



Giovanni Maria Farinella  
Sebastiano Battiato  
Roberto Cipolla *Editors*

# Advanced Topics in Computer Vision

 Springer

# **Advances in Computer Vision and Pattern Recognition**

For further volumes:  
[www.springer.com/series/4205](http://www.springer.com/series/4205)

Giovanni Maria Farinella • Sebastiano Battiato •  
Roberto Cipolla  
Editors

# Advanced Topics in Computer Vision



*Editors*

Dr. Giovanni Maria Farinella  
Dipartimento di Matematica e Informatica  
Università di Catania  
Catania, Italy

Prof. Roberto Cipolla  
Department of Engineering  
University of Cambridge  
Cambridge, UK

Prof. Sebastiano Battiato  
Dipartimento di Matematica e Informatica  
Università di Catania  
Catania, Italy

*Series Editors*

Prof. Sameer Singh  
Research School of Informatics  
Loughborough University  
Loughborough, UK

Dr. Sing Bing Kang  
Interactive Visual Media Group  
Microsoft Research  
Redmond, WA, USA

ISSN 2191-6586

Advances in Computer Vision and Pattern Recognition

ISBN 978-1-4471-5519-5

DOI 10.1007/978-1-4471-5520-1

Springer London Heidelberg New York Dordrecht

ISSN 2191-6594 (electronic)

ISBN 978-1-4471-5520-1 (eBook)

Library of Congress Control Number: 2013950636

© Springer-Verlag London 2013

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](http://www.springer.com))

# Preface

Computer vision is the science and technology of making machines that see. It is concerned with the theory, design and implementation of algorithms that can automatically process visual data to recognize objects, track and recover their shape and spatial layout.

This edited volume contains a selection of articles covering both theoretical and practical aspects of the three main area in Computer Vision: Reconstruction, Registration, and Recognition. The book provides both, an in-depth overview of challenging areas, as well as novel advanced algorithms which exploit Machine Learning and Pattern Recognition techniques to infer the semantic content of images and videos. The topics covered by the chapters include visual feature extraction, feature matching, image registration, 3D reconstruction, object detection and recognition, human actions recognition, image segmentation, object tracking, metric learning, loopy belief propagation, etc. Each chapter contains key references to the existing literature.

The authors of the chapters have been selected among the best students who attended the International Computer Vision Summer School (ICVSS) in the last years, and are co-authored by world renowned researchers in Computer Vision. ICVSS was established in 2007 to provide both an objective and clear overview and an in-depth analysis of the state-of-the-art research in Computer Vision. The courses are delivered by experts in the field, from both academia and industry, and cover both theoretical and practical aspects of real Computer Vision problems. The school is organized every year by University of Cambridge (Computer Vision and Robotics Group) and University of Catania (Image Processing Lab). Different topics are covered each year. A summary of the past Computer Vision Summer Schools can be found at: <http://www.dmi.unict.it/icvss>.

It is our hope that graduate students, young and senior researchers, and academic/industrial professionals will find the book useful for understanding and reviewing current approaches in Computer Vision, thereby continuing the mission of the International Computer Vision Summer School.

Sicily, Italy  
June 2013

Giovanni Maria Farinella  
Sebastiano Battiato  
Roberto Cipolla

## **Acknowledgements**

We would like to take this opportunity to thank all contributors of this book, and all people involved in the organization of ICVSS.

# Contents

<b>1</b>	<b>Visual Features—From Early Concepts to Modern Computer Vision</b>	<b>1</b>
	Martin Weinmann	
<b>2</b>	<b>Where Next in Object Recognition and how much Supervision Do We Need?</b>	<b>35</b>
	Sandra Ebert and Bernt Schiele	
<b>3</b>	<b>Recognizing Human Actions by Using Effective Codebooks and Tracking</b>	<b>65</b>
	Lamberto Ballan, Lorenzo Seidenari, Giuseppe Serra, Marco Bertini, and Alberto Del Bimbo	
<b>4</b>	<b>Evaluating and Extending Trajectory Features for Activity Recognition</b>	<b>95</b>
	Ross Messing, Atousa Torabi, Aaron Courville, and Chris Pal	
<b>5</b>	<b>Co-recognition of Images and Videos: Unsupervised Matching of Identical Object Patterns and Its Applications</b>	<b>113</b>
	Minsu Cho, Young Min Shin, and Kyoung Mu Lee	
<b>6</b>	<b>Stereo Matching—State-of-the-Art and Research Challenges</b>	<b>143</b>
	Michael Bleyer and Christian Breiteneder	
<b>7</b>	<b>Visual Localization for Micro Aerial Vehicles in Urban Outdoor Environments</b>	<b>181</b>
	Andreas Wendel and Horst Bischof	
<b>8</b>	<b>Moment Constraints in Convex Optimization for Segmentation and Tracking</b>	<b>215</b>
	Maria Klodt, Frank Steinbrücker, and Daniel Cremers	
<b>9</b>	<b>Large Scale Metric Learning for Distance-Based Image Classification on Open Ended Data Sets</b>	<b>243</b>
	Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka	

<b>10</b>	<b>Top-Down Bayesian Inference of Indoor Scenes . . . . .</b>	<b>277</b>
	Luca Del Pero and Kobus Barnard	
<b>11</b>	<b>Efficient Loopy Belief Propagation Using the Four Color Theorem . . . . .</b>	<b>313</b>
	Radu Timofte and Luc Van Gool	
<b>12</b>	<b>Boosting <math>k</math>-Nearest Neighbors Classification . . . . .</b>	<b>341</b>
	Paolo Piro, Richard Nock, Wafa Bel Haj Ali, Frank Nielsen, and Michel Barlaud	
<b>13</b>	<b>Learning Object Detectors in Stationary Environments . . . . .</b>	<b>377</b>
	Peter M. Roth, Sabine Sternig, and Horst Bischof	
<b>14</b>	<b>Video Temporal Super-resolution Based on Self-similarity . . . . .</b>	<b>411</b>
	Mihoko Shimano, Takahiro Okabe, Imari Sato, and Yoichi Sato	
<b>Index . . . . .</b>		<b>431</b>

# Contributors

## *Editors' Biographies*

### **Giovanni Maria Farinella**

Giovanni Maria Farinella obtained the Master degree in Computer Science (egregia cum laude) from University of Catania in 2004. He was awarded a Ph.D. degree (Computer Vision) in 2008. He became Associate Member of the Computer Vision and Robotics Research Group at University of Cambridge in 2006. He joined the Image Processing Laboratory (IPLAB) at the Department of Mathematics and Computer Science, University of Catania in 2008 as Contract Researcher. He is Contract Professor of Computer Vision at the School of Arts of Catania (since 2004) and Adjunct Professor of Computer Science at the University of Catania (since 2008). His research interests lie in the fields of Computer Vision, Pattern Recognition and Machine Learning. He has edited three volumes and co-authored more than 50 papers in international journals, conference proceedings and book chapters. He is a co-inventor of 3 international patents. Dr. Farinella also serves as a reviewer and on the programme committee for major international journals and international conferences. He has participated to several international and national research projects. Dr. Farinella founded (in 2006) and currently directs the International Computer Vision Summer School ([www.dmi.unict.it/icvss](http://www.dmi.unict.it/icvss)).

### **Sebastiano Battiato**

Sebastiano Battiato received his degree in computer science (summa cum laude) in 1995 from University of Catania and his Ph.D. in computer science and applied mathematics from University of Naples in 1999. From 1999 to 2003 he was the leader of the “Imaging” team at STMicroelectronics in Catania. He joined the Department of Mathematics and Computer Science at the University of Catania as assistant professor in 2004 and became associate professor in the same department in 2011. His research interests include image enhancement and processing, image coding, camera imaging technology and multimedia forensics. He has edited 4 books

and co-authored more than 150 papers in international journals, conference proceedings and book chapters. He is a co-inventor of about 15 international patents, reviewer for several international journals, and he has been regularly a member of numerous international conference committees. Professor Battiato has participated in many international and national research projects. Chair of several international events (IWCV2012, ECCV2012, VISAPP 2012-2013-2014, ICIAP 2011, ACM Mi-For 2010-2011, SPIE EI Digital Photography 2011-2012-2013, etc.). He is an associate editor of the IEEE Transactions on Circuits and System for Video Technology and of the SPIE Journal of Electronic Imaging. Guest editor of the following special issues: “Emerging Methods for Color Image and Video Quality Enhancement” published on EURASIP Journal on Image and Video Processing (2010) and “Multimedia in Forensics, Security and Intelligence” published on IEEE Multimedia Magazine (2012). He is the recipient of the 2011 Best Associate Editor Award of the IEEE Transactions on Circuits and Systems for Video Technology. He is director (and co-founder) of the International Computer Vision Summer School (ICVSS). He is a senior member of the IEEE.

### **Roberto Cipolla**

Roberto Cipolla obtained the B.A. degree (Engineering) from the University of Cambridge in 1984 and an M.S.E. (Electrical Engineering) from the University of Pennsylvania in 1985. From 1985 to 1988 he studied and worked in Japan at the Osaka University of Foreign Studies (Japanese Language) and Electrotechnical Laboratory. In 1991, he was awarded a D.Phil. (Computer Vision) from the University of Oxford and from 1991–1992 was a Toshiba Fellow and engineer at the Toshiba Corporation Research and Development Centre in Kawasaki, Japan. He joined the Department of Engineering, University of Cambridge in 1992 as a Lecturer and a Fellow of Jesus College. He became a Reader in Information Engineering in 1997 and a Professor in 2000. He is a Fellow at the Royal Academy of Engineering (since 2010); also a Professor of Computer Vision at the Royal Academy of Arts, London (since 2004) and Director of Toshiba’s Cambridge Research Laboratory (since 2007). His research interests are in computer vision and robotics and include the recovery of motion and 3D shape of visible surfaces from image sequences; object detection and recognition; novel man–machine interfaces using hand, face and body gestures; real-time visual tracking for localization and robot guidance; applications of computer vision in mobile phones, visual inspection and image-retrieval and video search. He has authored 3 books, edited 8 volumes and co-authored more than 300 papers. Professor Cipolla founded (in 2006) and currently directs the International Computer Vision Summer School.

## ***List of Contributors***

**Lamberto Ballan** Media Integration and Communication Center, University of Florence, Florence, Italy

**Michel Barlaud** I3S Laboratory, University of Nice-Sophia Antipolis, Sophia Antipolis, France

**Kobus Barnard** University of Arizona, Tucson, USA

**Wafa Bel Haj Ali** I3S Laboratory, University of Nice-Sophia Antipolis, Sophia Antipolis, France

**Marco Bertini** Media Integration and Communication Center, University of Florence, Florence, Italy

**Horst Bischof** Institute for Computer Graphics and Vision, Graz University of Technology, Graz, Austria

**Michael Bleyer** Vienna University of Technology, Vienna, Austria; Microsoft Redmond, Redmond, USA

**Christian Breiteneder** Vienna University of Technology, Vienna, Austria

**Minsu Cho** INRIA/École Normale Supérieure, Paris, France

**Aaron Courville** Université de Montréal, Montréal, Quebec, Canada

**Daniel Cremers** Department of Informatics, TU München, Garching, Germany

**Gabriela Csurka** Xerox Research Centre Europe, Meylan, France

**Alberto Del Bimbo** Media Integration and Communication Center, University of Florence, Florence, Italy

**Luca Del Pero** University of Arizona, Tucson, USA

**Sandra Ebert** Max Planck Institute for Informatics, Saarbrücken, Germany

**Maria Klodt** Department of Informatics, TU München, Garching, Germany

**Kyoung Mu Lee** Seoul National University, Seoul, Korea

**Thomas Mensink** LEAR Team – INRIA Grenoble, Montbonnot, France

**Ross Messing** University of Rochester, Rochester, NY, USA; Tendant Vision Science, Inc., Pittsburgh, PA, USA

**Frank Nielsen** Department of Fundamental Research, Sony Computer Science Laboratories, Inc., Tokyo, Japan; LIX Department, Ecole Polytechnique, Palaiseau, France

**Richard Nock** CEREGMIA, Université Antilles-Guyane, Martinique, France

**Takahiro Okabe** The University of Tokyo, Tokyo, Japan

**Chris Pal** École Polytechnique de Montréal, Montréal, Quebec, Canada

**Florent Perronnin** Xerox Research Centre Europe, Meylan, France

**Paolo Piro** Istituto Italiano di Tecnologia, Genova, Italy

**Peter M. Roth** Institute for Computer Graphics and Vision, Graz University of Technology, Graz, Austria

**Imari Sato** National Institute of Informatics, Tokyo, Japan

**Yoichi Sato** The University of Tokyo, Tokyo, Japan

**Bernt Schiele** Max Planck Institute for Informatics, Saarbrücken, Germany

**Lorenzo Seidenari** Media Integration and Communication Center, University of Florence, Florence, Italy

**Giuseppe Serra** Media Integration and Communication Center, University of Florence, Florence, Italy

**Mihoko Shimano** The University of Tokyo, Tokyo, Japan; PRESTO, Japan Science and Technology Agency, Tokyo, Japan

**Young Min Shin** Seoul National University, Seoul, Korea

**Frank Steinbrücker** Department of Informatics, TU München, Garching, Germany

**Sabine Sternig** Institute for Computer Graphics and Vision, Graz University of Technology, Graz, Austria

**Radu Timofte** VISICS, ESAT-PSI/iMinds, KU Leuven, Leuven, Belgium

**Atousa Torabi** Université de Montréal, Montréal, Quebec, Canada

**Luc Van Gool** VISICS, ESAT-PSI/iMinds, KU Leuven, Leuven, Belgium; Computer Vision Lab, D-ITET, ETH Zurich, Zurich, Switzerland

**Jakob Verbeek** LEAR Team – INRIA Grenoble, Montbonnot, France

**Martin Weinmann** Institute of Photogrammetry and Remote Sensing, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

**Andreas Wendel** Institute for Computer Graphics and Vision, Graz University of Technology, Graz, Austria

# Chapter 1

## Visual Features—From Early Concepts to Modern Computer Vision

Martin Weinmann

**Abstract** Extracting, representing and comparing image content is one of the most important tasks in the fields of computer vision and pattern recognition. Distinctive image characteristics are often described by visual image features which serve as input for applications such as image registration, image retrieval, 3D reconstruction, navigation, object recognition and object tracking. The awareness for the need of adequately describing visual features emerged in the 1920s in the domain of visual perception, and fundamental concepts have been established to which almost every approach for feature extraction can be traced back. After the transfer of the basic ideas to the field of computer vision, much research has been carried out including the development of new concepts and methods for extracting such features, the improvement of existing ideas and numerous comparisons of different methods. In this chapter, a definition of visual features is derived, and different types are presented which address both the spatial and the spatio-temporal domain. This includes local image features, which are used in a variety of computer vision applications, and their evolution from early ideas to powerful feature extraction and matching methods.

### 1.1 Introduction

“A picture is worth a thousand words.” According to this proverb, a single image already contains as much information as a large amount of descriptive text. However, this information has to be extracted and analyzed in an efficient way. Hence, in image processing, it has become popular to transform images or relevant image patches into a compact description using features. The number of different features proposed in literature has increased significantly within the last decades. However, usually features are selected which visually show some special characteristics. These features are now referred to as *visual features*. Such visual features are of great importance in computer vision and pattern recognition as they alleviate crucial tasks such as image registration, image retrieval, object recognition, object tracking, navigation of autonomous vehicles, scene reconstruction or scene interpre-

---

M. Weinmann (✉)

Institute of Photogrammetry and Remote Sensing, Karlsruhe Institute of Technology (KIT),

Englerstr. 7, 76131 Karlsruhe, Germany

e-mail: [martin.weinmann@kit.edu](mailto:martin.weinmann@kit.edu)

tation. Surprisingly, approaches for extracting such features representing important visual details of an image even allow for predicting image memorability [64]. For these reasons, the goal of this chapter is to provide a survey on visual features. In contrast to previous work [85, 103, 136], the contribution of this chapter is to categorize different approaches to different types of visual features, to link all types of visual features to a common source in visual perception as well as to extend the definition from single images to image sequences. Additionally, it is shown for the very popular type of local features arising from image locations with special characteristics within a small spatial neighborhood how these features can be detected, how they can be described and how they can be recognized in a similar image again.

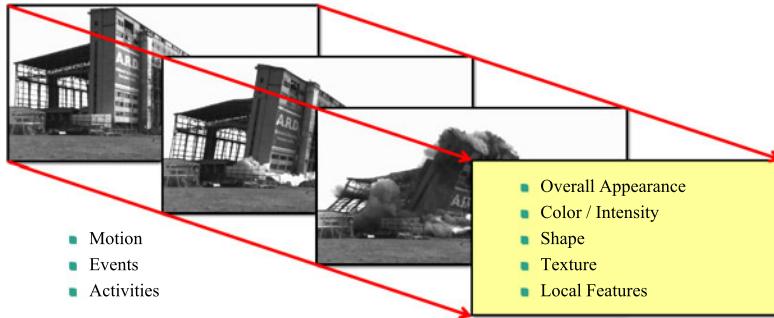
General considerations on visual features are carried out in Sect. 1.2. This includes deriving a definition of the term *visual feature*, checking the traces back to the origin of visual features resulting from investigations on visual perception, considering the suitability of visual features with respect to typical computer vision applications as well as revisiting properties of a *good* visual feature. These general considerations are followed by a brief summary of various existing types of visual features in Sect. 1.3. The intention of this chapter is not to provide a complete and detailed survey over all of the existing approaches for extracting visual features, but rather to revisit the most important ideas. As local features represent a very promising type of visual features suitable for various kinds of applications, they are of great interest in current research. Hence, Sect. 1.4 provides a more detailed analysis by showing how an idea concerning the detection and the description of such features has been transferred to digital image processing and how it evolved over time. Finally, in Sect. 1.5, the chapter is concluded with a summary and an outlook on current research trends related to visual features.

## 1.2 Visual Features

Throughout the past decades, there has been growing interest in visual features and many investigations have been carried out as such features significantly alleviate common tasks in computer vision. Hence, it seems to be necessary to derive a general definition of the term *visual feature* which is done in Sect. 1.2.1. The idea of using such visual features is then motivated in Sect. 1.2.2, and the necessity of such features for typical computer vision applications is discussed in Sect. 1.2.3. As it is possible to distinguish between different types of visual features, the question about the best feature type might arise which is finally addressed in Sect. 1.2.4.

### 1.2.1 What Is a Visual Feature?

In order to describe the characteristics of a visual feature, it is worth thinking of the main idea of any feature. In general, a *feature* describes a property by which real or abstract elements or objects can be distinguished. Thus, a feature represents a piece of information which may be relevant for solving a special task. This definition is



**Fig. 1.1** Different types of visual features in images and image sequences

very general and, depending on the respective application, different types of features may differ in their suitability.

The meaning of a visual feature has even been investigated beyond computer vision. According to the *Designs Act 2003* [36], a *visual feature*, in relation to a product, includes the shape, configuration, pattern and ornamentation of the product. Furthermore, a visual feature may serve a functional purpose, but not necessarily, and special attributes like the feel of the product and the materials used in the product are explicitly excluded. Based on this definition, a *design*, in relation to a product, then means the overall appearance of the product resulting from one or more visual features of the product.

In the field of computer vision, a general description of a visual feature can be derived in a very similar way. A visual feature describes a special property of an image as a whole or an object within the image and it can either be a local property or a global characteristic of the image. Thus, for instance, a visual feature might be color in case of color images, intensity in case of grayscale images, shape, size, orientation or texture. Extending this definition to whole image sequences and thus videos, further properties like the motion of objects, the respective velocity, acceleration and direction, single trajectories or the time span within which an object is present may also be considered as visual features. This extension also allows for including more complex features in form of events in an image sequence as well as high-level features like activities describing the behavior of persons or objects within a scene. As a video represents a special image sequence, where the temporal difference between successive images is constant, only the more general term is used in the following. Consequently, a first but very coarse scheme for the classification of different types of visual features can be derived which is illustrated in Fig. 1.1.

### 1.2.2 What Is the Idea of Using Visual Features?

The derived definition of a visual feature is based on typical characteristics which can be observed in images or image sequences. Early studies on human visual perception can be traced back to ancient Greek theories on how vision is carried out.



**Fig. 1.2** Illusion showing that an image might be more than the sum of its parts (“*The Forest Has Eyes*” © Bev Doolittle/The Greenwich Workshop, Inc.)

Since then, many investigations yielded improved insights into human vision. Different psychological theories have been established among which *Structuralism*, *Gestalt Theory* and the *Theory of Ecological Optics* cover important ideas which finally led to cognitive and computational approaches to visual perception. With all these ideas, a growing awareness of the importance of visual features emerged.

Structuralism, where perception is based on a large number of sensory atoms measuring color at specific locations within the field of view, represents a very simple principle of visual perception. However, when considering visual illusions such as the one depicted in Fig. 1.2, it becomes apparent that the whole might be different from the sum of its parts, which has already been that concisely summarized by Aristotle. Hence, visual perception not only involves local image characteristics such as lines, angles, colors or shape, but also grouping processes for what has been detected with the senses and the respective interpretation. In Fig. 1.2, the grouping processes arise from the fact that human vision is biased and trained to see faces. The concept of *Holism* where the whole cannot be represented by utilizing the sum of its parts alone due to relations between the parts has been addressed in Gestalt Theory. The term *gestalt* can be seen as a German synonym for shape, form, figure or configuration. Summarizing the main ideas of Gestalt Theory, it has already been stated in 1924 that “there are entities where the behavior of the whole cannot be derived from its individual elements nor from the way these elements fit together; rather the opposite is true: the properties of any of the parts are determined by the intrinsic structural laws of the whole” [146]. These structural laws of the whole correspond to relations between the parts and, as a consequence, special principles for grouping elements of an image have been proposed. According to [145] and [70], the visual system uses different principles for automatically grouping elements into patterns such as *proximity*, *similarity*, *closure*, *symmetry*, *common fate* (i.e. *common motion*) and *continuity*. In [50], further principles like *homogeneity* and *contour* are proposed.

The homogeneity is introduced as *texture* and the contour represents a physical edge caused by an abrupt change in texture or color. However, describing the whole content of an image might be very complex and therefore, the concept of *Reductionism* according to which an image can be explained by reduction to its fundamental parts became more and more important. In [50] and [51], the Theory of Ecological Optics has been presented which focuses on active and direct perception of visual information. Involving basic principles of Gestalt Theory as well as aspects of information theory, a modified set of 10 principles concerning varieties of continuous regularity, discontinuous regularity or recurrence, proximity and situations involving interaction is presented in [7]. These principles arise from three main observations:

- Information is not uniformly distributed but rather concentrated along contours and especially at points on a contour at which its direction changes, that is, corners.
- Contours are caused by changes of homogeneity with respect to color or texture.
- The degree of homogeneity, that is, texture, can be considered as a characteristic.

Consequently, the perceived information contains a certain degree of redundancy. Visual features should therefore provide a compact representation of the image content. Hence, typical types of visual features in an image might be corners, contours (i.e., shape), color/intensity or homogeneity (i.e., texture). However, exactly these types of visual features have already been addressed in Fig. 1.1.

The cognitive and computational approaches to visual perception are based on combining descriptive principles already proposed in Gestalt Theory and the Theory of Ecological Optics with a model of perceptual processing. According to [87], the early processing of visual information therefore focused on achieving a primitive but rich description of intensity changes present in an image which has been denoted as *primal sketch*. This description is derived from different kinds of intensity changes like edges, lines and blobs as well as additional parameters specifying attributes like position, size, orientation, contrast, termination and fuzziness. As the primal sketch represents a relatively impractical description, further processes for grouping elements are essential which are similar to those of Gestalt Theory. Further involving concepts of Reductionism has proven to be feasible, as an image typically contains very much and redundant information. Exploiting all this information for a special application results in a large amount of input data which leads to problems if tasks cannot be solved fast enough. Hence, only the most significant information of an image can be exploited which typically corresponds to visual features. Thus, feature extraction is commonly introduced for a variety of tasks in computer vision. Suitable features even allow for comparing images or image patches with different size as well as image content in form of objects which vary in scale and orientation.

### 1.2.3 Are Visual Features Always Necessary?

Although many investigations within the last few decades have clearly demonstrated the high potential of using visual features in a variety of applications, some investi-

gations show that several tasks can also be solved without using such features. The automatic and accurate alignment of captured point clouds, for instance, is an important task for digitization, reconstruction and interpretation of 3D scenes. Active sensors such as terrestrial laser scanners can cope with measuring the 3D distance of scene points and simultaneously capturing image information in form of either co-registered camera images or panoramic reflectance images representing the respective energy of the backscattered laser light. The recorded 3D point clouds typically provide a high point density as well as a high measurement accuracy. Hence, the registration of two partially overlapping scans can be carried out based on the 3D geometry alone and thus without the need of visual features if the 3D structure of the scene is distinctive enough.

Considering the example of point cloud registration, standard approaches such as the *Iterative Closest Point (ICP) algorithm* [18, 117] or *Least Squares 3D Surface Matching (LS3D)* [53] only exploit spatial 3D information. Whereas the ICP algorithm iteratively minimizes the difference between two point clouds, the LS3D approach minimizes the distance between matched surfaces. Other approaches focus on the distribution of the points on 2D scan slices [22] or in 3D [86]. For environments with regular surfaces, various types of geometric primitives such as planes [22, 106, 139] or more complex geometric features like spheres, cylinders or tori [109] have been proposed. In scenes without regular surfaces, the registration can rely on descriptors representing local surface patches which may, for instance, be derived from geometric curvature or normal vectors of the local surface [8].

Several investigations, however, have shown that the registration of point clouds can efficiently be supported by involving visual features derived from 2D imagery. As both range and intensity information are typically measured on a regular scan grid resulting from a cylindrical or spherical projection, they can be represented as images. From these images, distinctive feature points can be extracted and reliable feature point correspondences between the images of different scans can be derived. The extraction of such features has been proposed from range images [11, 127], from intensity images [20, 66, 140] and from co-registered camera images [4, 10, 17]. In general, features in the intensity images provide a higher level of distinctiveness than features in the respective range images [122] and, probably, information not yet represented in the range measurements. Projecting the information of distinctive 2D points to 3D space according to the respective range information yields sparse point clouds describing physically almost identical 3D points. The point cloud registration may then exploit the reliable 3D/3D correspondences [122] or 3D/2D correspondences [143, 144] between different scans which typically involves a RANSAC-based scheme [42]. Thus, the reduction to sparse point clouds significantly reduces the time effort and even tends to improve the accuracy of the registration results as the amount and influence of outliers can be reduced [140, 143, 144]. Furthermore, these approaches can directly be transferred to Time-of-Flight cameras or devices based on the use of structured light (e.g., Microsoft Kinect). Consequently, most of the current approaches addressing point cloud registration consider both range and intensity information for reaching an increased performance, although the alignment can also be carried out without using visual features if the scene provides a sufficiently distinctive 3D structure.

### 1.2.4 What Is the Best Visual Feature?

Considering the very general definition, a large variety of visual feature types can be extracted which indicates that, depending on the respective application and the data available, different types of visual features will surely differ in their suitability. For instance, it might be sufficient to detect people via skin color extraction or to extract bright objects in front of a dark background by simple thresholding techniques. Both tasks can be realized very easily but if additional aspects are essential, for example, when the detected persons or objects should be identified by comparison to elements of a large database, the selected visual features may not provide enough information. For this purpose, other feature types are more appropriate. This example clearly shows that, instead of searching for the *best* feature type, it is more desirable to get characteristics for a *good* feature type.

Based on approaches for extracting adequate features in pattern recognition, it can be noticed that several ideas may obviously be adapted to characterize visual features. Thus, in general, besides being observable, a good visual feature could at least satisfy several of the most commonly desired characteristics:

- The extracted features should be *distinctive* and thus significantly differ from their spatial neighborhood.
- The extracted features should be *invariant* to irrelevant changes of the respective image or image sequence from which they have been extracted, and *robust* against noise effects.
- The feature extraction and the comparison of extracted features should be *efficient* with respect to computational effort.
- A visual feature or a set of visual features should be *comparable* and thus allow for detecting similar content in different images or image sequences.
- The feature type should be highly *relevant* with respect to the application. For different classes with (possibly abstract) objects, the features should provide only small variations when comparing objects within the same class and large variations when comparing objects of different classes.

## 1.3 The Different Types of Visual Features

Once the awareness for transferring the idea of visual features to computational concepts had emerged, different methods for extracting these features evolved. One of the first surveys on feature extraction has already been carried out in 1969 [76]. Since then, a very large number of approaches for detecting different kinds of features has been presented of which only a relatively small number still has significant impact on current research. Hence, describing all of these methods for feature extraction is beyond the scope of this chapter and, as visual features typically correspond to a certain location or region within an image and its temporal behavior, only the most important ideas concerning the *spatial domain* (Sect. 1.3.1) and the *spatio-temporal domain* (Sect. 1.3.2) are considered. The main focus is on spatial



**Fig. 1.3** An image and an image sequence showing different types of visual features

features as image analysis is also required when considering image sequences and many ideas concerning spatial features have therefore been transferred to image sequences. The existence of a preferred feature type with almost general applicability is finally discussed in Sect. 1.3.3. For all of the following considerations, an example is provided in Fig. 1.3 which illustrates a variety of visual features in images and image sequences such as color, shape, texture, local features and features arising from scene dynamics.

### 1.3.1 Spatial Features

*Spatial features* only address the image domain and can therefore be directly extracted from an image. The variety of such spatial features covers overall appearance, color or intensity, shape, texture and local features. Whereas those features addressing color/intensity, shape and texture may be rather global or local, the local features only rely on a small spatial image neighborhood. Furthermore, visual features may even be defined depending on the respective application. For recognizing the same or similar information in a different image, the features are often assigned a special description encapsulating the respective image properties.

#### 1.3.1.1 Overall Appearance

Currently, several approaches still address the idea of finding an adequate method for achieving a holistic representation of an image. Early approaches were based on the *Fourier transform*, *generic Fourier descriptors* [150] or *moment invariants* [61] and modified variants such as the *Zernike moment invariants (ZMIs)* [132, 133] or the *orthogonal Fourier–Mellin moments (OFMMs)* [123]. For the latter two, those moments with a lower order capture the low-frequency information and thus the coarse structure of an image, whereas the moments with a higher order capture the

high-frequency information and thus image details. Hence, an image can be represented as a superposition of such moments. A further approach exploiting a coarse-to-fine image representation is based on the *Discrete Cosine Transform (DCT)* [3] which has proven to be well-suited for image compression. Some more recent approaches directly address the issue of scene recognition. The *GIST descriptor* [105] focuses on obtaining the *gist*, an abstract representation of the scene which spontaneously activates memory representations of scene categories. This descriptor contains perceptual properties (*naturalness*, *openness*, *roughness*, *ruggedness* and *expansion*) which are meaningful to human observers. A further descriptor denoted as *CENTRIST descriptor* [148] is based on the *CENSus TRansform hISTogram* and has been proposed for recognizing topological places or scene categories.

### 1.3.1.2 Color/Intensity

Considering Fig. 1.3, it becomes obvious that *color* or *intensity* is an important visual feature. The respective information represents a perceptual attribute of objects and might therefore be important to distinguish between different objects or to classify objects. Color usually is represented in a color space with 3 channels of intensity information such as RGB or HSV. However, it strongly depends on the illumination of the scene and therefore, a normalized RGB representation and other invariant color representations have been presented in literature as well as using color ratios between pixels and their neighbors. Furthermore, *color histograms* representing the global distribution of colors and *color correlograms* capturing the spatial correlation of colors have been introduced in [128] and [62]. Using *color moments*, other image properties such as the mean value, the variance and the skewness of color distributions may be derived [41].

Simple thresholding techniques for detecting image regions with special color seem to be quite obvious, but they are only suitable if the colors to be detected are not likely to change which indicates that the illumination of the scene may not change significantly. More sophisticated approaches have been proposed for tasks like skin color detection [65, 107], but these techniques can also be adapted for detecting image regions representing the sky or the respective skyline within an image. The rough color structure of an image can also be derived by segmenting an image into different homogeneous color regions, where the mean-shift based approach yields promising results [31].

### 1.3.1.3 Shape

The objects present in a scene can often be identified just by considering their *shape*. Hence, different approaches for adequately describing shape have been proposed in the past. Early investigations on descriptions of shape were based on contours as these represent physical edges caused by abrupt changes in color or texture [7, 50]. Such abrupt changes or discontinuities in image brightness, image color or image texture are likely to correspond to discontinuities with respect to distance or ori-

entation of a surface as well as to different material properties. Furthermore, the region within a closed contour may be considered. Hence, shape description techniques can be categorized into contour-based and region-based approaches. According to [151], contour-based approaches focus on either global properties of a contour such as *perimeter*, *compactness*, *eccentricity*, *shape signature*, *Haussdorff distance*, *Fourier descriptors* or *Wavelet descriptors*, or structural properties such as *chain codes*, *polygonal approximations*, *splines* or *boundary moments*. In contrast to this, region-based approaches take the whole region of a shape into consideration and they may cover global properties such as *area*, *eccentricity*, *geometric moments*, *Zernike moments*, *pseudo-Zernike moments*, *Legendre moments* or *generic Fourier descriptors*, or structural properties such as *convex hull*, *media axis* or *core*. Some of these approaches also address *size* and *orientation* which may be useful for typical applications such as object detection and object recognition.

For obtaining the contours within an image, even local techniques exploiting the convolution with different filter masks have been proposed, for example, the *Roberts operator* [110], the *Prewitt operator* [108], the *Sobel operator* [126], the *Canny operator* [26] or the *Laplacian-of-Gaussian (LoG) filter* [88]. The further importance of angles and sides of a shape has been addressed in [113] and [34]. For detecting special kinds of contours which can be described via mathematical models, the *Hough transform* [60] has been proposed, which originally has been designed for extracting lines. In order to achieve a higher level of generality, the Hough transform has been extended to detect circles and ellipses and, finally, to detect arbitrary shapes [9].

#### 1.3.1.4 Texture

When considering Fig. 1.3, it becomes quite obvious that the homogeneity of certain image regions can also be sufficient to distinguish between different objects or whole image regions with a different semantic content. In [50], this homogeneity has been introduced as *texture*. However, in general, texture is very difficult to define and for this reason, many different definitions of texture can be found in literature [27, 56]. A general but very suitable definition for a variety of applications defines texture as a function of the spatial variation of intensity values within an image [27]. This encapsulates a contextual property involving intensity values within a spatial neighborhood as well as the spatial distribution of these intensity values and different scales or levels of resolution. According to [72], texture can be described with respect to *uniformity*, *density*, *coarseness*, *roughness*, *regularity*, *linearity*, *directionality*, *direction*, *frequency* and *phase*. However, not all of these attributes are independent. In general, the approaches for describing texture can be categorized into three different groups:

- *Statistical approaches* describe texture via the statistical distribution of intensity values. They include *Fourier power spectra*, *Tamura features* [131] including *coarseness*, *contrast*, *directionality*, *linelikeness*, *regularity* and *roughness*, *co-occurrence matrices* or measures derived from the co-occurrence matrix such as *energy*, *entropy*, *contrast*, *homogeneity* and *correlation*.

- *Structural approaches* define texture as composed of *texture primitives* or *textural elements (texels)* occurring repeatedly with respect to certain placement rules [138, 152]. The structural properties of these elements are commonly described via *average intensity*, *area*, *perimeter*, *eccentricity*, *orientation*, *compactness*, *moments*, etc. [152].
- *Model-based approaches* focus on modeling an image as a probability model or as a set of basis functions [152]. The coefficients of the utilized models are assumed to adequately characterize the image. Such approaches are often based on *simultaneous autoregressive (SAR) models*, *Wold-like models* describing properties such as *periodicity*, *directionality* or *randomness*, *Markov models* or multi-resolution filtering techniques such as *Gabor transform* and *wavelet transform* [152].

### 1.3.1.5 Local Features

Features arising from small parts of an image are referred to as *local features*. Using such local features has been investigated since the 1950s, as highly informative image points are located at corners or blobs in an image [7]. Different methods for detecting *convex blobs* [111], *points of inflection* [49] representing zero-crossings of the curvature which separate convex and concave parts of a curve, *maxima or minima of curvature* [113] and *dominant points* [112] have been proposed. Further local features may be derived from *junctions* or *intersection of edges*. As only a small region is involved and not the whole edge, local *edge elements (edgels)* can also be defined as local features. Considering the general structure of an image [69], the approaches can be categorized based on the distribution of intensity values, the image derivatives, the spatial frequency, etc. [136].

An analysis of early approaches to corner detection in intensity images has been presented in [154], where a large number of approaches representing template-based corner detection and geometry-based corner detection with respect to edges, topology and autocorrelation have been analyzed. Then, within a few years, other comparisons including existing methods, improved methods and new methods followed [92–97, 118]. Many of these approaches are based on the *structure tensor* or on the *Hessian matrix*. However, the improvements not only addressed feature extraction, but also feature description in order to achieve an increased robustness when comparing images. A performance evaluation of various local feature descriptors has been carried out in [94] and [96]. A valuable survey over different local feature detectors and feature descriptors is provided in [136].

### 1.3.1.6 Application-Specific Features

Some visual features strongly depend on the respective application. If, for instance, only human faces are to be found within an image, it might be sufficient to shift a small face patch taken from a database over the image and compare it to the respective image content. Then the face patch itself becomes a visual feature. Furthermore,

characteristic parts of a human face such as eyes, eyebrows, nose or mouth can also become visual features. Such features are however only important for face detection, and not for other applications. Different approaches for detecting faces are presented in [149]. A further example is the analysis of building façades, where a symmetric arrangement of simple primitives representing windows and doors can be expected [37].

### ***1.3.2 Spatio-Temporal Features***

Whereas spatial features cover many characteristics of an image, there are still some further types of visual features which only appear when analyzing image sequences, that is, these features also address the temporal domain. As a consequence, appropriate spatio-temporal features are required. Hence, different types of such spatio-temporal features are presented in the following sections.

#### **1.3.2.1 Motion**

When considering an image sequence, a further feature type might arise from moving objects or persons. Sub-categories describing motion could then be defined, for example, direction, velocity, acceleration, the respective trajectory or the time span within which an object or person is visible in the image sequence.

#### **1.3.2.2 Events**

It also becomes apparent that special events themselves represent visual features. For instance, in the image sequence depicted in Fig. 1.3, it becomes visible that an event can be located within an image sequence. Other events which might be of importance are shot-boundaries in videos as these usually separate different scenes.

#### **1.3.2.3 Activities**

Depending on the scene, activities of objects or persons may be visually perceptible. However, activities might be difficult to detect as they include interpretation as well as context information and thus, activities represent high-level features.

### ***1.3.3 Is There a Preferred Type of Visual Features?***

A key desire for many applications would be a type of visual features with almost general applicability. Considering Fig. 1.3, the global information of the whole image leads to the interpretation that the building is in a very bad condition and a demolition by controlled blasting can be expected to follow within the next time.

This is a high-level feature which cannot properly be captured with other feature types. Color or texture information allow for separating special image regions containing sky, building and ground. The shape information is concentrated in the area of the building as well as most of the local features.

For technical and time-critical applications such as object recognition or navigation, the presence of clutter and occlusions represents a major problem to global features, and methods for extracting and comparing color regions, texture information or shape information still remain computationally challenging. In contrast, local features can be extracted very efficiently, and, due to their accurate localization, their stability over time and the possibility of individual identification, they are suited for a wide range of applications such as object recognition, autonomous navigation and exploration, image and video retrieval, image registration or the reconstruction, interpretation and understanding of 3D scenes. Hence, these features tend to provide a well-suited feature type for typical computer vision applications [136]. For this reason, the following section addresses their evolution from early approaches to modern computer vision.

## 1.4 The Evolution of Local Features

According to [136], a *local feature* is an image pattern varying from its spatial neighborhood. The variation typically arises from a change of one or more image properties such as color, intensity or texture. In general, local features can be corners, blobs, edgels or small image patches. If only the location of the feature is of interest, the term *interest point* is commonly used. For most applications, however, a description of each feature is derived from local image properties in order to identify and match similar features. When localizing or describing features, a local neighborhood has to be analyzed, and thus not only the location of a feature, but also the size and shape of the neighborhood have to be determined. In this section, the basic ideas with the most significant impact on current research are presented. These ideas address both an adequate extraction of the respective image locations, that is, *feature detection* (Sect. 1.4.1), and an adequate representation of the local image characteristics, that is, *feature description* (Sect. 1.4.2). In general, the impact of an approach depends on its performance with respect to common evaluation criteria in reasonable time. Such criteria are considered in Sect. 1.4.3. Finally, in Sect. 1.4.4, the focus of current and future research trends is described which addresses feature detection and feature description (even beyond images and image sequences) as well as challenges arising from more complex scenarios with respect to reflectance behavior.

### 1.4.1 Feature Detection

Different strategies for feature extraction have been proposed in literature which aim at finding the respective image locations. In general, the definition of a local feature postulates the following characteristics [57, 136]:

- *Distinctiveness/Informativeness*: Local features should significantly differ from their spatial neighborhood within the image and thus be highly distinctive.
- *Repeatability (Invariance/Robustness)*: A method for extracting local features should be able to produce similar features for similar images. A high degree of reproducibility often is correlated with invariance and robustness. In general, the extraction of local features should be unaffected by simple mathematical transformations of the image, that is, invariant to image rotation and image scaling, and it should be tolerant to image noise, changes in illumination and small deformations arising from minor changes in viewing direction.
- *Locality*: Local features should only depend on a small spatial neighborhood within the image.
- *Quantity*: The total number of local features being extracted from an image should be sufficiently high.
- *Accuracy*: The detected local features should be localized accurately. This includes the spatial location within the image as well as other aspects such as scale and possibly shape.
- *Efficiency*: The extraction of local features should be relatively easy and thus be suited for time-critical applications such as object recognition or navigation.

#### 1.4.1.1 Moravec Corner Detector

The *Moravec corner detector* presented in [99] is based on considering local image patches and determining the average change of intensity resulting from small shifts of the respective patch into various directions  $(u, v)$ . This average change is measured via the sum of squared differences (SSD) between the two respective patches. Denoting the image intensities with  $I(x, y)$  and introducing a window function  $w(x, y)$ , the change  $E_{uv}$  resulting from a shift  $(u, v)$  is defined as

$$E_{uv} = \sum_{(x,y) \in \mathcal{A}} w(x, y) [I(x+u, y+v) - I(x, y)]^2, \quad (1.1)$$

where  $\mathcal{A}$  describes the considered spatial neighborhood. If, for all possible shifts  $(u, v) \in \mathcal{B}$ , the minimum change and thus the measure

$$c_{\text{Moravec}} = \min\{E_{uv} | (u, v) \in \mathcal{B}\} \quad (1.2)$$

is above a given threshold, a *corner* is detected. In literature, this measure is also referred to as *cornerness* or *corner strength*. For pixels within homogeneous regions, nearby image patches look very similar and thus, the cornerness becomes small. For pixels on edges, nearby image patches in direction parallel to the edge are similar, whereas nearby image patches in direction perpendicular to the edge differ significantly, which together yields a small value for the cornerness. Only for pixels showing varying intensities in all directions, nearby image patches will significantly differ and the cornerness reaches higher values. However, the Moravec corner detector is limited on a small number of possible and discrete shifts in discrete directions which, for instance, leads to problems if edges are not oriented along these shifts.

### 1.4.1.2 Harris Corner Detector and Modified Variants

Overcoming the limitations of the Moravec corner detector, the *Harris corner detector* has been presented in [58] as an improved corner detector directly considering the derivatives of the image  $I(x, y)$  instead of shifted patches. Assuming only small shifts  $(u, v)$ , the term  $I(x+u, y+v)$  can be approximated by a Taylor series which ignores the higher order terms. This finally leads to the expression

$$E_{uv} = [u, v] \mathbf{S} [u, v]^T \quad (1.3)$$

with

$$\mathbf{S} = \sum_{(x,y) \in \mathcal{A}} w(x, y) \begin{bmatrix} I_x^2(x, y) & I_x(x, y)I_y(x, y) \\ I_x(x, y)I_y(x, y) & I_y^2(x, y) \end{bmatrix}, \quad (1.4)$$

where  $I_x(x, y)$  and  $I_y(x, y)$  denote the partial derivatives of the image  $I(x, y)$  in  $x$ - and  $y$ -direction. The function  $w(x, y)$  is a window function, and the second moment matrix  $\mathbf{S}$  is also referred to as *autocorrelation matrix* or *structure tensor* as it describes the gradient distribution in the local neighborhood and thus local image structures. The eigenvalues  $\lambda_1$  and  $\lambda_2$  of  $\mathbf{S}$  are proportional to the principal curvatures of the local autocorrelation function and form a rotationally invariant description of  $\mathbf{S}$  [58]:

- If both eigenvalues  $\lambda_1$  and  $\lambda_2$  are small, the local autocorrelation function is flat. Thus, shifts  $(u, v)$  only cause little change in  $E_{uv}$ . This indicates a local neighborhood of approximately constant intensity.
- If one eigenvalue is high and the other one is low, the local autocorrelation function is ridge shaped. Thus, only shifts along the ridge (i.e., along the edge) cause little change in  $E_{uv}$ . This indicates an edge.
- If both eigenvalues  $\lambda_1$  and  $\lambda_2$  are high, the local autocorrelation function is sharply peaked. Thus, shifts in any direction will increase  $E_{uv}$ . This indicates a corner.

The direct use of the eigenvalues has been proposed in [124] with

$$c_{\text{ShiTomasi}} = \min\{\lambda_1, \lambda_2\} \quad (1.5)$$

as definition of the cornerness. This definition tends to be robust, but it only determines whether a pixel corresponds to a corner or not. As using the terms  $\text{trace}(\mathbf{S})$  and  $\det(\mathbf{S})$  is computationally less expensive than explicitly calculating the eigenvalues  $\lambda_1$  and  $\lambda_2$  of  $\mathbf{S}$ , the cornerness has been defined as

$$c_{\text{HarrisStephens}} = \lambda_1 \lambda_2 - \kappa (\lambda_1 + \lambda_2)^2 = \det(\mathbf{S}) - \kappa \text{trace}^2(\mathbf{S}) \quad (1.6)$$

where  $\kappa$  is a tunable sensitivity parameter typically chosen between 0.04 and 0.06 [58, 68]. As a consequence, it is possible to distinguish between edges and corners:

- If both eigenvalues  $\lambda_1$  and  $\lambda_2$  are small, both terms are relatively small. Thus, the cornerness reaches a small absolute value which indicates a local neighborhood of approximately constant intensity.

- If one eigenvalue is high and the other one is low, the second term is higher than the first one. Thus, the cornerness is negative which indicates an edge.
- If both eigenvalues  $\lambda_1$  and  $\lambda_2$  are high, the first term is much greater than the second one. Thus, the cornerness is high which indicates a corner.

The use of a constant parameter which has to be set empirically, however, is a drawback. Hence, a further definition of the cornerness has been presented in [104] with

$$c_{\text{Noble}} = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2 + \varepsilon} = \frac{\det(\mathbf{S})}{\text{trace}(\mathbf{S}) + \varepsilon} \quad (1.7)$$

where  $\varepsilon$  is a very small, but constant value. This value has to be introduced in order to avoid a singular denominator which might occur in case of a rank zero matrix. The values of the cornerness are non-negative and only high values indicate a corner.

As the Harris corner detector is not invariant to changes in image scale, a multi-scale representation has been introduced in [92] in order to reach a scale-invariant corner detector. Using a scale-space has already been introduced in 1983 as this yields a concise but complete qualitative description of an image over all scales of observation [147]. Investigations presented in [69] and [79] showed that the Gaussian function is the most suitable scale-space kernel. If a Gaussian function is utilized, the scale-space of the image is defined as a function  $L(x, y, \sigma)$  with

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (1.8)$$

which is derived by convolving the original image  $I(x, y)$  with Gaussians  $G$  of variable scale  $\sigma$ . The resulting *multi-scale Harris detector* can be described via the multi-scale second order matrix which includes a parameter  $\sigma_D$  for the scale of Gaussian kernels (*differentiation scale*) and a parameter  $\sigma_I$  for the scale of a Gaussian window used for averaging the derivatives in the local neighborhood (*integration scale*). Thus, the cornerness can again be determined via the measures described above. However, it might not be optimal to select the scale parameters empirically. Hence, it has been proposed in [92] to combine the multi-scale Harris detector with an automatic scale selection involving the scale-normalized Laplacian operator [80] which finally yields the *Harris–Laplace detector*.

The Harris–Laplace detector can further be modified by introducing an iterative procedure proposed in [79] which modifies the location and the scale as well as the shape of the local neighborhood of each detected corner. This yields the *Harris–Affine detector* [93] which allows for an affine invariant detection of corners. An improved Harris corner selection strategy involving the  $z$ -score function has recently been published [16] which increases the repeatability as well as the matching score and avoids the use of an empirically determined parameter.

#### 1.4.1.3 Förstner Detector and SFOP

A further method for extracting local features has been introduced with the *Förstner detector* in [47]. This approach is also based on local statistics of the image function

encapsulated in the structure tensor. It uses the trace of this tensor for measuring the energy of the image function at the respective image point, the ratio of the eigenvalues as a degree of orientation or of an isotropy, and the largest eigenvalue as estimate for the local gradient [46]. Initially, the existence and the approximate location of features are analyzed. Subsequently, the detected features are precisely localized. As this involves further statistics, the algorithm shows a higher computational effort when compared to other detectors. The original Förstner detector has been extended in [46] by combining it with a spiral feature model proposed in [19] for detecting junctions and spiral features. A further extension involving a scale-space has been proposed with the *Scale-invariant Feature OPerator (SFOP)* [48]. In comparison to other feature detectors, the SFOP can be used to detect features with complementary properties over scale-space, even in poorly textured areas, and the high localization accuracy is very suitable for tasks such as camera calibration.

#### 1.4.1.4 Hessian Detector

Characteristic features can also be extracted by exploiting the second order derivatives. The *Hessian detector* presented in [15] is based on the Hessian matrix

$$\mathbf{H}(x, y) = \begin{bmatrix} I_{xx}(x, y) & I_{xy}(x, y) \\ I_{xy}(x, y) & I_{yy}(x, y) \end{bmatrix} \quad (1.9)$$

where  $I_{xx}(x, y)$ ,  $I_{xy}(x, y)$  and  $I_{yy}(x, y)$  are the second order image derivatives which locally describe how the normal to an isosurface changes [136]. The determinant of this matrix is also known as *discriminant*, and the trace is also referred to as *Laplacian*. These provide two measures for detecting blob-like structures in an image, as blobs are detected where the respective measure reaches a local maximum above a predefined threshold. The determinant of the Hessian matrix is related to the product of the principal curvatures which represents the Gaussian curvature.

The Hessian detector is only invariant to image rotation and for this reason, a multi-scale representation has been introduced in [35] in order to obtain a scale invariant detector. Using the automatic scale selection based on the Laplacian operator which has been presented in [80] yields the *Hessian–Laplace detector* [92]. Alternatively, a scale could be selected for which both the trace and the determinant of the Hessian matrix assume a local extremum [91]. In the same way the Harris–Laplace detector has been extended to the Harris–Affine detector by including an affine adaptation process to obtain invariance to affine image transformations, the Hessian–Laplace detector can be modified to the *Hessian–Affine detector* [97].

#### 1.4.1.5 DoG Detector/SIFT Detector

The *Scale Invariant Feature Transform (SIFT)* introduced in [83] and later improved in [84] can be utilized for detecting distinctive keypoints in an image and extracting appropriate local feature descriptors. The detection of keypoints is based on an efficient approximation of the Laplacian function and carried out in two steps:

- *Scale-space extrema detection:* At first, the image  $I$  is convolved with Gaussian kernels of variable scale, reduced in size and again convolved with Gaussian kernels of variable scale in order to build the Gaussian scale-space. The blurred images are grouped according to their size and sorted with respect to their scale. Subtracting neighboring images yields the scale-space of Gaussian differences also known as Difference-of-Gaussian (DoG) pyramid. Stable keypoint locations can then be determined by comparing each sample point to its eight neighbors at the same scale and its nine neighbors at both neighboring scales. Resulting from this, local extrema are selected as keypoint candidates.
- *Keypoint localization:* Once keypoint candidates have been found, the next step is to improve their location to subpixel accuracy by fitting a 3D quadratic function to the local sample points in order to determine the interpolated location of the extremum. Furthermore, keypoint candidates with a low stability are removed, that is, points with low contrast which are sensitive to noise or points located along edges which can hardly be distinguished from each other.

#### 1.4.1.6 SURF Detector

A further approach for detecting distinctive features is directly based on a scale-space representation of the Hessian matrix. It involves an approximation of the Hessian matrix via box filters [13, 14]. These filters introduce an equal weight of +1 or -1 for the contribution of each pixel within a rectangular area and allow for using integral images which can be computed efficiently. Once such an integral image is available, only three additions are necessary to obtain the sum of all intensities within any upright rectangular area, independent of its size. Distinctive features in an image can be detected at locations where the determinant of the approximated Hessian matrix reaches a maximum. Subsequently, similar to the extraction of SIFT features, the location of the resulting feature points is improved to subpixel accuracy and a non-maximum suppression is carried out by considering  $3 \times 3 \times 3$  neighborhoods. The resulting features are denoted as *Speeded-Up Robust Features (SURF)*.

#### 1.4.1.7 CenSurE Detector

The Harris–Laplace detector [92] and the Hessian–Laplace detector [92] are based on calculating the respective cornerness measure at all locations and all scales, and, additionally, the Laplacian at all scales where peaks of the cornerness have been detected [2]. As the Laplacian is supposed to be very suitable, the *Center Surround Extremas (CenSurE) detector* presented in [2] uses efficient bi-level approximations of the Laplacian (LoG filter) and thus reaches real-time potential.

At all locations and all scales, center-surround bi-level filters are calculated which multiply the respective image pixel by either +1 or -1. Local extrema then indicate potential interest point candidates. Except for the circular symmetric filter, these filters can efficiently be calculated via box filters and thus integral images.

In order to improve the approximation by using polygonal filters, modified versions of integral images covering trapezoidal areas have been proposed in [2], which are derived by combining different slanted integral images. Finally, a non-maximum suppression is performed over the scale space by considering local  $3 \times 3 \times 3$  neighborhoods of potential interest point candidates. As the magnitude of the response indicates the strength of a feature and weak responses are less stable, the performance can further be increased by simple thresholding with respect to the filter response, and applying the scale-adapted Harris measure is proposed in order to suppress responses along edges or lines.

A special filter can be derived by utilizing two of the boxes if one is rotated by an angle of  $45^\circ$ . This yields the star detector which is also known as *Star Keypoint detector*. However, as the CenSurE detector does not exploit the sparseness of the filter responses, significant improvements have been proposed with the *Speeded-Up Surround Extremas (SUSurE) detector* in [38].

#### 1.4.1.8 SUSAN Detector

Instead of considering image gradients being sensitive to noise and computationally more expensive, the *Smallest Univalue Segment Assimilating Nucleus (SUSAN) detector* [125] relies on comparing the intensity of each image pixel to the intensities of all pixels within a small circular neighborhood. This comparison yields two sets of pixels, one consisting of those pixels with intensities similar to the intensity of the center pixel and one consisting of pixels with different intensities. Hence, according to [125], the center pixel is referred to as *nucleus*, and all pixels with similar intensities define a local area denoted as *Univalue Segment Assimilating Nucleus (USAN)*. This USAN contains much information about the local structure of the image. Its size is at maximum for a center pixel within homogeneous regions where the USAN covers almost the entire neighborhood. Near edges, the ratio of similar pixels to all pixels within the circular neighborhood decreases to approximately 50 % and near corners even further to approximately 25 % [136]. Thus, interest points can be detected at locations in the image where the size of the USAN reaches a local minimum. At these locations, potential candidates with Smallest Univalue Segment Assimilating Nucleus (SUSAN) are detected. However, these have to be filtered by applying several constraints in order to achieve an increased robustness. This involves a limitation of the maximum size of the USAN as well as a consideration of the distance between centroid and center pixel. If, for instance, the centroid of the USAN is not located close to the center pixel, and all pixels on the line between the center pixel and the centroid and even further to the end of the circular neighborhood are within the USAN, this indicates the presence of a corner more reliably. An additional non-maximum suppression finally yields interest points with a SUSAN.

#### 1.4.1.9 FAST Detector

Based on only a special part of local image neighborhoods, the *Features from Accelerated Segment Test (FAST) detector* has been introduced in [114] which still nowa-

days is one of the fastest feature detectors. This feature detector is closely related to the SUSAN detector [125] which, for each pixel  $(x, y)$  of the image, considers the fraction of pixels within a local neighborhood being similar to the respective center pixel  $(x, y)$ . Instead of all pixels within a local neighborhood, the FAST detector only uses the pixels on a Bresenham circle [23] of radius  $r = 3$  surrounding the center pixel and thus only 16 pixels. Hence, for each pixel  $(x, y)$  of the image, a test is performed by comparing the intensity values corresponding to the pixels on the surrounding Bresenham circle to its intensity value  $I(x, y)$ . Introducing a threshold  $t$ , the investigated pixel is assumed to be a potential candidate interest point if the intensities of at least  $n = 12$  contiguous pixels are above  $I(x, y) + t$  or below  $I(x, y) - t$ . This test can be optimized by first comparing the pixels in opposite horizontal and vertical direction to the center pixel and checking whether three of the selected pixels satisfy the constraint. Finally, a non-maximum suppression may be carried out in order to avoid adjacent responses. Additionally, *positive features* where the circular pixels are brighter than the center pixel and *negative features* where the center pixel is brighter than the circular pixels can be distinguished. This allows for a more efficient matching process, as positive features do not necessarily need to be compared to negative ones [114].

A modification with enhanced repeatability (ER) and only little loss of efficiency has been proposed with the *FAST-ER detector* in [115] which represents a generalization of the FAST detector involving concepts of machine learning. As the FAST detector is not invariant to changes in image scale, an extension to a multi-scale detector involving a scale selection based on the Laplacian function has been proposed in [74]. Extensions considering scaling and rotation have been proposed with the detector of the *Oriented FAST and Rotated BRIEF (ORB) operator* [116] and the *Binary Robust Invariant Scalable Keypoint (BRISK) detector* [75].

#### 1.4.1.10 Detection of MSERs

A method for detecting distinctive image regions has been presented with *Maximally Stable Extremal Regions (MSERs)* in [89]. Initially, the image pixels are sorted by intensity and placed in the image either in increasing or in decreasing order. The list of connected components and the respective areas is maintained using the union-find algorithm [121]. The whole process yields a data structure where the area of each connected component is stored as a function of intensity. MSERs then correspond to thresholds for which a function describing the relative area change reaches an extremal value. Hence, an MSER is a connected component of an appropriately thresholded image [136], and it is either darker than its surrounding area (dark extremal regions) or brighter (bright extremal regions). For being able to derive point correspondences, the MSERs are assigned the respective centroid and (maybe) the respective threshold value. A further optimized implementation for extracting MSERs has been introduced in [102] which replaces the union-find algorithm with a more efficient strategy and thus allows for faster calculation with less requirements concerning memory. The extension of MSERs to multi-resolution

involving a scale pyramid has been proposed as *Multi-Resolution MSERs* in [45], and an extension of MSERs to color images has been presented in [44].

### 1.4.2 Feature Description

Once a local feature has been detected, it has proven to be feasible to assign an adequate description in order to allow for reliably finding the feature again in different images of the same scene. Hence, in the following, different techniques are considered with respect to the main ideas.

#### 1.4.2.1 Early Approaches

The early feature detectors had no specific descriptors and only relied on the comparison of intensity values in the local neighborhoods. Hence, an interest point is described via a small image patch covering its local neighborhood. For comparing the small image patches, a *template matching* can be carried out between small image patches around the respective interest points. The template matching is typically based on normalized cross-correlation, that is, when comparing the local neighborhood of a feature in one image to the local neighborhoods of the features of the other image, a resulting extremum indicates the most similar feature point in the other image. Consequently, feature correspondences can be derived, and their quality may be measured with respect to the similarity of the respective neighborhoods. Instead of using the intensity values within the local neighborhoods, it is also possible to use different information for establishing feature correspondences, for example, information derived from the respective first or second order derivatives, or the information derived from the *monogenic signal* [39, 40, 130]. In contrast, more modern descriptors rely on a vectorized representation which, for instance, allows for determining the most similar descriptor via Euclidean distances.

#### 1.4.2.2 SIFT Descriptor

The detection of similar image patches is challenging if these provide the respective information at a different scale and maybe even with a different orientation. Hence, in addition to detecting distinctive keypoints in an image, the SIFT algorithm [83, 84] also extracts local feature descriptors which are invariant to image scaling and image rotation, and robust with respect to image noise, changes in illumination and small changes in viewpoint. With these descriptors, it is possible to locate correspondences between different images and, finally, to derive common image contents. The *SIFT descriptor* is calculated in two steps:

- *Orientation assignment*: Based on the local image gradient directions of the closest Gaussian smoothed image, each keypoint is assigned an orientation histogram. Each sample added to this histogram is weighted by its gradient magnitude and a Gaussian window centered on the keypoint. Dominant directions of the local

gradients lead to peaks in the orientation histogram. The highest peak detected describes the orientation of a keypoint, and for all peaks within 80 % of this maximum value, an additional keypoint with the corresponding orientation is generated. Finally, an interpolation of the dominant peaks in the orientation histogram is carried out in order to reach a more accurate orientation assignment.

- *Generation of keypoint descriptors:* The descriptor of each remaining keypoint is derived from the image gradients of the closest Gaussian smoothed image in the local neighborhood of the keypoint. In order to achieve rotational invariance, this gradient information is rotated and aligned with the assigned keypoint orientation. Then, the local neighborhood of each keypoint is divided into  $4 \times 4$  subregions and, for each of these subregions, an orientation histogram with eight entries is created. Again, the contribution of each pixel in the neighborhood of a keypoint to the corresponding orientation histogram is weighted by the gradient magnitude and by a Gaussian window centered on the keypoint. This yields a local image descriptor with 128 elements. Finally, the feature descriptors are normalized in order to reduce the effects of changes in illumination [84].

For two images of a scene, corresponding SIFT features can then be detected by comparing the Euclidean distances of a keypoint descriptor in the first image to the nearest neighbor and to the second-nearest neighbor in the second image. The ratio of these distances has to be below a given threshold  $t$ . This ratio test significantly reduces ambiguities in the matching process.

The SIFT algorithm is probably one of the most powerful algorithms for feature extraction. However, it shows a relatively high computational effort which is the reason for a large amount of investigations based on the general idea of SIFT. For instance, to further improve SIFT, modified variants like *Fast approximated SIFT* [52], *Rotation-Invariant Feature Transform (RIFT)* [73], *Gradient Location-Orientation Histogram (GLOH)* [96], *Coloured SIFT (CSIFT)* [1], *Affine-SIFT (ASIFT)* [100], *PCA-SIFT* [67], *Salient-SIFT* [77] or *Kernel Projection Based SIFT (KPB-SIFT)* [153] have been proposed. SIFT also inspired establishing the *Mirror Reflection Invariant Feature (MIFT)* descriptor preserving robustness to changes in image scale, image rotation, affine transformations and mirror reflections [54].

#### 1.4.2.3 HOG Descriptor

A feature descriptor similar to the SIFT descriptor has been presented in [33]. It is based on the distribution of the gradient orientations within a local neighborhood of an interest point from which the *Histogram of Oriented Gradient (HOG) descriptors* are derived. In the original implementation, the image has been divided into small spatial regions referred to as *cells*. For all pixels within a cell, the gradient orientations are calculated and quantized into a fixed number of angular bins representing an orientation histogram. Concatenating the histogram entries and normalizing the resulting vector finally yields HOG descriptors.

In [81], the HOG descriptors are extracted along the normal direction and along the tangential direction of edges by using slices of six cells, each consisting of ori-

ented gradients quantized into 12 angular bins. The resulting features are then defined as *edge-slice* and *edge-ribbon*. The HOG descriptor has also been combined with the Bag of Visual Words representation which yields the *Pyramid Histogram of Words (PHOW) descriptor*, and it has been extended to using different resolution levels which yields the *Pyramid Histogram of Oriented Gradients (PHOG) descriptor*, to also involving soft connecting edges which yields the *Histogram of Canny Oriented Gradients (CHOG) descriptor*, or to achieving a more dense HOG-based feature type in form of *PixelHOG (PiHOG)* [30].

#### 1.4.2.4 SURF Descriptor

The *SURF descriptor* [13, 14] is aligned to the dominant orientation of an interest point. This dominant orientation is calculated via Haar wavelet responses in  $x$ - and  $y$ -direction within a circular neighborhood. The respective wavelet responses are weighted by a Gaussian window centered on the keypoint and represented as points in a 2D space derived from the response strengths in  $x$ - and  $y$ -direction. All points within a sliding orientation window of  $60^\circ$  are summarized which yields a local orientation vector, and the orientation for which this vector reaches the maximum length is defined as dominant orientation.

In order to achieve rotational invariance, the squared region aligned with the assigned keypoint orientation is considered and divided into  $4 \times 4$  subregions. For each of these subregions, the Haar wavelet responses at  $5 \times 5$  regularly spaced sample points are calculated with respect to the local  $x$ - and  $y$ -direction within the subregion. The contribution of each wavelet response is weighted by a Gaussian window centered at the keypoint. For each subregion, the sums over the weighted wavelet responses in  $x$ - and  $y$ -direction as well as the sums over the weighted absolute wavelet responses in  $x$ - and  $y$ -direction are computed. This yields four characteristic values. The concatenation of these values for all of the  $4 \times 4$  subregions yields a vector with 64 elements. Finally, a normalization is carried out in order to increase the robustness with respect to changes in illumination.

A faster method for calculating a descriptor for images where rotations are not likely to occur has also been proposed with the *Upright-SURF (U-SURF) descriptor* which does not rely on an assigned dominant orientation [12–14].

#### 1.4.2.5 CenSurE Descriptor

A further modification of the U-SURF descriptor has been presented in [2] as *Modified Upright-SURF (MU-SURF) descriptor*. In order to avoid boundary effects in which the descriptor changes abruptly, paddings are introduced to each boundary in the descriptor increasing the whole region size. These paddings lead to smoothing effects in the descriptor as the extended subregions overlap. Similar to the extension of the CenSurE detector yielding the SUSurE detector, the MU-SURF or *CenSurE descriptor* has been extended by exploiting sparse sampling for increased efficiency which leads to the *SUSurE descriptor* [38].

#### 1.4.2.6 DAISY Descriptor

Being inspired from earlier descriptors such as SIFT and GLOH, a local descriptor denoted as *DAISY descriptor* has been presented in [134]. This descriptor provides the same properties as the histogram-based descriptors concerning changes in image scale, image quality, illumination and viewpoint. However, the SIFT and GLOH descriptors have been designed for sparse wide-baseline matching, as they are computationally demanding. Their suitability arises from orientation histograms based on gradient information within the local neighborhood of an interest point. Using integral images, the SURF descriptor is computationally more efficient, but the missing spatial weighting results in equally contributing gradients. This causes artifacts when densely computing the descriptor for all image pixels. In contrast, the DAISY descriptor provides both a robustness comparable to the SIFT descriptor and a fast calculation. For this reason, it has even been proposed for dense depth map estimation from wide-baseline image pairs [134, 135].

For establishing the DAISY descriptor, an orientation map  $G_o$  is derived for each quantized direction  $o$ . For this, usually eight directions with an equal angular spacing are used. Subsequently, each orientation map is convolved with several Gaussian kernels  $G_\sigma$  with different  $\sigma$  values. This yields convolved orientation maps  $G_o^\sigma$ . For an interest point located at  $(x, y)$ , the respective DAISY descriptor consists of a vector with entries being derived from these convolved orientation maps located at positions  $(u, v)$  on concentric circles around  $(x, y)$ , which is the reason for the name of the descriptor. A normalization is then carried out for each histogram independently. Finally, the full DAISY descriptor  $\mathcal{D}(x, y)$  for an interest point at location  $(x, y)$  is defined as a concatenation of the respective normalized vectors.

DAISY descriptors share histograms so that a histogram computed for the descriptor belonging to one image pixel does not need to be computed for the descriptors of the surrounding pixels again. This fact in combination with using separable convolutions is the main reason for the efficiency of DAISY. An efficient memory access and the early separation of the histogram layers further improve efficiency. Thus, the computation of the DAISY descriptor allows for a parallelized implementation. According to [135], the current implementation can even further be optimized, and a real-time or even faster implementation for computing the DAISY descriptors for all pixels of an image might be possible. A fast and dense computation of robust image descriptors, however, may also be very useful for applications beyond stereo reconstruction such as object or material recognition.

#### 1.4.2.7 Descriptors for MSERs

A method for describing MSERs has been introduced in [29] and relies on local affine frames derived from triplets of points affine-invariantly selected with respect to contour and centroid of each MSER. Affine invariant shape descriptors for MSERs have been introduced in [45] and using the SIFT descriptor for MSERs has been proposed in [101].

### 1.4.2.8 Binary Representations

Binary strings have the advantage that they can be compared using the Hamming distance [55] which is much more efficient than using the Euclidean distance. Hence, binary feature descriptors have also been investigated. Originally proposed for texture description, the idea of using *Local Binary Patterns (LBPs)* has been transferred in order to describe local regions [59]. Thus, LBPs are derived by thresholding the neighborhood of a feature and considering the result as a binary pattern. For reducing the number of different binary patterns, comparing center-symmetric pairs of pixels has been proposed as *Center-Symmetric Local Binary Patterns (CS-LBPs)* [59]. The extension to multi-resolution levels of local image regions has been addressed with *Multi-Resolution Local Binary Patterns (MR-LBPs)* in [78].

Exploiting the fact that the relative order of pixel intensities remains unchanged in case of monotonic intensity changes, the *Local Intensity Order Pattern (LIOP) descriptor* has been proposed in [142]. After using the overall intensity order to divide the local patch into subregions, a LIOP of each point is derived which represents a binary vector and encodes the relative relationships among the intensities of its neighboring sample points. Accumulating the binary LIOPs in each ordinal bin and the respective concatenation for all subregions finally yields the real-valued LIOP descriptor. This descriptor tends to be highly discriminative as both local and overall intensity order information of the local patch are encoded.

Recently, with *Binary Robust Independent Elementary Features (BRIEF)*, a further descriptor has been presented in [25] which is reported to be highly discriminative and very efficient to compute. This descriptor relies on a number of difference tests with respect to intensity values. However, it is not invariant to changes in scale or rotation which has to be addressed to compete with other current descriptors such as SIFT, SURF or DAISY. The most recent advancements address this issue while maintaining a fast, compact and robust descriptor. For instance, the descriptor calculated with the *Oriented FAST and Rotated BRIEF (ORB) operator* [116] uses oriented BRIEF descriptors, and the *Binary Robust Invariant Scalable Keypoint (BRISK) descriptor* [75] is based on a more qualitative variant of difference tests on a circular sampling pattern similar to the DAISY pattern [134, 135], and it also considers orientation and scale. The *Compact and Real-time Descriptor (CARD)* [6] exploits lookup tables to extract histograms of oriented gradients from arbitrary layouts of bins and applies learning-based sparse hashing for converting these histograms into short binary codes. Being inspired by the human visual system, a further descriptor denoted as *Fast Retina Keypoint (FREAK) descriptor* [5] is based on computing a cascade of binary strings for comparing image intensities over a retinal sampling pattern.

### 1.4.3 Evaluation of Feature Detectors and Feature Descriptors

Depending on the application or the used datasets, the performance of feature detectors and feature descriptors may vary significantly as well as the requirements

**Table 1.1** Qualitative comparison of popular feature detectors

Feature Detector	Type	Invariance	Time Efficiency
Harris	corner	rotation	++
DoG / SIFT	blob (corner)	rotation, scale	++
SURF	blob (corner)	rotation, scale	+++
FAST	corner	rotation	+++
MSER	region	rotation, scale, affine	+++
ORB	corner	rotation, scale	+++
BRISK	corner	rotation, scale	+++

**Table 1.2** Qualitative comparison of popular feature descriptors

Feature Descriptor	Size	Descriptor Entries	Time Efficiency
SIFT	128	real-valued	+
DAISY	200	real-valued	++
SURF	64	real-valued	++
LIOP	144	real-valued	+
BRIEF	32	binary	+++
ORB	32	binary	+++
BRISK	64	binary	+++

which have to be satisfied. Hence, a certain trade-off has to be found which may, for instance, consider the persistence of features and feature correspondences across a large variety of image transformations involving image rotation, scale change, viewpoint change, illumination change, noise, blur or compression [118]. Furthermore, the trade-off may address different characteristics with respect to computational effort, data storage, geometric accuracy and applicability to different image contents. State-of-the-art measures for evaluation are the *repeatability* measuring the persistence of features in terms of the overlap of the detected feature regions, and the *matching score* measuring the ability to correctly assign corresponding features which also involves the similarity of the respective feature descriptors [96, 118]. Further commonly used measures are the *number of detected correspondences*, the *number of correct correspondences* or *precision-recall characteristics*. Additionally, for feature detection, the *localization accuracy* describing whether an interest point is accurately located at a specific ground-truth location may be important, and, for feature description, the *information content* describing the distinctiveness of a feature seems to be an appropriate measure as well. Following [136] and [98], a qualitative comparison of feature detectors and feature descriptors is provided in Tables 1.1 and 1.2.

#### 1.4.4 The Future of Local Features

Unfortunately, the question concerning the best feature type still remains open. Hence, some approaches are based on using a *Bag-of-Words*, where different types

of features are extracted and an appropriate combination of these features is derived by learning processes, for example, for material recognition [81] or visual concept classification [137]. As the performance of a feature descriptor is relatively independent of the feature detector, different detector-descriptor combinations may also be applied [32, 119]. An evaluation for finding the best combination of several feature detectors and various feature descriptors has been carried out in [32] and concludes that the MSER and Difference-of-Gaussian (DoG) detectors combined with a SIFT or DAISY descriptor provide the best performance. Recent comparisons with respect to computational effort show that FAST, STAR, BRISK and ORB provide a fast detector, whereas SURF, BRISK and ORB provide an efficient descriptor [98].

A research topic closely related to the extraction of local features focuses on extracting and describing features from point cloud data, and several of the proposed methods are directly or indirectly linked to the same ideas. Thus, SIFT has been extended to 2.5D [82], to 3D [120] and even to  $n$ -dimensional data [28]. Combining a 3D feature detector based on SURF and a 3D descriptor extending SIFT has been presented with *THRIFT* [43]. A method extending the 2D structure matrix to the 3D structure tensor has been proposed for feature detection in depth images which extracts multi-scale *surface normal interest points (SNIPs)* [129]. SIFT has also been extended to multispectral images with *multispectral SIFT (MSIFT)* [24] and to fused intensity-range images with *Complex SIFT (CSIFT)* [21]. Besides, interest points have been transferred to the spatio-temporal domain [71, 141].

The focus of this chapter has been set on image features for passive techniques and thus features which arise from natural circumstances, that is, the scene is not actively manipulated. However, there are situations when passive techniques tend to get unreliable. Features relying on texture will not appear on homogeneous objects and hence, no information can be captured for this part of the scene making tasks such as object recognition, autonomous navigation or scene reconstruction more challenging. In contrast, there are active techniques for establishing robust feature correspondences by actively taking influence on the scene by manipulating the illumination, for example, by projecting illumination patterns. Thus, scene points are assigned with particular labels. These labels can be decoded in the images and hence, they can be interpreted as synthetically generated features allowing for a robust matching.

Furthermore, depending on the material of present object surfaces, the occurrence of features and their localization might be affected by the light transport within a scene. Occurring phenomena such as specular highlights or subsurface scattering heavily depend on light- and view-direction. As texture-based approaches towards feature representation usually are based on assuming that object surfaces exhibit a Lambertian reflectance behavior, they are not suited for objects with complex reflectance behavior, where this assumption fails. Then, effects such as specular highlights, interreflections or sub-surface scattering represent serious problems, and a reliable retrieval of the feature classes is very difficult or even impossible [63]. For overcoming these limitations, multiple-view features which exhibit non-Lambertian reflectance have been proposed in [90]. An overview for techniques addressing the reconstruction of objects with complex reflectance behavior can be found in [63].

## 1.5 Conclusion

In this chapter, visual features have been investigated. A general definition has been derived and the idea of using such features has been traced back far beyond the early days of computer vision. Different types of visual features have been considered among which local features are of special interest due to their almost general applicability. For these, the concepts with the most significant impact on current research have been analyzed and traced over time. The presented concepts provide fundamental ideas still visible in current research, as these are used in a variety of computer vision applications, for example, image registration, image and video retrieval, object recognition, object tracking, navigation of autonomous vehicles, scene reconstruction or scene interpretation. These ideas may further be extended in 2D, to 3D, to the spatio-temporal domain or to more complex scenes with respect to reflectance behavior. The availability of modern active sensors such as Time-of-Flight cameras or the Microsoft Kinect even motivates the combination of 2D images and spatial 3D measurements. Promising results may be expected.

## References

1. Abdel-Hakim AE, Farag AA (2006) CSIFT: a SIFT descriptor with color invariant characteristics. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), vol 2, pp 1978–1983
2. Agrawal M, Konolige K, Blas MR (2008) CenSurE: center surround extrema for realtime feature detection and matching. In: Forsyth D, Torr P, Zisserman A (eds) ECCV 2008, part IV. LNCS, vol 5305. Springer, Heidelberg, pp 102–115
3. Ahmed N, Natarajan T, Rao KR (1974) Discrete cosine transform. IEEE Trans Comput C-23(1):90–93
4. Al-Manasir K, Fraser CS (2006) Registration of terrestrial laser scanner data using imagery. Photogramm Rec 21(115):255–268
5. Alahi A, Ortiz R, Vandergheynst P (2012) FREIA: fast retina keypoint. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), pp 510–517
6. Ambai M, Yoshida Y (2011) CARD: compact and real-time descriptors. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 97–104
7. Attnave F (1954) Some informational aspects of visual perception. Psychol Rev 61(3):183–193
8. Bae K-H, Lichti DD (2008) A method for automated registration of unorganised point clouds. ISPRS J Photogramm Remote Sens 63(1):36–54
9. Ballard DH (1981) Generalizing the Hough transform to detect arbitrary shapes. Pattern Recognit 13(2):111–122
10. Barnea S, Filin S (2007) Registration of terrestrial laser scans via image based features. Int Arch Photogramm Remote Sens Spat Inf Sci 36(part 3):32–37
11. Barnea S, Filin S (2008) Keypoint based autonomous registration of terrestrial laser point clouds. ISPRS J Photogramm Remote Sens 63(1):19–35
12. Bay H, Fasel B, Van Gool L (2006) Interactive museum guide: fast and robust recognition of museum objects. In: Proceedings of the international workshop on mobile vision
13. Bay H, Tuytelaars T, Van Gool L (2006) SURF: speeded up robust features. In: Leonardis A, Bischof H, Pinz A (eds) ECCV 2006, part I. LNCS, vol 3951. Springer, Heidelberg, pp 404–417

14. Bay H, Ess A, Tuytelaars T, Van Gool L (2008) Speeded-up robust features (SURF). *Comput Vis Image Underst* 110(3):346–359
15. Beaudet PR (1978) Rotationally invariant image operators. In: Proceedings of the international joint conference on pattern recognition (ICPR), pp 579–583
16. Bellavia F, Tegolo D, Valenti C (2009) Improving Harris corner selection strategy. *IET Comput Vis* 5(2):87–96
17. Bendels GH, Degener P, Körtgen M, Klein R (2004) Image-based registration of 3D-range data using feature surface elements. In: Proceedings of the international symposium on virtual reality, archaeology and cultural heritage, pp 115–124
18. Besl PJ, McKay ND (1992) A method for registration of 3-D shapes. *IEEE Trans Pattern Anal Mach Intell* 14(2):239–256
19. Bigün J (1990) A structure feature for some image processing applications based on spiral functions. *Comput Vis Graph Image Process* 51(2):166–194
20. Böhm J, Becker S (2007) Automatic marker-free registration of terrestrial laser scans using reflectance features. In: Proceedings of the 8th conference on optical 3D measurement techniques, pp 338–344
21. Bradley PE, Jutzi B (2011) Improved feature detection in fused intensity-range images with complex SIFT (CSIFT). *Remote Sens* 3(9):2076–2088
22. Brenner C, Dold C, Ripperda N (2008) Coarse orientation of terrestrial laser scans in urban environments. *ISPRS J Photogramm Remote Sens* 63(1):4–18
23. Bresenham JE (1965) Algorithm for computer control of a digital plotter. *IBM Syst J* 4(1):25–30
24. Brown M, Süsstrunk S (2011) Multi-spectral SIFT for scene category recognition. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), pp 177–184
25. Calonder M, Lepetit V, Strecha C, Fua P (2010) BRIEF: binary robust independent elementary features. In: Daniilidis K, Maragos P, Paragios N (eds) ECCV 2010, part IV. LNCS, vol 6314. Springer, Heidelberg, pp 778–792
26. Canny J (1986) A computational approach to edge detection. *IEEE Trans Pattern Anal Mach Intell* 8(6):679–698
27. Chen CH, Pau LF, Wang PSP (1998) Handbook of pattern recognition and computer vision, 2nd edn. World Scientific, Singapore
28. Cheung W, Hamarneh G (2007) n-SIFT: n-dimensional scale invariant feature transform for matching medical images. In: Proceedings of the IEEE international symposium on biomedical imaging: from nano to macro, pp 720–723
29. Chum O, Matas J (2006) Geometric hashing with local affine frames. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), vol 1, pp 879–884
30. Collins M, Zhang J, Miller P, Wang H (2009) Full body image feature representations for gender profiling. In: Proceedings of the ICCV workshop on visual surveillance, pp 1235–1242
31. Comaniciu D, Meer P (2002) Mean shift: a robust approach toward feature space analysis. *IEEE Trans Pattern Anal Mach Intell* 24(5):603–619
32. Dahl AL, Aanæs H, Pedersen KS (2011) Finding the best feature detector-descriptor combination. In: Proceedings of the joint conference on 3D imaging, modeling, processing, visualization and transmission (3DIMPVT), pp 318–325
33. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), vol 1, pp 886–893
34. Davis LS (1973) Understanding shape: angles and sides. *IEEE Trans Comput C-26(3):236–242*
35. Deriche R, Giraudon G (1993) A computational approach for corner and vertex detection. *Int J Comput Vis* 10(2):101–124

36. Designs Act 2003 (2003) Office of Legislative Drafting and Publishing, Attorney-General's Department, Canberra
37. Dick AR, Torr PHS, Cipolla R (2004) Modelling and interpretation of architecture from several images. *Int J Comput Vis* 60(2):111–134
38. Ebrahimi M, Mayol-Cuevas WW (2009) SUSR<sub>E</sub>: speeded up surround extrema feature detector and descriptor for realtime applications. In: Proceedings of the CVPR workshop on feature detectors and descriptors: the state of the art and beyond, pp 9–14
39. Felsberg M (2007) Optical flow estimation from monogenic phase. In: Jähne B, Barth E, Mester R, Scharr H (eds) IWCMI 2004. LNCS, vol 3417. Springer, Heidelberg, pp 1–13
40. Felsberg M, Sommer G (2001) The monogenic signal. *IEEE Trans Signal Process* 49(12):3136–3144
41. Feng D, Siu WC, Zhang HJ (2003) Multimedia information retrieval and management: technological fundamentals and applications. Springer, Berlin
42. 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
43. Flint A, Dick A, Van Den Hengel A (2007) Thrift: local 3D structure recognition. In: Proceedings of the Biennial conference of the Australian pattern recognition society on digital image computing techniques and applications (DICTA), pp 182–188
44. Forssen P-E (2007) Maximally stable colour regions for recognition and matching. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), pp 1–8
45. Forssen P-E, Lowe DG (2007) Shape descriptors for maximally stable extremal regions. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 1–8
46. Förstner W (1994) A framework for low-level feature extraction. In: Eklundh J-O (ed) ECCV 1994, part II. LNCS, vol 801. Springer, Heidelberg, pp 383–394
47. Förstner W, Gülich E (1987) A fast operator for detection and precise location of distinct points, corners and centres of circular features. In: Proceedings of the ISPRS conference on fast processing of photogrammetric data, pp 281–305
48. Förstner W, Dickscheid T, Schindler F (2009) Detecting interpretable and accurate scale-invariant keypoints. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 2256–2263
49. Freeman H (1969) A review of relevant problems in the processing of line-drawing data. In: Automatic interpretation and classification of images, pp 155–174
50. Gibson JJ (1950) The perception of the visual world. Houghton Mifflin, Boston
51. Gibson JJ (1961) Ecological optics. *Vis Res* 1(3–4):253–262
52. Grabner M, Grabner H, Bischof H (2006) Fast approximated SIFT. In: Narayanan PJ, Nayar SK, Shum H-Y (eds) ACCV 2006, part I. LNCS, vol 3851. Springer, Heidelberg, pp 918–927
53. Gruen A, Akca D (2005) Least squares 3D surface and curve matching. *ISPRS J Photogramm Remote Sens* 59(3):151–174
54. Guo X, Cao X, Zhang J, Li X (2010) MIFT: a mirror reflection invariant feature descriptor. In: Zha H, Taniguchi R-I, Maybank S (eds) ACCV 2009, part II. LNCS, vol 5995. Springer, Heidelberg, pp 536–545
55. Hamming RW (1950) Error detecting and error correcting codes. *Bell Syst Tech J* 29(2):147–160
56. Haralick RM (1979) Statistical and structural approaches to texture. *Proc IEEE* 67(5):786–804
57. Haralick RM, Shapiro LG (1993) Computer and robot vision. Addison-Wesley, Reading
58. Harris CG, Stephens M (1988) A combined corner and edge detector. In: Proceedings of the Alvey vision conference, pp 147–151
59. Heikkilä M, Pietikäinen M, Schmid C (2009) Description of interest regions with local binary patterns. *Pattern Recognit* 42(3):425–436
60. Hough PVC (1962) Method and means for recognizing complex patterns. United States Patent 3069654

61. Hu M-K (1962) Visual pattern recognition by moment invariants. *IRE Trans Inf Theory* 8(2):179–187
62. Huang J, Kumar SR, Mitra M, Zhu W-J, Zabih R (1997) Image indexing using color correlograms. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), pp 762–768
63. Ihrke I, Kutulakos KN, Lensch HPA, Magnor M, Heidrich W (2008) State of the art in transparent and specular object reconstruction. In: STAR proceedings of Eurographics, pp 87–108
64. Isola P, Xiao J, Torralba A, Oliva A (2011) What makes an image memorable? In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), pp 145–152
65. Kakumanu P, Makrogiannis S, Bourbakis N (2007) A survey of skin-color modeling and detection methods. *Pattern Recognit* 40(3):1106–1122
66. Kang Z, Li J, Zhang L, Zhao Q, Zlataanova S (2009) Automatic registration of terrestrial laser scanning point clouds using panoramic reflectance images. *Sensors* 9(4):2621–2646
67. Ke Y, Sukthankar R (2004) PCA-SIFT: a more distinctive representation for local image descriptors. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), vol 2, pp 506–513
68. Kerr D, Coleman S, Scotney B (2008) Comparing cornerness measures for interest point detection. In: Proceedings of the international machine vision and image processing conference (IMVIP), pp 105–110
69. Koenderink JJ (1984) The structure of images. *Biol Cybern* 50(5):363–370
70. Koffka K (1935) Principles of gestalt psychology. Harcourt, Brace & World, New York
71. Laptev I (2005) On space-time interest points. *Int J Comput Vis* 64(2–3):107–123
72. Laws KI (1980) Textured image segmentation. PhD thesis, University of Southern California, United States
73. 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
74. Lepetit V, Fua P (2006) Keypoint recognition using randomized trees. *IEEE Trans Pattern Anal Mach Intell* 28(9):1465–1479
75. Leutenegger S, Chli M, Siegwart RY (2011) BRISK: binary robust invariant scalable keypoints. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 2548–2555
76. Levine MD (1969) Feature extraction: a survey. *Proc IEEE* 57(8):1391–1407
77. Liang Z, Fu H, Chi Z, Feng D (2010) Salient-SIFT for image retrieval. In: Blanc-Talon J, Bone D, Philips W, Popescu D, Scheunders P (eds) ACIVS 2010, part I. LNCS, vol 6474. Springer, Heidelberg, pp 62–71
78. Liang P, Li S-F, Qin J-W (2010) Multi-resolution local binary patterns for image classification. In: Proceedings of the international conference on wavelet analysis and pattern recognition (ICWAPR), pp 164–169
79. Lindeberg T (1994) Scale-space theory: a basic tool for analysing structures at different scales. *J Appl Stat* 21(2):224–270
80. Lindeberg T (1998) Feature detection with automatic scale selection. *Int J Comput Vis* 30(2):77–116
81. Liu C, Sharan L, Adelson EH, Rosenholtz R (2010) Exploring features in a Bayesian framework for material recognition. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), pp 239–246
82. Lo T, Siebert JP (2009) Local feature extraction and matching on range images: 2.5D SIFT. *Comput Vis Image Underst* 113(12):1235–1250
83. Lowe DG (1999) Object recognition from local scale-invariant features. In: Proceedings of the IEEE international conference on computer vision (ICCV), vol 2, pp 1150–1157
84. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110
85. Ma Y, Soatto S, Kosecka J, Sastry SS (2005) An invitation to 3-D vision: from images to geometric models. Springer, New York

86. Magnusson M, Lilienthal A, Duckett T (2007) Scan registration for autonomous mining vehicles using 3D-NDT. *J Field Robot* 24(10):803–827
87. Marr D (1976) Early processing of visual information. *Philos Trans R Soc Lond B, Biol Sci* 275(942):483–519
88. Marr D, Hildreth E (1980) Theory of edge detection. *Proc R Soc Lond B, Biol Sci* 207(1167):187–217
89. Matas J, Chum O, Urban M, Pajdla T (2002) Robust wide baseline stereo from maximally stable extremal regions. In: Proceedings of the British machine vision conference (BMVC), vol 1, pp 384–393
90. Meltzer J, Soatto S (2005) Shiny correspondence: multiple-view features for non-lambertian scenes. Technical report CSD2006-004, University of California
91. Mikolajczyk K (2002) Detection of local features invariant to affine transformations. PhD thesis, Institut National Polytechnique de Grenoble, France
92. Mikolajczyk K, Schmid C (2001) Indexing based on scale invariant interest points. In: Proceedings of the IEEE international conference on computer vision (ICCV), vol 1, pp 525–531
93. Mikolajczyk K, Schmid C (2002) An affine invariant interest point detector. In: Heyden A, Sparr G, Nielsen M, Johansen P (eds) *ECCV 2002*, part I. LNCS, vol 2350. Springer, Heidelberg, pp 128–142
94. Mikolajczyk K, Schmid C (2003) A performance evaluation of local descriptors. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), vol 2, pp 257–263
95. Mikolajczyk K, Schmid C (2004) Scale & affine invariant interest point detectors. *Int J Comput Vis* 60(1):63–86
96. Mikolajczyk K, Schmid C (2005) A performance evaluation of local descriptors. *IEEE Trans Pattern Anal Mach Intell* 27(10):1615–1630
97. 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
98. Miksik O, Mikolajczyk K (2012) Evaluation of local detectors and descriptors for fast feature matching. In: Proceedings of the international conference on pattern recognition (ICPR), pp 2681–2684
99. Moravec HP (1977) Towards automatic visual obstacle avoidance. In: Proceedings of the international joint conference on artificial intelligence (IJCAI), vol 2, p 584
100. Morel JM, Yu G (2009) ASIFT: a new framework for fully affine invariant image comparison. *SIAM J Imaging Sci* 2(2):438–469
101. Nister D, Stewenius H (2006) Scalable recognition with a vocabulary tree. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), vol 2, pp 2161–2168
102. Nister D, Stewenius H (2008) Linear time maximally stable extremal regions. In: Forsyth D, Torr P, Zisserman A (eds) *ECCV 2008*, part II. LNCS, vol 5303. Springer, Heidelberg, pp 183–196
103. Nixon MS, Aguado AS (2008) Feature extraction & image processing, 2nd edn. Academic Press, Oxford
104. Noble JA (1988) Finding corners. *Image Vis Comput* 6(2):121–128
105. Oliva A, Torralba A (2001) Modeling the shape of the scene: a holistic representation of the spatial envelope. *Int J Comput Vis* 42(3):145–175
106. Pathak K, Birk A, Vaskevicius N, Poppinga J (2010) Fast registration based on noisy planes with unknown correspondences for 3-D mapping. *IEEE Trans Robot* 26(3):424–441
107. Phung SL, Bouzerdoum A, Chai D (2005) Skin segmentation using color pixel classification: analysis and comparison. *IEEE Trans Pattern Anal Mach Intell* 27(1):148–154
108. Prewitt JMS, Mendelsohn ML (1966) The analysis of cell images. *Ann NY Acad Sci* 128(3):1035–1053

109. Rabbani T, Dijkman S, van den Heuvel F, Vosselman G (2007) An integrated approach for modelling and global registration of point clouds. *ISPRS J Photogramm Remote Sens* 61(6):355–370
110. Roberts LG (1965) Machine perception of three-dimensional solids. In: Tippett J, Berkowitz D, Clapp L, Koester C, Vanderburgh A (eds) *Optical and electro-optical information processing*. MIT Press, Cambridge, pp 159–197
111. Rosenberg B (1972) The analysis of convex blobs. *Comput Graph Image Process* 1(2):183–192
112. Rosenberg B (1974) Computing dominant points on simple shapes. *Int J Man-Mach Stud* 6(1):1–12
113. Rosenfeld A, Johnston E (1973) Angle detection on digital curves. *IEEE Trans Comput* 22(9):875–878
114. Rosten E, Drummond T (2005) Fusing points and lines for high performance tracking. In: *Proceedings of the IEEE international conference on computer vision (ICCV)*, vol 2, pp 1508–1515
115. Rosten E, Porter R, Drummond T (2010) Faster and better: a machine learning approach to corner detection. *IEEE Trans Pattern Anal Mach Intell* 32(1):105–119
116. Rublee E, Rabaud V, Konolige K, Bradski G (2011) ORB: an efficient alternative to SIFT or SURF. In: *Proceedings of the IEEE international conference on computer vision (ICCV)*, pp 2564–2571
117. Rusinkiewicz S, Levoy M (2001) Efficient variants of the ICP algorithm. In: *Proceedings of the international conference on 3d digital imaging and modeling (3DIM)*, pp 145–152
118. Schmid C, Mohr R, Bauckhage C (2000) Evaluation of interest point detectors. *Int J Comput Vis* 37(2):151–172
119. Schmidt A, Kraft M, Kasinski A (2010) An evaluation of image feature detectors and descriptors for robot navigation. In: Bolc L, Tadeusiewicz R, Chmielewski LJ, Wojciechowski KW (eds) *ICCVG 2010, part II. LNCS*, vol 6375. Springer, Heidelberg, pp 251–259
120. Scovanner P, Ali S, Shah M (2007) A 3-dimensional SIFT descriptor and its application to action recognition. In: *Proceedings of the international conference on multimedia*, pp 357–360
121. Sedgewick R (1988) *Algorithms*, 2nd edn. Addison-Wesley, Boston
122. Seo JK, Sharp GC, Lee SW (2005) Range data registration using photometric features. In: *Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR)*, vol 2, pp 1140–1145
123. Sheng Y, Shen L (1994) Orthogonal Fourier–Mellin moments for invariant pattern recognition. *J Opt Soc Am A* 11(6):1748–1757
124. Shi J, Tomasi T (1994) Good features to track. In: *Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR)*, pp 593–600
125. Smith SM, Brady JM (1997) SUSAN—a new approach to low level image processing. *Int J Comput Vis* 23(1):45–78
126. Sobel IE (1970) Camera models and machine perception. PhD thesis, Stanford University, United States
127. Steder B, Grisetti G, Burgard W (2010) Robust place recognition for 3D range data based on point features. In: *Proceedings of the IEEE international conference on robotics and automation (ICRA)*, pp 1400–1405
128. Stricker M, Swain M (1994) The capacity of color histogram indexing. In: *Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR)*, pp 704–708
129. Stückler J, Behnke S (2011) Interest point detection in depth images through scale-space surface analysis. In: *Proceedings of the IEEE international conference on robotics and automation (ICRA)*, pp 3568–3574
130. Takaya K (2007) Feature point correspondence of stereo images by monogenic phase. In: *Proceedings of the IEEE Pacific Rim conference on communications, computers and signal processing*, pp 272–275

131. Tamura H, Mori S, Yamawaki T (1978) Textural features corresponding to visual perception. *IEEE Trans Syst Man Cybern* 8(6):460–473
132. Teague MR (1980) Image analysis via the general theory of moments. *J Opt Soc Am* 70(8):920–930
133. Teh C-H, Chin RT (1988) On image analysis by the methods of moments. *IEEE Trans Pattern Anal Mach Intell* 10(4):496–513
134. Tola E, Lepetit V, Fua P (2008) A fast local descriptor for dense matching. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR), pp 1–8
135. Tola E, Lepetit V, Fua P (2010) DAISY: an efficient dense descriptor applied to wide-baseline stereo. *IEEE Trans Pattern Anal Mach Intell* 32(5):815–830
136. Tuytelaars T, Mikolajczyk K (2007) Local invariant feature detectors: a survey. *Found Trends Comput Graph Vis* 3(3):177–280
137. Uijlings JRR, Smeulders AWM, Scha RJH (2010) Real-time visual concept classification. *IEEE Trans Multimed* 12(7):665–681
138. Van Gool L, Dewaele P, Oosterlinck A (1985) Texture analysis anno 1983. *Comput Vis Graph Image Process* 29(3):336–357
139. Von Hansen W (2006) Robust automatic marker-free registration of terrestrial scan data. *Int Arch Photogramm Remote Sens Spat Inf Sci* 36(part 3):105–110
140. Wang Z, Brenner C (2008) Point based registration of terrestrial laser data using intensity and geometry features. *Int Arch Photogramm Remote Sens Spat Inf Sci* 37(part B5):583–589
141. Wang H, Ullah MM, Kläser A, Laptev I, Schmid C (2009) Evaluation of local spatio-temporal features for action recognition. In: Proceedings of the British machine vision conference (BMVC), pp 127–138
142. Wang Z, Fan B, Wu F (2011) Local intensity order pattern for feature description. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 603–610
143. Weinmann M, Jutzi B (2008) Fully automatic image-based registration of unorganized TLS data. *Int Arch Photogramm Remote Sens Spat Inf Sci* 38(part 5):55–60
144. Weinmann Ma, Weinmann Mi, Hinz S, Jutzi B (2011) Fast and automatic image-based registration of TLS data. *ISPRS J Photogramm Remote Sens* 66(6):S62–S70
145. Wertheimer M (1923) Laws of organization in perceptual forms. *Psychologische Forschung* 4:301–350
146. Westheimer G (1999) Gestalt theory reconfigured: Max Wertheimer's anticipation of recent developments in visual neuroscience. *Perception* 28(1):5–15
147. Witkin AP (1983) Scale-space filtering. In: Proceedings of the international joint conference on artificial intelligence (IJCAI), pp 1019–1022
148. Wu J, Rehg JM (2011) CENTRIST: a visual descriptor for scene categorization. *IEEE Trans Pattern Anal Mach Intell* 33(8):1489–1501
149. Yang M-H, Kriegman DG, Ahuja N (2002) Detecting faces in images: a survey. *IEEE Trans Pattern Anal Mach Intell* 24(1):34–58
150. Zhang D, Lu G (2002) Generic Fourier descriptor for shape-based image retrieval. In: Proceedings of the 2002 IEEE international conference on multimedia and expo (ICME), vol 1, pp 425–428
151. Zhang D, Lu G (2004) Review of shape representation and description techniques. *Pattern Recognit* 37(1):1–19
152. Zhang J, Tan T (2002) Brief review of invariant texture analysis methods. *Pattern Recognit* 35(3):735–747
153. Zhao G, Chen L, Chen G, Yuan J (2010) KPB-SIFT: a compact local feature descriptor. In: Del Bimbo A, Chang S-F, Smeulders AWM (eds) *Multimedia 2010*. ACM, New York, pp 1175–1178
154. Zheng Z, Wang H, Teoh EK (1999) Analysis of gray level corner detection. *Pattern Recognit Lett* 20(2):149–162

# Chapter 2

## Where Next in Object Recognition and how much Supervision Do We Need?

Sandra Ebert and Bernt Schiele

**Abstract** Object class recognition is an active topic in computer vision still presenting many challenges. In most approaches, this task is addressed by supervised learning algorithms that need a large quantity of labels to perform well. This leads either to small datasets (<10,000 images) that capture only a subset of the real-world class distribution (but with a controlled and verified labeling procedure), or to large datasets that are more representative but also add more label noise. Therefore, semi-supervised learning has been established as a promising direction to address object recognition. It requires only few labels while simultaneously making use of the vast amount of images available today. In this chapter, we outline the main challenges of semi-supervised object recognition, we review existing approaches, and we emphasize open issues that should be addressed next to advance this research topic.

### 2.1 Introduction

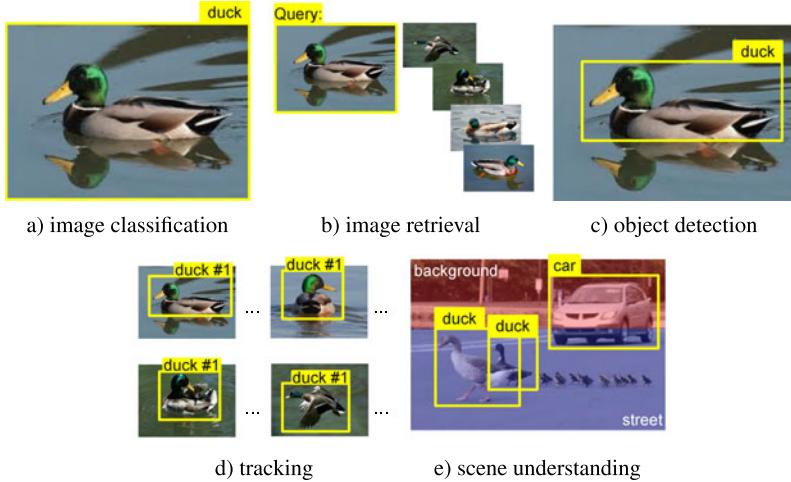
Object recognition is one of the central topics in computer vision and an integral part of many computer vision tasks. To mention only a few, image classification (Fig. 2.1a) is one of the more basic tasks that includes object recognition to classify an image, for example, *duck*. Content-based image retrieval (CBIR, Fig. 2.1b) contains object recognition to search systematically for images that contain these objects. Object detection (Fig. 2.1c) must in addition specify the actual position of the recognized object in the image (marked as a bounding box), thus a clear separation between foreground and background is essential. Tracking (Fig. 2.1d) is based on object detection and tries to track the localized object across a sequence of frames. Finally, scene understanding (Fig. 2.1e) aims to capture the whole scene including all interactions among objects and the environment, for example, to

---

S. Ebert (✉) · B. Schiele

Max Planck Institute for Informatics, Saarbrücken, Germany  
e-mail: [ebert@mpi-inf.mpg.de](mailto:ebert@mpi-inf.mpg.de)

B. Schiele  
e-mail: [schiele@mpi-inf.mpg.de](mailto:schiele@mpi-inf.mpg.de)

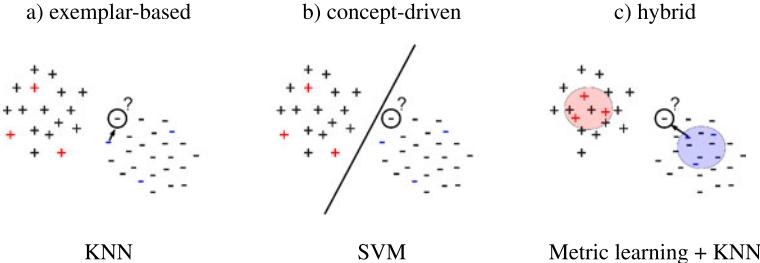


**Fig. 2.1** Computer vision applications with object recognition as an integral part

warn the car driver before an accident happens with the ducks. This list of computer vision tasks could be continued arbitrarily. But although object recognition is a crucial part it is surprising that even image classification, which only aims to provide the image label, does not provide satisfactory results on more challenging datasets.

In contrast, humans can quickly and accurately recognize objects in images or video sequences. They can categorize them into thousands of categories [10]. Beyond that they can track objects in videos and they are able to interpret the entire scene and to infer subsequent events. Of course, human perception, recognition, and inference also have their limits but they might serve as a good starting point to improve upon. Therefore it is not surprising that computer vision, in particular the machine learning part of it, is mainly driven by cognitive science—the science of understanding the learning and thinking of humans. But as controversial theories in cognition are, as diverse are the approaches in computer vision and machine learning.

One of these long-lasting debates in cognition focuses on the question whether a human learns exemplar-based or concept-driven. The exemplar-based model [67, 74, 129] assumes that humans store a list of exemplars for each category. A category decision will be made based on a similarity to existing exemplars. One of the most prominent representative in machine learning is the  $k$  nearest neighbor classifier visualized in Fig. 2.2a. This classifier looks for the nearest labeled neighbor in the training set marked as red and blue data point and uses the label of this training sample for classification. In contrast, concept-driven learning assumes that people abstract to a model or a prototype that is used for classifying objects [68, 69, 84]. This paradigm can be found in many algorithms that learn a model of a category and do any kind of generalization such as SVM that learns a decision boundary (Fig. 2.2b). More recently, there is a tendency towards the theory that humans use



**Fig. 2.2** Illustration of several learning principles that are used to classify the marked unlabeled data point: (a) exemplar-based learning, e.g., KNN classification, (b) concept-driven, e.g., SVM, or (c) hybrid approach, e.g., that groups exemplars around a general concept by transformation with metric learning and than applying KNN. *Blue* and *red* points are the labels of two different classes, and *black points* are the unlabeled data

either multiple learning systems in parallel [4, 35] or a hybrid version that groups exemplars around a general concept. This approach can be found for example in a combination of metric learning and KNN that first maps the exemplars to a more discriminative description, that is, examples of the same class are closer together visualized as red and blue area in Fig. 2.2c, and then applies KNN.

Another equally controversial but much older debate revolves around the question how we gain the insight that forms the base of our decisions. This leads to one of the most fundamental questions in machine vision: whether and how much supervision do we need? On one hand, a human learns provable faster with supervised feedback [3]. Therefore, it is not surprising that state-of-the-art performance is achieved by supervised algorithms. However, this success has to be put into perspective as the dataset construction itself contains an enormous amount of supervision. Each method is only as good as the underlying training data. If the learner sees only the side view of a car during training, then the resulting classifier will fail on cars shown in front views or from above. This aspect is often neglected in the subsequent evaluation and leads to datasets that are either small and strongly biased [82, 113] or large and error-prone [124]. On the other hand, it is also clear that many human decisions are driven to some extent by intuition, that is, more or less unsupervised, particularly in unfamiliar or risky situations [51]. But although unsupervised learning is an important research area [103, 123], for example, for object discovery or novelty detection, a minimum level of supervision is required at the end to judge the quality and to gain insight. Therefore, semi-supervised learning (SSL) has to be turn out the paradigm that combines the advantages of supervised learning and unsupervised learning by using both labels as well as the underlying structure or geometry in the data.

In the following, we discuss the large potential of semi-supervised learning in Sect. 2.2. After that, we outline in Sect. 2.3 the main challenges of object recognition with respect to semi-supervised learning and how it is already addressed in previous work. Finally, we point out open issues and we give recommendations how we could tackle those in Sect. 2.4.

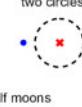
## 2.2 How much Supervision Do We Need?

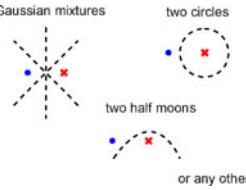
*Mind without structure is empty and  
perception without labels is blind.*  
Translated from [52]

Imitation, intuition, and experience play an important role in human decision making [23] especially under risk [51]. But only a small fraction of this accumulated knowledge is labeled. The discussion around the question on *whether and how much supervision a human need* can be traced back at least to the philosophical theories of the 17th century. [52] was the first who argued that labels (knowledge) and structure (perception or observation) are closely intertwined, summarized in one of his key phrases (see beginning of this section). His theory fundamentally changed and influenced our way of thinking and acting. After 200 years of research, some of his basic assumptions and argumentations might be obsolete. But the underlying theory and the main argumentation itself is still up-to-date. Indeed this theory seems to be obvious because in addition to the things we learn supervised at home or in school, there are many other things that we learn without any teacher. For example, how we move, how we grab a glass, how we use language before we start school, how we make fast decisions and so on. Of course, many of those things are learned feedback-driven in the sense that an action is completed successfully or not. However, there are still many actions or feelings that we cannot explain let alone derive solely based on knowledge. In that sense, semi-supervised learning seems a natural choice to address object recognition as it allows us to improve state-of-the-art supervised learning approaches with more data without the need of correct annotations.

But in defense of the more skeptical people, one has to state that this large unlabeled part of semi-supervised learning is almost impossible to grasp. Actually, it is even not clear whether humans will ever be capable to understand it in their completeness. For the simple reason that in the course of evolution, we only had to understand and infer simple causalities, for example, *I take the glass of water because I am thirsty*. But we were never forced to understand the entire chain of actions and decisions that leads to this final action, for example, *grasping the glass and drinking*. In fact, every physical movement is a highly individual action that is based on imitation and experience. Although this might lead to sub-optimal movements or actions, the acquired knowledge is quite sufficient to survive in everyday life—even if we notice some limitation and ask for more supervision, for example, to get rid of pain induced by suboptimal movements or to run faster in a marathon. Most advices only give a direction and do not describe a muscle-induced action in its full complexity.

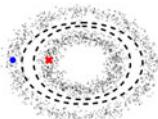
One might argue that these examples are indeed more physical. But even if we limit these considerations to our decisions that could be purely driven by our knowledge, we still observe many decision based on the so called *gut feeling* or other feelings that we cannot explain. Why do we know that someone is annoyed or sad or impatient although this person uses exactly the same words as he does every day? Why is it impossible to imagine in advance how we will react or feel after a certain event, for example if we loose a competition. Even more complicated is to infer how other people will react on an event. The reason is simple and devastating at the same

Given two labels	Space of hypotheses	
	without unlabeled data  <b>Gaussian mixtures</b>	 <b>two circles</b>



or any other

with unlabeled data



**Fig. 2.3** Unlabeled data reduce the space of hypotheses if there are only few labels

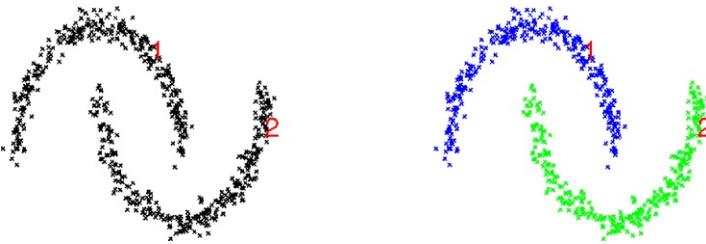
time: We are overwhelmed by millions of sensations per minute of which only a small fraction of impressions are processed consciously and the rest subconsciously and this is only a tiny fraction of what the entire world perceives. Thus, to answer at least the questions with respect to our own, we have to assimilate all sensations. With this insight, it becomes clear why research in semi-supervised learning (SSL) is still not where it should be. To bring a machine into the closer range of human thinking, we have to tap into the vast amount of unlabeled data meaning a ratio of 1 labeled to 1 million unlabeled data points and not the common ratio of today's task descriptions of 1 labeled to 100 unlabeled observations.

Besides these more philosophical considerations, there are also many practical reasons for SSL. One obvious argument is the reduction of the hypotheses space [5] in particular if there are only few labels. Figure 2.3 shows one point per class in the leftmost image. Without any additional information, the space of possible hypotheses is less goal-oriented (Fig. 2.3 middle) while unlabeled data reduces this space as shown in the right visualization. This speed up of concept learning through relevant prior knowledge has been also verified in cognition by [70, 78]. A mechanical engineer will proceed faster and more goal-oriented when assembling a machine in comparison to a layman because he already knows how to use the tools and where the single items should be approximately placed.

Another reason for SSL is the low amount of supervision. In particular for tasks such as semantic image labeling or image understanding, where we need pixel-wise annotations, this advantage becomes increasingly important. But also for tasks such as object detection or recognition, we observe a substantial improvement the more data are used. The most image descriptions are high dimensional resulting in a strong demand for data. But the labeling process is not always reliable—for example, when using Mechanical Turk [124]. Finally, some applications need a continuous update, for example, for separating spam emails from valid emails.

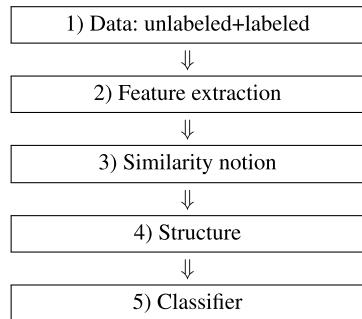
## 2.3 Challenges of Object Recognition

Semi-supervised learning (SSL) seems a reasonable approach to tackle object recognition as it makes use of both supervision in terms of labels and structure (geom-



**Fig. 2.4** Two half moons dataset with exactly one label per class (marked by a *red number*): before classification (*left*) and after classification (*right*) with 100 % accuracy

**Fig. 2.5** Workflow of semi-supervised learning algorithms in vision

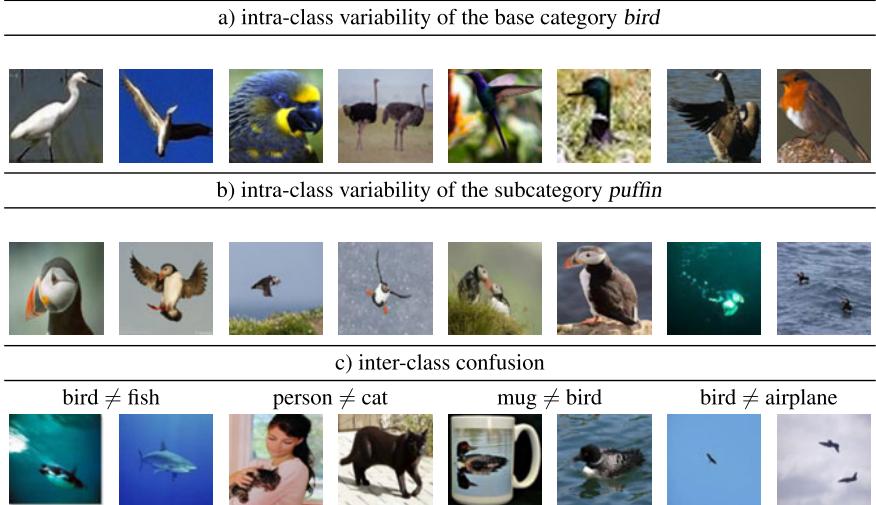


etry) that comes with the unlabeled and labeled data. Therefore, it is obvious that both parts strongly influence the performance of the classifier. In the following, we discuss the challenges of both components separately starting with the structural problems in Sect. 2.3.1 followed by the difficulties in getting representative labels in Sect. 2.3.2.

### 2.3.1 Structural Problems

The apparently most promising but also much more complicated direction for SSL is the improvement of the structure itself. Imagine a dataset like the *two half moons* shown in Fig. 2.4 with two labels marked with red numbers. It does not matter which label is used for classification. The used SSL algorithm [132] achieves always a classification performance of 100 %. Although this is an artificial example it still reflects our common sense assumption that there is exactly one concept for each class and each instance of this class is organized around this central prototype [19, 75]. But often, there is a large gap between our base assumption and today's computer vision task descriptions and solutions.

Figure 2.5 visualizes the general workflow of object recognition with SSL algorithms: Based on our dataset that consists of labeled and unlabeled data, we compute image descriptors for each image. After that we compute the similarities between each image pair with some measure. The resulting structure is used by SSL algo-

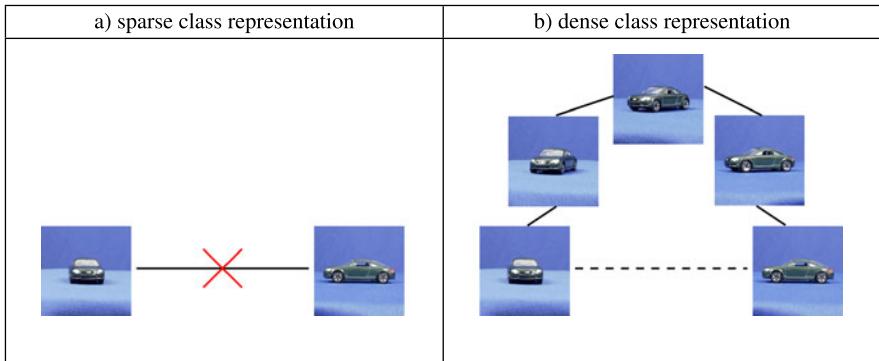


**Fig. 2.6** Examples for (a) large intra-class variability for the base category *bird* (top row) and (b) for the subcategory *puffin* of the category *bird*, and (c) small inter-class differences (bottom row)

rithms, for example, for EM clustering [73], as a regularization term to improve SVM [50, 98], or to build a graph structure and to find a solution with Mincut [12] or by label spreading [132]. However, each of these classifiers can be only as good as the extracted geometry of the data and this strongly depends on three main sources: (1) data, (2) image description, and (3) similarity notion. Furthermore, the quality of each single source is dependent on both the approaches that are used for these steps but also on the quality of the previous steps. This means, information loss for example through an incomplete dataset will be propagated to the classifier and cannot be compensated by one of the intermediate steps. Similar argument holds for image description: if one aspect is neglected, for example, color, the best similarity measure will be not able to properly distinguish between *green apples* and *red tomatoes*. In the following, we discuss each of these three components separately.

### 2.3.1.1 Data

Most common datasets for image classification like Caltech 101 [38], PASCAL VOC [36], Animals with Attribute [59], LabelMe [114], animals on the web [9], or ImageNet [27], are generated for supervised classification. They provide full label information that might be error-prone in particular when crowd-source services like Mechanical Turk are used [124]. They contain a large intra-class variety within a base category such as *bird* (Fig. 2.6a) but also within one specific subcategory of this class such as *puffin* (Fig. 2.6b). Without a good description and understanding of the concept, it will be difficult to connect those examples and group them together to one class. Small inter-class variation is the other end of the scale (Fig. 2.6c) leading



**Fig. 2.7** Example of (a) a sparse class description that makes it difficult to find a connection between both images and of (b) a dense class representation that makes it possible to find a way from the front view to the side view of a car over several viewpoints

to many overlapping and confusing areas that are even for humans difficult to learn. Finally and most unfortunately, they contain usually a limited amount of images because labeling is expensive.

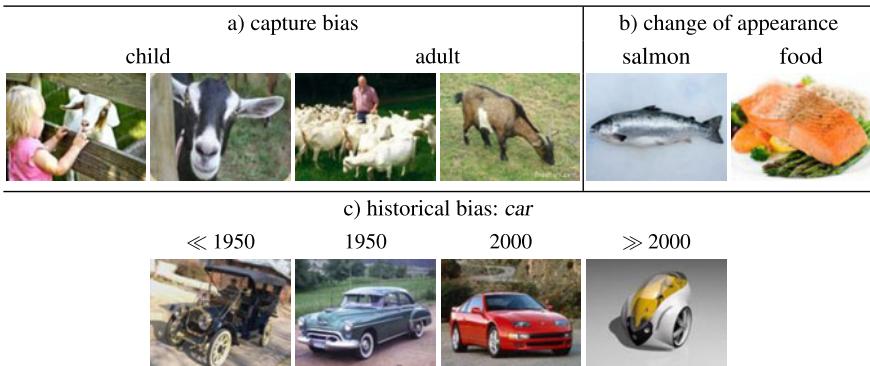
An ideal dataset for SSL should be *dense* enough that means each class should be densely sampled that allows to find compact and well separated clusters. In this dataset, we might be able to connect the front view of a car and a side view of car as in Fig. 2.7. But on the other hand, this dataset should be also *sparse* enough to avoid overheads due to space and time complexity. Usually, SSL approaches such as graph-based algorithms come with a quadratic time complexity in the number of images. Because of this complexity, the *the-more-data-the-better* strategy can usually not be applied.

A second reason why *the-more-data-the-better* strategy does not work well in practice is that only a small fraction of the data, for example, added from the internet, is helpful for our classification task. Furthermore, these data sources often have a certain bias in terms of image type. One example is the data source bias. Flickr, that is the base for PASCAL VOC [36], contains mostly holiday pictures. Therefore *person* is with 31.2 % the most common object in this dataset. The second most frequent object is *chair* with 8.5 %. Another bias can be also seen in Fig. 2.8 (top row) that shows the first results of approx. 5.8 million results for the query *car* in Flickr. In contrast, Google shows more professional images that are often generated for marketing purpose as you can see in Fig. 2.8 (bottom row).

Another problem comes with the capture bias. This is usually a result from human properties such as body height. Most images are taken from an adult person in a standing position thus from an average height of 1.6–1.8 meters (Fig. 2.9a). A simple change to a child position, that is, <1 meter, leads to a different viewpoint and thus perception, for example, some things appear larger (buildings) or more frightening (animals). Furthermore, most people are right handed resulting, for example, in many images of mugs with the handle on the right side. Some objects have a quite different appearance (Fig. 2.9b), e.g., salmon considered as an animal vs. food, and



**Fig. 2.8** Data source bias: First results for the query *car* with (a) Flickr that contains more holiday pictures of cars (*top row*), and (b) Google images with focus on racing cars (*bottom row*)



**Fig. 2.9** Dataset bias due to (a) the sighting angle, (b) the semantic of an object, or (c) because of historical trends

other categories might change their appearance over time (Fig. 2.9c). Of course, it seems likely that massive amounts of data also contain relevant images but to find these images we have to process over millions of images for this single class.

**Related Work** Many SSL algorithms in particular graph-based algorithms come at least with a runtime of  $O(n^2m)$  with  $n$  the number of data and  $m$  the number of feature dimensions. This runtime is needed to compute all similarities between image pairs and to construct the graph. Thus, the applied algorithm depends strongly on the number of data and the dimensionality of the features but also on the approach itself. For example, [132] provide both a closed form solution that would need the inversion of a  $n \times n$  matrix and an iterative procedure that is faster and often avoids over-fitting. In general, there are two different strategies to reduce the runtime: (i) a reduction of the data space ( $\ll n$ ) to a representative subset of unlabeled data or (ii) an approximation either of the similarity matrix or the eigenvectors.

- (i) *Data reduction.* The most common approach to data reduction is clustering to find representative unlabeled data either by hierarchical clustering [64], or by k-means clustering [65, 100]. [26] propose a Greedy approach that starts with the labels only and successively add unlabeled samples farthest away from the

current set of labeled and unlabeled data. [37] find similar nodes by spectral decomposition and merge these together. Another technique is to treat this task as an optimization problem that considers each point in a data set as a convex combination of a set of archetypical or prototypical examples either with a fixed number of archetypes [21] or with an automatically learned number of these prototypes [85]. These techniques are used, e.g., to find typical poses [8], or to summarize a video sequence [34].

- (ii) *Approximation.* In contrast, Nyström approximation is employed to approximate the entire kernel matrix. This approximation is estimated also on a subset of data that are retrieved either by random sampling [131] or with k-means clustering [130]. This approximation can then be used to find a segmentation [42], for similarity search [122], or face recognition [109]. To speed up the algorithms, [40] propose an approximation of the eigenvectors of the normalized graph Laplacian. [115] solve the dual optimization problem by introducing a sparsity constraint, and [54] use stochastic gradient descent to solve the TSVM.

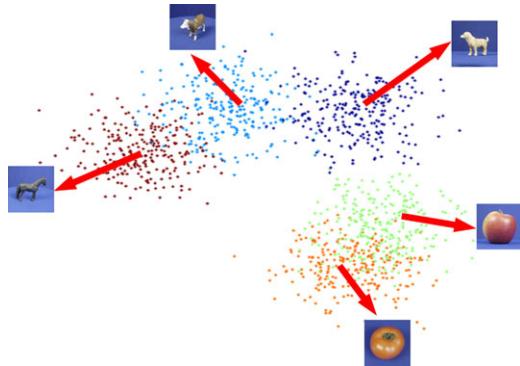
In [33], we focus mainly on the question if more unlabeled data have really a positive impact on the classification performance. In particular, we challenge the *the-more-data-the-better* strategy that is common sense in the computer vision community but also comes with an increase in runtime and space. This question is difficult to answer as adding unlabeled data leads to a different field of research due to the dataset bias [82, 113] and the data source bias (Sect. 2.3.1). Therefore, we focus on ILSVRC 2010 with 1 million images and reduce this large amount of data to a representative subset of unlabeled data showing that this representative subset leads to a better graph structure than using all unlabeled data. We compare our approach to [26] and [65] that can be considered state-of-the-art methods to reduce the graph size. But in comparison to previous work, we analyze also the effect of more unlabeled data. Additionally, our work is the first attempt to process more than one million data points that is far more than 30,000 data points used in previous work [26, 65, 130]. But this can be only seen as good starting point to improve on.

### 2.3.1.2 Image Description

Suppose we have a dataset that captures the broad variety of each class, for example, different viewpoints, several contexts and so on. Thus, there is a chance to build a compact cluster structure similar to Fig. 2.10. Then it does not automatically mean that we are also able to exhaust this potential. Today's image descriptors are far away from capturing all these different aspects that humans can easily recognize. In the following, we list briefly most of the common problems. See also [43] for a short overview of today's problem in computer vision.

**Intra-class/Inter-class Variability** As it mentioned before, many classes come with a large intra-class variance in their appearance and their surrounding environment. The class *bird* is one of the extreme cases where even the height varies from few centimeters like the hummingbird to almost 2 meters like the flightless ostrich

**Fig. 2.10** Structure with dense representation but still overlapping regions



(Fig. 2.6a) not to mention the large variation in shape and in color. Of course, a limitation to one species (Fig. 2.6b) might constrain the general appearance of an object but not the number of different poses or the context around this object. On the other side, there are classes that look similar to each other in particular in some poses, for example, a bird and an airplane in the sky (Fig. 2.6c), or when two classes jointly appear in an image, for example, a cat in the arms of a person or a sticker from an animal at a mug.

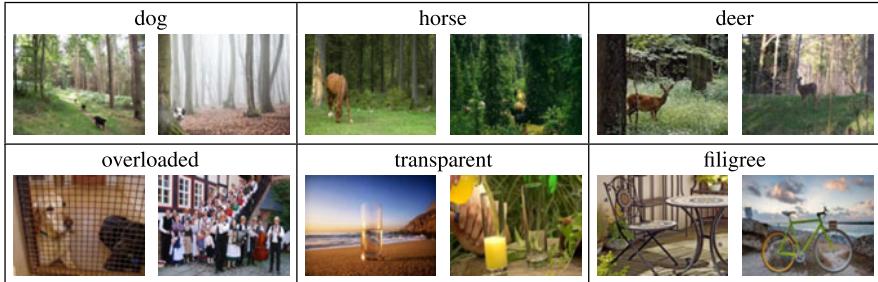
**Background Clutter** Some images are dominated by their background as can be seen in Fig. 2.11 where it is almost impossible to see some objects because of the trees. Often, these images are confused because of their similar-looking background. There are images with an overloaded background structure that are similar to many other images in a dataset. Finally, some objects are difficult to distinguish from the background because they are transparent (like glass) [44], or filigree like a bicycle or a chair.

**Illumination Changes** Another problem is the change in illumination depending on the time of the day and the season (Fig. 2.12). For example, a lake is susceptible to lighting conditions due to its surface and volume properties resulting in a wide color spectrum. But also many other objects look different during the day and at night, for example, trees are green during the day and dark at night.

**Truncation and Partial Occlusion** Partial occlusions are an omnipresent property. Herd animals like sheep or gazelles occur frequently in groups. As already mentioned before, some objects can be covered to a large extend by a person using that object like a bicycle or chair. And other object classes are very large so that they are only partly captured or truncated like a cathedral (Fig. 2.13).

**Shape Variation** Some categories have a large variation in shape and appearance, for example, *chair* (Fig. 2.14), *table*, or *lamp*. These categories can be often only described by their function such as *something to sit on*.

**Mimesis and Other** Another set of problems comes with the evolutionary adaptation of some species to their background so that they are difficult to recognize by



**Fig. 2.11** Examples with a dominating background that is shared among different classes (*top row*), and examples with overloaded background and object that are transparent or filigree so that background is always a part of the object (*bottom row*)



**Fig. 2.12** Examples of a lake with different illuminations depending on the time of day and the season



**Fig. 2.13** Examples of truncations and partial occlusions



**Fig. 2.14** Examples of the large variety in shape for the class *chair*

other animals, for example, the chameleon or the flounder. Other animals such as zebras are indeed visible but it is difficult to point out an individual animal due to their pattern structure (Fig. 2.15).

Basically, the ideal image descriptor should emerge with some prior knowledge about what and where the object is located in the image, how to separate the background from the main object, which color is trustable or rather how to adjust this color, and what are the possible and feasible poses of an object. Furthermore, this descriptor should have a general idea of the shape and texture of an object to infer which part is occluded or truncated. While a human focuses led on the main object,



**Fig. 2.15** Examples of objects that are difficult to distinguish from their background or to identify the object-specific shape



**Fig. 2.16** Examples of images that are difficult to understand without color information

many of today's image descriptors such as dense SIFT analyze every single blade of grass or every single leaf from a tree leading to an overcrowded image description that often considers only one aspect in the image like color or gradients. Of course a good similarity metric can handle this high dimensionality. But an information loss in this partial extraction propagates to the classifier. Figure 2.16 shows some examples with and without color information. Even for a human it is hard to follow a soccer game or to distinguish between eatable and poisonous mushrooms by just omitting color, not to mention information such as texture or shape.

**Related Work** As mentioned before, most image descriptors lack still expressiveness. They consider only one aspect when describing an image, for example, texture, shape or color. This leads inherently to an enormous information loss. Therefore, a combination of several features are essential, for example image-based features with geometry [14, 83, 125], shape with texture [20], several local appearances [94], local and global appearances [61], HOG with texture [121], multiple kernels for global as well as local features learned with an SVM [46, 105, 117], with boosting [28], or with conditional random fields [96].

In SSL, the combination of multiple graphs offers the possibility to capture different aspects in the data. For graph-based methods, there are few works that combine graph structures similar to multiple kernel learning (MKL) [2, 116]. In [55], they learn weights for combining graph Laplacian within an EM framework. [22] propose a method to find one graph from a set of graphs that best fits the data. [112] formulate this combination as a multi-modality learning problem that fuses different modalities either linearly or sequentially. [6] use domain knowledge to ex-

tract three different sources, that is, time, color and face features, that are combined with different hyperparameters. Finally, [47] and [110] combine a similarity and a dissimilarity graph and apply label propagation on this mixed graph structure. Most of these previous works are developed for applications in bioinformatics. But more importantly, they combine often only graphs based on different parameters, that is, a different number of neighbors  $k$  or different weight functions.

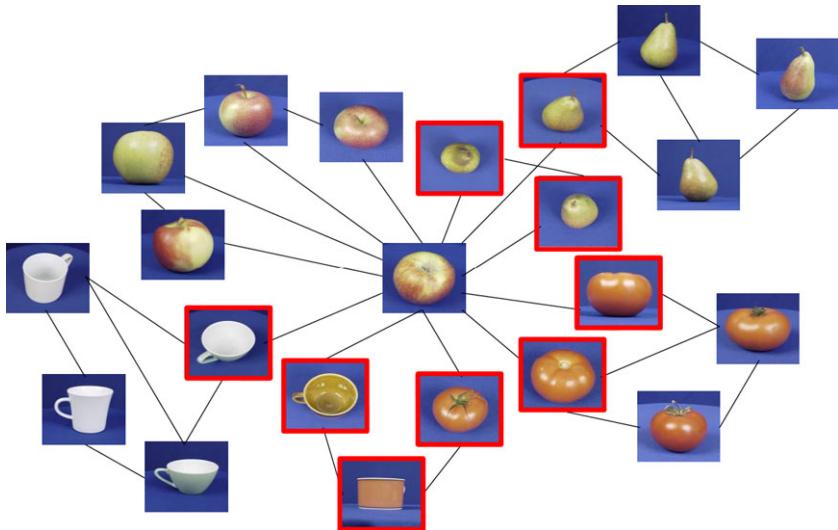
In [30], we show the strong influence on the graph quality when combining different image descriptors leading to a completely different and more powerful graph structure. Additionally, we use the SVM output to construct a new graph. But in comparison to [86] who use the SVM output to delete and insert existing edges, we build a complete new graph based on these decision values and combine this graph with our original graph. This leads to a richer and better connected graph structure than the graph structure in [86].

### 2.3.1.3 Similarity Notion

The final crucial part of the structure extraction is the similarity measure. Most frequently used is the Euclidean distance with a Gaussian kernel weighting. This is usually a good choice for feature vectors of low dimensionality ( $\ll 100$ ) containing only little noise. But as mentioned before, most image descriptors aggregate many not preprocessed information that leads to a high-dimensional vector ( $> 10,000$ ) from which only a small fraction of dimensions are relevant for a object class. In particular Euclidean distance is known to be sensitive to noise that becomes more prominent the more dimensions are used. One phenomenon that we observe with Euclidean distance is that some images are similar to almost all other images. The resulting structure (such as shown in Fig. 2.17) harms almost every classification algorithm because there is no clear separation between different classes [119].

Another problem comes with the missing weighting of the single dimensions in the feature space, that is, all dimensions are equally considered. But usually only a small fraction of this high dimensional feature vector is relevant for each class. Finally, we often consider image pairs instead of groups of images. This is easy to implement but seems suboptimal for good generalization. A human who has never seen a zebra before and only gets the first image from Fig. 2.18 will certainly have problems to build a general concept or model of a zebra because there is no information about the shape, the size, or the environment around this animal. Without these higher order relations extracted from a group of images, it might be difficult to distinguish the first image from the sofa shown in the last image.

**Related Work** Metric learning is a promising direction to tackle this problem. These methods find a better data representation such that examples within a class are close together and examples from different classes are far away, that is, small intra-class distances and large inter-class distances. Metric learning approaches can be split into unsupervised, supervised, and semi-supervised methods that are further divided into global and local learning methods. See also [126] and [29] who provide a more detailed exploration of metric learning methods.



**Fig. 2.17** Problem of Euclidean distance in a high dimensional space: The image in the *middle* is similar to many other images. *Red boxes* indicate false neighbors

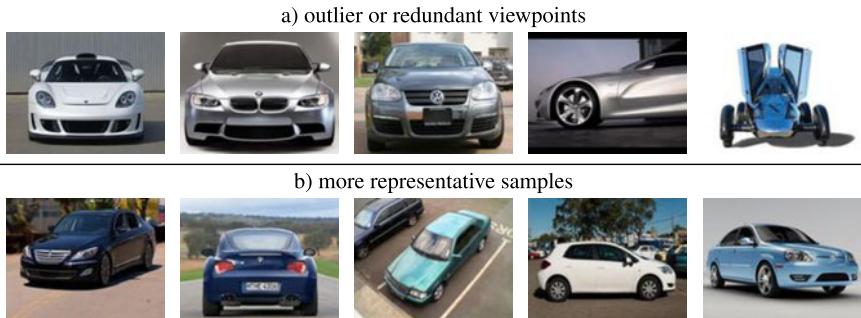


**Fig. 2.18** Pairwise image similarities might be problematic due to the missing generalization. From the *first image* it is not clear how to generalize so that this image do not get confused with the sofa in the *last image*

In [31] and [29], we analyze several supervised and unsupervised metric learning approaches with respect to a better graph construction. Particularly, we apply PCA [79] and LDA [41] to reduce the dimensionality of our feature representation and compare both methods. Furthermore, we also analyze ITML [24]. Instead of reducing the number of dimensions, it learns a weighting of the feature dimensions. The advantage of ITML is that it can be transformed into a kernelized optimization problem. Thus, the runtime depends only on the number of labels that is usually smaller than the number of dimensions ( $n \ll d$ ). Additionally, this approach shows state-of-the-art performance on Caltech 101 [58]. Finally, we integrate ITML in a semi-supervised metric learning scheme that leads to an increased performance.

### 2.3.2 Labeling Issues

The second import issue besides the structure is the label information. As mentioned before, supervision causes no problems if the structure itself perfectly separates



**Fig. 2.19** SSL is strongly dependent on the representativeness of the small training set: **(a)** less representative samples for the entire class *car* vs. **(b)** more representative samples in terms of viewpoints

the classes. But usually this is not the case. Therefore, the label information plays an important role in particular for semi-supervised learning where we have only few labels per class. While for supervised learning more data is labeled, in SSL we have to deal with a ratio of 1 %–5 % labeled to 95 %–99 % unlabeled data. Thus, there is a need for high quality labels that are representative for the class and allow a better generalization. Additionally, we have to ensure that there is at least one label for each mixture (e.g., different viewpoints or appearances) of one class otherwise it might be difficult to classify unseen viewpoints. Figure 2.19 shows five less representative samples for the class *car* in the first row, assuming that the test set contains also the back or the side view of a car. In contrast, the second row shows more representative samples of this class so that the main properties of this object will be apparent such as the shape and the surface.

Coming back to Fig. 2.17 if the image in the middle with these many false neighbors is labeled, then most of the neighboring images will be false classified (marked with a red bounding box) because of the strong impact on the direct neighbors. Another problem occurs when a class is split into separate clusters, for example, front view of a car and side view of a car, and there are only labels for one of these sub-clusters. The other sub-clusters cannot be classified correctly anymore. Ideally, we have labels that are representative or prototypical for a class that means they lie in a dense region and consider each aspect or viewpoint of a class.

**Related Work** Active learning is a well known strategy to reduce the amount of supervision to a small but representative subset and to improve the quality of the learner at the same time. This is also verified by cognitive science as [3] shows that a higher accuracy is achieved with feedback during the learning in comparison to the scenario where supervision is only provided at the beginning. In machine learning, active learning leads in most cases to better performance. [1] show that some NP-complete learning problems become polynomial in computation time. On the other side, active learning in combination with some classification algorithms might lead to poor performance, for example, SVM with few examples at the beginning [120]. Model selection is critical for these algorithms [108].

Most popular is pool-based active learning. These methods consider all unlabeled data as a pool from which samples are drawn to be labeled. In general, pool-based methods can be divided by their sampling strategy into three different types. Exploitation-driven methods [97, 111] focus mainly on uncertain regions during the learning process. In contrast, methods based on exploration sampling [13, 72] estimate the overall distribution of the entire data space and query samples that represent and cover this space. Finally, there are also strategies that combine both exploration and exploitation to get samples that are uncertain but also diverse. We refer also to [99] and [29] who provide a general overview on different active learning strategies.

In [32], we propose a novel sampling criteria for exploration that shows significant better performance in comparison to previous exploration criteria. Furthermore, we address active learning by proposing a reinforced active learning formulation (RALF) that considers the entire active learning sequence as a process. Our approach can deal with multiple criteria, is able to have time-varying trade-offs between exploration and exploitation, and is fast and efficient due to a compact parameterization of the model without dataset-specific tuning. In comparison to [7] who also use a reinforcement procedure, our model comes with fewer parameters, more flexibility in terms of sampling criteria, and provides always a linear combination of exploration and exploitation instead of switching between criteria. For the linear combination, we extend the work proposed by [15] to a time-varying combination that leads to a better adaptivity to dataset requirements. Finally, we show in [33] that a potentially large improvement is possible when applying metric learning with more representative labels. To this end, we combine active learning with metric learning and show improvements of more than 10 % over our previous publication [31] and more than 20 % improvement as compared to our first publication [30].

## 2.4 Future Perspective

In this last section, we will mention open issues and how we could tackle those in Sect. 2.4.1. These suggestions are closely intertwined to cognitive science for the simple reason that particularly object recognition raises automatically the question how humans form class concepts. Thus, it is not surprising that this close relationship has been studied earlier, for example, in the *Roadmap of Cognitive Vision* [118]. Of course, the human object recognition should be only seen as a good starting point to improve on. Therefore, we finally discuss in Sect. 2.4.2 also the human weaknesses in perception, learning and inference and how we might overcome those with machine learning and computer vision.

### 2.4.1 What Can We Learn from Human Object Recognition?

Following the general structure of this chapter, we discuss open issues of each of the components of SSL separately, that is, (1) data, (2) image description, (3) sim-



**Fig. 2.20** Visualization of an incomplete object class description that makes it difficult to find relation between these images



**Fig. 2.21** Visualization of a more complete object class description extracted from a video sequence

ilarity notion and (4) supervision. Additionally, we also challenge the use of label propagation in the last subsection called (5) exemplar-based vs. concept-driven learning.

#### 2.4.1.1 Data

[133] stated in their position paper about graph-based SSL that one of the limitations of these algorithms results from the common sense assumption that each class can be projected to a single manifold. Thus, they suggest to model classes by a mixture of multiple manifolds. This observation is particularly in computer vision not novel. Also [92] distinguish between visual classes and object classes as there are classes such as *chair* that cannot be modeled with one mixture. Even if this is an important aspect, an at least equally important problem is the incompleteness of today's datasets in terms of viewpoints and variations for a class to fully leverage the power of SSL algorithms. In [29], we assume that most classes can be described with one concept and all exemplars of these classes are grouped around this concept [19]. But often we have to deal with class descriptions as shown in Fig. 2.20 for the class *kingfisher*. Even for a human who have never seen this class before might have problems to merge these images in one compact mixture because color, shape, and appearance are quite different. But an image sequence might provide a path among those images as visualized in Fig. 2.21 and helps to extract an object models similar to the work of [90] shown for car tracking.



**Fig. 2.22** Visualization of feasible and unfeasible poses and scenes for the class *anteater* that will be more obvious with informations such as 1.5–1.8 m long

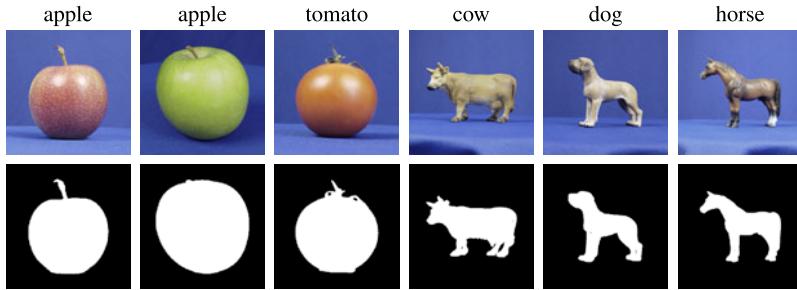
In [33], we analyze the problem of incomplete datasets by looking at existing datasets such as ImageNet with more than one million images and their behavior and performance when adding more unlabeled data. But this should be only considered as a first attempt towards a smooth manifold structure. As a next step, we have to get away from this controlled setting because we still depend on the quality of the datasets given by the limited (although larger) amount of images and the quality of labels. Instead, we have to tap into other sources. In general, there are three possibilities: (1) combining several datasets, (2) adding synthetic data or (3) browsing the internet.

Merging different datasets is problematic because most of the available datasets have an inherent bias attached to the dataset [82, 113]. Although there are works that try to undo the damage of dataset by estimating the bias for each dataset [56], combining itself seems an unsatisfactory strategy because each dataset has different classes and the amount of images is also strongly limited. In contrast, adding synthetic data is a more promising direction but is still in a early stage of development. There are only few works that either generate new training images [63, 81], or add synthetic data points (so-called *ghost points*) in the distance space itself [18, 127]. The former approach is currently bound to certain classes such as *people* for where 3D shape models exist that ensure the generation of feasible poses and shape variations. The latter one is hard to control because the semantic meaning of these *ghost points* is not clear and it can lead to a blending of different classes. To expand approaches suggested by [81] to other classes, we have to integrate more physical constraints to guarantee feasible poses and appearances of the object. Figure 2.22 shows some images of the class *anteater*. A human does not necessarily have to know and to observe this animal to decide if a pose is likely or not. Information such as size of 1.5–1.8 m or weight of  $\approx 60$  kg might be already a good advice.

