

Geometric Level Set Methods

in Imaging,

Vision, and

Graphics

STANLEY OSHER NIKOS PARAGIOS *Editors*

Geometric Level Set Methods in Imaging, Vision, and Graphics

Springer

New York

Berlin

Heidelberg

Hong Kong

London

Milan

Paris

Tokyo

Stanley Osher Nikos Paragios

Geometric Level Set Methods in Imaging, Vision, and Graphics

With 195 Illustrations



Springer

Stanley Osher
Department of Mathematics
UCLA
520 Portola Plaza, Rm. 6363
Los Angeles, CA 90095-1555
USA

Nikos Paragios
Imaging and Visualizing Department
Siemens Corporate Research
755 College Road East
Princeton, NJ 08540
USA

Library of Congress Cataloging-in-Publication Data
Osher, Stanley.

Geometric level set methods in imaging, vision, and graphics / Stanley Osher, Nikos Paragios.
p. cm.
Includes bibliographical references and index.
ISBN 0-387-95488-0 (alk. paper)
1. Image processing—Digital techniques. I. Paragios, Nikos. II. Title.
TA1637.084 2003
621.36'7—dc21

2003045583

ISBN 0-387-95488-0 Printed on acid-free paper.

© 2003 Springer-Verlag New York, Inc.

All rights reserved. This work may not be translated or copied in whole or in part without the written permission of the publisher (Springer-Verlag New York, Inc., 175 Fifth Avenue, New York, NY 10010, USA), except for brief excerpts in connection with reviews or scholarly analysis. Use in connection with any form of information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed is forbidden.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

Printed in the United States of America.

9 8 7 6 5 4 3 2 1 SPIN 10876495

Typesetting: Pages created by the authors using a Springer T_EX macro package.

www.springer-ny.com

Springer-Verlag New York Berlin Heidelberg
A member of BertelsmannSpringer Science+Business Media GmbH

Contents

Preface	xv
List of Contributors	xxi
I Level Set Methods & Lagrangian Approaches	1
1 Level Set Methods	3
Stanley Osher	
1.1 Introduction	3
1.2 Level Set Dictionary and Technology	5
1.3 Numerical Methods	7
1.4 Imaging Science Applications	12
1.4.1 Dynamic Visibility	13
1.4.2 Interpolation from Unorganized Points via the Level Set Method	14
1.4.3 PDE's on Implicit Surfaces	14
1.4.4 The Level Set Method Links Active Contours, Mumford-Shah Segmentation and Total Variation Restoration	14
1.4.5 Modeling Textures with Total Variation Minimization and Oscillating Patterns	16
1.5 Conclusion	20
2 Deformable Models: Classic, Topology-Adaptive and Generalized Formulations	21
Demetri Terzopoulos	
2.1 Introduction	21
2.2 Classic Deformable Models	23
2.2.1 Energy-Minimizing Snakes	23
2.2.2 Dynamic Snakes	25
2.2.3 Discretization and Numerical Simulation	26
2.2.4 Probabilistic Interpretation	27
2.2.5 Higher-Dimensional Generalizations	28

2.2.6	Connections to Curve Evolution	29
2.3	Topology-Adaptive Deformable Models	30
2.3.1	Topology-Adaptive Snakes	30
2.3.2	Topology-Adaptive Deformable Surfaces	32
2.4	Generalized Deformable Models	33
2.4.1	Dynamic NURBS	33
2.4.2	Deformable Superquadrics	36
2.5	Conclusion	40
II	Edge Detection & Boundary Extraction	41
3	Fast Methods for Implicit Active Contour Models	43
Joachim Weickert and Gerald Kühne		
3.1	Introduction	43
3.2	Implicit Active Contour Models	45
3.3	Numerical Implementation	46
3.4	Experimental Results	51
3.4.1	Accuracy Evaluation in Case of Mean Curvature Motion	51
3.4.2	Efficiency Gain for Implicit Active Contour Models	51
3.5	Conclusions	54
3.6	Appendix: The Thomas Algorithm	55
4	Fast Edge Integration	59
Ron Kimmel		
4.1	Introduction	59
4.2	Mathematical Notations	60
4.3	Geometric Integral Measures for Active Contours	61
4.3.1	Alignment Term	62
4.3.2	Weighted Region	62
4.3.3	Minimal Variance	63
4.3.4	Geodesic Active Contour	63
4.4	Calculus of Variations for Geometric Measures	64
4.5	Gradient Descent in Level Set Formulation	69
4.6	Efficient Numerical Schemes	71
4.7	Examples	73
4.8	Conclusions	74
5	Variational Snake Theory	79
Agnès Desolneux, Lionel Moisan and Jean-Michel Morel		
5.1	Introduction	79
5.2	Curve flows maximizing the image contrast	83
5.2.1	The first variation of the image contrast functional	83
5.2.2	A parameterless edge equation	86

5.2.3	Numerical scheme	87
5.2.4	Choice of the function g	88
5.3	Meaningful boundaries	94
5.4	Snakes versus Meaningful Boundaries	97
III	Scalar & Vector Image Reconstruction, Restoration	101
6	Multiplicative Denoising and Deblurring: Theory and Algorithms	103
Leonid Rudin, Pierre-Louis Lions and Stanley Osher		
6.1	Introduction	103
6.2	Restoration Algorithms	105
6.3	Constrained Nonlinear Partial Differential Equations	108
6.4	Restoration of Blurry Images Corrupted by Multiplicative Noise	114
7	Total Variation Minimization for Scalar/Vector Regularization	121
Francoise Dibos, Georges Koepfler and Pascal Monasse		
7.1	Introduction	121
7.2	A global approach for Total Variation minimization	123
7.2.1	The BV function space	123
7.2.2	Presentation of the model	124
7.2.3	Validity of the model for $u \in BV(\Omega) \cap C^\infty(\Omega)$	125
7.3	A practical algorithm	128
7.3.1	The Fast Level Sets Transform	128
7.3.2	Total Variation Minimization with the FLST	132
7.3.3	TV Minimization property	133
7.3.4	Noise removal and other properties	135
7.4	Experimental results	135
7.4.1	Comparison to other models and algorithms	135
7.4.2	Application to gray scale images	137
7.4.3	Applications to color images	137
7.4.4	Optical flow regularization	139
8	Morphological Global Reconstruction and Levelings: Lattice and PDE Approaches	141
Petros Maragos		
8.1	Introduction	141
8.2	Multiscale Levelings and Level Sets	143
8.3	Multiscale Image Operators on Lattices	145
8.3.1	Multiscale Operators on Complete Lattices	146
8.3.2	Image Operators on Reference Semilattices	147
8.4	Multiscale Triphase Operators and Levelings	148
8.5	Partial Differential Equations	150
8.5.1	PDEs for 1D Levelings and Semilattice Erosions	150

8.5.2 PDEs for 2D Levelings and Semilattice Erosions	153
8.6 Discussion	153
IV Grouping	157
9 Fast Marching Techniques for Visual Grouping & Segmentation	159
Eftychis Sifakis and George Tziritas	
9.1 Introduction	159
9.2 The Multi-Label Fast Marching algorithm	161
9.2.1 The stationary level set equation	161
9.2.2 Multiple interface extensions and Fast Marching algorithm	161
9.2.3 Fast Marching algorithm and labeling	162
9.2.4 Label propagation	164
9.3 Colour and texture segmentation	166
9.3.1 Texture and colour description	166
9.3.2 Automatic feature extraction	167
9.3.3 Initial level sets	168
9.3.4 Label propagation	169
9.4 Change detection	170
9.4.1 Inter-frame difference modeling	170
9.4.2 Level set-based labeling	171
9.5 Conclusion	174
10 Multiphase Object Detection and Image Segmentation	175
Luminita Vese	
10.1 Introduction	175
10.2 Variational models for image segmentation and image partition	178
10.3 Level set formulations of minimization problems on $SBV(\Omega)$.	179
10.4 Experimental results	185
10.5 Conclusion	193
11 Adaptive Segmentation of Vector Valued Images	195
Mikaël Rousson and Rachid Deriche	
11.1 Introduction	195
11.2 The segmentation problem	196
11.3 On finding the minima	198
11.3.1 First approach: extension of the integral terms to all the domain	198
11.3.2 Second approach: direct derivation using shape derivation principle	199
11.4 Experiments	200
11.4.1 Gray-valued images	200

11.4.2	Color images	200
11.4.3	Color image sequences	203
11.4.4	Implementation remarks	203
11.5	Generalization to N regions	204
11.6	Conclusion and Future Work	205
12	Mumford-Shah for Segmentation and Stereo	207
Anthony Yezzi, Stefano Soatto, Hailin Jin, Andy Tsai, Alan Willsky		
12.1	Introduction	207
12.2	Mumford-Shah based curve evolution	209
12.2.1	Image estimates and boundary-value stochastic processes	210
12.2.2	Gradient flows to minimize the Mumford-Shah functional	212
12.2.3	Remarks on the Mumford-Shah active contour model	212
12.3	Mumford-Shah on a Moving Manifold: Stereoscopic Segmentation	216
12.3.1	Notation	216
12.3.2	Cost functional	218
12.3.3	Evolution equation	219
12.3.4	Estimating scene radiance	221
12.4	Implementation	221
12.4.1	Level set implementation	222
12.4.2	Second fundamental form	222
12.4.3	Radiance estimation	223
12.5	Experiments	223
V	Knowledge-based Segmentation & Registration	229
13	Shape Analysis towards Model-based Segmentation	231
Nikos Paragios and Mikael Rousson		
13.1	Introduction	231
13.2	Shape Modeling	232
13.3	Shape Registration	236
13.3.1	Sum of Square Differences	237
13.3.2	Robust Estimators	239
13.3.3	Global-to-Local Registration	241
13.3.4	Experimental Results & Validation	242
13.4	Segmentation & Shape Prior Constraints	242
13.4.1	Prior Global Shape Constraints	244
13.4.2	Visual Support, Shape Constraints & Segmentation	247
13.5	Discussion	249
14	Joint Image Registration and Segmentation	251

Baba Vemuri and Yunmei Chen	
14.1 Introduction	251
14.1.1 Previous Work: Brief Review	252
14.1.2 Overview of Our Algorithms	253
14.2 The PDE-based Approach	254
14.3 The Variational Approach	256
14.3.1 Geometric Active Contour with a Shape Prior	257
14.3.2 Mumford-Shah Model with a Shape Prior	259
14.4 Experimental Results and Applications	260
14.4.1 Results from the PDE-based Approach	260
14.4.2 Results from the Variational Approach	262
15 Image Alignment	271
Francoise Dibos, Georges Koepfler and Pascal Monasse	
15.1 Introduction	271
15.1.1 Plan of the chapter	271
15.1.2 Generalities	272
15.1.3 Correlation methods	273
15.1.4 Features matching	273
15.1.5 Overview of the method	273
15.2 Correspondences	274
15.2.1 Choice of features	274
15.2.2 Characteristics	274
15.2.3 Finding correspondences	276
15.3 Votes	277
15.4 Accuracy	278
15.5 Complexity	280
15.6 Projective registration	280
15.6.1 A 3D representation for projective deformation	281
15.6.2 The reciprocity principle	282
15.6.3 The registration group	283
15.6.4 An algorithm for the matches.	284
15.7 Experimental results	284
15.7.1 Pose estimation	284
15.7.2 Similarity	289
15.7.3 Accuracy	290
15.7.4 Projective registration	292
VI Motion Analysis	297
16 Variational Principles in Optical Flow Estimation and Tracking	299
Nikos Paragios and Rachid Deriche	
16.1 Introduction	299
16.2 Geodesic Active Regions	301

16.2.1	Setting the Boundary Module	302
16.2.2	Setting the Region Module	302
16.2.3	Geodesic Active Region Objective function	304
16.2.4	<i>N – Partition</i> Geodesic Active Regions & Tracking	305
16.3	Optical Flow Estimation & Tracking	305
16.3.1	Boundary & Smoothness Component	305
16.3.2	Background Subtraction tracking	306
16.3.3	Visual Consistency	308
16.3.4	Implementation Issues	313
16.4	Complete Recovery of the Apparent Motion	314
16.5	Discussion	315
17	Region Matching and Tracking under Deformations or Occlusions	319
Stefano Soatto, Anthony Yezzi and Alessandro Duci		
17.1	Introduction	319
17.1.1	Prior related work	321
17.2	Defining motion and shape average	323
17.3	Shape and deformation of a planar contour	325
17.3.1	Representation of motions	326
17.3.2	Variation with respect to the group action	326
17.3.3	Evolution	327
17.3.4	Distance between shapes	328
17.4	Moving average and tracking	328
17.5	Averaging and registering non-equivalent shapes	329
17.6	Matching with missing parts	330
17.6.1	Minimization with respect to shape	331
17.6.2	Minimization with respect to the group action	331
17.6.3	Evolution equations	332
17.6.4	Generalization to graylevel images	333
17.7	Experiments	335
VII	Computational Stereo & Implicit Surfaces	341
18	Computational Stereo: A Variational Method	343
Olivier Faugeras, Jose Gomes and Renaud Keriven		
18.1	Introduction and preliminaries	343
18.2	The simplified models	347
18.2.1	A simple object and matching model	347
18.2.2	Fronto parallel correlation functional	348
18.2.3	Taking into account the tangent plane to the object	349
18.3	The complete model	351
18.3.1	Error criterion	352
18.3.2	The Euler-Lagrange equations	353
18.3.3	The normal velocity and the regularization property	354

18.3.4	Correctness	354
18.4	Level Set Implementation	355
18.4.1	The velocity extension problem	355
18.4.2	Preserving the distance function	356
18.4.3	Error criterion	356
18.4.4	Visibility	356
18.4.5	Why and how well does it work?	357
18.4.6	A simple but important case	357
18.5	Results	359
18.6	Conclusion	360
19	Visualization, Analysis and Shape Reconstruction of Sparse Data	361
Hongkai Zhao and Stanley Osher		
19.1	Introduction	361
19.2	Fast multiscale visualization and analysis of large data sets using distance functions	363
19.2.1	The distance function and the fast sweeping method	364
19.2.2	Dissection and Characterization of Disconnected Components	365
19.2.3	Extraction of distance contours and visualization of the data set	368
19.2.4	Examples	369
19.3	Construction of implicit surfaces using the level set method	372
19.3.1	The Weighted Minimal Surface Model	373
19.3.2	The Convection Model	374
19.3.3	The Level Set Formulation	375
19.3.4	Finding a good initial guess	376
19.3.5	Multiresolution and Efficient Storage	378
19.3.6	Numerical Implementations and Examples	378
20	Variational Problems and Partial Differential Equations on Implicit Surfaces: Bye Bye Triangulated Surfaces?	381
Marcelo Bertalmío, Facundo Mémoli, Li-Tien Cheng, Guillermo Sapiro and Stanley Osher		
20.1	Introduction	381
20.1.1	The background and our contribution	383
20.2	The framework	384
20.2.1	Surface and data representation	384
20.2.2	A simple example: Heat flow on implicit surfaces	385
20.2.3	From variational problems to PDE's	388
20.2.4	Anisotropic diffusion on implicit surfaces	388
20.3	Experimental examples	389
20.3.1	Diffusion of scalar images on surfaces	389
20.3.2	Diffusion of directional data on surfaces	391

20.3.3	Pattern formation on surfaces via reaction-diffusion flows	392
20.3.4	Flow visualization on 3D surfaces	394
20.4	Concluding remarks	395
VIII	Medical Image Analysis	399
21	Knowledge-Based Segmentation of Medical Images	401
Michael Leventon, Eric Grimson, Olivier Faugeras, Ron Kikinis and William Wells III		
21.1	Introduction	401
21.2	Probability distribution on shapes	403
21.2.1	Curve representation	403
21.2.2	The correspondence problem	406
21.3	Shape priors and geodesic active contours	407
21.3.1	Geodesic active contours for segmentation	407
21.3.2	Estimation of pose and shape	408
21.3.3	Evolving the surface	411
21.4	Statistical Image-Surface Relationship	411
21.4.1	Intensity Model	413
21.4.2	Curvature Model	414
21.4.3	Surface Estimation	418
21.5	Results	419
21.6	Conclusions	420
22	Topology Preserving Geometric Deformable Models for Brain Reconstruction	421
Xiao Han, Chenyang Xu and Jerry Prince		
22.1	Introduction	421
22.1.1	Digital topology	424
22.1.2	Isocontour algorithms	424
22.2	Topology Preserving Geometric Deformable Model	426
22.2.1	Algorithm	426
22.2.2	Examples	428
22.3	Brain Cortical Surface Reconstruction	431
22.3.1	Background	431
22.3.2	Preprocessing and surface initialization	433
22.3.3	Nested surface reconstruction algorithm	433
22.3.4	Results	436
22.4	Conclusion	438

IX Simulations & Graphics	439
23 Editing Geometric Models	441
Ken Museth, Ross Whitaker and David Breen	
23.1 Introduction	441
23.1.1 New Surface Editing Operators	442
23.1.2 Benefits and Issues	443
23.1.3 Contributions	444
23.2 Previous Work	445
23.3 Overview of the Editing Pipeline	446
23.3.1 Input and Output Models	446
23.4 Level Set Surface Modeling	447
23.4.1 Level Set Speed Function Building Blocks	447
23.4.2 Regionally Constraining Level Set Deformations	448
23.4.3 Limiting Geometric Property Values	449
23.4.4 Constraining the Direction of LS Motions	450
23.5 Definition of Surface Editing Operators	450
23.5.1 CSG Operations	450
23.5.2 Automatic Localized Level Set Blending	451
23.5.3 Localized Level Set Smoothing/Sharpening	452
23.5.4 Point Set Attraction and Embossing	455
23.5.5 Global Morphological Operators	455
23.5.6 Editing Session Details	456
23.6 Conclusion and Future Work	458
23.7 Appendix: Curvature of Level Set Surfaces	459
24 Simulating Natural Phenomena	461
Ronald Fedkiw	
24.1 Introduction	461
24.2 Smoke	463
24.3 Water	468
24.4 Fire	474
Bibliography	481
References	481

Preface

Introduction

Image processing, computer vision and computer graphics are now established research areas. Pattern recognition and artificial intelligence were the origins of the exploration of the space of images. Simplistic digital techniques used at the beginning of 60's for gray image processing operations have been now replaced with a complex mathematical framework that aims to exploit and understand images in two and three dimensions. Advances in computing power continue to make the use and processing of visual information an important part of our lives.

The evolution of these techniques was a natural outcome of the need to process an emerging information space, the space of natural images. Images in space and time are now a critical part of many human activities. First, pictures and now video streams were used to eternalize small and significant moments of our life. Entertainment including movies, TV-programs and video games are part of our every-day life where capturing, editing, understanding and transmitting images are issues to be dealt with. The medical sector is also a major area for the use of images. The evolution of the acquisition devices led to new ways of capturing information, not visible by the human eye. Medical imaging is probably the most established market for processing visual information [405]. Visualization of complex structures and automated processing towards computer aided diagnosis is used more and more by the physicians in the diagnostic process. Safety and security are also important areas where images and video play a significant role [432]. Security cameras huge amount of data that cannot be processed by the human eye. Automated techniques for event detection are now successfully deployed to prevent and inform us of life threatening events. Augmented reality, robotics and computer graphics are also areas where the use images is a crucial. Simulation of events using graphics techniques like special effects, is oftenly used by the entertainment industry (movies, games). Similar simulation and graphics techniques are important to drug discovery. Autonomous or semi-autonomous vision systems can replace humans in hostile environments and allow them to deal with critical situations while the observer stays at a safe remote location. There are many other areas which can benefit from the use of images.

Applied Mathematics in Imaging, Vision and Graphics

Pure and applied mathematics has been the basis for a number of tools dealing with the complete understanding of images. Different formulations and techniques (statistical, graph theory-based, harmonic analysis, variational, etc.) as well as philosophies are part of the active literature of the past two decades. Their classification is not straightforward. Image processing, Computer Vision and Graphics have also benefited from recent progress in other discipline areas such as statistics, numerical analysis, computational physics, etc. Variational methods and partial differential equations are increasingly important tools for dealing with image understanding related applications.

Many problems in image understanding reduce an optimization problem with the objective being to recover a set of unknown variables that can lead towards partial or complete understanding. Variational formulations [458, 24] are part of such techniques where the estimation of the unknown variables is done iteratively. Image denoising and restoration is an important example where, given some corrupted data, one would like to recover the original signal. One can also consider other applications such as image reconstruction, segmentation and registration, motion and shape analysis, computational stereo as core components needed to understand images.

Level Set Methods

Most of these applications can be re-formulated as problems of tracking moving interfaces. Image segmentation is perhaps the most well studied topic in image processing and computer vision. This involves understanding the visual data or extraction of specific objects and structures of interest that follow either some known intensity properties or some specific form. Evolving an initial curve towards the boundaries of the structure of interest is a method that can be used to deal with this problem. The evolution of curves and surfaces is also well explored technique in other fields such as computational physics. Such techniques led to a breakthrough in computer vision after the introduction of the snake model [262], presented by *Kass, Witkin and Terzopoulos* that aimed at evolving an initial curve towards the lowest potential of a cost function. The definition of the cost function can vary according to the application and can consist of a term that accounts for the visual support and a term that imposes some internal geometric (regularity) constraints.

The snake model was the origin of a significant number of mathematical formulations that have been used in imaging, vision and graphics during the last 15 years [50, 360]. Level set techniques [401], the most suitable method to track moving interfaces, were introduced in parallel by *Osher and Sethian* motivated by applications and geometry. Such techniques are implicit, intrinsic, parameter-free and can deal with topological changes while for tracking moving interfaces.

People in imaging, begun using these techniques at the beginning of 90s [81, 330] and they have now became an established area of research in vision and graphics as well. Books [472, 458, 398] and special sessions and workshops [556] dedicated to this area are now a basic part of the vision community. The objective of this book is to present the theoretical foundations of these techniques, appropriate numerical tools and a collection of topics that demonstrate their use in the areas of imaging, vision and graphics.

Contributions

This edited volume consists of 24 chapters. The most well known researchers in the field (47), from 32 world-wide leading academic institutions and industrial research centers contributed to this volume; Stanfod, Caltech, UC-LA, UC-Irvine, UC-San Diego, Gatech, UUT, UFL, WU (St. Louis), JHU, NYU, MIT, Harvard, Brigham and Women's Hospital, UMN (*US*), U-Paris 5, U-Paris 9, Ecole Normale Supérieure-Paris, Ecole Normale Supérieure-Cashan, Ecole de Ponts, INRIA (*France*), University of Mennheim, University of Saarland (*Germany*), University of Pompeau-Farba (*Spain*), Scuola Normale Supérieure (*Italy*), National Technical University of Athens (*Greece*), University of Crete (*Greece*), Technion (*Israel*), Universidad de la Repblica (*Uruguay*), Cognitech, IBM and Siemens Corporate Research. The book is partitioned into 9 thematic areas.

Level Set and Langrangian Methods

In the introductory part, the theoretical foundations of these methods, appropriate numerical techniques and a flavor of applications in imaging, vision and graphics is presented by OSHER. The Lagrangian technique through Deformable models is a popular, active, alternative of the Level Set Method initially explored by TERZOPOULOS and used by many other researchers. Discussion/Comparison between these two methods in evolving interfaces will be presented by TERZOPOULOS.

Edge Detection and Boundary Extraction

Edge detection is a core component in low level vision. Such a module can be the basis of high level vision tasks leading to complete image understanding. The second thematic area consists of three chapters dedicated to boundary extraction. The first chapter by WEICKERT AND KUEHNE presents novel numerical approximation schemes for the Level Set Method in boundary extraction. The proposed technique is more stable and computational efficient. The other two chapters introduce a novel variational formulation to detect continuous edges (DESOLNEUX, MOISSAN AND MOREL) and link these techniques with existing edge detectors (KIMMEL). Dealing with noisy and physically corrupted data is usually a limitation of most of the existing boundary extraction methods.

Scalar and Vector Image Reconstruction, Restoration

Image enhancement/reconstruction, noise reduction in scalar and vector images is a fundamental area of research in imaging. This topic is addressed in this book for scalar and vector images. RUDIN, LIONS AND OSHER propose a total variation minimization approach to image restoration, an excellent unpublished piece of work that was written about a decade ago. A slightly different problem is considered by DIBOS, MONASSE AND KOEPFLER , the regularization of scalar and vector images and is dealt with by a total variation minimization approach, while a connection between the level set method and self-dual morphological reconstruction operators is established by MARAGOS. Spatial consistency/smoothness in the data domain is a widely explored constraint in the reconstruction process. A step further is the introduction of more global constraints that aim at partitioning the image in regions that correspond to different objects in the real world.

Grouping

A more sophisticated task in imaging related to boundary extraction, is image segmentation and grouping. The assumption is that objects appear in the image refer to similar visual characteristics and one would like to recover a complete separation of the objects present in the image. This concern is adressed in the following 4 chapters. SIFAKIS AND TZIRITAS introduce a multi-label implementation of the Fast Marching algorithm that is based on Bayesian perinciples. VESE proposes an unconstrained segmentation approach to detect objects based on a level set implementation of the Mumford-Shah [387] framework. ROUSSON AND DERICHE introduce an adaptive level set method for the segmentation of vector-valued images and YEZZI, SOATTO, JIN, TSAI AND WILLSKY introduce extensions of Mumford-Shah framework to deal with the 3D case.

Knowledge-based Segmentation and Registration

The extraction of structures of interest, a particular case of segmentation, that requires shape modeling and registration, is a topic addressed by PARAGIOS AND ROUSSON. Introduction of global prior shape knowledge can imporve the segmentation performance of the level set method and deal with corrupted, occluded and incomplete data. Registration is also a core component in medical image analysis that can be also addressed jointly with segmentation as shown by VEMURI AND CHEN. Pure scalar and vector-valued image alignment techniques were considered using the level set method as shown by DIBOS, MONASSE AND KOEPFLER.

Motion Analysis

Real-time acquisition requires processing of images in the temporal domain. Static images can provide only limited information when solving problems in

imaging, vision and graphics. On the other hand the use of the time domain, can provide information and account on the dynamic behavior of the objects. Motion is an important visual cue to many different applications, giving the user the ability to track and observe the deformations of a structure of interest across time. PARAGIOS AND DERICHE propose a level set framework for motion estimation and tracking where the objective is to recover the trajectories and the temporal positions of the moving structures. A different problem is considered by SOATTO, YEZZI AND DUCI where objects undergoing heavy local and global deformations are detected and tracked in a dynamic fashion.

Computational Stereo and Implicit Surfaces

Motion analysis is a step towards improved understanding of images. Complete image understanding is associated with the recover of the real structure of the observed scene. Images correspond to projections into the 2D plane of 3D scenes according to some intrinsic camera transformation. One camera can provide limited information regarding the real 3D structure. However, a combination of several cameras that observed the same 3D environment can lead to partial or complete recovery of the 3D properties of the scene. FAUGERAS, KERIVEN AND GOMES propose a variational formulation for computational stereo that processes the input from multiple cameras, while ZHAO AND OSHER introduce a level set technique for stereo reconstruction when the limitation of sparse 3D data is to be addressed. Evolution on implicit surfaces can be viewed as a step further to computational stereo. BERTALMIO, MÉMOLI, CHENG, SAPIRO AND OSHER introduce a novel level set technique to solve variational problems and differential equations that are defined in a surface for scalar and vector-valued data. For example the problem of 3D texture mapping goes beyond computational stereo and "lives" on an implicit surface.

Medical Image Analysis

Medical imaging is a rapidly evolving area, where there is a strong need to understand scalar or vector-valued images. The outcome of such processing can have strong diagnostic implications and can be used as an additional tool to detect and treat in a timely and proper fashion different diseases. LEVENTON, GRIMSON, FAUGERAS, WELLS AND KIKINIS propose a level set approach for knowledge-based segmentation in medical image analysis while HAN, XU AND PRINCE introduce a topology preserving deformable model to deal with the segmentation and reconstruction of brain images.

Graphics and Simulations

Computer graphics is an established research in which the Level Set Method has led to major advances. These techniques are suitable in graphics and simulations,

since they are implicit, intrinsic and parameter free. Moreover, since they do not require some specific "parameterization/discretization" techniques, they can be used in editing and refining geometric models as proposed by MUSETH, BREEN AND WHITAKER. Dealing with non-solid entities like water, smoke, etc. is difficult to be done efficiently with model-based approaches. FEDIKIW introduces a level set formulation in simulating natural phenomena based on solving the real equations of physics.

Acknowledgments

The editors would like to thank all contributors for accepting and participating in this book. The effort was completed successfully mainly because of their prompt reaction. Most of the contributions refer to unpublished work or to techniques that now are the state-of-the art in imaging, vision and graphics. Due to the lack of space, only a small number of leading researchers were invited to contribute. The editors would like to acknowledge that many other outstanding contributions are available and apologize to scientists who have been omitted. Last, but not least, the editors would like to thank Wayne Wheeler and Wayne Yuhasz from the Computing and Information Science department of Springer New York for supporting this project.

Stanley Osher would like to thank his numerous colleagues and students at UCLA for providing a unique research environment over the past two decades. He would also like to thank the various granting agencies: NSF, AFOSR, ARO, NIH, DARPA, NASA and particularly ONR for their support.

Nikos Paragios would like to acknowledge the support of Siemens Corporate Research and his colleagues - Gareth Funka-Lea, Bernhard Geiger, Thomas Grandke, Alok Gupta, Marie-Pierre Joly, Thomas O'Donnell and Visvanathan Ramesh - during the preparation of this volume. He is also grateful to his former students and collaborators, in particular to Mikael Rousson, Rachid Deriche and Olivier Faugeras.

List of Contributors

Bertalmío, Marcelo

Dept. de Tecnología
Universitat Pompeu Fabra
Passeig de Circumvalació, 8
08003 Barcelona, Spain
mail-to:marcelo.bertalmio@tecn.upf.es
<http://www.iua.upf.es/~mbertalmio/>

Breen, David

Center for Advanced Computing Research
107 Powell-Booth, MS 107-79
California Institute of Technology
Pasadena, CA 91125, USA
mail-to:david@cacr.caltech.edu
<http://www.cacr.caltech.edu/~david/>

Chen, Yunmei

Department of Mathematics
University of Florida, 458 Little Hall
Gainesville, FL 32611-8105 USA
mail-to:yun@math.ufl.edu
<http://www.math.ufl.edu/~yun/>

Cheng, Li-Tien

Department of Mathematics
University of California, San Diego
9500 Gilman Drive,
La Jolla, California 92093-0112, USA
mail-to:lcheng@math.ucsd.edu
<http://www.ucsd.edu/~lcheng/>

Deriche, Rachid

INRIA Sophia Antipolis
BP. 93, 2004 route des Lucioles
06902 Sophia Antipolis Cedex, France
mail-to:Rachid.Deriche@inria.fr
<http://www-sop.inria.fr/odyssee/team/Rachid.Deriche/index.en.html>

Desolneux, Agnès

MAP 5, UFR de Math-Info
Université Paris 5
45 rue des Saints-Pères
75270 Paris cedex 06, France
mail-to:desolneux@math-info.univ-paris5.fr
<http://www.math-info.univ-paris5.fr/~desolneux/>

Dibos, Francoise

CEREMADE, Université Paris 9-Dauphine
Place du Maréchal de Lattre de Tassigny
75775 Paris Cedex, France
mailto:dibos@ceremade.dauphine.fr
<http://www.ceremade.dauphine.fr/~fd/>

Duci, Alessandro

Scuola Normale Superiore
Piazza dei Cavalieri, 7, Pisa 56100, Italy
mailto:alessandro.duci@sns.it

Faugeras, Olivier

INRIA Sophia Antipolis
BP. 93, 2004 route des Lucioles
06902 Sophia Antipolis Cedex, France
mail-to:Olivier.Faugeras@inria.fr
<http://www-sop.inria.fr/odyssee/team/Olivier.Faugeras/index.en.html>

Fedkiw, Ronald

Computer Science Department
Stanford University
Gates Computer Science Bulding, Room 207
Stanford, CA 94305-9020, USA
mailto:fedkiw@cs.stanford.edu
<http://graphics.stanford.edu/~fedkiw/>

Gomes, Jose

IBM Watson Research Center
1101 Kitchawan Road, Route 134
Yorktown Heights, NY 10598, USA
mail-to:josegome@us.ibm.com

Grimson, Eric

Artificial Intelligence Laboratory
Massachusetts Institute of Technology
400 Technology Square, Cambridge, MA 02139, USA
mail-to:welg@ai.mit.edu
<http://www.ai.mit.edu/people/welg/welg.html>

Han, Xiao

Department of Electrical and Computer Engineering
105 Barton Hall,
Johns Hopkins University
3400 North Charles St.
Baltimore, MD 21218, USA
mail-to:xhan@iron.ece.jhu.edu
<http://iacl.ece.jhu.edu/~xhan/>

Jin, Hailin

Department of Electrical Engineering
Washington University
Campus Box 1161, One Brookings Drive
St. Louis, MO 63130-4899, USA
mail-to:hjin@ee.wustl.edu
<http://ee.wustl.edu/~hjin/>

Keriven, Renaud

Ecole Normale Supérieure
Département d’Informatique
45 Rue d’Ulm
75230 Paris, France
mail-to:Renaud.Keriven@ens.fr
<http://cermics.enpc.fr/~keriven/home.html>

Kikinis, Ron

Surgical Planning Laboratory
AMB II, L1-Room 0069, Radiology
Brigham and Women's Hospital
75 Francis St., Boston, MA 02115, USA
mail-to:kikinis@bwh.harvard.edu
<http://splweb.bwh.harvard.edu:8000/pages/ppl/kikinis/>

Kimmel, Ron

Computer Science Department, Technion
518 Taub (new CS) Bldg.
Haifa 32000, Israel
mail-to:ron@cs.technion.ac.il
<http://www.cs.technion.ac.il/~ron/>

Koepfler, Georges

MAP5, UFR de Mathématiques et Informatique
Université René Descartes – Paris 5
45, rue des Saints-Pères
75270 Paris cedex 06, France
mail-to:gk@math-info.univ-paris5.fr
<http://www.math-info.univ-paris5.fr/~gk/>

Kuehne, Gerald

University of Mannheim
L15, 16 68131 Mannheim, Germany
mail-to:kuehne@informatik.uni-mannheim.de

Leventon, Michael

Artificial Intelligence Laboratory
Massachusetts Institute of Technology
77 Massachusetts Avenue, Cambridge, MA 02139, USA
mail-to:leventon@alu.mit.edu
<http://www.ai.mit.edu/people/leventon/>

Lions, Pierre-Louis

Ecole Normale Supérieure
45 rue d'Ulm, 75230 Paris Cedex 05, France
mail-to:pierre-louis.lions@ens.fr

Maragos, Petros

National Technical University of Athens
School of Electrical and Computer Engineering
GR-157 73 Zografou, Athens, Greece
mail-to:maragos@cs.ntua.gr

Memoli, Facundo

Department of Electrical and Computer Engineering
University of Minnesota, 200 Union Street SE
Minneapolis, MN 55455, USA
mail-to:memoli@ece.umn.edu
<http://www.ece.umn.edu/users/memoli/>

Moisan, Lionel

Centre de Mathématiques et de Leurs Applications
Ecole Normale Supérieure de Cachan
61 avenue du président Wilson
94235 Cachan cedex, France
mail-to:moisan@cmla.ens-cachan.fr
<http://www.cmla.ens-cachan.fr/Utilisateurs/moisan/>

Monasse, Pascal

Cognitech, Inc
225 S.Lake Ave. Suite 601
Pasadena, CA 91101, USA
mail-to:pascal.monasse@free.fr
<http://pascal.monasse.free.fr/>

Morel, Jean-Michel

Centre de Mathématiques et de Leurs Applications
Ecole Normale Supérieure de Cachan
61 avenue du président Wilson
94235 Cachan cedex, France
mail-to:Jean-Michel.Morel@cmla.ens-cachan.fr
<http://www.cmla.ens-cachan.fr/Cmla/InfoMembre/Fiche/morel.html>

Museth, Ken

Caltech Computer Graphics Laboratory
Computer Science Department
Beckman Institute, MS 139-74
California Institute of Technology
Pasadena, California 91125, USA
mail-to:kmu@caltech.edu
<http://www.gg.caltech.edu/~kmu>

Osher, Stanley

Department of Mathematics
UCLA, Box 951555
Los Angeles, CA 90095-1555, USA
mail-to:sjo@math.ucla.edu
<http://www.math.ucla.edu/~sjo/>

Paragios, Nikos

Siemens Corporate Research
755 College Road East
Princeton, NJ 08540, USA
mail-to:nikos@scr.siemens.com
<http://mywebpages.comcast.net/aparagio/>

Prince, Jerry

Department of Electrical and Computer Engineering
105 Barton Hall,
Johns Hopkins University
3400 North Charles St.
Baltimore, MD 21218, USA
mail-to:prince@jhu.edu
<http://iacl.ece.jhu.edu/~prince/>

Rousson, Mikael

INRIA Sophia Antipolis
BP. 93, 2004 route des Lucioles
06902 Sophia Antipolis Cedex, France
mail-to:Mikael.Rousson@sophia.inria.fr
<http://www-sop.inria.fr/odyssee/team/Mikael.Rousson/index.fr.html>

Rudin, Leonid

Cognitech, Inc
225 South Lake Avenue, Suite 601
Pasadena, CA 91101, USA
mail-to:rudin@cognitech.com

Sapiro, Guillermo

Department of Electrical and Computer Engineering
University of Minnesota, 200 Union Street SE
Minneapolis, MN 55455, USA
mail-to:guille@ece.umn.edu
<http://www.ece.umn.edu/users/guille/>

Sifakis, Eftychis

Department of Computer Science
Stanford University, 353 Serra Mall
Stanford CA 94305-9025, USA
mail-to:sifakis@cs.stanford.edu

Soatto, Stefano

Computer Science Department
UCLA, Boelter hall 3531d
Los Angeles, CA 90095-1596, USA
mail-to:soatto@cs.ucla.edu
<http://www.cs.ucla.edu/~soatto/>

Terzopoulos, Demetri

Courant Institute of Mathematical Sciences
New York University
719 Broadway, 12th Floor
New York, NY 10003, USA
mail-to:dt@cs.nyu.edu
<http://www.mrl.nyu.edu/~dt/>

Tsai, Andy

Department of Electrical Engineering
Massachusetts Institute of Technology
Room 35-427
77 Massachusetts Avenue, Cambridge, MA 02139, USA
mail-to:atsai@alum.mit.edu
<http://ssg.mit.edu/group/atsai/>

Tziritas, Georgios

Department of Computer Science, University of Crete
P.O.Box 2208, Heraklion, Crete, GR-714 09, Greece
mail-to:tziritas@csd.uch.gr
<http://www.csd.uch.gr/~tziritas/>

Vemuri, Baba

Professor & Director
Center for Computer Vision and Visualization
Department of CISE, PO BOX 116120
University of Florida, Gainesville, FL 32611, USA
<http://www.cise.ufl.edu/~vemuri>

Vese, Luminita Aura

Department of Mathematics, UCLA
405 Hilgard Avenue
Los Angeles, CA 90095-1555, USA
mail-to:lvese@math.ucla.edu
<http://www.math.ucla.edu/~lvese/>

Weickert, Joachim

Faculty of Mathematics and Computer Science
Saarland University
Building 27.1, 66041 Saarbrücken, Germany
mail-to:weickert@mia.uni-saarland.de
<http://www.mia.uni-saarland.de/weickert/>

Wells III, William M.

Department of Radiology
Harvard Medical School and Brigham and Women's Hospital
75 Francis St., Boston, MA 02115, USA
mail-to:sw@bwh.harvard.edu
<http://splweb.bwh.harvard.edu:8000/pages/ppl/sw/homepage.html>

Whitaker, Ross

School of Computing, University of Utah
3450 Merrill Engineering Building
Salt Lake City, UT 84112-9205, USA
mail-to:whitaker@cs.utah.edu
<http://www.cs.utah.edu/~whitaker/>

Willsky, Alan

Department of Electrical Engineering
Massachusetts Institute of Technology
77 Massachusetts Avenue, Cambridge, MA 02139, USA
mail-to:willsky@mit.edu
<http://ssg.mit.edu/group/willsky/willsky.shtml>

Xu, Chenyang

Siemens Corporate Research
755 College Road East
Princeton, NJ 08540, USA
mail-to:Chenyang.Xu@scr.siemens.com
<http://iacl.ece.jhu.edu/~chenyang/>

Yezzi, Anthony

Department of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, GA 30332, USA
mail-to:ayezzi@ece.gatech.edu
<http://www.ece.gatech.edu/profiles/ayezzi/>

Zhao, Hongai

Department of Mathematics
103 Multipurpose Science and Technology Building
University of California, Irvine
Irvine, CA 92697-3875, USA
mail-to:zhao@math.uci.edu
<http://www.math.uci.edu/~zhao/>

Part I

Level Set Methods & Lagrangian Approaches

1

Level Set Methods

Stanley Osher

Abstract

The level set method for capturing moving fronts was introduced in 1987 by Osher and Sethian [401]. It has proven to be phenomenally successful as a numerical device. For example, as of June 2002, typing in “Level Set Methods” on Google’s search engine gives roughly 2800 responses and the original article has been cited over 530 times (according to web of science). Applications range from capturing multiphase fluid dynamical flows, to graphics, e.g. special effects in Hollywood, to visualization, image processing, control, epitaxial growth, computer vision and include many others. In this chapter we shall give an overview of the numerical technology and of applications in imaging science. These will include surface interpolation, solving PDE’s on manifolds, visibility, ray tracing, segmentation (including texture segmentation) and restoration.

1.1 Introduction

The original idea behind the level set method was a simple one. Given an interface Γ in R^n of codimension one, bounding a (perhaps multiply connected) open region Ω , we wish to analyze and compute its subsequent motion under a velocity field v . This velocity can depend on position, time, the geometry of the interface (e.g., its normal or its mean curvature), and the external physics. The idea, as devised in 1987 by S. Osher and J.A. Sethian [401] is merely to define a smooth (at least Lipschitz continuous) function $\varphi(\mathbf{x}, t)$, that represents the interface as the set where $\varphi(\mathbf{x}, t) = 0$. Here $\mathbf{x} = (x_1, \dots, x_n) \in R^n$.

The level set function φ has the following properties:

$$\begin{aligned}\varphi(\mathbf{x}, t) &< 0 && \text{for } \mathbf{x} \in \Omega \\ \varphi(\mathbf{x}, t) &> 0 && \text{for } \mathbf{x} \notin \bar{\Omega} \\ \varphi(\mathbf{x}, t) &= 0 && \text{for } \mathbf{x} \in \partial\Omega = \Gamma(t).\end{aligned}$$

Thus, the interface is to be captured for all later time, by merely locating the set $\Gamma(t)$ for which φ vanishes. This deceptively trivial statement is of great significance for numerical computation, primarily because topological changes such as breaking and merging are well defined and performed “without emotional involvement”.

The motion is analyzed by convecting the φ values (levels) with the velocity field v . This elementary equation is

$$\frac{\partial \varphi}{\partial t} + \mathbf{v} \cdot \nabla \varphi = 0. \quad (1.1)$$

Here \mathbf{v} is the desired velocity on the interface and is arbitrary elsewhere.

Actually, only the normal component of v is needed, $v_N = \mathbf{v} \cdot \frac{\nabla \varphi}{|\nabla \varphi|}$, so (1.1) becomes

$$\frac{\partial \varphi}{\partial t} + v_N |\nabla \varphi| = 0. \quad (1.2)$$

Here $|\nabla \varphi| = \sqrt{\sum_{i=1}^n \varphi_{x_i}^2}$.

In those cases where v_N is a constant, equation (1.2) is nonlinear. In fact it is a first order Hamilton-Jacobi equation. If $v_N \equiv 1$ what the zero level set, $\Gamma(t)$, of $\varphi(\mathbf{x}, t)$ at later time $t > 0$ represents is the set of all points at a positive distance t from the original interface Γ . The first thing one notices is that this distance function develops “kinks”.

For example, if one starts with a square Γ and moves it outwards a unit distance, obtaining $\Gamma(1)$, then associated with each of four quarter circles of radius 1 centered at each vertex of the original square, we have a continuum of points whose closest point on Γ is the same for the whole continuum – namely the closest vertex of Γ . If we move $\Gamma(1)$ backwards one unit, we thus have a singular distance function at each vertex of Γ .

Such mild singularities are typical of the correct solutions to this Hamilton-Jacobi equation. Fortunately the theory of viscosity solutions, which was invented in [139],[137], exactly picks out this unique Lipschitz continuous solution as do the solutions to the discrete approximations developed in [401],[402],[256] and discussed below. Thus, users can have confidence that their computer simulations give accurate, unique solutions. A particularly interesting result is in [183], where motion by mean curvature, as defined by Osher and Sethian in [401], is shown to be essentially the same motion as is obtained from the asymptotics in the phase field reaction diffusion equation. The motion in the level set method involves no superfluous stiffness as required in phase field methods.

The outline of this chapter is as follows: In section 2 we present the key definition and basic level set technology. In section 3 we discuss the numerical implementation. In section 4 we briefly describe some applications in imaging science, including several very recent results. We present some concluding remarks in section 5.

1.2 Level Set Dictionary and Technology

We list key terms and define them by their level set representation.

1. The interface boundary $\Gamma(t)$ is defined by $\{\mathbf{x}|\varphi(\mathbf{x}, t) = 0\}$. The region $\Omega(t)$ is bounded by $\Gamma(t) : \{\mathbf{x}|\varphi(\mathbf{x}, t) < 0\}$ and its exterior is defined by $\{\mathbf{x}|\varphi(\mathbf{x}, t) > 0\}$.

2. The unit normal \mathbf{N} to $\Gamma(t)$ is given by

$$\mathbf{N} = \frac{\nabla \varphi}{|\nabla \varphi|}.$$

3. The mean curvature κ of $\Gamma(t)$ is defined by

$$\kappa = -\nabla \cdot \left(\frac{\nabla \varphi}{|\nabla \varphi|} \right).$$

4. The Dirac delta function concentrated on an interface is

$$\delta(\varphi)|\nabla \varphi|,$$

where $\delta(x)$ is a one-dimensional delta function.

5. The characteristic function χ of a region $\Omega(t)$ is

$$\chi = H(-\varphi)$$

where

$$\begin{aligned} H(x) &\equiv 1 && \text{if } x > 0 \\ H(x) &\equiv 0 && \text{if } x < 0 \end{aligned}$$

is a one-dimensional Heaviside function.

6. The surface (or line) integral of a quantity $P(\mathbf{x}, t)$ over Γ is

$$\int_{R^n} p(\mathbf{x}, t) \delta(\varphi) |\nabla \varphi| d\mathbf{x}.$$

7. The volume (or area) integral of $p(\mathbf{x}, t)$ over Ω is

$$\int_{R^n} p(\mathbf{x}, t) H(-\varphi) d\mathbf{x}.$$

Next we describe some key technological advances which are important in many, if not most, level set calculations.

8. The distance reinitialization procedure replaces a general level set function $\varphi(\mathbf{x}, t)$ by $d(\mathbf{x}, t)$ which is the value of the distance from \mathbf{x} to $\Gamma(t)$, positive outside and negative inside. This assures us that φ does not become too flat or too steep near $\Gamma(t)$. Let $d(\mathbf{x}, t)$ be signed distance of \mathbf{x} to the closest point on Γ . The quantity $d(\mathbf{x}, t)$ satisfies $|\nabla d| = 1$, $d > 0$ in Ω , $d < 0$ in $(\bar{\Omega})^c$, and is the steady state solution (as $\tau \rightarrow \infty$) to

$$\begin{aligned} \frac{\partial \psi}{\partial \tau} + \operatorname{sgn}(\varphi)(|\nabla \psi| - 1) &= 0 \\ \psi(\mathbf{x}, 0) &= \varphi(\mathbf{x}, t), \end{aligned} \tag{1.3}$$

where $\text{sgn}(x) = 2H(x) - 1$ is the one-dimensional signum function. This procedure was designed in [501]. The key observation is that in order to define d in a band of width ϵ around Γ , we need solve (1.3) only for $\tau = 0(\epsilon)$. It can easily be shown that this can be used globally to construct distance (with arbitrary accuracy) in $O(N \log N)$ iterations [421]. Alternatively, we may use Tsitsiklis' fast algorithm [532, 533], which is also $O(N \log N)$, with a much smaller constant, but which is only first-order accurate. A locally second-order accurate (in the high-resolution sense) fast marching method was proposed in [475]. While this method has a much lower local truncation error than a purely first-order accurate method, it is still globally first-order accurate except for special cases. We might also use the fast sweeping method from [60], [531] which has $O(N)$ complexity (rigorously proven in [606]) and which is also only first-order accurate. Finally, we note that recently Tsai [529] developed an alternative elegant and interesting higher order approach for computing distance quite accurately.

9. Smooth extension of a quantity, e.g., v_n on Γ to a neighborhood of Γ . Let the quantity be $p(\mathbf{x}, t)$. Solve to steady state ($\tau \rightarrow \infty$)

$$\frac{\partial q}{\partial \tau} + \text{sgn}(\varphi) \left(\frac{\nabla \varphi}{|\nabla \varphi|} \cdot \nabla q \right) = 0 \\ q(\mathbf{x}, 0) = p(\mathbf{x}, t).$$

Again, we need only solve this for $\tau = O(\epsilon)$ in order to extend p to be constant in the direction normal to the interface in a tube of width ϵ . This was first suggested and implemented in [108], analyzed carefully in [607], and further discussed and implemented in both [192], and [421]. A computationally efficient algorithm based on heap sort technology and fast marching methods was devised in [4]. There are many reasons to extend a quantity off of Γ , one of which is to obtain a well-conditioned normal velocity for level contours of φ close to $\varphi = 0$ [108].

10. The basic level set method concerns a function $\varphi(\mathbf{x}, t)$ which is defined throughout space. Clearly this is wasteful if one only cares about information near the zero level set. The local level set method defines φ only near the zero level set. We may solve (1.2) in a neighborhood of Γ of width $m\Delta x$, where m is typically 5 or 6. Points outside of this neighborhood need not be updated by this motion. This algorithm works in “ φ ” space—so not too much intricate computer science is used. For details see [421]. Thus this local method works easily in the presence of topological changes and for multiphase flow. An earlier local level set approach called “narrow banding” was devised in [2].

11. High codimensional motion. The level set method was originally developed for curves in R^2 and surfaces in R^3 . Problems involving the motion of curves in R^3 are usually done by front tracking. However, the usual problems of merging and pinching off may occur. In [74] a formulation was derived to make use of two level set functions to model a curve in R^3 . In this formulation, a curve is represented by the intersection between the zero level set functions ϕ and ψ , i.e., where $\phi = \psi = 0$. From this, many properties of the curve can be derived, such as the tangent vectors, $\mathbf{T} = \nabla \psi \times \nabla \psi / |\nabla \psi \times \nabla \phi|$, the curvature vectors,

$\kappa \mathbf{N} = \nabla \mathbf{T} \cdot \mathbf{T}$, and even the torsion, $\tau \mathbf{N} = -\nabla \mathbf{B} \cdot \mathbf{T}$, where \mathbf{N} and \mathbf{B} are the normal and binormal respectively.

Motions of the curve can then be studied under the appropriate system of PDE's involving the two level set functions. The velocity can depend on external physics, as well as on the geometry of the curve (as in the standard level set approach). The resulting system of PDEs for ψ and ϕ is

$$\begin{aligned}\phi_t &= -\mathbf{v} \cdot \nabla \phi \\ \psi_t &= -\mathbf{v} \cdot \nabla \psi.\end{aligned}$$

A simple example involves moving the curve according to its curvature vectors, for which $v = \kappa \mathbf{N}$. See [74] for many examples and more details. This method is called the vector level set method.

An interesting application of high codimensional motion came in [396]. There a level set based approach for ray tracing and for the construction of wavefronts in geometric optics was developed. The goal, as described in [396], is to solve for an evolving curve in (x, y, θ) space (where θ is the angle of the normal to the curve) or on $(x, y, z, \theta, \varphi)$ space (where θ, φ are the polar coordinate angles of the normal to the surface). This leads to 2 and 3 codimensional problems – evolving curves in 3D and surfaces in 5D. We solve for 2 level set functions in 3D or 3 level set functions in 5D to carry this out. This is all tractable because of new local level set methods.

1.3 Numerical Methods

In the important special case where v_N in Eq. 2 is a function only of \mathbf{x}, t , and $\nabla \varphi$ (e.g., $v_N = 1$), Eq. 2 becomes a Hamilton-Jacobi (H-J) equation whose solutions generally develop kinks (jumps in derivatives). We seek the unique viscosity solution. Many good references exist for this important subject; see, e.g. [29],[138]. The appearance of these singularities in the solution means that special, but not terribly complicated, numerical methods have to be used, usually on uniform Cartesian grids. This was first discussed in [401] and numerical schemes developed there were generalized in [256],[402]. The key ideas involve monotonicity, upwind differencing, essentially nonoscillatory (ENO) schemes, and weighted essentially nonoscillatory (WENO) schemes. In this section we present some of the details and motivation.

Typically a level set equation involves a velocity field \mathbf{v} which depends on the geometry of the zero level set and external quantities which can be thought of as depending on \mathbf{x} and t . A prototype is

$$\mathbf{v} = a(\mathbf{x}) \frac{\nabla \varphi}{|\nabla \varphi|} - \mu(\mathbf{x}) \nabla \cdot \left(\frac{\nabla \varphi}{|\nabla \varphi|} \right) \quad (1.4)$$

or a more “linear” version is

$$\mathbf{v} = \mathbf{a}(\mathbf{x}) - \mu(\mathbf{x})\nabla \cdot \left(\frac{\nabla\varphi}{|\nabla\varphi|} \right) \quad (1.5)$$

Here $\mu > 0$ and the equations

$$\varphi_t + a(\mathbf{x})|\nabla\varphi| = \mu(\mathbf{x})|\nabla\varphi|\nabla \cdot \left(\frac{\nabla\varphi}{|\nabla\varphi|} \right) \quad (1.6)$$

$$\varphi_t + a\mathbf{x} \cdot \nabla\varphi = \mu(\mathbf{x})|\nabla\varphi|\nabla \cdot \left(\frac{\nabla\varphi}{|\nabla\varphi|} \right) \quad (1.7)$$

involve convection (in the normal direction with normal velocity $\mathbf{a}(\mathbf{x})$ in (1.6) or $\mathbf{a}(x) \cdot \left(\frac{\nabla\varphi}{|\nabla\varphi|} \right)$ in (1.7)) and curvature dependent motion (defined on as the right hand side with velocity $-\mu(x)\frac{\nabla\varphi}{|\nabla\varphi|}\nabla \cdot \left(\frac{\nabla\varphi}{|\nabla\varphi|} \right)$ or equivalently normal velocity $-\mu(x)\nabla \cdot \frac{\nabla\varphi}{|\nabla\varphi|}$).

For simplicity of exposition only we will compute problems in R^2 , so $\mathbf{x} = (x, y)$. Also, we will describe the methods on a uniform Cartesian grid, $x_j = j\Delta x$, $j = 0, \pm 1, \pm 2 \dots$ $y_k = k\Delta y$, $k = 0, \pm 1, \pm 2 \dots$ and again for simplicity only, we take $\Delta x = \Delta y = \Delta$. Also $t^n = n\Delta t$, $n = 0, 1, 2 \dots$ We shall be using φ_{jk}^n as an approximation to $\varphi(j\Delta, k\Delta, n\Delta t)$. The finite difference scheme we use will involve the following operators:

$$D_{\pm}^x \varphi_{jk} = \pm \frac{(\varphi_{j\pm 1,k} - \varphi_{jk})}{\Delta}, D_{\pm}^y \varphi_{jk} = \pm \frac{(\varphi_{j,k\pm 1} - \varphi_{jk})}{\Delta}.$$

We shall discretize (1.6) and (1.7) in three stages

(1) Approximate the convection term using ideas developed in [401],[402], borrowed from their origins in the numerical solution of conservation laws [231],[480, 481].

(2) Approximate the curvature term by central difference methods.

(3) Use TVD Runge-Kutta schemes, derived in [480] to do the time discretization.

We begin with the approximation of the convection term in (1.7). A simple first order accurate upwind difference approximation to $\mathbf{a}(x) \cdot \nabla\varphi$ is

$$(a_1(x_j, y_k, t^n))^+ D_-^x \varphi_{jk} + (a_1(x_j, y_k, t^n)^- D_+^x \varphi_{jk} \\ + (a_2(x_j, y_k, t^n))^+ D_-^y \varphi_{jk} + a_2(x_j, y_k, t^n)^- D_+^y \varphi_{jk}.$$

Here $a = (a_1, a_2)$ and $x^+ = \max(x, 0)$, $x^- = \min(x, 0)$. The forward and backwards differences are chosen so as to follow the direction of propagation of the characteristics.

We notice that the linear interpolant through the data points $(x_{j-1}, \varphi_{j-1,k})$, (x_j, φ_{jk}) which is

$$I_{j-\frac{1}{2},k}(x) = \varphi_{j,k} + (x - x_j)D_- \varphi_{jk}$$

has the property

$$\frac{d}{dx} I_{j-\frac{1}{2}}^1(x) \Big|_{x=x_j} = D_- \varphi_{jk}.$$

To get higher order, say m^{th} order, space accuracy, we must use a polynomial of degree m which interpolates $(x_{j-1}, \varphi_{j-1,k}), (x_j, \varphi_{jk})$ and $m - 1$ additional data points. These should be of the form

$$(x_{j-\nu}, \varphi_{j-\nu,k}), (x_{j-\nu+1}, \varphi_{j-\nu+1,k}) \dots (x_{j-\nu+m}, \varphi_{j-\nu+m,k})$$

with $1 \leq \nu \leq m$. The question is: which polynomial to choose? This question was originally answered in [231]. We choose the smoothest polynomial i.e. the one with the smaller 2nd, then 3rd then... m^{th} divided differences. This is done inductively. For example,

$$I_{j-\frac{1}{2}jk}^2(x) = I_{j-\frac{1}{2}jk}^1(x) + \frac{(x - x_j)(x - x_{j-1})}{2} m(D_-^2 \varphi_{jk}, D_- D_+ \varphi_{jk})$$

where

$$\begin{aligned} m(x, y) &= x \text{ if } |x| \leq |y| \\ &= y \text{ otherwise.} \end{aligned}$$

This general procedure is called ENO (essentially nonoscillatory) interpolation. For more details see [231].

Thus the m^{th} order ENO approximation to $a(x) \cdot \nabla \varphi$ is:

$$\begin{aligned} &(a_1(x_j, y_k, t^n))^+ \left(\frac{d}{dx} I_{j-\frac{1}{2},k}^m(x) \right)_{x=x_j} \\ &+ (a_1(x_j, y_k, t^n))^- \left(\frac{d}{dx} I_{j-\frac{1}{2},k}^m(x) \right)_{x=x_j} \\ &+ (a_2(x_j, y_k, t^n))^+ \left(\frac{d}{dy} I_{j,k-\frac{1}{2}}^m(y) \right)_{y=y_k} \\ &+ (a_2(x_j, y_k, t^n))^- \left(\frac{d}{dy} I_{j,k+\frac{1}{2}}^m(y) \right)_{y=y_k}. \end{aligned}$$

We typically use $m = 3$ in real applications.

For general nonlinear Hamilton-Jacobi equations of the form

$$\varphi_t + H(x, y, t, \varphi, \varphi_x, \varphi_y) = 0 \quad (1.8)$$

where H is the Hamiltonian, we can again use ENO interpolation to approximate H . We began by constructing a numerical Hamiltonian having the following properties. (For simplicity of exposition, we ignore the $(x_j, y_k, \varphi_{jk}^n)$ dependence):

$$\hat{H}(D_+^x \varphi_{jk}, D_-^x \varphi_{jk}, D_+^y \varphi_{jk}, D_-^y \varphi_{jk}) = \hat{H}(u^+, u^-; v^+, v^-)$$

\hat{H} is a Lipschitz continuous function which is

- (1) Consistant: $\hat{H}(u, u; v, v) = H(u, v)$
- (2) Monotone: $\hat{H}(\downarrow, \uparrow; \downarrow, \uparrow)$, i.e. \hat{H} is nonincreasing in its first and third arguments and nondecreasing in its second and fourth arguments.

There are four monotone numerical Hamiltonians of interest:

$$\hat{H}^G(u^+, u^-; v^+, v^-) = \text{ext}_{u \in I(u^-, u^+)} \text{ext}_{v \in I(v^-, v^+)} H(u, v) \quad (1.9)$$

where $I(a, b) = [\min(a, b), \max(a, b)]$ and

$$\text{ext}_u I(a, b) = \begin{cases} \min_{a \leq u \leq b} & \text{if } a \leq b \\ \max_{a \geq u \geq b} & \text{if } a > b \end{cases}.$$

See [28],[402] for a derivation and motivation. This is the canonical monotone upwind scheme. As an example, if

$$H(u, v) = \sqrt{u^2 + v^2}$$

then

$$\hat{H}^G(u^+, u^-, v^+, v^-) = \sqrt{\max((u^+)^-)^2, ((u^-)^+)^2 + \max((v^+)^-)^2, ((v^-)^+)^2},$$

see [445].

Unfortunately, the formula in (1.9) often becomes complicated near sonic points, i.e. those for which $H_u = 0$ or $H_v = 0$.

A simpler monotone numerical Hamiltonian is

$$\begin{aligned} \hat{H}^{LF}(u^+ u^-; v^+, v^-) &= H\left(\frac{u^+ + u^-}{2}; \frac{v^+ + v^-}{2}\right) \\ &- \alpha \frac{(u^+ - u^-)}{2} - \beta \frac{(v^+ - v^-)}{2} \end{aligned} \quad (1.10)$$

where α, β are positive constants chosen so that

$$\begin{array}{ll} \alpha > \max_{\substack{u \in [A, B] \\ v \in [C, D]}} |H_u(u, v)|, & \beta > \max_{\substack{u \in [A, B] \\ v \in [C, D]}} |H_v(u, v)| \end{array}$$

where $[A, B]$ is the value range for u^\pm and $[C, D]$ is the value range for v^\pm .

A third candidate is:

$$\begin{aligned} \hat{H}^{LLF}(u^+, u^-; v^+, v^-) &= H\left(\frac{u^+ + u^-}{2}, \frac{v^+ + v^-}{2}\right) \\ &- \alpha^+(u^+, u^-) \frac{(u^+ - u^-)}{2} - \beta^+(v^+, v^-) \frac{(v^+ - v^-)}{2} \end{aligned} \quad (1.11)$$

where

$$\begin{array}{ll} \alpha(u^+, u^-) = \max_{\substack{u \in I(u^-, u^+) \\ v \in [C, D]}} |H_u(u, v)|, & \beta(v^+, v^-) = \max_{\substack{v \in I(v^-, v^+) \\ u \in [A, B]}} |H_v(u, v)| \end{array}.$$

The fourth and final candidate is

$$\hat{H}^{RF}(u^+, u^-, v^+, v^-) = \begin{cases} H(u^*, v^*) & \text{if } H_1(u, v) \text{ and } H_2(u, v) \\ & \text{do not change sign in} \\ & u \in I(u^-, u^+) \text{ and} \\ & v \in I(v^-, v^+); \\ \hat{H}^{RF}(u^+, u^-, v^+, v^-) - \alpha(u^+, u^-) \frac{u^+ - u^-}{2} & \text{else if } H_2(u, v) \text{ does not} \\ & \text{change sign in } u \in [A, B] \\ & \text{and } v \in I(v^-, v^+); \\ H\left(u^*, \frac{v^+ + v^-}{2}\right) - \beta(v^+, v^-) \frac{v^+ - v^-}{2} & \text{else if } H_1(u, v) \text{ does not} \\ & \text{change sign in } v \in [C, D] \\ & \text{and } u \in I(u^-, u^+); \\ \hat{H}^{LLF}(u^+, u^-, v^+, v^-) & \text{otherwise,} \end{cases} \quad (1.12)$$

where u^*, v^* are defined by upwinding

$$u^* = \begin{cases} u^+ & \text{if } H_1(u, v) \leq 0, \\ u^- & \text{if } H_1(u, v) \geq 0; \end{cases} \quad v^* = \begin{cases} v^+ & \text{if } H_2(u, v) \leq 0, \\ v^- & \text{if } H_2(u, v) \geq 0. \end{cases}$$

The four Hamiltonians may be ordered monotonically increasing in terms of increasing numerical diffusion and increasing programming simplicity as

$$\hat{H}^G < \hat{H}^{RF} < \hat{H}^{LLF} < \hat{H}^{LF}$$

i.e. the left side is generally less diffusive and more complicated to program than the right side of each inequality.

Now to obtain an m^{th} order ENO approximation to $H(x, y, t, \varphi, \varphi_x, \varphi_y)$ we merely use

$$\hat{H}(x_j, y_k, t^n, \varphi_{jk}, \left(\frac{d}{dx} I_{j+\frac{1}{2}, k}^m(x) \right)_{x=x_j} \left(\frac{d}{dx} I_{j-\frac{1}{2}, k}^m(x) \right)_{x=x_j}; \quad (1.13)$$

$$\left(\frac{d}{dy} I_{j, k+\frac{1}{2}}^m(y) \right)_{y=y_k}, \left(\frac{d}{dy} I_{j, k-\frac{1}{2}}^m(y) \right)_{y=y_k}).$$

In [318] and then later [257, 256] the idea was put forward of taking a weighted combination of polynomials to approximate φ_x, φ_y . The formalism is still as in equation 1.13, but using different polynomials I^m . The weights are chosen so as to default to ENO in the presence of discontinuities, but to obtain optimal accuracy in smooth regions. See [256] for the details. The method seems robust and is desirable for problems which are of one scale and where high order accuracy is important.

We now know how to approximate the convective terms in (1.6) and (1.7). The right hand side diffusive term is handled with central differencing of the following

sort

$$\begin{aligned} |\nabla \varphi| &\approx \sqrt{\frac{1}{2}(D_+^x \varphi_{jk})^2 + (D_-^x \varphi_{jk})^2 + (D_+^y \varphi_{ij})^2 + (D_-^y \varphi_{jk})^2} \quad (1.14) \\ \frac{\partial}{\partial x} \frac{\varphi_x}{|\nabla \varphi|} &\approx D_-^x \left(\frac{D_+^x \varphi_{jk}}{\sqrt{(D_+^x \varphi_{jk})^2 + (D_-^y \varphi_{jk})^2 + \epsilon}} \right) \end{aligned}$$

where

$$(D_-^y \varphi_{jk})^2 = \frac{1}{4} ((D_+^y \varphi_{jk})^2 + (D_+^y \varphi_{j+1,k})^2 + (D_-^y \varphi_{jk})^2 + (D_-^y (\varphi_{j+1,k}))^2)$$

and $\epsilon > 0$ is a small parameter, used to avoid dividing by zero, $\epsilon \approx 10^{-10}$ in our calculations.

An analogous expression is used to approximate $\frac{\partial}{\partial y} \frac{\varphi_y}{|\nabla \varphi|}$.

This discretization results in a second order accurate approximation to the diffusion term, which is adequate for all reasonable calculations.

Finally, we use a third order total variation diminishing (TVD) Runge-Kutta scheme, devised by Shu-Osher in [480] to perform the time integration.

If we have a semidiscrete approximation of the form

$$\frac{\partial \varphi_{jk}}{\partial t} = L_{jk},$$

then the 1st, 2nd and 3rd order TVD Runge-Kutta methods are

$$\begin{aligned} \varphi_{jk}^{n+1} &= \varphi_{jk}^n + \Delta t L_{jk}^n \quad (1\text{st order}) \\ \begin{cases} \varphi_{jk}^{n+\frac{1}{2}} &= \varphi_{jk}^n + \Delta t L_{jk}^n \\ \varphi_{jk}^{n+1} &= \frac{\varphi_{jk}^{n+\frac{1}{2}} + \varphi_{jk}^n}{2} + \frac{\Delta t}{2} L_{jk}^{n+\frac{1}{2}} \end{cases} & \quad (2\text{nd order}) \\ \begin{cases} \varphi_{jk}^{n+\frac{1}{3}} &= \varphi_{jk}^n + \Delta t L_{jk}^n \\ \varphi_{jk}^{n+\frac{2}{3}} &= \frac{3}{4} \varphi_{jk}^n + \frac{1}{4} \varphi_{jk}^{n+\frac{1}{3}} + \frac{1}{4} \Delta t L_{jk}^{n+\frac{1}{3}} \\ \varphi_{jk}^{n+1} &= \frac{1}{3} \varphi_{jk}^n + \frac{2}{3} \varphi_{jk}^{n+\frac{2}{3}} + \frac{2}{3} \Delta t L_{jk}^{n+\frac{2}{3}} \end{cases} & \quad (3\text{rd order}) \end{aligned} \quad (1.15)$$

The time step restriction is of the form

$$\frac{\Delta t}{\Delta} \leq c_1 + c_2 \Delta$$

with c_2 proportional to $|\mu|$ and c_1 proportional to $\max_{u \in [A, B], v \in [C, D]} (|H_u|, |H_v|)$

1.4 Imaging Science Applications

There are far too many applications of geometric level set methods in imaging science to discuss here. I will confine myself to a few classical and new applications which I find to be of special interest.

1.4.1 Dynamic Visibility

The problem is easily stated: Given a collection of closed surfaces representing objects in space, determine quickly the regions (in space or on the surfaces) that are visible to an observer. This question is crucial to applications in fields as diverse as rendering, visualization, etching, and the solution of inverse problems. In a 3D virtual-reality environment, knowing the visible region speeds up the rendering by enabling us to skip costly computations on occluded regions.

As the theme of this book emphasizes, representing a family of surfaces implicitly as the zero level set of a *single* function $\varphi(\mathbf{x})$, $\mathbf{x} = (x, y, z)$, has several advantages, particularly when things are moving. Topological changes are easily handled (without “emotional involvement”), and geometric quantities, such as normals and curvatures, are also easily computed dynamically. Most published work in computer graphics and computer vision uses explicit surfaces, usually constructed with triangles, but this is changing. The upcoming SIGGRAPH conference (in San Antonio, July 2002) will have many LSM-related papers and a full-day course on LSM and PDE-based methods in graphics. For a recent detailed report on the visibility problem with explicit surfaces, see [169].

In our approach to the dynamic visibility problem, as described in [530], we begin by laying down a simple Cartesian grid—this is step one in almost all level set applications. Next, we compute the signed distance function $\varphi(\mathbf{x})$ to the occluding region Ω (which generally has many disjoint pieces). Here we can use the optimally fast algorithm of Tsitsiklis [532, 533]. The function φ approximately satisfies the eikonal equation:

$$\sqrt{\varphi_x^2 + \varphi_y^2} = |\nabla \varphi| = 1 \quad (1.16)$$

with

$$\begin{aligned} \varphi(\mathbf{x}) &< 0 \text{ in } \Omega, \\ \varphi(\mathbf{x}) &> 0 \text{ in } \Omega^c, \end{aligned}$$

and

$$\varphi(\mathbf{x}) = 0 \text{ on } \partial\Omega. \quad (1.17)$$

Then, for a given vantage point \mathbf{x}_0 , we use a simple, easily parallelizable, multiresolution algorithm of optimal complexity to compute the visibility function $\psi_{\mathbf{x}_0}(\mathbf{x})$:

$$\psi_{\mathbf{x}_0}(\mathbf{x}) \geq 0 \Leftrightarrow \mathbf{x} \text{ is visible to } \mathbf{x}_0. \quad (1.18)$$

Next, we allow \mathbf{x} to move with velocity $d\mathbf{x}_0/dt$. Along the way, we obtain fairly elegant geometric-based formulae for the motion of the horizon. This is defined to be the set of visible points \mathbf{x} lying in $\partial\Omega$ for which $\mathbf{x} - \mathbf{x}_0$ is orthogonal to $\partial\Omega$, i.e., for which

$$\psi_{\mathbf{x}_0}(\mathbf{x}) = 0 = \varphi(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_0) \cdot \nabla \varphi(\mathbf{x}). \quad (1.19)$$

We do the same for the motion of points on the cast horizon – that is, points that lie both on the extension of the ray connecting a horizon point \mathbf{x}_0 and to \mathbf{x} and on an occluder.

$\psi_{\mathbf{x}_0(t)}(\mathbf{x}, t)$ can be found quickly at each discrete time step. As $\mathbf{x}_0(t)$ flies through a region, we can compute the invisible region at time t by computing the intersection of the sets

$$S_\tau = \{\mathbf{x} | \psi_{x_0(\tau)}(\mathbf{x}, \tau) < 0\}. \quad (1.20)$$

1.4.2 Interpolation from Unorganized Points via the Level Set Method

Hong-Kai Zhao and I have written a review article on this subject in chapter 16 of this book. See that chapter and [610],[608] for details. Briefly, one finds a level set function whose zero level set passes through a collection of unorganized data points, curves and/or surface patches. Among the advantages of this method is the following: The interpolating surfaces obtained this way can easily be denoised. I shall discuss this in section 4.4 below.

1.4.3 PDE's on Implicit Surfaces

M. Bertalmio, L.-T. Cheng, G. Sapiro and myself have also written a review article on this subject Chapter 18. See that chapter and [38],[40] for details. We believe that this new technique will enable a user to solve PDE's on very complicated surfaces without intricate gridding. One interesting related effort involves moving curves on fixed surfaces in [115]. There we fix one of the two level set functions ψ , defined in the discussion of high codimension motion in Section 2 of this chapter, i.e. we just set $\psi_t = 0$ and choose \mathbf{v} as desired. See [115] for results and details.

1.4.4 The Level Set Method Links Active Contours, Mumford-Shah Segmentation and Total Variation Restoration

This is based on work done jointly with Luminita Vese in [552]. There we proposed new models for active contours and image segmentation based on the Rudin-Osher-Fatemi [449] total variation (TV) minimization, done using a level set framework. We basically do TV restoration on functionals of level set functions. The new TV-based segmentation model has an additional scaling feature given by the magnitude of the jump. We are now minimizing a convex functional over a nonconvex set. The resulting method is free of some of the restrictive properties of the Mumford-Shah model. As in the Chan-Vese active contour segmentation model (see e.g. L. Vese's Chapter 7 in this book and the references therein) we see that interior contours are automatically detected as well as contours with or without steep gradients.

We begin by introducing our notation. Let $\Omega \subset \mathbb{R}^N$ be an open and bounded domain, and $u_0 : \Omega \rightarrow \mathbb{R}$ be a given observed image. We consider $N = 1$ for signals, $N = 2$ for plane images, and $N = 3$ for volumetric images. We also use $\mathcal{H}^{N-1}(\Gamma)$ to denote the $(N-1)$ -dimensional Hausdorff measure of a hypersurface $\Gamma \subset \Omega$. \mathcal{H}^0 gives the counting measure, \mathcal{H}^1 the length or the perimeter, and \mathcal{H}^2 the area.

The Mumford and Shah model [388] has been proposed for image segmentation and partition. By this model, one tries to find piecewise-smooth optimal approximations u of the initial image u_0 . The model is

$$\inf_{u,\Gamma} F^{MS}(u, \Gamma) = \int_{\Omega} (u - u_0)^2 dx + \nu \int_{\Omega \setminus \Gamma} |\nabla u|^2 dx + \mu \mathcal{H}^{N-1}(\Gamma), \quad (1.21)$$

where μ, ν are positive (tuning) parameters. The first term in (1.1) insures that the segmented-reconstructed image u is an approximation of the observation u_0 ; the second term insures that u is smooth outside the set Γ ; finally, the third term asks that the set Γ be smooth enough. This set will approximate the edges or contours of u .

Another important model proposed for image reconstruction is the total variation minimization of Rudin-Osher-Fatemi [449],[448]:

$$\inf_u G^{TV}(u) = \int_{\Omega} (u - u_0)^2 dx + \mu \int_{\Omega} |\nabla u|. \quad (1.22)$$

By this model, the initial noisy image u_0 is well denoised, while preserving edges.

If we minimize the two functionals above restricted to the subset $\{u = H(\phi) | \phi : \Omega \rightarrow \mathbb{R} \text{ is Lipschitz}\}$, i.e. of characteristic functions, then the two functionals are the same:

$$\begin{aligned} F^{MS}(u = H(\phi), \Gamma = \{\phi = 0\}) &= G^{TV}(u = H(\phi)) \\ &= \int_{\Omega} (u_0 - 1)^2 H(\phi) dx + \int_{\Omega} (u_0 - 0)^2 (1 - H(\phi)) dx + \mu \int_{\Omega} |\nabla H(\phi)| . \end{aligned}$$

The minimization problem:

$$\inf_{\phi} E(\phi) = \int_{\Omega} (u_0 - 1)^2 H(\phi) dx + \int_{\Omega} (u_0 - 0)^2 (1 - H(\phi)) dx + \mu \int_{\Omega} |\nabla H(\phi)|$$

has appeared in Osher-Kang [399] for denoising curves representing letters from fax transmissions, but this is also a particular case of the Chan-Vese model for active contours without edges based level sets and Mumford-Shah segmentation form [100],[102]. It also acts as a general method for denoising surfaces where we take $u_0 = H(\varphi_0)$ and the initial surface to denoise is the zero level set of φ_0 . See also [573]. If we restrict the energy (1.3) to the more general piecewise constant case

$$\{u(x) = c^+ H(\phi(x)) + c^- (1 - H(\phi(x))), c^+, c^- \in \mathbb{R}, \phi : \Omega \rightarrow \mathbb{R} \text{ Lipschitz}\},$$

we can write the above energy function of c^+, c^-, ϕ as:

$$\begin{aligned} G(c^+, c^-, \phi) &= \int_{\Omega} (u_0 - c^+)^2 H(\phi) dx + \int_{\Omega} (u_0 - c^-)^2 (1 - H(\phi)) dx \\ &\quad + \mu |c^+ - c^-| \int_{\Omega} |\nabla H(\phi(x))|. \end{aligned} \quad (1.23)$$

We consider now $C^1(\mathbb{R})$ approximations and regularizations H_ε and δ_ε of the Heaviside function and the Dirac Delta function δ_0 , as $\varepsilon \rightarrow 0$, such that $H'_\varepsilon = \delta_\varepsilon$.

Minimizing the above energy, and embedding the gradient descent into a dynamic scheme, we obtain:

$$\begin{aligned} c^+(t) &= \frac{\int_{\Omega} u_0(x) H_\varepsilon(\phi(t, x)) dx}{\int_{\Omega} H_\varepsilon(\phi(t, x)) dx} \\ &\quad - \left(\frac{c^+(t) - c^-(t)}{|c^+(t) - c^-(t)|} \right) \left(\frac{\mu \int_{\Omega} \delta_\varepsilon(\phi(t, x)) |\nabla \phi(t, x)| dx}{2 \int_{\Omega} H_\varepsilon(\phi(t, x)) dx} \right), \\ c^-(t) &= \frac{\int_{\Omega} (x)(1 - H_\varepsilon(\phi(t, x))) dx}{\int_{\Omega} (1 - H_\varepsilon(\phi(t, x))) dx} \\ &\quad - \left(\frac{c^-(t) - c^+(t)}{|c^-(t) - c^+(t)|} \right) \left(\frac{\mu \int_{\Omega} \delta_\varepsilon(\phi(t, x)) |\nabla \phi(t, x)| dx}{2 \int_{\Omega} (1 - H_\varepsilon(\phi(t, x))) dx} \right), \\ \frac{\partial \phi}{\partial t} &= \delta_\varepsilon(\phi) \left[\mu |c^+ - c^-| \operatorname{div} \left(\frac{\nabla \phi}{|\nabla \phi|} \right) - (u_0 - c^+)^2 + (u_0 - c^-)^2 \right]. \end{aligned}$$

If we compare the equations from this TV-based model and the Chan-Vese model from [100],[102] now the values c^+ and c^- are no longer the averages of the image u_0 on the corresponding regions. In addition, there is an extra factor $|c^+ - c^-|$ in the regularization term, multiplied by μ . As we see in the numerical results of [552], this extra factor, the jump has the following role: if we apply both models to the same image and with the same regularizing parameter μ , the TV-based model has a stronger constraint on the total length, therefore less noise will be kept by the new TV-based model, comparing with the Chan-Vese model based on Mumford-Shah.

There are many extensions possible. In particular $u(x)$ can be taken to be of the form:

$$u(x) = u^+(x)H(\varphi(x)) + u^-(x)(I - H(\varphi)(x)) \text{ for } u^+, u^- \text{ both } C^1$$

functions or defined in their domains of definition. Typically, a space of polynomials of a fixed degree might be appropriate.

for numerical results and more details see [552].

1.4.5 Modeling Textures with Total Variation Minimization and Oscillating Patterns

In many problems of image analysis, we have an observed image f , representing a real scene. The image f may contain noise (some random pattern of zero mean for instance) and/or texture (some repeated pattern of small scale details). The image

processing task is to extract the most meaningful information from f . This is usually formulated as an inverse problem: given f , find another image u , “close” to f , such that u is a cartoon or simplification of f . In general, u is an image formed by homogeneous regions and with sharp boundaries. Most models assume the following relation between f and u : $f = u + v$, where v is noise or small scale repeated detail (texture), and extract only the u component from f . Usually, the component v is not kept, assuming that this models the noise. A successful algorithm of this type is due to Rudin-Osher-Fatemi [449],[448], see equation (1.22).

In some cases, the v component is important, especially if it represents texture. Texture can be defined as a repeated pattern of small scale details. The noise is also a pattern of small scale details, but of random, uncorrelated values. Both types of patterns (additive noise or texture) can be modeled by oscillatory functions taking both positive and negative values, and of zero mean [368].

With Luminita Vese in [553], we showed how we can extract from f both components u and v , in a simple total variation minimization framework of Rudin-Osher-Fatemi [449]. The obtained decomposition can then be useful for segmentation of textured images and texture discrimination, among other possible applications. The textured component v is completely represented using only two functions (g_1, g_2). This is much simpler and much more efficient than the classical techniques for textures, such as wavelet decomposition, or more specifically Gabor transform. These two techniques use a large number of channels to represent a textured image.

Let $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ be a given image (we assume that the image initially defined on a rectangle in \mathbb{R}^2 , has been extended by reflection to the entire space). We assume that $f \in L^2(\mathbb{R}^2)$. In real applications, the observed image f is just a noisy version of a true image u , or it is a textured image, and u would be a simple sketchy approximation or a cartoon image of f . In the presence of additive noise, the relation between u and f can be expressed by the linear model, introducing another function v , such that

$$f(x, y) = u(x, y) + v(x, y).$$

In the Rudin-Osher-Fatemi restoration model [449], v represents noise or small scale repeated details, while u is an image formed by homogeneous regions, and with sharp edges. Given f , both u and v are unknown (if v is noise, we may know some statistics of v , such that it is of zero mean and given variance). In [449], the problem of reconstructing u from f is posed as a minimization problem in the space of functions of bounded variation $BV(\mathbb{R}^2)$, this space allowing for edges or discontinuities along curves. Their model, very efficient for denoising images while keeping sharp edges, is

$$\inf_{u \in L^2} F(u) = \int |\nabla u| + \lambda \int |f - u|^2 dx dy. \quad (1.24)$$

Formally minimizing $F(u)$ yields the associated Euler-Lagrange equation

$$u = f + \frac{1}{2\lambda} \operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right).$$

In practice, to avoid division by zero, the curvature term $\operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right)$ is approximated by $\operatorname{div} \left(\frac{\nabla u}{\sqrt{\varepsilon^2 + |\nabla u|^2}} \right)$, but this is not computed explicitly. Only the component u is kept in the ROF model.

Note that the curvature of the level contours of the final processed image is proportional to the noise in this model.

Note that the above function v in the ROF model can be formally written as: $v = \operatorname{div} g$, where $g = (g_1, g_2)$ and $g_1 = -\frac{1}{2\lambda} \frac{u_x}{|\nabla u|}$, $g_2 = -\frac{1}{2\lambda} \frac{u_y}{|\nabla u|}$. We have that $g_1(x, y) + g_2^2(x, y) = \frac{1}{2\lambda}$ for all (x, y) , so that $\|\sqrt{g_1^2 + g_2^2}\|_{L^\infty} = \frac{1}{2\lambda}$ (later we will use the notation $|g| = \sqrt{g_1^2 + g_2^2}$).

In [368], Yves Meyer proves that the ROF model will remove the texture, if λ is small enough. In order to extract both the u component in BV and the v component as an oscillating function (texture or noise) from f , Meyer [368] proposes the use of a space of functions, which is in some sense the dual of the BV space. He introduces the following definition.

Definition 1. Let G denote the Banach space consisting of all generalized functions $f(x)$ which can be written as

$$f(x) = \partial_x g_1(x, y) + \partial_y g_2(x, y), \quad g_1, g_2 \in L^\infty(\mathbb{R}^2), \quad (1.25)$$

induced by the norm $\|f\|_*$ defined as the lower bound of all L^∞ norms of the functions $|g|$ where $g = (g_1, g_2)$, $|g(x, y)| = \sqrt{g_1(x, y)^2 + g_2(x, y)^2}$ and where the infimum is computed over all decompositions (1.4) of f .

Meyer proposes a new image restoration model

$$\inf_u E(u) = \{|\nabla \varphi| + \lambda \|v\|_*, \quad f = u + v\}. \quad (1.26)$$

A procedure for computing the minimum of this functional based on the Euler-Lagrange equation is not easily obtained. Instead in [553] we proposed a variant of this model.

We are motivated by the following approximation to the L^∞ norm of $|g| = \sqrt{g_1^2 + g_2^2}$, for $g_1, g_2 \in L^\infty(\mathbb{R}^2)$:

$$\left\| \sqrt{g_1^2 + g_2^2} \right\|_{L^\infty} = \lim_{p \rightarrow \infty} \left\| \sqrt{g_1^2 + g_2^2} \right\|_{L^p}.$$

Then, we propose the following minimization problem, inspired by (1.26):

$$\begin{aligned} \inf_{u, g_1, g_2} \{G_p(u, g_1, g_2)\} &= \int |\nabla u| + \lambda \int |f - u - \partial_x g_1 - \partial_y g_2|^2 dx dy \\ &\quad + \mu \left[\int \left(\sqrt{g_1^2 + g_2^2} \right)^p dx dy \right]^{\frac{1}{p}}, \end{aligned} \quad (1.27)$$

where $\lambda, \mu > 0$ are tuning parameters, and $p \rightarrow \infty$.

The first term insures that $u \in BV(\mathbb{R}^2)$, the second term insures that $f \approx u + \operatorname{div} g$. Clearly, if $\lambda \rightarrow \infty$ and $p \rightarrow \infty$, this model is formally an approximation of the model (1.26) originally proposed by Y. Meyer.

Formally minimizing the above energy with respect to u_1, g_1, g_2 , yields the following Euler-Lagrange equations:

$$u = f - \partial_x g_1 - \partial_y g_2 + \frac{1}{2\lambda} \operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right), \quad (1.28)$$

$$\begin{aligned} \mu \left(\|\sqrt{g_1^2 + g_2^2}\|_p \right)^{1-p} \left(\sqrt{g_1^2 + g_2^2} \right)^{p-2} g_1 \\ = 2\lambda \left[\frac{\partial}{\partial x} (u - f) + \partial_{xx}^2 g_1 + \partial_{xy}^2 g_2 \right], \end{aligned} \quad (1.29)$$

$$\begin{aligned} \mu \left(\|\sqrt{g_1^2 + g_2^2}\|_p \right)^{1-p} g_2 \\ = 2\lambda \left[\frac{\partial}{\partial y} (u - f) + \frac{\partial^2}{\partial_x \partial_y} g_1 + \frac{\partial^2}{\partial_y^2} g_2 \right]. \end{aligned} \quad (1.30)$$

In our numerical calculations, we have tested the model for different values of p , with $1 \leq p \leq 10$. The obtained results are very similar. The case $p = 1$ yields faster calculations per iteration, so we give here the form of the Euler-Lagrange equations in this case ($p = 1$):

$$u = f - \partial_x g_1 - \partial_y g_2 + \frac{1}{2\lambda} \operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right), \quad (1.31)$$

$$\mu \frac{g_1}{\sqrt{g_1^2 + g_2^2}} = 2\lambda \left[\frac{\partial}{\partial x} (u - f) + \partial_{xx}^2 g_1 + \partial_{xy}^2 g_2 \right], \quad (1.32)$$

$$\mu \frac{g_2}{\sqrt{g_1^2 + g_2^2}} = 2\lambda \left[\frac{\partial}{\partial y} (u - f) + \partial_{xy}^2 g_1 + \partial_{yy}^2 g_2 \right]. \quad (1.33)$$

If the domain is finite, with exterior normal to the boundary denoted by (n_x, n_y) , the associated boundary conditions are:

$$\begin{aligned} \frac{\nabla u}{|\nabla u|}(n_x, n_y) &= 0, \\ (f - u - \partial_x g_1 - \partial_y g_2)n_x &= 0, \\ (f - u - \partial_x g_1 - \partial_y g_2)n_y &= 0. \end{aligned}$$

We have seen in the numerical results of [553], the proposed minimization model (1.27) allows to extract from a given real textured image f the components u and v , such that u is a sketchy (cartoon) approximation of f , and $v = \operatorname{div} (g_1, g_2)$ represents the texture or the noise. In addition, the minimizer obtained for $g = (g_1, g_2)$ allows us to discriminate two textures, by looking at the functions $|g| = \sqrt{g_1^2 + g_2^2}$, $|g_1|$ or $|g_2|$.

Our model is of the form

$$f = u + v + w$$

where w is the residual after the TV and texture parts are removed. According to equation (1.28) it is once again proportional to the level sets of u .

See [553] for details, and analytic justification and very promising numerical results.

1.5 Conclusion

The level set method and related PDE based techniques have helped create a paradigm shift in imaging science. Older books in image processing had the word “digital” in their title. Now it appears that imaging science has developed an important “continuum” point of view. Of course this shift began before the invention of the level set method but it is fair to say that the method and the associated numerical technology have greatly impacted research in this area.

Acknowledgments

This work was supported in part by NSF DMS 0074735, ONR N00014-97-0027 and NIH P20MH65166.

2

Deformable Models: Classic, Topology-Adaptive and Generalized Formulations

Demetri Terzopoulos

Abstract

“Deformable models” refers to a class of physics-based modeling methods with an extensive track record in computer vision, medical imaging, computer graphics, geometric design, and related areas. Unlike the Eulerian (fluid) formulations associated with level set methods, deformable models are characterized by Lagrangian (solid) formulations, three variants of which are reviewed herein.

2.1 Introduction

This chapter reviews *Deformable Models*, a powerful class of physics-based modeling techniques widely employed in image synthesis (computer graphics), image analysis (computer vision), shape design (computer-aided geometric design) and related fields. It may at first seem odd to find a chapter on deformable models, whose governing equations are based on Lagrangian continuum mechanics formulations, in a volume dedicated to the Eulerian formulations associated with level set methods. Upon reflection, however, it becomes self-evident that the two approaches are complementary in precisely the same sense that Lagrangian solid models complement Eulerian fluid models in continuum mechanics. The substantial literature on deformable models and level set methods is a testament to the fact that both approaches are useful in imaging, vision, and graphics.

By numerically simulating their governing equations of motion, typically expressed as PDEs in a continuous setting or ODEs in a discrete setting, deformable models mimic various generic behaviors of natural nonrigid materials in response to applied forces, such as continuity, smoothness, elasticity, plasticity, etc. For the

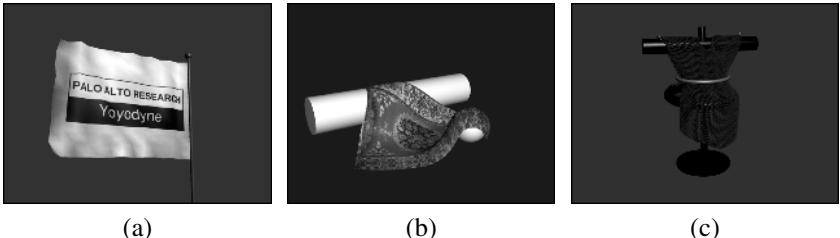


Figure 2.1. Early computer graphics images of physics-based deformable surface model simulations depicting objects with cloth-like behaviors [517, 514]. (a) Flag in wind. (b) Rug falling over geometric obstacles. (c) Draped robe and constraints.

purposes of computer graphics, realistic images and animations of elastic, inelastic, and thermoelastic objects may be synthesized when the applied forces stem from animation controllers and from model interactions within simulated physical environments [514] (see, e.g., Fig. 2.1). In the related domain of computer-aided geometric design, deformable models have inspired a new approach known as “physics-based geometric design”. Here, the parameters of standard geometric primitives become generalized coordinates in Lagrangian formulations that govern their automatic evolution in response to simulated (sculpting) forces, subject to geometric constraints [518].

Among model-based techniques for image analysis, deformable models offer a potent approach that combines geometry, physics, and approximation theory. In applications to computer vision, deformable models may be used to infer image disparity fields, image flow fields, and to infer the shapes and motions of objects from still or video images [520, 513]. In this context, deformable models are subjected to external forces that impose constraints derived from image data. The forces actively shape and move models to achieve maximal consistency with imaged objects of interest and to maintain consistency over time. There has been a tremendous surge of interest in deformable models in the context of medical image analysis [352]. Here, deformable models have proven useful in segmenting, matching, and tracking anatomic structures by exploiting (bottom-up) constraints derived from the image data in conjunction with (top-down) *a priori* knowledge about the location, size, and shape of these structures. With regard to the latter, deformable models support intuitive interaction mechanisms that enable medical scientists and practitioners to bring their expertise to bear on the model-based image interpretation task.

This remainder of this chapter comprises three main sections. Section 2.2 covers the mathematical foundations of “classic” deformable models, including energy-minimizing and dynamic snakes [263], their discretization, numerical simulation and probabilistic interpretation. The parametric geometry of snakes allows controlled, piecewise continuity and both closed and open curves (note that current level set methods have difficulty representing open curves). The section also covers higher dimensional generalizations of snakes, such as *deformable surfaces*. In image segmentation, the introduction of level set methods was motivated by

the success of deformable models and the need for a related technique that makes no prior assumption about the topology of the underlying structure of interest. As other chapters in this volume make evident, Eulerian models of fluids, particularly those implemented via level set methods, seem more appropriate in the face of unknown topology. In Section 2.3, however, I review *topology-adaptive deformable models*, which provide the topological flexibility of level set methods without sacrificing the explicit geometric substrate upon which the classic deformable model formulations are based. In Section 2.4, I review deformable models constructed on more sophisticated geometric substrates, including *dynamic non-uniform rational B-splines* for physics-based geometric design and *deformable superquadrics* for computer vision and computer graphics animation. Thus far, such generalized deformable models, in which arbitrary geometric parameters play the role of Lagrangian generalized coordinates, seem to fall outside the scope of level set methods. Finally, Section 2.5 concludes the chapter.¹

2.2 Classic Deformable Models

This section reviews the mathematics of classic deformable models. First, I consider planar *active contour models*, also known as “snakes”, including energy-minimizing snakes and dynamic snakes. Next, I discuss the discretization and numerical simulation of snakes, as well as their probabilistic interpretation. Finally, I review higher-dimensional generalizations of snakes, in particular, deformable surfaces.

2.2.1 Energy-Minimizing Snakes

Snakes are planar deformable contours that are useful in a variety of image analysis tasks [263]. They are often used to approximate the locations and shapes of object boundaries in images, based on the assumption that boundaries are piecewise continuous or smooth (Fig. 2.2(a)). In its basic form, the mathematical formulation of snakes draws from the theory of optimal approximation involving functionals.

Geometrically, a snake is an explicit, parametric contour embedded in the image plane $(x, y) \in \mathbb{R}^2$. The contour is represented as $\mathbf{v}(s) = (x(s), y(s))^\top$, where x and y are the coordinate functions and $s \in [0, 1]$ is the parametric domain (the symbol $^\top$ denotes transposition). The shape of the contour subject to an image

¹This chapter makes no pretense to being a literature survey. In fact, I shall continue herein to cite only my own published work on deformable models. The interested reader may refer to the more recent of these sources for a broader perspective on the extensive deformable models literature, as well as to this volume’s other chapters and integrated bibliography for the associated literature on level set methods.

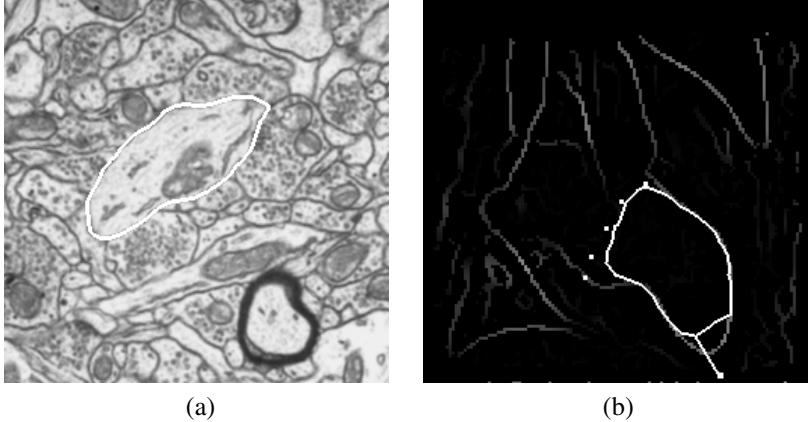


Figure 2.2. Medical image analysis with snakes. (a) Snake (white) segmenting a cell in an EM photomicrograph [78]. The snake is attracted to the dark cell membrane. (b) Snake deforming towards high gradients in a processed cardiac image, influenced by “pin” constraints and an interactive “spring” with which the user pulls the contour towards an edge [351].

$I(x, y)$ is dictated by the functional

$$\mathcal{E}(\mathbf{v}) = \mathcal{S}(\mathbf{v}) + \mathcal{P}(\mathbf{v}), \quad (2.1)$$

which represents the energy of the contour. The final shape of the contour corresponds to the minimum of this energy.

The first term in (2.1),

$$\mathcal{S}(\mathbf{v}) = \frac{1}{2} \int_0^1 w_1(s) \left| \frac{\partial \mathbf{v}}{\partial s} \right|^2 + w_2(s) \left| \frac{\partial^2 \mathbf{v}}{\partial s^2} \right|^2 ds, \quad (2.2)$$

is the internal deformation energy. It characterizes the deformation of a stretchy, flexible contour. Two physical parameter functions, the non-negative functions $w_1(s)$ and $w_2(s)$, dictate the simulated physical characteristics of the contour at any point s on the snake: $w_1(s)$ controls the “tension” of the contour while $w_2(s)$ controls its “rigidity”. For example, increasing the magnitude of $w_1(s)$ tends to eliminate extraneous loops and ripples by reducing the length of the snake. Increasing $w_2(s)$ makes the snake smoother and less flexible. Setting the value of one or both of these functions to zero at a point s permits discontinuities in the contour at s .

The second term in (2.1) couples the snake to the image. Traditionally,

$$\mathcal{P}(\mathbf{v}) = \int_0^1 P(\mathbf{v}(s)) ds, \quad (2.3)$$

where $P(x, y)$ denotes a scalar potential function defined on the image plane. To apply snakes to images, external potentials are designed whose local minima coincide with intensity extrema, edges, and other image features of interest.

For example, the contour will be attracted to intensity edges in an image $I(x, y)$ by choosing a potential $P(x, y) = -c|\nabla[G_\sigma * I(x, y)]|$, where c controls the magnitude of the potential, ∇ is the gradient operator, and $G_\sigma * I$ denotes the image convolved with a (Gaussian) smoothing filter whose characteristic width σ controls the spatial extent of the local minima of P .

In accordance with the calculus of variations, the contour $\mathbf{v}(s)$ which minimizes the energy $\mathcal{E}(\mathbf{v})$ must satisfy the Euler-Lagrange equation

$$-\frac{\partial}{\partial s} \left(w_1 \frac{\partial \mathbf{v}}{\partial s} \right) + \frac{\partial^2}{\partial s^2} \left(w_2 \frac{\partial^2 \mathbf{v}}{\partial s^2} \right) + \nabla P(\mathbf{v}(s, t)) = \mathbf{0}. \quad (2.4)$$

This vector-valued partial differential equation (PDE) expresses the balance of internal and external forces when the contour rests at equilibrium. The first two terms represent the internal stretching and bending forces, respectively, while the third term represents the external forces that couple the snake to the image data. The usual approach to solving (2.4) is through the application of numerical algorithms (see Sec. 2.2.3).

2.2.2 Dynamic Snakes

While it is natural to view energy minimization as a static problem, a potent approach to computing the local minima of a functional such as (2.1) is to construct a dynamical system that is governed by the functional and allow the system to evolve to equilibrium. The system may be constructed by applying the principles of Lagrangian mechanics. This leads to dynamic deformable models that unify the description of shape and motion, making it possible to quantify not just static shape, but also shape evolution through time. Dynamic models are valuable for, e.g., time-varying medical image analysis, since most anatomical structures are deformable and continually undergo nonrigid motion *in vivo*. Moreover, dynamic models exhibit intuitively meaningful physical behaviors, making their evolution amenable to interactive guidance from a user (Fig. 2.2(b)).

A simple example is a dynamic snake which can be represented by introducing a time-varying contour $\mathbf{v}(s, t) = (x(s, t), y(s, t))^\top$ along with a mass density $\mu(s)$ and a damping density $\gamma(s)$. The Lagrange equations of motion for a snake with the internal energy given by (2.2) and external energy given by (2.3) is

$$\mu \frac{\partial^2 \mathbf{v}}{\partial t^2} + \gamma \frac{\partial \mathbf{v}}{\partial t} - \frac{\partial}{\partial s} \left(w_1 \frac{\partial \mathbf{v}}{\partial s} \right) + \frac{\partial^2}{\partial s^2} \left(w_2 \frac{\partial^2 \mathbf{v}}{\partial s^2} \right) = -\nabla P(\mathbf{v}(s, t)). \quad (2.5)$$

The first two terms on the left hand side of this partial differential equation represent inertial and damping forces. As in (2.4), the remaining terms represent the internal stretching and bending forces, while the right hand side represents the external forces. Equilibrium is achieved when the internal and external forces balance and the contour comes to rest (i.e., $\partial \mathbf{v} / \partial t = \partial^2 \mathbf{v} / \partial t^2 = 0$), which yields the equilibrium condition (2.4).

2.2.3 Discretization and Numerical Simulation

In order to compute numerically a minimum energy solution, it is necessary to discretize the energy $\mathcal{E}(\mathbf{v})$. The usual approach is to represent the continuous geometric model \mathbf{v} in terms of linear combinations of local-support or global-support basis functions. Finite elements, finite differences, and geometric splines are local representation methods, whereas Fourier bases are global representation methods. The continuous model $\mathbf{v}(s)$ is represented in discrete form by a vector \mathbf{u} of shape parameters associated with the basis functions. The discrete form of energies such as $\mathcal{E}(\mathbf{v})$ for the snake may be written as

$$E(\mathbf{u}) = \frac{1}{2} \mathbf{u}^\top \mathbf{K} \mathbf{u} + \mathbf{P}(\mathbf{u}) \quad (2.6)$$

where \mathbf{K} is called the *stiffness matrix*, and $\mathbf{P}(\mathbf{u})$ is the discrete version of the external potential. The minimum energy solution results from setting the gradient of (2.6) to 0, which is equivalent to solving the set of algebraic equations

$$\mathbf{K} \mathbf{u} = -\nabla \mathbf{P} = \mathbf{f} \quad (2.7)$$

where \mathbf{f} is the generalized external force vector.

Finite elements and finite differences generate local discretizations of the continuous snake model, hence the stiffness matrix will have a sparse and banded structure. To illustrate the discretization process, suppose we apply the finite difference method to discretize the energy (2.2) on a set of nodes $\mathbf{u}_i = \mathbf{v}(ih)$ for $i = 0, \dots, N-1$ where $h = 1/(N-1)$ and suppose we use the finite differences $\mathbf{v}_s \approx (\mathbf{u}_{i+1} - \mathbf{u}_i)/h$ and $\mathbf{v}_{ss} \approx (\mathbf{u}_{i+1} - 2\mathbf{u}_i + \mathbf{u}_{i-1})/h^2$. For cyclic boundary conditions (i.e., a closed contour), we obtain the following symmetric pentadiagonal matrix (unspecified entries are 0):

$$\mathbf{K} = \begin{bmatrix} a_0 & b_0 & c_0 & & & c_{N-2} & b_{N-1} \\ b_0 & a_1 & b_1 & c_1 & & & c_{N-1} \\ c_0 & b_1 & a_2 & b_2 & c_2 & & \\ & c_1 & b_2 & a_3 & b_3 & c_3 & \\ & & \ddots & \ddots & \ddots & \ddots & \\ & & & c_{N-5} & b_{N-4} & a_{N-3} & b_{N-3} & c_{N-3} \\ c_{N-2} & & & & c_{N-4} & b_{N-3} & a_{N-2} & b_{N-2} \\ b_{N-1} & c_{N-1} & & & & c_{N-3} & b_{N-2} & a_{N-1} \end{bmatrix}, \quad (2.8)$$

where

$$a_i = (w_{1i-1} + w_{1i})/h^2 + (w_{2i-1} + 4w_{2i} + w_{2i+1})/h^4, \quad (2.9)$$

$$b_i = -w_{1i}/h^2 - 2(w_{2i} + w_{2i+1})/h^4, \quad (2.10)$$

$$c_i = w_{2i+1}/h^4, \quad (2.11)$$

assuming that $w_{1i} = w_1(ih)$ and $w_{2i} = w_2(ih)$ are sampled at the same nodes. All indices in these expressions are interpreted modulo N .

The discretized version of the Lagrangian dynamics equation (2.5) may be written as a set of second order ordinary differential equations (ODEs) for \mathbf{u} :

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{D}\dot{\mathbf{u}} + \mathbf{K}\mathbf{u} = \mathbf{f}, \quad (2.12)$$

where \mathbf{M} is the mass matrix and \mathbf{D} is a damping matrix. In a finite difference discretization, the mass and damping matrices are diagonal matrices.

To simulate the snake dynamics, the system of ordinary differential equations (2.12) in the shape parameters \mathbf{u} must be integrated forward through time. The finite element literature offers several suitable explicit and implicit direct integration methods, including the central difference, Houbolt, Newmark, or Wilson methods [33]. We can illustrate the basic idea with a semi-implicit Euler method that takes time steps Δt . We replace the time derivatives of \mathbf{u} with the backward finite differences $\ddot{\mathbf{u}} \approx (\mathbf{u}^{(t+\Delta t)} - 2\mathbf{u}^{(t)} + \mathbf{u}^{(t-\Delta t)})/(\Delta t)^2$, and $\dot{\mathbf{u}} \approx (\mathbf{u}^{(t+\Delta t)} - \mathbf{u}^{(t-\Delta t)})/2\Delta t$, where the superscripts denote the quantity evaluated at the time given in parentheses. This yields the update formula

$$\mathbf{A}\mathbf{u}^{(t+\Delta t)} = \mathbf{b}^{(t)}, \quad (2.13)$$

where $\mathbf{A} = \mathbf{M}/(\Delta t)^2 + \mathbf{D}/2\Delta t + \mathbf{K}$ is a pentadiagonal matrix and $\mathbf{b}^{(t)} = (2\mathbf{M}/(\Delta t)^2)\mathbf{u}^{(t)} - (\mathbf{M}/(\Delta t)^2 - \mathbf{D}/2\Delta t)\mathbf{u}^{(t-1)} + \mathbf{f}^{(t)}$. The pentadiagonal system can be solved efficiently ($O(N)$ complexity) by factorizing \mathbf{A} into lower and upper triangular matrices, then solving the two resulting sparse triangular systems. We compute the unique normalized factorization $\mathbf{A} = \mathbf{LYU}$ where \mathbf{L} is a lower triangular matrix, \mathbf{Y} is a diagonal matrix, and $\mathbf{U} = \mathbf{L}^\top$ is an upper triangular matrix [33]. The solution $\mathbf{u}^{(t+\Delta t)}$ to (2.13) is obtained by first solving $\mathbf{L}\mathbf{s} = \mathbf{b}^{(t)}$ by forward substitution, then $\mathbf{U}\mathbf{u} = \mathbf{Y}^{-1}\mathbf{s}$ by backward substitution. For the linear snakes described above, only a single factorization is necessary, since \mathbf{A} is constant. Note that the factorization and forward/backward substitutions are inherently sequential, recursive operations.

Researchers have investigated alternative approaches to numerically simulating snake models, including dynamic programming and greedy algorithms (see [352] for a survey in the context of medical image analysis).

2.2.4 Probabilistic Interpretation

An alternative view of deformable models emerges from casting the model fitting process in a probabilistic framework. This permits the incorporation of prior model and sensor model characteristics in terms of probability distributions. The probabilistic framework also provides a measure of the uncertainty of the estimated shape parameters after the model is fitted to the image data.

Let \mathbf{u} represent the deformable model shape parameters with a prior probability $p(\mathbf{u})$ on the parameters. Let $p(I|\mathbf{u})$ be the imaging (sensor) model—the probability of producing an image I given a model \mathbf{u} . Bayes' theorem

$$p(\mathbf{u}|I) = \frac{p(I|\mathbf{u})p(\mathbf{u})}{p(I)} \quad (2.14)$$

expresses the posterior probability $p(\mathbf{u}|I)$ of a model given the image, in terms of the imaging model and the prior probabilities of model and image.

It is easy to convert the internal energy measure (2.2) of the deformable model into a prior distribution over expected shapes, with lower energy shapes being the more likely. This is achieved using a Boltzmann (or Gibbs) distribution of the form

$$p(\mathbf{u}) = \frac{1}{Z_s} \exp(-S(\mathbf{u})), \quad (2.15)$$

where $S(\mathbf{u})$ is the discretized version of $\mathcal{S}(\mathbf{v})$ in (2.2) and Z_s is a normalizing constant (called the partition function). This prior model is then combined with a simple sensor model based on linear measurements with Gaussian noise

$$p(I|\mathbf{u}) = \frac{1}{Z_I} \exp(-P(\mathbf{u})), \quad (2.16)$$

where $P(\mathbf{u})$ is a discrete version of the potential $\mathcal{P}(\mathbf{v})$ in (2.3), which is a function of the image $I(x, y)$.

Models may be fitted by finding \mathbf{u} which locally maximize $p(\mathbf{u}|I)$ in (2.14). This is known as the maximum a posteriori solution. With the above construction, it yields the same result as minimizing (2.1), the energy configuration of the deformable model given the image.

The probabilistic framework can be extended by assuming a time-varying prior model, or system model, in conjunction with the sensor model, resulting in a Kalman filter. The system model describes the expected evolution of the shape parameters \mathbf{u} over time. If the equations of motion of the physical snakes model (2.12) are employed as the system model, the result is a sequential estimation algorithm known as “Kalman snakes” [519], which is useful for tracking objects in video.

2.2.5 Higher-Dimensional Generalizations

Snakes are a special case within the general framework of continuous (multidimensional) deformable models in a Lagrangian dynamics setting that is based on deformation energies in the form of (controlled-continuity) generalized splines [512].

In a p -dimensional domain $\mathbf{x} = (x_1, \dots, x_p) \in \mathbb{R}^p$, the natural generalization of the smoothness functional (2.2) defined on a q -dimensional vector of coordinate functions $\mathbf{v}(\mathbf{x}) = [v_1(\mathbf{x}), \dots, v_q(\mathbf{x})]$ is

$$\mathcal{S}_n(\mathbf{v}) = \frac{1}{2} \int_{\mathbb{R}^p} \sum_{m=1}^n \sum_{|j|=m} \frac{m!}{j_1! \dots j_p!} w_j(\mathbf{x}) \left| \frac{\partial^m \mathbf{v}(\mathbf{x})}{\partial x_1^{j_1} \dots \partial x_p^{j_p}} \right|^2 d\mathbf{x}. \quad (2.17)$$

Here, $j = (j_1, \dots, j_p)$ is a multi-index with $|j| = j_1 + \dots + j_p$. Note that this functional offers higher-order smoothness by generalizing (2.2) beyond second-order derivatives, to derivatives of order n . Analogous to equation (2.4), assuming

that the control functions $\mathbf{w}(\mathbf{x})$ are differentiable to order p , the Euler–Lagrange equation is:

$$\sum_{m=1}^n (-1)^m \Delta_w^m \mathbf{v}(\mathbf{x}) + \nabla P(\mathbf{v}(\mathbf{x})) = \mathbf{0}, \quad (2.18)$$

where

$$\Delta_w^m = \sum_{|j|=m} \frac{m!}{j_1! \dots j_p!} \frac{\partial^m}{\partial x_1^{j_1} \dots \partial x_p^{j_p}} \left(w_j(\mathbf{x}) \frac{\partial^m}{\partial x_1^{j_1} \dots \partial x_p^{j_p}} \right) \quad (2.19)$$

is a spatially weighted m th-order iterated Laplacian operator.

Deformable Surfaces:

In the special case $n = 2$, $q = 3$, and $p = 2$, where we define $\mathbf{x} = (x_1, x_2) = (x, y)$ for notational convenience and restrict the 2-dimensional domain to the unit square, (2.17) can be written as

$$\begin{aligned} \mathcal{S}(\mathbf{v}) = & \frac{1}{2} \int_0^1 \int_0^1 w_{10} \left| \frac{\partial \mathbf{v}}{\partial x} \right|^2 + w_{01} \left| \frac{\partial \mathbf{v}}{\partial y} \right|^2 + \\ & w_{20} \left| \frac{\partial^2 \mathbf{v}}{\partial x^2} \right|^2 + 2w_{11} \left| \frac{\partial^2 \mathbf{v}}{\partial x \partial y} \right|^2 + w_{02} \left| \frac{\partial^2 \mathbf{v}}{\partial y^2} \right|^2 dx dy, \end{aligned} \quad (2.20)$$

where the subscripts on \mathbf{v} denote its partial derivatives with respect to x and y . This functional, the natural, two-dimensional generalization of the snake energy (2.2), pertains to the problem of deformable surfaces. The physical parameter functions $w_{10}(x, y)$ and $w_{01}(x, y)$ control the tension of the surface, while $w_{20}(x, y)$, $w_{11}(x, y)$, and $w_{02}(x, y)$ control its “rigidity”.

This *thin plate under tension* functional has seen considerable application, notably to visible-surface reconstruction in computer vision [513].

2.2.6 Connections to Curve Evolution

There is a well-known relationship between classic deformable models and level set methods. The typical curve evolution equations that are computed as level sets correspond to a reduced version of the equations of motion (2.5) that characterize a massless snake $\mu(s) = 0$ with no rigidity $w_2(s) = 0$. This special case results in snakes that, like conventional level set curves, minimize arc length in the metric induced by the image.

The analogous special case in 3D is surfaces that minimize surface area in the metric induced by a volume image. This is equivalent to a massless deformable surface governed by the functional \mathcal{S}_n in (2.20) with $w_{20}(x, y) = w_{11}(x, y) = w_{02}(x, y) = 0$; i.e., the membrane functional.

2.3 Topology-Adaptive Deformable Models

As physics-based models of nonrigid solids, deformable models have had an enormous impact in medical image analysis [352]. The complexity of human anatomy, comprising numerous nonrigid organs with intricate substructures at multiple scales of resolution, means that deformable models must generally be able to deal with non-simple and even non-constant topologies; for example, in serial reconstruction the topology of a segmented cross-section can vary dramatically as the image data are sliced in different ways. Unfortunately, without additional machinery, classic, Lagrangian deformable models cannot alter their prescribed topology.

Level set image segmentation methods were motivated by the need for a related technique that makes no prior assumption about the topology of the underlying object of interest. To this end, level-set methods formulate boundary curve or surface estimation as an Eulerian problem defined over the entire image domain. The dimensionality of the Eulerian problem typically is one greater than the dimensionality of the associated Lagrangian problem for deformable models. The primary feature of this approach is that the higher-dimensional hypersurface remains a simple function, even as the level set changes topology (or ceases to be simply connected). Hence, topological changes are handled naturally.

Topology-adaptive deformable models are an alternative approach to non-fixed topology [354, 353]. They circumvent the increased dimensionality of the level set methods while retaining the strengths of standard parametric deformable models, including the explicit geometric representation, the natural user interaction mechanisms, and the constraint mechanisms implemented through energy or force functions. Topology-adaptive deformable models can segment and reconstruct some of the most complex biological structures from 2D and 3D images. In this section, I will first review *topology-adaptive snakes*, or T-snakes, followed by *topology-adaptive deformable surfaces*, or T-surfaces.

2.3.1 Topology-Adaptive Snakes

T-snakes are hybrid models that combine Lagrangian, parametric snakes with aspects of the Eulerian, level set approach. They employ an Affine Cell Image Decomposition (ACID) of the image domain. The ACID extends the abilities of conventional snakes, enabling topological flexibility, among other features [354]. In particular, the ACID framework enables a novel snake reparameterization mechanism, which enables snakes to “flow” into geometrically complex objects, conforming to the object boundaries (Fig. 2.3(a)). One or more T-snakes can be dynamically created or destroyed and can seamlessly split or merge as necessary in order to adapt to object topology (Fig. 2.3(b)–(e)). See references [355] for numerous additional examples of T-snakes applied to images.

As a T-snake deforms under the influence of external and internal forces, it is systematically reparameterized with a new set of nodes and elements. This is done by efficiently computing the intersection points of the model with the su-

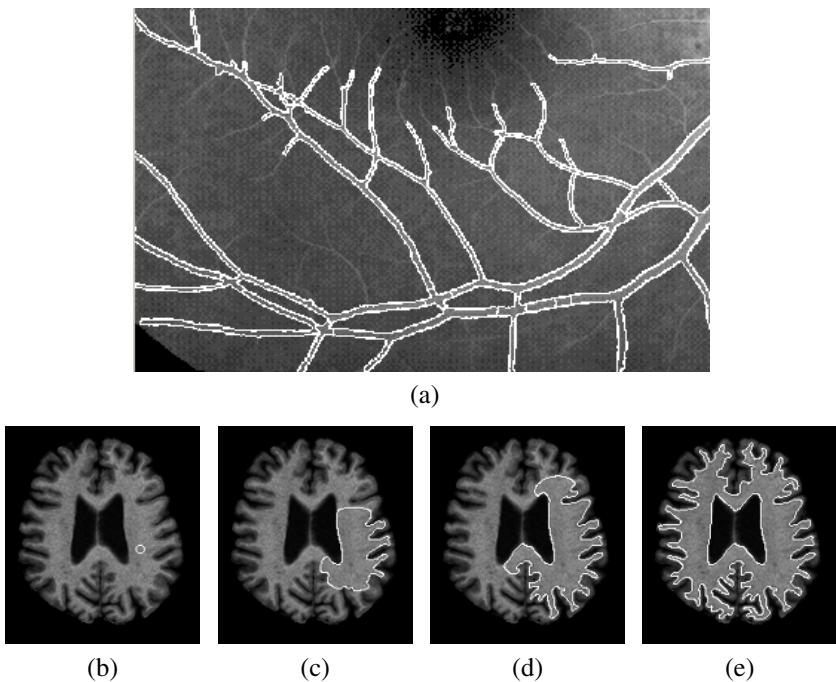


Figure 2.3. Segmentation with T-snakes. (a) T-snakes segmenting blood vessels in a retinal angiogram. Geometric flexibility allows the T-snakes to grow into the complex vessel shapes. (b–e) T-snake segmenting gray-matter/white-matter interface and ventricles in an MR brain image slice. The initially circular T-snake (b) changes its topology to a highly deformed annular region (e).

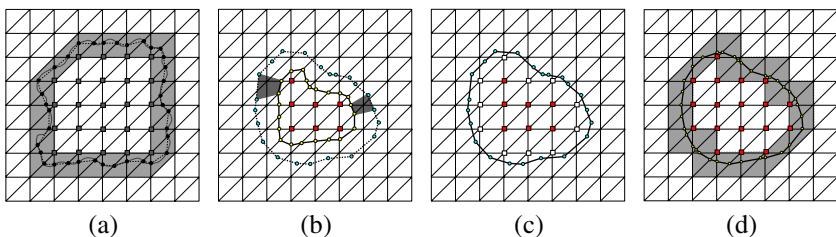


Figure 2.4. (a) Simplicial approximation (dashed-line) of an object contour (solid-line) using a Freudenthal triangulation. The model nodes (intersection points) are marked and the boundary triangles are shaded. (b–d) Illustration of the T-snake reparameterization process. (b) Shaded regions show examples of grid vertices that are turned on by the expanding contour, (c) new inside grid vertices (white) added to current inside vertices (dark), (d) new contour after one deformation step showing new grid intersections, inside grid vertices, and boundary grid cells (gray shaded).

perposed affine cell grid; for example, the Coxeter-Freudenthal decomposition (Fig. 2.4(a)). At the end of each deformation step, the nodes have moved relative to the grid cell edges (Fig. 2.4(b)–(d)). In phase I of the reparameterization algorithm, the intersection points between the T-snake elements and the grid cell edges are computed. These intersection points will become the nodes of the new T-snake. In phase II, grid cell vertices that have moved from the exterior to the interior of the T-snake are marked as “on”; in this manner, the interior of a T-snake is continuously tracked.

A closed T-snake defines a region. When a T-snake bounded region collides with itself or with another T-snake region, or splits into two or more subregions, (or shrinks and disappears,) a topological transformation must take place. Topology changes are performed automatically via the ACID (Fig. 2.3(b)–(e)). The boundary can always be determined unambiguously by keeping track of the inside grid vertices (and hence the boundary grid cells) and re-establishing the correspondence of the T-snake boundary with the grid after every deformation step. New elements are constructed based on the “signs” (i.e. inside or outside) of the grid vertices in each boundary cell and from the intersection points computed in phase I, such that the inside and outside grid vertices in these cells are separated by a single line.

Compared to conventional snakes, the T-snake is relatively insensitive to its initial placement within regions of interest in the image. It flows into complex shapes, modifying its topology as necessary in order to fit the relevant image data. The ACID provides a principled, computational geometry framework for topological transformations, but disabling the ACID reduces the T-snakes model to a conventional parametric snake model. Consequently, T-snakes also incorporate shape constraints in the form of energy functionals and applied forces. An important advantage of the latter is user control, which caters to medical image analysis, where it is often essential for an expert user to be able to control and refine the segmentation process in an interactive manner.

2.3.2 Topology-Adaptive Deformable Surfaces

The main components of the three-dimensional T-surfaces formulation (see [353] for the details) are analogous to those for the two-dimensional T-snakes. The first component is a discrete form of the conventional parametric deformable surfaces [351]. The second component is the extension of the ACID framework to three dimensions using simplicial (tetrahedral) cells or nonsimplicial (e.g. hexahedral) cells.² The third component of T-surfaces is a reparameterization process analogous to the one for T-snakes. To determine if a T-surface triangular element intersects a grid cell edge, a standard ray-triangle intersection algorithm is used

²Most nonsimplicial methods employ a rectangular tessellation of space. The formulation of T-surfaces using a non-simplicial grid is essentially identical to its simplicial counterpart except for the addition of the disambiguation scheme.

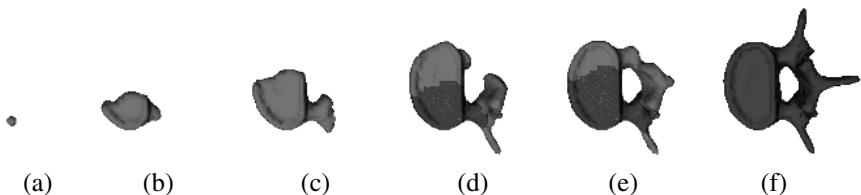


Figure 2.5. T-surface segmenting vertebra phantom from CT volume image [353].

and local neighborhood searches are employed to speed up the process. The user can interact with a T-surface by applying 3D interaction forces, by applying 2D interaction forces to cross-sections of the surface that are overlaid on image slices thorough the 3D dataset.

Fig. 2.5 demonstrates the topological adaptability of a T-surface when applied to a $120 \times 128 \times 52$ CT volume image of a human vertebra phantom. This example uses a $32 \times 30 \times 13$ cell grid (where each cubical cell is divided into 6 tetrahedra).

2.4 Generalized Deformable Models

Next, I will review the systematic Lagrangian formulation of physics-based deformable models that are built upon the standard geometric substrates of computer graphics and geometric modeling. These explicit geometric substrates are by no means trivial; certainly more complicated than those encountered in the previous two sections. Nonetheless they yield generalized varieties of deformable models possessing geometric degrees of freedom and novel physical behaviors that are beneficial in certain applications to computer-aided geometric design, computer graphics animation, and computer vision.

The first part of this section reviews *Dynamic Non-Uniform Rational B-Splines* (D-NURBS). The second part reviews *Deformable Superquadrics*. There exist other interesting generalized deformable model formulations that I will not review because of lack of space, among them “united snakes”—a hybrid between snakes and “livewire” segmentation tools based on dynamic programming [312]—and dynamic free-form deformations (FFDs) [185].

2.4.1 Dynamic NURBS

Deformable models have given impetus to a new, physics-based paradigm for computer-aided geometric design. The natural behavior of deformable models suggests a “computational modeling clay” metaphor which is particularly intuitive in the sculpting of free-form shapes [514]. An important goal, however, is to formulate physics-based modeling primitives that generalize the purely geometric free-form shape primitives employed in conventional shape design systems. To this end, we have developed a physics-based generalization of industry-standard, non-uniform rational B-splines (NURBS) and associated constraint

methods for physics-based geometric design and interactive sculpting [518]. The shape parameters of conventional, geometric NURBS play the role of generalized (physical) coordinates in “Dynamic NURBS” (D-NURBS). We introduce time, mass, and deformation energy into the standard NURBS formulation and employ Lagrangian dynamics to arrive at the system of nonlinear ordinary differential equations that govern the shape and motion of D-NURBS.

D-NURBS Curves:

A kinematic NURBS curve extends the geometric NURBS definition by explicitly incorporating time. The kinematic curve is a function of both the parametric variable u and time t :

$$\mathbf{c}(u, t) = \frac{\sum_{i=0}^n \mathbf{p}_i(t) w_i(t) B_{i,k}(u)}{\sum_{i=0}^n w_i(t) B_{i,k}(u)}, \quad (2.21)$$

where the $B_{i,k}(u)$ denote the usual recursively defined piecewise basis functions, $\mathbf{p}_i(t)$ are the $n + 1$ control points, and $w_i(t)$ are associated non-negative weights. Assuming basis functions of degree $k - 1$, the curve has $n + k + 1$ knots t_i in non-decreasing sequence: $t_0 \leq t_1 \leq \dots \leq t_{n+k}$.

To simplify notation, we define the vector of generalized coordinates $\mathbf{q}_i(t)$ and weights $w_i(t)$ as

$$\mathbf{q}(t) = [\mathbf{p}_0^\top \quad w_0 \quad \dots \quad \mathbf{p}_n^\top \quad w_n]^\top.$$

We then express the spline curve (2.21) as $\mathbf{c}(u, \mathbf{q})$.

The velocity of the kinematic spline is

$$\dot{\mathbf{c}}(u, \mathbf{q}) = \mathbf{J}\dot{\mathbf{q}}, \quad (2.22)$$

where the overstruck dot denotes a time derivative and $\mathbf{J}(u, \mathbf{q})$ is the Jacobian matrix. Because \mathbf{c} is a 3-component vector-valued function and \mathbf{q} is a $4(n + 1)$ dimensional vector, \mathbf{J} is the $3 \times 4(n + 1)$ matrix

$$\mathbf{J} = \left[\dots \begin{bmatrix} \frac{\partial \mathbf{c}_x}{\partial \mathbf{p}_{i,x}} & 0 & 0 \\ 0 & \frac{\partial \mathbf{c}_y}{\partial \mathbf{p}_{i,y}} & 0 \\ 0 & 0 & \frac{\partial \mathbf{c}_z}{\partial \mathbf{p}_{i,z}} \end{bmatrix} \frac{\partial \mathbf{c}}{\partial w_i} \dots \right], \quad (2.23)$$

where

$$\frac{\partial \mathbf{c}_x}{\partial \mathbf{p}_{i,x}} = \frac{\partial \mathbf{c}_y}{\partial \mathbf{p}_{i,y}} = \frac{\partial \mathbf{c}_z}{\partial \mathbf{p}_{i,z}} = \frac{w_i B_{i,k}}{\sum_{j=0}^n w_j B_{j,k}};$$

$$\frac{\partial \mathbf{c}}{\partial w_i} = \frac{\sum_{j=0}^n (\mathbf{p}_i - \mathbf{p}_j) w_j B_{i,k} B_{j,k}}{(\sum_{j=0}^n w_j B_{j,k})^2}.$$

The subscripts x , y , and z denote the components of a 3-vector.

The equations of motion of our D-NURBS are derived from Lagrangian dynamics. Applying the Lagrangian formulation to D-NURBS curves, we obtain

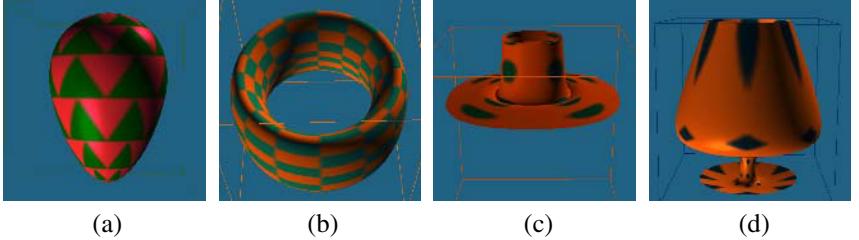


Figure 2.6. Interactive Sculpting of D-NURBS Swung Surfaces. Open and closed surfaces shown were sculpted interactively from prototype shapes noted in parentheses (a) Egg shape (sphere). (b) Deformed toroid (torus). (c) Hat (open surface). (d) Wine glass (cylinder) (from [429]).

the second-order nonlinear equations of motion

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{f}_q + \mathbf{g}_q, \quad (2.24)$$

where $\mathbf{M}(\mathbf{q})$ is the mass matrix, $\mathbf{D}(\mathbf{q})$ is the damping matrix, and $\mathbf{K}(\mathbf{q})$ is the stiffness matrix. The $N \times N$ mass and damping matrices are given by

$$\mathbf{M}(\mathbf{q}) = \int \mu \mathbf{J}^\top \mathbf{J} du; \quad \mathbf{D}(\mathbf{q}) = \int \gamma \mathbf{J}^\top \mathbf{J} du, \quad (2.25)$$

where $\mu(u, v)$ is the prescribed mass density function over the parametric domain of the surface and $\gamma(u, v)$ is the prescribed damping density function. To define an elastic potential energy for the D-NURBS curve, we use the snake energy (2.2). This yields the $N \times N$ stiffness matrix

$$\mathbf{K}(\mathbf{q}) = \int w_1(u) \mathbf{J}_u^\top \mathbf{J}_u + w_2(u) \mathbf{J}_{uu}^\top \mathbf{J}_{uu} du, \quad (2.26)$$

where the subscripts on \mathbf{J} denote parametric partial derivatives. As in snakes, the elasticity functions $w_1(u)$ and $w_2(u)$ control tension and rigidity, respectively. The generalized force $\mathbf{f}_p(\mathbf{q}) = \int \mathbf{J}^\top \mathbf{f}(u, t) du$, where $\mathbf{f}(u, t)$ is the applied force distribution. Because of the geometric nonlinearity, generalized inertial forces $\mathbf{g}_q(\mathbf{q})$ are also associated with the models (for the details, see [518]).

D-NURBS Surfaces:

Beyond D-NURBS curves, we have formulated three varieties of D-NURBS surfaces: tensor-product D-NURBS surfaces [518], swung D-NURBS surfaces [429], and triangular D-NURBS surfaces [430]. Examples of topologically different shapes interactively sculpted using D-NURBS swung surfaces are shown in Fig. 2.6.

For example, in analogy to the kinematic curve of (2.21), a tensor-product D-NURBS surface

$$\mathbf{s}(u, v, t) = \frac{\sum_{i=0}^m \sum_{j=0}^n \mathbf{p}_{i,j}(t) w_{i,j}(t) B_{i,k}(u) B_{j,l}(v)}{\sum_{i=0}^m \sum_{j=0}^n w_{i,j}(t) B_{i,k}(u) B_{j,l}(v)} \quad (2.27)$$

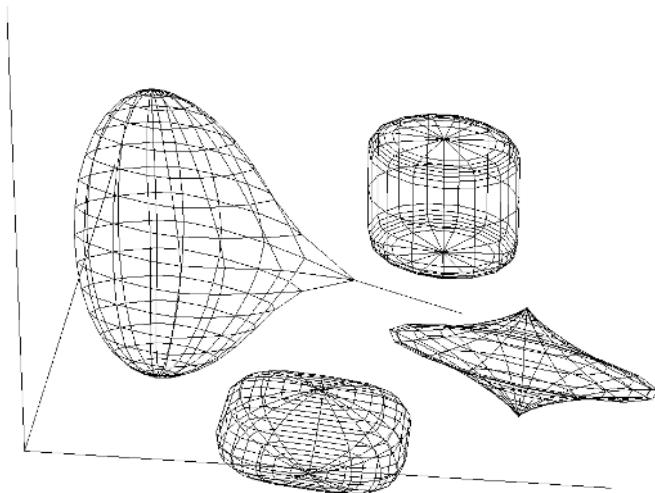


Figure 2.7. Interaction with deformable superquadrics.

generalizes the standard, geometric NURBS surface. The $(m + 1)(n + 1)$ control points $\mathbf{p}_{i,j}(t)$ and weights $w_{i,j}(t)$, which are functions of time, comprise the tensor-product D-NURBS surface generalized coordinates. We concatenate these $N = 4(m + 1)(n + 1)$ coordinates into the vector $\mathbf{p}(t)$. Analogous to (2.22), we have $\dot{\mathbf{s}}(u, v, \mathbf{p}) = \mathbf{J}\dot{\mathbf{p}}$, where $\mathbf{J}(u, v, \mathbf{p})$ is the $3 \times N$ Jacobian matrix of the D-NURBS surface with respect to \mathbf{p} . Note that \mathbf{J} is now a $3 \times 4(m + 1)(n + 1)$ matrix. Refer to [518] for the mathematical details and examples.

2.4.2 Deformable Superquadrics

We will next review a systematic Lagrangian approach for creating dynamic solid models capable of realistic physical behaviors, starting from the common, globally parameterized solid primitives, such as spheres, cylinders, cones, or superquadrics. Such primitives can deform kinematically in simple ways; for example, a geometric cylinder deforms as its radius or length is changed. To gain additional modeling power we allow the solid primitives to undergo parameterized global deformations (bends, tapers, twists, shears, etc.). To further enhance the geometric flexibility, we permit local free-form deformations. All of the geometric parameters are once again collected into a vector of generalized coordinates, along with the six degrees of freedom of rigid-body motion. As usual, Lagrange equations of motion govern the dynamics of these parameterized deformable models, dictating the evolution of the generalized coordinates in response to applied forces, so as to exhibit correct mechanical behavior subject to prescribed mass distributions, elasticities, and energy dissipation rates. Such models are useful in physics-based computer animation.

For example, Fig. 2.7 shows several deformable superquadrics. A superellipsoid is deforming in response to the traction from a linear spring attached to its surface and pulled interactively. In general, the models are abstract viscoelastic solids. It is possible, for instance, to mold a supersphere into any of the deformable superquadrics shown in the figure by applying appropriate forces.

Geometric Formulation:

A superquadric is a solid model whose intrinsic (material) coordinates are $\mathbf{u} = (u, v, w)$. Referring to Fig. 2.8, $\mathbf{x}(\mathbf{u}, t) = (x_1(\mathbf{u}, t), x_2(\mathbf{u}, t), x_3(\mathbf{u}, t))^\top$ gives the positions of points on the model relative to the fixed reference frame Φ . We can write

$$\mathbf{x} = \mathbf{c} + \mathbf{R}\mathbf{p}, \quad (2.28)$$

where $\mathbf{p}(\mathbf{u}, t)$ denotes the positions of points relative to the noninertial, model-centered frame ϕ whose instantaneous position is $\mathbf{c}(t)$ and orientation relative to Φ is given by the rotation matrix $\mathbf{R}(t)$. We further express

$$\mathbf{p} = \mathbf{s} + \mathbf{d}, \quad (2.29)$$

the sum of a reference shape $\mathbf{s}(\mathbf{u}, t)$ and a displacement function $\mathbf{d}(\mathbf{u}, t)$. We define the reference shape as

$$\mathbf{s} = \mathbf{T}(\mathbf{e}(\mathbf{u}; a_1, a_2, \dots); b_1, b_2, \dots). \quad (2.30)$$

Here, a geometric primitive \mathbf{e} , defined parametrically in \mathbf{u} and parameterized by the variables a_i , is subjected to the *global deformation* \mathbf{T} which depends on the parameters b_i . Although generally nonlinear, \mathbf{e} and \mathbf{T} are assumed to be differentiable (so that we may compute the Jacobian of \mathbf{s}) and \mathbf{T} may be a composite sequence of primitive deformation functions. We define the vector of global deformation parameters

$$\mathbf{q}_s = (a_1, a_2, \dots, b_1, b_2, \dots)^\top. \quad (2.31)$$

Next, we express the displacement as a linear combination of basis functions $\mathbf{b}_i(\mathbf{u})$. The basis functions can be local or global; however, finite element shape functions are the natural choice for representing *local deformations*

$$\mathbf{d} = \mathbf{S}\mathbf{q}_d. \quad (2.32)$$

Here \mathbf{S} is a shape matrix whose entries are the shape functions and

$$\mathbf{q}_d = (\dots, \mathbf{d}_i^\top, \dots)^\top \quad (2.33)$$

is the vector of local deformation parameters. Typically, finite elements have nodes at their vertices, and the parameter \mathbf{q}_i denotes a displacement vector associated with node i of the model.

In [516, 362] we provide the formulas for a superquadric ellipsoid \mathbf{e} with tapering, bending, shearing, and twisting deformations.

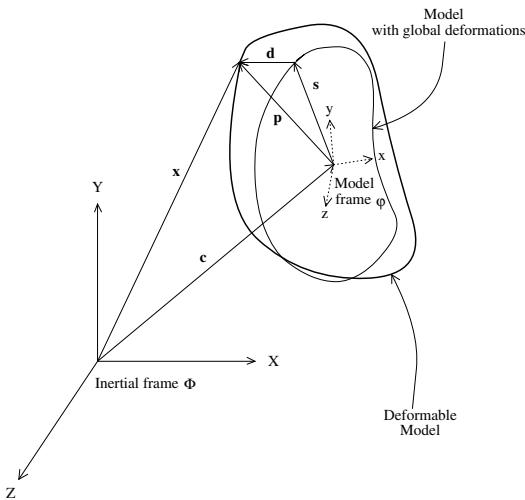


Figure 2.8. Geometric structure.

Kinematics and Dynamics:

To convert the geometric representation into a physical model that responds dynamically to forces, we first consider the kinematics implied by the geometry and then introduce mass, damping, and elasticity into the model to derive its mechanics.

The velocity of points on the model is given by,

$$\dot{\mathbf{x}} = \dot{\mathbf{c}} + \dot{\mathbf{R}}\mathbf{p} + \mathbf{R}\dot{\mathbf{p}} = \dot{\mathbf{c}} + \mathbf{B}\dot{\boldsymbol{\theta}} + \mathbf{R}\dot{\mathbf{s}} + \mathbf{R}\mathbf{S}\dot{\mathbf{q}}_d, \quad (2.34)$$

where $\boldsymbol{\theta} = (\dots, \theta_i, \dots)^\top$ is a vector of rotational coordinates and the matrix $\mathbf{B} = [\dots \partial(\mathbf{R}\mathbf{p})/\partial\theta_i \dots]$. Now, $\dot{\mathbf{s}} = [\partial\mathbf{s}/\partial\mathbf{q}_s]\mathbf{q}_s = \mathbf{J}\mathbf{q}_s$, where \mathbf{J} is the Jacobian of the reference shape with respect to the global deformation parameter vector. We can therefore write the model kinematics compactly as

$$\mathbf{x} = \mathbf{c} + \mathbf{R}(\mathbf{s} + \mathbf{d}) = \mathbf{h}(\mathbf{q}), \quad (2.35)$$

$$\dot{\mathbf{x}} = [\mathbf{I} \ \mathbf{B} \ \mathbf{R}\mathbf{J} \ \mathbf{R}\mathbf{S}]\dot{\mathbf{q}} = \mathbf{L}\dot{\mathbf{q}}, \quad (2.36)$$

where

$$\mathbf{q} = (\mathbf{q}_c^\top, \mathbf{q}_\theta^\top, \mathbf{q}_s^\top, \mathbf{q}_d^\top)^\top, \quad (2.37)$$

(with $\mathbf{q}_c = \mathbf{c}$ and $\mathbf{q}_\theta = \boldsymbol{\theta}$) serves as the vector of generalized coordinates for the dynamic model.

To specify the dynamics, we introduce a mass distribution $\mu(u)$ over the model and assume that the material is subject to frictional damping. We also assume that the material may deform elastically or viscoelastically [515]. Applying the Lagrangian mechanics approach, we obtain second-order equations of motion which

take the form

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{g}_q + \mathbf{f}_q. \quad (2.38)$$

The mass matrix $\mathbf{M} = \int \mu \mathbf{L}^\top \mathbf{L} du$. The stiffness matrix \mathbf{K} may be obtained from a deformation strain energy $(\mathbf{q}^\top \mathbf{K} \mathbf{q})/2$. The Raleigh damping matrix $\mathbf{D} = \alpha \mathbf{M} + \beta \mathbf{K}$. The generalized inertial forces $\mathbf{g}_q = -\int \mu \mathbf{L}^\top \dot{\mathbf{L}} \dot{\mathbf{q}} du$ include generalized centrifugal, Coriolis, and transverse forces due to the dynamic coupling between \mathbf{q}_θ , \mathbf{q}_s , and \mathbf{q}_d . Finally, $\mathbf{f}_q = \int \mathbf{L}^\top \mathbf{f} du$ are generalized external forces associated with the components of \mathbf{q} , where $\mathbf{f}(u, t)$ is the force distribution applied to the model. See [516, 362] for explicit formulas for the above matrices and vectors.

Multibody Constraints:

Deformable superquadric models raise interesting challenges related to the application of constraints in order to construct composite models and control animation. We describe a method for computing generalized constraint forces between our models which is based on Baumgarte's constraint stabilization technique. Our algorithm may be used to assemble complex objects satisfying constraints from initially mispositioned and misshaped parts, and it enables us to construct and animate articulated objects composed of rigid or nonrigid components. We have applied these multibody models to computer animation [362] and computer vision [363].

We can extend (2.38) to account for the motions of composite models with interconnected deformable parts by forming a composite generalized coordinate vector \mathbf{q} , as well as force vectors \mathbf{g}_q and \mathbf{f}_q for an n -part model by concatenating the \mathbf{q}_i , \mathbf{g}_{q_i} , and \mathbf{f}_{q_i} associated with each part $i = 1, \dots, n$. Similarly, the composite matrices \mathbf{M} , \mathbf{D} , and \mathbf{K} for the n -part model are block diagonal matrices with submatrices \mathbf{M}_i , \mathbf{D}_i , and \mathbf{K}_i , respectively, for each part i . The method solves the composite equations of motion $\mathbf{M}\ddot{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{g}_q + \mathbf{f}_q - \mathbf{C}_q^\top \boldsymbol{\lambda}$. The generalized constraint forces $\mathbf{f}_{g_c} = -\mathbf{C}_q^\top \boldsymbol{\lambda}$ acting on the parts stem from the holonomic constraint equations

$$\mathbf{C}(\mathbf{q}, t) = \mathbf{0}; \quad (2.39)$$

i.e., $\mathbf{C} = [\mathbf{C}_1^\top, \mathbf{C}_2^\top, \dots, \mathbf{C}_k^\top]^\top$ expresses k constraints among the n parts of the model. The term \mathbf{C}_q^\top is the transpose of the constraint Jacobian matrix and $\boldsymbol{\lambda} = (\lambda_1^\top, \dots, \lambda_n^\top)^\top$ is a vector of Lagrange multipliers that must be determined.

To obtain an equal number of equations and unknowns, we differentiate (2.39) twice with respect to time, yielding $\boldsymbol{\sigma} = \mathbf{C}_q \ddot{\mathbf{q}} = -\mathbf{C}_{tt} - (\mathbf{C}_q \dot{\mathbf{q}})\mathbf{q} \dot{\mathbf{q}} - 2\mathbf{C}_{qt} \dot{\mathbf{q}}$. The augmented equations of motion are

$$\begin{bmatrix} \mathbf{M} & \mathbf{C}_q^\top \\ \mathbf{C}_q & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} -\mathbf{D}\dot{\mathbf{q}} - \mathbf{K}\mathbf{q} + \mathbf{g}_q + \mathbf{f}_q \\ \boldsymbol{\sigma} - 2\alpha \dot{\mathbf{C}} - \beta^2 \mathbf{C} \end{bmatrix}. \quad (2.40)$$

Fast constraint stabilization means choosing $\beta = \alpha$ to obtain the critically damped solution $\mathbf{C}(\mathbf{q}, 0)e^{-\alpha t}$ which, for given α , has the quickest asymptotic decay towards constraint satisfaction $\mathbf{C} = \mathbf{0}$.

2.5 Conclusion

Deformable models are a rich family of physics-based modeling primitives that have seen extensive use in computer vision and graphics. They have also been applied heavily in the associated areas of medical image analysis and geometric design. In this chapter, I have reviewed

1. the formulation of classic deformable models, including snakes and higher-dimensional models;
2. topology-adaptive deformable models that offer the topological flexibility associated with level set methods, but retain the parametric, Lagrangian formulation;
3. formulations of generalized Lagrangian deformable models that are built on nontrivial geometric foundations, such as non-uniform rational B-splines and superquadrics.

Lagrangian models of solids complement Eulerian models of fluids in continuum mechanics. Deformable models and level set methods are complementary techniques in precisely the same sense. Together, these methods have given impetus to a voluminous and rapidly growing literature. They continue to have a promising future in multiple application areas.

Acknowledgments: I thank Nikos Paragios and Stan Osher for encouraging me to prepare this chapter. I am grateful to many colleagues who have collaborated with me in the development of deformable model theory and applications; their names appear in the citations: Carlbom, Faloutsos, Fleischer, Kass, Liang, McInerney, Metaxas, Qin, Szeliski, Witkin. In view of the material presented in this chapter, I would like especially to acknowledge the key contributions of three former PhD students: Tim McInerney, whose dissertation developed topology-adaptive deformable models, Dimitri Metaxas, whose dissertation developed deformable superquadrics, and Hong Qin, whose dissertation developed D-NURBS.

Part II

Edge Detection & Boundary Extraction

3

Fast Methods for Implicit Active Contour Models

Joachim Weickert and Gerald Kühne

Abstract

Implicit active contour models belong to the most popular level set methods in computer vision. Typical implementations, however, suffer from poor efficiency. In this chapter we survey an efficient algorithm that is based on an additive operator splitting (AOS). It is suitable for geometric and geodesic active contour models as well as for mean curvature motion. It uses harmonic averaging and does not require to compute the distance function in each iteration step. We prove that the scheme satisfies a discrete maximum-minimum principle which implies unconditional stability if no balloon forces are present. Moreover, it possesses all typical advantages of AOS schemes: simple implementation, equal treatment of all axes, suitability for parallel computing, and straightforward generalization to higher dimensions. Experiments show that one can gain a speed up by one order of magnitude compared to the widely used explicit time discretization.

3.1 Introduction

Active contour models (also called deformable models or snakes) [263] have been used in a variety of different image processing and computer vision tasks, ranging from interactive image segmentation to object tracking in image sequences. The basic idea is that the user specifies an initial guess of an interesting contour (e.g. an organ, a tumour, or a person to be tracked). Then this contour is moved by image-driven forces to the boundaries of the desired object.

Implicit active contour models [81] constitute a very interesting applications of level set ideas within the active contour framework. They embed the active contour as a level set in a suitable image evolution that is determined by a partial differential equation (PDE). Then the final contour is extracted when the evolution is stopped. The main advantages of implicit active contours over classical

explicit snakes are the automatic handling of topological changes, high numerical stability and independence of parametrization. However, their main drawback is the additional computational complexity. In their simplest implementation, most approaches are based on an explicit or forward Euler scheme which requires very small time steps. This severely limits their efficiency.

A number of fast implementations for implicit snakes have been proposed to circumvent this problem. Often they concentrate on narrow-band techniques and multi-scale computations [2, 421, 412], but more recently also methods based on additive operator splittings (AOS) have become popular [216, 215, 296, 415, 566]. It is the goal of this chapter to give an introduction to AOS schemes for implicit active contour models and related PDEs.

AOS schemes has been introduced to image analysis in the context of nonlinear diffusion filtering [570]. They have been used for medical imaging problems [435], for regularization methods [567], image registration [194] and for optic flow computations [569]. The basic idea behind AOS schemes is to decompose a multi-dimensional problem into one-dimensional ones that can be solved very efficiently. Then the final multi-dimensional solution is approximated by averaging the one-dimensional solutions. AOS schemes perform well not only on sequential architectures: experiments have demonstrated that they are also well-suited for parallel computing on systems with shared [568] as well as distributed memory [72].

The usefulness of AOS ideas has also been shown in a number of other applications ranging from Navier–Stokes equations [511, 324] to sandpile growth simulations [238]. It seems that Navier–Stokes equations have constituted one of their historically first application domains.

The description in this chapter follows our previous publications [296, 566, 570]. Our approach is suitable both for the geometric [81] and the geodesic active contour model [85, 271], and it may also be used for mean curvature motion [10, 275]. It differs from recent work by Goldenberg et al. [216, 215] by the fact that it does not require to recompute a distance transformation in each iteration step. This may lead to significant savings in computation time. Indeed, we shall see that – under realistic accuracy requirements – AOS schemes allow to speed up implicit active contour models by one order of magnitude.

The present chapter is organized as follows: Section 3.2 introduces the geometric and the geodesic active contour model. Section 3.3 describes our numerical implementation of both models based on the AOS scheme. Section 3.4 presents an evaluation of the accuracy and efficiency of the AOS scheme. Finally, Section 3.5 concludes the chapter by giving a summary and mentioning extensions that can further increase the computational efficiency.

3.2 Implicit Active Contour Models

In active contour models one places a closed planar parametric curve $C_0(s) = (x(s), y(s))$, $s \in [0, 1]$, around image parts of interest. Then this curve evolves under smoothness control (internal energy) and the influence of an image force (external energy).

In the classical *explicit* snake model [263] the parametric curve is embedded into an energy minimization framework. Apart from energy minimization the parametric curve can also evolve directly under motion equations derived from geometric considerations [470].

However, the parametrization of the curve causes difficulties with respect to topological changes and numerical implementations. Thus, to prevent these difficulties, *implicit* active contour models have been developed. Here the basic idea is to represent the initial curve $C_0(s)$ *implicitly* within a higher dimensional function, and to evolve this function under a partial differential equation. Usually, C_0 is embedded as a zero level set into a function $u_0 : \mathbb{R}^2 \rightarrow \mathbb{R}$ by using the *signed distance function*:

$$u_0(x) = \begin{cases} d(x, C_0), & \text{if } x \text{ is inside } C_0 \\ 0, & \text{if } x \text{ is on } C_0 \\ -d(x, C_0), & \text{if } x \text{ is outside } C_0, \end{cases} \quad (3.1)$$

where $d(x, C_0)$ denotes the distance between some point x and the curve C_0 .

The *implicit geometric active contour model* discovered by Caselles et al. [81] includes geometrical considerations similar to [470]. Let $\Omega := (0, a_x) \times (0, a_y)$ be our image domain in \mathbb{R}^2 . We consider a scalar image $u_0(x)$ on Ω . Then, the geometric active contour model investigates the evolution of u_0 under the PDE

$$\begin{aligned} \frac{\partial u}{\partial t} &= g(x) |\nabla u| \left(\operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right) + k \right) \quad \text{on } \Omega \times (0, \infty), \\ u(x, 0) &= u_0(x) \quad \text{on } \Omega. \end{aligned} \quad (3.2)$$

Here, k is a constant force term comparable to the balloon force [127] known from explicit models, and $g : \mathbb{R}^2 \rightarrow (0, 1]$ denotes a stopping function that slows down the snake as it approaches selected image features such as edges. Note that normal and curvature to a level set are given by

$$n = -\frac{\nabla u}{|\nabla u|}, \quad (3.3)$$

$$\kappa = \operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right) = \frac{u_{xx}u_y^2 - 2u_xu_yu_{xy} + u_{yy}u_x^2}{(u_x^2 + u_y^2)^{3/2}}. \quad (3.4)$$

In the *implicit geodesic active contour model* proposed simultaneously by Caselles et al. [84] and Kichenassamy et al. [271] the function u is embedded into an energy functional that can be related to the explicit snake model. The



Figure 3.1. Temporal evolution of a geodesic active contour superimposed on the original image with $\Omega = (0, 256)^2$. From left to right: $t = 0, 1500, 7500$. From [566].

corresponding evolution equation is given by

$$\begin{aligned} \frac{\partial u}{\partial t} &= |\nabla u| \left(\operatorname{div} \left(g(x) \frac{\nabla u}{|\nabla u|} \right) + k g(x) \right) && \text{on } \Omega \times (0, \infty), \\ u(x, 0) &= u_0(x) && \text{on } \Omega. \end{aligned} \quad (3.5)$$

Figure 3.1 gives an example of a geodesic active contour evolution.

3.3 Numerical Implementation

While implicit active contour models avoid several of the difficulties known from explicit models, their main disadvantage is poor efficiency. First, in their simplest implementation, the partial differential equation must be evaluated on the complete image domain. Second, most approaches are based on explicit updating schemes which require very small time steps. While the first limitation can be addressed by narrow-band and/or multi-scale techniques [2, 476, 421], the latter requires different discretizations. In the following we focus on the second problem and develop semi-implicit schemes for both the geometric and the geodesic active contour model based on the additive operator splitting (AOS) scheme. Note that narrow-band and multi-scale techniques can be easily combined with our implementation.

A Unified Model. Let us consider the following equation, which unifies the geometric and the geodesic model by introducing two additional functions a and b :

$$\frac{\partial u}{\partial t} = a(x) |\nabla u| \operatorname{div} \left(\frac{b(x)}{|\nabla u|} \nabla u \right) + |\nabla u| k g(x). \quad (3.6)$$

Setting $a := g$ and $b := 1$ yields the geometric model, while $a := 1$ and $b := g$ results in the geodesic model. Moreover, for $a := b := 1$ and $k := 0$, we obtain

the mean curvature motion

$$\frac{\partial u}{\partial t} = |\nabla u| \operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right), \quad (3.7)$$

which plays an important role in image denoising and morphological scale-space analysis [9, 10, 274, 275].

Semi-Implicit Scheme. Now we are in a position to derive suitable numerical schemes for our unified model. For the sake of clarity we assume a constant force $k = 0$ in the following and discuss the integration of the balloon force later on. Interpreting the term $\frac{b(x)}{|\nabla u|}$ as “diffusivity” we can employ techniques similar to those as described in [570] in the context of nonlinear diffusion filtering. To derive a numerical algorithm one has to consider discretizations of space and time. We employ discrete times $t_n := n\tau$, where $n \in \mathbb{N}_0$ and τ denotes the time step size. Additionally, an image is divided by a uniform mesh of spacing $h = 1$ into grid nodes (i, j) . Using standard notation, u_{ij}^n denotes the approximation of $u(ih, jh, t_n)$.

Let us recall a simple spatial discretization of the term $\operatorname{div} \left(\frac{b}{|\nabla u|} \nabla u \right)$ since it is a prerequisite for our semi-implicit scheme. Setting $c := \frac{b}{|\nabla u|}$ the term may be approximated as follows:

$$\begin{aligned} \operatorname{div}(c \nabla u) &\approx \partial_x \left(c_{ij} \frac{u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}}{h} \right) + \partial_y \left(c_{ij} \frac{u_{i,j+\frac{1}{2}} - u_{i,j-\frac{1}{2}}}{h} \right) \\ &\approx c_{i+\frac{1}{2},j} \frac{u_{i+1,j} - u_{ij}}{h^2} - c_{i-\frac{1}{2},j} \frac{u_{ij} - u_{i-1,j}}{h^2} \\ &\quad + c_{i,j+\frac{1}{2}} \frac{u_{ij+1} - u_{ij}}{h^2} - c_{i,j-\frac{1}{2}} \frac{u_{ij} - u_{ij-1}}{h^2} \end{aligned} \quad (3.8)$$

The values for $c_{i\pm\frac{1}{2},j}$ and $c_{i,j\pm\frac{1}{2}}$ can be determined by linear interpolation.

To simplify the notation in the following, a discrete image is represented further on by a vector $f \in \mathbb{R}^N$, whose components $f_i, i \in \{1, \dots, N\}$, contain the pixel values. For instance, this vector might be constructed by concatenating the rows of an image. Consequently, pixel i corresponds to some grid node x_i . Thus, u_i^n denotes the approximation of $u(x_i, t_n)$. Hence, following [570] and using the above mentioned discretization, Equation 3.6 with $k = 0$ reads in its semi-implicit formulation as

$$u_i^{n+1} = u_i^n + \tau a_i |\nabla u|_i^n \sum_{j \in \mathcal{N}(i)} \frac{\left(\frac{b}{|\nabla u|}\right)_i^n + \left(\frac{b}{|\nabla u|}\right)_j^n}{2} \frac{u_j^{n+1} - u_i^{n+1}}{h^2}, \quad (3.9)$$

where $\mathcal{N}(i)$ denotes the 4-neighbourhood of the pixel at position x_i . However, in order to compute Equation 3.9 one must assure that $|\nabla u|$ does not vanish in the 4-neighbourhood. Here, straightforward finite difference implementations

would give rise to problems. These problems do not appear if one uses a finite difference scheme with *harmonic averaging* [566], thus replacing the arithmetic mean $\frac{1}{2} \left(\left(\frac{b}{|\nabla u|} \right)_i^n + \left(\frac{b}{|\nabla u|} \right)_j^n \right)$ in Equation 3.9 by its harmonic counterpart. This yields

$$u_i^{n+1} = u_i^n + \tau a_i |\nabla u|_i^n \sum_{j \in \mathcal{N}(i)} \frac{2}{\left(\frac{|\nabla u|}{b} \right)_i^n + \left(\frac{|\nabla u|}{b} \right)_j^n} \frac{u_j^{n+1} - u_i^{n+1}}{h^2}. \quad (3.10)$$

Note that by evaluating only image positions with $|\nabla u|_i \neq 0$, the denominator in this scheme cannot vanish. If $|\nabla u|_i = 0$, one sets $u_i^{n+1} := u_i^n$. In general such an harmonic averaging scheme may turn out to be rather dissipative. For the specific application to active contour models, however, this does not seem to create specific problems.

