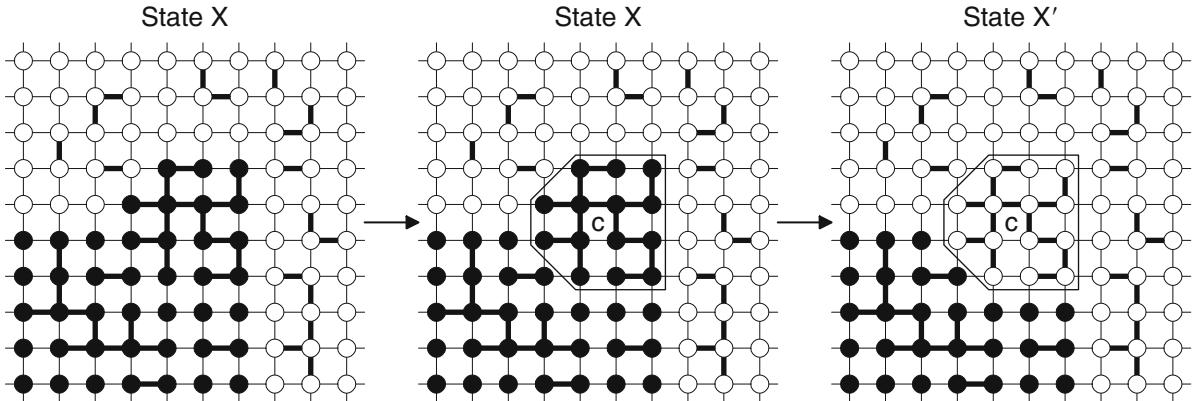
end if

end for

Pick a connected component C of $G' = (V, F)$ at random.

Sample $l \sim \text{Unif}\{1, \dots, L\}$

Set $X'_i = l, \forall i \in C, X'_i = X_i, \forall i \notin C$



Swendsen-Wang Algorithm, Fig. 1 The Swendsen-Wang method. *Left:* In the current labeling state X (*left*), the graph edges (shown with *thin lines*) between same label nodes are turned “on” with probability $p_{ij} = 1 - e^{-\beta_{ij}}$. *Middle:* A

connected component C of the graph of “on” edges is randomly selected, and a new label l is randomly chosen. *Right:* All nodes in C are relabeled with label l , obtaining the new labeling state X'

and bigger components have a higher chance to be selected.

The SW method described above is different from what was presented in the original paper [1]. This description follows the interpretation of Edward and Sokal [6], where the variables $u_{ij}|_{<i,j>\in E}$ are collected into the vector U and the set of active edges is $F(U) = \{<i,j>\in E, u_{ij} = 1\}$. The SW algorithm is explained as an auxiliary variable method that samples from the joint model

$$\begin{aligned} p_{ES}(X, U) &\propto \prod_{<i,j>\in E} [(1 - p_{ij})\delta_{u_{ij}=0} + p_{ij}\delta_{u_{ij}=1} \cdot \delta_{x_i=x_j}] \\ &\propto (1 - p_{ij})^{|E \setminus F(U)|} \cdot p_{ij}^{|F(U)|} \cdot \prod_{<i,j>\in F(U)} \delta_{x_i=x_j}. \end{aligned}$$

The SW algorithm samples from the above joint model $p_{ES}(X, U)$ by alternatively sampling $p_{ES}(U|X)$ and $p_{ES}(X|U)$. Note that the sampling of $p_{ES}(U|X)$ is exactly the part of the SW algorithm that constructs the random edges F , while the sampling of $p_{ES}(X|U)$ is the part that flips the labels of one or all the connected components C of the graph $G' = \langle V, F \rangle$. By sampling (X, U) from the joint model $p_{ES}(X, U)$, the labelings X will follow the marginal $p_{ES}(X)$ which is exactly the Potts model $p_{\text{Potts}}(X)$. On the other hand, the random edges U follow the marginal $p_{ES}(U)$ which is the random cluster model.

Another explanation of the SW algorithm is due to Higdon [7] through the perspective of slice sampling and decoupling. He also introduces partial decoupling, which gives a data-driven clustering step.

The SW algorithm was generalized to arbitrary probabilities in [8], by interpreting it as a Metropolis-Hastings [9, 10] step. This generalization is named the Swendsen-Wang cut algorithm.

An exact sampling method for the Potts model using the SW algorithm was developed by Huber [11], based on coupling from the past [12]. This method eliminates the need for the burn-in period for obtaining samples; however, it is quite conservative and can only be used in practice for small graphs.

Application

Due to being restricted to the Ising/Potts model, there are only a few applications of the SW algorithm to computer vision.

Higdon [7] introduced partial decoupling and presented an application of SW to image reconstruction from positron emission tomography (PET) data. The SW algorithm with partial decoupling was also used in [13] for texture segmentation using a model with Potts prior and a data term. The SW algorithm was compared with the Gibbs sampler and SW with partial decoupling in [14].

Morris [15] introduced a higher-order prior model named the “chien” model that is not based on pairwise interactions, but on 3×3 cliques. He used the SW algorithm with partial decoupling for obtaining samples from this model.

The SW cut algorithm, a SW generalization to arbitrary probabilities and edge weights, has seen many applications to image, motion, and object segmentation as well as stereo matching and curve grouping, to name only a few. For more details, see the SW cut entry of the encyclopedia.

References

1. Swendsen R, Wang J (1987) Nonuniversal critical dynamics in Monte Carlo simulations. *Phys Rev Lett* 58(2):86–88
2. Ising E (1925) Beitrag zur theorie des ferromagnetismus. *Zeitschrift fur Physik A Hadrons and Nuclei* 31(1):253–258
3. Potts R (1952) Some generalized order-disorder transformations. In: Mathematical proceedings of the Cambridge philosophical society, vol 48. Cambridge University Press, Cambridge, pp 106–109
4. Geman S, Geman D, Relaxation S (1984) Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans Pattern Anal Mach Intell* 6(2):721–741
5. Wolff U (1989) Collective Monte Carlo updating for spin systems. *Phys Rev Lett* 62(4):361–364
6. Edwards R, Sokal A (1988) Generalization of the Fortuin-Kasteleyn-Swendsen-Wang representation and Monte Carlo algorithm. *Phys Rev D* 38(6):2009–2012
7. Higdon D (1998) Auxiliary variable methods for Markov Chain Monte Carlo with applications. *J Am Stat Assoc* 93(442):585–595
8. Barbu A, Zhu S (2005) Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities. *IEEE Trans Pattern Anal Mach Intell* 27(8):1239–1253
9. Metropolis N, Rosenbluth A, Rosenbluth M, Teller A, Teller E, et al (1953) Equation of state calculations by fast computing machines. *J Chem Phys* 21(6):1087
10. Hastings W (1970) Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57(1):97
11. Huber M (2003) A bounding chain for Swendsen-Wang. *Random Struct Algorithm* 22(1):43–59
12. Propp J, Wilson D (1996) Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Struct Algorithm* 9(1–2):223–252
13. Barker S, Rayner P (2002) Unsupervised image segmentation. In: Proceedings of the ICASSP 2002, Orlando, vol 5. pp 2757–2760
14. Hurn M (1997) Difficulties in the use of auxiliary variables in Markov chain Monte Carlo methods. *Stat Comput* 7(1):35–44
15. Morris R (1999) Auxiliary variables for Markov random fields with higher order interactions. *EMMCVPR*, York, pp 731–731

Swendsen-Wang Cut Algorithm

Adrian Barbu

Department of Statistics, Florida State University, Tallahassee, FL, USA

Synonyms

[SW cut](#)

Related Concepts

► [Simulated Annealing](#); ► [Swendsen-Wang Algorithm](#)

Definition

The Swendsen-Wang Cut algorithm is an efficient Markov chain Monte Carlo (MCMC) algorithm for sampling arbitrary distributions $p(X)$ defined on partitions X of a graph $G = \langle V, E \rangle$. It uses a set of weights on the graph edges to form data-driven clusters and change the label of an entire cluster in one move.

Background

Many computer vision problems can be formulated as the optimization of an energy or probability function defined over the space of partitions or labelings of a graph. Efficient algorithms such as graph cuts [1], belief propagation [2], or dual decomposition [3] can be used for certain types of energy functions.

However, for general forms of the energy or probability functions, many times one has to resort to stochastic relaxation (Gibbs sampler) [4] methods that relabel one node at a time. Such algorithms can be very slow in practice, especially when strong regularization is imposed on the partitions.

For Ising/Potts models [5, 6], the Swendsen-Wang (SW) algorithm [7, 8] was designed to overcome some of the limitations of the stochastic relaxation. The SW algorithm achieves speedup by relabeling a cluster of adjacent nodes of the graph in a single move.

Theory

The Swendsen-Wang cuts algorithm is a generalization of the Swendsen-Wang algorithm to arbitrary distributions by adding an acceptance step for the proposed MCMC cluster relabeling move. The acceptance probability can be computed efficiently using Eq.(3).

Let $G = \langle V, E \rangle$ be a graph, $q_e \in [0, 1], \forall e \in E$ a set of weights on the graph edges, and $p(X)$ a probability defined for any partition or labeling $X = (X_1, \dots, X_n)$ of the graph nodes $V = (V_1, \dots, V_n)$. The probability $p(X)$ can be defined up to a constant. Assume also given a probability mass function $q(l|C, X)$ defined over the set possible labels l , given a labeling state X and a set of nodes $C \subset V$. The function $q(l|C, X)$ could be as simple as a uniform distribution or can be driven by the image data.

For any labeling X of the graph nodes and any label l , define $V_l(X)$ to be the set of nodes with label l :

$$V_l(X) = \{i \in V, X_i = l\}. \quad (1)$$

For any two subsets $C, D \subset V$, define the *cut* from C to D as the set of edges:

$$\mathcal{C}(C, D) = \{e = \langle i, j \rangle \in E, i \in C, j \in D\}. \quad (2)$$

With these notations, one move of the SW cut algorithm proceeds as follows, also illustrated in Fig. 1.

The acceptance probability $\alpha(X \rightarrow X')$ has a factor defined in terms of the cuts $\mathcal{C}(C, V_l(X) \setminus C)$ and $\mathcal{C}(C, V_{l'}(X) \setminus C)$ that contain the edges from C toward the nodes with the old label l and the proposed new label l' , respectively. These cuts are shown with dotted lines in Fig. 1.

The algorithm was observed to be hundreds of times faster than the Gibbs sampler when the edge weights q_e were chosen appropriately. The edge weights should approximate the probability that the two adjacent nodes belong to the same label. This can be achieved using the image information available at the graph nodes and could be learned in a discriminative way, for example, using boosting or regression.

The SW cut algorithm can be used for maximum a posteriori (MAP) estimation using a simulated annealing schedule, in which the probability

$p(X)$ is raised to increasingly larger powers, forcing the sampling to focus on the labelings X of highest probability.

A Wolff [9] version of the SW cut algorithm has also been proposed [10, 11], which grows a connected component from a seed node. This version reduces the amount of computation required for the clustering step, and larger connected components have a higher chance to be selected.

Application

The SWC algorithm has many applications in computer vision. One application was image segmentation [10, 12], where the graph had image superpixels as nodes and edges based on the superpixel adjacency. The edge weights q_e were based on the similarity between the intensity histograms of the corresponding superpixels. The probability $p(X)$ was a Bayesian

Algorithm 1 The Swendsen-Wang Cut Algorithm

Given: Weighted graph $G = \langle V, E \rangle$, $V = (V_1, \dots, V_n)$ and a probability $p(X)$.

Input: Current labeling state $X = (X_1, \dots, X_n)$.

Output: New labeling state X' .

Set $F = \emptyset$

for all edges $e = \langle i, j \rangle \in E$ with $X_i = X_j$ **do**

 Sample $u \sim \text{Bernoulli}(q_e)$

if $u = 1$ **then**

$F \leftarrow F \cup \{\langle i, j \rangle\}$

end if

end for

Pick a connected component C of $G' = (V, F)$ at random, with some label $X_C = l$.

Sample a new label $l' \sim q(l'|C, X)$

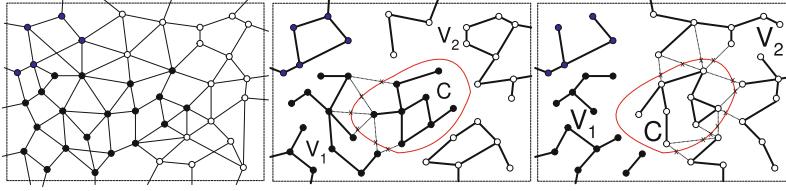
Sample $u \sim \text{Bernoulli}(\alpha(X \rightarrow X'))$ where $X'_i = l', \forall i \in C, X'_i = X_i, \forall i \notin C$ and

$$\alpha(X \rightarrow X') = \min \left(1, \frac{\prod_{e \in \mathcal{C}(C, V_{l'}(X) \setminus C)} (1 - q_e)}{\prod_{e \in \mathcal{C}(C, V_l(X) \setminus C)} (1 - q_e)} \cdot \frac{q(l|C, X')}{q(l'|C, X)} \cdot \frac{p(X')}{p(X)} \right). \quad (3)$$

if $u = 0$ **then**

 Reject the move i.e. make $X' = X$.

end if



Swendsen-Wang Cut Algorithm, Fig. 1 Illustration of the SW cut algorithm. *Left:* A graph $G = \langle V, E \rangle$ with a labeling X . *Middle:* At each step, a new graph $G' = \langle V, F \rangle$ is formed by going through all edges between same label nodes and keeping

an edge e with probability given by its weight q_e . *Right:* Then a connected component C of G' is selected at random, a new label l' is chosen for C , and if the move is accepted, the new labeling state has $X_C = l'$

model with a prior based on connected components and was maximized by SW cut with simulated annealing. Experiments showed that the SW cut algorithm was two orders of magnitude faster in total CPU time when compared to the Gibbs sampler tuned to obtain the same optimal result.

Other applications of the SW cut algorithm include curve grouping [12], dense stereo matching [10, 13], motion segmentation [11, 14] and estimation [15], object segmentation [16], and task allocation [17] in robotics. Furthermore, the SW cut algorithm was used for discovering composite features for object detection [18] and for graph matching [19].

Open Problems

In problems, higher-level objects can sometimes be detected only when at least three components are considered simultaneously. For such cases, it is an open question how to generalize the SW cut algorithm to use higher-order cliques instead of graph edges to construct the clusters.

For example, finding lines in point clouds can be seen as the problem of labeling the points that belong to each line and the remaining points as background. But since nontrivial collinearity exists only between at least three points, it is difficult to define edge weights directly between the points. A generalization of the SW cut algorithm could use triplets of collinear points instead of graph edges to form point clusters that are likely to be on the same line. It is not known how exactly to form the clusters from such higher-order cliques and what is the acceptance probability for relabeling such a cluster.

References

1. Boykov Y, Veksler O, Zabih R (2002) Fast approximate energy minimization via graph cuts. *IEEE Trans Pattern Anal Mach Intell* 23(11):1222–1239
2. Yedidia J, Freeman W, Weiss Y (2001) Generalized belief propagation. In: NIPS 2001, Vancouver, pp 689–695
3. Komodakis N, Paragios N, Tziritas G (2007) MRF optimization via dual decomposition: message-passing revisited. In: ICCV 2007, Rio de Janeiro
4. Geman S, Geman D, Relaxation S (1984) Gibbs distributions, and the bayesian restoration of images. *IEEE Trans Pattern Anal Mach Intell* 6(2):721–741
5. Ising E (1925) Beitrag zur theorie des ferromagnetismus. *Zeitschrift fur Physik A Hadrons and Nuclei* 31(1):253–258
6. Potts R (1952) Some generalized order-disorder transformations. In: Mathematical proceedings of the cambridge philosophical society, vol 48. Cambridge University Press, Cambridge, pp 106–109
7. Swendsen R, Wang J (1987) Nonuniversal critical dynamics in Monte Carlo simulations. *Phys Rev Lett* 58(2):86–88
8. Edwards R, Sokal A (1988) Generalization of the fortuin-kasteleyn-swendsen-wang representation and monte carlo algorithm. *Phys Rev D* 38(6):2009–2012
9. Wolff U (1989) Collective Monte Carlo updating for spin systems. *Phys Rev Lett* 62(4):361–364
10. Barbu A, Zhu S (2005) Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities. *IEEE Trans Pattern Anal Mach Intell* 27(8):1239–1253
11. Barbu A, Zhu S (2007) Generalizing swendsen-wang for image analysis. *J Comput Graph Stat* 16(4):877–900
12. Barbu A, Zhu S (2003) Graph Partition by Swendsen-Wang Cuts. In: ICCV 2003, Nice, p 320
13. Kim W, Park J, Lee K (2009) Stereo matching using population-based MCMC. *Int J Comput Vis* 83(2):195–209
14. Barbu A, Zhu S (2004) Multigrid and multi-level Swendsen-Wang cuts for hierachic graph partition. In: CVPR 2004, Washington, DC
15. Barbu A, Yuille A (2004) Motion estimation by Swendsen-Wang cuts. In: CVPR 2004, Washington, DC
16. Wang J, Betke M, Gu E (2005) MosaicShape: stochastic region grouping with shape prior. In: IEEE conference on computer vision pattern recognition (CVPR 2005), San Diego, vol 1, pp 902–908

17. Zhang K, Collins E, Barbu A (2010) A novel stochastic clustering auction for task allocation in multi-robot teams. In: IROS 2010, Taipei
18. Han F, Shan Y, Sawhney H, Kumar R (2008) Discovering class specific composite features through discriminative sampling with swendsen-wang cut. In: IEEE conference on computer vision pattern recognition (CVPR 2008), Anchorage, pp 1–8
19. Lin L, Zhu S, Wang Y (2007) Layered graph match with graph editing. In: IEEE conference on computer vision pattern recognition (CVPR 2007), Minneapolis, pp 1–8

Symmetry Detection

► Computational Symmetry

Symmetry-Based X

► Computational Symmetry

T

Terminological Logics

- ▶ [Description Logics](#)
-

Texture Classification

- ▶ [Texture Recognition](#)
-

Texture Recognition

Matti Pietikäinen
Department of Computer Science and Engineering,
University of Oulu, Oulu, Finland

Synonyms

- [Texture classification](#)

Related Concepts

- ▶ [Bidirectional Texture Function and 3D Texture](#)

Definition

Texture recognition deals with classification of images or regions based on their textural properties.

Background

Texture is an important characteristic of images. It can be seen almost anywhere. A textured region in an

image can be characterized by a nonuniform or varying spatial distribution of intensity or color. The specific structure of the texture depends on the surface topography and albedo, the illumination of the surface, and the position and frequency response of the viewing camera.

In texture recognition, the goal is to assign an unknown sample image to one of a set of known texture classes. A large number of different approaches for texture description have been proposed. A major problem is that textures in the real world are often not uniform, due to changes in orientation, scale, or other visual appearance. In addition, the degree of computational complexity of many of the proposed texture measures is very high. In recent years, very discriminative and computationally efficient descriptors have been developed, such as local binary patterns (LBP), which allows to use texture information as a generic tool in a wide variety of computer vision applications.

Overview of Texture Recognition

Texture recognition process involves two phases: the learning phase and the recognition phase. In learning, the goal is to build a model for the texture content of each texture class present in the training data, which generally comprises of images with known class labels. The texture content of the training images is analyzed with the chosen texture operators, yielding a set of textural features for each image. These features characterize certain textural properties of the images, such as spatial structure, contrast, roughness, and orientation. In the recognition phase, the texture content

of the unknown sample is first described with the same texture features. Then the features of the given sample are compared to those of the training images with a classification algorithm, and the sample is assigned to the category with the best match. Optionally, if the best match is not sufficiently good according to some predefined criteria, the unknown sample can be rejected.

A large number of different approaches for texture description have been proposed [1, 2]. Tuceryan and Jain [1] divided these methods into four categories: statistical, geometrical, model-based, and signal processing. Among the most widely used traditional approaches are statistical methods based on co-occurrence matrices of second-order gray-level statistics or first-order statistics of local property values (difference histograms), signal processing methods based on local linear transforms, multichannel Gabor filtering or wavelets, and model-based methods based on Markov random fields or fractals. More recent approaches include local binary pattern (LBP) and its different variants [3, 4], methods based on 2D and 3D textons [5], and the local statistical operators representing textures as a set of exemplar patches [6]. For classification, the nearest neighbor or k nearest neighbor classifier has been commonly used. In recent years, however, the support vector machine (SVM) has gained major interest, providing excellent performance in many studies. Also boosting-based methods such as AdaBoost and bagging classifiers like the random forest classifier have been successfully applied to texture recognition problems.

In recent years, the division between texture descriptors and more general image or video descriptors has been disappearing. The function of descriptors is to convert the pixel-level information into a useful form, which captures the most important factors of the imaged scene but is insensitive to irrelevant aspects caused by the varying environment. The effective descriptor is able to ignore the irrelevant aspects caused by the changes in the environment. This should be furthermore done without compromising the descriptive power of the method. While the definition of irrelevant depends on the application, the most common cases are related to imaging conditions like illumination, viewing angle, scale, noise, and blur. Currently, LBP, SIFT (scale-invariant feature transform) [7], and HOG (histogram of oriented gradients) [9], and their variants, are the most

effective and commonly used descriptors, providing complementary information about the image contents. In many applications, a single descriptor is not enough, but a proper combination of different descriptors should be used.

The local descriptors are normally used in three different ways. One is a sparse descriptor (SIFT) which first detects salient interest points in a given image and then samples a local patch and describes its invariant features. In [8, 10], a sparse descriptor is used for texture and object recognition. Another is HOG, which is computed on a dense grid of uniformly spaced cells. Most often, the ordinary texture descriptors are used densely, obtained by regular sampling of the input image or region. In recent years, the LBP has been the most widely used dense texture descriptor but can also be used as a sparse descriptor like SIFT or computed on a grid like HOG.

Local Binary Patterns

The local binary pattern is a simple yet very efficient texture operator. It is based on the assumption that texture has locally two complementary aspects: a pattern and its strength. The pixels of an image are labeled by thresholding the neighborhood of each pixel, and the result is considered as a binary number. The distribution of the LBP labels computer over a region is then used for texture description.

The original LBP operator shown Fig. 1 [4, 11] works in a 3×3 neighborhood, using the center value as a threshold. The thresholded values are multiplied with weights of the corresponding pixels, and summing up the result an LBP code is obtained. The contrast measure is obtained by subtracting the average gray levels below the threshold from that of the gray levels above (or equal to) the center pixel. If all neighbors of the center pixel have the same value (1 or 0), the value of C is set to 0.

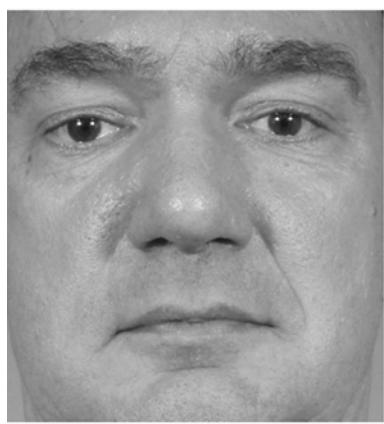
The distributions of LBP codes or 2D distributions of LBP and C are used as features in texture recognition. Figure 2 shows an example of LBP image and histogram obtained for a face image [4].

The LBP operator was extended to use neighborhoods of different sizes [3]. Using a circular neighborhood and bilinearly interpolating values at non-integer pixel coordinates allow any radius and number of pixels in the neighborhood. The gray-scale variance of the

Texture Recognition, Fig. 1

The original LBP

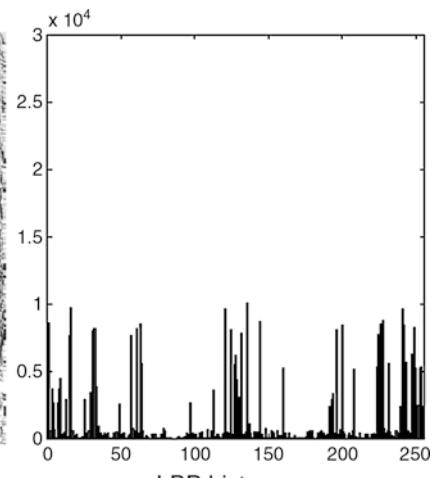
| example | | | thresholded | | | weights | | |
|---------------------------|---|---|-------------|---|---|---------|---------------------------------------|----|
| 6 | 5 | 2 | 1 | 0 | 0 | 1 | 2 | 4 |
| 7 | 6 | 1 | 1 | | 0 | 128 | | 8 |
| 9 | 8 | 7 | 1 | 1 | 1 | 64 | 32 | 16 |
| Pattern = 11110001 | | | | | | | $LBP = 1 + 16 + 32 + 64 + 128 = 241$ | |
| | | | | | | | $C = (6+7+9+8+7)/5 - (5+2+1)/3 = 4.7$ | |



Input Image



LBP Image



LBP histogram

Texture Recognition, Fig. 2 Example of an input image, the corresponding LBP image and histogram

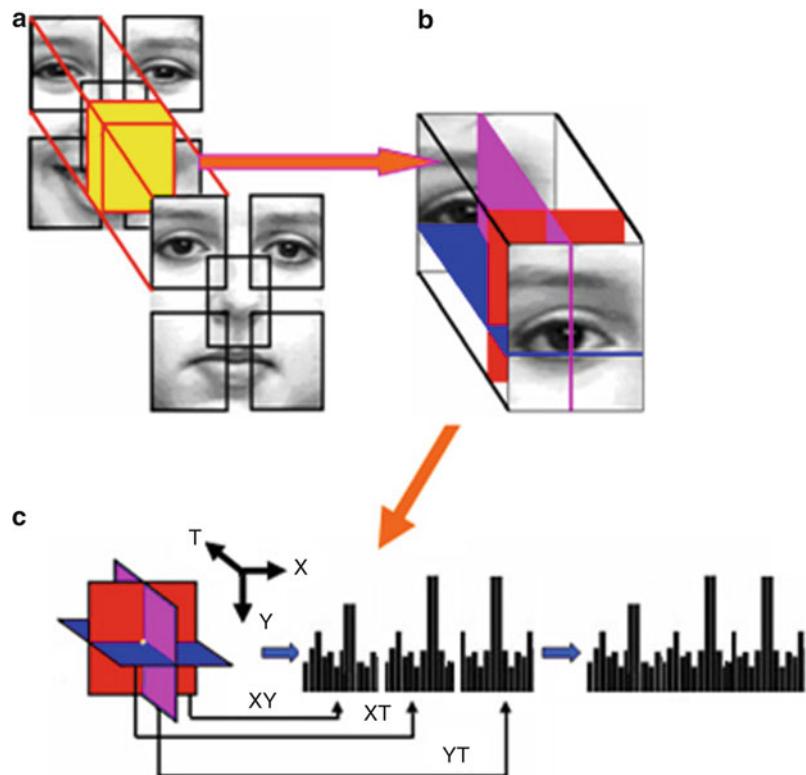
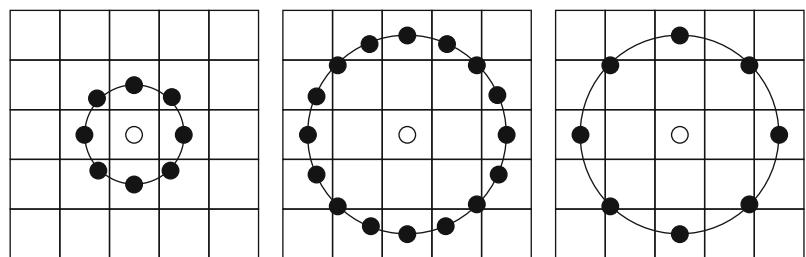
local neighborhood can be used as the complementary contrast measure. [Figure 3](#) presents examples of circular sampling [4].

Due to its discriminative power and computational simplicity, LBP texture operator has become a popular approach in various applications. It can be seen as a unifying approach to the traditionally divergent statistical and structural models of texture analysis. Perhaps the most important property of the LBP operator in real-world applications is its robustness to monotonic gray-scale changes caused, for example, by illumination variations. Another important property is its computational simplicity, which makes it possible to analyze images in challenging real-time settings.

Dynamic texture recognition has received considerable interest in recent years [12]. Dynamic texture descriptors provide a new and very effective tool for motion analysis. The original LBP operator was

defined to only deal with the spatial information. Later, it was extended to a spatiotemporal representation for dynamic texture analysis [13]. For this purpose, the so-called volume local binary pattern (VLBP) operator was proposed, in which dynamic texture is considered as a set of volumes in the (X,Y,T) space where X and Y denote the spatial coordinates and T denotes the frame index (time). The neighborhood of each pixel is thus defined in three-dimensional space. The VLBP combines motion and appearance together to describe dynamic textures. To make VLBP computationally simple and easy to extend, an operator based on co-occurrences of local binary patterns on three orthogonal planes (LBP-TOP) considers three orthogonal planes, XY, XT, and YT, and concatenates local binary pattern co-occurrence statistics in these three directions as shown [Fig. 4](#) [13]. The circular neighborhoods are generalized to elliptical sampling to fit to the space-time statistics.

Texture Recognition, Fig. 3
Examples of circular sampling



Texture Recognition, Fig. 4
LBP-TOP histogram
computed from three
orthogonal planes

The success of LBP in different types of problems and applications has inspired much new research on different variants of LBP, with an aim to increase its discriminative power and robustness [4].

Application

The traditional areas of application considered for texture recognition included, for example, biomedical image analysis, industrial inspection, analysis of satellite or aerial imagery, document image analysis, and texture synthesis for computer graphics or animation.

With recent progress in research, texture information can now be used as a generic tool in various modern applications of computer vision, including face analysis, biometrics, object recognition, motion and activity analysis, and content-based retrieval from image or video databases [4]. For the LBP method, the most widely used application area so far has been facial image analysis, including face recognition, face authentication, spoofing detection, facial expression recognition, gender recognition, age estimation, and visual speech recognition. In biometrics, it has been also applied to iris recognition, fingerprint recognition, palmprint recognition, finger vein recognition, and gait recognition, for example. The use of

texture recognition for various biomedical applications using different types of images has increased rapidly in recent years, ranging from microscopic images to X-ray or magnetic resonance images.

Summary

Texture is an important characteristic of images and videos. In recent years, very discriminative and computationally efficient descriptors been developed, such as local binary patterns, which allow to use texture information as a generic tool in a wide variety of computer vision problems and applications. The focus of the research has broadened from 2D textures to 3D textures and dynamic textures. The division between texture descriptors and more general image or video descriptors is disappearing.

References

1. Tuceryan M, Jain AK (1998) Texture analysis. In: Chen CH, Pau LF, Wang PSP (eds) *The handbook of pattern recognition and computer vision*, 2nd edn. World Scientific, Singapore, pp 207–248
2. Mirmehdi M, Xie X, Suri J (eds) (2008) *Handbook of texture analysis*. Imperial College Press, London, p 413
3. Ojala T, Pietikäinen M, Mäenpää T (2002) Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans Pattern Anal Mach Intell* 24(7):971–987
4. Pietikäinen M, Hadid A, Zhao G, Ahonen T (2011) Computer vision using local binary patterns. Springer, London, p 207
5. Leung T, Malik J (2001) Representing and recognizing visual appearance of materials using three-dimensional textures. *Int J Comput Vis* 43(1):29–44
6. Varma M, Zisserman A (2009) A statistical approach to material classification using image patch exemplars. *IEEE Trans Pattern Anal Mach Intell* 31(11):2032–2047
7. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110
8. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: Proceedings of computer vision and pattern recognition (CVPR), San Diego, vol 1, pp 886–893
9. Lazebnik S, Schmid C, Ponce J (2005) A sparse texture representation using local affine regions. *IEEE Trans Pattern Anal Mach Intell* 27(8):1265–1278
10. Zhang J, Marszalek M, Lazebnik S, Schmid C (2007) Local features and kernels for classification of texture and object categories: a comprehensive study. *Int J Comput Vis* 73(2):213–238
11. Ojala T, Pietikäinen M, Harwood D (1996) A comparative study of texture measures with classification based on feature distributions. *Pattern Recognit* 29(1):51–59
12. Saisan P, Doretto G, Wu YN, Soatto S (2001) Dynamic texture recognition. In: Proceedings of computer vision and pattern recognition (CVPR), Kauai, vol 2, pp 58–63
13. Zhao G, Pietikäinen M (2007) Dynamic texture recognition using local binary patterns with an application to facial expressions. *IEEE Trans Pattern Anal Mach Intell* 29(6):915–928

Thermal Radiator

- [Blackbody Radiation](#)
- [Planckian Locus](#)

Three Dimensional Descriptor

- [Three Dimensional Shape Descriptor](#)

Three Dimensional Estimation

- [Depth Estimation](#)

Three Dimensional Face Modeling

- [Face Modeling](#)

Three Dimensional Modeling from Images

- [Image-Based Modeling](#)

Three Dimensional Reconstruction

- [Factorization](#)

Three Dimensional Shape Descriptor

Federico Tombari
DEIS, University of Bologna, Bologna, Italy

Synonyms

[Three dimensional descriptor](#)

Definition

A 3D shape descriptor is a computational representation, in the form of a vector, of a set of points belonging to a surface. This set can span from a small local neighborhood of a point to the entire surface.

Background

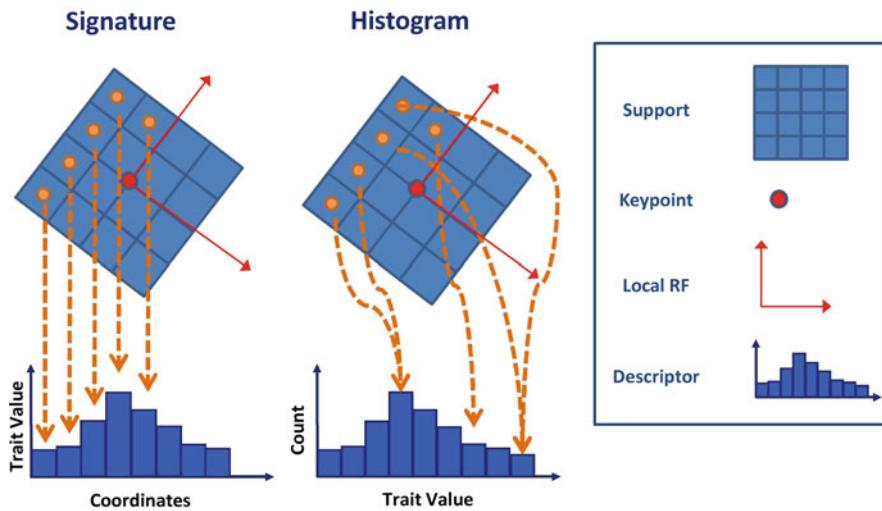
The use of 3D descriptors stems from the need, in many 3D computer vision tasks, of measuring similarities between surfaces. By computing *descriptions* of either single parts or the entire surface in the form of vectors, these can then be classified using machine learning tools, or matched by applying suitable metrics (e.g., Euclidean distance) so as to establish correspondences among surfaces. The theory of descriptors can be easily applied to all common representations of 3D data, such as point clouds, meshes, vowels, range images, etc. In addition, this process of translating the original 3D representation to the form of vectors is suitable for evaluating similarities between different kinds of 3D representations (e.g., comparing a 3D mesh against a range map).

The two main characteristics of a 3D shape descriptor are *descriptiveness* and *robustness*. The former can be defined as the ability of embedding into the descriptor salient characteristics of the set of points being described, so that the resulting vector is as different (or distinctive) as possible from the same descriptor related to different shapes, or different neighborhoods on the same shape. As for robustness, a descriptor should appropriately deal with the various nuisances that can be present in 3D data. More specifically, descriptors are generally designed to be invariant to the class of transformations that a shape can undergo within a certain application domain. Typical classes of transformations include arbitrary rotations, translations, scale changes, and reflections. If a descriptor is robust to such transformations, the successive matching/classification stage can be performed independently from the specific data nuisances. More challenging nuisances are represented by partial occlusions/missing parts of the surface, topology/point density variations, nonrigid deformations up to isometries, etc. [1, 2]. In addition, also sensor noise often represents a relevant nuisance.

Theory

A common dichotomy divides 3D shape descriptors between *local* (or *feature-based*) and *global* approaches. The distinctive trait of such a subdivision regards the fraction of the surface used to compute the description (i.e., the *support*): as for local methods, it is usually a small neighborhood around distinctive 3D points (i.e., local *features*, or *key-points*), whereas for global methods it coincides with the entire surface, since a single description encompassing the whole shape is generally computed. The choice between a local and a global method typically depends on the application. Local methods can better deal with the presence of clutter, which typically occurs when the 3D shape has to be recognized and localized amid other (sometimes similar) surfaces. By spatially limiting the support to a small neighborhood of a feature point, the presence of clutter does not affect the description of internal points of the surface. For the same reason, local descriptors are also useful to deal with partial surface occlusion. On the other hand, global descriptors have proved to be particularly effective in the absence of clutter, since they can be designed to handle a broad range of invariants such as those aforementioned [1]. Also, they can generally lead to higher efficiency, since only one descriptor has to be computed for each surface. It is important to note that global and local approaches are often interchangeable, with local descriptors being deployed for describing the whole surface and vice versa.

Relevant references for global descriptor approaches can be found in [3–5]. In addition, [4] proposes also a taxonomy of global descriptors, grouping methods into four different classes (*histograms*, *transforms*, *2D views*, *graphs*). On the other hand, most recent proposals concern local descriptors, due to their aforementioned capabilities of withstanding clutter and occlusions, and, at the same time, given recent outcomes where they proved to be suitable also for object categorization [1]. A recent taxonomy [6], encompassing most recent proposals, in turn divides local approaches between two main subcategories recalled as *Histograms* and *Signatures* (see Fig. 1 for a graphical explanation). According to this classification, a histogram-based descriptor accumulates local geometrical or topological measurements (e.g., point counts, mesh triangle areas, etc.) into histograms according to a specific quantized domain (e.g., point coordinates,



Three Dimensional Shape Descriptor, Fig. 1 According to [6], most local descriptors can be divided between signatures and histograms. The former class encodes one or more geometric traits computed on each point of the support according to local coordinates, the latter accumulates the number of points

or the triangle area according to a specific quantized domain. Signatures always require a local Reference Frame to define the coordinates within the support, histograms only if their domain is based on coordinates

curvatures, etc.). A typical example of a histogram-based descriptor is *Spin Images* [7], which computes a 2D histogram by accumulating geometrical attributes of the subset of points falling within a cylindrical volume centered on the point being described. These attributes are the 2D coordinates of each point projected on a plane that “spins” around the normal of the point being described. Another relevant approach is *3D Shape Context* [8], which modifies the basic idea of Spin Images by representing the descriptor vector as a 3D histogram, where the geometrical attributes being accumulated are the coordinates of the points falling within a sphere centered at the point being described.

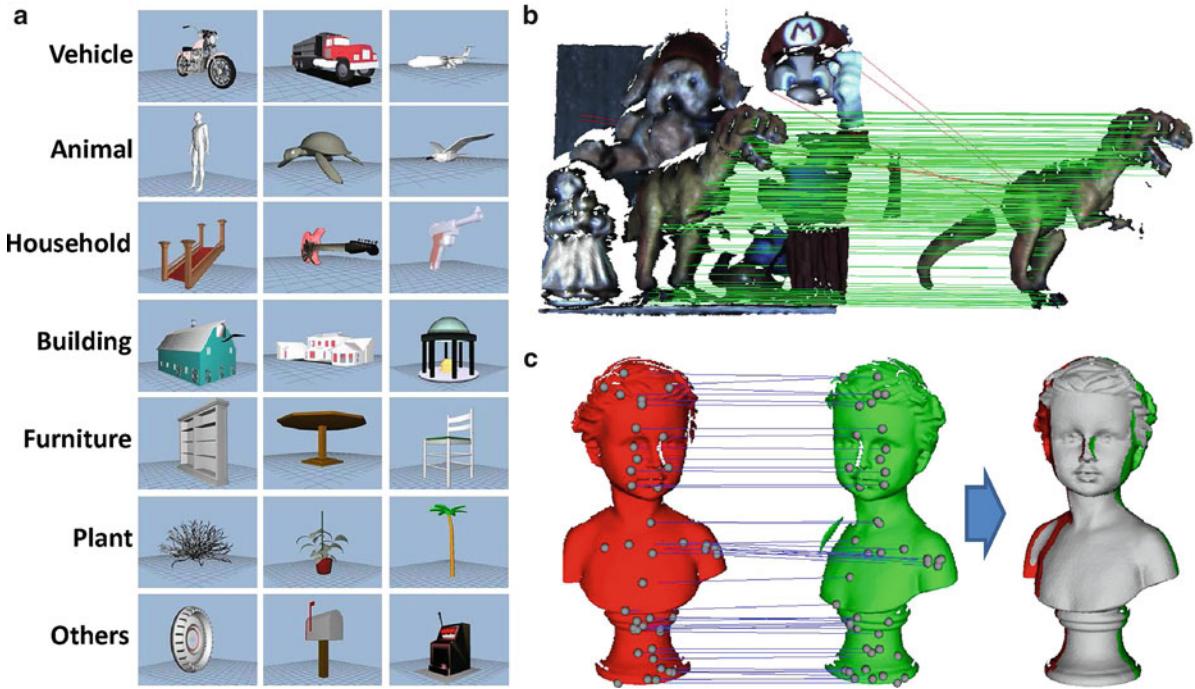
Differently, a signature-based descriptor encodes one or more geometric measurements computed individually on each point within the support according to the point coordinates. For the sake of invariance, this encoding has to be ordered invariantly with respect to the point coordinates: this is usually dealt with by computing a local Reference Frame over the support and encoding each measurement in an order expressed via the point coordinates of this local Reference Frame. One of the most prominent signature-based descriptors is *Point Signatures* [9], where the measurement set is given by the signed height of the 3D curve resulting at the intersection between the spherical support and

the surface. The signature is then obtained by orderly sweeping this curve, with the starting point indicated by one axis of the local Reference Frame defined in [9]. Generally speaking, signatures are potentially highly descriptive, thanks to the use of spatially well-localized information, whereas histograms trade off descriptive power for robustness by compressing geometric structure into bins [6]. An additional category is represented by the synergistic deployment of both signatures and histograms (*hybrid* approaches). Two recent examples of this category are SHOT [6] and MeshHoG [10].

Application

Numerous applications require the analysis of similarities between 3D surfaces. Three well-established applications inherent to disciplines such as computer vision, computer graphics, and robotics are the following:

- *3D shape retrieval*, aiming at determining the most similar shapes included in a 3D database to a query shape [4]. The main challenges are the heterogeneity of the data typically included in the database, as well as its size, that poses efficiency constraint on the search algorithm. In addition, shape retrieval algorithms often need to be flexible enough to handle



Three Dimensional Shape Descriptor, Fig. 2 Exemplars of 3D shape descriptors applications. (a) The Princeton Shape Benchmark (PSB) dataset [3], commonly used for benchmarking 3D shape retrieval algorithms, exhibits a challenging intra-class variance at its *coarse2* taxonomy level. (b) Example of 3D object recognition in clutter and occlusion using local descriptors;

left: current scene, right: recognized model, red and green lines: correspondences yielded among keypoints by matching local descriptors. (c) 3D registration of two partial views performed via point-to-point correspondences obtained by matching local descriptors

the intra-class variations related to object categories (see Fig. 2a).

- *3D object recognition*, whose goal is to determine the presence of a set of object models within a scene. For those models recognized in the scene, often the 3D pose needs to be estimated, too, e.g., if the application concerns grasping or manipulation by means of a robotic agent.
- *3D reconstruction*, aiming at registering together two or more partially overlapping views of an object so as to reconstruct its shape.

Figure 2 shows some examples related to these three applications. As mentioned in the section of theory, local methods are particularly suited to deal with clutter and partial occlusions; hence, they are the general instruments employed within 3D object recognition scenarios. In such an application, the general approach relies on determining point-to-point correspondences between each model of the database and the current scene based on matching 3D features descriptors (see

Fig. 2b). The 3D model pose can then be estimated by selecting a subset of correspondences based on consensus and/or geometrical constraints. Within the object recognition scenario, global methods generally need, instead, to be paired with a segmentation algorithm, by computing a descriptor on each segment. Local descriptors are also commonly employed for 3D reconstruction from partial views, since they are suited to handle the presence of nonoverlapping parts of two surfaces that need to be registered together (see Fig. 2c).

On the other hand, global approaches are usually employed for 3D shape retrieval applications, where similar shapes to a query have to be found in 3D object databases, with the query shape possibly undergoing a high number of distortions [4]. Given the potentially high efficiency capabilities of global methods, they are also particularly suited to deal with large databases and/or real-time constraints, two typical aspects of 3D shape retrieval and 3D object recognition applications.

Open Problems

Open issues concerning 3D shape descriptors mainly deal with the ability of being invariant toward particularly challenging nuisances present in the data being processed. Two of the most challenging nuisances currently being addressed by research efforts are point density variations and nonrigid transformations (e.g., articulated deformations) of the 3D data (see, e.g., [2]). Another challenging issue is the computational burden concerning the use of 3D shape descriptors, since many applicative scenarios either require real-time processing (e.g., robotic applications) or have to deal with huge amounts of data (e.g., 3D shape retrieval databases). Finally, 3D shape descriptors are being used also to classify surface within a predefined set of categories. In this case, algorithms have to be flexible enough to handle the intra-class variance related to each object category.

References

1. Bronstein AM, Bronstein MM, Bustos B, Castellani U, Crisani M, Falcidieno B, Guibas LJ, Kokkinos I, Murino V, Ovsjanikov M, Patané G, Sipiran I, Spagnuolo M, Sun J (2010) Shrec 2010: robust feature detection and description benchmark. EUROGRAPHICS workshop on 3D object retrieval (3DOR), European Association for Computer Graphics, Norrköping
2. Ovsjanikov M, Bronstein A, Bronstein M, Guibas L (2009) Shape google: a computer vision approach to isometry invariant shape retrieval. In: Computer vision workshops (ICCV Workshops), 2009 IEEE 12th international conference on, IEEE, Piscataway, pp 320–327
3. Shilane P, Min P, Kazhdan M, Funkhouser T (2004) The princeton shape benchmark. In: Proceedings of shape modeling international, IEEE Computer Society, Los Alamitos
4. Akgül CB, Sankur B, Yemez Y, Schmitt F (2009) 3D model retrieval using probability density-based shape descriptors. *IEEE Trans Pattern Anal Mach Intell* 31(6): 1117–1133
5. Kazhdan M, Funkhouser T, Rusinkiewicz S (2003) Rotation invariant spherical harmonic representation of 3D shape descriptors. In: Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on geometry processing, Eurographics/ACM SIGGRAPH, Aachen, pp 156–164
6. Tomboli F, Salti S, Di Stefano L (2010) Unique signatures of histograms for local surface description. In: Proceedings of 11th European conference on computer vision (ECCV 10), Springer, Berlin
7. Johnson A, Hebert M (1999) Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(5): 433–449
8. Frome A, Huber D, Kolluri R, Bülow T, Malik J (2004) Recognizing objects in range data using regional point descriptors. In: Proceedings of 8th European conference on computer vision (ECCV 04), Springer, Berlin, pp 224–237
9. Chua CS, Jarvis R (1997) Point signatures: a new representation for 3D object recognition. *IJCV* 25(1): 63–85
10. Zaharescu A, Boyer E, Varanasi K (2009) Surface feature detection and description with applications to mesh matching. In: International conference on computer vision and pattern recognition (CVPR), IEEE, Miami, Florida, USA

Three Dimensional View Integration

Denis Laurendeau

Department of Electrical and Computer Engineering,
Laval University, Quebec City, Canada

Synonyms

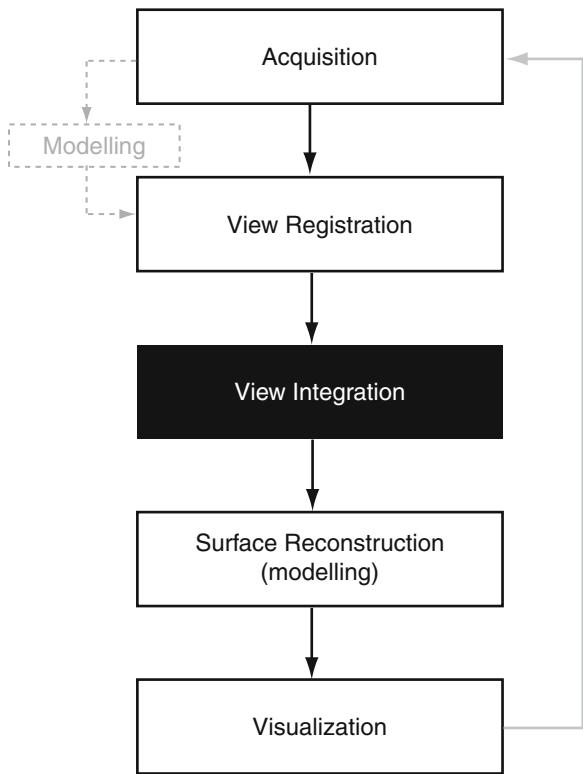
[Geometric fusion](#); [Surface reconstruction](#)

Definition

View integration consists of integrating several 3D views (images) in a non-redundant model, one that describes each part of the surface of an object or a scene with a single geometric primitive.

