Advances in Computer Vision and Pattern Recognition

Klaus D. Toennies

Guide to Medical Image Analysis

Methods and Algorithms

Second Edition

Springer

Advances in Computer Vision and Pattern Recognition

Founding editor

Sameer Singh, Rail Vision, Castle Donington, UK

Series editor

Sing Bing Kang, Microsoft Research, Redmond, WA, USA

Advisory Boards

Horst Bischof, Graz University of Technology, Austria
Richard Bowden, University of Surrey, Guildford, UK
Sven Dickinson, University of Toronto, ON, Canada
Jiaya Jia, The Chinese University of Hong Kong, Hong Kong
Kyoung Mu Lee, Seoul National University, South Korea
Yoichi Sato, The University of Tokyo, Japan
Bernt Schiele, Max Planck Institute for Computer Science, Saarbrücken, Germany
Stan Sclaroff, Boston University, MA, USA

More information about this series at http://www.springer.com/series/4205

Klaus D. Toennies

Guide to Medical Image Analysis

Methods and Algorithms

Second Edition

Springer

Klaus D. Toennies
Computer Science Department, ISG
Otto-von-Guericke-Universität Magdeburg
Magdeburg
Germany

The author(s) has/have asserted their right(s) to be identified as the author(s) of this work in accordance with the Copyright, Design and Patents Act 1988.

ISSN 2191-6586 ISSN 2191-6594 (electronic)
Advances in Computer Vision and Pattern Recognition
ISBN 978-1-4471-7318-2 ISBN 978-1-4471-7320-5 (eBook)
DOI 10.1007/978-1-4471-7320-5

Library of Congress Control Number: 2017932114

© Springer-Verlag London Ltd. 2012, 2017
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer-Verlag London Ltd.
The registered company address is: 236 Gray's Inn Road, London WC1X 8HB, United Kingdom

Preface to the 2nd Edition

The methodology presented in the first edition was considered established practice or settled science in the medical image analysis community in 2010–2011. Progress in this field is fast (as in all fields of computer science) with several developments being particularly relevant to subjects treated in this book:

- Image-based guidance in the operating room is no longer restricted to the display of planning images during intervention. It is increasingly meant to aid the operator to adapt his or her intervention technique during operation. This requires reliable and intuitive analysis methods.
- Segmentation and labeling of images is now mostly treated as solution of an optimization problem in the discrete (Chap. 8) or in the continuous domain (Chap. 9). Heuristic methods such as the one presented in Chap. 6 still exist in non-commercial and commercial software products, but searching for results that optimize an assumption about how the information is mapped to the data produces more predictable methods.
- Deep learning gives new impulses to many areas in medical image analysis as it combines learning of features from data with the abstraction ability of multilayer perceptrons. Hence, learning strategies can be applied directly to pixels in a labeling task. It promises analysis methods that are not designed for a specific problem but can be trained from examples in this problem domain.

Besides actualizing all chapters and removing typos from the first edition, we focused on methods that relate to these three points in the new edition. The general structure of the book has not been changed which also means that the different uses of the book for courses on medical image analysis suggested in the preface for the first edition remain valid. However, the focus of the book on segmentation, classification, and registration has been strengthened further. These tasks are particularly important if image guidance in the operation room is needed since it requires well-understood and reliable tools to extract information from the image data.

In Chaps. 4 and 5, we added established methods for feature generation and edge-preserving smoothing. Feature generation, i.e., the reduction and transformation of image data to a small set of features, is important for efficient and fast extraction of information from images. Edge-preserving smoothing reduces the noise, which is inherent to most imaging techniques, while keeping edges as major

contributors to feature recognition intact. It is a difficult problem, since noise and other artefacts have to be differentiated from relevant edge features prior to determining just these edges.

In the Chaps. 8 and 9, we added methods to include a priori knowledge into graph-based and level set-based segmentation methods. The basic methods in the two chapters represent two strategies to solve segmentation and labeling problems as optimization tasks by minimizing the total variation of an energy functional and have been adopted in many solutions in medical image analysis. In the extended treatment, we present different local characteristics of information and artefacts and how they can be included in the respective energy functional. We also present methods in Chap. 11 how to integrate high-level knowledge about shape, appearance, and expected position of searched objects into the two frameworks presented in Chaps. 8 and 9.

In Chap. 10, we extended the treatment of non-rigid registration and discuss joint segmentation and registration with more depth. As image-guided intervention planning often requires to register organs that move differently with respect to each other, we added a section on locally varying regularizers to model a sliding motion between organs.

Finally, we added a section in Chap. 12 about using deep convolutional networks that describes how this network architecture differs from multilayer perceptrons (MLPs) and how these networks can be used for segmentation and labeling tasks by adding a feature detection and reduction stage to an MLP.

Magdeburg, Germany Klaus D. Toennies

Acknowledgements

Several persons supported me in writing this new edition. I wish to thank my Ph.D. students Georg Hille, Tim König, Marko Rak, and Johannes Steffen who read and corrected part of the chapters. They surely helped to clarify the presentation of the book. Johannes Steffen also contributed to the section on deep learning to make it more concise. I also wish to thank Dr. Laura Astola who used this book in her course and gave me an embarrassingly long list of all the errors that I produced in the first edition. I corrected them all and hopefully produced not too many new typos. Finally, I wish to thank Stefanie Quade for proofreading and correcting the text. I learned a lot about the English language from her, and it certainly improved the readability of the book.

Preface to the 1st Edition

Hans Castorp, in Thomas Mann's Magic Mountain, keeps an X-ray of his love as it seems to him the most intimate of her to possess. Professionals will think different of medical images, but the fascination with the ability to see the unseeable is similar. And, of course, it is no longer just X-ray. Today, it is not sparseness but wealth and diversity of the many different methods to generate images of the human body that make understanding of the depicted content difficult. At any point in time in the last twenty years, at least one or two ways to acquire a new kind of image have been in the pipeline from research to development and application. Currently, optical coherence tomography and MEG are among those somewhere between development and first clinical application. At the same time, established techniques such as CT or MRI reach new heights with respect to depicted content, image quality, or speed of acquisition, opening them to new fields in the medical sciences.

Images are not self-explanatory, however. Their interpretation requires professional skill that has to grow with the number of different imaging techniques. The many case reports and scientific articles about the use of images in diagnosis and therapy bear witness to this. Since the appearance of digital images in the 1970s, information technologies have had a part in this. The task of computer science has been and still is the quantification of information in the images by supporting detection and delineation of structures from an image or from the fusion of information from different image sources. While certainly not having the elaborate skills of a trained professional, automatic or semi-automatic analysis algorithms have the advantage of repeatedly performing tasks of image analysis with constant quality, hence relieving the human operator from the tedious and fatiguing parts of the interpretation task.

By the standards of computer science, computer-based image analysis is an old research field with first applications in the 1960s. Images in general are such a fascinating subject because the data elements contain so little information, while the whole image captures such a wide range of semantics. Just take a picture from your last vacation and look for information in it. It is not just Uncle Harry but also the beauty of the background, the weather and time of the day, the geographical location, and many other kinds of information that can be gained from a collection of pixels of which the only information is intensity, hue, and saturation.

Consequently, a wealth of methods has been developed to integrate the necessary knowledge in an interpretation algorithm for arriving at this kind of semantics.

Although medical images differ from photography in many aspects, similar techniques of image analysis can be applied to extract meaning from medical images. Moreover, the profit from applying image analysis in a medical application is immediately visible, as it saves times or increases reliability of an interpretation task needed to carry out a necessary medical procedure. It requires, however, that the method is selected adequately, applied correctly, and validated sufficiently.

This book originates from lectures about the processing and analysis of medical images for students in computer science and computational visualistics who want to specialize in medical imaging. The topics discussed in the lectures have been rearranged in order to provide a single comprehensive view on the subject. The book is structured according to the potential applications in medical image analysis. It is a different perspective if compared to image analysis, where usually a bottom-up sequence from pixel information to image content is preferred. Wherever it was possible to follow the traditional structure, this has been done. However, if the methodological perspective conflicted with the view from an application perspective, the latter has been chosen. The most notable difference is in the treatment of classification and clustering techniques that appears twice, since different methods are suitable for segmentation in low-dimensional feature space compared to classification in high-dimensional feature space.

The book is intended to be used for medical professionals who want to get acquainted with image analysis techniques, for professionals in medical imaging technology, and for computer scientists and electrical engineers who want to specialize in the medical applications. A medical professional may want to skip the second chapter as he or she will be more intimately acquainted with medical images than the introduction in this chapter can provide. It may be necessary to acquire some additional background knowledge in image or signal processing. However, only the most basic material was omitted (e.g., definition of the Fourier transform and convolution), information about which is freely available on the Internet. An engineer, on the other hand, may want to get more insight into the clinical workflow, in which analysis algorithms are integrated. The topic is presented briefly in this book, but a much better understanding is gained from collaboration with medical professionals. A beautiful algorithmic solution can be virtually useless if constraints from the application are not adhered to.

As it was developed from course material, the book is intended to be used in lectures on the processing and analysis of medical images. There are several possibilities to use subsets of the book for single courses which can be combined. Three of the possibilities that I have tried myself are listed below (Cx refers to the chapter number):

- *Medical Image Generation and Processing*

  (Bachelor course supplemented with exercises to use MATLAB or another toolbox for carrying out image processing tasks)

- C2: Imaging techniques in detail (4 lectures)
- C3: DICOM (1 lecture)
- C4: Image enhancement (2 lectures)
- C5: Feature generation (1 lecture)
- C6: Basic segmentation techniques (2 lectures)
- C12: Classification (1 lecture)
- C13: Validation (1 lecture)

- *Introduction to Medical Image Processing and Analysis*

  (Bachelor course supplemented with a student's project to solve a moderately challenging image analysis task; requires background on imaging techniques):

  - C2: Review of major digital imaging techniques: X-ray, CT, MRI, ultrasound, and nuclear imaging (1 lecture)
  - C3: Information systems in hospitals (1 lecture)
  - C4: Image enhancement (1 lecture)
  - C6: Basic segmentation techniques (2 lectures)
  - C7: Segmentation as a classification task (1 lecture)
  - C8-C9: Introduction to graph cuts, active contours, and level sets (2 lectures)
  - C10: Rigid and non-rigid registration (2 lectures)
  - C11: Active shape model (1 lecture)
  - C13: Validation (1 lecture)

- *Advanced Image Analysis*

  (Master course supplemented with a seminar on hot topics in this field):

  - C7: Segmentation by using MRFs (1 lectures)
  - C8: Segmentation as operation on graphs (3 lectures)
  - C9: Active contours, active surfaces, level sets (4 lectures)
  - C11: Object detection with shape (4 lectures)

Most subjects are presented so that they can also be read on a cursory level omitting derivations and details. This is intentional to allow a reader to understand dependencies of a subject on other subjects without having to go into detail in each one of them. It should also help to teach medical image analysis on the level of a Bachelor's course.

Medical image analysis is a rewarding field for investigating, developing, and applying methods of image processing, computer vision, and pattern recognition. I hope that this book gives the reader a sense of the breadth of this area and its many challenges while providing him or her with the basic tools to take the challenge.

Magdeburg, Germany Klaus D. Toennies

Acknowledgements

There are many who contributed to this book who I wish to thank. First and foremost, there is the Unknown Student. Many of the students who took part in the lectures on which this book is based took a real interest in the subject, even though image processing and image analysis require more background in mathematics than many students care to know. Their interest to understand this subject certainly helped to clarify much of the argumentation.

Then, there are the Ph.D. and Master students, who contributed with their research work to this book. The work of Stefan Al-Zubi, Steven Bergner, Lars Dornheim, Karin Engel, Clemens Hentschke, Karsten Rink, and Sebastian Schäfer produced important contributions in several fields of medical image analysis which have been included in the book. I also wish to thank Stefanie Quade for proof-reading a first version of this book, which certainly improved the readability.

Finally, I wish to thank Abdelmalek Benattayallah, Anna Celler, Tyler Hughes, Sergey Shcherbinin, MeVis Medical Solutions, the National Eye Institute, Siemens Sector Healthcare, and Planilux who provided several of the pictures that illustrate imaging techniques and analysis methods.