In matrix-vector notation, Equation 3.10 becomes

$$u^{n+1} = u^n + \tau \sum_{l \in \{x, y\}} A_l(u^n) u^{n+1}, \quad (3.11)$$

where A_l describes the interaction in l direction. In detail, the matrix $A_l(u^n) = (\hat{a}_{ijl}(u^n))$ is given by

$$\hat{a}_{ijl}(u^n) := \begin{cases} a_i |\nabla u|_i^n \frac{2}{\left(\frac{|\nabla u|}{b} \right)_i^n + \left(\frac{|\nabla u|}{b} \right)_j^n}, & j \in \mathcal{N}_l(i) \\ -a_i |\nabla u|_i^n \sum_{m \in \mathcal{N}_l(i)} \frac{2}{\left(\frac{|\nabla u|}{b} \right)_i^n + \left(\frac{|\nabla u|}{b} \right)_m^n}, & j = i \\ 0, & \text{else,} \end{cases} \quad (3.12)$$

where $\mathcal{N}_l(i)$ represents the neighbouring pixels with respect to direction $l \in \{x, y\}$. However, the solution u^{n+1} cannot be directly determined from this scheme. Instead, it requires to solve the linear system of equations

$$\left(I - \tau \sum_{l \in \{x, y\}} A_l(u^n) \right) u^{n+1} = u^n. \quad (3.13)$$

where I denotes the unit matrix.

Since the system matrix is strictly diagonally dominant, it follows from Gershgorin's theorem that it is invertible [542]. In practice, however, it may be rather expensive to solve such a linear system in the 2-D case. Since its number of unknowns coincides with the pixel number, it is typically a very large sparse system with at most five nonvanishing entries per row. Although the actual structure of the system matrix depends on the pixel numbering, it is not possible to order the pixels in such a way that in the i -th row all nonvanishing matrix elements can be found within the positions $[i, i - 2]$ to $[i, i + 2]$: Usually, the matrix reveals a much larger bandwidth. Applying direct algorithms such as Gaussian elimination would destroy the zeros within the band and would lead to an immense storage and computation effort.

Hence, iterative algorithms should be applied. Classical methods like Gauß–Seidel or SOR iterations [599] do not need additional storage, and convergence can be guaranteed for the special structure of the system matrix. This convergence, however, may be rather slow since the condition number of the system matrix increases with the image resolution. Faster iterative methods such as pre-conditioned conjugate gradient methods [453] need significantly more storage, which can become prohibitive for very large images or 3-D problems. Iterative methods suffer also from the fact that their convergence slows down for increasing τ , since this increases the condition number of the system matrix. Multigrid methods [69] appear to be one possibility to circumvent many of these problems, but their implementation is more complicated.

In the following we shall focus on a splitting-based alternative. It is simple to implement and does not require to specify any additional parameters. This may make it attractive in a number of practical applications.

AOS Scheme. Instead of using the semi-implicit scheme

$$u^{n+1} = \left(I - \tau \sum_{l \in \{x,y\}} A_l(u^n) \right)^{-1} u^n \quad (3.14)$$

we may consider its *additive operator splitting (AOS)* variant

$$u^{n+1} = \frac{1}{2} \sum_{l \in \{x,y\}} (I - 2\tau A_l(u^n))^{-1} u^n. \quad (3.15)$$

By means of a Taylor expansion it is easy to see that the semi-implicit scheme and its AOS version differ by an $O(\tau^2)$ term. However, this does not create any problems since the time discretization in the semi-implicit scheme has already introduced an error of the same order. Hence, from an approximation viewpoint, both schemes have the same order of numerical consistency to the continuous equation.

The AOS scheme, however, offers one important advantage: The operators $B_l(u^k) := I - 2\tau A_l(u^n)$ lead to strictly diagonally dominant tridiagonal linear systems which can be solved very efficiently with a Gaussian algorithm (also called Thomas algorithm in this context [384]). This algorithm has linear complexity and can be implemented very easily. Details are given in the appendix.

In order to implement Equation 3.15, one proceeds in three steps:

1. Evolution in x direction with step size 2τ :

Solve the tridiagonal system $(I - 2\tau A_x(u^n)) v^{n+1} = u^n$ for v^{n+1} .

2. Evolution in y direction with step size 2τ :

Solve the tridiagonal system $(I - 2\tau A_y(u^n)) w^{n+1} = u^n$ for w^{n+1} .

3. Averaging:

Compute $u^{n+1} := 0.5(v^{n+1} + w^{n+1})$.

The fact that AOS schemes are based on an *additive* splitting guarantees that both axes are treated in exactly the same manner. This is in contrast to conven-

tional splitting techniques from the literature such as ADI methods, D'yakonov splitting or LOD techniques [338, 372, 587]: they are *multiplicative* and may produce different results in the nonlinear setting if the image is rotated by 90 degrees, since the operators do not commute.

One should notice that AOS schemes are also well-suited for parallel computing as they possess two granularities of parallelism:

- Coarse grain parallelism: The evolution in different directions can be performed simultaneously on different processors.
- Mid grain parallelism: Each 1-D evolution decouples into independent evolutions along each row or column.

AOS schemes are not only efficient, they are also unconditionally stable. This can be seen as follows: Since $B_l(u^k)$ is strictly diagonally dominant, $b_{il} > 0$ for all i and $b_{il} \leq 0$ for $i \neq j$, we may conclude from [357, p. 192] that B_l^{-1} is nonnegative in all its arguments. Moreover, from the fact that all row sums of $A_l(u^n)$ vanish, it follows that $B_l(u^n)$ and $Q(u^n) := \frac{1}{2} \sum_l B_l^{-1}(u^n)$ have row sums 1. Together with the nonnegativity this implies that the AOS scheme $u^{n+1} = Q(u^n)u^n$ computes u^{n+1} from convex combinations of the elements of u^n . This guarantees the discrete maximum–minimum principle

$$\min_j u_j^n \leq u_i^{n+1} \leq \max_j u_j^n \quad \forall i. \quad (3.16)$$

which implies stability of the scheme in the maximum norm for all time step sizes τ .

In practice, it makes of course not much sense to use extremely large time steps, since the accuracy will deteriorate significantly and splitting artifacts may become visible. Experiments show that, if a spatial grid size of 1 is used and if $a(x)b(x) \leq 1$, a time step size of $\tau = 5$ is a good compromise between accuracy and efficiency.

Supplementing the balloon force. So far we have neglected the constant force term $|\nabla u|kg$ (cf. Equation 3.6). This term stems from the hyperbolic dilation/erosion equation $\partial_t u = \pm |\nabla u|$. Consequently, it is advantageous to approximate the gradient by an upwind scheme [401]:

$$|\nabla u|_i^n \approx \begin{cases} |\nabla^- u|_i^n = (\max(D^{-x} u_i^n, 0)^2 + \min(D^{+x} u_i^n, 0)^2 + \\ \quad \max(D^{-y} u_i^n, 0)^2 + \min(D^{+y} u_i^n, 0)^2)^{1/2}, & \text{if } k \leq 0 \\ |\nabla^+ u|_i^n = (\min(D^{-x} u_i^n, 0)^2 + \max(D^{+x} u_i^n, 0)^2 + \\ \quad \min(D^{-y} u_i^n, 0)^2 + \max(D^{+y} u_i^n, 0)^2)^{1/2}, & \text{if } k > 0 \end{cases}, \quad (3.17)$$

where D^{+x} , D^{+y} , D^{-x} , and D^{-y} denote forward resp. backward approximations of the spatial derivatives (see e. g. [476]). Integrating the constant force term into

Equation 3.15 is straightforward and yields for $k < 0$:

$$u^{n+1} = \frac{1}{2} \sum_{l \in \{x, y\}} (I - 2\tau A_l(u^n))^{-1} (u^n + \tau |\nabla^- u|^n k g). \quad (3.18)$$

Since the dilation/erosion equation approximated on a grid with size $h = 1$ can be shown [401] to be stable only for $\tau \leq 0.5$, the constant force term limits the applicable time step. Consequently, Equation 3.18 is stable only for $|\tau k g| \leq 0.5$. However, since g is bounded by one, k is usually a small fraction of 1.0, and very large time steps ($\tau > 5.0$) degrade the accuracy of the AOS scheme significantly, this constraint is not severe.

3.4 Experimental Results

In this section we evaluate the accuracy and the efficiency of AOS schemes for implicit active contour models. For the accuracy evaluation we focus on a specific example where an analytic solution is known, and efficiency is studied by comparing the semi-implicit AOS schemes to a corresponding explicit (Euler forward) discretization in time.

3.4.1 Accuracy Evaluation in Case of Mean Curvature Motion

In order to assess the numerical errors of our AOS scheme, we consider in our first experiment an evolution of a disk under mean curvature motion. Since it is well-known that a disk-shaped level set with area $S(0)$ shrinks under mean curvature motion such that

$$S(t) = S(0) - 2\pi t, \quad (3.19)$$

simple accuracy evaluations are possible. To this end we use a distance transformation of some disk-shaped initial image and consider the evolution of a level set with an initial area of 24845 pixels. Table 3.1 shows the area errors for different time step sizes τ and two stopping times. We observe that for $\tau \leq 5$, the accuracy is sufficiently high for typical image processing applications. Figure 3.2 demonstrates that in this case no violations regarding rotational invariance are visible.

3.4.2 Efficiency Gain for Implicit Active Contour Models

With the AOS-based implementation it is possible to choose time steps much larger than in explicit updating schemes. Consequently, the evolution of the contour to its final location requires only a small number of iterations compared to explicit algorithms. However, a semi-implicit AOS iteration is more expensive than its explicit counterpart. In order to compare both approaches, we implemented the AOS-based models according to Equation 3.18. For the explicit model

Table 3.1. Area errors for the evolution of a disk-shaped area of 24845 pixels under mean curvature motion using an AOS-based semi-implicit scheme with harmonic averaging. The pixels have size 1 in each direction. From [566].

step size τ	stopping time $T = 2250$	stopping time $T = 3600$
0.5	-0.27 %	-0.60 %
1	-0.26 %	-0.88 %
2	-0.27 %	-0.88 %
5	-0.34 %	-1.73 %
10	-5.18 %	51.20 %

we employed standard techniques [476, 24]. In addition, we used a stopping criterion to indicate that the curve location has stabilized. Every time a certain period Δt_k has elapsed the average gray value of the evolving image u is calculated. E. g., when setting $\Delta t_k = 50$ and $\tau = 0.25$, the average gray value is computed every 200 iterations. The process stops if two consecutive measurements differ by less than an accuracy parameter α . In all experiments the parameters for the stopping criterion were set to $\Delta t_k = 50$ and $\alpha \in \{0.01, 0.1\}$. To assess the final placement of the contour with regard to the underlying algorithm, a simple distance measure was developed. Given a result contour and a reference contour, we calculated for each pixel on the result contour the distance to the nearest pixel on the reference contour. Averaging these distances over all result contour pixels yields the average distance between the two contours. As reference contour we used in all cases the explicit implementation with a small time step $\tau = 0.1$.

We applied both algorithms to sample images (cf. Figures 3.3–3.5). A stopping function according to the Perona-Malik diffusivity [423] was used:

$$g(x) = \frac{1}{1 + |\nabla f_\sigma(x)|^2 / \lambda^2}, \quad (3.20)$$

where f_σ denotes the convolution of image f with a Gaussian kernel of standard deviation σ , and λ is a contrast factor. Close to edges (high gradient magnitudes)

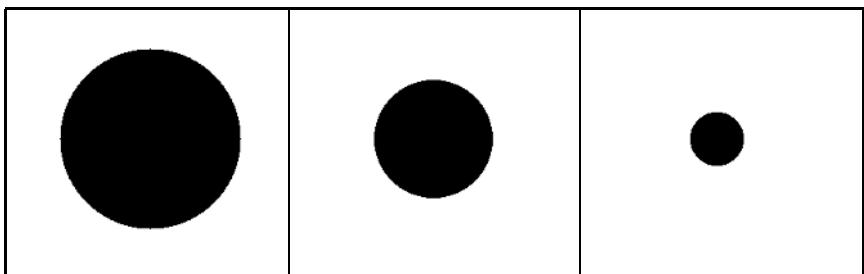


Figure 3.2. Temporal evolution of a disk-shaped level set under mean curvature motion. The results have been obtained using an AOS-based semi-implicit scheme with harmonic averaging and step size $\tau = 5$. From left to right: $t = 0, 2250, 3600$. From [566].

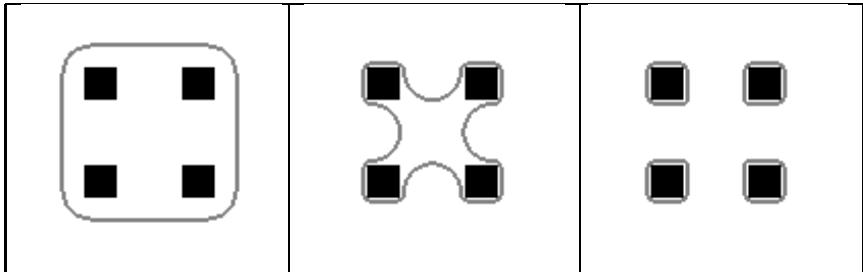


Figure 3.3. AOS-based geometric active contour model on a synthetic image (size 128×128 , $\tau = 5.0$, $k = -0.1$, $\sigma = 0.5$, $\lambda = 1$). From left to right: 10, 150, 250 iterations. From [296].

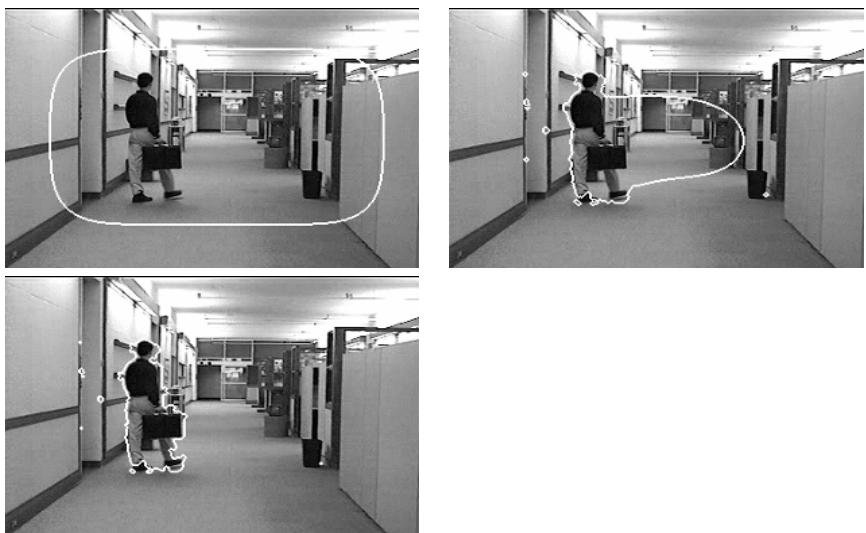


Figure 3.4. AOS-based geodesic active contour model on hall-and-monitor image (size 352×240 , $\tau = 5.0$, $k = -0.02$, $\sigma = 0.5$, $\lambda = 1$). Top left: 100 iterations. Top right: 500 iterations. Bottom left: 1000 iterations. From [296].

of the image f , the stopping function approaches 0, whereas it reaches 1 in flat image areas (low gradient magnitudes). To extract the person in the hall-and-monitor sequence we replaced the gradient term in the above equation by the results of a motion detector [297]. In each case the image u_0 was initialized to a signed distance function [501, 421, 476] from a mask that covered nearly the complete image domain. In addition, we did not employ any reinitialization procedure throughout the computations to prove the stability of our scheme. However, we should note that for certain applications it is necessary to maintain the signed distance function during curve evolution [501, 218].



Figure 3.5. AOS-based geometric active contour model on medical image (size 284×284 , $\tau = 5.0$, $k = -0.1$, $\sigma = 1$, $\lambda = 1.5$). From left to right: 50, 150, 300 iterations. From [296].

Table 3.2 summarizes the results calculated on a standard personal computer with 1.4 GHz. As expected, the AOS-based implementation reduced the number of iterations on the average by a factor of 20. Due to the coarse stopping criterion the reduction varies from 18 to 22. Furthermore, we observe that an AOS-based iteration is about twice as expensive, and in some cases three times as expensive as an explicit iteration. Combining those results, we observe that using AOS-based implementations of implicit active contour models yields a significant speedup. In our examples the speedup ranges from a factor of 5 to a factor of 9. Additionally, we applied the simple distance measure to the final contours of the AOS-based and the explicit algorithms. The distance column in Table 3.2 shows the average distance (in pixels) of the contours to the reference contour obtained by an explicit algorithm with $\tau = 0.1$. In all cases the results indicate that the accuracy of the final placements are sufficient with respect to the underlying segmentation task. We should note that the accuracy might be further improved by refining the simple stopping criterion.

3.5 Conclusions

In this chapter we have surveyed an additive operator splitting (AOS) algorithm for a class of PDEs that comprises mean curvature motion, geometric and geodesic active contour models. This algorithm uses harmonic averaging and it does not require any recomputations of the distance transformation in each iteration step.

AOS schemes have the same approximation order as their corresponding semi-implicit schemes. They come down to solving tridiagonal linear systems of equations which can be done in linear complexity with a very simple algorithm. If no balloon forces are present, then they are absolutely stable in the maximum norm. Under typical accuracy requirements one can gain a speed-up by one order of magnitude compared to the widely used explicit time discretization.

Further speed up may be possible by exploiting one of the following options:

Table 3.2. Comparison of explicit and AOS-based schemes. From [296].

<i>geometric model (explicit scheme)</i>					
image	τ	k	iterations	CPU time	distance
synthetic	0.25	-0.1	20200	49.0 s	0
hall-and-monitor	0.25	-0.1	20000	324.5 s	0
medical	0.25	-0.1	6600	126.3 s	0.01
<i>geometric model (AOS scheme)</i>					
image	τ	k	iterations	CPU time	distance
synthetic	5.0	-0.1	950	7.4 s	0.75
hall-and-monitor	5.0	-0.1	1040	54.0 s	0.87
medical	5.0	-0.1	370	25.0 s	0.48
<i>geodesic model (explicit scheme)</i>					
image	τ	k	iterations	CPU time	distance
synthetic	0.25	-0.02	10400	36.9 s	0
hall-and-monitor	0.25	-0.02	30800	634.9 s	0
medical	0.25	-0.05	12200	306.1 s	0.01
<i>geodesic model (AOS scheme)</i>					
image	τ	k	iterations	CPU time	distance
synthetic	5.0	-0.02	480	4.2 s	1
hall-and-monitor	5.0	-0.02	1390	70.2 s	1.79
medical	5.0	-0.05	640	36.8 s	1.32

- Implementing AOS schemes on a parallel system. Recent experiments with AOS-based nonlinear diffusion filtering on a PC cluster with 256 processors showed that speed up factors of 209 are possible [72].
- Embedding AOS schemes in a pyramid framework. This may yield an acceleration by one order of magnitude [567].
- The integration of narrow-band techniques is another possibility for improving the computational efficiency [216].

Finally it should be mentioned that we have focussed on the 2-D case for didactical reasons only. It is straightforward to generalize all ideas in this chapter to higher dimensions.

3.6 Appendix: The Thomas Algorithm

The semi-implicit scheme requires to solve a linear system, where the system matrix is tridiagonal and diagonally dominant. The most efficient way to achieve

this goal is the so-called *Thomas algorithm*, a Gaussian elimination algorithm for tridiagonal systems. It can be found in many textbooks on numerical analysis, e.g. [464, pp. 43–45]. However, since it builds the backbone of our AOS algorithm and since we want to keep this chapter selfcontained, we survey its algorithmic features here. The principle is as follows. Suppose we want to solve a tridiagonal linear system $Bu = d$ with an $N \times N$ matrix

$$B = \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ \gamma_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \gamma_{N-2} & \alpha_{N-1} & \beta_{N-1} \\ & & & \gamma_{N-1} & \alpha_N \end{pmatrix}. \quad (3.21)$$

Then the Thomas algorithm consists of three steps. **Step 1: LR decomposition.**

We decompose B into the product of a lower bidiagonal matrix

$$L = \begin{pmatrix} 1 & & & \\ l_1 & 1 & & \\ & \ddots & \ddots & \\ & & l_{N-1} & 1 \end{pmatrix} \quad (3.22)$$

and an upper bidiagonal matrix

$$R = \begin{pmatrix} m_1 & r_1 & & \\ & \ddots & \ddots & \\ & & m_{N-1} & r_{N-1} \\ & & & m_N \end{pmatrix} \quad (3.23)$$

Comparing the coefficients shows that $r_i = \beta_i$ for all i , and m_i and l_i can be obtained as follows:

```

 $m_1 := \alpha_1$ 
for  $i = 1, 2, \dots, N-1 :$ 
   $l_i := \gamma_i/m_i$ 
   $m_{i+1} := \alpha_{i+1} - l_i \beta_i$ 

```

Solving $LRu = d$ for u is done in two steps:

Step 2: Forward substitution.

We solve $Ly = d$ for y . This gives

```

 $y_1 := d_1$ 
for  $i = 2, 3, \dots, N :$ 
   $y_i := d_i - l_{i-1} y_{i-1}$ 

```

Step 3: Backward substitution.

We solve $Ru = y$ for u . This leads to

$u_N := y_N / m_N$
for $i = N-1, N-2, \dots, 1 :$
$u_i := (y_i - \beta_i u_{i+1}) / m_i$

This completes the Thomas algorithm. It is stable for every strictly diagonally dominant system matrix. One may also regard it as a recursive filtering: The LR decomposition determines the filter coefficients, Step 2 is a causal filter and Step 3 an anticausal one. The whole scheme is very efficient: it requires only

$$2(N-1) + (N-1) + 1 + 2(N-1) = 5N - 4 \quad (3.24)$$

multiplications/divisions, and

$$(N-1) + (N-1) + (N-1) = 3N - 3 \quad (3.25)$$

subtractions. Hence the CPU effort is *linear* in N . The same holds for the memory requirement.

4

Fast Edge Integration

Ron Kimmel

Abstract

A two-dimensional variational explanation for the Marr-Hildreth and the Haralick-Canny like edge detectors was recently presented in [280, 281]. For example, the zero crossings of the image Laplacian were shown to be optimal edge integration curves that solve a geometric problem. These are the curves whose normals best align with the image gradient field. Based on these observations, an improved active contour model was suggested, and its performances were shown to be better than classical geodesic active contours when directional information about the edge location is provided. We present a general model that incorporates the alignment as part of other driving forces of an active contour, together with the geodesic active contour model for regularization, and the minimal variance criterion suggested by Chan and Vese [100]. The minimal variance functional is related to segmentation by a threshold and to the Max-Lloyd quantization procedure. Here, we integrate all these geometric measures as part of one unified framework. Finally, we introduce unconditionally stable and simple numerical schemes for efficiently implementing the improved geometric active contour edge integration procedure.

4.1 Introduction

We start with a brief history of edge detection and relate it to modern geometry and variational principles. The most simple edge detectors try to locate points defined by local maxima of the image gradient magnitude. The Marr and Hildreth edges are a bit more sophisticated, and were defined as the zero crossing curves of a Laplacian of Gaussian (LoG) applied to the image [343, 342]. The Marr-Hildreth edge detection and integration process can be regarded as a way to determine curves in the image plane that pass through points where the gradient is high and whose normal direction best aligns with the local edge direction as predicted by the image gradient. This observation was first presented in [280]. The

importance of orientation information in a variational setting for delicate segmentation tasks was recently also thought of by Vasilevskiy and Siddiqi [543]. They used alignment with a general vector field as a segmentation criterion of complicated closed thin structures in 3D medical images. In [281] it was shown that the Haralick edge detector [228, 77], which is the main procedure in the Canny edge detector, can be interpreted as a solution of a two-dimensional variational principle that combines the alignment term with a topological homogeneity measure. We will not explore this observation here, as we have found the geodesic active contour to have somewhat better regularization performances in most cases of geometric active contour edge integration processes, which is the main topic of this chapter.

Section 4.2 introduces some of the mathematical notations we use in this chapter. In Section 4.3, we formulate the idea of geometric curve evolution for segmentation, and review various types of variational measures (geometric functionals). These functionals describe an integral quantity defined by the curve. Our goal would be to search for a curve that maximizes or minimizes these integral measures. Next, in Section 4.4 we compute the first variation of each of these functionals, and comment on how to use it in a dynamic gradient descent curve evolution process. Section 4.5 gives the level set formulation for the various curve evolution procedures. In Section 4.6, motivated by [216], we present an efficient numerical scheme that couples an alternating direction implicit multiplicative scheme, with narrow band [118, 2], and re-distancing via the fast marching method [472]. The scheme is unconditionally stable and thus allows large time steps for fast convergence.

4.2 Mathematical Notations

Consider a gray level image as a function $I : \Omega \rightarrow \mathbb{R}^+$ where $\Omega \in \mathbb{R}^2$ is the image domain. The image gradient vector field is given by $\nabla I(x, y) \equiv \{I_x, I_y\}$, were we used subscripts to denote the partial derivatives in this case, e.g., $I_x \equiv \partial I(x, y)/\partial x$. We search for a contour, $C : [0, L] \rightarrow \mathbb{R}^2$, given in a parametric form $C(s) = \{x(s), y(s)\}$, where s is an arclength parameter, and whose normal is defined by $\mathbf{n}(s) = \{-y_s(s), x_s(s)\}$. This contour somehow interacts with the given image, for example, a curve whose normal aligns with the gradient vector field, where the alignment of the two vectors can be measured by their inner product that we denote by $\langle \mathbf{n}, \nabla I \rangle$. We also use subscripts to denote full derivatives, such that the curve tangent is given by $C_s = \{x_s, y_s\} = \{dx(s)/ds, dy(s)/ds\}$. In some cases we will also use p to indicate an arbitrary (non-geometric) parameterization of the planar curve. In which case, the tangent is $C_p = C_p/|C_p|$, and the normal can be written as

$$\mathbf{n} = \frac{\{-y_p, x_p\}}{|C_p|},$$

where $|C_p| = \sqrt{x_p^2 + y_p^2}$. We have the relation between the arclength s and a general arbitrary parameter p , given by

$$ds = \sqrt{dx^2 + dy^2} = \sqrt{\left(\frac{dx(p)}{dp}\right)^2 + \left(\frac{dy(p)}{dp}\right)^2} dp = |C_p| dp.$$

Define, as usual, κ to be the curvature of the curve C , and the curvature vector $\kappa\mathbf{n} = C_{ss}$. We also define Ω_C to be the domain inside the curve C , see Figure 4.1.

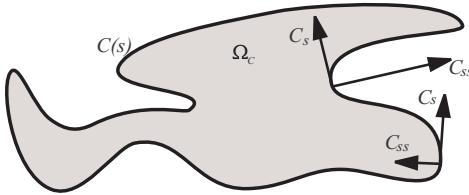


Figure 4.1. A closed curve C , with C_s the unit tangent, $\kappa\mathbf{n} = C_{ss}$ the curvature vector, and Ω_C the area inside the curve.

In this chapter we deal with two types of integral measures that are related via the Green theorem. The first is defined along the curve by the general form of

$$E(C) = \int_0^L g(C(s)) ds.$$

The second functional integrates the values of the function $f(x, y)$ inside the curve, and is usually referred to as a region based measure,

$$E(C) = \iint_{\Omega_C} f(x, y) dx dy,$$

where Ω_C is the region inside the curve C . Formally, we search for the optimal planar curve C , such that

$$C = \arg \max_C E(C).$$

4.3 Geometric Integral Measures for Active Contours

The evolution of dynamic edge integration processes and active contours started with the classical snakes [263], followed by non-variational geometric active contours [331, 81], and more recent geodesic active contours [85]. Here, we restrict our discussion to parameterization invariant (geometric) functionals, that do not depend on the internal parameterization of the curve, but rather on its geometry

and the properties of the image. From these functionals we extract the first variation, and use it as a gradient descent process, also known as geometric active contour.

4.3.1 Alignment Term

Consider the geometric functional

$$E_A(C) = \int_0^L \langle \nabla I(x(s), y(s)), \mathbf{n}(s) \rangle ds,$$

or in its ‘robust’ form

$$E_{AR}(C) = \int_0^L |\langle \nabla I(x(s), y(s)), \mathbf{n}(s) \rangle| ds,$$

where the absolute value of the inner product between the image gradient and the curve normal is our alignment measure. See Figure 4.2. The motivation is the fact

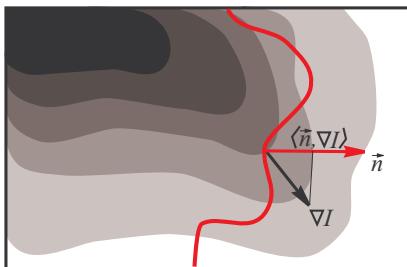


Figure 4.2. The curve C , its normal \mathbf{n} at a specific point, and the image gradient ∇I at that point. The alignment term integrates the projection of ∇I on the normal along the curve.

that in many cases, the gradient direction is a good estimator for the orientation of the edge contour. The inner product gets high values if the curve normal aligns with the image gradient direction. This measure also uses the gradient magnitude as an edge indicator. Therefore, our goal would be to find curves that maximize this geometric functional.

4.3.2 Weighted Region

In some cases we have a quantity we would like to maximize by integration inside the region Ω_C , defined by the curve C . In its most general setting, this weighted area measure is

$$E_W(C) = \iint_{\Omega_C} f(x, y) dx dy,$$

where $f(x, y)$ is any scalar function. A simple example is $f(x, y) = 1$, for which the functional $E(C)$ measures the area inside the curve C , that is, the area of the region Ω_C that we also denote by $|\Omega_C|$.

4.3.3 Minimal Variance

In [100], Chan and Vese proposed a minimal variance criterion, given by

$$\begin{aligned} E_{MV}(C, c_1, c_2) &= \frac{1}{2} \iint_{\Omega_C} (I(x, y) - c_1)^2 dx dy \\ &\quad + \frac{1}{2} \iint_{\Omega \setminus \Omega_C} (I(x, y) - c_2)^2 dx dy. \end{aligned}$$

As we will see, in the optimal case, the two constants, c_1 and c_2 , get the mean intensities in the interior (inside) and the exterior (outside) the contour C , respectively. The optimal curve would best separate the interior and exterior with respect to their relative average values. In the optimization process we look for the best separating curve, as well as for the optimal expected values c_1 and c_2 . Such optimization problems, in higher dimensions, are often encountered in color quantization and classification problems.

In order to control the smoothness of their active contour, Chan and Vese also included the arclength $\int ds$ as a regularization term. Here we propose to use the more general weighted arclength, $\int g(C(s))ds$, also known as the geodesic active contour functional [85], as a data sensitive regularization term. It can be shown to yield better results in most cases and simplifies to the regularization used by Chan and Vese for the selection of $g = 1$.

One could consider more generic region based measures like

$$\begin{aligned} E(C) &= \iint_{\Omega_C} \|T(I(x, y)) - \bar{c}_1\|_{L_p} dx dy \\ &\quad + \iint_{\Omega \setminus \Omega_C} \|T(I(x, y)) - \bar{c}_2\|_{L_p} dx dy \end{aligned}$$

where T is a general transformation of the image, the norm $\|\cdot\|_{L_p}$ can be chosen according to the problem in hand, and \bar{c}_1 and \bar{c}_2 are vectors of possible parameters. One such example is the robust measure

$$E_{RMV}(C) = \iint_{\Omega_C} |I(x, y) - c_1| dx dy + \iint_{\Omega \setminus \Omega_C} |I(x, y) - c_2| dx dy.$$

4.3.4 Geodesic Active Contour

The geodesic active contour [85] model is defined by the functional

$$E_{GAC}(C) = \int_0^L g(C(s)) ds.$$

It is an integration of an inverse edge indicator function, like $g(x, y) = 1/(1 + |\nabla I|^2)$, along the contour. The search, in this case, would be for a curve along

which the inverse edge indicator gets the smallest possible values. That is, we would like to find the curve C that minimizes this functional. The geodesic active contour was shown in [280, 281] to serve as a good regularization for other dominant terms like the minimal variance in noisy images, or the alignment term in cases we have good orientation estimation of the edge. A well studied example is $g(x, y) = 1$, for which the functional measures the total arclength of the curve.

4.4 Calculus of Variations for Geometric Measures

Given a curve integral of the general form,

$$E(C) = \oint_C L(C_p, C) dp,$$

where p is an arbitrary parameter, we compute the first variation by

$$\frac{\delta E(C)}{\delta C} = \left(\begin{array}{c} \frac{\partial}{\partial x} - \frac{d}{dp} \frac{\partial}{\partial x_p} \\ \frac{\partial}{\partial y} - \frac{d}{dp} \frac{\partial}{\partial y_p} \end{array} \right) L(C, C_p).$$

The extremals of the functional $E(C)$ can be identified by the Euler Lagrange equation $\delta E(C)/\delta C = 0$. A dynamic process known as gradient descent, that takes an arbitrary curve towards a maximum of $E(C)$, is given by the curve evolution equation

$$\frac{\partial C}{\partial t} = \frac{\delta E(C)}{\delta C},$$

where we added a virtual ‘time’ parameter t to our curve to allow its evolution into a family of planar curves $C(s, t)$. Our hope is that this evolution process would take an almost arbitrary initial curve into a desired configuration, which gives a significant extremum of our functional. In this chapter, we restrict ourselves to closed contours. When considering open contours, one should also handle the end points and add additional constraints to determine their optimal locations, as done for example in [203, 280].

LEMMA 1 *Given the vector field $\mathbf{V}(x, y) = \{u(x, y), v(x, y)\}$, we have the alignment measure,*

$$E_A(C) = \oint_C \langle \mathbf{V}, \mathbf{n} \rangle ds$$

for which the first variation is given by

$$\frac{\delta E(C)}{\delta C} = \text{div}(\mathbf{V}) \mathbf{n}.$$

Proof. We first change the integration variable from an arclength s to an arbitrary parameter p .

$$E_A(C) = \int_0^L \langle \mathbf{V}, \mathbf{n} \rangle ds = \int_0^L \langle \{u, v\}, \{-y_s, x_s\} \rangle ds$$

$$\begin{aligned}
&= \int_0^1 \left\langle \{u, v\}, \frac{\{-y_p, x_p\}}{|C_p|} \right\rangle |C_p| dp \\
&= \int_0^1 (vx_p - uy_p) dp.
\end{aligned}$$

Next, we compute the first variation for the x component,

$$\begin{aligned}
\frac{\delta E_A(C)}{\delta x} &= \left(\frac{\partial}{\partial x} - \frac{d}{dp} \frac{\partial}{\partial x_p} \right) (vx_p - uy_p) = v_x x_p - u_x y_p - \frac{d}{dp} v \\
&= v_x x_p - u_x y_p - v_x x_p - v_y y_p \\
&= -y_p(u_x + v_y) = -y_p \operatorname{div}(\mathbf{V}).
\end{aligned}$$

Similarly, for the y component we have $\frac{\delta E_A}{\delta y} = x_p \operatorname{div}(\mathbf{V})$. By freedom of parameterization, we end up with the first variation, $\frac{\delta E_A}{\delta C} = \operatorname{div}(\mathbf{V})\mathbf{n}$. ■

An important example is $\mathbf{V} = \nabla I$, for which we have as first variation

$$\frac{\delta E_A(C)}{\delta C} = \Delta I \mathbf{n},$$

where $\Delta I = I_{xx} + I_{yy}$ is the image Laplacian. The Euler Lagrange equation $\delta E_A / \delta C = 0$ gives a variational explanation for Marr-Hildreth edge detector that is defined by the zero crossings of the Laplacian, as first reported in [280].

LEMMA 2 *The robust alignment term given by*

$$E_{AR}(C) = \oint_C |\langle \mathbf{V}, \mathbf{n} \rangle| ds,$$

yields the first variation

$$\frac{\delta E_{AR}(C)}{\delta C} = \operatorname{sign}(\langle \mathbf{V}, \mathbf{n}(s) \rangle) \operatorname{div}(\mathbf{V}) \mathbf{n},$$

Proof. We start by changing to an arbitrary parameter,

$$\begin{aligned}
E_{AR}(C) &= \int_0^L |\langle \mathbf{V}, \mathbf{n} \rangle| ds = \int_0^1 \left| \left\langle \{v, u\}, \frac{\{-y_p, x_p\}}{|C_p|} \right\rangle \right| |C_p| dp \\
&= \int_0^1 \sqrt{(ux_p - vy_p)^2} dp.
\end{aligned}$$

Next, we compute the first variation for the x component,

$$\begin{aligned}
\frac{\delta E_{AR}(C)}{\delta x} &= \left(\frac{\partial}{\partial x} - \frac{d}{dp} \frac{\partial}{\partial x_p} \right) \sqrt{(vx_p - uy_p)^2} \\
&= -y_p \operatorname{sign}(vx_p - uy_p)(u_x + v_y) = -y_p \operatorname{sign}(\langle \mathbf{V}, \mathbf{n} \rangle) \operatorname{div}(\mathbf{V}).
\end{aligned}$$

Similarly, for the y component we have $\frac{\delta E_{AR}}{\delta y} = x_p \operatorname{sign}(\langle \mathbf{V}, \mathbf{n} \rangle) \operatorname{div}(\mathbf{V})$. By freedom of parameterization, we end up with the first variation,

$$\frac{\delta E_{AR}}{\delta C} = \operatorname{sign}(\langle \mathbf{V}, \mathbf{n} \rangle) \operatorname{div}(\mathbf{V}) \mathbf{n}. ■$$

An important example is $\mathbf{V} = \nabla I$, for which we have

$$\frac{\delta E_{AR}(C)}{\delta C} = \operatorname{sign}(\langle \nabla I, \mathbf{n}(s) \rangle) \Delta I \mathbf{n}.$$

LEMMA 3 *The weighted region functional*

$$E_W(C) = \iint_{\Omega_C} f(x, y) dx dy,$$

yields the first variation

$$\frac{\delta E_W(C)}{\delta C} = -f(x, y) \mathbf{n}.$$

Proof. Following [611], we define the two functions $P(x, y)$ and $Q(x, y)$, such that $P_y(x, y) = -\frac{1}{2}f(x, y)$ and $Q_x(x, y) = \frac{1}{2}f(x, y)$. We readily have that $f(x, y) = Q_x - P_y$. Next, using Green's theorem we can write

$$\begin{aligned} E(C) &= \iint_{\Omega_C} f(x, y) dx dy &= \iint_{\Omega_C} (Q_x - P_y) dx dy \\ &= \int_C (P dx + Q dy) \\ &= \int_C (P x_s + Q y_s) ds = \int_C \langle \{-Q, P\}, \mathbf{n} \rangle ds, \end{aligned}$$

for which the first variation is given by Lemma 1, for $\mathbf{V} = \{-Q, P\}$, as

$$\frac{\delta E(C)}{\delta C} = \operatorname{div}(\{-Q, P\}) \mathbf{n} = -(Q_x - P_y) \mathbf{n} = -f \mathbf{n}. \quad \blacksquare$$

This term is sometimes called the weighted area [611] term, and for f constant, its resulting variation is known as the ‘balloon’ [127] force. If we set $f = 1$, the gradient descent curve evolution process is the constant flow. It generates offset curves via $C_t = \mathbf{n}$, and its efficient implementation is closely related to Euclidean distance maps [147, 116] and fast marching methods [472].

LEMMA 4 *The geodesic active contour model is*

$$E_{GAC}(C) = \oint_C g(C(s)) ds,$$

for which the first variation is given by

$$\frac{\delta E_{GAC}(C)}{\delta C} = -(\kappa g - \langle \nabla g, \mathbf{n} \rangle) \mathbf{n}.$$

Proof.

$$\begin{aligned} E_{GAC}(C) &= \int_0^L g(C(s)) ds &= \int_0^1 g(C(p)) |C_p| dp \\ &= \int_0^1 g(C(p)) \sqrt{x_p^2 + y_p^2} dp. \end{aligned}$$

Next, we compute the first variation for the x component,

$$\frac{\delta E_{GAC}(C)}{\delta x} = \left(\frac{\partial}{\partial x} - \frac{d}{dp} \frac{\partial}{\partial x_p} \right) \left(g(x(p), y(p)) \sqrt{x_p^2 + y_p^2} \right)$$

$$\begin{aligned}
&= g_x |C_p| - \frac{d}{dp} g \frac{x_p}{\sqrt{x_p^2 + y_p^2}} \\
&= g_x |C_p| - (g_x x_p + g_y y_p) \frac{x_p}{|C_p|} \\
&\quad - g \frac{x_{pp}|C_p| - x_p(x_p x_{pp} + y_p y_{pp})/|C_p|}{|C_p|^2} \\
&= y_p (\kappa g - \langle \nabla g, \mathbf{n} \rangle).
\end{aligned}$$

where we used the curvature defined by

$$\kappa = \frac{x_{pp}y_p - y_{pp}x_p}{|C_p|^3}.$$

Similarly, for the y component we have $\frac{\delta E_{GAC}}{\delta y} = -x_p(\kappa g - \langle \nabla g, \mathbf{n} \rangle)$. By freedom of parameterization we end up with the above first variation. ■

We will use this term mainly for regularization. If we set $g = 1$, the gradient descent curve evolution result given by $C_t = -\delta E_{GAC}(C)/\delta C$ is the well known curvature flow, $C_t = \kappa \mathbf{n}$ or equivalently $C_t = C_{ss}$, also known as the geometric heat equation.

LEMMA 5 *The Chan-Vese minimal variance criterion [100] is given by*

$$E_{MV}(C, c_1, c_2) = \frac{1}{2} \iint_{\Omega_C} (I - c_1)^2 dx dy + \frac{1}{2} \iint_{\Omega \setminus \Omega_C} (I - c_2)^2 dx dy,$$

for which the first variation is

$$\begin{aligned}
\frac{\delta E_{MV}}{\delta C} &= (c_2 - c_1) \left(I - \frac{c_1 + c_2}{2} \right) \mathbf{n} \\
\frac{\delta E_{MV}}{\delta c_1} &= \iint_{\Omega_C} I dx dy - c_1 \iint_{\Omega_C} dx dy \\
\frac{\delta E_{MV}}{\delta c_2} &= \iint_{\Omega \setminus \Omega_C} I dx dy - c_2 \iint_{\Omega \setminus \Omega_C} dx dy.
\end{aligned}$$

Proof. Using Lemma 3, we have the first variation given by

$$\begin{aligned}
\frac{\delta E_{MV}}{\delta C} &= \frac{1}{2} ((I - c_1)^2 - (I - c_2)^2) \mathbf{n} \\
&= \frac{1}{2} (I^2 - 2Ic_1 + c_1^2 - I^2 + 2Ic_2 - c_2^2) \mathbf{n} \\
&= \left((c_2 - c_1)I - \frac{(c_1 + c_2)(c_2 - c_1)}{2} \right) \mathbf{n} \\
&= (c_2 - c_1) \left(I - \frac{c_1 + c_2}{2} \right) \mathbf{n}.
\end{aligned}$$

The optimal c_1 and c_2 , extracted from $\delta E_{MV}/\delta c_1 = 0$ and $\delta E_{MV}/\delta c_2 = 0$, are the average intensities of the image inside and outside the contour, respectively. ■

One could recognize the variational interpretation of segmentation by the threshold $(c_1 + c_2)/2$ given by the Euler Lagrange equation $\delta E_{MV}/\delta C = 0$.

Finally, we treat the robust minimal deviation measure E_{RMV} .

LEMMA 6 *The robust minimal total deviation criterion is given by*

$$E_{RMV}(C, c_1, c_2) = \iint_{\Omega_C} |I - c_1| dx dy + \iint_{\Omega \setminus \Omega_C} |I - c_2| dx dy,$$

for which the first variation is

$$\begin{aligned} \frac{\delta E_{RMV}}{\delta C} &= (|I - c_1| - |I - c_2|) \mathbf{n} \\ \frac{\delta E_{RMV}}{\delta c_1} &= \iint_{\Omega_C} sign(I - c_1) dx dy \\ \frac{\delta E_{RMV}}{\delta c_2} &= \iint_{\Omega \setminus \Omega_C} sign(I - c_2) dx dy \end{aligned}$$

Proof. Using Lemma 3, we have the first variation $\frac{\delta E_{RMV}(C)}{\delta C}$.

The optimal c_1 and c_2 , extracted from $\delta E_{RMV}/\delta c_1 = 0$ and $\delta E_{RMV}/\delta c_2 = 0$, are the median intensities of the image inside and outside the contour, respectively:

$$\begin{aligned} \frac{\delta E_{RMV}}{\delta c_1} &= \frac{d}{dc_1} \iint_{\Omega_C} \sqrt{(I - c_1)^2} dx dy \\ &= - \iint_{\Omega_C} \frac{I - c_1}{|I - c_1|} dx dy \\ &= - \iint_{\Omega_C} sign(I - c_1) dx dy. \end{aligned}$$

We have that $\iint_{\Omega_C} sign(I - c_1) dx dy = 0$ for the selection of c_1 as the value of $I(x, y)$ in Ω_C that splits its area into two equal parts. From obvious reasons we refer to this value as the median of I in Ω_C , or formally,

$$\begin{aligned} c_1 &= \text{median}_{\Omega_C} I(x, y) \\ c_2 &= \text{median}_{\Omega \setminus \Omega_C} I(x, y). \end{aligned}$$

■

The computation of c_1 and c_2 can be efficiently implemented via the intensity histograms in the interior or the exterior of the contour. One rough discrete approximation is the median of the pixels inside or outside the contour.

The robust minimal deviation term should be preferred when the penalty for isolated pixels with wrong affiliation is insignificant. The minimal variance measure penalizes large deviations in a quadratic fashion and would tend to over-segment such events or require large regularization that could over-smooth the desired boundaries.

4.5 Gradient Descent in Level Set Formulation

We embed a closed curve in a higher dimensional $\phi(x, y)$ function, which implicitly represents the curve C as a zero set, i.e., $C = \{x, y\} : \phi(x, y) = 0\}$. This way, the well known Osher-Sethian [401] level-set method can be employed to implement the curve propagation toward its optimal location. Figure 4.3 presents a planar curve, and two implicit representations for this curve, displayed in two different ways.



Figure 4.3. Left: A given planar curve. Middle: An implicit representation of the curve as a threshold set of a gray level image. Right: An implicit representation of the curve as a level set of a smooth function displayed by its level set contours.

Given the curve evolution equation $C_t = \gamma \mathbf{n}$, its implicit level set evolution equation reads

$$\phi_t = \gamma |\nabla \phi|.$$

The equivalence of these two evolutions can be easily verified using the chain rule and the relation $\mathbf{n} = \nabla \phi / |\nabla \phi|$,

$$\phi_t = \phi_x x_t + \phi_y y_t = \langle \nabla \phi, C_t \rangle = \langle \nabla \phi, \gamma \mathbf{n} \rangle = \gamma \left\langle \nabla \phi, \frac{\nabla \phi}{|\nabla \phi|} \right\rangle = \gamma |\nabla \phi|.$$

Those familiar with the optical flow problem in image analysis could easily recognize this derivation. There is an interesting relation between the classical optical flow problem and the level set method. Level set formulation for the evolution of a family of embedded curves can be interpreted as a dynamic image synthesis process. On the other hand, optical flow in image analysis is a search for the motion of features in the image. These two inverse problems share the same fundamental equation. Computing a vector field that represents the flow of the gray level sets from a given sequence of images is known as the ‘normal flow’ computation. Next, at a higher level of image understanding, the motion field of objects in an image is known as the ‘optical flow’. On the other hand in the level set formulation, the goal is to compute the dynamics of an arbitrary image, in which one level set represents a specific curve, from a given motion vector field

of that specific level set. The image in this case is an implicit representation of its level sets, while the vector field itself could be extracted from either the geometric properties of the level sets, or from another image or external data.

As a first example, consider the explicit curve evolution as a gradient descent flow for the robust alignment term, given by

$$C_t = \text{sign}(\langle \nabla I, \mathbf{n} \rangle) \Delta I \mathbf{n}.$$

The corresponding level set evolution is

$$\phi_t = \text{sign}(\langle \nabla I, \nabla \phi \rangle) \Delta I |\nabla \phi|.$$

We can now add to our model the geodesic active contour term, the threshold term, or its dynamic expectation version defined by the minimal variance criterion. The optimization problem takes the form of

$$\arg \max_{C, c_1, c_2} E(C, c_1, c_2),$$

for the functional

$$\begin{aligned} E(C, c_1, c_2) &= E_{AR}(C, c_1, c_2) - \alpha E_{GAC}(C) - \beta E_{MV}(C) \\ &= \oint_C |\langle \nabla I, \mathbf{n} \rangle| ds - \alpha \oint_C g(C(s)) ds \\ &\quad - \beta \frac{1}{2} \left(\iint_{\Omega_C} (I - c_1)^2 dx dy + \iint_{\Omega \setminus \Omega_C} (I - c_2)^2 dx dy \right), \end{aligned}$$

where α and β are constants, and α is small so that the geodesic active contour term is used mainly for regularization. The first variation as a gradient descent process, is

$$\begin{aligned} C_t &= [\text{sign}(\langle \mathbf{n}, \nabla I \rangle) \Delta I + \alpha (g(x, y) \kappa - \langle \nabla g, \mathbf{n} \rangle) \\ &\quad + \beta(c_2 - c_1)(I - (c_1 + c_2)/2)] \mathbf{n}. \\ c_1 &= \frac{1}{|\Omega_C|} \iint_{\Omega_C} I(x, y) dx dy \\ c_2 &= \frac{1}{|\Omega \setminus \Omega_C|} \iint_{\Omega \setminus \Omega_C} I(x, y) dx dy, \end{aligned}$$

where $|\Omega_C|$ denotes the area of the regions Ω_C . One could recognize the relation to Max-Lloyd quantization method, as the simplest implementation for this system is a sequential process that involves a change of the segmentation set followed by an update of the mean representing each set. The difference is the additional penalties and resulting forces we use for the smoothness and alignment of the boundary contours.

The level set formulation of the curve evolution equation is

$$\begin{aligned} \phi_t &= \left[\text{sign}(\langle \nabla \phi, \nabla I \rangle) \Delta I + \alpha \text{div} \left(g(x, y) \frac{\nabla \phi}{|\nabla \phi|} \right) \right. \\ &\quad \left. + \beta(c_2 - c_1) \left(I - \frac{c_1 + c_2}{2} \right) \right] |\nabla \phi|. \end{aligned}$$

Efficient solutions for this equation can use either AOS [324, 325, 570] or ADI methods, coupled with a narrow band approach [117, 2], as first introduced in [216] for the geodesic active contour. In the next section we use a simple first order implicit alternating direction multiplicative scheme.

The following table summarizes the functionals, the resulting first variation for each functional, and the level set formulations for these terms.

Measure	$E(C)$	$\delta E/\delta C$	level set form
Weighted Area	$\iint_{\Omega_C} f(x, y) dx dy$	$-f(x, y)\mathbf{n}$	$-f(x, y) \nabla\phi $
Minimal Variance	$\iint_{\Omega_C} (I - c_1)^2 dx dy + \iint_{\Omega \setminus \Omega_C} (I - c_2)^2 dx dy$	$(c_2 - c_1) \times (I - (c_1 + c_2)/2)\mathbf{n}$	$(c_2 - c_1) \times (I - (c_1 + c_2)/2) \nabla\phi $
GAC	$\oint_C g(C(s)) ds$	$(\langle \nabla g, \mathbf{n} \rangle - \kappa g)\mathbf{n}$	$-\operatorname{div}\left(g \frac{\nabla \phi}{ \nabla \phi }\right) \nabla \phi $
Alignment	$\oint \langle \nabla I, \mathbf{n} \rangle ds$	$\operatorname{sign}(\langle \nabla I, \mathbf{n} \rangle) \Delta I \mathbf{n}$	$\operatorname{sign}(\langle \nabla I, \nabla \phi \rangle) \Delta I \nabla \phi $

4.6 Efficient Numerical Schemes

In [570] Weickert et al. used an unconditionally stable, and thus efficient, numerical scheme for non-linear isotropic image diffusion known as *additive operator splitting* (AOS), that was first introduced in [324, 325], and has some nice symmetry and parallel properties. Goldenberg et al. [216] coupled the AOS with Sethian's fast marching on regular grids [471] (see [76, 116] for related approaches), with multi-resolution [412], and with the narrow band approach [117, 2], as part of their fast geodesic active contour model for segmentation and object tracking in video sequences. Here, motivated by these results, we extend the efficient numerical methods for the geodesic active contour [85] presented in [216], for the variational edge integration models introduced in [280, 281], and the minimal variance [100]. We review efficient numerical schemes and modify them in order to solve the level set formulation of edge integration and object segmentation problem in image analysis.

Let us analyze the following level set formulation,

$$\begin{aligned}\phi_t &= \left(\alpha \operatorname{div} \left(g(x, y) \frac{\nabla \phi}{|\nabla \phi|} \right) + \eta(\phi, \nabla I) \right) |\nabla \phi|, \\ \eta(\phi, \nabla I) &= \operatorname{sign}(\langle \nabla I, \nabla \phi \rangle) \Delta I + \beta(c_2 - c_1) \left(I - \frac{c_1 + c_2}{2} \right).\end{aligned}$$

If we assume $\phi(x, y; t)$ to be a distance function of its zero set, then, we could approximate the short time evolution of the above equation by setting $|\nabla \phi| = 1$. The short time evolution for a distance function ϕ is thereby

$$\begin{aligned}\phi_t &= \alpha \operatorname{div}(g(x, y) \nabla \phi) + \eta(\phi, \nabla I) \\ &= \alpha \frac{\partial}{\partial x} \left(g(x, y) \frac{\partial}{\partial x} \phi \right) + \alpha \frac{\partial}{\partial y} \left(g(x, y) \frac{\partial}{\partial y} \phi \right) + \eta(\phi, \nabla I).\end{aligned}$$

Note, that when using a narrow band around the zero set to reduce computational complexity on sequential computers, the distance from the zero set needs to be re-

computed in order to determine the width of the band at each iteration. Thus, there is no additional computational cost in simplifying the model while considering a distance map rather than an arbitrary smooth function. We thereby enjoy both the efficiency of the simplified almost linear model, and the low computational cost of the narrow band.

Denote the operators

$$\begin{aligned} A_1 &= \frac{\partial}{\partial x} g(x, y) \frac{\partial}{\partial x} \\ A_2 &= \frac{\partial}{\partial y} g(x, y) \frac{\partial}{\partial y}. \end{aligned}$$

Using these notations we can write the evolution equation as

$$\phi_t = \alpha (A_1 + A_2) \phi + \eta(\phi, \nabla I).$$

Next, we approximate the time derivative using forward approximation $\phi_t \approx \frac{\phi^{n+1} - \phi^n}{\tau}$, that yields the explicit scheme

$$\begin{aligned} \phi^{n+1} &= \phi^n + \tau (\alpha (A_1 + A_2) \phi^n + \tau \eta(\phi^n, \nabla I)) \\ &= (\mathcal{I} + \tau \alpha (A_1 + A_2)) \phi^n + \tau \eta(\phi^n, \nabla I), \end{aligned}$$

where, after sampling the x, y plane, \mathcal{I} is the identity matrix and I is our input image. The operators A_l become matrix operators, and ϕ^n is represented as a vector in either column or row stack, depending on the acting operator. This way, the operators A_l are tri-diagonal, which makes its inverse computation fairly simple using Thomas algorithm. Note that in the explicit scheme there is no need to invert any operator, yet the numerical time step is bounded for stability.

Let us first follow [570], and use a simple discretization for the A_l , $l \in \{1, 2\}$ operators. For a given row, let

$$\frac{\partial}{\partial x} \left(g(x) \frac{\partial}{\partial x} \phi \right) \approx \sum_{j \in \mathcal{N}(i)} \frac{g_j + g_i}{2h^2} (\phi_j - \phi_i),$$

where $\mathcal{N}(i)$ is the set $\{i - 1, i + 1\}$, representing the two horizontal neighbors of pixel i , and h is the space between neighboring pixels. The elements of A_1 are thereby given by

$$a_{ij} = \begin{cases} \frac{g_i + g_j}{2h^2} & j \in \mathcal{N}(i) \\ -\sum_{k \in \mathcal{N}(i)} \frac{g_i + g_k}{2h^2} & j = i \\ 0 & \text{else.} \end{cases}$$

In order to construct an unconditionally stable scheme we use a locally one-dimensional (LOD) scheme adopted for our problem. We use the following scheme

$$\phi^{n+1} = \prod_{l=1}^2 (\mathcal{I} - \tau \alpha A_l)^{-1} (\phi^n + \tau \eta(\phi^n, \nabla I)).$$

In one-dimension it is also known as fully implicit, or backward Euler scheme. It is a first order implicit numerical approximation, since we have that

$$\begin{aligned} (\mathcal{I} - \tau A_1)^{-1}(\mathcal{I} - \tau A_2)^{-1}(\phi + \tau\eta) &= (\mathcal{I} - \tau A_1 - \tau A_2 + O(\tau^2))^{-1}(\phi + \tau\eta) \\ &= \phi + \tau(A_1 + A_2)\phi + \tau\eta + O(\tau^2), \end{aligned}$$

where we applied the Taylor series expansion in the second equality. First order accuracy is sufficient, as our goal is the steady state solution. We also included the weighted region-balloon, minimal variance, and the alignment terms within this implicit scheme, while keeping the first order accuracy and stability properties of the method. The operators $\mathcal{I} - \tau A_l$ are positive definite symmetric operators, which make the implicit process unconditionally stable, using either the above multiplicative or the additive (AOS) schemes. If we have an indication that the contour is getting closer to its final destination, we could switch to an explicit scheme for the final refinement steps with a small time step. In this case, the time step should be within the stability limits of the explicit scheme. In our implementation we also use a multi-resolution pyramidal approach, where the coarse grid still captures the details of the objects we would like to detect.

4.7 Examples



Figure 4.4. Test images left to right: numbers with a small amount of noise, numbers with a large amount of noise, and a number with non-uniform illumination.

Figure 4.4 presents a set of simple input images in which the goal is to segment the numbers from the background. In all examples we used the image frame as initial condition for the active contour model, and applied a multi-resolution coarse to fine procedure, as in [412, 216], to speed up the segmentation process.

Figure 4.5 shows the segmentation result of the active contour model in the first case. In this example, the alignment and minimal variance terms were tuned to play the dominant role in the segmentation. The right frame, here and in the rest of the examples, shows the final contour in black painted on the original image in which the dynamic range is mapped into a sub-interval of the original one.

In the next example, shown in Fig. 4.6, high noise and uniform illumination calls for minimal variance coupled with regularization of the contour. The alignment term does not contribute much in such a high noise.

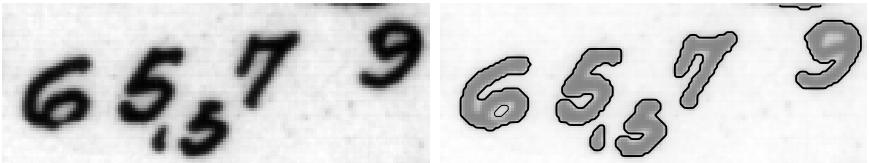


Figure 4.5. The simplest case in which alignment and minimal variance played the dominant factors in finding the exact location of the edges.

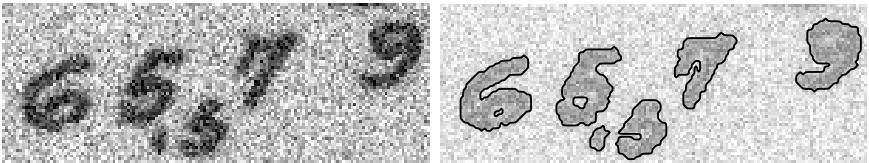


Figure 4.6. For noisy images the alignment term is turned off, while the minimal variance and regularization by the geodesic active contour are the important factors.

The last example demonstrates the effect of a non-uniform illumination. In this case, the minimal variance would be of little help, while the alignment term leads the active contour to the accurate edge locations, see Fig. 4.7.

In the appendix we present an oversimplified matlab code for the whole process. Note that in order to keep the program simple the multi-resolution is not presented, and the redistancing procedure was only roughly approximated, see [215] for more details.

4.8 Conclusions

We explored fundamental geometric variational measures for edge integration and uniform region segmentation. Next, we computed the resulting first variation for these integral measures. The level set formulation for the resulting curve evolution equations was then implemented by efficient numerical schemes.

We are still far from the end of the road of implementing low level vision operators. Good numerical schemes for so-called ‘solved’ problems would probably change the way we process and analyze images in the future. Simple operations we take for granted, like edge detection and shape reconstruction, should be revisited and refined for the best possible solution. The exploration of the basic image analysis tools would improve our understanding of these processes and enable faster progress of feature extraction, learning, and classification procedures. Our philosophy of geometric variational interpretation for fundamental low level image analysis operators seem to be one promising direction for improving existing tools and designing new ones.

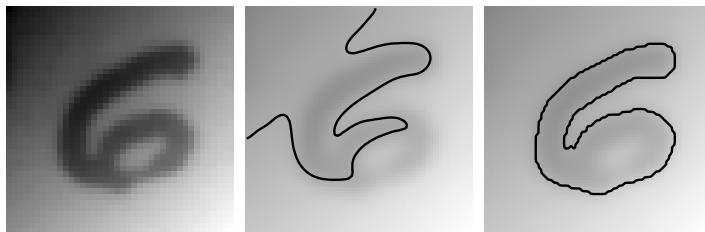


Figure 4.7. For non-uniform illumination without noise. The minimal variance term is turned off, while the alignment term plays the leading role in a good segmentation of the number from the background. Left: input image. Middle: Segmentation by threshold yields a meaningless result. Right: Active contour segmentation with a dominant alignment term.

Acknowledgments

Stimulating discussions with Alfred Bruckstein, Moshe Israeli, Irad Yavne, Roman Goldenberg, Yossi Rubner, and Alexander Brook at the Technion are gratefully acknowledged.

Appendix

```

% Locally One-Dimensional (LOD) implicit scheme for closed geometric
% active contour model+minimal variation+GAC+robust alignment.
% I = Input image matrix, Phi = Initial contour matrix (implicit form).
% Balloon = Weighted area factor (scalar)
% Align = Alignment force factor (scalar)
% Max_Lloyd = The Max-Lloyd/Chan-Vese threshold factor (scalar)
% k = The time step (tau scalar)
% iter = Maximal number of iterations (scalar)
function Phi=LOD_Active_Contour(I,Phi,Balloon,Align,Max_Lloyd,k,iter)
D2I=Dxx(I)+Dyy(I); P=Dx(I); Q=Dy(I);
g = 1./sqrt(1+P.^2+Q.^2); % example for computing g(x,y)
delta = 2; count=1;
while and(delta > 0.0001,count<iter) %check if converged
    Phi_old = Phi;
    threshold = LloydMax(Phi,I); % Max-Lloyd/Chan-Vese term
    alignment = -sign(P.*Dx(Phi)+Q.*Dy(Phi)).*D2I; % Laplacian term
    Phi = Phi+k*(Balloon*g+Align*alignment+Max_Lloyd*threshold);
    for i=1:2, % i=1 => (I-tau*Ax) i=2 => (I-tau*Ay)
        Phi = Implicit(g(:,k),Phi); % (1/(I-tau*A))Phi
        Phi = Phi'; g = g'; % Transpose for Ay
    end % for i
end

```

```

Phi = redistance(Phi); % Use fast marching for re-distancing
delta = sum(sum((Phi_old-Phi).A2)) % Compute L2 norm
count = count+1;
imshow(I,[]); hold on; contour(Phi,[0 0],r'); hold off; drawnow;
end % while and function
%-----
% Compute (1/(I- k*Al))Phi using Thomas algorithm
% k=time step, g=g(x,y) in column stack form
function u = Implicit(g,k,Phi)
gm = -k.* (g + g([end 1:end-1]))/2; % lower diag
gc = 1-k.*(-2*g - g([end 1:end-1]) -g([2:end 1]))/2; % main diag
gp = -k.* (g + g([2:end 1]))/2; % upper diag
u = Thomas(gc, gp(1:end-1), gm(2:end), Phi(:));
u = reshape(u,size(Phi));
%-----
% Compute the Lloyd-Max/Chan-Vese thresholding
function force = LloydMax(Phi,I)
mask_in = (Phi<0); % inside the contour
mask_out = 1-mask_in; % rest of the domain
I_in = sum(sum(mask_in.*I))/sum(mask_in(:)); % mean value
I_out = sum(sum(mask_out.*I))/sum(mask_out(:));
force = (I_out-I_in).* (I-(I_in+I_out)/2);
%-----
% 'Roughly' correct Phi to be a distance map of its zero set
function u=redistance(Phi);
u = (sign(Phi)+1)*999999; % set to infinity all positive
for i=1:2,
l2 = 2;
if i>1 u=(1-sign(Phi))*999999; end % set to infinity all negative
while l2 > 1,
v = Update(u,1);
l2 = sum(sum((u-v).A2));
u = v;
end % while
if i>1 u=up-u; else up=u; end %if
end % for
%-----
% Solve |grad u|=F(x,y) parallel version of the FMM
function res = Update(u,F)
mx = min(u([2:end end],:), u([1 1:end-1],:));
my = min(u(:,[2:end end]), u(:,[1 1:end-1]));
delm = (mx -my);
mask = (abs(delm) < F);
res= min(mask.* (mx+my+sqrt(2.*F.A2- delm .A2))./2 + ...
(1-mask).* (min(mx,my)+F) , u);

```

```

%-----
function f = Dmx(P)
f = P - P([1 1:end-1],:);
%-----
function f = Dpx(P)
f = P([2:end end],:) - P;
%-----
function f = Dx(P)
f = (Dpx(P)+Dmx(P))/2;
%-----
function f = Dy(P)
f = (Dx(P'))';
%-----
function f = Dxx(P)
f = Dpx(P)-Dmx(P);
%-----
function f = Dyy(P)
f = (Dxx(P'))';
%-----
% Thomas Algorithm for trilinear diagonally dominant system:
% B u = d (solve for u); B is given by its 3 diagonals:
% alpha(1:N)=main diagonal, beta(1:N-1)=upper diagonal,
% gamma(1:N-1) = lower diagonal, (Compile first!)
function u = Thomas(alpha,beta,gamma,d)
N = length(alpha); r = beta;
l = zeros(N-1,1); u = zeros(N,1);
m = u; % zero
m(1) = alpha(1);
for i=1:N-1,
    l(i) = gamma(i)/m(i);
    m(i+1) = alpha(i+1)-l(i)*beta(i);
end % for
y = u; % zero
y(1) = d(1);
for i = 2:N,
    y(i) = d(i)-l(i-1)*y(i-1);
end % for
u(N) = y(N)/m(N);
for i = N-1:-1:1,
    u(i) = (y(i)-beta(i)*u(i+1))/m(i);
end % for

```

5

Variational Snake Theory

Agnès Desolneux, Lionel Moisan and Jean-Michel Morel

Abstract

In this chapter, we review briefly the theory of edge detection and its non local versions, the "snakes". We support the idea that the class of snake energies proposed recently by Kimmel and Bruckstein, namely the "average contrast across the snake" is optimal. We reduce this class to a single model by proving a particular form of their contrast function to be optimal. This form is as close as possible to a threshold function of the image contrast across the snake. Eventually, we show by arguments and experiments that the resulting snakes simply coincide with the well contrasted level lines of the image. For a sake of completeness, we give all formal computations needed for deriving the main models, their evolution equation and steady state equation. If, as we sustain, the snakes can be replaced in most practical cases by optimal level lines, the topological changes are simply handled by using their nested inclusion tree.

5.1 Introduction

It seems to be as useful as ever to discuss the way salient boundaries, or edges, can be computed in an image. In the past 30 years, many methods have been proposed and none has imposed itself as a standard. All have in common a view of "edginess" according to which an edge is a curve in the image across which the image is contrasted. Hildreth and Marr [343] proposed to define the boundaries in a grey level image $u(x, y)$ as lines of points where the Laplacian

$$\Delta u(x, y) = \frac{\partial^2 u}{\partial x^2}(x, y) + \frac{\partial^2 u}{\partial y^2}(x, y)$$

crosses zero. This was based on the remark, true for a one-dimensional function $u(x)$, that points with the highest gradient satisfy $u''(x) = 0$. Haralick [228] improved this a lot by simply proposing as "edge points" the points where the

magnitude of the gradient $|Du|$ attains a maximal value along gradient lines. An easy computation shows that such points (x, y) satisfy $D^2u(Du, Du)(x, y) = 0$, where we take as a notation

$$D^2u(x, y) = \begin{pmatrix} \frac{\partial^2 u}{\partial x^2} & \frac{\partial^2 u}{\partial x \partial y} \\ \frac{\partial^2 u}{\partial y \partial x} & \frac{\partial^2 u}{\partial y^2} \end{pmatrix}, \quad Du = \begin{pmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{pmatrix}.$$

(If $A = \begin{pmatrix} a & b \\ b & c \end{pmatrix}$ and $\xi = \begin{pmatrix} v \\ w \end{pmatrix}$, we set $A(\xi, \xi) = av^2 + 2bvw + cw^2$.)

Thus, in the following, we shall talk about Hildreth-Marr's and Haralick's edge points respectively. The Haralick's edge points computation was proved by Canny [77] to be optimal for "step edges" under a Gaussian noise assumption. Now, image analysis aims at global structures. Edge points are from that point of view a poor basis for the further building up of geometric structures in image perception. This is why edge detection methods have evolved towards "boundary detection" methods, i.e. methods which directly deliver curves in the image along which the gradient is, in some sense, highest.

There is agreement about the following criteria : if the image contrast along a curve is high, so much the better. If in addition such a contrasted curve is smooth, its "edginess" is increased. The requirement that the curve be smooth must be tempered, however, since simple objects can be ragged or have corners.

Let us give some notations. A smooth curve in the image will be denoted by $\gamma(s)$, a one to one map from some real interval into the image plane. Unless otherwise specified, s is an Euclidean parameterization, which means by definition that

$$|\gamma'(s)| = 1.$$

We shall denote by $L(\gamma)$ the length of γ . When $v = (x, y)$ is a vector, we set $v^\perp = (-y, x)$, a vector orthogonal to v . The unit tangent vector to the curve is $\gamma'(s)$ and we set

$$\mathbf{n}(s) = \gamma'(s)^\perp,$$

a vector normal to the curve γ . We finally consider

$$\gamma''(s) = Curv(\gamma(s)), \tag{5.1}$$

which is a vector normal to the curve whose magnitude is proportional to its curvature. If $Curv(\gamma(s))$ is bounded, the curve behaves locally as a circle with radius $1/|Curv(\gamma(s))|$ and is therefore smooth. Thus, if some privilege has to be given to smooth curves, one is led to define an "edge" as a curve $\gamma(s)$ such that

$$\int_{[0, L(\gamma)]} g(|Du(\gamma(s))|)ds$$

is minimal (g being any decreasing function), and such that

$$\int_{[0, L(\gamma)]} (1 + |Curv(\gamma(s))|)ds$$

is minimal. Both requirements can be combined in a single energy functional : the Kass-Witkin-Terzopoulos “snake” original model [262], where the minimum of

$$\int_0^{L(\gamma)} g(|Du(\gamma(s))|)ds + C \int_0^{L(\gamma)} (a + |Curv(\gamma(s))|)ds \quad (5.2)$$

is sought for. According to this formulation, a snake must be as smooth as possible and have a contrast as strong as possible.

This model has been generally abandoned for the derived “geodesic snakes” [85]. This model proposed a geometric form of the snakes functional, where the minimum of

$$\int_0^{L(\gamma)} g(|Du(\gamma(s))|)ds \quad (5.3)$$

is sought for. Since $g > 0$, this last functional looks for a compromise between length and contrast and yields a well contrasted curve with bounded curvature. Indeed, if $g(0) > 0$, which is usual, the first term also gives a control on the length of the snake, and therefore actually forces the curvature to be bounded. Notice also that the minimization process therefore tends to decrease the length and forces the snake to shrink, which is not exactly what is wished !

This may explain why Fua and Leclerc [203] proposed to minimize, for edge detection, the average functional

$$\frac{1}{L(\gamma)} \int_0^{L(\gamma)} g(|Du(\gamma(s))|)ds. \quad (5.4)$$

Here again, g is decreasing. Minimizing this functional amounts to finding a curve in the image with, so to say, maximal average contrast. One of the main advances in the formulation adopted in [203] and [85] is the reduction of the number of model parameters or functions to a single one : the contrast function g . The Fua-Leclerc model is in our opinion better since it focuses on contrast only and is therefore apparently no more a hybrid combination of contrast and smoothness requirements.

Now, all above mentioned models were in back with respect to the first edge detectors, defined in the beginning of the seventies. Indeed, the Montanari [378] and Martelli [344] original boundary detection models were more accurate in one aspect : instead of $|Du(\gamma(s))|$, they used as contrast indicator a discrete version of

$$u_n(s) = \frac{\partial u}{\partial n}(s) = Du(\gamma(s)) \cdot \mathbf{n}(s), \quad (5.5)$$

that is, the *contrast of the image across the curve*. At a point $\gamma(s)$, we can see that $u_n(s)$ is larger if the magnitude of the gradient, $|Du(\gamma(s))|$, is larger, but also if this the gradient is as much as possible normal to the curve. Clearly, the above Kass, Witkin, Terzopoulos, the Fua-Leclerc and the Caselles-Kimmel-Sapiro contrast measures are worse at that point : they only take into account the the magnitude of the gradient, independently of its angle with the curve normal.

As a general remark on all variational snakes, we must also notice that if g is nonnegative, which is usual, the best snake in the energetical sense is reduced to a single point at which the maximum magnitude of the gradient of the image is attained. Thus, in all snake models, local minima of the snake energy should be sought for, the global ones being irrelevant. Such local minima usually depend upon :

- the initial position of the snake,
- the variance of the usual preliminary smoothing of the gradient,
- the form of the contrast function g .

Recently, Kimmel and Bruckstein [280] [281] made several important advances on the formulation of edge detectors and the snakes method, which we shall discuss, and push a bit further in this chapter. We can summarize the Kimmel-Bruckstein results as follows.

- Maximizers of the contrast along γ , $\int_0^{L(\gamma)} u_n(s)ds$ satisfy $\Delta u(\gamma(s)) = 0$, provided u_n does not change sign along the curve. This yields a variational interpretation of Hildreth-Marr edges.
- Active contours can more generally be performed by maximizing a nonlinear function of contrast, $E(\gamma) = \int_0^{L(\gamma)} g(u_n(s))ds$, where g is even and increasing, a good example being $g(t) = |t|$. This is basically the energy (5.3) but where the isotropic contrast indicator $|Du(\gamma(s))|$ is replaced by the better term $u_n(s) = Du(\gamma(s)).\mathbf{n}(s)$ used in Montanari-Martelli. The case $g(t) = t^2$ was actually considered earlier, in the founding Mumford-Shah paper [388]. More precisely, Mumford and Shah considered the minimization of

$$E_\infty = \int_0^{L(\gamma)} \nu - u_n(s)^2 ds,$$

but they discovered that this functional has no minimizers because of the folding problem, which we shall address further on.

Also, Kimmel and Bruckstein consider maximizing the *average contrast*, namely

$$E(\gamma) = \frac{1}{L(\gamma)} \int_0^{L(\gamma)} g(u_n(s))ds, \quad (5.6)$$

where g is some increasing function. We then get an improved version of the Fua-Leclerc functional.

All this looks good and fine : the energy functional is simpler, it does not enforce a priori smoothness constraints and it measures the real contrast.

- The evolution equation towards an optimum boundary can be written in much the same way as in the geodesic snake method.

- In [281], it is also shown that the Haralick operator can receive a global variational snake interpretation. In order to give this interpretation, let us define a (non local) orientation at each point x_0 of the image plane in the following way : we consider the level line $\{x, u(x) = u(x_0)\}$ passing by x_0 and we set $O(x) = +1$ if the level line surrounds $x + \varepsilon Du(x)$ for $\varepsilon > 0$ small enough, $O(x) = -1$ otherwise (all level lines are supposed to be closed and compact ; this can be ensured (e.g.) by taking $u = -\infty$ outside the image domain). Call $\Omega(\gamma)$ the domain surrounded by a Jordan curve γ and consider the functional

$$E(\gamma) = \int_0^{L(\gamma)} O(\gamma(s)) u_n(s) ds - \int \int_{\Omega(\gamma)} O(x) |Du(x)| curv(u)(x) dx.$$

Maximizing the first term means finding a curve across which the image u has maximal contrast. As Kimmel and Bruckstein show, the second term is a topological complexity measure of the image inside the curve : when γ is a level line, this term is exactly 2π times the number of hills and dips of the image, weighted by their heights. Jordan curves along which $D^2u(Du, Du) = 0$ are critical curves for the preceding functional.

Our purpose in this chapter is to complete the above discussion. We shall start by explaining in detail some of the above mentioned results. In particular, we shall discuss the main point left out, namely the shape of g in (5.6) and prove that not all g 's are convenient. More precisely, we shall prove that *the form of g is roughly fixed if one wants to avoid errors in the snake method due to disparities of contrast along the snake*. In continuation, we shall prove that in a normal quality image, *all snakes without reverse contrast can be defined as a subset of the level lines of the image*. To be more precise, we shall show that a *simple selection* of the level lines of the image, by choosing the best contrasted level lines in the level line tree of the image, gives back all perceptually expected contours. This evidence will be given by showing that when we try to maximize average contrast of the locally best contrasted level lines of the image by moving them as initial curves to the Kimmel-Bruckstein snake method, the curves do not move in general.

5.2 Curve flows maximizing the image contrast

In this section, we compute the Euler-Lagrange equations for the main snake energies introduced in the introduction. This derivation is made in [280], [281] ; we tried to make it a bit more explicit.

5.2.1 The first variation of the image contrast functional

Let $\gamma(s)$ be a closed Jordan curve parameterized by arc length, and let g be a real even function, increasing on \mathbb{R}^+ , and C^2 except possibly at 0 (e.g. $g(t) = |t|^\alpha$).

The Kimmel-Bruckstein energy is defined by

$$E(\gamma) = \int_0^{L(\gamma)} g(u_n(s)) ds. \quad (5.7)$$

We now compute its first variation.

PROPOSITION 1 Set $h(t) = g(t) - tg'(t)$. The Gateaux derivative of $E(\gamma)$ with respect to perturbations ε is

$$\begin{aligned} \nabla E_\varepsilon(\gamma) = \int_0^{L(\gamma)} & \left[(g'(u_n))' Du^\perp(\gamma) + g'(u_n) \Delta u(\gamma) \mathbf{n} \right. \\ & \left. - (h(u_n))' \gamma' - h(u_n) Curv(\gamma) \right] . \varepsilon ds \end{aligned} \quad (5.8)$$

We shall need the simple formal computation of the next lemma.

LEMMA 7 Let A be 2×2 symmetric matrix. Then for all pairs of two dimensional vectors v and w , $A(v, w^\perp) - A(v^\perp, w) = \text{Trace}(A)v.w^\perp$.

Proof of proposition 1

We consider a perturbation $\lambda\varepsilon$ of γ , with $\lambda \in \mathbb{R}$ and $\varepsilon : [0, L(\gamma)] \rightarrow \mathbb{R}^2$. We want to compute

$$\nabla_\varepsilon E(\gamma) = \frac{d}{d\lambda} E(\gamma + \lambda\varepsilon)_{/\lambda=0}.$$

Since $|\gamma'(s) + \lambda\varepsilon'(s)|$ has no reason to be equal to 1, s is not an arc length parameterization of the curve $s \mapsto \gamma(s) + \lambda\varepsilon(s)$. Thus, we differentiate the general (non-arc-length) form of E from (5.7) and write

$$E(\gamma + \lambda\varepsilon) = \int_0^{L(\gamma)} g \left(Du(\gamma + \lambda\varepsilon) \cdot \frac{\gamma'^\perp + \lambda\varepsilon'^\perp}{|\gamma' + \lambda\varepsilon'|} \right) |\gamma' + \lambda\varepsilon'| ds.$$

Now, since $|\gamma'| = 1$ and $\mathbf{n} = \gamma'^\perp$, we have

$$\frac{d}{d\lambda} |\gamma' + \lambda\varepsilon'|_{/\lambda=0} = \varepsilon' \cdot \gamma' \quad \text{and} \quad \frac{d}{d\lambda} \left(\frac{\gamma'^\perp + \lambda\varepsilon'^\perp}{|\gamma' + \lambda\varepsilon'|} \right)_{/\lambda=0} = \varepsilon'^\perp - (\gamma' \cdot \varepsilon') \mathbf{n}.$$

Hence,

$$\begin{aligned} & \nabla_\varepsilon E(\gamma) \\ &= \int_0^L g'(u_n) Du(\gamma) \cdot (\varepsilon'^\perp - (\gamma' \cdot \varepsilon') \mathbf{n}) + g'(u_n) D^2 u(\gamma)(\mathbf{n}, \varepsilon) + g(u_n) \gamma' \cdot \varepsilon' \\ &= \int_0^L g'(u_n) Du(\gamma) \cdot \varepsilon'^\perp + h(u_n) \gamma' \cdot \varepsilon' + g'(u_n) D^2 u(\gamma)(\mathbf{n}, \varepsilon), \end{aligned}$$

since $Du(\gamma) \cdot (\gamma' \cdot \varepsilon') \mathbf{n} = u_n \gamma' \cdot \varepsilon'$ and $h(t) = -u_n g'(u_n) + g(u_n)$. We notice that all terms are assumed to be C^2 and are integrated on a closed contour. Thus, we

can perform all integrations by parts we wish without boundary terms. This yields

$$\begin{aligned}\nabla_\varepsilon E(\gamma) &= \int_0^L -(g'(u_n))' Du(\gamma) \cdot \varepsilon^\perp - g'(u_n) D^2 u(\gamma)(\gamma', \varepsilon^\perp) \\ &\quad - (h(u_n))' \gamma' \cdot \varepsilon - h(u_n) Curv(\gamma) \cdot \varepsilon + g'(u_n) D^2 u(\gamma)(\mathbf{n}, \varepsilon).\end{aligned}$$

We conclude by using the fact that $Du(\gamma) \cdot \varepsilon^\perp = -Du(\gamma)^\perp \cdot \varepsilon$ and that

$$-D^2 u(\gamma)(\gamma', \varepsilon^\perp) + D^2 u(\gamma)(\mathbf{n}, \varepsilon) = \Delta u(\gamma) \mathbf{n} \cdot \varepsilon$$

thanks to Lemma 7. \square

Proposition 1 permits to deduce the *maximization gradient flow* for $E(\gamma)$, namely $\frac{\partial \gamma}{\partial t} = \nabla E(\gamma)$. As usual with curve evolution, we can restrict ourselves to depict the evolution by its velocity in the direction of the normal $\gamma'^\perp = \mathbf{n}$ to the curve γ .

COROLLARY 1 *The curve flow maximizing the curve contrast (5.7) is*

$$\frac{\partial \gamma}{\partial t} = (g'(u_n))' (Du^\perp(\gamma) \cdot \mathbf{n}) + g'(u_n) \Delta u(\gamma) \mathbf{n} - h(u_n) Curv(\gamma). \quad (5.9)$$

Notice that the term $-(h(u_n))' \gamma'$ disappears in the normal flow formulation because this tangential motion does not influence the geometric motion of γ . It can instead be interpreted as a motion of the parameterization of the curve $\gamma(s)$ itself. For the same reason, we only kept the component of $Du^\perp(\gamma)$ normal to the curve, namely $(Du^\perp(\gamma) \cdot \mathbf{n})$ instead of $Du^\perp(\gamma)$. The last two terms of the above expression have been left unaltered, since they both are terms orthogonal to the tangent to the curve, γ' .

A special case of (5.9) is obtained when $g(t) = |t|$. In this case, we have $h(t) = 0$ and the normal flow equation boils down to

$$\frac{\partial \gamma}{\partial t} = ((g'(u_n))' (Du^\perp(\gamma) \cdot \mathbf{n}) + g'(u_n) \Delta u(\gamma) \mathbf{n}). \quad (5.10)$$

If u_n has constant sign on γ , i.e. if u does not reverse contrast along γ , we have $g'(u_n) = \text{sign}(u_n) = \pm 1$ and therefore $(g'(u_n))' = 0$. Thus, the equation becomes

$$\frac{\partial \gamma}{\partial t} = \text{sign}(u_n) \Delta u(\gamma) \mathbf{n}. \quad (5.11)$$

Thus, one obtains a *variational interpretation* of the Hildreth-Marr edge detector. Those edges can be obtained as steady states for a *boundary contrast maximizing flow*.

PROPOSITION 2 (Kimmel-Bruckstein) *The curves of an image which are local extrema of the contrast $E(\gamma) = \int_\gamma |u_n|$ satisfy $\Delta(\gamma(s)) = 0$ and are therefore Hildreth-Marr edges, provided u_n does not change sign along the curve.*

5.2.2 A parameterless edge equation

Following Fua and Leclerc [203] and Kimmel and Bruckstein [280], we now consider a second, probably better, definition of a boundary as a closed curve along which the *average* contrast is maximal. Thus, we consider the energy

$$F(\gamma) = \frac{1}{L(\gamma)} \int_0^{L(\gamma)} g(u_n) ds = \frac{E(\gamma)}{L(\gamma)}, \quad (5.12)$$

where $L(\gamma)$ is the length of γ . Again, we can compute the first variation of F with respect to C^1 perturbations of γ , which yields

$$\nabla_\varepsilon F(\gamma) = \frac{1}{L(\gamma)} \left(\nabla_\varepsilon E(\gamma) - F(\gamma) \nabla_\varepsilon L(\gamma) \right).$$

Now,

$$\begin{aligned} \nabla_\varepsilon L(\gamma) &= \frac{d}{d\lambda} \left(\int_0^{L(\gamma)} |\gamma' + \lambda\varepsilon'| ds \right)_{/\lambda=0} \\ &= \int_0^{L(\gamma)} \gamma' \cdot \varepsilon' = - \int_0^{L(\gamma)} Curv(\gamma) \cdot \varepsilon. \end{aligned}$$

Thus, we can again write an evolution equation for the average contrast maximizing flow,

$$\frac{\partial \gamma}{\partial t} = ((g'(u_n))' (Du^\perp(\gamma) \cdot \mathbf{n}) + g'(u_n) \Delta u(\gamma)) \mathbf{n} + (F(\gamma) - h(u_n)) Curv(\gamma). \quad (5.13)$$

If we again consider the important case where the sign of u_n is constant along γ and if we choose $g(t) = |t|$, we obtain, since $g'(t) = \text{sign}(t)$,

$$\frac{\partial \gamma}{\partial t} = \text{sign}(u_n) \Delta u(\gamma) \mathbf{n} + F(\gamma) Curv(\gamma). \quad (5.14)$$

The steady states of the preceding equation yield an interesting equilibrium relation, if we notice that $F(\gamma)$ is nothing but the average contrast along the boundary.

We get what we can qualify a parameterless edge or image boundary equation, namely

$$\text{sign}(u_n) \Delta u(\gamma(s)) \mathbf{n} + F(\gamma) Curv(\gamma(s)) = 0 \quad (5.15)$$

(this equation is derived in [280, 281]). Notice that in this equation, $F(\gamma)$ is a global term, not depending upon s . The preceding equation is valid if $Du \cdot \mathbf{n}$ does not change sign, which means that we observe no reverse contrast along γ . Recall also that $Curv(\gamma) = \gamma''(s)$ (s being the arclength). Since $F(\gamma) \geq 0$, Equation (5.14) is well posed and its implementation is very similar to the implementation of the *geodesic snake* equation, which also has a propagation term and a curvature

term :

$$\frac{\partial \gamma}{\partial t} = f(\gamma) \text{Curv}(\gamma) - (Df \cdot \mathbf{n}) \mathbf{n}, \quad (5.16)$$

where here $f(\mathbf{x})$ denotes an “edge map”, i.e. a function vanishing (or close to 0) where u has edges. Typically, $f(\mathbf{x}) = (1 + |Du(\mathbf{x})|)^{-1}$. We can interpret the geodesic snake equation by saying that the “snake” is driven by the second term $Df \cdot \mathbf{n}$ towards the smaller values of f , while the first term roughly is a length minimizing term, but tuned by f . Thus, the snake tends by this term to shrink, but this smoother slows down where f is small, namely in a neighborhood of high gradient points for u .

5.2.3 Numerical scheme

We now describe in detail a numerical scheme implementing the maximization of (5.12). For a non-Euclidean parameterization $\gamma(p) : [a, b] \rightarrow \mathbb{R}^2$, the energy we want to maximize writes

$$F(\gamma) = \frac{\int_a^b g \left(Du \cdot \frac{\gamma'(p)^\perp}{|\gamma'(p)|} \right) |\gamma'(p)| dp}{\int_a^b |\gamma'(p)| dp}. \quad (5.17)$$

Rather than writing the Euler equation for (5.17) and *then* discretizing it, we discretize the energy and compute its exact derivative with respect to the discrete curve. Let us suppose that the snake is represented by a polygonal curve $M_1..M_n$ (either closed or with fixed endpoints). For the curve length, we can take

$$L = \sum_i \|\Delta_i\| \quad \text{with} \quad \Delta_i = M_{i+1} - M_i.$$

Now the discrete energy can be written $F = \frac{E}{L}$, with

$$E = \sum_i g(t_i) \|\Delta_i\|,$$

$$t_i = w_i \cdot \frac{\Delta_i}{\|\Delta_i\|}, \quad w_i = Du^\perp(\Omega_i), \quad \text{and} \quad \Omega_i = \frac{M_i + M_{i+1}}{2}.$$

Differentiating E with respect to M_k , we obtain

$$\nabla_{M_k} F = \frac{1}{L} \left(\nabla_{M_k} E - F \nabla_{M_k} L \right)$$

with

$$\nabla_{M_k} L = \frac{\Delta_{k-1}}{\|\Delta_{k-1}\|} - \frac{\Delta_k}{\|\Delta_k\|},$$

$$\nabla_{M_k} E = v_k + v_{k-1} + z_{k-1} - z_k,$$

$$v_i = \frac{1}{2} g'(t_i) D((Du^\perp)^T)(\Omega_i) \Delta_i,$$

$$z_i = g'(t_i)w_i + h(t_i)\frac{\Delta_i}{\|\Delta_i\|}$$

and $h(t) = g(t) - tg'(t)$. Note that

$$D((Du^\perp)^T) = D(-u_y \ u_x) = \begin{pmatrix} -u_{xy} & u_{xx} \\ -u_{yy} & u_{xy} \end{pmatrix}.$$

Numerically, we compute Du at integer points with a 3×3 finite differences scheme, and D^2u with the same scheme applied to the computed components of Du . This introduces a slight smoothing of the derivatives, which counterbalances a little the strong locality of the snake model. These estimations at integer points are then extended to the whole plane using a bilinear interpolation.

To compute the evolution of the snake, we use a two-steps iterative scheme.

1. The first step consists in a reparameterization of the snake according to arc length. It can be justified in several ways : aside from bringing stability to the scheme, it guarantees a *geometric* evolution of the curve, it ensures an homogeneous estimate of the energy, and it prevents singularities to appear too easily. However, we do not prevent self-intersections of the curve.
2. The second step is simply a gradient evolution with a fixed step. If $(M_i^n)_i$ represents the (polygonal) snake at iteration n and $(\tilde{M}_i^n)_i$ its renormalized version after step 1, then we set

$$M_i^{n+1} = \tilde{M}_i^n + \delta \nabla_{\tilde{M}_i^n} F.$$

A numerical experiment realized with this scheme is shown on Figure 5.1.

5.2.4 Choice of the function g

In this part, we shall show that the shape of the contrast function g is extremely relevant. Let us consider the average contrast functional (5.12), where g is an increasing function. This energy, which has to be maximized, is an arc length weighting of the contrast $g(u_n)$ encountered along the curve. In particular, it increases when the curve is lengthened by “adding” a high-contrasted part, or when it is shortened by “removing” a low-contrasted part (here, the qualities “high” and “low” are to be considered with respect to the average contrast of the curve). Of course, these operations are not so easy to realize, because the snake must remain a Jordan curve, but we shall see now that this remark has consequences on the local and the global behavior of the snake.

It is all the more easy to increase the energy by shrinking the curve in low-contrasted parts that the function g increases more quickly. This means that if we want to be able to reach object contours presenting strong variations of contrast, we must choose a slowly increasing function for g . Let us illustrate this by a little computation, associated to the numerical experiment of Figure 5.2.

Consider a white square with side length 1 superimposed to a simple background image whose intensity is a linear function of x , varying from black to

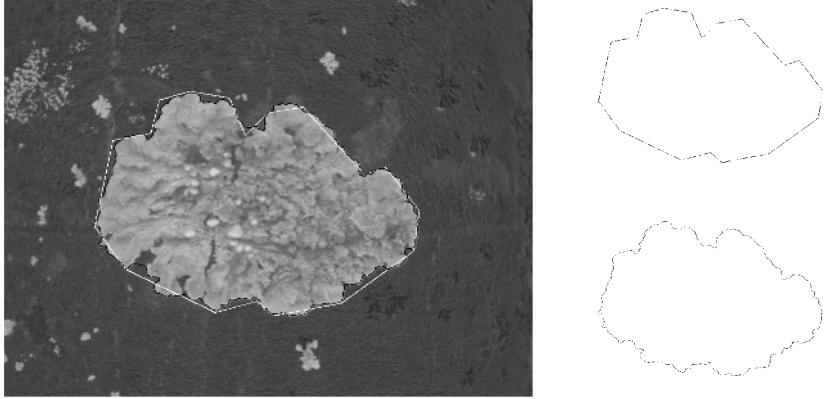


Figure 5.1. An initial contour drawn by hand on the lichen image (curve upright, in white on left image) and its final state (curve on the bottom right of left image, in black) computed with the average contrast snake model (5.12) for $g(t) = |t|$. The evolution allows an important improvement in the localization of the boundary of the object, as illustrated by the energy gain (360%, from 8.8 to 40.8). This shows the usefulness of the snake model as an interactive tool for contour segmentation.

light gray. If a and b are the values of u_n on the left and right sides of the square (we assume that $b < a$), and if γ is the boundary of the square, we have

$$F(\gamma) = \frac{E}{L}, \quad \text{with } L = 4 \quad \text{and} \quad E = g(a) + g(b) + 2 \int_0^1 g(a + (b - a)t) dt.$$

Now we would like to know if γ is an admissible final state of the snake model (that is a local maximum of F) or not. If we shrink a little the curve γ by “cutting” by $\varepsilon > 0$ the two right corners at 45°, we obtain a curve γ_ε whose energy is

$$F_\varepsilon(\gamma) = \frac{E_\varepsilon}{L_\varepsilon}, \quad \text{with} \quad L_\varepsilon = L - 4\varepsilon + 2\varepsilon\sqrt{2}$$

$$\text{and} \quad E_\varepsilon = E - 2 \int_{1-\varepsilon}^1 g(a + (b - a)t) dt - 2\varepsilon g(b) = E - 4\varepsilon g(b) + o(\varepsilon).$$

Since ε may be arbitrarily small, we know that γ cannot be optimal as soon as

$$\frac{L - L_\varepsilon}{L} > \frac{E - E_\varepsilon}{E}$$

for $\varepsilon > 0$ small enough. Using the previous estimates of E_ε and L_ε and the fact that $E > 3g(b) + g(a)$, we can see that this condition is satisfied for $\varepsilon > 0$ small enough as soon as

$$\frac{4 - 2\sqrt{2}}{4} > \frac{4g(b)}{3g(b) + g(a)},$$

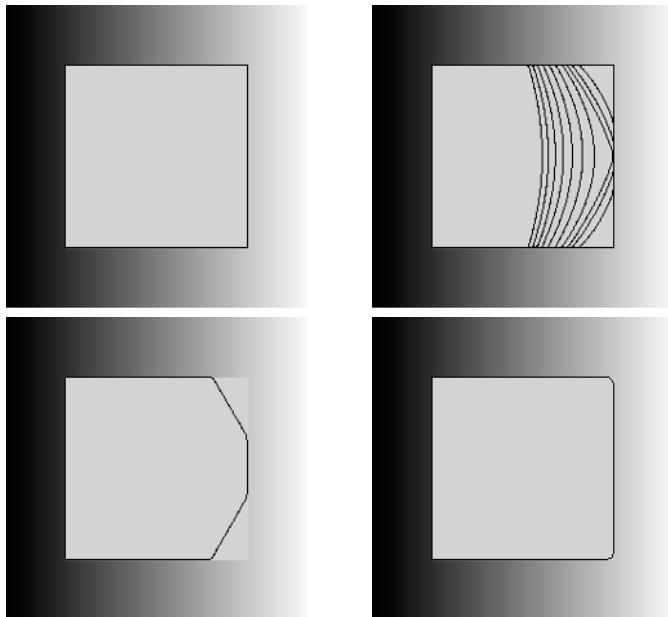


Figure 5.2. Influence of the function g for a synthetic image. The snake model is applied to a synthetic image made of a bright square on a ramp background. Up left: original contour (can be detected as the unique maximal meaningful boundary of this image, see Section 5.3). Top, right: for $g(t) = |t|$, the snake collapses into a “flat curve” enclosing the left side of the square (some intermediate states are shown). Down, left: for $g(t) = |t|^{0.85}$, the snake converges to an intermediate state. Downright: for $g(t) = |t|^{0.5}$, the snake hardly moves, which means that the initial contour is nearly optimal despite the large difference of contrast between the left side and the right side of the square. Contours with large variations of contrast are more likely to be optimal curves for low powers, for which the energy is less sensitive to outliers.

which can be rewritten

$$\frac{g(a)}{g(b)} > \frac{2 + 3\sqrt{2}}{2 - \sqrt{2}} = 10.65\dots$$

Hence, the ratio $g(a)/g(b)$ must be kept (at least) below that threshold, and as small as possible in general in order to avoid the shrinkage of low-contrasted boundaries. If we choose a power function for g (that is $g(t) = |t|^\alpha$), this is in favor of a small value of α , as illustrated on Figure 5.2.

More generally, all this is in favor of a function g increasing as slowly as possible, that is to say almost constant. On the other hand, it is well known that all digital images have a level of noise (e.g. quantization noise to start with) which makes unreliable all gradients magnitudes below some threshold θ . This leads us to the following requirement.

Flatness contrast requirement for snakes. *The contrast function for snake energy must satisfy, for some θ :*

- if $t \leq \theta$, $g'(t)$ is high ;
- if $t \geq \theta$, $g(t)$ is flat and $g(t) \rightarrow g(\infty)$, with $g(\infty) < \infty$.

In the next section, we describe a way to compute θ as a meaningfulness threshold for the gradient, in function of the length of the curve.

Now we shall focus on the special family of power functions $g(t) = |t|^\alpha$. For any of these functions, the snakes method is zoom invariant, namely, up to time scale change, the snake's evolution remains the same when we zoom both the image and the initial contour. Conversely, this identity of snake evolutions leads to ask that the energy of a snake and the one of its zoomed counterpart are proportional. It is easy to prove that this implies that $g(\lambda t) = f(\lambda)g(t)$ for some function $f(\lambda)$. If g is continuous and even, this implies that $g(t) = C|t|^\alpha$.

According to the requirements above, we can predict the following behaviors.

Experimental predictions. *Consider the average contrast energy*

$$F(\gamma) = \frac{1}{L(\gamma)} \int_0^{L(\gamma)} g(u_n(\gamma(s)))ds, \quad \text{with} \quad g(t) = |t|^\alpha. \quad (5.18)$$

- When α is large, all snakes present straight parts and shrink from the parts of the boundaries with weaker gradient.
- The smaller α is the better : snakes will be experimentally more stable and faithful to perceptual boundaries when $\alpha \rightarrow 0$.

We checked these predictions on several numerical experiments. First, in a synthetic case (a white “comb” on a ramp background), we can see the evolution of the snake (Figure 5.3) and its final state in function of the power α (Figure 5.4). As expected, some teeth of the comb are not contrasted enough to be kept for $\alpha = 1$, and a smaller value is required to have an accordance between the snake and the perceptual boundary.

Our predictions remain true when the snake model (5.18) is applied to the “real” image of a bird (see Figure 5.5) : as expected, we notice that the low contrasted parts of the contour are kept only when the power α is small and replaced by straight lines when α is large.

The trend to favor high contrasted parts in the snake model, which becomes very strong for large powers, has some consequences on the numerical simulations. As we noticed before, if the contrast is not constant along the curve one can always increase the average contrast (5.12) by lengthening the curve in the part of the curve which has the highest contrast. If the curve γ was not required to be a regular Jordan curve, there would be no nontrivial local maximum of the functional $F(\gamma)$, since the curve could “duplicate” itself infinitely many times in the highest contrasted part by creating cusps. This formation of cusps was proven by Mumford and Shah [388] in the case $g(t) = t^2$. In the model we presented,

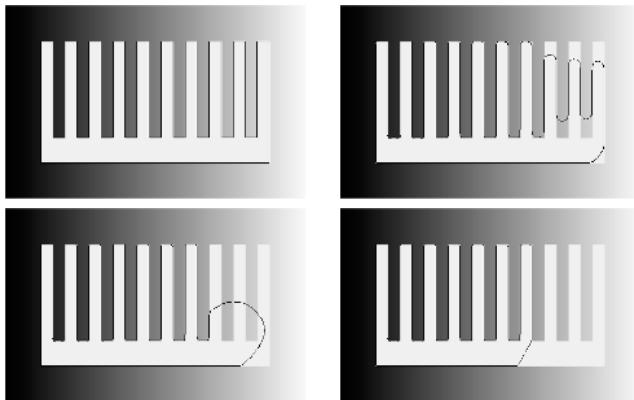


Figure 5.3. Evolution of the snake for $g(t) = |t|$. The snake model is applied to a synthetic image made of a bright comb on a ramp background. Uptleft: original contour (detected as the unique maximal meaningful boundary of this image).

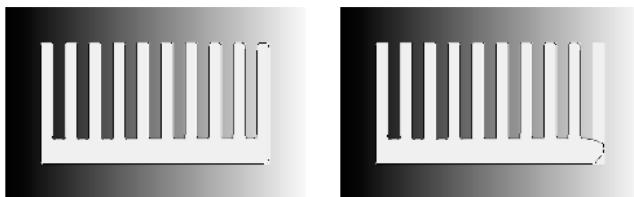


Figure 5.4. Influence of the function g for a synthetic image. The snake model is applied to a synthetic image made of a bright comb on a ramp background. $g(t) = |t|^{0.4}$ (left), $g(t) = |t|^{0.55}$ (right), $g(t) = |t|$ (bottom, right of Figure 5.3).

cusps may be avoided because the duplication of the curve cannot be realized directly for a geometric curve evolution, which consists in infinitesimal *normal* deformations. In general, the creation of a cusp with a normal curve evolution will not be compatible with the need for the energy to increase during the evolution, so that in many cases the snake will not be able to fold itself during the energy maximization process.

Numerically, this effect is more difficult to avoid since the curve does not evolve continuously but step by step, so that the energy gap required to develop a cusp may be too small to stop the process. This is especially true for large powers, for which self-folding is often observed in practice (though, of course, this phenomenon depends a lot on the time step used in the gradient maximization scheme). In the bird image for example, one may notice a “thick part” of the curve for $\alpha = 3$ (Figure 5.6), which corresponds to such a self-folding. The numerical experiment of Figure 5.7, which is in some way the “real case” analog of Figure 5.2, shows that curve duplication may also be attained continuously in the

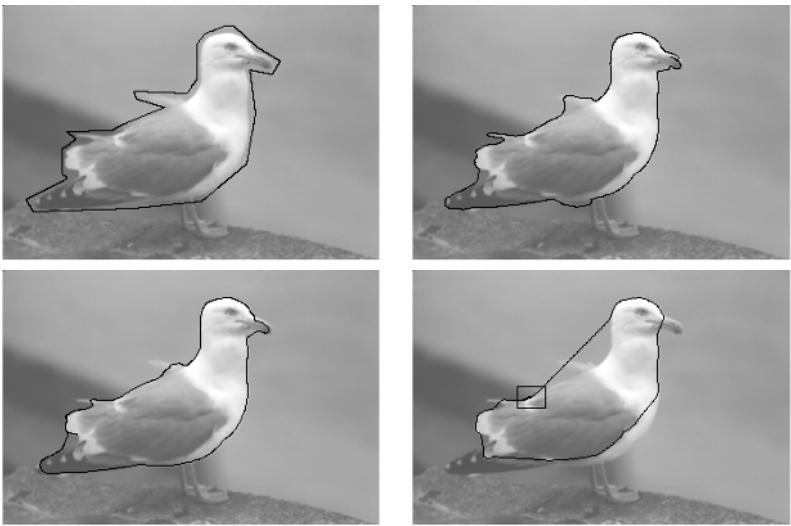


Figure 5.5. Contour optimization in function of g for the bird image. An initial contour (up, left) was first designed by hand. Then, it was optimized by the snake model for different functions $g : g(t) = |t|^{0.5}$ (upright), $g(t) = |t|$ (down, left), and $g(t) = |t|^3$ (downright). As the power increases, the snake becomes more sensitive to edges with high contrast and smooth (or cut) the ones with low contrast. (original bird image from F. Guichard)

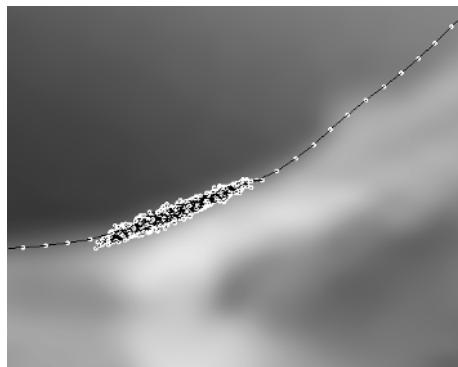


Figure 5.6. Zoom on the curve duplication that appears for $g(t) = |t|^3$ in the highest contrasted part (rectangle drawn on the bottom, right image of Figure 5.5). The discretization of the snake (black curve) is shown by white dots.

energy maximization process (the initial region enclosed by the snake collapses into a zero-area region enclosed by a “flat curve”).

In the next section, we introduce a boundary detector which we recently developed and which we shall prove to be theoretically close and experimentally

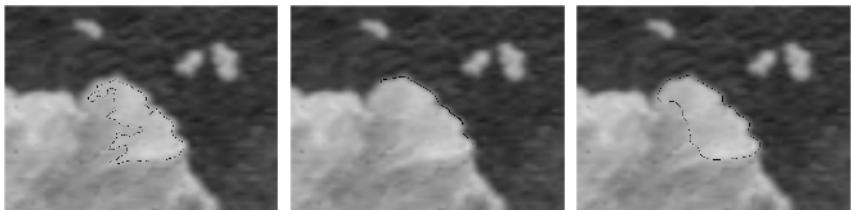


Figure 5.7. Influence of the function g in the self-folding phenomenon. The initial boundary (left) is a maximal meaningful boundary of the lichen image. Like for the square image (see Figure 5.2), the contrast along this curve present strong variations. The snake collapses into a self-folded “flat curve” with two cusps for $g(t) = |t|$ (middle) but remains a Jordan curve for $g(t) = \sqrt{|t|}$ (right).

almost identical to the Kimmel-Bruckstein detector, when g satisfies the flatness requirement.

5.3 Meaningful boundaries

In very much the same way as different species of animals can converge to the same morphology by the way of the evolutive pressure¹, two image analysis structures with very different origins, namely the variational snakes and the maximal meaningful level lines, arrive at almost exactly the same numerical results. Level lines for image representation have been proposed in [83] as an efficient *contrast invariant* representation of any image. This representation stems from Mathematical Morphology [467] where connected components of level sets are extensively used as image features and indeed are contrast invariant features (level lines are nothing but the boundaries of level sets). Level lines are closed curves, like the snakes. They are a complete representation of the image. They have a tree structure which permits a fast computation, the so called Fast Level Set Transform [367]. In addition, they satisfy the easy topological change requirement : their topology changes effortless at saddle points and permit level lines to merge or to split numerically by just evolving their level. The only drawback of level lines is : they are many. Let us now describe a pruning algorithm proposed in [158] to reduce drastically the number of level lines without changing the image aspect and having thus an image representation with *only a few closed curves*. The result consists roughly in giving all essential (in some information theoretical sense) best contrasted level lines. Two examples of this zero-parameter method are given in Figures 5.8 and 5.9. An approach very close to this idea of meaningful level lines has been proposed independently by Kervrann and Trubuil in [268].

¹In Australia, some marsupials have evolved into a wolf-like predator species : the Tasmanian Thylacine (*Thylacinus cynocephalus*). This species unfortunately disappeared in 1936, but good drawings and photographs are available.

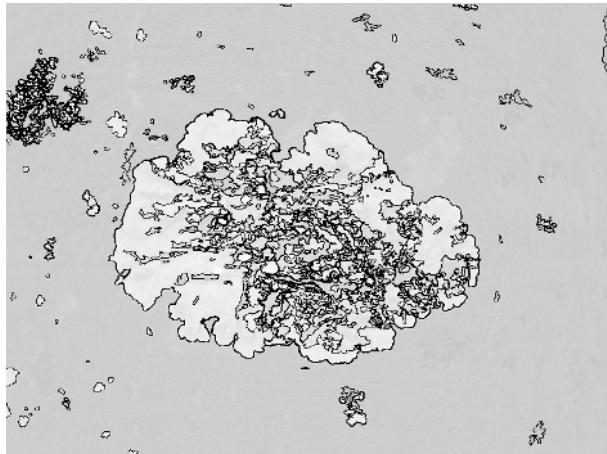


Figure 5.8. The maximal meaningful boundaries of the lichen image (superimposed in grey, see Figure 5.1).



Figure 5.9. The maximal meaningful boundaries of the bird image (superimposed in grey, see Figure 5.5).

Let u be a discrete image, of size $N \times N$. We consider the level lines at quantized levels. The quantization step is chosen in such a way that level lines make a dense covering of the image: if e.g. this quantization step is 1 and the natural image ranges 0 to 255, we get such a dense covering of the image. A level line can be computed as a Jordan curve contained in the boundary of an upper (or lower) level set with level λ ,

$$\chi_\lambda = \{\mathbf{x}, u(\mathbf{x}) \leq \lambda\} \quad \text{and} \quad \chi^\lambda = \{\mathbf{x}, u(\mathbf{x}) \geq \lambda\}.$$

It can also be given a simple linear spline description if one uses a bilinear interpolation : in that case, level lines with level λ are solved by explicitly solving the equation $u(\mathbf{x}) = \lambda$ [317].

For obvious stability reasons, we consider in the following only level lines along which the gradient is not zero. What follows is a very fast summary of the theory developed in [158].

Let \mathcal{L} be a level line of the image u . We denote by l its length counted in number of “independent” points. In the following, we will consider that points at a geodesic distance (along the curve) larger than 2 are independent. Let $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_l$ denote the l considered points of \mathcal{L} . For a point $\mathbf{x} \in \mathcal{L}$, we will denote by $c(\mathbf{x})$ the contrast at \mathbf{x} . It is defined by

$$c(\mathbf{x}) = |Du|(\mathbf{x}), \quad (5.19)$$

where Du is computed by a standard finite difference on a 2×2 neighborhood. For $\mu > 0$, we consider the event : “for all $i = 1..l$, $c(\mathbf{x}_i) \geq \mu$, i.e. each point of \mathcal{L} has a contrast larger than μ ”. From now on, all computations are performed in the Helmholtz framework explained in [157]: we make all computations as though the contrast observations at \mathbf{x}_i were mutually independent. If the gradient magnitudes of the l points were independent, the probability of this event would be $P[c(\mathbf{x}_1) \geq \mu] \cdot P[c(\mathbf{x}_2) \geq \mu] \cdots P[c(\mathbf{x}_l) \geq \mu] = H(\mu)^l$, where $H(\mu)$ is the empirical probability for a point on any level line to have a contrast larger than μ . Hence, $H(\mu)$ is given by the image itself,

$$H(\mu) = \frac{1}{M} \# \{ \mathbf{x} / |Du|(\mathbf{x}) \geq \mu \}, \quad (5.20)$$

where M is the number of pixels of the image where $Du \neq 0$. In order to define a meaningful event, we have to compute the expectation of the number of occurrences of this event in the observed image. Thus, we first define the number of false alarms.

DEFINITION 1 (NUMBER OF FALSE ALARMS) *Let \mathcal{L} be a level line with length l , counted in independent points. Let μ be the minimal contrast of the points $\mathbf{x}_1, \dots, \mathbf{x}_l$ of \mathcal{L} . The number of false alarms of this event is defined by*

$$NF(\mathcal{L}) = N_{ll} \cdot [H(\mu)]^l, \quad (5.21)$$

where N_{ll} is the number of level lines in the image.

Notice that the number N_{ll} of level lines is provided by the image itself. We now define ε -meaningful level lines.

DEFINITION 2 (ε -MEANINGFUL BOUNDARY) *A level line \mathcal{L} with length l and minimal contrast μ is ε -meaningful if $NF(\mathcal{L}) \leq \varepsilon$.*

We can summarize the method in the following way : not all level lines are meaningful ; some can cross flat regions where noise predominates. In order to eliminate such level lines, we first compute the minimum gradient, μ , on a given level line \mathcal{L} with length l . We then compute the probability of the following event

: a level line in a white noise image with the same gradient histogram as our image has contrast everywhere above μ . This probability is the probability of a “contrasted level line happening by chance”. We then compute the false alarm rate, namely the product of this probability by the number of tested level lines. If the false alarm rate is less than ε , the level line is said to be ε -meaningful. In practice, one takes $\varepsilon = 1$, since one does not care much having on the average one wrong among the many meaningful level lines.

Because of the image blur, contrasted level lines form bundles of nearly parallel lines along the edges. We call *interval of level lines* a set of level lines such that each one is enclosed in only one, and contains only another one.

DEFINITION 3 *We say that a level line is maximal meaningful if it is meaningful and if its number of false alarms is minimal among the level lines in the same interval.*

With this definition, one should retain in the simple case of a contrasted object again a uniform background, a single maximal meaningful level line. In all experiments below, we only display maximal meaningful level lines.

5.4 Snakes versus Meaningful Boundaries

In this section, we would like to compare the snake model and the meaningful boundaries model (abbreviated MB).

In terms of *boundary detection*, the MB model has one serious advantage : it is fully automatic. Comparatively, realizing boundary detection with the snake model is much more difficult and an automatic algorithm (that is, with no parameters to set) seems unreachable. Indeed, there are so many ways to change the large set of local maxima of the snake functional that an important user interaction is needed to ensure a good compromise between false detections and missed contours. Among the parameters to set are :

- the function g (we have shown that not many possibilities are left to obtain reliable detections. We were led to choose $g(t) = |t|^\alpha$ with some small α) ;
- the initial contour, which represents a high number of parameters. Starting with a set of fixed contours (e.g. a covering of the image with circles with different radii) requires a real multi-scale strategy and a strong smoothing of the image since the actual contours may be quite different from the ones fixed *a priori*;
- the parameters of the numerical scheme used to implement the snake, including the gradient step and the above-mentioned initial smoothing, required for non-interactive detection. The set of maxima of the discrete snake functional depends heavily on them in general, as shown (for the gradient step) on Figure 5.10.

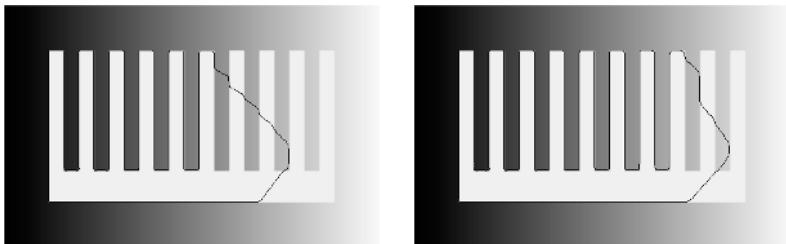


Figure 5.10. Sensitivity of the snake model to numerical parameters (here the time step used for the gradient descent). *The snake model ($g(t) = |t|$) is applied for several values of the gradient step δ : $\delta = 1$ (left), $\delta = 3$ (Figure 5.3, downright), $\delta = 10$ (right).* Due to the huge number of local maxima of the snake functional, the final result is very sensitive to the numerical implementation of the model, in particular to δ .

In terms of *boundary optimization*, that is, the refinement of a raw automatically detected or interactively selected contour, the snake model does not bring very substantial improvements compared to the MB model. We checked this by applying the snake model to the contours detected by the MB model (Figure 5.11 and 5.12). The very little changes brought in these experiments by the snake evolution prove that the curves detected with the MB model are very close to local maxima of the snake model. This is not very surprising since first, the curves delivered by the MB model are level lines (that is, curves for which Du is colinear to the normal \mathbf{n} of the curve at each point), and second, the MB criterion is, like the snake model, based on gradient maximization. This is all the more true that as we discussed previously, the function g used in the snake model should be close to the *gradient thresholding* (that is, a step function selecting points with large enough gradient) realized by the MB model.

In our opinion, these experiments tend to prove that the snake model should be only used when interactive contour selection and optimization is required and when, in addition, the sought object presents contrast inversions. In all other cases and in particular for automatic boundary detection, the meaningful boundaries method appears to be much more practicable.

Acknowledgments

Work partially supported by Office of Naval Research under grant N00014-97-1-0839, Centre National d'Etudes Spatiales, Centre National de la Recherche Scientifique et Ministère de la Recherche et de la Technologie. We thank the Fondation des Treilles for its hospitality during the writing of this paper.

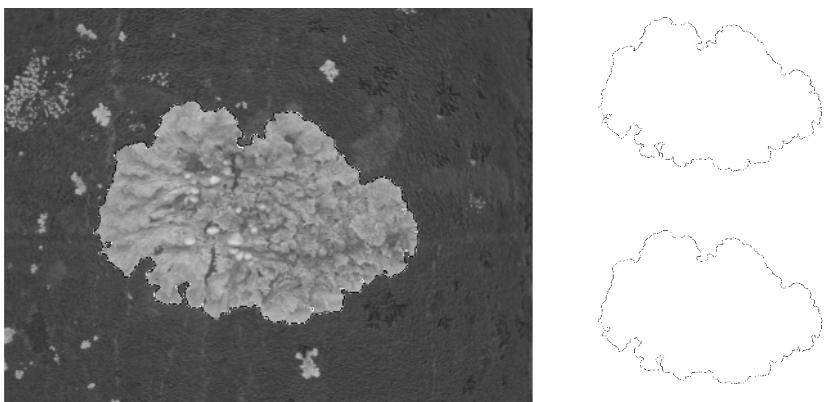


Figure 5.11. Optimization of a maximal meaningful boundary by the snake model. *The contour optimization brought by the snake model (here $g(t) = |t|$) is generally low when the contour is initialized as a contrasted level line (here a maximal meaningful boundary). In this experiment, the total energy is only increased by 17%, from 34.6 to 40.6.*

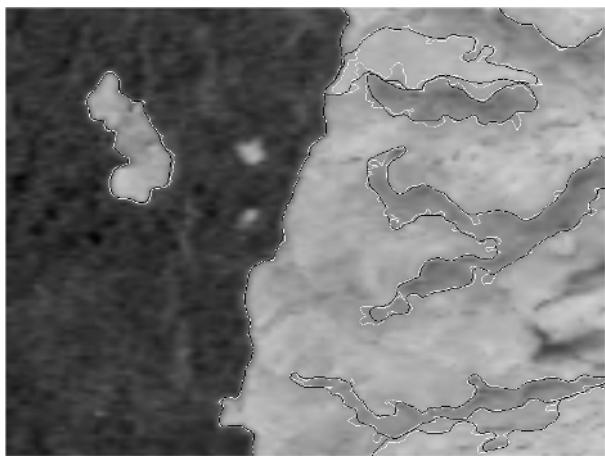


Figure 5.12. Optimization of all maximal meaningful boundaries ($g(t) = \sqrt{|t|}$). *The “objects” in this image are well detected with the maximal meaningful boundaries (in white). The optimization of these contours brought by the snake model (in black) is quite low, as shown by the little gain obtained for the total energy (sum of the energy of each curve) : 7%, from 35.9 to 38.5.*

Part III

Scalar & Vector Image Reconstruction, Restoration

6

Multiplicative Denoising and Deblurring: Theory and Algorithms

Leonid Rudin, Pierre-Luis Lions and Stanley Osher

Abstract

In [447, 449, 450], a constrained optimization type of numerical algorithm for restoring blurry, noisy images was developed and successfully tested. In this paper we present both theoretical and experimental justification for the method. Our main theoretical results involve constrained nonlinear partial differential equations. Our main experimental results involve blurry images which have been further corrupted with multiplicative noise. As in additive noise case of [447, 450] our numerical algorithm is simple to implement and is nonoscillatory (minimal ringing) and noninvasive (recovers sharp edges).

To the memory of our former student, colleague, friend, mathematician Emad Fatemi.

6.1 Introduction

This paper was originally written in 1993 but never appeared in print. Even now, almost ten years later it still contains material that has never appeared in published literature. First, even though there are tens of papers published since on the subject of the Total Variation based image restoration, none of them considers case of the multiplicative noise. Second we present theoretical results on uniqueness and existence of solution of our variational problem that is general enough to apply to the multiplicative noise case. In [1] a proof of existence and uniqueness for TV based variational image restoration problem was presented. In this paper, the proofs are more general and distinct from any published theoretical analysis. Therefore we consider publication of this paper at this time. A very short summary of the TV based approach to image restoration was published in [448]. For

the historical accuracy, as applied to the now established field of PDE based image processing, we mention here that our first publication on the TV based image denoising and the related PDE has been developed in late 80's and was in use by US Government [447], prior to [423]. It has become fashionable in computer vision literature to cite the Variational TV based image processing methods as a special case of the anisotropic diffusion as in the above reference. It is a fact that the anisotropic diffusion formulations never led to correct, well-posed formulation of the basic image restoration problems, i.e. presence of blur and noise, as presented here. In view of the above, and the latest developments in variational methods applied to image processing [368], the TV based formulation, and not the anisotropic diffusion, was the first proper approach to restoring piece-wise continuous functions from noisy and blurry signals.

Below, we follow closely the text of our original paper:

This is the third in a sequence of papers involving the notion of enhancement of images which have been corrupted by noise and blur [449, 450]. The total variation of the image is again minimized subject to constraints involving the point spread function of the blurring process and the statistics of the noise. The constraints are implemented using Lagrange multipliers.

Our first theoretical result is an existence theorem, Proposition 3 below, for the minimization problem described in [450] and in Section 2.

Next, we examine our solution procedure which involves the gradient projection method. This amounts to solving a time dependent partial differential equation on a manifold determined by the constraints. The main theoretical results of this paper involve the study of constrained nonlinear partial differential equations, i.e. existence and uniqueness of solutions to these problems.

Although these theoretical results are obtained for a specific model involving additive noise the theorems could easily be extended to more challenging problems such as multiplicative noise. Our numerical results, presented in Section 4, indicate that the method works well in the more difficult cases. The method again appears to be noninvasive and non-oscillatory.

Our view of the basic algorithm is the premise that the paper image fidelity class is not based on the L^2 norm of the image or its derivatives, but rather L^1 . There is much empirical evidence that human beings "see" L^1 , e.g., [159].

Additionally, the numerical method for performing the minimization, which involves the gradient projection method of Rosen [441], has an interesting geometric interpretation. Minimizing the total variation leads to a numerical step whereby each level set of the intensity function is shifted with velocity equal to its curvature divided by the Euclidean norm of the gradient of the intensity function. Then the result is projected onto the constraint manifold. Recent work by Alvarez, Guichard, Lions and Morel [9], proves rigorously that all morphological operators involve this procedure or a slight variant, (up to scaling by the norm of the gradient). Linear methods, such as those based on minimizing the L^2 norm of the gradient or some higher derivatives as in [248, 539, 540] cannot satisfy the axioms of morphology. The partial differential equations generated by these morphological operators were earlier used to propagate fronts in [401].

Additionally, of course, the BV norm allows piecewise smooth functions with jumps and is the proper space for the analysis of discontinuous functions of this type - e.g. in the mathematical theory of shock waves. Motivated by this, the last author devised the notion of shock filter in his Ph.D. thesis [446], and the last two authors refined and applied this notion to the enhancement of slightly blurred images in [400].

6.2 Restoration Algorithms

As in [450] we solve the following problem. Let

$$u_0(x, y) = (Au)(x, y) + \bar{n}(x, y) \quad (6.1)$$

where \bar{n} is Gaussian white noise. Also $u_0(x, y)$ is the observed intensity function; while $u(x, y)$ is the image to be restored. Both are defined on a region in R^2 . The method is quite general - A needs only to be a compact operator on $L^2(\Omega)$.

Examples we have experimented with include motion, diffraction limited, defocus and Gaussian blur. These are all convolution type integral operators. See [450] for precise definitions.

Our theory will apply only to this problem (however it easily generalizes). In Section 4 we shall experiment with the more complicated models

$$u_0(x, y) = [(Au)(x, y)][\bar{n}(x, y)] \quad (6.2)$$

(multiplicative noise/blurring, model one)

$$u_0(x, y) = (Au)(x, y) + (u(x, y))(\bar{n}(x, y)) \quad (6.3)$$

(multiplicative noise/blurring, model two)

Model one has been treated in [130], for example, using homomorphic filtering, i.e., basically taking the log of u_0 and treating it as a problem involving additive noise, then filtering and applying the exponential. This is not appropriate for model two.

In Section 4, we shall present our restoration algorithms for (6.2) and (6.3). Namely, we shall discuss what the proper constraints are, apply the gradient projection method and demonstrate the results on real images.

Our constrained minimization problem involves the variation of the image, which is a direct measure of how oscillatory it is. The space of functions of bounded variation is appropriate for discontinuous functions. This is well known in the field of shock calculations - see e.g., [480] and the references therein.

Thus, our constrained minimization problem is:

$$\text{minimize } \int_{\Omega} \sqrt{u_x^2 + u_y^2} dx dy \quad (6.4a)$$

subject to constraints involving u_0 .

Our theoretical results involve the following two constraints involving the mean:

$$\int_{\Omega} u \, dx \, dy = \int_{\Omega} u_0 \, dx \, dy \quad (6.4b)$$

and the variance

$$\int_{\Omega} (Au - u_0)^2 \, dx \, dy = \sigma^2. \quad (6.4c)$$

Of course, the x and y derivatives in (6.4a) above and throughout are to be numerically approximated by letting the distance between pixels go to zero, in a standard numerical analysis fashion.

Here (6.4b) indicates that the white noise $\bar{n}(x, y)$ is of zero mean and (6.4c) uses a priori information that the standard deviation of the noise $\bar{n}(x, y)$ is σ .

Following the usual procedure we arrive at the Euler-Lagrange equations

$$0 = \frac{\partial}{\partial x} \left(\frac{u_x}{\sqrt{u_x^2 + u_y^2}} \right) + \frac{\partial}{\partial y} \left(\frac{u_y}{\sqrt{u_x^2 + u_y^2}} \right) - \lambda(A^* Au - A^* u) \text{ in } \Omega \quad (6.5a)$$

with

$$\frac{\partial u}{\partial n} = 0 \text{ on } \partial\Omega. \quad (6.5b)$$

Here A^* is just the adjoint integral operator.

The constraint λ is a Lagrange multiplier chosen so that the constraint (6.4b) is automatically satisfied (see [449]) if

$$\int Au \equiv \int u \text{ and } \int A^* u \equiv \int u \quad (6.6)$$

for each u . This is true up to normalization for a convolution. Thus, we assume (6.6) for simplicity only.

We shall use the gradient-projection method of Rosen [441], which, in this case, becomes the interesting “constrained” partial differential equation

$$u_t = \frac{\partial}{\partial x} \left(\frac{u_x}{\sqrt{u_x^2 + u_y^2}} \right) + \frac{\partial}{\partial y} \left(\frac{u_y}{\sqrt{u_x^2 + u_y^2}} \right) - \lambda A^*(Au - u_0) \quad (6.7a)$$

for $t > 0$, $(x, y) \in \Omega$

$$\frac{\partial u}{\partial n} = 0 \text{ on } \partial\Omega \quad (6.7b)$$

and $u(x, y, 0)$ is given such that (6.4b,c) are satisfied. If 6.4b is satisfied initially, e.g. $u(x, y, 0) = u_0(x, y)$, then, by conservation form of (6.7a,b) and by (6.6), it is always satisfied. Satisfying (6.4c) can be done through a process defined in [449] from a practical point of view. The fact that it is possible to find such a $u(x, y, 0)$ follows from the assumption that the range of A is dense in $L^2(\Omega)$. See Remark 3.1 below.

The projection step in the gradient projection method just amounts to updating $\lambda(t)$ so that (6.4c) remains true in time. This follows (see [450]) if we define

$$\lambda(t) = \frac{\int_{\Omega} \left(\frac{\partial}{\partial x} \left(\frac{u_x}{\sqrt{u_x^2 + u_y^2}} \right) + \frac{\partial}{\partial y} \left(\frac{u_y}{\sqrt{u_x^2 + u_y^2}} \right) \right) \cdot A^*(Au - u_0) dx dy}{\int_{\Omega} (A^*(Au - u_0))^2 dx dy}. \quad (6.8)$$

We thus have a dynamic procedure for restoring the image. As $t \rightarrow \infty$, the steady state solution is the devised restoration.

We would like to conclude this section with a simple mathematical result concerning (6.4a). We first need to formulate the problem in $BV(\Omega)$ - we assume that Ω is smooth enough so that extensions to $BV(\mathbb{R}^2)$ are possible, a fact that is true as long as Ω is smooth or if Ω is a rectangle... We denote by $\int_{\Omega} |Du|$ the seminorm on BV that coincides with $\int_{\Omega} \sqrt{u_x^2 + u_y^2} dx dy$ when u is smooth (or $W^{1,1}$...). Then, we consider the following minimization problem:

$$\inf \left\{ \int_{\Omega} |Du| / u \in BV(\Omega), \int_{\Omega} |Au - u_0|^2 = \sigma^2, \int_{\Omega} (u - u_0) = 0 \right\}. \quad (6.9)$$

PROPOSITION 3 *The minimization problem (6.9) admits at least one solution if A is compact from $L^2(\Omega)$ into $L^2(\Omega)$.*

Proof We first assume that because of (6.6) $A1 = 1$ and thus by standard functional analysis arguments, $\int_{\Omega} |Du| + \|Au\|_{L^2(\Omega)}$ is a norm on $BV(\Omega)$ which is equivalent to the usual norm.

Then, this implies that minimizing sequences of (6.9) are bounded in $BV(\Omega)$ and thus in $L^2(\Omega)$ by Sobolev embedding. Therefore, if $\{u_n\}$ denotes an arbitrary minimizing sequence of (6.9), we may assume without loss of generality that $\{u_n\}$ converges weakly to some u in $BV(\Omega)$ (weak-* convergence) and in $L^2(\Omega)$. Therefore, we recover at the limit (6.4b) and (6.4c) follows from the assumption that A is compact so that Au_n converges to Au in $L^2(\Omega)$. We conclude then easily that u is a minimum since we have by weak convergence

$$\int_{\Omega} |Du| \leq \underline{\lim}_n \int_{\Omega} |Du_n|.$$

Remark 2.1: Notice that the assumption made in Proposition (3) on A excludes the obviously interesting case when A is the identity operator, (the pure denoising case) in which case (6.9) turns out to be a highly non-trivial minimization problem related to isoperimetric inequalities and geometrical problems. Any discussion of this would be too technical for the main purpose of this paper.

6.3 Constrained Nonlinear Partial Differential Equations

In this section, we study equations of the form (6.7a) namely

$$u_t = \sum_{i=1}^2 \frac{\partial}{\partial x_i} \{a_i(\nabla u)\} - \lambda A^*(Au - u_0) \text{ for } x \in \Omega, t > 0 \quad (6.10a)$$

where $\lambda = \lambda(t)$ is a Lagrange multiplier associated to the constraint (6.5b), A is a bounded linear operator from $L^2(\Omega)$ into $L^2(\Omega)$ and A^* denotes its adjoint. Finally, $a_i(p) = \frac{\partial}{\partial p_i} a(p)$ ($i = 1, 2$) where a is smooth (say, $C^2(\mathbb{R}^2)$) with bounded derivatives up to order 2), a is spherically symmetric (rotational invariance). Of course, the model introduced in the preceding section corresponds to $a(p) = |p|$. However, this choice induces such singularities that a mathematical analysis does not seem to be possible. This is why we shall study some model cases involving slightly regularized variants of this choice. In fact, numerically, scaling (6.7a) also induces some regularizations of a quite similar to the ones we make below - and, thus, our analysis covers situations that are quite realistic from the numerical viewpoint.

We shall therefore assume that there exists $\nu \in (0, 1)$ such that

$$(\nu \delta_{ij}) \leq \left(\frac{\partial^2 a}{\partial p_i \partial p_j}(p) \right) \leq (\frac{1}{\nu} \delta_{ij}) \text{ for all } p \in \mathbb{R}^2 \quad (6.11)$$

in the sense of symmetric matrices.

If we go back to our real choice of a , namely $a(p) = |p|$, we see that (6.11) does not hold for p near zero and for $|p| \rightarrow \infty$. The singularity at $p = 0$ induces mathematical and numerical difficulties. In practice we truncate $\frac{\partial a}{\partial p}$ near $p = 0$. The assumption for p large can be relaxed by proving some upper bounds with a rather technical argument (contained in the proof below). We prefer to skip this technical argument in order to avoid confusing the main issue which concerns the imposition of constraints.

Finally, we observe that enforcing (6.4c) while scaling (6.10a) amounts to requiring that

$$\lambda = \left(\int A[u] A^*(Au - u_0) dx \right) \|A^*(Au - u_0)\|_{L^2(\Omega)}^{-2}, \|Au - u_0\|_{L^2(\Omega)} = \sigma \quad (6.10b)$$

where $A[u] = \sum_{i=1}^2 \frac{\partial}{\partial x_i} (a_i(\nabla u))$.

We also prescribe an initial condition

$$u|_{t=0} = u^0 \text{ in } \Omega \quad (6.10c)$$

and boundary condition

$$\frac{\partial u}{\partial n} = 0 \text{ on } \partial\Omega. \quad (6.10d)$$

Of course, we need to assume that u^0 satisfies

$$\|Au^0 - u_0\|_{L^2(\Omega)} = \sigma. \quad (6.12)$$

Remark 3.1: The existence of some u^0 satisfying (6.12) is not obvious and depends very much on the properties of A . This existence is certainly ensured by the following assumption

$$R(A)(\text{range of } A) \text{ is dense in } L^2(\Omega). \quad (6.13)$$

Indeed, this condition implies that there exist $u^1, u^2 \in L^2(\Omega)$ (or as smooth as we wish by density and continuity of A) satisfying

$$\|Au^1 - u_0\|_{L^2(\Omega)} < \sigma, \quad \|Au^2 - u_0\|_{L^2(\Omega)} > \sigma. \quad (6.14)$$

And it is enough to take $u^0 = \theta u^1 + (1 - \theta)u^2$ for some convenient $\theta \in (0, 1)$.

In order to illustrate clearly the mathematical difficulties and results associated with the system (6.10a-d) we begin with the model case where $a(p) = \frac{1}{2}|p|^2$ so that $a_i(\nabla u) = \nabla u$ and $A[u]$ reduces to the classical linear operator $A[u] = \Delta u$ (the Laplace operator). In that case, we can prove the

THEOREM 1 *Let A satisfy (6.13), let $u^0 \in H^1(\Omega)$ satisfy (6.12), let $u_0 \in L^2(\Omega)$ and let $a(p) = \frac{1}{2}|p|^2$ on \mathbb{R}^2 . Then, there exists a unique solution u of (6.10a-d) satisfying: $u \in C([0, \infty); H^1(\Omega)) \cap L^2(0, T; H^2(\Omega))$, $u_t \in L^2(0, T; L^2(\Omega))$, $\lambda \in L^2(0, T)$ for all $T \in (0, \infty)$.*

Remark 3.2: If A is a convolution operator i.e. $Au = k \star u$ then (6.13) holds if and only if the Fourier transform \hat{k} of k satisfies

$$\text{meas } \{\xi \in \mathbb{R}^2 / \hat{k}(\xi) = 0\} = 0. \quad (6.15)$$

Remark 3.3: If A and A^* map boundedly $L^p(\Omega)$ into $L^p(\Omega)$ for $2 \leq p < \infty$ then the proof below also shows that $u \in C([0, \infty); W^{1,p}(\Omega)) \cap L^p(0, T; W^{2,p}(\Omega))$, $u_t \in L^p(0, T; L^p(\Omega))$, $\lambda \in L^p(0, T)$ for all $T \in (0, \infty)$ if $u^0 \in W^{1,p}(\Omega)$, $u_0 \in L^p(\Omega)$.

Remark 3.4: The proof also applies to different types of constraints that can even be nonquadratic constraints. Let us only mention a few possibilities for which the same result as the one above holds. In the case of multiplicative noise and no blur, we can replace (6.10b) by

$$\int_{\Omega} \left(\frac{u}{u_0} \right)^2 dx = \sigma^2 > 0 \quad (6.16)$$

assuming for instance that $\frac{1}{u_0^2} \in L^{\infty}(\Omega)$. Also, we might want to enforce local constraints on a finite partition (or subpartition) of Ω , that in practice can be obtained by a segmentation algorithm. In that case, we consider $\omega_1, \dots, \omega_m$ ($m \geq 1$) measurable sets in Ω such that $\text{meas}[\omega_i \cap \omega_j] = 0$ for all $1 \leq i \neq j \leq m$ and we replace (6.10b)

$$\int_{w_i} |Au - u_0|^2 dx = \sigma_i^2 > 0. \quad (6.17)$$

Then, Theorem 1 still holds for the corresponding equation that involves now m different Lagrange multipliers $\lambda_i = \int_{\omega_i} A[u]A^*(Au - u_0) dx$.

Proof of Theorem 1:

Step 1: General a priori estimates.

Here we list some general consequences of the fact that the evolution equation we are considering is a gradient flow (of a constrained functional). Indeed, multiplying (6.10a) by u_t and using (6.10b), we deduce

$$\int_{\Omega} |u_t|^2 dt + \frac{d}{dt} \int_{\Omega} a(\nabla u) dx = 0 \text{ for } t \geq 0. \quad (6.18)$$

Hence, u_t is bounded in $L^2(0, \infty; L^2(\Omega))$ and ∇u is bounded in $L^\infty(0, \infty; L^2(\Omega))$. In particular, $u = \int_0^t u_t ds + u^0$ is bounded in $L^\infty(0, T, L^2(\Omega))$ for all $T \in (0, \infty)$ and thus u is bounded in $C([0, T]; H^1(\Omega))$ for all $T \in (0, \infty)$.

Step 2: A lower bound

We want to show that $\|A^*(Au - u_0)\|_{L^2(\Omega)}$ is bounded from below uniformly on $[0, T]$ (for all $T \in (0, \infty)$) and that the lower bound depends only on T and on the H^1 norm of u^0 . Indeed, if this were not the case, in view of the estimates shown in Step 1 and in view of (6.10b), this would yield the existence of a sequence $\{u_j\}_{j \geq 1}$ such that u_j is bounded in $H^1(\Omega)$ and

$$\|Au_j - u_0\|_{L^2(\Omega)} = \sigma, \quad \|A^*(Au_j - u_0)\|_{L^2(\Omega)} \xrightarrow{j} 0. \quad (6.19)$$

Without loss of generality, we may assume, extracting a subsequence if necessary, that u_j converges weakly in $H^1(\Omega)$ to some u and thus by Rellich-Kondrakov theorem, u_j converges strongly in $L^2(\Omega)$ to some u . Since A and A^* are bounded from $L^2(\Omega)$ into $L^2(\Omega)$, (6.19) then implies

$$\|Au - u_0\|_{L^2(\Omega)} = \sigma, \quad A^*(Au - u_0) = 0. \quad (6.20)$$

In other words, $Au - u_0$ belongs to the kernel of A^* . But (6.13) implies that this kernel is trivial (reduces to $\{0\}$) therefore $Au = u_0$ and we reach a contradiction with the first statement in (6.20).

Step 3: $L_t^2(H_\lambda^2)$ estimates.

We multiply (6.10a) by $-\Delta u$ and we find

$$\begin{aligned} \frac{d}{dt} \int_{\Omega} \frac{1}{2} |\nabla u|^2 dx + \int_{\Omega} (-\Delta u)^2 dx = \\ (\int_{\Omega} (-\Delta u)[A^*(Au - u_0)] dx)^2 (\|A^*(Au - u_0)\|_{L^2(\Omega)})^{-2}. \end{aligned} \quad (6.21)$$

We then fix $T \in (0, \infty)$, use elliptic regularity and Steps 1 and 2 to deduce

$$\|u\|_{L^2(0, T; H^2(\Omega))}^2 \leq C_0 (1 + \int_0^T \left| \int_{\Omega} (-\Delta u) \{A^*(Au - u_0)\} dx \right|^2 dt) \quad (6.22)$$

where C_0 depends only on T and on H^1 bounds of u^0 .

Next, we observe that since $\{u(t)/t \in [0, T]\}$ is bounded in $H^1(\Omega)$, by Rellich-Kondrakov theorem, $\{u(t)/t \in [0, T]\}$ is relatively compact in $L^2(\Omega)$ and thus

since A and A^* are bounded from $L^2(\Omega)$ into $L^2(\Omega)$, $\{A^*(Au(t) - u_0)/t \in [0, T]\}$ is relatively compact in $L^2(\Omega)$. This implies that we can decompose $A^*(Au(t) - u_0)$ as follows for all $\varepsilon > 0$

$$A^*(Au(t) - u_0) = f(t) + g(t) \quad \forall t \in [0, T] \quad (6.23a)$$

$$\|f(t)\|_{L^2(\Omega)} \leq \varepsilon, \quad \|g(t)\|_{H^1(\Omega)} \leq C(\varepsilon), \quad \forall t \in [0, T] \quad (6.23b)$$

for some $C(\varepsilon)$ that depends only on ε , T and H^1 bounds of u^0 .

Therefore, we have for all $t \in [0, T]$

$$\begin{aligned} \left| \int_{\Omega} (-\Delta u) \{A^*(Au - u_0)\} dx \right| &\leq \varepsilon \|u\|_{H^2(\Omega)} + \left| \int_{\Omega} (-\Delta u) g(t) dx \right| \\ &\leq \varepsilon \|u\|_{H^2(\Omega)} + \left| \int_{\Omega} \nabla u \cdot \nabla g(t) dx \right|. \end{aligned}$$

Since u satisfies (6.10d), and thus finally

$$\left| \int_{\Omega} (-\Delta u) \{A^*(Au - u_0)\} dx \right| \leq \varepsilon \|u\|_{H^2(\Omega)} + C(\varepsilon) \|u\|_{H^1(\Omega)}.$$

Hence, if we input this bound in (6.22) and use the bound shown in Step 1, we deduce

$$\|u\|_{L^2(0,T;H^2(\Omega))} \leq C'(\varepsilon) + 2C_0\varepsilon \|u\|_{L^2(0,T;H^2(\Omega))}^2$$

and we conclude by choosing $\varepsilon = \frac{1}{4C_0}$.

Step 4: Uniqueness.

We consider two solution u, v of (6.10a-d) and denote by λ, μ the corresponding Lagrange multipliers. Obviously, we have, for $u - v = w$

$$w_t - \Delta w = \lambda A^* Aw = (\lambda - \mu) A^*(Av - u_0).$$

Multiplying this equation by w and $-\Delta w$, integrating by parts and summing up, we find easily for all $T \in (0, \infty)$

$$\begin{aligned} \|w(t)\|_{H^1(\Omega)}^2 + \|w\|_{L^2(0,T;H^2(\Omega))}^2 &\leq C_1 \left| \int_0^t \lambda(s) ds \int_{\Omega} (A^* Aw)(-\Delta w + w) dx \right| + \\ &\quad \left| \int_0^t |\lambda - \mu| ds \int_{\Omega} [A^*(Au - u_0)](-\Delta w + w) dx \right| \text{ for all } t \in [0, T] \quad (6.24) \end{aligned}$$

for some positive constant C^1 depending only on T .

Using the same argument as in Step 3, we deduce the following bounds for all $\varepsilon > 0$

$$\left| \int_{\Omega} (A^* Aw)(-\Delta w + w) dx \right| \leq \varepsilon \|w\|_{H^2(\Omega)} \|w\|_{H^1(\Omega)} + C(\varepsilon) \|w\|_{H^1(\Omega)}^2 \quad (6.25a)$$

$$\left| \int_{\Omega} \{A^*(Av - u_0)\}(-\Delta w + w) dx \right| \leq \varepsilon \|w\|_{H^2(\Omega)} + C(\varepsilon) \|w\|_{H^1(\Omega)} \quad (6.25b)$$

where $C(\varepsilon)$ denotes various positive constants depending only on ε and T . Inserting these bounds in (6.24) we find

$$\begin{aligned} \|w(t)\|_{H^1(\Omega)}^2 + \|w\|_{L^2(0,T;H^2(\Omega))}^2 &\leq \\ C_1 \int_0^t |\lambda - \mu| \{\varepsilon \|w\|_{H^2(\Omega)} \|w\|_{H^1(\Omega)} + C(\varepsilon) \|w\|_{H^1(\Omega)}^2\} ds + \\ C_1 \int_0^t |\lambda - \mu| \{\varepsilon \|w\|_{H^2(\Omega)} + C(\varepsilon) \|w\|_{H^1(\Omega)}\} ds. \end{aligned}$$

Using the Cauchy-Schwarz inequality, we deduce easily

$$\begin{aligned} \|w(t)\|_{H^1(\Omega)}^2 + \|w\|_{L^2(0,T;H^2(\Omega))}^2 &\leq \varepsilon \|w\|_{L^2(0,T;H^2(\Omega))}^2 + \\ C(\varepsilon) \int_0^t a(s) \|w(s)\|_{H^1(\Omega)}^2 ds + \\ C_1 \int_0^t |\lambda - \mu| \{\varepsilon \|w\|_{H^2(\Omega)} + C(\varepsilon) \|w\|_{H^1(\Omega)}\} ds \quad (6.26) \end{aligned}$$

where $a \geq 0$, $a \in L^1(0, T)$ (for all $T \in (0, \infty)$).

$$\frac{d}{dt} \|w\|_{L^2(\Omega)}^2 + \nu \|w\|_{H^1(\Omega)}^2 \leq C(1 + |\lambda|) \|w\|_{L^2(\Omega)}^2$$

Next we estimate $|\lambda - \mu|$. In view of Step 2, we have

$$\begin{aligned} |\lambda - \mu| &\leq C_2 \|w\|_{L^2(\Omega)} \left| \int_{\Omega} (-\Delta u) \{A^*(Au - u_0)\} dx \right| + \left| \int_{\Omega} (-\Delta u) A^* Aw dx \right| + \\ &\quad \left| \int_{\Omega} (-\Delta w) \{A^* Av - u_0\} dx \right| \end{aligned}$$

for some $C_2 \geq 0$ which depends only on T . But this yields immediately

$$|\lambda - \mu| \leq b \|w\|_{L^2(\Omega)} + C_3 \|w\|_{H^2(\Omega)}, \text{ for some } b \in L^2(0, T).$$

Going back to (6.26), we obtain for all $t \in [0, T]$.

$$\begin{aligned} \|w(t)\|_{H^1(\Omega)}^2 + \|w\|_{L^2(0,T;H^2(\Omega))}^2 &\leq \varepsilon (1 + C_1 C_3) \|w\|_{L^2(0,T;H^2(\Omega))}^2 + \\ C(\varepsilon) \int_0^t a(s) \|w(s)\|_{H^1(\Omega)}^2 ds + C_1 C_3 C(\varepsilon) \int_0^t \|w\|_{H^1(\Omega)} \|w\|_{H^2(\Omega)} ds + \\ C_1 \varepsilon \int_0^t b(s) \|w\|_{L^2(\Omega)} \|w\|_{H^2(\Omega)} ds + C_1 C(\varepsilon) \int_0^t b(s) \|w\|_{L^2(\Omega)} \|w\|_{H^1(\Omega)} ds. \end{aligned}$$

Using the Cauchy-Schwarz inequality, this yields:

$$\begin{aligned} \|w(t)\|_{H^1(\Omega)}^2 + \|w\|_{L^2(0,T;H^2(\Omega))}^2 &\leq \varepsilon (2 + C_1 C_3) \|w\|_{L^2(0,T;H^2(\Omega))}^2 + \\ &\quad \int_0^t c(s) \|w(s)\|_{H^1(\Omega)}^2 ds \end{aligned}$$

where $c \geq 0$, $c \in L^1(0, T)$.

We then choose $\varepsilon = (2 + C_1 C_3)^{-1}$ and we conclude using Gronwall's inequality.

Conclusion: We conclude the proof of Theorem 1 here since only the existence part has not been completed. But this part is a straightforward consequence of solving approximate problems, proving the same a priori bounds uniformly for the approximate solution and passing to the limit using these bounds. We do not want to give all the details of such a tedious argument. Let us only mention a few possible approximations like a penalty method (penalizing the constraint), implicit time discretization (solving each stationary problem, for each iteration, by a minimization problem similar to the ones solved in Proposition 3), or splitting methods similar to the numerical method presented in the following section (where we solve first the equation without constraints on a time interval of length Δt and we then project back to the obtained solution on the constraints manifold by a simple affine rule). For all these approximation methods, one can adopt the a priori estimates shown above. But we certainly do not want to do so here in order to avoid confusing the main issues in this paper.

We now turn to a nonlinear equation (6.10a) where $a(p)$ satisfies the conditions mentioned in the beginning of this section and (6.11) in particular.

THEOREM 2 *Let $a(p)$ satisfy (6.11), let $u^0 \in H^1(\Omega)$ satisfy (6.12), let $u_0 \in L^2(\Omega)$ and let A satisfy (6.13). Then,*

1. *there exists a solution u of (6.10a-d) satisfying: $u \in C([0, \infty); H^1(\Omega)) \cap L^2(0, T; H^2(\Omega))$, $u_t \in L^2(0, T; L^2(\Omega))$, $\lambda \in L^2(0, T)$ for all $T \in (0, \infty)$.*
2. *If $u_0 \in H^1(\Omega)$ and A, A^* are bounded from $H^1(\Omega)$ into $H^1(\Omega)$, then the solution is unique and $\lambda \in L^\infty(0, T)$ for all $T \in (0, \infty)$.*

Remark 3.5. The analogues of Remarks 3.3 and 3.4 hold here. In particular, using this extra regularity, one can show the uniqueness of solutions by an argument quite similar to the one given in the proof of Theorem 1 and which does not use a regularity result which is too technical to be detailed here.

Remark 3.6. If $\Omega = \mathbb{R}^2$ and A is a convolution operator then A, A^* are bounded operators from $H^1(\Omega)$ into $H^1(\Omega)$ since they are bounded from $L^2(\Omega)$ into $L^2(\Omega)$ and they commute with differentiation.

Proof of Theorem 2. We only explain the modifications that have to be made in the proof of Theorem 1. In particular, Steps 1 and 2 are identical. However, Step 3 has to be modified substantially. The final result being the same, these facts and the uniqueness argument shown below allow us to complete the proof of part i).

The $L_t^2(H_x^2)$ bound follows from multiplying (6.10a) by $-\Delta u$ and making some observations. First of all, $\mathcal{A}[u] = \sum_{i,j=1}^2 \alpha_{ij} \frac{\partial^2 u}{\partial x_i \partial x_j}$ where $\alpha_{ij} = \frac{\partial^2 a}{\partial p_i \partial p_j}(\nabla u)$ and thus $|\mathcal{A}[u]| \leq C|D^2 u|$. Next, we recall an adaptation of a famous inequality due to H.O. Cordes [136] (see also A.I. Koshelev [293]) shown in P.L. Lions [315] in the case of Neumann boundary conditions. There exist

$\alpha > 0, C \geq 0$ such that for all $u \in H^2(\Omega)$ satisfying (6.10d)

$$\int_{\Omega} \mathcal{A}[u] \Delta u \, dx \geq \alpha \|u\|_{H^2(\Omega)}^2 - C \|u\|_{L^2(\Omega)}^2. \quad (6.27)$$

This inequality allows us to adapt easily the rest of the proof made in Step 3.

We conclude with the proof of the uniqueness statement (part ii) above. Let u, v be two solutions of (6.10a-d) and let λ, μ be the corresponding Lagrange multipliers. We denote by $w = u - v$ and multiply by w the equation satisfied by w . We then find in view of (6.11)

$$\frac{1}{2} \frac{d}{dt} \int_{\Omega} w^2 \, dx + \nu \int_{\Omega} |\nabla w|^2 \, dx \leq C \{ |\lambda| \int_{\Omega} w^2 \, dx + |\lambda - \mu| (\int_{\Omega} w^2 \, dx)^{\frac{1}{2}} \} \quad (6.28)$$

Next, we observe that λ can be written as

$$\lambda = - \left\{ \int_{\Omega} \sum_{i=1}^2 a_i(\nabla u) \frac{\partial}{\partial x_i} \{ A^*(Au - u_0) \} \, dx \right\} \|A^*(Au - u_0)\|_{L^2(\Omega)}^{-2} \quad (6.29)$$

and a similar expression holds for μ with u replaced by v .

From these expressions we deduce using the assumptions made about a, A, A^* and u_0

$$|\lambda - \mu| \leq C \|w\|_{H^1(\Omega)} \quad (6.30)$$

(recall that u, v are bounded in $L^\infty(0, T; H^1(\Omega))$ for all $T \in (0, \infty)$).

Inserting this bound in (6.28) we finally deduce

$$\frac{1}{2} \frac{d}{dt} \|w\|_{L^2(\Omega)}^2 + \nu \|w\|_{H^1(\Omega)}^2 \leq C \{ |\lambda| \|w\|_{L^2(\Omega)}^2 + \|w\|_{H^1(\Omega)} \|w\|_{L^2(\Omega)} \}. \quad (6.31)$$

Hence, we have for all $t \in [0, T]$, by the Cauchy-Schwarz inequality

$$\frac{d}{dt} \|w\|_{L^2(\Omega)}^2 + \nu \|w\|_{H^1(\Omega)}^2 \leq C(1 + |\lambda|) \|w\|_{L^2(\Omega)}^2 \quad (6.32)$$

and the uniqueness follows from Gronwall's inequality.

Let us finally observe that the fact that $\lambda \in L^\infty(0, T)$ (for all $T \in (0, \infty)$) is straightforward in view of (6.29) since u is bounded in $H^1(\Omega)$ and $\|A^*(Au - u_0)\|_{L^2(\Omega)}$ is bounded from below (Step 2).

6.4 Restoration of Blurry Images Corrupted by Multiplicative Noise

We begin by considering pure denoising. We are given an image $u_0(x, y)$ where

$$u_0 = u\eta \quad (6.33)$$

The unknown function $u(x, y)$ is the image we wish to restore, and $\eta(x, y)$ is the noise (which we take here to be Gaussian white noise).

We are given the following information

$$\int \eta = 1 \quad (\text{mean one}) \quad (6.34\text{a})$$

$$\int (\eta - 1)^2 = \sigma^2 \quad (\text{given variance}) \quad (6.34\text{b})$$

Thus our constrained optimization algorithm is (6.4a) subject to the following constraints

$$\int \frac{u_0}{u} = 1 \quad (6.35\text{a})$$

$$\frac{1}{2} \int \left(\frac{u_0}{u} - 1 \right)^2 = \frac{\sigma^2}{2} = \frac{1}{2} \int \left(\left(\frac{u_0}{u} \right)^2 - 1 \right). \quad (6.35\text{b})$$

The gradient projection method leads to

$$u_t = \frac{\partial}{\partial x} \left(\frac{u_x}{\sqrt{u_x^2 + u_y^2}} \right) + \frac{\partial}{\partial y} \left(\frac{u_y}{\sqrt{u_x^2 + u_y^2}} \right) - \lambda \frac{u_0^2}{u^3} - \mu \frac{u_0}{u^2}. \quad (6.36)$$

We now have two Lagrange multipliers which we compute by requiring

$$0 = \frac{\partial}{\partial t} \int \frac{u_0}{u} = - \int \frac{u_0}{u^2} u_t = 0 \quad (6.37\text{a})$$

$$0 = \frac{\partial}{\partial t} \frac{1}{2} \int \left(\left(\frac{u_0}{u} \right)^2 - 1 \right) = - \int \frac{u_0^2}{u^3} u_t = 0. \quad (6.37\text{b})$$

This together with (6.36) leads us to two algebraic equations for the two unknowns and the resulting Gram determinant is nonzero.

To demonstrate the denoising results, we select an image that has well defined discontinuities and the fine structures 6.1. This image is degraded by a multiplicative noise $\sigma = 0.2$ and model 1 is used. The noisy image is shown on Figure 6.2. Figure 6.3 shows multiplicative noise removal by the TV method described above. The denoised image demonstrates significant quality improvement particularly in the areas of piecewise smooth data. Some of the fine structures have also been restored, however we believe that the variational problem would have to be modified to be able to reconstruct linear non-piecewise smooth structures.

Next, we consider images which are both blurry and noisy. Our first model is as follows. We are given an image $u_0(x, y)$ for which

$$u_0 = (Au)\eta. \quad (\text{model 1}) \quad (6.38)$$

Here we take A to be convolution with a Gaussian kernel, but the generality is as great as in Section 2 above. The noise η is as above – (6.34a) and (6.34b) are still satisfied.