Background

The availability of accurate 3D cameras in the early 1980s has opened a new field of investigation in computer vision, namely, the acquisition and processing of 3D images [3]. The development of this field has been spectacular in the last three decades, and the use of 3D sensors, and the images they provide, is now common practice in a wide spectrum of human activities such as assembly, inspection, Computer-Aided Design (CAD), reverse engineering, medicine, and entertainment to name just a few.



Three Dimensional View Integration, Fig. 1 Processing steps of 3-D vision applications

In comparison with standard cameras which provide images of the appearance – either black and white or color – of the surface of objects, 3D cameras provide images containing information on the geometry of the surface of these objects. Some modern 3D sensors now even capture both appearance and geometric images of objects at the same time.

For the sake of brevity, in the following, the word “object” will be used to designate indifferently single objects or scenes made of more than one object. The problem of 3D view integration is concerned with the task of building the model of an object using three-dimensional data captured from several vantage points. The model should represent, in one form or another, the surface of the object from all directions of observation.

Equation 1 shows the steps usually adopted by 3D computer vision applications: acquisition, registration, integration, reconstruction, and visualization. A modeling step (shown in gray shade in the figure) is often executed before the registration step since registration and integration often use local models

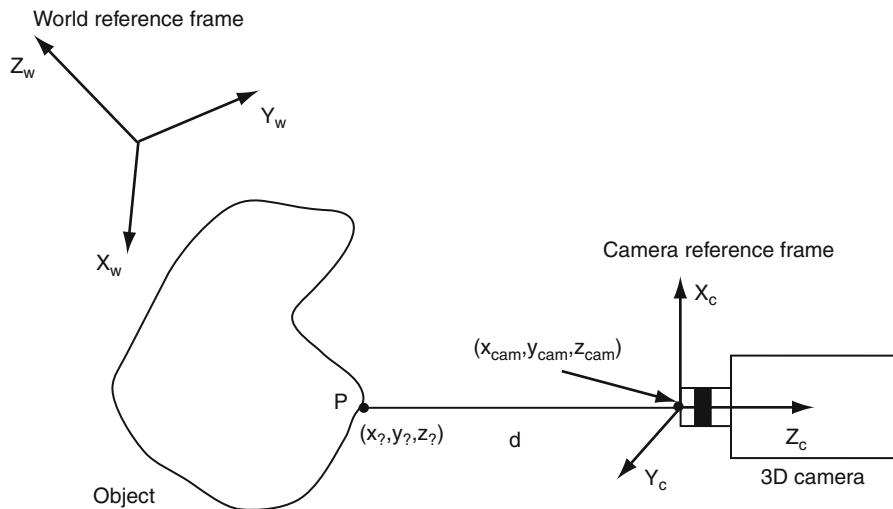
(e.g., triangulations) instead of raw data. In the case of interactive modeling applications, one can loop dynamically through these steps since the model of the object is progressively built as new data is acquired with the 3D sensor. Except maybe for visualization, all steps have a direct or indirect outcome on view integration.

In the following, the steps shown in Eq. 1 are discussed in the context of view integration.

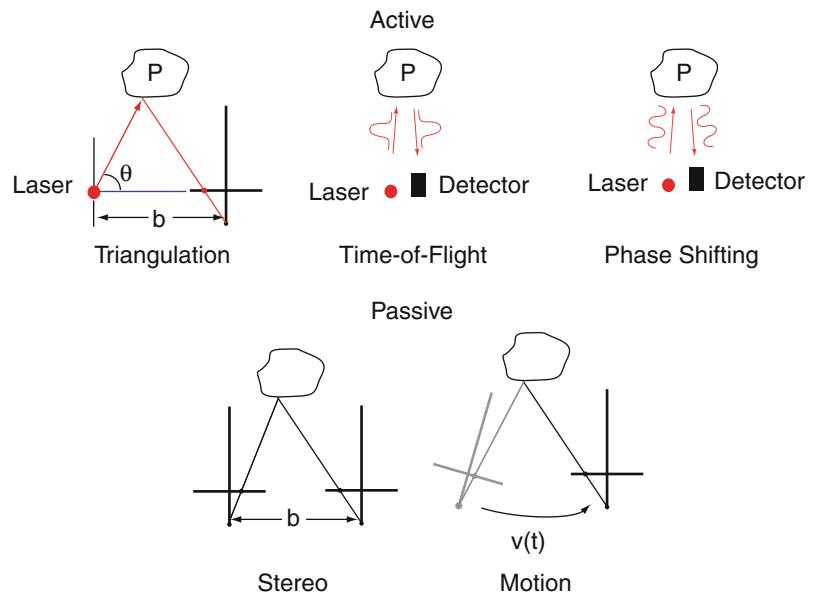
Theory and Application

3D Data Acquisition and View Integration

As mentioned above, 3D cameras acquire image data that conveys information on the surface geometry of objects. Figure 2 shows an object with a complex shape located in reference frame $X_w-Y_w-Z_w$ called the “World” reference frame. A 3D sensor, with its own reference frame $X_c-Y_c-Z_c$, observes this object from location $(x_{cam}, y_{cam}, z_{cam})$ in the “World” reference frame. For a given point P at the surface of the object, the 3D camera measures the coordinates $(x_?, y_?, z_?)$ of the point (some cameras compute the distance “d” between P and the origin of frame $X_c-Y_c-Z_c$). The reason for using “?” in the measured coordinates of P is that some cameras provide these coordinates in frame $X_c-Y_c-Z_c$ while others provide these coordinates in frame $X_w-Y_w-Z_w$ or some other frame in the “World” reference frame used for calibrating the camera. Several types of 3D cameras exist and use different approaches (active or passive) for measuring 3D coordinates of surface points and provide different types of raw data. Figure 3 shows some of these techniques (see [3] for a good coverage of range sensing technologies). On one hand, active techniques use some sort of light source, often a laser, for projecting a point or pattern (light stripe, grid) on the surface of the object. The pattern reflected by the surface is observed by a camera and, knowing (or calibrating) the baseline b (i.e., the distance between the source and the camera) and the orientation θ of the source, the coordinates of point P in the reference frame of the camera can be obtained by triangulation. Dense 3D images can be obtained by sweeping the pattern on the surface and capturing a large number of points. Another popular active approach computes the time of flight between the emission of a light pulse and its detection by a light sensitive sensor (an ultrasound pulse



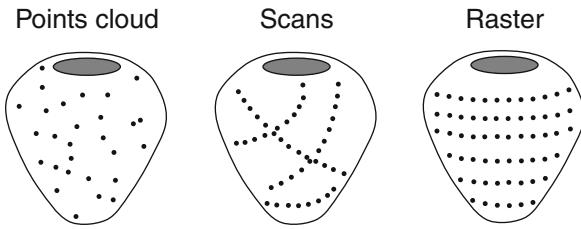
Three Dimensional View Integration, Fig. 2 Acquisition of a 3-D image



Three Dimensional View Integration, Fig. 3 Different types of range finders

and microphone are also used in cheap 3D cameras). The distance between the sensor and point P can be obtained from $d = 2t \times c$, where c is the speed of light. A similar approach, which uses a continuous wave (sine for example) instead of a pulse, computes the distance between the source and the sensor from the phase shift between the emitted and received signal. On the other hand, passive techniques do not use any external light source for measuring surface geometry. The most well-known passive approach is

stereovision which computes the coordinates of point P by triangulating the rays produced by its images captured by 2 (or more) cameras. A variant of stereovision uses only one camera that moves in front of the object and captures two (or more) images of it from different locations. Although it is not absolutely required that the scale of the object be known [7], the stereo pair of cameras is usually calibrated before image acquisition in order to provide accurate 3D coordinates at real scale.



Three Dimensional View Integration, Fig. 4 Different types of range images

Depending on the approach used, 3D cameras provide different types of raw data such as point clouds, scans, or rasters. As shown in Fig. 4, point clouds, also called “unorganized sets of points,” do not show any specific spatial organization and the connectivity between points is not available directly. Such images are often provided by techniques such as stereo vision. A second type of 2D image is shown in the middle diagram of Fig. 4 and consists of points organized as lines (the lines or scans need not be parallel). This type of image often results from the use of a light stripe scanned over the object during data acquisition by an active 3D camera (for instance, a handheld 3D scanner). The connectivity between neighboring points on a scan is generally available (in smooth areas of the surface), but the connectivity of points on different scans may not be available explicitly. Finally, a third type of 3D image, known as a “raster” and which generally provides dense 3D data, results from the acquisition of dense 3D images on a parametric grid for which the connectivity (4-neighbor or 8-neighbor) is directly available in the image. The availability or not of connectivity information as well as the density and accuracy of the 3D points in the images has a significant impact on view registration and view integration as discussed later.

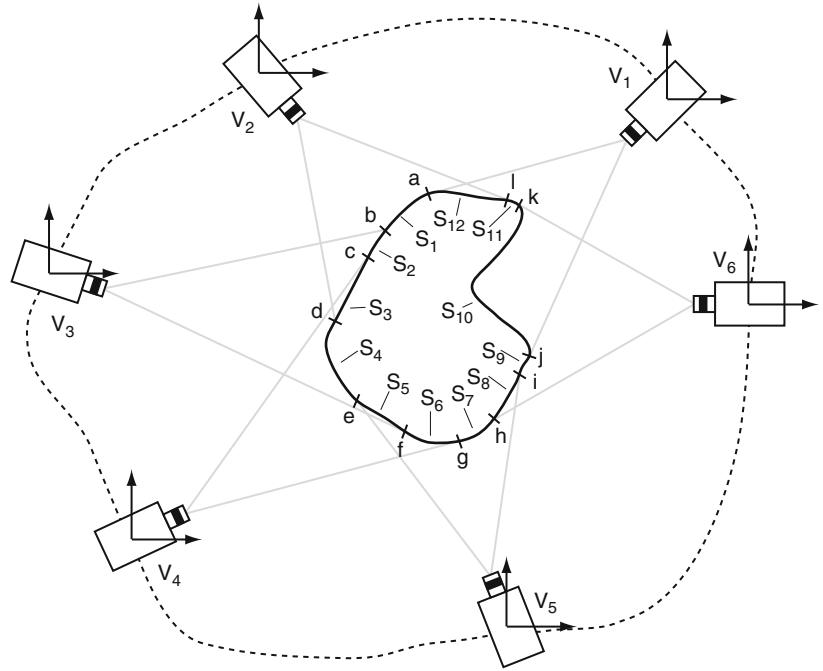
View Registration in the Context of View Integration

All 3D cameras, no matter what the approach used for measuring the 3D points, have a finite field of view, and, as shown in Fig. 5 for a 2D case, several images are thus required for gathering information on all parts of the surface of an object. The use of several views implies that (i) some parts of the surface of the object are observed in more than one view (segment

S_3 between c and d in Fig. 5 is visible in views V_2 , V_3 , and V_4); (ii) parts of the surface are observed at a grazing angle in some views (segment S_5 between e and f in V_3), meaning that the 3D data in this part of the view is sometimes not accurate; and (iii) concavities on the surface (such as segment S_{10} between k and j) are present and may require a large number of views if the entire surface is to be sampled. Acquiring multiple views implies that either the camera or the object must be moved during the acquisition process. In earlier days of 3D vision (and in many current applications), sensors were mostly static and objects were installed on a turntable while 3D images of the surface were captured. The advent of handheld 3D sensors [8] or sensors mounted at the end of a robotic arm now allows the object to remain static while the camera is moving during data acquisition. As mentioned above, the 3D information in each view is often expressed in a reference frame X_c - Y_c - Z_c attached to the camera. This means that the data acquired by the camera at each position (on its path around the object) is local to this position. Once the entire surface of the object has been covered, the data acquired in each local camera reference frame (V_i , $i = 1 \dots 6$ in Fig. 5) must be transformed in a common reference frame which can be the “World” reference frame, the reference frame of one of the views, for instance, V_1 , or any other frame that is relevant for the application. The task of finding the rigid transformation (translation and rotation) required to express the data in local reference frames in a common global reference frame is called “view registration.” View registration is a very complex problem that has received a lot of attention in the 3D vision research community, and several approaches for finding the rigid transformation between views have been proposed in the literature, the most well-known being the Iterative Closest Point (ICP) algorithm for pose refinement [2]. These approaches will not be discussed here but it suffices to say that all approaches that use the data alone (in opposition to methods that make use of an external positioning device) for computing the registration are based on the overlap between views (i.e., the data common to two or more views) to estimate the parameters of the rigid transformation between views by minimizing the distance between the overlapping regions. The computational complexity of the registration step, which needs to find the nearest neighbors of points on overlapping views in its distance minimization cost function, becomes a major

Three Dimensional View Integration, Fig. 5

Capturing capture of the entire geometry of an object requires multiple views acquired from different overlapping vantage points

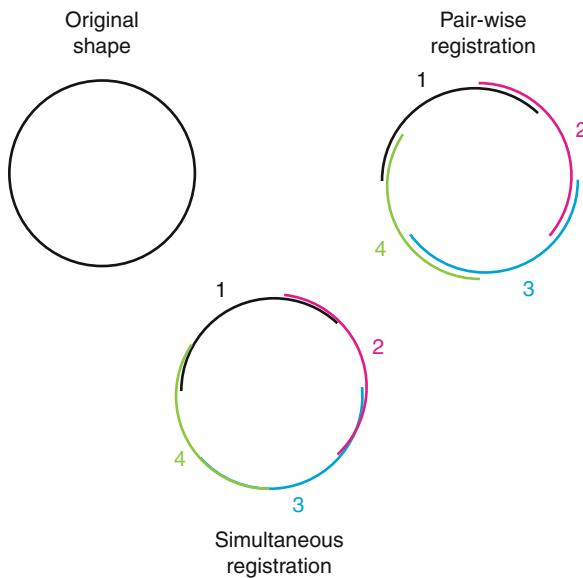


hurdle when the amount of 3D data is large, although some solutions have been proposed for achieving real-time performance on common-off-the-shelf computers for man-made objects of reasonable size [16]. The accuracy with which view registration can be achieved (meaning that the error in the estimation of the parameters of the rigid transformation is small) has a direct impact on the view integration step since the quality and conformity of the integrated data is closely related to registration errors. Three main approaches exist for view registration, two of which are illustrated in Fig. 6. Pair-wise registration estimates the rigid transformation two views at a time (for instance, 1–2 are registered first and yield 1/2, view 3 is then registered with 1/2 leading to 1/2/3 that is finally registered with view 4 leading to the final registered set of views 1/2/3/4). The main advantage of this approach is that it is both conceptually and computationally simple. Its main drawback is that registration errors accumulate in the sequential process, and the final result may often be far from the exact shape of the object as shown in the figure. A better approach, which is more computationally demanding, consists of registering all views simultaneously. This means that the parameters of all rigid transformations are estimated at once by minimizing the global registration error. This imposes

that all available data be processed simultaneously, which may lead to important memory requirements and long processing times when a large number of views containing huge amounts of 3D points need to be considered. The reward for the extra memory and computational cost is that registration accuracy is much better than for the pair-wise process because of global optimization. A last approach that is not illustrated in the figure and which is a compromise between the two other approaches, consists of registering a view with a computer model built from previously registered (and potentially integrated) views.

The need for registering several views in order to capture the geometry of the whole object raises the problem of choosing the next-best views that will bring the most relevant data and help in reducing the number of images required for surface coverage [1].

The accuracy of the 3D data has a significant impact on the view registration. As illustrated in Fig. 7, 3D measurements are obtained with different accuracy over different parts of a surface. For areas where the surface normal makes a large angle with respect to the direction of observation of the camera (point P₂ with normal n₂), measurements are less accurate than for the parts where the surface normal and direction of observation are mostly parallel (point P₁ with normal n₁).



Three Dimensional View Integration, Fig. 6 Different view registration approaches: pair wise (top right), simultaneous (bottom)

For active systems using a laser, for instance, a smaller light energy reaches the sensor which causes measurement to be less accurate at P_2 . The measurement error is often modeled as a Gaussian distribution with its main axis aligned with the direction of measurement as shown on the right hand side of the figure. The Gaussian model is far from being ideal since it fails to describe the “impulsive” nature of the acquisition noise in areas such as P_2 , but it is a convenient approach for taking measurement errors into account in the registration process. Some measurements which cannot be modeled by a Gaussian error model are completely unreliable and are labeled as “outliers.” Such outliers have a dramatic effect on the minimization process in view registration and must be eliminated (RANSAC-based approaches can help in the elimination process [6]). Outliers also occur at step discontinuities (parts of the surface where there is a large variation of range over a small patch of the surface).

3D View Integration

With the above material on 3D data acquisition and registration, the problem of 3D view integration can now be approached as such. In the following, it is assumed that 3D data has been acquired in enough views to cover the whole surface of an object and

that view registration has been performed with enough accuracy to allow reliable integration. For the sake of simplicity, the focus will be put on the integration of 3D raster images acquired by active range finders. Raster images will be called “range views” or “range images.” Integration of color information often acquired simultaneously with range information will not be addressed since it is a research problem of its own.

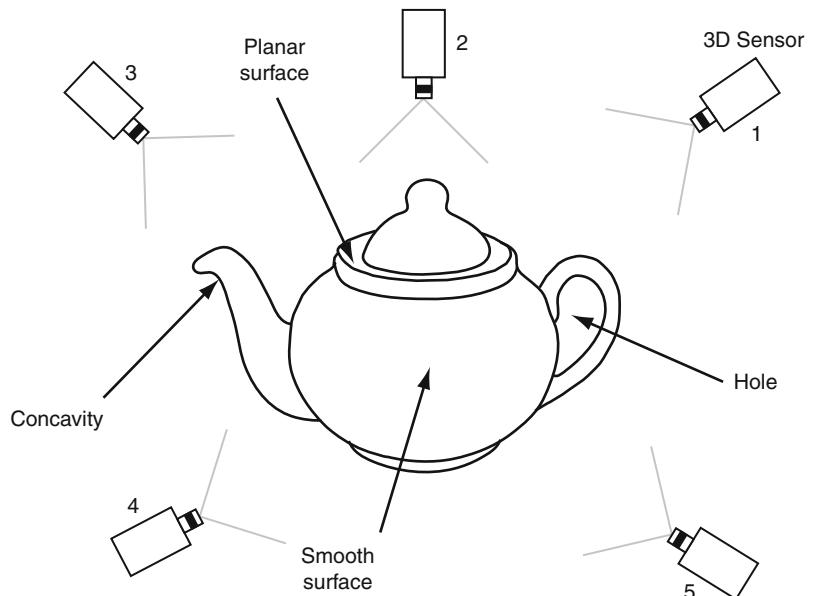
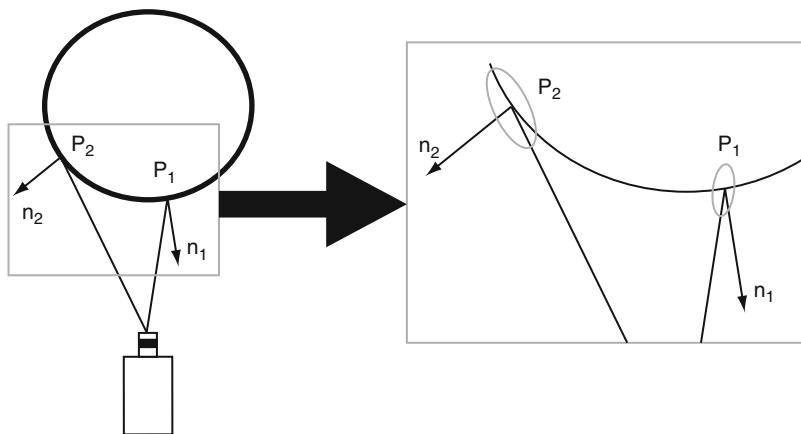
Ideally, a view integration approach should meet the following requirements [4] (see Fig. 8):

1. Be able to deal with multi-part objects with holes and concavities such as the teapot shown in the figure. For instance, parts of an object may occlude another part in a given view creating step discontinuities in the range image (the handle of the teapot occludes the body in view 1).
2. Be independent of the number of views and be able to cope with the large number of views (and 3-D points) needed to capture the full geometry of an object.
3. Be able to perform incremental reconstruction meaning that new views can be added to an existing model.
4. Be “order independent” meaning that the resulting integrated model should not depend on the order with which the views were integrated.
5. Be able to take data accuracy into consideration in the integration process and be robust to outliers.
6. Be able to take into consideration the connectivity of the data when it is available as well as the redundancy of the data (i.e., data acquired on the same area of the object’s surface in different views).
7. Be able to fill gaps in the integrated model caused by lack of 3D data on small parts of the surface.
8. Be computationally efficient with respect to CPU and memory requirements. Although real-time view integration performance is not always requested, processing a large number of views must be executed in a reasonable amount of time on COTS computers. Ideally, complexity should be linear with respect to the number of points (and triangles when a local triangulation model is used for describing each view) as well as the number of views.

At this point, it is important to mention that the process of view integration is also called “surface reconstruction” or “geometric fusion” in the relevant literature. All terms will be used in the following.

Three Dimensional View Integration, Fig. 7

Illustration of the acquisition error on range points. The error is usually larger when the camera observes the surface at a grazing angle

**Three Dimensional View Integration, Fig. 8**

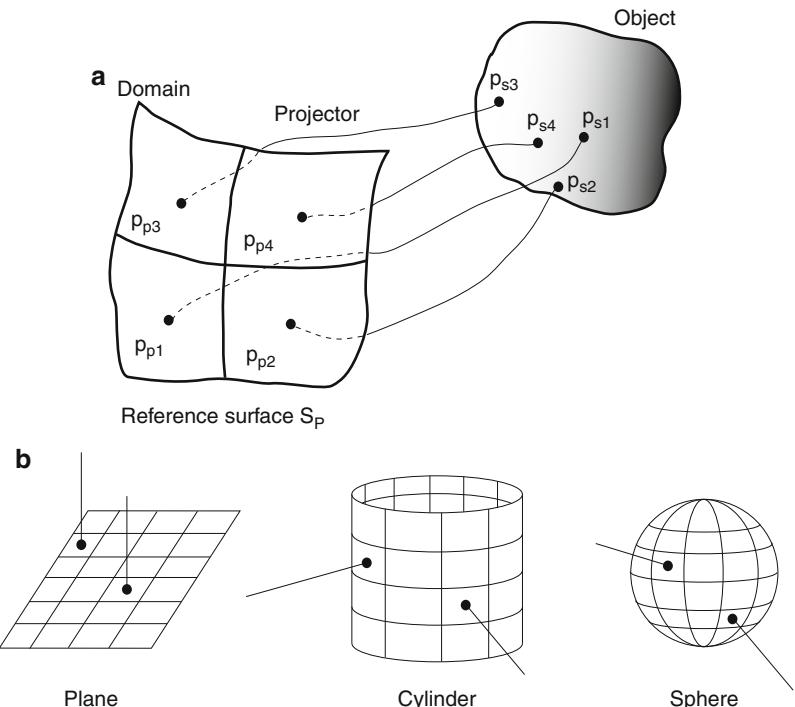
Acquisition of the geometry of a complex object with concavities, self-occlusions in some views, and holes

Before going further, let us define more formally what is meant by the term “range image.” This definition is proposed in [14]. A range image is a function which associates domain points to image points. Domain points p_p are sampled on a reference surface S_p while image points p_s are located on the surface of 3D objects in a scene. The range image function results from the projection of domain points p_p onto object surfaces. The projection can be directed along curved lines called projectors as shown in Fig. 9a. Objects are assumed to be solids of finite volume bounded by a manifold, orientable, and closed surface. Imposing that the surface be closed implies that view integration must be able to fill holes resulting

from an imperfect integration process, a feature that is in line with requirement section “Open Problems” above.

There are basically two main approaches to view integration: surface based and volume based. Both approaches share the property that they can generally generate an integrated surface model at several resolutions. The generated model is usually a linear approximation of the integrated surface. This means that the integrated model approximates the surface locally by small planar patches such as triangles organized in a connected mesh, and each area of the surface is described by one and only one model. More sophisticated representations such as bicubic splines,

Three Dimensional View Integration, Fig. 9 General definition of a range image (a). Planar, cylindrical, and spherical reference surfaces with straight line projectors (b)



superellipsoids, or generalized cylinders can also be used as a model of the integrated surface.

Both approaches also make the assumption that the resolution of the range images used in the integration process is high enough to deal with objects showing large curvature in some areas of their surface otherwise, too small a resolution would cause curved areas to appear as step discontinuities.

Surface-based approaches are addressed first followed by volume-based approaches.

Surface-Based View Integration

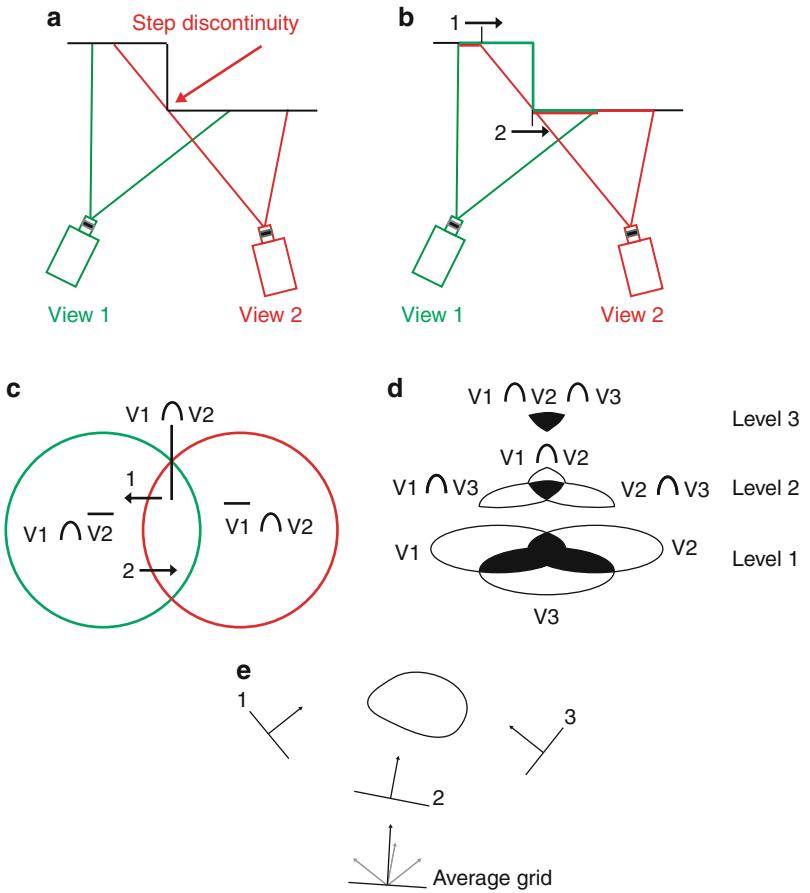
The definition of a range image given above is operational and quite general since no specific shape is specified either for the reference surface or the projectors. In practice, the reference surface can be a plane, a cylinder, or a sphere, and the projectors are straight lines as shown in Fig. 9b. Although cylindrical and spherical reference surfaces can prove useful in some situations, they also have significant drawbacks. For instance, the elements of a spherical grid do not have the same area. In addition, projectors may intersect for both the cylindrical and spherical grids. For a planar reference surface, the elements of the grid can be made square, and the projectors are parallel with each

other and perpendicular to the reference surface. Planar reference surfaces are thus preferred over cylindrical or spherical grids for obvious reasons. The advantage of using a reference surface is that connectivity information between image data is directly available from the connectivity between grid elements except at step discontinuities.

The first step in view registration consists of eliminating outlier data from the range image. Outliers frequently appear near step discontinuities or occlusion contours when laser scanners attempt to acquire samples where the laser impinges the object surface at grazing angles. Outliers appear as if they were “floating” in space and are often very far from other points. A simple thresholding operation applied on the range image is generally all that is needed to eliminate outliers.

A second processing operation that is frequently performed on range images is the detection of step discontinuities in the different views. An example of such a step discontinuity is shown in Fig. 10a. The scene shows a surface orientation discontinuity in View 1 but this discontinuity is seen as a step discontinuity in View 2. Many approaches can be used for detecting step discontinuities, the simplest one being

Three Dimensional View Integration, Fig. 10 Step discontinuity in a view (view 2) (a). Crossing a step discontinuity in a view (b). Membership discontinuity in the Venn diagram (c). Hierarchy of redundant subsets of the Venn diagram (d). Average reference surface (grid) for the integration of views (d)



to implement a thresholding operation on the 3D data corresponding to horizontal and vertical neighbors on the grid of the reference surface. Although it is a simple operation from the conceptual standpoint, it is not so easy to implement a robust step discontinuity algorithm since range data is sometimes noisy near the steps, and false alarms and non-detections can occur and may cause problems for some integration algorithms [14].

The view integration task begins as such once the three pre-processing steps described above have been performed for all views. The first goal is to detect the redundant information in the views, i.e., the parts of the scene that are observed in more than one view. Redundant information is a key element for view registration but once registration is performed, redundancy in the data must be eliminated in order to produce the model.

Set theory has been proposed as a framework for addressing the view integration problem [14]. If a view

is seen as a set of points as in Fig. 10c, then, finding redundant information reduces to finding subsets of the Venn diagram which are the intersection between sets, in this case $V_1 \cap V_2$. Figure 10b shows the parts in the scene which are visible in V_1 (green), V_2 (red), both V_1 and V_2 (green and red), or not visible in any view (black) for the simple scene in Fig. 10a. It is important to note that there is a link between the presence of a step discontinuity in a range image and a membership discontinuity in the Venn diagram. In the figure, when one crosses the step discontinuity in View 1 reported in the reference frame of View 2 (marked as “1” on the figure), one crosses the frontier between subset $V_1 \cap V_2$ and subset $V_1 \cap \bar{V}_2$. The same situation occurs when one crosses the step discontinuity in View 2 (marked 2 on the figure) since one crosses the frontier between subset $V_1 \cap \bar{V}_2$ and $V_1 \cap \bar{V}_2$. By processing pairs of views in turn, one ends up building the Venn diagram of N views and thus finding the redundant information corresponding to subsets of

the Venn diagram. The information contained in only one view can thus be modeled while the information contained in more than one view can be merged and then reconstructed as a single model. This procedure is described next.

Finding the information common to a pair of range views consists of taking each 3D point in one view and finding whether or not, when transferred in the reference frame of the other view using the frame transformation obtained from view registration, it is close to a triangle in the mesh of the first view. This first operation is called a “neighborhood test.” When the 3D point is indeed close to a triangle in the second view, a second test is run to check whether or not this point is “visible” from the planar parametric grid of the second view. This “visibility test” is required since a point may be close to the surface of a view but may not be visible from this view. This happens for instance near the rim of a cup where points in one view may have been sampled on the interior of the cup, while points in the other view have been sampled on the exterior of the cup. Such points are close in space, but they do not belong to the same part of the object. The “visibility test” checks if the scalar product between the normal to the surface at the point being processed and the direction of the optical axis of the camera is positive (i.e., the angle between the two directions is smaller than 90°). If it is so and the point is close to the surface, then this point is labeled as being common to the two views.

The two tests described above yield poor results near step discontinuities because it is (i) difficult to define a surface patch near a discontinuity which causes the spatial neighborhood test to fail and it is (ii) also difficult to estimate the normal to the surface near a discontinuity, causing the surface visibility test to fail.

This is where step discontinuities come into play. Steps discontinuities detected in one view are reported in the second view (again using the frame transformation obtained from the view registration process). The points labeled as common to two views and forming a connected component on the parametric grid of this view are used as seed regions for a region-growing process which expands the regions until they reach a contour of step discontinuity. Knowing that a step discontinuity marks a membership discontinuity in the subsets of the Venn diagram, when the expanded region makes contact with a step

discontinuity, the region growing stops, and the points which were merged to the regions are labeled as being common to both views.

The above procedure described how data common to two views can be detected. However, complex objects require much more views if their geometry is to be captured with details. Consider the hierarchy of subsets of the Venn diagram shown in Fig. 10d. These subsets are called redundant subsets. The integration procedure begins at the top of the hierarchy and finds a description of the surface for points belonging to $V_1 \cap V_2 \cap V_3$. The strategy consists of defining a new parametric grid whose orientation is the average of the orientation of the grids of the views participating in the subset as shown in Fig. 10e. Each range view is modeled by a step discontinuity constrained Delaunay triangulation. A Delaunay triangulation ensures that triangles in the model are as equiangular as possible. Stopping the triangulation at step discontinuities ensures that triangles do not overlap a discontinuity and do not model two parts of the surface separated by a large step [14, 17]. The triangulation of each participating view is projected on the new parametric grid and is re-sampled on this grid. This reparameterization can be seen as acquiring a new image from the vantage point of the average parametric grid. Now, since several views participate in a subset, each view can contribute a range value to the “synthesized” image of this grid. This means that these range values can be merged in order to take advantage of the redundancy brought by the multiple views. The resulting range value of the average grid is a weighted average of the range value in each view. The weight for each contributing range value is obtained from the square of the cosine between the direction of the sensor and the surface normal at that point. This ensures that a range value acquired at normal incidence in a view will contribute more to the synthesized view than one obtained at a grazing angle in another view. Finally, once the synthesized image has been generated, a Delaunay triangulation is built to describe the subset at a given level of the hierarchy.

Then, levels lower in the hierarchy are processed similarly. Processing at all levels of the hierarchy means that several Delaunay triangulations exist for the same part of the surface and redundancy must be eliminated. This redundancy is useful though because it helps in eliminating errors near membership

discontinuities at the higher level of the hierarchy (remember that it is difficult to build a model near these discontinuities). Redundancy between triangulations is eliminated by finding the overlap between them starting from the top of the hierarchy to the bottom.

Once the overlap has been eliminated, each canonical subset of the Venn diagram is modeled by a single triangulation. The next step consists in stitching the triangulations in order to fill small gaps between them.

In the procedure described above, data fusion between range values coming from different views and elimination of the overlap between redundant models of the same part of the surface are operations that are performed before stitching the resulting triangulations. Due to errors in view registration, the stitching may cause small ridges to appear in the integrated model. An improvement of the procedure consists of eliminating the overlap and stitching triangulations first and then to improve the final geometry by finding a consensus position of the triangle vertices based on the contributions of all participating views (again, a weighted average is used). Since it is better to avoid lateral movement of the vertices in the triangulation, the consensus position is computed for the direction along the average surface normal at each vertex [17] or by re-tessellating the mesh using redundant data [13].

Volume-Based View Integration

Signed Distance Fields

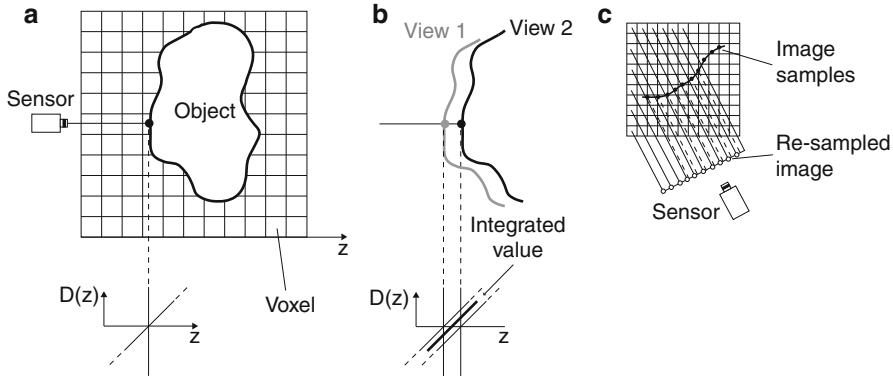
Volume-based view integration approaches make use of the fact that objects occupy a volume in 3D space. Since range images provide samples of the frontier between the “outside” and the “inside” of the object, combining registered volume information implicitly contained in range images is a way of integrating the different views.

The world of robotics has made use of volume-based representations of surfaces through the concept of probabilistic “occupancy grids” [5]. An occupancy grid is one that encodes the probability of the presence of an object in each cell of the grid. The evidence of the presence or absence of an object is derived from the probabilistic model of the range sensor, the range measurements provided by the sensor, and Bayesian updating of the cell’s occupancy probability based on

repeated measurements. Occupancy grids have been used mostly for path planning and collision avoidance in robotics but have not generally been used for surface reconstruction because of the difficulty of transforming the probabilistic model stored in the grid into an accurate surface description.

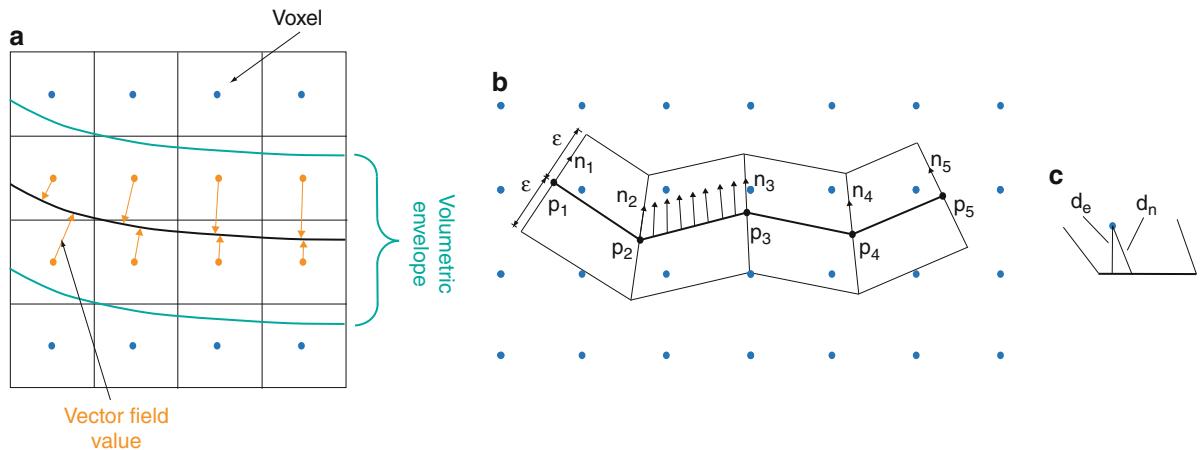
The most general volumetric approaches for view integration make use of a continuous objective function to combine range data [4, 11]. The range data points in the images are considered as samples of this objective function. A very popular objective function that is used by volume-based view integration approaches is the signed distance function $D(z)$. Function $D(z)$ is an implicit representation of the surface, and, as discussed in the following, a triangular mesh built from the zero-crossings of this function is the explicit representation of the surface.

Let us assume that the volume in which the object is located is represented as a grid of cubic voxels as shown in Fig. 11a. When a range image is acquired, it is first modeled as a triangular mesh. The mesh must not overlap step discontinuities for the same reasons as those mentioned above for surface-based approaches. Each vertex in the mesh is assigned a weight according to a weighing function that takes into account the accuracy of the measurement. Several approaches have been proposed for modeling this accuracy [9]. The signed distance between the weighted mesh and each voxel center in the direction of the ray between the sensor and the surface is computed and stored in the voxel grid structure. When a second view is acquired, the signed distance function is computed once again as shown in Fig. 11b, and the two views are integrated by computing the weighted distance resulting from the previous view and the current view. More formally, the surface is located at the zero-crossing of the weighted objective function $D(z)$ at the voxel. This approach is obviously order independent and is also independent of the number of views. It takes data accuracy into account since the signed distance function and the weighing function can be computed separately as images are acquired and the zero-crossing of the cumulative weighted signed distance function can be computed last. When pushed forward, the approach also allows “hole filling” to be implemented in a straightforward manner. As a matter of fact, when casting a ray between the sensor and the surface, the ray can be cast back toward the sensor, and voxels crossed by the ray are assigned the label



Three Dimensional View Integration, Fig. 11 Illustration of the signed distance between a surface and the centers of the cells of a volumetric grid (a). Merging of the distance field for two

views (b). Re-sampling of the range images for speeding up the integration process (c)



Three Dimensional View Integration, Fig. 12 Basic principle of the vector field representation (a). Closest point along the interpolated normals (b). Difference between the Euclidean distance and the distance along the interpolated normal (c)

“empty” (with $D(z) = D_{\min}$ and weight zero). Voxels on the surface are labeled as “near” the surface, and the remaining voxels are assigned the “unseen” label (with $D(z) = D_{\max}$ and weight zero). “Hole filling” is performed implicitly by finding the zero-crossing of the resulting distance function [4]. Final surface extraction of the integrated model is obtained with the well-known “Marching Cubes” algorithm [10] adapted to the above procedure [12].

Despite the conceptual simplicity of using distance fields, practical implementations face some challenges. The first challenge is of course the computer memory requirements for storing the signed distance field. A coarse resolution helps in reducing memory space and

processing time but the resulting model is not faithful to the object surface geometry. A fine resolution for the volumetric grid yields a high-quality surface model, but computational costs explode. Encoding the volumetric grids in clever data structures (run-length encoding [4], kd-trees [11]) has been proposed to reduce both memory space and computer time requirements.

A second issue that must be considered is the distance to which the weighing function should extend on both sides of the surface. Ideally, the function could extend indefinitely, but, as it is the case for surface-based approaches, this would cause integration to fail for thin parts of the object’s surface such as the rim of a

cup since functions on the sides of both surfaces would interfere when computing the resulting weighted distance function for a voxel. Choosing the extension of the weighing function to be half the maximum uncertainty interval of the range measurements is usually a good rule of thumb since the generated model cannot be more accurate than the measurements themselves. Other error models can also be considered as well [9, 13].

Another important algorithmic issue is the updating of the content of the volume with a new range image since it rarely happens that both are aligned (as a matter of fact, the orientation of the volume in 3D space can be chosen arbitrarily as far as it circumscribes the volume of the object). Ray casting between the sensor and a voxel for finding the intersection is very costly in terms of computing time and is tractable only for volumes with coarse resolution. Consequently, range images can be re-sampled along the volumetric grid prior to volume updating. This re-sampling can be compared with the computation of a reparameterized surface grid mentioned for the surface-based approaches described above. Range image re-sampling is shown in Fig. 11c.

Vector Fields

Another powerful representation has been proposed for describing surfaces and for performing surface reconstruction from a set of range images: the “vector field” [16]. The “vector field” representation also uses a volume for integrating range views and for extracting a unique surface reconstruction from the marching cubes algorithm. However, the entry in each voxel is the distance between the voxel center and the closest point on the surface (the definition of “closest point” will be given below) as well as the direction of the vector toward the closest point (see Fig. 12a for a 2D cut of the volume). For each surface, the field is computed only for voxels located inside a region of space called the “volumetric envelope” which is defined by the iso-surfaces located at a positive distance ε on both sides of the surface. The volumetric envelope plays the same role as the limited extent for the weighing function of the approach using the distance function. It allows the vector field approach to cope with registration errors between views and to limit the computational cost of computing the field. Several constraints guide the

choice of the value for ε : size of the voxel, error of the registration between views, as well as other geometric considerations [15]. Although vector fields can be computed for point clouds, scans, and range images, only the case of range images will be covered in the following.

More formally, assuming a set $A = \{a_i \in R^3, i = 1, \dots, N\}$ of N 3D points in a range image, the “vector field” representation of set A is the field $F : VCR^3 \rightarrow R^3$ such that $F(v) = \arg \min_{a \in A} d(a, v) - v$. Where $V = \{V_{i,j,k}\}$ is a set of $N_x N_y N_z$ voxel centers $V_{i,j,k}$ and $d(a, v)$ is some distance measure between a and v . The choice of the distance measure is key to the concept of vector field.

Figure 12b shows a 2D view of how the vector field is computed for a range image for which a triangular mesh has been constructed beforehand. First, the normal to the surface of each triangle in the mesh is computed and the normal at a vertex is computed as the average vector of the normals of all triangles containing this vertex. The direction of the normals should be consistent with the position of the sensor (i.e., reside on the same side of the surface). Then, the normal to each point on the triangle is obtained by interpolating average normals at the vertices, each normal being weighted by the barycentric coordinates of the point on the triangle. For a lattice point of the volumetric grid, the distance stored in the corresponding voxel is d_n , the distance along the interpolated normal (see Fig. 12d) instead of the classical Euclidean distance d_e . In addition, the direction of the interpolated normal is also stored in the voxel.

Integrating N range views simply consists of computing the normalized vector field resulting from the combination of all contributing images. The normalized vector field is one for which each vector is divided by its norm. This can be expressed by the following equations:

$$F_{int}(p) = \left[\sum_{i=1}^N F_i(p) w_i(p) \right] / \sum_{i=1}^N \omega_i(p)$$

and

$$d_{int}(p) = \left[\sum_{i=1}^N d_i(p) \omega_i(p) \right] / \sum_{i=1}^N \omega_i(p).$$

The weight ω represents the confidence in the data and is given by the cosine of the angle between the

direction of the sensor and the surface normal. The weights ω_i are also interpolated using the barycentric coordinates. The final surface model (i.e., the reconstructed surface) is obtained by the marching cubes algorithm.

Clearly, the surface reconstruction approach using vector fields is independent of the number of views. It makes use of connectivity information in a view since the computation of normals at vertices uses this connectivity. It is order independent since the field for each view is computed separately and fields of all views are combined. The approach is also incremental in that it allows a view to be integrated with an existing field. Sensor noise is taken into account in the integration process as it is the case for all view integration approaches presented so far. However, the vector field requires more storage space than the classical signed distance field described above so what is the gain of storing direction information in the volumetric model in addition to distance information? Could the vector field be obtained from the signed distance field? In a volumetric lattice of infinite resolution (i.e., voxel size = 0), the vector field could be obtained by computing the gradient of the distance field which provides the direction of the surface normal everywhere in the lattice. However, this does not hold for a finite resolution lattice since the gradient of the signed distance field must be estimated by finite differences, an approach that is not accurate enough to yield useful direction information. Another gain offered by vector fields over signed distance fields is when the final surface model is obtained by the marching cubes algorithm [15].

But the major gain of using the vector field over signed distance fields is that this representation can be used for all steps of the surface reconstruction process (registration, integration, reconstruction, and visualization) with linear complexity with respect to the number of points making real-time surface reconstruction possible. The reason for this is that storing distance to and direction toward the surface in the field results in solving the nearest neighbor problem implicitly, something that is not the case with the signed distance field (with the same resolution). Using the same representation – the vector field – for all steps of the surface reconstruction process avoids conversion between representations, an operation that is sometimes computationally expensive and numerically inaccurate.

Open Problems

The surface reconstruction approaches presented above, both surface based and volume based, allow a significant number of range images to be integrated in a non-redundant surface representation. The main issue common to all approaches is still the ability to cope with huge amounts of data (i.e., several millions of points) in a reasonable amount of time and with limited memory space. The increasing power of modern computers and the low cost of memory help in allowing more and more views to be integrated in a single surface representation. More research is required on approaches for compressing the representations without too much increase in computational complexity while still minimizing the error in the compressed model. Compression approaches suitable for transmission over the Internet would be a significant asset.

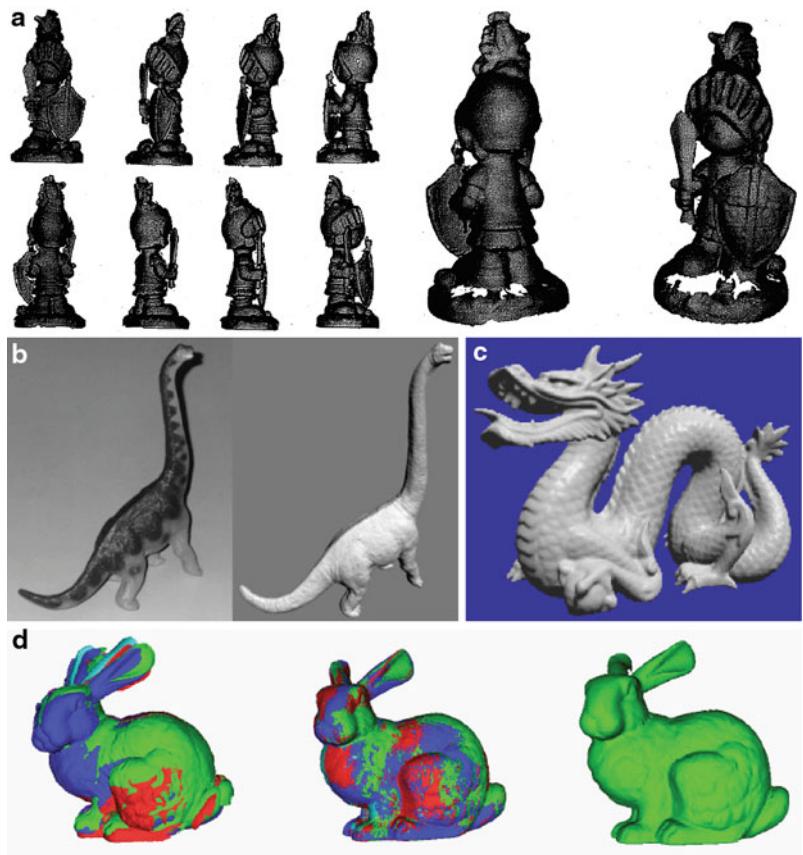
Another important problem to be addressed that is closely related to view integration is the accuracy with which view registration can be achieved. All view integration approaches described above make the assumption that view registration is accurate. However, although accurate view registration approaches are available for finding the rigid transformation between views before integration, most of these approaches are pose refinement techniques, meaning that they can estimate the parameters of the rigid transformation between views that are already close to each other. Initial pose estimation (i.e., estimation of the rigid transformation between views when they are far apart) is a much more complex problem, and further research is needed on this topic.