The original version of the book frontmatter was revised: Copyright comment text has been added in copyright page. The erratum to the book frontmatter is available at 10.1007/978-1-4471-7320-5_15

Contents

1 The Analysis of Medical Images 1
1.1 Image Analysis in the Clinical Workflow 6
1.2 Using Tools 9
1.3 An Example: Multiple Sclerosis Lesion Segmentation in Brain MRI 15
1.4 Concluding Remarks 21
1.5 Exercises 21
References 22

2 Digital Image Acquisition 23
2.1 X-ray Imaging 26
2.1.1 Generation, Attenuation, and Detection of X-rays 26
2.1.2 X-ray Imaging 31
2.1.3 Fluoroscopy and Angiography 35
2.1.4 Mammography 37
2.1.5 Image Reconstruction for Computed Tomography 38
2.1.6 Contrast Enhancement in X-ray Computed Tomography 46
2.1.7 Image Analysis on X-ray-Generated Images 46
2.2 Magnetic Resonance Imaging 48
2.2.1 Magnetic Resonance 48
2.2.2 MR Imaging 52
2.2.3 Some MR Sequences 55
2.2.4 Artifacts in MR Imaging 57
2.2.5 MR Angiography 58
2.2.6 BOLD Imaging 59
2.2.7 Perfusion Imaging 61
2.2.8 Diffusion Imaging 62
2.2.9 Image Analysis on Magnetic Resonance Images 64
2.3 Ultrasound 66
2.3.1 Ultrasound Imaging 67
2.3.2 Image Analysis on Ultrasound Images 70

2.4 Nuclear Imaging . . . 71
2.4.1 Scintigraphy . . . 71
2.4.2 Reconstruction Techniques for Tomography in Nuclear Imaging . . . 72
2.4.3 Single Photon Emission Computed Tomography (SPECT) . . . 78
2.4.4 Positron Emission Tomography (PET) . . . 79
2.4.5 Image Analysis on Nuclear Images . . . 81
2.5 Other Imaging Techniques . . . 81
2.5.1 Optical Coherence Tomography . . . 81
2.5.2 Photography . . . 82
2.5.3 Optical Microscopy . . . 83
2.5.4 Electron Microscopy . . . 85
2.5.5 EEG and MEG . . . 87
2.6 Concluding Remarks . . . 89
2.7 Exercises . . . 89
References . . . 91

3 Image Storage and Transfer . . . 95
3.1 Information Systems in a Hospital . . . 96
3.2 The DICOM Standard . . . 102
3.3 Establishing DICOM Connectivity . . . 109
3.4 The DICOM File Format . . . 111
3.5 Technical Properties of Medical Images . . . 113
3.6 Displays and Workstations . . . 115
3.7 Compression of Medical Images . . . 121
3.8 Concluding Remarks . . . 122
3.9 Exercises . . . 122
References . . . 123

4 Image Enhancement . . . 125
4.1 Measures of Image Quality . . . 126
4.1.1 Spatial Resolution . . . 126
4.1.2 Definition of Contrast . . . 127
4.1.3 The Modulation Transfer Function . . . 131
4.1.4 Signal-to-Noise Ratio (SNR) . . . 131
4.2 Image Enhancement Techniques . . . 133
4.2.1 Contrast Enhancement . . . 133
4.2.2 Resolution Enhancement . . . 136
4.2.3 Edge Enhancement . . . 139
4.3 Noise Reduction . . . 144
4.3.1 Noise Reduction by Linear Filtering . . . 144
4.3.2 Median Filtering . . . 148
4.3.3 Diffusion Filtering . . . 151
4.3.4 Gradient Adaptive Smoothing . . . 156

4.3.5 Bilateral Filtering . . . 157
4.3.6 Guided Filtering . . . 160
4.3.7 Bayesian Image Restoration . . . 162
4.4 Concluding Remarks . . . 169
4.5 Exercises . . . 169
References . . . 170

5 Feature Detection . . . 173
5.1 Edge Tracking . . . 174
5.2 Hough Transform . . . 179
5.3 Corners . . . 181
5.4 Template Matching . . . 185
5.5 SIFT Feature and SURF . . . 188
5.6 Binary Key Point Descriptor and Detectors . . . 191
5.7 Superpixel . . . 193
5.8 MSER Features . . . 195
5.9 Local Shape Context . . . 196
5.10 Texture . . . 198
5.11 Histogram of Oriented Gradients . . . 201
5.12 Saliency and Gist . . . 202
5.13 Concluding Remarks . . . 203
5.14 Exercises . . . 204
References . . . 205

6 Segmentation: Principles and Basic Techniques . . . 209
6.1 Segmentation Strategies . . . 210
6.2 Data Knowledge . . . 213
6.2.1 Homogeneity of Intensity . . . 215
6.2.2 Homogeneity of Texture . . . 217
6.3 Domain Knowledge About the Objects . . . 218
6.3.1 Representing Domain Knowledge . . . 220
6.3.2 Variability of Model Attributes . . . 221
6.3.3 The Use of Interaction . . . 222
6.4 Interactive Segmentation . . . 225
6.5 Thresholding . . . 226
6.6 Homogeneity-Based Segmentation . . . 232
6.7 The Watershed Transform: Computing Zero Crossings . . . 234
6.8 Seeded Regions . . . 237
6.9 Live Wire . . . 240
6.10 Concluding Remarks . . . 244
6.11 Exercises . . . 245
References . . . 246

7 Segmentation in Feature Space 249
7.1 Segmentation by Classification in Feature Space 251
7.1.1 Computing the Likelihood Function 252
7.1.2 Multi-dimensional Feature Vectors 256
7.1.3 Computing a Priori Probability 258
7.1.4 Extension to More Than Two Classes 260
7.2 Clustering in Feature Space 260
7.2.1 Partitional Clustering and K-means Clustering. 261
7.2.2 Mean Shift Clustering 264
7.2.3 Kohonen's Self-organizing Maps. 267
7.3 Concluding Remarks. 270
7.4 Exercises. 271
References. 272

8 Segmentation as a Graph Problem. 275
8.1 Graph Cuts 277
8.1.1 Graph Cuts for Computing a Segmentation 278
8.1.2 Graph Cuts to Approximate a Bayesian Segmentation 284
8.1.3 Problem-Specific Adaptation of *t*-Link Weights. 289
8.1.4 Adding Generic Domain Knowledge 291
8.1.5 Normalized Graph Cuts 293
8.2 Segmentation as a Path Problem. 296
8.2.1 Fuzzy Connectedness. 296
8.2.2 The Image Foresting Transform. 298
8.2.3 Random Walks 300
8.2.4 Other Methods Based on Seed Placements 304
8.3 Similarities and Integrations of Graph-Based Strategies 305
8.4 Concluding Remarks. 306
8.5 Exercises. 306
References. 307

9 Active Contours and Active Surfaces. 311
9.1 Explicit Active Contours and Surfaces 313
9.1.1 Deriving the Model 313
9.1.2 The Use of Additional Constraints. 316
9.1.3 T-Snakes and T-Surfaces 319
9.2 The Level Set Model 321
9.2.1 Level Sets 322
9.2.2 Level Sets and Wave Propagation 324
9.2.3 Schemes for Computing Level Set Evolution 329
9.2.4 Computing Stationary Level Set Evolution 333
9.2.5 Computing Dynamic Level Set Evolution 336
9.2.6 Segmentation and Speed Functions 338
9.2.7 Geodesic Active Contours 341

9.2.8 Level Sets and the Mumford–Shah Functional 343
9.2.9 Avoiding Reinitialization 351
9.2.10 Noise Reduction and Boundary Enhancement 352
9.2.11 Initial Level Sets and Initial Regions of Interest 353
9.2.12 Topologically Constrained Level Sets 354
9.3 Concluding Remarks 355
9.4 Exercises 356
References 357

10 Registration and Normalization 361
10.1 Feature Space and Correspondence Criterion 363
10.1.1 Correspondence Criteria for Extrinsic and Intrinsic Markers 364
10.1.2 Requirements for Extrinsic and Intrinsic Markers 367
10.1.3 Similarity for Image-Based Features 370
10.2 Rigid Registration 375
10.2.1 Registration Using Paired Landmarks 375
10.2.2 Registration with Unpaired Landmarks 378
10.2.3 Outlier Detection 379
10.2.4 Registration in the Frequency Domain 380
10.2.5 The Head-Hat Registration 381
10.2.6 The ICP Algorithm 382
10.3 Registration of Projection Images to 3d Data 383
10.3.1 Registration Using Digitally Reconstructed Radiographs 384
10.3.2 Optimization of the Similarity Criterion 386
10.4 Search Space and Optimization in Non-rigid Registration 387
10.4.1 Elastic Registration 388
10.4.2 Using a Viscous Fluid Model 388
10.4.3 Free-Form Deformations 390
10.4.4 Neighborhood Similarity 390
10.4.5 Multi-scale Registration 391
10.5 Normalization 392
10.6 Joint Segmentation and Registration 395
10.7 Non-rigid Discontinuous Registration 397
10.8 Concluding Remarks 400
10.9 Exercises 400
References 401

11 Shape, Appearance and Spatial Relationships 405
11.1 Shape Models 407
11.2 Simple Models 411
11.2.1 Template Matching 411
11.2.2 Hough Transform 413
11.3 Implicit Models 415

11.4 The Medial Axis Representation 417
11.4.1 Computation of the Medial Axis Transform 417
11.4.2 Shape Representation by Medial Axes. 419
11.5 Active Shape and Active Appearance Models 420
11.5.1 Creating an ASM. 421
11.5.2 Using ASMs for Segmentation 424
11.5.3 The Active Appearance Model 426
11.5.4 Deficiencies of Statistical Shape Models 427
11.5.5 Changing the Parametric Model. 429
11.5.6 Using a Part-Based Model. 429
11.5.7 Using a Non-parametric Model 430
11.6 Physically-Based Shape Models 432
11.6.1 Mass-Spring Models 434
11.6.2 Finite Element Models. 439
11.7 Shape, Appearance and Pose Priors for Segmentation 453
11.7.1 Atlas-Based Segmentation 453
11.7.2 Combining Shape Information with Level Set Segmentation 455
11.7.3 Solutions Based on Graph Cuts. 460
11.7.4 Graph Cuts on Pre-segmented Images 461
11.8 Using a Graphical Model for Detecting Object Parts 463
11.9 Concluding Remarks. 465
11.10 Exercises. 466
References. 468

12 Classification and Clustering 473
12.1 Features and Feature Space. 474
12.1.1 Linear Decorrelation of Features 475
12.1.2 Linear Discriminant Analysis. 477
12.1.3 Independent Component Analysis 478
12.2 Bayesian Classifier 480
12.3 Classification Based on Distance to Training Samples 482
12.4 Decision Boundaries 485
12.4.1 Adaptive Decision Boundaries. 486
12.4.2 The Multilayer Perceptron 488
12.4.3 Support Vector Machines 495
12.5 Convolutional Neural Networks and Deep Learning 498
12.5.1 Structure of a CNN 499
12.5.2 What Does a CNN Learn 502
12.5.3 Applications in Medical Image Analysis 504
12.6 Classification by Association 506
12.7 Clustering Techniques. 508
12.7.1 Agglomerative Clustering 508
12.7.2 Fuzzy c-Means Clustering 510

12.8 Bagging and Boosting . . . 512
12.9 Decision Trees . . . 514
12.9.1 Creating a Decision Tree . . . 514
12.9.2 Random Forest . . . 516
12.10 Multiple Instance Learning . . . 516
12.11 Bag of Visual Words . . . 518
12.12 Dealing with Imbalanced Data . . . 521
12.13 Concluding Remarks . . . 524
12.14 Exercises . . . 524
References . . . 526

13 Validation . . . 529
13.1 Measures of Quality . . . 531
13.1.1 Quality for a Delineation Task . . . 532
13.1.2 Considering Interaction in a Segmentation Task . . . 536
13.1.3 Quality for a Detection Task . . . 537
13.1.4 Quality for a Registration Task . . . 541
13.2 The Ground Truth . . . 542
13.2.1 Ground Truth from Real Data . . . 543
13.2.2 Ground Truth from Phantoms . . . 546
13.3 Representativeness of Data . . . 552
13.3.1 Separation Between Training and Test Data . . . 552
13.3.2 Identification of Sources of Variation and Outlier Detection . . . 553
13.3.3 Robustness with Respect to Parameter Variation . . . 555
13.4 Significance of Results . . . 555
13.5 Concluding Remarks . . . 559
13.6 Exercises . . . 559
References . . . 560

14 Appendix . . . 563
14.1 Optimization of Markov Random Fields . . . 563
14.1.1 Markov Random Fields . . . 563
14.1.2 Simulated Annealing . . . 566
14.1.3 Mean Field Annealing . . . 568
14.1.4 Iterative Conditional Modes . . . 569
14.2 Variational Calculus . . . 569
14.3 Principal Component Analysis . . . 572
14.3.1 Computing the PCA . . . 572
14.3.2 Robust PCA . . . 575
References . . . 577