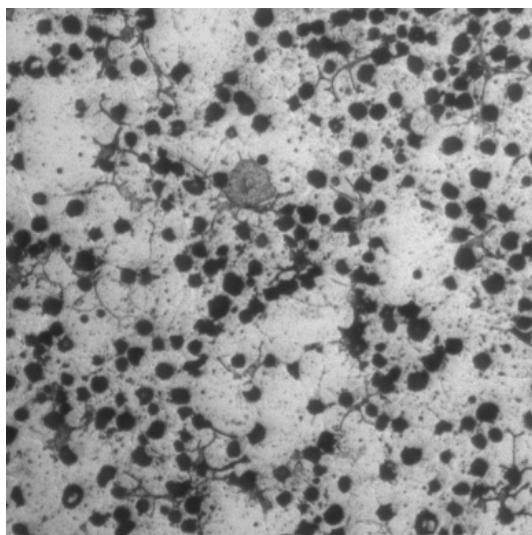


Figure 6.1. Original image

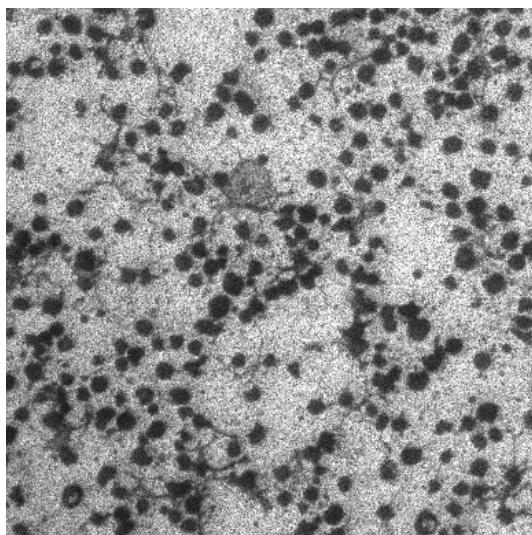


Figure 6.2. Noisy image

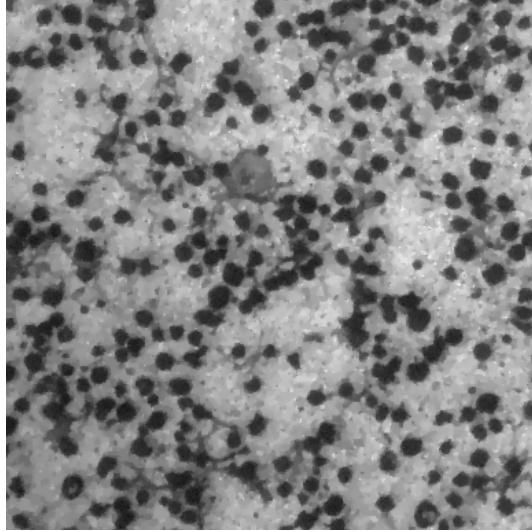


Figure 6.3. Denoised image

Our constraints are

$$\int \frac{u_0}{Au} = 1 \quad (6.39a)$$

$$\frac{1}{2} \int \left(\frac{u_0}{Au} - 1 \right)^2 = \frac{\sigma^2}{2} = \frac{1}{2} \int \left(\left(\frac{u_0}{Au} \right)^2 - 1 \right). \quad (6.39b)$$

The gradient projection method leads to

$$u_t = \frac{\partial}{\partial x} \left(\frac{u_x}{\sqrt{u_x^2 + u_y^2}} \right) + \frac{\partial}{\partial y} \left(\frac{u_y}{\sqrt{u_x^2 + u_y^2}} \right) - \lambda A^* \left(\frac{u_0}{(Au)^2} \right) \left(\frac{u_0}{Au} - 1 \right) - \mu A^* \left(\frac{u_0}{(Au)^2} \right). \quad (6.40)$$

The two Lagrange multipliers are chosen as follows. We take as initial data $u(x, y, 0) = u_0(x, y)$. However we initially set $\lambda = 0$ and choose μ so that

$$\frac{\partial}{\partial t} \int \left(\frac{u_0}{Au} \right) = 0. \quad (6.41a)$$

We monitor the quantity in (6.39b), and let time evolve until the second constraint is satisfied. We then allow λ to be nonzero, continue to enforce (6.41a), and set

$$\frac{\partial}{\partial t} \frac{1}{2} \int \left(\left(\frac{u_0}{Au} \right)^2 - 1 \right)^2 = 0. \quad (6.41b)$$

This leads to a solution for μ and λ .

Figure 6.4 shows the results of a Gaussian blur with $\sigma^2 = 2$ and multiplicative noise with $\sigma = 0.2$. In Figure 6.5 we show the result of TV based restoration.

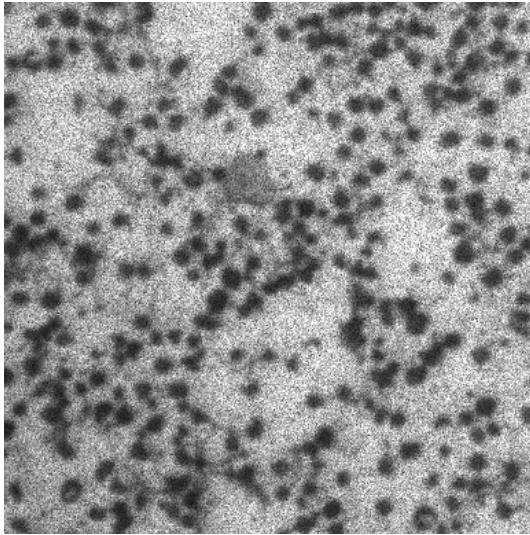


Figure 6.4. Blurred image

Our second model is as follows. We are given an image $u_0(x, y)$ for which

$$u_0 = Au + u\eta. \quad (\text{model 2}) \quad (6.42)$$

Again we chose a Gaussian blur, but the techniques are general. The noise is Gaussian white noise with mean zero and variance σ^2

$$\int \left(\frac{u_0 - Au}{u} \right) = 0 \quad (6.43a)$$

$$\frac{1}{2} \int \left(\frac{u_0 - Au}{u} \right)^2 = \frac{\sigma^2}{2}. \quad (6.43b)$$

The gradient projection method leads us to

$$\begin{aligned} u_t = & \frac{\partial}{\partial x} \left(\frac{u_x}{\sqrt{u_x^2 + u_y^2}} \right) + \frac{\partial}{\partial y} \left(\frac{u_y}{\sqrt{u_x^2 + u_y^2}} \right) - \\ & \lambda \left[A^* \left(\frac{u_0 - Au}{u^2} \right) + \left(\frac{(u_0 - Au)^2}{u^3} \right) \right] - \\ & \mu \left[A^* \left(\frac{1}{u} \right) + \left(\frac{u_0 - Au}{u^2} \right) \right]. \end{aligned} \quad (6.44)$$

We follow a procedure analogous to that of model 1 to compute λ and μ . The experimental results are very similar to model 1 presented above.

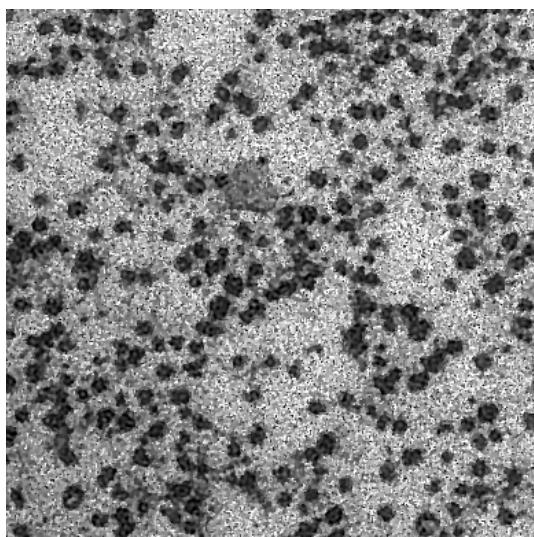


Figure 6.5. Deblurred image

Total Variation Minimization for Scalar/Vector Regularization

Francoise Dibos, Georges Koepfler and Pascal Monasse

Abstract

In this chapter we present regularization by using Total Variation (TV) minimization based on a level set approach. After introducing the problem of regularization and citing related work, we introduce our main tool, the Fast Level Sets Transform (FLST). This algorithm decomposes an image in a tree of shapes which gives us a non-redundant and complete representation of the image. This representation, associated to some theoretical facts about functions of bounded variation, leads us to a TV minimization algorithm based on level sets. We conclude by comparing this approach to other implementation and/or other models and show applications to regularization of gray scale and color images, as well as optical flow.

7.1 Introduction

Due to reception and/or transmission problems, image information might be disturbed by “noise”, which can be considered as a random addition of high frequency components to the original image. In order to further exploit these images regularization is needed. We will not discuss in this chapter statistical/probabilistic regularization techniques, neither will we talk about linear algorithms (*e.g.* convolution). Indeed, it is now well known that image restoration techniques using regularization by smoothing, will destroy on one hand noise and on the other important features (*e.g.* edges, textures). Or images, seen as functions of two variables, are at most piecewise smooth and important data like edges are discontinuities. Thus, by working in the space of functions of bounded total variation (BV) we allow discontinuities in the result of the minimization and thus preserve sharp boundaries. We refer the reader to [345, 550, 170, 24, 219] for more applications of BV modelization in image processing.

The importance of minimizing total variation for image processing purposes, has first been noticed by Osher and Rudin (see [400]). Let u be a real valued function defined on an open subset Ω of \mathbb{R}^2 (Ω is usually a rectangle). Then Rudin and Osher's BV principle consists in the following problem :

$$\text{minimize } \int_{\Omega} |\nabla u| \quad \text{subject to constraints.}$$

The quantity we minimize, the integral of the magnitude of the gradient of u , will be defined more precisely below.

Now the minimization of the Total Variation can be achieved by a gradient descent which yields the following evolution equation :

$$\frac{\partial u}{\partial t} = \operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right) = \operatorname{curv}(u). \quad (7.1)$$

Research on weak solutions for this PDE is developing, interesting approaches are given in [19, 18]. More PDE based methods, applied to image processing problems, can be found in other chapters of this book (e.g. the chapter on PDEs on implicit surfaces).

Since the seminal paper by Osher and Rudin, the TV minimization problem has been largely studied and different approaches have been proposed. The main problem of the above minimization problem comes from the non differentiable argument $|\nabla u|$. Often regularization techniques are used and combined with classical, non linear, numerical schemes.

Chambolle and Lions [96] have shown existence results for the TV minimization with constraints due to different noise types. Blomgren and Chan [53] apply TV minimization to color images. We mention also the related work of Chan and Strong [99], they consider the minimization of piecewise constant radial symmetric functions.

Marquina and Osher [341] propose a convection-diffusion equation with morphological convection and anisotropic diffusion.

We refer the interested reader to just a few authors [448, 22, 253, 54, 165, 557] who studied the TV minimization problem for denoising. Notice that a classical step in the discretization of (7.1) is to replace $|\nabla u|$ by $\sqrt{|\nabla u|^2 + \varepsilon}$, with $\varepsilon > 0$ small.

In this chapter we will give a sense to Equation (7.1) for an image u with bounded variation such that we can deduce a TV minimization algorithm which needs no prior regularization and which is completely based on a level set, *i.e.* geometric, representation of u (see [166, 162]).

The outline of the chapter is as follows.

In Section 7.2.1 we recall some fundamental properties of the space of BV functions. In Section 7.2.2 we present and motivate our global minimization model of Total Variation, therefore we reformulate Equation (7.1) using the Coarea formula on the level sets of u . This leads us to consider for almost each

$\lambda \in \mathbb{R}$ the following equation

$$\int_{E_\lambda} \frac{\partial u}{\partial t} dx = \mathcal{P}(E_\lambda), \quad (7.2)$$

where $E_\lambda = \{x \mid u(x) < \lambda\}$ is the level set associated to λ and $\mathcal{P}(E_\lambda)$ its perimeter.

In order to be able to investigate the properties of (7.2) we will consider two different frameworks, in each we will be able to assure Total Variation minimization.

First we take $u \in BV(\Omega) \cap C^\infty(\Omega)$ and suppose that $\frac{\partial u}{\partial t}$ is constant on each level line ∂E_λ . In Section 7.2.3 we prove that (7.2) minimizes the Total Variation.

In Section 7.3.1 we present an essential tool for our numerical algorithm, the Fast Level Sets Transform (FLST), which gives a complete representation of an image based on shapes. We will give a detailed description of the FLST which is also used in the chapter on image registration: the shapes obtained through the FLST are the features for which correspondences are sought.

In this chapter, this representation will be used in Section 7.3.2 in order to adapt (7.2) to digital image data. This leads us to an operator on digital images which diminishes the Total Variation. The properties of the proposed algorithm are shown in Sections 7.3.3 and 7.3.4.

In Section 7.4 we show and discuss experimental results. We compare our algorithm to classical and less classical PDE based denoising techniques.

7.2 A global approach for Total Variation minimization

7.2.1 The BV function space

Let us first recall some results about BV functions, the reader might consult [182] for more technical details.

Let Ω be an open subset of \mathbb{R}^2 , $C_c^1(\Omega, \mathbb{R}^2)$ the set of functions from Ω to \mathbb{R}^2 , continuously differentiable and of compact support in Ω .

The image u is a function from Ω to \mathbb{R} , by definition $u \in BV(\Omega)$ if $u \in L^1(\Omega)$ and

$$TV(u) = \sup \left\{ \int_{\Omega} u \operatorname{div}(\phi) dx \mid \phi \in C_c^1(\Omega, \mathbb{R}^2), |\phi| \leq 1 \right\} < +\infty.$$

$TV(u)$ is called the *total variation* of u and sometimes written as $\int_{\Omega} |\nabla u|$.

A measurable subset $E \subset \Omega$ has a finite perimeter in Ω if

$$\chi_E \in BV(\Omega);$$

and the perimeter of E , $\mathcal{P}(E)$, is defined by

$$\mathcal{P}(E) = TV(\chi_E).$$

Moreover, if E has a Lipschitz boundary, we have

$$\mathcal{P}(E) = \mathcal{H}^1(\partial E).$$

Let us denote $E_\lambda = \{x \in \Omega \mid u(x) < \lambda\}$, then, if $u \in BV(\Omega)$, for almost all $\lambda \in \mathbb{R}$, E_λ has finite perimeter and the *Coarea formula* gives

$$TV(u) = \int_{-\infty}^{+\infty} \mathcal{P}(E_\lambda) d\lambda. \quad (7.3)$$

7.2.2 Presentation of the model

In order to give a sense to Equation (7.1) let us first suppose that u is smooth and integrate (7.1) on the level sets E_λ . Thus we have

$$\int_{E_\lambda} \frac{\partial u}{\partial t} dx = \int_{E_\lambda} \operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right) dx.$$

By the GAUSS-GREEN formula and as $\frac{\nabla u}{|\nabla u|}$ is the exterior normal η to the level lines ∂E_λ , we obtain

$$\int_{E_\lambda} \frac{\partial u}{\partial t} dx = \int_{\partial E_\lambda} \frac{\nabla u}{|\nabla u|} \cdot \eta d\mathcal{H}^1 = \mathcal{H}^1(\partial E_\lambda).$$

Now we take $u \in BV(\Omega)$ and as, for almost all $\lambda \in \mathbb{R}$, E_λ is a set of finite perimeter, we will consider that the minimization of the Total Variation of u leads us to an evolution law for u , which is given for almost all $\lambda \in \mathbb{R}$, by (7.2):

$$\int_{E_\lambda} \frac{\partial u}{\partial t} dx = \mathcal{P}(E_\lambda).$$

Before going further in the analysis of these equations let us show how (7.2) will act on image data.

Suppose we keep the variation of the gray level constant on each connected component of the level sets. For images this seems quite natural, now consider the following example:

Take an image u with two discs with $u(x) = 1$, on a background where $u(x) = 0$. Let D_1 and D_2 be the discs with radii r_1 and r_2 , and $r_1 < r_2$. In Figure 7.1 we show the initial function $u(0, x)$ and the result after time t . Indeed, applying our model we have

$$\forall x \in D_1 \quad : \quad u(t, x) = u(x) + t \frac{\mathcal{H}^1(\partial D_1)}{|D_1|} = \frac{2t}{r_1},$$

$$\forall x \in D_2 \quad : \quad u(t, x) = u(x) + t \frac{\mathcal{H}^1(\partial D_2)}{|D_2|} = \frac{2t}{r_2},$$

$$\forall x \in \Omega \setminus (D_1 \cup D_2) \quad : \quad u(t, x) = u(x) = 1.$$

We obtain a selective diminution of contrast, indeed, because of the higher ratio $\mathcal{H}^1(\partial D_1)/|D_1|$, the value of the small disc tends faster to 0 than that of the large disc.

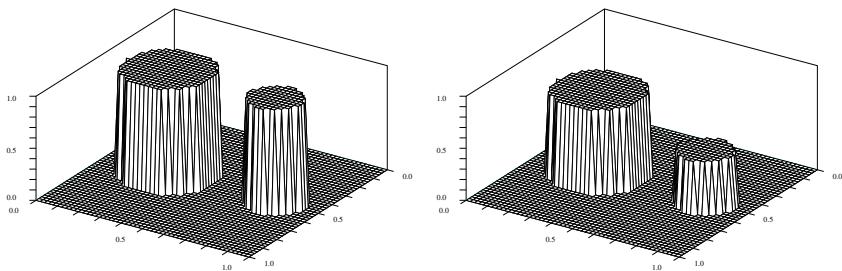


Figure 7.1. On the left the initial data $u(0, x)$, on the right the result after time t , $u(t, x)$.

Eventually the small disc will vanish before the large one. This corresponds to the second way of decreasing the Total Variation of the image u . The first way is to decrease the radii of the circles as performed by the morphological equations of image processing, see for example [9] and [90]. We will illustrate this in Section 7.4.

7.2.3 Validity of the model for $u \in BV(\Omega) \cap C^\infty(\Omega)$

In this section we will study more precisely the behavior of Equation (7.2) for regular data u . Suppose that $u \in BV(\Omega) \cap C^\infty(\Omega)$, by using almost the same idea than in the preceding section, i.e. by supposing $\frac{\partial u}{\partial t}$ to be constant on each level line $\partial E_\lambda = \{x \mid u(x) = \lambda\}$, we show that the Total Variation decreases if we realize (7.2) for such u .

Proposition: Let $u \in BV(\Omega) \cap C^\infty(\Omega)$, evolve according to Equation (7.2). Moreover assume that $\frac{\partial u}{\partial t}$ is constant on each level line ∂E_λ , then

$$TV(u(t, x)) \leq TV(u(x)),$$

where $TV(u(x))$ is the Total Variation of $u(x)$.

In order to prove this proposition we need the following lemma where we look at the variation of the perimeter of the level sets when u is a smooth function.

Lemma: If u is a smooth function then, for almost all $\lambda \in \mathbb{R}$,

$$\frac{d}{d\lambda} \mathcal{H}^1(\partial E_\lambda) = \int_{\partial E_\lambda} \frac{\text{curv}(u)}{|\nabla u|} d\mathcal{H}^1.$$

Proof of the lemma:

As u is smooth, Equation (7.2) may be written as

$$\int_{E_\lambda} \frac{\partial u}{\partial t} dx = \mathcal{H}^1(\partial E_\lambda) .$$

Moreover SARD's theorem tells us that, for almost all $\lambda \in \mathbb{R}$

$$\{|\nabla u| = 0\} \cap u^{-1}(\lambda) = \emptyset . \quad (7.4)$$

Choosing λ such that (7.4) is realized and $h > 0$ small enough, we have

$$u \left(x + h \frac{\nabla u}{|\nabla u|^2} \right) = u(x) + h + o(h) .$$

Thus, if $x(s)$ is a parameterization of ∂E_λ , we may formally consider that

$$\tilde{x}(s) = x(s) + h \frac{\nabla u}{|\nabla u|^2}$$

is a parameterization of $\partial E_{\lambda+h}$.

Now, for convenience, let us write $x = (x_1, x_2)$ and use the fact that

$$\partial E_\lambda = \{(x_1, x_2) \mid u(x_1, x_2) = \lambda\} .$$

We have

$$u_{x_1} x'_1(s) + u_{x_2} x'_2(s) = 0 .$$

Then, if we parameterize ∂E_λ by its Euclidean arc length in order to obtain $x'_1(s)^2 + x'_2(s)^2 = 1$, we have, by taking care of the orientation of ∂E_λ ,

$$x'_1(s) = -\frac{u_{x_2}}{|\nabla u|} \quad x'_2(s) = \frac{u_{x_1}}{|\nabla u|} .$$

Therefore, if we denote $V = \frac{u_{x_1}}{|\nabla u|^2}$ $W = \frac{u_{x_2}}{|\nabla u|^2}$,
we obtain

$$\begin{aligned} & (\widetilde{x_1}'(s))^2 + (\widetilde{x_2}'(s))^2 \\ &= 1 + \frac{2h}{|\nabla u|^2} \left(V_{x_1} u_{x_2}^2 - (V_{x_2} + W_{x_1}) u_{x_1} u_{x_2} + W_{x_2} u_{x_1}^2 \right) + o(h) \\ &= 1 + \frac{2h}{|\nabla u|^4} \left(u_{x_1 x_1} u_{x_2}^2 - 2u_{x_1 x_2} u_{x_1} u_{x_2} + u_{x_2 x_2} u_{x_1}^2 \right) + o(h) \\ &= 1 + 2h \frac{\text{curv}(u)}{|\nabla u|} + o(h) . \end{aligned}$$

Thus

$$\begin{aligned}\mathcal{H}^1(\partial E_{\lambda+h}) &= \int_0^{\mathcal{H}^1(\partial E_\lambda)} \sqrt{(\widetilde{x_1}'(s))^2 + (\widetilde{x_2}'(s))^2} ds \\ &= \mathcal{H}^1(\partial E_\lambda) + h \int_{\partial E_\lambda} \frac{\text{curv}(u)}{|\nabla u|} d\mathcal{H}^1 + o(h),\end{aligned}$$

and finally

$$\frac{d}{d\lambda} \mathcal{H}^1(\partial E_\lambda) = \int_{\partial E_\lambda} \frac{\text{curv}(u)}{|\nabla u|} d\mathcal{H}^1.$$

Proof of the proposition:

Choose λ as in the proof of the lemma and $h > 0$ small enough. As $E_\lambda = \{x | u(x) < \lambda\}$, we have, for almost all λ ,

$$\begin{aligned}\mathcal{H}^1(\partial E_{\lambda+h}) - \mathcal{H}^1(\partial E_{\lambda-h}) &= \int_{E_{\lambda+h} \setminus E_{\lambda-h}} \frac{\partial u}{\partial t} dx \\ &= \int_{\lambda-h}^{\lambda+h} \left(\int_{\partial E_\nu} \frac{\partial u}{\partial t} \frac{1}{|\nabla u|} d\mathcal{H}^1 \right) d\nu \\ &= 2h \int_{\partial E_\lambda} \frac{\partial u}{\partial t} \frac{1}{|\nabla u|} d\mathcal{H}^1 + o(h).\end{aligned}$$

Thus, as we assumed that $\frac{\partial u}{\partial t} = C_\lambda$ on ∂E_λ , we obtain

$$C_\lambda = \frac{d}{d\lambda} \mathcal{H}^1(\partial E_\lambda) = \frac{\int_{\partial E_\lambda} \frac{\text{curv}(u)}{|\nabla u|} d\mathcal{H}^1}{\int_{\partial E_\lambda} \frac{1}{|\nabla u|} d\mathcal{H}^1}.$$

Let us define $E_{t,\lambda} = \{x | u(t, x) < \lambda\}$ and $\partial E_{t,\lambda} = \{x | u(t, x) = \lambda\}$. We will now look for $\mathcal{H}^1(\partial E_{t,\lambda})$.

Let $x \in \partial E_\lambda$, $t > 0$ small enough and y such that

$$\vec{xy} = -t \frac{\partial u}{\partial t}(x) \frac{\nabla u}{|\nabla u|^2} = -t C_\lambda \frac{\nabla u}{|\nabla u|^2}.$$

$$\begin{aligned} \text{We have } u(t, y) &= u(y) + t \frac{\partial u}{\partial t}(y) + o(t) \\ &= u(x) - t \frac{\partial u}{\partial t}(x) + t \frac{\partial u}{\partial t}(y) + o(t) = u(x) + o(t). \end{aligned}$$

Then, if $x(s)$ is a parameterization of ∂E_λ , we may take

$$y(s) = x(s) - t C_\lambda \frac{\nabla u}{|\nabla u|^2}$$

as a parameterization for $\partial E_{t,\lambda}$. Therefore, as in the proof of the lemma, we obtain

$$\mathcal{H}^1(\partial E_{t,\lambda}) = \mathcal{H}^1(\partial E_\lambda) - t C_\lambda \int_{\partial E_\lambda} \frac{\text{curv}(u)}{|\nabla u|} d\mathcal{H}^1 + o(t),$$

and, by using the value of C_λ ,

$$\mathcal{H}^1(\partial E_{t,\lambda}) = \mathcal{H}^1(\partial E_\lambda) - t \frac{\left(\int_{\partial E_\lambda} \frac{\text{curv}(u)}{|\nabla u|} d\mathcal{H}^1 \right)^2}{\int_{\partial E_\lambda} \frac{1}{|\nabla u|} d\mathcal{H}^1} + o(t).$$

Thus, applying the Coarea formula (7.3), we obtain

$$TV(u(t, x)) \leq TV(u(x)).$$

This finishes the proof of the TV minimizing property of (7.2) for a regular u . In the next section we will introduce an algorithm inspired from the above but appropriate for digital image data.

7.3 A practical algorithm

In this section we will focus on digital image data. Usually image data is defined on a discrete grid and the pixels take a finite number of values, the gray values. We will apply the ideas of Section 7.2.2 and suppose that the variation of the gray level ($\partial u / \partial t$) on the connected components of the level sets is constant.

But first, we will introduce our main tool for image representation. It is the FLST which allows us to obtain a fast algorithm.

7.3.1 The Fast Level Sets Transform

For a complete presentation of the FLST see [376] or [367]. We recall also that the FLST is an essential tool for the image registration method proposed in the chapter “Image Registration”.

Let us introduce some notations, for an image u we denote

$$X^\lambda = \{x \mid u(x) \geq \lambda\}$$

$$X_\mu = \{x \mid u(x) \leq \mu\}$$

the upper level set of value λ and the lower level set of value μ . The data $(X^\lambda, \lambda \in \mathbb{R})$, or $(X_\mu, \mu \in \mathbb{R})$, is sufficient to reconstruct the image u , as

$$u(x) = \sup\{\lambda \mid x \in X^\lambda\} = \inf\{\mu \mid x \in X_\mu\}.$$

As a connected component of a level set can have holes, *shapes* are introduced. A shape is a connected component of a level set where the holes are filled.

The Fast Level Sets Transform decomposes an image into its shapes and constructs a tree structure to represent the inclusions. In [376] it is shown that this yields a non redundant and complete representation of the image (see Figure 7.2). The output of an FLST module ([202]) gives

- the family of shapes (\mathcal{T}) arranged in a tree structure (if $C \subset F$, F is the parent of C).
- for each shape its area, its boundary, the gray value λ and if it is a connected component of an upper level set ($\geq \lambda$) or a lower level set ($\leq \lambda$).

Note: In the tree of Figure 7.2 we have given a name to each shape, moreover we indicate if it is an upper or lower set and the associated gray value.

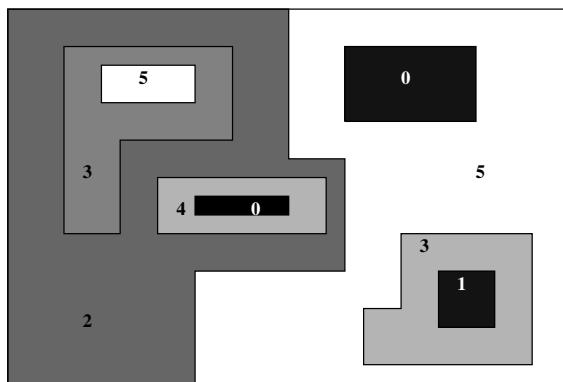


Figure 7.2. On the top the image and its FLST-tree on the bottom.

The interest of introducing this new notion of shape is to formalize the natural idea that level lines do not cross. We will present, in quite detail, the FLST of a bilinear interpolation of the array of values of the image. In that case, the image is modeled as a continuous function, and the level lines are defined as the connected components of isolevel sets. This model is appropriate for our task: the level lines are smoother than if the image were considered constant on each pixel and less affected by quantization. This gives a better value for the perimeter of a level line.

Bilinear interpolation has the advantage of topological simplicity: the singular points are on the initial grid and at a finite number of saddle points. In a square delimited by four adjacent points in the initial grid, which we will call from now on a *Qpixel*, the level lines are hyperbolae and the saddle point is their common center of symmetry, provided it is included in the square.

The problem is that the number of level lines is infinite for a continuous function, so that we must represent the image with its level lines at a finite number of levels.

The output of the algorithm will be called TBLL, for Tree of Bilinear Level Lines. Our algorithm proceeds in two steps: first it extracts the fundamental TBLL, corresponding to level lines passing through points on the initial grid or through saddle points; from this fundamental TBLL, the TBLL for any finite number of levels can be extracted.

We first find the *Qpixels* containing a saddle point, and the corresponding saddle value. It is the case when the maximum of values at diagonally opposed pixels is strictly less than the minimum of the other two values, see Figure 7.3. In the algorithm we describe, a *point* is either a center of pixel or a saddle point. Each has an associated value, thus we store in memory two images: the values at centers of pixels and the values at saddle points.

We associate to each center of pixel P a shape S_P , the smallest shape containing P . Initially, it is NULL.

We scan all centers of pixels, and each time we meet a local extremum P (comparison with 4-neighbors) at level λ , we perform the following steps:

1. Initialize a list \mathcal{P} of points to \emptyset and of neighbor points \mathcal{N} to $\{P\}$.
2. While $\mathcal{N}_\lambda \neq \emptyset$, \mathcal{N}_λ being the set of points of \mathcal{N} at level λ , remove \mathcal{N}_λ from \mathcal{N} and append it to \mathcal{P} , and add to \mathcal{N} the neighbors of \mathcal{N}_λ not already in \mathcal{P} .
3. If the set of points in \mathcal{P} has no hole and the points in \mathcal{N} are all at level $< \lambda$ or all at level $> \lambda$, create a new shape with associated points \mathcal{P} ; otherwise, put all points of image stored in \mathcal{P} to level λ and exit.
4. For each point $Q \in \mathcal{P}$, if S_Q is NULL, put it to the new shape. Otherwise, follow up the tree starting from S_Q , and put the resulting shape as child of the new shape.
5. Set λ to the closest level of points in \mathcal{N} and go back to step (2).

Let us precise what we mean by neighbor: the neighbors of a center of pixel P are its 4-neighbors and the saddle points in the *Qpixels* whose one corner is

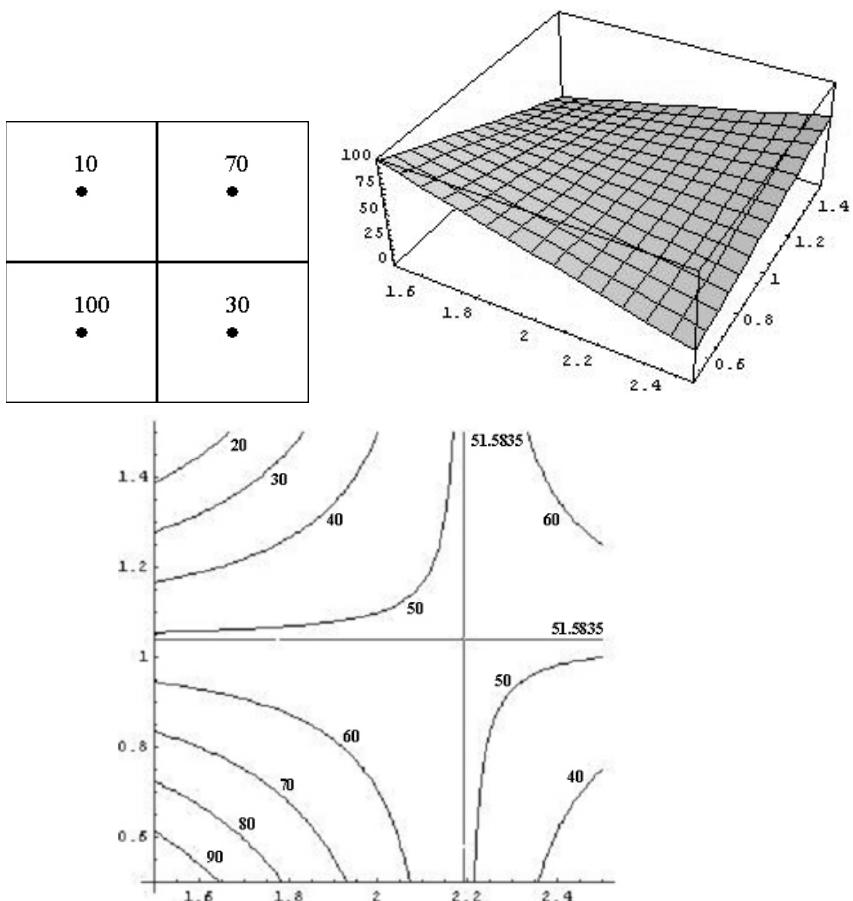


Figure 7.3. Top: example of a Qpixel (left) and its bilinear interpolation (right). Bottom: Level lines (pieces of hyperbolae) inside the Qpixel. Level lines are computed from level 20 to level 90 with quantization step 10. We get a saddle point for gray level 51.5835.

P ; the neighbors of a saddle point Q are the centers of pixels being corners of its containing *Qpixel*.

The number of holes is computed locally, with the following rule: two centers of pixels are connected if they are 4-neighbors, or if they are diagonally opposed in the *Qpixel* and at least one of the other two corners or the saddle point inside the *Qpixel* (provided there is one) is in the set. Then the number of holes is computed by counting patterns of connected neighbors, in a manner similar to that exposed in [291, 292].

We can notice that it is important to order the points in \mathcal{N} by their level. This can be done with a balanced binary tree, similar to the one used in *heap sort* [466].

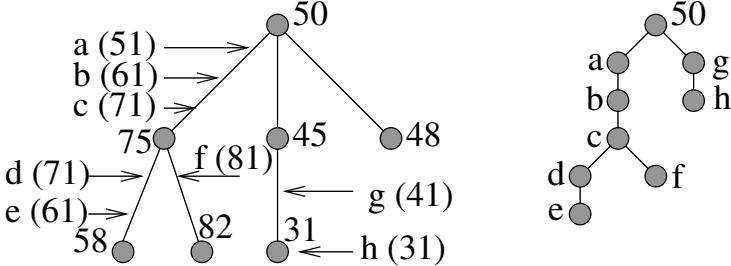


Figure 7.4. Computing the TBLL of quantization levels $\{1, 11, 21, \dots\}$ from the fundamental TBLL. Left: fundamental TBLL, showing associated levels. Right: the resulting TBLL is obtained by sampling of the fundamental TBLL.

After the image is scanned, its gray levels have been changed and it becomes uniform. The resulting constant level is the one of the root of the tree.

From the knowledge of the fundamental TBLL and a given quantization, it is direct to compute the resulting TBLL. For each shape S in the fundamental TBLL, find the levels of the quantization comprised between the level of S (strictly) and the level of its parent S' in the fundamental TBLL. Create a shape for each one; the level line passes through the same chain of adjacent *Qpixels* as S' .

For each shape created in this manner, their order in the TBLL is the same as the one of their least greater shapes in the fundamental TBLL. We could consider that the TBLL is a sampling of the fundamental TBLL, see Figure 7.4.

7.3.2 Total Variation Minimization with the FLST

We want to reduce the Total Variation of an image u , by removing the shapes with too high global curvature. Instead of using the connected components of the level sets, we will use the shapes provided by the FLST.

In order to treat in a symmetric way black forms on white background and white forms on black background, we have to construct $T_t(u)$ by using alternatively lower level sets and upper level sets. We will define this operator by using the FLST-tree of an image.

Let u be an image and $E_1, E_2, \dots, E_N = E_R$ the shapes of the FLST arranged in inverse level-order (e.g. in Figure 7.2 $A, B, C, D, \dots, H, K, E_R$), E_R is the root. Denote by λ_i the gray value associated to the shape E_i and by $E_i(\leq \lambda_i)$, resp. $E_i(\geq \lambda_i)$, the fact that E_i has been obtained from a lower, resp. upper, level set. As before, let Ω be the domain of the image and

$$\mathcal{P}(E_i) = \mathcal{H}^1(\partial E_i \setminus \partial \Omega).$$

Then the Coarea formula (7.3) may be written

$$TV(u) = \sum_{i=1}^{N-1} \mathcal{P}(E_i) |\lambda_i - \lambda_i^p|, \quad (7.5)$$

where λ_i^p is the gray value associated to E_i^p , the parent shape of E_i .

In order to implement (7.2) or, more precisely, to implement on each shape E_i

$$\int_{E_i} \frac{\partial u}{\partial t} dx = \mathcal{P}(E_i),$$

we will build a sequence (u^m) of images such that

$$\begin{cases} u^0 &= u \\ u^m &= T_t(u^{m-1}) = (T_t)^m(u), \quad m \geq 1. \end{cases}$$

Definition of the operator T_t :

For the original image u we construct a finite sequence $(u_0 = u, u_1, u_2, \dots, u_{N-1})$, where N is the number of shapes in the FLST-tree. Then we set

$$T_t(u) = u_{N-1}.$$

If u_{i-1} is known we obtain u_i as follows:

- if $x \in E_i$ and $E_i(\leq \lambda_i)$: $u_i(x) = \inf \left\{ u_{i-1}(x) + t \frac{\mathcal{P}(E_i)}{|E_i|}, \lambda_i^p \right\},$
- if $x \in E_i$ and $E_i(\geq \lambda_i)$: $u_i(x) = \sup \left\{ u_{i-1}(x) - t \frac{\mathcal{P}(E_i)}{|E_i|}, \lambda_i^p \right\},$
- if $x \notin E_i$: $u_i(x) = u_{i-1}(x).$

This concludes the presentation of the algorithm, we will give proofs of its properties in the following subsections.

7.3.3 TV Minimization property

In this paragraph we will prove that the total variation is decreasing when applying the preceding algorithm

Let us recall that if E_i is a lower shape ($E_i(\leq \lambda_i)$), we have $\lambda_i \leq \lambda_i^p$ and if E_i is an upper shape ($E_i(\geq \lambda_i)$), $\lambda_i \geq \lambda_i^p$, where λ_i^p is the gray value associated to E_i^p , the parent shape of E_i .

Let E_1, E_2, \dots, E_j be the terminal nodes of the tree and suppose we have already constructed u_j . The shapes being unchanged we write $\lambda_{i,j}$ and $\lambda_{i,j}^p$ for

the new gray levels at step j . We use (7.5) to obtain

$$\begin{aligned} TV(u_j) &= \sum_{i=1}^{N-1} \mathcal{P}(E_i) |\lambda_{i,j} - \lambda_{i,j}^p| \\ &= \sum_{i=1}^j \mathcal{P}(E_i) |\lambda_{i,j} - \lambda_i^p| + \sum_{i=j+1}^{N-1} \mathcal{P}(E_i) |\lambda_i - \lambda_i^p|. \end{aligned}$$

Indeed, as we only change the value of u on the terminal nodes, we have for $1 \leq i < N$: $\lambda_{i,j}^p = \lambda_i^p$;

and for $j < i < N$: $\lambda_{i,j} = \lambda_i$.

The new values $\lambda_{i,j}$, for $1 \leq i \leq j$, are obtained through our construction of T_t in Section 7.3.2, they are equal to

$$\inf\{\lambda_i + t \frac{\mathcal{P}(E_i)}{|E_i|}, \lambda_i^p\} \quad \text{or to} \quad \sup\{\lambda_i - t \frac{\mathcal{P}(E_i)}{|E_i|}, \lambda_i^p\},$$

depending on the type of shape E_i (lower or upper). Thus we conclude that $TV(u_j) \leq TV(u) = TV(u_0)$.

For the general case consider a shape E_i with $1 \leq i < N$. The gray value associated to it will first be modified during the construction of u_i .

We have $|\lambda_{i,i} - \lambda_{i,i}^p| < |\lambda_{i,i-1} - \lambda_{i,i-1}^p|$
as $\lambda_{i,i}^p = \lambda_{i,i-1}^p$ and $\lambda_{i,i}$ being equal to

$$\inf\{\lambda_{i,i-1} + t \frac{\mathcal{P}(E_i)}{|E_i|}, \lambda_{i,i-1}^p\} \quad \text{or} \quad \sup\{\lambda_{i,i-1} - t \frac{\mathcal{P}(E_i)}{|E_i|}, \lambda_{i,i-1}^p\}.$$

Let us now examine how the gray value associated to the shape E_i might be modified by the remaining construction of $T_t(u)$.

During the evaluation of u_k , $i < k < N$, we encounter two cases:

- $E_i \cap E_k = \emptyset$, then $\lambda_{i,k-1} = \lambda_{i,k}$ and $\lambda_{i,k-1}^p = \lambda_{i,k}^p$.

- $E_i \subset E_k$, in this case we have

$$\lambda_{i,k} = \inf\{\lambda_{i,k-1} + t \frac{\mathcal{P}(E_k)}{|E_k|}, \lambda_{i,k-1}^p\} \quad \text{or} \quad \sup\{\lambda_{i,k-1} - t \frac{\mathcal{P}(E_k)}{|E_k|}, \lambda_{i,k-1}^p\},$$

depending on the type of shape E_k . Now, as the parent shape of E_i will also be included in E_k , the value $\lambda_{i,k-1}^p$ will be modified the same way than $\lambda_{i,k-1}$ and thus we have again that $|\lambda_{i,k} - \lambda_{i,k}^p| = |\lambda_{i,k-1} - \lambda_{i,k-1}^p|$.

At each step of the construction of the sequence (u_k) , $1 \leq k \leq N-1$, we have the same major properties:

- (i) the shapes remain unchanged,

they might disappear if $|\lambda_{i,k} - \lambda_{i,k}^p| = 0$;

- (ii) $|\lambda_{i,k-1} - \lambda_{i,k-1}^p| \leq |\lambda_{i,k} - \lambda_{i,k}^p|$.

Therefore we conclude

$$TV(T_t(u)) = TV(u_{N-1}) \leq TV(u) .$$

7.3.4 Noise removal and other properties

Let us define the curvature of a subset C of Ω by

$$\text{curv}(C) = \frac{\mathcal{P}(C)}{|C|} .$$

By construction, if the curvature of a shape C is too big then C disappears.

Moreover, the edges of well defined “large” objects are preserved, this is the main difference between our model and the Mean Curvature Motion

$$\frac{\partial u}{\partial t} = |\nabla u| \text{curv}(u) , \quad (7.6)$$

or the Erosion

$$\frac{\partial u}{\partial t} = |\nabla u| ,$$

which both lead to a minimization of the Total Variation of u by continuous curve shortening. Moreover, with (7.6) the curves are smoothed. In our model of minimization of Total Variation, a “big” square stays the same, the only change is in its gray level.

If we consider that noise corrupts isolated pixels (or small, irregular regions) all these properties make sure that the noise is removed. Moreover, as we only change the FLST-tree by removing shapes, we don’t change the edges of shapes, thus our method is edge preserving.

Notice that we have no stopping criteria, as in the usual PDE models like (MCM) and erosion. The amount of denoising depends on perceptual criteria.

The choice of the time step t controls the speed at which gray values associated to the shapes move.

The overall execution time depends mainly on the speed of the FLST, although very fast (a couple of seconds) the complexity of the tree depends on the complexity of the geometry of the image. The minimization procedure are quite forward and is obtained in fractions of seconds.

7.4 Experimental results

7.4.1 Comparison to other models and algorithms

We will demonstrate how different algorithms compare for image denoising purposes. In Figure 7.5 we present on the top left the noisy squares picture, we will apply three different techniques on this synthetic data.

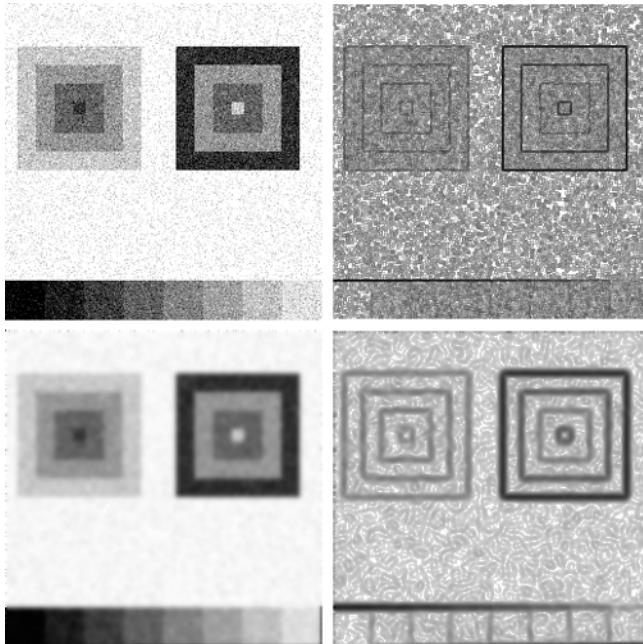


Figure 7.5. Left column: On the top initial noisy squares picture u ; on the second line the heat equation applied to u , notice the thickness of the boundaries. The right column shows the image gradient.

The first, classical, technique is to apply a convolution with a Gaussian to u : the diffusion will smooth the image and unsharpen the boundaries of objects, see Figure 7.5 second line.

In the first line of Figure 7.6, we show the result obtained by (7.6). As predicted by the theory we can observe how the level lines move where curvature is high (e.g. the corners). Thus the boundaries are much more precise and the corners have evolved.

The picture on the bottom left of Figure 7.6 is the result of our algorithm. We recover the squares with some errors on the boundary. This is due to the strong noise which destroys the boundary.

Notice that the linear diffusion and the mean curvature motion use classical numerical schemes taken from the software package Megawave2 [202].

To enhance the visibility of the results of these algorithms, we have applied an elementary edge detection and computed a discrete gradient on each of the four images. We show the norm in the right column. Notice that we have coded high gradient values with black and zero gradient with white, the inverse of the usual coding.

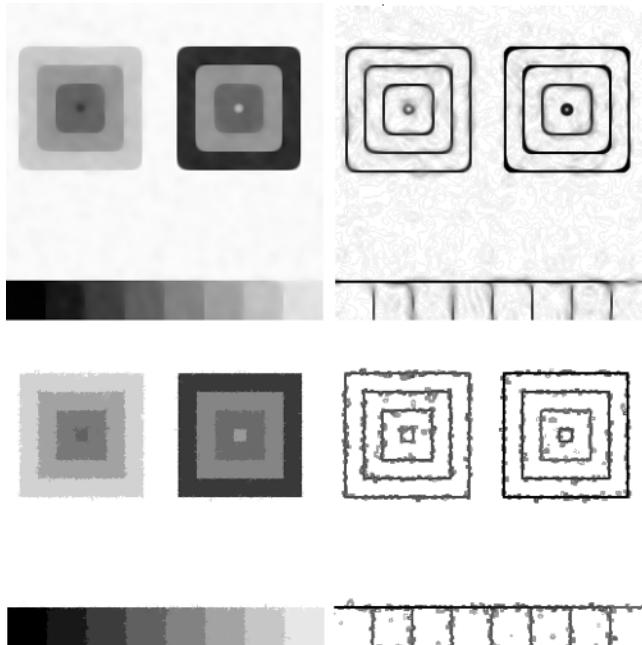


Figure 7.6. Left column top the noisy squares image filtered by the nonlinear MCM. Bottom left, our FLST TV Minimization algorithm. The right column shows again the discrete gradient.

7.4.2 Application to gray scale images

Figure 7.7 shows a satellite image with impulse noise on 10% of the pixels, the result of our algorithm is shown on the right (with $t = 1$ and 45 iterations). Using formula (7.5), one can compute easily the Total Variation for both images, we obtain $TV = 2,748,396$ for 12,901 shapes in the noisy image and $TV = 287,849$ for 355 shapes in the denoised image.

Figure 7.8 shows an image with additive Gaussian noise ($\sigma = 30$) and the result of our algorithm ($t = 0.5$ and 60 iterations).

7.4.3 Applications to color images

Manipulation of color images is usually more complicated because an adapted color space has to be chosen. The common RGB space will often give bad results: by operating on each color channel separately the final image can contain “wrong” colors, *i.e.* colors which do not exist in the original image. The literature on color is very large, we refer to the book of Stiles and Wyszecki, [581].

For our purpose of demonstrating the capacities of the TV minimization, it will be sufficient to operate in the RGB coordinate system. Indeed, using the bilinear

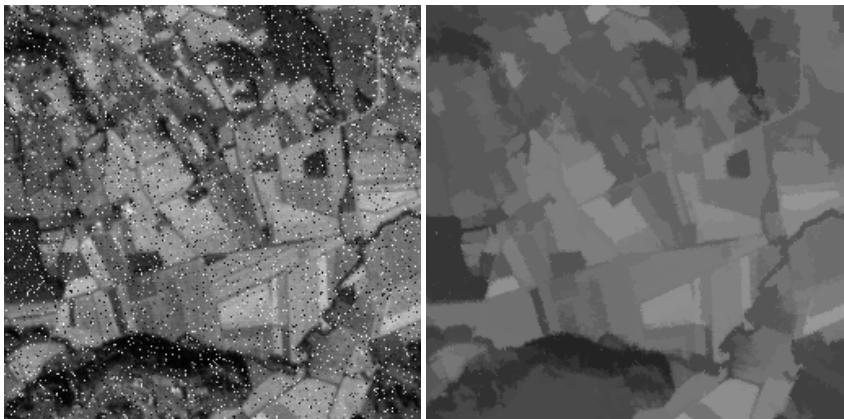


Figure 7.7. The noisy satellite image (impulse noise), with denoised image



Figure 7.8. A noisy house image (gaussian noise) and the denoised image

FLST (and with an adaptive time step) we obtain results were no visible “wrong” color can be detected (for other techniques we refer to [53]).

We minimize the total variation on the three channels R, G and B and we construct a denoised image whose quality is very good, see the Plane Image (the experiments with color images can be found on the CDROM).

An experiment on an image with additive gaussian noise shows (the Mandrill Image) that the algorithm gives also nice results on this type of image.

This leads us to infer that we can choose for the channels to be treated those which are noisy (the three for the kind of image we have chosen) and not to treat the others.



Figure 7.9. First and last taxi sequence images

7.4.4 Optical flow regularization

Let us first recall some facts about optical flow determination. The computation of the optical flow field of an image sequence has been pioneered by Horn and Schunck ([245]). Their assumption is that a point (x, y) has a constant brightness at any time z : $u(x, y, z) = u(x + \delta x, y + \delta y, z + \delta z)$. If (σ_1, σ_2) denote the components of the velocity, this leads to $D_z u = \nabla u \cdot \sigma + u_z = 0$. As this equation is scalar, it is not sufficient to determine the two components of the velocity field. This is the *aperture problem*: other conditions have to be found.

Horn and Schunck proposed to determine optical flow as the minimum of the functional

$$E(\sigma) = \int_{\mathbb{R}^2} \alpha^2 (\|\nabla \sigma_1\|^2 + \|\nabla \sigma_2\|^2) + (\nabla u \cdot \sigma + u_z)^2 dx dy,$$

where the real parameter α weighs the importance of the isotropic quadratic regularization term in the functional.

In [569], Weickert and Schnörr propose a functional which leads to an anisotropic spatio-temporal regularization of the optical flow. Their method initially uses the whole sequence of data, and is thus computationally more expensive than Horn and Schunck's method.

In Figure 7.9, we show the first and the last image of a Hamburg Taxi sequence. On this data we applied first the Horn and Schunck algorithm. The first line of Figure 7.10 shows the result for $\alpha^2 = 49$ on the left and $\alpha^2 = 500$ on the right (1000 iterations each time). One notices the very unregular flow estimation, but also that for large α (e.g. 500), the quadratic regularization term yields pretty large blobs, especially the lower left vehicle.

We applied our method directly on the almost unregularized Horn and Schunck result ($\alpha^2 = 49$), see the lower left image in Figure 7.10. This can be compared to the result obtained by the Weickert and Schnörr method, which is shown in the lower right image. Notice that in both results the size of the blob, obtained for the lower left vehicle, is closer to the vehicle size than the result obtained by Horn and Schunck with $\alpha^2 = 500$.

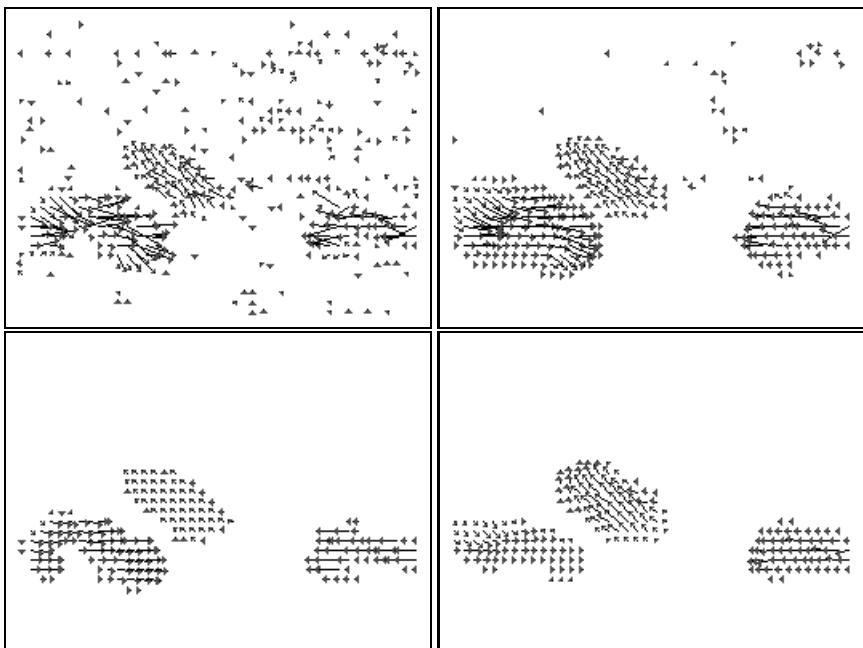


Figure 7.10. Optical flow between image 13 and image 14: Horn and Schunk with different parameters on the top, below left, regularization with TV minimization obtained from the top left image. On the lower right, the result of the Weickert-Schnörr method (see text).

Acknowledgments: The color and optical flow experiments have been done by Florent Ranchin (ranchin@ceremade.dauphine.fr) at Ceremade, University Paris Dauphine

8

Morphological Global Reconstruction and Levelings: Lattice and PDE Approaches

Petros Maragos

Abstract

This chapter begins with analyzing the theoretical connections between levelings on lattices and scale-space erosions on reference semilattices. They both represent large classes of self-dual morphological reconstruction operators that exhibit both local computation and global constraints. Such operators are useful in numerous image analysis and vision tasks including edge-preserving multiscale smoothing, image simplification, feature and object detection, segmentation, shape, texture and motion analysis. Previous definitions and constructions of levelings were either discrete or continuous using a PDE. We bridge this gap by introducing generalized levelings based on triphase operators that switch among three phases, one of which is a global constraint. The triphase operators include as special cases useful classes of semilattice erosions. Algebraically, levelings are created as limits of iterated or multiscale triphase operators. The subclass of multiscale geodesic triphase operators obeys a semigroup, which we exploit to find PDEs that can generate geodesic levelings and continuous-scale semilattice erosions. We discuss theoretical aspects of these PDEs, propose discrete algorithms for their numerical solution which converge as iterations of triphase operators, and provide insights via image experiments.

8.1 Introduction

Nonlinear scale-space approaches that are based on morphological operators are useful for edge-preserving multiscale smoothing, image simplification, geometric feature detection, segmentation, shape, texture and motion analysis, and object recognition.

The theory and implementations behind the standard multiscale morphological filters evolved first [347, 467, 333] from a geometric viewpoint that focused on shape-size analysis and an algebraic viewpoint that was based on set theory, level sets and min-max filtering. During the previous decade both the algebraic and geometric aspects of morphology were generalized and improved, by extending its algebra using the theory of complete lattices [468, 233] and by modeling the dynamics and geometry of multiscale morphology using PDEs and curve evolution [9, 70, 459, 336]. The simplest morphological smoothers are translation-invariant (TI) Minkowski openings and closings, i.e., compositions of Minkowski erosions and dilations by compact disk-like sets. These nonlinear smoothers preserve well the edges of the remaining image signal parts but may blur the boundaries of their supports at places where the structuring element cannot fit. The opening increasingly reconstructs some of the image parts lost via erosion, whereas the closing reconstructs by shrinking the parts added via dilation. This reconstruction is *local* and extends only up to the scale of these filters.

A much more powerful class of filters are the *reconstruction* openings and closings which, starting from a *reference* image consisting of several parts and a *marker* (initial seed) inside some of these parts, can reconstruct whole objects with exact preservation of their boundaries and edges [554, 455]. In this *global* reconstruction process they simplify the original image by completely eliminating smaller objects inside which the marker cannot fit. The reference image plays the role of a *global constraint*. One disadvantage of both the simple as well as the reconstruction openings/closings is that they are not self-dual and hence they treat asymmetrically the image foreground vs. background or the bright vs. dark objects. A recent solution to this asymmetry problem came from the development of a more general powerful class of self-dual morphological filters, the *levelings* which include as special cases the reconstruction openings and closings. The levelings were introduced by Meyer [366] for digital spaces and further studied in [348, 469]. They possess many useful algebraic scale-space properties, as explored in [370]. Maragos & Meyer [369] extended them to continuous spaces by generating levelings with the following nonlinear PDE:

$$\begin{aligned}\partial u(x, y, t)/\partial t &= -\text{sign}(u - r)\|\nabla u\| \\ u(x, y, 0) &= f(x, y)\end{aligned}\tag{8.1}$$

where $u(x, y, t)$ is the scale-space function, $r(x, y)$ is the reference image and $f(x, y)$ is a marker. At scale-space points where $u > r$ (resp., $u < r$), the above PDE generates multiscale erosions (resp., dilations) by disks. The leveling $\Lambda(f|r)$ of r w.r.t. f is produced when $t \rightarrow \infty$. In [369, 370] it was explained that, if $f \leq r$ (resp., $f \geq r$), the leveling is a reconstruction opening (resp., closing). Examples are shown in Fig. 8.1.

A relatively new algebraic approach to self-dual morphology was developed by Keshet [269] and Heijmans & Keshet [234, 235] based not on complete lattices but on inf-semilattices. Specifically, by using self-dual partial orderings the image space becomes an inf-semilattice on which self-dual erosion operators can

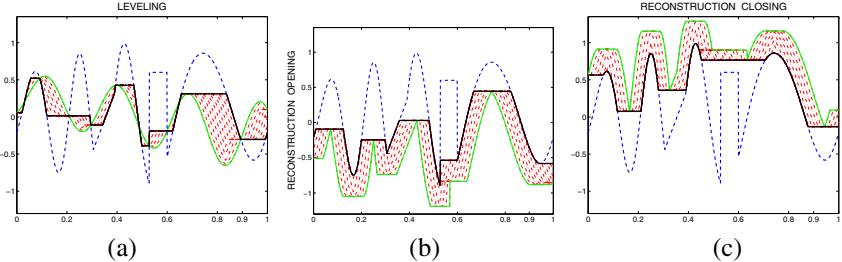


Figure 8.1. Evolutions of 1D leveling PDE $u_t = -\text{sign}(u - r)|u_x|$ for 3 different markers $u(x, 0) = f(x)$. Each figure shows the reference signal r (dash line), the marker f (thin solid line), its evolutions $u(x, t)$ (thin dashdot line) at $t = n25\Delta t$, $n = 1, 2, \dots$, and the leveling $u(x, \infty)$ (thick solid line). The 3 markers f were: (a) Arbitrary. (b) An erosion of r minus a constant; hence, the leveling is a reconstruction opening. (c) A dilation of r plus a constant; hence the leveling is a reconstruction closing.

be defined that have many interesting properties and promising applications in nonlinear image analysis.

This chapter continues with an intuitive discussion of multiscale levelings and their interpretation using level sets. Then, after a brief background on lattice operators, we analyze algebraically multiscale *triphasic* operators, whose limits are levelings. Special cases of these triphasic operators are semilattice erosions. The semigroup of *geodesic* triphasic operators is emphasized. Afterwards, we focus on PDEs that can generate both geodesic levelings and TI semilattice self-dual erosions. We discuss theoretical aspects of these PDEs, propose algorithms for their numerical solution which converge as iterations of discrete triphasic operators, and provide insights via image experiments. The proofs of all propositions and theorems of this chapter can be found in a recent paper by Maragos [335].

8.2 Multiscale Levelings and Level Sets

Consider a reference image r and a leveling Λ . If we can produce various markers f_i , $i = 1, 2, 3, \dots$, that are related to some increasing scale parameter i , let us construct the levelings $g_i = \Lambda(f_i | g_{i-1})$, $i = 1, 2, 3, \dots$, with $g_0 = r$. The signals g_i constitute a hierarchy of *multiscale levelings* possessing the causality property that g_j is a leveling of g_i for $j > i$. One way to construct such multiscale levelings is to use a sequence of multiscale markers obtained from sampling a Gaussian scale-space. As shown in Fig. 8.2, the image edges and boundaries which have been blurred and shifted by the Gaussian scale-space are better preserved across scales by the multiscale levelings.

The main approach in mathematical morphology to extend set operators to image function operators is using level sets and threshold superposition [467, 333,

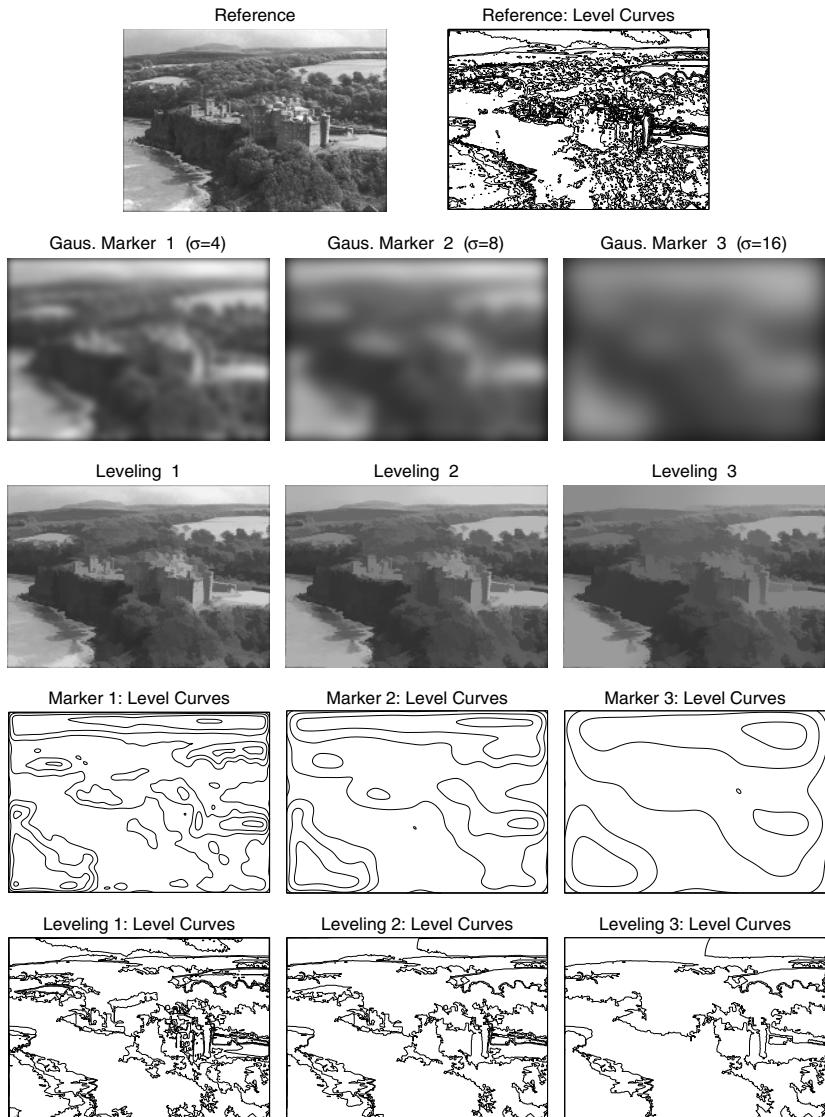


Figure 8.2. Multiscale image levelings $u(x, y, \infty)$ generated by the PDE $u_t = -\text{sign}(u - r) \|\nabla u\|$. The markers $u(x, y, 0) = f(x, y)$ were obtained by convolving the reference image with 2D Gaussians of standard deviations $\sigma_1 = 4$, $\sigma_2 = 8$, $\sigma_3 = 16$. At each scale σ_i as reference r was used the leveling of the previous scale σ_{i-1} . The last two rows show level curves of the markers and the corresponding levelings (6 level curves for each image).

233]. Specifically, if

$$X_h(f) \triangleq \{x : f(x) \geq h\}, \quad -\infty < h < \infty \quad (8.2)$$

are the upper *level sets* of a real image function f , and we are given an increasing set operator Ψ , then we can construct a so-called *flat operator* $\psi(f) = \sup\{h : x \in \Psi(X_h(f))\}$ via *threshold superposition* of the outputs of the set operator acting on all input level sets. The flat operator ψ can process both binary and graylevel images. The levelings produced by the PDE (8.1) are flat operators and hence they can be constructed by their set counterparts via threshold superposition. Thus,

$$\Lambda(f|r) = \sup\{h : x \in \Lambda(X_h(f)|X_h(r))\} \quad (8.3)$$

The set equivalent of the PDE (8.1) is a curve evolution that propagates the level curves of u with normal speed ± 1 whose sign corresponds to $\text{sign}(r-u)$. Namely, when $u > r$ ($u < r$), the level curves are moved toward (opposite) the direction of the gradient of u . In the limit, the curves will converge to the level curves of the leveling. The level curves of the multiscale levelings are shown in Fig. 8.2 to preserve the global boundaries of image regions much more accurately than the multiscale Gaussian convolutions.

8.3 Multiscale Image Operators on Lattices

A poset is any set equipped with a partial ordering \leq . The supremum (\vee) and infimum (\wedge) of any subset of a poset is its lowest upper bound and greatest lower bound, respectively; both are unique if they exist. A poset is called a (sup-) inf-semilattice if the (supremum) infimum of any finite collection of its elements exists. A (sup-) inf-semilattice is called complete if the (supremum) infimum of arbitrary collections of its elements exist. A poset is called a (complete) lattice if it is simultaneously a (complete) sup- and an inf-semilattice. An operator ψ on a complete lattice is called: *increasing* if it preserves the partial ordering [$f \leq g \implies \psi(f) \leq \psi(g)$]; *idempotent* if $\psi^2 = \psi$; *antiextensive* (resp., *extensive*) if $\psi(f) \leq f$ (resp., $f \leq \psi(f)$). An operator ε (resp., δ) on a complete inf-semilattice (resp., sup-semilattice) is called an *erosion* (resp., *dilation*) if it distributes over the infimum (resp., supremum) of any collection of lattice elements; namely $\delta(\bigvee_i f_i) = \bigvee_i \delta(f_i)$ and $\varepsilon(\bigwedge_i f_i) = \bigwedge_i \varepsilon(f_i)$. A lattice operator is called an *opening* (resp., *closing*) if it is increasing, idempotent, and antiextensive (resp., extensive). A *negation* is a bijective operator $\nu \neq \text{id}$ such that both ν and ν^{-1} are either decreasing or increasing and $\nu^2 = \text{id}$, where id is the identity. An operator is called *self-dual* if it commutes with a negation.

In this chapter, the image space is the collection of signals defined on a continuous or discrete domain \mathbb{E} and assuming values in \mathbb{V} , where $\mathbb{E} = \mathbb{R}^m$ or \mathbb{Z}^m , $m = 1, 2, \dots$, and $\mathbb{V} \subseteq \overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$. The value set \mathbb{V} is equipped with some partial ordering that makes it a complete lattice or inf-semilattice. This lattice structure is inherited by the image space by extending the partial order of \mathbb{V} to signals pointwise.

8.3.1 Multiscale Operators on Complete Lattices

Classical lattice-based morphology [233, 468] uses as image space the *complete lattice* \mathcal{L} of signals $f : \mathbb{E} \rightarrow \mathbb{V}$ with values in $\mathbb{V} = \overline{\mathbb{R}}$ or $\overline{\mathbb{Z}}$. In \mathcal{L} the signal ordering is defined by $f \leq g \Leftrightarrow f(x) \leq g(x), \forall x$, and the signal infimum and supremum are defined by $(\bigwedge_i f_i)(x) = \sup_i f_i(x)$ and $(\bigvee_i f_i)(x) = \inf_i f_i(x)$. Assume first $\mathbb{E} = \mathbb{R}^m$. Let $B = \{x : \|x\| \leq 1\}$ denote the unit-radius ball in \mathbb{R}^m w.r.t. the Euclidean metric $\|\cdot\|$ and let $tB = \{tb : b \in B\}$ be its version at scale $t \geq 0$. The simplest multiscale dilation/erosion on \mathcal{L} are the *Minkowski flat dilation/erosion* of an image f by tB :

$$\begin{aligned}\delta_B^t(f)(x) &\triangleq (f \oplus tB)(x) = \bigvee_{a \in tB} f(x - a) \\ \varepsilon_B^t(f)(x) &\triangleq (f \ominus tB)(x) = \bigwedge_{a \in tB} f(x + a)\end{aligned}\quad (8.4)$$

We shall also need the *multiscale conditional dilation and erosion* of a *marker* ('seed') image f given a *reference* ('mask') image r :

$$\delta_{tB}(f|r) \triangleq (f \oplus tB) \wedge r, \quad \varepsilon_{tB}(f|r) \triangleq (f \ominus tB) \vee r \quad (8.5)$$

Iterating the unit-scale conditional dilation (erosion) yields the *conditional reconstruction opening (closing)* of r from f :

$$\begin{aligned}\rho_B^-(f|r) &\triangleq \delta_B^\infty(f|r) = \bigvee_{n \geq 1} \delta_B^n(f|r) \\ \rho_B^+(f|r) &\triangleq \varepsilon_B^\infty(f|r) = \bigwedge_{n \geq 1} \varepsilon_B^n(f|r)\end{aligned}\quad (8.6)$$

where, for any operator ψ and any positive integer n , ψ^n denotes the n -fold composition of ψ with itself and $\psi^\infty = \lim_{n \rightarrow \infty} \psi^n$ if the limit exists.

Another important pair is the geodesic dilation and erosion. First we define them for sets $X \subseteq \mathbb{E}$. Let $R \subseteq \mathbb{E}$ be a *reference* (mask) set and consider its *geodesic metric* $d_R(x, y)$ equal to the length of the geodesic path connecting the points x and y inside R . If $B_R(x, t) = \{p \in R : d_R(x, p) \leq t\}$ is the geodesic closed ball with center x and radius $t \geq 0$, then the multiscale geodesic set dilation of X given R is defined by $\Delta^t(X|R) \triangleq \bigcup_{p \in X} B_R(p, t)$. By using threshold decomposition and synthesis of an image f from its level sets $X_h(f)$ we can synthesize a flat geodesic dilation for images by using as generator its set counterpart. Then, a possible definition of geodesic erosion is via negation. The resulting *multiscale geodesic dilation and erosion* of f given a reference image r are

$$\begin{aligned}\delta^t(f|r)(x) &\triangleq \sup\{h \leq r(x) : x \in \Delta^t(X_h(f)|X_h(r))\} \\ \varepsilon^t(f|r)(x) &\triangleq -\delta^t(-f| -r)\end{aligned}\quad (8.7)$$

The geodesic dilation and erosion possess a semigroup property:

$$\delta^t \delta^s = \delta^{t+s}, \quad \varepsilon^t \varepsilon^s = \varepsilon^{t+s}, \quad \forall s, t \geq 0 \quad (8.8)$$

whereas their conditional counterparts do not: $\delta_{tB}(\delta_{sB}(f|r)|r) \neq \delta_{(t+s)B}(f|r)$. By letting $t \rightarrow \infty$ the geodesic dilation (erosion) yields the *geodesic reconstruction*

tion opening ρ^- (closing ρ^+) of r from f :

$$\begin{aligned}\rho^-(f|r) &\triangleq \delta^\infty(f|r) = \bigvee_{t \geq 0} \delta^t(f|r) \\ \rho^+(f|r) &\triangleq \varepsilon^\infty(f|r) = \bigwedge_{t \geq 0} \varepsilon^t(f|r)\end{aligned}\quad (8.9)$$

The above limit δ^∞ (ε^∞) can also be reached using iterations δ^n (ε^n) for $n \rightarrow \infty$ since, due to the semigroup property, the geodesic dilation (erosion) at integer scales $t = n$ can be obtained via n -fold iteration of the unit-scale operator.

8.3.2 Image Operators on Reference Semilattices

In [269, 234, 235] a recent approach for a self-dual morphology was developed based on inf-semilattices. Now, the image space is the collection of signals $f : \mathbb{E} \rightarrow \mathbb{V}$, where $\mathbb{V} = \mathbb{R}$ or \mathbb{Z} . The value set \mathbb{V} becomes a *complete inf-semilattice (cisl)* if we select an arbitrary *reference* element $r \in \mathbb{V}$ and use the following partial ordering

$$a \preceq_r b \iff r \wedge b \leq r \wedge a \text{ and } r \vee b \geq r \vee a \quad (8.10)$$

which coincides with the *activity ordering* in Boolean lattices [365, 233].

Given a reference image $r(x)$, a valid signal cisl ordering is $f \preceq_r g$ defined as $f(x) \preceq_{r(x)} g(x) \forall x$. The corresponding signal cisl infimum becomes

$$\begin{aligned}\left(\bigwedge_i f_i\right)(x) &\triangleq [r(x) \wedge \bigvee_i f_i(x)] \vee \bigwedge_i f_i(x) \\ &= \text{med}[r(x), \bigvee_i f_i(x), \bigwedge_i f_i(x)]\end{aligned}\quad (8.11)$$

where $\text{med}(\cdot)$ denotes the median. Under the above cisl infimum, the image space becomes a cisl denoted henceforth by \mathcal{F}_r . Varying the reference signal r yields cisl's that are all isomorphic to each other. Significant in this chapter is the cisl \mathcal{F}_0 with $r(x) = 0$. An isomorphism between \mathcal{F}_0 and an arbitrary cisl \mathcal{F}_r is the bijection $\xi(f) = f + r$. Thus, if ψ_0 is an operator on \mathcal{F}_0 , then its corresponding operator on \mathcal{F}_r is given by $\psi_r(f) = \xi\psi_0\xi^{-1}(f)$. If ψ_0 is an erosion on \mathcal{F}_0 that is translation-invariant (TI) and self-dual, then ψ_r is also a self-dual TI erosion on \mathcal{F}_r . Note: the infimum, translation operator and negation operator on \mathcal{F}_0 are different from those on \mathcal{F}_r . For example, if $\nu_0(f) = -f$ is the negation on \mathcal{F}_0 , then self-duality of ψ_0 means $\psi_0\nu_0 = \nu_0\psi_0$, whereas self-duality on \mathcal{F}_r means $\psi_r\nu_r = \nu_r\psi_r$ where $\nu_r(f) = 2r - f$.

The simplest *multiscale TI self-dual erosion* on the cisl \mathcal{F}_0 is the operator

$$\psi_0^t(f)(x) = [0 \wedge \bigvee_{a \in tB} f(x-a)] \vee \bigwedge_{a \in tB} f(x-a) \quad (8.12)$$

The corresponding multiscale TI self-dual erosion on the cisl \mathcal{F}_r is

$$\psi_r^t(f) = r + \psi_0^t(f-r) \quad (8.13)$$

8.4 Multiscale Triphase Operators and Levelings

DEFINITION 4 . Given two operators α and β from \mathcal{L}^2 to \mathcal{L} that are increasing w.r.t. both arguments and, $\forall f, r$,

$$f \wedge r \leq \beta(f|r) \leq r \leq \alpha(f|r) \leq f \vee r, \quad (8.14)$$

a *triphasic operator* λ is defined by

$$\lambda(f|r; \alpha, \beta) \triangleq \alpha(f|\beta(f|r)), = \beta(f|\alpha(f|r)) \quad (8.15)$$

where the operators α and β are assumed to *commute* in the definition (8.15). (Sufficient conditions are given in [335].) The triphase operators have four arguments: two signals f and r and two operators α and β . The signal arguments (f, r) are written as $(f|r)$ to emphasize their asymmetric roles (marker vs. reference) and to connect them with conditional operators that use the same notation. If the operators α and β are known and fixed, we shall omit them and write $\lambda(f|r)$, or simply $\lambda(f)$ if r is assumed. The prototypical example of a triphase operator is when $\alpha(f|r) = \varepsilon_B(f) \vee r$ is a conditional erosion and $\beta(f|r) = \delta_B(f) \wedge r$ is a conditional dilation.

As shown in [335], the action of a triphase operator λ at points x where $f(x) > r(x)$ (resp., $f(x) < r(x)$) is determined only by α (resp., β):

$$\lambda(f|r)(x) = \begin{cases} \beta(f|r)(x), & \text{if } f(x) < r(x) \\ \alpha(f|r)(x), & \text{if } f(x) < r(x) \\ r(x), & \text{if } f(x) = r(x) \end{cases} \quad (8.16)$$

Some general properties of triphase operators follow next.

PROPOSITION 4 ([335]). (a) λ is antiextensive in the *cisl* \mathcal{F}_r : $\lambda(f|r) \preceq_r f$.

(b) λ is increasing in \mathcal{F}_r ; i.e., $f \preceq_r g \implies \lambda(f|r) \preceq_r \lambda(g|r)$.

(c) If α and β are dual of each other, then λ is self-dual; i.e., if $\alpha(-f| -r) = -\beta(f|r)$, then $\lambda(-f| -r) = -\lambda(f|r)$.

On digital spaces, Meyer [366] and Matheron [348] defined as a leveling of r any signal f such that $\delta_B(f) \wedge r \leq f \leq \varepsilon_B(f) \vee r$. This is equivalent to $f = \lambda(f|r)$, where λ is the conditional triphase formed by the conditional erosion $\varepsilon_B(\cdot| \cdot)$ and dilation $\delta_B(\cdot| \cdot)$ in place of α and β . By generalizing the triphase, we propose the following alternative definition of levelings, valid for discrete or continuous spaces.

DEFINITION 5 . A signal f is called a λ -induced *leveling* of r iff it is a fixed point of the triphase operator λ , i.e. $f = \lambda(f|r)$.

The following is a necessary and sufficient condition for f to be a λ -induced leveling of r :

$$f = \lambda(f|r) \iff \beta(f|r) \leq f \leq \alpha(f|r) \quad (8.17)$$

Since any triphase operator λ is antiextensive, a leveling of a reference r from a marker f can possibly be obtained by *iterating* λ to infinity, or equivalently by taking the *cisl infimum* of all iterations of λ . The limit of these iterations

$$\Lambda(f|r) \triangleq \lambda^\infty(f|r) = \bigwedge_{n \geq 1} \lambda^n(f) \preceq_r \cdots \preceq_r \lambda(f) \preceq_r f \quad (8.18)$$

exists in the *cisl* \mathcal{F}_r . Henceforth, we shall deal only with triphase operators λ for which $\lambda\lambda^\infty = \lambda^\infty$; e.g., this happens if α is a lattice erosion and β is a lattice dilation [335]. In such cases, $\Lambda(f|r)$ is a leveling, and the map $r \mapsto \Lambda(f|r)$ is called a *leveling operator*. Note that, Λ is an increasing, antiextensive and idempotent operator, and hence a *semilattice opening*, in the *cisl* \mathcal{F}_r . It is also an increasing and idempotent operator, and hence a *morphological filter*, in the complete lattice \mathcal{L} .

If we replace the operators α and β with the multiscale conditional flat erosion and dilation by B of (8.5) we obtain a *multiscale conditional triphase* operator

$$\lambda_{tB}(f|r) \triangleq \varepsilon_{tB}(f|\delta_{tB}(f|r)) = \delta_{tB}(f|\varepsilon_{tB}(f|r)) \quad (8.19)$$

By replacing the conditional dilation and erosion in (8.19) with their geodesic counterparts from (8.7) we obtain a *multiscale geodesic triphase* operator

$$\lambda^t(f|r) \triangleq \varepsilon^t(f|\delta^t(f|r)) = \delta^t(f|\varepsilon^t(f|r)) \quad (8.20)$$

For both the conditional and the geodesic triphase, its constituent erosion and dilation operators satisfy the basic properties of the operators α and β required for the definitions of triphase operators. Further, they commute.

Comparing (8.19) with (8.13) reveals that $\lambda_{tB}(\cdot|r)$ becomes a multiscale *translation-invariant (TI)* semilattice erosion on \mathcal{F}_r if r is constant. In particular, if $r = 0$, then λ_{tB} becomes a multiscale TI self-dual erosion on \mathcal{F}_0 . For non-constant r , λ_{tB} is generally neither TI nor an erosion. In contrast, the geodesic triphase is a semilattice erosion, although *not* TI.

PROPOSITION 5 ([335]). *The geodesic triphase operator $\lambda^t(f|r) = \varepsilon^t(f|\delta^t(f|r))$ is a semilattice erosion in the *cisl* \mathcal{F}_r ; i.e., $\lambda^t(\bigwedge_i f_i|r) = \bigwedge_i \lambda(f_i|r)$.*

The geodesic triphase is the most important triphase operator because it obeys a semigroup. This will allow us later to find its PDE generator.

PROPOSITION 6 ([335]). *(a) As $t \rightarrow \infty$, $\lambda^t(f|r)$ yields the geodesic leveling which is the composition of the geodesic reconstruction opening and closing:*

$$\Lambda(f|r) \triangleq \lambda^\infty(f|r) = \rho^-(f|\rho^+(f|r)) = \rho^+(f|\rho^-(f|r)) \quad (8.21)$$

(b) The multiscale family $\{\lambda^t(\cdot|r) : t \geq 0\}$ forms an additive semigroup:

$$\lambda^t(\lambda^s(\cdot|r)|r) = \lambda^{t+s}(\cdot|r), \quad \forall t, s \geq 0. \quad (8.22)$$

(c) For a zero reference ($r = 0$), the multiscale geodesic triphase operator becomes identical to its conditional counterpart and the multiscale TI semilattice erosion: $\psi_0^t(f) = \lambda^t(f|0) = \lambda_{tB}(f|0)$.

(d) For any r , the multiscale TI semilattice erosion $\psi_r^t(f) = r + \psi_0^t(f - r)$ obeys a semigroup: $\psi_r^t \psi_r^s = \psi_r^{t+s}$.

From the semigroup property (8.22), the n -fold iteration of the unit-scale geodesic triphase operator concides with its multiscale version at integer scale $t = n$. The same is true for the multiscale TI semilattice erosions. It is not generally true, however, for the conditional triphase operator $\lambda_B(f|r)$, which does not obey a semigroup. Further, its iterations converge to the *conditional leveling* $\Lambda_B(f|r) = \lambda_B^\infty(f|r)$ which is smaller w.r.t. \preceq_r than the geodesic leveling $\Lambda(f|r) = \lambda^\infty(f|r)$.

8.5 Partial Differential Equations

8.5.1 PDEs for 1D Levelings and Semilattice Erosions

Consider a 1D reference image $r(x)$ and a marker image $f(x)$ on \mathbb{R} , both real and continuous. We start evolving the marker image by producing the *multiscale geodesic triphase evolutions*

$$u(x, t) = \lambda^t(f|r)(x) = \delta^t(f|\varepsilon^t(f|r))(x) \quad (8.23)$$

The initial value is $u(x, 0) = f(x)$. In the limit we obtain the final result $u(x, \infty)$ which will be the leveling $\Lambda(f|r)$. Attempting to find a generator PDE for the function u , we shall analyze the following evolution rule: $\partial u(x, t)/\partial t = \lim_{s \downarrow 0} [u(x, t + s) - u(x, t)]/s$. By using the semigroup (8.22) that u satisfies and the pointwise representation (8.16) of triphase operators, the evolution rule becomes

$$\frac{\partial u}{\partial t} = \begin{cases} \lim_{s \downarrow 0} [\delta^s(u(x, t)|r)(x) - u(x, t)]/s, & \text{if } u(x, t) < r(x) \\ \lim_{s \downarrow 0} [\varepsilon^s(u(x, t)|r)(x) - u(x, t)]/s, & \text{if } u(x, t) > r(x) \\ 0, & \text{if } u(x, t) = r(x) \end{cases} \quad (8.24)$$

We shall show later that, at points where the partial derivatives exist, this rule becomes the following PDE: $u_t = -\text{sign}(u - r)|u_x|$. Starting from a continuous marker $f(x)$, the evolutions $u(x, t)$ remain continuous for all x, t . However, even if the initial image f is differentiable, at finite scales $t > 0$, the above triphase evolution may create shocks (i.e., discontinuities in the derivatives). One way to propagate these shocks (as done in solving evolution PDEs of the Hamilton-Jacobi type with level-set methods [401]) is to use conservative monotone difference schemes that pick the correct weak solution satisfying the entropy condition. An alternative way we propose to deal with shocks is to replace the standard derivatives with morphological sup/inf derivatives. For example, let

$$\mathcal{M}_x u(x, t) \triangleq \lim_{s \downarrow 0} [\bigvee_{|a| \leq s} u(x + a, t) - u(x, t)]/s$$

be the *sup-derivative* of $u(x, t)$ along the x -direction, if the limit exists. If the one-sided right derivative $\partial_{+x} u(x, t)$ and left derivative $\partial_{-x} u(x, t)$ of u along the

x -direction exist, then its sup-derivative also exists and is equal to $\mathcal{M}_x u(x, t) = \max[0, \partial_{+x} u(x, t), -\partial_{-x} u(x, t)]$. Obviously, if the left and right derivatives are equal, then the \mathcal{M} becomes equal to the magnitude $|u_x(x, t)|$ of the standard derivative. The nonlinear derivative \mathcal{M} leads next to a more general PDE that can handle discontinuities in $\partial u / \partial x$.

THEOREM 3 ([335]). *Let $u(x, t) = \lambda^t(f|r)(x)$ be the scale-space function of multiscale geodesic triphase operations with initial condition $u(x, 0) = f(x)$. Assume that r is continuous and f is continuous with left and right derivatives at all x . (a) If the partial left and right derivatives $\partial_{\pm x} u$ exist at some (x, t) , then*

$$\frac{\partial u}{\partial t}(x, t) = \begin{cases} \max[0, \partial_{+x} u(x, t), -\partial_{-x} u(x, t)], & \text{if } u(x, t) < r(x) \\ \min[0, \partial_{+x} u(x, t), -\partial_{-x} u(x, t)], & \text{if } u(x, t) > r(x) \\ 0, & \text{if } u(x, t) = r(x) \end{cases} \quad (8.25)$$

(b) If the two-sided partial derivative $\partial u / \partial x$ exists at some (x, t) , then u satisfies

$$\frac{\partial u}{\partial t}(x, t) = -\text{sign}[u(x, t) - r(x)] \left| \frac{\partial u}{\partial x}(x, t) \right| \quad (8.26)$$

Thus, assuming that $\partial u / \partial x$ exists and is continuous, the nonlinear PDE (8.26) can generate the multiscale evolution of the initial image $u(x, 0) = f(x)$ under the action of the geodesic triphase operator. However, even if f is differentiable, as the scale t increases, this evolution can create shocks. In such cases, the more general PDE (8.25) that uses morphological derivatives still holds and can propagate the shocks provided the equation evolves in such a way as to give solutions that are piecewise differentiable with left and right derivatives at each point.

Consider now on the cisl \mathcal{F}_0 the *multiscale TI semilattice erosions* $v(x, t) = \psi_0^t(f)(x)$ of a real 1D image $f(x)$ by 1D line segments $tB = [-t, t]$, defined in (8.12). Since $v(x, t)$ is the special case of the corresponding function $u(x, t)$ for multiscale geodesic triphase operations when $r = 0$, we can use the leveling PDE (8.26) to generate the evolutions $v(x, t)$:

$$\partial v / \partial t = -\text{sign}(v) |\partial v / \partial x|, \quad v(x, 0) = f(x) \quad (8.27)$$

If $r(x)$ is not zero, we can generate multiscale TI semilattice erosions $\psi_r^t(f) = r + \psi_0^t(f - r)$ of f by the following PDE system

$$\begin{aligned} \partial v / \partial t &= -\text{sign}(v) |v_x|, \quad v(x, 0) = f(x) - r(x) \\ \psi_r^t(f)(x) &= r(x) + v(x, t) \end{aligned} \quad (8.28)$$

If $f - r$ has both negative and positive values and is non-constant, then as $t \rightarrow \infty$ we obtain the reference r ; i.e., $\psi_r^\infty(f) = r$ because $\psi_0^\infty(f - r) = 0$.

To find a numerical algorithm for solving the previous PDEs, let U_i^n be the approximation of $u(x, t)$ on a grid $(i\Delta x, n\Delta t)$. Similarly, define $R_i \triangleq r(i\Delta x)$ and $F_i \triangleq f(i\Delta x)$. Consider the forward and backward difference operators:

$$D_{+x} U_i^n \triangleq (U_{i+1}^n - U_i^n) / \Delta x, \quad D_{-x} U_i^n \triangleq (U_i^n - U_{i-1}^n) / \Delta x \quad (8.29)$$

To produce a shock-capturing and entropy-satisfying numerical method for solving the leveling PDE (8.26) we approximate the more general PDE (8.25) by replacing time derivatives with forward differences and left/right spatial derivatives with backward/forward differences. This yields the following algorithm:

$$U_i^{n+1} = U_i^n - \Delta t [(P_i^n)^+ \max(0, D_{-x} U_i^n, -D_{+x} U_i^n) + (P_i^n)^- \max(0, -D_{-x} U_i^n, D_{+x} U_i^n)] \quad (8.30)$$

where $P_i^n = \text{sign}(U_i^n - R_i)$, $q^+ = \max(0, q)$, and $q^- = \min(0, q)$. Further, to avoid spurious numerical oscillations around zerocrossings of $f - r$, at each iteration we enforce the sign consistency

$$\text{sign}(U_i^n - R_i) = \text{sign}(F_i - R_i), \quad \forall n, i \quad (8.31)$$

We iterate the above scheme for $n = 1, 2, \dots$, starting from the initial data $U_i^0 = F_i$. For *stability*, $(\Delta t / \Delta x) \leq 0.5$ is required.

The above scheme can be expressed as *iteration of a discrete triphase operator* Φ acting on the *cisl* \mathcal{F}_R of 1D sampled real-valued signals with reference R :

$$U_i^{n+1} = \Phi(U_i^n), \quad \Phi(F_i) \triangleq \alpha(F_i) \vee [R_i \wedge \beta(F_i)], \quad (8.32)$$

where the operators α, β are given by, for $\theta = \Delta t / \Delta x$,

$$\begin{aligned} \alpha(F_i) &= \min[F_i, \theta F_{i-1} + (1 - \theta) F_i, \theta F_{i+1} + (1 - \theta) F_i], \\ \beta(F_i) &= \max[F_i, \theta F_{i-1} + (1 - \theta) F_i, \theta F_{i+1} + (1 - \theta) F_i]. \end{aligned} \quad (8.33)$$

By using ideas from methods of solving PDEs corresponding to hyperbolic conservation laws [401], we can easily show that this scheme¹ is conservative and monotone increasing for $\theta = \Delta t / \Delta x \leq 1$. Hence, it satisfies the entropy condition. Examples of running this algorithm are shown in Fig. 8.1. An important question is whether the above algorithm converges. The answer is given affirmative next.

PROPOSITION 7 ([335]). *If $\Phi(\cdot) = \alpha(\cdot) \vee [R \wedge \beta(\cdot)]$ and (α, β) are as in (8.33), then Φ is a parallel triphase operator and the sequence $U^{n+1} = \Phi(U^n)$, $U^0 = F$, converges to a unique limit $U^\infty = \Phi^\infty(F)$. For digital images F, R assuming a finite number of gray levels, the limit $\Phi^\infty(F)$ is a conditional leveling of R from F .*

If $\Delta t = \Delta x$, then the α and β operators (8.33) of the discrete triphase operator Φ in (8.32) become erosion and dilation, respectively, by a unit-scale window $B = \{-1, 0, 1\}$. Further, the corresponding PDE numerical algorithm coincides with the iterative discrete algorithm of [366] for constructing levelings.

¹There are also other possible approximation schemes, e.g. the scheme proposed in [400] to solve the edge-sharpening PDE $u_t = -\text{sign}(u_{xx})|u_x|$, which is however more diffusive and requires more computation than the above scheme; see [335].

8.5.2 PDEs for 2D Levelings and Semilattice Erosions

A straightforward extension of the leveling PDE from 1D to 2D images results by replacing the term $-|u_x|$ creating 1D multiscale erosions with the term $-||\nabla u||$ generating multiscale erosions by disks. Then the 2D leveling PDE becomes:

$$\begin{aligned}\partial u(x, y, t) / \partial t &= -\text{sign}[u(x, y, t) - r(x, y)] ||\nabla u(x, y, t)|| \\ u(x, y, 0) &= f(x, y)\end{aligned}\quad (8.34)$$

As in the 1D case, $u(x, y, t) = \lambda^t(f|r)(x, y)$ is a scale-space function holding the 2D multiscale geodesic triphase evolutions of the marker image $f(x, y)$ within the reference image $r(x, y)$. Of course, we could select any other PDE modeling the intermediate growth kernel by shapes other than the disk, but the disk has the advantage of creating an isotropic growth.

For discretization, let $U_{i,j}^n$ be the approximation of $u(x, y, t)$ on a computational grid $(i\Delta x, j\Delta y, n\Delta t)$ and set the initial condition $U_{ij}^0 = F_{ij} = f(i\Delta x, j\Delta y)$. Then, by replacing the magnitudes of standard derivatives with morphological derivatives and by expressing the latter with left and right derivatives which are approximated with backward and forward differences, we arrive at the following entropy-satisfying scheme for solving the 2D leveling PDE (8.34):

$$\begin{aligned}U_{i,j}^{n+1} &= \Phi(U_{i,j}^n), \quad \Phi(F_{ij}) \triangleq [R_{ij} \wedge \beta(F_{ij})] \vee \alpha(F_{ij}), \\ \alpha(F_{ij}) &= F_{ij} \\ &\quad - \Delta t \sqrt{\max^2[0, D_{-x}F_{ij}, -D_{+x}F_{ij}] + \max^2[0, D_{-y}F_{ij}, -D_{+y}F_{ij}]} \\ \beta(F_{ij}) &= F_{ij} \\ &\quad + \Delta t \sqrt{\max^2[0, -D_{-x}F_{ij}, D_{+x}F_{ij}] + \max^2[0, -D_{-y}F_{ij}, D_{+y}F_{ij}]}\end{aligned}\quad (8.35)$$

For stability, $(\Delta t / \Delta x + \Delta t / \Delta y) \leq 0.5$ is required. As in the 1D case, this scheme converges to a discrete conditional leveling. Examples of running the above 2D algorithm are shown in Fig. 8.3. In all image experiments based on PDEs we used $\Delta x = \Delta y = 1$, $\Delta t = 0.25$ as space-time steps.

As a by-product of the 2D leveling PDE, the multiscale TI semilattice erosions (8.13) of a marker image f by disks B w.r.t. a reference image r can be generated as follows:

$$\begin{aligned}\partial v / \partial t &= -\text{sign}(v) ||\nabla v||, \quad v(x, y, 0) = f(x, y) - r(x, y) \\ \psi_r^t(f)(x, y) &= r(x, y) + v(x, y, t)\end{aligned}\quad (8.36)$$

8.6 Discussion

We conclude by providing some insights on the behavior of levelings and multiscale semilattice erosions via several image experiments. Then, we also

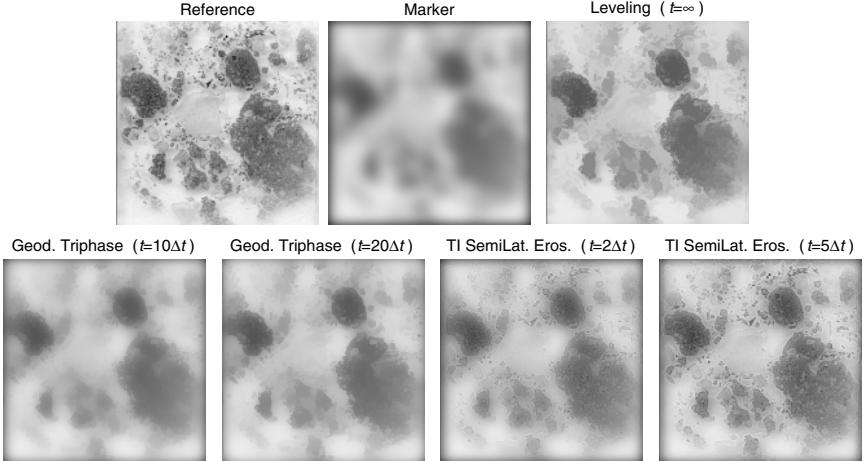


Figure 8.3. Multiscale triphase evolutions and leveling of a soil section image r generated by PDEs. The marker image f was obtained from a convolution of the reference r with a 2D Gaussian of $\sigma = 8$. Second row: left two images show geodesic triphase evolutions generated by the leveling PDE (8.34); right two images show multiscale TI semilattice erosions generated by the PDE (8.36).

comment on the advantages of PDE-based algorithms for generating these lattice scale-spaces.

As shown in Figs. 8.3 and 8.4, the leveling limit is strongly dominated by the structure of the reference image. Although the selection of markers suitable for producing levelings with various designable properties is still an open issue, it appears that a smooth version of the reference works well as a marker for applications of image simplification and segmentation. This choice also works for image denoising where the marker may be a linear or nonlinear smoothing of the noisy reference. In a different scenario shown in Fig. 8.4, we experimented with a binary edge map as reference whereas the marker was a smooth version of the same original image. Here the intermediate triphase evolutions (geodesic semilattice erosions) toward the leveling seemed useful for adding image region details back to the edge map. Finally, as discussed in [235, 335], the intermediate multiscale TI semilattice erosions seem potentially applicable to mixing or morphing the marker image into the reference, even if the two images are completely unrelated. On comparing the speed of convergence, we have experimentally found that the geodesic triphase evolutions toward levelings converge to the limit more slowly than the multiscale TI semilattice erosions.

The basic *algebraic discrete* algorithm that Meyer [366] developed to construct levelings of digital images is the iteration of the conditional triphase operator $\lambda(F_i) = \varepsilon_B(F_i) \vee [R_i \wedge \delta_B(F_i)]$, where δ_B and ε_B are flat dilation and erosion by a discrete unit-scale disk-like set B . Now we know that this converges to a discrete conditional leveling $\Lambda_{algdiscr}$. The new PDE-based numerical algorithm (8.32),(8.33) converges to another discrete conditional leveling Λ_{pdenum} . If Λ_{true}

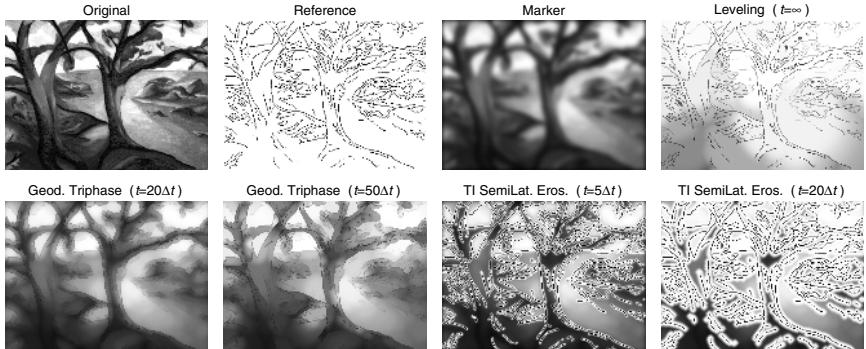


Figure 8.4. First row: original image g , reference image r resulting from applying the Canny edge detector to g , marker image f which is a Gaussian convolution of g , and the leveling $\Lambda(f|r)$. Second row: first two images show multiscale geodesic triphase evolutions (converging to the leveling); last two images show multiscale TI semilattice erosions. (All evolutions were generated by PDEs.)

is the sampled true (geodesic) leveling, then $r \preceq_r \Lambda_{algdiscr} \preceq_r \Lambda_{pdenum} \preceq_r \Lambda_{true}$. Hence, the algebraic discrete algorithm yields a result that has a larger absolute deviation from the true solution than the PDE numerical algorithm. Further, the algebraic discrete algorithm is a special case of the PDE algorithm using the value $\theta = \Delta t / \Delta x = 1$, which makes it *unstable* (amplifies small errors). Thus, in addition to its well-known advantages (such as better geometry and accuracy, physics, and insightful modeling), the PDE approach also has some advantages over the discrete modeling that are specific for the operators examined in this chapter. In the 2D case we have an additional comparison issue: Although for the triphase evolutions toward levelings the desired result in segmentation applications is mainly the final limit, there may be other applications, for instance such as mixing/morphing images, where we need to stop the marker growth before convergence. In such cases as evolutions of 2D multiscale (geodesic or TI) semilattice erosions, the isotropy of the partially grown marker offered by the PDE approach is an advantage over the discrete algebraic approach.

Part IV

Grouping

Fast Marching Techniques for Visual Grouping & Segmentation

Eftychis Sifakis and George Tziritas

Abstract

A new region growing method is proposed for segmenting images. The region boundaries are formulated as level sets and the pixel labeling process is implemented using a new multi-label fast marching algorithm. The region contours are propagated with a velocity proportional to the *a posteriori* probability of the respective label. Statistical tests are performed to generate the initially labeled sets. Any image feature, given it is semantically relevant, can be considered for the segmentation process. Illustrations are given for combined luminance, chrominance and texture classification and segmentation in natural scenes. Moving object extraction based on change detection is also considered, which is performed as a two-label classification.

9.1 Introduction

Image segmentation is a vital component in any system involving image content analysis and computer vision. In the MPEG-4 standard, segmentation plays an important role in coding performance and object manipulation [488]. In the MPEG-7 standard a spatio-temporal locator is needed for the description of visual content [454]. A continuous effort has been made by the research community to solve the segmentation problem. The numerous existing approaches may be classified into two main categories: boundary-based and region-based.

Edge detection is the earliest of the boundary-based methods, based on local gradients [77]. Active contours [50], based on local gradients as well, have been introduced for tracking deformable moving objects [263], by minimizing a functional whose local maximum lies on the object boundary. Nevertheless, active contours are relatively noise sensitive. Moreover, their result depends on the initialization and they are not sufficiently topologically adaptive. Some progress has been made with the balloon model [127], where the external force applied to the

curve is modified in order to make the active contour less sensitive to weak edges and spurious isolated edge points.

In the region-based approaches, techniques such as seeded region growing [6] or split-and-merge [420] were introduced. The labeling problem can also be globally formulated using Markov random field modeling. The final solution is obtained by minimizing an energy function, where stochastic relaxation [207] may be used, but deterministic relaxation [45, 119] is often preferred, being less computationally expensive.

Efforts have also been made towards the unification of the contour- and the region-based approaches. Zhu and Yuille [613] proposed a region competition method which combines the geometrical features of snakes/balloon models and the statistical techniques of region growing. Paragios and Deriche [413] introduced the concept of geodesic active regions. The active contour evolves under the influence of two forces: a boundary force, which also contains curvature constraints and a region force, which aims to move the curve in the direction that maximizes the *a posteriori* segmentation probability.

Level set theories have been used in the formulation of several region- or boundary-based approaches for image segmentation. The mapping of active contours to the level set formulation [331, 401] has lifted many of the inconveniences of active contours, while the fast marching algorithm [473, 475] provides a computationally efficient method for tracking an evolving contour. The introduction of the geodesic active contours [85] has allowed the unification of the classical active contour based on energy minimization and the geometric active contours based on the theory of curve evolution. In this last approach, the algorithm initialization and termination problems are solved and more stable boundaries are obtained, by computing a level set solution using a geometric flow. In [100, 594] level set formulations are used for the maximization of a segment uniformity criterion, defined over a given classification, in conjunction with smoothness constraints over the boundaries of the resulting segments. Furthermore, in [456] the segmentation of an arbitrary number of classes is addressed leading to the combined evolution of several level set modeled contours. The last approach is based on the minimization of a functional that enforces region uniformity, contour smoothing, and classification coherence.

In [484] a new region-based methodology for image segmentation has been introduced, where statistical approaches for modeling the different region features are applied and labeling is achieved through a novel algorithm based on level sets and the monotonical evolution of region boundaries. As multiple simultaneously propagating contours are considered, we propose an extension of the level set approach to a multi-label framework, while allowing the propagation speed to depend on the respective region label. The segmentation performance strongly depends on the description of the label content and on the capability of incorporating the label description into the propagation velocity. For that purpose, we propose to define the propagation speed as the *a posteriori* probability of the respective label. A statistical approach, where the number of labels is assumed to

be known, is therefore adopted, which requires adequate models. Pattern analysis techniques are used for the identification of the corresponding models.

The rest of this chapter is organized as follows. In Section 2, we review the fast marching algorithm and we describe how we extend it to the multi-label case for classification purposes. In Section 3, we consider the case of combined colour and texture segmentation, where the intensity and chromaticity features are mainly captured by the histogram distribution, while Gaussian assumptions are possible too. The Discrete Wavelet Analysis is performed for the description of the texture content. The very important problem of automatic feature extraction for the description of label content is also considered, as well as the initialization of the level sets. In Section 4 we consider the moving object localization problem based on intensity change detection. Finally, conclusions are drawn in Section 5.

9.2 The Multi-Label Fast Marching algorithm

9.2.1 *The stationary level set equation*

Level set theory provides a framework for tracking the evolution of any curve in the plane given the velocity of the curve along its normal direction. In the pioneering work by Osher and Sethian [401] the various instances of the evolving contour are embedded as level sets of a function of higher dimensionality. In this framework the velocity function is free to include terms dependent on the geometrical characteristics of the evolving contour, such as the curvature or the outward normal to the moving contour.

Given the limitation of a constantly positive (or constantly negative) velocity function, leading to a monotonical motion of the propagating front, an *arrival time function* $T(s)$ corresponding to the time point when the moving contour crosses over the point s is well defined. Under this formulation the arrival time function T satisfies the stationary level set equation

$$F|\nabla T| = 1, \quad (9.1)$$

which simply states that the gradient of the arrival time function is inversely proportional to the velocity F of the contour at any given point. The preceding formulation allows for the constructive calculation of the arrival time function T without resorting to iterative methods. The tradeoff for the computational efficiency is an inherent difficulty in integrating local properties of the evolving contour, such as curvature, in the velocity function F . Under those limitations the well-known Fast Marching level set algorithm [473], constructs a solution to Equation (9.1) from initial data with an execution cost of $n \log n$.

9.2.2 *Multiple interface extensions and Fast Marching algorithm*

The original formulation of the level set technique, as given in [401], applies specifically where there exists a clear distinction between an ‘outside’ and an ‘in-

side' region, separated by the evolving contour. Nevertheless, several applications, including multiple object segmentation and clustering require the consideration of more than two regions. In the simplest of cases where the distinct regions exhibit a smooth behavior and no triple points appear as the result of interface evolution, boundaries between different regions could be formulated as different level sets of the same function. Moreover, a technique for the proper handling of triple points and other singularities induced by multi-interface propagation can be found in [474]. These methods apply to the time-dependent level set formulation.

The work presented herein is motivated by the large number of applications that could be addressed by the tracking of the monotonical evolution of distinct regions into a special *blank* or *unlabeled* region and observing the final result of the initial regions' convergence over each other. The input to this approach would consist of an initialization for the expanding regions and a rule for their expansion into the blank region, in terms of their propagation velocity. Since the proposed framework includes strictly monotonical (expanding) motion of the considered regions, the stationary level set formulation would be best suited and the utilization of the Fast Marching algorithm would yield a favorable algorithmic complexity.

In the original two-region context, most shape modeling, feature extraction or segmentation applications of the Fast Marching level set algorithm involve initializing the arrival time map with seed regions, calculating arrival times for the rest of the spatial domain considered and either explicitly selecting a proper level set or utilizing an adequate criterion for picking the most appropriate propagation instance as the segmentation result. In the proposed framework the initialization consists of high confidence representatives of the regions in question, namely the *outside* and *inside* of the object(s) to be extracted. A third region corresponding to yet *undecided* sites of the segmentation domain is considered and velocities for the propagation of either region into the undecided one are supplied. The boundaries of both propagating regions are prescribed to freeze on contact, yielding the final segmentation solution, while eliminating the need for explicit selection of a propagation instance.

A trivial way of achieving the described functionality is to use the initialization of every region as the zero level set of an independent propagation, using the Fast Marching algorithm. Upon completion of all distinct propagations the first region managing to arrive at each site would be selected to specify its label and arrival time. This approach allows for the independent definition of propagation velocity for each expanding region, a property greatly exploited in the range of applications presented herein. Nevertheless, the execution cost for this algorithm scales with the number of independent regions and it can be shown that it is also subject to morphological instability, *e.g.* two regions that are separable with a single curve upon initialization are not bound to converge onto a single interface.

9.2.3 Fast Marching algorithm and labeling

The new Multi-Label Fast Marching algorithm presented in this chapter is an extension of the well-known Fast Marching algorithm introduced by Sethian

[473], capable of manipulating multiple propagating contours, corresponding to the boundaries of competitively expanding regions. The low computational cost of the classical Fast Marching algorithm is maintained, since it is effectively made independent of the number of distinct regions present in the initialization. The new algorithm targets applications requiring static segmentation as well as labeling and clustering problems.

The Multi-Label Fast Marching algorithm computes a constructive solution to the stationary level set Equation (9.1) given initial conditions in terms of the zero level set of the arrival time function $T(s)$. Initializations may be provided for multiple non-intersecting regions for which the propagation velocity is allowed to follow an independent definition. All distinct regions (or *labels*) are propagated simultaneously according to their respective velocity definitions with the constraint that one region may not infiltrate a region that has been swept by another. The propagating regions evolve in a competitive fashion, with the algorithm reaching a deterministic halt once all sites of the considered domain have been labeled.

For the purposes of this text we shall limit the description of the new algorithm to the case of a two-dimensional image, although the algorithm can be trivially extended to three or more dimensions. All image pixels are either idle or carry a number of *candidacies* for different labels. Candidacies can be either *trial*, *alive* or *finalized*. *Trial* candidacies for a certain label are introduced to a specific pixel lacking a finalized candidacy when a neighboring pixel acquires a finalized candidacy for the same label. *Trial* candidacies carry an arrival time estimate which is subject to adjustment according to the process of its neighboring candidacies for the same label. *Alive* candidacies are selected from the set of trial candidacies according to a minimum arrival time criterion and have their arrival time estimate fixated. The first trial candidacy to be turned alive per pixel is considered a finalized candidacy and is used in specifying the pixel label and arrival time in the final propagation result.

A symbolic description of the Multi-Label Fast Marching algorithm is as follows

```

InitTValueMap()
InitTrialLists()
while (ExistTrialPixels()) {
    pxi = FindLeastTValue()
    MarkPixelAlive(pxi)
    UpdateLabelMap(pxi)
    AddNeighboursToTrialLists(pxi)
    UpdateNeighbourTValues(pxi)
}

```

The algorithm is supplied with a label map partially filled with classification decisions. The arrival time for the initially labeled pixels is set to zero, while for all others it is set to a special value, *e.g.* infinity. A map of pointers to linked lists of candidacies is also maintained. Candidacy lists are initially empty, with the exception of unlabeled pixels that are neighbors to initial decisions, for which a trial candidacy is introduced carrying the label of the neighboring decision and an

arrival time estimate is allocated. All trial candidacies are contained in a common priority queue.

If the candidacy queue is not empty, the trial candidacy with the smallest arrival time is selected and marked alive. If no other alive candidacies exist for this pixel, the candidacy is considered finalized and copied to the final label map. For all neighbors of this pixel lacking an alive candidacy, a trial candidacy for the same label is introduced. Finally, all neighboring trial candidates update their arrival times according to the revised condition. The re-estimation of the arrival times is performed with the utilization of the stationary level set Equation (9.1). Under a common gradient approximation robust in the presence of shocks, the equation is written

$$\begin{aligned} 1/F_{ij}^2 = \max(\max(D_{ij}^{-x}T, 0), -\min(D_{ij}^{+x}T, 0))^2 + \\ \max(\max(D_{ij}^{-y}T, 0), -\min(D_{ij}^{+y}T, 0))^2. \end{aligned} \quad (9.2)$$

Equation (9.2) is solved for the value of the function T at the specified pixel. If the quadratic equation yields more than one solution, the greatest is used.

Although it seems possible that candidacies for all available labels may occur in a single site, it should be noted that a trial candidacy is only introduced by a neighboring candidacy being finalized, limiting the number of possible candidacies per pixel to a maximum of four. In practice, trial pixels of different labels coexist only in region boundaries, giving an average of at most two label candidacies per pixel. Even in the worst case, though, it is evident that the time and space complexity of the algorithm is independent of the number of different labels. Experiments have illustrated a running time no more than twice the time required by the single contour fast marching algorithm.

9.2.4 Label propagation

The multi-label fast marching level set algorithm, presented in the previous subsection, is applied to all sets of points initially labeled. The contour of each region propagates according to a velocity field which depends on the label and on the distance of the considered point from the candidate class. The label-dependent propagation speed is defined according to the *a posteriori* probability. The candidate label is ideally propagated with a speed in the interval $(0, 1]$, which is equal to the *a posteriori* probability of the candidate label at the considered point. Let us define at a site s , for a candidate label $l(s)$ and for a data vector $x(s)$ the propagation speed as

$$F_l(s) = \Pr\{l(s)|x(s)\}.$$

Then we can write

$$F_l(s) = \frac{p(x(s)|l(s))\Pr\{l(s)\}}{\sum_k p(x(s)|k(s))\Pr\{k(s)\}} = \frac{1}{1 + \sum_{k \neq l} \frac{p(x(s)|k(s))}{p(x(s)|l(s))} \frac{\Pr\{k(s)\}}{\Pr\{l(s)\}}} \quad (9.3)$$

Therefore the propagation speed depends on the likelihood ratios and on the *a priori* probabilities. The likelihood ratios can be evaluated according to the assumptions on the data, and the *a priori* probabilities could be estimated or assumed all equal.

Under several commonly adopted models the probability density function is an exponential function of a distance measure of the data

$$p(x(s)|l(s)) = e^{-d_l(x(s))}$$

If we assume that the *a priori* probabilities are all equal, we obtain

$$F_l(s) = \frac{1}{1 + \sum_{k \neq l} e^{d_l(x(s)) - d_k(x(s))}}. \quad (9.4)$$

This expression of the propagation speed illustrates that when the propagated label is the correct one, all the exponents in the sum are negative and the speed is therefore close to unity. On the other hand, when the propagated label is incorrect, at least one exponent is positive, and therefore the speed is biased towards zero.

In order to compare the two speeds, let us now consider the case of two equiprobable labels. The mean time for advancing one unit length, if the curve evolves with a force corresponding to the region properties (without loss of generality, assume with label 0), is

$$E\{|\nabla T(s)|; 0\} = 1 + \int \frac{p(x|1)}{p(x|0)} p(x|0) dx = 2.$$

If the curve evolves in the opposite labeled region, we have

$$E\{|\nabla T(s)|; 1\} = 1 + \int \frac{p(x|1)}{p(x|0)} p(x|1) dx > 2 + \int p(x|1) \ln \frac{p(x|1)}{p(x|0)} dx.$$

The right-hand term is the Kullback distance D between the two distributions, and it is always positive. Therefore the ratio of the two mean times is

$$\frac{E\{|\nabla T(s)|; 1\}}{E\{|\nabla T(s)|; 0\}} = 1 + \frac{D(p(x|1), p(x|0))}{2} > 1. \quad (9.5)$$

The more discriminating the two probability distributions are, the more important the ratio of the two propagation speeds is, making more confident that the evolving curves are trapped by the boundary.

To illustrate the above using an example, let us suppose that the data is scalar distributed according to the Gauss law, with identical variance and two different mean values (μ_0 and μ_1). It is straightforward to show that

$$D(p(x|1), p(x|0)) = \exp \left(\frac{(\mu_1 - \mu_0)^2}{\sigma^2} \right). \quad (9.6)$$

Clearly, the evolution of the curve in a region which is differently labeled is decelerated, and the amount of deceleration depends in general on the signal-to-noise ratio. In practice a SNR equal to 10 is sufficient for stopping the evolution.

In this work, several features are assumed to follow the generalized zero-mean Gaussian distribution

$$p(x) = \frac{c}{2\sigma \Gamma(\frac{1}{c})} e^{-(\frac{|x|}{\sigma})^c}, \quad (9.7)$$

where the parameter σ is the standard deviation and c reflects the sharpness of the probability density function. For $c = 2$, we obtain the Gaussian distribution and for $c = 1$, the Laplacian distribution. Then we obtain

$$D(p(x|1), p(x|0)) = \frac{1}{c} \left(\ln \frac{\sigma_0^c}{\sigma_1^c} + \frac{\sigma_1^c}{\sigma_0^c} - 1 \right) \quad (9.8)$$

We use the fast marching algorithm for advancing the contours towards the unlabeled space. The dependence of the propagation speed only on the pixel properties, and not on contour curvature measures, is not a disadvantage here, because the propagation speed takes into account the region properties.

9.3 Colour and texture segmentation

Luminance, colour and/or texture features may be used, either alone or in combination, for segmentation. In our approach luminance and colour classes are described using the corresponding empirical probability distributions. For texture analysis and characterisation a multichannel scale/orientation decomposition is performed using Wavelet Frame Analysis [313, 314].

9.3.1 Texture and colour description

The features for texture segmentation are derived from the discrete wavelet frames analysis [541] using a pair of translation-invariant linear filters. The image is decomposed into components corresponding to different scales and orientations. The application of the separable 2-D filter bank on a given image yields three high-frequency detail components for each analysis level plus a low-frequency approximation component at the last level. All components are shown [314] to be uncorrelated in the case of ideal filters.

The feature vectors for the texture content considered are the variances of the $N - 1 = 3L$ high frequency components, for L levels of analysis, calculated over the distinct texture classes present in the image. The low frequency approximation is not used. If the luminance is sufficiently discriminating, it is also used as an additional feature described by its empirical probability distribution.

Lab colour space, designed to be perceptually uniform, was used here for colour feature extraction. Because the luminance, or intensity, component is used separately, only the chromaticity components (a, b) are used. In our work the local 2-D histograms of the (a, b) components were used as features. When some model of the distribution of the (a, b) histograms is fit (e.g., Gaussian or Laplacian), the

parameters of the model are used as the features. Often no such modeling is feasible, in which case local histogram estimation is required, making the procedure time consuming. The histograms are smoothed with a Gauss kernel to improve statistical robustness.

9.3.2 Automatic feature extraction

An essential step of the whole framework consists of estimating the features associated to the different labels. The only assumption we make is that the number of labels is known.

If the adoption of a model is reliable and, in particular, if this model is tractable, the mixture analysis is preferred. Nevertheless, the use of an *a priori* model might be difficult, if not arbitrary. After the multi-channel wavelet analysis, the generalized Gaussian model is plausible, but it is difficult to obtain accurate parameters using a mixture analysis, because the distributions are all zero-mean. Additionally, in some cases, luminance or chromaticity only may be sufficient for the segmentation, but no general model is applicable. Therefore, when the model estimation is practically impossible using mixture analysis or when the adoption of a model is not plausible, a clustering technique could lead to the discrimination of the labels and to the estimation of their description. For that purpose, we use a hierarchical clustering method [168]. Any other clustering algorithm may be used as well. The clustering is applied on blocks resulting from a systematic division of the image. The blocks are hierarchically clustered using the Bhattacharya distance as a dissimilarity measure.

For continuous variables this measure is defined as

$$d^B = -\ln \left(\int_x \sqrt{p_1(x)p_2(x)} dx \right), \quad (9.9)$$

where p_1 and p_2 are probability density functions of a feature vector x of any dimension. The discrete version of the Bhattacharya distance is

$$d^B = -\ln \left(\sum_i \sqrt{p_1(i)p_2(i)} \right), \quad (9.10)$$

where $p_j(i)$ is the probability of the i^{th} value of class j . The Bhattacharya distance is strongly linked to the minimum classification error for the two-classes case. In addition, it is a special case of the Chernoff bound of classification error probability [600].

If a model of the distribution is known, a simpler expression of the Bhattacharya distance can be deduced. The extracted features may often be assumed uncorrelated. The simplified expression assuming generalized Gaussian distribution (Equation (9.7)) and uncorrelated features is

$$d^B = \frac{1}{c} \sum_{i=1}^{N-1} \ln \frac{\sigma_{i,1}^c + \sigma_{i,2}^c}{2\sqrt{\sigma_{i,1}^c \sigma_{i,2}^c}}, \quad (9.11)$$

where $\sigma_{i,n}$ corresponds to the standard deviation of the i^{th} feature of class n . When the clustering is complete, the description of the labels is determined. If a parametric model is used, then the parameters are estimated. In the absence of a parametric model, Equation (9.10) is used for computing distances. In both cases the estimation is performed on the clustered blocks.

In order to estimate the feature vectors of the various classes present in the image, a hierarchical clustering algorithm [168] is applied to the blocks. All blocks in the image are used as the initial clusters. Each step of the algorithm merges the pair of clusters having the most similar feature vectors and the features of the new cluster are updated accordingly. The procedure terminates when the number of clusters is reduced to the number of different classes the image to be segmented is known to contain.

9.3.3 Initial level sets

The next step consists of determining the initial seed regions. An initial map of labeled sites is obtained using statistical tests which classify points with high confidence. The probability of classification error is set to a small value. At first, all pixels are classified according to their distance from the different labels. The distribution of the data in a window centered at each site is approximated. Then, the Bhattacharya distances from this distribution to the features of each label are computed and assigned to the site. The distances at each site are subsequently averaged in a series of windows B_w of dimension $(2w + 1) \times (2w + 1)$, ($w = 1, \dots, P$). The mean distance in each window is used for classifying the central site to one of the possible labels. The candidate label $k(s)$ of site s is selected by finding the label which minimizes the sum of its distances from the neighbor sites p in window B_w ,

$$k(s) = \arg \min_l \sum_{p \in B_w} d_l^B(s + p). \quad (9.12)$$

The confidence criterion for classification of site s into the candidate label $k(s)$ results from comparing the distance of the considered site from the candidate label against the distance from the nearest label among all the others, as described in the following expression,

$$V_{k(s)} = \sum_{p \in B_w} \left(\min_{l \neq k(s)} d_l^B(k(s) + k(p)) - d_{k(s)}^B(s + p) \right). \quad (9.13)$$

Sites are then sorted according to their confidence measure and a specific percentage of the sites with highest confidence are retained and subsequently labeled. A small percentage is generally sufficient. Sites which are retained for each of these P window sizes are considered as forming the initial sets of labeled points. Parameter P ranges from 3 to 6 in most applications.

9.3.4 Label propagation

The labels of the initial level sets are then propagated according to the principle presented in Section 9.2.4. Assuming that the probability density function of the texture images is Gaussian, and given that the high frequency components are zero-mean, the distance of a site s represented by the vector $x(s)$ from a texture class j with variances $\sigma_{i,j}^2$ is

$$d_j(x(s)) = \frac{1}{2} \sum_{i=1}^{N-1} \left(\frac{x_i^2(s)}{\sigma_{i,j}^2} + \log \sigma_{i,j}^2 \right), \quad (9.14)$$

where $N - 1$ is the number of high frequency components.

If the probability density functions of the intensity and colour features are also Gaussian, the three corresponding components are added in Equation (9.14). Otherwise, the empirical probability distribution is used for the colour and intensity features,

$$d_j(y_C(s)) = -\ln p_j(y_C(s)), \quad (9.15)$$

$$d_j(y(s)) = -\ln p_j(y(s)), \quad (9.16)$$

where $y_C(s)$ is the vector of (a, b) colour components and $y(s)$ is the intensity.

The multi-label fast marching level set algorithm is then applied to all sets of points initially labeled. The contour of each region propagates according to a velocity field which depends on the label and on the distance of the considered point from the candidate label. The exact propagation velocity for a given label is

$$F_l(s) = \frac{\Pr(l)}{\sum_{k \neq l} \Pr(k) e^{d_l(x(s)) - d_k(x(s))}}, \quad (9.17)$$

where the *a priori* probabilities $\Pr(k)$ are estimated from the classification of the sites against the prototype feature classes according to the Bhattacharya distance. The distance of each site from the prototype feature classes is computed using Equations (9.14)–(9.16) with the variances for a considered site being calculated in a window centered at it.

Figure 9.1 presents results on the *SeaStones* image in which three classes are distinguished. Two of the classes contain similar colour distributions in the *Lab* space, while two classes contain similar textures. As a result the classification is difficult with a significant error probability. In Fig. 9.1(c) the classification result according to the Bhattacharya distance is shown. In Fig. 9.1(e) the pointwise classification result according to the *maximum likelihood* criterion is shown. In Fig. 9.1(b) the initial labeled map with 20% of the image points labeled according to the technique of Section 9.3.3 is shown. In Fig. 9.1(d) an intermediate instance of the propagation process where 60% of the image points are labeled is shown. The final segmentation result is considered very satisfactory and is shown in Fig. 9.1(f).

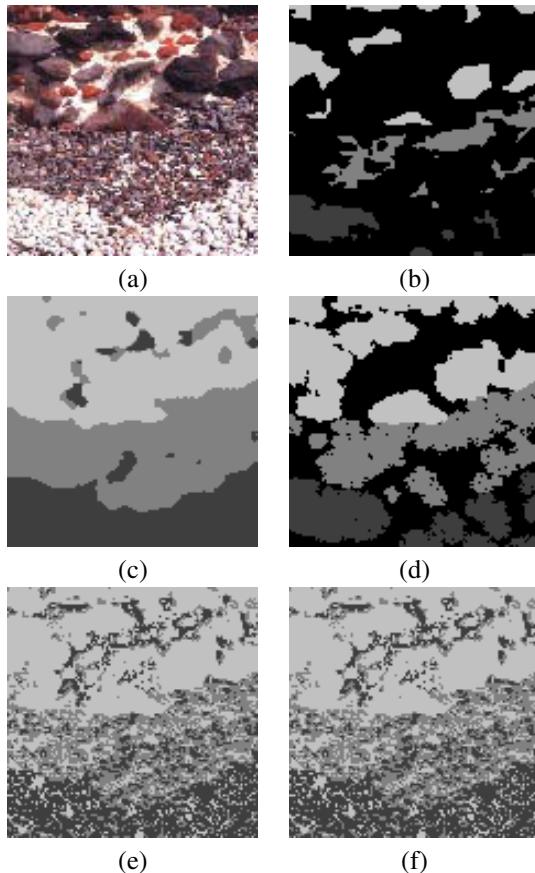


Figure 9.1. The segmentation result on the *SeaStones* image (a).

9.4 Change detection

Change detection in a video sequence is an important issue in object tracking, video-conferencing and traffic-monitoring among others. The change detection problem consists of labeling each pixel s of one frame t of a video sequence as static ($\Theta(s) = \text{static}$) or moving ($\Theta(s) = \text{mobile}$).

9.4.1 Inter-frame difference modeling

In our approach [485, 486, 487], the simple inter-frame grey level difference $x(s)$ is considered:

$$x(s) = I(s, t+1) - I(s, t) \quad (9.18)$$

Therefore, a pixel is an *unchanged pixel* if the observed difference $x(s)$ supports the hypothesis for static pixel, and a *changed pixel*, if the observed difference

supports the alternative hypothesis, for mobile pixel. Let $p_0(x|static)$ (resp. $p_1(x|mobile)$) be the probability density function of the observed inter-frame difference under the respective hypothesis. These probability density functions are assumed to be homogeneous, *i.e.*, independent of the pixel location, and usually they are under Laplacian or Gaussian law. A zero-mean Laplacian distribution function is used here to describe the statistical behavior of the pixels under both hypotheses. Thus the conditional probability density function of the observed temporal difference values is given by

$$p(x(s)|\Theta(s) = l) = \frac{\lambda_l}{2} e^{-\lambda_l|x(s)|}. \quad (9.19)$$

If P_{static} (resp. P_{mobile}) be the two *a priori* probabilities, the probability density function of the difference is given by

$$p_X(x) = P_{static} p_0(x|static) + P_{mobile} p_1(x|mobile). \quad (9.20)$$

In this mixture distribution $\{P_l, \lambda_l : l \in \{static, mobile\}\}$ are the unknown parameters. Mixture analysis aims at determining the *a priori* probabilities of the labels, P_l , and the parameters, λ_l , of the probability density functions of the data given the labels. The most frequently used method for parameter estimation uses the Maximum Likelihood principle, which results in an iterative algorithm [168, 356].

9.4.2 Level set-based labeling

An initial map of labeled sites is obtained using statistical tests. The initialization process is different from that described in Section 9.3.3, because *changed* sites should be detected in a point-wise way in order to avoid false alarms which could be propagated, while block-wise decisions are preferable for the *unchanged* initial labels. The first test detects changed sites with high confidence. The false alarm probability is set to a small value, say P_{FA} . For the Laplace distribution used here, the corresponding threshold is

$$T_1 = \frac{1}{\lambda_0} \ln \frac{1}{P_{FA}}. \quad (9.21)$$

Subsequently a series of tests is used for finding unchanged sites with high confidence, that is with small probability of non-detection. For these tests, a series of six windows of dimension $(2w+1)^2$, ($w = 2, \dots, 7$), are considered and the corresponding thresholds are pre-set as a function of λ_1 . Finally the union of the so defined sets determines the initial set of “unchanged” pixels.

The multi-label fast marching level set algorithm is then applied for all sets of points initially labeled. The contour of each region propagates according to a motion field, which depends on the label and on the absolute inter-frame difference.

In the case of a decision between the “changed” and the “unchanged” labels, according to the assumption of Laplacian distributions, the likelihood ratios are

exponential functions of the absolute value of the inter-frame difference. At a pixel level, the decision process is highly noisy, and could be made more robust by taking into account the known labels in the neighborhood of the considered pixel. This means that the *a priori* probabilities in Equation (9.3) are locally adapted. Finally, the exact propagation velocity for the “unchanged” label is

$$F_0(s) = \frac{1}{1 + e^{\beta_0(|x(s)| - n\zeta + \theta_0)}} \quad (9.22)$$

and for the “changed” label

$$F_1(s) = \frac{1}{1 + e^{\beta_1(\theta_1 - |x(s)| - (n+\alpha)\zeta)}}, \quad (9.23)$$

where n is the number of the neighboring pixels already labeled with the same candidate label, and α takes a positive value if the pixel at the same site of the previous label map is an interior point of a “changed” region, else it takes a zero value. The parameters $\beta_0, \beta_1, \theta_0, \theta_1$ and ζ are adapted according to the initial label map and the features characterizing the data (P_l, λ_l) . In the current implementation the parameters are set as follows:

$$\zeta = 0.1T_1, \quad \theta_0 = 4\zeta, \quad \theta_1 = 3.5 \left(\zeta + \frac{1}{\lambda_0} \right), \text{ and } \beta_1 = 1.$$

The value of parameter β_0 must take into account the initial label map. If the percentage of the pixels labeled “unchanged” is less than the estimated probability of this label, parameter β_0 is given a value less than 1, because this means that the thresholds used were probably relatively low. Otherwise the value given is greater than 1. In Figure 9.2 the two speeds are mapped as functions of the absolute inter-frame difference for typical parameter values near the boundary.

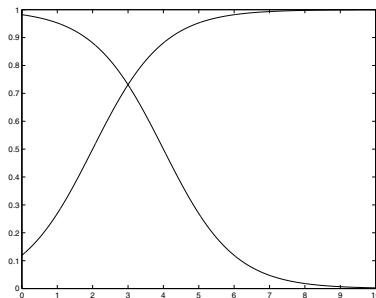


Figure 9.2. The propagation speeds of the two labels.

Figure 9.3 shows two different initial labeled maps and the corresponding final labeled images based on the interframe difference between the two first frames of the *Erik* sequence. The background is shown in black, the foreground in white and unlabeled points in gray. We observe that quite different initializations lead to very similar final segmentations because of the label propagation velocity. More results are given in our Web page:

<http://www.csd.uoc.gr/~tziritas/cost.html>

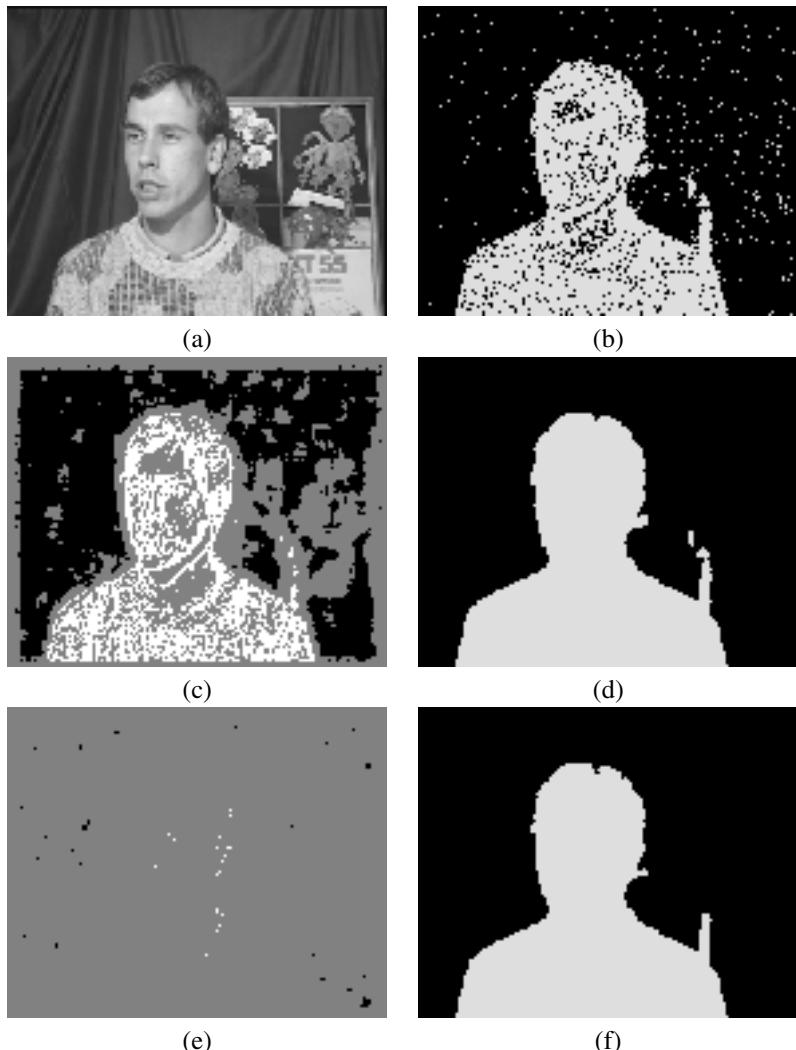


Figure 9.3. The segmentation result on the *Erik* sequence: (a) the first frame of the sequence, (b) the *maximum a posteriori* probability classification, (c) initialization of a large number of sites, (d) the corresponding segmentation, (e) initialization of a small number of sites, and (f) the corresponding segmentation.

9.5 Conclusion

A new level set-based framework for image segmentation was presented. The Multi-Label Fast Marching algorithm has been introduced for the propagation of high-confidence classification decisions in accordance with the *a posteriori* probability of the competing classes. Two specific segmentation applications are addressed in order to illustrate the usability of this new algorithm as a fast, precise and generic technique for unsupervised pixel labeling.

This approach's objective of maximizing the *a posteriori* probability is shared by other techniques, such as stochastic or deterministic relaxation. Such approaches utilize an objective function whose global minimum yields the optimal segmentation map. Concerning the algorithm presented here, the absence of a global objective function can be considered as a weakness, yet the dependence of the propagation velocities on the *a posteriori* probabilities of the competing classes clearly leads the segmentation process toward the same goal, while allowing an extremely efficient noniterative implementation.

In comparison with existing level set implementations of active contours, the Multilabel Fast Marching algorithm, as presented here, clearly lacks the capability of incorporating a smoothness constraint into the propagation process, thus sometimes resulting in noisy expansion of the moving contours. Nevertheless the competition of the expanding regions and their convergence over each other significantly reduce the amount of noise in the final region boundaries, while retaining a high level of localization precision. In addition, our algorithm can handle multiple segmentation classes with a complexity that outperforms most existing multiclass level set methods.

Inherent limitations and shortcomings of the proposed framework include the strong dependence of the segmentation quality on the extracted features, which must be both sufficient and discriminant. Additionally, the initial high-confidence classification decisions impose strict constraints on the morphology of the final regions. Subsequently, narrow or small and isolated parts of a given class exhibit an inherent difficulty of detection.

10

Multiphase Object Detection and Image Segmentation

Luminita Vese

Abstract

This chapter presents a class of techniques for object detection and image segmentation, using variational models formulated in a level set approach. We consider in particular Mumford and Shah like energies, whose minimizers are in the space of special functions of bounded variation. For such functions, all points are of two types: points where the functions have an approximate gradient, and points of discontinuities along curves or edges. The set of discontinuities is represented implicitly, using the level set method. Minimizing these energies in a level set formulation, yields coupled curve evolution and diffusion equations, which can be used for object detection and image segmentation. Finally, the proposed methods are validated by various numerical results in two dimensions.

10.1 Introduction

An important problem in image processing is the partition or segmentation of a given image u_0 into regions and their boundaries. Another closely related problem is the object detection by curve evolution and active contours (this is also called shape extraction or shape segmentation).

The image segmentation problem in computer vision is often posed as a variational problem. Given $u_0 : \Omega \rightarrow R$, with $\Omega \subset R^2$, the problem is to find an optimal piecewise smooth approximation u of u_0 , and a set of boundaries K , such that u varies smoothly within the connected components of $\Omega \setminus K$, and rapidly or discontinuously across K .

To solve this problem, D. Mumford and J. Shah [388] proposed the following minimization problem:

$$\inf_{u,K} \left\{ F^{MS}(u, K) = \int_{\Omega} |u - u_0|^2 dx + \mu \int_{\Omega \setminus K} |\nabla u|^2 dx + \nu \int_K d\mathcal{H}^1 \right\}, \quad (10.1)$$

where $\mu > 0$, $\nu > 0$ are fixed parameters, to weight the different terms in the energy. For (u, K) a minimizer of the above energy, u is an “optimal” piecewise smooth approximation of the initial, possibly noisy, image u_0 , and K has the role of approximating the edges of u_0 ; u will be smooth only outside K , i.e. on $\Omega \setminus K$. Here, \mathcal{H}^1 is the one-dimensional Hausdorff measure. Theoretical results of existence and regularity of minimizers of (10.1) can be found for example in [388], [379], [380], [381], [382].

A reduced case of the above model is obtained by restricting the segmented image u to piecewise constant functions, i.e. $u = \text{constant } c_i$ inside each connected component Ω_i of $\Omega \setminus K$. Then the problem is often called the “minimal partition problem”, and in order to solve it, in [388] it is proposed to minimize the following functional:

$$F_0^{MS}(u, K) = \sum_i \int_{\Omega_i} |u_0 - c_i|^2 dx + \nu \int_K d\mathcal{H}^1. \quad (10.2)$$

Again, ν is a positive parameter, having a scaling role. It is easy to see that, for a fixed closed set K (a finite union of curves), the energy (10.2) is minimized in the variables c_i by setting $c_i = \text{mean}(u_0)$ in Ω_i , or $c_i = \frac{\int_{\Omega_i} u_0(x) dx}{|\Omega_i|}$. Theoretical results for existence and regularity of minimizers of (10.2) can be found for example in [388], [346], [503], [504], [308].

It is not easy to minimize in practice the functionals (10.1) and (10.2), because of the unknown set K of lower dimension, and because these problems are not convex.

A weak formulation of (10.1) has been proposed in [145], where K is replaced by the set J_u of jumps of u , in order to prove the existence of minimizers. Also, it is known that a global minimizer of (10.1), or of the weak formulation, is not unique in general. In [379], [380], the authors proposed a constructive existence result for the weak formulation of the Mumford and Shah problem, and in [287], a practical multi-scale algorithm based on regions growing and merging is proposed in the piecewise-constant case. For a general exposition of the segmentation problem by variational methods, both in theory and practice, we refer the reader to [381], [382]. We also refer to [12], [65] for theoretical results on functionals defined on the appropriate space for image segmentation: the $SBV(\Omega)$ space of special functions of bounded variation, that will be introduced later.

Two elliptic approximations by Γ -convergence to the weak formulation of the Mumford and Shah functional have been proposed in [13], [14]. The authors approximated a minimizer (u, J_u) of $F^{MS}(u, J_u)$, by smooth functions (u_ρ, v_ρ) , such that, as $\rho \rightarrow 0$, we have $u_\rho \rightarrow u$ and $v_\rho \rightarrow 1$ in the $L^2(\Omega)$ -topology, and v_ρ

is different from 1 only in a small neighborhood of J_u , which shrinks as $\rho \rightarrow 0$. The elliptic approximations lead to a coupled system of two equations in the unknowns u_ρ and v_ρ , to which standard PDE numerical methods can be applied. Related approximations and numerical results can be found in [337], [92], [93], [94], [63], [62], [549]. Also, in [95], the authors provide an approximation by Γ – *convergence* based on the finite element method, to the weak formulation of the Mumford and Shah problem. Note that, most of these methods solving the weak formulation by elliptic approximations do not explicitly compute the partition of the image and the set of curves K . In general, only an approximation to K is obtained, by a sequence of regions enclosing K , but converging in the limit to the empty set.

As it was mentioned earlier, a related problem to image segmentation is the object detection problem by snakes and active contours. An initial curve evolves in the image under some speed, and it stops on boundaries of objects, for instance where the magnitude gradient of the image is large. Some of the most well known active contour models with edge-function are: [263], [81], [331], [85], [271], [585]. The active contour models using the gradient of the image for the stopping criteria are also called boundary based models.

Many active contour models use the level set method introduced by S. Osher and J. Sethian [401], to represent the evolving curve. This method works on a fixed rectangular grid, and allows for automatic topology changes, such as merging and breaking. A closed curve $K = \partial\omega \subset \Omega$ (with $\omega \subset \Omega$ an open subset), is represented by the zero level set of a Lipschitz-continuous function $\phi : \Omega \rightarrow R$, such that $K = \{x \in \Omega : \phi(x) = 0\}$, and $\phi > 0$ on one side of K and $\phi < 0$ on the other side of K . These properties can be expressed as follows:

$$\begin{aligned}\phi(x) &= 0 \text{ if } x \in K, \\ \phi(x) &> 0 \text{ if } x \in \omega, \\ \phi(x) &< 0 \text{ if } x \in \Omega \setminus \overline{\omega}.\end{aligned}$$

More recently, new active contour and level set methods have been proposed for image segmentation, some of them involving region based techniques, in addition to the previous boundary based techniques. Among these methods, we would like to mention those closely related with the present work.

First, the present work is a summary and generalization of the active contour model without edges and its generalizations to segmentation of images, from [100], [101], [103], [102] and [551]. These work propose a level set method for active contours and segmentation via the Mumford and Shah model [388].

Other closely related variational level set methods are: “Inward and outward curve evolution using level set method”, from [11]; “A variational level set approach to multiphase motion” from [607]; “A level set model for image classification”, from [456], [457]; “A statistical approach to snakes for bimodal and trimodal imagery”, from [593]; “A Fully Global Approach to Image Segmentation via Coupled Curve Evolution Equations”, from [594]; “Binary Flows and Image Segmentation”, from [592]; “Geodesic Active Regions: A New Framework to Deal with Frame Partition Problems in Computer Vision”, from [411]; “Cou-

pled geodesic active regions for image segmentation: a level set approach”, from [413]. Finally, a closely related work is “Curve evolution implementation of the Mumford-Shah functional for image segmentation, denoising, interpolation, and magnification”, from [528].

Many other contributions have been proposed for image segmentation by variational or PDE methods, and it is impossible to mention all of them. However, we would like to give a few more references: “Region competition: Unifying snakes, region growing, and Bayes/MDL for multi-band image segmentation”, from [611], [613]; “Normalized cuts and image segmentation”, from [479]; “A common framework for curve evolution, segmentation and anisotropic diffusion”, from [477]; “Riemannian Drums, Anisotropic Curve Evolution and Segmentation”, from [478]; “Filtering, Segmentation and Depth”, from [386]; “Codimension-Two Geodesic Active Contours for the Segmentation of Tubular Structures”, from [322].

As we have already mentioned, the proposed approach follows and generalizes the ideas from [100], [101], [97], [103], [102], [551], for image segmentation and object detection by a variational level set approach. First, we extend the piecewise constant models from [100], [101], [551], to piecewise linear versions. We also consider other general anisotropic Mumford and Shah like functionals, where the energy term along K also depends on the jump of u .

10.2 Variational models for image segmentation and image partition

We will consider here a class of functionals with solutions in the space $SBV(\Omega)$, with $\Omega \subset R^2$ an open and bounded set. For a general exposition of such functionals in a weak formulation, we refer the reader to [65] and [12]. By definition [12], [65], a function $u \in L^1(\Omega)$ is a special function of bounded variation, if its distributional derivative can be written as $Du = \nabla u dx + (u^+ - u^-)n_u \chi_K \mathcal{H}^1|_K$, where ∇u is called the approximate gradient of u at $x \in \Omega \setminus K$, and K is a set of finite 1-dimensional Hausdorff measure. So, for any point $x \in \Omega$, either u has an approximate gradient ∇u at x , or x is a jump point of u , with distinct approximate limits $u^+(x)$ and $u^-(x)$ of u on each side of K . Here, n_u denotes the exterior unit normal to K at every point $x \in K$, where $|u^+(x) - u^-(x)| > 0$, and $\mathcal{H}^1|_K$ is the restriction of the measure \mathcal{H}^1 to K . The space of special functions of bounded variation is denoted by $SBV(\Omega)$.

A class of functionals defined on $SBV(\Omega)$ is given by [65]:

$$F(u, K) = \int_{\Omega} |u - u_0|^p dx + \mu \int_{\Omega \setminus K} f(\nabla u) dx + \nu \int_K g(|u^+ - u^-|) d\mathcal{H}^1, \quad (10.3)$$

where $p \geq 1$, $f : R^2 \rightarrow [0, +\infty)$ is a convex function satisfying

$$\lim_{|z| \rightarrow \infty} \frac{f(z)}{|z|} = +\infty,$$

and $g : [0, \infty) \rightarrow [0, \infty)$ is a sub-additive and increasing function satisfying

$$\lim_{t \rightarrow 0} \frac{g(t)}{t} = \infty.$$

The function g is sub-additive in the following sense: $g(t_1 + t_2) \leq g(t_1) + g(t_2)$, for all $t_1, t_2 \geq 0$. The assumptions on the functions f and g are necessary for existence results of minimizers on the $SBV(\Omega)$ space.

$F(u, K)$ reduces to the Mumford and Shah functional [388] when $p = 2$, $f(\nabla u) = |\nabla u|^2$ and $g(|u^+ - u^-|) = 1$ is a constant function.

Other examples of functionals, in addition to the Mumford and Shah functional, can be obtained with: $f(\nabla u) = |\nabla u|^q$, with $q > 1$, and $g(|u^+ - u^-|) = \sqrt{|u^+ - u^-|}$.

Given $u_0 \in L^\infty(\Omega)$, the minimization problem

$$\inf_{u, K} F(u, K)$$

can be seen as a partitioning or segmentation problem of the given image u_0 . If (u, K) is a minimizer, then the connected components of $\Omega \setminus K$ will be smooth regions or objects in the image, while the closed set K of lower dimension will represent edges, contours or boundaries of objects and of regions.

In the case when the function g is not constant and depends on the magnitude of the jump $|u^+ - u^-|$, then the quantity $g(|u^+ - u^-|)$ along the discontinuity set K of contours plays an additional anisotropic scaling role.

In the next section we will show that, by representing the set K using the level set method, and minimizing the above energy, we obtain interesting coupled curve evolution and diffusion equations. The obtained level set models can be used for object detection, denoising, image partition and segmentation.

10.3 Level set formulations of minimization problems on $SBV(\Omega)$

In this section, we present several extensions and generalizations of the results from [100], [101], [97], [103], [102], [551]. We follow the notations and terminology from [607], [101].

In what follows, to a level set function $\phi : \Omega \rightarrow R$, we associate the Heaviside function $H(\phi)$ defined by: $H(\phi(x)) = 1$ if $\phi(x) \geq 0$ and $H(\phi(x)) = 0$ if $\phi(x) < 0$.

Let us consider various subsets of functions u having discontinuities only along $K = \{x \in \Omega : \phi(x) = 0\}$, defined as follows:

$$\left\{ u(x) = P^+(x)H(\phi(x)) + P^-(x)(1 - H(\phi(x))) \right\}, \quad (10.4)$$

with P^+, P^- polynomials of degree at most m (in this paper, we will consider only constant and linear polynomials, i.e. $m = 0$ and $m = 1$). P^+ is defined on $\{x \in \Omega : \phi(x) \geq 0\}$ and P^- is defined on $\{x \in \Omega : \phi(x) \leq 0\}$. Another subset of functions is

$$\left\{ u(x) = u^+(x)H(\phi(x)) + u^-(x)(1 - H(\phi(x))) \right\}, \quad (10.5)$$

with u^+, u^- functions, such that $u^+ \in C^1(\{x \in \Omega : \phi(x) \geq 0\})$, and $u^- \in C^1(\{x \in \Omega : \phi(x) \leq 0\})$.

As in [103], [102], [551], in order to represent triple junctions and more complex topologies, we can also consider the following subsets of functions, where discontinuities are along a set defined using two level set functions ϕ_1 and ϕ_2 :

$$K = \{x \in \Omega : \phi_1(x) = 0\} \cup \{x \in \Omega : \phi_2(x) = 0\}.$$

Then similarly, the corresponding subsets of functions u will be:

$$\begin{aligned} \left\{ u(x) = & P^{++}(x)H(\phi_1(x))H(\phi_2(x)) \right. \\ & + P^{+-}(x)H(\phi_1(x))(1 - H(\phi_2(x))) \\ & + P^{-+}(x)(1 - H(\phi_1(x)))H(\phi_2(x)) \\ & \left. + P^{--}(x)(1 - H(\phi_1(x)))(1 - H(\phi_2(x))) \right\}, \end{aligned} \quad (10.6)$$

with $P^{++}, P^{+-}, P^{-+}, P^{--}$ polynomials of degree at most m , defined respectively on $\{x \in \Omega : \phi_1(x) \geq 0, \phi_2(x) \geq 0\}$, $\{x \in \Omega : \phi_1(x) \geq 0, \phi_2(x) \leq 0\}$, $\{x \in \Omega : \phi_1(x) \leq 0, \phi_2(x) \geq 0\}$, $\{x \in \Omega : \phi_1(x) \leq 0, \phi_2(x) \leq 0\}$, and

$$\begin{aligned} \left\{ u(x) = & u^{++}(x)H(\phi_1(x))H(\phi_2(x)) \right. \\ & + u^{+-}(x)H(\phi_1(x))(1 - H(\phi_2(x))) \\ & + u^{-+}(x)(1 - H(\phi_1(x)))H(\phi_2(x)) \\ & \left. + u^{--}(x)(1 - H(\phi_1(x)))(1 - H(\phi_2(x))) \right\}, \end{aligned} \quad (10.7)$$

with $u^{++}, u^{+-}, u^{-+}, u^{--}$ C^1 functions, defined respectively on the following subsets: $\{x \in \Omega : \phi_1(x) \geq 0, \phi_2(x) \geq 0\}$, $\{x \in \Omega : \phi_1(x) \geq 0, \phi_2(x) \leq 0\}$, $\{x \in \Omega : \phi_1(x) \leq 0, \phi_2(x) \geq 0\}$, $\{x \in \Omega : \phi_1(x) \leq 0, \phi_2(x) \leq 0\}$.

In what follows, we will write and minimize the energy $F(u, K)$ from (10.3) restricted to the subsets defined above, and we will solve some of these minimizations in a few particular cases.

For instance, the minimization of (10.3) restricted to the subset from (10.5) can be written as:

$$\begin{aligned} & \inf_{u^+, u^-, \phi} F(u^+, u^-, \phi) \\ = & \int_{\Omega} |u^+ - u_0|^p H(\phi) dx + \int_{\Omega} |u^- - u_0|^p (1 - H(\phi)) dx \\ + & \mu \int_{\Omega} f(\nabla u^+) H(\phi) dx + \mu \int_{\Omega} f(\nabla u^-) (1 - H(\phi)) dx \\ + & \nu \int_{\Omega} g(|u^+ - u^-|) |\nabla H(\phi)|, \end{aligned}$$

and similarly in the other cases. Then, writing the Euler-Lagrange equations associated with the minimization problem, interesting coupled curve evolution and diffusion equations will be obtained, with applications to object detection and image segmentation.

For the purpose of illustration, we will only consider the following particular cases: $p = 2$, $f(\nabla u) = |\nabla u|^2$, $g(|u^+ - u^-|) = 1$ and $g(|u^+ - u^-|) = \sqrt{|u^+ - u^-|}$. These particular cases yield two functionals. The first one is the isotropic classical Mumford and Shah energy [388]

$$F^{MS}(u, K) = \int_{\Omega} |u - u_0|^2 dx + \mu \int_{\Omega \setminus K} |\nabla u|^2 dx + \nu \int_K d\mathcal{H}^1. \quad (10.8)$$

The second one is an “anisotropic” Mumford and Shah like energy

$$E^{MS}(u, K) = \int_{\Omega} |u - u_0|^2 dx + \mu \int_{\Omega \setminus K} |\nabla u|^2 dx + \nu \int_K \sqrt{|u^+ - u^-|} d\mathcal{H}^1. \quad (10.9)$$

Let us now consider only these last two energies, restricted to some of the sets mentioned above. We will explicitly write in each case the form of the minimization problem and the associated Euler-Lagrange equations. We will parametrize by an artificial time the descent direction in ϕ , ϕ_1 , or ϕ_2 . We will also regularize the Heaviside function H by $H_\varepsilon \in C^1(R)$ as $\varepsilon \rightarrow 0$, as in [100], [101], where H_ε is defined by:

$$H_\varepsilon(s) = \frac{1}{2} \left(1 + \frac{2}{\pi} \arctan\left(\frac{s}{\varepsilon}\right) \right).$$

We will also use the notation $\delta_\varepsilon = H'_\varepsilon$ for an approximation and regularization of the one-dimensional Dirac function δ_0 , concentrated at the origin.

Minimizing the restriction of (10.8) to the subset

$$\left\{ u(x) = c^+ H(\phi(x)) + c^- (1 - H(\phi(x))) \right\},$$

with c^+, c^- unknown (polynomials of degree 0), yields:

$$\inf_{c^+, c^-, \phi} \int_{\Omega} |u_0(x) - c^+|^2 H(\phi) dx + \int_{\Omega} |u_0(x) - c^-|^2 (1 - H(\phi)) dx \\ + \nu \int_{\Omega} |\nabla H(\phi)|, \quad (10.10)$$

i.e. the active contour model without edges from [100], [101]. The minimizers have to satisfy the following coupled equations, given $\phi(0, x) = \phi_0(x)$:

$$\begin{aligned} c^+(\phi(t, \cdot)) &= \frac{\int_{\Omega} u_0(x) H(\phi(x)) dx}{\int_{\Omega} H(\phi(x)) dx}, \\ c^-(\phi(t, \cdot)) &= \frac{\int_{\Omega} u_0(x) (1 - H(\phi(x))) dx}{\int_{\Omega} (1 - H(\phi(x))) dx}, \\ \frac{\partial \phi}{\partial t} &= \delta_{\varepsilon}(\phi) \left[\nu \operatorname{div} \left(\frac{\nabla \phi}{|\nabla \phi|} \right) - |u_0 - c^+|^2 + |u_0 - c^-|^2 \right]. \end{aligned}$$

A numerical result obtained with this model is shown in Figure 10.1. This model performs active contours, and has the following advantages, when compared with boundary-based models: it automatically detects interior contours; it detects both contours with or without gradient; the position of the initial curve can be anywhere in the image.

An extension of the previous model is introduced here, which is obtained by considering linear approximations: minimizing the restriction of (10.8) to the subset

$$\left\{ u(x) = ((a^+, b^+) \cdot x + c^+) H(\phi(x)) + ((a^-, b^-) \cdot x + c^-) (1 - H(\phi(x))) \right\},$$

with $a^+, b^+, c^+, a^-, b^-, c^-$ unknown (as coefficients of polynomials of degree 1), yields:

$$\begin{aligned} \inf_{a^+, b^+, c^+, a^-, b^-, c^-, \phi} &\int_{\Omega} \left| u_0(x) - ((a^+, b^+) \cdot x + c^+) \right|^2 H(\phi) dx \\ &+ \int_{\Omega} \left| u_0(x) - ((a^-, b^-) \cdot x + c^-) \right|^2 (1 - H(\phi)) dx \\ &+ \mu \int_{\Omega} ((a^+)^2 + (b^+)^2) H(\phi) dx + \mu \int_{\Omega} ((a^-)^2 + (b^-)^2) (1 - H(\phi)) dx \\ &+ \nu \int_{\Omega} |\nabla H(\phi)|. \quad (10.11) \end{aligned}$$

The minimizers have to satisfy the following coupled equations, given $\phi(0, x) = \phi_0(x)$ (note the linear algebraic systems for (a^+, b^+, c^+) and

(a^-, b^-, c^-)):

$$\begin{aligned} a^+ \int_{\Omega} (x_1^2 + \mu) H(\phi(x)) dx + b^+ \int_{\Omega} x_1 x_2 H(\phi(x)) dx + c^+ \int_{\Omega} x_1 H(\phi(x)) dx \\ = \int_{\Omega} x_1 u_0(x) H(\phi(x)) dx, \\ a^+ \int_{\Omega} x_1 x_2 H(\phi(x)) dx + b^+ \int_{\Omega} (x_2^2 + \mu) H(\phi(x)) dx + c^+ \int_{\Omega} x_2 H(\phi(x)) dx \\ = \int_{\Omega} x_2 u_0(x) H(\phi(x)) dx, \\ a^+ \int_{\Omega} x_1 H(\phi(x)) dx + b^+ \int_{\Omega} x_2 H(\phi(x)) dx + c^+ \int_{\Omega} H(\phi(x)) dx \\ = \int_{\Omega} u_0(x) H(\phi(x)) dx, \end{aligned}$$

(similarly for (a^-, b^-, c^-) , substituting $H(\phi)$ by $(1 - H(\phi))$, and

$$\begin{aligned} \frac{\partial \phi}{\partial t} = \delta_{\varepsilon}(\phi) \left[\nu \operatorname{div} \left(\frac{\nabla \phi}{|\nabla \phi|} \right) - \left| u_0(x) - \left((a^+, b^+) \cdot x + c^+ \right) \right|^2 \right. \\ \left. + \left| u_0(x) - \left((a^-, b^-) \cdot x + c^- \right) \right|^2 \right. \\ \left. - \mu ((a^+)^2 + (b^+)^2) + \mu ((a^-)^2 + (b^-)^2) \right]. \end{aligned}$$

Numerical results obtained with this model are presented in Figures 10.4 and 10.5. This linear case has also been discussed by the author with P. Hamilton, during a collaboration on medical image segmentation.

Minimizing the restriction of (10.8) to the subset

$$\left\{ u(x) = u^+(x)H(\phi(x)) + u^-(x)(1 - H(\phi(x))) \right\},$$

yields (see [102], [551]):

$$\begin{aligned} \inf_{u^+, u^-, \phi} & \int_{\Omega} |u^+ - u_0|^2 H(\phi) dx + \int_{\Omega} |u^- - u_0|^2 (1 - H(\phi)) dx \quad (10.12) \\ & + \mu \int_{\Omega} |\nabla u^+|^2 H(\phi) dx + \mu \int_{\Omega} |\nabla u^-|^2 (1 - H(\phi)) dx + \nu \int_{\Omega} |\nabla H(\phi)|. \end{aligned}$$

The minimizers have to satisfy the following coupled equations, given the initial condition $\phi(0, x) = \phi_0(x)$:

$$u^+ = u_0 + \mu \Delta u^+ \text{ on } \{\phi > 0\}, \frac{\partial u^+}{\partial \mathbf{n}} = 0 \text{ on } \{\phi = 0\} \cup \partial \Omega,$$

$$u^- = u_0 + \mu \Delta u^- \text{ on } \{\phi < 0\}, \frac{\partial u^-}{\partial \mathbf{n}} = 0 \text{ on } \{\phi = 0\} \cup \partial \Omega,$$

$$\frac{\partial \phi}{\partial t} = \delta_{\varepsilon}(\phi) \left[\nu \operatorname{div} \left(\frac{\nabla \phi}{|\nabla \phi|} \right) - |u^+ - u_0|^2 + |u^- - u_0|^2 - \mu |\nabla u^+|^2 + \mu |\nabla u^-|^2 \right].$$

We would like to mention that this last case was also proposed and solved independently by [528]. Figure 10.2 shows a numerical result using this case.