One of the goals of view registration is to produce reliable and accurate geometric models of complex objects. Including appearance information (i.e., color) in the models is an interesting problem which has received attention, but research still needs to be conducted in this direction. For instance, one may ask the following question: Can the data structures used for surface-based or volume-based view integration be extended to include color information? Can these enhanced data structures be compressed efficiently?

Experimental Results

[Figure 13](#) shows the results obtained with four view integration approaches. [Figure 13a](#) shows eight

Three Dimensional View Integration, Fig. 13 Surface reconstruction with surface-based and volume-based approaches. Final polygonal mesh of a soldier figurine produced by the surface-based approach exploiting the Venn diagram formalism (Taken from [14]) (a). Final polygonal mesh of a dinosaur figurine using the “zippered polygon” meshes surface-based approach (Taken from [17]) (b). Final polygonal mesh obtained by the volume-based approach based on the signed distance field formalism (Taken from [4]) (c). Final polygonal mesh of a duck generated from the volume-based approach exploiting the vector field formalism. View registration is also achieved with the same vector field model (Taken from [15]) (d)



256 × 256 range images of a toy soldier (left) with two views of the resulting model made of 70,842 triangles provided by the surface-based approach in [14]. Figure 13b presents the polygon mesh of a plastic dinosaur model [17]. The mesh contains 360,000 polygons.

Figure 13c shows the model of a dragon built from the volume-based approach using a signed distance field [4]. This model, made of 1.7 million triangles, was produced from the integration of 61 scans totalizing 15 million triangles.

Finally, Fig. 13d presents the model of a duck built from 12 range images and using the vector field representation [15]. The lattice is made of 128 × 128 × 128 voxels with a volumetric envelope of 3 voxels. The left image shows a superimposition of the original range images. The center image shows a superimposition of the same range images after registration using the vector field approach. The right image shows the reconstructed surface built from the

integration of the vector fields built with the original images.

References

1. Arbel T, Ferrie FP (2001) Entropy-based gaze planning. *Image Vis Comput* 19(11):779–786
2. Besl P, McKay N (1992) A method for registration of 3-D shapes. *IEEE Trans Pattern Anal Mach Intell* 14(2):239–256
3. Blais F (2004) Review of 20 years of range sensor development. *J Electron Imaging* 13(1):231–224
4. Curless B, Levoy M (1996) A volumetric method for building complex models from range images. In: SIGGRAPH 96; proceedings of the 23rd annual conference on computer graphics and interactive techniques, New Orleans, pp 303–312
5. Elfes A (1989) Using occupancy grids for mobile robot perception and navigation. *Computer* 22(6):46–57
6. Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun ACM* 24:381–395

7. Hartley R, Zisserman A (2000) Multiple view geometry in computer vision. Cambridge University Press. ISBN:0521623049
8. Hébert P (2001) A self-referenced hand-held range sensor. In: Proceedings of the IEEE international conference on recent advances in 3-D digital imaging and modeling, Québec, pp 5–12
9. Hébert P, Laurendeau D, Poussart D (1994) Scene reconstruction and description: geometric primitive extraction from multiple view scattered data. In: IEEE conference on computer vision pattern recognition (CVPR 93), New York, pp 286–292
10. Lorensen WE, Kline HE (1987) Marching cubes: a high-resolution 3D surface construction algorithm. In: Proceedings of the SIGGRAPH'87, Anaheim, pp 163–169
11. Masuda T (2003) Registration and integration of multiple range images by matching signed distance fields for object shape modeling. Comput Vis Image Underst 87: 51–65
12. Montani C, Scateni R, Scopigno R (1994) A modified look-up table for implicit disambiguation of marching cubes. Vis Comput 10(6):353–355
13. Rutishauser M, Stricker M, Tröbina M (1994) Merging range images of arbitrarily shaped objects. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR 94), Seattle, pp 573–580
14. Soucy M, Laurendeau D (1995) A general surface approach to the integration of a set of range views. IEEE Trans PAMI 17(4):344–358
15. Tubic D (2006) On surface representation in 3D modelling: a framework for interactive 3D real-time modeling. Ph.D. thesis, Laval University, p 178
16. Tubic D, Hébert P, Laurendeau D (2002) A volumetric approach for the registration and integration of range images: towards interactive modeling systems. In: Proceedings of the ICPR 2002, Quebec, pp 283–286
17. Turk G, Levoy M (1994) Zippered polygon meshes from range images. In: Proceedings of the SIGGRAPH 94, Orlando, pp 311–318

Total Variation

Bastian Goldluecke

Department of Computer Science, Technische Universität, München, München, Germany

Definition

The total variation (TV) is a nonnegative, convex, and lower semicontinuous functional on the space of integrable functions. For a function $u \in \mathcal{L}_{\text{loc}}^1(\Omega)$ on a domain $\Omega \subset \mathbb{R}^n$, it is defined as

$$J(u) := \sup \left\{ - \int_{\Omega} u \cdot \operatorname{div}(\xi) \, dx : \xi \in \mathcal{C}_c^{\infty}(\Omega, \mathbb{R}^n), \|\xi\|_{\infty} \leq 1 \right\}. \quad (1)$$

For differentiable functions $u \in \mathcal{C}^1(\Omega)$, this definition can be reduced to the familiar expression

$$J(u) = \int_{\Omega} |\nabla u|_2 \, dx \quad (2)$$

with the help of Gauss' integral theorem. A function with $J(u) < \infty$ is called of *bounded variation*; the space of all functions on Ω with bounded variation is denoted by $\mathcal{BV}(\Omega)$.

Background

The total variation is a favorite prior term and regularizer in variational models for image processing. Since it is a convex functional, it allows to formulate convex variational models which can be efficiently globally minimized. Furthermore, while it favors smoother functions, abrupt discontinuities are not penalized stronger than slow transitions, and discontinuous solutions are possible. For this reason, the total variation can be viewed as an edge-preserving regularizer.

Since the length of the boundary of a set can be expressed as the total variation of its characteristic function, total variation can also successfully be employed to formulate variational segmentation problems. In particular, it can be used to express weighted minimal surface problems, which leads to further applications in, for example, 3D reconstruction.

Theory

Total variation and convex analysis In computer vision applications, J is usually employed as a regularizer for a *variational model* of the form

$$\operatorname{argmin}_{u \in \mathcal{V}} J(u) + F(u), \quad (3)$$

where F is a data fidelity term which measures how well u fits the given data, while \mathcal{V} is chosen as a subspace of the Hilbert space $\mathcal{L}^2(\Omega)$. The Hilbert space

setting is necessary to apply optimization methods from *convex analysis* and certainly general enough for most practical applications. Any minimizer is necessarily of bounded variation. On $\mathcal{L}^2(\Omega)$, the total variation can be rewritten as the support functional of a convex set K ,

$$J(u) = \sup_{v \in K} (u, v) \text{ with } K = \{\operatorname{div}(\xi) : \xi \in \mathcal{C}_c^\infty(\Omega, \mathbb{R}^n), \|\xi\|_\infty \leq 1\} \subset \mathcal{L}^2(\Omega), \quad (4)$$

as one can see immediately from definition (1). From the fact that J is a support functional, one can deduce that J is convex, one homogenous, and lower semicontinuous on $\mathcal{L}^2(\Omega)$. Furthermore, the convex conjugate of J is given by the indicator function of the closure of K ,

$$J^*(v) = \begin{cases} 0 & \text{if } v \in \operatorname{cl}(K), \\ \infty & \text{otherwise.} \end{cases} \quad (5)$$

It also follows that if, in addition, the data fidelity term F is convex, lower semicontinuous, and coercive, the model (3) has a minimizer, which is unique if F is strictly convex [2].

The perimeter of a set Perhaps the most remarkable fact about the total variation is its geometric properties, which become apparent when one computes the total variation of a *characteristic function* 1_U of a subset $U \subset \Omega$. If U is a “sufficiently nice” set, it can easily be shown with Gauss’ integral theorem that $J(1_U)$ is equal to the $(n - 1)$ -dimensional Hausdorff measure $\mathcal{H}^{n-1}(\partial U)$ of the boundary of U . For $n = 2$, this corresponds to the length of the boundary of U , for $n = 3$, to the surface area. This observation motivates the definition of the *perimeter* of an arbitrary set U in Ω as

$$\operatorname{Per}(U, \Omega) := J(1_U). \quad (6)$$

Sets of finite perimeter are known as *Caccioppoli sets*, by definition $1_U \in \mathcal{BV}(\Omega)$ if and only if U is a Caccioppoli set. The geometric insight that $\operatorname{Per}(U, \Omega)$ measures boundary surface area is leveraged to reformulate weighted minimal surface problems in terms of minimization problems over binary functions of bounded variation.

Co-area The co-area formula in its geometric form says that the total variation of a function equals the

integral over the $(n - 1)$ -dimensional area of the boundaries of all its lower level sets, see Fig. 1. More precisely [8],

$$J(u) = \int_{-\infty}^{\infty} J(1_{\{u \leq t\}}) dt = \int_{-\infty}^{\infty} \operatorname{Per}(\{u \leq t\}, \Omega) dt. \quad (7)$$

It is an important tool to analyze convex relaxation techniques in labeling problems [15], where it can sometimes be employed to show global optimality of a method.

Variants Interesting variants of the total variation used in image processing include nonlocal TV [9] which replaces the gradient operator with interactions across the whole image domain as well as total generalized variation [4] which incorporates higher-order derivatives of the test functions. Total variation can also be defined for vector-valued functions [1, 2] and used as a regularizer for color image processing problems [7] or geometric penalizer in multi-label problems [11].

Application

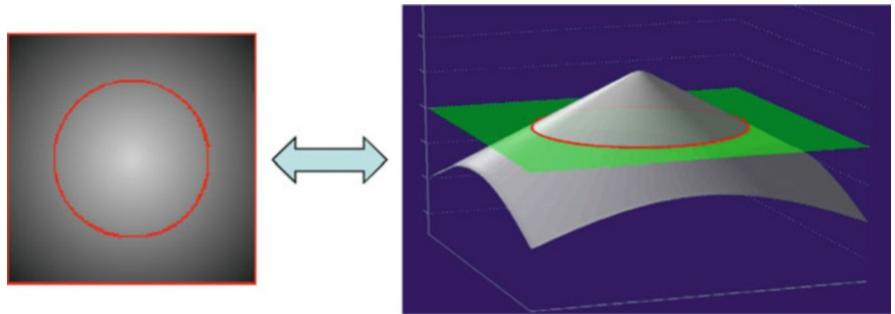
There are several important problems in image processing and computer vision which make use of the total variation. One application is to use J as a prior term in inverse problems; others leverage the geometrical properties of J to formulate segmentation problems. Weighted minimal surfaces can be viewed as a special class of segmentation problems and are heavily used in 3D reconstruction. Total variation can also be formulated for scalar fields on manifolds, which makes it possible to solve image processing problems on curved surfaces.

TV as a prior Assume one observes an image $f \in \mathcal{L}^2(\Omega)$, which is related to an unknown variable $u \in \mathcal{L}^2(\Omega)$ by a linear operator A and the ideal model $f = Au$. Typically, the problem is ill-posed and the observation f is noisy, so a regularization is required. Instead of trying to solve the linear model directly, one minimizes

$$\operatorname{argmin}_{u \in \mathcal{L}^2(\Omega)} \left\{ J(u) + \frac{1}{2\sigma^2} \|Au - f\|_2^2 \right\} \quad (8)$$

Total Variation, Fig. 1

Co-area formula: The total variation of a function can be computed by integrating the boundary length of all lower level sets; (red lines)



This can also be interpreted as the maximum a posteriori estimate [18] for the unknown u under the assumption that f is contaminated with point-wise independent Gaussian noise with standard deviation σ . In this probabilistic interpretation, the total variation has taken the place of the negative log probability for the unknown u . Since this prior probability distribution is rarely known in practice, an objective prior like TV has to be used. Equation (8) describes a typical inverse problem in image processing formulated as a *variational model* and has been intensively studied. A recent overview of minimization methods is given, for example, in [5]. In the case that A is the identity, problem (8) is known as the Rudin-Osher-Fatemi (ROF) model for image denoising, named after the authors of [17], in which total variation made its first appearance as a regularizer in image processing. If A is instead a convolution with a blur kernel, one arrives at total variation deblurring models [6]. Similar models are also used for super-resolution [13].

TV in segmentation problems The archetypical binary segmentation problem can be formulated as an optimization problem over the space of binary functions of bounded variation

$$\operatorname{argmin}_{u \in \mathcal{BV}(\Omega, \{0,1\})} \left\{ J(u) + \int_{\Omega} c \cdot u \, dx \right\}. \quad (9)$$

If $u(x) = 1$, the point x belongs to region 1 and vice versa. The real-valued function c denotes the assignment cost given by the model, where a negative cost $c(x) < 0$ indicates a preference of x to be assigned to region 1. For example, if probabilities p_1, p_2 are given for each point to belong to region 1 or 2, respectively, then a typical choice is $c = \log(p_2/p_1)$. The use of the total variation of u as a regularizer corresponds to a

penalization of the length of the interface between the regions, as discussed for Eq. (6).

The simple binary model above can be viewed as the piecewise constant two-region case of the famous Mumford-Shah functional [14], which is very hard to minimize. In contrast, the model (9) can be minimized globally by relaxation to a convex model and subsequent thresholding [15]. The proof relies on the co-area formula (7), which allows to write the energy as an integral over the energies of all level sets of u . A global solution is also available for the generalization from binary labeling to a continuous interval of labels with linear penalizer [16].

TV in 3D reconstruction. In variational approaches to 3D reconstruction, one looks for a surface which minimizes the local photo-consistency error ρ , given as a nonnegative function on a domain $\Omega \subset \mathbb{R}^3$. This is modeled as a weighted minimal surface problem, where the desired surface minimizes the surface integral over ρ . Finding a weighted minimal surface is a special case of binary segmentation in the sense the space Ω is partitioned into a region inside and outside the surface, respectively. A more general form of the co-area which includes a positive weight function ρ states that the weighted total variation of a characteristic function 1_U equals the weighted surface area of ∂U . Formally [8],

$$\int_{\Omega} \rho |\nabla 1_U|_2 \, dx = \int_{\partial U} \rho \, d\mathcal{H}^{n-1}.$$

One noteworthy application of this relationship is to formulate 3D reconstruction with a convex functional, such that globally optimal solutions can be obtained [12].

Total variation and related image processing problems can also be defined and solved on manifolds [3].

In the context of 3D reconstruction, a total variation super-resolution model on the surface can be employed to reconstruct high-resolution texture maps from multiple views [10].

References

1. Ambrosio L, Fusco N, Pallara D (2000) Functions of bounded variation and free discontinuity problems. Oxford University Press, Oxford/New York
2. Attouch H, Buttazzo G, Michaille G (2006) Variational analysis in Sobolev and BV spaces. MPS-SIAM series on optimization. Society for Industrial and Applied Mathematics, Philadelphia
3. Bertalmio M, Sapiro G, Cheng LT, Osher S (2001) Variational problems and PDEs on implicit surfaces. In: Proceedings of the IEEE workshop on variational and level set methods (VLSM'01), pp 186–193
4. Bredies K, Kunisch K, Pock T (2010) Total generalized variation. SIAM J Imaging Sci 3(3):492–526
5. Chambolle A, Caselles V, Cremers D, Novaga M, Pock T (2010) An introduction to total variation for image analysis. Radon Ser Comput Appl Math 9:263–340
6. Chan T, Wong C (1998) Total variation blind deconvolution. IEEE Trans Image Process 7(3):370–375
7. Duval V, Aujol JF, Vese L (2009) Projected gradient based color image decomposition. In: Tai X-C, Mørken K, Lysaker M, Lie K-A (eds) Scale space and variational methods in computer vision. Springer, Berlin/Heidelberg, 295–306
8. Federer H (1996) Geometric measure theory. Number 153 in classics in mathematics, Springer reprint of the 1969 edn. Springer, Berlin/Heidelberg/New York
9. Gilboa G, Osher S (2008) Nonlocal operators with applications to image processing. Multiscale Model Simul 7(3):1005–1028
10. Goldluecke B, Cremers D (2009) Superresolution texture maps for multiview reconstruction. In: IEEE international conference on computer vision (ICCV), Kyoto, Japan
11. Lellmann J, Becker F, Schnörr C (2009) Convex optimization for multi-class image labeling with a novel family of total variation based regularizers. IEEE Int Conference on Computer Vision (ICCV). Kyoto, Japan
12. Kolev K, Klodt M, Brox T, Cremers D (2009) Continuous global optimization in multiview 3D reconstruction. Int J Comput Vis 84(1):80–96
13. Marquina A, Osher S (2008) Image super-resolution by TV-regularization and Bregman iteration. J Sci Comput 37(3):367–382
14. Mumford D, Shah J (1989) Optimal approximation by piecewise smooth functions and associated variational problems. Commun Pure Appl Math 42:577–685
15. Nikolova M, Esedoglu S, Chan T (2006) Algorithms for finding global minimizers of image segmentation and denoising models. SIAM J Appl Math 66(5):1632–1648
16. Pock T, Schoenemann T, Gruber G, Bischof H, Cremers D (2008) A convex formulation of continuous multi-label problems. In: European conference on computer vision (ECCV), Marseille, France, pp 792–805
17. Rudin LI, Osher S, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. Physica D 60(1–4):259–268
18. Szeliski R (1990) Bayesian modeling of uncertainty in low-level vision. Int J Comput Vis 5(3):271–301

Tracker-Camera Calibration

► Hand-Eye Calibration

Transparency

► Transparent Layers

Transparency and Translucency

Manish Singh

Department of Psychology, Rutgers University,
Piscataway, NJ, USA

Related Concepts

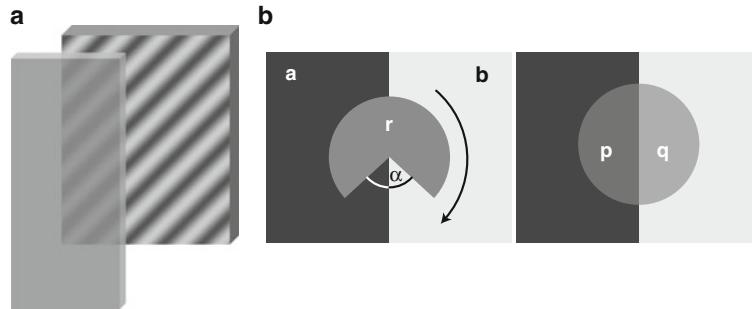
► Image Decompositions; ► Opacity; ► Transparent Layers

Definition

Transparency is the property of some materials that allows light to be partially transmitted through. The proportion of light a material transmits through determines its transmittance, α . The term “translucency” is generally used in cases where light is transmitted through diffusely.

Background

When a surface is viewed through a partially transmissive material, the optical contributions of the two layers in a given viewing direction are collapsed onto a single intensity in the projected image. If a computer-vision system is to recover the scene correctly, it must be able to decompose or *scission* the image intensity into the separate contributions of the two material layers (see Fig. 1a).



Transparency and Translucency, Fig. 1 (a) Illustration of the problem of transparency: In each visual direction, the contributions of two distinct layers are collapsed onto a single pixel

The inverse problem of recovering layered surface structure is generally divided into two subproblems: (1) the qualitative problem of inferring the *presence* of an interposed partially transmissive layer in parts of the image and (2) the quantitative problem of assigning surface properties – reflectance and transmittance – to the separate layers.

Theory

Physical Models

Metelli [15] used Talbot's equation for color mixing to model transparency: The "color" of the partially transmissive surface or filter (r) and that of the underlying opaque surface (a) are mixed linearly, with the mixing proportions determined by the transmittance α of the transparent layer:

$$p = (1 - \alpha)r + \alpha a \quad (1)$$

Metelli noted that this equation is consistent with a simple physical model of transparency involving an *episcotister* – a rapidly rotating disk with an open sector – placed in front of a background surface (see Fig. 1b, c). The same equation is also consistent with an opaque surface with a large number of holes that are too small to be resolved individually (e.g., a mesh). In the latter case, the "color mixing" takes place spatially, rather than over time. (Metelli himself considered only the achromatic case, so the colors he refers to are "achromatic colors," but his model has also been extended to the chromatic domain, e.g., [8, 10]).

Physically more accurate models of transparency incorporate the pattern of light reflection and

intensity. These contributions must be disentangled if the scene structure is to be recovered. (b) Metelli's model of transparency based on a linear combination of the two contributions

transmission between a transparent filter and the underlying opaque surface (see Fig. 2a)[6, 7, 10, 11, 17, 22], leading to the following equation:

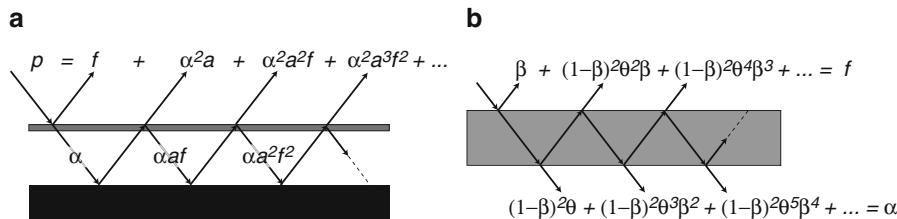
$$p = f + \frac{\alpha^2 a}{1 - fa} \quad (2)$$

In this equation, reflectance f plays the same role as the product $(1 - \alpha)r$ in Metelli's model. Moreover, f and α can be further expressed in terms of intrinsic physical parameters of the partially transmissive filter (see Fig. 2b) – its *reflectivity* β (at the air-filter interface) and *inner transmittance* θ (ratio of radiant flux that reaches the back surface of the transparent filter to the flux that enters the filter at its front surface) for example, [10, 17, 22]. (For models of subsurface scattering within volumetric translucent materials, see e.g., [14]. In such cases, background structure is often not visible through the translucent material, e.g., marble or cheese.)

It has been argued that, despite its simplicity, Metelli's linear equation provides a reasonable approximation to the more complete filter model [11], as well as to other more complex cases such as fog and haze [12]. Computer-vision algorithms aimed at undoing the effects of fog and haze similarly employ a linear generative model, containing a multiplicative – attenuative – component due to absorption and scattering (a negative exponential of the fog extinction coefficient, generally taken to be a constant), plus an additive contribution of the airlight [9, 13, 16, 18, 21].

The Inverse Problem

Given a single equation such as Eq. 1) above, it is clearly impossible to determine transmittance α and



Transparency and Translucency, Fig. 2 (a) Model of a transparent filter that takes into account the pattern of transmission and reflection between the filter and the underlying opaque

reflectance r from knowledge of p and a alone. However, if the background surface contains *two* distinct “colors” a and b – both partly visible through the same transparent filter – the two equations:

$$p = (1 - \alpha)r + \alpha a \text{ and } q = (1 - \alpha)r + \alpha b \quad (3)$$

can be uniquely solved for α and r :

$$\alpha = \frac{p - q}{a - b}; \quad r = \frac{aq - bp}{a + q - b - p} \quad (4)$$

(Note that, more generally, with multiple colored patches visible through the transparent layer, a common solution may not exist and least-squares methods may be required; see, e.g., [8].)

Transparent overlap of surfaces generically leads to X-junctions in the image (see Fig. 3). However X-junctions differ greatly in the degree of local support they provide for interpretations of transparency. Compare, for instance, the interpretations of perceived transparency elicited by the three displays in Fig. 3. In addition to making quantitative predictions for transmittance and reflectance, the above solutions also yield qualitative predictions for interpretations of transparent overlap and depth layering [6, 7, 15]. In order to have valid transparency, α must be nonnegative with magnitude no larger than 1. Hence, based on the expression for α above, it follows that:

1. $\text{sign}(p - q) = \text{sign}(a - b)$. The contrast polarity must be the same inside and outside a putative transparency boundary.
2. $|p - q| \leq |a - b|$. The magnitude of the luminance difference must be no greater inside the transparency boundary than outside it.

Based on the polarity constraint above, researchers have classified X-junctions into three qualitative kinds [1, 2]. In a *non-reversing* junction, contrast polarity is

surface. (b) The overall transmittance and reflectance of the filter are functions of two intrinsic parameters: reflectivity (β) and inner transmittance (θ)

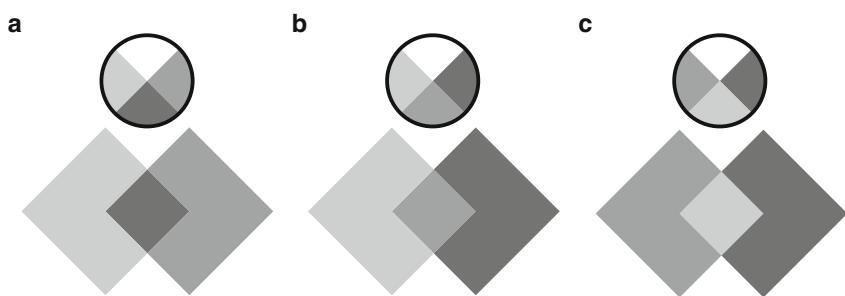
preserved across both edges of the “X”; hence, either edge could be the boundary of a transparent layer (Fig. 3a). In a *single-reversing* junction, contrast polarity is preserved across one edge only; hence, there is only one possible interpretation of transparent overlap (Fig. 3b). Finally, in a *double-reversing* junction, neither edge preserves contrast polarity, so this junction type does not support an interpretation of transparency (Fig. 3c).

The above analysis provides local constraints for the interpretation of transparency. Local support can often be overruled by global context, however. Mechanisms are needed to integrate local evidence across the image. In the achromatic domain, an integration algorithm was proposed by Singh and Huang [20], which propagates local junction information by searching for chains of polarity-preserving X-junctions with consistent sidedness (i.e., side with lower contrast), and then propagating the transparency labeling to interior regions. Despite its simple, deterministic nature, the algorithm performs well on synthetic images and simple real images with well-defined X-junctions.

In the chromatic domain, D’Zmura et al. [8] developed an algorithm for separating transparent overlays that (1) searches for chains of X-junctions and (2) tests for a consistent chromatic convergence across this chain. This algorithm is based on the finding that a convergence (possibly accompanied by a translation) in color space across a boundary tends to generate the percept of an overlying color filter, whereas other transformations such as shears and rotations do not. (This result is largely consistent with Metelli’s model extended to the color domain – although some transformations that cannot be achieved with Metelli’s equations, such as equiluminous translations in color space, can also generate a percept of transparency [8].)

Transparency and Translucency, Fig. 3

Classification of X-junctions based on their support for the interpretation of transparent overlap. (a) Non-reversing junctions. (b) Single-reversing junctions. (c) Double-reversing junctions



Going Beyond Junctions

The schemes reviewed above rely on the accurate extraction of junctions in images. A reliance on junctions may ultimately be too restricting, however, especially in images of natural scenes. For example, partially transmissive media with varying opacity, such as fog and haze, generate gradual changes in contrast along an edge in the background, without well-defined junctions. In order to deal with such cases, Anderson [3] proposed the *transmittance-anchoring principle* which states that, as long as contrast polarity is preserved across a contour, the visual system anchors the highest-contrast regions to full transmittance (i.e., surfaces seen in plain view), whereas lower-contrast regions are perceived as containing an overlying transparent layer with varying degrees of transmittance. Perceptual experiments suggest that human observers do anchor transmittance in this way [4, 5].

Application

Recent work in computer vision has focused on undoing the effects of fog and haze in images, and recovering versions of these images that are free of atmospheric disturbance. One class of methods relies on having multiple images of a scene, taken either under different levels of atmospheric disturbance [16], or with different degrees of polarization [18]. More recent methods, relying on a single image, have adopted a number of strategies, including maximizing local contrast in the restored image [21], the assumption that transmission and surface shading are uncorrelated [9], and the “dark-channel prior” – the assumption that in images of outdoor scenes taken under clear viewing conditions, most local patches tend to contain pixels with very low intensity in at least one of the color channels [13].

As noted above, another class of algorithms designed to separate transparent overlays from the background relies on the explicit extraction of X-junctions [8, 20]. These algorithms work well in simple images with well-defined junctions; it remains to be seen how well this approach scales up to complex natural images.

Physical Versus Perceptual Models

One issue that computer-vision systems must consider is whether they are aimed at recovering physically accurate estimates of the transmittance of a transparent material or obtaining estimates that are aligned with the way human observers perceive transmittance. One can readily imagine applications in which either may be more appropriate. Perceptual experiments have shown that the human perception of surface transmittance is not well predicted by Metelli’s equations, and can deviate significantly from physical predictions. Specifically, these studies have shown that the visual estimate of transmittance is based on the *perceived contrast* within the region of transparency (suitably normalized by the perceived contrast in surrounding regions); and perceived contrast is not well predicted by the luminance differences (or luminance range) that appear in Metelli’s solution for transmittance α ; see for example, [4, 17, 19]. Consistent with a perceived-contrast model, a dark-colored surface can visually appear to be significantly more transmissive than a light-colored surface of the same physical transmittance.

References

1. Adelson EH (2000) Lightness perception and lightness illusions. In: Gazzaniga M (ed) The new cognitive neurosciences. MIT, Cambridge, pp 339–351

2. Adelson EH, Anandan P (1990) Ordinal characteristics of transparency. In: AAAI-90 workshop on qualitative vision, Boston
3. Anderson BL (2003) The role of occlusion in the perception of depth, lightness, and opacity. *Psychol Rev* 110: 762–784
4. Anderson BL, Singh M, Meng J (2006) The perceived transmittance of inhomogeneous surfaces and media. *Vision Res* 46: 1982–1995
5. Anderson BL, Winawer J (2008) Layered image representations and the computation of surface lightness. *J Vision* 8(7):18
6. Beck J (1985) The perception of transparency in man and machine. *Computer Vis Graph Image Process* 31: 127–138
7. Brill MH (1984) Physical and informational constraints on the perception of transparency and translucency. *Computer Vis Graph Image Process* 28:356–362
8. D'Zmura M, Colantoni P, Knoblauch K, Laget B (1997) Color transparency. *Perception* 26:471–492
9. Fattal R (2008) Single-image dehazing. In: ACM transactions on graphics (Proceedings of SIGGRAPH), Los Angeles, CA
10. Faul F, Ekroll V (2002) Psychophysical model of chromatic perceptual transparency based on subtractive color mixture. *J Opt Soc Am A* 19:1084–1095
11. Gerbino W (1994) Achromatic transparency. In: Gilchrist AL (ed) Lightness, brightness, and transparency. Erlbaum, Hillsdale, pp 215–255
12. Hagedorn J, D'Zmura M (2000) Color appearance of surfaces viewed through fog. *Perception* 29: 1169–1184
13. He K, Sun J, Tang X (2009) Single image haze removal using dark channel prior. Computer Vision and Pattern Recognition in proceeding of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), Miami Beach, FL
14. Jensen HW, Marschner SR, Levoy M, Hanrahan P (2001) A practical model for subsurface light transport. In: ACM Transactions on Graphics (Proceedings of SIGGRAPH), Los Angeles, CA, pp 511–518
15. Metelli F (1974) The perception of transparency. *Sci Am* 230:90–98
16. Narasimhan SG, Nayar SK (2003) Contrast restoration of weather-degraded images. *IEEE Trans Pattern Anal Mach Intell* 25:713–724
17. Robillotto R, Zaidi Q (2004) Perceived transparency of neutral density filters across dissimilar backgrounds. *J Vision* 4:183–195
18. Schechner YY, Narasimhan SG, Nayar SK (2003) Polarization-based vision through haze. *Appl Opt* 42: 511–525
19. Singh M, Anderson BL (2002) Toward a perceptual theory of transparency. *Psychol Rev* 109:492–519
20. Singh M, Huang X (2003) Computing layered surface representations: an algorithm for detecting and separating transparent overlays. *Proc Comput Vis Pattern Recognit (CVPR)* 2:11–18
21. Tan RT (2008) Visibility in bad weather from a single image. Computer vision pattern recognition in proceeding of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), Anchorage, Alaska, USA
22. Wyszecki G, Stiles WS (1982) Color science: concepts and methods, quantitative data and formulae. Wiley, New York

Transparent Layers

Yanghai Tsin

Corporate R&D, Qualcomm Inc., San Diego, CA, USA

Synonyms

Transparency

Related Concepts

► [Polarized Light in Computer Vision](#); ► [Transparency and Translucency](#)

Definition

In computer vision, the term *transparent layers* refers to physical objects located at different depths from an imaging device, in such a configuration that irradiance coming from all these objects contributes to non-negligible portions of the final intensities of the *same* pixels, giving the perception of transparency.

Solving a *transparent layer problem* entails estimating a subset of the following values: intensity/color of each layer, depth of each layer, geometric parameters related to the camera pose and scene structure, and optical properties of the transparent layers.

Background

Phenomena of transparent layers can appear in many imaging processes. The most typical transparent layers occur due to *transparency*. For instance, objects seen through a textured and transparent material, for example, a tinted glass window or a thin curtain, where the layers include a front layer of transparent object and a rear layer of the background objects. [Figure 1](#) illustrates this case.



Transparent Layers, Fig. 1 Transparent layers caused by transparency. *Left*, a photograph of a lake containing transparent layers: a layer composed of the water surface that is textured by

ripples and a farther layer of pebbles on the lake bottom (Photograph courtesy of Amanda Y. Fu). *Right*, a photograph of a transparent silk screen in front of a couple of dolls

Some transparent layers are caused by *reflection*, see Fig. 2. Specular surfaces, such as a picture frame, reflect scene structures at a distance. A reflected image from a surface appears to be at a different depth from the reflecting surface itself. In the above example, the picture frame forms the closer layer and the reflected scene forms the more distant layer. The apparent depth of the reflected layer depends on multiple factors, including shape of the reflector and scene geometry.

Transparent layers can also appear because of *point spread*. Cameras have finite-sized imaging elements. In addition, due to imperfection in optical focusing, an observed image is usually a kernel-smoothed version of the scene radiance. The convolution kernel, known as the *point spread function* (PSF), is due to a combination of camera optics and the finite size of an imaging device. As a result of PSF blurring, pixels corresponding to boundaries of opaque objects at different depths contain contributions from both layers, giving the perception of transparency. Figure 3 illustrates two of such examples.

Penetrating electromagnetic radiation, such as X-ray or visible light, produces the perception of transparent layer. X-ray images are among the most sophisticated transparent effects, composed of many layers of objects that undergo nonrigid complex motions. Another prime example of this case is optical microscopy, where layers of cells at different depths are superposed into a final visible spectral image. Figure 4 shows such examples.

When the relative speed between a camera and an imaged scene is not negligible compared to the shutter

speed, *motion blur* can cause blending of foreground and background scenes and give the impression of transparency. Blurring at a pixel is caused by integrating radiance contributions from scene points from different depths or different parts of an object. See Figure 5 for an example.

Irradiance, i.e., light coming into a camera, travels through a *transmitting medium*, for example, air, water, and glass. Sometimes, the effects of the media in between are nontrivial, for example, due to excessively heated air, fog, milky water, or thick tinted glasses. The medium causes color shifts and distortion of the images, and sometimes, it is interesting to study the medium and background in the context of transparent layers. See Fig. 6 for two such examples.

People study transparent layers for many different reasons, for instance, to more accurately estimate motion or depth of a scene by separating the layers, thus removing non-Lambertian effects due to layer blending [3–9], to render images containing transparent layers from novel viewing angles [10, 11], for inexpensive content creation and manipulation in applications such as gaming and movies, and to enhance visualization of one or all component layers, such as in medical imaging [12, 13].

Theory

A linear superposition model of two layers can be summarized as

$$I(x) = \mathcal{T}_1(\theta_1) \circ I_1(x) + \pi(\mathcal{T}_2(\theta_2) \circ I_2(x), \alpha(x)), \quad (1)$$



Transparent Layers, Fig. 2 Transparent layers due to *reflection*. *Left*, a textured pitcher surface reflects objects in the scene. *Right*, photograph in a picture frame reflects a dodecahedron and a tea box



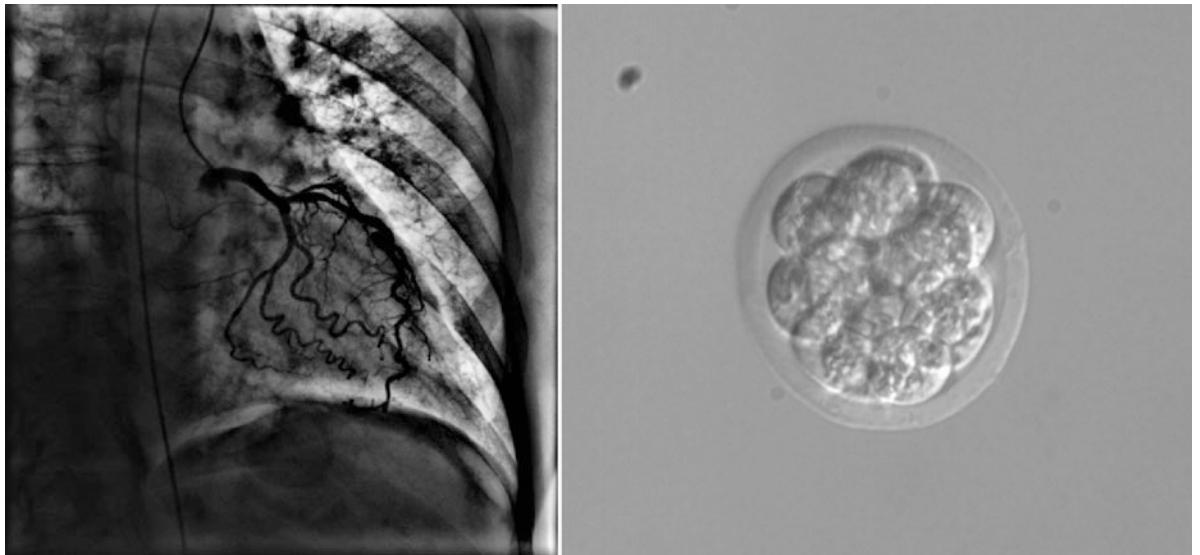
Transparent Layers, Fig. 3 Sub-pixel blending due to *point spread*. *Left*: sub-pixel blending occurring at boundaries of surfaces at different depths. The *insets* show boundary image regions (image courtesy of Scharstein and Szeliski [1]). *Right*:

sub-pixel blending due to hairy foreground objects. The *inset* shows the α -map of the foreground object, where the intensity of the map is proportional to the opacity of a foreground pixel (image courtesy of Levin et al. [2])

where:

- $I(x)$ is the observed image that is a superposition of different layers.
- $I_l, l = 1, 2$ represents radiance toward the viewing camera in the absence of the other layer. I_2 can itself be a superposition of layers, which generalizes (Eq. 1) to more than two layers in a recursive fashion.
- $\alpha(x)$ is a vector of blending parameter of the first layer, affected by properties such as constituent material types, distribution of material types, or thickness of the layer. In general, $\alpha(x)$ can be spatially varying.

- $\mathcal{T}_l(\theta_l) \circ, l = 1, 2$ represents the function that warps a layer (I_1 or I_2) to the current camera view from a reference camera view. The warping function is parameterized by geometric parameters $\theta_l, l = 1, 2$, for example, point depths or affine motion model parameters. The warping function $\mathcal{T}_l \circ$ can be parametric motions [8], nonparametric, pixel-wise motion such as in a stereo setting [9] or smooth, nonrigid motion such as in X-ray images.
- The function $\pi(\cdot)$ describes the physical process of composing a second layer given blending parameters of the first layer.



Transparent Layers, Fig. 4 Penetrating electromagnetic radiation produces transparent layers. *Left*, the image shows an X-ray image (angiogram) of a human chest. Visible in the image are dye contrast-enhanced coronary arteries of the heart, ribs, spine bones, and soft tissues appearing at different distance to the

X-ray machine. *Right*, image shows transparent mouse embryo cells overlay on top of each other (image courtesy of Bill Warger and Judy Newmark and the Mouse embryos DIC image set is available from the Broad Bioimage Benchmark Collection (www.broad.mit.edu/bbbc))

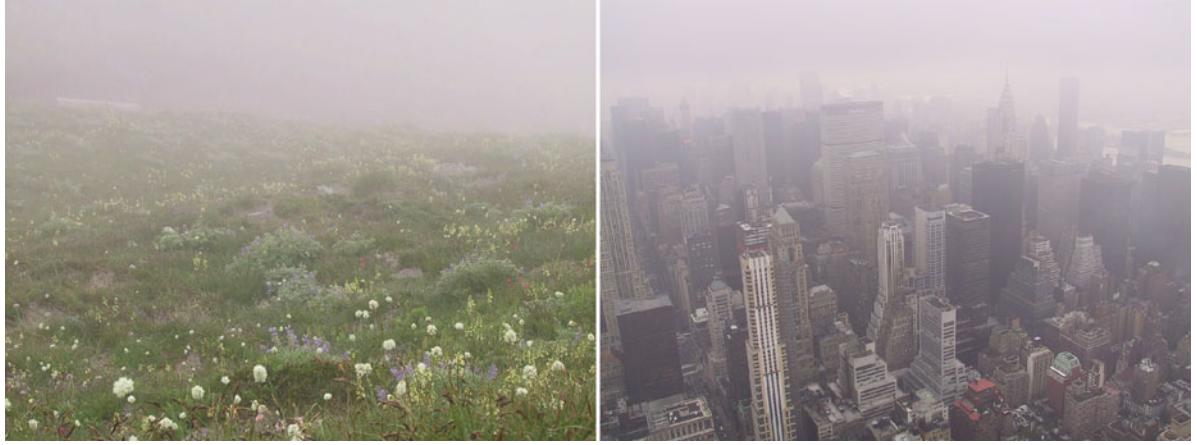


Transparent Layers, Fig. 5 Transparent layers due to *motion blur*. *Left*: when the train is static. *Right*: when the train is moving very fast compared to the camera shutter speed, the train appears to be transparent due to the long temporal integration

process. The contribution of a point on a background object to the intensity/color of a pixel is proportional to the amount of time the point is visible

Note that the linear superposition model, i.e., by adding up (weighted) contributions from all layers, is a good approximation to many computer vision problems. For X-ray images, due to the exponential decaying nature of radiation rays, the raw X-ray image I_x is not a linear superposition of layers. However, $\log I_x$ can be fairly accurately modeled by (Eq. 1).

In general, estimating all parameters of all layers is a *severely ill-posed problem*. For instance, for each pixel x in a single observed image, there is one set of observations $I(x)$ but many unknowns, i.e., geometric parameters θ_l , blending parameters α , and the colors of the constituent layers I_1 and I_2 . Fortunately, it is not always necessary to estimate all parameters at once. We are not confined to use a single image to



Transparent Layers, Fig. 6 *Translucent Media* can contribute nontrivial transparent layers. The above images show photographs of foggy days, when the transparent layer due to the medium, i.e., air mixed with water droplets, becomes nontrivial

solve a transparent layer problem either. In fact, in the majority of studies, researchers use multiple images of the underlying scene. In specific problem settings, we can define well-posed problems.

There is a rich literature that studies transparent layers in computer vision. These methods utilize characteristics of transparent layers, such as physical model, statistical priors, or motion, to analyze transparent layers. The approaches range from explicitly separating the transparent layers to eliminating layers of no interest by means of integration or difference. The following is a summary of these approaches and how they constrain the problems to be well-posed:

Using defocusing and polarization cues. The first set of approaches utilizes physical properties of the image formation process to define well-posed layer separation problems. In these models, it is usually assumed that both layers can be approximated by fronto-parallel planes, and multiple images are taken from a *static camera*. As a result, scene points within the same layer share a single depth value, $\theta_l(x) = d_l, l = 1, 2$. The static configuration makes the warping functions \mathcal{T}_l trivial, i.e., identity transformations. Schechner et al. [14] showed that the depth values d_l can be estimated using defocusing cues. They utilized a set of photographs taken with different focuses. Using a “focus measure,” the two focal lengths that correspond to the sharpest focuses in both transparent layers are estimated. Knowing camera calibration *a priori*, the layer depths and PSF’s for each layer can be computed

from the estimated focal lengths. The observed pair of images that correspond to the most sharply focused images can be written as

$$\begin{cases} I^{(1)} &= I_1 + I_2 \otimes h_a \\ I^{(2)} &= I_1 \otimes h_b + I_2 \end{cases}, \quad (2)$$

where \otimes means convolution; $I^{(k)}, k = 1, 2$ is the observed composite image when I_k is in focus. The above problem is still ill posed in that we can add a constant (or even a low frequency) component C to the solution of one layer and subtract the same from the other, and the resulting layer images, $I^{(1)} + C$ and $I^{(2)} - C$, are still valid solutions to (Eq. 2). However, if we fix the DC components of one layer, the problem can be solved by least squares. In the same spirit, using a rotating polarizer in front of the camera lens, Schechner et al. [15] show that the observed image is a linear combination of the layers, where the linear combination coefficients are functions of inclination angles. The inclination angles can be estimated by minimizing the correlation between component layers. The key insights for formulating a well-constrained problem in both methods are using static cameras and geometric regularity of scenes to reduce the dimension of θ to 2, trivializing the warping functions \mathcal{T} , and estimating blending coefficients (PSF kernels or inclination angles) independent of layer radiances.

Utilizing an atmospheric scattering model. Unfavorable weather conditions can cause lowered visibility,

due to floating particles in the air, including aerosols, water droplets, rain, or snow. In these conditions, atmospheric effects can no longer be ignored, and the atmosphere itself can be considered as a transparent layer: it alters and adds to the color of the scene radiance. Narasimhan and Nayar [16] conducted a thorough study of transparent layers due to atmospheric scattering. They showed that through reasonable assumptions and simplification of an atmospherical model, the transparent layer problem can be well constrained and solved: the depth from a scene point can be estimated up to a constant scale and the scene color can be estimated. For example, in its simplest form, images taken at *night* can be modeled as a simplified version of (Eq. 1)

$$I = \pi(I_2, \alpha) = g e^{-\beta d} I_2, \quad (3)$$

where g is a geometric constant; β being a constant attenuation coefficient, and d the scene depth. Equation Eq. 3 implies that in the absence of sunlight or skylight that adds to the *air light*, i.e., color of the fog, the observed image, I is an attenuated version of the scene radiance I_2 . Given two images of the scene under different atmospherical conditions (clear nights, light fog, or dense fog), corresponding to different β , log ratio of the two observed images reveals that

$$\ln\left(\frac{I^{(1)}}{I^{(2)}}\right) = -(\beta^{(1)} - \beta^{(2)})d. \quad (4)$$

As a result, we can estimate the scene depth up to unknown constant scale factor. In the presence of sunlight and skylight, the air light can no longer be ignored and the transmission layer composes the first layer, i.e., fog color is I_1 . In this case, an additional *dichromatic* model can be used to constrain the problem [16], resulting in well-posed problems. By utilizing a *dark channel prior*, He et al. [17] introduced a method that reconstructs the transparent layers and scene depth using a single image.

Statistical Approaches. Another effective way of transforming (Eq. 1) into a well-posed problem is by using statistical properties of I_1 and I_2 . Farid and Adelson [18] studied the case where an observed image is a linear mixture of two-component layers. By rotating a polarizer in front of the camera, they were able to capture images with different mixing

parameters for different polarization angles. A simplified version of (Eq. 1) now has the form

$$\begin{cases} I^{(1)}(x) = a \cdot I_1(x) + b \cdot I_2(x) \\ I^{(2)}(x) = c \cdot I_1(x) + d \cdot I_2(x) \end{cases}. \quad (5)$$

The key assumption for their approach is that $I_1(x)$ and $I_2(x)$ are independent. As a result, the mixing coefficients should be sought such that after the inversion

$$\begin{pmatrix} I_1(x) \\ I_2(x) \end{pmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} \begin{pmatrix} I^{(1)}(x) \\ I^{(2)}(x) \end{pmatrix}. \quad (6)$$

$I_1(x)$ and $I_2(x)$ are maximally decorrelated (an approximation to the independence assumption). This is done by rotating and stretching the joint intensity distribution such that the resulting distribution is as separable as possible (the most decorrelated distribution). The parameters for rotation and stretching can be computed in close form if the observed joint distribution $(I^{(1)}(x), I^{(2)}(x))$ is approximated by a principal component analysis model. Schechner et al. [19] proposed a similar approach, where the mixture is caused by focusing at one layer at a time. The point spread function of the defocused layer is estimated by minimizing mutual information between the two layers. Instead of using the original intensity, Bronstein et al. [20] convolve the input image pairs with high pass filters. The filtered images have mostly zero values and present sparse distributions. They show that sparse ICA can better separate the component layers. Levin et al. [21] demonstrate that for simple images the component layers can be recovered from just one image. From natural images, they learn the exponential density functions of several filter responses. The best separation of the input image is the one that maximizes the learned density function. However, their algorithm fails in more complicated images where texture is abundant.

Motion Analysis. Under certain conditions, by using images taken at multiple viewpoints (due to relative motion between the camera and the scene), we can transform (Eq. 1) into well-posed problems. In Szeliski et al. [8], the component layers are assumed to undergo parametric motions, for example, the motion observed between two frames can be parameterized by a 2D affine motion model. Furthermore, by assuming constant linear blending coefficients between the two

layers, the linear superposition model (Eq. 1) can be simplified

$$\begin{aligned} I^{(i)}(x) &= \mathcal{T}_1^{(i)}(\theta_1) \circ I_1 + \mathcal{T}_2^{(i)}(\theta_2) \circ I_2(x) \\ &= I_1(T(x, \theta_1^{(i)})) + I_2(T(x, \theta_2^{(i)})) , \end{aligned} \quad (7)$$

where $T(x, \theta)$ is the parametric transformation that maps a point in the i th observed frame to the reference frame. Given F frames from the motion sequence, assuming dimensionality (length) of θ being d and that there are N pixels in a frame, we have $2N+2d(F-1)$ unknowns and approximately $F \cdot N$ constraints (excluding boundary effects). Because $N \gg d$, the number of constraints grows much faster than the number of unknowns as more frames are observed. As a result, given a few frames, the problem can be well constrained.