Erratum to: Guide to Medical Image Analysis . . . E1

Index . . . 579

Abbreviations

AAM	Active appearance model
AC	Anterior commissura
ACR	American College of Radiology
AdaBoost	Adaptive boosting
AE	Application entity (DICOM)
AHE	Adaptive histogram equalization
ANSI	American National Standard Institute
ART	Algebraic reconstruction technique
ASM	Active shape model
ASSM	Active structural shape model
AVM	Arteriovenous malformation
Bagging	Bootstrap aggregating
BOLD	Blood oxygen level dependency
BRIEF	Binary robust independent elementary features
CAT	Computed axial tomography (=CT)
CCA	Connected component analysis
CLAHE	Contrast limited adaptive histogram equalization
CNN	Convolutional neural network
CRT	Cathode ray tube
CSF	Cerebrospinal fluid
CT	Computed tomography
CTA	CT angiography
dCNN	Deep CNN
DICOM	Digital communication in medicine
DIMSE	DICOM message service element
DoG	Difference of Gaussians
DRR	Digitally reconstructed radiograph
DSA	Digital subtraction angiography
DSM	Deformable shape model
DTI	Diffusion tensor imaging
EEG	Electroencephalogram
EM	Expectation maximization
EPI	Echoplanar imaging

eV	Electron volt (unit)
FAST	Features from accelerated segment test
FBP	Filtered backprojection
FEM	Finite element model
FID	Free induction decay
fMRI	Functional MRI
FOM	Figure of merit
FOV	Field of view
FUS	Focused ultrasound
GLCM	Gray-level co-occurrence matrix
GM	Gray matter
GVF	Gradient vector flow
HIS	Hospital information system
HL7	Health level 7
HOM	Hit-or-miss operator
HU	Hounsfield unit
ICA	Independent component analysis
ICM	Iterative conditional modes
ICP	Iterative closest point
IFT	Image foresting transform
IOD	Information object description (DICOM)
ISO	International Standards Organization
ITK	Insight Toolkit
IVUS	Intravenous ultrasound
KDE	Kernel density estimator
keV	Kilo-electron volt (unit)
kNN	k-nearest-neighborhood
LBP	Local binary pattern
LDA	Linear discriminant analysis
LoG	Laplacian-of-Gaussian
LOGISMOS	Layered optimal graph image segmentation of multiple objects and surfaces
LOR	Line of response
lpmm	Line pairs per millimeter
m-rep	Medial axis representation
MAP-EM	Maximum a posteriori-expectation maximization
MEG	Magnetoencephalogram
MIL	Multiple instance learning
MIP	Maximum intensity projection
MLEM	Maximum likelihood expectation maximization (reconstruction)
MRA	MR angiography
MRF	Markov random field
MRI	Magnetic resonance imaging
MSER	Maximally stable extremal regions
MTF	Modulation transfer function

MTT	Mean transit time
mWST	Marker-based watershed transform
NEMA	National Electrical Manufacturers Association
OCT	Optical coherence tomography
OpenCV	Open source computer vision
ORB	Oriented FAST and rotated BRIEF
OSEM	Ordered set expectation maximization
OSI	Open Systems Interconnect
OSL	One-step-late (algorithm)
PACS	Picture archiving and communication system
PC	Posterior commissura
PCA	Principal component analysis
PDM	Point distribution model
PET	Positron emission tomography
PSF	Point spread function
PVE	Partial volume effect
QoF	Quality of fit
R	Roentgen (unit)
rad	Radiation absorbed dose (unit)
RAG	Region adjacency graph
RARE	Rapid enhancement with relaxation enhancement
rCBF	Relative cerebral blood volume
rCBV	Relative cerebral blood volume
RIS	Radiology information system
rms	Root mean square
ROC	Receiver operator characteristic
ROI	Region of interest
SCP	Service class provider (DICOM)
SCU	Service class user (DICOM)
SEM	Scanning electron microscopy
SIFT	Scale-invariant feature transform
SLIC	Simple linear iterative clustering
SMOTE	Synthetic minority oversampling technique
SNR	Signal-to-noise ratio
SOP	Service object pair (DICOM)
SPECT	Single photon emission tomography
SPM	Statistical parametric mapping
SSM	Statistical shape model
STAPLE	Simultaneous truth and performance level estimation
SURF	Speeded-up robust features
SUSAN	Smallest univalue segment assimilating nucleus
SVM	Support vector machine
T	Tesla (unit)
TE	Echo time (MRI)
TEM	Transmission electron microscopy

teUS	Temporally enhanced ultrasound
TFT	Thin film transistor
TI	Inversion time (MRI)
TR	Repetition time (MRI)
TRUS	Transrectal ultrasound
UID	Unique identifier
US	Ultrasound
WM	White matter
WST	Watershed transform

The Analysis of Medical Images

1

Abstract

Medical images are different from other pictures in that they depict distributions of various physical features measured from the human body. They show attributes that are otherwise inaccessible. Furthermore, analysis of such images is guided by very specific expectations which gave rise to acquiring the images in the first place. This has consequences on the kind of analysis and on requirements for algorithms that carry out some or all of the analysis. Image analysis as part of the clinical workflow will be discussed in this chapter as well as the types of tools that exist to support the development and carrying out such an analysis. We will conclude with an example for the solution of an analysis task in order to illustrate important aspects for the development of methods for analyzing medical images.

Concepts, Notions, and Definitions Introduced in this Chapter

- Introduction to basic development strategies;
- Common analysis tasks: delineation, object detection, classification;
- Image analysis for cohort studies, clinical studies, diagnosis support, treatment planning, and computer-assisted therapy;
- Tool types: viewers, workstation software, development tools.

Why is there a need for a book on medical image analysis when there are plenty of good texts on image analysis around? Medical images differ from photography in many ways. Consider the picture in Fig. 1.1 and potential questions and problems related to its analysis. The first question that comes to mind would probably be to detect certain objects (e.g., persons). Common problems that have to be solved are to recover the 3d information, i.e., missing depth information and the true shape, to separate illumination effects from object appearance, to deal with partially hidden objects, and to track objects over time.

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_1

Fig. 1.1 Analysis questions for a photography are often based on a detection or tracking task (such as detecting real persons in the image). Problems relate to reducing effects from the opacity of most depicted objects and to the reconstruction of depth information (real persons are different from those on the picture because they are 3d, and—if a sequence of images is present—because they can move)

Medical images are different. Consider the image in Fig. 1.2. The appearance of the depicted object is not caused by light reflection but from the absorption of X-rays. The object is transparent with respect to the depicted physical attribute. Although detection of some structure may be goal of the analysis, exact delineation of the object and its substructures may be the first task. Variation of object shape and appearance may be characteristic for some evaluation and needs to be captured. Furthermore, this is not the only way to gain insight in the human body. Different imaging techniques produce mappings of several physical attributes in various ways that may be subjects of inspection (compare Fig. 1.2 with Fig. 1.3). Comparing this information with reality is difficult; however, since few if any non-invasive methods exist to verify the information gained from the pictures. Hence, the focus on analysis methods for medical images is different if compared to the analysis of many other images. Delineation, restoration, enhancement, and registration for fusing images from different sources are comparably more important than classification, reconstruction of 3d information, and tracking (although it does not mean that the last three topics are irrelevant for medical image analysis). This shift in focus is reflected in our book and leads to the following structure:

- Medical images, their storage and use will be discussed in Chaps. 2 and 3.
- Enhancement techniques and feature computation will be the subject of Chaps. 4 and 5.
- Delineation of object boundaries, finding objects, and registering information from different sources will make up the majority of the book. It will be presented in Chaps. 6–12.

Fig. 1.2 Detail of one of the first radiographs showing wrist and part of the hand. The image projects a physical entity that is transparent with respect to the detection technique. Bone structures are clearly visible

Fig. 1.3 Detail of a slice through similar region than the one depicted in Fig. 1.2 by a magnetic resonance image (MRI). The acquisition technique produces a 3d volume and the imaged physical entity highlights soft tissue structures (from the website http://www.exeter.ac.uk/~ab233/, with kind permission of Dr. Abdelmalek Benattayallah)

Fig. 1.4 Different tasks in medical image analysis require different methodology and validation techniques

- A separate Chap. 13 will be devoted to the validation of an analysis procedure, since this is particularly difficult for methods developed for medical images.

Computer-assisted analysis of medical images is meant to support an expert (the radiologist, the surgeon, etc.) in some decision task. It is possible to associate analysis tasks to the kind of decision in which the expert shall be supported (see Fig. 1.4):

- *Delineation* of an object requires solving a segmentation task.
- *Detection* of an object requires solving a classification task.
- Comparison of object appearance from pictures at different times or from different modalities requires solving a *registration* task.

Although the characterization above is helpful in deciding where to look for solutions, practical applications usually involve aspects from not just one of the fields. Hence, before deciding on the kind of methodology that is needed, it is important to understand the technical questions associated with the practical application. Several aspects need to be discussed:

- Analysis in the *clinical work flow*: How does the analysis fit into the clinical routine within which it has been requested?
- *Strategies* to develop an analysis tool: How can it be assured that an efficient and effective solution has been chosen?
- Acquisition of necessary a priori *information*: What kind of information is necessary to solve the analysis task and how can it be acquired?
- Setup of an *evaluation scenario*: How can the quality of the analysis method be tested?
- *Tools* to support an analysis task: How can tools be used to spend as little effort as necessary to solve the analysis task?

It is easy to forget about these more general questions when being confronted with some analysis problem, because

- each analysis task is different from most other analysis tasks (different organ or pathology to be analyzed, different imaging system, imaging parameters, imaging equipment).
- when being asked, finding a solution fast is often the main motivation and other aspects such as fitting the solution into the workflow appear to be of lesser importance.
- development of a method usually takes place well separated from the regular practice in which the method is supposed to be used.
- it is more fun to experiment with some new methods or to apply a methodology with which the developer has experience rather than truly looking for the most appropriate solution.

Nonetheless, the final result needs to be an effective and ideally efficient solution. The following sections will present strategies for developing a method in the clinical environment, for deciding on the type of methodology and for deciding on an evaluation scenario.

Digital medical images and computer-assisted methods for their interpretation have been around for quite some time. Several books exist that treat the subject. Books on specific aspects of medical image analysis will be referenced in the respective chapters. Books on medical image analysis in general are the following:

- The Handbook of Medical Imaging and Analysis (Bankman 2008) is an edited book containing almost 60 articles of active researchers in the field on all relevant aspects of the topic. It is a well-structured text of more than 1000 pages.
- Handbook of Biomedical Imaging: Methodologies and Clinical Research (Paragios et al. 2015) is an edited book that in its first part reviews current strategies for medical image analysis followed by parts comprising general treatment and specific examples on segmentation, classification, and registration of medical images.
- Biomedical Image Analysis Recipes in MATLAB: For Life Scientists and Engineers (Rayes-Aldaroso 2015) introduces the use of MATLAB for solving typical image analysis tasks. It is well-suited for those with a good background on basic analysis strategies without having a lot of programming experience who want to put their knowledge on image analysis into practice.
- Medical Image Analysis (Dhawan 2011) is strong on physics, generation, and information content of modern imaging modalities.

There is still a need for another text, since the subject is either treated with focus on the generation of images rather than on their analysis, or the treatment requires a very good background in order to appreciate the information. The book at hand will introduce the subject and present an overview and detailed look at the many dependencies between different strategies for computer-assisted interpretation of medical images.