Let us now consider the restriction of (10.8) to the subset

$$\begin{cases} u(x) = u^{++}(x)H(\phi_1(x))H(\phi_2(x)) \\ + u^{+-}(x)H(\phi_1(x))(1 - H(\phi_2(x))) \\ + u^{-+}(x)(1 - H(\phi_1(x)))H(\phi_2(x)) \\ + u^{--}(x)(1 - H(\phi_1(x)))(1 - H(\phi_2(x))) \end{cases}. \quad (10.13)$$

In this case, the minimization problem can be written as:

$$\begin{aligned} & \inf_{u^{++}, u^{+-}, u^{-+}, u^{--}, \phi_1, \phi_2} \int_{\Omega} \left[|u^{++} - u_0|^2 H(\phi_1)H(\phi_2) \right. \quad (10.14) \\ & + |u^{+-} - u_0|^2 H(\phi_1)(1 - H(\phi_2)) + |u^{-+} - u_0|^2 (1 - H(\phi_1))H(\phi_2) \\ & \quad \left. + |u^{--} - u_0|^2 (1 - H(\phi_1))(1 - H(\phi_2)) \right] dx \\ & + \mu \int_{\Omega} \left[|\nabla u^{++}|^2 H(\phi_1)H(\phi_2) + |\nabla u^{+-}|^2 H(\phi_1)(1 - H(\phi_2)) \right. \\ & + |\nabla u^{-+}|^2 (1 - H(\phi_1))H(\phi_2) + |\nabla u^{--}|^2 (1 - H(\phi_1))(1 - H(\phi_2)) \left. \right] dx \\ & + \nu \int_{\Omega} |\nabla H(\phi_1)| + \nu \int_{\Omega} |\nabla H(\phi_2)| \left((1 - H(-\phi_1)) + (1 - H(\phi_1)) \right). \end{aligned}$$

Note that the term $\left((1 - H(-\phi_1)) + (1 - H(\phi_1)) \right)$ is used to avoid counting more than once the segments of curves belonging to both $\{\phi_1 = 0\}$ and $\{\phi_2 = 0\}$. The minimizers $(u^{++}, u^{+-}, u^{-+}, u^{--}, \phi_1, \phi_2)$ satisfy coupled curve evolution and diffusion equations, similar with those from the previous case. We think that in this case, based on the Four Color Theorem, defining the set of minimizers by this set with four functions $u^{++}, u^{+-}, u^{-+}, u^{--}$ and with only two level set functions, should formally suffice to represent any case. The connection with the Four Color Theorem in image segmentation has been also made in [571]. A numerical result obtained in this case is presented in Figure 10.9, from [551].

Finally, for the purpose of illustration, let us consider one example of minimization problem involving the anisotropic energy (10.9). The minimization of the energy (10.9) restricted for instance to the subset

$$\left\{ u(x) = c^+ H(\phi(x)) + c^- (1 - H(\phi(x))) \right\},$$

with c^+, c^- unknown (polynomials of degree 0), yields:

$$\begin{aligned} & \inf_{c^+, c^-, \phi} \int_{\Omega} |u_0(x) - c^+|^2 H(\phi(x)) dx + \int_{\Omega} |u_0(x) - c^-|^2 (1 - H(\phi(x))) dx \\ & + \nu \sqrt{|c^+ - c^-|} \int_{\Omega} |\nabla H(\phi(x))|. \quad (10.15) \end{aligned}$$

The minimizers have to satisfy the following coupled equations, given the initial condition $\phi(0, x) = \phi_0(x)$:

$$\begin{aligned} c^+ &= \frac{\int_{\Omega} u_0(x) H(\phi(x)) dx}{\int_{\Omega} H(\phi(x)) dx} - \frac{sgn(c^+ - c^-)}{4\sqrt{|c^+ - c^-|}} \frac{\int_{\Omega} |\nabla H(\phi)|}{\int_{\Omega} H(\phi(x)) dx}, \\ c^- &= \frac{\int_{\Omega} u_0(x)(1 - H(\phi(x))) dx}{\int_{\Omega} (1 - H(\phi(x))) dx} - \frac{sgn(c^- - c^+)}{4\sqrt{|c^+ - c^-|}} \frac{\int_{\Omega} |\nabla H(\phi)|}{\int_{\Omega} (1 - H(\phi(x))) dx}, \\ \frac{\partial \phi}{\partial t} &= \delta_{\varepsilon}(\phi) \left[\nu \sqrt{|c^+ - c^-|} \operatorname{div} \left(\frac{\nabla \phi}{|\nabla \phi|} \right) - |u_0 - c^+|^2 + |u_0 - c^-|^2 \right]. \end{aligned}$$

A numerical result obtained using this model is presented in Figure 10.3. If the coefficient ν is the same as in the isotropic case (10.10), this new anisotropic model has a stronger constraint on the length term. In addition, in the general case, the presence of the additional factor $\sqrt{|u^+ - u^-|}$ in the energy term along K should remove the limitations on the type of edges obtained by the isotropic Mumford and Shah model. Some of these limitations are [388]: junctions can be only triple junctions with angles of 120° , and if one edge intersects the boundary $\partial\Omega$, it has to be at right angle.

Similarly, the anisotropic Mumford and Shah like energy (10.9) can also be written for the other cases.

The general functional in (10.3) has the advantage of having minimizers on the space $SBV(\Omega)$, the appropriate space for image segmentation: for $u \in SBV(\Omega)$, each point $x \in \Omega$ will be either a point in a homogeneous region, or an edge point. Unfortunately, the minimization of (10.3) is not convex, and it is not always guaranteed that the numerical algorithm will converge to a global minimizer. In addition, the global minimizer is not unique in general. To overcome this difficulties, an idea could be to consider convex minimization problems on the larger space $BV(\Omega)$ of functions of bounded variation, and to apply the same level set techniques. An example of such convex functional is given by the total variation minimization [449]. In [552], a similar level set method is proposed to the minimization of this energy, again with applications to active contours and image segmentation.

10.4 Experimental results

We present in this section some numerical results obtained with the models from the previous section. For the details of the numerical schemes and for other numerical results, we refer the reader to [100], [101], [97], [102], [103], [551].

As we will see in this section, these models have the abilities of automatic detection of interior contours, of detection of contours with or without gradient, and of detection and representation of complex topologies. The multiphase level set approach employed here has been introduced in [102], [103], [551], and has the advantages of always keeping the multiple phases disjoint and with their union the

entire domain, by definition. Triple junctions can be represented, with an optimal number of level set functions.

In Figure 10.1 we show a result obtained using the model (10.10). In Figure 10.2, we show a result obtained using the model (10.12). In Figures 10.3 and 10.6, the anisotropic model from (10.15) is used. In Figures 10.4 and 10.5, we have used the linear approximation model from (10.11). In Figure 10.7, the linear approximation model in a four phase fashion is used, with two level set functions (the obtained final four segments are shown in Figure 10.8). Finally, in Figure 10.9, a numerical result is presented using the general four-phase Mumford and Shah level set algorithm from (10.14), previously introduced in [102], [551].

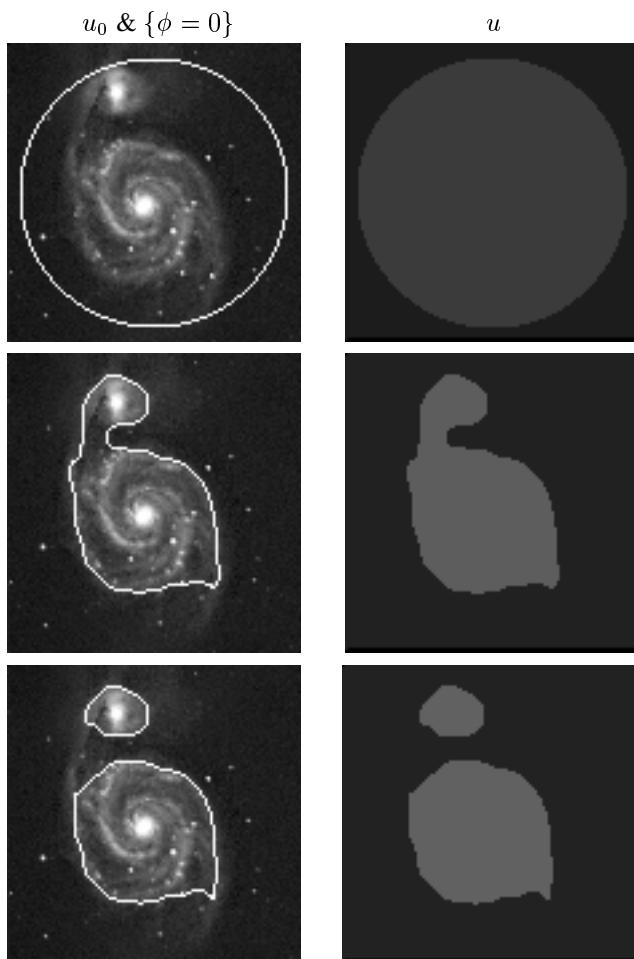


Figure 10.1. Numerical result using the piecewise-constant Mumford and Shah level set algorithm from (10.10), with $\nu = 0.09 * 255^2$.

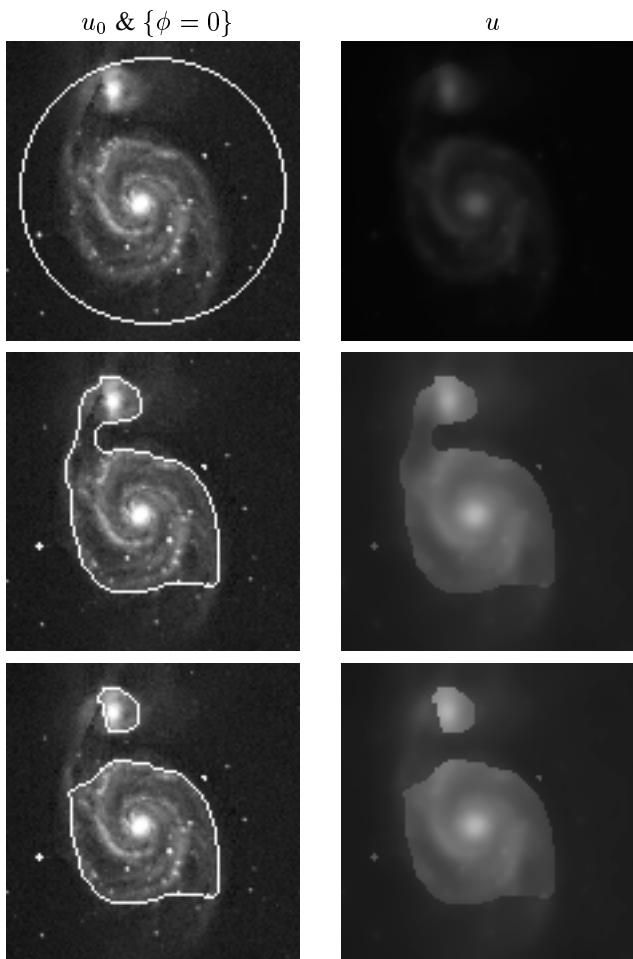


Figure 10.2. Numerical result using the piecewise-smooth Mumford and Shah level set algorithm from (10.12), with $\nu = 0.0305 * 255^2$ and $\mu = 10$.

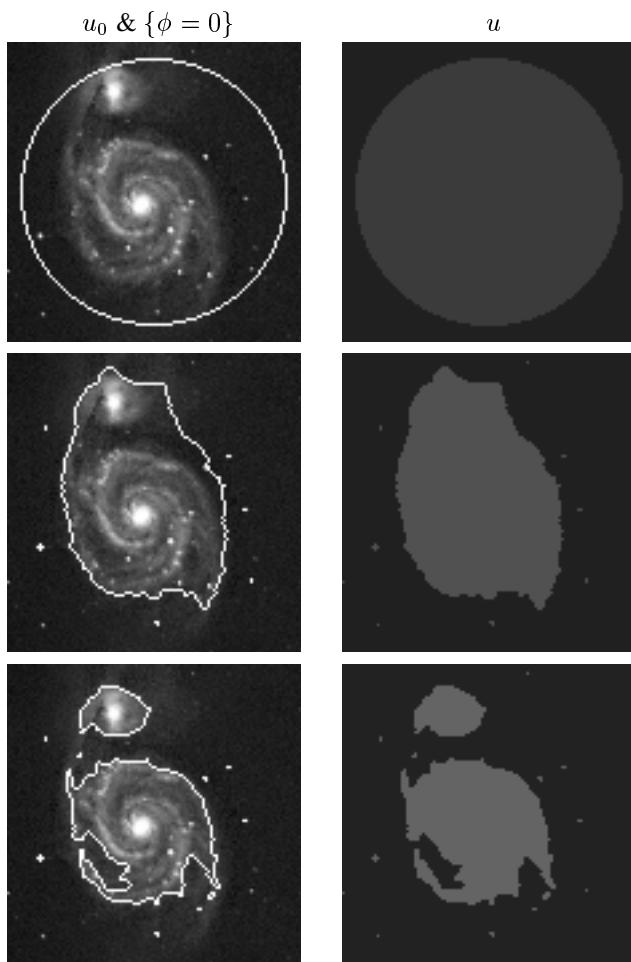


Figure 10.3. Numerical result using the piecewise-constant anisotropic Mumford and Shah like level set algorithm from (10.15), with $\nu = 0.0015 * 255^2$.

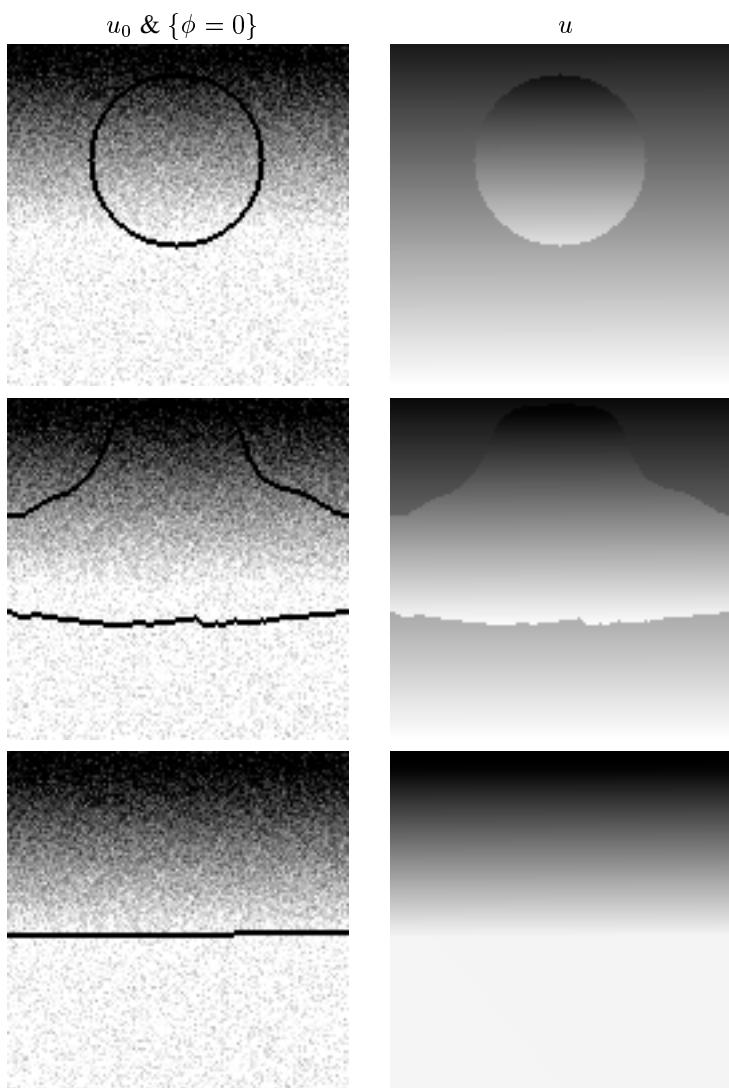


Figure 10.4. Numerical result using the piecewise-linear Mumford and Shah level set from (10.11), with $\mu = 1$, $\nu = 0.1 * 255^2$.

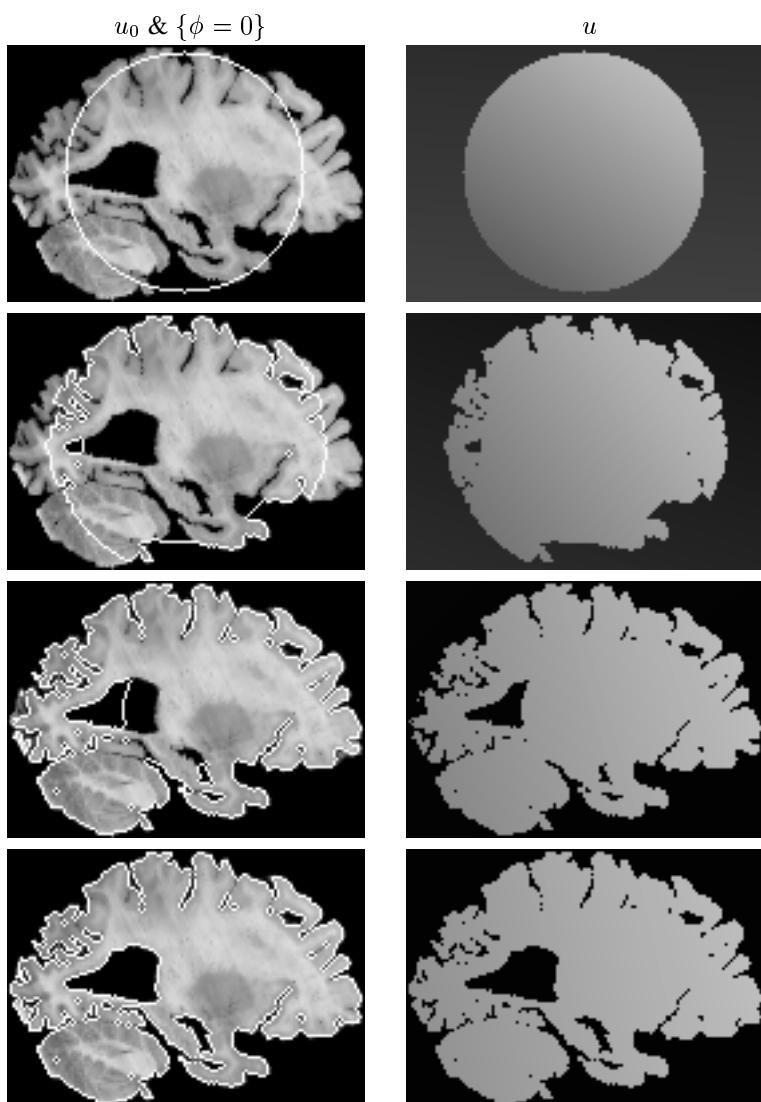


Figure 10.5. Numerical result using the piecewise-linear Mumford and Shah level set algorithm from (10.11), with $\mu = 0.001$ and $\nu = 0.001 * 255^2$.

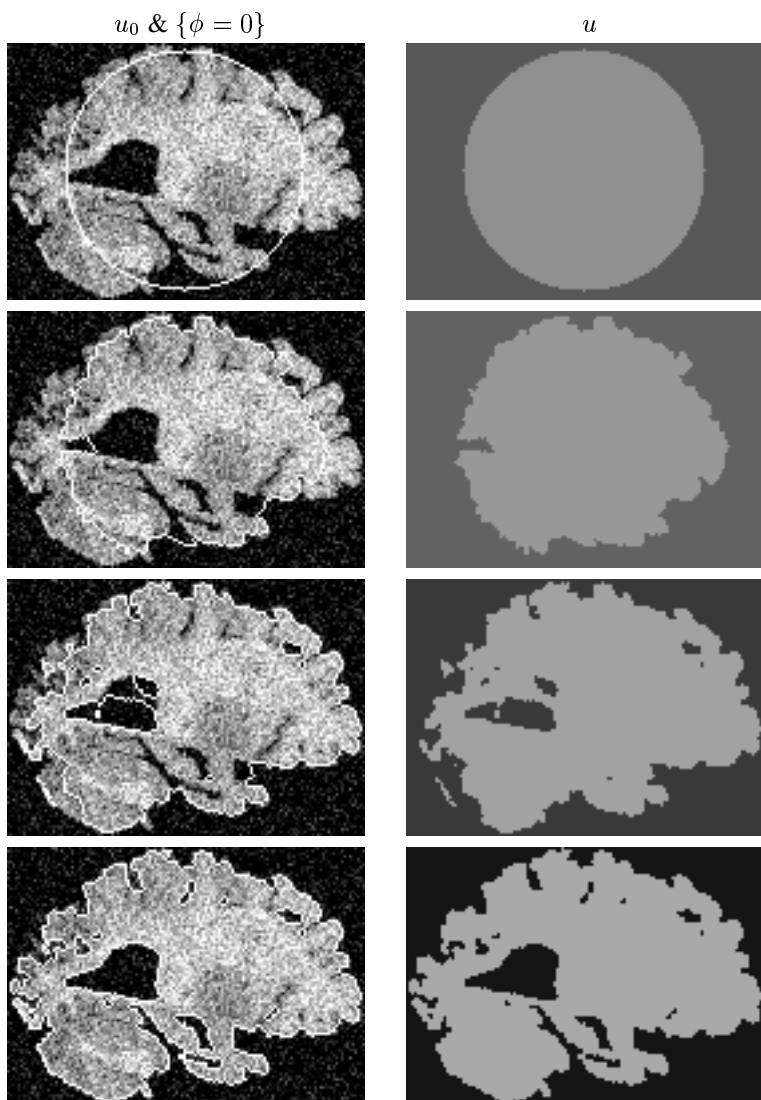


Figure 10.6. Numerical result using the piecewise-constant anisotropic Mumford and Shah like level set algorithm from (10.15), with $\nu = 0.01 * 255^2$.



Figure 10.7. Numerical result using the piecewise-linear four phase Mumford and Shah level set algorithm, with $\mu = 1$, $\nu = 0.014 \cdot 255^2$.



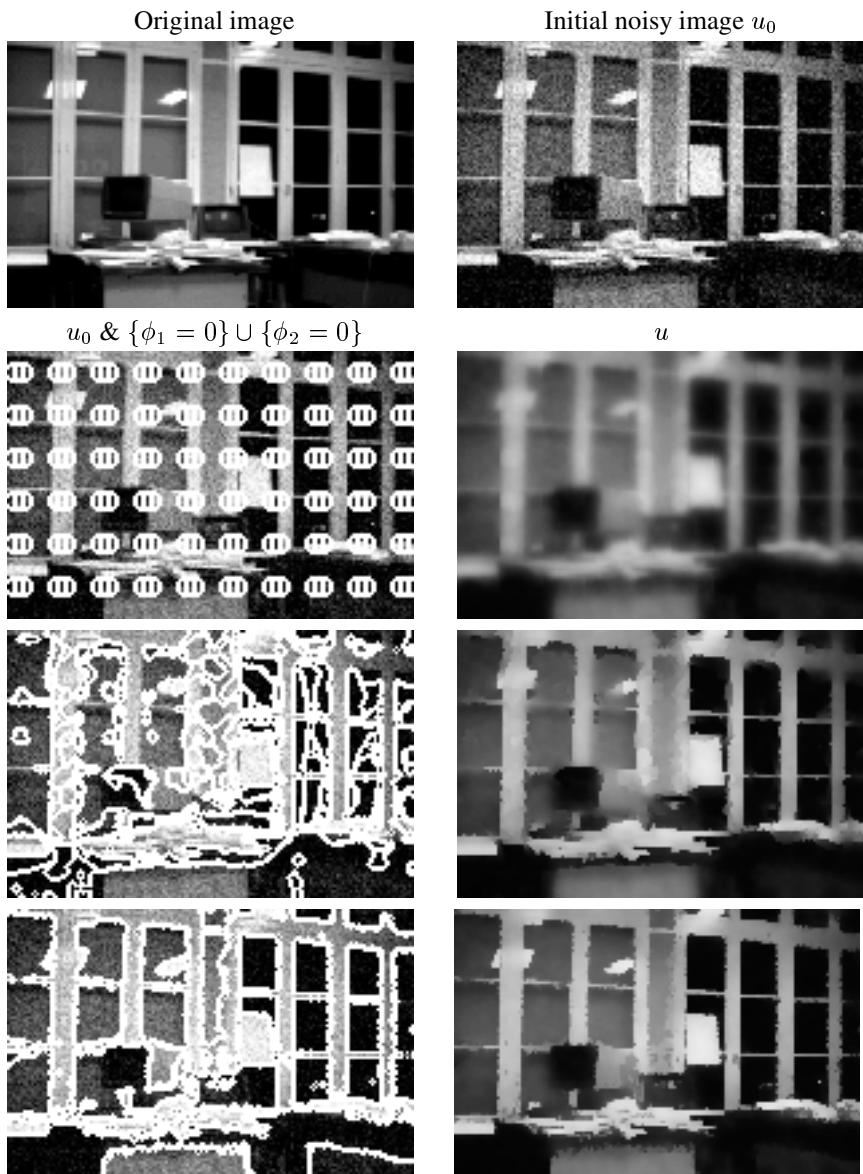
Figure 10.8. Final four segments obtained for the result from Figure 10.7.

10.5 Conclusion

We have presented in this chapter a level set technique for the minimization of a class of functionals defined on the space of special functions of bounded variation, arising in image segmentation. The obtained variational level set models yield coupled geometric and diffusion partial differential equations. Applications to object detection and image segmentation have been illustrated.

Acknowledgments

The author would like to thank the editors Stanley Osher and Nikos Paragios, for the invitation to contribute to this book. This work has been supported in part by grants NSF ITR-0113439, ONR N00014-02-1-0015, and NIH P20MH65166.



11

Adaptive Segmentation of Vector Valued Images

Mikaël Rousson and Rachid Deriche

Abstract

In the recent years, various forms of integration of different visual cues within variational formulations based on the propagation of planae curves was considered for image segmentation as a step forward to early gradient-based methods. To this end, appearence-driven information as well as prior knowledge on the form of the structures of interest to be recovered were introduced to improve segemntation performance. The main concern for using such modules is the requirement of having them formed off-line, either through pre-processing or defined by the user. In the recent literature, one can find segmentation methods that do not require neither intensity or shape-based prior knowledge. Going in this direction, in this chapter - driven by [413, 97] -, we propose a vector-valued segmentation method where the optimal partition and the statistical parameters of each class are dynamically recovered during the segmentation process.

11.1 Introduction

Considering the last studies in the domain, a variational formulation seems adequate to tackle the problem of segmenting images. In particular, this approach has shown its ability to integrate various cues in a common framework. Hence, different kind of information can be used as direct constraint on the boundaries (regularity, tension, high gradient, shape prior) or indirect constraint induced by modules accounting for region integrity (homogeneity [97], texture classification [414], prior on spatial intensity distributions [309],...).

In most of the cases, a preprocessing step is necessary to extract the relevant information from the image before the partitioning process itself. Different types of information can be extracted: the creation of an edge image can provide boundary

information, data clustering may be useful to determine a simple region module, complex filtering can help in extracting texture characteristics...

We present a method that do not need prior information. An entirely variational formulation is proposed, where region cues extraction is made jointly with the partitioning process. Recent related works can be found. In [97], the authors propose to use the mean to distinguish the regions, the mean of the regions being dynamically estimated during the evolution of the curve which delimits the regions. They generalize their model to deal with vector-valued data.

In [255], the criteria used to separate two smooth regions is their entropy, this one depending again on the moving border position. In [593], the authors have also proposed similar approaches. Bimodal images are segmented using Gaussian distributions with fixed variances and adaptative means. More recently, a non-parametric method using information theory have been presented in [272].

The method proposed in the next section can be seen as a generalization of parts of the models in [97, 413]. We generalize part of the work in [413] because we work on segmenting multi-valued images while in [413], only scalar images were considered but for different applications (texture, motion and segmentation). The *CV model* presented in [97] is also generalized because we deal with full covariance matrices and not only the Identity as in [97]. In [97], only the means of the regions are considered as unkown while here the unkown are the means and the entire covariance matrices. This gives us, for instance, the ability to segment two regions with same means but different variances. Actually, a general variational formulation is obtained from the maximization of the a posteriori segmentation probability given an observed data.

First, limiting our study to the 2-phase case, we will present two different ways of minimizing the functional driving both to the same evolution equations. One use a modified energy by an early introduction of level set functions, whereas the other method consists in differentiating directly the functional using recent results on shape derivation. Next, we will show some experimental results on gray-valued and color images, and also on sequences of images. Then, we will finish by making some comments on the limitations of the method and the possible extensions to an arbitrary number of regions.

11.2 The segmentation problem

Segmenting images consists of finding a partition of an observed data I into homogeneous regions. These regions are characterized by statistical properties which represent a visual consistency. The interface between the regions is assumed to be smooth.

Let $\Omega \in \mathbb{R}^2$ be open and bounded, and let $I : \Omega \rightarrow \mathbb{R}^p$ be the observed data. Let $\mathcal{P}(\Omega)$ be a partition of the domain and $\partial\Omega$ be the boundaries between the regions. We make the following assumptions: 1) I is composed by a maximum of

N regions Ω_i verifying an hypothesis h_i , 2) the interface between the regions $\partial\Omega$ is regular.

Let $p_i(I(x))$ be the conditional probability density function of a given value $I(x)$ with respect to the hypothesis h_i . The good segmentation of a given observed data respecting the hypotheses is obtained by solving the optimization problem with respect to the *a posteriori* segmentation probability, given the observation set: $p(P(\Omega)|I)$.

Following the Geodesic Active Regions [413], we can reformulate the problem in terms of energy. Only two assumptions are made in [413]: all the partitions are equally possible and the pixels within each region are independent. Since these assumptions are reasonable in our case, the optimal frame partition is obtained by minimizing the energy:

$$F(\partial\Omega) = \sum_{i=1}^N \int_{\Omega_i} -\log p_i(I) dx + \text{length}(\partial\Omega) \quad (11.1)$$

To describe the visual consistency of a region, we need to define a *family of density function*. The choice of this family must be done such that it can represent/approximate the information of each region and it should be able to discriminate two different regions. For smooth non-textured images, a common choice is to use Gaussian distributions. It means that the conditional probability with respect to h_i for a vector a of the vector valued image I is:

$$p_i(a) = p_{\Sigma_i, \mu_i}(a) = \frac{1}{(2\pi)^{\frac{p}{2}} |\Sigma_i|^{1/2}} e^{-\frac{1}{2}(a - \mu_i)^T \Sigma_i^{-1} (a - \mu_i)}$$

where μ_i and Σ_i are respectively the mean vector and the covariance matrix associated to the region i of the vector valued image I (for color images a is a vector with 3 components, so p is equal to 3).

Remark : As we will see on some experiments, the use of Gaussian densities may be also efficient on particular textured images. Actually, these images can be segmented by fitting Gaussian models even if these models do not give a good estimation of the region properties.

Considering this family of distribution, we can rewrite the energy:

$$\begin{aligned} F(\Sigma, \mu, \partial\Omega) &= \frac{1}{2} \sum_{i=1}^N \int_{\Omega_i} \left(p \log(2\pi) + \log |\Sigma_i| \right. \\ &\quad \left. + (I(x) - \mu_i)^T \Sigma_i^{-1} (I(x) - \mu_i) \right) dx + \text{length}(\partial\Omega) \end{aligned} \quad (11.2)$$

Such an energy has already been studied in [306] by looking for the minimum length description and in [613] with a region completion scheme. The first one is a variant of graduated non-convexity and the second one propose to use snakes in a region completion scheme. This last method solve the problem by iterating two steps: 1) estimate statistical parameters with fixed boundaries, 2) move the boundaries with constant statistical parameters.

Here, we propose a global variational framework using the level set theory. The energy minimum is obtained via a gradient descent with respect to the statistical parameters and the boundary position.

11.3 On finding the minima

We study two different approaches driving to a local minima of the functional (11.2) when considering the 2-phase problem for smooth, non-textured images. The energy corresponding to this particular case is:

$$F(\partial\Omega, \Theta) = \int_{\Omega_1} e_1(x) \, dx + \int_{\Omega_2} e_2(x) \, dx + \text{length}(\partial\Omega)$$

where $\Theta = (\Sigma_1, \mu_1, \Sigma_2, \mu_2)$

and $e_i(x) = \log |\Sigma_i| + (I(x) - \mu_i)^T \Sigma_i^{-1} (I(x) - \mu_i)$.

Remark : Active contours without edges for vector-valued images presented in [97] can be seen as a particular case of this formulation where Σ_1 and Σ_2 are set to the identity matrix (i.e., only the means μ_1 and μ_2 were considered).

11.3.1 First approach: extension of the integral terms to all the domain

A difficulty encountered when we want to derive (11.3) is the dependence with respect to the border position of the integration domains Ω_1 and Ω_2 . As it has been done in the past [97], we extend these integrals to all the domain by using the level set function $\phi : \Omega \rightarrow \mathbb{R}$ defined as:

$$\begin{cases} \phi(x) = \mathcal{D}(x, \partial\Omega), & \text{if } x \in \Omega_1 \\ \phi(x) = -\mathcal{D}(x, \partial\Omega), & \text{if } x \in \Omega_2 \end{cases} \quad (11.3)$$

Using the same regularized form $H_\epsilon(z)$ of the heaviside function as in [97], the functional (11.3) can be written as:

$$\begin{aligned} E(\phi, \Theta) = & \int_{\Omega} (e_1(x) H_\epsilon(\phi) + e_2(x)(1 - H_\epsilon(\phi))) \, dx \\ & + \text{length}(\partial\Omega) \end{aligned} \quad (11.4)$$

The length term can also be expressed with respect to ϕ :

$$\text{length}(\partial\Omega) = \int_{\Omega} |\nabla H_\epsilon(\phi(x))| \, dx$$

The Euler-Lagrange equations obtained for the Gaussian parameters μ_i and Σ_i can be directly solved. The solution gives expressions of μ_i and Σ_i with respect

to the level set function ϕ :

$$\begin{cases} \mu_i(\phi) = \frac{\int_{\Omega} (\mu_i - I(x)) \chi_i(\phi(x)) dx}{\int_{\Omega} \chi_i(\phi(x)) dx} \\ \Sigma_i(\phi) = \frac{\int_{\Omega} (\mu_i - I(x)) (\mu_i - I(x))^T \chi_i(\phi(x)) dx}{\int_{\Omega} \chi_i(\phi(x)) dx} \end{cases} \quad (11.5)$$

where $\chi_1(\phi) = H_\epsilon(\phi)$ and $\chi_2(\phi) = 1 - H_\epsilon(\phi)$. These expressions are the estimation of the Gaussian parameters into the respective region which is coherent with the formulation of the problem.

Hence the energy depends only on ϕ :

$$E(\phi, \Theta) = E(\phi, \Theta(\phi)) = G(\phi)$$

Then, it is possible to compute the first variations of G with respect to ϕ . A detailed description of the derivation can be found in [443]. Using the result of [443], we can write the following evolution equation for ϕ :

$$\phi_t(x) = \delta_\epsilon(\phi(x)) \left[\nu \cdot \operatorname{div} \left(\frac{\nabla \phi}{|\nabla \phi|} \right) + e_2(x) - e_1(x) \right] \quad (11.6)$$

with $\delta_\epsilon(\phi) = H'_\epsilon(\phi)$.

The implementation is straightforward: **the level set function is evolved with a gradient descent using the equation (11.6) while the Gaussian parameters are updated at each iteration with respect to (11.5).**

11.3.2 Second approach: direct derivation using shape derivation principle

We can wonder how the use of a regularized function in the energy modifies our objective function. Using the shape derivative tool introduced in [23], it is possible to differentiate directly the functional (11.3).

We suppose that the statistical parameters are obtained using an estimation on the respective region as in (11.5):

$$\begin{cases} \mu_i(\Omega_i) = \frac{\int_{\Omega_i} I(x) dx}{\int_{\Omega_i} dx} \\ \Sigma_i(\Omega_i) = \frac{\int_{\Omega_i} (\mu_i - I(x)) (\mu_i - I(x))^T dx}{\int_{\Omega_i} dx} \end{cases}$$

Then the functional depends only on the border position $\partial\Omega$:

$$E(\partial\Omega) = \int_{\Omega_1} e_1(x, \Omega_1) dx + \int_{\Omega_2} e_2(x, \Omega_2) dx + \nu \int_{\partial\Omega} \partial\dot{\Omega}(x) dx \quad (11.7)$$

The integrals $D(\Omega_i) = \int_{\Omega_i} e_i(x, \Omega_i) dx$ may be differentiated with respect to the position of Ω_i using shape derivative. Concerning our case, the derivative gives a simple term (see [443] for details):

$$\langle D'(\Omega_i), V \rangle = + / - \int_{\partial\Omega} e_i(x, \Omega_i) (V(x) \cdot N(x)) da(x)$$

where $N(x)$ is the unit normal vector to $\partial\Omega$ at point x . The sign depends on the direction of N , i.e. if Ω_i is the inside region, the sign is minus otherwise it is positive. Now we can write the new evolving equation:

$$\partial\Omega(x)_t = (e_2(x) - e_1(x) + \nu\kappa)\mathbf{N}$$

where κ is the curvature of the curve $\partial\Omega$ and \mathbf{N} is the normal vector to $\partial\Omega$.

It is efficient to use the level set technique [401, 398] to implement this equation. The level set function is defined as the distance function like in (11.3) and we get the following level set evolution equation:

$$\phi_t(x) = \left[e_2(x) - e_1(x) + \nu \operatorname{div} \left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right] |\nabla\phi| \quad (11.8)$$

The equation (11.6) can be obtained: 1) by approximating $|\nabla\phi|$ by 1 since ϕ should be the distance function; 2) by introducing $\delta_\alpha(x)$ to impose the Narrow Band hypothesis: only local pixels have influence on the curve propagation.

11.4 Experiments

The complexity of the variational method presented in the last section is high. The unknown parameters for each region can be up to 9 per region for color images (6 for the symmetric 3x3 covariance matrix and 3 more for the mean).

11.4.1 Gray-valued images

For scalar images, the complexity is smaller. Only two parameters (mean and variance) are necessary to represent a region. The simplified evolving equation for this case is:

$$\begin{aligned} \phi_t(x) = & \delta_\epsilon(\phi(x)) \left(\nu \cdot \operatorname{div} \left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right. \\ & \left. + \log \frac{\sigma_2^2}{\sigma_1^2} - \frac{(I(x) - \mu_1)^2}{\sigma_1^2} + \frac{(I(x) - \mu_2)^2}{\sigma_2^2} \right) \end{aligned}$$

We show results on two synthetical images. The first one is composed by two regions with different means but same variances [Fig.11.1] while the second one has two regions having the same mean but variances [Fig.11.2].

11.4.2 Color images

Concerning color images, the complexity is higher but images composed by complex regions can be considered. First, we show results on synthetical data where 3-dimensional Gaussian models were used to generate data for each region. In both examples, the regions have the same mean but their covariance matrix is

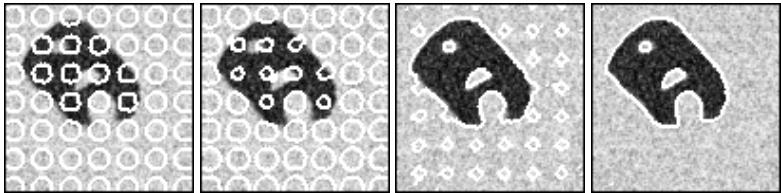


Figure 11.1. Two regions with different means - Contour evolution

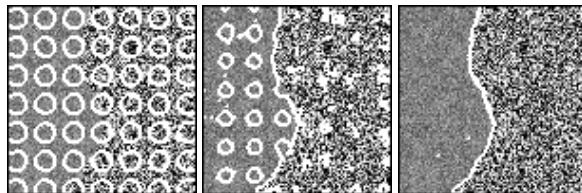
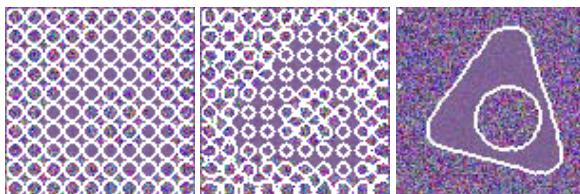
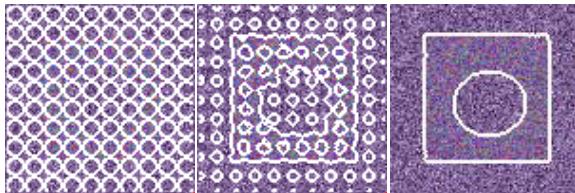


Figure 11.2. Two regions with different variances - Contour evolution



$$\mu_1 = \mu_2$$

$$\Sigma_1 = \begin{pmatrix} \alpha_1 & 0 & 0 \\ 0 & \beta_1 & 0 \\ 0 & 0 & \gamma_1 \end{pmatrix} \text{ and } \Sigma_2 = \begin{pmatrix} \alpha & a_1 & b_1 \\ b_1 & \beta & c_1 \\ c_1 & b_1 & \gamma \end{pmatrix}$$



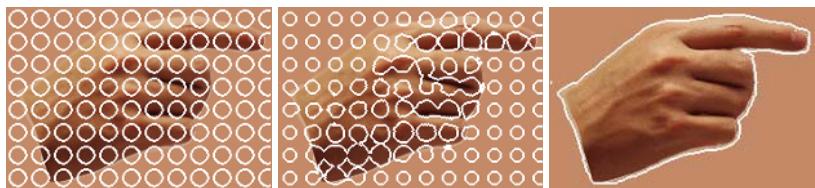
$$\mu_1 = \mu_2$$

$$\Sigma_1 = \begin{pmatrix} \alpha_2 & 0 & 0 \\ 0 & \beta_2 & 0 \\ 0 & 0 & \gamma_2 \end{pmatrix} \text{ and } \Sigma_2 = \begin{pmatrix} \alpha & a_2 & b_2 \\ b_2 & \beta & c_2 \\ c_2 & b_2 & \gamma \end{pmatrix}$$

Figure 11.3. Synthetic test for color images - Contour evolution

different. For the first example, auto-correlation coefficients (between color components) are different while only cross-correlation coefficients are different for the second example. Contrary to more classical approaches as the one presented in [97], our method is able to capture these differences [Fig.11.3].

Experiments on real images gave interesting results. Let us precise that we used the CIE-Lab color space such that distance between 3D points corresponds to perceptual color difference.



(a) Uniform background - Contour evolution



(b) Noisy background - Contour evolution



(c) Textured background - Contour evolution

Figure 11.4. “Hand” Color Image(left: initial contour, center: evolving contour, right: final segmentation)

The first test consists in segmenting a hand photography placed on different backgrounds. Three backgrounds are considered: uniform (hue close to the hand’s) [Fig.11.4(a)], noisy [Fig.11.4(b)] and textured [Fig.11.4(c)]. The algorithm succeeded in giving the expected partitioning for each test. Two more tests on *natural* images are shown. First, the “squirrel” image [Fig.11.5]: the regions are textured but they have a different color. The partitioning result looks very accurate [Fig.11.5]. Second example, the “Rocks on mars” image: the rock is slightly darker than the sand with a different texture. The obtained result is still satisfactory [Fig.11.6].

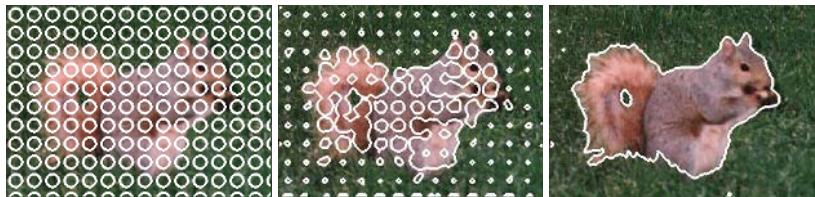


Figure 11.5. “Squirrel” Color Image - Contour evolution



Figure 11.6. “Rocks on mars” Color Image - Contour evolution

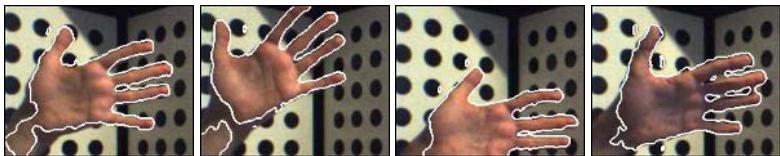


Figure 11.7. Hand sequence (Color Images)

11.4.3 Color image sequences

The dynamical property of our method can be useful to segment a sequence of images. The hand examples presented here is composed by a moving object (a hand) placed in front of a textured background. Since the hand motion is composed by various rotations and translations, the luminosity of the object varies very much during the sequence. The result of the previous frame is used as initialization for the next frame. The method does not integrate additional terms based on motion properties as optical flow. Hence, the method is not limited to small displacements. As you can see on the example, the algorithm can naturally integrate luminosity modifications thanks to the dynamical Gaussian parameters. We can also remark that a Gaussian approximation for the background is not good at all but it is sufficient to differentiate it from the hand [Fig.11.7].

11.4.4 Implementation remarks

For all the tests, except the sequence, the same initialization with tiny circles was used. Such an initialization has been already used in the past. The reasons why we chose this initialization was its ability to detect easily holes and the convergence speed. However, other initializations have been tested such as a single circle. It

still gives good results but the convergence speed decreases significantly. Moreover, we have to set only two parameters: the time step and the regularization weight ν . For all the color examples, we used the same values: $dt = 1$ and $\nu = 1$.

Concerning the speed, we use an explicit discretization in time, then the evolution must be done slowly to be stable. On 100x100 images, only few seconds are required for scalar images while color images need around twenty seconds on a 1 Ghz CPU. It seems possible to get close to real time using a semi-implicit scheme as the AOS [570]. To deal with larger images, we are in the process to develop a multi-scale approach.

11.5 Generalization to N regions

We start from the functional (11.1) and we would like to extend each integral over Ω_i to all the image domain Ω . We must find characteristic functions $\{\chi_i, i = 1, \dots, N\}$ such that:

$$\begin{cases} \phi_i(x) > 0 & \text{if } x \in \Omega_i \\ \phi_i(x) = 0 & \text{if } x \in \partial\Omega_i \\ \phi_i(x) < 0 & \text{otherwise} \end{cases}$$

Actually, different kinds of characteristic functions have been proposed for this problem. One can simply associate one level set per region but pixels can be associated to multiple regions or to no region. An additional coupling term must be added to avoid that. A second way of defining χ_i has been propose in [97] by associating a characteristic function to each combination of level set signs. Hence only $\log(N)$ level sets are needed to segment an image into N regions and each pixel is associated to one and only one region. For example, an image can be segmented in 4 regions using 2 level sets ϕ_1 and ϕ_2 by minimizing the following functional:

$$E(\phi_1, \phi_2, \Theta) = \sum_{i=1}^4 \int_{\Omega} -\log p_{\theta_i}(I(x)) \chi_i(\phi_1, \phi_2) dx + \text{length}(\partial\Omega_1 \cup \partial\Omega_2) \quad (11.9)$$

$$\text{with } \begin{cases} \chi_1(\phi_1, \phi_2) = H_{\epsilon}(\phi_1)H_{\epsilon}(\phi_2), \\ \chi_2(\phi_1, \phi_2) = (1 - H_{\epsilon}(\phi_1))H_{\epsilon}(\phi_2), \\ \chi_3(\phi_1, \phi_2) = H_{\epsilon}(\phi_1)(1 - H_{\epsilon}(\phi_2)), \\ \chi_4(\phi_1, \phi_2) = (1 - H_{\epsilon}(\phi_1))(1 - H_{\epsilon}(\phi_2)) \end{cases}$$

The derivation of (11.9) is similar to one for the 2-phase case and two evolution equations are obtained for ϕ_1 and ϕ_2 . Making the notations $e_i(x) = \log|\Sigma_i| + (I(x) - \mu_i)^T \Sigma_i^{-1} (I(x) - \mu_i)$ and $H_i(x) = H_{\epsilon}(\phi_i(x))$, the energy can be simply

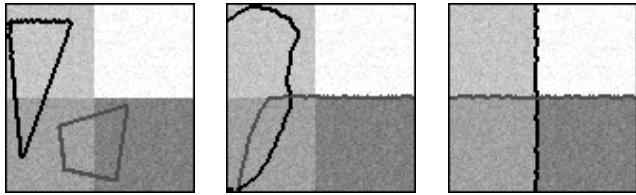


Figure 11.8. Four different means - Contour evolution

expressed as follows:

$$\begin{aligned} E(\phi_1, \phi_2) = & \int_{\Omega} (|H_\epsilon(\phi_1(x))| + |H_\epsilon(\phi_2(x))|) dx \\ & + \int_{\Omega} (e_1 H_1 H_2 + e_2 (1 - H_1) H_2 + e_3 H_1 (1 - H_2) + e_4 (1 - H_1)(1 - H_2)) dx \end{aligned} \quad (11.10)$$

and the corresponding evolution equations for ϕ_1 and ϕ_2 are:

$$\begin{cases} \frac{\partial \phi_1}{\partial t}(x) = \delta_\epsilon(\phi_1(x)) \left(\nu \cdot \operatorname{div} \left(\frac{\nabla \phi_1}{|\nabla \phi_1|} \right) + (e_1 - e_2) H_2 + (e_3 - e_4)(1 - H_2) \right) \\ \frac{\partial \phi_2}{\partial t}(x) = \delta_\epsilon(\phi_2(x)) \left(\nu \cdot \operatorname{div} \left(\frac{\nabla \phi_2}{|\nabla \phi_2|} \right) + (e_1 - e_3) H_1 + (e_2 - e_4)(1 - H_1) \right) \end{cases}$$

A nice example is shown [Fig.11.8] but the method becomes more sensible to the initial contour position on more complicated images. Moreover, the complexity becomes very high when the number of regions increase.

11.6 Conclusion and Future Work

We have presented a dynamical framework to segment scalar or vector-valued images. Two different approaches have been considered to tackle the problem of energy minimization, that drove to the same evolution equations implemented using the powerful Level Set technique [401, 398]. Convincing results were shown on synthetical and real images. The method looks particularly powerful in segmenting various type of images. A simple multi-dimensional Gaussian to represent each region is often sufficient to discriminate complex regions. Regarding the tracking example, improvements can be done by using for instance priors on the shape [141, 444] or by integrating a more complex model for the background and a skin model for the hand. Moreover, we used here a parametric model assuming that probability densities of each region are close to Gaussian distributions. The development of non-parametric methods can be useful when dealing with textures which are characterized by moments of higher order. We are currently concentrating on these last issues to improve and generalize the method.

12

Mumford-Shah for Segmentation and Stereo

**Anthony Yezzi, Stefano Soatto, Hailin Jin, Andy Tsai,
Alan Willsky**

Abstract

We begin by presenting an active contour model which utilizes the Mumford-Shah energy functional for the purpose of piecewise smooth image segmentation. We then show how the use of simultaneous piecewise smooth image segmentation on a set of calibrated 2D images of a common 3D scene may be utilized for reconstructing the unknown shapes and radiances of scene objects. To do so, we must lift the support of the unknown smooth functions in the traditional Mumford-Shah framework, which will now represent the unknown radiance of scene objects in this application, onto a manifold which will represent the unknown shape of scene objects. This constitutes a significant mathematical departure from the traditional Mumford-Shah model since the unknown functions now live directly on the unknown geometric surfaces rather than their surrounding ambient space. The intuition, however, follows from the original model in that the estimates must closely match the observed data while maintaining a high degree of smoothness both in the estimated functions as well as the estimated geometry.

12.1 Introduction

Image segmentation and smoothing constitute two important applications of partial differential equations in image processing and computer vision. For segmentation, the technique of *snakes* or *active contours* has grown significantly since the seminal work of Kass, Witkin, and Terzopoulos [263]. This includes the introduction of geometric models based on curve evolution theory [81, 85, 127, 263, 271, 331, 510, 520, 590] to region-based models [100, 409, 440, 593, 611, 613]. For image smoothing, techniques of anisotropic diffusion

have been developed under a widespread field of research ranging from models of Perona and Malik [423, 424] to curve and surface evolution methods linked to geometric scale spaces [177, 273, 277, 462]. While most of these methods have been formulated primarily for greyscale imagery, a number of techniques have also been formulated for color imagery and other forms of vector-valued data [460, 490, 565, 575, 589, 597].

In general, the goal of most active contour algorithms is to extract the boundaries of homogeneous regions within an image, while the goal of most anisotropic diffusion algorithms is to smooth the values of an image within homogeneous regions but not across the boundaries of such regions. We note that one of the most widely studied mathematical models in image processing and computer vision addresses both goals simultaneously, namely that of Mumford and Shah [387, 388] who presented the variational problem of minimizing a functional involving a piecewise smooth representation of an image.

In the first part of this chapter we show how an active contour model, implemented using level set methods of Osher and Sethian [401], may be folded naturally into the Mumford-Shah framework for simultaneous image segmentation and smoothing. By utilizing an active contour, there is no need to consider the relaxed elliptic approximations put forth by Ambrosio and Tortorelli [13] (see also [436, 477]) for ease of implementation. Rather, the original formulation may be used directly, since the contour itself represents the discontinuity set directly. A more extensive development of this curve evolution approach to Mumford-Shah segmentation can be found in [527] as well as a related contemporaneous approach in [551]. This development may be regarded as an extension of several recent region-based approaches to active contours [100, 409, 456, 594].

We note that, in general, region-based approaches enjoy a number of attractive properties including greater robustness to noise (by avoiding derivatives of the image intensity) and initial contour placement (by being less local than most edge-based approaches). contrast to most other region based techniques however, which typically assume highly constrained parametric models for pixel intensities within each region, our approach employs the statistical model directly implied by the Mumford-Shah functional. That is, the image is modeled as a random field within each region, a model that naturally accommodates variability across each region without the need to model such variability parametrically. In addition, while many region-based methods require a priori knowledge of the number of region types (such as [100] which assumes exactly two region types with two different mean intensities or [593] which requires separate sets of curves to deal with more than two region types), the Mumford-Shah based approach can automatically segment images with multiple region types (e.g. each with different mean intensities) without such a priori knowledge.

In the second part of this chapter, we will lift the Mumford-Shah formulation from a flat image space onto a curved evolving manifold representing the 3D objects in a scene which are captured in a given set of 2D images. This approach naturally arises by considering the use of a collection of 2D active contours to segment a corresponding collection of scene images. However, rather than allowing

the contours to be independent (in which case the Mumford-Shah model presented thusfar may simply be applied separately to each image of the 3D scene), we instead assume that the contours correspond to the occluding boundaries of a common estimated 3D surface in space and therefore depend upon each other according to the geometry of this surface. The contours are then deformed indirectly, simultaneously, and in a tightly coupled manner by evolving the unknown surface itself. The evolution of this surface is, in turn, chosen to yield the desired motion of these occluding contours. Finally, these desired contour motions are determined using a Mumford-Shah type model which measures the quality of the corresponding image segmentations. The smooth estimates of the image data inside and outside of these occluding contours are also derived from a common function living on the surface (and a common “blue sky” background) via a perspective projection and are therefore evolved indirectly as well by evolving this common function (the radiance) itself. The fact that the function we are estimating lives directly on the surface we are estimating, rather than the interior or exterior of its surrounding space, represents a substantial mathematical (and computational) departure from the more traditional model discussed in the first part of this chapter. However, the basic intuition underlying both models is similar; namely, one seeks a smooth function as well as a smooth geometry which generates a faithful approximation of the image data.

Finally, in this work, we adopt the level set techniques of Osher and Sethian [401, 472] in the implementation of all our models. Level sets, in conjunction with upwind, conservative, monotone difference schemes [331, 395, 472], allow for automatic handling of cusps, corners, and topological changes as the curves evolve.

12.2 Mumford-Shah based curve evolution

The point of reference for this paper is the Mumford-Shah functional¹

$$E(\mathbf{f}, \mathbf{C}) = \beta \iint_{\Omega} (\mathbf{f} - I)^2 \, dA + \alpha \iint_{\Omega \setminus \mathbf{C}} |\nabla \mathbf{f}|^2 \, dA + \gamma \oint_{\mathbf{C}} ds \quad (12.1)$$

in which \mathbf{C} denotes the smooth, closed segmenting curve, I denotes the observed data, \mathbf{f} denotes the piecewise smooth approximation to I with discontinuities only along \mathbf{C} , and Ω denotes the image domain [387, 388]. This energy functional is also referred to as the weak membrane by Blake and Zisserman [52]. The parameters α , β , and γ are positive real scalars which control the competition between the various terms above and determine the “scale” of the segmentation and smoothing. Of course one of these parameters can be eliminated by setting

¹The Hausdorff penalty in the original Mumford-Shah functional applies to more general sets of discontinuities than just simple smooth curves. In this more restricted case considered here, however, an arclength penalty is equivalent.

it to 1 but, for clarity of exposition, we will keep it as is. From an estimation-theoretic standpoint, the first term in $E(\mathbf{f}, \mathbf{C})$, the data fidelity term, can be viewed as the measurement model for \mathbf{f} with β inversely proportional to the variance of the observation noise process. The second term in $E(\mathbf{f}, \mathbf{C})$, the smoothness term, can be viewed as the prior model for \mathbf{f} given \mathbf{C} . The third term in $E(\mathbf{f}, \mathbf{C})$ is a prior model for \mathbf{C} which penalizes excessive arc length. With these terms, the Mumford-Shah functional elegantly captures the desired properties of segmentation and reconstruction by piecewise smooth functions. The Mumford-Shah problem is to minimize $E(\mathbf{f}, \mathbf{C})$ over admissible \mathbf{f} and \mathbf{C} . The removal of any of the three terms in equation (12.1) results in trivial solutions for \mathbf{f} and \mathbf{C} , yet with all three terms, it becomes a difficult problem to solve. In this paper, we constrain the set of discontinuities in the Mumford-Shah problem to correspond to evolving sets of curves, enabling us to tackle the problem via a curve-evolution-based approach.

12.2.1 Image estimates and boundary-value stochastic processes

For any arbitrary closed curve \mathbf{C} in the image domain, Ω is partitioned into R and R^c , corresponding to the image domain inside and outside the curve, respectively. Fixing such a curve, minimizing (12.1) corresponds to finding estimates $\hat{\mathbf{f}}$ and $\hat{\mathbf{g}}$ in regions R and R^c respectively, to minimize

$$\begin{aligned} E_C(\mathbf{f}, \mathbf{g}) = & \beta \iint_R (\mathbf{f} - I)^2 dA + \alpha \iint_R |\nabla \mathbf{f}|^2 dA \\ & + \beta \iint_{R^c} (\mathbf{g} - I)^2 dA + \alpha \iint_{R^c} |\nabla \mathbf{g}|^2 dA. \end{aligned} \quad (12.2)$$

The estimates $\hat{\mathbf{f}}$ and $\hat{\mathbf{g}}$ that minimize (12.2) satisfy (decoupled) PDE's which can be obtained using standard variational methods [388]. Alternatively, each of these estimates can be obtained from the theory of optimal estimation. This statistical interpretation is potentially of more than just intellectual interest, as it suggests lines of inquiry beyond the scope of this chapter. Specifically, the estimate $\hat{\mathbf{f}}$ that minimizes (12.2) can be interpreted as the optimal estimate of a boundary-value stochastic process [5] \mathbf{f} on the domain R whose measurement equation is

$$I = \mathbf{f} + \mathbf{v} \quad (12.3)$$

and whose prior probabilistic model is given by

$$\nabla \mathbf{f} = \mathbf{w} \quad (12.4)$$

where \mathbf{v} and \mathbf{w} are independent white Gaussian random fields with covariance intensities $\frac{1}{\beta}$ and $\frac{1}{\alpha}$, respectively.

One effective approach to characterizing \mathbf{f} is through the use of complementary processes [5]. In particular, we seek a process \mathbf{z} which *complements* the observation I in (12.3) in that \mathbf{z} and I are uncorrelated and, together, they are

informationally equivalent to $\zeta = \{\mathbf{v}, \mathbf{w}\}$ (i.e. to all of the underlying random processes defining the estimation problem). Moreover, since the specification of the statistics of I in (12.3) and (12.4) is via a differential model and involving an internal “state” (namely \mathbf{f}), we seek an analogous model for \mathbf{z} . We refer the reader to [5] for the complete methodology for the direct construction of such complementary models, employing Green’s identity and formal adjoints of differential operators. The application of this methodology to (12.3) and (12.4) yields the following model for the \mathbf{z} :

$$\mathbf{z} = \boldsymbol{\lambda} - \alpha \mathbf{w} \quad (12.5)$$

where the internal state $\boldsymbol{\lambda}$ satisfies

$$-\nabla \boldsymbol{\lambda} = \beta \mathbf{v} \quad (12.6)$$

with boundary condition

$$\mathcal{N}^T \boldsymbol{\lambda} = 0 \quad \text{on } \mathbf{C} \quad (12.7)$$

where \mathcal{N} denotes the outer normal of the curve \mathbf{C} .

Eliminating \mathbf{v} and \mathbf{w} from (12.3)–(12.6) we can express \mathbf{f} and $\boldsymbol{\lambda}$ completely in terms of I and \mathbf{z} . Then, since I and \mathbf{z} are uncorrelated, we obtain an internal realization of the optimal estimate $\hat{\mathbf{f}}$:

$$\begin{bmatrix} \nabla & -\frac{1}{\alpha} \mathbf{I} \\ \beta \mathbf{I} & -\nabla \end{bmatrix} \begin{bmatrix} \hat{\mathbf{f}} \\ \hat{\boldsymbol{\lambda}} \end{bmatrix} = \begin{bmatrix} 0 \\ \beta I \end{bmatrix} \quad \text{on } R$$

with the boundary condition

$$\mathcal{N}^T \hat{\boldsymbol{\lambda}} = 0 \quad \text{on } \mathbf{C}.$$

Eliminating $\hat{\boldsymbol{\lambda}}$ and noticing that the product $\mathcal{N} \cdot \nabla \hat{\mathbf{f}}$ is the derivative of $\hat{\mathbf{f}}$ in the direction of \mathcal{N} , we obtain the following damped Poisson equation with Neumann boundary condition for $\hat{\mathbf{f}}$

$$\hat{\mathbf{f}} - \frac{\alpha}{\beta} \nabla^2 \hat{\mathbf{f}} = I \quad \text{on } R \quad (12.8a)$$

$$\frac{\partial \hat{\mathbf{f}}}{\partial \mathcal{N}} = 0 \quad \text{on } \mathbf{C}. \quad (12.8b)$$

In a similar fashion, $\hat{\mathbf{g}}$ is given as the solution to following:

$$\hat{\mathbf{g}} - \frac{\alpha}{\beta} \nabla^2 \hat{\mathbf{g}} = I \quad \text{on } R^c \quad (12.9a)$$

$$\frac{\partial \hat{\mathbf{g}}}{\partial \mathcal{N}} = 0 \quad \text{on } \mathbf{C}. \quad (12.9b)$$

We will refer to the two sets of equations (12.8) and (12.9) as the estimation PDE’s. The conjugate gradient (CG) method is employed as a fast and efficient solver for these estimation PDE’s.

12.2.2 Gradient flows to minimize the Mumford-Shah functional

With the ability to calculate $\hat{\mathbf{f}}$ and $\hat{\mathbf{g}}$ for any given \mathbf{C} , we now wish to derive a curve evolution for \mathbf{C} that minimizes (12.1). That is, as a function of \mathbf{C} , we wish to find \mathbf{C}_t that minimizes

$$\begin{aligned} E_{\hat{\mathbf{f}}, \hat{\mathbf{g}}}(\mathbf{C}) = & \beta \iint_R (\hat{\mathbf{f}} - I)^2 dA + \alpha \iint_R |\nabla \hat{\mathbf{f}}|^2 dA \\ & + \beta \iint_{R^c} (\hat{\mathbf{g}} - I)^2 dA + \alpha \iint_{R^c} |\nabla \hat{\mathbf{g}}|^2 dA + \gamma \oint_{\mathbf{C}} ds. \end{aligned} \quad (12.10)$$

The first four terms in (12.10) are of the form:

$$J = \iint_D \mathcal{H} dA \quad (12.11)$$

where D denotes either the interior or the exterior of \mathbf{C} , and $\mathcal{H} : \mathbf{R}^2 \rightarrow \mathbf{R}$ is a continuous function. The gradient flow to minimize (12.11) is given by (see [594] for a derivation)

$$\mathbf{C}_t = -\mathcal{H}\mathcal{N}. \quad (12.12)$$

In addition, the gradient flow that minimizes the arc length of \mathbf{C} is given by

$$\mathbf{C}_t = -\kappa\mathcal{N} \quad (12.13)$$

where κ denotes the signed curvature of \mathbf{C} . Knowing gradient flows (12.12) and (12.13), the curve evolution that minimizes (12.10) is given by

$$\mathbf{C}_t = \frac{\alpha}{2} \left(|\nabla \hat{\mathbf{g}}|^2 - |\nabla \hat{\mathbf{f}}|^2 \right) \mathcal{N} + \frac{\beta}{2} \left((I - \hat{\mathbf{g}})^2 - (I - \hat{\mathbf{f}})^2 \right) \mathcal{N} - \gamma\kappa\mathcal{N}. \quad (12.14)$$

For the rest of the paper, we will refer to this gradient flow as the *Mumford-Shah flow*. This flow is implemented via the level set method [401, 472] which offers a natural and numerically reliable implementation of these solutions within a context that handles topological changes in the interface without any additional effort. Furthermore this flow together with the optimal estimation PDE's makes explicit the coupling between the optimal estimates and the curve evolution.

12.2.3 Remarks on the Mumford-Shah active contour model

One very attractive feature associated with our Mumford-Shah active contour model (and also present in other region-based methods) is that it automatically proceeds in the correct direction without relying upon additional inflationary terms commonly employed by many active contour algorithms. We illustrate this in Figures 12.1–12.4 with a noisy synthetic image of a hand. An initial contour completely contained within the hand will flow outward toward the boundary (Figure 12.1); an initial contour partially inside and partially outside the hand will

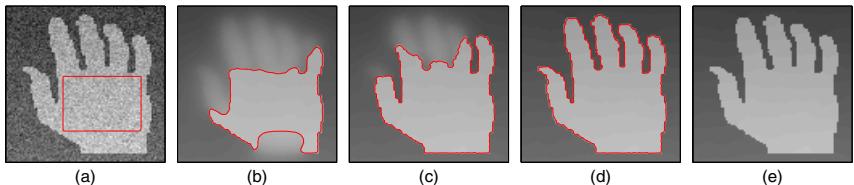


Figure 12.1. Outward flow from inside.

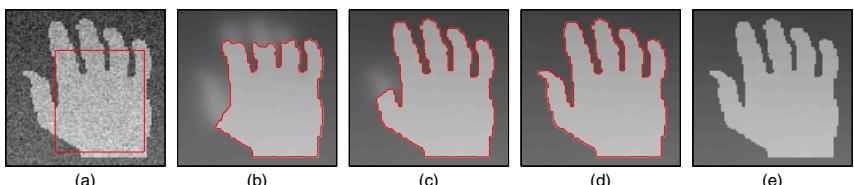


Figure 12.2. Bi-directional flow.

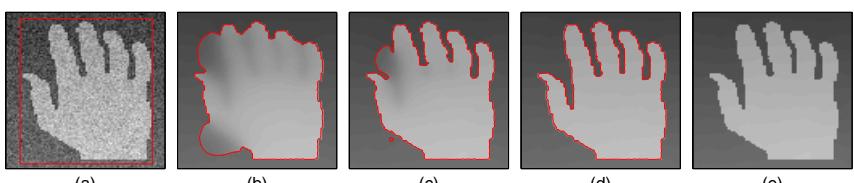


Figure 12.3. Inward flow from outside.



Figure 12.4. Outward flow from outside.

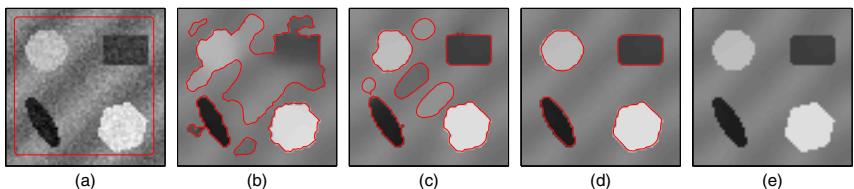


Figure 12.5. Segmentation and smoothing of an image with 4 distinct foreground regions.

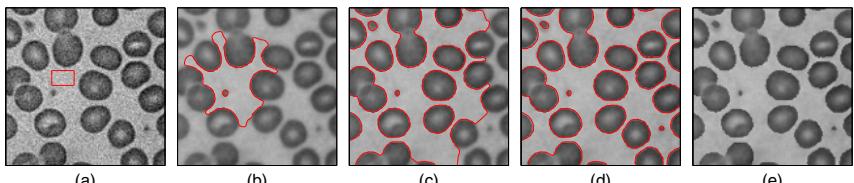


Figure 12.6. Segmentation and smoothing of an image containing multiple red blood cells.

flow in both directions toward the boundary (Figure 12.2); an initial contour encircling the hand will flow inward toward the boundary (Figure 12.3); and finally, an initial contour situated outside the hand will flow outward and wrap around the boundary (Figure 12.4). In these figures, Frame (a) shows the initializing contour with the original image; Frames (b) and (c) show the estimate of curve C and the estimates of f and g associated with two intermediate steps of the algorithm; Frame (d) shows the final segmenting curve C and the final reconstruction of the image (based on the estimates \hat{f} and \hat{g}); and finally, Frame (e) shows the reconstruction of the image without the overlaying curve for comparison to the original noisy image. Note that the smooth estimate of the image is continuously estimated based on the current position of the curve. In Figure 12.4, in addition to the curves that outline the boundary of the hand, there exist extraneous curves around the four corners of the image which do not correspond to image edges. This is due to the fact that the algorithm has descended upon and settled on to a local minimum—a common problem faced by all algorithms which rely on gradient descent methods for minimization. However, notice that the piecewise smooth reconstruction of the image shown in Figure 12.4(e) does not exhibit any ill effects from these extraneous curves; that is, the reconstruction does not show any semblance of an edge along these extraneous curves. Thus even if the curve is trapped at a local minimum, the reconstruction of the image is still accurate.

The class of imagery that our algorithm can handle is not restricted just to images with only two distinct means but is equally applicable to images with multiple non-overlapping regions each with different means. Moreover, we do not need to know in advance the number of such regions or distinct means are present. As shown in Figure 12.5, segmentation and smoothing are performed on a noisy synthetic image with four foreground regions of different means situated on a spatially varying background region. Multiple regions are captured by a single contour demonstrating the topological transitions allowed by the model's level set

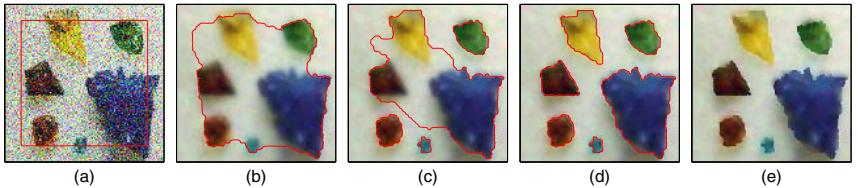


Figure 12.7. Segmentation and smoothing of a color image with 6 distinct foreground regions.

implementation. Figure 12.6 demonstrates this same effect on a noisy real image of multiple red blood cells.

The model can also be generalized, in a very straight forward manner, to handle vector-valued images (e.g. color images or images obtained from scale and orientation decompositions commonly used for texture analysis). Consider the following vector version of the Mumford-Shah functional:

$$E(\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_k, \mathbf{C}) = \beta \iint_{\Omega} \sum_{i=1}^k (\mathbf{f}_i - I_i)^2 \, dA + \alpha \iint_{\Omega \setminus \mathbf{C}} \sum_{i=1}^k |\nabla \mathbf{f}_i|^2 \, dA + \gamma \oint_{\mathbf{C}} ds$$

where I_i and \mathbf{f}_i denote the i th component of the k -dimensional vector-valued observed data and its smooth estimate, respectively. The curve evolution that minimizes this energy functional is given by

$$\begin{aligned} \mathbf{C}_t = & \frac{\alpha}{2} \sum_{i=1}^k \left(|\nabla \hat{\mathbf{g}}_i|^2 - |\nabla \hat{\mathbf{f}}_i|^2 \right) \mathcal{N} + \\ & \frac{\beta}{2} \sum_{i=1}^k \left((I_i - \hat{\mathbf{g}}_i)^2 - (I_i - \hat{\mathbf{f}}_i)^2 \right) \mathcal{N} - \gamma \kappa \mathcal{N} \end{aligned} \quad (12.15)$$

The $\hat{\mathbf{f}}_i$ and $\hat{\mathbf{g}}_i$ for $i = 1, \dots, k$ in (12.15) is given by the solutions to the following:

$$\begin{aligned} \hat{\mathbf{f}}_i - \frac{\alpha}{\beta} \nabla^2 \hat{\mathbf{f}}_i &= I_i \quad \text{on } R \\ \frac{\partial \hat{\mathbf{f}}_i}{\partial \mathcal{N}} &= 0 \quad \text{on } \mathbf{C} \end{aligned}$$

and

$$\begin{aligned} \hat{\mathbf{g}}_i - \frac{\alpha}{\beta} \nabla^2 \hat{\mathbf{g}}_i &= I_i \quad \text{on } R^c \\ \frac{\partial \hat{\mathbf{g}}_i}{\partial \mathcal{N}} &= 0 \quad \text{on } \mathbf{C}. \end{aligned}$$

For demonstration, in Figure 12.7, we show the segmentation and smoothing of a noisy color image of 6 different types of gemstones.

12.3 Mumford-Shah on a Moving Manifold: Stereoscopic Segmentation

In the remainder of this chapter we consider the use of multiple active contours to simultaneously segment multiple 2D images of a common 3D scene. However, rather than representing and evolving the contours separately, we will assume that the evolving contours in different images correspond to the occluding boundaries of an object in space. We will represent this object by a surface in 3D space (on a 3D Cartesian grid using the level set framework of Osher and Sethian [401]) and will therefore evolve the contours indirectly in a coordinated, simultaneous manner by evolving the surface itself. The evolution of the surface will be chosen to yield the desired evolution of these occluding boundary contours. In other words, the joint evolution of the contours corresponds to the evolution of the unknown shape of objects in space.

To begin with, we assume that a scene is composed of a number of smooth Lambertian surfaces supporting smooth Lambertian radiance functions (or dense textures with spatially homogeneous statistics). Under such assumptions, most of the significant irradiance discontinuities (or texture discontinuities) within any image of the scene correspond to occlusions between objects (or the background). These assumptions make the segmentation problem well-posed, although not general. In fact, “true” segmentation in this context corresponds directly to the shape and pose of the objects in the scene². Therefore, we set up a cost functional to minimize variations within each image region, where the free parameters are not the boundaries in the image themselves, but the shape of a surface in space whose occluding contours happen to project onto such boundaries.

12.3.1 Notation

In what follows $\mathbf{x} = (x, y, z)$ will represent a generic point of a scene in \mathbb{R}^3 expressed in global coordinates (based upon a fixed inertial reference frame) while $\mathbf{x}_i = (x_i, y_i, z_i)$ will represent the same point expressed in “camera coordinates” relative to an image I_i (from a sequence of images I_1, \dots, I_n of the scene). To be more precise, we assume that the domain Ω_i of the image I_i belongs to a 2D plane given by $z_i = 1$ and that (x_i, y_i) constitute Cartesian coordinates within this image plane (the important point here is that the coordinate z_i corresponds to a direction which is perpendicular to the image plane). We let $\pi_i : \mathbb{R}^3 \rightarrow \Omega_i; \mathbf{x} \mapsto \hat{\mathbf{x}}_i = (\hat{x}_i, \hat{y}_i)$ denote an ideal perspective projection onto this image plane, where $\hat{x}_i = x_i/z_i$ and $\hat{y}_i = y_i/z_i$. The primary objects of interest will be a regular surface S in \mathbb{R}^3 (with area element dA) supporting a radiance function $\mathbf{f} : S \rightarrow \mathbb{R}$, and a background B which we treat as a sphere of infinite radius (“blue sky”) with angular coordinates $\Theta = (\theta, \phi)$ that may be related in a one-to-one manner with

²We consider the background to be yet another object that happens to occupy the entire field of view (the “blue sky” assumption).

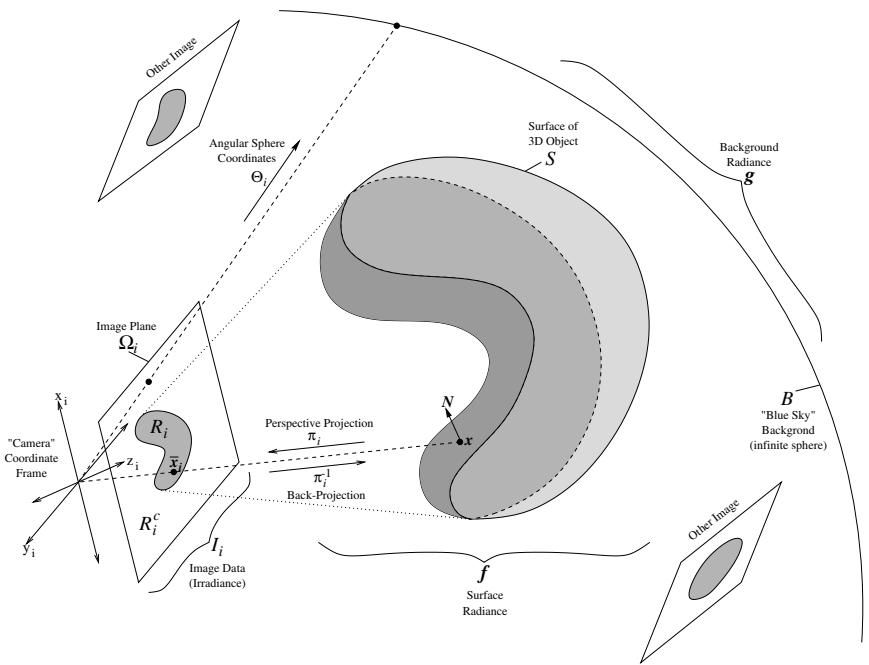


Figure 12.8. Illustration of notation used to describe the unknown surface and background (and their respective radescences), the camera images, and the relevant mappings between these objects.

the coordinates $\hat{\mathbf{x}}_i$ of each image domain Ω_i through the mapping Θ_i (i.e. $\Theta = \Theta_i(\hat{\mathbf{x}}_i)$). We assume that the background supports a different radiance function $\mathbf{g} : B \rightarrow \mathbb{R}$. Given the surface S , we may partition the domain Ω_i of each image I_i into a “foreground” region $R_i = \pi_i(S) \subseteq \Omega_i$, which back-projects onto the surface S , and its complement R_i^c (the “background” region), which back-projects onto the background. Although the perspective projection π_i is not one-to-one (and therefore not invertible), the operation of back-projecting a point from R_i onto the surface S (by tracing back along the ray defined by $\pi_i(\text{ray}) = \hat{\mathbf{x}}_i$ until the first point on S is encountered) is indeed one-to-one with π_i as its inverse. Therefore, we will make a slight abuse of notation and denote back-projecting onto the surface S by $\pi_i^{-1} : R_i \rightarrow S$. Finally, in our computations, we will make use of the relationship between the area measure dA of the surface S and the measure $d\Omega_i = d\hat{x}_i d\hat{y}_i$ of each image domain. This arises from the form of the corresponding projection π_i and is given by $z_i^3 d\Omega_i = -(\mathbf{x}_i \cdot N_i) dA$, where N_i denotes the outward unit normal N of S expressed in the same coordinate system as \mathbf{x}_i . This notation is illustrated in Figure 12.8.

12.3.2 Cost functional

In order to infer the shape of a surface S , we impose a cost on the discrepancy between the projection of a model surface and the actual measurements. Such a cost, $E(\mathbf{f}, \mathbf{g}, S)$, depends upon the surface S as well as upon the radiance of the surface \mathbf{f} and of the background \mathbf{g} . We will then adjust the model surface and radiance to match the measured images and impose a smoothness prior on radiance and a geometric prior on shape. We therefore consider the composite cost functional³

$$E(\mathbf{f}, \mathbf{g}, S) = E_{data}(\mathbf{f}, \mathbf{g}, S) + E_{smooth}(\mathbf{f}, \mathbf{g}, S) + E_{geom}(S) \quad (12.16)$$

We conjecture that, like in the case of the Mumford-Shah functional [388], these ingredients are sufficient to define a unique solution to the minimization problem.

In particular, the geometric and smoothness terms are given by

$$E_{geom} = \int_S dA \quad (12.17)$$

$$E_{smooth} = \int_S \|\nabla_S \mathbf{f}\|^2 dA + \int_B \|\nabla \mathbf{g}\|^2 d\Theta \quad (12.18)$$

which favor surfaces S of least surface area and radiance functions \mathbf{f} and \mathbf{g} of least quadratic variation. (∇_S denotes the intrinsic gradient on the surface S). Finally, the data fidelity term may be measured in the sense of \mathcal{L}^2 by

$$E_{data} = \sum_{i=1}^n \int_{R_i} (\mathbf{f}(\pi_i^{-1}(\hat{\mathbf{x}}_i)) - I_i(\hat{\mathbf{x}}_i))^2 d\Omega_i + \int_{R_i^c} (\mathbf{g}(\Theta_i(\hat{\mathbf{x}}_i)) - I_i(\hat{\mathbf{x}}_i))^2 d\Omega_i. \quad (12.19)$$

In order to facilitate the computation of the first variation with respect to S , we would rather express these integrals over the surface S as opposed to the partitions R_i and R_i^c . We start with the integrals over R_i and note that they are equivalent to

$$-\int_{\pi_i^{-1}(R_i)} (\mathbf{f}(\mathbf{x}) - I_i(\pi_i(\mathbf{x})))^2 \frac{\mathbf{x}_i \cdot N_i}{z_i^3} dA = \int_{\pi_i^{-1}(R_i)} \epsilon_i^2(\mathbf{x}) \sigma_i(\mathbf{x}, N) dA$$

where $\epsilon_i(\mathbf{x}) = \mathbf{f}(\mathbf{x}) - I_i(\pi_i(\mathbf{x}))$ and $\sigma_i(\mathbf{x}, N) = -(\mathbf{x}_i \cdot N_i)/z_i^3$. Now we move to the integrals over R_i^c and note that they are equivalent to

$$\int_{\Omega_i} \epsilon_i^2(\hat{\mathbf{x}}_i) d\Omega_i - \int_{\pi_i^{-1}(R_i)} \epsilon_i^2(\pi_i(\mathbf{x})) \sigma_i(\mathbf{x}, N) dA$$

³Again we could consider each term weighted by a coefficient; however, for simplicity of notation and expositions we will just set these weights to one in this chapter.

where $\varepsilon_i(\hat{\mathbf{x}}_i) = \mathbf{g}(\Theta_i(\hat{\mathbf{x}}_i) - I_i(\hat{\mathbf{x}}_i))$. Combining these “restructured” integrals yields:

$$\int_{\Omega_i} \varepsilon_i^2(\hat{\mathbf{x}}_i) d\Omega_i + \int_{\pi_i^{-1}(R_i)} (\epsilon_i^2(\mathbf{x}) - \varepsilon_i^2(\pi_i(\mathbf{x}))) \sigma_i(\mathbf{x}, N) dA$$

Note that the first integral in the above expression is independent of the surface S (and its radiance function \mathbf{f}) and that the second integral is taken over only a subset of S given by $\pi_i^{-1}(R_i)$. We may express this as an integral over all of S (and thereby avoid the use of π_i^{-1} in our expression) by introducing a characteristic function $\chi_i(\mathbf{x}) \in \{0, 1\}$ into the integrand where $\chi_i(\mathbf{x}) = 1$ for $\mathbf{x} \in \pi_i^{-1}(R_i)$ (represented by the shaded part of the surface S in Figure 12.8) and $\chi_i(\mathbf{x}) = 0$ for $\mathbf{x} \notin \pi_i^{-1}(R_i)$ (i.e. for points that are occluded by other points on S). We therefore obtain the following equivalent expression for E_{data} given in (12.19):

$$E_{data} = \sum_{i=1}^n \left(\int_{\Omega_i} \varepsilon_i^2(\hat{\mathbf{x}}_i) d\Omega_i + \int_S \chi_i(\mathbf{x}) (\epsilon_i^2(\mathbf{x}) - \varepsilon_i^2(\pi_i(\mathbf{x}))) \sigma_i(\mathbf{x}, N) dA \right). \quad (12.20)$$

Finally, we express the entire cost functional by adding the geometric and smoothness priors:

$$\begin{aligned} E(\mathbf{f}, \mathbf{g}, S) &= \sum_{i=1}^n \left(\int_{\Omega_i} \varepsilon_i^2(\hat{\mathbf{x}}_i) d\Omega_i + \int_S \chi_i(\mathbf{x}) (\epsilon_i^2(\mathbf{x}) - \varepsilon_i^2(\pi_i(\mathbf{x}))) \sigma_i(\mathbf{x}, N) dA \right) \\ &\quad + \int_S \|\nabla_S \mathbf{f}\|^2 dA + \int_B \|\nabla \mathbf{g}\|^2 d\Theta + \int_S dA. \end{aligned}$$

12.3.3 Evolution equation

The variation of E_{geom} , which is just the surface area of S , is given by

$$-\frac{d}{dS} E_{geom} = -HN,$$

where H denotes mean curvature and N the outward unit normal. The variation of E_{smooth} is given by

$$-\frac{d}{dS} E_{smooth} = (\Pi(\nabla_S \mathbf{f} \times N) - \|\nabla_S \mathbf{f}\|^2 H) N,$$

where ∇_S denotes the intrinsic gradient of \mathbf{f} on the surface S and where $\Pi(\nabla_S \mathbf{f} \times N)$ denotes the second fundamental form of S applied to the tangent vector $\nabla_S \mathbf{f} \times N$.

The variation of E_{data} requires some attention. In fact, the data fidelity term in (12.20) involves an explicit model of occlusions⁴ via a characteristic function.

⁴The geometric term and the smoothness term are independent of occlusions.

Discontinuities in the kernel cause major problems, for they can result in variations that are zero almost everywhere (e.g. for the case of constant radiance). One easy solution is to mollify the corresponding gradient flow. This can be done in a mathematically sound way by interpolating a smooth force field on the surface in space. Alternatively, the characteristic functions χ_i in the data fidelity term can be mollified, thereby making the integrands differentiable everywhere.

In order to arrive at an evolution equation, we note that the components of the data fidelity term, as expressed in equation (12.20), that depend upon S have the following form

$$E_i(S) = \int_S G_i(\mathbf{x}) \cdot N_i \, dA. \quad (12.21)$$

The gradient flows corresponding to such energy terms have the form

$$-\frac{d}{dS} E_i = -(\nabla_i \cdot G_i) N, \quad (12.22)$$

where ∇_i denotes the gradient with respect to \mathbf{x}_i (recall that \mathbf{x}_i is the representation of a point using the camera coordinates associated with image I_i as described in Section 12.3.1). In particular,

$$G_i(\mathbf{x}) = -\chi_i(\mathbf{x}) (\epsilon_i^2(\mathbf{x}) - \varepsilon_i^2(\pi_i(\mathbf{x}))) \frac{\mathbf{x}_i}{z_i^3} \quad (12.23)$$

and the divergence of G_i , after simplification, is given by

$$-\nabla_i \cdot G_i = \frac{1}{z_i^3} \left((\mathbf{f} - \mathbf{g}) \left[(I_i - \mathbf{f}) + (I_i - \mathbf{g}) \right] (\nabla_i \chi_i \cdot \mathbf{x}_i) + 2\chi_i(I_i - \mathbf{f})(\nabla_i \mathbf{f} \cdot \mathbf{x}_i) \right), \quad (12.24)$$

where we have omitted the arguments of \mathbf{f} , \mathbf{g} , and I_i for the sake of simplicity. A particularly nice feature of this final expression (which is shared by the standard Mumford-Shah formulation for direct image segmentation) is that it depends only upon the image values, *not upon the image gradient*, which makes it less sensitive to image noise when compared to other variational approaches to stereo (and therefore less likely to cause the resulting flow to become “trapped” in local minima). Notice that the first term in this flow involves the gradient of the characteristic function χ_i and is therefore non-zero only on the portions of S which project (π_i) onto the *boundary* of the region R_i (illustrated by the dashed curve on the surface S in Figure 12.8). As such, this term may be directly associated with a curve evolution equation for the boundary of the region R_i within the domain Ω_i of the image I_i . The second term, on the other-hand, may be non-zero over the entire patch $\pi_i^{-1}(R_i)$ of S (illustrated by the shaded part of the surface S in Figure 12.8).

We may now write down the complete gradient flow for $E = E_{data} + E_{smooth} + E_{geom}$ as

$$\begin{aligned} \frac{dS}{dt} &= \sum_{i=1}^n \frac{1}{z_i^3} \left((\mathbf{f} - \mathbf{g}) \left[(I_i - \mathbf{f}) + (I_i - \mathbf{g}) \right] (\nabla_i \chi_i \cdot \mathbf{x}_i) + 2\chi_i (I_i - \mathbf{f}) (\nabla_i \mathbf{f} \cdot \mathbf{x}_i) \right) N \\ &\quad + (\Pi(\nabla_S \mathbf{f} \times N) - \|\nabla_S \mathbf{f}\|^2 H) N - H N. \end{aligned} \quad (12.25)$$

12.3.4 Estimating scene radiance

Once an estimate of the surface S is available, the radiance estimates \mathbf{f} and \mathbf{g} must be updated. For a given surface S we may regard our energy functional $E(S, \mathbf{f}, \mathbf{g})$ as a function only of \mathbf{f} and \mathbf{g} and minimize it accordingly. A necessary condition is that \mathbf{f} and \mathbf{g} satisfy the Euler-Lagrange equations for E based upon the current surface S . These optimal estimate equations are given by the following elliptic PDE's on the surface S and the background B ,

$$\Delta_S \mathbf{f} = \sum_{i=1}^n \chi_i (\mathbf{f} - I_i) \sigma_i \quad \text{and} \quad \Delta_\Theta \mathbf{g} = \sum_{i=1}^n \hat{\chi}_i (\mathbf{g} - I_i) \quad (12.26)$$

where Δ_S denotes the Laplace-Beltrami operator on the surface S , where Δ_Θ denotes the Laplacian on the background B with respect to its spherical coordinates Θ , and where $\hat{\chi}_i(\Theta)$ denotes a characteristic function for the background B where $\hat{\chi}_i(\Theta)=1$ if $\Theta_i^{-1}(\Theta) \in R_i^c$ and $\hat{\chi}_i(\Theta)=0$ otherwise.

12.4 Implementation

The level set method was first introduced by Osher and Sethian in [401] as a numerical device to evolve interfaces. In this formulation, hyper-surfaces are represented implicitly as isophotes of a Lipschitz continuous function sampled over a uniform Cartesian grid. With the robust numerical methods for Hamilton-Jacobi equations, see e.g. [140, 402, 491], the level set method can handle topological changes of the surfaces automatically. Geometrical measurements of the surfaces, such as their normals and curvatures, can be computed easily via finite differencing.

Both the curve evolution approach to the original Mumford-Shah model, presented in the first part of this chapter, as well as the related model for piecewise smooth stereoscopic segmentation, presented afterward, were implemented using level set methods in this work. Because the stereoscopic model involves an evolving function on an evolving manifold, something which is not standard in the level set literature to date, we will present expressions for some key terms (e.g. second fundamental form) needed in corresponding level set evolutions for this model. The detailed development of these expressions, the algorithms used to treat them, and the stability considerations connected with them are the subject of a paper in progress. Here, we will merely present the final expressions.

12.4.1 Level set implementation

In this section we outline the level set implementation of flow (12.25). To simplify the notation, we may first re-write (12.25) as: $S_t = \mathcal{F}N$, where \mathcal{F} denotes a general speed function. In our case,

$$\begin{aligned}\mathcal{F} = & \sum_{i=1}^n \frac{1}{z_i^3} ((\mathbf{f} - \mathbf{g})[(I_i - \mathbf{f}) + (I_i - \mathbf{g})](\nabla_i \chi_i \cdot \mathbf{X}_i) + 2\chi_i(I_i - \mathbf{f})(\nabla_i \mathbf{f} \cdot \mathbf{X}_i)) \\ & + (\Pi(\nabla_S \mathbf{f} \times N) - \|\nabla_S \mathbf{f}\|^2 H) + H.\end{aligned}\quad (12.27)$$

The level set implementation of any geometric flow begins by embedding the initial interface $S(\mathbf{X}, 0)$ as the zero level set of a scalar function $\psi_0(\mathbf{X})$. $\psi_0(\mathbf{X})$ has the property that it is positive outside S , negative inside S and zero on S . $\psi_0(\mathbf{X})$ is then taken to be the initial condition for a function over time $\psi(\mathbf{X}, t)$

$$\psi_0 : \mathbb{R}^3 \rightarrow \mathbb{R}, \quad \psi : \mathbb{R}^3 \times \mathbb{R}^+ \rightarrow \mathbb{R}, \quad \psi(\mathbf{X}, 0) = \psi_0(\mathbf{X}). \quad (12.28)$$

The key point is that we continue to embed the interface as the zero level set of ψ for all time. Thus, we have:

$$\psi_0(S(\mathbf{X}, 0)) = 0, \quad (12.29)$$

$$\psi(S(\mathbf{X}, t), t) = 0. \quad (12.30)$$

Differentiating (12.30) with respect to t therefore yields $\psi_t + \nabla \psi^T S_t = 0$, where $\nabla \psi$ is the gradient of ψ with respect to the spatial coordinates: $\nabla \psi = [\psi_x, \psi_y, \psi_z]^T$. Finally noting that the inward normal of S can be computed based upon ψ as $N = -\nabla \psi / \|\nabla \psi\|$, we have

$$\psi_t = \mathcal{F} \|\nabla \psi\|. \quad (12.31)$$

12.4.2 Second fundamental form

To compute the speed function \mathcal{F} , we need to compute the second fundamental form of the surface in the level set framework. The second fundamental form, which operates only on tangent vectors of the level surface, may be computed using the following operator (which acts only on the tangential portion of any vector)

$$\Pi = \frac{1}{\|\nabla \psi\|} P_N^\perp \nabla^2 \psi P_N^\perp, \quad (12.32)$$

where P_N^\perp is the projection onto the orthogonal complement of the direction $N \in \mathbb{R}^3$ and where $\nabla^2 \psi$ denotes the Hessian matrix of ψ with respect to the spatial coordinates of \mathbb{R}^3 . The mean curvature may be computed as:

$$H = \frac{1}{2} \text{trace}(\Pi) = \frac{1}{2} \nabla \cdot \left(\frac{\nabla \psi}{\|\nabla \psi\|} \right). \quad (12.33)$$

Note that the projection operator P_N^\perp may be ignored in our application since it is applied to the vector $\nabla_S \mathbf{f} \times N$ which has no normal component.

12.4.3 Radiance estimation

Bertalmio et. al. [38] proposed a framework for solving variational problems and PDEs on manifolds, which are defined in level set functions. The key idea is to implicitly represent the static surface using the level set function, and solve the surface equations in a fixed Cartesian coordinate system using this new embedding function.

Their method goes in two steps: first extend the function, which is originally defined on the manifold into the whole space⁵; second solve the original PDE in space, instead on the manifold. One possibility for doing the first step is to extend the radiance function constant along the normals to the manifold. It is equivalent to solving the following PDE:

$$\nabla \mathbf{f} \cdot \mathbf{N} = 0 \iff \nabla \mathbf{f} \cdot \nabla \psi = 0. \quad (12.34)$$

Recall that \mathbf{f} is a scalar function defined on S . After adding a time variable, we can numerically seek the solution to equation (12.34) by iteratively searching for the steady state of the following PDE:

$$\frac{\partial \mathbf{f}}{\partial t} + \text{sign}(\psi)(\nabla \mathbf{f} \cdot \nabla \psi) = 0. \quad (12.35)$$

For the second step, Bertalmio et. al. derived expressions for isotropic and anisotropic diffusions on manifolds. In the case of isotropic diffusion $\frac{\partial \mathbf{f}}{\partial t} = \Delta_S \mathbf{f}$, which is related to our problem, the Laplace-Beltrami operator $\Delta_S \mathbf{f}$ can be computed using finite difference as follows:

$$\Delta_S \mathbf{f} = \frac{1}{\|\nabla \psi\|} \nabla \cdot (P_{\nabla \psi}^\perp \nabla \mathbf{f} \|\nabla \psi\|). \quad (12.36)$$

12.5 Experiments

Here we show the results of this piecewise smooth stereoscopic segmentation algorithm on two illustrative synthetic scenes as well as one real scene calibrated using standard techniques. Notice that the images of this final scene include fine textures, specular highlights and even substantial calibration errors.

In Figure 12.9 we see in the leftmost column (of the bottom two rows) two different images (from sixteen total) of a synthetic pair of spheres illuminated from two point light sources at opposite ends of their common axis. Clearly, from these images, we can see that the spheres are shaded on the inside while brightly illuminated on the outside. Along the top row, from left to right, we see the evolving surface which is driven by the evolving segmentations seen immediately below. Note that the final steady state segmentations in the rightmost column (bottom two rows), which are generated by the final surface shown in the upper right corner

⁵If one is using a narrow-band level set implementation, then this extension only needs to be done in the band where the computation operates.

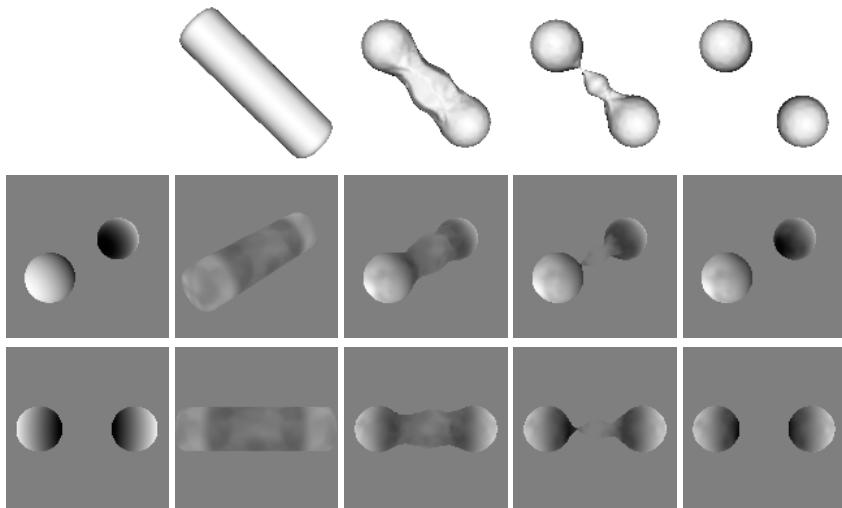


Figure 12.9. (Top) Evolution of the 3D reconstructed surface. (Bottom) Evolution of two of 16 image segmentations (original images shown in leftmost column).

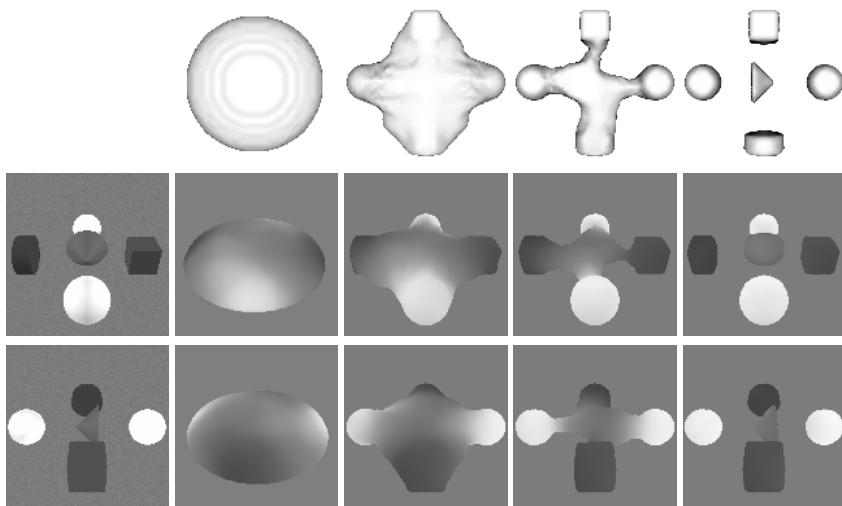


Figure 12.10. (Top) Evolution of the 3D reconstructed surface. (Bottom) Evolution of two of 16 image segmentations (original images shown in leftmost column).

of the figure, closely match the original image data in the in the leftmost column. Also note the topological change as the initial simply connected surface splits into two separate pieces for each sphere.

In Figure 12.10 we see a similar experiment on a more complicated synthetic data set, this time with additive Gaussian noise and five different objects of dif-

ferent shapes and albedos. Again, we begin with a single simply connected initial surface which splits into five separate pieces as it evolves to capture the individual objects. The final steady state surface is shown in the upper right corner of the figure. Below, in the same rightmost column, we see the final segmentations/reconstructions of two of the original images (from sixteen total) shown in the leftmost column. Below the evolving surface, from left to right, we see the evolving segmentations which are driving the deformation of the surface itself.

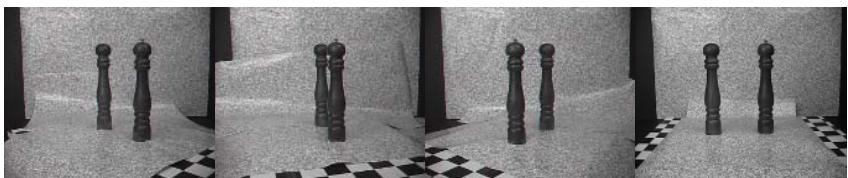


Figure 12.11. Original “salt and pepper” sequence (4 of 18 views).

Finally, in figure 12.11 we show 4 of 18 calibrated views of a scene that contains three objects: two shakers and the background. This scene would represent an insurmountable challenge to traditional correspondence-based stereo algorithms: the shakers exhibit very little texture (making local correspondence ill-posed), while the background exhibits very dense texture (making local correspondence prone to local minima). In addition, the shakers have a dark but shiny surface, that reflects highlights that move relative to the camera since the scene is rotated while the light is kept stationary.

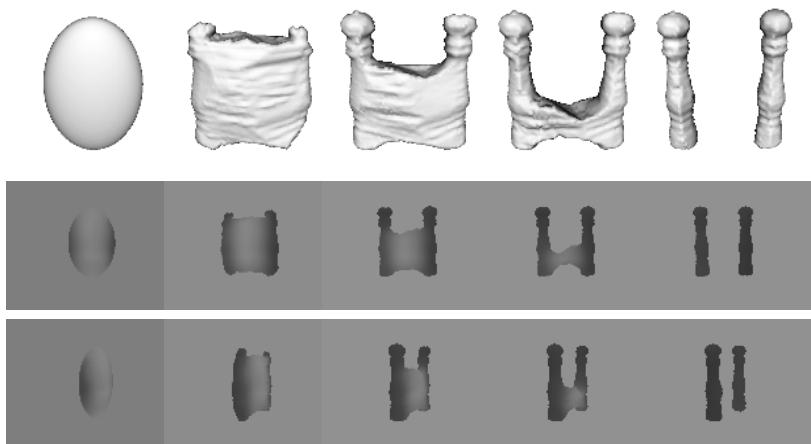


Figure 12.12. (Top) Evolution of the 3D reconstructed surface: notice that the initial surface neither contains nor is contained by the final surface. (Bottom) Evolution of two of the 18 image segmentations.

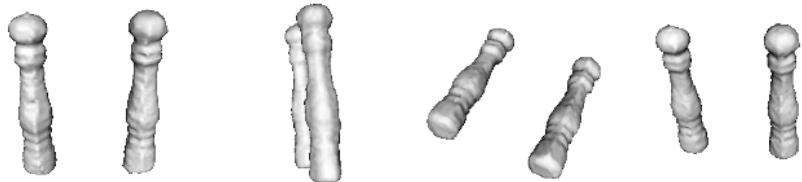


Figure 12.13. Final estimated surface shown from several viewpoints. Notice that the bottoms of the salt and pepper shakers are flat, even though no data was available. This is due to the geometric prior which, in the absence of data, results in a minimal surface being computed.

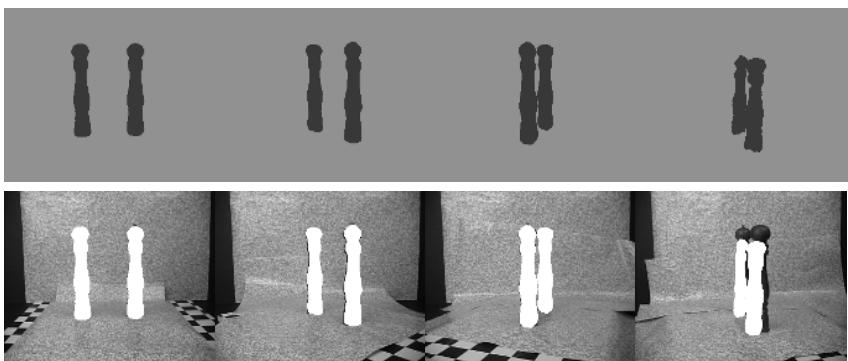


Figure 12.14. (Top) Final image segmentations for the salt and pepper sequence. (Bottom) Segmented foregrounds superimposed to the original images. The calibration of the last image (and one other not shown here) was inaccurate. However the overall reconstructed shape is only marginally affected by the calibration errors due to the global integration of all image information.

In Figure 12.12 we show the surface evolving from a large ellipse that neither contains nor is contained in the shape of the scene, to a final solid model. Notice that some parts of the initial surface evolve outward, while other parts evolve inward in order to converge to the final shape. This bi-directionality is a feature of our algorithm, which is not shared - for instance - by shape carving methodologies. There, once a pixel has been deleted, it cannot be retrieved.

In Figure 12.13 we show the final result from various vantage points. In Figure 12.14 we show the final segmentation in some of the original views (top). We also show the segmented foreground superimposed to the original images. Two of the 18 views were poorly calibrated, as it can be seen from the large reprojection error. However, this does not significantly impact the final reconstruction, for there is an averaging effect by integrating data from all views.

Acknowledgements

This research is supported in part by NSF grants IIS-9876145 and CCR-0133736, ARO grant DAAD19-99-1-0139, and Intel grant 8029.

Part V

Knowledge-based Segmentation & Registration

13

Shape Analysis towards Model-based Segmentation

Nikos Paragios and Mikael Rousson

Abstract

In this chapter, we investigate shape modeling and registration towards model-based shape-driven object extraction and segmentation. The chapter consists of two major contributions: (i) a variational level set approach for global-to-local shape registration and (ii) an energetic formulation to impose prior shape knowledge within implicit representations. Distance transforms are proven to be a very efficient feature space to perform shape registration. Prior knowledge is an important tool in the segmentation process. Following the example of shape registration, we introduce a stochastic level set prior that constrains the segmentation result to be in a family of shapes defined through a similarity transformation of the prior model. Promising results and systematic validation for each of the topics under investigation demonstrate the performance and the potentials of our approach.

13.1 Introduction

Shape analysis is a particularly interesting area of study in computer vision. The application domain is wide and includes pattern recognition, (object recognition) medical imaging (registration), image processing (segmentation), etc. where the analysis of shapes can provide strong support to image understanding and all its possible applications. Two important issues in shape analysis are shape modeling and shape registration.

Shape modeling can be stated as follows: given a selection of points in 2D or 3D space, find an appropriate representation that can describe efficiently the underlying structure from which these points originate. Limiting the number of parameters as well as efficiently estimating them are key objectives of these representations. Additionally, invariant shape representations (according to some particular transformations) are very attractive to imaging and vision applications.

Parametric curves and surfaces, volumes, or point clouds are some of the representation that can be found in the literature. Applications of shape modeling are numerous. In medical image analysis for example, one can consider the use of organ shape models for patient-drivent simulations. Registration between the model and the observed structures is then required. Recognition of faces is another example that requires proper modeling of the face using shape as well as visual descriptors.

Shape registration is a task under heavy consideration in shape recognition and medical image analysis. A general registration formulation can be stated as follows: given two shapes, an input \mathcal{D} and a target \mathcal{S} , a set of possible transformations and a dissimilarity measure, find the transformation between \mathcal{D} and \mathcal{S} that minimizes the dissimilarity measure between the transformed model shape $\hat{\mathcal{D}}$ and the target shape \mathcal{S} . This dissimilarity can be defined either along the boundaries (contours/surfaces/etc.) (*shape-based*) or in the entire region (*area-based*) enclosed by them. Application in medical imaging, motion analysis, etc. can be treated as image/region registration/segmentation problems. Furthermore, shape registration can be used as basis to shape recognition where the objective is to find from a given set of examples the shape that provides the lower dissimilarity or the higher similarity measurement with the target. Knowledge-based image segmentation can be considered as step further from shape registration and is one of the most prominent applications of shape analysis.

Using shape prior constraints to improve segmentation performance is a common technique in computer vision. Dealing with physically corrupted data, occlusions and sensor noise are some of the most common problems when trying to segment real data. In many applications the objective is to recover some specific objects of interest with known geometric shape properties. Towards this end, shape models can be used to enhance segmentation performance. Shape representation is the most important factor within this process and has to be chosen in conjunction with the optimization framework that is used to solve the segmentation problem.

The reminder of this chapter is organized as follows. In Section 2 we propose an invariant to translation, rotation and local deformations shape representation with strong discrimination power. This representation is used in section 3 within a robust statistical framework to provide an elegant solution to the problem of shape registration. Furthermore, the modeling component as well as the registration component are combined efficiently in Section 4 to introduce global shape prior constraints in the segmentation process. Discussion is part of the last section.

13.2 Shape Modeling

A core component within the task of the analysis of shapes is the underlying shape representation. The selection of this representation is a crucial aspect towards the

efficient analysis of shapes within several computer vision applications such as registration, recognition, segmentation, etc.

Point clouds is the simplest way to represent shapes in the 2D or the 3D space. Distance maps [442], snake models with elastic properties [262], Fourier descriptors, deformable models/templates [50], active shapes [135], shock graphs [465, 483] medial axis/skeletons [612] are representations that can be recovered from the literature.

Although, some of these representations are powerful enough to capture a certain number of local deformations, they require a large number of parameters to deal with important shape deformations, and they cannot deal with changes of topology. An emerging way to represent shapes can be derived using level set representations [401]. This selection is invariant to translation and rotation. Under these assumptions, given a shape C , one can define the following representation¹:

$$\Phi(x, y) = \begin{cases} 0, & (x, y) \in C \\ -D((x, y), C), & (x, y) \in \mathcal{R}_C \\ +D((x, y), C), & (x, y) \in I - \mathcal{R}_C \end{cases} \quad (13.1)$$

where \mathcal{R}_C is the region defined by the shape and $D((x, y), C)$ is the minimum Euclidean distance between the image location (x, y) and the cloud of points C .

Distance maps, have some nice properties that describe shape in a powerful way; they refer to structures of higher dimension where the information space refers to clones of the original shape positioned coherently in the image plane (iso-contours). Furthermore such representations can account for local deformations that are not visible for iso-contours that far away from the original shape. Shape geometrical properties can be naturally derived from these representations, while they refer to smooth and "monotonic" information space, with partially continuous derivatives, a desirable property for most of the existing optimization frameworks.

Shape representation is a complex procedure. In the most general case, one would like to create a compact model that can represent a family of similar objects or the same object under various conditions and view points. One way to approach the problem is by considering a set of training examples and seeking for a common representation that can best express the training set. Given a training set of N registered shapes² and their representations $[\Phi_1, \Phi_2, \Phi_3, \dots, \Phi_N]$ we would like to recover a representation Φ_G that can account for all examples in the training set while being a level set representation using distance maps as embedding function. Towards this end, we consider a stochastic framework [416], where

¹To simplify the notation, the 2D case will be considered when introducing the different components of shape analysis.

²Shape registration will be considered in the next section and therefore the registration assumption considered in the modeling phase is plausible.

the underlying shape representation refers to a stochastic level set $[p_G, \Phi_G, \sigma_G]$;

$$p_{G,(x,y)}(\phi) = \frac{1}{\sqrt{2\pi}\sigma_G(x,y)} e^{-\frac{(\phi - \Phi_G(x,y))^2}{2\sigma_G^2(x,y)}} \quad (13.2)$$

where Φ_G is the shape level set. If we assume that for every given pixel (x, y) the best value to represent the family of shapes in the training set is $\Phi_G(x, y)$, then the selected representation can account for local variations. Gaussian density functions are used pixel-wise to describe the deviation between the selected level set $\Phi_G(x, y)$ and the training values set at this particular location

$$[\Phi_1(x, y), \Phi_2(x, y), \Phi_3(x, y), \dots, \Phi_N(x, y)]$$

Estimating the parameters of such representation is not trivial. Given N registered training examples one can recover the model parameters using the maximum likelihood principle. Thus for each pixel of the image, a tuple of N values is available; the values of the training representations on this particular location. One can ignore the specific application constraints (the overall representation has to be a distance map) and then use the maximum likelihood principle to solve the inference problem, find the best estimates for Φ_G and σ_G pixel-wise. Towards this end, the use of [-log] function can be used;

$$\begin{aligned} E(\Phi_G, \sigma_G) &= - \iint_I \sum_{t=1}^N [\log(p_{G,(x,y)}(\Phi_t(x,y)))] dxdy \\ &= + \iint_I \sum_{t=1}^N \left[\log(\sigma_G(x,y)) + \frac{(\Phi_t(x,y) - \Phi_G(x,y))^2}{2\sigma_G^2(x,y)} \right] dxdy \end{aligned} \quad (13.3)$$

where we do not impose the distance transform constraint on the form of the representation Φ_G to be recovered. Furthermore, one can assume that the field σ_G is locally smooth. Due to the absence of data support during the estimation of the shape representation (limited number of samples per pixel), such smoothness constraints is a standard component when solving the inference problem;

$$\begin{aligned} E(\Phi_G, \sigma_G) &= \iint_I \sum_{t=1}^N \left[\log(\sigma_G(x,y)) + \frac{(\Phi_t(x,y) - \Phi_G(x,y))^2}{2\sigma_G^2(x,y)} \right] dxdy \\ &\quad + \alpha \iint_I \left[L_2^2 \left(\frac{d}{dx} \sigma_G(x,y) \right) + L_2^2 \left(\frac{d}{dy} \sigma_G(x,y) \right) \right] dxdy \end{aligned} \quad (13.4)$$

where α is a blending parameter. The latest objective function does not account for the distance transform constraint. One can consider the use of Lagrange multipliers to impose such a constraint. In this particular case such selection will not be helpful due to the form of the cost function and the significant number of the unknown variables. If such constraint is not considered, one can optimize the

objective function by using the calculus of variations;

$$\begin{aligned} \frac{d}{dt}\Phi_G &= 2 \sum_{t=1}^N \frac{(\Phi_t - \Phi_G)}{2\sigma_G^2} \\ \frac{d}{dt}\sigma_G &= \alpha \left[\frac{d^2}{dxdx}\sigma_G + \frac{d^2}{dydy}\sigma_G \right] + \sum_{i=1}^t \left[-\frac{1}{\sigma_G} + \frac{(\Phi_t - \Phi_G)^2}{\sigma_G^3} \right] \end{aligned} \quad (13.5)$$

The steady-state solution of the above motion equations can be used to determine the best model parameters using the set of training examples. Such solution can account for optimal data support from the training set but cannot respect the constraint of being a distance transform (Φ_G).

Optimization problems referring to solutions that are part of a constrained manifold are common in imaging and vision. Definition of an objective function constrained within this manifold is the optimal solution that can be considered. Such techniques are not always tractable when large number of unknown variables are considered. One can consider to decouple the problem into two stages. Within the first stage (fitting) the objective is to find the appropriate solution with maximal data support without considering the specific constraints. The second stage aims at finding the projection of this solution to the desired manifold. These two steps can alternate until a steady-state solution is obtained (i.e. bundle adjustment).

Projection of the current solution of the level set representation of the model (Φ_G) to the space of distance transforms can be considered within our approach. Once an appropriate projection of the solution is obtained, then it can be used as a starting point in the first (fitting) stage of the new iteration; estimation of the model parameters that best account for the shape information provided by the training data.

Given a shape form (zero-level set), several methods to obtain a representation that is a distance map of this shape can be found in the literature. Our approach has to deal with a more challenging task since the objective is to obtain a distance transform from an image that refers to a level set representation. The extraction of the shape using zero-crossings first and then the creation of a distance map is not an proper solution since the boundaries of the shape can be displaced unless super-resolution of fine sub-pixel accuracy is considered. In order to overcome this limitation, in [501] a partial differential equation that performs this task - without altering the boundaries - was proposed;

$$\frac{d}{dt}\Phi_G = \text{sign}(\Phi_G^0)(1 - |\nabla\Phi_G|) \quad (13.6)$$

where Φ_G^0 is the start point of the evolution process. One can easily alternate between the equation. (13.5) and equation. (13.6) to recover a solution that best expresses the training set while being a distance transform.

The obtained shape representation has all the advantages of distance transforms and can account for local variability. The recovery of the variability field [σ_G] leads to a natural way to separate the rigid from the non-rigid parts for a given family of shapes. Such property can be used accordingly to improve performance

of shape analysis techniques like registration where the shape sub-components can be considered in a quality-driven manner. Modeling techniques with some conceptual similarities with our shape representation can be found in [564].

13.3 Shape Registration

Shape registration has been approached into a number of ways [328]. Classification of these methods can be done according to several criteria like: (i) nature of transformation, (ii) domain of transformation and (iii) optimization procedure.

The underlying motion model (nature of transformation) that is used to map the current shape to the target is a critical component that can have an important impact on the performance of the registration procedure. The simplest model refers to rigid transformations (translation, rotation and scale), a compromise between low complexity and fairly acceptable matching between the different structures when they present a dominant rigid component. Affine transformations are quite popular and a step further due to their invariance to a variety of motions. Projective motion models as well as elastic methods have been also considered to match shapes. Regarding the transformation domain one can separate global transformations (valid for the entire shape) and local ones (pixel-wise deformation models). The selection of a mathematical framework to provide the solution by means of finding an optimum of some functional defined on the registration parameter space is another classification criterion. These functionals attempt to quantify the similarity between the two shapes and can be based either in variational [105], or stochastic principles [555].

Euclidean distance transforms [442] is the most closely related representation with the one adopted for shapes by our approach. They have been considered mainly for image registrations in the past [126, 197, 294, 304, 547]. Comparison between these methods and the proposed framework can be found at [416]. The objective of our work is to provide a robust global-to-local solution to the registration problem. Towards this end, we introduce an optimization criterion that can account for global (rigid) and local pixel-wise deformations. This criterion is defined in the space of signed distance transforms, and is optimized using a gradient descent method. Global and local registration parameters are recovered simultaneously with emphasis on the global model. In order to facilitate the presentation of our approach we will consider a rigid transformation $A = (s, \theta, T)$ for the global motion model that consists of a translation component $T = (T_x, T_y)$, a rotation angle θ and a scale parameter s .

Level set shape representations can augment the potentials on the registration task. A higher dimension is to be considered, where the objective is to recover a transformation A that creates pixel-wise intensity correspondences (level sets can be seen as intensity images) between the current shape representation Φ_D and the target shape Φ_S . We can easily prove that the selected representations $[\Phi_D, \Phi_S]$ are invariant to translation and rotation [416, 417]. Dealing with scale



Figure 13.1. Shape registration using level set representations and global rigid models. Evolution of the registration process according to the projection of the source shape to the target shape using the estimates of the motion model; presented in a raster-scan format.

variations is not a standard property of these representations and requires some particular handling. One can easily show that by up-scaling or down-scaling the level set representations according to the global scale registration parameter, rigid invariant representations can be recovered;

$$A = (s, \theta, T)$$

$$A(x, y) = \begin{pmatrix} A_x \\ A_y \end{pmatrix} = s \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} T_x \\ T_y \end{pmatrix} \quad (13.7)$$

$$\forall (x, y) \in \Omega : s \underline{\Phi_D(x, y)} = \underline{\Phi_S(A(x, y))}$$

Registration of shapes in the 2D plane can be viewed as a global optimization problem involving all pixels in the image plane. Several techniques can be used to recover the unknown transformation parameters like the sum of squared differences, robust estimators, optimization of the correlation ratio, mutual information, etc. In order to introduce and demonstrate the performance of our method, at the very beginning we will consider the simplest possible criterion, the sum of square differences.

13.3.1 Sum of Square Differences

Shape registration of contours on the level set space can be considered as an image registration problem, a widely explored application, where the objective is to recover a global parametric transformation that creates intensity-driven correspondences with a minimal error. The sum of square differences can be used to measure the quality of the motion map;

$$r(x, y) = s \Phi_D(x, y) - \Phi_S(A(x, y))$$

$$E(s, \theta, T) = \iint_{\Omega} (s \Phi_D(x, y) - \Phi_S(A(x, y)))^2 dx dy \quad (13.8)$$

where r is the residual term and the scale factor s appears in the objective function to cope with scale variations. Registration is done in an augmented shape-driven (level set) space that is robust to very local deformations and missing data since the selected representation is obtained through a global procedure (Euclidean distance). Moreover, the proposed framework is invariant to rigid

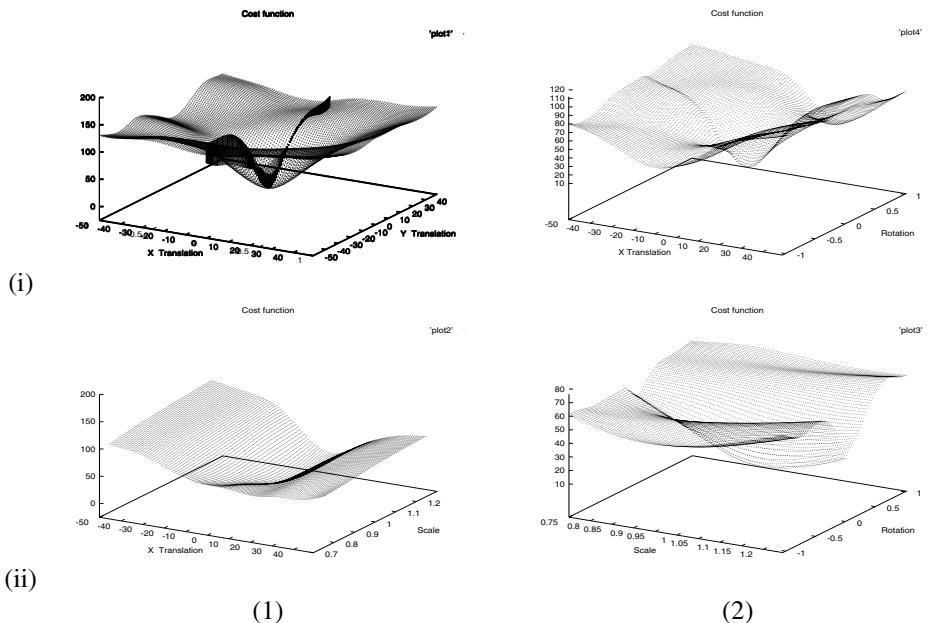


Figure 13.2. Empirical evaluation of the cost function: (i.1) Unknown translation $[x, y]$, (i.2) Unknown transaltion $[x]$ and rotation, (ii.1) Unknown translation $[x]$ and scale, (ii.2) Unknown scale and rotation.

transformations and involves multiple matching of shapes (isophotes) that are clones of the original ones to be registered. Using the proposed formulation we were able to convert a geometry driven point-correspondence problem into an image-registration application where space as well feature-based (intensity) correspondences are considered.

One can consider a gradient descent method to find the minimum of the objective function figure (13.1). Convexity of the objective function is a desirable property that can lead to a unique global solution and can be recovered by a gradient descent method. Distance transforms have a certain number of desirable properties that can provide some support towards this property. They are smooth and have partially continuous derivatives. A theoretical proof regarding the convexity of the objective function is not trivial, and therefore one can consider an experimental validation.

Towards this end, we have considered the shapes shown in figure (13.1). Evaluation of the objective function in the 4D optimization space is rather impossible. In order to perform some empirical validation, one can consider several 2D sub-optimization spaces; (i) fixed scale & rotation, unknown translation, (ii) fixed translation, unknown rotation & scale, (iii) fixed rotation & vertical translation, unknown scale & horizontal translation and (iv) fixed rotation & horizontal translation, unknown scale & vertical translation. Then, for validation purposes, one can quantize the search space (uniform sampling of 100 elements are used) for

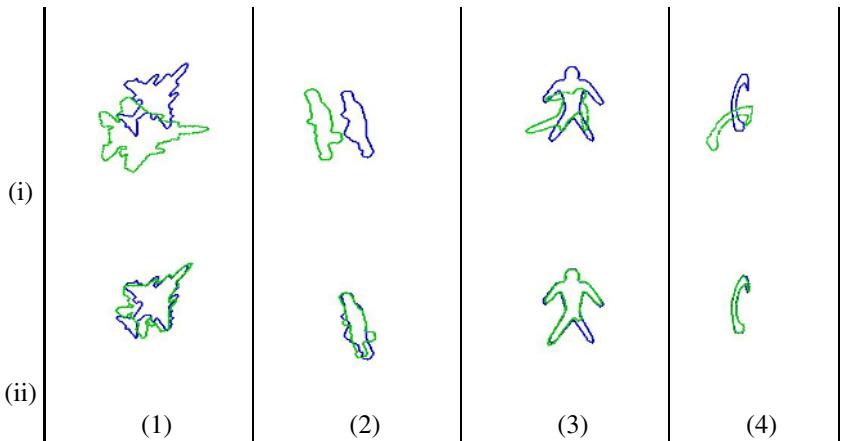


Figure 13.3. Global-registration for different shape categories using level set representations and global rigid models. (i) Initial Conditions (source: gray, target:dark) , (ii) Registration result according to the projection of source shape to the target shape using the recovered motion parameters of the rigid model.

the translation between $[-50, +50]$, the rotation between $[-\frac{\pi}{3}, +\frac{\pi}{3}]$ and the scale space between $[0.75, 1.25]$ and evaluate the objective function as shown in figure (13.2). Then as shown from the experiments, form of the objective function can support the convex hypothesis.

Gradient descent is a standard optimization technique given the form of the objective function leading to the following partial differential equations;

$$\begin{aligned} \frac{d}{dt}\theta &= 2 \iint_{\Omega} (\nabla \Phi_S \cdot \nabla_{\theta} A) [(s\Phi_D - \Phi_S(A))] \\ \frac{d}{dt}s &= 2 \iint_{\Omega} (\Phi_D + \nabla \Phi_S \cdot \nabla_s A) [(s\Phi_D - \Phi_S(A))] \\ \frac{d}{dt}T &= 2 \iint_{\Omega} \left(\nabla \Phi_S \cdot \begin{pmatrix} \nabla_{T_x} A \\ \nabla_{T_y} A \end{pmatrix} \right) [(s\Phi_D - \Phi_S(A))] \end{aligned} \quad (13.9)$$

The performance of the proposed module is shown in figure (13.3).

13.3.2 Robust Estimators

Robustness is a critical component to registration algorithms. In order to recover un-biased estimates using least-squares methods, numerous hard assumptions have to be met. In the literature one can recover studies showing the instability of least squares estimators when a very small number of outliers is present [246]. In our case due to the selected shape representation, the rich information space and the data support this problem was not observed unless extreme initial conditions are considered (experimental validation of the convexity property of the objective function). However, local non-rigid deformations can produce outliers.

Robust estimators is an alternative to the least-squares ones and can deal with perturbed data.

M-estimators is a well known technique that can deal with outliers. Towards this end, the squared residuals are replaced by another function;

$$E(s, \theta, T) = \iint_{\Omega} \rho(r(x, y)) dx dy \quad (13.10)$$

where ρ is a symmetric and positive-definite function with a unique minimum at zero. One then can define the following influence $\psi(r) = \frac{d}{dr}\rho(r)$ and weight function $\omega(r) = \frac{\psi(r)}{r}$. Some constraints are to be met by M-estimators; bounds on the influence function, uniqueness that reflects to convex functions and non-zero gradient.

A gradient descent method can be also used to recover the solution from objective function leading to:

$$\begin{aligned} \frac{d}{dt} \theta &= \iint_{\Omega} \psi(s\Phi_D - \Phi_S(A)) (\nabla \Phi_S \cdot \nabla_{\theta} A) [(s\Phi_D - \Phi_S(A))] \\ \frac{d}{dt} s &= \iint_{\Omega} \psi(s\Phi_D - \Phi_S(A)) (\Phi_D + \nabla \Phi_S \cdot \nabla_s A) [(s\Phi_D - \Phi_S(A))] \\ \frac{d}{dt} T &= 2 \iint_{\Omega} \psi(s\Phi_D - \Phi_S(A)) \left(\nabla \Phi_S \cdot \begin{pmatrix} \nabla_{T_x} A \\ \nabla_{T_y} A \end{pmatrix} \right) [(s\Phi_D - \Phi_S(A))] \end{aligned} \quad (13.11)$$

The selection of the influence function is critical. Several alternatives can be found in the literature. We have considered two well known functions, the *fair* and the *Cauchy* given by

$$\rho_{fair}(r) = c^2 \left[\frac{|r|}{c} - \log \left(1 + \frac{|r|}{c} \right) \right], \quad \rho_{cauchy}(r) = \frac{c^2}{2} \log \left(1 + \left(\frac{r}{c} \right)^2 \right) \quad (13.12)$$

The *fair* function has continuous derivatives up to third order and can provide unique solution while *Cauchy* function does not guarantee a unique solution and has the tendency to yield erroneous solutions due to the form of its first derivative. However such selection can eliminate to a significant factor the influence of large errors. One can consider the combination of these two functions [246]; using the convex one (*fair*) until convergence and then apply the *Cauchy* to eliminate large errors.

One can now interpret the robust registration flows presented in equation (13.11). The influence function can be used to decrease the effect of outliers. In particular, modulo the selection of the function parameter one can recover a form for the weight function ψ that can ignore the use of outliers during the estimation process.

13.3.3 Global-to-Local Registration

Global registration has limited applicability when non-rigid shapes are considered. In that case global methods will be able to recover the global motion of the structure but will fail to capture the motion details. Stability and robustness are the main characteristics of global methods while accuracy and precise registration can be the outcome of local methods. The integration of global motion models with local deformation fields is a more appropriate way to tackle the registration problem.

Towards this end, we assume that the observed shape is a rigid transformation A of the target combined with some local deformations (u, v) defined in the pixel level for the shape components that do not follow the global motion model. This assumption can lead to following condition between the level set representations of the source and the target shape;

$$\begin{aligned}
 E(s, \theta, T, (u, v)) = & \underbrace{\delta \iint_{\Omega} u^2 + v^2}_{magnitude\ constraint} + \\
 & \alpha \underbrace{\iint_{\Omega} \rho(r_{rigid})}_{global\ registration} + \beta \underbrace{\iint_{\Omega} \rho(r_{rigid+local})}_{local\ deformations} \\
 & \gamma \underbrace{\iint_{\Omega} \left(L_2^2 \left(\frac{d}{dx} u \right) + L_2^2 \left(\frac{d}{dy} u \right) + L_2^2 \left(\frac{d}{dx} v \right) + L_2^2 \left(\frac{d}{dy} v \right) \right)}_{smoothness\ constraint}
 \end{aligned} \tag{13.13}$$

where $\alpha, \beta, \gamma, \delta$ are blending parameters. The minimization of this functional is done using a gradient descend approach and the calculus of variations;

$$\begin{aligned}
 \frac{d}{dt} s &= \alpha \iint_{\Omega} \psi(s\Phi_D - \Phi_S(A)) \left(\Phi_D - \frac{\partial}{\partial s} \Phi_S(A) \right) + \\
 &\quad \beta \iint_{\Omega} \psi(s\Phi_D - \Phi_S(A + (u, v))) \left(\Phi_D - \frac{\partial}{\partial s} \Phi_S(A + (u, v)) \right) \\
 \frac{d}{dt} (\theta, T) &= -\alpha \iint_{\Omega} \psi(s\Phi_D - \Phi_S(A)) \left(\frac{\partial}{\partial \theta}, \frac{\partial}{\partial T} \right) \Phi_S(A) - \\
 &\quad \beta \iint_{\Omega} \psi(s\Phi_D - \Phi_S(A + (u, v))) \left(\frac{\partial}{\partial \theta}, \frac{\partial}{\partial T} \right) \Phi_S(A + (u, v)) \\
 \frac{d}{dt} (u, v) &= 2\gamma \left(\frac{d^2}{dxdx}(u, v) + \frac{d^2}{dydy}(u, v) \right) - 2\delta (u, v) + \\
 &\quad - \beta \iint_{\Omega} \psi(s\Phi_D - \Phi_S(A + (u, v))) \left(\frac{\partial}{\partial u}, \frac{\partial}{\partial v} \right) \Phi_S(A + (u, v))
 \end{aligned} \tag{13.14}$$

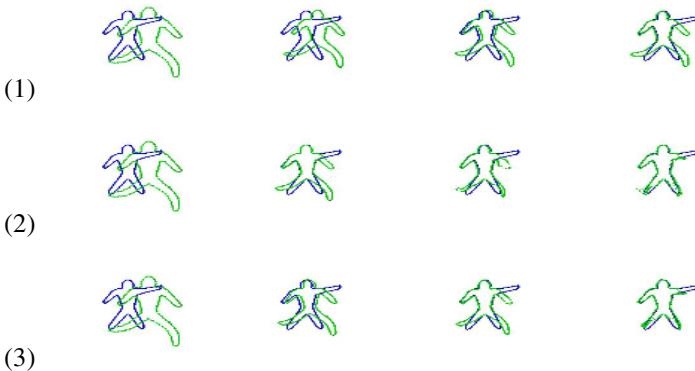


Figure 13.4. (1) Global $\{s = 0.79, \theta = 1.34^\circ, T_x = -16.34, T_y = -15.76\}$, (2) Global-to-Local $\{s = 0.81, \theta = 2.07^\circ, T_x = -14.97, T_y = -15.43\}$, (3) Local Registration with regularization constraints.

13.3.4 Experimental Results & Validation

Some results on the performance of the complete system are shown in figures (13.3,13.4,13.5). In order to validate our approach, we have considered the following experiments; (i) recover known registration parameters, (ii) validate the performance of the method to missing and occluded parts when registration parameters are known. Towards this direction, we have used two sets of real shapes figures (13.1,13.3.3) where the source was globally deformed using a set of known transformations generated according to a random process in the four dimensional parameter space (s, θ, T_x, T_y):

$$(\theta, s, T_x, T_y) \in \left([- \frac{\pi}{3}, \frac{\pi}{3}], [0.8, 1.2], [-30, 30], [-30, 30]\right]$$

Using this random variable, 100 registration trials were considered to determine the global transformation. The proposed algorithm (equation (13.11)) was used to estimate the known inverse transformation and was able to recover the optimal registration parameters in all cases. In order to have a more reliable validation, registration for the cases of occlusions as well missing components was considered using the same random generator. As shown in figure (13.5) the performance of the method was excellent for moderate and satisfactory for heavy occlusions..

13.4 Segmentation & Shape Prior Constraints

Segmentation can be the basis to many image processing and computer vision applications [613]. Such a problem has an enormous complexity due to the variety of conditions to be dealt with.

Propagating curves and surfaces is an attractive approach to cope with this application domain. Inspired by the pioneering work of Witkin, Kass and Ter-

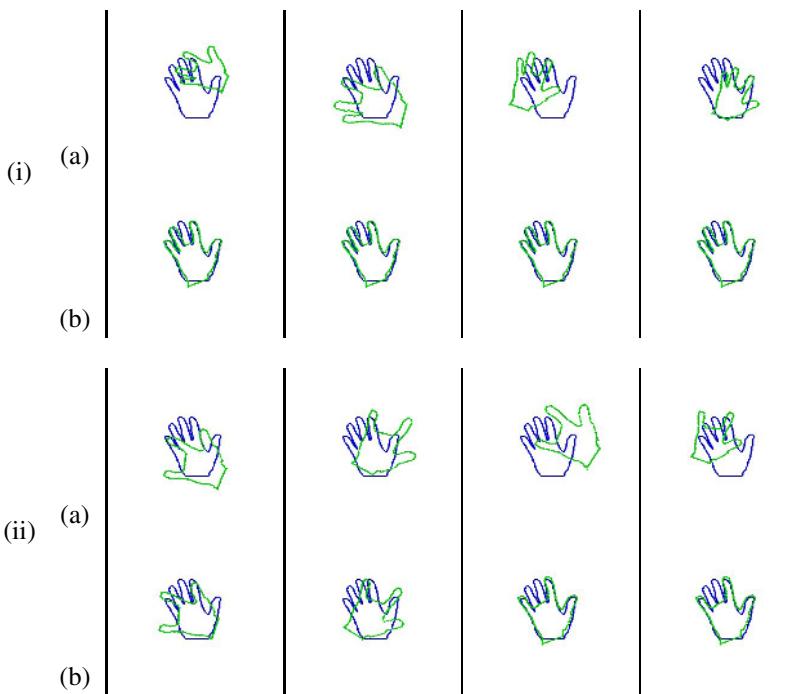


Figure 13.5. *Empirical Evaluation to Occlusions & Missing Components;* (i) moderate shape deformation (one missing finger) - Registration Performance: 100 %, (ii) heavy shape deformation (two missing fingers)- Registration Performance: 77 %. (a) Initial Conditions, (b) Registration Result.

zopoulos [262] numerous techniques have been proposed that aim at recovering a curve that corresponds to the lowest potential of an objective function. This function can account for the desired image properties as well as some internal properties of the curve.

Level set representations [401] are well suited computational methods to perform this task, tracking moving curves/interfaces. They can be used for any dimension (curves, surfaces, hyper-surfaces, ...), are parameter free and can change naturally the topology of the evolving interface. Moreover, they provide a natural way to determine and estimate geometric properties of the evolving interface.

Our visual space consists of rigid as well as non-rigid objects. Performance of geometric flows when implemented using level set methods for non-rigid objects is expected to be exceptional since these methods can account for very local deformations [81, 330, 415]. On the other hand sensitivity to noise as well as to occlusions when rigid objects are considered is expected to be poor.

Medical imaging is an area of growing attention in the vision community. Many structures of interest in this domain cannot be considered neither rigid nor non-rigid. One can claim that they follow some global shape consistency, while important local deformations can also be present. Recovering these deformations

is of great importance and therefore geometric flows implemented using level sets is a promising technique. Deformable models/templates [50, 360] is an alternative that can account for global shape consistency and local variations when the model is represented using an important number of basic functions.

Introducing global shape constraints according to a flexible model while preserving the ability of dealing with local deformations when level set-driven methods are used is a challenging perspective [111, 310, 444, 526]. Towards this end, we will combine the shape model and the registration components introduced in the previous sections to deal with a limited segmentation problem; the extraction of objects of particular interest.

Given an image and a model for a structure of interest, one would like to recover the image region that corresponds to this structure. Such region should follow some shape as well as visual-driven intensity properties. If we assume that the region is known, then it may be helpful also to recover the transformation between this region and the shape model. In medical imaging diagnostic purposes can require such information where important local or global deviations from the shape can be used as an indicator for diseases. Our approach will be based on the propagation of curves and address both issues simultaneously. Recovering the image region that corresponds to the structure of interest and registering the segmentation result to the prior shape model.

13.4.1 Prior Global Shape Constraints

Inspired by the existing geometric flows, one can assume the propagation of an initial curve/interface $C = \partial\mathcal{R}$ using an implicit level set representation Φ according to some data driven terms towards the optimal segmentation result according to the following assumptions;

- Visual support had to be used to derive the optimal segmentation map;
- This representation is evolving while respecting the global shape properties of the object to be recovered. In other words the evolving contour had to be registered to the prior model;
- Given the registration between the evolving representation and the prior shape, one should refine its local form to better account for the prior shape knowledge while respecting the global visual properties of the object.

Then, rigid registration between the evolving representation and the prior model can be recovered. However, one can assume that this global transformation is known. Then, evolving locally the level set representation to better match with the prior model given the correspondence map (registration) is a task that has to be considered

In order to facilitate the introduction of a segmentation module that can account for global shape consistency given a model Φ_G , one can define the approximations

of Dirac and Heaviside [607] distributions as:

$$\delta_a(\phi) = \begin{cases} 0, |\phi| > \alpha \\ \frac{1}{2\alpha} \left(1 + \cos\left(\frac{\pi\phi}{a}\right) \right), |\phi| < \alpha \end{cases}$$

$$H_\alpha(\phi) = \begin{cases} 1, \phi > \alpha \\ 0, \phi < -\alpha \\ \frac{1}{2} \left(1 + \frac{\phi}{\alpha} + \frac{1}{\pi} \sin\left(\frac{\pi\phi}{a}\right) \right), |\phi| < \alpha \end{cases}$$

Then it can be shown easily that

$$\{(x, y) \in \Omega : \lim_{\alpha \rightarrow 0^+} [H_\alpha(\Phi((x, y); t))] = 1\} = \mathcal{R}$$

$$\{(x, y) \in \Omega : \lim_{\alpha \rightarrow 0^+} [\delta_\alpha(\Phi((x, y); t))] = 1\} = \partial\mathcal{R}$$

The approximations of Dirac [$\delta_\alpha()$] and Heaviside [$H_\alpha()$] distributions can be used to introduce optimization criteria in the space of level set representations as shown in [607]. Towards this end, given a level set representation Φ , one can define boundary as well as region-driven modules;

$$(i) \underbrace{\int_{\partial\mathcal{R}} b(I(p)) dp}_{\text{boundary module}}, \quad \underbrace{\iint_{\Omega} \delta_\alpha(\Phi(x, y)) b(I(x, y)) dx dy}_{\text{boundary module}}$$

$$(ii) \underbrace{\iint_{\mathcal{R}} r(I(x, y)) dx dy}_{\text{regional module}}, \quad \underbrace{\iint_{\Omega} H_\alpha(\Phi(x, y)) r(I(x, y)) dx dy}_{\text{regional module}} \quad (13.15)$$

where r and g are *region* and *boundary* attraction functions.

In order to introduce the shape constraint, one can assume that all instances of the evolving representation belong to the family shapes that is generated by applying all possible global transformations (according to a predefined model) to the prior shape model. Then, there is an ideal transformation A between the prior model Φ_G and the observed representation Φ that satisfies (similar to the registration case) the following condition;

$$(x, y) \rightarrow A(x, y)$$

$$s\Phi(x, y) \approx \Phi_G(A(x, y)), \quad \forall(x, y) : H_a(\Phi(x, y)) \geq 0$$

where Φ is an evolving representation. In order to better introduce our global shape consistency component, the absence of visual information will be considered and the problem can be stated as follows; given a initial contour and its level set representation perform a pixel-wise propagation that transforms this contour to a structure that belongs to the family of shapes defined by the shape prior. One can consider the same objective function as the one used for the registration;

$$r(x, y) = s\Phi(x, y) - \Phi_G(A(x, y))$$

$$E(\Phi, A) = \iint_{\Omega} H_\alpha((\Phi(x, y)) \rho(r(x, y))) dx dy \quad (13.16)$$

with an augmented set of unknown variables; the transformation A and the level set representation Φ . Then, a gradient descent method with respect to the registration parameters as well as to Φ can be used to recover their optimal estimates. The motion equations with respect to the transformation A are similar to the ones presented in equation (13.9) while the level set representation Φ has to evolve according to:

$$\frac{d}{dt}\Phi(x, y) = -H_\alpha((\Phi(x, y))s\psi(r(x, y)) - \delta_\alpha(\Phi(x, y))\rho(r(x, y))) \quad (13.17)$$

A detailed interpretation of this flow can be found in [444]. The second component of this flow is a constant deflation force that aims at shrinking the evolving interface and consequently decreasing the load of the objective function. This component can be ignored. Some results on the performance of this flow for preserving a global shape consistency within the propagation of level set representations along with the projection to the prior model (registration) are shown in figure (13.6).

Shape variability is not considered in this model and one can claim that the prior constraint is hard. Given the definition of the prior model, one can easily modify the objective function to better account for shape variations. The stochastic level set representation (Φ_G, σ_G) introduced in equation (13.2) can be used and one can seek the maximum likelihood between the evolving representation Φ and the prior model leading to the following condition;

$$(x, y) \rightarrow A(x, y) \\ p_{G, A(x, y)}(\Phi(x, y)) \rightarrow \sup_{\phi \in R} \{p_{G, A(x, y)}(\phi)\} \quad \forall (x, y) : H_a(\Phi(x, y)) \geq 0 \quad (13.18)$$

Recovering Φ and A according to the Maximum Likelihood criterion is equivalent with minimizing the [-log] function leading (after the subtraction of the constant terms) to;

$$\rho(r) = r^2, \quad \psi(r) = \frac{d\rho}{dr} = 2r \\ E(\Phi, A) = \iint_{\Omega} H_\alpha((\Phi(x, y)) \left\{ \log(\sigma_G(A(x, y)) + \frac{\rho(r(x, y))}{2\sigma_G^2(A(x, y))}) \right\} dx dy \quad (13.19)$$

The interpretation of the above function is different to the one used for registration. The shape is considered in a qualitative manner. Shape components with low variance correspond to areas where the training examples are perfectly aligned during the learning phase. It is natural that these components are more important and therefore they tend to load the objective function more than the ones with high variability. As a consequence, registration as well as local propagation it to be done efficiently for the shape parts with low variability. A gradient descent

method can be used to recover the registration parameters³ and the flow to be used for the deformation of the evolving representation Φ (the deflation force is not considered).

$$\begin{aligned} \frac{d}{dt}\Phi(x, y) &= -s H_\alpha(\Phi(x, y)) \frac{\psi(r(x, y))}{2\sigma_G^2(A(x, y))} \\ \frac{d}{dt}s &= \iint_{\Omega} H_\alpha(\Phi) \left(\frac{\nabla\sigma_G(A) \cdot \nabla_s A}{\sigma_G(A)} + \right. \\ &\quad \left. + \frac{\psi(r)\sigma_G(A) \frac{\partial\rho}{\partial s} r - 2\rho(r) \nabla\sigma_G(A) \cdot \nabla_s A}{2\sigma_G^3(A)} \right) \end{aligned} \quad (13.20)$$

where the image coordinates (x, y) have been omitted from the registration flow. The introduction of the shape uncertainties to the segmentation/registration process can further improve the registration performance especially when global transformations are considered for objects with non-rigid components.

Robust estimators have been used to deal with the non-rigid components when global registration is considered. The shape variability map provides a natural way to better account for their inconsistency with the global motion. According to these (robust) estimators, the condition that at least 50% of the samples supports the dominant case has to be met in order to recover reliable registration estimates when outliers are present. The use of variability maps (as it will be shown later, see equation (13.19)) can remove this constraint and theoretically can provide accurate estimates with significantly less data support.

13.4.2 Visual Support, Shape Constraints & Segmentation

Numerous variational frameworks based on the propagation of curves using level set representations have been proposed for image segmentation [11, 84, 100, 270, 456, 593]. Our prior shape model can be naturally integrated with these frameworks as an additional shape-driven term along with data-driven terms.

Following our previous work, we will consider this integration within the geodesic active region model [404, 409, 411] that aims at combining boundary (in the form of Geodesic Active Contours [84, 270]) with some regional/global properties of the object to be recovered. The original model was defined on the image plane, and the obtained motion equations were implemented using level set representations. Here, we will introduce a shape-constrained version of this model directly on the level set representation space. We will consider the bi-modal segmentation case to facilitate the introduction.

An evolving level set representation can be used to define an image partition into two regions, the inner region \mathcal{R}_I and the background $\mathcal{R}_B = \Omega - \mathcal{R}_I$. Given this partition, one can estimate some global intensity (region-based) descriptors

³The motion equation of scale component is presented. Same derivations are applicable for the other unknown variables of the rigid transformation.

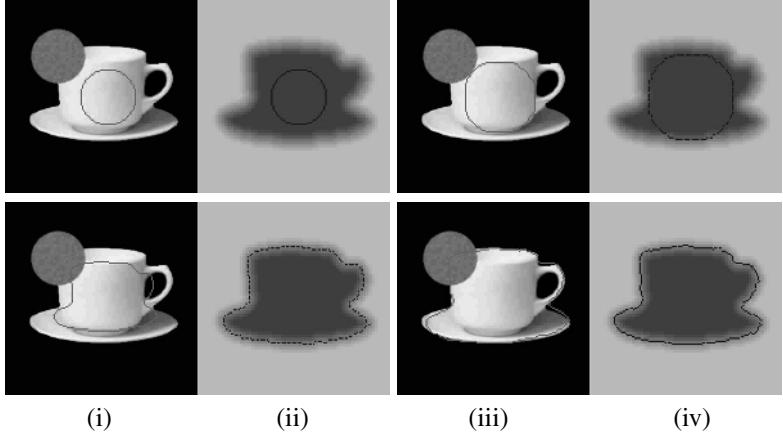


Figure 13.6. Knowledge-based Image Segmentation (raster-scan format; (i, iii) contour propagation in the image plane presented in a raster0scan format, (ii, iv) registration of the current result to the prior model using the projection of the evolving interface to the prior model according to the current estimates of the global rigid model parameters.

p_I, p_B that can account for the desired visual/intensity properties (underlying distributions) of these regions. Such descriptors can capture the visual properties of the structure of interest and can be assumed to be known or determined and updated within the process. The geodesic active region model aims at partitioning the image according to boundary as well region-driven criteria;

$$\begin{aligned}
 E(\Phi, A) = & w_1 \iint_{\Omega} \delta_\alpha((\Phi)) g(|\nabla I|) |\nabla \Phi| d\Omega + \\
 & w_2 \iint_{\Omega} H_\alpha((\Phi)) \left\{ \log(\sigma_G(A)) + \frac{\rho(r)}{2\sigma_G^2(A)} \right\} d\Omega - \\
 & w_3 \iint_{\Omega} H_\alpha(\Phi) \log(p_I(I)) + (1 - H_\alpha(\Phi)) \log(p_B(I)) d\Omega
 \end{aligned} \quad (13.21)$$

where w_1, w_2, w_3 are constants balancing the contribution of the different terms. The first two terms account for visual consistency properties⁴ of the segmentation map (region of interest, background) [613], the third term stands for the boundary attraction ($g()$) is a monotonically positive decreasing function that can be replaced with more efficient edge detectors [151] [84, 270] and the last term is the shape prior applied to the interface [444]. As shown in [444], the method can be extended to deal with N objects using the same number of evolving functions under the assumption that N shape prior models are available.

A detailed interpretation of this objective function can be found in [409, 444]. Conceptually, we are seeking for a curve (level set representation) of minimal length attracted by the region boundaries, that defines an image partition sup-

⁴Homogeneity is a particular case of this condition.

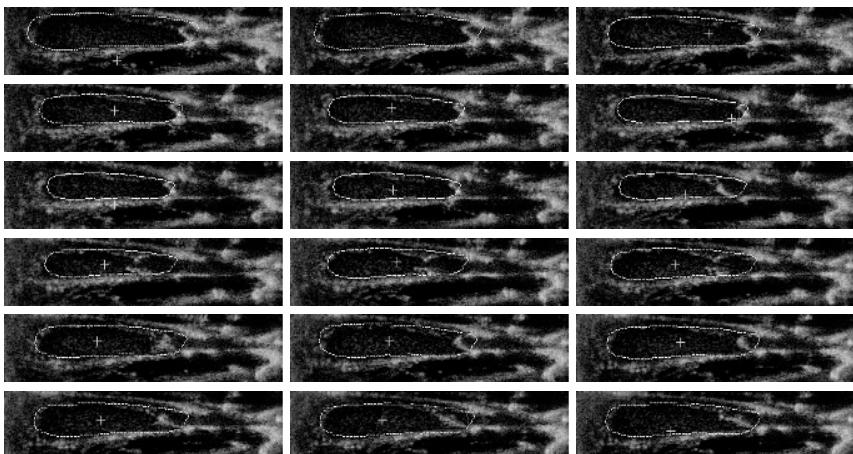


Figure 13.7. Shape Priors, Level Set Representations for the real-time segmentation of ultrasonic images (temporal results presented in a raster-scan format).

ported by the visual/intensity information given some prior knowledge of the desired visual properties of the structure of interest and follows global shape consistency according to a known prior model.

13.5 Discussion

In this chapter we have explored some aspects of shape analysis. Level set representations have been used for shape representation and have been integrated within a variational framework to deal with registration. A novel term for introducing global shape constraints was proposed to better account for the segmentation problem when occlusions or corrupted information is to be dealt with.

The use of non-parametric level set models is required for family of shapes with significant variability. This issue is to be addressed along with the integration of intensity properties to an improved version of shape model that does account for the geometric as well as visual properties of the object to be recovered. Alternative techniques like the use of mutual information are currently under investigation to perform real-time segmentation/registration and tracking. Changes of topology is the most solid argument for using level set methods. Our approach to image segmentation doesn't support changes of topology within the prior shape model. One can claim that this is an important limitation of the method. The implicit representation of curve using distance functions is a powerful way to describe shapes and provides a natural way to estimate their geometric properties. Such properties make these techniques suitable to a large number of applications. A natural extension is to be able to deal with multiple objects using a single level set repre-

smentation that can change the topology while respecting some shape properties of the objects to be recovered.

Acknowledgments

The author would like to acknowledge Benedicte Bascle, Marie-Pierre Jolly, Visvanathan Ramesh and Chenyang Xu for fruitful discussion and suggestions.

14

Joint Image Registration and Segmentation

Baba Vemuri and Yunmei Chen

Abstract

Image segmentation is ubiquitous in computer vision and image processing. In some applications such as in medical imaging, the problem may be very complex due to lack of sufficient image contrast, signal to noise ratio, volume averaging, inhomogeneities caused due to nonuniform magnetic field – in the case of MRI data sets, and sometimes lack of any real boundary due to the desired shape blending into the surrounding tissue. Such scenarios pose a formidable segmentation problem and provide motivation for incorporation of prior information in to the segmentation algorithms. There are several ways to incorporate priors into the segmentation formulation and we choose to use an atlas-based segmentation technique which requires that the given atlas shape/image be registered with the target image in order to find the desired shape segmentation in the target image thus requiring simultaneous registration and segmentation to be achieved. This is a natural choice for medical applications pertinent to some of the aforementioned scenarios. In this chapter, two approaches are presented, one is a pure PDE-based formulation and the other is a variational formulation involving a level-set representation of the shape being segmented. The former approach copes with 3D non-rigid motion and the latter although easily applicable to the non-rigid motion case is shown here only for the 2D rigid motion case. Efficient numerical methods are used in both the approaches and the performance of the algorithms is depicted via examples on synthetic and real 3D and 2D image data.

14.1 Introduction

Segmentation of noisy image data is a fundamental problem in Computer Vision and Image Processing. There are numerous existing methods in literature for achieving this goal. In some applications domains such as Medical Imaging,

segmentation can be a daunting task due to possibly large inhomogeneities in image intensities across an image e.g., in Magnetic Resonance (MR) images. These inhomogeneities combined with volume averaging during the imaging and possible lack of precisely defined shape boundaries for certain anatomical structures complicates the segmentation problem immensely. One possible solution for such situations is atlas-based segmentation where the atlas is a ground-truth segmentation. The atlas can be constructed in various ways and it allows one to incorporate prior information about the anatomical structure of interest and this information can be input from one or more experts. The atlas once constructed can be used as a template and can be registered non-rigidly to the image being segmented (henceforth called a target image) thereby achieving the desired segmentation. This approach has been quite popular in the recent past for segmenting cortical and sub-cortical structures in the human brain from MRI images. We will now briefly review some of these approaches in the following.

14.1.1 Previous Work: Brief Review

Many of the methods that achieve atlas-based segmentation are based on estimating the non-rigid deformation field between the atlas image and the target image and then applying the estimated deformation field to the desired shape in the atlas to achieve the segmentation of the corresponding structure in the target/subject image. In the following, we will briefly review some of these methods.

In Chen et.al., [106], an atlas-based segmentation scheme is described that uses expert input to define the atlas which is then warped to the subject brain MRI to segment the subject brain. The coordinate warp was determined using the popular sum of squared differences (SSD) measure defined over the image pair intensity functions. The warp was computed only at the control points and interpolated elsewhere using B-spline basis, thus accruing computational savings because the number of control points is dramatically smaller than the original grid points – for a faster and more accurate technique, see [546]. The vector field describing the coordinate warp was then improved using a simple but ad-hoc heuristic iterative improvement technique. The computed vector field was then used to warp the given segmentation in the source image on to the target image. The governing equations in this iterative improvement are almost identical to the non-rigid deformation estimation equations reported in Dawant et.al., [148] which are in turn identical to the governing equation of non-rigid registration in Thirion [521]. Another popular technique based on the fluid flow non-rigid registration model introduced in Christensen et.al., [120] has been used quite widely in achieving atlas-based segmentation and we refer the reader to [564, 259, 209]. All these methods do not achieve registration and segmentation simultaneously in a unified mathematical framework.

Recently, Mutual Information (MI) [555, 131] based non-rigid registration methods have been used in [451, 200] for constructing statistical deformation models. These models are then used in constructing atlases along with information about the variability in the population used for constructing the atlas. Once

again, these methods do not use a unified framework for registration and segmentation but instead they first estimate the deformation field and then apply it to the atlas/mask to detect the corresponding shape in the target/subject image.

Approaches that use prior models to achieve the task of segmentation within the level-set curve evolution framework have been quite popular in the recent past. Various techniques have been developed that incorporate shape priors into the active contour-based segmentation techniques that use gradient as well as region-based information (e.g. see [133, 563, 135, 134, 492, 602, 502]). In Leventon et.al., [310], a principal component analysis (PCA) technique was used to build a statistical shape prior from a set of training curves. A term depending on this learnt shape was added at each iteration of the level-set surface evolution to force the geometric active contour towards high image gradients and a maximum posteriori estimate of shape and pose. Note that the formulation in [310] was not posed in a variational framework and hence was not the result of any minimization principle. In Chen et.al., [111] the segmentation and the registration, which maps the evolving shape to the shape prior, were obtained by minimizing an energy function containing a “shape energy” and the energy of geometric active contour. There are numerous reports in literature on ways to incorporate statistical shape models into region-based segmentation schemes including Cremers et.al., [142], Tsai et.al., [526], Rousson et.al., [444] and Chen et al. [112]. In [142], an energy functional is minimized that includes both a ”shape energy” term corresponding to a Gaussian probability as well as a ”Mumford-Shah energy” [388] term. In [526] a parametric shape model similar to the one described by Leventon et.al., [310] was built, where parameters are calculated to minimize a region based objective function which provides the segmentation. The shape model in Rousson et.al., [444] was formed using a level set representation and the variability of the shape. It was designed to maximize a log-likelihood function represented by a Gaussian density function whose mean corresponds to the level set representation of the shape and the variance refers to the variation of the aligned samples. The shape model was then combined into a geodesic active region model [409] in a variational framework. Note that shape priors do not enter the active contour-based segmentation process via a registration term and hence are not achieving simultaneous registration and segmentation. In [112], a modified Mumford-Shah [388] energy function is minimized that incorporates shape information – via a parameterized registration term – and is represented in a variational level set formulation. In the following, we will present an overview of the methods in [111, 112] and then describe them in detail subsequently.