Another limitation comes with the fixed pool of unlabeled data in particular if we use existing datasets. This is similar to use the knowledge of a child for our entire life without any update. But in fact, our knowledge base will be permanently updated. That is why children regard a lost rabbit in a magic show as the reality while an adult knows that this can be only a trick. However, the internet provides us a large amount of images as well as videos and it is steadily updated with new data. A transformation of current SSL algorithms into on-line learning algorithms might be necessary [48, 89, 107] to benefit from these changes. Nevertheless, tapping into this data source poses many problems and questions: How do we get representative samples for each class out of these large amounts of data? How should we deal with



**Fig. 2.23** First examples for the query *fish* in Google images (out of 1.5 billion results) that are not representative for this class



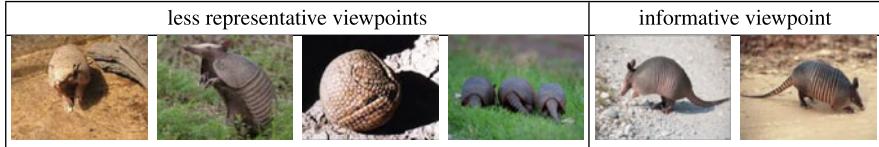
**Fig. 2.24** Most confusing classes for ETH-80 although the conditions are optimal for SSL, i.e., smooth manifold structure and no background clutter

incorrect tags? Do we get enough images for each class, for example, endangered animals/plants or deep sea fish? But even the first question is of great importance if we look at the first examples of the query *fish* in Google (Fig. 2.23) that contains drawings, a robot fish (3rd image), body paintings and other atypical examples.

#### 2.4.1.2 Image Description

Missing data is clearly not the only bottleneck that can be seen in [29] for ETH-80. This dataset is well suited for semi-supervised learning because each object is photographed from different viewpoints and there is no background clutter, occlusion, or truncation of the object. But in our experiments, we achieve at most 80 % with 5 randomly drawn labels per class and a combination of three different descriptors. Figure 2.24 shows the most confusing classes for this dataset with the corresponding binary masks, that is, tomatoes are mixed up with apples and the animals (cow, dog, horse) are confused with each other. By using also a color descriptor, we are able to distinguish green apples and red tomatoes but the final improvement is only minor.

This poses many questions. Which information do we miss? Do we need a better texture description to distinguish the different surfaces of tomato and apple? Why can a human easily guess the object class for the animals by only looking at the binary masks in the second row in Fig. 2.24? Do we describe a concept of a class in terms of proportions? It seems obvious that a better shape description is needed. But it is still not clear how to extract, to store and to use this structural description. In [95], they show that only 15 % to 30 % of an object is required to recognize 100 classes correctly independent of the orientation and the view point of the object. But many of today's shape descriptors lack on this generalization. They extract



**Fig. 2.25** Representativeness of viewpoints for the class *armadillo*: images on the *left side* show less representative viewpoints as they miss important properties of this species while the viewpoints in the *right images* are most informative for this class

often too detailed and too specific contours of an object and store this description as a template that will be later used for classification by matching. Although this is a good starting point, this approach needs too many templates to perform reasonably well. One promising direction is to use 3D information [80, 106] that allows to extract structural information about the object and to make assumptions about unseen parts in the image. In these mentioned works, they use a CAD model of an object to get this information. In general, this information is difficult to get for the most object classes that we tackle in this work. Furthermore, they consider the detection of the object also as a matching problem. But ideally we would use these rich 3D models to extract structural invariances for a set of viewpoints to get a more general description assuming that we need only a representative set of salient points to maximize the discrimination between objects [91].

Apparently, there is a similar discussion in cognitive science [49]. Supporter of the viewpoint-based model theory believe that we can extract all information from different viewpoints assuming infinite many viewpoints, i.e., templates, that would clearly exceed our brain capacity. In contrast, advocates of the structural description models argue that each object can be explained by viewpoint-independent invariances. For some classes, this assumption might be true such as *bottle* or *orange*. But for many other classes it will be difficult to find such invariances over all viewpoints, for example, the table legs are invisible in the top view. More recently, there is a common agreement that we need both templates for completeness and structural description for generalization. But this trade-off is currently missing in computer vision. Thus, we need a set of representative and most discriminative viewpoints [93] as shown for the class *armadillo* in Fig. 2.25 but we also need a more general description, for example, properties, and proportions that are invariant over a set of different viewpoints.

Another issue in our experimental setting is that we compute the image descriptor on the entire image. In fact, this is a fast and simple way of extraction but it is not clear whether this is an advantage due to the additional context information or a disadvantage because of the background clutter. But this could be analyzed by using for example part-based models [39], or foreground extraction [60].

Finally, we also miss prior knowledge and context to speed up and enhance image description similar to the work of [45] for image segmentation. [70] show that humans learn three times faster and more accurate if the features of an object were related to each other. A human learns even more although this additional knowledge is not strictly necessary for accurate performance [53]. Thus, there is obviously

a strong correlation between associations among features and final performance, for example, animals with feathers are more likely to have also wings in comparison to animals with fur. Equally important is the grouping of features or objects otherwise it would be impossible to follow a soccer match if the player do not wear an uniform. But in many applications, the raw image description is fed directly into the classifier without any intermediate steps such as grouping, ranking, or finding associations. This is rather disappointing because we cannot really reconstruct and understand what went wrong during the classification. Apart from that, some categories are almost only defined by their function, for example, *chair*. Thus, to boost the recognition of those classes, we need associations for example with human poses as shown in [25].

#### 2.4.1.3 Similarity Notion

Encouraged from the positive results of previous metric learning literature, we integrate in [29] several of those methods in the graph construction procedure. But the outcome did not meet our expectations. The main reason is that previous work almost exclusively compare their methods to the Euclidean distance (L2). In this work, we also observe a larger improvements for the L2 distance but these final numbers are lower than just applying Manhattan (L1) distance. In fact, it is almost impossible to improve L1 distance with any metric learning procedure. PCA decrease the performance of L1 and also the most supervised metric learning approaches decrease the performance or do not have any effect. Only with ITML [24], we observe a small improvement of approximately 1.5 %. But this benefit seems rather out of proportion if we consider the runtime and the tedious parameter search.

One problem is that the supervised approaches tend to over-fit due to the small amount of labeled data. [66] addresses this problem by including the geometry of the entire dataset as an additional regularization parameter. But this geometry is not updated during the learning that strongly limits the outcome of this algorithm. In principle, any change of the metric space should also cause a change in the geometry of the data. In [31], we tackled this issue by using an interleaved procedure that integrates successively unlabeled data with their highest prediction values. This method works fine for datasets with an already high graph quality. Otherwise the predictions are often incorrect so that the algorithm drifts to a worse solution. Additionally, we cannot control the label distribution leading to an unbalanced metric learning as some classes are more often requested than other classes. To further improve this approach, we have to incorporate a balancing factor and we should find a way to adjust and update the predictions.

In the long term, we require also different models and levels of granularity to express the similarity between objects. The properties and the description is completely different between base categories such as *cat* and *dog* and two species of the base category *dog*. Also [88] argue that basic level categories carry most information of a category and the categorization of objects into sub- or super-categories takes usually longer than the assignment of a base level category because super-classes ask for a generalization and sub-classes need a specification. Therefore, it is

not surprising that many learning algorithms do not improve their performance when using also a hierarchy for learning as shown in [87] because they assume always the same level of similarity description. A better approach would be to start with base level categories (mid-level of a hierarchy) and to switch the strategies when learning super- and sub-classes. The general benefit of a hierarchy should be more obvious as it allows to structure our data. Another important issue might be to integrate also relations into the similarity notion such as *larger head, more compact body, thinner legs* similar to the work of [77] that use relative attributes.

Finally, we also need a better visualization of the resulting graph structure. [11] visualized in their work a neural network to answer the questions what has the network learned and how is the knowledge represented inside this network. For graph structures, similar questions cannot be answered or only insufficiently. In [29], we look usually at the next nearest neighbors. But this is only one aspect of structure. It does not reflect the interactions in the entire graph. The shortest path between two nodes might be an interesting information. But usually this does not offer any valuable clue to the graph structure as the average shortest path length is  $\approx 2$  due to the previously mentioned *hub* nodes [119]. Also information visualization strategies such as multidimensional scaling do not produce revealing results.

#### 2.4.1.4 Supervision

In [32], we improve the quality of labels with active learning. This is a promising direction and should be always considered within semi-supervised learning due to the small amount of labels and the stronger dependency of the quality of those. However, our model, that automatically estimates the trade-off between exploration and exploitation and combines more than two criteria, has still some open issues. The trade-off is modeled with discrete states and not continuously. The feedback given by the overall entropy might be unreliable. The number of parameters is in comparison to previous work [7, 76] smaller but still to high. Finally, the initialization for this reinforcement learning is difficult and time-consuming as we start with no prior knowledge. Thus, one improvement could be the integration of domain knowledge or by using counterexamples [16].

#### 2.4.1.5 Concept-Driven vs. Exemplar-Based Learning

Graph-based algorithms are a popular choice for SSL. These algorithms reflect more or less the exemplar-based theory in cognition [57, 67, 74] assuming that humans store a list of exemplars and use a nearest neighbor approach to categorize objects. But this theory seems inconsistent as [84] and [128] show that people abstract to prototypes sometimes even without seeing those [68]. Thus, we possess a generalization ability from which the used algorithms are far away. In [31, 33], we approach this problem by combining label propagation with some prototype-based methods. Metric learning transforms the data space such that classes are more compact and in [33] we add prototypical unlabeled examples.

Although these approaches are a step in the right direction, they still miss a notion of the concept that is flexible enough to classify also unseen constellations and appearances of one object. Concepts allows us to go beyond the information given or visible [104], for example, if a human knows that an object is an apple then he also knows that there is most likely a core inside. This leads to one of the fundamental questions: “What makes a category seem coherent?” that is not yet satisfactory answered. [71] argue that similarity alone is not sufficient to describe a concept. We need also feature correlations, a structure of the attributes that are internal to a concept, and background knowledge as already discussed in the previous subsections. Beyond that we also require a relation of the concepts to each other. One possibility to get away from this purely similarity-driven approach of label propagation is to consider groups of images instead of pairwise similarities.

### 2.4.2 Beyond Human Perception, Learning, and Inference

In the previous subsection, we discussed some future work strongly based on the insight of cognitive science. This focus on human object recognition might serve as a good starting point. But also human perception and inference has their weaknesses that might be tackled by computers. One of these shortcomings is the selective attention also known as *change blindness*. There are several studies showing that a human does not recognize large differences such as a complete different clothing of a person in a video sequence of the same situation when focusing on the conversation [62]. In [102], one person is exchanged by another while the other person explains the direction without noticing the exchange. Most famous is the *invisible gorilla* [17] that runs through a video sequences and most people overlook this disguised person. But 78 % of the people are sure to recognize unexpected objects [101] that is also called *memory illusion* meaning that we have the feeling of continuous attention because we cannot remember the unconscious moments. In this point, computers are trustworthy and this is one reason why most of the assembly line work or other production steps are done by a machine. Also in computer vision we can benefit from this advantage by completely analyzing video sequences (not partially like a human) or by scanning through millions of images to find prototypical examples of one class.

Another problem comes with the limited knowledge base of a human. Even if a person learns day and night, he will never be capable to acquire the entire knowledge and experience existing in our world. Also in the case that we bound this knowledge to a particular area for example a lawyer who read all cases to his topic or a doctor who is specialized to one organ. We cannot be sure that this specialist will remember the appropriate precedent or the disease pattern if it is needed. In contrast, with a computer we are able to get more information at the same time and to remind humans on the existence of some facts, e.g. to assist the diagnosis. This ability is also in computer vision of great importance as we can acquire more and better knowledge from the internet that might be helpful for semi-supervised learning.



**Fig. 2.26** Visualization of rare categories and their effect on our inference: (a) Wolpertinger a fake object, and (b) duckbill platypus a real object that seems like a fake as it mix up properties of different species

Finally, also human inference is highly dependent on the knowledge of a person. Sure we infer quickly the position of a glass and can grasp it within few seconds and we immediately recognize the *Wolpertinger*—a bavarian mythical creature—shown in Fig. 2.26 as a fake because no hare has a dear head and bird wings. But on the other side, rare species such as the duckbill platypus (Fig. 2.26 right) looks also like an elaborate fraud to us as if someone stick the duckbill on this animal. In fact, this species comes with an unusual appearance and atypical properties for a mammal such as laying eggs like a bird or a reptile, having a tail like a beaver, a bill like a duck, and foots like an otter. Assuming that we can collect more knowledge with a computer then this added information should also improve the inference beyond that of a human. In particular in the shown case from Fig. 2.26, a computer should be in a better position to decide which one is a fake. First, each imitation of the *Wolpertinger* looks different in comparison to images of the duckbill platypus. Second, we can also take into account the trustability of the source.

## References

1. Angluin D, Laird P (1988) Learning from noisy examples. *Mach Learn* 2:343–370
2. Argyriou A, Herbster M, Pontil M (2005) Combining graph Laplacians for semi-supervised learning. In: NIPS
3. Ashby FG (1992) Multidimensional models of categorization. In: Multidimensional models of perception and cognition. Erlbaum, Hillsdale, pp 449–483
4. Ashby FG, Todd WT (2011) Human category learning 2.0. *Ann NY Acad Sci* 1224:147–161
5. Balcan M-F, Blum A (2005) A PAC-style model for learning from labeled and unlabeled data. In: COLT
6. Balcan M-f, Blum A, Pakyan Choi P, Lafferty J, Pantano B, Rwebangira MR, Zhu X (2005) Person identification in webcam images: an application of semi-supervised learning. In: ICML WS
7. Baram Y, El-yaniv R, Luz K (2004) Online choice of active learning algorithms. *J Mach Learn Res* 5:255–291
8. Bauckhage C, Thurau C (2009) Making archetypal analysis practical. In: DAGM
9. Berg TL, Forsyth DA (2006) Animals on the web. In: CVPR
10. Biederman I (1987) Recognition-by-components: a theory of human image understanding. *Psychol Rev* 94(2):115–147
11. Bischof H, Pinz A, Kropatsch WG (1992) Visualization methods for neural networks. In: IAPR

12. Blum A, Chawla S (2001) Learning from labeled and unlabeled data using graph mincuts. In: ICML
13. Buhmann JM, Zöller T (2000) Active learning for hierarchical pairwise data clustering. In: ICPR
14. Burl MC, Perona P (1996) Recognition of planar object classes. In: CVPR
15. Cebron N, Berthold MR (2009) Active learning for object classification: from exploration to exploitation. Data Min Knowl Discov 18(2):283–299
16. Cebron N, Richter F, Lienhart R (2012) “I can tell you what it’s not”: active learning from counterexamples. In: Progress in artificial intelligence
17. Chabris C, Simons D (2010) The invisible gorilla: how our intuitions deceive us. Crown Publishing Group
18. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: synthetic minority over-sampling technique. J Artif Intell Res 16:341–378
19. Cohen B, Murphy GL (1984) Models of concepts. Cogn Sci 8(1):27–58
20. Cootes TF, Edwards GJ, Taylor CJ (1998) Active appearance models. In: ECCV
21. Cutler A, Breiman L (1994) Archetypal analysis. Technometrics 36(4):338–347
22. Daitch SI, Kelner JA, Spielman DA, Haven N (2009) Fitting a graph to vector data. In: ICML
23. Damasio A (1994) Descartes’ error: emotion, reason, and the human brain. Penguin Group
24. Davis J, Kulis B, Jain P, Sra S, Dhillon I (2007) Information-theoretic metric learning. In: ICML
25. Delaitre V, Fouhey DF, Laptev I, Sivic J, Gupta A, Efros AA (2012) Scene semantics from long-term observation of people. In: ECCV
26. Delalleau O, Bengio Y, Le Roux N (2005) Efficient non-parametric function induction in semi-supervised learning. In: AISTATS
27. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L (2009) ImageNet: a large-scale hierarchical image database. In: CVPR, June 2009. IEEE
28. Dubout C, Fleuret F (2011) Tasting families of features for image classification. In: ICCV
29. Ebert S (2012) Semi-supervised learning for image classification. PhD thesis, Saarland University
30. Ebert S, Larlus D, Schiele B (2010) Extracting structures in image collections for object recognition. In: ECCV
31. Ebert S, Fritz M, Schiele B (2011) Pick your neighborhood—improving labels and neighborhood structure for label propagation. In: DAGM
32. Ebert S, Fritz M, Schiele B (2012) Active metric learning for object recognition. In: DAGM
33. Ebert S, Fritz M, Schiele B (2012) Semi-supervised learning on a budget: scaling up to large datasets. In: ACCV
34. Elhamifar E, Sapiro G, Vidal R (2012) See all by looking at a few: sparse modeling for finding representative objects. In: CVPR
35. Erickson MA, Kruschke JK (1998) Rules and exemplars in category learning. J Exp Psychol Gen 127(2):107–140
36. Everingham M, Van Gool L, Williams CK (2008) The PASCAL VOC
37. Farajtabar M, Shaban A, Reza Rabiee H, Rohban MH (2011) Manifold coarse graining for online semi-supervised learning. In: ECML
38. Fei-Fei L, Fergus R, Perona P (2006) One-shot learning of object categories. IEEE Trans Pattern Anal Mach Intell 28(4):594–611
39. Felzenszwalb PF, Girshick RB, McAllester D, Ramanan D (2010) Object detection with discriminatively trained part-based models. IEEE Trans Pattern Anal Mach Intell 32(9):1627–1645
40. Fergus R, Weiss Y, Torralba A (2009) Semi-supervised learning in gigantic image collections. In: NIPS
41. Fisher RA (1936) The use of multiple measurements in taxonomic problems. Ann Eugen 7:179–188
42. Fowlkes C, Belongie S, Chung F, Malik J (2004) Spectral grouping using the Nyström method. IEEE Trans Pattern Anal Mach Intell 26(2):214–225

43. Freeman WT (2011) Where computer vision needs help from computer science. In: ACM-SIAM symposium on discrete algorithms
44. Fritz M, Black M, Bradski G, Darrell T (2009) An additive latent feature model for transparent object recognition. In: NIPS
45. Fussennegger M, Roth PM, Bischof H, Pinz A (2006) On-line, incremental learning of a robust active shape model. *Pattern Recognit* 41(7):122–131
46. Gehler P, Nowozin S (2009) On feature combination for multiclass object classification. In: ICCV
47. Goldberg AB, Zhu X, Wright S (2007) Dissimilarity in graph-based semi-supervised classification. In: AISTATS
48. Grabner H, Leistner C, Bischof H (2008) Semi-supervised on-line boosting for robust tracking. In: ECCV
49. Hayward WG (2003) After the viewpoint debate: where next in object recognition? *Trends Cogn Sci* 7(10):425–427
50. Joachims T (1999) Transductive inference for text classification using support vector machines. In: ICML
51. Kahneman D, Tversky A (1979) Prospect theory: an analysis of decision under risk. *Econometrica* 47(2):263–291
52. Kant I (1781) Kritik der reinen Vernunft. Johann Friedrich Hartknoch Verlag. English edition: Kant I (1838) Critique of pure reason (trans: Haywood F)
53. Kaplan AS, Murphy GL (2000) Category learning with minimal prior knowledge. *J Exp Psychol* 26(4):829–846
54. Karlen M, Weston J, Erkan A, Collobert R (2008) Large scale manifold transduction. In: ICML. ACM Press, New York
55. Kato T, Kashima H, Sugiyama M (2009) Robust label propagation on multiple networks. *IEEE Trans Neural Netw* 20(1):35–44
56. Khosla A, Zhou T, Malisiewicz T, Efros AA, Torralba A (2012) Undoing the damage of dataset bias. In: ECCV
57. Kruschke JK (1992) ALCOVE: an exemplar-based connectionist model of category learning. *Psychol Rev* 99(1):22–44
58. Kulic B, Jain P, Grauman K (2009) Fast similarity search for learned metrics. *IEEE Trans Pattern Anal Mach Intell* 31(12):2143–2157
59. Lampert CH, Nickisch H, Harmeling S (2009) Learning to detect unseen object classes by between-class attribute transfer. In: CVPR
60. Lee YJ, Grauman K (2009) Foreground focus: unsupervised learning from partially matching images. *Int J Comput Vis* 85:143–166
61. Leibe B, Seemann E, Schiele B (2005) Pedestrian detection in crowded scenes. In: CVPR. IEEE
62. Levin DT, Simons DJ (1997) Failure to detect changes to attended objects in motion pictures. *Psychon Bull Rev* 4(4):501–506
63. Li W, Fritz M (2012) Recognizing materials from virtual examples. In: ECCV
64. Li Y-F, Zhou Z-H (2011) Improving semi-supervised support vector machines through unlabeled instances selection. In: AAAI
65. Liu W, He J, Chang SF (2010) Large graph construction for scalable semi-supervised learning. In: ICML, pp 1–8
66. Lu Z, Jain P, Dhillon IS (2009) Geometry-aware metric learning. In: ICML
67. Medin DL, Schaffer MM (1978) Context theory of classification learning. *Psychol Rev* 85(3):207–238
68. Minda JP, Smith JD (2001) Prototypes in category learning: the effects of category size, category structure, and stimulus complexity. *J Exp Psychol Learn Mem Cogn* 27(3):775–799
69. Murphy GL (2002) The big book of concepts
70. Murphy GL, Allopenna PD (1994) The locus of knowledge effects in concept learning. *J Exp Psychol Learn Mem Cogn* 20(4):904–919

71. Murphy GL, Medin DL (1985) The role of theories in conceptual coherence. *Psychol Rev* 92(3):289–316
72. Nguyen HT, Smeulders A (2004) Active learning using pre-clustering. In: ICML
73. Nigam K, McCallum AK, Thrun S, Mitchell T (2000) Text classification from labeled and unlabeled documents using EM. *Mach Learn* 39:103–134
74. Nosofsky RM (1984) Choice, similarity, and the context theory of classification. *J Exp Psychol* 10(1):104–114
75. Osherson DN, Smith EE (1981) On the adequacy of prototype theory as a theory of concepts. *Cognition* 9(1):35–58
76. Osugi T, Kun D, Scott S (2005) Balancing exploration and exploitation: a new algorithm for active machine learning. In: ICDM
77. Parikh D, Grauman K (2011) Relative attributes. In: ICCV, November 2011. IEEE
78. Pazzani MJ (1991) Influence of prior knowledge on concept acquisition: experimental and computational results. *J Exp Psychol* 17(3):416–432
79. Pearson K (1901) On lines and planes of closest fit to systems of points in space. *Philos Mag* 2(6):559–572
80. Pepik B, Stark M, Gehler P, Schiele B (2012) Teaching 3D geometry to deformable part models. In: CVPR
81. Pishchulin L, Jain A, Andriluka M, Thormälen T, Schiele B (2012) Articulated people detection and pose estimation: reshaping the future. In: CVPR
82. Ponce J, Berg TL, Everingham M, Forsyth DA, Hebert M, Lazebnik S, Marszalek M, Schmid C, Russell BC, Torralba A, Williams CKI, Zhang J, Zisserman A (2006) Dataset issues in object recognition. In: Ponce J, Hebert M, Schmid C, Zisserman A (eds) *Towards category-level object recognition*. LNCS. Springer, Berlin, pp 29–48
83. Pope A, Lowe DG (1996) Learning appearance models for object recognition. In: Object representation in computer vision II
84. Posner MI, Goldsmith R, Welton KE (1967) Perceived distance and the classification of distorted patterns. *J Exp Psychol* 73(1):28–38
85. Prabhakaran S, Raman S, Vogt JE, Roth V (2012) Automatic model selection in archetype analysis. In: DAGM
86. Rohban MH, Rabiee HR (2012) Supervised neighborhood graph construction for semi-supervised classification. *Pattern Recognit* 45(4):1363–1372
87. Rohrbach M, Stark M, Schiele B (2011) Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In: CVPR
88. Rosch E, Mervis CB, Gray WD, Johnson DM, Boyes-Braem P (1976) Basic objects in natural categories. *Cogn Psychol* 8:382–439
89. Saffari A, Godec M, Pock T, Leistner C, Bischof H (2010) Online multi-class LPBoost. In: CVPR, June 2010. IEEE
90. Schiele B (2000) Towards automatic extraction and modeling of objects from image sequences. In: Int sym on intelligent robotic systems
91. Schiele B, Crowley JL (1996) Where to look next and what to look for. In: IROS
92. Schiele B, Crowley JL (1997) The concept of visual classes for object classification. In: Scand conf image analysis
93. Schiele B, Crowley JL (1998) Transformation for active object recognition. In: ICCV
94. Schiele B, Crowley JL (2000) Recognition without correspondence using multidimensional receptive field histograms. *Int J Comput Vis* 36(1):31–52
95. Schiele B, Pentland A (1999) Probabilistic object recognition and localization. In: ICCV
96. Schnitzspan P, Fritz M, Roth S, Schiele B, Berkeley Eecs UC (2009) Discriminative structure learning of hierarchical representations for object detection. In: CVPR
97. Schohn G, Cohn D (2000) Less is more: active learning with support vector machines. In: ICML
98. Seeger M (2001) Learning with labeled and unlabeled data. Technical report, University of Edinburgh

99. Settles B (2009) Active Learning Literature Survey. Technical report, University of Wisconsin–Madison
100. Simon I, Snavely N, Seitz SM (2007) Scene summarization for online image collections. In: ICCV. IEEE
101. Simons DJ, Chabris CF (1999) Gorillas in our midst: sustained inattentional blindness for dynamic events. *Perception* 28(9):1059–1074
102. Simons DJ, Levin DT (1998) Failure to detect changes to people during a real-world interaction. *Psychon Bull Rev* 5(4):644–649
103. Sivic J, Russell BC, Efros AA, Zisserman A, Freeman WT (2005) Discovering object categories in image collections. In: ICCV
104. Smith E, Medin DL (1981) Categories and concepts. Harvard University Press, Cambridge
105. Sonnenburg S, Rätsch G, Schäfer C, Schölkopf B (2006) Large scale multiple kernel learning. *J Mach Learn Res* 7:1531–1565
106. Stark M, Goesele M, Schiele B (2010) Back to the future: learning shape models from 3D CAD data. In: BMVC
107. Sternig S, Roth PM, Bischof H (2012) On-line inverse multiple instance boosting for classifier grids. *Pattern Recognit Lett*, 33(7):890–897
108. Sugiyama M, Rubens N (2008) Active learning with model selection in linear regression. In: DMKD
109. Talwalkar A, Kumar S, Rowley H (2008) Large-scale manifold learning. In: CVPR, June 2008, pp 1–8
110. Tong W, Jin R (2007) Semi-supervised learning by mixed label propagation. In: AAAI, vol 22
111. Tong S, Koller D (2001) Support vector machine active learning with applications to text classification. *J Mach Learn Res* 2:45–66
112. Tong H, He J, Li M, Zhang C, Ma WY (2005) Graph based multi-modality learning. In: ACM multimedia
113. Torralba A (2011) Unbiased look at dataset bias. In: CVPR
114. Torralba BA, Russell BC, Yuen J (2010) LabelMe: online image annotation and applications. In: Proc IEEE
115. Tsang IW, Kwok JT (2006) Large-scale sparsified manifold regularization. In: NIPS
116. Tsuda K, Shin H, Schoelkopf B (2005) Fast protein classification with multiple networks. *Bioinformatics* 21:59–65
117. Vedaldi A, Gulshan V, Varma M, Zisserman A (2009) Multiple kernels for object detection. In: ICCV, pp 606–613
118. Vernon D (2005) A research roadmap of cognitive vision. Technical report, ECVision: the European research network for cognitive computer vision systems
119. Von Luxburg U, Radl A, Hein M (2010) Getting lost in space: large sample analysis of the commute distance. In: NIPS
120. Wang L, Chan KL, Zhang Z (2003) Bootstrapping SVM active learning by incorporating unlabelled images for image retrieval. In: CVPR. IEEE Comput. Soc., Los Alamitos
121. Wang X, Han TX, Yan S (2009) An HOG-LBP human detector with partial occlusion handling. In: ICCV, September 2009. IEEE
122. Wang G, Wang B, Yang X, Yu G (2012) Efficiently indexing large sparse graphs for similarity search. *IEEE Trans Knowl Data Eng* 24(3):440–451
123. Weber M, Welling M, Perona P (2000) Unsupervised learning of models for recognition. In: ECCV
124. Welinder P, Branson S, Belongie S, Perona P (2010) The multidimensional wisdom of crowds. In: NIPS, pp 1–9
125. Wiskott L, von der Malsburg C (1993) A neural system for the recognition of partially occluded objects in cluttered scenes. *Int J Pattern Recognit Artif Intell* 7(4):935–948
126. Yang L (2006) Distance metric learning: a comprehensive survey. Technical report, Michigan State University

127. Yang X, Bai X, Köknar-Tezel S, Latecki LJ (2013) Densifying distance spaces for shape and image retrieval. *J Math Imaging Vis* 46:12–28
128. Zaki SR, Nosofsky RM (2007) A high-distortion enhancement effect in the prototype-learning paradigm: dramatic effects of category learning during test. *Mem Cogn* 35(8):2088–2096
129. Zaki SR, Nosofsky RM, Stanton RD, Cohen AL (2003) Prototype and exemplar accounts of category learning and attentional allocation: a reassessment. *J Exp Psychol Learn Mem Cogn* 29(6):1160–1173
130. Zhang Z, Zha H, Zhang M (2008) Spectral methods for semi-supervised manifold learning. In: *CVPR*
131. Zhang K, Kwok JT, Parvin B (2009) Prototype vector machine for large scale semi-supervised learning. In: *ICML*. ACM Press, New York
132. Zhou D, Bousquet O, Navin Lal T, Weston J, Schölkopf B (2004) Learning with local and global consistency. In: *NIPS*
133. Zhu X, Goldberg AB, Khot T (2009) Some new directions in graph-based semi-supervised learning. In: *ICME*

# Chapter 3

## Recognizing Human Actions by Using Effective Codebooks and Tracking

**Lamberto Ballan, Lorenzo Seidenari, Giuseppe Serra, Marco Bertini,  
and Alberto Del Bimbo**

**Abstract** Recognition and classification of human actions for annotation of unconstrained video sequences has proven to be challenging because of the variations in the environment, appearance of actors, modalities in which the same action is performed by different persons, speed and duration and points of view from which the event is observed. This variability reflects in the difficulty of defining effective descriptors and deriving appropriate and effective codebooks for action categorization. In this chapter, we present a novel and effective solution to classify human actions in unconstrained videos. In the formation of the codebook, we employ radius-based clustering with soft assignment in order to create a rich vocabulary that may account for the high variability of human actions. We show that our solution scores very good performance with no need of parameter tuning. We also show that a strong reduction of computation time can be obtained by applying codebook size reduction with Deep Belief Networks with little loss of accuracy.

### 3.1 Introduction

With the continuous growth of video production and archiving, the need for automatic annotation tools that enable effective retrieval by content has accordingly gained increasing importance. In particular, action recognition is a very active re-

---

L. Ballan (✉) · L. Seidenari · G. Serra · M. Bertini · A. Del Bimbo  
Media Integration and Communication Center, University of Florence, Viale Morgagni 65, 50134  
Florence, Italy  
e-mail: [lamberto.ballan@unifi.it](mailto:lamberto.ballan@unifi.it)

L. Seidenari  
e-mail: [lorenzo.seidenari@unifi.it](mailto:lorenzo.seidenari@unifi.it)

G. Serra  
e-mail: [serra@dsi.unifi.it](mailto:serra@dsi.unifi.it)

M. Bertini  
e-mail: [marco.bertini@unifi.it](mailto:marco.bertini@unifi.it)

A. Del Bimbo  
e-mail: [alberto.delbimbo@unifi.it](mailto:alberto.delbimbo@unifi.it)

search topic with many important applications such as human-computer interaction, video indexing and video-surveillance. Existing approaches for human action recognition can be classified as using holistic or part-based information [3, 45]. Most of the holistic-based methods usually perform better in a controlled environment and are also computationally expensive due to the requirement of pre-processing the input data. Moreover, these representations can be influenced by motions of multiple objects, variations in the background and occlusions. Instead, part-based representations that exploit interest point detectors combined with robust feature descriptors, have been used very successfully for object and scene classification tasks in images [15, 62]. As a result, nowadays most video annotation solutions have exploited the bag-of-features approach to generate textual labels that represent the categories of the main and easiest to detect entities (such as objects and persons) in the video sequence [18, 48].

The definition of effective descriptors that are able to capture both spatial and temporal features has opened the possibility of recognizing dynamic concepts in video sequences. In particular, interesting results have been obtained in the definition of solutions to automatically recognize human body movements, which usually represent a relevant part of video content [38, 40, 41, 50]. However, the recognition and classification of such dynamic concepts for annotation of generic video sequences has proven to be very challenging because of the very many variations in environment, people and occurrences that may be observed. These can be caused by cluttered or moving background, camera motion and illumination changes; people may have different size, shape and posture appearance; semantically equivalent actions can manifest differently or partially, due to speed, duration or self-occlusions; the same action can be performed in different modes by different persons. This great variability on the one hand reflects in the difficulty of defining effective descriptors and on the other makes it hard to obtain a visual representation that may describe such dynamic concepts appropriately and efficiently. Furthermore, these part-based approaches usually do not attempt to localize and track actions that is necessary in video surveillance applications.

### ***3.1.1 Effective Spatio-temporal Descriptors***

Holistic descriptors of body movements have been proposed by a few authors. Among the most notable solutions, Bobick et al. [6] proposed motion history images and their low-order moments to encode short spans of motion. For each frame of the input video, the motion history image is a gray scale image that records the location of motion; recent motion results into high intensity values whereas older motion produces lower intensities. Efros et al. [14] created stabilized spatio-temporal volumes for each action video segment and extracted a smoothed dense optic flow field for each volume. They have proved that this representation is particularly suited for distant objects, where the detailed information of the appearance is not available. Yilmaz and Shah [60] used a spatio-temporal volume, built stacking object regions; descriptors encoding direction, speed and local shape of the resulting 3D surface were

generated by measuring local differential geometrical properties. Gorelick et al. [17] analyzed three-dimensional shapes induced by the silhouettes and exploited the solution to the Poisson equation to extract features, such as shape structure and orientation. Global descriptors that jointly encode shape and motion were suggested in Lin et al. [29]; Wang et al. [54] exploited global histograms of optic flow together with hidden conditional random fields. Although encoding much of the visual information, these solutions have shown to be highly sensitive to occlusions, noise and change in viewpoint. Most of them have also proven to be computationally expensive due to the fact that some preprocessing of the input data is needed, such as background subtraction, segmentation and object tracking. All these aspects make these solutions only suited for representation of body movements in videos taken in controlled contexts.

Local descriptors have shown better performance and are in principle better suited for videos taken in both constrained and unconstrained contexts. They are less sensitive to partial occlusions and clutter and overcome some of the limitations of the holistic models, such as the need of background subtraction and target tracking. In this approach, local patches at spatio-temporal interest points are used to extract robust descriptors of local moving parts and the bag-of-features approach is employed to have distinctive representations of body movements. Laptev [26] and Dollár [13] approaches have been among the first solutions. Laptev [26, 43] proposed an extension to the Harris–Förstner corner detector for the spatio-temporal case; interesting parts were extracted from voxels surrounding local maxima of spatio-temporal corners, that is, locations of videos that exhibit strong variations of intensity both in spatial and temporal directions. The extension of the scale-space theory to the temporal dimension permitted to define a method for automatic scale-selection. Dollár et al. [13] proposed a different descriptor than Laptev's, by looking for locally periodic motion. While this method produces a denser sampling of the spatio-temporal volume, it does not provide automatic scale-selection. Despite of it, experimental results have shown that it improves with respect to [43].

Following these works, other authors have extended the definition of local interest point detectors and descriptors to incorporate time or combined static local features with other descriptors so to model the temporal evolution of local patches. Sun et al. [49] have fused spatio-temporal SIFT points with holistic features based on Zernike moments. In [56], Willems et al. extended SURF feature to time and defined a new scale-invariant spatio-temporal detector and descriptor that showed high efficiency. Scovanner et al. [44], have proposed to use grouping of 3D SIFT, based on co-occurrence, to represent actions. Kläser et al. [23] have proposed a descriptor based on histograms of oriented 3D gradients, quantized using platonic solids. Gao et al. [16] presented MoSIFT, an approach that extend the SIFT algorithm to find visually distinctive elements in the spatial domain. It detects spatio-temporal points with a high amount of optical flow around the distinctive points motion constraints. More recently, Laptev et al. [27] proposed a structural representation based on dense temporal and spatial scale sampling, inspired by the spatial pyramid approach of [28] with interesting classification results in generic video scenes. Kovashka et al. [25] extended this work by defining a hierarchy of discriminative neighborhoods instead of using spatio-temporal pyramids. Liu et al. [32] combined

MSER and Harris-Affine [37] regions with Dollár’s space-time features and used AdaBoost to classify YouTube videos. Shao et al. [46] applied transformation based techniques (i.e., Discrete Fourier Transform, Discrete Cosine Transform and Discrete Wavelet Transform) on the local patches and used the transformed coefficients as descriptors. Yu et al. [61] presented good results using the Dollár’s descriptor and random forest-based template matching. Niebles et al. [39] trained an unsupervised probabilistic topic model using the same spatio-temporal features, while Cao et al. [8] suggested to perform model adaptation in order to reduce the amount of labeled data needed to detect actions in videos of uncontrolled scenes. Comparative evaluations of the performance of the most notable approaches were recently reported by Wang et al. [55] and Shao et al. [45].

### **3.1.2 Suitable Visual Codebooks**

According to the bag-of-features model actions are defined as sets of codewords obtained from the clustering of local spatio-temporal descriptors. Most of the methods have used the k-means algorithm for clustering because of its simplicity and speed of convergence [15, 21, 39, 47]. However, both the intrinsic weakness of k-means to outliers and the need of some empirical pre-evaluation of the number of clusters hardly fit with the nature of the problem at hand. Moreover, with k-means the fact that cluster centers are selected almost exclusively around the most dense regions in the descriptor space results into ineffective codewords of action primitives. To overcome the limitations of the basic approach, Liu et al. [30] suggested a method to automatically find the optimal number of visual word clusters through maximization of mutual information (MMI) between words and actions. MMI clustering is used after k-means to discover a compact representation from the initial codebook of words. They showed some performance improvement. Recently Kong et al. [24] have proposed a framework that unifies reduction of descriptor dimensionality and codebook creation, to learn compact codebooks for action recognition optimizing class separability. Differently, Uemura and Mikolajczyk [35] explored the idea of using a large number of features represented in many vocabulary trees instead of a single flat vocabulary. Yao et al. [59] recently proposed a similar framework using a training procedure based on a Hough voting forest. Both these methods require higher efforts in the training phase.

### **3.1.3 Our Contribution**

In this chapter, we propose a novel and effective solution to classify human actions in unconstrained videos. It improves on previous contributions in the literature through the definition of a novel local descriptor and the adoption of a more effective solution for the codebook formation. We use image gradient and optic flow to respectively model the appearance and motion of human actions at regions in the neighborhood of local interest points and consider multiple spatial and temporal

scales. These two descriptors are used in combination to model local features of human actions and activities. Unlike similar related works [23, 44], no parameter tuning is required.

In the formation of the codebook, we recognize that the clusters of spatio-temporal descriptors should be both in a sufficiently large number and sufficiently distinguished from each other so to represent the augmented variability of dynamic content with respect to the static case. To this end, radius-based clustering [22] with soft assignment has been used. In fact, with radius-based clustering cluster centers are allocated at the modes corresponding to the maximal density regions, so resulting into a statistics of the codewords that better fits with the variability of human actions with respect to k-means clustering. To obtain a precise spatio-temporal localization of each action, the detected spatio-temporal points are associated to each person, present in the scene, by a particle filter visual tracker. Experiments carried on standard datasets show that the approach followed outperforms the current state of the art methods. To avoid too large codebooks, we performed codebook compression with Deep Belief Networks. The solution proposed shows good accuracy even with very small codebooks. Finally, we provide several experiments on the Hollywood2 dataset [34] and on a new surveillance dataset (MICC-Surveillance), to demonstrate the effectiveness and generality of our method for action recognition in unconstrained video domains.

The rest of the chapter is organized as follows.<sup>1</sup> The full framework of the proposed solution is shown in Sect. 3.2, while the spatio-temporal features are presented in Sect. 3.3. Action representation and categorization is presented in Sect. 3.4. The experimental results, with an extensive comparison with the state-of-the-art approaches, are hence discussed in Sect. 3.5. Here we also included experiments on unconstrained videos to demonstrate the effectiveness of the approach also in this case. Conclusions are drawn in Sect. 3.6.

## 3.2 Action Classification Architecture

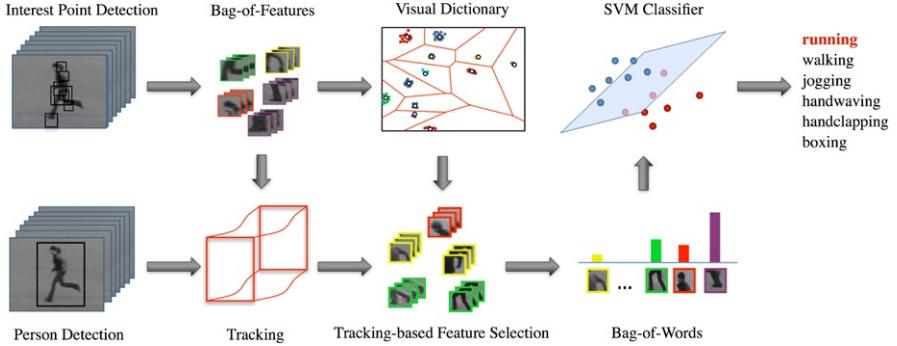
### 3.2.1 Visual Dictionary Formation

The architectural design of the proposed solution, based on an effective bag-of-features model, is shown in Fig. 3.1. The basic idea of the bag-of-features approach is to represent visual content as an unordered collection of “visual words”. To this end, it is necessary to define a visual dictionary from the local features extracted in the video sequences, performing a quantization of the original feature space. The descriptors used to represent the spatio-temporal interest points are presented in the following Sect. 3.3.

The visual dictionary (codebook) is generated by clustering of a set of local descriptors and each cluster is treated as a visual word. Typically it is used the k-means

---

<sup>1</sup>Please note that an earlier version of this work has recently appeared in *IEEE Transactions on Multimedia* [4].



**Fig. 3.1** The proposed solution architecture

algorithm because of its simplicity and convergence speed. However, it has been shown that using this algorithm the cluster centers tend to coalesce around the denser regions of the feature space, thus not describing other informative regions. This issue is particularly important in the densely sampled space of the spatio-temporal features used in our approach. In the work of Jurie and Triggs [22] it has been shown that a different approach, namely radius-based clustering, is able to generate better visual dictionaries for the images that arise in natural scenes. We have therefore used a radius-based clustering technique, following a mean-shift approach [11], that improves the performance of the system over k-means. This issue is presented in detail in Sect. 3.4.

### 3.2.2 Person Tracking and Data Association

Person tracking is used to assign the detected spatio-temporal interest points to each person present in a video, to localize both in space and time each recognized action. The tracker adopted in our system implements a particle filter based tracking algorithm, presented by [2], that tracks position, size and speed of the target, describing the target appearance with its color histogram (using hue and saturation channels). The tracker is initiated using the human detector of [12], implemented in OpenCV. The detector is run frame-wise to obtain both new targets to follow and measures for existing tracks. Measures obtained from the people detector are associated to targets by solving a data association problem, using a fast greedy algorithm that has a much lower complexity than the optimal solution obtainable with the Hungarian algorithm [58]. This greedy algorithm can be executed in real-time, as needed in video-surveillance applications, and works as follows: a matrix  $M$  that contains all the matching scores  $m_{i,j}$  between the  $i$ th target and the  $j$ th measure of the person detector is computed. The matching score is computed as:

$$m_{i,j} = e^{-\frac{d_{i,j}^2}{D}} \quad (3.1)$$

where  $d_{i,j}$  is the Euclidean distance between the static part of the model (position and size) of the target and the position and size of the detected person (represented using top-left and bottom-right coordinates of the bounding boxes) and  $D$  is adaptively chosen based on the target size.

The maximum  $m_{i,j}$  are iteratively selected, and the  $i$  rows and  $j$  columns belonging to target and detector in  $M$  are deleted. This is repeated until no further valid  $m_{i,j}$  is available. To avoid the erroneous association of a detection to a target two approaches are followed: (i) only the associated detections with a matching score  $m_{i,j}$  above a threshold are used, to avoid that a detection that is far from a target is matched; (ii) if a detection overlaps more than one target no association is performed. If a detection is not associated to any target and does not overlap any existing target then it is used to start a new track.

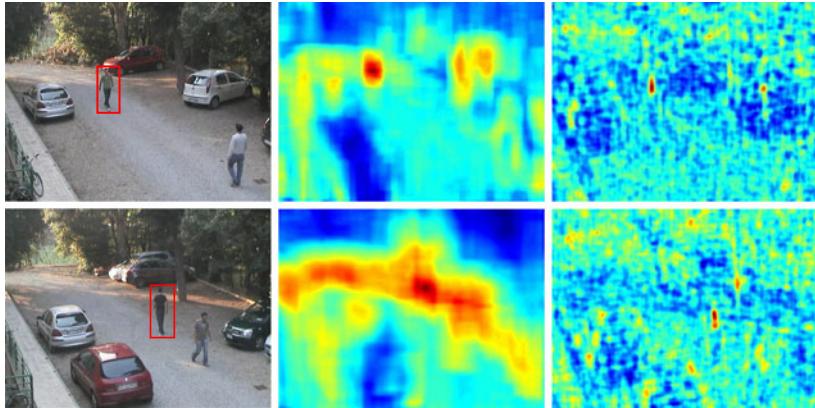
The template of the target appearance is updated every time a new detection is associated to the track. In this way, we prevent template drift and we allow the color histogram to adapt with respect to illumination changes and maneuvers which can change the target appearance. The state update equation, defined over the 8-dimensional state vector  $x_k$  (composed by 4 components for position and size and 4 components for their velocities), realizes a 1st-order dynamic model:

$$x_k = Ax_{k-1} + v_{k-1}, \quad A = \begin{bmatrix} I_4 & I_4\Delta t \\ 0 & I_4 \end{bmatrix} \quad (3.2)$$

where  $I_4$  is an  $4 \times 4$  identity matrix,  $\Delta t$  is the time step and  $v_{k-1}$  is an additive, zero mean, isotropic Gaussian uncertainty term that represents the uncertainty in the state update. This uncertainty is parametrized in terms of the standard deviation on each component of the state vector. The measurement model exploits the results of the person detector whenever they are available.

The person detector likelihood is strongly peaked in presence of a target, as shown in the third column of Fig. 3.2. This behavior allows to detect as distinct objects even very close pedestrians, but is not suitable to use it as likelihood of the target [1] since in particle weight computation it could assign very high weights to a few or no particles, and almost uniform low weights to the remaining population, leading thus to a degeneracy problem. To deal with this issue, the target model of the particle filter is based on the color histogram of the tracked object, aiming at robustness against non-rigidity, rotation and partial occlusion; after updating the template histogram with the new measure histogram, weights are computed according to the Bhattacharya distance between the particle and the template histograms. On the other hand the color histogram is too weak to be used as an aspect model in a real-world video-surveillance scenario and should not be used as a sole measurement provider, as shown in the second column of Fig. 3.2; this is due to background pixels contaminating the template and the lack of discriminativity of the histogram caused also by its subsampling (we used eight hue bins and eight saturation bins, to reduce sensitivity to light conditions).

To improve the particle filter capability to effectively track the target, even if its appearance is not strongly characterized, the tracking method implements a particular technique, based on the use of the similarity of the current estimate with the orig-



**Fig. 3.2** Original frame, hue/saturation histogram and person detector generated likelihood computed for the farthest target (highlighted with *red bounding box*). In this example the pedestrian detector is run at a single scale; histogram likelihood is generated using the values of the Batthacharya distance between the template histogram and a corresponding (same scale and aspect ratio) window. In both cases scale and aspect ratio variations are not considered, for the sake of visualization

inal target histogram as an index of tracking quality, to manage the uncertainty in the state update equation by means of on-line adaptation of the error  $v_{k-1}$ . In particular, let us consider the case where the variances of position and size of the target are set to very high values. In this case the filter samples over a wide enough area to maximize the possibility of capturing the target in case of erratic changes in direction or velocity. The pitfall in this strategy, however, is that it also increases the likelihood that the particle filter will become distracted by spurious similar patches in the background. Considering also the variances of the velocities the problem is even worse: from Eq. 3.2, in the update rule for propagating a particle from time  $k-1$  to  $k$ , the uncertainty in the dynamic component is propagated to the static component. To reduce this effect, a *blindness* value is computed by passing the similarity of estimate and original target histogram through a sigmoid; this *blindness* value is used to adjust the variances in such a way that the noise in the static component of the state observations is never amplified by the noise in the dynamic components. This allows the tracker to switch between two different behaviors: one that relies on the predicted motion of the target and one that behaves like a random-walk model.

### 3.2.3 Action Classification and Track Annotation

By mapping the features associated to each tracked person in a video to the vocabulary, we can represent it by the frequency histogram of visual words. In order to reduce outliers, histograms of tracks that contain too few interest points, are discarded. Then, the remaining histograms are fed to a classifier to predict the action category. In particular, classification is performed using non-linear SVMs with the  $\chi^2$  kernel.



**Fig. 3.3** Example of multiple person tracking, spatio-temporal interest point detection and their association to the tracks

To perform multi-class classification we use the *one-vs.-one* approach. To this end we train a binary SVM classifier for each pair of action classes for a total of  $\frac{n(n-1)}{2}$  classifiers. Action is predicted considering the output of each SVM as a vote for the correspondent action and using a majority voting procedure. Figure 3.3 shows an example of the tracker results and features association.

### 3.3 Fusing Spatio-temporal Local Descriptors of Appearance and Motion

Spatio-temporal interest points are detected at video local maxima of the Dollár's detector [13] applied over a set of spatial and temporal scales. Using multiple scales is fundamental to capture the essence of human activity. To this end, linear filters are separately applied to the spatial and temporal dimension: on the one hand, the spatial scale permits to detect visual features of high and low detail; on the other, the temporal scale allows to detect *action primitives* at different temporal resolutions. The filter response function is defined as:

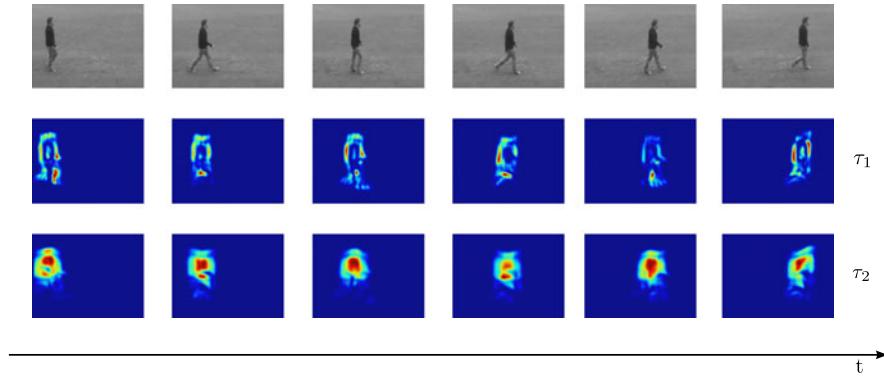
$$R = (I * g_\sigma * h_{ev})^2 + (I * g_\sigma * h_{od})^2 \quad (3.3)$$

where  $I(x, y, t)$  is the image sequence,  $g_\sigma(x, y)$  is a spatial Gaussian filter with scale  $\sigma$ ,  $h_{ev}$  and  $h_{od}$  are a quadrature pair of 1D Gabor filters that provide a strong response to temporal intensity changes for periodic motion patterns, respectively defined as:

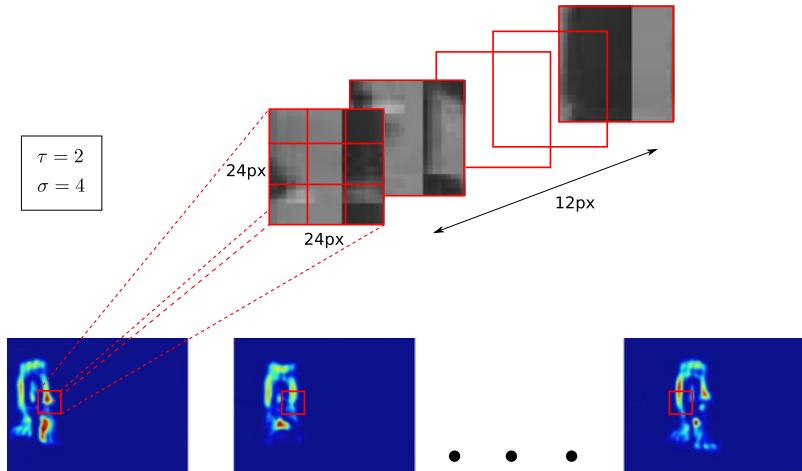
$$h_{ev}(t; \tau, \omega) = -\cos(2\pi t\omega)e^{-t^2/\tau^2} \quad (3.4)$$

$$h_{od}(t; \tau, \omega) = -\sin(2\pi t\omega)e^{-t^2/\tau^2} \quad (3.5)$$

where  $\omega = 4/\tau$ . In the experiments we used  $\sigma = \{2, 4\}$  as spatial scales and  $\tau = \{2, 4\}$  as temporal scales. Figure 3.4 shows an example of temporal scaling of human body parts activity during walking: torso has high response at high temporal scale, while limbs respond at the lower scale.



**Fig. 3.4** Response of the spatio-temporal interest point detector at two temporal scales  $\tau_1 < \tau_2$  (low response in blue, high response in red); *first row*: original video frames, *second row*: detector response at temporal scale  $\tau_1$  (mostly due to motion of human limbs); *third row*: detector response temporal scale  $\tau_2$  (mostly due to motion of human torso)



**Fig. 3.5** Three-dimensional region at the spatio-temporal interest point corresponding to a swinging arm

Three-dimensional regions of size proportional to the detector scale (6x) are considered at each spatio-temporal interest point, and divided into equally sized sub-regions (three for each spatial dimensions along the  $x$  and  $y$ , and two for the temporal dimension  $t$ ), as shown in Fig. 3.5.

For each sub-region, image gradients on  $x$ ,  $y$  and  $t$  are computed as:

$$G_x = I(x+1, y, t) - I(x-1, y, t) \quad (3.6)$$

$$G_y = I(x, y+1, t) - I(x, y-1, t) \quad (3.7)$$

$$G_t = I(x, y, t+1) - I(x, y, t-1) \quad (3.8)$$

**Table 3.1** Comparison of accuracy and efficiency of our H3DGrad with other gradient based descriptors on KTH and Weizmann datasets. Computation time for a single descriptor measured on a 2.66 GHz Intel Xeon with 12 GB RAM; H3DGrad, [23] and [27] are C++ implementations while [44] is a MATLAB implementation

Descriptor	KTH	Weizmann	Time (ms)
H3DGrad	90.38	92.30	1
Kläser et al. [23]	91.40	84.30	2
Laptev et al. [27]	81.60	–	12
Scovanner et al. [44]	–	82.60	419

and the optic flow with relative apparent velocity  $V_x$ ,  $V_y$  is estimated according to [33].

Orientations of gradients and optical flow are computed for each pixel as:

$$\phi = \tan^{-1}\left(G_t / \sqrt{G_x^2 + G_y^2}\right) \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \quad (3.9)$$

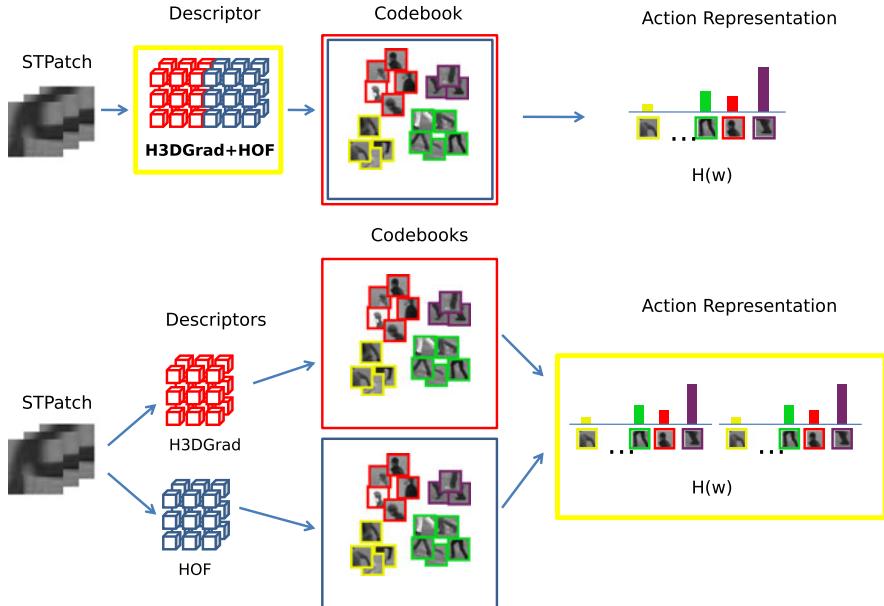
$$\theta = \tan^{-1}(G_y / G_x) \in [-\pi, \pi] \quad (3.10)$$

$$\psi = \tan^{-1}(V_y / V_x) \in [-\pi, \pi] \quad (3.11)$$

where  $\phi$  and the  $\theta$  are quantized in four and eight bins, respectively.

The local descriptor obtained by concatenating  $\phi$  and  $\theta$  histograms (H3DGrad) has therefore size  $3 \times 3 \times 2 \times (8 + 4) = 216$ . There is no need to reorient the 3D neighborhood, since rotational invariance, typically required in object detection and recognition, is not desirable in the action classification context. This approach is much simpler to compute than those proposed in [44] and [23]. In particular, in [44] the histogram is normalized by the solid angle value to avoid distortions due to the polar coordinate representation (instead of quantizing separately the two orientations as in our approach), moreover the size of the descriptor is 2048; in [23] the 3D gradient vector is projected on the faces of a platonic solid. In this latter approach requires additional parameter tuning, to optimize the selection of the solid used for the histogram computation and whether to consider the orientations of its faces or not. Differently from [27] our 12-bin H3DGrad descriptor models the dynamic appearance of the three-dimensional region used for its computation, instead of being a 4-bin 2D histogram cumulated over time. A comparison between our H3DGrad descriptor and the other HOG features (i.e., [23, 27, 44]) is reported in Table 3.1, in terms of both accuracy and feature computation time.

The  $\psi$  is quantized in eight bins with an extra “no-motion” bin added to improve performance. The local descriptor of  $\psi$  (HOF) has size  $3 \times 3 \times 2 \times (8 + 1) = 162$ . Histograms of  $\phi$ ,  $\theta$  and  $\psi$  are respectively, derived by weighting pixel contributions respectively, with the gradient magnitude  $M_G = \sqrt{G_x^2 + G_y^2 + G_t^2}$  (for  $\phi$  and  $\theta$ ), and the optic flow magnitude  $M_O = \sqrt{V_x^2 + V_y^2}$  (for  $\psi$ ).



**Fig. 3.6** Two fusion strategies: early-fusion (at the descriptor level) and late-fusion (at the codebook level)

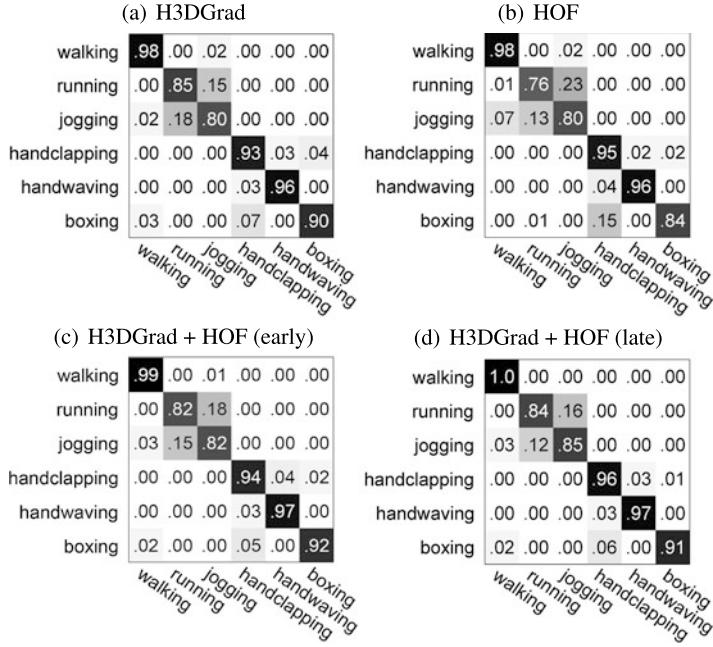
In order to obtain an effective codebook for human actions these two descriptors can be combined according to either early or late fusion. In the former case, the two descriptors are first concatenated and the combined descriptor is hence used for the definition of the human action codebook. In the latter, a codebook is obtained from each descriptor separately; then the histograms of codewords are concatenated to form the representation (see Fig. 3.6).

Figure 3.7 shows the classification accuracy measured with the KTH dataset, using codebooks based on the H3DGrad descriptor (a), HOF descriptor (b), and early (c) and late fusion (d), with 4000 codewords. Each action, is represented by an histogram  $H$  of codewords  $w$  obtained according to k-means clustering with hard assignment:

$$H(w) = \frac{1}{n} \sum_{i=1}^n \begin{cases} 1 & \text{if } w = \operatorname{argmin}_{v \in V}(D(v, f_i)); \\ 0 & \text{otherwise;} \end{cases} \quad (3.12)$$

where  $n$  is the number of the spatio-temporal features,  $f_i$  is the  $i$ th spatio-temporal feature, and  $D(v, f_i)$  is the Euclidean distance between the codeword  $v$  of the vocabulary  $V$  and  $f_i$ .

We present in Table 3.2 the average accuracy obtained by H3DGrad and HOF respectively, and by the early and late fusion. From the figures, it appears clearly that late fusion provides the best performance. This can be explained with the fact that H3DGrad and HOF descriptors have quite complementary roles (for example



**Fig. 3.7** Classification accuracy on the KTH dataset using k-means clustering, hard assignment and different descriptors combination strategies (i.e. early or late fusion)

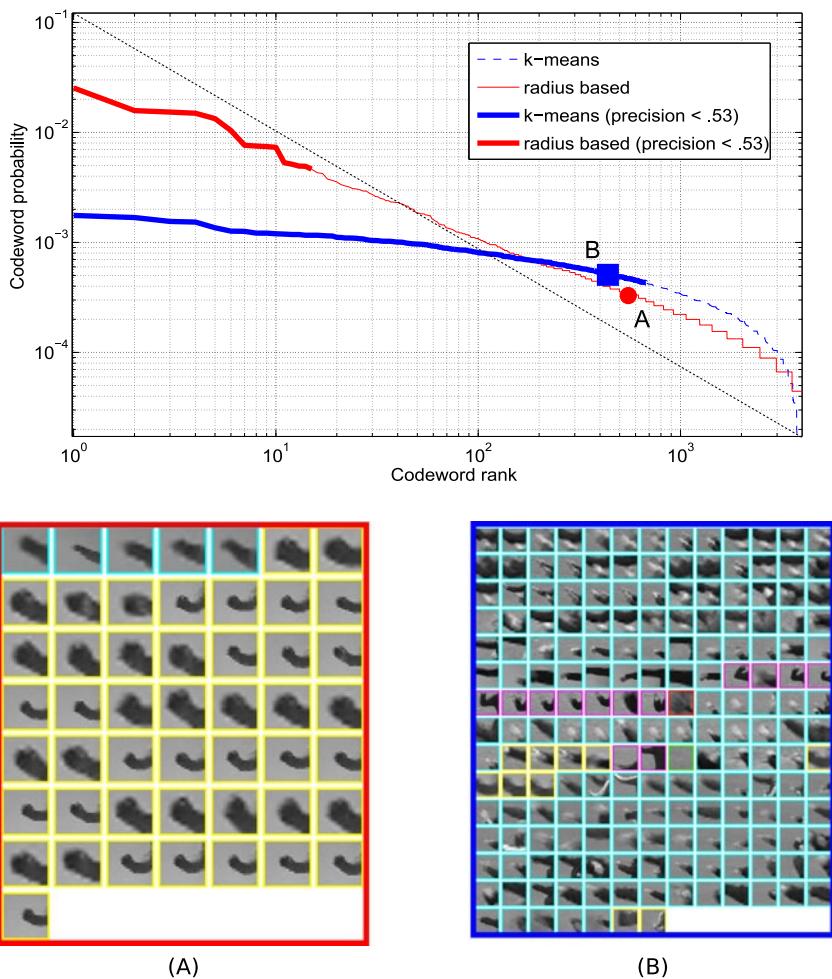
**Table 3.2** Average class accuracy of our descriptors, alone and combined, on the KTH and Weizmann datasets

Descriptor	KTH	Weizmann
H3DGrad	90.38	92.30
HOF	88.04	89.74
H3DGrad + HOF (early fusion)	91.09	92.38
H3DGrad + HOF (late fusion)	92.10	92.41

the *boxing* action is better recognized when using H3DGrad descriptor while *handclapping* action is better recognized by HOF, as shown in Fig. 3.7(a), (b)). Late fusion improves recognition performance for all the classes except one. A similar behavior was observed with the Weizmann dataset, although in this case the improvement was not so significant mainly due to the limited size and intra-class variability of the dataset (see Table 3.2).

### 3.4 Action Representation and Classification

In order to improve with respect to k-means and to account for the high variability of human actions in terms of appearance or motion, we used radius-based clustering for codebook formation.



**Fig. 3.8** Log–log plots of codeword frequency using k-means and radius-based clustering with hard assignment. **Bold lines** indicate regions where the average cluster precision [36] is below 0.53. The **dotted diagonal line** represents the Zipfian distribution. Two sample clusters are shown at near frequencies, respectively obtained with radius-based clustering (A) (most of the features in the cluster represent spatio-temporal patches of the same action) and with k-means (B) (features in the cluster represent patches of several actions). Patches of actions have different colors: *boxing* (cyan), *hand-waving* (magenta), *hand-clapping* (yellow), *running* (green), *walking* (red), *jogging* (blue)

Figure 3.8 shows the codeword frequency of radius-based clustering and k-means with hard quantization on the KTH dataset. It is interesting to note that with k-means most of the codewords have similar probability of occurrence, so making it difficult to identify a set of words that have at the same time high discrimination capability and good probability of occurrence. In contrast radius-based shows a much less uni-

form frequency distribution. Interestingly, with radius-based clustering, the codeword distribution of the human action vocabulary is similar to the Zipf's law for textual corpuses. It seems therefore reasonable to assume that codewords at intermediate frequencies are the most informative also for human action classification, and the best candidates for the formation of the codebook.

Due to the high dimensionality of the descriptor, codebooks for human actions usually have cluster centers that are spread in the feature space, so that two or more codewords are equally relevant for a feature point (codeword *uncertainty*); moreover cluster centers are often too far from feature points so that they are not anymore representative (codeword *plausibility*). With radius-based clustering, codeword *uncertainty* is critical because it frequently happens that feature points are close to the codewords boundaries [52]. Instead, codeword *plausibility* is naturally relaxed due to the fact that clusters are more uniformly distributed in the feature space. To reduce the *uncertainty* in codeword assignment, we therefore performed radius-based clustering with soft assignment by Gaussian kernel density estimation smoothing. In this case, the histogram  $H$  is computed as:

$$H(w) = \frac{1}{n} \sum_{i=1}^n \frac{K_\sigma(w, f_i)}{\sum_{j=1}^{|V|} K_\sigma(v_j, f_i)} \quad (3.13)$$

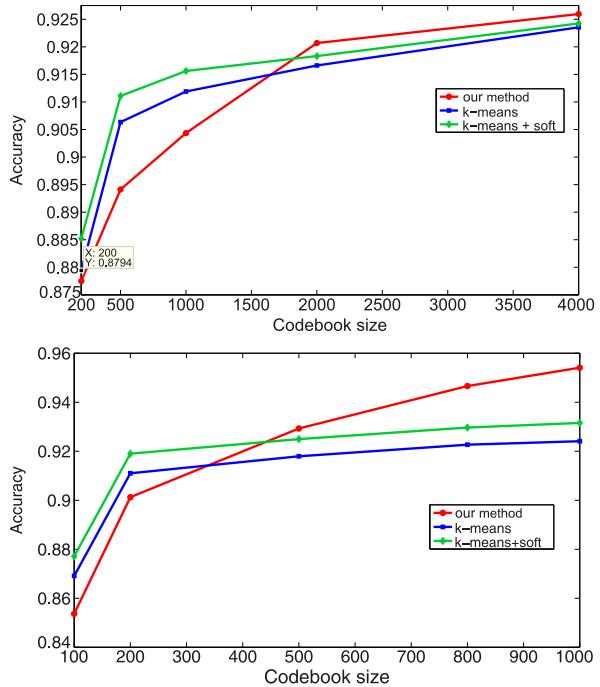
where  $K_\sigma$  is the Gaussian kernel:  $K_\sigma(\cdot, \cdot) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{d(\cdot, \cdot)^2}{2\sigma^2}}$  being  $\sigma$  the scale parameter tuned on the training set, and  $d(\cdot, \cdot)$  is the Euclidean distance.

Figure 3.9 compares the classification accuracy with codebooks obtained with k-means clustering with both hard and soft assignment, and radius-based clustering with soft assignment, respectively, for the KTH and Weizmann dataset. The plots have been obtained by progressively adding less frequent codewords to the codebooks (respectively up to 4000 and 1000 codewords for the two datasets). The performance of k-means is improved by the use of soft assignment. With a small number of words radius-based clustering with soft assignment has lower performance than k-means due to the fact that the codewords used have higher frequency than those used by k-means (see Fig. 3.8). As the number of codewords in the codebook increases, radius-based clustering outperforms k-means, whether with hard or soft assignment. This reflects the fact that in this case radius-based clustering permits to have also sparse regions being represented in the codebook. Besides, soft assignment helps to reduce *uncertainty* in the dense regions. Figure 3.10 shows the confusion matrix for different human actions on KTH and Weizmann datasets with radius-based soft assignment. The average accuracy is respectively, 92.66 % and 95.41 % for the two datasets.

### 3.5 Experimental Results

We have assessed our approach for categorization of human actions in different conditions. Particularly, it has been tested on the KTH and Weizmann datasets that

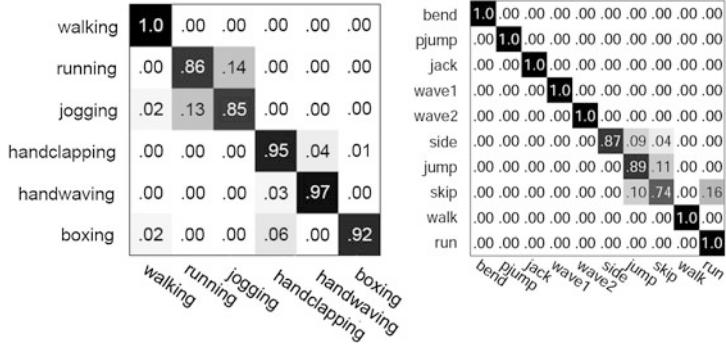
**Fig. 3.9** Classification accuracy on KTH (*top*) and Weizmann (*bottom*) datasets with codebooks created with k-means with hard assignment, k-means with soft assignment and radius-based with soft assignment



show staged actions performed by an individual in a constrained non-cluttered environment. Moreover, in order to have a more complete assessment of the performance of the proposed solution even in real world scenes with high variability and unconstrained videos, we also carried out experiments on the Hollywood2 and MICC-UNIFI Surveillance datasets. This latter, made publicly available at [www.openvisor.org](http://www.openvisor.org) [53], includes real world video surveillance sequences containing actions performed by individuals with cluttering and varying filming conditions. Experiments were performed using non-linear SVMs with the  $\chi^2$  kernel [62].

### 3.5.1 Experiments on KTH and Weizmann Datasets

The KTH dataset, currently the most common dataset used for the evaluations of action recognition methods [55], contains 2391 short video sequences showing six basic actions: *walking*, *running*, *jogging*, *hand-clapping*, *hand-waving*, *boxing*. They are performed by 25 actors under four different scenarios with illumination, appearance and scale changes. They have been filmed with a hand-held camera at  $160 \times 120$  pixel resolution. The Weizmann dataset contains 93 short video sequences showing nine different persons, each performing ten actions: *run*, *walk*, *skip*, *jumping-jack*, *jump-forward-on-two-legs*, *jump-in-place-on-two-legs*, *gallop-sideways*, *wave-two-hands*, *wave-one-hand* and *bend*. They have been filmed with a fixed camera, at  $180 \times 144$  pixel resolution, under the same lighting condition.



**Fig. 3.10** Classification accuracy on KTH (*left*) and Weizmann (*right*) datasets using radius-based clustering with soft assignment

**Table 3.3** Comparison of classification accuracy with some state-of-the-art methods on KTH and Weizmann datasets

Method	KTH	Weizmann	Features	Optimizations
<i>Our method</i>	92.66	95.41	H3DGrad + HOF	—
Yu et al. [61]	91.8	—	HoG + HOF	—
Wang et al. [55]	92.1	—	HOF	—
Gao et al. [16]	91.14	—	MoSIFT	—
Sun et al. [49]	89.8	90.3	2D SIFT + 3D SIFT + Zernike	—
Rapantzikos et al. [42]	88.3	—	PCA-Gradient	—
Laptev et al. [27]	91.8	—	HoG + HOF	codebook, sampling
Dollár et al. [13]	81.2	—	PCA-Gradient	—
Wong and Cipolla [57]	86.62	—	PCA-Gradient	—
Scovanner et al. [44]	—	82.6	3D SIFT	codebook
Niebles et al. [39]	83.33	90	PCA-Gradient	—
Liu et al. [31]	—	90.4	PCA-Gradient + Spin images	codebook
Kläser et al. [23]	91.4	84.3	3D HoG	descriptor
Willem et al. [56]	84.26	—	3D SURF	—
Schüldt et al. [43]	71.7	—	ST-Jets	—

Table 3.3 reports the average accuracy of our method in comparison with the most notable research results published in the literature. The performance figures reported are those published in their respective papers. For a fair comparison, our experiments have been performed with the setup suggested by the creators of the KTH and Weizmann datasets [17, 43], that has been used in [16, 23, 27, 31, 42–44, 49, 55–57, 61]. In particular, with the KTH dataset, SVM classifiers have been trained on sequences of 16 actors and performance was evaluated for the sequences of the remaining 9 actors according to 5-fold cross-validation. With the Weizmann

dataset, SVM classifiers have been trained on the videos of 8 actors and tested on the one remaining, following leave-one-out cross-validation.

While showing the best performance, our solution has also the nice property that it does not require any adaptation to the context under observation. Instead other solutions require some tuning of the descriptor to the specific context. Namely, Laptev et al. [27] perform different spatio-temporal sampling of video frames and define a set of descriptors; hence they represent each action with the best combination of sampling and descriptors; Kläser et al. [23] use a parameterized 3D gradient descriptor; parameter values are optimized for the dataset used; Liu et al. [31] use both local and global descriptors and select the best combination of them according to an optimization procedure; Scovanner et al. [44] optimize the codebook by associating co-occurrent visual words.

Other researchers have claimed higher performance on the KTH dataset: 93.17 % Breganzio et al. [7]; 94.2 % Liu and Shah [30]; 93.43 % Lin et al. [29]; 95.83 % Chen et al. [10]. However, these results were obtained with a Leave-One-Out Cross-Validation setting that uses more training data and therefore are not directly comparable. For the sake of fairness, they have not been included in Table 3.3. An exhaustive list of the different experimental setups and results has been recently published by Gao et al. [16].

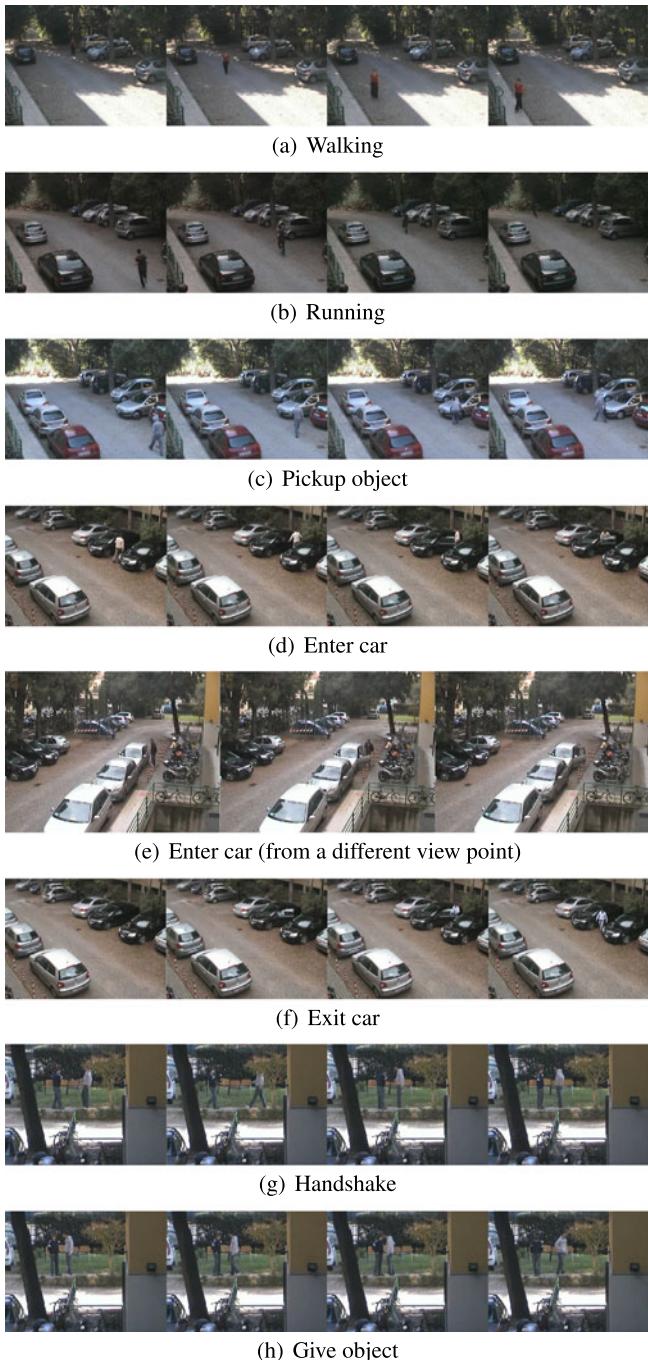
### 3.5.2 Experiments on MICC-UNIFI Surveillance Dataset

The MICC-UNIFI Surveillance dataset is composed by 175 real world video sequences of human actions with durations ranging from 3 to 20 seconds. The videos have been taken from wall mounted Sony SNC RZ30 cameras at  $640 \times 480$  pixel resolution, in a parking lot. The scenes are captured from different viewpoints, at different degrees of zooming, with different shadowing and unpredictable occlusions, at different duration, speed and illumination conditions. Eight subjects perform seven everyday actions: *walking*, *running*, *pickup object*, *enter car*, *exit car*, *handshake* and *give object*. A few examples are shown in Fig. 3.11. We followed a repeated stratified random sub-sampling validation, using 80 % of the videos of each class as training set. Experiments were performed using a 2000 codeword codebook. The confusion matrix of classification accuracy is reported in Fig. 3.12: the average accuracy is 86.28 %. Most of the misclassifications observed with our method occurred with the *give object* and *handshake* actions. They are both characterized by a very fast motion pattern and small motion of the human limbs. Figure 3.13 reports sample sequences of these actions with evidence of details. In Table 3.4, we report a comparison of our method with other codebook creation approaches (k-means with hard and soft assignment) and with other state-of-the-art descriptors that publicly make their implementation available: MoSIFT<sup>2</sup> [16] and Dollár et al.<sup>3</sup> [13]. The results show that the proposed method outperforms the other approaches, and that

---

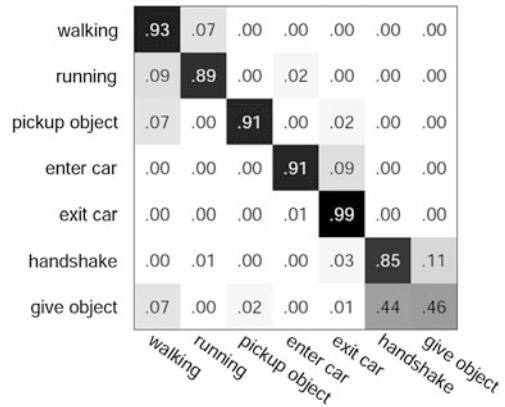
<sup>2</sup><http://lastlaugh.inf.cs.cmu.edu/libscom/downloads.htm>

<sup>3</sup><http://vision.ucsd.edu/%7edollar/research.html>



**Fig. 3.11** Sample frames of sequences from the MICC-UNIFI Surveillance dataset

**Fig. 3.12** Classification accuracy on the MICC-Surveillance dataset using radius-based clustering with soft assignment



**Table 3.4** Comparison of classification accuracy on MICC-Surveillance dataset with our method, k-means with soft assignment, k-means with hard assignment, and with the descriptors proposed in [13] and [16]

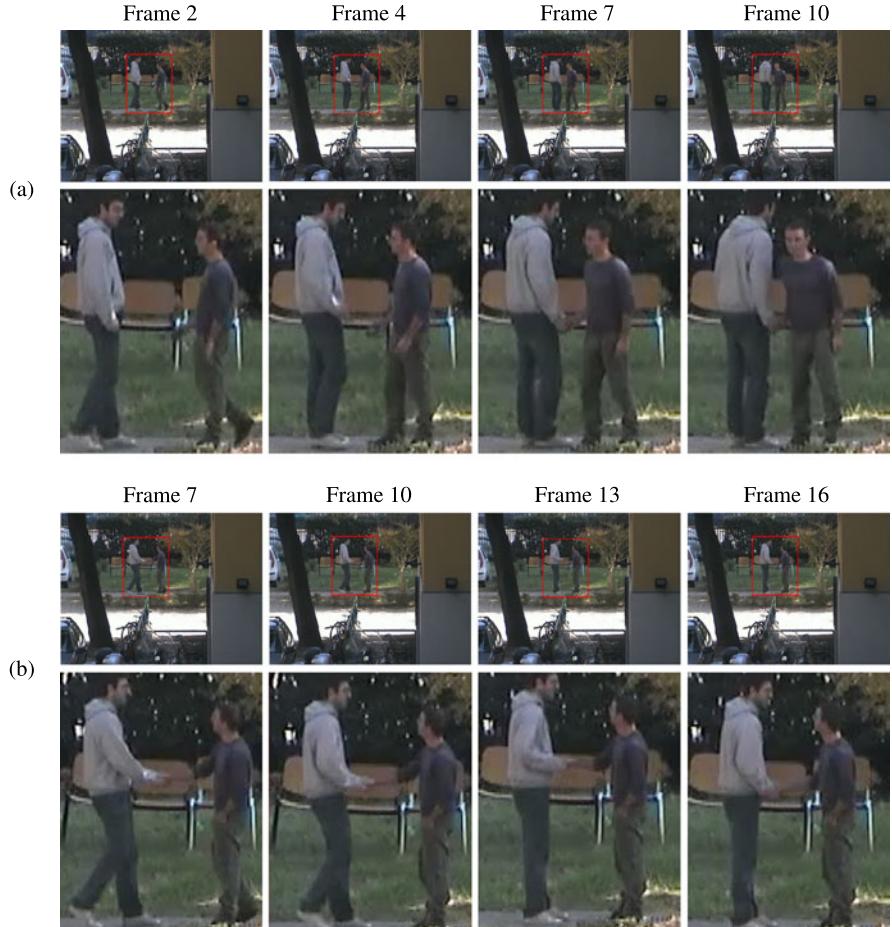
Method	MICC-Surveillance
<i>Our method</i>	86.28
<i>k-means + soft</i>	83.74
<i>k-means</i>	82.90
Dollár et al. [13]	72.50
MoSIFT [16]	75.88

the proposed codebook creation approach performs better than the typical k-means clustering whether with hard and soft assignment.

### 3.5.3 Experiments on Hollywood2 Dataset

The Hollywood2 dataset [34] is composed by sequences extracted from DVDs of 69 Hollywood movies, showing 12 different actions in realistic and challenging settings: *answer phone, drive car, eat, fight person, get out of car, handshake, hug person, kiss, run, sit down, sit up, stand up*. We performed our experiments with the same setup of [27, 55] using the “clean” training dataset, containing scenes that have been manually verified. This dataset is composed by 1707 sequences divided in training set (823) and test set (884), with different frame size and frame rate; train and test set videos have been selected from different movies. To be comparable with other experimental results, the performance has been evaluated computing the average precision (AP) for each class and reporting also the mean AP over all classes. Codebooks have been created using 4000 codewords, as in [55]. We have compared our codebook creation approach with k-means clustering using both soft and hard assignments, and with an implementation of the method proposed in [27] using the provided descriptor and detector.<sup>4</sup> Results are reported in Table 3.5, showing that

<sup>4</sup><http://www.irisa.fr/vista/Equipe/People/Laptev/download.html>



**Fig. 3.13** Sample frames of *give object* (a) and *handshake* (b) action sequences in the MICC-Surveillance dataset. For each sequence, the *second row* shows the detail indicated in *red* in the *first row*

the proposed method outperforms the other approaches in the majority of action classes and in terms of mean AP.

### 3.5.4 Experiments on Reducing the Codebook Size

Large codebooks, although being able to exploit the most informative codewords as illustrated in Fig. 3.8, imply high time and space complexity. Reduction of codebook size with preservation of descriptive capability is therefore desirable. Linear dimensionality reduction techniques such as Principal Component Analysis or Latent Semantic Analysis, are not suited to this end because they are not able to handle

**Table 3.5** Comparison of per-class AP performance on Hollywood2 dataset with codebooks created with our method, k-means with soft assignment, k-means with hard assignment and with the detector+descriptor proposed by Laptev et al. [27]

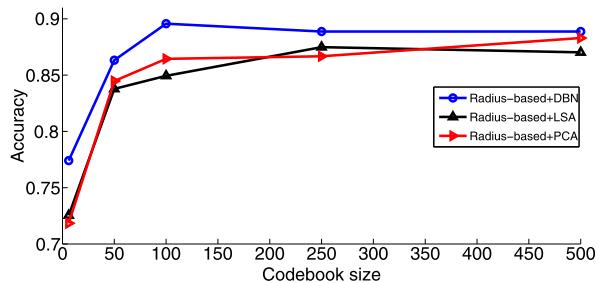
Action	<i>k-means</i>	<i>k-means + soft</i>	<i>Our method</i>	Laptev et al. [27]
Answer phone	0.178	0.186	0.195	0.134
Drive car	0.864	0.865	0.863	0.861
Eat	0.552	0.564	0.564	0.596
Fight person	0.564	0.557	0.578	0.643
Get put of car	0.362	0.364	0.362	0.297
Handshake	0.142	0.143	0.167	0.179
Hug person	0.251	0.257	0.275	0.345
Kiss	0.494	0.510	0.503	0.467
Run	0.631	0.636	0.659	0.619
Sit down	0.483	0.493	0.509	0.505
Sit up	0.215	0.231	0.227	0.143
Stand up	0.511	0.513	0.514	0.485
Mean AP	0.437	0.443	0.451	0.439

high order correlations between codewords that are present in human action representation [51]. We have therefore applied nonlinear dimensionality reduction with Deep Belief Networks (DBNs) [20, 51]. A DBN is composed of several Restricted Boltzmann Machines (RBM) building blocks that encode levels of non-linear relationships of the input vectors. It is pre-trained by learning layers incrementally using contrastive divergence [9]. After pre-training, the auto-encoder is built by reversing the network and connecting the top layer of the network to the bottom layer of its reversed version. The auto-encoder is then used to fine-tune the network using a standard back-propagation algorithm.

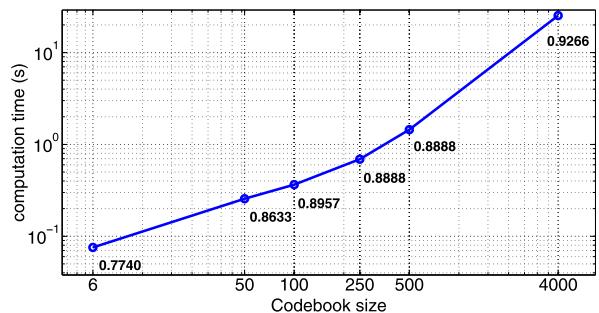
Since the action representation  $H(w)$  can be considered as a coarse probability density estimation of the features of a human action (see Eq. 3.13), given a set of space-time features  $\mathcal{F} = \{f_1, f_2, \dots, f_n\}$ , the value of the  $i$ th bin of  $H$  can be considered as the probability that a space-time descriptor  $f \in \mathcal{F}$  is represented by the codeword  $w_i$ . This probability can hence be used as an input for an RBM according to [19].

Figure 3.14 reports plots of accuracy measured at different codebook sizes, with PCA, LSA and DBN codebook reduction and radius-based clustering with soft assignment, on the KTH dataset. Codebook reduction was applied to a 4000 codewords codebook. The dimension of the input layer is equal to the size of the uncompresssed codebook and the dimension of the output layer is the compressed codebook size. Each hidden layer is one half the dimension of its input layer. The network depth ranges between five and eight depending on the size of the output codebook. The performance of our approach outperforms that of the method recently proposed in [24], especially for the smaller codebook sizes.

**Fig. 3.14** Classification accuracy on KTH dataset at different codebook sizes, with different codebook reduction techniques, for radius-based clustering with soft assignment



**Fig. 3.15** Mean computation times for a KTH video sequence at different codebook sizes with radius-based clustering and DBNs. The numbers associated to the markers indicate the classification accuracy



**Fig. 3.16** Classification accuracy as a function of codebook size, for DBN-compressed and uncompressed codebooks. Radius-based clustering with soft assignment is compared with k-means clustering with hard assignment

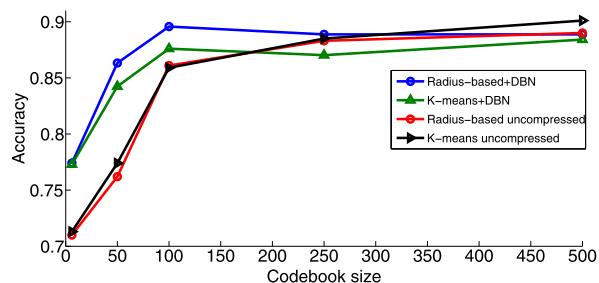


Figure 3.15 reports plots of mean computation times for a KTH video sequence as a function of codebook size for radius-based clustering with soft assignment. The accuracy values of Fig. 3.14 have been reported on the plot for the sake of completeness. It can be noticed that strong codebook size reductions result into time improvements of more than two orders of magnitude. A compressed codebook with 100 codewords scores 89.57 % recognition accuracy with respect to 92.66 % of a 4000 codewords codebook.

Figure 3.16 shows that DBN-compressed codebooks on the one hand provide good accuracy even with very small codebook sizes, and on the other hand make radius-based clustering still competitive with respect to k-means clustering with 100 or less codewords.

Table 3.6 reports a comparison in terms of classification accuracy at different codebook sizes with DBN, PCA and LSA on the MICC-UNIFI surveillance dataset. Codebook reduction was applied to the 2000 codeword codebook obtained with

**Table 3.6** Classification accuracy on MICC-UNIFI dataset at different codebook sizes, with different codebook reduction techniques, for radius-based clustering with soft assignment. Using PCA and LSA, it is not possible to create codebooks larger than the number of training videos; using DBNs this issue is not present

Codebook size	6	50	100	250	500
DBN	0.386	0.397	0.412	0.431	0.474
PCA	0.333	0.378	0.405	–	–
LSA	0.330	0.346	0.335	–	–

**Table 3.7** Classification of MAP performance on Hollywood2 dataset at different codebook sizes, with different codebook reduction techniques, for radius-based clustering with soft assignment

Codebook size	6	50	100	250	500
DBN	0.281	0.372	0.383	0.375	0.374
PCA	0.191	0.323	0.329	0.337	0.338
LSA	0.204	0.322	0.316	0.311	0.314

radius-based clustering and soft assignment in the previous classification experiment. The smaller number of available training videos, with respect to KTH, is responsible for the reduction in classification accuracy, although the DBNs largely outperform the other methods. This experiment shows another advantage of the use of DBNs over PCA and LSA when the number of sequences available for training is relatively small, that is, the possibility to create larger dictionaries that usually yield higher classification accuracy although maintaining a speed improvement of an order of magnitude. Table 3.7 reports a comparison of MAP performance obtained using compressed codebooks created with DBN, PCA and LSA on the Hollywood2 dataset. Codebook reduction was applied to the 4000 codeword codebook obtained with radius-based clustering and soft assignment used in the classification experiment. Despite the challenging dataset, the performance is still comparable with that obtained with full sized codebooks by several approaches reported in [55].

### 3.5.5 Tracker Evaluation and Experiments on Recognizing Multiple Actions

These experiments have been performed on a subset of the MICC-UNIFI Surveillance dataset. First of all, we have evaluated our tracking module quality by measuring multiple object tracking accuracy (MOTA) as defined by Bernardin and Stiefelhagen [5]. MOTA is an intuitive performance metric for multiple object trackers and measures a tracker performance at keeping accurate trajectories. For each frame processed a tracker should produce a set of object hypotheses, each of which should ideally correspond to a real visible object. In order to compute MOTA, a consistent

**Table 3.8** Multiple object tracking accuracy (MOTA) together with false positive rate (FPR), false negative rate (FNR) and amount of identity switches (SWITCH)

Seq.	FPR	FNR	SWITCH	MOTA
1	27.92	2.92	0	68.35
2	38.56	12.40	2	49.82
3	13.15	32.16	0	54.67
4	23.65	9.18	0	67.20
5	15.02	27.48	0	57.74
6	14.59	3.82	52	79.38

hypothesis-object mapping over time must be produced; the complete procedure to obtain this mapping is specified in detail in [5]. MOTA takes into account all possible errors that a multi-object tracker makes: false positives, missed objects and identity switches. False positives ( $fp$ ) arise when, for example, the tracker is initiated on a false detection or when an object is missed and consequently a wrong pattern replaces the correct object hypothesis. Misses or false negatives ( $fn$ ) arise whenever an object is not mapped to any of the hypotheses proposed by the tracker; finally identity switches ( $sw$ ) happen whenever an object hypothesis is mapped to the wrong object, for example after an occlusion or when an object tracker fails and another tracker is reinitialized. Errors are normalized by the number of objects present ( $gt$ ) with respect to the whole sequence.

MOTA is defined as follows:

$$MOTA = 1 - \frac{\sum_t fp_t + fn_t + sw_t}{\sum_t gt_t} \quad (3.14)$$

We represent persons as bounding boxes and we consider a mapping correct if  $\frac{O \cap H}{O \cup H} \geq 0.5$ , where  $O$  and  $H$  are the areas of the object and the hypothesis bounding boxes mapped. We measured MOTA for all five sequences in which our final recognition experiments were performed and another sequence (Table 3.8). The last sequence is recorded with a PTZ camera, panning tilting and zooming on targets and targets are instructed to produce overlapping trajectories in order to create difficult situations for a multiple object tracker. In the first five test sequences, most of the errors are caused by false alarms of the pedestrian detector that cause instantiation of trackers; in the classification stage this empty tracks can be filtered since they usually do not contain enough detected space-time interest points. In the last sequence, most of the errors are due to identity switches since target maneuvers are more complex. MOTA is quite satisfying in all sequences, considering also that, in order to attain real-time performance, our appearance model is weak and no online classifier is used to perform data association or learn the template.

We have further evaluated the performance of our approach on five complex video sequences containing multiple actions performed concurrently (two examples are shown in Fig. 3.17). These sequences have different durations ranging from a minimum of  $\sim 120$  to a maximum of  $\sim 300$  frames. Our method has been applied to recognize and localize two basic actions: *walking* and *running*. As training set,



**Fig. 3.17** Example of two sequences from our multiple-action surveillance dataset. In the first sequence (*seq. 3*), our actors perform a *pickpocketing* event. In the second sequence (*seq. 5*), a *snatch* is performed

**Table 3.9** System performance on complex video sequences: for each sequence the number of tracks, action ground-truth ( $\mathbf{W}_{\text{GT}}$ ,  $\mathbf{R}_{\text{GT}}$ ,  $\mathbf{O}_{\text{GT}}$ ), and classification accuracy are reported

Seq.	Detected	Filtered	$\mathbf{W}_{\text{GT}}$	$\mathbf{R}_{\text{GT}}$	$\mathbf{O}_{\text{GT}}$	Acc
1	8	5	3	2	0	4/5
2	7	6	3	2	1	5/6
3	11	5	2	2	1	4/5
4	8	6	2	3	1	4/6
5	8	5	3	2	0	4/5
						21/27

we used the videos containing a single person performing the same action multiple times.

Table 3.9 shows the performance of our approach on surveillance videos. For each sequence, we report the detected tracks identified from our person detector and tracker. The tracks that contain less than 30 interest points are discarded and the filtered tracks are then used to perform action classification. These tracks are manually annotated in walking, running and other action (reported in Table 3.9 in the columns  $\mathbf{W}_{\text{GT}}$ ,  $\mathbf{R}_{\text{GT}}$ ,  $\mathbf{O}_{\text{GT}}$  respectively). Details of classification accuracy are shown. We note that 21/27 tracks are recognized correctly. The performance of action classification is evaluated in terms of two standard metrics that is, precision and recall, defined as:

$$\text{precision} = \frac{\# \text{ of correctly predicted actions}}{\# \text{ of predicted actions}} \quad (3.15)$$

$$\text{recall} = \frac{\# \text{ of correctly predicted actions}}{\# \text{ of ground-truth actions}} \quad (3.16)$$

Precision and recall performance of the action recognition, also shown in Table 3.10, are mostly affected by mistaken classification of the tracks that contain the “other”

**Table 3.10** Precision and recall for the *running* and *walking* actions

Action	Precision	Recall
Walking	73 %	85 %
Running	77 %	71 %

action, since only one track that contains a walking action was classified as running action.

### 3.6 Conclusions

In this chapter, we have presented a novel method for human action categorization that exploits a new descriptor for spatio-temporal interest points that combines appearance (3D gradient descriptor) and motion (optic flow descriptor), and effective codebook creation based on radius-based clustering and a soft assignment of feature descriptors to codewords. The approach was validated on KTH and Weizmann datasets, on the Hollywood2 dataset and on a new surveillance dataset that contain unconstrained video sequences that include more realistic and complex actions. Results outperform the state-of-the-art with no parameter tuning. We have also shown that a strong reduction of computation time can be obtained by applying codebook size reduction with Deep Belief Networks, with small reduction of classification performance.

## References

1. Arulampalam M, Maskell S, Gordon N, Clapp T (2002) A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans Signal Process* 50(2):174–188
2. Bagdanov AD, Dini F, Del Bimbo A, Nunziati W (2007) Improving the robustness of particle filter-based visual trackers using online parameter adaptation. In: Proc of AVSS
3. Ballan L, Bertini M, Del Bimbo A, Seidenari L, Serra G (2011) Event detection and recognition for semantic annotation of video. *Multimed Tools Appl* 51(1):279–302
4. Ballan L, Bertini M, Del Bimbo A, Seidenari L, Serra G (2012) Effective codebooks for human action representation and classification in unconstrained videos. *IEEE Trans Multimed* 14(4):1234–1245
5. Bernardin K, Stiefelhagen R (2008) Evaluating multiple object tracking performance: the CLEAR MOT metrics. *EURASIP J Image Video Process* 2008:246309
6. Bobick AF, Davis JW (2001) The recognition of human movement using temporal templates. *IEEE Trans Pattern Anal Mach Intell* 23(3):257–267
7. Bregonzio M, Gong S, Xiang T (2009) Recognising action as clouds of space-time interest points. In: Proc of CVPR
8. Cao L, Zicheng L, Huang T (2010) Cross-dataset action detection. In: Proc of CVPR
9. Carreira Perpinan MA, Hinton GE (2005) On contrastive divergence learning. In: Proc of AISTATS
10. Chen MY, Hauptmann AG (2009) MoSIFT: recognizing human actions in surveillance videos. Technical report, CMU

11. Comaniciu D, Meer P (2002) Mean shift: a robust approach toward feature space analysis. *IEEE Trans Pattern Anal Mach Intell* 24(5):603–619
12. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: Proc of CVPR
13. Dollár P, Rabaud V, Cottrell G, Belongie S (2005) Behavior recognition via sparse spatio-temporal features. In: Proc of VSPETS
14. Efros AA, Berg AC, Mori G, Malik J (2003) Recognizing action at a distance. In: Proc of ICCV
15. Fergus R, Perona P, Zisserman A (2003) Object class recognition by unsupervised scale-invariant learning. In: Proc of CVPR
16. Gao Z, Chen MY, Hauptmann AG, Cai A (2010) Comparing evaluation protocols on the KTH dataset. In: Proc of HBU workshop
17. Gorelick L, Blank M, Schechtman E, Irani M, Basri R (2007) Actions as space-time shapes. *IEEE Trans Pattern Anal Mach Intell* 29(12):2247–2253
18. Hauptmann AG, Christel MG, Yan R (2008) Video retrieval based on semantic concepts. Proc IEEE 96(4):602–622
19. Hinton EG, Salakhutdinov R (2006) Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507
20. Hinton EG, Osindero S, Teh Y (2006) A fast learning algorithm for deep belief nets. *Neural Comput* 18(7):1527–1554
21. Jiang YG, Yang J, Ngo CW, Hauptmann AG (2010) Representations of keypoint-based semantic concept detection: a comprehensive study. *IEEE Trans Multimed* 12(1):42–53
22. Jurie F, Triggs B (2005) Creating efficient codebooks for visual recognition. In: Proc of ICCV
23. Kläser A, Marszałek M, Schmid C (2008) A spatio-temporal descriptor based on 3D-gradients. In: Proc of BMVC
24. Kong Y, Zhang X, Hu W, Jia Y (2011) Adaptive learning codebook for action recognition. *Pattern Recognit Lett* 32(8):1178–1186
25. Kovashka A, Grauman K (2010) Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. In: Proc of CVPR
26. Laptev I (2005) On space-time interest points. *Int J Comput Vis* 64(2–3):107–123
27. Laptev I, Marszałek M, Schmid C, Rozenfeld B (2008) Learning realistic human actions from movies. In: Proc of CVPR
28. Lazebnik S, Schmid C, Ponce J (2006) Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: Proc of CVPR
29. Lin Z, Jiang Z, Davis LS (2009) Recognizing actions by shape-motion prototype trees. In: Proc of ICCV
30. Liu J, Shah M (2008) Learning human actions via information maximization. In: Proc of CVPR
31. Liu J, Ali S, Shah M (2008) Recognizing human actions using multiple features. In: Proc of CVPR
32. Liu J, Luo J, Shah M (2009) Recognizing realistic actions from videos “in the wild”. In: Proc of CVPR
33. Lucas BD, Kanade T (1981) An iterative image registration technique with an application to stereo vision. In: Proc of DARPA IU workshop
34. Marszałek M, Laptev I, Schmid C (2009) Actions in context. In: Proc of CVPR
35. Mikolajczyk K, Uemura H (2008) Action recognition with motion-appearance vocabulary forest. In: Proc of CVPR
36. Mikolajczyk K, Leibe B, Schiele B (2005) Local features for object class recognition. In: Proc of ICCV
37. 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
38. Moeslund T, Hilton A, Krüger V (2006) A survey of advances in vision-based human motion capture and analysis. *Comput Vis Image Underst* 104(2–3):90–126

39. Niebles JC, Wang H, Fei-Fei L (2008) Unsupervised learning of human action categories using spatial-temporal words. *Int J Comput Vis* 79(3):299–318
40. Poppe R (2007) Vision-based human motion analysis: an overview. *Comput Vis Image Underst* 108(1–2):4–18
41. Poppe R (2010) A survey on vision-based human action recognition. *Image Vis Comput* 28(6):976–990
42. Rapantzikos K, Avrithis Y, Kollia S (2009) Dense saliency-based spatiotemporal feature points for action recognition. In: Proc of CVPR
43. Schüldt C, Laptev I, Caputo B (2004) Recognizing human actions: a local SVM approach. In: Proc of ICPR
44. Scovanner P, Ali S, Shah M (2007) A 3-dimensional SIFT descriptor and its application to action recognition. In: Proc of ACM multimedia
45. Shao L, Mattivi R (2010) Feature detector and descriptor evaluation in human action recognition. In: Proc of CIVR
46. Shao L, Gao R, Liu Y, Zhang H (2011) Transform based spatio-temporal descriptors for human action recognition. *Neurocomputing* 74(6):962–973
47. Sivic J, Zisserman A (2003) Video Google: a text retrieval approach to object matching in videos. In: Proc of ICCV
48. Snoek CGM, Worring M, van Gemert JC, Geusebroek JM, Smeulders AWM (2006) The challenge problem for automated detection of 101 semantic concepts in multimedia. In: Proc of ACM multimedia
49. Sun X, Chen M, Hauptmann AG (2009) Action recognition via local descriptors and holistic features. In: Proc of CVPR4HB workshop
50. Turaga P, Chellappa R, Subrahmanian V, Udrea O (2008) Machine recognition of human activities: a survey. *IEEE Trans Circuits Syst Video Technol* 18(11):1473–1488
51. van der Maaten L, Postma E, van den Herik H (2009) Dimensionality reduction: a comparative review. Technical report TiCC-TR 2009-005, Tilburg University
52. van Gemert JC, Veenman CJ, Smeulders AWM, Geusebroek JM (2010) Visual word ambiguity. *IEEE Trans Pattern Anal Mach Intell* 32(7):1271–1283
53. Vezzani R, Cucchiara R (2010) Video surveillance online repository (ViSOR): an integrated framework. *Multimed Tools Appl* 50(2):359–380
54. Wang Y, Mori G (2009) Max-margin hidden conditional random fields for human action recognition. In: Proc of CVPR
55. Wang H, Ullah MM, Kläser A, Laptev I, Schmid C (2009) Evaluation of local spatio-temporal features for action recognition. In: Proc of BMVC
56. Willems G, Tuytelaars T, Van Gool L (2008) An efficient dense and scale-invariant spatio-temporal interest point detector. In: Proc of ECCV
57. Wong SF, Cipolla R (2007) Extracting spatiotemporal interest points using global information. In: Proc of ICCV
58. Wu B, Nevatia R (2007) Detection and tracking of multiple, partially occluded humans by Bayesian combination of edgelet based part detectors. *Int J Comput Vis* 75(2):247–266
59. Yao A, Gall J, Van Gool L (2010) A hough transform-based voting framework for action recognition. In: Proc of CVPR
60. Yilmaz A, Shah M (2005) Actions sketch: a novel action representation. In: Proc of CVPR
61. Yu G, Goussies N, Yuan J, Liu Z (2011) Fast action detection via discriminative random forest voting and top-k subvolume search. *IEEE Trans Multimed* 13(3):507–517
62. Zhang J, Marszałek 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

# Chapter 4

## Evaluating and Extending Trajectory Features for Activity Recognition

Ross Messing, Atousa Torabi, Aaron Courville, and Chris Pal

**Abstract** Trajectory features are a powerful new way to describe video data. By leveraging the spatio-temporal range and structure of trajectories, they improve activity recognition performance compared to systems based on fixed local spatio-temporal volumes. This chapter places them in context, compares a sparse, generative model of extended trajectories to a dense, discriminative model of local trajectories, and explores ways to extend the sparse system with new kinds of information.

### 4.1 Introduction and Background

Activity recognition in video is an increasingly important field of computer vision, with many practical applications in areas like surveillance and smart environments. As activity recognition techniques mature, it is important to take stock of new developments, and evaluate how to integrate them with established techniques.

---

R. Messing (✉)  
University of Rochester, Rochester, NY, USA  
e-mail: [rmessing@cs.rochester.edu](mailto:rmessing@cs.rochester.edu)

*Present address:*  
R. Messing  
Tandent Vision Science, Inc., Pittsburgh, PA, USA

A. Torabi · A. Courville  
Université de Montréal, Montréal, Quebec, Canada

A. Torabi  
e-mail: [torabi@iro.umontreal.ca](mailto:torabi@iro.umontreal.ca)

A. Courville  
e-mail: [courvila@iro.umontreal.ca](mailto:courvila@iro.umontreal.ca)

C. Pal  
École Polytechnique de Montréal, Montréal, Quebec, Canada  
e-mail: [christopher.pal@polymtl.ca](mailto:christopher.pal@polymtl.ca)

### 4.1.1 Activity Recognition

Early work on activity recognition required very explicit, rigid models of the activities to be recognized, and the people performing those actions [1, 19, 21]. While these systems are not as sophisticated and flexible as modern, state-of-the-art systems, they encoded many important intuitions, like what kinds of information can be usefully combined for activity recognition.

Statistical methods, most notably the bag of words [14] approach to recognition, first became very popular in object recognition in still images [7]. By running an interest point operator to find repeatable interest points in images (usually distinctive corners), extracting a local appearance patch description from the interest point’s neighborhood, and codebooked those descriptions, the histogram of local feature codewords becomes a very powerful description of an image, which loses broad spatial structure, but keeps a great deal of local neighborhood structure. This histogram of local patches representation can be treated like a histogram of word counts in a document, and techniques from natural languages processing can be used to classify that histogram.

Statistical approaches inspired by the success of bag-of-words models in still-image object recognition have come to dominate activity recognition in video. A lot of research has gone into the best way to describe small pieces of video, and how to combine those little bits to describe a whole video. The initial work involved straightforward generalization of spatial interest point detectors and descriptors to the temporal dimension [6, 11, 12], recent work has taken a more sophisticated approach to the spatio-temporal structure of video, yielding descriptors that follow trajectories across video [17, 24]. Inspired by psychophysical results showing the power of trajectories in human perception [9], these features seem to outperform local features attached to fixed points.

Trajectory features harness the idea that motion is extremely informative for the perception of human actions. Johansson [9] showed that by attaching point-lights to humans performing actions in the dark, those actions were easily recognizable. Cutting [4] introduced the first computational models of visual trajectories in a psychophysics subfield called “event perception”.

### 4.1.2 Trajectory Features

There are several steps required to use feature trajectories for activity recognition. First, trajectories must be detected in the video. One way to do this is to randomly sample trajectories from uniformly distributed points throughout the video. This is known as dense sampling [23]. Depending on the sampling frequency, it leads to a large number of features, both relevant and irrelevant. The other, more traditional way to detect features is with an interest point detector, just as in still images [20], but with a temporal component as well. By finding reliable corners in space and time, features are more likely to be repeatable, and often result in better tracks.

These interest point detections can be called sparse detection, and leads to fewer features, but often better features, although good features to track that the detector misses may be present with dense features, but absent with sparse features. Recent work by Wang et al. show improved performance with dense local spatio-temporal features [23].

Once features are detected, they must be tracked, generally using optical flow. Optical flow involves finding the velocity transformation between frames, either locally or globally, that minimizes the difference between frames. Sparse trajectory features tend to use local patch-based optical flow, like the KLT tracker [15]. These algorithms tend to be much faster than global optical flow estimates, that compute a full optical flow field across the whole frame. Global optical flow techniques may, on the other hand, offer more robust, non-local motion constraints [3]. Once features are detected, and local or global optical flow is calculated, trajectories are just the concatenation of the optical flow field at the sequence of points indicated by following the optical flow field.

Whole image optical flow techniques consider a broader spatial area than sparse techniques, can incorporate softer constraints, and may be more robust to unusual motion. For example, Brox and Malik integrate rich spatial descriptors into a flow estimator minimizing a variational energy form to produce a more expensive, more robust global optical flow estimate [3]. Depending on how densely dense features are sampled, they can yield a large number of features, some of which are irrelevant, or a small number of features, few of which are irrelevant. When the dense features have some larger fixed distance between them, dense sampling can be much faster than feature detection, as it saves the computational overhead of feature detection. Dense features can still be more expensive than sparse features, however, as computing a dense optical flow field can also be expensive, particularly if it incorporates many global and soft probabilistic constraints.

#### 4.1.3 Recent Developments

Local spatio-temporal features and descriptors in combination with the bag of features models have proven successful for action recognition [12, 16, 18]. In using local descriptors, there is always a trade-off between their discriminative power and their invariance to irrelevant changes in the video, like camera motion and illumination. This limitation causes ambiguities in video representation by local spatio-temporal features, especially when dealing with a large number of actions, resulting in low recognition performance. To cope with this problem, Ullah et al. have proposed an extension of BOF action recognition using non-local cues, including motion-based foreground segmentation, person detection, and object detection [22].

A recent approach to video representation is based on unsupervised feature learning as a way to learn features directly from video rather than applying hand-designed local features described in previous approaches. Le et al. [13] introduce an Independent Subspace Analysis algorithm that learns invariant spatio-temporal features

from unlabeled video data. These learned spatio-temporal features achieve state-of-the-art performance on datasets like HoHA2, UCF, and Youtube.

We investigate two state-of-the-art trajectory descriptor systems in Sect. 4.2.

## 4.2 Trajectory Descriptors for Action Recognition

Activity recognition in video is dominated by approaches based on the bag-of-features models that work so well in still images. As the field matures, descriptors directly extending still-image descriptors into time are being supplanted by descriptors relying more fundamentally on the temporal structure of video. One type of descriptor that has yielded excellent performance is the family of trajectory-based descriptors. These descriptors follow trajectories extracted from the video, and their descriptiveness relies primarily or entirely on their motion, information which is unavailable in still images. We consider two very different kinds of trajectory features. Messing et al. [17] introduced a descriptor based on extended tracking of sparse keypoints, while Wang et al. [24] leverage dense optical flow to track and describe a set of densely-sampled features that span a fixed temporal range, but extend beyond a local spatial window. To our knowledge, these features have never been directly compared. We first review these features, and then compare their performance on a dataset where local spatio-temporal features do not perform well.

### 4.2.1 Velocity Histories of Sparse Tracked Keypoints

Messing et al. [17] introduce a video descriptor designed to capture the detailed motion of tracked features over long periods of time. By modeling trajectories with a log-polar velocity Markov chain, they naturally deal with variable-length trajectories, something that all other trajectory descriptors have had to deal with by converting them to fixed-length trajectories in one of a variety of ways. Velocity Markov chain parameters are learned via EM as part of a mixture model of velocity Markov chains. Their model can be thought of as extending a kind of supervised topic model, like Latent Dirichlet Allocation [2], to trajectories, where a distribution over discrete words is instead replaced by a Markov chain distribution over sequences of velocity transitions.

These features were explicitly designed to extend beyond local spatio-temporal boundaries of most existing activity recognition features, like Dollar et al.’s cuboids [6], or Laptev et al.’s STIP features [12]. Their performance on small videos with simple motion, where local space-time volumes adequately captured most of the relevant information, was unimpressive. However, on datasets with large videos containing complex, confusable motion, their performance was state-of-the-art. Messing et al. further extended these features by augmenting trajectory features with ancillary position and appearance information. This further improved performance, but at the price of translation-invariance, an important feature for a robust activity recognition system. Examples of Messing et al.’s sparse trajectory features can be seen in Figs. 4.2 and 4.4.

**Fig. 4.1** Wang et al.’s dense features found on a sample frame from URADL’s Answer Phone activity. *Red dots* denote feature candidates. *Green trails* denote trajectory histories



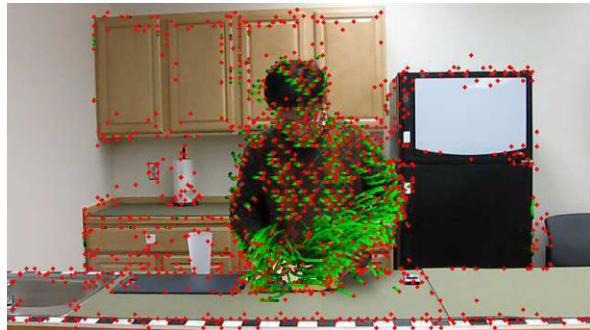
**Fig. 4.2** Messing et al.’s sparse features found on another sample frame from URADL’s Answer Phone activity. *Red dots* denote feature candidates. *Green trails* denote trajectory histories



#### 4.2.2 Dense Trajectories

Wang et al. [24] extended existing work on dense local features for activity recognition [23] by linking their dense local features to trajectories. The original dense local feature work showed improved results for activity recognition using densely sampled features, rather than features detected by an interest point detector [23]. Wang et al. describe videos by first extracting a dense optical flow field (which can incorporate non-local constraints, and so may produce higher quality optical flow than the sparse methods used by Messing et al. [17]), densely sampling features across the image frame, and following their trajectories frame-to-frame. As features stop moving, become difficult to track, or reach the fixed trajectory duration of 15 frames, new features are sampled, keeping the total number of features roughly constant. Feature descriptors are applied to each fixed-duration trajectory. Example sparse and dense features extracted with Wang et al.’s detector are shown in Figs. 4.1 and 4.3. The descriptors included histograms of oriented gradients and optical flow (HOG and HOF) descriptors from [12], modified to follow the feature trajectory, but otherwise unchanged. In addition, the velocity history of the trajectory feature was used as a descriptor, differing from Messing et al.’s velocity history descriptor by being fixed-length, and not requiring the discretization required by Messing et al. [17]. Lastly, the motion boundary histogram (MBH), a feature based on motion boundaries detected by the horizontal and vertical gradients of the optical flow field, was introduced to activity recognition.

**Fig. 4.3** Wang et al.’s dense features found on a sample frame from URADL’s Lookup in Phonebook activity. Red dots denote feature candidates. Green trails denote trajectory histories



**Fig. 4.4** Messing et al.’s sparse features found on another sample frame from URADL’s Lookup in Phonebook activity. Red dots denote feature candidates. Green trails denote trajectory histories



Wang et al. [24] demonstrated impressive performance on established activity recognition datasets, outperforming local spatio-temporal features with their trajectory-linked descriptors.

#### 4.2.3 Comparison

We compare Messing et al.’s [17] sparse, generative trajectory features with Wang et al.’s [24] dense, discriminative trajectory features. We control the high level model, and different kinds of information, to get the closest possible comparison between the two methods. We perform our evaluation on the URADL dataset [17].

Wang et al. [24] efficiently extract the dense trajectories by densely sampling the points in an image grid and tracking them in multiple scales separately using the median filtering in a dense optical flow field. In this way, once the dense optical flow is computed, points can be tracked at no additional cost. Wang et al. impose several heuristic filters to improve the quality of extracted dense trajectories: the length of the trajectories are limited to  $L$  frames to avoid drifting problem in tracking, densely sampled points on homogeneous areas are removed using the same criterion as Shi and Tomasi [20], and static trajectories and trajectories with sudden large displacements are pruned since they are most likely to be either erroneous or non-informative. Moreover in their work, dense trajectories are augmented using HOG, HOF, and MBH as trajectory-aligned descriptors in a 3D video volume of

size  $N \times N$  pixels and  $L$  frames around tracked points. The video volume is subdivided into a spatio-temporal grid of size  $n_\sigma \times n_\sigma \times n_\tau$  to embed in the feature representation.

Messing et al. [17] use a generative mixture model to describe sparse trajectories found by Laptev et al.'s STIP detector [12] and tracked using the KLT tracker [15] as a mixture of Markov chains. The only limit imposed on trajectories is that they must be at least 10 frames long (Corresponding in this dataset to one third of a second).

#### 4.2.3.1 Method

We implemented Wang et al.'s [24] bag-of-features model. We use the default feature extraction parameters as suggested in this paper. We set, the trajectory length  $L = 15$  and video volume sizes  $N = 32$ ,  $n_\sigma = 2$ ,  $n_\tau = 3$ . In their work, a codebook for each descriptor (dense trajectory, HOG, HOF, MBH) is constructed separately. Since the URADL dataset is smaller than Hollywood2 (i.e., dataset that is used in [24]), we use smaller set of visual words. We construct a set of 100 visual words by clustering 5000 randomly selected training features using k-means.

We implemented Messing et al.'s soft generative bag of features model [17]. We use the same settings as they did for log-polar discretization of trajectories (8 angular bins, 5 magnitude bins), and 100 mixture components.

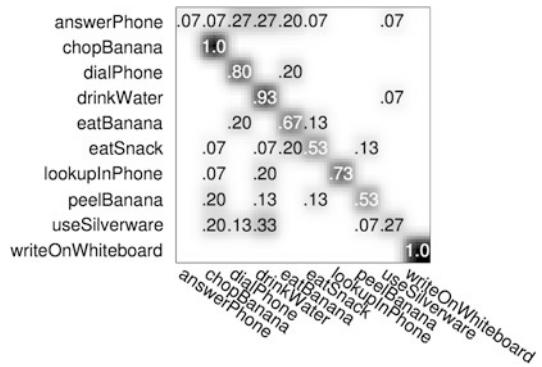
For classification, we use two approaches: Naive Bayes and the non-linear SVM with chi-squared multi-channel kernel as described by Wang et al. [24], using *SVM-light*'s support vector machine implementation [8].

We extract bags of words from Messing et al.'s generative model by using a video's marginal distribution over mixture component, removing the contribution from the higher level class model's distribution over the mixture component.

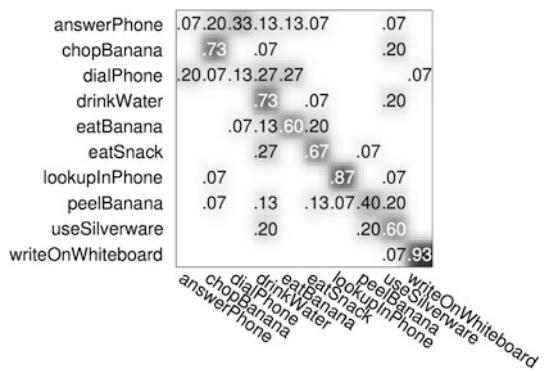
We compare bags of words against several versions of Wang et al.'s bags of words. First, we compare using just their dense trajectory feature, as it is most similar to Messing et al.'s trajectory features. Messing et al.'s features have longer, variable duration, while Wang et al.'s don't require discretization. We also combine Wang et al.'s dense trajectory features with their histogram of optical flow (HOF) and motion boundary histogram (MBH) features. This could be interpreted as a fairer comparison Messing et al.'s features, as those features only rely on motion, as well, and are cheap to calculate given the dense optical flow required for the dense trajectory features. Lastly, we combine all of Wang et al.'s motion-based features with their histogram of oriented gradients (HOG) appearance descriptor. This uses extra information beyond motion, and so we would expect it to improve performance.

We compare these feature sets using two different high level models. First, we use Naive Bayes, as in Messing et al. [17], a very simple model that applies equal weight to all features, thus penalizing irrelevant features. Naive Bayes is fast and easy to fit, an advantage in datasets like URADL which contain a relatively small number (150 movies) of extremely complex data points. For smoothing, we used a uniform Dirichlet prior, and adjusted it to maximize performance. We also use the multi-channel non-linear SVM described in Wang et al. [24] to illustrate the value of discriminative methods on sparse and dense features.

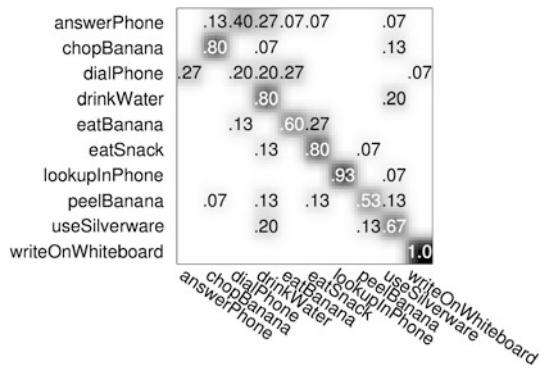
**Fig. 4.5** Confusion matrix for Messing et al.'s sparse generative features combined using Naive Bayes. Average accuracy is 65.3 %. Zeros are omitted for clarity



**Fig. 4.6** Confusion matrix for Wang et al.'s dense trajectory features combined using Naive Bayes. Average accuracy is 57.3 %. Zeros are omitted for clarity



**Fig. 4.7** Confusion matrix for Wang et al.'s dense trajectory features, HOF features and MBH features combined using Naive Bayes. Average accuracy is 63.3 %. Zeros are omitted for clarity

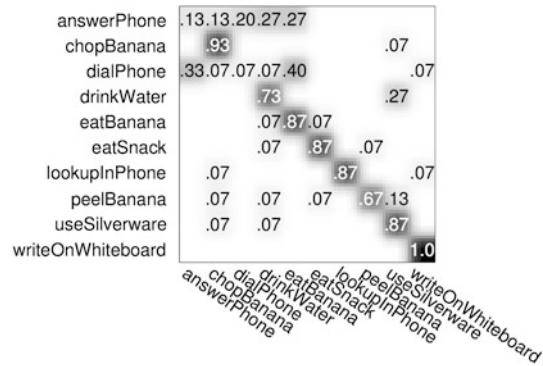


#### 4.2.3.2 Results

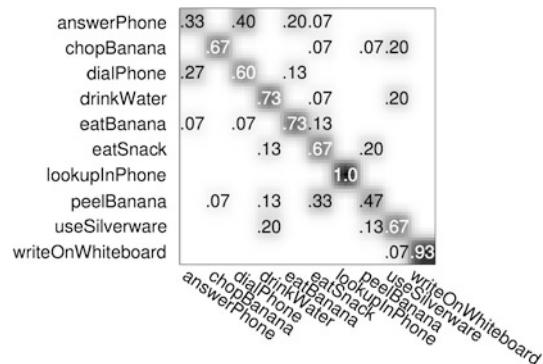
Figures 4.5–4.12 show confusion matrices for the different experimental conditions.

Performance is summarized in Table 4.1.

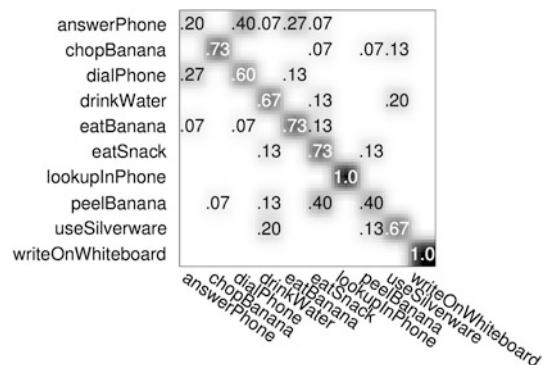
**Fig. 4.8** Confusion matrix for Wang et al.'s dense trajectory features, HOF features, MBH features and HOG appearance features combined using Naive Bayes. Average accuracy is 70 %. Zeros are omitted for clarity



**Fig. 4.9** Confusion matrix for Messing et al.'s sparse generative features combined using an SVM. Average accuracy is 68 %. Zeros are omitted for clarity



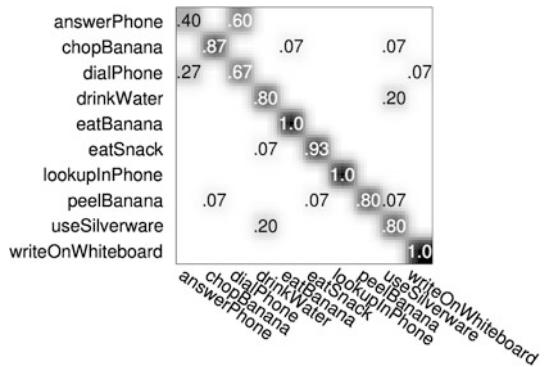
**Fig. 4.10** Confusion matrix for Wang et al.'s dense trajectory features combined using an SVM. Average accuracy is 67.3 %. Zeros are omitted for clarity



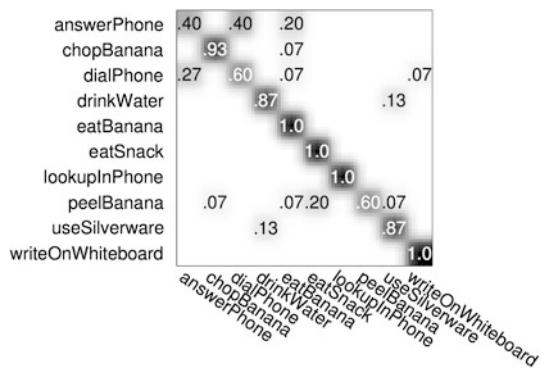
#### 4.2.4 Discussion

Table 4.1 illustrates the results of our comparison. Unsurprisingly, in the pattern of the dense trajectory results as features are added, we see that adding information improves performance. In comparing the trajectory features themselves, it seems likely that the Messing et al. features, with their extended and variable-length trajectories, are more descriptive than Wang et al.'s fixed-length trajectory features, even though

**Fig. 4.11** Confusion matrix for Wang et al.’s dense trajectory features, HOF features and MBH features combined using an SVM. Average accuracy is 82.6 %. Zeros are omitted for clarity



**Fig. 4.12** Confusion matrix for Wang et al.’s dense trajectory features, HOF features, MBH features and HOG appearance features combined using an SVM. Average accuracy is 82.6 %. Zeros are omitted for clarity



**Table 4.1** Performance of Messing et al.’s velocity history features [17] and sets of Wang et al.’s dense trajectory features [24] using Naive Bayes and SVMs on the URADL dataset

Method	Naive Bayes	SVM
Messing et al. Sparse Trajectories	65.3 %	68.0 %
Wang et al. Dense Trajectories	57.3 %	67.3 %
Wang et al. Dense Trajectories + HOF + MBH	63.3 %	82.6 %
Wang et al. Dense Trajectories + HOF + MBH + HOG	70 %	82.6 %

they are un-discretized. The advantage of Messing et al.’s trajectories goes away once other motion information (HOF and MBH) are added to Wang et al.’s dense trajectories, illustrating the powerful information in an optical flow field. The addition of appearance information substantially improved performance under a generative higher level model, but appears not to have added much to the discriminative model. It is not clear why this is the case, and it may be worth investigating further to maximize performance.

In comparing the performance of the sparse and dense trajectory features using the simple generative and more complex discriminative model, we see that the

**Table 4.2** Performance of Messing et al.’s velocity history features [17] using a single augmentation on the URADL dataset

Method	Percent correct
Only Trajectories	69.3 %
Trajectories + Absolute Start and End	78.6 %
Trajectories + Relative Start and End	72.6 %
Trajectories + Appearance	70.6 %

features designed for a generative system receive only a marginal boost from the discriminative model, while the dense features receive a substantial one. Because discriminative models can vary the amount of modeling power applied to a given data point, while generative models spend the same amount of power on each data point, discriminative models are clearly superior for dense features, where many irrelevant features should be ignored. When sparse interest point detectors perform particularly well, and when less data is available, we would suggest sparse features, combined with a simpler, easier-to-fit model like Naive Bayes. The correct choice of a model depends on the problem, but as long as there is sufficient data available, these results argue strongly that discriminative methods should be the default choice.

## 4.3 Extending Augmentations

Messing et al. [17] used “augmentations” to associate useful information with their velocity history features. In particular, they used absolute position, relative position, and appearance. We extend their initial exploration of augmentations, focusing on the positional augmentations, as position is a very natural kind of information to pair with trajectories.

### 4.3.1 Messing et al.’s Augmentations

Messing et al. [17] augment their trajectory features with the absolute position of a trajectory’s start and end, the position of it’s start and end relative to a face, and the PCA-SIFT [10] descriptor appearance of a patch extracted from the feature’s initial detection. These augmentations are combined with the trajectories as independent parts of a mixture model. So a mixture component whose parameters include a distribution over discretized velocity sequences also independently included a distribution over feature position, or appearance. In extending this augmentation model, we first examine the augmentations used in the original work [17].

Table 4.2 shows the performance of Messing et al.’s trajectory model with and without each of their augmentations, individually. This table shows two interesting things. First, we see that absolute position is a very powerful source of information. Second, we see that while relative position is more useful than appearance, it doesn’t

contain as much extra information as absolute position. This is particularly interesting given the powerful boost given by appearance to dense features in Sect. 4.2.3.2, although many other details of the models were different. On close examination, we found that at least part of the performance difference between absolute and relative position is due to unreliable detection of the face, the relative position anchor. Rather than simply relying on absolute position information, we will try to improve relative position information, because position relative to a meaningful landmark can carry a lot of useful information. Additionally, absolute position augmentations yield a model that is not translation-invariant, while relative position augmentations are fully translation invariant, as long as the feature and the positional anchor experience the same translation.

### 4.3.2 Body Part Anchors

While the face anchor used by Messing et al. [17] proved unreliable, the underlying idea of anchoring features generated by human activities to body parts has obvious merit. To further investigate body-part augmentations, we must first find a set of reliable body part positions to use as anchors. We do this using a combination of hand-annotation and well-established automatic inference methods. The parts we use are the head, hands, and torsos.

#### 4.3.2.1 Acquiring Parts

We use manual annotation to detect hand and head positions in every 50–100 frames of video. We detect bodies on the same frames using Dalal and Triggs’ person detector [5]. While there are a number of fully automatic body and pose models, we could have used to find these parts automatically, we are less interested in their reliability than in how a set of good detections could be used. Given these periodic part “detections”, we infer the position of parts on every frame in between two detections by using the KLT tracker to track the part forward and backward between those detections, and averaging the position of those forward and backward tracks to find the position at any frame. This technique may seem very simple, but the errors it produces will serve in the place of the noise that would be produced by a fully automatic detection and tracking system.

### 4.3.3 Feature Selection

Messing et al. [17] used discretized positional information, whether absolute or relative to an anchor, as an augmentation for their trajectory descriptor in their mixture model. We will investigate using the head, hand and body augmentations in this way, but we also consider using this information for feature selection.

**Table 4.3** Performance of velocity history features using mean or maximum distance to closest part for feature selection on the URADL dataset. Note that this includes manually detected head and hand parts

Threshold	Percent correct (mean)	Percent correct (max)
All features	69.3 %	69.3 %
50 pixels	65.3 %	44.6 %
100 pixels	70.6 %	66.6 %
200 pixels	70 %	69.3 %

**Table 4.4** Performance of velocity history features using mean or maximum distance to either hand for feature selection on the URADL dataset. Note that hand detection was done manually

Threshold	Percent correct (mean)	Percent correct (max)
All features	69.3 %	69.3 %
50 pixels	56 %	36.6 %
100 pixels	68.6 %	61.3 %
200 pixels	72 %	72 %

Feature selection is the practice of selecting a subset of features to consider for a task. It is usually used to speed up a task by limiting the number of features that must be considered, but it can potentially improve performance by throwing out irrelevant features. Most feature-based work practices some form of implicit feature selection, by not considering, for example, feature candidates that are not at the local maximum of interest point detectors, or ignoring trajectories below a duration threshold. Experiments with Messing et al.’s model were slow enough that they explored hardware acceleration, so the primary function of this investigation of feature selection is to accelerate their model without hurting performance.

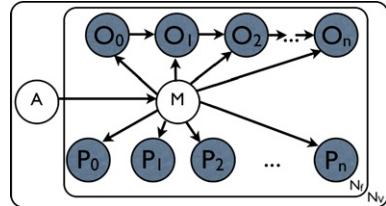
Feature selection was performed by considering all positions along the course of a trajectory feature. Features were thrown out if either their mean distance to a part, or their maximum distance to a part, exceeded a certain threshold {50, 100, or 200 pixels}. Note that the body parts were used only for feature selection. The velocity history descriptor was the only descriptor used for these experiments.

Tables 4.3 and 4.4 show that using feature selection can maintain and even improve accuracy. In particular, we see that selecting features close to the hands can improve performance while increasing speed, as only a fraction of the trajectory features are near the hands, while most interesting object manipulation involves hand movements.

### 4.3.4 Feature Augmentation

Messing et al. [17] discretized position, and used that discretized position at the start and end of a trajectory feature as independent parts (called “augmentations”) of a mixture component’s trajectory model. We consider alternative ways to include the information in position augmentations. One option is to increase the amount of modeling power spent on the position augmentation, by including the position not

**Fig. 4.13** Graphical model for velocity histories with augmentations at each timestep (dark circles denote observed variables)



**Table 4.5** Performance of velocity history features augmented by their position relative to the body on the URADL dataset. Note that this involved no manual detection of parts

Augmentation	Percent correct
Trajectories Alone	69.3 %
Trajectories + Start and End Relative Position	76.6 %
Trajectories + Relative Position at Each Timestep	47.3 %
Start and End Relative Position	46.6 %
Relative Position at Each Timestep	46 %

only at the start and the end of each trajectory, but every position over the course of the trajectory. We call this a temporal augmentation, and model it similarly to Messing et al.’s model of discrete velocity, but without the dependence on the previous observed position. Using  $P_{t,f}$  to denote the position of feature  $f$  at time  $t$ , the new version of Messing et al.’s PDF becomes

$$\begin{aligned}
 P(A, P, O) &= \sum_M P(A, M, P, O) \\
 &= P(A) \prod_f \sum_i^{N_f} \sum_m^{N_m} P(M_f^i | A) P(O_{0,f} | M_f^i) \\
 &\quad \times \prod_{t=1}^T P(O_{t,f} | O_{t-1,f}, M_f^i) P(P_{t,f} | M_f^i)
 \end{aligned} \tag{4.1}$$

The graphical model corresponding to (4.1) is shown in Fig. 4.13.

Table 4.5 clearly shows that by themselves the start and end relative position contains about as much information as the relative position at each time step. Despite this, augmenting the trajectory model with the relative position at each time step gives only a marginal performance boost over just using the relative position at each time step. This strongly suggests that the temporal augmentation is dominating the probability model, and swamping the more useful trajectory model. Given that the performance of the temporal augmentation does not exceed that of the start and end position augmentation, it appears that start and end positions convey all or most of the relevant information.

We also consider a reduced-complexity representation of trajectory position relative to body parts. We investigate augmenting the trajectory not with discretized

**Table 4.6** Performance of velocity history features augmented in one of several ways by their position relative to one of several parts on the URADL dataset. Parts involving manual detection are marked in *italics*

Augmentation type	Body	Head	Left hand	Right hand
Relative Position	76 %	80.6 %	81.3 %	82 %
Absolute Distance: Mean	75.3 %	75.3 %	75.3 %	73.3 %
Absolute Distance: Max	70.6 %	74.6 %	72.6 %	72 %
Binary: Mean Distance < 200	72 %	72.6 %	73.3 %	71.3 %
Binary: Mean Distance < 100	70 %	70.6 %	72 %	70 %
Binary: Mean Distance < 50	69.3 %	67.3 %	71.3 %	72.6 %
Binary: Max Distance < 200	70.6 %	77.3 %	71.3 %	73.3 %
Binary: Max Distance < 100	70.6 %	68 %	67.3 %	72 %
Binary: Max Distance < 50	68 %	70 %	68.6 %	71.3 %

relative position, but with either mean or maximum absolute distance from the trajectory to the part, and a binary variable indicating whether the part’s mean or maximum distance to a feature is within a fixed threshold.

Table 4.6 clearly shows that scalar distance, whether or not it is thresholded, is a less effective representation of position relative to semantic landmarks than is discretized two-dimensional position. This is not very surprising, as distinguishing between “below the hand” and “to the upper right of the hand” are intuitively useful distinction for an activity recognition system to make.

### 4.3.5 Combining Selection and Augmentation

The relative locations of features to body parts can be used for effective feature selection and augmentation of Messing et al.’s trajectory features. Here, we consider the combination of feature selection and augmentation. For augmentations, we only consider Messing et al.’s original discretized two-dimensional relative position at the start and end of trajectories, as the other representations we have explored are not as effective. For feature selection, we eliminate features that exceed a mean distance of 200 pixels from the parts being considered

Table 4.7 shows that once features are augmented, feature selection reduces performance by a small amount, in exchange for reduced runtime. We also see that of the three relative position augmentations we’ve added to Messing et al.’s model, the position of trajectories relative to the positions of the hands is the most informative. This is not very surprising, as the dataset is characterized by broad and fine hand motions. If body parts were efficiently and automatically detected, it is likely that good, fast, translation invariant performance could be achieved on tasks like the activities of daily living in Messing et al.’s URADL dataset [17].

**Table 4.7** Performance of velocity history features selected for proximity to the part(s) in the first row, and augmented by their start and end position relative to the named semantic parts on the URADL dataset, as measured by percent correct using leave-one-out cross validation. The absolute position augmentation is present for comparison only. Note that the first 3 rows (Trajectories alone, Trajectories and Absolute Position, and Trajectories and Body Position) involve no manual detection of parts, and rows in *italics* involve manual detection

Augmentation	No selection	Hands	All parts
Trajectories Alone	68 %	72 %	71.3 %
Traj. + Abs. Pos	80.6 %	81.3 %	83.3 %
Traj. + Body	76.6 %	82.6 %	77.3 %
<i>Traj. + Head</i>	79.3 %	75.3 %	75.3 %
<i>Traj. + Hands</i>	88.6 %	85.3 %	87.3 %
<i>Traj. + Body + Head</i>	82.6 %	78.6 %	78.6 %
<i>Traj. + Body + Hands</i>	86 %	86.6 %	86.6 %
<i>Traj. + Head + Hands</i>	90.6 %	87.3 %	87.3 %
<i>Traj. + Body + Head + Hands</i>	88 %	84 %	88 %

## 4.4 Conclusion

Trajectory descriptors are an increasingly important trend as the field of activity recognition in video grows beyond techniques developed for object recognition in still images. As new trajectory-based methods are introduced, it will become increasingly important to carefully consider all parts of the models that use them. Additionally, as trajectory information becomes more standard, it will be more and more important to be able to seamlessly integrate trajectory information with other information, like appearance, or higher level knowledge of scene structure or pose. In comparing Messing et al.’s sparse generative extended trajectory features with Wang et al.’s discriminative dense trajectories, we have provided a template for how to examine new methods, and choose model components. In addition, by illustrating how to add new and more refined augmentations to Messing et al.’s trajectory model, we have illustrated how to explore which new kinds of information are most beneficial to recognition models, and how to add them. It is our hope that these experiments will inform future work developing trajectory-based activity recognition.

**Acknowledgements** RM thanks the University of Rochester for support and guidance during his graduate education. AT thanks FQRNT for their financial support. CP thanks Ubisoft, NSERC, Google and the University of Rochester for their financial support.

## References

- Black MJ, Yacoob Y (1995) Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. In: ICCV, pp 374–381

2. Blei DM, Ng AY, Jordan MI (2003) Latent Dirichlet allocation. *J Mach Learn Res* 3:993–1022
3. Brox T, Malik J (2011) Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Trans Pattern Anal Mach Intell* 33(3):500–513
4. Cutting JE (1981) Six tenets for event perception. *Cognition* 10:71–78
5. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: Proceedings of the 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)—volume 1—volume 01, CVPR'05. IEEE Computer Society, Washington, pp 886–893
6. Dollár P, Rabaud V, Cottrell G, Belongie S (2005) Behavior recognition via sparse spatio-temporal features. In: VS-PETS, October 2005
7. Jebara T (2003) Images as bags of pixels. In: ICCV
8. Joachims T (1999) Making large-scale SVM learning practical. *Advances in kernel methods—support vector learning*. MIT Press, Cambridge
9. Johansson G (1973) Visual perception of biological motion and a model for its analysis. *Percept Psychophys* 14:201–211
10. Ke Y, Sukthankar R (2004) Pca-sift: a more distinctive representation for local image descriptors. In: CVPR, pp II-506–II-513
11. Laptev I, Lindeberg T (2004) Local descriptors for spatio-temporal recognition. In: Int workshop on spatial coherence for visual motion analysis
12. Laptev I, Marszałek M, Schmid C, Rozenfeld B (2008) Learning realistic human actions from movies. In: CVPR, Anchorage, Alaska, June 2008, pp 1–8
13. Le QV, Zou WY, Yeung SY, Ng AY (2011) Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In: CVPR
14. Lewis D (1998) Naive (Bayes) at forty: the independence assumption in information retrieval. In: ECML
15. Lucas BD, Kanade T (1981) An iterative image registration technique with an application to stereo vision. In: IJCAI, pp 674–679
16. Marszałek M, Laptev I, Schmid C (2009) Actions in context. In: CVPR
17. Messing R, Pal C, Kautz H (2009) Activity recognition using the velocity histories of tracked keypoints. In: ICCV
18. Niebles J-C, Wang H, Fei-fei L (2006) Unsupervised learning of human action categories using spatial-temporal words. In: Proc BMVC
19. Ramanan D, Forsyth DA (2003) Automatic annotation of everyday movements. In: NIPS, December 2003
20. Shi J, Tomasi C (1994) Good features to track. In: CVPR, pp 593–600
21. Sidenbladh H, Black M, Fleet DJ (2000) Stochastic tracking of 3d human figures using 2d image motion. In: ECCV
22. Ullah MM, Parizi SN, Laptev I (2010) Improving bag-of-features action recognition with non-local cues. In: Proc BMVC, pp 95.1–95.11. doi:[10.5244/C.24.95](https://doi.org/10.5244/C.24.95)
23. Wang H, Ullah MM, Klaser A, Laptev I, Schmid C (2009) Evaluation of local spatio-temporal features for action recognition. In: BMVC
24. Wang H, Klaser A, Schmid C, Liu C-L (2011) Action recognition by dense trajectories. In: CVPR

# Chapter 5

## Co-recognition of Images and Videos: Unsupervised Matching of Identical Object Patterns and Its Applications

Minsu Cho, Young Min Shin, and Kyoung Mu Lee

**Abstract** In this chapter, we address the problem of detecting, matching, and segmenting all identical object-level patterns from images or videos in an unsupervised way, called the “co-recognition” problem. In an unsupervised setting without any prior knowledge of specific target objects, it relies entirely on geometric and photometric relations of visual features. To solve this problem, a multi-layer match-growing framework is proposed which explores given visual data by intra-layer expansion and inter-layer merge. We demonstrate the effectiveness of this approach on identical object detection, image retrieval, symmetry detection, and action recognition. These applications will validate the usefulness of co-recognition to several vision problems.