1.1 Image Analysis in the Clinical Workflow

A newly developed method or a newly adapted method for carrying out some analysis, e.g., for determining the tumor boundary and tumor volume in brain MRI, will most likely not be implemented on the computer that is used to generate or to evaluate the images. The reason is that this piece of software will often not be certified as part of the equipment to be used in clinical routine. Hence, the method will be separated from other analysis devices while still intended to be used within some medical procedure. This has to be accounted for when developing a method. The developer will not only have to create the method but also needs to provide an environment in which the method can be applied. The type of environment depends on the problem that has to be solved. At least five different scenarios can be differentiated (see Table 1.1):

- For a c*linical study*, images are analyzed outside a clinical routine task to understand or confirm findings based on images. In this case, images that are part of the study are often copied to some workstation where the study is taking place. The image analysis method is then implemented on this workstation and results from analysis are kept here as well. Transfer of data to this workstation has to be organized and bookkeeping must enable a later checking of results.
- A relatively new field are images in large *cohort studies* in epidemiology. With the advent of magnetic resonance imaging (MRI) as a non-invasive imaging technique that does not use potentially harmful X-rays it became ethically acceptable to image healthy subjects for the purpose of studying public health on a grand scale. Several such studies have been initiated in recent years (Preim et al. 2016). Analysis methods should be largely automatic since the number of subjects is very large (up to several tens of thousands). Different to a clinical study with its well-specified analysis goals, data acquisition for cohort studies is often started with rather generic goals (such as public health in general or the effects of an aging society). This leads to a number of unique requirements

Table 1.1 Different scenarios for computer-assisted image analysis have very different requirements

	Cohort study	Clinical study	Computer aided diagnosis	Treatment planning	Computer-assisted surgery
No. of cases	Very large	Large	Small	Small	Small
Time-constraints	Low	Low	Medium	Medium	High
Location	Anywhere	Anywhere	Office, reading room	Office, ward	Operating room
Interaction	Not acceptable	Not acceptable	Acceptable	Acceptable	Acceptable
Archival requirements	Very high	High	High	Medium	Medium

because image acquisition is not optimized toward a specific goal, because many analysis questions are developed after image acquisition is terminated, and because image acquisition is carried out at regular time intervals on the same subjects over a long period of time (see Toennies et al. 2015 for a discussion of requirements on image analysis that are specific for cohort studies). A new strategy to deal with the analysis in large image repositories is *radiomics* (Kumar et al. 2012; Gillies et al. 2015). Radiomics adopts analysis techniques from genomics for the analysis of radiological images in that, a large number of fairly general features are generated from each data set and then interpreted either by a human observer to which the feature sets are displayed (similar to genome data) or by subsequent application of machine learning algorithms to the data. The beauty (and danger) of this approach is that the reduction in features constitutes a substantial data reduction making exploration feasible if the selected features capture the information needed for the analysis.

- For *diagnosis support* (computer-aided detection, computer-aided diagnosis), single cases which may consist of several studies containing images are analyzed on a medical workstation. Transfer of images to this workstation is often done by other software, but access to the images, which may be organized by the export module of the image acquisition system, has to be part of the analysis method. Diagnosis support systems often involve interaction, since the user has to accept or reject the analysis result anyway. If interaction is used to supply the method with additional information by the data expert, it has to be ensured that it is intuitive and efficient and that contradictions between results from analysis and any overwriting actions by the user are clear.
- Analysis in *treatment planning* precedes treatment. It may be carried out in a radiology department or at a surgery department, depending on who is doing the treatment planning. Methods have to take the time into account that is acceptable for doing this task (which may be critical in some cases). Furthermore, results from treatment planning are input for the treatment. This input may happen by the medical expert taking numerical results from planning and using it for some kind of parameterization. Generated results should be well-documented, and measures should be enacted that help to avoid mistakes during transfer from results to parameterization. Input into some treatment module may also happen automatically (after acceptance of the result by the expert). The interface for the transfer has then to be specified as well.
- Image analysis for *computer-assisted surgery* is time-critical. Since non-certified software is usually not allowed on the imaging system, fast transfer to the system on which the analysis method is implemented has to be organized. Furthermore, time constraints have to be known and kept by the method. With programmable graphic cards, time constraints are often adhered to by implementing some or all of the analysis method on the graphics hardware. Any constraints from use of the system in an operating theatre have to be considered and adhered to.

Any constraints, such as speed requirements, and additional methods, such as access to the data, should be included in the specification of the method to be developed. While this is standard software engineering routine, it is sometimes neglected in practice.

An important part of the description is the definition of the problem to be solved and underlying assumptions for a solution. This is not to be underestimated. A lot of domain knowledge exists about the appearance of structures of interest in medical images. Nevertheless, anatomical variation, artefacts from image generation, and changes in appearance due to different imaging techniques make it unlikely that conditions to extract information from the image for some specific analysis task can be proven to be sufficient.

Arriving at a problem description is a result of a discussion between a data expert (the radiologist, surgeon, or physician) and the methodology expert (the computer scientist or engineer). An experienced radiologist will certainly have the knowledge to come to a well-founded decision based on images presented to him or her. However, a radiologist is seldom forced to formalize this decision in some general way a priori, i.e., before looking at some evidence from images. Hence, generating domain knowledge for a computer-assisted method will be more difficult than making decisions based on a set of images. Furthermore, the different scientific background of the data expert and the methodology expert will make it difficult for either side to decide on validity, representativeness, and exhaustiveness of any fact conveyed from one side to the other. Experience helps, of course, but there is still room for misunderstandings.

Reviewing underlying assumptions for an analysis method later helps to identify sources of error. Such description should contain the following information

- Description of the images on which the analysis is performed, i.e., kind of images, technical parameters.
- Description of the patient group on which the analysis is performed.
- All image features that are used for the analysis method, including any assumptions about reliability of the features.
- All a priori information that is used as domain knowledge to perform the analysis.

Any change in method to correct errors found on test data will have to result in changes of this description as well.

The description also helps to set up an evaluation scenario. Evaluation ensures that information generated by the analysis method really reflects an underlying truth. Since the truth is usually not known, evaluation has to abstract from specific cases and has to provide some kind of "generic truth." This can be of two different types. Either the method is evaluated with respect to some other method of which performance and quality has been proven, or the method is evaluated with respect to the assumptions made a priori. The former is the simpler way to test a method, as it shifts the responsibility of defining truth to the method against which our new method is tested. For the latter, two aspects have to be tested. The first is whether

the analysis result is as expected when assumptions from domain knowledge hold. The second is whether the assumptions hold. This evaluation has the advantage of not relying on some gold standard, which may be difficult to come by. However, it is usually difficult to show that the domain knowledge sufficiently describes the problem. Hence, the topic of validation requires a careful look and it will be discussed in detail in the final chapter of this book.

1.2 Using Tools

Fortunately, developing an analysis method does not mean that everything has to be created from scratch. Different types of software greatly support speedy development:

- viewer software
- analysis software
- rapid prototyping software
- software libraries

An extensive list of free software (either open source software or at least free to use for academic and/or educational purposes) to view, export, analyze, and transfer images is found at http://www.idoimaging.com.

Viewer software has been primarily developed to look at the data, but usually contains some methods for analyzing the data as well. Although it is not thought to be extended, it facilitates the development of a new analysis method by providing quick access to the image data (see Fig. 1.5 for an example of a free DICOM viewer). Commercial software can be quite comfortable, and if it is found that a combination of existing methods solves the problem, development stops here. But even if it is just a way of accessing and looking at the image data, it helps to get a first impression on the kind of data, to organize a small data base on which the development of an analysis method is based, and to discuss the problem and possible solutions with the data expert.

Analysis software is different from viewer software in that it is intended to provide the user with a set of parameterizable analysis modules that perform different steps of image analysis ranging from image enhancement methods to segmentation, registration, or classification tools. An example for such analysis software is MevisLab, which exists in a commercial and a non-commercial version (www.mevis.de, see Fig. 1.6).

MevisLab provides the user with a set of implemented, parameterizable modules that produce output from data. Different modules can be combined using a graphic interface that allows the user to connect output of one module with the input of another module (see Fig. 1.7). It is for instance possible to create a processing chain, where data is first enhanced by removing artefacts and noise using suitable filters, and then separated into different regions by a segmentation technique of

Fig. 1.5 The free MicroDicom viewer (www.microdicom.com) allows viewing DICOM images and simple operations such as annotations and some filtering operations

which one segment is then selected and analyzed (e.g., by measuring its volume). This kind of modularization provides much more flexibility than the usual workstation analysis software. It does, of course, require some knowledge about the implemented modules in order to use them in an appropriate fashion.

An open source software for segmentation, registration, and other analysis methods for medical images is *3D Slicer*, www.slicer.org (Fedorov et al. 2012). It has been developed at Harvard Medical School together with several other collaborators to provide the community with a common platform for often used methods in medical imaging. 3D Slicer is an open platform with many contributors adding modules. Similar to MevisLab, it provides the user with user interfaces for a variety of modules that can be carried out stand-alone or combined to form a processing pipeline. Different to MevisLab, modules are rather high-level (e.g., a registration module), and it thus does not provide a tool for combining low level submodules to arrive at a new high-level module..

One step further is to use a rapid prototyping programming language such as MATLAB or IDL. These are interpreter languages that are geared toward rapidly processing arrays. It makes them particularly suitable to be used for working with 2d and 3d image arrays.

It is possible to write programs in both languages that can be executed later (see Figs. 1.8 and 1.9 for a view at the program development interfaces of the two prototyping languages). A wealth of methods for signal and image processing makes it easy to program even more complex methods. The possibility to use the methods in interpreter mode allows experimenting with different methods for finding a solution for some image analysis problem. For efficient use, the user

Fig. 1.6 Example of the user interface for an analysis module implemented under MevisLab (with kind permission of MeVis Medical Solutions AG, Bremen, Germany)

should be familiar with the basic vocabulary of image processing and image analysis.

Since routines for input of images (filters for the most common formats including DICOM) and for display are provided by IDL and MATLAB, they are also excellent tools to discuss potential methods and their contribution to analysis. Without having to go into detail the developer can simply show what is happing.

There is the danger that the simple way to construct analysis modules results in ad hoc developments. They may be difficult to justify later except for the fact that they worked on the data provided. Still, if prototyping languages are so useful, why should they not be used for software development? If software development consists of combining modules for which MATLAB or IDL have been optimized, there is little to say against such a decision except for the fact that both environments are commercial products and require licenses for the development and run-time environment.

A free alternative to MATLAB is *Octave* which is distributed under GNU General Public License, see www.gnu.org/software/octave. Octave has almost the same syntax than MATLAB and many of the MATLAB functions exist under

Fig. 1.7 Interface for generating analysis modules in MevisLab. The programmer can combine existing modules, e.g., for filtering, segmentation, and visualization, in a way that is suitable to solve some specific problem using an intuitive graphical user interface (with kind permission of MeVis Medical Solutions AG, Bremen, Germany)

Octave as well (albeit in different toolboxes, see (Nakamura 2016) for an introduction). Hence, most software that is developed under Octave should be interpretable under MATLAB as well. Octave exists for quite some time but development of the graphical user interface had ceased until 2014 so that newer versions were usable only in command line mode. Since 2014, Octave is again distributed with a graphical interface for program development that is similar to the MATLAB interface (see Fig. 1.10 for an example).

If substantial program development is necessary, other reasons than licensing problems may speak against using a rapid prototyping language. The environments allow object-oriented programming, but they do not enforce it (simply because this would interfere with the non-object-oriented fashion of the interpreter mode). It requires disciplined development of software that is meant to be extended and used by others. Furthermore, both environments optimize computation of matrix operations. Computation may become slow if software development requires non-matrix operations. In such case, performance suffers from translating the implemented program into the underlying environment (which in both cases is C). It usually pays to implement directly into C/C++ or any other suitable language using techniques and strategies for efficient computation.

If a method is to be implemented completely without the help of some specific programming environment, various libraries aid the developer. For analysis tasks in medical imaging, mainly two environments are of interest: OpenCV and ITK.

Fig. 1.8 MATLAB interface is similar to the programming environment for some programming language. The main differences are that MATLAB can be used in interpreter modes and that it has toolboxes for many methods in image analysis, pattern recognition, and classification

OpenCV started its life to promote Intel's chips by providing an extensive and fast image processing library for Windows and Intel chips. The restriction no longer applies. OpenCV is intended to support general image processing and computer vision tasks. Input is assumed to consist of one or several 2d images. Analysis can be almost anything in the field of computer vision including, image enhancement, segmentation, stereo vision, tracking, multi-scale image analysis, and classification. The software has been published under BSD license, which means that it may be used for commercial or academic purposes. With respect to medical image analysis its main disadvantage is that the processing of 3d or 4d scenes is not supported.

This is different for ITK (Insight Toolkit), which focuses on segmentation and registration of medical images. It is an open source project as well. Being meant to support medical image analysis, it also contains plenty of auxiliary methods for accessing and enhancing images. Furthermore, registration methods not included in OpenCV for rigid, and non-rigid registration are part of the software. Segmentation, which plays a much bigger role in medical imaging compared to the analysis of other images, is extensively covered by including state-of-the-art methods.