14.1.2 Overview of Our Algorithms

In this chapter, we present two techniques for simultaneously achieving registration and segmentation. The first technique is a direct PDE-based formulation and the second is a variational formulation leading to a PDE, whose solution yields the desired goal of registration and segmentation.

The first approach is very simple and was originally described in [547, 548]. This formulation consists of coupled PDE model for achieving the registration and segmentation. The basic idea in this technique is to let the source image evolve by letting its level-sets move along their respective normals with a speed that is proportional to the difference between the target and the evolving source image. The evolution will automatically stop when the evolved source becomes the target image. The resulting partial differential equation is a nonlinear hyperbolic equation and can be numerically solved using upwind finite difference techniques [401, 311]. We establish the existence and uniqueness of the solution to this nonlinear hyperbolic equation in a Sobolev space (Note: not using viscosity methods). The aforementioned evolution will only provide a mapping from the source image intensity function to the target image intensity. It does not however explicitly provide the geometric (coordinate) deformation between the source and target images. In order to obtain this deformation defined by a vector field, we develop another nonlinear PDE which is in a vector form and is coupled to the first one and whose solution describes the coordinate i.e., geometric transformation between the source and the target images. The coupling however is only one way i.e., from the vector form to the intensity evolution. This allows us to determine the vector field transformation at any stage of the intensity evolution, more on this subsequently. By restricting this vector field transformation to a pre-segmented shape in the source/atlas image, we can easily evolve this segmentation to the target image and thereby find the corresponding shape in the target image. We have successfully applied these evolution models to 2D and 3D image data sets yielding fast and accurate registrations. Examples depicting the performance of our algorithm are presented for real data sets in section 14.4.

The second approach primarily involves a variational formulation of finding the desired shape boundaries in an image using shape priors which would constraint the detection process via an explicit registration term between the prior shape and the shape being sought. The registrations considered include rigid motions with a uniform scale. The desired shape boundaries are referred to here as a segmentation. A level-set representation is used for the desired shape boundary and one variational form using image gradient information and another using region-based measures are presented. The key contribution here is the process by which the shape priors augment the gradient-based and the region-based active contour model, which is achieved via the incorporation of a registration term in a variational framework. This leads to simultaneous registration and segmentation. Examples depicting the performance of results from both these formulations for the case of synthetic and real data are depicted in section 14.4.

14.2 The PDE-based Approach

If we consider the image registration problem from the point of curve/surface evolution, registering two given intensity images can be intuitively thought of as

determining ways to evolve the level-sets of the intensity function of one image (say the source image) into level-sets of the intensity function of another image (call it the target image). So, given two images $I_1(X)$ and $I_2(X)$, we want $I_1(X)$ to evolve into $I_2(X)$. Intuitively, it seems reasonable to evolve $I_1(X)$ by letting its level-sets evolve along their normals i.e., by letting $I_1(X)$ evolve along its gradient until it becomes the target image. The evolution can thus be written down as

$$I_t(X, t) = S \|\nabla I(X, t)\| \text{ with } I(X, 0) = I_1(X) \quad (14.1)$$

where S is the speed term. From curve evolution theory, we know that (14.1) will evolve the level-sets of the image in the direction ∇I . Now, we need to determine an appropriate speed term S . Since this evolution should only stop when image $I(X)$ changes from $I_1(X)$ to $I_2(X)$, we need to include this stopping mechanism in the speed term. Therefore the natural choice for it will be $S = I_2(X) - I(X, t)$. Hence, equation (14.1) becomes

$$\begin{aligned} I_t(X, t) &= (I_2(X) - I(X, t)) \|\nabla I(X, t)\| \\ \text{with } I(X, 0) &= I_1(X) \end{aligned} \quad (14.2)$$

A similar equation was derived in Bertalmio *et al.* [41] in the context of tracking 2D shapes in a time sequence of 2D images. Their formulation however does not provide a mechanism to explicitly retrieve the geometric transformation between the two given image data sets. In some applications, it is quite useful to have this deformation field explicitly. For e.g., in automatic shape analysis of cortical and sub-cortical structures, the displacement field can be used to characterize shape differences corresponding structures in the left and right halves of the brain in order to pin down asymmetries.

Note that (14.2) does not give the geometric transformation/motion between the data sets explicitly which is not necessary in some applications such as image morphing. In all the registration applications however, we need to determine this geometric transformation explicitly between the two images and this can be achieved via the use of the following equation which is coupled to the equation (14.2):

$$\begin{aligned} \mathbf{V}_t &= [I(X) - I_1(\mathbf{V}(X))] \frac{\nabla I_1(\mathbf{V}(X))}{\|\nabla I_1(\mathbf{V}(X))\|} \\ \text{with } \mathbf{V}(X, 0) &= \mathbf{0} \end{aligned} \quad (14.3)$$

where I evolves according to equation 14.2, $\mathbf{V} = (u, v)^T$ is the displacement vector at X and the operation $\mathbf{V}(X) = (x - u, y - v)^T$. In the 3D case, $\mathbf{V} = (u, v, w)^T$ and $\mathbf{V}(X) = (x - u, y - v, z - w)^T$. Note that the transformation V is centered in the image I_2 i.e., $I_1(x - u, y - v) = I_2(x, y)$. We can equivalently center the transformation in I_1 and in this case, we would have $I_1(x, y) = I_2(x + u, y + v)$. Equation (14.3) can be used to evolve feature maps including a segmentation by imposing the condition that $X \in C$, where C is the given segmentation. Note that at any point in time t , during the evolution of the

intensity function, I using equation (14.2), one may use equation (14.3) to determine the vector field transformation and hence the desired segmentation at this intermediate stage. Examples of segmenting the 3D hippocampal shape from a brain MRI are shown subsequently using this technique.

In the following we present a theorem which guarantees the existence and uniqueness of the solution to the PDE (14.2). The existence and uniqueness issues for Equation (14.3) are much harder to answer and will be the focus of our future efforts. To prove the existence and uniqueness result for equation (14.2), we will rewrite the equation here and where appropriate, for brevity, drop the coordinates X from the notation.

$$I_t = (I_2 - I)\|\nabla I\|, \quad \text{in } \Omega \times \mathbb{R}_+, \quad (14.4)$$

$$I(x, 0) = I_1, \quad \text{on } \Omega, \quad I(x, t) = 0, \quad \text{on } \partial\Omega \times \mathbb{R}_+, \quad (14.5)$$

where Ω is the domain of definition of the image, $\partial\Omega$ denotes the boundary of domain Ω and $C(\Omega^T)$ is the set of all continuous functions which map the domain Ω^T to \mathbb{R} . Let $\Omega^T = \Omega \times [0, T]$. We assume that I_1, I_2 are Lipschitz continuous functions on Ω . Let $W^{1,\infty}(\Omega) = \{u | u, Du \in L^\infty(\Omega)\}$ where u is a locally summable function: $\Omega \rightarrow \mathbb{R}$ and Du is 1st-weak partial derivative of u [181].

THEOREM 1 *Assume I_1 and I_2 are two functions in $W^{1,\infty}(\Omega)$. Then, for any $T > 0$ the problem (14.4)-(14.5) admits a unique solution I on Ω^T such that $I \in C(\Omega^T) \cap L^\infty(0, T; W^{1,\infty}(\Omega))$, $I_t \in L^2(\Omega^T)$. Moreover, if u_1 and u_2 are the solutions of (14.4) corresponding to the initial data F_1 and F_2 , respectively, then*

$$\sup_{0 \leq t \leq T} \|u_1(\cdot, t) - u_2(\cdot, t)\|_{L^\infty(\Omega)} \leq 2\|F_1 - F_2\|_{L^\infty(\Omega)}. \quad (14.6)$$

As a consequence of this theorem, for any given initial condition, the mapping between the two intensity functions I_1 and I_2 is uniquely determined. A proof of this theorem is given in the appendix. Note that I_1 and I_2 are (assumed) Lipschitz continuous. The solution to (14.4) and (14.5) is Lipschitz continuous as well. Also, at all time t , during the evolution, the evolving model remains in $W^{1,\infty}(\Omega)$.

The numerical solution to equations (14.2) and (14.3) is achieved efficiently using a pyramid (coarse-to-fine) implementation of an adaptive time step integration scheme that satisfies the CFL (Courant-Friedrichs-Levy) condition [311]. This leads to a stable and efficient numerical algorithm. The intensity morphing PDE (14.2) was implemented using upwind finite difference techniques and the PDE characterizing the vector field transformation (14.3) was implemented using min-mod finite difference schemes. For further details on the numerical techniques, we refer the reader to [548].

14.3 The Variational Approach

In this section, we present an alternative approach involving the minimization of a variational principle for the simultaneous registration and segmentation problem

discussed earlier. The key contribution here is the mechanism for incorporating priors (via model/prior-based registration). The minimization leads to a PDE which is solved numerically. In this section, we will describe the basic ideas and approaches in [111] and [112] that incorporate shape information into edge or region based active contour for simultaneous segmentation and registration.

14.3.1 Geometric Active Contour with a Shape Prior

In the context of image segmentation, geometric active contours based on geometric flows represented in a level-set framework were introduced to the vision community by Malladi et.al., [329, 330, 331] and [81] and since have had numerous variants that have been quite successfully employed in the computer vision, medical imaging and other communities. In the following, we focus on the description of a variant that allows for incorporation of a level-set representation of shape priors and pose the segmentation and registration problem in a variational framework.

Let $C(p) = (x(p), y(p))$ ($p \in [0, 1]$) be a differentiable parameterized curve in an image I . A geodesic active contour [85, 270] minimizes the energy function:

$$\min_C E(C) = \min_C \int_0^1 g(|\nabla I|)(C(p))|C'(p)|dp,$$

where $g(|\nabla I|)$ can be chosen as

$$g(|\nabla I|) = \frac{1}{1 + \beta|\nabla G_\sigma * I|^2},$$

with a parameter $\beta > 0$, and $G_\sigma(x) = \frac{1}{\sigma}e^{-\frac{|x|}{4\sigma^2}}$. The active contour stops when its trace is at points of high image gradients. Therefore, if the boundary has segments which have low gradients, it will often "leak" through such "gaps" in the boundary. One solution to the problem is to incorporate (into the active contour) a shape prior describing the expected overall boundary of the shape of interest. Then, at each iteration of the evolving active contour, the active contour can be compared with the expected shape in order to fill any of the aforementioned "leaks".

The key idea in our algorithm described in detail in [111] is the creation of a shape-based term in the energy of the active contour. By minimizing this new energy the evolution of the active contour stops when the active contour arrives at high image gradients and achieves a shape similar to the shape prior. To begin the mathematical description, we first specify that two contours C_1 and C_2 have the same shape, if there exist a scale μ , a rotation matrix R , and a translation vector T such that C_1 matches with $C_2 = \mu RC_1 + T$. Let C^* be a curve representing the *shape prior*, and $d(x, y) = d(C^*, (x, y))$ be the distance of the point (x, y) from C^* . To find a curve C that captures higher gradients and the shape prior C^* , we introduce the energy function $E(C, \mu, R, T)$,

$$E(C, \mu, R, T) = \int_0^1 \{g(|\nabla I|(C(p))) + \frac{\lambda}{2}d^2(\mu RC(p) + T)\}|C'(p)|dp, \quad (14.7)$$

where $\lambda > 0$ is a parameter. The first term in the energy function related to a measure of high image gradients along the trace of the curve and the second term measures the closeness to the shape prior. Using a level-set representation of the two contours can avoid specifying the explicit correspondence between the two contours. The specific level-set form we use is a distance function for each of the contours. The curve C and the transformation parameters μ, R and T evolve to minimize $E(C, \mu, R, T)$. In problems of curve evolution, the level set methods have been used extensively, because they allow for cusps, corners, and automatic changes in topology [401, 607, 101]. Due to these positive features of a level-set representation, we now pose the above formulation in a level-set framework.

Let the contour C be the zero level set of a Lipschitz function u such that $\{x|u(x) > 0\}$ is the set inside C . Let $H(z)$ be the Heaviside function: $H(z) = 1$ if $z \geq 0$, and $H(z) = 0$ if $z < 0$. $\delta = H'(z)$ (in the sense of distribution) is the Dirac measure. Then, the length of the zero level set of u in the conformal metric $ds = g|C'(p)|dp$ can be computed by $\int_{\Omega} g|\nabla H(u)| = \int_{\Omega} \delta(u)g|\nabla u|$. The similarity of the shapes between the zero level set of u and C^* can be evaluated by $\int_{\Omega} \delta(u)d^2(\mu Rx + T)dx$, where the distance function d is the same as in equation (14.7). Therefore, the variational level set formulation of our approach is given by

$$\min_{u, \mu, R, T} \int_{\Omega} \delta(u)\{g(|\nabla I|) + \frac{\lambda}{2}d^2(\mu Rx + T)\}|\nabla u|. \quad (14.8)$$

The evolution equations related to the Euler-Lagrange equations for this problem (14.8) are

$$\frac{\partial u}{\partial t} = \delta(u) \operatorname{div}\{(g + \frac{\lambda}{2}d^2)\frac{\nabla u}{|\nabla u|}\}, \quad x \in \Omega, \quad t > 0, \quad (14.9)$$

$$\frac{\partial u}{\partial n} = 0, \quad x \in \partial\Omega, \quad t > 0, \quad u(x, 0) = u_0(x), \quad x \in \Omega, \quad (14.10)$$

$$\frac{\partial \mu}{\partial t} = -\lambda \int_{\Omega} \delta(u)d\nabla d \cdot (Rx)|\nabla u|dx, \quad t > 0, \quad \mu(0) = \mu_0, \quad (14.11)$$

$$\frac{\partial \theta}{\partial t} = -\lambda \int_{\Omega} \delta(u)\mu d\nabla d \cdot (\frac{dR}{d\theta}x)|\nabla u|dx, \quad t > 0, \quad \theta(0) = \theta_0, \quad (14.12)$$

$$\frac{\partial T}{\partial t} = -\lambda \int_{\Omega} \delta(u)d\nabla d|\nabla u|dx, \quad t > 0, \quad T(0) = T_0, \quad (14.13)$$

where R is the rotation matrix in terms of the angle θ , and the function d is evaluated at $\mu Rx + T$.

Normally, edge based segmentation is sensitive to the initialization of the active contour model in spite of the shape priors used. The active contour may stop at false edges with high gradients, if the initial contour is not close to the shape being segmented. Region based segmentation schemes are less sensitive to this initial choice. In particular, in the cases where edge information is insufficient and/or unreliable, region based segmentation is a good alternative solution. In the following we present a region-based method which will replace the image gradient-based term.

14.3.2 Mumford-Shah Model with a Shape Prior

In the celebrated paper [388], Mumford and Shah proposed to obtain a segmentation f of a given image I (in R^n) as a minimum of the following functional:

$$E(S, f) = \alpha \int_{\Omega} (f - I)^2 dx + \beta \int_{\Omega \setminus S} |\nabla f| dx + H^{n-1}(S), \quad (14.14)$$

where $H^{n-1}(S)$ stands for the $(n-1)$ -dimensional Hausdorff measure, $\alpha > 0$ and $\beta > 0$ are parameters. However, in certain medical imaging modalities, due to the inhomogeneity of intensities, region based methods do not yield the desired segmentation either. In [112], a method for incorporating shape priors into the Mumford-Shah's scheme was developed. This was done in an active contour framework such that the active contour shape was compared with the expected shape and a correction of its position using region information was performed. The approach in [112] involves considering S as the boundary C of the feature to be detected, and replacing the term $H^{n-1}(S)$ in (14.14) by a weighted length of C , where the weight function depends on the shape prior. A variational level set formulation of the energy functional was proposed as follows:

Let the contour C be the zero level set of a Lipschitz function u such that $\{x|u(x) > 0\}$ is the set inside C . Let $H(z)$ be the Heaviside function: $H(z) = 1$ if $z \geq 0$, and $H(z) = 0$ if $z < 0$. $\delta = H'(z)$ (in the sense of distribution) is the Dirac measure. Let f^+ and f^- be the estimates of I on $\{x|u(x) > 0\}$ and $\{x|u(x) < 0\}$, respectively. To find the feature segmentation C and estimate the transformation (μ, R, T) , that aligns C to the prior shape C^* , the following function is minimized:

$$\begin{aligned} E(u, f^+, f^-, \mu, R, T) = & \alpha \int_{\Omega} H(u)(f^+ - I)^2 dx + \alpha \int_{\Omega} (1 - H(u))(f^- - I)^2 dx \\ & + \beta \int_{\Omega} H(u)|\nabla f^+|^2 dx + \beta \int_{\Omega} (1 - H(u))|\nabla f^-|^2 dx \\ & + \int_{\Omega} \delta(u)d^2(\mu Rx + T)|\nabla u|dx, \end{aligned} \quad (14.15)$$

w.r.t. u, f^+, f^-, μ, R, T , where $d(x)$ is the same as in (14.7).

This problem is solved by finding the steady state solution of the following system:

$$\frac{\partial u}{\partial t} = \delta(u) \left\{ \operatorname{div} \left(d^2 \frac{\nabla u}{|\nabla u|} \right) + \right.$$

$$\left. \alpha[(f^+ - I)^2 - (f^- - I)^2] + \beta[|\nabla f^+|^2 - |\nabla f^-|^2] \right\}, \quad (14.16)$$

$$\frac{\partial u}{\partial n} = 0, \quad x \in \partial\Omega, \quad t > 0, \quad u(x, 0) = u_0(x), \quad x \in \Omega, \quad (14.17)$$

$$\Delta f^+ = \alpha/\beta(f^+ - I), \quad \text{on } u > 0, \quad \frac{\partial f^+}{\partial n} = 0, \quad \text{on } u = 0, \quad (14.18)$$

$$\Delta f^- = \alpha/\beta(f^- - I), \quad \text{on } u < 0, \quad \frac{\partial f^-}{\partial n} = 0, \quad \text{on } u = 0, \quad (14.19)$$

$$\frac{\partial \mu}{\partial t} = -2 \int_{\Omega} \delta(u) d\nabla d \cdot (Rx) |\nabla u| dx, \quad t > 0, \quad \mu(0) = \mu_0, \quad (14.20)$$

$$\frac{\partial \theta}{\partial t} = -2 \int_{\Omega} \delta(u) \mu d\nabla d \cdot \left(\frac{dR}{d\theta}x\right) |\nabla u| dx, \quad t > 0, \quad \theta(0) = \theta_0, \quad (14.21)$$

$$\frac{\partial T}{\partial t} = -2 \int_{\Omega} \delta(u) d\nabla d |\nabla u| dx, \quad t > 0, \quad T(0) = T_0. \quad (14.22)$$

The numerical algorithms for obtaining the steady state solutions for the systems (14.9)-(14.13), and (14.16)-(14.22) follow from the work of Chan and Vesse [101, 102]. This includes regularizing the Heaviside function H and its weak derivative δ , the re-initialization of u^n to the signed distance function of the front (the zero level set of u^n), and the proper extensions of $(f^+)^n$ and $(f^-)^n$ across the interface $u^n = 0$ to compute their derivatives. All the computations are performed in the narrow band around the interface. For more details on the numerical methods used, we refer the reader to [111, 112].

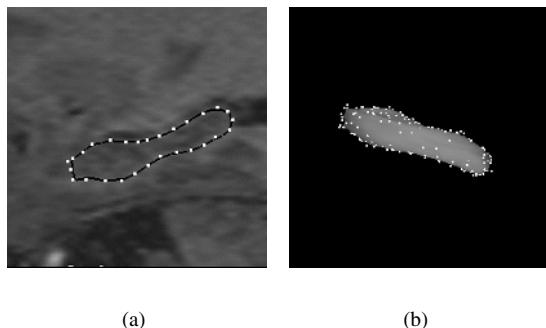
14.4 Experimental Results and Applications

In this section, we present the experimental results of both our approaches to the simultaneous registration and segmentation problem. The order of presentation is as in the description of the methods.

14.4.1 Results from the PDE-based Approach

To segment the desired shape from the image of an unknown subject, we apply our PDE-based simultaneous registration and segmentation scheme described earlier in section 14.2. A known segmentation called the atlas is used to achieve this task. The vector field corresponding to the transformation between the atlas image and the subject image obtained using equation (14.3) is applied to the desired atlas shape. This is done by restricting the transformation field to the region of interest, in this case, hippocampal shape in the human brain.

Manual segmentation to create an atlas shape is extremely time-consuming and labor intensive. Instead, we use a deformable pedal surface (see Vemuri *et al.* [545]) to semi-automatically extract the anatomical shape of interest, a hippocampus in this study, for the construction of an atlas. Figure (14.1) depicts the fitting results of the deformable pedal surface. The left image is one slice of the MR image. The superimposed curve (in black) is the intersection between the surface



(a)

(b)

Figure 14.1. Results of fitting a deformable pedal surface: (a) one slice of image superimposed by the model curve and points; (b) 3D view of deformable pedal model and data points

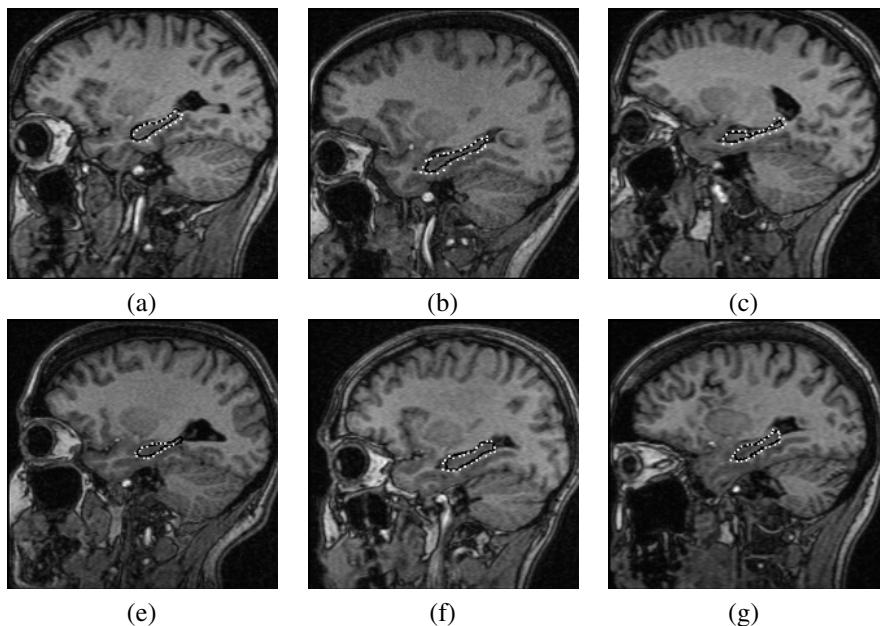


Figure 14.2. A slice from each MRI (being segmented) superimposed with the corresponding slice (in black) from the automatic 3D hippocampal segmentation, validated by the boundary (bright) points placed by an expert.

of the deformable pedal surface model and the slice. The points (bright dots) are placed by an expert Neuro-scientist to define the boundary of hippocampus. The image on the right depicts these 3D points and the fitted deformable pedal surface model in 3D. After fitting, we use the deformable pedal surface model to construct a binary volume enclosed by the hippocampal surface, i.e., one for hippocampus and zero elsewhere. During the estimation of the transformation between atlas and the unknown subject image, the transformation is simultaneously applied to this binary shape volume to yielding a warped segmentation representing the desired segmented hippocampus in the image of the unknown subject.

Six subject images were automatically segmented for the hippocampal shape in our study. Figure 14.2 depicts the contours (slices) from our 3D segmentation results. The bright points were placed by an expert Neuro-scientist to validate our results. Evidently, the transformed hippocampal surfaces fit the (bright) points quite accurately. For a quantitative validation of our results, we have used the ratio of the left to right hippocampal volumes, an important measure in clinical diagnosis for subjects who are afflicted with epilepsy and need possible surgical treatment [213, 307]. These ratios in all the six experiments are very close to the ground truth obtained via manual measurements. For more details on the validation results, we refer the reader to [548].

14.4.2 Results from the Variational Approach

In this section, we present the results of applying the models (14.7), (14.8), and (14.15) – obtained using the two variational approaches described in section 14.3 – to functional MR brain images and cardiac ultrasound images.

We present a synthetic data example demonstrating the power of incorporating prior shape information into the Mumford-Shah model for image segmentation implemented using an active contour model in a level-set framework. Synthetic data examples for the geometric active contour model augmented with shape priors described earlier maybe found in [111]. The purpose of the first experiment is to detect a disk in a binary image partially occluded by a square (figure 14.3b) by using model (14.15). The shape prior C^* is a disk in figure 14.3a. Evolving an initial curve C in figure 14.3b according to the system of equations (14.16)-(14.22) we get the stationary contour C in figure 14.3c, and the transformation parameters μ , θ , and T . The contour $\mu RC(p) + T$ is shown in figure 14.3d. Figures 14.3a, 14.3c, and 14.3d show that the solution C separates two homogeneous region and fills up the missing portion of the boundary of the disc by forming a shape similar to the prior C^* .

The aim of the second experiment is to show that our active contour not only provides an appropriate segmentation, but also provides accurate estimates of the registration parameters. These estimates can be used to align functional MR images. Very briefly, functional MR images are time series images, that permit visualization of local changes in cortical blood volume, flow, and oxygenation. These changes are correlated with mental activity. Typically a subject is exposed to baseline and activation conditions while his/her brain is being scanned in the

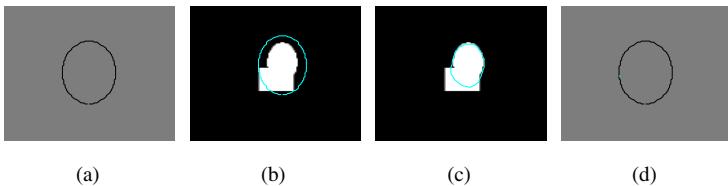


Figure 14.3. (a) Prior shape C^* , (b) synthetic image superposed with the initialized active contour (an ellipse), (c) the final stationary contour C , (c) The contour $\mu RC + T$.

MR machine. A spin sequence is used that enables the visualization of blood volume and oxygenation. The images from baseline are compared with images from activation to obtain those pixels which are statistically different. These pixels are assumed to indicate locations of brain activity. Due to the long scanning time, some head movement from the subject is unavoidable. Head movement introduces a mis-registration between the images which in turn causes spurious signal changes. Typical T_2^* weighted brain images show that the spurious signal changes due to motion are of the same order as the signal changes from the task. Therefore, small motions impact the fMRI time series adversely. Proper image registration (realignment of the images) is needed to minimize the effect of motion on the fMRI signal.

It has been pointed out in [47, 113] that intensity based image registration methods do not provide good results for aligning these time series images. This is because of the complex nature of intensity variations, such as different distribution of signal intensity during a task activation and a rest scan, and nonuniform intensity variation in an imaging slice due to motion and temporal variation of the magnetic field homogeneity. Feature-based methods are not adversely affected by such changes. Feature based methods first find the important features in both images, and then align the images by aligning the features. The shape prior based active contours 14.8 (or 14.7) and 14.15 offer reliable feature extraction (because prior information is used) as well as reliable estimates of translation, rotation and scale which can be used to align the images. To apply our method to fMRI, we used a segmented corpus callosum in a high resolution MR image as the shape prior C^* , and used our active contour to find the boundary of the poorly determined corpus callosum in each of the time series images. The spatial transform that maps the segmented contour to the prior C^* is used to register the time series images.

Figure 14.4a shows the high resolution image and the outlined corpus callosum that was used as the prior shape. Figure 14.4b is an image in the fMRI time series. It has a much lower resolution than the image in figure 14.4a. The active contour was initialized as an ellipse as shown in figure 14.4b. Evolving the initial contour according to the system of equations (14.16)-(14.22) we get the stationary contour C shown in figure 14.4c, and the transformation parameters μ , θ , T that map C

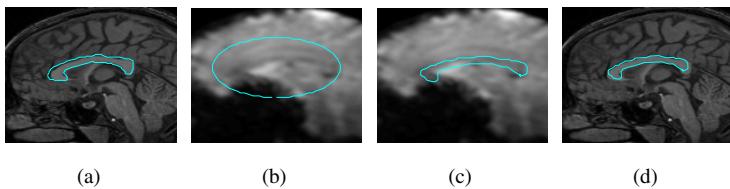


Figure 14.4. (a) The corpus callosum from a high resolution image, (b) fMRI images showing the initialized active contour (the ellipse), (c) The final active contour, (d) The final contours superimposed on the high resolution images.

to the prior curve C^* . To validate the segmentation, we plotted $\mu RC + T$ back on the high resolution image from which the prior was constructed. This is shown in figures 14.4d. By comparing 14.4d with 14.4a we can see that the contour C does segment the corpus callosum and remains faithful to the prior shape.

The aim of the last experiment is to segment the epicardium (the outer boundary of the myocardium surrounding the left ventricle) in a four-chamber image of the heart (see figure 14.5b for a typical image). The epicardium is not completely imaged in the image, and our task is to find and complete the boundary using a shape prior.

To create the prior shape, epicardial boundaries were outlined in 112 patients by an expert echo-cardiographer. An SOM (self-organizing maps) algorithm was used to find clusters. Let the 112 contours be denoted by X_i ($i = 1, \dots, 112$), and group them into three clusters using the following procedure: Take three arbitrary contours as the initial contours $m_j(0)$ ($j = 1, 2, 3$). At the $(t+1)$ th iteration, randomly select a contour denoted by $X(t+1)$ from X_i ($i = 1, \dots, 112$). Compare the disparity in shape between $X(t+1)$ and each of $m_j(t)$ ($j = 1, 2, 3$). To do this comparison we first get the alignment parameters (μ_j, R_j, T_j) by

$$\min_{\mu_j, R_j, T_j} A_j = \min_{\mu_j, R_j, T_j} \text{area of } (A \cup B_j - A \cap B_j), \quad (14.23)$$

where A and B_j denote the interior regions of the curves $X(t+1)$ and $m_j(t)$ ($j = 1, 2, 3$), respectively. Let $\tilde{X}_j(t+1) = \mu_j R_j X(t+1) + T_j$. Suppose A_1 is the smallest number in A_j ($j = 1, 2, 3$). We keep $m_2(t)$ and $m_3(t)$ unchanged, and update $m_1(t)$ by

$$m_1(t+1) = m_1(t) + \alpha(t)[\tilde{X}_1(t+1) - m_1(t)],$$

where $\alpha(t)$ is a smooth function of t , and decreases to zero as $t \rightarrow \infty$. After large number of iteration, say N , three *average* shapes are generated by $m_j(N)$ ($j = 1, 2, 3$). Then, distinct clusters are formed by the groups of curves that are closest to each of the three *average* shapes. The closeness is measured by (14.23).

Figure 14.5a shows one of the clusters of 79 contours and its average contour. Our goal was to segment the epicardium in figure 14.5b, using the average contour as the prior shape. The contour shown in figure 14.5b is used as an initial

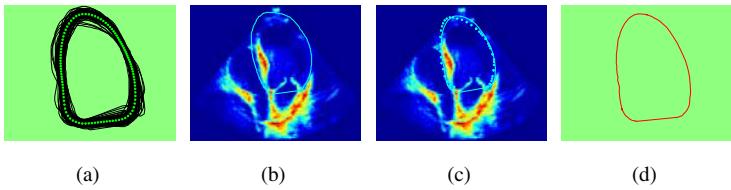


Figure 14.5. (a) A cluster of 79 curves and their average shape C^* . (b) The initial contour in the ultrasound image. (c) The active contour at its stationary point (solid curve) and the expert’s epicardium (dotted curve), (d) The contour $\mu RC + T$, where C , μ , R , and T are the transform parameters.

contour. This contour is allowed to evolve according to the flow associated with the Euler-Lagrange equations of the minimization problem (14.7), and it finally stopped at the location of the solid contour in 14.5c. To validate this result we had the expert manually segment the epicardium. The dotted contour in 14.5c is the expert’s epicardium. As evident, our segmentation is visually close to the expert’s. In figure 14.5d we present the contour $\mu RC + T$ for inspection and as evident, it is similar to the average shape C^* in figure 14.5a.

Appendix A

In this appendix, we present the proof of theorem 1. Restating the theorem here for convenience,

THEOREM 1 *Assume I_1 and I_2 are two functions in $W^{1,\infty}(\Omega)$. Then, for any $T > 0$ the problem (14.4)-(14.5) admits a unique solution I on Ω^T such that $I \in C(\Omega^T) \cap L^\infty(0, T; W^{1,\infty}(\Omega))$, $I_t \in L^2(\Omega^T)$. Moreover, if u_1 and u_2 are the solutions of (14.4) corresponding to the initial data F_1 and F_2 , respectively, then*

$$\sup_{0 \leq t \leq T} \|u_1(\cdot, t) - u_2(\cdot, t)\|_{L^\infty(\Omega)} \leq 2\|F_1 - F_2\|_{L^\infty(\Omega)}. \quad (14.24)$$

Proof: Consider the following approximating problem:

$$\frac{\partial I^\varepsilon}{\partial t} = \varepsilon \Delta I^\varepsilon + \sqrt{|\nabla I^\varepsilon|^2 + \varepsilon^2}(I^\varepsilon - I_2), \quad \text{in } \Omega \times R_+, \quad (14.25)$$

with the initial and boundary conditions as in (14.5).

By the theory of quasi-linear uniformly parabolic equations [300] the problem (14.5) and (14.25) admits a smooth solution $I^\varepsilon \in C^\infty(R^n \times R_+)$.

Differentiating (14.25) with respect to x_k , then multiplying the resulting equation by $2I_{x_k}^\varepsilon$ and summing with respect to k , we get (using the notation $b(p) =$

$$\begin{aligned}
& \sqrt{p^2 + \varepsilon^2}) \\
& \frac{\partial |\nabla I^\varepsilon|^2}{\partial t} - \varepsilon \Delta |\nabla I^\varepsilon|^2 \\
& + 2\varepsilon |\nabla^2 I^\varepsilon|^2 + 2b(\nabla I^\varepsilon) |\nabla I^\varepsilon|^2 - \nabla b(\nabla I^\varepsilon) \cdot \nabla (|\nabla I^\varepsilon|^2)(I^\varepsilon - I_2) \\
& = -2b(\nabla I^\varepsilon) (\nabla I^\varepsilon \cdot \nabla I_2) \leq C(I_2) |\nabla I^\varepsilon|^2.
\end{aligned} \tag{14.26}$$

Applying the maximum principle [300] to (14.26) yields for all $T > 0$ and $t \in [0, T]$

$$\|\nabla I^\varepsilon(\cdot, t)\|_{L^\infty(\Omega)} \leq C e^{cT} \|\nabla I_1\|_{L^\infty(\Omega)} \leq C(I_1, I_2, T). \tag{14.27}$$

Moreover, from (14.27) and the argument developed in [10], we have

$$|I^\varepsilon(x, s) - I^\varepsilon(x, t)| \leq C(I_1, I_2, T) |t - s|^{\frac{1}{2}}, \quad \forall x \in \Omega \text{ and } \forall s, t \in [0, T]. \tag{14.28}$$

Next, multiplying (14.25) by $2I_t^\varepsilon$, and integrating over Ω^T , we get

$$\begin{aligned}
& \int_{\Omega^T} |I_t^\varepsilon|^2 dx dt + \varepsilon \int_{\Omega} |\nabla I^\varepsilon|^2(x, T) dx \\
& = \varepsilon \int_{\Omega} |\nabla I_1|^2 dx + \int_{\Omega} b(\nabla I^\varepsilon)(I_2 - I^\varepsilon) dx.
\end{aligned} \tag{14.29}$$

Combining this with (14.27)-(14.28), one finds

$$\|I_t^\varepsilon\|_{L^2(\Omega^T)} \leq C(I_1, I_2, T, \Omega). \tag{14.30}$$

From (14.27)-(14.30), there exist a subsequence I^{ε_k} of I^ε , and a function $I \in C(\Omega^T) \cap L^\infty(0, T; W^{1,\infty}(\Omega))$ with $I_t \in L^2(\Omega^T)$, such that as $\varepsilon_k \rightarrow 0$,

$$I^{\varepsilon_k} \rightarrow I, \quad \text{uniformly in } \Omega^T, \quad \text{and weakly* in } L^\infty(0, T; W^{1,\infty}(\Omega)), \tag{14.31}$$

$$I_t^{\varepsilon_k} \rightarrow I_t, \quad \text{weakly in } L^2(\Omega^T). \tag{14.32}$$

Applying the limit $\varepsilon_k \rightarrow 0$ to the weak form of (14.25), it is not difficult to conclude from (14.31)-(14.32) that I is a solution of (14.4)-(14.5), and (14.4) holds on $L^2(\Omega^T)$. This completes the existence proof. ■

Appendix B

There is much work done on the discussion of the existence of minima for Mumford-Shah functional, for instance, see [145, 149, 287, 382], and references therein. Here, we present the existence of a weak solution to the minimization problem (14.8). The same argument can also be applied to equation (14.7). As suggested in [101] we may rewrite (14.8) in the form

$$\min_{\chi_E, \mu, \theta, T} \int_{\Omega} \{g(|\nabla I|(x) + \frac{\lambda}{2} d^2(\mu Rx + T))|D\chi_E|, \tag{14.33}$$

where R is the rotation matrix in terms of an angle θ , and χ_E is the characteristic function of $E = \{x \in \Omega | u(x) \geq 0\}$. This minimization is over all the characteristic functions of E in $BV(\Omega)$ (note that the set E varies as u evolves).

The minimization problem (14.33) involves a weighted total variation norm for the functions with finite perimeters. It also minimizes more than one argument. To study the existence for this problem, it is necessary to introduce the concept of weighted total variation norms for functions of bounded variation. Let us begin with recalling the definitions of functions with bounded variation [214].

DEFINITION: 1 Let $\Omega \subset R^n$ be an open set and let $f \in L^1(\Omega)$. Define

$$\int_{\Omega} |\nabla f| =: \sup_{\phi \in \Phi} \left\{ \int_{\Omega} f(x) \operatorname{div} \phi(x) dx \right\},$$

where

$$\Phi =: \{\phi \in C_0^1(\Omega, R^n) | |\phi(x)| \leq 1, \text{ on } \Omega\}.$$

DEFINITION: 2 A function $f \in L^1(\Omega)$ is said to have bounded variation in Ω , if $\int_{\Omega} |\nabla f| < \infty$. We define $BV(\Omega)$ as the space of all functions in $L^1(\Omega)$ with bounded variation. For a function $f \in BV(\Omega)$, we define

$$\|f\|_{BV(\Omega)} = \|f\|_{L^1(\Omega)} + \int_{\Omega} |\nabla f|.$$

DEFINITION: 3 A measurable subset E of R^n has finite perimeter in Ω , if the characteristic function $\chi_E \in BV(\Omega)$.

It has been known that if $f \in BV(\Omega)$, then for any $t \in R$, the level set $E_t = \{x \in \Omega | f(x) > t\}$ has finite perimeter, i.e. $\chi_{E_t} \in BV(\Omega)$. Next we define the weighted total variation norm with the weight function $\alpha(x)$.

DEFINITION: 4 Let $\Omega \subset R^n$ be an open set. Let also $f \in L^1(\Omega)$ and $\alpha(x)$ be positive valued continuous and bounded functions on Ω . The weighted total variation norm of f with the weight function $\alpha(x)$, denoted by $\int_{\Omega} \alpha(x) |\nabla f|$, is defined by

$$\int_{\Omega} \alpha(x) |\nabla f| =: \sup_{\phi \in \Phi_{\alpha}} \left\{ \int_{\Omega} f(x) \operatorname{div} \phi(x) dx \right\},$$

where

$$\Phi_{\alpha} =: \{\phi \in C_0^1(\Omega, R^n) | |\phi(x)| \leq \alpha(x), \text{ on } \Omega\}. \quad (14.34)$$

We also need the following lemma for our proof.

LEMMA: 1 (see [214]) Let $\Omega \subset R^n$ be an open set with a Lipschitz boundary. Suppose that f_n is a bounded sequence in $BV(\Omega)$. Then, there is a subsequence converging in $L^p(\Omega)$ to a function $f \in BV(\Omega)$ for any $1 \leq p < n/(n-1)$.

THEOREM 2 Let Ω be a bounded subset of R^n with Lipschitz boundary. Let also C^* be a differentiable contour, and $I \in L^{\infty}(\Omega)$. The minimization problem (14.33) has a solution $\chi_E \in BV(\Omega)$, μ , θ , and T .

Proof: Let E_m , μ_m , R_m , and T_m be minimizing sequences of (A.1). Denote

$$\begin{aligned} P(x) &= P(x, \mu, R, T) \\ &= g(|\nabla I|)(x) + \frac{\lambda}{2} d^2(\mu Rx + T). \end{aligned}$$

and

$$P_m(x) = P(x, \mu_m, R_m, T_m).$$

Then, as $m \rightarrow \infty$

$$\int_{\Omega} P_m |D\chi_{E_m}| \rightarrow \inf_{\chi_E \in BV, \mu, R, T} P |D\chi_E|. \quad (14.35)$$

Since $g(|\nabla I|)(x) \geq 1/(1 + C\|I\|_{L^\infty(\Omega)})$ for a suitable constant $C = C(\beta, \sigma) > 0$, the function P is bounded below by a constant $p_0 > 0$. Therefore, χ_{E_m} is a bounded sequence in $BV(\Omega)$. By Lemma B.5, there exist subsequences of χ_{E_m} , μ_m , θ_m , and T_m such that, without changing notation, χ_{E_m} converges to a function f in $L^1(\Omega)$ and a.e. on Ω . Since χ_{E_m} is either 0 or 1, f is either 0 or 1 a.e.. We may view f as the characteristic function χ_E of a set E . Therefore, we have

$$\chi_{E_m} \rightarrow \chi_E, \text{ in } L^1(\Omega).$$

Moreover, since μ_m , R_m , and T_m are bounded sequences, there exist constants μ , a constant θ , and T , such that

$$\mu_m \rightarrow \mu, \quad \theta_m \rightarrow \theta, \quad T_m \rightarrow T.$$

Noticing that μ , R , and T are involved in the continuous function P through a distance function, we have P_m converges to P uniformly on Ω . Therefore, for any $0 < \epsilon < p_0$ there exists a constant $M = M(\epsilon) > 0$ such that if $m > M$

$$P - \epsilon \leq P_m \leq P + \epsilon.$$

Then by the definition of Φ_α (ref. 14.34) we have $\Phi_{P-\epsilon} \subset \Phi_P$. Therefore, for any fixed $\phi \in \Phi_{P-\epsilon}$, if $m > M$,

$$\begin{aligned} \int_{\Omega} \chi_{E_m} \operatorname{div} \phi dx &\leq \sup_{\phi \in \Phi_{P_m}} \int_{\Omega} \chi_{E_m} \operatorname{div} \phi dx \\ &\leq \int_{\Omega} P_m |\nabla \chi_{E_m}|. \end{aligned} \quad (14.36)$$

Furthermore, since χ_{E_m} converges in $L^1(\Omega)$ to χ_E , for any $\phi \in \Phi_{P-\epsilon}$,

$$\int_{\Omega} \chi_E \operatorname{div} \phi dx = \lim_{m \rightarrow \infty} \int_{\Omega} \chi_{E_m} \operatorname{div} \phi. \quad (14.37)$$

The combination of (14.36) and (14.37) shows for any $\phi \in \Phi_{P-\epsilon}$,

$$\int_{\Omega} \chi_E \operatorname{div} \phi dx \leq \lim_{m \rightarrow \infty} \int_{\Omega} P_m |\nabla \chi_{E_m}|.$$

Therefore,

$$\sup_{\phi \in \Phi_{P-\epsilon}} \int_{\Omega} \chi_E \operatorname{div} v \phi dx \leq \lim_{m \rightarrow \infty} \int_{\Omega} P_m |\nabla \chi_{E_m}|.$$

By letting $\epsilon \rightarrow 0$, and then using (14.34) and (14.33), we get

$$\begin{aligned} \int_{\Omega} P |D\chi_E| &= \sup_{\phi \in \Phi_P} \int_{\Omega} \chi_E \operatorname{div} v \phi dx \\ &\leq \lim_{m \rightarrow \infty} \int_{\Omega} P_m |\nabla \chi_{E_m}| \rightarrow \inf_{\chi_E \in BV, \mu, \theta, T} P |D\chi_E|. \end{aligned}$$

This shows that solution $\chi_E \in BV(\Omega)$, and χ_E μ , θ , and T are the solutions to the minimization problem (14.33). ■

Acknowledgments

We would like to thank J. Ye, S.Thiruvenkadam and F.Huang for their efforts in generating the experimental results. Dr. C. M. Leonard for providing some of the brain MRI data and discussions on the hippocampal segmentations, Dr. H.D.Tagare for the valuable discussion in the idea of incorporating shape information into segmentation, Dr. Wilson and Dr. MD. E.A. Geiser for numerous discussion on the cardiac border detection and providing ultrasound images and expert's borders, Dr. R.Briggs and K.S.Gopinath for numerous discussions on the fMRI registration problems as well as providing the fMR brain images, Dr. M.Chang for the discussion of statistical method for clustering. Vemuri was supported in part by the grants NSF IIS9811042, NIH RR13197 and NS42075. Chen was supported in part by the NIH grant P50-DC03888 and the NSF grant DMS-9972662.

15

Image Alignment

Francoise Dibos, Georges Koepfler and Pascal Monasse

Abstract

Besides stating the problem of image registration this chapter is built on the following parts : 1) Similarity estimation by level sets. We use the Fast Level Sets Transform (see the chapter on TV minimization) to extract reliable features from both images. To each feature we associate similarity invariants, then we consider pairs of features (F_1, F_2), where F_1 is in image 1 and F_2 in image 2, that can match modulo a similarity. We call these pairs correspondences. These correspondences vote for the four parameters of the global similarity. The set of four parameters that gets the maximum number of votes is the estimated similarity. 2) Projective registration. Thanks to a new model for projective deformation, the registration group, we are able, after elimination of the pure projective deformation (i.e. 2 parameters), to reduce the projective matching to a similarity matching.

15.1 Introduction

15.1.1 *Plan of the chapter*

Let us briefly give a plan of this chapter on image registration. Notice that a main tool for the proposed algorithm is the Fast Level Sets Transform (FLST). This image representation using “shapes” derived from level sets, has been introduced, and applied, in the chapter on Total Variation Minimization. It is highly recommended to read the section dealing with the FLST (see page 128) before going into details of the present algorithm.

In the sequel of the introduction we will review and comment on registration techniques, this will lead us to introduce our methodology. In Section 15.2 we discuss our way of defining features and finding correspondences between two images, this will be based on similarity estimation by level sets.

Section 15.3 presents the way to determine meaningful correspondences, we propose elections to determine the four parameters of the similarity. In Section 15.4 we discuss accuracy issues of the proposed method and Section 15.5 presents the complexity of the algorithm.

In Section 15.6 we replace the similarity deformation we have been looking for between two images, by a projective transform. We explain how a projective matching can be reduced to a similarity matching. To conclude the chapter we present in Section 15.7 numerical experiments.

15.1.2 Generalities

The 2D image registration problem can be posed as follow: given two images that differ *globally* by a geometrical transformation, how to recover this transformation. The transformation is in a class of geometrical maps depending on a small number of parameters. The most classical ones are: translation (two parameters), rotation-translation (three parameters, translation plus angle of rotation) and similarity (four parameters, angle of rotation, scaling factor and translation). The rotation-translation allows to recover the pose of a flat object when the distance of the camera to the object remains constant, the camera pointing perpendicularly to the object plane and the similarity transform when the distance of the object to the camera can change, but the camera remains pointed perpendicularly to the object plane. If the camera is not perpendicular to the object plane, we obtain a projective transform (see Section 15.6).

Actually, the modeling of the problem corresponds loosely to experimental conditions. Apart from the fact that the observed images are sampled (and often not at Nyquist rate) and quantized versions of the analog ones (which is not a negligible problem), and apart from the presence of some noise of observation, the conditions of observations are often the following:

1. The lightning conditions may have changed between both snapshots, resulting in a global, or even local, contrast change;
2. The observed images are composed in general of several objects, some of them not obeying the global motion, leading in particular to occlusions.

There are roughly two kinds of techniques for registration, we refer to the review of Brown [71] and references therein. Most techniques are based on global or local correlation, and lead to fast computations in the Fourier domain, as for instance DeCastro and Morandi [150], or Yaroslavsky and Eden [588, chapter 9]. A second kind of method (like Wang and Bhattacharya [562]) first computes some image salient features, which should be sufficiently stable to appear in both images.

15.1.3 Correlation methods

The correlation methods cover in fact a large class of possibilities, sharing the common characteristic that they rely on a measure of difference between the images. Suppose we have a measure giving the distance between two images. Then the displacement between the images is the one that minimizes this distance (that cancels it, in the ideal case). The problem is therefore the minimization of a function depending on the parameters of the displacement.

The most classically used distance is given by the L^2 norm. It is equivalent to solving the optimization problem

$$\arg \max_A \int u \circ A(\mathbf{x}) v(\mathbf{x}) d\mathbf{x}$$

where A belongs to the class of admissible displacements. This quantity to be maximized is the correlation of $u \circ A$ and v . Variants of this are common, such as taking the correlation of $u \circ A - \int u$ and $v - \int v$ to be insensitive to a global shift of the contrast, or using a measure different from the Lebesgue measure in the integral, so as to highlight some parts of the image for example.

The correlation method is global, and if there are secondary motions in the images, the manner they affect the result is the following: the computed motion is a kind of average of all the motions in the images. A way to prevent this is to make local correlation, that is, compute the correlation restricted to some windows. Moreover, a change of contrast may modify the result in complex ways.

This makes the correlation method an easy but blind method: it can be sensitive to several factors in unknown ways.

15.1.4 Features matching

The features matching method is more complex to devise than the correlation based method, because of the variety of possible features and the number of ways to match them. The features can be lines, edges, corner points, etc. See for example Wang *et al.*, [558]. The only constraint is that these points must be robust enough, and their location accurate enough. These features can be local (like corner points, inflexion points of curves) or semi-local (segments).

Local correlation can be considered in some sense also as a feature matching method: a neighborhood (a window) is taken around each point and the correlation is restricted to this neighborhood.

15.1.5 Overview of the method

In contrast to local correlation methods, the proposed method does not fix any *a priori* neighborhood for matching. The semi-local methods permit to take advantage of global enough features for accurate displacement estimation, whereas being local enough to raise the hope that many of the features in both images are not altered by occlusion. Indeed, our features are the shapes. They depend on the

image and there are of all sizes. Each shape is tentatively matched with another one in the second image. For the whole registration to be contrast invariant, there are two requirements (see [375]):

1. The extracted features do not depend on the contrast in the image.
2. The matching of features does not into account their contrast.

Once we have a set of matchings of shapes between both images, we make the matchings vote for some motion. The set of parameters having received the maximum number of votes must be the global dominant displacement. Once it is found, the result can be improved by a post processing: we can make a mean of displacements given by the set of matchings compatible with the dominant motion, that is, the winners of the election. This ensures that false matchings, or matchings corresponding to secondary motions, do not mix with the dominant matchings, and hopefully the resulting average may be accurate. This is an advantage of registration by features matching: the features can be discriminated according to their vote.

15.2 Correspondences

15.2.1 *Choice of features*

Our choice is to find correspondences of features between the images. Since the contrast change that can occur between the images is one of our main concerns, we have to carefully select our features so that they do not depend themselves on the contrast. Good candidates for that are *level sets*, or better (because they are less global) connected components of level sets; level lines satisfy also the requirements. Almost equivalent features, which can be computed by a fast algorithm, are given by the inclusion tree of shapes. Actually, in the present work, the inclusion information is rather poorly used. But we have to keep in mind that it is a high level structuring information, and that it could improve dramatically the efficiency of the registration. Anyway, the features that we use here are the shapes of the images, that is, the connected components of level sets whose holes we fill.

15.2.2 *Characteristics*

The success of an image registration method based on features correspondences relies on the amount of invariance put in these features and in the correspondence rules. After having chosen contrast invariant features, we try to find contrast invariant characteristics to match them. But these characteristics must also be invariant with respect to any allowed displacement. That means that if we are looking for a similarity, the characteristics of the shapes must be similarity invariant.

We choose global and elementary characteristics of the shapes: their moments. The moments of order n of a subset S of \mathbb{R}^2 are:

$$m_{i,j} = \iint_S x^i y^j dx dy = \iint_{\mathbb{R}^2} \chi_S(x, y) x^i y^j dx dy \quad (15.1)$$

where $i + j = n$. There are $n + 1$ moments of order n . The moment of order 0 is the area of S , the moments of order 1 are $m_{1,0} = m_{0,0} \bar{x}_S$ and $m_{0,1} = m_{0,0} \bar{y}_S$, the coordinates of the barycenter of S multiplied by its area.

Except $m_{0,0}$, the area of the shape, none of the moments is invariant with respect to translation. For this, we need to use *central* moments, which are defined as

$$\begin{aligned} \mu_{i,j} &= \iint_S (x - \bar{x}_S)^i (y - \bar{y}_S)^j dx dy \\ &= \iint_{\mathbb{R}^2} \chi_S(x, y) (x - \bar{x}_S)^i (y - \bar{y}_S)^j dx dy \end{aligned} \quad (15.2)$$

Notice that $\mu_{1,0} = \mu_{0,1} = 0$. The central moments of order n are polynomial functions of the moments of order less or equal than n .

The translation invariance is not enough for the applications we aim to. We would need at least to add rotation invariance, and maybe also scale invariance. On the other hand, each invariance added disables some moments. For example, the translation invariance requirement cancels the moments of order 1, which become useless for correspondence. This effect forces us to consider higher order moments. This is a limit of this method, since it is well known that higher order moments are more sensitive to noise. That is why we restrict ourselves to such simple displacements as rotation (with unknown center) or similarity, and we will not consider moments of order larger than 3.

The inertia matrix of a subset S of \mathbb{R}^2 is the 2×2 symmetric matrix based on moments of order 2:

$$I_S = \begin{pmatrix} \mu_{2,0} & \mu_{1,1} \\ \mu_{1,1} & \mu_{0,2} \end{pmatrix}. \quad (15.3)$$

If we consider the inertia function $i : \mathbb{S}^1 \rightarrow \mathbb{R}$ which maps an angle α to the integral of the square distances of the points of S to the line passing by the barycenter and of orientation α , i is a quadratic form of the vector $(\cos \alpha, \sin \alpha)^T$ of matrix I_S . If S undergoes a rotation of angle α , the new inertia matrix of S becomes:

$$I_{R_\theta S} = R_\theta S R_{-\theta} \quad (15.4)$$

where R_θ is the rotation matrix of angle θ . This shows that the inertia matrix is not rotation invariant, but that the two numbers $\det I_S$ and $\text{tr } I_S$, respectively the determinant and the trace of the inertia matrix, are.

If we allow to go up to order 3 for the moments, two other translation-rotation invariants can be constructed:

$$\begin{aligned} r_4 &= (\mu_{30} - 3\mu_{12})^2 + (\mu_{03} - 3\mu_{21})^2 \\ r_5 &= (\mu_{30} + \mu_{12})^2 + (\mu_{03} + \mu_{21})^2. \end{aligned}$$

To sum up, we associate to each shape its vector of rotation invariant characteristics $r = (r_1, r_2, r_3, r_4, r_5)$, with

$$\begin{aligned} r_1 &= m_{00}, & r_4 &= (\mu_{30} - 3\mu_{12})^2 + (\mu_{03} - 3\mu_{21})^2, \\ r_2 &= \text{tr } I_S, & r_5 &= (\mu_{30} + \mu_{12})^2 + (\mu_{03} + \mu_{21})^2, \\ r_3 &= \det I_S. \end{aligned}$$

If we expect an important amount of noise in the images, we had better drop the characteristics r_4 and r_5 since they are likely to become insignificant.

Moreover, if we expect a scale change between the images, the area m_{00} is not any more an invariant. Nevertheless, all of these characteristics are relative invariants, meaning that we can divide them by the area to a certain power to get absolute invariants with respect to a scale change. So in the case of similarity registration, the vector of characteristics is $s = (s_1, s_2, s_3, s_4)$, with

$$\begin{aligned} s_1 &= r_2/m_{00}^2, & s_3 &= r_4/m_{00}^5, \\ s_2 &= r_3/m_{00}^4, & s_4 &= r_5/m_{00}^5. \end{aligned}$$

Again, s_3 and s_4 would rather not be used if much noise is suspected. The normalization by the area to a certain power is likely to make all these characteristics less reliable than the rotation invariant ones, so that if no change of scale is expected, it is preferable to use the rotation invariant characteristics rather than the similarity invariant ones.

Other invariants relying on moments of higher order, and the way to construct them, can be found in Reiss [431].

15.2.3 Finding correspondences

Now, given the two images u_1 and u_2 to register, we extract their shapes S_1, \dots, S_k and S'_1, \dots, S'_l (see page 128) and compute their associated vectors of characteristics, c_1, \dots, c_k and c'_1, \dots, c'_l , which can be rotation-translation invariants or similarity invariants. Since they are composed of invariants, we can compare them. We would say that shape S_i of u_1 and shape S'_j of u_2 are *matching* if their associated characteristics vectors, respectively $c_i = (c_i^1, \dots, c_i^m)$ and $c'_j = (c'_j^1, \dots, c'_j^m)$ satisfy:

$$\forall p \in \{1, \dots, m\} \quad \frac{1}{t_p} \|c'^p_j\| \leq \|c^p_i\| \leq t_p \|c'^p_j\|, \quad (15.5)$$

where $t = (t_1, \dots, t_m)$ is composed of thresholds of tolerance, $t_p \geq 1$ for all p . The nearer t_p to 1, the less tolerant we are about an error in characteristic c^p .

If S_i and S'_j are matching, we note this correspondence $C = (S_i, S'_j)$.

Remark: It is to be noticed that Equation (15.5) is clearly reflexive and symmetric but *not* transitive (unless $t = (1, \dots, 1)$), which is what we expect of a similarity relation: every shape can be mapped to any other shape by a sufficient number of small perturbations, each intermediary shape being similar to the previous one.

15.3 Votes

Once we have found the set of correspondences $\mathcal{C} = (C_1, \dots, C_q)$ between images u_1 and u_2 , we are in position to determine the displacement. Nevertheless, with the crude criteria of correspondence we use, we expect to find a large number of erroneous correspondences. That is why we would not like to make all the correspondences to participate in the determination of the displacement, but rather to find a way of automatic selection of the “good” ones.

A good means to do that is to design an election: the correspondences vote for a displacement, that is, for a set of parameters, and the set of parameters having received the largest number of votes is our estimation of the displacement. This is the principle of a voting procedure, but this is not exactly what we do: actually, each correspondence is theoretically sufficient to determine a displacement, whether it be a rotation-translation or a similarity, by comparing the moments (for example, the scale change would be the ratio of areas, the rotation angle would be the angle between the directions of eigenvectors of the inertia matrix, the translation the vector determined by the barycenters). But again, we are not confident enough about the values of the moments to estimate directly the parameters. As we have said, these moments are just good enough to make a preliminary selection of the correspondences. Nevertheless, the barycenters, since they are based on moments of order 1, are likely to be fairly reliable. In this manner, one correspondence is just sufficient to estimate a translation, but not more. On the contrary, based on *two* correspondences, we have two points and their displaced images, so this is just what is needed to estimate a similarity, and more than necessary to estimate a rotation-translation.

If we consider two correspondences (S_i, S'_j) and $(S_{i'}, S'_{j'})$ of respective barycenters \mathbf{x}_i , \mathbf{y}_j , $\mathbf{x}_{i'}$ and $\mathbf{y}_{j'}$, the parameters of the similarity, s , θ and \mathbf{t} are given by:

$$s = \frac{\|\mathbf{y}_{j'} - \mathbf{y}_j\|}{\|\mathbf{x}_{i'} - \mathbf{x}_i\|} \quad (15.6)$$

$$\sin\theta = \frac{(\mathbf{x}_{i'} - \mathbf{x}_i) \wedge (\mathbf{y}_{j'} - \mathbf{y}_j)}{\|\mathbf{x}_{i'} - \mathbf{x}_i\| \cdot \|\mathbf{y}_{j'} - \mathbf{y}_j\|} \quad (15.6')$$

$$\cos\theta = \frac{(\mathbf{x}_{i'} - \mathbf{x}_i) \cdot (\mathbf{y}_{j'} - \mathbf{y}_j)}{\|\mathbf{x}_{i'} - \mathbf{x}_i\| \cdot \|\mathbf{y}_{j'} - \mathbf{y}_j\|} \quad (15.6'')$$

$$\mathbf{t} = \mathbf{y}_j - s R_\theta \mathbf{x}_i \quad (= \mathbf{y}_{j'} - s R_\theta \mathbf{x}_{i'}) \quad (15.6''')$$

If we estimate only a rotation-translation, we should check that $s = 1$, allowing however a certain tolerance. If that is not the case, both correspondences are considered incompatible and they do not vote together. Notice that these formulas involve divisions by the quantities $\|\mathbf{x}_{i'} - \mathbf{x}_i\|$ and $\|\mathbf{y}_{j'} - \mathbf{y}_j\|$, which can sometimes nearly vanish. We should therefore check that they are sufficiently large, so that the vote be precise enough. Otherwise, we should hinder their vote. Another verification before the votes would be to check that the ratios of area of corresponding shapes correspond approximately to the estimated scale factor s .

If these conditions are fulfilled, the pair of correspondences (S_i, S'_j) and $(S_{i'}, S'_{j'})$ is allowed to vote. That means that a counter corresponding to a place in a 3-D (for rotation-translation) or 4-D (for similarity) parameter space is incremented. However, finding a maximum in such a histogram can take time. We would rather restrict ourself to 1-D or 2-D histograms. The translation part of the displacement, t , can be estimated only after s and θ , so that we can separate the estimation of the linear part and of the affine part. In a first step we can make all pairs of correspondences vote (in fact only the compatible pairs) for the scale (for the case of similarity) and the rotation, find the maximum counter in this histogram and then, in a second step, vote in a 2-D histogram for the translation by using the estimated linear part of the displacement. In this manner, only 1-D or 2-D histograms are used.

Notice this voting procedure has already been used by Chang *et al.* in [104].

15.4 Accuracy

Since we work with digital images, the positions of the shapes are precise in the best case to one pixel¹. This hinders the voting procedure to yield an estimation of the parameters better than one pixel. In some applications, this is far from sufficient. An accuracy of one tenth of a pixel or one hundredth of a pixel is needed. If the errors due to digitization are Gaussian distributed, a linear regression of the estimated parameters would yield a more accurate result. In this regression, that is a mean of different estimations, we must not include the false estimations, due to erroneous correspondences. Therefore, we estimate it only with the electorate of the elected displacement. This bootstrap estimation is likely to be much better.

We begin by selecting the correspondences that are compatible with the winning displacement (the electorate of the dominant motion). The correspondence (S_i, S'_j) is considered compatible if

$$\|\mathbf{y}_j - s R_\theta \mathbf{x}_i - \mathbf{t}\| \leq \epsilon.$$

The threshold ϵ can be estimated from the shape of the histograms near their maximum. Sharper peaks would mean that the votes are precise, and ϵ can be

¹This restriction also applies to correlation methods. To have better accuracy, it is necessary to interpolate the images, or equivalently the correlation surface, see Tian and Huhns [524].

small, whereas smoother peaks would mean that many votes are not precise, so a larger ϵ would be adapted. More precisely, what we actually do is not only find the maximum in the histogram for each parameter, but also select a mode around this maximum, the mode corresponding to the largest interval around the maximum where the histogram remains concave; then we consider that the correspondence is compatible with the motion if there is a set of parameters in each mode such that \mathbf{x}_i would be mapped exactly to \mathbf{y}_j . This way of doing avoids the introduction of a supplementary parameter in the algorithm, ϵ .

We note the set of compatible correspondences \mathcal{C}_c . Then we solve the minimization problem:

$$\arg \min_{s, \theta, \mathbf{t}} \sum_{(S_i, S'_j) \in \mathcal{C}_c} \|s R_\theta \mathbf{x}_i + \mathbf{t} - \mathbf{y}_j\|^2. \quad (15.7)$$

The problem is made linear by changing the unknown variables:

$$\arg \min_{s_1, s_2, \mathbf{t}} \sum \left\| \begin{pmatrix} s_1 & -s_2 \\ s_2 & s_1 \end{pmatrix} \mathbf{x}_i + \mathbf{t} - \mathbf{y}_j \right\|^2. \quad (15.8)$$

Noting $\mathbf{S} = (s_1 \ s_2)^T$ and taking into account the equality

$$\begin{pmatrix} s_1 & -s_2 \\ s_2 & s_1 \end{pmatrix} \mathbf{x}_i = A^{(i)} \mathbf{S} \text{ where } A^{(i)} = \begin{pmatrix} x_i & -y_i \\ y_i & x_i \end{pmatrix},$$

we can rewrite Equation (15.8) into

$$\arg \min_{\mathbf{S}, \mathbf{t}} \sum \|A^{(i)} \mathbf{S} + \mathbf{t} - \mathbf{y}_j\|^2. \quad (15.9)$$

In order to solve Equation (15.9), we compute the partial derivatives relative to the parameters and equate them to 0. This yields the system

$$\begin{aligned} \sum A^{(i)T} A^{(i)} \mathbf{S} + \left(\sum A^{(i)} \right)^T \mathbf{t} &= \sum A^{(i)T} \mathbf{y}_j \\ \sum A^{(i)} \mathbf{S} + N \mathbf{t} &= \sum \mathbf{y}_j \end{aligned}$$

if N is the number of correspondences compatible with the modes. After some easy algebraic manipulations, we get

$$\begin{aligned} &\left[\sum A^{(i)T} A^{(i)} - \frac{1}{N} \left(\sum A^{(i)} \right)^T \left(\sum A^{(i)} \right) \right] \mathbf{S} \\ &= \sum A^{(i)T} \mathbf{y}_j - \frac{1}{N} \left(\sum A^{(i)} \right)^T \left(\sum \mathbf{y}_j \right) \end{aligned}$$

and

$$\mathbf{t} = \frac{1}{N} \left[\sum \mathbf{y}_j - \sum A^{(i)} \mathbf{S} \right]$$

which allows to compute the vector \mathbf{S} and then the vector \mathbf{t} .

To compute S , we have to invert the 2×2 matrix

$$\sum_i A^{(i)T} A^{(i)} - \frac{1}{N} \left(\sum_i A^{(i)} \right)^T \left(\sum_i A^{(i)} \right)$$

which, by the Cauchy-Schwarz inequality, is singular if and only if all the $A^{(i)}$ are proportional, that is, all the \mathbf{x}_i are aligned.

15.5 Complexity

If k is the number of shapes of the first image and l of the second image, we can theoretically have up to $k.l$ correspondences and the complexity of the voting step is $(k.l)^2$ since we have to take into account all pairs of correspondences. Even if we remove too small shapes (less than 20 pixels for example), the number of *shapes* in each image can be of the order of tens of thousands. So the voting procedure becomes at this condition non affordable. However, this is a worst case configuration. In general, the number of correspondences is far less than $k.l$; it is fortunately rather of the order of magnitude of $\max(k, l)$. But even as this, the voting procedure can be too greedy in computational time for us.

To reduce even more the number of correspondences, the most basic solution is to quantize the images before extracting the shapes. The number of shapes is almost automatically divided by the quantization step. But this quantization can lose important objects in the image. A better idea is to group the shapes into “objects”, closely related to branches in the tree. Indeed, owing to the smoothing done by the lens of the camera, objects in the scene often correspond to several nested shapes. The idea is to group these nested shapes into the same “object” and make objects vote instead of shapes directly, with a weight corresponding to the number of shapes included in the objects, since objects composed of many shapes are likely to represent important things in the scene. We are quite confident that preventing the votes of pairs of corresponding shapes inside the same “object” would not disturb the histograms of votes. Actually, shapes inside the same object are very close, so that their common vote given by Equations (15.6), (15.6') and (15.6'') would be very imprecise.

15.6 Projective registration

When the camera does not remain pointed perpendicularly to the object plane, we can observe a projective deformation (see [186, 191, 232]). Usually eight parameters have to be determined for the projective deformation. The registration group, which will be defined below, reduces the number of parameters to six. Indeed, by removing the pure projective deformation (2 parameters), the registration group allows to reduce the projective matching to a similarity matching.

15.6.1 A 3D representation for projective deformation

In this section we present results previously used in a similar way in [160]. Let f be a planar image living in the affine plane ($Z = 1$). Let \mathcal{E} be the affine space and (X, Y, Z) the coordinates of a point $M \in \mathcal{E}$ in the orthonormal frame of reference (O, i, j, k) . The initial location of the camera is O and we may consider that, when we observe f , we observe in fact the restriction to the plane P of the 3D image

$$F(X, Y, Z) = f\left(\frac{X}{Z}, \frac{Y}{Z}\right) \quad \text{if } Z \neq 0.$$

This representation may be considered as a simplification of the classical pinhole camera model.

Thus, let D be a displacement of the camera or, in an equivalent way, a displacement of the plane P . D may be written in a unique way as $D = T \circ R$, where R is a rotation with axis containing O , and T a translation.

Let Q be the plane $D(P)$ with orthonormal basis $(\omega = D(I), R(i), R(j))$, with the point I ($I = (1, 0, 0)$) belonging to the plane P .

Moreover, let g be the image obtained by intersecting the 3D image F with the plane Q . If $M \in Q$, we have $g(M) = f(m)$. With $m \in P$ and such that m is the intersection of the line (OM) with the plane P . Therefore, if we write

$$\overrightarrow{OM} = Xi + Yj + Zk \quad \text{then} \quad m = \left(\frac{X}{Z}, \frac{Y}{Z}\right).$$

If, moreover, $\overrightarrow{\omega M} = xR(i) + yR(j)$, with

$$R = \begin{pmatrix} a & b & c \\ a' & b' & c' \\ a'' & b'' & c'' \end{pmatrix}.$$

and, if the translation T is given by the vector $\vec{T} = (\alpha, \beta, \gamma)$, thus

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} ax + by + c + \alpha \\ a'x + b'y + c' + \beta \\ a''x + b''y + c'' + \gamma \end{pmatrix}.$$

Finally, as $g(M) = f(m)$, we obtain

$$g(x, y) = f\left(\frac{ax + by + c + \alpha}{a''x + b''y + c'' + \gamma}, \frac{a'x + b'y + c' + \beta}{a''x + b''y + c'' + \gamma}\right) = (\varphi f)(x, y).$$

So, a 3×3 matrix associated to the projective application φ may be written

$$\mathcal{M}\varphi = \begin{pmatrix} a & b & c + \alpha \\ a' & b' & c' + \beta \\ a'' & b'' & c'' + \gamma \end{pmatrix} = R \begin{pmatrix} 1 & 0 & \vec{T} \cdot R(i) \\ 0 & 1 & \vec{T} \cdot R(j) \\ 0 & 0 & 1 + \vec{T} \cdot R(k) \end{pmatrix} = RH,$$

and this decomposition is shown to be unique.

Remarks:

1) We never obtain an affine application which is not a similitude. Indeed, in order to obtain an affine application, we must have $a'' = b'' = 0$ and as R is a rotation, we have $c = c' = 0$ and $c'' = 1$. Thus R is a rotation with axis k . Nevertheless, it may occur that g is also a general affine deformation of f . For example, let \mathcal{E}_α be the ellipse

$$\mathcal{E}_\alpha : (\cos 2\alpha)x^2 + (\cos^2 2\alpha)(y - \tan 2\alpha)^2 = 1,$$

obtained as the deformation of the circle $\mathcal{C} \subset P$, ($\mathcal{C} : x^2 + y^2 = 1$), by the projective transformation associated to the rotation with axis i and angle α (α small). \mathcal{E}_α is also the image of \mathcal{C} under the affine transformation

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{\cos 2\alpha}} & 0 \\ 0 & \frac{1}{\cos 2\alpha} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 0 \\ \tan 2\alpha \end{pmatrix}.$$

2) In order to have a significant deformation, the rotation R must be such that $a'' \approx 0$ and $b'' \approx 0$. Therefore affine deformations are good approximations of significant projective deformation. This may explain the large number of papers on this subject ([9],[129],[288], [374],[461]).

3) In the decomposition of $\mathcal{M}\varphi$, R is a rotation and H is a 3×3 matrix, associated to the homothety translation defined by the translation component of the displacement of the camera.

Let \mathcal{O} be the special orthogonal group and \mathcal{T} the group of the 3×3 matrices

$$\begin{pmatrix} C & 0 & A \\ 0 & C & B \\ 0 & 0 & 1 \end{pmatrix} \quad C^2 = 1.$$

We can prove that the special linear group, and thus the projective group, is generated by $\mathcal{O} \cup \mathcal{T}$, this may explain the common modelization of the matching problem with the projective group. But the application φ^{-1} , which is used in that case for recovering f from g , is not the projective application associated with D^{-1} . The reciprocity principle, developed next, allows to give an answer to this problem.

15.6.2 The reciprocity principle

If g is a deformation of f , we may also consider that f is a deformation of g . Thus, let $Q(\omega, R(i), R(j))$ be the new reference plane, and O' be the new optical center ($\overrightarrow{O'\omega} = R(k)$). If we write

$$g(M) = f(m), \quad M \in Q, \quad m \in P, \quad m \in (O', M),$$

we have

$$g(x, y) = f(\psi^{-1}(x, y));$$

where a 3×3 matrix associated to ψ is

$$\mathcal{M}\psi = \begin{pmatrix} a & a' & a'' - R^{-1}(\vec{T}) \cdot i \\ b & b' & b'' - R^{-1}(\vec{T}) \cdot j \\ c & c' & c'' - R^{-1}(\vec{T}) \cdot k \end{pmatrix} = R^{-1} \begin{pmatrix} 1 & 0 & -\alpha \\ 0 & 1 & -\beta \\ 0 & 0 & 1 - \gamma \end{pmatrix} = R^{-1} \tilde{H}.$$

Remark: If the displacement of the camera allowing to go from f to g may be written $D = T \circ R$, the displacement allowing to go from g to f is

$$D^{-1} = R^{-1} \circ T^{-1} = \tilde{T} \circ R^{-1}$$

where the vector of the translation \tilde{T} is $-R^{-1}(\vec{T})$. Thus, we notice that the decomposition of the matrix $\mathcal{M}\psi$ follows the same rules as the decomposition of $\mathcal{M}\varphi$.

This allows to define a new group, the registration group, for modeling the matching problem.

15.6.3 The registration group

Let \mathcal{A} be the subset of the 3D matrices Φ , such that Φ may be written as

$$\Phi = \begin{pmatrix} a & b & c + \alpha \\ a' & b' & c' + \beta \\ a'' & b'' & c'' + \gamma \end{pmatrix} = RH, \quad \text{where } R = \begin{pmatrix} a & b & c \\ a' & b' & c' \\ a'' & b'' & c'' \end{pmatrix}$$

is a rotation.

DEFINITION: *The registration group* $(\mathcal{A}, *)$, is the group isomorphic to the group of the displacements (\mathcal{D}, \circ) , such that the isomorphism \mathcal{I} is defined in the following way

$$\forall \Phi \in \mathcal{A} \quad \mathcal{I}(\Phi) = T \circ R,$$

where T is the translation with vector $\vec{T} = (\alpha, \beta, \gamma)$ and R the rotation defined above.

Thus $\Phi_1 * \Phi_2 = \Phi$ where Φ is the 3D matrix associated to the displacement $T_1 \circ R_1 \circ T_2 \circ R_2 = T \circ R$ where T is the translation with vector $\vec{T} = \vec{T}_1 + R_1(\vec{T}_2)$ and $R = R_1 \circ R_2$.

The definition of the registration group has been introduced in [161].

Remark: Let g be a projective deformation of f . We may assume that $g(x, y) = f(\varphi(x, y))$ where φ is defined by a matrix Φ of the registration group. If the projective group is used we define f as $f(x, y) = g(\varphi^{-1}(x, y))$.

In the new model, we define f as $f(x, y) = g(\psi(x, y))$ where ψ is the projective transformation defined by the inverse $R^{-1} \tilde{H}$ of Φ in $(\mathcal{A}, *)$.

15.6.4 An algorithm for the matches.

Let r, h, \tilde{h} the projective transformations associated to R, H, \tilde{H} .
We have, on one side

$$g(x, y) = f(r \circ h(x, y)) = rf(h(x, y)) = h(rf)(x, y)$$

and on the other side

$$f(x, y) = \tilde{h}(r^{-1}g)(x, y).$$

Therefore, g may be retrieved from rf by an homothety translation and in the same way f may be retrieved from $r^{-1}g$ by an homothety translation. It can be

shown that, in order to have a significant deformation, the 3D rotation associated with r must be such that $R(k) \approx k$. Moreover, let R and \hat{R} be two rotations such that $R(k) = \hat{R}(k)$. There exists a rotation R_k with axis k such that $\hat{R} = R \circ R_k$. Thus, if $g(x, y) = f(\hat{R} \circ h(x, y)) = (r_k \circ h)(rf)(x, y)$, g may be retrieved from rf by a similitude.

We therefore have just to consider the rotations with axis Δ , $O \in \Delta$ and Δ belonging to the plane ($Z = 0$). This needs two parameters, θ for the location of Δ in the plane and α for the angle of the rotation (α has to be small).

Then the matching between f and g can be performed in the following way: we calculate the images $r_{\Delta, \alpha}f$ and, afterwards, we match these images with g , by using a similitude (i.e. 4 parameters, see [375]).

15.7 Experimental results

In this section, we present a variety of real-world and synthetic examples of registration. The real-world experiments are chosen difficult on purpose, each one emphasizing its own type of difficulty. However, these difficulties are quite common case and happen frequently. This shows that image registration is not a trivial task, and that all these difficulties must be taken into account when devising a strategy of image registration.

15.7.1 Pose estimation

The classical problem of pose estimation is to find the orientation of a flat object viewed from a fixed distance and perpendicularly to its plane. The goal is thus to recover a rotation-translation. This is typically the first task to accomplish in automatic industrial quality inspection on production lines. One object must be compared to a template image to check its conformity. Small displacements are to be excepted in the process, and they must be compensated prior to comparison with the template.

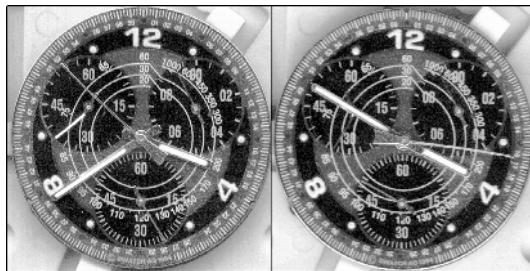


Figure 15.1. Two shots of the same watch at different time and with a slightly different position.

What time is it? Our first experiment is about images of a watch taken at two different times (as the hands witness) and with different poses, see Figure 15.1. The face of the watch was lain on the glass of a scanner of average quality and two shots were taken, the watch being moved and the hour changed between both. The good point for our algorithm is the multiplicity of distinct marks on the face, of all sizes. A large number of them is not occluded by the hands, raising hope for a good registration. Nevertheless, this is also challenging, for the following reasons.

- The images are fairly noisy, this is particularly visible on zones which should be uniform.
- The right image is a bit blurred.
- The hands move with a motion completely different from the global one, and are occluding some shapes.
- Many shapes are present multiple times in the image, and some of them are represented a large number of times, especially the tick marks, but also some digits, yielding concurrent motions.
- The structural rotational self-similarity could give a large number of votes for rotations around the center of the watch, hiding the motion due to displacement.

The images are of approximate size 500×500 . We keep only shapes of area at least 20 pixels and not meeting the frame (because shapes meeting the frame represent objects occluded by the framing). Owing to the large number of meaningful shapes, we limit the distance of two shapes extracted from both images not to exceed 50 pixels to allow them to match. Without this limitation, the number of correspondences would be tremendously high, making the election impossible to accomplish in a reasonable time. With the above restrictions, the number of correspondences of shapes is 171,015, involving 20,308 shapes in the left image and 19,247 shapes in the right one. When the shapes are glued into sections, this reduces to 7887 correspondences of sections, involving 4100 sections of the left image and 3568 in the right one. The image reconstructed from the inclusion tree

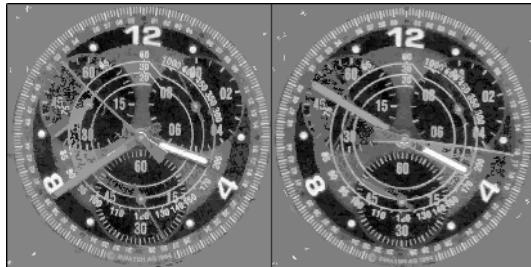


Figure 15.2. The images reconstructed from the shapes of each image of Figure 15.1 that have at least one corresponding shape in the other image. Shapes meeting the frame of the image are voluntarily removed.

when we keep only the shapes that match in the other image are shown in Figure 15.2.

The election is performed in the following manner: we vote first for zoom/rotation (by considering all pairs of correspondences of sections), determine the dominant one, apply it to the left image (with arbitrary center of rotation) and then vote for the translation. The sampling is chosen of 1 pixel for the translation, and scales are sampled equally between $\frac{1}{1.10}$ and 1.10 (we allow at most 10% of rate of change between the images) in 145 samples and the angle of rotation in 1450 samples between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$. These samplings are chosen so that the vote of two correspondences of sections has the precision of one sample on average. Notice that this accuracy highly depends on the distance of the (centers of) shapes in the same image. Since the repartition of shapes in the images is *a priori* unknown, this average distance cannot be computed, but this typically should be a fraction of the highest distance in the image, its diagonal length. The important point to notice here is that the sampling of the angle of rotation and of the change of scale depend on the size of the image: we cannot expect the same accuracy in the parameters for small and large images, but this is not a limitation; indeed, the important point is not the absolute accuracy of the parameters, but the precision of the registration, in terms of number of pixels.

With such samplings the dominant motion corresponds to almost no change of scale (1.00006), a small rotation (0.44 deg clockwise) and a translation of $(-13, -1)$ when the center of rotation is the upper left corner of the image. The 2-D histograms of votes are shown in Figure 15.3. The graph of these histograms around their maximum is shown in Figure 15.4. The number of votes for zoom/rotation is much higher because this corresponds to votes of *pairs* of correspondences, whereas the vote for the translation corresponds to single correspondences. The precision is inspected in the following manner: we apply the dominant motion to the left image and superimpose it on the right one (we take the mean of two superimposed pixels), see Figure 15.5. Notice the resulting image is good, except that each hand appears two times, because the hands move with a different motion.



Figure 15.3. The votes for the parameters of the similarity in the watch images of Figure 15.1. Left: the histogram of votes for the zoom factor (horizontal axis) and the angle of rotation (vertical axis), the gray level being proportional to the number of votes. Right: the histogram of votes for the translation in x (horizontal axis) and y (vertical axis). Notice how the peaks in these histograms are sharp and unambiguous, raising confidence in the estimated motion.

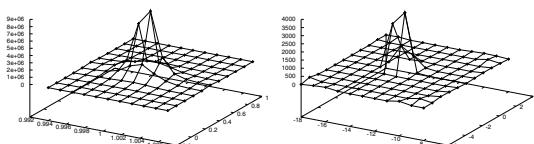


Figure 15.4. The graphs of the histograms of votes of Figure 15.3 around their maximum.

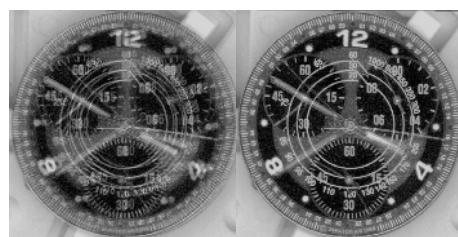


Figure 15.5. Registration of the images of Figure 15.1. Left: superimposition without registration. Right: superimposition after application of the estimated motion to the first image.

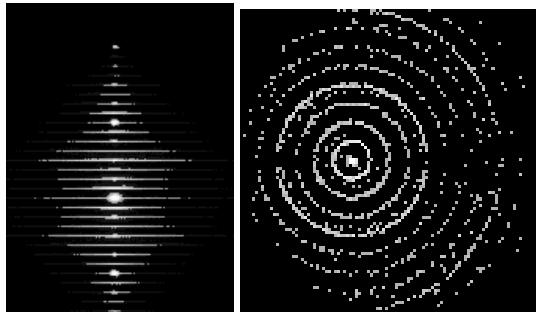


Figure 15.6. The votes for the similarity parameters in the registration of the watches in Figure 15.1, as in Figure 15.3, but with a nonlinear change of contrast enhancing dark pixels. This makes some secondary peaks visible. Notice in particular both symmetric main secondary peaks in the histogram for zoom/rotation (left). They correspond to rotations of 1 minute clockwise and counter clockwise.



Figure 15.7. The registrations corresponding to the main secondary peaks in the histogram of votes for zoom/rotation of the images of Figure 15.1. They correspond to a rotation of 6 deg clockwise (right image) and counter clockwise relatively to the dominant motion. Notice the ticks are perfectly registered in both images, showing that the secondary peaks correspond to their votes.

A close inspection of the vote images witnesses the quasi self similarity in rotation of these images: the main two secondary peaks in the image correspond precisely to the shift of 1 minute tick, that is, the numbered ticks (see Figure 15.6). Figure 15.7 shows the registration corresponding to these secondary peaks. The shifts of multiple ticks do not appear because the distance becomes greater than 50 pixels. Once the zoom-rotation is fixed, this quasi-invariance is found again in the polls for translation, in the form of concentric circles.

Whereas globally satisfactory, this registration is not a total success. When we try to reach subpixel accuracy, as explained in the previous chapter, we get a partial stroboscopic effect due to the ticks in the bottom right part of the image, whereas the top left part is correctly registered. Figure 15.8 shows the computed registration and Figure 15.9 the shapes participating to the least squares estimate. The new zoom factor is 1.0003, the rotation 0.42 clockwise and the translation vector $(-12.9, 0.0)$. A close look at the registered images shows that the accuracy was not really improved, and small shifts remain present. The estimated motion



Figure 15.8. The registration of the image of Figure 15.1 obtained by least squares estimate of the electorate of the dominant motion. The result is fairly deceptive, due to the mixing of concurrent incompatible motions.

comprises some false correspondences: the computed motion is not the right one. The reason is that some shapes in the bottom right part are considered as matching correctly with several other shapes, and the average yields a bad result. The reason this happens in the bottom right part is that the center of rotation was chosen as the top left corner of the image, since the choice is arbitrary. A tolerance of error in the angle of rotation yields a small shift in the shapes of near this center, selecting only the right correspondences, whereas it authorizes a larger shift far from it (in the bottom right part), making some correspondences coming from the stroboscopic effect compatible with the dominant motion. This effect can happen in all images with high self similarity. The solution to this would be to select more reliably the correspondences compatible with the computed motion.

The images shown in Figure 15.9 can be interpreted as a kind of intersection of images. The left image is reconstructed from the shapes that have been recognized at the right location in the right image, and conversely. This is very close to what is proposed by Ballester *et al.* in [27]: they extract connected components of bilevel sets ($[\lambda \leq u(\mathbf{x}) \leq \mu]$) and look for a corresponding one in the same location in the other image. The correspondences are established in a manner very similar to ours, by comparing moments, perimeter, and so on. Each connected component of bilevel set having found a correspondent in the other image is kept, and the image is reconstructed from them. This differs from our “intersection” mainly because, apart from the fact that there is supposed to be no motion in their case, bilevel sets are not sufficient to reconstruct an image unambiguously. They have no tree structure, and they define for each image two intersections: one for which each pixel get the maximum of the lower thresholds of the remaining connected components of bilevel sets containing it, the other one with the minimum of the upper thresholds. The inclusion tree structure of shapes allows us to reconstruct *one* image of intersection for each original image.

15.7.2 Similarity

We perform a full similarity estimate on Mona Lisa images shown in Figure 15.10. The left image, of size 374×562 , is a scanned photograph. The right image, of size

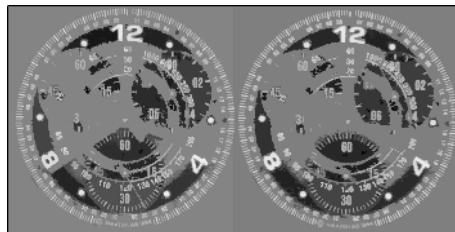


Figure 15.9. The images reconstructed from the shapes of each image of Figure 15.1 involved in a correspondence participating to the least squares estimate. The result seems fairly coherent, except some shapes which appear in one image and not in the other one.

560×864 , is extracted from a database on the World Wide Web. These images are a real challenge from the point of view of registration, for the following reasons:

- There is a strong contrast change between the images.
- The zoom factor is important.
- The images are of poor quality. Moreover, they come from different sources, each adding its own type of noise.
- Strong edges and clearly visible details are scarce. The style of Da Vinci relies on gradations and smooth contrasts, the associated level lines become very noisy in the digital image.
- It is not ensured that the global motion is a similarity. A small difference in the angle from which the picture was photographed could yield a small projective transform.

There are 925 sections in left image, and 8920 in right one. The number of correspondences of shapes is 6718, which translate into 3842 correspondences of sections. Parts of the histograms of votes are shown in Figure 15.11. Notice that peaks are not very sharp. The estimated parameters are a zoom factor of 1.54, a rotation of 1.34 clockwise and a translation of $(-18.5, -0.4)$.

The superimposition of the images is shown in Figure 15.12. The result is globally correct, although small shifts can be observed in the lower part. When we try to register them by hand, we can observe that no similarity gives perfect result. The transformation is in fact a bit more complex, affine or maybe projective.

15.7.3 Accuracy

We investigate in this section the accuracy of the computed registration, and particularly the question of whether subpixel accuracy is reached by that method. However, we want to make the experiment “realistic”. This means that the conditions in which the images we register are created must be conform to what real captors do. In particular, we do not deal with images of functions of simple analytic form. Therefore, our procedure is the following: we take an image of large

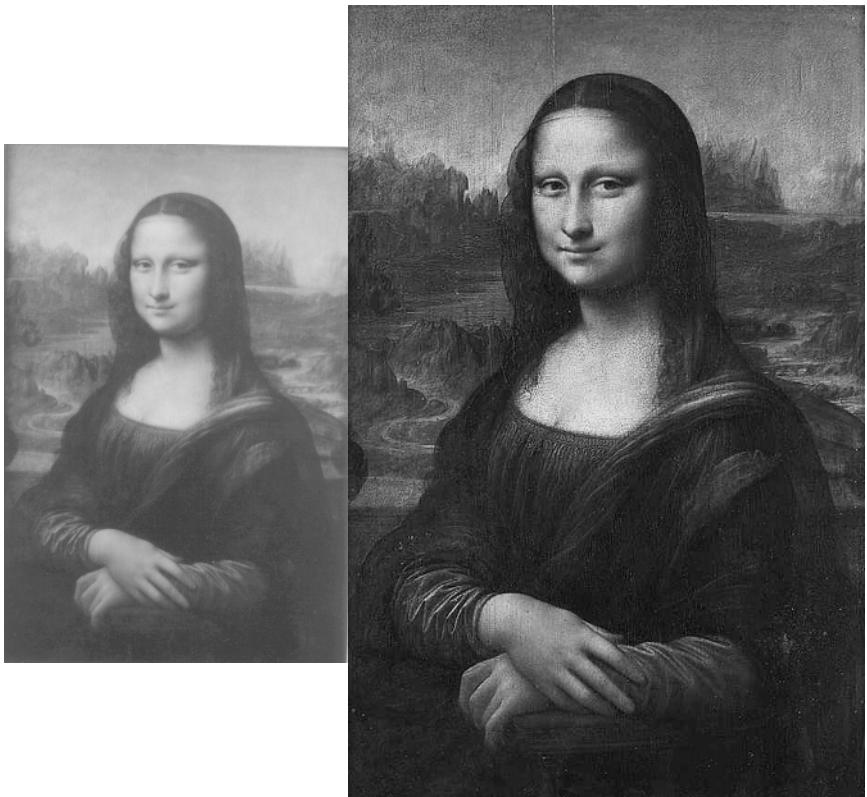


Figure 15.10. The Mona Lisa images used to test the similarity registration.

size and subsample it by some factor; the second image is created by translating the original image by a small amount (typically, not a multiple of the sampling rate) and then subsampling it. In this manner, the shift between the two images is really a fraction of pixel. We compare the estimated translation to this shift.

The crucial point in this procedure is that we do not smooth the image before sampling it: we voluntarily create aliasing. The reason is that almost all captors do not sample the continuous image smoothed by the lens at Nyquist rate. Most, if not all, the images we deal with, are not sampled in the conditions of Shannon reconstruction theorem. If this were the case, the registration by correlation would not only reach subpixel accuracy, as reported in [524], but would have a theoretical infinite accuracy. This would be the definitive method. But in real cases, the images are not sampled according to Nyquist rate, so their interpolation by cardinal sines does not correspond to the original continuous domain image, so the accuracy of the registration by correlation can be quite bad, all the more that the sampling rate is lower than Nyquist requirement.

The image used to lead our investigations is a satellite image of captor Spot 2, of size 6000×6000 . We sample it by a factor ten in x and y , that is, we

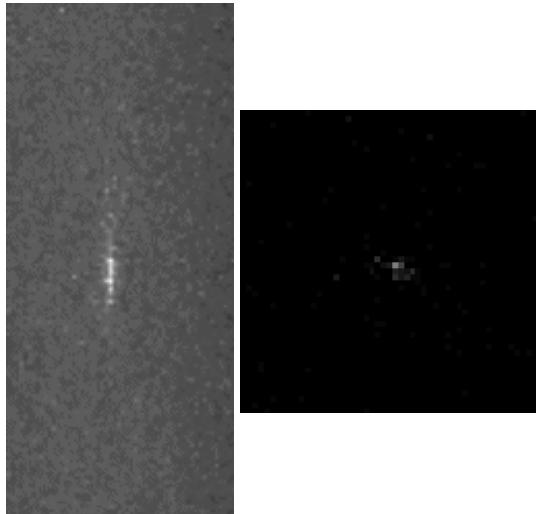


Figure 15.11. The histograms of votes, around their maximum, in the registration of the Mona Lisa images of Figure 15.10. Left: votes for zoom (horizontal axis)/rotation (vertical axis). Right: votes for the translation.

keep only one pixel for 100 pixels in the original image. This is our left image, shown in Figure 15.13. Notice this is not an easy image to deal with, it is fairly oscillatory. The right image is created by shifting the original image by n pixels before sampling, $n = 1, \dots, 9$. The true translation in our registration experiment is thus $n/10$.

In the experiments, the number of shapes of area larger than 20 pixels and not meeting the frame of the image, is around 20,000 in each image. This induces around 30,000 correspondences of shapes. The shapes are grouped into sections, whose number is approximatively 8000 in each image, and we have 8500 correspondences of sections. The errors in the estimated translation are reported in Figure 15.14. As expected, the error is all the more important that the real shift is close to half a pixel. Nevertheless, the errors remain moderate, and we could talk about an overall tenth of pixel accuracy.

However, this accuracy depends probably on the number of shapes in the image. Since the estimates are the results of an average, this can be expected to be all the lower than the image is rich in details, on the contrary to registration by correlation, for which the accuracy is independent of the size of the image (if we neglect border effects).

15.7.4 Projective registration

A first experiment of registration with a synthetic image is shown Figure 15.15. Let f be a projective deformation of image g . First, the parameters of the projective transformations, $r_{\Delta, \alpha}$, (see 15.6.4) are discretized. Experiences show that



Figure 15.12. Superimposition of the two images of Figure 15.10 after application of the estimated similarity registration.

about twenty ϕ_i , $i \in I$, out of those discretized transformations are enough. We construct all the images $\phi_i f$ and apply the similarity registration between $\phi_i f$ and g . For each i we will find (or not) a similarity S_i with a corresponding number of votes n_i . Let us choose $n_{i_0} = \sup_{i \in I} n_i$, then in $\phi_{i_0} f = \tilde{f}$ the purely projective part of f has been removed. The registration is achieved by matching \tilde{f} and g with the similarity S_{i_0} .



Figure 15.13. The Spot 2 image we use to test the accuracy of our registration method. Its size is 600×600 , constructed by sampling of a factor 10 along each axis the original image of size 6000×6000 .

t_x	t'_x	$ t_x - t'_x $	t_y	t'_y
0.1	0.078	0.022	0	0.013
0.2	0.138	0.062	0	0.000
0.3	0.230	0.070	0	0.040
0.4	0.289	0.111	0	0.138
0.5	0.457	0.043	0	0.049
0.6	0.531	0.069	0	0.163
0.7	0.767	0.067	0	0.046
0.8	0.834	0.034	0	0.001
0.9	0.912	0.012	0	0.008

Figure 15.14. The error in the estimated translation for various shifts. See text for details. t is the true translation vector, t' is the estimated translation.

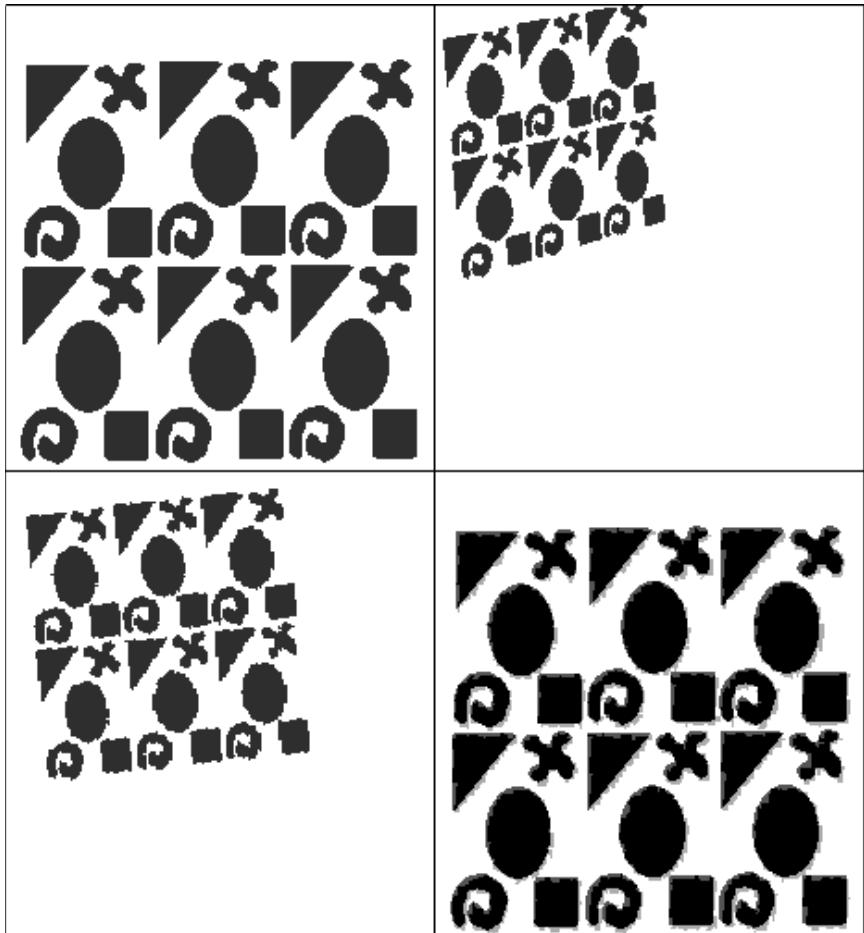


Figure 15.15. Projective registration experiment. Top left: original image. Top right: image transformed via a projective transformation. Bottom left: the purely projective part of the transformation is removed. Bottom right: the estimated similarity (zoom factor: 1.5, rotation angle: 5.7 deg) is applied to the preceding image and we superimpose it with the original image.

Part VI

Motion Analysis

16

Variational Principles in Optical Flow Estimation and Tracking

Nikos Paragios and Rachid Deriche

Abstract

In this chapter, we propose a variational formulation to motion analysis where two important topics are addressed; optical flow estimation and tracking. To this end, the propagation of smooth interfaces is used to perform tracking and the incremental estimation of parametric models to recover the apparent motion. The objective function is defined in the space of level set representations and couples motion estimation and tracking. It consists of a boundary attraction term, a background subtraction component and a visual consistency constraint. Such functional is minimized using a gradient descent method leading to a flow that deforms a set of initial curve towards the objects boundaries and an incremental estimator of their apparent motion. Promising results demonstrate the potentials of the method.

16.1 Introduction

Tracking is a well explored problem in computer vision. The application domain refers to motion analysis, video-based surveillance, medical imaging, post-production, etc. Different methodologies that exploit various visual cues were proposed to tackle the problem. Object boundaries is a suitable feature space for tracking. Boundary based methods rely on the generation of a strength image, the extraction of prominent edges and their consistent detection in the temporal domain. Model-based tracking [289, 323, 361] is the common technique used within this scenario. Given a set of training examples, one can recover a parametric representation of the target. Tracking is then performed by seeking the lowest potential of a cost function that exploits image characteristics along the parametric representation of the object. Snakes [262], deformable models/templates [316], fourier descriptors, active shapes and active appearance models [135] are parametric techniques that were investigated. Robustness is the most prominent

strength of such methods. On the other hand dealing with non-rigid objects, heavy local deformations and changes of topology are their limitations. One can consider the use of geometric flows - an alternative to the snake-driven models - to deal (to some extend) with such limitations [41]. Visual constancy constraints [245, 326] were also considered to perform tracking. To this end, a feature space and a visual representation of the object is considered in the temporal domain [132]. A matching process according to some similarity techniques is then used to recover the object positions in time [32]. For example, given an initial region of interest in the image plane (moving object), one can seek for a region across time with similar characteristics. Such task can be dealt with either using simple techniques like correlation or more advanced mathematical formulations that aim to recover dense motion flow. These methods deal fairly well with global transformations like rigid or affine motion [364]. More elaborated methods integrate prediction capabilities in the tracking process. Kalman [519] or particle filters [251, 525] are examples of prediction tools. Such techniques aim at modeling the motion of the target based on prior history, and predicting its position over time. Validation of the prediction as well as tracking adjustments according to the characteristics of the observed image are required. Upon convergence of the process, the tracking result is used to improve the performance of the prediction model. Dealing with occlusions as well as corrupted data is the most attractive property of such methods. Optical flow estimation is a complementary to tracking component of motion analysis. Implicitly or explicitly most of the tracking algorithms deal with this task when the assumption of global correspondence between the target features is considered. Such constraint [394] can be implemented using a parametric model to describe the motion of the entire object. Such models are a compromise between low complexity and acceptable solution to the correspondence problem. Rigid, affine, projective as well as quadratic models have been considered in the motion estimation process. These techniques perform well when the object respects the properties induced by the model. This constraint does not limit their applicability; approximate tracking is an acceptable solution in various applications (video-based surveillance). However, global motion models fail to capture local deformations. Complete recovery of the dense motion field is suitable technique to cope with such limitation. Pixel-wise motion estimates can improve tracking performance by dealing with local deformations. To this end, one can seek for a smooth local motion field that provides pixel-wise intensity correspondences for the object region. Visual constancy [245, 326] across frames for the object to be tracked is a well explored constraint to perform such estimation. These techniques suffer from being computational expensive. Non-rigid objects often undergo heavy local deformations from one frame to the next and in some extreme cases change the topology. Model-based tracking techniques can be very efficient when dealing with a certain degree of deformations [258]. On the other hand, complexity of these methods becomes an issue when objects undergo significant structural changes. Such cases can be dealt with very complicated modeling, a computationally expensive process. One can consider the use of non-parametric (model-free) methods to recover such complicated structures. These methods are

based on geometric flows [407] and implemented using the level set method [401], a well-known technique for tracking moving interfaces. Some of the previously described methods can perform tracking when the acquisition device (camera) is moving. In that case tracking and in particular motion estimation is a more complicated process that may involve dominant motion compensation [206]. We will assume a static camera scenario and propose a three-module unified framework for motion estimation and tracking [408]. Our approach consists of a boundary detection, a background subtraction and a visual consistency module. Snake-driven boundary detection is a standard technique to perform tracking. In the absence of prior shape knowledge one deform an initial curve towards boundaries (strong edges) while respecting some internal geometric constraints. Such techniques require a good initial guess and assume flat/uniform background to reach convergence. Background subtraction or motion/change detection [418] are components able to deal with cluttered background. The essence of these methods is to built a model that can account for the static visual properties of the scene. Then, using simple statistical tests, one can separate objects from the static background. Such methods are used in the form of region-components to detect and track object boundaries from either sides [410]. Last, but not least, visual consistency based on the optical flow constraint is used to couple tracking with the motion estimation problem. We consider linear models (affine) and we also discuss the extension of the method to perform complete recovery of the motion field. These components are integrated within a variational level-set framework. Closely related techniques with our approach can be found in [41, 82, 254, 332, 403]. The reminder of this chapter is organized as follows; in section 2 introduce the geodesic active region model, the corresponding level set variant and some notation. Motion estimation and tracking are addressed in section 3 while implementation details and some possible extensions of the method are presented in section 4. Discussion is part of section 5.

16.2 Geodesic Active Regions

The Geodesic Active Region [413] refers to a variational framework able to deal with frame partition problems in imaging and vision. It was initially introduced in [409] for supervised texture segmentation, exploited in [408] to motion estimation and tracking and extended in [411] to deal with the task of un-supervised image segmentation. In order to facilitate the introduction of the model, the bi-modal case will be considered. To this end, the following definitions and assumptions regarding *a priori* knowledge required to introduce such model are considered;

- Let I be the input image composed of two classes (h_A, h_B),
- Let $\mathcal{P}(\mathcal{R}) = \{\mathcal{R}_A, \mathcal{R}_B = \Omega - \mathcal{R}_A\}$ be a partition of the image domain Ω into two non-overlapping regions [Figure (16.1.a)],
- Let $\partial\mathcal{R}$ be the common boundaries of the $\mathcal{R}_A, \mathcal{R}_B$ partition.

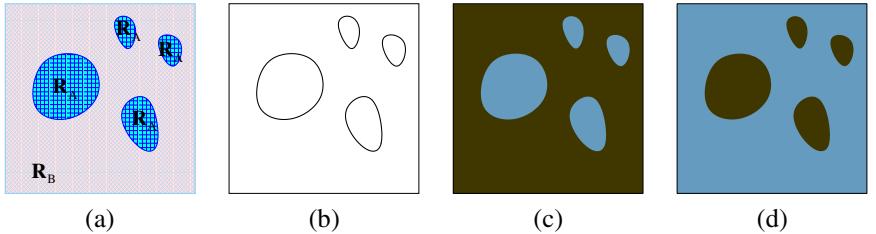


Figure 16.1. Geodesic Active Region Model: (a) the input, (b) the boundary-based information, (c) the region-based information corresponding to hypothesis h_A , [the information is proportional to the frame intensities] (d) the region-based information corresponding to hypothesis h_B [the information is proportional to the frame intensities].

- Let us assume prior knowledge on the partition position, namely the density function p_C that measures the likelihood of a given pixel being at the boundaries [Figure (16.1.b)],
- Let us assume prior knowledge on the expected region properties of h_A, h_B namely the densities $p_r(), p_B()$ that correspond to the conditional likelihood of a given intensity coming from the (h_A, h_B) hypotheses [Figure (16.1.c,16.1.d)].

16.2.1 Setting the Boundary Module

Recovering the optimal partition is equivalent with accurately extracting the boundaries between \mathcal{R}_A and \mathcal{R}_B . This can be done using the geodesic active contour model [84, 270], thus minimizing

$$E(\partial\mathcal{R}) = \int_0^1 g \left(\underbrace{p_C(I(\partial\mathcal{R}(c)))}_{\text{boundary probability}} \right) \underbrace{|\partial\dot{\mathcal{R}}(c)|}_{\text{regularity}} dc \quad (16.1)$$

where $\partial\mathcal{R}$ is a parameterization of the partition boundaries in a planar form and g is a positive, monotonically decreasing function.

16.2.2 Setting the Region Module

The visual properties of the h_A, h_B hypotheses are additional cues to perform segmentation. To this end, one would like to recover a consistent frame partition between *the observed data, the associated hypotheses and their expected properties*. One can consider the *posterior* probability as criterion to derive such partition; Let $[ps(\mathcal{P}(\mathcal{R})|I)]$ be the *posterior* partition density function with respect to $\mathcal{P}(\mathcal{R})$ given the input image I . This density function can be written

according to the Bayes rules as follows:

$$p_S(\mathcal{P}(\mathcal{R})|I) = \frac{p(I|\mathcal{P}(\mathcal{R}))}{p(I)} p(\mathcal{P}(\mathcal{R})) \quad (16.2)$$

where

- $p(I|\mathcal{P}(\mathcal{R}))$ is the *posterior* segmentation probability for the image I , given the partition $\mathcal{P}(\mathcal{R})$,
- $p(\mathcal{P}(\mathcal{R}))$ is the probability of the partition $\mathcal{P}(\mathcal{R})$ among the space of all possible partitions of the image domain,
- and $p(I)$ is the probability of having as input the image I among the space of all possible images.

If we assume that all the partitions are equally probable [$p(\mathcal{P}(\mathcal{R})) = \frac{1}{Z}$] (Z is the number of possible partitions), then one can ignore the constant terms $p(I)$, $p(\mathcal{P}(\mathcal{R}))$ and we can rewrite the density function as:

$$p_S(\mathcal{P}(\mathcal{R})|I) = p(I|\{\mathcal{R}_A, \mathcal{R}_B\}) \quad (16.3)$$

Besides, one can further consider no correlation between the region labeling where the region probabilities depend on the observation set within the region. Such assumption can further simplify the form of the posterior probability:

$$p_S(\mathcal{P}(\mathcal{R})|I) = p([I|\mathcal{R}_A] \cap [I|\mathcal{R}_B]) = p(I|\mathcal{R}_A) p(I|\mathcal{R}_B) \quad (16.4)$$

where $p(I|\mathcal{R}_A)$ is the *a posterior* probability for the region \mathcal{R}_A given the corresponding image intensities (resp. $p(I|\mathcal{R}_B)$). Last, but not least, independence on the pixel level can be considered to replace the region posterior with joint probability among the region pixels:

$$p(I|\mathcal{R}_X) = \prod_{s \in \mathcal{R}_X} p_X(I(s)) \quad (16.5)$$

where $X \in \{A, B\}$. Such assumptions can lead to the following conditional (*I*) *posterior* partition probability for $\mathcal{P}(\mathcal{R})$;

$$p_S(\mathcal{P}(\mathcal{R})|I) = \prod_{s \in \mathcal{R}_A} p_A(I(s)) \prod_{s \in \mathcal{R}_B} p_B(I(s)). \quad (16.6)$$

Optimal grouping is equivalent with recovering the partition tha corresponds to the highest posterior. The optimization of the *posterior* probability is equivalent to the minimization of the corresponding [-log()] function;

$$E(\partial\mathcal{P}(\mathcal{R})) = - \underbrace{\iint_{\mathcal{R}_A} \log \left[\underbrace{p_A(I(x, y))}_{h_A \text{ probability}} \right] dx dy}_{\mathcal{R}_A \text{ fitting measurement}} - \underbrace{\iint_{\mathcal{R}_B} \log \left[\underbrace{p_B(I(x, y))}_{h_B \text{ probability}} \right] dx dy}_{\mathcal{R}_B \text{ fitting measurement}}. \quad (16.7)$$

Such component is defined using the partition determined by the curve and aims at maximizing the *posterior* segmentation probability given the input image. It aims at separating the image regions according to their intensity properties.

16.2.3 Geodesic Active Region Objective function

The Geodesic Active Region framework consists of integrating these two different frame partition modules;

$$E(\partial\mathcal{P}(\mathcal{R})) = (1 - \alpha) \underbrace{\int_0^1 g(p_C(I(\partial\mathcal{R}(c)) | \partial\dot{\mathcal{R}}(c)) dc}_{\text{boundary term}} \\ - \sum_{X \in \{A, B\}} \alpha \underbrace{\iint_{\mathcal{R}_X} \log [p_X(I(x, y))] dx dy}_{\text{region term}} \quad (16.8)$$

where α is a positive constant that balances the contributions of the two terms $[0 \leq \alpha \leq 1]$. One can optimize this cost function using a gradient descent method and the calculus of variations. Furthermore, the resulting motion equations can be implemented using the level set technique that is intrinsic, implicit, parameter free and topology. A more elegant solution is to define the objective function directly on the level set space [607]. Such attractive formulation was considered to deal with the problems of segmentation [103, 456] and stereo. To this end, using the approximations of Dirac $[\delta_\alpha()]$ and Heaviside $[H_\alpha()]$ distributions

$$\delta_a(\phi) = \begin{cases} 0, & |\phi| > \alpha \\ \frac{1}{2\alpha} \left(1 + \cos\left(\frac{\pi\phi}{a}\right) \right), & |\phi| < \alpha \end{cases}$$

$$H_\alpha(\phi) = \begin{cases} 1, & \phi > \alpha \\ 0, & \phi < -\alpha \\ \frac{1}{2} \left(1 + \frac{\phi}{\alpha} + \frac{1}{\pi} \sin\left(\frac{\pi\phi}{a}\right) \right), & |\phi| < \alpha \end{cases}$$

one can define a level set variant of the geodesic active contour model using level set representation Φ ;

$$E(\Phi) = (1 - \alpha) \iint_{\Omega} H_\alpha(\Phi) g(p_C(I)) |\nabla \Phi| d\Omega \\ - \alpha \iint_{\Omega} H_\alpha(\Phi) \log [p_A(I)] d\Omega - \alpha \iint_{\Omega} (1 - H_\alpha(\Phi)) \log [p_B(I)] d\Omega \quad (16.9)$$

that refers ta 2-partition variational framework that integrates boundary attraction component with a region homogeneity term while being constrained to respect some internal properties.

16.2.4 N – Partition Geodesic Active Regions & Tracking

Such model can be extended to deal with the N -Partition case and in particular with the motion estimation and tracking of N objects for a given time instant t ;

- N moving objects are visible in a sequence of T frames $[I_1, \dots, I_T]$,
- N curves represented using level set functions $\Phi_i \in [\Phi_1, \dots, \Phi_N]$ are used to track these objects,
- $N \times T$ parametric motion models describe the transformation of the objects from one frame to the next; $A^{i,t}$ is associated with object i and describes its 2D apparent motion between frames t and $t + 1$.

Motion estimation and tracking are equivalent with recovering at each frame $[t]$ the objects positions and the corresponding motion models A^{it} between $t - 1$ and t given the tracking result at $t - 1$.

$$\begin{aligned} E(\Phi_1, \dots, \Phi_N) = & (1 - \alpha) \sum_{i=1}^N \iint \delta_\alpha(\Phi_i) b_i(I) |\nabla \Phi_i| d\Omega + \\ & \alpha \sum_{i=1}^N \iint H_\alpha(\Phi_i) r_i(I) d\Omega + \underbrace{\alpha \iint \left[\prod_{i=1}^N (1 - H_\alpha(\Phi_i)) \right] r_0(I) d\Omega}_{\text{background attraction component}} \end{aligned} \quad (16.10)$$

where b_i can be boundary-driven tracking modules and r_i regions descriptors for the moving objects. Some properties of static background are as well considered (r_0). Our objective will be to define boundary and region-based functional that when optimized can provide a solution to the optical flow estimation and the tracking problem.

16.3 Optical Flow Estimation & Tracking

16.3.1 Boundary & Smoothness Component

Strong discontinuities between the moving objects and the static background - often assumed to be homogeneous or with limited structure - is a well explored tracking assumption. To this end, the outcome of standard edge-detection processing techniques [77, 151] was used within snake-driven methods to perform tracking. Within the level set formulation, one can consider the use of geodesic active contour to track the boundaries of the moving objects;

$$E_{boundary}(\Phi_1, \dots, \Phi_N) = \sum_{i=1}^N \iint_{\Omega} \delta_\alpha(\Phi_i) g(|\nabla I(t)|) |\nabla \Phi_i| d\Omega \quad (16.11)$$

for a given frame t . One can consider replacing the attraction term with more sophisticated terms that can better account for the object boundaries [410]. Such

objective function involves N level set functions (one for each object) to perform tracking. The calculus of variations and a gradient descent method can be used to obtain a minimum of the objective function;

$$\frac{d}{d\tau} \Phi_i = \delta_a(\Phi) \operatorname{div} \left(g(|\nabla I(t)|) \frac{\nabla \Phi_i}{|\nabla \Phi_i|} \right) \quad (16.12)$$

that deforms an initial contour towards the object boundaries while being constrained by the curvature. Such flow is single-directional and reaches the object boundaries from one side. Optimal results are recovered when the initial curve is either interior to the object or encloses it. One can overcome this limitation by considering directional data terms like the ones introduced in [280, 415, 543, 585].

16.3.2 Background Subtraction tracking

Background subtraction [178, 373, 496] and change detection [418] are basic components of motion analysis for static sequences. The basic assumption behind these modules is that a representation of the background can be recovered and maintained. Statistical tests can be employed to separate the pixels that belong to the moving objects from the static ones. The outcome of this process can be used then as feature space to perform tracking [412]. Such techniques can deal with cluttered background. However, generating a background reference frame is not always feasible. We assume that such representation can be recovered in a gradual fashion from the tracking process using positive feedback avoid perturbing the background statistics due to moving objects. Such techniques require some prior history to create a reasonable representation of the background. Parametric [496] as well as non-parametric [178] statistical modeling is an efficient way to represent the static scene and recover detection maps. Global modeling of the (inter) frame-difference is a computational efficient method to detect moving objects [418] that does not require significant prior history. Static versus and non-static hypothesis can be represented using two zero-mean exponential functions. Such distributions can be derived from the empirical distribution of the difference frame; Let B be the background reference frame, and $D(t)$ the difference frame at moment t ;

$$D(x, y; t) = B(x, y; t) - I(x, y; t)$$

An example of the empirical distribution $H(D(t))$; p_D is shown in [Figure (16.2)]. One can assume that such distribution is a mixture model of two components, one that corresponds to the static hypothesis p_{st} (noise) and one that corresponds to the mobile hypothesis p_{mb} . Such assumption can lead to the following continuous form for the observed distribution:

$$p_D(d) = P_{st} p_{st}(d) + P_{mb} p_{mb}(d) \quad (16.13)$$

where P_{st}, P_{mb} are the a priori probabilities. The use of two exponential functions has been efficiently considered in the past [418]. Such simplification is valid when the static hypothesis is far more popular than the mobile one. Otherwise,

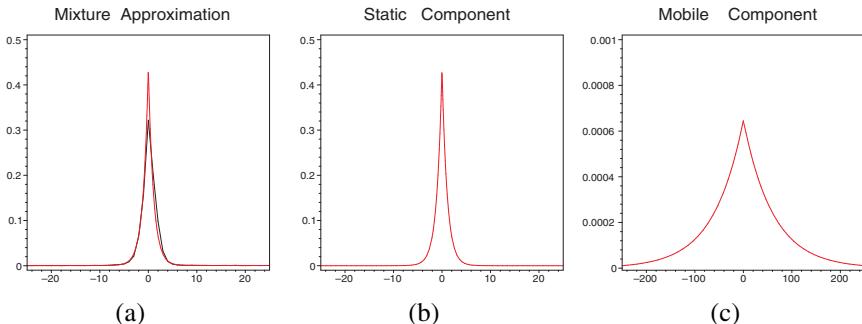


Figure 16.2. Differene frame analysis for background subtraction using a Maximum Likeli-hood. (a) Input Distribution (dark), mixture approximation (bright), (b) Static Component, (c) Mobile Component.

more complicated models that can account for multiple populations within the mobile component are to be considered. Maximum Likelihood principles can be used to recover the parameters of this mixture model as shown in Figure (16.2). The analysis and modeling of the difference frame provides a fast and reliable way to perform background subtraction through the densities of the two conflicting classes (p_{st}, p_{mp}). One can expect that the moving objects are composed of mobile pixels. Therefore, tracking is equivalent with grouping pixels that do not refer to the background hypothesis. Furthermore, the density of the static hypothesis can be used to define a grouping metric for the background pixels.

Binary decisions from the background subtraction process can cause non-reversible errors in the tracking process. Decisions taken using Bayes rule compare the probabilities of the two conflicting hypotheses given the input image. Cases where both hypotheses do equally well or bad can produce important errors through a binary classification module. One can overcome this limitation by introducing a background subtraction tracking term in the form of a continuous region-defined component [410];

$$E_{detection}(\Phi_1, \dots, \Phi_N) = - \sum_{i=1}^N \iint_{\Omega} H_{\alpha}(\Phi_i) \log(p_{mb}(I(t))) d\Omega \\ - \iint_{\Omega} \left[\prod_{i=1}^N (1 - H_{\alpha}(\Phi_i)) \right] \log(p_{st}(I(t))) d\Omega \quad (16.14)$$

This terms assumes that moving objects have different visual properties compared to the static background. In areas where such objects are present, the pdf for the object (mobile) hypothesis is much stronger than the static one. Similar interpretation is valid for the background subtraction component. One can replace the $-\log$ with more appropriate terms that exhibit stable behavior. A gradient descent method is a naturally way to recover a solution that minimizes the previously



Figure 16.3. Background Subtraction Tracking for Oxford Stereo Sequence. The results are presented in raster-scan format. Temporal initialization is done by simple projection of tracking result in the previous frame. Such technique requires some minimal overlap between the object positions from one frame to the next.

defined objective function;

$$\frac{d}{d\tau} \Phi_i = \delta_\alpha(\Phi_i) \log \left(\frac{p_{st}(I(t))}{p_{mb}(I(t))} \right) \quad (16.15)$$

where we assume the absence of occlusions for the moving objects. The interpretation of such flow is quite clear; the evolving shrinks when located on the background and expands otherwise (inside a moving object). One can consider this term as an adaptive balloon force. In [410], boundary and background subtraction components were combined to perform tracking [Figure (16.3)] with encouraging results. In the absence of background model, similar analysis on the inter-frame difference frame can be used to separate the static from the mobile image components.

16.3.3 Visual Consistency

Visual consistency through motion recovery for the moving objects in the temporal domain is a standard constraint to perform tracking. Most of the 2-D motion estimation approaches are based on the measurement of the apparent motion of the intensity patterns over time [7, 31]. Such methods assume that the image brightness along the motion trajectory is constant [245] thus is not always the case. Changes on the object pose, global/local illumination conditions, etc. violate the brightness constancy constraint, a core assumption during flow recovery. Furthermore, the motion vectors satisfying the image brightness constraint are not unique and external factors like surface reflections properties, sensor noise and distortions can cause changes not related with motion. Last, but not least, physical models that are explicitly or implicitly embedded in the estimation process, are often inadequate for accounting accurately for the image brightness formation. Motion can be determined either using global motion models or by considering correspondences pixel-wise. Global motion models assume the existence of a valid transformation for the entire object. Opposite to that, local motion (optical flow)

is estimated independently pixel-wise. Robustness is the main advantage of global motion components while their inability to deal with local deformations is a strong limitation. One can claim that for a sufficiently small field of view and planar moving objects, the image velocity field (projection of the real 3D motion) can be approximated by a linear model $A(x, y) = (A_x(x, y), A_y(x, y))$ while in the absence of motion (static background), the brightness remains constant in time;

$$\begin{cases} I(x, y; t) = I(x, y; t + 1); \text{ background} \\ I(x, y; t) = I((x, y) + A(x, y); t + 1); \text{ planar object} \end{cases} \quad (16.16)$$

Parametric motion estimation [36, 48, 364, 559] is a computationally efficient method to recover optical flow. Low complexity and robust estimates are their strengths. One can assume that when the model assumptions are satisfied by the moving object, reasonable motion estimates can be recovered. On the other hand, such approaches cannot deal with local object deformations or objects that exhibit depth discontinuities (non-planar). One can consider transformations that involve limited number of parameters like rigid, or more complicated ones that can account for more complex scenes and motions. In latest case the use of affine, homographic or quadratic models can be used to approximate the motion of the target. We consider affine transformations [408], a compromise between low complexity, stability and fairly good approximation of the motion field. Such model consists of six motion parameters:

$$A(x, y) = \begin{bmatrix} A^x(x, y) \\ A^y(x, y) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_{13} \\ a_{23} \end{bmatrix} \quad (16.17)$$

The most common way to derive motion estimates is the optical flow constraint [245]. In the absence of global illumination changes, one can assume that the observed intensities at the position (x, y) in the frame $t + 1$, and at $(x, y) + A(x, y)$ in the frame t are the same. This constraint relies on minimizing the *sum of squared differences* (SSD)

$$E(A) = \iint_{\Omega} (I(A; t + 1) - I(t))^2 d\Omega \quad (16.18)$$

Global (affine) motion models introduce certain limitations in the estimation process. Local deformations as well as depth changes do not satisfy the brightness constraint and can perturb the estimation of the motion parameters. At the same time, least estimators (sum of square differences) are sensitive to the presence of noise and outliers [246]. A simple way to deal with such limitation is to consider local deformations as outliers of the estimation process and ignore them during the estimation of the motion parameters. Towards this direction within the considered application one would like recover N parametric motion models A_i that create visual correspondences for each object in the temporal domain for the moving targets. Furthermore one can assume the absence of motion for the static

background;

$$\begin{aligned}
 E_{motion}((\Phi_1, A_1), \dots, (\Phi_N, A_N)) = & \\
 & \underbrace{\iint_{\Omega} H_{\alpha}(\Phi_i) \rho(I(A_i; t+1) - I(t)) d\Omega}_{\text{moving object visual consistency}} \\
 & + \underbrace{\iint_{\Omega} \left[\prod_{i=1}^N (1 - H_{\alpha}(\Phi_i)) \right] \rho(I(t+1) - I(t)) d\Omega}_{\text{background visual consistency}}
 \end{aligned} \tag{16.19}$$

where ρ is a bounded error function. We consider the *fair* error estimator given by

$$\rho_{fair}(r) = c^2 \left[\frac{|r|}{c} - \log \left(1 + \frac{|r|}{c} \right) \right] \tag{16.20}$$

Such objective function couples motion estimation and tracking. The unknown variables are the motion parameters and the targets position at frame $t+1$. This functional implicitly assumes the absence of occlusions between the moving objects. In order to interpret the proposed term, we will consider the motion transformation known. In that case, the lowest potential of the objective function $E_{motion}((\Phi_i, A_1), \dots, (\Phi_N, A_N))$ refers to an image region composed of pixels that satisfy the visual constancy constraint with the target position in the previous frame. On the other hand, for known objects positions, the lowest potential of the objective function corresponds to an optimal motion transformation A_I that creates pixel-wise visual correspondences for the target in the temporal domain.

One can optimize this functional with respect to the targets positions (Φ_i) and the optical flow estimates (A_i) using a gradient decent method;

$$\begin{aligned}
 \frac{d}{d\tau} \Phi_i &= \delta_{\alpha}(\Phi_i) (\rho(I(t+1) - I(t)) - \rho(I(A_i; t+1) - I(t))) \\
 \frac{d}{d\tau} a_{kl}^i &= \iint_{\Omega} H_{\alpha}(\Phi_i) \psi(I(A_i; t+1) - I(t)) \\
 &\quad \left(\frac{\partial I(A_i; t+1)}{\partial x}, \frac{\partial I(A_i; t+1)}{\partial y} \right) \left(\frac{\partial A_i^x}{\partial a_{kl}^i}, \frac{\partial A_i^y}{\partial a_{kl}^i} \right)
 \end{aligned} \tag{16.21}$$

where $\phi(r) = \rho'(r)$ is the derivative of ρ known as influence function and a_{kl} is the (k, l) parameter of the motion model $[A_i]$. One can interpret the obtained motion equation as follows;

- ***i* Level Set Flow:** a force that aims to move the i curve towards the direction that decreases the visual correspondence error. A relative comparison between the background and the motion hypotheses is used to determine the propagation direction. If the error produced by the static hypothesis is greater than the one of the object (given the current estimation of the mo-

tion), then the contour expands to include this pixel in the object hypothesis and vice-versa.

- ***i Motion Estimation Flow:*** an iterative mechanism to update the *i* motion estimates given the current position of the object. Such updates are driven by a term that tends to improve the quality of visual correspondences between the current and the previous frame position of the objects.

Tracking and motion estimation can be jointly recovered in an iterative manner. Encouraging experimental results¹ shown in [Figure (16.4)] demonstrate the potentials of performing tracking and estimating the optical flow simultaneously [406]. However, one can claim that the use of a gradient descent method to estimate the motion parameters is not the most prominent solution. Such parameters have different rate of update and the use of the same time step can cause discrepancies on the estimation. The system becomes quite unstable due to the different convergence rates of the motion model components. Such limitation can be addressed by updating the motion estimates using a closed form solution [408]. To this end, the motion estimation task is reformulated as follows; given a current estimate of the motion model A , recover a complementary affine model

$$\Delta A(x, y) = \begin{bmatrix} \Delta A^x(x, y) \\ \Delta A^y(x, y) \end{bmatrix} = \begin{bmatrix} \delta a_{11} & \delta a_{12} \\ \delta a_{21} & \delta a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \delta a_{13} \\ \delta a_{23} \end{bmatrix} \quad (16.22)$$

that when combined with the current estimates improves the matching between the object positions in two consecutive frames. Using the notation introduced earlier, one can re-define the motion-related component of the objective function;

$$\begin{aligned} E_{\text{motion}}((\Phi_i, \Delta A_1), \dots, (\Phi_N, \Delta A_N)) = & \\ & \iint_{\Omega} H_{\alpha}(\Phi_i) \rho(I(A_i + \Delta A_i; t+1) - I(t)) d\Omega \\ & + \iint_{\Omega} \left[\prod_{i=1}^N (1 - H_{\alpha}(\Phi_i)) \right] \rho(I(t+1) - I(t)) d\Omega \end{aligned} \quad (16.23)$$

The optimal solution of such objective function with respect to the components of ΔA_i can now be recovered using the following constraints:

$$\begin{aligned} n \in [1, 2], \quad m \in [1, 3] \\ \frac{\partial}{\partial \delta a_{mn}^i} E_{\text{motion}}((\Phi_i, \Delta A_1), \dots, (\Phi_N, \Delta A_N)) = 0 \end{aligned} \quad (16.24)$$

leading to a linear system with respect to $[\delta a_{11}, \delta a_{12}, \delta a_{13}]$ and $[\delta a_{21}, \delta a_{22}, \delta a_{23}]$ that has a solution in a closed form. We perform this motion estimation step until the motion model converges. Such mechanism can be used to recover the optimal estimates of the motion model according to the latest position of the object as

¹To this end, an additional component that accounts for prior shape knowledge [444] of the structure of interest was also integrated with the proposed framework [406].

defined from the tracking module (Φ_i). One can consider performing such motion correction step after each iteration. Motion correction is required when the object

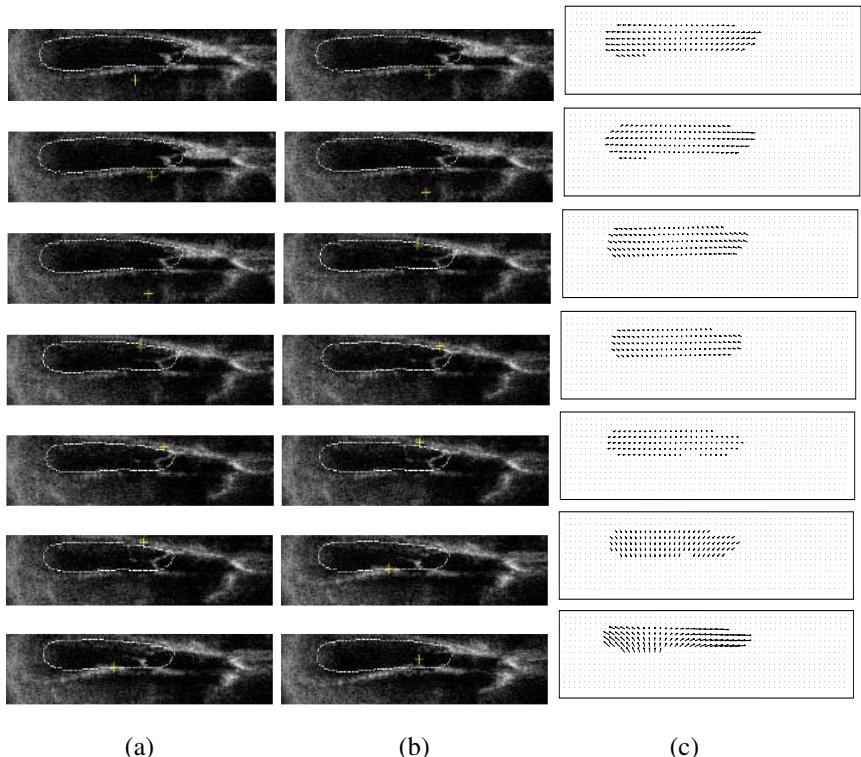


Figure 16.4. Cardiac Segmentation/Tracking for Ultrasonic Images presented in raster-scan format. Shape-drive constraints are required to deal with the corrupted and noisy data. (a) segmented t frame, (b) segmented $t + 1$ frame, (c) motion estimation (flow) for the structure of interest between frame t and frame $t+1$ up-scaled four times for demonstration purposes.

position (tracking) changes significantly. The use of the proposed tracking framework will gradually update the object position from one iteration to the next until tracking is optimized. Therefore, updating the motion model in each iteration is not necessary [408].

16.3.4 Implementation Issues

The proposed framework can be used to detect, track and recover the trajectories of rigid as well as non-rigid objects. The number of moving objects can be determined either by the user or by processing the first frame². To this end, the background subtraction module can be used. Upon convergence of the background subtraction flow, a connected component analysis method can determine

²Similar process has to be considered to deal with objects that appear in the camera view in some later time.

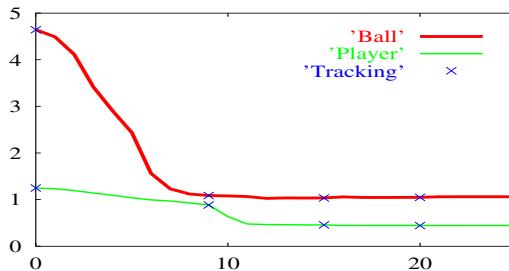


Figure 16.5. Motion Estimation: Mean Square Error. The X-axis of the graph corresponds to time (iteration number), while the Y-axis, to the mean square error. Crosses denote the moments that curve position is updated (tracking).

the number of moving objects in the scene. Then, a level set function can describe the motion of each object that is a computational intensive procedure. An elegant way to reduce complexity without reducing the model capacity is to couple the level set functions as proposed in [551] where N level set functions can be used to detect and track 2^N objects. In case of un-occluded objects computational complexity can be reduced by considering a single level set representation that consists of multiple non-connected components (objects). Optimization of the proposed framework can be done in two different ways. Motion estimation and tracking can be either recovered simultaneously through a gradient descent method [Figure (16.4)] or separated and determined in an iterative process that alternates between motion recovery and tracking. Stability on the estimates is the main reason of using such a de-coupled approach. Promising experimental results demonstrate the potentials of such selection for different outdoor sequences with respect to the motion estimation task [Figure (16.5)]³ and the tracking [Figure (16.6,16.7,16.8)]. The incremental estimation of the apparent motion is a promising solution to the optical flow recovery problem. However, it suffers from being a global method that considers a linear model to recover the object motion. The use of robust estimators will lead to reasonable handling of the non-rigid parts. Hopefully within the estimation process such parts will be considered as outliers and will not perturb the motion estimates for the rigid part of the object. On the other hand, such errors will propagate to the tracking component of our technique and may cause certain discrepancies.

16.4 Complete Recovery of the Apparent Motion

Complete recovery of the apparent motion is a most prominent solution when dealing with non-rigid objects. In that case, we assume that existence of a

³This graph represents the visual consistency term of the objective function for the first two frames of the Soccer sequence that consists of two objects (soccer player, soccer ball) [Figure (16.5)].

$(U(x, y), V(x, y)) = (u, v)$ field define in the image plane as follows;

$$\begin{cases} (x, y) \in \Omega : (U(x, y), V(x, y)) \\ I(x, y; t) = I(u(x, y), v(x, y); t + 1) \end{cases} \quad (16.25)$$

The constant brightness assumption can be considered to define an objective function that can recover the vector field (u, v) in the pixel level;

$$E(U, V) = \iint_{\Omega} \rho(I((u, v); t + 1) - I(t)) d\Omega \quad (16.26)$$

However the recovery of the optical flow using the above constraint is an ill-posed problem. The number of unknown variables is larger than the number of constraints. A common technique to overcome this limitation is to consider additional smoothness constraints on the flow;

$$E(U, V) = \iint_{\Omega} \rho(I((u, v); t + 1) - I(t)) + \epsilon \zeta (|U| + |V|) d\Omega \quad (16.27)$$

where ζ is a regularization term that penalizes discontinuities on the optical flow and ϵ a positive constant that balances importance of the two terms.

This framework can be used to recover dense optical flow and perform tracking. Opposite to the affine case, theoretically such functional can deal with the case of moving camera as well. However, such simplistic estimation component may fail to deal with complex 3D scenes and do not account for discontinuities on the optical flow that are quite natural along the object boundaries. This limitation can be easily addressed by eliminating the flow smoothness constraint in the vicinity of the object boundaries. Last, but not least, one can replace this term with more elaborated optical flow terms under the condition that they can be expressed as regional terms within the geodesic active region model.

16.5 Discussion

In this chapter we have proposed an elegant variational framework for optical flow estimation and tracking. This framework was derived from the Geodesic Active Region model and exploits various information modules (boundary, background subtraction, visual constancy) and couples motion estimation and tracking through an objective function that involves both unknown parameters. The proposed formulation can deal with important shape deformations and changes of the object topology. Future directions of our research involve tracking and motion estimation for sequences of images acquired by a moving camera. Furthermore, partial reconstruction of the camera scene using motion information is a promising direction to investigate.

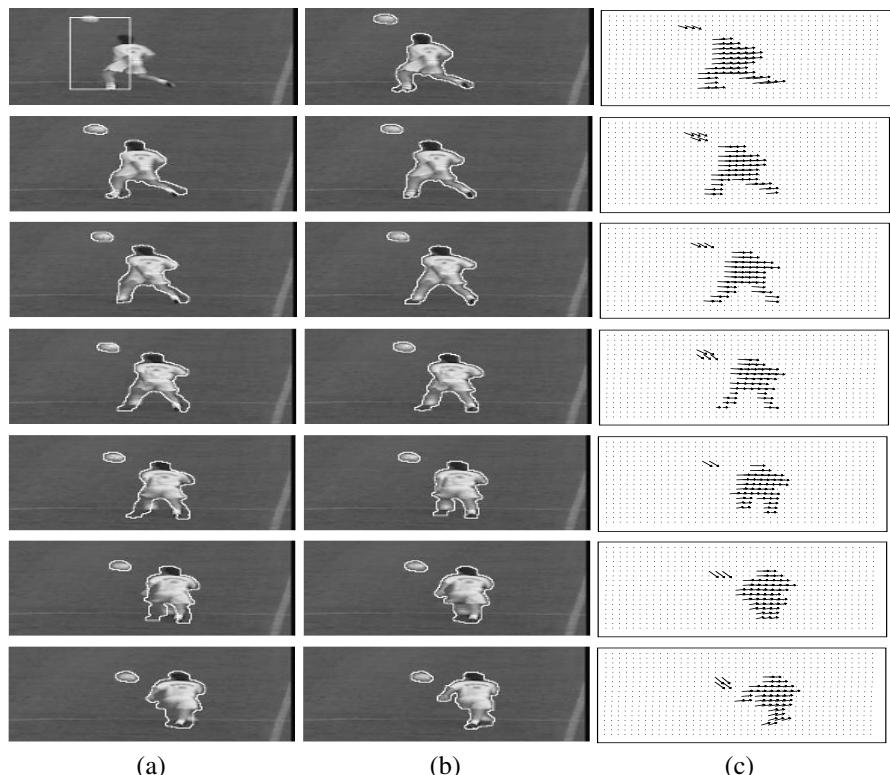


Figure 16.6. Soccer Sequence:(a) Initial Curve , (b) Final Curve, (c) Motion Estimation.

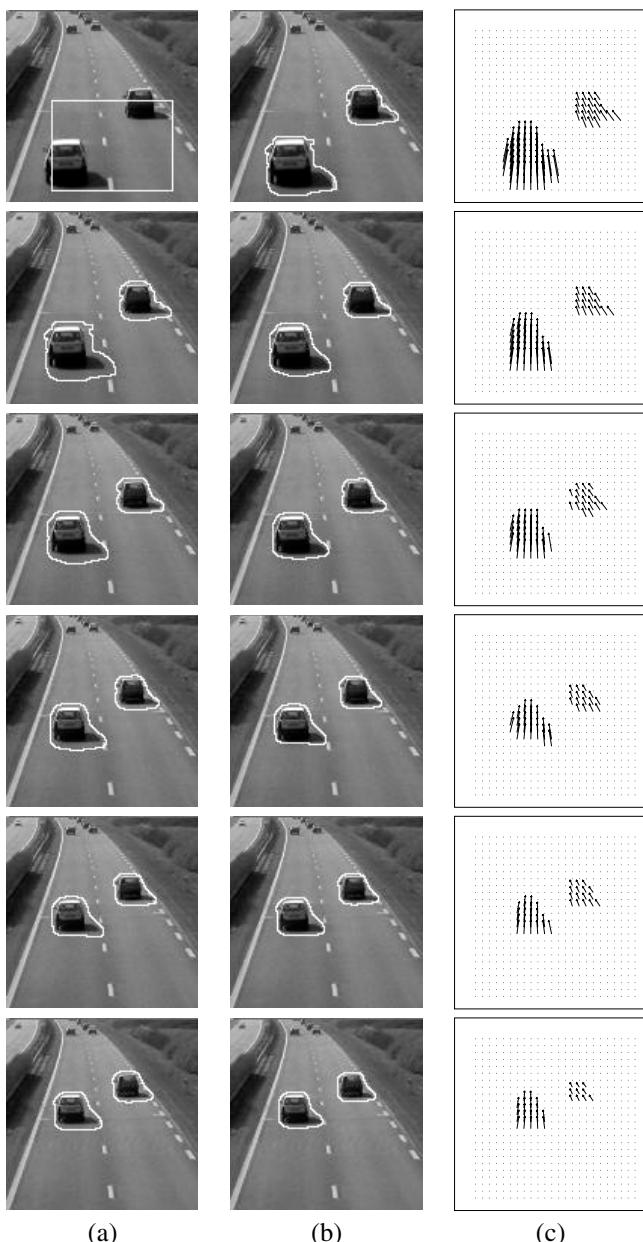


Figure 16.7. Highway Sequence:(a) Initial Curve , (b) Final Curve, (c) Motion Estimation.

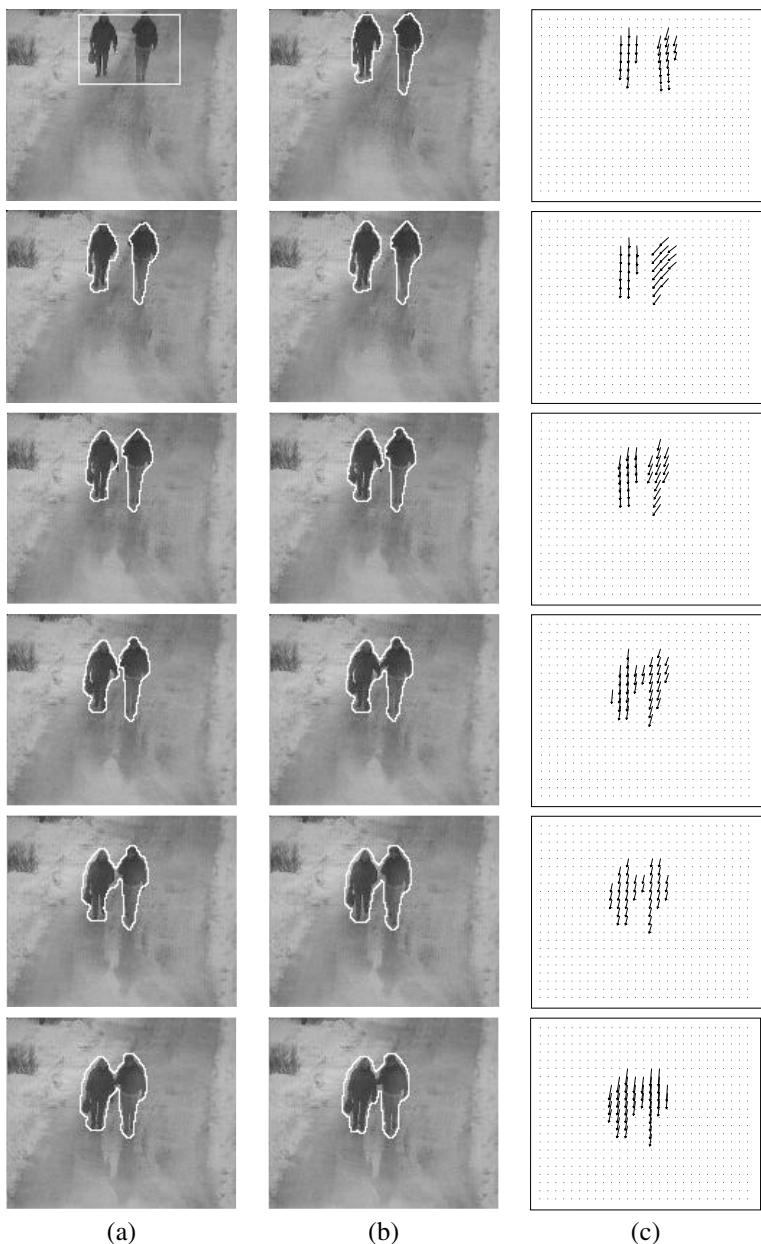


Figure 16.8. Swedish Pedestrian Sequence:(a) Initial Curve , (b) Final Curve, (c) Motion Estimation.

17

Region Matching and Tracking under Deformations or Occlusions

Stefano Soatto, Anthony Yezzi and Alessandro Duci

Abstract

We present a variational approach to the problem of registering non-equivalent shapes with missing parts. Registration is achieved through the evolution of partial differential equations that simultaneously estimates the shape of the missing region, the underlying “average complete shape” and the collection of group elements (Euclidean or affine) performing the registration. Our techniques apply both to shapes, for instance represented as characteristic functions (binary images), and to grayscale images, where all intensity levels evolve simultaneously in a system of partial differential equations.

17.1 Introduction

Consider a sheet of paper falling. If it were a rigid object, one could describe its *motion* by providing the coordinates of one particle and the orientation of an orthogonal reference frame attached to that particle. That is, 6 numbers would be sufficient to describe the object at any instant of time. However, being a non-rigid object, in order to describe it at any instant of time one should really specify the trajectory of each individual particle on the sheet [21]. That is, if γ_0 represents the initial collection of particles, one could provide a function f that describes how the entire set of particles evolves in time: $\gamma_t = f(\gamma_0, t)$. Indeed, if each particle can move independently, there may be no notion of “overall motion,” and a more appropriate description of f is that of a “*deformation*” of the sheet. That includes as a special case a rigid motion, described collectively by a rotation matrix $R(t) \in SO(3)$ and a translation vector $T(t) \in \mathbb{R}^3$, so that $\gamma_t = f(\gamma_0, t) = R(t)\gamma_0 + T(t)$ with $R(t)$ and $T(t)$ independent of the particle in γ_0 . In practice, however, that is *not* how one usually describes a sheet of paper falling. Instead, one may say that the sheet is “moving” downwards along the vertical direction

while “deforming.” That is, even when the object is not rigid, one may still want to retain a notion of overall, or “global,” motion, and describe departures from rigidity as a “deformation.” This stems from one’s desire to capture the fact that the sheet of paper is somehow moving as a whole, and its particles do not just behave like a swarm of bees.

But what does it even mean for a deforming object to be “moving”? From a mathematical standpoint, rigorously defining a notion of motion for deforming objects presents a challenge. In fact, if we describe the deformation f as the composition of a rigid motion $(R(t), T(t))$ and a “deformation” function $h(\cdot, t)$, so that $\gamma_t = h(R(t)\gamma_0 + T(t), t)$, we can always find infinitely many different choices $\tilde{h}(\cdot, t), \tilde{R}(t), \tilde{T}(t)$ that give rise to the same overall deformation f : $\gamma_t = f(\gamma_0, t) = h(R(t)\gamma_0 + T(t), t) = \tilde{h}(\tilde{R}\gamma_0 + \tilde{T}(t), t)$ by simply choosing $\tilde{h}(\gamma, t) \doteq h(R\tilde{R}^T(\gamma - T) + T, t)$ for any rigid motion (\tilde{R}, \tilde{T}) . Therefore, we could describe the motion of our sheet with (R, T) as well as with (\tilde{R}, \tilde{T}) , which is arbitrary, and in the end we would have failed in defining a notion of “motion” that is unique to the event observed.

So, how can we define a notion of motion for a deforming object in a mathematically sound way that reflects our intuition? For instance, in Fig. 17.5, how do we describe the “motion” of a jellyfish? Or in Fig. 17.4 the “motion” of a storm? In neuroanatomy, how can we “register” a database of images of a given structure, say the corpus callosum (Fig. 17.7), by “moving” them to a common reference frame?

All these questions ultimately boil down to an attempt to *separate the overall motion from the more general deformation*. Before proceeding, note that this is not always possible or even meaningful. In order to talk about the “motion” of an object, one must assume that “*something*” of the object is preserved as it deforms. For instance, it may not make sense to try to capture the “motion” of a swarm of bees, or of a collection of particles that indeed all move independently. In cases where such a notion meaningful, though, we wish to capture it both mathematically and intuitively.

Consider now a further extension of the above scenario, where different images of the same scene, taken for instance from a moving camera, have been corrupted, so that an entire part is missing. This problem arises, for instance, in the presence of occlusions when one or more parts of an object may be absent in each view, and in “movie inpainting” where one or more frames are damaged and one wants to “transfer” adjacent frames to fill in the damaged part. We consider a simplified version of the problem, where we have a compact region in each image i , bounded by a closed planar contour, γ_i , and a region of the image, with support described by a characteristic function χ_i , is damaged. We do not know a-priori what the region χ_i is, and we do not know the transformation mapping one image onto the other. However, we make the assumption that such a transformation can be well approximated by a finite-dimensional group g_i , for instance the affine or the projective group. In addition, we do not know the value of the image in the missing region. Therefore, given a sequence of images, one has to simultaneously

infer the missing regions χ_i as well as the transformations g_i and the occluded portions of each contour γ_i .

We now proceed to making the discussion above precise in a formal setting. We first propose our definitions for the simplest case where the “object” is a one-dimensional contour in Sect. 17.2, and later extend it to more general objects and more general notions of motion. We then give a detailed derivation of an algorithm to compute shape and motion in Sect. 17.3, which also results in an efficient way to compute the distance between planar shapes. When an object is being tracked over time, the notion of shape average is extended to that of a “moving average” (Sect. 17.4). We then extend these results from geometric shapes to images, resulting in their simultaneous approximation and registration in Sect. 17.5. We then proceed to describe the case of matching with missing parts in Section 17.6. Finally, in Sect. 17.7, we show results on a representative set of synthetic shapes as well as on real image sequences that illustrate our approach.

Before all that, in the next two sections we give a succinct description of the vast literature on shape and motion and how it relates to the contributions of our research.

17.1.1 Prior related work

The study of shape spans at least a hundred years of research in different communities from mathematical morphology to statistics, geology, neuroanatomy, paleontology, astronomy etc. Some of the earlier attempts to formalize a notion of shape include D’Arcy Thompson’s treatise “Growth and Form” [523], the work of Matheron on “Stochastic Sets” [347] as well as that of Thom, Giblin and others [522, 211].

In statistics, the study of “Shape Spaces” was championed by Kendall, Mardia and Carne among others [264, 305, 79, 339]. These tools have proven useful in contexts where distinct “*landmarks*” are available, for instance in comparing biological shapes with N distinct “parts.” However, comparing objects that have a different number of parts, or objects that do not have any distinct landmark, is elusive under the aegis of statistical shape spaces. Although the framework clearly distinguishes the notion of “motion” from the “deformation”, the analytical tools are essentially tied to the point-wise representation. One of our goals in this paper is to extend the theory to smooth curves and surfaces that do not have distinct “landmarks.”

In computer vision, a wide literature exists for the problem of “matching” or “aligning” objects based on their images, and space limitations do not allow us to do justice to the many valuable contributions. We refer the reader to [544] for a recent survey. A common approach consists of matching collections points organized in graphs or trees (e.g. [299, 196]). Belongie et al. [34] propose comparing planar contours based on their “shape context.” See also [121, 171]. Koenderink [286] is credited with providing some of the key ideas involved in formalizing a notion of shape that matches our intuition. However, Mumford has critiqued cur-

rent theories of shape on the grounds that they fail to capture the essential features of perception [385].

“Deformable Templates,” pioneered by Grenander [221], do not rely on “features” or “landmarks;” rather, images are directly deformed by a (possibly infinite-dimensional) group action and compared for the best match in an “image-based” approach [601]. There, the notion of “motion” (or “alignment” or “registration”) coincides with that of deformation, and there is no clear distinction between the two [35]. Grenander’s work sparked a current that has been particularly successful in the analysis of medical images, for instance [222]. We would like to retain some of the power and flexibility of deformable templates, but within this framework mark a clear distinction between “motion” and “deformation.”

Another line of work uses variational methods and the solution of partial differential equations (PDEs) to model shape and to compute distances and similarity. In this framework, not only can the notion of alignment or distance be made precise [25, 598, 371, 279, 456], but quite sophisticated theories that encompass perceptually relevant aspects, can be formalized in terms of the properties of the evolution of PDEs (e.g. [282, 276]). We use region-based cost functionals, similarly to what Chan and Vese have done for the problem of segmentation of static images [100], and we use numerical methods derived from Osher and Sethian [401]. None of these approaches, however, distinguishes a notion of motion that is separate from the deformation; the evolution of shapes is driven by energy and regularization terms, rather than by the action of a finite-dimensional group of transformations. We would like to extend this framework to evolve contours simultaneously with respect to a group element and a generic deformation, and try to infer both from data and render them separate or “independent” in a precise sense.

The “alignment,” or “registration,” of curves has also been used to define a notion of “shape average” by several authors (see [310] and references therein). The shape average, or “prototype,” can then be used for recognition in a nearest-neighbor classification framework, or to initialize image-based segmentation by providing a “prior.” Leventon et al. [310] perform principal component analysis in the aligned frames to regularize the segmentation of regions with low contrast in brain images. However, the alignment is performed ad-hoc by pre-processing the images, rather than posing it as part of the inference problem. Errors in the pre-processing stage can never be compensated, and the “registration” cannot be adapted as the original contours evolve towards their average. Similarly, [591] performs the joint segmentation of a number of images by assuming that their registration (stereo calibration) is given. We wish to extend these approaches to situations where the calibration/registration is not known a-priori. A somewhat complementary work is [595], where objects, assumed to be identical except for a group action, are registered by minimizing a region-based cost functional. We wish to extend that approach to cases where the objects are not equivalent (modulo the group) but undergo added deformations.

Also related to this paper is the recent work of Paragios and Deriche, where active regions are tracked as they “move.” In [412] the notion of motion is not made

distinct from the general deformation, and therefore what is being tracked is a general (infinite-dimensional) deformation. Our aim is to define tracking as a trajectory on a finite-dimensional group, despite infinite-dimensional deformations. Substantially different in methods, but related in the intent, is the work on stochastic filters for contour tracking and snakes (see [50] and references therein). There, however, what is being tracked over time is a general deformation (although finitely parametrized via splines or other parametric descriptions), rather than a (group) motion. Therefore, the end product of these tracking algorithms is not a trajectory on a finite-dimensional group, but a generic sequence of deformations.

17.2 Defining motion and shape average

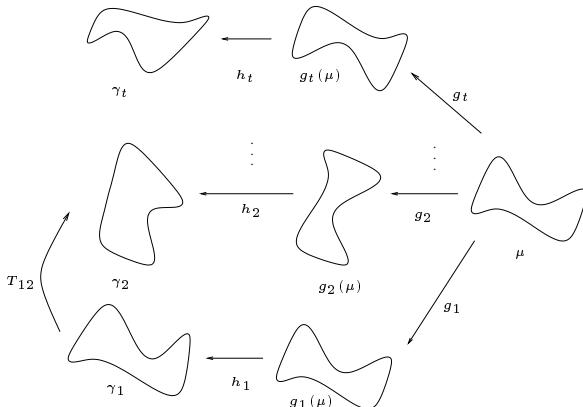


Figure 17.1. A model (commutative diagram) of a deforming contour.

The key idea underlying our framework is that the notion of *motion* throughout a deformation is very tightly coupled with the notion of *shape average*. In particular, if a deforming object is recognized as moving, there must be an underlying object (which will turn out to be the shape average) moving with the same motion, from which the original object can be obtained with minimum deformations. Therefore, we will model a general deformation as the composition of a group action g on a particular object, on top of which a local deformation is applied. The shape average is defined as the one that minimizes such deformations.

Let $\gamma_1, \gamma_2, \dots, \gamma_n$ be n “shapes” (we will soon make the notion precise.) Let the map between each pair of shapes be T_{ij}

$$\gamma_i = T_{ij}\gamma_j, \quad i, j = 1 \dots n. \quad (17.1)$$

It comprises the action of a group $g \in G$ (e.g. $G = SE(2)$) and a more general transformation h that belongs to a pre-defined class \mathcal{H} (for instance diffeomorphisms). The deformation h is not arbitrary, but depends upon another “shape” μ , defined in such a way that

$$\gamma_i = h_i \circ g_i(\mu), \quad i = 1 \dots n. \quad (17.2)$$

Therefore, in general, following the commutative diagram of Fig. 17.1, we have

$$T_{ij} \doteq h_i \circ g_i \circ g_j^{-1}(\mu) \circ h_j^{-1} \quad (17.3)$$

so that $g = g_i g_j^{-1}$ and h is a transformation that depends on h_i, h_j and μ . Given two or more “shapes” and a cost functional $E : \mathcal{H} \rightarrow \mathbb{R}^+$ defined on the set of diffeomorphisms, the motion g_t and the shape average are defined as the minimizers of $\sum_{t=1}^n E(h_t)$ subject to $\gamma_t = h_t \circ g_t(\mu)$. Note that all that matter in the cost of h_t are the “shapes” before and after the transformation, $\mu_i \doteq g_i(\mu)$ and γ_i , so that we can write, with an abuse of notation, $E(h(\mu_i, \gamma_i)) \doteq E(\mu_i, \gamma_i)$. We are therefore ready to define our notion of motion during a deformation.

DEFINITION: 5 Let $\gamma_1, \dots, \gamma_n$ be smooth boundaries of closed subsets of a differentiable manifold embedded in \mathbb{R}^N , which we call pre-shapes. Let \mathcal{H} be a class of diffeomorphisms acting on γ_i , and let $E : \mathcal{H} \rightarrow \mathbb{R}^+$ be a positive, real-valued functional. Consider now a group G acting on γ_i via $g(\gamma_i)$. We say that $\hat{\gamma}_1, \dots, \hat{\gamma}_n$ is a **motion** undergone by γ_i , $i = 1 \dots n$ if there exists a pre-shape $\hat{\mu}$ such that

$$\hat{\gamma}_1, \dots, \hat{\gamma}_n, \hat{\mu} = \arg \min_{g_t, \mu} \sum_{i=1}^n E(h_i) \text{ subject to } \gamma_i = h_i \circ g_i(\mu) \quad (17.4)$$

The pre-shape $\hat{\mu}$ is called the **shape average** relative to the group G , or G -average, and the quantity $\hat{g}_i^{-1}(\gamma_i)$ is called the **shape** of γ_i .