However, solution to this well-constrained problem is not trivial. Szeliski et al. [8] took an iterative approach. An initial set of transformations θ_l^i was estimated using image alignment methods, assuming that one of the layers has dominant texture patterns. The images are warped back to the reference view, and by observing that image intensities are nonnegative, they used a *min-composite* algorithm to initialize intensity. Once an initialization is obtained, the algorithm then iteratively refines the motion and intensities of each layer.

Tsin et al. [9] studied the case for stereo matching when linear superposition of layers occurs. The unknowns in this case are per-pixel color and depth, i.e., $4N$ unknowns, $2N$ for intensities, and $2N$ for layer depths. In a first step, depths are estimated using a *nested plane sweep* for computing matching costs associated with a depth hypothesis pair, one depth for each layer. An optimization algorithm, i.e., graph-cut optimization, was used to find a strong minimum. The key observation in this step is that a cost associated with a depth hypothesis pair can be derived without knowing the layer intensities. After the depths are estimated, layer intensities can be solved using least squares in a second step.

There is a plethora of other work using motion sequences for layer separation. One of the earliest work on layer extraction is that of Wang and Adelson [22]. They compute per-pixel flow, perform K-means clustering in affine motion space, and assign each pixel to one of the sets of clustered affine parameters.

Darrell and Pentland [23] improved on the formulation by having a direct connection between layers and robust estimation. Shizawa and Mase [3] provide a theoretically elegant solution to extract multiple flow from an image sequence. They formulate the constraint for double optical flow in space-time frequency domain and use it as the energy minimization function. In a follow-up work [24], the principle of superposition is applied to the multiple optical flow problem. Ju et al. [6] used a layered piecewise affine motion model to simultaneously estimate multiple motions in both occlusive and transparent motion sequences. Swaminathan et al. [25] handled the problem of highlight detection and removal by *epipolar plane image* (EPI) analysis, explicitly detecting the saturated highlights in a local EPI stripe. Bergen et al. [4] introduced a differencing technique to deal with transparency. In contrast to differencing techniques, the integration technique of Irani et al. [5] stabilizes one motion region and blurs all other regions by integration or averaging.

In summary, the transparent layer problems are in general ill-posed and hard to solve. However, we can define and solve well-posed transparent layer problems by adding additional information, for example, using multiple images, known calibration of the camera (PSF or camera pose), simplified scene geometry, and polarized observations or statistical priors. A reader is encouraged to study closely related topics, especially *matting* [2, 7] and motion *deblurring* [26], for dealing with transparent layers at opaque object boundaries.

Open Problems

Open problems for transparent layers are listed below.

1. Solving transparent layer problems in *general settings* robustly. The existing solutions dealt with specific cases of transparent layer problems, but a solution that works in general real-world applications is still elusive.
2. Robustly separating transparent layers from *a single image*. The difficulty comes from the fact that the problem is severely under-constrained. To solve the problem, unbiased, proper priors need to be added to the problem. Very insightful algorithms have been developed in the literature using priors

- derived from natural image statistics [21]. But to be able to robustly recover component layers to the degree of human capability, these priors are still insufficient.
3. Robustly separating *more than two layers or unknown number of layers*. The ability to recover transparent layers in these cases critically depends on the robustness of the algorithm in dealing with distractions from un-modeled layers and the ability to handle potentially very low signal-to-noise ratio of layers in the background, due to signal attenuation.
 4. Robustly separating transparent layers *when the blending factor is unknown and spatially varying*.
 5. Motion-based separation of transparent layers undergoing *unknown and nonrigid motion*. The difficulties of layer separation and nonrigid motion estimation are coupled in this case.
- A good example of open problems is estimating motion and component layers in X-ray image sequences. The problem is complicated by the following factors: (1) unknown number of layers, depending on where an X-ray image is taken, it may contain images of many organs/bones superposed with each other; (2) Nonrigid motions of the layers, such as those caused by respiratory or cardiac motion. On the other hand, solving the problem has clinically important applications, for example, visualization enhancement and digital subtraction angiography [27].
- ## References
1. Scharstein D, Szeliski R (2002) A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV* 47(1):7–42
 2. Levin A, Rav-Acha A, Lischinski D (2008) Spectral matting. *IEEE TPAMI* 30(10):1699–1712
 3. Shizawa M, Mase K (1990) Simultaneous multiple optical flow estimation. In: *ICPR'90*, Atlantic City. IEEE Computer Society
 4. Bergen JR, Burt PJ, Hingorani R, Peleg S (1992) A three-frame algorithm for estimating two-component image motion. *IEEE TPAMI* 14(9):886–896
 5. Irani M, Rousso B, Peleg S (1992) Detecting and tracking multiple moving objects using temporal integration. In: European conference on computer vision (ECCV'92), Santa Margherita Liguere. Springer, pp 282–287
 6. Ju SX, Black MJ, Jepson AD (1996) Skin and bones: multi-layer, locally affine, optical flow and regularization with transparency. In: *IEEE conference on computer vision pattern recognition (CVPR'96)*, San Francisco. IEEE, pp 307–314
 7. Szeliski R, Golland P (1999) Stereo matching with transparency and matting. *IJCV* 32(1):45–61
 8. Szeliski R, Avidan S, Anandan P (2000) Layer extraction from multiple images containing reflections and transparency. In: *IEEE conference on computer vision pattern recognition (CVPR'2000)*, vol 1, Hilton Head Island. IEEE Computer Society, pp 246–253
 9. Tsai Y, Kang SB, Szeliski R (2006) Stereo matching with linear superposition of layers. *IEEE TPAMI* 28(2):290–301
 10. Sun J, Jia J, Tang CK, Shum HY (2004) Poisson matting. In: *SIGGRAPH 2004*, Los Angeles, SIGGRAPH: ACM
 11. Wexler Y, Fitzgibbon A, Zisserman A (2002) Bayesian estimation of layers from multiple images. In: European conference on computer vision (ECCV'02), Volume 2352 of Lecture Notes in Computer Science, Copenhagen. Springer, pp 487–501
 12. Sarel B, Irani M (2004) Separating transparent layers through layer information exchange. In: European conference on computer vision (ECCV'04), Prague. Springer
 13. Chen Y, Chang TC, Zhou C, Fang T (2009) Gradient domain layer separation under independent motion. In: *ICCV 2009*, Kyoto. IEEE
 14. Schechner Y, Kiryati N, Basri R (1998) Separation of transparent layers using focus. In: *ICCV'98*, Bombay. IEEE Computer Society, pp 1061–1066
 15. Schechner Y, Shamir J, Kiryati N (1999) Polarization-based decorrelation of transparent layers: the inclination angle of an invisible surface. In: *ICCV'99*, Kerkyra. IEEE Computer Society, pp 814–819
 16. Narasimhan SG, Nayar SK (2002) Vision and the atmosphere. *IJCV* 48(3):233–254
 17. He K, Sun J, Tang X (2009) Single image haze removal using dark channel prior. In: *IEEE conference on computer vision pattern recognition (CVPR'09)*, Miami. IEEE
 18. Farid H, Adelson E (1999) Separating reflections and lighting using independent components analysis. In: *IEEE conference on computer vision pattern recognition (CVPR'99)*, vol 1, Fort Collins. IEEE Computer Society, pp 262–267
 19. Schechner Y, Kiryati N, Shamir J (2000) Blind recovery of transparent and semireflected scenes. In: *IEEE conference on computer vision pattern recognition (CVPR'00)*, vol 2, Hilton Head Island. IEEE Computer Society, pp 38–43
 20. Bronstein A, Bronstein M, Zibulevsky M, Zeevi Y (2003) Blind separation of reflections usnig sparse ICA. In: 4th international symposium on independent component analysis and blind signal separation (ICA'03), Nara <http://www.kecl.ntt.co.jp/icl/signal/ica2003/>
 21. Levin A, Zomet A, Weiss Y (2002) Learning to perceive transparency from the statistics of natural scenes. In: *NIPS'02*, Vancouver. MIT
 22. Wang JYA, Adelson EH (1994) Representing moving images with layers. *IEEE Trans Image Process* 3(5): 625–638
 23. Darrell TJ, Pentland AP (1995) Cooperative robust estimation using layers of support. *IEEE TPAMI* 17(5):474–487
 24. Shizawa M, Mase K (1991) Principle of superposition: a common computational framework for analysis of multiple motion. *IEEE workshop on visual motion*, Princeton. IEEE, pp 164–172
 25. Swaminathan R, Kang S, Szeliski R, Criminisi A, Nayar S (2002) On the motion and appearance of specularities in

- image sequences. In: European conference on computer vision (ECCV'02), Volume 2350 of lecture notes in computer science, Copenhagen. Springer, pp 508–523
26. Levin A, Weiss Y, Durand F, Freeman WT (2009) Understanding and evaluating blind deconvolution algorithms. In: IEEE conference on computer vision pattern recognition (CVPR'09), Miami. IEEE
27. Zhu Y, Prummer S, Chen T, Ostermeier M, Comaniciu D (2009) Coronary DSA: enhancing coronary tree visibility through discriminative learning and robust motion estimation. In: SPIE medical imaging (2009). SPIE Lake Buena Vista (Orlando Area), Florida. <http://spie.org/x33859.xml>

Transportation Problem

►Many-to-Many Graph Matching

Trichromatic Theory

Michael H. Brill
Datacolor, Lawrenceville, NJ, USA

Related Concepts

►Chromaticity; ►Color Constancy

Definition

Trichromatic theory is the conceptual framework by which human vision matches the color of any test light to that of an additive mixture of the spectra of three primary lights in appropriate proportions. For highly chromatic colors, such a match might require one of the primary lights to mix with the test light instead of with the other primaries; the algebra of adding the light spectra would represent this condition as subtraction.

Background

The basic correctness of trichromatic theory (acknowledged at least since Thomas Young in 1802) underlies the success of such technologies as color photography and television, both of which need only three primaries to produce acceptable color rendition. The amounts of three primary lights needed to match a test light are called the *tristimulus values* of that test light, and are used to specify color quantitatively. Two lights with

different spectra and the same tristimulus values are called *metamers* (or *metameric matches*). One powerful consequence of trichromatic theory is that, if two lights are a metamer match, they can never be made to mismatch by any functional processing of the visual signal. Since illumination changes (or putting on sunglasses) may make certain objects match in color that did not match before (i.e., make the lights reflected from two objects a metamer match), that means that no visual processing is completely effective in returning the viewed world to what it would have looked like before the light changed or the sunglasses were put on. Even so, the visual system approximates this feat of restoration much of the time. The feat is called *color constancy*, and is described in another entry.

The biological basis of trichromatic theory is a set of three kinds of light sensors in the retina, called cones, each kind having a different type of light-sensitive molecule (photopigment). Whereas trichromatic theory was confirmed functionally by color-matching experiments in the nineteenth century, the physiological and anatomical confirmation did not occur until the mid-twentieth century.

Theory and Application

Underlying trichromatic theory is a set of rules known as Grassmann's laws, disclosed in 1855 by Hermann Grassmann. The laws are assertions about color matches that are more or less true empirically. In the statement of these rules below, “=” denotes a color match; k denotes a scalar multiplier; \mathbf{A} , \mathbf{B} , \mathbf{C} , \mathbf{D} are the spectra of lights that participate in a match; and “+” denotes superposition (addition) of the spectral power distributions of the lights. All of color technology, including color television and color photography, depends on these rules [1]:

Symmetry: If $\mathbf{A} = \mathbf{B}$, then $\mathbf{B} = \mathbf{A}$.

Transitivity: If $\mathbf{A} = \mathbf{B}$ and $\mathbf{B} = \mathbf{C}$, then $\mathbf{A} = \mathbf{C}$.

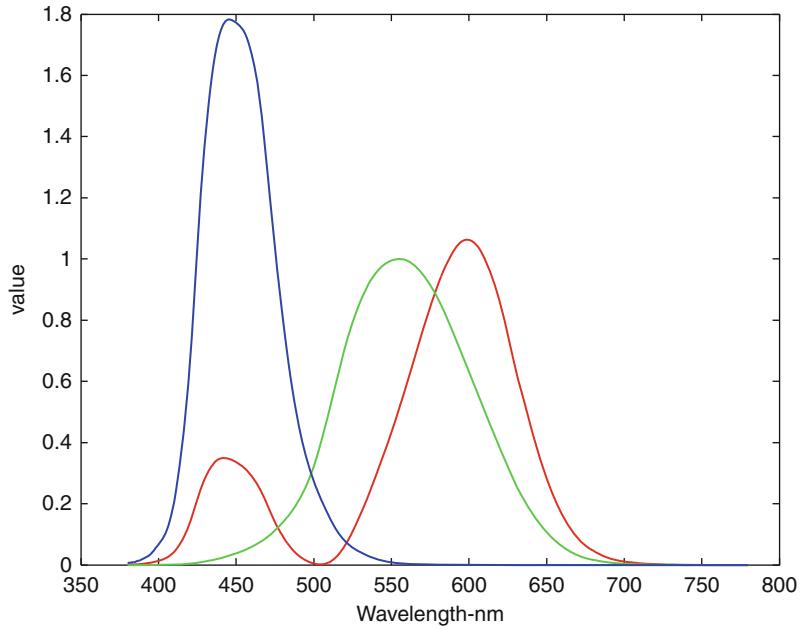
Proportionality: If $\mathbf{A} = \mathbf{B}$, then $k\mathbf{A} = k\mathbf{B}$.

Additivity: If $\mathbf{A} = \mathbf{B}$ and $\mathbf{C} = \mathbf{D}$, then $\mathbf{A} + \mathbf{C} = \mathbf{B} + \mathbf{D}$.
If $\mathbf{A} = \mathbf{B}$ and $\mathbf{A} + \mathbf{C} = \mathbf{B} + \mathbf{D}$, then $\mathbf{C} = \mathbf{D}$.

Grassmann's laws are tested by what is called a *symmetric-matching* experiment: An observer compares two lights that are presented on identical backgrounds and with a visual system adapted the same for both sides of the match.

Trichromatic Theory, Fig. 1

CIE 1931 color-matching functions



For such technologies as color television and photography, it is important to understand how to create the same color on an image as was in an original scene. Because of trichromatic theory and Grassmann's laws, it can be found out what spectra will match each other through a set of three functions of wavelength called the *color-matching functions*. Color-matching functions are specified by the Commission Internationale de l'Eclairage (CIE) based on a set of symmetric color-matching experiments performed in 1931 by several human observers whose data were combined to produce a Standard Observer. The functions so derived are called $\bar{x}(\lambda)$, $\bar{y}(\lambda)$, and $\bar{z}(\lambda)$, where λ is visible wavelength (from 380 to 780 nm). Two spectral power distributions, $S_1(\lambda)$ and $S_2(\lambda)$, are said to match and produce tristimulus values X, Y, and Z when

$$\begin{aligned} X &= \int \bar{x}(\lambda)S_1(\lambda)d\lambda = \int \bar{x}(\lambda)S_2(\lambda)d\lambda \\ Y &= \int \bar{y}(\lambda)S_1(\lambda)d\lambda = \int \bar{y}(\lambda)S_2(\lambda)d\lambda \\ Z &= \int \bar{z}(\lambda)S_1(\lambda)d\lambda = \int \bar{z}(\lambda)S_2(\lambda)d\lambda. \end{aligned}$$

Of course, if two lights match according to this definition, they will match using any linearly independent linear combinations of $\bar{x}(\lambda)$, $\bar{y}(\lambda)$, and $\bar{z}(\lambda)$.

Also, when one adds two spectral power distributions, one adds their vectors (X, Y, Z).

For illustration, Fig. 1 below shows the CIE 1931 color-matching functions $\bar{x}(\lambda)$ [in red], $\bar{y}(\lambda)$ [in green], and $\bar{z}(\lambda)$ [in blue].

Deriving the color-matching functions for a human observer requires doing a symmetric color-matching experiment. In practice there are two experimental methods.

The more common way, called the *maximum-saturation method*, arranges three narrowband primary lights and a variable-wavelength test light to form a match. The test light (assumed monochromatic) is presented with unit power, and the primary lights are adjusted in intensity so, for each wavelength of the test light, two of the primaries superposed will match the other primary superposed with the test light. The name "maximum saturation" derives from the fact that the chromaticity of the light on either side of the match is always on one of the sides of the triangle defined by the primaries. Color-matching functions derive from the maximum-saturation experiment in a straightforward way: The powers of the primaries required to make the match (including a negative sign for the primary on the same side of the match as the test light) are the color-matching functions at the test-light wavelength. Linearly transformed according to the mandates of the CIE [2], they become the \bar{x} , \bar{y} , and \bar{z} color-matching



functions above. (There are effects of field size, etc. in the experiment, but those go beyond the scope of this brief tutorial.)

The other, less common kind of color matching, the *Maxwell method*, uses a chosen white light (always the same) on one side of the match and variable powers of the test light and two out of three monochromatic primaries on the other side of the match. Color-matching functions derive from the Maxwell color match by a somewhat involved process (see [5]), and should (if Grassmann's laws were true) be the same functions as given by the maximum-saturation experiment.

Open Problems

Grassmann's laws are known not to be exactly true in human color matching [5, 6]. Besides the three cone types that herald the trichromacy of vision at high (photopic) light intensities, a fourth photoreceptor type (rods) contributes to vision at low (mesopic and scotopic) light intensities and away from the center of vision (fovea). At very high light intensities, unbleached photopigments deplete and, in aggregate, change their action spectrum. At still higher light intensities, a photopigment molecule can absorb multiple photons but respond as if it absorbed only one photon. These effects compromise Grassmann's laws, but the successful application of the laws, e.g., in photography and television, is evidence that the compromises are not serious.

Nonetheless, changing the way the matching experiment is done can alter the experimental results significantly. For example, as defined above, Maxwell and maximum-saturation color matches have long been known to be inconsistent even at high luminance levels and statistically reinforced by match replication [1]. A practical consequence of the failure

of additivity could be problems observed in cross-media color matching, although cross-media studies also have other well-known sources of imprecision because in that case the color matching is asymmetric.

References

- Wyszecki G, Stiles WS (1982) Color science: concepts and methods, quantitative data and formulae, 2nd edn. Wiley, New York, pp 117–248, 278–485
- Fairman HS, Brill MH, Hemmendinger H (1997) How the CIE 1931 color-matching functions were derived from Wright-Guild data. *Color Res Appl* 22(1):11–23
- Brill MH (2000) Understanding color matches: What are we taking for granted? In: Davis S (ed) *Color perception: philosophical, psychological, artistic, and computational perspectives*. Oxford University Press, New York, pp 141–151
- Fairchild MD (2005) *Color appearance models*, 2nd edn. Wiley, Chichester, pp 53–82
- Brill MH, Robertson AR (2007) Open problems on the validity of Grassmann's laws. In Schanda J (ed) *CIE colorimetry: understanding the CIE system*. Wiley, Hoboken, pp 243–257
- Commission Internationale de l'Eclairage (CIE) (2009) Reappraisal of colour matching and Grassmann's laws. Technical Report CIE 185:2009, Vienna: CIE
- Koenderink JJ (2010) *Color for the sciences*, Chapter 4. MIT Press, Cambridge
- Berns RS (2000) Billmeyer and Saltzman's principles of color technology, 3rd edn. Wiley, New York
- Hunt RWG (1998) *The measurement of colour*, 3rd edn. Fountain Press, Kingston-Upon-Thames
- Hunt RWG (2004) *The reproduction of colour*, 6th edn. Wiley, Chichester
- Wandell BA (1995) *Foundations of vision*. Sinauer, Sunderland

Two Dimensional Conditional Random Fields

- ▶ [Discriminative Random Fields](#)



U

Underwater Effects

Frank Michael Caimi
IEEE OES, Vero Beach, FL, USA

Synonyms

Inherent optical properties; Underwater light field;
Underwater radiative transfer processes

Related Concepts

► [Illumination Estimation, Illuminant Estimation](#);
► [Radiance](#)

Definition

Both scattering and absorption processes are dominant effects that determine the propagation of light in water and limit the ability to see or form images. These are characterized by inherent optical properties (IOPs) of the medium, which can be related to imaging theoretical parameters such as modulation transfer function (MTF) and point spread function (PSF). Underwater vision is affected by differences in the radiance distribution, which is an apparent optical property (AOP). Specially developed sensor systems are required to extend range over which visual information can be obtained.

Background

Observation and exploration of the underwater realm has been of interest since ancient times and has forged the development of many technological advances, for example:

- Conventional imaging methods that utilize high-sensitivity, dynamic range cameras and advanced light sources
- Extended range imaging techniques that utilize time gating or angular field restriction to reduce unwanted image noise resulting from optical scatter in the transmissive medium
- Laser line scan methods using “push broom” configurations to produce 2-D or 3-D imagery
- Holographic or other techniques that exploit spatial coherency over relatively short distances for high depth-of-field particle imaging
- Optical methods using frequency conversion (fluorescence, Raman, etc.) to develop higher-dimensional image data
- Multiple-perspective methods that achieve range depth or 3-dimensional (3-D) image formation and display (including mosaics)
- Image intensity algorithms based on image processing techniques that exploit effects such as shading, filtering, or motion
- Electromagnetic (EM) methods that utilize transparent regions of the spectrum (near infrared, ultraviolet, visible, and very low frequency (VLF))
- Acoustic imaging techniques that include 3-D reconstruction and mosaicking

- Temporally discriminant or signal-coded methods that support multidimensional image representations

Detailed descriptions of many of these preceding methods have been published elsewhere [1–3] where an emphasis is placed on extended range imaging.

Historically, advancements in underwater imaging have been slow to evolve and have taken decades to reach the present state of the art. This has been due, in part, to limitations stemming from incomplete understanding of radiative transfer in seawater, limitations in image sensor electronic and optical hardware, and computational restrictions in real-time data transfer, recording, and signal processing. Underwater vision in the optical spectrum is limited by absorption and scattering processes that are governed not only by the properties of water but also by dissolved and suspended substances. These result in limited visibility over very short distances in harbor and coastal regions. Imaging with conventional cameras beyond several optical attenuation lengths (c_z) is generally not practical due to so-called “veiling luminance” stemming from scattering centers in the medium that produce photon flux well in excess of that arriving from the scene or object plane. Therefore, the image content is not retrievable even with image stretching unless extremely high dynamic range sensors are employed. More advanced imaging systems using pulsed laser illuminators, gated detectors, narrow FOV synchronous scanners, etc. have been demonstrated, resulting in dramatic increases in the image resolution and standoff distances achievable over conventional cameras and illumination systems [1]. Hardware advances in lasers, video technology, electronic ballasting of HMI lighting, detectors, mechanical scanners, and processing algorithms have benefitted both optical and acoustic sensors systems [4]. Although these advances are often associated with imaging hardware and methodology, credit must also be given to developments in computer vision and pattern recognition that allow extraction or interpretation of useful information from raw imagery or data. This focus has been noted in a recent publication written for the computer vision community that underscores the difficulties and challenges of underwater imaging [5].

Theory

Inherent Optical Properties (IOPs)

Basic physical processes affecting underwater light propagation are absorption [1] and both elastic [2] and inelastic [3] scattering. These processes are partially characterized in terms of the inherent optical properties [6, 7], which are dependent only on the material properties of the medium or more accurately are invariant under changes of the light field radiance distribution. The primary IOPs, absorption coefficient a , scattering coefficient b , and beam attenuation coefficient c at optical wavelength λ are related by

$$\begin{aligned} c(\lambda) &= a(\lambda) + b(\lambda) \\ a(\lambda) &= a_{\text{water}}(\lambda) + a_{\text{naph}}(\lambda) + a_{\text{phy}}(\lambda) + a_{\text{CDOM}}(\lambda) \end{aligned} \quad (1)$$

The absorption coefficient may be decomposed into several additive components due to inherent absorption of water, suspended non-algal (minerogenic) particles, phytoplankton, and dissolved organic material (CDOM). The measured absorption spectrum of water from various publications is shown in Fig. 1. A transparency window is clearly observed in the blue-green spectral range.

A collimated light beam or plane wave of small cross-section A relative to c^{-1} and intensity I is therefore attenuated according to

$$I(z + \Delta z, \lambda) = I_0(z, \lambda) \exp[-c(\lambda) \Delta z] \quad (2)$$

where $c\Delta z \ll 1$. The scattering coefficient b is defined by

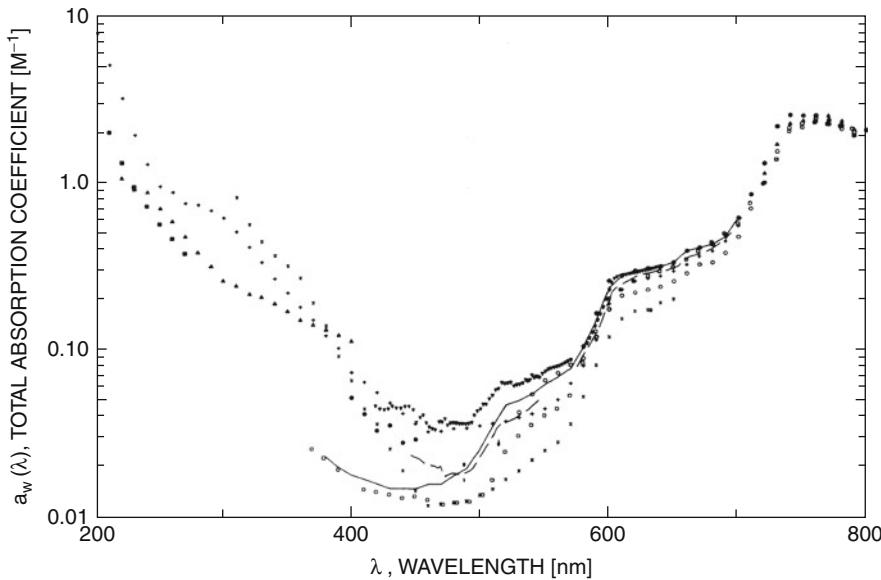
$$b(\lambda) = 2\pi \int_0^\pi \beta(\theta, \lambda) \sin(\theta) d\theta \quad (3)$$

where the volume scattering function is defined by

$$\beta(\theta, \lambda) \equiv |\Delta J(\theta, \lambda)| / I_0(\lambda) A \Delta z \quad (4)$$

and it is assumed that the irradiance ΔJ is scattered in direction θ . The volume scattering function β normalized to b is called the scattering phase function $p(\theta, \lambda)$. Measurements of phase functions for various ocean waters have been measured extensively [9].

Phase functions for open-ocean, coastal, and harbor waters are remarkably similar in shape, leading



Underwater Effects, Fig. 1 Measured absorption spectra of pure freshwater (various authors) [8]

to the development of an average particle phase function derived by subtraction of the volume scattering function for clear water [10, 11]. Although analytic forms such as Henyey-Greenstein [12] with $g \sim 0.95$ have been used, higher accuracy forms are preferred [13, 14].

The relationship of β or p to the modulation transfer function (MTF) has been described [15], and point spread function measurements have been conducted at small angles [16–20].

The parameters a , b , and c are measured by modern optical instruments and have been synoptically characterized for many natural waters [21–23].

The MTF and PSF are related by the Hankel transform pair

$$\begin{aligned} MTF(\Psi, z, \lambda) &= 2\pi \int J_0(2\pi\theta\Psi) PSF(\theta, z, \lambda) \theta d\theta \\ PSF(\theta, z, \lambda) &= 2\pi \int J_0(2\pi\theta\Psi) MTF(\Psi, z, \lambda) \Psi d\Psi \end{aligned} \quad (5)$$

The relationship to the IOPs has been derived [24] and for a non-stratified medium is given by

$$\begin{aligned} MTF(\Psi, z, \lambda) &= \exp\left[-\int_0^R c(z) dz + \int_0^R b P(\Psi, \lambda)\right] \\ P(\Psi, \lambda) &\equiv 2\pi \int J_0(2\pi\theta\Psi) p(\theta, \lambda) \theta d\theta \end{aligned} \quad (6)$$

where $p(\theta, \lambda)$ is the single scattering phase function and no z dependence of b or c is presumed, yielding

$$MTF(\Psi, R, \lambda) = \exp\left[-cR + b \int_0^R P\left(\frac{\Psi r}{R}, \lambda\right) dr\right] \quad (7)$$

Equations 5–7 link the MTF, PSF, and IOP parameters. The single scattering phase function p may be approximated analytically, and it is found that at multiple scattering lengths ($bR > 2$), the PSF is relatively insensitive to its functional form [25].

Apparent Optical Properties (AOPs)

Understanding of the radiative transfer process is paramount in designing sensors for derivation of information such as quantity, distribution, and types of suspended materials; presence and identification of manmade objects; as well as the behavior of optical imaging and communications systems. Much of this information is obtained through the radiance distribution L , which is an apparent optical property and is described by the radiative transfer equation (RTE):

$$\begin{aligned} & \left[\cos \theta \frac{d}{dz} + c(z) \right] L(z, \theta, \phi) \\ &= b(z) \int_{\Omega'} p(z; \theta; \theta', \phi') L(z, \theta', \phi') d\Omega' \quad (8) \end{aligned}$$

Perhaps the simplest solution of the RTE is known as Gershun's law [37], and relates the absorption coefficient to the derivative of the difference of the downwelling E_d and upwelling E_u irradiance when no embedded light sources are present:

$$\frac{d}{dz} [E_d(z, \lambda) - E_u(z, \lambda)] = -a(z, \lambda) E_0(z, \lambda) \quad (9)$$

Solving the general integrodifferential equation for radiance given IOPs and boundary conditions is difficult, but analytic solutions do exist for special cases, such as for non-scattering waters. Sophisticated numerical methods therefore must be employed to solve the RTE for realistic oceanic conditions.

This definition assumes that L is a scalar function of position z , polar angle θ , and azimuth angle ϕ and neglects polarizing properties of the medium. Inclusion of polarizing properties of the medium requires matrix formalism using a Stokes vector notation for the RTE [26]. In the presence of Raman or fluorescent inelastic scattering, wavelength dependence must also be included.

Estimation of the radiance distribution can be accomplished using a Monte Carlo approach, but the computational burden associated with most environments requires parallel processing and large memory resources for solution of practical problems. Consequently, analytical models have been advanced to predict radiance performance for different IOPs and geometric configurations [27, 35, 36] for both imaging and remote sensing applications. Recently developed time-dependent solutions of the RTE for pulsed sources [27] are practical for communications and LIDAR or other pulsed imaging systems. Pulsed imaging systems can discriminate against scattered light by gating the detector (LIDAR) or by synchronously scanning a small divergence source and narrow field-of-view detector (so-called laser line scan systems). In addition, an understanding of the impulse response of

the medium and its variability is necessary in developing and optimizing any digital or analog communications system.

Remote Sensing

Inversion of the radiance distribution is the basis for remote sensing of water constituents as the IOPs are determined in part by spectral absorption characteristics of various absorbing species of phytoplankton and dissolved materials. Solid objects may also be detected. *Passive remote sensing* involves observation light emanating from the ocean surface due to solar illumination consisting of surface-reflected light and upwelling light scattered toward the surface from particles in water column and from bottom reflection. Examples of *passive remote sensing systems* [28] are aircraft- or satellite-borne color sensors which exploit the wavelength properties of the absorption coefficient associated with various dissolved or suspended substances, such as of chlorophyll, CDOM, or mineral particles under solar illumination. The ocean also can contain light sources such as bioluminescent organisms, which may be observed at night using high-sensitivity detectors. Phytoplankton types and physiological state may be estimated as well as the bottom depth and type in very shallow waters.

Active remote sensing system using lasers, such as LIDAR, uses pulsed illumination to estimate coastal depths or discernable layers of suspended substances. Since laser light is reflected from the sea surface and subsequently from the subsurface layer, the layer depth is easily calculated if the pulse is physically much shorter than the depth to be measured. The depth is given approximately by

$$d = (c/n) \Delta t_{rt}/2 \quad (10)$$

where Δt_{rt} is the time difference between surface return and the seafloor return, c is the speed of light, and n is the index of refraction for water (approximately 1.33).

Another active remote sensing method, *laser-induced fluorosensing* uses pulsed ultraviolet or short wavelength blue laser sources to detect and estimate the concentration of layered chlorophyll containing constituents and pollutants. *Thermal sensing* vertically



through the water column is also possible using comparisons of relative Raman Stokes and anti-Stokes emission spectrally adjacent to the excitation wavelength [34].

A large number of remote sensor systems have been deployed. Well known in the USA and Canada are the coastal zone color scanner launched in 1978 [29], the sea-viewing wide field-of-view sensor (SeaWiFS) launched in 1997 [30], the moderate resolution imaging spectroradiometer (MODIS) launched 1999 and 2002 [31], and the compact airborne hyperspectral imager (CASI) [32, 33]. Geographic fidelity of remote sensor systems is typically limited by spatial resolution (meters typical for satellite and sub-meter for airborne) as well as the time required for collecting data over a large area.

Open Problems: Remote Sensing

Many remote sensing platforms exist, and great strides have been made in ocean color remote sensing. However, remote sensing problems generally require inverting the radiance distribution to obtain constituents affecting IOPs. This class of inverse problems can be based on either *explicit* or *implicit* solutions of the RTE using either active or passive illumination. Explicit solutions rely on functional dependence between the radiance distribution $L(z, \theta, \phi, \lambda)$ and the set of IOPs. Determining the absorption coefficient $a(\lambda)$ from a series of measured irradiances, as in Eq. (9), is one example of an explicit solution for an IOP given irradiance measurements. *Implicit solutions* are obtained by solving a sequence of direct or forward problems for different sets of IOPs. The set providing the best fit radiance distribution is then chosen as representing the inverse solution.

Since radiation measurements are not exact and because of overlapping spectra, time-dependent illumination during measurement, inadequate spatial, or wavelength sampling, the inversion process can be plagued with various inaccuracies. These include model sensitivity to slight changes in the radiance distribution, model uniqueness errors, radiometer calibration error, and other noise effects. Forward models may be simple based on observed radiance distributions in a geographic location, and these dependencies may not be universally unique. Inversions are usually

based on an assumed model that relates known quantities to a desired output. The inversion takes place using the known quantities as inputs to the model, whose output is an estimate of the desired quantities. Due to costs and geographic scales involved in making measurements to establish a model, constraints are imposed to limit the range of solutions possible from a given set of data. Constraints can be internal to the RTE or external, typically as additional required measurements (such as a measurement of the water leaving radiance or bottom depth at one point in an image). Implicit constraints may also be imposed, such as a limitation of retrieved values to the range of values found in the data set used to pre-determine certain parameters in the inverse model.

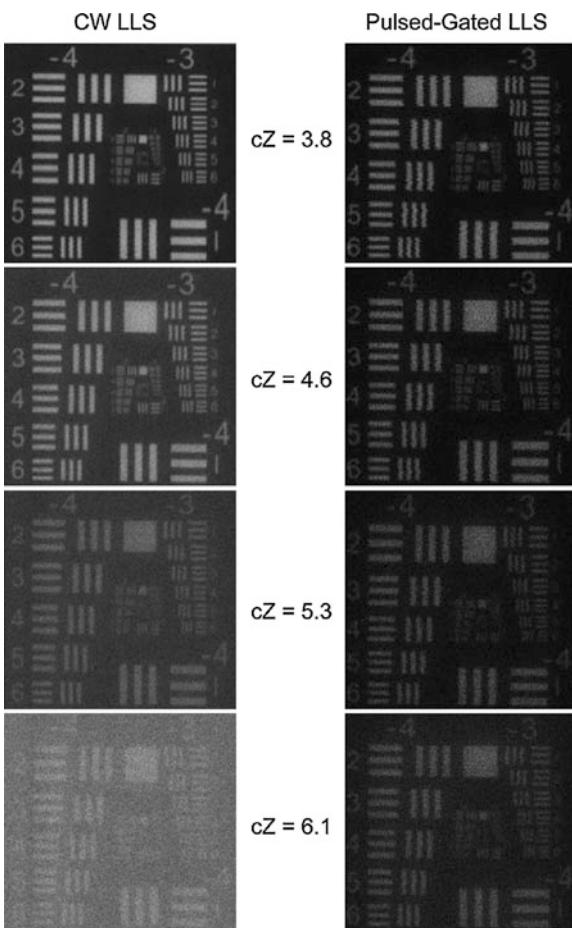
Open Problems: Polarization

New trends in underwater imaging have also involved understanding the complex behavior of polarized light fields and remote sensing of optical properties for object detection, biological monitoring, and pollution identification/tracking. The RTE as posed in Eq. (8) neglects polarization. However, scattering within the water column, refraction, reflection and transmission by the air-water surface, and reflection by submerged objects induces polarization even if the incident beam is not polarized. Objects exhibit a complex reflection distribution – the so-called bidirectional reflectance distribution or BRDF (http://en.wikipedia.org/wiki/Bidirectional_reflectance_distribution_function) rather than a single-valued parameter. Thus, accurate estimation of the underwater radiance in a particular direction will depend on polarization and BRDF from the bottom or submerged objects. Understanding the complexity of these effects on the vector radiance through the vector form of the RTE is still an open problem. Multiple scattering events, particle size distributions larger than the wavelength of light, and spatial averaging tend to negate the effects of polarization. Numerous studies have been made of individual particles of different type, shape, and refractive index with characterization of scattering and its effect on polarization. Understanding the relationship of these on the radiation distribution and the inverse problem also remains a difficult problem.



Experimental Results in Imaging

Recent experimentation in underwater imaging has been focused on obtaining images at greater optical distance (in terms of attenuation lengths cZ) with either natural [38] or artificial [39] illumination. The former uses polarization characteristics of the light field for enhancement, while the latter has relied on various methods such as pulsed illumination with restricted FOV using quasi-monostatic scanners. Continued improvements are being made in image quality as shown in Fig. 2 [40]. Image formation at over 20 receiver-to-target attenuation lengths has been demonstrated with bistatic imagers with the source-to-target



Underwater Effects, Fig. 2 Demonstrated image improvement with pulsed-gated laser line scan imager (PLLS) versus continuous-wave laser line scan systems (LLS) at multiple attenuation lengths (cZ) [40]

distance being the limiting factor due to point spread of the source illumination, and new experimental systems are being developed.

References

- Kocak DM, Caimi FM (2005) The current art of underwater imaging-with a Glimpse of the past and vision of the future. *Mar Tech Soc J* 39(3):5–26
- Kocak DM, Caimi FM (2001) Computer vision in ocean engineering. In: El-Hawary F (ed) *Ocean engineering handbook*. CRC, Boca Raton, pp 20–43
- Jaffe JS, McLean J, Strand MP, Moore KD (2001) Underwater optical imaging: status and prospects. *Oceanography* 14(3):66–76
- Olsson M (1999) Undersea imaging. In: MTSJ, State of the technology report – advanced marine technology division. Marine Tech. Soc. Journal, Jaeger J, ed., 33(3): 103–104
- Murino V, Trucco A (2000a) Underwater computer vision and pattern recognition. *Comput Vis Image Underst* 79(1):1–3
- Preisendorfer R (1976) *Hydrologic optics vol. 1, introduction*. NTISPB-259 793/8ST, Springfield
- Jerlov NG (1968) *Optical Oceanography*. Elsevier, Amsterdam, London, New York
- Smith RC, Baker KS (1996) Optical properties of the clearest natural waters (200–800 nm). *Appl Opt* 20(2):177–184. In: Caimi FM (ed) (1981) *Underwater optics*. SPIE Milestone Series, vol MS-118
- Petzold TJ (1972) Volume scattering functions for selected ocean waters. *Scripts Inst. Oceanogr., Visibility Laboratory Report SIO* 72–78
- Mobley CD, Gentili B, Gordon Jin Z, Kattawar GW, Morel A, Reinersman P, Starnes K, Stavn R (1993) Comparison of numerical models for the computation of underwater light fields. *Appl Opt* 32(36):7484–7504
- Mobley CD, Sundman LK, Boss E (2002) Phase function effects on oceanic light fields. *Appl Opt* 41(6): 1035–1050
- Henyey LC Greenstein JL (1941) Diffuse radiation in the galaxy. *Astrophys J* 93:70–83
- Fournier G, Jonas M (1999) Computer based underwater imaging analysis. In: Gilbert G (ed) *Airborne and in-water underwater imaging*. SPIE, vol 3761. SPIE, Bellingham, pp 62–77 (with corrections)
- Kattawar GW (1975) A three-parameter analytic phase function for multiple scattering calculations. *J Quant Spectrosc Radiat Trans* 15:839–849
- Hodara H (1973) AGARD lecture series 61 on optics of the Sea. Neuilly Sur Seine, FRANCE, NATO
- Wells W, Hodara H, Wilson O (1972) Long range vision in seawater. Final Report ARPA order 1737. TetraTech Inc., Pasadena
- Honey RC (1979) Beam spread and point spread functions and their measurement in the ocean. *Proc Soc Photo-Opt Instrum Eng* 208:242–248



18. Mertens LE, Repleglo LE Jr (1977) Use of point spread and beam spread functions for analysis of imaging systems in water. *J Opt Soc Am* 67:1105–1117
19. McLean JW, Crawford DR, Hindman CL (1987) Limits of small angle scattering theory. *Appl Opt* 26:2053–2054
20. Voss KJ Chapin AL (1990) Measurement of the point spread function in the ocean. *Appl Opt* 29:3638–3642
21. Caimi FM (ed) (1996) Underwater optics. SPIE milestone series, pub Soc. Photo-Optical Engrs.vol MS-118.
22. Hale M, Querry MR (1973) Optical constants of water in the 200-nm–200- μ m wavelength region. *Appl Opt* 12:555–563
23. Smith RC, Baker KS (1981) Optical properties of the clearest natural waters (200–800 nm). *Appl Opt* 20(2):177–184
24. Wells WH (1973) Theory of small angle scattering. In: Optics of the sea. AGARD lecture series, No. 61 (NATO) Paris, NTIS Publication
25. McLean JW, Voss K (1991) Point spread function in ocean water: comparison between theory and experiment. *Appl Opt* 30(15):2027–2030
26. http://www.oceanopticsbook.info/view/radiative_transfer_theory/level_2/the_vector_radiative_transfer_equation
27. Giddings TE, Shirron JJ (2009) Numerical simulation of the incoherent electrooptical imaging process in plane-stratified media. *Opt Eng* 48(12):126001
28. Measures RM (1992) Laser remote sensing: fundamentals and applications. Krieger, Malabar
29. <http://oceancolor.gsfc.nasa.gov/CZCS/>
30. <http://oceancolor.gsfc.nasa.gov/SeaWiFS/TEACHERS/INTRO/>
31. <http://modis.gsfc.nasa.gov/data/>
32. <http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&id.Identifier=ADA495178>
33. Terrie, Gregory E (1995) Applications of hyperspectral data in coastal marine environments. Report Number AD-A302222, NRL/FR/7442-95-9630
34. Leonard DA, Caputo B, Hoge FE (1979) Remote sensing of subsurface water temperature by raman scattering. *Appl Opt* 18(11):1732–1745
35. Mobley CD (1994) Light and water: radiative transfer in natural waters, Academic. (Out of print, but can be downloaded as a pdf document or obtained on CD from Curtis Mobley)
36. Mobley CD, Sundman LK (2008) HydroLight 5 – EcoLight 5 technical documentation (2008). Sequoia Scientific, Inc. Download pdf
37. Gershun AA (1939) The light field. *J Math Phys* 18(2): 51–151
38. Schechner YY, Karpel N (2004) Clear underwater vision. In: IEEE CVPR 2004, Conference on Computer Vision and Pattern Recognition, Washington DC, 27th June–2nd July, vol 1, pp 536–543
39. Laux A, Mullen LJ, Cochenour B (2008) A comparison of extended range laser line scan imaging techniques in turbid underwater environments. In: Proceedings of ocean optics XIX, Barga, Italy
40. Dalgleish FR, Caimi FM, Britton W, Andren CF (2009) Improved LLS imaging performance in scattering-dominant waters. In: Proceeding of SPIE, vol 7317, Ocean Sensing and Monitoring, Orlando, FL

Underwater Light Field

► Underwater Effects

Underwater Radiative Transfer Processes

► Underwater Effects

User-Assisted

► Interactive Segmentation

User-Guided

► Interactive Segmentation



V

Variable Selection

► [Feature Selection](#)

Variational Analysis

Bastian Goldluecke

Department of Computer Science, Technische Universität München, München, Germany

Synonyms

[Calculus of variations](#)

Related Concepts

► [Total Variation](#); ► [Variational Method](#)

Definition

In mathematics, the term *variational analysis* usually denotes the combination and extension of methods from *convex optimization* and the classical *calculus of variations* to a more general theory [5]. However, in computer vision literature, the term is frequently encountered as just a synonym for *calculus of variations*. This branch of mathematics deals with the minimization of functionals, which are real-valued functions on infinite-dimensional spaces, most frequently spaces of functions.

Background

In the continuous world view, images are modeled as functions on a domain $\Omega \subset \mathbb{R}^n$. Geometric entities like curves and surfaces are manifolds, which can be described as level sets of functions or by the characteristic functions of their interior region. Consequently, computer vision problems can often successfully be formulated as minimization problems on infinite-dimensional spaces, where the solution is given as the minimizer of an *energy functional* $E: \mathcal{V} \rightarrow \mathbb{R}$ on the space \mathcal{V} of admissible functions. The minimization of such a functional thus requires a calculus on infinite-dimensional spaces, which is provided by the classical branch of mathematics called *calculus of variations* [3, 4]. It provides necessary conditions which have to be satisfied by the minimizing function, probably most well known is the *Euler-Lagrange equation*, a partial differential equation for the unknown. However, the methods of the calculus of variations usually provide only conditions for local minima, while it is of course desirable to formulate models where one has some guarantees about the optimality of the solution.

Recently, there has therefore been much effort to formulate computer vision in a way that the final minimization problem is a *convex functional*. The main advantage is that there are no local minima in the sense that each local minimizer is automatically a global optimum. Furthermore, for optimization, it is possible to enhance the techniques from the calculus of variations with methods from *convex optimization*, thus obtaining very efficient minimization algorithms [1, 4, 5].

The formulation of variational models as convex problems requires convex regularizers. A particularly powerful regularizer was found to be the *total variation* of a function [2]. Besides convexity, it has certain interesting geometric properties which make it possible to obtain convex formulation of segmentation and minimal surface problems. See the entry on *total variation* for more details.

Theory

See the entry on the *variational method*.

References

- Attouch H, Buttazzo G, Michaille G (2006) Variational analysis in Sobolev and BV spaces. MPS-SIAM series on optimization. Society for Industrial and Applied Mathematics, Philadelphia
- Chambolle A, Caselles V, Cremers D, Novaga M, Pock T (2010) An introduction to total variation for image analysis. *Radon Ser Comput Appl Math* 9:263–340
- Gelfand IM, Fomin SV (2003) Calculus of variations. Dover publications reprint of the 1963 edn. Dover Publications Inc., Mineola, NY
- Luenberger D (1969) Optimization by vector space methods. Wiley, New York
- Rockafellar RT, Wets R (2005) Variational analysis. Springer, New York

Variational Method

Bastian Goldluecke

Department of Computer Science, Technische Universität München, München, Germany

Related Concepts

► [Total Variation](#); ► [Variational Analysis](#)

Definition

The variational method is a way to solve problems given in the form of a *variational model*, i.e., as an energy minimization problem on an infinite-dimensional space which is typically a

function space. It employs tools from the mathematical framework of *variational analysis*.

Background

Ikeuchi and Horn's shape-from-shading paper and Horn and Schunck's optical flow paper, which appeared in AIJ simultaneously, are the earliest representative works to introduce a variational method to computer vision [7]. Inspired by the extraordinary success of the idea, variational methods have been extensively studied in computer vision and become a very popular tool for a wide variety of problems. They are particularly successful in mathematical image processing, where they are used to describe fundamental low-level problems, like image segmentation [9, 11], denoising [15], and deblurring [3], but have also been employed for high-level tasks like 3D reconstruction [4, 10].

In a variational model, the solution to a problem is obtained as the minimizer of an energy defined on a typically infinite-dimensional space, for example, a function space. The underlying world view is a continuous one similar to classical physics, where images are regarded as functions on \mathbb{R}^n , while curves and surfaces, encountered, for instance, in image segmentation and 3D reconstruction, are regarded as manifolds.

In view of this, it is not surprising that the mathematical framework of *variational analysis* used to deal with these problems was developed in close conjunction with new insights in physics in the early twentieth century [5]. It is a calculus for functionals on infinite-dimensional spaces, which together with tools from differential geometry and measure theory yields the Euler-Lagrange equations of such a functional as a necessary condition for a minimum. Commonly, solutions to the Euler-Lagrange equations are obtained either by directly solving a discretization or via a gradient descent technique.