Fig. 1.9 IDL interface is similar to the MATLAB interface. Windows for program development, interactive evaluation of programs and functions, status of system variables, and generated output enable development of methods while inspecting results from the current chain of processes

Fig. 1.10 Octave-programmer GUI is similar to the MATLAB GUI showing the working directory, defined variables and their values, the sequence of commands and potentially windows with computation results

ITK is written in C++, and the use of classes and methods is fairly simple. Being an open source project, new contributions are invited, provided that guidelines for software development within the ITK community are followed. As with OpenCV, efficient use of ITK requires some background knowledge about methods

implemented if only for deciding whether, e.g., the implemented level set framework can be used for a segmentation solution for which you have worked out a level set formulation. Information about OpenCV and ITK can be accessed by their respective websites (http://opencv.willowgarage.com/wiki/ and http://www.itk.org/) with wiki and user groups. Being open source projects with voluntary supporters, it is usually expected that the questioner spent some time on research work before asking the community. A question such as "how can I use ITK for solving my (very specific) registration problem" might fetch a "we suggest reading some literature about registration first".

As with the prototyping languages, the open source libraries may be all that is needed to solve an analysis problem. But even if not, they will provide for quick access to methods that can be tested with respect to their capability for solving the problem. They also may provide state-of-the-art methods for data access, preprocessing, and post-processing. If the solution of image analysis problems is a recurring task, it should well justify the effort to understand and efficiently use these toolkits.

1.3 An Example: Multiple Sclerosis Lesion Segmentation in Brain MRI

So far, the discussion has been rather abstract. We conclude this chapter with a presentation of an example. We selected a rather complex segmentation task (see Fig. 1.11) for finding multiple sclerosis lesions in order to exemplify the range of models and decisions in performing the analysis. The description is based on the two publications (Al-Zubi et al. 2002; Admasu et al. 2003).

Fig. 1.11 White matter lesions can be recognized in MR images as their signal differs from surrounding tissue. However, this information alone is insufficient as it can be seen by comparing a simple intensity-based segmentation (**b**) of an MRI slice (**a**) with the result of a segmentation by a human expert (**c**)

Multiple sclerosis (MS) is a disease of the central nervous system where the myelin shield of axons is receding. Estimating the extent and distribution of MS lesions supports understanding the development of the disease and estimating the influence of treatment. MS lesions can be detected in magnetic resonance images using spin echo imaging. The method produces registered proton density (ρ), T_1 and T_2 images (see next chapter for details on this imaging technique), all of which are going to be used in the segmentation process.

The choice of an appropriate analysis technique was guided by the goal of replacing manual delineation of MS lesions by some automatic procedure. Lesion detection is an active research field, and presentation of the method below by no means should indicate that this is the only or even the best way to do the segmentation. It was chosen because the author of the book contributed to the result and can comment on the various decisions and point out problems that are usually not stated in a publication.

Selecting, adapting, or developing an analysis algorithm follows standard practice of scientific work. Information about data and possible a priori information is gathered, a representation is selected in which this information can be described and which can be used during segmentation. Then, the method is implemented and tested.

Since the goal is to separate MS lesions from the background, it points to segmentation of foreground segments. Their characteristics describe properties of MS lesions. Properties of other structures need to be captured only to the extent of enabling separation of lesions from the background.

Discussions with physicians, MR physicists, and the study of related literature resulted in the following description of the appearance of MS lesions in T_2-weighted spin echo images:

1. 90–95% of all lesions occur in white matter. White matter segmentation can be used for imposing a location constraint on the lesion search.
2. The anterior angle of the lateral ventricles, the corpus callosum, and the periventricular areas are more often affected by MS than other regions of the nervous system. Identifying those regions will further restrict localization.
3. There is considerable size variation for lesions but their shape is roughly ellipsoidal. It points to the use of shape constraints as part of the a priori knowledge.
4. MS lesions tend to appear in groups. This will require introduction of a priori knowledge about neighboring segments.
5. The intensity of an MS lesion varies from being bright in its center to being lower at its boundaries. The intensity range is brighter than white matter intensity but overlaps with intensity for the cerebrospinal fluid (CSF). The data-driven criterion will thus be some homogeneity constraint on intensity, but it is clear that data-driven segmentation alone will be insufficient.
6. Shading in the MR images affects brightness of white matter as well as of MS lesions. Thus, the above-mentioned homogeneity criterion will be more efficient in separating lesions if shading is considered.

It was possible to accumulate such an extensive list of attributes because of the high interest of the research community in the topic. There will be other cases, where the medical partner who requested an analysis solution may be the only source of information. It will still be possible to generate a good description as the partner is an expert and trained to develop such assumptions. He or she will also be able to point to medical literature where such information can be found (even if it does not have been used for generating computer-assisted solutions).

The way of describing the attributes typically involves expressions such as "tends to" or "intensity varies" indicating a rather fuzzy knowledge about permissible ranges for attribute values. Even mentioning numbers such as "90–95% of all lesions" may refer to a single study with other studies coming up with different numbers.

Research in medical science is often conducted empirically with observations leading to hypotheses. This is no problem in general. Much of scientific research—especially if involving complex dependencies that do not easily point to simple underlying facts—is empirical. However, for making this information useful as a priori knowledge, it has to be treated as factual. The transition from empirical evidence to known facts may be a source of failure of an analysis method because underlying model assumptions turn out to be false.

Considering possible model knowledge, a number of consequences arose regarding a successful segmentation strategy for the problem:

- Segmentation will need to include a strong model component to account for the ambiguities between the appearance of CSF and MS lesions.
- Model constraints regarding the shape and the location of lesions will need to be stated probabilistically or in some fuzzy manner (at least in some informal fashion).
- The first location constraint (condition 1 above) requires white matter segmentation. Application of the second constraint (condition 2) further requires some representation of relative positions.
- Constraint 4 requires definition of neighborhood among segments.

The fuzziness of described lesion properties led us to use a Bayesian Markov random field (MRF) for representation (MRFs will be described in Sect. 14.1). An MRF allows a probabilistic description of known neighborhood dependencies which is just what we need to describe constraints 1, 2, and 4.

Segmentation takes place in three stages (see Fig. 1.12). At the first stage, white matter and lesions are segmented based on their intensities. A classifier for separating white matter is trained which determines expected values and variances of a multivariate Gaussian distribution for ρ, T_1, T_2. Intensity variation due to magnetic field inhomogeneities is estimated based on segmented white matter regions (which are assumed to have constant intensity). Segmentation is repeated (see Fig. 1.13a). Subsequent MRF-based restoration reduces noise in the result, since the a priori knowledge about the neighborhood states that adjacent pixels belong more likely to the same kind of object rather than to different objects. Finding the most probable

Fig. 1.12 Overall structure of the algorithm with the three different analysis phases

segmentation given, the MRF formulation resulted in a first estimate of potential MS lesions (see Fig. 1.13b).

The white matter segmentation of the first stage is used to find neighboring structures by atlas matching (see Fig. 1.13c). Segmentation of gray matter is difficult but necessary. We needed to differentiate between white matter bordering gray matter or the CSF of the ventricles in order to apply the second localization condition. Spatial continuity of gray matter and CSF regions is exploited when assuming that elastic registration of patient images with an anatomical atlas gives sufficient approximation of the different structures. Lesions segmented in the previous step are reassessed using the deformed atlas as a priori information.

The shape constraint of lesions is used as final confirmation. Because of the fuzzy a priori knowledge, thresholds in the two previous stages for segment membership were set in a way as to not exclude potential lesion sites. The final step treats labels of lesion segments as random variables being either a lesion or gray matter. Criteria are the deviation from the ellipsoidal shape and the distance of a segment from the ventricles. An (abstract) neighborhood system is defined. Segments in this neighborhood are assumed to have the same label, as lesions tend to appear in groups (condition 4). Labelling optimizes this MRF (see Fig. 1.14).

Fig. 1.13 In the first two stages, the initial segmentation (**a**) is first post-processed using a local homogeneity constraint arriving at (**b**). Then, atlas matching is used to remove lesions at unlikely locations arriving at (**c**)

Fig. 1.14 In the final step, the result from atlas matching in (**a**) is post-processed by defining a shape-based MRF. The result in (**b**) compares favorably with the expert segmentation in (**c**)

The stages described above accumulate constraints derived from attributes that characterize lesions. Selecting a sequential order was necessitated by the nature of some of the attributes (e.g., groups of lesions can only be identified when potential lesions are already segmented, neighborhood of white matter to other tissues requires white matter segmentation first.). Using a common representation for neighborhood relations at the pixel and the segment level was intentional, as it enabled an easy reconsideration of segment labelling in subsequent steps.

The resulting structure of the analysis methods accumulates and evaluates semantics gradually in several steps that introduce different kinds of domain knowledge. This is typical for many automatic analysis methods, as it allows keeping the fuzzyness of knowledge through many steps of the decision process. If it enables correcting erroneous decisions, it also contributes to the robustness of the method. While this is generally desired, it may hide the use of incomplete, incorrect, or inadequate domain knowledge. Evaluation should account for this by validating expected results of intermediate steps as well.

The modular structure of the method may also be a problem from the viewpoint of a developer. A large part of the solution lies in the proper combination of the modules. Reusing modules for a different problem solution requires expertise of a developer, since the purpose of single modules within the overall problem solution is rather abstract. This may be unwanted in a clinical environment where clinical and radiological expertise are available but not the expertise of an engineer for image analysis solutions.

Using the methodology described above is by no means the only way to encode the prior knowledge about lesion attributes. This is proved by the numerous publications on MS lesion detection using different knowledge representations and segmentation strategies. Selecting a specific method is often influenced by reasons, such as existing experience with specific types of representation or segmentation techniques, existing software, or other preferences by the developer. If justification for some technique is conclusive nonetheless, it may not influence the effectiveness of a method. Effectiveness may only be affected when existing prior knowledge was used incompletely or inaccurately despite the conclusiveness of the argument.

Reviewing our own decisions above, several points came to mind:

- The sequential order of removing lesions does not enable reconsideration of segments excluded from being lesions at later stages.
- Assuming Gaussianity for the probability distributions in the framework is an approximation which may not always be true.
- Repeated MRF optimization in the first two stages may be redundant because it uses rather similar kinds of a priori knowledge.
- Using the Bayesian framework was influenced by the type of expertise in our group at that time.

In a later revision (Admasu et al. 2003), the sequential order of removal of lesions was replaced by a sequence of two steps, which explicitly addressed two different aspects of the prior knowledge. At a data-driven first step, segments were created by adaptive intensity region growing using the concept of fuzzy connectedness of (Udupa and Samarasekera 1996). It created CSF, gray matter (GM), and white matter (WM) regions. The uncertainty of the homogeneity criterion in the data-driven step was not modeled by Gaussian distributions but by introducing interaction to indicate seed points for growing fuzzily connected GM, WM, and CSF regions.

Regions that are enclosed by GM, WM, or CSF and that are not found by the method are labeled as potential lesion sites. The method does involve interaction, but it is robust to input errors and not very costly. Only few seed points have to be selected for each of the connected components of GM, WM, or CSF. If any of those components are missed, they will be erroneously classified as potential lesion site but this will be remedied in the next step. The advantage of using interaction is to avoid specification of absolute homogeneity criteria.

Features of potential lesion sites were then fed to a trained backpropagation network. Shape and location features described a potential lesion site. The beauty of using a neural network was that it essentially creates an approximation of a decision boundary in feature space. The decision boundary describes locations in feature space where the probability of a segment being a lesion equals that of a segment not being a lesion. Other properties of the two probability distributions are not needed and are not approximated. This results in a good estimate of the decision boundary even if only relatively few samples are present. Although the relative spatial relation among lesions was not considered—an important step of the previous approach—success was in the range of the previous method, and computation speed was much faster.

What can be learnt from the simplification? Sometimes it pays to consider limited interaction for supplying a priori knowledge. Reconsidering the limitations of the use of model knowledge, even if the performance is very satisfactory, may lead to a more efficient computation without sacrificing quality. After all, developing a solution for a complex problem and experimenting with the results usually

produces a lot of insight into the importance of applied domain knowledge and its representation. It should be noted that redesigning a method may again slant the view of the developer toward methods supporting a chosen strategy.

1.4 Concluding Remarks

Analysis in medical imaging differs from other image analysis tasks. The various medical images carry very different information, and many of the analysis tasks are related to individual quantification of entities (volume, extent, delineation, number of occurrences) in the human body. Normal anatomy that is subject to the quantification varies enormously. This is even more the case for pathological deviations. Hence, a major part of analysis is to acquire, represent, and integrate this knowledge in the image interpretation method.

It is not always necessary to develop an analysis method from scratch. Existing commercial and free software packages provide methods that can be adapted to the purpose of carrying out some specific analysis. They range from parameterizable analysis methods offered through a graphical user interface to class libraries that can be used or extended to serve a developer's purpose for a new method. The question of how to employ which domain knowledge to solve a specific analysis task still resides with the developer when deciding on the use and adaptation of such a method.