Remark 1 (Invariance) In the definition above, one will notice that the shape average is actually a pre-shape, and that there is an arbitrary choice of group action g_0 that, if applied to γ_i and μ , leaves the definition unchanged (the functional E is invariant with respect to g_0 because $T(g \circ g_0, h \circ g_0) = T(g, h) \forall g_0$). For the case of the Euclidean group $SE(N)$, a way to see this is to notice that the reference frame where μ is described is arbitrary. Therefore, one may choose, for instance, $\mu = h_1^{-1}(\gamma_1)$.

Remark 2 (G-average) Notice that the notion of shape average above is relative to the particular choice of group G . For instance, given a number of pre-shapes $\gamma_1, \dots, \gamma_n$, the shape average $\hat{\mu}$ relative to the Euclidean group will, in general, be different than the shape average relative to the affine or the projective group (e.g. Fig. 17.2, 17.2). Therefore, when we talk about average, we must specify the group G , e.g. Euclidean average, affine average etc.

Remark 3 (Symmetries) The minimizer of (17.4) will not be unique when the pre-shape γ is invariant (i.e. symmetric) with respect to an element or subgroup of G . The simplest case is a circle, which is invariant to rotations around its center. It is clear that by matching circles we can determine the relative position of their centers, but not the relative orientation of the reference frame attached to each circle. Notice, however, that the notion of shape average is still well-defined even when the notion of motion is not unique. This is because any element in the symmetry group suffices to register the pre-shapes and thereby compute the shape average (Fig. 17.2).

In Sect. 17.3 we specialize this definition for the case of a planar contour undergoing Euclidean or affine motion and differentiable deformations, and we show how to compute motion, shape average, as well as distances between shapes and principal modes of variation.

17.3 Shape and deformation of a planar contour

In this section we consider the implementation of the program above for a simple case: two closed planar contours, γ_1 and γ_2 , where we choose as cost functional for the deformations h_1, h_2 either the set-symmetric difference Δ of their interior (the union minus the intersection), or what we call the signed distance transform score¹ ψ

$$\psi(\mu, \gamma) \doteq \int_{\bar{\mu}} \zeta(\gamma) d\mathbf{x} + \int_{\bar{\gamma}} \zeta(\gamma) d\mathbf{x} \quad (17.5)$$

where $\bar{\mu}$ denotes the interior of the contour μ and ζ is the signed distance function of the contour γ ; $d\mathbf{x}$ is the area form on the plane. In either case, since we have an arbitrary choice of the global reference frame, we can choose $g_1 = e$, the group identity. We also call $g \doteq g_2$, so that $\mu_2 = g(\mu)$. The problem of defining the motion and shape average can be written as

$$\hat{g}, \hat{\mu} = \arg \min_{g, \mu} \sum_{i=1}^2 E(h_i) \text{ subject to } \gamma_1 = h_1(\mu); \gamma_2 = h_2 \circ g(\mu). \quad (17.6)$$

As we have anticipated, we choose either $E(h_i) = \Delta(g_i(\mu), \gamma_i)$ or $E(h_i) \doteq \psi(g_i(\mu), \gamma_i)$. Therefore, abusing the notation as anticipated before Def. 5, we can write the problem above as an unconstrained minimization

$$\boxed{\hat{g}, \hat{\mu} = \arg \min_{g, \mu} \phi(\gamma_1, \gamma_2)} \quad \text{where} \quad \boxed{\phi(\gamma_1, \gamma_2) \doteq E(\mu, \gamma_1) + E(g(\mu), \gamma_2)} \quad (17.7)$$

and E is either Δ or ψ . The estimate \hat{g} defines the motion between γ_1 and γ_2 , and the estimate $\hat{\mu}$ defines the average of the two contours.

If one thinks of contours and their interior, represented by a characteristic function χ , as a binary image, then the cost functional above is just a particular case of a more general cost functional where each term is obtained by integrating a function inside and a function outside the contours

$$\boxed{\phi = \sum_{i=1}^2 \int_{\bar{\mu}_{in}} f_{in}(\mathbf{x}, \gamma_i) d\mathbf{x} + \int_{\bar{\mu}_{out}} f_{out}(\mathbf{x}, \gamma_i) d\mathbf{x}} \quad (17.8)$$

¹The rationale behind this score is that one wants to make the signed distance function as positive as possible outside the contour to be matched, and as negative as possible inside. This score can be interpreted as a weighted Monge-Kantorovic functional where the mass of a curve is weighted by its distance from the boundary.

where the bar in $\bar{\mu}$ indicates that the integral is computed on a *region* inside or outside μ and we have emphasized the fact that the function f depends upon the contour γ_i . For instance, for the case of binary images, we have $f_{in} = (\chi_\gamma - 1)^2$ and $f_{out} = \chi_\gamma^2$. To solve the problem, therefore, we need to minimize the following functional

$$\int_{\bar{\mu}_{in}} f_{in}(\mathbf{x}, \gamma_1) d\mathbf{x} + \int_{\bar{\mu}_{out}} f_{out}(\mathbf{x}, \gamma_1) d\mathbf{x} + \int_{g(\bar{\mu}_{in})} f_{in}(\mathbf{x}, \gamma_2) d\mathbf{x} + \int_{g(\bar{\mu}_{out})} f_{out}(\mathbf{x}, \gamma_2) d\mathbf{x} \quad (17.9)$$

which can be written, after a change of variable in the last two terms, as

$$\int_{\bar{\mu}_{in}} f_{in}(\mathbf{x}, \gamma_1) + f_{in}(g(\mathbf{x}), \gamma_2) |J_g| d\mathbf{x} + \int_{\bar{\mu}_{out}} f_{out}(\mathbf{x}, \gamma_1) + f_{out}(g(\mathbf{x}), \gamma_2) |J_g| d\mathbf{x} \quad (17.10)$$

where $|J_g|$ is the determinant of the Jacobian of the group action g . This makes it easy to compute the component of the first variation of ϕ along the normal direction to the contour μ , so that we can impose

$$\nabla_\mu \phi \cdot N = 0 \quad (17.11)$$

to derive the first-order necessary condition. If we choose $G = SE(2)$, an isometry, it can be easily shown that

$$\boxed{\nabla_\mu \phi = f_{in}(\mathbf{x}, \gamma_1) - f_{out}(\mathbf{x}, \gamma_1) + f_{in}(g(\mathbf{x}), \gamma_2) - f_{out}(g(\mathbf{x}), \gamma_2)} \quad (17.12)$$

17.3.1 Representation of motions

For the case of matrix Lie groups (e.g. $SE(2)$), there exist twist coordinates ξ that can be represented as a skew-symmetric matrix $\hat{\xi}$ so that²

$$\boxed{g = e^{\hat{\xi}}} \quad \text{and} \quad \boxed{\frac{\partial g}{\partial \xi_i} = \frac{\partial \hat{\xi}}{\partial \xi_i} g} \quad (17.13)$$

where the matrix $\frac{\partial \hat{\xi}}{\partial \xi_i}$ is composed of zeros and ones and the matrix exponential can be computed in closed form.

17.3.2 Variation with respect to the group action

To compute the variation of the functional ϕ with respect to the group action g , we first notice that the first two terms in ϕ do not contribute since they are

²The “widehat” notation $\hat{\cdot}$, which indicates a lifting to the Lie algebra, should not be confused with the “hat” $\hat{\cdot}$, which indicates an estimated quantity.

independent of g . We therefore consider the variation of

$$\int_{g(\bar{\mu}_{in})} f_{in}(\mathbf{x}, \gamma_2) d\mathbf{x} + \int_{g(\bar{\mu}_{out})} f_{out}(\mathbf{x}, \gamma_2) d\mathbf{x}. \quad (17.14)$$

To simplify the derivation, we consider the case of $SE(3)$. Other cases follow along similar lines (except for the Jacobian of the transformations, which is absent in the isometric case); we also note that both terms above are of the generic form $A(g) \doteq \int_{g(\bar{\mu})} f(\mathbf{x}) d\mathbf{x}$. Therefore, we consider the variation of A with respect to the components of the twist ξ_i , $\frac{\partial A}{\partial \xi_i}$, which we will eventually use to

compute the gradient with respect to the natural connection $\nabla_G \phi = \widehat{\left(\frac{\partial \phi}{\partial \xi}\right)} g$. We first rewrite $A(g)$ using the change of measure $\int_{g(\bar{\mu})} f(\mathbf{x}) d\mathbf{x} = \int_{\bar{\mu}} f \circ g(\mathbf{x}) |J_g| d\mathbf{x}$ which leads to $\frac{\partial A(g)}{\partial \xi_i} = \int_{\bar{\mu}} \frac{\partial}{\partial \xi_i}(f \circ g(\mathbf{x})) |J_g| d\mathbf{x} + \int_{\bar{\mu}} (f \circ g(\mathbf{x})) \frac{\partial}{\partial \xi_i} |J_g| d\mathbf{x}$ and note that the Euclidean group is an isometry and therefore the determinant of the Jacobian is one and the second integral is zero. The last equation can be re-written, using Green's theorem, as $\int_{g(\mu)} \left\langle f(\mathbf{x}) \frac{\partial g}{\partial \xi_i} \circ g^{-1}(\mathbf{x}), N \right\rangle ds = \int_{\mu} \left\langle f \circ g(\mathbf{x}) \frac{\partial g}{\partial \xi_i}, g_* N \right\rangle ds$ where g_* indicates the push-forward. Notice that g is an isometry and therefore it does not affect the arc length; we then have

$$\frac{\partial A(g)}{\partial \xi_i} = \int_{\mu} f(g(\mathbf{x})) \left\langle \frac{\partial \widehat{\xi}}{\partial \xi_i} g, g_* N \right\rangle ds \quad (17.15)$$

After collecting all the partial derivatives into an operator $\frac{\partial \phi}{\partial \xi}$, we can write the evolution of the group action.

17.3.3 Evolution

The algorithm for evolving the contour and the group action consists of a two-step process where an initial estimate of the contour $\hat{\mu} = \gamma_1$ is provided, along with an initial estimate of the motion $\hat{g} = e$. The contour and motion are then updated in an alternating minimization fashion where motion is updated according to

$$\boxed{\frac{d\hat{g}}{dt} = \widehat{\left(\frac{\partial \phi}{\partial \xi}\right)} \hat{g}}$$

(17.16)

Notice that this is valid not just for $SE(2)$, but for any (finite-dimensional) matrix Lie group, although there may not be a closed-form solution for the exponential map like in the case of $SE(3)$ and its subgroups. In practice, the group evolution (17.16) can be implemented in local (exponential) coordinates by evolving ξ defined by $g = e^{\hat{\xi}}$ via $\frac{d\xi}{dt} = \frac{\partial \phi}{\partial \xi}$. In the level set framework, the derivative of the cost function ϕ with respect to the coordinates of the group action ξ_i can be computed as the collection of two terms, one for f_{in} , one for f_{out} where $\frac{\partial \phi}{\partial \xi_i} = \int_{g(\gamma_{1,2})} \left\langle \frac{\partial g(\mathbf{x})}{\partial \xi_i}, f_{\{in,out\}}(g(\mathbf{x}), \gamma_{1,2}) J(g_* T) \right\rangle ds$. The contour $\hat{\mu}$ evolves

according to

$$\boxed{\frac{d\hat{\mu}}{dt} = (f_{in}(\mathbf{x}, \gamma_1) - f_{out}(\mathbf{x}, \gamma_1) + f_{in}(g(\mathbf{x}), \gamma_2) - f_{out}(g(\mathbf{x}), \gamma_2))N.} \quad (17.17)$$

As we have already pointed out, the derivation can be readily extended to surfaces in space.

17.3.4 Distance between shapes

The definition of motion \hat{g} and shape average $\hat{\mu}$ as a minimizer of (17.6) suggests defining the distance³ between two shapes as the “energy” necessary to deform one into the other via the average shape:

$$d(\gamma_i, \gamma_j) \doteq E(\gamma_i, T(\hat{g}, \hat{h})\gamma_j). \quad (17.18)$$

For instance, for the set-symmetric difference of two contours, we have

$$d_\Delta(\gamma_1, \gamma_2) \doteq \int \chi_{\hat{\mu}} \chi_{\gamma_1} + \chi_{\hat{g}(\hat{\mu})} \chi_{\gamma_2} d\mathbf{x} \quad (17.19)$$

and for the signed distance transform we have

$$d_\psi(\gamma_1, \gamma_2) \doteq \int_{\hat{\mu}} \zeta(\gamma_1) d\mathbf{x} + \int_{\hat{g}(\hat{\mu})} \zeta(\gamma_2) d\mathbf{x}. \quad (17.20)$$

In either case, given two contours, a gradient flow algorithm based on Eq. (17.16) and (17.17), when it converges to the global minimum, returns as the minimum value the distance between the shapes of the two contours.

17.4 Moving average and tracking

The discussion above assumes that an unsorted collection of shapes is available, where the deformation between any two shapes is “small,” so that the whole collection can be described by a single average shape. Consider however the situation where an object is evolving in time, for instance Fig. 17.4. While the deformation between adjacent time instants could be captured by a group action and a small deformation, as time continues the object may change so drastically that a global time average may not make sense.

³Here we use the term distance informally, since we do not require that it satisfies the triangular inequality. The term pseudo-distance would be more appropriate.

One way to approach this issue is by defining a notion of “*moving average*,” similarly to what is done in time series analysis⁴. We adapt this model by changing the representation of the uncertainty during the evolution. In classical linear time series, uncertainty is modeled via additive noise. In our case, the uncertainty is an infinite-dimensional deformation h of the measured contour. The model becomes

$$\begin{cases} \mu(t+1) = g(t)\mu(t) \\ \gamma(t) = h(\mu(t)) \end{cases} \quad (17.22)$$

where $\mu(t)$ represents the moving average of order $k = 1$. The procedure described in Sect. 17.3, initialized with $\mu(0) = \gamma_1$, provides an estimate of the moving average of order 1, as well as the *tracking* of the trajectory $g(t)$ in the group G , which in (17.22) is represented as the model parameter. Note that the procedure in Sect. 17.3 simultaneously estimates the state $\mu(t)$ and identifies the parameters $g(t)$ of the model (17.22). It does so, however, without imposing restrictions on the evolution of $g(t)$. If one wants to impose additional constraints on the motion parameters, one can augment the state of the model to include the parameters g .

$$\begin{cases} g(t+1) = e^{\hat{\xi}(t)}g(t) \\ \mu(t+1) = g(t)\mu(t) \\ \gamma(t) = h(\mu(t)) \end{cases} \quad (17.23)$$

and specify restrictions on ξ . In Fig. 17.4 we show the results of tracking a storm with a moving average of order one.

17.5 Averaging and registering non-equivalent shapes

So far we have assumed that the given shapes are obtained by moving and deforming a common underlying “template” (the average shape). Even though the given shapes are not *equivalent* (i.e. there is no group action g that maps one exactly onto the other), g is found as the group element that minimizes the cost of the deviation from such an equivalence. In the algorithm proposed in Eq. (17.16)-(17.17), however, there is no explicit requirement that the deformation between the given shapes be small. Therefore, the procedure outlined can be seen as an algorith to

⁴For instance, consider a point $\mathbf{x}(t)$ moving on the plane according to a simple linear dynamics, observed through a “noisy” measurement $\mathbf{y}(t)$:

$$\begin{cases} \mathbf{x}(t) = A_1\mathbf{x}(t-1) + A_2\mathbf{x}(t-2) + \cdots + A_k\mathbf{x}(t-k) + \mathbf{v}(t) \\ \mathbf{y}(t) = C\mathbf{x}(t) + \mathbf{w}(t). \end{cases} \quad (17.21)$$

This model describes a dynamical system where the state \mathbf{x} can be interpreted as the (autoregressive) moving average of \mathbf{y} . Without loss of generality one may assume that $k = 1$ since the difference equation above can always be reduced to first-order by augmenting the dimension of the state (see [319] for details).

register shapes that are not equivalent under the group action. A *registration* is a group element \hat{g} that minimizes the cost functional (17.4).

To illustrate this fact, consider the two considerably different shapes shown in Fig. 17.6, γ_1, γ_2 . The simultaneous estimation of their average μ , for instance relative to the affine group, and of the affine motions that best matches the shape average onto the original ones, g_1, g_2 , provides a registration that maps γ_1 onto γ_2 and viceversa: $g = g_2 g_1^{-1}$.

If instead of considering the images in Fig. 17.6 as binary images that represent the contours, we consider them as grayscale images, then the procedure outlined, for the case where the score is computed using the set-symmetric difference, provides a way to simultaneously jointly segment the two images and register them. This idea is illustrated in Fig. 17.7 for true grayscale (magnetic resonance) images of brain sections. Therefore, this procedure can be used to simultaneously segment a collection of uncalibrated images, thus extending the approach of [591].

17.6 Matching with missing parts

The formulation of the problem and the derivation of the evolution equations are introduced in this section for the case of a planar shape under isometric transformations (rigid motions). In this case, the dimension of the space is 2 and the determinant of the Jacobian of the group is $J(g) = 1$ for all the elements g of the group G . The main assumption about the shape is that it must be a *regular domain*, that is an open and bounded subset of \mathbb{R}^2 with a finite number of connected components and a piece-wise C^∞ boundary. This regularity is required to avoid singular pathologies and to make the computation possible. In this section we indicate with $\bar{\gamma}_i, \bar{\mu}$ regular domains in \mathbb{R}^2 , and with γ_i, μ their boundaries.

Let $\bar{\gamma}_1, \dots, \bar{\gamma}_k$ be regular domains of \mathbb{R}^2 , all obtained from the same regular domain $\bar{\mu} \subset \mathbb{R}^2$ by composition with characteristic functions

$$\chi_1, \dots, \chi_k : \mathbb{R}^2 \rightarrow \mathbb{R}$$

and actions of Lie group elements $g_1, \dots, g_k \in G$. We want to find the best solution in the sense expressed by the functional

$$\phi = \sum_{i=1}^k A(\bar{\gamma}_i \setminus g_i(\bar{\mu})) + \alpha A(\bar{\mu}) \quad (17.24)$$

where A denotes the area, α is a design constant, $\bar{\mu}, \chi_i$ and g_i are the unknowns and the sets $\bar{\gamma}_i$ and the structure of G are given. The rationale behind the choice of the cost function ϕ is that one wants to maximize the overlap between the incomplete shapes and the registered complete shape (first term) while keeping the complete shape as small as possible (second term). This is equivalent to minimizing the area of the $\bar{\gamma}_i$ that is not covered by the image of the complete shape after the application of the group action g_i . At the same time, one needs to minimize a

quantity related to the complete shape (e.g. the area) to constrain the solution to be non-singular. Without the second term, it is always possible to choose a compact complete shape that covers all the incomplete ones (e.g. a big square) and minimizes the first term.

17.6.1 Minimization with respect to shape

The functional ϕ can be written in integral form

$$\phi = \sum_{i=1}^k \int_{\gamma_i} (1 - g_i \bar{\mu}) d\mathbf{x} + \alpha \int_{\bar{\mu}} d\mathbf{x} \quad (17.25)$$

and using the characteristic function notation

$$\phi = \sum_{i=1}^k \int \chi(\gamma_i) (1 - \chi(g_i \mu)) d\mathbf{x} + \alpha \int \chi(\mu) d\mathbf{x} \quad (17.26)$$

$$= \sum_{i=1}^k \int \chi(\gamma_i) d\mathbf{x} - \sum_{i=1}^k \int \chi(\gamma_i) \chi(g_i \mu) d\mathbf{x} + \alpha \int \chi(\mu) d\mathbf{x}. \quad (17.27)$$

Since the first term of ϕ is independent of μ, g and remembering that g_i are isometries, the problem of minimizing ϕ is equivalent to that of finding the minimum of the energy

$$E(g_i, \mu) = \int_{\bar{\mu}} \left(\alpha - \sum_{i=1}^k \chi(g_i^{-1} \gamma_i) \right) d\mathbf{x}. \quad (17.28)$$

One can show that the first variation of this integral along the normal direction of the contour is simply its integrand, by using the divergence theorem, and therefore conclude that a gradient flow that minimizes the energy E with respect to the shape of the contour μ is given by

$$\frac{\partial \mu}{\partial t} = \left(\alpha - \sum_{i=1}^k \chi(g_i^{-1} \gamma_i) \right) N \quad (17.29)$$

where N is the normal vector field of the contour μ .

17.6.2 Minimization with respect to the group action

In computing the variation of the functional ϕ with respect to the group actions g_i , we note that there is only one term that depends on g_i in equation (17.28). Therefore, we simply compute the variation of

$$- \int_{\bar{\mu}} \chi(g_i^{-1} \gamma_i) d\mathbf{x}. \quad (17.30)$$

To simplify the notation, we note that the term above is of the generic form

$$W(g) \doteq \int_{\bar{\mu}} f(g(\mathbf{x})) d\mathbf{x} \quad (17.31)$$

with $f = \chi(\gamma_i)$. Therefore, we consider the variation of W with respect to the components of the exponential coordinates⁵ ξ_i of the group $g_i = e^{\hat{\xi}_i}$

$$\frac{\partial W}{\partial \xi_i} = \frac{\partial}{\partial \xi_i} \int_{\bar{\mu}} f(g(\mathbf{x})) d\mathbf{x} \quad (17.32)$$

$$= \int_{\bar{\mu}} \nabla f(g(\mathbf{x})) \frac{\partial}{\partial \xi_i} g(\mathbf{x}) d\mathbf{x}. \quad (17.33)$$

Using Green's theorem it is possible to write the variation as an integral along the contour of μ and one over $g(\bar{\mu})$ with a divergence integrand

$$\frac{\partial W}{\partial \xi_i}(g) = \int_{\mu} f(g(x)) \left\langle \frac{\partial}{\partial \xi_i} g(x), g^* N \right\rangle ds - \int_{g(\bar{\mu})} f(y) \nabla_y \cdot \left(\frac{\partial}{\partial \xi_i} g(g^{-1}(y)) \right) dy. \quad (17.34)$$

Therefore, the derivative with respect of the group action is

$$\boxed{\frac{\partial \phi}{\partial \xi_i} = \int_{\bar{\mu} \cap g_i^{-1}(\gamma_i)} \left\langle \frac{\partial}{\partial \xi_i} g_i(x), g_i^* N_i \right\rangle ds.} \quad (17.35)$$

Where N_i is the normal vector field to the boundary of $g_i^{-1}(\gamma_i)$ and g_i^* is the push forward induced by the map g_i .

17.6.3 Evolution equations

Within the level set framework, a function ψ is evolved instead of the contour μ . The function ψ is negative inside μ , positive outside and zero on the contour. The evolution of ψ depends on the velocity of μ via the Hamilton-Jacobi equation

$$\begin{cases} u_t + \left(\alpha - \sum_{i=1}^k \chi(g_i^{-1} \gamma_i) \right) |\nabla u| = 0, \\ u(0, x) = \psi^{(t)}(x). \end{cases} \quad (17.36)$$

The evolution equations follow

$$\psi^{(t+1)}(x) = u(1, x | \psi^{(t)}) \quad (17.37)$$

$$\bar{\mu}^{(t)} = \{x : \psi(x) < 0\} \quad (17.38)$$

$$\xi_i^{(t+1)} = \xi_i^{(t)} - \beta_\xi \int_{\bar{\mu} \cap g_i^{-1}(\gamma_i)} \left\langle \frac{\partial}{\partial \xi_i} g_i(x), g_i^* N_i \right\rangle ds \quad (17.39)$$

⁵Every finite-dimensional Lie group admits exponential coordinates. For the simple case of the isometries of the plane, the exponential coordinates can be computed in closed-form using Rodrigues' formula.

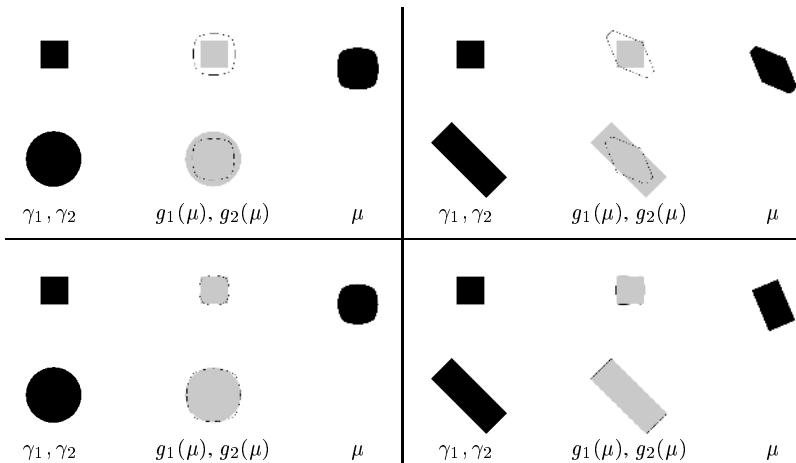


Figure 17.2. **Euclidean (top) vs. affine (bottom) registration and average.** For each pair of objects γ_1, γ_2 , the registration $g_1(\mu), g_2(\mu)$ of the average shape μ is shown. Note that the affine average can simultaneously “explain” a square and a rectangle, whereas the Euclidean average cannot.

where β_ξ is a step parameter and $u(\cdot, \cdot | \psi^{(t)})$ is the solution of (17.36) with initial condition $u(0, x) = \psi^{(t)}(x)$.

17.6.4 Generalization to graylevel images

There are many possible generalizations of the above formulas. Here we present the case of gray level images. The case of color images is very similar so the equations will not be stated. The main idea is to work with a generic level set between the minimum and the maximum intensity levels and write the functional to match all the level sets *simultaneously* using the partial differential equation in Eq. (17.47). To derive this equation, let k images of the same object I_i , $i = 1, \dots, k$ be given and

$$\bar{\gamma}_i^\delta = \{x : I_i < \delta\} \quad (17.40)$$

be the generic δ -underlevel of I_i , where δ is a positive constant. Then let

$$\mu : \mathbb{R}^2 \rightarrow \mathbb{R} \quad (17.41)$$

be a function that represents the complete image intensity (the choice of the name μ is not casual, since this will turn out to be the complete shape for the case of greylevel images) and

$$\begin{aligned} \bar{\mu}^\delta &= \{x : \mu < \delta\} \\ \mu^\delta &= \partial \bar{\mu}^\delta \end{aligned} \quad (17.42)$$

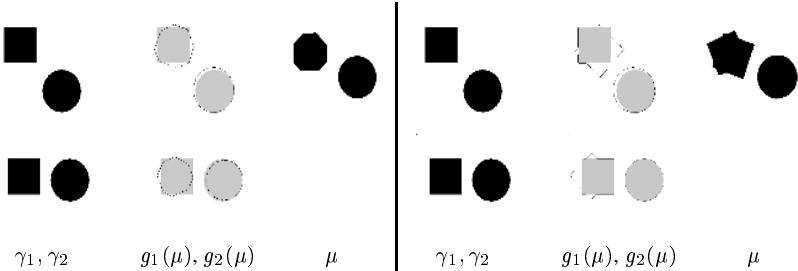


Figure 17.3. **Signed distance (left) vs. set-symmetric difference (right).** Original contours γ_1, γ_2 , and registrations $g_1(\mu), g_2(\mu)$ of average shape μ (original contours are disconnected unions of a circle and a square). The choice of pseudo-distance between contours influences the resulting average. The signed distance transform captures more of the features that are common to the two shapes, whereas the symmetric difference captures the features of both.

respectively its underlevel and the boundary of the underlevel. From the same considerations developed for the case of a contour, we write the functional ϕ as

$$\phi = \sum_{i=1}^k \sum_{j=1}^n A(\bar{\gamma}_i^\delta \setminus g_i(\bar{\mu}^j)) + \alpha \sum_{j=1}^n A(\mu^j) \quad (17.43)$$

where j is a discretization of the level set δ

$$\phi = \sum_{i=1}^k \sum_{j=1}^n \int \chi(\gamma_i^\delta)(1 - \chi(g_i(\mu^j))) d\mathbf{x} + \alpha \sum_{j=1}^n \int \chi(\mu^j) d\mathbf{x} \quad (17.44)$$

and in the same way the derivatives of the functional can be obtained as

$$\frac{\partial \mu^j}{\partial t} = \left(\alpha - \sum_{i=1}^k \chi(g_i^{-1} \gamma_i^\delta) \right) N^j, \quad (17.45)$$

$$\frac{\partial \phi}{\partial \xi_i} = - \sum_{j=1}^n \int_{\bar{\mu}^j \cap g_i^{-1}(\gamma_i^\delta)} \left\langle \frac{\partial}{\partial \xi_i} g_i(x), g_i^* N_i^\delta \right\rangle ds. \quad (17.46)$$

After letting j go to the limit, Eq. (17.45) gives the Hamilton-Jacobi equation for the function μ :

$$\mu_t(t, x) + \left(\sum_{i=1}^k H(I_i(g_i(x)) - \mu(t, x)) - \alpha \right) |\nabla \mu(t, x)| = 0 \quad (17.47)$$

where $H(s)$ denotes the standard Heaviside function (1 if $s > 0$, else 0). Thus, the evolution of the function μ and the parameters g_i is given by

$$\begin{cases} \mu^{(t+1)}(x) = \mu^{(t)}(x) - \beta_\mu \left(\sum_{i=1}^k H(I_i(g_i^{(t)}(x)) - \mu^{(t)}(x)) - \alpha \right) |\nabla \mu^{(t)}(x)| \\ \xi_i^{(t+1)} = \xi_i^{(t)} - \beta_\xi \int H(I_i(g_i^{(t)}(x)) - \mu^{(t)}(x)) \left\langle \frac{\partial g_i^{(t)}}{\partial \xi_i}, (\nabla \gamma_i)(g_i^{(t)}(x)) \right\rangle dx \end{cases}$$

where β_μ and β_ξ are step parameters.

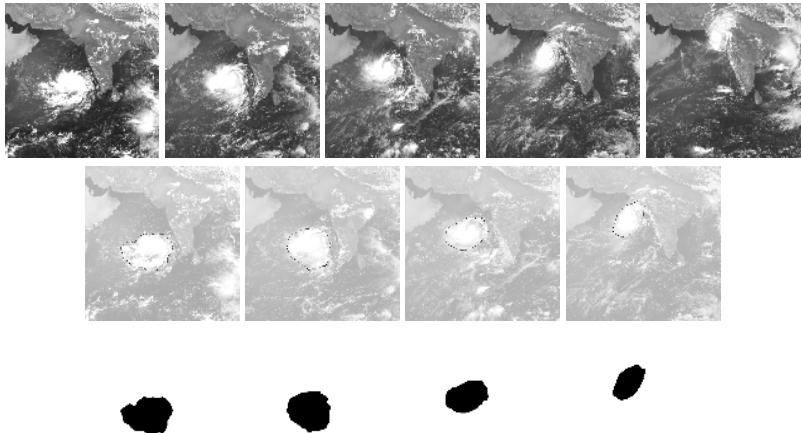


Figure 17.4. Storm Motion. A moving affine average storm shape (bottom row) is captured from neighboring pairs of EUMETSAT ©2001 images (top row). The registrations (to the first image of each pair) are shown along the middle row.

17.7 Experiments

Fig. 17.2 illustrates the difference between the motion and shape average computed under the Euclidean group, and the affine one. The three examples show the two given shapes γ_i , the mean shape registered to the original shapes, $g_i(\mu)$ and the mean shape μ . Notice that affine registration allows to simultaneously capture the square and the rectangle, whereas the Euclidean average cannot be registered to either one, and is therefore only an approximation.

Fig. 17.3 compares the effect of choosing the signed distance transform score (left) and the set-symmetric difference (right) in the computation of the motion and average shape. The first choice results in an average that captures the common features of the original shapes, whereas the second captures more of the features in each one. Depending on the application, one may prefer one or the other.

Fig. 17.4 shows the results of tracking a storm. The affine moving average is computed, and the resulting affine motion is displayed. The same is done for the jellyfish in Fig. 17.5.

Fig. 17.6 is meant to challenge the assumptions underlying our method. The chosen shapes are not simply local deformations of one another. Therefore, the notion of shape average is not meaningful *per se* in this context, but serves to compute the change of (affine) pose between the two shapes (Fig. 17.6). Nevertheless, it is interesting to observe how the shape average allows registering even apparently disparate shapes. Notice that our framework is not meant to capture such a wide range of variation. In particular, it does not possess a notion of “parts” and it is neither hierarchical nor compositional. In the context of non-equivalent shapes (shapes for which there is no group action mapping one exactly onto the

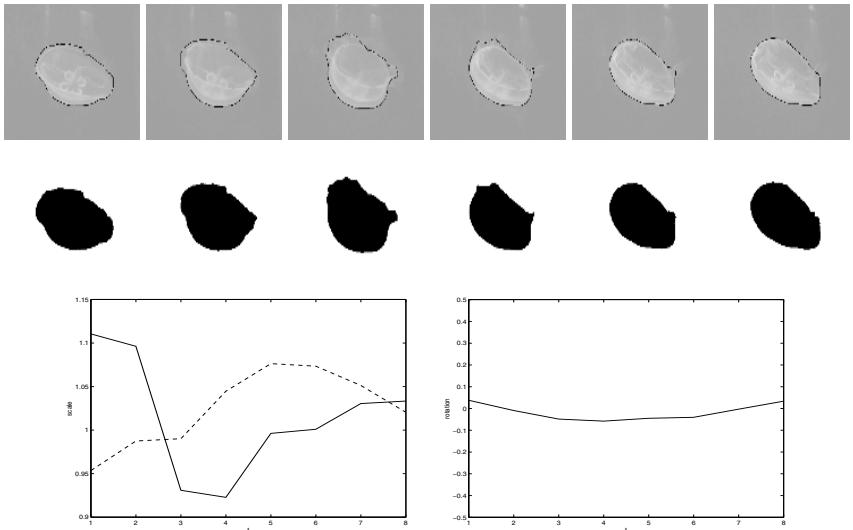


Figure 17.5. Jellyfish Motion. Affine registration (top), moving average (middle), and affine motion (bottom) for a moving and deforming jellyfish. Last row: affine scales along x and y , and rotation about z during the sequence.

other), the *average shape serves purely as a support to define and compute motion in a collection of images of a given deforming shape*.

Fig. 17.7 shows the results of simultaneously segmenting and computing the average motion and registration for 4 images from a database of magnetic resonance images of the corpus callosum.

A numerical implementation of the evolution equations (17.37),(17.38),(17.39) has been written within the level set framework proposed by Osher and Sethian [401] using an ultra narrow band algorithm and an alternating minimization scheme. A set of hand shapes has been chosen and converted into binary images of 256×256 pixels. For each image of this set, a group of binary images with missing parts and with different poses has been generated (Fig. 17.8, curves $\gamma_1, \dots, \gamma_5$).

The following level set evolution equation has been used

$$\mu_t + \left(\alpha - \sum_{i=1}^k \chi(g_i^{-1} \gamma_i) \right) |\nabla \mu| = 0 \quad (17.48)$$

with a first-order central scheme approximation. The evolution of the pose parameters has been carried out using the integral (17.39) with the following approximation of the arclength ds

$$ds \approx |\nabla(\mu)| d\mathbf{x}. \quad (17.49)$$

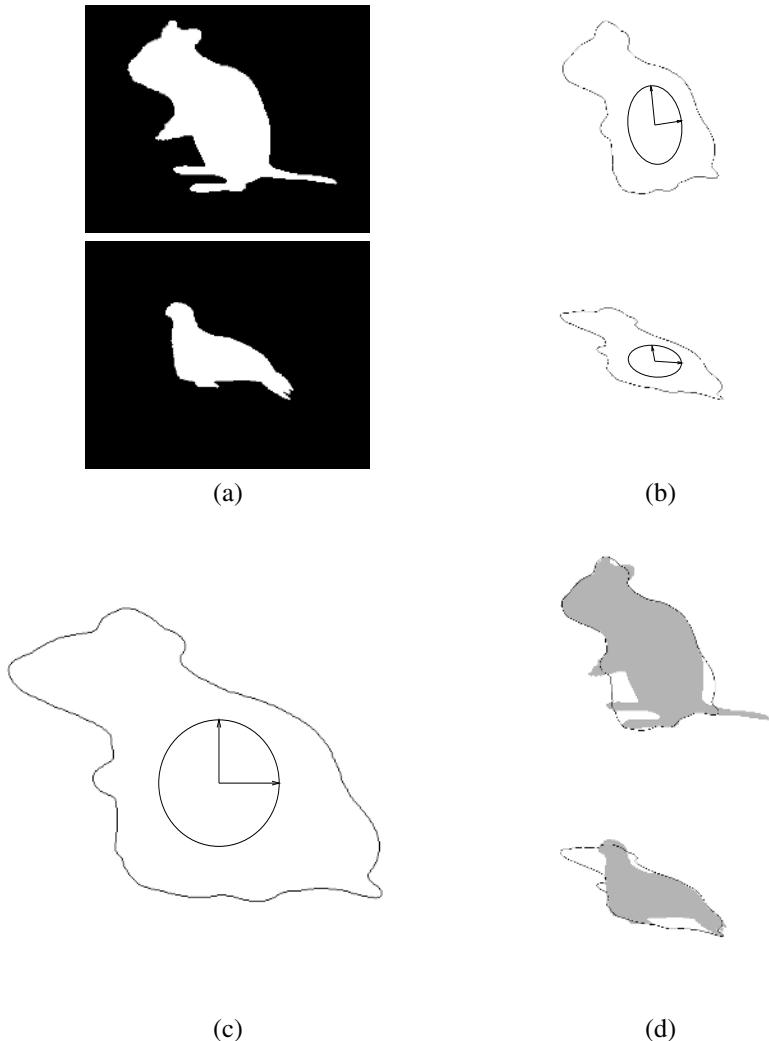


Figure 17.6. Registering non-equivalent shapes. (a) Two binary images representing two different shapes; (b) affine registration; (c) average affine shape; and (d) approximations of the original shapes via the registered average shape.

The evolution has been initialized with the following settings

$$\begin{aligned} \mu_{t=0} &= \gamma_1 \\ T_i &= B_{\gamma_i} - B_{\gamma_1} \\ R_i &= \begin{pmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{pmatrix}, \quad \text{with } \theta_i = \widehat{E_{\gamma_i} O E_{\gamma_1}} \end{aligned} \tag{17.50}$$

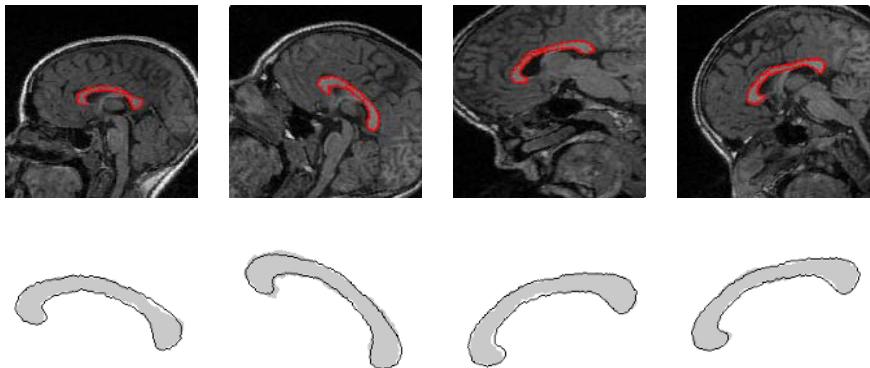


Figure 17.7. **Corpus Callosum** (top row) a collection of (MR) images from different patients (courtesy of N. Dutta and A. Jain [171]) and (bottom row) alignments of average shape corresponding to the affine group.

where B_{γ_i} is the centroid of γ_i and E_{γ_i} is the principal axis of inertia of the region $\bar{\gamma}_i$. The value of α has been set between 0 and 1. R_i, T_i are the rotational and translational components of $g_i = (R_i, T_i) \in SE(2)$.

Results are illustrated in Fig. 17.8. γ_j are the starting curves and μ is the complete shape in an absolute system after the computation. The figure shows the computed $g_j(\mu)$, the estimated rigid motions.

Figure 17.10 shows the method applied to grayscale images of a face, where different portions have been purposefully erased.

The experiments show that this method works very well even when the missing part in each image is pretty significant, up to about 20% of the area.

Acknowledgements

This research is supported in part by NSF grant IIS-9876145, ARO grant DAAD19-99-1-0139 and Intel grant 8029.

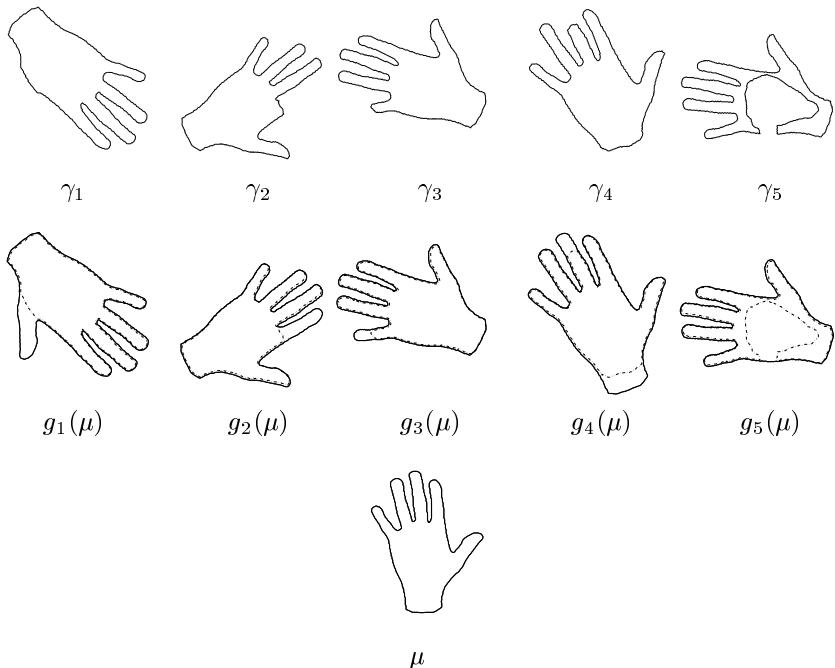


Figure 17.8. **Hands.** (Top) a collection of images of the same hand in different poses with different missing parts. The support of the missing parts is unknown. (Middle) similarity group, visualized as a “registered” image. (Bottom) estimated template corresponding to the similarity group (“complete shape”).

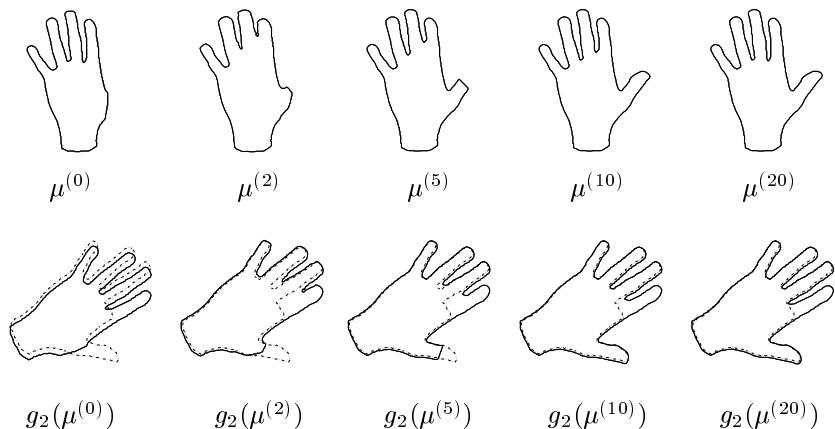


Figure 17.9. **Hands Evolution.** (Top) evolution of the complete shape for $t = 0, \dots, 20$. (Bottom) evolution of $g_2(\mu)$ for $t = 0, \dots, 20$.

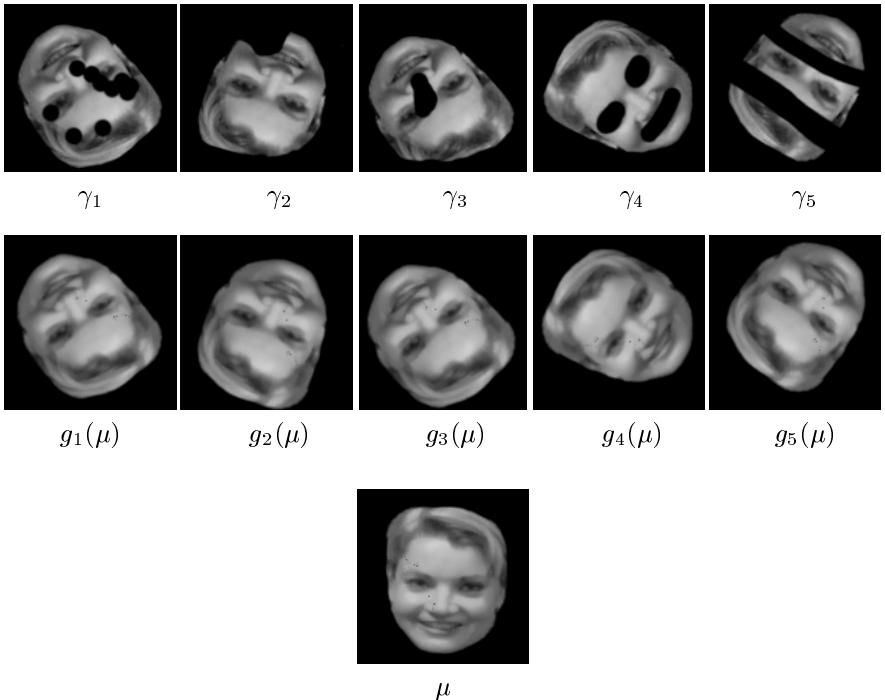


Figure 17.10. **Faces** (Top) a collection of images of the same face in different poses with different missing parts. The support of the missing parts is unknown. (Middle) similarity group, visualized as a “registered” image. (Bottom) estimated template corresponding to the similarity group (“complete image”).

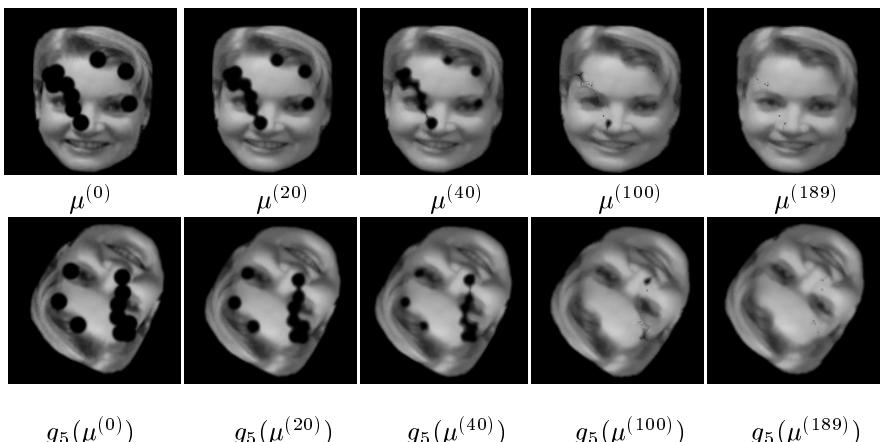


Figure 17.11. **Face evolution.** (Top) evolution of the complete image for $t = 0, \dots, 189$. (Bottom) evolution of $g_5(\mu)$ for $t = 0, \dots, 189$.

Part VII

Computational Stereo & Implicit Surfaces

18

Computational Stereo: A Variational Method

Olivier Faugeras, Jose Gomes and Renaud Keriven

Abstract

Given a set of simultaneously acquired images of a scene, the stereo problem consists of reconstructing the three-dimensional shape of the objects in the scene. Traditionally, this problem has been tackled by separating the *correspondence* problem, i.e. that of establishing which points in the images are the projections in the retina of the same 3D point, and the *reconstruction* problem, i.e. that of reconstructing the 3D shapes from the correspondences.

The approach described in this chapter does not separate the two problems. Given a set of objects (surfaces) in 3D space, we propose a function that measures the adequacy of these objects with the measured images. The problem is then to find a set of surfaces in the scene which maximize our function. This can be achieved by a variational approach. A level set implementation is described and results are given on synthetic and real images.

18.1 Introduction and preliminaries

The idea that is put forward in this paper is that the methods of curve and surface evolutions which have been developed in computer vision under the name of snakes [263] and then reformulated by Caselles, Kimmel and Sapiro [84] and Kichenassamy et al. [270] in the context of PDE driven evolving curves can be used effectively for solving 3D vision problems such as stereo and motion analysis.

As a first step in this direction we present a mathematical analysis of the stereo problem in this context as well as a level set implementation.

The problem of curve evolution driven by a PDE has been recently studied both from the theoretical standpoint [204, 220, 461] and from the viewpoint of implementation [401] with the development of level set methods that can efficiently

and robustly solve those PDE's. The problem of surface evolution has been less touched upon even though some preliminary results have been obtained [86].

The path we will follow to attack the stereo problem from that angle is, not surprisingly, a variational one. In a nutshell, we will describe the stereo problem (to be defined more precisely later) as the minimization of a functional with respect to some parameters (describing the geometry of the scene); we will compute the Euler-Lagrange equations of this functional, thereby obtaining a set of necessary conditions, in effect a set of partial differential equations, which we will solve as a time evolution problem by a level set method.

Stereo is a problem that has received considerable attention for decades in the psychophysical, neurophysiological and, more recently, in the computer vision literatures. It is impossible to cite all the published work here, we will simply refer the reader to some basic books on the subject [260, 223, 237, 243, 186]. To explain the problem of stereo from the computational standpoint, we will refer the reader to figure 18.1.a. Two, may be more, images of the world are taken simultaneously. The problem is, given those images, to recover the geometry of the scene. Given the fact that the relative positions and orientations and the internal parameters of the cameras are known which we will assume in this article (the cameras are then said to be calibrated [186]), the problem is essentially (but not only) one of establishing correspondences between the views: one speaks about the *matching* or *corespondence* problem. The matching problem is usually solved by setting up a matching functional for which one then tries to find extrema. Once a pixel in view i has been identified as being the image of the same scene point as another pixel in view j , the 3D point can then be reconstructed by intersecting the corresponding optical rays (see figure 18.1.a again): this is the *reconstruction* problem.

The approach described in this chapter is different in the sense that it does not separate the *matching* and the *reconstruction* problems. Given a set of objects (surfaces) in 3D space, we propose a function that measure the adequacy of these objects with the measured images. The problem is then to find a set of surfaces in the scene which maximize our function. This can be achieved by establishing the corresponding Euler-Lagrange equations with respect to the objects shape and applying a steepest infinitesimal gradient descent by solving the resulting parabolic equations. These parabolic equations describe the way the surfaces of the objects evolve toward a local maximum of our function.

We represent the objects shapes by the signed distance function to their boundary, this allows us to perform the implementation of the evolution equations through the level set method. Among the advantages of the level set method, the more important here is that the changes in the topology of the evolving surface (mainly its number of connected components) are handled automatically: typically, the initial shape is a large sphere enclosing the whole scene which shrinks down to the objects surfaces, splitting itself seamlessly into several surfaces whenever this is required. An unfortunate feature of the level set method is that the evolution does not in general preserve the distance function. We show that

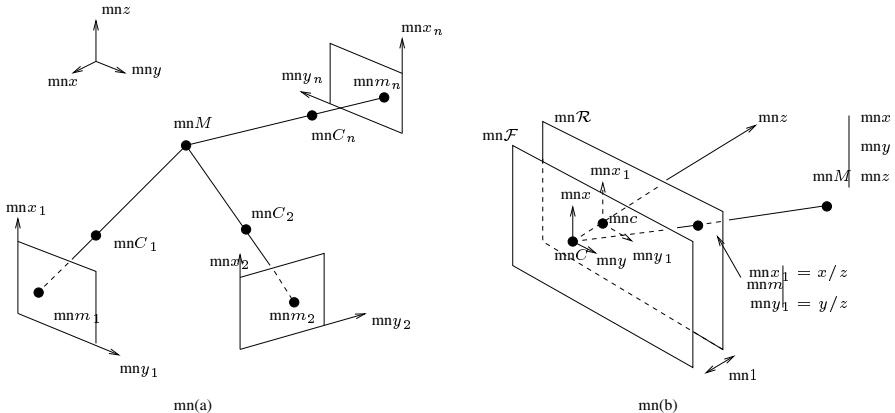


Figure 18.1. (a) The multi-camera stereo vision problem is, given a pixel m_1 in image 1, to find the corresponding pixel m_2 in image 2, ..., the corresponding pixel m_n in image n , i.e. the pixels which are the images of the same 3D point M . Once such a correspondence has been established, the point M can be reconstructed by intersecting the optical rays $\langle m_i, C_i \rangle$, $i = 1, \dots, n$. (b) The focal plane (x, y) is parallel to the retinal plane (x_1, y_1) and at a unit distance from it.

a modification of the level set equation can guarantee this invariance, resulting in a significant gain in computing time.

In order to go any further, we need to be a little more specific about the process of image formation. We will assume here that the cameras perform a perspective projection of the 3D world on the retinal plane as shown in figure 18.1.b. The optical center, noted C in the figure, is the center of projection and the image of the 3D point M is the pixel m at the intersection of the optical ray $\langle C, m \rangle$ and the retinal plane \mathcal{R} . As described in many recent papers in computer vision, this operation can be conveniently described in projective geometry by a matrix operation. The projective coordinates of the pixel m (a 3×1 vector) are obtained by applying a 3×4 matrix \mathbf{P}_1 to the projective coordinates of the 3D point M (a 4×1 vector). This matrix is called the perspective projection matrix. If we express the matrix \mathbf{P}_1 in the coordinate system (C, x, y, z) shown in the figure 18.1.b, it then takes a very simple form:

$$\mathbf{P}_1 = [\mathbf{I}_3 \ 0]$$

where \mathbf{I}_3 is the 3×3 identity matrix. If we now move the camera by applying to it a rigid transformation described by the rotation matrix \mathbf{R} and the translation vector \mathbf{t} , the expression of the matrix \mathbf{P} changes accordingly and becomes:

$$\mathbf{P}_2 = [\mathbf{R}^T \ -\mathbf{R}^T \mathbf{t}]$$

With these preliminaries in mind we are ready to proceed with our program which we will do by progressing along two related axes. The first axis is that of object complexity, the second axis is that of matching functional complexity.

In section 18.2, we first consider a simple object model which is well adapted to the binocular stereo case where it is natural to consider that the objects in the scene can be considered mathematically as forming the graph of an unknown smooth function (the depth function in the language of computer vision). We start with an extremely simplified matching criterion which allows us to convey to the reader the flavor of the ideas that we are trying to push here. We then move to a more sophisticated albeit classical matching criterion which is at the heart of the techniques known in computer vision as correlation based methods. Within the framework of this model we study two related shape models. Finally in section 18.3 we introduce a more general shape model in which we do not assume anymore that the objects are the graph of a function and model them instead as a set of general smooth surfaces in three dimensional space. The next steps would be to relax the smoothness assumption and to consider more complex matching criteria but we do not address them here.

Section 18.4 goes into some important implementation details, while section 18.5 presents results with synthetic and real images.

Let us give some definitions and fix notations. Images are denoted by I_k , k taking some integer values which indicate the camera with which the image has been acquired. They are considered as smooth (i.e. C^2 , twice continuously differentiable) functions of pixels m_k whose coordinates are defined in some orthonormal image coordinate systems (x_k, y_k) which are assumed to be known. We note $I_k(m_k)$ or $I_k(x_k, y_k)$ the intensity value in image k at pixel m_k . We will use the first and second order derivatives of these functions, i.e. the gradient ∇I_k , a 2×1 vector equal to $[\frac{\partial I_k}{\partial x_k}, \frac{\partial I_k}{\partial y_k}]^T$, and the Hessian \mathbf{H}_k , a 2×2 symmetric matrix.

The pixels in the images are considered to be functions of the 3D geometry of the scene, i.e. of some 3D point M on the surface of an object in the scene, and of the unit normal vector \mathbf{N} to the surface at this point.

Vectors and matrices are generally represented in boldface, e.g. \mathbf{x} . The dot or inner product of two vectors \mathbf{x} and \mathbf{y} is denoted by $\mathbf{x} \cdot \mathbf{y}$. The cross-product of two 3×1 vectors \mathbf{x} and \mathbf{y} is noted $\mathbf{x} \times \mathbf{y}$.

Partial derivatives are represented either with the ∂ symbol, e.g. $\frac{\partial f}{\partial \mathbf{x}}$, or with a lower index, e.g. $f_{\mathbf{x}}$.

Our approach is an extension of previous work by Robert et al. and Robert and Deriche, [438, 437], where the idea of using a variational approach for solving the stereo problem was first proposed in the classical Tikhonov regularization framework and then by using regularization functions more proper to preserve discontinuities. Our work can be seen as a 3D extension of the approach proposed in [152] where we limit ourselves to the binocular case, to finding cross-sections of the objects with a fixed plane, and do not take into account the orientation of the tangent plane to the object. Preliminary versions of our work can be found in [188, 189, 190, 266, 218].

18.2 The simplified models

Let us now describe the different object models and the functions that measure the adequacy of the objects with the images. We will proceed from the simplest to the most sophisticated one. Note that, in order to turn our variational problem into a minimization problem, we consider error measures instead of adequacy measures.

18.2.1 A simple object and matching model

This section introduces in a simplified framework some of the basic ideas of this paper. We assume, and it is the first important assumption, that the objects which are being imaged by the stereo rig (a binocular stereo system) are modeled as the graph of an unknown smooth function $z = f(x, y)$ defined in the first retinal plane which we are trying to estimate. A point M of coordinates $[x, y, f(x, y)]^T$ is seen as two pixels m_1 and m_2 whose coordinates $(g_i(x, y), h_i(x, y))$, $i = 1, 2$, can be easily computed as functions of $x, y, f(x, y)$ and the coefficients of the perspective projection matrices \mathbf{P}_1 and \mathbf{P}_2 . Let I_1 and I_2 be the intensities of the two images. Assuming, and it is the second important assumption, that the objects are perfectly Lambertian, we must have $I_1(m_1) = I_2(m_2)$ for all pixels in correspondence, i.e. which are the images of the same 3D point.

This reasoning immediately leads to the variational problem of finding a suitable function f defined, to be rigorous, over an open subset of the focal plane of the first camera and which minimizes the following integral:

$$C_1(f) = \int \int (I_1(m_1(x, y)) - I_2(m_2(x, y)))^2 dx dy \quad (18.1)$$

computed over the previous open subset. Our first variational problem is thus to find a function f in some suitable functional space that minimizes the error measure $C_1(f)$. The corresponding Euler-Lagrange equation is readily obtained:

$$(I_1 - I_2)(\nabla I_1 \cdot \frac{\partial \mathbf{m}_1}{\partial f} - \nabla I_2 \cdot \frac{\partial \mathbf{m}_2}{\partial f}) = 0 \quad (18.2)$$

The values of $\frac{\partial \mathbf{m}_1}{\partial f}$ and $\frac{\partial \mathbf{m}_2}{\partial f}$ are functions of f which are easily computed. The terms involving I_1 and I_2 are computed from the images. In order to solve (18.2) one can adopt a number of strategies.

One standard strategy is to consider that the function f is also a function $f(x, y, t)$ of time and to solve the following PDE:

$$f_t = \varphi(f)$$

where $\varphi(f)$ is equal to the left hand side of (18.2), with some initial condition $f(x, y, 0) = f_0(x, y)$. We thus see for the first time appear the idea that the shape of the objects in the scene, described by the function f , is obtained by allowing a surface of equation $z = f(x, y, t)$ to evolve over time, starting from some initial configuration $z = f(x, y, 0)$, according to some PDE, to hopefully converge toward the real shape of the objects in the scene when time goes to infinity. This

convergence is driven by the data, i.e. the images, as expressed by the error criterion (18.1) or the Euler-Lagrange term $\varphi(f)$. It is known that if care is not taken, for example by adding a regularizing term to (18.1), the solution f is likely not to be smooth and therefore any noise in the images may cause the solution to differ widely from the real objects. This is more or less the approach taken in [438, 437]. We will postpone the solution of this problem until section 18.3 and in fact solve it differently from the usual way which consists in adding a regularization term to $C_1(f)$.

It is clear that the error measure (18.1) is a bit simple for practical applications. We can extend it in at least two ways. The first is to replace the difference of intensities by a measure of correlation, the hypothesis being that the scene is made of fronto parallel planes. The second is to relax this hypothesis and to take into account the orientation of the tangent plane to the surface of the object. We explore those two avenues in the next two sections.

18.2.2 Fronto parallel correlation functional

To each pair of values (x, y) , corresponds a 3D point M , $\mathbf{M} = [x, y, f(x, y)]^T$ which defines two image points m_1 and m_2 as in the previous section. We can then classically define the unnormalized cross-correlation between I_1 and I_2 at the pixels m_1 and m_2 . We note this cross-correlation $\langle I_1, I_2 \rangle(f, x, y)$ to acknowledge its analogy with an inner product and the fact that it depends upon M :

$$\langle I_1, I_2 \rangle(f, x, y) = \frac{1}{4pq} \int_{-p}^{+p} \int_{-q}^{+q} (I_1(m_1 + m) - \overline{I_1}(m_1)) (I_2(m_2 + m) - \overline{I_2}(m_2)) dm \quad (18.3)$$

equation where the averages $\overline{I_1}$ and $\overline{I_2}$ are classically defined as:

$$\overline{I_k}(m_k) = \frac{1}{4pq} \int_{-p}^{+p} \int_{-q}^{+q} I_k(m_k + m') dm' \quad k = 1, 2 \quad (18.4)$$

Finally, we note $|I|^2$ the quantity $\langle I, I \rangle$. Note that $\langle I_1, I_2 \rangle = \langle I_2, I_1 \rangle$.

To simplify notations we write \int^* instead of $\frac{1}{4pq} \int_{-p}^{+p} \int_{-q}^{+q}$ and define a matching functional which is the integral with respect to x and y of minus the normalized cross-correlation score $-\frac{\langle I_1, I_2 \rangle}{|I_1| \cdot |I_2|}$:

$$C_2(f) = - \int \int \frac{\langle I_1, I_2 \rangle}{|I_1| \cdot |I_2|} dx dy = \int \int {}_2\Phi(f, x, y) dx dy \quad (18.5)$$

the integral being computed, as in the previous section, over an open set of the focal plane of the first camera. The functional ${}_2\Phi$ is $-\frac{\langle I_1, I_2 \rangle}{|I_1| \cdot |I_2|}(f, x, y)$. This quantity varies between -1 and +1, -1 indicating the maximum correlation. We have to compute its derivative with respect to f in order to obtain the Euler-Lagrange equation of the problem. The computations are simple but a little fastidious. They can be found in [188]. We could then proceed to the corresponding steepest gradi-

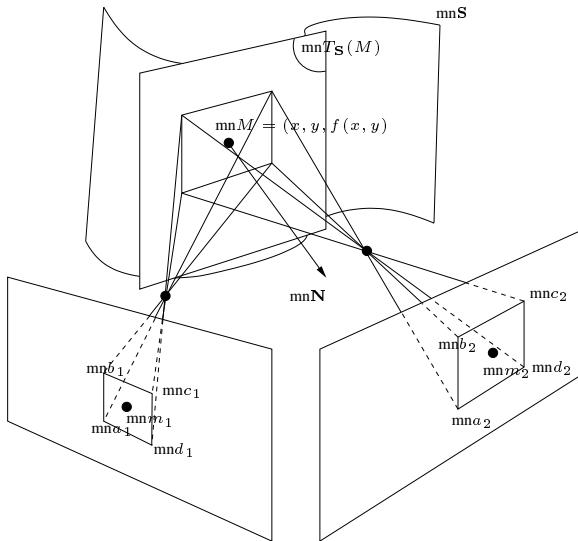


Figure 18.2. The square window (a_1, b_1, c_1, d_1) in the first image is back projected onto the tangent plane to the object S at point M and reprojected in the retinal plane of the second camera where it is generally not square. The observation is that the distortion between (a_1, b_1, c_1, d_1) and (a_2, b_2, c_2, d_2) can be described by a collineation which is function of M and the normal N to the surface of the object.

ent descent as mentioned in the previous section. But we will not pursue this task and explore rather a better functional.

18.2.3 Taking into account the tangent plane to the object

We now take into account the fact that the rectangular window centered at m_2 should be the image in the second retina of the back-projection on the tangent plane to the object at the point $M = (x, y, f(x, y))$ of the rectangular window centered at m_1 (see figure 18.2). In essence, we approximate the object S in a neighborhood of M by its tangent plane but without assuming, as in the previous section, that this plane is fronto parallel, and in fact also that the retinal planes of the two cameras are identical. Let us first study the correspondence induced by this plane between the two images.

Image correspondences induced by a plane

Let us consider a plane of equation $\mathbf{N}^T \mathbf{M} - d = 0$ in the coordinate system of the first camera. d is the algebraic distance of the origin of coordinates to that plane and \mathbf{N} is a unit vector normal to the plane. This plane induces a projective transformation between the two image planes. This correspondence plays an essential role in the sequel.

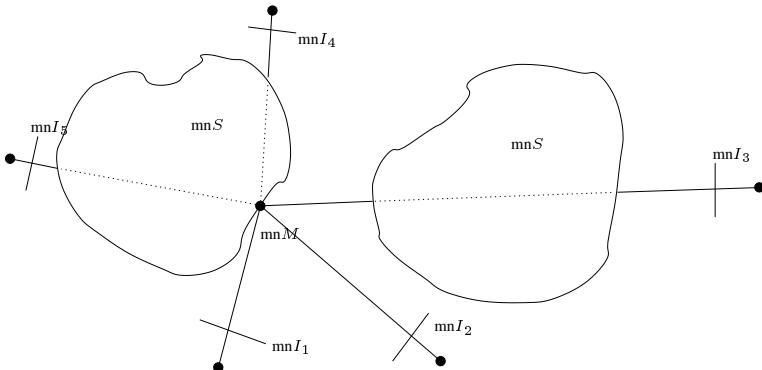


Figure 18.3. Occlusion and visibility are taken into account: only the cameras seeing the points on the current surface are used in the error criterion, thus avoiding the computation of irrelevant correlation. Here, only the cameras 1 and 2 are considered for point M

To see why we obtain a projective transformation, let M be a 3D point in that plane, \mathbf{M}_1 and \mathbf{M}_2 be the two 3D vectors representing this point in the coordinate systems attached to the first and second cameras, respectively. These two 3×1 vectors are actually coordinate vectors of the two pixels m_1 and m_2 seen as projective points (see Sect. 18.1). Furthermore, they are related by the following equation:

$$\mathbf{M}_2 = \mathbf{R}^T(\mathbf{M}_1 - \mathbf{t})$$

Since M belongs to the plane, $\mathbf{N}^T \mathbf{M}_1 = d$, and we have:

$$\mathbf{M}_2 = (\mathbf{R}^T - \frac{\mathbf{R}^T \mathbf{t} \mathbf{N}^T}{d}) \mathbf{M}_1$$

which precisely expresses the fact that the two pixels m_1 and m_2 are related by a collineation, or projective transformation K . The 3×3 matrix representing this collineation is $(\mathbf{R}^T - \frac{\mathbf{R}^T \mathbf{t} \mathbf{N}^T}{d})$. This transformation is one to one except when the plane goes through one of the two optical centers when it becomes degenerate. We will assume that it does not go through either one of those two points and since the matrix of K is only defined up to a scale factor we might as well take it equal to:

$$\mathbf{K} = d\mathbf{R}^T - \mathbf{R}^T \mathbf{t} \mathbf{N}^T \quad (18.6)$$

The new criterion and its Euler-Lagrange equations

We just saw that a plane induces a collineation between the two retinal planes. This is the basis of the method proposed in [152] although for a very different purpose. The window alluded to in the introduction to this section is therefore the image by the collineation induced by the tangent plane of the rectangular window in image 1. This collineation is a function of the point M and of the normal to the object at M . It is therefore a function of f and ∇f that we denote by K . It

satisfies the condition $K(m_1) = m_2$. The inner product (18.3) must be modified as follows:

$$\langle I_1, I_2 \rangle(f, \nabla f, x, y) = \int^{*} (I_1(m_1 + m) - \overline{I_1}(m_1)) \\ (I_2(K(m_1 + m)) - \overline{I_2}(m_2)) dm, \quad (18.7)$$

Note that, the definition of $\langle I_1, I_2 \rangle$ is no longer symmetric, because of K . This problem could be addressed as mentioned in [190]. Yet, we will see in section 18.3 that our final error measure will solve it in a more natural way.

We now want to minimize the following error measure:

$$C_3(f, \nabla f) = \int \int {}_3\Phi(f, \nabla f, x, y) dx dy \\ \text{with } {}_3\Phi = -\frac{\langle I_1, I_2 \rangle}{|I_1| \cdot |I_2|}(f, \nabla f, x, y) \quad (18.8)$$

Since the functional ${}_3\Phi$ now depends upon both f and ∇f , its Euler-Lagrange equations have the form ${}_3\Phi_f - \text{div}({}_3\Phi_{\nabla f}) = 0$. We must therefore recompute ${}_3\Phi_f$ to take into account the new dependency of K upon f and compute ${}_3\Phi_{\nabla f}$.

We could simplify the computations by assuming that the collineation K can be well approximated by an affine transformation. Because of the condition $K(m_1) = m_2$, this transformation can be written:

$$K(m_1 + m) \approx m_2 + \mathbf{A}m$$

where \mathbf{A} is a 2×2 matrix depending upon f and ∇f .

In practice this approximation is often sufficient and we will assume that it is valid in what follows. We will not pursue this derivation (see [188] for details) since we present in section 18.3 a more elaborate model that encompasses this one and for which we will perform the corresponding computation.

18.3 The complete model

We are now ready to consider the case when the objects in the scene are not defined as the graph of a function of x and y as in the previous sections, but as a surface \hat{S} of \mathbf{R}^3 which we assume to be smooth, i.e. C^2 . Note that, by relaxing the graph assumption, it potentially allows us to use an arbitrary number of cameras to analyze the scene. Let us consider a family of smooth surfaces $S(t)$ where t is the time. Our goal is, starting from an initial surface S_0 , to derive a partial differential equation

$$\mathbf{S}_t = \beta \mathbf{N}, \quad (18.9)$$

where \mathbf{S} denotes a point on S and \mathbf{N} is the inner unit normal to the surface at \mathbf{S} , which, when solved with initial condition $S(0) = S_0$, will yield a solution that closely approximates \hat{S} .

The function β is determined by the matching functional that we minimize in order to solve the stereo problem. We define such a functional in the next paragraph. An interesting point is that, if β is intrinsic (see below), the evolution equation (18.9) can be solved using the level set method which has the advantage of coping automatically with several objects in the scene.

Using the same ideas as in the section 18.2.3, we can define the following error measure:

$$C_4(\mathbf{S}, \mathbf{N}) = \int_S {}_4\Phi(\mathbf{S}, \mathbf{N}) d\sigma \quad (18.10)$$

where the integration is carried over with respect to the area element $d\sigma$ on the surface S and where ${}_4\Phi(\mathbf{S}, \mathbf{N})$ measures the error made when assuming that there is an object at point \mathbf{S} with a normal \mathbf{N} . Here, $d\sigma$ plays the role of $dx dy$ in our previous analysis, \mathbf{S} that of f , and \mathbf{N} that of ∇f . Note that this is a significant departure from what we had before because we are multiplying our error criterion Φ with an area element on the surface. As we will see, this has two dramatic consequences, like in the geodesic snakes approach [84]:

1. it automatically regularizes the variational problem, and
2. it makes the problem intrinsic, i.e. independent from the parametrization of the surface, thus allowing a level set implementation.

18.3.1 Error criterion

While we still compute our error criterion Φ from the correlation as in the section 18.2.3, we will see that some changes are needed. First, we address the problems of regularization and symmetry that we have left aside so far. Then, we deal with a more complex case, using more than two cameras and taking occlusions into account.

Given a point \mathbf{S} on S and its normal \mathbf{N} , we could still let ${}_4\Phi$ be the opposite of $\rho_{12} = \frac{\langle I_1, I_2 \rangle}{|I_1| \cdot |I_2|}(\mathbf{S}, \mathbf{N}, m_1)$, the correlation between image I_1 and image I_2 taken at point m_1 , image of \mathbf{S} in I_1 , defined as in section 18.2.3. Yet, it is now required that ${}_4\Phi$ stays positive: because we want to minimize its integral over S with respect to $d\sigma$, the surface would tend to maximize its area in the regions where ${}_4\Phi$ is negative, thus yielding an unstable evolution. ${}_4\Phi$ should also be small when correlation is high, otherwise the surface would tend to be as smallest as possible, regardless the correlation measure. Like in the geodesic snakes approach [84], we take ${}_4\Phi = g(\rho_{12})$, where $g : [-1, 1] \rightarrow \mathbf{R}^+$ is some positive decreasing function with $g(1) = 0$. We will see in section 18.3.3 that doing so solves the regularization problem.

The symmetry of the correlation could also be recovered, just taking ${}_4\Phi = g(\frac{1}{2}(\rho_{12} + \rho_{21}))$, the mean correlation $\frac{1}{2}(\rho_{12} + \rho_{21})$ being a symmetric value between -1 and $+1$. Actually, we could use the same idea to address the multi-camera problem. Given n images I_n ($1 \leq i \leq n$), we could use $g(\frac{1}{n(n-1)} \sum_{i \neq j} \rho_{ij})$, but we go further and model *visibility and occlusion*. This is

essential to avoid making mistakes by using incorrect information from cameras which do not see some parts of the objects (see figure 18.3). Given the surface S and a point \mathbf{S} on it, it is possible to determine the cameras where \mathbf{S} is visible, i.e. not hidden by another part of S , and to consider the correlation ratio for those cameras only. Let $\Gamma(\mathbf{S}, S)$ be the set of those cameras, we thus take:

$$_4\Phi(\mathbf{S}, \mathbf{N}) = g\left(\frac{1}{|\Gamma|(|\Gamma| - 1)} \sum_{i,j \in \Gamma, i \neq j} \rho_{ij}\right) \quad (18.11)$$

This will be our symmetric, regularizing, multi-camera error criterion, taking *visibility* into account.

18.3.2 The Euler-Lagrange equations

In order to set up a surface evolution equation such as (18.9) and implement it by a level-set method, we have to write the Euler-Lagrange equations of the variational problem (18.10) and consider their component $-\beta$ along the normal to the surface. Although technically more complicated, this is similar to the derivations in the previous section. This is all fairly straightforward except for the announced result that the resulting value of β is *intrinsic* i.e. does not depend upon the parametrization of the surface S .

We have in fact proved a more general result. Let $\Phi : \mathbf{R}^3 \times \mathbf{R}^3 \rightarrow \mathbf{R}$ be a smooth function of class at least C^2 defined on the surface S and depending upon the point \mathbf{S} and the unit normal \mathbf{N} at this point, which we denote by $\Phi(\mathbf{S}, \mathbf{N})$. Let us now consider the following error measure:

$$C(\mathbf{S}, \mathbf{N}) = \int_S \Phi(\mathbf{S}, \mathbf{N}) d\sigma \quad (18.12)$$

We prove in [188] the following theorem:

THEOREM 4 *Under the assumptions of smoothness that have been made for the function Φ and the surface S , the component of the Euler-Lagrange equations for criterion (18.12) along the normal to the surface is intrinsic, i.e. it does not depend upon the parametrization of S . Furthermore, this component is equal to*

$$\begin{aligned} -\beta &= (\Phi_{\mathbf{S}} + 2H\Phi_{\mathbf{N}})\mathbf{N} - 2H\Phi \\ &\quad + \text{Trace}((\Phi_{\mathbf{SN}})_{T_S} + d\mathbf{N} \circ (\Phi_{\mathbf{NN}})_{T_S}) \end{aligned} \quad (18.13)$$

where all quantities are evaluated at the point \mathbf{S} of normal \mathbf{N} of the surface, T_S is the tangent plane to the surface at the point \mathbf{S} . $d\mathbf{N}$ is the differential of the Gauss map of the surface, H is its mean curvature, $\Phi_{\mathbf{SN}}$ and $\Phi_{\mathbf{NN}}$ are the second order derivatives of Φ , $(\Phi_{\mathbf{SN}})_{T_S}$ and $(\Phi_{\mathbf{NN}})_{T_S}$ their restrictions to the tangent plane T_S of the surface at the point S .

Note that the error criterion (18.10) is of the form (18.12). According to the theorem 4, in order to compute the velocity β along the normal in the evolution equation (18.9), we only need to compute $\Phi_{\mathbf{S}}$, $\Phi_{\mathbf{N}}$, $\Phi_{\mathbf{SN}}$ and $\Phi_{\mathbf{NN}}$ as well as

the second order intrinsic differential properties of the surface S . The problem is obviously broken down into the problem of computing the corresponding derivatives of the ρ_{ij} 's, which, for the first order derivatives is extremely similar to what we have done in the section 18.2.3. The computations are carried out in [188].

18.3.3 The normal velocity and the regularization property

The normal velocity β given by equation (18.13) looks a bit complicated. Actually, this is just an extension of the one obtained for 3D geodesic active contours by Caselles, Kimmel, Sapiro and Sbert [86] to the case where the error criterion Φ depends not only upon the point S but also upon the normal N . Had we taken some error criterion $\Phi(S)$ with no dependency upon the normal (e.g. by restricting the model to the fronto parallel case), we would have obtained $\beta = 2H\Phi - \nabla\Phi \cdot N$ as in [86].

Our case is more complicated since our error criterion is a function $\Phi(S, N)$ of the point and its normal but we see that our normal velocity is the sum of the following three terms:

$$\beta = \begin{cases} 2H\Phi & \text{regularization term} \\ -(\Phi_S + 2H\Phi_N)N & \text{first order data term} \\ -\text{Trace}((\Phi_{SN})_{T_S} + dN \circ (\Phi_{NN})_{T_S}) & \text{higher order data term} \end{cases} \quad (18.14)$$

The regularization term is the same as usual. The first order data term is an extension of the one in [84]. The higher order data term is certainly a bit tricky to understand. Nevertheless, our experiments seem to indicate that its influence is negligible.

We see that, as announced, the regularization problem has been solved in a natural way by using an intrinsic definition of the error criterion. We note the fact that Φ should be positive otherwise the curvature term induces instability, and that it should be small for high correlation values this preventing the surface to disappear, just like it does with the mean curvature flow [204].

18.3.4 Correctness

In the case of the active contours [84], the authors proved the existence of a unique viscosity solution to their evolution equation. In our case, it seems that existence and uniqueness of a viscosity solution is not immediate. While we are aware that proving such results is important, this is not the only research direction to investigate.

In effect, the problem of shape recovery from images using variational methods is still in its early days. As far as we know, the ideal energy is still to be found. This energy should certainly take into account stereo-correlation, but also contours [143, 122, 212, 64, 303], shape from shading [244], shape from texture [124, 57, 51, 265] and the ideas around stereo segmentation [591].

An interesting parallel can be drawn with the problem of contour extraction where a first significant progress was achieved with the proposal of the snake model [263] which was mostly heuristic. Formalisation and mathematical proofs of well-posedness came later, as a second stage, with the geodesic snakes model [84, 86]. In the case of stereo we are still struggling to reach the corresponding first stage.

18.4 Level Set Implementation

Let us now go into some very important implementation details. Our normal velocity β being intrinsic, we use the level set method. Among the difficulties we will have to overcome, the so-called *velocity extension problem* will be crucial. We give a solution to this problem that also eliminates another problem attached to the level set method, the need to reinitialize periodically the distance function.

18.4.1 The velocity extension problem

Let us denote by $u(\mathbf{X}, t)$ the standard level set function, the zero level set of which is our moving surface S . The evolution equation (18.9) thus becomes an equation for u :

$$\frac{\partial u}{\partial t}(\mathbf{X}, t) = \beta(\mathbf{X}, t) |\nabla u(\mathbf{X}, t)| \quad (18.15)$$

It is important to notice that β in (18.15) is defined in \mathbf{R}^3 whereas in (18.9) it is defined on the surface S . The extension of β from S to the whole domain \mathbf{R}^3 is a crucial point for the analysis and implementation of (18.15). There are mainly two ways of doing this.

(i) Most of the time this extension is natural. For example, if $\beta(S) = H$, the mean curvature of S in (18.9), one can choose $\beta(\mathbf{X}) = H_u$, the mean curvature of the level set of u passing through \mathbf{X} in (18.15). In the geodesic active contours approach [86], $\beta(S) = 2H\Phi - \nabla\Phi.\mathbf{N}$, where $\Phi(\mathbf{X}) = g(|\nabla I(\mathbf{X})|)$ for a given image I and some real function g . Although one only wants to minimize the sum of Φ along S , just because Φ is defined in \mathbf{R}^3 , $\beta(\mathbf{X}) = 2H_u\Phi - \nabla\Phi.\mathbf{N}_u$ is often taken for equation (18.15). Everything happens as if every level set of u was evolving according to the surface equation (18.9), i.e. was trying to reach a contour.

(ii) In some cases [108, 498], this extension is not possible. Then one may choose to assign to $\beta(\mathbf{X})$ in (18.15) the value of $\beta(S)$ in (18.9) where S is the closest point to \mathbf{X} belonging to S . One may also choose to extend β so that it remains constant along the characteristics of u (the characteristics of u are the integral curves of ∇u). This is what is done in by Adalsteinsson and Sethian in [3], and by Peng et al. in [421], where a PDE based procedure extends β .

Note that, if u is the signed distance to S , then the characteristics of u are lines passing through the closest point of S , so that the two previous choices become one (see next section).

Our evolution velocity (18.14) only makes sense on S . It does not seem either reasonable or meaningful to consider some correlation measure or some visibility test for all the level sets of u : we are in case **(ii)**. Another point is that the computation of our β is expensive. In our case, computing β on S and extending it will be faster than computing it everywhere in the domain of u , even if this domain is only a narrow-band around S [2].

18.4.2 Preserving the distance function

In [218], we propose to use the following equation

$$\begin{aligned} \frac{\partial u}{\partial t}(\mathbf{X}, t) &= \beta(\mathbf{X} - u\nabla u) \\ u(\mathbf{X}, 0) &= \text{signed distance function to } S(0) \end{aligned} \quad (18.16)$$

instead of the classical level set one (18.15). We show that u remains the signed distance function to its zero level, the evolution of which is still the desired one (18.9). Moreover, $\mathbf{X} - u\nabla u$ is the closest point to \mathbf{X} on S , and our PDE is exactly (18.15) when β is extended via the closest point principle or, which is equivalent in this case, via the characteristics of u (Remember that $|\nabla u| = 1$ for the distance function). A detailed description of an implementation can be found in [218]. Also note that standard implementations of the level set method periodically reinitialize the function u to be a distance function. With our method, this step becomes unnecessary.

18.4.3 Error criterion

The estimation of β requires that of ${}_4\Phi$ and its derivatives. The function g in the definition (18.11) of ${}_4\Phi$ can be very simple. Our implementation uses $g(x) = 1 - x$.

Despite the use of a narrow-band and the fact that β is only computed on S and then extended, one may still find the computation of β too expensive. Depending upon the images, one may want to use only two cameras instead of all in the set Γ in (18.11). Good candidates are the cameras of Γ whose optical axes are the closest to the normal to the surface S at point \mathbf{S} . Experiments indicate that the higher order term for β in equation (18.14) can be ignored without any significant difference in the results.

18.4.4 Visibility

Estimating ${}_4\Phi$ requires the crucial step of computing the hidden parts of the surface for all the cameras. We first extract the zero level set of u as a triangulated

mesh using the marching cube algorithm [320]. We then use a Z-buffer algorithm [89] to project this mesh onto each image with visibility information. This is not the most expensive part of our method. Purists will object that an evolution of u relying on the extraction of its zero level set does not agree with the philosophy of the level set method. It should probably be easy to adapt the work on visibility in an implicit framework by Tsai et al. [530].

18.4.5 Why and how well does it work?

We will see in section 18.5, that the results are promising. Yet the role of visibility in the reconstruction process is still an open question. Determining what are the local minima and to what extent they are avoided is also a hard problem. As a preliminary step toward answering these questions, we make a few remarks:

(i) Correlation based techniques are often fooled by false matches. In our case, we observed that the points in space inducing high correlation scores were often isolated. As a result, thanks to the regularization term, the evolving surface does not get “stuck” at such points.

(ii) Local minima may be avoided using a multi-scale approach. The problem may be solved at a rough 3D scale first and its solution refined at finer scales. This technique could also be used to increase the convergence speed, although adaptive methods [500] seem an even better choice.

(iii) Depending on the images, the derivatives of the correlation function may not help the minimization process when the surface is too far from a minimum, i.e. may fail to predict the right direction of descent. In such cases, the regularization term, acting as a deflating term, helps to alleviate the problem. Thus one should choose an initial surface surrounding the objects to be recovered. In the active contours case, smoothing the images is often used to make the data term more efficient far from the contours. In our case, doing so would delete the texture information needed for the correlation. Other texture matching criteria are being investigated [123] to address this problem.

18.4.6 A simple but important case

Let us consider the case where the cameras are set up in such a way that every part of the objects is seen by at least two of them that are more or less close to each other. This happens when many cameras are used, or when the images are acquired with stereo rigs. If so we can, without much loss of information, use only those two cameras when estimating Φ , or if several choices are possible, the two cameras that are the most “in front” of the considered point S . We can even rectify [186] the two corresponding images as a preliminary stage. Then, not only can the correlation be efficiently computed but it also does not depend anymore upon the normal to the surface, a square window in the first image approximately

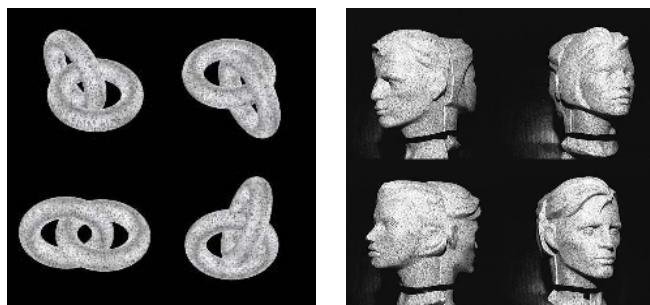


Figure 18.4. Multi-camera images of 3D objects. Left: synthetic images of two tori (24 images). Right: real images of two heads (18 images).

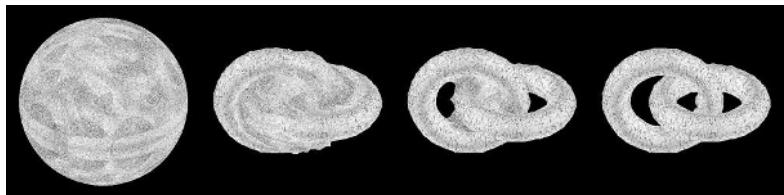


Figure 18.5. Four steps of the stereo algorithm for the two tori.

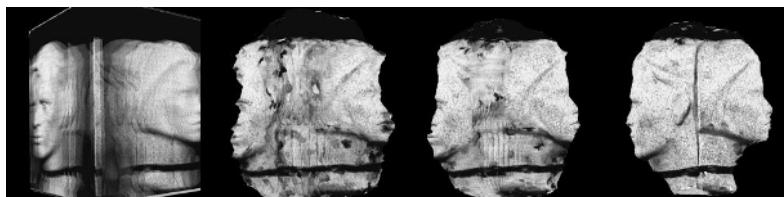


Figure 18.6. Four steps of the stereo algorithm for the two heads.

corresponding to a window of the same size in the second image¹. As a result, such a situation yield fast correlation as well as a simplified normal velocity, similat to the active contours case: $\beta = 2H\Phi - \nabla\Phi.\mathbf{N}$

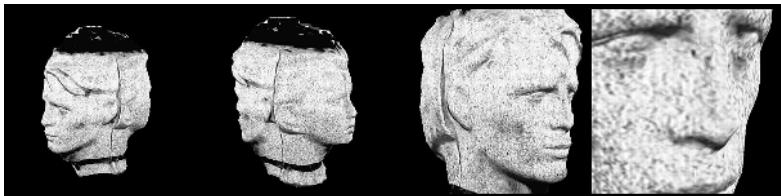


Figure 18.7. Some views of the reconstructed 3D object.



Figure 18.8. Eight of the twenty images used to reconstruct a human head despite bad lighting conditions and little texture information (courtesy of the RealViZ company).

18.5 Results

We now present some results obtained with both synthetic and real images.

We first synthesized images of two tori, seen from enough points of views so that each part was seen at least twice (a total of 24 views – figure 18.4 left). See how the initial surface splits and how even the internal parts are reconstructed (figure 18.5).

We then used real images of two heads acquired by rotating them in front of the camera (figure 18.4 right). All visible parts (i.e.. neither the top nor the bottom) are correctly recovered (figures 18.6 and 18.7)

We finally applied the method to twenty images of a real human head (figure 18.8). As expected, the poorly textured regions such as the hairs were not reconstructed well. The results shown in figure 18.9, where these regions

¹The choice of the best two cameras depends upon the normal and even upon the whole surface through the visibility estimation. Yet, the Euler-Lagrange equations have been derived given such a choice.

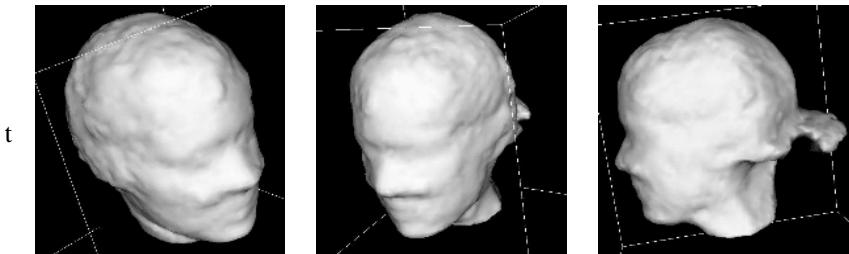


Figure 18.9. Three views of the reconstructed shape. Note that the recovery of poorly textured areas like the hairs is made possible by an extension of the method presented in this chapter and described in [267].

are nonetheless reconstructed, were obtained with a modified version of the functional to minimize, taking into account the contours information (see [267]).

18.6 Conclusion

We have presented in this chapter a novel approach for solving the stereo problem from an arbitrary number of views. It is based upon a variational principle that must be satisfied by the surfaces of the objects to be reconstructed. The design of the variational principle allows us to cleanly incorporate the hypotheses we make about the objects in the scene (smooth, opaque, lambertian), the methods used to rate the correspondences between image points (mostly cross-correlation). The Euler-Lagrange equations which are deduced from the variational principle provide a set of PDE's that are used to deform an initial set of surfaces which move toward the objects to be detected. The level set implementation of these PDE's provides an efficient and robust way of achieving the surface evolution, of dealing automatically with changes in the surface topology during the evolution and of taking into account visibility and occlusion. The whole objects (at least the parts seen by at least two cameras) are reconstructed without any constraints on the cameras positions unlike in methods such as space carving [298] or graph cuts [290].

19

Visualization, Analysis and Shape Reconstruction of Sparse Data

Hongkai Zhao and Stanley Osher

Abstract

In this chapter mathematical models and efficient algorithms are developed for the visualization, analysis and shape reconstruction for an arbitrary data set that can include unorganized points or continuous manifolds of any codimension, such as pieces of curves and surface patches. The distance function to the data set and its contours are used for fast visualization and analysis of the data set. A minimal surface and a convection model are used for shape reconstruction from the data set. All formulations and numerical algorithms are based on implicit representations on simple rectangular grids which extend to any number of dimensions and which also can easily be combined with the level set method for dynamic shape deformation and other manipulations.

19.1 Introduction

Visualization and analysis of large data sets of unorganized points have important applications in scientific computing, computer graphics/vision, computer aided design, medical imaging etc. Mathematically, interpolation or shape reconstruction from a set of unorganized data points is an ill-posed problem, i.e., there is no unique solution. For large data sets in three and higher dimensions the problem becomes more challenging due to the following: (1) the geometry and topology of the real shape is not known *a priori* and can be very complicated, (2) finding the connection or ordering of the data points can be difficult and expensive, especially for data that are not uniformly spaced. In real applications, noise or other uncertainty in the data makes things even more complicated. A desirable procedure should be able to deal with all these difficulties and should have a representation and data structure that is not only good for static rendering but also good for dynamic deformation and other manipulations. Most of the previous approaches to

this problem can be classified as continuous vs. discrete in formulation and explicit vs. implicit in representation. For continuous formulations, one often uses variational methods or partial differential equations, e.g., [56, 91, 350, 463], to define the desired solution in an appropriate function space equipped with a certain regularity (smoothness). The formulation is meant to combine the interpolation or other constraints with a regularization to remove the ill-posedness. Approaches using continuous formulations are usually robust and allow flexibility in dealing with noise. However the choice of regularization in the formulation is not obvious and can make the problem difficult to solve numerically. For discrete approaches, one tries to sort out connections or orderings among data points based on exploring precise local structure and relation of points, lines and planes etc. Interpolation is then used to connect points or reconstruct the shape in a piecewise smooth fashion. Discrete approaches are usually based on simple geometric relations and attempt to interpolate the data exactly. However, in three and higher dimensions, sorting out correct connections for an arbitrary data set can be very difficult. Furthermore, the global structure pieced together from local information may not be consistent or may lack smoothness. Noise or uncertainty in the data can also be a problem.

The representation of reconstructed surfaces (shape) can be classified as explicit or implicit. Explicit surfaces prescribe the precise location of a surface while implicit surfaces represent a surface as a particular isocontour of a scalar function. Popular explicit representations include parametric surfaces using NURBS e.g., [427, 439], and triangulated surfaces using Voronoi diagrams and Delaunay triangulations e.g., [16, 17, 58, 174]. Tracking of large deformations and topological changes can be a problem using explicit surfaces.

Recently, implicit surfaces or volumetric representations have attracted a lot of attention. The traditional approach [55, 389, 536] uses a combination of smooth basis functions (primitives) to find a scalar function such that all data points are close to an isocontour of that scalar function. This isocontour represents the constructed implicit surface. The computational cost is very high for large data sets, since the construction is global. This requires solving a large linear system and a single data point change can result in changes of all the coefficients. Recently, in [80] polyharmonic Radial Basis Functions (RBF) and multipole methods were used, enabling the authors to model large data sets by a single RBF. However, human interaction and dynamic deformation are still difficult tasks. More recently, the signed distance function has been used to reconstruct and represent an implicit surface on a rectangular grid with the signs to distinguish inside and outside [26, 59, 242]. Most of the constructions of signed distance functions from unorganized points are based on discrete approaches. Similar ideas have been applied to shape reconstruction from range data and image fusion [144, 239]. The main advantages of implicit surfaces include topological flexibility, a simple data structure and depth/volumetric information. Using the signed distance representation, many surface operations such as Boolean operations, ray tracing and computing offsets become quite simple [419, 582]. Efficient algorithms, see e.g. [393, 583], are available to turn an implicit surface into a triangulated surface. In fact the level set

method [401] provides a general framework for the deformation of implicit surfaces according to arbitrary physical and/or geometric rules. Recent applications based on implicit surfaces and level set method range from computer animations, dynamic visibility, and solving partial differential equations on manifolds, see e.g., [153, 198, 38, 115, 530].

In this chapter, we present some recent work by the authors and collaborators [610, 608, 605] on visualization, analysis and shape reconstruction for unorganized data set based on continuous formulations and implicit representations. Our algorithms are developed based on rectangular grids and work in any number of dimensions.

19.2 Fast multiscale visualization and analysis of large data sets using distance functions

In many applications, for data sets that are noisy and of large size, perhaps involving multiple dimensions and with complicated geometry and topology, visualization or analysis techniques based on interpolation, which requires one to explore the exact relations among all data points, often result in high computational cost. In many situations we only need representations that are good enough for approximate analysis and visualization, i.e., one can compromise between the interpolation accuracy and efficiency. At the same time we also want to keep the consistency and fidelity of desired features and topological/geometric properties of the data set as much as possible. A consistent multiresolution and multiscale framework is another desired property for these approximate procedures.

In [606] efficient multiscale data analysis and processing procedures based on simple applications of distance function, distance contours and connectivity are proposed that can:

- find all disconnected components on a given scale,
- provide quick visualization on different resolution and scale,
- characterize and approximate some important topological or geometric properties of the data set,
- extend to any number of dimensions.

The basic idea is that the distance contours of the data set can be viewed as approximate offsets of the shape represented by the data set and can be used to visualize the data, to dissect disconnected components and to characterize some important topological and geometric information for the data set. Efficient numerical algorithms are developed based on rectangular grids to compute the distance function to an arbitrary data set, extract appropriate distance contours, and identify and characterize each disconnected component. The grid resolution and the choice of the distance contour depend on the prescribed scale or the data sampling

density. A natural multiresolution framework can be implemented in a hierarchical way. The complexity of the whole algorithm is of order $N + M$, where N is the number of grid points and M is the number of the data points.

19.2.1 The distance function and the fast sweeping method

The distance function $d(\mathbf{x}) = \text{dist}(\mathbf{x}, \mathcal{S})$ is the most important and intrinsic information for a data set \mathcal{S} . It is invariant under rotation and translation. Given the distance function to a data set, we can choose an appropriate scale parameter ϵ , which may be prescribed or may depend on the sampling density of the data, so that:

1. the distance contour $d(\mathbf{x}) = \epsilon$, with a good choice of ϵ , is an approximate ϵ -offset of the manifold represented by the data set and thus can be used to approximately represent and visualize the shape of the data set.
2. the ϵ distance contour can be used to dissect and analyze the disconnected components of the data set on the scale ϵ .
3. the ϵ -shell = $\{\mathbf{x} : d(\mathbf{x}) < \epsilon\}$ can be viewed as an approximate ϵ -covering of the manifold represented by the data set and thus can be used to approximate the Hausdorff dimension of each disconnected piece of the data set

In numerical computations we compute the distance function on a rectangular grid that contains the data set with a grid size h that resolves the prescribed scale ϵ . We then construct the ϵ distance contour or ϵ -shell, and dissect and characterize the disconnected components on this grid. To compute the distance function to an arbitrary data set on a rectangular grid, we use the fast sweeping algorithm that was used in [610] and analyzed in detail in [606]. The distance function $d(\mathbf{x})$ to an arbitrary data set \mathcal{S} solves the following Eikonal equation:

$$|\nabla u(\mathbf{x})| = 1, \quad u(\mathbf{x}) = 0, \quad \mathbf{x} \in \mathcal{S}. \quad (19.1)$$

From a PDE point of view, the information propagates along straight lines away from the data set, and the solution at a grid point should be determined only by its neighboring grid points that have smaller distance values. We use the following fast sweeping method that combines upwind differencing with Gauss-Seidel iterations of different sweeping orders to solve the equation (19.1) on rectangular grids. For simplicity of exposition, the algorithm is presented in two dimensions. Extensions to higher dimensions are straightforward. We use $\mathbf{x}_{i,j}$ to denote the grid points at which distance values are to be computed. h is the grid size and $u^h(\mathbf{x})$ denotes the numerical solution.

The fast sweeping algorithm:

1. Discretization:

At interior grid points the following upwind difference scheme is used to

discretize the PDE (19.1):

$$\begin{aligned} & [(u_{i,j}^h - u_{x\min}^h)^+]^2 + [(u_{i,j} - u_{y\min}^h)^+]^2 = h^2, \\ & i = 2, \dots, I-1, j = 2, \dots, J-1, \end{aligned} \quad (19.2)$$

where $u_{x\min}^h = \min(u_{i-1,j}^h, u_{i+1,j}^h)$, $u_{y\min}^h = \min(u_{i,j-1}^h, u_{i,j+1}^h)$ and $(x)^+ = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$. At the boundary, one sided difference is used if needed. For example, at the left boundary, one sided difference is used in the x direction,

$$[(u_{1,j}^h - u_{2,j}^h)^+]^2 + [(u_{1,j} - u_{y\min}^h)^+]^2 = h^2.$$

2. Initialization:

Assign exact values or interpolated values at grid points in or near \mathcal{S} . These values are fixed in later calculations. Assign large positive values at all other grid points. These values will be updated later.

3. Gauss-Seidel iterations with alternating sweeping orders:

At each grid $\mathbf{x}_{i,j}$ compute the solution, denoted by \bar{u} , of (19.2) from the current value of its neighbors $u_{i\pm 1,j}^h, u_{i,j\pm 1}^h$ and then update $u_{i,j}^h$ to be the smaller one of \bar{u} and its current value, i.e., $u_{i,j}^{new} = \min(u_{i,j}^{old}, \bar{u})$. We sweep the whole domain with four alternating orderings repeatedly,

- (1) $i = 1 : I, j = 1 : J$
- (2) $i = I : 1, j = 1 : J$
- (3) $i = I : 1, j = J : 1$
- (4) $i = 1 : I, j = J : 1$

Computing local distance: The computation of the distance function can be easily restricted to a neighborhood of the the data set using a simple cutoff criterion. For example, if we want to restrict the distance computation to the neighborhood $\{\mathbf{x}_{i,j} : d(\mathbf{x}_{i,j}) < \bar{d}\}$, in the Gauss-Seidel iteration we update the distance value at a grid point $\mathbf{x}_{i,j}$ only if at least one of its neighbors has a distance value smaller than \bar{d} , i.e., if $\min(u_{x\min}^h, u_{y\min}^h) < \bar{d}$.

Implementation details can be found in [606], in which it is shown that 2^n number of sweeps is enough to compute the distance function to an arbitrary data set in n dimensions. The alternating sweeping idea is similar to the Danielsson's algorithm [147]. However our formulation is based on solving the PDE (19.1) and can treat much more general situations [531].

19.2.2 Dissection and Characterization of Disconnected Components

After we have computed the distance function on a rectangular grid, we can use an appropriate distance contour and a connectivity condition to classify all grid points as exterior points, neighboring points and interior points associated with each disconnected component. The boundaries between these points are corresponding distance contours which can be used to visualize and characterize the

data set. For simplicity of exposition our notation and algorithms are defined for two dimensions. The extension to three and higher dimensions is straightforward.

Given a scale ϵ , which is resolved by the grid on which the distance function has been computed, we define the following three sets for all grid points $\mathbf{x}_{i,j}$.

- the set of exterior points, denoted by Ω^E :

$$\Omega^E = \{\mathbf{x}_{i,j} | d(\mathbf{x}_{i,j}) > \epsilon \text{ and } \mathbf{x}_{i,j} \text{ is connected to infinity}\}$$
- the set of neighboring points, denoted by Ω^N :

$$\Omega^N = \{\mathbf{x}_{i,j} | d(\mathbf{x}_{i,j}) < \epsilon\}$$
- the set of interior points, denoted by Ω^I , are the grid points complimentary to $\Omega^E \cup \Omega^N$.

We define the connected neighboring points of a grid point $\mathbf{x}_{i,j}$ to be $\mathbf{x}_{i\pm 1,j}, \mathbf{x}_{i,j\pm 1}$ in two dimensions and define the boundary of a set Ω , denoted by $\partial\Omega$, to be the set of those grid points in Ω for which at least one of its four neighbors is not in Ω .

We first identify the set of exterior grid points, Ω^E . We start with an arbitrary initial subset $\Omega_0^E \subset \Omega^E$. For example, Ω_0^E can be a known exterior region or simply a seed point in Ω^E such as a vertex point of the computational domain. We expand the initial set of exterior points Ω_0^E to Ω^E by repeatedly marching the boundary of the set of temporary exterior points, denoted by Ω_t^E , by adding those grid points that are connected to the boundary $\partial\Omega_t^E$ and have a distance value that is greater than ϵ . The expansion stops when there is no more connected exterior grid point to add. After the identification of all exterior points we also have the boundary between Ω^E and Ω^N , which can be used to visualize the data set as is shown below. For the unmarked grid points, we can easily identify those that have a distance value less than ϵ and mark them as neighboring points, and the remaining unmarked points are the interior points. With careful bookkeeping and marking, every grid point needs to be visited and checked only once.

The most subtle point in the classification of all grid points is the choice of the scale ϵ . If the sampling density of the data set satisfies a separation condition, i.e., the distance between two disconnected components or disjoint parts is larger than the largest distance between two connected neighboring data points, (which is a reasonable assumption to avoid any ambiguity about connectivity,) then we can use any ϵ in this range to separate all disconnected components. The set of neighboring points can be regarded as an ϵ covering of the real manifold represented by the data set. Moreover, if the real manifold represented by the data set is closed, the union of the neighboring set Ω^N and the interior set Ω^I has the same topology as the interior region enclosed by the real manifold. Hence the boundary between the set of exterior points and the set of neighboring points, which we have identified in the above classification algorithm, is homeomorphic to the real manifold and can be regarded as an approximate ϵ offset. If the real manifold represented by the data set is open and the distance between disjoint parts is larger than the largest distance between two connected neighboring points, the boundary between the set of exterior points and the set of neighboring points looks like a

thin shell enclosing the true manifold and can be still used as a good approximation in visualization, as is shown below. All the above situations are demonstrated in figure 19.1, in which the red curves correspond to an distance contour of the distance function to a data set represented by the blue dots. The data set contains several disconnected components. The sets of exterior points, neighboring points and interior points can be identified easily. We can also see clearly how the distance contour separates the disconnected components and how well it approximates the shape of the data set. By using this ϵ offset we can avoid finding complicated connections among data points.

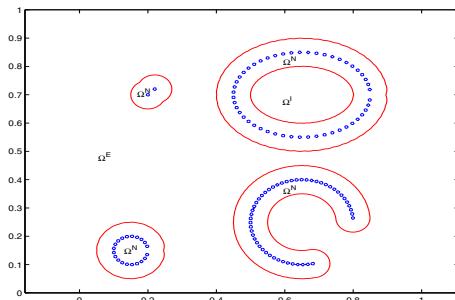


Figure 19.1. distance contours of a data set

Using the classification of the grid points we can dissect all disconnected components and characterize their properties on the scale ϵ . We start with any grid point in $\Omega^N \cup \Omega^I$ and find all grid points in $\Omega^N \cup \Omega^I$ that are connected to it and label them as the first component denoted by Ω_1 . If there is no grid point left in $\Omega^N \cup \Omega^I - \Omega_1$, then there is only one connected component, i.e., Ω_1 . Otherwise we start with any left point and find all grid points in $\Omega^N \cup \Omega^I - \Omega_1$ that are connected to it as the second connected component denoted by Ω_2 . We go on until all grid points in $\Omega^N \cup \Omega^I$ are marked. Now we have identified all disconnected components represented by the data set on the scale ϵ . By using the above algorithm for identifying all disconnected components, we can also find out the following useful geometric and topological properties of each component on the scale of ϵ :

- The total number of interior grid points in each connected component, i.e., $\Omega_k \cap \Omega^I$, which can be regarded as an approximation of the volume of the component Ω_k . In particular if a component has no interior point then we can say that data points in that component represent an open manifold on the scale of ϵ .
- The total number of neighboring grid points in each connected component, i.e., $\Omega_k \cap \Omega^N$, which can be regarded as an approximation of the Hausdorff dimension (i.e. surface area) of the manifold represented by the data in the k -th component. Also the ratio between the number of interior points and the number of neighboring points approximate the ratio between sur-

face area and the volume, which illustrates the “thickness” of a volumetric object.

- The total number of data points contained in each component (by counting and adding the number of data points in each grid cell in Ω_k) can be used to tell the significance of the cluster of data points. For example, we can use the number of data points in each component to single out and remove those isolated outliers.

If the sampling density varies for different components of the data set, we can first extract any particular component and apply a different scale ϵ on each component for the above analysis. In a more general setting, we can define the scale ϵ as a spatially varying function $\epsilon(\mathbf{x})$ that takes into account local sampling density, uncertainty or statistics of the data.

19.2.3 Extraction of distance contours and visualization of the data set

After the classification of all grid points, we can extract an appropriate distance contour as an offset to the real shape to visualize the data set. As is shown in figure 19.1, the distance contour $d(\mathbf{x}) = \epsilon$ which is exactly $\partial\Omega^N$, may be composed of two pieces for data points that form a closed manifold. The first one is the boundary between the set of exterior points, Ω^E , and the set of neighboring points, Ω^N , which we call the exterior distance contour and is denoted by Γ_e^ϵ . The second one is the boundary between the set of interior points, Ω^I , which is not empty if the data set consists of a closed manifold on the scale of ϵ , and the set of neighboring points, Ω^N , which we call the interior distance contour and is denoted by Γ_i^ϵ . No matter what ϵ is, the exterior distance contour Γ_e^ϵ always exists, assuming that the scale ϵ is resolved by the grid size. Thus we extract the exterior distance contour as an approximate offset of the shape representing the data set. Denote \underline{d} to be the largest distance between points that should be connected and \bar{d} to be the smallest distance between two disconnected components or disjoint parts, we say the sampling density of a data set satisfies the separation condition if $\underline{d} < \bar{d}$. The separation condition is to exclude possible connectivity ambiguity. If the sampling density of the data set satisfies the separation condition, we can choose ϵ as small as possible in the range $(\underline{d}/2, \bar{d}/2)$ so that the exterior distance contour is homeomorphic to the true shape. As the sampling density increases, i.e., $\underline{d} \rightarrow 0$, we can also take

$$\underline{d}/2 < \epsilon \rightarrow 0, \text{ so that } \|\Gamma - \Gamma_e^\epsilon\| = O(\epsilon) \rightarrow 0,$$

where Γ is the C^1 manifold represented by the data. Moreover, using the distance contour as an offset for the true shape can also automatically reduce the noise in the data set to some extent. In the examples shown below in section 19.2.4, we see that even for large real data sets that are noisy the visualization results look quite good.

To construct and visualize the distance contour, we construct an implicit representation, i.e., we construct a signed distance function to the exterior distance contour as follows. For those grid points in Ω^E (that are outside the exterior distance contour), their distance $\bar{d}(\mathbf{x}_{i,j})$ to the contour is just a shift by ϵ of its distance to the data set, i.e., $\bar{d}(\mathbf{x}_{i,j}) = d(\mathbf{x}_{i,j}) - \epsilon$. We also assign $\bar{d}(\mathbf{x}_{i,j}) = |d(\mathbf{x}_{i,j}) - \epsilon|$ to those boundary points in Ω^N that are immediate neighbors of Ω^E . Now we fix these correct distance values and use them as the initial values to solve the Eikonal equation (19.1) for $\bar{d}(\mathbf{x}_{i,j})$ by the fast sweeping method. Then we negate $\bar{d}(\mathbf{x}_{i,j})$ for those grid points in $\Omega^N \cup \Omega^I$ to get the signed distance function to the exterior distance contour. The whole procedure is again of $O(N)$ complexity for N grid points. Using the same procedure we can also find the signed distance to the interior distance contour. Since we can think of the distance contour we construct as an offset to the real shape represented by the data set, we can use the distance contour as an initial guess and move it closer to the data set to get a better approximation. We will discuss this in section 19.3.

Data coarsening Here we propose a simple data processing procedure that can be used to reduce the amount of data to a prescribed resolution. We lay down a grid according to a prescribed resolution. For each grid cell that contains data points, we compute the weighted mass center of all data points in that grid cell or data points in a neighborhood with a given radius and assign the total weight of those data points to the mass center. For example, the weight can be dependent on the uncertainty of each data point. We can replace the original set of data points by those weighted mass centers on this resolution for visualization or other analysis. Now each grid cell only has one point and is associated with some weight. Computing the mass center is a kind of averaging and can also help to remove noise or redundancy in the data to some extent. We demonstrate in one of our numerical examples that the data coarsening process can greatly reduce the total number of data points of the original data set and we see no difference in the visualization result.

19.2.4 Examples

In this section, we present results and timings of our algorithms on real data sets. In particular we would like to demonstrate (1) the efficiency and quality of using distance contours for visualization of large data sets, (2) dissection, analysis and process of large data sets without surface reconstruction. All our computations are done on a Linux PC with Pentium 600Mhz processor and 1GB memory, which allows a maximum number of grid points around 220^3 . The CPU time is measured in second. The timing includes every step except the rendering time using Data Explorer. Table 19.1 shows the number of data points, size of the grid and timing for our examples.

Figure 19.2 shows the data processing and visualization of a drill. Figure 19.2(a) shows the visualization of the raw data obtained by a 3D scanner from a few different angles. The data has outliers and disconnected components. The exterior distance contour $d(\mathbf{x}) = 2h$, where h is the grid size, is used for the visu-

Model	Data points	Grid size	CPU (second)
drill (raw)	50,643	100x69x54	3
drill base	34,106	149x118x151	14
drill cap	12,247	117x74x120	7
drill bit	4,283	24x120x24	0.4
dragon (raw)	1,769,513	149x124x147	49
dragon	1,723,751	311x222x146	93
dragon (scaled)	285,231	311x222x146	94
Buddha	543,652	156x371x156	39
terrain (raw)	100,860	600x118x99	51
terrain	98,725	601x452x29	28

Table 19.1. timing table

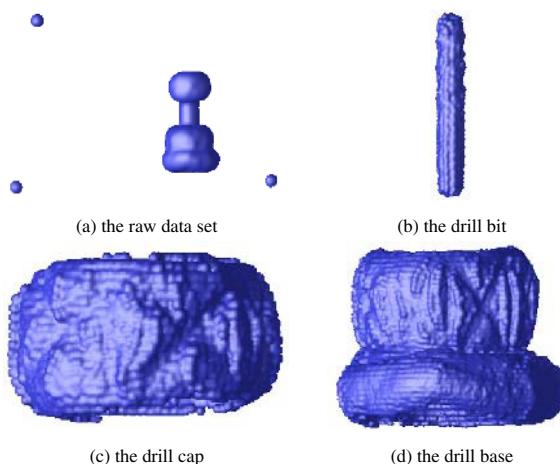


Figure 19.2. Data processing and visualization of a drill.

alization and data processing. The whole data set has a total of 50,643 points. The number of data points in each of the three sets of outliers is 2, 4, 1 respectively. There are three disconnected components after the removal of the outliers which are: the drill base which has 34,106 points, the drill cap which has 12,247 data points, and the drill bit which has 4,283 points. We can dissect these three parts and visualize them using distance contours on different grids accordingly in figure 19.2(b),(c),(d). Figure 19.3 shows the visualization and processing of the raw data from 3D scanning data for a dragon statue. We did not use those backdrop data for hole filling that is used for the construction at www-graphics.stanford.edu/data/3Dscanrep. The whole data set has 1,769,513 points. Figure 19.3(a) visualizes the raw data on a 149x124x147 grid using the distance contour $d(x) = h$. On this grid and with scale $\epsilon = 2h$ we can dissect 43 disconnected components, 42 of which correspond to outliers that have 45,518 points all together. After removal of the outliers we visualize the dragon on a

$311 \times 222 \times 146$ grid (the largest possible on our PC) using an exterior distance contour $d(\mathbf{x}) = h$ in figure 19.3(b). We visualize the same data set on a coarse grid of size $100 \times 73 \times 49$ in figure 19.3(c), which takes only 32 seconds. We can also rescale the data set after the removal of the outliers to the grid resolution of a $311 \times 222 \times 146$ grid, as described in section 19.2.3. The total number of data points is reduced to 285,231 which is visualized using $d(\mathbf{x}) = h$ in figure 19.3(d). Almost no difference can be seen from using the original data set in figure 19.3(b). In figure 19.4 we show the visualization of a Buddha statue on a $156 \times 371 \times 156$ grid from a 3D scanning data set of 543,652 points. The distance contour used is $d(\mathbf{x}) = h$ and the computation takes only 39 seconds. Figure 19.5 is the visualization of laser radar data for a terrain. The data set is very noisy and there are many bad data points due to occlusions and non-reflections. Moreover, the scale is very different in the horizontal and vertical directions. Figure 19.5(a) is the visualization of the raw data and shows how bad it is. After removal of the a total of 2,135 bad points, e.g. disconnected outliers, using our procedure, we visualize the data in figure 19.5(b). Now we can see quite clearly the buildings, the road, bushes and shadows.

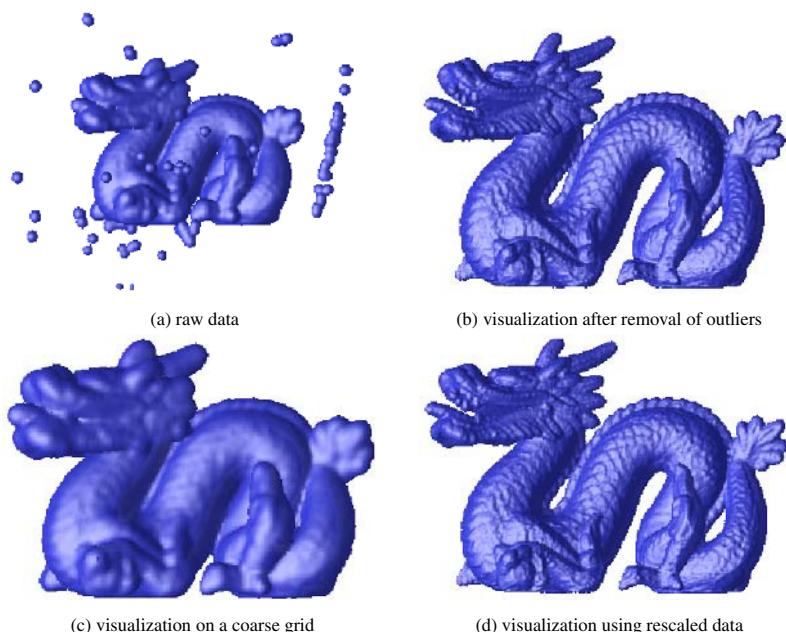


Figure 19.3. Visualization of a Buddha statue on a $156 \times 371 \times 156$ grid from a 3D scanning data set of 543,652 points.



Figure 19.4.

19.3 Construction of implicit surfaces using the level set method

In [610],[608] mathematical formulations and numerical algorithms were developed for surface reconstruction for unorganized data sets using differential geometry and partial differential equations. The level set method and dynamic implicit surfaces were used to provide a general framework for surface modeling, analysis, deformation and many other applications. A “weighted” minimal surface model, which takes into account both the surface area and closeness to the data set, was proposed. The variational formulation allowed us to balance them in an optimal way. The reconstructed surface is smoother than piecewise linear, thus the results look good on relatively coarse data sets. In addition, there is a regularization that is adapted to the local sampling density in the spirit of [15] and sharp features can be kept if a simple local sampling condition is satisfied. The formulation handles noisy as well as non-uniform data and works in an arbitrary number of dimensions. In [73] we have recently extended the method to also interpolate data giving the value of the unit normal to the surface at arbitrary points, curves and surface patches. A physically motivated convection model and a very fast tagging algorithm were also developed to give a very good initial approximation to the local minimizer, and thus to our minimal surface reconstruction.

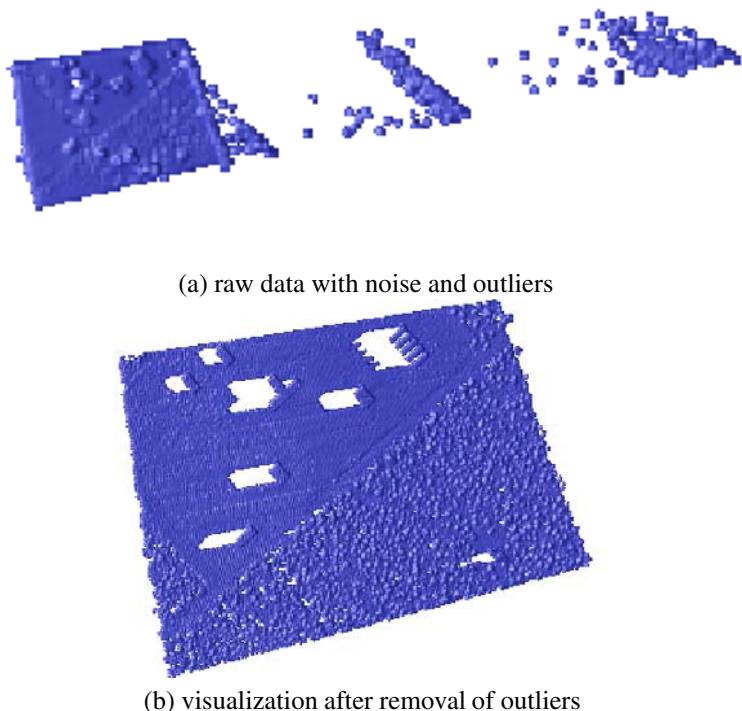


Figure 19.5.

19.3.1 The Weighted Minimal Surface Model

Let \mathcal{S} denote a general data set which can include data points, curves or pieces of surfaces. Define $d(\mathbf{x}) = \text{dist}(\mathbf{x}, \mathcal{S})$ to be the distance function to \mathcal{S} . In [610] the following surface energy is defined for the variational formulation:

$$E(\Gamma) = \left[\int_{\Gamma} d^p(\mathbf{x}) ds \right]^{\frac{1}{p}}, \quad 1 \leq p \leq \infty, \quad (19.3)$$

where Γ is an arbitrary surface and ds is the surface area. The energy functional is independent of parametrization and is invariant under rotation and translation. When $p = \infty$, $E(\Gamma)$ is the value of the distance of the point on Γ most remote from \mathcal{S} . For $p < \infty$, The surface energy $E(\Gamma)$ is equivalent to $\int_{\Gamma} d^p(\mathbf{x}) ds$, the surface area weighted by some power of the distance function. We take the local minimizer of our energy functional, which mimics a weighted minimal surface or an elastic membrane attached to the data set, to be the reconstructed surface.

As derived in [610] the gradient flow of the energy functional (19.3) is

$$\frac{d\Gamma}{dt} = - \left[\int_{\Gamma} d^p(\mathbf{x}) ds \right]^{\frac{1}{p}-1} d^{p-1}(\mathbf{x}) \left[\nabla d(\mathbf{x}) \cdot \mathbf{n} + \frac{1}{p} d(\mathbf{x}) \kappa \right] \mathbf{n}, \quad (19.4)$$

and the minimizer or steady state solution of the gradient flow satisfies the Euler-Lagrange equation

$$d^{p-1}(\mathbf{x}) \left[\nabla d(\mathbf{x}) \cdot \mathbf{n} + \frac{1}{p} d(\mathbf{x}) \kappa \right] = 0, \quad (19.5)$$

where \mathbf{n} is the unit outward normal and κ is the mean curvature. We see a balance between the attraction $\nabla d(\mathbf{x}) \cdot \mathbf{n}$ and the surface tension $d(\mathbf{x})\kappa$ in the equations above. Moreover the nonlinear regularization due to surface tension has a desirable scaling $d(\mathbf{x})$. Thus the reconstructed surface is more flexible in the region where sampling density is high and is more rigid in the region where the sampling density is low. In the steady state equation(19.5) above, since $|\nabla d \cdot \mathbf{n}| \leq 1$, we have a local sampling density condition similar to the one proposed in [16], which says sampling densities should resolve fine features locally. To construct the minimal surface we used a continuous deformation in [610]. We start with an initial surface that encloses all data and follow the gradient flow (19.4). The parameter p affects the flexibility of the membrane to some extent. When $p = 1$, the surface energy defined in (19.3) has the dimension of volume and the gradient flow (19.4) is scale invariant i.e., dimensionless. In practice we find that $p = 1$ or 2 (similar to a least squares formulation) are good choices. More details can be found in [610].

In two dimensions, it was shown in [610] that a polygon which connects adjacent points by straight lines is a local minimum. This result shows a connection between the variational formulation and previous approaches. On the other hand this result is not surprising since a minimal surface passing through two points is a straight line in two dimensions. However in three dimensions the situation becomes much more interesting. The reconstructed minimal surface has no edges and is smoother than a polyhedron.

19.3.2 The Convection Model

The evolution equation (19.4) involves the mean curvature of the surface and is a nonlinear parabolic equation. A time implicit scheme is not currently available. A stable time explicit scheme requires a restrictive time step size, $\Delta t = O(h^2)$, where h is the spatial grid size. Thus it is very desirable to have an efficient algorithm to find a good approximation before we start the gradient flow for the minimal surface. We propose the following physically motivated convection model for this purpose. We convect a flexible surface Γ in the potential field of the distance function $d(\mathbf{x})$ to the data set \mathcal{S} ,

$$\frac{d\Gamma(t)}{dt} = -\nabla d(\mathbf{x}). \quad (19.6)$$

The velocity field at any point, except those equal distance points, is a unit vector pointing toward its closest point in \mathcal{S} . The set of equal distance points has measure zero. Hence points on a curve or a surface, except those equal distance points, are attracted by their closest points in the data set (see Fig. 19.6(a)). The ambiguity at those equal distance points is resolved by adding a small surface tension force which automatically exists as numerical viscosity in our finite difference schemes. Those equal distance points on the curve or surface are dragged by their neighbors and the whole curve or surface is attracted to the data set until it reaches a local equilibrium, which is a polygon or polyhedron whose vertices belong to the data set as the viscosity tends to zero. Since the convection equation is a first order linear differential equation, we can solve it using a time step $\Delta t = O(h)$. The convection model very often results in a good surface reconstruction by itself.

19.3.3 The Level Set Formulation

In general we do not have any *a priori* knowledge about the topology of the shape to be reconstructed. Topological changes may occur during the continuous deformation process. This makes explicit tracking, which requires consistent parametrization, almost impossible to implement. Here we use the level set method as a powerful numerical technique for the dynamic deformation of implicit surfaces. The level set method is based on a continuous formulation using PDEs and allows one to deform an implicit surface according to various laws of motion depending on geometry, external forces, or a desired energy minimization. In numerical computations, instead of explicitly tracking a moving surface we implicitly capture it by solving a PDE for the level set function on rectangular grids. The data structure is extremely simple and topological changes are handled easily. The level set formulation works in any number of dimensions and the computation can easily be restricted to a narrow band near the zero level set, see e.g. [2, 421]. Two key steps for the level set method are: (1) Embed the surface: we represent a surface Γ as the zero isocontour of a scalar (level set) function $\phi(\mathbf{x})$, i.e. $\Gamma = \{\mathbf{x} : \phi(\mathbf{x}) = 0\}$. Geometric properties of the surface Γ can be easily computed using ϕ . (2) Embed the motion: we derive the time evolution PDE for the level set function such that the zero level set has the same motion law as the moving surface, i.e., $\Gamma(t) = \{\mathbf{x} : \phi(\mathbf{x}, t) = 0\}$,

For geometric motions, i.e. where the motion law (velocity) depends only on the geometry of the moving surface, the most natural way is to apply the same motion law for all level sets of the level set function, which will result in a morphological PDE [9]. For example, the gradient flow (19.4) is a geometric motion. If we use $p = 1$ and extend the geometric motion to all level sets, the gradient flow in level set formulation becomes

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \nabla \cdot \left[d \frac{\nabla \phi}{|\nabla \phi|} \right] = |\nabla \phi| \left[\nabla d \cdot \frac{\nabla \phi}{|\nabla \phi|} + d \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} \right], \quad (19.7)$$

For the convection model (19.6), since the velocity field $-\nabla d(\mathbf{x})$ is defined everywhere, we can naturally extend the convection to all level sets of $\phi(\mathbf{x}, t)$ to

obtain

$$\frac{\partial \phi}{\partial t} = \nabla d(\mathbf{x}) \cdot \nabla \phi. \quad (19.8)$$

Although all level set functions are equally good theoretically, in practice the signed distance function is preferred to avoid stiffness and inaccuracy in numerical computations. However even if we start with a signed distance function the level set function will generally not remain a signed distance function. We use a numerical procedure called reinitialization, see e.g. [421, 501], to redistance the level set function locally without interfering with the motion of the zero level set. As a result the implicit surface is a signed distance function after the deformation procedure stops.

19.3.4 Finding a good initial guess

We can use an arbitrary initial surface that contains the data set, such as a rectangular bounding box, to begin with. However, a good initial surface is important for the efficiency and convergence of our PDE based method. On a rectangular grid, we view an implicit surface as an interface that separates the exterior grid points from the interior grid points. An extremely efficient tagging algorithm was proposed in [608] that tries to identify as many correct exterior grid points as possible and hence provides a good initial implicit surface. As always, we start from any initial exterior region that is a subset of the true exterior region.

All grid points that are not in the initial exterior region are labeled as interior points. Those interior grid points that have at least one exterior neighbor are labeled as temporary boundary points. Now we use the following procedure to march the temporary boundary inward toward the data set. We put all the temporary boundary points in a heapsort binary tree structure sorting according to distance values. Take the temporary boundary point that has the largest distance (which is on the heap top) and check to see if it has an interior neighbor that has a larger or equal distance value. If it does not have such an interior neighbor, turn this temporary boundary point into an exterior point, take this point out of the heap, add all this point's interior neighbors into the heap and re-sort according to distance values. If it does have such an interior neighbor, we turn this temporary boundary point into a final boundary point, take it out of the heap and re-sort the heap. None of its neighbors are added to the heap. We repeat this procedure on the temporary boundary points until the the maximum distance of the temporary boundary points is smaller than some tolerance, e.g. the grid size, which means all the temporary boundary points in the heap are close enough to the data set. Finally, we turn these temporary boundary points into the final set of boundary points and our tagging procedure is finished. Now we have the final sets of interior, exterior and boundary points. Since each interior grid point is visited at most once, the procedure will be completed in no more than $O(N \log N)$ operations, where $\log N$ comes from the heap sort algorithm.

This general tagging algorithm can incorporate human interaction easily by putting any new exterior point(s) or region(s) into our tagged exterior region at any stage in our tagging algorithm. After the tagging algorithm is finished we again use the fast distance algorithm to compute a signed distance to the tagged final boundary. We can use either a bounding box of data set or an outer contour of the distance function, $d(\mathbf{x}) = \epsilon$, as the initial temporary boundary to start the fast tagging algorithm.

Remark: Since the maximum distance for the boundary heap is strictly decreasing, the algorithm converges and we can prove that those interior points which have a distance no smaller than the maximum distance of the temporary boundary heap at any time will remain as interior points, i.e. there is a non-empty interior region when the tagging algorithm is finished. We can also show that at least one of the final boundary points is within the tolerance distance to the data set.

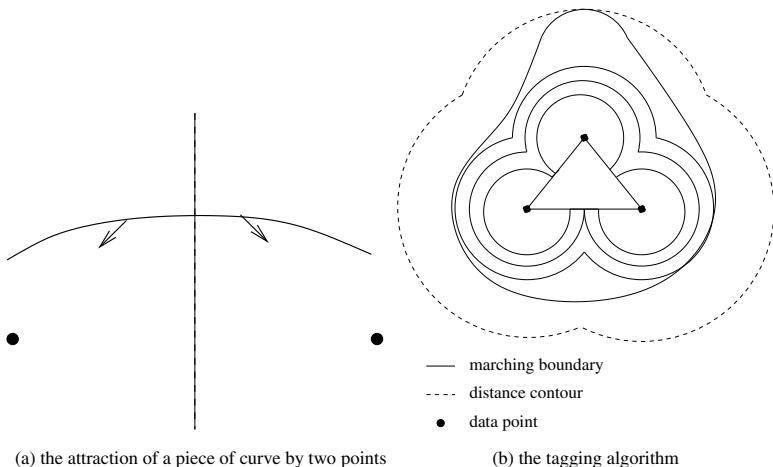


Figure 19.6.

Figure 19.6(b) illustrates how the fast tagging algorithm works. The furthest point on the initial temporary boundary is tangent to a distance contour and does not have an interior neighbor that is farther away. The furthest point is tagged as an exterior point and the boundary moves inward at that point. Another point on the temporary boundary becomes the furthest point and hence the whole temporary boundary moves inward. Gradually the temporary boundary follows distance contours and moves closer and closer to the data set until the distance contours begin to break at the equal distance point. We see that the temporary boundary at the breaking point of the distance contour has neighboring interior points that have a larger distance. So this temporary boundary point will be tagged as a final boundary point and the temporary boundary will stop moving inward at this breaking point. The temporary boundary starts deviating from the distance contours and continues moving closer to the data set until all temporary boundary points either have been tagged as final boundary points or are close to the data

points. The final boundary is approximately a polyhedron (polygon in 2D) with vertices belonging to the data set.

19.3.5 Multiresolution and Efficient Storage

There are two scales in our surface reconstruction. One is the resolution of the data set. The other is the resolution of the grid. The computational cost generally depends mainly on the grid size. To achieve the best results those two resolutions should be comparable. However our grid resolution can be independent of the sampling density. For example, we can use a low resolution grid when there is noise and redundancy in the data set or when memory and speed are important. From our numerical results, see e.g., figure 19.9(b) our reconstruction is quite smooth even on a very low resolution grid. We can also use a multiresolution algorithm, i.e., reconstruct the surface first on coarser grids and interpolate the result to a finer resolution grid for further refinement in an hierarchical way.

To store or render an implicit surface, we only need to record the values and locations (indices) of those grid points that are next to the surface, i.e., those grid points that have a different sign from at least one of their neighbors. These grid points form a thin grid shell surrounding the implicit surface. No connectivity or other information needs to be stored. We reduce the file size by at least an order of magnitude by using this method. Moreover we can easily reconstruct the signed distance function in $O(N)$ operations for the implicit surface using the following procedure. (1) Use the fast distance finding algorithm to find the distance function using the absolute value of the stored grid shell as an initial condition. (2) Use a tagging algorithm, similar to the one used above to find exterior points outside a distance contour, to identify all exterior points and interior points separated by the stored grid shell and turn the computed distance into the signed distance. For example, if we store the signed distance function for our reconstructed Happy Buddha on a $146 \times 350 \times 146$ grid in binary form, the file size is about 30MB. If we use the above efficient way of storage the file size is reduced to 2.5MB without using any compression procedure and we can reconstruct the signed distance function very quickly.

19.3.6 Numerical Implementations and Examples

There are three steps in our implicit surface reconstruction algorithm. First, we compute the distance function to an arbitrary data set on a rectangular grid. Second, we find a good initial surface. Third, we start the continuous deformation following either the gradient flow (19.4) or the convection (19.6) using the corresponding level set formulation (19.7) or (19.8). Our numerical implementations are based on standard algorithms for the level set method. Details can be found in, for example, [421, 607, 610]. The convection model is simple but the reconstructed surface is close to a piecewise linear approximation. In contrast the gradient flow is more computationally expensive but reconstructs a smoother weighted minimal surface. In particular, the gradient flow can be used

as a smoothing process for implicit surfaces. In most of our applications, less than one hundred time steps in total are enough for our continuous deformation to converge.

Figure 19.7 shows data points for a torus, a few curves (longitudes and latitudes) on a sphere, data points from MRI slices for a rat brain. Figure 19.8 shows the final surface reconstruction from the above data. We see that the hole in the torus is filled nicely with a minimal surface. For the sphere reconstruction we only provide the unsigned distance function to the curves which can be viewed as an extreme case of non-uniform data. After the initial reconstruction using a distance contour and/or fast tagging algorithm, we first use the convection model and then use the gradient flow to finish the final reconstruction. In our reconstruction, the grid resolution is much lower than the data samples and yet we get final results that are comparable to other reconstructions. Figure 19.9 shows the reconstruction of the Happy Buddha. Figure 19.9(a) is the reconstruction on a fine grid. Figure 19.9(b) shows the reconstruction on a coarse grid.

Model	Data points	Grid size	CPU (minute)
Rat brain	1506	80x77x79	3
Buddha	543652	146x350x146	68
Buddha	543652	63x150x64	7

Table 19.2. timing table

Acknowledgments. Data sets for the drill, the dragon and Buddha are from The Stanford 3D Scanning Repository. The laser radar data is from Naval Air Warfare Center at China Lake.

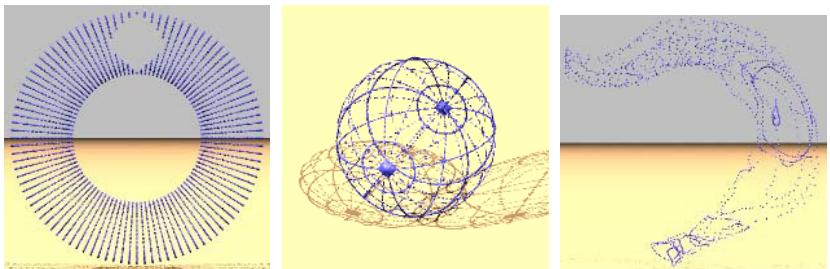


Figure 19.7. initial data

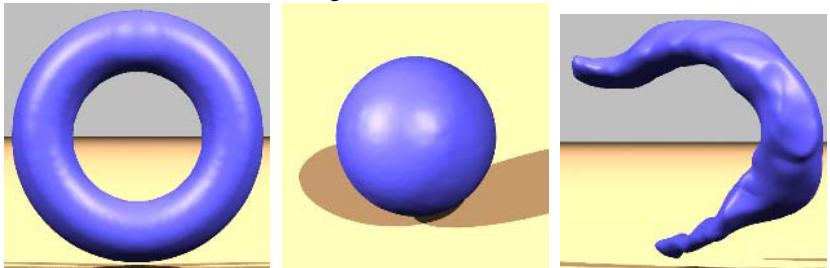


Figure 19.8. final reconstruction



(a) reconstruction on a fine grid

(b) reconstruction on a coarse grid

Figure 19.9. reconstruction of the Happy Buddha

20

Variational Problems and Partial Differential Equations on Implicit Surfaces: Bye Bye Triangulated Surfaces?

**Marcelo Bertalmío, Facundo Mémoli, Li-Tien Cheng,
Guillermo Sapiro and Stanley Osher**

Abstract

A novel framework for solving variational problems and partial differential equations for scalar and vector-valued data defined on surfaces is described in this chapter. The key idea is to implicitly represent the surface as the level set of a higher dimensional function, and solve the surface equations in a fixed Cartesian coordinate system using this new embedding function. The equations are then both intrinsic to the surface and defined in the embedding space. This approach thereby eliminates the need for performing complicated and not-accurate computations on triangulated surfaces, as it is commonly done in the literature. We describe the framework and present examples in computer graphics and image processing applications, including texture synthesis, flow field visualization, as well as image and vector field intrinsic regularization for data defined on 3D surfaces.

20.1 Introduction

In a number of applications, variational problems and partial differential equations need to be intrinsically solved for data defined on arbitrary manifolds, three dimensional surfaces in particular. Examples of this exist in the areas of mathematical physics, fluid dynamics, image processing, medical imaging, computer graphics, and pattern formation. In computer graphics, examples of this include texture synthesis [535, 580], vector field visualization [163], and weathering [167]. In other numerous applications, data defined on surfaces often needs to

be regularized, e.g., as part of a vector field computation or interpolation process [428, 577], for inverse problems [187], or for surface parameterization [172]. These last regularization examples can be addressed solving a variational problem defined on the surface, or its corresponding gradient-descent flow on the surface, using for example the well developed theory of harmonic maps [175, 176], which has recently been demonstrated to be of use for image processing and computer graphics applications as well, e.g., [98, 172, 422, 490, 505, 604]. All these equations are generally solved on triangulated or polygonal surfaces. That is, the surface is given in polygonal (triangulated) form, and the data is discretely defined on it. Solving the problems then in this representation involves the non-trivial discretization of the equations in general polygonal grids, as well as the difficult numerical computation of other quantities like projections onto the discretized surface (when computing gradients and Laplacians for example). Although the use of triangulated surfaces is extremely popular in all areas dealing with 3D models, mainly in computer graphics, there is still *not* a widely accepted technique to compute simple differential characteristics such as tangents, normals, principal directions, and curvatures; see for example [156, 383, 508] for a few of the approaches in this direction. On the other hand, it is widely accepted that computing these objects for iso-surfaces (implicit representations) is straightforward and much more accurate and robust. This problem in triangulated surfaces becomes even bigger when we not only have to compute these first and second order differential characteristics of the surface, but also have to use them to solve variational problems and PDE's for data defined on the surface. Moreover, virtually no analysis exists on numerical PDE's on non-uniform grids in the generality needed for the wide range of applications mentioned above, making it difficult to understand the behavior of the numerical implementation and its proximity (or lack thereof) to the continuous model. In this chapter we present the framework introduced in [38] to solve variational problems and PDE's for scalar and vector-valued data defined on surfaces. We use, instead of a triangulated/polygonal representation, an implicit representation: our surface will be the zero-level set of a higher dimensional *embedding* function (i.e., a 3D volume with real values, positive outside the surface and negative inside it). Implicit surfaces have been widely used in many areas including computational physics, e.g., [401], computer graphics, e.g., [55, 201, 579], and image processing [458], as an alternative efficient representation to triangulated surfaces. We smoothly extend the original (scalar or vector-valued) data lying on the surface to the 3D volume, adapt our PDE's accordingly, and then perform all the computations on the fixed Cartesian grid corresponding to the embedding function. These computations are nevertheless intrinsic to the surface. The advantages of using the Cartesian grid instead of a triangulated mesh are many: we can use well studied numerical techniques, with accurate error, stability, and robustness measures; the topology of the underlying surface is not an issue; and we can derive simple, accurate, robust and elegant implementations. If the original surface is not already in implicit form, and it is for example triangulated, we can use any of a number of implicitation algorithms that achieve this representation given a triangulated input, e.g.,

[173, 295, 507, 596]. For example, the public domain software [349] can be used. If the data is just defined on the surface, an extension of it to the whole volume is also easily achieved using a PDE, as we will later see. Therefore, the method here proposed works as well for non-implicit surfaces after the preprocessing is performed. This preprocessing is quite simple and no complicated regridding needs to be done to go from one surface representation to another (see below). Finally, we will solve the variational problem or PDE only in a band surrounding the zero level set (a classical approach; see [421]). Therefore, although we will be increasing by one the dimension of the space, the computations remain of the same complexity, while the accuracy and simplicity are significantly improved. A final note before we proceed. As we have mentioned, the data defined on the surfaces will only be scalar or vector valued, i.e., if we see our variational problems and PDE's as involving maps form a domain manifold into a target manifold, then our domain manifolds in this chapter will be arbitrary implicit surfaces while our target manifolds will only be the Euclidean space \mathbb{R}^n or the unit ball. But the same formulation that is presented here to deal with the domain manifold as an implicit surface can be adapted to treat the target manifold as an implicit surface as well, always working on Cartesian grids, as shown in [358]. So in general there is no limitation as to the nature of the target manifold, and we can work not just with scalar and vector-valued data but with any sort of data defining a target manifold.

20.1.1 The background and our contribution

Representing *deforming* surfaces as level sets of higher dimensional functions was introduced in [401] as a very efficient technique for numerically studying the deformation (see [397] for a review of this technique and also [579] for studies on the deformation and manipulation of implicit surfaces for graphics applications). The idea is to represent the surface deformation via the embedding function deformation, which adds accuracy, robustness, and, as expected, topological liberty. When the velocity of the deformation is given by the minimization of an energy, the authors in [607] proposed a “variational level set” method, where they extended the energy (originally defined only on the surface) to the whole space. This allows for the implementation to be in the Cartesian grid. The key of this approach is to go from a “surface energy” to a “volume energy” by using a Dirac’s delta function that concentrates the penalization on the given surface.

We will follow this general direction with our fixed, *non deforming* surfaces. In our case, what is being “deformed” is the (scalar or vector-valued) data on the surface. If this deformation is given by an energy-minimization problem (as is the case in data smoothing applications), we will extend the definition of the energy to the whole 3D space, and its minimization will be achieved with a PDE, which, despite its being intrinsic to the underlying surface, is also defined in the whole space. Therefore, it is easily implementable. This is straightforward, as opposed to approaches where one maps the surface data onto the plane, performs the required operations there and then maps the results back onto the triangulated

representation of the surface; or approaches that attempt to solve the problem directly on a polygonal surface.

Very interestingly, the new framework proposed here also tells us how to translate into surface terms PDE's that we know that work on the plane but which do not necessarily minimize an energy (e.g., texture synthesis or flow visualization PDE's). Instead of running these PDE's on the plane and then mapping the results onto a triangulated representation of the surface, or running them directly on the triangulated domain, we obtain a 3D straightforward Cartesian grid realization that implements the equation intrinsically on the surface and whose accuracy depends only on the degree of spatial resolution.

Moreover, we consider that for computing differential characteristics and solving PDE's even for triangulated surfaces, it might be appropriate to run an implicitation algorithm as any of the ones used for the examples in this chapter and then work on the implicit representation. Current algorithms for doing this, some of them publicly available [349], are extremely accurate and efficient.

The contribution of this work is then a new technique to efficiently solve a common problem in many computational physics, computer graphics and engineering applications: the implementation of variational problems and PDE's on 3D surfaces. In particular, we show how to transform any intrinsic variational or PDE equation into a corresponding problem for implicit surfaces. In this chapter we are then proposing a new framework to better solve existent problems and to help in building up the solutions for new ones. To exemplify the technique and its generality, we implement and extend popular equations previously reported in the literature. Here we solve them with our framework, while these problems were solved in the literature with elaborate discretizations on triangulated representations.

Before proceeding, we should comment on a few of the basic characteristics of our framework. First, as stated before, although we solve the equations in the embedding space, the basic computational complexity of our technique is not increased, since all operations are performed on a narrow band surrounding the given surface. Secondly, since the work is now on a Cartesian grid, all classical numerical analysis results on issues like robustness and stability, apply here as well. Note that for triangulated representations, new theoretical results are needed to justify the common methods proposed in the literature, while with our framework, classical and well established numerical techniques can be used, as accurate, robust, and computationally efficient as dictated by the application.

20.2 The framework

20.2.1 Surface and data representation

As mentioned before, our approach requires us to have an implicit representation of the given fixed surface, and the data must be defined in a band surrounding it and not just on the surface. The implicit surfaces used in this chapter have been de-

rived from public-domain triangulated surfaces via the computation of a (signed) distance function $\psi(x, y, z)$ to the surface \mathcal{S} . Arriving at an implicit representation from a triangulated one is currently not a significant issue, there are publicly available algorithms that achieve it in a very efficient fashion. To exemplify this, in our chapter we have used several of these techniques. For some surfaces the classical Hamilton-Jacobi equation $\|\nabla\psi\|=1$ was solved on a pre-defined grid enclosing the given surface via the computationally optimal approach devised in [533]. Accurate implicit surfaces from triangulations of the order of one million triangles are obtained in less than two minutes of CPU-time with this technique. Alternatively we used the implementation of the Closest Point Transform available in [349]. The teapot and knot surfaces were obtained from unorganized data points using the technique devised in [610]. We therefore assume from now on that the three dimensional surface \mathcal{S} of interest is given in implicit form, as the zero level set of a given function $\psi : \mathbb{R}^3 \rightarrow \mathbb{R}$. This function is negative inside the closed bounded region defined by \mathcal{S} , positive outside, Lipschitz continuous a.e., with $\mathcal{S} \equiv \{x \in \mathbb{R}^3 : \psi(x) = 0\}$. To ensure that the data, which needs not to be defined outside of the surface originally, is now defined in the whole band, one simple possibility is to extend this data u defined on \mathcal{S} (i.e the zero level set of ψ) in such a form that it is constant normal to each level set of ψ . This means the extension satisfies $\nabla u \cdot \nabla \psi = 0$. (For simplicity, we assume now u to be a scalar function, although we will also address in this chapter problems where the data defined on \mathcal{S} is vector-valued. This is solved in an analogous fashion.) To solve this we numerically search for the steady state solution of the Cartesian PDE

$$\frac{\partial u}{\partial t} + \text{sign}(\psi)(\nabla u \cdot \nabla \psi) = 0.$$

This technique was first proposed and used in [108]. Note that this keeps the given data u on the zero level set of ψ (the given surface) unchanged.

Both the implicitation and data extension (if required at all by the given data), need to be done only once off line. Moreover, they will remain for all applications that need this type of data.

20.2.2 A simple example: Heat flow on implicit surfaces

We will exemplify our framework with the simplest case, the heat flow or Laplace equation for scalar data defined on a surface. For scalar data u defined on the plane, that is, $u(x, y) : \mathbb{R}^2 \rightarrow \mathbb{R}$ it is well known that the heat flow

$$\frac{\partial u}{\partial t} = \Delta u \tag{20.1}$$

where $\Delta := \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$ is the Laplacian, is the gradient descent flow of the Dirichlet integral

$$\frac{1}{2} \int_{\mathbb{R}^2} \|\nabla u\|^2 dx dy, \tag{20.2}$$

where ∇ is the gradient.

Eq. (20.1) performs smoothing of the scalar data u , and this smoothing process progressively decreases the energy defined in eq. (20.2). If we now want to smooth scalar data u defined on a surface \mathcal{S} , we must find the minimizer of the *harmonic* energy given by

$$\frac{1}{2} \int_{\mathcal{S}} \| \nabla_{\mathcal{S}} u \|^2 d\mathcal{S}, \quad (20.3)$$

The equation that minimizes this energy is its gradient descent flow:

$$\frac{\partial u}{\partial t} = \Delta_{\mathcal{S}} u. \quad (20.4)$$

Here $\nabla_{\mathcal{S}}$ is the intrinsic gradient and $\Delta_{\mathcal{S}}$ the intrinsic Laplacian or Laplace-Beltrami operator. These are classical concepts in differential geometry, and basically mean the natural extensions of the gradient and Laplacian respectively, considering all derivatives intrinsic to the surface. For instance, the intrinsic gradient is just the projection onto \mathcal{S} of the regular 3D gradient while the Laplace-Beltrami operator is the projected divergence of it [489].

Classically, eq. (20.4) would be implemented in a triangulated surface, giving place to sophisticated and elaborate algorithms even for such simple flows. We now show how to simplify this when considering implicit representations.

Recall that \mathcal{S} is given as the zero level set of a function $\psi : \mathbb{R}^3 \rightarrow \mathbb{R}$, ψ is negative inside the region bounded by \mathcal{S} , positive outside with $\mathcal{S} \equiv \{x \in \mathbb{R}^3 : \psi(x) = 0\}$. We proceed now to redefine the above energy and compute its corresponding gradient descent flow. Let \mathbf{v} be a generic three dimensional vector, and $P_{\mathbf{v}}$ the operator that projects a given three dimensional vector onto the plane orthogonal to \mathbf{v} :

$$P_{\mathbf{v}} := I - \frac{\mathbf{v} \otimes \mathbf{v}}{\| \mathbf{v} \|^2} \quad (20.5)$$

It is then easy to show that the harmonic energy (20.3) is equivalent to (see for example [489])

$$\frac{1}{2} \int_{\mathcal{S}} \| P_{\mathbf{N}} \nabla u \|^2 d\mathcal{S}, \quad (20.6)$$

where \mathbf{N} is the normal to the surface \mathcal{S} . In other words, $\nabla_{\mathcal{S}} u = P_{\mathbf{N}} \nabla u$. That is, the gradient intrinsic to the surface ($\nabla_{\mathcal{S}}$) is just the projection onto the surface of the 3D Cartesian (classical) gradient ∇ .¹ We now embed this in the function ψ :

$$\begin{aligned} \frac{1}{2} \int_{\mathcal{S}} \| \nabla_{\mathcal{S}} u \|^2 d\mathcal{S} &= \frac{1}{2} \int_{\mathcal{S}} \| P_{\mathbf{N}} \nabla u \|^2 d\mathcal{S} \\ &= \frac{1}{2} \int_{\Omega \in \mathbb{R}^3} \| P_{\nabla \psi} \nabla u \|^2 \delta(\psi) \| \nabla \psi \| dx, \end{aligned}$$

¹Note that using this fact, we have transformed the computation of the norm of the intrinsic 2D gradient into an equivalent 3D Euclidean computation, see below.

where $\delta(\cdot)$ stands for the Dirac delta function, and all the expressions above are considered in the sense of distributions. Note that first we got rid of intrinsic derivatives by replacing ∇_S by $P_N \nabla u$ (or $P_{\nabla \psi} \nabla u$) and then replaced the intrinsic integration ($\int_S dS$) by the explicit one ($\int_{\Omega \in \mathbb{R}^3} dx$) using the delta function. Intuitively, although the energy lives in the full space, the delta function forces the penalty to be effective only on the level set of interest. The last equality includes the embedding, and it is based on the following simple facts:

1. $\nabla \psi \parallel \mathbf{N}$.
2. $\int_{\Omega} \delta(\psi) \parallel \nabla \psi \parallel dx = \int_S dS = \text{surface area}$.

In [38] it is shown that the gradient descent of this energy is given by

$$\frac{\partial u}{\partial t} = \frac{1}{\parallel \nabla \psi \parallel} \nabla \cdot (P_{\nabla \psi} \nabla u \parallel \nabla \psi \parallel). \quad (20.7)$$

In other words, this equation corresponds to the intrinsic heat flow or Laplace-Beltrami for data on an implicit surface. But all the gradients in this PDE are defined in the three dimensional Cartesian space, not in the surface S (this is why we need the data to be defined at least on a band around the surface). The numerical implementation is then straightforward. This is the beauty of the approach! Basically, for this equation we use a classical scheme of forward differences in time and a succession of forward and backward differences in space (see [38] for details). The other equations in this chapter are similarly implemented. This follows techniques as those in [449]. Once again, due to the implicit representation and embedding in a Cartesian grid, classic numerical techniques are used, avoiding elaborate projections onto discrete surfaces and discretization on general meshes, e.g., [156, 247]. Classical numerical approaches and theoretical findings on robustness, accuracy, and error bounds, apply then for our framework.

It is easy to show a number of important properties of this equation:

1. For any second embedding function $\phi = \phi(\psi)$, with $\phi' \neq 0$ and $\phi(0) = 0$, we obtain the same gradient descent flow. Since both ψ and ϕ have to share the zero level set, and we are only interested in the flow around this zero level set, this means that the flow is (locally) independent of the embedding function.
2. If ψ is the signed distance function, a very popular implicit representation of surfaces (obtained for example from the implicitation algorithms previously mentioned), the gradient descent simplifies to

$$\frac{\partial u}{\partial t} = \nabla \cdot (P_{\nabla \psi} \nabla u). \quad (20.8)$$

We have then obtained the basic approach for embedding intrinsic variational problems. We proceed now to embed general PDE's.

20.2.3 From variational problems to PDE's

We note that we could also have derived eq. (20.7), directly from the harmonic maps flow

$$\frac{\partial u}{\partial t} = \Delta_{\mathcal{S}} u,$$

via the simple geometry exercise of computing the Laplace-Beltrami $\Delta_{\mathcal{S}} u$ for \mathcal{S} in implicit form (this is simply done by means of the projected derivatives as explained above, e.g., [489]). That is, the same equation is obtained when embedding the energy and then computing the gradient descent and when first looking at the gradient descent followed by the embedding of all of its components. This property is of particular significance. It basically shows how to solve general PDE's, not necessarily gradient-descent flows, for data defined on implicit surfaces. All that we need to do is to recompute the components of the PDE for implicit representations of the surface. Note that in this way, conceptually, we can redefine classical planar PDE's on implicit surfaces, making them both intrinsic to the underlying surface and defined on the whole space.

20.2.4 Anisotropic diffusion on implicit surfaces

From this very simple example on the Laplace-Beltrami flow we have seen the key point of our approach. If the process that we want to implement comes from the minimization of an energy, we derive a PDE for the whole space by computing the gradient-descent of the whole-space-extension of that energy. Otherwise, given a planar PDE we recompute its components for an implicit representation of the surface. For instance, anisotropic diffusion can be performed on the plane by

$$\frac{\partial u}{\partial t} = \nabla \cdot \left(\frac{\nabla u}{\| \nabla u \|} \right), \quad (20.9)$$

which minimizes the TV energy $\int_{\mathbb{R}^2} \| \nabla u \| dx dy$ (see [449] and also [10, 49, 423] for related formulations).

If we now want to perform intrinsic anisotropic diffusion of scalar data on a surface \mathcal{S} , we can either recompute the gradient-descent flow for the intrinsic TV energy $\int_{\mathcal{S}} \| \nabla_{\mathcal{S}} u \| d\mathcal{S}$, which for \mathcal{S} in implicit form becomes $\int_{\Omega \in \mathbb{R}^3} \| P_{\nabla \psi} \nabla u \| \delta(\psi) \| \nabla \psi \| dx$, or just substitute in eq. (20.9) the corresponding expressions as explained in the previous section. Either way we obtain the same result, the following PDE, which is valid in the embedding Euclidean space:

$$\frac{\partial u}{\partial t} = \frac{1}{\| \nabla \psi \|} \nabla \cdot \left(\frac{P_{\nabla \psi} \nabla u}{\| P_{\nabla \psi} \nabla u \|} \| \nabla \psi \| \right). \quad (20.10)$$

Note that general p -harmonic maps, that is, maps for L_p , $p \neq 2$, norms of the intrinsic surface gradient have been studied in the literature as well, e.g., [98, 110, 210, 229, 505]. In the following section additional equations will be presented.



Figure 20.1. Intrinsic isotropic diffusion. Left: original image. Middle: after 15 diffusion steps. Right: after 50 diffusion steps.

20.3 Experimental examples

We now exemplify the framework just described for a number of important cases. The numerical implementation used is quite simple, and requires a few lines of *C++* code. The CPU time required for the diffusion examples is of a few seconds on a PC (512Mb RAM, 1GHz) under Linux. For the texture synthesis examples, the CPU time ranges from a few minutes to one hour, depending on the pattern and parameters chosen. All the volumes used contain roughly 128^3 voxels. Note once again that due to the use of only a narrow band surrounding the zero level set, the order of the algorithmic complexity remains the same. On the other hand, the use of straightforward Cartesian numerics reduces the overall algorithmic complexity, improving accuracy and simplifying the implementation.

20.3.1 Diffusion of scalar images on surfaces

The use of PDE's for image enhancement has become one of the most active research areas in image processing [458]. In particular, diffusion equations are commonly used for image regularization, denoising, and multiscale representations (representing the image simultaneously at several scales or levels of resolution). This started with the works in [285, 578], where the authors suggested the use of the linear heat flow (20.1) for this task, where u represents the image gray values (the original image is used as initial condition). Note of course that this is the basic regularization needed for inverse problems defined on surfaces, e.g., [187]. By deriving the heat flow or Laplace-Beltrami equation on implicit surfaces we then derive the basic PDE used for image processing as well as the fundamental data regularization energy/flow. As we have seen, this flow is the gradient-descent of (20.2), and the generalizations of these equations for data on the surface are given by (20.4) and (20.3) respectively. In implicit form, the heat flow on surfaces is given by (20.7). Figure 20.1 shows a simple example of image diffusion on a surface. Please note that this is *not* equivalent to performing 3D smoothing of the data and then looking to see what happened on \mathcal{S} . Our flow,



Figure 20.2. Intrinsic anisotropic diffusion with constraints (automatic stop term). Left: original noisy image. Middle: after 50 diffusion steps. Right: after 90 diffusion steps. Notice how the diffusion stops and information is not smeared.

though using extended 3D data, performs smoothing *directly* on the surface, it is an intrinsic heat flow (Laplace-Beltrami on the surface and *not* Laplace on the 3D space). The complete details of the numerical implementation of this flow are given in [38] (they once again show how the implementation is significantly simplified with the framework here described). In Figure 20.2 we show an example for the anisotropic flow (20.10). In this case, we have a noisy image with known variance. We can then easily add this constraint to the flow and the corresponding variational formulation. The energy corresponding to this constraint is given by ($\lambda \in \mathbb{R}$ is a parameter and u_0 is the given noisy image)

$$\frac{\lambda}{2} \int_S (u - u_0)^2 dS,$$

which after it is made intrinsic and implicit becomes

$$\frac{\lambda}{2} \int_{\mathbb{R}^3} (u - u_0)^2 \delta(\psi) \| \nabla \psi \| dx.$$

In order to incorporate the constraint on the noise variance into the diffusion/denoising process, we add to the flow (20.10) the corresponding Euler-Lagrange of this energy, given by

$$\lambda(u - u_0).$$

Note in the figure how the noise is removed while the image details are preserved, as expected from an anisotropic flow. The parameter λ is estimated in a way suggested in [449], see [38]. The same approach, that of anisotropic diffusion with a stopping term given by the constraint, may be used to perform intrinsic *deblurring*, see [114].