### 5.1 Introduction

In detection and recognition of visual objects in images or actions in videos, most of computer vision approaches require some level of supervision to specify or learn a model [5, 12, 14, 24, 34, 45, 50, 52]. In such cases, labeled images with bounding boxes or uncluttered video clips are usually adopted for the purpose. Recent categorization methods based on latent topic models have proposed weakly supervised or unsupervised learning approaches using a decent amount of training images [23, 53, 55]. In real-world images or videos, however, multiple objects of similar appearance often show up even in a single view. For example, an image can contain replicas or multiple shots of identical objects, and a video clip often includes the same actions in different spatio-temporal regions. Humans have no difficulty

---

M. Cho (✉)

INRIA/École Normale Supérieure, Paris, France

e-mail: [minsu.cho@ens.fr](mailto:minsu.cho@ens.fr)

Y.M. Shin · K.M. Lee

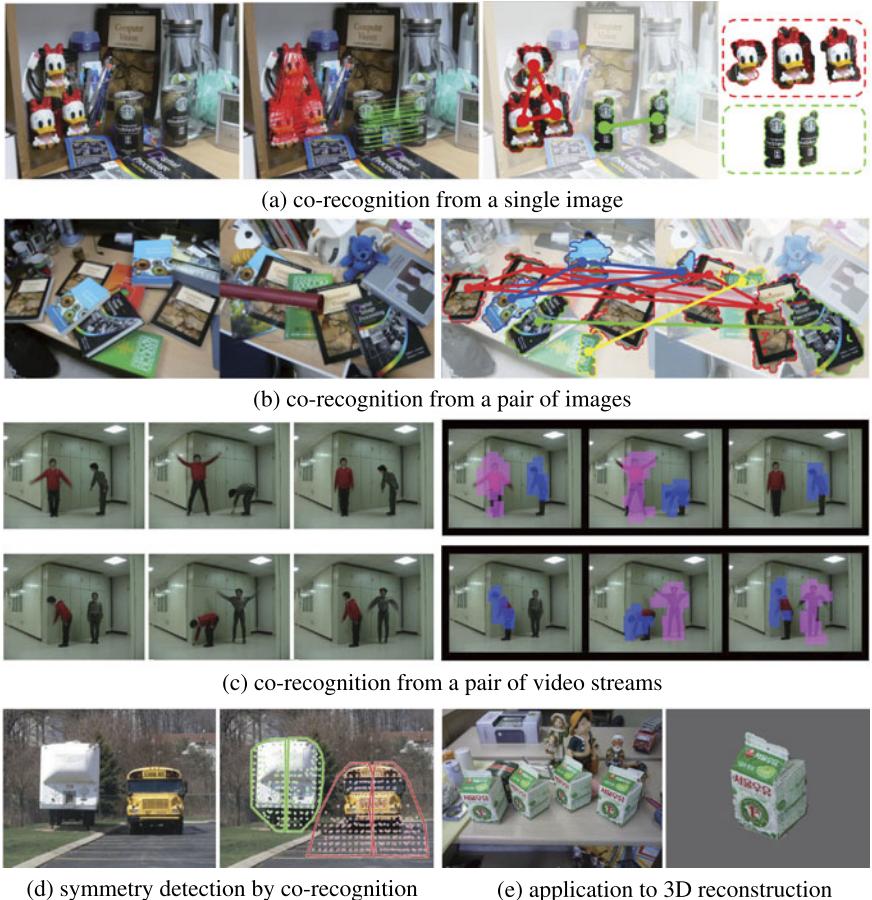
Seoul National University, Seoul, Korea

Y.M. Shin

e-mail: [shinyoungmin@gmail.com](mailto:shinyoungmin@gmail.com)

K.M. Lee

e-mail: [kyoungmu@snu.ac.kr](mailto:kyoungmu@snu.ac.kr)



**Fig. 5.1** Co-recognition results on images and videos. Our method detects and segments all identical objects in images or actions in video streams without supervision, and can be applied to various applications. (a) Two sets of identical objects are detected, matched, and segmented from a single image. (b) Many-to-many object matching is obtained across two images. (c) Two kinds of actions are matched and segmented from video clips. (d) Symmetry detection is done by co-recognition on symmetric feature pairs. (e) Matching and segmentation of different views enables object-based 3D reconstruction even from a single image

in finding those visual objects without any prior information, whereas it is still a challenging problem in computer vision.

In this chapter, we pose a problem of detecting, matching, and segmenting all the identical object-level patterns by considering geometric and photometric relations of visual features. It supposes an unsupervised setting without any prior knowledge of specific target objects. According to the naming conventions of related work [49, 56], we term it *co-recognition*, which emphasizes explicit geometric matching unlike other methods. We extend it to video domain and explore further applications. As shown in Fig. 5.1a, our approach can detect, match, and segment

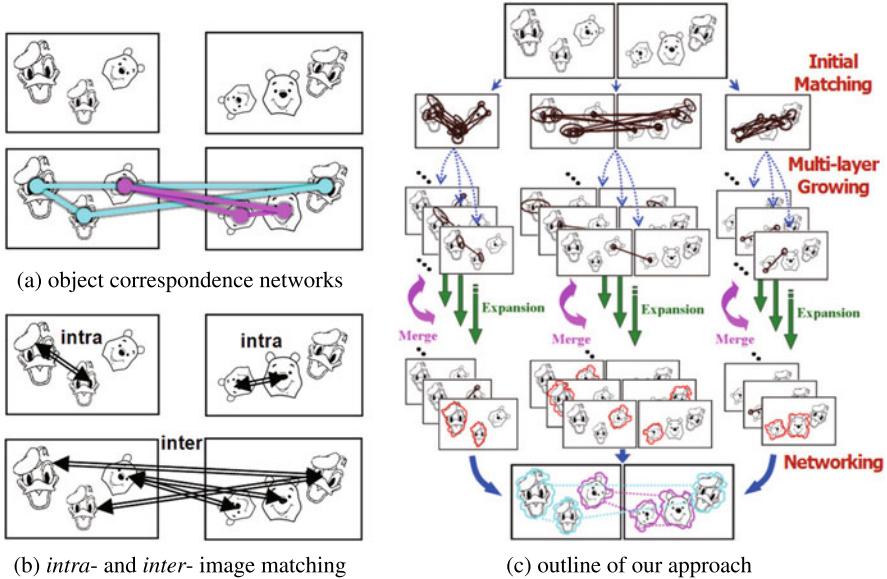
identical objects or actions directly from a single image or video clip. Interestingly, this framework generalizes to *many-to-many* object matching across images as in Fig. 5.1b (or action matching across video streams) because two or more images can be treated as a single image by combining all of them; the problem of discovering many-to-many object matches across the original multiple images is addressed by detecting multiple sets of identical objects within the combined single image. We cast it as a maximum a posteriori (MAP) task on a generative model in a Bayesian formulation. Based on it, a multi-layer match-growing algorithm is proposed to detect and segment all the object-level pairwise correspondences in the given images or video streams. The proposed approach goes beyond the conventional assumptions such as one-to-one object matching constraints or model-test settings, and addresses object matching in general. The resultant object correspondence networks can provide useful information for structural pattern analysis and scene understanding in images and videos. As shown in Fig. 5.1, this chapter will present its several applications on unsupervised identical object detection, image retrieval, symmetry detection, action recognition, and object-based reconstruction.

The remainder of this chapter is organized as follows. In Sect. 5.2, we describe our approach and its formulation, and propose the algorithm for co-recognition. In Sects. 5.3, 5.4, and 5.5, the proposed approach is applied to identical object detection, symmetry detection, and action recognition. Experimental evaluations and comparisons are carried out in each section. Finally, in Sect. 5.6, conclusion and future work are stated with some discussions. The research in this chapter has been extended from our previous papers related to this topic [6, 8–10, 51].

## 5.2 Co-recognition Approach

For the description of our approach, let us take an example of identical object detection problem where two images are given. Relations of matching objects in the images can be represented by networks connecting the object regions as illustrated in Fig. 5.2a. Each isolated network of matching objects means an identical object set detected from the given images. This still holds in general cases where a single or multiple images are given, and the same notion also applies to the spatio-temporal domain of video streams; the more images or videos we consider, the larger networks we obtain. These networks reveal many-to-many object matching relations across images or videos.

Our algorithm is briefly illustrated in Fig. 5.2c. First of all, we separate the whole problem with  $N$  images into smaller subproblems. As shown in Fig. 5.2b, since an object correspondence lies either within an image or across two images, object correspondences can be divided into two kinds, intra-matching within an image and inter-matching across two images. Thus, the subproblems are composed of  $N$  intra-matching and  $\binom{N}{2}$  inter-matching problems. Although they can be considered as one intra-matching problem using a combined image, this decomposition makes the algorithm parallelizable and increases its efficiency by reducing the complexity in



**Fig. 5.2** Overview of our approach. (a) General object matching relation can be represented by correspondence networks. (b) Object correspondences can be divided into two kinds, intra-matching within an image and inter-matching across two images. (c) The figure illustrates the outline of our algorithm, where the goal is to detect, match, and segment identical objects from visual data [10]. For more details, see text

the subsequent match-growing process. Second, based on local feature detectors, we obtain a sufficient number of initial matches. In this step, one-to-many feature matching, that allows each feature to match with multiple matching features, is required to deal with the multiple identical patterns in images. Third, the initial matches are progressively updated and expanded by our multi-layer growing algorithm. At the start, singleton match clusters are created from the initial matches, and each of the clusters is endowed with its *expansion layer* that assigns a space for expansion to the match cluster. Then, the match clusters iteratively grow through *intra-layer expansion* and *inter-layer merge* until the posterior probability model of object correspondences is maximized. Finally, reliable object correspondences are selected from the match clusters and are connected into object networks.

In order to model the object correspondences in an unsupervised way, we introduce a perceptually meaningful entity, termed the *Maximal Correspondence* (MC) which is generalized from [8] and is defined as follows:

1. An MC is a semi-global region (or volume) pair, composed of local matches.
2. Matching regions (or volumes) of an MC are mutually consistent in both geometry and photometry.
3. An MC is maximal in its size so that it cannot be further enlarged.

Supposing the appearances of matching objects are not severely deformed in given images (or videos), an MC from given images (or videos) can be detected as an

object correspondence under two conditions: (1) Each object correspondence lies on different backgrounds in photometry. (2) Object correspondences appear distinguishable each other in geometry. With the first condition unsatisfied, an MC includes a portion of a common background. With the second unsatisfied, multiple object correspondences can be recognized as one. For example, if two objects always appear aligned next to each other, we cannot discern them as separate objects without additional information. These are common intrinsic ambiguities of unsupervised object discovery. In this work, we explore how we can detect object-level correspondences based on the notion of MC and apply it to vision problems, rather than attempt to eliminate its ambiguities completely.

Here, we formally describe the object correspondences in visual data as a set of MCs  $\Theta$  consisting of individual MCs  $\Gamma_i$ :

$$\Theta = \{\Gamma_1, \Gamma_2, \dots, \Gamma_K\}, \quad \Gamma_i = \{\mathcal{M}_{m|i} \mid m = 1, \dots, N_i\}, \quad (5.1)$$

where  $K$  is the number of MCs in  $\Theta$  and  $i$ th MC  $\Gamma_i$  contains its member matches  $\mathcal{M}_{m|i}$ .  $N_i$  denotes the number of member matches in the match cluster  $\Gamma_i$ . We specifically call an MC in images a Maximal Region Correspondence (MRC) while we term an MC in videos a Maximal Cuboid Correspondence (MCC).

In case of images, each match  $\mathcal{M}_{m|i}$  is defined by a local region pair of  $(\mathcal{R}_{m|i}, \mathcal{R}'_{m|i})$  and the homography  $\mathbf{H}_{m|i}$  between them:

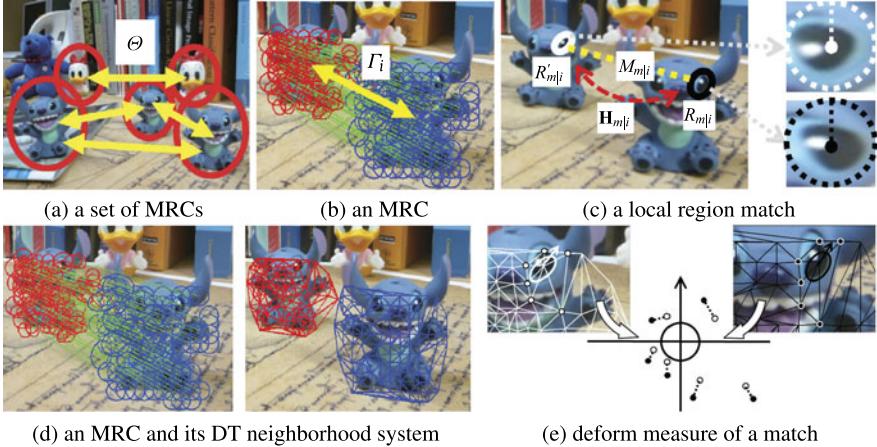
$$\mathcal{M}_{m|i} = \{(\mathcal{R}_{m|i}, \mathcal{R}'_{m|i}), \mathbf{H}_{m|i}\}. \quad (5.2)$$

As shown in Fig. 5.3b–c, each MRC has a pair of corresponding object regions, which in turn consist of local region pairs of matches,  $\mathcal{R}_{m|i}$  and  $\mathcal{R}'_{m|i}$ . Homography  $\mathbf{H}_{m|i}$  transforms the points on  $\mathcal{R}_{m|i}$  into the points on  $\mathcal{R}'_{m|i}$ .

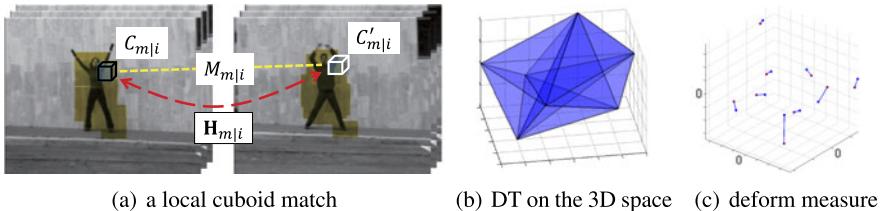
On the other hand, Fig. 5.4a illustrates the MCC and its local cuboid matches in video data. In case of videos, due to its spatio-temporal dimension, each match  $\mathcal{M}_{m|i}$  is expressed by a local cuboid pair of  $(\mathcal{C}_{m|i}, \mathcal{C}'_{m|i})$  and its homography  $\mathbf{H}_{m|i}$ :

$$\mathcal{M}_{m|i} = \{(\mathcal{C}_{m|i}, \mathcal{C}'_{m|i}), \mathbf{H}_{m|i}\}. \quad (5.3)$$

The detailed description of regions or cuboid matches depends on the type of features in use. In our model of local matches for images, a local region  $\mathcal{R}$  is represented by an affine region [41], which has its characteristic orientation estimated by the gradient histogram of the local region [34]. Given two local regions  $\mathcal{R}_{m|i}$  and  $\mathcal{R}'_{m|i}$ , its affine homography  $\mathbf{H}_{m|i}$  from  $\mathcal{R}_{m|i}$  to  $\mathcal{R}'_{m|i}$  is determined by the elliptical parameters and the orientations [10]. Similarly, as for the local matches in videos, we define  $\mathcal{C}$  as a local cuboid in 3D coordinates, which occupies  $\sigma$  pixels in width and height, and  $\tau$  pixels in temporal length. Here, the homography  $\mathbf{H}$  transforms a cuboid to another cuboid with translation and scale change.  $\mathbf{H}$  becomes a transformation function with 5 degrees of freedom: translation in each coordinates  $(d_x, d_y, d_t)$  and spatio-temporal scales  $(s_\sigma, s_\tau)$ .



**Fig. 5.3** A model of Maximal Region Correspondences (MRC). **(a)** Object correspondences can be represented by a set of MRCs. **(b)** Each MRC consists of local region matches. **(c)** Each local region match is expressed by two affine regions and the homography between them. **(d)** A neighborhood system of the MRC is constructed by Delaunay triangulation. **(e)** Based on this neighborhood system, geometric deformation of a member match is defined as the average transfer error of neighbor points on a normalized domain [10]



**Fig. 5.4** Maximal Cuboid Correspondence (MCC) in video. **(a)** An MCC is composed of local cuboid matches. A local match is defined by two cuboids and the homography between them. **(b)** Three-dimensional Delaunay triangulation on the cuboid centers defines the neighborhood relation. Note that only a part of the DT is displayed here for visibility. **(c)** After normalizing the neighbors to have unit Euclidean distance on average, the geometric deformation is obtained by calculating the average discrepancy between corresponding points

### 5.2.1 A Bayesian Formulation for Object Correspondence

Based on the concepts described so far, we design a Bayesian model of MCs. The posterior probability  $p(\Theta|\mathcal{I})$  is the probability of  $\Theta$  being a set of MCs given a set of images (or videos)  $\mathcal{I}$ . In the Bayesian framework, it is decomposed as  $p(\Theta|\mathcal{I}) = p(\mathcal{I}|\Theta)p(\Theta)$ . In our model, the prior probability  $p(\Theta)$  evaluates the geometric coherency and maximality of MCs, and the likelihood probability  $p(\Theta|\mathcal{I})$  represents the photometric coherency of MCs. All these terms are defined in the following subsections.

### 5.2.1.1 Prior

In order to design a robust prior model, we endow each MC  $\Gamma_i$  with a neighborhood system that defines the neighborhood relation of each match  $\mathcal{M}_{m|i}$  in  $\Gamma_i$  and adaptively varies with evolution of  $\Gamma_i$ . Suppose an undirected graph  $G_i = (\mathcal{V}_i, \mathcal{E}_i)$  where the member matches of  $\Gamma_i$  consist in node set  $\mathcal{V}_i = \{\mathcal{M}_{m|i}\}$ . Edge set  $\mathcal{E}_i$  is constructed by the Delaunay triangulation (DT) [47] on the center points  $\mathbf{c}_{m|i}$  of  $\mathcal{R}_{m|i}$  as shown in Fig. 5.3d. Then, the neighbor set of  $\mathcal{M}_{m|i}$  is defined as  $\mathcal{N}_{\mathcal{M}_{m|i}} = \{\mathcal{M}_{n|i} \mid (\mathbf{c}_{m|i}, \mathbf{c}_{n|i}) \in \mathcal{E}_i\}$ . Likewise, for video data, 3D neighborhood relations are defined on spatio-temporal ( $x, y, t$ ) coordinates as shown in Fig. 5.4b. Since DT has low computational complexity of  $\mathcal{O}(N_i \log N_i)$  and can be incrementally updated [13], the DT-based neighborhood system is efficiently constructed and evolved along expansion/merge moves.

To evaluate the geometric consistency of an MC  $\Gamma_i$ , we use this neighborhood system. For images, consider a local region match  $\mathcal{M}_{m|i}$  in  $\Gamma_i$ , and let  $T_{m|i}$  and  $T'_{m|i}$  the transformations that map  $\mathcal{R}_{m|i}$  and  $\mathcal{R}'_{m|i}$  onto a unit circle, respectively. As shown in Figs. 5.3e and 5.4c, the centers of its neighbor nodes  $\mathcal{N}_{\mathcal{M}_{m|i}}$  are normalized into the unit domain by  $T_{m|i}$  and  $T'_{m|i}$ , and the distances between the corresponding points in the unit domain are computed as deformation measures [9]. For videos, the normalizing transformations  $T_{m|i}$  and  $T'_{m|i}$  denotes the transformations that map  $\mathcal{C}_{m|i}$  and  $\mathcal{C}'_{m|i}$  onto a unit cuboid [51].

The local deformation measure of  $\mathcal{M}_{m|i}$  is formulated as

$$d_G(\mathcal{M}_{m|i}) = \frac{1}{|\mathcal{N}_{\mathcal{M}_{m|i}}|} \sum_{\mathcal{M}_{n|i} \in \mathcal{N}_{\mathcal{M}_{m|i}}} |T_{m|i}(\mathbf{c}_{n|i}) - T'_{m|i}(\mathbf{c}'_{n|i})|, \quad (5.4)$$

where  $\mathbf{c}_{n|i}$  and  $\mathbf{c}'_{n|i}$  denote the center points of  $\mathcal{R}_{n|i}$  and  $\mathcal{R}'_{n|i}$ , or  $\mathcal{C}_{n|i}$  and  $\mathcal{C}'_{n|i}$ . This measure represents the average transfer error of neighbors on the normalized domain. Summing up all the local deformations, the geometric energy of the MC  $\Gamma_i$  is defined as

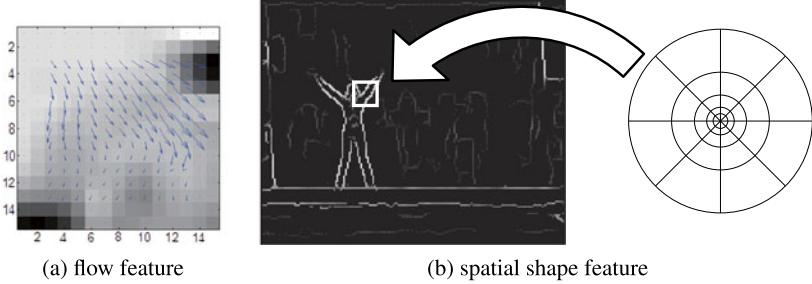
$$E_G(\Gamma_i) = \sum_{m=1}^{N_i} d_G(\mathcal{M}_{m|i}). \quad (5.5)$$

Due to the DT-based neighborhood system, it effectively adapts to variation of the MC configuration and generates robust deformation measures for smoothly deformed surfaces or volumes.

In addition, to grow MCs to larger ones satisfying the maximality in their sizes, rewards for both expansion and merge are required in the prior model. Thus, we design the maximality energy as

$$E_M(\Theta) = \sum_{i=1}^K (-N_i - |\Delta_i|), \quad (5.6)$$

where  $N_i$  and  $|\Delta_i|$  denote the number of member matches and the number of Delaunay triangles of  $\Gamma_i$ , respectively. Since expansion of any MC increases the total



**Fig. 5.5** Features used in the video data. **(a)** The flow feature in the cuboid is represented as the field of optical flow displacement vectors computed between the consecutive frames. **(b)** The local shape of the edge map is described by the log-polar descriptor. Each pixel within the log-polar bin casts a vote to the corresponding histogram channel. Then, the shape feature of the cuboid is described by the normalized histogram

number of local matches, the first term encourages them to expand. On the other hand, the second term provides a reward for merge of MCs because merge makes additional triangles of DT in a merged MC.

### 5.2.1.2 Likelihood

The likelihood probability  $p(\mathcal{I}|\Theta)$  evaluates the photometric coherency based on local appearance of images or videos  $\mathcal{I}$ .

For images, we define the dissimilarity [8] of two local image regions in a member match  $\mathcal{M}_{m|i}$  by

$$d_P(\mathcal{M}_{m|i}) = 1 - \text{NCC}(\mathcal{R}_{m|i}, \mathcal{R}'_{m|i}) + \frac{\text{dRGB}(\mathcal{R}_{m|i}, \mathcal{R}'_{m|i})}{100}, \quad (5.7)$$

where  $\text{NCC}(\cdot, \cdot)$  is the normalized cross-correlation between the gray patterns of two regions, while  $\text{dRGB}(\cdot, \cdot)$  is the average pixel-wise Euclidean distance in a normalized RGB color-space [8, 14].

For videos, we use both optical flow and spatial edge shape information in local spatio-temporal cuboids. As shown in Fig. 5.5, the optical flow for a cuboid is computed across adjacent frames by the Lucas–Kanade algorithm [36], and the edge map is obtained by the Berkeley edge detector [38]. The dissimilarity of two local cuboids in a member match  $\mathcal{M}_{m|i}$  is described by

$$d_P(\mathcal{M}_{m|i}) = \left( \frac{\sum_{x,y,t} \|F_{x,y,t} - F'_{x,y,t}\|}{\frac{1}{2} \sum_{x,y,t} (\|F_{x,y,t}\| + \|F'_{x,y,t}\|)} \right)^2 + \left( \frac{\|D - D'\|}{\|D\| + \|D'\|} \right)^2, \quad (5.8)$$

where the vector fields  $F_{x,y,t}$  and  $F'_{x,y,t}$  denote the optical flow vectors at  $(x, y, t)$ , and  $D$  and  $D'$  means the edge descriptor extracted from  $\mathcal{C}$  and  $\mathcal{C}'$ , respectively.

$\|\cdot\|$  denotes L2 norm.  $D$  is a 40-dimensional log-polar descriptor similar to the *shape context* descriptor [2] with 5 bins of log-polar location grid and 8 orientation bins.

Assuming that local matches in images or videos are mutually independent in photometric coherency, the photometric energy of MC  $\Gamma_i$  is measured by

$$E_P(\Gamma_i) = \sum_{m=1}^{N_i} d_P(\mathcal{M}_{m|i}). \quad (5.9)$$

Finally, using Eqs. (5.5)–(5.7), we formulate the overall prior and the likelihood in the posterior model as follows:

$$p(\Theta) \propto \exp \left( - \sum_{i=1}^K E_G(\Gamma_i) - E_M(\Theta) \right), \quad p(\mathcal{I} | \Theta) \propto \exp \left( - \lambda_P \sum_{i=1}^K E_P(\Gamma_i) \right), \quad (5.10)$$

where  $\lambda_P$  controls the balance between the likelihood and the prior in the model. Hence, given image or video observation  $\mathcal{I}$ , an optimal set of MCs  $\Theta^*$  is obtained by maximizing the posterior probability  $p(\Theta | \mathcal{I})$  as

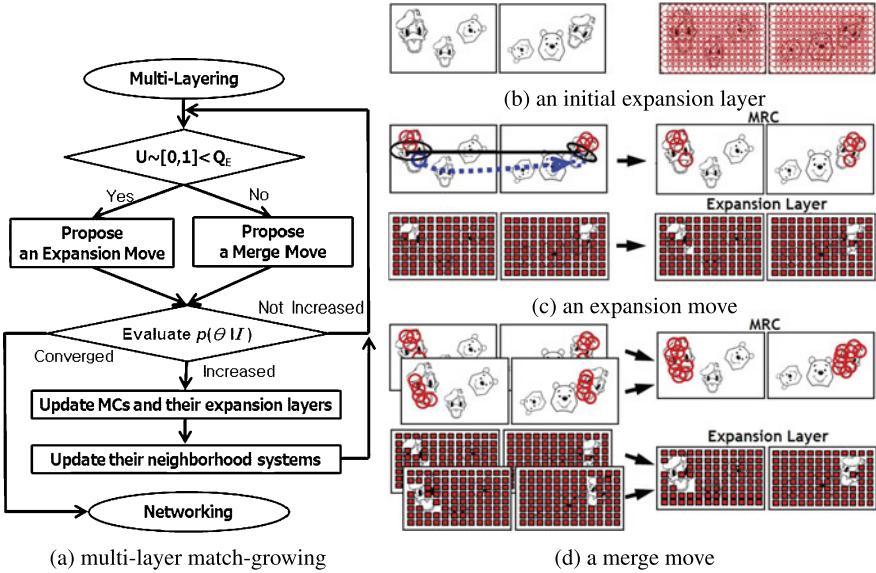
$$\Theta^* = \arg \max_{\Theta} p(\mathcal{I} | \Theta) p(\Theta). \quad (5.11)$$

### 5.2.2 Algorithm for Co-recognition

In this section, we explain our algorithm for co-recognition. As briefly illustrated in Fig. 5.2, the proposed algorithm consists of three main phases: initial matching, multi-layer growing, and object correspondence networking, which will be described one by one in the following subsections.

#### 5.2.2.1 Initial Matching

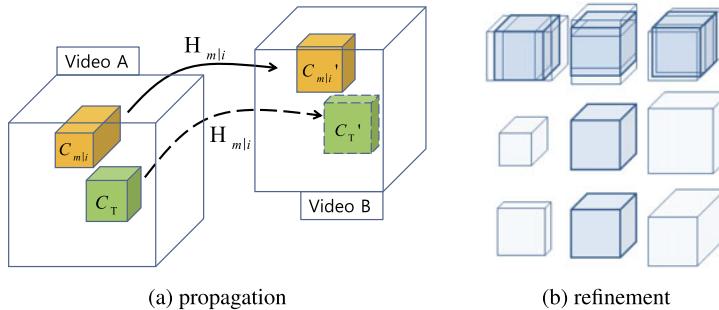
To form seeds for MCs, initial matches are established using local feature detectors [27, 29, 42, 57]. Basically, feature pairs having similar descriptors consist in the initial seed matches. For images, we adopted Harris-affine [40] and MSER [39] detectors and the SIFT descriptor [34]. For videos, the spatio-temporal interest point (STIP) detector [27] and the HOG-HOF descriptor [29] are used. In initial matching, the nearest neighbor (NN) matching schemes are widely used in the literature [8, 14, 22, 34, 52]. The NN methods, however, severely degrade inlier detection in the presence of multiple identical objects or similar patterns because in those cases the features have multiple matching features rather than the NN feature. To avoid this problem, we allow each feature to pair with multiple features under a loose threshold; all the matching feature pairs exceeding a loose similarity threshold are simply collected among the possible feature pairs. Matching a feature with itself is avoided in intra-matching within one image.



**Fig. 5.6** Multi-layer match-growing. **(a)** After multi-layering is performed using the initial seed matches, the iterative growing loop starts. At each iteration, the algorithm grows the matches by intra-layer expansion or inter-layer merge, and increases the posterior probability  $p(\Theta|I)$ . The growing procedure is iterated until  $p(\Theta|I)$  no longer increases. **(b)** Each expansion layer consists of overlapping circular regions covering the image domain. **(c)** In an expansion proposal, a cluster attempts to expand a region (*blue circle in the bottom*) using a support match (*solid black line*). When the proposal is accepted, the MC and its expansion layer is updated. **(d)** In a merge proposal, two clusters attempt to merge into one. When the proposal is accepted, the MC and their expansion layers are combined into one. In **(c)** and **(d)**, local regions in expansion layers are represented by *red squares* for better visualization. Expansion and merge in videos works in the similar manner but in spatio-temporal dimensions. For details, refer to [9, 10]

### 5.2.2.2 Multi-layer Growing

For growing initial seed matches to object-level correspondences, we propose a multi-layer growing algorithm driven by expansion/merge moves. The growing procedure is summarized in Fig. 5.6a. First, multi-layering is performed using the initial seed matches. Each initial seed match forms an initial singleton MC having its own expansion layer that provides space for expansion. As shown in Fig. 5.6c, each expansion layer consists of an overlapping circular grid of regular local regions that covers the entire image [8] or an overlapping spatio-temporal grid that covers the entire video [51]. In the iterative growing loop, expansion is proposed with probability  $Q_E$  or merge with probability  $1 - Q_E$ . In an expansion proposal illustrated in Fig. 5.6c, a match in an MC is selected as a *supporter* match (shown as the black solid line), and a local region around it is chosen as a *target* region (shown as the blue circle) on the expansion layer of the MC. Then, the target region establishes a new match (the blue dotted arrow), which is propagated using the support match and refined by a local search [9]. Likewise, for videos as depicted in Fig. 5.7, an expansion



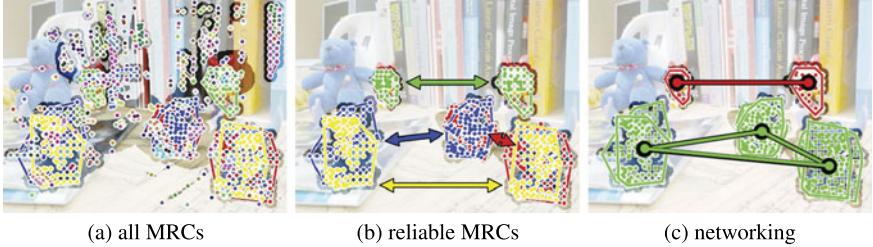
**Fig. 5.7** Expansion move in video data. **(a)** The support match  $\mathcal{M}_{m|i}$  propagates a target cuboid  $C_T$  by applying transform  $H_{m|i}$ . **(b)** Local search around  $H_{m|i}$  along the coordinate axis (*top row*), spatial scale  $s_\sigma$  (*second row*), and temporal scale  $s_\tau$  (*third row*) refines the new match to adapt to the deformation

sion proposal propagates a new cuboid match and then refines it by a local search in a 5-dimensional parameter space [51]. If the expansion proposal is accepted, the propagated match is included in the MC and the target region is eliminated from the expansion layer. On the other hand, in a merge move proposal illustrated in Fig. 5.6d, two geometrically similar MCs are selected and proposed to merge. If the merge move proposal is accepted, the expansion layers are also combined into their intersection. The algorithm accepts the expansion/merge proposals when it increases the posterior probability  $p(\Theta|\mathcal{I})$ . The growing procedure is iterated until  $p(\Theta|\mathcal{I})$  no longer increases.

Although the multi-layer approach multiplies expansion layers at the start, merge moves gradually reduce the number of layers and guide expansion moves to concentrate on potentially matching region. Likewise, expansion moves also guide merge moves to find compatible clusters by gradually growing them. Through these co-operative moves, our algorithm efficiently finds object correspondences in spite of significant number of outliers in initial matches. Therefore, while the multi-layer growing algorithm maximizes the posterior probability of MCs in a greedy manner, its expansion/merge proposals efficiently drive the solution to a good local optimum. We refer the reader to [9, 10] for details concerning the expansion/merge proposals.

### 5.2.2.3 Networking of Maximal Correspondences

To reveal the relations of detected object regions, we establish the object correspondence networks from the set of MCs  $\Theta^*$ . Since the resultant set of MCs usually includes trivial MCs arising from outliers of initial matches, we first eliminate such unreliable MCs from  $\Theta^*$ . Typically, object region correspondences are likely to grow larger and have distinctive textures in their regions. Thus, we evaluate the reliability of an MC by its expanded area and its mean variance of the intensity patterns, and discard MCs not satisfying the threshold values. Then, the reliable MCs are considered as object correspondences and connected to construct the networks



**Fig. 5.8** Object correspondence networking. **(a)** Each MRC obtained from the growing step is represented by convex hull region pairs of the same color. Many small and spurious MRCs are observed arising from outliers of initial matches. **(b)** Based on the reliability criteria, MRCs are selected as object correspondences. **(c)** Reliable MRCs are connected by SL-HAC based on region overlaps, and constitutes the object correspondence networks [10]

according to their overlapping regions. In this work, we use a simple and popular algorithm of the Single-Link Hierarchical Agglomerative Clustering (SL-HAC) [20] to group the MCs. Similarity between two MCs is defined as the ratio of overlapping area to the area of the smaller MC region, and SL-HAC sequentially connects the most similar MC pairs until the similarity becomes less than 0.8. As shown in Fig. 5.8, the procedure assembles MRCs of the given image into networks, each representing connected object correspondences, and the detected object regions are all classified into sets of identical objects. The same scheme can be used likewise for networking MCCs in videos. Notably, this networking scheme does not require the pre-determined number of identical object sets, and is robust to missing object correspondences since others can provide indirect pathways. More advanced clustering algorithms could be adopted for this purpose as well. In particular, the partial linkage HAC [7] or the noise-robust spectral clustering [32] could make the networks robust to falsely detected MCs.

### 5.3 Unsupervised Object Detection, Matching, and Segmentation

In this section, we demonstrate the proposed method on unsupervised object detection and segmentation tasks. Among early works closely related to this topic are co-segmentation [49], co-saliency [56], and common visual pattern discovery [59]. Rother et al. [49] defined co-segmentation as segmenting common regions from two images. They exploited a generative MRF-based graph model and the color histogram similarity measure. Toshev et al. [56] proposed co-saliency matching that searches for region pairs with strong intra-image coherency and high inter-image similarity. Yuan and Wu [59] used spatial random partition to discover common visual patterns from a collection of images. However, none of these methods recognize multiple common objects as distinct entities, and they do not consider sufficient geometrical consistency in the detected region. To handle these limitations, we adopt a match-growing approach that has been developed in several computer

vision tasks to boost initial putative matches and eliminate the matching ambiguities [14, 31, 52, 54, 58]. Recently, Cho et al. [8] and Kannala et al. [22] used it to demonstrate unsupervised object matching beyond the limitation of conventional model-test settings in image matching and registration. All these previous methods however, commonly rely on the restrictive assumptions of one-to-one constraints both in feature matching and in object matching to make their problems simpler.

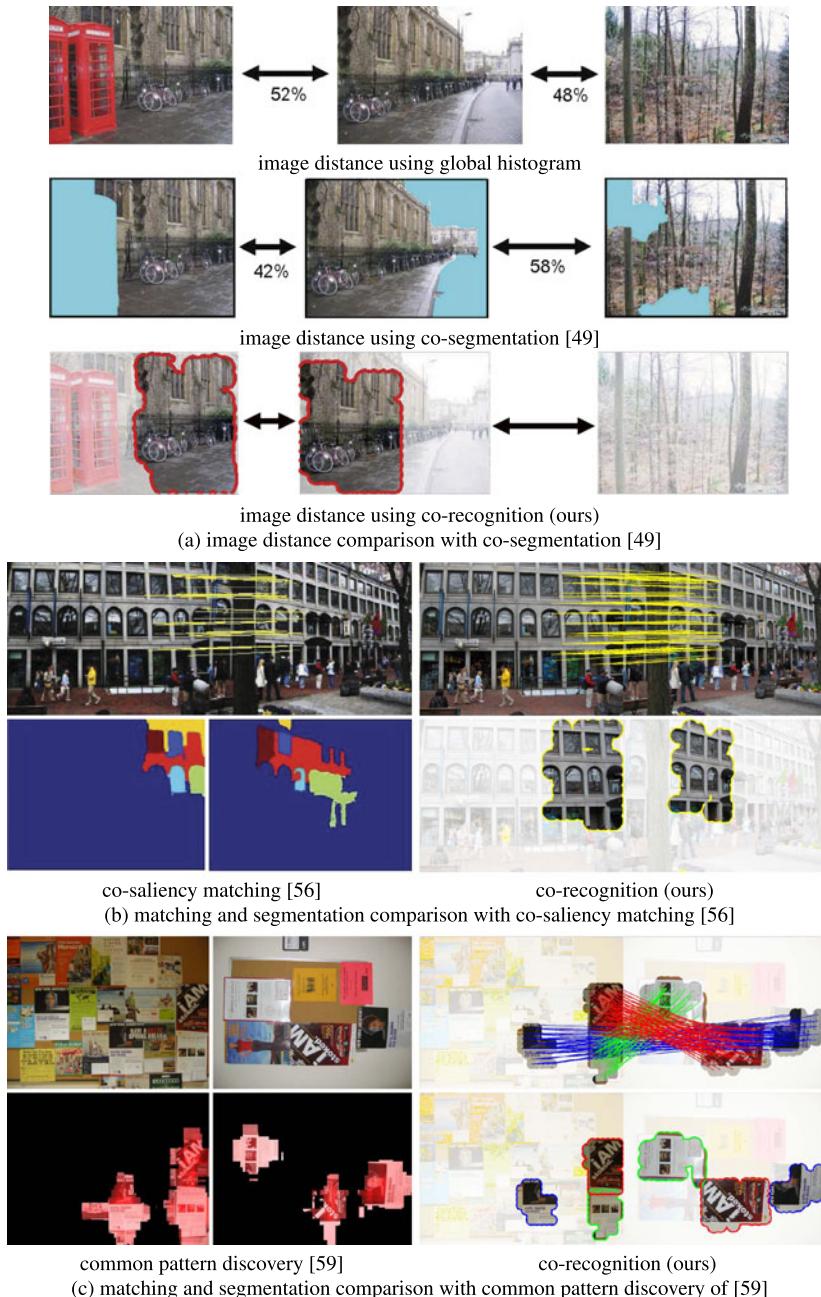
The experimental results in this section suggest that our method can solve general problems of many-to-many object matching, and also outperforms the previous methods [8, 22, 49, 56, 59] even under the one-to-one constraint. The experiments were performed under the following settings. In the initial matching step, affine region detectors of MSER [39] and Harris-affine [40] are used for extracting features, and the SIFT descriptor [34] to measure the similarity between the features. The threshold value for initial matching was set to 0.4 of distance in the SIFT descriptor space. The multi-layer growing was performed with  $Q_E = 0.9$ ,  $\lambda_P = 2.0$ . In networking step, the thresholds for reliable MRCs as the expanded area over 3 % of the given image and the mean intensity variance over 0.005.

### 5.3.1 Common Object Detection and Segmentation

In the first experiment, we perform common object detection and segmentation, which is restricted to the case of one-to-one object matching, and compare our results with those of other related methods [22, 49, 56, 59]. All their results are borrowed from the papers.

Fig. 5.9 shows comparative examples with the methods of co-segmentation [49], co-saliency matching [56], common pattern discovery [59]. Co-segmentation, which segments a common part based on a Markov random field model encoding both of spatial coherency and histogram similarity, showed that its result can be used as more robust similarity measure than conventional global histogram comparison [49]. As shown in Fig. 5.9a, our co-recognition provides an even better measure than both co-segmentation and global histogram comparison. Since co-recognition discovers common image regions based on dense correspondences, it does not suffer from accidental similarity of color histogram and gives a more discriminative measure. Figure 5.9b compares ours with co-saliency matching, which jointly segment and match common region based on spectral analysis [56]. While co-saliency matching does not cover all the common regions and gives inaccurate region correspondences, co-recognition generates more dense and accurate region matching. Similarly, it also show better performance than common pattern discovery of [59] in Fig. 5.9c. Unlike all these methods, co-recognition provides object-level correspondence as well as its dense correspondence.

Figure 5.10 shows comparative examples on segmentation of deformed object with the quasi-dense matching method [22] and single-layer match-growing [8]. As can be observed in the detailed segmentation boundaries, other methods fail to segment severely deformed regions, for example, the corner part of the deformed magazine, but our method discovers the deformed common parts with better accu-



**Fig. 5.9** Comparison with other methods. **(a)** Co-recognition gives a more discriminative measure of image distance than co-segmentation of [49] as well as conventional global histogram comparison. **(b)** Co-recognition generates more dense and accurate matching than co-saliency matching [56]. **(c)** Co-recognition achieves better segmentation and matching than common pattern discovery of [59]



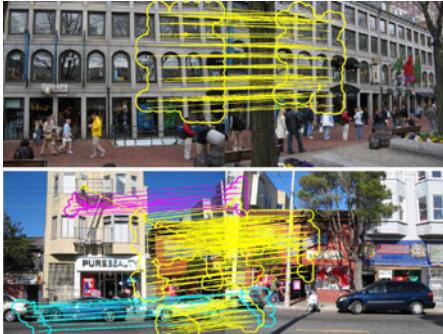
**Fig. 5.10** Common object detection and segmentation. **(a)** Compared to results of quasi-dense matching [22] and single-layer growing [8], the proposed multi-layer growing shows better robustness to non-rigid deformation so that it covers the whole deformed part more accurately. **(b)** Two example results on the image pairs from the SNU dataset are shown

racy than quasi-dense matching as well as single-layer match-growing. It shows that the proposed multi-layer growing based on the dynamic neighborhood system effectively adapts to deformed surfaces of the objects. We performed quantitative evaluation and comparison on the SNU dataset,<sup>1</sup> that is organized for common object matching and segmentation [8]. The results of segmentation accuracy are reported in Table 5.1, where the higher hit ratios (Hit) and the lower background ratios (Bk) represents the better results. It indicates that the proposed multi-layer growing gives better performance than the previous single-layer growing used in [8]. Figure 5.10b shows two of the results, where detected common objects are delineated by the same color and the background is faded. Object boundaries could be further refined by seeded segmentation algorithms [26, 30, 48] using the current result as seeds.

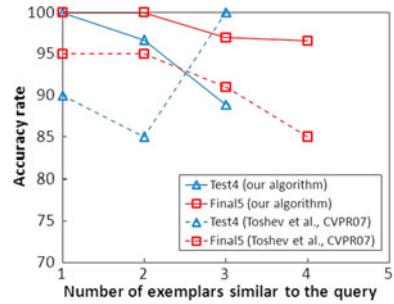
<sup>1</sup><http://cv.snu.ac.kr/~corecognition>

**Table 5.1** Segmentation performance on the SNU dataset [8]

Data	Mickey's	Minnie's	Jigsaws	Toys	Books	Bulletins	Avg.	[8]
Hit (%)	84.4	87.8	81.6	85.2	97.2	93.8	88.3	85.5
Bk (%)	19.4	31.1	19.4	20.8	11.3	15.2	19.5	22.4



(a) co-recognition on ICCV2005 datasets



(b) accuracy rate for Test4 and Final5

**Fig. 5.11** Image retrieval by co-recognition. (a) Co-recognition deals with object-level correspondence, which is higher than segment-level correspondence. (b) Comparison with co-saliency matching [56] on ICCV2005 datasets

### 5.3.2 Image Retrieval

In the second experiment, we have conducted the image retrieval as in [56] on the ICCV 2005 place recognition contest datasets. Each of two datasets (*Test4* and *Final5*) has been split into exemplar and query set. *Test4* has 19 query images and 9 exemplar images, while *Final5* has 22 query images and 16 exemplar images. Each of query images is compared with all exemplar images, and all the matched image pairs are ranked according to the total area of reliable MRCs. For every query image having at least  $k$  similar exemplars, the accuracy rate is evaluated with how many of them are included in top  $k$  ranks.

The result in Fig. 5.11 reveals that our co-recognition outperforms co-saliency matching [56] in this experiment. The reason can be explained by comparing our result of the top in Fig. 5.11a with the result of the same pair in [56]. Co-recognition deals with object-level correspondences, which are higher than segment-level correspondences as [56], our method generates larger, denser, and more accurate correspondences.

### 5.3.3 Identical Object Matching and Segmentation

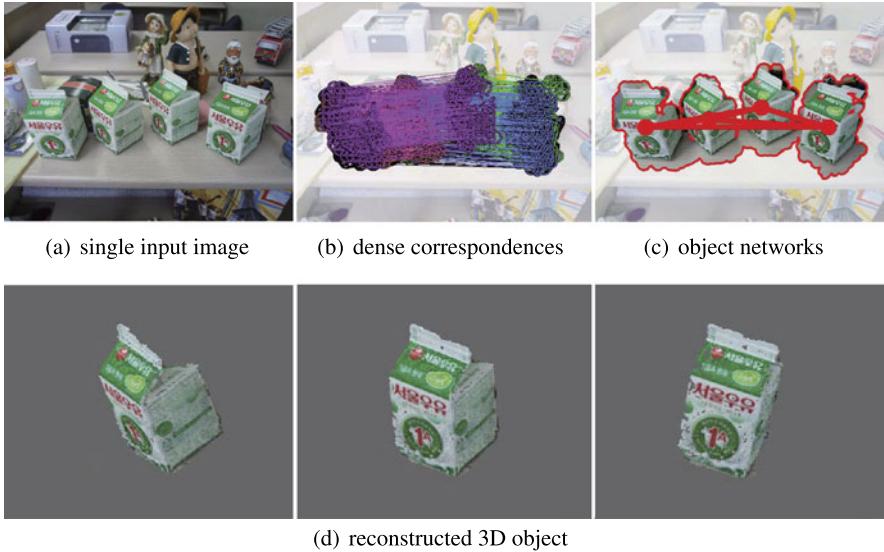
In the third experiment, we tested the proposed method on a more general problem beyond the one-to-one matching assumptions, which the previous methods



**Fig. 5.12** Unsupervised object matching and segmentation on single images. For each result, the input images, segmentation with object correspondence networks, and their classified regions are shown. *Each color* illustrates the identity of an object. In the *last row*, additional results are shown on images with repetitive patterns. For details, see text

[8, 22, 49, 56, 59] cannot deal with. This experiment aimed at demonstrating unsupervised identical object matching and segmentation on various real-world images. For evaluation, we constructed a dataset of 30 images, each image of which has several identical objects with occlusion and clutter.

Figure 5.12 shows some results from the dataset. The object boundaries are determined based on the local regions of MRCs, and delineated by lines of different colors that indicate which object set they belong to. The object correspondence networks are represented by thick lines connecting detected object regions. In the third



**Fig. 5.13** 3D object reconstruction from a single image. The detected regions of identical objects from a single image provide multiple-views of an object, which are used to reconstruct 3D shape of the object. See text

column of each result, object regions of each identical object are displayed in groups to better understand the object matching results. Our method provides impressive detection and segmentation results in spite of deformation, occlusion, and clutters. Although some false pairwise object correspondences were detected and some were missing, most of the identical objects were correctly localized and classified into identical object groups by the networking step. For evaluating the detection accuracy of the pairwise object correspondences, we computed recall and precision rates for our dataset considering all the possible pairwise correspondences. For the detection criterion, we used the standard detection criterion of 50 % area overlap with the ground truth. The average precision and recall rates are 0.92 and 0.75, respectively. The loss of the recall rate was mainly due to the challenging cases of densely stacked objects as shown in the bottom of Fig. 5.12. In those results, several identical objects often merged into one, thus object correspondences were not distinguishable within the merged MRCs. For more results on the dataset, refer to [10].

### 5.3.4 Application to 3D Reconstruction

Since co-recognition detects and segments identical objects in an unsupervised way, we can obtain multiple views of an object from the result and use them for the object-based 3D reconstruction. It enables multi-view reconstruction even from a single image. Figure 5.13 shows such a result of unsupervised 3D object reconstruction.

Given a single image containing 4 instances of the same object, the differences in relative camera poses between identical objects offer multi-view observations of the object. Each of the detected object regions is separated and treated as independent images of the same object taken at different viewpoints. The separated segments are camera-calibrated using the conventional structure-from-motion algorithm [17]. Here, we assumed the square pixel aspect ratio, and exploited the prior information that all the separated image segments have same intrinsic camera parameters. The final 3D shape of the object is obtained using the multi-view stereo reconstruction of Furukawa et al.'s [16]. As shown in Fig. 5.13d, the rendered images of the target object in novel views reveal that the 3D model of the object is successfully reconstructed from the single image. This approach enables to reconstruct multiple objects from unsupervised images.

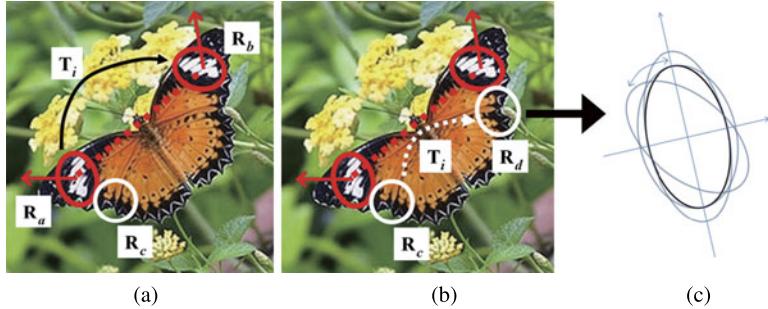
## 5.4 Symmetry Detection

A variety of symmetries occur in nature, living organisms, and manufactured artifacts, and provide humans with pre-attentive cues that enhance object recognition. Despite the long history of the research, the recent performance evaluation [46] shows that we are still short of a robust and widely applicable symmetry detector. The previous methods for symmetry detection can be broadly classified into global and local feature-based methods. The global methods [25, 37] treat the entire image as a signal from which symmetric patterns are inferred, but are limited to detecting a single incidence of symmetry, and also greatly influenced by background clutters. On the contrary, the local feature-based methods [11, 35] use local features such as edges, contours, boundary points, and regions to detect symmetry by grouping symmetric sets of local features. Their main advantage is to more efficiently detect local symmetries against background clutters in images that are not globally symmetric. However, they are largely influenced by feature detection step, and cannot exploit further information beyond the detected features.

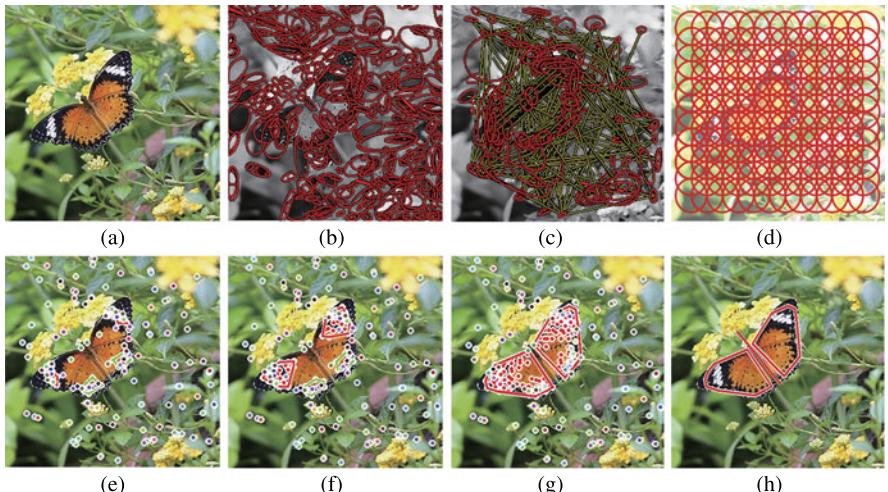
Co-recognition approach can apply to localizing and segmenting symmetric patterns [6]. It overcomes the limitations of the previous local-feature based approaches by efficiently exploring the image space beyond the detected symmetric features. Multiple clusters of consistent symmetric feature pairs are directly detected in our growing process without conventional voting procedure like the Hough transform or RANSAC used in the local feature-based methods [11, 35]. Here, we aim to detect and segment the bilaterally symmetric patterns.

For symmetry detection, the algorithm takes initial matches of symmetric feature pairs and grows them in a symmetric manner. The main difference lies on *symmetric expansion* in multi-layer growing as illustrated in Fig. 5.14. Suppose that a local symmetry match  $(\mathcal{R}_a, \mathcal{R}_b)$  and its neighboring region  $\mathcal{R}_c$  are given. Then, using the reflective homography  $T_i$  from  $\mathcal{R}_a$  to  $\mathcal{R}_b$ , a new symmetric match  $(\mathcal{R}_c, \mathcal{R}_d)$  is propagated and refined in expansion.

A brief overview of the approach is illustrated in Fig. 5.15. Using the appearance around the detected features and its mirrored features, we obtain potential symmet-



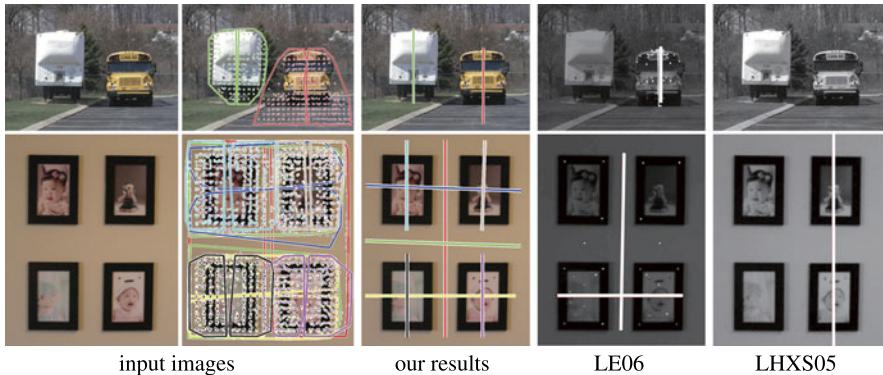
**Fig. 5.14** Symmetric expansion in multi-layer growing. **(a)** A support match ( $\mathcal{R}_a, \mathcal{R}_b$ ) and its neighboring region  $\mathcal{R}_c$ . **(b)** Propagation of the region  $\mathcal{R}_c$  by symmetric homography of the supporter match. **(c)** Refinement by locally fitting the ellipse parameters



**Fig. 5.15** Overview of our symmetry-growing approach. **(a)** Input image. **(b)** Local feature extraction. **(c)** Initial matches of symmetric feature pairs. **(d)** Expansion layers. **(e)–(g)** Multi-layer symmetry-growing. **(h)** Segmented symmetric object. For details, see text

ric matching pairs of features [35] as depicted in Fig. 5.15c. Then, starting from singleton symmetry clusters each containing a single symmetry match, we simultaneously expand and merge the symmetry clusters by exploring the image space in our multi-layer growing framework. The algorithm gradually grows reliable symmetry clusters as shown in Fig. 5.15e–f, where the dots with the same color represent features in the same cluster. Finally, the reliable symmetry patterns grown well enough are chosen in Fig. 5.15h, where the detected symmetry is indicated by the convex hull of the features.

We quantitatively evaluated our method on the test dataset of 91 images used in the performance evaluation of symmetry detection [46]. Examples of our results are

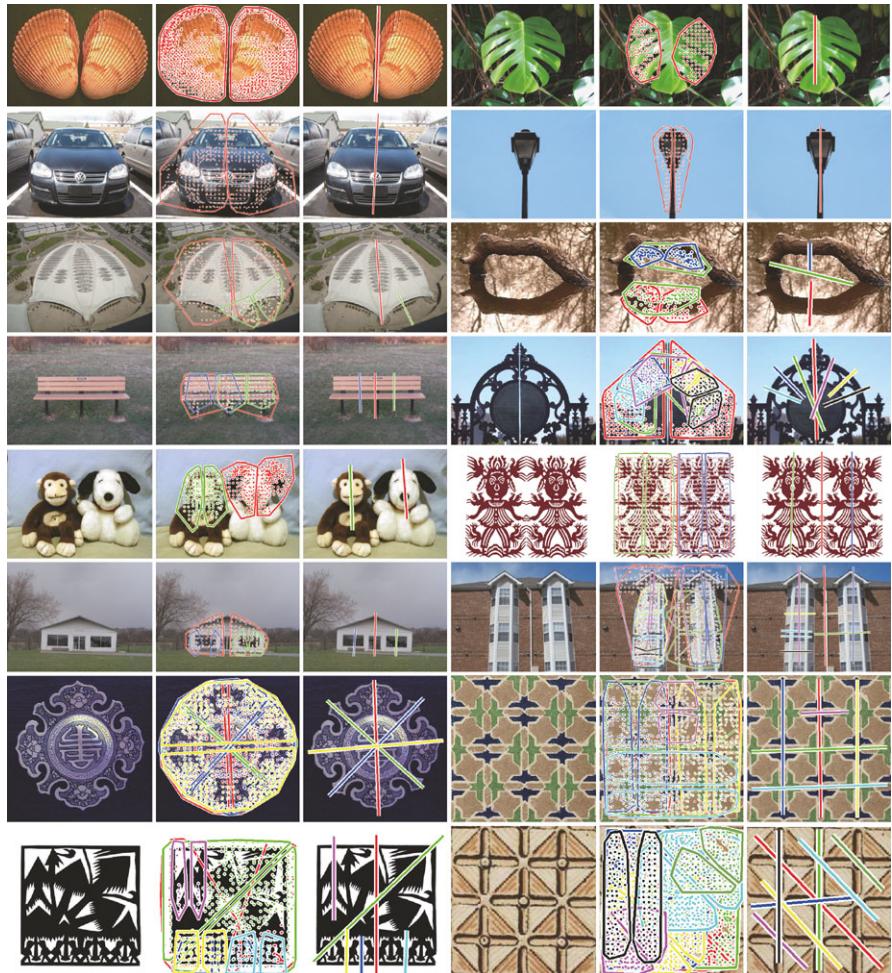


**Fig. 5.16** Symmetry detection comparison on the dataset of [46]. LE06 and LHSX05 denote the methods of [35] and [33], respectively

shown in Figs. 5.16–5.17<sup>2</sup> where the convex-hull segmentation and the major axis of each symmetry cluster are visualized. Each color represents the identity of each symmetry cluster. The examples demonstrate that our symmetry-growing method detects the entire region of symmetric pattern with dense correspondences in each symmetry cluster, so that it provides more accurate and robust performance than the previous methods.

Park et al. [46] compared two state-of-the-art bilateral symmetry detection algorithms [33, 35] in their evaluation. We compared our results with them. For comparison, sensitivity and false positive rate are measured. Suppose  $TP$  is the number of true positives (correctly detected symmetries), and  $FP$  is the number of false positives (wrongly detected symmetries), and  $GT$  is the number of ground truth symmetries. Then, the sensitivity is defined as  $S_0 = TP/GT$  and the false positive rate as  $R_{FP} = FP/GT$ . All the results are summarized in Table 5.2. Note that the results of [35] and [33] are obtained by testing the algorithms on four image scales (from 1 to 1/4 of the original size) and choosing the best result [46], whereas our results are obtained by testing only on the original size of the images. Nevertheless, for sensitivity  $S_0$ , our algorithm clearly outperforms the methods of [33, 35] in all image types. In the sense of false positive rate  $R_{FP}$ , our method appears not clearly better than the two other methods except for the case of real images with multiple symmetry. However, the reason is that our method detects more true symmetric patterns than the ground truth labels of the dataset. That is, a great portion of our false positive detections are not false in fact. For examples, each image at the 3rd and 4th row in Fig. 5.17 has only one symmetric pattern according to the ground truth data, but our method detects even more true bilateral symmetric patterns. It also happens in multiple symmetry dataset as images in the bottom row. These cases were prevalent in our results. These results demonstrate that our method is highly robust to deformation and is effective for real-world complex images.

<sup>2</sup>The dataset, the ground truth, and the result images of [35] and [33] are borrowed from <http://vision.cse.psu.edu/evaluation.html>.



**Fig. 5.17** Examples of symmetry detection and segmentation on the dataset of [46]. See text

**Table 5.2** Performance comparison on the dataset of 91 images from [46]

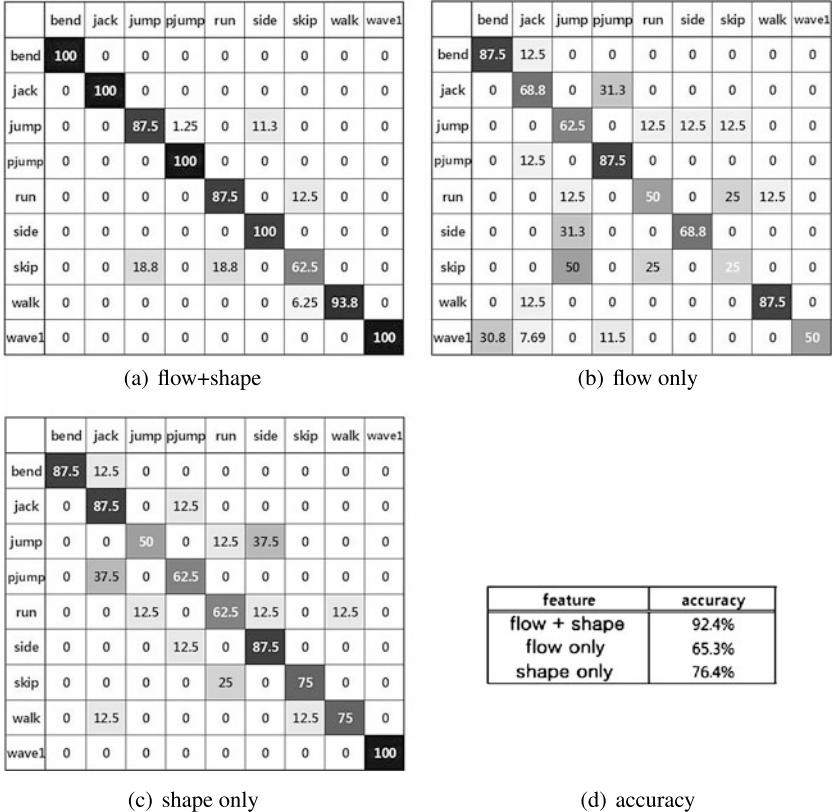
Image type	Synthetic single			Synthetic multiple				
	Algorithm	LE06	LHS05	Ours	Algorithm	LE06	LHS05	Ours
$S_0$		92 %	62 %	100 %		35 %	28 %	77 %
$R_{FP}$		15 %	0 %	15 %		4 %	8 %	33 %
Image type	Real single			Real multiple				
	Algorithm	LE06	LHS05	Ours	Algorithm	LE06	LHS05	Ours
$S_0$		84 %	29 %	94 %		43 %	18 %	68 %
$R_{FP}$		68 %	3 %	69 %		44 %	0 %	17 %

## 5.5 Action Recognition

Action recognition is a challenging problem since there are rich intra-class variations and clutter signals. The intra-class variations come from the diversity of actions such as the actor's appearance, performing style, performing speed, and spatial scale. Thus, we are unlikely to obtain identical observations even if the actions are perceived to belong to the same class. The clutter signals are induced by the other motions of the actors or the visual stimuli from the background part of the video, which make unwanted noise and degrade the performance of the system. Thus, for robust action recognition, we need to deal with these ambiguity of action and clutters. Several approaches have been proposed to tackle the challenges [3, 4, 12, 24, 28, 44, 50]. They can be roughly divided into global approaches [3, 12, 24, 50] and local approaches [4, 28, 44] according to the feature types they rely on. The global methods classify human actions based on global descriptors of human centered windows such as optic flow descriptors [12], correlation of 3D gradients [50], space-time shape of silhouettes [3], and the flow features combined with shape [24]. On the contrary, the local methods extract local features from the video sequences, and they represent the action as a set of spatio-temporal interest points(STIP) [28, 44] or composition of local cuboids [4]. The local methods are found to be generally more robust to the recording conditions, action variance, and occlusion. Although they have achieved significant progress in action recognition, most of them required manually supervised input data or a number of category-specific training data to overcome the aforementioned challenges.

The co-recognition approach naturally extends to the action recognition in the video domain based on the spatio-temporal features [51]. Our intuition is that similar human actions occupy similar spatio-temporal regions with consistent local correspondence topology, and they share similar visual features. The match-growing technique is applied to detect and segment the same actions without manually supervised input videos. In this experiment, there are slight differences in implementation of expansion move. We explicitly designed the *contraction* move to prevent the algorithm from being stuck in bad local optimum. The contraction move selects a target cluster with probability proportional to the cluster size, and a target match in it with probability proportional to the deformation error and photometric error. Half of the expansion move is converted to the reversible contraction move. The overlapping latent cuboids are generated every  $7 \times 7 \times 3$  pixels. The density of the cuboids in the expansion layer determines the localization accuracy. Although the use of denser cuboids will generate smoother result of the detected action, it requires to deal with larger solution space at the cost of more computation time.

To evaluate our algorithm quantitatively, we performed an action classification test on the Weizmann human action dataset [3]. The Weizmann dataset contains 10 categories of actions {*walk, run, jump, gallop sideways, bend, one-hand wave, two-hand wave, jump inplace, jumping jack, skip*}, which are performed by 9 subjects. In our experiment, we used 9 actions excluding two-hand wave, because one-hand wave is a sub-action of two-hand wave. Prior to running the proposed algorithm, some of the video clips are mirror-reflected to make the actors move to the same direction, and some periodic actions are temporally cropped to remove the periodicity.



**Fig. 5.18** Classification confusion matrices on Weizmann dataset. (a)–(c) Different types of features are used to measure the likelihood. (d) The accuracy is calculated by taking average of the diagonal entries of confusion matrix. See text

We conducted a leave-one-out cross validation to measure the performance. One of the video clips of an action category is removed from the database and serves as a reference video to all of the remaining videos one by one. The similarity score between two video clips is evaluated by the number of expanded cuboids and classified using the nearest neighbor procedure.

To test the effectiveness of the features, we also examined the performance of the proposed algorithm on different feature types. We performed the experiments on the same dataset using the flow and the shape features separately. The classification confusion matrices and accuracy table are shown in Fig. 5.18. The result suggests that the shape feature has more discriminative power than the flow feature and the composition of flow and shape features is superior to using only one type of feature. We interpret this superiority of the shape feature to the flow feature in terms of the ambiguity. The flow feature has the explicit movement information of the cuboid, but the erroneous value at the object boundary is the major drawback of the flow feature. On the other hand, the shape feature effectively captures the local

**Table 5.3** Comparison results on Weizmann dataset

	Classification accuracy
<i>Co-recognition</i>	92.4 %
Boiman et al. [4]	97.5 %
Jhuang et al. [21]	93.8 %
Filipovych et al. [15]	88.9 %
Niebles et al. [43]	72.8 %

pose information of the actor. The shape feature is computed from the 2D image at the center of the cuboid, thus it does not contain any explicit movement information. However, since the MCCs are composed of many overlapping cuboids, their configuration indirectly represents the movement of the actor.

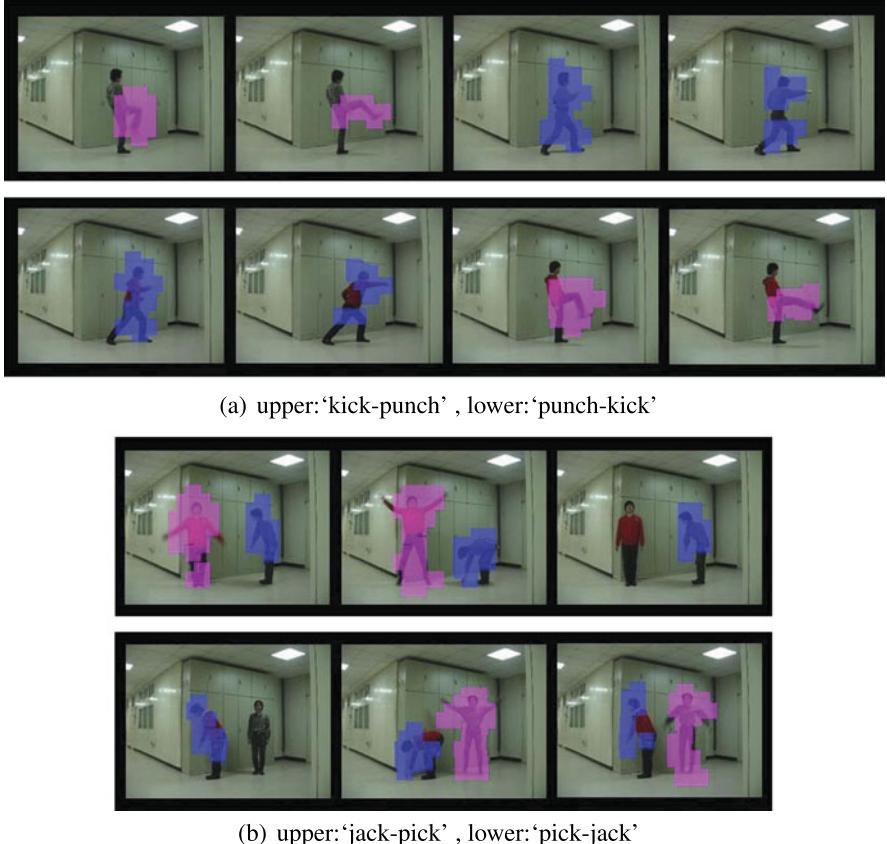
The comparison with other methods on Weizmann dataset is given in Table 5.3. Note that it does not provide the best comparison because each of them has its own limitations and constraints. For example, [4, 15, 43] cannot distinguish multiple actions within a video, and [15, 21, 43] require many training data which are separated by each action class. Our co-recognition shows fair performance as compared with other approaches, considering that only ours has the ability to handle multiple actions without manually split training data.

Since the Weizmann dataset contains a single action per each video, we captured the videos that contain multiple different actions in a clip, and tested our algorithm on them.

Figure 5.19a displays co-recognition results between two actions performed in different temporal order. The actor in the first video performs ‘kick’ and ‘punch’ actions sequentially, while the actor in the second video performs ‘punch’ first and ‘kick’ later. Figure 5.19b shows another scenario. In the videos, the actors perform two actions in different spatial configuration. The video in the top row shows two actors performing ‘jumping jack’ and ‘bend’ simultaneously. The bottom row depicts ‘bend’ and ‘jumping jack’ actions performed in different spatial positions. We ran the co-recognition on the video pairs containing multiple actions. The qualitative results are overlaid on the video frames shown in Fig. 5.19. We display the detected location of spatio-temporal cuboids in color-shaded region of the scene. Each color means the identity of each action. The results indicate that the proposed algorithm successfully recognized the multiple common actions and their correspondences in the video pairs. Our results on the video clips are encouraging since we can obtain the action-level correspondence information between videos with their spatio-temporal location despite the presence of multiple actions. These results can provide very useful information for further analysis of video data.

## 5.6 Summary and Future Work

In this chapter, we have presented the co-recognition approach to images and videos that detects, matches, and segments multiple sets of identical objects or actions in



**Fig. 5.19** The qualitative results of the proposed algorithm on our dataset containing multiple actions in a video clip. **(a)** Selected frames from two videos containing temporally distinguished actions. **(b)** Selected frames from two videos containing spatially distinguished actions

an unsupervised way. The basic idea is to grow initial matches into reliable object correspondences in a multi-layer match-growing framework and analyze their relations by their matching regions or volumes. Unlike other unsupervised segmentation or object discovery methods, it effectively considers both geometry and appearance to discover the detailed dense matches and segmentation from complex images or videos. We have shown its robust performance on a variety of unsupervised vision applications, such as unsupervised object detection and segmentation [8, 9], image retrieval, symmetry analysis [6], action detection [51], and 3D reconstruction.

While the proposed approach provides wide applications and impressive results on complex scenes and videos, the current method still has some limitations. As already noted in Sect. 5.2, co-recognition can detect an object correspondence under the geometric and photometric distinctiveness of the objects appeared in given images or videos. The condition, however, is not strictly satisfied in usual data.

As the consequences, in the presence of similar background or clutter, the detected object regions are over-expanded or non-object regions are falsely detected. To deal with these ambiguity of foreground, the foregroundness [18, 19] or objectness [1] could be further incorporated for better results. In the case of geometrically similar arrangements of objects such as stacked objects or cycling actions, the visual data has intrinsic ambiguity between repetitive patterns of one object and dense arrangements of several objects. This problem requires a statistical inference on more observations of various scenes. Thus, to address this, further analysis on larger networks of object correspondences needs to be investigated to infer the objects, their whole-part relations, as well as object interactions. In the future, pursuing the direction, we plan to improve the co-recognition approach for more complex scene understanding based on mutual relations of various objects.

## References

1. Alexe B, Deselaers T, Ferrari V (2010) What is an object? In: IEEE conference on computer vision and pattern recognition
2. Belongie S, Malik J, Puzicha J (2002) Shape matching and object recognition using shape contexts. *IEEE Trans Pattern Anal Mach Intell* 24(4):509–522
3. Blank M, Gorelick L, Shechtman E, Irani M, Basri R (2005) Actions as space-time shapes. In: IEEE international conference on computer vision
4. Boiman O, Irani M (2006) Similarity by composition. In: Neural information processing system
5. Cho M, Lee KM (2007) Partially occluded object-specific segmentation in view-based recognition. In: IEEE conference on computer vision and pattern recognition
6. Cho M, Lee KM (2009) Bilateral symmetry detection and segmentation via symmetry-growing. In: British machine vision conference
7. Cho M, Lee KM (2009) Feature correspondence and deformable object matching via agglomerative correspondence clustering. In: IEEE international conference on computer vision
8. Cho M, Shin YM, Lee KM (2008) Co-recognition of image pairs by data-driven Monte Carlo image exploration. In: European conference on computer vision
9. Cho M, Shin YM, Lee KM (2010) Unsupervised detection and segmentation of identical objects. In: IEEE conference on computer vision and pattern recognition
10. Cho M, Shin YM, Lee KM (2011) Object correspondence networks for unsupervised recognition of identical objects. In: Emerging topics in computer vision and its applications, vol 1, p 313
11. Cornelius H, Perd'och M, Matas J, Loy G (2007) Efficient symmetry detection using local affine frames. In: SCIA, pp 152–161
12. Efros AA, Berg AC, Mori G, Malik J (2003) Recognizing action at a distance. In: IEEE international conference on computer vision
13. Faugeras O (1993) Three-dimensional computer vision: a geometric viewpoint. MIT Press, Cambridge
14. Ferrari V, Tuytelaars T, Gool L (2006) Simultaneous object recognition and segmentation from single or multiple model views. *Int J Comput Vis* 67(2):159–188
15. Filipovych R, Ribeiro E (2008) Learning human motion models from unsegmented videos. In: IEEE conference on computer vision and pattern recognition
16. Furukawa Y, Ponce J (2007) Accurate, dense, and robust multi-view stereopsis. In: IEEE conference on computer vision and pattern recognition
17. Hartley R, Zisserman A (2004) Multiple view geometry in computer vision. Cambridge University Press, Cambridge

18. Hou X, Zhang L (2007) Saliency detection: a spectral residual approach. In: IEEE conference on computer vision and pattern recognition
19. Itti L, Koch C, Niebur E (1998) A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans Pattern Anal Mach Intell* 20:1254–1259
20. Jain AK, Dubes RC (1998) Algorithms for clustering data. Prentice Hall, New York
21. Jhuang H, Serre T, Wolf L, Poggio T (2007) A biologically inspired system for action recognition. In: IEEE international conference on computer vision
22. Kannala J, Rahtu E, Brandt S, Heikkila J (2008) Object recognition and segmentation by non-rigid quasi-dense matching. In: IEEE conference on computer vision and pattern recognition
23. Karlinsky L, Dinerstein M, Levi D, Ullman S (2008) Unsupervised classification and part localization by consistency amplification. In: European conference on computer vision
24. Ke Y, Sukthankar R, Hebert M (2007) Event detection in crowded videos. In: IEEE international conference on computer vision
25. Keller Y, Shkolnisky Y (2004) An algebraic approach to symmetry detection. In: IEEE international conference on pattern recognition
26. Kim TH, Lee KM, Lee SU (2010) Nonparametric higher-order learning for interactive segmentation. In: IEEE conference on computer vision and pattern recognition
27. Laptev I (2005) On space-time interest points. *Int J Comput Vis* 64(2/3):107–123
28. Laptev I, Lindeberg T (2003) Space-time interest points. In: IEEE international conference on computer vision
29. Laptev I, Marszałek M, Schmid C, Rozenfeld B (2008) Learning realistic human actions from movies. In: IEEE conference on computer vision and pattern recognition
30. Lempitsky V, Kohli P, Rother C, Sharp T (2009) Image segmentation with a bounding box prior. In: IEEE international conference on computer vision
31. Lhuillier M, Quan L (2002) Match propagation for image-based modeling and rendering. *IEEE Trans Pattern Anal Mach Intell* 24(8):1140–1146
32. Li Z, Liu J, Chen S, Tang X (2007) Noise robust spectral clustering. In: IEEE international conference on computer vision
33. Liu Y, Hays JH, Xu YQ, Shum HY (2005) Digital papercutting. Technical sketch, SIGGRAPH
34. Lowe DG (1999) Object recognition from local scale-invariant features. In: IEEE international conference on computer vision
35. Loy G, Eklundh JO (2006) Detecting symmetry and symmetric constellations of features. In: European conference on computer vision, pp II-508–II-521
36. Lucas BD, Kanade T (1981) An iterative image registration technique with an application to stereo vision. In: DARPA image understanding workshop
37. Marola G (1989) On the detection of the axes of symmetry of symmetric and almost symmetric planar images. *IEEE Trans Pattern Anal Mach Intell* 11:104–108
38. 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
39. Matas J, Chum O, Urban M, Pajdla T (2002) Robust wide baseline stereo from maximally stable extremal regions. In: British machine vision conference
40. Mikolajczyk K, Schmid C (2002) An affine invariant interest point detector. In: European conference on computer vision
41. Mikolajczyk K, Schmid C (2004) Scale and affine invariant interest point detectors. *Int J Comput Vis* 60(1):63–86
42. Mikolajczyk K, Schmid C (2005) A performance evaluation of local descriptors. *IEEE Trans Pattern Anal Mach Intell* 27(10):1615–1630
43. Niebles JC, Fei-Fei L (2007) A hierarchical model of shape and appearance for human action classification. In: IEEE conference on computer vision and pattern recognition
44. Niebles JC, Wang H, Fei-Fei L (2006) Unsupervised learning of human action categories using spatial-temporal words. In: British machine vision conference
45. Obdržálek S, Matas J (2002) Object recognition using local affine frames on distinguished regions. In: British machine vision conference

46. Park M, Lee S, Chen PC, Kashyap S, Butt AA, Liu Y (2008) Performance evaluation of state-of-the-art discrete symmetry detection algorithms. In: IEEE conference on computer vision and pattern recognition
47. Preparata F, Shamos M (1985) Computational geometry. Springer, Berlin
48. Rother C, Kolmogorov V, Blake A (2004) Grabcut—interactive foreground extraction using iterated graph cuts. In: ACM SIGGRAPH
49. Rother C, Minka TP, Blake A, Kolmogorov V (2006) Cosegmentation of image pairs by histogram matching—incorporating a global constraint into MRFs. In: IEEE conference on computer vision and pattern recognition, pp 993–1000
50. Shechtman E, Irani M (2005) Space-time behavior based correlation. In: IEEE conference on computer vision and pattern recognition
51. Shin YM, Cho M, Lee KM (2010) Co-recognition of actions in video pairs. In: International conference on pattern recognition
52. Simon I, Seitz SM (2007) A probabilistic model for object recognition, segmentation, and non-rigid correspondence. In: IEEE conference on computer vision and pattern recognition
53. Sivic J, Russell BC, Efros AA, Zisserman A, Freeman WT (2005) Discovering object categories in image collections. In: IEEE international conference on computer vision
54. Steele KL, Egbert PK (2005) Correspondence expansion for wide baseline stereo. In: IEEE conference on computer vision and pattern recognition
55. Todorovic S, Ahuja N (2007) Unsupervised category modeling, recognition, and segmentation in images. *IEEE Trans Pattern Anal Mach Intell* 30(12):2158–2174
56. Toshev A, Shi J, Daniilidis K (2007) Image matching via saliency region correspondences. In: IEEE conference on computer vision and pattern recognition
57. Tuytelaars T, Mikolajczyk K (2008) Local invariant feature detectors: a survey. *Found Trends Comput Graph Vis* 3(3):177–280
58. Vedaldi A, Soatto S (2006) Local features, all grown up. In: IEEE conference on computer vision and pattern recognition
59. Yuan J, Wu Y (2007) Spatial random partition for common visual pattern discovery. In: IEEE international conference on computer vision, pp 1–8

# Chapter 6

## Stereo Matching—State-of-the-Art and Research Challenges

Michael Bleyer and Christian Breiteneder

**Abstract** Stereo matching denotes the problem of finding dense correspondences in pairs of images in order to perform 3D reconstruction. In this chapter, we provide a review of stereo methods with a focus on recent developments and our own work. We start with a discussion of local methods and introduce our algorithms: geodesic stereo, cost filtering and PatchMatch stereo. Although local algorithms have recently become very popular, they are not capable of handling large untextured regions where a global smoothness prior is required. In the discussion of such global methods, we briefly describe standard optimization techniques. However, the real problem is not in the optimization, but in finding an energy function that represents a good model of the stereo problem. In this context, we investigate data and smoothness terms of standard energies to find the best-suited implementations of which. We then describe our own work on finding a good model. This includes our combined stereo and matting approach, Surface Stereo, Object Stereo as well as a new method that incorporates physics-based reasoning in stereo matching.

### 6.1 The Stereo Matching Problem

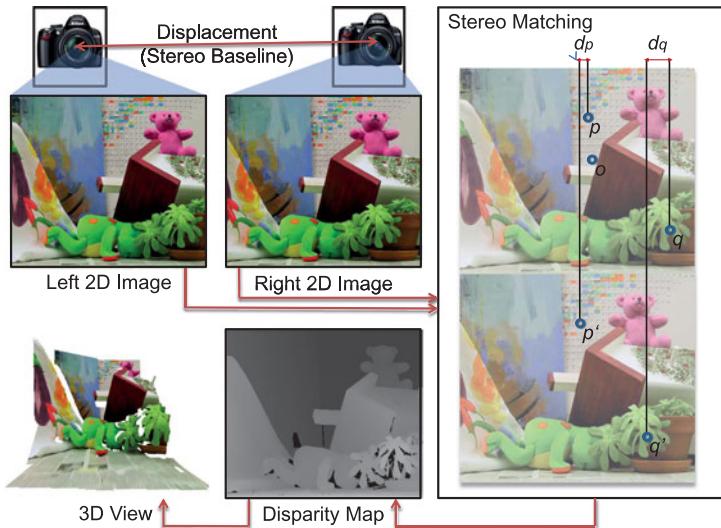
This chapter concentrates on the stereo matching problem, which is one of the oldest, but still yet unsolved problems in computer vision. Solving this matching problem is the central step in a shape from stereo method. Figure 6.1 outlines the pipeline of this approach. In analogy to human depth perception that uses two eyes, there are two cameras. These cameras are slightly displaced such that they see the same scene

---

M. Bleyer (✉) · C. Breiteneder  
Vienna University of Technology, Vienna, Austria  
e-mail: [bleyer@ims.tuwien.ac.at](mailto:bleyer@ims.tuwien.ac.at)

C. Breiteneder  
e-mail: [breiteneder@ims.tuwien.ac.at](mailto:breiteneder@ims.tuwien.ac.at)

M. Bleyer  
Microsoft Redmond, Redmond, USA



**Fig. 6.1** Depth reconstruction via stereo. Two slightly displaced cameras record the scene. The stereo matching module finds dense correspondences between left and right images and encodes them in a disparity map. This disparity map is sufficient to reconstruct the 3D coordinates of each pixel

from different viewpoints.<sup>1</sup> Note that throughout this chapter we will assume that the stereo pair has been rectified such that corresponding points lie on the same horizontal scanlines in left and right views.<sup>2</sup>

Left and right images then form the input for the stereo matching module. In stereo matching, the task is to find a corresponding pixel in the right image for each pixel of the left image. Let us look at Fig. 6.1 where we have placed the left image on top of the right one to understand why this allows depth reasoning. We have marked the background pixel  $p$  of the left image as well as its corresponding pixel  $p'$  in the right view. Note that  $p$  and  $p'$  are displaced in horizontal direction due to the different perspectives under which left and right images have been recorded. The amount of this displacement (in pixels) is called disparity ( $d_p$  in the figure). We have also marked a pixel  $q$  and its correspondence  $q'$  of a foreground object. The important observation is that the disparity of the foreground pixel is larger than that of the background pixel (compare  $d_p$  and  $d_q$  in Fig. 6.1) and it is easy to show (e.g., in [29]) that disparity is inversely proportional to the distance of a pixel to the camera. Note that also the human brain perceives depth using disparity information.

In computational stereo, disparity information is typically stored in an intensity image (a so-called disparity map) where dark pixels encode low disparity val-

<sup>1</sup>The distance between the cameras is thereby referred to as the stereo baseline.

<sup>2</sup>Rectification can be accomplished using standard methods (e.g., [29]), once the stereo camera system has been calibrated.

ues (high distances from the camera) and bright intensities encode large disparities (close distances to the camera). See Fig. 6.1 for an example. In a calibrated stereo system, the disparity map is sufficient to reconstruct a metric 3D model of the recorded scene, which is the final goal in a shape from stereo approach.

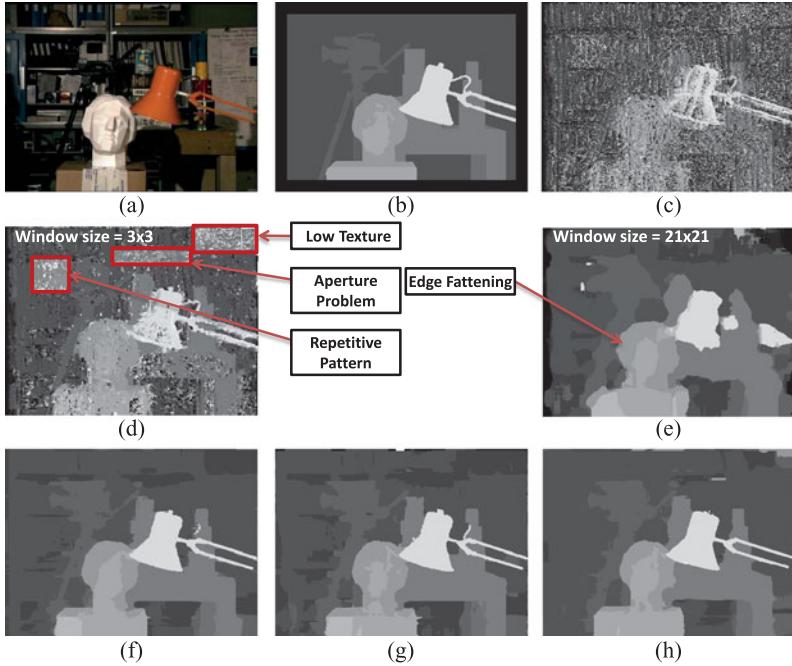
The challenge in shape from stereo is to solve the stereo correspondence problem. This problem is permanently and unconsciously solved in the human brain, but turns out to be very challenging for a computer. This is for various reasons. First, as common in computer vision, images are usually contaminated by sensor noise. Second, in the absence of texture, stereo matching becomes highly ambiguous. Note that also a human is not able to perceive the correct depth if, for example, standing in front of a completely white wall without any texture. Third, not every pixel of one image has a correspondence in the other image, because due to the different image perspectives, a pixel can be occluded. To illustrate this, we have also marked a pixel  $o$  in Fig. 6.1. Note that this pixel cannot be found in the right view, that is, it is occluded.

Being able to solve the stereo matching problem is important in two respects. First, it may help to get a better understanding of how human depth perception works. Second, there are various applications in computer vision. For example, 3D reconstruction of cities from internet photography has recently gained popularity [1, 70]. Depth maps can also be fused to automatically generate high-quality 3D models of persons or even whole rooms as demonstrated by KinectFusion [56]. Stereo reconstructions can be applied for robot navigation (e.g., autonomously driving car), but also in human motion capture where Microsoft Kinect has recently demonstrated that depth information is vital [68]. There is also potential in next generation television where depth maps enable novel view synthesis, which allows the user to interactively control its viewing perspective [90]. Other applications include 3D tracking (surveillance, pose estimation, augmented reality, human-computer interaction), depth segmentation (z-keying) and industrial applications (quality assurance), to name just a few of them. Basically, whenever one needs to infer geometric information from the surrounding world, stereo vision represents a low-cost and non-intrusive alternative to active devices such as range finders.

In the following, we give a review of stereo methods with a focus on recent developments. This review provides the context for our own algorithms. We follow Scharstein and Szeliski [67] by dividing stereo algorithms in local and global ones.

## 6.2 Local Methods

**A Naive Approach** Let us start by describing the simplest possible stereo algorithm. It is reasonable to assume that corresponding pixels in left and right images have similar colors, which is known as the photo consistency assumption. Furthermore, we know that corresponding pixels lie on the same horizontal scanline as we assume that our images have been rectified. Hence a simple algorithm works as follows. For each pixel  $p$  of the left image, we search along the corresponding scanline



**Fig. 6.2** Local methods. (a) Left image of the Middlebury Tsukuba image. (b) Ground truth disparity map. (c) Result of a naive algorithm. (d) Window-based aggregation with a  $3 \times 3$  window. (e) Aggregation with a  $21 \times 21$  window. (f) Result of the adaptive support weight approach [86]. (g) Our geodesic approach [36]. (h) Our cost filter method [61]

in the right image. We then select the pixel  $p'$  whose color is most similar to that of  $p$  as a matching point.

Figure 6.2c shows the disparity map of this naive approach which is very noisy. The problem is the high ambiguity of the data, i.e., if we want to find the correspondence of a red pixel of the left image (e.g., on the Tsukuba lamp of Fig. 6.2a) there is usually a relatively large candidate set of red pixels in the right image. The common approach taken in computer vision (and in all stereo algorithms) is to regularize the problem by imposing a smoothness assumption in order to cope with this ambiguity. This smoothness assumption means that spatial neighboring pixels are likely to have similar disparities. Stereo algorithms differ in the way how this assumption is implemented, which defines the difference between local and global methods [67].

**The Principle of Local Algorithms** Let us now come back to the naive approach above. Instead of matching single pixels, we can match small image areas.<sup>3</sup> It is important to note that by using areas we have made use of the smoothness assumption

<sup>3</sup>This is the reason why local algorithms are also sometimes referred to as area-based methods in literature.

in an implicit way, that is, we assume that all pixels of the area have exactly the same disparity. Let us now formulate the corresponding algorithm. For each pixel  $p$  of the left image, we compute its disparity  $d_p$  as

$$d_p = \operatorname{argmin}_{0 \leq d \leq d_{\max}} \sum_{q \in W_p} c(q, q - d). \quad (6.1)$$

Here,  $d_{\max}$  is a parameter defining the maximum allowed disparity.  $W_p$  denotes a square window centered on  $p$ . The function  $c(p, q)$  computes the color dissimilarities between a pixel  $p$  of the left and a pixel  $q$  of the right image (e.g., summed-up absolute differences in RGB values). We write  $q - d$  to denote the pixel coordinate that is derived by subtracting  $d$  from  $q$ 's x-coordinate.

**Selecting an Appropriate Window Size** The local algorithm above has one important parameter, i.e., the size of the window  $W$ . From a computational point of view, the algorithm's runtime complexity is  $\mathcal{O}(N \cdot d_{\max} \cdot |W|)$  where  $N$  is the number of pixels in the image and  $|W|$  denotes the number of pixels in the window. Hence, larger windows would lead to higher run times if the algorithm was implemented in a naive way. However, in an efficient implementation, one can take advantage of high redundancy in the computation to reduce runtime complexity to  $\mathcal{O}(N \cdot d_{\max})$  so that run time no longer depends on the size of the support window. The trick is to use a so-called sliding window technique [21, 55]. However, the size of the support window has a large effect on the quality of the disparity map as discussed next.

The problem of small support windows is that they may not capture enough texture variation to resolve matching ambiguities. In particular, the algorithm fails in untextured regions, areas with only horizontal texture (aperture problem) and repetitive image regions. This is illustrated in Fig. 6.2d where we have used a  $3 \times 3$  window to compute the disparity map. However, note that disparity discontinuities are well preserved as a consequence of the small window size.

A remedy to overcome the ambiguity problem is the use of large support windows. A matching result using a large  $21 \times 21$  window is shown in Fig. 6.2e. While this disparity map is considerably smoother than that of Fig. 6.2d, it is evident that object borders are badly preserved. The problem is our implicit smoothness assumption, that is, pixels within the window are supposed to have constant disparity. In the proximity of depth discontinuities, this assumption is broken as the window captures a mixture of foreground and background disparities. Whether the foreground or the background disparity leads to lower color dissimilarity depends on the texturedness of objects. In many cases the foreground's texture is dominant, which leads to the well-known foreground flattening problem (see Fig. 6.2e).

The traditional problem of local algorithms is that there is no ideal setting for the parameter defining the window size such that the algorithm gives correct results in low textured areas and regions close to object borders at the same time. Almost all work on local stereo matching focuses on the use of adaptive windows.

**Adaptive Windows** The idea of adaptive window algorithms is to select an individual window at each pixel such that the support region of a pixel remains large

(in order to capture enough intensity variation), but does not overlap a disparity discontinuity (in order to avoid the edge fattening problem). For example, Fusiello et al. [24] test nine different square windows of constant size per pixel. These windows differ from each other in that they are centered at different positions, and the hope is that at least on one of these positions the window does not overlap a depth discontinuity. Hirschmüller et al. [34] divide the search window into a set of nine sub-windows. Only five of these sub-windows are used to compute the aggregated matching costs, that is, if the other four subwindows capture a different disparity than the center pixel they do not have any influence. As a final example, Veksler [78] estimates an arbitrarily sized and shaped window per pixel by optimizing over a large class of compact windows. In practice, none of the above methods has been able to compete with the quality of global methods (described in Sect. 6.3). Hence, the local stereo approach has been believed to be obsolete for some time.

**Adaptive Support Weights** In recent years, local stereo has experienced a renaissance due to the introduction of adaptive support weights [86].<sup>4</sup> The key idea is to assign an individual weight to each pixel that determines the pixel's influence in the matching process. Let us reformulate (6.1) accordingly as

$$d_p = \operatorname{argmin}_{0 \leq d \leq d_{\max}} \sum_{q \in W_p} w(p, q) \cdot c(q, q - d). \quad (6.2)$$

We have now introduced a weight function  $w(p, q)$  which should ideally return a value of 1 if pixel  $q$  lies on the same disparity as the center pixel  $p$  and 0 otherwise. Since disparities are not known in advance, defining  $w(p, q)$  is challenging, that is, leads to a chicken-and-egg problem. Adaptive support weight algorithms differ in the way how they define  $w(p, q)$  and we discuss this below. To our knowledge, all of them use the color cue for computing this function.

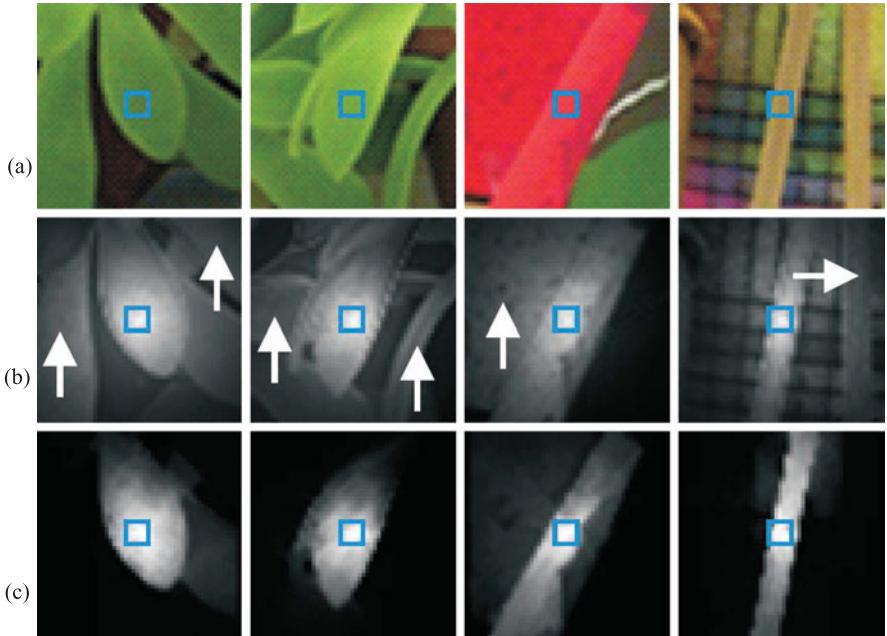
**The Weight Function** The original adaptive support weight paper [86] makes the assumption that spatially close pixels that are similar in color are likely to originate from the same scene object. Hence, they are also likely to share the same disparity. We define the corresponding function  $w_{BL}(p, q)$  as

$$w_{BL}(p, q) = \exp\left(-\left(\frac{\text{color}(p, q)}{\gamma_c} + \frac{\text{spatial}(p, q)}{\gamma_s}\right)\right). \quad (6.3)$$

Here,  $\text{color}(p, q)$  computes the color dissimilarity of  $p$  and  $q$  as the Euclidean distance between  $p$ 's and  $q$ 's color values in RGB space. The function  $\text{spatial}(p, q)$  computes the Euclidean distance between  $p$ 's and  $q$ 's image coordinates.  $\gamma_c$  and  $\gamma_s$  are user-defined parameters. Note that a pixel  $q$  obtains high weight only if it is similar to the center pixel  $p$  in terms of color and spatial position (see assumption

---

<sup>4</sup>It is interesting to note that the majority of recent submissions to the Middlebury benchmark [67] are adaptive support weight techniques.



**Fig. 6.3** Our geodesic approach [36]. We show adaptive support windows for the Middlebury Teddy and Cones images. (a) Color patch. The window’s center pixel is marked by a *blue rectangle*. (b) Support weights computed by [86]. *Bright values* mean high weight. As marked by the *white arrows*, high weights are assigned to pixels that lie on a different disparity than the center pixel (e.g., the background leaves in the two left-most images). (c) Our geodesic support weights [36]. We avoid these wrong high weights by enforcing connectivity

above). Also note that the function above is equivalent to the weighting function of the bilateral filter.

Yoon and Kweon’s work [86] has been a breakthrough in local stereo matching as (at least in the Middlebury benchmark) results on-par with global methods could be achieved for the first time. To illustrate the good quality of disparity maps, we plot the Tsukuba result of [86] in Fig. 6.2f. The downside of this approach is that these good-quality results come at the price of considerably increased run times. Due to the weighting function the sliding window technique (mentioned above) can no longer work and the algorithm’s run time depends on the size of the support window. This is particularly bad as adaptive support weight techniques typically operate on large windows (e.g.,  $33 \times 33$  pixels in [86]). We will discuss later on one of our techniques [61] that has a considerably better runtime property, i.e., runs in real time, and even outperforms [86] in terms of quality of results.

**Our Contribution—Geodesic Stereo [36]** Our paper [36] proposes a new weight function. The idea is to introduce a connectivity property, i.e., two pixels are likely to lie on the same object (and disparity, respectively) if they are connected by a path of approximately the same color. Let us look at Fig. 6.3 to explain the ad-

vantage of this connectivity property. Figure 6.3a shows four image crops taken from the left images of the Middlebury Teddy and Cones pairs. Figure 6.3b shows that the weights computed by the method of [86] are not ideal. By only looking at color and spatial differences, the method assigns high weights to pixels that lie on a different disparity than the center pixel, which happens, for example, at the background leaves in the two left-most images of Fig. 6.3b. In contrast, our method [36] (see Fig. 6.3c) avoids these wrong high weights. In the weight computation at each pixel  $q$ , we check if  $q$  is connected to the center pixel  $p$  by a path of constant color. This is not the case in the leaves example as the foreground and background leaves are separated by a color edge. More formally, our weighting function  $w_{\text{GEO}}(p, q)$  is inversely proportional to the geodesic distance:

$$w_{\text{GEO}}(p, q) = \exp\left(-\frac{\text{geo}(p, q)}{\gamma}\right) \quad (6.4)$$

where the parameter  $\gamma$  controls the strength of the segmentation.  $\text{geo}(p, q)$  is defined as

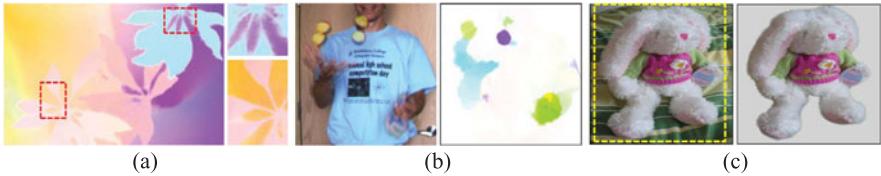
$$\text{geo}(p, q) = \min_{P \in \mathcal{P}_{p,q}} d(P). \quad (6.5)$$

Here  $\mathcal{P}_{p,q}$  denotes all possible paths  $\langle p_1, p_2, \dots, p_n \rangle$  that connect  $p$  and  $q$  within the support window, that is,  $p_1 = p$  and  $p_n = q$ . The costs of a path  $d(P)$  are computed as

$$d(P) = \sum_{i=2}^{i=n} \text{color}(p_i, p_{i-1}). \quad (6.6)$$

In our experiments on the Middlebury data, we demonstrate that our geodesic approach outperforms the original adaptive support weight method [86]. At the time of publication (2009), our method has been the top-performer among all local methods in the Middlebury benchmark. The disparity map generated for the Tsukuba images is shown in Fig. 6.2g. In a follow-up work [37], we have shown how to speed up our geodesic approach such that near real time frame rates can be achieved without considerable loss of quality.

**Adaptive Support Weight Stereo via Image Filtering** As stated above, the weight function of the original adaptive support weight function [86] corresponds to the filter weights of a bilateral filter. It is known that the aggregation step of Yoon and Kweon's method [86] can be understood as filtering the cost volume with a joint bilateral filter (see our cost filter paper [61] and [62]). This insight forms the basis for speeding up the adaptive support weight approach. To be more precise, the cost volume is a three-dimensional array that is derived by computing the costs for matching each pixel  $(x, y)$  at each allowed disparity  $d$ . The joint bilateral filter is then applied on each individual  $xy$ -slice of this volume where filter weights are computed from the left color image. Finally, a disparity map is obtained by selecting the disparity of minimum costs in the filtered volume at each pixel.



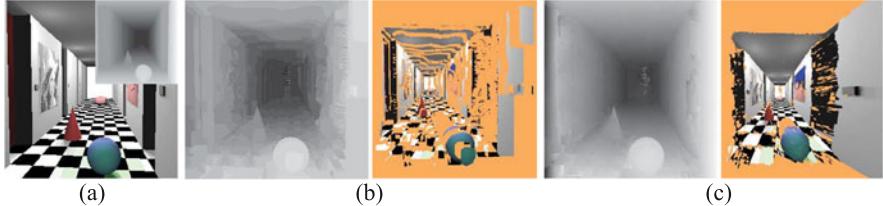
**Fig. 6.4** Our cost filter approach [61] applied on problems outside of stereo. **(a)** Optical flow. We use the *color coding* of [3] for visualization of the flow field. Our approach accurately reconstructs flow borders and thin structures. **(b)** Large displacement optical flow. In contrast to many competing algorithms, our method handles large motions. **(c)** Interactive segmentation. User input is given by drawing the *yellow rectangle* (see *left image*). Our method then computes a binary segmentation using cost filtering (see *right image*)

Being able to implement the original adaptive support weight approach with runtime independent of the window size boils down to the question whether it is possible to implement joint bilateral filtering with runtime complexity independent of the filter kernel size. According to the current state of research, a so-called  $O(1)$  implementation only works for approximations of the joint bilateral filter. Several authors [42, 62, 89] have used such approximations to derive fast implementations of Yoon and Kweon’s algorithm [86]. In [42] and [89], the joint bilateral filter is approximated by using integral histograms as described in [60], while [62] uses an approximation based on the bilateral grid of [59]. The problem of these approximative approaches is that they sacrifice quality of disparity maps for speed.

A different strategy to derive an  $O(1)$  implementation of adaptive support weight matching is to replace the joint bilateral filter with a different filter that shares the joint bilateral filter’s edge-preserving property, but can innately be implemented with runtime independent of the filter kernel size. In this line of research, [53, 88] use a cross-shaped filter. Due to using a cross-shaped support region, the algorithm fails at fine structures that are neither horizontal nor vertical. A better alternative is discussed next.

**Our Contribution—Cost Filter [61]** In [61], we propose to use the very recent guided filter [31] for cost filtering. This filter is similar to the bilateral one in that it shares its edge-preserving property. However, the advantage is that an exact (non-approximative) implementation can be accomplished by running a series of box filters so that the filter’s runtime does not depend on the filter kernel size. We show in [39] that a GPU-based implementation runs at 33.3 frames per second for  $640 \times 480$  pixel images and 40 allowed disparity levels. This makes it the fastest available adaptive support weight algorithm. Apart from that, our method has also been the best-performing method among all local algorithms in the Middlebury table [67] at the time of publication in 2011. In particular, this also means that it outperforms Yoon and Kweon’s algorithm [86]. An example result is shown in Fig. 6.2h. In [38], we show how to extend the cost filter approach in order to compute temporally consistent disparity maps given a stereo video as an input.

An important contribution of our paper [61] is to show that the concept of cost filtering is not restricted to stereo, but can be applied to other computer vision tasks



**Fig. 6.5** Our PatchMatch Stereo algorithm [14]. **(a)** Left image and ground truth disparities of the Corridor pair that contains highly slanted surfaces. **(b)** The disparity map computed with fronto-parallel windows approximates the slanted surfaces via many fronto-parallel ones. **(c)** PatchMatch Stereo correctly reconstructs the scene as a collection of slanted planar surfaces via using slanted support windows and continuous sub-pixel disparities

that can be formulated as labeling problems. We first instantiate our framework for optical flow computation and show that our algorithm can reach a top position in the Middlebury optical flow benchmark [3] using almost identical parameter settings as for the stereo problem. Figure 6.4a shows a flow map for the Middlebury Schefflera test set. As seen from Fig. 6.4b, an advantage of our optical flow method is that it can handle large displacements, which is challenging for most competing methods.

We then instantiate our framework for interactive image segmentation, which is a very different problem from stereo or optical flow computation. Note that in this case, the cost volume does not store color dissimilarities, but the likelihood to which a pixel belongs to a foreground and a background color model, respectively. The other steps, that is, filtering of the cost volume and minimum selection, remain the same. We show that our segmentation method can compete with a state-of-the-art method, that is, GrabCut [63], but runs considerably faster. It takes 5 milliseconds to segment a  $1000 \times 1000$  pixel image. An example result is shown in Fig. 6.4c.

**Slanted Surfaces and Sub-pixel Precision** Let us now come back to the implicit smoothness assumption of local algorithms, that is, all pixels within the support window have the same disparity. We have already discussed that this assumption is broken at disparity borders and that adaptive support weight algorithms represent a good remedy to this problem. However, there are two additional problems. (1) Disparity values of pixels within the support window will be different if the window captures a slanted (non-fronto-parallel) surface. (2) So far we have only spoken about integer-valued disparities and ignored that we should ideally match at continuous sub-pixel disparity values.

Figure 6.5b shows the effect of these two problems on the Corridor test set that contains highly slanted surfaces. To derive the disparity map, we have used fronto-parallel windows that are matched at integer-valued disparities. Note that this is exactly what all algorithms described above do. As can be seen from Fig. 6.5b, the 3D model derived from the computed disparity map is relatively poor, as the slanted planes are reconstructed via many fronto-parallel surfaces. It is known that this problem can be overcome by using slanted planar support windows matched at continuous sub-pixel disparities. However, this model leads to a difficult optimization problem as it is not known in advance which slanted windows occur in

the scene and the number of candidate planes is infinite. After discussing previous methods, we will present our algorithm [14] that can effectively handle this complex optimization problem. The result of our algorithm on the Corridor scene is shown in Fig. 6.5c.

**Previous Sub-pixel/Slanted Window Algorithms** Some local methods (e.g., [84]) obtain sub-pixel precision in a post-processing step by fitting a parabola in the cost volume. From our experience, this method leads to relatively noisy sub-pixel information. A better option is to account for sub-pixel displacements directly in the matching process. This can be accomplished by extending the label space, that is, some fractional disparity values (half- or quarter-pixel) are considered in addition to the integer-valued ones (e.g., in [26]). Note that this is still a discrete approach and that run time doubles (quadruples) if half-pixel (quarter-pixel) precision is used. The same principle works for slanted windows, that is, in addition to fronto-parallel windows a set of slanted windows can be included in the label space (e.g., in [25]). This is known as plane sweeping in the literature. As before, this is a discrete approach, that is, the chances for not having the correct plane in the label space are high, and this strategy leads to considerably longer run times.

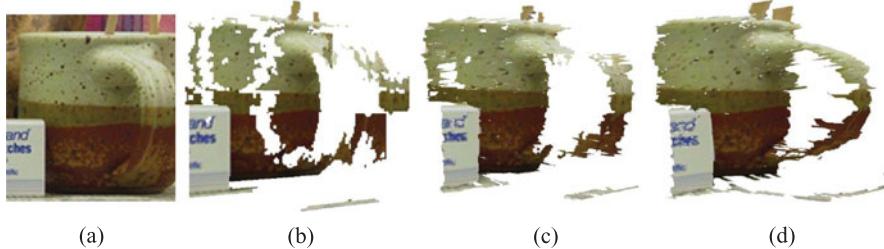
Finally, we also want to mention the approach of [87]. The authors first compute an initial disparity map and then apply plane fitting at each pixel. The extracted slanted planes are then used in a second matching round. The problem is that this method fails for highly slanted surfaces as the disparity results of the first round are too poor to extract the correct planes.

**Our Contribution—PatchMatch Stereo [14]** Let us reformulate (6.2) such that optimization is now performed over the set of all possible 3D planes, that is,

$$d_p = \operatorname{argmin}_{f \in \mathcal{F}} \sum_{q \in W_p} w(p, q) \cdot c(q, q - (a_f q_x + b_f q_y + c_f)). \quad (6.7)$$

Here  $\mathcal{F}$  is the set of all disparity planes and  $a_f$ ,  $b_f$  and  $c_f$  are the three parameters of a plane  $f$ . We write  $q_x$  and  $q_y$  to denote pixel  $q$ 's x- and y-coordinates. Note that we also use an adaptive weight function  $w(p, q)$  to handle the edge fattening problem. For simplicity, we use the one of [86] (see (6.3)). As stated above, minimizing (6.7) is difficult as the set  $\mathcal{F}$  has infinite cardinality. Hence, the approach of checking all possible labels (implemented by all algorithms above) can no longer work. We propose an algorithm based on PatchMatch [4] to approximate the minimum of (6.7) at each pixel.

The basic observation is that relatively large regions of an image can be modeled by approximately the same plane. In the initialization step of the algorithm, each pixel is assigned to a plane with random parameters. It is relatively obvious that most of these random planes will be wrong, but the hope is that at least one pixel of a region carries a plane that is close to the optimal one. Note that this is very likely, since we have many guesses. For example, in the Corridor scene of Fig. 6.5a the ground plane consists of approximately 25000 pixels, that is, we have 25000

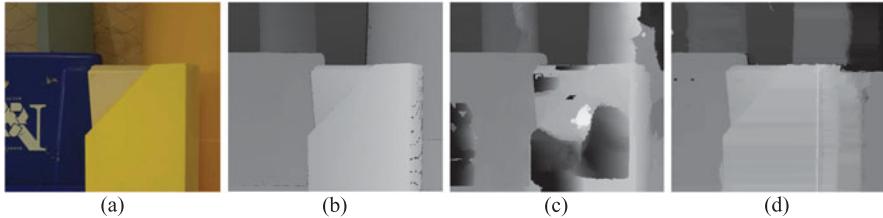


**Fig. 6.6** 3D reconstructions of our algorithm [14] and its competitors. **(a)** Crop of the Middlebury Cones image. **(b)** Matching using fronto-parallel windows and integer disparities. **(c)** Matching using fronto-parallel windows and continuous disparities. **(d)** Our PatchMatch stereo algorithm that uses slanted windows and continuous disparities. Note that the *rounded shape* of the cup’s handle is well reconstructed

guesses to find the correct plane for this object. If there is at least a single correct guess, then this is already sufficient, since our algorithm propagates this plane to neighboring pixels. We implement three different forms of propagation, that is, spatial propagation, view propagation and temporal propagation (see our paper [14] for details).

Our PatchMatch stereo algorithm is currently the top performer among all local algorithms when sub-pixel precision is considered (Middlebury error threshold 0.5). It is even the top performer among all algorithms for the challenging Teddy set (Middlebury default error threshold 1.0) that has a highly slanted ground plane where competing methods run into problems. Figure 6.6 demonstrates the high amount of disparity detail that our method achieves. In contrast to competing methods (Figs. 6.6b and 6.6c), our method successfully reconstructs the handle of the cup in Fig. 6.6d.

**Limitations of Local Methods** Despite of the high research attention that local methods have gained over the last couple of years, they do not render global approaches useless. The problem is that even large support windows are not sufficient for handling highly ambiguous data, that is, very untextured image regions. To illustrate this, we have used our PatchMatch stereo algorithm [14] that is considered to represent the state-of-the-art in local matching to compute the disparity map for the Middlebury Plastic image. As can be seen from Fig. 6.7c, the algorithm fails to handle the completely untextured yellow surface, as this high ambiguity cannot be resolved completely locally. In such cases, a global smoothness prior is essential. To demonstrate this, we have embedded PatchMatch stereo as a data term in a variant of the second order stereo algorithm of [81] (see our paper [14] for details on how this is accomplished). As can be seen from Fig. 6.7d, the yellow plastic surface can now be handled successfully. Note that while local stereo approaches have started to take over the top positions in the Middlebury benchmark [67] from global methods, this is not the case on more realistic imaginary such as that of the very recent KITTI street scene benchmark [27] that is dominated by global methods.



**Fig. 6.7** Limitations of local methods. **(a)** Left image of the Middlebury Plastic pair. **(b)** Ground truth disparity map. **(c)** Results of our local algorithm PatchMatch stereo [14]. The *yellow plastic surface* is poorly reconstructed due to the lack of texture. **(d)** Result of a global method. The global smoothness prior is essential to correctly reconstruct the *yellow surface*

The important point is that the aggregation schemes of local methods can supplement global methods in the form of a stronger (less ambiguous) data term. Due to the use of adaptive support weights, aggregation also improves the performance of global methods near depth discontinuities. Slanted support windows may also improve the results of global algorithms at highly slanted surfaces. Finally it is important to note that many robust match measures require a window to be computed (e.g., Census or Normalized Cross Correlation). As described later, such window-based measures are specifically important on realistic images with illumination differences, for example, the left image is darker than the right one.

Finally, we would also like to mention another limitation of local algorithms, that is, it is difficult to incorporate occlusion handling directly in the matching process. Typically occlusions are treated in a post-processing step by left-right consistency checking [23]. This means that the disparity map for the right image is computed in addition to that of the left view. The check then invalidates pixels of the left view whose disparity value is different in the right image. In the final step, these invalidated pixels are filled by replicating disparities of spatially close valid pixels (e.g., see our filling procedure in [61]). As we will see in the next section, global methods allow for occlusion handling directly in the matching process by modeling the occlusion problem in their energy functions.

### 6.3 Global Methods

**The Principle of Global Algorithms** Global methods differ from local ones in that they express the smoothness assumption in an explicit form via a so-called smoothness term. Global algorithms define an energy function  $E(D)$  that measures the quality of a disparity map  $D$ . In the subsequent optimization step, the goal is to find the disparity map of lowest energy, that is, of highest quality according to our energy function. The typical form of a stereo energy function is:

$$E(D) = E_{\text{data}}(D) + \lambda \cdot E_{\text{smooth}}(D). \quad (6.8)$$



**Fig. 6.8** Disparity maps for increasing values of  $\lambda$  in (6.8). The *left-most image* shows the result that is optimal according to the data term ( $\lambda := 0$ ), while the *right-most image* shows the disparity map optimal according to the smoothness term ( $\lambda := \infty$ )

Here,  $\lambda$  is a user-defined parameter that balances the influence of  $E_{\text{data}}$  and  $E_{\text{smooth}}$ . The data term  $E_{\text{data}}$  measures photo consistency and is defined as

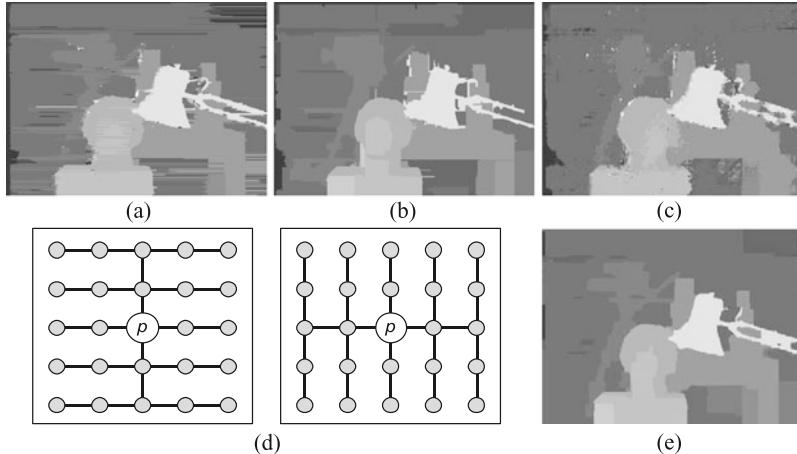
$$E_{\text{data}}(D) = \sum_{p \in I_l} c(p, p - d_p) \quad (6.9)$$

where  $I_l$  denotes all pixels of the left image and  $c(p, p')$  computes the pixel dissimilarity between pixel  $p$  of the left view and pixel  $p'$  of the right image (e.g., absolute difference of intensity values).  $d_p$  represents the disparity value of pixel  $p$  in disparity map  $D$ . Let us now focus on the smoothness term  $E_{\text{smooth}}$ . It is responsible for preferring disparity maps that are spatially smooth over such that are not by assigning lower energy.  $E_{\text{smooth}}$  is defined as

$$E_{\text{smooth}}(D) = \sum_{(p,q) \in \mathcal{N}} s(d_p, d_q) \quad (6.10)$$

where  $\mathcal{N}$  denotes all pairs of spatially neighboring pixels in the left image. Note that since the term is defined on pairs of pixels, it is often referred to as pairwise term, whereas the data term is referred to as unary term. The function  $s(d_p, d_q)$  assigns a penalty if  $p$ 's disparity is different from that of  $q$ . As discussed below, there are different options for implementing the smoothness function  $s()$ , which also defines the complexity of the optimization problem. To show the influence of  $E_{\text{data}}$  and  $E_{\text{smooth}}$ , we plot disparity maps for the Tsukuba images using different settings of  $\lambda$  in (6.8) in Fig. 6.8. Note that if we set  $\lambda := 0$ , that is, switch off the smoothness term, then our method “degenerates” to a purely local one.

**Optimization** Let us now focus on the problem of finding a disparity map that minimizes (6.8). This problem is difficult as the disparity assignment of each pixel influences the disparity assignment of every other pixel of the image via the smoothness term. This is known as the “Knock-on” effect in literature. It is known that even for the simplest discontinuity-preserving smoothness function, that is, the Potts model where  $s(d_p, d_q) = 0$  if  $d_p = d_q$  and  $s(d_p, d_q) = 1$  otherwise, finding the global minimum of (6.8) is an np-complete problem (see [18] for a formal proof). However, there exist powerful optimization strategies that approximate the energy minimum. In the following, we will discuss three of them, that is, (1) dynamic programming, (2) graph-cuts and (3) message passing. Note that there is also



**Fig. 6.9** Dynamic programming-based methods. **(a)** The scanline optimization method of [67]. The disparity map suffers from horizontal streaks. **(b)** The tree-based method of [79]. The streaking problem is reduced, but in addition to horizontal streaks, vertical streaks are introduced. **(c)** Our reimplementation of [32]. The streaking problem is eliminated, but the algorithm produces isolated pixels. **(d)** The tree structures used in our work [10]. **(e)** Corresponding disparity map. Our disparity result does not suffer from streaks or isolated pixels

an online competition [74] that compares the effectiveness of optimization strategies.

**Dynamic Programming** As stated above, exact minimization of (6.8) is an np-complete problem in general. However, there exists a special case. If the smoothness interactions ( $\mathcal{N}$  in (6.10)) form a tree in the image grid, the global optimal solution can efficiently be computed via dynamic programming (DP). The simplest method to modify  $\mathcal{N}$  such that this set does not contain cycles is to remove all vertical smoothness interactions and this is exactly what most DP-based algorithms do (e.g., [6, 17, 58] or the scanline optimization method of [67]). In this case, each horizontal scanline is optimized independently from the others and this leads to the well-known scanline streaking problem (see Fig. 6.9a). This streaking problem is the reason why such methods clearly fall behind the state-of-the-art in stereo matching [67].

Veksler [79] proposes a smarter way to transform  $\mathcal{N}$  into a tree that contains horizontal as well as vertical smoothness edges. The idea is that some smoothness edges are more important than others. In particular, if two neighboring pixels are similar in color, they are likely to lie on the same disparity. Hence, the corresponding smoothness edge is more important than one connecting two pixels of different color. This prioritization is then transformed into weights and the tree is built using a minimum spanning tree algorithm. While this approach reduces horizontal streaks, it, however, introduces vertical streaks (see Fig. 6.9b).

One of the most popular stereo algorithms (e.g., implemented in the Open Computer Vision library) is the semi-global matching approach [32]. Note that we will

describe it from the viewpoint of tree DP, which is very different from the description given in [32].<sup>5</sup> Each pixel  $p$  represents the root of a tree. This tree has a star-shaped form, i.e., captures all pixels that lie on  $p$ 's horizontal, vertical and diagonal scanlines. The root  $p$  is then assigned to the disparity of optimal costs on this tree structure. As can be seen from Fig. 6.9c, the approach effectively overcomes the scanline streaking problem, but produces isolated pixels. The problem is that the star-shaped tree is composed only of a subset of all image pixels. This subset may not capture enough texture to derive the correct disparity at the tree's root.

**Our Contribution—Simple Trees [10]** Our Simple Tree approach [10] also constructs individual trees per pixel. However, in contrast to [32], our trees contain all pixels of the reference view and hence the problem of missing texture (see above) is avoided. As shown in Fig. 6.9d, we use two different trees at each pixel  $p$ . The horizontal tree (Fig. 6.9d (left)) captures all horizontal smoothness edges as well as the vertical smoothness links of  $p$ 's scanline. The vertical tree (Fig. 6.9d (right)) represents the complementary tree, i.e., captures all vertical smoothness links and the horizontal edges of  $p$ 's scanline. Note that there is high redundancy in computation of all tree's optima. This can be accomplished with a few DP scanline passes, which takes less than a second on standard images and in a CPU-based implementation.<sup>6</sup> Hence, the algorithm shows an excellent speed versus quality trade-off. We show our Tsukuba result in Fig. 6.9e. As can be seen, this result is free of scanline streaks and isolated pixels.

**Graph-Cuts** Graph-cuts have been used in stereo vision for some years (e.g., [66]). In the following, we will speak about “modern” graph-cut based algorithms, so-called move making algorithms. These algorithms are capable of effectively minimizing (6.8) and have been the winner in the optimization benchmark of [74] on some problem instances. The idea is to start with an arbitrary disparity map<sup>7</sup> and then to iteratively apply moves. Here, a move leads to a new disparity map of lower (or at least the same) energy. There are two well-known moves, that is,  $\alpha\beta$ -swaps and  $\alpha$ -expansions [18]. We will focus on  $\alpha$ -expansions, because they are known to outperform  $\alpha\beta$ -swaps. To apply an  $\alpha$ -expansion, a discrete disparity label  $d_\alpha$  is selected. The  $\alpha$ -expansion move changes the disparity map such that each pixel either keeps its current disparity or changes it to  $d_\alpha$ . The challenge is to compute the  $\alpha$ -expansion that leads to the largest decrease of energy (6.8) among all possible  $\alpha$ -expansions. This problem can be solved exactly by computing the minimum cut

---

<sup>5</sup>Hirschmüller stresses a relationship to local methods. He calls the method semi-global matching, as he uses the aggregation step of local methods, but aggregates (global) DP path costs instead of pixel dissimilarities of spatially close pixels.

<sup>6</sup>It is likely that the approach would run in real time if implemented on a modern GPU.

<sup>7</sup>Note that although we speak about disparity, these move making algorithms can be applied to arbitrary computer vision labeling problems outside stereo vision.

/ maximum flow in a special purpose graph.<sup>8</sup> Note that this is only true if the energy is sub-modular [18], which is for example, the case for the Potts model. There are many stereo algorithms that apply  $\alpha$ -expansions as their optimization engine (e.g., [35, 46, 47, 52] or our algorithm [9]). Note that graph-cuts can optimize more complex energies than that of (6.8) such as that discussed below.

Lempitsky et al. [50] have recently proposed a new move, that is, the fusion move that is interesting because it enables graph-cut-based optimization over continuous disparity values. The idea is to have two proposal disparity maps. In a fusion move, each pixel either takes the disparity of proposal 1 or that of proposal 2. Note that if proposal 2 only has a single disparity for all pixels, then this fusion move is identical to an  $\alpha$ -expansion. Hence, fusion moves are a generalization of  $\alpha$ -expansions. The problem is that in the general case the optimization graph contains non-submodular edges, and hence computing the optimal fusion move becomes an np-complete problem. However, there are graph-cut based algorithms, i.e., quadratic pseudo-boolean optimization (QPBO) [45] that can handle non-sub-modular energies.<sup>9</sup> The fusion move framework has been used in the second order stereo algorithm [81] and our algorithms [13, 15].

**Message Passing** The most prominent algorithm based on message passing is Belief Propagation (BP) [22]. BP is an iterative procedure where each pixel communicates with its four spatial neighbors via sending messages. A message is thereby a vector that has an entry for each allowed disparity. A message  $m_{pq}^t(d_q)$  sent from pixel  $p$  to pixel  $q$  at iteration  $t$  encodes  $p$ 's belief that  $q$  should be assigned to a particular disparity  $d_q$  and is computed as

$$m_{pq}^t(d_q) = \min_{0 \leq d_p \leq d_{\max}} \left( s(d_p, d_q) + c(p, p - d_p) + \sum_{r \in \mathcal{N}(p) \setminus \{q\}} m_{rp}^{t-1}(d_p) \right) \quad (6.11)$$

where  $d_{\max}$  is the maximum allowed disparity and  $\mathcal{N}(p) \setminus \{q\}$  denotes  $p$ 's spatial neighbors excluding  $q$ . After  $T$  iterations the final disparity  $d_q^*$  at pixel  $q$  is computed as

$$d_q^* = \operatorname{argmin}_{0 \leq d_q \leq d_{\max}} \left( c(q, q - d_q) + \sum_{p \in \mathcal{N}(q)} m_{pq}^T(d_q) \right). \quad (6.12)$$

Note that by using the strategy of [22] the computational complexity of calculating the messages (6.11) for all image pixels can be reduced to  $\mathcal{O}(N \cdot d_{\max})$  where  $N$  is the number of pixels. This makes BP fast and GPU-based real time implementations

<sup>8</sup>This is why these algorithms are referred to as graph-cuts.

<sup>9</sup>Due to the np-hardness of the problem, QPBO can only guarantee to find a part of the global optimal solution and will, in general, leave a subset of pixels unlabeled. The autarky property of QPBO guarantees that by assigning unlabeled pixels to the disparities of proposal 1 the energy of the fusion result will be lower or the same as that of proposal 1. Alternative ways for handling these unlabeled pixels are QPBO-I and QPBO-P [64].

for stereo are available [83]. BP forms the optimization engine for many stereo algorithms (e.g., [43, 71, 72, 75]).

Finally, we also mention tree-reweighted message passing (TRW) [80], which is similar to BP, but uses a different message update rule. TRW allows to compute a lower bound on the energy. Comparison of the current energy against this lower bound gives an estimate of how close the current solution is to a global optimal one [54, 74], that is, if the energy is equal to the lower bound, a global energy optimum has been found.

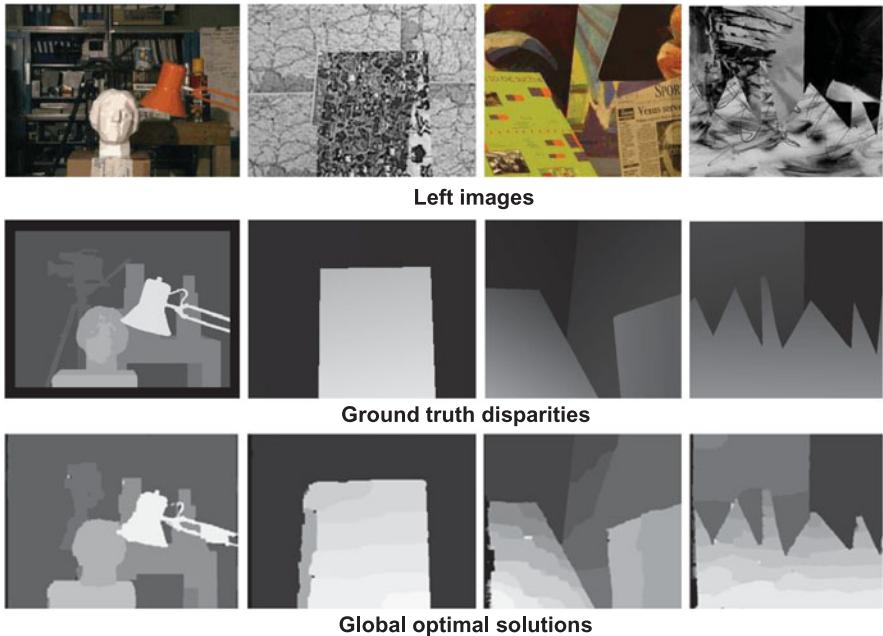
**The Problem Is the Model** In general, when an energy minimization approach delivers suboptimal results, there are two reasons: (1) The energy function does not represent a good model of the stereo problem or (2) the optimization technique fails to effectively minimize the energy. The problem is that one often does not know whether bad performance is for the first or the second reason (or a combination thereof), since the global energy optimum is usually unknown.

Recent studies give indication that with the use of modern optimization schemes (graph-cuts or TRW) the formulation of the energy function has become the limiting factor. For example, in the above mentioned optimization algorithm comparison [74], the authors have shown that the best-performing optimization algorithms applied on the energy of (6.8) are capable of computing local energy minima extremely close to the exact solution. However, it has been pointed out that, due to flaws in the energy formulation, this does not necessarily translate into improved reconstruction performance (with respect to ground truth data). A specifically interesting result of this study (also noted in [77]) is that the energy of the ground truth image is considerably higher than that produced by modern optimization algorithms. This clearly shows that the energy function represents an unsatisfactory model for the stereo problem.

This is also consistent with the findings of Meltzer et al. [54], who have shown that, despite the np-hardness of the optimization problem, an exact optimum can be obtained for some standard benchmark stereo pairs (the old Middlebury set [67]) using tree-reweighted message passing. The corresponding results are shown in Fig. 6.10. Even the global optimal solution has not led to a better disparity reconstruction in comparison to standard energy minimization approaches. Consequently, the authors have concluded that “*the problem is not in the optimization algorithm, but rather in the energy function*”. In other words, progress in stereo can only be achieved by improving the energy function. In the remainder of this chapter, we will discuss options to improve the stereo model.

**Data Term** Let us start by investigating the data term ((6.9) of our energy (6.8)). Note that from our experience, the match measure has a very large influence on the quality of disparity maps (also see our papers [7, 11]), which is oftentimes larger than the influences of smoothness terms and optimization algorithms.

There are various options to implement the pixel dissimilarity function  $c()$ . The simplest strategy is to compute the absolute difference (AD) of intensity values, that is,  $c(p, q) = |I_p - I_q|$  where  $I_p$  denotes the intensity at pixel  $p$ . An alternative



**Fig. 6.10** Global optimal solutions for energy (6.8) on the old Middlebury set. (The Potts model is used to implement the smoothness function.) Images are taken from [54]. Note that even the global energy minimum leads to disparity maps that are relatively far off from the ground truth solution (e.g., see handle of the lamp or the camera in the Tsukuba image). This suggests that the energy formulation does not represent an ideal model for the stereo problem

approach is to calculate the squared difference (SD), that is,  $c(p, q) = (I_p - I_q)^2$ . According to the experiments of [67], there is relatively little performance difference between AD and SD, although SD is potentially more sensitive to outliers. It makes sense to truncate AD and SD values to reduce the influence of outlier pixels. While this improves matching performance [67], the selection of the threshold value that should be chosen according to the images' noise level represents a problem. Birchfield and Tomasi [5] present a match measure that reduces the influence of image sampling. The idea is to perform linear interpolation to derive the intensity at sub-pixel locations and to check whether these sub-pixel intensities lead to lower matching costs. From our experience, this match measure does not necessarily improve quality.

An important weakness of all above dissimilarity functions is that they cannot handle radiometric distortions (e.g., the left image is darker than the right one). Note that in practical stereo such radiometric distortions are almost always present and the treatment of this problem is vital. There are various competing radiometric insensitive measures and an evaluation of which has been conducted in [33].

The first option is to transform left and right images via a filter in a pre-processing step. Matching is then performed on the filtered images using a standard measure such as AD or SD. An edge filter (e.g., Sobel) is an obvious choice. Note that the

gradient is not affected if radiometric differences are due to a constant intensity offset, e.g., the intensity of corresponding points is always 10 levels higher in the right image.<sup>10</sup> An alternative filter to handle intensity offsets is to subtract the mean computed in a window from the original intensity images. Note that since the mean is computed within a window, this leads to edge fattening. Hence, a better option is to subtract the image processed by an edge-preserving filter (e.g., bilateral or guided filter) from the original image. According to the experiments of [33], the filter-based methods fall behind the match measures discussed next in terms of quality.

Mutual Information (MI) as implemented in [32] builds a single model of the intensity change for the whole image, that is, given intensities  $I_p$  and  $I_{p'}$  of pixels in left and right images, the MI score (looked up from the global model) gives the likelihood that  $I_p$  and  $I_{p'}$  match each other. To compute this global model a disparity map is required, which leads to a chicken-and-egg problem. This dilemma can be solved in an iterative fashion. Given an initial disparity map, the MI scores are computed. These MI scores are then used to compute a new disparity map and so on.<sup>11</sup> The advantage is that MI is a pixel-based measure, that is, there is no window required. Hence, edge fattening and problems at slanted surfaces are avoided. This potential advantage should, however, be regarded in the light of new aggregation schemes that use adaptive support weights and slanted windows as discussed in Sect. 6.2. The main disadvantage is that radiometric distortions are usually not the same all over the image (e.g., only the left bottom image part is darker in the left view). Such local changes cannot be handled by the global distortion model, which may represent the reason why MI has been outperformed by the window-based measures described next in the study of [33].

Zero mean Normalized Cross-Correlation (ZNCC) is a window-based measure than can handle intensity offsets and gains. It is defined as

$$c(p, p-d) = \frac{\sum_{q \in W_p} (I_q - \bar{I}_p)(I_{q-d} - \bar{I}_{p-d})}{\sum_{q \in W_p} (I_q - \bar{I}_p)^2 \sum_{q \in W_p} (I_{q-d} - \bar{I}_{p-d})^2} \quad (6.13)$$

where  $W_p$  represents a squared window centered at  $p$  and  $\bar{I}_p$  denotes the mean intensity computed over all pixels inside  $W_p$ . A second popular window-based measure is Census. Instead of directly matching intensities (that may be affected by radiometric distortion), Census generates a bit string representation that describes the texture within the window. Each pixel  $q$  of the window is thereby compared against the center pixel  $p$ . If  $I_q < I_p$  then 0 is concatenated to the bit string and 1 otherwise. The Census costs between the reference window and the window in the match frame are then computed as the Hamming distance between the corresponding two bit strings.

---

<sup>10</sup>In practice, it is sufficient to compute the gradient in x-direction, as vertical edges contain more disparity information than horizontal ones.

<sup>11</sup>Hirschmüller [32] uses a hierarchical approach to speed up this iterative procedure.

**Our Contribution—The Role of Color [7]** So far we have only spoken about matching intensities and ignored the fact that color information is typically available. Note that although color obviously represents additional information, many researcher still convert the input images to grey-scale when computing the match measure. We have published two evaluation papers [7, 11] that investigate the usefulness of color.

In the first paper [11], we have concentrated on the standard match measures AD and SD (see above) that we compute in eight different color spaces<sup>12</sup> as well as on the intensity images. We apply the Simple Tree method [14] (see above) to optimize energy (6.8) using the resulting match measures as data terms. We evaluate on a relatively large test set of 30 Middlebury ground truth pairs. We report a considerable quality improvement when using color. The best-performing color space gives 25 % less disparity errors in comparison to only using intensity.

In our follow-up paper [7], we take a closer look at the image regions where color improves matching performance over intensity-based matching when using AD or SD. The key insight is that these regions correspond to image areas that are affected by radiometric distortions. The paper shows that the benefit of color is small, given that considerably better performance at radiometric distorted regions is achieved by directly using a radiometric insensitive measure such as ZNCC and Census instead of color-based AD. An interesting observation is that in comparison to the intensity-based versions of ZNCC and Census, performance even decreases if these measures are “enriched” with color information. Therefore, we suggest not to use color at all, but radiometric insensitive measures computed on intensity images.

**Occlusion-Aware Data Term** The data term defined in (6.9) has a systematical problem in that it does not consider the occlusion problem. It does not make sense to compute a match measure for occluded pixels, as the matching point in the other image does not exist. Note that ignoring the occlusion problem leads to wrong disparity results near disparity discontinuities, that is, in occluded regions and their proximity. For example, in the second column of Fig. 6.10, the disparity to the left side of the foreground object is completely wrong due to this problem.

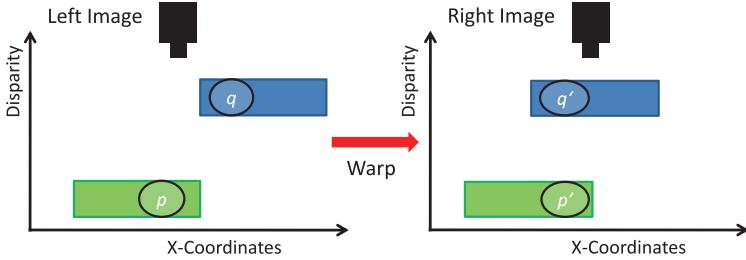
Let us now redefine (6.9) such that it accounts for occlusions:

$$E_{\text{data}}(D) = \sum_{p \in I_l} c(p, p - d_p)(1 - O(p)) + \lambda_{\text{occ}} O(p). \quad (6.14)$$

Here  $O(p)$  is a function that returns 1 if pixel  $p$  is occluded and 0 otherwise.  $\lambda_{\text{occ}}$  is a user-defined penalty for occlusion. This parameter is required to prevent the algorithm from maximizing the number of occluded pixels. Note that according to (6.14) the match measure is only evaluated for visible pixels, while we impose the occlusion penalty for occluded ones. The challenge is to implement the occlusion

---

<sup>12</sup>In our study, the extension of AD and SD to color works by computing AD and SD for each color channel separately. We then sum up the differences over the 3 color channels.



**Fig. 6.11** Occlusion Reasoning. The pixel  $p$  is occluded by pixel  $q$ , because both of them project to the same pixel in the *right image* and  $q$  has higher disparity than  $p$

function  $O(p)$  and we follow the definition of [81]:

$$O(p) = \begin{cases} 1 & \text{if } \exists q \in I_l : p - d_p = q - d_q \text{ and } d_p < d_q \\ 0 & \text{otherwise.} \end{cases} \quad (6.15)$$

Let us use Fig. 6.11 to explain this equation. We have two pixels  $p$  and  $q$ . We can use their disparity values  $d_p$  and  $d_q$  to warp them into the geometry of the right view, that is, we compute  $p' = p - d_p$  and  $q' = q - d_q$ . In our example,  $p'$  and  $q'$  both lie on the same x-coordinate in the right view and given that the pixels lie on opaque (non-transparent) objects, only one of them can be seen by the right camera. From Fig. 6.11 it is clear that the visible pixel is  $q'$  as it has higher disparity, that is, lies closer to the camera. This is exactly what is written in (6.15), that is, a pixel  $p$  is occluded if there exists a pixel  $q$  such that both pixels project to the same pixel in the right image and  $p$ 's disparity is smaller than that of  $q$ . Note that this data term can be optimized via graph-cuts, which has been done in [81].

Other approaches that handle occlusions in their energy functions are based on a symmetrical formulation, that is, disparity maps are computed for both images and not just for the left one as we have done above. This makes sense as there is no plausible reason for treating the left image different from the right one.<sup>13</sup> Most authors (e.g., [46, 47, 52] and our paper [9]) implement the uniqueness assumption. This assumption enforces that each pixel has at most one matching point. Note that this assumption is broken for highly slanted surfaces where due to surface slant one pixel may have multiple (visible) correspondences in the other view [57]. Sun et al. [72] proposes a so-called visibility constraint to handle this problem and we show that the problem can be overcome using a surface-based representation in [13].

**Smoothness Term** Let us now concentrate on the smoothness term. As stated above, there are several possibilities to implement the smoothness function  $s()$  of (6.10). We will first focus on first order functions that penalize the gradient of the disparity map. Let us start with the linear smoothness function, i.e.,

<sup>13</sup>The stereo problem is symmetrical.