1.5 Exercises

- What analysis task would be typical for analyzing a photography that would be untypical for a medical image?
- What is an example for the kind of information that is measured for a medical image?
- Name at least two analysis tasks that could arise when using images for diagnosis support.
- What is the main problem when validating an analysis method?
- Name a difference between analysis tasks for a clinical study compared to image analysis in computer-assisted surgery.
- What are the differences between a clinical study and an epidemiological cohort study? Why is it important that image analysis in cohort studies should be automatic?
- Where in a hospital does image analysis take place? Who are the involved persons?
- Name a rapid-prototyping language that could be used for developing an analysis method.
- Name a rapid-prototyping language distributed under GNU license.

- How can workstation software be used for developing an analysis solution if the workstation software itself does not provide the solution?
- Why is it so important to provide detailed documentation for all information and assumptions used for developing an analysis algorithm?

References

Admasu F, Al-Zubi S, Toennies KD, Bodammer N, Hinrichs H (2003) Segmentation of multiple sclerosis lesions from MR brain images using the principles of fuzzy-connectedness and artificial neuron networks. In: Proceedings of international conference on image processing (ICIP 2003), pp 1081–1084

Al-Zubi S, Toennies KD, Bodammer N, Hinrichs H (2002) Fusing markov random fields with anatomical knowledge and shape based analysis to segment multiple sclerosis white matter lesions in magnetic resonance images of the brain.In: Proceeding of SPIE (Medical Imaging 2002), vol 4684, pp 206–215

Bankman IN (2008) Handbook of medical image processing and analysis, 2nd edn. Academic Press

Dhawan AP (2011) Medical image analysis. IEEE Press Series on Biomedical Engineering, 2nd edn. Wiley

Fedorov A, Beichel R, Kalpathy-Cramer J, Finet J, Fillion-Robin J-C, Pujol S, Bauer C, Jennings D, Fennessy F, Sonka M, Buatti J, Aylward SR, Miller JV, Pieper S, Kikinis R (2012) 3d slicer as an image computing platform for the quantitative imaging network. Magn Reson Imaging 30(9):1323–1341

Gillies RJ, Kinahan PE, Hricak H (2015) Radiomics: images are more than pictures, they are data. Radiology 278(2):563–577

Kumar V, Gu Y, Basu S, Berglund A, Eschrich SA, Schabath MB, Forster K, Aerts HJWL, Dekker A, Fenstermacher D, Goldgof DB, Hall LO, Lambin P, Balagurunathan Y, Gatenby RA, Gillies RJ (2012) Radiomics: the process and the challenges. Magn Reson Imaging 30(9):1234–1248

Nakamura S (2016) GNU OCTAVE primer for beginners: EZ guide to the commands and graphics, 2nd edn. CreateSpace Independent Publishing Platform

Paragios N, Duncan J, Ayache N (2015) Handbook of biomedical imaging: methodologies and clinical research. Springer

Preim B, Klemm P, Hauser H, Hegenscheid K, Oeltze S, Toennies KD, Völzke H (2016) Visual analytics of image-centric cohort studies in epidemiology. In: Visualization in medicine and life sciences III. Springer, pp 221–248

Reyes-Aldasoro, CC (2015) Biomedical image analysis recipes in MATLAB: for life scientists and engineers. Wiley

Toennies KD, Gloger O, Rak M, Winkler C, Klemm P, Preim B, Völzke H (2015) Image analysis in epidemiological applications. Inf Technol 57(1):22–29

Udupa JK, Samarasekera S (1996) Fuzzy connectedness and object definition: theory, algorithms, and applications in image segmentation. Graph Models Image Process 58(3):246–261

Digital Image Acquisition

2

Abstract

Medical images are pictures of distributions of physical attributes captured by an image acquisition system. Most of today's images are digital. They may be post-processed for analysis by a computer-assisted method. Medical images come in one of two varieties: Projection images project a physical parameter in the human body on a 2d image, while slice images produce a one-to-one mapping of the measured value. Medical images may show anatomy including pathological variation in anatomy if the measured value is related to it, or physiology when the distribution of substances is traced. X-ray imaging, CT, MRI, nuclear imaging, ultrasound imaging, optical coherence tomography, photography, light and electron microscopy, EEG, and MEG will be discussed in this chapter. The discussion focuses on the relationship between imaged physical entity and information shown in the image as well as on reconstruction methods and resulting artifacts.

Concepts, Notions, and Definitions Introduced in this Chapter

- *Imaging techniquesa:* X-ray, fluoroscopy and angiography, digital subtraction angiography, X-ray CT, CT angiography, MR imaging, MR angiography, functional MRI, perfusion MRI, diffusion MRI, scintigraphy, SPECT, PET, ultrasound imaging, optical coherence tomography, photography, optical and electron microscopy, EEG, and MEG.
- *Reconstruction techniques*: filtered backprojection, algebraic reconstruction, EM algorithms
- *Image artifacts*: noise, motion artifacts, partial volume effect, MR-specific artifacts, ultrasound-specific artifacts.

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_2

A major difference between most digital medical images and pictures acquired from photography is that depicted physical parameters in medical images are usually inaccessible for inspection (see Fig. 2.1). Features or quantities determined by computer-assisted analysis cannot easily be compared with true features or quantities. It would be, e.g., infeasible to open the human body in order to verify whether a tumor volume measured in a sequence of CT images in some post-treatment confirmation scan corresponds to the true volume.

Fortunately, the physical property depicted, its diagnostic value, and possible artifacts are usually well-known. Furthermore, the imaging technique has been chosen on purpose because it is known to produce images that depict diagnostically relevant information. The development of efficient analysis techniques often uses this knowledge as part of the domain knowledge in order to make up for the inaccessibility of the measured property.

A physical property measured by an imaging device and presented as a picture must meet three conditions in order to be useful. It has to penetrate the human body, it must not unduly interfere with it, and it must be meaningful for answering some medically relevant questions.

With respect to digital imaging, four major[1] and several minor imaging techniques meet these requirements. The major techniques are as follows:

- *X-ray imaging* measures absorption of short-wave electromagnetic waves, which is known to vary between different tissues.
- *Magnetic resonance imaging* measures density and molecular binding of selected atoms (most notably hydrogen which is abundant in the human body), which varies with tissue type, molecular composition, and functional status.
- *Ultrasound imaging* captures reflections at boundaries between and within tissues with different acoustic impedance.
- *Nuclear imaging* measures the distribution of radioactive tracer material administered to the subject through the blood flow. It measures function in the human body.

Other imaging techniques include *EEG* and *MEG imaging*, *optical* and *electron microscopy*, *optical coherence tomography*, and *photography*. All techniques have in common that an approximate mapping is known between the diagnostic question that was the reason for making the image and the measurement value that is depicted. This can be very helpful when selecting an analysis technique. If, for instance, bones need to be detected in an X-ray CT slice, a good first guess would be to select a thresholding technique with a high threshold because it is known that X-ray attenuation in bone is higher than in soft tissues and fluids.

[1]Imaging techniques are ordered by importance with respect to digital imaging and digital image analysis. Orders of importance with respect to relevance to diagnosis or with respect to frequency of examination are different, of course.

Fig. 2.1 Information from photography is quite different from that of a medical image (in this case a bone scintigraphy, published under Creative Commons license). While the human depicted in the photograph looks familiar, interpretation of the image on the *right* requires expertise with respect to the meaning of the intensities. On the other hand, specific domain knowledge exists as to how to interpret image intensity in the scintigraphy and the image is acquired in a way that makes analysis as easy as possible, all of which cannot be said about the picture on the *left*. Obviously, the kind of task for computer-based image analysis is different for these two pictures

Many of the imaging techniques come in two varieties: *Projection images* show a projection of the 3d human body onto a 2d plane and *slice images* show a distribution of the measurement value in a 2d slice through the human body. Slice images may be stacked to form a volume. Digitized images consist of a finite number of image elements. Elements of a 2d picture are called *pixels* (picture elements), and elements of stacked 2d slices are called *voxels* (volume elements). We will call pixels or voxels *scene elements* if the dimension of the scene is not known or not important.

2d and 3d images may have an additional time dimension if variation along the time axis provides additional diagnostic information (e.g., if normally and abnormally beating hearts are compared). Slice images are usually reconstructed from some kind of projection. Reconstruction may cause additional artifacts.

The chapter provides an overview on image acquisition systems for digital images in medicine. Emphasis is put on semantics of the image and on system-specific artifacts. A more detailed look at the imaging techniques and the

reconstruction of medical images is provided by specialized literature such as Prince and Links (2005) or the text of Bushberg et al. (2002) (the latter is more directed toward radiology residents). Imaging techniques in this chapter are listed by the kind of entity that is measured.

2.1 X-ray Imaging

X-rays have been discovered in 1895 by Wilhelm Röntgen.[2] He noticed an unknown kind of rays emitted by a cathode ray tube (CRT) which easily penetrated paper, aluminum, and many other materials but not a lead plate. He also found that this kind of rays is not reflected, refracted, diffracted, or deflected by electrical fields. He discovered that the rays blackened film so that photographic images can be produced. Röntgen called this unknown type of radiation *X-rays*.

A material-specific amount of the energy of an X-ray is attenuated when penetrating a material. For the first time in history, a technique allowed noninvasive insight into the human body (see Fig. 2.2).

The harmful aspects of X-rays were not known in the early years. It was employed for all kinds of purposes without balancing potential information gain through imaging against harmfulness of exposure to radiation. For instance, until the 1960s, some shoe stores offered a service by which customer's feet wearing shoes to be fitted were X-rayed. Few if any precautions were taken to secure operators of X-ray machines from harmful exposure.

2.1.1 Generation, Attenuation, and Detection of X-rays

X-rays are electromagnetic waves with a wavelength above the visible spectrum. Electromagnetic radiation has the characteristics of waves but is actually traveling as clusters of energy called *photons* with a given wavelength. Electromagnetic waves do not need a carrier such as sound waves and travel at the speed of light, c. Their wavelength λ and frequency f are related by

$$c = \lambda f. \tag{2.1}$$

The energy of a photon measured in electron volts (eV) is the energy that a single electron acquires when moving through a potential of 1 V. The energy of a photon is characterized by its wavelength. It is given by

$$e = 1.24/\lambda \tag{2.2}$$

[2]Carl Wilhelm Röntgen was professor for physics at the Julius-Maximilian-Universität Würzburg, when he discovered X-rays in his experiments with the cathode ray tube in 1895. He received the Nobel Prize in Physics for his discovery.

Fig. 2.2 X-rays penetrate the human body and produce an image that shows the integral of tissue-specific absorption along a path from the X-ray source to a detector

where the unit of measurement is kilo electron volt (keV) and the wavelength is measured in nanometers (nm).

The wavelength of an electromagnetic wave corresponds to its energy. Examples for electromagnetic waves, in the order of increasing energy, are radio waves (being used for magnetic resonance imaging), visible light, and X-rays or gamma rays (see Fig. 2.3). The difference between X-rays and gamma rays is not their wavelength but the fact that gamma rays are created in the nucleus of an atom while X-rays are not. The energy of X-ray photons is sufficient to release electrons from an atom, a process which is called *ionizing radiation*.

X-rays are characterized by their *exposure*, i.e., the amount of charge per volume of air, which is measured in Roentgen (R). Exposure measures the energy of the radiation source, but it does not describe how much of the radiation is absorbed by a body under radiation. Absorption per unit mass is called *dose,* and it is measured in *r*adiation *a*bsorbed *d*ose (rad) or gray (Gy) with 1 Gy = 100 rad. The ratio between exposure and dose varies with the X-ray energy and is often called the *f*-factor. The *f*-factor at low exposure for hard tissues such as bone is much higher than for soft tissues and water. Hence, bone at low doses absorbs a significantly higher amount of radiation than soft tissues.

Fig. 2.3 Spectrum of diagnostic X-ray is above the spectrum of visible light in the range of 0.01–15 nm (10^{-11} to 1.5×10^{-8} m)

2.1.1.1 X-ray Generation

Understanding the different types of X-rays requires some understanding about their generation. Electrons in an atom are organized in shells around the nucleus. Since the negatively charged electrons are attracted to the protons in the nucleus, the innermost shell contains electrons with the lowest energy. Energy is needed for moving an electron from an inner shell to an outer shell, which is equivalent to the difference between energy levels of the two shells. If an electron shall be released from a shell, the required energy amounts to the difference between its current energy level and the level of the outermost shell plus the energy to remove an electron from the outermost shell. Electrons on the outermost shell are thus easiest removed and are called *valence electrons*.