Usually, only a local minimum is obtained this way. Recently, however, much effort has been devoted to formulate variational problems with convex energies, either directly or using relaxation techniques. In the case of a convex energy, the solution to the Euler-Lagrange equation yields a global optimum. Furthermore, the methods from variational analysis can be enhanced with methods from convex analysis and



convex optimization, yielding powerful and efficient optimization methods.

Theory

There are two classical types of variational problems, which require a slightly different mathematical apparatus. The first class is concerned with the minimizer of a functional which is defined on a (usually infinite-dimensional) vector space \mathcal{V} , for example, a function space. It typically arises when the goal is to recover images, as in the problems of image denoising and deblurring.

The second type is more complex and deals with weighted minimal surfaces, i.e., minimizers of a class of functionals defined for surfaces. This type of problem arises when one tries to recover manifolds, like curves in image segmentation or surfaces in 3D reconstruction. It can be related to the first case by introducing a variation of a surface and employing methods from differential geometry.

Functionals on vector spaces The foundation for the variational method for functionals on vector spaces is the *variational principle*, which states that a minimum of a functional is a *stationary point* [2, 5]. Let $E : \mathcal{V} \rightarrow \mathbb{R}$ be a functional on the Banach space \mathcal{V} and then a point $u \in \mathcal{V}$ is called stationary if the *Gâteaux derivatives* $\delta E(u; h)$ in all directions h vanish, i.e.,

$$0 = \delta E(u; h) := \lim_{\alpha \rightarrow 0} \frac{1}{\alpha} (E(u + \alpha h) - E(u)). \quad (1)$$

Typically, the functional E is given as an integral over a domain Ω . In order to apply the variational principle in this case, one writes the Gâteaux differential in the form

$$\delta E(u; h) = \int_{\Omega} \phi_u \cdot h \, dx, \quad (2)$$

where ϕ_u is a function on Ω which depends on u .

The *du Bois-Reymond Lemma* then implies that if $\delta E(u; h) = 0$ for all h , then in fact $\phi_u = 0$ on Ω . This is a partial differential equation for the unknown u and a necessary condition which has to be satisfied by a minimizer. It is called the *Euler-Lagrange equation* of the functional E .

A frequent form for the energy E is the formulation

$$E(u) = \int_{\Omega} \mathcal{L}(u(x), \nabla u(x), x) \, dx, \quad (3)$$

with a *Lagrangian* function \mathcal{L} . In this case, an explicit formula for the Euler-Lagrange equation can be derived using the chain rule and theorem of Gauss for integration by parts. It reads

$$\frac{\partial \mathcal{L}}{\partial u} - \operatorname{div} \left(\frac{\partial \mathcal{L}}{\partial (\nabla u)} \right) = 0, \quad (4)$$

with either Neumann boundary conditions on u or Dirichlet boundary conditions on possible directions h .

Minimal surfaces. A minimal surface $\Sigma \subset \mathbb{R}^n$ is a local minimizer of the surface area integral

$$\mathcal{A}(\Sigma) := \int_{\Sigma} \Phi \, dA, \quad (5)$$

with a real-valued weight function $\Phi \geq 0$ and dA denoting the surface area element. The dimension of the surface is one less than the embedding space. Problems of this form can be handled by introducing a *variation of the surface* as a differentiable map

$$X : \Sigma \times (-\epsilon, \epsilon) \rightarrow \mathbb{R}^n, \quad (6)$$

where $X(\Sigma, 0) = \Sigma$ and a one-parameter family $\Sigma_{\tau} = X(\Sigma, \tau)$ of regular surfaces is given depending on $\tau \in \mathbb{R}$. The variational principle then implies that a minimal surface satisfies

$$\left. \frac{d}{d\tau} \right|_{\tau=0} \mathcal{A}(\Sigma_{\tau}) = 0 \quad (7)$$

for all possible variations. In [6], the Euler-Lagrange equation was derived in arbitrary dimension using methods from Cartan geometry [16], for the general case that Φ depends on the location s on the surface and the local normal \mathbf{n} . For a more elementary deduction for two-dimensional surfaces, see, for example, [4, 14], where [14] also deals with other classes of minimal surfaces which minimize weighted mean or Gaussian curvature.

The main result of [6] is that a minimal surface necessarily satisfies

$$(\Phi_s, \mathbf{n}) - \operatorname{Tr}(\mathbf{S}) \Phi + \operatorname{div}_{\Sigma}(\Phi \mathbf{n}) = 0, \quad (8)$$

where $\operatorname{Tr}(\mathbf{S})$ is the trace of the shape operator of the surface, also known as the Weingarten map or second fundamental tensor. Using either explicit surface evolution methods or the *level set method* [13], a



local minimum can be obtained via gradient descent as a stationary solution of a corresponding evolution equation.

In a more modern framework, the weighted surface area is replaced with the *weighted total variation* [1, 2] of the characteristic function of the surface interior. Optimization then takes place over the set of binary functions of bounded variation, which means that the difficult minimal surface problem has been reduced to the more simple first case discussed above. Using convex relaxation techniques, this allows to solve certain classes of weighted minimal surface problems, which, for instance, arise in segmentation [12] and 3D reconstruction [10], in a globally optimal manner. The entry on *total variation* provides more details on this topic.

References

1. Ambrosio L, Fusco N, Pallara D (2000) Functions of bounded variation and free discontinuity problems. Oxford University Press, Oxford/New York
2. Attouch H, Buttazzo G, Michaille G (2006) Variational analysis in Sobolev and BV spaces. MPS-SIAM series on optimization. Society for Industrial and Applied Mathematics, Philadelphia
3. Chan T, Wong C (1998) Total variation blind deconvolution. *IEEE Trans Image Process* 7(3):370–375
4. Faugeras O, Keriven R (1998) Variational principles, surface evolution, PDE's, level set methods, and the stereo problem. *IEEE Trans Image Process* 7(3):336–344
5. Gelfand IM, Fomin SV (2003) Calculus of variations. Dover publications reprint of the 1963 edn. Dover Publications Inc., Mineola, NY
6. Goldluecke B, Ihrke I, Linz C, Magnor M (2007) Weighted minimal hypersurface reconstruction. *IEEE Trans Pattern Anal Mach Intell* 29(7):1194–1208
7. Ikeuchi K, Horn BKP (1981) Numerical shape from shading and occluding boundaries. *Artif Intell* 17: 141–184
8. Horn B, Schunck B (1981) Determining optical flow. *Artif Intell* 17:185–203
9. Kass M, Witkin A, Terzopoulos D (1987) Snakes: active contour models. *Int J Comput Vis* 1:321–331
10. Kolev K, Klodt M, Brox T, Cremers D (2009) Continuous global optimization in multiview 3D reconstruction. *Int J Comput Vis* 84(1):80–96
11. Mumford D, Shah J (1989) Optimal approximation by piecewise smooth functions and associated variational problems. *Commun Pure Appl Math* 42:577–685
12. Nikolova M, Esedoglu S, Chan T (2006) Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM J Appl Math* 66(5): 1632–1648
13. Osher S, Fedkiw R (2003) Level set methods and dynamic implicit surfaces. Volume 153 of applied mathematical sciences. Springer, New York
14. Overgaard N, Solem J (2007) The variational origin of motion by gaussian curvature. In: Proceedings of the 1st international conference on Scale space and variational methods in computer vision (SSVM)
15. Rudin LI, Osher S, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. *Physica D* 60(1–4):259–268
16. Sharpe RW (1997) Differential geometry. Volume 166 of graduate texts in mathematics. Springer, New York

Variational Methods

- [Geodesics, Distance Maps, and Curve Evolution](#)

Veiling Glare

- [Lens Flare and Lens Glare](#)

Velvety Reflectance

- [Asperity Scattering](#)

Video Alignment and Stitching

- [Video Mosaicing](#)

Video Mosaicing

Zhigang Zhu

Computer Science Department, The City College of New York, New York, NY, USA

Synonyms

[Panoramic image generation](#); [Video alignment and stitching](#); [Video mosaicking](#)

Related Concepts

- [Image Registration](#)



Definition

Image mosaicing is the process of generating a composite image (mosaic) from a video sequence, or in general from a set of overlapping images of a scene or an object, usually resulting in a mosaic image with a larger field of view than any of the original images.

Background

When collecting video of a scene or object, each individual image in the video may be limited compared to the desired final product, including limitations in the field of view, dynamic range, or image resolution. This is the case not only with personal video capture [1, 9, 13] but also with image-based rendering [11, 14, 15], aerial videography [7, 10, 18–20], and document digitization [5]. Generating mosaics with larger fields of view [5, 6, 9, 13, 14, 19], higher dynamic ranges [4], and/or higher image resolutions [8] facilitates video viewing, video understanding, video transmission, and archiving. When the major objective of video mosaicing is to generate a complete (e.g., 360°) view of an object (or a scene) by aligning and blending a set of overlapping images, the resulting image is also called a video panorama [9, 14, 15].

Theory and Application

Video mosaicing takes in a video sequence and generates one or more mosaiced images with either a larger field of view, a higher dynamic range, a higher image resolution, or a combination of them. This entry will mainly discuss the principles in generating large field of view mosaics (panoramas), but the principles can also be (mostly) applied to mosaics for other objectives (high dynamic range imaging and super-resolution imaging). Here, *video* mosaicing implies that the images in the sequence are taken by a video camera, usually at 30 frames per second, but images taken by a digital camera such that there is a large amount of spatial overlap between two consecutive frames can also be viewed as a video sequence.

There are three key components in a typical video mosaicing algorithm: motion modeling, image alignment, and image composition. Depending on the type of camera motion and the structure of the objects or scenes, the *motion model* can be a 2D rigid motion model (rotation, translation, scaling), an affine model, a perspective model (homography), or a full 3D motion model.

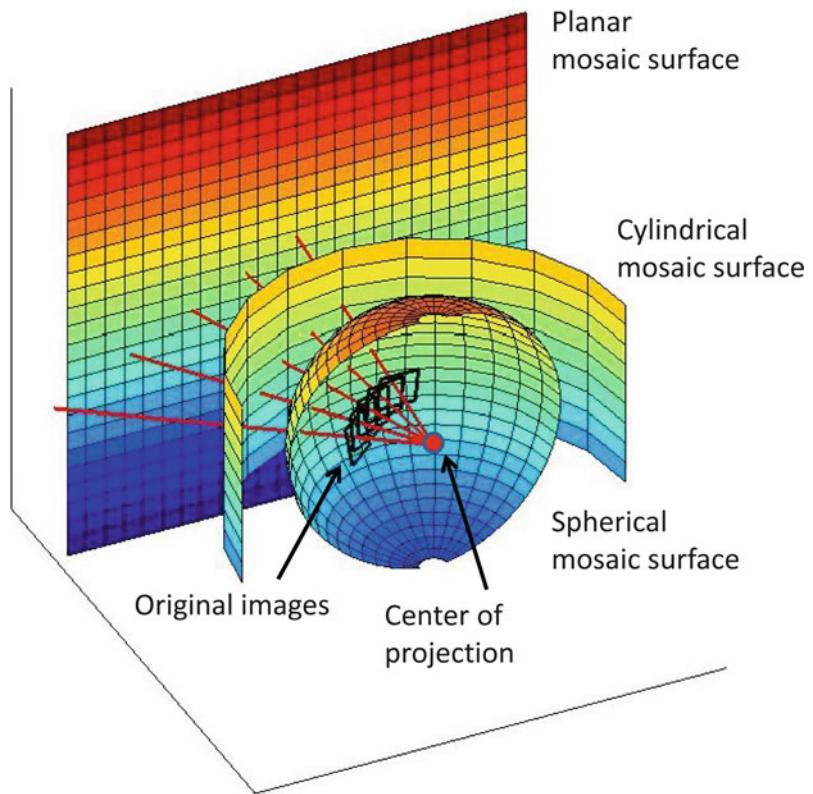
Many popular video mosaicing methods [16], for example, in [4, 15], assume a pure rotation model of the camera in which the camera rotates around its center of projection (i.e., the optical center, sometimes called nodal point). In this case, the motion between two consecutive frames can be modeled by a homography, which is a 3×3 matrix. Then, depending on the fields of view (FOVs) of the mosaic, the projection model of the mosaic can be a perspective projection (FOV is less than 180°), a cylindrical projection (FOV is 360° in one direction), or a spherical projection (full 360° FOV in both directions). Figure 1 illustrates the relations between the original images and the three types of mapping surfaces each image can be projected onto: planar, cylindrical, and spherical.

However, the applications of video mosaics from a pure rotation camera are limited to mostly consumer applications such as personal photography, entertainment, and online maps. For more specialized applications such as surveillance, remote sensing, robot navigation, and land planning, to name a few, the motion of the camera cannot be limited to a pure rotation. Translational motion usually cannot be avoided, causing the *motion parallax* problem to arise. There are three kinds of treatments for the motion parallax problem. First, when the translational components are relatively small, the motion models can be approximated by a pure rotation. In this case, the generated mosaics lack geometric accuracy, but with some treatments for the small motion parallax and moving targets, such as de-ghosting [15], the mosaics generally look very good. Second, if the scene can be regarded as planar, for example, because the distance between the camera and the scene is much larger than the depth range of the scene, the perspective motion model (homography) or, in some applications, a 2D rigid motion model or an affine model can be used [6, 10, 20]. In these cases, the problems are much simpler due to the 2D scene assumption. Finally, a



Video Mosaicing, Fig. 1

Mapping a set of overlapping images into a mosaic: planar, cylindrical, or spherical



3D camera motion model is applied when the translational components of the camera motion are large and the scene is truly 3D. In this case, motion parallax cannot be ignored or eliminated. Examples include a camera mounted on an airplane or a ground vehicle translating a large distance [7, 11, 13, 19], or a camera's optical center moving on a circular path [9, 14]. Here, multi-perspective projection models are used to generate the mosaics, enabling stereo mosaics or stereo panoramas to be created that preserve the 3D information in the scene, allowing the structure to be reconstructed and viewed in 3D. In this case, the accuracy of geometric modeling and image alignment is crucial for achieving the accuracy of 3D reconstruction and viewing.

Image alignment (or *image registration*) is the process of finding the alignment parameters (e.g., the homography in the rotational case) between two consecutive images. Image alignment is a critical step in mosaic generation, for both seamless mosaicing and for accurate geometric representation. There are two approaches to image registration: direct methods or feature-based methods. In a direct method, a

correlation approach is used to find the motion parameters. Here, the images are divided into small blocks and each block in the first image is searched for over a predefined spatial range in the second image. The best match is determined by finding the maximal correlation value. Other approaches such as using optical flow or using an iterative optimization framework also belong to the direct methods, in which no explicit feature points are extracted. In a feature-based method, a feature detection operator such as the Harris corner or SIFT (Scale Invariant Feature Transform) detector is used first, then the detected features are matched over the two frames to build up matches [16]. Either way, a parameter model is fitted using all the matches, usually using a robust parameter estimation method to eliminate erroneous feature matches. For more accurate or consistent results, a global optimization can be applied among more than two frames. For example, global alignment may be applied to all the frames in a full 360° circle in order to avoid gaps between the first and the last frame [15].

Image composition is the step of combining aligned images together to form the viewable mosaic. There

are three important issues in this step: compositing surface determination, coordinate transformation and image sampling, and pixel selection and blending. Mosaicing with the rotational camera model is a good starting point to discuss these issues (Fig. 1); mosaic compositing under other motion models are discussed afterwards.

If the video sequence only has a few images, then one of the images can be selected as the reference image, and all the other images are warped and aligned with this reference image. In this case, the reference image with a perspective projection is the compositing surface, and therefore the final mosaic is a larger perspective image, which is an extension of the field of view of the reference image. However, this approach only works when the view angles of the images span less than 90° . If the camera rotates more than 90° , a cylindrical or a spherical surface should be selected as the compositing surface. A cylindrical surface is a good representation when a full 360° panoramic mosaic is to be generated, in one direction. And a spherical surface is suitable if $360^\circ \times 360^\circ$ mosaics are to be created.

After a compositing surface is selected, the next issue is coordinate transformation and sampling. This is also called image warping. Given the motion parameters obtained in the image registration step, the mapping between each frame to the final compositing surface can be calculated: For any pixel in an original image frame, its pixel location in the compositing surface can be calculated. For generating dense pixels, an interpolation schema is needed, such as nearest neighbor, bilinear, or cubic interpolation methods. Usually a backward mapping relation is utilized such that in the mapping area on the compositing surface, each pixel obtains a value from the original image frame, line by line, and column by column. Therefore, for each integer pixel location in the mosaic, a decimal pixel location can be found in the original image; then an interpolation method is used in the original image to generate the value of the pixel in the mosaic.

The third important issue in image composition is pixel selection and blending. Naturally in generating mosaics, there are overlaps among consecutive frames, resulting in two key questions: First, *where do we place the seam (i.e., the stitching line)* (the pixel selection problem)? Second, *how do we select the values of pixels in the overlapping areas* (the pixel blending problem)? For the second problem, the simplest

methods are to average all the pixels in the same location in the overlapping area, or to use their median value. The former might create a so-called ghost effect due to moving objects, small motion parallax, or illumination changes, while the latter approach may generate a slightly better view effect. More sophisticated blending methods include Laplacian pyramid blending [3] and gradient domain blending [1]. The pixel selection problem is important when moving objects or motion parallax exists in the scene. In these cases, to avoid a person being cut in half or appearing twice in the mosaic, or to avoid cutting a 3D object that exhibits obvious motion parallax and hence could produce obvious misalignment in the mosaic, an optimal seam line can be selected at pixel locations where there are minimum misalignments between two frames [4].

Other considerations in image composition are high dynamic range imaging [4] and improved image resolution mosaicing [8]. For the former, a composite mosaic represents larger dynamic ranges than individual frames using varying shutter speeds and exposures, while the latter uses the camera motion to generate higher spatial resolution in the mosaiced image than that of the original images.

So far the discussions on image composition have focused primarily on 2-D mosaics, assuming either the camera motion is (almost) a pure rotation, or the scene is flat or very far from the camera, in order to avoid or reduce the motion parallax problem. When motion parallax cannot be avoided, 3-D mosaics have to be considered. Methods have been proposed to generate mosaics, for example, for curved documents based on 3-D reconstruction [5], when the camera motion has translational components. Needless to say, with 3-D reconstruction, a composite image with a new perspective view, or a new projection representation (such as orthogonal projection), can be synthesized from the original images. However, the drawback of this approach is a full 3-D reconstruction is needed, which is both computationally expensive and prone to noise. A more practical yet still fundamental approach without 3-D reconstruction is to generate multi-perspective mosaics from a video sequence, such as mosaics on an adaptive manifold [10], creating stitched images of scenes with parallax [7], and creating multiple-center-of-projection images [11]. When the dominant motion of the camera is translation, the projection model of the mosaic can be a parallel-perspective projection, in that the projection in the direction of the motion is parallel,



Video Mosaicing, Fig. 2 A 360° panoramic mosaic generated on a cylindrical surface <http://www-cs.engr.ccny.cuny.edu/~zhu/ThlibCylinder.JPG>



Video Mosaicing, Fig. 3 A pair of concentric mosaics of the City College of New York campus <http://www-cs.engr.ccny.cuny.edu/~zhu/CSCI6716/CCNYCampus.jpg>



Video Mosaicing, Fig. 4 A pair of pushbroom mosaics of the Amazon rain forest <http://www-cs.engr.ccny.cuny.edu/~zhu/57z10StereoColor.jpg>

whereas the projection perpendicular to the motion remains perspective. This kind of mosaic is also called pushbroom mosaic [17] since the projection model of the mosaic in principle is the same as pushbroom imaging in remote sensing. A more interesting case is that by selecting different parts of individual frames, a pair of stereo mosaics can be generated that exhibit motion parallax, while each of them represent a particular viewing angle of parallel projection [19]. To generate stereo mosaics, the motion model is 3D, and therefore, a bundle adjustment for 3D camera orientation is needed. The projection model is parallel perspective, and therefore, the composting surface is a plane that holds the parallel-perspective image. To generate a true parallel-perspective view in each mosaic for accurate 3D reconstruction, pixel selection is carried out for that particular viewing angle and a coordinate transformation is performed based on matches between at least two original images for each pixel. A similar principle can be applied to concentric mosaics with circular projection [9, 14].

In some applications such as surveillance and mapping, geo-referencing mosaicing is also an important topic. This is usually done when geo-location metadata is available, for example, from GPS and IMU measurements [18, 20] taken with the video/images. Geo-referenced mosaics assign a geo-location to each

pixel either by directly using the metadata from the video frames used to generate the mosaic or, when metadata is not available, by aligning the video frames to a geo-referenced reference image such as a satellite image.

Video mosaicing techniques are also used for dynamic scenes, such as to generate dynamic pushbroom mosaics for moving target detection [17] and to create animated panoramic video textures in which different portions of a panoramic scene are animated with independently moving video loops [2, 12].

Open Problems

Some open problems can be found in a good survey paper on image alignment and stitching [16]. These include robust alignments for stereo mosaics (or mosaics with motion parallax), mosaics for high dynamic range imaging and for super-resolution imaging, and dynamic mosaics.

Experimental Results

Figure 2 shows a 360° panoramic mosaic represented on a cylindrical surface, which is generated from a

video sequence taken by a video camera that roughly rotates around its optical center. [Figures 3](#) and [4](#) show two stereo mosaics that can be viewed with a pair of 3D glasses, red for the right eye and the cyan for the left eye. High-resolution mosaics can be viewed by clicking the images in the figures in the online edition. The concentric stereo mosaic in [Fig. 3](#) is generated from a video sequence taken by a handheld video camera that undertakes an off-center rotation with 360 degrees of field of view coverage. [Figure 4](#) is a pair of pushbroom stereo mosaics created from a video sequence taken by a camera looking down from an airplane flying over the Amazon rain forest.

References

1. Agarwala A, Dontcheva M, Agrawala M, Drucker S, Colburn A, Curless B, Salesin D, Cohen M (2004) Interactive digital photomontage. *ACM Trans Graph* 23(3): 292–300
2. Agarwala A, Zheng C, Pal C, Agrawala M, Cohen M, Curless B, Salesin D, Szeliski R (2005) Panoramic video textures. *ACM Trans Graph* 24(3):821–827
3. Burt PJ, Adelson EH (1983) A multiresolution spline with applications to image mosaics. *ACM Trans Graph* 2(4): 217–236
4. Eden A, Uyttendaele M, Szeliski R (2006) Seamless image stitching of scenes with large motions and exposure differences. In: IEEE computer society conference on computer vision and pattern recognition (CVPR’2006). IEEE Computer Society, Los Alamitos, pp 2498–2505
5. Iketani A, Sato T, Ikeda S, Kanbara M, Nakajima N, Yokoya N (2006) Video mosaicing for curved documents based on structure from motion. In: ICPR, Hong Kong, vol 4, pp 391–396
6. Irani M, Anandan P, Hsu SC (1995) Mosaic based representations of video sequences and their applications. In: ICCV. IEEE Computer Society, Los Alamitos, pp 605–611
7. Kumar R, Anandan P, Irani M, Bergen J, Hanna K (1995) Representation of scenes from collections of images. In: IEEE workshop on representations of visual scenes. IEEE Computer Society, Los Alamitos, pp 10–17
8. Marzotto R, Fusello A, Murino V (2004) High resolution video mosaicing with global alignment. In: CVPR, vol 1. IEEE Computer Society, Los Alamitos, pp 692–698
9. Peleg S, Ben-Ezra M (1999) Stereo panorama with a single camera. In: IEEE conference on computer vision pattern recognition (CVPR). IEEE Computer Society, Los Alamitos, pp 1395–1401
10. Peleg S, Rousso B, Rav-Acha A, Zomet A (2000) Mosaicing on adaptive manifolds. *IEEE Trans Pattern Anal Mach Intell* 22:1144–1154
11. Rademacher P, Bishop G (1998) Multiple-center-of-projection images. In: Computer graphics proceedings. Annual conference series. Association for Computing Machinery, New York, pp 199–206
12. Rav-Acha A, Pritch Y, Lischinski D, Peleg S (2005) Dynamosaics: video mosaics with non-chronological time. In: IEEE computer society conference on computer vision and pattern recognition (CVPR’2005). IEEE Computer Society, Los Alamitos, pp 58–65
13. Rousso B, Peleg S, Finci I, Rav-Acha A (1998) Universal mosaicing using pipe projection. In: ICCV. Narosa Publishing House, New Delhi, pp 945–952
14. Shum H-Y, Szeliski R (1999) Stereo reconstruction from multiperspective panoramas. In: Seventh international conference on computer vision (ICCV’99), IEEE Computer Society, Los Alamitos, pp 14–21
15. Shum H, Szeliski R (2000) Systems and experiment paper: construction of panoramic image mosaics with global and local alignment. *Int J Comput Vis* 36:101–130
16. Szeliski R (2006) Image alignment and stitching: a tutorial. *Found Trends Comput Graph Vis* 2(1):1–104
17. Tang H, Zhu Z, Wolberg G (2006) Dynamic 3D urban scene modeling using multiple pushbroom mosaics. In: 3DPVT, Chapel Hill, USA, pp 456–463
18. Taylor CN, Andersen ED (2008) An automatic system for creating geo-referenced mosaics from MAV video. In IROS. IEEE, Piscataway, pp 1248–1253
19. Zhu Z, Hanson AR, Riseman EM (2004) Generalized parallel-perspective stereo mosaics from airborne video. *IEEE Trans Pattern Anal Mach Intell* 26: 226–237
20. Zhu Z, Riseman EM, Hanson AR, Schultz HJ (2005) An efficient method for geo-referenced video mosaicing for environmental monitoring. *Mach Vis Appl* 16: 203–216

Video Mosaicking

► Video Mosaicing

Video Retrieval

Cees G. M. Snoek^{1,3} and Arnold W. M. Smeulders^{2,3}

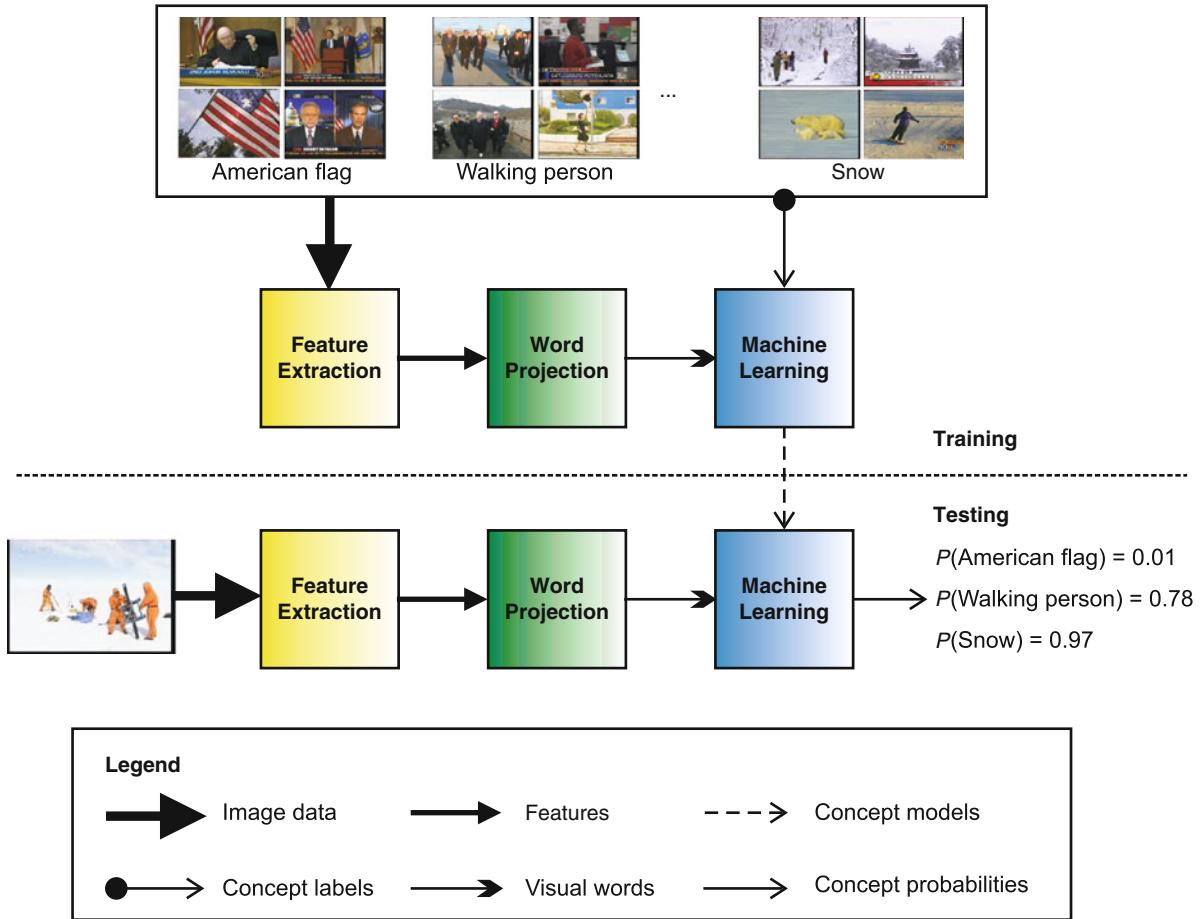
¹University of Amsterdam, Amsterdam,
The Netherlands

²Centre for Mathematics and Computer Science
(CWI), University of Amsterdam, Amsterdam,
The Netherlands

³Intelligent Systems Lab Amsterdam, Informatics
Institute University of Amsterdam, Amsterdam,
The Netherlands

Synonyms

Multimedia retrieval; Video search



Video Retrieval, Fig. 1 General scheme for detecting visual concepts in images, with three typical concepts highlighted. First, researchers project extracted image features into visual words. Then they train concept models from both the visual

words and the concept labels using machine learning. Finally, during testing, researchers assign concept probabilities to previously unlabeled images

Definition

Video retrieval is the process of searching in video based on an analysis of its visual content.

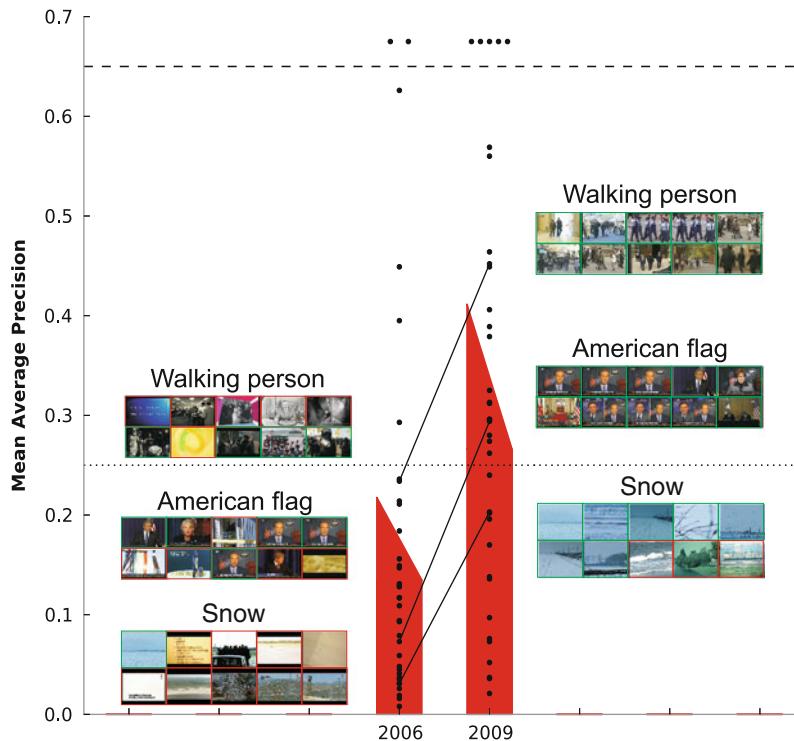
Background

The cause for the general video retrieval problem is the semantic gap: the lack of correspondence between the low-level features that machines extract from the visual signal and the high-level conceptual interpretations a human gives [1]. In order to bridge the gap, many retrieval solutions have been proposed in the past, e.g., by using text, speech, tags, or example images [2, 3]. But the most authentic and cognitive hardest is to type

a concept from visual information and to retrieve the images carrying that concept [4].

Theory

The video retrieval method of choice in the field is rendered in Fig. 1. The first step is to extract from an image locally measured features, lots of them, ranging from 40 to 100,000. The features are invariant descriptors which cancel out accidental circumstances of the recording caused by differences in lighting, viewpoint, or scale. In order to capture the complexity of the world, many texture, shape, and color descriptors need to be extracted.



Video Retrieval, Fig. 2 Video retrieval progress as evaluated on 36 concept detectors (●) derived from broadcast video data using state-of-the-art search engines from 2006 and 2009. The figure highlights performance for three typical concepts. The *top* of the skewed bar indicates the maximum average performance

by training on similar examples, and the *bottom* indicates the minimum performance when training on a data set of completely different origin [7]. Progress in video retrieval is substantial and quickly maturing in robustness for real-world usage of any concept

The second step is to project the descriptors per pixel onto one of 4,000 words. They are not real words but rather summarizations of one local patch of the image describing one detail: a corner, a texture, or a point. Researchers initially only summarized the image at the most salient points, but it now appears that full-density descriptions are superior.

In the third step, a machine-learning algorithm converts the visual words into one of the semantic concepts. In fact, it assigns a probability to all of the concepts simultaneously, which are used for ranking images in terms of concept presence. Researchers train the algorithm with the help of manually labeled examples. Because there are far more negative examples than positive ones, they intensively compute the optimal machine-learning parameters using grids and GPUs.

Detecting an object such as the American flag is relatively simple if one has an answer to the variance in sensory conditions like illumination and shading,

as the flag always shows the same colors and color transitions. Note that a geometrical model of a flag would fail almost always as it rarely appears like a straight square. To detect a walking person from one image requires a richness of poses learned from a labeled set, and snow is even harder to detect as it is white, is texture-free, and may assume all sorts of shapes. Remarkably, although none of the features in current detection methods is specific to any of the concepts, the technique can still detect any of them with sufficient success.

Application

Crucial drivers for progress in video retrieval are international search engine benchmarks such as ImageCLEF (Cross-Language Evaluation Forum), Pascal VOC (Visual Object Classes), PETS (Performance Evaluation of Tracking and Surveillance),

and VACE (Video Analysis and Content Extraction). However, thus far the National Institute of Standards and Technology TRECVID (TREC Video Retrieval) benchmark [5] has played the most significant role.

The aim of the TRECVID benchmark is to promote progress in video retrieval by providing a large video collection, uniform evaluation procedures, and a forum for researchers interested in comparing their results. NIST performs an independent examination of results using standard information retrieval evaluation measures, like average precision. With participation from more than 100 teams, including University of Oxford, Tsinghua University, and Columbia University, TRECVID has become the de facto standard for evaluating video retrieval research.

TRECVID has been an important driver for the community in sharing resources for validity of video retrieval experiments, most notably the manual annotations provided by the Large Scale Concept Ontology for Multimedia [6]. Due to the open character of benchmarks, effective concept detection approaches are quickly handed over from one group to another implementing fast convergence on successful methods. It was shown recently that in just 3 years, performance has doubled [7]. For learned concepts, detection rates degenerate when applied to data of a different origin. However, in this setting also, performance has doubled in just 3 years; see Fig. 2.

Open Problems

Despite good progress many problems in video retrieval remain to be solved. Apart from the need to further improve the robustness and efficiency of concept detectors, there is also a need to expand the number of detectors and to facilitate retrieval mechanisms that cater for on-the-fly search [8]. For both scenarios, the widely available socially tagged image examples on the web seem suited. Another open problem considers complex search requests by the combination of detectors at query time into events or short stories. Even though individual detectors may be reasonable, their combination often yields nothing but noise. Finally, the problem of explaining to a user what visual information in the video was the characteristic evidence in making a meaningful retrieval decision is unsolved.

References

1. Smeulders AWM, Worring M, Santini S, Gupta A, Jain R (2000) Content-based image retrieval at the end of the early years. *IEEE Trans Pattern Anal Mach Intell* 22(12):1349–1380
2. Waclaw HD, Christel MG, Gong Y, Hauptmann AG (1999) Lessons learned from building a terabyte digital video library. *IEEE Comput* 32(2):66–73
3. Smith JR, Chang SF (1997) Visually searching the web for content. *IEEE MultiMed* 4(3):12–20
4. Snoek CGM, Worring M (2009) Concept-based video retrieval. *Found Trends Inf Retr* 4(2):215–322
5. Smeaton AF, Over P, Kraaij W (2006) Evaluation campaigns and TRECVID. In: Proceedings of the ACM SIGMM international workshop on multimedia information retrieval. ACM, New York, pp 321–330
6. Naphade MR, Smith JR, Tešić J, Chang SF, Hsu W, Kennedy LS, Hauptmann AG, Curtis J (2006) Large-scale concept ontology for mul- timedia. *IEEE MultiMed* 13(3):86–91
7. Snoek CGM, Smeulders AWM (2010) Visual-concept search solved? *IEEE Comput* 43(6):76–78
8. Hauptmann AG, Yan R, Lin WH, Christel MG, Waclaw H (2007) Can high-level concepts fill the semantic gap in video retrieval? A case study with broadcast news. *IEEE Trans Multimed* 9(5):958–966

Video Search

► [Video Retrieval](#)

View and Rate-Invariant Human Action Recognition

Vasu Parameswaran¹ and Ashok Veeraraghavan²

¹Microsoft Corporation, Sunnyvale, CA, USA

²Department of Electrical and Computer Engineering, Rice University, Houston, TX, USA

Related Concepts

► [Gait Recognition](#); ► [Gesture Recognition](#)

Definition

View and rate-invariant human action recognition is the recognition of actions independent of the camera viewpoint, action speed, and frame rate of capture of the video.



Background

At its essence, human action is the movement of the body through a sequence of poses. The visual appearance of a given action in a video sequence depends upon several classes of variables: (1) the geometry of the person performing the action, (2) the style of the action being performed, (3) the clothing worn by the person, (4) the camera viewpoint, and (5) the time taken, not only for the entire action but also for each individual pose transition to complete. A generally applicable human action recognition system needs the ability to classify an action from its visual appearance regardless of the values of the above classes of variables. In other words, the system needs to be *subject invariant, style invariant, clothing invariant, view invariant, and rate invariant*. Subject invariance, style invariance and clothing invariance have seen relatively little progress so far. While progress has been made on view invariance and rate invariance, the collective body of work has not yet matured. Therefore, this entry summarizes the main approaches that have been proposed so far, highlighting their motivations, theoretical foundations, and limitations.

Theory

Human action recognition is dependent upon modules isolating different humans in the scene. Therefore, in the following, it is assumed that human detection has been performed and image regions corresponding to a single human performing an action have been identified and are provided as an input to the view and rate-invariant human action recognition system.

View-Invariance

The objective of view invariance arises mainly in the case of monocular video. Multiple cameras allow the possibility of acquiring the depths of image points, which makes the view-invariance problem much easier. Therefore, this section focuses only on monocular video.

Detecting an action requires a model, either explicit or implicit. An explicit model for human action can be based on the temporal sequences of body joint angles. Recovery of joint angles from image sequences is difficult due to joints being physically hidden behind

skin and clothing and due to the unknown viewpoint. Although it may be possible to theoretically establish viewpoint invariance using explicit models, such approaches are limited by the difficulties involved in recovering joint angles.

Alternatively, actions can be modeled implicitly, based on quantities derived directly from the observed images. However, such models maintain at best a tenuous link to the hidden joint angles, making it difficult to theoretically establish view invariance.

Given difficulties with both types of approaches, it is not surprising to note that no theoretically sound and practically applicable algorithm has been designed so far that can recognize human actions invariant to viewpoint changes. Nevertheless, two classes of approach have emerged based on explicit and implicit models for human action that are described in more detail below.

Geometry-Inspired Approaches

Geometry-inspired approaches draw upon results from geometric invariance and apply them to points on the body as it moves. Geometry driven approaches depend upon body parts having been detected, tracked, and labeled across frames. Extremities of the body such as the head, hands, and feet have a better chance of being detected, whereas internal joints such as shoulders, elbows, and knees are harder to detect. Therefore, methods based on extremities have a better chance of working in practice than methods based on the full body. However, such methods are limited in applicability due to the exclusion of internal joints. Methods to infer the full 2D body pose from silhouettes have been proposed [2, 11], but a system integrating such methods with a full-body-based view-invariant action recognition method has not been demonstrated yet. An example of an extremity-based approach is [9]. Examples of full-body approaches are [6, 7], and [13].

Image-Based Approaches

An implicit model of an action can be based upon a manifold of appearances arising from the same action as seen from different viewpoints. Given an unknown action from an unknown viewpoint, appearance descriptors are extracted and matched with learned appearance descriptors. Matching error is calculated either based on pixel-wise dissimilarity of the silhouettes [5, 18], or in a derived space such as Hu moment space or Fourier shape space [14].



An implicit model for an action can also be based upon exploiting self-similarity. The observation used in such a method is that although a given pose may appear different from different views, repeating instances of the same pose across time will be self-similar in a given view, provided that the viewing angle between the subject and the camera for the pose is the same. Constructing a pair-wise self-similarity matrix across time will produce structures that appear similar. Such an approach assumes the existence of repeated poses in the action and works best when there are many repeating instances within one action cycle. Examples of such approaches include [3] and [12].

Rate Invariance

A robust action recognition system has to be invariant to the rate at which the action is performed. This will enable the results of the action recognition system to be invariant to the frame rate of the camera and the rate at which the actor is performing these actions. While a linear rate invariance is sufficient to handle frame-rate variations, it is necessary to model and be able to handle nonlinear rate variations in order to handle the rate changes that are due to actors, speed of performance. This is because the actors may perform different sub-actions at varying relative rates thereby leading to a nonlinear rate change across the entire action.

Motivation

Consider the INRIA iXmas activity recognition dataset [17]. Shown in Fig. 1(L) is the distribution of the number of frames in different executions of the same activity for four distinct activities. Figure 1(L) clearly shows that for the same activity, the rate of execution and consequently the number of frames during the execution varies significantly. Moreover, in most realistic scenarios, this temporal warping might also be inherently nonlinear making simple resampling methods ineffective. This implies that for uncontrolled scenarios, the variations due to temporal warpings could be even more significant. Ignoring this temporal warping might lead to structural inconsistencies apart from providing poor recognition performance. The sequence of images shown in the first two rows of Fig. 1(R) corresponds to two different instances of the same individual performing the same activity. There is an obvious temporal warping between the two sequences. If this temporal warping is ignored, the distance between

these two sequences will be large, leading to incorrect matching. Moreover, if some statistical description of the activity, like an average sequence, is required, then ignoring the temporal warping could lead to structural inconsistencies like the presence of four arms and two heads in the average sequence, shown in the third row of Fig. 1(R). If temporal warping is accounted for, then such inconsistencies are avoided and the distance between the two sequences is rightly small. The fourth row shows a typical average sequence obtained after accounting for time warping.

Rate Invariance in Gait Recognition

Results on gait-based person identification shown in [1] indicate that it is very important to take into account the temporal variations in the person's gait. In [15], preliminary work indicating that accounting for execution rate enhances recognition performance for action recognition was presented. Typical approaches for accounting for variations in execution rate are either directly based on the dynamic time-warping (DTW) algorithm [8] or some variation of this algorithm [15]. A method for computing an average shape for a set of dynamic shapes inspite of the existence of varying rates of execution is provided in [4]. A method to learn the best class of time-warping transformations for a given classification problem is proposed in [10].

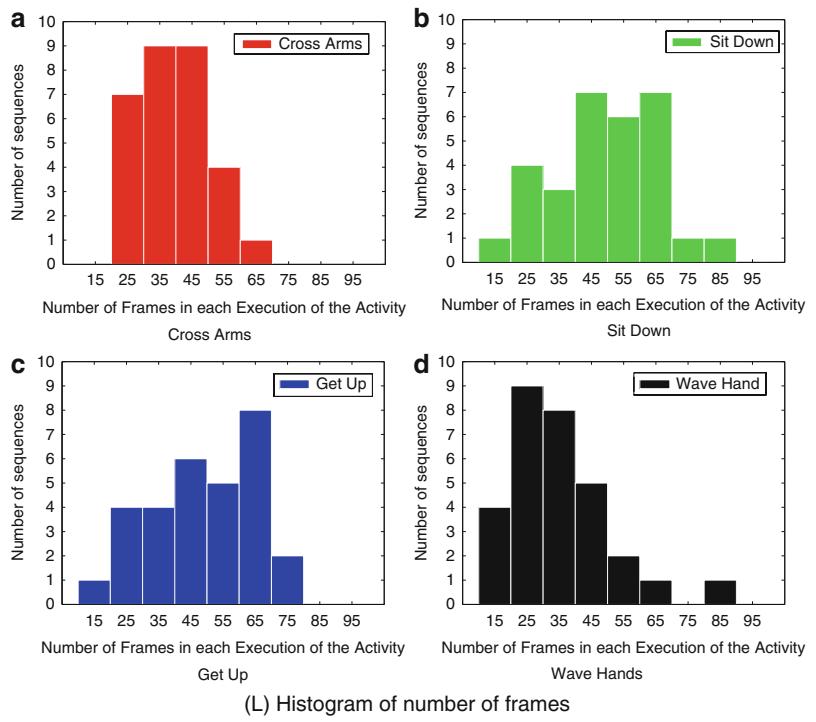
Decoupling Feature Variability from Rate Variability

The most common way to handle rate variations in the execution of an action is to decouple the variations in features from the variations in dynamics due to rate changes. This is done by modeling an action sequence as a composition of these two sources of variability – variability on the feature space and variability due to execution rate. Keeping the model on the feature space completely independent of the model on the space of execution rates allows for the ability to exploit any of the above-mentioned viewpoint-invariant features. Therefore, as more sophisticated features become available, such models will be able to exploit the characteristics of those features while retaining the ability to deal with variations in execution rate. If the chosen features are viewpoint and anthropometry invariant, then the resulting algorithm becomes invariant to all the three significant modes of variations – viewpoint, anthropometry, and execution rate.

View and Rate-Invariant Human Action

Human Action Recognition, Fig. 1

Figure courtesy [16] (L) Histogram of the number of frames in different executions of the same action in the INRIA iXmas dataset. The histograms for four different activities are shown. (a) Cross arms (b) Sit down (c) Get up (d) Wave hands. (R) Row 1, Row 2: Two instances of the same activity. Row 3: A simple average sequence. Row 4: average sequence after accounting for time warps



(R) Two sequences with differing rates of executions and their normal and warped average sequences.

Dynamic Time Warping

Dynamic time warping (DTW) [8] is the most common algorithm that is used for accounting for the nonlinear execution rate variations. The DTW algorithm which is based on dynamic programming computes the best nonlinear time normalization of the test sequence in order to match the template sequence by searching over the space of all time warpings. The advantage of using DTW is that by cleverly using dynamic programming,

the complexity of the search space is considerably reduced. The usual temporal consistency constraints used in order to reduce the space of time warpings are:

- End point constraints: The start and the end of the activity trajectories must match exactly.
- Monotonicity: The warping function should be monotonically increasing, i.e., the sequence of action units must be unchanged.

- Continuity: The warping function must be continuous.

The DTW algorithm thus computes the best time warping between a test sequence and a template sequence. Once the best time warping is computed, then the test sequence is then unwarped according to the computed warp. The unwarped action sequences are now time-synchronized since all temporal rate variations have been accounted for. From each frame of the unwarped sequence, features (typically view-invariant features) are extracted and then matched in order to result in an algorithm that is both view and rate invariant.

Open Problems

Approaches that are theoretically view invariant are based on inherent parameters of an action such as joint angles or positions. Unless these methods are coupled with methods that infer the inherent joint parameters, they will not be of practical value. On the other hand, image-based approaches are practical but not generally and probably view invariant. An open problem in view-invariant action recognition is to find a method that is theoretically sound and yet practical.