$s(d_p, d_q) = |d_p - d_q|$ . Note that from the viewpoint of optimization this linear function represents an interesting special case as exact optimization of (6.8) does no longer represent an np-complete problem, but can be accomplished by computing a single cut in a graph [66].<sup>14</sup> The problem of a linear function is that it is not ideal for the stereo problem. Let us assume that we want to reconstruct a disparity border where disparity abruptly changes from 5 to 15. The problem of the linear term is that reconstructing this disparity discontinuity via several small jumps (e.g., from disparity 5 to 10 and then a few pixels later from 10 to 15) is as expensive as the large jump (from 5 to 15) that we prefer. Hence, disparity borders are blurred.

As stated above, the simplest discontinuity preserving function is the Potts model, that is,  $s(d_p, d_q) = 0$  if  $d_p = d_q$  and  $s(d_p, d_q) = 1$  otherwise. While this term works well for reconstructing depth discontinuities, it is suboptimal for reconstructing slanted surfaces where there is a gradual change of disparity. Note that each of these small disparity transitions receives a large penalty, that is, the same penalty that we assign to depth discontinuities. The consequence is that slanted surfaces are reconstructed via various fronto-parallel ones.

A compromise between accuracy at disparity borders and accuracy at slanted surfaces is the truncated linear function, that is,  $s(d_p, d_q) = \min(|d_p - d_q|, k)$  where  $k$  is a user-defined truncation value. Let us assume that we set  $k := 3$  and again consider the example of reconstructing a depth discontinuity where disparity changes from 5 to 15. The costs of using two jumps (from disparity 5 to 10 and from 10 to 15) is 6, while that of the single large jump (from 5 to 15) is 3, that is, the single large jump is cheaper. This means that the model is discontinuity preserving. Let us now consider a small disparity transition (e.g., from 5 to 6) that occurs at a slanted surface. This transition only obtains a small penalty (1 in our example). The truncated linear term works well in practice and is implemented in many global stereo approaches (e.g., [32, 72, 85] to cite a few).

Note that none of the above first order smoothness functions is ideal for handling slanted surfaces. All of them assign higher energy to slanted surfaces than to fronto-parallel ones. This fronto-parallel bias cannot be justified as slanted surfaces are as likely to occur in natural images as fronto-parallel ones. The solution is to put a penalty on the curvature of the disparity map [51, 81], which is a so-called second order smoothness prior. This term is defined as  $s(d_o, d_p, d_q) = |d_o - 2d_p + d_q|$  and penalizes the deviation of the three disparity values from a line (regardless of the line's slant). Note that minimization of energy (6.8) under second order smoothness becomes complex as the smoothness function has three arguments, that is, leads to higher-order cliques in the optimization graph. However, for the triple cliques of the second order term, a scheme for transformation into pairwise cliques is known [48] and has provided the basis for the graph-cut-based optimization algorithm used in the second order stereo method of [81].<sup>15</sup>

---

<sup>14</sup>In general, such a construction works if the smoothness term is convex [40].

<sup>15</sup>Note that in [81], the smoothness term is also truncated to make it discontinuity preserving.

**Edge-Sensitive and Highly-Connected Smoothness Terms** In Sect. 6.2 on local stereo, we have stressed that the color image provides valuable information about the location of depth discontinuities. In particular, in most natural images, a disparity discontinuity is more likely to occur between neighboring pixels of different color than between homogeneously colored neighbors. Let us reformulate the smoothness term of (6.10) such that it incorporates this observation:

$$E_{\text{smooth}}(D) = \sum_{(p,q) \in \mathcal{N}} w(p, q) \cdot s(d_p, d_q). \quad (6.16)$$

This is known as an edge-sensitive smoothness term. The difference to the standard term is the weighting function  $w()$ . This function can, for example, be implemented using the color term of the bilateral weight function in (6.3), that is,  $w(p, q) = \exp(-\text{color}(p, q)/\gamma_c)$ . Note that a high weight is assigned if  $p$  and  $q$  are similar in color and a low weight otherwise. Hence, the energy for aligning depth discontinuities with color edges is lower than for placing depth discontinuities inside a homogeneously colored region.

The standard smoothness term has a bias towards reconstructing compact objects, because it assigns low energy if the length of a disparity border is small. Hence, it is not well-suited for reconstructing complex outlines and thin structures. This is known as the edge shrinking bias in literature. The edge-sensitive smoothness term attenuates this shrinking bias to some extent, but does not work sufficiently well in practice.

A promising approach to overcome the shrinking bias<sup>16</sup> is to increase the neighborhood, which leads to a so-called highly-connected smoothness term defined as

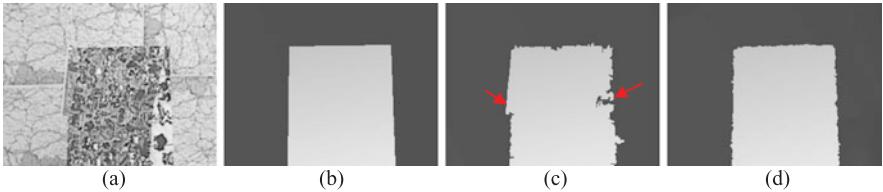
$$E_{\text{smooth}}(D) = \sum_{p \in I_l} \sum_{q \in W_p} w(p, q) \cdot s(d_p, d_q) \quad (6.17)$$

where  $I_l$  denotes all pixel coordinates of the left image. Note that the smoothness function  $s()$  is no longer only evaluated between a pixel  $p$ 's four spatial neighbors, but within a window  $W_p$  centered on  $p$ . An obvious choice for  $w()$  is the bilateral function of (6.3), that is, the strength of the smoothness connection decreases proportional to the distance and color dissimilarity of connected pixels. Note that this approach can be regarded as the “global version” of the (local) adaptive weight approach [86] and Smith et al. [69] have shown that it achieves impressive results on the stereo problem. However, due to the large number of smoothness edges, optimization (e.g., via graph-cuts) is computationally very expensive. To reduce the number of edges, Smith et al. [69] eliminate “less important” smoothness connections using a strategy similar to Veksler’s tree DP [79] (see paragraph on DP above), but their approach is still slow.

Ideally, the window  $W_p$  in (6.17) should capture all pixels of the whole image. This leads to a so-called fully-connected smoothness term where each pixel is directly connected to every other pixel via a smoothness link. Optimization of the

---

<sup>16</sup>There are also alternative methods (e.g., cooperative cuts [41]).



**Fig. 6.12** Limitations of segmentation-based methods. **(a)** The intensity image of the Middlebury Map pair. **(b)** Ground truth disparity map. **(c)** Due to segments that overlap disparity discontinuities, our segmentation-based method [8] produces inaccurate results at depth discontinuities (see red arrows). **(d)** Our method [13] that uses segmentation only as a soft constraint can successfully handle this image pair

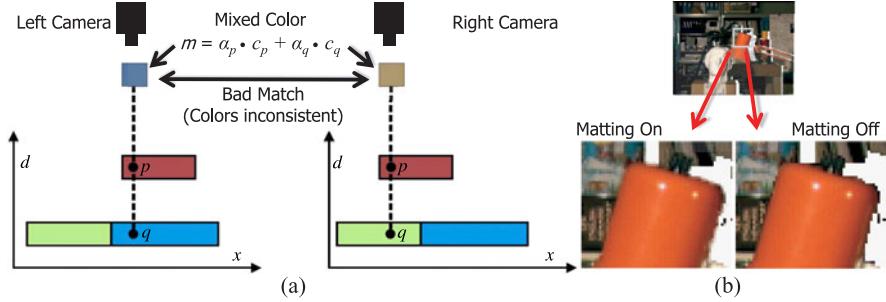
resulting energy is intractable using standard methods (e.g., graph-cuts or standard BP) even on small images. Krähenbühl and Koltun [49] have recently presented an efficient optimization algorithm for this problem based on message passing. The “trick” is to compute message updates via a fast edge-preserving filter operation.<sup>17</sup>

**Segmentation-Based Methods** Segmentation-based algorithms (e.g., [20, 35, 76, 90] and our papers [8, 9]) also use the color cue. They apply a color segmentation of the left input image. The assumption of segmentation-based algorithms is that disparity varies smoothly within a region of homogeneous color (i.e., within a segment), while depth discontinuities coincide with segment borders. Hence, instead of matching individual pixels, these methods match whole segments at once. Disparity within a segment is thereby assumed to be constant [90] or (in a more sophisticated approach) to lie on a slanted 3D plane [8, 9, 76].

Segmentation-based methods have been very popular for some time, most likely due to their excellent performance on the (new) Middlebury benchmark [67]. However, generalization to other images represents an issue. To illustrate this, we plot results on the Map test set in Fig. 6.12 that has traditionally represented a pitfall for segmentation-based methods in the old Middlebury benchmark. When segmenting the image of Fig. 6.12a there occur segments that overlap a depth discontinuity. Segmentation-based methods cannot recover from such situations, because they model whole segments via a single plane (see errors of our method [8] along the depth discontinuity in Fig. 6.12c). The second problem is the planar model that oversimplifies the 3D shape in the presence of rounded objects. To alleviate both issues, segmentation-based methods apply a strong oversegmentation, but as seen from Fig. 6.12c, this cannot completely overcome these problems. At a later point, we will discuss our Surface Stereo method [13] that can effectively handle both problems. We show the result of this method in Fig. 6.12d.

**Stereo and the Matting Problem** Before presenting our first algorithm that aims at providing a better model for the stereo problem, let us discuss the following problem. Natural images usually contain pixels that receive contributions from more

<sup>17</sup>It is interesting to note some similarity to cost filter approaches discussed in Sect. 6.2.



**Fig. 6.13** The matting problem in stereo matching and results of our approach [12]. **(a)** Due to lens blur, the *color* of pixels at object borders is a mixture between the *colors* of foreground and background objects. The problem is that this mixture is different in left and right images, which leads to color inconsistencies. **(b)** We use our stereo and matting results for novel view synthesis. Due to using our matting information, the lamp naturally melts with its new background in the *left image*. The *right image* is generated without matting information. Notice disturbing artifacts at the lamp’s border

than one real-world object as a consequence of lens blur. Such mixed pixels occur along depth boundaries where a point’s color is the composite of colors reflected by background and foreground objects. This so-called matting problem is well-known in photo-montaging. When cutting out an object and pasting it against a new background without considering mixed pixels, the results at the object’s border look unnatural as colors are “contaminated” by the color of the old background.

Let us look at Fig. 6.13a to understand the relevance of the matting problem in stereo matching. In this example, the goal is to reconstruct the depth of point  $p$ . Due to lens blur, the left camera sees a pixel color that is a mixture between the red foreground and the blue background, whereas in the right camera the red color of  $p$  is mixed with the green background. Hence  $p$  has different colors in left and right images, i.e., the photo consistency assumption is broken. This makes stereo matching difficult.

Although early work [2, 73] on stereo matching in combination with matting dates back 15 years, there has been relative little progress since then. Some authors [30, 90] compute alpha matting information in a postprocessing step, that is, after computing a disparity map. Xiong and Jia [82] propose a method that can handle the color inconsistency problem of Fig. 6.13a using a joint stereo and matting formulation. However, their approach only works for images that consist of exactly two layers (i.e., foreground and background layers) and generalization to other images is an issue. Taguchi et al. [75] propose a method that can handle an arbitrary number of layers. However, they do not use matting information to overcome the color inconsistency problem.

**Our Contribution—Combined Stereo and Matting [12]** We present an algorithm for solving the stereo matching and alpha matting problems simultaneously. In contrast to the papers above, our method operates on an arbitrary number of layers and still handles the color inconsistency problem. Our algorithm is based on

image segmentation (see discussion of such methods above). Hence, we first apply color segmentation to the left image. Each segment is then enlarged using morphological dilation. The idea is that since mixed pixels typically occur at disparity (segment) borders (see discussion above), we only allow transparencies in a small band around segment borders. In accordance to alpha matting literature, we call this band a segment’s unknown region. Note that due to this enlargement, we have many overlapping segments.

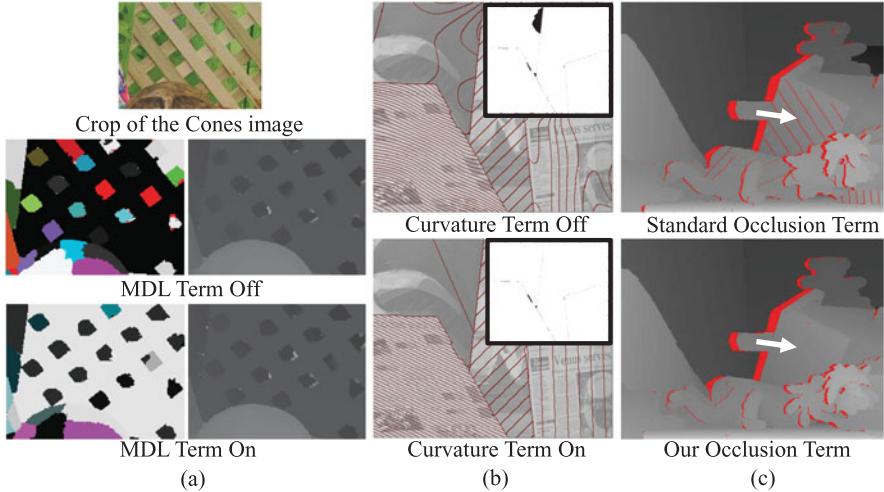
We now estimate a disparity plane for each overlapping segment and an alpha value as well as the true color for each pixel via energy minimization. Given alpha values and colors, our energy function computes the mixed colors for the left view. If alpha values and colors were correct, the resulting artificial image should be very similar to the real left view. Our energy function also computes an artificial right view via warping the pixels into the geometry of the right image according to our current disparity solution. This artificial right view is then compared against the real right image to measure the quality of alpha values, colors and disparity planes. The crucial question in generating the artificial right view is how alpha-values are affected by image warping. To correctly perform the warping operation, we introduce the concept of solidity (see our paper).

In the experimental results, we show that our method can compete with the state-of-the-art on the Middlebury benchmark [67]. More importantly, in contrast to all methods listed on Middlebury, our method also provides matting information. This makes it an ideal algorithm for performing depth segmentation and novel view generation. Figure 6.13 shows a novel view example, that is, an image that shows the scene from a virtual viewpoint not recorded by the stereo cameras. As can be seen, our matting information leads to realistic results at object borders when a foreground object is pasted against a new background.

Let us recall a limitation of segmentation-based methods, that is, they fail if a segment overlaps a depth discontinuity. This is only partially true for our method, as it can change the shape of segments by adjusting the alpha values within a segment’s unknown region. However, the algorithm fails in the presence of large segmentation errors. A more sophisticated solution is discussed next.

**Our Contribution—Surface Stereo [13]** In our paper [13], we stress the importance of a surface-based representation. Instead of directly assigning pixels to disparity values, we assign them to 3D surfaces. In our implementation, surfaces thereby correspond to planes and B-Splines (to model rounded surfaces). We claim that this surface-based representation is superior in comparison to the commonly-used disparity-based one as it allows incorporating energy terms that are very challenging or even impossible in the disparity-based representation.

Our approach simultaneously infers (1) which surfaces are present in the scene and (2) which pixels belong to which surface. We search an assignment of pixels to surfaces that minimizes an energy. Apart from data and smoothness terms (as in the standard energy of (6.8)), our energy function includes three new terms, that is, (1) a soft segmentation term, (2) a minimum description length (MDL) term and (3) a curvature term. We will discuss these terms in the following.



**Fig. 6.14** Effect of different terms in the energy of Surface Stereo [13]. **(a)** MDL term. **(b)** Curvature term. *Contour lines* of the disparity maps are superimposed on the left intensity image. **(c)** Our improved occlusion handling strategy. See text for explanation

As stated above, segmentation-based methods fail if (1) segments overlap disparity discontinuities (see Fig. 6.12c) or (2) if the planar model oversimplifies the real 3D shape (e.g., rounded objects). Note that we handle problem (2) by using a more general surface model, that is, a B-spline model. Problem (1) is handled by our soft segmentation term. This term prefers disparity maps that are consistent with a precomputed color segmentation over such that are not by assigning lower energy. Note that in contrast to most segmentation-based methods, inconsistent disparity maps are allowed (at the price of higher energy) and hence segmentation is only used as a soft constraint. For each pixel  $p$ , the soft segmentation term builds the intersection of  $p$ 's segment with a squared window centered at  $p$ , which gives us  $p$ 's subsegment. The soft segmentation term imposes 0 costs if all pixels within  $p$ 's subsegment are assigned to the same surface and a constant penalty, otherwise.<sup>18</sup> Fig. 6.12d demonstrates that our soft segmentation term can handle the Map test set where color segmentation is misleading. The reason is that the data term of our energy overrules the segmentation term and supports the correct disparity solution.

The MDL term implements the assumption that a simple scene explanation is better than an unnecessarily complex one. We therefore impose a penalty on the number of surfaces, that is, a solution containing five surfaces is cheaper than one

<sup>18</sup>This term forms a higher-order clique in the optimization step. In general, optimization of such higher-order cliques is intractable. We take advantage of the fact that these cliques are sparse, i.e., only one state generates 0 costs and all other states generate constant costs. There exist graph-cut based algorithms that can “efficiently” optimize such sparse higher-order cliques [44, 65] and we make use of them.

with 100 surfaces. Figure 6.14a shows the effect of the MDL term on a crop of the Middlebury Cones images. The interesting part is the background surface behind the fence. If the MDL term is switched off, the green squared background regions are all modeled by different surfaces. (We use a color coding in Fig. 6.14a where all pixels assigned to the same surface carry the same color.) In contrast, a large portion of the isolated background regions are correctly modeled by a single surface if our MDL term is switched on.

Our curvature term puts a penalty on the second order derivative of surfaces. Note that this term is very similar to the second order smoothness term of [81] (see paragraph about smoothness terms), but has two advantages. First, we can analytically and exactly compute the second order derivative, because we have a surface-based representation. This is in contrast to [81] where curvature is only approximated from the disparity values of three neighboring pixels. Second, our curvature term represents a unary potential in optimization and we can therefore avoid the complex and time-consuming triple clique construction of [81]. Figure 6.14b shows the effect of the curvature term. We switch off the curvature term in the top figure. Due to low texture and no penalization of curvature, the spline in the background (top left part of the image) erroneously adapts to the data. This leads to a large disparity error (see black pixels in the error image). When using the curvature term (see bottom figure) representation of the background via a plane leads to lower energy than modeling it as a spline and hence the correct disparity solution is obtained.

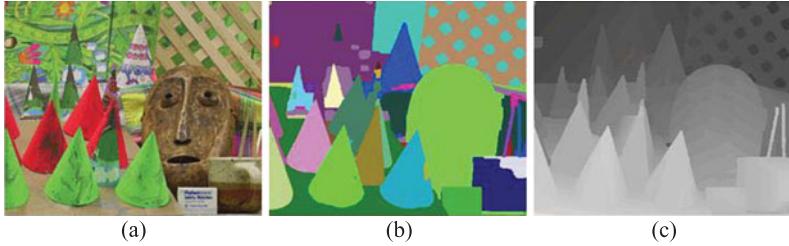
Finally, we also make a contribution in the context of occlusion treatment. As stated in the paragraph on occlusion handling, the uniqueness assumption is broken for slanted surfaces where a pixel of one view may have multiple correspondences in the other view. This leads to wrongly detected occlusions. For example, in the top image of Fig. 6.14c, occlusions are erroneously detected on the roof of the toy house. (We use red color to mark occlusions.) We overcome this problem by stating that two pixels must not occlude each other if they originate from the same surface. As seen from the bottom image of Fig. 6.14c, this avoids wrongly detected occlusions on the roof.

Let us now focus on optimization of the resulting energy, which is accomplished using fusion moves [50] (see paragraph on graph-cuts). We start with an initial solution. We have a proposal generator that produces six different types of proposal solutions (see paper for details). We now compute the “optimal” fusion move of our current solution and a proposal obtained from our generator.<sup>19</sup> The fusion result is then fused with the next proposal of our generator and so on.

In the experiments, we show that our method achieves an excellent Middlebury ranking, that is, rank six out of 74 methods at the time of publication. On the complex Teddy test set, our method even takes the first rank on all error measures. Moreover, we demonstrate that each term of our energy contributes to the quality of results (see paper for more information).

---

<sup>19</sup>By optimal we mean the solution that leads to the largest energy decrease according to our energy function among all possible fusion moves.



**Fig. 6.15** Object Stereo [15]. (a) Left image of a stereo pair that forms the input of the algorithm. (b) Computed object labeling. (Pixels of the *same color* lie on the same object.) (c) Computed disparity map

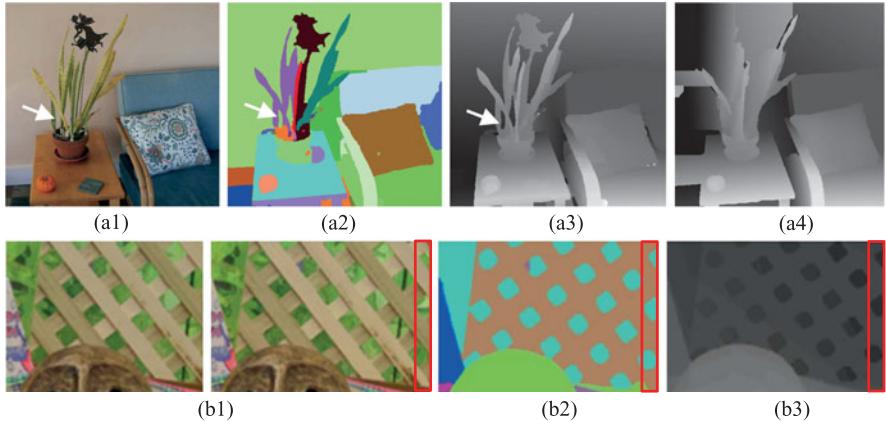
**Our Contribution—Object Stereo [15]** In [15], we combine stereo matching with the object segmentation problem where the task is to label pixels that belong to the same object. Figure 6.15 gives an example. The method’s input is formed by left and right views of a stereo pair. Our algorithm then jointly computes a segmentation of the image into objects (Fig. 6.15b) and a disparity map (Fig. 6.15c). The idea is that there exist synergies between the stereo and object segmentation problems. We describe our model that is encoded in an energy function in the following.

Our model considers the scene as a collection of a small number of objects.<sup>20</sup> An object is characterized by two aspects. First, an object contains a color model. We use this color model to implement the assumption that pixels of the same object are similar in appearance, that is, in color. The distribution of colors is thereby modeled using a Gaussian mixture model. Second, an object is characterized by a surface model, which implements the assumption that pixels of the same object lie approximately on the same 3D plane. We use a plane-plus-parallax model to allow deviations from this plane. An obvious aspect of our model is that disparities of pixels shall be estimated such that photo consistency is maximized (which represents a standard stereo data term). Finally, we introduce a constraint stating that an object shall be connected in 3D and describe this in more detail below.

Our model allows depth reasoning based on the appearance cue. This is because pixels of similar appearance (color) are grouped in the same object and the disparities for these pixels are biased to lie on the object’s plane. Advantages of our color-based depth reasoning are twofold. First, similar to Surface Stereo, this allows implementing color segmentation as a soft constraint. Second and more importantly, this allows to recover disparity for regions where stereo information is poor (low texture) or not present at all (occlusions).

For example, consider the image of Fig. 6.16a1 that has a challenging background of very low texture. Especially, the background region marked by the arrow poses a problem as the data costs for this region are highly ambiguous (no texture) and the pairwise smoothness term motivates to assign this isolated region to the foreground disparity. Our algorithm looks at the color of this small region. It finds that

<sup>20</sup>Small number means that we apply the MDL term above to penalize the number of objects.

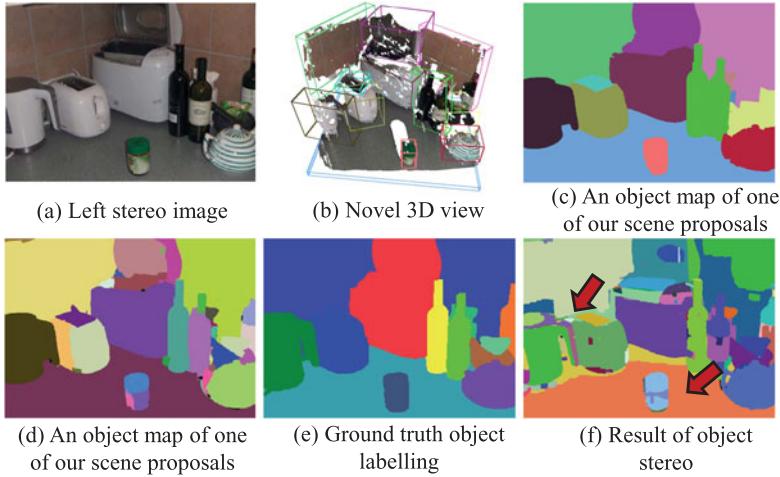


**Fig. 6.16** Results of Object Stereo [15]. We can handle regions of very low texture (a) and regions that are completely occluded (b). (a1) Left image of our Fairy test set. The region marked by the white arrow is difficult to match. (a2) Computed object labeling. (a3) Computed disparity map. Note that the disparity of the background is correctly recovered, whereas the competitor Surface Stereo [13] fails badly in these areas as shown in (a4). (b1) Left and right images of a crop of the Cones images. Note that the red rectangle marks pixels that are completely occluded in the left view. Our method combines all green pixels in a single background object as shown in (b2). This helps to find the correct depth even for the completely occluded image regions (b3)

the color matches that of the background object “Wall” and biases the disparity towards the plane of the object “Wall” (Fig. 6.16a3). Note that our Surface Stereo method described above fails on this challenging pair (Fig. 6.16a4), because it does not apply color-based depth reasoning.

Figure 6.16b shows a challenging occlusion case. We show crops of left and right views of the Middlebury Cones images in Fig. 6.16b1. Note that the isolated green background regions inside the red rectangle are only visible in the right image, that is, they are completely occluded in the left image. This makes correct computation of depth impossible for standard algorithms. Note that standard methods would extrapolate disparity via the smoothness term, that is, they would assign the occluded background regions to the disparity of the (foreground) fence. In contrast, our method combines all isolated squares into a single background object based on their green color (Fig. 6.16b2). Disparities inside the isolated regions are then biased towards the disparity of the background plane, which gives the correct disparity despite of some green regions being completely occluded (Fig. 6.16b3).

Let us now focus on our 3D connectivity constraint mentioned above. This constraint states that disconnected 2D regions in an image may belong to the same object only if they are separated by an occluding object with smaller depth (that is, closer to the camera). According to this constraint, the green squared background regions of Fig. 6.16b1 may all belong to the same object, because they are separated by an occluder of smaller depth, that is, the fence. In contrast to this, the two green cones of Fig. 6.15a must not lie in the same object. Although both cones are similar in color and depth, there is no occluder that separates them. Note that our 3D con-



**Fig. 6.17** Our approach [16]. Given a stereo pair (a), our algorithm jointly estimates a 3D reconstruction (b) and object maps (c), (d) using physics-based reasoning. The result is considerably closer to ground truth (e) than the one of Object Stereo [15] (f)

nnectivity constraint demonstrates the usefulness of depth information for the object segmentation task, that is, this constraint can only work, because we have depth. In practice, it helps to improve object segmentation results.

At the time of publication of the corresponding paper [15], Object Stereo has taken rank 10 on Middlebury and the first rank on the challenging Cones images. Note that the major advantage over other stereo algorithms is that Object Stereo also provides a segmentation of the input images into objects.

**Our Contribution—Physics-Based Segmentation and Stereo [16]** As in Object Stereo, our paper [16] concentrates on jointly computing a disparity map as well as a segmentation of the input images into objects. One main difference to Object Stereo is that we give objects a third dimension. In contrast to stating that an object is approximately planar (as in Object Stereo), we compute an enclosing 3D bounding box per object where the bounding box represents a proxy for the real 3D extent of the object. One of our 3D reconstructions and bounding boxes of extracted objects are shown in Fig. 6.17b. Note that this bounding box representation is more general than the “billboard” world of Object Stereo. For example, in Fig. 6.17f, the water kettle is not flat and hence Object Stereo oversegments the kettle using multiple flat objects (see left arrow in Fig. 6.17f). In contrast, our bounding box representation allows modeling the water kettle as a single object (see Figs. 6.17c and 6.17d). However, the main argument for using objects with 3D extents is that it enables modeling physical constraints [28], which is discussed in the following.

In particular, we model intersection and gravity constraints. The intersection constraint thereby reasons about occupancy in 3D space, that is, the spatial extents of 3D objects should not intersect each other. For example, in Fig. 6.17f (right arrow),

Object Stereo assigns the top and the bottom part of the can to the same object, while the middle part is assigned to a different object. This volume intersection is physically very unlikely, whereas our results in Figs. 6.17c and 6.17d are physically plausible. Our second physics-based constraint, that is, the gravity constraint, encodes the observation that objects usually do not float in the air, but stand on top of each other due to gravity. Analogously to the intersection constraint, this could not be realized using Object Stereo’s surface-based representation.

In the experimental results, we demonstrate that our approach is on par with the state-of-the-art in stereo matching. We also demonstrate that matching accuracy is improved if our physics-based constraints are enabled. A major focus lies on the object segmentation results. We show that our method can be applied to generate a large set of object proposals. This large set of object proposals has been the key to success in [19], which is the recent winner of the PASCAL object recognition challenge. After automatically ranking the proposals we show experimentally that our results are considerably closer to ground truth than state-of-the-art techniques which either use stereo or monocular images.

## 6.4 Summary

In this chapter, we have reviewed the state-of-the-art in stereo matching and highlighted our contributions in this field. We have followed the standard categorization of stereo methods and divided them between local and global algorithms.

Local algorithms operate on windows that are displaced in the right image to find the matching point of lowest color dissimilarity. The implicit assumption that all pixels within the support window have constant disparity is broken at disparity borders, which leads to edge fattening. We have discussed that adaptive weights [86] represent a remedy to the edge fattening problem and have presented our contributions in this domain. We have first presented our geodesic support weight algorithm [36]. A limitation of [36] and [86] is the relatively slow runtime. In this context, we have presented our cost filter approach [61] that allows high-quality adaptive support weight matching in real-time. We have then focused on the problem of reconstructing slanted surfaces where standard local methods show poor performance due to using fronto-parallel windows. We have presented our PatchMatch Stereo work [14] that overcomes this problem by using slanted support windows.

In the discussion of global methods, we have started by describing a standard stereo energy function and then briefly focused on the problem of optimizing this energy. We have discussed the following optimization techniques: dynamic programming (including our Simple Tree method [10]), graph-cuts and message passing. However, the real problem is not in the optimization, but in finding an energy function that represents a good model of the stereo problem. Hence, we have concentrated on improving the model in the remainder of this chapter.

We have discussed the data term of our energy and presented different match measures. In the context of data terms, we have presented our work [7], which in-

vestigates the usefulness of color information in stereo matching. We have then explained how to incorporate occlusion treatment into the data terms of global methods. In the discussion of smoothness terms, we have presented first and second order terms and stressed the importance of edge-sensitive and highly-connected smoothness terms. After a brief overview of segmentation-based methods, we have explained our combined stereo and matting method [12]. Finally, we have presented our other papers on improving the stereo model. This has included Surface Stereo [13], Object Stereo [15] and our physics-based approach [16].

**Acknowledgements** This work was supported in part by the Vienna Science and Technology Fund (WWTF) under project ICT08-019.

## References

1. Agarwal S, Snavely N, Simon I, Seitz S, Szeliski R (2009) Building Rome in a day. In: ICCV
2. Baker S, Szeliski R, Anandan P (1998) A layered approach to stereo reconstruction. In: CVPR, pp 434–441
3. Baker S, Scharstein D, Lewis J, Roth S, Black M, Szeliski R (2011) A database and evaluation methodology for optical flow. *Int J Comput Vis* 92(1):1–31
4. Barnes C, Shechtman E, Finkelstein A, Goldman D (2009) PatchMatch: a randomized correspondence algorithm for structural image editing. *ACM Trans Graph* 28(3):24 (SIGGRAPH Proc)
5. Birchfield S, Tomasi C (1998) A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Trans Pattern Anal Mach Intell* 20(4):401–406
6. Birchfield S, Tomasi C (1999) Depth discontinuities by pixel-to-pixel stereo. *Int J Comput Vis* 35(3):269–293
7. Bleyer M, Chambon S (2010) Does color really help in dense stereo matching? In: International symposium on 3D data processing, visualization and transmission (3DPVT)
8. Bleyer M, Gelautz M (2005) A layered stereo matching algorithm using image segmentation and global visibility constraints. *ISPRS J Photogramm Remote Sens* 59(3):128–150
9. Bleyer M, Gelautz M (2007) Graph-cut-based stereo matching using image segmentation with symmetrical treatment of occlusions. *Signal Process Image Commun* 22(2):127–143
10. Bleyer M, Gelautz M (2008) Simple but effective tree structures for dynamic programming-based stereo matching. In: VISAPP, vol 2, pp 415–422
11. Bleyer M, Chambon S, Poppe U, Gelautz M (2008) Evaluation of different methods for using colour information in global stereo matching. In: International archives of the photogrammetry, remote sensing and spatial information sciences, vol XXXVII, pp 415–422
12. Bleyer M, Gelautz M, Rother C, Rhemann C (2009) A stereo approach that handles the matting problem via image warping. In: CVPR, pp 501–508
13. Bleyer M, Rother C, Kohli P (2010) Surface stereo with soft segmentation. In: CVPR
14. Bleyer M, Rhemann C, Rother C (2011) PatchMatch stereo—stereo matching with slanted support windows. In: BMVC
15. Bleyer M, Rother C, Kohli P, Scharstein D, Sinha S (2011) Object stereo—joint stereo matching and object segmentation. In: CVPR
16. Bleyer M, Rhemann C, Rother C (2012) Extracting 3D scene-consistent object proposals and depth from stereo images. In: ECCV
17. Bobick A, Intille S (1999) Large occlusion stereo. *Int J Comput Vis* 33(3):181–200
18. Boykov Y, Veksler O, Zabih R (2001) Fast approximate energy minimization via graph cuts. *IEEE Trans Pattern Anal Mach Intell* 23(11):1222–1239

19. Carreira J, Li F, Sminchisescu C (2012) Object recognition by sequential figure-ground ranking. *Int J Comput Vis* 98(3):243–262
20. Deng Y, Yang Q, Lin X, Tang X (2005) A symmetric patch-based correspondence model for occlusion handling. In: *ICCV*, pp 542–567
21. Faugeras O, Hotz B, Mathieu H, Viéville T, Zhang Z, Fua P, Théron E, Moll L, Berry G, Vuillemin J, Bertin P, Proy C (1996) Real time correlation based stereo: algorithm implementations and applications. Technical report, RR-2013, INRIA
22. Felzenszwalb P, Huttenlocher D (2006) Efficient belief propagation for early vision. *Int J Comput Vis* 70(1):41–54
23. Fua PV (1991) Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities. In: *International joint conference on artificial intelligence*, pp 1292–1298
24. Fusielo A, Roberto V, Trucco E (1997) Efficient stereo with multiple windowing. In: *CVPR*, pp 858–863
25. Gallup D, Frahm J, Mordohai P, Yang Q, Pollefeys M (2007) Real-time plane-sweeping stereo with multiple sweeping directions. In: *CVPR*
26. Gehrig S, Franke U (2007) Improving sub-pixel accuracy for long range stereo. In: *ICCV VRML workshop*
27. Geiger A, Lenz P, Urtasun R (2012) Are we ready for autonomous driving? The KITTI vision benchmark suite. In: *CVPR*
28. Gupta A, Efros A, Hebert M (2010) Blocks world revisited: image understanding using qualitative geometry and mechanics. In: *ECCV*
29. Hartley R, Zisserman A (2003) Multiple view geometry in computer vision
30. Hasinoff S, Kang SB, Szeliski R (2006) Boundary matting for view synthesis. *Comput Vis Image Underst* 103(1):22–32
31. He K, Sun J, Tang X (2010) Guided image filtering. In: *ECCV*
32. Hirschmüller H (2005) Accurate and efficient stereo processing by semi-global matching and mutual information. In: *CVPR*, vol 2, pp 807–814
33. Hirschmüller H, Scharstein D (2009) Evaluation of stereo matching costs on images with radiometric differences. *IEEE Trans Pattern Anal Mach Intell* 31:1582–1599
34. Hirschmüller H, Innocent P, Garibaldi J (2002) Real-time correlation-based stereo vision with reduced border errors. *Int J Comput Vis* 47:229–246
35. Hong L, Chen G (2004) Segment-based stereo matching using graph cuts. In: *CVPR*, vol 1, pp 74–81
36. Hosni A, Bleyer M, Gelautz M, Rhemann C (2009) Local stereo matching using geodesic support weights. In: *ICIP*
37. Hosni A, Bleyer M, Gelautz M (2010) Near real-time stereo with adaptive support weight approaches. In: *3DPVT*
38. Hosni A, Rhemann C, Bleyer M, Gelautz M (2011) Temporally consistent disparity and optical flow via efficient spatio-temporal filtering. In: *PSIVT*
39. Hosni A, Rhemann C, Bleyer M, Rother C, Gelautz M (2013) Fast cost-volume filtering for visual correspondence and beyond. *IEEE Trans Pattern Anal Mach Intell* 35(2):504–511
40. Ishikawa H (2000) Global optimization using embedded graphs. PhD thesis, New York University
41. Jegelka S, Bilmes J (2011) Submodularity beyond submodular energies: coupling edges in graph cuts. In: *CVPR*
42. Ju M, Kang H (2009) Constant time stereo matching. In: *MVIP*, pp 13–17
43. Klaus A, Sormann M, Karner K (2006) Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In: *ICPR*, pp 15–18
44. Kohli P, Kumar M, Torr P (2007) P3 & beyond: solving energies with higher order cliques. In: *CVPR*
45. Kolmogorov V, Rother C (2007) Minimizing non-submodular functions with graph cuts—a review. *IEEE Trans Pattern Anal Mach Intell* 29(7):1274–1279

46. Kolmogorov V, Zabih R (2002) Computing visual correspondence with occlusions using graph cuts. In: ICCV, vol 2, pp 508–515
47. Kolmogorov V, Zabih R (2002) Multi-camera scene reconstruction via graph cuts. In: ECCV
48. Kolmogorov V, Zabih R (2004) What energy functions can be minimized via graph cuts? *IEEE Trans Pattern Anal Mach Intell* 26(2):147–159
49. Krähenbühl P, Koltun V (2011) Efficient inference in fully connected CRFs with gaussian edge potentials. In: Advances in neural information processing systems
50. Lempitsky V, Rother C, Blake A (2007) Logcut—efficient graph cut optimization for Markov random fields. In: ICCV
51. Li G, Zucker SW (2006) Surface geometric constraints for stereo in belief propagation. In: CVPR, pp 2355–2362
52. Lin M, Tomasi C (2003) Surfaces with occlusions from layered stereo. In: CVPR, pp 710–717
53. Mei X, Sun X, Zhou M, Jiao S, Wang H, Zhang X (2011) On building an accurate stereo matching system on graphics hardware. In: GPUVCV, pp 467–474
54. Meltzer T, Yanover TC, Weiss Y (2005) Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. In: ICCV, pp 428–435
55. Mühlmann K, Maier D, Hesser J, Männer R (2002) Calculating dense disparity maps from color stereo images, an efficient implementation. *Int J Comput Vis* 47(1):79–88
56. Newcombe R, Izadi S, Hilliges O, Molyneaux D, Kim D, Davison A, Kohli P, Shotton J, Hodges S, Fitzgibbon A (2011) KinectFusion: real-time dense surface mapping and tracking. In: ISMAR
57. Ogale AS, Aloimonos Y (2004) Stereo correspondence with slanted surfaces: critical implications of horizontal slant. In: CVPR, pp 568–573
58. Ohta Y, Kanade T (1985) Stereo by intra- and inter-scanline search. *IEEE Trans Pattern Anal Mach Intell* 7(2):139–154
59. Paris S, Durand F (2009) A fast approximation of the bilateral filter using a signal processing approach. *Int J Comput Vis* 81:24–52
60. Porikli F (2005) Integral histogram: a fast way to extract histograms in cartesian spaces. In: CVPR, vol 1, pp 829–836
61. Rhemann C, Hosni A, Bleyer M, Rother C, Gelautz M (2011) Fast cost-volume filtering for visual correspondence and beyond. In: CVPR
62. Richardt C, Orr D, Davies I, Criminisi A, Dodgson NA (2010) Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid. In: ECCV, vol 6313, pp 510–523
63. Rother C, Kolmogorov V, Blake A (2004) GrabCut: interactive foreground extraction using iterated graph cuts. *ACM Trans Graph* 23:309–314
64. Rother C, Kolmogorov V, Lempitsky V, Szummer M (2007) Optimizing binary MRFs via extended roof duality. In: CVPR
65. Rother C, Kohli P, Feng W, Jia J (2009) Minimizing sparse higher order energy functions of discrete variables. In: CVPR, pp 1382–1389
66. Roy S, Cox I (1998) A maximum-flow formulation of the n-camera stereo correspondence problem. In: ICCV, pp 492–499
67. Scharstein D, Szeliski R (2002) A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int J Comput Vis* 47(1/2/3):7–42. <http://vision.middlebury.edu/stereo/>
68. 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: CVPR
69. Smith B, Zhang L, Jin H (2009) Stereo matching with nonparametric smoothness priors in feature space. In: CVPR, pp 485–492
70. Snavely N, Seitz S, Szeliski R (2006) Photo tourism: exploring photo collections in 3D. *ACM Trans Graph* 25:835–846 (SIGGRAPH Proc)
71. Sun J, Zheng NN, Shum HY (2003) Stereo matching using belief propagation. *IEEE Trans Pattern Anal Mach Intell* 25(7):787–800
72. Sun J, Li Y, Kang SB, Shum HY (2005) Symmetric stereo matching for occlusion handling. In: CVPR, vol 25, pp 399–406

73. Szeliski R, Golland P (1998) Stereo matching with transparency and matting. In: ICCV, pp 517–525
74. Szeliski R, Zabih R, Scharstein D, Veksler O, Kolmogorov V, Agarwala A, Tappen M, Rother C (2006) A comparative study of energy minimization methods for Markov random fields. In: ECCV, vol 2, pp 19–26
75. Taguchi Y, Wilburn B, Zitnick L (2008) Stereo reconstruction with mixed pixels using adaptive over-segmentation. In: CVPR, pp 1–8
76. Tao H, Sawhney H, Kumar R (2001) A global matching framework for stereo computation. In: ICCV, pp 532–539
77. Tappen MF, Freeman WT (2003) Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In: ICCV, vol 2, pp 900–906
78. Veksler O (2002) Stereo correspondence with compact windows via minimum ratio cycle. IEEE Trans Pattern Anal Mach Intell 24(12):1654–1660
79. Veksler O (2005) Stereo correspondence by dynamic programming on a tree. In: CVPR, pp 384–390
80. Wainwright M, Jaakkola T, Willsky A (2003) Tree reweighted belief propagation algorithms and approximate ML estimation by pseudo-moment matching. In: AISTATS
81. Woodford O, Torr P, Reid I, Fitzgibbon A (2008) Global stereo reconstruction under second order smoothness priors. In: CVPR
82. Xiong W, Jia J (2007) Stereo matching on objects with fractional boundary. In: CVPR, pp 1–8
83. Yang Q, Wang L, Yang R, Wang S, Liao M, Nister D (2006) Real-time global stereo matching using hierarchical belief propagation. In: BMVC
84. Yang Q, Yang R, Davis J, Nister D (2007) Spatial-depth super resolution for range images. In: CVPR
85. Yang Q, Wang L, Yang R, Stewenius H, Nister D (2009) Stereo matching with color-weighted correlation, hierarchical belief propagation and occlusion handling. IEEE Trans Pattern Anal Mach Intell 31(3):492–504
86. Yoon KJ, Kweon IS (2005) Locally adaptive support-weight approach for visual correspondence search. In: CVPR
87. Zhang Y, Gong M, Yang Y (2008) Local stereo matching with 3D adaptive cost aggregation for slanted surface modeling and sub-pixel accuracy. In: ICPR
88. Zhang K, Lu J, Lafruit G (2009) Cross-based local stereo matching using orthogonal integral images. IEEE Trans Circuits Syst Video Technol 19:1073–1079
89. Zhang K, Lafruit G, Lauwereins R, Gool L (2010) Joint integral histograms and its application in stereo matching. In: ICIP, pp 817–820
90. Zitnick L, Kang S, Uyttendaele M, Winder S, Szeliski R (2004) High-quality video view interpolation using a layered representation. ACM Trans Graph 23(3):600–608

# Chapter 7

## Visual Localization for Micro Aerial Vehicles in Urban Outdoor Environments

Andreas Wendel and Horst Bischof

**Abstract** Accurate localization of a micro aerial vehicle (MAV) with respect to a scene is important for a wide range of applications, in particular autonomous navigation, surveillance, and inspection. In this context, visual localization in urban outdoor environments is gaining importance as common methods such as GPS positioning are often not accurate enough or even fail. We present recent approaches and results for robust 3D reconstruction of suitable visual landmarks, for the alignment in a world coordinate system, and for fast, high-accuracy monocular localization. We introduce a scalable representation of the prior knowledge about the scene and demonstrate how in-flight information can be integrated to facilitate long-term operation. Our method outperforms a state-of-the-art visual SLAM approach and achieves localization accuracies comparable to differential GPS.

### 7.1 Introduction

Micro aerial vehicles (MAVs) are gaining importance as image acquisition tools in photogrammetry, but also for industrial applications such as power-line inspection or building reconstruction [16, 31]. Areas of interest for close-up aerial images are often in urban environments, close to the ground, close to buildings, and possibly close to human beings. Therefore, we see the need for intelligent autonomy in terms of systems which can navigate accurately and safely in an urban outdoor environment.

The usage of unmanned aerial vehicles with an integrated global positioning system (GPS) for photogrammetric applications started as early as 1995 [8]. These systems had the dimensions of a small aircraft to be able to carry the camera systems of the time, which allowed them to fly only above an altitude of about 300 meters. With the advent of MAVs in form of very small and light aircraft or quad-rotor helicopters it became possible to fly much closer to buildings. While this permits to

---

A. Wendel (✉) · H. Bischof

Institute for Computer Graphics and Vision, Graz University of Technology, Graz, Austria  
e-mail: [wendel@icg.tugraz.at](mailto:wendel@icg.tugraz.at)

H. Bischof  
e-mail: [bischof@icg.tugraz.at](mailto:bischof@icg.tugraz.at)

obtain high-quality, close-up images of scenes, it also poses a problem to localization. GPS can be replaced by much more accurate differential GPS, if high costs are acceptable; still, both technologies are not able to detect obstacles and are prone to shadowing effects caused by neighboring buildings.

While GPS is still the most widely used sensor for outdoor localization, the trend goes towards visual localization. Zufferey et al. [77] combined GPS-based path following with vision-based collision avoidance. Their bio-inspired approach allowed them to fly close to the ground, approx. 9 m above the terrain, while effectively detecting and avoiding trees during flight. Hrabar and Sukhatme [22] used a similar approach based on optical flow and differential GPS to navigate an autonomous helicopter safely through a simulated urban canyon. Both approaches focus on responding to obstacles using vision, but not on accurate localization of the vehicle. Other approaches focus on marker-based visual localization [51] or visual outside-in tracking [40] which is not suitable for outdoor usage.

Visual SLAM (simultaneous localization and mapping) dominates the field of visual localization in aerial robotics. Bloesch et al. [4] first used Parallel Tracking and Mapping (PTAM) [28] for monocular MAV navigation in unstructured indoor environments. Recent work of Achtelik et al. [1] showed how the same approach can be used outdoors by fusing it with additional sensors available to an MAV. They demonstrate that visual input can be successfully used for the control of an MAV despite the insufficient pose update rates on constrained hardware. However, their localization framework requires a nadir view of the camera and a textured ground plane. This significantly limits the possible applications, as for instance paved roads or grass cause problems.

In contrast, we rely on a prior reconstruction in form of a visual landmark which allows us to localize the MAV with respect to this map. Such a method has several key advantages over a traditional SLAM approach that solves the mapping and localization problem simultaneously. First, given a geo-registered visual 3D model of the environment, localization can be done in a metric scale. This would not be possible for a monocular SLAM approach without considering additional input modalities. Moreover, once a detailed visual model of an environment is available, the localization approach is fully scalable and allows direct data fusion with other sensors such as GPS or inertial measurement units (IMU). Obstacles or points of interest defined in world coordinates can be easily integrated in the map. Finally, once a good localization is available, it is much easier to fill the gaps of the rough model with dynamic visual data.

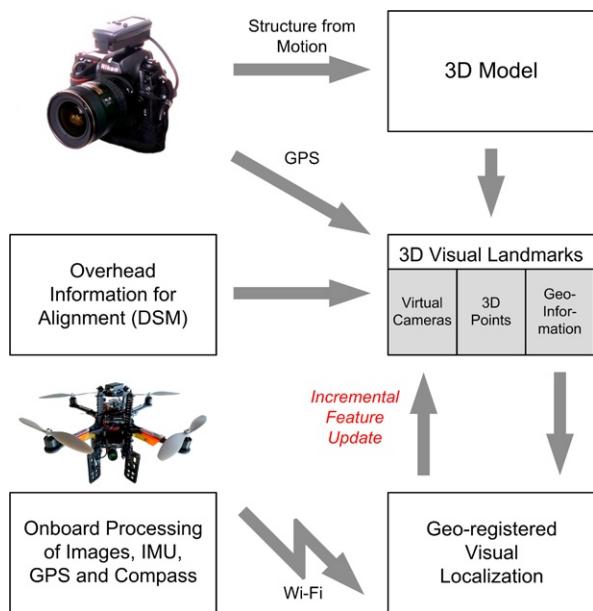
In this work, we present a framework (Fig. 7.1, Fig. 7.2) which aims for long-term visual localization. We create metric, geo-referenced 3D models [66, 68] by reconstructing and aligning an unordered set of images. Monocular localization in such a landmark can then be achieved based on the concept of virtual views and incremental feature updates [67, 69], with an accuracy comparable to differential GPS in a global coordinate system, and at a speed of 4 frames per second.

We first give an overview of image-based techniques to model an urban outdoor environment in Sect. 7.2. The resulting 3D reconstructions are then aligned to a geo-referenced digital surface model as presented in Sect. 7.3. In Sect. 7.4, we introduce

**Fig. 7.1** We present an approach to visual localization which exploits prior knowledge in form of pictures taken at ground-level to enhance the accuracy and reliability of position and attitude estimation. Our algorithm (*red trajectory*) is neither prone to drift nor bias and therefore outperforms pure SLAM approaches (*blue trajectory*). ©2011 IEEE. Reprinted, with permission, from [67]



**Fig. 7.2** Our framework for visual localization is based on *Visual Landmarks*. 3D models of the environment are geo-referenced and a set of virtual views is generated for the area. The proposed system is able to provide geo-localizations to an MAV with about 4 frames per second in an outdoor environment. This is not fast enough to navigate the vehicle based on vision only, but a bias-free pose estimate is crucial to correct the drift of other sensors. ©2012 IEEE. Reprinted, with permission, from [69]



our method for fast and drift-free localization. Our framework is finally evaluated in Sect. 7.5, and Sect. 7.6 concludes the chapter.

## 7.2 Scene Reconstruction

*Visual landmarks* are the fundamental concept of our work. We assume that significant parts of the scene such as buildings or trees do not alter their geometry for days, probably even years, and can therefore serve as natural landmarks. To cope with appearance changes we rely on imagery taken at different times of the day and in different seasons, which should be directly integrated in the 3D reconstruction

if possible, or combined by a 3D alignment of reconstructions as discussed in the next section. Imagery for our approach can be collected from many sources such as Microsoft Bing Streetside or Google Street View, or by manually taking pictures of the area with a consumer camera. Additionally, one can exploit novel image acquisition tools such as micro aerial vehicles in form of quad- or octo-rotor helicopters which allow to take pictures from completely different points of view with ground sampling distances below 1 cm.

In the following, we present our 3D reconstruction pipeline and highlight important algorithmic details. We also present two approaches to densify the resulting sparse point cloud, which is an important contribution to subsequent proper alignment and occlusion handling during localization.

### **7.2.1 Related Work**

Structure from Motion (SfM) deals with the problem of estimating the 3D structure of a scene while simultaneously solving for camera orientations using only 2D image measurements. It is a fundamental and well studied problem in both computer vision and photogrammetry [19]. In contrast to approaches which directly estimate a dense depth map for every acquired frame, for instance, by using stereo imaging [42] or active sensing [45], SfM systems extract depth information from a set of images which are acquired one after another. SfM has reached maturity over the past years, and meanwhile fully automatic pipelines for ordered [49] and unordered [59, 60] image collections exist. Processing is typically done offline, which allows to quickly capture images in the field and then do the processing in the lab with powerful hardware.

The recent success of Structure from Motion techniques can be traced back to significant advances in wide-baseline feature matching [39] and multiple-view geometry [46, 64]. Also, the advent of much more powerful computational hardware, and especially general-purpose graphic processing units (GPGPU) [12], supported the applicability of 3D reconstruction from image sequences. Still, exhaustive pairwise feature matching to find correspondences in the individual views is the most time demanding task in today's reconstruction pipelines. To circumvent this issue, Agarwal et al. [2] introduced a two-stage matching approach which is now considered best practice. First, every image is described by a single feature vector and coarse matches between images are detected. This process is based on image retrieval and bag-of-words concepts [58]. Then, only those image pairs which achieve a high similarity score are used for detailed feature matching. Our pipeline incorporates all of these ideas to create robust and accurate 3D models.

### **7.2.2 Sparse Structure-from-Motion Modeling**

For sparse 3D model reconstruction, we rely on a Structure from Motion (SfM) approach that is able to reconstruct a scene from unorganized image sets. Our solution

to the 3D reconstruction problem is based on the work of [25] and [67]. It is widely applicable since no prior knowledge about the scene is necessary (i.e., no sequential ordering of the input images has to be provided) and can therefore be used to create 3D models from terrestrial as well as aerial imagery. To accelerate the computations, we take advantage of graphic processing units (GPUs) for efficient parallelized computing. A typical SfM framework consists of three processing steps, namely feature extraction, matching, and geometry estimation. These steps are summarized in the following paragraphs.

### 7.2.2.1 Establishing the Epipolar Graph

First, salient features are extracted from each frame. Our method utilizes the very effective combination of DoG keypoint detector and SIFT descriptor [39] which achieves excellent repeatability performance for wide baseline image matching [43]. In particular we rely on the publicly available SiftGPU [72] software.

We then match keypoint descriptors between each pair of images. A variety of approaches has been proposed to speedup nearest neighbor matching in high-dimensional spaces (like the 128-dimensional SIFT descriptor space). Among the most promising methods are randomized kd-trees [57] with priority search and hierarchical k-means trees [14]. These algorithms are in general designed to run on a single CPU and are known to provide speedups of about one or two orders of magnitude over linear search, but the speedup comes with the cost of a potential loss in accuracy [44]. On the other hand, given that the number of features is limited to some thousands, nearest neighbor search, implemented as a dense matrix multiplication on recent graphics hardware, can achieve an equivalent speedup but delivers the exact solution. Hence, we employ a GPU-accelerated feature matching approach based on the CUBLAS library with subsequent instructions to apply the distance ratio test and to report the established correspondences.

After matching relevant images to each query view, geometric verification based on the Five-Point algorithm [47] is performed. Since matches that arise from descriptor comparisons are often highly contaminated by outliers, we employ the random sample consensus (RANSAC) [11] algorithm for robust estimation. In its basic implementation, RANSAC acts as a hypothesize-and-verify approach. From a minimal set of samples, several hypotheses are generated and the consensus of the model is evaluated on the full set of observations. The RANSAC termination confidence,

$$p = 1 - \exp(N \log(1 - (1 - \varepsilon)^s)) \quad (7.1)$$

is used to decide whether two images satisfy the epipolar geometry. Here,  $N$  is the total number of evaluated models,  $w = 1 - \varepsilon$  the probability that any selected data point is an inlier, and  $s = 5$  is the cardinality of the sample point set used to compute a minimal model. We require  $p > 0.999$  in order to accept an epipolar geometric relation. In our experiments, we use up to  $N = 2000$  models which corresponds to a maximal outlier fraction of  $\varepsilon = 0.67$ .

The matching output is a graph structure denoted as epipolar graph, that consists of the set of vertices  $\mathcal{V} = \{I_1 \dots I_N\}$  corresponding to the images and a set of edges  $\mathcal{E} = \{e_{ij} | i, j \in \mathcal{V}\}$  that are pairwise reconstructions, that is, relative orientations between view  $i$  and  $j$ ,  $e_{ij} = \langle P_i, P_j \rangle$ ,

$$P_0 = K_0[I, 0] \quad \text{and} \quad P_1 = K_1[R, t] \quad (7.2)$$

and a set of triangulated points with respective image measurements. Since the cameras are calibrated, a linear triangulation method [18] is sufficient to accurately estimate the 3D point location. This procedure is followed by a pruning step that discards ill-conditioned 3D points (points that do not satisfy the cheirality criterion) and points that have a high depth uncertainty (points where the roundness of the confidence ellipsoid [3] is larger than a given threshold).

### 7.2.2.2 Structure Initialization

Our SfM method follows an incremental approach [59] based on the epipolar graph. In order to reconstruct a consistent 3D model, a robust and reliable start configuration is required. When the initial structure is prone to errors, a subsequent iterative optimization procedure will eventually end up in a wrong local minimum, hence good initialization is critical. As proposed in [30], we initialize the geometry in the most connected parts of the graph, therefore the vertex  $I^*$  with highest degree, that is the node having the largest number of edges, is determined. We start from the vertex  $I^*$  and determine all connected neighbors. Next, all pair-wise measurements are linked into point tracks and the global scale factor of the initial structure is estimated. Then, bundle adjustment [64] is used to optimize camera orientations  $P_i$  and 3D points  $\mathbf{X}_j$  by minimizing the reprojection error,

$$\mathcal{C}(P, \mathbf{X}) = \sum_i \sum_j v_{ij} d(P_i \mathbf{X}_j, \mathbf{x}_{ij})^2 \quad (7.3)$$

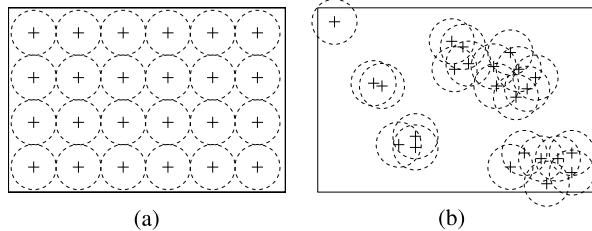
where  $\mathbf{x}_{ij}$  are 2D point measurements of observed 3D points and  $v_{ij}$  is an indicator variable that is 1 if the point  $\mathbf{X}_j$  is visible in camera  $P_i$  and 0 otherwise. To limit the impact of outliers, we use a robust Huber M-Estimator [23] to compute the costs  $d$ .

Given the initial optimized structure, each 3D point is back-projected and searched for in every image. We utilize a 2D kd-tree for efficient search and restrict the search radius to a constant factor  $r_t$ . Again, given the new measurements, bundle adjustment is used to optimize 3D points and camera parameters. This method ensures strong connections within the current reconstruction.

### 7.2.2.3 Incremental Reconstruction

Next, for every image  $I$  that is not reconstructed and has a potential overlap to the current 3D scene (estimated from the epipolar graph), 2D-to-3D correspondences are established. A three-point pose estimation algorithm [17, 32] inside a RANSAC loop is used to insert the position of a new image. When a pose can be determined

**Fig. 7.3** Effective inlier measure. Coverage of (a) uniformly and (b) non-uniformly distributed image measurements. ©2011 IEEE. Reprinted, with permission, from [67]



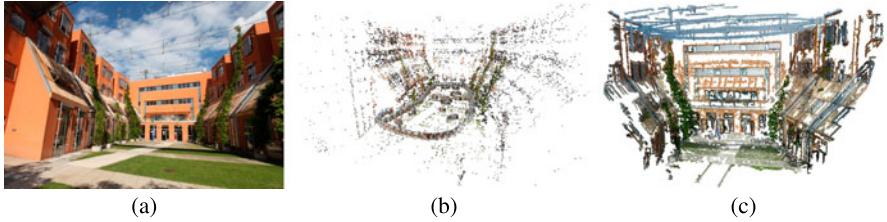
(i.e., a sufficient inlier confidence is achieved), the structure is updated with the new camera and all measurements visible therein. A subsequent procedure expands the current 3D structure by triangulation of new correspondences. We follow the approach of Snavely et al. [59] and use a priority queue to guide the insertion order. Our insertion order is based on a saliency measure that favors early insertion of images that have a strong overlap with the given 3D structure. Rather than using the raw number of potential 2D-to-3D matches, we compute an effective matching score that further takes the spatial match distribution into account. This idea is depicted in Fig. 7.3. While the number of features is equal in (a) and (b), the uniform spatial distribution of point features in (a) can be regarded as more reliable than the one shown in (b). Hence, we weight the raw number of features by an estimate for the covered image fraction yielding the effective inlier count.

Whenever a number of  $N$  images is added (we use  $N = 10$ ), bundle adjustment is used to simultaneously optimize the structure and all camera poses. The iterative view insertion procedure is repeated until the priority queue is empty. The sparse reconstruction result can be seen in Fig. 7.4(b).

### 7.2.3 Structure Refinement

Structure from Motion yields camera orientations and a sparse set of triangulated points. These entities are sufficient for a localization task, however for georeferencing and occlusion handling a denser model is required. We thus apply two stages of refinement to our models: First, we densify the point cloud using a patch-based multiview stereo approach, and then we generate a surface mesh using Delaunay triangulation and free space carving.

The publicly available patch-based multiview stereo (PMVS) software of Furukawa and Ponce [15] is one of the most popular approaches to automatically reconstruct a scene when the camera positions are known. One of the main benefits is that this approach densifies a point cloud directly in 3D space rather than estimating the depth of every pixel which would introduce noise in untextured regions. PMVS initially estimates the position and orientation of small rectangular patches on the surface of the object based on sparse feature matching. Then, an expansion procedure iteratively adds patches in the neighborhood of already known regions. The approach successfully copes with outliers and thus is very robust, but the computational effort is also considerable. Since our structure from motion pipeline de-



**Fig. 7.4** Atrium scene reconstruction from 157 views. (a) Original view of the scene. (b) Sparse model and source cameras obtained by structure from motion reconstruction. (c) Semi-dense point model obtained by refining the sparse model with PMVS [15]. ©2011 IEEE. Reprinted, with permission, from [67]

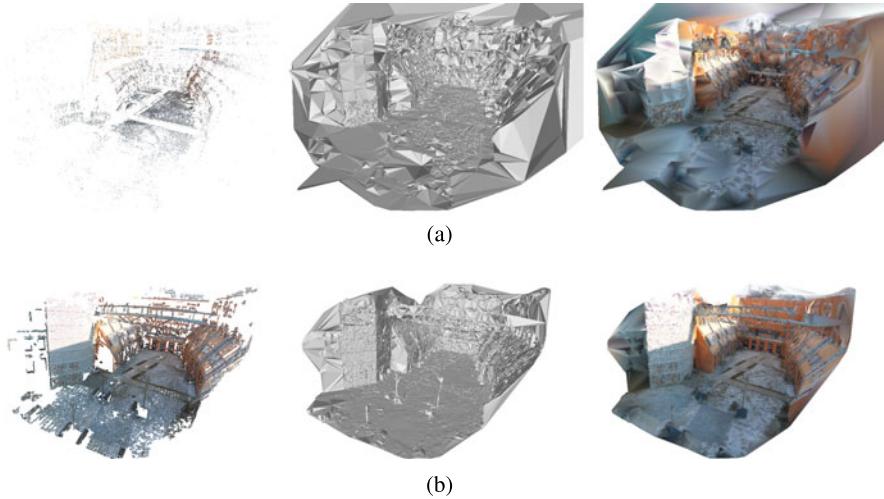
livers sub-pixel accurate 3D camera orientations, PMVS can be run at full image resolution. However, even PMVS requires at least weakly textured surfaces for densification, thus a constant ground sampling distance cannot be guaranteed. The PMVS dense reconstruction result of the captured atrium scene is depicted in Fig. 7.4(c).

An alternative to point cloud densification, or as in our case a further refinement step, is to create watertight 3D models by performing a 3D Delaunay triangulation [5] followed by probabilistic space carving based on visibility information. Specifically, we employ the approach of Labatut et al. [34] who minimize an energy consisting of two terms to model the surface of objects: First, the visibility-based energy term votes for tetrahedra as being part of the object or part of free space, and it penalizes the number of conflicts of the line of sight with triangles crossing the surface. The second term takes care of the smoothness of the surface. It geometrically constrains the shape of a tetrahedra by penalizing the surface area, and by constraining the size of the circumsphere of the two adjacent tetrahedra of a triangle. This removes for example tiny elongated facets which are not desired in the resulting mesh. Finally, this energy functional is optimized via graph cuts [6]. The approach is very robust against outliers, and it preserves edges and fine structures while the result is at the same time free of self-intersections. However, the accuracy of the surface estimation depends on the number of 3D points in the mesh, as triangles between points can only linearly interpolate the surface. Therefore, the densification using PMVS as mentioned above is very beneficial to increase the level of detail in the model. The meshing results given a sparse point cloud versus a densified point cloud are compared in Fig. 7.5.

#### 7.2.4 Discussion

We have described a state-of-the-art Structure from Motion pipeline and two additional steps for densification and meshing of the resulting point cloud. This process delivers the required input for subsequent geo-referencing and localization tasks.

While we carefully extract as much information of the input data as possible, the reconstruction quality and completeness heavily depends on the acquisition strategy

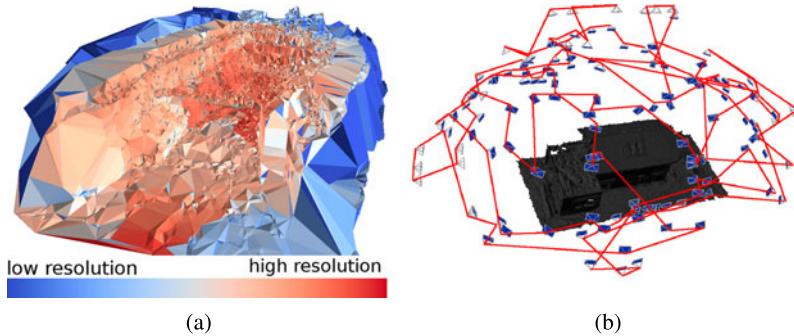


**Fig. 7.5** Meshing results based on the approach of [34]. From left to right, the point cloud, the surface mesh, and a textured mesh are visualized, based on (a) the SfM point cloud (36,948 points), and (b) the densified point cloud (318,540 points). The densification procedure clearly supports the extraction of the true surface and leads to visually appealing results

and can typically only be assessed after the offline processing step has finished; this is the major drawback of current systems. Therefore, we have recently worked on assessing and refining the acquisition strategies for Structure from Motion [20, 21]. We have come up with several requirements which have to be fulfilled: First, the viewing angle between two images must not be too large to allow feature matching, and the viewing cones need to overlap. Second, the images have to be textured, but the texture should not be too repetitive and the illumination must not change too much between images. Finally, the entire scene needs to be covered with sufficient overlap. For a typical user it is impossible to estimate if the acquired images fulfill all demands, so Hoppe et al. [20] have come up with an SfM system which meshes and visualizes the overlap and resolution of the model in real-time (Fig. 7.6(a)). When a prior model of the same scene is available, for instance for an evolving construction site or when the geometry can be estimated using geographic information systems (GIS), a sophisticated approach for automated view planning [21] has been proposed (Fig. 7.6(b)). Finally, our recent system for dense reconstruction on-the-fly [70] can also provide a live preview of the scene geometry and thus facilitates user interaction. We expect that future research in SfM techniques will increasingly incorporate the important aspect of acquisition strategies.

### 7.3 Alignment in a Global Coordinate System

Alignment of visual maps in a global coordinate system is important for several reasons: First, it facilitates the incorporation of additional sensors at flight time and for



**Fig. 7.6** Optimization of the acquisition strategy. (a) The user can get live feedback during the acquisition process by visualizing pose estimation failures as well as overlap and resolution per triangle [20]. (b) Alternatively, automatic view planning algorithms [21] can be employed. Blue acquisition positions and orientations are connected by the optimal flight path for an MAV

evaluation, as coordinates obtained by visual localization are directly comparable to IMU and GPS readings. This also allows an MAV to switch seamlessly between GPS waypoint mode and visual localization, which is very important when connecting different visual landmarks, and it allows to represent point clouds with metric scaling. Second, proper alignment is beneficial to applications where the model should be set into context, for instance in industrial applications such as construction site monitoring [31], or whenever further algorithmic steps depend on it as in automatic view planning [21, 56].

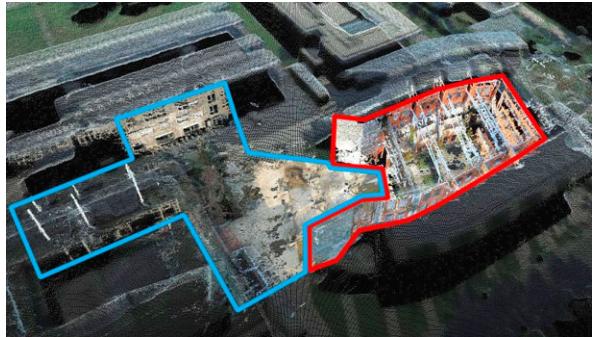
However, two major challenges have to be tackled. On the one hand, georeferencing of 3D reconstructions is often difficult because of shadowed GPS signals in urban areas, so accurate alignment purely based on GPS information is not possible. On the other hand, flight space might be restricted in urban areas, which leads to missing views for accurate 3D reconstruction and causes fracturing of large models. This could also happen due to vegetation or simply a change of illumination during image acquisition. For the resulting fractured models, no visual feature correspondences—or at least not enough for fusing the models—can be established.

In this section, we therefore address the automatic alignment of such partial Structure from Motion (SfM) reconstructions to an overhead digital surface model (DSM) [66, 68] as depicted in Fig. 7.7. Our approach is to get an initial rough estimate of the model’s pose using noisy GPS locations delivered by the consumer camera, and subsequently refine the pose based on a 2D correlation of the DSM height map with a projected model height map.

### 7.3.1 Related Work

The problem of aligning 2D images or 3D models to a 3D structure is well studied, especially in the context of large-scale city modeling. Frueh and Zakhor [13] present

**Fig. 7.7** Automatic fusion of two semi-dense 3D point clouds with an overhead, textured Digital Surface Model (DSM). The atrium model (red) has been reconstructed from ground-level, whereas the images for the campus model (blue) originate from a micro aerial vehicle. ©2011 IEEE. Reprinted, with permission, from [66]



an algorithm to fuse close-range facade models acquired at ground level with a far-range DSM recorded by a plane. The models are created using both ground-based and airborne laser scanners, as well as digital cameras for texturing. Their approach is based on registering the edges of the DSM image to the horizontal scans of a ground model using Monte-Carlo-Localization. Similarly, Strecha et al. [62] register facades segmented from a 3D point cloud to building footprints. Their approach combines various visual and geographical cues in a generative model, which allows robust treatment of outliers. However, both approaches are focused on large-scale city models with flat facades to both sides, resulting in fairly clean edges. In contrast, our approach takes the height over ground into account and therefore even benefits from complex structures.

A popular approach to aligning and fusing SfM point clouds is to use random sample consensus (RANSAC)-based geometric verification [11]. A typical issue is the estimation of a reasonable inlier threshold, however this has been resolved in recent work [50]. Still, such an approach is not feasible for our purpose as on the one hand feature correspondences cannot be established and the algorithm would have to solve a huge combinatoric problem. On the other hand, we want to align data with significant variations of the ground sampling distance which would not be possible either.

Another well-known method of aligning two point clouds is the Iterative Closest Points (ICP) algorithm [74]. ICP estimates a transformation to minimize the overall distance between points by iteratively assigning closest points as correspondences and solving for the best rigid transform. While ICP is mainly used for registering 3D laser scans, Zhao et al. [75] use it to align dense motion stereo from videos to laser scan data. However, 3D ICP can take very long and suffers from getting stuck in local minima due to its typically small convergence radius. In other words, a very good initialization is necessary for ICP to converge. However, using a point-to-plane variant [38] the ICP method can still be exploited on top of our method to improve the results.

Kaminsky et al. [27] use 2D ICP to compute the optimal alignment of a sparse SfM point cloud to an overhead image using an objective function that matches 3D points to image edges. Additionally, the objective function contains free space constraints which avoid an alignment to extraneous edges in the overhead image.

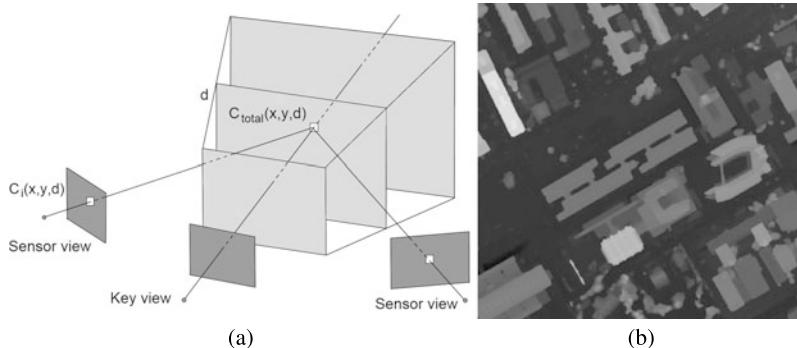
While their approach is suitable to align many 3D models obtained from ground level, it has problems with points on the ground and would therefore fail to align the models acquired using our micro aerial vehicle. Typical models presented in their paper show vertical walls which can be easily projected to the ground and fitted to respective edges. However, when many extraneous edges are visible in the overhead image their method fails. In contrast, given a sufficient point density in the reconstruction, our approach is less prone to errors caused by objects on the ground, it implicitly follows a free-space constraint and it works with models covering a small area.

### 7.3.2 Surface Model Reconstruction

A digital surface model (DSM) is a 2.5D height representation of the Earth's surface and all objects on it as seen from an orthogonal aerial viewpoint. In contrast to a full 3D model, just the maximum height over ground is considered, rather than looking at all structures that might be below. In other words, for instance the space underneath a bridge is filled instead of empty. The DSM can thus be stored in form of a 2D grayscale image with bright values reflecting a large height over ground.

The creation of a DSM typically requires a set of aerial images acquired by a plane or a micro aerial vehicle. After recovering the sparse geometry and camera orientations using Structure from Motion techniques, a dense 3D reconstruction is generated. Our method of choice is the multi-view plane sweep algorithm [7]. In short, the scene is traversed by planes parallel to the key view as shown in Fig. 7.8(a). For each discrete depth  $d$ , the sensor images are projected onto the plane and the similarity  $C_i(x, y, d)$  between pixels of the key view and the sensor view  $i$  are calculated as the zero-mean normalized cross correlation (ZNCC). Once the cost volume  $C_{\text{total}}$  is filled with the accumulated matching scores, a variational optimization problem in 3D voxel space as proposed by Irschara et al. [26] is solved to recover the final depth value for every pixel. The resulting depth maps are smoothed but edges are preserved, as can be seen in Fig. 7.8(b). In contrast to close-up scenes as discussed in the previous section, this approach works very well here because the distance between individual cameras is small compared to the distance between the cameras and the surface.

High-quality aerial photographs or digital surface models may not be accessible for everyone, and they may not be available for some locations on earth. However, web-based map providers such as OpenStreetMap provide fairly accurate information not only regarding roads and directions, but also concerning buildings and vegetation. We have shown in [52, 66] that it is straight-forward to estimate a rough DSM by assigning a single, estimated height to all buildings. While the resulting building heights are obviously not correct and thus cause errors, the estimate is in our experience good enough to align a 3D model using the proposed approach, with a better accuracy than the raw GPS alignment.



**Fig. 7.8** Surface Model Reconstruction. (a) Given the camera orientations as a result of the SfM reconstruction, the multi-view plane sweep approach [7] is used to generate dense 3D reconstructions. (b) Combined with a variational refinement [26] accurate digital surface models are created

### 7.3.3 Alignment to an Overhead Surface Model

Our alignment pipeline is based on the registration of 2D depth maps showing the scene in a nadir view, rather than 3D point clouds. We can thus handle geometric configurations where ICP fails due to unknown correspondences or local minima, as well as significant differences in appearance because there is no need to establish sparse feature correspondences. We follow a three step approach: First, we convert all GPS and pixel coordinates into a local, metric coordinate system. Second, we roughly align the acquired 3D point cloud to the DSM by exploiting the available GPS information. Finally, we search for the best height map alignment between the DSM and the semi-dense model point cloud using normalized cross-correlation.

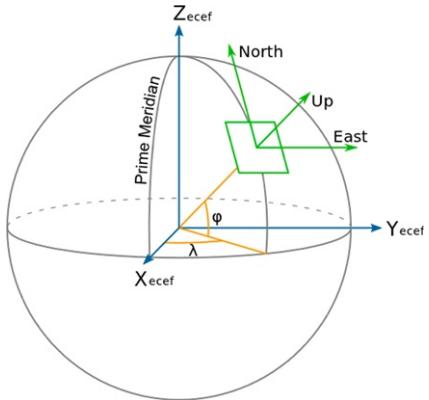
#### 7.3.3.1 Conversion to Local ECEF Coordinates

Geo-referenced model coordinates are represented in a local Earth-centered, Earth-fixed (local ECEF) coordinate system. While the global ECEF coordinate system has its origin at the center of the Earth, with the  $x$  axis passing through the equator at the prime meridian and the  $z$  axis passing through the north pole, local ECEF employs a tangent plane to the Earth's surface at a reference point (Fig. 7.9). By definition, the  $x$  axis heads East, the  $y$  axis North, and the  $z$  axis up into the sky, thus it is also called *ENU* coordinate system. While the individual reference point needs to be known using this representation, the benefits include a more intuitive Cartesian system which allows to obtain metric measurements in the transformed point cloud, and a better numerical stability of the transformations. We choose the reference point to be the arithmetic center of the DSM's known corner points.

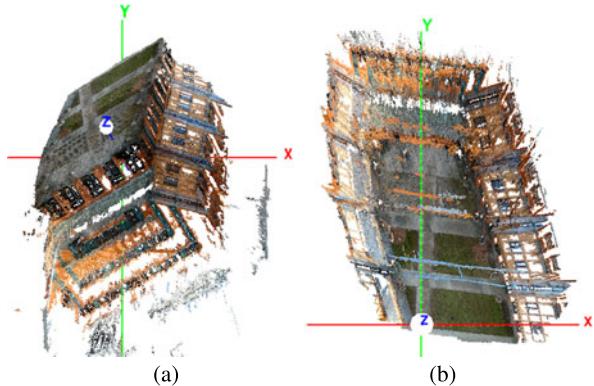
#### 7.3.3.2 Rough Alignment Using GPS Information

We exploit the GPS information available for every camera in our 3D models for rough alignment to the DSM. If the coordinates measured by the GPS system were

**Fig. 7.9** Relationship between GPS coordinates  $(\lambda, \phi)$  and the Cartesian local ECEF coordinate system (east, north, up). A tangent plane to the Earth's surface at a reference point is established for a more intuitive and numerically stable representation

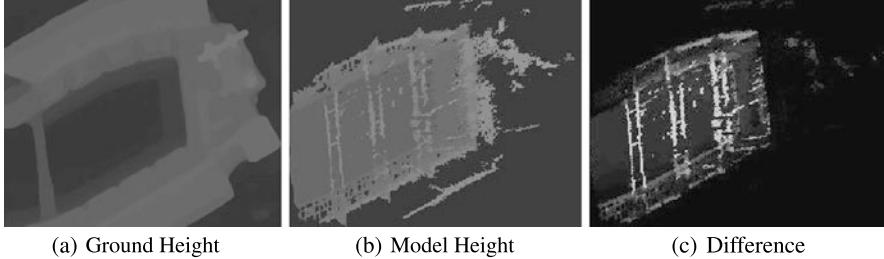


**Fig. 7.10** Rotation for ground plane alignment based on the method of [63].  
**(a)** SfM model in the coordinate system of the first model camera.  
**(b)** SfM model after performing the estimated rotation. ©2011 IEEE. Reprinted, with permission, from [66]



correct, it would be straightforward to estimate a 3D similarity transform. However, longitude and latitude are noisy, and especially the altitude estimate can be very inaccurate.

Therefore, our approach is to solve for a 2D similarity transform between the camera positions  $(x_{\text{sfm},i}, y_{\text{sfm},i})$  in the SfM model and the GPS coordinates  $(x_{\text{gps},i}, y_{\text{gps},i})$ , both in local ECEF coordinates. Neglecting the altitude is just possible when the ground planes are aligned. However, SfM pipelines typically store resulting models in the coordinate system of the first camera. As a result, the axes of partial reconstructions do not align and have to be rotated to a common ground plane. We employ a reasonable assumption to approximate this plane, namely that the horizontal axis in every image coordinate system is approximately parallel to the ground plane, which is the case when taking upright photographs from the ground but also when taking nadir and oblique pictures on a micro aerial vehicle. The approach of Szeliski [63] can then be used to compute the ground plane normal and the corresponding rotation, as visualized in Fig. 7.10. Finally, a robust 2D similarity transformation can be found using RANSAC [11].



**Fig. 7.11** Precise Alignment using Correlation. The normalized height values  $G$  of the ground **(a)** are compared to those of the projected SFM model  $M_t$  **(b)**, with undefined points marked in black. **(c)** Differences occur not only when a building height does not fit, but also when the ground height does not fit. ©2011 IEEE. Reprinted, with permission, from [66]

### 7.3.3.3 Precise Alignment Using Correlation

Given the rough alignment, we use correlation for precise alignment. We project the semi-dense 3D point cloud into the pixel grid of the DSM image, storing only the maximum height value per pixel. Pixel clusters with a radius  $r \leq 2px$  are removed using morphological operations to get rid of reconstruction outliers. The model template  $M_0$  is finally created by cropping an axis-aligned box containing the defined values. The template is visualized in Fig. 7.11(b), with undefined values in black color.

As the uncertainty of the rough alignment can introduce rotation and scale errors next to the translational uncertainty  $\Delta T$ , we rotate the model by the angles  $\Delta\phi$ ,  $\Delta\theta$ , and  $\Delta\psi$  (roll, pitch, yaw) and scale it with a factor  $s = 1.0 \pm \Delta s$  to generate the model templates  $M_t$ . We cover the search space using the coarse-to-fine approach by [27] to speed up computation, and crop the ground template  $G_{\text{gsd}}$  from the DSM image according to the pyramid level gsd (Fig. 7.11(a)).

The score of template  $t$  is finally computed by normalized cross-correlation of ground and model templates,

$$d(t) = \frac{1}{n_t - 1} \sum_{x,y} \frac{(G_{\text{gsd}}(x, y) - \overline{G_{\text{gsd}}})(M_t(x, y) - \overline{M_t})}{\sigma_{G_{\text{gsd}}} \sigma_{M_t}}, \quad (7.4)$$

where  $n_t$  is the number of *defined* pixels for every template  $t$ , and  $\overline{G_{\text{gsd}}}$ ,  $\sigma_{G_{\text{gsd}}}$ ,  $\overline{M_t}$  as well as  $\sigma_{M_t}$  are computed only for *defined* pixels. Additionally, we introduce a term for penalizing alignments which contain a large amount of *undefined* pixels,

$$r(t) = \frac{n_t}{N_t}, \quad (7.5)$$

where  $N_t$  is the number of all pixels in template  $t$ . The best height map alignment is then associated with the best model template

$$t_{\text{best}} = \arg \max_t d(t) + \lambda r(t). \quad (7.6)$$



**Fig. 7.12** Season-Invariant Matching. **(a)** Accurate alignment allows to collect local features in different seasons (*blue, red*) and fuse the overlapping point clouds. **(b)** The aligned models have been acquired in different seasons (summer and winter), as can be seen when looking at the vegetation. ©2011 IEEE. Reprinted, with permission, from [66]

The mode of the difference between the ground template and the best model template, and the ratio of the respective variances, can finally be used to estimate the translation and scaling on the vertical axis.

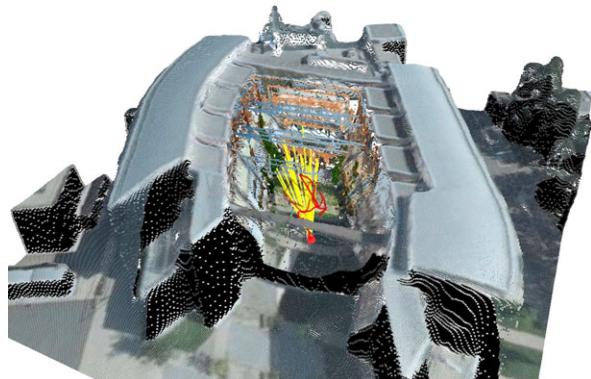
Using height maps for alignment can be seen as an implicit free space constraint, if compared to the work of Kaminsky et al. [27]. Differences between ground and model templates as in Fig. 7.11(c) occur not only when a building height does not fit, but also when the ground height does not fit.

### 7.3.4 Discussion

In this section, we have presented an algorithm for the automatic alignment of partial 3D reconstructions based on [66]. Given an overhead Digital Surface Model (DSM) and approximate GPS tags for the individual camera positions, we refine the alignment based on the correlation of orthographic depth maps. We can handle complex cases where previous methods had problems, including models which do not share any appearance features and models with considerably different ground sampling distances. This allows not only to fuse data acquired by an MAV, but also combining aerial and terrestrial data sources. The accuracy of our approach mainly depends on the ground sampling distance of the overhead digital surface model, which is in the range of 5–30 cm per pixel for modern cameras.

Our alignment approach has several applications: Accurate alignment helps to fuse partial SfM models which could not be connected in the original reconstruction process, and detailed SfM model information can be fused into the original DSM. The alignment of models also helps to achieve season-invariant matching, as models with different appearance and therefore different feature descriptors can be placed in the same coordinate system. The concept of virtual cameras, as presented in the next section, fully integrates this increased amount of descriptors during a localization task. A visualization of two overlapping and two adjacent models, taken in different seasons, can be found in Fig. 7.12.

**Fig. 7.13** Localization in a geo-referenced, metric visual landmark allows to convert the resulting local ECEF coordinates back to a GPS position. It is therefore possible to seamlessly switch between GPS and visual navigation, as well as to exploit onboard GPS and IMU data in the localization process. ©2012 IEEE. Reprinted, with permission, from [69]



## 7.4 Image-Based Localization

A large number of interesting applications for micro aerial vehicles such as surveillance, agriculture, inspection, or simply 3D reconstruction for visualization share one common feature: The need for highly accurate localization of the vehicle with respect to the scene. According to the work of Konolige and Bowman [33] on life-long maps, long-term visual localization requires an accurate map with good features, but should also allow incremental mapping and recovery from localization failures. An additional requirement for MAVs is the possibility to switch between GPS and visual localization, as visual maps are unnecessary when flying high above buildings or in wide open spaces.

In the previous sections, we have shown how to create metric, geo-referenced 3D models by taking pictures with a consumer camera from eye-level above ground or by using micro aerial vehicles as acquisition tools. Now we demonstrate how the concept of virtual views helps to incorporate prior knowledge and fulfills the requirements listed above. We present a method for monocular localization with an accuracy comparable to differential GPS in a global coordinate system (Fig. 7.13). Further, we introduce an incremental feature update step which allows to fuse in-flight information back into the original scene model, and we exploit additional sensor information to ensure localization robustness.

### 7.4.1 Related Work

The problem of vision-based MAV navigation in unstructured environments has been addressed only recently. To retrieve position estimates for visual control, most systems [1, 4, 65] employ simultaneous localization and mapping (SLAM) techniques in indoor and outdoor settings.

SLAM addresses the problem of exploring a previously unknown environment. This requires to determine the sensor pose given an estimated map, and to optimize the map given the sensor information at the estimated pose; two tasks which highly depend on each other and thus have to be solved simultaneously. Early SLAM sys-

tems using a single camera as the main perceptual sensor were based on filtering approaches. Davison [9] fused measurements from a sequence of images by updating probability distributions over features and extrinsic camera parameters using an Extended Kalman Filter (EKF). This requires tracking and mapping to be closely linked, as the camera pose and feature locations are updated together at every single frame. In contrast, Klein and Murray [28, 29] presented a real-time visual SLAM system based on keyframes named Parallel Tracking and Mapping. PTAM employs two separate threads, one for tracking the camera pose through every single frame, and another for mapping the environment by applying bundle adjustment to a set of spatially distributed keyframes. The major drawback of filtering approaches, namely the limited number of features that can be updated in between two frames, is circumvented in PTAM by leaving more time to the parallel mapping process. Strasdat et al. [61] experimentally showed that localization using a large amount of features measured at low temporal frequency is in general superior to a small amount of features measured at every frame, thus keyframe-based methods typically outperform filtering approaches. Still, as the reconstruction is created online, the mapping process is affected by drift and does not scale well to outdoor environments.

We thus want to emphasize the importance of prior knowledge for vision-based navigation tasks. Model-based localization approaches have been extensively investigated over the past decade [37, 55, 73, 76]. Most closely related to our approach is the urban localization approach of Irschara et al. [24] which uses image retrieval-based methods for location recognition. Compared to this class of methods, considerable improvements have recently been achieved [36, 53] using prioritized, direct matching of 3D points to 2D features. However, Sattler et al. [54] have investigated this performance gap. It turns out that it is caused by a large number of incorrect votes cast by standard re-ranking schemes, and can be fixed using their proposed method. Instead, in this work we employ a motion prior to select a small, feasible set of virtual views which is then ranked. The search space is thus much smaller and we are hardly affected by the issue. Apart from that, our approach using virtual views has many benefits over direct matching as described in the next section.

#### **7.4.2 Scalable Localization Using Virtual Cameras**

Our framework provides a visual landmark in terms of a large geo-registered 3D point cloud with associated visual features. However, the information in the landmark is based on a low number of fixed viewpoints and thus does not provide views which are sufficiently close to the images taken by an MAV. Additionally, matching descriptors of a query image against thousands of descriptors in the model is infeasible as it would result in a high number of non-unique matches, and would not scale in terms of computational effort. In contrast, the partitioning of the search space using virtual cameras [24, 67] enables efficient visual localization in outdoor environments. Combined with an incremental feature update step [69] to cope with missing viewing angles in the original model, a robust and scalable localization system is proposed.

### 7.4.2.1 Placement of Virtual Cameras

Virtual cameras hold the correspondences of only those 3D points and features which are visible given the respective camera center, camera orientation, and occlusion surfaces. Motivated by [24], who use synthetic views for fast location recognition in a flat environment, we have extended the concept to 3D space in [67]. We sample camera centers on a regular grid  $G$ , and camera vectors in uniform angular steps of  $\gamma$  on a unit sphere. Additionally, we use the intrinsics of the camera mounted on the MAV to retrieve the correct scale of the features.

After setting up the necessary views, we project all 3D points into the respective cameras. A projected feature has to be in front of the camera and within the extent of the image, and it needs to have a scale larger than one pixel after projection. Additionally, if a surface mesh is available for the model, we restrict the maximum distance of projected points to the distance of the closest surface in viewing direction; this approach avoids adding occluded features. For every 3D point  $X$ , we use the SIFT descriptor [39] with the smallest extraction angle

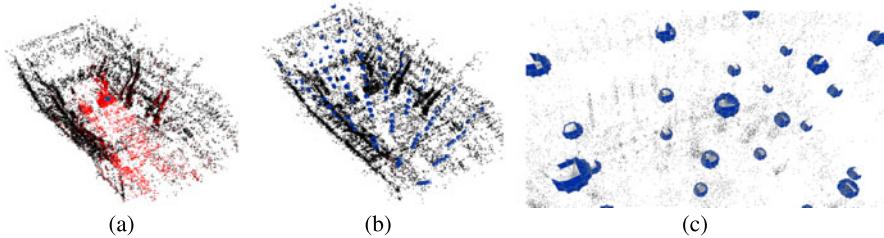
$$\delta_{\min}(X, i) = \min_j \arccos \left( \frac{\mathbf{X}V_i}{\|X V_i\|} \cdot \frac{\mathbf{X}E_j}{\|X E_j\|} \right) \quad (7.7)$$

between the virtual camera  $V_i$  and the model camera  $E_j$ . Consequently,  $\delta_{\min}(X, i)$  increases with any deviation from the camera positions used for acquisition, but especially in higher altitudes when using images taken at ground level to create the model. This increase in extraction angle is visualized in Fig. 7.16(a). However, according to Mikolajczyk et al. [43] SIFT is only robust to off-image-plane rotations up to  $\delta_{\min} < 30^\circ$ . Given, for instance, a distance between camera and point of 10 m, this would mean that the localization only works up to a height of 5 m above the acquisition height, because for higher altitudes matching fails and the pose cannot be estimated robustly. This problem can be circumvented by adding in-flight information to the model; our approach called *incremental feature updates* is described in detail in Sect. 7.4.2.3.

We determine the number of necessary virtual cameras based upon the 3D reconstruction of the scene. The extent and the alignment of the grid  $G$  are defined by the 3D point cloud. To ensure overlapping coverage of the entire model, we increase the number of grid points along each axis until the number of feature points which are visible in less than  $N_{\text{coverage}}$  cameras converges. During this process, all views with less than  $|F|_{\min}$  features are removed. The placement of virtual cameras is visualized in Fig. 7.14.

### 7.4.2.2 Localization Using Virtual Cameras

The resulting virtual cameras  $V$  provide a highly redundant representation of the scene. At first sight, the usage of synthetic views seems to cause an enormous need for feature matching. However, the approach scales very well for image sequences,



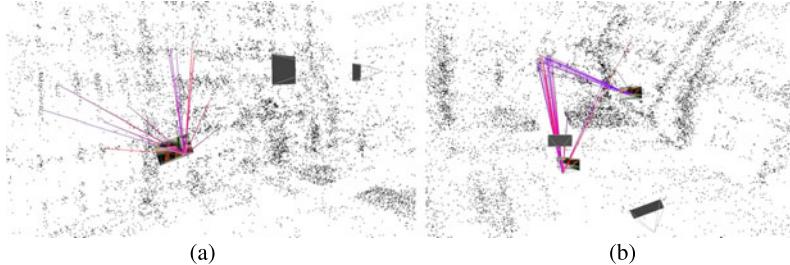
**Fig. 7.14** Placement of virtual cameras. (a) A placement with  $G = \{1, 1, 1\}$  and  $\gamma = 30^\circ$  does not satisfy the required coverage of  $N_{\text{coverage}} = 3$ . Uncovered points are marked in red, virtual cameras in blue. (b) The coverage converges for  $G = \{7, 4, 4\}$  and  $\gamma = 30^\circ$ . (c) A detailed view of the same grid highlights the removal of cameras with less than  $|F|_{\min} = 200$  projected features. ©2011 IEEE. Reprinted, with permission, from [67]

as MAVs do not exceed a predetermined translational and rotational speed in normal operation. The feasible set of virtual cameras  $V_t$  at a certain time  $t$  is defined by the spatial distance to the previously estimated position and orientation of the camera. We define the translational and rotational search radii  $t_{\max}$  and  $R_{\max}$  around that position, which are enlarged in case the MAV cannot localize its position for a number of frames, for instance due to close-up views of an obstacle or unusual movements.

Virtual cameras within that search area are ranked based on appearance using a vocabulary tree [48] and corresponding inverted files. Restricting the search space by cascading prior knowledge and appearance features to the feasible set  $V_t$  allows our approach to handle scenes with unlimited size, and successfully copes with repetitive structures. We sort  $V_t$  according to the scores obtained by the vocabulary tree query, and subsequently use the top-scoring  $k$  views  $V_t^{1:k}$  to geometrically verify the pose of the MAV.

For geometric verification, SIFT features extracted in the query view are matched to those in the virtual view using the GPU and the distance ratio test (see Sect. 7.2.2). We then exploit the knowledge about the 2D to 3D correspondences of points in the virtual view, and estimate a robust absolute pose from three point correspondences (P3P) [32] using fast RANSAC methods [11]. Finally, only poses with a minimal number of effective inliers,  $I_{\text{eff}} \geq t_{\text{eff}}$ , are accepted. The measure of effective inliers [24] reports the spatial distribution of inliers within the image (Fig. 7.3); a high value indicates that the inliers are well distributed and therefore more reliable. Figure 7.15 visualizes the top three virtual cameras  $V_t^{1:3}$  and the corresponding estimated absolute poses  $P_t$  for selected query images.

In summary, the main idea is to fill a scene with a large number of virtual cameras which each hold  $N_{V_i}$  correspondences between visible 3D points and features, given the respective camera center and orientation. This overcomes the problem of matching all  $N_{\text{query}}$  features from a query image to all features in the model. Instead, we match only  $N_{\text{query}} \times N_{V_i} \times k$  features for a small, feasible subset of  $k$  virtual cameras. As a result, the approach scales very well and it can cope with ambiguous environments.



**Fig. 7.15** Localization initiated by virtual cameras. **(a)** In most cases, pose estimation based on the virtual cameras  $V_t^{1:3}$  (visualized without texture) results in the same final absolute pose  $P_t$  (textured with the query image). **(b)** If the results do not coincide, the pose  $P_t$  with most effective inliers  $I_{\text{eff}}$  (red matches) is preferred to other poses (blue matches). ©2011 IEEE. Reprinted, with permission, from [67]

Due to the previously discussed process of creating and aligning visual landmarks, the result of the localization process is a metrically correct position which can be converted back to GPS coordinates. The concept of virtual cameras supports the storage of feature descriptors from different acquisition days, thus it is possible to localize the camera in different seasons if the necessary data is available. It is also possible to update virtual cameras, as discussed in the next section.

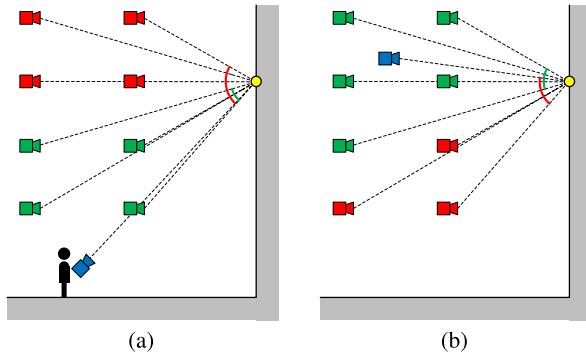
#### 7.4.2.3 Incremental Feature Update

The previously presented approach has a major drawback, which is the lack of good features whenever the angle to the closest view used for modeling the scene is large. As images are typically taken at ground level, this especially applies when the MAV flies in altitudes higher than 5 m, but may also occur in many other scenarios.

In [69], we have proposed to tackle this issue by adding in-flight information to the model, which we call incremental feature update. The visual localization of every single frame results in a set of inlier features which each correspond to a 3D point  $X$ . We evaluate the new extraction angle

$$\delta_{\text{update}}(X, i) = \delta_{\min}(X, i) - \arccos \left( \frac{\mathbf{XL}}{\|\mathbf{XL}\|} \cdot \frac{\mathbf{XVi}}{\|\mathbf{XVi}\|} \right) \quad (7.8)$$

between the localized camera  $L$  and the virtual camera  $V_i$ . Respective inlier features in virtual camera  $i$  are updated, if  $\delta_{\text{update}}(X, i) > th_{\text{update}}$ , that is, if the new extraction angle is at least  $th_{\text{update}}$  smaller than the previous one. Note that just the descriptor has to be exchanged; 3D points and 2D feature positions in the virtual camera remain constant. This update step is very fast and does not reduce the localization rate of 4 fps, as we exploit a precomputed list of relevant virtual cameras for every 3D point. By computing the extraction angle for every virtual view individually, we ensure that the average extraction angle is optimized. Figure 7.16(b) visualizes



**Fig. 7.16** Incremental feature update. **(a)** Depending on the acquisition strategy, some virtual views (*red cameras*) might have an extraction angle larger than  $30^\circ$  compared to the model view (*blue camera*) which contains the real visual information. **(b)** The problem can be circumvented by incrementally updating virtual cameras with in-flight information (*blue camera*). The extraction angle for *green cameras* is reduced and features are updated; the extraction angle for *red cameras* would be increased so the features stay the same. ©2012 IEEE. Reprinted, with permission, from [69].

the update of cameras which initially had a high extraction angle  $\delta_{\min}(X, i)$ . Virtual cameras in high altitudes are updated, while the features for lower altitudes remain the same.

Incremental feature updates are generally applied in an *online* fashion during the flight of the MAV. This allows a large boost in performance, but early frames do not profit from new information in later stages of the flight. Therefore, we run an additional *batch* update with several iterations (until convergence) after the flight. Even a single additionally registered camera can provide the necessary descriptors for later iterations. During the batch update process, we also update the inverted file tables which are used in vocabulary tree ranking.

#### 7.4.2.4 Validation Using Additional Sensors

A well-known problem in online learning and tracking is drift due to wrong feature updates, called the *template update problem* [41]. If wrong localizations are taken to be correct, the model is disturbed and the localization drifts towards the erroneous features. Incremental feature updates as presented in the previous section can also be affected by these issues, even though robust methods are applied already. To overcome this limitation, we propose to validate visual localization by using the additional global sensors available to an MAV.

We employ the magnetic field sensors and compass of the IMU to validate the estimated attitude, and the GPS readings and pressure sensor to evaluate the estimated position. Thanks to the alignment of our visual landmark introduced in Sect. 7.3, the comparison in a global coordinate system is straight-forward.

For representing the attitudinal error, we choose a measure based on a quaternion representation. A quaternion is defined as  $Q = [\rho^T \ q_w]^T$ , with  $\rho = \mathbf{a} \sin(\frac{\theta}{2})$  and  $q_w = \cos(\frac{\theta}{2})$ , where  $\mathbf{a}$  corresponds to the axis and  $\theta$  to the angle of rotation. Thus, the attitude error between the camera attitude quaternion  $Q_{\text{cam}}$  and the IMU attitude quaternion  $Q_{\text{magnetic}}$  is calculated as

$$e_{\text{attitude}} = 2 \arccos(\| Q_{\text{cam}}^{-1} Q_{\text{magnetic}} \|_w), \quad (7.9)$$

where  $\| \cdot \|_w$  is an operation that first normalizes the quaternion resulting from the quaternion multiplication and then extracts  $q_w$ . The position error is given as

$$e_{\text{position}} = \left\| \begin{pmatrix} x_{\text{cam}} \\ y_{\text{cam}} \\ z_{\text{cam}} \end{pmatrix} - \begin{pmatrix} x_{\text{latitude}} \\ y_{\text{longitude}} \\ z_{\text{pressure}} \end{pmatrix} \right\|, \quad (7.10)$$

with  $x_{\text{latitude}}$  and  $y_{\text{longitude}}$  as GPS position converted to a local ECEF coordinate system, and  $z_{\text{pressure}}$  as pressure height over ground. These error measures are required to be smaller than a threshold  $th_{\text{attitude}}$  and  $th_{\text{position}}$ , respectively. Shadowing effects, weather changes, and magnetic fields may disturb these sensors as well, so the validation should serve as a sanity check rather than a narrow rejection criterion.

The combination of information about the rough orientation and position of the MAV in a world coordinate system also allows us to switch seamlessly between GPS and visual localization, for instance when navigating between several visual landmarks. Moreover, it enhances scalability to a world environment as the feasible set of virtual cameras discussed before is further restricted.

### 7.4.3 Discussion

We have presented an algorithm for monocular visual localization of MAVs using prior knowledge about the scene. Our work is based on the concept of virtual views in 3D space, which allows to partition the search space for speeding up localization in large environments, but also supports the integration of in-flight information to improve the initial scene representation. Localization results generated by our algorithm are in a world coordinate system, which allows the MAV to switch seamlessly between GPS and visual localization.

In contrast to visual SLAM, our localization method directly allows global registration and is neither prone to drift nor bias. However, with 4 fps it is also much slower than typical feature tracking approaches, and a speed of at least 25 fps is required for effective visual serving [1, 71]. Using our distributed visual SLAM approach [70] which tracks features onboard the MAV but sources the mapping task out, it is straight-forward to achieve short-term localization on the vehicle and correct its map with geo-registered, drift- and bias-free information delivered by our approach. This would push our framework even more in the direction of long-term operation.

## 7.5 Experiments and Results

In the following paragraphs, we evaluate our localization approach, and accordingly the visual landmark creation and alignment compared to visual and GPS-based groundtruth. The evaluation site, the atrium of a modern building, has already been used to visualize the individual steps of our approach. This environment is very challenging, as there is little texture on the facades and repetitive structures such as windows and vegetation prevail.

### 7.5.1 Evaluation Dataset and Hardware

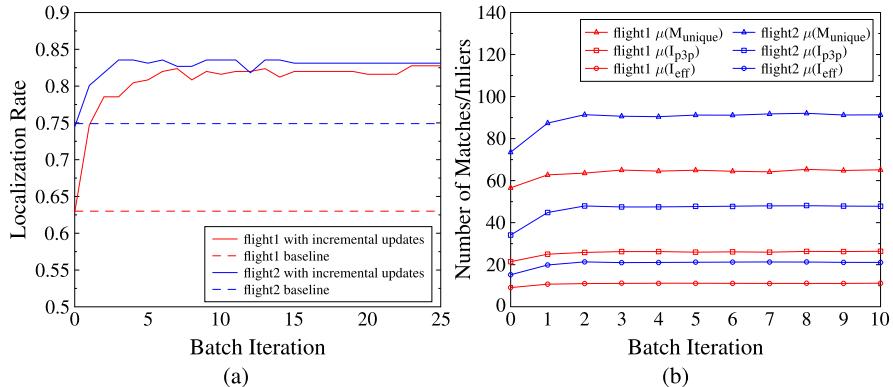
For the creation of the initial visual landmark we captured 157 images from eye-level above ground using a Canon EOS 5D. This SLR camera has a resolution of  $5616 \times 3744$  pixels with a fixed 20 mm lens.

Aerial imagery is obtained using a Pelican quad-rotor MAV by Ascending Technologies and a single industrial camera (IDS UI-1240SE) with a resolution of  $1280 \times 1024$  pixels, global shutter, and a 12 mm lens. Thanks to the on-board computer running the Robot Operating System (ROS) and a wireless 802.11n link, we are able to stream high-quality images to the ground at 4 fps. Our ground station is equipped with a powerful graphics card running NVIDIA CUDA, which allows highly parallel data processing for feature extraction and matching.

Additional sensor data from the inertial measurement unit (IMU), compass, and global positioning system (GPS) are automatically preprocessed and fused by the MAV. While hardware synchronization with the camera trigger would be desirable, we currently exploit the high update rates of the sensors and collect IMU and GPS messages between triggering and receiving an image. Attitude and position are then estimated using robust statistics.

For the iterative update of the model, we acquired two image sequences ( $1280 \times 1024$  px at 4 fps) of 260 (*flight1*) and 230 images (*flight2*) by exploring the atrium in manual flight mode. Each flight started at the ground and covered a volume of about  $10 \times 10 \times 15$  m<sup>3</sup>. Additionally, we have acquired a video stream showing the flight from an observer's point of view. This stream has been recorded with a resolution of  $1920 \times 1028$  pixels at 25 fps, and is used as groundtruth for evaluating our localization performance qualitatively. Finally, for a quantitative validation of the position accuracy we captured images at fixed points which were globally localized using differential GPS (D-GPS).

The localization process including the online update runs at 4 fps. The runtime of a batch update depends on the length of the flight and the number of modified virtual cameras, and results to approx. 5 min per iteration on our datasets. For the following experiments, a minimal virtual camera coverage  $N_{\text{coverage}} = 3$  is achieved by setting the grid to  $G = \{7, 4, 4\}$  and the angular step to  $\gamma = 30^\circ$ . This results in  $112 \times 62 = 6944$  virtual cameras, which are reduced to a set of 1903 virtual views



**Fig. 7.17** Localization improvement for two test sets. **(a)** Dashed lines show the baseline using the non-adaptive algorithm [67] whereas solid lines depict correctly localized frames ( $R_{\text{valid}}$ ) based on IMU validation. Our adaptive method improves the localization performance  $R_{\text{valid}}$  from 62.8 % to 83.1 % for *flight1*, and 74.5 % to 83.5 % for *flight2*. **(b)** For both testsets, all quality measures improve as well when applying incremental feature updates. ©2012 IEEE. Reprinted, with permission, from [69]

by thresholding the minimal number of features to  $|F|_{\min} = 200$ . For localization, we estimate the pose of at most  $k = 3$  top-scoring views in a neighborhood of  $t_{\max} = 1.0$  m and  $R_{\max} = 45^\circ$  with respect to the previously established pose, and accept the pose if it contains at least  $th_{\text{eff}} = 3$  effective inliers. Incremental updates are evaluated with  $th_{\text{altitude}} = 10^\circ$ ,  $th_{\text{position}} = 20$  m, and  $th_{\text{update}} = 1^\circ$ .

### 7.5.2 Comparison of Adaptive and Non-adaptive Localization

Our first experiment evaluates the localization performance as the ratio of correctly localized frames based on additional sensor data, as previously described in Sect. 7.4.2.4. The ratio of frames which are localized correctly is named  $R_{\text{valid}}$ .  $R_{\text{invalid}}$  describes the ratio of frames which exceed  $th_{\text{altitude}}$  or  $th_{\text{position}}$  and therefore fail validation.

We achieve a baseline of 63 % for *flight1* and 75 % for *flight2* using the non-adaptive approach [67]. Our evaluation shows that issues with high-altitude views occur starting at a height of about 6 m, which corresponds to the theoretical result shown in the previous section. In contrast, Fig. 7.17(a) shows that adaptive localization can considerably improve these results. After applying 25 incremental updates, meaning that the inliers of all frames were taken into account at every iteration except for the first, we achieve a localization rate of 83 % for both *flight1* and *flight2*. This corresponds to an improvement of up to 30 % in comparison to non-adaptive localization. Even using just the online update during the flight, and neglecting the batch updates afterwards, localization rates of 71 % and 80 % are reached. The on-

**Table 7.1** Localization Results for Different Attitude Errors after 25 Incremental Update Iterations. ©2012 IEEE. Reprinted, with permission, from [69]

Flight	$th_{\text{attitude}}$	$R_{\text{valid}}$	$R_{\text{invalid}}$	$\mu(e_{\text{attitude}})$
1	5°	43.7 %	41.4 %	$3.61^\circ \pm 1.03^\circ$
	10°	82.8 %	2.3 %	$5.04^\circ \pm 1.85^\circ$
	15°	84.3 %	0.8 %	$5.05^\circ \pm 1.90^\circ$
2	5°	53.2 %	42.0 %	$3.52^\circ \pm 0.96^\circ$
	10°	83.1 %	12.1 %	$4.77^\circ \pm 2.00^\circ$
	15°	86.5 %	8.7 %	$5.12^\circ \pm 2.55^\circ$

line update heavily depends on the flight trajectory, as early frames do not benefit from later frames and thus many camera poses might be initially unknown.

Not only the localization rate but also the localization quality in terms of matches and pose estimation inliers is improved by using the adaptive localization. Figure 7.17(b) thus compares the mean values of unique matches ( $M_{\text{unique}}$ ), inliers ( $I_{\text{p3p}}$ ) and effective inliers ( $I_{\text{eff}}$ ). These values considerably increase during the first update and then converge to a maximum, for both test datasets. Especially,  $I_{\text{eff}}$  is important to reliably estimate the MAVs pose, as it additionally incorporates the distribution of RANSAC inliers in an image.  $\mu(I_{\text{eff}})$  increases from 9 to 12 points for *flight1*, and from 15 to 21 points for *flight2*.

The update of the model is a critical point because miss-aligned frames degrade the model quality. We expect that all localizations passing the additional validation step are correct. However, this heavily depends on the parameters  $th_{\text{attitude}}$  and  $th_{\text{position}}$ , so the residual errors have to be checked. The selection of the attitude threshold  $th_{\text{attitude}}$  was done by evaluating the valid localizations, the according drop rates by validation, and the resulting mean attitude errors  $\mu(e_{\text{attitude}})$ . Table 7.1 shows that for increasing  $th_{\text{attitude}}$  the ratio  $R_{\text{valid}}$  and  $\mu(e_{\text{attitude}})$  increase, whereas  $R_{\text{invalid}}$  decreases. The selection of the threshold, in our case  $th_{\text{attitude}} = 10^\circ$ , is a trade-off between detection rates, outlier removal, and expected IMU accuracy, and it corresponds to earlier work on comparing visual and inertial data [35]. In contrast, GPS is in our experience very noisy. We use the real-time accuracy delivered by the GPS receiver, which results in an average position threshold of  $\mu(th_{\text{position}}) = 16.2$  m. It thus only acts as a sanity check for localization in models with repetitive structures.

Figure 7.18 shows the resulting camera positions after applying our approach. Initially localized cameras are depicted in gray, positions found after several batch updates are marked in red. A common source of error are frames distorted by motion blur and joggle, because only little features can be detected. Nevertheless, our proposed approach is able to handle such local failures since it automatically recovers with the next successful pose estimate. Qualitative examples for localization can be found online<sup>1</sup> in form of a video.

<sup>1</sup><http://aerial.icg.tugraz.at>

**Fig. 7.18** Visual localization result for an entire MAV flight. In a non-adaptive setting, we can localize 62.8 % of all frames of this dataset (*gray camera positions*). By using our technique of incremental feature updates, we are able to boost the localization performance to 83.1 %, especially for high-altitude views (*red camera positions*). ©2012 IEEE. Reprinted, with permission, from [69]



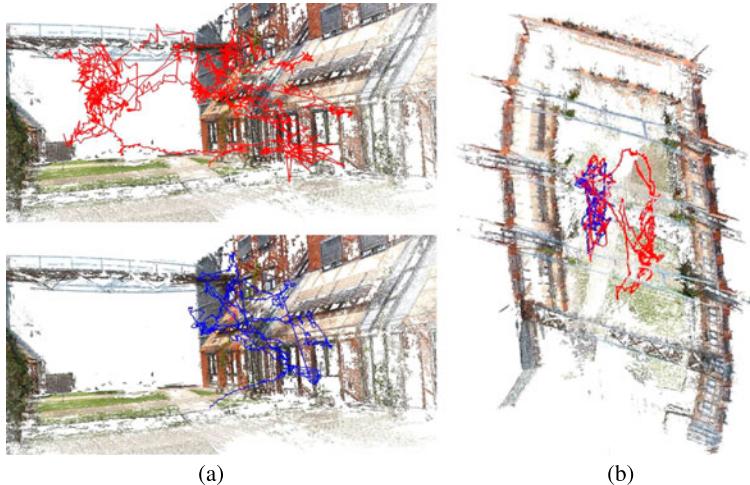
### 7.5.3 Comparison to Visual SLAM

We compare our approach to localization using PTAM [28], which has recently been proposed for MAV navigation [1, 4, 65, 71]. While this is an unfair comparison as PTAM has to create the map on-the-fly while we use prior knowledge, our intention is to demonstrate the significance of using prior information. Our experiments showed that the camera positions obtained by running the original PTAM code vary considerably according to the stereo initialization. Therefore, we used the recorded stream of images (with radial distortion corrected) and selected the initialization which showed best results for comparison. As the global coordinate system and the scale are not defined and differ in every execution, we aligned the camera centers of PTAM to those retrieved by our proposed algorithm in a least-squares sense.

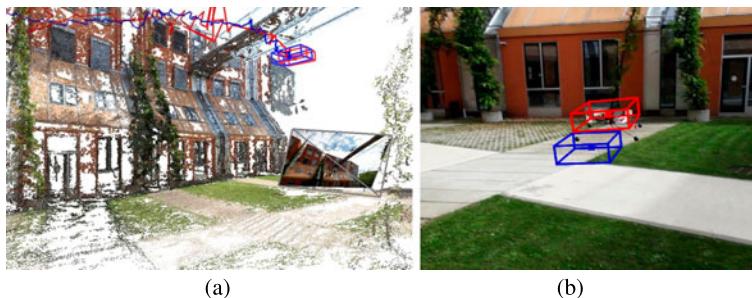
Figure 7.19 depicts the reconstructed flight path from take-off to landing. While PTAM can compete with our approach locally, that is, in the environment it was initialized, it only returns a valid pose for 36 % of all frames. In a direct comparison to our approach, it is clearly visible that PTAM misses large parts of the flight. As soon as the track is lost, the relative pose cannot be established anymore and the localization fails. Another problem is the repetitiveness of the scene, which causes misleading model updates and inhibits successful re-localization within a larger space.

### 7.5.4 Comparison to Visual Groundtruth

We use the video stream showing the quad-rotor's flight from a ground observer's viewpoint as a visual groundtruth. It shares the set of virtual cameras, so we can localize the video within our scene using the same localization method as above.



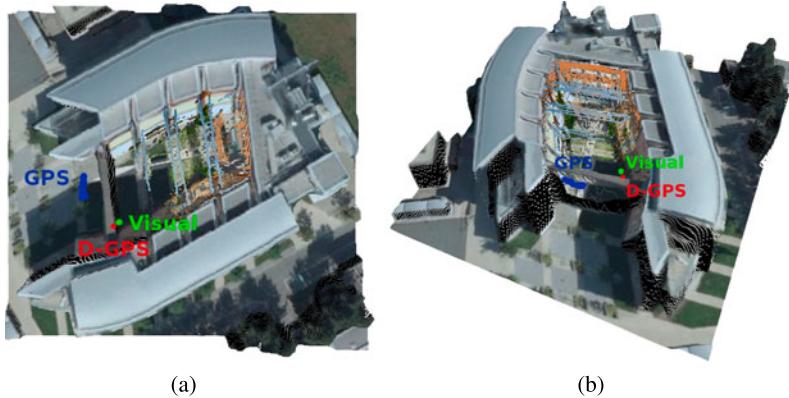
**Fig. 7.19** Visual localization results. **(a)** *Top*: Flight path of our visual landmark-based approach with 83 % of all frames correctly localized. *Bottom*: In comparison, a state-of-the-art visual SLAM approach (PTAM [28]) is not suitable for such a large-scale outdoor environment, and achieves a localization rate of only 36 %. **(b)** The overlay of both paths shows that PTAM has lost track of its initial map at some point during the flight and is not able to recover. ©2011 IEEE. Reprinted, with permission, from [67]



**Fig. 7.20** Visual comparison of the localization results via registered groundtruth video stream. **(a)** Oblique view of registered groundtruth showing the current camera frame and MAV positions (*path and bounding box*) estimated by our localization approach (*red*) and the aligned PTAM result (*blue*). **(b)** Back-projected bounding box from the observer's point of view. ©2011 IEEE. Reprinted, with permission, from [67]

Under the premise of synchronized video streams and a correct current pose estimate from the quad-rotor's flight and ground observing video, the backprojection of the MAV camera center must coincide with the quad-rotor position in the observer's view.

Figure 7.20(a) shows the 3D box of the tracked quad-rotor and the respective ground video frame. We use the widespread PASCAL criterion [10] and consider



**Fig. 7.21** Evaluation of visual and GPS accuracies. Camera poses based on GPS, vision, and differential GPS as reference point are visualized in **(a)** a nadir view and **(b)** an oblique view. Visual localization has been performed on a visual landmark trained by *flight2* to overcome the height limitation, and performs considerably better than GPS. ©2012 IEEE. Reprinted, with permission, from [69]

the back-projected quad-rotor bounding box  $\mathcal{B}_Q$  as successfully registered if the criterion,

$$\frac{\mathcal{B}_Q \cap \mathcal{B}_G}{\mathcal{B}_Q \cup \mathcal{B}_G} > 0.5 \quad (7.11)$$

is satisfied.  $\mathcal{B}_G$  denotes the bounding box of the quad-rotor in the groundtruth view. By visual inspection, we conclude that the localization approach as presented in this paper satisfies the overlap criterion in all frames. This is not very surprising, as the automatic validation process using the IMU removes erroneous localizations. In contrast, visual SLAM without validation (PTAM) is only correct in 22 % of all frames. Figure 7.20(b) gives an example of the typical offset. As such an image can just give an impression, the annotated groundtruth video is again available online.

#### 7.5.4.1 Comparison to Differential GPS

For the evaluation of the on-board sensors, we acquired images as well as GPS and IMU data at a fixed reference point. Further, at the same point we recorded ground-truth position data using a Novatel OEMV-2 L1/L2 Real-Time Kinematic (RTK) receiver with an estimated precision of 0.02 m at the time of image acquisition. Figure 7.21 depicts the results in a nadir and oblique view. The reference point has been chosen on the bridge of the atrium scene because even D-GPS was not able to measure accurate positions on the ground. The mean offset in the  $xy$ -plane for 10 measured GPS points is  $9.76 \pm 1.31$  m, whereas the mean offset of the 10 localized cameras is  $0.31 \pm 0.04$  m. In comparison to our previous work [69], we were able to

**Table 7.2** Long-Term Localization Results: Localization based on the initial visual landmark (flight 1@0, flight 2@0) is compared to localization based on adapted visual landmarks. No further incremental updates were performed on the result. ©2012 IEEE. Reprinted, with permission, from [69]

Flight	$R_{\text{valid}}$	$R_{\text{invalid}}$	$\mu(M_u)$	$\mu(I_{\text{p3p}})$	$\mu(I_{\text{eff}})$
1@0	62.8 %	3.1 %	56.6	21.4	9.1
1@2	73.6 %	2.7 %	60.4	23.7	10.2
2@0	74.5 %	11.3 %	73.5	34.1	15.2
2@1	79.7 %	11.3 %	80.4	38.9	17.4

further reduce the constant offset from 0.52 m to 0.31 m by refining the calibration of the camera. The residual error depends on accurate alignment in a global coordinate system, which in turn depends on the ground sampling distance of the given digital surface model. However, we want to highlight the very good standard deviation of 4 cm in a global coordinate system given the distance to the scene of approx. 20 m; this clearly shows that visual localization is capable of outperforming GPS in urban environments.

We made another interesting observation in the course of producing these ground-truth results: Visual localization did not work for any acquired image on the atrium's bridge in a height of approx. 13 m over ground when using the initial visual landmark. Only after updating the landmark using our adaptive localization algorithm were we able to localize the MAV, which again shows that our proposed incremental update approach is beneficial.

### 7.5.5 Visual Landmarks for Long-Term Localization

Long-term localization represents the localization of a flight using a model that has been updated by previous flights. In this experiment, we evaluate the improvements in localization rate and quality from the initial model to an automatically adapted model. Table 7.2 shows that using a model updated by previous flights improves the detection rate by up to 17 %, and the measurements for the localization quality  $\mu(M_{\text{unique}})$ ,  $\mu(I_{\text{p3p}})$ , and  $\mu(I_{\text{eff}})$  are also improved.

In other words, flying several times in the same area and using the updated model of previous flights improves the localizations of the current flight and allows incremental improvement of the visual landmark. In contrast to visual SLAM, where the environment is expected to be unknown and the map is recreated for every flight, our approach is able to incorporate and propagate prior knowledge in a certain area to improve localization results. We expect that many other localization algorithms which rely on prior knowledge, such as [37], will also benefit from our findings.

## 7.6 Conclusion

We have presented a framework for visual localization of micro aerial vehicles in outdoor environments. Our approach makes use of metric, geo-referenced 3D models. Visual landmarks serve as prior knowledge to the MAV and allow robust, high-accuracy localization in urban environments at 4 frames per second. We have demonstrated an incremental feature update algorithm which fuses in-flight information back into the original scene model and improves localization results by up to 30 %, reaching a total localization rate of 83 % in two experiments. Convergence is ensured by comparing the visual localization to attitude and position information obtained from magnetic sensors and GPS. The ground-truth evaluation using an observer's view and a differential GPS receiver shows an excellent localization accuracy comparable to differential GPS. Our framework is applicable to long-term localization tasks in urban environments and facilitates the combination with short-term, dynamic visual SLAM in future work.

**Acknowledgements** This work has been supported by the Austrian Research Promotion Agency (FFG) projects FIT-IT Pegasus (825841), Construct (830035), and Holistic (830044). The authors would like to thank Arnold Irschara, Christof Hoppe, Michael Maurer, Markus Rumperl, and Christian Mostegel for contributions to earlier work on this topic.

## References

1. Achtelik M, Achtelik M, Weiss S, Siegwart R (2011) Onboard IMU and monocular vision based control for MAVs in unknown in- and outdoor environments. In: IEEE international conference on robotics and vision (ICRA)
2. Agarwal S, Snavely N, Simon I, Seitz SM, Szeliski R (2009) Building Rome in a day. In: IEEE international conference on computer vision (ICCV)
3. Beder C, Steffen R (2006) Determining an initial image pair for fixing the scale of a 3d reconstruction from an image sequence. In: Proc DAGM, pp 657–666
4. Bloesch M, Weiss S, Scaramuzza D, Siegwart R (2010) Vision based MAV navigation in unknown and unstructured environments. In: IEEE international conference on robotics and vision (ICRA)
5. Boissonnat J-D, Yvinec M (1998) Algorithmic geometry. Cambridge University Press, Cambridge
6. Boykov Y, Veksler O, Zabih R (2001) Fast approximate energy minimization via graph cuts. *IEEE Trans Pattern Anal Mach Intell* 23(11):1222–1239
7. Collins RT (1996) A space-sweep approach to true multi-image matching. In: IEEE conference on computer vision and pattern recognition (CVPR)
8. Conway AR (1995) Autonomous control of an unstable model helicopter using carrier phase GPS only. PhD thesis, Stanford University, USA
9. Davison AJ (2003) Real-time simultaneous localisation and mapping with a single camera. In: IEEE international conference on computer vision (ICCV)
10. Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A (2007) The PASCAL visual object classes challenge 2007 results. <http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html>
11. Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun ACM* 24(6):381–395

12. Frahm J-M, Georgel P, Gallup D, Johnson T, Raguram R, Wu C, Jen Y-H, Dunn E, Clipp B, Lazebnik S, Pollefeys M (2010) Building Rome on a cloudless day. In: European conference on computer vision (ECCV)
13. Frueh C, Zakhor A (2003) Constructing 3D city models by merging ground-based and airborne views. In: IEEE conference on computer vision and pattern recognition (CVPR)
14. Fukunaga K, Narendra PM (1975) A branch and bound algorithm for computing k-nearest neighbors. *IEEE Trans Comput* C-24(7):750–753
15. Furukawa Y, Ponce J (2009) Accurate, dense, and robust multi-view stereopsis. In: IEEE transactions on pattern analysis and machine intelligence (PAMI)
16. Golparvar Fard M, Peña-Mora F, Savarese S (2011) Monitoring changes of 3D building elements from unordered photo collections. In: IEEE international conference on computer vision (ICCV) workshops
17. Haralick RM, Lee C, Ottenberg K, Nölle M (1991) Analysis and solutions of the three point perspective pose estimation problem. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 592–598
18. Hartley RI, Sturm PF (1997) Triangulation. *Comput Vis Image Underst* 68(2):146–157
19. Hartley RI, Zisserman A (2004) Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, Cambridge
20. Hoppe C, Klopschitz M, Rumpf M, Wendel A, Kluckner S, Bischof H, Reitmayr G (2012) Online feedback for structure-from-motion image acquisition. In: British machine vision conference (BMVC)
21. Hoppe C, Wendel A, Zollmann S, Pirker K, Irschara A, Bischof H, Kluckner S (2012) Photogrammetric camera network design for micro aerial vehicles. In: Computer vision winter workshop (CVWW)
22. Hrabar S, Sukhatme G (2009) Vision-based navigation through urban canyons. *J Field Robot* 26:431–452
23. Huber PJ (1981) Robust statistics. Wiley, New York
24. Irschara A, Zach C, Frahm JM, Bischof H (2009) From structure-from-motion point clouds to fast location recognition. In: IEEE conference on computer vision and pattern recognition (CVPR)
25. Irschara A, Kaufmann V, Klopschitz M, Bischof H, Leberl F (2010) Towards fully automatic photogrammetric reconstruction using digital images taken from UAVs. In: Proceedings of the ISPRS symposium, 100 years ISPRS—advancing remote sensing science
26. Irschara A, Rumpf M, Meixner P, Pock T, Bischof H (2012) Efficient and globally optimal multi view dense matching for aerial images. In: ISPRS annals of photogrammetry, remote sensing and spatial information sciences
27. Kaminsky RS, Snavely N, Seitz SM, Szeliski R (2009) Alignment of 3D point clouds to overhead images. In: IEEE conference on computer vision and pattern recognition (CVPR) workshops. IEEE, pp 63–70
28. Klein G, Murray DW (2007) Parallel tracking and mapping for small AR workspaces. In: International symposium on mixed and augmented reality (ISMAR)
29. Klein G, Murray D (2008) Improving the agility of keyframe-based SLAM. In: European conference on computer vision (ECCV)
30. Klopschitz M, Irschara A, Reitmayr G, Schmalstieg D (2010) Robust incremental structure from motion. In: International symposium on 3D data processing, visualization and transmission (3DPVT)
31. Kluckner S, Birchbauer JA, Windisch C, Hoppe C, Irschara A, Wendel A, Zollmann S, Reitmayr G, Bischof H (2011) Construction site monitoring from highly-overlapping MAV images. In: IEEE international conference on advanced video- and signal-based surveillance (AVSS)
32. Kneip L, Scaramuzza D, Siegwart R (2011) A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In: IEEE conference on computer vision and pattern recognition (CVPR)

33. Konolige K, Bowman J (2009) Towards lifelong visual maps. In: International conference on intelligent robots and systems (IROS)
34. Labatut P, Pons JP, Keriven R (2007) Efficient multi-view reconstruction of large-scale scenes using interest points, delaunay triangulation and graph cuts. In: IEEE international conference on computer vision (ICCV)
35. Labrosse F (2006) The visual compass: performance and limitations of an appearance-based method. *J Field Robot* 23(10):913–941
36. Li Y, Snavely N, Huttenlocher DP (2010) Location recognition using prioritized feature matching. In: European conference on computer vision (ECCV)
37. Lim H, Sinha SN, Cohen MF, Uyttendaele M (2012) Real-time image-based 6-dof localization in large-scale environments. In: IEEE conference on computer vision and pattern recognition (CVPR)
38. Low K (2004) Linear least-squares optimization for point-to-plane ICP surface registration. Technical report, TR04-004, University of North Carolina
39. Lowe D (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110
40. Lupashin S, Schoellig A, Sherback M, D’Andrea R (2010) A simple learning strategy for high-speed quadrocopter multi-flips. In: IEEE international conference on robotics and vision (ICRA)
41. Matthews L, Ishikawa T, Baker S (2004) The template update problem. In: IEEE transactions on pattern analysis and machine intelligence (PAMI)
42. Mei C, Sibley G, Cummins M, Newman P, Reid I (2009) A constant-time efficient stereo SLAM system. In: British machine vision conference (BMVC)
43. 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* 65:43–72
44. Muja M, Lowe DG (2009) Fast approximate nearest neighbors with automatic algorithm configuration. In: International conference on computer vision theory and applications (VISAPP), pp 331–340
45. Newcombe RA, Davison AJ, Izadi S, Kohli P, Hilliges O, Shotton J, Molyneaux D, Hodges S, Kim D, Fitzgibbon A (2011) KinectFusion: real-time dense surface mapping and tracking. In: International symposium on mixed and augmented reality (ISMAR)
46. Nistér D (2003) An efficient solution to the five-point relative pose problem. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 195–202
47. Nistér D (2004) An efficient solution to the five-point relative pose problem. *IEEE Trans Pattern Anal Mach Intell* 26(6):756–770
48. Nistér D, Stewenius H (2006) Scalable recognition with a vocabulary tree. In: IEEE conference on computer vision and pattern recognition (CVPR)
49. Pollefeys M, Van Gool L, Vergauwen M, Verbiest F, Cornelis K, Tops J, Koch R (2004) Visual modeling with a hand-held camera. *Int J Comput Vis* 59(3):207–232
50. Raguram R, Frahm J-M (2011) RECON: scale-adaptive robust estimation via residual consensus. In: IEEE international conference on computer vision (ICCV)
51. Rudol P, Wzorek M, Doherty P (2010) Vision-based pose estimation for autonomous indoor navigation of micro-scale unmanned aircraft systems. In: IEEE international conference on robotics and vision (ICRA)
52. Rumpler M, Irschara A, Wendel A, Bischof H (2012) Rapid 3d city model approximation from publicly available geographic data sources and georeferenced aerial images. In: Computer vision winter workshop (CVWW)
53. Sattler T, Leibe B, Kobbelt L (2011) Fast image-based localization using direct 2d-to-3d matching. In: IEEE international conference on computer vision (ICCV)
54. Sattler T, Weyand T, Leibe B, Kobbelt L (2012) Image retrieval for image-based localization revisited. In: British machine vision conference (BMVC)
55. Schindler G, Brown M, Szeliski R (2007) City-scale location recognition. In: IEEE conference on computer vision and pattern recognition (CVPR)

56. Schmid K, Hirschmueller H, Doemel A, Grix I, Suppa M, Hirzinger G (2012) View planning for multi-view stereo 3d reconstruction using an autonomous multicopter. *J Intell Robot Syst* 65:309–323
57. Silpa Anan C, Hartley RI (2008) Optimised KD-trees for fast image descriptor matching. In: IEEE conference on computer vision and pattern recognition (CVPR)
58. Sivic J, Zisserman A (2003) Video google: a text retrieval approach to object matching in videos. In: IEEE international conference on computer vision (ICCV), pp 1470–1477
59. Snavely N, Seitz S, Szeliski R (2006) Photo tourism: exploring photo collections in 3D. In: ACM SIGGRAPH
60. Snavely N, Seitz SM, Szeliski RS (2008) Modeling the world from internet photo collections. *Int J Comput Vis* 80(2):189–210
61. Strasdat H, Montiel JMM, Davison AJ (2010) Real-time monocular SLAM: why filter? In: IEEE international conference on robotics and vision (ICRA)
62. Strecha C, Pylvaenaeinen T, Fua P (2010) Dynamic and scalable large scale image reconstruction. In: IEEE conference on computer vision and pattern recognition (CVPR)
63. Szeliski R (2006) Image alignment and stitching: a tutorial. *Found Trends Comput Graph Vis* 2(1):1–104
64. Triggs B, McLauchlan P, Hartley R, Fitzgibbon A (2000) Bundle adjustment—a modern synthesis. In: Vision algorithms: theory and practice, pp 298–375
65. Weiss S, Achterlik M, Kneip L, Scaramuzza D, Siegwart R (2011) Intuitive 3D maps for MAV terrain exploration and obstacle avoidance. *J Intell Robot Syst* 61:473–493
66. Wendel A, Irschara A, Bischof H (2011) Automatic alignment of 3D reconstructions using a digital surface model. In: IEEE international conference on computer vision and pattern recognition (CVPR), workshop on aerial video processing
67. Wendel A, Irschara A, Bischof H (2011) Natural landmark-based monocular localization for MAVs. In: IEEE international conference on robotics and vision (ICRA)
68. Wendel A, Hoppe C, Bischof H, Leberl F (2012) Automatic fusion of partial reconstructions. In: ISPRS annals of photogrammetry, remote sensing and spatial information sciences
69. Wendel A, Maurer M, Bischof H (2012) Visual landmark-based localization for MAVs using incremental feature updates. In: Joint 3DIM/3DPVT conference: 3D imaging, modeling, processing, visualization & transmission (3DIMPVT)
70. Wendel A, Maurer M, Gruber G, Pock T, Bischof H (2012) Dense reconstruction on-the-fly. In: IEEE conference on computer vision and pattern recognition (CVPR)
71. Wendel A, Maurer M, Katusic M, Bischof H (2012) Fuzzy visual servoing for micro aerial vehicles. In: Austrian robotics workshop (ARW)
72. Wu C (2007) SiftGPU: a GPU implementation of scale invariant feature transform (SIFT). <http://cs.unc.edu/~ccwu/siftgpu>
73. Zamir AR, Shah M (2010) Accurate image localization based on google maps street view. In: European conference on computer vision (ECCV)
74. Zhang Z (1994) Iterative point matching for registration of free-form curves and surfaces. *Int J Comput Vis* 13(2):119–152
75. Zhao W, Nister D, Hsu S (2005) Alignment of continuous video onto 3D point clouds. In: IEEE transactions on pattern analysis and machine intelligence (PAMI), pp 1305–1318
76. Zhu ZW, Oskiper T, Samarakone S, Kumar R, Sawhney HS (2008) Real-time global localization with a pre-built visual landmark database. In: IEEE conference on computer vision and pattern recognition (CVPR)
77. Zufferey J-C, Beyeler A, Floreano D (2010) Autonomous flight at low altitude with vision-based collision avoidance and GPS-based path following. In: IEEE International conference on robotics and vision (ICRA)

# Chapter 8

## Moment Constraints in Convex Optimization for Segmentation and Tracking

Maria Klodt, Frank Steinbrücker, and Daniel Cremers

**Abstract** Convex relaxation techniques have become a popular approach to shape optimization as they allow to compute solutions independent of initialization to a variety of problems. In this chapter, we will show that shape priors in terms of moment constraints can be imposed within the convex optimization framework, since they give rise to convex constraints. In particular, the lower-order moments correspond to the overall area, the centroid, and the variance or covariance of the shape and can be easily imposed in interactive segmentation methods. Respective constraints can be imposed as hard constraints or soft constraints. Quantitative segmentation studies on a variety of images demonstrate that the user can impose such constraints with a few mouse clicks, leading to substantial improvements of the resulting segmentation, and reducing the average segmentation error from 12 % to 0.35 %. GPU-based computation times of around 1 second allow for interactive segmentation.

### 8.1 Introduction

Shape optimization is at the heart of several classical computer vision problems such as image segmentation and multi view reconstruction. Following a series of seminal papers [1, 12, 19], functional minimization has become the established paradigm to solve shape optimization problems such as image segmentation or 3D reconstruction. In the spatially discrete setting the study of the corresponding binary labeling problems goes back to the spin-glas models introduced in the 1920s [11]. Popular algorithms to solve the arising shape optimization problems include level set methods [20], graph cuts [9] or convex relaxation [4].

---

M. Klodt (✉) · F. Steinbrücker · D. Cremers  
Department of Informatics, TU München, Garching, Germany  
e-mail: [klodt@tum.de](mailto:klodt@tum.de)

F. Steinbrücker  
e-mail: [steinbrf@tum.de](mailto:steinbrf@tum.de)

D. Cremers  
e-mail: [cremers@tum.de](mailto:cremers@tum.de)

In this chapter, we focus on a class of functionals of the form:

$$E(S) = \int_{\text{int}(S)} f(x) \, dx + \int_S g(x) \, dA, \quad (8.1)$$

where  $S$  denotes a hyper surface in  $\mathbb{R}^d$ , that is, a set of closed boundaries in the case of 2D image segmentation or a set of closed surfaces in the case of 3D segmentation and multi view reconstruction. The functions  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  and  $g : \mathbb{R}^d \rightarrow \mathbb{R}^+$  are application dependent. In a statistical framework for image segmentation, for example,

$$f(x) = \log p_{\text{bg}}(I(x)) - \log p_{\text{ob}}(I(x)), \quad (8.2)$$

may denote the log likelihood ratio for observing the color  $I(x)$  at a point  $x$  given that  $x$  is part of the background or the object, respectively. A common choice for  $g$  is the function

$$g(x) = \frac{1}{1 + \beta |\nabla I_\sigma(x)|^2} \quad (8.3)$$

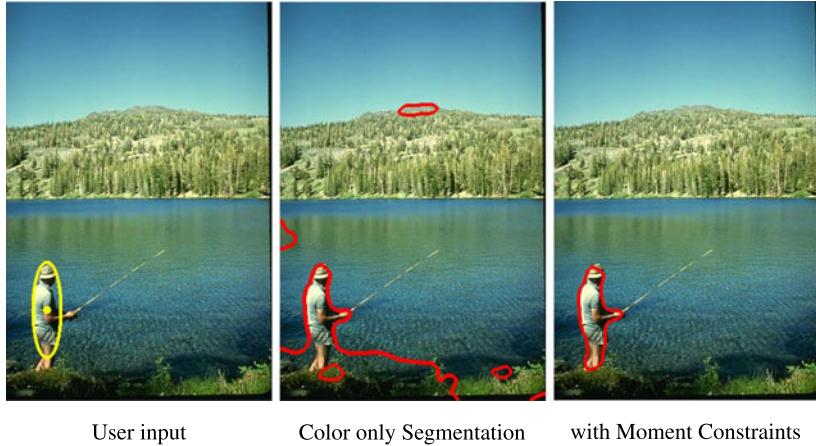
with suitable choices for the parameters  $\beta$  and  $\sigma$ . Here,  $I_\sigma$  is a Gaussian smoothed version of the input image.

The second term in (8.1) corresponds to the area (for  $d = 3$ ) or the boundary length (for  $d = 2$ ), measured in a metric given by the function  $g$ . In the context of image segmentation,  $g$  may be a measure of the local edge strength—as in the geodesic active contours [3, 13]—which energetically favors segmentation boundaries along strong intensity gradients. In the context of multi view reconstruction,  $g(x)$  typically measures the photo-consistency among different views of the voxel  $x$ , where low values of  $g$  indicate a strong agreement from different cameras on the observed patch intensity (Fig. 8.1).

### 8.1.1 Shape Priors for Image Segmentation

There has been much research on imposing prior shape knowledge into image segmentation. While it was shown that segmentation results can be substantially improved by imposing shape priors [5, 8, 10], existing approaches typically suffer from the following problems:

- Apart from a few exceptions such as [23]—computable solutions are only *locally* optimal. As a consequence, one typically needs appropriate initializations and solutions may be arbitrarily far from the globally optimal ones. Other notable exceptions were designed by Veksler and coworkers for specific scenarios such as “compact objects” [6] or “star-shaped objects” [25].
- Many shape priors have a rather fine granularity in the sense that they impose the object silhouette to be consistent with those silhouettes observed in a training set [5, 7]. The degree of abstraction is typically rather small. In particular, deviations of the observed shape from the training shapes are (elastically) suppressed by the



**Fig. 8.1** We propose a convex formulation for interactive image segmentation which allows to impose constraints on moments of arbitrary order. In particular, constraints on the lower order moments (area, centroid, covariance) are easily transmitted through mouse interaction (*left*). They allow to stabilize the segmentation process while preserving fine-scale details of the shape (*right*)

shape prior. This is particularly undesirable in medical image segmentation where malformations of organs (that make it deviate from the training shapes of healthy organs) should be detected rather than ignored. It may therefore be of interest to merely impose some coarse-level shape information rather than imposing the exact form of the object.

An alternative approach that may provide a remedy for both of the above problems is to impose moment constraints. In particular, the lower-order moments allow to constrain the area/volume, the centroid and the size or covariance of objects without imposing any constraints on their local shape. A related idea of using *Legendre moments* (albeit in a local optimization scheme) was developed in [8].

In a convex formulation of multiple view 3D reconstruction, it was recently shown [15] that one can impose additional convex constraints which assure that the computed minimal surfaces are silhouette-consistent. Essentially this constraint can be seen as a *volume constraint*: The volume along any ray from the camera center must be at least 1 if that ray passes through the silhouette and zero otherwise. In the two-dimensional case, a related constraint was recently proposed as a bounding box prior for image segmentation [17].

### 8.1.2 Contribution

We show that one can impose an entire family of moment constraints in the framework of convex shape optimization, thereby generalizing from the zeroth order moment (volume) to higher order moments (centroid, scale, covariance, etc). In particular, all moment constraints—both soft or hard—correspond to convex constraints.

As a consequence we can compute moment-constrained shapes which are independent of initialization and lie within a bound of the optimum.

The outline of the paper is as follows. In Sect. 8.2, we will briefly review a framework for convex relaxation and thresholding which allows to efficiently compute global minima of the above energies in a spatially continuous setting. In Sect. 8.3, we will then show that moment constraints can be imposed as convex constraints within the optimization framework. In Sect. 8.4, we show how the arising optimization problem can be minimized using efficient GPU-accelerated PDE solving. We will furthermore show that computing projections onto the moment constraint sets reduces to solving systems of linear equations. Section 8.5 shows how the presented method for image segmentation with moment constraints can be extended to a method for object tracking in videos. In Sect. 8.6, we present experimental results and a quantitative evaluation showing that interactive segmentation results can be drastically improved using moment constraints.