X-rays are generated as excess energy from electrons in the material of a *cathode ray tube* (CRT)[3] when heating the cathode. Energy from heating causes electrons to be released from the cathode and accelerated toward the anode. In the anode, electrons lose their kinetic energy by excitation, ionization, and radiation. *Excitation* and *ionization* cause electrons of the anode material to move from an outer shell to an inner shell (see Fig. 2.4). For excitation, this happens directly, whereas ionization causes electrons of an outer shell to be released, which then excites electrons of another atom. The excess energy being released as X-rays depends on the energy difference between the outer and the inner shell. Hence, radiation from this process is monochrome. This kind of X-rays is called *characteristic* or *monochrome* X-ray radiation.[4]

Most of the X-ray radiation, however, is *polychrome*. An incident electron is slowed down by passing the nucleus of an atom. Slowing down means that the frequency of the electron changes. The excess energy is emitted as photon. Its amount depends on how close the incident electron passes the nucleus. All its energy is released as X-rays if it annihilates in the nucleus. If it passes the nucleus, more energy is released for an inner shell passage than for an outer shell passage. This type of radiation is called *bremsstrahlung*, and it is inherently polychrome (see Fig. 2.5).

The likelihood of passing an atom at a given distance increases with distance to the nucleus because the surface of a sphere of a given radius increases with the radius. Thus, the spectrum of the bremsstrahlung of a cathode ray tube should roughly be of triangular shape. However, the surrounding glass of the tube filters some of the low energy radiation. Hence, the actual curve is zero at low energy levels, then increases to some maximum at intermediate energy levels, and finally decreases to zero at the maximum energy (which is the maximum energy of electrons incident on the anode of the tube). The complete spectrum of a typical

[3]The material for anode and cathode is usually tungsten (also called Wolfram), a chemical element belonging to the metals with the highest melting point of all metals. In CRTs for mammography, another metal, molybdenum, is used.

[4]The detection of characteristic X-ray radiation by Charles G. Barkla in 1905 resulted in another Nobel Prize in Physics presented to its discoverer in 1917.

Fig. 2.4 X-ray generation by excitation and ionization. In excitation, external energy pushes an electron from an outer shell to an inner shell. Excess energy is released as X-rays. The ionization process is similar, except for the fact that excitation happens indirectly by an electron that is released from an outer shell of a different atom

Fig. 2.5 Polychrome radiation is a result of a passage of an electron. The frequency of the released radiation depends on the extent to which the electron loses its energy

X-ray tube is a combination of monochrome and polychrome radiation, where some spikes indicate various characteristic radiation energy levels in the otherwise smooth curve.

X-ray tubes are characterized by the total amount of energy that is emitted as X-rays and the quality of the radiation. A high-quality tube has a higher ratio of high energy radiation and of monochrome radiation. High-quality radiation imposes a lower dose on the patient and generally produces better images.

Quantity and quality of X-ray radiation depend on a number of characteristics of the tube (such as the anode material) and on parameters to be selected by the operators (such as potential between cathode and anode, voltage, and exposure time).

The quality of emitted X-rays can be enhanced by a filtering process called *beam hardening*. The glass surrounding the tube already filters some of the low energy rays of the bremsstrahlung. It reduces the total energy of emitted X-rays but shifts the spectrum toward the high-quality range. The emitted frequency range can be further improved by additional filtering of low-frequency components. The amplitude of monochrome radiation is usually higher than that of bremsstrahlung. Beam hardening may thus be used for an overall reduction in X-ray radiation enhancing monochrome radiation in the spectrum as well.

2.1.1.2 X-ray Attenuation

When X-rays enter the human body, four types of attenuation can happen: Rayleigh and Compton scatter, photoelectric absorption, and pair production (see Fig. 2.6). Of these, Compton scatter and photoelectric absorption are the main factors for image generation from medium range energy X-ray imaging such as used in diagnostic imaging.

Rayleigh scatter mainly happens in low-energy imaging such as mammography. A photon causing Rayleigh scatter loses its energy by exciting the whole atom which immediately releases a photon with almost the same energy although usually scattered in a different direction.

A photon causing *Compton scatter* will release a valence electron from its shell. It will lose as much energy as was necessary for the release. Scattered photons will change their direction.

Scatter, be it Rayleigh or Compton scatter, increases noise and reduces contrast in an image. Noise stems from the randomness of scattering and the random change of direction for scattered photons. Contrast is reduced because scattered photons reaching the receptor do not carry position information of the locus of scattering, thus only increasing the overall brightness in the image.

Photoelectric absorption is the main contributor to imaging even though absorption in diagnostic X-ray happens less often than Compton scattering. A photon releases its energy through photoelectric absorption by removing one of the electrons of the inner shells. The photon loses its energy completely. The released electron leaves a vacancy in the shell which is filled by an electron of one

Fig. 2.6 Four kinds of X-ray attenuation exist. Rayleigh and Compton scatter causes noise in the image since the scatter direction is random. Photoelectric absorption produces the X-ray image, since the released energy is too low to be detected. The energy levels used in X-ray imaging do not produce pair production

of the outer shells. This initiates a cascade of filling in vacancies from outer shells. The energy gain from each transition is released as characteristic radiation. Characteristic radiation caused by this process has a much lower energy than that of the incident photon. Hence, it is absorbed by surrounding atoms and does not contribute to the degradation of the image. Absorption is the cause of a meaningful image but also increases the radiation dose absorbed by the tissue.

The probability of absorption increases with atomic number (e.g., the calcium in bone absorbs more than hydrogen of the cerebrospinal fluid) and decreases with beam energy. Absorption increases dramatically when the photon energy equals the binding energy of an inner electron. This value is called *absorption edge*, and its existence is exploited by *contrast agents*. Contrast agents such as iodine and barium have a high atomic number and an absorption edge in the range of diagnostic X-rays (33 and 37 keV). The high atomic number already increases the probability of absorption. Energy in the range of the absorption edge of the radiation spectrum will be further reduced.

Pair production, the last of the four effects, happens only if the energy exceeds 1022 keV, which is beyond the usual diagnostic range for X-ray imaging. In pair production, a photon annihilates and produces an electron–positron pair. Positron and electron are sent in opposite directions. Although this is not relevant for X-ray imaging, it should be noted that the opposite process, i.e., the annihilation of a positron when merging with an electron producing two photons with 511 keV, is used for positron emission tomography (PET), an imaging technique that is described later in this chapter.

2.1.2 X-ray Imaging

X-ray imaging uses the dependency of photoelectric absorption on the atomic number for producing a diagnostically meaningful image (see Fig. 2.7 for examples). Being the oldest technique, a vast number of different imaging methods evolved from the original method presented by W. Röntgen. In this section, we will only touch the subject to give an impression how the images are created and what kind of different X-ray imaging techniques exists. A more detailed treatment can be found, e.g., in Bushberg et al. (2002), and Prince and Links (2005). Much of this section has been taken from there and from the AAPN/RSNA Physics Tutorial for Residents that appeared in various issues of the Radiographics journal (McCollough 1997; Bushberg 1998; McKetty 1998; Pooley et al. 2001; Wang and Blackburn 2000).

Diagnostic equipment for X-ray imaging consists at least of a cathode ray tube emitting X-rays and a receptor with the patient placed between emitter and receptor. The receptor may be a film, an image intensifier, or a flat panel detector with the latter two producing digital images.

If the X-ray tube is assumed to be a point source for X-rays and the receptor is planar, the image intensity at every location of the receptor will be proportional to

Fig. 2.7 Two radiographs (with kind permission from Siemens Sector Healthcare, Erlangen, Germany). Bone structures in the two images are clearly visible. Differentiating between different soft tissues is more difficult as is the depth order of the projected structures

the attenuation along a ray from the X-ray tube to the receptor. The measured intensity for a monochromatic beam at a location (x, y) on the receptor is then

$$I_{\text{out}} = I_{\text{in}} \cdot \exp\left(-\int_{s_0}^{s_1} \mu(s)\mathrm{d}s\right) \tag{2.3}$$

where I_{in} is the incident intensity of X-rays when entering the body, s is a ray from the X-ray source to (x, y) on the image plane, s_1 is the point where the ray enters the body, and s_2 is the point where it exits the body. The function $\mu(s)$ is the *attenuation*. Attenuation, as pointed out in the previous section, is mainly caused by Compton scatter and photoelectric absorption and to a small extent by Rayleigh scatter. The intensity at some location (x, y) is given by X-ray attenuation plus intensity due to scattered photons. If scattered photons are assumed to be distributed evenly over the image, they increase the brightness in the image by a noise component thus reducing contrast.

The imaging process described above is idealized in that it assumes that the X-ray source is a point source. In reality, the *focal spot* of an X-ray source covers a finite area leading to a loss of resolution due to penumbrae. Its extent depends on the distances between source, object, and receptor, as well as on the diameter of the focal spot (see Fig. 2.8). Regular X-ray CRTs have a focal spot with a diameter of 1 mm, fine focus CRTs have one with 0.5 mm diameter, and microfocus CRT has a 0.2-mm-diameter focal spot.

Fig. 2.8 Different sizes of the focal spot cause different blurring even if the aperture is the same

Integrated attenuation along *s* of polychromatic X-rays is different to the monochromatic case described above, since low energy radiation is absorbed earlier than high energy radiation. Placing the patient between source and receptor causes additional beam hardening. Continuous beam hardening of an X-ray traveling through the body has the effect that attenuation changes depending on the distance traveled in the body. This makes image content dependent on patient positioning. It also causes some of the low energy radiation being absorbed by the patient without having an impact on the images and unnecessarily increases the dose. Prior beam hardening by filtering lessens these unwanted effects.

Of the three types of receptors, *analogue film* is the oldest and still the most widespread. Film may be digitized, but receptors such as image intensifiers and flat panel detectors are preferred if computer-assisted post-processing is desired.

An *image intensifier* produces a visible image from X-rays in a similar fashion than a conventional CRT. X-rays are turned into visible light on a phosphor screen in a vacuum tube, which is then converted into electrons by a photocathode. The electrons are focused and accelerated—this is the enhancement step—toward the anode and cast onto the output phosphor producing the intensified image. Enhancement through an image intensifier increases the brightness at the output phosphor more than 1000 times with respect to the very weak signal at the input phosphor.

The image intensifier was originally invented for creating a visible signal without the need to use and develop film. It has the additional advantage of enabling transmittance of the electronic signal and its digitization through an A/D converter.

Images from an image intensifier suffer from a number of artifacts of which the following three are relevant for post-processing (see Fig. 2.9):

- *Vignetting* is caused by the angle at which rays fall onto the input screen. The angle is perpendicular to the image screen in the center. In this case, the incident X-ray energy is distributed over the smallest possible area. The intensity, i.e., the energy per unit area, is maximal. The angle decreases with distance to the center causing the incident X-ray energy to be distributed over a larger area. Hence, intensity decreases with distance to the center.

Fig. 2.9 Vignetting causes shading in an image as it can be seen in the picture of homogeneous material on the *left*. The location-dependent magnification by a pincushion distortion of an image intensifier is seen on the test pattern of equal-sized squares on the right (*dashed lines*). This image also shows the much less prominent deformation due to the earth magnetic field

- *Pincushion distortion* is caused by the curvedness of the input screen and results in magnification. Magnification increases with deviation of the input surface from a tangent plane to the center of the screen.
- The *S-distortion* is caused by external electromagnetic fields that influence the course of the electron beam between input and output phosphor.

A completely digital receptor for X-rays is the TFT *flat panel detector* which has been developed in recent years. A flat panel detector combines a flat panel scintillator, which turns X-rays into visible light with a detector that transforms the light into an analog electric signal which may then be digitized. Several studies established the adequateness of using a flat panel detector in diagnostic imaging with the added advantage of lower doses necessary for imaging [e.g., Bacher et al. (2003), Fink et al. (2002), Garmer et al. (2000)].

The *blackening curve*, which measures the dependency of incident radiation on the receptor and the intensity of the resulting images, has a larger interval of linear increase for flat panel detectors and image intensifiers than for film. This decreases the likelihood of an accidental overexposure or underexposure. The spatial resolution in digital radiography is still lower than that of film. However, recent advances in TFT detector design have pushed the limits of achievable resolution into the range of film (about 4000 × 4000 pixels).