References

1. Bobick A, Tanawongsuwan R (2003) Performance analysis of time-distance gait parameters under different speeds. In: Proceedings of the 4th international conference on IEEE conference on computer vision and pattern recognition (CVPR), Madison
2. Jiang H (2011) Human pose estimation using consistent max-covering. Pattern Anal Mach Intell IEEE Trans PP(99):1
3. Junejo IN, Dexter E, Laptev I, Perez P (2011) View-independent action recognition from temporal self-similarities. IEEE Trans Pattern Anal Mach Intell 33(1):172–185
4. Maurel P, Sapiro G Dynamic shapes average. www.ima.umn.edu/preprints/may2003/1924.pdf
5. Ogale A, Karapurkar A, Aloimonos Y (2007) View-invariant modeling and recognition of human actions using grammars. In: Vidal R, Heyden A, Ma Yi (eds) Dynamical vision. Volume 4358 of lecture notes in computer science. Springer, Berlin/Heidelberg, pp 115–126
6. Parameswaran V, Chellappa R (2005) Human action recognition using mutual invariants. Comput Vis Image Underst J 98(2):294–324
7. Parameswaran V, Chellappa R (2006) View invariance for human action recognition. Int J Comput Vis 66(1):83–101
8. Rabiner L, Juang B (1993) Fundamentals of speech recognition. Prentice Hall, Englewood Cliffs. <http://www.amazon.com/Fundamentals-Speech-Recognition-Lawrence-Rabiner/dp/0130151572>
9. Rao C, Yilmaz A, Shah M (2002) View-invariant representation and recognition of actions. Int J Comput Vis 50(2):203–226
10. Ratanamahatana CA, Keogh E (2004) Making time-series classification more accurate using learned constraints. In: Proceedings of the SIAM international conference on data mining, Orlando pp 11–22
11. Rosales R, Sclaroff S (2000) Inferring body pose without tracking body parts. In: Proceedings of the 2000 IEEE conference on computer vision and pattern recognition (CVPR), Hilton Head, Island, vol.2, pp 721–727
12. Seitz SM, Dyer CR (1997) View-invariant analysis of cyclic motion. Int J Comput Vis 25:231–251
13. Shen YP, Foroosh H (2009) View-invariant action recognition from point triplets. IEEE Trans Pattern Anal Mach Intell 31(10):1898–1905
14. Souvenir R, Parrigan K (2009) Viewpoint manifolds for action recognition. EURASIP J Image Video Process
15. Veeraraghavan A, Chellappa R, Roy-Chowdhury AK (2006) The function space of an activity. In: Proceedings of the 2006 IEEE computer society conference on computer vision and pattern recognition (CVPR), New York, vol 1, pp 959–968
16. Veeraraghavan A, Srivastava A, Roy-Chowdhury AK, Chellappa R (2009) Rate-invariant recognition of humans and their activities. Image Process IEEE Trans 18(6): 1326–1339
17. Weinland D, Ronfard R, Boyer E (2006) Free viewpoint action recognition using motion history volumes. Comput Vis Image Underst 104(2–3):249–257
18. Weinland D, Boyer E, Ronfard R (2007) Action recognition from arbitrary views using 3D exemplars. In: Proceeding of the IEEE 11th international conference on computer vision, ICCV 2007, Rio de Janeiro, 14–21 Oct 2007, pp 1, 7

Vignetting

Amit Agrawal
Mitsubishi Electric Research Laboratories,
Cambridge, MA, USA

Related Concepts

► Radiance; ► Radiometric Calibration

Definition

Vignetting is a reduction of an image's brightness at the periphery compared to the center of the image. It describes the effective falloff in irradiance for off-axis points for imaging systems.

Background

In real imaging systems, the image brightness is often reduced at the periphery compared to the center of the image. This effect is known as vignetting and is undesirable for computer vision algorithms that rely on measured pixel intensities. Vignetting can be caused by several mechanisms. The image irradiance varies across the field of view according to the fourth power of the cosine of the field angle. This off-axis illumination falloff is one of the prominent reasons for vignetting and is also referred to as cosine-fourth falloff [1].

Vignetting can also be caused by optical and mechanical effects. Light rays arriving at oblique angles to the optical axis may be obstructed by the aperture stop, lens rim, or improper lens hood. This could lead to spatially varying attenuation over the entire image. A large aperture causes more vignetting, and using a small aperture (increasing the F-number) can reduce it. Real lenses often have multiple lens elements to correct for spherical and chromatic aberrations. Rear lens elements may be partially occluded by front lens elements, reducing off-axis illumination.

For digital cameras, pixel vignetting is another vignetting effect caused by the variation in angular sensitivity of digital sensors to the incoming light. A micro-lens array on top of the sensor helps to collect off-axis light and can reduce pixel vignetting.

Pupil aberration [2] is another cause of irradiance falloff at the periphery. This is attributed to nonlinear refraction of rays resulting in a nonuniform light distribution across the aperture.

Theory

It is important to remove vignetting effects for computer vision applications. A simple way to remove vignetting is to capture a reference photo of a uniformly illuminated white background. This reference photo can be used to cancel the vignetting effect on subsequent images captured by the camera. However, such a reference photo is valid only for the images captured under the same camera settings (zoom, aperture) and the same illumination conditions. This approach cannot be used on photos captured under settings other than those used for the reference photo or on photos captured by unknown cameras. Techniques that



Vignetting, Fig. 1 An example of vignetting

model the vignetting effects to recover a vignetting-free image [3–8] are useful in such scenarios (Fig. 1).

Let $\mathbf{x} = [u, v]$ denote a pixel \mathbf{x} with coordinates u and v . The observed image intensities $I(\mathbf{x})$ can be modeled as

$$I(\mathbf{x}) = \text{CRF}(kL(\mathbf{x})V(\mathbf{x})), \quad (1)$$

where $L(\mathbf{x})$ is the true irradiance, $V(\mathbf{x})$ denotes the attenuation due to vignetting, k is the exposure value, and $\text{CRF}(\cdot)$ is the camera response function. Several models for the vignetting function $V(\mathbf{x})$ exist [3, 5, 6]. The polynomial vignetting model is given by

$$V(\mathbf{x}) = 1 + \sum_{n=1}^D \beta_n r(\mathbf{x})^{2n}, \quad (2)$$

where $r(\mathbf{x})$ is the distance of pixel \mathbf{x} from the image center. Similar model based on hyperbolic cosine functions is described in [7].

In [9], Kang and Weiss proposed a model to correct for vignetting effects using a single photo of a flat, textureless surface. Their model describes the overall image attenuation in terms of an off-axis illumination factor $A(\mathbf{x})$, a camera tilt factor $T(\mathbf{x})$, and a geometric factor $G(\mathbf{x})$:

$$V(\mathbf{x}) = A(\mathbf{x})G(\mathbf{x})T(\mathbf{x}), \quad (3)$$

where

$$A(\mathbf{x}) = \frac{1}{(1 + r^2(\mathbf{x})/f^2)^2}, \quad (4)$$

$$G(\mathbf{x}) = 1 - \alpha r(\mathbf{x}), \quad (5)$$

$$T(\mathbf{x}) = \cos \tau \left(1 + \frac{\tan \tau}{f} (u \sin \psi - v \cos \psi)\right)^3, \quad (6)$$

f denotes the effective focal length and α is a constant describing the geometric factor $G(\mathbf{x})$. The camera tilt factor $T(\mathbf{x})$ is described using a tilt axis in a plane parallel to the image plane at an angle ψ with respect to the x -axis, and the tilt angle is denoted by τ . $T(\mathbf{x})$ is used to account for the foreshortening of the imaged surface with respect to the camera. An extended version of Kang-Weiss model is described in [3]. These models can be used to correct the vignetting effects.

Application

Vignetting is usually an undesirable effect caused by real imaging systems. Due to vignetting, the observed image intensities deviate from those predicted by standard models of image formation. Thus, computer vision algorithms may not perform well in presence of vignetting. On the other hand, photographers sometimes introduce intentional vignetting for artistic effects, such as to draw attention to the center of the frame. Vignetting can also be used for camera calibration under certain conditions [9, 10].

Vignetting models can be used to recover a vignetting-free image. Examples of vignetting removal are presented in [3–5]. Vignetting correction is also used in stitching multiple photos to compensate for the varying intensity of scene points across different images [5, 6].

References

1. Horn B (1986) Robot vision. McGraw-Hill, New York
2. Aggarwal M, Hua H, Ahuja N (2001) On cosine-fourth and vignetting effects in real lenses. In: International conference on computer vision, Vancouver, vol 1, pp 472–479
3. Zheng Y, Lin S, Kang SB (2006) Single image vignetting correction. In: IEEE computer society conference on computer vision and pattern recognition (CVPR), New York, pp 461–468
4. Zheng Y, Yu J, Kang SB, Lin S, Kambhamettu C (2008) Single-image vignetting correction using radial gradient symmetry. In: IEEE computer society conference on computer vision and pattern recognition (CVPR), Anchorage, pp 1–8
5. Kim SJ, Pollefeys M (2008) Robust radiometric calibration and vignetting correction. IEEE Trans Pattern Anal Mach Intell 30:562–576
6. Goldman DB, Chen JH (2005) Vignette and exposure calibration and compensation. In: International conference on computer vision, Beijing, pp 899–906
7. Yu W (2004) Practical anti-vignetting methods for digital cameras. IEEE Trans Consum Electron 50:975–983
8. Yu W, Chung Y, Soh J (2004) Vignetting distortion correction method for high quality digital imaging. In: Proceedings of the 17th IEEE international conference on pattern recognition, Cambridge, pp 666–669
9. Kang SB, Weiss R (2000) Can we calibrate a camera using an image of a flat, textureless lambertian surface? In: European conference on computer vision, Dublin, pp 640–653
10. Zheng Y, Kambhamettu C, Lin S (2009) Single-image optical center estimation from vignetting and tangential gradient symmetry. In: IEEE computer society conference on computer vision and pattern recognition (CVPR), Miami Beach, pp 2058–2065

Vignetting Estimation

▶ Calibration of Radiometric Falloff (Vignetting)

Viscosity Solution

Fabio Camilli¹ and Emmanuel Prados²

¹Dipartimento SBAI, “Sapienza”, Università di Roma, Rome, Italy

²INRIA Rhône-Alpes, Montbonnot, France

Definition

Viscosity solution is a notion of weak solution for a class of partial differential equations of Hamilton-Jacobi type.

Background

A first-order partial differential equation of the type

$$H(x, u(x), Du(x)) = 0 \quad (1)$$

is called a Hamilton-Jacobi equation. A function u is said to be a classical solution of (Eq. 1) over a domain if u is continuous and differentiable over the entire



domain and x , $u(x)$, and $Du(x)$ (the gradient of u at x) satisfy the above equation at every point of the domain. Consider the boundary value problem

$$|u'(x)| - 1 = 0 \text{ for } x \in (-1, 1), u(\pm 1) = 0. \quad (2)$$

By Rolle's theorem, it is easily seen that classical solutions of the previous problem do not exist, whereas there exist infinite many weak solutions, that is, continuous functions which satisfy the equation at almost every point (the saw-tooth solutions, see Fig. 1a).

At first, this situation can seem rather atypical, but in fact, it is nothing of the sort. For example, it concerns the distance functions which are widely used in computer vision. In particular, it is easy to see that the distance function of a closed curve in a plan, which typically has strong edges (see Fig. 1), is almost everywhere a solution of the Eikonal equation

$$|Du(x)| - F(x) = 0 \quad (3)$$

with $F(x) = 1$ and $u(x) = 0$ on the curve. Also, in general, as in the case of (Eq. 2), this last equation has no classical solution. Another typical example concerns the shape-from-shading problem which naturally yields a Hamilton-Jacobi equations having the same behavior. In particular, by modeling the problem with an orthographic camera, a directional front lighting, and a Lambertian surface, the problem consists then in solving an Eikonal equation in which the function F depends on the considered image.

It is therefore very important to have a theory which allows merely continuous functions to be solutions of Hamilton-Jacobi equations and to provide at the same time a way to select the relevant solution among the weak solutions of the problem.

Theory

The notion of viscosity solution was introduced at the beginning of the 1980s by M. G. Crandall and P. L. Lions [11], and it is related to Kruzkov's theory of entropy solutions for scalar conservation laws.

The basic idea is to replace the differential $Du(x)$ at a point x where it does not exist (e.g., because of a kink in u) with the differential $D\phi(x)$ of a smooth function

ϕ touching the graph of u , from above for the subsolution condition and from below for the supersolution one, at the point x .

Definition 1 (i) A continuous function u is said to be a viscosity subsolution of (Eq. 1) if for any x and for any smooth function ϕ such that $u - \phi$ has a maximum point at x , then

$$H(x, u(x), D\phi(x)) \leq 0.$$

(ii) A continuous function u is said to be a viscosity supersolution of (Eq. 1) if for any x and for any smooth function ϕ such that $u - \phi$ has a minimum point at x , then

$$H(x, u(x), D\phi(x)) \geq 0.$$

(iii) A continuous function u is said to be a viscosity solution of the Hamilton-Jacobi equation if it is a viscosity subsolution and supersolution.

There is also an equivalent definition for viscosity solution which involves the notion of sub- and superdifferentials (see [5]).

It is straightforward to observe that solutions in the classical sense are viscosity solutions. Inversely, if a viscosity solution u is differentiable at x , then it solves the equation at this point in the classical sense. Hence, the notion of viscosity solution includes the one of classical solution.

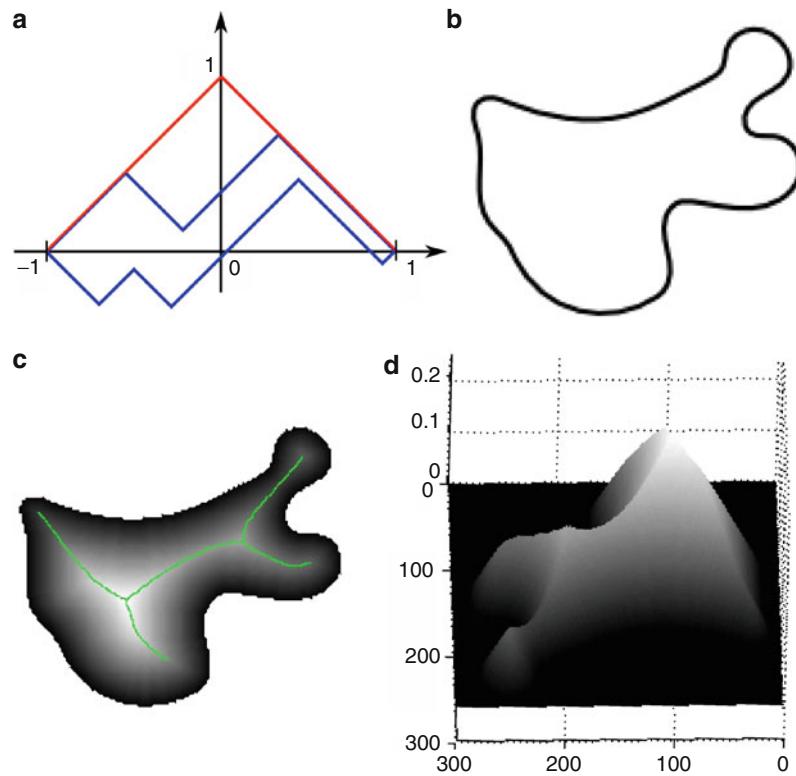
By looking closely at the definition, one can understand rather intuitively how it allows a specific weak solution to be selected among the other ones and which of them is selected. In fact, this definition eliminates a certain type of edges. In particular for Hamilton-Jacobi (Eq. 2), the definition allows upward edges, but not downward edges. Thus, in Fig. 1, all the weak solutions of (Eq. 2) which have downward edges are then excluded. Also, only the maximal weak solution (represented in red) is a viscosity solution.

In addition, the definition of viscosity solutions selects among the almost everywhere solutions the one which is consistent with the regularized problem. In fact, the name "viscosity" is motivated by the consistency of the notion with the method of "vanishing viscosity": the viscosity solution of (Eq. 2) can be obtained as the limit for $\epsilon \rightarrow 0$ of the classical solutions of

Viscosity Solution, Fig. 1

Distance functions in 1D and 2D: (a) three examples of weak solutions of (Eq. 2) The red curve corresponds to the viscosity solution. (b)

Example of a closed curve in the plan. (c) Distance function (represented as a color map) to the curve displayed in (b). To improve the visibility, the distance function is only displayed inside the curve. As usually, one can distinguish strong edges we have partly highlighted by green curves. This gives the skeleton of the shape. (d) 3D representation of the function distance (c). The distance function is the viscosity solution of the Eikonal Equation (Eq. 3), with $F(x) = 1$ and $u(x) = 0$ on the curve



$$-\epsilon u''(x) + |u'(x)| - 1 = 0 \text{ for } x \in (-1, 1), u(\pm 1) = 0 \quad (4)$$

(the term ϵ has the physical meaning of a viscosity coefficient).

The main characteristics of the notion of viscosity solution are:

- (i) A very efficient and flexible way to prove uniqueness theorems and comparison principles
- (ii) General existence results obtained via the adaptation of the classical Perron's method, by approximation arguments (such as the vanishing viscosity method in (Eq. 3)), by means of representation formulas (via dynamic programming methods in optimal control theory), etc.
- (iii) The stability of the notion of viscosity solution with respect to the uniform convergence, and its generalizations, which allows to prove, for example, the convergence of numerical schemes
- (iv) The correct formulation of various boundary conditions, including the classical Dirichlet, Neumann, and oblique derivative conditions

For good accounts of the viscosity solution theory, we refer to [4, 5, 7, 13]. Finally, let us note that the notion and the theory of viscosity solutions have been extended to a large class of partial differential equations. In particular, a number of results allow us to deal with second-order equations [10], degenerate equations [9], and integro-differential equations [1, 8]. The application to convergence of numerical schemes is also very important; see, for example, [6] and the appendix by Falcone in [5].

Application

The range of applications of the notion of viscosity solution is enormous, including common class of partial differential equations such as evolutive problems and problems with boundary conditions, equations arising in optimal control theory (the Hamilton-Jacobi-Bellman equation), differential games (the Isaacs' equation), second-order equations arising in stochastic optimal control and stochastic differential games,

and geometric equations (mean curvature and Monge-Ampere equations).

In computer vision, it has various applications. In particular, the distance functions and the Eikonal equations are widely used. Nowadays, thanks to the links between the viscosity solutions and the optimal control theory [5], one can easily prove that the distance functions correspond to the viscosity solutions of the Eikonal equations, which, moreover, provide various convenient tools for computing them. Also, all these notions have played an important role in shape representation [14, 23], in morphology [2, 21], in tractography [12, 15, 18], and in general, in image processing [3, 18, 22]. Furthermore, they are intensively used in the level set framework where the curves and the surfaces are represented by their signed distance functions [17, 22]. The latter framework is used extensively, for example, in segmentation and 3D reconstruction.

Another main application of the notion of viscosity solution is to shape-from-shading problems, which give rise to first-order differential and integro-differential equations of Hamilton-Jacobi type; see [16, 20], and the entry *Shape from Shading*. A natural question in this context is why the viscosity solutions provide suitable solutions to this specific problem. In other words, why would the viscosity solutions have more sense than any other weak solution? In fact, here, the values of the viscosity solutions mainly come from its amazing stability combined with its consistency with the classical solution. To be more clear, let us consider the shape-from-shading problem with a continuous image I of a real scene. In such a case, to be physically plausible, the real surface u^* behind this image must be smooth (C^1); otherwise, any infinitesimal displacement of the light direction would break this continuity property. Let I^* be the virtual image generated by the considered image formation model with surface u^* . I^* is necessary close to I , if not this would mean that the considered model is not appropriate. Then, thanks to the stability properties, the unique (To be well posed the problem in the viscosity sense, we can assume that we have adequate boundary constraints, for example, only Soner constraints on the boundary of the image if we consider the model of [19].) viscosity solution u to the shape-from-shading Hamilton-Jacobi equation associated with the real image I is close to u^* , because u^* is the viscosity solution to the same equation in which we replace

I by I^* (since it is also a solution in the classical sense). In other words, among all the weak solutions of the considered shape-from-shading equation (which has no solution in the classical sense with the real image I because of the modeling errors and the noise), the viscosity solution is necessarily close to the real surface which has been photographed.

References

1. Alvarez O, Tourin A (1996) Viscosity solutions of nonlinear integro-differential equations. Annales de l'institut Henri Poincaré (C) Analyse non linéaire 13(3): 293–317
2. Arehart A, Vincent L, Kimia BB (1993) Mathematical morphology: the Hamilton-Jacobi connection. In: Proceedings of ICCV, Berlin. IEEE Computer Society, pp 215–219
3. Aubert G, Kornprobst P (2006) Mathematical problems in image processing: partial differential equations and the calculus of variations. Applied mathematical sciences. Springer, New York/Secaucus
4. Bardi M, Crandall MG, Evans LC, Soner HM, Souganidis PE (1997) Viscosity solutions and applications. Lecture notes in mathematics, vol 1660. Springer, Berlin. Lectures given at the 2nd C.I.M.E. Session held in Montecatini Terme, June 12–20, 1995. Dolcetta IC, Lions PL (eds). Fondazione C.I.M.E. (C.I.M.E. Foundation)
5. Bardi M, Capuzzo-Dolcetta I (1997) Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations. Systems & control: foundations & applications. Birkhäuser, Boston. With appendices by Maurizio Falcone and Pierpaolo Soravia
6. Barles G, Souganidis PE (1991) Convergence of approximation schemes for fully nonlinear second order equations. Asymptot Anal 4(3):271–283
7. Barles G (1994) Solutions de viscosité des équations de Hamilton-Jacobi. Volume 17 of Mathématiques & Applications (Berlin) (Mathematics & Applications). Springer, Paris
8. Barles G, Imbert C (2008) Second-order elliptic integro-differential equations: viscosity solutions' theory revisited. Annales de l'Institut Henri Poincaré (C) Non Linear Analysis 25(3):567–585
9. Camilli F, Siconolfi A (1999) Maximal subsolutions for a class of degenerate Hamilton-Jacobi problems. Indiana Univ Math J 48(3):1111–1132
10. Crandall MG, Ishii H, Lions PL (1992) User's guide to viscosity solutions of second order partial, differential equations. Bull Am Math Soc 27:1–67
11. Crandall MG, Lions P-L (1983) Viscosity solutions of Hamilton-Jacobi equations. Trans Amer Math Soc 277(1):1–42
12. Donnell LO, Haker S, Westin C-F (2002) New approaches to estimation of white matter connectivity in diffusion tensor mri: elliptic pdes and geodesics in a tensor-warped space. In: Dohi T, Kikinis R (eds) Medical image

- computing and computer-assisted intervention MICCAI 2002. Lecture notes in computer science, vol 2488. Springer, Berlin/Heidelberg, pp 459–466
13. Fleming WH, Soner HM (2006) Controlled Markov processes and viscosity solutions. Stochastic modelling and applied probability, 2nd edn., vol 25. Springer, New York
 14. Kimia BB, Tannenbaum AR, Zucker SW (1994) Shapes, shocks, and deformations i: the components of two-dimensional shape and the reaction-diffusion space. *Int J Comput Vis* 15:189–224
 15. Lenglet C, Prados E, Pons J-P, Deriche R, Faugeras O (2009) Brain connectivity mapping using Riemannian geometry, control theory and PDEs. *SIAM J Imaging Sci* 2:285–322
 16. Lions P-L, Rouy E, Tourin A (1993) Shape-from-shading, viscosity solutions and edges. *Numer Math* 64(3): 323–353
 17. Osher S, Fedkiw R (2002) Level set methods and dynamic implicit surfaces. Applied Mathematics 153, Springer Verlag, New York
 18. Pechaud M (2009) Shortest paths calculations, and applications to medical imaging. PhD thesis, University of Paris Diderot
 19. Prados E, Faugeras O (2005) Shape from shading: a well-posed problem? In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR'05), San Diego, California, vol II, pp 870–877. IEEE
 20. Prados E, Faugeras O (2006) Shape from shading. In: Handbook of mathematical models in computer vision. Springer, New York, pp 375–388
 21. Sapiro G, Kimia BB, Kimmel R, Shaked D, Bruckstein AM (1993) Implementing continuous-scale morphology. *Pattern Recognit* 26(9):1363–1372
 22. Sethian JA (1999) Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science. Cambridge University Press, Cambridge
 23. Tari ZSG, Shah J, Pien H (1997) Extraction of shape skeletons from grayscale images. *Comput Vis Image Underst* 66:133–146

Visibility Enhancement in Bad Weather

► Dehazing and Defogging

Vision-Based Control

► Visual Servoing

Vision-Based Feedback

► Visual Servoing

Visual Cognition

David Vernon

Informatics Research Centre, University of Skövde, Skövde, Sweden

Synonyms

[Visual Inference](#)

Related Concepts

► [Cognitive Vision](#)

Definition

Visual cognition is the branch of psychology that is concerned with combining visual data with prior knowledge to construct high-level representations and make unconscious decisions about scene content [1].

Background

Although the terms visual cognition and cognitive vision are strikingly similar, they are not equivalent. Cognitive vision refers to goal-oriented computer vision systems that exhibit adaptive and anticipatory behavior. In contrast, visual cognition is concerned with how the human visual system makes inferences about the large-scale composition of a visual scene using partial information [1–3].

Theory

Visual cognition, often associated with high-level vision and top-down visual processing, constructs visual entities by collecting perceived parts into coherent wholes, determining which parts belong together. Since the sensory data on which the processes of visual cognition operate are typically incomplete and insufficient to specify the percept of which we are



aware, there are many possible solutions or interpretations. Consequently, additional extraretinal information, often referred to as object information, is used by visual cognition to infer what the percept is.

The entities that are constructed by visual cognition include both static structures, such as perceived surfaces and objects, and dynamic entities that emerge over time, such as patterns of biological motion. The dynamics of these visual entities can be used to infer a causal relationship between events or to attribute some sense of intentionality to the entity. Thus, the unconscious inferences of visual cognition also impact on the construction of a theory of mind for other cognitive agents, i.e., the inference of the goals of other agents [4].

A key role of the representations of visual cognition is their use to communicate with other centers of the brain. Thus, visual cognition provides a bridge to general high-level cognitive function while still making its own independent cognitive inferences. In essence, visual cognition constructs a representation of the visual world which is constantly updated on the basis of new visual data and which encapsulates knowledge about the world in a high-level descriptive manner that can be exchanged with the rest of the brain.

Visual cognition addresses several distinct areas such as visual attention (including spatial attention, selective attention, visual search, change detection, and the control of eye movements) [5–8], short-term and long-term visual memory [9], and object, face, and scene recognition [10, 11].

The processes of visual cognition are held to be principally unconscious, operating rapidly on the flux of visual data sensed by the retina in order to choose the conscious percept of which we become aware. Consequently, visual cognition embraces both the selectivity of visual attention and unconscious inferential decision-making.

Although the primary concern of visual cognition is human visual perception and not computer vision, the two fields share some common ground. For example, many of the theories of visual cognition have their roots in cognitivist psychology which asserts that cognition is intrinsically computational [12, 13]. This has led to several computational models of visual cognition, combining relevant aspects of computer and human vision.

Open Problems

There is some debate in the psychology community as to where one should draw the line between vision and cognition and how sharply one should draw it; for comprehensive discussion, see the paper by Pylyshyn [14], the many commentaries on it (e.g., [15]), and his response [16]. The issue revolves around the cognitive impenetrability of visual perception: whether or not any cognitive functionality such as inference or rationality is involved in visual perception, especially early vision. Cavanagh's recent review [1] suggests that the visual system does have its own independent cognitive processes, quite apart from the more general cognition that occurs in other parts of the brain and with which the processes of visual cognition interact.

References

1. Cavanagh P (2011) Visual cognition. *Vis Res* 51(13):1538–1551
2. Coltheart V (ed) (2010) Tutorials in visual cognition. Macquarie monographs in cognitive science. Psychology Press, London
3. Pinker S (1984) Visual cognition: an introduction. *Cognition* 18:1–63
4. Blakemore S, Decety J (2001) From the perception of action to the understanding of intention. *Nat Rev Neurosci* 2(1):561–567
5. Carrasco M (2011) Visual attention: the past 25 years. *Vis Res* 51(13):1484–1525
6. Rensink RA (2002) Change detection. *Annu Rev Psychol* 53:245–277
7. Simons DJ (2000) Current approaches to change blindness. *Vis Cogn* 7(1–3):1–15
8. Torralba A, Oliva A, Castelhano MS, Henderson JM (2006) Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. *Psychol Rev* 113(4):766–786
9. Deco G, Rolls E (2005) Attention, short term memory, and action selection: a unifying theory. *Prog Neurobiol* 76: 236–256
10. Spelke ES (1990) Principles of object perception. *Cogn Sci* 14:29–56
11. Oliva A, Torralba A (2006) Building the gist of a scene: the role of global image features in recognition. *Prog Brain Res* 155:23–36
12. Newell A (1990) Unified theories of cognition. Harvard University Press, Cambridge
13. Newell A, Simon HA (1976) Computer science as empirical inquiry: symbols and search. *Commun Assoc Comput Mach* 19:113–126. Tenth turing award lecture, ACM, 1975

14. Pylyshyn ZW (1999) Is vision continuous with cognition? The case for cognitive impenetrability of visual perception. *Behav Brain Sci* 22(3):341–365
15. Cavanagh P (1999) The cognitive impenetrability of cognition. *Behav Brain Sci* 22(3):370–371
16. Pylyshyn ZW (1999) Vision and cognition: how do they connect? *Behav Brain Sci* 22(3):401–414

Visual Cortex Models for Object Recognition

Tomaso Poggio and Shimon Ullman
Department of Brain and Cognitive Sciences,
McGovern Institute, Massachusetts Institute
of Technology, Cambridge, MA, USA

Related Concepts

► [Object Class Recognition \(Categorization\)](#)

Definition

Visual cortex model-based methods aim to develop algorithms for object detection, representation and recognition that attempt to mimic human visual systems.

Background

Object recognition is difficult Like other natural tasks that our brain performs effortlessly, visual recognition has turned out to be difficult to reproduce in artificial systems. In its general form, it is a highly challenging computational problem which is likely to play a significant role in eventually making intelligent machines. Not surprisingly, it is also an open and key problem for neuroscience.

Within object recognition, it is common to distinguish two main tasks: identification, for instance, recognizing a specific face among other faces, and categorization, for example, recognizing a car among other object classes. We will discuss both of these tasks below, and use “recognition” to include both.

Models of the visual cortex Over the last two decades, some of the best performing recognition systems have come from research at the intersection

of computational neuroscience and computer vision. Recent models of visual cortex based directly on known functional anatomy [20, 21] and building on earlier attempts (e.g., [1, 6, 15, 19, 22–24]) were able to account for and predict a number of physiological data from areas of the ventral stream from V1 and V2 to V4 and IT. This family of models was able to mimic human performance in rapid categorization tasks [20]. Surprisingly, some of these models of visual cortex were among the best computer vision systems at the time [16–18, 21].

Computer Vision and the Visual Cortex: Fundamental Differences

The recent past has shown convergence of computational schemes and brain modeling. There still are, however, major differences between models and the cortex, as well as large differences in performance between models and the brain. We will discuss below two examples of prominent features of cortical structure which have only a minor role in current computational models.

Why Hierarchies

The organization of visual cortex is hierarchical, with features of increasing complexity represented at successive layers. Models of the visual cortex have naturally adopted hierarchical structures. In contrast, in computer vision, the large majority of current schemes are nonhierarchical, with no clear difference in performance between hierarchical and nonhierarchical models. Some computational schemes, however, may be implicitly hierarchical and possibly derive some of their power from their hierarchical organization. For instance, SIFT [13] can be regarded as a three-layer network with the output roughly corresponding to intermediate units in a hierarchical cortical model [19]. What is the possible advantage of hierarchical visual representations, and can artificial systems benefit from adopting such representations?

Scale and position invariance One possible role of feature hierarchies is the need to achieve a useful trade-off between selectivity to complex patterns and sufficient tolerance for changes in position and scale, as seen in the response of IT neurons [10–12]. While scale and position invariance can be achieved quite

readily in computer vision systems by sequentially scanning the image at different positions and scales, such a strategy appears unlikely to be realized in neural hardware. When properly measured, scale and position tolerance for new objects is less than originally claimed [2], but still substantial [8, 12]: for at least some of the cells in AIT, position tolerance is on the order of 2–4 degrees in the fovea and scale invariance is on the order of a factor of 2–4, which is remarkable large. It is still not entirely clear how such generalization can be achieved without training for different positions and sizes for each object. It appears possible that hierarchical representations make it possible to transfer invariant recognition from previously trained objects to novel objects by the reuse of shared parts at multiple levels.

A second possible advantage of hierarchical representations has to do with efficiency – computational speed and use of computational resources. For instance, hierarchy may increase the efficiency of dealing with multiple classes in parallel, by allowing the use of shared features at multiple levels. An increase in efficiency may also be related to the issue of *sample complexity*. Hierarchical architectures in which each layer is adapted through learning to properties of the visual world may reduce the complexity of the learning task and thus the overall number of labeled examples required for training. Finally, hierarchies also offer an advantage in not only obtaining recognition of the object as a whole but also recognizing and localizing parts and subparts at multiple levels, such as a face together with the eyes, nose, mouth, eyebrow, nostril, upper lip, and the like (see [5]).

Feed Forward vs. Back Projections

A feed-forward architecture from V1 to prefrontal cortex, in the spirit of the Hubel and Wiesel simple-complex hierarchy, seems to account for several properties of visual cells. In particular, recent “readout” experiments measuring information that could be read out from populations of IT cells [8] confirm previous estimates that, after about ~ 100 ms from onset of the stimulus, performance of the classifier was essentially at its asymptotic performance during passive viewing.

In addition, feed-forward models also appear to account for recognition performance of human subjects for images flashed briefly and followed by a mask [20–22].

The evidence suggests, therefore, that a feed-forward process is sufficient for a fast initial recognition phase, during which primates can already complete difficult recognition tasks involving “what” is in the image. What, then, is the role of the extensive anatomical back projections in the primate visual system?

Their role may be restricted to learning, but we believe it is broader. Even when the feed-forward projections by themselves may be capable of answering the “what” question of vision, the back projections may play a role in answering the detailed question of “what is where” [14]. This proposal is similar to previous ideas suggesting that visual cortex follows a hypothesis-and-verification strategy [3, 7] or a Bayesian inference procedure in which top-down priors are used to compute a set of mutually consistent conditional probabilities at various stages of the visual pathway [9]. A recent model [5] along these lines demonstrated the use of initial classification using a bottom-up sweep, followed by precise localization of the object and its parts and subparts by a top-down pass.

Top-down pathways in the visual cortex also include the dorsal stream and connections between the dorsal and the ventral stream that are likely to be involved in attentional effects. A Bayesian model [4], which takes into account these bottom-up and top-down signals, performs well in recognition tasks and predicts some of the main psychophysical and physiological properties of attention. For natural images, the top-down signal improves object recognition performance and predicts human eye fixations well. The top-down flow of information combined with a hierarchical representation allows the system to answer not only the *what* question, that is, to perform object identification and categorization, but also the *what is where* question, that is, identification and localization at multiple levels. This is a useful addition, but at the same time we do not believe that the process of vision can be fully characterized in terms of answering “what is where” [14]. For example, humans can recognize subtle aspects of actions, goals, and social interactions at a level which is far beyond the capabilities of our present algorithms. They also can answer essentially any reasonable question, beyond what and where, on any given image – in a kind of Turing test for vision. The top-down pathway is likely to play an important role for this broader range of visual tasks.

Discussion: Future

We briefly consider two problem domains for future studies. The first focuses on how to close the gap between computer and human vision in the tasks considered above – object categorization and identification. The second part considers broader aspects of vision and its roots in evolution.

Closing the Performance Gap

One general question regarding possible improvements in visual recognition is whether recognition is obtained by multiple specialized mechanisms or by a uniform scheme applied to different recognition tasks. For example, suggestions have been made that general categorization and individual recognition may be subserved by different mechanisms or that face recognition may depend on special mechanisms, not used for other object categories. It appears to us that the underlying computational problems in different recognition tasks are similar and can therefore be approached by the same general scheme, applied to different training sets (and possibly implemented by more than a single neuronal mechanism). It is also possible that certain cortical regions could specialize in specific categories (such as faces or locations) not because they implement different recognition strategies, but to facilitate selective readout by other cortical regions. The basic recognition scheme could be augmented, however, by specialized mechanisms, dealing with special cases and exceptions. The full system could then be a combination of a scheme that may be characterized as rule based, which can capture the main properties of a category and generalize broadly to novel examples, and a memory-based recognition scheme, which can deal with atypical cases and exceptions to the rule-based scheme.

We next consider future directions which we think could play a useful role in bringing the performance of artificial recognition models closer to the performance level of human vision. These are not the only possible routes for closing the performance gap, but they provide examples of promising general directions motivated by human perception that could usefully be incorporated into artificial systems.

Continuous learning of rich models In current computational schemes, a model for the object or category of interest is constructed during a learning stage and then used for recognition. In contrast, the primate visual system shows continuous plasticity and can continue to learn when confronted with new examples. The disadvantage of a fixed limited training stage is that the resulting object model may remain too simple. A visual category often contains a core of typical examples but also a large number of possible variations, atypical members, and counterexamples. An object can be recognized by its overall shape, but also by small distinguishing parts, and both aspects need to be included in its representation. To achieve human-level performance, it appears therefore that it will be necessary to construct rich object models, learned continuously from a large number of examples.

Such use of continuous learning raises interesting computational challenges: new methods will be required to learn from errors, and to continuously modify an existing representation based on new incoming information, possibly combining rule-based and memory-based mechanisms mentioned above.

Integrating segmentation and recognition Recognition and segmentation are related tasks in the perception of objects: we can usually recognize the object and at the same time identify in the image the precise region containing the object of interest. Historically, segmentation and recognition were treated in computer vision as sequential processes: figure-ground segmentation first identifies in the image a region likely to correspond to a single object; recognition processes are subsequently applied to the selected region to identify the segmented object. More recently, computational models started to treat the two tasks together, performing object segmentation not only in a bottom-up manner based on image properties, but also in a top-down manner based on object representations stored in memory. This led to substantial progress in object segmentation; however, most current recognition systems do not include segmentation as an integral part of the recognition process. It seems to us that recognition and segmentation are closely linked tasks, and their solutions constrain each other. This integration appears to be supported by considerable physiological and psychophysical evidence [25, 26]. A closer



integration of recognition and segmentation at both the object and part levels is likely therefore to improve the recognition of objects and their parts.

A Greater Challenge: Vision and Evolution

The brain uses vision, together with other senses, to obtain knowledge about the world and act upon it. This knowledge goes beyond object recognition and categorization: vision is also used, for example, to recognize actions performed by agents in the surrounding environment as well as their goals and social interactions.

It seems to us that these broader aspects of visual recognition cannot be efficiently handled by simple extension of existing recognition methods. It is likely that in addition to the general learning mechanisms currently used in object recognition models, the brain also uses specialized mechanisms, which have evolved to focus on and extract information required for making judgments about actions, goals, social interactions, and the like. Innate structures and circuits in the brain by themselves do not incorporate full solutions to these challenging problems, but are more likely to provide useful constraints and initial biases which later lead, guided by learning from the environment, to powerful specific mechanisms.

A general broad question for future studies is therefore the nature of the innate machinery used by the visual system, its genetic encoding, and how the combination of innate machinery and learning from the environment leads to our understanding of the visual world.

References

1. Amit Y, Mascaro M (2003) An integrated network for invariant visual detection and recognition. *Vis Res* 43(19): 2073–2088
2. Bruce C, Desimone R, Gross C (1981) Visual properties of neurons in a polysensory area in the superior temporal sulcus of the macaque. *J Neurophysiol* 46:369–384
3. Carpenter G, Grossberg S (1987) A massively parallel architecture for a self-organizing neural pattern recognition. *Comput Vis Graph Image Process* 37:54–115
4. Chikkerur S, Serre T, Poggio T (2009) A Bayesian inference theory of attention: neuroscience and algorithms, MIT-CSAIL-TR-2009-047/CBCL-280. Massachusetts Institute of Technology, Cambridge
5. Epshtein B, Lifshitz I, Ullman S (2008) Image interpretation by a single bottom-up top-down cycle. *PNAS* 105(38):14298–14303
6. Fukushima K (1975) Cognition: a self-organizing multilayered neural network. *Biol Cyber* 20(3–4):121–136
7. Hawkins J, Blakeslee S (2004) On intelligence. Times Books, New York
8. Hung C, Kreiman G, Poggio T, DiCarlo J (2005) Fast read-out of object identity from macaque inferior temporal cortex. *Science* 310:863–866
9. Lee TS, Mumford D (2003) Hierarchical Bayesian inference in the visual cortex. *J Opt Soc Am A Opt Image Sci Vis* 20(7):1434–1448
10. Logothetis NK, Sheinberg DL (1996) Visual object recognition. *Ann Rev Neurosci* 19:577–621
11. Logothetis NK, Pauls J, Bülthoff HH, Poggio T (1994) View-dependent object recognition by monkeys. *Curr Biol* 4:401–413
12. Logothetis NK, Pauls J, Poggio T (1995) Shape representation in the inferior temporal cortex of monkeys. *Curr Biol* 5:552–563
13. Lowe D (2004) Distinctive image features from scale-invariant key-points. *Int J Comput Vis* 60(2):91–110
14. Marr D (1982) Vision: a computational investigation into the human representation and visual information. W.H. Freeman, New York
15. Mel BW (1997) SEEMORE: combining color, shape and texture histogramming in a neurally-inspired approach to visual object recognition. *Neural Comput* 9: 777–804
16. Mutch J, Lowe D (2006) Multiclass object recognition using sparse, localized features. In: Proceedings of the IEEE conference on computer vision pattern recognition (CVPR), New York
17. Mutch J, Lowe DG (2008) Object class recognition and localization using sparse features with limited receptive fields. *Int J Comput Vis (IJCV)* 80(1):45–57
18. Pinto N, Doukhan D, DiCarlo JJ, Cox DD (2009) A high-throughput screening approach to discovering good forms of biologically inspired visual representation. *PLoS Comput Biol* 5(11):1–12
19. Riesenhuber M, Poggio T (1999) Hierarchical models of object recognition in cortex. *Nat Neurosci* 2: 1019–1025
20. Serre T, Oliva A, Poggio T (2007) A feedforward architecture accounts for rapid categorization. *Proc Natl Acad Sci* 104(15):6424
21. Serre T, Kreiman G, Kouh M, Cadieu C, Knoblich U, Poggio T (2007) A quantitative theory of immediate visual recognition. *Prog Brain Res* 165:33–56
22. Thorpe S (2002) Ultra-rapid scene categorisation with a wave of spikes. In: Second international workshop on biologically motivated computer vision (BMCV), Tübingen, pp 1–15
23. Wallis G, Rolls ET (1997) A model of invariant object recognition in the visual system. *Prog Neurobiol* 51: 167–194
24. Wersing H, Koerner E (2003) Learning optimized features for hierarchical models of invariant recognition. *Neural Comput* 15(7):1559–1588

25. Zemel RS, Behrmann M, Mozer MC, Bavelier D (2002) Object recognition processes can and do operate before figure-ground organization. *Exp Psychol* 28(1):202–217
26. Zhang J, Zisserman A (2006) Dataset issues in object recognition. In: Ponce J, Hebert M, Schmid C, Zisserman A (eds) *Toward category-level object recognition*. Springer, Berlin, pp 29–48
27. Zhou H, Howard S, Friedman HS, von der Heydt R (2000) Coding of border ownership in monkey visual cortex. *J Neurosci* 20(17):6594–6611

The term “visual hull” was coined by Laurentini [1, 2] who studied the properties of hulls obtained from an infinite number of cameras. However, the underlying concepts – the intersection of viewing cones – can be tracked back to the early 1970s (e.g., [3]). Sometimes, the term “visual hull” is restricted to the theoretical case of infinite camera count; hulls computed from a finite number of cameras are then regarded as approximations.

Visual Hull

David C. Schneider
Image Processing Department, Fraunhofer Heinrich Hertz Institute, Berlin, Germany

Synonyms

[Shape from Silhouette](#)

Definition

The visual hull of a three-dimensional object is a bounding volume of the object which is computed from the object’s silhouettes in the images of multiple calibrated cameras by intersecting the object’s viewing cones.

Background

The concept of visual hull is based on two intuitions: Firstly, the shape of a three-dimensional object is bounded by the silhouette of its projection in an image. Secondly, if several projections from multiple viewpoints are available, these constraints can be combined to obtain an approximation of the object’s shape. The practical relevance of visual hull algorithms lies in the fact that they can compute an approximate 3D shape using only silhouettes and calibration data. These are often easier and faster to obtain than image-to-image correspondences, especially for camera configurations with few cameras and/or wide baselines. Visual hull computation is the prototypical example of a shape-from-silhouette method.

Theory

Definition

The visual hull of an object can be defined either as a surface or as a volume. The following definition is volumetric. Assume that the object can be represented as a surface \mathcal{S} in three-dimensional space. Further assume that there are n cameras viewing the object. The *viewing cone* \mathcal{C}_i of \mathcal{S} with respect to camera $i \in 1 \dots n$ is a generalized cone defined as the union of all rays that originate from the camera’s optical center \mathbf{c}_i and pass through any point \mathbf{p} on \mathcal{S} ([Fig. 1](#), right):

$$\mathcal{C}_i = \bigcup_{\mathbf{p} \in \mathcal{S}, \lambda \geq 0} \lambda \mathbf{p} + (1 - \lambda) \mathbf{c}_i$$

The *visual hull* \mathcal{H} of \mathcal{S} is the intersection of all viewing cones ([Fig. 1](#), left):

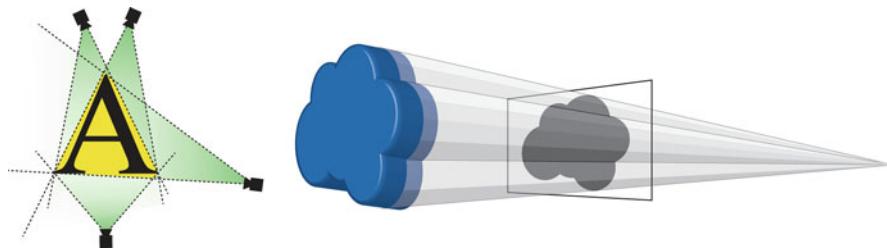
$$\mathcal{H} = \bigcap_{i=1 \dots n} \mathcal{C}_i$$

Seen as a surface, the visual hull is the boundary of this volume.

Properties

The visual hull bounds the shape it is constructed from, that is, the volumetric intersection of the hull and the original shape is the shape itself. How closely an object’s true 3D shape can be approximated by its visual hull depends on two factors:

- (1) *The geometry of the object.* The visual hull can represent some but not all concavities of an object. Intuitively, a concavity can be represented if (and as far as) there exists a line through the concavity which does not intersect the object. Or, equivalently, it can be represented if and as far as there exists a location and orientation for a camera such



Visual Hull, Fig. 1 *Left:* The visual hull (yellow) of the object (A) is the intersection of the four viewing cones (green). *Right:* The viewing cone of a 3D object and its relation to the object’s silhouette (mask) in an image

that the concavity in the object is represented as a concavity in the silhouette of the object in the camera image. A straightforward example of a representable concavity is the handle of a tea cup. The inside of the cup, on the other hand, is not representable (unless a camera is placed inside the cup).

- (2) *The number and placement of the cameras used for computing the hull.* A high count and a regular distribution of cameras are beneficial. With a finite number of cameras, the visual hull typically has a “piecewise extruded” look Fig. 2 shows an example of representable on non-representable geometry in a visual hull reconstruction.

Algorithms

The above definition of the viewing cone only refers to *whether* a ray intersects the surface and not *where*. Therefore, the viewing cones can be computed without knowledge of the shape S as long as the set of intersecting rays can be computed. Whether a ray of a specific view intersects the surface can be determined from the view’s camera matrix and a foreground-background mask, or silhouette, which classifies each pixel as showing a part of S or the background. Therefore, the visual hull is a shape approximation which can be computed solely from calibrated cameras and a foreground-background segmentation.

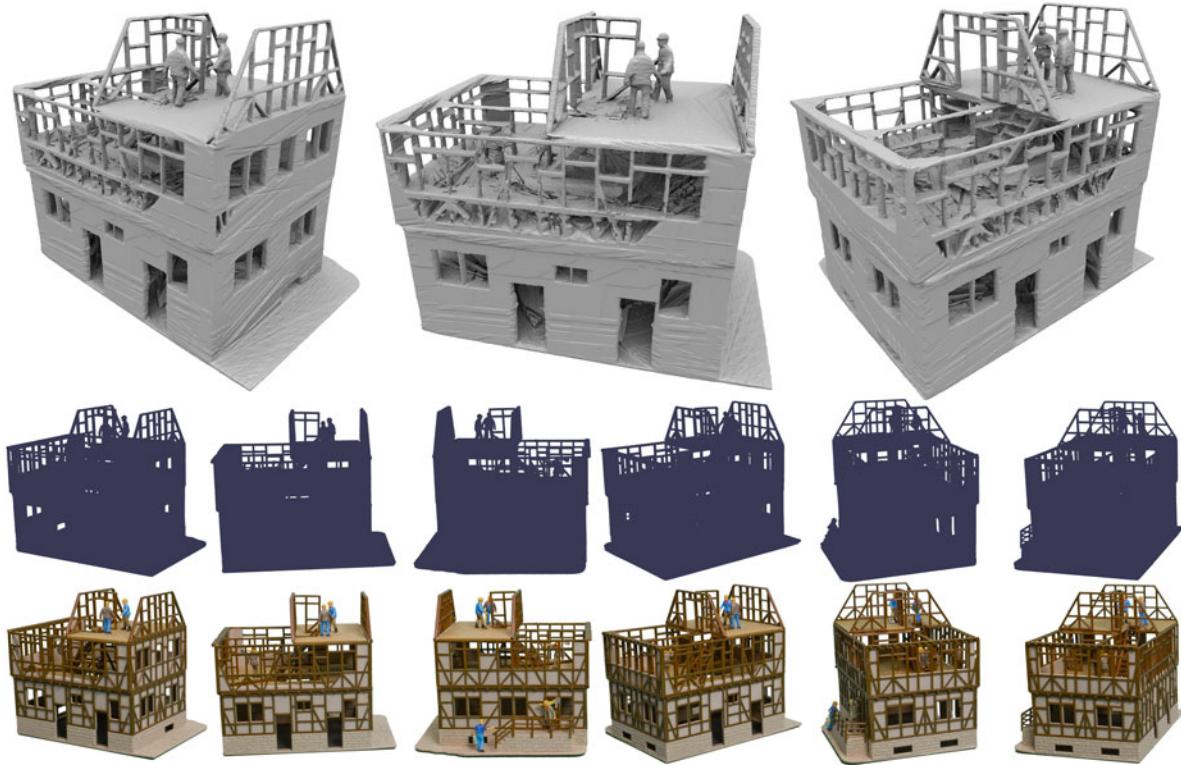
In contrast to the above definition, it is easier to project voxels from the volume into the images than to shoot rays into the volume. The most simple visual hull algorithm is the following: First, all voxels of the volume are marked as “occupied.” Then the voxels are traversed and the center of each voxel is projected into all available segmentation masks. A voxel is marked as “unoccupied” if its center projects to a background pixel in *any* of the masks.