## 8.2 Shape Optimization via Convex Relaxation

Functionals of the form (8.1) can be globally optimized in a spatially continuous setting by means of convex relaxation and thresholding [4]. To this end, one reverts to an implicit representation of the hyper surface  $S$  using an indicator function  $u \in BV(\mathbb{R}^d; \{0, 1\})$  on the space of binary functions of bounded variation, where  $u = 1$  and  $u = 0$  denote the interior and exterior of  $S$ . The functional (8.1) defined on the space of surfaces  $S$  is therefore equivalent to the functional

$$E(u) = \int_{\Omega} f(x)u(x) dx + \int_{\Omega} g(x)|Du(x)|, \quad (8.4)$$

where the second term in (8.4) is the weighted total variation. Here  $Du$  denotes the distributional derivative which for differentiable functions  $u$  boils down to  $Du(x) = \nabla u(x) dx$ . By relaxing the binary constraint and allowing the function  $u$  to take on values in the interval between 0 and 1, the optimization problem becomes that of minimizing the convex functional (8.4) over the convex set  $BV(\mathbb{R}^d; [0, 1])$ . Global minimizers  $u^*$  of this relaxed problem can therefore efficiently be computed, for example by a simple gradient descent procedure.

The thresholding theorem [4] assures that thresholding the solution  $u^*$  of the relaxed problem preserves global optimality for the original binary labeling problem. We can therefore compute global minimizers for functional (8.4) in a spatially continuous setting as follows: Compute a global minimizer  $u^*$  of (8.4) on the convex set  $BV(\mathbb{R}^d; [0, 1])$  and threshold the minimizer  $u^*$  at any value  $\mu \in (0, 1)$ .

The following theorem [4] assures that thresholding the solution  $u^*$  of the relaxed problem provides a minimizer of the original binary labeling problem (8.4), in other words the convex relaxation preserves global optimality for the original binary labeling problem.

**Theorem 8.1** Let  $u^* \in BV(\mathbb{R}^d; [0, 1])$  be a global minimizer of the functional (8.4). Then all upper level sets (i.e. thresholded versions)

$$\Sigma_{\mu, u^*} = \{x \in \mathbb{R}^d \mid u^*(x) > \mu\}, \quad \mu \in (0, 1), \quad (8.5)$$

of  $u^*$  are minimizers of the original binary labeling problem (8.1).

*Proof* Using the layer cake representation of the function  $u^* \in BV(\mathbb{R}^d; [0, 1])$ :

$$u^*(x) = \int_0^1 1_{\Sigma_{\mu, u^*}} d\mu \quad (8.6)$$

we can rewrite the first term in the functional (8.4) as

$$\int_{\mathbb{R}} f u^* dx = \int_{\mathbb{R}} f \left( \int_0^1 1_{\Sigma_{\mu, u^*}} d\mu \right) dx = \int_0^1 \int_{\Sigma_{\mu, u^*}} f(x) dx. \quad (8.7)$$

As a consequence, the functional (8.4) takes on the form:

$$E(u^*) = \int_0^1 \left\{ \int_{\Sigma_{\mu, u^*}} f dx + |\partial \Sigma_{\mu, u^*}|_g \right\} d\mu \equiv \int_0^1 \widehat{E}(\Sigma_{\mu, u^*}) d\mu, \quad (8.8)$$

where we have used the coarea formula to express the weighted total variation norm in (8.4) as the integral over the length of all level lines of  $u$  measured in the norm induced by  $g$ . Clearly the functional (8.8) is now merely an integral of the original binary labeling problem  $\widehat{E}$  applied to the upper level sets of  $u^*$ .

Assume that for some threshold value  $\tilde{\mu} \in (0, 1)$  Theorem 8.1 was not true, that is, there exists a minimizer  $\Sigma^*$  of the binary labeling problem with smaller energy:

$$\widehat{E}(\Sigma^*) < \widehat{E}(\Sigma_{\tilde{\mu}, u^*}). \quad (8.9)$$

Then—using continuity arguments—we have for the indicator function  $1_{\Sigma^*}$  of the set  $\Sigma^*$ :

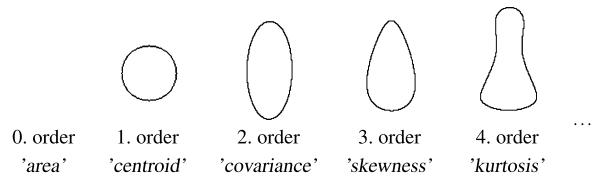
$$E(1_{\Sigma^*}) = \int_0^1 \widehat{E}(\Sigma^*) d\mu < \int_0^1 \widehat{E}(\Sigma_{\mu, u^*}) d\mu = E(u^*), \quad (8.10)$$

which contradicts the assumption that  $u^*$  was a global minimizer of (8.4).  $\square$

The above theorem allows to compute global minimizers of the functional (8.4) in a spatially continuous setting as follows:

1. Compute a global minimizer  $u^*$  of (8.4) on the convex set  $BV(\mathbb{R}^d; [0, 1])$ .
2. Threshold the minimizer  $u^*$  at any value  $\mu \in (0, 1)$  to obtain a binary solution of the original shape optimization problem.

**Fig. 8.2** 2D central moments of a shape. With increasing order of the moment more details of the shape can be described



## 8.3 Moment Constraints for Segmentation

In the following, we will describe the moment constraints presented in [14]. We will successively constrain the moments of the segmentation and show how all of these constraints give rise to nested convex sets. To this end, we will represent shapes in  $d$  dimensions as binary indicator functions  $u \in BV(\Omega; \{0, 1\})$  of bounded variation on the domain  $\Omega \subset \mathbb{R}^d$ . We will denote the convex hull of this set by  $\mathcal{B} = BV(\Omega; [0, 1])$ .

Figure 8.2 shows examples for the first five 2D central moments of a shape. The first order moment describes the centroid of a shape, the second describes the covariance, that is, the relation between width and height of an object. While with the third order moment egg-shapes of shapes can be described, for the fourth order moments the effect on the shape already becomes less intuitive. The figure shows how the higher the order of a moment, the more sophisticated properties can be imposed on the corresponding shape. In the following, we will show how these moments can be used to constrain properties of shapes for segmentation in a convex framework, and how especially the lower order moments can constrain shape optimization in a very intuitive way.

### 8.3.1 Area Constraint

We can impose that the area of the shape  $u$  to be bounded by a constant  $c \in \mathbb{R}^+$  by constraining  $u$  to lie in the set:

$$\mathcal{C}_0 = \left\{ u \in \mathcal{B} \mid \int_{\Omega} u \, dx = c \right\}. \quad (8.11)$$

In the case of a 3 dimensional domain  $\Omega$ , the constant  $c$  constrains the volume of a shape.

**Proposition 8.1** *For any constant  $c \geq 0$ , the set  $\mathcal{C}_0$  is convex.*

*Proof* Let  $u_1, u_2 \in \mathcal{C}_0$  be two elements from this set. Then for any convex combination  $u_{\alpha} = \alpha u_1 + (1 - \alpha) u_2$ ,  $\alpha \in [0, 1]$  of these elements we have:

$$\int_{\Omega} u_{\alpha} \, dx = \alpha \int_{\Omega} u_1 \, dx + (1 - \alpha) \int_{\Omega} u_2 \, dx. \quad (8.12)$$

As a consequence, we have  $\int_{\Omega} u_{\alpha} \, dx = c$  such that  $u_{\alpha} \in \mathcal{C}_0$ .  $\square$

In practice, we can either impose an exact area or we can impose upper and lower bounds on the area with two constants  $c_1 \leq c_2$  and constrain the area to lie in the range  $[c_1, c_2]$ .

Alternatively, we can impose a soft area constraint by enhancing the functional (8.4) as follows:

$$E_0(u, \lambda_0) = E(u) + \lambda_0 \left( \int u \, dx - c \right)^2, \quad (8.13)$$

which imposes a soft constraint with a weight  $\lambda_0 > 0$  favoring the area of the estimated shape to be near  $c \geq 0$ . Clearly, the functional (8.13) is also convex.

### 8.3.2 Centroid Constraint

Assume that someone gave us some bounds about the centroid (center of gravity) for the object we want to reconstruct. We can impose the centroid of the shape by constraining the solution  $u$  to the set  $\mathcal{C}_1$ :

$$\mathcal{C}_1 = \left\{ u \in \mathcal{B} \mid \frac{\int_{\Omega} xu \, dx}{\int_{\Omega} u \, dx} = \mu \right\}, \quad (8.14)$$

where equality is to be taken point wise and  $\mu \in \mathbb{R}^d$ . Alternatively, we can impose the centroid to lie between two constants  $\mu_1, \mu_2 \in \mathbb{R}^d$ . For  $\mu_1 = \mu_2$ , the centroid is fixed.  $\mathcal{C}_1$  contains all shapes whose first order moment is  $\mu$ . In probability theory, the first order moment is also called the expected value.

**Proposition 8.2** *For any constant  $\mu \geq 0$ , the set  $\mathcal{C}_1$  is convex.*

*Proof* The equality constraint in (8.14) is equivalent to

$$\int_{\Omega} xu \, dx = \mu \int_{\Omega} u \, dx, \quad (8.15)$$

which is clearly a linear constraint.  $\square$

Reformulating the constraint equation in (8.14) leads to

$$\int_{\Omega} (\mu - x)u \, dx = 0, \quad (8.16)$$

allowing us to impose the centroid as a soft constraint by minimizing the energy:

$$E_1(u, \lambda_1) = E(u) + \lambda_1 \left( \int_{\Omega} (\mu - x)u \, dx \right)^2. \quad (8.17)$$

Energy (8.17) is also convex in  $u$ . Interestingly this soft constraint does not minimize the quadratic difference to the specified center  $\mu$ ; the latter would not be

convex. In contrast to the hard constraint, this soft constraint unfortunately exhibits a preference toward smaller shapes as it vanishes with decreasing object size:

$$\left( \int_{\Omega} (\mu - x) u \, dx \right)^2 = \left( \mu - \frac{\int_{\Omega} x u \, dx}{\int_{\Omega} u \, dx} \right)^2 b^2, \quad (8.18)$$

where  $b = \int u \, dx$  denotes the size of the object.

### 8.3.3 Covariance Constraint

The proposed concept can be generalized to moments of successively higher order, where we shall focus on so-called central moments (i.e., moments with respect to a specified centroid). In particular, the respective structures will generally be tensors of higher dimension. One can impose the covariance structure by considering the following convex set:

$$\mathcal{C}_2 = \left\{ u \in \mathcal{B} \mid \frac{\int_{\Omega} (x - \mu)(x - \mu)^T u \, dx}{\int_{\Omega} u \, dx} = A \right\}, \quad (8.19)$$

where the equality constraint should be taken element wise. Here  $\mu \in \mathbb{R}^d$  denotes the center and  $A \in \mathbb{R}^{d \times d}$  denotes a symmetric matrix. Alternatively, we can impose the covariance to lie in a range between two symmetric matrices  $A_1, A_2 \in \mathbb{R}^{d \times d}$  such that  $A_1 \leq A_2$  element-wise. Constraining the covariance, that is, the second order moment, we are able to constrain the relation between width and height of an object. This constraint is particularly meaningful if one additionally constrains the centroid to be  $\mu$ , that is, considers the intersection of the set (8.19) with a set of the form (8.14).

**Proposition 8.3** *For any constant  $A \geq 0$ , the set  $\mathcal{C}_2$  is convex.*

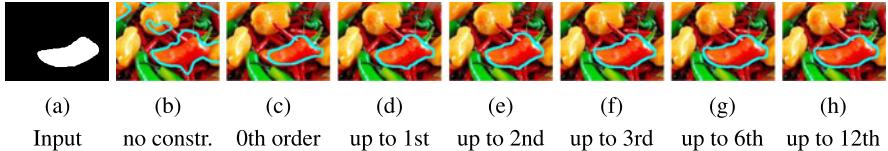
*Proof* The proof is analogous to that of Proposition 8.2. □

We can derive the corresponding soft constraint on the covariance matrix by adding a respective term to the original energy:

$$E_2(u, \lambda_2) = E(u) + \lambda_2 \left( \int_{\Omega} (A - (x - \mu)(x - \mu)^T) u \, dx \right)^2. \quad (8.20)$$

Note that this allows, in particular, to constrain the scale  $\sigma$  of the object, because:

$$\sigma^2 = \frac{\int_{\Omega} (x - \mu)^2 u \, dx}{\int_{\Omega} u \, dx} = \text{tr} \frac{\int_{\Omega} (x - \mu)(x - \mu)^T u \, dx}{\int_{\Omega} u \, dx}. \quad (8.21)$$



**Fig. 8.3** Segmentation results with higher order moment constraints: By imposing constraints of increasing order (up to 12th order), more and more fine scale details of the shape are restored

From the constraint in (8.19), it follows that:

$$\text{tr}(A_1) \leq \sigma^2 \leq \text{tr}(A_2). \quad (8.22)$$

Here,  $\text{tr}$  denotes the trace of a matrix.

### 8.3.4 Higher Order Moment Constraints

In general, we can impose constraints on moments of any order  $k \in \mathbb{N}$ :

$$\mathcal{C}_k = \left\{ u \in \mathcal{B} \mid \frac{\int_{\Omega} (x_1 - \mu_1)^{i_1} \dots (x_d - \mu_d)^{i_d} u \, dx}{\int_{\Omega} u \, dx} = a_{i_1 \dots i_d} \right\}, \quad (8.23)$$

where  $i_1 + \dots + i_d = k$  and  $a_{i_1 \dots i_d}$  can be chosen arbitrarily to constrain the moment tensor of order  $k$ . Here  $x_i$  and  $\mu_i$  denotes the  $i$ th component of  $x$  and  $\mu$ , respectively.

**Proposition 8.4** *For all  $i_1, \dots, i_d \in \mathbb{N}$  and for any constants  $a_{i_1 \dots i_d} \leq b_{i_1 \dots i_d}$ , the set  $\mathcal{C}_{i_1 \dots i_d}$  is convex.*

*Proof* The proof is analogous to that of Proposition 8.2. □

The above properties allow to impose various constraints on the shape associated with the indicator function  $u$ . Imposing more and more constraints of increasingly higher order leads to a smaller and smaller intersection of the associated convex sets as a feasible domain of the shape and a corresponding hierarchy of shape details being imposed in the segmentation. How much shape detail can one impose in this manner?

**Proposition 8.5** *Similarity to any given shape can be imposed at arbitrary detail by imposing convex moment constraints of increasingly higher order.*

*Proof* According to the uniqueness theorem of moments [21], the function  $u$  is uniquely defined by its moment sequence. □

Figure 8.3 shows an example of segmentations with high order moment constraints: While the higher-order moments allow to recover fine-scale shape details, the shape improvements due to higher order constraints are fairly small. Furthermore imposing moments of higher order is not very practical: First, the user can-

not estimate these moments visually. Second, the user cannot transmit respective higher-order tensors through a simple mouse interaction. Instead, having the image data determine the shape's fine scale structure turns out to be far more useful.

Figure 8.3(a) shows the manually segmented input shape which is used to compute color histograms for foreground and background, as well as the moments that constrain the subsequent image segmentation. Figure 8.3(b) shows that color-only segmentation based on color histograms for foreground and background is not sufficient in this example image to correctly segment one red pepper from the others. Figure 8.3(c)–(e) show that the first three moment constraints—area, centroid and covariance—are able to substantially improve segmentations, however are in this hard case, not enough for an exact segmentation. In Fig. 8.3(f)–(h), we can see how with increasing order of moment constraints more and more fine scale details of the shape can be reconstructed. Even the 12th order moment constraint still results in an improvement of segmentation.

In the example of Fig. 8.3, the respective higher order moment constraints have been learned from a training shape. For interactive applications however, the higher order moments are too less intuitive to be applicable. Therefore, for the application we consider—namely interactive image segmentation—we shall in the following limit ourselves to imposing moments up to 2nd order (area/volume, center of mass, scale and covariance).

## 8.4 Shape Optimization with Moment Constraints

Shape optimization and image segmentation with respective moment constraints can now be done by minimizing convex energies under respective convex constraints.

Let  $\mathcal{C}$  be a specific convex set containing knowledge about respective moments of the desired shape—given by an intersection of the above convex sets. Then we can compute segmentations by solving the convex optimization problem

$$\min_{u \in \mathcal{C}} E(u), \quad (8.24)$$

with  $E(u)$  given in (8.4). In this work, we solve the Euler–Lagrange equation:

$$0 = \operatorname{div} \left( g \frac{\nabla u}{|\nabla u|} \right) - f. \quad (8.25)$$

The equation system can be solved using gradient descent, or the lagged diffusivity approach that was presented in [16]. We use the latter because in our experiments it achieves a speed up of computation times of a factor of  $\sim 5$ . An update step for  $u$  at pixel  $i$  and time step  $k$  yields

$$u_i^{l,k+1} = (1 - \omega)u_i^{l,k} + \omega \frac{v \sum_{j \in \mathcal{N}(i), j < i} g \cdot g_{i \sim j}^l u_j^{l,k+1} + v \sum_{j \in \mathcal{N}(i), j > i} g \cdot g_{i \sim j}^l u_j^{l,k} - f_i}{v \sum_{j \in \mathcal{N}(i)} g \cdot g_{i \sim j}^l}, \quad (8.26)$$

where  $g_{i \sim j}^l$  is the diffusivity between pixel  $i$  and  $j$ , and  $\mathcal{N}(i)$  is the 4-connected neighborhood around  $i$ . The method converges for overrelaxation parameters  $\omega \in (0, 2)$ . We obtained the fastest convergence rate for  $\omega = 1.85$ .

In the case of segmentation without moment constraints, there is only the constraint that  $u \in \mathcal{B}$ , where we project  $u$  such that  $u \in [0, 1]$ . This constraint can be enforced by clipping the values of  $u$  at every point  $x$  after each iteration:

$$u(x) = \begin{cases} 0 & \text{if } u(x) < 0, \\ u(x) & \text{if } 0 \leq u(x) \leq 1, \\ 1 & \text{if } 1 < u(x). \end{cases} \quad (8.27)$$

### 8.4.1 Minimization with Soft Constraints

Soft constraints are implemented by adding additional terms to the cost function, as has been done in (8.13), (8.17) and (8.20). Minimization is then performed by computing the corresponding Euler–Lagrange equations with the additional terms. This leads to solutions that prefer shapes that fulfill the constraints, but do not necessarily exactly fulfill them.

For the area constraint, the corresponding Euler–Lagrange equation is computed by derivation of (8.13). We obtain the equation system:

$$0 = \operatorname{div}\left(g \frac{\nabla u}{|\nabla u|}\right) - f - 2\lambda_0 \left(\int_{\Omega} u - \frac{c}{|\Omega|} dx\right). \quad (8.28)$$

Note that the area update term for  $u$  is equal for all points in  $\Omega$  since the last term is completely independent of  $x$ . A positive value is added in the case that the area of the current segmentation is smaller than the desired area constraint, and a negative value in the case that it is greater. The term is zero in the case that the constraint is exactly fulfilled. The constraint parameter  $c$  is divided by the size of the image domain  $\int_{\Omega} dx = |\Omega|$  in order to equally distribute the update value over all points.

The centroid soft constraint (8.17) yields an Euler–Lagrange equation where the update term for the centroid constraint is dependent on the location of  $x$ :

$$0 = \operatorname{div}\left(g \frac{\nabla u}{|\nabla u|}\right) - f - 2 \sum_{i=1}^d \lambda_{1i} \left(\int_{\Omega} (\mu_i - x_i)u dx (\mu_i - x_i)\right), \quad (8.29)$$

where  $\mu_i$  and  $x_i$  denote the  $i$ th elements of vectors  $\mu$  and  $x$ . Note that it yields  $d$  additional terms to the equation system, as well as  $d$  additional optimization parameters  $\lambda_{11}, \dots, \lambda_{1d}$ .

Similar functions can be derived for the soft covariance and higher order moment constraints. The corresponding Euler–Lagrange term for an arbitrary moment of order  $k$  gives the following more general formulation:

$$0 = -2 \sum_{i_1=1}^d \dots \sum_{i_k=1}^d \lambda_{k i_1 i_k} \left[ \int_{\Omega} \left( a_{i_1 \dots i_k} - \prod_{l=1}^k (\mu_{i_l} - x_{i_l}) \right) u \, dx \left( a_{i_1 \dots i_k} - \prod_{l=1}^k (\mu_{i_l} - x_{i_l}) \right) \right].$$

In this formulation, some terms appear multiple times. This is due to the fact that for all permutations  $\sigma$  of  $i_1 \dots i_k$  we have  $a_{i_1 \dots i_k} = a_{\sigma(i_1) \dots \sigma(i_k)}$ , because of the symmetry of the constraint tensors.

For all soft constraints, the additional parameters  $\lambda_{k1}, \dots, \lambda_{kd}$  have to be optimized, with  $k$  being the order of the respective moment.

### 8.4.2 Computing Projections to Moment Constraint Sets

The moment hard constraints can be implemented by projecting onto the constraint sets during optimization. In our implementation we prefer the projection method over soft constraints, because firstly the constraints can be fulfilled exactly, and secondly there are no additional parameters that need to be optimized.

In the case of moment hard constraints, we enforce all constraints during the optimization by back-projecting the current segmentation onto the constraint set after every iteration. Segmentations  $u$  are projected onto the intersection of two convex sets:

1. the convex set  $[0, 1]$  and
2. the intersection of the respective moment constraints.

Since all moment constraints presented in this chapter are linear in  $u$  an orthogonal projection can be directly computed for all combinations of moment constraints. Hence, iterative projections are needed between two sets only—Independent of the order or the number of moment constraints. In the following, we will show how these projections can be easily computed and implemented. A special case arises in the case of the area constraint, because the projection can be computed in a single step. In all other cases, we use the algorithm of [2] to iteratively project onto the intersection of the convex sets.

#### 8.4.2.1 Projection to Area Constraint

The projection onto the range  $[0, 1]$  and onto the area constraint in (8.11) can be combined in one step.

For any  $u_0 \in \mathcal{B}$ , the projection  $u$  onto those constraints has to solve the convex program

$$\begin{aligned} u &= \arg \min_{v \in \mathcal{B}} \frac{1}{2} \|v - u_0\|_2^2 \\ \text{s.t. } & v(x) - 1 \leq 0 \quad \forall x \\ & -v(x) \leq 0 \quad \forall x \\ & \|v\|_1 - c = 0 \end{aligned}$$

By means of the Karush–Kuhn–Tucker conditions for convex problems, if we can find the functions  $u$ ,  $\xi_1$ , and  $\xi_0$  and a scalar  $v$  that fulfil the following conditions

$$u(x) - u_0(x) + \xi_1(x) - \xi_0(x) + v = 0 \quad \forall x \quad (8.30)$$

$$u(x) \leq 1 \wedge \xi_1(x) \geq 0 \quad \forall x \quad (8.31)$$

$$u(x) \geq 0 \wedge \xi_0(x) \geq 0 \quad \forall x \quad (8.32)$$

$$\xi_1(x) = 0 \vee u(x) = 1 \quad \forall x \quad (8.33)$$

$$\xi_0(x) = 0 \vee u(x) = 0 \quad \forall x \quad (8.34)$$

$$\|u\|_1 = c, \quad (8.35)$$

then  $u$  is the correct projection.

With the following method we can find a solution for the conditions above:

1. Set  $u := u_0$ ,  $\xi_1 := 0$ ,  $\xi_0 := 0$ ,  $v := 0$
2. On all positions where  $u(x) > 1$ , set  $u(x) := 1$ ,  $\xi_1(x) := u_0(x) - 1$
3. On all positions where  $u(x) < 0$ , set  $u(x) := 0$ ,  $\xi_0(x) := -u_0(x)$

All constraints but the area constraint are now fulfilled.

Without loss of generality, let the projection of  $u_0$  to  $[0, 1]$  have a smaller norm than  $c$ . If not, set  $u_2(x) := 1 - u(x)$  and  $c_2 := \|\Omega\| - c$ , switch  $\xi_0$  and  $\xi_1$ , perform the projection onto the volume  $c_2$  as described in the following, and afterwards again reflect  $u_2$  at 1 and switch  $\xi_0$  and  $\xi_1$ .

To fulfill the area constraint (8.35), we can now raise the sum  $u(x) + \xi_1(x) - \xi_0(x)$  by the same amount in all pixels, and adjust  $v$  such that (8.30) is fulfilled. If  $0 < u(x) < 1$ , raise  $u(x)$ , if  $u(x) = 1$ , raise  $\xi_1(x)$ , if  $\xi_0(x) > 0$ , lower  $\xi_0(x)$ .

Unfortunately, the difference  $v$  is nontrivial, because the area constraint (8.35) depends on  $u$ , not on  $u + \xi_1 - \xi_0$ . Therefore, we have to employ Algorithm 1: Afterwards, we have to adjust the last iteration, if  $\Delta_c < 0$ . This algorithm computes the desired projection and it does not require to explicitly store  $\xi_0$  and  $\xi_1$ , as the sum in (8.30) can be stored in one field  $u$  and the checks in the algorithm above can be performed on  $u$  as well. Unfortunately though, the algorithm requires several  $O(n)$  computations in the regression to find the minima and in the update steps.

However, if we sort the discrete pixel positions by their value  $u_0$ , and remap them after the projection, the projection can be performed very easily with Algorithm 2. Let us assume now, that  $u_0$  is stacked to a vector and monotonically decreasing in  $x$  from pixel  $x = 1$  to pixel  $x = |\Omega|$ .

Figure 8.4 demonstrates the input and output of Algorithm 2, assuming that  $u_0$  and  $u$  are sorted in decreasing order. The optimal value  $v$  that needs to be added to  $u_0$  in order to fulfill both the area constraint  $\int u \, dx = c$  and the constraint  $u \in [0, 1]$ , is computed in one step of Algorithm 2.

After Algorithm 2 terminates, we simply subtract  $v$  from  $u_0(x)$  in every pixel to get  $u$ , clamp  $u$  to  $[0, 1]$  and put the difference into the (virtual)  $\xi_0$  and  $\xi_1$ .

**Algorithm 1** Simple projection algorithm

---

```

 $v \leftarrow 0$ 
 $\Delta_c \leftarrow c - \|u\|$ 
while  $\Delta_c > 0$  do
     $\Delta_u \leftarrow \min\{\min_{\{x \in \Omega | \xi_0(x) > 0\}}\{\xi_0(x)\}, \min_{\{x \in \Omega | u(x) < 1\}}\{1 - u(x)\}\}$ 
     $\Delta_k \leftarrow 0$ 
    for all  $x \in \Omega$  do
        if  $\xi_0(x) > 0$  then
             $\xi_0(x) \leftarrow \xi_0(x) - \Delta_u$ 
        else if  $u(x) < 1$  then
             $u(x) \leftarrow u(x) + \Delta_u$ 
             $\Delta_k \leftarrow \Delta_k + \Delta_u$ 
        else
             $\xi_1(x) \leftarrow \xi_1(x) + \Delta_u$ 
        end if
         $v \leftarrow v + \Delta_u$ 
    end for
     $\Delta_c \leftarrow \Delta_c - \Delta_k$ 
end while

```

---

**Algorithm 2** Projection algorithm with sorting

---

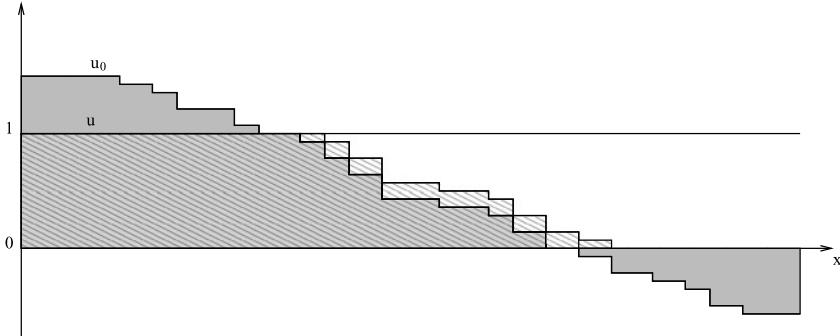
```

 $v \leftarrow 0$ 
 $\Delta_c \leftarrow c - \|u_0\|$ 
 $x_1 \leftarrow \min\{x | u_0(x) < 1\}$ 
 $x_2 \leftarrow \min\{x | u_0(x) < 0\}$ 
while  $\Delta_c > 0$  do
     $\Delta_1 \leftarrow 1 - u_0(x_1) + v$ 
     $\Delta_2 \leftarrow -u_0(x_2) + v$ 
    if  $\Delta_1 > \Delta_2$  then
         $v \leftarrow v - \min\{\Delta_2, \frac{\Delta_c}{x_2 - x_1}\}$ 
         $\Delta_c \leftarrow \Delta_c - \min\{(x_2 - x_1)\Delta_2, \Delta_c\}$ 
         $x_2 \leftarrow \min\{x | u_0(x) - v < 0\}$ 
    else
         $v \leftarrow v - \min\{\Delta_1, \frac{\Delta_c}{x_2 - x_1}\}$ 
         $\Delta_c \leftarrow \Delta_c - \min\{(x_2 - x_1)\Delta_1, \Delta_c\}$ 
         $x_1 \leftarrow \min\{x | u_0(x) - v < 1\}$ 
    end if
end while

```

---

Now all KKT conditions are fulfilled and  $u$  is the desired projection. Note that every assignment in the algorithm is performed at most  $2n$  times and the algorithm only requires constant memory. Therefore, its complexity is dominated by the sorting algorithm.



**Fig. 8.4** Area of  $u$  before and after applying Algorithm 2. The *gray area* denotes the area before the projection: neither the area constraint  $\int u \, dx = c$  nor the constraint  $u \in [0, 1]$  is fulfilled. The *shaded area* shows the area of  $u$  after projection: both constraints are fulfilled after two steps of the algorithm

#### 8.4.2.2 Projection to Higher Order Moment Constraints

For moments of an order higher than 0, we use the method of [2] to iteratively compute projections of  $u$  onto the intersection of the convex set  $[0, 1]$  and the linear moment constraints. The algorithm of [2] projects to the intersection of convex sets by alternatingly projecting to the respective sets. The correct solution is found by remembering the previous projection for each step. This leads to an algorithm that converges to the solution, although slow.

We can summarize all projections to an arbitrary number of moment constraints in one projection step because all moment constraints are linear sets. Hence, we project between the two sets  $[0, 1]$  and the intersection of the moment constraints.

**Centroid Constraint** We will in the following show the projection formula for the centroid constraint.

An equivalent formulation of (8.14) is the constraint set

$$\mathcal{C}_1 = \left\{ u \in \mathcal{B} \mid \int_{\Omega} (\mu_i - x_i) u \, dx = 0, \forall 1 \leq i \leq d \right\}. \quad (8.36)$$

We are looking for the orthogonal projection  $\hat{m}$  that projects  $u$  to the nearest  $\hat{u}$  in  $\mathcal{C}_1$  s.t.  $\hat{u} = u + \hat{m}$ . Since  $\mathcal{C}_1$  is linear in  $u$  this projection is unique. We have

$$\hat{m} = \arg \min_{m \in \mathcal{M}} \|m\| \quad (8.37)$$

with

$$\mathcal{M} = \left\{ m \in \mathcal{B} \mid \int_{\Omega} (\mu_i - x_i) m \, dx = - \int_{\Omega} (\mu_i - x_i) u \, dx, \forall 1 \leq i \leq d \right\}. \quad (8.38)$$

The left-hand sides of the constraint equations in  $\mathcal{M}$  are linear in  $m$ , and the right-hand sides are constants (independent of  $m$ ).

Then the following projection theorem [18] holds.

**Theorem 8.2** *The orthogonal projection  $\hat{m}$  (projection of minimum norm) to a set of constraints  $(m, y_i) = c_i, \forall 1 \leq i \leq d$  is given by*

$$\hat{m} = \sum_{i=1}^d \beta_i y_i \quad (8.39)$$

where the coefficients  $\beta_i \in \mathbb{R}$  are the solution to the linear equation system

$$(y_1, y_i)\beta_1 + \cdots + (y_d, y_i)\beta_d = c_i. \quad (8.40)$$

*Proof* A proof can be found in [18].  $\square$

In our case, the  $y_i$  and  $c_i$  are given by

$$y_i = \mu_i - x_i, \quad c_i = - \int_{\Omega} (\mu_i - x_i) u \, dx, \quad \forall 1 \leq i \leq d, \quad (8.41)$$

reducing the problem to solving a linear system of equations of the size  $d \times d$  to obtain the coefficients  $\beta_1, \dots, \beta_d$  and the resulting projection is

$$\hat{m} = \sum_{i=1}^d \beta_i (\mu_i - x_i). \quad (8.42)$$

This means in particular, that points that have a high distance to the constraint centroid  $\mu$ , get assigned a higher value to change. The reason is the orthogonal projection that finds a projection where the smallest change in  $u$  is achieved that fulfills the constraint. Points with a high distance to  $\mu$  contribute higher values to the sum (8.42).

**Covariance Constraint** Similarly, we can compute the projection to the covariance constraint which yields

$$\hat{m} = \sum_{i=1}^d \sum_{j=1}^d \beta_{ij} (a_{ij} - (\mu_i - x_i)(\mu_j - x_j)), \quad (8.43)$$

where  $a_{ij}$  are the entries of constraint matrix  $A$ . Here we solve a  $d^2 \times d^2$  system of linear equations to obtain the coefficients  $\beta_{ij}$ . If we exploit the symmetry of  $A$  and merge terms that appear duplicate times in the sum, the number of coefficients reduces to

$$d' = \sum_{i=1}^d \sum_{j=i}^d 1 = \frac{1}{2}d(d+1), \quad (8.44)$$

which is still in the same complexity class  $O(d^2)$ , however reduces the number of coefficients that need to be computed by a factor of almost 2.

**Moment Constraints of Arbitrary Order** Higher order moment constraints and combinations of different order moment constraints can be computed in an analogous way.

The general projection for a moment of order  $k$  can be computed by

$$\hat{m} = \sum_{i_1=1}^d \dots \sum_{i_k=1}^d \beta_{i_1 \dots i_k} \left( a_{i_1 \dots i_k} - \prod_{l=1}^k (\mu_{il} - x_{il}) \right). \quad (8.45)$$

The corresponding linear equation system has  $d^k$  unknowns  $\beta_{i_1 \dots i_k}$ .

### 8.4.3 Optimality Bound

Unfortunately, the threshold theorem [4] guaranteeing optimality for the unconstrained binary labeling problem does not generalize to the constrained optimization problems considered here. Nevertheless, we can prove the following optimality bound:

**Proposition 8.6** *Let  $u^* = \arg \min_{u \in \mathcal{C}} E(u)$  be a minimizer of the relaxed problem and  $E_{\text{opt}}$  the (unknown) minimum of the corresponding binary problem. Then any thresholded version  $\hat{u}$  of the relaxed solution  $u^*$  is within a computable bound of the optimum  $E_{\text{opt}}$ .*

*Proof* Since  $E_{\text{opt}}$  lies energetically in between the minimum of the relaxed problem and the energy of the thresholded version, we have:

$$E(\hat{u}) - E_{\text{opt}} \leq E(\hat{u}) - E(u^*). \quad (8.46)$$

□

For all experiments in this paper, this bound is on average around 5 %. How to assure that the binarized version still exactly fulfills the moment constraints remains an open challenge.

## 8.5 Moment Constraints for Object Tracking

Object tracking over a sequence of images is the problem of finding the shape of an object that was given in the first frame in the subsequent frames of the sequence. This section shows how segmentation with moment constraints can be generalized with a few modifications to a method for object tracking.

Constraining the moments of a shape during a sequence of images leads—in combination with the presented method for image segmentation—to a method for object tracking. Given the moments of the shape in the first frame, we can constrain these moments for all subsequent frames as well.  $u$  is now a function defined on the image plane  $\Omega$  evolving over time  $T \subseteq \mathbb{R}^+$ , that is,  $u : \Omega \times T \rightarrow [0, 1]$ .

First, we will show how the area constraint can be used to track an object over time: We assume that the area of the shape should not change over time, and therefore constrain the area of a shape at time  $t$  to be equal to the area of the shape in the previous frame  $t - 1$ :

$$\min_{u \in \mathcal{B}} E(u) \quad \text{s.t.} \quad \left| \int_{\Omega} u(x, t) dx - c_{t-1} \right| = 0, \quad \forall t \in T, \quad (8.47)$$

where  $c_{t-1}$  is the area of the shape in time frame  $t - 1$  and we define  $c_{-1}$  as the area of the ellipse drawn by the user in the first frame.

Alternatively, we can allow for a small change of area and thereby model motion of the object towards or away from the camera (or the camera towards/away from the object) by replacing the  $= 0$  in (8.47) by  $\leq v$  with a constant value  $v \in \mathbb{R}$ . Here,  $v$  corresponds to the amount that the area of the shape is allowed to change from one frame to the next, that is, depends on the velocity of the motion.

Equally, we can impose a constraint that the object should not move too far from one frame to the other: The centroid of the shape should not change more than a given value  $v$  which corresponds to the velocity of the object:

$$\min_{u \in \mathcal{B}} E(u) \quad \text{s.t.} \quad \left| \frac{\int_{\Omega} xu(x, t) dx}{\int_{\Omega} u(x, t) dx} - \mu_{t-1} \right| \leq v, \quad \forall t \in T, \quad (8.48)$$

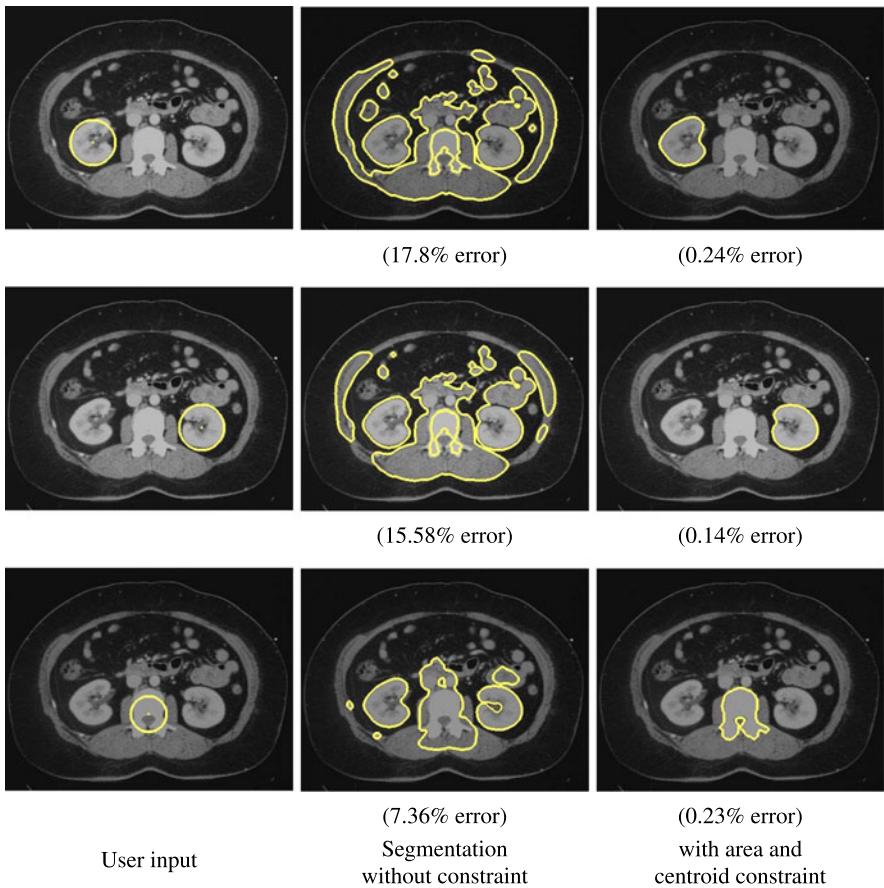
where  $v$  corresponds to the maximum length of the vector between the centroid in one frame and the centroid in the next frame, and  $\mu_{t-1}$  is the centroid of  $u$  in the previous frame. Again, we define  $\mu_{-1}$  as the centroid of the ellipse drawn by the user in the first frame in order to obtain an initialization for  $t = 0$ .

Similar constraints can be imposed on the covariance and higher order moment constraints. For the covariance constraint, for example, rotation of the object can be constrained by assuming that the covariance matrix should not change more than a given value.

Optimization can be performed with the same method as explained in Sect. 8.4 while the respective moment constraints are updated in each frame.

## 8.6 Experimental Results

In this section, we present a qualitative and quantitative evaluation of the proposed method on medical imagery and other real-world images and videos. For all experiments, we use  $g(x) = 1$  and  $f(x) = \log(p_{\text{bg}}(I(x))/p_{\text{obj}}(I(x)))$  with input image  $I : \Omega \rightarrow \mathbb{R}$ . We compute the likelihoods  $p_{\text{obj}}$  and  $p_{\text{bg}}$  using color or gray-scale histograms from inside and outside regions defined by the user input. Respective moment constraints on centroid, area or covariance structure are easily imposed by simple mouse interactions. Solutions to the constrained convex optimization problems are computed on the fly. In all experiments shown, moment constraints are enforced by iteratively projecting solutions to the respective constraint sets after each iteration of optimization. Typical run-times on the GPU are around 1 second for an image of the size  $300 \times 400$ .



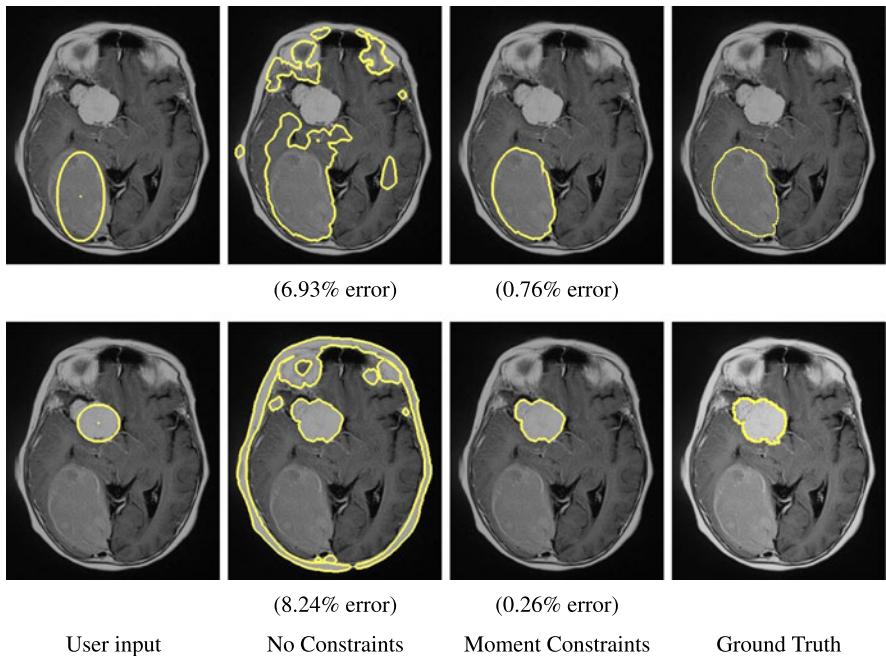
**Fig. 8.5** Segmentation of a CT image with kidneys and spine. The centroid and area constraints enable the user to specify the approximate location and size of the desired object that should be segmented. Imposing these moment constraints during optimization leads to drastic improvements in the segmentation

### **8.6.1 Quantitative Evaluation on Medical Images**

We evaluate segmentations with and without moment constraints on a set of medical images and quantitatively compare the results to a manually labelled ground truth.

### 8.6.1.1 Area and Centroid Constraints

Figure 8.5 shows a comparison of segmentation with and without a constraint on the area and centroid for a CT image of kidneys and spine: without constraints no shape information is taken into account for the segmentation, resulting in a segmentation that includes many different regions. Enabling the area and centroid con-



**Fig. 8.6** Tumor extraction in brain MR images using segmentation with and without constraints on covariance and area. While the algorithm does not require any local boundary information, constraining its second order moments by a simple user interaction suffices to generate the desired segmentation

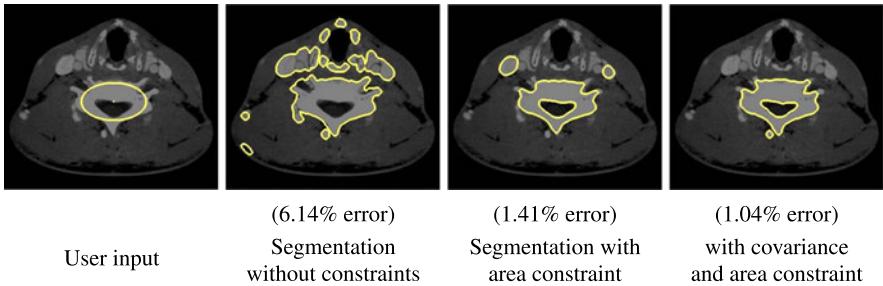
straints leads to segmentations that prefer the center and the size of the circle that was clicked by the user. This leads to substantial improvements of the segmentations without affecting the fine-scale boundary estimation.

### 8.6.1.2 Higher Order Constraints

More sophisticated structures can be specified with higher order moments. Since covariance matrices can be represented by ellipsoids, an intuitive user input is achieved by clicking an ellipse with the mouse. The axes of the ellipse define the entries of the corresponding covariance matrix, while the center and area of the ellipse define the centroid and area constraints. Figures 8.6 and 8.7 show segmentations with and without constraints resulting from user defined ellipses describing the approximated size, location and shape of the desired object.

### 8.6.1.3 Quantitative Performance Evaluation

Clearly, the user-specified moment constraints allow to visibly improve the segmentation. To quantify this improvement, Table 8.1 shows average relative errors (i.e.



**Fig. 8.7** Segmentation without and with constraints for a CT image of the neck. Constraining the area yields a segmentation which prefers the size of the ellipse that was clicked by the user, resulting in less incorrectly labeled pixels, compared to the segmentation without constraints. The covariance constraint additionally considers the dimensions of the ellipse yielding an even more accurate segmentation. Again, the convex constraints merely constrain respective moments of the solution leading to drastic improvements of the segmentation results

**Table 8.1** Average relative errors with standard deviations for segmentation without and with moment constraints

	Average relative error
segmentation without constraint	$12.02\% \pm 0.89\%$
with area constraint	$2.36\% \pm 0.11\%$
with area and centroid constraint	$0.41\% \pm 0.05\%$
with area, centroid and covariance	$0.35\% \pm 0.09\%$

the percentage of incorrectly labeled pixels per image) with standard deviations for an evaluation of the segmentation without constraint, with area constraint only, and with area and centroid and covariance constraint, respectively. Some of the images that were used for the tests and their segmentations are shown in Figs. 8.5, 8.6 and 8.7. The table shows that the use of these rather simple and easy to transmit constraints yield a reduction of incorrectly classified pixels by a factor of about 10.

### 8.6.2 Computation Times for Moment Constraint Projections

Figure 8.9 shows run times in seconds for projections onto the presented moment hard constraints. We computed the number of seconds that were needed on a 3.4 GHz Intel Core i7-2600 CPU to compute one projection onto all moment constraints of order 0 to  $k$  for  $k \in \{0, \dots, 30\}$  using the projection formula (8.45). The respective constraints were obtained by computing the moments of the manually segmented reference shape shown in Fig. 8.3(a). While projections onto the lower order moment constraints are computed in just a few milliseconds, the figure implies exponentially growing run time for an increasing number of constraints.

We conclude that segmentation with moment constraints is applicable to real-time tasks for the lower order moments, whereas higher order moments that can theoretically constrain arbitrary shapes, are applicable only to off-line tasks.

### 8.6.3 Segmentation of Real-World Images

Figure 8.8 shows how moment constraints can improve segmentation of non-medical images. Here the data term is based on RGB histograms. The purely color-based segmentations without moment constraints shown in the second column demonstrate that the color distributions of respective objects are not sufficiently different to discriminate the objects of interest. The third column of Fig. 8.8 shows the segmentation results with constraints on area, centroid and covariance. All moment constraints are extracted from the user-specified ellipse, allowing a deviation of 10 % for each constraint to handle imprecise user input. The moment constraints allow to quickly disambiguate the color information leading to substantial improvements of the segmentation.

### 8.6.4 Optimality Bounds

The threshold theorem [4] states that in the unconstrained case the thresholded version is a globally optimal solution of the binary energy (8.4). Since this is not the case when we impose additional moment constraints on the segmentation, we analyzed the distance of the continuous result to the thresholded version. To this end, we compute the relative energy bound

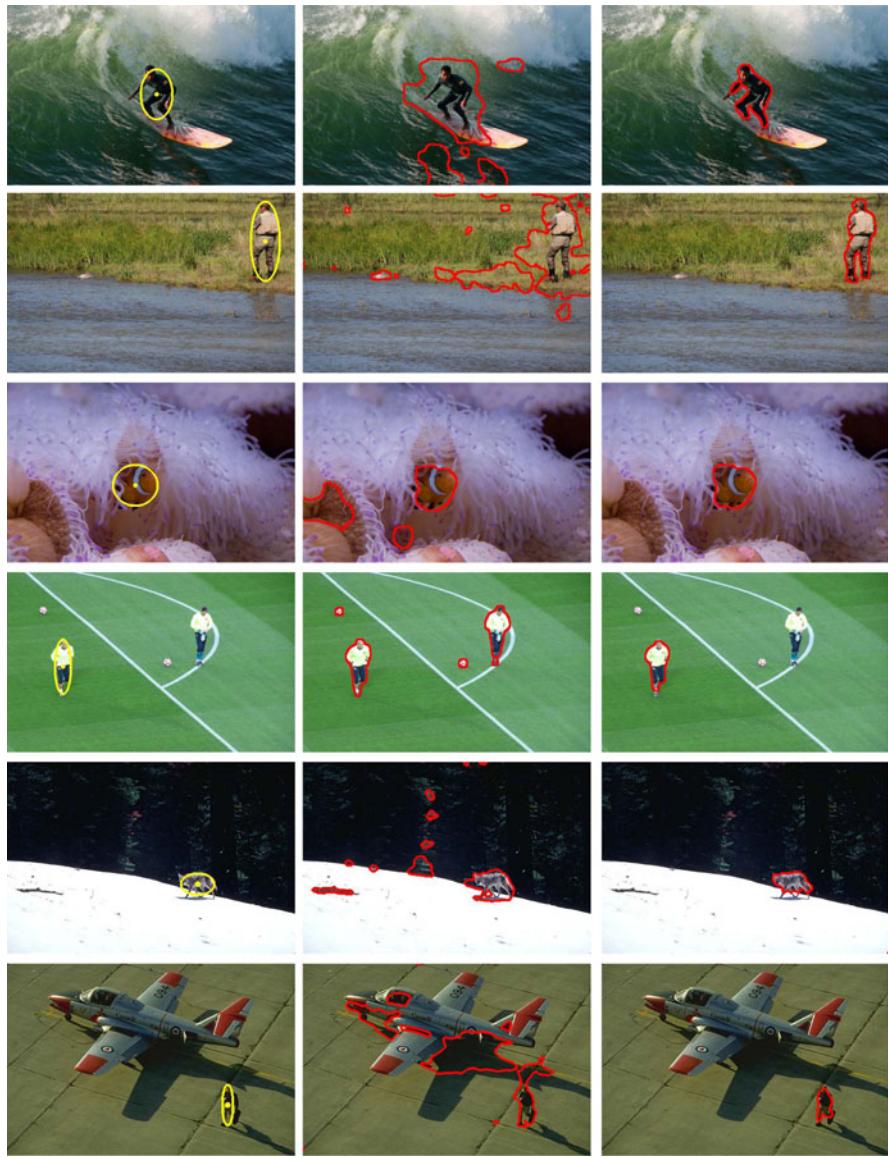
$$\frac{E(\hat{u}) - E(u^*)}{E(u^*)}, \quad (8.49)$$

where  $u^*$  is the continuous solution before thresholding and  $\hat{u}$  is the solution after thresholding. Figure 8.10 shows segmentation results for  $u$  before thresholding. The percent values below the images refers to the energy bound (8.49). In all four experiments that are shown in the figure, the distance to the thresholded version is below 1 %. In our experiments, we further observed that the more constraints are imposed on a segmentation, the larger the distance to the thresholded version increases.

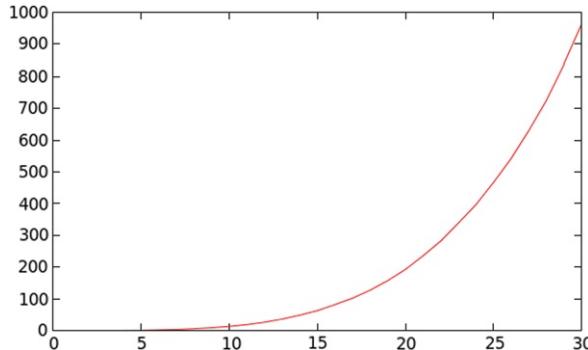
### 8.6.5 Comparison to Segmentation with User Scribbles

Many interactive segmentation methods that utilize manual input from the user implement scribbles to obtain initial values for the histograms that generate the data term  $f$ . Scribbles in this context are arbitrary pixels marked by the user with the mouse while different labels are used for foreground and background pixels. The methods are implemented in discrete, for example [22], or continuous, for example [24], settings.

Figure 8.11 shows a comparison of two segmentation methods with priors on color and location from user input: first the presented method with moment constraints and second a segmentation method based on user scribbles. The purpose of



**Fig. 8.8** Segmentation of real world images without and with moment constraints. **(a)** The user marks an ellipse at the approximate size and location of the object with two mouse clicks. Color histograms for foreground and background are determined from the inside and outside of the ellipse, respectively. **(b)** Segmentation results without constraints using only the histograms. **(c)** Segmentation results with constraints on area, centroid and covariance. Parameters of the constraints are derived from the ellipse marked by the user.



**Fig. 8.9** Run time (in seconds) vs. number of moment constraints  $k$  in projection. The plot shows the number of seconds needed to compute a projection onto the moments of order 0 to  $k$ . While lower order moment constraints are computed in a few milliseconds, computation times appear to grow exponentially for increasing numbers of constraints. Fortunately, for interactive segmentation applications one is mostly interested in constraining the low-order moments only, leaving the fine-scale details to be determined by the image data

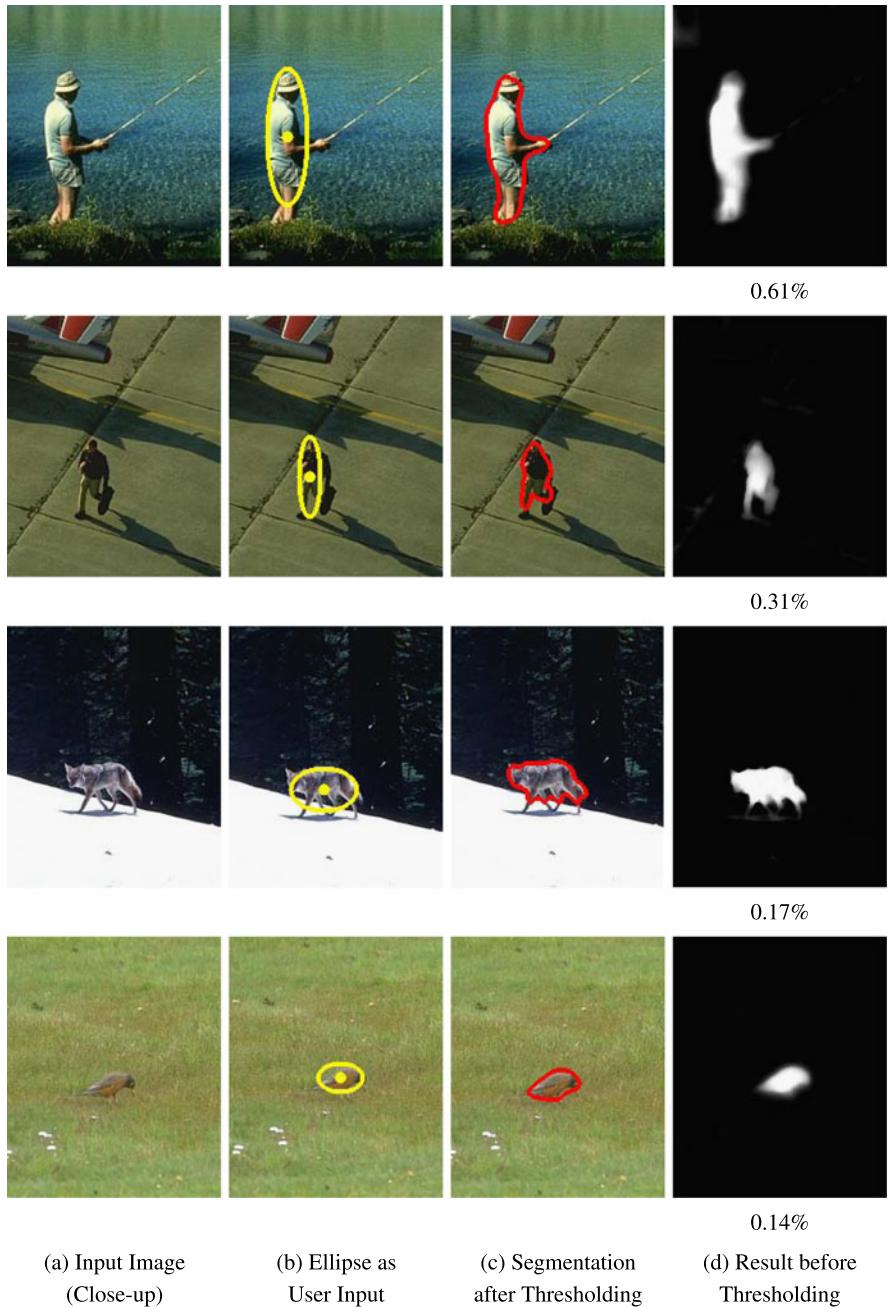
this experiment was to compare the effort that needs to be brought in by the user in order to sufficiently segment the respective object of interest from the background. For the scribble segmentation the user draws mouse strokes in two different colors: blue for foreground and green for background. Histograms for the data term  $f$  are computed from these marked pixels. Segmentation results with user scribbles were computed by minimizing functional (8.4), using the same method and same parameters as in the case of the moment constraints, except that the moment constraints were replaced by a constraint that all pixels that were marked as foreground by the user belong to the segmented region, and that the background pixels are not contained by the segmented region.

The figure shows that segmentations with user scribbles need much more mouse interaction compared to segmentation with moment constraints and therefore are able to substantially simplify the task of image segmentation for the user.

### 8.6.6 Tracking with Moment Constraints

Figures 8.12 and 8.13 show how the proposed method can be applied to tracking objects in videos. As can be seen in Fig. 8.13, the purely color-based segmentation does not suffice to correctly segment object from background in the case of non-unique color distributions.

We impose shape information by constraining the low order moments (area, centroid and covariance) throughout the entire image sequence. As can be seen in the first image of each sequence, the user initializes the method with two mouse clicks: an ellipse of the approximate size and location of the object is drawn on the first frame of the sequence. This is sufficient user input, since histograms and moment



**Fig. 8.10** Segmentation with the convex moment constraints converges to nearly binary solutions, making the method robust to the chosen threshold. The percentage below the images in (d) refers to the distance between the continuous result and its corresponding thresholded version



**Fig. 8.11** Comparison of segmentation with moment constraints and user scribbles: Segmentation with user scribbles needs more mouse interaction to obtain similar results, since the implementation with moment constraints needs just two mouse clicks for drawing the respective ellipse



**Fig. 8.12** Moment constraints for object tracking. The user initializes the tracking by clicking an ellipse in the *first frame*, the moments of which constrain the segmentation in subsequent images. A small deviation of the centroid is allowed to track the moving object. Note that this approach is generic, as no shapes have to be previously learned



**Fig. 8.13** Moment constraints for object tracking. The user initializes the tracking in form of an ellipse in the *first frame (left)*, from which histograms and constraint parameters are derived. The *first row* shows results with moment constraints, where a deviation of the centroid is allowed from each frame to the next one to account for the object’s motion. The *second row* shows results of histogram based tracking without constraints. This comparison shows that moment constraints can realize acceptable real-world object tracking with no previous learning of shapes

constraint parameters are derived from the ellipse: again, histograms for foreground and background are computed from the inside and outside of the ellipse, respectively, and the constraint parameters for area, centroid and covariance are derived from the ellipse’s area, center point and principal axes. The subsequent frames of the video use the histograms and moment constraints from the first frame, allowing a small deviation of the centroid from each frame to the next, which corresponds to a constraint on the maximum velocity. Since no previous learning of shapes is necessary, the approach naturally applies to arbitrary object shapes.

## 8.7 Conclusion

In this chapter, we proposed the use of moment constraints in a convex shape optimization framework. In particular, we showed that for an entire family of constraints on the area, the centroid, the covariance structure of the shape and respective higher-order moments, the feasible constraint sets are all convex. While we cannot guarantee global optimality of the arising segmentations, all computed solutions are independent of initialization and within a known bound of the optimum. In both qualitative and quantitative experiments on interactive image segmentation, we demonstrated that respective moment constraints are easily imposed by the user and lead to drastic improvements of the segmentation results, reducing the average segmentation error from 12 % to 0.35 %. In contrast to existing works on shape priors in segmentation, the use of low-order moment constraints does not require shape learning and is easily applied to arbitrary shapes since the recovery of fine scale shape details is not affected through the moment constraints. Efficient GPU-accelerated PDE solvers allow for computation times of about one second for images of size  $300 \times 400$ , making this a practical tool for interactive image segmentation.

## References

1. Blake A, Zisserman A (1987) Visual reconstruction. MIT Press, Cambridge
2. Boyle JP, Dykstra RL (1986) An method for finding projections onto the intersection of convex sets in Hilbert spaces. In: Lecture Notes in Statistics, vol 37, pp 28–47
3. Caselles V, Kimmel R, Sapiro G (1995) Geodesic active contours. In: Proc IEEE int conf on computer vision, Boston, USA, pp 694–699
4. 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
5. Cremers D, Osher SJ, Soatto S (2006) Kernel density estimation and intrinsic alignment for shape priors in level set segmentation. Int J Comput Vis 69(3):335–351
6. Das P, Veksler O, Zavadsky V, Boykov Y (2008) Semiautomatic segmentation with compact shape prior. Image Vis Comput 27(1–2):206–219
7. Etyngier P, Segonne F, Keriven R (2007) Shape priors using manifold learning techniques. In: IEEE int conf on computer vision, Rio de Janeiro, October 2007
8. Foulonneau A, Charbonnier P, Heitz F (2006) Affine-invariant geometric shape priors for region-based active contours. IEEE Trans Pattern Anal Mach Intell 28(8):1352–1357
9. Greig DM, Porteous BT, Seheult AH (1989) Exact maximum a posteriori estimation for binary images. J R Stat Soc B 51(2):271–279
10. Grenander U, Chow Y, Keenan DM (1991) Hands: a pattern theoretic study of biological shapes. Springer, New York
11. Ising E (1925) Beitrag zur theorie des ferromagnetismus. Z Phys 23:253–258
12. Kass M, Witkin A, Terzopoulos D (1988) Snakes: active contour models. Int J Comput Vis 1(4):321–331
13. Kichenassamy S, Kumar A, Olver PJ, Tannenbaum A, Yezzi AJ (1995) Gradient flows and geometric active contour models. In: IEEE int conf on computer vision, pp 810–815
14. Klodt M, Cremers D (2011) A convex framework for image segmentation with moment constraints. In: IEEE int conf on computer vision, Barcelona, Spain
15. Kolev K, Cremers D (2008) Integration of multiview stereo and silhouettes via convex functionals on convex domains. In: European conference on computer vision (ECCV), Marseille, France, October 2008
16. Kolev K, Klodt M, Brox T, Cremers D (2009) Continuous global optimization in multiview 3d reconstruction. Int J Comput Vis 84(1):80–96
17. Lempitsky V, Kohli P, Rother C, Sharp T (2009) Image segmentation with a bounding box prior. In: IEEE int conf on computer vision, Kyoto, Japan
18. Luenberger DG (1997) Optimization by vector space methods, 1st edn. Wiley, New York
19. Mumford D, Shah J (1989) Optimal approximations by piecewise smooth functions and associated variational problems. Commun Pure Appl Math 42:577–685
20. Osher SJ, Sethian JA (1988) Fronts propagation with curvature dependent speed: algorithms based on Hamilton–Jacobi formulations. J Comp Physiol 79:12–49
21. Papoulis A, Pillai SU (2002) Probability, random variables, and stochastic processes, 4th edn. McGraw-Hill, New York
22. Rother C, Kolmogorov V, Blake A (2004) GrabCut: interactive foreground extraction using iterated graph cuts. ACM Trans Graph 23(3):309–314
23. Schoenemann T, Cremers D (2009) A combinatorial solution for model-based image segmentation and real-time tracking. IEEE Trans Pattern Anal Mach Intell 32(7):1153–1164
24. Unger M, Pock T, Cremers D, Bischof H (2008) Tvsseg—interactive total variation based image segmentation. In: British machine vision conference (BMVC), Leeds, UK, September 2008
25. Veksler O (2008) Star shape prior for graph-cut image segmentation. In: Europ conf on computer vision, pp 454–467

## Chapter 9

# Large Scale Metric Learning for Distance-Based Image Classification on Open Ended Data Sets

Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka

**Abstract** Many real-life large-scale datasets are open-ended and dynamic: new images are continuously added to existing classes, new classes appear over time, and the semantics of existing classes might evolve too. Therefore, we study large-scale image classification methods that can incorporate new classes and training images continuously over time at negligible cost. To this end, we consider two distance-based classifiers, the k-nearest neighbor (k-NN) and nearest class mean (NCM) classifiers. Since the performance of distance-based classifiers heavily depends on the used distance function, we cast the problem into one of learning a low-rank metric, which is shared across all classes. For the NCM classifier, we introduce a new metric learning approach, and we also introduce an extension to allow for richer class representations.

Experiments on the ImageNet 2010 challenge dataset, which contains over one million training images of thousand classes, show that, surprisingly, the NCM classifier compares favorably to the more flexible k-NN classifier. Moreover, the NCM performance is comparable to that of linear SVMs which obtain current state-of-the-art performance. Experimentally we study the generalization performance to classes that were not used to learn the metrics. Using a metric learned on 1,000 classes, we show results for the ImageNet-10K dataset which contains 10,000 classes, and ob-

---

This chapter is largely based on previously published work [30, 31].

T. Mensink (✉) · J. Verbeek

LEAR Team – INRIA Grenoble, 655 Avenue de l’Europe, 38330 Montbonnot, France

e-mail: [thomas.mensink@inria.fr](mailto:thomas.mensink@inria.fr)

J. Verbeek

e-mail: [jakob.verbeek@inria.fr](mailto:jakob.verbeek@inria.fr)

F. Perronnin · G. Csurka

Xerox Research Centre Europe, 6 chemin de Maupertuis, 38240 Meylan, France

F. Perronnin

e-mail: [florent.perronnin@xrce.xerox.com](mailto:florent.perronnin@xrce.xerox.com)

G. Csurka

e-mail: [gabriela.csurka@xrce.xerox.com](mailto:gabriela.csurka@xrce.xerox.com)

tain performance that is competitive with the current state-of-the-art, while being orders of magnitude faster.

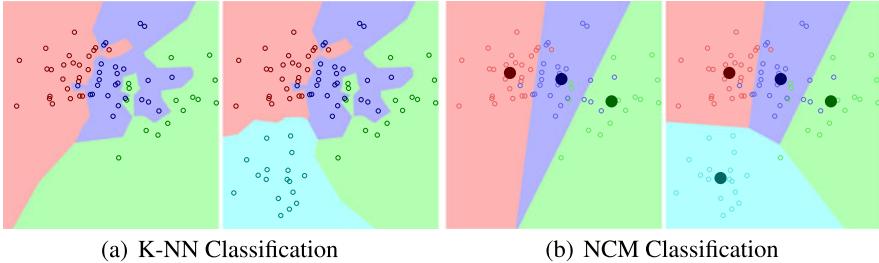
## 9.1 Introduction

In the last decade, we have witnessed an explosion in the amount of images and videos that are digitally available, for example, in broadcasting archives or social media sharing websites. Scalable automated methods are needed to handle this huge volume of data, for retrieval, annotation and visualization purposes. This need has been recognized in the computer vision research community and large-scale methods have become an active topic of research in recent years. The introduction of the ImageNet dataset [9], which contains more than 14M manually labeled images of 22K classes, has provided an important benchmark for large-scale image classification and annotation algorithms.

In this chapter, we focus on the problem of large-scale, multi-class image classification, where the goal is to assign automatically an image to one class out of a finite set of alternatives, for example, the name of main object appearing in the image, or a general label like the scene type of the image. The predominant approach to this problem is to treat it as a classification problem. To ensure scalability, often linear classifiers such as linear SVMs are used [27, 39]. Additionally, to speed-up classification, dimension reduction techniques could be used [46], or a hierarchy of classifiers could be learned [2, 12]. Recently, impressive results have been reported on 10,000 or more classes [10, 39, 46]. For all these methods hold that they are trained by using stochastic gradient descend (SGD) algorithms, which access only a small fraction of the training data at each iteration [3].

Many real-life large-scale datasets are open-ended and dynamic: new images are continuously added to existing classes, new classes appear over time, and the semantics of existing classes might evolve too. At first glance, one-vs.-rest classifiers (such as SVMs) trained with SGD algorithms appear to be the perfect solution. Indeed, classes can be trained independently, thus enabling to easily add new classes. Also since SGD algorithms are online algorithms they can accommodate for new samples easily. However, in order to apply these algorithms successfully we need to address several challenging problems. First, it is unclear to which classifier a new training sample should be fed; surely the sample should be used for the corresponding class, however feeding it as a negative sample to all other classes is not only costly but also suboptimal [36]. Second, for state-of-the-art performance the parameters of the SVM (such as the regularizer and the balance between positive and negative samples) are set by cross validation. The optimal value of these parameters depend, among others, on the number of training samples and classes, and it is unclear how these should be adapted when applied on open ended datasets to allow for an increasing number of samples and classes over time. Therefore, we believe that standard one-vs.-rest SVM classifiers are unsuitable for this task.

In this work, as an alternative, we explore distance-based classifiers which enable the addition of new classes and new images to existing classes at (near) zero cost. These methods can be used continuously as new data becomes available, and



**Fig. 9.1** Illustration of the classification rules of k-NN (a) and NCM (b). We also illustrate the effect of adding data from a new class to the data set

additionally alternated from time to time with a computationally heavier method to learn a good metric using all available training data. In particular, we consider two distance-based classifiers.

The first is the k-nearest neighbor (k-NN) classifier, which uses all examples to represent a class, and is a highly nonlinear classifier that has shown competitive performance for image classification [10, 18, 46]. New images (of new classes) are simply added to the database, and can be used for classification without further processing, see Fig. 9.1(a) for an illustration.

The second is the nearest class mean classifier (NCM), which represents classes by their mean feature vector of its elements, see, for example, [42]. Contrary to the k-NN classifier, this is an efficient linear classifier. To incorporate new images (of new classes), the relevant class means have to be adjusted or added to the set of class means, see Fig. 9.1(b) for an illustration. In Sect. 9.3, we introduce an extension which uses several prototypes per class, allowing a trade-off between the model complexity and the computational cost of classification.

The success of these methods critically depends on the used distance functions. Therefore, we cast our classifier learning problem as one of learning a low-rank Mahalanobis distance which is shared across all classes. The dimensionality of the low-rank matrix is used as regularizer, and to improve computational and storage efficiency. In this chapter, we explore several strategies for learning such a metric. For the NCM classifier, we propose a novel metric learning algorithm based on multi-class logistic discrimination (NCMML), where a sample from a class is enforced to be closer to its class mean than to any other class mean in the projected space. We show qualitatively and quantitatively the advantages of our NCMML approach over the classical Fisher Discriminant Analysis [42]. For k-NN classification, we rely on the Large Margin Nearest Neighbor (LMNN) framework [44] and investigate two variations similar to the ideas presented in [6, 44] that significantly improve classification performance.

Most of our experiments are conducted on the ImageNet Large Scale Visual Recognition Challenge 2010 (ILSVRC'10) dataset, which consists of 1.2M training images of 1,000 classes. To apply the proposed metric learning techniques on such a large-scale dataset, we employ stochastic gradient descend (SGD) algorithms, which access only a small fraction of the training data at each iteration [3]. To allow metric

learning on high-dimensional image features of datasets that are too large to fit in memory, we use in addition product quantization [17], a data compression technique that was recently used with success for large-scale image retrieval [21] and classifier training [39]. As a baseline approach, we use the state-of-the-art approach of [39], which was also the winning entry in the 2011 edition of the challenge: Fisher vector image representations [35] are used to describe images and one-vs.-rest linear SVM classifiers are learned independently for each class.

Surprisingly, we find that the NCM classifier outperforms the more flexible k-NN classifier, after learning a low-rank metric on the Fisher vector image representations. Moreover, the NCM classifier performs on par with the SVM baseline, and results are improved when we use the extension to allow for multiple centroids per class. Further we consider, among others, the generalization performance to new classes on the ImageNet-10K dataset (which consist of 4.5M training images of 10K classes) [10], a zero-shot setting where we estimate the mean of novel classes based on related classes in the ImageNet hierarchy, and image retrieval where we use a metric learned with our NCM classifier on the ILSVRC’10 dataset, to retrieve the most similar images for a given query on the INRIA Holidays [20] or the University of Kentucky Benchmark dataset (UKB) [32].

The rest of the chapter is organized as follows. In Sect. 9.2, we discuss a selection of related work which is most relevant to this chapter. In Sect. 9.3, we introduce the NCM classifier and the NCML metric learning approach, together with an extension to use multiple centroids (NCMC). In Sect. 9.4, we review LMNN metric learning for k-NN classifiers, and present two variants. We present extensive experimental results in Sect. 9.5, analyzing different aspects of the proposed methods and comparing them to the current state-of-the-art in different application settings such as large scale image annotation, transfer learning and image retrieval. Finally, we present our conclusions in Sect. 9.6.

## 9.2 Related Work

In this section, we review related work on large-scale image classification, metric learning, and transfer learning.

### 9.2.1 Large-Scale Image Classification

The ImageNet dataset [9] has been a catalyst for research on large-scale image annotation. The current state-of-the-art [27, 39] uses efficient linear SVM classifiers trained in a one-vs.-rest manner in combination with high-dimensional bag-of-words [8, 47] or Fisher vector representations [35]. Besides one-vs.-rest training, large-scale ranking-based formulations have also been explored in [46]. Interestingly, their WSABIE approach performs joint classifier learning and dimensionality reduction of the image features. Operating in a lower-dimensional space acts as

a regularization during learning, and also reduces the cost of classifier evaluation at test time. Our proposed NCM approach also learns low-dimensional projection matrices but the weight vectors are constrained to be the projected class means. This allows for efficient addition of novel classes.

In [10, 46] k-NN classifiers were found to be competitive with linear SVM classifiers in a very large-scale setting involving 10,000 or more classes. The drawback of k-NN classifiers, however, is that they are expensive in storage and computation, since in principle all training data needs to be kept in memory and accessed to classify new images. The storage issue is also encountered when SVM classifiers are trained since all training data needs to be processed in multiple passes. Product quantization (PQ) was introduced in [21] as a lossy compression mechanism for local SIFT descriptors in a bag-of-features image retrieval system. It has been subsequently used to compress bag-of-words and Fisher vector image representations in the context of image retrieval [22] and classifier training [39]. We also exploit PQ encoding in our work to compress high-dimensional image signatures when learning our metrics.

### **9.2.2 Metric Learning**

There is a large body of literature on metric learning, but in this section we limit ourselves to highlight just several methods that learn metrics for (image) classification problems. Other methods aim at learning metrics for verification problems and essentially learn binary classifiers that threshold the learned distance to decide whether two images belong to the same class or not, see, for example, [19, 23, 33]. Yet another line of work concerns metric learning for ranking problems, that is, to learn a metric between a query and the documents in the database, for example to address text retrieval tasks as in [1].

Among those methods that learn metrics for classification, the Large Margin Nearest Neighbor (LMNN) approach of [44, 45] is specifically designed to support k-NN classification. It tries to ensure that for each image a predefined set of target neighbors from the same class are closer than samples from other classes. Since the cost function is defined over triplets of points—that can be sampled in an SGD training procedure—this method can scale to large datasets. The set of target neighbors is chosen and fixed using the  $\ell_2$  metric in the original space; this can be problematic as the  $\ell_2$  distance might be quite different from the optimal metric for image classification. Therefore, we explore two variants of LMNN that avoid using such a pre-defined set of target neighbors, similar to the ideas presented in [6, 44], both variants leading to significant improvements.

The large margin nearest local mean classifier [5] assigns a test image to a class based on the distance to the mean of its nearest neighbors in each class. This method was reported to outperform LMNN but requires computing all pairwise distances between training instances and therefore does not scale well to large datasets.

The TagProp method of [18] is a probabilistic nearest neighbor classifier; it consists in assigning weights to training samples based on their distance to the test in-

stance and in computing the class prediction by the total weight of samples of each class in a neighborhood. However, similar as [5], for training also TagProp requires pairwise distances between all training examples. Other closely related methods are metric learning by collapsing classes [14] and neighborhood component analysis [15]. As TagProp, for each data point these define weights to other data points proportional to the exponent of negative distance. In [14], the target is to learn a distance that makes the weights uniform for samples of the same class and close to zero for other samples. While in [15], the target is only to ensure that zero weight is assigned to samples from other classes. These methods also require computing distances between all pairs of data points, and therefore we do not consider any of these methods in our experiments.

Closely related to our NCMMIL metric learning approach for the NCM classifier is the LESS model of [41]. They learn a diagonal scaling matrix to modify the  $\ell_2$  distance by rescaling the data dimensions, and include an  $\ell_1$  penalty on the weights to perform feature selection. However, in their case, NCM is used to address small sample size problems in binary classification, that is, cases where there are fewer training points (tens to hundreds) than features (thousands). Our approach differs significantly in that (i) we work in a multi-class setting and (ii) we learn a low-dimensional projection which allows efficiency in large-scale.

Another closely related method is the Taxonomy-embedding method of [43], where a nearest prototype classifier is used in combination with a hierarchical cost function. Documents are embedded in a lower dimensional space in which each class is represented by a single prototype. In contrast to our approach, they use a predefined embedding of the images and learn low-dimensional classifiers, and therefore their method resembles more to the WSABIE method of [46].

The centroid-based classification method explored in [48] is also related to our method. It uses a NCM classifier and an  $\ell_2$  distance in a subspace that is orthogonal to the subspace with maximum within-class variance. To obtain the optimal subspace, it computes the first eigenvectors of the within-class covariance matrix, which has a computational cost between  $O(D^2)$  and  $O(D^3)$ , this is undesirable for high-dimensional feature vectors. Moreover, this metric is heuristically obtained, rather than directly optimized for maximum classification performance.

### 9.2.3 Transfer Learning

The term transfer learning is used to refer to methods that share information across classes during learning. Examples of transfer learning in computer vision include the use of part-based or attribute class representations. Part-based object recognition models [11] define an object as a spatial constellation of parts, and share the part detectors across different classes. Attribute-based models [24] characterize a category (e.g., a certain animal) by a combination of attributes (e.g., is yellow, has stripes, is carnivore), and share the attribute classifiers across classes. Other approaches include biasing the weight vector learned for a new class towards the weight vectors of classes that have already been trained [40]. Zero-shot learning [25] is an extreme

case of transfer learning where for a new class no training instances are available but a description is provided in terms of parts, attributes, or other relations to already learned classes. Transfer learning is related to multi-task learning, where the goal is to leverage the commonalities between several distinct but related classification problems, or classifiers learned for one type of images (e.g., ImageNet) are adapted to a new domain (e.g., imagery obtained from a robot camera), see for example, [34, 38].

In [37], various transfer learning methods were evaluated in a large-scale setting using the ILSVRC'10 dataset. They found transfer learning methods to have little added value when training images are available for all classes. In contrast, transfer learning was found to be effective in a zero-shot learning setting, where classifiers were trained for 800 classes, and performance was tested in a 200-way classification across the held-out classes.

In this chapter, we also aim at transfer learning, in the sense that we allow only a trivial amount of processing on the data of new classes (storing in a database, or averaging), and rely on a metric that was trained on other classes to recognize the new ones. In contrast to most works on transfer learning, we do not use any intermediate representation in terms of parts or attributes, nor do we train classifiers for the new classes. While also considering zero-shot learning, we further evaluate performance when combining a zero-shot model inspired by [37] with progressively more training images per class, from one up to thousands. We find that the zero-shot model provides an effective prior when a small amount of training data is available.

### 9.3 The Nearest Class Mean Classifier

The nearest class mean (NCM) classifier assigns an image to the class  $c^* \in \{1, \dots, C\}$  with the closest mean:

$$c^* = \operatorname{argmin}_{c \in \{1, \dots, C\}} d(\mathbf{x}, \boldsymbol{\mu}_c), \quad (9.1)$$

$$\boldsymbol{\mu}_c = \frac{1}{N_c} \sum_{i:y_i=c} \mathbf{x}_i, \quad (9.2)$$

where  $d(\mathbf{x}, \boldsymbol{\mu}_c)$  is the Euclidean distance between an image  $\mathbf{x}$  and the class mean  $\boldsymbol{\mu}_c$ , and  $y_i$  is the ground-truth label of image  $i$ , and  $N_c$  is the number of training images for class  $c$ . The NCM classifier is a linear classifier, which allows for efficient evaluation at test time, see Fig. 9.1(b) for an illustration.

Next, we introduce our NCM metric learning approach, and its relations to existing models. Then, we present an extension to use multiple centroids per class, which transforms the NCM into a non-linear classifier. Finally, we explore some variants of the objective which allow for smaller SGD batch sizes, and we give some insights in the critical points of the objective function.

### 9.3.1 Metric Learning for the NCM Classifier

In this section, we introduce our metric learning approach, which learns a metric by maximizing the log-likelihood of correct classification. We will refer to our method as “nearest class mean metric learning” (NCMML). In our method, we replace the Euclidean distance in NCM, Eq. (9.1), by a learned (squared) Mahalanobis distance:

$$d_M(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^\top M (\mathbf{x} - \mathbf{x}'), \quad (9.3)$$

where  $\mathbf{x}$  and  $\mathbf{x}'$  are  $D$  dimensional vectors, and  $M$  is a positive definite matrix. We focus on low-rank metrics with  $M = W^\top W$  and  $W \in \mathbb{R}^{d \times D}$ , where  $d \leq D$  acts as regularizer and improves efficiency for computation and storage. The Mahalanobis distance induced by  $W$  is equivalent to the squared  $\ell_2$  distance after linear projection of the feature vectors on the rows of  $W$ :

$$\begin{aligned} d_W(\mathbf{x}, \mathbf{x}') &= (\mathbf{x} - \mathbf{x}')^\top W^\top W (\mathbf{x} - \mathbf{x}') \\ &= \|W\mathbf{x} - W\mathbf{x}'\|_2^2. \end{aligned} \quad (9.4)$$

In this chapter, we do not consider using the more general formulation of  $M = W^\top W + S$ , where  $S$  is a diagonal matrix, as in [1]. While this formulation requires only  $D$  additional parameters to estimate, it still requires computing distances in the original high-dimensional space. This is costly for the dense and high-dimensional (4K–64K) Fisher vectors representations we use, as detailed in Sect. 9.5.

We formulate the NCM classifier using a probabilistic model based on multi-class logistic regression and define the probability for a class  $c$  given a feature vector  $\mathbf{x}$  as:

$$p(c|\mathbf{x}) = \frac{\exp(-\frac{1}{2}d_W(\mathbf{x}, \boldsymbol{\mu}_c))}{\sum_{c'=1}^C \exp(-\frac{1}{2}d_W(\mathbf{x}, \boldsymbol{\mu}_{c'}))}. \quad (9.5)$$

This definition may also be interpreted as giving the posterior probabilities of a generative model, where  $p(\mathbf{x}|c) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_c, \Sigma)$ , is a Gaussian with mean  $\boldsymbol{\mu}_c$ , and a covariance matrix  $\Sigma = (W^\top W)^{-1}$ , which is shared across all classes.<sup>1</sup> Using Bayes rule, we obtain:

$$p(c|\mathbf{x}) = \frac{p(\mathbf{x}|c)p(c)}{p(\mathbf{x})} = \frac{\mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_c, \Sigma)p(c)}{\sum_{c'} \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_{c'}, \Sigma)p(c')} \quad (9.6)$$

$$= \frac{Z(\Sigma) \exp(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_c)^\top \Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu}_c))p(c)}{\sum_{c'} Z(\Sigma) \exp(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_{c'})^\top \Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu}_{c'}))p(c')} \quad (9.7)$$

---

<sup>1</sup>Strictly speaking the covariance matrix is not properly defined as the low-rank matrix  $W^\top W$  is non-invertible.

$$= \frac{\exp(-\frac{1}{2}d_W(\mathbf{x}, \boldsymbol{\mu}_c))}{\sum_{c'=1}^C \exp(-\frac{1}{2}d_W(\mathbf{x}, \boldsymbol{\mu}_{c'}))}, \quad (9.8)$$

where the class probabilities  $p(c)$  are set to be uniform over all classes. Later, in Eq. (9.27), we formulate an NCM classifier with non-uniform class probabilities.

We learn the projection matrix  $W$  in a discriminative manner, by maximizing the log-likelihood of the correct predictions of the training images:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^N \ln p(y_i | \mathbf{x}_i). \quad (9.9)$$

The gradient of the NCMMML objective Eq. (9.9) is:

$$\nabla_W \mathcal{L} = \frac{1}{N} \sum_{i=1}^N \sum_{c=1}^C \alpha_{ic} W z_{ic} z_{ic}^\top, \quad (9.10)$$

where  $\alpha_{ic} = p(c|\mathbf{x}_i) - \llbracket y_i = c \rrbracket$ ,  $z_{ic} = \boldsymbol{\mu}_c - \mathbf{x}_i$ , and we use the Iverson brackets  $\llbracket \cdot \rrbracket$  that equals one if its argument is true and zero otherwise.

Although not included above for clarity, the terms in the log-likelihood in Eq. (9.9) could be weighted in cases where the class distributions in the training data are not representative for those when the learned model is applied.

### 9.3.2 Relation to Existing Linear Classifiers

In this section, we related the NCM classifier and the proposed NCMMML approach to other linear models.

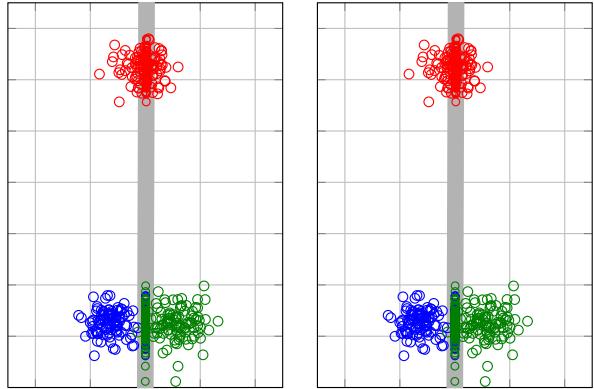
First, we compare the NCMMML objective with the classical Fisher Discriminant Analysis (FDA). The objective of FDA is to find a projection matrix  $W$  that maximizes the ratio of between-class variance to within-class variance:

$$L_{\text{FDA}} = \text{tr}\left(\frac{W S_B W^\top}{W S_W W^\top}\right), \quad (9.11)$$

where  $S_B = \sum_{c=1}^C \frac{N_c}{N} (\boldsymbol{\mu} - \boldsymbol{\mu}_c)(\boldsymbol{\mu} - \boldsymbol{\mu}_c)^\top$  is the weighted covariance matrix of the class centers (and  $\boldsymbol{\mu}$  is the data center), and  $S_W = \sum_{c=1}^C \frac{N_c}{N} \Sigma_c$  is the weighted sum of within class covariance matrices  $\Sigma_c$ . Also to obtain a well-defined problem,  $W$  has a constraint on the norms of its columns. See, for example, [42] for more details on the FDA.

In the case where the within class covariance for each class equals the identity matrix, the FDA objective seeks the direction of maximum variance in  $S_B$ . This equals to a PCA projection on the class means, which has the objective to maximize  $\text{tr}(W^\top S_B W)$ , also with a constraint on the norms of  $W$ . The result of using the unsupervised PCA technique, is that it ignores the class information in the projected space, it just maximizes the variance between all class means. To illustrate this,

**Fig. 9.2** Illustration to compare FDA (left) and NCML (right), the obtained projection direction is indicated by the gray line on which also the projected samples are plotted. For FDA, the result is clearly suboptimal since the blue and green classes are collapsed in the projected space. While the proposed NCML method finds a projection which separates the classes reasonably well



we show an example of a two-dimensional problem with three classes in Fig. 9.2. In contrast to FDA, our NCML method only aims at separating class means which are nearby in the projected space, so as to ensure correct predictions. The resulting projection direction separates the three classes reasonably well.

To relate the NCM classifier to other linear classifiers, we represent them with the class specific score function:

$$f(c, \mathbf{x}) = \mathbf{w}_c^\top \mathbf{x} + b_c, \quad (9.12)$$

that assigns a sample  $\mathbf{x}$  to the class with maximum score. NCM can be seen as a linear classifier by defining  $f_{\text{NCM}}$  with bias and weight vectors given by:

$$b_c = -\frac{1}{2} \|\mathbf{W}\boldsymbol{\mu}_c\|_2^2, \quad (9.13)$$

$$\mathbf{w}_c = \mathbf{W}^\top \mathbf{W} \boldsymbol{\mu}_c. \quad (9.14)$$

This is proportional to the Mahalanobis distance up to an additive constant that is constant with respect to the class  $c$ , and therefore irrelevant for classification.

These definitions allows us relating the NCM classifier to other linear methods. For example, we obtain standard multi-class logistic regression, if the restrictions on  $b_c$  and  $\mathbf{w}_c$  are removed. Note that these are precisely the restrictions that allows us adding new classes at near-zero cost, since the class specific parameters  $b_c$  and  $\mathbf{w}_c$  are defined by just the class means  $\boldsymbol{\mu}_c$  and the class-independent projection  $\mathbf{W}$ .

In the WSABIE method [46], the classifier  $f_{\text{WSABIE}}$ , is defined using  $b_c = 0$  and,

$$\mathbf{w}_c = \mathbf{W}^\top \mathbf{v}_c \quad (9.15)$$

where  $\mathbf{W} \in \mathbb{R}^{d \times D}$  is also a low-rank projection matrix shared between all classes, and  $\mathbf{v}_c$  is a class specific weight vector of dimensionality  $d$ , both learned from data. This is similar to NCM if we set  $\mathbf{v}_c = \mathbf{W} \boldsymbol{\mu}_c$ . As in multiclass logistic regression, however, for WSABIE the  $\mathbf{v}_c$  need to be learned from scratch for new classes.

The NCM classifier can also be related to the solution of ridge-regression (RR, or regularized linear least-squares regression), which also uses a linear score function. The parameters  $b_c$  and  $\mathbf{w}_c$  are learned by optimizing the squared loss:

$$L_{\text{RR}} = \frac{1}{N} \sum_i (f_{\text{RR}}(c, \mathbf{x}_i) - y_{ic})^2 + \lambda \|\mathbf{w}_c\|_2^2, \quad (9.16)$$

where  $\lambda$  acts as regularizer, and where  $y_{ic} = 1$ , if image  $i$  belongs to class  $c$ , and  $y_{ic} = 0$  otherwise. The loss  $L_{\text{RR}}$  can be minimized in closed form and leads to:

$$b_c = \frac{N_c}{N}, \quad (9.17)$$

$$\mathbf{w}_c = \frac{N_c}{N} \boldsymbol{\mu}_c^\top (\Sigma + \lambda I)^{-1}, \quad (9.18)$$

where  $\Sigma$  is the (class-independent) data covariance matrix, see, for example, [42]. Just like the NCM classifier, the RR classifier also allows to add new classes at low cost, since the class specific parameters can be found from the class means and counts once the data covariance matrix  $\Sigma$  has been estimated. Moreover, if  $N_c$  is equal for all classes, RR is similar to NCM with  $W$  set such that  $W^\top W = (\Sigma + \lambda I)^{-1}$ .

Finally, the Taxonomy-embedding method of [43], can be rewritten such that it equals the linear classifier  $f_{\text{TAX}}$  using:

$$b_c = -\frac{1}{2} \|\mathbf{v}_c\|_2^2, \quad (9.19)$$

$$\mathbf{w}_c = W^\top W \mathbf{v}_c, \quad (9.20)$$

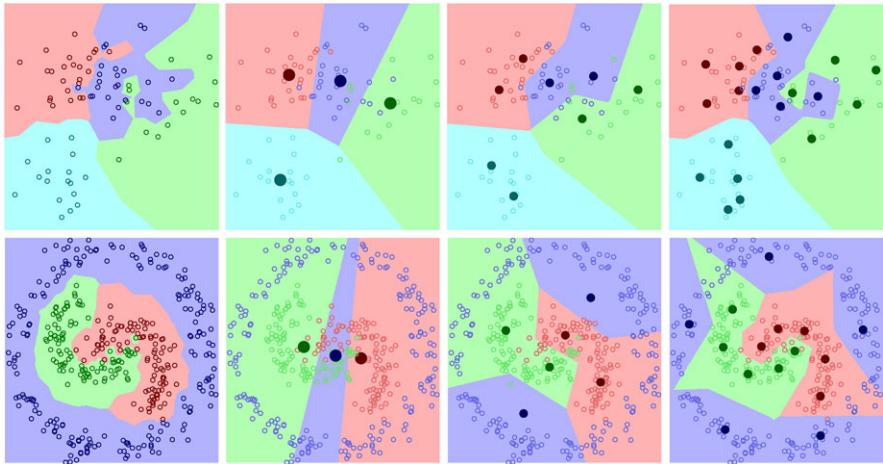
where the class-specific weight vectors  $\mathbf{v}_c$  are learned from the data and  $W \in \mathbb{R}^{C \times D}$  projects the data to a  $C$  dimensional space. The projection matrix  $W$  is set using a closed-form solution based on ridge-regression. This method relates to the WSABIE method since it also learns the classifier in low-dimensional space (if  $C < D$ ), however in this case the projection matrix  $W$  is given in closed-form. It also shares the disadvantage of the WSABIE method: it cannot generalize to novel classes without retraining the low-dimensional class-specific vectors  $\mathbf{v}_c$ .

### 9.3.3 Nonlinear NCM with Multiple Class Centroids

In this section, we extend the NCM classifier to allow for more flexible class representations, which result in non-linear classification, see Fig. 9.3 for an illustration. The idea is to represent each class by a set of centroids, instead of only the class mean. Assume that we have obtained a set of  $k$  centroids  $\{\mathbf{m}_{cj}\}_{j=1}^k$  for each class  $c$ . We define the posterior probability for a centroid  $\mathbf{m}_{cj}$  as:

$$p(\mathbf{m}_{cj} | \mathbf{x}) = \frac{1}{Z} \exp\left(-\frac{1}{2} d_W(\mathbf{x}, \mathbf{m}_{cj})\right), \quad (9.21)$$

where  $Z = \sum_c \sum_j \exp(-\frac{1}{2} d_W(\mathbf{x}, \mathbf{m}_{cj}))$  is the normalizer. The posterior probability for class  $c$  is then given by:



**Fig. 9.3** Illustration of nonlinear classification using multiple centroids on not linearly separable data. From left to right we show, k-NN classification ( $k = 1$ ), NCM classification, and NCMC classification with 2 and 5 centroids, respectively

$$p(c|\mathbf{x}) = \sum_{j=1}^k p(\mathbf{m}_{cj}|\mathbf{x}). \quad (9.22)$$

This model also corresponds to a generative model, where the probability for a feature vector  $\mathbf{x}$ , to be generated by class  $c$ , is given by a Gaussian mixture distribution:

$$p(\mathbf{x}|c) = \sum_{j=1}^k \pi_{cj} \mathcal{N}(\mathbf{x}_i; \mathbf{m}_{cj}, \Sigma), \quad (9.23)$$

with equal mixing weights  $\pi_{cj} = 1/k$ , and the covariance matrix  $\Sigma$  shared among all classes. We refer to this method as the nearest class multiple centroids (NCMC) classifier. A similar model was independently developed recently for image retrieval in [29]. Their objective, however, is to discriminate between different senses of a textual query, and they use a latent model to select the sense of a query.

To learn the projection matrix  $W$ , we again maximize the log-likelihood of correct classification, for which the gradient w.r.t.  $W$  in this case is given by:

$$\nabla_W \mathcal{L} = \frac{1}{N} \sum_{i,c,j} \alpha_{icj} W z_{icj} z_{icj}^\top, \quad (9.24)$$

where  $z_{icj} = \mathbf{m}_{cj} - \mathbf{x}_i$ , and

$$\alpha_{icj} = p(\mathbf{m}_{cj}|\mathbf{x}_i) - \llbracket c = y_i \rrbracket \frac{p(\mathbf{m}_{cj}|\mathbf{x}_i)}{\sum_{j'} p(\mathbf{m}_{cj'}|\mathbf{x}_i)}. \quad (9.25)$$

To obtain the centroids of each class, we apply k-means clustering on the features  $\mathbf{x}$  belonging to that class, using the  $\ell_2$  distance. The value  $k$  offers a transition

between NCM ( $k = 1$ ), and a weighted k-NN ( $k$  equals all images per class), where the weight of each neighbor is defined by the soft-min of its distance, c.f. Eq. (9.21). This is similar to TagProp [18], used for multi-label image annotation, which assigns a probability to a class  $c$  based on the class labels and distances of the images in the training set:

$$p(c|\mathbf{x}_i) = \sum_j \pi_{ij} \llbracket y_j = c \rrbracket, \quad \pi_{ij} = \frac{\exp(-\frac{1}{2}d_W(\mathbf{x}_i, \mathbf{x}_j))}{\sum_{j'} \exp(-\frac{1}{2}d_W(\mathbf{x}_i, \mathbf{x}_{j'}))}. \quad (9.26)$$

In Fig. 9.3, we illustrate the influence of increasing  $k$  on the obtained classification boundaries, and made the comparison with the k-NN ( $k = 1$ ) classifier.

Instead of using a fixed set of class means, it could be advantageous to iterate the k-means clustering and the learning of the projection matrix  $W$ . Such a strategy allows the set of class centroids to represent more precisely the distribution of the images in the projected space, and might further improve the classification performance. However, the experimental validation of such a strategy falls beyond the scope of this paper.

### 9.3.4 Alternative Objective for Small SGD Batches

Computing the gradients for NCMLM in Eq. (9.10) and NCMC in Eq. (9.24) is relatively expensive, regardless of the number of  $m$  samples used per SGD iteration. The cost of this computation is dominated by the computation of the squared distances  $d_W(\mathbf{x}, \boldsymbol{\mu}_c)$ , required to compute the  $m \times C$  probabilities  $p(c|\mathbf{x})$  for  $C$  classes in the SGD update. To compute these distances we have two options. First, we can compute the  $m \times C$  difference vectors  $(\mathbf{x} - \boldsymbol{\mu}_c)$ , project these on the  $d \times D$  matrix  $W$ , and compute the norms of the projected difference vectors, at a total cost of  $O(dD(mC) + mC(d + D))$ . Second, we can first project both the  $m$  data vectors and  $C$  class centers, and then compute distances in the low dimensional space, at a total cost of  $O(dD(m + C) + mC(d))$ . Note that the latter option has a lower complexity, but still requires projecting all class centers at a cost  $O(dDC)$ , which will be the dominating cost when using small SGD batches with  $m \ll C$ . Therefore, in practice we are limited to using SGD batch sizes with  $m \approx C = 1,000$  samples.

In order to accommodate for fast SGD updates based on smaller batch sizes, we replace the Euclidean distance in Eq. (9.5) by the dot-product plus a class specific bias  $s_c$ . The probability for class  $c$  is now given by:

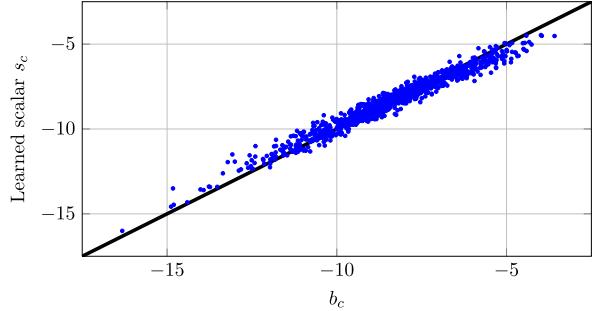
$$p(c|\mathbf{x}_i) = \frac{1}{Z} \exp(\mathbf{x}_i^\top W^\top W \boldsymbol{\mu}_c + s_c), \quad (9.27)$$

where  $Z$  denotes the normalizer. The objective is still to maximize the log-likelihood of Eq. (9.9). The efficiency gain stems from the fact that we can avoid projecting the class centers on  $W$ , by twice projecting the data vectors:  $\hat{\mathbf{x}}_i = \mathbf{x}_i^\top W^\top W$ , and then

**Table 9.1** Comparison of complexity of the considered alternatives to compute the class probabilities  $p(c|\mathbf{x})$

Distances in $D$ dimensions	$O(dD(mC) + mC(d + D))$
Distances in $d$ dimensions	$O(dD(m + C) + mC(d))$
Dot product formulation	$O(dD(m) + mC(D))$

**Fig. 9.4** The learned class-specific biases  $s_c$  and the norm of the projected means  $b_c$  are strongly correlated



computing dot-products in high dimensional space  $\langle \hat{\mathbf{x}}_i, \boldsymbol{\mu}_c \rangle$ . For a batch of  $m$  images, the first step costs  $O(mDd)$ , and the latter  $O(mCD)$ , resulting in a complexity of  $O(dD(m) + mC(D))$ . This complexity scales linearly with  $m$ , and is lower for small batches with  $m \leq d$ , since in that case it is more costly to project the class vectors on  $W$  than on the double-projected data vectors  $\hat{\mathbf{x}}_i$ . For clarity, we summarize the complexity of the different alternatives we considered in Table 9.1.

A potential disadvantage of this approach is that we need to determine the class-specific bias  $s_c$  when data of a new class becomes available, which would require more training than just computing the data mean for the new class. However, we expect a strong correlation between the learned bias  $s_c$  and the biased based on the norm of the projected mean  $b_c$ . Similar as used for Eq. (9.5), we could interpret the class probabilities in Eq. (9.27) as those being generated by a generative model where the class-conditional models  $p(\mathbf{x}|c)$  are Gaussian with a shared covariance matrix. We continue from Eq. (9.7) and obtain:

$$p(c|\mathbf{x}) = \frac{Z(\Sigma) \exp(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_c)^\top \Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu}_c)) p(c)}{\sum_{c'} Z(\Sigma) \exp(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_{c'})^\top \Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu}_{c'})) p(c')} \quad (9.28)$$

$$= \frac{\exp(\mathbf{x}_i^\top W^\top W \boldsymbol{\mu}_c - \frac{1}{2} \|W \boldsymbol{\mu}_c\|_2^2) p(c)}{\sum_{c'} \exp(\mathbf{x}_i^\top W^\top W \boldsymbol{\mu}_{c'} - \frac{1}{2} \|W \boldsymbol{\mu}_{c'}\|_2^2) p(c')} \quad (9.29)$$

In this interpretation, the class specific biases  $s_c$  define class prior probabilities given by  $p(c) \propto \exp(\frac{1}{2} \|W \boldsymbol{\mu}_c\|_2^2 + s_c)$ . Therefore, a uniform prior is obtained by setting  $s_c = -\frac{1}{2} \|W \boldsymbol{\mu}_c\|_2^2 = b_c$ . A uniform prior is reasonable for the ILSVRC'10 data, since the classes are near uniform in the training and test data.

Experimentally, we find that using this formulation yields comparable results as using the Euclidean distance. As expected, we find a strong correlation between the learned bias  $s_c$  and the norm of the projected mean  $b_c$ , shown in Fig. 9.4. Indeed, the

classification performance differs insignificantly if at evaluation time we set  $s_c = b_c$  instead of the value that was found during training.

Thus, even if we train the metric by using class-specific biases, we can use the learned metric in the NCM classifier with the bias based on the norm of the projected mean, which is easily computed for data of new classes.

### 9.3.5 Critical Points of Low Rank Metric Learning

We use a low-rank Mahalanobis distance where  $M = W^\top W$ , as a way to reduce the number of parameters and to gain in computational efficiency. Learning a full Mahalanobis distance matrix  $M$ , however, has the advantage that the distance is linear in  $M$  and that the multi-class logistic regression objective of Eq. (9.9) is therefore concave in  $M$  [4, p. 74]. Using a low-rank formulation, on the other hand, yields a distance which is quadratic in the parameters  $W$ , therefore the objective function is no longer concave. In this section, we investigate the critical-points of the low-rank formulation by analyzing  $W$  when the optimization reaches a (local) minimum, and considering the gradient for the corresponding full matrix  $M = W^\top W$ .

The gradient of the objective of Eq. (9.9) w.r.t. to  $M$  is:

$$\nabla_M \mathcal{L} = \frac{1}{N} \sum_{i,c} \alpha_{ic} z_{ic} z_{ic}^\top \equiv H, \quad (9.30)$$

where  $\alpha_{ic} = p(c|\mathbf{x}_i) - \llbracket y_i = c \rrbracket$ , and  $z_{ic} = \boldsymbol{\mu}_c - \mathbf{x}_i$ . Then Eq. (9.10) follows from the matrix chain rule, and we redefine  $\nabla_W \mathcal{L} \equiv 2WH$ . From the gradient w.r.t.  $W$ , we immediately observe that  $W = 0$  leads to a degenerate case to obtain a zero gradient, and similarly for each row of  $W$ . Below, we concentrate on the non-degenerate case.

We observe that  $H$  is a symmetric matrix, containing the difference of two positive definite matrices. Further, we observe that when  $H = 0$ , that is, when we reach the minimum of the full Mahalanobis distance, we obtain zero gradient for  $W$ . Here we analyze  $H$  when the gradient w.r.t.  $W$  reaches a zero point. In the analysis below, we use the eigenvalue decomposition of  $H = V\Lambda V^\top$ , with the columns of  $V$  being the eigenvectors, and the eigenvalues are on the diagonal of  $\Lambda$ .

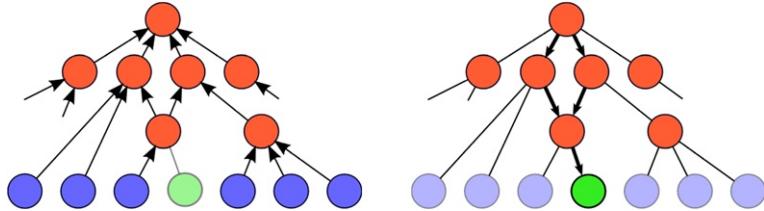
We can now express the gradient for  $W$  as

$$\nabla_W \mathcal{L} = 2WV\Lambda V^\top \equiv G. \quad (9.31)$$

Thus the gradient of the  $i$ th row of  $W$ , which we denote by  $\mathbf{g}_i$ , is a linear combination of the eigenvectors of  $H$ :

$$\mathbf{g}_i \equiv \sum_j \lambda_j \langle \mathbf{w}_i, \mathbf{v}_j \rangle \mathbf{v}_j, \quad (9.32)$$

where  $\mathbf{w}_i$  and  $\mathbf{v}_j$  denote the  $i$ th row of  $W$  and the  $j$ th column of  $V$ , respectively. Thus, an SGD gradient update will drive a row of  $W$  towards the eigenvectors of  $H$  that (i) have a large positive eigenvalue, and (ii) are most aligned with that row



**Fig. 9.5** Illustration of the estimation of the zero-shot prior on a mean. In the first step (*left*), the means of train classes (*blue*) are propagated upwards in the ImageNet hierarchy to the internal nodes (*red*). In the second step (*right*), the prior is estimated as the average of all the ancestor nodes of the new class (*green*)

of  $W$ . This is intuitive, since we would expect the low-rank formulation to focus on the most significant directions of the full-rank metric.

Moreover, the expression for the gradient in Eq. (9.32) shows that at a critical point  $W^*$  of the objective function, all linear combination coefficients are zero:  $\forall i, j : \lambda_j \langle \mathbf{w}_i^*, \mathbf{v}_j \rangle = 0$ . This indicates that at the critical point, for each row  $\mathbf{w}_i^*$  and each eigenvector  $\mathbf{v}_j$  it holds that either  $\mathbf{w}_i^*$  is orthogonal to  $\mathbf{v}_j$ , or that  $\mathbf{v}_j$  has a zero associated eigenvalue, that is,  $\lambda_j = 0$ . Thus, at a critical point  $W^*$ , the corresponding gradient for the full rank formulation at that point, with  $M^* = W^{*\top} W^*$ , is zero in the subspace spanned by  $W^*$ .

Given this analysis, we believe it is unlikely to attain poor local minima using the low rank formulation. Indeed, the gradient updates for  $W$  are aligned with the most important directions of the corresponding full-rank gradient, and at convergence the full-rank gradient is zero in the subspace spanned by  $W$ . To confirm this, we have also experimentally investigated this by training several times with different random initializations of  $W$ . We observe that the classification performance differs at most  $\pm 0.1\%$  on any of the error measures used in Sect. 9.5, and that the number of SGD iterations selected by the early stopping procedure are of the same order.

### 9.3.6 Transfer Learning with the Nearest Class Mean Classifier

In this section, we describe how we can use the NCM classifier in a zero-shot setting. Inspired by [37], we propose to use the ImageNet hierarchy to estimate the mean of novel classes from the means of related training classes, see Fig. 9.5 for an illustration. We follow ideas of [37] and estimate the mean of a novel class  $\mu_z$  using the means of its ancestor nodes in the ImageNet class hierarchy:

$$\mu_z = \frac{1}{|\mathcal{A}_z|} \sum_{a \in \mathcal{A}_z} \mu_a, \quad (9.33)$$

where  $\mathcal{A}_z$  denotes the set of ancestors of node  $z$ , and  $\mu_a$  is the mean of ancestor  $a$ . The mean of an internal node,  $\mu_a$ , is computed as the average of the means of all

its descendant training classes. In our experiments, the classes of interest are always leaf-nodes of the hierarchy.

In the setting where we also have a few images of the new class, we can combine the zero-shot prior with the mean of the sample images. If we view the estimation of each class mean as the estimation of the mean of a Gaussian distribution, then the mean of a sample of images  $\mu_s$  corresponds to the Maximum Likelihood (ML) estimate, while the zero-shot estimate  $\mu_z$  can be thought of as a prior. We can combine the prior with the ML estimate to obtain a maximum a-posteriori (MAP) estimate  $\mu_p$  on the class mean. The MAP estimate of the mean of a Gaussian is obtained by:

$$\mu_p = \frac{n\mu_s + m\mu_z}{n + m}, \quad (9.34)$$

where  $n$  is the number of images used to compute the ML estimate of the sample mean  $\mu_s$ , and the prior obtains a weight  $m$  determined on the validation set [13].

## 9.4 k-NN Metric Learning

We compare the NCM classifier to the k-NN classifier, a frequently used distance based classifier. The k-NN classifier is attractive since it is very intuitive, just assigning the class of the nearest neighbors to a test image, see Fig. 9.1(a) for an illustration. Just as for the NCM classifier, the k-NN classifier relies on distances, and thus it is essential to use a metric in which nearby images are also semantically related. In this section we discuss metric learning for k-NN classifiers, used to learn a low-rank Mahalanobis distance  $M = W^\top W$ , where  $W \in \mathbb{R}^{d \times D}$ .

For successful k-NN classification, the majority of the nearest neighbors should be of the same class. This is reflected in the Large Margin Nearest Neighbors (LMNN) metric learning objective of [44, 45]. LMNN is defined over triplets consisting of a query image  $q$ , a positive image  $p$  from the same class, and a negative image  $n$  from another class. The objective is to get the distance between  $q$  and  $p$  smaller than the distance between  $q$  and  $n$ , using the hinge-loss to upper bound the zero/one loss:

$$L_{qpn} = [1 + d_W(\mathbf{x}_q, \mathbf{x}_p) - d_W(\mathbf{x}_q, \mathbf{x}_n)]_+, \quad (9.35)$$

where  $[z]_+ = \max(0, z)$  is the positive part of  $z$ . The hinge-loss for a triplet is zero if the negative image  $n$  is at least one distance unit farther from the query  $q$  than the positive image  $p$ , and the loss is positive otherwise. The final learning objective sums the losses over all triplets:

$$L_{\text{LMNN}} = \sum_{q=1}^N \sum_{p \in P_q} \sum_{n \in N_q} L_{qpn}, \quad (9.36)$$

where  $P_q$  and  $N_q$  denote a predefined set of positive and negative images for each query image  $q$ . An important design choice is how to set  $P_q$  and  $N_q$  for each query.

For the set of negative images  $N_q$ , we follow [44] and use all images not belonging to the class of the query image. Below, we describe several variants for the set of positive images  $P_q$ .

Also in this case we can weight the terms in the loss function to account for non-representative class proportions in the training data.

### 9.4.1 Choice of Target Neighbors

For LMNN a set of target  $P_q$  for a query  $q$  is set to some images from the same class. The rationale is that if we ensure that these targets are closer than the instances of the other classes, then the k-NN classification will succeed. To select the set of targets, we consider three alternatives:

1. In the basic version of LMNN the set of targets  $P_q$  is set to the query's  $k$  nearest neighbors from the same class, using the  $\ell_2$  distance. Since this selection method tries to ensure that the  $\ell_2$ -targets will also be the closest points using the learned metric, it implicitly assumes that the  $\ell_2$  distance in the original space is a good similarity measure. In practice, however, this might not be the case.
2. The set of targets  $P_q$  is defined to contain all images of the same class as  $q$ , hence the selection is independent on the metric. This is similar to [6] where the same type of loss was used to learn image similarity defined as the scalar product between feature vectors after a learned linear projection.
3. The set of targets  $P_q$  is dynamically updated to contain the  $k$  images of the same class that are closest to  $q$  using the current metric  $W$ . Hence, different target neighbors can be selected depending on the current metric. This method corresponds to minimizing the loss function also with respect to the choice of  $P_q$ . A similar approach has been proposed in [44], where every  $T$  iterations  $P_q$  is redefined using target neighbors according to the current metric.

A potential disadvantage of the last method is that it requires frequent recomputation of the target neighbors. However, below we describe an efficient gradient evaluation method, which allows to approximate the dynamic set of  $P_q$  at each iteration at a negligible additional cost compared to using a fixed set of target neighbors or using all images of the same class as targets.

Next, we will discuss an efficient triplet sampling and gradient evaluation algorithms to increase the efficiency of the SGD training.

### 9.4.2 Triplet Sampling Strategy

Here, we describe a sampling strategy which obtains the maximal number of triplets from  $m$  images selected per SGD iteration. Using a small  $m$  is advantageous since the cost of the gradient evaluation is in large part determined by computing the projections  $Wx$  of the images, and the cost of decompressing the PQ encoded signatures, if these are used.

To generate triplets, we first select uniformly at random a class  $c$ , that will provide the query and positive images. When  $P_q$  is set to contain all images of the same class, we sample  $\rho m$  images from class  $c$ , with  $0 < \rho < 1$ , and the remaining  $(m - \rho m)$  images are uniformly sampled from the other classes. We can consider the number of triplets  $t$  we can generate as a function of  $\rho$  for a given ‘budget’ of  $m$  images to be accessed. In the case where  $P_q$  is set to contain all images from the same class, the number of triplets  $t$  we can generate for a specific  $\rho$  is given by:

$$t(\rho) = (\rho m)(\rho m - 1)(m - \rho m), \quad (9.37)$$

since we can pair the  $\rho m$  images with the  $\rho m - 1$  other images from the same class, and each pair forms a triplet with any of the  $m - \rho m$  negative sampled images. The number of triplets  $t$  can be approximated by:

$$t(\rho) \approx m^3 \rho^2 (1 - \rho), \quad (9.38)$$

and hence, the number of triplets is maximized when we choose  $\rho \approx \frac{2}{3}$ , in which case we can construct about  $\frac{4}{27}m^3$  triplets. In our experiments, we use  $\rho = \frac{2}{3}$  and  $m = 300$  images per iteration, leading to about 4 million triplets per iteration.

For other choices of  $P_q$ , we do the following:

- For a fixed set of target neighbors, we still sample  $\frac{1}{3}m$  negative images, and take as many query images together with their target neighbors until we obtain  $\frac{2}{3}m$  images allocated for the positive class.
- For a dynamic set of target neighbors, we simply select the closest neighbors among the  $\frac{2}{3}m$  sampled positive images using the current metric  $W$ . Although approximate, this avoids computing the dynamic target neighbors among all the images in the positive class.

An alternative to obtain roughly 4 million triplets is to sample two images from each of the  $C = 1,000$  classes. In this case, there are two query images per class, each forming a pair with the other positive image, and each pair can form a triplet with the  $2(C - 1)$  images of other classes, leading to  $4C(C - 1) \approx 4$  million triplets. A potential advantage of this method is that the gradient is computed in each iteration from triplets generated using all possible combinations of classes, and therefore, the gradient might be more informative. However, this sampling strategy does not allow for fast approximation of the dynamic neighbors, and we would need to access  $m = 2,000$  images, which is about 7 times more costly than using the described approach with  $m = 300$ .

### 9.4.3 Efficient Gradient Evaluation

For either choice of the target set  $P_q$ , the gradient can be computed without explicitly iterating over all triplets. In this section, we introduce an efficient gradient evaluation method, which uses sorting of the distances w.r.t. query images.

**Algorithm 1** Compute coefficients  $A_{qn}$  and  $A_{qp}$ 

- 
1. For positive images redefine  $d_W(\mathbf{x}_q, \mathbf{x}_p) \leftarrow d_W(\mathbf{x}_q, \mathbf{x}_p) + 1$  to account for the margin.
  2. Sort distances w.r.t.  $q$  in ascending order.
  3.  $C_n(i) \leftarrow \sum_{j=1}^i [\![j \in N_q]\!]$ , the number of negative images up to each position.
  4.  $C_p(i) \leftarrow \sum_{j=i+1}^m [\![j \in P_q]\!]$ , the number of positive images after each position.
  5. Read-off the number of hinge-loss generating triplets of image  $p$  or  $n$ :

$$A_{qn} = -2C_p(\text{rnk}(q, n)), \quad A_{qp} = 2C_n(\text{rnk}(q, p)),$$

where  $\text{rnk}(q, p)$  indicates the rank of document  $p$  for the query  $q$ , and similar for  $\text{rnk}(q, n)$ .

---

The sub-gradient of the loss of a triplet is given by:

$$\nabla_W L_{qpn} = [\![L_{qpn} > 0]\!] 2W(\mathbf{x}_{qp}\mathbf{x}_{qp}^\top - \mathbf{x}_{qn}\mathbf{x}_{qn}^\top), \quad (9.39)$$

where  $\mathbf{x}_{qp} = \mathbf{x}_q - \mathbf{x}_p$ , and  $\mathbf{x}_{qn} = \mathbf{x}_q - \mathbf{x}_n$ . By observing that the gradient takes the form of outer products of the feature vectors, we can write the gradient w.r.t.  $L_q = \sum_{p,n} L_{qpn}$  in matrix form as:

$$\nabla_W L_q = 2WXAX^\top, \quad (9.40)$$

where  $X$  contains the  $m$  feature vectors used in an SGD iteration, and  $A$  is a coefficient matrix. This shows that once  $A$  is available, the gradient can be computed in time  $O(m^2)$ , even if a much larger number of triplets is used.

When  $P_q$  contains all images of the same class, the gradient per query can be rewritten as:

$$\begin{aligned} \nabla_W L_q = & +2W \sum_p \left( \sum_n [\![L_{qpn} > 0]\!] \right) (\mathbf{x}_p\mathbf{x}_p^\top - \mathbf{x}_q\mathbf{x}_p^\top - \mathbf{x}_p\mathbf{x}_q^\top) \\ & - 2W \sum_n \left( \sum_p [\![L_{qpn} > 0]\!] \right) (\mathbf{x}_n\mathbf{x}_n^\top - \mathbf{x}_q\mathbf{x}_n^\top - \mathbf{x}_n\mathbf{x}_q^\top). \end{aligned} \quad (9.41)$$

Which shows that the coefficient matrix  $A$  can be computed from the number of hinge-loss generating triplets in which each  $p \in P_q$  and each  $n \in N_q$  for a query  $q$  occurs:

$$A_{qn} = 2 \sum_p [\![L_{qpn} > 0]\!], \quad A_{pq} = -2 \sum_n [\![L_{qpn} > 0]\!], \quad (9.42)$$

$$A_{qq} = \sum_p A_{qp} - \sum_n A_{qn}, \quad A_{pp} = \sum_q A_{qp}, \quad A_{nn} = \sum_q A_{qn}. \quad (9.43)$$

In Algorithm 1, we describe how to efficiently compute the coefficients, which sorts the distances w.r.t. the query  $q$  and then can read off the number of hinge-loss generating triplets. The same algorithm can be applied when using a small set of fixed, or dynamic target neighbors. In particular, the sorted list allows to dynami-

cally determine the target neighbors at a negligible additional cost. In this case only the selected target neighbors obtain nonzero coefficients, and we only accumulate the number of target neighbors after each position in step 3 of the algorithm.

The cost of this algorithm is  $O(m \log m)$  per query, and thus  $O(m^2 \log m)$  when using  $O(m)$  query images per iteration. This is significantly faster than explicitly looping over all  $O(m^3)$  triplets.

Note that while this algorithm enables fast computation of the sub-gradient of the loss, the value of the loss itself cannot be determined using this method. However, this is not a problem when using an SGD approach, as it only requires gradient evaluations, not function evaluations.

## 9.5 Experimental Evaluation

In this section, we experimentally validate our models described in the previous sections. We first describe the dataset and evaluation measures used in our experiments, followed by the presentation of the experimental results.

### 9.5.1 Experimental Setup and Baseline Approach

In this section, we describe the experimental setup, our image representations and our baseline methods.

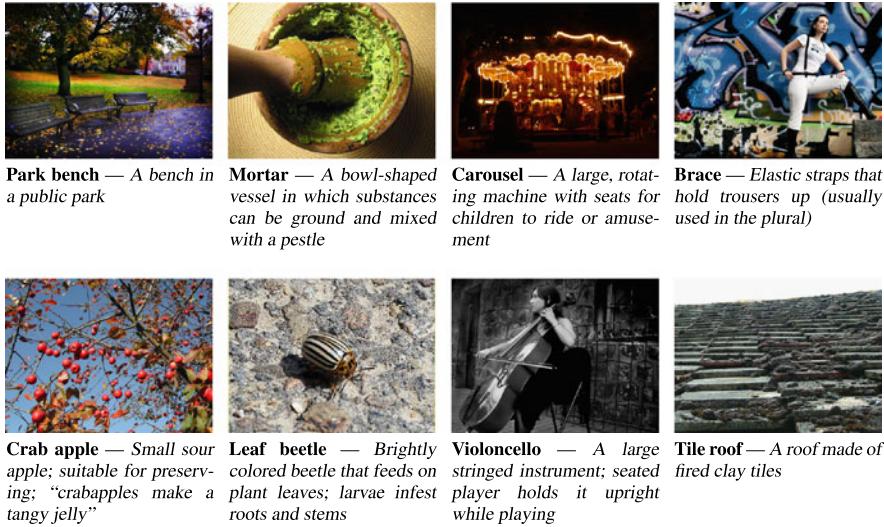
**Dataset** In most of our experiments, we use the dataset of the ImageNet Large Scale Visual Recognition 2010 challenge (ILSVRC’10),<sup>2</sup> see Fig. 9.6 for a few examples. This dataset contains 1.2M training images of 1,000 object classes (with between 660 to 3047 images per class), a validation set of 50K images (50 per class), and a test set of 150K images (150 per class).

In some of the experiments, we use the ImageNet-10K dataset introduced in [10], which consists of 10,184 classes from the nodes of the ImageNet hierarchy with more than 200 images. We follow [39] and use 4.5M images as training set, 50K as validation set and the rest as test set.

**Image Representation** We represent each image with a Fisher vector (FV) [35] computed over densely extracted 128 dimensional SIFT descriptors [28] and 96 dimensional local color features [7], both projected with PCA to 64 dimensions. FVs are extracted and normalized separately for both channels and then combined by concatenating the two feature vectors. We do not make use of spatial pyramids. In our experiments, we use FVs extracted using a vocabulary of either 16 or 256 Gaussians. For 16 Gaussians, this leads to a 4K dimensional feature vector, which re-

---

<sup>2</sup>See <http://www.image-net.org/challenges/LSVRC/2010/index>.



**Fig. 9.6** Example images from the ILSVRC'10 data set, with their class names and descriptions. The data set contains 1.2M training images of 1,000 different classes

quires about 20GB for the 1.2M training set (using 4-byte floating point arithmetic). This fits into the RAM of our 32GB servers.

For 256 Gaussians, the FVs are 16 times larger, that is, 64K dimensional, which would require 320GB of memory. To fit the data in memory, we compress the feature vectors using product quantization [17, 21]. In a nutshell, it consists in splitting the high-dimensional vector into small sub-vectors, and vector quantizing each sub-vector independently. We compress the dataset to approximately 10GB using 8-dimensional sub-vectors and 256 centroids per sub-quantizer, which allows storing each sub-quantizer index in a single byte, combined with a sparse encoding of the zero sub-vectors, c.f. [39]. In each iteration of SGD learning, we decompress the features of a limited number of images, and use these (lossy) reconstructions for the gradient computation.

**Evaluation Measures** We report the average top-1 and top-5 flat error used in the ILSVRC'10 challenge. The flat error is one if the ground-truth label does not correspond to the top-1 label with highest score (or any of the top-5 labels), and zero otherwise. The motivation for the top-5 error is to allow an algorithm to identify multiple objects in an image and not being penalized if one of the objects identified was in fact present but not included in the ground truth of the image which contains only a single object category per image. Unless specified otherwise, we report the top-5 flat error on the test set using the 4K dimensional features; we use the validation set for parameter tuning only.

**Baseline Approach** For our baseline, we follow the state-of-the-art approach of [36] and learn weighed one-vs.-rest SVMs with SGD, where the number of neg-

ative images in each iteration is sampled proportional to the number of positive images for that class. The proportion parameter is cross validated on the validation set. The results of the baseline can be found in Table 9.3 and Table 9.6. We see that the 64K dimensional features lead to significantly better results than the 4K ones, despite the lossy PQ compression.

In Table 9.3, the performance using the 64K features is slightly better than the ILSVRC'10 challenge winners [27] (28.0 vs. 28.2 flat top-5 error), and very close to the results of [39] (25.7 flat top-5 error), wherein a much higher dimensional image representation of more than 1M dimensions was used. In Table 9.6, our baseline shows state-of-the-art performance on ImageNet-10K when using the 64K features, obtaining 78.1 vs. 81.9 flat top-1 error [36]. We believe this is due to the use of the color features, in addition to the SIFT features used in [36].

**SGD Training and Early Stopping** To learn the projection matrix  $W$ , we use SGD training and sample at each iteration a fixed number of  $m$  training images to estimate the gradient. Following [1], we use a fixed learning rate and do not include an explicit regularization term, but rather use the projection dimension  $d$ , as well as the number of iterations as an implicit form of regularization. For all experiments, we use the following early stopping strategy:

- we run SGD training for a large number of iterations ( $\approx 750K$ – $2M$ ),
- the performance on the validation set is computed every 50k iterations (for the k-NN) or every 10k iterations (for the NCM), and
- the metric which yields the lowest top-5 error on the validation set is selected.

In case of a tie, the metric giving the lowest top-1 error is chosen. Similarly, all hyper-parameters, like the value of  $k$  for the k-NN classifiers, are validated in this way. Unless stated otherwise, training is performed using the ILSVRC'10 training set, and validation using the described early stopping strategy on the provided 50K validation set.

It is interesting to notice that while the compared methods (k-NN, NCM, and SVM) have different computational complexities, the number of images seen by each algorithm before convergence is rather similar. For example, training of the SVMs, on the 4K features, converge after  $T \approx 100$  iterations, and each iteration takes about 64 negative images per positive image, per class. In the ILSVRC'10 dataset, each class has roughly  $p = 1,200$  positive images, and consist of  $C = 1,000$  classes. Therefore, the total number of images seen during training of the SVMs is  $TC(65p) = 7.800M$  images. The NCM classifier requires much more iterations,  $T \approx 500K$ , but uses at each iteration only  $m = 1,000$  images, and trains only a single metric. Therefore, the total number of images seen during training is roughly  $Tm = 500M$ . Finally, the k-NN classifier, requires even more iterations,  $T \approx 2M$ , but uses only  $m = 300$  images per iteration, the total number of images seen before convergence is therefore about  $Tm = 600M$ .

**Table 9.2** Comparison of flat error for different k-NN classification methods using 4K dimensional features. For all methods, except those indicated by ‘Full’, the data is projected to a 128 dimensional space

k-NN classifiers								
SVM	$\ell_2$	$\ell_2$	LMNN		All	Dynamic		
Full	Full	+ PCA	10	20		10	20	
Top-5	38.2	55.7	57.3	50.6	50.4	44.2	39.7	40.7

### 9.5.2 k-NN Metric Learning Results

We start with an assessment of k-NN classifiers in order to select a baseline for comparison with the NCM classifier. Given the cost of k-NN classifiers, we focus our experiments on the 4K dimensional features, and consider the impact of the different choices for the set of target images  $P_q$  (see Sect. 9.4), and the projection dimensionality.

We initialize  $W$  as a PCA projection, and determine the number of nearest neighbors to use for classification on the validation set. Typically using 100 to 250 neighbors is optimal, which is rather large for k-NN classification, for example in [44]  $k = 3$  is used, and indicates that the classification function is rather smooth.

#### 9.5.2.1 Target Selection for k-NN Metric Learning

In the first experiment, we compare the three different options of Sect. 9.4 to define the set of target images  $P_q$ , while learning projections to 128 dimensions. For LMNN and dynamic targets, we experimented with various numbers of targets on the validation set and found that using 10 to 20 targets yields the best results.

The results in Table 9.2 show that all methods lead to metrics that are better than the  $\ell_2$  metric in the original space, or after a PCA projection to 128 dimensions. Furthermore, we can improve over LMNN by using all within-class images as targets, or even further by using dynamic targets. The success of the dynamic target selection can be explained by the fact that among the three alternatives, the learning objective is the most closely related to the k-NN classification rule. The best performance on the flat top-5 error of 39.7 using 10 dynamic targets is, however, slightly worse than the 38.2 error rate of the SVM baseline.

#### 9.5.2.2 Impact of Projection Dimension on k-NN Classification

Next, we evaluate the influence of the projection dimensionality  $d$  on the performance, by varying  $d$  between 32 and 1024. We only show results using 10 dynamic targets, since this performed best among the evaluated k-NN methods. From the results in Table 9.3 we see that a projection to 256 dimensions yields the lowest error of 39.0, which still remains somewhat inferior to the SVM baseline.

**Table 9.3** Flat top-5 error of k-NN and NCM classifiers, as well as baselines, using the 4K and 64K dimensional features, for various projection dimensions, and comparison to related methods, see text for details

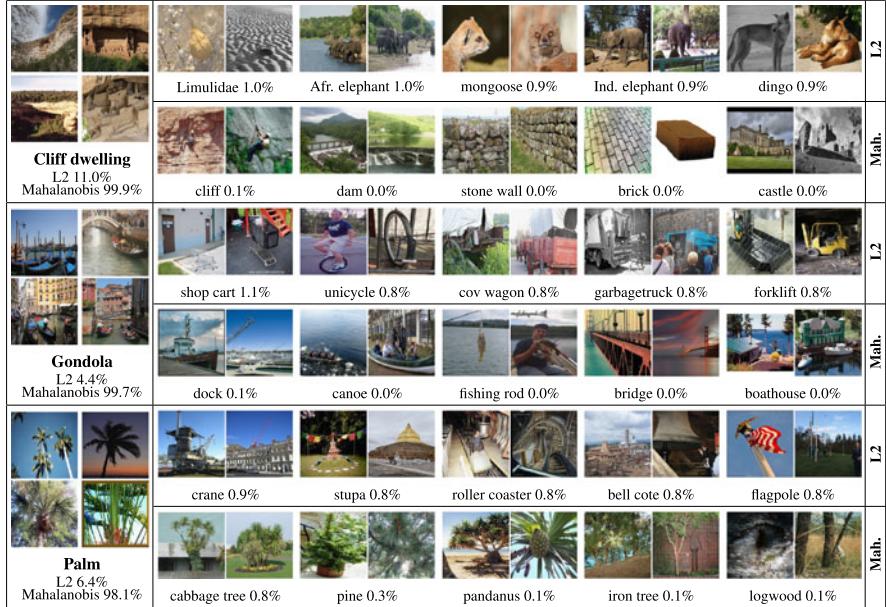
Proj. dim.	4K dimensional features							64K dimensional features			
	32	64	128	256	512	1024	Full	128	256	512	Full
SVM baseline								38.2			
k-NN, dynamic 10	47.2	42.2	39.7	39.0	39.4	42.4					
NCM, NCMLL	49.1	42.7	39.0	37.4	37.0	37.0		31.7	31.0	30.7	
NCM, FDA	65.2	59.4	54.6	52.0	50.8	50.5					
NCM, PCA + $\ell_2$	78.7	74.6	71.7	69.9	68.8	68.2	68.0				63.2
NCM, PCA + inv. cov.	75.5	67.7	60.6	54.5	49.3	46.1	43.8				
Ridge-regression, PCA	86.3	80.3	73.9	68.1	62.8	58.9	54.6				
WSABIE	51.9	45.1	41.2	39.4	38.7	38.5		32.2	30.1	29.2	

### 9.5.3 Nearest Class Mean Classifier Results

We now consider the performance of NCM classifiers and the related methods described in Sect. 9.3. In Table 9.3, we show the results for various projection dimensionalities.

We first consider the results for the 4K dimensional features. As observed for the k-NN classifier, using a learned metric outperforms using the  $\ell_2$  distance (68.0), which is far worse than using the k-NN classifier (55.7, see Table 9.2). However, unexpectedly, with metric learning we observe that our NCM classifier (37.0) outperforms the more flexible k-NN classifier (39.0), as well as the SVM baseline (38.2) when projecting to 256 dimensions or more. Our implementation of WSABIE [46] scores slightly worse (38.5) than the baseline and our NCM classifier, and does not generalize to new classes without retraining.

We also compare our NCM classifier to several algorithms which do allow generalization to new classes. First, we consider two other supervised metric learning approaches, NCM with FDA (which leads to 50.2) and ridge-regression (which leads to 54.6). We observe that NCMLL outperforms both methods significantly. Second, we consider two unsupervised variants of the NCM classifier where we use PCA to reduce the dimensionality. In one case, we use the  $\ell_2$  metric after PCA. In the other, inspired by ridge-regression, we use NCM with the metric  $W$  generated by the inverse of the regularized covariance matrix, such that  $W^\top W = (\Sigma + \lambda I)^{-1}$ , see Sect. 9.3.2. We tuned the regularization parameter  $\lambda$  on the validation set, as was also done for ridge-regression. From these results, we can conclude that, just like for k-NN, the  $\ell_2$  metric with or without PCA leads to poor results (68.0) as compared to a learned metric. Also, the feature whitening implemented by the inverse covariance metric leads to results (43.8) that are better than using the  $\ell_2$  metric, and also substantially better than ridge-regression (54.6). The results are however significantly worse than using our learned metric, in particular when using low-dimensional projections.

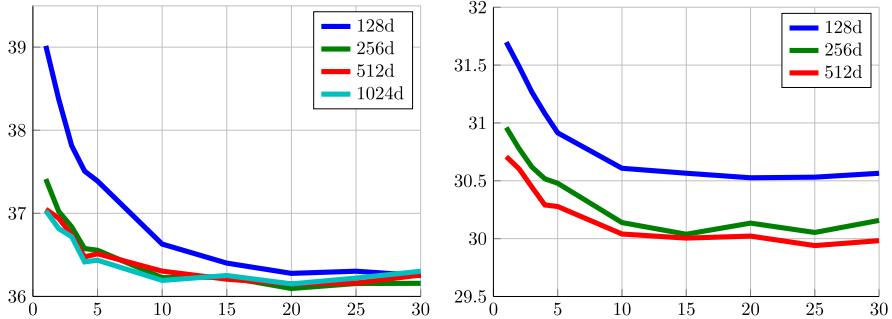


**Fig. 9.7** The nearest classes for two reference classes using the the  $\ell_2$  distance and metric learned by NCML. Class probabilities are given for a simulated image that equals the mean of the reference class, see text for details

When we use the 64K dimensional features, the results of the NCM classifier (30.8) are somewhat worse than the SVM baseline (28.0); again the learned metric is significantly better than using the  $\ell_2$  distance (63.2). WSABIE obtains an error of 29.2, in between the SVM and NCM.

### 9.5.3.1 Illustration of Metric Learned by NCML

In Fig. 9.7, we illustrate the difference between the  $\ell_2$  and the Mahalanobis metric induced by a learned projection from 64K to 512 dimensions. For three reference classes we show the five nearest classes, based on the distance between class means. We also show the posterior probabilities on the reference class and its five neighbor classes according to Eq. (9.5). The feature vector  $x$  is set as the mean of the reference class, that is, a simulated perfectly typical image of this class. For the  $\ell_2$  metric, we used our metric learning algorithm to learn a scaling of the  $\ell_2$  metric to minimize Eq. (9.9). This does not change the ordering of classes, but ensures that we can compare probabilities computed using both metrics. We find that, as expected, the learned metric has more visually and semantically related neighbor classes. Moreover, we see that using the learned metric most of the probability mass is assigned to the reference class, whereas the  $\ell_2$  metric leads to rather uncertain classifications.



**Fig. 9.8** The flat top-5 error of the NCMC-test classifier, which at test time uses  $k > 1$  on a metric obtained with  $k = 1$ , for the 4K features (*left*) and the 64k (*right*) features

**Table 9.4** The flat top-5 error of the NCMC classifier using the 4K features, compared to the NCM baseline and the best NCMC-test classifier (with  $k$  in brackets)

Method	NCM	NMC-test		NMC		
		( $k$ )		5	10	15
128	39.0	36.3	(30)	36.2	35.8	36.1
256	37.4	36.1	(20)	35.0	34.8	35.3
512	37.0	36.2	(20)	34.8	34.6	35.1

### 9.5.3.2 Non-linear Classification Using Multiple Class Centroids

In these experiments, we use the non-linear NCMC classifier, introduced in Sect. 9.3.3, where each class is represented by a set of  $k$  centroids. We obtain the  $k$  centroids per class by using the  $k$ -means algorithm in the  $\ell_2$  space.

Since the cost of training these classifiers is much higher, we run two sets of experiments. In Fig. 9.8, we show the performance of using the NCMC classifier only at test time with  $k = [2, \dots, 30]$ , while using a metric obtained by the NCM objective ( $k = 1$ ). This method is denoted as NCMC-test. In Table 9.4, we show the performance of the NCMC classifier, trained with the NCMC objective, using the 4K features. In the same table, we compare the results to the NCM method and the best NCMC-test method.

From the results, we observe that a significant performance improvement can be made by using the non-linear NCMC classifier, especially when using a low number of projection dimensionalities. When learning the NCMC classifier, we can further improve the performance of the nonlinear classification, albeit for a higher training cost. When using as little as 512 projection dimensions, we obtain a performance of 34.6 on the top-5 error, using  $k = 10$  centroids. That is an improvement of about 2.4 absolute points over the NCM classifier (37.0), and 3.6 absolute points over SVM classification (38.2), c.f. Table 9.3.

For the 64K features learning for the NCMC objective (with,  $k = 10$  and  $d = 512$ ) improves the performance to 29.4, about 1.3 points over the NCM classifier.

**Table 9.5** Flat top-5 error for 1,000-way classification among test images of 200 classes not used for metric learning, and control setting with metric learning using all classes

Method	4K dimensional features								64K dimensional features								
	SVM k-NN				NCM				SVM				NCM				
	Full	128	256	$\ell_2$	128	256	512	1024	Full	128	256	512	$\ell_2$				
Trained on all	37.6	39.0	38.4		38.6	36.8	36.4	36.5		27.7	31.7	30.8	30.6				
Trained on 800		42.2	42.4	54.2	42.5	40.4	39.9	39.6	66.6		39.3	37.8	37.8	61.9			

### 9.5.4 Generalizing to New Classes with Few Samples

Given the encouraging classification accuracy of the NCM classifier observed above—and its superior efficiency as compared to the k-NN classifier—we now explore its ability to generalize to novel classes. We also consider its performance as a function of the number of training images available to estimate the mean of novel classes.

**Generalization to Classes not Seen During Training** In this experiment, we use approximately 1M images corresponding to 800 random classes to learn metrics, and evaluate the generalization performance on 200 held-out classes. The error is evaluated in a 1,000-way classification task, and computed over the 30K images in the test set of the held-out classes. The early stopping strategy uses the validation set of the 200 unseen classes. Performance among test images of the 800 train classes changes only marginally and would obscure the changes among the test images of the 200 held-out classes.

In Table 9.5, we show the performance of NCM and k-NN classifiers, and compare it to the control setting where the metric is trained on all 1,000 classes. The results show that both classifiers generalize remarkably well to new classes. For comparison, we also include the results of the SVM baseline, and the k-NN and NCM classifiers using the  $\ell_2$  distance, evaluated over the 200 held-out classes. In particular for 1024 dimensional projections of the 4K features, the NCM classifier achieves an error of 39.6 over classes not seen during training, as compared to 36.5 when using all classes for training. For the 64K dimensional features, the drop in performance is larger, but still surprisingly good considering that training for the novel classes consists only in computing their means.

To further demonstrate the generalization ability of the NCM classifier using learned metrics, we also compare it against the SVM baseline on the ImageNet-10K dataset. We use projections learned and validated on the ILSVRC’10 dataset, and only compute the means of the 10K classes. The results in Table 9.6 show that even in this extremely challenging setting the NCM classifier performs remarkably well compared to earlier mentioned SVM-based results of [10, 36, 39] and our baseline, all of which require training 10K classifiers. We note that, to the best of our knowledge, our baseline results exceed the previously known state-of-the-art [26, 36] on this dataset. Training our SVM baseline system took 9 and 280 CPU days respec-

**Table 9.6** Flat error rate of the NCM classifier on the ImageNet-10K dataset, using metrics learned on the ILSVRC’10 dataset, with comparison to the baseline SVM, the NCM using  $\ell_2$  distance (denoted as full), and previously reported SVM results [10, 36, 39] and the Deep Learning framework of [26]

Method	4K dimensional features				64K dimensional features				Previous Results			
	NCM	SVM	NCM	SVM	[10]	[39]	[36]	[26]				
Proj. dim.	128	256	512	1024	Full	Full	128	256	512	Full	Full	21K 128K 128K
Top-1 err.	91.8	90.6	90.5	90.4	95.5	86.0	87.1	86.3	86.1	93.6	78.1	93.6 83.3 80.9 80.8
Top-5 err.	80.7	78.7	78.6	78.6	89.0	72.4	71.7	70.5	70.1	85.4	60.9	

tively for the 4K and 64K features, while the computation of the means for the NCM classifier took approximately 3 and 48 CPU minutes, respectively. This represents roughly a 8,500 fold speed-up as compared to the baseline, without counting the time to learn the projection matrix.

**Accuracy as Function of Sample Size of Novel Classes** In this experiment, we consider the error as a function of the number of images that are used to compute the means of novel classes. Inspired by [37], we also include results of a zero-shot learning experiment, where we use the ImageNet hierarchy to estimate the mean of novel classes from the means of related training classes, see Sect. 9.3.6 for details.