We should note before proceeding that [278] also showed how to regularize images defined on a surface. The author's approach is limited to graphs (not generic surfaces) and only applies to level set based motions. The approach is simply to project the deformation of the data on the surface onto a deformation on the plane.

20.3.2 Diffusion of directional data on surfaces

A particularly interesting example is obtained when we have unit vectors defined on the surface. That is, we have data of the form $u : \mathcal{S} \rightarrow S^{n-1}$. When $n = 3$ our unit vectors lie on the sphere. Examples of this data include principal directions (or general directional fields on 3D surfaces) and chromaticity vectors (normalized RGB vectors) for color images defined on the surface. This is also one of the most studied cases of the theory of harmonic maps due to its physical relationship with liquid crystals, and it was introduced in [505] for the regularization of directional data, unit vectors, on the plane (see also [98, 422, 490]). This framework of harmonic maps was used in computer graphics for texture mapping and surface parameterization, as pointed out earlier.

We still want to minimize an energy of the form

$$\int_{\mathcal{S}} \| \nabla_{\mathcal{S}} u \|_{}^p d\mathcal{S},$$

though in this case $\nabla_{\mathcal{S}}$ is the vectorial gradient and the minimizer is restricted to be a unit vector. It is easy to show, e.g., [68, 499], that the gradient descent of this energy is given by the coupled system of PDE's

$$\frac{\partial u_i}{\partial t} = \operatorname{div}_{\mathcal{S}} (\| \nabla_{\mathcal{S}} u \|^{p-2} \nabla_{\mathcal{S}} u_i) + u_i \| \nabla_{\mathcal{S}} u \|_{}^p, \quad 1 \leq i \leq n.$$

This flow guarantees that the initial unit vector $u(x, y, z, 0)$ remains a unit vector $u(x, y, z, t)$ all the time, thereby providing an equation for isotropic ($p = 2$) and anisotropic ($p = 1$) diffusion and regularization of unit vectors on a surface.

We can now proceed as before, and embed the surface \mathcal{S} into the zero level-set of ψ , obtaining the following gradient descent flows (see [38] for the derivation):

$$\frac{\partial u_i}{\partial t} = \frac{1}{\| \nabla \psi \|} \nabla \cdot \left(\frac{P_{\nabla \psi} \nabla u_i}{\| P_{\nabla \psi} \nabla u \|^{2-p}} \| \nabla \psi \| \right) + u_i \| P_{\nabla \psi} \nabla u \|_{}^p. \quad (20.11)$$

Note once again that although the regularization is done intrinsically on the surface, this equation only contains Cartesian gradients. An example of this flow for anisotropic diffusion of principal direction vectors is given in Figure 20.3. On the left, we see the surface of a bunny with its correspondent vector field for the major principal direction. Any irregularity on the surface produces a noticeable alteration of this field, as can be seen in the details a and b . In the details a' and b' , we see the result of applying the flow (20.11). Once again, the implementation of this flow with our framework is straightforward, while it would require very sophisticated techniques on triangulated surfaces (techniques that, in addition, are not supported by theoretical results).

Following also the work [505, 506] for color images defined on the plane, we show in Figure 20.4 how to denoise a color image painted on an implicit surface.

The basic idea is to normalize the RGB vector (a three dimensional vector) to a unit vector representing the chroma, and diffuse this unit vector with the harmonic

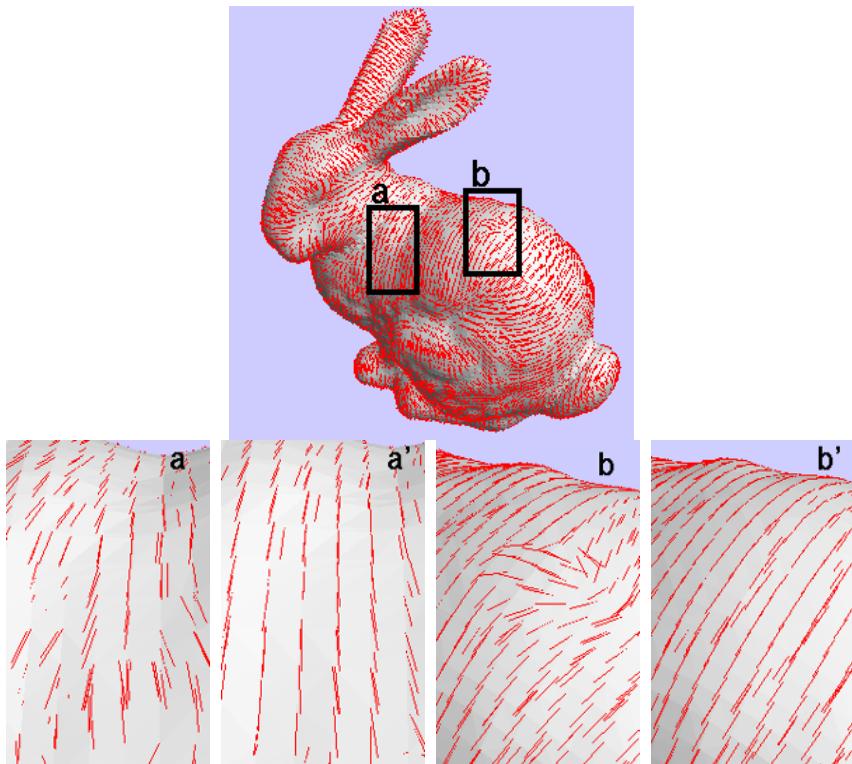


Figure 20.3. Intrinsic vector field regularization. Top: original field of major principal direction of the surface. Details a and b : original field. Details a' and b' : after anisotropic regularization.

maps flow (20.11).² The corresponding magnitude, representing the brightness, is smoothed separately via scalar diffusion flows as those presented before (e.g., the intrinsic heat flow or the intrinsic anisotropic heat flow). That is, we have to regularize a map onto S^2 (the chroma) and another one onto \mathbb{R} (the brightness).

20.3.3 Pattern formation on surfaces via reaction-diffusion flows

The use of reaction-diffusion equations for texture synthesis became very popular in computer graphics following the works of Turk [535] and Witkin and Kass [580]. These works follow original ideas by Turing [534], who showed how reaction diffusion equations can be used to generate patterns. The basic idea in these models is to have a number of “chemicals” that diffuse at different rates

²We re-normalize at every discrete step of the numerical evolution to address deviations from the unit norm due to numerical errors [128]. We could also extend the framework in [8] and apply it to our equations.

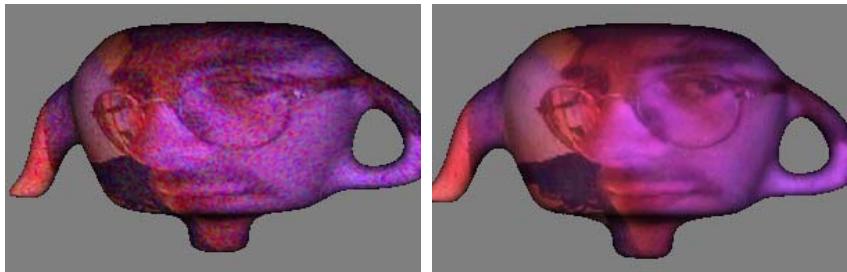


Figure 20.4. Intrinsic vector field regularization. Left: heavy noise has been added to the 3 color channels. Right: color image reconstructed after 20 steps of anisotropic diffusion of the chroma vectors.

and that react with each other. The pattern is then synthesized by assigning a brightness value to the concentration of one of the chemicals. The authors in [535, 580] used their equations for planar textures and textures on triangulated surfaces. By using the framework here described, we can simply create textures on (implicit/implicitized) surfaces, without the elaborate schemes developed in those papers.

Assuming a simple isotropic model with just two chemicals u_1 and u_2 , we have

$$\frac{\partial u_1}{\partial t} = F(u_1, u_2) + D_1 \Delta u_1,$$

$$\frac{\partial u_2}{\partial t} = G(u_1, u_2) + D_2 \Delta u_2,$$

where D_1 and D_2 are two constants representing the diffusion rates and F and G are the functions that model the reaction.

Introducing our framework, if u_1 and u_2 are defined on a surface S implicitly represented as the zero level set of ψ we have

$$\frac{\partial u_1}{\partial t} = F(u_1, u_2) + D_1 \frac{1}{\|\nabla\psi\|} \nabla \cdot (P_{\nabla\psi} \nabla u_1 \parallel \nabla\psi\|), \quad (20.12)$$

$$\frac{\partial u_2}{\partial t} = G(u_1, u_2) + D_2 \frac{1}{\|\nabla\psi\|} \nabla \cdot (P_{\nabla\psi} \nabla u_2 \parallel \nabla\psi\|). \quad (20.13)$$

For simple isotropic patterns, Turk [535] selected

$$F(u_1, u_2) = s(16 - u_1 u_2),$$

$$G(u_1, u_2) = s(u_1 u_2 - u_2 - \beta),$$

where s is a constant and β is a random function representing irregularities in the chemical concentration. Examples of this, for implicit surfaces, are given in Figure 20.5 (the coupled PDE's shown above are run until steady state is achieved).

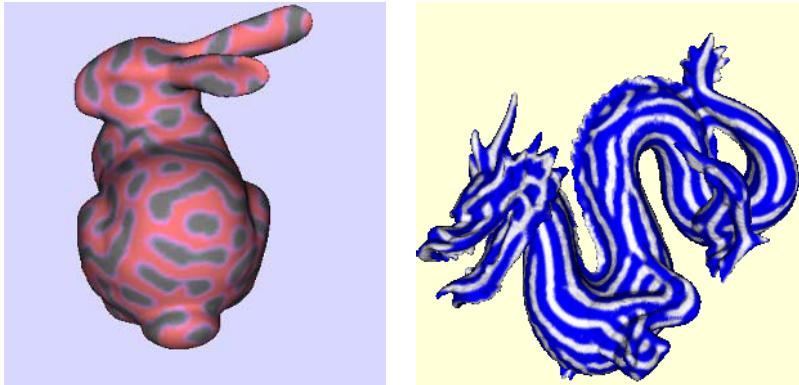


Figure 20.5. Texture synthesis via intrinsic reaction-diffusion flows on implicit surfaces. Left: isotropic. Right: anisotropic. Pseudo-color representation of scalar data is used. The numerical values used in the computations were $D_1 = 1.0$, $D_2 = 0.0625$, $s = 0.025$, $\beta = 12.0 \pm 0.1$, $u_1(0) = u_2(0) = 4.0$.

To simulate anisotropic textures, instead of using additional chemicals as in [535], we use anisotropic diffusion, as suggested in [580]. For this purpose, we replace eq. (20.12) with:

$$\frac{\partial u_1}{\partial t} = F(u_1, u_2) + D_1 \frac{1}{\| \nabla \psi \|} \nabla \cdot ((\vec{d} \cdot P_{\nabla \psi} \nabla u_1) \vec{d} \| \nabla \psi \|), \quad (20.14)$$

where \vec{d} is a vector field tangent to the surface, e.g., the field of the major principal direction (which for our examples has been also accurately computed directly on the implicit surface, using the technique proposed in [377]). Note how this particular selection of the anisotropic reaction-diffusion flow direction provides a texture that helps on the shape perception of the object. Additional patterns can be obtained with different combinations of the reaction and diffusion parts of the flow.

20.3.4 Flow visualization on 3D surfaces

Inspired by the work on line integral convolution [75] and that on anisotropic diffusion [423], the authors of [163] suggested to use anisotropic diffusion to visualize flows in 2D and 3D. The basic idea is, starting from a random image, anisotropically diffuse it in the directions dictated by the flow field. The authors presented very nice results both in 2D (flows on the plane) and 3D (flows on a surface), but once again using triangulated surfaces which introduce many computational difficulties. In a straightforward fashion we can compute these anisotropic diffusion equations on the implicit surfaces with the framework here described, and some results are presented in Figure 20.6. Note the complicated topology and how both the inside and outside parts of the surfaces are easily handled with our



Figure 20.6. Flow visualization on implicit 3D surfaces via intrinsic anisotropic diffusion flows. Left: flow aligned with the major principal direction of the surface. Right: flow aligned with the minor principal direction of the surface. Pseudo-color representation of scalar data is used.

implicit approach. Also note that, when we choose the vector field to be that of one of the principal directions, the result emphasizes the surface shape.

20.4 Concluding remarks

In this chapter, we have described a framework for solving variational problems and PDE's for data defined on surfaces. The technique borrows ideas from the level set theory and the theory of intrinsic flows via harmonic maps. The surface is embedded in a higher dimensional function, and the Euler-Lagrange flow or PDE is solved in the Cartesian coordinate system of this embedding function. The equations are simultaneously intrinsic to the implicit surface and defined on the embedding Cartesian space. With this framework we enjoy accuracy, robustness, and simplicity, as expected from the computation of differential characteristics on iso-surfaces (implicit surfaces) and the use of classical and well established numerical techniques in Cartesian grids. In addition to presenting the general approach, we have exemplified it with equations arising in image processing and computer graphics.

We believe this new framework opens up a large number of theoretical and practical questions. In the theoretical arena, the large amount of results available for harmonic maps (see for example [505] for a review on this) need to be extended to the “implicit harmonic maps” equations presented in this chapter. The effect of perturbations on the surface (zero-level set) on the solutions of the intrinsic PDE should be investigated. This is crucial to understand the desired accuracy of surface implication algorithms. We expect that as with the level set theory (e.g., [109, 184]), these theoretical results will follow. On the practical side, it

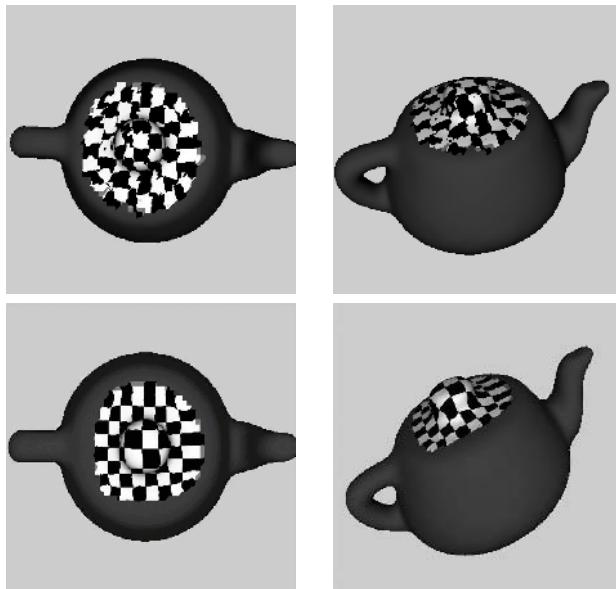


Figure 20.7. Diffusion of a texture map for an implicit teapot (noisy on the top and regularized on the bottom). A checkerboard texture is mapped.

is of interest to address other related equations that appear in the mathematical physics, image processing, and computer graphics literature. For example, how to extend the use of harmonic maps for texture mapping (and not just texture synthesis). This was done for triangulated surfaces in [20, 172, 224]. Also to investigate threshold dynamics and convolution generated motions [252, 359, 452] for implicit surfaces. Other PDE's, like those for image inpainting [39] or image segmentation [85] can be extended to work on implicit surfaces following the theory described in this chapter, expecting the same quality of results that were obtained on the plane. We could also use this framework to experimentally study results as those in [250]. Finally, the use of the approach here presented for regularization in inverse problems, e.g., [187], is of interest as well.

To conclude, we should note that it is natural to ask about the target manifold for the most general form of harmonic maps, when this target is not just the Euclidean space or a unit ball, but a general hypersurface. In [358] we have shown how to extend the framework described here to arbitrary *implicit* target surfaces. The key idea is again to implicitly represent the target manifold as the level-set of a higher dimensional function, and then implement the equations in the Cartesian coordinate system of this new embedding function. In the case of a variational problem, the search for the minimizing map is restricted to the class of maps whose target is the level-set of interest. In the case of partial differential equations, all the equation characteristics are implicitly represented. A set of equations that while defined on the whole Euclidean space, are intrinsic to the implicit target manifold and map into it, is then obtained. See figure 20.7 for an example of

denoising of a texture map from \mathbb{R}^2 onto a “teapot” surface. This result was obtained with the following evolution equation, that guarantees that the diffusion of the map, though performed in a 3D Cartesian grid, is intrinsic to the implicit surface that is the teapot:

$$\frac{\partial \mathbf{u}}{\partial t} = \Delta \mathbf{u} - (\Delta \mathbf{u} \cdot \nabla \psi) \nabla \psi, \quad (20.15)$$

where \mathbf{u} is the map and ψ is the signed distance function to the teapot.

Note also that general motion of curves on implicit surfaces is studied in [42, 114]. These works, together with the one here presented, provide then the basic framework for solving generic PDE’s on implicit surfaces.

Acknowledgements

The implicitation software was provided by Santiago Betelú and H. Zhao. This work was partially supported by grants from the Office of Naval Research ONR-N00014-97-1-0509 and ONR-N00014-97-1-0027, the Office of Naval Research Young Investigator Award, the Presidential Early Career Awards for Scientists and Engineers (PECASE), a National Science Foundation CAREER Award, the National Science Foundation Learning and Intelligent Systems Program (LIS), NSF-DMS-9706827, NSF DMS 0074735, ARO-DAAG-55-98-1-0323 and IIE-Uruguay.

Part VIII

Medical Image Analysis

21

Knowledge-Based Segmentation of Medical Images

Michael Leventon, Eric Grimson, Olivier Faugeras, Ron Kikinis and William Wells III

Abstract

Knowledge-based medical image segmentation provides application-specific context by constructing prior models and incorporating them into the segmentation process. In this chapter, we present recent work that integrates intensity, local curvature, and global shape information into level set based segmentation of medical images. The object intensity distribution is modeled as a function of signed distance from the object boundary, which fits naturally into the level set framework. Curvature profiles act as boundary regularization terms specific to the shape being extracted, as opposed to uniformly penalizing high curvature. We describe a representation for deformable shapes and define a probability distribution over the variances of a set of training shapes. The segmentation process embeds an initial curve as the zero level set of a higher dimensional surface, and evolves the surface such that the zero level set converges on the boundary of the object to be segmented. At each step of the surface evolution, we estimate the pose and shape of the object in the image, based on the prior shape information and the image information. We then evolve the surface globally, towards the estimate, and locally, based on image information and curvature. Results are demonstrated on magnetic resonance (MR) and computed tomography (CT) imagery.

21.1 Introduction

Medical image processing applications such as surgical planning, navigation, simulation, diagnosis, and therapy evaluation all benefit from segmentation of anatomical structures from medical images. By segmentation, we refer to the process of labeling individual voxels in the volumetric scan by tissue type, based

on properties of the observed intensities as well as anatomical knowledge about normal subjects.

Segmentation is typically performed using a mix of automated techniques and semi-automated techniques. With CT data, segmentation of some structures can be performed just using intensity thresholding. In general, however, segmentation is challenging and requires more sophisticated algorithms and significant human input. For example, the distribution of intensity values corresponding to one structure may vary throughout the structure and also overlap those of another structure, defeating intensity-based segmentation techniques. The strength of an “edge” at the boundary of the structure may vary or be weak relative to the texture inside the object, creating difficulties for gradient-based boundary detection methods.

Boundary finding segmentation methods such as Snakes [263], are generally local algorithms that require some feature (such as an edge) to be present along the boundary of the object, and gravitate toward that feature. These methods may be sensitive to the starting position and may “leak” through the boundary of the object if the edge feature is not salient enough in a certain region in the image.

Level set segmentation involves solving the energy-based active contours minimization problem by the computation of geodesics or minimal distance curves [85, 270, 331]. In this approach, a curve is embedded as a zero level set of a higher dimensional surface [401, 472]. The entire surface is evolved to minimize a metric defined by the curvature and image gradient.

In [593], segmentation is performed by evolving a curve to maximally separate predetermined statistics inside and outside the curve. Paragios and Deriche in [409], build prior texture models and perform segmentation by combining boundary and region information. These methods include both global and local information, adding robustness to noise and weak boundaries and are formulated using level set techniques providing the advantages of numerical stability and topological flexibility.

Two common segmentation techniques, pixel classification [208, 261] and boundary localization [85, 263, 331, 593], typically include both an image term and a regularization term. In [236, 261], a regularization effect is included as a Markov prior expressing that structure labels should not vary sporadically in a local area (eliminating fragmentation). In [85, 263, 593], a smoothness term is added to penalize high curvature in the evolving curve. In [321], the regularizer penalizes only the smaller of the two curvatures when segmenting curve-like structures such as blood vessels. In most curve evolution methods, the amount of regularization required is a manually tuned parameter dependent on the shape properties of the object and noise in the image.

When segmenting or localizing an anatomical structure, having prior information about the expected shape of that structure can significantly aid in the segmentation process. Both Cootes, *et al.* [133] and Wang and Staib [564] find a set of corresponding points across a set of training images and construct a statistical model of shape variation that is then used in the localization of the boundary. Staib and Duncan [492] incorporate global shape information into the segmenta-

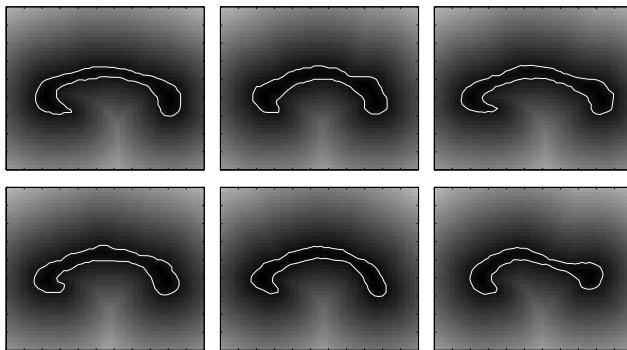


Figure 21.1. Corpus callosum outlines for 6 out of 51 patients in the training set embedded as the zero level set of a higher dimensional signed distance surface.

tion task by using an elliptic Fourier decomposition of the boundary and placing a Gaussian prior on the Fourier coefficients.

Our approach to object segmentation extends geodesic active contours by incorporating prior information into the evolution process. In section 21.2, we compute a statistical shape model over a training set of curves. Section 21.3 describes the segmentation of a structure from an image, where an active contour is evolved both locally, based on image gradients and curvature, and globally to a maximum *a posteriori* estimate of shape and pose. Section 21.4 explores the addition of intensity and curvature priors to the segmentation process, and the results are presented in Section 21.5.

21.2 Probability distribution on shapes

To incorporate shape information into the process of segmenting an object in an image, we consider a probabilistic approach, and compute a prior on shape variation given a set of training instances. To build the shape model, we choose a representation of curves, and then define a probability density function over the parameters of the representation.

21.2.1 Curve representation

Given a training set of example segmentations of an object (*i.e.* boundary curves in the 2d case), we describe a method of learning a shape distribution. Each curve in the training dataset is roughly aligned and embedded as the zero level set of a higher dimensional surface, u , whose height is sampled at regular intervals (say N^d samples, where d is the number of dimensions). The embedding function chosen is the commonly used signed distance function [472], where each sample encodes the distance to the nearest point on the curve, with negative values inside

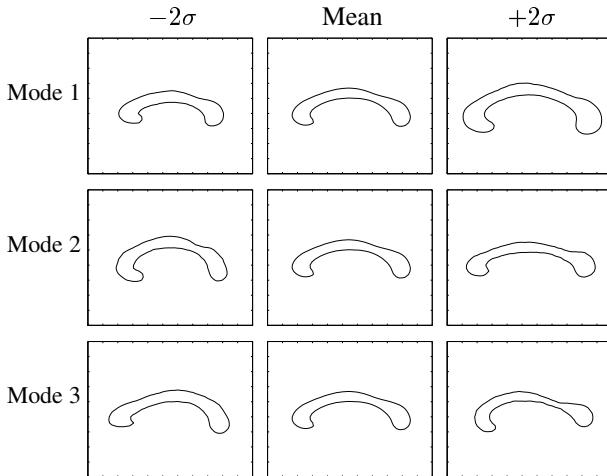


Figure 21.2. The three primary modes of variance of the corpus callosum training dataset.

the curve. Each such surface (distance map) can be considered a point in a high dimensional space ($u \in \mathbb{R}^{N^d}$). The training set, \mathcal{T} , consists of a set of surfaces $\mathcal{T} = \{u_1, u_2, \dots, u_n\}$. Our goal is to build a shape model over this distribution of surfaces. Since a signed distance map is uniquely determined from the zero level set, each distance map has a large amount of redundancy. Furthermore, the collection of curves in the training set presumably has some dependence, as they are shapes of the same class of object, introducing more redundancy in the training set. The cloud of points corresponding to the training set is approximated to have a Gaussian distribution, where most of the dimensions of the Gaussian collapse, leaving the principal modes of shape variation.

The mean surface, μ , is computed by taking the mean of the signed distance functions, $\mu = \frac{1}{n} \sum u_i$. The variance in shape is computed using Principal Component Analysis (PCA). The mean shape, μ , is subtracted from each u_i to create an mean-offset map, \hat{u}_i . Each such map, \hat{u}_i , is placed as a column vector in an $N^d \times n$ -dimensional matrix M . Using Singular Value Decomposition, the covariance matrix $\frac{1}{n} M M^\top$ is decomposed as:

$$U \Sigma U^\top = \frac{1}{n} M M^\top \quad (21.1)$$

where U is a matrix whose column vectors represent the set of orthogonal modes of shape variation and Σ is a diagonal matrix of corresponding singular values. An estimate of a novel shape, u , can be represented by k principal components in a k -dimensional vector of coefficients, α .

$$\alpha = U_k^\top (u - \mu) \quad (21.2)$$

where U_k is a matrix consisting of the first k columns of U that is used to project a surface into the eigen-space. Given the coefficients α , an estimate of the shape u , namely \tilde{u} , is reconstructed from U_k and μ .

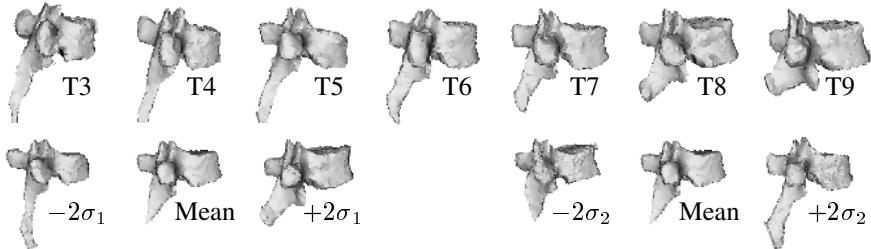


Figure 21.3. **Top:** Three-dimensional models of seven thoracic vertebrae (T3-T9) used as training data. **Bottom left and right:** Extracted zero level set of first and second largest mode of variation respectively.

$$\tilde{u} = U_k \alpha + \mu \quad (21.3)$$

Note that in general \tilde{u} will not be a true distance function, since convex linear combinations of distance maps do not produce distance maps. However, the surfaces generally still have advantageous properties of smoothness, local dependence, and zero level sets consistent with the combination of original curves.

Under the assumption of a Gaussian distribution of shape represented by α , we can compute the probability of a certain curve as:

$$P(\alpha) = \frac{1}{\sqrt{(2\pi)^k |\Sigma_k|}} \exp\left(-\frac{1}{2}\alpha^\top \Sigma_k^{-1} \alpha\right) \quad (21.4)$$

where Σ_k contains the first k rows and columns of Σ . We also have considered modeling the distribution of shapes as a mixture of Gaussians or using a Parzen window density estimator, but keep to a Gaussian prior in this work.

Figure 21.1 shows a few of the 51 training curves used to define the shape models of the corpus callosum. The original segmentations of the images are shown as white curves. The outlines are overlaid on the signed-distance map. Before computing and combining the distance maps of these training shapes, the curves were aligned using centroids and second moments to approximate the correspondence. Figure 21.2 illustrates zero level sets corresponding to the means and three primary modes of variance of the shape distribution of the corpus callosum. Figure 21.3 shows the zero level set (as a triangle surface model) of seven rigidly aligned vertebrae of one patient used as training data. The zero level sets of the two primary modes are shown as well. Note that for both the corpus and the vertebrae, the mean shapes and primary modes appear to be reasonable representative shapes of the classes of objects being learned. In the case of the corpus callosum, the first mode seems to capture size, while the second mode roughly captures the degree of curvature of the corpus. The third mode appears to represent the shifting of the bulk of the corpus from front to back.

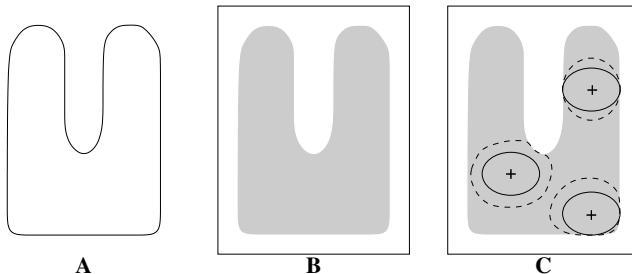


Figure 21.4. (a) The curve expected to be found in the image. (b) The image containing the shape to be segmented. (c) The same solid curve appears in three different locations, and the direction of evolution depends on the position of the evolving curve with respect to the object. The dotted lines show a later step in evolution given the curve's position and shape.

21.2.2 *The correspondence problem*

When measuring shape variance of a certain part of an object across a population, it is important to compare like parts of the object. For example, when looking at variances in the shape of the vertebrae, if two training examples are misaligned and a process of one is overlapping a notch of the other, then the model will not be capturing the appropriate anatomical shape variance seen across vertebrae.

One solution to the correspondence problem is to explicitly generate all pointwise correspondences to ensure that comparisons are done consistently. Finding correspondences in general is a difficult problem. Manually localizing corresponding landmarks is tedious, while automatic landmark detection is prone to errors, especially when dealing with 3D objects. In [133], Cootes requires the labeling of 123 corresponding landmarks on each of 72 training instances when building a 2D model of the region of the brain around the ventricles. While this method does an excellent job of performing the model-based matching, in many applications, especially in 3D, careful labeling of such a large training set is infeasible.

Another approach to correspondence is to roughly align the training data before performing the comparison and variance calculation. A rough alignment will not match every part of each training instance perfectly, so one must consider the robustness of the representation to misalignment. Turk and Pentland [537] introduced Eigenfaces as a method of building models for recognition. Each image in a set of face images ($N \times N$ array of intensities) is considered as a point in an N^2 -dimensional space, from which the principal components are computed. The Eigenface method is similar to our method of combining signed distance maps of binary images, with an important distinction. Any slight misalignment in the faces compares the intensity variance between independent objects, while slightly misaligned pixels in a distance map are generally very highly correlated. Smoothing a grayscale or binary image propagates information spatially as well, increasing the correlation between neighboring pixels, but results in loss of infor-

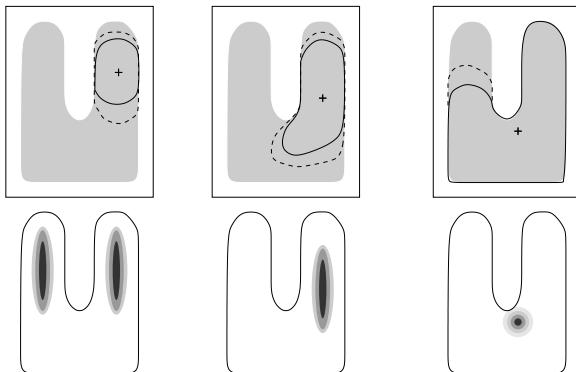


Figure 21.5. Three steps in the evolution process. The evolving curve is shown solid, superimposed on the image (top row). The curve is matched to the expected curve to obtain a PDF over pose (bottom row). The next evolution step (based on pose and shape) is shown as the dotted line.

mation, whereas no information about the binary image is lost by computing its signed distance function.

Using the signed distance map as the representation of shape provides tolerance to slight misalignment of object features, in the attempt to avoid having to solve the general correspondence problem. In the examples presented here, the rough rigid alignment of the training instances resulted in the model capturing the shape variances inherent in the population due to the dependence of nearby pixels in the shape representation.

21.3 Shape priors and geodesic active contours

Given a curve representation (the k -dimensional vector α) and a probability distribution on α , the prior shape information can be folded into the segmentation process. This section describes adding a term to the level set evolution equation to pull the surface in the direction of the maximum *a posteriori* shape and position of the final segmentation.

21.3.1 Geodesic active contours for segmentation

The snake methodology defines an energy function $E(\mathcal{C})$ over a curve \mathcal{C} as the sum of an internal and external energy of the curve, and evolves the curve to minimize the energy [263].

$$E(\mathcal{C}) = \beta \int |\mathcal{C}'(q)|^2 dq - \lambda \int |\nabla I(\mathcal{C}(q))| dq \quad (21.5)$$

In [85], Caselles, *et al.* derive the equivalence of geodesic active contours to the traditional energy-based active contours (snakes) framework by first reducing the

minimization problem to the following form:

$$\min_{\mathcal{C}(q)} \int g(|\nabla I(\mathcal{C}(q))|) |\mathcal{C}'(q)| dq \quad (21.6)$$

where g is a function of the image gradient (usually of the form $\frac{1}{1+|\nabla I|^2}$). Using Euler-Lagrange, the following curve evolution equation is derived [85]

$$\frac{\partial \mathcal{C}(t)}{\partial t} = g\kappa\mathcal{N} - (\nabla g \cdot \mathcal{N})\mathcal{N} \quad (21.7)$$

where κ is the curvature and \mathcal{N} is the unit normal. By defining an embedding function u of the curve \mathcal{C} , the update equation for a higher dimensional surface is computed.

$$\frac{\partial u}{\partial t} = g(c + \kappa) |\nabla u| + \nabla u \cdot \nabla g \quad (21.8)$$

where c is an image-dependent balloon force added to force the contour to flow outward [85, 127]. In this level set framework, the surface, u , evolves at every point perpendicular to the level sets as a function of the curvature at that point and the image gradient.

21.3.2 Estimation of pose and shape

In addition to evolving the level set based on the curvature and the image term, we include a term that incorporates information about the shape of the object being segmented. To add such a global shape force to the evolution, the pose of the evolving curve with respect to the shape model must be known (see Figures 21.4 and 21.5). Without an estimate of the pose, the shape model cannot adequately constrain or direct the evolution. Therefore, at each step of the curve evolution, we seek to estimate the shape parameters, α , and the rigid pose parameters, p , of the final curve using a maximum *a posteriori* approach.

$$\langle \alpha_{\text{MAP}}, p_{\text{MAP}} \rangle = \underset{\alpha, p}{\operatorname{argmax}} P(\alpha, p | u, \nabla I) \quad (21.9)$$

In this equation, u is the evolving surface at some point in time, whose zero level set is the curve that is segmenting the object. The term ∇I is the gradient of the image containing the object to be segmented. By our definition of shape and pose, the final segmentation curve is completely determined by α and p . Let u^* be the estimated final curve, which can be computed from α and p . Therefore, we also have

$$u_{\text{MAP}}^* = \underset{u^*}{\operatorname{argmax}} P(u^* | u, \nabla I) \quad (21.10)$$

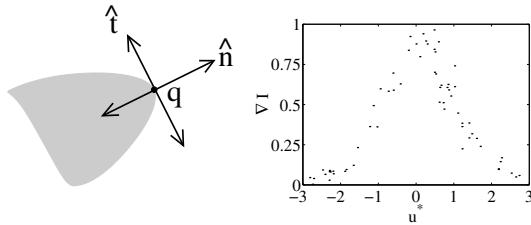


Figure 21.6. The relationship between the distance map and the image gradient.

To compute the maximum *a posteriori* final curve, we expand the terms from Eq. 21.9 using Bayes' Rule.

$$\begin{aligned} P(\alpha, p \mid u, \nabla I) &= \frac{P(u, \nabla I \mid \alpha, p)P(\alpha, p)}{P(u, \nabla I)} \\ &= \frac{P(u \mid \alpha, p)P(\nabla I \mid \alpha, p, u)P(\alpha)P(p)}{P(u, \nabla I)} \end{aligned} \quad (21.11)$$

Note that the preceding step assumes that shape is independent from pose. Since our current model does not attempt to capture the relationships between these two quantities, this is reasonable. Future work may incorporate positional priors (and relative positional priors between objects) into our shape model. We proceed by defining each term of Eq. 21.11 in turn. We discard the normalization term in the denominator as it does not depend on shape or pose.

Inside Term.

The first term in Eq. 21.11 computes the probability of a certain evolving curve, u , given the shape and pose of the final curve, u^* (or $\langle \alpha, p \rangle$). Notice that this term does not include any image information whatsoever. Given our method of initializing the curve with a point *inside* the object, it is reasonable to assume that the curve should remain inside the object throughout the evolution. Therefore, if the evolving curve lies completely inside the final curve, then it is more likely than a curve that lies partially or fully outside the final curve. We model this term as a Laplacian density function over V_{outside} , the volume of the curve u that lies outside the curve u^* .

$$P(u \mid \alpha, p) = \exp(-V_{\text{outside}}) \quad (21.12)$$

This term assumes that any curve u lying inside u^* is equally likely. Since the initial curve can be located at any point inside the object and the curve can evolve along any path, we do not favor any such curve.

Gradient Term.

The second term in Eq. 21.11 computes the probability of seeing certain image gradients given the current and final curves. Consider the relationship between u^* and $|\nabla I|$ when u^* correctly outlines the boundary of the object (see Figure 21.6). Notice that the distance map u^* is linear along the normal direction of the curve at

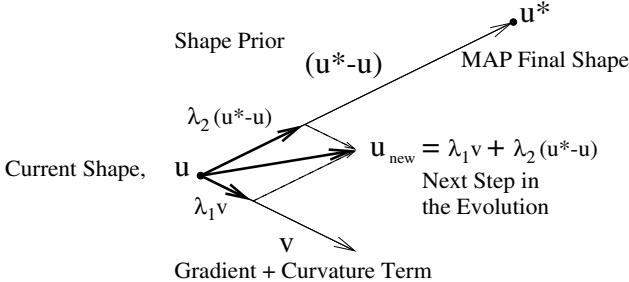


Figure 21.7. Illustration of the various terms in the evolution of the surface, u . The surface u^* is the maximum *a posteriori* final shape. To update u , we combine the standard gradient and curvature update term, v , and the direction of the MAP final shape, $u^* - u$.

any boundary point, q , and $u^*(q) = 0$. Furthermore, under the assumption that the object boundary is a smoothed step edge, $|\nabla I|$ approximates a Gaussian along the normal at q . Therefore, we'd expect the relationship between $|\nabla I|$ and u^* to be Gaussian in nature. Figure 21.6 shows an example scatter plot of these quantities when u^* is aligned with the object boundary. Let $h(u^*)$ be the best fit Gaussian to the samples $(u^*, |\nabla I|)$. We model the gradient probability term as a Laplacian of the goodness of fit of the Gaussian.

$$P(\nabla I | u^*, u) = \exp(-|h(u^*) - |\nabla I||^2) \quad (21.13)$$

Shape and pose priors.

The last two terms in Eq. 21.11 are based on our prior models, as described in Section 21.2. Our shape prior is a Gaussian model over the shape parameters, α , with shape variance Σ_k .

$$P(\alpha) = \frac{1}{\sqrt{(2\pi)^k |\Sigma_k|}} \exp\left(-\frac{1}{2} \alpha^\top \Sigma_k^{-1} \alpha\right) \quad (21.14)$$

In our current framework, we seek to segment one object from an image, and do not retain prior information on the likelihood of the object appearing in a certain location. Thus, we simply assume a uniform distribution over pose parameters, which can include any type of transformation, depending on application.

$$P(p) = \mathcal{U}(-\infty, \infty) \quad (21.15)$$

Currently we are modeling translation and rotation. We feel, however, that positional priors could provide a rich source of information to explore in the future, especially when segmenting multiple objects from a single image that may have clear prior relative poses, or when a distribution over pose in a fixed image-based coordinate system is known.

These terms define the maximum *a posteriori* estimator of shape and pose, which estimates the final curve or segmentation of the object. For efficiency, these quantities are computed only in a narrow band around the zero level set of the evolving surface, and the MAP pose and shape are re-estimated at each evolution

step using simple gradient ascent on the log probability function in Eq. 21.11. While each ascent may yield a local maximum, the continuous re-estimation of these parameters as the surface evolves generally results in convergence on the desired maximum. Next, we incorporate this information into the update equation commonly used in level set segmentation.

21.3.3 Evolving the surface

Initially, the surface, u , is assumed to be defined by at least one point that lies inside the object to be segmented. Given the surface at time t , we seek to compute an evolution step that brings the curve closer to the correct final segmentation based on local gradient and global shape information.

The level set update expression shown in Eq. 21.8 provides a means of evolving the surface u over time towards the solution to the original curve-minimization problem stated in Eq. 21.6. Therefore, the shape of the surface at time $t+1$ can be computed from $u(t)$ by:

$$u(t+1) = u(t) + \lambda_1 (g(c + \kappa) |\nabla u(t)| + \nabla u(t) \cdot \nabla g) \quad (21.16)$$

where λ_1 is a parameter defining the update step size.

By estimating the final surface u^* at a given time t , (Section 21.3.2), we can also evolve the surface in the direction of the maximum *a posteriori* final surface:

$$u(t+1) = u(t) + \lambda_2 (u^*(t) - u(t)) \quad (21.17)$$

where $\lambda_2 \in [0, 1]$ is the linear coefficient that determines how much to trust the maximum *a posteriori* estimate. Combining these equations yields the final expression for computing the surface at the next step.

$$\begin{aligned} u(t+1) = & u(t) + \lambda_1 (g(c + \kappa) |\nabla u(t)| + \nabla u(t) \cdot \nabla g) \\ & + \lambda_2 (u^*(t) - u(t)) \end{aligned} \quad (21.18)$$

Figure 21.7 illustrates this evolution. The two parameters λ_1 and λ_2 are used to balance the influence of the shape model and the gradient-curvature model. The parameters also determine the overall step size of the evolution. The tradeoff between shape and image depends on how much faith one has in the shape model and the imagery for a given application. Currently, we set these parameters empirically for a particular segmentation task, given the general image quality and shape properties.

21.4 Statistical Image-Surface Relationship

This section describes a method of incorporating prior intensity and curvature models into the segmentation process. We define now the surface U as the signed distance function to the boundary curve \mathcal{C} . Therefore, $U(\mathbf{x})$ is both the height of the surface U at the position \mathbf{x} and the signed distance from \mathbf{x} to the nearest point

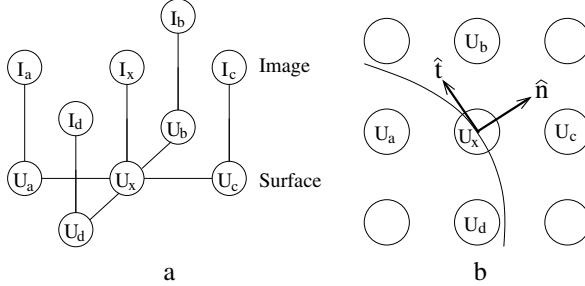


Figure 21.8. (a) Statistical dependency of node samples of the surface U and pixels of image I . (b) The directions of the normal and tangent of U are shown at the position x . The contour curve of height U_x is drawn.

on the curve \mathcal{C} . Instead of estimating the position of \mathcal{C} directly, we estimate the value of U at every position \mathbf{x} in the image. Once U is computed, the boundary of the object is found by extracting the zero level set of U .

We define a statistical dependency network over the surface U and the image I . To estimate the value U_x of U at a certain position \mathbf{x} in the image, we maximize $P(U_x | I, U \setminus U_x)$, namely the probability of the height of the surface at that point, given the entire image and the rest of the surface. This expression is difficult to model, given all the dependencies. We therefore simplify the problem to a Markov network with only local dependencies. The links of the nodes in Figure 21.8a represent the dependencies considered. We assume the height of the surface at a certain point depends only on the intensity value of the image at that point, and the neighboring heights of the surface. This assumption can be expressed as:

$$P(U_x | I, U \setminus U_x) = P(U_x | I_x, U_{\mathcal{N}(x)}) \quad (21.19)$$

where $\mathcal{N}(\mathbf{x})$ is the neighborhood of \mathbf{x} .

Using properties of our network, we derive an expression for the above probability consisting of terms we will estimate from a set of training data.

$$P(U_x | I_x, U_{\mathcal{N}(x)}) = \frac{P(U_x, I_x, U_{\mathcal{N}(x)})}{P(I_x, U_{\mathcal{N}(x)})} \quad (21.20)$$

$$= \frac{P(I_x, U_x)}{P(I_x, U_{\mathcal{N}(x)})} \frac{P(U_x, I_x, U_{\mathcal{N}(x)})}{P(I_x, U_x)} \quad (21.21)$$

$$= \frac{P(I_x, U_x)}{P(I_x, U_{\mathcal{N}(x)})} P(U_{\mathcal{N}(x)} | I_x, U_x) \quad (21.22)$$

$$= \frac{P(I_x, U_x) P(U_{\mathcal{N}(x)} | U_x)}{P(I_x, U_{\mathcal{N}(x)})} \quad (21.23)$$

The definition of conditional probability is used in Equations 21.20 and 21.22, and we are multiplying by the identity in Equation 21.21. In the final expression, we note that when conditioned on U_x , the heights of the surface at neighboring locations do not depend on I_x .

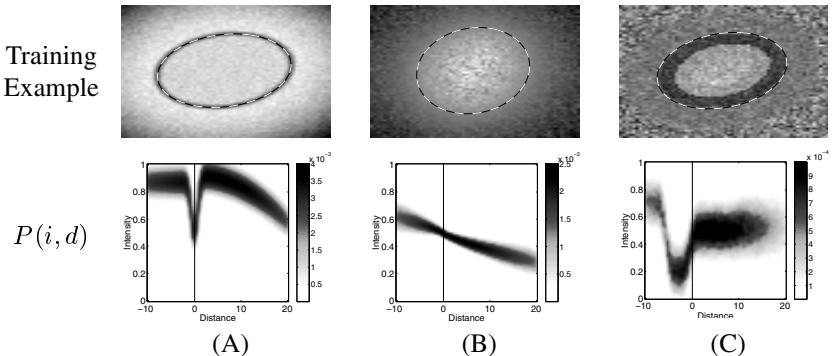


Figure 21.9. Top: One example of three training sets of random ellipses under different imaging conditions. The dotted contour shows the boundary of the object. Bottom: The joint intensity/distance-to-boundary PDF derived from the training set. Left and right of the black vertical line indicate the intensity profile inside and outside the object.

The first term in Equation 21.23 is the *image term*, that relates the intensity and the surface at \mathbf{x} . The next term relates the neighborhood of U about \mathbf{x} to $U_{\mathbf{x}}$, and will act as a *regularization term*. The following two sections describe a means of estimating these functions based on prior training data. The denominator of Equation 21.23 does not depend on $U_{\mathbf{x}}$, and thus is ignored during the estimation process.

21.4.1 Intensity Model

We define a joint distribution model, $P(i, d)$, that relates intensity values, i , to signed distances d , based on a set of training images and training segmentations. Let $\{I_1, I_2, \dots, I_n\}$ be a set of images of the same modality containing the same anatomical structure of various subjects and $\{\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_n\}$ be the set of corresponding segmented boundary curves of the anatomical structure. Let U_j be the signed distance map to the closed curve \mathcal{C}_j . The training set \mathcal{T} , is a set of image-surface pairs, $\mathcal{T} = \{(I_1, U_1), \dots, (I_n, U_n)\}$. We use a Parzen window density estimator with a Gaussian windowing function to model the joint distribution. The PDF is the mean of many 2D Gaussians centered at the positions $\langle I_j(\mathbf{x}), U_j(\mathbf{x}) \rangle$ for every pixel in every training image:

$$P(i, d) = \frac{1}{Z} \sum_{j=1}^n \sum_{\mathbf{x} \in \mathcal{X}} \exp \left(-\frac{(i - I_j(\mathbf{x}))^2}{2\sigma_i^2} - \frac{(d - U_j(\mathbf{x}))^2}{2\sigma_d^2} \right) \quad (21.24)$$

where \mathcal{X} is the set of all positions in the image, σ_i and σ_d are the variances of the Parzen kernel, and the normalization factor $Z = (2\pi\sigma_i\sigma_d n|\mathcal{X}|)^{-1}$.

Figure 21.9 illustrates the estimated joint probability density functions for random ellipses with various contrast scenarios. Notice that Ellipse A can be easily segmented using a gradient based method, but cannot be thresholded, as the inten-

sity distribution inside and outside the object overlap significantly. On the other hand, simple gradient methods would fail to extract Ellipse B due to high texture, while a single intensity threshold isolates the object. Neither scheme would work for Ellipse C, as higher level knowledge is required to know which edge is the correct boundary.

Figure 21.10 illustrates the estimated joint PDFs for the corpus callosum and slices of two acquisitions of the femur. In the sagittal scan of the femur (Figure 21.10d), the dark layer around the brighter, textured region is cortical bone, which gives essentially no MR signal. Without prior knowledge of the intensity distribution of the entire femur, the cortical bone could easily be missed in a segmentation. However, by training a joint distribution over intensity and signed distance, we generate a PDF that encodes the fact that the dark region is a part of the object. This example is similar to the synthetic ellipse in Figure 21.10a.

21.4.2 Curvature Model

Boundary detection methods commonly use a regularization term to uniformly penalize regions of high curvature, independent of the structure being segmented [263, 85, 593]. Our goal is to use the term $P(U_{\mathcal{N}(\mathbf{x})} | U_{\mathbf{x}})$ as a local regularizer, but one that can be estimated from a set of training data, and thereby tuned to the appropriate application. One method of modeling this function is to create a 5D Parzen joint PDF over $U_{\mathbf{x}}$ and its four neighbors. A drawback of this approach is that a 5D space is difficult to fill and thus requires many training samples. Furthermore, the neighborhood depends on the embedded coordinate system, and would give different results based on the pose of the object in the image (consider a square aligned with the axes versus one rotated by 45°). Therefore, instead of considering the four neighbors (two horizontal and two vertical) in the “arbitrary” embedded coordinate system, we use four (interpolated) neighbors in the direction of the local normal (\hat{n}) and tangent (\hat{t}) to the embedded curve (see Figure 21.8b). In other words, we reparameterize to an intrinsic coordinate system based on the surface:

$$P(U_{\mathcal{N}(\mathbf{x})} | U_{\mathbf{x}}) = P(U_{h+}, U_{h-}, U_{v+}, U_{v-} | U_{\mathbf{x}}) \quad (21.25)$$

$$= P(U_{\hat{n}+}, U_{\hat{n}-}, U_{\hat{t}+}, U_{\hat{t}-} | U_{\mathbf{x}}) \quad (21.26)$$

$$= P(U_{\hat{n}+}, U_{\hat{n}-} | U_{\mathbf{x}})P(U_{\hat{t}+}, U_{\hat{t}-} | U_{\mathbf{x}}) \quad (21.27)$$

The last step makes the assumption that the neighbors in the normal direction and those in the tangent direction are conditionally independent given $U_{\mathbf{x}}$.

We now model the terms in Eq. 21.27 based on properties of the surface U and prior knowledge of the boundary to be segmented. While U is considered to be a finite network of nodes and $U_{\mathbf{x}}$ a particular node, we define $u(\mathbf{x})$ to be the analogous continuous surface over \mathbf{x} . We derive the relationship between $U_{\mathbf{x}}$ and its neighbors by considering the first and second derivatives of $u(\mathbf{x})$ in the directions of the normal and tangent to the level set at \mathbf{x} . The unit normal and unit

tangent to the underlying level set of u are defined as:

$$\hat{\mathbf{n}} = \frac{\nabla u}{|\nabla u|} = \frac{1}{(u_x^2 + u_y^2)^{\frac{1}{2}}} \begin{bmatrix} u_x \\ u_y \end{bmatrix} \quad (21.28)$$

$$\hat{\mathbf{t}} = \frac{1}{(u_x^2 + u_y^2)^{\frac{1}{2}}} \begin{bmatrix} -u_y \\ u_x \end{bmatrix} \quad (21.29)$$

To compute the directional derivatives, we first evaluate the second-order Taylor expansion of u at \mathbf{x} in the arbitrary direction of \mathbf{w} :

$$u(\mathbf{x} + \lambda\mathbf{w}) = u(\mathbf{x}) + \lambda \frac{\partial u}{\partial \mathbf{x}} \mathbf{w} + \frac{\lambda^2}{2} \mathbf{w}^\top \frac{\partial^2 u}{\partial \mathbf{x}^2} \mathbf{w} + \dots \quad (21.30)$$

which yields a general expression for derivatives of u in the direction of \mathbf{w} :

$$\frac{\partial u}{\partial \mathbf{w}} = \frac{\partial u}{\partial \mathbf{x}} \mathbf{w} \quad (21.31)$$

$$\frac{\partial^2 u}{\partial \mathbf{w}^2} = \mathbf{w}^\top \frac{\partial^2 u}{\partial \mathbf{x}^2} \mathbf{w} \quad (21.32)$$

Evaluating the first partial in both the directions of $\hat{\mathbf{n}}$ and $\hat{\mathbf{t}}$ yields:

$$\frac{\partial u}{\partial \hat{\mathbf{n}}} = \frac{\partial u}{\partial \mathbf{x}} \hat{\mathbf{n}} = |\nabla u| \quad (21.33)$$

$$\frac{\partial u}{\partial \hat{\mathbf{t}}} = \frac{\partial u}{\partial \mathbf{x}} \hat{\mathbf{t}} = 0 \quad (21.34)$$

These results are not surprising given that $\hat{\mathbf{n}}$ by definition lies in the direction of the gradient. Note that when u is a signed distance map, $\frac{\partial u}{\partial \hat{\mathbf{n}}} = |\nabla u| = 1$.

We now compute expressions for the second derivatives:

$$\frac{\partial^2 u}{\partial \hat{\mathbf{n}}^2} = \hat{\mathbf{n}}^\top \frac{\partial^2 u}{\partial \mathbf{x}^2} \hat{\mathbf{n}} = \frac{u_{xx}u_x^2 + 2u_{xy}u_xu_y + u_{yy}u_y^2}{u_x^2 + u_y^2} \quad (21.35)$$

$$\frac{\partial^2 u}{\partial \hat{\mathbf{t}}^2} = \hat{\mathbf{t}}^\top \frac{\partial^2 u}{\partial \mathbf{x}^2} \hat{\mathbf{t}} = \frac{u_{xx}u_y^2 - 2u_{xy}u_xu_y + u_{yy}u_x^2}{u_x^2 + u_y^2} \quad (21.36)$$

In the special case when u is a signed distance function, it can be shown that $\frac{\partial^2 u}{\partial \hat{\mathbf{n}}^2} = 0$ and $\frac{\partial^2 u}{\partial \hat{\mathbf{t}}^2} = \kappa$ where the curvature of the underlying level set, κ , is given as:

$$\kappa = \frac{u_{xx}u_y^2 - 2u_xu_yu_{xy} + u_{yy}u_x^2}{(u_x^2 + u_y^2)^{\frac{3}{2}}} \quad (21.37)$$

In summary, here are the values of the first and second partial derivatives of u in the tangent and normal directions under the constraint that the surface u is a continuous signed distance map:

$$\frac{\partial u}{\partial \hat{\mathbf{n}}} = 1 \quad \frac{\partial u}{\partial \hat{\mathbf{t}}} = 0 \quad \frac{\partial^2 u}{\partial \hat{\mathbf{n}}^2} = 0 \quad \frac{\partial^2 u}{\partial \hat{\mathbf{t}}^2} = \kappa \quad (21.38)$$

We can use this information about the ideal distance map and underlying curvatures to regularize U at \mathbf{x} . Consider the formulas for (centered) finite differences

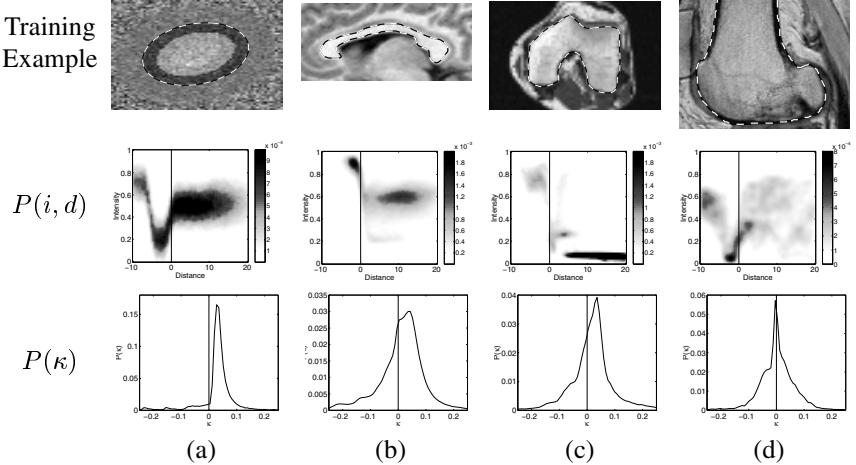


Figure 21.10. Top: One example of each of four training sets of objects. Middle: The joint intensity/distance-to-boundary PDF derived from the training set. Bottom: The curvature profile of each object class. Notice that the ellipse class has only positive curvature.

of an arbitrary 1D function.

$$\frac{df(x)}{ds} \approx \frac{f(x+s) - f(x-s)}{2s} \quad (21.39)$$

$$\frac{d^2f(x)}{ds^2} \approx \frac{f(x+s) + f(x-s) - 2f(x)}{s^2} \quad (21.40)$$

Notice that $f(x)$ does not appear in $\frac{df(x)}{ds}$, indicating that changing f only at one position at a time cannot change the local slope. However, changing $f(x)$ does directly affect the second derivative, so we model the relationship between U_x and its neighbors to adhere to the second derivative constraints.

We define the likelihood of the neighbors in the normal direction to be:

$$P(U_{\hat{n}+}, U_{\hat{n}-} | U_x) = \frac{1}{Z_1} \exp \left(-\frac{(U_{\hat{n}+} + U_{\hat{n}-} - 2U_x)^2}{2\sigma_1^2} \right) \quad (21.41)$$

where Z_1 is a normalization factor and σ_1 determines the strength of the constant-normal-direction constraint. This has the effect of keeping the gradient magnitude of the surface constant (however not necessarily 1), preventing the surface from evolving arbitrarily. Typically one needs to extract the zero level set and reinitialize the distance function from time to time during the evolution. This direction of regularization reduces the need to reinitialize the surface.

As for the likelihood of the neighbors in the tangent direction, we seek an expression that takes into account the curvature profile of the training data. A convex shape, for example, has non-negative curvature everywhere, so if the training set consists of all convex shapes, the level sets of the surface should stay convex throughout the estimation. The fact that a polygon has bi-modal curvature (0 and

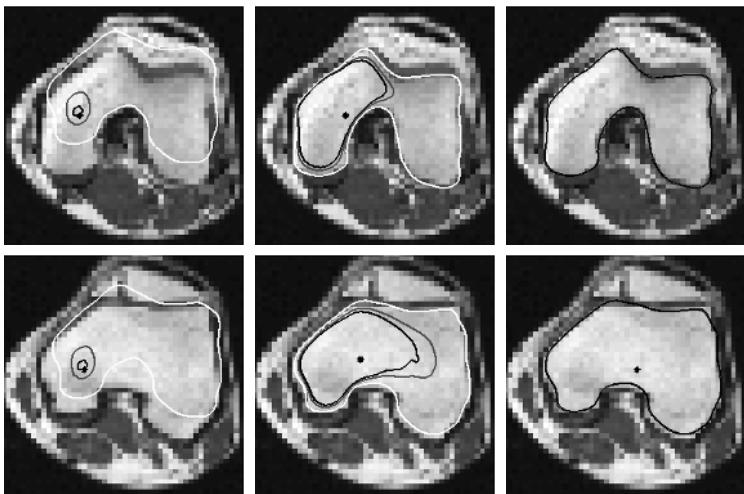


Figure 21.11. Initial, middle, and final steps in the evolution process of segmenting two slices of the femur. The training set consisted of 18 slices of the same femur, leaving out the slice being segmented and its neighbors.

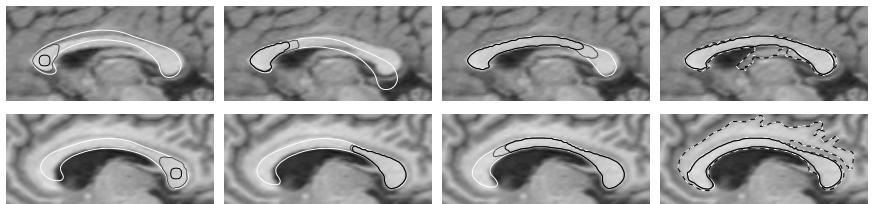


Figure 21.12. Four steps in the segmentation of two different corpora callosa. The last image in each case shows the final segmentation in black. The dotted contour is the standard ***without*** the shape influence.

∞) could be used to define a polygon-finder, if the surface can be constrained to match the distribution of curvatures of the training set. For each training surface, U_j , we compute a curvature map κ_j using Equation 21.37. We then define a PDF which encourages the curvature as reflected in the discrete differences ($U_{t+} + U_{t-} - 2U_x$) to be consistent with curvatures observed in the training data ($\kappa_j(x)$). The PDF is derived using Parzen windowing, similar to the intensity PDF defined in Section 21.4.1.

$$P(U_{t+}, U_{t-} | U_x) = \frac{1}{Z_2} \sum_{j=1}^n \sum_{x \in \mathcal{X}} \exp \left(-\frac{\| (U_{t+} + U_{t-} - 2U_x) - \kappa_j(x) \|^2}{2\sigma_2^2} \right) \quad (21.42)$$

The third row of Figure 21.10 shows the curvature profile of training sets of various objects.

In summary, the regularization of the surface is broken down into a regularization in the local normal and tangent directions. The second derivative in the normal direction is modeled as a zero mean, low variance Gaussian to keep the surface linear in that direction. A distribution over the second derivative in the tangent direction (e.g. the curvature of the level sets) is derived from the training data and used as an object-specific curvature regularization term.

21.4.3 Surface Estimation

In the previous section, we defined a model of the embedding surface of the boundary that we wish to estimate given a novel image. Starting with an initial closed curve, we build the higher dimensional surface by computing the signed distance to the curve at every point. Given the prior model and the image, the height of the surface at location \mathbf{x} is related to the image intensity at \mathbf{x} and the local neighborhood of the surface by the following equation:

$$P(U_{\mathbf{x}} | I_{\mathbf{x}}, U_{\mathcal{N}(\mathbf{x})}) \propto \underbrace{P(I_{\mathbf{x}}, U_{\mathbf{x}})}_{\text{Image Term}} \underbrace{P(U_{\hat{\ell}+}, U_{\hat{\ell}-} | U_{\mathbf{x}})}_{\text{Curvature Term}} \underbrace{P(U_{\hat{n}+}, U_{\hat{n}-} | U_{\mathbf{x}})}_{\text{Linearity Term}} \quad (21.43)$$

We use this equation to re-estimate each surface point independently, maximizing its log probability while assuming the rest of the surface is constant.

$$U_{\mathbf{x}} = \max_{U_{\mathbf{x}}} \log P(U_{\mathbf{x}} | I_{\mathbf{x}}, U_{\mathcal{N}(\mathbf{x})}) \quad (21.44)$$

Instead of finding the global maximum of the PDF for each $U_{\mathbf{x}}$, we adjust $U_{\mathbf{x}}$ in the direction of increasing probability towards the local maximum by differentiating the log probability in Equation 21.43.

$$U_{\mathbf{x}} = U_{\mathbf{x}} + \lambda \left(\frac{d}{dU_{\mathbf{x}}} \log P(U_{\mathbf{x}} | I_{\mathbf{x}}, U_{\mathcal{N}(\mathbf{x})}) \right) \quad (21.45)$$

This update is repeated until there is little change in the surface. While small, local oscillations in the surface can occur if the step size λ is too high, in practice, an empirical value of λ can easily be found such that the surface evolves in reasonable time without oscillations.

The linearity regularization term that acts on the surface in the direction normal to the level set keeps the gradient magnitude locally constant, but in general the surface does not remain a true distance function. We therefore extract the boundary and reinitialize the distance map when the gradient magnitudes stray from 1. The regularization term greatly reduces the need to reinitialize, but it does not eliminate it. Typically reinitialization occurs once or twice during the segmentation, or about every 50 iterations.

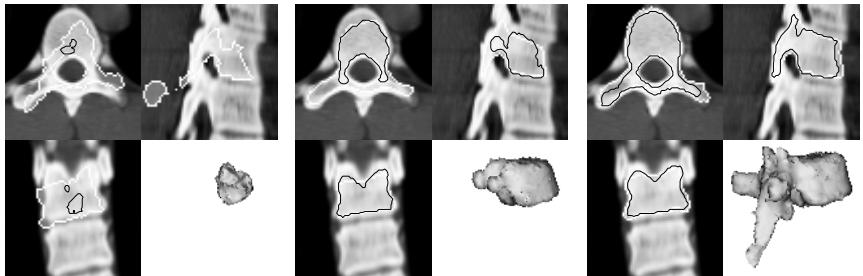


Figure 21.13. Early, middle, and final steps in the segmentation of the vertebra T7. Three orthogonal slices and the 3D reconstruction are shown for each step. The black contour is a slice through the evolving surface. The white contour is a slice through the inside of the MAP final surface.

K	Corpus 1	Corpus 2	Vertebra
95%	1.3 mm	1.5 mm	2.7 mm
99%	1.6 mm	2.0 mm	4.4 mm

Table 21.1. Partial Hausdorff distance between our segmentation and the manually-segmented ground truth.

21.5 Results

Segmentation experiments were performed on 2D slices of MR images of the femur and corpus callosum (Figures 21.11 and 21.12). For the femur experiments, the training set consisted of 18 nearby slices of the same femur, leaving out the slice being segmented and its neighbors. In both femur examples, the same initialization point was used to seed the evolution. As the curve evolves, the MAP estimator of shape and pose locks into the shape of the femur slice.

The corpus callosum training set consisted of 49 examples like those in Figure 21.1. The segmentations of two corpora callosa are shown in Figure 21.12. Notice that while the MAP shape estimator is initially incorrect, as the curve evolves, the pose and shape parameters converge on the boundary.

The vertebrae example illustrates the extension of the algorithm to 3D datasets. Figure 21.13 illustrates a few steps in the segmentation of vertebra T7. The training set in this case consisted of vertebrae T3-T9, with the exception of T7. The initial surface was a small sphere placed in the body of the vertebra. The black contour is a slice through the zero level set of the evolving hyper-surface. The white contour is the MAP pose and shape estimate. Segmenting the vertebra took approximately six minutes.

To validate the segmentation results, we compute the undirected partial Hausdorff distance [249] between the boundary of the computed segmentation and the boundary of the manually-segmented ground truth. The directed partial Hausdorff

distance over two point sets A and B is defined as

$$h_K(A, B) = \underset{a \in A}{\text{K}^{\text{th}}} \min_{b \in B} \|a - b\|$$

where K is a quantile of the maximum distance. The undirected partial Hausdorff distance is defined as $H_K(A, B) = \max(h_K(A, B), h_K(B, A))$. The results for the corpora and vertebrae shown in Table 21.1 indicate that virtually all the boundary points lie within one or two voxels of the manual segmentation.

21.6 Conclusions

This work presents a means of incorporating prior knowledge into the geodesic active contour method of medical image segmentation. The shape representation of using PCA on the signed distance map was chosen with the intention of being robust to slight misalignments without requiring exact point-wise correspondences. Our extension to active contours that estimates the model parameters and then evolves the curve based on that estimation provides a method of robustly converging on the boundary even with noisy inputs. The representation and the curve evolution technique merge well together since the evolution requires a distance map of the evolving curve, which is inherent in the shape model. The intensity and curvature models provide a means of representing the object's appearance and shape to further direct the segmentation process.

Topology Preserving Geometric Deformable Models for Brain Reconstruction

Xiao Han, Chenyang Xu and Jerry Prince

Abstract

Geometric deformable models, implemented via level-set methods, have advantages over parametric models due to their intrinsic behavior, parameterization independence, topological flexibility, lack of self-intersections, and good numerical stability. But topological flexibility actually hinders the application of geometric deformable models in cases where the model must conform to the known topology of the final object. In this chapter, we present a new geometric deformable model that preserves topology using the simple point concept from digital topology. The new model, which we refer to as topology preserving geometric deformable model (TGDM), conforms to the topology constraint while maintaining other desirable characteristics of standard geometric deformable models including sub-pixel accuracy and production of non-intersecting curves (or surfaces). We then use TGDM to find the inner, central, and outer surfaces of the human brain cortex from magnetic resonance (MR) images. The resulting algorithm is fast and numerically stable, and yields accurate brain surface reconstructions that are guaranteed to be topologically correct and free from self-intersections.

22.1 Introduction

Deformable models, are curves or surfaces that deform within two-dimensional (2D) or three-dimensional (3D) digital images under the influence of both internal and external forces and user defined constraints. Ever since their introduction by Kass et al. [262], these algorithms have been at the heart of one of the most active and successful research areas in edge detection, image segmentation, shape modeling, and visual tracking. Deformable models are broadly classified as

either parametric deformable models (cf., [262, 127, 585]) or geometric deformable models (cf., [81, 331, 85, 590, 482]) according to their representation and implementation. In particular, parametric deformable models are represented *explicitly* as parameterized contours¹ and implemented in a Lagrangian framework. Geometric deformable models, on the other hand, are represented implicitly as the zero level set of higher-dimensional level set functions and evolve according to an Eulerian formulation known as the level set method [401, 476]. Let $I(\mathbf{x}) : U \rightarrow \mathcal{R}^+$ be a given image, where $U \subset \mathcal{R}^2$ in 2D and $U \subset \mathcal{R}^3$ in 3D. In geometric deformable models, the evolving contours are embedded as the zero level set of a higher dimensional level set function $\Phi(\mathbf{x}, t) : U \times \mathcal{R}^+ \rightarrow \mathcal{R}$, and propagate implicitly through the temporal evolution of $\Phi(\mathbf{x}, t)$. By convention, $\Phi(\cdot, t)$ is a signed distance function to the contour at each time instant with negative value inside the contour and positive outside. Such a level set function can be computed efficiently from an initial contour using the *fast marching* method [533, 473, 476].

Many different forms of geometric deformable models exist in the literature (e.g. [81, 331, 84, 85, 87, 270, 271, 510, 482, 592, 412]). One of the most famous is the *geodesic active contour model* proposed independently by Caselles et al [84, 85, 87] and Kichenassamy et al [270, 271]. It can be described by the following level set evolution equation:

$$\frac{\partial \Phi(\mathbf{x}, t)}{\partial t} = g(\mathbf{x})\kappa(\mathbf{x}, t)|\nabla\Phi(\mathbf{x}, t)| + cg(\mathbf{x})|\nabla\Phi(\mathbf{x}, t)| + \nabla g(\mathbf{x}) \cdot \nabla\Phi(\mathbf{x}, t), \quad (22.1)$$

where ∇ is the gradient operator and $\kappa(\mathbf{x}, t)$ is the (mean) curvature of the level set of $\Phi(\cdot, t)$ that passes through \mathbf{x} . The curvature $\kappa(\mathbf{x}, t)$ can be computed directly from the spatial derivatives of Φ . In this model, c is a constant and $g(\cdot)$ is an image-derived metric, which is often a monotonically decreasing function of the gradient magnitude of the image I .

In spite of the large variety of geometric deformable models, we can summarize their evolution equations in the following general form [476, 586]:

$$\begin{aligned} \frac{\partial \Phi(\mathbf{x}, t)}{\partial t} = & F_{\text{prop}}(\mathbf{x}, t)|\nabla\Phi(\mathbf{x}, t)| \\ & + F_{\text{curv}}(\mathbf{x}, t)|\nabla\Phi(\mathbf{x}, t)| + F_{\text{adv}}(\mathbf{x}, t) \cdot \nabla\Phi(\mathbf{x}, t), \end{aligned} \quad (22.2)$$

where F_{prop} , F_{curv} , and F_{adv} are speed terms (sometimes called “forces”) that can be spatially varying. In particular, F_{prop} is an expansion or contraction force, F_{curv} is the part of the force that depends on the intrinsic geometry, especially the curvature of the contour and/or its derivatives, and F_{adv} is an advection force that passively transports the contour.

Eq. (22.2) yields the geodesic active contour model when $F_{\text{prop}}(\mathbf{x}, t) = cg(\mathbf{x})$, $F_{\text{curv}}(\mathbf{x}, t) = \kappa(\mathbf{x}, t)g(\mathbf{x})$, and $F_{\text{adv}}(\mathbf{x}, t) = \nabla g(\mathbf{x})$. In this case, the model arises as the result of a gradient descent minimization of an energy functional

¹In this chapter, “contour” refers to either a “curve” or a “surface.”

(because that is how the geodesic active contour model was derived). In general, however, one can choose a different form for each speed term in (22.2) to yield a different behavior of the model. As an example, we can choose $F_{\text{prop}}(\mathbf{x}, t) = R(\mathbf{x})$ to be a region force² (cf. [412, 586]) or a binary flow force [592], $F_{\text{curv}}(\mathbf{x}, t)$ to be proportional to the (mean) curvature $\kappa(\mathbf{x}, t)$, and $F_{\text{adv}}(\mathbf{x}, t) = \mathbf{v}(\mathbf{x})$ to be a gradient vector flow force [585]. With these choices the evolution equation becomes

$$\frac{\partial \Phi(\mathbf{x}, t)}{\partial t} = \omega_R R(\mathbf{x}) |\nabla \Phi(\mathbf{x}, t)| + \omega_\kappa \kappa(\mathbf{x}, t) |\nabla \Phi(\mathbf{x}, t)| + \omega_v \mathbf{v}(\mathbf{x}) \cdot \nabla \Phi(\mathbf{x}, t), \quad (22.3)$$

where ω_R , ω_κ , and ω_v are weights for the respective forces. For a binary-valued image I having values zero or one, it is convenient to define $R(\mathbf{x}) = 2I(\mathbf{x}) - 1$ to provide an expansion force inside the object and a contraction force outside. The model in (22.3) is used in our brain cortical surface reconstruction algorithm.

The numerical solution of Eq. (22.2) [or (22.3)] can be obtained by approximating the time derivative by a forward difference and the spatial derivatives by upwind numerical schemes (for details see [401, 476]), which gives:

$$\Phi(\mathbf{x}_i, t_{m+1}) = \Phi(\mathbf{x}_i, t_m) + \Delta t \Delta \Phi(\mathbf{x}_i, t_m), \quad (22.4)$$

where \mathbf{x}_i , $i = 1, 2, \dots$, denotes the discrete grid points, t_m , $m = 0, 1, \dots$, denotes the discrete time steps, and $\Delta t = t_{m+1} - t_m$ is the time-step size. Since we are interested in a generic geometric deformable model, we use $\Delta \Phi$ to denote the upwind finite difference approximation to the generic right hand side of (22.2) (see [476] for an explicit formula). Then, at each time step $t_{m+1} = (m+1)\Delta t$, we update the value of the level set function $\Phi(\cdot, t_{m+1})$ at each grid point \mathbf{x}_i from its previous values $\Phi(\cdot, t_m)$, until convergence or after a user specified number of time steps.

In this framework, the updating of the level set function Φ is performed on fixed grid points; thus, no parameterization of the deforming contour is needed. The explicit parametric representation need only be computed after the evolution is completed by taking the zero level set of Φ at the last time step, which requires an isocontour algorithm. For efficiency, the *narrow-band method* can be used to update the level set function only at a small subset of points in the neighborhood of the zero level set instead of at all the points in the computational domain [118, 2, 476]. This scheme requires recomputing the level set function after a certain number of time steps since the zero level set might move out of the updating region. It is well known that the topology of the embedded contour can change during the evolution of the level set function Φ . This means that the topology of the final contour is unpredictable, and this is the main problem we address in this work.

²Also known as a signed pressure force.

22.1.1 Digital topology

A 2D (resp. 3D) binary image $V \subset \mathbb{Z}^2$ (resp. \mathbb{Z}^3) is defined as a square (resp. cubic) array of lattice points. We follow the conventional definition of *n*-neighborhood and *n*-connectivity, where $n \in \{4, 8\}$ in 2D and $n \in \{6, 18, 26\}$ for a 3D image [291]. We denote the *n*-neighborhood of a point x by $N_n(x)$, and the set comprising the neighborhood of x with x removed by $N_n^*(x)$. The set of all *n*-connected components of $X \subset V$ is denoted by $\mathcal{C}_n(X)$.

In order to avoid a connectivity paradox, different connectivities, n and \bar{n} , must be used in a binary image comprising an object (foreground) X and a background \bar{X} . For example, in 2D, if n is chosen to be 4, then \bar{n} must be 8, and vice versa. In 3D, (18, 6) and (26, 6) are the two pairs of compatible connectivities. The following definitions are from [43] and [44].

Definition 1 (Geodesic Neighborhood) Let $X \subset V$ and $x \in V$. The geodesic neighborhood of x with respect to X of order k is the set $N_n^k(x, X)$ defined recursively by: $N_n^1(x, X) = N_n^*(x) \cap X$ and $N_n^k(x, X) = \cup\{N_n(y) \cap N_M^{k-1}(x) \cap X, y \in N_n^{k-1}(x, X)\}$, where $M = 8$ in 2D and $M = 26$ in 3D.

Definition 2 (Topological Numbers) Let $X \subset V$ and $x \in V$. The topological numbers of the point x relative to the set X are: $T_4(x, X) = \#\mathcal{C}_4(N_4^2(x, X))$ and $T_8(x, X) = \#\mathcal{C}_8(N_8^1(x, X))$ in 2D; and $T_6(x, X) = \#\mathcal{C}_6(N_6^2(x, X))$, $T_{6+}(x, X) = \#\mathcal{C}_6(N_6^3(x, X))$, $T_{18}(x, X) = \#\mathcal{C}_{18}(N_{18}^2(x, X))$, and $T_{26}(x, X) = \#\mathcal{C}_{26}(N_{26}^1(x, X))$ in 3D, where $\#$ denotes the cardinality of a set.

We note that in the above definition of topological numbers in the 3D case, there are two notations for 6-connectivity. This follows the convention introduced in [43], wherein the notation “6+” implies 6-connectivity whose dual connectivity is 18, while the notation “6” implies 6-connectivity whose dual connectivity is 26. This distinction is needed in order to correctly compute topological numbers under 6-connectivity, and does not imply a different definition of connectivity.

Topological numbers are used to classify the topology type of a grid point, especially for the characterization of *simple* points. A point is *simple* if its addition to or removal from the object does not change the topology of either the object or the background, in other words, it does not change the number of connected components, the number of cavities, and the number of handles of both the foreground and the background. It is proven in [43] that a point x is *simple* if and only if $T_n(x, X) = 1$ and $T_{\bar{n}}(x, \bar{X}) = 1$, where (n, \bar{n}) is a pair of compatible connectivities.

22.1.2 Isocontour algorithms

The design of a geometric deformable model is not complete without studying the isocontour algorithm that produces an explicit representation of the final contour from the embedding level set function. The choice of a suitable isocontour algorithm is especially critical for our topology-preserving framework where the algorithm must faithfully recover the topology of the implicit contour from the discrete samples of the level set function. In the following, we will focus on the

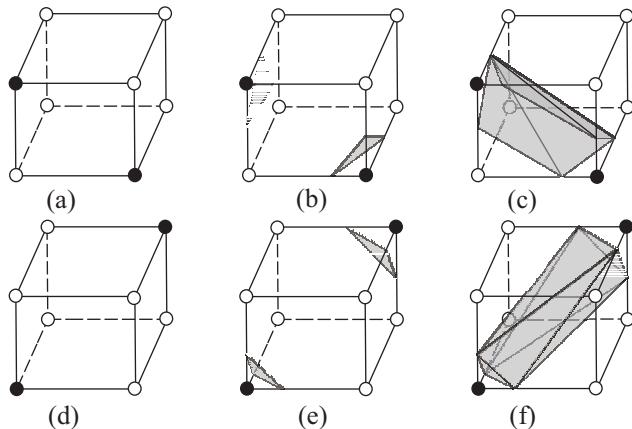


Figure 22.1. (a) An ambiguous face; (b) and (c) are two possible tilings. (d) An ambiguous cube; (e) and (f) are two possible tilings.

marching cubes isosurface algorithm, and show a modification that is necessary to make it consistent with our topology preservation principle.

The *marching cubes* (MC) algorithm is a standard isosurface algorithm that produces a triangulated surface whose vertices lie on the edges of the cubic lattice [320]. As shown in Fig. 22.1, the way in which an isosurface intersects a cube is not always unique, which results in the so-called *ambiguous face* and *ambiguous cube* cases. The major difference between different MC algorithms lies in how they choose between the two possible tilings for the ambiguous cases. A well-accepted criterion is that the surface tiling should correctly reflect the topology of the true implicit surface. Under the assumption that the embedding function is densely sampled so that it is approximately linear on each cube, *face saddle points* and *body saddle points* can be used to produce isosurfaces that are topologically consistent with the embedded implicit surfaces [390]. We note that the saddle points are the critical points of the embedding function — that is, the points where the first order derivatives of the function are all zero.

In this work, we need an isosurface algorithm that can correctly represent the topology of a binary object under the prescribed digital connectivity rule. For this purpose, we proposed the use of a *connectivity consistent* MC (CCMC) algorithm [226]. In this algorithm, the coordinates of surface intersections are still computed using linear interpolation, but the resolution of ambiguous faces and cubes now depends on the pre-defined digital connectivity rule. In particular, we choose the tilings in Figs. 22.1(c) and 22.1(e) for the corresponding ambiguous cases respectively if the black points are assumed to be 18-connected while the white points are 6-connected. If the black points are 26-connected, then Figs. 22.1(c) and 22.1(f) should be used instead.

The corresponding isocontour algorithm in 2D can be called the *connectivity consistent marching squares* (CCMS) algorithm. The only ambiguous case that

needs special care is an ambiguous square (e.g., the front face of the cube in Fig. 22.1(a)). The correct tiling should separate the white points while connect the black ones if the black points are 8-connected, and vice versa.

22.2 Topology Preserving Geometric Deformable Model

22.2.1 Algorithm

Our topology preserving mechanism exploits the binary nature of the object that is delineated by the level set function. In this framework, topology changes at the zero level set are directly related to the sign changes of the level set function. A constraint can then be imposed to keep the topology unchanged while the implicit contour deforms. The resulting deformable model behaves exactly as the unconstrained model except at places where topology changes can occur. In particular, the new model still deforms continuously, and sub-pixel accuracy is maintained.

Assume that the implicit contour is embedded as the zero level set of a level set function. Then at a given time step each grid point is either *inside* or *outside* the contour depending on the sign of the level set function at that point. We arbitrarily assign points with zero distance to be inside the contour. In this way, the implicit contour defines a unique digital object on the computational grid lattice, which consists of all the *inside* points. We can then relate the topology change of the implicit contour to that of the digital object. The topology of the digital object can only change if the inside/outside status at a point is changed or, equivalently, if the level set function changes sign at a grid point. Therefore, to preserve the topology of the object and hence the topology of the implicit contour, the level set function can only be allowed to change sign at simple points. This is the principle of the topology preserving constraint that we impose. We now give a detailed description of its implementation.

We start with the narrow band implementation of geometric deformable models since it is computationally fast, but modify it to conform to the topology constraint. In the following implementation, it is necessary to store a binary-valued indicator function B , defined on the digital grid, to record the dynamically changing digital object. For a grid point \mathbf{x}_i , $B(\mathbf{x}_i)$ equals 1 if $\Phi(\mathbf{x}_i, t_m) \leq 0$, and equals 0 otherwise, where t_m is the last time the point \mathbf{x}_i is visited. The array $B(\cdot)$ is initialized using $\Phi(\cdot, 0)$, and is updated whenever the level set function Φ undergoes a sign change at a grid point \mathbf{x}_i . The algorithm is summarized below, where \mathbf{x}_i is used to denote a general grid point and \mathbf{y}_i denotes a narrow band point.

Algorithm 1 (Topology Preserving Narrow Band Algorithm)

1. *Initialize* — Set $m = 0$ and $t_m = 0$. Initialize $\Phi(\cdot, t_0)$ to be the signed distance function of the initial contour. Initialize the binary indicator array B .

2. *Build the Narrow Band* — Find all grid points $\mathbf{y}_i, i \in \{1, \dots, Q\}$ such that $|\Phi(\mathbf{y}_i, t_m)| < W_{\text{nb}}$, where W_{nb} is the user-specified narrow band width, and Q denotes the total number of narrow band points.
3. *Update* — For $i = 1, \dots, Q$, compute the level set function at the narrow band point \mathbf{y}_i at time $t_{m+1} = t_m + \Delta t$ by:
 - (a) Using (22.4), compute $\Phi_{\text{temp}}(\mathbf{y}_i) = \Phi(\mathbf{y}_i, t_m) + \Delta t \Delta \Phi(\mathbf{y}_i, t_m)$.
 - (b) If $\text{sign}(\Phi_{\text{temp}}(\mathbf{y}_i)) = \text{sign}(\Phi(\mathbf{y}_i, t_m))$, then set $\Phi(\mathbf{y}_i, t_{m+1}) = \Phi_{\text{temp}}(\mathbf{y}_i)$, keep $B(\mathbf{y}_i)$ unchanged, and go to Step 3(f). Otherwise continue to Step 3(c).
 - (c) Compute the topological numbers $T_n(\mathbf{y}_i, X)$ and $T_{\bar{n}}(\mathbf{y}_i, \bar{X})$, where (n, \bar{n}) is the chosen digital connectivity pair, $X = \{\mathbf{x}_i | B(\mathbf{x}_i) = 1\}$, and $\bar{X} = \{\mathbf{x}_i | B(\mathbf{x}_i) = 0\}$.
 - (d) If the point is simple — i.e., $T_n(\mathbf{y}_i, X) = T_{\bar{n}}(\mathbf{y}_i, \bar{X}) = 1$ — then set $\Phi(\mathbf{y}_i, t_{m+1}) = \Phi_{\text{temp}}(\mathbf{y}_i)$, $B(\mathbf{y}_i) := (B(\mathbf{y}_i) + 1) \bmod 2$, and go to Step 3(f). Otherwise continue to Step 3(e).
 - (e) Point \mathbf{y}_i is not simple. To preserve the topology, we do not allow the sign change and set $\Phi(\mathbf{y}_i, t_{m+1}) = \epsilon \cdot \text{sign}(\Phi(\mathbf{y}_i, t_m))$, where ϵ is a small positive number used to avoid the arbitrariness of a zero value. Also, B does not change.
 - (f) Increase i . If $i > Q$, go to Step 4.
4. *Reinitialize* — If the zero level set of $\Phi(\cdot, t_{m+1})$ is near the boundary of the current narrow band, reinitialize $\Phi(\cdot, t_{m+1})$ to be the signed distance function of its own zero level set.
5. *Convergence Test* — Test whether the zero level set stops movement. If yes, stop; otherwise, set $m = m + 1$. If reinitialization is performed in Step 4, then go back to Step 2 to rebuild the narrow band; otherwise, go back to Step 3. ■

Note that in the above algorithm, the *sign* function $\text{sign}(\cdot)$ returns a value of -1 if its argument is zero, which reflects our convention that a zero-valued grid point belongs to the interior of the zero level set.

Compared with a standard narrow band algorithm [118, 2], our new algorithm differs only in the updating step, which performs a simple point criterion check whenever the level set function is going to change sign at a grid point. The sign change is prohibited if the point is not a simple point, and the evolution of the level set function at that point is truncated. We want to point out that there can be some arbitrariness in the specific result of this algorithm depending on the order in which the points are visited in the narrow band. This situation is well known in skeletonization algorithms where the result depends on the order of simple point removal. Currently, we do not have a criterion to prefer one ordering over another one; but, we feel that this problem is not as significant as in skeletonization since the overall motion of the deforming contour is controlled by the speed (force) terms. The simple point criterion only takes place at locations where topological changes are bound to occur otherwise, which is ordinarily a very small portion of

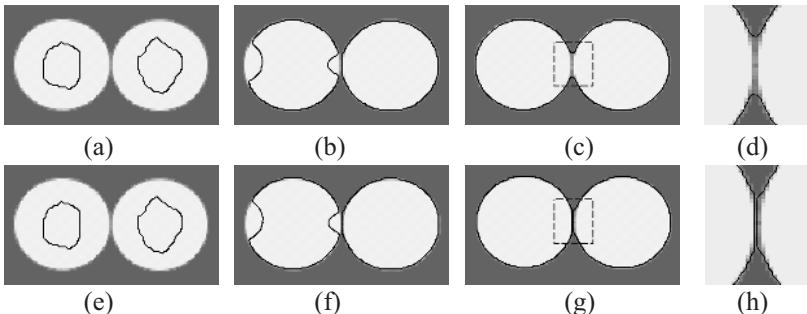


Figure 22.2. A 2D image and boundary detection using both SGDM and TGDM (see text for details).

the overall contour. In the results reported in this chapter, we followed exactly the same ordering as in the standard narrow band implementation, where points are ordered by their natural coordinates.

After the level set iterations have converged, we extract the final contour using the CCMC (or CCMS) algorithm. In these algorithms, the contour location is computed by linear interpolation of the level set function, but the tiling for the ambiguous cases is selected based on the chosen digital connectivity pair. If the level set function value is exactly zero at a grid point, it is explicitly changed to $-\epsilon$ before interpolation to prevent a singularity in the resulting explicit contour. This prevents the anomalous “touching” or “point sharing” artifact that is produced by most other isocontour algorithms and has a negligible effect on the accuracy of the resulting contour.

22.2.2 Examples

In this section, we present several examples that apply our new topology preserving geometric deformable model in 2D and 3D. Since a new model can be obtained by imposing the topology preservation constraint on an existing geometric deformable model (GDM), we refer to a *standard* model without the topology constraint as an SGDM and the corresponding (i.e., with the same set of speed terms) *topology* preserving model as a TGDM. Note that for TGDM’s, the CCMC or CCMS algorithm must be used in order to correctly extract the final curves or surfaces from the level set function, while the SGDM’s require a standard isocontour algorithm (preferably using face saddle points in 2D and both face and body saddle points in 3D). In these examples, we chose $(n, \bar{n}) = (4, 8)$ as the pair of 2D digital connectivities and $(n, \bar{n}) = (18, 6)$ for 3D.

Fig. 22.2 shows a 2D example that illustrates the topology preservation ability of the TGDM model. Here, the underlying geometric deformable model was the geodesic active contour model of Eq. 22.1. Fig. 22.2(a) (and 22.2(e)) shows the original image (65×140 pixels) consisting of two circular cells placed side-by-side. The two initial curves are also shown in this figure. Figs. 22.2(b) and 22.2(c)

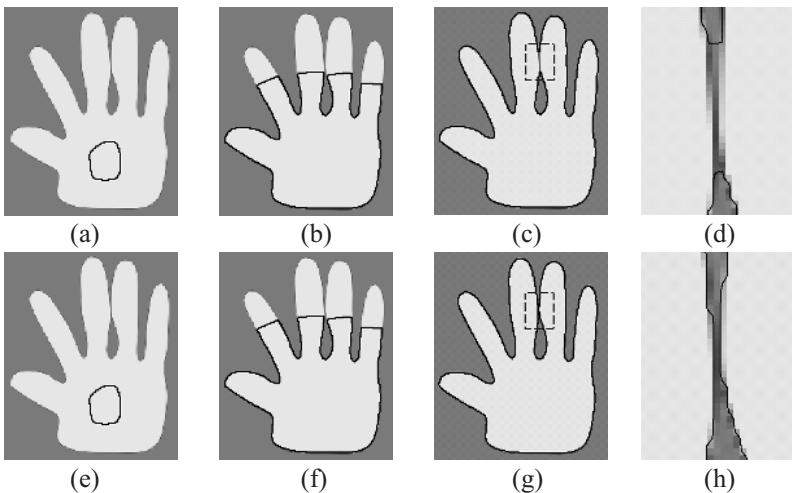


Figure 22.3. Segmentation of a hand phantom using both SGDM and TGDM (see text for details).

show the location of the implicit curves of the SGDM at an intermediate and the final stage. Because of the weak edge between the two cells, the two initial curves are merged into one final curve — an undesirable result in this example. Figs. 22.2(f) and 22.2(g) show the corresponding deformations when the topology preservation constraint is enforced (and otherwise the two models are identical in internal and external force terms). We note that TGDM keeps the two curves separated throughout the evolution and also correctly finds the boundary of each cell. Figs. 22.2(d) and 22.2(h) demonstrate the sub-pixel resolution of the SGDM and the TGDM results respectively.

Fig. 22.3 shows another 2D example in which the two deformable models of the previous example were applied to find the boundary of a hand-shaped object. The original image (220×190 pixels) and the initial curve are shown in Fig. 22.3(a) and also 22.3(e). Figs. 22.3(b) and 22.3(c) illustrate the deformation of the SGDM contour at an intermediate and the final stage. Again, without the topology preservation constraint, the initial curve changes topology and gives two separate curves as the final result [a larger outer curve and a disjoint inner curve as shown in Fig. 22.3(c) and zoomed up in Fig. 22.3(d)]. We note that the two middle fingers in this hand become one “finger” with a hole in it in the final segmentation. The corresponding deformations of the TGDM contour are illustrated in Figs. 22.3(f) and 22.3(g). TGDM keeps the boundary of each finger separated, and the final contour correctly reflects the shape of the hand, as can be seen clearly in the zoomed view of Fig. 22.3(h).

Next, we applied a 3D version of the geometric deformable model of Eq. (22.3) to find the boundary surface of the 3D object depicted in Fig. 22.4(a). The object is actually a piece of white matter (WM) segmented from a magnetic resonance (MR) brain image. Due to data noise, the WM piece has a handle in it, which is

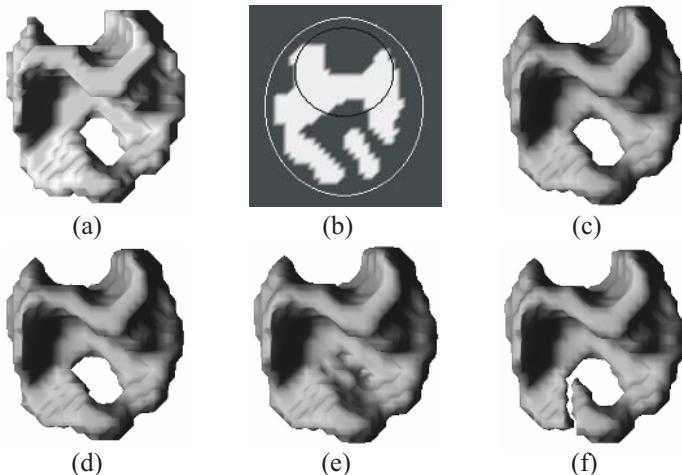


Figure 22.4. A 3D phantom and the segmentation results of using both SGDM and TGDM and two different initializations.

the wrong topology from an anatomical standpoint. In fact, we desire a topology equivalent to that of a sphere. We applied both SGDM and TGDM starting from two different initializations: a large sphere that encloses the whole object and a small ellipsoid that intersects with the object. A 2D slice showing the object and the two initial surfaces is shown in Fig. 22.4(b).

Figs. 22.4(c) and 22.4(d) are the final surfaces obtained by SGDM. The two results are the same since standard geometric deformable models are insensitive to initialization. The final surfaces both have a handle, however, which is the incorrect topology. With the sphere as the initialization, TGDM gives the final surface shown in Fig. 22.4(e), and with the ellipsoid, it gives the result shown in Fig. 22.4(f). Both surfaces have the correct topology, but the topology is preserved in different ways. The surface obtained from the sphere initialization yields a thin membrane across the tunnel (dual of the handle as viewed from the background) through the original object, while the ellipsoid initialization makes a cut in the handle.

Our final example applies SGDM, TGDM, and a parametric deformable surface model (PDM) to extract the central cortical surface from an initial fuzzy segmentation of a brain MRI image volume. (More details about cortical surface reconstruction can be found in the next section.) In this experiment, we used exactly the same initialization, the same external forces and similar internal forces for the geometric deformable models as in the parametric model. The results are presented in Fig. 22.5.

Fig. 22.5(a) shows the final surface extracted from the parametric model. The SGDM and TGDM surfaces look very similar, but on close examination there are important differences. The parametric model result, for example, has self-intersections as shown in Fig. 22.5(b), while the TGDM surface does not, as

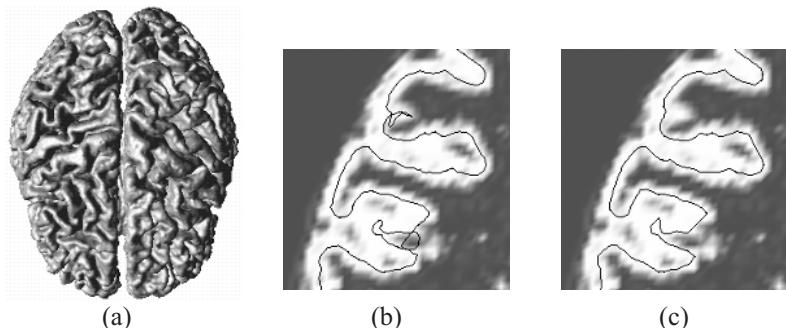


Figure 22.5. (a) Result of cortical surface reconstruction. (b) Self-intersection from PDM, and (c) no intersection with TGDM.

shown in Fig. 22.5(c). Also, the genus (number of handles) of the SGDM result is 40, while that of both the parametric model result and the TGDM result is 0. Thus, TGDM produces both the correct topology and a valid manifold; hence it is the only model that gives a legal cortical surface reconstruction.

22.3 Brain Cortical Surface Reconstruction

In this section we describe an automatic cortical surface reconstruction procedure that incorporates the topology preserving geometric deformable surface model. The method begins by finding a topologically correct representation of the WM isosurface that is close to the desired GM/WM interface. Three implementations of the new topology preserving geometric deformable surface model with well-chosen internal and external forces are then used to find three characteristic surfaces of the cortex in sequence. In the following, we first give a brief introduction of the brain cortex segmentation problem, and then present our method in detail.

22.3.1 Background

The brain cortex is a highly convoluted layer of gray matter(GM) that lies between the white matter (WM) and cerebrospinal fluid (CSF), as illustrated in Fig. 22.6. Reconstructing the surface of the cerebral cortex from magnetic resonance (MR) images is an important step in brain visualization, quantitative analysis of brain geometry, multimodal registration, surgical planning, and unfolding and mapping the cortex [146, 195]. Conventionally, the interface between GM and WM is first sought, largely because this interface is readily visualized by T1-weighted MR images. In some cases, the *central cortical surface* is sought because this surface best represents the overall cortical geometry and algorithms designed to find it tend to be robust to noise and other imaging artifacts. Despite their highly convoluted geometries, when either surface is connected across the corpus callosum

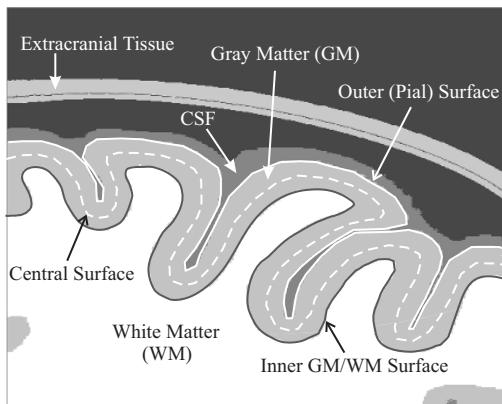


Figure 22.6. The brain cortex and its three characteristic surfaces.

and diencephalon, it has a topology equivalent to that of a sphere. Unfolding the cortex and mapping it to a sphere (or other topologically equivalent surface) can then be accomplished for visualization, measurement, and the establishment of a global coordinate system on the cortex.

Estimation of the interface between the GM and the cerebrospinal fluid (CSF), or the *pial* surface, is also an important step in the analysis of brain cortex [327, 603]. The GM/WM interface and the pial surface together establish a segmentation of the cortical gray matter, and the distance between the two surfaces measures the *cortical thickness*, which varies with cortical location.

In the literature, deformable models have been widely applied for the cortical surface reconstruction problem (e.g., [584, 146, 327, 603]). However, they typically face difficulties caused by imaging noise, image intensity inhomogeneities, the partial volume effect, and the highly convoluted nature of the brain cortex itself. For example, due to the partial volume averaging effect, opposing banks of sulci are often blurred together, which makes it difficult to reconstruct the pial interface in these areas. Also, to accurately reflect the true geometry of the cortical surface, reconstructions that connect the hemispheres across the corpus callosum and diencephalon should not self-intersect and should be topologically equivalent to a sphere [146, 327]. Although a large amount of research effort has been spent on improving the performance of deformable models in this area, it remains a difficult problem to produce surface representations of the brain cortex that are topologically correct, geometrically accurate, and non-intersecting. The method described here is a continuation of the work of [584], but improves the previous work in several aspects, which includes a better model initialization procedure and compensation for partial volume effect. A significant change was made by replacing the previous parametric deformable model by TGDM, which brings the advantage of speed, stability, and producing surfaces without self-intersections. We now describe the method in detail.

22.3.2 Preprocessing and surface initialization

The MR brain images were T1-weighted with voxel size $0.9375 \times 0.9375 \times 1.5\text{mm}^3$. The volumes were preprocessed to remove extracranial tissue, the cerebellum, and the brain stem. Each volume was then interpolated to isotropic voxels, and robustly segmented into GM, WM, and CSF fuzzy membership functions [426], which represent a partial volume segmentation of the image volume. These membership functions were used in all subsequent processing (rather than using the raw MR data) because they offer robustness to noise and the partial volume effect. The WM membership function was then automatically processed to fill the ventricles and subcortical GM structures such as the thalamus, hypothalamus, caudate nucleus, and putamen [226].

The filled WM volume was binarized with a threshold of 0.5, since a membership value of 0.5 indicates the interface between two classes. The binary volume was then processed to correct its topology as described in [225]. The 0.5-isosurface of the resulting volume would yield a topologically correct explicit representation of the WM boundary surface if computed using the CCMC algorithm. We note that both the topology correction, the CCMC algorithm, and TGDM must assume the same underlying digital connectivity rule. In all the results presented herein, we assume 18-connectivity for the foreground object (i.e., the white matter) and 6-connectivity for the background.

We then embed the topology correct WM boundary surface into a signed distance function using the fast marching method. The topology preserving deformable surface model can then be applied to get the desired cortical surfaces. The general deformable model given by Eq. (22.3) is used to find all three surfaces, but the forces are designed differently, as we now discuss.

22.3.3 Nested surface reconstruction algorithm

GM/WM surface. The first TGDM model is targeted at the inner GM/WM surface. Although the initialization step already provides a very close and topologically correct representation of the GM/WM boundary, we need to smooth out its staircase artifacts while simultaneously maintaining the accuracy of the surface and keeping the correct topology. To accomplish this, we use the curvature force and the regional signed pressure force but not the gradient vector flow force. The curvature force aims to regularize the surface, and is proportional to the mean curvature κ of the surface. The signed pressure force is defined as

$$R(x) = 2\mu'_{\text{WM}}(x) - 1,$$

where μ'_{WM} denotes the new WM membership function obtained after filling the ventricles and subcortical GM structures. Since $\mu'_{\text{WM}} \in [0, 1]$, $R(x)$ falls in the range of $[-1, 1]$. If $\mu'_{\text{WM}} > 0.5$ then $R(x) > 0$, whereas if $\mu'_{\text{WM}} < 0.5$ then $R(x) < 0$. Therefore, $R(x)$ provides outward balloon forces when the surface resides within the WM and inward forces when it resides outside the WM. Thus, $R(x)$ will force the surface to the GM/WM boundary. The weights for the two

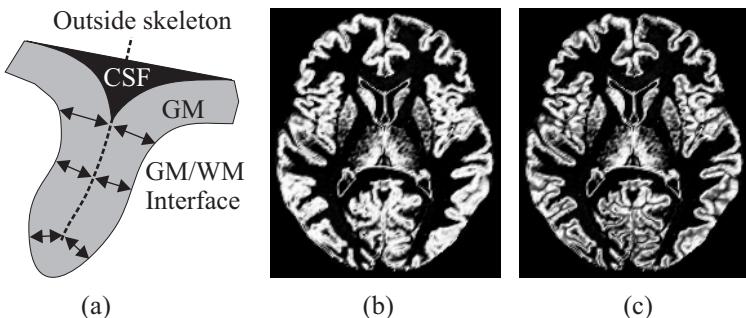


Figure 22.7. (a) A tight sulcus and the outside skeleton. (b) Original GM membership function. (c) ACE edited GM membership function.

forces are chosen to be $\omega_R = 1$ and $\omega_\kappa = -0.02$. These are presently determined empirically, and will be optimized in future work. After convergence, the final level set function, denoted by Φ_{in} , gives an implicit representation of the inner cortical surface, i.e., the GM/WM interface.

Anatomically consistent GM editing. The GM/WM interface is relatively unaffected by the partial volume effect. Reconstruction of both the central and pial surfaces, however, is very difficult when opposing sulcal banks are blurred together. Before proceeding to estimate these surfaces, we present our *anatomically consistent editing* (ACE) approach, which automatically modifies the GM membership function in tight sulci to produce evidence of CSF where it is otherwise obscured by the partial volume effect.

The first step in ACE is to detect the medial axis of the sulci, as illustrated in Fig. 22.7(a). Since the level set function of a geometric deformable model is itself a signed distance function, we can use Φ_{in} to identify points that are outside the GM/WM interface and are equidistant to two (or more) GM/WM surface patches, that is, the outer skeletal points. In particular, we extract the outside skeleton by taking the discrete Laplacian of Φ_{in} , clip negative values, set it to zero whenever Φ_{in} is negative, and then normalize the result to the range $[0, 1]$. Locations where the resulting function — which we denote by $L(x)$ — is large represent the outside skeleton.

We now use the outside skeleton function $L(x)$ to modify the original GM membership function. One problem that must be addressed is the presence of medial axes that approach the GM/WM interface. Although the normalized Laplacian decreases as it approaches the interface, we do not want to risk actually modifying the GM membership function very near its interior surface. Accordingly, modifications to the GM membership function are only made at a distance greater than 1 mm from the GM/WM interface — an arbitrary distance chosen because it represents a lower bound on cortical thickness. Following this rationale,

we produce an ACE GM membership function as follows

$$\mu'_{\text{GM}}(x) = \begin{cases} [1 - L(x)]\mu_{\text{GM}}(x) & \Phi_{\text{in}}(x) > 1\text{mm} \\ \mu_{\text{GM}}(x) & \text{otherwise} \end{cases} \quad (22.5)$$

The result of the GM editing is illustrated in Figs. 22.7(b) and 22.7(c), where a 2-D cross-sectional view is shown. Clearly, there is a marked improvement in the appearance of sulcal gaps in the ACE result as compared to the original GM membership function. We note that ACE does not make any assumptions about the maximum cortical thickness, so it will not restrict the cortical width anywhere in the cortex (cf. [327, 603]. We also note that it only has an effect within sulci, and even then only when there is actual gray matter on the outside skeleton.

Central cortical surface. The ACE GM membership function better represents the true anatomy of the brain cortex. We now use this membership function to derive forces for a second TGDM model to find the central surface of the cortex.

As shown in [584], the use of a gradient vector flow (GVF) force makes it easy to find the central layer of a thick sheet. Suppose we take the ACE GM membership function μ'_{GM} itself as an edge map, and compute the gradient vector flow from it. Then, the resulting GVF force $\mathbf{v}(\cdot)$ will point to the center of the thick “edge”, that is, the center of the GM sheet. We refer the reader to [585, 584] for details about computing the GVF force from a given edge map.

To make sure the surface does not move out of the cortex, we also apply a regional force using [584]

$$R(x) = \begin{cases} 0, & \text{if } \|2\mu'_{\text{WM}}(x) + \mu'_{\text{GM}}(x) - 1\| < 0.5; \\ 2\mu'_{\text{WM}}(x) + \mu'_{\text{GM}} - 1, & \text{otherwise.} \end{cases} \quad (22.6)$$

This region force pushes the surface outward if it is in the WM, inward if in the CSF, but has no effect within the GM. Thus, the surface moves toward the central layer of the GM under the influence of GVF forces alone. We also apply a small curvature force to keep the surface smooth. The relative strengths of the individual forces are determined by their coefficients, which we fix to be $\omega_R = \omega_{\mathbf{v}} = 1$ and $\omega_{\kappa} = -0.02$ for all the brain studies.

Using these internal and external forces, the second TGDM is applied starting from the GM/WM surface Φ_{in} . To ensure the correct relative geometry of the surfaces, we also apply a barrier constraint that prevents the central surface from staying inside the GM/WM interface (which could be otherwise caused by noise in GVF). In particular, during the narrow band update we do not allow the evolving level set function to get larger than the initialization Φ_{in} at any grid point. After convergence, the output level set function gives the signed distance from the central cortical surface, which is denoted by Φ_{central} .

Pial surface. TGDM is now used with new forces to find the pial surface, starting from the central surface obtained in the last step. Here, we use a GVF force that is designed to transport the surface towards the GM/CSF interface. We first compute a new edge map $\text{edge}_{\text{outer}}$ that has large values at the GM/CSF interface, as follows

$$\text{edge}_{\text{outer}}(\mathbf{x}) = \|\nabla(\mu'_{\text{GM}}(\mathbf{x}) + \mu'_{\text{WM}}(\mathbf{x}))\|.$$

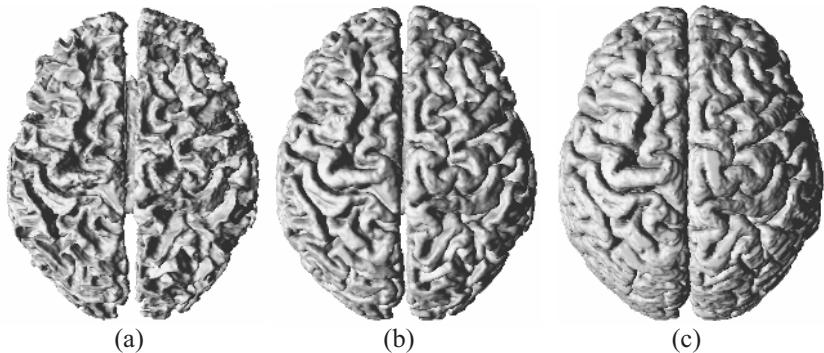


Figure 22.8. Top view of the reconstructed (a) GM/WM, (b) central, and (c) pial surfaces for one brain image.

We then compute the new GVF force $\mathbf{v}(\cdot)$ from the edge map $\text{edge}_{\text{outer}}$. To accelerate the movement of the deformable surface inside the GM and to help it stay at the GM/CSF interface, the following region force $R(x)$ is also used

$$R(x) = 2\mu'_{\text{GM}} - 1.$$

This region force makes the surface expand if inside the GM and contract if outside the GM (i.e., inside the CSF). Mean curvature is still used as the internal force.

A third TGDM is run with the above forces and the weights as before: $\omega_R = \omega_{\mathbf{v}} = 1$ and $\omega_{\kappa} = -0.02$. We also apply a barrier constraint as in the previous section to make sure the outer surface stays outside of the central surface. The converged level set function of this TGDM model, denoted by Φ_{out} , embeds the pial surface as its zero level set.

22.3.4 Results

We have applied our new cortical surface reconstruction method on 21 MR brain images obtained from the Baltimore Longitudinal Study on Aging [434]. Figs. 22.8(a)–(c) show the reconstructed GM/WM, central, and pial surfaces, respectively, for one of these data sets. Fig. 22.9 shows these surfaces overlaying various cross sections of the original MR data. From these figures, it can be seen that the estimated surfaces follow the folds of the cortex very well. It can also be seen that ACE helps the outer surface stay inside tight sulci.

To estimate the accuracy of the proposed method, we computed a set of landmark errors on six of the reconstructed central cortical surfaces. The landmarks, ten on each brain, were manually picked on several major sulci and gyri (see [584]). The landmark error was then computed as the minimum distance between the given landmarks and the reconstructed surfaces. The overall average landmark error produced is about 0.87 mm, which shows significant improvement over our previously reported 1.22 mm in [584], where a parametric deformable

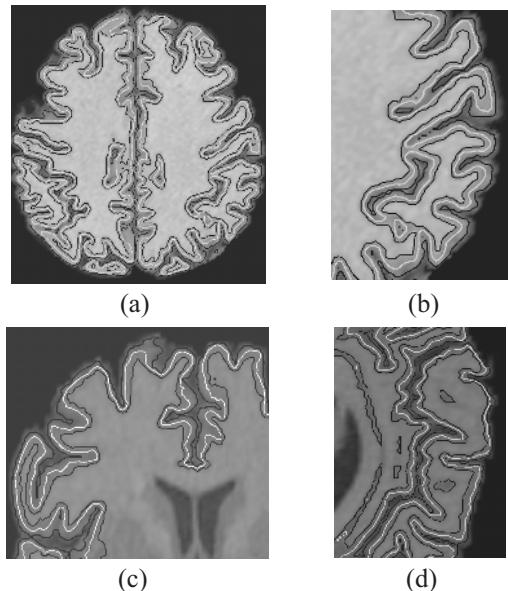


Figure 22.9. (a) Three reconstructed surfaces displayed on an axial slice of the original MR data. (b) A zoom of the bottom-right corner of (a). (c),(d) Zoomed views on coronal and sagittal slices, respectively.

surface model was used to extract the central surface and without using the ACE. The new method also exhibits substantially increased stability, as indicated by a much lower standard deviation of 0.50 mm in average as compared to the previous 1.01 mm (see [227] for details).

We also ran a program to detect self-intersections on the final surface meshes. The new results have no self-intersections anywhere, as expected. In contrast, most cortical surface reconstructions obtained from parametric deformable models that do not explicitly prohibit them will have self-intersections. This is especially true for the pial surface, where many parts of the surface are essentially back-to-back in the tight sulci. Those methods that explicitly prohibit self-intersections (cf. [146] and [327]) are very computationally intensive. Another way to try to reduce self-intersections with parametric models is to increase the internal forces, but this can have a very negative effect on the surface accuracy. Our method prevents self-intersections while simultaneously maintaining low internal forces to permit highly accurate (subvoxel) surface reconstructions.

Given the initial fuzzy segmentation, our method currently takes about 40 minutes on an SGI O2 workstation (174 MHz, R10000 processor) to reconstruct all the three surfaces. Thus, it is comparable to the geometric deformable model described in [603], and much faster than parametric models reported in the literature, even those that do not impose self-intersection detection. Given that most algo-

rithms currently report many hours to obtain similar results, we believe that our algorithm represents an important development for brain image analysis.

22.4 Conclusion

Deformable models constitute a set of powerful image segmentation tools. The two class of deformable models, parametric and geometric, both have their advantages and disadvantages with respect to specific applications. In recent years, many efforts have been conducted to overcome their shortcomings and combine their advantages. In this work, we propose a method to overcome the topology arbitrariness of standard geometric deformable models, which is critical for applications where topology correctness is of major concern. The significance of the proposed technique is demonstrated by its application in the brain cortex segmentation tasks. Future work includes reducing the sensitivity to model initialization and investigating other areas of applications.

Acknowledgments: We thank Drs. Susan Resnick and Dzung Pham for their guidance in using the brain image data. This work was supported in part by NSF/ERC grant CISST#9731748 and NIH/NINDS grant R01NS37747.

Part IX

Simulations & Graphics

Editing Geometric Models

Ken Museth, Ross Whitaker and David Breen

Abstract

We have developed a level set framework for editing closed geometric models. Level set models are deformable implicit surfaces where the deformation of the surface is controlled by a speed function in the level set partial differential equation. In this chapter we define several speed functions that provide a set of surface editing operators. The speed functions describe the velocity at each point on the evolving surface in the direction of the surface normal. All of the information needed to deform a surface is encapsulated in the speed function, providing a simple, unified computational framework. The user combines pre-defined building blocks to create the desired speed function. The surface editing operators are quickly computed and may be applied both regionally and globally. The level set framework offers several advantages. 1) By construction, self-intersection cannot occur, which guarantees the generation of physically-realizable, simple, closed surfaces. 2) Level set models easily change topological genus, and 3) are free of the edge connectivity and mesh quality problems associated with mesh models. We present five examples of surface editing operators: blending, smoothing, sharpening, openings/closings and embossing. We demonstrate their effectiveness on several scanned objects and scan-converted models.

23.1 Introduction

The creation of complex models for such applications as movie special effects, graphic arts, and computer-aided design can be a time-consuming, tedious, and error-prone process. One of the solutions to the model creation problem is 3D photography [61], i.e. scanning a 3D object directly into a digital representation. However, the scanned model is rarely in a final desired form. The scanning process is imperfect and introduces errors and artifacts, or the object itself may be flawed.

3D scans can be converted to polygonal and parametric surface meshes [174, 26, 17]. Many algorithms and systems for editing these polygonal and

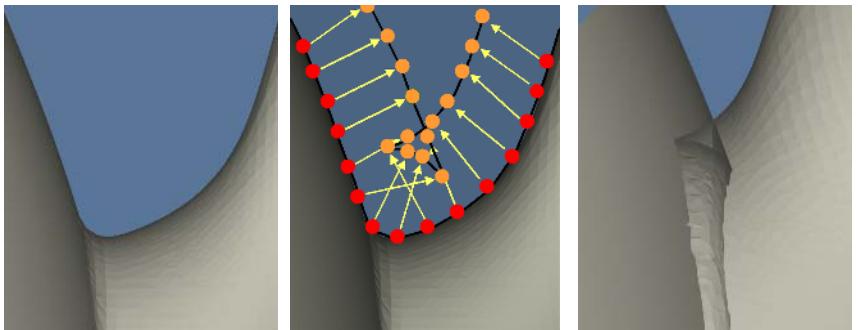


Figure 23.1. Self-intersection may occur when performing surface offsets directly on a mesh: (left) A cross-section of the Utah-teapot model near the spout. (middle) Vertices are offset in the direction of the local surface normals. (right) Self-intersections occur when vertices are re-meshed.

parametric surfaces have been developed [125], but surface mesh editing has its limitations and must address several difficult issues. For example, it is difficult to guarantee that a mesh model will not self-intersect when performing a local editing operation based on the movement of vertices or control points, producing non-physical, invalid results. See Figure 23.1. If self-intersection occurs, it must be fixed as a post-process. Also, when merging two mesh models the process of clipping individual polygons and patches may produce errors when the elements are small and/or thin, or if the elements are almost parallel. In addition while it is not impossible to change the genus of a surface mesh model [46], it is certainly difficult and requires significant effort to maintain the consistency/validity of the underlying vertex/edge connectivity structure.

23.1.1 New Surface Editing Operators

In order to overcome these difficulties we present a level set approach to implementing operators for locally and globally editing closed surfaces. Level set models are deformable implicit surfaces that have a volumetric representation [401]. They are defined as an iso-surface, i.e. a level set, of some implicit function ϕ . The surface is deformed by solving a partial differential equation (PDE) on a regular sampling of ϕ , i.e. a volume dataset. To date level set methods have not been developed for adaptive grids, a limitation of current implementations, but not of the mathematics. It should be emphasized that level set methods do not manipulate an explicit closed form representation of ϕ , but only a sampling of it. Level set methods provide the techniques needed to change the voxel values of the volume in a way that deforms the embedded iso-surface to meet a user-defined goal. The user controls the deformation of the level set surface by defining a speed function $\mathcal{F}(x, \dots)$, the speed of the level set at point x in the direction of the normal to the surface at x . Therefore all the information needed to deform a level set

model may be encapsulated in a single speed function $\mathcal{F}()$, providing a simple, unified computational framework.

We have developed a number of surface editing operators within a level set framework by defining several new level set speed functions. The cut-and-paste operator (Section 23.5.1) gives the user the ability to copy, remove and merge level set models (using volumetric CSG operations) and automatically blends the intersection regions (See Section 23.5.2). Our smoothing operator allows a user to define a region of interest and smooths the enclosed surface to a user-defined curvature value. See Section 23.5.3. We have also developed a point-attraction operator. See Section 23.5.4. Here, a regionally constrained portion of a level set surface is attracted to a single point. By defining line segments, curves, polygons, patches and 3D objects as densely sampled point sets, the single point attraction operator may be combined to produce a more general surface embossing operator. As noted by others, the opening and closing morphological operators may be implemented in a level set framework [459, 334]. We have also found them useful for performing global blending (closing) and smoothing (opening) on level set models. Since all of the operators accept and produce the same volumetric representation of closed surfaces, the operators may be applied repeatedly to produce a series of surface editing operations. See Figure 23.11.

23.1.2 Benefits and Issues

Performing surface editing operations within a level set framework provides several advantages and benefits. Many types of surfaces may be imported into the framework as a distance volume, a volume dataset that stores the signed shortest distance to the surface at each voxel. This allows a number of different types of surfaces to be modified with a single, powerful procedure. By construction, the framework always produces non-self-intersecting surfaces that represent physically-realizable objects, an important issue in computer-aided design. See Figure 23.2. Level set models easily change topological genus, and are free of the edge connectivity and mesh quality problems associated with deforming and modifying mesh models. Additionally, some reconstruction algorithms produce volumetric models [144, 573, 609] and volumetric scanning systems are increasingly being employed in a number of diverse fields. Therefore volumetric models are becoming more prevalent and there is a need to develop powerful editing operators that act on these types of models directly. There are implementation issues to be addressed when using level set models. Given their volumetric representation, one may be concerned about the amount of computation time and memory needed to process level set models. Techniques have been developed to limit level set computations to only a narrow band around the level set of interest [2, 573, 421] making the computational complexity proportional to the surface area of the model. We have also developed computational techniques that allow us to perform the narrow band calculations only in a portion of the volume where the level set is actually moving. Additionally, fast marching methods have been developed to rapidly evaluate the level set equation under certain circumstances

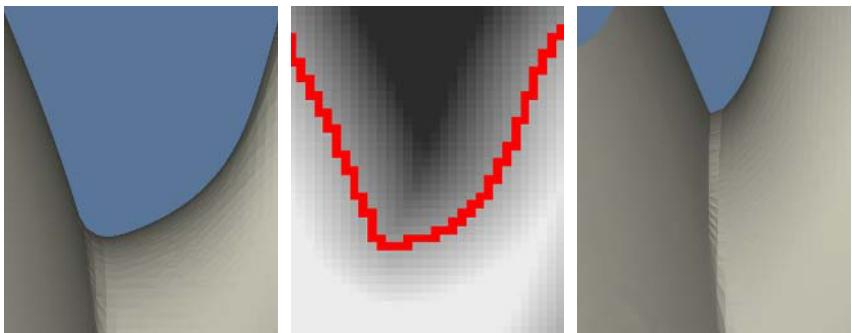


Figure 23.2. Self-intersection cannot occur when performing surface offsets in a level set framework: (left) A cross-section of the Utah-teapot model near the spout. (middle) Implicit representation of the surface as a discretely sampled level set, *i.e.* isosurface of a voxelized distance volume. (right) No self-intersection occurs, by construction, when the level set surface is offset.

[533, 473]. Memory usage has not been an issue when generating the results in this paper. The memory needed for our results (512 MB) is available on standard workstations and PCs. We have implemented our operators in an interactive environment that allows us to easily edit a number of complex surfaces. Additionally, concerns have been raised that volume-based models cannot represent fine or sharp features. Recent advances [201, 283] have shown that it is possible to model these kinds of structures with volume datasets, without excessively sampling the whole volume. These advances will also be available for our operators once adaptive level set methods, an active research area, are developed.

23.1.3 Contributions

The major contributions of our work can be summarized as follows: We introduce a unified, robust and powerful approach to surface editing within a level set framework. The editing operators are encoded by speed functions, that in turn are defined by combining relatively simple pre-defined building blocks. We present level set speed functions that implement blending, smoothing and embossing surface editing operators. Blending is automatic and is constrained to only occur within a user-specified distance to an arbitrarily complex intersection curve. Smoothing and embossing are constrained to occur within a user-specified region. For these surface editing operators the user specifies the local geometric properties of the resulting surface modifications. Additionally the user may constraint the direction of the surface movements, thus determining if material should be added and/or removed during editing operations. The corresponding speed functions also utilize a new technique for localizing the level set calculations, thereby making them interactive. Finally, in the Appendix, we present a new, numerically-stable curvature measure for level set surfaces.

23.2 Previous Work

Three areas of research are closely related to our level set surface editing work; volumetric sculpting, mesh-based surface editing/fairing and implicit modeling. Volumetric sculpting provides methods for directly manipulating the voxels of a volumetric model. CSG Boolean operations [241, 560] are commonly found in volume sculpting systems, providing a straightforward way to create complex solid objects by combining simpler primitives. One of the first volume sculpting systems is presented in [205]. [561] improved on this work by introducing tools for carving and sawing. More recently [425] implemented a volumetric sculpting system based on the Adaptive Distance Fields (ADF) [201], allowing for models with adaptive resolution.

Performing CSG operations on mesh models is a long-standing area of research [433, 301]. Recently CSG operations were developed for multi-resolution subdivision surfaces by [46], but this work did not address the problem of blending or smoothing the sharp features often produced by the operations. However, the smoothing of meshes has been studied on several occasions [572, 509, 284]. [155] have developed a method for fairing irregular meshes using diffusion and curvature flow, demonstrating that mean-curvature based flow produces the best results for smoothing.

There exists a large body of surface editing work based on implicit models [55]. This approach uses implicit surface representations of analytic primitives or skeletal offsets. The implicit modeling work most closely related to ours is found in [582]. They describe techniques for performing blending, warping and boolean operations on skeletal implicit surfaces. [154] address the converse problem of preventing unwanted blending between implicit primitives, as well as maintaining a constant volume during deformation.

Level set methods have been successfully applied in computer graphics, computer vision and visualization [476, 458], for example medical image segmentation [331, 574], shape morphing [67], 3D reconstruction [573, 609], and recently for the animation of liquids [198].

Our work stands apart from previous work in several ways. We have not developed volumetric modeling tools. Our editing operators act on surfaces that happen to have an underlying volumetric representation, but are based on the mathematics of deforming implicit surfaces. Our editing operators share several of the capabilities of mesh-based tools, but are not hampered by the difficulties of maintaining vertex/edge information. Since level set models are not tied to any specific implicit basis functions, they easily represent complex models to within the resolution of the sampling. Our work is the first to utilize level set methods to perform user-controlled editing of complex geometric models.

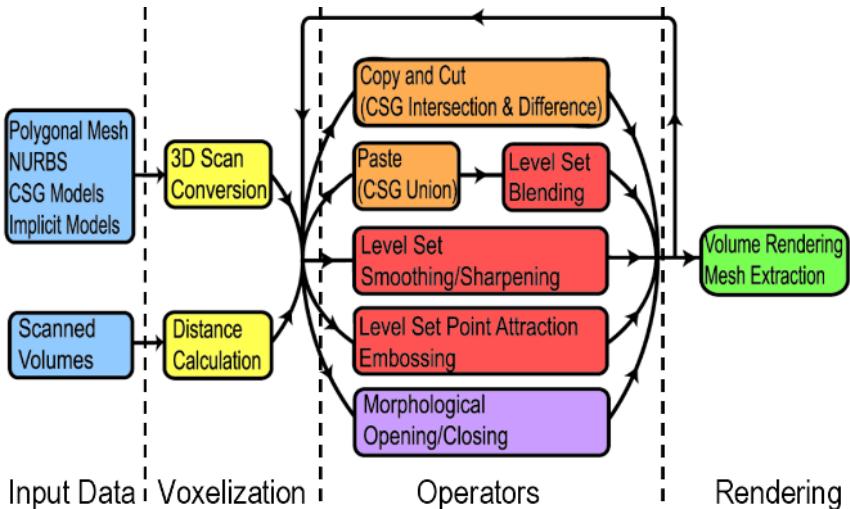


Figure 23.3. Our level set surface editing operators fit into a larger editing framework. The pipeline consists of: input models, pre-processing (voxelization), CSG operations, our new local LS operators, global morphological operators and rendering.

23.3 Overview of the Editing Pipeline

The level set surface editing operators should be viewed as components of a larger modeling framework. The pipeline for this framework is presented in Figure 23.3. The three level set components contain the new speed functions that we have developed for localized surface editing. The remaining components contain the data and operations needed for level set modeling, input models, pre-processing (voxelization), CSG operations, global morphological operators and rendering. The pipeline provides the context for the details of our speed functions.

23.3.1 Input and Output Models

We are able to import a wide variety of closed geometric models into the level set environment. We represent a level set model as an iso-surfaces embedded in a distance volume. Frequently we only store distance information in a narrow band of voxels surrounding the level set surface. As illustrated in Figure 23.3 we have developed and collected a suite of scan conversion methods for converting polyedral meshes, CSG models [66], implicit primitives, and NURBS surfaces into distance volumes. Additionally many types of scanning processes produce volumetric models directly, e.g. MRI, CT and laser range scan reconstruction. These models may be brought into our level set environment as is or with minimal pre-processing. We frequently utilize Sethian's Fast Marching Method [473] to convert volume datasets into distance volumes. In the final stage of the pipeline we can either volume render the surface directly or render a polyedral mesh ex-

tracted from the volume. While there are numerous techniques available for both approaches, we found extracting and rendering Marching Cubes meshes [320] to be satisfactory.

23.4 Level Set Surface Modeling

The Level Set Method, first presented in [401], can be considered a mathematical tool for modeling surface deformations. A deformable (*i.e.* time-dependent) surface is implicitly represented as an iso-surface of a time-varying scalar function, $\phi(\mathbf{x}, t)$. A detailed description of level set models is presented in the first chapter of this book.

23.4.1 Level Set Speed Function Building Blocks

Given the definition

$$\mathcal{F}(\mathbf{x}, \mathbf{n}, \phi, \dots) \equiv \mathbf{n} \cdot \frac{d\mathbf{x}}{dt}, \quad (23.1)$$

the fundamental level set equation can be written as

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \mathcal{F}(\mathbf{x}, \mathbf{n}, \phi, \dots) \quad (23.2)$$

where $d\mathbf{x}/dt$ and $\mathbf{n} \equiv -\nabla \phi / |\nabla \phi|$ are the velocity and normal vectors at \mathbf{x} on the surface. We assume a positive-inside/negative-outside sign convention for $\phi(\mathbf{x}, t)$, *i.e.* \mathbf{n} points outward. Eq. (23.1) introduces the speed function \mathcal{F} , which is a user-defined scalar function that can depend on any number of variables including \mathbf{x} , \mathbf{n} , ϕ and its derivatives evaluated at \mathbf{x} , as well as a variety of external input data. $\mathcal{F}()$ is a *signed* scalar function that defines the motion (*i.e.* speed) of the level set surface in the direction of the local normal \mathbf{n} at \mathbf{x} .

The speed function is usually based on a set of geometric measures of the implicit level set surface and input data. The challenge when working with level set methods is determining how to combine the building blocks to produce a local motion that creates the desired global or regional behavior of the surface. The general structure for the speed functions used in our surface editing operators is

$$\mathcal{F}(\mathbf{x}, \mathbf{n}, \phi) = \mathcal{D}_q(d)\mathcal{C}(\gamma)\mathcal{G}(\gamma) \quad (23.3)$$

where $\mathcal{D}_q(d)$ is a distance-based cut-off function which depends on a distance measure d to a geometric structure q . $\mathcal{C}(\gamma)$ is a cut-off function that controls the contribution of $\mathcal{G}(\gamma)$ to the speed function. $\mathcal{G}(\gamma)$ is a function that depends on geometric measures γ derived from the level set surface, e.g. curvature. Thus, $\mathcal{D}_q(d)$ acts as a region-of-influence function that regionally constrains the LS calculation. $\mathcal{C}(\gamma)$ is a filter of the geometric measure and $\mathcal{G}(\gamma)$ provides the geometric contribution of the level set surface. In general γ is defined as zero, first, or second order measures of the level set surface.

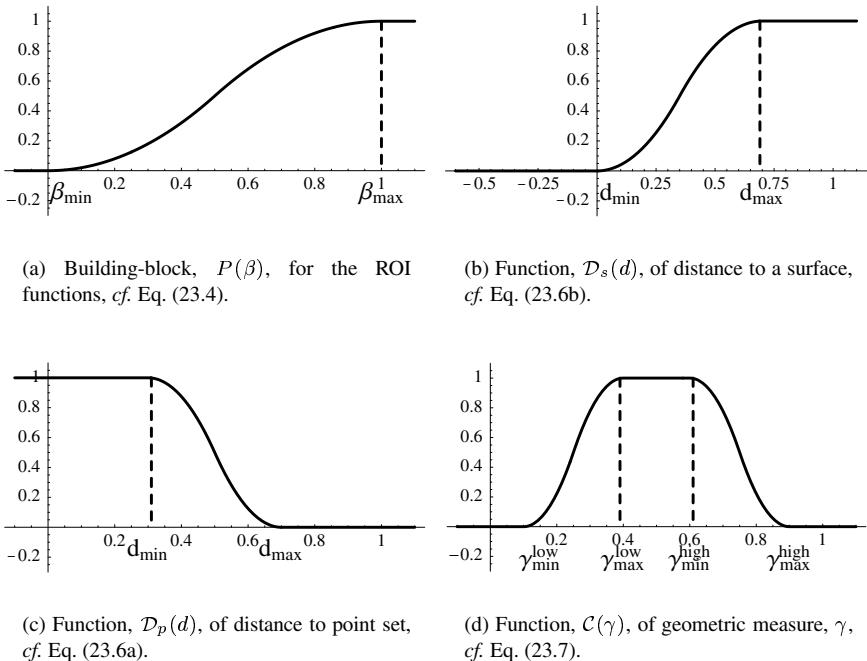


Figure 23.4. Graph of functions used to build the speed functions for our local level set operations, cf. Eq. (23.3).

23.4.2 Regionally Constraining Level Set Deformations

Most of our surface operators may be applied locally in a small user-defined region on the edited surface. In order to regionally restrict the deformation during the level set computation, a technique is needed for driving the value of $\mathcal{F}()$ to zero outside of the region. This is accomplished in three steps. The first step involves defining the region of influence (ROI), *i.e.* the region where $\mathcal{F}()$ should be non-zero. This is done by either the user interactively placing a 3D object around the region, or by automatically calculating a region from properties of the surface. Both cases involve defining a geometric structure that we refer to as a “region-of-influence (ROI) primitive”. The nature of these primitives will vary for the different operations and will explicitly be defined in Section 23.5. The second step consists of calculating a distance measure to the ROI primitive. The final step involves defining a function that smoothly approaches zero at the boundary of the ROI.

We define a region-of-influence function $D_q(d)$ in Eq. (23.3), where d is a distance measure from a point on the level set surface to the ROI primitive q . The functional behavior of $D_q(d)$ clearly depends on the specific ROI primitive, q , but we found the following piecewise polynomial function to be useful as a common

speed function building block:

$$P(\beta) = \begin{cases} 0 & \text{for } \beta \leq 0 \\ 2\beta^2 & \text{for } 0 < \beta \leq 0.5 \\ 1 - 2(\beta - 1)^2 & \text{for } 0.5 < \beta < 1 \\ 1 & \text{for } \beta \geq 1. \end{cases} \quad (23.4)$$

$P(\beta)$ and its derivatives are continuous and relatively inexpensive to compute. See Figure 23.4(a). Other continuous equations with the same basic shape would also be satisfactory. We then define

$$\mathcal{P}(d; d_{min}, d_{max}) \equiv P\left(\frac{d - d_{min}}{d_{max} - d_{min}}\right) \quad (23.5)$$

where d_{min} and d_{max} are user-defined parameters that define the limits and sharpness of the cut-off. Let us finally define the following region-of-influence functions

$$\mathcal{D}_p(d) = 1 - \mathcal{P}(d; d_{min}, d_{max}) \quad (23.6a)$$

$$\mathcal{D}_s(d) = \mathcal{P}(d; 0, d_{max}) \quad (23.6b)$$

for a point set, p , and a closed surface, s .

In Eq. (23.6a) d denotes the distance from a point on the level set surface to the closet point in the point set p . In Eq. (23.6b) d denotes a signed distance measure from a point on the level set surface to the implicit surface s . The signed distance measure does not necessarily have to be Euclidean distance - just a monotonic distance measure following the positive-inside/negative-outside convention. Note that $\mathcal{D}_p(d)$ is one when the shortest distance, d , to the point set is smaller than d_{min} , and decays smoothly to zero as d increases to d_{max} , after which it is zero. $\mathcal{D}_s(d)$, on the other hand, is zero everywhere outside, as well as on, the surface s ($d \leq 0$), but one inside when the distance measure d is larger than d_{max} .

An additional benefit of the region-of-influence functions is that they define the portion of the volume where the surface cannot move. We use this information to determine what voxels should be updated during the level set deformation, significantly lowering the amount of computation needed when performing editing operations. This technique allows our operators to be rapidly computed when modifying large models.

23.4.3 Limiting Geometric Property Values

We calculate a number of geometric properties from the level set surface. The zero order geometric property that we utilize is shortest distance from the level set surface to some ROI primitive. The first order property is the surface normal, $\mathbf{n} \equiv -\nabla\phi/|\nabla\phi|$. Second order information includes a variety of curvature measures of the surface. In Appendix 23.7 we outline a new numerical approach to deriving the mean, Gaussian and principle curvatures of a level set surface. Our scheme has numerical advantages relative to traditional central finite difference schemes

for computing the second order derivatives. We found the mean curvature to be a useful second order measure [184].

Another desirable feature of our operators is that they allow the user to control the geometric properties of surface in the region being edited. This feature is implemented with another cut-off function, $\mathcal{C}()$, within the level set speed function. $\mathcal{C}()$ allows the user to slow and then stop the level set deformation as a particular surface property approaches a user-specified value. We reuse the cut-off function, Eq. (23.5), defined in the previous section, as a building block for $\mathcal{C}()$. We define

$$\mathcal{C}(\gamma) = \begin{cases} \mathcal{P}(\gamma; \gamma_{min}^{low}, \gamma_{max}^{low}) & \text{for } \gamma \leq \bar{\gamma} \\ 1 - \mathcal{P}(\gamma; \gamma_{min}^{high}, \gamma_{max}^{high}) & \text{for } \gamma > \bar{\gamma} \end{cases} \quad (23.7)$$

where $\bar{\gamma} \equiv (\gamma_{max}^{low} + \gamma_{min}^{high})/2$. The four parameters γ_{min}^{low} , γ_{max}^{low} , γ_{min}^{high} , and γ_{max}^{high} define respectively the upper and lower support of the filter, see Figure 23.4(d).

23.4.4 Constraining the Direction of LS Motions

Another important feature of the level set framework is its ability to control the direction of the level set deformation. We are able to restrict the motion of the surface to only add or remove material during the level set editing operations. At any point the level set surface can only move in the direction of the local surface normal. Hence, we can simply redefine the speed function as $\min(\mathcal{G}, 0)$ to remove material (inward motion only) and $\max(\mathcal{G}, 0)$ to add material (outward motion only). In the case of curvature driven speed functions this produces min/max flows [476]. Of course no restriction on the direction of the motion need be imposed.

23.5 Definition of Surface Editing Operators

Given the building blocks described in the previous section, the level set surface editing operators outlined in Figure 23.3 may be defined. We begin by defining the well-known CSG operations that are essential to most editing systems. We then define the new level set speed functions that implement our surface editing operators by combining the geometric measures with the region-of-influence and cut-off functions.

23.5.1 CSG Operations

Since level set models are volumetric, the constructive solid geometry (CSG) [241] operations of union, difference and intersection may be applied to them. This provides a straightforward approach to copy, cut and paste operations on the level set surfaces. In our level set framework with a positive-inside/negative-outside sign convention for the distance volumes these are implemented as min/max operations [560] on the voxel values as summarized in Table 23.1. Any

two closed surfaces represented as signed distance volumes can be used as either the main edited model or the cut/copy primitive. In our editing system the user is able to arbitrarily scale, translate and rotate the models before a CSG operation is performed.

Table 23.1. Implementation of CSG operations on two level set models, A and B , represented by distance volumes V_A and V_B with positive inside and negative outside values.

Action	CSG Operation	Implementation
Copy	Intersection, $A \cap B$	$\text{Min}(V_A, V_B)$
Paste	Union, $A \cup B$	$\text{Max}(V_A, V_B)$
Cut	Difference, $A - B$	$\text{Min}(V_A, -V_B)$

23.5.2 Automatic Localized Level Set Blending

The surface models produced by the CSG paste operation typically produces sharp and sometimes jagged creases at the intersection of the two surfaces. We can dramatically improve this region of the surface by applying an automatic localized blending. The method is automatic because it only requires the user to specify a few parameter values. It is localized because the blending operator is only applied near the surface intersection region. One possible solution to localizing the blending is to perform the deformation in regions near both of the input surfaces. However, this naive approach would result in blending the two surfaces in *all* regions of space where the surfaces come within a user-specified distance of each other, creating unwanted blends. A better solution, and the one we use, involves defining the region of influence based on the distance to the *intersection curve* shared by both input surfaces. A sampled representation of this curve is the set of voxels that contains a zero distance value (within some sub-voxel value ϵ) to both surfaces. We have found this approximate representation of the intersection curve as a point set to be sufficient for defining a shortest distance d for the region-of-influence function, $\mathcal{D}_p(d)$, cf. Eq. (23.3). Representing the intersection curve by a point set allows the curve to take an arbitrary form - it can even be composed of multiple curve segments without introducing any complication to the computational scheme.

The blending operator moves the surface in a direction that minimizes a curvature measure, \mathcal{K} , on the level set surface. This is obtained by making the speed function, \mathcal{G} , Eq. (23.3), proportional to \mathcal{K} , leading to the following blending speed function:

$$\mathcal{F}_{blend}(\mathbf{x}, \mathbf{n}, \phi) = \alpha \mathcal{D}_p(d) \mathcal{C}(\mathcal{K}) \mathcal{K} \quad (23.8)$$

where α is a user-defined positive scalar that is related to the rate of convergence of the level set calculation, $\mathcal{D}_p(d)$ is defined in Eq. (23.6a) where d is the shortest distance from the level set surface to the intersection curve point set, and $\mathcal{C}(\mathcal{K})$

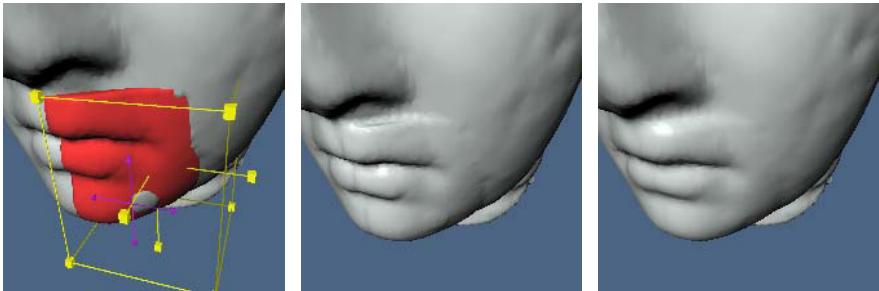


Figure 23.5. Close-up of the automatic blending used to repair the broken cheek on the Greek bust. Left: Positioning a mirrored copy of the left cheek on the cracked right cheek. Middle: The models are pasted together (CSG union operation), producing sharp, undesirable creases. Right: Same region after automatic blending based on mean curvature.

is given by Eq. (23.7) where \mathcal{K} is one of the curvatures define in Appendix 23.7. Through the functions \mathcal{D}_p and \mathcal{C} the user has full control over the region of influence of the blending (d_{min} and d_{max}) and the upper and lower curvature values of the blend ($\gamma_{min}^{low}, \gamma_{max}^{low}$ and $\gamma_{min}^{high}, \gamma_{max}^{high}$). Furthermore we can control if the blend adds or removes material, or both as described in Section 23.4.4.

Automatic blending is demonstrated in Figure 23.5. A piece of a cheek is positioned on the right side of a Greek bust to repair a crack. The two models are pasted together and automatic mean curvature-based blending is applied to smooth the creased intersection region. Note that blending only occurs along the intersection curve of the two models, thereby leaving the sharp features of the lips intact.

23.5.3 Localized Level Set Smoothing/Sharpening

The smoothing operator smooths the level set surface in a user-specified region. This is accomplished by enclosing the region of interest by a geometric primitive. The “region-of-influence primitive” can be any closed surface for which we have signed inside/outside information, e.g. a level set surface or an implicit primitive. We use superellipsoids [30] as a convenient ROI primitive, a flexible implicit primitive defined by two shape parameters. The surface is locally smoothed by applying motions in a direction that reduces the local curvature. This is accomplished by moving the level set surface in the direction of the local normal with a speed that is proportional to the curvature. Therefore the speed function for the smoothing operator is

$$\mathcal{F}_{smooth}(\mathbf{x}, \mathbf{n}, \phi) = \alpha \mathcal{D}_s(d) \mathcal{C}(\mathcal{K}) \mathcal{K}. \quad (23.9)$$

Here d denotes the signed value of the monotonic inside/outside function of the ROI primitive s evaluated at \mathbf{x} . As before, $\mathcal{D}_s(d)$ ensures that the speed function smoothly goes to zero as \mathbf{x} approaches the boundary of the ROI primitive.

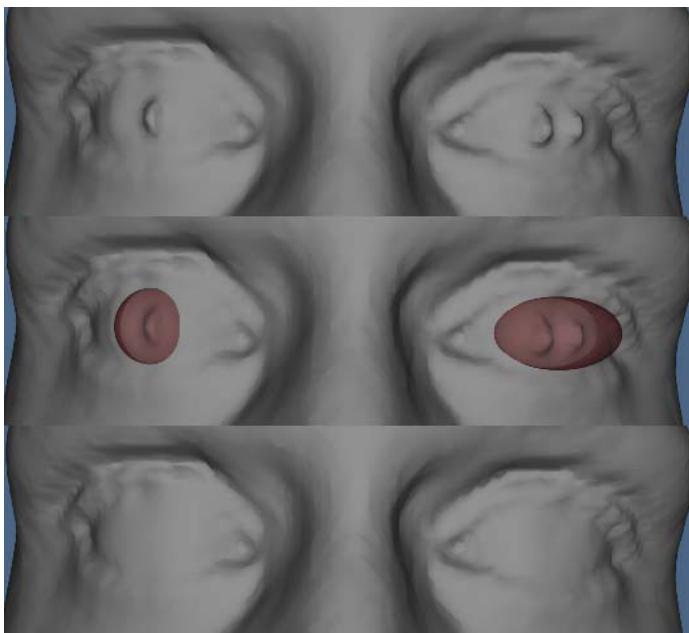


Figure 23.6. Regionally constrained smoothing. Top: Laser scan reconstruction with unwanted, pointed artifacts in the eyes. Middle: Defining the regions to be smoothed with two superellipsoids. Bottom: Smoothing the surface within the superellipsoids. The surface is constrained to only move inwards.

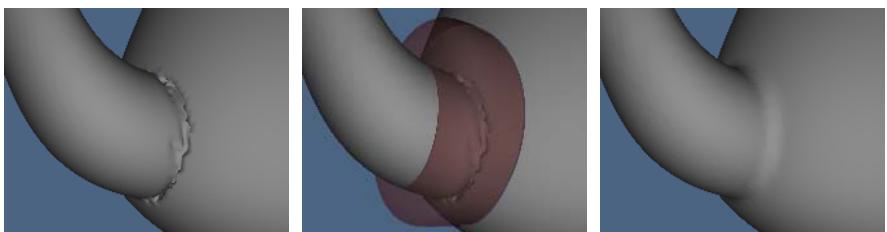


Figure 23.7. (left) Scan conversion errors near the teapot spout. (middle) Placing a superellipsoid around the errors. (right) The errors are smoothed away in 15 seconds. The surface is constrained to only move outwards.

Figure 23.6 demonstrates our smoothing operator applied to a laser scan reconstruction. Unwanted artifacts are removed from the eyes by first placing superellipsoids around the regions of interest. A smoothing operator constrained to only remove material is applied and the spiky artifacts are removed. Figure 23.7 demonstrates our smoothing operator applied to a preliminary 3D scan conversion of the Utah teapot. Unwanted artifacts are removed from the region where the spout meets the body of the teapot by first placing a superellipsoid around the region of interest. A smoothing operator constrained to only add material is

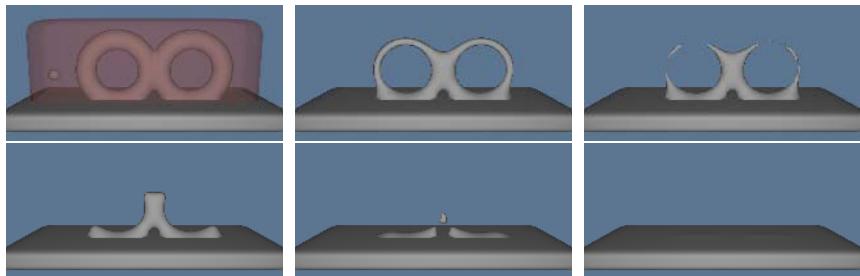


Figure 23.8. Changes in topological genus and the number of disconnected components are easily handled within a level set framework during smoothing. The superellipsoid defines the portion of the surface to be smoothed. The surface is constrained to move only inwards.

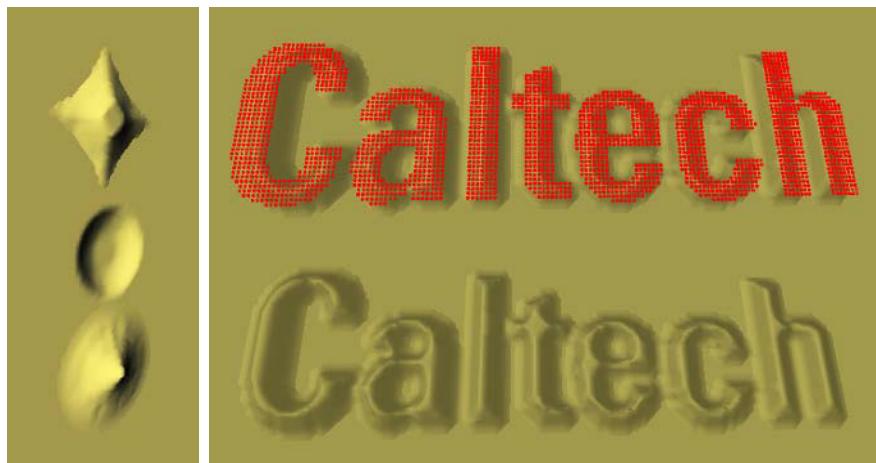


Figure 23.9. Left: Three types of single point attractions/repulsions using different ROI primitives and γ values. Right: A surface embossed with a point-sampled “Caltech” logo. Top: 2535 “Caltech” points. Bottom: Resulting embossing.

applied and the unwanted artifacts are removed. In our final, artificial smoothing example in Figure 23.8 a complex structure is completely smoothed away. This example illustrates that changes of topological genus and number of disconnected components are easily handled within a level set framework during smoothing.

We obtain a sharpening operator by simply inverting the sign of α in Eq. (23.9) and applying an upper cut-off to the curvature in $\mathcal{C}()$ in order to maintain numerical stability. The sharpening operator has been applied to the hair of the Greek bust in Figure 23.11.

23.5.4 Point Set Attraction and Embossing

We have developed an operator that attracts and repels the surface towards and away from a point set. These point sets can be samples of lines, curves, planes, patches and other geometric shapes, *e.g.* text. By placing the point sets near the surface, we are able to emboss the surface with the shape of the point set. Similar to the smoothing operator, the user encloses the region to be embossed with a ROI primitive *e.g.* a superellipsoid. The region-of-interest function for this operator is $\mathcal{D}_s(d)$, Eq. (23.6b).

First, assume that all of the attraction points are located outside the level set surface. \mathbf{p} denotes the attraction point in the set that is closest to \mathbf{x} , a point on the level set surface. Our operator only allows the level set surface to move towards \mathbf{p} if the unit vector, $\mathbf{u} \equiv (\mathbf{p} - \mathbf{x})/|\mathbf{p} - \mathbf{x}|$, is pointing in the same direction as the local surface normal \mathbf{n} at \mathbf{x} . Hence, the speed function should only be non-zero when $0 < \mathbf{n} \cdot \mathbf{u} \leq 1$. Since the sign of $\mathbf{n} \cdot \mathbf{u}$ is reversed if \mathbf{p} is located inside the level set surface, we simply require $\gamma \equiv -\text{sign}[\phi(\mathbf{p}, t)]\mathbf{n} \cdot \mathbf{u}$ to be positive for any closest attraction point \mathbf{p} . This amounts to having only positive cut-off values for $\mathcal{C}(\gamma)$. Finally we let $\mathcal{G} = -\alpha\phi(\mathbf{p}, t)$ since this will guarantee that the level set surface will stop once it reaches \mathbf{p} . The following speed function implements the point set attraction operator:

$$\mathcal{F}_{point}(\mathbf{x}, \mathbf{n}, \phi) = -\alpha\mathcal{D}_s(d)\mathcal{C}(\gamma)\phi(\mathbf{p}, t), \quad (23.10)$$

where d is a signed distance measure to a ROI primitive evaluated at \mathbf{x} on the level set surface, \mathbf{p} is the closest point in the set to \mathbf{x} , and γ is defined in the text above. The shape of the primitive and the values of the four positive parameters in Eq. (23.7) define the footprint and sharpness of the embossing. See Figure 23.9, left. Point repulsion is obtained by making α negative. Note that Eq. (23.10) is just one example of many possible point set attraction speed functions.

In Figure 23.9, right, a plane surface is embossed with 2535 points that have been acquired by scanning an image of a “Caltech” logo. The actual points are shown at the top, and the resulting embossing at the bottom.

23.5.5 Global Morphological Operators

The new level set operators presented above were designed to perform localized deformations of a level set surface. However, if the user wishes to perform a global smoothing of a level set surface, it is advantageous to use an operator other than \mathcal{F}_{smooth} . For a global smoothing the level set propagation is computed on the whole volume, which can be slow for large volumes. However, in this case morphological opening and closing operators [467] offer faster alternatives for globally smoothing level set surfaces. While we are not the first to explore morphological operators within a level set framework [459, 334], we have implemented them and find them useful. Morphological openings and closings consist of two fundamental operators, dilations D_ω and erosions E_ω . Dilation creates an offset surface a distance ω outward from the original surface, and ero-

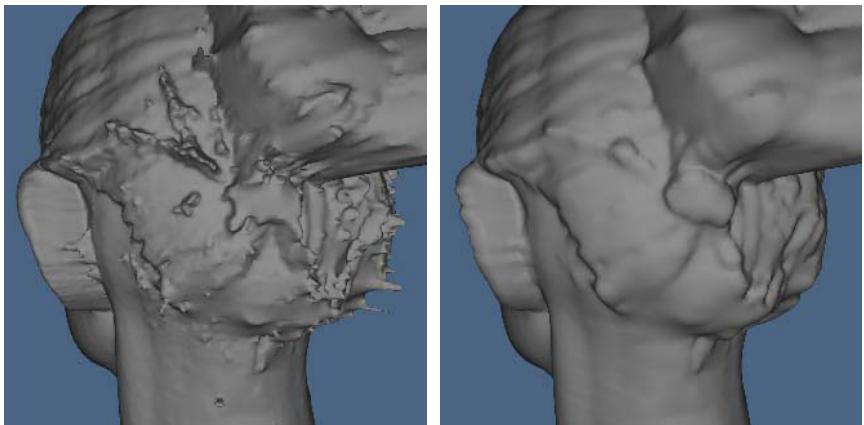


Figure 23.10. Applying a morphological opening to a laser scan reconstruction of a human head. The opening performs global smoothing by removing protruding structures smaller than a user-defined value.

sion creates an offset surface a distance ω inwards from the original surface. The morphological opening operator O_ω is an erosion followed by a dilation, i.e. $O_\omega = D_\omega \circ E_\omega$, which removes small pieces or thin appendages. A closing is defined as $C_\omega \phi = E_\omega \circ D_\omega \phi$, and closes small gaps or holes within objects. Morphological operators may be implemented by solving a special form of the level set equation, the Eikonal equation, $\pm \partial \phi / \partial t = |\nabla \phi| = 1$, up to a certain time t , utilizing Sethian's Fast Marching Method [473]. The value of t corresponds to the offset distance, ω , from the original surface, $\phi(t = 0)$. Figure 23.10 contains a model from a laser scan reconstruction that has been smoothed with an opening operator with ω equal to 3.

23.5.6 Editing Session Details

Figure 23.11 contains a series of screen shots taken of our level set modeling program while repairing a Greek bust. The Greek bust model was repaired by copying the nose from the human head model of Figure 23.10, and pasting and blending the copied model onto the broken nose. A piece from the right side of the bust was copied, mirrored, pasted and blended onto the left side of the face. Local smoothing operators were applied to various portions of her cheeks to clean minor cracks. Finally, the sharpening operator was applied within a user-defined region around her hair.

Table 23.2 lists typical operator execution times on a Silicon Graphics Onyx2 (R10K 250MHz) and Table 23.3 lists values of the parameters used to produce the examples shown in this chapter.

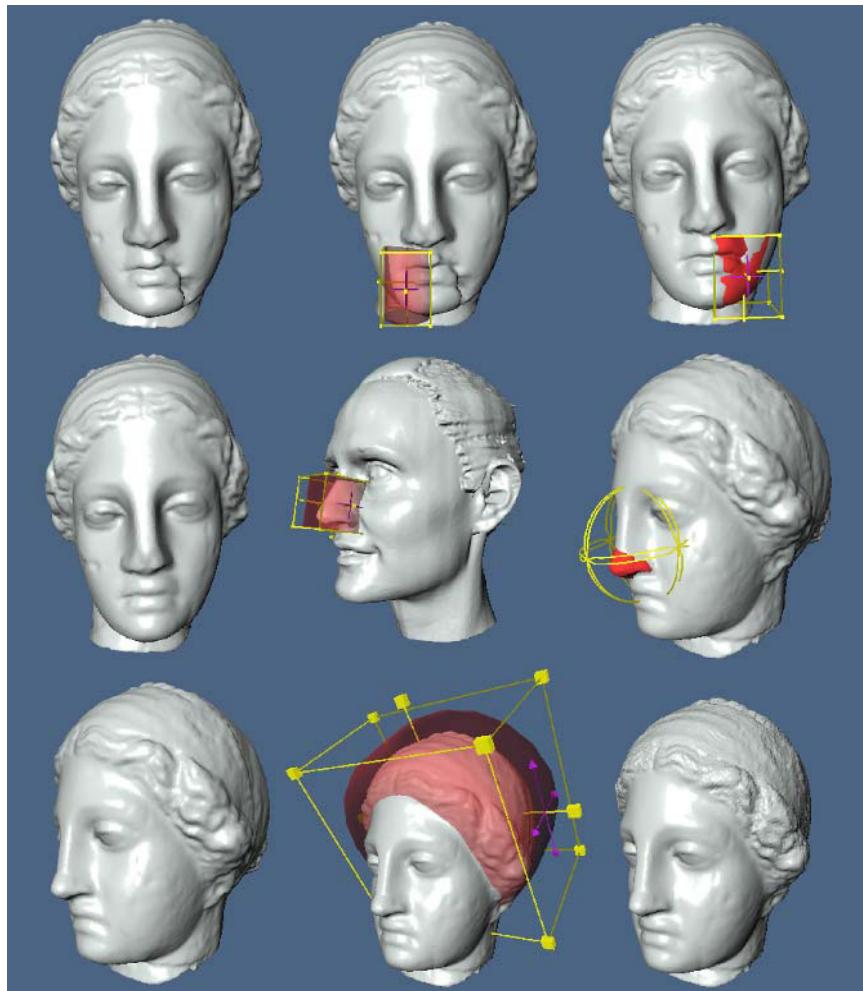


Figure 23.11. Repairing a Greek bust. The right cheek is first copied, mirrored, pasted, and blended back onto the left side of the bust. Next a nose is copied from a human head model, scaled and blended onto the broken nose of the Greek bust. Finally the hair of the bust is chiseled by a localized sharpening.