For high volume resolutions, traversing all voxels is inefficient with respect to memory usage and computation time. Therefore, many algorithms represent the volume using a hierachic spatial data structure such as an octree, for example, [4, 5]. When octree cells are projected into the segmentation masks, their spatial extent must be accounted for: An octree cell must be subdivided if any of its projections spans foreground as well as background pixels.

While the volumetric approach makes hull computation relatively simple, it may complicate the further use of the visual hull itself. For many applications, such as rendering on graphics hardware or CAD, a polyhedral surface representation is required. Computing a surface from volume data is a nontrivial task which is often approached with the marching cubes algorithm. The resulting surfaces, however, may require further processing to reduce their high polygon count. This has led to the development of algorithms which directly compute a polyhedral representation of the visual hull, for example, [6, 7].

Application

The visual hull is often used as an initialization and/or as a constraint for 3D reconstruction algorithms. Since the visual hull bounds the true shape, the true shape can be found by removing voxels from the hull. Hull computation, on the other hand, ignores consistency of appearance between multiple views (“photoconsistency”). Therefore, volumetric 3D reconstruction algorithms often aim at minimizing a photoconsistency-based error by removing voxels from the hull. Early examples are space carving [8] or the multi-hypothesis algorithm of [9]. Visual hulls are also used to initialize non-volumetric algorithms such as patch-based stereo [10].



Visual Hull, Fig. 2 Renderings of the visual hull of a toy house (top row). The hull was computed from the masks (middle row) of 216 turntable images of the object (bottom row). Note that

The visual hull is sometimes used directly as shape approximation (“proxy”) for image-based rendering and visualization. This approach is particularly successfull for objects where the hull is a good approximation of the actual shape, for example the human body. Hull-based free-viewpoint rendering systems for scenes with humans are described, for example, by [12]. For this type of application, real-time capable hull algorithms have been developed. In [11], for example, new viewpoints are rendered without explicitly representing the geometry of the visual hull.

Obtaining a good segmentation is often a challenge when using visual hull in practice. However, the hull is more sensitive to some types of errors in the masks than others: A voxel is only falsely marked as occupied if the corresponding pixel is falsely classified as foreground in all masks. Conversely, a voxel is falsely marked as unoccupied if a pixel is falsely marked as background in a single mask.

fine structures, like the beams, are well represented in the hull as long as they are individuated in the masks. Surface details, on the other hand, cannot be reproduced by the hull

References

1. Laurentini A (1991) The visual hull: a new tool for contour-based image understanding. In: Proceedings of the 7th scandinavian conference on image analysis. Aalborg, Denmark
2. Laurentini A (1994) The visual hull concept for silhouette-based image understanding. IEEE Trans Pattern Anal Mach Intell 16(2):150–162
3. Baumgart BG (1974) Geometric modeling for computer vision. PhD thesis, Stanford, Stanford, CA, USA
4. Ahuja N, Veenstra J (1989) Generating octrees from object silhouettes in orthographic views. IEEE Trans Pattern Anal Mach Intell 11(2):137–149
5. Szeliski R (1993) Rapid octree construction from image sequences. CVGIP 58:23–32
6. Lazebnik S, Furukawa Y, Ponce J (2007) Projective visual hulls. Int J Comput Vis 74:137–165
7. Franco JS, Boyer E (2009) Efficient polyhedral modeling from silhouettes. IEEE Trans Pattern Anal Mach Intell 31(3):414–427
8. Kutulakos KN, Seitz SM (1999) A theory of shape by space carving. In: Proceedings of the seventh IEEE international computer vision conference, Kerkyra, Greece, vol 1, pp 307–314



9. Eisert P, Steinbach E, Girod B (1999) Multi-hypothesis, volumetric reconstruction of 3-d objects from multiple calibrated camera views. In: Proceedings of the IEEE international conference on acoustics, speech, and signal processing, Phoenix, AZ, USA, vol 6, pp 3509–3512
10. Furukawa Y, Ponce J (2008) Accurate, dense, and robust multi-view stereopsis. *IEEE Trans Pattern Anal Mach Intell* 1:1–14
11. Matusik W, Buehler C, Raskar R, Gortler SJ, McMillan L (2000) Image-based visual hulls. In: SIGGRAPH New Orleans, Louisiana, USA
12. Guillemaut J-Y, Hilton A (2011) Joint Multi-Layer Segmentation and Reconstruction for Free-Viewpoint Video Applications, *Int J Comput Vis* 93(1):73–100

Visual Inference

► [Visual Cognition](#)

Visual Patterns

► [Model-Based Object Recognition](#)

Visual Servoing

François Chaumette
Inria, Rennes, France

Synonyms

[Vision-based control](#); [Vision-based feedback](#)

Related Concepts

► [Pan-Tilt-Zoom \(PTZ\) Camera](#)

Definition

Visual servoing refers to the use of computer vision data as input of real-time closed loop control schemes to control the motion of a dynamic system, a robot typically.

Background

Visual servoing can be seen as sensor-based schema control from a vision sensor. An iterative iteration of the control scheme consists of the 16 following steps: An iteration of the control scheme consists of the following steps:

- Acquire an image.
- Extract some useful image measurements.
- Compute the current value of the visual features used as inputs of the control scheme.
- Compute the error between the current and the desired values of the visual features.
- Update the control outputs, which is usually the robot velocity, to regulate that error to zero, i.e., to minimize its norm.

For instance, for the first example depicted on [Fig. 1](#), the image processing part consists in extracting and tracking the center of gravity of the moving people, the visual features are composed of the two Cartesian coordinates of this center of gravity, and the control schemes computes the pan and tilt velocities so that the center of gravity is as near as possible of the image center despite the unknown motion of the people. In the second example where a camera mounted on a six degrees of freedom robot arm is considered, the image measurements are a set of segments that are tracked in the image sequence. From these measurements and the knowledge of the 3D object model, the pose from the camera to the object is estimated and used as visual features. The control scheme now computes the six components of the robot velocity so that this pose reaches a particular desired value corresponding to the object position depicted in blue on the images.

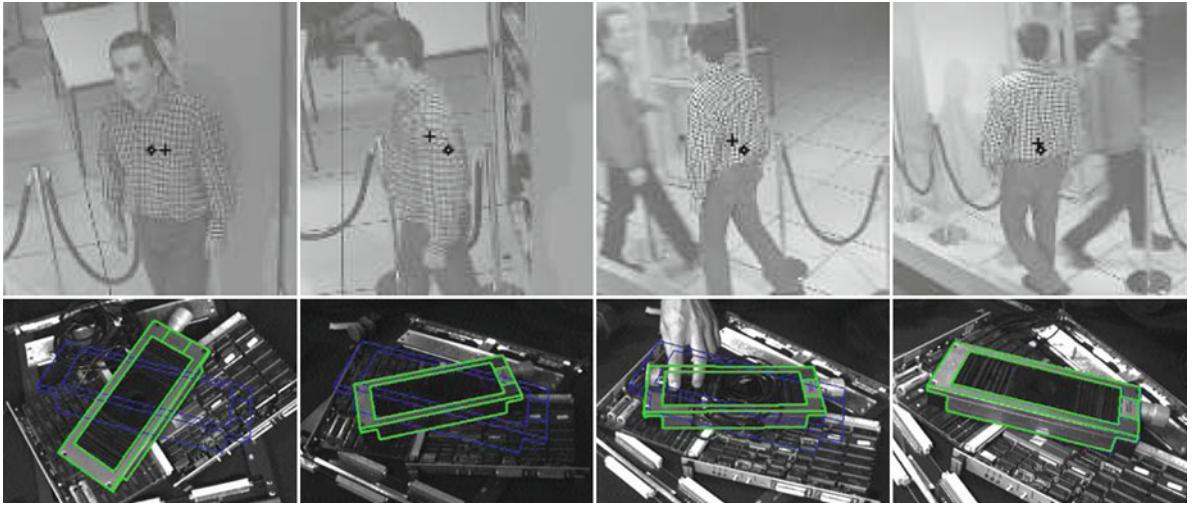
Theory

Main if not all visual servoing tasks can be expressed as the regulation to zero of an error $\mathbf{e}(t)$ which is defined by

$$\mathbf{e}(t) = \mathbf{s}(\mathbf{m}(\mathbf{r}(t)), \mathbf{a}) - \mathbf{s}^*(t). \quad (1)$$

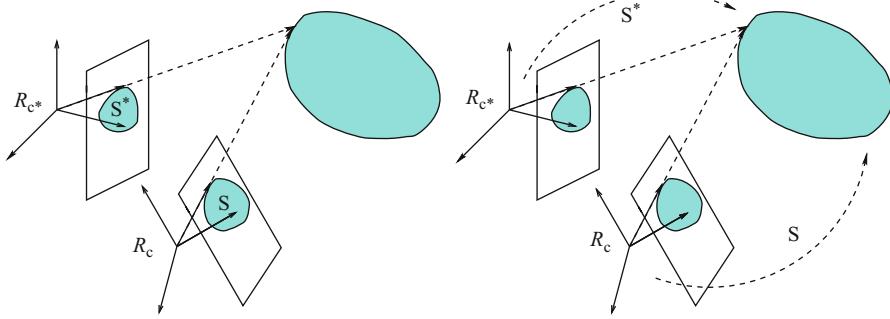
The parameters in [\(Eq. 1\)](#) are defined as follows [1]. The vector $\mathbf{m}(\mathbf{r}(t))$ is a set of image measurements (e.g., the image coordinates of interest points, or the





Visual Servoing, Fig. 1 Few images acquired during two visual servoing tasks: on the *top*, pedestrian tracking using a pan-tilt camera; on the *bottom*, controlling the 6 degrees of freedom

of an eye-in-hand system so that an object appears at a particular position in the image (shown in *blue*)



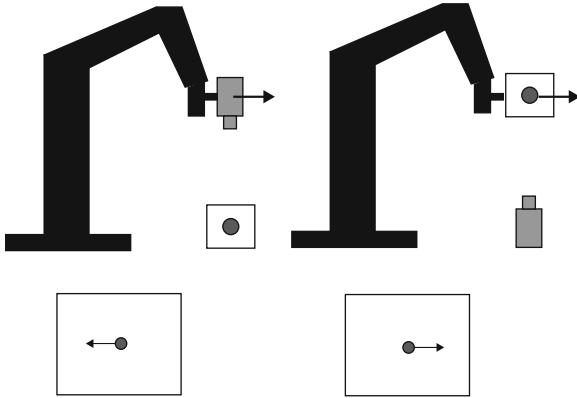
Visual Servoing, Fig. 2 If the goal is to move the camera from frame R_c to the desired frame R_{c^*} , two main approaches are possible: IBVS on the *left*, where the features s and s^* are expressed

in the image, and PBVS on the *right*, where the features s and s^* are related to the pose between the camera and the observed object

area, the center of gravity, and other geometric characteristics of an object). These image measurements depend on the pose $\mathbf{r}(t)$ between the camera and the environment. They are used to compute a vector $\mathbf{s}(\mathbf{m}(\mathbf{r}(t)), \mathbf{a})$ of visual features, in which \mathbf{a} is a set of parameters that represent potential additional knowledge about the system (e.g., coarse camera intrinsic parameters or 3D model of objects). The vector $\mathbf{s}^*(t)$ contains the desired value of the features, which can be either constant in the case of a fixed goal or varying if the task consists in following a specified trajectory.

Visual servoing schemes mainly differ in the way that the visual features are designed. As represented in Fig. 2, the two most classical approaches are named image-based visual servoing (IBVS), in which

\mathbf{s} consists of a set of 2D parameters that are directly expressed in the image [6, 11], and pose-based visual servoing (PBVS), in which \mathbf{s} consists of a set of 3D parameters related to the pose between the camera and the target [11, 12]. In that case, the 3D parameters have to be estimated from the image measurements either through a pose estimation process using the knowledge of the 3D target model, or through a partial pose estimation process using the properties of the epipolar geometry between the current and the desired images, or finally through a triangulation process if a stereovision system is considered. Inside IBVS and PBVS approaches, many possibilities exist depending on the choice of the features. Each choice will induce a particular behavior of the system. There also



Visual Servoing, Fig. 3 In visual servoing, the vision sensor can either be mounted on the robot (eye-in-hand configuration) or observing it (eye-to-hand configuration). For the same robot motion, the motion produced in the image will be opposite from one configuration to the other

exist hybrid approaches, named 2-1/2D visual servoing, which combine 2D and 3D parameters in \mathbf{s} in order to benefit from the advantages of IBVS and PBVS while avoiding their respective drawbacks [10].

The design of the control scheme is based on the link between the time variation of the features and the robot control inputs, which are usually the velocity of the robot joints $\dot{\mathbf{q}}$. This relation is given by

$$\dot{\mathbf{s}} = \mathbf{J}_s \dot{\mathbf{q}} + \frac{\partial \mathbf{s}}{\partial t} \quad (2)$$

where \mathbf{J}_s is the features Jacobian matrix, defined from the equation above similarly as the classical robot Jacobian. For an eye-in-hand system (see the left part of Fig. 3), the term $\frac{\partial \mathbf{s}}{\partial t}$ represents the time variation of \mathbf{s} due to a potential object motion, while for an eye-to-hand system (see the right part of Fig. 3), it represents the time variation of \mathbf{s} due to a potential sensor motion.

As for the features Jacobian, in the eye-in-hand configuration, it can be decomposed as [2]

$$\mathbf{J}_s = \mathbf{L}_s {}^c \mathbf{V}_n \mathbf{J}(\mathbf{q}) \quad (3)$$

where

- $\mathbf{J}(\mathbf{q})$ is the robot Jacobian such that $\mathbf{v}_n = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}$ where \mathbf{v}_n is the robot end effector velocity;

- ${}^c \mathbf{V}_n$ is the spatial motion transform matrix from the vision sensor to the end effector. It is given by

$${}^c \mathbf{V}_n = \begin{bmatrix} {}^c \mathbf{R}_n & [{}^c \mathbf{t}_n]_x {}^c \mathbf{R}_n \\ \mathbf{0} & {}^c \mathbf{R}_n \end{bmatrix} \quad (4)$$

where ${}^c \mathbf{R}_n$ and ${}^c \mathbf{t}_n$ are, respectively, the rotation matrix and the translation vector between the sensor frame and the end effector frame and where $[{}^c \mathbf{t}_n]_x$ is the skew symmetric matrix associated to ${}^c \mathbf{t}_n$. Matrix ${}^c \mathbf{V}_n$ is constant when the vision sensor is rigidly attached to the end effector, which is usually the case. Thanks to the robustness of closed loop control schemes, a coarse approximation of ${}^c \mathbf{R}_n$ and ${}^c \mathbf{t}_n$ is sufficient in practice to get an estimation of ${}^c \mathbf{V}_n$. If needed, an accurate estimation is possible through classical hand-eye calibration methods.

- \mathbf{L}_s is the interaction matrix of \mathbf{s} defined such that $\mathbf{s} = \mathbf{L}_s \mathbf{v}$ where \mathbf{v} is the relative velocity between the camera and the environment.

In the eye-to-hand configuration, the features Jacobian \mathbf{J}_s is composed of [2]

$$\mathbf{J}_s = -\mathbf{L}_s {}^f \mathbf{V}_f {}^c \mathbf{V}_n \mathbf{J}(\mathbf{q}) \quad (5)$$

where

- ${}^f \mathbf{V}_f$ is the spatial motion transform matrix from the robot reference frame to the end effector frame. It is known from the robot kinematics model.
- ${}^c \mathbf{V}_f$ is the spatial motion transform matrix from the camera frame to the reference frame. It is constant as long as the camera does not move. In that case, similarly as for the eye-in-hand configuration, a coarse approximation of ${}^c \mathbf{R}_f$ and ${}^c \mathbf{t}_f$ is usually sufficient to get an estimation of ${}^c \mathbf{V}_f$.

A lot of works have concerned the modeling of the visual features and the determination of the analytical form of the interaction matrix. To give just an example, in the case of an image point with normalized Cartesian coordinates $\mathbf{x} = (x, y)$ and whose 3D corresponding point has depth Z , its interaction matrix is given by [6]

$$\mathbf{L}_x = \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & 1+y^2 & -xy & -x \end{bmatrix} \quad (6)$$

where the three first columns contain the elements related to the three components of the translational velocity, and where the three last columns contain the elements related to the three components of the rotational velocity.

$$\mathbf{L}_{(\rho,\theta)} = \begin{bmatrix} -\cos\theta/Z & -\sin\theta/Z & \rho/Z & (1+\rho^2)\sin\theta & -(1+\rho^2)\cos\theta & 0 \\ \sin\theta/(\rho Z) & -\cos\theta/(\rho Z) & 0 & \cos\theta/\rho & \sin\theta/\rho & -1 \end{bmatrix} \quad (7)$$

which implies that using the Cartesian coordinates or the cylindrical coordinates as visual features will induce a different behavior, that is, a different robot trajectory and thus a different trajectory of the image point.

Currently, the analytical form of the interaction matrix is available for most classical geometrical primitives, such as segments, straight lines, ellipses, moments related to planar objects of any shape, and also coordinates of 3D points and pose parameters. Methods also exist to estimate off-line or online a numerical value of the interaction matrix. Omnidirectional vision sensors, the coupling between a camera and structured light, and even 2D echographic probes have also been studied. A large variety of visual features is thus available for many vision sensors.

Once the modeling step has been performed, the design of the control scheme can be quite simple. The most classical control scheme has the following form [2]:

$$\dot{\mathbf{q}} = -\lambda \widehat{\mathbf{J}_s}^+ (\mathbf{s} - \mathbf{s}^*) + \widehat{\mathbf{J}_s}^+ \frac{\partial \mathbf{s}^*}{\partial t} - \widehat{\mathbf{J}_s}^+ \frac{\partial \mathbf{s}}{\partial t} \quad (8)$$

where λ is a positive gain tuning the rate of convergence of the system and $\widehat{\mathbf{J}_s}^+$ is the Moore-Penrose pseudoinverse of an approximation or an estimation of the features Jacobian. The exact value of all its elements is indeed generally unknown since it depends of the intrinsic and extrinsic camera parameters, as well as of some 3D parameters such as the depth of the point in (Eqs. 6) and (7).

The second term of the control scheme anticipates for the variation of \mathbf{s}^* in the case of a nonconstant desired value. The third term compensates as much as possible a possible target motion in the eye-in-hand case and a possible camera motion in the eye-to-hand case. They are both null in the case of a fixed desired

By just changing the parameters representing the same image point, that is, by using the cylindrical coordinates defined by $\rho = \sqrt{x^2 + y^2}$ and $\theta = \text{Arctan}(y/x)$, the interaction matrix of these parameters has a completely different form [2]:

value and a motionless target or camera. They try to remove the tracking error in the other cases.

Following the Lyapunov theory, the stability of the system can be studied [1]. Generally, visual servoing schemes can be demonstrated to be locally asymptotically stable (i.e., the robot will converge if it starts from a local neighborhood of the desired pose) if the errors introduced in $\widehat{\mathbf{J}_s}$ are not too strong. Some particular visual servoing schemes can be demonstrated to be globally asymptotically stable (i.e., the robot will converge whatever its initial pose) under similar conditions.

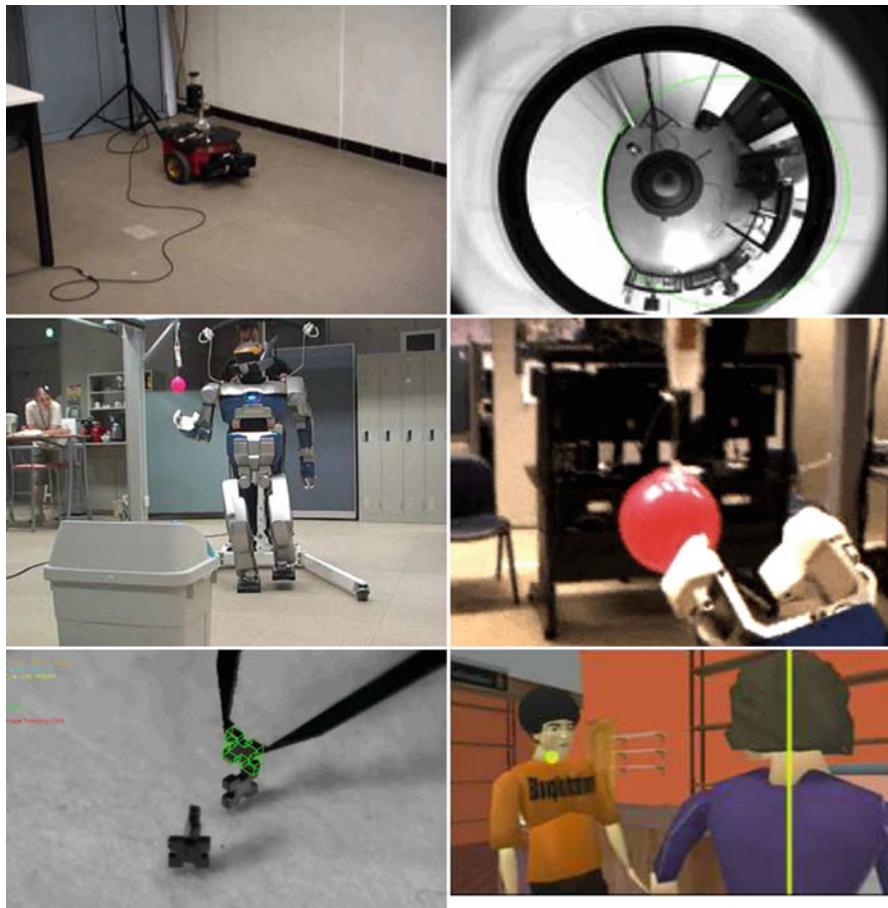
Finally, when the visual features do not constrain all the robot degrees of freedom, it is possible to combine the visual task with supplementary tasks such as joint limits avoidance or the visibility constraint (to be sure that the target considered will always remain in the camera field of view). In that case, the redundancy framework can be applied and the error to be regulated to zero has the following form:

$$\mathbf{e} = \widehat{\mathbf{J}_s}^+ (\mathbf{s} - \mathbf{s}^*) + (\mathbf{I} - \widehat{\mathbf{J}_s}^+ \widehat{\mathbf{J}_s}) \mathbf{e}_2 \quad (9)$$

where $(\mathbf{I} - \widehat{\mathbf{J}_s}^+ \widehat{\mathbf{J}_s})$ is a projection operator on the null space of the visual task so that the supplementary task \mathbf{e}_2 will be achieved at best under the constraint that the visual task is realized [6]. A similar control scheme to (Eq. 8) is now given by

$$\dot{\mathbf{q}} = -\lambda \mathbf{e} - \frac{\partial \mathbf{e}}{\partial t} \quad (10)$$

This scheme has for instance been applied for the first example depicted on Fig. 4 where the rotational motion of the mobile robot is controlled by vision while its translational motion is controlled by the odometry to move at a constant velocity.



Visual Servoing, Fig. 4 Few applications of visual servoing: navigation of a mobile robot to follow a wall using an omnidirectional vision sensor (*top line*), grasping a ball with a humanoid

robot (*middle line*), assembly of MEMS and film of a dialogue within the constraints of a script in animation (*bottom line*)

Application

Potential applications of visual servoing are numerous. It can be used as soon as a vision sensor is available and a task is assigned to a dynamic system to control its motion. A nonexhaustive list of examples is as follows (see Fig. 4):

- The control of a pan-tilt-zoom camera, as illustrated in Fig. 1 for the pan-tilt case
- Grasping using a robot arm
- Locomotion and dexterous manipulation with a humanoid robot
- Micro- or nanomanipulation of MEMS or biological cells
- Pipe inspection by an underwater autonomous vehicle

- Autonomous navigation of a mobile robot in indoor or outdoor environment
- Aircraft landing
- Autonomous satellite rendezvous
- Biopsy using ultrasound probes or heart motion compensation in medical robotics
- Virtual cinematography in animation

References

1. Chaumette F, Hutchinson S (2006) Visual servo control, part I: basic approaches. *IEEE Robot Autom Mag* 13(4):82–90
2. Chaumette F, Hutchinson S (2007) Visual servo control, part II: advanced approaches. *IEEE Robot Autom Mag* 14(1):109–118

3. Chesi G, Hashimoto K (eds) (2010) Visual servoing via advanced numerical methods. LNCIS, vol 401. Springer, Berlin
4. Corke P (1997) Visual control of robots: high-performance visual servoing. Wiley, New York
5. Corke P (2011) Robotics, vision and control. In: Springer tracts in advanced robotics, vol 73. Springer, Berlin
6. Espiau B, Chaumette F, Rives P (1992) A new approach to visual servoing in robotics. IEEE Trans Robot Autom 8(3):313–326
7. Hashimoto K (ed) (1993) Visual servoing: real-time control of robot manipulators based on visual sensory feedback. World Scientific Publishing, Singapore
8. Hutchinson S, Hager G, Corke P (1996) A tutorial on visual servo control. IEEE Trans Robot Autom 12(5): 651–670
9. Kriegman D, Hager G, Morse S (eds) (1998) The confluence of vision and control. LNCIS, vol 237. Springer, London
10. Malis E, Chaumette F, Bouabdellah S (1999) 2-1/2D visual servoing. IEEE Trans Robot Autom 15(2): 238–250
11. Weiss L, Sanderson A, Neuman C (1987) Dynamic sensor-based control of robots with visual feedback. IEEE J Robot Autom 3(5):404–417
12. Wilson W, Hulls C, Bell G (1996) Relative end-effector control using cartesian position-based visual servoing. IEEE Trans Robot Autom 12(5):684–696

Visual SLAM

- Exploration: Simultaneous Localization and Mapping (SLAM)

Volumetric Texture

- Bidirectional Texture Function and 3D Texture

von Kries Hypothesis

Rajeev Ramanath¹ and Mark S. Drew²

¹DLP® Products, Texas Instruments Incorporated, Plano, TX, USA

²School of Computing Science, Simon Fraser University, Vancouver, BC, Canada

Synonyms

Adaptive gains; Coefficient rule; Color adaptation; Ives transform; von Kries-Ives adaptation

Related Concepts

- White Balance

Definition

The von Kries hypothesis as applied to chromatic adaptation is an approach that is the basis of most modern color adaptation models. It was first hypothesized by Johannes von Kries in 1902 [3]. The approach requires one to apply a gain to each of the cone responses, independently, so as to keep the adapted appearance of a reference white constant.

Theory

In general terms, let us denote the three cone responses in the human retina as L , M , S , and let the spectral sensitivities of the cones be denoted by $\bar{l}(\lambda)$, $\bar{m}(\lambda)$, and $\bar{s}(\lambda)$. Then for any given stimulus $i_r(\lambda)$ that is incident on the retina, the cone responses are assumed to be given by:

$$\begin{aligned} L &= k \int_{\lambda} \bar{l}(\lambda) i_r(\lambda) d\lambda \\ M &= k \int_{\lambda} \bar{m}(\lambda) i_r(\lambda) d\lambda \\ S &= k \int_{\lambda} \bar{s}(\lambda) i_r(\lambda) d\lambda \end{aligned} \quad (1)$$

with k a normalizing constant, and $\bar{l}(\lambda)$, $\bar{m}(\lambda)$, $\bar{s}(\lambda)$ are the spectral sensitivity functions of the three types of color receptors in the eye. In the above equations, $i_r(\lambda)$ is given by:

$$i_r(\lambda) = i(\lambda)r(\lambda) \quad (2)$$

where $i(\lambda)$ denotes the illuminant and $r(\lambda)$ denotes the spectral reflectivity of a given surface.

The von Kries model is used to predict the response of the cones under changes in the viewing conditions. Specifically, an important change in the viewing conditions is a change in the illuminant. Let us assume that the illuminant $i(\lambda)$ is replaced by a different illuminant, $i_a(\lambda)$. In such a case, let us denote the



cone responses by L_a , M_a , and S_a . Then von Kries hypothesis may be expressed as:

$$\begin{bmatrix} L_a \\ M_a \\ S_a \end{bmatrix} = \begin{bmatrix} k_L & 0 & 0 \\ 0 & k_M & 0 \\ 0 & 0 & k_S \end{bmatrix} \begin{bmatrix} L \\ M \\ S \end{bmatrix} \quad (3)$$

where k_L , k_M , and k_S denote scale factors that are applied independently to the three cone responses. It is perhaps most common to see this above equation represented with $k_L = 1/L_{\max}$, $k_M = 1/M_{\max}$, and $k_S = 1/S_{\max}$ where the max subscript denotes the maximal responses for the LMS cone functions under a normative white stimulus.

In terms of representing the adaptation between the original and new illuminants, in the von Kries model the adapted stimuli are assumed to be given by:

$$\begin{bmatrix} L_a \\ M_a \\ S_a \end{bmatrix} = \begin{bmatrix} \frac{L_{\max a}}{L_{\max}} & 0 & 0 \\ 0 & \frac{M_{\max a}}{M_{\max}} & 0 \\ 0 & 0 & \frac{S_{\max a}}{S_{\max}} \end{bmatrix} \begin{bmatrix} L \\ M \\ S \end{bmatrix}. \quad (4)$$

The von Kries approach, although described in the space of LMS cone functions, can be easily extended to be used in the space of XYZ tristimulus values, defined in terms of colorimetry, rather than in the psychophysically motivated LMS space. And indeed this is typically done, by using a simple 3×3 transformation between the LMS cone fundamentals and XYZ color-matching functions:

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \mathbf{M} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \quad (5)$$

The exact values used for matrix \mathbf{M} depend on the specific cone fundamentals and XYZ color-matching functions chosen [1, 4].

Open Problems

This approach, although not truly accurate with regard to experimental findings in color adaptation, is surprisingly general in its application [2]. Many researchers have been exploring the limitations of this hypothesis and adapting it for use in modern color adaptation models. In Chap. 9 of his book Fairchild provides a comprehensive comparison of the von Kries model to several other color adaptation models developed in the past century [1].

References

1. Fairchild MD (2005) Color appearance models. The Wiley-IS&T series in imaging science and technology, 2nd edn. Wiley, Chichester
2. Hunt RWG (2004) Reproduction of colour, 6th edn. Wiley, Chichester
3. von Kries J (1970) Chromatic adaptation. Festschrift der Albrecht-Ludwig-Universität, 1902. (trans: MacAdam DL, Sources of color science) MIT, Cambridge
4. Wyszecki G, Stiles WS (1982) Color science: concepts and methods, quantitative data and formulas, 2nd edn. Wiley, New York

von Kries-Ives Adaptation

► von Kries Hypothesis



W

Weak Perspective Projection

Zhengyou Zhang
Microsoft Research, Redmond, WA, USA

Related Concepts

- ▶ [Affine Camera](#); ▶ [Calibration of Projective Cameras](#);
- ▶ [Camera Calibration](#); ▶ [Camera Parameters \(Intrinsic, Extrinsic\)](#);
- ▶ [Depth Distortion](#); ▶ [Perspective Camera](#);
- ▶ [Perspective Transformation](#)

Definition

Weak perspective projection is a linear approximation of the (full) *perspective projection*. In this entry, we also describe other forms of linear approximation: *orthographic projection* and *paraperspective projection*. Finally, the *affine camera* is presented as the most general form of linear approximation of the perspective projection.

Background

As can be seen in the entry ▶ [perspective camera](#), the perspective projection is a nonlinear mapping. This makes many vision problems difficult to solve, and more importantly, they can become ill-conditioned when the perspective effects are small. Sometimes, if certain conditions are satisfied, for example, when the camera field of view is small and the object size is

small enough with respect to the distance from the viewer to the object, the projection can be approximated by a linear mapping [1]. This simplification also leads to great reduction of complexity in computation of the epipolar geometry from images [2].

Theory

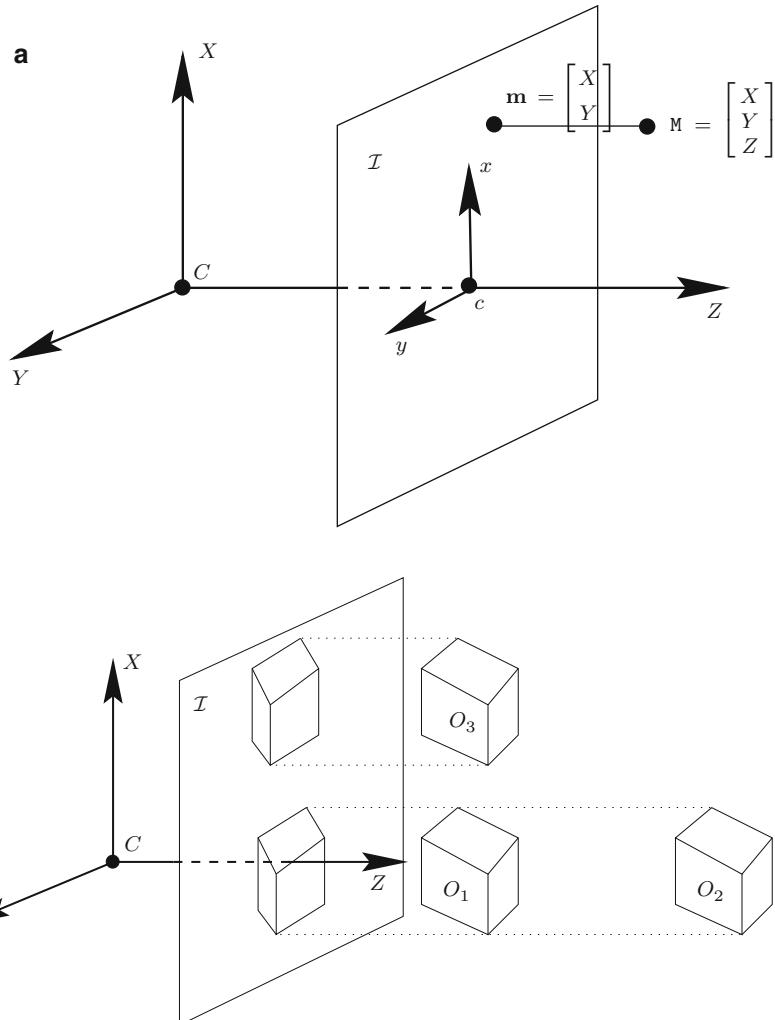
Orthographic Projection The simplest approximation is the so-called *orthographic projection* (Fig. 1a). It ignores completely the depth dimension. There is some evidence that rays near the fovea are projected orthographically (see, for example, [3]). Because of this, orthographic projection has been used in human vision research [4]. But the orthographic projection can lead to two identical objects having the same image, even if one is much further away from the camera than the other (*distance effect*) or if one is much more distant from the optical axis than the other (*position effect*) (see Fig. 1b). Therefore, methods that use orthographic projection are only valid in a limited domain where the distance and position effects can be ignored.

Weak Perspective Projections A much more reasonable approximation is the so-called *weak perspective projection*. When the object size is small enough with respect to the distance from the camera to the object, the depth, or Z , can be replaced by a common depth, Z_c . Then the projection equations become linear:

Weak Perspective

Projection, Fig. 1 (a)

Orthographic projection. (b)
 Three identical objects in different positions give the same image under orthographic projection:
 Object O_2 is much further away from the camera, and object O_3 is much distant from the optical axis, than object O_1



$$\begin{aligned} x &= \frac{X}{Z_c} \\ y &= \frac{Y}{Z_c} \end{aligned} \quad (1)$$

Here, we assume that the focal length f is normalized to 1.

Suppose the common depth is the depth of the centroid of the object. Then this approximation can be understood as a two-step projection (Fig. 2). The first one is a parallel projection of all the object surface points onto a plane which goes through Z_c , the depth of the object centroid, and is parallel to the image plane or focal plane. The second step is a perspective projection of that plane onto the image plane, which is actually a uniform scaling of that plane. Sometimes,

this projection is also called the *scaled orthographic projection*. When Z_c is unity, the projection becomes the *orthographic* one.

Let

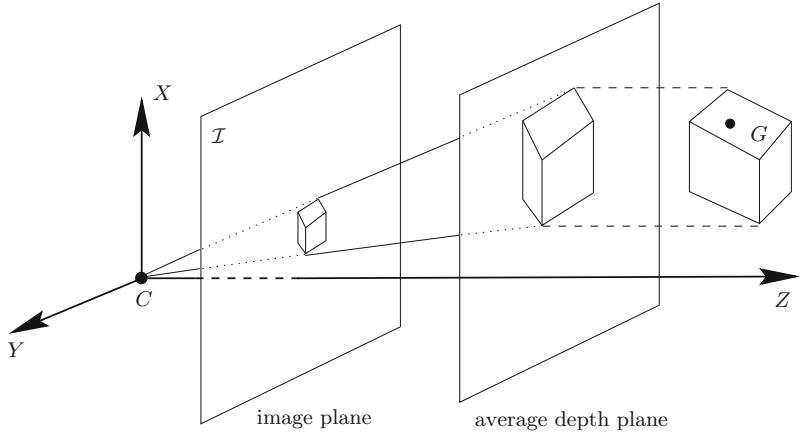
$$\mathbf{P}_{wp} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & Z_c \end{bmatrix}. \quad (2)$$

Equation (1) can then be written in matrix form, similar to the perspective projection matrix, as

$$s \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{P}_{wp} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}.$$

Weak Perspective

Projection, Fig. 2 The weak perspective model. The average depth plane is drawn in front for clarity



Taking into account the intrinsic and extrinsic parameters of the camera yields

$$s\tilde{\mathbf{m}} = \mathbf{AP}_{wp}\mathbf{D}\tilde{\mathbf{M}}, \quad (3)$$

where \mathbf{A} is the intrinsic matrix and \mathbf{D} is the rigid transformation. Expand it and eliminate the scale factor s , and we have

$$\mathbf{m} = \mathbf{T}_{wp}\mathbf{M} + \mathbf{t}_{wp}, \quad (4)$$

where

$$\begin{aligned} \mathbf{T}_{wp} &= \frac{1}{Z_c} \begin{bmatrix} \alpha_u(\mathbf{r}_1^T + \cot\theta\mathbf{r}_2^T) \\ \alpha_v\mathbf{r}_2^T / \sin\theta \end{bmatrix} \\ \mathbf{t}_{wp} &= \frac{1}{Z_c} \begin{bmatrix} \alpha_u(t_1 + t_2 \cot\theta) \\ \alpha_v t_2 / \sin\theta \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \end{bmatrix}. \end{aligned}$$

Here \mathbf{r}_i denotes the i th row vector of the rotation matrix \mathbf{R} , and t_i is the i th element of the translation vector \mathbf{t} . It is clear that the relation between 3D coordinates and image coordinates is linear.

Let us examine the approximation error introduced by the weak perspective projection. For simplicity, we consider a normalized camera with the camera and world coordinate systems aligned. For a point \mathbf{M} with depth $Z = Z_c + \Delta Z$, its perspective projection can be developed as a Taylor series about Z_c :

$$\begin{aligned} \mathbf{m}_p &= \frac{1}{Z_c + \Delta Z} \begin{bmatrix} X \\ Y \end{bmatrix} \\ &= \frac{1}{Z_c} \left(1 - \frac{\Delta Z}{Z_c} + \left(\frac{\Delta Z}{Z_c} \right)^2 - \dots \right) \begin{bmatrix} X \\ Y \end{bmatrix}. \end{aligned}$$

When $|\Delta Z| \ll Z_c$, only the zero-order term remains, giving Eq. (1). Weak perspective is thus the zero-order approximation of the full perspective projection. The absolute error in image position is then

$$\mathbf{m}_{error} = \mathbf{m}_p - \mathbf{m}_{wp} = -\frac{1}{Z} \frac{\Delta Z}{Z_c} \begin{bmatrix} X \\ Y \end{bmatrix},$$

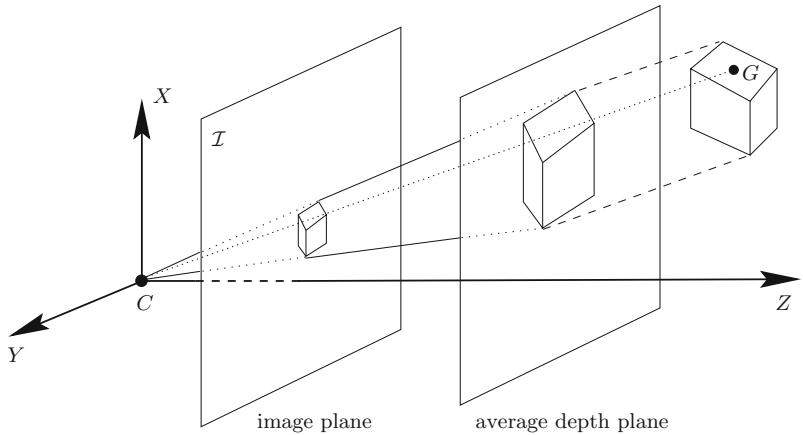
which shows that a small field of view (X/Z and Y/Z) and small depth variation ($\Delta Z/Z_c$) contribute to the validity of the model [5]. The error is clearly not uniform across the image. A useful rule of thumb requires Z_c to exceed $|\Delta Z|$ by an order of magnitude [6], i.e., $Z_c \geq 10|\Delta Z|$.

Paraperspective Projection Under weak perspective projection, a world point is first projected onto the average depth plane using the rays parallel to the optical axis. This causes a significant approximation error when an object is distant to the optical axis (large X/Z and Y/Z). The *paraperspective projection* generalizes this by projecting points using the rays parallel to the central projecting ray CG , where G is the centroid of the object [1]. As with weak perspective, the perspective projection is also approximated with a two-step process, but the projection in the first step is no more realized along rays parallel to the optical axis (see Fig. 3).

Let the coordinates of the centroid of the object G be $[X_c, Y_c, Z_c]^T$. A world point $\mathbf{M} = [X, Y, Z]^T$ is first projected, parallel to CG , to the average depth plane at $[X - \frac{X_c}{Z_c}Z + X_c, Y - \frac{Y_c}{Z_c}Z + Y_c, Z_c]^T$. Finally, this point is projected perspectively onto the image plane as

Weak Perspective Projection, Fig. 3

Paraperspective projection.
The average depth plane is drawn in front for clarity



$$\begin{aligned}x &= \frac{1}{Z_c} \left(X - \frac{X_c}{Z_c} Z + X_c \right) \\y &= \frac{1}{Z_c} \left(Y - \frac{Y_c}{Z_c} Z + Y_c \right).\end{aligned}$$

$$(5) \quad \begin{aligned}\mathbf{t}_{pp} &= \frac{1}{Z_c} \begin{bmatrix} \alpha_u \left(t_1 - \frac{X_c}{Z_c} t_3 + X_c \right) + \alpha_u \cot \theta \left(t_2 - \frac{Y_c}{Z_c} t_3 + Y_c \right) \\ \alpha_v \left(t_2 - \frac{Y_c}{Z_c} t_3 + Y_c \right) / \sin \theta \\ u_0 \\ v_0 \end{bmatrix} \\ &+ \begin{bmatrix} u_0 \\ v_0 \end{bmatrix}.\end{aligned}$$

They can be written in matrix form as

$$s \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{P}_{pp} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix},$$

with

$$\mathbf{P}_{pp} = \begin{bmatrix} 1 & 0 & -X_c/Z_c & X_c \\ 0 & 1 & -Y_c/Z_c & Y_c \\ 0 & 0 & 0 & Z_c \end{bmatrix}. \quad (6)$$

Taking into account the intrinsic and extrinsic parameters of the camera yields

$$s \tilde{\mathbf{m}} = \mathbf{A} \mathbf{P}_{pp} \mathbf{D} \tilde{\mathbf{M}}. \quad (7)$$

Here, of course, X_c , Y_c , and Z_c in \mathbf{P}_{pp} must be still measured in the camera coordinate system. In terms of image and world coordinates, we have

$$\mathbf{m} = \mathbf{T}_{pp} \mathbf{M} + \mathbf{t}_{pp}, \quad (8)$$

where

$$\mathbf{T}_{pp} = \frac{1}{Z_c} \begin{bmatrix} \alpha_u \left(\mathbf{r}_1^T - \frac{X_c}{Z_c} \mathbf{r}_3^T \right) + \alpha_u \cot \theta \left(\mathbf{r}_2^T - \frac{Y_c}{Z_c} \mathbf{r}_3^T \right) \\ \alpha_v \left(\mathbf{r}_2^T - \frac{Y_c}{Z_c} \mathbf{r}_3^T \right) / \sin \theta \end{bmatrix}$$

It is clear that the relation between 3D coordinates and image coordinates is linear under paraperspective projection.

Let us examine the approximation error introduced by the weak perspective projection. For simplicity, we consider a normalized camera with the camera and world coordinate systems aligned. For a point \mathbf{M} with coordinates $[X, Y, Z]^T = [X_c + \Delta X, Y_c + \Delta Y, Z_c + \Delta Z]^T$, its perspective projection can be developed as a Taylor series about Z_c :

$$\begin{aligned}\mathbf{m}_p &= \frac{1}{Z_c + \Delta Z} \begin{bmatrix} X_c + \Delta X \\ Y_c + \Delta Y \end{bmatrix} \\ &= \frac{1}{Z_c} \left(1 - \frac{\Delta Z}{Z_c} + \left(\frac{\Delta Z}{Z_c} \right)^2 - \mathcal{O}^3 \right) \begin{bmatrix} X_c + \Delta X \\ Y_c + \Delta Y \end{bmatrix},\end{aligned}$$

where \mathcal{O}^3 is used to denote terms of order higher than 2, i.e., $\mathcal{O}^3 = O \left(\frac{\Delta Z}{Z_c} \right)^3$. The projection under paraperspective, \mathbf{m}_{pp} , is given by Eq. (5). The absolute error in image plane, after some algebra, is given by

$$\mathbf{m}_{error} = \mathbf{m}_p - \mathbf{m}_{pp} = \begin{bmatrix} -\frac{\Delta X \Delta Z}{Z_c^2} + \frac{X}{Z_c} \left(\frac{\Delta Z}{Z_c} \right)^2 - \mathcal{O}^3 \\ -\frac{\Delta Y \Delta Z}{Z_c^2} + \frac{Y}{Z_c} \left(\frac{\Delta Z}{Z_c} \right)^2 - \mathcal{O}^3 \end{bmatrix}.$$

As can be seen, the approximation error is of second order under paraperspective projection, and we say that it is the first-order approximation of the full perspective projection. The weak perspective projection, as was seen in the last subsection, is only the zero-order approximation. Figure 4 illustrates well the difference between all these projection models. Similar analysis is presented in [7] for pose determination.

An even better approximation is called *orthoperspective*. The difference from the paraperspective projection consists in replacing the average depth plane, which is parallel to the image plane, by another plane that is perpendicular to the central projecting ray (which connects the optical center C of the camera and the centroid G of the object (see Fig. 5 for a pictorial description). It can be understood as the result of the following sequence of operations [1]:

- A virtual rotation of the camera around C to make the central projecting ray CG coincide with the optical axis
- A weak perspective projection of the object
- An inverse camera rotation to bring the camera back to its original position

The orthoperspective projection involves more complicated formula and will not be discussed anymore in this book.

Affine Cameras

If we examine the camera projection matrices for orthographic, weak perspective, and paraperspective projections (see Eq. (2) and (6)), we find that they all have the same form:

$$\mathbf{P}_A = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ 0 & 0 & 0 & P_{34} \end{bmatrix}. \quad (9)$$

Depending on different projection models, some constraints exist on the elements of matrix \mathbf{P}_A except P_{31} , P_{32} , and P_{33} , which are equal to 0. If we ignore the constraints on the matrix elements, \mathbf{P}_A becomes the so-called *affine camera*, introduced by Mundy and Zisserman [8]. Unlike the general perspective projection matrix (see entry ►I), an affine camera, which is also defined up to a scale factor, has only eight degrees of freedom. It can be determined from four points in general position.

In terms of image and world coordinates, the affine camera is written as

$$\mathbf{m} = \mathbf{T}_A \mathbf{M} + \mathbf{t}_A, \quad (10)$$

where \mathbf{T}_A is a 2×3 matrix with elements $T_{ij} = P_{ij}/P_{34}$ and \mathbf{t}_A is a 2D vector $[P_{14}/P_{34}, P_{24}/P_{34}]^T$.

A key property of the affine camera is that it preserves parallelism [5]: lines that are parallel in the world remain parallel in the image, which is not the case with the perspective camera. The proof is simple. Consider two parametric parallel lines $\mathbf{M}_1(\lambda) = \mathbf{M}_a + \lambda \mathbf{u}$ and $\mathbf{M}_2(\mu) = \mathbf{M}_b + \mu \mathbf{u}$, where \mathbf{u} is the 3D direction vector. They project onto the image as lines

$$\begin{aligned} \mathbf{m}_1(\lambda) &= (\mathbf{T}_A \mathbf{M}_a + \mathbf{t}_A) + \lambda \mathbf{T}_A \mathbf{u}, \\ \mathbf{m}_2(\mu) &= (\mathbf{T}_A \mathbf{M}_b + \mathbf{t}_A) + \mu \mathbf{T}_A \mathbf{u}. \end{aligned}$$

They are clearly parallel to the 2D direction vector $\mathbf{T}_A \mathbf{u}$.