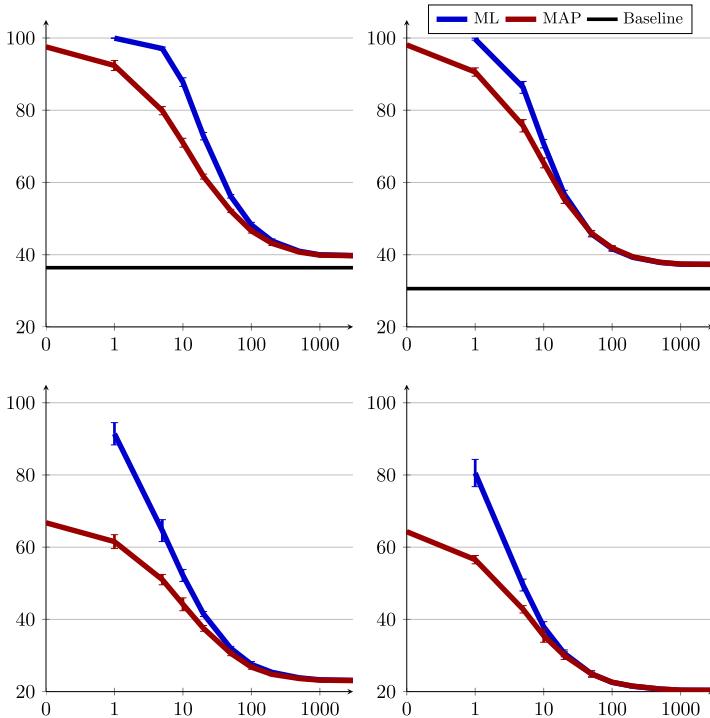
In Fig. 9.9, we analyze the performance of the NCM classifier trained on the images of the same 800 classes used above, with a learned projection from 4K and 64K to 512 dimensions. The metric and the parameter  $m$  are validated using the images of the 200 held-out classes of the validation set. We again report the error on the test images of the held-out classes, both in a 1,000-way classification as above, and in a 200-way classification as in [37]. We repeat the experiment 10 times, and show error-bars at three times standard deviation. For the error to stabilize, we only need approximately 100 images to estimate the class means.

The results also show that the prior leads to zero-shot performance of 66.5 (4K features) and 64.0 (64K features), in the 200-way classification setting. These results are comparable to the result of 65.2 reported in [37], even though a different set of 200 test-classes were used. Note that they also used different features, however their baseline performance of 37.6 top-5 error is comparable to our 4K features (38.2).

More importantly, we show that the zero-shot prior can be effectively combined with the empirical mean to provide a smooth transition from the zero-shot setting to a setting with many training examples. Inclusion of the zero-shot prior leads to a significant error reduction in the regime where ten images or less are available.

### 9.5.5 Instance Level Image Retrieval

Query-by-example image retrieval can be seen as an image classification problem where only a single positive sample (the query) is given and negative examples



**Fig. 9.9** Flat top-5 error of NCM as a function of the number of images used to compute the means for test classes. We compare the ML (blue) and MAP (red) mean estimates, for the 4K (left) and 64K (right) features. We show the results for 1,000-way (top), including baseline when trained on all classes in black, and 200-way classification (bottom)

are not explicitly provided. In this case, the class mean simplifies to the query. We propose to use a metric learned for our NCM classifier on an auxiliary supervised dataset to retrieve the most similar images for a given query.

Using classifiers to learn a metric for image retrieval was recently considered also in [16]. They found the Joint Subspace and Classifier Learning (JSCL) method to be most effective. This basically amounts to jointly learning a set of classifiers and a projection matrix  $W$  using the WSABIE scoring function, Eq. (9.15), and minimizing the hinge-loss on class labels. After training, the classifiers are discarded and only the learned projection matrix  $W$  is used to compute distances between query and database images.

For this experiment, we use the same public benchmarks as in [16]. First, the INRIA Holidays data set introduced by [20] consists of 1,491 images of 500 scenes and objects. In the standard evaluation protocol, one image per scene/object is used as query to search withing the remaining images. The accuracy is measured as the mean average precision over the 500 queries (mAP). Second, the University of Kentucky Benchmark dataset (UKB) introduced by [32], which contains of 4 images for each of the 2,550 objects (10,200 images). We follow the standard evaluation

**Table 9.7** Instance level image retrieval accuracy on the Holidays dataset, and the UKB dataset. NCMMIL is compared to a PCA baseline and the JSCL method [16]

Dim.	INRIA Holidays dataset				UKB dataset		
	No projection: 77.4 %				No projection: 3.19		
	PCA	JSCL	NCM	NCM*	PCA	JSCL	NCM
32	61.3	67.7	69.3	63.3	2.82	3.04	3.07
64	68.0	73.6	75.4	68.8	3.01	3.23	3.23
128	72.3	76.4	79.6	73.1	3.08	3.31	3.33
256	75.0	78.3	80.2	74.0	3.15	3.36	3.32
512	76.8	78.9	80.6	73.5	3.18	3.36	3.31

protocol, where each image is used as query to search in the database. The performance is measured by  $4 \times \text{recall}@4$  averaged over all queries, hence the maximal score is 4. For both datasets, we extract the 4K image features also used in our earlier experiments, which are the same ones as those used in [16]. To compute the distance between two images, we use the cosine-distance, that is, the dot-product on  $\ell_2$ -normalized vectors.

In analogy to [16], we use NCMMIL to train a metric from the ILSVRC’10 data set, and do the early stopping based on retrieval performance. To avoid tuning on the test data, the cross-validation is performed on the other dataset, that is, when testing on UKB and we regularize based on Holidays and vice-versa. In Table 9.7, we compare the performance of the NCM based metric with that of JSCL, and also include a baseline PCA method and the performance using the high-dimensional descriptors without any projection. Finally, for the Holidays dataset we included the NCM metric while using early-stopping based on classification performance on the ILSVRC’10 validation data set (NCM-class).

From the results, we observe that the NCM metric yields similar performance gains as the JSCL method on both datasets. In both cases, a projection to only 128 dimensions yields an equal or better retrieval performance as using the original 4K dimensional features. On the Holidays dataset, the NCM metric outperforms the JSCL metric, while on the UKB dataset JSCL slightly outperforms NCM. Both the NCM and JSCL methods are effective to learn a projection metric for instance level retrieval, employing class level labels, and outperform the unsupervised PCA projection.

Note that it is crucial to use retrieval performance for early stopping; without it the results (see NCM\*) are in fact worse than the original descriptors, and comparable to using PCA. Thus, the classification objective determines a good “path” through the space of projection matrices, yet it is crucial to regularize for retrieval performance, where the selected number of iterations is typically an order of magnitude smaller than for classification. We explain this discrepancy by the fact that instance level retrieval does not require the suppression of the within class variations needed for good classification. This observation suggests also that even better metrics may be learned by training NCM on a large set of queries with corresponding matches.

## 9.6 Conclusions

In this chapter, we have considered distance-based classifiers for large-scale image classification. The advantage of distance-based classifiers is that new data (possibly of new classes) can be integrated at a negligible cost. This advantageous property is not shared by the one-vs.-rest SVM approach that is used in most current state-of-the-art approaches, but is essential when dealing with real-life open-ended datasets where new images and classes are continuously added over time.

Since the performance of distance-based classifiers is heavily dependent on the used metric, we employ supervised metric learning techniques to improve classification performance. For k-NN classifiers, we rely on the large margin nearest neighbor (LMNN) framework, and consider several variants to select the set of target neighbors. For the NCM classifier, we have introduced a new metric learning technique, which we coined NCMLL. It is based on maximizing the log-likelihood of correct prediction using a soft-min over the distances to class centers to define the class probabilities of a sample. Moreover, we introduced the NCMC classifier, a non-linear extension of the NCM classifier, that uses multiple centroids to represent each class. The used number of centroids offers a complexity trade-off from the linear NCM classifier to the non-linear and non-parametric k-NN classifier where all samples are used as a class centroids.

We have experimentally compared the k-NN and NCM classifiers to a strong baseline, the one-vs.-rest SVM approach. Using a high-dimensional Fisher vector image representation, our baseline attains current state-of-the-art results. Surprisingly we found that the NCM classifier outperforms the more flexible k-NN approach. Moreover, the performance of the NCM classifier is comparable to that of SVM baseline (slightly better with 4K dimensional features, but somewhat worse using the 64K dimensional features), while projecting the data to as few as 256 dimensions. Furthermore, we find that using multiple centroids per class improves the performance of the NCM classifier.

Our learned metrics generalize well to unseen classes, as shown by the experiments where the metric is learned on a subset of 800 classes and evaluated on the 200 held out classes. In this case we observe only a modest drop in performance, in spite of the fact that for the held-out classes “training” consists only in computing the mean vector of each class. These results are further corroborated by our experiments on the ImageNet-10K dataset, where we obtain competitive performance at a negligible cost compared to the feature extraction process. We only need to compute the class means, as opposed to training 10,000 binary one-vs.-rest SVM classifiers, which represents roughly a 8,500 fold speedup.

In addition, we have shown that our NCM classifiers can be used in a zero-shot setting where no training images are available for novel classes, and that the zero-shot prior significantly improves performance when combined with a class mean estimated from a limited amount of training images.

Finally, we have shown that NCM provides a unified way to treat classification and retrieval problems, as query-by-example image retrieval can be seen as a classification problem where only a single positive sample per class is provided. We have

evaluated the NCM metric for image retrieval and found performance that is comparable to previous published metric learning approaches.

## References

1. Bai B, Weston J, Grangier D, Collobert R, Qi Y, Sadamasa K, Chapelle O, Weinberger K (2010) Learning to rank with (a lot of) word features. *Inf Retr* 13(3):291–314 (special issue on learning to rank)
2. Bengio S, Weston J, Grangier D (2011) Label embedding trees for large multi-class tasks. In: *NIPS*
3. Bottou L (2010) Large-scale machine learning with stochastic gradient descent. In: *COMPSTAT*
4. Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge University Press, Cambridge
5. Chai J, Liua H, Chenb B, Baoa Z (2010) Large margin nearest local mean classifier. *Signal Process* 90(1):236–248
6. Checkik G, Sharma V, Shalit U, Bengio S (2010) Large scale online learning of image similarity through ranking. *J Mach Learn Res* 11:1109–1135
7. Clinchant S, Csurka G, Perronnin F, Renders J-M (2007) XRCE’s participation to ImagEval. In: ImageEval workshop at CVIR
8. Csurka G, Dance C, Fan L, Willamowski J, Bray C (2004) Visual categorization with bags of keypoints. In: *ECCV* int workshop on stat learning in computer vision
9. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L (2009) ImageNet: a large-scale hierarchical image database. In: *CVPR*
10. Deng J, Berg A, Li K, Fei-Fei L (2010) What does classifying more than 10,000 image categories tell us? In: *ECCV*
11. Fei-Fei L, Fergus R, Perona P (2006) One-shot learning of object categories. *IEEE Trans Pattern Anal Mach Intell* 28(4):594–611
12. Gao T, Koller D (2011) Discriminative learning of relaxed hierarchy for large-scale visual recognition. In: *ICCV*
13. Gauvain J-L, Lee C-H (1994) Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Trans Speech Audio Process* 2(2):291–298
14. Globerson A, Roweis S (2006) Metric learning by collapsing classes. In: *NIPS*
15. Goldberger J, Roweis S, Hinton G, Salakhutdinov R (2005) Neighbourhood component analysis. In: *NIPS*
16. Gordo A, Rodríguez J, Perronnin F, Valveny E (2012) Leveraging category-level labels for instance-level image retrieval. In: *CVPR*
17. Gray R, Neuhoff D (1998) Quantization. *IEEE Trans Inf Theory* 44(6):2325–2383
18. Guillaumin M, Mensink T, Verbeek J, Schmid C (2009) TagProp: discriminative metric learning in nearest neighbor models for image auto-annotation. In: *ICCV*
19. Guillaumin M, Verbeek J, Schmid C (2009) Is that you? Metric learning approaches for face identification. In: *ICCV*
20. Jégou H, Douze M, Schmid C (2008) Hamming embedding and weak geometric consistency for large scale image search. In: *ECCV*
21. Jégou H, Douze M, Schmid C (2011) Product quantization for nearest neighbor search. *IEEE Trans Pattern Anal Mach Intell* 33(1):117–128
22. Jégou H, Perronnin F, Douze M, Sánchez J, Pérez P, Schmid C (2012) Aggregating local image descriptors into compact codes. *IEEE Trans Pattern Anal Mach Intell* 34(9):1704–1716
23. Köstinger M, Hirzer M, Wohlhart P, Roth P, Bischof H (2012) Large scale metric learning from equivalence constraints. In: *CVPR*
24. Lampert C, Nickisch H, Harmeling S (2009) Learning to detect unseen object classes by between-class attribute transfer. In: *CVPR*

25. Larochelle H, Erhan D, Bengio Y (2008) Zero-data learning of new tasks. In: AAAI conference on artificial intelligence
26. Le Q, Ranzato M, Monga R, Devin M, Chen K, Corrado G, Dean J, Ng A (2012) Building high-level features using large scale unsupervised learning. In: ICML
27. Lin Y, Lv F, Zhu S, Yang M, Cour T, Yu K, Cao L, Huang T (2011) Large-scale image classification: fast feature extraction and SVM training. In: CVPR
28. Lowe D (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110
29. Lucchi A, Weston J (2012) Joint image and word sense discrimination for image retrieval. In: ECCV
30. Mensink T, Verbeek J, Perronnin F, Csurka G (2012) Metric learning for large scale image classification: generalizing to new classes at near-zero cost. In: ECCV
31. Mensink T, Verbeek J, Perronnin F, Csurka G (2013) Distance-based image classification: generalizing to new classes at near-zero cost. *IEEE Trans Pattern Anal Mach Intell* (to appear)
32. Nistér D, Stewénius H (2006) Scalable recognition with a vocabulary tree. In: CVPR
33. Nowak E, Jurie F (2007) Learning visual similarity measures for comparing never seen objects. In: CVPR
34. Parameswaran S, Weinberger KQ (2010) Large margin multi-task metric learning. In: NIPS
35. Perronnin F, Sánchez J, Mensink T (2010) Improving the Fisher kernel for large-scale image classification. In: ECCV
36. Perronnin F, Akata Z, Harchaoui Z, Schmid C (2012) Towards good practice in large-scale learning for image classification. In: CVPR
37. Rohrbach M, Stark M, Schiele B (2011) Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In: CVPR
38. Saenko K, Kulis B, Fritz M, Darrell T (2010) Adapting visual category models to new domains. In: ECCV
39. Sánchez J, Perronnin F (2011) High-dimensional signature compression for large-scale image classification. In: CVPR
40. Tommasi T, Caputo B (2009) The more you know, the less you learn: from knowledge transfer to one-shot learning of object categories. In: BMVC
41. Veenman C, Tax D (2005) LESS: a model-based classifier for sparse subspaces. *IEEE Trans Pattern Anal Mach Intell* 27(9):1496–1500
42. Webb AR (2002) Statistical pattern recognition. Wiley, New York
43. Weinberger KQ, Chapelle O (2009) Large margin taxonomy embedding for document categorization. In: NIPS
44. Weinberger K, Saul L (2009) Distance metric learning for large margin nearest neighbor classification. *J Mach Learn Res* 10:207–244
45. Weinberger K, Blitzer J, Saul L (2006) Distance metric learning for large margin nearest neighbor classification. In: NIPS
46. Weston J, Bengio S, Usunier N (2011) WSABIE: scaling up to large vocabulary image annotation. In: IJCAI
47. Zhang J, Marszałek 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
48. Zhou X, Zhang X, Yan Z, Chang S-F, Hasegawa-Johnson M, Huang T (2008) Sift-bag kernel for video event analysis. In: ACM multimedia

# Chapter 10

## Top–Down Bayesian Inference of Indoor Scenes

Luca Del Pero and Kobus Barnard

**Abstract** The task of inferring the 3D layout of indoor scenes from images has seen many recent advancements. Understanding the basic 3D geometry of these environments is important for higher level applications, such as object recognition and robot navigation. In this chapter, we present our Bayesian generative model for understanding indoor environments. We model the 3D geometry of a room and the objects within it with non-overlapping 3D boxes, which provide approximations for both the room boundary and objects like tables and beds. We separately model the imaging process (camera parameters), and an image likelihood, thus providing a complete, generative statistical model for image data. A key feature of this work is using prior information and constraints on the 3D geometry of the scene elements, which addresses ambiguities in the imaging process in a top–down fashion. We also describe and relate this work to other state-of-the-art approaches, and discuss techniques that have become standard in this field, such as estimating the camera pose from a triplet of vanishing points.

### 10.1 Introduction

There has been much recent interest in estimating the 3D layout of indoor scenes from monocular images [4, 9–11, 15, 16, 20, 24], as this provides crucial geometric context for higher level tasks, such as object recognition and prediction of human activities. Consider for example, Fig. 10.1. From just a single image, human observers can infer the 3D structure of the room, even when many cues are hidden. For example, we can estimate the boundary between the floor and the walls, even if it is mostly occluded by furniture. This kind of geometric understanding informs interaction with the environment. For example, one could infer that the bed provides a surface for sitting, or that a possible path to the couch goes around the bed.

---

L. Del Pero (✉) · K. Barnard  
University of Arizona, Tucson, USA  
e-mail: [delpero@email.arizona.edu](mailto:delpero@email.arizona.edu)

K. Barnard  
e-mail: [kobus@sista.arizona.edu](mailto:kobus@sista.arizona.edu)

**Fig. 10.1** Inferring the 3D layout of a room from an image is challenging due to occlusions and clutter. Here, the boundary between floor and walls is occluded by the furniture. We show on the right the correct position of the 3D box approximating the room



A number of computational methods for recovering the 3D scene layout have been recently developed based on modeling rooms with simple geometric primitives, such as sets of orthogonal surfaces [15], or 3D boxes [9, 16]. Despite the coarse geometric approximation, these approaches have enabled several interesting applications. For example, Gupta et al. [7] used extracted 3D information to identify locations where people can sit or lie, while Hedau et al. [10] showed how knowledge of the 3D environment helps detecting pieces of furniture, such as beds. Other notable applications include inserting realistic computer graphics objects into indoor images [14], and robot navigation [22].

This chapter extends our previous work on indoor scene understanding [2, 3]. Specifically, we provide additional details on the model and the inference discussed in these papers. New contributions include making the inference multi-threaded by allowing threads to exchange information (Sect. 10.3.4), a new method to make the inference more robust (Sect. 10.3.1.4), and a comprehensive evaluation of all the techniques proposed (Sect. 10.4).

### 10.1.1 Background

In the domain of inferring 3D geometry from indoor images, it is common to parametrize the scene layout as a 3D box [9–11, 16, 20, 24]. This is often referred to as *room box*, as it coarsely models the space of a room as if it were empty. An example is shown in Fig. 10.1, where the estimated room box is shown on the right. Inferring the room box from an image entails determining the 3D position and orientation of the floor, ceiling and walls that typically define an indoor environment. This process presents two main challenges. First, recovering the 3D box requires inferring the perspective transformation that generated the image (camera estimation). Second, clutter and occlusions are a major source of confusion, since the room box boundaries are often mostly hidden (Fig. 10.1).

Estimating the camera in indoor scenes has been tackled by using a strong model [9–11, 16, 20, 24], specifically the Manhattan world assumption [1]. This states that most surfaces in the scene are aligned with three principal orthogonal directions, which can be estimated by detecting a triplet of orthogonal vanishing

points on the image plane [18]. The parameters of the perspective projection can then be recovered analytically from the position of the vanishing points, and this is a well understood problem [8]. In Manhattan world indoor scenes, the three orthogonal directions are invariably those of the room box.

Addressing occlusions and clutter requires reasoning about the objects in the room. Hedau et al. [9] detect clutter in 2D with an appearance based classifier. More recently, Lee et al. [15] showed the benefits of reasoning about objects in 3D. Specifically, they proposed modeling objects as 3D boxes, which provide reasonable bounding approximations of objects typically found in rooms, such as beds and tables, and further exploit the Manhattan assumption by constraining these boxes to be aligned with the room box. Their results showed that joint inference of the room and the 3D objects, which addresses ambiguities caused by clutter and occlusions in a top–down fashion, improves on estimating the room box. Our approach is related to Lee’s work [15], but we advocate an even more top–down approach with a more unified representation.

### ***10.1.2 Overview of Our Approach***

Similarly to Lee et al. [15], our goal is to simultaneously estimate the camera, detect and localize in 3D the floor, ceiling, and walls comprising the room “box”, and determine the position of the objects in the room. However, instead of using an object box only to explain occlusions, we want to identify it (e.g., a couch or a bed) as well. This simultaneously achieves a fuller understanding of the scene, and allows object knowledge to help fit the overall geometry. For example, knowing that a specific block is approximating a bed rather than, say, a wardrobe, adds constraints on the box’s 3D size and position, as a bed is typically much lower. Conversely, the size and position of a 3D block provide strong cues on the identity of the object.

Our goal is to provide a comprehensive parsing of the scene that is globally consistent in 3D, both geometrically and semantically, which is on the lines of the work of Hoiem et al. [13]. This introduces complex constraints, which make the inference process challenging. Examples of constraints include preventing objects from overlapping in 3D, and enforcing that each object is contained in the room box. Previous work [9, 16, 20] relied on the framework of structured prediction for inferring indoor scene models, where inference and modeling are strongly coupled. This makes it harder to adapt the method to handle more complex sources of information, such as conditioning the position and size of objects on their identity.

Bayesian inference is a natural way to handle these complexities, as it allows to separate the modeling and the inference. Specifically, we propose a Bayesian generative model for images of indoor scenes, where we separately model the 3D geometry (room box and objects), the imaging process (camera parameters), and an image likelihood. We assume that the image features are generated statistically from the projected 3D scene under the camera parameters. We impose further structure by

introducing the notion of an object type. The Bayesian framework naturally allows using prior information of the world, which in our case are prior distributions on a box’s 3D size and position conditioned on its type and on the room box. For example, beds are typically against a wall, while tables are likely found in the center of the room. In this work, we allow four different types of object boxes, approximating beds, cabinets, couches and tables. We also introduce the notion of *frames*, which are thin boxes “anchored” to a wall, to approximate objects such as doors. We use three types of frames: doors, picture frames, and windows.

Bayesian inference allows us to jointly infer all the elements in our model, without having to commit to partial solutions. For example, previous work [9–11, 16, 20, 24] rely on initial estimates of the camera parameters, and they cannot recover from mistakes in this step. Similarly, both Hedau et al. [9] and Gupta et al. [7] use an initial estimate of the room box for identifying beds and predicting human activities in the room, respectively. Again, errors in the estimate of the room box cannot be recovered from. In principle, our approach does not have this problem.

In what follows, we detail with our model for the 3D scene geometry and the camera (Sect. 10.2). We then develop priors that distinguish among object boxes based on their position and size (Sect. 10.2.1), and then detail the imaging model (Sect. 10.2.2), including an analysis of the standard image features used in this field. We then describe the Markov chain Monte Carlo sampling method that we use for inference (Sect. 10.3). Important aspects include how to handle constraints during sampling, how to use data-driven methods for efficient sampling, and how to do inference with multiple threads. Finally, we provide extensive evaluation of our approach on two standard datasets (Sect. 10.4).

## 10.2 A Bayesian Generative Model for Indoor Scenes

We use a Bayesian generative model, where we assume that images are generated by the projection of the 3D scene. We partition model parameters,  $\theta$ , into scene parameters,  $s$ , encoding the 3D geometry, and camera parameters,  $c$ , modeling the perspective transformation

$$\theta = (s, c). \quad (10.1)$$

We define the posterior distribution as

$$p(\theta|D) \propto p(D|\theta)p(\theta), \quad (10.2)$$

where  $D$  are features detected on the image plane and  $p(\theta)$  is the prior distribution over model parameters.

Scene parameters include the room box and objects in it

$$s = (r, o_1, \dots, o_n), \quad (10.3)$$

where the number of objects  $n$  and their type are not known a priori. We model the room as a right-angled parallelepiped whose floor is parallel to the  $x$ – $z$  plane of the world reference frame. It is defined by its 3D center, width, height, length, and rotation angle,  $\gamma_r$ , around an axis parallel to the world  $y$ -axis and through the room center (yaw):

$$r = (x_r, y_r, z_r, w_r, h_r, l_r, \gamma_r). \quad (10.4)$$

Objects in the room are similarly modeled by blocks, but include an object category,  $t_i$ , (e.g., bed, table, door). A block on the floor could approximate a convex object such as a bed, or provide a bounding box for a more complex object, such as a table. Windows, doors and pictures are approximated with thin blocks (frames) attached to a wall. All objects share the same orientation  $\gamma_r$  of the room block, following the Manhattan world assumption. Objects must be entirely inside the room box, and they can not intersect each other.

Objects coordinates are relative to the coordinate frame defined by the room center, and whose axes are aligned with the room walls, which we call “room coordinates”. We define a coordinate transformation function  $r_{\text{coord}}(x, y, z) = (x^r, y^r, z^r)$  for later use, to transform a point defined relatively to the world coordinate system  $(x, y, z)$  into room coordinates. Since the world  $x$ – $z$  plane is parallel to the room floor,  $r_{\text{coord}}$  simply applies a translation and a rotation around the  $y$ -axis defined by the room yaw  $\gamma_r$ . An advantage of storing objects in room coordinates, is that it allows for efficient computation of intersections among objects, and between an object and the room box.

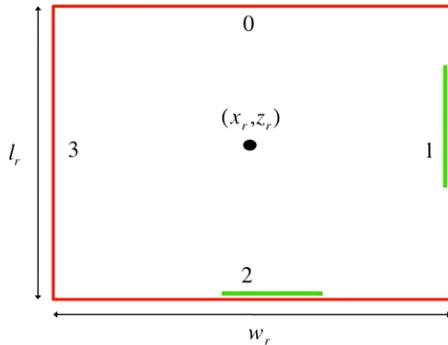
Each object is “anchored” to a room surface, whose index is stored as a discrete variable ( $s_i$ ). Furniture objects lie on the floor ( $s_i = 4$ ), implying that, given the object height, the  $y$  coordinate of the object center is not a free parameter. Specifically,  $y_i = -(y_r/2) + (h_i/2.0)$ , where  $-(y_r/2)$  is the position of the floor in room coordinates. Similarly, frames are anchored to one of the walls, which analogously constrains their parameters (see Fig. 10.2). To summarize the object parameters,

$$o_i = (t_i, s_i, x_i, y_i, z_i, w_i, h_i, l_i). \quad (10.5)$$

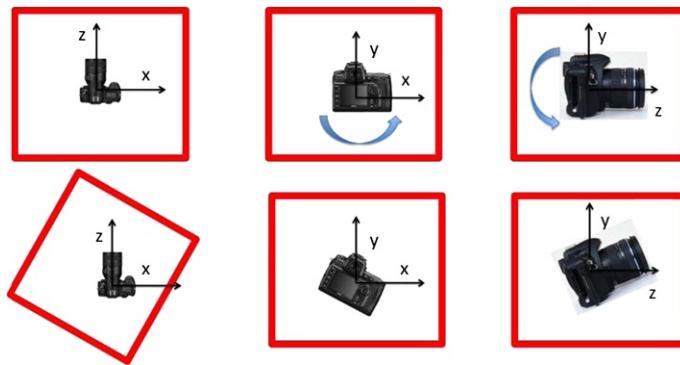
The imaging process is modeled with a standard perspective camera

$$c = (f, \phi, \psi), \quad (10.6)$$

where  $f$ ,  $\phi$  and  $\psi$  are, respectively, the focal length, the pitch, and the roll angle. Since absolute positions cannot be determined when reconstructing from single images, we arbitrarily position the camera at the origin of the world coordinate system, pointing down the  $z$ -axis. The extrinsic camera rotations, specified by three degrees of freedom, are fully determined by  $\phi$ ,  $\psi$  and the yaw  $\gamma_r$  of the room (see Fig. 10.3). Pitch and roll are constrained within ranges of plausible values for indoor scenes ( $\phi \in [-60^\circ, 60^\circ]$ ,  $\psi \in [-10^\circ, 10^\circ]$ ), while the focal length has to be positive. We further assume unary aspect ratio, no skew, and that the principal point coincides with the image center.



**Fig. 10.2** The coordinates of the objects are stored in a coordinate frame relative to the center of the room box, and whose axes are aligned with the room walls. Here, two frames are “anchored” to a wall of the room box, seen from above. Walls are numbered from 0 to 3, and this index is stored in variable  $s_i$ , which imposes constraints on the parameters in Eq. 10.5. For example, for the frame on the *right*, we have  $s_i = 1$  and  $x_i = (w_r/2)$ . Further, since frames are approximated with thin blocks, we set  $w_i = \varepsilon$ , with  $\varepsilon = 0.01$  units. For the frame at the *bottom*,  $s_i = 2$ ,  $z_i = (l_r/2)$ , and  $l_i = \varepsilon$



**Fig. 10.3** Camera parameters. The extrinsic parameters define the position and orientation of the camera with respect to the world reference frame. Since absolute sizes and position cannot be determined, we arbitrarily position the camera at the origin of the world coordinate system, pointing down the negative  $z$ -axis. The room box can rotate around its  $y$ -axis, thus determining the yaw of the camera (*left column*). Two more angles, a rotation around the camera  $z$ -axis (*mid column*) and a rotation about the  $x$ -axis (*pitch, right column*) complete the camera orientation specification

### 10.2.1 Model Priors

Priors on scene elements improve global scene understanding by helping resolve ambiguity during inference, and also support identifying objects based on geometry cues, such as size and location, alone. Previous work has used blocks in the scene to explain occlusions [16] and to infer what regions of the 3D space are occupied by generic objects [11]. In this work, we assign a semantic label to our blocks

(e.g., couch, door, etc.), with each label corresponding to specific prior probabilities on object size and position in 3D. Because we are reconstructing from a single image, we have one overall scale ambiguity, and thus priors on object “size” and position are defined relatively to the room box.

As an example of information captured by the priors, wardrobe cabinets are tall and narrow and typically against a wall, while tables are usually shorter, wider, and in the middle of the room. Similarly, a door is quite tall and touches the floor, while picture frames are much shorter and are typically found in the higher half of a wall.

Assuming independence, the overall prior for the model parameters is given by

$$p(\theta) = \pi(r)\pi(c) \prod_{i=1}^n \pi(o_i), \quad (10.7)$$

where  $\pi(r)$  is the prior on the room box,  $\pi(c)$  is the prior on camera parameters, and  $\pi(o_i)$  is the prior for one of  $n$  objects in the room. We now describe each of these components and how we set their parameters from training data.

### 10.2.1.1 Prior on Room Box

The room box is defined in terms of the center position in 3D ( $x_r, y_r, z_r$ ) and its width, height, and length ( $w_r, h_r, l_r$ ). First, we define a prior over the ratio between the long dimension to the short dimension, with

$$r_{r1} = \frac{\max(w_r, l_r)}{\min(w_r, l_r)}. \quad (10.8)$$

We use this formulation since we do not know in advance which dimension is the largest. We also put a prior on the ratio of the long dimension to the height

$$r_{r2} = \frac{\max(w_r, l_r)}{h_r}. \quad (10.9)$$

The prior distributions over these two quantities are set to be relatively non-informative, but help reduce the time spent in regions of the sampling space with low probability, especially during the early stages of the inference process. We set both parameters to be normal distributions, independent from each other

$$\pi(r) = \mathcal{N}(r_{r1}, \mu_{r1}, \sigma_{r1})\mathcal{N}(r_{r2}, \mu_{r2}, \sigma_{r2}). \quad (10.10)$$

### 10.2.1.2 Prior on Camera Parameters

We found that the camera height from the floor  $c_h$  is a particularly indicative property in indoor scenes, as small variations in this quantity result in major changes in the image plane. Intuitively, pictures of indoor images are rarely taken by putting



**Fig. 10.4** Distinguishing objects using their size. The ratio between the height of an object and the largest between its width and length varies considerably between beds and cabinets. In this example, also the ratio between width and length is quite discriminative

the camera close to the floor or to the ceiling. Since we cannot use absolute sizes, the prior is defined on the ratio  $r_{\text{ch}}$  between camera height and room height

$$\pi(c) = \mathcal{N}(r_{\text{ch}}, \mu_{\text{ch}}, \sigma_{\text{ch}}). \quad (10.11)$$

### 10.2.1.3 Prior on Objects

Several categories of furniture and frames have a very distinctive size (Fig. 10.4). Here we introduce a general formulation for a prior for a specific object of category  $t_i = \tau$  that exploits this intuition. Given an object  $o_i$  defined in terms of its size  $(w_i, h_i, l_i)$ , and a room with dimensions  $(w_r, h_r, l_r)$ , we use the following quantities

- Ratio between the object height and its largest other dimension  $r_{i1} = h_i / \max(w_i, l_i)$  (Fig. 10.4). This helps distinguish between categories that are taller than they are wide, such as wardrobes, and short and wide objects, such as beds.
- Ratio between the object long and short dimensions,  $r_{i2} = \max(w_i, l_i) / \min(w_i, l_i)$  (Fig. 10.4). Again, we use this formulation since we do not know in advance which dimension is the largest. This quantity discriminates between furniture with a roughly square base, and furniture with a rectangular base. This component is not used for frames, since one of their width or length is always negligible.
- Ratio between room height and object height  $r_{i3} = h_r / h_i$ . Intuitively, the height of a bed is quite small with respect to the room height, whereas the height of a wardrobe or of a door is quite large (Fig. 10.5).
- Whether the object is against a wall or not. This is based on the intuition that some objects tend to be against a wall (e.g., beds) more than others (tables). For frames, we use whether or not the frame touches the floor. For example, doors touch the floor, while windows typically do not.



**Fig. 10.5** Learning the ratio between room height and object height. This quantity is very informative, and we estimate its statistics from training images. We use the ratio between the two arrows, provided that the object is against or close to a wall. While ratios of lengths of collinear segments are normally not preserved by projective transformations (only affine), in this domain the vanishing point for vertical segments is usually at infinity, and this method provides a reasonable approximation

The first two ratios carry information on the object structure, and do not depend on the scene, while the last two encode information on the size and position of an object relatively to the room box. The first three quantities follow a normal distribution

$$\pi_j(o_i|t_i = \tau) = \mathcal{N}(r_{ij}; \mu_{\tau j}, \sigma_{\tau j}), \quad (10.12)$$

for  $j = 1, 2, 3$ . Each category  $\tau$  has different  $(\mu_{\tau j}, \sigma_{\tau j})$ , and for object  $o_i$  we use the prior distribution for the category it belongs to, denoted by  $t_i$ . Notice that from now on we will use the shorthand  $\pi_j(o_i)$  for  $\pi_j(o_i|t_i = \tau)$ . Last,  $d_i$  follows a Bernoulli distribution  $\pi(d_i)$ . Given an object  $o_i$ , we combine the components of its prior probability as follows

$$\pi(o_i) = \pi(d_i) \prod_{j=1}^3 \pi_j(o_i). \quad (10.13)$$

#### 10.2.1.4 Setting Prior Probabilities from Data

As mentioned above, the first two components of the object prior do not depend on the scene. For each category  $\tau$ , we set  $(\mu_{\tau 1}, \sigma_{\tau 1}, \mu_{\tau 2}, \sigma_{\tau 2})$  using fifty random examples selected from online furniture and appliances catalogs. We recorded their dimensions, provided in the text description, and the means and variances of the relevant ratios. We used the Ikea catalog<sup>1</sup> for beds, couches, cabinets and tables, and the Home Depot catalog<sup>2</sup> for windows, doors and picture frames.

Setting the parameters for the remaining two priors is more challenging, since they relate the size of an object category to that of the room, and this information is

<sup>1</sup><http://www.ikea.com/us/en/catalog/categories/departments/bedroom/>

<sup>2</sup><http://www6.homedepot.com/cyber-monday/index.html>

not available in furniture catalogs. In this case, we use image data, and set  $(\mu_{c3}, \sigma_{c3})$  as explained in Fig. 10.5. We also use images to set  $\pi(d)$ , which we approximate as the frequency at which an instance of an object of category  $\tau$  is against a wall, or floor if it is a frame. For training, we used the images in the training split of the Hedau dataset [9], omitting images where we could not tell whether a piece of furniture was against the wall or not.

Last, we set the parameters of the priors on camera and room box from training images. We manually fit an empty room box and the camera to images in the Hedau training set, from which we set the parameters [3].

### 10.2.2 The Image Model

Our image model is similar to the one used by Schlecht and Barnard [19]. Specifically, we assume that image features  $D = (f_1, \dots, f_s)$  are generated by the projection of the 3D scene under the given camera. We use three feature types that proved useful in this domain, specifically edges [2, 3, 15], orientation surfaces [15, 16], and geometric context [9, 16].

#### 10.2.2.1 Image Edges

We assume image edges to be generated by the blocks in the scene. We measure the quality of a fit by comparing the set of edges  $E_d$  detected on the image plane to the set of edges  $E_\theta$  generated by projecting the model. As in Schlecht et al. [19], we define a likelihood function  $p(E_d|E_\theta)$ , which we specify using the following intuitions:

- An edge point  $e_{dj} \in E_d$  detected in the image plane should be matched to an edge point  $e_{\theta k} \in E_\theta$  generated by the model. If the match is good the two points should be very close to each other, and the difference in orientation between the two edges should be minimal. We use  $p(e_{dj}|e_{\theta k}) = \mathcal{N}(d_{jk}, 0, \sigma_d)\mathcal{N}(\phi_{jk}, 0, \sigma_\phi)$ , where  $d_{jk}$  is the distance between the points, and  $\phi_{jk}$  the difference in orientation between the edges.
- We penalize a detected edge point that is not matched to any model edge (noise). We define  $p_n$  as the probability of such an event occurring.
- We explain points in  $E_\theta$  not matched to any point in  $E_d$  as missing detections, and define probabilities  $p_{\text{hmiss}}$  and  $p_{\text{smiss}}$ . The former is used for “hard” edges arising from occlusion boundaries, such as the edges that belong to the silhouette of an object. The latter is used for “soft” edges that are less likely to be found by the edge detector, such as the room edges and non-silhouette edges from objects. It is less likely that a detector will miss a “hard” edge compared with a “soft” edge. Note that one of the advantages of using a full 3D model, is that we can determine whether hypothesized edge points in  $E_\theta$  are hard or soft.

We then have

$$\tilde{p}(E_d|E_\theta) = p_n^{N_n} p_{\text{smiss}}^{N_{\text{smiss}}} p_{\text{hmiss}}^{N_{\text{hmiss}}} \prod_{(j,k) \in \text{matches}} p(e_{dj}|e_{\theta k}), \quad (10.14)$$

where  $N_n$  is the number of edge points labeled as noise, and  $N_{\text{smiss}}(N_{\text{hmiss}})$  the number of missed soft (hard) edges. We match points in a greedy fashion by finding the closest point  $e_\theta$  to a data edge  $e_d$  along the edge gradient, provided that this distance is smaller than 40 pixels, and the difference in orientation is less than 0.7 radian. We further adjust this likelihood function to make it less sensitive to the number of edge points, which we found makes it more stable over a larger variety of input data. Specifically, we use

$$p(E_d|E_\theta) \approx \tilde{p}(E_d|E_\theta)^{(N_{\text{hmiss}} + N_{\text{smiss}} + N_n + N_{\text{matches}})^{-1}}. \quad (10.15)$$

### 10.2.2.2 Orientation Surfaces

Based on the Manhattan world assumption, most pixels in the scene are generated by a plane aligned with one of three orthogonal directions, and we can estimate which one using the approach by Lee et al. [15]. We compare the pixel orientation  $O_d$  detected from the image plane with the orientation surfaces  $O_\theta$  generated by projecting our model. We approximate  $p(O_d|O_\theta)$  by the ratio between the number of pixels where the orientation detected on the image plane agrees with the orientation predicted by the model, and the total number of pixels.

### 10.2.2.3 Geometric Context

Following Hedau et al. [9], we also consider geometric context labels, which estimate the geometric class of each pixel, choosing between object, floor, ceiling, left, middle and right wall. This is done using a probabilistic classifier trained on a mixture of color and oriented gradient features [12]. We use the code and the pre-trained classifier available online [12]. For each pixel  $p_k$ , this provides a probability distribution  $g_{ck} = [g_{ck1}, \dots, g_{ck6}]$  over the six classes. Given the label  $l$  predicted by the model for pixel  $p_k$ , we define  $p(g_{ck}|p_k) = p(g_{cl}|p_k = l) = g_{cl}$ , and

$$p(GC|\theta) = \frac{\sum_{p_k \in I} p(g_{ck}|p_k)}{\text{size}(I)}, \quad (10.16)$$

where we average the contributions of all image pixels. Since the available classifier was trained against data where only furniture was labeled as objects, and not frames, we consider frames as part of the wall they are attached to, and not as objects when we evaluate on geometric context.

#### 10.2.2.4 Combining the Three Features

Assuming independence among the features, we define our likelihood function

$$p(D|\theta) = p(E_d|E_m)p(O_d|O_m)^\alpha p(GC|\theta)^\beta, \quad (10.17)$$

where  $\alpha$  and  $\beta$  weigh the importance of the orientation and geometric context likelihoods, relative to the edge likelihood. We set  $\beta = 12$  and  $\alpha = 6$  by running our algorithm on the training portion of the Hedau dataset [9]. Here we used a coarse grid search over  $\alpha$  and  $\beta$ , with a step of 2, using the room box layout error (defined in Sect. 10.4) as an objective function.

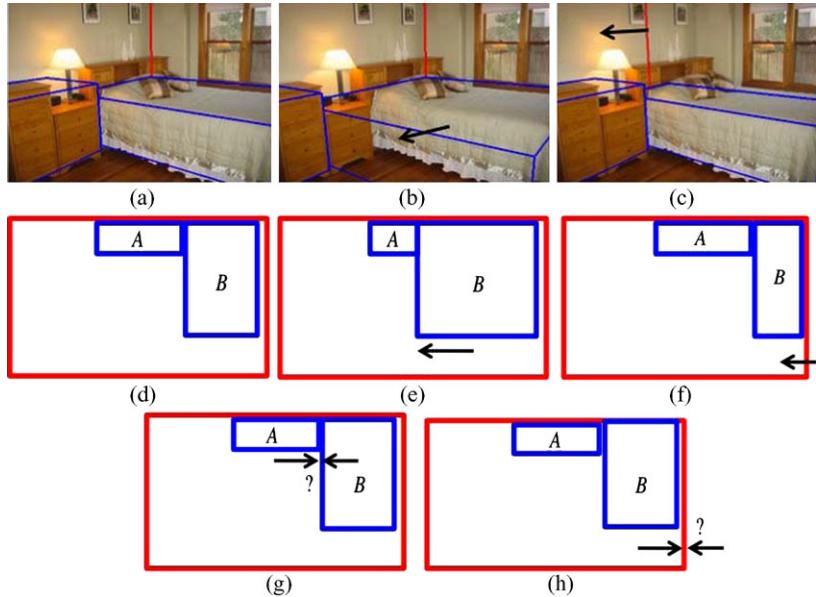
Later, in the results (Fig. 10.18), we illustrate how these three features work together. Errors in the edge detection process can be fixed using orientation surfaces and geometric context, and vice versa. Using edges also helps improving the camera fit when starting from a wrong estimate of the vanishing points, which are detected at the beginning and used to initialize the camera parameters. In fact, since the algorithms for computing orientation maps and geometric context depend on the initial vanishing point estimation, this feature is compromised by this initial error, whereas edges are not.

### 10.3 Inference

We use Markov chain Monte Carlo (MCMC) sampling to search the parameter space, defined by camera and room box parameters, the unknown number of objects, their type, and the parameters of each object. To change the discrete structure of the model, which includes the unknown number of objects, and the type of each of them, we use reversible jump Metropolis-Hastings (MH) [5, 6]. To change the continuous parameters, which comprise the room box, the camera, and size and position of each object, we use Hamiltonian dynamics sampling [17]. The proposals from these two samplings strategies are often referred to as “jump” and “diffusion” moves [23].

The 3D structure of the model introduces several constraints and dependencies among parameters, which must be carefully taken into account during inference. Specifically, objects cannot overlap in 3D, and they have to be entirely inside the room box. Further, the camera must be inside the room box, and not within the volume occupied by any of the objects. Enforcing these constraints during inference can introduce several ambiguities, as illustrated in Fig. 10.6(g), using a birdview of the room box. Consider sampling over objects  $A$  and  $B$  simultaneously. This could result in a conflict where object  $B$  tries to expand towards the left, and  $A$  towards the right, as illustrated by the arrows. A similar situation can happen when the room box is trying to shrink and object  $B$  is trying to expand Fig. 10.6(h).

To avoid these ambiguities, we use three different types of continuous moves over subsets of the scene parameters, and define rules for adapting the model so that no constraint is violated. In the first one, we sample over the parameters of a single object, and we enforce constraints by shrinking other objects in case of overlap, and expanding the room box in case it is not big enough to contain the



**Fig. 10.6** Sampling over subsets of parameters handles containment constraints (top two rows) and avoids ambiguities (bottom row). Consider sampling over the parameters of a single object at a time. As the object expands, other objects have to shrink to avoid overlap. For example,  $A$  has to shrink to allow  $B$  to grow (a)–(b), as shown also in the corresponding birdviews (d)–(e). Similarly, when sampling over the room box only, objects have to shrink so that they are entirely in the room, like object  $B$  in (c) and (f). Instead, sampling over two objects at the same time could result in conflicts such as in (g), where both  $A$  and  $B$  are contending the same 3D space. Sampling over the room box and an object jointly would create similar conflicts (h)

object (Fig. 10.6(a)–(b)). The second move samples over the parameters of the room box only, and adapt the objects to respect the constraints. As the room box becomes smaller, we shrink objects in case they end outside the room (Fig. 10.6(c)). Third, we jointly sample over camera and room box parameters, by enforcing constraints as in the previous move. Further, for all the three moves just discussed, we enforce that the camera is inside the room box, and is not within the volume occupied by an object. These are corner cases that occur less often, and we simply reject samples violating these constraints.

Diffusion moves and Jump moves are alternated throughout inference. We summarize the entire process here, and describe each component in detail in the following sections:

1. Initialize the parameters of the camera and of the room box
2. Repeat  $K$  times
  - a. Generate a new sample with one of the following moves, chosen randomly
    - Jump 1: add an object to the scene (furniture or frame)
    - Jump 2: remove an object from the scene

- Jump 3: pick a random object, and change its type
  - Diffusion 1: pick a random object and sample over its parameters
  - Diffusion 2: sample over the room box parameters only
  - Diffusion 3: sample over room box and camera parameters
- b. Reject samples that violate the constraints on the camera position (camera outside the room or inside an object).
3. Return the sample with the highest posterior

We rely on a multi-threaded strategy to efficiently explore more of the space on modern multi-core workstations. Each thread executes the procedure above, and, at the end, threads are allowed to exchange information, as some of the objects might be found by a thread and missed by the others, and vice versa (Sect. 10.3.4). This exchanging procedure is followed by additional sampling, and we output the best sample found. In our experiments we used 20 threads, and the whole inference process takes on average ten minutes per image.

In what follows, we first detail with the diffusion moves in Sect. 10.3.1, where we develop concepts needed to better discuss the Jump moves Sect. 10.3.2, and how we initialize the room box and the camera Sect. 10.3.3. Last, we explain how to exchange objects among threads Sect. 10.3.4.

### 10.3.1 Diffusion Moves

As illustrated in Fig. 10.6, we sample over subsets of the scene parameters and specify rules to make the model comply with all constraints. We use Hamiltonian dynamics sampling, which were used by Schlecht et al. [19] for learning the continuous parameters of geometric models for furniture with a similar parametrization.

We use Neal’s formulation of Hamiltonian dynamics [17] to sample over phase space, where the energy function is defined in terms of the joint distribution of the parameters and the image data  $p(\theta, D)$

$$E(\theta) = -\log(p(D|\theta)) - \log(p(\theta)). \quad (10.18)$$

We follow the dynamics using leapfrog discretization [17], and compute the derivative of the potential energy with numerical approximation, which is the current bottleneck for computation.

#### 10.3.1.1 Sampling over the Continuous Parameters of an Object (Diffusion 1)

When sampling the parameters of object  $o_i$ , we adapt the room box and the other objects so that no constraints are violated. When we detect an overlap with another object, we shrink the latter, and delete it if it is completely contained in  $o_i$ . We also check whether  $o_i$  is partly outside the room. If this is the case, we expand the room box so that  $o_i$  is entirely inside, and further adjust the position of the other objects (Sect. 10.3.1.2).

To reduce the sampling time, we designed efficient ways for sampling object parameters. For example, consider the window in Fig. 10.7(a). To find the correct fit, the window height must be stretched, and its center must be shifted downwards. To do this efficiently, we vary the height of the window by keeping the upper edge fixed. In this example, we would run Hamiltonian dynamics on the object height  $h_i$ . At each leapfrog step  $t$ , a proposed change in the height  $h_i^t = h_i^{t-1} + \delta$  is followed by changing the y position of the object center as  $y_i^t = y_i^{t-1} - \frac{\delta}{2}$ , achieving the result in Fig. 10.7(b). This technique is effective in our framework, since typically one edge of the object is correctly “latched” to an image edge, given our proposal mechanism from image corners discussed in Sect. 10.3.2.1, and we thus want to find the correct size of the object without displacing that edge. There are four directions to sample a frame using this strategy, and six for furniture objects, as illustrated in Fig. 10.7(c), (d) and (e). When executing move Diffusion 1 for object  $o_i$ , we iterate over all four possible directions if  $o_i$  is a frame, six if it is a furniture object. For each direction, we use 20 leapfrog steps, and at each step we enforce the containment constraints.

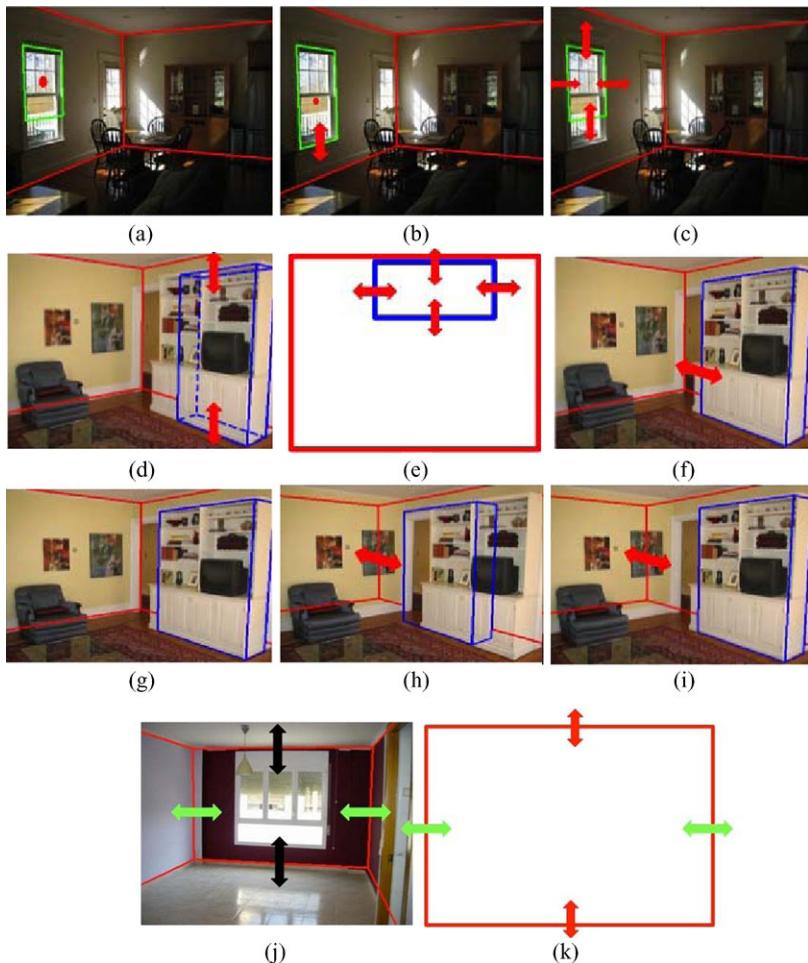
### 10.3.1.2 Sampling over the Continuous Parameters of the Room Box (Diffusion 2)

When sampling the parameters of the room box  $r$ , we adapt all the objects so that no constraints are violated. When we detect that an object is partly outside the box, we shrink it, and delete it if it is completely outside. As for the object parameters, we sample along one direction of the room by keeping one edge “latched” (Fig. 10.7, third row). Six sampling directions are available (fourth row), and we use 20 leapfrog steps per direction. At each leapfrog step, we enforce the containment constraint, and we further apply a transformation to the objects in the room, to preserve their projection on the image plane. Since room objects parameters are relative to the room box coordinate system, changing the 3D position of the room box would not preserve the projection of the room objects. An example is shown in Fig. 10.7(g)–(h), where a small change in the room center causes the object to move as well.

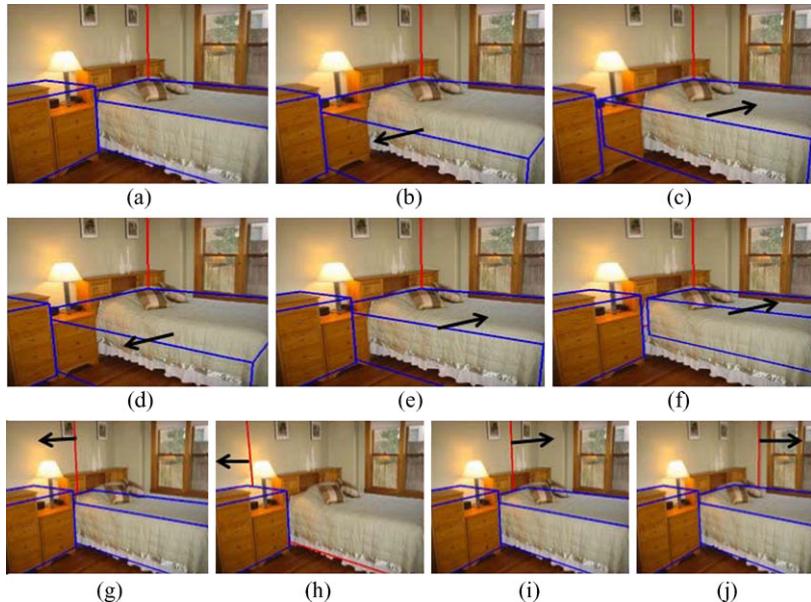
We address this as follows. We define the room 3D center  $(r_x + \delta_{rx}, r_y + \delta_{ry}, r_z + \delta_{rz})$  at leapfrog step  $t$ , where all the parameters are relative to the world coordinate system, and  $(r_x, r_y, r_z)$  is the room center at leapfrog step  $(t-1)$ . Let us also define  $(x_i, y_i, z_i)$  as the center of object  $o_i$  at  $(t-1)$ , this time in room coordinates. For each object  $o_i$ , we compute the position of its center at step  $t$  as  $(x_i + \delta'_{rx}, y_i + \delta'_{ry}, z_i + \delta'_{rz})$ , where  $(\delta'_{rx}, \delta'_{ry}, \delta'_{rz}) = r_{\text{coord}}(\delta_{rx}, \delta_{ry}, \delta_{rz})$ . The result of applying this transformation is shown in Fig. 10.7(i), where the room box has changed, but the object block kept its position on the image plane.

### 10.3.1.3 Sampling over Camera and Room Box Parameters (Diffusion 3)

We sample over camera and room box parameters jointly, for a total of 10 parameters. We found that sampling over the camera parameters independently typically produces samples with a lower posterior, due to correlations with other scene pa-



**Fig. 10.7** Efficient strategies for sampling over continuous parameters. The upper edge of the window in (a) is positioned correctly, and the correct fit for the window can be obtained efficiently by dragging down the lower edge while keeping the upper edge fixed (b). We designed moves that achieve this by sampling the continuous parameters of the window in 3D (see text). For a frame, this principle can be applied to any of the edges, and this creates four possible sampling directions, denoted by the *arrows* in (c). For furniture we have six alternatives, as illustrated in (d) and (e), which is a birdview of the model. (f) is an example of the effects of this move when sampling over the direction denoted by the *arrow*. The same strategy is also used when sampling over the room box (*third row*), where we have six possible directions, illustrated in (j) and (k) (two of those are shown in both, denoted by the *green arrows*). Since object coordinates are relative to the room box, changes in the latter have the undesirable effect of changing the projection of objects already in the room. For example, we only changed the 3D center of the room box from (g) to (h), and this “shifted” the projection of the *blue block* as well. By introducing a transformation that preserves the projection of the objects in the room (see text), we obtain the result in (i)



**Fig. 10.8** Sensible sampling with interacting boxes. Suppose sampling over the block projected over the bed (a), along the direction denoted by the arrow (b) for  $N$  leapfrog steps. For a few steps, the bed tries to expand, and the containment constraints force the block on the left to shrink (b). Then, the bed tries to shrink (c), but we see that the *left block* does not return to its initial position. The behavior in the *second row* is more desirable, where the *left block* “grows” back to its initial location as the bed shrinks (e), but does not grow beyond that point (f). In the *third row*, we show the desired behavior of the sampler when the room box is trying to expand along the direction of the *arrow*. As the room shrinks, so does the bed (g), which completely disappears when it is fully outside the room (h). When the box starts to expand again, the bed grows back to its initial size (i), but no further (j)

rameters. For example, sampling over the focal length should drive the model to the current perspective distortion, which provides a better alignment of the projected model edges and the image edges. However, changing the focal length also modifies the size of the projection of the 3D scene, and these two forces are conflicting.

Hence, we jointly sample over camera and room box, as this allows to account for some of the correlations between camera and scene parameters. For this move, we use Hamiltonian dynamics for 20 iterations. At each step, we enforce the same constraints as in the case where we sample over the room box only.

#### 10.3.1.4 Sensible Dynamics with Multiple Boxes

The constraints in the model give rise to an additional problem illustrated in Fig. 10.8. Suppose sampling over the block projected over the bed Fig. 10.8(a), along the direction denoted by the arrow Fig. 10.8(b) for  $N$  leapfrog steps. For a few steps, the bed tries to expand, and the containment constraints force the block

on the left to shrink Fig. 10.8(b). Then, the bed tries to shrink Fig. 10.8(c), but we see that the left block does not return to its initial position. The behavior in the second row is more desirable, where the left block “grows” back to its initial location as the bed shrinks Fig. 10.8(e), but does not grow beyond that point Fig. 10.8(f). In the third row, we show a similar situation, where we show the desired behavior of the sampler when the room box is trying to expand. As the room shrinks, so does the bed Fig. 10.8(g), which completely disappears when it is fully outside the room Fig. 10.8(h). When the box starts to expand again, the bed grows back to its initial size Fig. 10.8(i), but no further Fig. 10.8(j). The procedure implementing this strategy for the sampling of a set of continuous parameters is as follows ( $\theta_{\text{in}}$  is the initial sample, and  $\theta_{\text{out}}$  is the sample after  $N$  leapfrog steps).

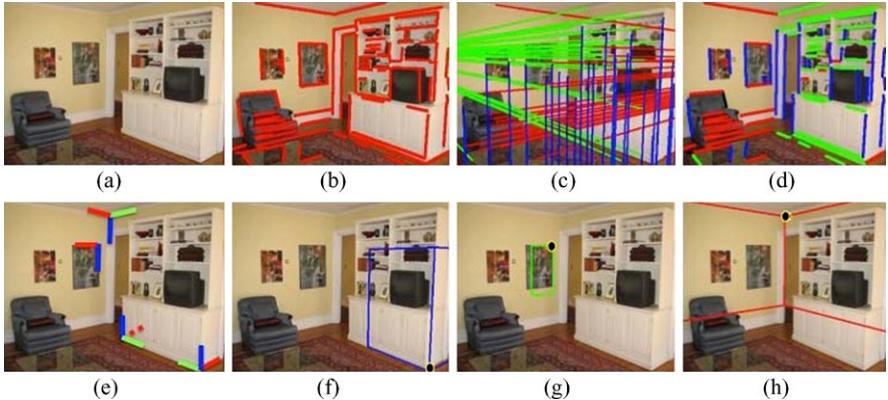
1. Set  $\theta_0 = \theta_{\text{in}}$
2. For every step  $i = 1, \dots, N$ 
  - a. Compute  $\theta_i$  from  $\theta_{i-1}$  by following the Hamiltonian dynamics
  - b. Whenever the posterior needs to be evaluated, compute it on a copy of  $\theta_{i-1}$  where we apply the containment constraints. Never apply the constraints on either  $\theta_i$  or  $\theta_{i-1}$
3. Set  $\theta_{\text{out}} = \theta_N$
4. Apply the containment constraints on  $\theta_{\text{out}}$

This procedure implements the desired behavior illustrated in Fig. 10.8, by keeping track of the initial positions and sizes of all objects in the model, and applying the containment constraints only when the posterior needs to be evaluated. Finally, these constraints are applied to the last sample, which will be the starting point for the next sampling move.

### 10.3.2 Jump Moves

One of the main challenges in the inference process is designing jump moves to efficiently add objects to the scene. Since the sampling space is so large, naive jump proposals, such as samples from a prior distribution, are unlikely to be accepted, and this leads to unacceptably long running times. To face this challenge, we introduced a data-driven strategy [23, 26] to condition the sampling on the data, by proposing samples from image evidence in a bottom-up fashion. Further, we exploit the Manhattan world assumption [1] that most surfaces in the world are aligned with one of three orthogonal directions, as this provides strong constraints on the parameter space.

In Fig. 10.9, we illustrate how our proposal mechanism combines the Manhattan world constraints and the advantages of data-driven inference. Here, we show that intersections of line segments on the image plane, which we call *image corners*, typically correspond to the projection of corners that are orthogonal in the 3D world. Using projective geometry and Manhattan orthogonality constraints, an image cor-



**Fig. 10.9** Detecting vanishing points (*top*) to find orthogonal corners (*bottom*). Most surfaces in indoor scenes are aligned with three principal orthogonal directions. This defines a triplet of orthogonal vanishing points in the image plane, which we find in a RANSAC fashion (**c**) from straight detected segments (**b**). Segments are assigned to one of three groups based on the vanishing point they converge to (**d**). Intersections of edges in different groups, which we call image corners, are typically generated by the projection of an orthogonal 3D corner (**e**). Given an estimate of the camera pose computed from the vanishing points, an image corner can be used to propose a furniture object (**f**), a frame (**g**), or the room box (**h**)

ner can be used to propose furniture objects (Fig. 10.9(f)), frames Fig. 10.9(g) and room box candidates Fig. 10.9(h), which are accepted with high probability.

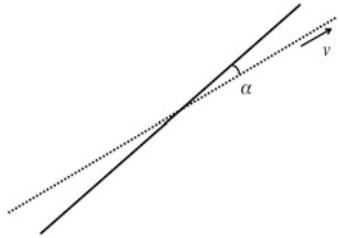
To achieve this goal, we first estimate the triplet of orthogonal vanishing points defining the Manhattan world directions [18]. This provides a good estimate of the camera focal length and pose, which are needed to propose 3D blocks from image corners. In what follows, we describe the procedure for estimating the vanishing points (Sect. 10.3.2.1), how to detect image corners (Sect. 10.3.2.2), and how to use them to propose both objects in the scene (Sect. 10.3.2.3) and room box candidates (Sect. 10.3.3).

### 10.3.2.1 Vanishing Point Estimation

We first detect straight edges and fit line segments to them using the straight connected edge detector by Hoiem et al. [12]. Then, we detect vanishing points following the RANSAC procedure proposed by Lee et al. [15]. At each step, we randomly select three pairs of line segments and position a vanishing point at the intersection of each pair. We then check the orthogonality of the vanishing points and reject triplets that are not orthogonal. We then estimate the intrinsic camera matrix  $K$  from the Choleski decomposition of the absolute conic matrix [8], which is fully determined by the position of the three vanishing points. We reject triplets that provide a non realistic focal length ( $f \notin [50, 2000]$ , measured in pixel).

For each valid triplet  $V_t(v_1, v_2, v_3)$ , we compute the objective function  $f(V_t)$  as follows. First, we compute the angular distance  $\alpha(s_i, v_k)$  between each line seg-

**Fig. 10.10** The angular distance  $\alpha$  between a segment  $s$  and its vanishing point  $v$ .  $\alpha$  measures the angle between  $s$  and the line through the mid point of  $s$  and  $v$



ment  $s_i$  and each vanishing point  $v_k$  (Fig. 10.10). A segment  $s_i$  is labeled as an outlier if  $\min_{k=1,2,3} \alpha(s_i, v_k) > 0.06$ . If  $n$  is the total number of outliers, we have

$$f(V_t) = \left( \sum_{i=1}^n \frac{\min_{k=1,2,3} \alpha(s_i, v_k)}{l_i} \right) / n \quad (10.19)$$

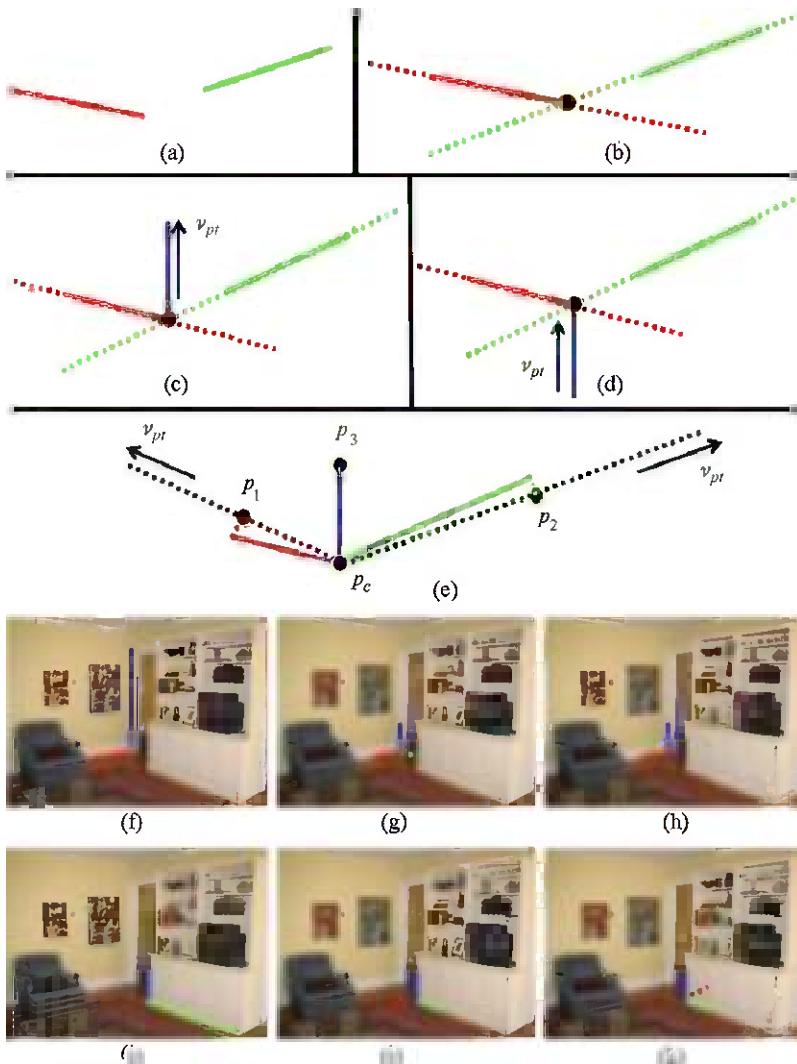
where  $n$  is the number of inlier line segments in the RANSAC procedure.  $l_i$  is the length of segment  $s_i$ , and is used to assign a larger weight to long segments.

We keep the valid triplet of vanishing points minimizing  $f(V_t)$  and such that the ratio  $r_{\text{out}}$  between the number of outliers and the total number of segments is less than 0.1. If no triplet satisfying these constraints is found, we increase this threshold by 0.1 and repeat until a triplet is found. This allows for considering first only triplets supported by a large number of segments, and, if none is available, we allow a larger number of outliers.

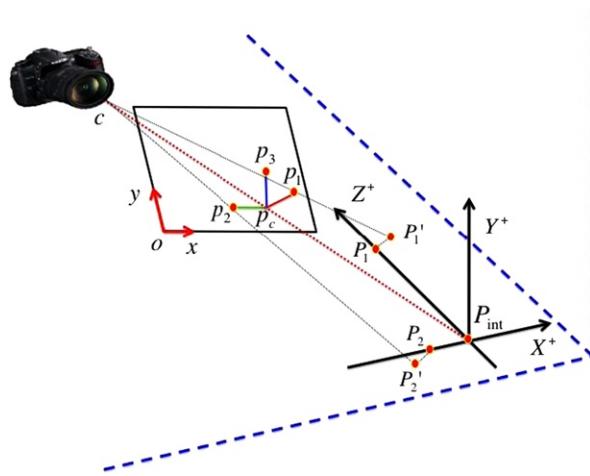
### 10.3.2.2 Image Corners

Given the estimated triplet of vanishing points, each line segment is assigned to the vanishing point minimizing the angular distance  $\alpha$  [9], thus partitioning them in three groups. Segments with  $\alpha > 0.12$  are not assigned to any group and considered as outliers. Notice that this threshold is less strict than the one used during RANSAC, as we want to have more candidates at this point. An example is shown in Fig. 10.9(d), where the three different groups are shown, respectively, in green, red and blue, and the outliers in black.

We assume that two image segments in different groups were generated by segments that are orthogonal in 3D. The intersection of two such segments, which we call image corner, is thus likely to be generated by a 3D orthogonal corner, such as the corners of the cabinet or the inner corner of the room shown in Fig. 10.9(e). Given an image corner and an estimate for the camera pose, we can propose 3D objects in likely positions. However, this requires knowing the three segments forming the image corner, but typically only two are visible due to occlusions, like the bottom left corner of the cabinet in Fig. 10.9(e). We address this problem by “hallucinating” the third segment when it is not visible. In the case of the bottom left cabinet corner, we consider the intersection between the two visible segments, which converge to two different vanishing point, and hypothesize that the third segment converges to the remaining vanishing point in the triplet. This is discussed in detail in Fig. 10.11,



**Fig. 10.11** Creating orthogonal corners from line segments. Given a pair of segments converging to different vanishing points (a), we first find their intersection (b). We do not create corners if the distance between the intersection and each segment is larger than 30 pixels. We position the third segment of the corner on the line through the corner position and the third vanishing point (e). Two corner configurations are possible (c)–(d). Last, we “rectify” the corner (e), to make sure it satisfies the orthogonality constraint imposed by the Manhattan world, which is needed to propose 3D objects from a 2D corner. In fact, corners are created from detected image segments, which do not necessarily satisfy these constraints due to errors in the edge detection. Specifically, instead of using the image segments, we consider their projections on the line through the corner center and the vanishing points (e), which are guaranteed to be orthogonal. The final corner is centered in  $p_c$ , and defined by segments  $p_1 p_c$ ,  $p_2 p_c$  and  $p_3 p_c$ . Notice that  $p_3 p_c$  was hallucinated by using a vanishing point, and does not need to be “rectified”. In the last two rows, we show examples of corners created from pairs of segments



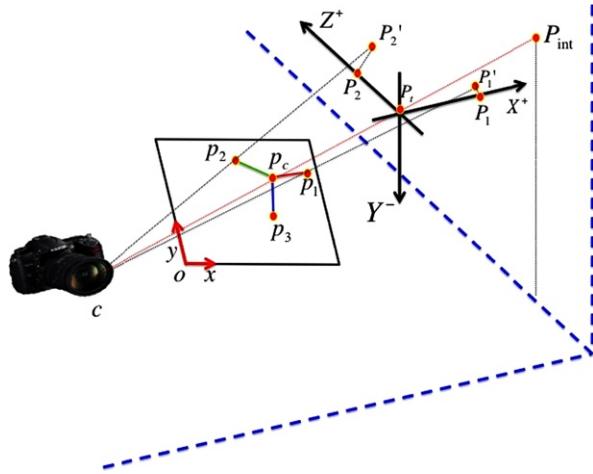
**Fig. 10.12** Finding the 3D corner of a furniture object from an image corner, conditioned on the current room box.  $p_c = (p_{cx}, p_{cy})$  denotes the position of the corner on the image plane, and  $P_c = (p_{cx}, p_{cy}, -f)$  is the corner in the 3D camera reference frame, where  $f$  is the focal length. We first consider the segment  $p_3p_c$ : if  $p_{3y} > p_{cy}$  in the image reference frame, the image corner is pointing upwards, and we assume that it was generated by an object corner on the room floor, delimited by the dashed lines (for the complimentary case of a downwards corner, see Fig. 10.13). Then, we cast a ray through the camera center  $c$  and the image corner  $P_c$ , and find its intersection with the room floor  $P_{\text{int}}$ , which defines the 3D position of the object corner. Assuming objects are aligned with the walls, we define the coordinate system  $(X, Y, Z)$ , which is aligned with the room walls and centered in  $P_{\text{int}}$ , the  $XZ$  plane coinciding with the room floor. We know the 3D corner will expand along the positive  $Y$  axis, and we use the rays between the camera and  $p_1$  and  $p_2$  to determine the directions along the  $X$  and  $Z$  axis. We first find the intersection  $P'_1$  between ray  $p_1c$  and the floor, which, due to small errors in the camera estimate, does not lie exactly on the  $Z$  or the  $X$  axis. We then compute  $P_1$  as the closest point to  $P'_1$  on either the  $X$  or the  $Z$  axis, which determines that the 3D corner in the picture expands along the positive  $Z$  axis. We repeat the same for  $P_2$  by using  $p_2$ , and in this example the 3D corner expands along the negative  $X$  axis

where we also explain how to use the Manhattan constraints to ensure the orthogonality of the corner. Last, we stress that, since the main purpose of corners is to propose the correct position and orientation of objects in 3D, and not their size, the length of corner segments in 2D does not provide any valuable information, as long as their direction is correct.

### 10.3.2.3 Adding Objects in the Scene Using Image Corners

As discussed above, we create image corners by considering the intersection of each pair of segments converging to different vanishing points. A corner is then used to add an object to the scene in a bottom-up data-driven fashion, in order to increase the acceptance probability in the MH acceptance formula [5].

First, the image corner is used to estimate the position of the furniture object in 3D (Figs. 10.12 and 10.13). The proposal is conditioned on the current estimate of both camera pose and room box. Second, we randomly select the type of the object,



**Fig. 10.13** Finding the 3D corner of a furniture object from a downwards image corner. We assume that a downwards image corner is generated by a corner on the *top* of a 3D object (For the complimentary case of an upwards corner, see Fig. 10.12). We cast a ray through the camera center and the corner position in the image  $P_c$ , and find the intersection  $P_{\text{int}}$  between this ray and the closest room wall. The 3D position of the corner can lie anywhere on the line  $((1 - t)c + tP_{\text{int}})$  between  $P_{\text{int}}$  and the camera center  $c$ . Any  $t \in [0, 1]$  defines a valid 3D position  $P_t$  for the corner, and we choose  $t$  by randomly choosing from the interval  $[0.4, 1.0]$ . We set the lower bound to 0.4, since values of  $t$  too close to 0 result in positioning the corner too close to the camera, which is a non realistic configuration. We assume that the object is aligned with the room walls and floor, and knowing that the 3D corner is pointing downwards, only the directions along the  $X$  and  $Z$  axis are left to be determined. Similarly to Fig. 10.12, we find the intersection  $P'_1$  between ray  $p_1c$  and a plane parallel to the room floor and passing through  $P_t$ , and  $P_1$  as the closest point to  $P'_1$  on either the  $X$  or the  $Z$  axis.  $P_1$  and  $P_2$  determine the corner directions in 3D along  $X$  and  $Z$ , which in this example are the positive  $X$  axis and the positive  $Z$  axis

such as bed, and use the priors for that category to propose the size of the object. Third, to further increase the acceptance ratio of jump moves, we use a delayed acceptance mechanism. In fact, corners help propose objects at the right position, but the size sampled from the prior is often inaccurate. Hence, we briefly sample over the continuous parameters of the newly added object before consulting the MH acceptance formula to decide whether to accept or reject the sample.

The full proposal procedure to add a furniture object in the scene is as follows.

1. Randomly choose an image corner, and determine whether it is pointing up or down (Fig. 10.12).
2. Determine the object category  $\tau$  by randomly choosing from the four available classes (bed, cabinet, couch, table), where each class has the same probability.
3. If the corner is pointing up, find the position of the 3D corner on the floor as explained in Fig. 10.12, otherwise find the position of the 3D corner as explained in Fig. 10.13.
4. Sample  $\mathcal{N}(r_{i3}; \mu_{\tau3}, \sigma_{\tau3})$  to propose the ratio  $r_{i3}$  between the room height and that of the proposed object height  $h_i$ . Set  $h_i = \frac{h_r}{r_{i3}}$ , where  $h_r$  is the current height of the room box.

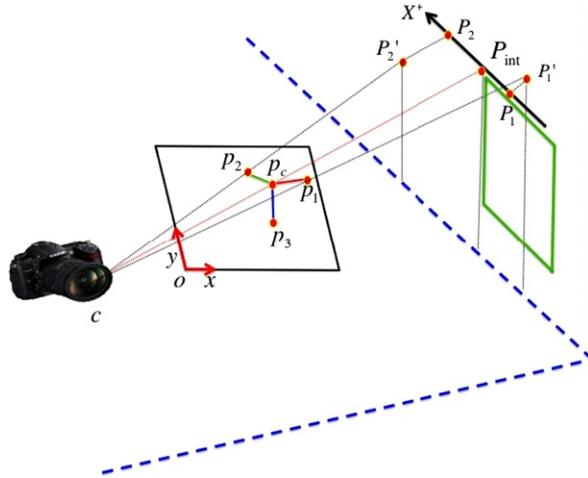
5. Sample  $u$  from uniform distribution  $\mathcal{U}(0, 1)$ . If  $u > 0.5$ , set the width  $w_i$  of the object to be larger than its length  $l_i$ , if  $u \leq 0.5$  set  $l_i$  to be larger. The next two steps are defined for the case  $u > 0.5$ , and can be adapted to the opposite case by swapping  $w_i$  with  $l_i$ .
6. Sample from  $\mathcal{N}(r_{i1}; \mu_{\tau_1}, \sigma_{\tau_1})$  to propose the ratio  $r_{i1}$  between  $h_i$  and its largest dimension  $w_i$ . Set  $w_i = \frac{h_i}{r_{i1}}$  using  $h_i$  from the previous step.
7. Sample from  $\mathcal{N}(r_{i2}; \mu_{\tau_2}, \sigma_{\tau_2})$  to propose the ratio  $r_{i2}$  between the object largest and shortest dimensions ( $w_i$  and  $l_i$ ). Set  $l_i = \frac{w_i}{r_{i2}}$  using  $w_i$  from the previous step.
8. Shrink any furniture object in the scene colliding with the proposed one.
9. In case the object does not fit in the room, expand the room box. This might involve lowering the position of the floor, raising the ceiling, or increasing the room width and/or length.
10. Briefly sample over the object continuous parameters using Diffusion move 3 (delayed acceptance).
11. Accept or reject the proposed object by consulting the MH acceptance formula.

Similarly, the procedure to add a frame in the scene is summarized below. Notice that the move adding an object chooses whether to add a furniture object or a frame with 0.5 probability.

1. Randomly choose an image corner.
2. Determine the object category  $\tau$  by randomly choosing from the three available classes (door, picture frame, window), where each class has the same probability.
3. Find the position of the 3D corner, and the room wall  $s_i$  it is anchored to as explained in Fig. 10.14.
4. Sample  $\mathcal{N}(r_{i3}; \mu_{\tau_3}, \sigma_{\tau_3})$  to propose the ratio  $r_{i3}$  between the room height and that of the proposed frame height  $h_i$ . Set  $h_i = \frac{h_r}{r_{i3}}$ , where  $h_r$  is the current height of the room box.
5. Conditioned on  $s_i$ , the frame has either negligible width or negligible length. In the latter case, sample from  $\mathcal{N}(r_{i1}; \mu_{\tau_1}, \sigma_{\tau_1})$  to propose the ratio  $r_{i1}$  between  $h_i$  and width  $w_i$ . Set  $w_i = \frac{h_i}{r_{i1}}$  using  $h_i$  from the previous step. When  $w_i$  is negligible, set  $l_i = \frac{h_i}{r_{i1}}$ .
6. Shrink any frame in the scene colliding with the proposed one.
7. If the frame does not fit on the wall, expand it.
8. Briefly sample over the object continuous parameters using Diffusion move 3 (delayed acceptance).
9. Accept or reject the proposed frame by consulting the MH acceptance formula.

#### 10.3.2.4 Other Jump Moves

The remaining two jump moves respectively remove an object from the room, or change the type of one of the objects. The former simply deletes a randomly selected object, and the proposed change is accepted or rejected using the MH acceptance formula. Instead, changing the type of an object involves only changes in the prior.

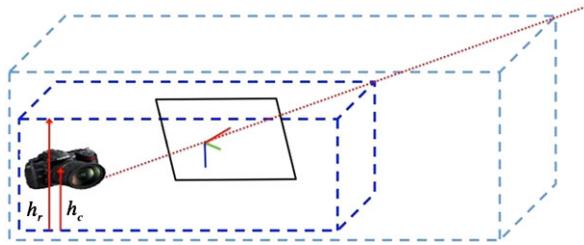


**Fig. 10.14** Finding the 3D corner of a frame from an image corner. We cast a ray between the camera center  $c$  and  $P_c$ , and find the intersection  $P_{\text{int}}$  with the closest wall, which determines the position of the corner in 3D. We know that the frame will expand downwards on the wall, as the corner on the image plane is pointing down, but we have to determine the 3D corner direction along the  $X$  axis. We do so by considering  $P'_1$  and  $P'_2$ , which are obtained by intersecting rays  $p_1c$  and  $p_2c$  with the wall the frame is anchored to. Due to errors in the camera estimate, we cannot expect  $P'_1$  and  $P'_2$  to lie exactly on  $X$ , and we then consider their projections on  $X$ ,  $P_1$  and  $P_2$ .  $P_1$  defines the direction if the distance between  $P_1$  and  $P'_1$  is smaller than the distance between  $P_2$  and  $P'_2$ , and use  $P_2$  otherwise. This is equivalent to choosing the direction that best satisfies the orthogonality constraints. In this example, we used  $P_1$  to set the direction of the frame drawn on the wall

First, an object is randomly selected, and we propose changing its current type  $\tau$  to a different category, also randomly selected. For example, changing the label of an object from “bed” to “couch” results in using the distribution on size and position for couches when evaluating the object prior probability, while before the prior for beds was used. Again, the proposed change is evaluated using the MH acceptance formula. Last, we do not propose changing furniture objects into frames or vice versa. This means that, for example, a couch can only be changed into a table, a bed or a cabinet, while a door only into a picture frame or a window.

### 10.3.3 Initializing the Room Box and the Camera Parameters

We initialize the parameters of the camera and of the room box by proposing candidates from the orthogonal corners detected on the image plane. Each corner is used to generate a candidate, and we use the  $N$  room box candidates with the highest posterior to initialize the  $N$  threads used for inference. While we do not solely commit to these proposals, since the inference process will modify room box and camera parameters, we found that a good initial estimate of the room box parameters makes



**Fig. 10.15** Finding the position of a 3D room corner from an image corner. We first cast a ray between the corner in the image and the camera center. We then position the 3D room corner along this line. Different positions generate different room boxes, as illustrated by the *two dashed examples* above. Once the 3D position is chosen, we set the room height  $h_r$  such that the ratio between the height of the camera from the floor  $h_c$  and  $h_r$  falls in the range of plausible values (see text). Further, we have to enforce that the room box is big enough to contain the camera

the inference more efficient. We now discuss how a corner is used to generate a candidate.

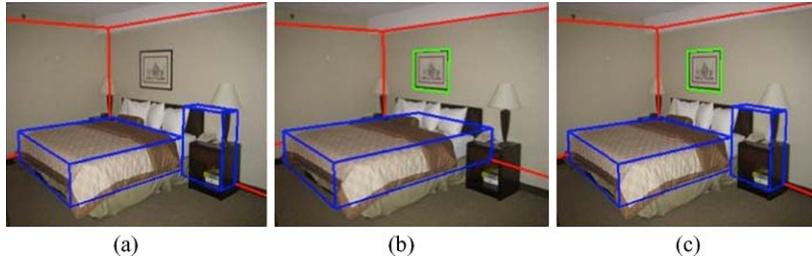
Shi et al. [21] showed how to estimate the camera pose from the projection of an orthogonal corner and the known focal length. We follow their procedure to estimate the pitch  $\phi$ , the roll  $\psi$  of the camera and the yaw  $\gamma_r$  of the room, which in our framework define the camera pose. Notice that the focal length is available from the estimated triplet of orthogonal vanishing points.

The method by Shi et al. [21] also recovers the 3D directions of the lines forming the corner. However, we still need to determine the 3D position of the corner, which can lie anywhere on the line defined by the camera center and the corner position on the image plane, as illustrated in Fig. 10.15. Since we cannot determine absolute positions and sizes from a single image, we arbitrarily position the corner on the line such that the distance between the corner and the camera center is 10 units.

This still leaves the room dimensions  $(h_r, w_r, l_r)$  as free parameters. Since absolute sizes cannot be used when reconstructing from a single image, we set the room height such that the ratio between the height of the camera from the floor  $h_c$  and  $h_r$  takes plausible values (Fig. 10.15). Specifically, we sample uniformly from the interval  $[\mu_{ch} - 2\sigma_{ch}, \mu_{ch} + 2\sigma_{ch}]$ , with step  $\frac{\sigma_{ch}}{5.0}$ . For each different height, we set  $w_r$  and  $l_r$  such that  $\frac{w_r}{h_r} = \mu_{r2}$  and  $\frac{l_r}{h_r} = \mu_{r2}$ , and if the room box is not big enough to contain the camera, we expand it. We then briefly sample over room box and camera parameters by alternating Diffusion moves 1 and 2, and keep the sample with the highest posterior.

#### 10.3.4 Exchanging Information Among Threads

At the end of the inference process, each thread outputs the sample with the highest posterior. In most cases, we found that some objects are found only by some of the threads, as illustrated in Fig. 10.16. One thread did not find the picture Fig. 10.16(a), while the other did not find the nightstand Fig. 10.16(b). While a longer running



**Fig. 10.16** The result (c) of exchanging objects between the samples found by two different threads (a and b)

time could potentially allow each thread to find all objects, we propose instead to let threads exchange objects at the end of the inference Fig. 10.16(c).

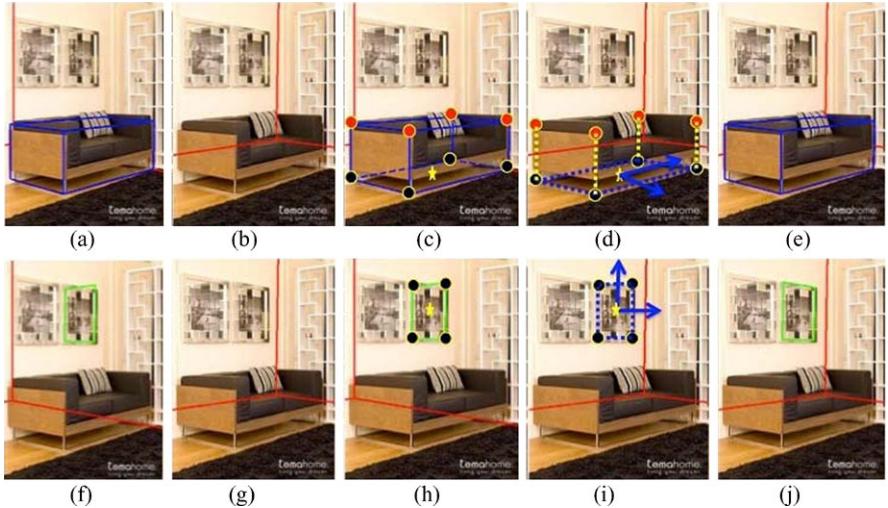
Since object position and size is defined relatively to the room, we need an exchanging mechanism taking into account that different threads have different estimates of the room box. Further, the camera parameters found by each thread are potentially different. Hence, we exchange an object between a source thread and a destination thread by enforcing that the projection of the object in the source matches as closely as possible the projection of the object in the destination (Fig. 10.17).

At the end of inference, we add to the best sample found by a thread all the objects found by the other threads, one at a time, and keep the one that provides the best posterior. We repeat this  $K = 10$  times, or until there is no improvement in the posterior. While less greedy methods are possible, this approach works well in practice. The complete procedure for exchanging objects among  $N$  threads is as follows:

1. For each thread  $i$ , save in  $\theta_i^0$  the best sample found by thread  $i$
2. For each  $\theta_i^0$ , and for  $k = 1, \dots, K$ 
  - a. Set  $\theta_i^k = \theta_i^{k-1}$ . Compute the posterior  $p_i^k$  of  $\theta_i^k$ . Get all the objects found by other threads  $O = \cup_{j=1}^N O_j$  with  $j \neq i$ , where  $O_j = (o_{j1}, \dots, o_{jn})$  is the list of objects in  $\theta_j^0$ .
  - b. For each object  $o_w$  in  $O$ , create  $\theta_i^{k-w}$  by adding  $o_w$  to  $\theta_i^k$ , and compute the posterior  $p_i^{k-w}$ . Set  $\theta_{i\_max}^k = \theta_i^{k-w}$  such that  $p_{i\_max}^{k-w} = \max_{w'} p_i^{k-w'}$ .
  - c. If the posterior of  $\theta_{i\_max}^k$  is larger than  $p_i^k$ , set  $\theta_i^k = \theta_{i\_max}^k$ . Otherwise, stop and return  $\theta_i^K = \theta_i^k$ .
3. Return  $\theta_i^K$  with the largest posterior.

## 10.4 Results

We start by evaluating the performances of the various components of our algorithm in terms of the error on the room layout estimation [9, 16, 24], which is a standard



**Fig. 10.17** Exchanging furniture objects (*top row*) and frames (*bottom row*) between two samples. Consider adding the object in **(a)** to **(b)**: the camera parameters and the room boxes are different, and this is a problem since object coordinates are relative to the room reference frame. We address this by making sure that the projection of the transferred object **(e)** matches that of the original one **(a)**. First, we use the original camera to project the object corners that touch the floor, shown in *black* in **(c)**. The *star* indicates the projection of the center of the object “floor”. We then cast a ray between the destination camera and the *star*, and intersect it with the floor of the destination room. This gives us the object “floor” center in the coordinate system defined by the destination room box. The transferred object will be aligned with the walls of the destination room, denoted by the *arrows* in **(d)**. We determine the object width and length by intersecting the *arrows* with the *dashed lines* **(d)**, which might not be aligned with the destination walls due to the differing camera parameters. The object height is found by projecting the corners on the top of the object under the original camera **(c)**. For each corner  $c$ , we consider ray  $r_1$  through its projection and the destination camera, and ray  $r_2$  orthogonal to the room floor and through the corresponding corner of the object on the floor. The 3D position of  $c$  in the destination room is given by the point on  $r_2$  closest to  $r_1$ . The 3D length of the *vertical dashed lines* determines the height of the transferred object. This length might not be the same for all four lines, and we set the height of the object by averaging them. The final transferred object is shown in **(e)**. The *second row* illustrates the equivalent procedure for frames, where we consider the projection of the frame’s corners on the closest wall

measure in this field. This error compares the projection of the estimated room box against the ground truth, where each pixel was labeled according to the room face it belongs to (ceiling, floor, left, middle and right wall), by computing the ratio of misclassified pixels.

In Table 10.1, we evaluate the impact of the different components of our approach on the test portion of the Hedau dataset [9], consisting of 104 color images. In the left column we show the contribution of the priors in the scenario where the likelihood function uses edges and orientation maps, as in our 2012 CVPR paper [3]. We see that the room prior reduces the layout error, and so do priors on objects.

**Table 10.1** Analysis of the components of our approach based on room layout error, evaluated on the Hedau test set [9]

	Edges + OM		
	Edges	Edges + OM	Edges + OM + GC
No prior		20.4 %	
Room prior		19.7 %	
Room + obj prior		17.8 %	
No objects	24.1 %	21.3 %	21.8 %
Objects	20.7 %	17.8 %	17.2 %
Exchange objects	14.2 %	14.6 %	13.6 %

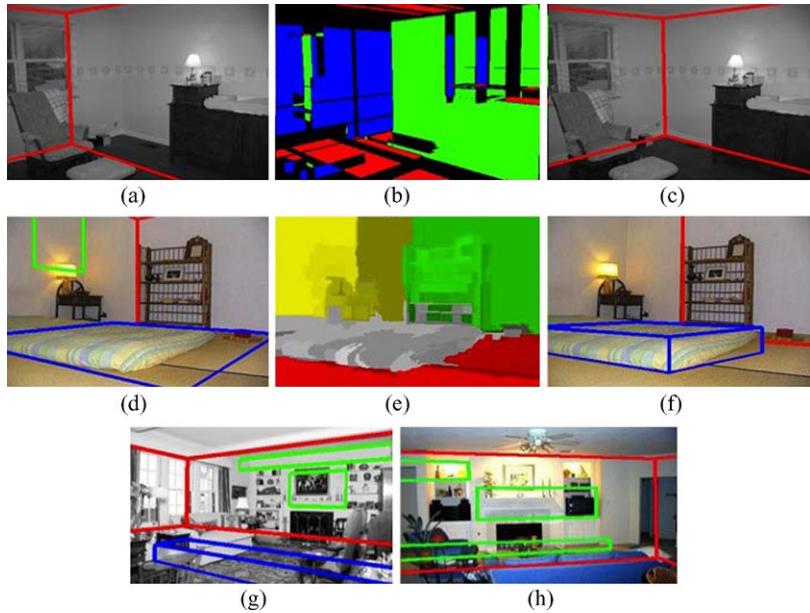
This suggests that adding realistic objects in the room drives the inference towards better spatial configurations.

We then evaluate the benefits of the different components of the likelihood, considering three cases: (1) we do not allow any object in the room, (2) we allow objects in the scene, and (3) we allow exchanging information among threads. Interestingly, exchanging objects allows choosing a better room box, since more objects provide more evidence on the correct room layout. There is no improvement when only edges are used, and we believe this happens because the likelihood is not robust enough in this case. Qualitative examples of these improvements are shown in Fig. 10.18.

Table 10.2 shows that our method is comparable to the state-of-the-art on the Hedau dataset, and also report results on the UCB dataset [25], consisting of 340 black and white images. When comparing with state-of-the-art, we consider our full algorithm, including all the likelihood components, and with the object exchange enabled.

Then, we evaluate on object recognition. We ground truthed the UCB [25] and Hedau dataset [9] by manually identifying the seven object classes we experimented with, not considering objects occupying less than 1 % of the image. To evaluate detection, we project the 3D object hypothesized by our model onto the image, and compare this with the ground truth object position. If the intersection of the two areas is more than 50 % of their union, we consider it a correct detection. We first measure how many objects we correctly identified for each of the two main categories (furniture and frames), even if there is confusion within the subcategories (e.g., when we label a table as a couch, or a window as a door). We provide precision and recall scores based on this criterion. Second, we measure the accuracy we achieved within each of the two categories, as the percentage of objects that were assigned to the correct subcategory.

We report in Table 10.3 that using object priors greatly improves precision and recall, for both furniture and frames. When we do not use priors, objects are not labeled (e.g., we do not know whether a furniture object is a couch or a bed), and subcategory accuracy cannot be evaluated. Table 10.4 shows that geometric context improves all recognition scores, except for frame subcategory classification on the



**Fig. 10.18** Effects of the individual components of our model. Using only edges we get a poor estimate of the room box, due to the edge detector missing the wall edges (a). By adding orientation maps to the likelihood (b), where we draw in green and blue the pixels assigned to the two orthogonal vertical surfaces and in red the horizontal pixels, we obtain a better room box (c). Also adding geometric context helps. In (d), the algorithm is confused by several mistakes made in estimating the orientation maps, but geometric context (e) helps the reconstruction process (f). In (e), we used red for pixels labeled as floor, yellow for left wall, green for middle wall, and gray for objects. Last, using priors on object 3D position and size avoids proposing objects with unrealistic size, which would otherwise often “latch” to image features (g)–(h)

**Table 10.2** Comparison with state-of-the-art on room layout error

	Dataset	Hedau [9]	Lee [16]	Schwing [20]	Our full approach
Hedau [9]	21.2 %	16.2 %	12.8 %	13.6 %	
UCB [25]	NA	NA	NA	14.2 %	

Hedau dataset. In general, performances are more modest on the UCB dataset, and this includes also the room layout error. This black and white dataset is in fact more challenging, as several images are extremely blurry. We also notice that in general geometric context only introduces small improvements in the subcategory classification, if we exclude furniture on the UCB dataset. This is because adding a new feature allows to obtain more accurate object fits, but the “burden” of classification is still entirely on the prior on size and position. We posit that having different geometric models for object classes, such as tables with legs or couches with backrests, as well as class-dependent appearance models, would improve this score.

**Table 10.3** Benefits of object priors evaluated on UCB and Hedau datasets. P, R, and S denote Precision, Recall and Subcategory Accuracy

	UCB [25]			Hedau [9]		
	P	R	S P	R	S	
Furn no prior	19.4 %	10.4 %	NA	27.1 %	9.2 %	NA
Furn prior	31.0 %	20.1 %	38.0 %	32.5 %	20.3 %	50.0 %
Frames no prior	21.8 %	14.0 %	NA	23.1 %	19.5 %	61.2 %
Frames prior	27.2 %	19.7 %	60.0 %	36.1 %	27.5 %	62.6 %

**Table 10.4** Benefits of geometric context evaluated on UCB and Hedau datasets

	UCB [25]			Hedau [9]		
	P	R	S	P	R	S
Furn no gc	31.0 %	20.1 %	38.0 %	32.5 %	20.3 %	50.0 %
Furn gc	33.7 %	27.7 %	50.1 %	50.0 %	24.2 %	51.6 %
Frames no gc	27.2 %	19.7 %	60.0 %	36.1 %	27.5 %	62.6 %
Frames gc	28.2 %	24.3 %	60.8 %	38.4 %	28.3 %	61.2 %

**Table 10.5** Effects of exchanging objects among threads on UCB and Hedau datasets

	UCB [25]			Hedau [9]		
	P	R	S	P	R	S
Furn no swap	33.7 %	27.7 %	50.1 %	50.0 %	24.2 %	51.6 %
Furn swap	33.0 %	29.7 %	50.0 %	53.5 %	28.5 %	51.3 %
Frames no swap	28.2 %	24.3 %	60.8 %	38.4 %	28.3 %	61.2 %
Frames swap	28.4 %	37.3 %	59.8 %	37.5 %	35.5 %	66.0 %

The effects of exchanging objects among threads are available in Table 10.5. There is a trend showing that exchanging objects achieves better recall at similar or slightly lower levels of precision. This is because adding objects from other threads allows to find many more objects, at the cost of a few additional mistakes. Variations in subcategory classification are mostly negligible. Last, we report confusion matrices for both furniture and frames on the Hedau dataset (Table 10.6). We here consider our full approach, including edges, orientation maps, geometric context, priors and object exchanging. We see that we recognize more objects in categories that are better approximated by a single block, such as beds and cabinets. Performances decrease for concave objects like table, that are not well approximated by a convex box. Also, there is more confusion between object categories similar in size, such as couches and tables, or windows and picture frames.

Qualitative results are available in Fig. 10.19, where we show the scene reconstructions provided by the full algorithm. The most typical failures are shown in Fig. 10.20.



**Fig. 10.19** Scene reconstructions provided by our full approach. We show the room box in *red*, furniture in *blue* and frames in *green*

**Table 10.6** Confusion matrices on Hedau test set [9]

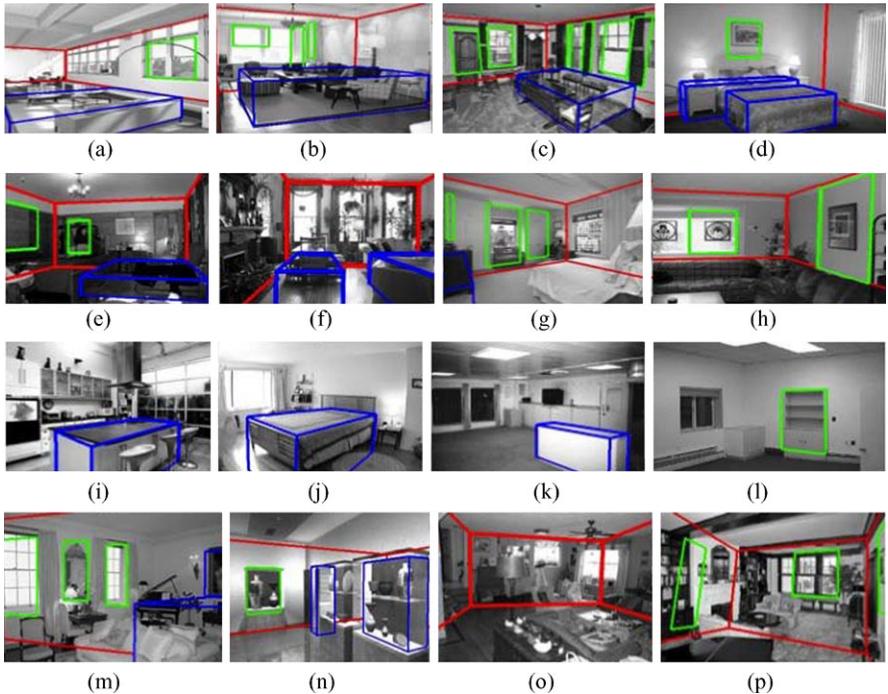
	Bed	Cabinet	Couch	Table
Bed	14	1	7	1
Cabinet	1	12	2	2
Couch	6	1	7	5
Table	4	3	4	6

	Door	Picture	Window
Door	7	0	7
Picture	0	29	13
Window	8	5	28

## 10.5 Conclusions

Our top-down Bayesian approach for understanding indoor scenes is competitive with state-of-the-art approaches on the task of recovering the room box. Interestingly, priors on 3D geometry both improved object recognition scores, and provided better room box estimates. We believe this is an interesting finding towards providing top-down scene interpretations that are globally consistent in terms of geometry



**Fig. 10.20** Some typical failures. Due to clutter we often hallucinate objects (a)–(d). For example, in (b) we propose a bed whose edges “latch” to those of the carpet, and to the top of the armchairs. However, in this case our approach still provides a reasonable approximation of the space occupancy of the scene. In other situations (e)–(f), we hallucinate boxes in the gap among objects, thus providing a wrong estimate of the occupied space, or completely miss objects (g)–(h). This mostly happens when faint edges are missed by the edge detector, such as those of the bed in (g). Further, we often confuse object categories similar in size and position. For example, the table in (i) is wrongly labeled as a bed, and the opposite happens in (j). The wall in (k) is labeled as a cabinet, and we sometimes confuse an object that is directly facing the camera for a frame (l). We also show some cases where the algorithm estimated the wrong room box (m)–(n). In (m), this is caused by the edge detector missing the edge between the room walls and the ceiling completely. Other more catastrophic failures are due to bad initial estimates of the camera, from which the algorithm could not recover (o)–(p)

and semantics. Another strength of this method is that it does not commit to partial configurations, and can recover from initial errors, one example being the improvements on the initial estimate of the room box.

We posit that our method will prove more powerful as it integrates more sophisticated object models, including non convex-approximations such as tables with legs. In fact, more detailed geometry should help distinguish between classes that are similar in position and size. Also, recent work by Hedau et al. [11] showed that scene interpretation is enhanced by more accurate geometric models, such as blocks with backrests. More sophisticated models introduce additional complex constraints on the structure of objects (for example, chairs with four symmetric legs), which our

Bayesian framework could accommodate relatively easily. Further, the Bayesian formulation also allows integrating additional image evidence that would make the model more robust. For example, visual inspection of the results suggested that adding category dependent appearance models is a promising direction of research, as well as enforcing that projections of objects are uniform in terms of color and texture.

**Acknowledgements** This material is based upon work supported by the National Science Foundation under Grant No. 0747511. We thank Joseph Schlecht for his contributions and suggestions in designing the code base. We also acknowledge the valuable help of Joshua Bowdish, Ernesto Brau, Andrew Emmott, Daniel Fried, Jinyan Guan, Emily Hartley, Bonnie Kermgard, and Philip Lee.

## References

1. Coughlan JM, Yuille AL (1999) Manhattan world: compass direction from a single image by Bayesian inference. In: ICCV
2. Del Pero L, Guan J, Brau E, Schlecht J, Barnard K (2011) Sampling bedrooms. In: CVPR
3. Del Pero L, Bowdish J, Fried D, Kermgard B, Hartley E, Barnard K (2012) Bayesian geometric modeling of indoor scenes. In: CVPR
4. Delage E, Lee HL, Ng AY (2005) Automatic single-image 3d reconstructions of indoor Manhattan world scenes. In: ISRR
5. Green PJ (1995) Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika* 82(4):711–732
6. Green PJ (2003) Trans-dimensional Markov chain Monte Carlo. In: Highly structured stochastic systems
7. Gupta A, Satkin S, Efros AA, Hebert M (2011) From 3D scene geometry to human workspace. In: CVPR
8. Hartley RI, Zisserman A (2004) Multiple view geometry in computer vision. Cambridge University Press, Cambridge
9. Hedau V, Hoiem D, Forsyth D (2009) Recovering the spatial layout of cluttered rooms. In: ICCV
10. Hedau V, Hoiem D, Forsyth D (2010) Thinking inside the box: using appearance models and context based on room geometry. In: ECCV
11. Hedau V, Hoiem D, Forsyth D (2012) Recovering free space of indoor scenes from a single image. In: CVPR
12. Hoiem D, Efros AA, Hebert M (2005) Geometric context from a single image. In: ICCV
13. Hoiem D, Efros AA, Hebert M (2006) Putting objects in perspective. In: CVPR
14. Karsch K, Hedau V, Forsyth D, Hoiem D (2011) Rendering synthetic objects into legacy photographs. In: SIGGRAPH Asia
15. Lee DC, Hebert M, Kanade T (2009) Geometric reasoning for single image structure recovery. In: CVPR
16. Lee DC, Gupta A, Hebert M, Kanade T (2010) Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In: NIPS
17. Neal RM (1993) Probabilistic inference using Markov chain Monte Carlo methods. Technical report
18. Rother C (2002) A new approach to vanishing point detection in architectural. *Image Vis Comput* 20(9–10):647–655
19. Schlecht J, Barnard K (2009) Learning models of object structure. In: NIPS
20. Schwing A, Hazan T, Pollefeys M, Urtasun R (2012) Efficient structure prediction with latent variables for general graphics models. In: CVPR

21. Shi F, Zhang X, Liu Y (2004) A new method of camera pose estimation using 2D-3D corner correspondence. *Pattern Recognit Lett* 25(10):1155–1163
22. Tsai G, Xu C, Liu J, Kuipers B (2011) Real-time indoor scene understanding using Bayesian filtering with motion cues. In: *ICCV*
23. Tu Z, Zhu S (2002) Image segmentation by data-driven Markov chain Monte-Carlo. In: *PAMI*
24. Wang H, Gould S, Koller D (2010) Discriminative learning with latent variables for cluttered indoor scene understanding. In: *ECCV*
25. Yu SX, Zhang H, Malik J (2008) Inferring spatial layout from a single image via depth-ordered grouping. In: *POCV*
26. Zhu S-C, Zhang R, Tu Z (2000) Integrating top–down/bottom–up for object recognition by data driven Markov chain Monte Carlo. In: *CVPR*

# Chapter 11

## Efficient Loopy Belief Propagation Using the Four Color Theorem

Radu Timofte and Luc Van Gool

**Abstract** Recent work on early vision such as image segmentation, image denoising, stereo matching, and optical flow uses Markov Random Fields. Although this formulation yields an NP-hard energy minimization problem, good heuristics have been developed based on graph cuts and belief propagation. Nevertheless both approaches still require tens of seconds to solve stereo problems on recent PCs. Such running times are impractical for optical flow and many image segmentation and denoising problems and we review recent techniques for speeding them up. Moreover, we show how to reduce the computational complexity of belief propagation by applying the Four Color Theorem to limit the maximum number of labels in the underlying image segmentation to at most four. We show that this provides substantial speed improvements for large inputs, and this for a variety of vision problems, while maintaining competitive result quality.

### 11.1 Introduction

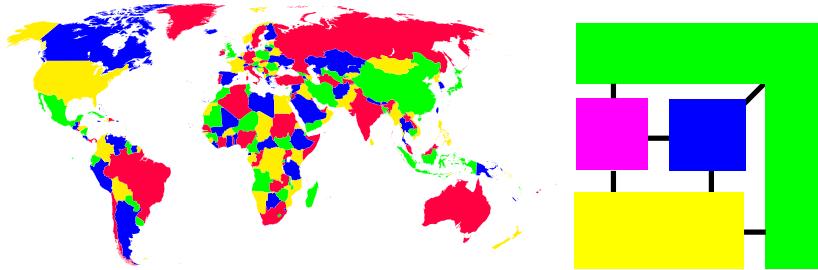
Much recent work on early vision algorithms—such as image segmentation, image denoising, stereo matching, and optical flow—models these problems using Markov Random Fields (MRF). Although this formulation yields an NP-hard energy minimization problem, good heuristics have been developed based on graph cuts [3] and belief propagation [21, 29]. A comparison between the two different approaches for the case of stereo matching is described in [23]. Both approaches still require tens of seconds to solve stereo problems on recent PCs. Such running times are impractical for many stereo applications, but also for optical flow and many image segmentation

---

R. Timofte (✉) · L. Van Gool  
VISICS, ESAT-PSI/iMinds, KU Leuven, Leuven, Belgium  
e-mail: [Radu.Timofte@esat.kuleuven.be](mailto:Radu.Timofte@esat.kuleuven.be)

L. Van Gool  
e-mail: [Luc.VanGool@esat.kuleuven.be](mailto:Luc.VanGool@esat.kuleuven.be)

L. Van Gool  
Computer Vision Lab, D-ITET, ETH Zurich, Zurich, Switzerland  
e-mail: [vangoool@vision.ee.ethz.ch](mailto:vangoool@vision.ee.ethz.ch)



**Fig. 11.1** Planar graphs: political world map colored with 4 colors illustrating the FCT (*left*, only the land boundaries are considered) and an example with 4 connected regions showing that less than 4 colors would indeed not suffice (*right*)

and denoising problems. Alternative, faster methods are available but generally give inferior results.

In the case of Belief Propagation (BP), a key reason for its slow performance is that the algorithm complexity is proportional to both the number of pixels in the image, and the number of *labels* in the underlying image segmentation, which is typically high. If we could limit the number of labels, its speed performance should improve greatly. Our key observation is that by modifying the propagation algorithms we can use a low number of placeholder labels, that we can reuse for non-adjacent segments. These placeholder labels can then be replaced by the full set of actual labels. Since image segments form a planar graph, they therefore require at most four placeholder labels by virtue of the Four Color Theorem (FCT) [17] to still have different colors for all adjacent segments. A joint optimization process provides a fast segmentation through the placeholder labels and a fine-grained labeling through the actual labels. The computational time is basically dependent on the number of placeholder rather than actual labels. This chapter is an extended version of our previous published work [24]. For the sake of self-consistency, the FCT is explained next.

The FCT states that for any 2D map there is a four-color covering such that contiguous regions sharing a common boundary (with more than a single point) do not have the same color. F. Guthrie first conjectured the theorem in 1852 (*Guthrie's problem*). The consequence of this theorem is that when an image, seen as a planar graph, is segmented into contiguous regions, there are only four colors to be assigned to each pixel/node for all segments to be surrounded only by segments of different colors (see Fig. 11.1). Once such 4-color scheme is adopted, for each pixel/node there is only one of four decisions that can be taken.

This work exploits the FCT result to substantially improve the running time of BP, thus providing fast alternatives to local methods for early vision problems. Our approach assigns one of 4 colors, that is, one of 4 placeholder labels, to each pixel, in order to arrive at a stable segmentation of the image. At the same time it assigns a more fine-grained label, like the intensities, disparities, or displacements to each of the 4 possible colors. The resulting fine-grained labeling—the actual outcome of the algorithm—changes continuously within the segments and abruptly across their

boundaries. In doing so, our approach provides a fast approximation to optimal MRF labeling. Henceforth, we will systematically refer to placeholder labels as *colors*, and to actual, fine-grained labels as *labels*.

In the case of image segmentation, we obtain results that are qualitatively comparable to traditional log-linear Minimum Spanning Tree (MST)-based methods [6], but with computation times only linear in the number of pixels. Also for image denoising, stereo matching, and optical flow, we obtain computation times linear in the number of pixels but independent of the number of labels (resp. intensities, disparities, and displacements). The results are as accurate as for the (slower) multi-scale loopy belief propagation proposed in [7].

The remainder of the chapter is structured as follows. Section 11.2 surveys the (recent) MAP inference literature focusing on Loopy Belief Propagation for MRF labeling and provides the links between the graph coloring theory and our settings. Section 11.3 goes into more details about the implementation of Loopy Belief Propagation (LBP). In Sect. 11.4, the Four Color Theorem-based techniques are incorporated in both the LBP framework and a fast forward BP approximation. Section 11.5 describes the experiments that were conducted, and how the proposed techniques can be used for tasks like image segmentation, image denoising, stereo matching, and optical flow. The conclusions are drawn in Sect. 11.6.

## 11.2 Related Literature

So far, the FCT has been used only sporadically in computer vision. To the best of our knowledge, Vese and Chan [26] were the first to use the FCT in computer vision for their multiphase level set framework, in the piecewise smooth case. Agarwal and Belongie [1] showed that the two bit upper bound for per-pixel label storage in color-coded image partitions as a result of FCT, still is sub-optimal according to information theory. As said, this work for the first time introduces the use of the FCT in MAP inference for MRF labeling. In what follows, we review the state-of-the-art in MRF labeling and provide further background information on graph coloring.

### 11.2.1 MAP Inference—Discrete MRF

The general framework for the problems we consider here can be defined as follows (we use the notation and formulation from [7]). Let  $\mathcal{P}$  be the set of pixels  $p$  in an image and  $\mathcal{L}$  be a set of labels. The labels correspond to the quantities that we want to estimate at each pixel, such as disparities, intensities, or classes. A labeling  $f$  then assigns a label  $f_p \in \mathcal{L}$  to each pixel  $p \in \mathcal{P}$ . Typically, a labeling varies smoothly except for a limited number of places, where it changes discontinuously, that is, at segment edges. A labeling is evaluated through an energy function,

$$E(f) = \sum_{(p,q) \in \mathcal{N}} V(f_p, f_q) + \sum_{p \in \mathcal{P}} D_p(f_p) \quad (11.1)$$

where the  $(p, q)$  in  $\mathcal{N}$  are the edges in the four-connected image grid.  $V(f_p, f_q)$  is the pairwise cost or ‘discontinuity cost’ of assigning labels  $f_p$  and  $f_q$  to two neighboring pixels  $p$  and  $q$ .  $D_p(f_p)$  is the unary cost or ‘data cost’ of assigning label  $f_p$  to pixel  $p$ . Finding a labeling with minimum energy corresponds to the maximum a posteriori (MAP) estimation problem for an appropriately defined MRF.

The MAP inference for discrete MRFs is NP-hard except for tree-structured MRFs, and pairwise MRFs with submodular energies, where

$$V(a, a) + V(b, b) \leq V(a, b) + V(b, a) \quad (11.2)$$

for two neighboring pixels and the labels  $a, b \in \mathcal{L}$ . In practice, approximate optimization methods are proposed [22, 28] for minimizing the energy in (11.1). Iterated Conditional Modes (ICM) [2], Simulated Annealing [10] and Highest Confidence First (HCF) [4] are among the oldest such methods. The 1990s turned Loopy Belief Propagation (LBP) [15] and Graph Cuts [11] into mainstream methods. A lot of recent efforts are going into optimization methods such as quadratic pseudo-boolean optimization [18], linear programming primal-dual, or other dual methods [28].

In this work, we focus mainly on LBP methods, and more particular on the max-product approach [29], which we adapt based on the FCT. Other variants include: factor graph BP higher-order factors [13], particles for continuous variables—non-parametric BP [20], top M solutions [33], tree-reweighted message passing BP [27].

When it comes to speeding up LBP, again several approaches can be mentioned: Felzenszwalb and Huttenlocher [7] introduce a distance transform and multiple scale guided LBP, Coughlan and Shen [5] employ a dynamic quantization of the state space, while Potetz and Lee [16] propose higher-order factors with linear interactions. Other solutions are tailored towards specific applications, for example, Yang et al. [32] propose a constant-space BP variant for stereo matching.

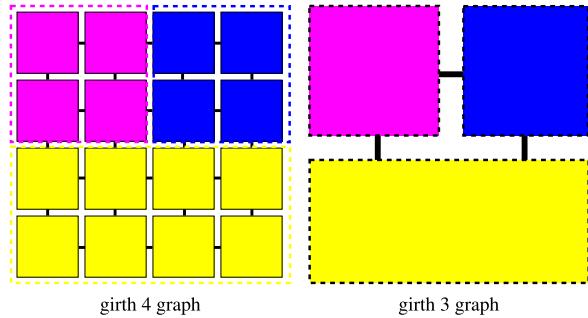
### 11.2.2 Graph Coloring

In graph theory, graph coloring is a subclass of graph labelings [9], which is the assignment of labels to the edges and/or vertices of a graph. We focus on vertex coloring, that is, the assignment of different labels to adjacent vertices. Traditionally, the labels were called “colors”, as coloring the countries on a map was the first problem tackled (see Fig. 11.1 (left)). This problem was later generalized to coloring the faces of a graph embedding in the plane. By planar duality, it became a vertex coloring problem.

A (proper)  $k$ -coloring is a vertex coloring using at most  $k$  different colors, with all neighboring vertices differing in color. Then, the graph is called  $k$ -colorable. The chromatic number of a graph is the smallest number of colors needed to color the graph. A  $k$ -colorable graph is  $k$ -chromatic when  $k$  is its chromatic number.

Deciding for an arbitrary graph if it admits a proper vertex  $k$ -coloring is NP-complete. Finding the chromatic number is thus an NP-hard problem. While there are greedy algorithms able to achieve  $(d + 1)$ -coloring solutions, where  $d$  is maximum degree (adjacencies) of any node in the graph, these do not allow for the derivation of the chromatic number.

**Fig. 11.2** Girth. While the 2D grid connectivity is a girth 4 planar graph (*left*), at a higher level, after merging the grid vertices into regions/segments the new planar graph can have girth 3 (*right*)



tion of the chromatic number, as the actual coloring uses at most  $d + 1$  colors, not exactly  $d + 1$  [31]. Providing a proper vertex  $k$ -coloring solution or counting the number of  $k$ -coloring solutions has been proven to have exponential time complexity.

The MRF labeling problems we work with are defined on grids, which often are combined with a 4-connectivity (see Fig. 11.2). This representation corresponds to a planar graph, that is, a graph that can be drawn in the plane without edge crossings. According to the FCT [17] the chromatic number of the graph is 4—at most 4 different colors are needed such that no adjacent vertices have the same color. For vertex coloring any map on a torus maximum 7 colors are necessary. On a projective plane, Klein bottle, or Moebius strip, one needs 6 colors. For spheres just 4 are necessary, as for planes. For more details, we refer the reader to [30]. Another vertex coloring result is given by Groetzsch's theorem [12]. If the planar graph is triangle-free, that is, the length of the shortest cycle (or girth, see Fig. 11.2) is at least 4, then the chromatic number of the graph is 3 (see Fig. 11.1 (right) for an example with girth 3 and thus requiring at least 4 colors).

The MRF labeling problems working on image grids can be seen as a joint optimization for the optimal image/grid segmentation into regions sharing the same label and the optimal assignment of such labels for each segmented region. The underlying segmentation can be seen as a planar graph, as if the regions are the equivalent of countries on a map and the region adjacencies are inherited from the local adjacencies on the initial grid. Thus, we have a vertex coloring problem for a planar graph. Provided that the regions/countries/segments are fixed, the four-color theorem guarantees that 4 colors are sufficient for a 4-coloring solution of the underlying segmentation. For each 4-colored segment, a final label assignment is to be optimized. Each image pixel is part of one color segment, and the number of color states is bounded by 4 as a result of the FCT. Moreover, the MRF label is carried at color state level as a property of each color segment, that is, for each color a single label is carried. During the MRF optimization process, both label and color state segment are continuously estimated by means of message passing under the Loopy Belief Propagation umbrella. Graph coloring theory thus provides a powerful argument that we can reduce the number of states when segmenting an image using an MRF labeling formulation.

Note that starting from the triangle-free planar graph of pixels does not imply that the planar graph of the underlying segmentation is a triangle-free planar graph

as well. These segment adjacencies follow from merging the pixel-wise adjacencies. On the other hand, edge crossings are still impossible as a planar graph is still a planar graph whenever two adjacent nodes are merged. In summary, we still have a planar graph, but it no longer needs to be triangle-free and the girth can thus decrease to 3 (one example in Fig. 11.2). This calls for not using only 3 color states (as Groetzsch's theorem would allow for triangle-free planar graphs). In the experimental Sect. 11.5.5, the proper number of color states is empirically validated and comes out to be 4, as theoretically expected.

### 11.3 Loopy Belief Propagation

For inference on MRFs, Loopy Belief Propagation can be used [29]. In particular, the max-product approach finds an approximate minimum cost labeling of energy functions in the form of (11.1). Indeed, as an alternative to a formulation in terms of probability distributions, an equivalent formulation uses negative log probabilities, where the max-product becomes a min-sum. Following Felzenszwalb and Huttenlocher [7], we use this formulation as it is numerically more robust and makes more direct use of the energy function.

The max-product BP algorithm passes messages around on a graph defined by the 4-connected image grid. Each message is a vector, with the number of possible labels as dimension (some examples in Fig. 11.3). Let  $m_{p \rightarrow q}^t$  be the message that node  $p$  passes on to a neighboring node  $q$  at time (iteration)  $t$ ,  $0 < t \leq T$ ,  $p, q \in \mathcal{P}$ .  $T$  is the total number of iterations. Consistent with the negative log formulation, all entries in  $m_{p \rightarrow q}^0$  are initialized to zero. At each iteration  $t$ , new messages are computed based on the previous ones from iteration  $t - 1$ , as follows:

$$m_{p \rightarrow q}^t(f_q) = \min_{f_p} \left( V(f_p, f_q) + D_p(f_p) + \sum_{g \in \mathcal{N}(p) \setminus q} m_{g \rightarrow p}^{t-1}(f_p) \right) \quad (11.3)$$

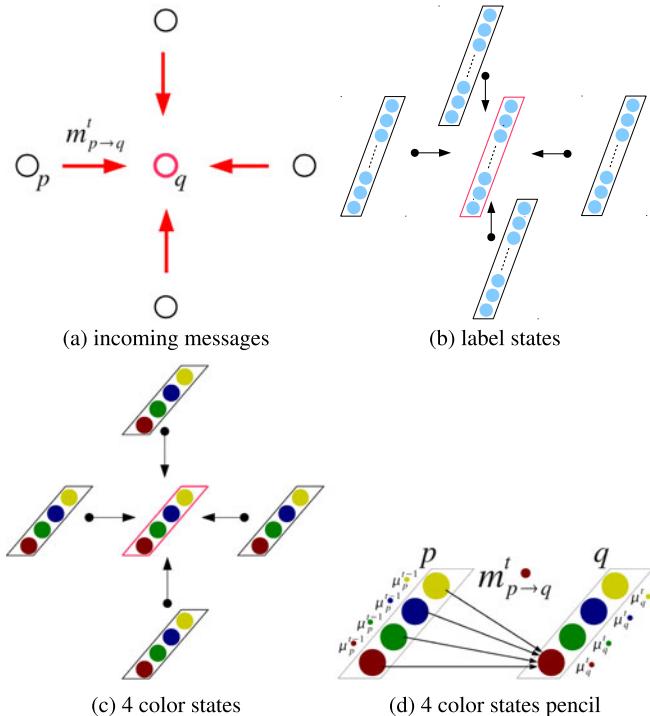
where  $\mathcal{N}(p) \setminus q$  denotes all of  $p$ 's neighbors, except  $q$ , and  $f_q \in \mathcal{L}$  is the label of  $q$ . After  $T$  iterations, a belief vector  $\hat{m}_q$  is computed for each node  $q \in \mathcal{P}$ .  $\hat{m}_q$  is of size equal with the number of labels,  $|\mathcal{L}|$ , having one entry for each potential label  $f_q \in \mathcal{L}$ , and  $\mathcal{N}(q)$  is the set of neighbors for  $q$ :

$$\hat{m}_q(f_q) = D_q(f_q) + \sum_{p \in \mathcal{N}(q)} m_{p \rightarrow q}^T(f_q) \quad (11.4)$$

Finally, for each node  $q$ , the best label  $\hat{f}_q$  that minimizes  $\hat{m}_q(f_q)$  is selected:

$$\hat{f}_q = \arg \min_{f_q} \hat{m}_q(f_q) \quad (11.5)$$

The standard implementation of this message passing algorithm runs in  $O(nk^2T)$  time, with  $n$  the number of pixels,  $n = |\mathcal{P}|$ ,  $k$  the number of possible labels,  $k = |\mathcal{L}|$ ,



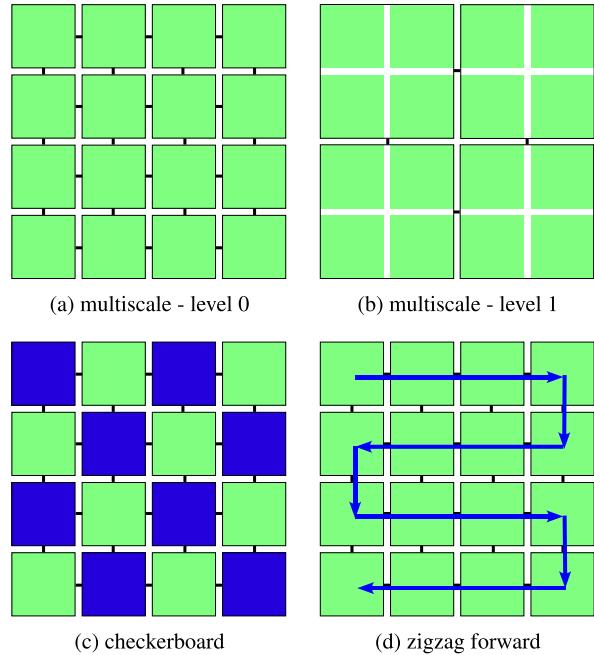
**Fig. 11.3** Local neighborhoods in BP with message passing (a) and a pencil example (d). Note the reduction in working states from the traditional approaches (= number of labels) (b) to only 4 color states (c)

and  $T$  the number of iterations. Essentially it takes  $O(k^2)$  time to compute each message vector between neighboring nodes (see (11.3)) and there are  $O(n)$  messages per iteration. In [7], the time of computing each message is reduced to  $O(k)$ . This acceleration is based on the distance transform for particular classes of data cost functions  $D$  such as truncated linear and quadratic models combined with a Potts model for the discontinuity cost  $V$ . Thus, the algorithm has  $O(nkT)$  time complexity.

### 11.3.1 Multiscale BP on the Grid Graph

As in [7], we can partition the 2D grid in a checkerboard pattern where every edge connects nodes of different partitions, so that the grid graph is bipartite (see Fig. 11.4(c)). We then compute only half of the messages, for example, from the green nodes to the blue nodes in Fig. 11.4(c). Then we swap partitions and do the same. This process of partial message passing in 2 subsequent tacts is repeated until convergence or a maximum number of iterations has been reached. This scheme has

**Fig. 11.4** Levels in multiscale **(a)**, **(b)**, a checkerboard **(c)**, and zigzag forward traversal for a 2D grid **(d)**



the advantage of requiring no additional memory space for storing the updated messages. Another technique explained in [7] and also used here addresses the problem of needing many iterations of message passing to cover large distances. One solution is to perform BP in a coarse-to-fine manner, so that the long-range interactions between nodes are captured by shorter ones in coarser graphs. The minimization function does not change. In this hierarchical approach, BP runs at one resolution level in order to get estimates for the next finer level. Better initial estimates from the coarser level help to speed up the convergence at the finer level. The pyramidal structure works as follows. The zero-th level is the original image (see Fig. 11.4(a)). The  $i$ th level corresponds to a lower-resolution version with blocks of  $2^i \times 2^i$  pixels grouped together. The resulting blocks are still connected in a grid structure (see Fig. 11.4(a), (b), showing the first two levels as an example). For a block  $\mathcal{B}$ , the adapted data cost of assigning label  $f_{\mathcal{B}}$  is

$$D_{\mathcal{B}}(f_{\mathcal{B}}) = \sum_{p \in \mathcal{B}} D_p(f_p) \quad (11.6)$$

where the sum runs over all pixels  $p$  in the block  $\mathcal{B}$ . The multiscale algorithm first solves the problem at the coarsest level, where the messages are initialized to zero. Subsequent, finer levels take the previous, coarser level as initialization. In the 4-connected grid each node  $p$  sends messages in all 4 directions, *right*, *left*, *up*, *down*. Let  $r_p^t$  be the message sent by node  $p$  to the *right* at iteration  $t$ , and similarly  $l_p^t$ ,  $u_p^t$ ,  $d_p^t$  for the other directions. This is just a renaming. If  $q$  is the

right neighbor of  $p$ , then  $r_p^t = m_{p \rightarrow q}^t$  and  $l_q^t = m_{q \rightarrow p}^t$ . If  $q$  is the upper neighbor of  $p$ , then  $u_p^t = m_{p \rightarrow q}^t$  and  $d_q^t = m_{q \rightarrow p}^t$ . For a node  $p$  at level  $i - 1$ , the messages  $(r_{p,i-1}^0, l_{p,i-1}^0, u_{p,i-1}^0, d_{p,i-1}^0)$  will be initialized with the ones obtained by solving the level  $i$  for the containing block  $\mathcal{B}$ ,  $p \in \mathcal{B}$ :

$$\begin{aligned} r_{p,i-1}^0 &\leftarrow r_{\mathcal{B},i}^T \\ l_{p,i-1}^0 &\leftarrow l_{\mathcal{B},i}^T \\ u_{p,i-1}^0 &\leftarrow u_{\mathcal{B},i}^T \\ d_{p,i-1}^0 &\leftarrow d_{\mathcal{B},i}^T \end{aligned} \quad (11.7)$$

The total number of nodes in a quad-tree is upper bounded by

$$n \left( 1 + \frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \dots \right) = n \sum_{i=0}^{\infty} \frac{1}{4^i} = n \frac{4}{3} \quad (11.8)$$

where  $n$  is the number of nodes at the finest level, that is, in our case, the number of pixels in the image,  $n = |\mathcal{P}|$ . We have  $4/3$  the number of nodes at the finest level,  $n$ , so the overhead introduced by the multiscale approach amounts to only  $1/3$  of the original single scale approach, but it results in a greatly reduced number of iterations (between five and ten iterations per scale instead of hundreds at the finest) for convergence. In general, the number of levels needed is proportional to  $\log_2$  of the image diameter. The diameter is defined as the longest shortest path between any two nodes (pixels) in the connected grid (graph) representation (of the image). The traditional LBP scheme using one level requires at least a number of iterations equal with the diameter of the image in order to allow information propagation between any two nodes.

### 11.3.2 Forward Belief Propagation

While, in theory, the LBP messages are updated simultaneously, we propose a traversing order for the nodes in the 2D grid as another way to speed up the propagation of beliefs. Sequentially applying the message updates in the imposed order allows a belief from one node to influence all the following nodes in the traversing order.

Once the traversal order has been fixed, updating the messages in that order corresponds to a Forward Belief Propagation (FBP), whereas Backward Belief Propagation (BBP) corresponds to going in the reverse order of the traversal. Here we consider the traversing order to be in a zigzag/square wave pattern (see Fig. 11.4(d)). Starting from top-left we first go to the top-right, then we go down one line and take the path back to the left, where we move to the next line and repeat the procedure until no line remains unvisited. This traversing order has some advantages in canceling/reducing the incorrectly propagated beliefs.

The message costs have to be normalized. One way of doing this is to simply divide by the number of messages used for computing them. In our 4-connected grid graph, we divide the costs of the newly updated messages by 4.

Note that our zigzag FBP and BBP can be used in the multiscale framework as described in Sect. 11.3.1. In our implementations, we use the forward traversal order (see Fig. 11.4(d)) in the odd iterations and the backward traversal order in the even iterations. Replacing the checkerboard scheme with FBP traversing in the multiscale BP framework (BP-FH) from [7] provides very similar performance in the stereo and denoising tasks. We refer to this method as Forward Belief Propagation-Multi Scale (FBP-MS). Without multiscale guidance, FBP usually gets stuck in a poor solution.

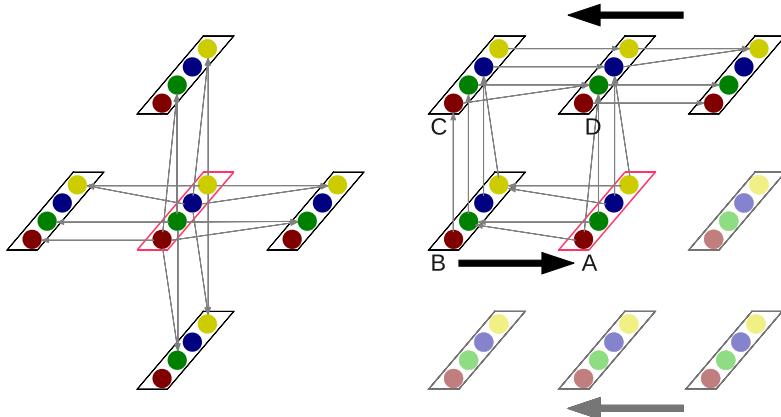
## 11.4 Four Color Theorem in BP

The Four Color Theorem guarantees us, that with only four colors, neighboring segments can always be colored differently. This is important for all low-level processes where keeping boundaries intact is crucial. Examples are segmentation, but also stereo, optical flow, and image denoising. Within the resulting segments, more refined labels have to be assigned to the different pixels (e.g., gradually changing disparities, displacements, or intensities). With only 4 colors to be considered, we have—at a single pixel—the limitation that we can associate with each of the 4 colors only 1 label. This may sound like a drastic restriction. Yet, one should consider the different pixels within the segments, that can each bring 4 such labels to the table. The total number of labels that are considered is therefore a lot larger. The hope is that the message passing will distribute the truly relevant labels. Messages actually contain both color and label information. Following our experimental results, the proper label is, at least for sufficiently large segments, seen to prevail. Note that one could add robustness by additionally considering a randomly selected label for each pixel when the message passing updates are computed. Our results did not call for such addition, however.

Importantly, by keeping only *four* labels, one for each possible color state of one pixel/node in the 2D grid graph, we obtain an algorithm that is linear in the number of pixels but not labels, and therefore is very fast. Moreover, our results are comparable to those obtained when using standard max-product BP, efficient multiscale BP [7], or graph cuts algorithms to minimize energy functions in the form of (11.1).

In Fig. 11.3(d), we show two neighboring nodes/pixels with 4 color states and a *pencil*. The pencil shows the possible connections between one state and the states of a neighboring node/pixel. Here, as usual, only the best connection is used, that is, the one that minimizes a certain energy.

The Four Color Theorem says that we can cover an image with disjoint segments, where each segment has one of no more than four colors, and no two segments sharing a border of more than one pixel have the same color. The total number of



**Fig. 11.5** Four color states pixel neighborhood used in computing message updates, the case for BP-FCT (left) and FBP-FCT (right)

segments for the problems considered here is a priori unknown but upper bounded by the number of image pixels,  $n = |\mathcal{P}|$ . Let  $\mathcal{S}$  be the set of segments. Thus, locally each pixel can belong to one of four color states and the color itself represents the segment of pixels connected through the same color. In (11.1), the labeling  $f$  should assign each pixel  $p$  from  $\mathcal{P}$  to a segment from  $\mathcal{S}$ .

In the next sections, *color state* is the term used to refer to the construct where for each of the 4 colors used for the underlying segmentation we keep energies, estimated labels, and possibly other parameters, locally assigned to each underlying color segment.

### 11.4.1 BP with Four Colors

In order to work with four color states, the labeling  $f$ , in our new formulation, assigns  $f_p \in \mathcal{C}$  to each pixel  $p \in \mathcal{P}$ , where the cardinality of  $\mathcal{C}$  is 4 ( $|\mathcal{C}| = 4$ ). In the left part of Fig. 11.5 a 4-connected grid neighborhood is depicted for the four colors case. The edges/connections to the neighboring nodes are picked to minimize the energies in the BP formulation. For each pixel  $p \in \mathcal{P}$ , each possible color state of that pixel  $c \in \mathcal{C}$ , and each of its neighbors  $q \in \mathcal{N}(p)$ , we have a data parameter that is continuously updated through message passing,  $\mu_{p \rightarrow q}^t(c)$  at iteration  $t$ . Thus,  $\mu_q^t(c)$  is updated by using all the incoming messages, including  $\mu_{p \rightarrow q}^t(c)$ . The data parameter  $\mu_q^t$  (see Fig. 11.3), which represents a quantity to be estimated in each pixel, is the equivalent of the labels in the original BP formulation, where there is a bijection between labels and quantities to be estimated.

The initial values for  $\mu_p^0(c)$  and  $\mu_{p \rightarrow q}^0(c)$  depend on the data; we set them to the best observation we have. For example, in stereo matching, these will be the best scoring disparities in each pixel (in a *winner-take-all* sense) and in image denoising,

the pixel values. When we compute the new messages (using (11.3)), we also store the color of the state,  $\hat{c}_{p \rightarrow q}^t$ , for which we have the minimum message energy at iteration  $t$ :

$$\hat{c}_{p \rightarrow q}^t(f_q) = \arg \min_{f_p \in \mathcal{C}} \left( V(f_p, f_q) + D_p(f_p) + \sum_{g \in N(p) \setminus q} m_{g \rightarrow p}^{t-1}(f_p) \right) \quad (11.9)$$

for every color state  $f_q \in \mathcal{C}$ . Also, at each iteration, part of the message is the data parameter estimation:

$$\mu_{p \rightarrow q}^t(f_q) = \mu_p^{t-1}(\hat{c}_{p \rightarrow q}^t(f_q)) \quad (11.10)$$

for every color state  $f_q \in \mathcal{C}$ . The exact nature of  $V$  and  $D_p$  depends on the application and will be specified in Sect. 11.5.

We call this formulation BP-FCT, standing for Belief Propagation with Four Color Theorem principle.

### 11.4.2 FBP with Four Colors

FBP (see Sect. 11.3.2) does not produce good quality results on its own (e.g., in the absence of multiscale guidance). A solution is to consider a different updating scheme at each step, employing local consistency. Standard message updating assumes that the history and neighborhood belief are stored/propagated in all the messages reaching the current node and takes the best updates. Instead, we exploit the ordering introduced in FBP, and keep track of the links/connections/edges which provided the best costs for each node and color state in the grid. Thus the nodes are processed sequentially, following the imposed order (see Fig. 11.5). We compute the current best costs only based on the connections to the previously processed nodes. These are (for an inner node, i.e.,  $A$  in Fig. 11.5) the previously processed node (in the traversal order,  $B$  in Fig. 11.5) and the neighboring node from the previously processed line ( $D$  in Fig. 11.5).

For each pixel  $p$  from the image,  $p \in |\mathcal{P}|$ , and each possible color state  $c \in \mathcal{C}$ , we keep the minimum obtained energy  $E(p, c)$ , the estimated average image intensity/color coined  $\mu_{p,c}$  (the estimated label) of the segment where  $p$  belongs to, and the estimated number of pixels in the current segment  $N_{p,c}$ .

We use the following notations:

- $c_{\leftarrow}(p, c)$  the color state from the preceding pixel in the FBP traversing order for  $p$  that contributed to the energy  $E(p, c)$ ,  $\forall c \in \mathcal{C}$  (e.g.,  $c_{\leftarrow}(A, blue) = yellow$  in Fig. 11.5).
- $l_{\leftarrow}(p)$  the link to the preceding pixel according to the FBP traversing order for  $p$  (e.g.,  $l_{\leftarrow}(A) = B$  in Fig. 11.5).
- $c_{\uparrow}(p, c)$  the color state from the neighboring pixel in the previous computed raw of pixels in the FBP traversing order for  $p$  that contributed to the energy  $E(p, c)$ ,  $\forall c \in \mathcal{C}$  (e.g.,  $c_{\uparrow}(A, red) = green$  in Fig. 11.5).

$l_{\uparrow}(p)$  the link to the neighboring pixel in the previous raw of pixels according to the FBP traversing order for  $p$  (e.g.,  $l_{\uparrow}(A) = D$  in Fig. 11.5).

The right part of Fig. 11.5 shows the general case where for the current node  $A$  we only use what we know from the already traversed nodes ( $B, C, D$ ). To enforce local consistency, we calculate the energy of each possible color state of a pixel  $A$  not only from the previous pixel  $B$ , but also from the color state of the pixel  $D$  that, through  $C$  influenced that particular color state in  $B$ . For example, in Fig. 11.5, to compute the energy to go from the *green* state in  $B$  to the *red* state in  $A$  we observe that if  $B$  is green, then  $D$  is also green (via the  $C$  node). Therefore, the possible energy in  $A$ 's red state, if  $B$ 's green state is considered as connection, is the sum of the transition energy from  $B$ 's green state and  $D$ 's green state. Computing the best energy for each of  $A$ 's 4 color states requires to first consider all of  $B$ 's states leading there, each time paired with the corresponding state in  $D$ , and to then take the minimum energy among these alternatives.

Thus, the energy  $E(p, c)$  is computed consistently if the links point correctly on the grid:

$$l_{\uparrow}(p) = l_{\leftarrow}(l_{\leftarrow}(l_{\leftarrow}(p))) \quad (11.11)$$

and

$$c_{\uparrow}(p, c) = c_{\leftarrow}(l_{\uparrow}(l_{\leftarrow}(p)), c_{\leftarrow}(l_{\leftarrow}(p), c_{\leftarrow}(p, c))) \quad (11.12)$$

which enforces that the connecting color state  $c_{\leftarrow}(p, c)$  of the previously processed pixel results from the connecting color state  $c_{\uparrow}(p, c)$  of the neighboring pixel from the previously processed line.

In order to compute  $E(p, c)$ , we first compute for each possible color state  $i \in \mathcal{C}$  of the previously processed pixel the energy

$$\begin{aligned} E_{\leftarrow}(p, c, i) &= E(l_{\leftarrow}(p), i) + h((p, c), (l_{\leftarrow}(p), i)) \\ &\quad + h((p, c), (l_{\uparrow}(p), c_{\leftarrow}(l_{\uparrow}(l_{\leftarrow}(p)), c_{\uparrow}(l_{\leftarrow}(p), i)))) \end{aligned} \quad (11.13)$$

cumulating the energy  $E(l_{\leftarrow}(p), i)$  of the previously processed pixel, the energy for color states of the previous and current pixels, as well as the energy for corresponding color states of the current pixel and the connected one on the previous line. Now we can take the lowest energy over all possible color states  $i$  as

$$E(p, c) = \min_{i \in \mathcal{C}} E(l_{\leftarrow}(p), c, i) \quad (11.14)$$

and store the color state from the previously processed pixel which provides the best energy:

$$c_{\leftarrow}(p, c) = \arg \min_{i \in \mathcal{C}} E(l_{\leftarrow}(p), c, i) \quad (11.15)$$

The energy  $h((p, c_p), (q, c_q))$  between two pixels  $p, q \in \mathcal{P}$ , given their corresponding color states  $c_p, c_q \in \mathcal{C}$ , is:

$$h((p, c_p), (q, c_q)) = \begin{cases} D(I(p), I(q), \sigma_c) + D(\mu_{p,c_p}, \mu_{q,c_q}, \sigma_m) & \text{if } c_p = c_q \\ 2 - [D(I(p), I(q), \sigma_c) + D(\mu_{p,c_p}, \mu_{q,c_q}, \sigma_m)] & \text{if } c_p \neq c_q \end{cases} \quad (11.16)$$

where  $I(p)$  is the color value of the pixel  $p$ ,  $\sigma_x$  indicates a noise level for  $x$  used in some of our applications,  $D(I(p), I(q), \sigma_c)$  is the smoothness cost between two pixels  $p$  and  $q$  with color values  $I(p)$  and  $I(q)$ , and  $D(\mu_{p,c_p}, \mu_{q,c_q}, \sigma_m)$  is the data cost agreeing between the estimated working labels/quantities at color segment level  $\mu_{p,c_p}$  and  $\mu_{q,c_q}$ .