Table 23.2. Typical operator execution times on a R10K 250MHz MIPS processor.

Operation	Objects	sub-volume	Time
Paste	nose on bust	$210 \times 210 \times 150$	18 sec.
Blend	nose on bust	$39 \times 49 \times 57$	12 sec.
Smooth	teapot spout	$60 \times 55 \times 31$	15 sec.
Opening	human head	$256 \times 246 \times 193$	22 sec.
Emboss	single point	$21 \times 29 \times 29$	1.5 sec.

Table 23.3. Parameters used in examples. γ_{min}^{high} and γ_{max}^{high} are only used during sharpening. Their values are 0.8 and 0.9. No upper limit is placed on γ in the other examples.

Example	d_{min}	d_{max}	γ_{min}^{low}	γ_{max}^{low}
Nose Blending	7.0	9.0	0.04	0.060
Eye Smoothing	0.9	1.0	0.04	0.070
Spout Smoothing	0.9	1.0	0.10	0.130
Hair Sharpening	0.9	1.0	0.01	0.013
Plane Embossing	0.9	1.0	0.90	0.901

23.6 Conclusion and Future Work

We have presented an approach to implementing surface editing operators within a level set framework. By developing a new set of level set speed functions automatic blending, localized smoothing and embossing may be performed on level set models. Additionally we have implemented morphological and volumetric CSG operators to fill out our modeling environment. All of the information needed to deform a level set surface is encapsulated in the speed function, providing a simple, unified computational framework. The level set framework offers several advantages. By construction, self-intersection cannot occur, which guarantees the generation of physically-realizable, simple, closed surfaces. Additionally, level set models easily change topological genus, and are free of the edge connectivity and mesh quality problems associated with mesh models.

Several issues still must be addressed to improve our work. Currently level set implementations are based on uniform samplings of space, a fact that effectively limits the resolution of the objects that can be modeled. The development of adaptive level set methods would allow our operators to be applied to adaptive distance fields. For certain surface editing situations it would be desirable to allow the user to have better control of the topological genus as the level set surface deforms. Additionally, it is possible to shorten the time needed to edit level set surfaces. Incrementally updating the mesh used to view the edited surface, utilizing direct volume rendering hardware, parallelizing the level set computations, and exploring multiresolution volumetric representations will lead to editing operations that require only a fraction of a second, instead of tens of seconds.

We have presented five example level set surface editing operators. Given the generality and flexibility of our framework many more can be developed. We intend to explore operators that utilize Gaussian and principal curvature, extend embossing to work directly with lines, curves and solid objects, and ones that may be utilized for general surface manipulations, such as dragging, warping, and sweeping.

Acknowledgements

We would like to thank Sean Mauch for his scan conversion programs, Jason wood for his useful visualization tools, Al Barr and Mathieu Desbrun for their helpful suggestions, and Katrine Museth for helping us with one of the figures. The Greek bust and human head models were provided by Cyberware Inc. The teapot model was provided by the University of Utah's Geometric Design and Computation Group. This work was financially supported by National Science Foundation grants ASC-89-20219, ACI-9982273 and ACI-0083287.

23.7 Appendix: Curvature of Level Set Surfaces

The principle curvatures and principle directions are the eigenvalues and eigenvectors of the *shape matrix* [164]. For an implicit surface, the shape matrix is the derivative of the normalized gradient (surface normals) projected onto the tangent plane of the surface. If we let the normals be $\mathbf{n} = \nabla\phi/|\nabla\phi|$, the derivative of this is the 3×3 matrix

$$\mathbf{N} = \begin{pmatrix} \frac{\partial \mathbf{n}}{\partial x} & \frac{\partial \mathbf{n}}{\partial y} & \frac{\partial \mathbf{n}}{\partial z} \end{pmatrix}^T. \quad (23.11)$$

The projection of this derivative matrix onto the tangent plane gives the shape matrix [164] $\mathbf{B} = \mathbf{N}(I - \mathbf{n} \otimes \mathbf{n})$, where \otimes is the exterior product. The eigenvalues of the matrix \mathbf{B} are k_1, k_2 and zero, and the eigenvectors are the principle directions and the normal, respectively. Because the third eigenvalue is zero, we can compute k_1, k_2 and various differential invariants directly from the invariants of \mathbf{B} . Thus the weighted curvature flow is computing from \mathbf{B} using the identities $D = \|\mathbf{B}\|_2$, $H = \text{Tr}(\mathbf{B})/2$, and $K = 2H^2 - D^2/2$. The choice of numerical methods for computing \mathbf{B} is discussed in the following section. The principle curvature are calculated by solving the quadratic

$$k_{1,2} = H \pm \sqrt{\frac{D^2}{2} - H^2}. \quad (23.12)$$

In many circumstances, the curvature term, which is a kind of directional diffusion, which does not suffer from overshooting, can be computed directly from first- and second-order derivatives of ϕ using central difference schemes.

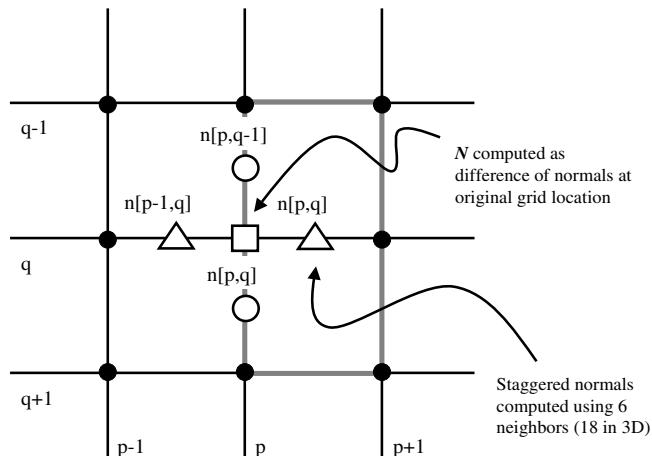


Figure 23.12. The shape matrix B is computed by using the differences of staggered normals.

However, we have found that central differences do introduce instabilities when computing flows that rely on quantities other than the mean curvature. Therefore we use the method of *differences of normals* [449, 576] in lieu of central differences. The strategy is to compute normalized gradients at staggered grid points and take the difference of these staggered normals to get centrally located approximations to N . See Figure 23.12. The shape matrix B is computed with gradient estimates from central differences. The resulting curvatures are treated as speed functions (motion in the normal direction), and the associated gradient magnitude is computed using the up-wind scheme.

24

Simulating Natural Phenomena

Ronald Fedkiw

Abstract

Level set methods have gained popularity in a number of research areas from computational physics to computer vision to computer graphics. They provide for a robust representation of geometry that can be dynamically evolved by solving partial differential equations in a manner that has been fine tuned by a community of researchers over a number of years. Traditionally, simulation of natural phenomena has been the focus of the computational physics community, but more recently computer graphics researchers have become interested in these methods in part because water, smoke and fire are exciting elements in high demand for special effects in feature films. Although computer graphics is the “inverse problem” to computer vision and this chapter is genuinely more concerned with synthesis than acquisition, understanding the application of level set methods to the physics problems that motivated their invention should enable computer vision researchers to gain a deeper understanding and appreciation of these methods and the related underlying mathematics.

24.1 Introduction

Consider the digital representation of a real word object. For example, suppose one starts with a set of data points such as those shown in figure 24.1 (left) which were obtained from an MRI image of a rat brain. In [610], variational and partial differential equation (PDE) methods were proposed to solve this problem in the context of level set methods. The authors first constructed a surface energy for the variational formulation and then obtained the gradient flow of the energy functional. This was recast in a level set framework and used to “shrink wrap” an implicit surface around the data points as shown in figure 24.1 (right) reminiscent of snakes or active contours [262]. The virtue of the approach proposed in [610] is that one can blindly obtain acceptable results on difficult problems. However, this same blind application of the mathematics can sometimes lead to

various difficulties such as algorithms that are too slow or those that do not perform well in certain situations. One reason to become well-versed in the PDE and level set technology is to overcome the difficulties commonly encountered when first devising methods of this type. For example, although the PDE based method in [610] was found to be impractical due to the costly solution of the nonlinear parabolic PDE proposed in that paper, later consideration in [608] showed that one could obtain a similar solution by solving a more efficient hyperbolic PDE, or by using an even more efficient marching type method. Thus, before abandoning a classical PDE approach, one should carefully explore for options that, for example, can lead to orders of magnitude improvements in efficiency. This emphasizes the importance of a thorough understanding of PDE and level set methods to the modern day computer vision researcher.

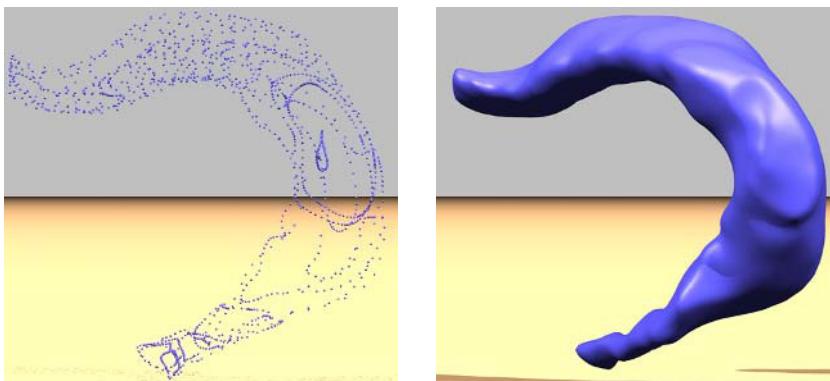


Figure 24.1. Sample data points acquired from an MRI image of a rat brain (left), and a reconstructed level set representation (right).

Although there is a rich field of computational fluid dynamics (CFD) in which many of the modern day algorithms for solving PDE's were and are still being developed, we instead present this chapter in the context of simulation of natural phenomena for computer graphics (CG) applications since the computer vision researcher will most likely already be familiar with, or even publish in, this area. However, we stress that the advanced computer vision researcher might wish to familiarize themselves with some of the underlying mathematical principles advocated in CFD, especially if the goal is to make clever observations such as those in [37] linking the Navier-Stokes equations for fluid flow to image inpainting.

We start with a general discussion of the equations of fluid flow discussing these equations in the context of smoke simulations. There we point out that one cannot always solve the equations exactly due limited computational resources. This is important to keep in mind when considering real-time or interactive-time applications. Then we discuss one particular way of modifying the equations in order to obtain the desired effect. Of course, the general problem of constructing equations for computational modeling drives a large portion of computer vision research. Next, we move on to two phase flows such as those involving liquids.

Here, level set methods thrive both in the computational physics and the computer graphics communities. The characteristically smooth liquid surfaces that are both physically accurate and visually pleasing are difficult to obtain with other interface tracking methods. Simulation of water is a fairly difficult problem demanding sophisticated methods for evolving surfaces highlighting the real potential of level set methods. Finally we move on to fire simulation which in some sense is still a partially unsolved problem in both graphics and physics (in physics one refers to combustion). In the context of fire simulation, we use the modeling tools discussed for smoke simulation, the advanced level set methods discussed for water simulation, and introduce new methods for modeling boundary conditions at discontinuities.

24.2 Smoke

The modeling of natural phenomena such as smoke remains a challenging problem in computer graphics. This is not surprising since the motion of gases is highly complex and turbulent. Visual smoke models have many obvious applications in the CG industry including special effects and interactive games. Ideally, a good CG smoke model should both be easy to use and produce highly realistic results. Obviously, the modeling of smoke and gases is of importance to other engineering fields as well. Only recently have researchers in computer graphics started to excavate the abundant CFD literature for algorithms that can be adopted and modified for CG applications. Unfortunately, most CG smoke models are either too slow or suffer from too much numerical dissipation. In [193], Fedkiw et al. proposed a method that exploits physics unique to smoke in order to design a numerical method that is both fast and efficient on the relatively coarse grids traditionally used in CG applications (as compared to the much finer grids used in the CFD literature).

In [193], the smoke's velocity was modeled with the inviscid incompressible Euler equations which are more appropriate for gas modeling and less computationally intensive than the full viscous Navier-Stokes equations. The advection part of these equations was solved using a semi-Lagrangian integration approach which is very popular in the atmospheric sciences community for modeling large scale flows dominated by constant advection where large time steps are desired, see e.g. [495] for a review. This was followed by a pressure-Poisson equation guaranteeing that the model is stable for any choice of the time step. One of the main contributions of [193] was to reduce the numerical dissipation inherent to semi-Lagrangian schemes by using a technique from the CFD literature known as “vorticity confinement” [497] to model the small scale rolling features characteristic of smoke that are usually absent on coarse grid simulations. The basic idea is to inject the energy lost due to numerical dissipation back into the fluid using a forcing term. This force is designed specifically to increase the vorticity of the flow, which visually keeps the smoke alive over time. This forcing term is consis-

tent with the Euler equations in the sense that it disappears as the number of grid cells is increased. In CFD this technique was applied to the numerical computation of complex turbulent flow fields around helicopters where it is not possible to add enough grid points to accurately resolve the flow field. The computation of the force only adds a small computational overhead, and consequently the resulting simulations are almost as fast as the one's obtained from the basic "stable fluids" semi-Lagrangian algorithm in [493] without the unappealing numerical dissipation. A sample calculation is shown in figure 24.2.

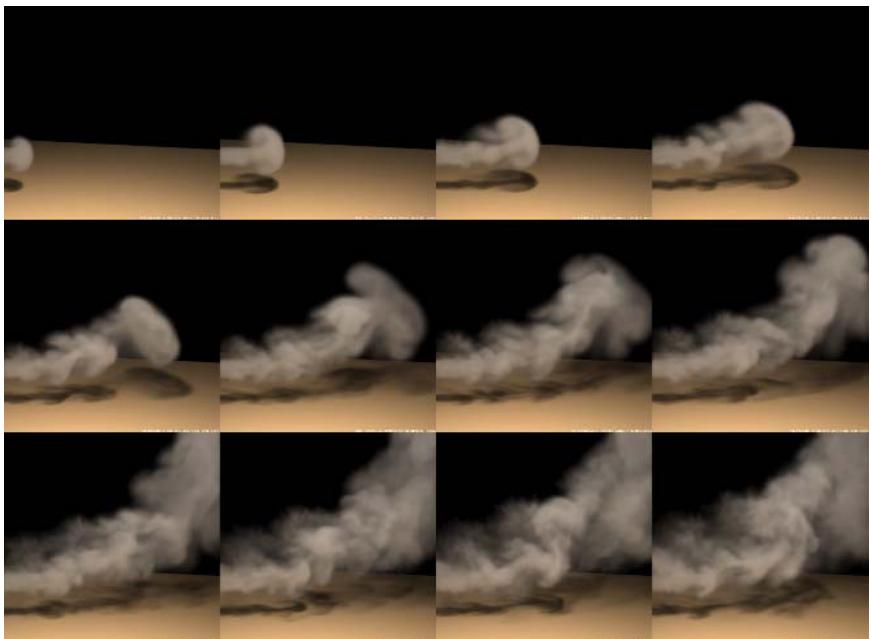


Figure 24.2. A warm smoke plume injected from left to right rises under the effects of buoyancy. Note the high level of detailed rolling effects present in the figures.

For simplicity, assume that our gases can be modeled as inviscid, incompressible, constant density fluids. The effects of viscosity are negligible in gases especially on coarse grids where numerical dissipation dominates physical viscosity and molecular diffusion. When the smoke's velocity is well below the speed of sound, the compressibility effects are negligible as well, and the assumption of incompressibility greatly simplifies the numerical methods. Consequently, the equations that model the smoke's velocity, $\mathbf{V} = (u, v, w)$, are given by the incompressible Euler equations [302]

$$\nabla \cdot \mathbf{V} = 0 \quad (24.1)$$

$$\frac{\partial \mathbf{V}}{\partial t} = -(\mathbf{V} \cdot \nabla) \mathbf{V} - \nabla p + \mathbf{f}. \quad (24.2)$$

These two equations state that the velocity should conserve both mass (equation 24.1) and momentum (equation 24.2). p is the pressure of the gas and \mathbf{f} accounts for external forces. The constant density of the fluid has been arbitrarily rescaled to unity. As in [179, 199, 493], [193] solved these equations in two steps. First an intermediate velocity field \mathbf{V}^* is computed solving equation 24.2 over a time step Δt without the pressure term

$$\frac{\mathbf{V}^* - \mathbf{V}}{\Delta t} = -(\mathbf{V} \cdot \nabla)\mathbf{V} + \mathbf{f}. \quad (24.3)$$

After this the velocity field \mathbf{V}^* is forced to be incompressible using a projection method. This is equivalent to computing the pressure from the following Poisson equation

$$\nabla^2 p = \frac{1}{\Delta t} \nabla \cdot \mathbf{V}^* \quad (24.4)$$

with pure Neumann boundary condition, i.e., $\frac{\partial p}{\partial \mathbf{N}} = 0$ at a boundary point with normal \mathbf{N} . Note that it is also straightforward to impose Dirichlet boundary conditions where the pressure is specified directly as opposed to specifying its normal derivative. The intermediate velocity field is then made incompressible by subtracting the gradient of the pressure from it

$$\mathbf{V} = \mathbf{V}^* - \Delta t \nabla p. \quad (24.5)$$

Besides evolving the velocity field, additional equations for the evolution of both the temperature T and the smoke's density ρ are needed. Assuming that these two scalar quantities are simply advected according to the smoke's velocity leads to

$$\frac{\partial T}{\partial t} = -(\mathbf{V} \cdot \nabla)T, \quad (24.6)$$

and

$$\frac{\partial \rho}{\partial t} = -(\mathbf{V} \cdot \nabla)\rho. \quad (24.7)$$

Both the density and the temperature affect the fluid's velocity. Heavy smoke tends to fall downwards due to gravity while hot gases tend to rise due to buoyancy. A simple model that accounts for these effects can be constructed by defining external forces that are directly proportional to the density and the temperature

$$\mathbf{f}_{buoy} = -\alpha \rho \mathbf{z} + \beta(T - T_{amb})\mathbf{z}, \quad (24.8)$$

where $\mathbf{z} = (0, 0, 1)$ points in the upward vertical direction, T_{amb} is the ambient temperature of the air and α and β are two positive constants with appropriate units. Note that when $\rho = 0$ and $T = T_{amb}$, this force is identically zero.

Usually smoke and air mixtures contain velocity fields with large spatial deviations accompanied by a significant amount of rotational and turbulent structure

on a variety of scales. Nonphysical numerical dissipation damps out these interesting flow features, and the goal of the approach in [193] is to add them back on the coarse grid. One way of adding them back would be to create a random or pseudo-random small scale perturbation of the flow field using either a heuristic or physically based model. For example, one could generate a divergence free velocity field using a Kolmogorov spectrum and add this to the computed flow field to represent the missing small scale structure (see [494] for some CG applications of the Kolmogorov spectrum). While this provides small scale detail to the flow, it does not place the small scale details in the physically correct locations within the flow field where they are missing. Instead, the details are added in a haphazard fashion and the smoke can appear to be “alive”, rolling and curling in a nonphysical fashion. The key to realistic animation of smoke is to make it look like a passive natural phenomena as opposed to a “living” creature made out of smoke.

The method proposed in [193] looks for the locations within the flow field where small scale features should be generated and adds the small scale features in these locations in a physically based fashion that promotes the passive rolling of smoke that gives it a realistic turbulent look on a coarse CG grid. With unlimited computing power, any consistent numerical method could be used to obtain acceptable results simply by increasing the number of grid points until the desired limiting behavior is observed. However, in practice, computational resources are limited, grids are fairly coarse (even coarser in CG than in CFD), and the discrete difference equations may not be asymptotically close enough to the continuous equations for a particular simulation to behave in the desired physically correct fashion. The key idea is to design a consistent numerical method that behaves in an interesting and physically plausible fashion on a coarse grid. In general, this is very difficult to do, but luckily a vorticity confinement method was recently invented by Steinhoff, see e.g. [497], for the numerical computation of complex turbulent flow fields around helicopters where it is not possible to add enough grid points to accurately resolve the flow.

The first step in generating the small scale detail is to identify where it comes from. In incompressible flow, the vorticity

$$\omega = \nabla \times \mathbf{V} \quad (24.9)$$

provides the small scale structure. Each small piece of vorticity can be thought of as a paddle wheel trying to spin the flow field in a particular direction. Artificial numerical dissipation damps out the effect of these paddle wheels, and the key idea is to simply add it back. First normalized vorticity location vectors

$$\mathbf{N} = \frac{\eta}{|\eta|}, \quad (\eta = \nabla |\omega|) \quad (24.10)$$

that point from lower vorticity concentrations to higher vorticity concentrations are computed. Then the magnitude and direction of the paddle wheel force is

computed as

$$\mathbf{f}_{conf} = \epsilon \Delta x (\mathbf{N} \times \boldsymbol{\omega}) \quad (24.11)$$

where $\epsilon > 0$ is used to control the amount of small scale detail added back into the flow field and the dependence on the spatial discretization Δx guarantees that as the mesh is refined the physically correct solution is still obtained.

In [193], a finite volume spatial discretization was used to numerically solve the equations. The computational domain was divided into identical voxels with the temperature, the smoke's density and the external forces defined at the center of each voxel while the velocity is defined on the appropriate voxel faces. This staggered MAC grid arrangement of the velocity field is standard for incompressible flows, see e.g. [230]. To handle boundaries immersed in the fluid all voxels that intersect an object are tagged as being occupied, and all occupied voxel cell faces have their velocity set to that of the object. Similarly, the temperature at the center of the occupied voxels is set to the object's temperature. Consequently an animator can create many interesting effects by simply moving or heating up an object. The smoke's density is of course equal to zero inside the object. However, to avoid a sudden drop-off of the density near the object's boundary, the density at boundary voxels is set equal to the density of the closest unoccupied voxel.

The fluid velocity is updated in three steps. First, the advection term in equation 24.3 is solved for using a semi-Lagrangian scheme. The semi-Lagrangian algorithm builds a new grid of velocities from the ones already computed by backward tracing the midpoints of each voxel face through the velocity field. New velocities are then interpolated at these points and their values are transferred to the face cells they originated from. It is possible that the point ends up in one of the occupied voxels, and in this case one simply clips the path against the object's boundary. This guarantees that the point always lies in the unoccupied fluid. Simple linear interpolation is easy to implement and combined with the vorticity confinement force gives satisfactory results. Next, the force fields are added to the velocity grid. These include user supplied fields, the buoyancy force defined by equation 24.8 and the confinement force defined by equation 24.11. This is done by simply multiplying each force by the time step and adding it to the velocity. Finally, the velocity field is made incompressible to conserve mass. As stated above, this involves the solution of a Poisson equation for the pressure (equation 24.4). The discretization of this equation results in a sparse linear system of equations. Free Neumann boundary conditions can be applied at the occupied voxels by setting the normal pressure gradient equal to zero at the occupied boundary faces. The system of equations is symmetric, and the most natural linear solver in this case is the conjugate gradient method which is easy to implement and has much better convergence properties than simple relaxation methods. An incomplete Choleski preconditioner is recommended to improve convergence. These techniques are quite standard and the reader is referred to [217] for more details. In practice, only about 20 iterations of this solver gives visually acceptable results, although the solution is not usually converged at this point. After the pressure is computed, its gradient is subtracted from the velocity, as mentioned above. After

the velocity is updated, both the temperature and the smoke's density are solved for, again using the semi-Lagrangian scheme.

24.3 Water

Water surrounds us in our everyday lives. Given the ubiquity of water and our constant interaction with it, the animation and rendering of water poses one of the greatest challenges in computer graphics. The difficulty of this challenge was underscored recently through the use of water effects in a variety of motion pictures including the recent feature film "Shrek" where water, mud, beer and milk effects were seen. In response to a question concerning what was the single hardest shot in "Shrek", DreamWorks SKG principal and producer of "Shrek", Jeffrey Katzenberg, stated, *It's the pouring of milk into a glass.* [240]. This illustrates the need for photorealistic simulation and rendering of water (and other liquids such as milk), especially in the case of complex, three dimensional behavior as seen when water is poured into a glass as in figure 24.3. A key to achieving this goal is the visually accurate treatment of the surface separating the water from the air. The behavior of this surface provides the visual impression that water is being seen. If the numerical simulation method used to model this surface is not robust enough to capture the essence of water, then the effect is ruined for the viewer.

In [198], Foster and Fedkiw noted that the level set method suffered from excessive numerical dissipation on coarse grids resulting in a visually disturbing nonphysical loss of water volume during the course of the simulation. They realized that this volume loss could be alleviated in part by introducing marker particles near the interface and by using these marker particles to correct errors in the level set function in regions of high curvature. Later, in [179], the authors worked to make the approach advocated in [198] computational accurate enough to be applied to flow physics problems. Notably, both of these papers were inspired by the control problems under consideration by the second and fourth authors in [179]. The approach advocated in [179] can be thought of as a "thickened" front tracking approach to modeling the surface and was called the "particle level set method". The improvements in [179] (over [198]) were achieved by focusing on modeling the surface as opposed to the liquid volume itself as was done in [198].

This shift in philosophy away from volume modeling and towards surface modeling is the key idea behind [180] resulting in better photorealistic behavior of the water surface. For example, [180] proposed a new treatment of the velocity at the surface in order to obtain more visually realistic water surface behavior. The motion of both the massless marker particles and the implicit function representing the surface is dependent upon the velocities contained on the underlying computational mesh. By extrapolating velocities across the water surface and into the region occupied by the air, more accurate and better visual results were obtained. In the limit as the computational grid is refined, the resulting surface condition is

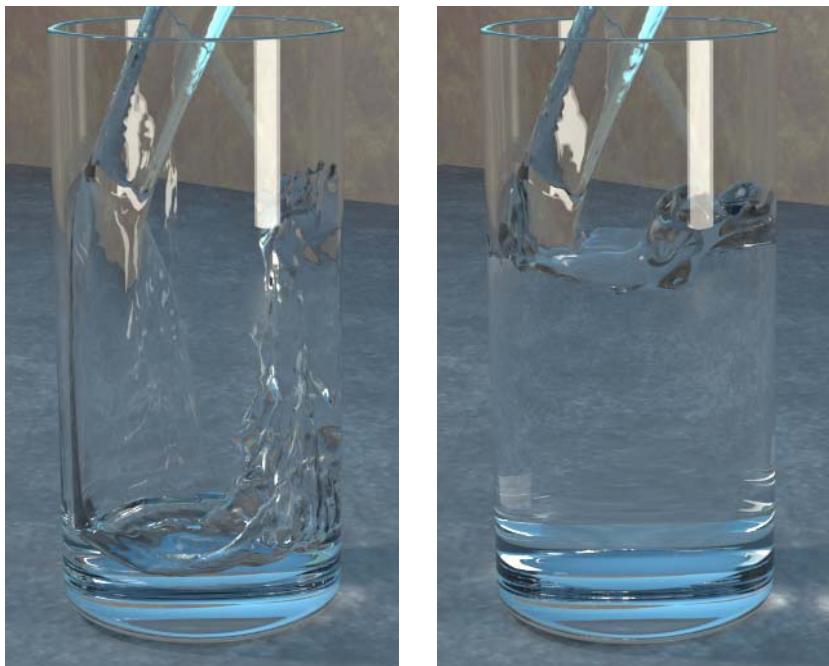


Figure 24.3. Water being poured into a glass (55x120x55 grid cells).

identical to the traditional approach of making the velocity divergence free, but it gives more visually appealing and physically plausible results on coarse grids. Furthermore, this velocity extrapolation procedure allows a degree of control to be applied to the behavior of the water surface. For example, one can add dampening and/or churning effects forcing it to quiet down or splash up faster than would be allowed by a straightforward physical simulation.

The liquid volume was simulated as one side of an isocontour of an implicit level set function, ϕ . The surface of the water is defined by the $\phi = 0$ isocontour with $\phi < 0$ representing the water and $\phi > 0$ representing the air. By using an implicit function representation of the liquid volume, one obtains a smooth, temporally coherent liquid surface. [198] rejected the use of particles alone to represent the liquid surface because it is difficult to calculate a visually desirable smooth liquid surface from the discrete particles alone. The implicit surface is dynamically evolved in space and time according to the underlying liquid velocity \mathbf{V} . The appropriate equation to do this is

$$\phi_t + \mathbf{V} \cdot \nabla \phi = 0. \quad (24.12)$$

A level set only approach to modeling the surface will not yield realistic surface behavior due to an excessive amount of volume loss on coarse grids. A seminal advance of [198] in creating realistic liquids for computer animation is the hybridization of the visually pleasing smooth implicit function modeling of the liquid volume with particles that can maintain the liquid volume on coarse grids. The inclusion of particles provides a way for capturing the liveliness of a real

liquid with spray and splashing effects. Curvature was used as an indicator for allowing particles to influence the implicit surface representation of the water. This is a natural choice since small droplets of water have very high curvature and dynamic implicit surfaces have difficulty resolving features with sharp corners.

Figure 24.4 (left) shows the initial data for a notched disk that was rotated for one rigid body rotation about the point (50,50). Figure 24.4 (right) shows the result obtained using a level set only approach where both the higher and lower corners of the disk are erroneously shaved off causing both loss of visual features and an artificially viscous look for a liquid. This numerical result was obtained using a highly accurate fifth order WENO discretization of equation 24.12 (see e.g. [256, 398]). Figure 24.5 (left) shows the result obtained with the method from [198]. The particles inside the disk do not allow the implicit surface to cross over them and help to preserve the two corners near the bottom. However, there is little they can do to stop the implicit surface from drifting away from them near the top corners. This represents loss of air or bubble volume as the method erroneously gains liquid volume. This is not desirable since many complex water motions such as wave phenomenon are due in part to differing heights of water columns adjacent to each other. Loss of air in a water column reduces the pressure forces from neighboring columns destroying many of the dynamic splashing effects as well as the overall visually stimulating liveliness of the liquid. While the hybrid liquid volume model of [198] attempts to maintain the volume of the liquid accurately, it fails to model the air or more generally the opposite side of the liquid surface. [180] shifts the focus away from maintaining a liquid volume towards maintaining the liquid surface itself. An immediate result of this approach is that it leads to symmetry in particle placement. That is, particles are placed on both sides of the surface and used to maintain an accurate representation of the surface itself regardless of what may be on one side or the other. The particles are not meant to track volume, they are intended to correct errors in the surface representation by the implicit level set function. See [179] for more details. Figure 24.5 (right) shows that this new method correctly computes the rigid body rotation for the notched disk preserving both the water and the air volumes so that more realistic water motion can be obtained.

In the particle level set method, two sets of particles are randomly placed in a “thickened” surface region (e.g. three grid cells on each side of the surface) with *positive* particles in the $\phi > 0$ region and *negative* particles in the $\phi < 0$ region. There is no need to place particles far away from the surface since the sign of the level set function readily identifies these regions gaining a large computational savings. The number of particles placed in each cell is an adjustable parameter that can be used to control the amount of resolution, e.g. 64 particles per cell works well in three spatial dimensions. Each particle possesses a radius, r_p , which is constrained to take a minimum and maximum value based upon the size of the underlying computational cells used in the simulation. A minimum radius of $.1 \min(\Delta x, \Delta y, \Delta z)$ and maximum radius of $.5 \min(\Delta x, \Delta y, \Delta z)$ appear to work well. The radius of a particle changes dynamically throughout the simulation, since a particle’s location relative to the surface changes. The radius

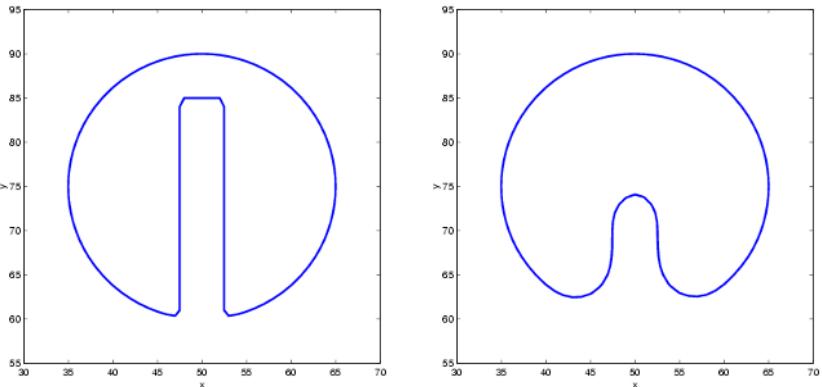


Figure 24.4. Initial data (left) and solution after one rotation (right) using the level set only method.

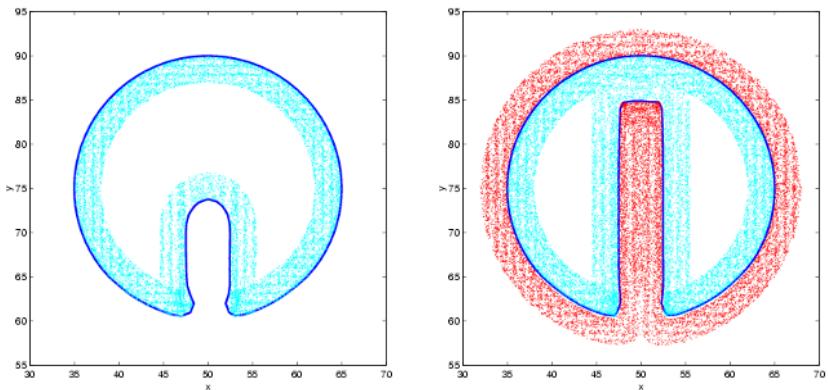


Figure 24.5. Solution after one rotation using particles on the inside only (left) and using particles on both sides (right).

is set according to:

$$r_p = \begin{cases} r_{max} & \text{if } s_p \phi(\mathbf{x}_p) > r_{max} \\ s_p \phi(\mathbf{x}_p) & \text{if } r_{min} \leq s_p \phi(\mathbf{x}_p) \leq r_{max} \\ r_{min} & \text{if } s_p \phi(\mathbf{x}_p) < r_{min} \end{cases}, \quad (24.13)$$

where s_p is the sign of the particle (+1 for positive particles and -1 for negative particles). This radius adjustment keeps the boundary of the spherical particle tangent to the surface whenever possible. This fact combined with the overlapping nature of the particle spheres allows for an enhanced reconstruction of the liquid surface.

The marker particles and the implicit function are separately integrated forward in time using a forward Euler time integration scheme. The implicit function is integrated forward using equation 24.12, while the particles are passively advected with the flow using $d\mathbf{x}_p/dt = \mathbf{V}_p$, where \mathbf{V}_p is the fluid velocity interpolated to the particle position \mathbf{x}_p .

The main role of the particles is to detect when the implicit surface has suffered inaccuracies due to the coarseness of the computational grid in regions with sharp features. Particles that are on the wrong side of the interface by more than their radius, as determined by a locally interpolated value of ϕ at the particle position \mathbf{x}_p , are considered to have *escaped* their side of the interface. This indicates potential errors in the implicit surface representation of the interface. In smooth, well resolved regions of the interface, our implicit level set surface representation is highly accurate and particles do not drift a non-trivial distance across the interface. We associate a spherical implicit function, designated ϕ_p , with each particle p whose size is determined by the particle radius, i.e.

$$\phi_p(\mathbf{x}) = s_p(r_p - |\mathbf{x} - \mathbf{x}_p|). \quad (24.14)$$

Any difference in ϕ from ϕ_p indicates potential errors in the implicit function representation of the surface. That is, the implicit version of the surface and the particle version of the surface disagree. Escaped positive particles are used to rebuild the $\phi > 0$ region and escaped negative particles are used to rebuild the $\phi < 0$ region. The reconstruction of the implicit surface occurs locally within the cell that each escaped particle currently occupies. Using equation 24.14, the ϕ_p values of escaped particles are calculated for the eight grid points on the boundary of the cell containing the particle. This value is compared to the current value of ϕ for each grid point and the smaller value (in magnitude) is taken, which is the value closest to the $\phi = 0$ isocontour defining the surface. This is done for all escaped positive and all escaped negative particles. The result is an improved representation of the surface of the liquid. This error correction method is applied after every computational step in which ϕ has been modified in some way. This occurs when ϕ is integrated forward in time and when the implicit function is reinitialized to a signed distance function using

$$\phi_\tau = -S(\phi_{\tau=0})(|\nabla\phi| - 1), \quad (24.15)$$

where τ is a fictitious time and $S(\phi)$ is a smoothed signed distance function given by

$$S(\phi) = \frac{\phi}{\sqrt{\phi^2 + (\Delta x)^2}}. \quad (24.16)$$

More details on this reinitialization equation are given in [501].

In complex flows, a liquid interface can be stretched and torn in a dynamic fashion. The use of only an initial seeding of particles will not capture these effects well, as regions will form that lack a sufficient number of particles to adequately perform the error correction step. Periodically, e.g. every 20 frames, particles are randomly reseeded about the “thickened” interface to avoid this dilemma. This is done by randomly placing particles near the interface, and then using geometric information contained within the level set function (e.g. the direction of the shortest possible path to the surface is given by $\mathbf{N} = \nabla\phi/|\nabla\phi|$) to move the particles to their respective domains, $\phi > 0$ or $\phi < 0$. The goal of this reseeding step is to preserve the initial particle resolution of the interface, e.g. 64 particles per cell.

Thus, if a given cell has too few or too many particles, some can be added or deleted respectively.

Although the Navier-Stokes equations can be used to find the velocity within the liquid volume, boundary conditions are needed for the velocity on the air side near the free surface. These boundary condition velocities are used in updating the Navier-Stokes equations, moving the surface, and moving the particles placed near the surface. The velocity at the free surface of the water can be determined through the usual enforcement of the conservation of mass (volume) constraint, $\nabla \cdot \mathbf{V} = 0$. This equation allows one to determine the velocities on all the faces of computational cells that contain the $\phi = 0$ isocontour. Unfortunately, the procedure for doing this is not unique when more than one face of a cell needs a boundary condition velocity. A variety of methods have been proposed, e.g. see [107] and [199]. [180] proposed a different approach altogether, the extrapolation of the velocity across the surface into the surrounding air. As the computational grid is refined, this method is equivalent to the standard one, but it gives a smoother and more visually pleasing motion of the surface on coarser (practical sized) grids. [180] extrapolated the velocity out a few grid cells into the air, obtaining boundary condition velocities in a band of cells on the air side of the surface. This allows one to use higher order accurate methods and to obtain better results when moving the implicit surface using equation 24.12, and also provides velocities for updating the position of particles on the air side of the surface. Velocity extrapolation also assists in the implementation of the semi-Lagrangian “stable fluid” method, since there are times when characteristics generated by this approach look back across the interface (a number of cells) into the air region for valid velocities.

The equation modeling this extrapolation for the x component of the velocity, u , is given by

$$\frac{\partial u}{\partial \tau} = -\mathbf{N} \cdot \nabla u, \quad (24.17)$$

where \mathbf{N} is the unit normal vector perpendicular to the implicit surface and τ is fictitious time. A similar equation holds for the v and w components of the velocity field. Fast methods exist for solving this equation in $O(n \log n)$ time, where n is the number of grid points that one needs to extrapolate over, in our case a five grid cell thick band on the air side of the interface. The fast method is based upon the observation that information in equation 24.17 propagates in only one direction away from the surface. This implies that one does not have to revisit previously computed values of \mathbf{u}_{ext} (the extrapolated velocity) if the calculation is preformed in the correct order. The order is determined by the value of ϕ allowing an $O(n \log n)$ sorting of the points before beginning the calculation. The value of u itself is determined by enforcing the condition at steady state, namely $\nabla \phi \cdot \nabla u = 0$ where the derivatives are determined using previously calculated values of ϕ and u . From this scalar equation, a new value of u can be determined, and then one can proceed to the next point corresponding to the next smallest value of ϕ , etc. Further details of this method are discussed in [4].

This velocity extrapolation method enabled [180] to apply a newly devised method for controlling the nature of the surface motion. This was done simply by modifying the extrapolated velocities on the air side of the surface. For example, to model wind-blown water as a result of air drag, one takes a convex combination of the extrapolated velocities with a pre-determined wind velocity field

$$\mathbf{V} = (1 - \alpha) \mathbf{V}_{ext} + \alpha \mathbf{V}_{wind}, \quad (24.18)$$

where \mathbf{V}_{ext} is the extrapolated velocity, \mathbf{V}_{wind} a desired air-like velocity, and $0 \leq \alpha \leq 1$ the mixing constant. This can be applied throughout the surface or only locally in select portions of the computational domain as desired. Note that setting $\mathbf{V}_{wind} = 0$ forces churning water to settle down faster with the fastest settling resulting from $\alpha = 1$.

24.4 Fire

The modeling of natural phenomena such as fire and flames remains a challenging problem in computer graphics. Although simulations of fluid behavior are in demand for special effects, fire effects are especially in demand due to the dangerous nature of this phenomenon. Fire simulations are also of interest for virtual reality effects, for example to help train fire fighters or to determine proper placement of exit signs in smoke filled rooms (i.e. so they can be seen). Combustion processes can be loosely classified into two rather distinct types of phenomena: detonations and deflagrations. In both of these processes, chemical reactions convert fuel into hot gaseous products. Deflagrations are low speed events such as the fire and flames we address in this section, while detonations are high speed events such as explosions where shock waves and other compressible effects are important. As low speed events, deflagrations can be modeled using the equations for incompressible flow (as opposed to those for compressible flow). Furthermore, since viscous effects are small, the incompressible inviscid Euler equations can be used.

An important, often neglected aspect of fire and flame modeling concerns the expansion of the fuel as it reacts to form hot gaseous products. This expansion is the reason for the visual fullness observed in many flames and is partly responsible for the visual turbulence as well. Since the incompressible equations do not account for expansion, [391] proposed a simple thin flame model for capturing these effects. This was accomplished by using an implicit surface to represent the reaction zone where the gaseous fuel is converted into hot gaseous products. Although real reaction zones have a nonzero (but small) thickness, the thin flame approximation works well for visual modeling and has been used by scientists as well, see for example [340] which first proposed this methodology. [391] used a dynamic implicit surface to track the reaction zone where the gaseous fuel is converted into hot gaseous products. Then both the gaseous fuel and the hot gaseous products were separately modeled using independent sets of incompressible flow equations. Finally, these incompressible flow equations were updated together in

a coupled fashion using the fact that both mass and momentum must be conserved as the gas reacts at the interface. While this gives rather pleasant looking laminar (smooth) flames, a vorticity confinement term was included to model the larger scale turbulent flame structures that are difficult to capture on the relatively coarse grids.

There are three basic visual phenomena associated with flames. The first of these is the blue or bluish-green core seen in many flames. These colors are emission lines from intermediate chemical species, such as carbon radicals, produced during the chemical reaction. In the thin flame model, this thin blue core is located adjacent to the implicit surface. Therefore, in order to track this blue core, one needs to track the movement of the implicit surface. The second visual phenomenon is the blackbody radiation emitted by the hot gaseous products, in particular the carbon soot. This is characterized by the yellowish-orange color familiarly associated with fire. In order to model this with visual accuracy one needs to track the temperatures associated with a flame. If the fuel is solid or liquid, the first step is the heating of the solid until it undergoes a phase change to the gaseous state. (Obviously, for gas fuels, we start in this gaseous state.) Then the gas heats up until it reaches its ignition temperature corresponding to the implicit surface and the beginning of the thin blue core region. The temperature continues to increase as the reaction proceeds reaching a maximum before radiative cooling and mixing effects cause the temperature to decrease. As the temperature decreases, the blackbody radiation falls off until the yellowish-orange color is no longer visible. The third and final visual effect addressed is the smoke or soot that is apparent in some flames after the temperature cools to the point where the blackbody radiation is no longer visible. This effect is modeled by carrying along a density variable in a fashion similar to the temperature.

In [391] the implicit surface separates the gaseous fuel from the hot gaseous products and surrounding air. Consider for example the injection of gaseous fuel from a cylindrically shaped tube. If the fuel were not burning, then the implicit surface would simply move at the same velocity as the gaseous fuel being injected. However, when the fuel is reacting, the implicit surface moves at the velocity of the unreacted fuel plus a flame speed S that indicates how fast fuel is being converted into gaseous products. S indicates how fast the unreacted gaseous fuel is crossing over the implicit surface turning into hot gaseous products. The approximate surface area of the blue core, A_S , can be estimated with the following equation

$$v_f A_f = S A_S, \quad (24.19)$$

where v_f is the speed the fuel is injected across the injection surface with area A_f , e.g. A_f is the cross section of the cylindrical tube. This equation results from canceling out the density in the equation for conservation of mass. The left hand side is the fuel being injected into the region bounded by the implicit surface, and the right hand side is the fuel leaving this region crossing over the implicit surface as it turns into gaseous products. From this equation, we see that injecting more (less) gas is equivalent to increasing (decreasing) v_f resulting in a larger

(smaller) blue core. Similarly, increasing (decreasing) the reaction speed S results in a smaller (larger) blue core. While one can turn the velocity up or down on a cylindrical jet, the reaction speed S is a property of the fuel. For example, S is approximately $.44\text{m/s}$ for a propane fuel that has been suitably premixed with oxidizer [538]. (Note that while this thin flame approximation is fairly accurate for premixed flames, diffusion flames behave somewhat differently.)

In order to get the proper visual look for flames, it is important to track individual elements of the flow and follow them through their temperature histories. This is particularly difficult because the gas expands as it undergoes reaction from fuel to hot gaseous products. This expansion is important to model since it changes the trajectories of the gas and the subsequent look and feel of the flame as individual elements go through their temperature profile. Individual elements do not go straight up as they pass through the reaction front, but instead turn outward due to the effects of expansion. It is difficult to obtain visually full turbulent flames without modeling this expansion effect. In fact, many practitioners resort to a number of low level hacks (and lots of random numbers) in an attempt to sculpt this behavior. In contrast, [391] obtained this behavior by modeling the expansion directly.

[391] used one set of incompressible flow equations to model the fuel with density ρ_f and a separate set of incompressible flow equations to model the hot gaseous products and surrounding airflow with density ρ_h . This requires a model for coupling these two sets of incompressible flow equations together across the interface in a manner that models the expansion that takes place across the reaction front. Given that mass and momentum are conserved one can derive the following equations for the coupling across the thin flame front:

$$\rho_h(V_h - D) = \rho_f(V_f - D), \quad (24.20)$$

$$\rho_h(V_h - D)^2 + p_h = \rho_f(V_f - D)^2 + p_f, \quad (24.21)$$

where V_f and V_h are the normal velocities of the fuel and the hot gaseous products, and p_f and p_h are their pressures. Here, $D = V_f - S$ is the speed of the implicit surface in the normal direction. These equations indicate that both the velocity and the pressure are discontinuous across the flame front. Thus, one needs to exercise caution when taking derivatives of these quantities as is required when solving the incompressible flow equations. (Note that the tangential velocities are continuous across the flame front.)

When considering solid fuels, there are two expansions that need to be accounted for. Besides the expansion across the flame front, a similar expansion takes place when the solid is converted to a gas. However, S is usually relatively small for this reaction (most solids burn slowly in a visual sense), so we can use the boundary of the solid fuel as the reaction front. Since [391] did not model the pressure in solids, only equation 24.20 applies. This equation can be rewritten as

$$\rho_f(V_f - D) = \rho_s(V_s - D), \quad (24.22)$$

where ρ_s and V_s are the density and the normal velocity of the solid fuel. Substituting $D = V_s - S$ and solving for V_f gives

$$V_f = V_s + (\rho_s / \rho_f - 1) S \quad (24.23)$$

indicating that the gasified solid fuel moves at the velocity of the solid fuel plus a correction that accounts for the expansion. [391] modeled this phase change by injecting gas out of the solid fuel at the appropriate velocity. This can be used to set arbitrary shaped solid objects on fire as long as they can be voxelized with a suitable surface normal assigned to each voxel indicating the direction of gaseous injection. The ability to inject (or not inject) gaseous fuel out of individual voxels on the surface of a complex solid object allows one to animate objects catching on fire, burn different parts of an object at different rates or not at all (by using spatially varying injection velocities), and extinguish solid fuels simply by turning off the injection velocity. Figure 24.6 shows a ball catching on fire as it passes through a flame.

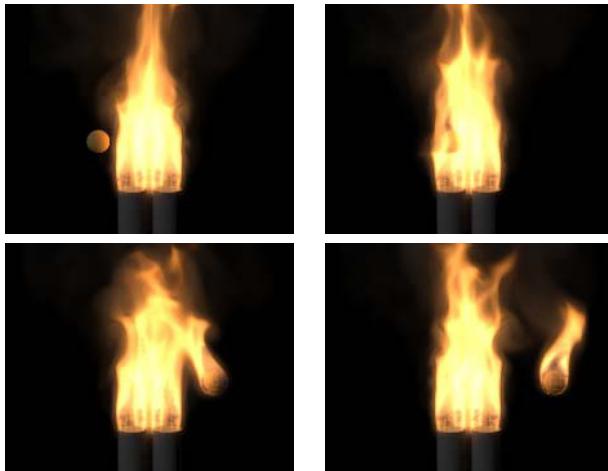


Figure 24.6. A flammable ball passes through a gas flame and catches on fire.

[391] tracked the reaction zone (blue core) using the level set method to track the moving implicit surface with ϕ positive in the region of space filled with fuel, negative elsewhere and zero at the reaction zone. The implicit surface moves with velocity $\mathbf{W} = \mathbf{V}_f + S\mathbf{N}$ where \mathbf{V}_f is the velocity of the gaseous fuel and the $S\mathbf{N}$ term governs the conversion of fuel into gaseous products. Standard averaging of voxel face values is used to define \mathbf{V}_f at the voxel centers. The motion of the implicit surface is defined through $\phi_t = -\mathbf{W} \cdot \nabla \phi$.

When solving the two sets of incompressible Euler equations, the equations for the velocity

$$\mathbf{V}_t = -(\mathbf{V} \cdot \nabla) \mathbf{V} - \nabla p / \rho + \mathbf{f} \quad (24.24)$$

are solved for in two parts. First, this equation is used to compute an intermediate velocity \mathbf{V}^* ignoring the pressure term, and then pressure (correction) term is

added using

$$\mathbf{V} = \mathbf{V}^* - \Delta t \nabla p / \rho. \quad (24.25)$$

The key idea to this splitting method is illustrated by taking the divergence of equation 24.25 to obtain

$$\nabla \cdot \mathbf{V} = \nabla \cdot \mathbf{V}^* - \Delta t \nabla \cdot (\nabla p / \rho) \quad (24.26)$$

and then realizing that we want $\nabla \cdot \mathbf{V} = 0$ to enforce mass conservation. Thus the left hand side of equation 24.26 should vanish leaving a Poisson equation of the form

$$\nabla \cdot (\nabla p / \rho) = \nabla \cdot \mathbf{V}^* / \Delta t \quad (24.27)$$

that can be solved to find the pressure needed for updating equation 24.25.

Since there are two sets of incompressible flow equations, we need to address the semi-Lagrangian update when a characteristic traced back from one set of incompressible flow equations crosses the implicit surface and queries the velocities from the other set of incompressible flow equations. Since the normal velocity is discontinuous across the interface, the straightforward approach fails. Instead, we need to use the balance equation 24.20 for conservation of mass to correctly interpolate a velocity. Suppose we are solving for the hot gaseous products and we interpolate across the interface into a region where a velocity from the gaseous fuel might incorrectly be used. Instead of using this value, we compute a ghost value as follows. First, we compute the normal velocity of the fuel, $V_f = \mathbf{V}_f \cdot \mathbf{N}$. Then we use the balance equation 24.20 to find a ghost value for V_h^G as

$$V_h^G = V_f + (\rho_f / \rho_h - 1) S. \quad (24.28)$$

Since the tangential velocities are continuous across the implicit surface, we combine this new normal velocity with the existing tangential velocity to obtain

$$\mathbf{V}_h^G = V_h^G \mathbf{N} + \mathbf{V}_f - (\mathbf{V}_f \cdot \mathbf{N}) \mathbf{N} \quad (24.29)$$

as a ghost value for the velocity of the hot gaseous products in the region where only the fuel is defined. This ghost velocity can then be used to correctly carry out the semi-Lagrangian update. Since both \mathbf{N} and \mathbf{V}_f are defined throughout the region occupied by the fuel, and ρ_f , ρ_h and S are known constants, a ghost cell value for the hot gaseous products, \mathbf{V}_h^G , can be found anywhere in the fuel region (even quite far from the interface) by simply algebraically evaluating the right hand side of equation 24.29. [392] showed that this ghost fluid method, invented in [192], could be used to compute physically accurate engineering simulations of deflagrations.

After computing the intermediate velocity \mathbf{V}^* for both sets of incompressible flow equations, equation 24.27 is solved for the pressure and finally equation 24.25 is used to find the new velocity field. Equation 24.27 is solved by assembling and solving a linear system of equations for the pressure as discussed in more detail in [198]. Once again, caution needs to be exercised here since the pressure is discontinuous across the interface. Using the ghost fluid method and

equation 24.21, one can obtain and solve a slightly modified linear system incorporating this jump in pressure. We refer the reader to [392] for explicit details and a demonstration of the physical accuracy of this approach in the context of deflagration waves.

The temperature profile has great effect on how we visually perceive flames. Thus, we need a way to track individual fluid elements as they cross over the blue core and rise upward due to buoyancy. That is, we need to know how much time has elapsed since a fluid element has passed through the blue core so that we can assign an appropriate temperature to it. This is easily accomplished using a reaction coordinate variable Y governed by the equation

$$Y_t = -(\mathbf{V} \cdot \nabla) Y - k, \quad (24.30)$$

where k is a positive constant which we take to be 1 (larger or smaller values can be used to get a good numerical variation of Y in the flame). Ignoring the convection term, $Y_t = -1$ can be solved exactly to obtain $Y(t) = -t + Y(0)$. If we set $Y(0) = 1$ in the region of space occupied by the gaseous fuel and solve equation 24.30 for Y , then the local value of $1 - Y$ is equal to the total time elapsed since a fluid element crossed over the blue reaction core. We can now use the values of Y to assign temperature values to the flow. The animator can sculpt both the temperature rise and the temperature falloff for a flame based on Y . However, for the temperature falloff, there is a physically correct, viable (i.e. computationally cheap) alternative. For the values of Y in the temperature falloff region, we simply solve

$$T_t = -(\mathbf{V} \cdot \nabla) T - c_T \left(\frac{T - T_{air}}{T_{max} - T_{air}} \right)^4 \quad (24.31)$$

which is derived from conservation of energy. This equation is solved by first using the semi-Lagrangian stable fluids method to solve for the convection term. Then the fourth power term is analytically integrated to cool down the flame at a rate governed by the cooling constant c_T . Similar to the temperature, the animator can sculpt a density curve for smoke and soot formation. The density should start low and increase as the reaction proceeds. In the temperature falloff region, the animator can switch from the density curve to a physically correct equation

$$\rho_t = -(\mathbf{V} \cdot \nabla) \rho \quad (24.32)$$

that can (once again) be solved using the semi-Lagrangian stable fluids method.

Bibliography

References

- [1] A. Acart and C. Vogel. Analysis of bounded variation penalty methods of ill-posed problems. *Inverse Problem*, 10:1217–1229, 1994.
- [2] D. Adalsteinsson and J. Sethian. A Fast Level Set Method for Propagating Interfaces. *Journal of Computational Physics*, 118:269–277, 1995.
- [3] D. Adalsteinsson and J. Sethian. The fast construction of extension velocities in level set methods. Technical Report PAM-738, Center for Pure and Applied Mathematics. University of California at Berkeley, 1997.
- [4] D. Adalsteinsson and J. Sethian. The Fast Construction of Extension Velocities in Level Set Methods. *Journal of Computational Physics*, 148:2–22, 1999.
- [5] M. Adams, A. Willsky, and B. Levy. Linear estimation of boundary value stochastic processes—part 1: The role and construction of complementary models. *IEEE Transactions Automatic Control*, 29:803–811, 1984.
- [6] R. Adams and L. Bischof. Seeded Region Growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:641–647, 1994.
- [7] J. Aggarwal and N. Nandhakumar. On the computation of motion from sequences of images - A Review. *Proceedings of the IEEE*, pages 917–935, 1988.
- [8] F. Alouges. An energy decreasing algorithm for harmonic maps. In J. Coron, J. Ghidaglia, and F. Helein, editors, *Nematics*, Nato ASI Series, pages 1–13. Kluwer Academic Publishers, Netherlands, 1991.
- [9] L. Alvarez, F. Guichard, P.-L. Lions, and J.-M. Morel. Axioms and Fundamental Equations of Image Processing. *Archives for Rational Mechanics and Analysis*, 16:199–257, 1993.
- [10] L. Alvarez, P.-L. Lions, and J.-M. Morel. Image selective smoothing and edge detection by nonlinear diffusion. *SIAM Journal on Numerical Analysis*, 29:845–866, 1992.
- [11] O. Amadieu, E. Debreuve, M. Barlaud, and G. Aubert. Inward and Outward Curve Evolution Using Level Set Method. In *IEEE International Conference on Image Processing*, volume III, pages 188–192, 1999.
- [12] L. Ambrosio. A compactness theorem for a special class of functions of bounded variation. *Bollettino dell'Unione Matematica Italiana*, 3(B):857–881, 1989.
- [13] L. Ambrosio and V. Tortorelli. Approximation of functionals depending on jumps by elliptic functionals via Gamma-convergence. *Communication Pure Applied Mathematics*, pages 999–1036, 1990.

- [14] L. Ambrosio and V. Tortorelli. On the Approximation of Free Discontinuity Problems. *Bollettino dell'Unione Matematica Italiana*, 6-B(7):105–123, 1992.
- [15] N. Amenta and M. Bern. Surface reconstruction by Voronoi filtering. In *14th ACM Symposium on Computational Geometry*, pages 39–48, 1998.
- [16] N. Amenta, M. Bern, and D. Eppstein. The crust and the β -skeleton: combinatorial curve reconstruction. *Graphical Models and Image Processing*, 60/2(2):125–135, 1998.
- [17] N. Amenta, M. Bern, and M. Kamvysselis. A new Voronoi-based surface reconstruction algorithm. In *ACM SIGGRAPH*, pages 415–421, 1998.
- [18] F. Andreu, C. Ballester, V. Caselles, and J.M. Mazón. The Dirichlet Problem for the Total Variation Flow. *Journal of Functional Analysis*, 180:347–403, 2001.
- [19] F. Andreu, C. Ballester, V. Caselles, and J.M. Mazón. Minimizing Total Variation Flow. *Differential and Integral Equations*, 14(3):321–360, 2001.
- [20] S. Angenent, S. Haker, A. Tannenbaum, and R. Kikinis. Laplace-Beltrami operator and brain flattening. Technical report, University of Minnesota ECE Report, 1998.
- [21] V. Arnold. *Mathematical Methods of Classical Mechanics*. Springer Verlag, 1978.
- [22] G. Aubert, M. Barlaud, P. Charbonnier, and L. Blanc-Féraud. Deterministic edge preserving regularization in computed imaging. Technical Report 94-01, I3S, CNRS URA 1376, Sophia-Antipolis, France, 1994.
- [23] G. Aubert, M. Barlaud, O. Faugeras, and S. Jehan-Besson. Image segmentation using active contours: calculus of variations of shape gradients? Technical Report 4483, INRIA, France, 2002.
- [24] G. Aubert and P. Kornprobst. *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations*. Springer-Verlag, 2001.
- [25] R. Azencott, F. Coldefy, and L. Younes. A distance for elastic matching in object recognition. In *IARP International Conference on Pattern Recognition*, volume 1, pages 687–691, 1996.
- [26] C. Bajaj, F. Bernardini, and G. Xu. Automatic Reconstruction of Surfaces and Scalar Fields from 3D Scans. In *ACM SIGGRAPH*, pages 109–118, August 1995.
- [27] C. Ballester, E. Cubero-Castan, M. González, and J.-M. Morel. Contrast Invariant Image Intersection. Preprint CEREMADE, 1998.
- [28] M. Bardi and S. Osher. The Nonconvex Multi-Dimensional Riemann Problem for Hamilton-Jacobi Equations. *SIAM Journal on Mathematical Analysis*, 22(2):344–351, 1991.
- [29] G. Barles. *Solutions de Viscosité des Équations de Hamilton-Jacobi*. Springer-Verlag, 1996.
- [30] A. Barr. Superquadrics and Angle-Preserving Transformations. *IEEE Computer Graphics and Applications*, 1:11–23, 1981.
- [31] J. Barron, D. Fleet, and S. Beauchemin. Performance of Optical Flow Techniques. *International Journal of Computer Vision*, 12:43–77, 1994.
- [32] B. Basile and R. Deriche. Region Tracking through Image Sequences. In *IEEE International Conference in Computer Vision*, pages 302–307, 1995.
- [33] K.-J. Bathe and E. Wilson. *Numerical Methods in Finite Element Analysis*. Prentice-Hall, 1976.

- [34] S. Belongie, J. Malik, and J. Puzicha. Matching Shapes. In *IEEE International Conference in Computer Vision*, pages 456–461, 2001.
- [35] D. Bereziat, I. Herlin, and L. Younes. Motion detection in meteorological images sequences: Two methods and their comparison. In *SPIE*, volume 3127, pages 332–341, 1997.
- [36] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani. Hierarchical Model-Based Motion Estimation. In *European Conference on Computer Vision*, pages 237–252, 1992.
- [37] M. Bertalmio, A. Bertozzi, and G. Sapiro. Navier-Stokes, Fluid Dynamics, and Image and Video Inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 355–362, 2001.
- [38] M. Bertalmio, L. Cheng, S. Osher, and G. Sapiro. Variational Problems and Partial Differential Equations on Implicit Surfaces. *Journal of Computational Physics*, 174(2):759–780, 2001.
- [39] M. Bertalmio, G. Sapiro, L.-T. Cheng, and S. Osher. Image Inpainting. In *ACM SIGGRAPH*, pages 417–424, 2000.
- [40] M. Bertalmio, G. Sapiro, L.-T. Cheng, and S. Osher. Variational Problems and PDE's on Implicit Surfaces. In *IEEE Workshop in Variational and Level Set Methods*, pages 186–193, 2001.
- [41] M. Bertalmio, G. Sapiro, and G. Randall. Morphing Active Contours. In *International Conference on Scale-Space Theories in Computer Vision*, pages 46–57, 1999.
- [42] M. Bertalmio, G. Sapiro, and G. Randall. Region Tracking on Level Set Methods. *IEEE Transactions on Medical Imaging*, 18:448–451, 1999.
- [43] G. Bertrand. Simple Points, Topological Numbers and Geodesic Neighborhoods in Cubic Grids. *Pattern Recognition Letters*, 15:1003–1011, 1994.
- [44] G. Bertrand, J. Everat, and M. Couprie. Image Segmentation through Operators Based on Topology. *Journal of Electronic Imaging*, 6:395–405, 1997.
- [45] J. Besag. On the statistical analysis of dirty images. *Journal of Royal Statistics Society*, 48:259–302, 1986.
- [46] H. Biermann, D. Kristjansson, and D. Zorin. Approximate Boolean Operations on Free-form Solids. In *ACM SIGGRAPH*, pages 185–194, August 2001.
- [47] B. Biswal and J. Hyde. Contour-based registration technique to differentiate between task-activation and head motion induced signal variations in fMRI. *Magnetic Resonance in Medicine*, 38:470–476, 1997.
- [48] M. Black and A. Jepson. Estimating optical flow in segmented images using variable-order parametric models with local deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:973–986, 1996.
- [49] M. Black, G. Sapiro, D. Marimont, and D. Heeger. Robust anisotropic diffusion. *IEEE Transactions on Image Processing*, 7:421–432, 1998.
- [50] A. Blake and M. Isard. *Active Contours*. Springer-Verlag, 1997.
- [51] A. Blake and C. Marinos. Shape from Texture: Estimation, Isotropy and Moments. *Artificial Intelligence*, 45(3):323–380, 1990.
- [52] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, 1987.

- [53] P. Blomgren and T. Chan. Color TV : Total Variation Methods for Restoration of Vector Valued Images. *IEEE Transactions on Image Processing*, 7(3):304–309, 1998.
- [54] P. Blomgren, T. Chan, P. Mulet, and C. Wong. Total Variation Image Restoration: Numerical Methods and Extensions. In *IEEE International Conference on Image Processing*, pages III, 384–387, 1997.
- [55] J. Bloomenthal, C. Bajaj, J. Blinn, M.-P. Cani-Gascuel, A. Rockwood, B. Wyvill, and G. Wyvill. *Introduction to Implicit Surfaces*. Morgan Kaufman, 1997.
- [56] M. Bloor and M. Wilson. Using partial differential equations to generate free-from surface. *Computer Aided Design*, pages 257–266, 1990.
- [57] D. Blostein and N. Ahuja. Shape from Texture: Integrating Texture-Element Extraction and Surface Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12):1233–1251, 1989.
- [58] J.-D. Boissonnat. Geometric structures for three dimensional shape reconstruction. *ACM Transactions in Graphics*, pages 266–286, 1984.
- [59] J.-D. Boissonnat and F. Cazals. Smooth shape reconstruction via natural neighbor interpolation of distance functions. In *ACM Symposium on Computational Geometry*, pages 223–232, 2000.
- [60] M. Boué and P. Dupuis. Markov Chain Approximations for Deterministic Control Problem with Affine Dynamics and Quadratic Cost in the Control. *SIAM Journal in Numerical Analysis*, 36(3):667–695, 1999.
- [61] J.-Y. Bouguet and P. Perona. 3D Photography Using Shadow in Dual Space Geometry. *International Journal of Computer Vision*, 35(2):129–149, 1999.
- [62] B. Bourdin. Image segmentation with a finite element method. *Mathematical Modelling and Numerical Analysis*, 33(2):229–244, 1999.
- [63] B. Bourdin and A. Chambolle. Implementation of an adaptive finite-element approximation of the Mumford-Shah functional. *Numerische Mathematik*, 85(4):609–646, 2000.
- [64] E. Boyer and M. Berger. 3D surface reconstruction using occluding contours. *International Journal of Computer Vision*, 22:219–233, 1997.
- [65] A. Braides. *Approximation of Free-Discontinuity Problems*. Lecture Notes in Mathematics No. 1694, Springer-Verlag, 1998.
- [66] D. Breen, S. Mauch, and R. Whitaker. 3D Scan Conversion of CSG Models into Distance, Closest-Point and Colour Volumes. In M. Chen, A. Kaufman, and R. Yagel, editors, *Volume Graphics*, pages 135–158. Springer-Verlag, 2000.
- [67] D. Breen and R. Whitaker. A Level Set Approach for the Metamorphosis of Solid Models. *IEEE Transactions on Visualization and Computer Graphics*, 7(2):173–192, 2001.
- [68] H. Brezis, J.-M. Coron, and E. Lieb. Harmonic maps with defects. *Communications in Mathematical Physics*, 107:649–705, 1986.
- [69] W. Briggs, V. Henson, and S. McCormick. *A Multigrid Tutorial*. SIAM, 2000.
- [70] R. Brockett and P. Maragos. Evolution Equations for Continuous-Scale Morphological Filtering. *IEEE Transactions on Signal Processing*, 42:3377–3386, 1994.

- [71] G. Brown. A Survey of Image Registration Techniques. *ACM Computing Surveys*, 24:325–376, 1992.
- [72] A. Bruhn, T. Jacob, M. Fischer, T. Kohlberger, J. Weickert, U. Brüning, and C. Schnorr. Designing 3-D nonlinear diffusion filters for high performance cluster computing. In L. Van Gool, editor, *Pattern Recognition*. Springer-Verlag, 2002.
- [73] P. Burchard, S. Chen, S. Osher, and H.K. Zhao. Surface Reconstruction From Normals. Technical report, Level Set Systems, 2000.
- [74] P. Burchard, L.-T. Cheng, B. Merriman, and S. Osher. Motion of Curves in Three Spatial Dimensions Using a Level Set Approach. *Journal of Computational Physics*, 170:720–741, 2001.
- [75] B. Cabral and C. Leedom. Imaging vector fields using line integral convolution. In *ACM SIGGRAPH*, pages 263–272, 1993.
- [76] E. Calabi, P. Olver, and A. Tannenbaum. Affine geometry, curve flows, and invariant numerical approximations. Technical report, University of Minnesota, 1995.
- [77] J. Canny. A computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:769–798, 1986.
- [78] I. Carlbom, D. Terzopoulos, and K. Harris. Computer-Assisted Registration, Segmentation, and 3D Reconstruction from Images of Neuronal Tissue Sections. *IEEE Transactions on Medical Imaging*, 13:351–362, 1994.
- [79] T. Carne. The geometry of shape spaces. *Proceedings of the London Mathematics Society*, 3:407–432, 1990.
- [80] J. Carr, R. Beatson, J. Cherrie, T. Mitchell, W. Fright, B. McCallum, and T. Evans. Reconstruction and Representation of 3D Objects with Radial Basis Functions. In *ACM SIGGRAPH*, pages 67–76, 2001.
- [81] V. Caselles, F. Catté, B. Coll, and F. Dibos. A geometric model for active contours in image processing. *Numerische Mathematik*, 66(1):1–31, 1993.
- [82] V. Caselles and B. Coll. Snakes in Movement. *SIAM Journal on Numerical Analysis*, 33:2445–2456, 1996.
- [83] V. Caselles, B. Coll, and J.-M. Morel. Topographic Maps and Local Contrast Changes in Natural Images. *International Journal of Computer Vision*, 33(1):5–27, 1999.
- [84] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic Active Contours. In *IEEE International Conference in Computer Vision*, pages 694–699, 1995.
- [85] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic Active Contours. *International Journal of Computer Vision*, 22:61–79, 1997.
- [86] V. Caselles, R. Kimmel, G. Sapiro, and C. Sbert. 3D Active Contours. In *International Conference on Analysis and Optimization of Systems*, pages 43–49, 1996.
- [87] V. Caselles, R. Kimmel, G. Sapiro, and C. Sbert. Minimal surfaces based object segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:394–398, 1997.
- [88] V. Caselles, J.-M. Morel, G. Sapiro, and A. Tannenbaum. Introduction to the special issue on PDEs and geometry-driven diffusion in image processing and analysis. *IEEE Transactions on Image Processing*, 7(3):269–274, 1998.

- [89] E. Catmull.*A Subdivision Algorithm for Computer Display of Curved Surfaces*.PhD thesis, Department of Computer Science, University of Utah, 1974.
- [90] F. Catté, F. Dibos, and G. Koepfler.A morphological scheme for mean curvature motion and applications to anisotropic diffusion and motion of level sets.*SIAM Journal on Numerical Analysis*, 32:1895–1909, 1995.
- [91] G. Celtniker and D. Gossard.Deformable curve and surface finite element for free-form shape design.*Computer Graphics*, pages 257–266, 1991.
- [92] A. Chambolle.Un théorème de γ -convergence pour la segmentation des signaux.*C. R. Académie de Science, Paris, Sér. I*, 314:191–196, 1992.
- [93] A. Chambolle.Image segmentation by variational methods: Mumford and Shah functional and the discrete approximations.*SIAM Journal in Applied Mathematics*, 55:827–863, 1995.
- [94] A. Chambolle.Finite-differences discretizations of the Mumford-Shah functional.*Mathematical Modelling and Numerical Analysis Modelisation*, 33:261–288, 1999.
- [95] A. Chambolle and G. Dal Maso.Discrete approximation of the Mumford-Shah functional in dimension two.*Mathematical Modelling and Numerical Analysis Modelisation*, 33:651–672, 1999.
- [96] A. Chambolle and P.-L. Lions.Image Recovery via Total Variation Minimization and Related Problems.*Numerische Mathematik*, 76:167–188, 1997.
- [97] T. Chan, B. Sandberg, and L. Vese.Active Contours without Edges for Vector-Valued Images.*Journal of Visual Communication and Image Representations*, 2:130–141, 2000.
- [98] T. Chan and J. Shen.Variational restoration of non-flat image features: Models and algorithms.Techanical Report 9920, UCLA-CAM, 1999.
- [99] T. Chan and D. Strong.Edge-preserving and Scale-Dependent Properties of Total Variation Regularization.Techical Report 0038, UCLA-CAM, 1999.
- [100] T. Chan and L. Vese.An Active Contour Model without Edges.In *International Conference on Scale-Space Theories in Computer Vision*, pages 141–151, 1999.
- [101] T. Chan and L. Vese.Active Contours without Edges.*IEEE Transactions on Image Processing*, 10:266–277, 2001.
- [102] T. Chan and L. Vese.A Level Set Algorithm for Minimizing the Mumford-Shah Functional in Image Processing.In *IEEE Workshop on Variational and Level Set Methods*, pages 161–168, 2001.
- [103] T. Chan and L. Vese.Active Contour and Segmentation Models Using Geometric PDE's for Medical Imaging.In R. Malladi, editor, *Geometric Methods in Bio-Medical Image Processing*. Springer-Verlag, 2002.
- [104] S. Chang, F. Cheng, W. Hsu, and Z. Wu.Fast algorithm for point pattern matching: Invariant to translations, rotations and scale changes.*Pattern Recognition*, 30(2):311–320, 1997.
- [105] C. Chefd'Hotel, G. Hermosillo, and O. Faugeras.A Variational Approach to Multi-Modal Image Matching.In *IEEE Workshop in Variational and Level Set Methods*, pages 21–28, 2001.
- [106] M. Chen, T. Kanade, H. Rowley, and D. Pomerleau.Quantitative Study of Brain Anatomy.In *IEEE Workshop on Biomedical Image Analysis*, pages 84–92, 1998.

- [107] S. Chen, D. Johnson, and P. Raad. Velocity Boundary Conditions for the Simulation of Free Surface Fluid Flow. *Journal of Computational Physics*, 116:262–276, 1995.
- [108] S. Chen, B. Merriman, S. Osher, and P. Smereka. A simple level set method for solving Stefan problems. *Journal of Computational Physics*, 135:8–29, 1995.
- [109] Y. Chen, Y. Giga, and S. Goto. Uniqueness and existence of viscosity solutions of generalized mean curvature flow equation. *Journal of Differential Geometry*, 33:749–786, 1991.
- [110] Y. Chen, M. Hong, and N. Hungerbuhler. Heat flow of p-harmonic maps with values into sphere. *Mathematische Zeitschrift*, 205:25–35, 1994.
- [111] Y. Chen, H. Thiruvenkadam, H. Tagare, F. Huang, and D. Wilson. On the Incorporation of Shape Priors int Geometric Active Contours. In *IEEE Workshop in Variational and Level Set Methods*, pages 145–152, 2001.
- [112] Y. Chen, S. Thiruvenkadam, K. Gopinath, and R. Brigg. FMR Image Registration Using the Mumford-Shah Functional and Shape Information. In *World Multiconference on Systems, Cybernetics and Informatics*, pages 580–583, 2002.
- [113] Y. Chen, S. Thiruvenkadam, F. Huang, K. Gopinath, and R. Brigg. FMR Image Registration Using the Mumford-Shah Functional and Shape Information. In *IARP International Conference on Pattern Recognition*, pages 747–750, 2002.
- [114] L.-T. Cheng. *The Level Set Method Applied to Geometrically Based Motion, Material Science, and Image Processing*. PhD thesis, Department of Mathematics, UCLA, 2000.
- [115] L.-T. Cheng, P. Burchard, B. Merriman, and S. Osher. Motion of Curves and Constrained on Surfaces Using a Level Set Approach. *Journal of Computational Physics*, 175:604–644, 2002.
- [116] C. Chiang, C. Hoffmann, and R. Lync. How to compute offsets without self-intersection. In *SPIE*, volume 1620, page 76, 1992.
- [117] D. Chopp. *Computing Minimal Surfaces via Level Set Curvature Flow*. PhD thesis, Department of Mathematics, Unibersity of California, Berkeley, 1991.
- [118] D. Chopp. Computing Minimal Surfaces via Level Set Curvature Flow. *Journal of Computational Physics*, 106:77–91, 1993.
- [119] P. Chou and C. Brown. The theory and practice of bayesian image labeling. *International Journal of Computer Vision*, 4:185–210, 1990.
- [120] G. Christensen, M. Miller, and M. Vannier. Individualizing neuroanatomical atlases using a massively parallel computer. *IEEE Computer*, 1:32–38, 1996.
- [121] H. Chui and A. Rangarajan. A New Algorithm for Non-Rigid Point Matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages II: 44–51, 2000.
- [122] R. Cipolla and A. Blake. Surface shape from deformation of apparent contours. *International Journal of Computer Vision*, 9:83–112, 1992.
- [123] M. Clerc. Wavelet-Based Correlation for Stereopsis. In *European Conference on Computer Vision*, pages 495–509, 2002.
- [124] M. Clerc and S. Mallat. Shape from Texture through Deformations. In *IEEE International Conference in Computer Vision*, pages 405–410, 1999.

- [125] E. Cohen, R. Riesenfeld, and G. Elber.*Geometric Modeling with Splines*.AK Peters, 2001.
- [126] I. Cohen and I. Herlin.Curve Matching Using Geodesic Paths.In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 741–746, 1998.
- [127] L. Cohen.On active contour models and balloons.*CVGIP: Image Understanding*, 53:211–218, 1991.
- [128] R. Cohen, R. Hardt, D. Kinderlehrer, S. Lin, and M. Luskin.Minimum energy configurations for liquid crystals: Computational results.In J. Ericksen and D. Kinderlehrer, editors, *Theory and Applications of Liquid Crystals*, pages 99–121. Kluwer Academic Publishers, 1987.
- [129] T. Cohignac, C. Lopez, and J.-M. Morel.Integral and Local Affine Invariant Parameter and Application to Shape Recognition.In *IARP International Conference on Pattern Recognition*, pages 164–168, 1994.
- [130] E. Cole.The Removal of Unknown Image Blurs by Homomorphic Filtering.Techical Report 0038, Department of Electrical Engineering, University of Toronto, 1973.
- [131] A. Collignon, F. Maes, D. Vandermeulen, P. Suetens, and G. Marchal.Automated multimodality image registration using information theory.In *Information Processing in Medical Imaging*, pages 263–274, 1995.
- [132] D. Comaniciu, V. Ramesh, and P. Meer.Real time tracking of non-rigid objects using Mean Shift.In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 142–149, 2000.
- [133] T. Cootes, C. Beeston, G. Edwards, and C. Taylor.Unified Framework for Atlas Matching Using Active Appearance Models.In *Information Processing in Medical Imaging*, pages 322–333, 1999.
- [134] T. Cootes and C. Taylor.Mixture model for representing shape variation.*Image and Vision Computing*, 17:567–574, 1999.
- [135] T. Cootes, C. Taylor, D. Cooper, and J. Graham.Active shape models - their training and application.*Computer Vision and Image Understanding*, 61:38–59, 1995.
- [136] H. Cordes.Zero Order A Priori Estimates for Solutions of Elliptic Differential Equations.In *Symposium in Pure Mathematics*, pages 157–166, 1961.
- [137] M. Crandall, L. Evans, and P.-L. Lions.Some Properties of Viscosity Solutions of Hamilton Jacobi Equations.*Transactions of American Mathematical Society*, 282:485–502, 1984.
- [138] M. Crandall, H. Ishii, and P.-L. Lions.User’s Guide to Viscosity Solutions of Second Order Partial Differential Equations.*American Mathematical Society Bulletin*, 27:1–67, 1992.
- [139] M. Crandall and P.-L. Lions.Viscosity Solutions of Hamilton-Jacobi Equations.*Transactions of American Mathematical Society*, 277:1–42, 1983.
- [140] M. Crandall and P.-L. Lions.Two approximations of solutions of Hamilton-Jacobi equations.*Mathematics of Computation*, 43:1–19, 1984.
- [141] D. Cremers, T. Kohlberger, and C. Schnorr.Nonlinear Shape Statistics in Mumford-Shah Based Segmentation.In *European Conference on Computer Vision*, volume 2, pages 93–108, 2002.

- [142] D. Cremers, C. Schnorr, and J. Weickert. Diffusion-snakes: Combining statistical shape knowledge and image information in a variational framework. In *IEEE Workshop in Variational and Level Set Methods*, pages 137–144, 2001.
- [143] G. Cross and A. Zisserman. Surface reconstruction from multiple views using apparent contours and surface texture. In A. Leonardis, F. Solina, and R. Bajcsy, editors, *Confluence of Computer Vision and Computer Graphics*, pages 25–47. Kluwer, 2000.
- [144] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *ACM SIGGRAPH*, pages 303–312, 1996.
- [145] G. Dal Maso, J.-M. Morel, and S. Solimini. A variational method in image segmentation: existence and approximation results. *Acta Mathematica*, pages 89–151, 1992.
- [146] A. Dale, B. Fischl, and M. Sereno. Cortical Surface-based Analysis I: Segmentation and Surface Reconstruction. *NeuroImage*, 9:179–194, 1999.
- [147] P. Danielsson. Euclidean Distance Mapping. *Computer Graphics and Image Processing*, 14:227–248, 1980.
- [148] B. Dawant, S. Hartmann, J.-P. Thirion, F. Maes, D. Vandermeulen, and P. Demaerel. Automatic 3-D Segmentation of Internal Structures of the Head in MR Images Using a Combination of Similarity and Free-Form Transformations: Part I, Methodology and Validation on Normal Subjects. *IEEE Transactions on Medical Imaging*, 18:909–916, 1999.
- [149] E. De Giorgi, M. Carriero, and A. Leaci. Existence theorem for a minimum problem with free discontinuity set. *Archives for Rational Mechanics and Analysis*, 108:195–218, 1989.
- [150] E. DeCastro and C. Morandi. Registration of Translated and Rotated Images Using Finite Fourier Transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:700–703, 1987.
- [151] R. Deriche. Using Canny’s Criteria to Derive a Recursively Implemented Optimal Edge Detector. *International Journal of Computer Vision*, 1:167–187, 1987.
- [152] R. Deriche, S. Bouvin, and O. Faugeras. Front Propagation and Level-Set Approach for Geodesic Active Stereovision. In *Asian Conference in Computer Vision*, pages 1:640–647, 1998.
- [153] M. Desbrun and M.-P. Cani-Gascuel. Active Implicit Surfaces for Computer Animation. In *Graphics Interface*, pages 143–150, 1998.
- [154] M. Desbrun and M.-P. Gascuel. Animating Soft Substances with Implicit Surfaces. In *ACM SIGGRAPH*, pages 287–290, 1995.
- [155] M. Desbrun, M. Meyer, P. Schröder, and A. Barr. Implicit Fairing of Irregular Meshes using Diffusion and Curvature Flow. In *ACM SIGGRAPH*, pages 317–324, 1999.
- [156] M. Desbrun, M. Meyer, P. Schröder, and A. Barr. Discrete differential-geometry operators in nD . Technical report, Multi-res modeling group, Caltech, 2000.
- [157] A. Desolneux, L. Moisan, and J.-M. Morel. Meaningful Alignments. *International Journal of Computer Vision*, 40:7–23, 2000.
- [158] A. Desolneux, L. Moisan, and J.-M. Morel. Edge Detection by Helmholtz Principle. *Journal of Mathematical Imaging and Vision*, 14:271–284, 2001.

- [159] R. DeVore, B. Jawerth, and B. Lucier. Image Compression Through Wavelet Transform Coding. *IEEE Transactions on Information Theory*, 38:719–746, 1992.
- [160] F. Dibos. Projective Analysis of 2-D Images. *IEEE Transactions on Image Processing*, 7:274–279, 1998.
- [161] F. Dibos. Du groupe projectif au groupe des recalages, une nouvelle modélisation. *C.R. Académie de Science de Paris*, 332:799–804, 2001.
- [162] F. Dibos and G. Koepfler. Total Variation Minimization by the Fast Level Sets Transform. In *IEEE Workshop in Variational and Level Set Methods*, pages 179–185, 2001.
- [163] U. Diewald, T. Preufer, and C. Vogel. Anisotropic Diffusion in Vector Field Visualization on Euclidean Domains and Surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 6:139–149, 2000.
- [164] M.P. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, 1976.
- [165] D. Dobson and C. Vogel. Convergence of an iterative method for total variation image denoising. *SIAM Journal on Numerical Analysis*, 34:1779–1971, 1997.
- [166] D. Dobson and C. Vogel. Global Total Variation Minimization. *SIAM Journal on Numerical Analysis*, 37:646–664, 2000.
- [167] J. Dorsey and P. Hanrahan. Digital materials and virtual weathering. *Scientific American*, 282:46–53, 2000.
- [168] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.
- [169] F. Durand. *3D Visibility: Analysis, Study and Application*. PhD thesis, MIT, 1999.
- [170] S. Durand and J. Froment. Reconstruction of wavelet coefficients using total variation minimization. Technical Report 18, CMLA, 2001.
- [171] N. Duta and A. Jain. Corpus Callosum shape analysis: a comparative study of group differences associated with dyslexia, gender and handedness. 2000.
- [172] M. Eck, T. DeRose, T. Duchamp, M. Lounsbery, and W. Stuetzle. Multi-resolution analysis of arbitrary meshes. In *ACM SIGGRAPH*, pages 173–182, 1995.
- [173] M. Eck and H. Hoppe. Automatic reconstruction of B-spline surfaces of arbitrary topological type. *Computer Graphics*, 30:325–334, 1996.
- [174] H. Edelsbrunner and E. Mücke. Three-dimensional Alpha Shapes. *ACM Transactions on Graphics*, 13:43–72, 1994.
- [175] J. Eells and L. Lemarie. A report on harmonic maps. *Bulletin of London Mathematics Society*, 10:1–68, 1978.
- [176] J. Eells and L. Lemarie. Another report on harmonic maps. *Bulletin of London Mathematics Society*, 20:385–524, 1988.
- [177] A. El-Fallah and G. Ford. Mean curvature evolution and surface area scaling in image filtering. *IEEE Transactions on Image Processing*, 6:750–753, 1997.
- [178] A. Elgammal, D. Harwood, and L. Davis. Non-Parametric Model for Background Subtraction. In *European Conference on Computer Vision*, pages II:751–767, 2000.
- [179] D. Enright, R. Fedkiw, J. Ferziger, and I. Mitchell. A Hybrid Particle Level Set Method for Improved Interface Capturing. *preprint*, 2002. Submitted: Journal of Computational Physics.

- [180] D. Enright, S. Marschner, and R. Fedkiw. Animation and Rendering of Complex Water Surfaces. In *ACM SIGGRAPH*, pages 736 – 744, 2002.
- [181] L. Evans. *Partial Differential Equations*. American Mathematical Society, 1998.
- [182] L. Evans and R. Gariepy. *Measure theory and fine properties of functions*. CRC Press, 1992.
- [183] L. Evans, H. Peng, and P. Souganidis. Phase Transitions and Generalized Motion by Mean Curvature. *Communication on Pure and Applied Mathematics*, 5:1097, 1992.
- [184] L. Evans and J. Spruck. Motion of Level Sets by Mean Curvature. *Journal of Differential Geometry*, 33:635–681, 1991.
- [185] P. Faloutsos, M. van de Panne, and D. Terzopoulos. Dynamic Free-Form Deformations for Animation Synthesis. *IEEE Transactions on Visualization and Computer Graphics*, 3:201–214, 1997.
- [186] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1993.
- [187] O. Faugeras, F. Clément, R. Deriche, R. Keriven, T. Papadopoulo, J. Gomes, G. Hermosillo, P. Kornprobst, D. Lingrad, J. Roberts, T. Viéville, and F. Devernay. The inverse EEG and MEG problems: The adjoint state approach I: The continuous case. Technical Report 3673, INRIA, France, 1999.
- [188] O. Faugeras and R. Keriven. Variational Principles, Surface Evolution, PDE's, Level Set Methods and the Stereo Problem. Technical Report 3021, INRIA, France, 1996.
- [189] O. Faugeras and R. Keriven. Complete Dense Stereovision Using Level Set Methods. In *European Conference on Computer Vision*, pages 379–393, 1998.
- [190] O. Faugeras and R. Keriven. Variational Principles, Surface Evolution, PDEs, Level Set Methods, and the Stereo Problem. *IEEE Transactions on Image Processing*, 7:336–344, 1998.
- [191] O. Faugeras, Q.-T. Luong, and T. Papadopoulo. *The Geometry of Multiple Images*. MIT Press, 2000.
- [192] R. Fedkiw, T. Aslam, B. Merriman, and S. Osher. Non-Oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (The Ghost Fluid Method). *Journal of Computational Physics*, 152:457–492, 1999.
- [193] R. Fedkiw, J. Stam, and H. Jensen. Visual Simulation of Smoke. In *ACM SIGGRAPH*, pages 23–30, 2001.
- [194] B. Fischer and J. Modersitzki. Fast diffusion registration. Technical Report A-01-18, Institute of Mathematics, Medical University of Lübeck, 2001. To appear in *Contemporary Mathematics*.
- [195] B. Fischl, M. Sereno, and A. Dale. Cortical Surface-based Analysis II: Inflation, Flattening, and a Surface-based Coordinate System. *NeuroImage*, 9:195–207, 1999.
- [196] M. Fischler and R. Elschlager. The representation and matching of pictorial images. *IEEE Transactions on Computers*, 22:67–92, 1973.
- [197] A. Fitzgibbon. Robust Registration of 2D and 3D Point Sets. In *British Machine Vision Conference*, volume 2, pages 411–420, 2001.
- [198] N. Foster and R. Fedkiw. Practical Animation of Liquids. *ACM SIGGRAPH*, pages 23 – 30, 2001.
- [199] N. Foster and D. Metaxas. Modeling og the Motion of Hot, Turbulent Gass. *Graphical Models and Image Processing*, 58:471–483, 1996.

- [200] A. Frangi, D. Rueckert, J. Schnabel, and W. Niessen. Automatic 3D ASM construction via atlas-based landmarking and volumetric elastic registration. In *Information Processing in Medical Imaging*, pages 78–91, 2001.
- [201] S. Frisken, R. Perry, A. Rockwood, and T. Jones. Adaptively Sampled Distance Fields: A General Representation of Shape for Computer Graphics. In *ACM SIGGRAPH*, pages 249–254, 2000.
- [202] J. Froment and L. Moisan, editors. MEGAWave2. Free image processing software obtainable at <http://www.cmla.ens-cachan.fr/Cmla/Megawave>.
- [203] P. Fua and Y. Leclerc. Model driven edge detection. *Machine Vision and Applications*, 3:45–56, 1990.
- [204] M. Gage and R. Hamilton. The heat equation shrinking convex plane curves. *Journal of Differential Geometry*, 23:69–96, 1986.
- [205] T. Galyean and J. Hughes. Sculpting: An Interactive Volumetric Modeling Technique. In *ACM SIGGRAPH*, pages 267–274, 1991.
- [206] M. Gelgon and P. Bouthemy. A region-level graph labeling approach to motion-based segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 514–519, 1997.
- [207] S. Geman and D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.
- [208] G. Gerig, J. Martin, R. Kikinis, O. Kbler, M. Shenton, and F. Jolesz. Unsupervised tissue type segmentation of 3D dual-echo MR head data. *Image and Vision Computing*, 10:349–360, 1992.
- [209] G. Gerig, M. Styner, M. Shenton, and J. Lieberman. Shape versus Size: Improved Understanding of the Morphology of Brain Structures. In *Medical Imaging Computing and Computer-Assisted Intervention*, 2001.
- [210] M. Giacomini, G. Modica, E. Rivlin, and J. Soucek. Variational problems for maps of bounded variation with values in s^1 . *Calculus of Variations*, 1:87–121, 1993.
- [211] P. Giblin. *Graphs, Surfaces and Homology*. Chapman and Hall, 1977.
- [212] P. Giblin and R. Weiss. Reconstruction of Surfaces from Profiles. Technical Report CS-1987-026, University of Maryland, 1987.
- [213] R. Gilmore, M. Childers, C. Leonard, R. Quisling, S. Roper, S. Eisenschenk, and M. Mahoney. Hippocampal volumetrics differentiates patients with temporal lobe epilepsy and extratemporal lobe epilepsy. *Archives of Neurology*, 52:819–824, 1995.
- [214] E. Giusti. *Minimal Surfaces and Functions of Bounded Variation*. Birkhauser, Basel, 1985.
- [215] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky. Cortex Segmentation - A Fast Variational Geometric Approach. In *IEEE Workshop in Variational and Level Set Methods*, pages 127–135, 2001.
- [216] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky. Fast Geodesic Active Contours. *IEEE Transactions on Image Processing*, 10:1467–1475, 2001.
- [217] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 1989.
- [218] J. Gomes and O. Faugeras. Reconciling distance functions and level sets. *Journal of Visual Communication and Image Representation*, 11:209–223, 2000.

- [219] Y. Gousseau and J.-M. Morel. Are natural images of bounded variation. *SIAM Journal on Mathematical Analysis*, 14:634–648, 2001.
- [220] M. Grayson. The heat equation shrinks embedded plane curves to round points. *Journal of Differential Geometry*, 26:285–314, 1987.
- [221] U. Grenander. *General Pattern Theory*. Oxford University Press, 1993.
- [222] U. Grenander and M. Miller. Representation of knowledge in complex systems. *Journal of Royal Statistics Society, Ser. B*, 56:549–603, 1994.
- [223] W.E.L. Grimson. *From Images to Surfaces*. MIT Press, 1981.
- [224] S. Haker, S. Angenent, A. Tannenbaum, R. Kikinis, G. Sapiro, and M. Halle. Conformal Surface Parameterization for Texture Mapping. Technical Report 1611, University of Minnesota, 1999.
- [225] X. Han, C. Xu, U. Braga-Neto, and J. Prince. Topology Correction in Brain Cortex Segmentation Using a Multiscale, Graph-based Algorithm. *IEEE Transactions on Medical Imaging*, 21:109–121, 2002.
- [226] X. Han, C. Xu, M. Rettmann, and J. Prince. Automatic segmentation editing for cortical surface reconstruction. In *SPIE Medical Imaging*, volume 4322, pages 194–203, 2001.
- [227] X. Han, C. Xu, D. Tosun, and J. Prince. Cortical Surface Reconstruction Using a Topology Preserving Geometric Deformable Model. In *IEEE Mathematical Methods in Biomedical Image Analysis*, pages 213–220, 2001.
- [228] R. Haralick. Digital step edges from zero crossing of second directional derivatives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:58–68, 1984.
- [229] R. Hardt. Singularities of harmonic maps. *Bulletin of the American Mathematical Society*, 34:15–34, 1997.
- [230] F. Harlow and J. Welch. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with a Free Surface. *The Physics of Fluids*, 8:2182–2189, 1965.
- [231] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy. Uniformly High Order Accurate Essentially Non-Oscillatory Schemes III. *Journal of Computational Physics*, 71:231–303, 1987.
- [232] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.
- [233] H.J.A.M. Heijmans. *Morphological Image Operators*. Academic Press, 1994.
- [234] H.J.A.M. Heijmans and R. Keshet. First Steps Towards A Self-Dual Morphology. In *IEEE International Conference on Image Processing*, 2000.
- [235] H.J.A.M. Heijmans and R. Keshet. Inf-Semilattice Approach to Self-Dual Morphology. Technical Report PNA-R0101, CWI, Amsterdam, 2001.
- [236] K. Held, E. Kops, B. Kraus, W. Wells, R. Kikinis, and H. W. Müller-Göttsche. Markov Random Field Segmentation of Brain MR Images. *IEEE Transactions on Medical Imaging*, 16:878–887, 1998.
- [237] H. L. F. von Helmholtz. *Treatise on Physiological Optics*. Dover, 1925.
- [238] C. Herbe. Numerical Methods for Nonlinear Diffusion Models of Sandpile Growth. Master’s thesis, Department of Mathematics, University of Kaiserslautern, Germany, 1999.

- [239] A. Hilton, A. Stoddart, J. Illingworth, and T. Windeatt. Implicit surface - based geometric fusion. *Computer Vision and Image Understanding*, 69:273–291, 1998.
- [240] M. Hiltzik and A. Pham. Synthetic Actors Guild. *Los Angeles Times, National Edition: AI+ 2001*, May 8, 2001.
- [241] C. Hoffmann. *Geometric and Solid Modeling*. Morgan Kaufmann, 1989.
- [242] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *ACM SIGGRAPH*, 1992.
- [243] B. Horn. *Robot Vision*. MIT Press, 1986.
- [244] B. Horn and M. Brooks. *Shape from Shading*. MIT Press, 1989.
- [245] B. Horn and B. Schunck. Determinating Optical Flow. *Artificial Intelligence*, 17:185–203, 1981.
- [246] P. Huber. *Robust Statistics*. John Wiley & Sons, 1981.
- [247] G. Huiskamp. Difference formulas for the surface laplacian on a triangulated surface. *Journal of Computational Physics*, 95:477–496, 1991.
- [248] B. Hunt. The application of constrained least-squares estimation to image restoration by digital computer. *IEEE Transactions on Computers*, C-22:805–812, 1973.
- [249] D. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:850–863, 1993.
- [250] T. Ilmanen. Generalized flows of sets by mean curvature on a manifold. *Indiana Journal of Mathematics*, pages 671–705, 1992.
- [251] M. Isard and A. Blake. Contour Tracking by Stochastic Propagation of Conditional Density. In *European Conference on Computer Vision*, volume I, pages 343–356, 1996.
- [252] H. Ishii. A generalization of Bence, Merriman, and Osher algorithm for motion by mean curvature. In A. Damlamian, J. Spruck, and A. Visintin, editors, *Curvature Flows and Related Topics*, pages 111–127. Gakkōtoshō, 1995.
- [253] K. Ito and K. Kunisch. An active set strategy formulation Based on the augmented Lagrangian Formulation for image restoration. *Mathematical Modelling and Numerical Analysis*, 33:1–21, 1999.
- [254] S. Jehan-Besson, M. Barlaud, and G. Aubert. Video Object Segmentation Using Eulerian Region-Based Active Contours. In *IEEE International Conference in Computer Vision*, volume I, pages 353–360, 2001.
- [255] S. Jehan-Besson, M. Barlaud, and G. Aubert. DREAM2S: Deformable Regions driven by an Eulerian Accurate Minimization Method for image and video segmentation, application to face detection in color video sequences. In *European Conference on Computer Vision*, volume 3, pages 365–380, 2002.
- [256] G.-S. Jiang and D. Peng. Weighted ENO schemes for Hamilton Jacobi equations. *SIAM Journal of Scientific Computing*, 21:2126–2143, 2000.
- [257] G.-S. Jiang and C.-W. Shu. Efficient Implementation of Weighted ENO Schemes. *Journal of Computational Physics*, 126:202–228, 1996.
- [258] M.-P. Jolly, S. Lakshmanan, and A. Jain. Vehicle Segmentation and Classification Using Deformable Templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:293–308, 1996.

- [259] S. Joshi, A. Banerjee, G. Christensen, J. Csernansky, J. Haller, M. Miller, and L. Wang. Gaussian Random Fields on Sub-Manifolds for Characterizing Brain Surfaces. In *Information Processing in Medical Imaging*, 1997.
- [260] B. Julesz. *Foundations of Cyclopean Perception*. The University of Chicago Press, 1971.
- [261] T. Kapur. *Model Based Three Dimensional Medical Image Segmentation*. PhD thesis, MIT, 1999.
- [262] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. In *IEEE International Conference in Computer Vision*, pages 261–268, 1987.
- [263] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *International Journal of Computer Vision*, 1:321–332, 1988.
- [264] D. Kendall. Shape manifolds, procrustean metrics and complex projective spaces. *Bulletin of London Mathematics Society*, 16, 1984.
- [265] J. Kender and T. Kanade. Mapping Image Properties into Shape Constraints: Skewed Symmetry and Affine-Transformable Patterns, and the Shape-from-Texture Paradigm. In *National Conference on Artificial Intelligence*, pages 4–6, 1980.
- [266] R. Keriven. *Equations aux Dérivées Partielles, Evolutions de Courbes et de Surfaces et Espaces d'Échelle: Applications à la Vision par Ordinateur*. PhD thesis, Ecole Nationale des Ponts et Chaussées, 1997.
- [267] R. Keriven. A variational framework to shape from contours. Technical Report 221, Ecole Nationale des Ponts et Chausses, 2002.
- [268] C. Kervrann and A. Trubuil. Optimal Level Curves and Global Minimisers of Cost Functionals in Image Segmentation. *Journal of Mathematical Imaging and Vision*, 17:153–173, 2002.
- [269] R. Keshet. Mathematical Morphology On Complete Semilattices and Its Applications to Image Processing. *Fundamenta Informatica*, 41:33–56, 2000.
- [270] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi. Gradient flows and geometric active contour models. In *IEEE International Conference in Computer Vision*, pages 810–815, 1995.
- [271] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi. Conformal curvature flows: from phase transitions to active vision. *Archive for Rational Mechanics and Analysis*, 134:275–301, 1996.
- [272] J. Kim, J. Fisher, A. Yezzi, M. Cetin, and A. Willsky. Non-Parametric Methods for Image Segmentation using Information Theory and Curve Evolution. In *IEEE International Conference on Image Processing*, 2002.
- [273] B. Kimia and K. Siddiqi. Geometric heat equation and nonlinear diffusion of shapes and images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 113–120, 1994.
- [274] B. Kimia and K. Siddiqi. Geometric heat equation and non-linear diffusion of shapes and images. *Computer Vision and Image Understanding*, 64:305–322, 1996.
- [275] B. Kimia, A. Tannenbaum, and S. Zucker. On the evolution of curves via a function of curvature. I. The classical case. *Journal of Mathematical Analysis and Applications*, 163:438–458, 1992.

- [276] B. Kimia, A. Tannenbaum, and S. Zucker. Shocks and Deformations i: The Components of two-dimensional Shape and the reaction-diffusion space. *International Journal of Computer Vision*, 15:189–224, 1995.
- [277] R. Kimmel. Affine differential signatures for gray level images of planar shapes. In *IARP International Conference on Pattern Recognition*, 1996.
- [278] R. Kimmel. Intrinsic scale space for images on surfaces: The geodesic curvature flow. *Graphical Models and Image Processing*, pages 365–372, 1997.
- [279] R. Kimmel and A. Bruckstein. Tracking level sets by level sets: a method for solving the shape from shading problem. *Computer Vision, Graphics and Image Understanding*, 62:47–58, 1995.
- [280] R. Kimmel and A. Bruckstein. Regularized Laplacian zero crossings as optimal edge integrators. In *Image and Vision Computing Conference*, New Zealand, 2001.
- [281] R. Kimmel and A. Bruckstein. On edge detection edge integration and geometric active contours. In *International Symposium on Mathematical Morphology*, volume 2, Sydney, Australia, 2002.
- [282] R. Kimmel and A. Kiryati, N. Bruckstein. Multivalued distance maps for motion planning on surfaces with moving obstacles. *IEEE Transactions on Robotics & Automation*, 14:427–435, 1998.
- [283] L. Kobbelt, M. Botsch, U. Schwanecke, and H.-P. Seidel. Feature Sensitive Surface Extraction from Volume Data. In *ACM SIGGRAPH*, pages 57–66, 2001.
- [284] L. Kobbelt, S. Campagna, J. Vorsatz, and H.-P. Seidel. Interactive multi-resolution modeling on arbitrary meshes. In *ACM SIGGRAPH*, pages 105–114, 1998.
- [285] J. Koenderink. The structure of images. *Biological Cybernetics*, 50:363–370, 1984.
- [286] J. Koenderink. *Solid Shape*. MIT Press, 1990.
- [287] G. Koepfler, C. Lopez, and J.-M. Morel. A Multiscale Algorithm for Image Segmentation by Variational Method. *SIAM Journal on Numerical Analysis*, 31:282–299, 1994.
- [288] G. Koepfler and L. Moisan. Geometric Multiscale Representation of Numerical Images. In *International Conference on Scale-Space Theories in Computer Vision*, pages 339–350, 1999.
- [289] D. Koller, K. Daniilidis, and H-H. Nagel. Model-Based Object Tracking in Monocular Image Sequences of Road Traffic Scenes. *International Journal of Computer Vision*, 10:257–281, 1993.
- [290] V. Kolmogorov and R. Zabih. Multi-camera Scene Reconstruction via Graph Cuts. In *European Conference on Computer Vision*, volume 3, pages 82–96, 2002.
- [291] T. Kong and A. Rosenfeld. Digital Topology: Introduction and Survey. *Computer Vision, Graphics and Images Processing*, 48:357–393, 1989.
- [292] T.Y. Kong and A. Rosenfeld. If we Use 4 or 8-Connectedness for Both the Objects and the Background, the Euler Characteristic Is Not Locally Computable. *Pattern Recognition Letters*, 11:231–232, 1990.
- [293] A. Koshelev. An Inequality for Elliptic Operators of Second-Order with Restricted Coefficients. *Soviet Mathematics*, 12:1009–1012, 1972.
- [294] D. Kozinska, O. Tretiak, J. Nissanov, and C. Ozturk. Multidimensional Alignment Using the Euclidean Distance Transform. *Graphical Models and Image Processing*, 6:373–385, 1997.

- [295] V. Krishnamurthy and M. Levoy.Fitting smooth surfaces to dense polygon meshes.*Computer Graphics*, pages 313–324, 1996.
- [296] G. Kühne, J. Weickert, M. Beier, and W. Effelsberg.Fast implicit active contour models.In L. Van Gool, editor, *Pattern Recognition*. Springer, 2002.
- [297] G. Kühne, J. Weickert, O. Schuster, and S. Richter.A tensor-driven active contour model for moving object segmentation.In *IEEE International Conference on Image Processing*, volume 2, pages 73–76, 2001.
- [298] K. Kutulakos and S. Seitz.A Theory of Shape by Space Carving.*International Journal of Computer Vision*, 38:199–218, 2000.
- [299] M. Lades, C. Borbruggen, J. Buhmann, C. von der Malsburg, R. Wurtz, and W. Konnen.Distortion invariant object recognition in the dynamic link architecture.*IEEE Transactions on Computers*, 42:300–311, 1993.
- [300] O. Ladyzhenskaya, V. Solonnikov, and N. Uraltseva.*Linear and quasilinear equations of parabolic type*.American Mathematical Society, 1968.
- [301] D. Laidlaw, W. Trumbore, and J. Hughes.Constructive Solid Geometry for Polyhedral Objects.In *ACM SIGGRAPH*, pages 161–170, 1986.
- [302] L. Landau and E. Lifshitz.*Fluid Mechanics*.Butterworth Heinemann, 1959.
- [303] A. Laurentini.The Visual Hull Concept for Silhouette-Based Image Understanding.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:150–162, 1994.
- [304] S. Lavallée and R. Szeliski.Recovery of the Position and Orientation of free-form Objects from Image Contours using 3D Distance Maps.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:378–390, 1995.
- [305] H. Le and D. Kendall.The Riemannian structure of Euclidean shape spaces: a novel environment for statistics.*The Annals of Statistics*, 4216:1225–1271, 1993.
- [306] Y. Leclerc.Constructing Simple Stable Description for Image Partitioning.*International Journal of Computer Vision*, 3:73–102, 1989.
- [307] C. Leonard, C. Williams, R. Nicholls, O. Agee, K. Voeller, J. Honeyman, and E. Staab.Angelman and prader-willi syndrome. A magnetic resonance imaging study of differences in cerebral structure.*American Journal of Medical Genetics*, 46:26–33, 1993.
- [308] G. P. Leonardi and I. Tamanini.On Minimizing Partitions with Infinitely Many Components.*Annali dell'Universit di Ferrara - Sez. VII - Sc. Mat.*, XLIV:41–57, 1998.
- [309] M. Leventon, E. Grimson, and O. Faugeras.Level Set Based Segmentation with Intensity and Curvature Priors.In *IEEE Mathematical Methods in Biomedical Image Analysis*, pages 4–11, 2000.
- [310] M. Leventon, E. Grimson, and O. Faugeras.Statistical Shape Influence in Geodesic Active Contours.In *IEEE Conference on Computer Vision and Pattern Recognition*, pages I:316–322, 2000.
- [311] R.J. LeVeque.*Numerical Method for Conservation Laws*.Birkhauser, 1992.
- [312] J. Liang, T. McInerney, and D. Terzopoulos.United Snakes.In *IEEE International Conference in Computer Vision*, pages 933–940, 1999.

- [313] S. Liapis, E. Sifakis, and G. Tziritas. Color and texture segmentation using deterministic relaxation and fast marching algorithms. In *IARP International Conference on Pattern Recognition*, pages 621–624, 2000.
- [314] S. Liapis, E. Sifakis, and G. Tziritas. Colour and texture segmentation using wavelet frame analysis, deterministic relaxation and fast marching algorithms. *Journal of Visual Communication and Image Representation*, to appear.
- [315] P.-L. Lions. Une Inégalité pour les Opérateurs Elliptiques du Second Ordre. *Annali di Matematica Pura ed Applicata*, CXXVII:1–11, 1981.
- [316] P. Lipson, A. Yuille, D. Okeefe, J. Cavanaugh, J. Taaffe, and D. Rosenthal. Deformable Templates for feature extraction from Medical Images. In *European Conference on Computer Vision*, pages 413–417, 1990.
- [317] J.-L. Lisani, P. Monasse, and L. Rudin. Fast Shape Extraction and Applications. Submitted: IEEE Transaction on Pattern Analysis and Machine Intelligence.
- [318] X.-D. Liu, S. Osher, and T. Chan. Weighted Essentially Non-Oscillatory Schemes. *Journal of Computational Physics*, 126:202–212, 1996.
- [319] L. Ljung. *System Identification: theory for the user*. Prentice Hall, 1987.
- [320] W. Lorensen and H. Cline. Marching cubes: a high resolution 3D surface construction algorithm. In *ACM SIGGRAPH*, volume 21, pages 163–170, 1987.
- [321] L. Lorigo, O. Faugeras, E. Grimson, R. Keriven, R. Kikinis, A. Nabavi, and C. Westin. Co-Dimension 2 Geodesic Active Contours for MRI Segmentation. In *Information Processing in Medical Imaging*, pages 126–139, 1999.
- [322] L. Lorigo, O. Faugeras, E. Grimson, R. Keriven, R. Kikinis, A. Nabavi, and C. Westin. Codimension-Two Geodesic Active Contours for the Segmentation of Tubular Structures. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages I:444–451, 2000.
- [323] D. Lowe. Robust Model-based Motion Tracking through the Integration of Search and Estimation. *International Journal of Computer Vision*, 8:113–122, 1992.
- [324] T. Lu, P. Neittaanmäki, and X.-C. Tai. A parallel splitting up method and its application to Navier-Stokes equations. *Applied Mathematics Letters*, 4:25–29, 1991.
- [325] T. Lu, P. Neittaanmäki, and X.-C. Tai. A parallel splitting up method for partial differential equations and its application to Navier-Stokes equations. *RAIRO Mathematical Modeling and Numerical Analysis*, 26:673–708, 1992.
- [326] B. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.
- [327] D. MacDonald, N. Kabani, D. Avis, and A. Evans. Automated 3-D Extraction of Inner and Outer Surfaces of Cerebral Cortex from MRI. *NeuroImage*, 12:340–356, 2000.
- [328] J. Maintz and M. Viergever. A Survey for Medical Image Registration. *Medical Image Analysis*, 2:1–36, 1998.
- [329] R. Malladi, J. Sethian, and B. Vemuri. A topology independent shape modeling scheme. In *Geometric Methods in Computer Vision II*, pages 246–256, 1993.

- [330] R. Malladi, J. Sethian, and B. Vemuri. Evolutionary fronts for topology independent shape modeling and recovery. In *European Conference on Computer Vision*, pages 1–13, 1994.
- [331] R. Malladi, J. Sethian, and B. Vemuri. Shape Modeling with Front Propagation: A Level Set Approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:158–175, 1995.
- [332] A. Mansouri. Region Tracking via Level Set PDEs without Motion Computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:947–961, 2002.
- [333] P. Maragos. Pattern Spectrum and Multiscale Shape Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1:701–716, 1989.
- [334] P. Maragos. Differential Morphology and Image Processing. *IEEE Transactions on Image Processing*, 5:922–937, 1996.
- [335] P. Maragos. Algebraic and PDE Approaches for Lattice Scale-Spaces with Global Constraints. *International Journal of Computer Vision*, 2003.
- [336] P. Maragos and M. Butt. Curve Evolution, Differential Morphology and Distance Transforms as Applied to Multiscale and Eikonal Problems. *Fundamenta Informatica*, 41:91–129, 2000.
- [337] R. March. Visual Reconstruction with Discontinuities Using Variational Methods. *Imaging and Vision Computing*, 10:30–38, 1992.
- [338] G. Marchuk. Splitting and alternating direction methods. In P. Ciarlet and J.-L. Lions, editors, *Handbook of Numerical Analysis*, volume I, pages 197–462. North Holland, 1990.
- [339] K. Mardia and I. Dryden. Shape distributions for landmark data. *Advances in Applied Probability*, 4:742–755, 1989.
- [340] G. Markstein. *Nonsteady Flame Propagation*. Pergamon Press, 1964.
- [341] A. Marquina and S. Osher. Explicit Algorithms for a New Time Dependant Model Based on Level Set Motion for Nonlinear Deblurring and Noise Removal. *SIAM Journal on Scientific Computing*, 22:387–405, 2001.
- [342] D. Marr. *Vision*. Freeman, 1982.
- [343] D. Marr and E. Hildreth. Theory of Edge Detection. *Proceedings of the Royal Society London*, 207:187–217, 1980.
- [344] A. Martelli. Edge Detection Using Heuristic Search Methods. *Computer Vision, Graphics and Image Processing*, 1:169–182, 1972.
- [345] S. Masnou and J.-M. Morel. Level lines based disocclusion. In *IEEE International Conference on Image Processing*, pages 3:259–263, 1998.
- [346] U. Massari and I. Tamanini. On the Finiteness of Optimal Partitions. *Annali dell'Università di Ferrara - Sez. VII - Sc. Mat.*, XXXIX:167–185, 1993.
- [347] G. Matheron. *Random Sets and Integral Geometry*. John Wiley & Sons, 1975.
- [348] G. Matheron. Les Nivellements. Technical report, Centre de Morphologie Mathématique, France, 1997.
- [349] S. Mauch. Closest Point Transform. Technical report, Caltech, 2000.
- [350] T. McInerney and D. Terzopoulos. A finite element model for 3D shape reconstruction and nonrigid motion tracking. In *IEEE International Conference in Computer Vision*, pages 518–523, 1993.

- [351] T. McInerney and D. Terzopoulos.A Dynamic Finite Element Surface Model for Segmentation and Tracking in Multidimensional Medical Images with Application to Cardiac 4D Image Analysis.*Computerized Medical Imaging and Graphics*, 19:69–83, 1995.
- [352] T. McInerney and D. Terzopoulos.Deformable Models in Medical Image Analysis: A Survey.*Medical Image Analysis*, 1:91–108, 1996.
- [353] T. McInerney and D. Terzopoulos.Topology Adaptive Deformable Surfaces for Medical Image Volume Segmentation.*IEEE Transactions on Medical Imaging*, 18:840–850, 1999.
- [354] T. McInerney and D. Terzopoulos.T-snakes: Topology Adaptive Snakes.*Medical Image Analysis*, 4:73–91, 2000.
- [355] T. McIreney and D. Terzopoulos.Topologically Adaptable Snakes.In *IEEE International Conference in Computer Vision*, pages 840–845, 1995.
- [356] G. McLachlan, D. Peel, and W. Whiten.Maximum likelihood clustering via normal mixture model.*Signal Processing: Image Communication*, 8:105–111, 1996.
- [357] T. Meis and U. Marcowitz.*Numerische Behandlung partieller Differentialgleichungen*.Springer, 1978.
- [358] F. Mémoli and S. Sapiro, G. abd Osher.Solving variational problems and partial differential equations mapping into general target manifolds.Technical Report 1827, IMA, University of Minnesota, 2002.
- [359] B. Merriman, J. Bence, and S. Osher.Diffusion generated motion by mean curvature.In *Workshop on Computational Crystal Growers*, pages 73–83, 1992.
- [360] D. Metaxas.*Physics-Based Deformable Models*.Kluwer Academic Publishers, 1996.
- [361] D. Metaxas and D. Terzopoulos.Constrained deformable superquadrics and nonrigid motion tracking.In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 337–343, 1991.
- [362] D. Metaxas and D. Terzopoulos.Dynamic Deformation of Solid Primitives with Constraints.In *ACM SIGGRAPH*, pages 309–312, 1992.
- [363] D. Metaxas and D. Terzopoulos.Shape and Nonrigid Motion Estimation Through Physics-Based Synthesis.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:580–591, 1993.
- [364] F. Meyer and P. Bouthemy.Region-Based Tracking Using Affine Motion Models in Long Image Sequences.*CVGIP: Image Understanding*, 60:119–140, 1994.
- [365] F. Meyer and J. Serra.Contrasts and Activity Lattice.*Signal Processing*, 16:303–317, 1989.
- [366] Y. Meyer.The Levelings.In *Mathematical Morphology and Its Applications to Image and Signal Processing*. Kluwer Academic Publishers, 1998.
- [367] Y. Meyer.Fast computation of a contrast-invariant image representation.*IEEE Transactions on Image Processing*, 9:860–872, 2000.
- [368] Y. Meyer.Oscillating Patterns in Image Processing and Nonlinear Evolution Equations.*American Mathematics Society, University Lecture Series*, 22:136, 2002.
- [369] Y. Meyer and P. Maragos.Nonlinear PDE's and Numerical Algorithms for Modeling Levelings and Reconstruction Filters.In *International Conference on Scale-Space Theories in Computer Vision*, pages 363–374, 1999.

- [370] Y. Meyer and P. Maragos. Nonlinear Scale-Space Representation with Morphological Levelings. *Journal of Visual Communication and Image Representation*, 11:245–265, 2000.
- [371] M. Miller and L. Younes. Group action, diffeomorphism and matching: a general framework. In *Statistical and Computational Theories of Vision*, Fort Collins, Colorado, 1999.
- [372] A. Mitchell and D. Griffiths. *The Finite Difference Method in Partial Differential Equations*. Wiley, 1980.
- [373] A. Mittal and D. Huttenlocher. Scene Modeling for Wide Area Surveillance and Image Synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2160–2167, 2000.
- [374] L. Moisan. Affine Plane Curve Evolution: A Fully Consistent Scheme. *IEEE Transactions on Image Processing*, 7:411–420, 1998.
- [375] P. Monasse. Contrast Invariant Image Registration. In *International Conference on Acoustics, Speech and Signal Processing*, volume 6, pages 3221–3224, 1999.
- [376] P. Monasse. *Contrast Invariant Representation of Digital Images and Application to Registration*. PhD thesis, Ceremade, University Paris 9–Dauphine, 2000.
- [377] O. Monga, S. Benayoun, and O. Faugeras. From partial derivatives of 3D density images to ridge line. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 354–359, 1992.
- [378] U. Montanari. On the Optimal Detection of Curves in Noisy Pictures. *ACM Communications*, 14:335–345, 1971.
- [379] J.-M. Morel and S. Solimini. Segmentation of images by variational methods: a constructive approach. *Revista Matemática de la Universidad Complutense de Madrid*, 1:169–182, 1988.
- [380] J.-M. Morel and S. Solimini. Segmentation d’images par méthode variationnelle: une preuve constructive d’existence. *C.R. Académie de Science de Paris Série I*, 308:465–470, 1989.
- [381] J.-M. Morel and S. Solimini. *Variational Methods in Image Segmentation: with Seven Image Processing Experiments*. Birkhäuser, 1994.
- [382] J.-M. Morel and S. Solimini. *Variational Methods in Image Segmentation*. Birkhäuser, 1995.
- [383] H. Moreton and C. Séquin. Functional minimization or fair surface design. In *ACM SIGGRAPH*, pages 167–176, 1992.
- [384] K. Morton and L. Mayers. *Numerical Solution of Partial Differential Equations*. Cambridge University Press, 1994.
- [385] D. Mumford. Mathematical theories of shape: do they model perception? In *Geometric Methods in Computer Vision*, pages 2–10, 1991.
- [386] D. Mumford, N. Nitzberg, and T. Shiota. Filtering, Segmentation and Depth. *Lecture Notes in Computer Science*, 662, 1993.
- [387] D. Mumford and J. Shah. Boundary detection by minimizing functionals. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 22–26, 1985.
- [388] D. Mumford and J. Shah. Optimal Approximation by Piecewise Smooth Functions and Associated Variational Problems. *Communications on Pure and Applied Mathematics*, 42:577–685, 1989.

- [389] S. Muraki.Volumetric Shape Description of Range Data Using "Blobby Model".In *ACM SIGGRAPH*, volume 25, pages 227–235, 1991.
- [390] B. Natarajan.On generating Topologically Consistent Isosurfaces from Uniform Samples.*Visual Computer*, 11:52–62, 1994.
- [391] D. Nguyen, R. Fedkiw, and H. Jensen.Physically Based Modeling and Animation of Fire.In *ACM SIGGRAPH*, pages 721 – 728, 2002.
- [392] D. Nguyen, R. Fedkiw, and M. Kang.A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow.*Journal of Computational Physics*, 172:71–98, 2001.
- [393] P. Ning and J. Bloomenthal.An evaluation of Implicit Surface Tilers.*IEEE Computer Graphics and Applications*, 13:33–41, 1993.
- [394] J-M. Odobez and P. Boutheny.Robust multiresolution estimation of parametric motion models.*Journal of Visual Communication and Image Representation*, 6:348–365, 1995.
- [395] S. Osher.Riemann solvers, the entropy condition, and difference approximations.*SIAM Journal in Numerical Analysis*, 21:217–235, 1984.
- [396] S. Osher, L.-T. Cheng, M. Kang, H. Shim, and Y.-H. Tsai.Geometric Optics in a Phase Space and Eulerian Framework.*Journal of Computational Physics*, 2001.to appear.
- [397] S. Osher and R. Fedkiw.Level set methods: An Overview and Some Recent Results.*Journal of Computational Physics*, pages 463–502, 2001.
- [398] S. Osher and R. Fedkiw.*The Level Set Method and Dynamic Implicit Surfaces*.Springer, 2002.
- [399] S. Osher and M. Kang.Private Communication.
- [400] S. Osher and L. Rudin.Feature-oriented image enhancement using shock filters.*SIAM Journal of Numerical Analysis*, 27:919–940, 1990.
- [401] S. Osher and J. Sethian.Fronts propagating with curvature-dependent speed : Algorithms based on the Hamilton-Jacobi formulation.*Journal of Computational Physics*, 79:12–49, 1988.
- [402] S. Osher and C.-W. Shu.High Order Essentially Non-Oscillatory Schemes for Hamilton-Jacobi Equations.*Journal of Computational Physics*, 28:902–921, 1991.
- [403] C. Papin, P. Boutheny, E. Mémin, and G. Rochard.Tracking and characterization of highly deformable cloud structures.In *European Conference on Computer Vision*, pages II: 428–442, 2000.
- [404] N. Paragios.*Geodesic Active Regions and Level Set Methods: Contributions and Applications in Artificial Vision*.PhD thesis, I.N.R.I.A./University of Nice-Sophia Antipolis, 2000.<http://www.inria.fr/RRRT/TU-0636.html>.
- [405] N. Paragios.A Variational Approach for the Segmentation of the Left Ventricle in Cardiac Image Analysis.*International Journal of Computer Vision*, 50:345–362, 2002.
- [406] N. Paragios.Shape-based Segmentation and Tracking in Cardiac Image Analysis.*IEEE Transactions on Medical Imaging*, 2003.To appear.
- [407] N. Paragios and R. Deriche.A PDE-based Level Set approach for Detection and Tracking of moving objects.In *IEEE International Conference in Computer Vision*, pages 1139–1145, 1998.

- [408] N. Paragios and R. Deriche.Geodesic Active Regions for Motion Estimation and Tracking.In *IEEE International Conference in Computer Vision*, pages 688–674, 1999.
- [409] N. Paragios and R. Deriche.Geodesic Active Regions for Supervised Texture Segmentation.In *IEEE International Conference in Computer Vision*, pages 926–932, 1999.Previous: INRIA Research Report, RR 3440, June 1998, <http://www.inria.fr/RRRT/RR-3440.html>.
- [410] N. Paragios and R. Deriche.Unifying Boundary and Region-based Information for Geodesic Active Tracking.In *IEEE Conference on Computer Vision and Pattern Recognition*, pages II:300–305, 1999.
- [411] N. Paragios and R. Deriche.Coupled Geodesic Active Regions for Image Segmentation: A Level Set Approach.In *European Conference in Computer Vision*, pages II:224–240, 2000.Previous: INRIA Research Report, RR 3783, October 1999, <http://www.inria.fr/RRRT/RR-3783.html>.
- [412] N. Paragios and R. Deriche.Geodesic Active Contours and Level Sets for the Detection and Tracking of Moving Objects.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:266–280, 2000.
- [413] N. Paragios and R. Deriche.Geodesic Active Regions: A New Framework to Deal with Frame Partition Problems in Computer Vision.*Journal of Visual Communication and Image Representation*, 13:249–268, 2002.
- [414] N. Paragios and R. Deriche.Geodesic Active Regions Level Set Methods for Supervised Texture Segmentation.*International Journal of Computer Vision*, 46(3):223–247, 2002.
- [415] N. Paragios, O. Mellina-Gottardo, and V. Ramesh.Gradient Vector Flow Fast Geodesic Active Contours.In *IEEE International Conference in Computer Vision*, pages I:67–73, 2001.
- [416] N. Paragios, M. Rousson, and V. Ramesh.Matching Distance Functions: A Shape-to-Area Variational Approach for Global-to-Local Registration.In *European Conference on Computer Vision*, pages II:775–790, 2002.
- [417] N. Paragios, M. Rousson, and V. Ramesh.Non-Rigid Registration Using Distance Functions.*Computer Vision and Image Understanding*, 2003.to appear.
- [418] N. Paragios and G. Tziritas.Adaptive Detection and Localization of Moving Objects in Image Sequences.*Signal Processing: Image Communication*, 14:277–296, 1999.
- [419] A. Pasko, V. Adzhiev, A. Sourin, and V. Savchenko.Function representation in geometric modeling: concepts, implementation and applications.*The Visual Computer*, 11:429–446, 1995.
- [420] T. Pavlidis.*Algorithms for Graphics and Image Processing*.Computer Science Press, 1982.
- [421] D. Peng, B. Merriman, S. Osher, H.-K. Zhao, and M. Kang.A PDE-Based Fast Local Level Set Method.*Journal of Computational Physics*, 155:410–438, 1999.
- [422] P. Perona.Orientation Diffusion.*IEEE Transactions on Image Processing*, 7:457–467, 1998.
- [423] P. Perona and J. Malik.Scale space and edge detection using anisotropic diffusion.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:629–639, 1990.

- [424] P. Perona, T. Shiota, and J. Malik. Anisotropic Diffusion. In *Geometry-Driven Diffusion in Computer Vision*. Kluwer Academic Publishers, 1994.
- [425] R. Perry and S. Frisken. Kizamu: A System for Sculpting Digital Characters. In *ACM SIGGRAPH*, pages 47–56, 2001.
- [426] D. Pham and J. Prince. Adaptive fuzzy segmentation of magnetic resonance images. *IEEE Transactions on Medical Imaging*, 18:737–752, 1999.
- [427] L. Piegl and W. Tiller. *The NURBS book*. Springer-Verlag, second edition edition, 1997.
- [428] E. Praun, A. Finkelstein, and H. Hoppe. Lapped Textures. In *ACM SIGGRAPH*, pages 465 – 470, 2000.
- [429] H. Qin and D. Terzopoulos. Dynamic NURBS Swung Surfaces for Physics-Based Shape Design. *Computer-Aided Design*, 27:111–127, 1995.
- [430] H. Qin and D. Terzopoulos. Triangular NURBS and their Dynamic Generalizations. *Computer-Aided Geometric Design*, 14:325–347, 1997.
- [431] T. Reiss. *Recognizing Planar Objects Using Invariant Image Features*. Springer Verlag, 1993.
- [432] P. Remagnino, G. Jones, N. Paragios, and C. Regazzoni. *Video-Based Surveillance Systems: Computer Vision and Distributed Processing*. Kluwer Academic Publishers, 2001.
- [433] A. Requicha and H. Voelcker. Boolean Operations in Solid Modeling: Boundary Evaluation and Merging Algorithms. *Proceedings of the IEEE*, 73:30–44, 1985.
- [434] S. Resnick, A. Goldszal, C. Davatzikos, S. Golski, M. Kraut, E. Metter, R. Bryan, and A. Zonderman. One-year age changes in MRI brain volumes in older adults. *Cerebral Cortex*, 10:464–472, 2000.
- [435] J. Rexilius. Anisotrope nichtlineare Diffusion für die Bildverarbeitung. Technical Report B-99-07, Institute of Mathematics and Computer Science, Medical University of Lübeck, 1999.
- [436] T. Richardson. *Scale independent piecewise smooth segmentation of images via variational methods*. PhD thesis, M.I.T., 1989.
- [437] L. Robert and R. Deriche. Dense Depth Map Reconstruction: A Minimization and Regularization Approach which Preserves Discontinuities. In *European Conference on Computer Vision*, pages I:439–451, 1996.
- [438] L. Robert, R. Deriche, and O. Faugeras. Dense Depth Recovery From Stereo Images. In *European Conference on Artificial Intelligence*, pages 821–823, 1992.
- [439] D. Rogers. *An Introduction to NURBS*. Morgan Kaufmann, 2000.
- [440] R. Ronfard. Region-based strategies for active contour models. *International Journal of Computer Vision*, 13:229–251, 1994.
- [441] J. Rosen. The Gradient Projection Method for Nonlinear Programming: Part II, Nonlinear Constraints. *Journal of Society for Industrial and Applied Mathematics*, 9:514–532, 1961.
- [442] A. Rosenfeld and J. Pfaltz. Distance Functions on Digital Pictures. *Pattern Recognition*, 1:33–61, 1968.
- [443] M. Rousson and R. Deriche. A Variational Framework for Active and Adaptive Segmentation of Vector Valued Images. Technical Report 4515, INRIA, France, 2002.

- [444] M. Rousson and N. Paragios. Shape Priors for Level Set Representations. In *European Conference on Computer Vision*, pages II:78–93, Copenhagen, Denmark, 2002.
- [445] E. Rouy and A. Tourin. A Viscosity Solutions Approach to Shape-from-Shading. *SIAM Journal on Numerical Analysis*, 29:867–884, 1992.
- [446] L. Rudin. Images, Numerical Analysis of Singularities and Shock Filters. Technical Report 5250, California Institute of Technology, 1987.
- [447] L. Rudin and S. Osher. Reconstruction and Enhancement of Signals Using Nonlinear Non-Oscillatory Variational Methods. Technical Report 7, Cognitech, 1990.
- [448] L. Rudin and S. Osher. Total Variational Based Image Restoration with Free Local Constraints. In *IEEE International Conference on Image Processing*, pages 31–35, 1994.
- [449] L. Rudin, S. Osher, and E. Fatemi. Nonlinear Total Variation Based Noise Removal. *Physica D*, 60:259–268, 1992.
- [450] L. Rudin, S. Osher, and C. Fu. Total Variation Based Restoration of Noisy, Blurred Images. *SIAM Journal of Numerical Analysis* - accepted for publication, 1993.
- [451] D. Rueckert, L.I. Sonda, C. Hayes, D. Hill, M. Leach, and D. Hawkes. Nonrigid registration using free-form deformations: Application to breast MR images. *IEEE Transactions on Medical Imaging*, 18:712–721, 1999.
- [452] S. Ruuth, B. Merriman, and S. Osher. Convolution generated motion as a link between cellular automata and continuum pattern dynamics. *Journal of Computational Physics*, 151:836–861, 1999.
- [453] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Thomson Learning, 1996.
- [454] P. Salembier. Overview of the MPEG-7 standard and of future challenges for visual information analysis. *EURASIP Journal on Applied Signal Processing*, pages 343–353, 2002.
- [455] P. Salembier and J. Serra. Flat Zones Filtering, Conencted Operators, and Filters by Reconstruction. *IEEE Transactions on Image Processing*, pages 1153–1160, 4.
- [456] C. Samson, L. Blanc-Feraud, G. Aubert, and J. Zerubia. A Level Set Model for Image Classification. In *International Conference on Scale-Space Theories in Computer Vision*, pages 306–317, 1999.
- [457] C. Samson, L. Blanc-Feraud, G. Aubert, and J. Zerubia. A Level Set Model for Image Classification. *International Journal of Computer Vision*, 40:187–197, 2000.
- [458] G. Sapiro. *Geometric Partial Differential Equations in Image Processing*. Cambridge University Press, 2001.
- [459] G. Sapiro, R. Kimmel, D. Shaked, B. Kimia, and A. Bruckstein. Implementing Continuous-scale Morphology via Curve Evolution. *Pattern Recognition*, 26:1363–1372, 1993.
- [460] G. Sapiro and D. Ringarch. Anisotropic diffusion of multivalued images with applications to color filtering. *IEEE Transactions on Image Processing*, 5:1582–1585, 1996.
- [461] G. Sapiro and A. Tannenbaum. Affine Invariant Scale Space. *International Journal of Computer Vision*, 11:25–44, 1993.

- [462] G. Sapiro and A. Tannenbaum.Area and length preserving geometric invariant scale-space.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:67–72, 1995.
- [463] V. Savchenko, A. Pasko, O. Okunev, and T. Kunii.Function Representation of Solids Reconstructed from Volume.*Computer Graphics Forum*, 14:181–188, 1995.
- [464] H. Schwarz.*Numerische Mathematik*.Teubner, 1988.
- [465] T. Sebastian, P. Klein, and B. Kimia.Recognition of Shapes by Editting Shock Graphs.In *IEEE International Conference in Computer Vision*, pages 755–762, 2001.
- [466] R. Sedgewick.*Algorithms in C++: Fundamentals, Data Structures, Sorting, Searching*.Addison-Wesley, 1988.
- [467] J. Serra.*Image Analysis and Mathematical Morphology*.Academic Press, 1982.
- [468] J. Serra.*Image Analysis and Mathematical Morphology*, volume 2.Academic Press, 1982.
- [469] J. Serra.Connections for Sets and Functions.*Fundamentae Informatica*, 41:147–186, 2000.
- [470] J. Sethian.Curvature and the evolution of fronts.*Communications in Mathematical Physics*, 101:487–499, 1985.
- [471] J. Sethian.A Review of the Theory, Algorithms, and Applications of Level Set Methods for Propagating Interfaces.*Cambridge University Press*, pages 487–499, 1995.
- [472] J. Sethian.*Level Set Methods*.Cambridge University Press, 1996.
- [473] J. Sethian.A Marching Level Set Method for Monotonically Advancing Fronts.*Proceedings of the National Academy of Science*, 93:1591–1595, 1996.
- [474] J. Sethian.Theory, algorithms, and applications of level set methods for propagating interfaces.*Acta Numerica*, pages 309–395, 1996.
- [475] J. Sethian.Fast Marching Methods.*SIAM Review*, 41:199–235, 1999.
- [476] J. Sethian.*Level Set Methods and Fast Marching Methods*.Cambridge University Press, second edition edition, 1999.
- [477] J. Shah.A Common Framework for Curve Evolution, Segmentation and Anisotropic Diffusion.In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 136–142, 1996.
- [478] J. Shah.Riemannian Drums, Anisotropic Curve Evolution, and Segmentation.In *International Conference on Scale-Space Theories in Computer Vision*, pages 129–140, 1999.
- [479] J. Shi and J. Malik.Normalized Cuts and Image Segmentation.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.
- [480] C. Shu and S. Osher.Efficient Implementation of Essentially Non-Oscillatory Shock Capturing Schemes.*Journal of Computational Physics*, 77:439–471, 1988.
- [481] C. Shu and S. Osher.Implementation of Essentially Non-Oscillatory Shock Capturing Schemes II.*Journal of Computational Physics*, 83:32–78, 1989.
- [482] K. Siddiqi, Y.-B. Lauziere, A. Tannenbaum, and S. Zucker.Area and Length Minimizing Flow for Shape Segmentation.*IEEE Transactions on Image Processing*, 7:433–443, 1998.

- [483] K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker. Shocks Graphs and Shape Matching. *International Journal of Computer Vision*, 35:13–32, 1999.
- [484] E. Sifakis, C. Garcia, and G. Tziritas. Bayesian Level Sets for Image Segmentation. *Journal of Visual Communication and Image Representation*, 13:44–64, 2002.
- [485] E. Sifakis, I. Grinias, and G. Tziritas. Video segmentation using fast marching and region growing algorithms. *EURASIP Journal on Applied Signal Processing*, pages 379–388, 2002.
- [486] E. Sifakis and G. Tziritas. Fast marching to moving object location. In *International Conference on Scale-Space Theories in Computer Vision*, pages 447–452, 1999.
- [487] E. Sifakis and G. Tziritas. Moving object localisation using a multi-label fast marching algorithm. *Signal Processing: Image Communication*, 16:963–976, 2001.
- [488] T. Sikora. The MPEG-4 video standard verification model. *IEEE Transactions on Circuits and Systems for Video Technology*, 7:19–31, 1997.
- [489] L. Simon. *Lectures on Geometric Measure Theory*. Australian National University, 1984.
- [490] N. Sochen, R. Kimmel, and R. Malladi. A general framework for low level vision. *IEEE Transactions on Image Processing*, 7:310–318, 1998.
- [491] P. Souganidis. Approximation Schemes for Viscosity Solutions of Hamilton-Jacobi Equations. *Journal of Differential Equations*, 59:1–43, 1985.
- [492] L. Staib and S. Duncan. Boundary finding with parametrically deformable models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:1061–1075, 1992.
- [493] J. Stam. Stable Fluids. In *ASM SIGGRAPH*, pages 121–128, 1999.
- [494] J. Stam and E. Fiume. Turbulent Wind Fields for Gaseous Phenomena. In *ACM SIGGRAPH*, pages 369–376, 1993.
- [495] A. Staniforth and J. Cote. Semi-Lagrangian Integration Schemes for Atmospheric Models - A Review. *Monthly Weather Review*, 119:2206–2223, 1991.
- [496] C. Stauffer and W. Grimson. Adaptive Background Mixture Models for Real-time Tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages II:246–252, Colorado, USA, 1999.
- [497] J. Steinhoff and D. Underhill. Modification of the Euler Equations for *vorticity confinement*: Application to the Computation of Interacting Vortex Rings. *Physics of Fluids*, 6:2738–2744, 1994.
- [498] J. Strain and J. Sethian. Crystal growth and dendritic solidification. *Journal of Computational Physics*, 98:231–253, 1992.
- [499] M. Struwe. On the evolution of harmonic mappings of riemannian surfaces. *Comment. Math. Helvetici*, pages 558–581, 1985.
- [500] M. Sussman, A. Almgren, J. Bell, P. Colella, L. Howell, and M. Welcome. An adaptive level set approach for incompressible two-phase flow. *Journal of Computational Physics*, 148:81–124, 1999.
- [501] M. Sussman, P. Smereka, and S. Osher. A Level Set Method for Computing Solutions to Incompreensible Two-Phase Flow. *Journal of Computational Physics*, 114:146–159, 1994.

- [502] H. Tagare.Deformable 2-D template matching using orthogonal curves.*IEEE Transactions on Medical Imaging*, pages 108–117, 1997.
- [503] I. Tamanini.Optimal Approximation by Piecewise Constant Functions.*Progress in Nonlinear Differential Equations and Their Applications*, pages 73–85, 1996.
- [504] I. Tamanini and G. Congedo.Optimal Segmentation of Unbounded Functions.*Rend. Sem. Mat. Univ. Padova*, pages 153–174, 1996.
- [505] B. Tang, G. Sapiro, and V. Caselles.Diffusion of general data on non-flat manifolds via harmonic maps theory: The direction diffusion case.*International Journal of Computer Vision*, 36:149–161, 2000.
- [506] B. Tang, G. Sapiro, and V. Caselles.Color Image Enhancement via Chromaticity Diffusion.*IEEE Transactions on Image Processing*, 10:701–707, 2001.
- [507] G. Taubin.Estimination of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:1115–1138, 1991.
- [508] G. Taubin.Estimating the tensor of curvature of a surface from a polyhedral approximation.In *IEEE International Conference in Computer Vision*, pages 852–857, 1995.
- [509] G. Taubin.A signal processing approach to fair surface design.In *ACM SIGGRAPH*, pages 351–358, 1995.
- [510] H. Tek and B. Kimia.Image Segmentation by Reaction-Diffusion Bubbles.In *IEEE International Conference in Computer Vision*, pages 156–162, 1995.
- [511] R. Temam.Sur l'approximation de la solution des équations de Navier–Stokes par la méthode de pas fractionnaires (I).*Archive for Rational Mechanics and Analysis*, 32:135–153, 1969.
- [512] D. Terzopoulos.On Matching Deformable Models to Images.Techical Report 60, Schlumberger Palo Alto Research, 1986.
- [513] D. Terzopoulos.The Computation of Visible-Surface Representations.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(4):417–438, 1988.
- [514] D. Terzopoulos and K. Fleischer.Deformable Models.*The Visual Computer*, 4:306–331, 1988.
- [515] D. Terzopoulos and K. Fleischer.Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture.In *ACM SIGGRAPH*, pages 269–278, 1988.
- [516] D. Terzopoulos and D. Metaxas.Dynamic 3D Models with Local and Global Deformations: Deformable Superquadrics.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):703–714, 1991.
- [517] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer.Elastically Deformable Models.In *ACM SIGGRAPH*, pages 205–214, 1987.
- [518] D. Terzopoulos and H. Qin.Dynamic NURBS with Geometric Constraints for Interactive Sculpting.*ACM Transactions on Graphics*, 13:103–136, 1994.
- [519] D. Terzopoulos and R. Szeliski.Tracking with Kalman Snakes.In A. Blake and A. Yuille, editors, *Active Vision*, pages 3–20. MIT Press, 1992.
- [520] D. Terzopoulos, A. Witkin, and M. Kass.Constraints on Deformable Models: Recovering 3D Shape and Nonrigid Motion.*Artificial Intelligence*, 36:91–123, 1988.

- [521] J.-P. Thirion.Image matching as a diffusion process: an analogy with maxwell's demons.*Medical Image Analysis*, 2:243–260, 1998.
- [522] R. Thom.*Structural Stability and Morphogenesis*.Benjamin; Reading, 1975.
- [523] D. Thompson.*On Growth and Form*.Dover, 1917.
- [524] Q. Tian and M. Huhns.Algorithms for Subpixel Registration.*Computer Vision, Graphics and Image Processing*, 35:220–233, 1986.
- [525] K. Toyama and A. Blake.Probabilistic Tracking in a Metric Space.In *IEEE International Conference in Computer Vision*, pages 50–59, 2001.
- [526] A. Tsai, A. Yezzi, W. Wells, C. Tempany, D. Tucker, A. Fan, A. Grimson, and A. Willsky.Model-based Curve Evolution Technique for Image Segmentation.In *IEEE Conference on Computer Vision and Pattern Recognition*, volume I, pages 463–468, 2001.
- [527] A. Tsai, A. Yezzi, and A. Willsky.A curve evolution approach to smoothing and segmentation using the Mumford-Shah functional.In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 119–124, 2000.
- [528] A. Tsai, A. Yezzi, and A. Willsky.Curve evolution implementation of the Mumford-Shah functional for image segmentation, denoising, interpolation, and magnification.*IEEE Transactions on Image Processing*, 10:1169–1186, 2001.
- [529] R. Tsai.Rapid and Accurate Computation of the Distance Function Using Grids.*Journal of Computational Physics*, 178:175–195, 2002.
- [530] R. Tsai, P. Burchard, L.-T. Cheng, S. Osher, and G. Sapiro.Dynamic Visibility in an Implicit Framework.Technical Report 02-06, UCLA CAM Report, 2002.
- [531] R. Tsai, L.-T. Cheng, S. Osher, and H.-K. Zhao.Fast Sweeping Algorithms for a Class of Hamilton-Jacobi Equations.Technical Report 0127, UCLA CAM Report, 2001.
- [532] J. Tsitsiklis.Efficient Algorithms for Globally Optimal Trajectories.In *33rd Conference on Decision and Control*, pages 1368–1373, 1994.
- [533] J. Tsitsiklis.Efficient Algorithms for Globally Optimal Trajectories.*IEEE Transactions on Automatic Control*, 40:1528–1538, 1995.
- [534] A. Turing.The chemical basis of morphogenesis.*Philosophical Transactions of the Royal Society*, pages 37–72, 1952.
- [535] G. Turk.Generating textures on arbitrary surfaces using reaction-diffusion.*Computer Graphics*, 21:289–298, 1991.
- [536] G. Turk and J. Ò'Brien.Shape Transformation Using Variational Implicit Functions.In *ACM SIGGRAPH*, pages 335–342, 1999.
- [537] M. Turk and A. Pentland.Eigenfaces for recognition.*Journal of Cognitive Neuroscience*, 3:71–86, 1991.
- [538] S. Turns.*An Introduction to Combustion*.McGraw-Hill, 1996.
- [539] S. Twomey.On the numerical solution of fredholm integral equations of the first kind by the inversion of the linear system produced by quadrature.*Journal of the Association for Computing Machinery*, 10:97–101, 1963.
- [540] S. Twomey.The Application of Numerical Filtering to the Solution of Integral Equations Encountered in Indirect Sensing Measurements.*Journal of Franklin Institute*, 297:95–109, 1965.

- [541] M. Unser.Texture Classification and Segmentation using wavelet frames.*IEEE Transactions on Image Processing*, 4:1549–1560, 1995.
- [542] R. Varga.*Matrix Iterative Analysis*.Prentice Hall, 1962.
- [543] A. Vasilevskiy and K. Siddiqi.Flux Maximizing Geometric Flows.In *IEEE International Conference in Computer Vision*, pages I: 149–154, 2001.
- [544] R. Veltkamp and M. Hagedoorn.State-of-the-art in Shape Matching.Techical Report UU-CS-1999-27, Utrecht University, 1999.<http://webdoc.gwdg.de/ebook/ah/2000/techrep/CS-1997/1997-27.pdf>.
- [545] B. Vemuri and Y. Guo.Snake Pedals: Compact and Versatile Geometric Models with Physics-Based Control.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:445–459, 2000.
- [546] B. Vemuri, S. Huang, S. Sahni, C.M. Leonard, C. Mohr, R. Gilmore, and J. Fitzsimmons.An efficient motion estimator with application to medical image registration.*Medical Image Analysis*, 2:79–98, 1998.
- [547] B. Vemuri, J. Ye, Y. Chen, and C. Leonard.A level-set based approach to image registration.In *IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, pages 86–93, June 2000.
- [548] B. Vemuri, J. Ye, Y. Chen, and C. Leonard.Image Registration via level-set motion: Applications to atlas-based segmentation.*Medical Image Analysis*, 2002.
- [549] L. Vese and T. Chan.Reduced Non-Convex Functional Approximations for Image Restoration & Segmentation.Techical Report 9756, UCLA CAM Report, 1997.
- [550] L. Vese and T. Chan.A study in the BV Space of a Denoising-Deblurring Variational Problem.*Applied Mathematics and Optimization*, 44:131–161, 2001.
- [551] L. Vese and T. Chan.A Multiphase Level Set Framework for Image Segmentation Using the Mumford and Shah Model.*International Journal of Computer Vision*, 50:271–293, 2002.
- [552] L. Vese and S. Osher.The Level Set Method Links Active Contours, Mumford-Shah Segmentation and Total Variation Restoration.Techical Report 0205, UCLA CAM Report, 2002.
- [553] L. Vese and S. Osher.Modeling Textures with Total Variation Minimization and Oscillating Patterns in Image Processing.Techical Report 0219, UCLA CAM Report, 2002.Journal of Scientific Computing (to appear).
- [554] L. Vincent.Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms.*IEEE Transactions on Image Processing*, 2:176–201, 1993.
- [555] P. Viola and W. Wells.Alignment by Maximization of Mutual Information.In *IEEE International Conference in Computer Vision*, pages 16–23, 1995.
- [556] VLSM.1st IEEE Workshop on Variational and Level Set Methods in Computer Vision.2001.Faugeras, O. and Osher, S. and Paragios, N. and Sethian, J. (eds).
- [557] C. Vogel and M. Oman.Iterative methods for total variation denoising.*SIAM Journal on Scientific Computing*, 17:227–238, 1996.
- [558] C.Y. Wang, H. Sun, S. Yada, and A. Rosenfeld.Some Experiments in Relaxation Image Matching Using Corner Features.*Pattern Recognition*, 16:167–182, 1983.
- [559] J. Wang and E. Adelson.Representing Moving Images with Layers.*IEEE Transactions on Image Processing*, 3:625–638, 1994.

- [560] S. Wang and A. Kaufman. Volume-Sampled 3D Modeling. *IEEE Computer Graphics and Applications*, 14:26–32, 1994.
- [561] S. Wang and A. Kaufman. Volume Sculpting. In *Symposium on Interactive 3D Graphics*, pages 151–156, 1995.
- [562] Y. Wang and P. Bhattacharya. Hierarchical Stereo Correspondence Using Features of Gray Connected Components. In *IEEE International Conference on Image Processing*, pages 264–267, 1997.
- [563] Y. Wang and L. Staib. Boundary Finding with Correspondence using Statistical Shape Models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 338–345, 1998.
- [564] Y. Wang and L. Staib. Elastic Model Based Non-rigid Registration Incorporating Statistical Shape Information. In *Medical Imaging Computing and Computer-Assisted Intervention*, pages 1162–1173, 1998.
- [565] J. Weickert. Scale-space properties of nonlinear diffusion filtering with a diffusion tensor. Technical Report 110, University of Kaiserslautern, Germany, 1994.
- [566] J. Weickert. Applications of nonlinear diffusion in image processing and computer vision. *Acta Mathematica Universitatis Comenianae*, 370:33–50, 2001.
- [567] J. Weickert. Efficient image segmentation using partial differential equations and morphology. *Pattern Recognition*, 34:1813–1824, 2001.
- [568] J. Weickert, J. Heers, C. Schnorr, K. Zuiderveld, O. Scherzer, and H. S. Stiehl. Fast parallel algorithms for a broad class of nonlinear variational diffusion approaches. *Journal of Real Time Imaging*, 7:31–45, 2001.
- [569] J. Weickert and C. Schnorr. Variational optic flow computation with a spatio-temporal smoothness constraint. *Journal of Mathematical Imaging and Vision*, 14:245–255, 2001.
- [570] J. Weickert, B. ter Haar Romeny, and M. Viergever. Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Transactions on Image Processing*, 7:398–410, 1998.
- [571] R. Weisenseel, W. Karl, and D. Castañon. A Region-based Alternative for Edge-preserving Smoothing. In *IEEE International Conference on Image Processing*, pages 778–781, 2000.
- [572] W. Welch and A. Witkin. Free-form shape design using triangulated surfaces. In *ACM SIGGRAPH*, pages 247–256, 1994.
- [573] R. Whitaker. A Level Set Approach to 3D Reconstruction from Range Data. *International Journal of Computer Vision*, 29:203–231, 1997.
- [574] R. Whitaker, D. Breen, K. Museth, and N. Soni. Segmentation of Biological Datasets Using a Level-Set Framework. In M. Chen and A. Kaufman, editors, *Volume Graphics 2001*, pages 249–263. Springer Verlag, 2001.
- [575] R. Whitaker and G. Gerig. Vector-valued diffusion. In *Geometry-Driven Diffusion in Computer Vision*, pages 93–133. Kluwer Academic Publishers, 1994.
- [576] R. Whitaker and X. Xue. Variable-Conductance, Level-Set Curvature for Image Denoising. In *IEEE International Conference on Image Processing*, pages 142–145, 2001.
- [577] G. Winkenbach and D. Salesin. Rendering parametric surfaces in pen and ink. In *ACM SIGGRAPH*, pages 469–476, 1996.

- [578] A. Witkin.Scale-space filtering.In *International Joint Conference on Artificial Intelligence*, pages 1019–1021, 1983.
- [579] A. Witkin and P. Heckber.Using particles to sample and control implicit surfaces.In *ACM SIGGRAPH*, pages 269–278, 1994.
- [580] A. Witkin and M. Kass.Reaction-Diffusion Textures.In *ACM SIGGRAPH*, pages 299–308, 1991.
- [581] G. Wyszecki and W.S. Stiles.*Color Science : Concepts and Methods, Quantitative Data and Formulas*.J. Wiley and Sons, 1982.
- [582] B. Wyvill, A. Guy, and E. Galin.Extending the CSG Tree, Warping, Blending and Boolean Operations in an Implicit Surface Modeling System.*Computer Graphics Forum*, 18:149–158, 1999.
- [583] B. Wyvill, C. McPheevers, and G. Wyvill.Data Structure for Soft Objects.*The Visual Computer*, 2:227–234, 1986.
- [584] C. Xu, D. Pham, M. Rettmann, D. Yu, and J. Prince.Reconstruction of the Human Cerebral Cortex from Magnetic Resonance Images.*IEEE Transactions on Medical Imaging*, 18:467–480, 1999.
- [585] C. Xu and J. Prince.Snakes, Shapes, and Gradient Vector Flow.*IEEE Transactions on Image Processing*, 7:359–369, 1998.
- [586] C. Xu, A. Yezzi, and J. Prince.A Summary of Geometric Level-Set Analogues for a General Class of Parametric Active Contour and Surface Models.In *IEEE Workshop in Variational and Level Set Methods*, pages 104–111, 2001.
- [587] N. Yanenko.*The Method of Fractional Steps: the Solution of Problems of Mathematical Physics in Several Variables*.Springer, 1971.
- [588] L. Yaroslavsky and M. Eden.*Fundamentals of Digital Optics*.Birkhäuser, 1996.
- [589] A. Yezzi.Modified curvature motion for image smoothing and enhancement.*IEEE Transactions on Image Processing*, 7:345–352, 1998.
- [590] A. Yezzi, S. Kichenassamy, A. Kumar, P. Olver, and A. Tannenbaum.A geometric snake model for segmentation of medical imagery.*IEEE Transactions on Medical Imaging*, 16:199–209, 1997.
- [591] A. Yezzi and S. Soatto.Stereoscopic Segmentation.In *IEEE International Conference in Computer Vision*, pages I:56–64, 2001.
- [592] A. Yezzi, A. Tsai, and A. Willsky.Binary Flows and Image Segmentation.In *IEEE International Conference on Image Processing*, page 26AS1, 1999.
- [593] A. Yezzi, A. Tsai, and A. Willsky.A Statistical Approach to Snakes for Bimodal and Trimodal Imagery.In *IEEE International Conference in Computer Vision*, pages 898–903, 1999.
- [594] A. Yezzi, A. Tsai, and A. Willsky.A Fully Global Approach to Image Segmentation via Coupled Curve Evolution Equations.*Journal of Visual Communication and Image Representation*, 13:195–216, 2002.
- [595] A. Yezzi, L. Zollei, and T. Kapur.A Variational Framework for Joint Segmentation and Registration.In *IEEE Mathematical Methods in Biomedical Image Analysis*, pages 44–51, 2001.
- [596] G. Yngve and G. Turk.Creating smooth implicit surfaces from polygonal meshes. Technical Report 99-42, Graphics, Visualization, and Usability Center, Georgia Institute of Technology, 1999.

- [597] Y. You, W. Zhu, A. Tannenbaum, and M. Kaveh. Behavioral analysis of anisotropic diffusion. *IEEE Transactions on Image Processing*, 5:1539–1553, 1996.
- [598] L. Younes. Computable elastic distances between shapes. *SIAM Journal of Applied Mathematics*, 58:565–586, 1998.
- [599] D. Young. *Iterative Solution of Large Linear Systems*. Academic Press, 1971.
- [600] T. Young and K.S. Fu. *Handbook of pattern recognition and image processing*. Academic Press, 1986.
- [601] A. Yuille. Deformable templates for face recognition. *Journal of Cognitive Neuroscience*, 3:59–70, 1991.
- [602] A. Yuille, P. Hallinan, and D. Cohen. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, pages 99–111, 1992.
- [603] X. Zeng, L. Staib, R. Schultz, and J. Duncan. Segmentation and Measurement of the cortex from 3D MR images using coupled surfaces propagation. *IEEE Transactions on Medical Imaging*, 18:100–111, 1999.
- [604] D. Zhang and M. Hebert. Harmonic maps and their applications in surface matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages II: 524–530, 1999.
- [605] H.-K. Zhao. Analysis and Visualization of Large set of Unorganized Data Points Using the Distance Function. *preprint*, 2002.
- [606] H.-K. Zhao. Fast Sweeping Method for Eikonal Equations I: Distance Function. *preprint*, 2002. Submitted: SIAM Journal on Numerical Analysis.
- [607] H-K. Zhao, T. Chan, B. Merriman, and S. Osher. A variational Level Set Approach to Multiphase Motion. *Journal of Computational Physics*, 127:179–195, 1996.
- [608] H-K. Zhao, S. Osher, and R. Fedkiw. Fast Surface Reconstruction Using the Level Set Method. In *IEEE Workshop in Variational and Level Set Methods*, pages 194–202, 2001.
- [609] H-K. Zhao, S. Osher, and R. Fedkiw. Implicit Surface Reconstruction Using the Level Set Method. Technical Report 01-01, UCLA, 2001.
- [610] H-K. Zhao, S. Osher, B. Merriman, and M. Kang. Implicit and Nonparametric Shape Reconstruction from Unorganized Data Using a Variational Level Set Method. *Computer Vision and Image Understanding*, 80:295–314, 2000.
- [611] S. Zhu, T. Lee, and A. Yuille. Region Competition: Unifying Snakes, Region Growing Energy/Bayes/MDL for Multi-band Image Segmentation. In *IEEE International Conference in Computer Vision*, pages 416–423, 1995.
- [612] S. Zhu and A. Yuille. FORMS: A flexible object recognition and modeling system. *International Journal of Computer Vision*, 20:187–212, 1996.
- [613] S. Zhu and A. Yuille. Region Competition: Unifying Snakes, Region Growing, and Bayes/MDL for Multiband Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:884–900, 1996.