Another attractive property of the affine camera is that the centroid of a set of 3D points and that of their image points correspond to each other. This is not the case for full perspective projections. Let \mathbf{M}_i ($i = 1, \dots, n$) be the set of 3D points and \mathbf{m}_i , their corresponding image points. From Eq. (10), the centroid of the image points is given by

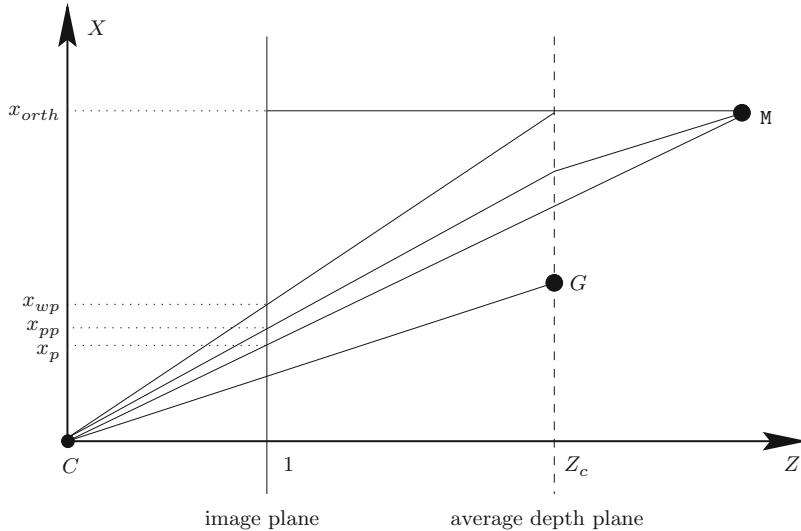
$$\bar{\mathbf{m}} = \frac{1}{n} \sum_{i=1}^n \mathbf{m}_i = \frac{1}{n} \sum_{i=1}^n (\mathbf{T}_A \mathbf{M}_i + \mathbf{t}_A) = \mathbf{T}_A \bar{\mathbf{M}} + \mathbf{t}_A,$$

where $\bar{\mathbf{M}}$ is the centroid of the 3D points. Thus, the 3D point centroid is projected by an affine camera to the image point centroid. It follows that if 3D points are expressed with respect to their centroid, i.e. $\hat{\mathbf{M}}_i = \mathbf{M}_i - \bar{\mathbf{M}}$, and if 2D points are also expressed with respect to their centroid, i.e. $\hat{\mathbf{m}}_i = \mathbf{m}_i - \bar{\mathbf{m}}$, then we have

$$\hat{\mathbf{m}}_i = \mathbf{T}_A \hat{\mathbf{M}}_i.$$

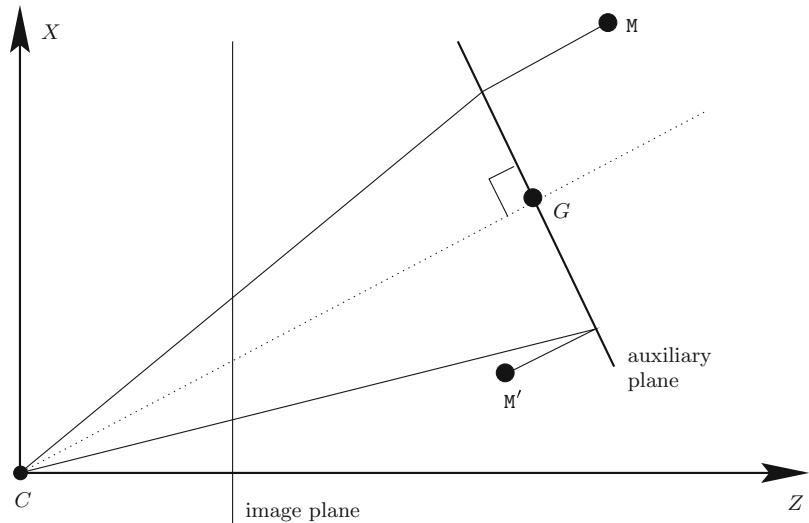
Therefore, an affine camera has only six degrees of freedom (the six parameters of \mathbf{T}_A).

While the exact projection performed by an affine camera is not very clear, it is clearly the generalization of the orthographic, weak perspective, or paraperspective models, as can be seen from Eq. (9) or (10). The generalization can be understood in two ways:



Weak Perspective Projection, Fig. 4 Comparison of different perspective approximations. Cross-sectional view sliced by a plane that includes the central projecting ray and is perpendicular to the Y - Z plane. The image plane is the line $Z = 1$. For x_p (full perspective), projection is along the ray connecting the world point M to the optical center C . For x_{orth} (orthographic),

projection is perpendicular to the image. For x_{wp} (weak perspective), M is first orthographically projected onto the average depth plane and then projected perspectively onto the image plane. For x_{pp} (paraperspective), M is first projected, parallel to the central projecting ray, onto the average depth plane and then projected perspectively onto the image plane (Adapted from [5]).



Weak Perspective Projection, Fig. 5

Orthoperspective projection. A cross-sectional view sliced by a plane that includes the central projecting ray and is perpendicular to the Y - Z plane

- Some *nonrigid* deformation of the object is permitted. Actually, if we multiply \mathbf{P}_A from the right by any 3D affine transformation of the object

$$\mathbf{D} = \begin{bmatrix} \mathbf{M} & \mathbf{t} \\ \mathbf{0}_3^T & 1 \end{bmatrix},$$

where \mathbf{M} is a general 3×3 matrix and \mathbf{t} is a 3-vector, the new projection matrix still has the form Eq. (9) of an affine camera.

- Camera calibration is unnecessary. Multiply \mathbf{P}_A from the left by any 2D affine transformation of the image

$$\mathbf{A} = \begin{bmatrix} \mathbf{C} & \mathbf{c} \\ \mathbf{0}_2^T & 1 \end{bmatrix},$$

where \mathbf{C} is a general 2×2 matrix and \mathbf{c} is a 2-vector; the new projection matrix still has the form Eq. (9) of an affine camera.

Without camera calibration, various affine measurements, such as parallelism and ratios of lengths in parallel directions, can be extracted. For certain vision tasks, such properties may be sufficient. Affine cameras are extensively studies in particular by the University of Oxford group [5, 9, 10].

References

1. Aloimonos J (1990) Perspective approximations. *Image Vis Comput* 8(3):179–192
2. Xu G, Zhang Z (1996) Epipolar geometry in stereo, motion and object recognition. Kluwer Academic, Dordrecht/Boston
3. Horn BKP (1986) Robot vision. MIT, Cambridge
4. Ullman S (1979) The interpretation of visual motion. MIT, Cambridge
5. Shapiro L (1993) Affine analysis of image sequences. PhD thesis, University of Oxford, Department of Engineering Science, Oxford, UK
6. Thompson D, Mundy J (1987) Three-dimensional model matching from an unconstrained viewpoint. In: Proceedings of the international conference on robotics and automation, Raleigh, pp 208–220
7. Horaud R, Christy S, Dornaika F (1994) Object pose: the link between weak perspective, para perspective, and full perspective. Technical report 2356, INRIA
8. Mundy JL, Zisserman A (eds) (1992) Geometric invariance in computer vision. MIT, Cambridge
9. Demey S, Zisserman A, Beardsley P (1992) Affine and projective structure from motion. In: British machine vision conference, Leeds, UK, pp 49–58
10. Hartley RI, Zisserman A (2004) Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, ISBN: 0521540518

Weakly Supervised Learning

Lorenzo Torresani

Computer Science Department, Dartmouth College,
Hanover, NH, USA

Synonyms

[Learning from partial labels](#)

Definition

Weakly supervised learning is a machine learning framework where the model is trained using examples that are only partially annotated or labeled.

Background

Most modern computer vision system involves models learned from human-labeled image examples. For instance, an object detector is typically trained on a large collection of images manually annotated with masks or bounding boxes denoting the location of the object of interest in each photo. The reliance on time-consuming human labeling poses a significant limitation to the practical application of these methods. Weakly supervised learning is aimed at reducing the amount of human intervention needed to train the models by making use of examples that are only partially labeled.

Theory

There are two main forms of weakly supervised learning, differing with respect to the type of partial labels used to annotate the examples:

1. *Semisupervised learning* involves training a model using a mixture of both fully labeled examples as well as unlabeled examples [16].
2. *Multiple-instance learning* uses training examples grouped into sets, commonly referred to as *bags* [5]. A single label (either positive or negative) is assigned to each bag. A positive label indicates that the bag includes at least one example belonging to the positive class. A negative label denotes that all instances of the bag belong to the negative class.

Application

In computer vision, weakly supervised learning has been predominantly used for classification tasks such as image categorization and object detection. In these settings, simple incremental approaches (e.g., self-training [14]) have been used effectively to improve the classification performance of fully supervised models

using unlabeled images: at a high level, the classifiers are first learned using the labeled training data; then the resulting classifiers are used on the unlabeled data to convert the most confident predictions into new labeled examples to be used to retrain the models; this process iterates until all the unlabeled examples are given labels. Rosenberg et al. [15] apply this scheme to train object detectors with a small number of fully labeled examples and an additional set of unlabeled examples. A related approach is co-training [3], where different classifiers are trained simultaneously on disjoint subsets of features using labeled examples. Unlabeled examples which are confidently labeled by one classifier are added, with labels, to the training set of the other classifiers. Levin et al. [11] demonstrate the use of co-training on the task of car detection. Incremental schemes of semisupervised learning have also been used to learn object categories using text-based Internet searches as sources of weakly labeled images [7]. The Internet is a natural source of semisupervised image data, where reliable labels are manually provided only for a small fraction of photos; noisy labels may be extracted automatically from surrounding text, while for many images there are no labels at all. In [8], the authors introduce an efficient semisupervised learning method that is linear in the number of image examples and thus designed to work on such large-scale collections of photos.

In object detection, the strategy of multiple-instance learning has been successfully employed to reduce or in some case even eliminate the need to provide manually labeled regions in training images. For example, in [13] and [4], methods are presented to learn object detectors from examples annotated with class labels indicating the type of object present, but not its location: each image is viewed as a bag of instances corresponding to all possible rectangles inside the image, and during training, the positive instance is automatically selected by identifying appearance patterns that occur frequently in bags of that class. In [6], objects are represented using mixtures of multiscale deformable part models that can be trained using a multiple-instance discriminative procedure requiring only rough bounding boxes as training input.

Weakly supervised learning has also been applied extensively for tracking objects in videos, most typically to specialize a tracker to a new sequence via semisupervised learning [9] or multiple-instance learning [1]: labeled data (or a previously trained model) is used as a prior, while the frames collected during

tracking are viewed as a collection of unlabeled or weakly labeled samples to perform the tracker update. In [12], unlabeled images are employed to improve 3D human pose estimation via semisupervised regression.

Image segmentation is a task that is inherently semisupervised [2]: given a collection of labeled pixels in an image (i.e., some segmented region, usually user defined), a semisupervised learning method is employed to classify the remaining unlabeled pixels (i.e., segment unknown regions in the image). Similarly, in Levin et al. [10], semisupervised learning is applied to perform colorization of gray-scale images, where the user provides the labeled data by specifying the desired color with only a few strokes on the image.

References

- Babenko B, Ming-Hsuan Yang, Belongie S (2009) Visual tracking with online multiple instance learning. In: IEEE conference on computer vision and pattern recognition (CVPR), Miami, pp 983–990
- Blake A, Rother C, Brown M, Pérez P, Torr PHS (2004) Interactive image segmentation using an adaptive gmmrf model. In: 8th European conference on computer vision (ECCV), Prague, pp 428–441
- Blum A, Mitchell TM (1998) Combining labeled and unlabeled data with co-training. In: Computational learning theory, Madison, Wisconsin, USA, pp 92–100
- Deselaers T, Alexe B, Ferrari V (2012) Weakly supervised localization and learning with generic knowledge. Int J Comp Vis 1–19
- Dietterich TG, Lathrop RH, Lozano-Pérez T (1997) Solving the multiple instance problem with axis-parallel rectangles. Artif Intell 89(1–2):31–71
- Felzenszwalb PF, Girshick RB, McAllester DA, Ramanan D (2010) Object detection with discriminatively trained part-based models. IEEE Trans Pattern Anal Mach Intell 32(9):1627–1645
- Fergus R, Perona P, Zisserman A (2004) A visual category filter for google images. In: ECCV 2004, 8th European conference on computer vision, Prague, pp 242–256
- Fergus R, Weiss Y, Torralba A (2009) Semi-supervised learning in gigantic image collections. In: Advances in neural information processing systems 22, Vancouver, pp 522–530
- Grabner H, Leistner C, Bischof H (2008) Semi-supervised on-line boosting for robust tracking. In: Forsyth D, Torr P, Zisserman A (eds) Computer vision (ECCV 2008). Lecture notes in computer science, vol 5302. Springer, Berlin/Heidelberg, pp 234–247
- Levin A, Lischinski D, Weiss Y (2004) Colorization using optimization. ACM Trans Graph 23(3):689–694
- Levin A, Viola PA, Freund Y (2003) Unsupervised improvement of visual detectors using co-training. In: 9th IEEE

- international conference on computer vision (ICCV), Nice, pp 626–633
12. Navaratnam R, Fitzgibbon AW, Cipolla R (2007) The joint manifold model for semi-supervised multi-valued regression. In: IEEE 11th international conference on computer vision (ICCV), Rio de Janeiro, pp 1–8
 13. Nguyen MH, Torresani L, Lorenzo de la Torre, Rother C (2009) Weakly supervised discriminative localization and classification: a joint learning process. In: IEEE 12th international conference on computer vision (ICCV), Kyoto, pp 1925–1932
 14. Nigam K, Ghani R (2000) Analyzing the effectiveness and applicability of co-training. In: ACM CIKM international conference on information and knowledge management, McLean, VA, USA, pp 86–93
 15. Rosenberg C, Hebert M, Schneiderman H (2005) Semi-supervised self-training of object detection models. In: 7th IEEE workshop on applications of computer vision/IEEE workshop on motion and video computing, Breckenridge, CO, USA, pp 29–36
 16. Zhu X, Goldberg AB (2009) Introduction to semi-supervised learning. Synthesis lectures on artificial intelligence and machine learning. Morgan & Claypool, San Rafael/California

White Balance

Rajeev Ramanath¹ and Mark S. Drew²

¹DLP® Products, Texas Instruments Incorporated, Plano, TX, USA

²School of Computing Science, Simon Fraser University, Vancouver, BC, Canada

Synonyms

Automatic white balance (AWB); Color balance

Related Concepts

►Chromaticity; ►Color Constancy; ►von Kries Hypothesis

Definition

White balance is the term given to the process of removing the color cast in a captured image that is induced by the color of the illuminant. This is typically done by changing the overall proportion of colors in an image to provide an overall balance by keeping neutral colors neutral in their reproduction.

Background

The human visual system has the capability of mapping “white” colors to the sensation of “white” even when lighting conditions change in color. This is easily illustrated by looking at a white sheet of paper under indoor incandescent lighting and outdoor sunlight. In both cases the paper would appear white to our visual system, but appear yellowish and bluish in color, respectively, to a digital color camera. This psychophysical phenomenon is referred to as *color constancy*.

Similarly, a yellowish-white light (such as an incandescent bulb) reflected from a white sheet of paper would be perceived as white to our visual system while daylight reflected from a yellowish-white flower appears yellowish, even though the spectral information received by our eyes is identical. This is a significant challenge to a camera – digital or film-based – as the camera does not “know” the color of an object: this typically results in a color cast in the captured image, with the overall color appearing wrong to a human observer. Stated differently, neutral colored objects appear neutral to the human visual system and a camera is expected to be aware of this color constancy. Under flash photography, however, the situation is somewhat simpler as we know the illuminant in part.

The process of removing such a color cast so that “white remains white” under the capture and viewing illuminants is termed *white balancing*. Ideally, of course, consumers expect this to be performed automatically by their cameras.

Theory

The primary cause of the need for white balance comes from the fact that the camera is unable to accurately mimic the human visual system. For the sake of simplicity let us consider a digital still camera with three color sensors and compare how the data flows in a camera, compared to (an abstraction of) how it might flow in the human visual system. A particular camera’s spectral sensitivities are shown in Fig. 1b while those of the cones in the human retina are shown in Fig. 1a [7]. When a reflectance spectrum of, say, a white color patch is captured by these sensors, under a D65 white illuminant (shown in Fig. 1c), the initial values captured by the eye and the camera are clearly different. But more importantly, the processing that

takes place on these two signals is radically different, resulting in a completely different sensation of white in the two “systems.” A generic overview of the structure of a digital still camera’s image processing pipeline may be found in a paper by Ramanath et al. [6].

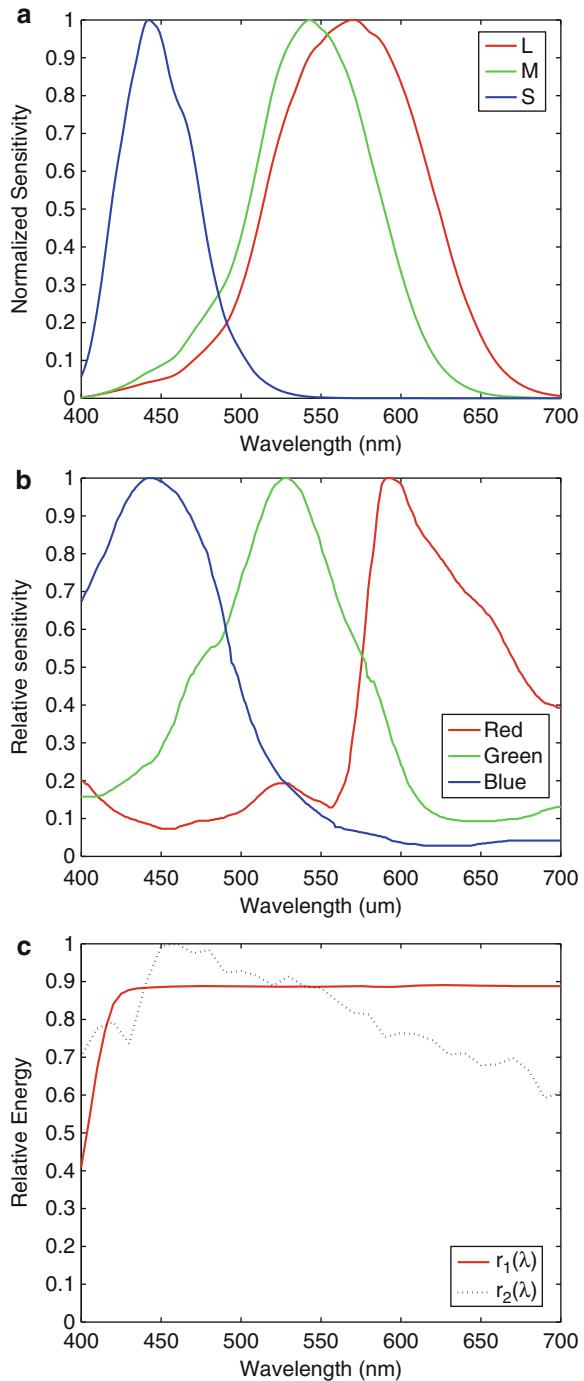
The challenge for the camera’s system is to mimic the performance of the human visual system with little or no knowledge of the illuminant under which the image was captured.

One means of performing white balance is to assume that a white patch must induce maximal response in one or more of the camera’s sensor in the three (or more) channels. If R , G , and B denote the red, green, and blue channels of the captured image, the white-balanced image has signals given by R/R_{\max} , G/G_{\max} , and B/B_{\max} . This type of white balance may be represented as:

$$\begin{bmatrix} R_{\text{out}} \\ G_{\text{out}} \\ B_{\text{out}} \end{bmatrix} = \begin{bmatrix} 1/R_{\max} & 0 & 0 \\ 0 & 1/G_{\max} & 0 \\ 0 & 0 & 1/B_{\max} \end{bmatrix} \begin{bmatrix} R_{\text{in}} \\ G_{\text{in}} \\ B_{\text{in}} \end{bmatrix} \quad (1)$$

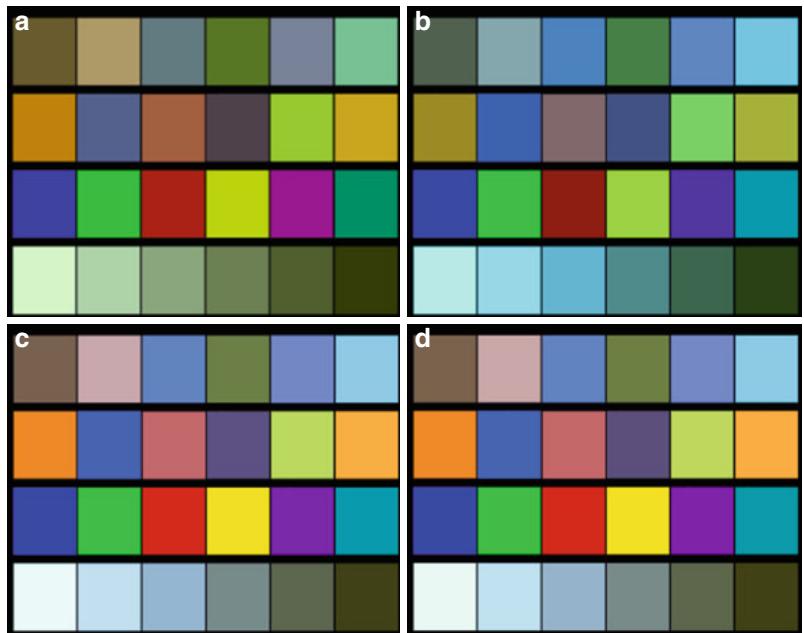
where the subscript “in” denotes the input pixels and subscript “out” denotes the output white-balanced pixel values. However, a simplistic maximum in the three channels (“max RGB” approach) is typically a poor estimate of the illuminant as it tends to take the maximum of each color from different pixels and the scaling typically results in contrast enhancement or a dynamic range expansion. A slightly more intelligent approach takes the maximum of the color that has the largest magnitude in the RGB color space (“brightest pixel” approach). This is also potentially error prone in images with no captured whites (or neutral colors). See Fig. 2 for examples of approaches that attempt to mimic the image captured under a daylight illuminant using these simple white balance approaches.

The next level of complication as an approach to color constancy (although an admittedly naïve one) is the “gray world” assumption, which assumes that all colors in an image will average out to gray, $R = G = B$. In other words, the channels are scaled based upon the deviation of the average image color from gray. In this approach, the white-balanced image has signals given by $k_r R$, G , and $k_b B$, where $k_r = G_{\text{mean}}/R_{\text{mean}}$ and $k_b = G_{\text{mean}}/B_{\text{mean}}$, the premise being



White Balance, Fig. 1 The human visual system and a camera’s capture system compared: (a) LMS cone functions of a human observer; (b) spectral sensitivities of filters used in a particular digital color camera; (c) spectral reflectance function of a white patch, $r_1(\lambda)$, and the spectral emission, $r_2(\lambda)$, for the standard D65 daylight illuminant

White Balance, Fig. 2 (a) Image captured under a daylight illuminant; (b) image captured in fluorescent lighting (which has a relatively bluish cast); (c) fluorescent lighting image after white balance using the “max RGB” approach; (d) fluorescent lighting image after white balance using the “brightest pixel” approach



that all off-balance neutral colors will get mapped to balanced neutrals. This type of white balance may be represented as:

$$\begin{bmatrix} R_{\text{out}} \\ G_{\text{out}} \\ B_{\text{out}} \end{bmatrix} = \begin{bmatrix} k_r & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & k_b \end{bmatrix} \begin{bmatrix} R_{\text{in}} \\ G_{\text{in}} \\ B_{\text{in}} \end{bmatrix} \quad (2)$$

This technique fails when most of the image is highly chromatic, e.g., a closeup image of a brightly colored flower. See Fig. 3 for an example. Clearly, there are cases where each of these approaches fail. The search for a generic white balance algorithm has led to heuristic approaches that perform the scaling only for less-chromatic colors. In [5], Kehtarnavaz et al. present a technique called “scoring” to perform automatic white balance.

The above approaches that deal with pixel values in the RGB color space have been augmented and significantly improved by being performed in transformed color spaces. A general representation of such approaches may be given by the following equation:

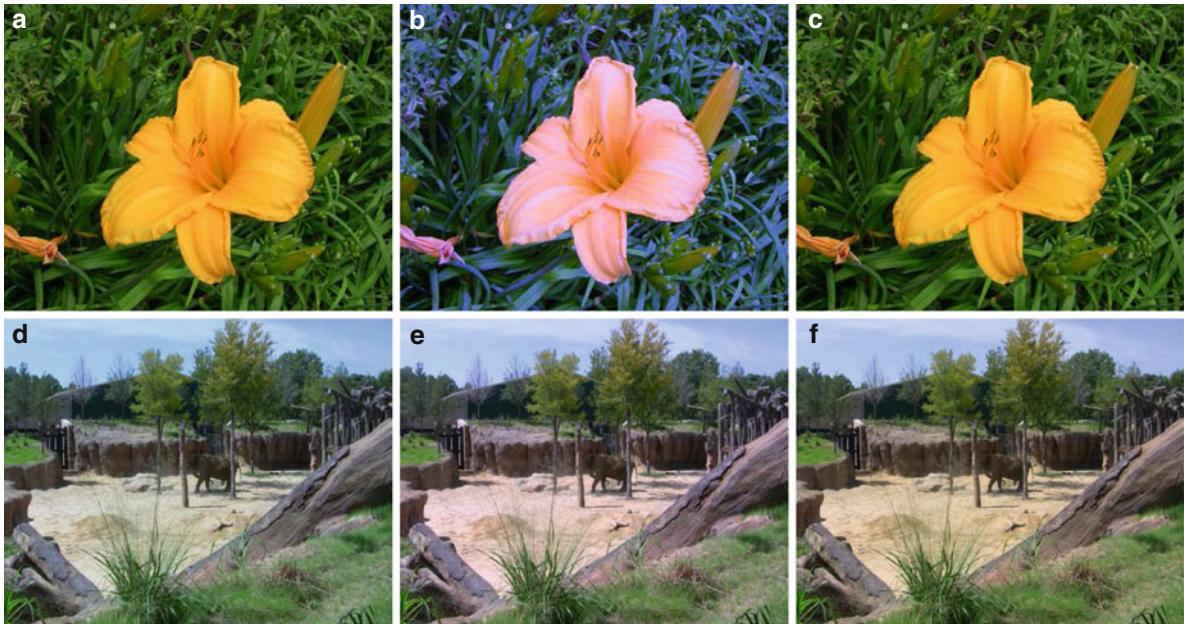
$$\begin{bmatrix} R_{\text{out}} \\ G_{\text{out}} \\ B_{\text{out}} \end{bmatrix} = f^{-1} \left(\mathbf{B} f \left(R_{\text{in}}, G_{\text{in}}, B_{\text{in}} \right) \right) \quad (3)$$

where the subscript “in” denotes the input pixels and subscript “out” denotes the output white-balanced pixel values, f and f^{-1} denote a potentially nonlinear transformation function and its inverse, respectively, and \mathbf{B} denotes a generic matrix operator on the transformed RGB input values. The choice of the transformation is highly dependent on the preference of the designer. Fairchild’s book documents the most commonly used color spaces and equations necessary to transform the input color into the appropriate color spaces [2].

In [1], Barnard et al. present white balance techniques in a historical perspective and illustrate the performance of the algorithms on synthetic data. It is not always the case that white balance must be done automatically. Some cameras, particularly the more expensive ones, put more emphasis on “preprogrammed white balance” or “manual white balance” than their automatic counterparts.

Open Problems

More sophisticated techniques for estimating the color of the illuminant (e.g., [3]) are being used in the industry, but their application in white balance continues to be an area of active research with (spatially



White Balance, Fig. 3 (a, d) Image captured with incorrect white balance; (b, e) the same images after performing white balance using the gray-world assumption; (c, f) images rendered with white balances preferred by the authors

aware) chromatic adaptation transforms [2, 4] being increasingly used to compensate for capture-related white balance issues. Some of these adaptation transforms are highly nonlinear in their formulation and require rather sophisticated hardware and software to be appropriately applied. It is important to understand that once the image is captured by the digital camera – without absolute knowledge of the illuminant – it is practically impossible to perform a perfect white balance and every rendition will be considered a “preferred” image rendition.

References

- Barnard K, Cardei V, Funt B (2002) A comparison of computational color constancy algorithms – part i: methodology and experiments with synthesized data. *IEEE Trans Image Process* 11(9):972–983
- Fairchild MD (2005) Color appearance models. The Wiley-IS&T series in imaging science and technology, 2nd edn. Wiley, Chichester
- Finlayson GD, Hordley SD, Tastl I (2003) Gamut constrained illuminant estimation. In: Proceedings of the ICCV03. Nice, France, pp 792–799
- Hunt RWG (2004) Reproduction of colour, 6th edn. Wiley, Chichester
- Kehtarnavaz N, Oh HJ, Yoo Y (2002) Development and real-time implementation of auto white balancing scoring algorithm. *J Real Time Imaging* 8:379–386
- Ramanath R, Snyder WE, Yoo Y, Drew MS (2005) Color image processing pipeline in digital still cameras. *IEEE Signal Process Mag Spec Issue Color Image Process* 22:34–43
- Wyszecki G, Stiles WS (1982) Color science: concepts and methods, quantitative data and formulas, 2nd edn. Wiley, New York

White Point

► [White Balance](#)

Wide Baseline Matching

Tinne Tuytelaars
KULEUVEN Leuven, ESAT-PSI, iMinds, Leuven, Belgium

Synonyms

[Wide field of view stereo matching](#)

Related Concepts

► [Epipolar Geometry](#)

Definition

Wide baseline matching is the process of finding correspondences between two images of the same scene taken from widely separated views.

Background

Finding correspondences between images is a first and crucial step in many image processing applications, such as image registration, 3D reconstruction, object recognition, or robot navigation. As long as the images are taken from more or less the same viewpoint, this task is relatively easy. Indeed, under these conditions (usually referred to as *small baseline*), the corresponding image patch will look very similar, and also its location in the image will have changed only slightly, so a local search for the most similar image patch suffices to find correspondences.

However, when the baseline between the cameras (i.e., the distance between the camera projection centers) increases, finding correspondences becomes a significantly harder problem. First, the correspondence can appear anywhere in the second image. As a result, one has to search over the whole image to find it. This also means there are typically many more distractors to cope with. Second, also the local appearance of the image patch can change drastically. When the viewpoint of the camera changes relative to the object or scene, this causes *geometric deformations* in the image, such as rotation, rescaling, as well as foreshortening, skew, and perspective deformations. Moreover, when the geometric setup changes, also the *photometric changes* are usually more severe. The deformations depend on the 3D structure of the scene, the illumination conditions, as well as the internal and external camera parameters. Since these are typically not known beforehand, efficient and effective algorithms to cope with them have been developed, mostly based on the concept of local invariant features and robust descriptors.

Theory

To cope with wide baseline conditions, one first needs to understand the deformations caused by the change in viewpoint. If one can assume that the object is

locally planar (or can be approximated well by a plane, which is usually the case on a local scale), the geometric deformations can be modelled, in general, by a projective transformation or homography:

$$\omega \begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

with (x_1, y_1) and (x_2, y_2) the coordinates in the first and second image, respectively and with ω a scalefactor. Since on a local scale the perspective effects can be ignored, this reduces to an affine transformation:

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

Photometric changes are usually modelled by a linear model, with a scale factor s and offset t :

$$I_2(x_2, y_2) = s I_1(x_1, y_1) + t$$

with $I_*(x_*, y_*)$ the image intensity of both images.

Additionally, it is important to realize that for some locations in the image, it is easier to find correspondences than for others. Indeed, pixels that lie on a uniform part of the image cannot be distinguished from each other. Therefore, it is impossible to tell which one corresponds to which. *Local features* or *interest points* are found at locations in the image that stand out locally and can be localized accurately. They often correspond to corners or blobs. Since they have a good *repeatability*, one does not have to search the second image exhaustively for a correspondence, but only among the interest points extracted in that image. Therefore, in a wide baseline matching context, one focusses on the matching of such interest points only. Once they have been matched, extra constraints such as *epipolar geometry* can be derived, and these can help in finding additional correspondences (e.g., aiming for *dense matching*).

When dealing with wide baseline conditions, simple interest points such as *Harris corners* [3] no longer suffice, as these are not robust to the geometric deformations described above. Instead, affine invariant local feature detectors have been proposed, e.g., [5, 6, 10]. These detectors do select not only a location in the



Wide Baseline Matching, Fig. 1 Two images of the same scene, taken with a wide baseline, with correspondences

image but also the size and shape of the image patch to be considered for matching. Usually the image patches are ellipses, but parallelograms have been used as well. The detectors are called affine invariant because they have been devised so as to select corresponding ellipses/parallelograms in spite of affine deformations of the image. Based on its shape and size, the image can then be locally normalized by applying an additional affine transformation mapping the ellipse onto a circle of fixed size. This way, the effect of the geometric deformations is compensated for. The normalized image patch is then described by a robust descriptor such as SIFT [4], resulting in a high-dimensional feature vector describing the intensity pattern around the interest point. This descriptor is also made robust to the photometric deformations. Finding correspondences then boils down to finding the most similar feature vectors.

Since on a local scale features are usually not very discriminative, false matches (i.e., wrong correspondences) are unavoidable at first. Therefore, a geometric consistency check is vital to select the inliers (correct matches) and reject the outliers (false matches). This can be done, e.g., by applying a random sampling approach such as RANSAC [2] to compute the epipolar geometry. For planar scenes, RANSAC combined with a homography is often used.

The pipeline described above is the most standard way of solving the wide baseline matching problem. Alternative schemes have been proposed as well, e.g., matching lines instead of interest points [1] or applying affine transformations to the image as a whole instead of making the feature detection invariant [9]. Also, it has been shown that in many cases, scale-invariant

interest points combined with robust descriptors can already deal with viewpoint changes up to 30° and beyond [4].

Experimental Results

Figure 1 shows an example of two images of the same scene taken from different viewpoints, with the found correspondences superimposed. Further, we refer to the literature where various evaluations and comparisons of wide baseline matching methods have been proposed, e.g., [7, 8].

References

1. Bay H, Ferrari V, Van Gool L (2005) Wide-baseline stereo matching with line segments. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), vol 1, San Diego, pp 329–336
2. Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 24:381–395
3. Harris C, Stephens M A (1988) combined corner and edge detector. In: Proceedings of the 4th Alvey vision conference, Manchester, pp 147–151
4. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60(2):91–110
5. Matas J, Chum O, Martin U, Pajdla T (2002) Robust wide baseline stereo from maximally stable extremal regions. In: Proceedings of the British machine vision conference, vol 1, Cardiff, pp 384–393
6. Mikolajczyk K, Schmid C (2004) Scale and affine invariant interest point detectors. Int J Comput Vis 60(1):63–86
7. Mikolajczyk K, Tuytelaars T, Schmid C, Zisserman A, Matas J, Schaffalitzky F, Kadir T, Van Gool L (2005)

- A comparison of affine region detectors. *Int J Comput Vis* 65(1–2): 43–72
8. Moreels P, Perona P (2006) Evaluation of features detectors and descriptors base on 3D objects. *Int J Comput Vis* 73(3):263–284
9. Morel JM, Yu G (2009) ASIFT, a new framework for fully affine invariant image comparison. *SIAM J Imaging Sci* 2(2):438–469
10. Tuytelaars T, Van Gool L (2004) Matching widely separated views based on affine invariant regions. *Int J Comput Vis* 59(1):61–85

Wide Field of View Stereo Matching

- [Wide Baseline Matching](#)

Wide-angle Camera

- [Omnidirectional Camera](#)

List of Entries

- Action Prototype Trees
Action Recognition
Active Calibration
Active Camera Calibration
Active Contours
Active Sensor (Eye) Movement Control
Active Stereo Vision
Active Vision
Activity Analysis
Activity Recognition
AdaBoost
Adaptive Boosting
Adaptive Gains
Affine Alignment
Affine Camera
Affine Invariants
Affine Projection
Affine Registration
Affordances
Affordances and Action Recognition
Algebraic Curve
Algebraic Surface
Analytic Reflectance Functions
Animat Vision
Anisotropic Diffusion
Aperture Ghosting
Appearance Scanning
Appearance-Based Human Detection
Appearance-Based Human Tracking
Appearance-Based Pedestrian Detection
Articulated Pose Estimation
Asperity Scattering
Automatic Gait Recognition
Automatic Scale Selection
Automatic White Balance (AWB)
Backscattering
Bas-Relief Ambiguity
Behavior Understanding
Bidirectional Texture Function and 3D Texture
Blackbody Radiation
Blackbody Radiator
Blind Deconvolution
Blur Estimation
Blur Kernel Estimation
Body Configuration Recovery
Boundary Detection
Boundary Extraction
BRDF Measurement
BRDF Models
Calculus of Variations
Calibration
Calibration of a Non-single Viewpoint System
Calibration of Multi-Camera Setups
Calibration of Projective Cameras
Calibration of Radiometric Falloff (Vignetting)
Camera Calibration
Camera Extrinsic Parameters
Camera Model
Camera Parameters (Internal, External)
Camera Parameters (Intrinsic, Extrinsic)
Camera Pose
Camera Response Function
Camera-Shake Blur
Catadioptric Camera
Catoptrics
Center of Projection
Change Detection
Chromaticity
CIE Standard Illuminant
Clifford Algebra
Clipping

Cluster Sampling
Coefficient Rule
Cognitive Agent
Cognitive System
Cognitive Vision
Color Adaptation
Color Appearance Models
Color Balance
Color Constancy
Color Difference
Color Discrepancy
Color Dissimilarity
Color Management
Color Model
Color Similarity
Color Spaces
Color Specification Systems
Compressed Sensing
Compressive Sensing
Computational Symmetry
Computer Vision
Computing Architectures for Machine Perception
Concept Languages
Concurrent Mapping and Localization (CML)
Conditional Random Fields
Contour Detection
Convex Minimization
Cook-Torrance BRDF
Cook-Torrance Model
Coplanarity Constraint
Corner Detection
Cross Entropy
Curvature
Curvedness
Curves
Curves in Euclidean Three-Space
Data Augmentation
Data Fusion
Deblurring
Deconvolution
Defocus Blur
Dehazing
Dehazing and Defogging
Denoising
Dense 3D Modeling
Dense Reconstruction
Depth Distortion
Depth Estimation
Depth from Scattering
Depth Imaging
Descattering
Description Logics
Dichromatic Reflection Model
Differential Geometry of Graph Spaces
Differential Geometry of Surfaces in Three-Dimensional Euclidean Space
Differential Invariants
Diffuse Reflectance
Diffuse Shading
Diffusely Scattered Reflectance
Diffusion
Diffusion Filtering
Diffusion PDEs
Digital Matting
Digitization
Dimension Reduction
Dimensional Compression
Dimensional Embedding
Dimensionality Reduction
Dioptrics
Discrete Adaboost
Discriminative Random Fields
Disocclusion
Distance Estimation
Dual Differential Geometry
Dynamic Optimization
Dynamic Programming
Edge Detection
Eigenspace Methods
Eight-Point Algorithm
Ellipse Fitting
Ellipse Matching
EM-Algorithm
Environment Mapping
Epipolar Constraint
Epipolar Geometry
Error Concealment
Error-Correcting Graph Matching
Error-Tolerant Graph Matching
Essential Matrix
Euclidean Geometry
Evolution of Robotic Heads
Expectation Maximization Algorithm
Exploration: Simultaneous Localization and Mapping (SLAM)
Extended Gaussian Image (EGI)
Extrinsic Parameters

| | |
|--|---|
| Face Identification | Image Enhancement and Restoration |
| Face Modeling | Image Inverse Problems |
| Face Reconstruction | Image Mosaicing |
| Factorization | Image Plane |
| Feature Reduction | Image Registration |
| Feature Selection | Image Stitching |
| Field of View | Image-Based Lighting |
| Field of Vision | Image-Based Modeling |
| Filling In | Image-Based Rendering |
| Fisher-Rao Metric | Image-Based Rendering (IBR) |
| Fisheye Camera | Implicit Polynomial Curve |
| Fisheye Lens | Implicit Polynomial Surface |
| Fluorescent Lighting | Incident Light Measurement |
| Focal Length | Inexact Matching |
| Focus Bracketing | Information Fusion |
| Foreground-background Assignment | Inherent Optical Properties |
| Fresnel Conditions | Inpainting |
| Fresnel Equations | Interactive Segmentation |
| Fresnel's Law | Interface Reflection |
| Fundamental Matrix | Interpretation of Line Drawings |
| Gait Analysis | Interreflections |
| Gait Biometrics | Intrinsic Images |
| Gait Recognition | Intrinsic Parameters |
| Gamut Mapping | Intrinsics |
| Gaze Control | Inverse Compositional Algorithm |
| Generalized Bas-Relief (GBR) Transformation | Ip Camera |
| Generic Object Recognition | Irradiance |
| Generically Describable Situation | Isotropic Differential Geometry in Graph Spaces |
| Geodesic Active Contour | Iterative Closest Point (ICP) |
| Geodesics, Distance Maps, and Curve Evolution | Ives Transform |
| Geometric Algebra | Kalman Filter |
| Geometric Calibration | Kalman-Bucy Filter |
| Geometric Fusion | Karhunen-Loëve Transform (KLT) |
| Geons | Kernel Estimation |
| Gesture Recognition | KF |
| Gradient Vector Flow | Kinematic Chain Motion Models |
| GVF | Kinematic Motion Models |
| Hand-Eye Calibration | Labeling |
| Hazard Detection | Lambertian Model |
| Heterogeneous Parallel Computing | Lambertian Reflectance |
| High-Performance Computing in Computer Vision | Lambert's Law |
| Histogram | Learning from Partial Labels |
| Human Appearance Modeling and Tracking | Lens Distortion Correction |
| Human Motion Classification | Lens Distortion, Radial Distortion |
| Human Pose Estimation | Lens Flare |
| ICP | Lens Flare and Lens Glare |
| Illumination Estimation, Illuminant Estimation | Light Field |
| Image Alignment | Light Transmission and Reflection Coefficients |
| Image Decompositions | |

| | |
|---|---|
| Line Drawing Labeling | Object Class Recognition (Categorization) |
| Linear Programming | Object Detection |
| Linguistic Techniques for Event Recognition | Object Extraction |
| Local Invariants | Object Models |
| LP | Object Motion Blur |
| Lumigraph | Object Parameterizations |
| Machine Recognition of Objects | Object Recognition |
| Many-to-Many Graph Matching | Object Representations |
| Matte Extraction | Object Segmentation |
| Maximum Likelihood Estimation | Obstacle Detection |
| Maximum Likelihood Estimator | Occlusion Boundaries from Motion |
| Mesostructure | Occlusion Boundaries from Stereo Matching |
| Methods of Image Recognition In a Low-Dimensional Eigenspace | Occlusion Detection |
| Microgeometry | Occlusion Handling |
| Micro Scale Structure | Omnidirectional Camera |
| Mirrorlike Reflection | Omnidirectional Vision |
| Mirrors | Opacity |
| Mobile Observers | Opaque |
| MoCap | Optic Flow |
| Model-Based Object Recognition | Optical Axis |
| Monte Carlo Annealing | Optical Center |
| Morphology, Form Analysis | Optical Flow |
| Motion Blur | Optimal Estimation |
| Motion Capture | Optimal Parameter Estimation |
| Motion Capturing | Oren-Nayar Reflectance Model |
| Motion Deblurring | Osculating Paraboloids |
| Motion Tracking | Osculating Quadric |
| Multi-baseline Stereo | Out of Focus Blur |
| Multi-camera Calibration | Panoramic Camera |
| Multi-camera Human Action Recognition | Panoramic Image Generation |
| Multi-focus Images | Panoramic Stitching |
| Multimedia Retrieval | Pan-Tilt Camera Calibration |
| Multiple Similarity Method | Pan-Tilt-Zoom (PTZ) Camera |
| Multiple View Geometry | Pan-Tilt-Zoom Camera Calibration |
| Multiple View Stereo | Parametric Curve |
| Multiplexed Illumination | Parametric Surface |
| Multiplexed Sensing | Partitioning |
| Multisensor Data Fusion | Penumbra and Umbra |
| Multiview Geometry | Performance Capture |
| Multiview Stereo | Perspective Camera |
| Mutual Illumination | Perspective Projection |
| Network Camera | Perspective Transformation |
| Noise Removal | Phong Model |
| Non-central Camera Calibration | Phong Reflectance Model |
| Nonlinear Diffusion | Photo-Consistency |
| Normalcy Modeling | Photogrammetry |
| Numerical Methods in Curve Evolution Theory | Photometric Consistency Function |
| Numerical Partial Differential Equations | Photometric Invariants |
| | Photometric Stereo |

| | |
|--|---|
| Photon, Poisson Noise | Rigid Alignment |
| Picture Understanding | Rigid Matching |
| Piecewise Polynomial | Rigid Positioning |
| Pinhole Camera | Rigid Registration |
| Pinhole Camera Model | Rigid Transformation Estimation |
| Planckian Locus | Robot-Camera Calibration |
| Plane Sweeping | Robust Clustering |
| Plenoptic Function | Robust Methods |
| Point Spread Function Estimation | Robust Regression |
| Polarization | Saturation (Imaging) |
| Polarization Filter | Scale Selection |
| Polarized Light in Computer Vision | Scale-Invariant Image Features and Image Descriptors |
| Polarizer | Scene/Image Parsing |
| Polarizing Film | Schott Noise |
| Principal Axis | Second Order Approximation |
| Principal Component Analysis (PCA) | Segmentation |
| Principal Distance | Semantic Image Segmentation |
| Probabilistic Hill climbing | Semiautomatic |
| Projection | Semidefinite Optimization |
| Projection Matrix | Semidefinite Programming |
| Projection Transformation | Sensor Fusion |
| Projective Reconstruction | Shadow |
| Projective Structure and Motion | Shape from Outlines of Projection |
| Prototype-Based Methods for Human Movement Modeling | Shape from Scatter |
| PSF Estimation | Shape from Shadows |
| PTZ Camera Calibration | Shape from Silhouette |
| Pulling a Matte | Shape from Specular Reflections |
| Purposive Vision | Shape from Specularities |
| Radiance | Shock Graph |
| Radiometric Calibration | Shot Noise |
| Radiometric Camera Calibration | Simulated Annealing |
| Radiometric Response Function | Single Viewpoint |
| Rao Metric | Situation Graph Trees |
| Rationale for Computational Vision | Situation Scheme |
| Recognition-By-Components (RBC) Theory | Solid Texture |
| Recovery of Reflectance Properties | Space Curves |
| Reference Plane | Sparse Coding |
| Reflectance Map | Sparse Representation |
| Reflectance Models | Spatiotemporal Reasoning |
| Reflection Mapping | SPDE |
| Reflectometry | Specular Highlight |
| Relation Between Objects and Their Digital Images | Specularity, Specular Reflectance |
| Relief Texture | Spherical Camera |
| Retina | Splines |
| Retinex Algorithm | Standard Illuminants |
| Retinex Theory | Statistical Cooling |
| Retroreflection | Statistical Independence |
| | Statistical Shape Analysis |

- Stochastic Differential Equations in Infinite Dimensions
Stochastic Partial Differential Equations
Stochastic Relaxation
Structure-from-Motion (SfM)
Subpixel Estimation
Subspace Methods
Superresolution
Surface Corrugations
Surface Orientation Histogram (Discrete Version of EGI)
Surface Reconstruction
Surface Roughness
Surface Scattering
Surface Undulations
Surfaces
Surveillance Camera
SW Cut
Swendsen-Wang Algorithm
Swendsen-Wang Cut Algorithm
Symmetry Detection
Symmetry-Based X
Terminological Logics
Texture Classification
Texture Recognition
Thermal Radiator
Three Dimensional Descriptor
Three Dimensional Estimation
Three Dimensional Face Modeling
Three Dimensional Modeling from Images
Three Dimensional Reconstruction
Three Dimensional Shape Descriptor
Total Variation
Tracker-Camera Calibration
Transparency
Transparency and Translucency
Transparent Layers
Transportation Problem
Trichromatic Theory
Two Dimensional Conditional Random Fields
Underwater Effects
Underwater Light Field
Underwater Radiative Transfer Processes
User-Assisted
User-Guided
Variable Selection
Variational Analysis
Variational Method
Variational Methods
Veiling Glare
Velvety Reflectance
Video Alignment and Stitching
Video Mosaicing
Video Mosaicking
Video Retrieval
Video Search
View and Rate-Invariant Human Action Recognition
Vignetting
Vignetting Estimation
Viscosity Solution
Visibility Enhancement in Bad Weather
Vision-Based Control
Vision-Based Feedback
Visual Cognition
Visual Cortex Models for Object Recognition
Visual Hull
Visual Inference
Visual Patterns
Visual Servoing
Visual SLAM
Volumetric Texture
von Kries Hypothesis
von Kries-Ives Adaptation
Weak Perspective Projection
Weakly Supervised Learning
White Balance
White Point
Wide Baseline Matching
Wide Field of View Stereo Matching
Wide-angle Camera