The distance/penalty  $D(u, v, \sigma)$  between pixel values penalizes for segment discontinuities between similar values ( $\|u - v\|_1 < 2\sigma^2$ ), whereas if the pixels are very different, the penalty is reasonably small.

$$D(u, v, \sigma) = 1 - \exp\left(-\frac{\|u - v\|_1}{2\sigma^2}\right) \quad (11.17)$$

The function models the distribution of the label noise among neighboring pixels, with  $\sigma$  its variance. For the smoothness cost, we can take a different  $\sigma$  than for the data cost. Here we use  $\sigma_c$  for agreement between pixel color values and  $\sigma_m$  for the agreement between the average labels of segments.

The link to the corresponding pixel state at the previous line is stored in

$$c_{\uparrow}(p, c) = c_{\leftarrow}(l_{\leftarrow}(l_{\leftarrow}(p)), c_{\uparrow}(l_{\leftarrow}(p), c_{\leftarrow}(p, c))) \quad (11.18)$$

Given the previous notations, the estimated label  $\mu_{p,c}$  for a pixel  $p$  and color state  $c$  in a segment and the approximated number of pixels belonging to the segment,  $N_{p,c}$ , are obtained as follows:

$$N_{p,c} = \begin{cases} N_{l_{\leftarrow}(p), c_{\leftarrow}(p, c)} + 1 & \text{if } c = c_{\leftarrow}(p, c) \\ N_{l_{\uparrow}(p), c_{\uparrow}(p, c)} + 1 & \text{if } c = c_{\uparrow}(p, c) \wedge c \neq c_{\leftarrow}(p, c) \\ 1 & \text{otherwise} \end{cases} \quad (11.19)$$

and

$$\mu_{p,c} = \begin{cases} (\mu_{l_{\leftarrow}(p), c_{\leftarrow}(p, c)} N_{l_{\leftarrow}(p), c_{\leftarrow}(p, c)} + I(p))/N_{p,c} & \text{if } c = c_{\leftarrow}(p, c) \\ (\mu_{l_{\uparrow}(p), c_{\uparrow}(p, c)} N_{l_{\uparrow}(p), c_{\uparrow}(p, c)} + I(p))/N_{p,c} & \text{if } c = c_{\uparrow}(p, c) \wedge c \\ & \neq c_{\leftarrow}(p, c) \\ I(p) & \text{otherwise} \end{cases} \quad (11.20)$$

For  $N_{p,c}$ , we either increment the estimated number of pixels of the color segment to which the current pixel is assigned, or we put it to 1 if a new color segment is formed. Following the same logic, the estimated label in the current pixel changes by taking a weighted average over that pixel's label ( $I(p)$ ) and the label of at most one of its

neighbors: the previously updated neighbor if it lies in the same 4-color segment; if not, the neighbor on the previous line if that falls within the same segment; and none if both neighbors fall outside.

## 11.5 Experiments

We now demonstrate how the proposed methods can be used in different early vision applications that can be formulated as energy minimization problems through an MRF model.

For all those applications, we provide details about discontinuity/pairwise costs ( $V(f_p, f_q)$ ), data/unary costs ( $D_p(f_p)$ ), and message updates/computations ( $\mu_p^t(f_p)$ ,  $m_{p \rightarrow q}^t(f_q)$ ). Also, we provide a comparison with well-known standard methods. The images used are from the Berkeley Segmentation Dataset [14], the Middlebury Stereo Datasets [19], and enhancement images from [7].

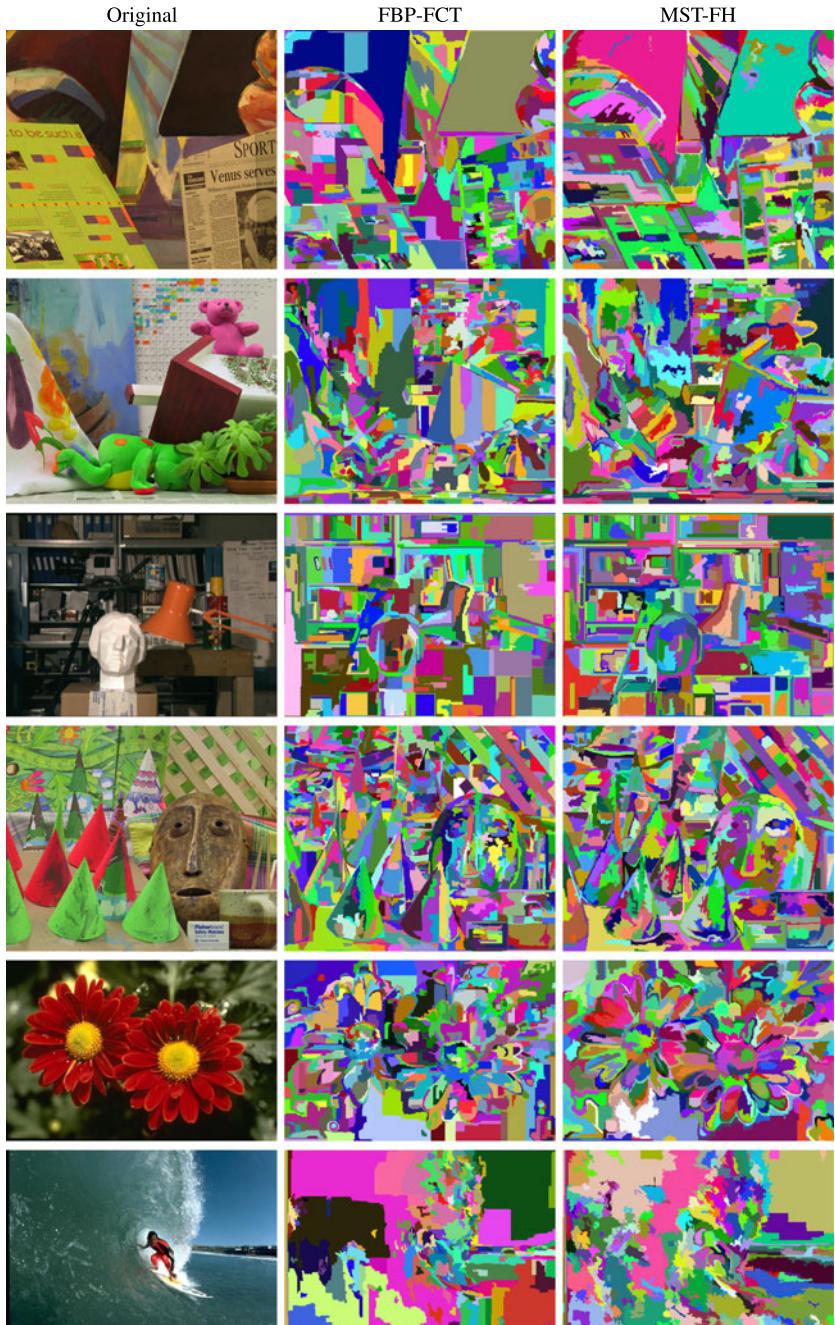
All the provided running times were obtained on an Intel Core 2 Duo T7250 (2.0 GHz/800 Mhz FSB, 2 MB Cache) notebook with 2 GB RAM. More results are available at: <http://homes.esat.kuleuven.be/~rtimofte/>.

### 11.5.1 Image Segmentation

For comparison, we use our proposed method based on the Four Color Theorem—FBP-FCT (see Sect. 11.4.2) versus a standard Minimum Spanning Tree based method—MST-FH from [6].

The oversegmentation in our FBP-FCT is addressed in a similar fashion to MST-FH in [6]. The smaller (than an imposed minimum size) segments are merged with other segments until no segment with a size less than the imposed minimum remains. We use the original MST-FH implementation with the original parameters ( $\sigma = 0.8$ ,  $k = 300$ ). Our FBP-FCT method uses an initial smoothing of the image with  $\sigma = 0.7$ ,  $\sigma_c = 3.2$ , and  $\sigma_m = 4.2$  (see (11.17)). We set  $k = 300$  and the minimum segment size for both methods is 50 pixels. Figure 11.6 depicts the image segmentation results for several cases.

MST-FH is an  $O(n \log n)$  algorithm, while FBP-FCT is  $O(n|\mathcal{C}|^2)$ , where  $n$  is the number of pixels in the image, i.e.  $n = |\mathcal{P}|$ . This means (and our tests show) that FBP-FCT and MST-FH have comparable running times on low resolution images while for high resolution images ( $\log n > |\mathcal{C}|^2$ ), the FBP-FCT is faster. For example, the *Venus* RGB image with  $434 \times 383$  pixels (top-left in Fig. 11.6) is processed by MST-FH in about 250 milliseconds, while our approach takes 320 milliseconds. For the upscaled image with  $3472 \times 3064$  pixels our FBP-FCT runs in 16 seconds, half a second less than the MST-FH method. Using fewer than 4 color states largely improves the running time of our method, as we show in Sect. 11.5.5.



**Fig. 11.6** Segmentation results. Minimum segment size set to 50 pixels

### 11.5.2 Image Denoising

The image denoising problem is a case where usually the number of labels (in an MRF/BP formulation) is very large since it is equal to the number of intensity levels/colors used. We argue that this way of seeing the problem, besides being computationally demanding, does not take advantage of the relation that exists between labels as intensities. The intensity values are obtained through uniform sampling of a continuous signal, therefore carry direct information on their relative closeness. Instead of updating labels through selection from neighboring labels, as with standard BP, our scheme—which maintains the discrete nature of the 4 colors—can submit the actual, neighboring labels to mathematical operations such as averaging.

In the BP-FCT formulation (see Sect. 11.4.1) we take the following updating function for the intensity data at each message:

$$\mu_q^t(f_q) = \frac{\alpha I(q) + \sum_{p \in \mathcal{N}(q) \setminus q} (\mu_{p \rightarrow q}^t(f_q) [\hat{c}_{p \rightarrow q}^t(f_q) = f_q])}{\alpha + \sum_{p \in \mathcal{N}(q) \setminus q} [\hat{c}_{p \rightarrow q}^t(f_q) = f_q]} \quad (11.21)$$

where  $[\cdot = \cdot]$  is the Iverson bracket for Kronecker's delta, that is, it returns 1 for equality, 0 otherwise, and  $\alpha$  is the weight/contribution of the observed value  $I(q)$  in the updated data term. Here we use  $\alpha = 0.25$ .  $\hat{c}_{p \rightarrow q}^t(f_q)$  is defined by (11.9) and represents the best color state for which we achieved the minimum message energy at iteration  $t$ .

The data cost for a pixel  $p$ , at iteration  $t$  is taken as

$$D_p(f_p) = \min(\|I(p) - \mu_p^{t-1}(f_p)\|_2, \tau_1) \quad (11.22)$$

where  $\tau_1$  acts as a truncation value, that is empirically set.  $D_p$  depends on the difference between the estimated intensity value  $\mu_p^t(f_p)$  at iteration  $t$  and the observed one  $I(p)$  for the pixel  $p$  under the labeling (color state assignment)  $f_p$ .

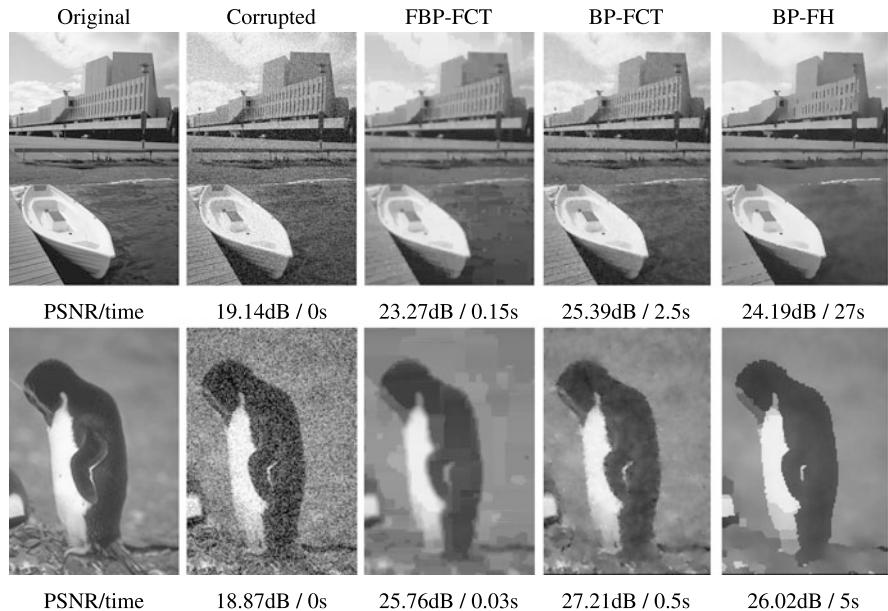
The discontinuity cost is given by:

$$V(f_p, f_q) = \beta \min(\|\mu_p^{t-1}(f_p) - \mu_q^{t-1}(f_q)\|_2, \tau_2) \quad (11.23)$$

where  $\beta$  is a scaling factor and  $\tau_2$  acts as a truncation value.

We compare BP-FCT (Sect. 11.4.1) and FBP-FCT (Sect. 11.4.2) with BP-FH [7]. For this purpose the gray-scale images are corrupted by adding independent and identically-distributed Gaussian noise with zero-mean and variance 30. We use the original available implementation for multiscale BP-FH. We apply for FBP-FCT a Gaussian smoothing with standard deviation 1.5 before processing the corrupted images, while BP-FH was demonstrated using a Gaussian filtering with a standard deviation of 1.5. We set  $\sigma_c = \sigma_m = 3.2$  for FBP-FCT, while BP-FCT uses for discontinuity truncation  $\tau_2 = 200$ , for data truncation  $\tau_1 = 10000$  and for the rate of increase the cost  $\beta = 1$ . While BP-FH uses 5 iterations, BP-FCT uses 20.

Reducing the number of labels when  $|\mathcal{L}| \gg |\mathcal{C}|^2$  assures a considerable speed-up of our proposed methods (BP-FCT— $O(n|\mathcal{C}|^2T)$ , FBP-FCT— $O(n|\mathcal{C}|^2)$ ), over



**Fig. 11.7** Denoising results. BP-FCT and FBP-FCT perform better or similar when compared with BP-FH (in terms of PSNR) while being 1 up to 2 orders of magnitude faster

the standard multi-scale BP-FH which has a time complexity of  $O(n|\mathcal{L}|T)$  (with  $n$  the number of pixels and  $T$  the number of iterations). Image denoising results are shown in Fig. 11.7. For all the test images BP-FCT and FBP-FCT had a better or similar denoising performance (in terms of PSNR) than the BP-FH method, while being up to 100 times faster. For example, the *Boat* gray-scale corrupted image with  $321 \times 481$  pixels (top row in Fig. 11.7) is processed by FBP-FCT in about 150 milliseconds, by BP-FCT in 2.5 seconds, while BP-FH takes more than 27 seconds. The computed PSNR (in decibels) for the *Penguin* images are:  $PSNR_{\text{Corrupted}} = 18.87$ ,  $PSNR_{\text{FBP-FCT}} = 25.76$ ,  $PSNR_{\text{BP-FCT}} = 27.21$ ,  $PSNR_{\text{BP-FH}} = 26.02$ . We refer the reader to the figure for the complete set of numbers for all images. Having a denoising method that works for a single channel image (gray levels), usually the multi-channel images (e.g., RGB) are processed for each channel individually and the denoised image is the union of the denoised channels.

### 11.5.3 Stereo Matching

For stereo matching, in the BP-FCT (see Sect. 11.4.1) framework we define the following cost and update functions. We employ the standard settings using a Disparity Space Image (DSI) computed as follows using the left and right intensity images,  $I_l$  and  $I_r$ .

For a pixel  $p = (i, j)$  whose intensity is  $I_l(i, j)$  in the left camera image, the cost for a disparity of  $d$  is:

$$DSI(p, d) = \beta \min(\|I_l(i, j) - I_r(i - d, j)\|^2, \tau_s) \quad (11.24)$$

where  $\beta = 0.1$  and  $\tau_s = 10000$ .

The data cost at iteration  $t$  is defined for each point  $p$  in the left image and color state  $f_p \in \mathcal{C}$  as:

$$D_p(f_p) = DSI(p, \mu_p^{t-1}(f_p)) \quad (11.25)$$

where  $\mu_p^{t-1}(f_p)$  is the previous disparity estimate for the pixel  $p$  and the same color state  $f_p \in \mathcal{C}$ .

The data update at iteration  $t$  is:

$$\mu_q^t(f_q) = \begin{cases} \arg \min_d DSI(p, d) & \text{if } \mathcal{W}(q) = \emptyset \\ \arg \min_{\mu_{p \rightarrow q}^t(f_q), p \in \mathcal{W}(q)} (DSI(p, \mu_{p \rightarrow q}^t(f_q))) & \text{otherwise} \end{cases} \quad (11.26)$$

where  $\mathcal{W}(q)$  is

$$\mathcal{W}(q) = \{p \in \mathcal{N}(q) | \hat{c}_{p \rightarrow q}^t(f_q) = f_q\} \quad (11.27)$$

and represents the set of neighbors from the 4-neighborhood  $\mathcal{N}(q)$ , as picked in the best energy computation that share the same color state  $f_q$  (thus underlying color segment) with  $q$ .  $\hat{c}_{p \rightarrow q}^t(f_q)$  is computed using (11.9).

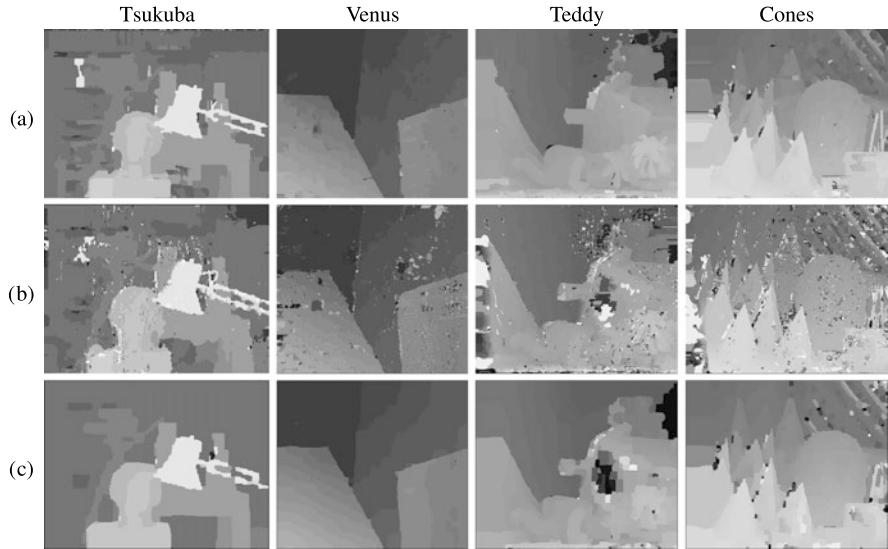
The discontinuity cost is given by:

$$V(f_p, f_q) = \begin{cases} 0 & \text{if } f_p = f_q \\ \tau_v & \text{otherwise} \end{cases} \quad (11.28)$$

where  $\tau_v = 40$  in our experiments.

Figure 11.8 depicts results of the BP-FCT method in comparison with BP-FH. Here we see a case where our approach does not improve upon the full BP-FH [7] formulation. The main reason for the poorer performance is the discrete nature of the cost function. There is no smooth transition in costs from one disparity to a neighboring one with respect to difference in absolute values. This makes it difficult to define an updating function for the data estimation when we work with four colors. Our intuition is to pick the best disparity from the same color segment and in the absence of connections to neighbors with the same color, to return to the best observed disparity (see (11.26)). The drawback is that our proposed method needs more iterations per level to achieve a performance similar to BP-FH. In our experiments, it takes between five and ten times more iterations than BP-FH to achieve similar performance, however this is not directly seen in the running time for large inputs, since our method has  $O(n|\mathcal{C}|^2T)$  complexity and BP-FH  $O(n|\mathcal{L}|T)$ . Starting from the case where  $|\mathcal{L}| > 4|\mathcal{C}|^2$  our proposed method (BP-FCT) is similar or considerably faster ( $|\mathcal{L}| \gg 4|\mathcal{C}|^2$ ).

For stereo matching based on image segmentation, our proposed segmentation method (FBP-FCT) can be integrated as a fast oversegmentation step. Figure 11.8



**Fig. 11.8** Stereo matching results. **(a)** Our implementation of [25] where for image segmentation we use our FBP-FCT, **(b)** BP-FCT, **(c)** BP-FH from [7]

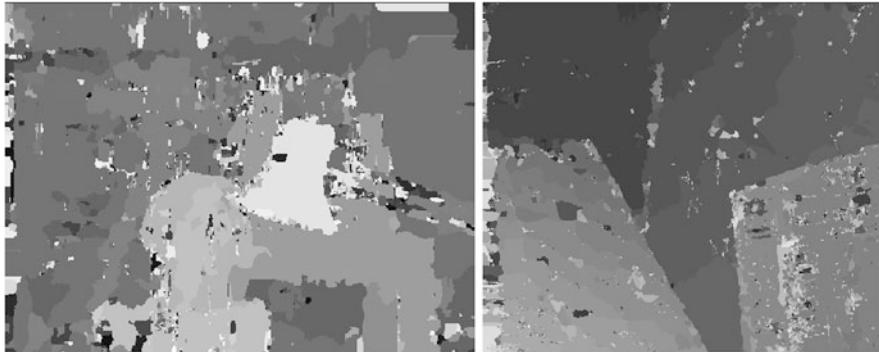
depicts results where we used our segmentation and our implementation of the pipeline from [25]. This is a *winner-take-all* method that combines fast aggregation of costs in a window around each pixel with costs from the segment support to which the pixel belongs. According to the Middlebury benchmark [19], this implementation ranks 76th out of 88 current methods.

#### 11.5.4 Optical Flow

In motion flow estimation, the labels correspond to different displacement vectors. While in stereo matching the disparities were evaluated along the scanline, here the displaced/corresponding pixels are to be found in a surrounding two-dimensional window in the paired images. Thus, the set of labels goes quadratic when compared with stereo. We are using the cost functions as defined for stereo matching in the BP-FCT framework (see Sect. 11.5.3), where  $d$  is a mapping for displacements. For a pixel  $p = (i, j)$  whose intensity is  $I_l(i, j)$  in the first (left) image, the cost for a displacement of  $d(i, j) = (d_i, d_j)$  is:

$$DSI(p, d) = \beta \min(\|I_l(i, j) - I_r(i - d_i, j - d_j)\|^2, \tau_s) \quad (11.29)$$

where  $\beta = 0.1$  and  $\tau_s = 10000$ . The other cost functions are given by the (11.25), (11.26), and (11.28). We keep the same parameter values as in the stereo case, Sect. 11.5.3.



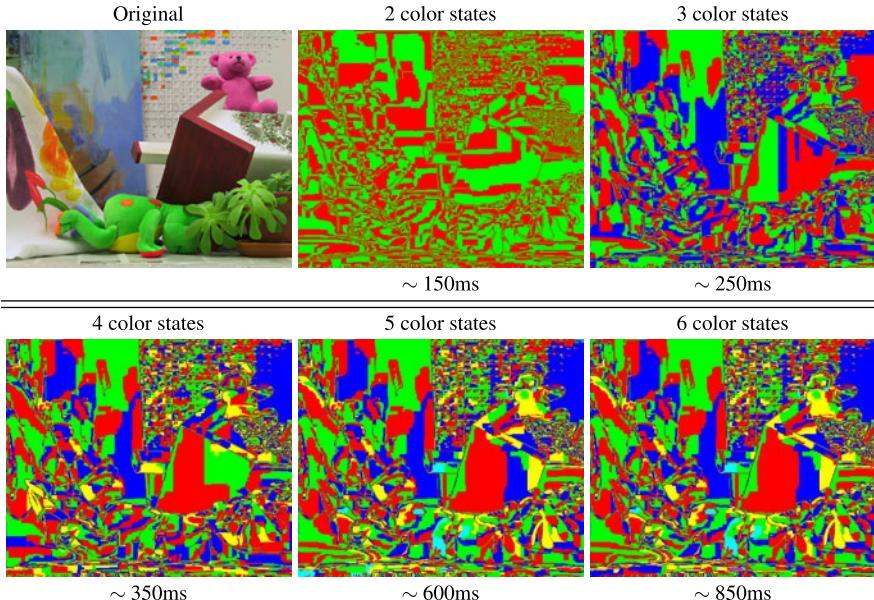
**Fig. 11.9** Optical flow results for the *Tsukuba* and *Venus* image pairs

Note that increasing the set/space of displacement values will not increase the computational time of our Four-Color Theorem based BP approach, since the size of the actual label set is decoupled from that of the placeholder labels, that is, the four colors. However, the *DSI* still needs to be computed for having good initial estimates. In our case, the optical flow case takes as much running time as the stereo case (see Sect. 11.5.3) for the same image size, apart from differences in initialization and the selection of the best energy displacement per pixel which is slower for optical flow, given its higher number of actual labels. Figure 11.9 shows the results for the optical flow computation on the same standard stereo image pairs. For this, we are considering as disparity the distance from the left pixel to the corresponding pixel in the right image. We see that the results are worse but very close to the ones obtained in the stereo-specific formulation (from Fig. 11.8). Increasing the number of displacements from 16(20) in the *Tsukuba* (*Venus*) pair in the stereo case to as much as 1024(1600) (about two orders magnitude bigger) causes a drop of only 4 % in the quality of the results, but does not increase the computational time of the BP-FCT algorithm. However, the *DSI* computation time (and the winner-take-all initialization) taken individually increases linearly with the number of possible displacements.

### 11.5.5 Theory and Parameters in Practice

In Sect. 11.2.2, we reviewed the graph theoretical foundations supporting our choice of 4 color states, which was based on the four-color theorem. Here we empirically verify that indeed 4 color states is the right number in our conditions. We want to prove that using more than 4 color states is: (i) suboptimal, slowing down the BP inference, and (ii) the solution does not improve. Also, if we reduce the number of color states below 4 then we have: (i) insufficient color states, but faster BP inference, and (ii) the obtained solution worsens.

We use the previous settings for image segmentation (Sect. 11.5.1) and image denoising (Sect. 11.5.2). For image segmentation with FBP-FCT, we first apply a

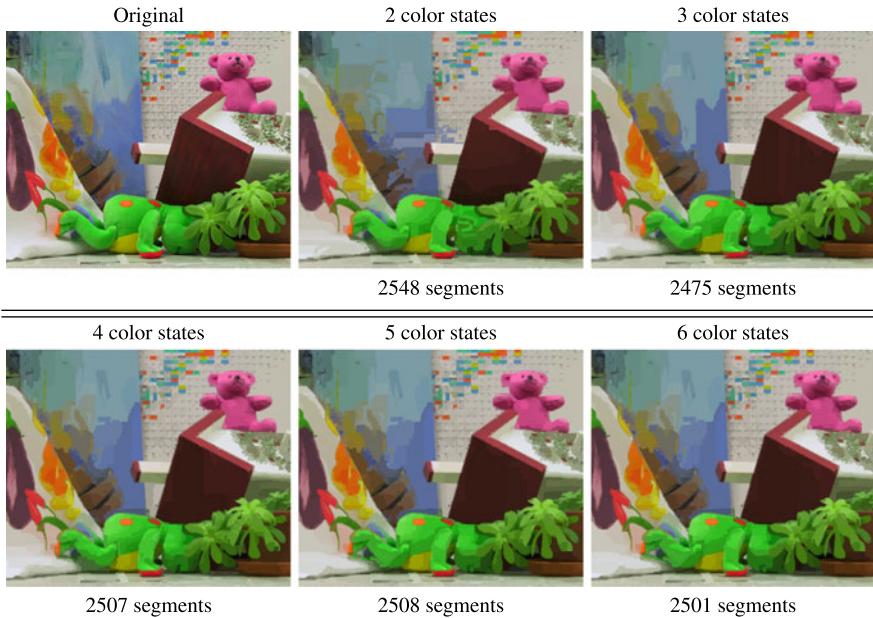


**Fig. 11.10** Number of color states versus FBP-FCT segmentation results on *Teddy* image. Using more than 6 color states usually does not change the segmentation. Already from 5 to 6 color states the changes in segmentation are minor

Gaussian smoothing with standard deviation 0.7, followed by our method with  $\sigma_c = 4.2$  for pairwise costs (at pixel level) and  $\sigma_m = 3.2$  for data costs (at color segment level).

For image denoising results with FBP-FCT, we apply a Gaussian smoothing of the image with standard deviation 0.8, followed by our method with  $\sigma_c = 3.2$  for pairwise costs (at pixel level) and  $\sigma_m = 3.2$  for data costs (at color segment level). For image denoising with BP-FCT, no prior smoothing is employed and the number of iterations of message passing updates per each scale is fixed to 20.

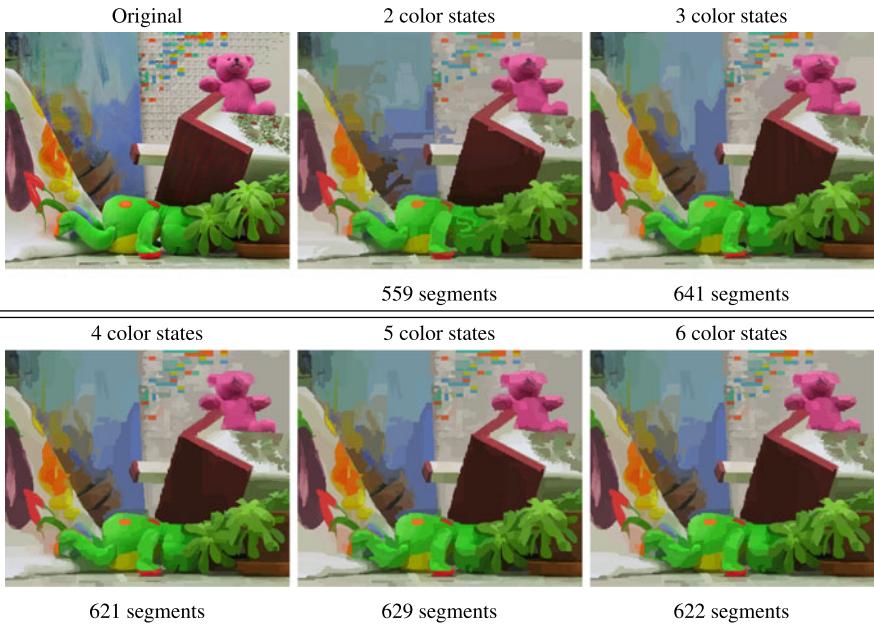
In Figs. 11.10, 11.11, and 11.12, we depict the impact of the number of color states, as used in the Forward BP algorithm, on the segmentation results. Figure 11.10 shows the underlying segmentation as obtained for different numbers of color states used for FBP-FCT, as well as the running times. Figures 11.11 and 11.12 illustrate the segmentation by filling in the segments with the color of the first pixel of the segment in forward traversal order (Fig. 11.11) or an average of such pixel colors whenever the segments were merged to meet the minimum 50 pixels size (Fig. 11.12). We qualitatively see that the best image segmentation results are achieved for 4 color states, and quantitatively, as expected, the running time increases quadratically with the number of color states. Also, the changes from 4 color states to 5 are small, while between 5 color states and 6 color states the differences are hardly noticeable. Using more than 6 color states does not change the segmentation when compared with the 6 color states result.



**Fig. 11.11** Number of color states versus FBP-FCT segmentation results on *Teddy* image. The minimum segment size is 1. Each segment is colored using the color of the first pixel in the segment in the forward traversal order

For the denoising experiment, we report the running time (in seconds, s) and the PSNR (in decibels, dB) for each FBP-FCT and BP-FCT setting with 2 up to 6 color states. As seen in Fig. 11.13, the best PSNR is reached for 4 color states. Using more color states does not necessarily help the denoising results but slows down the process. Note that the underlying segmentation of FBP-FCT does not necessarily imply pixelwise equal denoised labels/intensities within each color state segment. This is because FBP-FCT actually is a single, consistent FBP iteration and the assigned denoised intensities are the best as computed locally in the forward traversal order. Thus, locally within each color state segment the labels can evolve smoothly. The larger the segments are, the bigger the label differences within them can get.

In the underlying segmentations for both the segmentation and denoising experiments, the segments only change visibly up to the use of 6 colors. Within that initial range, the changes get smaller as the number of colors increases. We did not even try and match colors between corresponding segments. Anyway, these experimental results are in agreement with graph coloring theory. As a matter of fact, that theory also states that at most 6 colors are required to color any map on the plane or the sphere [8]. Of course, and as repeatedly mentioned in the paper, the Four Color Theorem [17] demonstrates that the necessary number of colors is 4. Yet, such coloring is NP-complete and using more colors relaxes the situation, allowing for polynomial time complexity colorings. Increasing the number of colors further eases the task.



**Fig. 11.12** Number of color states versus FBP-FCT segmentation results on *Teddy* image. The minimum segment size is 50. Each segment is colored using the color of the first pixel in the segment in the forward traversal order or the average of such colors when merged to form segments with  $\geq 50$  pixels

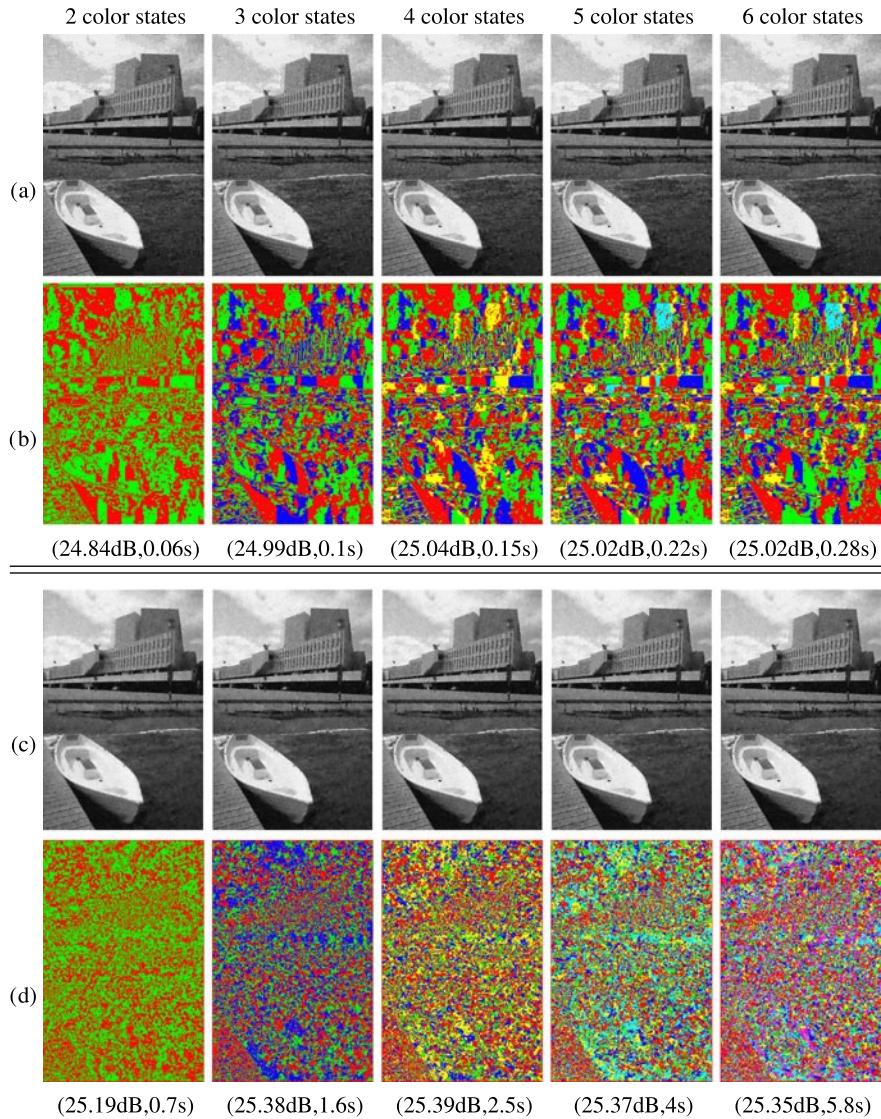
Our task is even harder, as we not only optimize the colors but also the number of segments, their labels, and their boundaries.

From these experiments we can empirically conclude that, under our settings, the four color theorem holds in practice. Four color states suffice for the best trade-off between performance and running time of our BP variants.

## 11.6 Conclusions

We have presented how the Four-Color Theorem based on the max-product belief propagation technique can be used in early computer vision for solving MRF problems where an energy is to be minimized. Our proposed methods yield results that are comparable with other methods, but improve either the speed for large images and/or large label sets (the case of image segmentation, stereo matching and optical flow), or both the performance and speed (the case of image denoising).

The Four Color Theorem principle is difficult to apply in cases where the label set is discrete and no natural order/relation between them can be inferred. This is the case for stereo matching and optical flow, where the disparity cost function takes discrete, unrelated values. This causes slower convergence, but is compensated by the low time complexity of the methods, independent of the number of labels. Thus,



**Fig. 11.13** Number of color states versus denoising results for the *Boat* image with FBP-FCT (a), (b) and BP-FCT (c), (d). We report (PSNR, running time) and show the final denoised images together with the underlying segmentations and corresponding color states. The PSNR peaks at 4 color states. Adding more color states does not improve on it

the proposed methods perform faster than the standard methods considered here, at least for large inputs.

**Acknowledgements** This work was supported by the Flemish Hercules Foundation project RICH and the Flemish iMinds project on Future Health.

## References

1. Agarwal S, Belongie S (2002) On the non-optimality of four color coding of image partitions. In: IEEE international conference on image processing
2. Besag J (1986) On the statistical analysis of dirty pictures Julian Besag (with discussion). *J R Stat Soc B* 48(3):259–302
3. Boykov Y, Veksler O, Zabih R (2001) Fast approximate energy minimization via graph cuts. *IEEE Trans Pattern Anal Mach Intell* 23:1222–1239
4. Chou PB, Brown CM (1990) The theory and practice of Bayesian image labeling. *Int J Comput Vis* 4(3):185–210
5. Coughlan J, Shen H (2007) Dynamic quantization for belief propagation in sparse spaces. *Comput Vis Image Underst* 106(1):47–58
6. Felzenszwalb PF, Huttenlocher DP (2004) Efficient graph-based image segmentation. *Int J Comput Vis* 59:167–181
7. Felzenszwalb PF, Huttenlocher DP (2006) Efficient belief propagation for early vision. *Int J Comput Vis* 70:261–268
8. Franklin P (1934) A six colour problem. *J Math Phys* 13:363–369
9. Gallian JA (2011) A dynamic survey of graph labeling. *Electron J Comb* 18:1–256
10. 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
11. Greig DM, Porteous BT, Seheult AH (1989) Exact maximum a posteriori estimation for binary images. *J R Stat Soc B* 51:271–279
12. Groetzschi H (1958/1959) Zur Theorie der diskreten Gebilde. VII. Ein Dreifarbensatz für dreikreisfreie Netze auf der Kugel. (German) *Wiss Z, Martin-Luther-Univ Halle-Wittenb, Math-Natwiss Reihe* 8:109–120
13. Kschischang FR, Frey BJ, Loeliger H-A (2001) Factor graphs and the sum-product algorithm. *IEEE Trans Inf Theory* 47:498–519
14. 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: International conference on computer vision
15. Pearl J (1982) Reverend Bayes on inference engines: a distributed hierarchical approach. In: Second national conference on artificial intelligence, AAAI-82, Pittsburgh, PA. AAAI Press, Menlo Park, pp 133–136
16. Potetz B, Lee TS (2008) Efficient belief propagation for higher-order cliques using linear constraint nodes. *Comput Vis Image Underst* 112(1):39–54
17. Robertson N, Sanders DP, Seymour PD, Thomas R (1996) A new proof of the four colour theorem. *Electron Res Announc Am Math Soc* 2:17–25
18. Rother C, Kolmogorov V, Lempitsky V, Szummer M (2007) Optimizing binary MRFs via extended roof duality. In: IEEE conference on computer vision and pattern recognition
19. Scharstein D, Szeliski R (2002) A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int J Comput Vis* 47:7–42
20. Sudderth E, Ihler A, Freeman W, Willsky A (2003) Nonparametric belief propagation. In: Conference on computer vision and pattern recognition
21. Sun J, Zheng NN, Shum HY (2003) Stereo matching using belief propagation. *IEEE Trans Pattern Anal Mach Intell* 25:787–800
22. Szeliski R, Zabih R, Scharstein D, Veksler O, Kolmogorov V, Agarwala A, Tappen M, Rother C (2008) A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. *IEEE Trans Pattern Anal Mach Intell* 30(6):1068–1080
23. Tappen MF, Freeman WT (2003) Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In: International conference on computer vision
24. Timofte R, Van Gool L (2010) Four color theorem for fast early vision. In: Asian conference on computer vision
25. Tombari F, Mattoccia S, Di Stefano L, Addimanda E (2008) Near real-time stereo based on effective cost aggregation. In: International conference on pattern recognition

26. Vese LA, Chan TF (2002) A multiphase level set framework for image segmentation using the Mumford and Shah model. *Int J Comput Vis* 50:271–293
27. Wainwright M, Jaakkola T, Willsky A (2005) Map estimation via agreement on trees: message-passing and linear programming. *IEEE Trans Inf Theory* 51:3697–3717
28. Wang C, Paragios N (2012) Markov random fields in vision perception: a survey. INRIA research report, 25 September (2012), pp 1–42
29. Weiss Y, Freeman WT (2001) On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. *IEEE Trans Inf Theory* 47:723–735
30. Weisstein EW (2012). Map coloring. From MathWorld—a Wolfram web resource. <http://mathworld.wolfram.com/MapColoring.html>
31. Welsh DJA, Powell MB (1967) An upper bound for the chromatic number of a graph and its application to timetabling problems. *Comput J* 10(1):85–86
32. Yang Q, Wang L, Ahuja N (2010) A constant-space belief propagation algorithm for stereo matching. In: IEEE conference on computer vision and pattern recognition, pp 1458–1465
33. Yanover C, Weiss Y (2003) Finding the M most probable configurations using loopy belief propagation. In: NIPS

# Chapter 12

## Boosting $k$ -Nearest Neighbors Classification

**Paolo Piro, Richard Nock, Wafa Bel Haj Ali, Frank Nielsen,  
and Michel Barlaud**

**Abstract** A major drawback of the  $k$ -nearest neighbors ( $k$ -NN) rule is the high variance when dealing with sparse prototype datasets in high dimensions. Most techniques proposed for improving  $k$ -NN classification rely either on deforming the  $k$ -NN relationship by learning a distance function or modifying the input space by means of subspace selection. Here we propose a novel boosting approach for generalizing the  $k$ -NN rule. Namely, we redefine the voting rule as a strong classifier that linearly combines predictions from the  $k$  closest prototypes. Our algorithm, called UNN (Universal Nearest Neighbors), rely on the  $k$ -nearest neighbors examples as weak classifiers and learn their weights so as to minimize a *surrogate risk*. These weights, called *leveraging coefficients*, allow us to distinguish the most relevant prototypes for a given class. Results obtained on several scene categorization datasets display the ability of UNN to compete with or beat state-of-the-art methods, while achieving comparatively small training and testing times.

---

P. Piro (✉)

Istituto Italiano di Tecnologia, Via Morego 30, 16163 Genova, Italy

e-mail: [paolo.piro@iit.it](mailto:paolo.piro@iit.it)

R. Nock

CEREGMIA, Université Antilles-Guyane, Campus de Schoelcher, Martinique, France  
e-mail: [rnock@martinique.univ-ag.fr](mailto:rnock@martinique.univ-ag.fr)

W. Bel Haj Ali · M. Barlaud

I3S Laboratory, University of Nice-Sophia Antipolis, 06903 Sophia Antipolis, France

W. Bel Haj Ali

e-mail: [belhajal@i3s.unice.fr](mailto:belhajal@i3s.unice.fr)

M. Barlaud

e-mail: [barlaud@i3s.unice.fr](mailto:barlaud@i3s.unice.fr)

F. Nielsen

Department of Fundamental Research, Sony Computer Science Laboratories, Inc., Tokyo, Japan

F. Nielsen

LIX Department, Ecole Polytechnique, Palaiseau, France

e-mail: [nielsen@lix.polytechnique.fr](mailto:nielsen@lix.polytechnique.fr)

## 12.1 Introduction

In this chapter, we describe the proposed approach to  $k$ -NN boosting. First, we introduce the scope of our work, which aims at automatic visual categorization of scenes (Sect. 12.1.1) and relies on prototype-based classification (Sect. 12.1.2). Then, in Sects. 12.2.1–12.2.3 we present the key definitions for surrogate risk minimization. Our UNN algorithm is detailed in Sect. 12.2.4 for the case of the exponential risk. Section 12.2.5 presents the generic convergence theorem of UNN and the upper bound performance for the exponential risk minimization. Then, in Sect. 12.3, we report our experiments on simulated and real data, comparing UNN with  $k$ -NN, support vector machines (SVM) and AdaBoost, using Gist and/or Bag-of-Feature descriptors. Real datasets include those proposed in [10, 30, 43], with a number of categories ranging from 8 to 60. Then, in Sects. 12.4 and 12.5 we discuss results, mention future works, and conclude. Finally, we postpone the general form and analysis of UNN to other surrogate risks to the Appendix.

### 12.1.1 Visual Categorization

In this work, we address the problem of generic visual categorization. This is a relevant task in computer vision, which aims at automatically classifying images into a discrete set of categories, such as *indoor* vs *outdoor* [15, 32], *beaches* vs *mountains*, *churches* vs *towers*. Generic categorization is distinct from object and scene recognition, which are classification tasks concerning particular instances of objects or scenes (e.g., *Notre Dame Cathedral* vs *St. Peter’s Basilic*). It is also distinct from other related computer vision tasks, such as content-based image retrieval (that aims at finding images from a database, which are semantically related or visually similar to a given query image) and object detection (which requires to find both the presence and the position of a target object in an image, e.g., person detection).

Automatic categorization of generic scenes is still a challenging task, due to the huge number of natural categories that should be considered in general. In addition, natural image categories may exhibit high inter-class variability (i.e., visually different images may belong to the same category) and low inter-class variability (i.e., distinct categories may contain visually similar images). Classifying images requires an effective and reliable description of the image content, for example, location and shape of specific objects or overall scene appearance. Although several approaches have been proposed in the recent literature to extract semantic information from images [36, 42], most of the state-of-the-art techniques for image categorization still rely on low-level visual information extracted by means of image analysis operators and coded into vector descriptors.

Examples of suitable low-level image descriptors for categorization purposes are Gist, that is, global image features representing the overall scene [30], and SIFT descriptors, that is, descriptors of local features extracted either at salient patches [24] or at dense grid points [23]. A Gist descriptor is based on the so-called “spatial

envelope” [30], which is a very effective low dimensional representation of the overall scene based on spectral information. Such a representation bypasses segmentation, extraction of key-points and processing of individual objects and regions, thus enabling a compact global description of images. Gist descriptors have been successfully used for categorizing locations and environments, showing their ability to provide relevant priors for more specific tasks, like object recognition and detection [40]. Another successful tool for describing the global content of a scene is the Bag-of-Features scheme [38], which represents an image by the histogram of occurrences of vector quantized local descriptors like SIFT.

### 12.1.2 $k$ -NN Classification

Apart from the descriptors used to compactly represent images, most image categorization methods rely on supervised learning techniques for exploiting information about known samples when classifying an unlabeled sample. Among these techniques,  $k$ -NN classification has proven successful, thanks to its easy implementation and its good generalization properties [37]. A generalization of the  $k$ -NN rule to the multi-label classification framework has been also proposed recently by [46], whose technique is based on the maximum-a-posteriori principle applied to multi-labeled  $k$ -NN. A major advantage of the  $k$ -NN rule is not to require explicit construction of the feature space and be naturally adapted to multi-class problems. Moreover, from the theoretical point of view, straightforward bounds are known for the true risk (i.e., error) of  $k$ -NN classification with respect to the Bayes optimum, even for finite samples [29].

Although such advantages make  $k$ -NN classification very attractive to practitioners, it is an algorithmic challenge to speed-up  $k$ -NN queries. It is also a statistical challenge to further improve the risk bounds of  $k$ -NN. In part due to the simplicity of the classification rule, many methods have been proposed to address either of these challenges. For example, many methods have been proposed for speeding up nearest neighbor retrieval, including locality sensitive hashing (LSH, [13]), product quantization for nearest neighbor search [21], and vector space embedding with boosting algorithms [2, 25].

It is yet another challenge to reduce the true risk of the  $k$ -NN rule, usually tackled by data reduction techniques [17]. In prior work, the classification problem has been reduced to tracking ill-defined categories of neighbors, interpreted as “noisy” [6]. Most of these recent techniques are in fact partial solutions to a larger problem related to the nearest neighbors’ true risk, which does not have to be the discrete prediction of labels, but rather a continuous estimation of class membership probabilities [19]. This problem has been reformulated by [7] as a strong advocacy for the formal transposition of *boosting* to nearest neighbors classification. Such a formalization is challenging as nearest neighbors rules are indeed not *induced*, whereas all formal boosting algorithms induce so-called *strong* classifiers by combining *weak* classifiers—also induced, such as decision trees—[35].

A survey of the literature shows that at least four different categories of approaches have been proposed in order to improve  $k$ -NN classification:

- learning local or global adaptive distance metric;
- embedding data in the feature space (kernel nearest neighbors);
- distance-weighted and difference-weighted nearest neighbors;
- boosting nearest neighbors.

The earliest approaches to generalizing the  $k$ -NN classification rule relied on learning an adaptive distance metric from training data (see the seminal works of [11]). An analogous approach was later adopted by [18], who carried out linear discriminant analysis to adaptively deform the distance metric. Recently, [31] has proposed a method for learning a weighted distance, where weights can be either global (i.e., only depending on classes and features) or local (i.e., depending on each individual prototype as well).

Other more recent techniques apply the  $k$ -NN rule to data embedded in a high-dimensional feature space, following the kernel trick approach of support vector machines. For example, [44] have proposed a straightforward adaptation of the kernel mapping to the nearest neighbors rule, which yields significant improvement in terms of classification accuracy. In the context of vision, a successful technique has been proposed by [47], which involves a “refinement” step at classification time, without relying on explicitly learning the distance metric. This method trains a local support vector machine on nearest neighbors of a given query, thus limiting the most expensive computations to a reduced subset of prototypes.

Another class of  $k$ -NN methods relies on weighting nearest neighbors votes based on their distances to the query sample [8]. Recently, [49] have proposed a similar weighting approach, where the nearest neighbors are weighted based on their vector difference to the query. Such a difference-weight assignment is defined as a constrained optimization problem of sample reconstruction from its neighborhood. The same authors have proposed a kernel-based non-linear version of this algorithm as well.

Finally, comparatively few work have proposed the use of boosting techniques for  $k$ -NN classification [1, 2, 12, 25, 33]. [1] use AdaBoost for learning a distance function to be used for  $k$ -NN search. [12] adopt the boosting approach in a non-conventional way. At each iteration a different  $k$ -NN classifier is trained over a modified input space. Namely, the authors propose two variants of the method, depending on the way the input space is modified. Their first algorithm is based on optimal subspace selection, that is, at each boosting iteration the most relevant subset of input data is computed. The second algorithm relies on modifying the input space by means of non-linear projections. But neither method is strictly an algorithm for inducing weak classifiers from the  $k$ -NN rule, thus not directly addressing the problem of boosting  $k$ -NN classifiers. Moreover, such approaches are computationally expensive, as they rely on a genetic algorithm and a neural network, respectively. [2, 25] map examples in a vector space by using the outputs of (Ada)boosted weak classifiers. It is not known whether these algorithms formally keep (or improve) the boosting properties known for AdaBoost [35]. More recently, [33] have built

upon the works of [27, 28] (see also the survey of the approach in [9]) to provide a provable boosting algorithm for  $k$ -NN classifiers. Guaranteed convergence speed is obtained for AdaBoost’s famed exponential loss, under a weak index assumption which parallels the weak learning assumption of boosting algorithms, making the approach of [33] among the first to provide a provable boosting algorithm for  $k$ -NN [7].

We propose in this work a full-fledged solution to the problem of boosting  $k$ -NN classifiers in the general multi-class setting and for general classes of losses. Namely, we propose the first provable boosting algorithm, called UNN, which *induces a leveraged* nearest neighbor rule that generalizes the uniform  $k$ -NN rule, and whose convergence rate is guaranteed for a wide (i.e., infinite) set of losses, encompassing popular choices such as the logistic loss or the squared loss. The voting rule is redefined as a strong classifier that linearly combines weak classifiers of the  $k$ -NN rule (i.e., the examples). Therefore, our approach does not need to learn a distance function, as it directly operates on the top of  $k$ -NN search. At the same time, it does not require an explicit computation of the feature space, thus preserving one of the main advantages of prototype-based methods. Our boosting algorithm is an iterative procedure which learns the weights for examples called *leveraging coefficients*. Then, our class encoding allows to generalize the guarantees on convergence rates for an *infinite* number of *surrogate risks*.<sup>1</sup> The generalization is highly desirable, not only for experimental purposes related *for example*, to no-free-lunch Theorems [28]: our generalization encompasses many *classification calibrated* surrogates, functions exhibiting particularly convenient guarantees in the context of classification [4]. Finally, an important characteristic of UNN is that it is naturally able, through the leveraging mechanism, to discriminate the most relevant prototypes for a given class.

## 12.2 Method

### 12.2.1 Preliminary Definitions

In this work, we address the task of *multi-class, single-label* image categorization. Although the multi-label framework is quite well established in literature [5], we only consider the case where each image is constrained to belong to one single category among a set of predefined categories. The number of categories (or classes) may range from a few to hundreds, depending on applications. For example, categorization with 67 indoor categories has been recently studied by [34]. We treat the multi-class problem as multiple binary classification problems as it is customary in machine learning. Hence, for each class  $c$ , a query image is classified either

---

<sup>1</sup>A *surrogate* is a function which is a suitable upperbound for another function (here, the non-convex non-differentiable empirical risk).

to  $c$  or to  $\bar{c}$  (the complement class of  $c$ , which contains all classes but  $c$ ) with a certain confidence (*classification score*). Then the label with the maximum score is assigned to the query. Images are represented by descriptors related to given local or global features. We refer to an image descriptor as an *observation*  $\mathbf{x} \in \mathcal{X}$ , which is a vector of  $n$  features and belongs to a *domain*  $\mathcal{X}$  (e.g.,  $\mathbb{R}^n$  or  $[0, 1]^n$ ). A label is associated to each image descriptor according to a predefined set of  $C$  classes. Hence, an observation with the corresponding label leads to an *example*, which is the ordered pair  $(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathbb{R}^C$ , where  $\mathbf{y}$  is termed the *class vector* that specifies the class memberships of  $\mathbf{x}$ . In particular, the sign of  $y_c$  gives the membership of example  $(\mathbf{x}, \mathbf{y})$  to class  $c$ , such that  $y_c$  is negative iff the observation does not belong to class  $c$ , positive otherwise. At the same time, the absolute value of  $y_c$  may be interpreted as a relative confidence in the membership. Inspired by the multi-class boosting analysis of [48], we constrain class vectors to be *symmetric*, that is:

$$\sum_{c=1}^C y_c = 0. \quad (12.1)$$

Hence, in the single-label framework, the class vector of an observation  $\mathbf{x}$  belonging to class  $\bar{c}$  is defined as:

$$y_{\bar{c}} = 1, \quad y_{c \neq \bar{c}} = -\frac{1}{C-1}. \quad (12.2)$$

This setting turns out to be necessary when treating multi-class classification as multiple binary classifications, as it balances negative and positive labels of a given example over all classes. In the following, we deal with an input set of  $m$  examples (or prototypes)  $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{y}_i), i = 1, 2, \dots, m\}$ , arising from annotated images, which form the *training set*.

### 12.2.2 Surrogate Risks Minimization

We aim at defining a one-versus-all classifier for each category, which is to be trained over the set of examples. This classifier is expected to correctly classify as many new observations as possible, that is, to predict their true labels. Therefore, we aim at determining a classification rule  $\mathbf{h}$  from the training set, which is able to minimize the classification error over all possible new observations. Since the underlying class probability densities are generally unknown and difficult to estimate, defining a classifier in the framework of supervised learning can be viewed as fitting a classification rule onto a training set  $\mathcal{S}$ , with the hope to minimize overfitting as well. In the most basic framework of supervised classification, one wishes to train a *classifier* on  $\mathcal{S}$ , that is, build a function  $\mathbf{h} : \mathcal{X} \rightarrow \mathbb{R}^C$  with the objective to minimize its *empirical risk* on  $\mathcal{S}$ , defined as:

$$\varepsilon^{0/1}(\mathbf{h}, \mathcal{S}) \doteq \frac{1}{mC} \sum_{c=1}^C \sum_{i=1}^m [\varrho(\mathbf{h}, i, c) < 0], \quad (12.3)$$

with  $[.]$  the indicator function (1 iff true, 0 otherwise), called here the *0/1 loss*, and:

$$\varrho(\mathbf{h}, i, c) \doteq y_{ic} h_c(\mathbf{x}_i) \quad (12.4)$$

the *edge* of classifier  $\mathbf{h}$  on example  $(\mathbf{x}_i, y_i)$  for class  $c$ . Taking the sign of  $h_c$  in  $\{-1, +1\}$  as its membership prediction for class  $c$ , one sees that when the edge is positive (resp. negative), the membership predicted by the classifier and the actual example's membership agree (resp. disagree). Therefore, (12.3) averages over all classes the number of mismatches for the membership predictions, thus measuring the goodness-of-fit of the classification rule on the training dataset. Provided the example dataset has good generalization properties with respect to the unknown distribution of possible observations, minimizing this empirical risk is expected to yield good accuracy when classifying unlabeled observations.

However, minimizing the empirical risk is computationally not tractable as it deals with non-convex optimization. In order to bypass this cumbersome optimization challenge, the current trend of supervised learning (including boosting and support vector machines) has replaced the minimization of the empirical risk (12.3) by that of a so-called *surrogate risk* [4], to make the optimization problem amenable. In boosting, it amounts to sum (or average) over classes and examples a real-valued function called the *surrogate loss*, thus ending up with the following rewriting of (12.3):

$$\varepsilon^\psi(\mathbf{h}, \mathcal{S}) \doteq \frac{1}{mC} \sum_{c=1}^C \sum_{i=1}^m \psi(\varrho(\mathbf{h}, i, c)). \quad (12.5)$$

Relevant choices available for  $\psi$  include:

$$\psi^{\text{sqr}} \doteq (1 - x)^2, \quad (12.6)$$

$$\psi^{\text{exp}} \doteq \exp(-x), \quad (12.7)$$

$$\psi^{\text{log}} \doteq \log(1 + \exp(-x)); \quad (12.8)$$

(12.6) is the squared loss [4], (12.7) is the exponential loss [35], and (12.8) is the logistic loss [4]. Such surrogates play a fundamental role in supervised learning. They are upper bounds of the empirical risk with desirable convexity properties. Their minimization remarkably impacts on that of the empirical risk, thus enabling to provide minimization algorithms with good generalization properties [28].

In the following, we move from recent advances in boosting with surrogate risks to redefine the  $k$ -NN classification rule. Our algorithm, UNN (Universal Nearest Neighbors), is first proposed for the exponential surrogate. We describe in the appendix the general formulation of the algorithm, not restricted to this surrogate. We show that UNN converges to the optimum of many surrogates with guaranteed convergence rates under mild assumptions, and more generally converges to the global optimum of the surrogate risk for an even wider set of surrogates.

### 12.2.3 Leveraging the $k$ -NN Rule

We denote by  $\text{NN}_k(\mathbf{x})$  the set of the  $k$ -nearest neighbors (with integer constant  $k > 0$ ) of an example  $(\mathbf{x}, \mathbf{y})$  in set  $\mathcal{S}$  with respect to a non-negative real-valued “distance” function. This function is defined on domain  $\mathcal{X}$  and measures how much two observations differ from each other. This dissimilarity function thus may not necessarily satisfy the triangle inequality of metrics. For sake of readability, we let  $i \sim_k \mathbf{x}$  denote an example  $(\mathbf{x}_i, \mathbf{y}_i)$  that belongs to  $\text{NN}_k(\mathbf{x})$ . This neighborhood relationship is intrinsically asymmetric, that is,  $\mathbf{x}_i \in \text{NN}_k(\mathbf{x})$  does not necessarily imply that  $\mathbf{x} \in \text{NN}_k(\mathbf{x}_i)$ . Indeed, a nearest neighbor of  $\mathbf{x}$  does not necessarily contain  $\mathbf{x}$  among its own nearest neighbors.

The  $k$ -nearest neighbors rule ( $k$ -NN) is the following multi-class classifier  $\mathbf{h} = \{h_c : c = 1, 2, \dots, C\}$  ( $k$  appears in the summation indices):

$$h_c(\mathbf{x}) = \sum_{j \sim_k \mathbf{x}} [y_{jc} > 0], \quad (12.9)$$

where  $h_c$  is the one-versus-all classifier for class  $c$  and square brackets denote the indicator function. Hence, the classic nearest neighbors classification is based on majority vote among the  $k$  closest prototypes.

We propose to weight the votes of nearest neighbors by means of real coefficients, thus generalizing (12.9) to the following *leveraged*  $k$ -NN rule  $\mathbf{h}^\ell = \{h_c^\ell : c = 1, 2, \dots, C\}$ :

$$h_c^\ell(\mathbf{x}) = \sum_{j \sim_k \mathbf{x}} \alpha_{jc} y_{jc}, \quad (12.10)$$

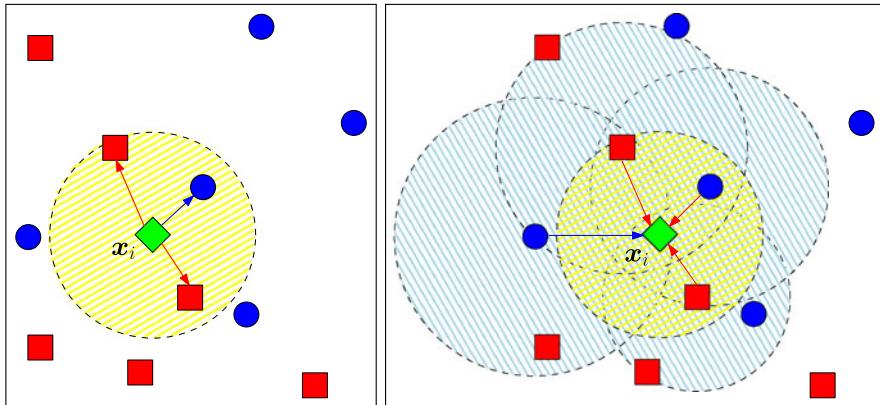
where  $\alpha_{jc} \in \mathbb{R}$  is the leveraging coefficient for example  $j$  in class  $c$ , with  $j = 1, 2, \dots, m$  and  $c = 1, 2, \dots, C$ . Hence, (12.10) linearly combines class labels of the  $k$  nearest neighbors (defined in Sect. 12.2.1) with their leveraging coefficients.

Our work is focused on formal boosting algorithms working on top of the  $k$ -NN methods. These algorithms do not affect the nearest neighbor search when inducing weak classifiers of (12.10). They are thus independent on the way nearest neighbors are computed, unlike most of the approaches mentioned in Sect. 12.1.2, which rely on modifying the neighborhood relationship via metric distance deformations or kernel transformations. This makes our approach fully compatible with any underlying (metric) distance and data structure for  $k$ -NN search, as well as possible kernel transformations of the input space.

For a given training set  $\mathcal{S}$  of  $m$  labeled examples, we define the  $k$ -NN *edge matrix*  $\mathbf{R}^{(c)} \in \mathbb{R}^{m \times m}$  for each class  $c = 1, 2, \dots, C$ :

$$\mathbf{r}_{ij}^{(c)} \doteq \begin{cases} y_{ic} y_{jc} & \text{if } j \sim_k i \\ 0 & \text{otherwise.} \end{cases} \quad (12.11)$$

The name of  $\mathbf{R}^{(c)}$  is justified by an immediate parallel with (12.4). Indeed, each example  $j$  serves as a classifier for each example  $i$ , predicting 0 if  $j \notin \text{NN}_k(\mathbf{x}_i)$ ,



**Fig. 12.1** Schematic illustration of the *direct* (left) and *reciprocal* (right)  $k$ -nearest neighbors ( $k = 1$ ) of an example  $x_i$  (green diamond). Red squares and blue circles represent examples of positive and negative classes. Each arrow connects an example to its  $k$ -nearest neighbors

$y_{jc}$  otherwise, for the membership to class  $c$ . Hence, the  $j$ th column of matrix  $\mathbf{R}^{(c)}$ ,  $\mathbf{r}_j^{(c)}$ , which is different from  $\mathbf{x}$  when choosing  $k > 0$ , collects all edges of “classifier”  $j$  for class  $c$ . Note that nonzero entries of this column correspond to the so-called *reciprocal nearest neighbors* of  $j$ , that is, those examples for which  $j$  is a neighbor (Fig. 12.1). Eventually, the edge of the leveraged  $k$ -NN rule on example  $i$  for class  $c$  reads:

$$\varrho(\mathbf{h}^\ell, i, c) = (\mathbf{R}^{(c)} \boldsymbol{\alpha}^{(c)})_i, \quad c = 1, 2, \dots, C, \quad (12.12)$$

where  $\boldsymbol{\alpha}^{(c)}$  collects all leveraging coefficients in a vector form for class  $c$ :  $\alpha_i^{(c)} = \alpha_{ic}$ ,  $i = 1, 2, \dots, m$ . Thus, the induction of the leveraged  $k$ -NN classifier  $\mathbf{h}^\ell$  amounts to fitting all  $\boldsymbol{\alpha}^{(c)}$ ’s so as to minimize (12.5), after replacing the argument of  $\psi(\cdot)$  in (12.5) by (12.12).

#### 12.2.4 UNN Boosting Algorithm

We explain our classification algorithm specialized for the exponential loss minimization in the multi-class one-versus-all framework, with pseudo-code shown in Algorithm 1. Like common boosting algorithms, UNN operates on a set of weights  $w_i$  ( $i = 1, 2, \dots, m$ ) defined over training data. Such weights are repeatedly updated to fit all leveraging coefficients  $\boldsymbol{\alpha}^{(c)}$  for class  $c$  ( $c = 1, 2, \dots, C$ ). At each iteration, the index to leverage,  $j \in \{1, 2, \dots, m\}$ , is obtained by a call to a *weak index chooser* oracle  $WIC(\cdot, \cdot, \cdot)$ , whose implementation is detailed later in this section.

Figure 12.2 presents a block diagram of UNN algorithm. In particular, notice how the initialization step, relying on  $k$ -NN and edge matrix computation, is clearly

**Algorithm 1** Universal Nearest Neighbors UNN( $\mathcal{S}$ ) for  $\psi = \psi^{\text{exp}}$ 


---

**Input:**  $\mathcal{S} = \{(\mathbf{x}_i, y_i), i = 1, 2, \dots, m, \mathbf{x}_i \in \mathcal{X}, y_i \in \{-\frac{1}{C-1}, 1\}^C\}$

$$\mathbf{r}_{ij}^{(c)} = \begin{cases} y_{ic} y_{jc} & \text{if } j \sim_k \mathbf{x}_i \\ 0 & \text{otherwise} \end{cases} \quad \forall i, j = 1, 2, \dots, m, c = 1, 2, \dots, C \quad \triangleright k\text{-NN edge matrix}$$

**for**  $c = 1, 2, \dots, C$  **do**

- $\alpha_{jc} \leftarrow 0, \quad \forall j = 1, 2, \dots, m \quad \triangleright$  Leveraging coefficients
- $w_i \leftarrow 1, \quad \forall i = 1, 2, \dots, m \quad \triangleright$  Example weights

**for**  $t = 1, 2, \dots, T$  **do**

- [I.1]  $j \leftarrow \text{WIC}(\{1, 2, \dots, m\}, t) \quad \triangleright$  Weak index chooser oracle
- [I.2]

$$w_j^+ = \sum_{i: \mathbf{r}_{ij}^{(c)} > 0} w_i, \quad w_j^- = \sum_{i: \mathbf{r}_{ij}^{(c)} < 0} w_i \quad (12.13)$$

$$\delta_j \leftarrow \frac{1}{2} \log \left( \frac{w_j^+}{w_j^-} \right) \quad (12.14)$$

[I.3]

$$w_i \leftarrow w_i \exp(-\delta_j \mathbf{r}_{ij}^{(c)}), \quad \forall i : j \sim_k \mathbf{x}_i \quad (12.15)$$

[I.4]  $\alpha_{jc} \leftarrow \alpha_{jc} + \delta_j$

end for

end for

**Output:**  $h_c(\mathbf{x}) = \sum_{i \sim_k \mathbf{x}} \alpha_{ic} y_{ic}, \quad \forall c = 1, 2, \dots, C$

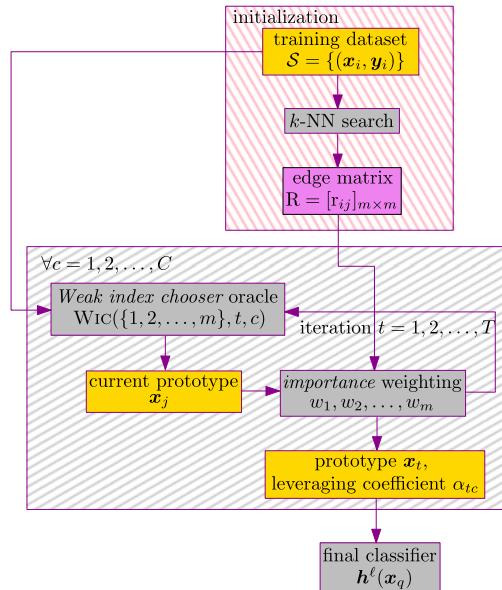
---

distinguished from the iterative procedure, where a new prototype is added at each iteration  $t$ , thus updating both the strong classifier  $\mathbf{h}(\mathbf{x})$  and the weights  $w_i$ .

The training phase is implemented in a one-versus-all fashion, that is,  $C$  learning problems are solved independently, and for each class  $c$  the training examples are considered as belonging to either class  $c$  or the complement class  $\bar{c}$ , that is, *any other class*. Eventually, one leveraging coefficient ( $\alpha_{jc}$ ) per class is learned for each weak classifier (indexed by  $j$ ).

The key observation when training weak classifiers with UNN is that, at each iteration, one single example (indexed by  $j$ ) is considered as a prototype to be leveraged. Indeed, all the other training data are to be viewed as observations for which  $j$  may possibly vote. In particular, due to  $k$ -NN voting,  $j$  can be a classifier only for its reciprocal nearest neighbors (i.e., those data for which  $j$  itself is a neighbor, corresponding to nonzero entries in matrix (12.11) on column  $j$ ). This brings to a remarkable simplification when computing  $\delta_j$  in step [I.2] and updating weights  $w_i$  in step [I.3] (Eqs. (12.14), (12.15)). Indeed, only weights of reciprocal nearest neighbors of  $j$  are involved in these computations, thus allowing us not to store the entire matrix  $\mathbf{R}^{(c)}$ ,  $c = 1, 2, \dots, C$ . Note that the set of reciprocal neighbors is split in two subsets, each containing examples that agree (disagree) with the class membership of  $j$ , thus yielding the partial sums  $w_j^+$  and  $w_j^-$  of (12.13).

**Fig. 12.2** Block diagram of the UNN learning scheme



Note that when whichever  $w_j^+$  or  $w_j^-$  is zero,  $\delta_j$  in (12.14) is not finite. There is however a simple alternative, inspired by [35], which consists in smoothing out  $\delta_j$  when necessary, thus guaranteeing its finiteness without impairing convergence. More precisely, we suggest to replace:

$$w_j^+ \leftarrow w_j^+ + \frac{1}{m}, \quad (12.16)$$

$$w_j^- \leftarrow w_j^- + \frac{1}{m}. \quad (12.17)$$

Also note that step [I.1] relies on oracle  $\text{WIC}(\cdot, \cdot, \cdot)$  for selecting index  $j$  of the next weak classifier. We propose two alternative implementations of this oracle, as follows:

- (a) a *lazy* approach:  $T = m$ ,  $\text{WIC}(\{1, 2, \dots, m\}, t, c) \doteq t$ ;
- (b) the *boosting* approach: we pick  $T \geq m$ , and let  $j$  be chosen by  $\text{WIC}(\{1, 2, \dots, m\}, t, c)$  such that  $\delta_j$  is large enough. Each  $j$  can be chosen more than once.

There are also schemes *mixing* (a) and (b): for example, we may pick  $T = m$ , choose  $j$  as in (b), but exactly once as in (a).

### 12.2.5 UNN Convergence

The main properties of UNN are summarized by the following three fundamental theorems. The first theorem ensures general *monotonic convergence* to the optimal

surrogate loss, for *any* given surrogate function. The second theorem further refines this general convergence theorem by providing an effective convergence bound for the exponential loss.

Suppose that  $\psi$  meets the following conditions:

- (i)  $\text{im}(\psi) = \mathbb{R}_+$ ;
- (ii)  $\nabla_\psi(0) < 0$  ( $\nabla_\psi$  is the conventional derivative);
- (iii)  $\psi$  is strictly convex and differentiable.

**Theorem 12.1** *As the number of iteration steps  $T$  increases, UNN converges to  $\mathbf{h}^\ell$  realizing the global minimum of the surrogate risk at hand (12.5), for any  $\psi$  meeting conditions (i), (ii) and (iii) above.*

*Proof* A proofsketch is given in [Appendix](#). □

Then, in order to obtain the specific convergence rate for  $\psi^{\exp}$ , suppose the following *weak index assumption* (WIA) holds. (See Eq. (12.13) in Algorithm 1 for the definition of  $w_j^{(c)+}$  and  $w_j^{(c)-}$ .)

(WIA) There exist some  $\gamma > 0$  and  $\eta > 0$  such that the following two inequalities hold for index  $j$  returned by  $\text{WIC}(., ., .)$ :

$$\left| \frac{w_j^{(c)+}}{w_j^{(c)+} + w_j^{(c)-}} - \frac{1}{2} \right| \geq \gamma, \quad (12.18)$$

$$\frac{w_j^{(c)+} + w_j^{(c)-}}{\|\mathbf{w}\|_1} \geq \eta. \quad (12.19)$$

**Theorem 12.2** *If the (WIA) holds for  $v \leq T$  steps in UNN (for each  $c$ ), then  $\varepsilon^{0/1}(\mathbf{h}^\ell, \mathcal{S}) \leq \exp(-\Omega(\eta\gamma^2 v))$ .*

*Proof* A proofsketch is given in [Appendix](#). □

Theorems 12.1 and 12.2 show that UNN converges (exponentially fast) to the global optimum of the surrogate risk on the training set. Most of the recent works that can be associated to boosting algorithms, or more generally to the minimization of some surrogate risk using whichever kind of procedure, have explored the universal consistency of the surrogate minimization problems (see [4, 26, 45], and references therein). The problem can be roughly stated as whether the minimization of the surrogate risk guarantees in probability for the classifier built to converge to the Bayes rule as  $m \rightarrow \infty$ . This question obviously becomes relevant to UNN given our results. Among the results contained in this rich literature, the one whose consequences directly impact on the universal consistency of UNN is Theorem 3 of [4]. We can indeed easily show that all our choices of surrogate loss are classification calibrated, so that minimizing the surrogate risk in the limit ( $m \rightarrow \infty$ ) implies minimizing the true risk, and implies uniform consistency as well. Moreover, this result,

proven for  $C = 2$ , holds as well for arbitrary  $C \geq 2$  in the single-label prediction problem. [3] proved an additional result for AdaBoost [35]: if the algorithm is run for a number  $T \geq m^\eta$  boosting rounds, for  $\eta \in (0, 1)$ , then there is indeed minimization in the limit of the exponential risk, and so AdaBoost is universally consistent. From our theorems above, this implies the consistency of UNN, and this even has the consequence to prove that the filtering procedure described in the experiments is also consistent, since indeed [3]’s bound implies that we leverage a proportion of  $1/m^{1-\eta}$  examples, “filtering out” the remaining ones.

Moreover, the results of [26] are also interesting in our setting, even when they are typically aimed at boosting algorithms with weak learners like decision-tree learning algorithms, that define *quantizations* of the observations (each decision tree defines a new description variable for the examples). They show that there exists conditions on the quantizers that yield conditions on the surrogate loss function for universal consistency. It is interesting to notice that the universal consistency of UNN does not need such assumptions, as weak learners are examples that do not make quantizations of the observation’s domain. Finally, the work of [45] explores the consistency of surrogate risk minimization in the case where rejects are allowed by classifiers, somehow refusing to classify an observation at a cost smaller than misclassifying. While this setting is not relevant to UNN in the general case, it becomes relevant as we filter out examples (see the experiments), which boils down to stating that they systematically reject on observations.

On the one hand, [45] show that filtering out examples does not impair UNN universal consistency, as long as filter thresholds are locally based. On the other hand, they also provide a way to quantify the actual loss  $\ell_{r,j}$  caused by filtering out example  $j$ , which we recall is in between 0 (the loss of good classification) and 1 (the loss of bad classification). For example, choosing the exponential loss and using Theorem 1 in [45] reveals that the reject loss is:

$$\ell_{r,j} = \frac{\min\{w_j^+, w_j^-\}}{w_j^+ + w_j^-}.$$

Let us now complete further the picture of boosting algorithms for  $k$ -NN, by showing that, under a mild additional assumption on  $\psi$ , we obtain a guaranteed convergence rate for UNN. Of particular interest is the assumption under which we are able to prove this result. Following [27, 28], we make a “Weak Edge Assumption”:

- (WEA) There exists some  $\vartheta > 0$  such that the following inequality holds for index  $j$  returned by  $\text{WIC}(\cdot, \cdot, \cdot)$ :

$$\left| \sum_{i:j \sim_k i} r_{ij}^{(c)} w_i \right| \geq \vartheta. \quad (12.20)$$

This assumption states that the average value (in absolute value) of  $y_{ic}y_{jc}$  over the reciprocal neighborhood of example  $j$  cannot be smaller than some constant  $\vartheta$ . It is *weak* for the following reason. If the classes in the reciprocal neighborhood were

picked at random, the quantity inside the absolute value in (12.20) would be zero in average because of the way we model classes in (12.1). So, we are assuming that, regardless of weights, we can always pick an example  $(\mathbf{x}_j, y_j)$  “beating” random by a potentially small advantage  $\vartheta$ . Note that (WEA) is weaker than (WIA) in the sense that we do not make any coverage assumption like (12.19).

Let us now turn to the assumption on  $\psi$ :

(iv)  $\psi$  is locally  $\omega$  strongly smooth, for some  $\omega > 0$ :

$$D_\psi(x' \| x) \leq \frac{\omega}{2}(x' - x)^2, \quad (12.21)$$

where  $x, x'$  range through the values  $\varrho(\mathbf{h}, i, c)$  over which UNN is run, and

$$D_\psi(x' \| x) \doteq \psi(x') - \psi(x) - (x' - x)\nabla_\psi(x) \quad (12.22)$$

is the Bregman divergence with generator  $\psi$ . There is an important duality between strong smoothness and strong convexity, with applications in machine learning and optimization [22]. The proof of the following theorem, in the [Appendix](#), is another example of its applicability in these fields.

**Theorem 12.3** *If the (WEA) holds and  $\psi$  meets assumptions (i)–(iv), then for any user-fixed  $\tau \in [0, 1]$ , UNN has fit a leveraged  $k$ -NN classifier with empirical risk no greater than  $\tau$  provided the number of boosting iterations  $T$  satisfies:*

$$T \geq \frac{2(1 - \tau)\psi(0)\omega km}{\vartheta^2(C - 1)} = \Omega\left(\frac{\omega km}{\vartheta^2}\right). \quad (12.23)$$

Theorem 12.3 does not obliterate the (better) convergence results for the exponential loss of Theorem 12.2, yet it opens the guarantees of convergence under weak assumptions to some of the most interesting surrogates in classification. These include *permissible convex surrogates* (PCS, [27]), a set containing as special cases the squared and logistic surrogates in (12.6), (12.8). Informally, any loss which meets regularity conditions and common requirements about losses, such as lower-boundedness, symmetry and the proper scoring property, can be represented by a PCS [27]. The exponential surrogate in (12.7) is not a PCS, yet it is a first-order approximation to the logistic surrogate. Up to translating and scaling by constants, any PCS meets  $\text{im}(\nabla_\psi) \subseteq [-1, 0]$  [27]. Reasoning on the second derivative of  $\psi$ , we see that there is not much room to violate (12.21), thus making many PCS  $\omega$  strongly smooth for small values of  $\omega$ . Simple calculations yield that we can take for example  $\omega = 1/4$  for the logistic loss (12.8), and  $\omega = 2$  for the squared loss (12.6), making the bound in (12.23) more favorable to the former. As a last example, consider the following parameterized choice for  $\psi$ , with  $\mu \in (0, 1)$ :

$$\psi_\mu^{\text{mat}} \doteq \frac{1}{1 - \mu}(-x + \sqrt{(1 - \mu)^2 + x^2});$$

this choice, which gives rise to Matsushita’s loss for  $\mu = 0$ , has important convexity properties [27]. In this case, we easily obtain that we can pick  $\omega = 1/(1 - \mu)$ .

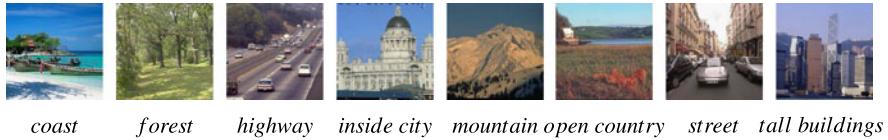
## 12.3 Experiments

In this section, we present experimental results of UNN for image categorization. In order to reduce numerical problems on the large databases on which we test UNN, we normalize weights to unity after the update in (12.15). Our experiments aim at carefully quantifying and explaining the gains brought by boosting on  $k$ -NN voting on real image databases. In particular, we propose in Sects. 12.3.1 and 12.3.2 an analysis and comparison of UNN vs  $k$ -NN for Gist and Bag-of-Features descriptors on two broadly used datasets of natural images. In Sect. 12.3.3, we drill down into precision and execution times comparisons between UNN vs  $k$ -NN, SVM and AdaBoost. We also introduce in this section a soft version of UNN which, to classify new observations, convolutes weighting with a simple density estimation suggested by boosting.

### 12.3.1 Image Categorization Using Global Gist Descriptors

We tested UNN on global descriptors for the categorization of natural images. In particular, we used the database of natural scenes collected by [30], which has been successfully used to validate several classification techniques relying on Gist image descriptors. A Gist descriptor provides a global representation of a scene directly, without requiring neither an explicit segmentation of image regions and objects nor an intermediate representation by means of local features. In the standard setting, an image is first resized to square, then represented by a single vector of  $d$  components (typically  $d = 512$  or  $d = 320$ ), which collects features related to the spatial organization of dominant scales and orientations in the image. The one-to-one mapping between images and Gist descriptors is one of the main advantages of using such a global representation instead of local descriptors. In particular, the ability to map any instance to a single point in the feature space is crucial for the effectiveness of  $k$ -NN methods, where computing the one-to-one similarity between testing and training instances is explicitly required at classification time. Conversely, representing an image with a set of multiple local descriptors is not directly adapted to such discriminative classification techniques, thus generally requiring an intermediate (usually unsupervised) learning step in order to extract a compact single-vector descriptor from the set of local descriptors [14]. For example, this is the case for Bag-of-Features methods, that we discuss in Sect. 12.3.2 along with an experimental comparison to our method. Finally, although Gist is not an alternative image representation method with respect to local descriptors, it has proven very successful in representing relevant contextual information of natural scenes, thus allowing, for instance, to compute meaningful priors for exploration tasks, like object detection and localization [40].

In the following, we denote as 8-cat the database of [30], which contains 2,688 color images of outdoor scenes of size  $256 \times 256$  pixels, divided in 8 categories: *coast*, *mountain*, *forest*, *open country*, *street*, *inside city*, *tall buildings* and *highways*. One example image of each category is shown in Fig. 12.3. In addition, we

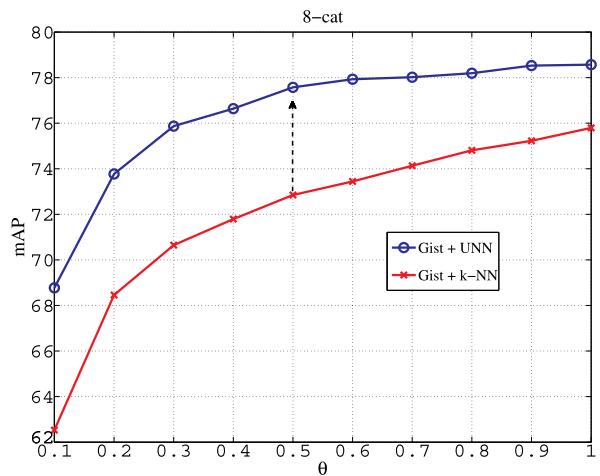


**Fig. 12.3** Examples of annotated images from the 8 categories database of [30]



**Fig. 12.4** Examples of the five additional categories included in the 13 categories database of [10]

**Fig. 12.5** Gist image classification performances of UNN compared to  $k$ -NN on the 8-cat database (see text for details)

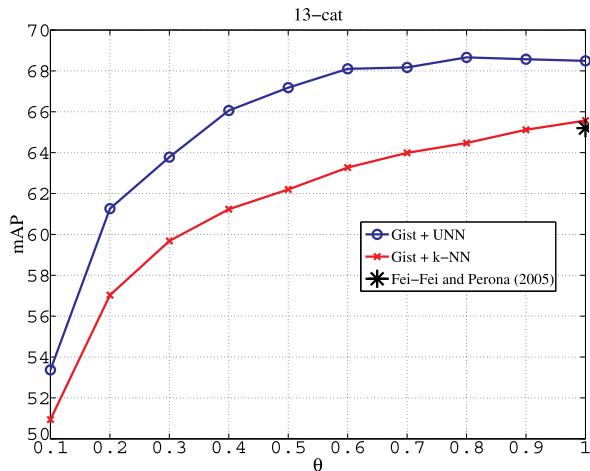


carried out categorization experiments on a larger database of 13 categories as well, denoted as *13-cat*. This dataset was firstly proposed by [10] and contains five more categories, as shown in Fig. 12.4. We extracted Gist descriptors from these images with the most common settings: 4 resolution levels of the Gabor pyramid, 8 orientations per scale and  $4 \times 4$  blocks.<sup>2</sup>

We evaluated classification performances when filtering the prototype dataset, that is, retaining a proportion  $\theta$  of the most relevant examples as prototypes for classification.

<sup>2</sup>The implementation by the authors is available at: <http://people.csail.mit.edu/torralba/code/spatialenvelope/sceneRecognition.m>.

**Fig. 12.6** Gist image classification performances of UNN compared to  $k$ -NN on the 13-cat database (see text for details)



In Figs. 12.5 and 12.6, we show classification performances in terms of the mean Average Precision (MAP)<sup>3</sup> as a function of  $\theta$ . We randomly chose half images to form a training set, while testing on the remaining ones. In each UNN experiment, we fixed the value of  $\theta = T/m$ , thus constraining the number of training iterations  $T$  such that at most  $T$  examples could be retained as prototypes.

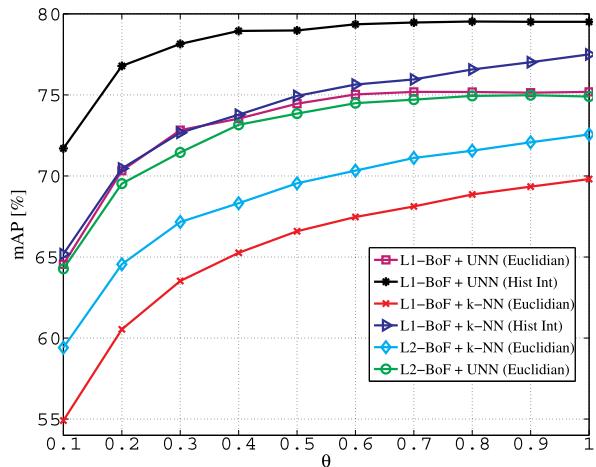
We compared UNN with the classic  $k$ -NN classification. Namely, in order for the classification cost of  $k$ -NN be roughly the same as UNN, we carried out random sampling of the prototype dataset for selecting proportion  $\theta$  (between 10 % and the whole set of examples). UNN significantly outperforms classic  $k$ -NN. Take for example  $\theta = 0.5$  in Fig. 12.5: UNN not only outperforms  $k$ -NN with  $\theta = 0.5$ , its MAP also exceeds that of  $k$ -NN with all data ( $\theta = 1$ ) by almost 2 %. Moreover, on the 13-cat database, UNN outperforms the technique proposed by [10] by 3 % (the asterisk in Fig. 12.6, which corresponds to the best result reported in their paper).

### 12.3.2 Image Categorization Using Bags-of-Features

We now describe experiments with UNN on the Bag-of-Features (BoF) image classification approach. This technique is based on extracting a “bag” of local descriptors (e.g., SIFT descriptors) from an image and vector quantizing them on a precomputed vocabulary of so-called “visual words” [38]. An image is then represented by the histogram of visual word frequencies. This approach provides an effective tool

<sup>3</sup>The MAP was computed by averaging classification rates over categories (diagonal of the confusion matrix) and then averaging those values after repeating each experiment 10 times on different folds.

**Fig. 12.7** Overall results of BoF classification with UNN compared to  $k$ -NN for different settings of histogram normalization (either L1- or L2-norm) and nearest neighbor matching (either Euclidean distance or Histogram Intersection)



for image categorization, as it relies on one single compact descriptor per image, while keeping the informative power of local features. We compare UNN and  $k$ -NN on the 8-cat database (see Sect. 12.3.1).

We used the VLFeat toolbox [41]<sup>4</sup> for extracting gray-scale dense SIFT descriptors at four resolution levels. In particular, a regular grid with spacing 10 pixels was defined over the image and at each grid point SIFT descriptors were computed over circular support patches with radii 4, 8, 12 and 16 pixels. As a result, each point was represented by four different SIFT descriptors. Therefore, given the image size  $256 \times 256$ , we obtained about 2,500 SIFT descriptors per image. Then we split the database in two distinct subsets of images, half for training and half for testing (i.e., 1,344 images in each dataset). In order to build the dictionary of visual words, we applied  $k$ -means clustering on 600,000 SIFT descriptors extracted from training images. For this purpose, we first selected a random subset of training images (about 30 images per class), then we collected all SIFT descriptors of these images and run  $k$ -means. In all the experiments, we computed dictionaries of 500 visual words.

The results obtained with the three different settings are depicted in Fig. 12.7. Notice that UNN using the Histogram Intersection matching outperforms all the compared curves. We also note an improvement (up to 5 % gap for  $k$ -NN and 7 % for UNN) when using L1-normalized Bag-of-Features descriptors compared to Euclidean distance. This similarity measure was firstly proposed by [39] for image indexing based on color histograms, and, more recently, it has been successfully used by [23] in the context of Bag-of-Features image categorization.

<sup>4</sup>Code available at <http://www.vlfeat.org/>.

### 12.3.3 Comparison with SVM and AdaBoost on Image Categorization

Two major issues arise when implementing our UNN algorithm in practice. The first one concerns the distance (or, more generally, the *dissimilarity*) measure used for the  $k$ -NN search. The second one consists in setting the value of  $k$  for both training and testing our prototype-based classifiers.

On the one hand, defining the most appropriate dissimilarity measure for  $k$ -NN search is particularly challenging when dealing with very high-dimensional feature vectors like image descriptors commonly used for categorization. Indeed, the classic metric distances may be inadequate when such vectors are generated by sophisticated pre-processing stages (e.g., vector quantization or unsupervised dictionary learning), thus lying on complex high-dimensional manifolds. In general, this should require an additional distance learning stage in order to define the optimal dissimilarity measure for the particular type of data at hand. In this respect, our UNN method has the advantage of being fully complementary with any metric learning algorithm, acting on the top of the  $k$ -NN search. In Sect. 12.3.2, we have described some examples of using different distances for  $k$ -NN search, particularly focusing on the most suitable dissimilarity measure for histogram-based descriptors.

On the other hand, selecting a good value for  $k$  amounts to learning parameter-dependent weak classifiers, where the parameter  $k$  specifies the size of the voting neighborhood in classification rule (12.10). From the theoretical standpoint, a brute-force approach is possible with boosting: one can define multiple candidate weak classifiers per example, one for each value of  $k$ , that is, for each neighborhood size, and then learn prototypes by optimizing the surrogate risk function over  $k$  as well. This strategy has the advantage of enabling direct learning of  $k$  at training time. However, training several weak classifiers per example without computation tricks would potentially severely impair the applicability of the algorithm on huge datasets. The solution we propose is subtler, as it relies on weighting the neighbors, exploiting the trick that boosting locally fits particular maximum likelihood estimators of class memberships [27]. Using (12.14), we can indeed rewrite (12.10) as:

$$h_c^\ell(\mathbf{x}) \approx \log \prod_{j \sim_k \mathbf{x}, y_{jc} > 0} \frac{\hat{p}(c|j)}{\hat{p}(\bar{c}|j)} - \log \prod_{j \sim_k \mathbf{x}, y_{jc} < 0} \frac{\hat{p}(c|j)}{\hat{p}(\bar{c}|j)}, \quad (12.24)$$

where  $\hat{p}(c|j)$  (resp.  $\hat{p}(\bar{c}|j)$ ) models a conditional probability (resp. not) to belong to class  $c$ . To make the right-hand side of (12.24) closer to a full-fledged maximum likelihood, we have to integrate the density estimators for nearest neighbors,  $\hat{p}(j)$ . We can obviously make the assumption that they are all equal: this would multiply the right-hand side of (12.24) by a positive constant factor, and would not change the outcome of (12.10). Instead, we have modified the classification phase of UNN, and tried a *soft* solution which considers a logistic estimator for a Bernoulli prior which vanishes with the rank of the example in the neighbors, thus decreasing the

importance of the farthest neighbors:

$$\hat{p}(j) = \beta_j = \frac{1}{1 + \exp(\lambda(j - 1))}, \quad (12.25)$$

with  $\lambda > 0$ . The shape prior is chosen this way because it was shown that boosting, as carried out in a number of algorithms—not restricted to the induction of linear separators [27]—locally fits logistic estimators for Bernoulli priors. The soft version of UNN we obtain, called  $\text{UNN}_s$  (for “Soft UNN”), replaces (12.10) by:

$$h_c^\ell(\mathbf{x}) = \sum_{j \sim_k \mathbf{x}} \beta_j \alpha_{jc} y_{jc}. \quad (12.26)$$

Notice that it is useless to enforce the normalization of coefficients  $\beta_j$  in (12.25), because it would not change the classification of  $\text{UNN}_s$ . Notice also that the  $\beta_j$ s in (12.26) are used only to classify new observations: the training steps of  $\text{UNN}_s$  are the same as UNN, and so  $\text{UNN}_s$  meets the same theoretical properties as UNN described in Theorems 12.1, 12.2 and 12.3.

We selected 100 categories from the SUN database [43]. We kept all the images of each category and the inherent unbalancing of the original database. We randomly chose half images to form a training set, while testing on the remaining ones. The MAP was computed by averaging classification rates over categories (diagonal of the confusion matrix) and then averaging those values after repeating each experiment 10 times on different folds. To speed-up processing time, we used the fast implementation of  $k$ -NN proposed by [21].<sup>5</sup> Furthermore, we also developed an optimized version of our program, which exploits *multi-thread* functionalities. We denote this version as  $\text{UNN}_s(\text{MT})$ . All the experiments were run on an Intel Xeon X5690 12-cores processor at 3.46 GHz.

We compared  $\text{UNN}_s$ , SVM with Gaussian RBF Kernel, and AdaBoost with decision stumps<sup>6</sup> (i.e., decision trees with a single internal node), using BoF descriptors. In particular, we followed the guidelines of [20] for carrying out the SVM experiments, thus carrying out cross-validation for selecting the best parameters values for SVM. For the sake of completeness, we also provide results for Gist descriptors with  $\text{UNN}_s$  and  $k$ -NN.

In Table 12.1, we report the MAP for each classification method. Results in these tables are provided as a function of the number of image categories. The most relevant results obtained are also displayed in Fig. 12.8 (mAP as a function of the number of categories) and Figs. 12.9 and 12.10, for the training and classification times, respectively.

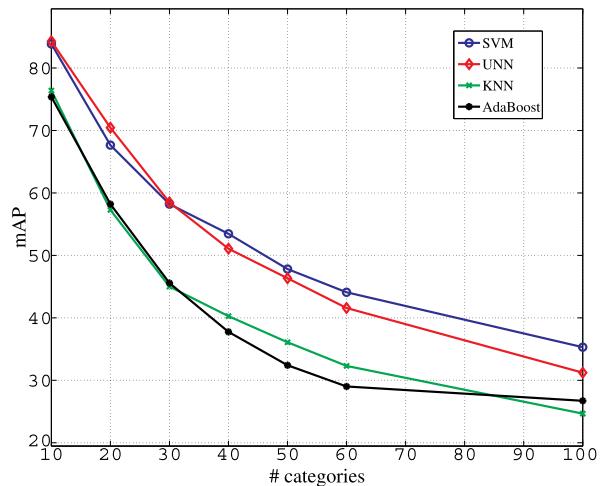
---

<sup>5</sup>Code available at <http://www.irisa.fr/texmex/people/jegou/src.php>.

<sup>6</sup>For AdaBoost, we used the code available at <http://www.mathworks.com/matlabcentral/fileexchange/22997-multiclass-gentleadaboosting>.

**Table 12.1** Classification performances of the different methods we tested in terms of the Mean Average Precision (mAP) as a function of the number of categories

# categories	10	20	30	40	50	60	100
$k$ -NN BoF	76.38	57.28	45.00	40.27	36.09	32.30	24.67
SVM BoF	83.85	67.65	58.21	53.45	47.81	44.09	35.31
AdaBoost BoF	75.37	58.21	45.57	37.75	32.41	29.01	26.72
UNN <sub>s</sub> BoF	84.28	70.44	58.49	51.07	46.34	41.80	31.61
$k$ -NN Gist	64.22	51.48	43.65	39.04	35.65	32.27	25.50
UNN <sub>s</sub> Gist	77.84	66.80	56.37	50.45	46.48	42.75	32.71

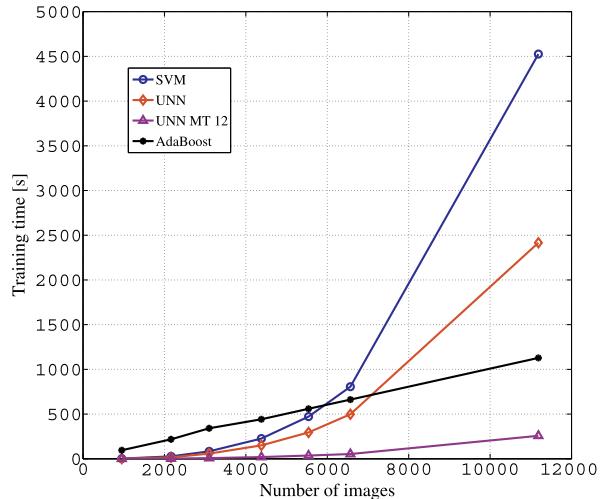
**Fig. 12.8** Classification performances of the tested methods as a function of the number of image categories

We can first notice that BoF descriptors generally outperform Gist, even when this phenomenon is damped as the number of categories increases (above 30). This, overall, follows the trend generally reported in the literature.

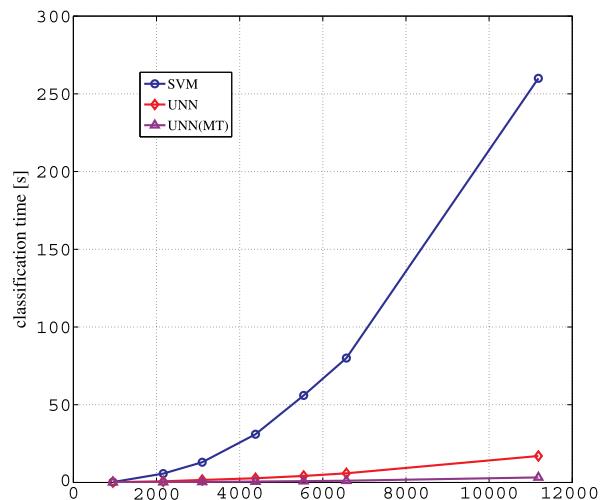
MAP results display that UNN<sub>s</sub> dramatically outperforms AdaBoost (and  $k$ -NN as well); this result, which somehow experimentally confirms that UNN successfully exploits the boosting theory, was quite predictable, as UNN builds a piecewise linear decision function in the initial domain  $\mathcal{O}$ , while AdaBoost builds a linear separator in this domain. SVM, on the other hand, have access to non-linear fitting of data, by lifting the data to a domain whose dimension far exceeds that of  $\mathcal{O}$ . Yet, SVM's testing results are somehow not as good as one might expect from this clearcut theoretical advantage over UNN, and also from the fact that we carried out SVMs with significant parameters optimization [20]. Indeed, UNN<sub>s</sub> even beats SVMs over 10 to 30 categories, being slightly outperformed by them on more categories.

In Tables 12.2 and 12.3, we report the corresponding computation time (in seconds) for the training and classification phase, respectively. Obviously, the computation times over training and testing are also a key for exploiting the experimental re-

**Fig. 12.9** Training time as a function of the number of image categories



**Fig. 12.10** Classification time for UNNs vs SVM as a function of the number of image categories with BoF



sults. Table 12.2 displays that, while the training time of AdaBoost is linear, UNN<sub>s</sub> is a logical clearcut winner over SVM for training: it achieves speedups ranging in between two and more than seventeen over SVM. To assess the validity of these comparisons, we have computed least-square fittings of the training and testing times of UNN<sub>s</sub> vs AdaBoost vs SVM (all with BoF), with both linear ( $s = aC + b$ ,  $s$  being the time in seconds, and  $C$  the number of categories) and polynomial ( $s = bC^a$ ) fittings, with the objective to foresee on the best models what might happen on domains with classes ranging from hundreds to (tens of) thousands. The best models are displayed in Table 12.4. The coefficients of determination show that only a slim portion of the data is not explained by the models shown.

**Table 12.2** Computation time [s] for the training phase

# categories	10	20	30	40	50	60	100
# training images	951	2,162	3,099	4,381	5,540	6,568	11,186
$k$ -NN BoF				0			
SVM BoF	2.4	27	83	226	472	806	4526
AdaBoost BoF	96	218	341	442	559	662	1128
UNN <sub>s</sub> BoF	1.7	16	58	150	295	498	2146
UNN <sub>s</sub> (MT) BoF	0.3	2.5	7.8	19	36	53	257

**Table 12.3** Computation time [s] for the testing phase

# categories	10	20	30	40	50	60	100
# test images	951	2,162	3,099	4,381	5,540	6,568	11,186
$k$ -NN BoF	0.20	1.0	2.0	4.0	6.0	9.0	22.0
SVM BoF	0.25	5.7	13	31	56	80	260
AdaBoost BoF	0.02	0.1	0.25	0.43	0.67	0.95	2.74
UNN <sub>s</sub> BoF	0.21	0.72	1.6	2.7	4.2	5.9	17
UNN <sub>s</sub> (MT) BoF	0.08	0.2	0.37	0.58	0.84	1.11	3.25

Models confirm that the training time of AdaBoost is linear. This is not a surprise, as it is ran with stumps as weak classifiers. Allowing decision trees with more than one internal node would have certainly blown the linear time barrier. While they are roughly equivalent for UNN<sub>s</sub> and AdaBoost (Table 12.3), testing times reveal a much bigger gap between UNN<sub>s</sub> and SVM, as displayed in Fig. 12.10. Exploiting the models of Table 12.4, we obtain the ratio:

$$s_{\text{SVM}}/s_{\text{UNN}} \approx \mathcal{O}(m), \quad (12.27)$$

while, for the multi-thread implementation, we obtain:

$$s_{\text{SVM}}/s_{\text{UNN},\text{MT}} \approx \mathcal{O}(m^{1.3}). \quad (12.28)$$

The ratio is always in favor of UNN, and of order the number of examples. Hence, the execution time for UNN<sub>s</sub> should allow to classify many images in reduced time compared to SVM: from Table 12.4, UNN should already classify almost twenty times as many images as SVM in a single minute. In such a case, UNN<sub>s</sub> should also classify almost twice as many images as AdaBoost. Thus, UNN provides the best MAP/time trade-off among the tested methods, which suggests that UNN might well be more than a legal contender to classification methods dealing with huge domains, or domains where the testing set is huge compared to the training set, which is the case, for instance, for cell classification in biological images [16]. Finally, we have only scratched experimental optimizations for UNN, and have not optimized

**Table 12.4** Best fits for training/testing times [s] as a function of the number of classes  $C$ , or the number of images  $m$  in the training sample/to be tested. The model indicated is the best fit among models of the type  $y = ax + b$  and  $y = bx^a$ , according to the coefficient of determination  $r^2$ . For all but two models,  $r^2 > 0.999 \approx 1.0$  (the exceptions are \*), for which  $r^2 \approx 0.97$ , and (\*\*), for which  $r^2 \approx 0.99$ .  $m_{1m}$  is the number, estimated by the model, of images that can be processed in 1 minute (see text for details)

	Training	Testing	
	$s = f(C)$	$s = f(m)$	$m_{1m}$
SYM BoF	$s = (1.42 \times 10^{-3}) \times C^{3.25}$	$s = (1.9 \times 10^{-9}) \times m^{3.05}$	$s = (4.94 \times 10^{-4}) \times C^{2.94} (*)$
AddBoost BoF	$s = -11.40 + 11.37 \times C$	$s = 7.63 + 0.10 \times m$	$s = (2.16 \times 10^{-4}) \times C^{2.05}$
UNNs BoF	$s = (1.24 \times 10^{-3}) \times C^{3.16}$	$s = (2.43 \times 10^{-9}) \times m^{2.96}$	$s = (4.19 \times 10^{-8}) \times m^{1.93}$
UNNs MT BoF	$s = (3.85 \times 10^{-4}) \times C^{2.91}$	$s = (2.07 \times 10^{-9}) \times m^{2.74}$	$s = (9.19 \times 10^{-7}) \times m^{1.78}$
			24 567
			$s = (1.82 \times 10^{-3}) \times C^{1.58}$
			$s = (2.57 \times 10^{-6}) \times m^{1.48} (**)$
			95 175

UNN from the complexity-theoretic standpoint, so we expect room space for further significant improvement of its training/testing times.

## 12.4 Discussion and Perspectives

UNN provides us with a sound blend of two powerful yet simple classification algorithms: nearest neighbors and boosting. While the analysis of the mixing is not straightforward—such as for the convergence and boosting properties in Theorems 12.1–12.3—UNN remains simple to state and implement, even in the multi-class case. It also appears to be a interesting contender to SVM: without using the kernel trick mapping examples to high dimensional feature spaces, UNN manages to fit nonlinear classifiers in the initial feature space whose accuracy clearly compete with SVM’s.

We think that this simplicity opens avenues for future research on the way separate extensions and improvements of nearest neighbors and boosting might be transferred to UNN. One example is the inclusion of powerful density estimation techniques that would fit better than our simple logistic convolution of priors in (12.25).

Another example involves improved sophistication from the classifier’s standpoint, in particular with metric distance learning and the kernelization of the input space [47]. This, we expect, would enable significant improvements of categorization performances.

A third example involves improvements from the nearest neighbor search standpoint. Novel techniques exist that make embeddings in a real-valued vector space of nearest neighbors queries, thus transforming the data space with the hope to achieve good compromises between reducing the processing complexity of nearest neighbor queries while not reducing the accuracy of (vanilla) nearest neighbors in the space learnt [2, 25]. Clearly, such approaches do not tackle the same problem as us, as UNN directly processes nearest neighbors on the data’s ambient space. Nevertheless, they are very interesting from the perspective standpoint because this new data space is learnt with (Ada)boosting. A neat combination with UNN might thus offer the possibility to kill two birds in one boosting shot for nearest neighbors: learn an improved data space, *and* learn in this data space an improved nearest neighbor classifier with UNN. The questions raised by such perspective are not only experimental, as basically only the contractiveness of the approach of [2] is formally known to date. Transferring, or even improving, the boosting properties of UNN in such sophisticated blends would thus be more than interesting.

## 12.5 Conclusion

In this work, we contribute to fill an important void of NN methods, showing how boosting can be transferred to  $k$ -NN classification, with convergence rates guarantees for a *large* number of surrogates. Our UNN algorithm generalizes classic  $k$ -NN

to weighted voting where weights, the so-called leveraging coefficients, are iteratively learned by UNN. We prove that this algorithm converges to the global optimum of many surrogate risks in competitive times under very mild assumptions.

Our work is also the first extensive assessment of UNN to computer vision related tasks. Comparisons with  $k$ -NN, support vector machines and AdaBoost, using Gist or Bag-of-Feature descriptors, on simulated and real domains, display the ability of UNN to be competitive with its contenders, achieving high mAP in comparatively reduced training and testing times.

Avenues for future research include blending UNN with other approaches that bias the domain towards the improvement of nearest neighbors rules, or that learn more sophisticated metrics over data.

## Appendix

**Generic UNN Algorithm** The general version of UNN is shown in Algorithm 2. This algorithm induces the leveraged  $k$ -NN rule (12.10) for the broad class of surrogate losses meeting conditions of [4], thus generalizing Algorithm 1. Namely, we constrain  $\psi$  to meet the following conditions: (i)  $\text{im}(\psi) = \mathbb{R}_+$ , (ii)  $\nabla_\psi(0) < 0$  ( $\nabla_\psi$  is the conventional derivative of  $\psi$  loss function), and (iii)  $\psi$  is strictly convex and differentiable. (i) and (ii) imply that  $\psi$  is *classification-calibrated*: its local minimization is roughly tied up to that of the empirical risk [4]. (iii) implies convenient algorithmic properties for the minimization of the surrogate risk [28]. Three common examples have been shown in Eqs. (12.7)–(12.6).

The main bottleneck of UNN is step [I.1], as Eq. (12.30) is non-linear, *but* it always has a solution, finite under mild assumptions [28]: in our case,  $\delta_j$  is guaranteed to be finite when there is no total matching or mismatching of example  $j$ 's memberships with its reciprocal neighbors', for the class at hand. The second column of Table 12.5 contains the solutions to (12.30) for surrogate losses mentioned in Sect. 12.2.2. Those solutions are always exact for the exponential loss ( $\psi^{\text{exp}}$ ) and squared loss ( $\psi^{\text{squ}}$ ); for the logistic loss ( $\psi^{\text{log}}$ ) it is exact when the weights in the reciprocal neighborhood of  $j$  are the same, otherwise it is approximated. Since starting weights are all the same, exactness can be guaranteed during a large number of inner rounds depending on which order is used to choice the examples. Table 12.5 helps to formalize the finiteness condition on  $\delta_j$  mentioned above: when either sum of weights in (12.29) is zero, the solutions in the first and third line of Table 12.5 are not finite. A simple strategy to cope with numerical problems arising from such situations is that proposed by [35]. (See Sect. 12.2.4.) Table 12.5 also shows how the weight update rule (12.31) specializes for the mentioned losses.

**Proofsketch of Theorem 12.1** We show that UNN converges to the global optimum of any surrogate risk (Sect. 12.2.5). For this purpose, let us consider the

**Algorithm 2** Universal Nearest Neighbors UNN( $\mathcal{S}, \psi$ )

---

**Input:**  $\mathcal{S} = \{(x_i, y_i), i = 1, 2, \dots, m, x_i \in \mathcal{X}, y_i \in \{-\frac{1}{C-1}, 1\}^C\}$ ,  $\psi$  meeting (i), (ii), (iii) ([Appendix](#));

$$r_{ij}^{(c)} \doteq \begin{cases} y_i c y_j c & \text{if } j \sim_k i \\ 0 & \text{otherwise} \end{cases} \quad \forall i, j = 1, 2, \dots, m, c = 1, 2, \dots, C \quad \triangleright k\text{-NN edge matrix}$$

**for**  $c = 1, 2, \dots, C$  **do**

$$\alpha_{jc} \leftarrow 0, \quad \forall j = 1, 2, \dots, m \quad \triangleright \text{Leveraging coefficients}$$

$$w_i \leftarrow -\nabla_\psi(0) \in \mathbb{R}_{+*}^m, \quad \forall i = 1, 2, \dots, m \quad \triangleright \text{Example weights}$$

**for**  $t = 1, 2, \dots, T$  **do**

[I.0]  $j \leftarrow \text{WIC}(\{1, 2, \dots, m\}, t)$   $\triangleright$  Weak index chooser oracle

[I.1]

$$w_j^+ = \sum_{i: r_{ij}^{(c)} > 0} w_i, \quad w_j^- = \sum_{i: r_{ij}^{(c)} < 0} w_i, \quad (12.29)$$

Let  $\delta_j \in \mathbb{R}$  solution of:

$$\sum_{i=1}^m r_{ij}^{(c)} \nabla_\psi (\delta_j r_{ij}^{(c)} + \nabla_\psi^{-1}(-w_i)) = 0; \quad (12.30)$$

[I.2]  $\forall i : j \sim_k i$ , let

$$w_i \leftarrow -\nabla_\psi (\delta_j r_{ij}^{(c)} + \nabla_\psi^{-1}(-w_i)). \quad (12.31)$$

[I.3]  $\alpha_{jc} \leftarrow \alpha_{jc} + \delta_j$

**end for**

**end for**

**Output:**  $h_c(x_{i'}) = \sum_{i \sim_k i'} \alpha_{ic} y_{ic}, \quad \forall c = 1, 2, \dots, C$

---

**Table 12.5** Three common loss functions and the corresponding solutions  $\delta_j$  of (12.30) and  $w_i$  of (12.31). (Vector  $r_j^{(c)}$  designates column  $j$  of  $R^{(c)}$  and  $\|\cdot\|_1$  is the  $L_1$  norm.) The rightmost column says whether it is (A)lways the solution, or whether it is when the weights of reciprocal neighbors of  $j$  are the (S)ame

Loss function	$\delta_j$ in (12.30)	$w_i$ in (12.31)	Opt
$\psi^{\exp} \doteq \exp(-x)$	$\frac{1}{2} \log(\frac{w_j^{(c)+}}{w_j^{(c)-}})$	$w_i \exp(-\delta_j r_{ij}^{(c)})$	A
$\psi^{\text{squ}} \doteq (1-x)^2$	$\frac{w_j^{(c)+} - w_j^{(c)-}}{2\ r_j^{(c)}\ _1}$	$w_i - 2\delta_j r_{ij}^{(c)}$	A
$\psi^{\log} \doteq \log(1 + \exp(-x))$	$\log(\frac{w_j^{(c)+}}{w_j^{(c)-}})$	$\frac{w_i \exp(-\delta_j r_{ij}^{(c)})}{1 - w_i(1 + \exp(-\delta_j r_{ij}^{(c)}))}$	S

surrogate risk (12.5) for a given class  $c = 1, 2, \dots, C$ :

$$\varepsilon_c^\psi(\mathbf{h}, \mathcal{S}) \doteq \frac{1}{m} \sum_{i=1}^m \psi(\varrho(\mathbf{h}, i, c)). \quad (12.32)$$

In this section, we use the following notations:

- $\tilde{\psi}(x) \doteq \psi^*( -x )$ , where  $\psi^*(x) \doteq x\nabla_{\psi}^{-1}(x) - \psi(\nabla_{\psi}^{-1}(x))$  is the Legendre conjugate of  $\psi$ , which is strictly convex and differentiable as well. ( $\tilde{\psi}$  is related to  $\psi$  in such a way that:  $\nabla_{\tilde{\psi}}(x) = -\nabla_{\psi}^{-1}(-x)$ .)
- $D_{\tilde{\psi}}(w_i \| w'_i) \doteq \tilde{\psi}(w_i) - \tilde{\psi}(w'_i) - (w_i - w'_i)\nabla_{\tilde{\psi}}(w'_i)$  is the Bregman divergence with generator  $\tilde{\psi}$  [28].

Let  $\mathbf{w}_t$  denote the  $t$ th weight vector inside the “**for**  $c$ ” loop of Algorithm 2 (assuming  $\mathbf{w}_0$  is the initialization of  $\mathbf{w}$ ); similarly,  $\mathbf{h}_t^\ell$  denotes the  $t$ th leveraged  $k$ -NN rule obtained after the update in [I.3]. The following fundamental identity holds, whose proof follows from [28]:

$$\psi(\varrho(\mathbf{h}_t^\ell, i, c)) = g + D_{\tilde{\psi}}(0 \| w_{ti}), \quad (12.33)$$

where  $g(m) \doteq -\tilde{\psi}(0)$  does not depend on the  $k$ -NN rule. In particular, Eq. (12.33) makes the connection between the real-valued classification problem and a geometric problem in the non-metric space of weights. Moreover, Eq. (12.33) proves in handy as one computes the *difference* between two successive surrogates:  $\varepsilon_c^\psi(\mathbf{h}_{t+1}^\ell, \mathcal{S}) - \varepsilon_c^\psi(\mathbf{h}_t^\ell, \mathcal{S})$ . Indeed, plugging Eq. (12.33) in Eq. (12.32), and computing  $\delta_j$  in Eq. (12.30) so as to bring  $\mathbf{h}_{t+1}^\ell$  from  $\mathbf{h}_t^\ell$ , we obtain the following identity:

$$\varepsilon_c^\psi(\mathbf{h}_{t+1}^\ell, \mathcal{S}) - \varepsilon_c^\psi(\mathbf{h}_t^\ell, \mathcal{S}) = -\frac{1}{m} \sum_{i=1}^m D_{\tilde{\psi}}(w_{(t+1)i} \| w_{ti}). \quad (12.34)$$

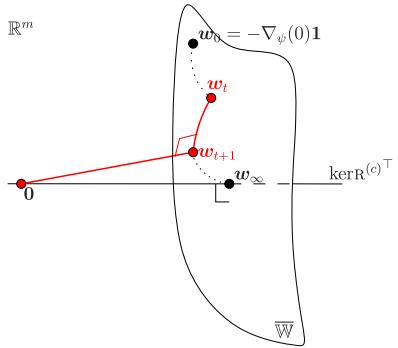
Since Bregman divergences are non negative and meet the identity of the indiscernibles, (12.34) implies that steps [I.1]–[I.3] guarantee the decrease of (12.32) as long as  $\delta_j \neq 0$ . But (12.32) is lowerbounded, hence UNN must converge.

In addition, it converges to the global optimum of the risk (12.5). Since predictions for each class are independent, the proof consists in showing that (12.32) converges to its global minimum for each  $c$ . Let us assume this convergence for the current class  $c$ . Then, following the reasoning of Nock and Nielsen [28], (12.30) and (12.31) imply that, when any possible  $\delta_j = 0$ , the weight vector, say  $\mathbf{w}_\infty$ , satisfies  $\mathbf{R}^{(c)\top} \mathbf{w}^\top = \mathbf{0}$ , that is,  $\mathbf{w}_\infty \in \ker \mathbf{R}^{(c)\top}$ , and  $\mathbf{w}_\infty$  is unique. But the kernel of  $\mathbf{R}^{(c)\top}$  and  $\overline{\mathbb{W}}$ , the closure of  $\mathbb{W}$  (i.e., the manifold where  $\mathbf{w}$ ’s live), are provably Bregman orthogonal [28], thus yielding:

$$\underbrace{\sum_{i=1}^m D_{\tilde{\psi}}(0 \| w_i)}_{m\varepsilon_c^\psi(\mathbf{h}^\ell, \mathcal{S}) - mg} = \underbrace{\sum_{i=1}^m D_{\tilde{\psi}}(0 \| w_{\infty i})}_{m\varepsilon_c^\psi(\mathbf{h}_\infty^\ell, \mathcal{S}) - mg} + \underbrace{\sum_{i=1}^m D_{\tilde{\psi}}(w_{\infty i} \| w_i)}_{\geq 0}, \quad \forall \mathbf{w} \in \overline{\mathbb{W}}. \quad (12.35)$$

Underbraces use (12.33) in (12.32), and  $\mathbf{h}^\ell$  is a leveraged  $k$ -NN rule corresponding to  $\mathbf{w}$ . One obtains that  $\mathbf{h}_\infty^\ell$  achieves the global minimum of (12.32), as claimed.

**Fig. 12.11** A geometric view of how UNN converges to the global optimum of (12.5). (See Appendix for details and notations.)



The proofsketch is graphically summarized in Fig. 12.11. In particular, two crucial *Bregman orthogonalities* are mentioned [28]. The red one symbolizes:

$$\sum_{i=1}^m D_{\tilde{\psi}}(0 \| w_{ti}) = \sum_{i=1}^m D_{\tilde{\psi}}(0 \| w_{(t+1)i}) + \sum_{i=1}^m D_{\tilde{\psi}}(w_{(t+1)i} \| w_{ti}), \quad (12.36)$$

which is equivalent to (12.34). The black one on  $w_\infty$  is (12.35).

**Proofsketch of Theorem 12.2** Using developments analogous to those of [28], UNN can be shown to be equivalent to AdaBoost in which  $m$  weak classifiers are available, each one being an example. Each weak classifier returns a value in  $\{-1, 0, 1\}$ , where 0 is reserved for examples outside the reciprocal neighborhood. Theorem 3 of [35] brings in our case:

$$\varepsilon^{0/1}(\mathbf{h}^\ell, \mathcal{S}) \leq \frac{1}{C} \sum_{c=1}^C \prod_{t=1}^T Z_t^{(c)}, \quad (12.37)$$

where  $Z_t^{(c)} \doteq \sum_{i=1}^m \tilde{w}_{it}^{(c)}$  is the normalizing coefficient for each weight vector in UNN. ( $\tilde{w}_{it}^{(c)}$  denotes the weight of example  $i$  at iteration  $(t, c)$  of UNN, and the Tilda notation refers to weights normalized to unity at each step.) It follows that:

$$\begin{aligned} Z_t^{(c)} &= 1 - \tilde{w}_{jt}^{(c)+-} \left( 1 - 2\sqrt{p_{jt}^{(c)}(1 - p_{jt}^{(c)})} \right) \\ &\leq \exp \left( -\tilde{w}_{jt}^{(c)+-} \left( 1 - 2\sqrt{p_{jt}^{(c)}(1 - p_{jt}^{(c)})} \right) \right) \\ &\leq \exp(-\eta(1 - \sqrt{1 - 4\gamma^2})) \leq \exp(-2\eta\gamma^2), \end{aligned}$$

where  $\tilde{w}_{jt}^{(c)+-} \doteq \tilde{w}_{jt}^{(c)+} + \tilde{w}_{jt}^{(c)-}$ ,  $p_{jt}^{(c)} \doteq \tilde{w}_{jt}^{(c)+}/\tilde{w}_{jt}^{(c)+-} = w_{jt}^{(c)+}/w_{jt}^{(c)+-}$ . The first inequality uses  $1 - x \leq \exp(-x)$ , and the second the (WIA). Since even when the (WIA) does not hold, we still observe  $Z_t^{(c)} \leq 1$ , plugging the last inequality in (12.37) yields the statement of the theorem.

**Proofsketch of Theorem 12.3** We plug in the weight notation the iteration  $t$  and class  $c$ , so that  $w_{ti}^{(c)}$  denotes the weight of example  $x_i$  prior to iteration  $t$  for class  $c$  in UNN (inside the “**for**  $c$ ” loop of Algorithm 2, letting  $\mathbf{w}_0$  denote the initial value of  $\mathbf{w}$ ). To save space in some computations below, we also denote for short:

$$\bar{\varepsilon}^\psi(\mathbf{h}_T^\ell, \mathcal{S}) \doteq \frac{1}{C} \sum_{c=1}^C \varepsilon_c^\psi(\mathbf{h}_T^\ell, \mathcal{S}). \quad (12.38)$$

$\psi$  is  $\omega$  strongly smooth is equivalent to  $\tilde{\psi}$  being strongly convex with parameter  $\omega^{-1}$  [22], that is,

$$\tilde{\psi}(w) - \frac{1}{2\omega} w^2 \quad (12.39)$$

is convex. Here, we have made use of the following notations:  $\tilde{\psi}(x) \doteq \psi^*(-x)$ , where  $\psi^*(x) \doteq x\nabla_\psi^{-1}(x) - \psi(\nabla_\psi^{-1}(x))$  is the Legendre conjugate of  $\psi$ . Since a convex function  $h$  satisfies  $h(w') \geq h(w) + \nabla_h(w)(w' - w)$ , applying inequality (12.39) taking as  $h$  the function in (12.39) yields,  $\forall t = 1, 2, \dots, T, \forall i = 1, 2, \dots, m, \forall c = 1, 2, \dots, C$ :

$$\begin{aligned} D_{\tilde{\psi}}(w_{(t+1)i}^{(c)} \| w_{ti}^{(c)}) &= D_{\tilde{\psi}}(w_{ti}^{(c)} + (w_{(t+1)i}^{(c)} - w_{ti}^{(c)}) \| w_{ti}^{(c)}) \\ &\geq \frac{1}{2\omega} (w_{(t+1)i}^{(c)} - w_{ti}^{(c)})^2, \end{aligned} \quad (12.40)$$

where we recall that  $D_\psi$  denotes the Bregman divergence with generator  $\psi$  (12.22). On the other hand, Cauchy–Schwarz inequality yields:

$$\begin{aligned} \forall j \in \mathcal{S}, \quad \sum_{i:j \sim_k i} (r_{ij}^{(c)})^2 \sum_{i:j \sim_k i} (w_{(t+1)i}^{(c)} - w_{ti}^{(c)})^2 &\geq \left( \sum_{i:j \sim_k i} r_{ij}^{(c)} (w_{(t+1)i}^{(c)} - w_{ti}^{(c)}) \right)^2 \\ &= \left( \sum_{i:j \sim_k i} r_{ij}^{(c)} w_{ti}^{(c)} \right)^2. \end{aligned} \quad (12.41)$$

The equality in (12.41) holds because  $\sum_{i:j \sim_k i} r_{ij}^{(c)} w_{(t+1)i}^{(c)} = 0$ , which is exactly (12.30). We obtain:

$$\begin{aligned} \frac{1}{m} \sum_{i=1}^m D_{\tilde{\psi}}(w_{(t+1)i}^{(c)} \| w_{ti}^{(c)}) &= \frac{1}{m} \sum_{i:j \sim_k i} D_{\tilde{\psi}}(w_{(t+1)i}^{(c)} \| w_{ti}^{(c)}) \\ &\geq \frac{1}{2\omega m} \sum_{i:j \sim_k i} (w_{(t+1)i}^{(c)} - w_{ti}^{(c)})^2 \end{aligned} \quad (12.42)$$

$$\geq \frac{1}{2\omega m} \frac{(\sum_{i:t \sim ki} r_{it}^{(c)} w_{ti}^{(c)})^2}{\sum_{i:t \sim ki} (r_{it}^{(c)})^2} \quad (12.43)$$

$$\geq \frac{\vartheta^2}{2\omega m} \times \frac{1}{\sum_{i:t \sim ki} (r_{it}^{(c)})^2} \quad (12.44)$$

Here, (12.42) follows from (12.40), (12.43) follows from (12.41), and (12.44) follows from (12.20). Adding (12.44) for  $c = 1, 2, \dots, C$  and  $t = 1, 2, \dots, T$ , and then dividing by  $C$ , we obtain:

$$\begin{aligned} & \frac{1}{C} \sum_{c=1}^C \sum_{t=1}^T \frac{1}{m} \sum_{i=1}^m D_{\tilde{\psi}}(w_{(t+1)i}^{(c)} \| w_{ti}^{(c)}) \\ & \geq \frac{T\vartheta^2}{2\omega m} \times \left( \frac{1}{TC} \times \sum_{c=1}^C \sum_{t=1}^T \frac{1}{\sum_{i:t \sim ki} (r_{it}^{(c)})^2} \right). \end{aligned} \quad (12.45)$$

We now work on the big parenthesis which depends solely upon the examples. We have:

$$\begin{aligned} & \left( \frac{1}{TC} \times \sum_{c=1}^C \sum_{t=1}^T \frac{1}{\sum_{i:t \sim ki} (r_{it}^{(c)})^2} \right)^{-1} \\ & \leq \frac{1}{TC} \sum_{c=1}^C \sum_{t=1}^T \sum_{i:t \sim ki} (r_{it}^{(c)})^2 \end{aligned} \quad (12.46)$$

$$\begin{aligned} & = \frac{1}{TC} \sum_{c=1}^C \sum_{t=1}^T \sum_{i \in \text{NN}_k(\mathbf{x}_t)} y_{tc}^2 y_{ic}^2 \\ & \leq \frac{1}{TC} \sum_{c=1}^C \sum_{t=1}^T \sum_{i \in \text{NN}_k(\mathbf{x}_t)} \left( \frac{|y_{tc}|}{2} + \frac{|y_{ic}|}{2} \right) \end{aligned} \quad (12.47)$$

$$\begin{aligned} & = \frac{k}{TC} \sum_{t=1}^T \sum_{c=1}^C \frac{|y_{tc}|}{2} + \frac{1}{TC} \sum_{t=1}^T \sum_{i \in \text{NN}_k(\mathbf{x}_t)} \sum_{c=1}^C \frac{|y_{ic}|}{2} \\ & = \frac{k}{(C-1)}. \end{aligned} \quad (12.48)$$

Here, (12.46) holds because of the Arithmetic-Geometric-Harmonic inequality, and (12.47) is Young's inequality<sup>7</sup> with  $p = q = 2$ . Plugging (12.48) into (12.45), we obtain:

---

<sup>7</sup>We recall young inequality: for any  $p, q$  Hölder conjugates ( $p > 1$ ,  $(1/p) + (1/q) = 1$ ), we have  $yy' \leq y^p/p + y'^q/q$ , assuming  $y, y' \geq 0$ .

$$\frac{1}{C} \sum_{c=1}^C \sum_{t=1}^T \frac{1}{m} \sum_{i=1}^m D_{\tilde{\psi}}(w_{(t+1)i}^{(c)} \| w_{ti}^{(c)}) \geq \frac{T(C-1)\vartheta^2}{2\omega mk}. \quad (12.49)$$

Now, UNN meets the following property, which can easily be shown to hold with our class encoding as well:

$$\varepsilon_c^\psi(\mathbf{h}_{t+1}^\ell, \mathcal{S}) - \varepsilon_c^\psi(\mathbf{h}_t^\ell, \mathcal{S}) = -\frac{1}{m} \sum_{i=1}^m D_{\tilde{\psi}}(w_{(t+1)i}^{(c)} \| w_{ti}^{(c)}). \quad (12.50)$$

Adding (12.50) for  $t = 0, 2, \dots, T-1$  and  $c = 1, 2, \dots, C$ , we obtain:

$$\frac{1}{C} \sum_{c=1}^C \varepsilon_c^\psi(\mathbf{h}_T^\ell, \mathcal{S}) - \psi(0) = -\frac{1}{C} \sum_{c=1}^C \sum_{t=1}^T \frac{1}{m} \sum_{i=1}^m D_{\tilde{\psi}}(w_{(t+1)i}^{(c)} \| w_{ti}^{(c)}). \quad (12.51)$$

Plugging (12.49) into (12.51), we obtain:

$$\bar{\varepsilon}^\psi(\mathbf{h}_T^\ell, \mathcal{S}) \leq \psi(0) - \frac{T(C-1)\vartheta^2}{2\omega mk}. \quad (12.52)$$

But the following inequality holds between the average surrogate risk and the empirical risk of the leveraged  $k$ -NN rule  $\mathbf{h}_T^\ell$ , because of (i):

$$\begin{aligned} \bar{\varepsilon}^\psi(\mathbf{h}_T^\ell, \mathcal{S}) &= \frac{1}{C} \sum_{c=1}^C \varepsilon_c^\psi(\mathbf{h}_T^\ell, \mathcal{S}) \\ &= \frac{1}{mC} \sum_{c=1}^C \sum_{i=1}^m \psi \left( y_{ic} \sum_{j:j \sim ki} \alpha_{jc} y_{jc} \right) \\ &\geq \frac{\psi(0)}{mC} \sum_{c=1}^C \sum_{i=1}^m \left[ y_{ic} \sum_{j:j \sim ki} \alpha_{jc} y_{jc} < 0 \right] \\ &= \psi(0) \varepsilon^{0/1}(\mathbf{h}_T^\ell, \mathcal{S}), \end{aligned} \quad (12.53)$$

so that, putting altogether (12.52) and (12.53) and using the fact that  $\psi(0) > 0$  because of (i)–(ii), we have after  $T$  rounds of boosting for each class: *that is,*

$$\varepsilon^{0/1}(\mathbf{h}_T^\ell, \mathcal{S}) \leq 1 - \frac{T(C-1)\vartheta^2}{2\psi(0)\omega mk}. \quad (12.54)$$

There remains to compute the minimal value of  $T$  for which the right-hand side of (12.54) becomes no greater than some user-fixed  $\tau \in [0, 1]$  to obtain the bound in (12.23).

## References

1. Amores J, Sebe N, Radeva P (2006) Boosting the distance estimation: application to the  $k$ -nearest neighbor classifier. *Pattern Recognit Lett* 27(3):201–209
2. Athitsos V, Alon J, Sclaroff S, Kollios G (2008) BoostMap: an embedding method for efficient nearest neighbor retrieval. *IEEE Trans Pattern Anal Mach Intell* 30(1):89–104
3. Bartlett P, Traskin M (2007) Adaboost is consistent. *J Mach Learn Res* 8:2347–2368
4. Bartlett P, Jordan M, McAuliffe JD (2006) Convexity, classification, and risk bounds. *J Am Stat Assoc* 101:138–156
5. Boutell MR, Luo J, Shen X, Brown CM (2004) Learning multi-label scene classification. *Pattern Recognit* 37(9):1757–1771
6. Brighton H, Mellish C (2002) Advances in instance selection for instance-based learning algorithms. *Data Min Knowl Discov* 6:153–172
7. Cucala L, Marin JM, Robert CP, Titterington DM (2009) A bayesian reassessment of nearest-neighbor classification. *J Am Stat Assoc* 104(485):263–273
8. Dudani S (1976) The distance-weighted  $k$ -nearest-neighbor rule. *IEEE Trans Syst Man Cybern* 6(4):325–327
9. Escalano Ruiz F, Suau Pérez P, Bonev BI (2009) Information theory in computer vision and pattern recognition. Springer, London
10. Fei-Fei L, Perona P (2005) A bayesian hierarchical model for learning natural scene categories. In: IEEE computer society conference on computer vision and pattern recognition (CVPR), pp 524–531
11. Fukunaga K, Flick T (1984) An optimal global nearest neighbor metric. *IEEE Trans Pattern Anal Mach Intell* 6(3):314–318
12. García-Pedrajas N, Ortiz-Boyer D (2009) Boosting  $k$ -nearest neighbor classifier by means of input space projection. *Expert Syst Appl* 36(7):10,570–10,582
13. Gionis A, Indyk P, Motwani R (1999) Similarity search in high dimensions via hashing. In: Proc international conference on very large databases, pp 518–529
14. Grauman K, Darrell T (2005) The pyramid match kernel: discriminative classification with sets of image features. In: IEEE international conference on computer vision (ICCV), pp 1458–1465
15. Gupta L, Pathangay V, Patra A, Dyana A, Das S (2007) Indoor versus outdoor scene classification using probabilistic neural network. *EURASIP J Appl Signal Process* 2007(1):123
16. Bel Haj Ali W, Piro P, Crescence L, Giampaglia D, Ferhat O, Darcourt J, Pourcher T, Barlaud M (2012) Changes in the subcellular localization of a plasma membrane protein studied by bioinspired UNN learning classification of biologic cell images. In: International conference on computer vision theory and applications (VISAPP)
17. Hart PE (1968) The condensed nearest neighbor rule. *IEEE Trans Inf Theory* 14:515–516
18. Hastie T, Tibshirani R (1996) Discriminant adaptive nearest neighbor classification. *IEEE Trans Pattern Anal Mach Intell* 18(6):607–616
19. Holmes CC, Adams NM (2003) Likelihood inference in nearest-neighbour classification models. *Biometrika* 90:99–112
20. Hsu CW, Chang CC, Lin CJ (2003) A practical guide to support vector classification. Technical report
21. Jégou H, Douze M, Schmid C (2011) Product quantization for nearest neighbor search. *IEEE Trans Pattern Anal Mach Intell* 33(1):117–128
22. Kakade S, Shalev-Shwartz S, Tewari A (2009) Applications of strong convexity–strong smoothness duality to learning with matrices. Technical report
23. Lazebnik S, Schmid C, Ponce J (2006) Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: IEEE computer society conference on computer vision and pattern recognition (CVPR), pp 2169–2178
24. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110

25. Masip D, Vitrià J (2006) Boosted discriminant projections for nearest neighbor classification. *Pattern Recognit* 39(2):164–170
26. Nguyen X, Wainwright MJ, Jordan MI (2009) On surrogate loss functions and  $f$ -divergences. *Ann Stat* 37:876–904
27. Nock R, Nielsen F (2009) Bregman divergences and surrogates for learning. *IEEE Trans Pattern Anal Mach Intell* 31(11):2048–2059
28. Nock R, Nielsen F (2009) On the efficient minimization of classification calibrated surrogates. In: Advances in neural information processing systems (NIPS), vol 21, pp 1201–1208
29. Nock R, Sebban M (2001) An improved bound on the finite-sample risk of the nearest neighbor rule. *Pattern Recognit Lett* 22(3/4):407–412
30. Oliva A, Torralba A (2001) Modeling the shape of the scene: a holistic representation of the spatial envelope. *Int J Comput Vis* 42(3):145–175
31. Paredes R (2006) Learning weighted metrics to minimize nearest-neighbor classification error. *IEEE Trans Pattern Anal Mach Intell* 28(7):1100–1110
32. Payne A, Singh S (2005) Indoor vs. outdoor scene classification in digital photographs. *Pattern Recognit* 38(10):1533–1545
33. Piro P, Nock R, Nielsen F, Barlaud M (2012) Leveraging  $k$ -NN for generic classification boosting. *Neurocomputing* 80:3–9
34. Quattoni A, Torralba A (2009) Recognizing indoor scenes. In: IEEE computer society conference on computer vision and pattern recognition (CVPR)
35. Schapire RE, Singer Y (1999) Improved boosting algorithms using confidence-rated predictions. *Mach Learn J* 37:297–336
36. Serrano N, Savakis AE, Luo JB (2004) Improved scene classification using efficient low-level features and semantic cues. *Pattern Recognit* 37:1773–1784
37. Shakhnarovich G, Darell T, Indyk P (2006) Nearest-neighbors methods in learning and vision. MIT Press, Cambridge
38. Sivic J, Zisserman A (2003) Video google: a text retrieval approach to object matching in videos. In: IEEE international conference on computer vision (ICCV), vol 2, pp 1470–1477
39. Swain MJ, Ballard DH (1991) Color indexing. *Int J Comput Vis* 7:11–32
40. Torralba A, Murphy K, Freeman W, Rubin M (2003) Context-based vision system for place and object recognition. In: IEEE international conference on computer vision (ICCV), pp 273–280
41. Vedaldi A, Fulkerson B (2008) VLFeat: an open and portable library of computer vision algorithms. <http://www.vlfeat.org>
42. Vogel J, Schiele B (2007) Semantic modeling of natural scenes for content-based image retrieval. *Int J Comput Vis* 72(2):133–157
43. Xiao J, Hays J, Ehinger KA, Oliva A, Torralba A (2010) SUN database: large-scale scene recognition from abbey to zoo. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 3485–3492
44. Yu K, Ji L, Zhang X (2002) Kernel nearest-neighbor algorithm. *Neural Process Lett* 15(2):147–156
45. Yuan M, Wegkamp M (2010) Classification methods with reject option based on convex risk minimization. *J Mach Learn Res* 11:111–130
46. Zhang ML, Zhou ZH (2007) ML-kNN: a lazy learning approach to multi-label learning. *Pattern Recognit* 40(7):2038–2048
47. Zhang H, Berg AC, Maire M, Malik J (2006) SVM-kNN: discriminative nearest neighbor classification for visual category recognition. In: IEEE computer society conference on computer vision and pattern recognition (CVPR), pp 2126–2136

48. Zhu J, Rosset S, Zou H, Hastie T (2009) Multi-class adaboost. *Stat Interface* 2:349–360
49. Zuo W, Zhang D, Wang K (2008) On kernel difference-weighted  $k$ -nearest neighbor classification. *Pattern Anal Appl* 11(3–4):247–257

# Chapter 13

## Learning Object Detectors in Stationary Environments

Peter M. Roth, Sabine Sternig, and Horst Bischof

**Abstract** The most successful approach for object detection is still applying a sliding window technique, where a pre-trained classifier is evaluated on different locations and scales. In this chapter, we interrogate this strategy in the context of stationary environments. In particular, having a fixed camera position observing the same scene a lot of prior (spatio-temporal) information is available. Exploiting this specific scene information allows for (a) improving the detection performance and (b) for reducing the model complexity; both on reduced computational costs! These benefits are demonstrated for two different real-world tasks (i.e., person and car detection). In particular, we apply two different evaluation/update strategies (holistic, grid-based), where any suited online learner can be applied. In our case we demonstrate the proposed approaches for different applications and scenarios, clearly showing their benefits compared to generic methods.

### 13.1 Introduction

A very prominent approach for object detection is to use a sliding window technique (e.g., [7, 28, 31, 42]). Each patch of an image is tested if it is consistent with a previously estimated model or not. Finally, all consistent patches are reported. The model may either represent a specific object (e.g., a specific cup, that is represented by different views) or a class of objects (e.g., faces, pedestrians, cars, or bikes), where specific instances are not distinguished. These models are mostly based on local features described by a classifier, (e.g., AdaBoost [12] or support vector machine [40]), which is typically obtained by learning. Hence, when discussing the problem of object detection, implicitly, the problem of visual learning is addressed.

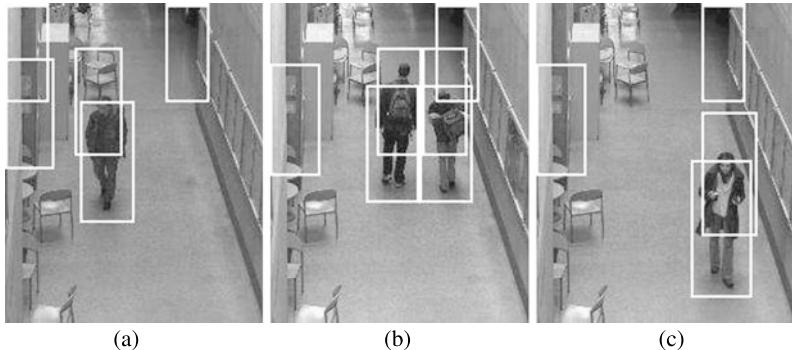
The goal of all of such approaches is to build a generic object model that is applicable for all possible scenarios and tasks (e.g., [7, 10, 20]). The drawback of these methods, however, is that the detectors are usually not very specific (i.e., they return false alarms). As can be seen from Fig. 13.1 even if general classifiers (“broad application”) are trained from a very large number of training samples, they often fail for

---

P.M. Roth (✉) · S. Sternig · H. Bischof

Institute for Computer Graphics and Vision, Graz University of Technology, Graz, Austria

e-mail: [pmroth@icg.tugraz.at](mailto:pmroth@icg.tugraz.at)



**Fig. 13.1** Typical false positives obtained from generic object detectors: background patches that are similar to persons and partially detected persons

specific situations. This is caused by the main limitation of such approaches—a representative dataset is needed, which is not available for many applications. Since not all variability, especially for the negative class (i.e., all possible backgrounds), can be captured this results in a low recall and an insufficient precision.

One way to overcome this problem is to additionally use scene specific information to reduce the number of false alarms [18]. To further improve the classification results specific classifiers (“narrow applications”) can be applied, which are designed to solve a specific task. In fact, for most object detection problems (e.g., visual surveillance) we can assume a stationary camera, allowing us also to include scene specific information also during training. Even though this aspect is often ignored, it was shown that in this way not only less training data is required but that such classifiers are usually also better in terms of accuracy and efficiency [22, 28, 31, 46]. Moreover, an online learning algorithm can be applied [19, 28, 46]. Thus, the system can adapt to changing environments (e.g., changing illumination conditions) and these variations need not to be handled by the model. In fact, in this way the complexity of the problem is reduced and a more efficient classifier can be trained.

Adaptive systems, however, suffer from one main problem: new unlabeled data has to be robustly included into an already built model. Typical approaches are self-training (e.g., [24, 30]), co-training (e.g., [6, 22]), semi-supervised learning (e.g., [15]), or the application of oracles<sup>1</sup> (e.g., [28, 45]). Semi-supervised methods, however, are often biased by the prior and thus only a “limited” information gain can be achieved whereas oracles often provide too less new information. Self- or co-training suffer from the problem that the theoretical constraints can not be assured in practice or that they rely on a direct feedback of the current classifier—both resulting in unreliable classifiers.

In this chapter, we address the problem of making the learning of adaptive scene-specific detectors more stable. Therefore, we discuss two different approaches to

<sup>1</sup>We refer a classifier to as an oracle, if it has a high precision, even at a low recall, and can thus be used to generate new training samples.

cope with this problem: *Conservative Learning* and *Classifier Grids*. The main idea of Conservative Learning is to use a small number of labeled samples to train a discriminative classifier and then to online adapt it to a specific scene. For that purpose, additionally a generative model, that can even be estimated from a smaller number of samples, is used to verify the discriminative model. In this way, using an active learning strategy the most valuable samples can be recognized, ensuring both, a fast adaption to the scene and a very powerful detector. Classifier Grids further simplify the task's complexity by reducing the problem from the static scene to just a small spatio-temporal region. The main idea is to apply a separate classifier on each image location (grid element) to learn an adaptive but still robust scene specific object representation.

The proposed learning strategies are quite general, and any suitable online learner can be applied. For our specific tasks, however, we found that online Boosting (see Sect. 13.2) is a valid choice. In the experiments, we demonstrate the benefits of both approaches for the task of object detection from stationary cameras and give a detailed comparison to state-of-the-art methods. Even just demonstrated for person and car detection, the approach is quite general and not limited to these specific applications.

The chapter is structured as follows. First, in Sect. 13.2 we review online GradientBoost and its variants, where our online learned detectors build on. Next, in Sects. 13.3 and 13.4 we discuss the ideas of Conservative Learning and Classifier Grids. Experimental results for both approaches compared to existing generic approaches are given in Sect. 13.5. Finally, we summarize and conclude our findings in Sect. 13.6.

## 13.2 Online Learning for Object Detection

The scene-specific object detection approaches presented in Sects. 13.3 and 13.4 are quite general and any suitable learning method could be applied. However, having real-world scenarios we would require classifiers which are efficient during training and evaluation, online capable, and easy to adapt to different applications. Therefore, we propose to use *Online GradientBoost* [21], which is not only very efficient but can also easily be adapted for the different requirements given by our specific tasks. In the following, we first summarize the main concept of online GradientBoost in Sect. 13.2.1 and then introduce two extensions coping better with unreliable data, that is, TransientBoost [36] and inverse MILBoost [37], in Sects. 13.2.2 and 13.2.3.

### 13.2.1 Online GradientBoost

Given a labeled dataset,  $\mathcal{X} = \{(x_1, y_1), \dots, (x_N, y_N)\}$ ,  $x_n \in \mathbb{R}^D$ ,  $y_n \in \{-1, +1\}$ , the goal of Boosting [33] is to estimate a strong classifier

$$H(x) = \sum_{j=1}^M \alpha_j h_j(x) \quad (13.1)$$

via a linear combination of  $M$  weak classifiers  $h_j(x)$ , which have only to perform better than random guessing. During evaluation then for a sample  $x$  the label  $\hat{y} = \text{sign}(H(x))$  should be predicted. Due to its effectiveness various different variants such as [11, 14, 26, 34] have been proposed. In particular, in computer vision Boosting for Feature Selection [38, 41] has gained a lot of interest. The main idea is that each feature  $f_j$  within a feature pool  $\mathcal{F}$  corresponds to a single weak classifier  $h_j$  and to use AdaBoost [12] to select an informative subset of  $M$  features.

In particular, we adopt these ideas for use with GradientBoost [13]. Let  $\ell(\cdot)$  be a loss function and

$$\mathcal{L}(H(x)) = \sum_{n=1}^N \ell(y_n H(x_n)) \quad (13.2)$$

be the empirical loss, the goal of GradientBoost is to estimate the strong classifier  $H(x)$  such that this loss is minimized. Hence, at stage  $t$ , we are searching a base function  $h_t$  which maximizes the correlation with negative direction of the loss function:

$$h_t(x) = \arg \max_{h(x)} -\nabla \mathcal{L}^T h(x), \quad (13.3)$$

where  $\nabla \mathcal{L}$  is the gradient vector of the loss at  $H_{t-1}(x) = \sum_{m=1}^{t-1} h_m(x)$ . This can be simplified to

$$h_t(x) = \arg \max_{h(x)} - \sum_{n=1}^N y_n \underbrace{\ell'(y_n H_{t-1}(x_n))}_{-w_n} h(x_n), \quad (13.4)$$

where  $\ell'(\cdot)$  are the derivatives of the loss with respect to  $H_{t-1}$  and  $w_n$  are the sample's weights. Thus, the optimization is equivalent to maximizing the weighted classification accuracy

$$h_t(x) = \arg \max_{h(x)} \sum_{n=1}^N w_n y_n h(x_n). \quad (13.5)$$

This formulation can simply be adopted for online learning by using selectors as introduced in [16]. Each selector  $s_m(x)$  holds a set of  $K$  weak classifiers  $\{h_{m,1}(x), \dots, h_{m,K}(x)\}$  and is represented by its best weak classifier  $h_{m,k}(x)$ , that is, the weak classifier with the lowest estimated error  $\varepsilon_{m,k}$ , which is estimated via

$$\varepsilon_{m,k} \leftarrow \varepsilon_{m,k} + w_n \mathbb{I}(\text{sign}(h_m^k(x_n)) \neq y_n), \quad (13.6)$$

where  $\mathbb{I}(\cdot)$  is the indicator function. The optimization step in Eq. (13.4) is then performed iteratively by propagating the samples through the selectors and updating

**Algorithm 1** On-line GradientBoost**Input:** Training sample  $(x_n, y_n)$ **Output:** Strong classifier  $H(x)$ 

- 1: Set  $H_0(x_n) = 0$
- 2: Set the initial weight  $w_n = -\ell'(0)$
- 3: **for**  $m = 1$  to  $M$  **do**
- 4:     **for**  $k = 1$  to  $K$  **do**
- 5:         Train weak learner  $h_{m,k}(x)$  with sample  $(x_n, y_n)$  and weight  $w_n$
- 6:         Compute the error:  $\varepsilon_{m,k} \leftarrow \varepsilon_{m,k} + w_n \mathbb{I}(\text{sign}(h_m^k(x_n)) \neq y_n)$
- 7:     **end for**
- 8:     Find the best weak learner:  $j = \arg \min_k \varepsilon_{m,k}$
- 9:     Set  $h_m(x_n) = h_m^j(x_n)$
- 10:    Set  $H_m(x_n) = H_{m-1}(x_n) + h_m(x_n)$
- 11:    Set the weight  $w_n = -\ell'(y_n H_m(x_n))$
- 12: **end for**
- 13: Output the final classifier  $H(x)$

the weight estimate  $w_n$  according to the negative derivative of the loss function. Thus, optimizing Eq. (13.4) is independent of the applied loss function. The overall approach is summarized in Algorithm 1.

### 13.2.2 TransientBoost

On-line GradientBoost was designed for a binary classification problem. However, by introducing weak learners that are able to handle more than two classes, we can extend it to the multi-class domain. For that purpose, any weak learner providing confidence-rated responses can be applied. In particular, following Friedman et al. [14], to define a  $J$ -class problem, we use symmetric multiple logistic transformation as weak learners:

$$h_j(x) = \log p_j(x) - \frac{1}{J} \sum_{l=1}^J \log p_l(x), \quad (13.7)$$

where  $p_j(x) = P(y_j = 1|x)$ . In particular, they showed that if the sum over the weak classifier responses over all classes is normalized to zero, that is,  $\sum_{j=1}^J h_j(x) = 0$ , the probability  $p_j(x)$  can be estimated by using histograms. Moreover, histograms are highly appropriate for online learning since they can easily be updated.

Now having a multi-class formulation we can model reliable and unreliable data using different classes, that is,  $y = [+1, -1]$  for the reliable data and  $y = [+2, -2]$  for the unreliable data. Thus, during an update the classifier is provided a sample

$x_t$  and a label  $y_t \in \{-2, -1, +1, +2\}$  and depending on the label the corresponding histograms are updated. Moreover, for the reliable samples the histogram updates are performed incrementally whereas for the unreliable transient samples an iir-like filtering of the histogram bins is applied (i.e., the knowledge is scaled down according to its age). In this way the reliable information is accumulated whereas the unreliable information allows higher adaptivity, but is fading out quickly (depending on the forgetting rate), thus, avoiding drifting.

The next, crucial step is to include the uncertainty of the sample  $\langle x_t, y_t \rangle$  into the feature selection procedure. In each update step, similar to the binary case, the best weak classifier  $h_{m,k}$  within a selector  $s_m$  is estimated according to its error. The error is updated depending on the weight of the correct classified samples  $\lambda_{m,n}^c$  and the misclassified samples  $\lambda_{m,n}^w$  within each weak classifier. However, the error updates must be adapted according to the multi-class formulation. If the prediction was correct, that is, the signum of the classifier response  $h_{m,n}$  equals the signum of class label used to update the classifier  $y_t$  ( $\text{sign}(h_{m,n}(x)) = \text{sign}(y_t)$ ), the weight  $\lambda_{m,n}^c$  is updated:

$$\lambda_{m,n}^c = \lambda_{m,n}^c + w_n; \quad (13.8)$$

otherwise the weight  $\lambda_{m,n}^w$  is updated:

$$\lambda_{m,n}^w = \lambda_{m,n}^w + w_n, \quad (13.9)$$

where  $w_n$  is the current estimated weight of the current sample. In the original GradientBoost formulation any differentiable loss function  $\ell$  can be used to update the weight by  $w_n = -\ell'(y_t H_m(x_t))$ , where  $H_m(x)$  is the combination of the first  $m$  weak classifiers and  $y_t$  is the label of the current sample. In our case, however, we have to re-formulate the weight update according to our multi-class model. Otherwise the classifier would try to distinguish between the reliable and the unreliable classes and would penalize samples that are already classified correctly. Hence, since we are interested in discrimination of positive and negative classes, we have to change the weight update to

$$w_n = -\ell'(\text{sign}(y_t) H_m(x)). \quad (13.10)$$

### 13.2.3 Inverse MILBoost

A different way to cope with unreliable, ambiguously labeled data is Multiple Instance Learning (MIL), which was first introduced by Dietterich et al. [8]. Since this is a widespread problem, there has been a considerable interest and many popular supervised learning algorithms such as SVM [3] or Boosting [4, 43] have been adapted in order to incorporate the MIL constraints.

In contrast to supervised learning algorithms, where each sample (instance) is provided a label, in multiple-instance learning the training samples are grouped into bags  $B_i \subset \mathbb{R}^d$ ,  $i = 1, \dots, N$ . Each bag consists of an arbitrary number of instances:

$B_i = \{x_{1i}, x_{2i}, \dots, x_{mi}\}$ . Negative bags  $B_i^-$  are required to consist only of negative instances, whereas for positive bags  $B_i^+$  it has only to be guaranteed that they contain at least one positive instance. There are no further restrictions to the non-positive instances within the positive bag  $B_i^+$ , they might not even belong to the negative class. The task now is to learn either a bag classifier  $f : B \rightarrow \{-1, 1\}$  or an instance classifier  $f : \mathbb{R}^d \rightarrow \{-1, 1\}$ . However, a bag classifier can follow automatically from instance prediction, e.g., by using the *max* operator over posterior probabilities over the instances  $p_{ij}$  within the  $i$ th bag:  $p_i = \max_j \{p_{ij}\}$ .

In context of GradientBoost this can easily be realized by defining a new loss function, which optimizes the binary log likelihood over bags:

$$\log \mathcal{L} = \sum_i (y_i \log p(y_i) + (1 - y_i) \log(1 - p(y_i))), \quad (13.11)$$

where the instance probability can be estimated using a sigmoid function

$$p(y|x) = \sigma(H(x)) = \frac{1}{1 + e^{-H(x)}}, \quad (13.12)$$

which requires a gradient descent in function space. The bag probability  $p(y|B)$  is modeled by the Noisy-OR (NOR) operator:

$$p(y_i|B_i) = 1 - \prod_{j=1} (1 - p(y_i|x_{ij})). \quad (13.13)$$

This formulation assumes the ambiguity in the positive data. However, for many tasks the uncertain labeled data mainly concerns the negative samples, requiring to adapt the original MIL idea. In fact, we require that the negative bags  $B_i^-$  contain only one negative example whereas the positive bag  $B_i^+$  consists only of positive examples:

$$\forall x_{ij}^+ \in B_i^+ : \quad y(x_{ij}^+) = 1, \quad (13.14)$$

$$\exists x_{ij}^- \in B_i^- : \quad y(x_{ij}^-) = -1. \quad (13.15)$$

Thus, in order to correctly calculate the loss  $\mathcal{L}$  by inverting the problem, we have to switch the labels between the positive and the negative class (*inverse MIL*). This causes to focus on examples that are more likely to be correct negative examples, which directly fits to our problem.

### 13.3 Scene-Specific Detectors

The main motivation for training scene-specific detectors is twofold. First, as illustrated for instance, in Fig. 13.1 generic classifiers cannot cope with all variabilities in the data. Second, often only small amount of labeled training data is available, thus, it might be useful also to exploit the information naturally given by unlabeled

**Algorithm 2** Active learning

---

**Input:** unlabeled samples  $\mathbf{U}$ , classifier  $C_{t-1}$ 
**Output:** classifier  $C_t$ 

- 1: **while** teacher can label samples  $u_j$  **do**
  - 2:     Apply  $C_{t-1}$  to all samples  $u_j$
  - 3:     Let  $Q$  find the  $m$  most informative samples  $u_q$
  - 4:     Let teacher  $S$  assign labels  $y_q$  to samples  $u_q$
  - 5:     Re-train classifier:  $C_t$
  - 6: **end while**
- 

data. Both issues can be addressed by online learning. Thus, a classifier can adapt to environmental conditions, which have not to be modeled in beforehand. Surprisingly a lot of research was focused on developing new (online) learning methods, however, only little work was done on sampling meaningful training data. Hence, typically a *passive learning* strategy [23] is applied, where a huge number of samples are randomly drawn from the training set. In the following, we introduce an approach which samples training data in a more sophisticated way exploiting information given by a fixed camera setup.

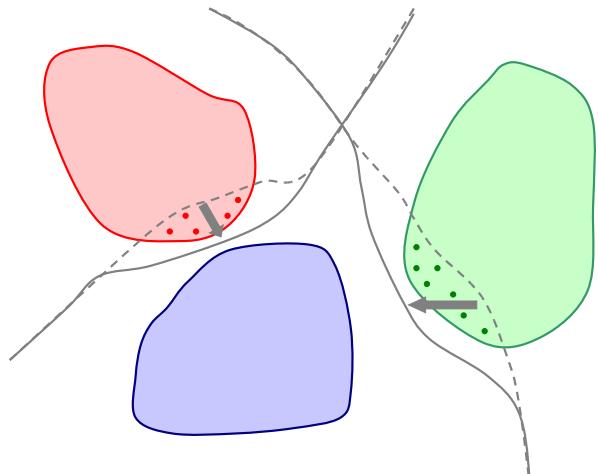
### 13.3.1 Active Learning

To overcome these problems an adaptive learning algorithm taking advantage of the ideas if *active learning* (e.g., [23, 39, 47]) can be applied. In general, an active learner can be considered a quintuple  $(C, Q, S, \mathbf{L}, \mathbf{U})$  [23], where  $C$  is a classifier,  $Q$  is a query function,  $S$  is a supervisor (teacher), and  $\mathbf{L}$  and  $\mathbf{U}$  are a set of labeled and unlabeled data, respectively. First, an initial classifier  $C_0$  is trained from the labeled set  $\mathbf{L}$ . Given a classifier  $C_{t-1}$ , then the query function  $Q$  selects the most informative unlabeled samples from  $\mathbf{U}$  and the supervisor  $S$  is requested to label them. Using the thus labeled samples the current classifier is re-trained obtaining a new classifier  $C_t$ . This procedure is summarized in Algorithm 2.

When considering an adaptive system we can start from a small set of labeled data  $\mathbf{L}$ . But usually the unlabeled data  $\mathbf{U}$  is not available in advance. Assuming we have already trained a classifier  $C_{t-1}$  the first crucial point at time  $t$  is to define a set of unlabeled data  $\mathbf{U}_t$  and a suitable query function  $Q$ , raising two questions. First, we have to discover the most valuable samples. It has been shown by [29] that it is more effective to sample at the current estimated decision boundary than the unknown true boundary. In context of online learning, the most valuable samples are exactly those that have been misclassified by the current classifier. This is illustrated in Fig. 13.2.

The red and green points indicate the samples that were wrongly classified by the current classifier. Obviously, these points are much better samples than randomly selected points from the class! Hence, the algorithm is focused on the hard samples and the current decision boundary (dashed gray line) can be moved such that those

**Fig. 13.2** Sampling at the current decision boundary is more efficient than random sampling: updating using only a small number of wrongly classified samples (red and green points) allows to move the current decision boundary (dashed gray line) in the right direction (solid gray line)



samples are correctly classified (solid gray line). For a detailed theoretical discussion, see [29].

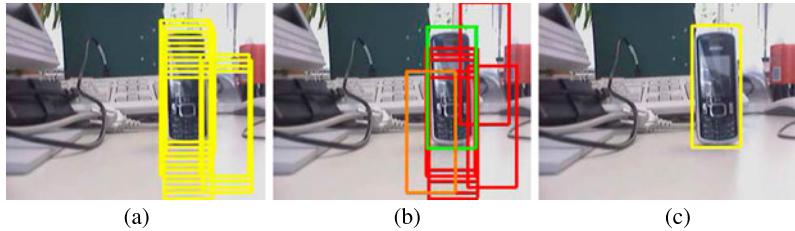
The second crucial point is the definition of the supervisor  $S$ . In fact, new unlabeled data has to be robustly included into the already built model. More formally, at time  $t$  given a classifier  $C_{t-1}$  and an unlabeled example  $x_t \in \mathbb{R}^m$ . Then, a reliable supervisor is needed, that robustly estimates a label  $y_t \in \{+1, -1\}$  for  $x_t$ .

### 13.3.2 Conservative Learning

The goal now is to identify meaningful choices for the query function  $Q$  as well as for the supervisor  $S$  for the given task. In fact, considering object detection the misclassified samples are the detected false positives. Thus, it would be desirable to identify those false positives and use them to update the classifier. This is illustrated in Fig. 13.3, where it can be seen that using especially those samples an initially insufficient classifier can be significantly improved.

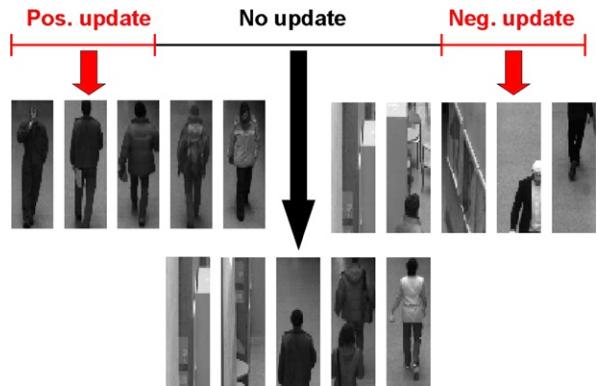
The remaining problem is how to conveniently decide if a selected detection is a true or a false positive! In fact, we additionally apply a generative model for this purpose, that is trained from the same (positive) samples. This is motivated by two properties of generative models: first, a sufficient generative model can be trained from a smaller number of samples and, second, generative models are robust against wrongly labeled data (up to some extent). Thus, we use two models in parallel, a discriminative one, which actually performs the detection, and a generative one, which assures robustness and serves for verification.

In this way the output of the discriminative classifier is verified by the generative model. The detected false positives are fed back into the discriminative classifier as negative examples and true positives as positive examples. Since we have a huge amount of unlabeled data, that is readily available (i.e., we have video stream),



**Fig. 13.3** Active learning—suitable updates for learning a detector: (a) classifier evaluated at time  $t - 1$ , (b) updates selected at time  $t$ , and (c) classifier evaluated at time  $t + 1$ .

**Fig. 13.4** Conservative updating the discriminant classifier using the generative model



we can use very *conservative* decision criteria. In fact, the generative model returns a similarity measure and we use two conservative thresholds  $\theta_{\text{pos}}$  and  $\theta_{\text{neg}}$  for this decision. Hence, we refer to this approach as *Conservative Learning*. This update rule is illustrated in Fig. 13.4.

Thus, only a very small number of samples, that were selected by the query function, are labeled by the supervisor and are finally used for updating. In this way we guarantee that the discriminative classifier is only learned from clean correctly labeled samples. In addition, the thus labeled samples can be used to update the generative model. Exploiting the huge amount of video data this process can be iterated to produce a stable and robust classifier. This update learning strategy is summarized more formally in Algorithm 3.

The update rules described here are very general and can be applied to any pair of discriminative/generative online learners. However, in our particular case we apply online boosting (see Sect. 13.2.1) as discriminative model and incremental PCA [35] as generative model.

To illustrate incremental learning process, we visualize the update over time in Fig. 13.5. Therefore, the first, the third, the 34th and the 64th frame of a sequence of total 300 frames are shown. A red bounding box indicates a negative update, a green bounding box a positive update and a white bounding box indicates a detection that is not used for updating the existing classifier. It can be seen that there is a great number of negative updates within the very first frames (see Figs. 13.5(a), (b)). If the

**Algorithm 3** Conservative Learning update strategy

---

**Input:** discriminative classifier  $D_{t-1}$ , generative classifier  $G_{t-1}$ , unknown image  $\mathbf{I}$

**Output:** classifier  $D_t$ , classifier  $G_t$

- 1: Evaluate  $D_{t-1}$  on  $\mathbf{I}$  obtaining  $J$  detections  $x_j$
- 2: **for**  $j = 1, \dots, J$  **do**

  - if**  $G_{t-1}(x_j) < \theta_{\text{pos}}$  **then**

    - $update(D_{t-1}, x_j, +)$ ,     $update(G_{t-1}, x_j)$

  - else if**  $G_{t-1}(x_j) > \theta_{\text{neg}}$  **then**

    - $update(D_{t-1}, x_j, -)$

  - end if**

- 3: **end for**

---

incremental learning process is running for a longer time, we finally get a stable classifier and only a small number of updates are necessary (see Figs. 13.5(c), (d)).

The learning framework discussed so far has two main disadvantages. First, still a small amount of labeled data is required. Second, new positives examples are only obtained from the detection results, that are already described well by the model. Thus, completely different appearances are not captured and the classifier is biased by the very first positive examples. To overcome these problems, we need a robust method to automatically label additionally positives samples. In fact, we can apply low level cues such as motion detection or tracking for this purpose. The whole learning approach is thus summarized in Algorithm 4.

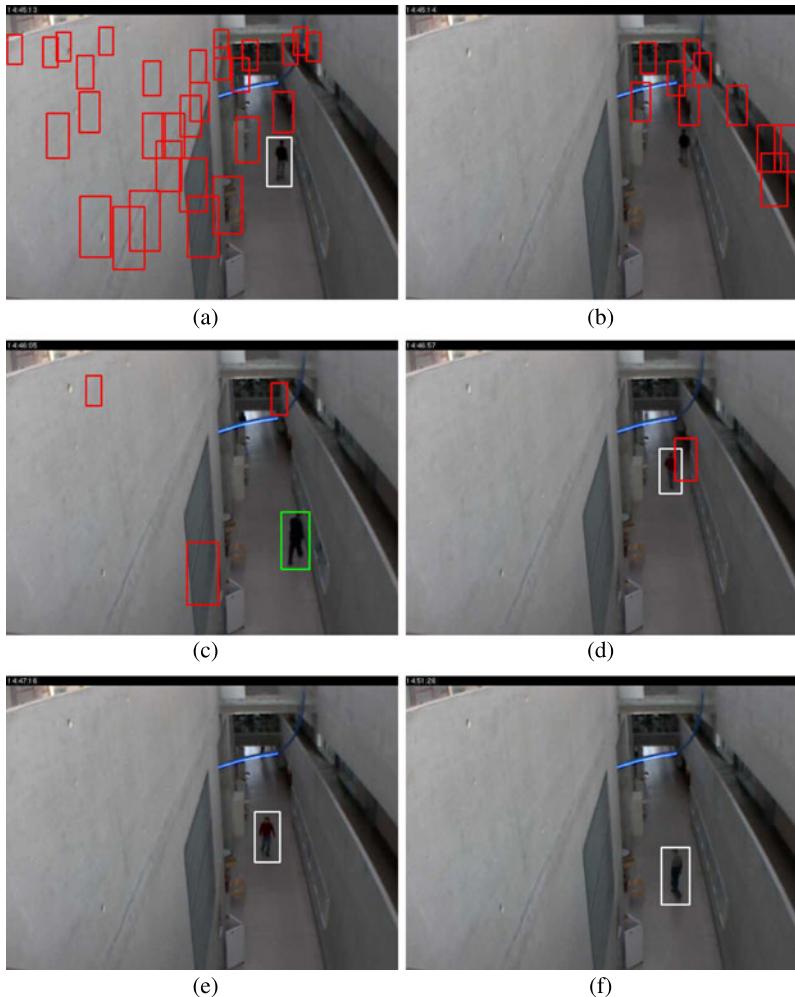
## 13.4 Classifier Grids

The idea of Classifier Grids is to sample an input image by using a fixed highly overlapping grid (both in location and scale), where each grid element  $i = 1, \dots, N$  corresponds to one classifier  $C_i$ . This is illustrated in Fig. 13.6. Thus, the classification task that has to be handled by one classifier  $C_i$  is reduced to discriminate the background of the specific grid element from the object-of-interest. Due to this simplification, less complex classifiers can be applied.

In particular, the grid-based representation is well suited for compact smart online classifier, which can be evaluated and updated very efficiently. However, as online systems suffer from the problem to robustly include new data they tend to drift, that is, the classifier starts to learn something completely wrong and would yield arbitrarily wrong results. Therefore, we introduce three different update strategies for Classifier Grids that cope with different aspects of online learning.

### 13.4.1 Fixed Update Rules

The first problem we address is to ensure stability over time. Therefore, we build on two observations. First, for many task labeled positive data, that is, describing the



**Fig. 13.5** On-line Conservative Learning—learning progress: (a) updates for 1st frame, (b) updates for 3rd frame, (c) updates for 34th frame, and (d) update for 64th frame, (e) detection but no update for 75th frame, (f) detection but no update for 172th frame

object of interest, is available. Second, in a typical fixed camera scenario a specific image position is occupied by an object very rarely. Thus, we can define fixed update rules for online learning that exploit previously given labeled positive data but also sample unlabeled negative information from the scene [17].

More formally, let  $\mathcal{X}^+$  be a set of representative positive (hand) labeled examples. Then, using

$$\langle \mathbf{x}, +1 \rangle, \quad \mathbf{x} \in \mathcal{X}^+ \quad (13.16)$$

**Algorithm 4** Conservative Learning

---

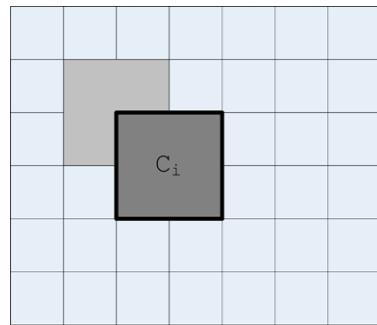
```

1: Automatically collect pos./neg. training samples
2: Train initial classifier  $D_0$ 
3: Train initial classifier  $G_0$ 
4: while  $t$  do
5:   update  $D_t$  (Algorithm 3)
6:   update  $G_t$  (Algorithm 3)
7:   Acquire new pos. samples  $x$ 
8:    $update(D_t, x, +)$ 
9:    $update(G_t, x)$ 
10: end while

```

---

**Fig. 13.6** Concept of grid-based classification:  
a highly overlapping grid is placed over the image, where each grid element corresponds to a single classifier



to update the classifier is a correct positive update by definition. Moreover, the probability that an object is present at patch  $\mathbf{x}_i$  is given by

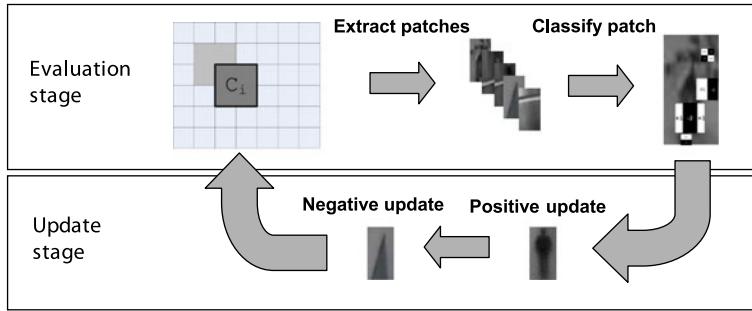
$$P(\mathbf{x}_i = \text{object}) = \frac{\#p_i}{\Delta t}, \quad (13.17)$$

where  $\#p_i$  is the number of objects entirely present in a particular patch within the time interval  $\Delta t$ . Thus, the negative update with the current patch

$$\langle \mathbf{x}_{i,t}, -1 \rangle \quad (13.18)$$

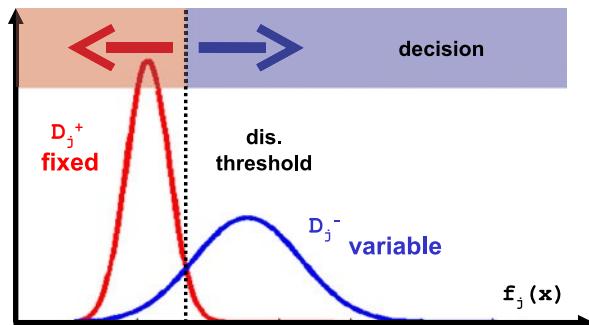
is correct most of the time (wrong with probability  $P(\mathbf{x}_i = \text{object})$ ). The probability of a wrong update for this particular image patch is indeed very low. To further decrease this probability instead of sampling from the original images a background model can be applied instead. Even though this strategy is quite general, in our case we apply the online Boosting approach presented in Sect. 13.2.1 within each grid element. The overall approach is illustrated in Fig. 13.7.

To further reduce the computational effort and to increase the stability, more sophisticated weak classifiers can be applied within the online Boosting framework [32]. Since the object-of-interest is known before, its appearance can be described in advance by a finite set  $\mathcal{X}^+$  of positive samples. Thus, for each feature



**Fig. 13.7** Online updating the classifier grid approach using fixed update rules

**Fig. 13.8** For each feature  $f_j$  the discriminative threshold  $\theta$  is calculated using the *fixed* (pre-trained) positive distribution  $D_j^+$  and the *variable* negative distribution  $D_j^-$



$f_j \in \mathcal{F}$  a generative model can be estimated via the distributions  $D_j^+$ . Since  $\mathcal{X}^+$  is fixed, the distributions  $D_j^+$  can also be kept fixed and can be calculated in an offline pre-training stage. In contrast, the negative class is changing over time and can therefore not be described by a finite set of samples. However, this information can be extracted online from the scene, that is, from the patch corresponding to the grid-element, as described before. Thus, for each feature  $f_j$  we can estimate a distributions  $D_j^-$  online over time. Using these distributions, we can define a weak classifier as illustrated in Fig. 13.8.

For the feature selection process, the errors of the features (weak classifiers) have to be calculated to select the best weak classifier within a selector. Since only negative updates are performed during online learning, only the error for negative samples  $\varepsilon_{\text{online}}$  can be estimated. However, since the distributions  $D_j^+$  were pre-estimated, we can also provide an error for the positive samples:  $\varepsilon_{\text{offline}}$ . Thus, instead of Eq. (13.6) we can use the combined error

$$\varepsilon = \frac{1}{2} (\varepsilon_{\text{+}}^{\text{offline}} + \varepsilon_{\text{-}}^{\text{online}}) \quad (13.19)$$

to select the best weak classifier within the selector.

Overall, this formulation has two main benefits. On the one hand, we can combine offline and online learning very easily by analyzing distribution on the feature

level. On the other hand, compared to typical update schemes such as self-training (e.g., [24, 30]) and co-training (e.g., [6, 22]), we do not rely on a direct feedback of the current classifier. Taking into account both, errors are not accumulated but are fading out, ensuring a stable system, even over a long period of time, which is demonstrated in Sect. 13.5.2.2.

### 13.4.2 Inverse MIL Updates

Even though the updates generated by the rules discussed before are correct most of the time, they might be wrong if an object is not moving over a long period of time. Hence, foreground information is used to perform negative updates, which causes the positive information to be temporally unlearned. This can be seen in the context of ambiguously labeled samples, which can be resolved with ideas from Multiple Instance Learning.

In particular, we apply the Inverse MILBoost described in Sect. 13.2.3. Since in the classifier grid scenario, the positive samples are well defined (positive samples are hand labeled) and the ambiguity concerns only the negative samples (coming directly from the scene without labeling), the original MIL idea makes only little sense. Therefore, as already defined in Eqs. (13.14) and (13.15), we modify the definition of bags such that the negative bags  $B_i^-$  need only to contain one negative example whereas the positive bag  $B_i^+$  consists only of positive examples. To generate the required negative bags, we collect a stack of input images from the image sequence over time, which we refer to as “temporal bag”. Having a large stack assures that the assumption for the negative bag containing at least one negative sample is mostly valid, since the probability that an object stays at one specific location over a longer period of time is very low.

Collecting a large stack of input images is adversarial for the runtime behavior. In order to avoid a large stack of input images within the temporal bag, we propose to use a small set of background images which operate on different time scales, which means that they are updated in different time intervals. Any kind of background model can be used. However, in our case we apply the approximated median background model [27]. Since these background models are updated in different time intervals, even a small number ensures that the temporal bags fulfill the MIL constraints. To give enough adaptivity to changing illumination conditions, we have background models which are updated in small time intervals, while to avoid objects staying at the same position for a while to become part of the background we have background models updated in long time intervals. Hence, the multiple instance learning property of inherently dealing with ambiguity in data can be exploited for improving the classifier grid approach and avoiding short-term drifting.

### 13.4.3 Robust Positive Updates by Co-training

The main focus of the approaches presented in Sects. 13.4.1 and 13.4.2 was to ensure stability over time. In this way, the system is able to adapt to changing environmental



**Fig. 13.9** Different illumination conditions with the corresponding background subtracted images. Even in case of totally different illumination conditions and differently appearing objects the background subtracted image gives a similar representation of the object

conditions, however, no new positive, object-specific information is gained. In the following, we introduce an update scheme based on co-training, which allows for adding new information in a robust way.

Co-training [6], in general, exploits the redundancy of unlabeled input data. The main idea is to train two conditionally independent classifiers  $h_1$  and  $h_2$  on some labeled data  $\mathcal{D}^L$  and then let these classifiers update each other using unlabeled data  $\mathcal{D}^U$ . An update is performed if one classifier is confident on a sample whereas the other one is not. Since Abney [1] showed that co-training classifiers aims at minimizing the error on the labeled samples while increasing the agreement on the unlabeled data, it is clear that the unlabeled data can help to improve the margin of the classifiers and to decrease the generalization error. The strong condition of conditionally independent classifiers was later relaxed by several authors (e.g., [1, 5]). Wang and Zhou [44] provided a PAC-style proof that co-training can converge to good accuracy if the classifiers are strong and highly uncorrelated. Thus, co-training can typically also be applied if the learners are strong and low-correlated. Thus, co-training has recently become popular in the field of computer vision and was applied for a variety of applications including background modeling [48], learning an object detector [22], or tracking [25].

For our purpose, we adopt the ideas of Levin et al. [22], who ran co-training on two boosted offline classifiers. Similarly, we train one classifier from gray-value images whereas the other is trained from background-subtracted images (i.e., approximated median background model [27]). However, in contrast, we run co-training only in an initial phase to adapt the main classifier to the scene and apply a more sophisticated online learner. In fact, after a settling phase we keep the classifier fixed and use it only as an oracle for two reasons. First, not all situations can be handled by co-training in a robust manner. Hence, if too many wrong updates are performed the co-trained classifier would start to drift and finally fail totally. Second, as illustrated in Fig. 13.9, most environmental changes are eliminated by the background subtraction and the variability in the positive class vanishes. Thus, no further information can be gained.

**Algorithm 5** Co-grid initialization

---

**Input:** grid-classifier  $G_j^{t-1}$ , classifier  $C^{t-1}$

**Input:** grid-element  $\mathbf{X}_j$ , BGS patch  $\mathbf{B}_j$

```

1: if  $C^{t-1}(\mathbf{B}_j) > \theta$  then
2:    $update(G_j^{t-1}, \mathbf{X}_j, +)$ 
3: else if  $C^{t-1}(\mathbf{B}_j) < -\theta$  then
4:    $update(G_j^{t-1}, \mathbf{X}_j, -)$ 
5: end if
6: if  $G_j^{t-1}(\mathbf{X}_j) > \theta$  then
7:    $update(C^{t-1}, \mathbf{B}_j, +)$ 
8: else if  $G_j^{t-1}(\mathbf{X}_j) < -\theta$  then
9:    $update(C_{t-1}, \mathbf{B}_j, -)$ 
10: end if

```

**Output:** grid-classifier  $G_j^t$ , classifier  $C^t$

---

In general, any online learner can be applied within the co-grid approach. However, to increase the robustness (the co-trained oracle may still suffer from a small amount of label noise), we apply online TransientBoost as introduced in Sect. 13.3.2, which allows to combine reliable (labeled) data with unreliable data (unlabeled from the scene). Thus, robustly new positive information can be gained, especially increasing the recall but preserving the accuracy. In our case, we apply three classes: one for the reliable positive data (+1), one for the possibly unreliable positive data (+2), and one for the classifier-specific background (−1). The class +1 is trained offline and is kept fix whereas the others are updated online over time. In the following, the two main steps are summarized more detailed.

**Co-training Stage** During the initial stage our system is trained in a co-training manner. Given  $n$  grid classifiers  $G_j$  operating on gray level image patches  $\mathbf{X}_j$  and one compact classifier  $C$  operating in a sliding window manner on background subtracted images  $\mathbf{B}$ . To start co-training, the classifiers  $G_j$  as well as the classifier  $C$  are initialized with the same offline pre-trained classifier, then the classifiers co-train each other. A confident classification (positive and negative) of a classifier  $G_j$  is used to update the classifier  $C$  with the background subtracted representation at position  $j$ . Vice versa, a confident classification of classifier  $C$  at position  $j$  generates an update for classifier  $G_j$ . Due to the offline pre-trained prior, already capturing the generic information, only a small number of updates is sufficient to adapt the classifiers to a new scene or changing environmental conditions. The update procedure during the initialization for a specific grid element  $j$  is summarized in Algorithm 5.

**Detection Stage** After the initial stage, as described above, the classifier  $C$  is not longer updated and is applied as an oracle to generate new positive and negative

**Algorithm 6** Co-grid update

---

**Input:** grid-classifier  $G_j^{t-1}$ , classifier  $C$   
**Input:** grid-element  $\mathbf{X}_j$ , BGS patch  $\mathbf{B}_j$

```

1: if  $C(\mathbf{B}_j) > \theta$  then
2:    $\forall i : update(G_i^{t-1}, \mathbf{X}_j, +)$ 
3: end if
4: if  $C(\mathbf{B}_j) < -\theta$  then
5:    $update(C_j^{t-1}, \mathbf{X}_j, -)$ 
6: end if
```

**Output:** grid-classifier  $G_j^t$

---

samples. Hence, we can abstain from fixed update rules, which are broadly used for Classifier Grids. Moreover, we perform negative updates for the classifiers  $G_j$  only if they are necessary, that is, if the scene is changing. Even if the oracle classifier  $C$  has a low recall, the precision is very high. Thus, only very valuable updates are generated, increasing the performance of the classifiers  $G_j$ . In particular, a confident classification result of classifier  $C$  at position  $j$  generates an update for all classifiers  $G_i$ ,  $i = 1, \dots, n$ . In this way, new scene specific positive samples are disseminated over the whole classifier grid. Negative updates are performed for classifiers  $G_j$  if there is no corresponding detection reported at this position for classifier  $C$ . The update procedure during the detection phase for a specific grid element  $j$  is summarized in Algorithm 6.

## 13.5 Experimental Results

The experimental results are split into three parts. First, we give an evaluation on scene specific detectors trained using Conservative Learning in Sect. 13.5.1. Then, in Sect. 13.5.2, we give a detailed evaluation on grid-based detectors. Finally, in Sect. 13.5.3 we compare both approaches to state-of-the-art methods for person detection. Even not limited to this specific setups, for all experiments we build on the online learning methods presented in Sect. 13.2 and use Haar-like features as underlying representation. Moreover, as we are dealing with stationary cameras we take into account a simple scene calibration, which allows for reducing the computational effort and also suppressing a number of false positives.

To allow for a fair quantitative evaluation, we build our results on precision  $Pr = TP/(TP+FP)$  and recall  $R = TP/P$ , where  $TP$  is the number of true positives,  $FP$  is the number of false positives, and  $P$  is the number of positives in the test data represented by the given ground-truth. The positives are estimated according to the overlap criterion [2], requiring a minimal overlap of 50 %. Once these parameters are estimated, we can plot recall-precision curves (RPC) [2]. Additionally, we estimate the F-measure  $FM = (2 \cdot R \cdot Pr)/(R + Pr)$  [2], which is the harmonic mean

between *recall* and *precision*. In particular, the characteristics given in this section are estimated such that the F-measure is maximized.

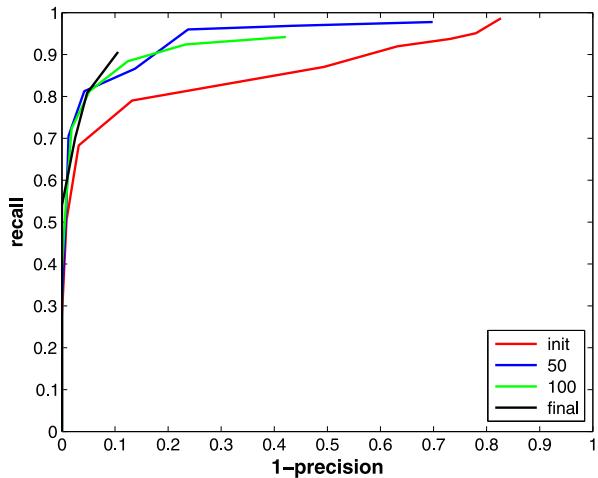
### 13.5.1 Scene Specific Classifiers

In the following, we analyze different aspects of Conservative Learning, where the experimental setup is as follows. We use a boosted classifier consisting of 50 selectors each holding 100 weak classifiers. If not stated otherwise, the initial classifier used for all setups is the same and is trained using 50 positive and negative samples, respectively. This classifier is then online trained and the finally obtained classifier is evaluated on an independent test sequence, which was not seen before during training but steaming from the same camera setup. To monitor the progress during online training, a classifier is saved after a pre-defined number of training samples. Thus, the obtained classifiers are evaluated on an independent test sequence and the precision, the recall, and the F-measure are estimated. Hence, we can plot learning curves that show the performance of an online learned classifier over time.

For these experiments, we generated the *CoffeeCam* dataset showing a corridor in a public building near to a coffee dispenser. Hence, we capture various representative real-world scenarios: walking people, people standing around while waiting for their coffee, or people building small crowds while drinking coffee. To create this dataset, we have recorded images over several days. In total, we have recorded over 35000 (gray-value) images, which have a resolution of  $192 \times 238$ . For evaluation purposes, we generated a training sequence containing 1200 frames and a challenging independent test set (containing groups of persons, persons partially occluding each other, and persons walking in different directions) and a corresponding ground-truth. In total, the test sequence consists of 300 frames and contains 224 person. The resulting precision-recall curves for the initial and the final classifier as well as for time steps 50 and 100 are shown in Fig. 13.10.

It can be seen that the performance of the initial classifier can be improved and gets saturated already after a small number (i.e., 50–100) of updates. The same behavior can also be recognized from Fig. 13.11, where precision, recall and F-measure are plotted individually over time. In contrast to Fig. 13.10, this clearly reveals that the main benefits arise from improving the precision. In fact, the recall is already on a satisfying level from the very beginning. Moreover, we also show the effect of using initial classifiers trained using a different number of labeled samples. Figure 13.11(a) depicts the performance curves if 50 labeled samples were used to train the initial classifier. One can see a clear improvement (especially in the first 100 steps), where most of the false positives can be eliminated. Later, if all scene-specific information is acquired the performance gets saturated (around step 500). The same behavior can be recognized from Fig. 13.11(b), which depicts the performance curves if only 10 labeled samples were used to train the initial classifier. Hence, there is a greater number of false positives in the beginning, but finally we get the same performance as for the first test case!

**Fig. 13.10** Precision-recall curves for Conservative Learning for different time steps: initial classifier, after 50 frames, after 100 frames, and final classifier



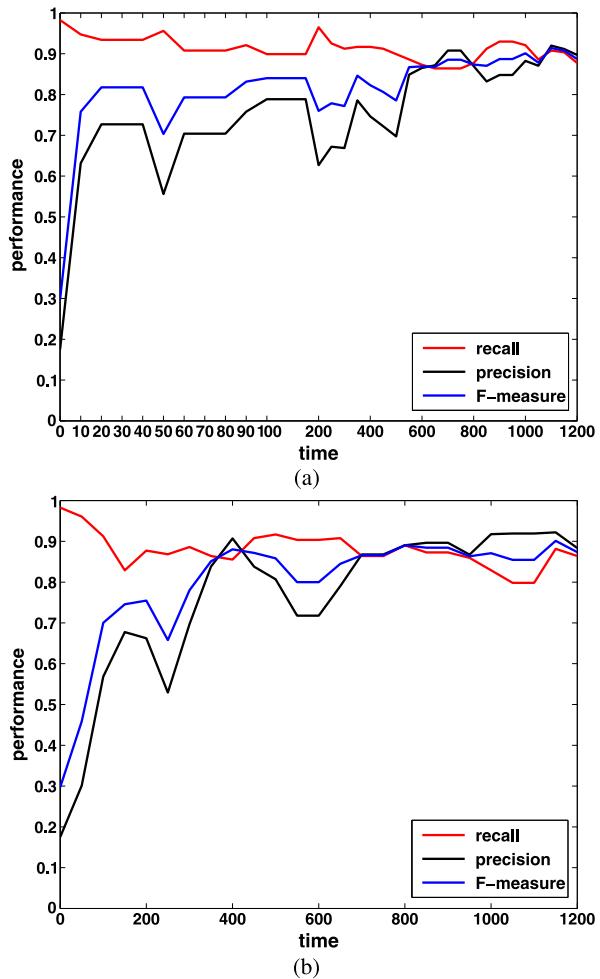
The same performance increase over time can also be seen from the illustrative detection results in Fig. 13.12 for three different online classifiers (initial, after 300, and after 1200 training frames). There are many false positives in the beginning, which are reduced via the online updates and are completely vanishing in the end.

To get a benchmark, we compared this final classifier to two commonly used person detection methods, that is, Dalal and Triggs [7] and Viola and Jones [41]. In particular, we show the detection characteristics that is, recall, precision, and F-measure, which are summarized in Table 13.1. It can be seen, even seeming to be a simple scenario, that the generic detectors yield an insufficient detection performance. In fact, these results show that on the same accuracy level the recall is much lower. Further illustrative detection results for the Conservative Learning are shown in Fig. 13.13.

Next, after we have shown that using the proposed update strategy the detection performance can be improved, we would like to demonstrate that in particular the active sampling strategy is beneficial. For that purpose, we train two detectors in parallel and run the same experiment as before. The first detector was trained using Conservative Learning described in Sect. 13.3. The second detector was trained in the same way, however, the negative samples were randomly selected from the background. Hence, we will further refer to this method as “Random Sampling”. To avoid a bias of the positive samples due to differently online trained detectors, the positive updates are generated via tracking. Hence, to avoid tracking confusion due to overlapping instances, for this experiments we created a single face detection setup, once again consisting of independent training and test sequences each consisting of 250 frames. The corresponding results are shown in Fig. 13.14.

From Fig. 13.14(a), it can be seen that for Conservative Learning less than 100 updates are sufficient to derive precision, recall, and hence F-measure of more than 0.90. In contrast, Fig. 13.14(b) shows that Random Sampling requires a longer settling phase. Even though a trend of increasing accuracy can be observed after approximate 150 updates the precision is still insufficient and the precision curve

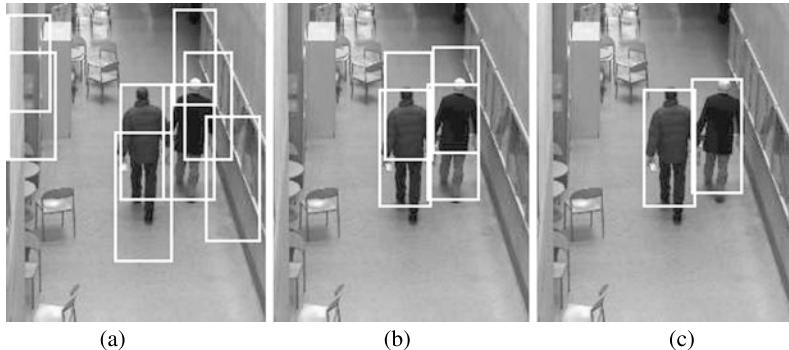
**Fig. 13.11** Improvement of online classifier: (a) started from 50 labeled samples and (b) started from 10 labeled samples



shows a stochastic trend. But, in fact, compared to Conservative Learning we have a higher recall. However, both experiments clearly indicate the benefits of the proposed approach: (a) a small number of labeled samples is sufficient to finally obtain a performative classifier, (b) the active learning strategy allows for sampling “better” samples, and thus (c) only a small number of updates is necessary to adapt to the scene.

### 13.5.2 Classifier Grids

The experiments on Classifier Grids tackle four different aspects. First, in Sect. 13.5.2.1 we give a proof-of-concept study. Next, in Sect. 13.5.2.2 we demon-



**Fig. 13.12** Improvement of online classifier: **(a)** initial classifier, **(b)** after 300 frames and **(c)** after 1200 frames

**Table 13.1** Final classifier obtained via Conservative Learning benchmarked against state-of-the-art person detection on the *CoffeeCam* dataset

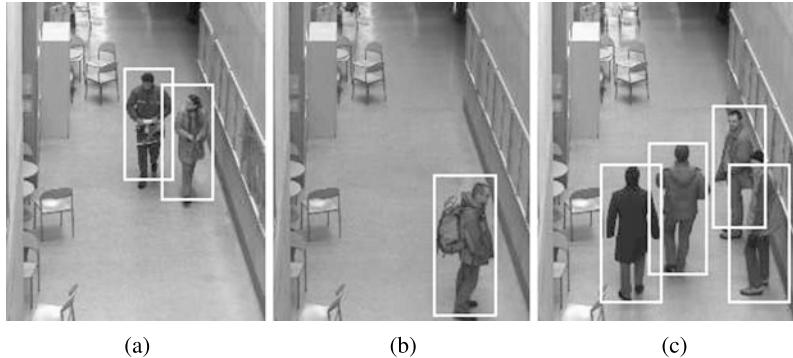
	Recall	Precision	F-measure
Cons. Learning	0.87	0.96	0.91
Dalal & Triggs	0.37	1.00	0.54
Viola & Jones	0.36	0.95	0.52

strate the key aspects of this approach, i.e., the long-term stability. Finally, in Sects. 13.5.2.4 and 13.5.2.3 we show the benefits of two extensions in sense of increasing the recall and avoiding short term drifting.

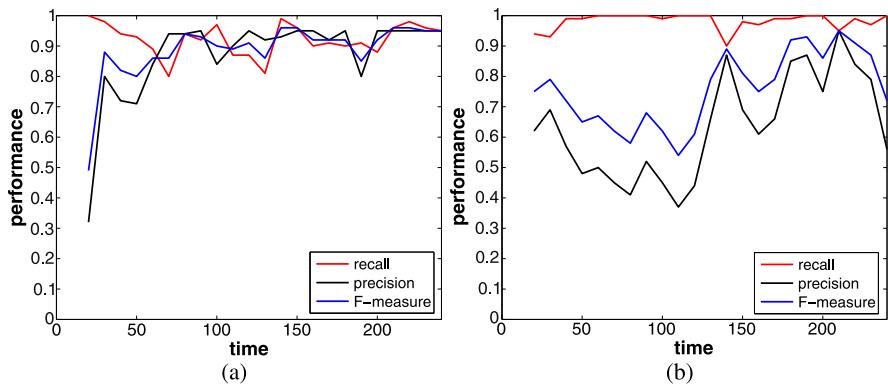
### 13.5.2.1 Not yet Another Background Model

When reviewing Classifier Grids it might seem that they are just “some kind of background model” or a “little bit more sophisticated template matching”. Thus, the following experiment faces two goals. First, to alleviate these arguments and, second, to illustrate that even using this quite simple strategy competitive results can be obtained. For that purpose, we generated a challenging pedestrian detection benchmark showing a corridor of a public building (*Corridor sequence*). The main difficulties arise from various moving objects (e.g., a ball, chairs, and an umbrella) and partly overlapping persons, situations that can neither be captured by a simple background model nor by a template matcher. The sequence consisting of 300 frames was taken at a resolution of  $320 \times 240$  and contains 296 persons in total.

Hence, we compared the Classifier Grid approach to low level methods (i.e., a simple background model (BGM) [27], template matching (TM), and a combination of both (TM+BGM), which might be considered a simple pendent to our method). Additionally, to get a benchmark, we compare these results to two generic state-of-the-art person detectors (i.e., Dalal and Triggs (D&T) [7] and deformable



**Fig. 13.13** Illustrative detection results from Conservative Learning on the *CoffeeCam* dataset



**Fig. 13.14** Importance of negative sampling: **(a)** Active Sampling ensures a faster adaption to the problem and finally yields robust results compared to **(b)** Random Sampling

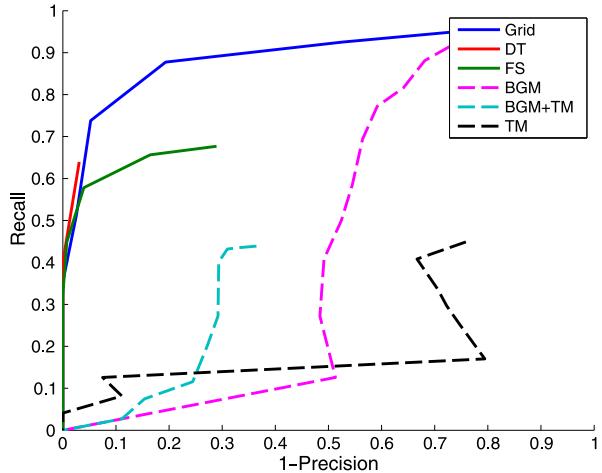
part model of Felzenszwalb et al. (FS) [10]). The thus obtained results are summarized in Fig. 13.15.

From these results, it can be seen that the generic detectors show an excellent precision, but the recall very low. Moreover, the low level cues totally fail, that is, for BM and TM either the recall or the precision is very poor such that even a combination of both (TM+BGM) yields insufficient results. Hence, we can conclude that these approaches are too weak to provide sufficient detection results. In contrast, the proposed approach yields competitive detection results, even using only very simple update rules and compact classifiers. Finally, typical results of the Classifier Grid approach are depicted in Fig. 13.16.

### 13.5.2.2 Long-Term Behavior

As illustrated in Fig. 13.18, within realistic setups we have to cope with heavily changing environmental conditions, which have to be handled by the system. Thus,

**Fig. 13.15** Precision-recall curves for Classifier Grids, simple baselines, and state-of-the-art methods for the *Corridor Sequence*



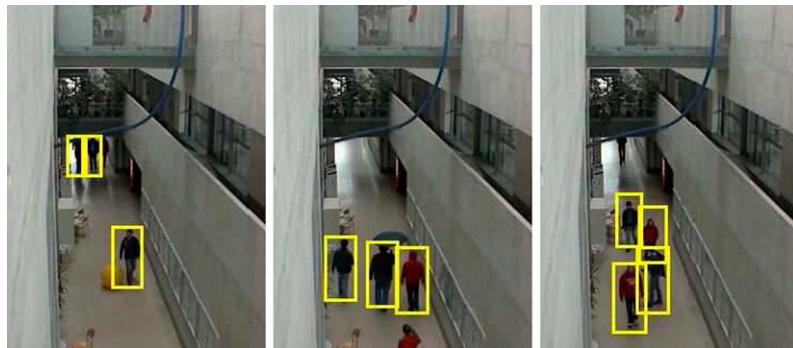
a main challenge of an online system that is learning 24 hours a day and 7 days a week is to guarantee the stability over time. Therefore, we demonstrate the long-term behavior of the Classifier Grid method by running it for a whole week. In fact, during a period of 7 days we updated our system 580,000 times (i.e., 1 fps). To show that the systems' performance is unchanged over time we provide the evaluations results on four different points in time. Each of these sequences consists of 2,500 frames, corresponding to approx. 40 minutes of video data. The details of the experimental setup are summarized in Table 13.2.

From the results shown in Fig. 13.17, it can be seen that the method is stable over time. The slightly variations in the curves can be explained by the different levels of complexity for the four sequences (i.e., number of persons, density of persons, etc.). These results are in particular of interest considering the significantly changing conditions we had to deal with during these 7 days (i.e., natural light, artificial lighting, inadequate lighting, etc.), as also illustrated in Fig. 13.18. Hence, these drastically changing conditions, which can be handled by our system considerably better than by other methods, clearly show the requirement of an adaptive system.

### 13.5.2.3 IMIL Grids

After we demonstrated the long-term stability of Classifier Grids, we address short-time drifting if an object might not move for a longer period of time. This problem is illustrated in Fig. 13.19(a).

Thus, in the following we demonstrate that this problem can be alleviated by using the extension introduced in Sect. 13.4.2. For that purpose, we generated two test sequences containing also non-moving objects, *Corridor-Standing* and *Vehicle-Standing*, and compared the IMIL to the standard Classifier Grid approach. The first sequence showing a corridor in a public building consists of 900 frames ( $640 \times 480$ )

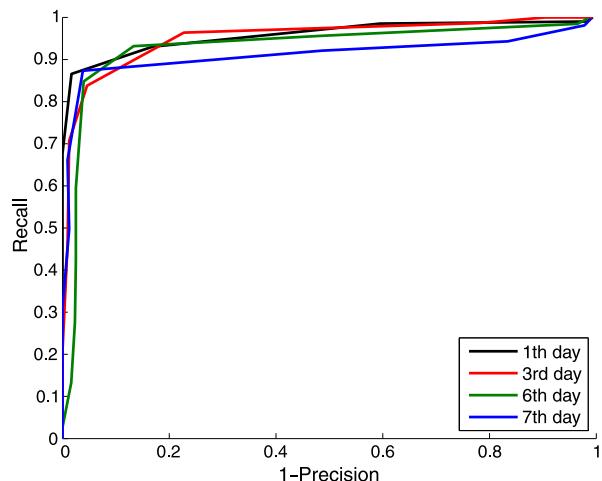


**Fig. 13.16** Illustrative detection results of the grid-based person detector for the *Corridor Sequence*

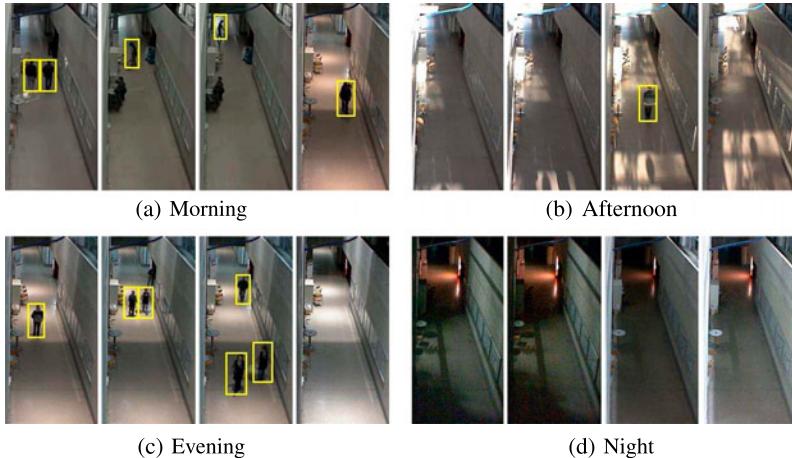
**Table 13.2** Description of the selected sequences of the long-term experiment

	# updates yet performed	# persons
1st day	3,390	201
3rd day	179,412	222
6th day	484,891	454
7th day	577,500	316

**Fig. 13.17** Recall-precision curves for the *long-term experiment* at different points in time



containing 2491 persons, which are staying at the same position over a long period of time. The second one consists of 500 frames ( $720 \times 576$ ), containing 2375 cars. One car broke down within this sequence and is standing at the same position for 400 frames. The obtained recall-precision curves, again for the proposed and the original CG approach, are shown in Fig. 13.20.



**Fig. 13.18** Illustrative detection results of the grid-based person detector obtained during to long-term experiment showing the difficulties over time

Since due to the IMIL formulation we get rid of short-term-drifting for both scenarios, the recall (at a reasonable precision level) can be significantly improved. This is also illustrated in Fig. 13.19, where the first row shows detection results of the original Classifier Grid approach, whereas the second row shows detection results using the proposed inverse multiple-instance learning. It can clearly be seen that the person on the right side, standing at the same position over 175 frames, is detected by the proposed approach whereas it is not in the other case.

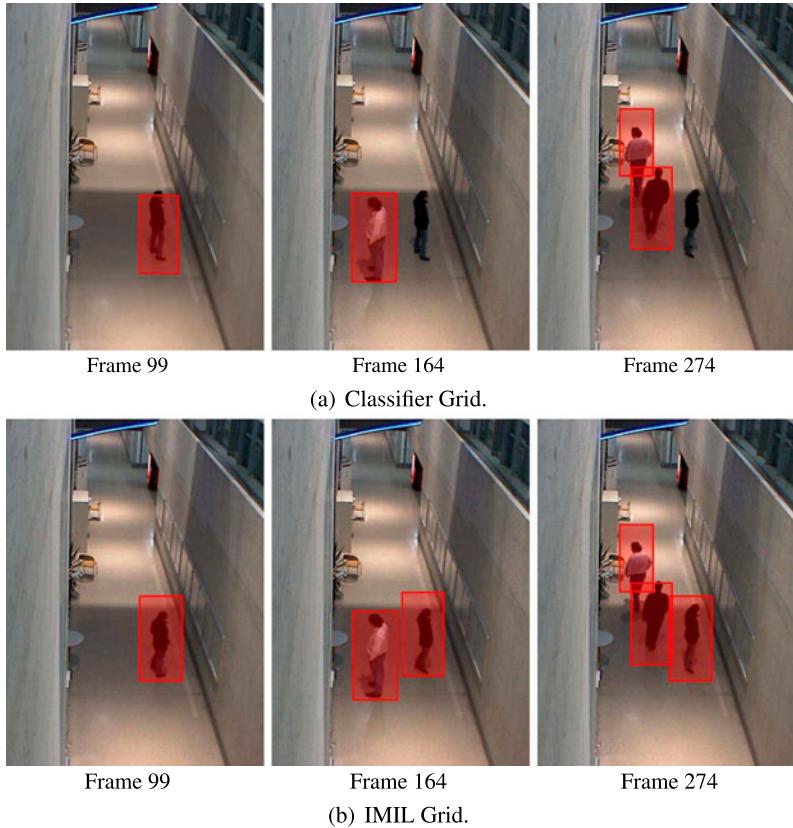
#### 13.5.2.4 Classifier Co-grids

Up to now the goal was to establish an efficient, robust object detector, ensuring both, short- and long-time stability. However, in this way we can only guarantee that the accuracy is preserved, but no new (positive) information can be gained. Thus, in the following we show that using the update scheme introduced in Sect. 13.4.3 also the recall can be increased at the same accuracy level.

We evaluated our approach on two publicly available standard benchmark datasets for pedestrian detection, i.e., PETS 2006,<sup>2</sup> and for car detection, that is, AVSS 2007.<sup>3</sup> The PETS 2006 dataset shows the concourse of a train station from four different views, having a resolution of  $720 \times 576$ . For our experiments we adapted a sequence from Camera view 4 consisting of 308 frames, which contains 1714 persons. The AVSS 2007 dataset is adopted from the i-LIDS Bag and Vehicle Detection Challenge (AVSS 2007). In particular, we run our approach

<sup>2</sup><http://www.cvg.rdg.ac.uk/PETS2006/> (November 30, 2012).

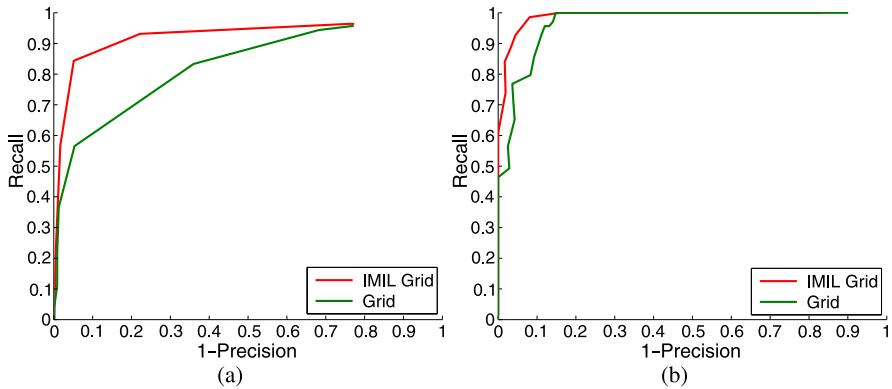
<sup>3</sup>[http://www.eecs.qmul.ac.uk/~andrea/avss2007\\_d.html](http://www.eecs.qmul.ac.uk/~andrea/avss2007_d.html) (November 30, 2012).



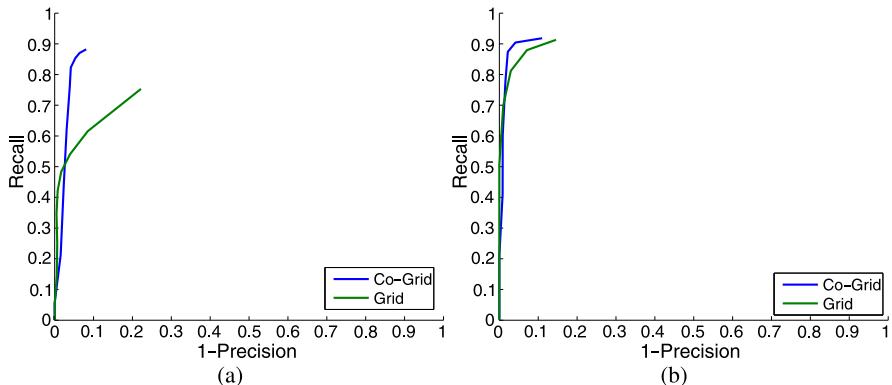
**Fig. 13.19** Temporal information incorporation by MIL avoids short-term drifting. The original Classifier Grid approach (*first row*) temporary drifts after about 60 frames whereas the proposed approach (*second row*) avoids temporal drifting even after more than 170 frames

on the first 500 frames ( $720 \times 576$  pixels) from the sequence AVSS\_PV\_Hard, which contains 673 cars. For both approaches, we also generated a corresponding groundtruth.

The estimated recall-precision curves are shown in Fig. 13.21. For the PETS scenario, it can be seen that additionally using object-specific information significantly increases (by 0.3), whereas there is only a little gain for the AVSS 2007 sequence (by 0.1). However, for both scenarios, as intended, the precision is preserved. The different amount of performance gain can be simply described by the nature of the tasks, showing significantly different variability within the positive class. Whereas persons can have a different appearance (frontal, back, or side views), having a stationary camera the cars are mainly shown from frontal and back views. Thus, also the baseline for the car detector is much better. Nevertheless, these experiments show that including scene-specific data is highly beneficial for the given tasks.



**Fig. 13.20** Recall-precision Curves for (a) *Corridor-Standing* and (b) *Vehicle-Standing* sequences for the original Classifier Grid and the IMIL Grid approach



**Fig. 13.21** Increasing recall by using classifier co-grids: (a) *PETS 2006* and (b) *AVSS 2007*

### 13.5.3 Person Detection

Finally, after analyzing both, the scene specific and the grid approach, in detail, we would like to give a more general evaluation for the task of person detection.<sup>4</sup> Please note, the main emphasis of this experiment is to illustrate that using the proposed online learning strategies competitive detectors can be trained even if only a very limited amount of training data is available and not to learn a perfect person detector. In fact, the approaches are more general and can be also applied for different objects. For a more detailed study on person detection, we would refer to [9].

<sup>4</sup>This particular task was chosen as implementations of existing approaches as well as a number of benchmark datasets are publicly available.

**Table 13.3** Person detection results for the *Corridor Sequence*

	Recall	Precision	F-measure
Cons. Learning	0.84	0.94	0.89
Class. Grids	0.88	0.83	0.85
Dalal and Triggs	0.63	0.97	0.76
Viola & Jones	0.00	0.00	0.00
Felzenszwalb et al.	0.61	0.83	0.70

In particular, we give a comparison to the approaches of Dalal and Triggs [7], Viola and Jones [41], and the deformable parts model of Felzenszwalb et al. [10]. All three detectors are general person detectors, which, in contrast to Conservative Learning and the Grid Classifiers, do not use any previous knowledge about the scene. To enable a fair comparison in a post-processing step all detections were removed, that do not fit to the estimated ground-plane. In fact, a detection was removed if the scale was smaller than 50 % or greater than 150 % of the expected patch-size. This post-processing does not reduce the recall since the removed detections would have been counted as false positives otherwise. Considering the evaluation setup, the grid classifiers (standard version as introduced in Sect. 13.4.1) are updated online whereas for Conservative Learning an independent test sequence was used to train a classifier, which is kept fixed later on.

**Corridor Sequence** Before showing results on publicly available datasets, we first show results for the Corridor Sequence introduced in Sect. 13.5.2.1. This sequence is in particular of interest since a lot of other objects are moving within the scene and there are severe occlusions due to the limited space. Both are typical problems making online learning much more difficult. The corresponding results are given in Table 13.3, which reveal that under the given difficulties the approaches additionally exploiting scene information yield a significantly better recall, that is, plus 0.25 (!), at a comparable or even better precision level.

**Caviar** Next, we run experiments on the *Caviar* dataset<sup>5</sup> showing a corridor in a shopping mall, taken at a resolution of  $384 \times 288$ . For our experiments, we have selected one typical (more complex) sequence for evaluation (*ShopAssistant2cor*). The frame-rate of the original sequence was reduced and the images were converted to gray-scale. In total, the test sequence consists of 144 frames and contains 364 persons. The provided ground-truth, which also contains partial persons (hands, head, etc.), was adapted such that only persons are included, that are detectable. For training, the Conservative Learning classifier additionally a training sequence of 1200 frames was used, that was compiled from different sequences.

The estimated characteristics are summarized in Table 13.4. Again we can recognize the same trend that Conservative Learning and Classifier Grids yield a sig-

<sup>5</sup><http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1> (November 30, 2012).

**Table 13.4** Person detection results for the *Caviar* dataset

	Recall	Precision	F-measure
Cons. Learning	0.81	0.92	0.86
Class. Grids	0.76	0.88	0.82
Dalal & Triggs	0.42	0.98	0.59
Viola & Jones	0.32	0.85	0.46
Felzenszwalb et al.	0.55	0.93	0.69

**Table 13.5** Person detection results—*PETS 2006* dataset, Camera 4

	Recall	Precision	F-measure
Cons. Learning	0.79	0.94	0.86
Class. Grids	0.78	0.78	0.78
Dalal & Triggs	0.43	0.95	0.60
Viola & Jones	0.16	0.91	0.28
Felzenszwalb et al.	0.73	0.86	0.79

nificantly higher recall (plus 0.20–0.50) compared to the generic classifiers. Representative detection results of our approaches are shown in Fig. 13.22.

**PETS 2006** Finally, the last experiment was run on the PETS 2006 dataset, already introduced in detail in Sect. 13.5.2.4. Again, for Conservative Learning additionally a training sequence consisting of 650 frames was generated. From Table 13.5, we can recognize the same trend as for the previous two experiments, however, in this case the approach of Felzenszwalb et al. also yields results comparable to the approaches exploiting scene-specific information. Nevertheless, from these different setups we can clearly conclude that additionally exploiting scene information can help to (significantly) improve the detection performance. Again, illustrative detection results for our approaches are shown in Fig. 13.23.

## 13.6 Conclusion

In this chapter, we addressed the problem of scene-specific object detection. In particular, by exploiting the fact that many object detection scenarios build on stationary cameras we can drastically ease the task for two main reasons: (a) the complexity of the task is reduced (b) an online classifier can be used and not all variability over time has to be captured. Both allow for using less complex and thus more efficient classifiers. Moreover, also the detection performance can be improved. In particular, we introduced two approaches working on a different scale: Conservative Learning—working on frame level and Classifier Grids—working only at one specific image position.



**Fig. 13.22** Person detection results—*Caviar* dataset



**Fig. 13.23** Person detection results—*PETS 2006* dataset, Camera 4

The main idea of Conservative Learning is to adopt active learning for the task of scene specific object detection. In this way, the most valuable samples are robustly identified, which allow for adapting an offline pre-trained classifier to a new scene as well as to changing environmental condition. To go one step further, Classifier Grids even further decrease the problem's complexity by reducing the problem to discriminate an object from the background on one particular position. Moreover, due to the used update strategy long-term stability can be assured, which is highly beneficial for real-world applications. Additionally, two extensions were discussed that allow for including further positive information and for further increasing the robustness.

The experimental results show both, a proof-of-concept and competitive results compared to state-of-the-art methods in the field of person and car detection. However, the approaches are quite general and can also be applied for different tasks. Even though not limited to this specific learner, both approaches build on online GradientBoost. In fact, the approach is very efficient and can easily be adapted for different task as the actually used loss function can be defined according to the problems requirements. Future work would include the usage of more efficient and

more sophisticated features and to adopt successful developments in the context of Classifier Grids also for Conservative Learning.

**Acknowledgements** The work was supported by the Austrian Science Foundation (FWF) project Advanced Learning for Tracking and Detection in Medical Workflow Analysis (I535-N23) and by the Austrian Research Promotion Agency (FFG) projects SHARE (831717) in the IV2Splus program and MobiTrick (8258408) in the FIT-IT program.

## References

1. Abney S (2002) Bootstrapping. In: Proc annual meeting of the association for computational linguistics, pp 360–367
2. Agarwal S, Awan A, Roth D (2004) Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans Pattern Anal Mach Intell* 26(11):1475–1490
3. Andrews S, Tschantzidis I, Hofmann T (2003) Support vector machines for multiple-instance learning. In: Advances in neural information processing systems, pp 561–568
4. Babenko B, Yang M-H, Belongie S (2009) Visual tracking with online multiple instance learning. In: Proc IEEE conf on computer vision and pattern recognition
5. Balcan M-F, Blum A, Yang K (2004) Co-training and expansion: towards bridging theory and practice. In: Advances in neural information processing systems, pp 89–96
6. Blum A, Mitchell T (1998) Combining labeled and unlabeled data with co-training
7. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: Proc IEEE conf on computer vision and pattern recognition
8. Dieterich TG, Lathrop RH, Lozano-Pérez T (1997) Solving the multiple instance problem with axis-parallel rectangles. *Artif Intell* 89(1–2):31–71
9. Dollár P, Wojek C, Schiele B, Perona P (2011) Pedestrian detection: an evaluation of the state of the art. *IEEE Trans Pattern Anal Mach Intell* 34(4):743–761
10. Felzenszwalb P, McAllester D, Ramanan D (2008) A discriminatively trained, multiscale, deformable part model. In: Proc IEEE conf on computer vision and pattern recognition
11. Freund Y, Schapire RE (1995) A decision-theoretic generalization of on-line learning and an application to boosting. In: Proc European conf on computational learning theory, pp 23–37
12. Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. *J Comput Syst Sci* 55:119–139
13. Friedman J (2001) Greedy function approximation: a gradient boosting machine. *Ann Stat* 29(5):1189–1232
14. Friedman J, Hastie T, Tibshirani R (2000) Additive logistic regression: a statistical view of boosting. *Ann Stat* 28(2):337–374
15. Goldberg AB, Li M, Zhu X (2008) Online manifold regularization: a new learning setting and empirical study. In: Proc European conf on machine learning and knowledge discovery in databases, vol I, pp 393–407
16. Grabner H, Bischof H (2006) On-line boosting and vision. In: Proc IEEE conf on computer vision and pattern recognition
17. Grabner H, Roth PM, Bischof H (2007) Is pedestrian detection really a hard task? In: Proc IEEE workshop on performance evaluation of tracking and surveillance
18. Hoiem D, Efros AA, Hebert M (2006) Putting objects in perspective. In: Proc IEEE conf on computer vision and pattern recognition
19. Javed O, Ali S, Shah M (2005) Online detection and classification of moving objects using progressively improving detectors. In: Proc IEEE conf on computer vision and pattern recognition
20. Leibe B, Leonardis A, Schiele B (2008) Robust object detection with interleaved categorization and segmentation. *Int J Comput Vis* 77(1–3):259–289

21. Leistner C, Amir R, Saffari AA, Roth PM, Bischof H (2009) On robustness of on-line boosting—a competitive study. In: Proc IEEE on-line learning for computer vision workshop
22. Levin A, Viola P, Freund Y (2003) Unsupervised improvement of visual detectors using co-training. In: Proc IEEE int'l conf on computer vision
23. Li M, Sethi IK (2006) Confidence-based active learning. *IEEE Trans Pattern Anal Mach Intell* 28(8):1251–1261
24. Li L-J, Wang G, Fei-Fei L (2007) Optimol: automatic online picture collection via incremental model learning. In: Proc IEEE conf on computer vision and pattern recognition
25. Liu R, Cheng J, Lu H (2009) A robust boosting tracker with minimum error bound in a co-training framework. In: Proc IEEE int'l conf on computer vision
26. Mason L, Baxter J, Bartlett P, Frean M (1999) Functional gradient techniques for combining hypotheses. In: Advances in large margin classifiers. MIT Press, Cambridge, pp 221–247
27. McFarlane NJB, Schofield CP (1995) Segmentation and tracking of piglets. *Mach Vis Appl* 8(3):187–193
28. Nair V, Clark JJ (2004) An unsupervised, online learning framework for moving object detection. In: Proc IEEE conf on computer vision and pattern recognition
29. Park J-H, Choi Y-K (1996) On-line learning for active pattern recognition. *IEEE Signal Process Lett* 3(11):301–303
30. Rosenberg C, Hebert M, Schneiderman H (2005) Semi-supervised self-training of object detection models. In: IEEE workshop on applications of computer vision
31. Roth PM, Bischof H (2008) Conservative learning for object detectors. Machine learning techniques for multimedia. Springer, Berlin
32. Roth PM, Sternig S, Grabner H, Bischof H (2009) Classifier grids for robust adaptive object detection. In: Proc IEEE conf on computer vision and pattern recognition
33. Schapire RE (1990) The strength of weak learnability. *Mach Learn* 5(2):197–227
34. Schapire RE, Singer Y (2000) Boostexter: a boosting-based system for text categorization. *Mach Learn* 39(2/3):135–168
35. Skočaj D, Leonardis A (2008) Incremental and robust learning of subspace representations. *Image Vis Comput* 26(1):27–38
36. Sternig S, Godec M, Roth PM, Bischof H (2010) TransientBoost: on-line boosting with transient data. In: Proc IEEE online learning for computer vision workshop (in conj CVPR)
37. Sternig S, Roth PM, Bischof H (2012) On-line inverse multiple instance boosting for classifier grids. *Pattern Recognit Lett* 33(1):890–897
38. Tieu K, Viola P (2000) Boosting image retrieval. In: Proc IEEE conf on computer vision and pattern recognition, vol I, pp 228–235
39. Turtinen M, Pietikänen M (2005) Labeling of textured data with co-training and active learning. In: Proc workshop on texture analysis and synthesis, pp 137–142
40. Vapnik VN (1995) The nature of statistical learning theory. Springer, New York
41. Viola P, Jones MJ (2001) Rapid object detection using a boosted cascade of simple features. In: Proc IEEE conf on computer vision and pattern recognition
42. Viola P, Jones MJ, Snow D (2003) Detecting pedestrians using patterns of motion and appearance. In: Proc IEEE int'l conf on computer vision
43. Viola P, Platt JC, Zhang C (2005) Multiple instance boosting for object detection. In: Advances in neural information processing systems
44. Wei W, Zhou Z-H (2007) Analyzing co-training style algorithms. In: Proc European conf on machine learning, pp 454–465
45. Wu B, Nevatia R (2005) Detection of multiple, partially occluded humans in a single image by Bayesian combination of edgelet part detectors. In: Proc IEEE int'l conf on computer vision
46. Wu B, Nevatia R (2007) Improving part based object detection by unsupervised, online boosting. In: Proc IEEE conf on computer vision and pattern recognition
47. Yan R, Yang J, Hauptmann A (2003) Automatically labeling video data using multi-class active learning. In: Proc IEEE int'l conf on computer vision, vol I, pp 516–523
48. Zhu Q, Avidan S, Cheng K-T (2005) Learning a sparse, corner-based representation for background modelling. In: Proc IEEE int'l conf on computer vision, vol I, pp 678–685

# Chapter 14

## Video Temporal Super-resolution Based on Self-similarity

Mihoko Shimano, Takahiro Okabe, Imari Sato, and Yoichi Sato

**Abstract** We introduce a method for making temporal super-resolution video from a single video by exploiting the self-similarity that exists in the spatio-temporal domain of videos. Temporal super-resolution is an inherently ill-posed problem because there are an infinite number of high temporal resolution frames that can produce the same low temporal resolution frame. The key idea in this work is exploiting self-similarity for solving this ambiguity. Self-similarity means self-similar motion blur appearances that often reappear at different temporal resolutions. Several existing methods generate plausible intermediate frames by interpolating input frames of a captured video, which has frame exposure time shorter than inter-frame period. In contrast with them, our method can increase the temporal resolution of a given video in which frame exposure time equals to inter-frame period, for instance, by resolving one frame to two frames.

### 14.1 Introduction

There is a great demand for video enhancement to increase temporal resolution by generating a high-frame rate video from a given video captured at low frame rate. Several methods [3, 6, 9, 10] based on temporal interpolation have been proposed

---

M. Shimano (✉) · T. Okabe · Y. Sato

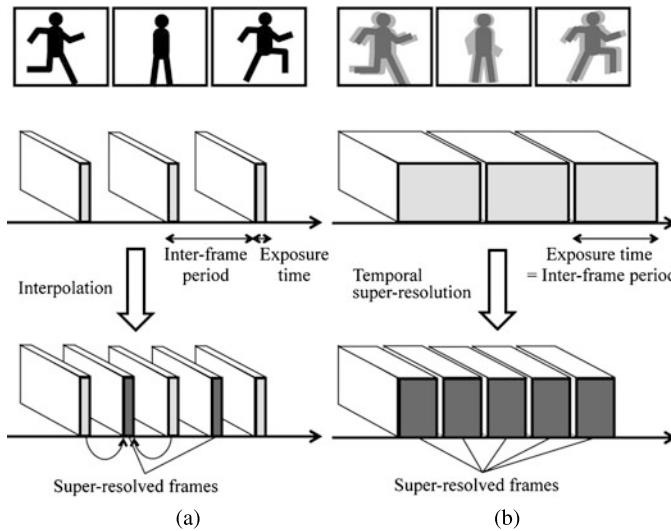
The University of Tokyo, 4-6-1 Komaba, Meguro-ku, Tokyo 153-8505, Japan  
e-mail: [miho@iis.u-tokyo.ac.jp](mailto:miho@iis.u-tokyo.ac.jp)

T. Okabe  
e-mail: [takahiro@iis.u-tokyo.ac.jp](mailto:takahiro@iis.u-tokyo.ac.jp)

Y. Sato  
e-mail: [ysato@iis.u-tokyo.ac.jp](mailto:ysato@iis.u-tokyo.ac.jp)

M. Shimano  
PRESTO, Japan Science and Technology Agency, Tokyo, Japan

I. Sato  
National Institute of Informatics, Tokyo, Japan  
e-mail: [imari@nii.ac.jp](mailto:imari@nii.ac.jp)



**Fig. 14.1** Videos with different exposure times. (a) LTR video with sub-exposure time (*upper*), HTR video produced by interpolation (*lower*), (b) LTR video with full-exposure time (*upper*), and HTR video produced by temporal super-resolution (*lower*)

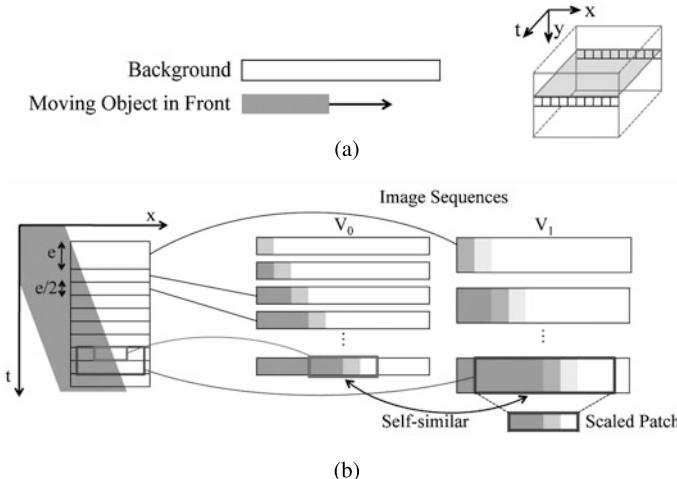
for increasing the frame rate of a given video. These techniques produce a high-frame rate video with intermediate frames generated by interpolating input frames with pixel-wise correspondences (Fig. 14.1(a)).

The quality of video largely depends on how much detailed events at different time can be resolved in relation to both the exposure time and frame rate. In a low-frame rate video recorded with *sub-exposure time*, that is, exposure time shorter than inter-frame period (Fig. 14.1(a)) discontinuous motion, called jerkiness, is often observed. On the other hand, motion blur may occur in a low-frame rate video recorded with *full-exposure time*, that is, exposure time equals to inter-frame period (Fig. 14.1(b)), due to camera shake or moving objects in a scene.

Videos often become underexposed if they are taken in sub-exposure time in those scenes like dark cloudy scenes, night scenes, and indoor scenes. To avoid images getting too dark in such cases, full-exposure time (Fig. 14.1(b)) should be selected automatically<sup>1</sup> or manually to capture the enough amount of light for suppressing noise. For such videos recorded with full-exposure time, the above temporal interpolation methods assuming sub-exposure time are not applicable because they cannot resolve one frame into two frames.

In this chapter, we propose a temporal super-resolution method for increasing the temporal resolution of a given video recorded with full-exposure time by resolving one frame to two frames as illustrated in Fig. 14.1(b) [15]. Temporal super-

<sup>1</sup>We confirmed that several consumer cameras automatically selected full-exposure mode for underexposed outdoor and indoor scenes.

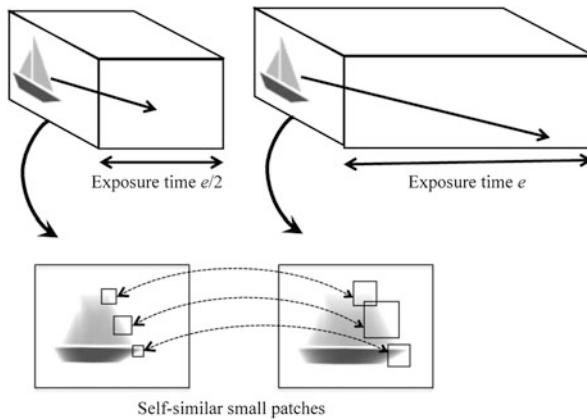


**Fig. 14.2** Self-similarity. (a) Dynamic scene (*left*) and its continuous spatio-temporal volume of video (*right*), (b)  $x$ - $t$  plane for single scan line, and 1D image sequences with different temporal resolutions

resolution is inherently ill-posed problem because there are an infinite number of high temporal resolution (HTR) frames that can produce the same low temporal resolution (LTR) frame. The key idea of this work is to exploit *self-similarity* in videos to resolve this ambiguity in creating HTR frames from LTR frames of the original video. Here, self-similarity means self-similar motion blur appearances in the spatio-temporal volume of videos with different temporal resolutions. This self-similar appearance in videos occurs because the appearance represents integrated motion of objects during each exposure time of videos with different temporal resolutions. Such self-similarity is often observed in various types of videos.

For simplicity, let us consider self-similarity in the case of 1D image sequences of a uniformly moving object (Fig. 14.2). Figure 14.2(a) shows a spatio-temporal slice of video (right) representing the dynamic scene (left). Figure 14.2(b) illustrates two image sequences,  $V_0$  and  $V_1$ , with exposure time (= inter-frame period)  $e/2$  and  $e$ . Consider, for example, a small bordered 1D image patch of  $V_0$  with exposure time  $e/2$ . In a bordered 1D image patch captured with exposure time  $e$  at the same position in  $x$ - $t$  plane, the same object moves twice the distance. If the spatial size of the bordered patch of  $V_1$  is twice as that of the bordered patch of  $V_0$ , the bordered patch becomes similar to the corresponding bordered patch because the object moves twice the distance in the image of  $V_1$ . This self-similar relationship can be extended to a 2D-image patch (Fig. 14.3).

We exploit not only the self-similar relationship at the same position but also the relationship between different spatio-temporal positions in a video. A moving object in the scene can be recursively seen at many positions along its moving path in a spatio-temporal volume of video. It is also reported that patches in a natural still image tend to recur inside the image with different scales [8]. Due to these recurrences



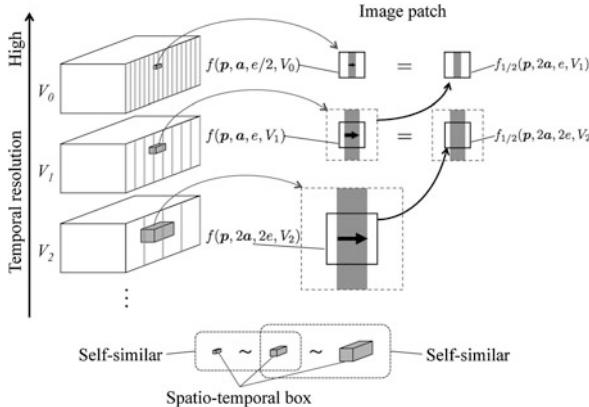
**Fig. 14.3** Self-similarity in 2D image sequences. Two image sequences of a uniformly moving sailboat with exposure time (= inter-frame period)  $e/2$  and  $e$  (*upper*) and their captured frames (*lower*). In the *right frame* captured with exposure time  $e$ , the same sailboat moves twice the distance. The blurred appearance of a small patch in the *left frame* is similar to that in the *right frame*, due to moving twice the distance in the *right frame*. Such self-similar appearance is observed in videos with different temporal resolutions

of the same and different objects, a lot of self-similar appearances between different positions can be found through a video. More thorough discussion on self-similarity will be given in Sect. 14.3.

On the basis of the self-similar appearances, we formulate the problem of temporal super-resolution as a Maximum a Posteri (MAP) estimate. Here, we use reconstruction constraints: a single frame of LTR video is equal to the average of the two corresponding frames of HTR video. Also, the following self-similar exemplar is utilized for computing a prior probability. We consider similarity at the different temporal resolutions. As shown in Fig. 14.4, consider patches in three videos of different temporal resolutions: HTR video  $V_0$  which we try to create, the original LTR video  $V_1$ , and another video  $V_2$  with even lower temporal resolution created by taking the average of two successive frames of  $V_1$ . If an image patch of  $V_2$  is similar to a patch in  $V_1$ , a patch of  $V_0$  is expected to be similar to the patch of  $V_1$ . In this way, we can create the prior probability of an HTR video  $V_0$  from scaled self-similar exemplars of LTR video  $V_1$ .

The contribution of this work is that we propose a video temporal super-resolution method from a single image sequence by exploiting the self-similarity in the spatio-temporal domain. Such self-similarity helps us to make the unconstrained problem of temporal super-resolution tractable, while avoiding the explicit motion estimations required for the previous methods. The probabilistic model that reflects both reconstruction constraints and a self-similar prior can yield an HTR sequence, which is consistent with the input LTR sequence.

The rest of this chapter is organized as follows. We briefly summarize related work in Sect. 14.2. We analyze self-similarity in videos with different temporal resolutions in Sect. 14.3. Our temporal super-resolution algorithm is introduced in



**Fig. 14.4** Self-similar exemplars. Videos capture a scene where an edge is in uniform linear motion. Since the exposure time of  $V_2$  is twice as long as that of  $V_1$ , the edge moves twice the distance in a frame of  $V_2$ . Therefore, the image patch  $f(\mathbf{p}, \mathbf{a}, \mathbf{e}, V_1)$  and the image patch  $f(\mathbf{p}, 2\mathbf{a}, 2\mathbf{e}, V_2)$  are similar. In other words,  $f(\mathbf{p}, \mathbf{a}, \mathbf{e}, V_1)$  is equal to  $f_{1/2}(\mathbf{p}, 2\mathbf{a}, 2\mathbf{e}, V_2)$  which can be synthesized from  $f(\mathbf{p}, 2\mathbf{a}, 2\mathbf{e}, V_2)$  by scaling the spatial size by 1/2

Sect. 14.4. This method is validated in our experiments in comparison to the related work in Sect. 14.5. Some applications of our temporal super-resolution techniques are introduced in Sect. 14.6, and concluding remarks are given in Sect. 14.7.

## 14.2 Related Work

One approach to increasing temporal resolution is frame rate conversion based on temporal interpolation from a single video. It is known that several existing methods have serious limitations, frame repetition suffers from jerky object motion as well as linear interpolation of successive frames enlarges blurred areas because of improperly mixed pixels of different objects. To cope with those problems, a number of methods based on motion compensation have been proposed [3, 6, 9, 10]. For these methods based on motion compensation, motion vectors such as optical flows are commonly used to blend corresponding blocks or pixels. Recently, Mahajan et al. [11] proposed an interpolation method in a gradient domain to get rid of annoying effects such as ghost, blur, and block artifacts caused by errors in motion vectors. These methods achieve good results for videos with sub-exposure times as shown in Fig. 14.1(a), but are not suitable for videos with full-exposure time as shown in Fig. 14.1(b) because establishing dense correspondence is not always possible due to motion blur. Our method has a distinct advantage over those interpolation-based approaches because dense motion estimation is not required.

Another approach to increasing the temporal resolution or deblurring is to use multiple image sequences. Ben-Ezra and Nayar [4] and Tai et al. [18] proposed the use of a hybrid camera system to capture a sequence of low spatial resolution

images at high frame rate and a sequence of high spatial resolution images at low frame rate. They estimate motion blur kernels from the former sequence and deblur the latter one. Watanabe et al. [19] proposed a method for combining two captured image sequences in the wavelet domain. Some methods used multiple image sequences that were simultaneously captured by co-located cameras with overlapping exposure times for both spatial and temporal super-resolution [1, 14]. The use of multiple sequences can significantly enhance the resolution of a video. In contrast to methods that use multiple input image sequences captured simultaneously or by using special hardware, our method uses only a single input image sequence for temporal super-resolution by exploiting a self-similar appearance in video. In the context of spatial super-resolution, Glasner et al. [8] reported that natural images have self-similar textures, that is, patches in a natural image tend to redundantly recur many times inside the image, both within the same scale as well as across different scales. On the basis of this observation, they propose a method for spatial super-resolution from a single image. Unlike this spatial super-resolution method using self-similarity in a still image, we focus on self-similarity in video for the first time to increase temporal resolution and can handle the dynamics of the scene. A recent work for a spatial and temporal super-resolution has been proposed [13] that combines the spatial super-resolution method [8] and the temporal super-resolution method [15]. In their method, the motion aliasing has been also resolved by accounting for sub-frame alignments.

In summary, we propose a method for temporal super-resolution from a single input image sequence with full-exposure time by introducing the self-similarity observed in the spatio-temporal domain. Unlike the frame rate conversion approach based on temporal interpolation, our method does not need dense motion estimation. Furthermore, unlike the approach using multiple image sequences, our method does not require multiple image sequences captured simultaneously or the use of special hardware.

## 14.3 Self-similarity

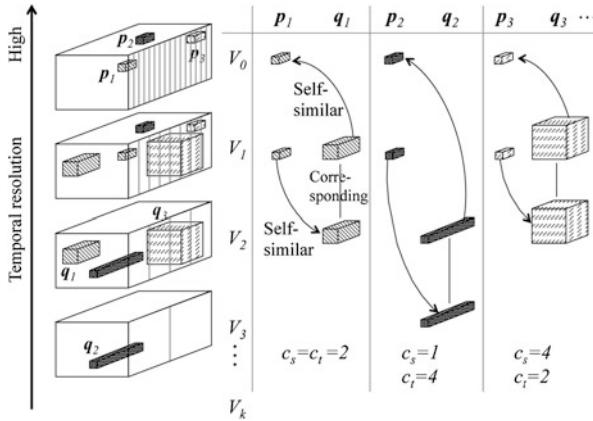
Here, we explain self-similarity in video in more detail, including the necessary conditions under which self-similarity is observed.

### 14.3.1 Definition

Let us consider an input video  $V_1$  captured with an exposure time  $e$  and construct a video  $V_2$  with an exposure time  $2e$  by averaging two adjacent image frames of  $V_1$ .<sup>2</sup> In the same manner, we recursively construct a video  $V_k$  with an exposure time

---

<sup>2</sup>We assume that the gain of  $V_2$  is half the gain of  $V_1$ . Therefore, an image frame of  $V_2$  is equal to the average of the two corresponding image frames of  $V_1$  rather than their sum.



**Fig. 14.5** Examples of self-similarity. The self-similar relationship can be observed in the videos with different temporal resolutions. If a spatio-temporal box in  $V_1$  at  $\mathbf{p}_i$  is similar to a spatio-temporal box in  $V_2$  at  $\mathbf{q}_j$ , a spatio-temporal box in  $V_0$  at  $\mathbf{p}_i$  is expected to be similar to the corresponding spatio-temporal box in  $V_1$  at  $\mathbf{q}_j$ . Namely, if spatio-temporal boxes with low temporal resolutions are similar, the corresponding spatio-temporal boxes with high temporal resolutions are similar

$2^{k-1}e$  from a video  $V_{k-1}$ . Self-similarity is a similarity structure observed among videos with different exposure times constructed from a single input video  $V_1$ .

For the sake of simplicity, let us begin with a video  $V_1$  capturing a scene where an edge with an infinite length is in uniform linear motion as shown in Fig. 14.4. We denote a small image patch in  $V_1$  as  $f(\mathbf{p}, \mathbf{a}, e, V_1)$ , where  $\mathbf{p} = (x, y, t)^T$  is the space-time coordinates of the center,  $\mathbf{a} = (a_x, a_y)$  is the spatial size, and  $e$  is the exposure time. Similarly, we denote a small image patch in  $V_2$  as  $f(\mathbf{p}, \mathbf{a}, 2e, V_2)$ , and so on. Since the exposure time of  $V_2$  is twice as long as that of  $V_1$ , the edge moves twice the distance in a single image frame of  $V_2$ . Therefore, the image patch  $f(\mathbf{p}, \mathbf{a}, e, V_1)$  and the image patch  $f(\mathbf{p}, 2\mathbf{a}, 2e, V_2)$  are similar. In other words,  $f(\mathbf{p}, \mathbf{a}, e, V_1)$  is equal to  $f_{1/2}(\mathbf{p}, 2\mathbf{a}, 2e, V_2)$  which can be synthesized from  $f(\mathbf{p}, 2\mathbf{a}, 2e, V_2)$  by scaling the spatial size by 1/2. In the same manner,  $f(\mathbf{p}, \mathbf{a}, e, V_1)$  is equal to  $f_{1/4}(\mathbf{p}, 4\mathbf{a}, 4e, V_3)$ , and so on.

Generally, image patches  $f(\mathbf{p}, \mathbf{a}, e, V_1)$  and  $f(\mathbf{q}, c_s \mathbf{a}, c_t e, V_k)$  where  $c_t = 2^{k-1}$  are self-similar, if there is a pair of scalars  $c_s$  and  $c_t$  such that  $f(\mathbf{p}, \mathbf{a}, e, V_1) = f_{1/c_s}(\mathbf{q}, c_s \mathbf{a}, c_t e, V_k)$ . Note that the self-similarity could be observed between image patches at different space-time coordinates as illustrated in Fig. 14.5.

### 14.3.2 Necessary Conditions

As defined in the previous section, self-similarity among videos with different exposure times requires similarities in both the spatial and temporal domains. For the spatial domain, self-similarity requires that small image patches appear similar when one changes the spatial scale according to the given temporal scale. For some

families of images, as Glasner et al. [8] reported, patches in a natural image tend to redundantly recur many times inside the image, both within the same scale as well as across different scales. In addition, since we deal with small patches instead of the entire image, a small patch containing an edge longer than the spatial size of the patch, for example, is considered to be primitives and be invariant to the scale change.

For the temporal domain, self-similarity requires that the motions observed in small patches are similar when one changes the temporal scale in accordance with the given spatial scale. This is exactly true for objects in uniform linear motion (e.g.,  $c_s = 2$  and  $c_t = 2$ ) and in uniformly-accelerated motion (e.g.,  $c_s = 4$  and  $c_t = 2$ ) (Fig. 14.5). In addition, since we deal with images captured during the exposure time, that is, a short period of time, the velocity of an object of interest is often considered to be constant. As a result, the behavior of the object such as a complex non-linear motion can often be approximated by uniform linear motion, and then various motions in small patches are similar each other.

Since multiple consecutive patches in a spatio-temporal box are self-similar to those in the corresponding spatio-temporal box, the self-similar relationship can be extended to the self-similar relationship between the spatio-temporal boxes including multiple consecutive patches. Therefore, whereas the spatio-temporal box in Fig. 14.4 or Fig. 14.5 represents an image patch for a simplified notion in the above examples, it can also represent consecutive small image patches. Note that we can use not only the image patches but the difference between image patches for considering self-similar relationship. In fact, multiple patches included in a spatio-temporal box and the difference between them are used in the experiment of Sect. 14.5. Furthermore, the number of the patches in the spatio-temporal box may change at each temporal-scale level. It should be noted here that whereas only two temporal-scale levels are shown in Fig. 14.5, multiple number of temporal-scale levels may be used in the consideration of self-similarity.

## 14.4 Temporal Super-resolution Algorithm

### 14.4.1 Overview

We propose an algorithm for increasing the temporal resolution of an input video twice.<sup>3</sup> That is, we reconstruct a video  $V_0$  with an exposure time  $e/2$  from a single input video  $V_1$  with an exposure time  $e$  and a set of videos  $V_k$  ( $k = 2, 3, 4, \dots$ ) with an exposure time  $2^{k-1}e$  constructed from  $V_1$  by averaging adjacent image frames recursively.

Since we assume that the exposure time of the input video is almost equal to the inter-frame period, a single image frame of  $V_1$  is equal to the average of the

---

<sup>3</sup>The algorithm for increasing the temporal resolution more than twice can be formulated straightforwardly.

two corresponding image frames of  $V_0$ . We impose this constraint termed a *reconstruction constraint*<sup>4</sup> in order to decompose a single image frame of  $V_1$  into two image frames of  $V_0$ , as described in Sect. 14.4.2. However, this decomposition is an ill-posed problem because an infinite number of combinations of image frames can satisfy the constraint.

To cope with this ill-posedness, we incorporate another constraint based on self-similarity. We take advantage of self-similar exemplars to make the ill-posed problem tractable (Sect. 14.4.3).

To this end, we formulate temporal super-resolution as a MAP estimation. Let us denote the image frame of  $V_k$  at time  $t$  as  $F_k(t)$ , or  $F_k$  for short. According to Bayes' law, the posterior probability  $P(F_0|F_1)$  of the random variable  $F_0$  given an observation  $F_1$  is

$$P(F_0|F_1) = \frac{P(F_1|F_0)P(F_0)}{P(F_1)}. \quad (14.1)$$

Here,  $P(F_1|F_0)$  and  $P(F_0)$  are the likelihood of  $F_1$  given  $F_0$  and the prior probability of  $F_0$  respectively. Since the probability  $P(F_1)$  is constant, the MAP estimation of  $F_0$  results in minimization of the negative logarithm of  $P(F_1|F_0)P(F_0)$ :

$$F_0^{\text{MAP}} = \arg \min_{F_0} [-\ln P(F_1|F_0) - \ln P(F_0)]. \quad (14.2)$$

In the next two sections, we describe how the likelihood and the prior probability are computed from the reconstruction constraint and self-similar exemplars.

#### 14.4.2 Likelihood Model

We compute the likelihood from the reconstruction constraint. Assuming that the pixel values of image frame  $F_1(t)$  are contaminated by additive noise, the constraint is described as

$$F_1(x, y, t) = \frac{F_0(x, y, t) + F_0(x, y, t + e/2)}{2} + \eta(x, y, t). \quad (14.3)$$

Here,  $F_k(x, y, t)$  is a pixel value of the image frame  $F_k(t)$  at the image coordinates  $(x, y)$ , and  $\eta(x, y, t)$  stands for additive noise.

We assume that the additive noise obeys an i.i.d. Gaussian distribution with standard deviation  $\sigma_{\text{like}}$ . Accordingly, the first term of Eq. (14.2) becomes

---

<sup>4</sup>Strictly speaking, this constraint assumes a linear camera response function (CRF). However, because adjacent image frames usually have similar pixel values, it is approximately satisfied when the CRF is approximated by a piecewise-linear function. In addition, one could calibrate the CRF and convert the pixel values in advance.

$$-\ln P(F_1|F_0) = \frac{1}{2\sigma_{\text{like}}^2} \sum_{\Omega} \left[ F_1(x, y, t) - \frac{F_0(x, y, t) + F_0(x, y, t + e/2)}{2} \right]^2, \quad (14.4)$$

where  $\Omega$  is the image plane over which the summation is taken, and we have omitted constant terms with respect to  $F_0$ .

### 14.4.3 Prior Model

We compute the prior probability from self-similar exemplars in a similar manner to image hallucination [2, 7]. The self-similar exemplars are exploited on the following assumption: if an image patch  $f(\mathbf{p}, \mathbf{a}, e, V_1)$  and a scaled patch  $f_{1/2}(\mathbf{q}, 2\mathbf{a}, 2e, V_2)$  are similar, the corresponding patch  $f(\mathbf{p}, \mathbf{a}, e/2, V_0)$  and the corresponding scaled patch  $f_{1/2}(\mathbf{q}, 2\mathbf{a}, e, V_1)$  are also similar as shown in Fig. 14.4 and Fig. 14.5.

Specifically, we find an image patch  $f_{1/2}(\mathbf{q}, 2\mathbf{a}, 2e, V_2)$  similar to  $f(\mathbf{p}, \mathbf{a}, e, V_1)$  by approximate Nearest Neighbors [12], and consider  $f_{1/2}(\mathbf{q}, 2\mathbf{a}, e, V_1)$  as a candidate for  $f(\mathbf{p}, \mathbf{a}, e/2, V_0)$ . The candidate image patch  $\tilde{f}(\mathbf{p}, \mathbf{a}, e/2, V_0)$  is given by

$$\begin{aligned} \tilde{f}(\mathbf{p}, \mathbf{a}, e/2, V_0) &= f_{1/2}(\mathbf{q}, 2\mathbf{a}, e, V_1), \\ \text{where } \mathbf{q} &= \arg \min_{\mathbf{p}'} |f(\mathbf{p}, \mathbf{a}, e, V_1) - f_{1/2}(\mathbf{p}', 2\mathbf{a}, 2e, V_2)|. \end{aligned} \quad (14.5)$$

The difference between image patches is defined as the sum of the squared differences (SSD) of the pixel values.

Next, the candidate image patches  $\tilde{f}(\mathbf{p}, \mathbf{a}, e/2, V_0)$  are concatenated into the candidate image frame  $\tilde{F}_0(t)$ . Here, we assume that the deviation of  $F_0$  from the candidate  $\tilde{F}_0$  obeys an i.i.d. Gaussian distribution with standard deviation  $\sigma_{\text{pri}}$ . In addition, we do not use the image frames but the difference between image frames for modeling the prior probability because it performs better experimentally. Hence, the second term of Eq. (14.2) is given by

$$-\ln P(F_0) = \frac{1}{2\sigma_{\text{pri}}^2} \sum_{\Omega, t'} [\Delta F_0(x, y, t') - \Delta \tilde{F}_0(x, y, t')]^2, \quad (14.6)$$

where  $t' = t, t + e/2$ , and constant terms have been omitted.

Note that we can use more temporal-scale levels for determining a candidate image patch in Eq. (14.6). The current implementation computes the SSD between  $f(\mathbf{p}, \mathbf{a}, 2e, V_2)$  and  $f_{1/2}(\mathbf{p}', 2\mathbf{a}, 4e, V_3)$  in addition to the SSD between  $f(\mathbf{p}, \mathbf{a}, e, V_1)$  and  $f_{1/2}(\mathbf{p}', 2\mathbf{a}, 2e, V_2)$ . That is, we use the videos  $V_k$  constructed from the input video  $V_1$  up to  $k = 3$ .

#### 14.4.4 Optimization

To reduce spatial noise and temporal flicker, we add a smoothness term  $S$  to Eq. (14.2). Substituting Eq. (14.4) and Eq. (14.6) into Eq. (14.2), the MAP estimation of  $F_0$  results in

$$\begin{aligned} F_0^{\text{MAP}} &= \arg \min_{F_0} \left[ -\ln P(F_1|F_0) - \ln P(F_0) + S \right] \\ &= \arg \min_{F_0} \left\{ \frac{1}{2\sigma_{\text{like}}^2} \sum_{\Omega} \left[ F_1(x, y, t) - \frac{F_0(x, y, t) + F_0(x, y, t + e/2)}{2} \right]^2 \right. \\ &\quad + \frac{1}{2\sigma_{\text{pri}}^2} \sum_{\Omega, t'} \left[ \Delta F_0(x, y, t') - \Delta \tilde{F}_0(x, y, t') \right]^2 \\ &\quad \left. + \lambda \sum_{\Omega, t'} \left[ \sum_{\Lambda} w(x, y, t', m, n, q) L(m, n, q) F_0(x+m, y+n, t'+q) \right]^2 \right\}, \end{aligned} \quad (14.7)$$

where  $\Delta F_0(x, y, t') = F_0(x, y, t' + e/2) - F_0(x, y, t')$ .

Our smoothness term is based on the discrete Laplacian filter  $L(m, n, q)$ . To preserve spatiotemporal edges, we incorporate a pixel-wise weight  $w(x, y, t', m, n, q)$ . It is defined as  $w(x, y, t', m, n, q) = w_d(x, y, t', m, n, q) / \sum_{m', n', q'} w_d(x, y, t', m', n', q')$ , where  $w_d(x, y, t', m, n, q) = \exp[-(F(x, y, t) - F(x + m, y + n, t' + q))^2 / 2\sigma_s^2]$ .

The optimization problem of Eq. (14.7) is iteratively solved according to the following steps.

1. Ignore the smoothness term in Eq. (14.7), and compute the initial estimation  $F_0^{r=0}$  by using the least-squares method.
2. Compute the weight  $w(x, y, t', m, n, q)$  from  $F_0^r$ , and update the estimation as  $F_0^{r+1}$  by minimizing Eq. (14.2) via least squares.
3. Repeat step 2 until the difference  $|F^{r+1} - F^r|$  converges.

Usually, only a few iterations are needed for convergence.

## 14.5 Experimental Results

We evaluated various aspects of our approach in experiments using real videos. In Sect. 14.5.1, we compare the performance of our method with that of previously proposed temporal interpolation methods. We then clarified the importance of each component of our algorithm in Sect. 14.5.2.

We used CANON HF-20 and CASIO EX-F1 video cameras. The exposure time of the camera was set almost equal to inter-frame period. Eight kinds of videos

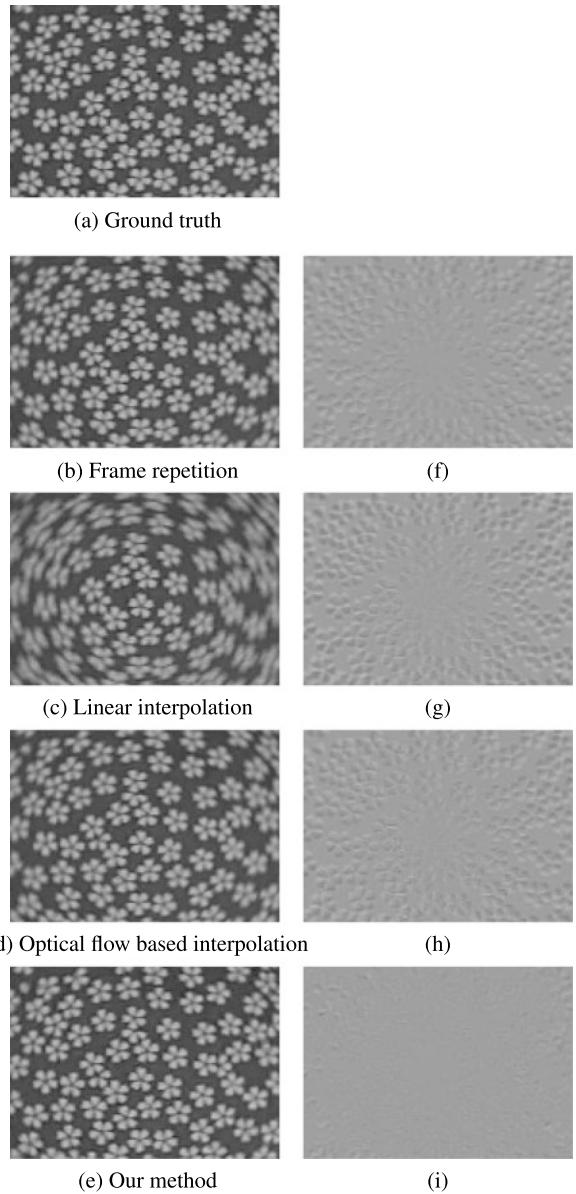
were collected as  $V_0$ , by capturing dynamic scenes for about a few hundred frames: three videos capturing indoor scenes and five videos captured outside. The video of textured papers 1 and 2 were captured as scenes of uniform circular motion. The video of a small merry-go-round was captured as scenes with uniform linear motion as well as various other motions. Textured paper 1 included repetitions of the same texture, and textured paper 2 included several textures. The videos captured outside included various scenes such as walking people, cars, and trees. The frame images of the captured videos were converted to grayscale in all experiments. The captured videos were considered to be the ground truth high-temporal-resolution video  $V_0$ . We reconstructed  $V_0$  from its low-temporal-resolution videos  $V_1$ ,  $V_2$ , and  $V_3$  that were provided from  $V_0$ , as mentioned in Sect. 14.3. Our algorithm used  $5 \times 5$  pixel patches. We empirically set  $\sigma_{\text{like}} = 1$ ,  $\sigma_{\text{pri}} = 1.25$ ,  $\lambda = 1.6$ , and  $\sigma_s$  was set from 10 to 100 to adjust to the brightness of the frames. The produced grayscale videos were converted to a YUV color space as follows. We obtained the Y channel as the produced grayscale video. For the U and V channels, we used those channels of the original low temporal resolution frame of  $V_1$  for two successive high temporal-resolution frames.

### 14.5.1 Comparison with Other Methods

We compared the results of our technique with those of three methods: simple frame repetition, linear interpolation, and interpolation based on optical flows [3, 5]. Linear interpolation and optical flow based interpolation generate same number of intermediate frames from input frames. The eight videos were used to examine the performance of each method on various motions.

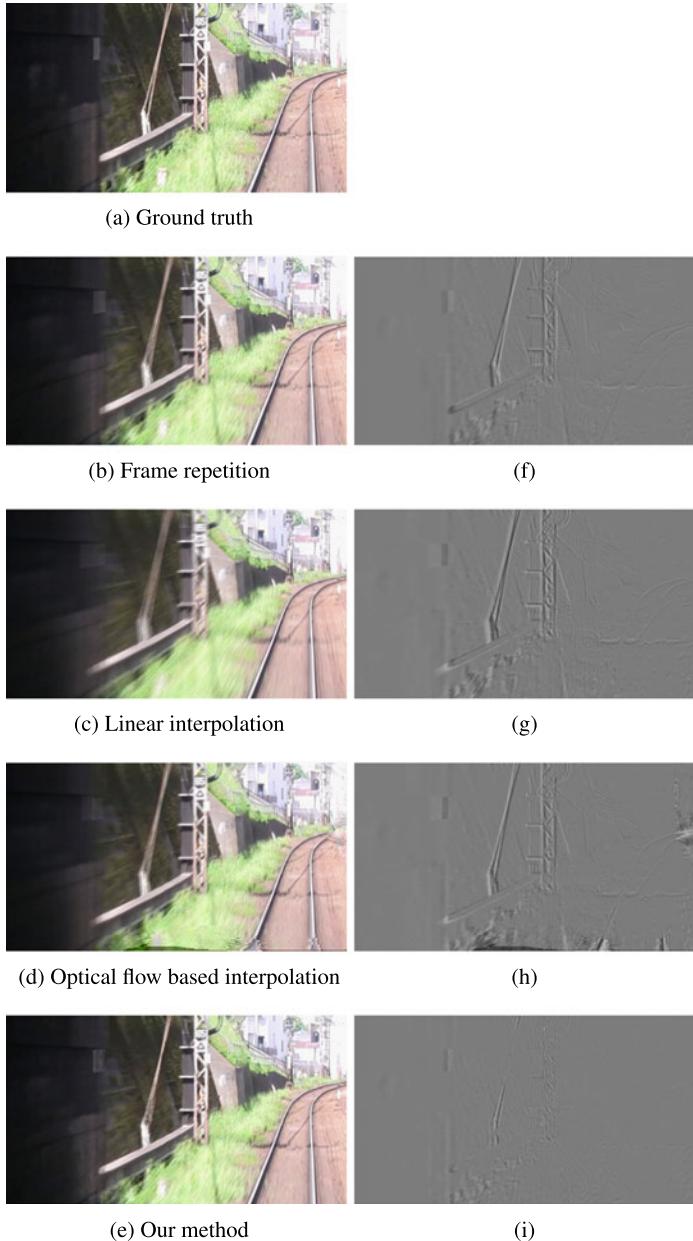
We made a qualitative comparison of our method and the other interpolation methods. In Fig. 14.6 and Fig. 14.7, (a) shows the ground truth and (f)–(i) show the differences between the ground truth and the reconstructed frame (b)–(e) respectively by various methods ((b) frame repetition, (c) linear interpolation, (d) optical flow based interpolation, and (e) our method). As we see in Fig. 14.6(e) and Fig. 14.7(e), the visual quality of the frames recovered using our method is much better than those of the other interpolation methods. In the results obtained by optical flow based interpolation (d), the motion blur region remained or was enlarged in the interpolated frame. Motion blur likewise remained in the videos produced by frame repetition and was enlarged in the videos produced by linear interpolation. In Fig. 14.8, we show our results, zoom-ups in interesting areas, to compare to those of the input and the results obtained by optical flow based interpolation and the ground truth. In the scene of a crowded intersection with diagonal crosswalks, the moving leg (Fig. 14.8(b)) and coming and going people (Fig. 14.8(c)) in the frames produced by our method matched very closely with the ground truth. On the other hand, the optical flow based interpolation have sometimes failed (compare, e.g., the left leg in frame 3, frame 5 and frame 6 of Fig. 14.8(b) and the lower part of frame 1, frame 3 and frame 5 in Fig. 14.8(c)). Our method could also be used to generate frames to

**Fig. 14.6** Comparison of our method with other interpolation methods applied video in which textured paper1 rotates uniformly:  
**(a)** ground truth frame, frames with increased temporal resolution **(b)** frame repetition, **(c)** linear interpolation, **(d)** optical flow based interpolation, and **(e)** our method. **(f), (g), (h), (i)** The difference (image values are added offset 128 to the difference) between each produced frame (**(b), (c), (d), or (e)**) and the ground truth **(a)**

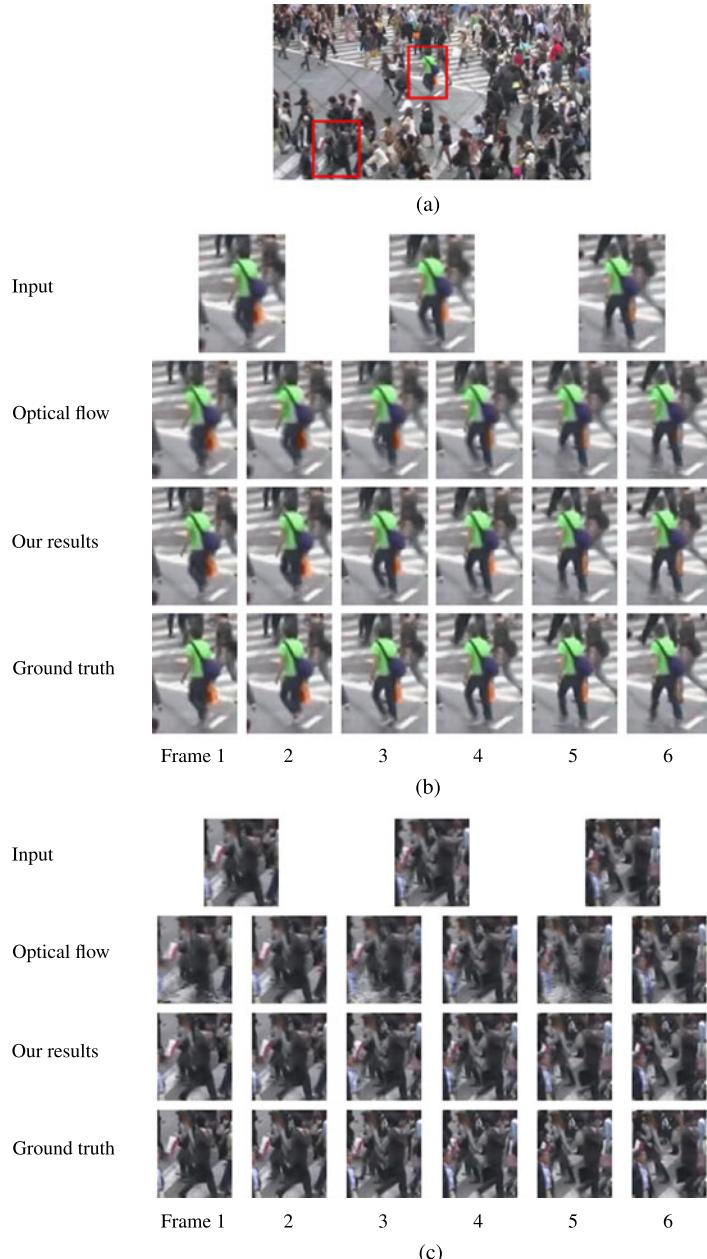


represent correctly two heads moving apart (Fig. 14.9(a)), the changing movie displayed on the big TV screen (Fig. 14.9(b)) and the shadow hovering beyond the vein of the leaf (Fig. 14.9(c)). Note that the information of the input frame was properly resolved to two frames with high temporal resolution produced by our method.

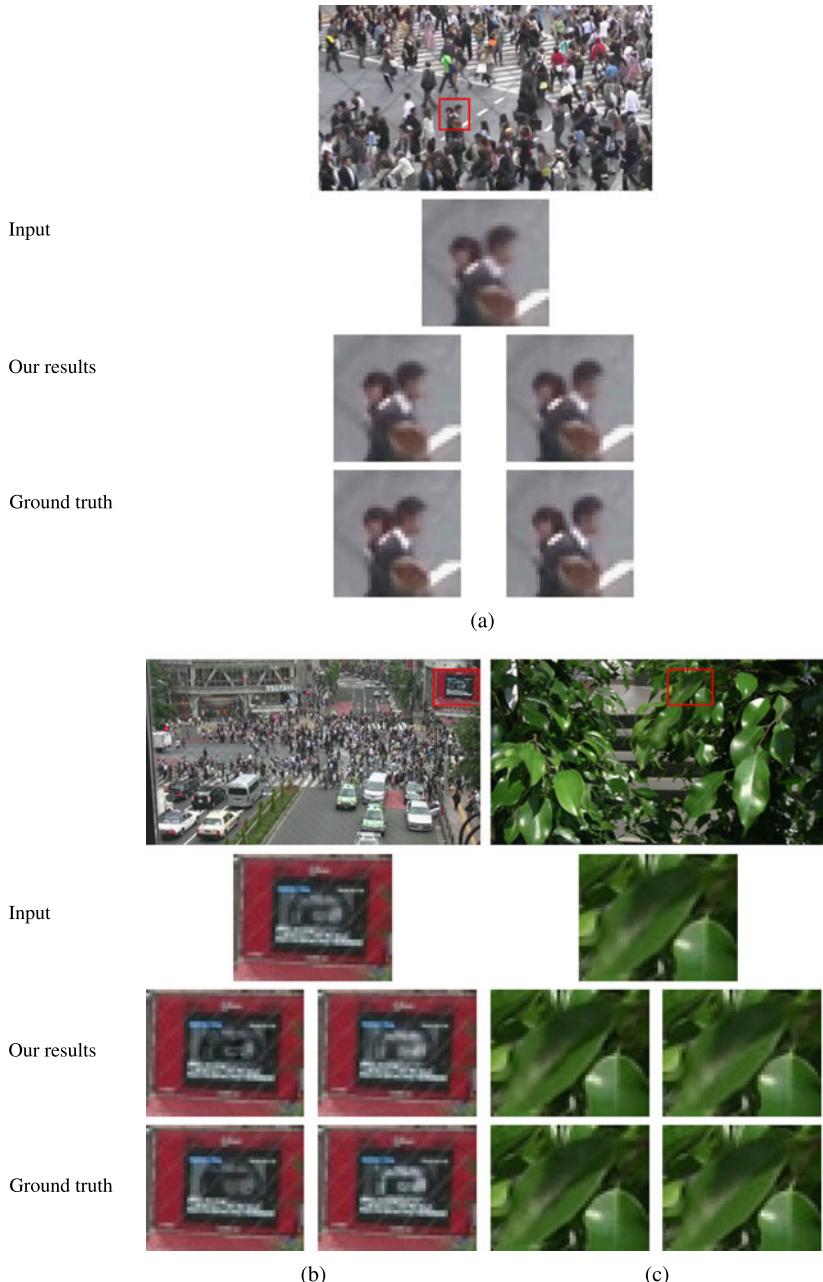
To make a quantitative comparison, we also evaluated the peak signal to noise ratio (PSNR) between the frames produced by all four temporal super-resolutions



**Fig. 14.7** Comparison of our method with other interpolation methods applied to video of outside1. (a) Ground truth frame, frames with increased temporal resolution (b) frame repetition, (c) linear interpolation, (d) optical flow based interpolation, and (e) our method. (f), (g), (h), (i) Difference (image values are added offset 128 to the difference) between each produced frame ((b), (c), (d), or (e)) and the ground truth (a)

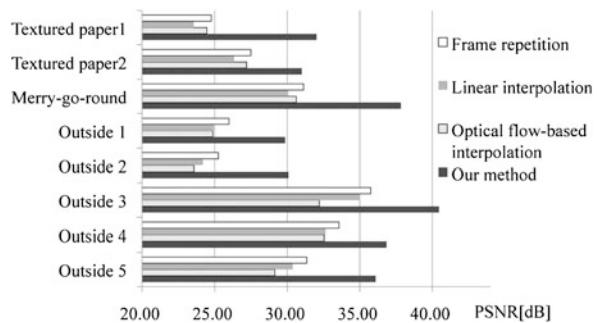


**Fig. 14.8** Zoom-ups of high temporal resolution frames produced by our method applied to video of outside2. **(a)** The example input frames, **(b), (c)**, Zoom-ups of the marked region in the input frames (*first row*), zoom-ups of the marked region in the frames produced by optical flow based interpolation method (*second row*), zoom-ups of the marked region in the frames with increased temporal resolution produced by our method (*third row*), and zoom-ups of the marked region in the ground truth frames (*fourth row*)



**Fig. 14.9** Zoom-ups of high temporal resolution frames produced by our method applied to video of (a) outside2, (b) outside3, and (c) outside4. The example input frames (*first row*), zoom-ups of the marked region in the input frames (*second row*), zoom-ups of the marked region in the two frames with increased temporal resolution produced by our method (*third row*), and zoom-ups of the marked region in the two ground truth frames (*fourth row*)

**Fig. 14.10** Comparison of our method with other interpolation methods



methods and the ground truth frames. The average PSNRs of all four methods are shown in Fig. 14.10. We clearly see that our method is superior to the other methods for all videos. The optical flow based interpolation have sometimes failed to estimate the optical flows because of the input blurred frames, and these errors may have led to more ghosting and blurring artifacts than in the frame repetition or the linear interpolation video. The qualitative and quantitative comparisons indicated that our method can be effectively used for compensating circular, linear, and various other motions.

#### 14.5.2 Evaluation of Components

We conducted experiments to evaluate the contribution of each component in our method such as reconstruction constraints, the smoothness term, the use of temporal-scale levels up to  $V_k$ , and features describing self-similar structures (images or their differences) by using 30 frames each of three captured videos, as follows.

First, we examined the effectiveness of considering multiple temporal-scale levels  $k$  of the videos. As shown in Table 14.1, cases using temporal-scale levels up to  $k = 3, 4$  superior to the case using temporal-scale levels up to  $k = 2$  and achieved a better PSNR. This shows that examining motion at different exposure times ( $V_1, V_2$ , and so on) is effective for revealing motion at higher frame rate.

To examine the contribution of each component, we conducted experiments by adding components incrementally. The PSNR of each case is shown in Table 14.2. Here, the quality of the produced videos is improved, as components are added. In other words, all the components play essential roles for improving the resulting image quality.

## 14.6 Application

The temporal super resolution techniques described in this chapter can be applied to video compression in addition to video enhancement. Some recent work has

**Table 14.1** Comparison of different temporal-scale levels. Our algorithm finds results using videos  $V_k$  constructed from the input video  $V_1$  up to  $k = 4$  are compared

Sequence	PSNR [dB]		
	$k = 2$	$k = 3$	$k = 4$
Textured paper1	15.68	24.06	24.01
Textured paper2	18.60	27.45	27.40
Merry-go-round	21.77	34.34	33.19

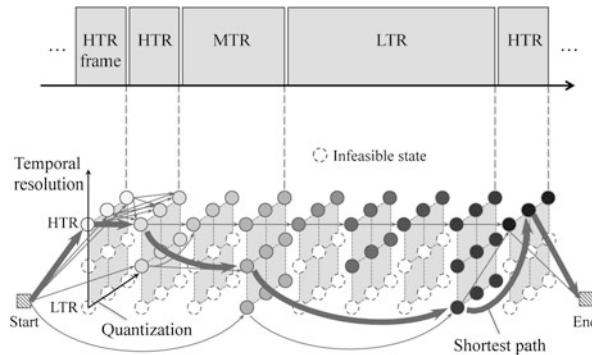
**Table 14.2** Evaluation of a variety of components. Note that use of several temporal-scale levels, image differences, reconstruction constraints, and the smoothness term are all important in our method

Case	$k = 3$	Image differences	Reconstruction constraints	Smoothness term	PSNR [dB]		
					Textured paper1	Textured paper2	Merry-go-round
1	–	–	–	–	15.55	19.06	17.96
2	✓	–	–	–	22.90	26.26	32.78
3	✓	✓	–	–	24.06	27.45	34.34
4	✓	✓	✓	–	25.99	28.82	35.49
5	✓	✓	✓	✓	26.19	28.72	35.58

started investigating what is the most suitable representation of the video at encoder, since a temporal super resolution video has to be constructed at decoder with low complexity [16, 17]. In order to optimize the rate-distortion performance for video compression, an adaptive encoding strategy to select frames at different temporal resolutions [17] and quantization parameters (QPs) [16] has been proposed as schematically shown in Fig. 14.11. The key idea is to encode at low temporal resolution (and possibly at coarse QP) frames that can be easily synthesized via temporal super-resolution at decoder to save bits, and to encode at high temporal resolution (and possibly at fine QP) frames that are difficult to synthesize after compression to preserve quality. The selection of frames at different temporal resolutions and QPs is formulated as a shortest path problem in a directed acyclic graph (DAG) as illustrated in Fig. 14.11.

## 14.7 Conclusions

We introduced an approach for making temporal super resolution video from a single input image sequence. To make this ill-posed problem tractable, we focused on *self-similarity* in video which is a self-similar appearances with different temporal resolutions. Our method increases the temporal resolution by using a MAP estimation that incorporates both a prior probability computed from self-similar appearances and reconstruction constraints that is, a single frame of LTR video should



**Fig. 14.11** Example of frame selection (*upper*) and a three-dimensional directed acyclic graph (DAG) representing selections of temporal resolutions and corresponding quantization parameters (QPs) for encoding of the sequence (*lower*). The video frames at appropriate temporal resolutions, i.e., HTR, middle temporal resolution (MTR) or LTR, and quantization parameters are selected for encoding by finding a shortest cost path

be equal to the average of the two corresponding frames of HTR video. The experimental results showed that our method could recover a high-quality video with increased temporal resolution, which keeps consistent with the input LTR video. We demonstrated that our approach produce results that are sharper and clearer than those of other techniques based on temporal interpolation.

## References

1. Agrawal A, Gupta M, Veeraraghavan A, Narasimhan S (2010) Optimal coded sampling for temporal super-resolution. In: Proc CVPR 2010, pp 599–606
2. Baker S, Kanade T (2002) Limits on super-resolution and how to break them. IEEE Trans Pattern Anal Mach Intell 24(9):1167–1183
3. Baker S, Scharstein D, Lewis J, Roth S, Black M, Szeliski R (2007) A database and evaluation methodology for optical flow. In: Proc ICCV 2007, pp 1–8
4. Ben-Ezra M, Nayar SK (2003) Motion deblurring using hybrid imaging. In: Proc CVPR 2003, pp I-657–I-664
5. Brox T, Bruhn A, Papenberg N, Weickert J (2004) High accuracy optical flow estimation based on a theory for warping. In: Proc ECCV 2004. LNCS, vol 3024, pp 25–36
6. Choi BT, Lee SH, Ko SJ (2000) New frame rate up-conversion using bi-directional motion estimation. IEEE Trans Consum Electron 46(3):603–609
7. Freeman W, Jones T, Pasztor E (2002) Example-based super-resolution. IEEE Comput Graph Appl 22(2):56–65
8. Glasner D, Bagon S, Irani M (2009) Super-resolution from a single image. In: Proc ICCV 2009, pp 349–356
9. Ha T, Lee S, Kim J (2004) Motion compensated frame interpolation by new block-based motion estimation algorithm. IEEE Trans Consum Electron 50(2):752–759
10. Kuo T-Y, Kim J, Kuo C-CJ (1999) Motion-compensated frame interpolation scheme for H.263 codec. In: Proc ISCAS '99, pp 491–494
11. Mahajan D, Huang FC, Matusik W, Ramamoorthi R, Belhumeur P (2009) Moving gradients: a path-based method for plausible image interpolation. In: Proc SIGGRAPH'99, pp 1–11

12. Muja M, Lowe DG (2009) Fast approximate nearest neighbors with automatic algorithm configuration. In: Proc VISSAPP 2009, pp 331–340
13. Shahar O, Faktor A, Irani M (2011) Space-time super-resolution from a single video. In: Proc CVPR 2011, pp 3353–3360
14. Shechtman E, Caspi Y, Irani M (2005) Space-time super-resolution. IEEE Trans Pattern Anal Mach Intell 27(4):531–545
15. Shimano M, Okabe T, Sato I, Sato Y (2010) Video temporal super-resolution based on self-similarity. In: Proc ACCV 2010. LNCS, vol 6492, pp 93–106
16. Shimano M, Cheung G, Sato I (2011) Adaptive frame and qp selection for temporally super-resolved full-exposure-time video. In: Proc ICIP 2011, pp 2253–2256
17. Shimano M, Cheung G, Sato I (2011) Compression using self-similarity-based temporal super-resolution for full-exposure-time video. In: Proc ICASSP 2011, pp 1053–1056
18. Tai YW, Du H, Brown MS, Lin S (2008) Image/video deblurring using a hybrid camera. In: Proc CVPR 2008, pp 1–8
19. Watanabe K, Iwai Y, Nagahara H, Yachida M, Suzuki T (2006) Video synthesis with high spatio-temporal resolution using motion compensation and image fusion in wavelet domain. In: Proc ACCV 2006. LNCS, vol 3851, pp I-480–I-489

# Index

## Symbols

$\alpha$ -expansions, 158

3D reconstruction, 185

## A

Acquisition strategy, 188

Action recognition, 65

Active learning, 35

Activities, 12

Activity recognition, 96

Adaptive support weights, 148

Adaptive weight functions, 148

Adaptive windows, 147

Alignment, 189, 195

Applications of stereo matching, 145

## B

Bayesian modeling, 277, 279, 280

Belief propagation, 159, 313

Bilateral weights, 148

Boosting, 341

BRIEF descriptor, 25

BRISK descriptor, 25

BRISK detector, 20

## C

Camera estimation, 277, 278, 281, 295, 302

CARD, 25

CenSurE descriptor, 23

CenSurE detector, 18

CENTRIST descriptor, 9

Classifier Grids, 387

Co-recognition, 114

Co-training, 391

Codebook, 69

Color, 9

Color in stereo matching, 163

Combined stereo and matting, 168

Conservative Learning, 385

Convex relaxation, 218

Coordinate system, 193

Corner strength, 14

Cornerness, 14

Cost filter stereo, 151

## D

DAISY descriptor, 24

Data term in stereo, 160

Deep belief networks, 86

Dense trajectories, 99

Differential GPS, 209

Diffusion moves, 277, 288, 290

Digital surface model, 192

Dimensionality reduction, 85

DSM, 192

Dynamic programming, 157

## E

Edge detector, 277, 295

Edge-sensitive smoothness terms, 166

Effective inlier, 187

Epipolar graph, 185

Events, 12

Exposure time, 412

## F

FAST detector, 19

FAST-ER detector, 20

Feature description, 21

Feature detection, 13

Filter-based weights, 150

Fisher discriminant analysis, 251

Förstner detector, 16

Four Color Theorem, 313, 315

- Frame rate, 412  
 FREAK descriptor, 25  
 Fusion moves, 159
- G**  
 Geodesic weights, 149  
 Geometric context, 277, 287, 305  
 Gestalt, 4  
 Gestalt Theory, 4  
 Gist, 9  
 GIST descriptor, 9  
 Global stereo methods, 155  
 GLOH, 22  
 Graph coloring, 316  
 Graph construction, 35  
 Graph cuts, 158, 313  
 Graph-based learning, 35
- H**  
 Hamiltonian dynamics, 277, 290  
 Harris corner detector, 15  
 Harris-Affine detector, 16  
 Harris-Laplace detector, 16  
 Hessian detector, 17  
 Hessian-Affine detector, 17  
 Hessian-Laplace detector, 17  
 Highly-connected smoothness terms, 166  
 HOG descriptor, 22  
 Holism, 4  
 Hough transform, 10
- I**  
 Identical object matching and segmentation, 128  
 Image classification, 244, 246, 341  
 Image retrieval, 271  
 Image segmentation, 215  
 Image-based localization, 197  
 Incremental feature update, 201  
 Indoor scenes, 277  
 Information content, 26  
 Intensity, 9  
 Inverse MILBoost, 382
- J**  
 Jump moves, 277, 288, 294
- K**  
 k-nearest neighbors, 259  
 k-NN classification, 341
- L**  
 Large-scale classification, 246  
 Large-scale recognition, 35
- Limitations of local stereo, 154  
 Linear classification, 252  
 LIOP descriptor, 25  
 Local Binary Patterns, 25  
 Local features, 11  
 Local stereo methods, 145  
 Localization, 197  
 Localization accuracy, 26  
 Long-term localization, 210
- M**  
 Mahalanobis distance, 250  
 Manhattan world, 277, 278, 281, 294, 295  
 Many-to-many object matching, 115  
 Markov Random Field, 313, 315  
 Matching score, 26  
 Matting problem in stereo, 167  
 MAV, 181  
 Maximal correspondence, 116  
 MCMC, 277, 280, 288  
 Message passing, 159  
 Metric learning, 35, 247, 250, 259  
 Micro aerial vehicle, 181  
 Minimum Spanning Tree, 315  
 Moment constraints, 217  
 Monogenic signal, 21  
 Moravec corner detector, 14  
 Motion, 12  
 Motion blur, 413, 422  
 MSER detector, 20  
 MU-SURF descriptor, 23  
 Multi-class logistic regression, 250  
 Multi-layer match-growing, 122  
 Multi-scale Harris detector, 16  
 Multiple-instance learning, 382, 391  
 Multithreaded, 277, 290, 302, 307
- N**  
 Nearest class mean classifier, 249
- O**  
 O(1) Approaches, 150  
 Object detection, 377  
 Object recognition, 35, 277, 279, 282, 305  
 Object Stereo, 172  
 Object tracking, 231  
 Occlusions in global stereo, 163  
 Online GradientBoost, 379  
 Online learning, 378  
 Optimization methods, 156  
 ORB descriptor, 25  
 ORB detector, 20

**P**

Patch-based multiview stereo, 187  
PatchMatch stereo, 153  
Person detection, 404  
Physics-based stereo, 174  
Pixel dissimilarity functions, 160  
Point cloud registration, 6  
Primal sketch, 5  
Principle of global stereo, 155  
Principle of local stereo, 146  
Prototype-based learning, 341

**R**

Radiometric distortions, 161  
Radius-based clustering, 70  
Reductionism, 5  
Regularized linear least-squares regression, 252  
Repeatability, 26  
Reversible jump, 277, 288, 298  
Ridge-regression, 252  
Room layout estimation, 277, 278, 303

**S**

Scene categorization, 341  
Scene-specific detectors, 378  
Segmentation-based stereo, 167  
Self-similarity, 413, 416  
Semi-global matching, 157  
Semi-supervised learning, 35  
SfM, 184  
SFOP, 17  
Shape, 9  
Shape optimization, 215  
Shape priors, 217  
SIFT descriptor, 21  
SIFT detector, 17  
Simple trees, 158  
Simultaneous localization and mapping, 197  
SLAM, 197  
Slanted surfaces, 152  
Sliding window, 377  
Smoothness term in stereo, 164  
Soft assignment, 79  
Sparse trajectories, 98  
Spatial features, 8  
Spatio-temporal features, 12, 73  
Star Keypoint detector, 19  
Stationary camera, 378  
Stereo matching problem, 143  
Stereo Model, 160  
Structuralism, 4

Structure from motion, 184  
Structure tensor, 15  
Super-resolution, 412, 418  
SURF descriptor, 23  
SURF detector, 18  
Surface mesh, 187  
Surface Stereo, 169  
Surrogate risk, 341  
SUSAN detector, 19  
SUSurE descriptor, 23  
SUSurE detector, 19  
Symmetry-growing, 131

**T**

Tamura features, 10  
Template matching, 21  
Temporal resolution, 411, 415  
Temporal super-resolution, 412, 418  
Texture, 10  
Theory of Ecological Optics, 4  
Total variation, 218  
Trajectory augmentations, 105  
Trajectory descriptors, 98  
Trajectory features, 96  
Transfer learning, 248, 258, 270  
TransientBoost, 381  
Tree dynamic programming, 157

**U**

U-SURF descriptor, 23  
Universal Nearest Neighbors, 341  
Unsupervised action matching and segmentation, 135  
Unsupervised object detection and segmentation, 124  
Unsupervised object discovery, 114

**V**

Vanishing points, 277, 279  
Video compression, 427  
Video enhancement, 411  
Virtual cameras, 199  
Virtual views, 199  
Visual feature, 2  
Visual landmark, 183

**W**

Window size in local stereo, 147

**Z**

Zero-shot classification, 258, 271