An advantage of digital radiography using either a TFT detector or an image intensifier is that the operator may choose the spatial resolution. If imaging at a lower resolution is sufficient for some diagnostic purpose, this can be achieved by analog or digital integration over several pixels of the image. Since integration reduces noise, the signal-to-noise ratio increases without having to increase exposure and, consequently, dose.

2.1.3 Fluoroscopy and Angiography

Fluoroscopy is a specific kind of X-ray imaging to visualize moving or changing objects in the human body (see Fig. 2.10a). Examples for using diagnostic fluoroscopy are as follows:

- to follow the heartbeat for detecting abnormal behavior;
- to follow the course of a contrast agent through the colon in order to detect potential abnormalities such as a tumor; and
- to image cardiac or cerebral blood flow by using a contrast agent.

The technique received its name because X-rays are turned into visible light using a fluorescent screen. Early fluoroscopy systems placed the screen directly behind X-ray tube and patient. The physician was sitting in front of the screen. Fluorescence from the X-rays is very weak so that the introduction of the image intensifier made fluoroscopic imaging a much more viable tool.

Most of today's fluoroscopic imaging devices produce digital images and enable the creation of X-ray films as well. Fluoroscopic imaging devices are not necessarily static. When mounted on a C-arm, they can be rotated around the patient for producing projections along arbitrary directions. Fluoroscopic imaging is used for diagnosis and supports surgical interventions. It is an attractive imaging technique for the latter because images can be produced during the intervention.

Fluoroscopic imaging of the vascular system using a contrast agent is called *angiography* (see Fig. 2.10b). The contrast agent is applied by a catheter being guided to the location to be imaged (e.g., the brain or the heart wall). Imaging cerebral or cardiovascular flow supports diagnosis of arteriosclerosis, arteriovenous malformations (AVMs), etc. Angiographic images show anatomy with blood vessels enhanced through the contrast agent. Angiographic images can be acquired in real time and can be used to guide a surgical intervention (see Fig. 2.11 for a

Fig. 2.10 **a** Fluoroscopy guiding pacemaker implanting (from knol.google.com/k/pacemakers, published under Creative Commons license). **b** Angiogram of cranial vessels showing an aneurism (the eye sockets and parts of the skull are visible as well and hide some of the small vessels). **c** Digital subtraction angiogram of (**b**)

Fig. 2.11 Angiographic imaging device (C-arm) which is used for guiding minimally invasive interventions (Siemens Artis zeego, with kind permission from Siemens Sector Healthcare, Erlangen, Germany)

modern angiographic image acquisition system). Such acquisition systems can also be used to reconstruct 3d images from a sequence of projection from different angles similar to computed tomography (see Sect. 2.1.5 below).

Anatomic information from all other structures can be removed when an image is subtracted, which was made prior to giving the contrast agent. Although it is possible—and has been done—to do the subtraction mechanically using film, it is now done on digital images. The technique is called *digital subtraction angiography* (DSA, see Fig. 2.10c).

DSA enhances vessels much more compared to angiography, but images may suffer from motion artifacts. This is particularly true in cardiac angiography. Cardiac motion is too fast for creating two images without heartbeat influencing the result. Gated imaging is possible, but it may still be difficult to choose two images with and without contrast agent which were taken at exactly the same point in the heart cycle. Furthermore, motion due to breathing and patient movement cannot be removed by gating. DSA images from cerebral blood flow are affected to a much smaller extent by motion artifacts from heartbeat and patient motion.

Motion artifacts cannot be corrected easily because of the location dependency between 3d motion and its projection onto a 2d plane. If corrected at all, motion correction consists of selecting an optimal null image (image without contrast

Fig. 2.12 Goal of human or computer-assisted analysis in mammography is to find calcifications which are potential tumor sites (images with kind permission from Siemens Sector Healthcare, Erlangen, Germany)

agent) from several images, which after subtraction minimizes some error measure. Some research work on non-rigid registration between the two images in DSA has been reported, but as far as we know, none of this has reached maturity up to a point that the technique is routinely applied (see Meijering et al. (1999) for a review).

2.1.4 Mammography

Bremsstrahlung is the major influence in most X-ray tubes with the exception of X-ray tubes for *mammography* (see Fig. 2.12). The purpose of mammography is to detect small non-palpable lesions in the female breast. This requires a much higher image quality than normal X-ray imaging with respect to contrast and spatial

resolution. Since contrast and resolution are affected by scattering, mammography tubes reduce bremsstrahlung by suitable filtering. Furthermore, mammography tubes use a material molybdenum) which produces an almost monochrome X-ray with peak energies around 17–19 keV. This would be unwanted in regular X-ray imaging as most—if not all—of the radiation would be absorbed and not reach the receptor. For the breast, however, the use of low energy beams increases contrast between subtle differences of different tissues. Using an (almost) monochromatic beam will also reduce scatter, which again increases contrast.

Differences and findings in mammograms are subtle. Hence, *digital mammography* with its potential for post-processing has received much attention for quite some time, see, e.g., Chan et al. (1987), Dengler et al. (1993), Cheng et al. (1998). Digital mammograms have numerous advantages. They can be easily distributed and accessed in a hospital network. The dynamic range of digital detectors (about 1000:1) is much higher than that of film (40:1). Digital mammograms can be created using currently available TFT flat panel detectors, but owing to the need for a high resolution, the development of techniques reaching an even better resolution is still an active research field.

2.1.5 Image Reconstruction for Computed Tomography

Images from X-ray attenuation discussed so far are projection images. Structures are projected on top of each other. High attenuating objects such as bone may hide other objects. In cranial images, for instance, the skull would hide most of the details of the enclosed soft tissue. Furthermore, the cone beam geometry makes measurement of absolute distances impossible.[5]

Tomography (from the Greek "tomos" = cut, slice) attempts to create an image of one or more slices through the body. A set of slices provides a detailed 3d distribution of X-ray attenuation per volume unit. The name *computed tomography* (CT, also called CAT = computed axial tomography), emphasizes that these images are not acquired directly by some clever imaging device but are computed from projection measurements. X-ray computed tomography has been invented in the 1970s by Godfrey N. Hounsfield based on work of Alan M. Cormack[6] (Buzug 2008).

Tomography had been known before as an analog technique. It produced slices through the body directly on film by moving film and X-ray source in opposite directions during exposure (Geluk 1979). Depending on the ratio between speeds of

[5]There are exceptions, if, e.g., images are made by two different cone beams of a rotating C-arm where the magnification of objects in the center of both cones can be computed.

[6]Alan M. Cormack has been a physicist at the University of Cape Town in South Africa when he developed the theoretical basis for Computer Tomography in the late 1950s. This work was taken up later by Godfrey N. Hounsfield, an electrical engineer at EMI Research Lab, who presented the first whole-body CT scanner in 1975. For their invention, both received the Nobel Prize of Medicine and Physiology in 1979.

film cassette and X-ray source, a specific single slice parallel to the image plane was imaged in focus overlaid by blurred layers above and below this slice.

Computed tomography goes a different way. It produces a digitized solution of the inverse *Radon transform* from projections in a slice without interference from other slices (see Buzug (2008) for a detailed treatment of reconstruction techniques). The inversion is computed numerically from projections by the imaging computer.[7]

For a two-dimensional attenuation function $\mu(x, y)$ describing the X-ray attenuation per unit volume in some slice z in the human body, the Radon transform is given by all line integrals through this function:

$$R(s,\theta)[\mu(x,y)] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mu(x,y)\delta(s - x\cos\theta - y\sin\theta)\mathrm{d}x\,\mathrm{d}y, \tag{2.4}$$

where δ is the Dirac-Delta function.

In other words, for a given angle θ, the Radon transform produces the projections onto a line s along rays perpendicular to s with angle θ to the x-axis. The Radon transform is invertible, which means that $\mu(x, y)$ may be reconstructed from all projections onto lines s for all angles $0° < \theta < 180°$. This essentially solves the reconstruction problem. Since

$$I_{\text{out}} = I_{\text{in}} \cdot \exp\left(-\int_{t_0}^{t_1} \mu(t)\mathrm{d}t\right), \tag{2.5}$$

we have

$$\int_{t_0}^{t_1} \mu(t)\mathrm{d}t = -\ln(I_{\text{out}}/I_{\text{in}}) \quad \text{with} \quad t = s - x\cos\theta - y\sin\theta. \tag{2.6}$$

If we assume that no attenuation takes place outside of $[t_1, t_2]$, we can extend the bounds of the integral to infinity and

$$R(s,\theta)[\mu(x,y)] = \int_{-\infty}^{\infty} \mu(t)\mathrm{d}t = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mu(x,y)\delta(s - x\cos\theta - y\sin\theta) \tag{2.7}$$

may be computed from radiographic projections at angles $0° < \theta < 180°$. If we had enough of these projections, the Radon transform could be inverted numerically and a 3d distribution of *attenuation coefficients* is reconstructed.

[7]An optical solution for the reconstruction exists as well which has been proposed in the 1970s in order to overcome limitations because of lack of computing power Geluk (1979).

A CT scanner that acquires projections consists of a single-slice or a multiple-slice detector ring around which an X-ray source rotates (see Fig. 2.13 for a schematic view and Fig. 2.14 for an image acquisition system). During rotation, detections are constantly made. It can be thought as a device that creates single or multiple lines of X-ray images from different angles. Measurements are created much faster than creating complete X-ray images from multiple angles.

Reconstruction by inversion of the Radon transform creates single or multiple slices of voxels containing attenuation per unit volume in the imaged patient.

A stack of slices is produced by scanning a sequence of different slices or groups of slices. For the scanning process, the patient lies on a table that moves into the detector ring in prespecified steps. The movement is called (translational) *gantry*. The step size of the horizontal movement determines the distance between slices. The thickness of the detector ring determines the slice thickness. Slice thickness is often lower than the distance between slices causing a gap between slices.

The number of slices and their thickness for a CT study has changed dramatically with the advent of multi-slice scanners. In early times of computed tomography, a slice thickness of 5 mm with 5-mm gap or even thicker slices was quite common. A brain image data set from CT may have contained less than 20 slices. A detailed analysis, e.g., of the fine structures in the brain, was difficult to impossible. Newer scanners produced thinner slices, but motion during image acquisition often caused artifacts and image quality deteriorated. An advance was the development of the *spiral scanner*. In a spiral scanner, the X-ray source rotates in a continuous spiral while measuring absorption data. Resolution and number of images have been further improved with the use of multi-slice detectors which acquire more than one slice simultaneously (Buzug 2008).

Fig. 2.13 Schematic view of CT image generation. The patient is placed on a moveable table. A CT image is a slice that is reconstructed from multiple projections taken from different angles. A sequence of CTs is created from translating the table and repeating the image acquisition procedure

Fig. 2.14 An X-ray CT scanner (Siemens Somatom, image with kind permission by Siemens Sector Healthcare, Erlangen, Germany). Image acquisition takes place in the doughnut-shaped gantry that houses X-ray source and detector rings (which, in this system, can be tilted). The patient is moved through the gantry while slices of CT images are acquired

A body CT study may easily contain 500–1000 slices with slice distances of 1 mm or less. Slice thickness may vary, however, because of dose considerations. Thin slices with high resolution may be acquired in the region of interest for capturing as much detail as possible. Slice thickness may be much larger in the context region for providing necessary anatomical context.

Inverting the Radon transform for slice reconstruction from projections implies parallel projection, whereas X-ray images are acquired in a cone beam projection. It turns out that this is solvable by transformations not essentially affecting the reconstruction method. We thus will assume parallel projection for the remainder of this discussion.

Another problem of image reconstruction is harder. The Radon transform assumes a continuous space for s, θ, x, and y. An image will be reconstructed from a limited number of projections and a limited number of measurements per projection. Fortunately, just a finite number of voxels needs to be reconstructed with attenuation coefficients assumed to be constant within the voxel. Nevertheless, the influence of the digitization on the inversion of the Radon transform needs to be investigated.

Inversion is by means of the Fourier transform, which leads to the *filtered backprojection* (FBP) algorithm. The Fourier transform of the Radon transform delivers what is known as the *Central Slice Theorem*. The theorem states that the Fourier transform $M(u, v)$ of a projection of a function $\mu(x, y)$ in a direction with angle θ to the x-axis equals the coefficients on a line with this angle θ in frequency

Fig. 2.15 Central Slice Theorem states that the Fourier transform of a projection onto a line with some angle α equals a line in frequency space of the Fourier transform with equal angle of the image

space of the Fourier transform of μ (see Fig. 2.15). This can easily be shown for a line with $\theta = 0$, where

$$\begin{aligned}\mathrm{FFT}[R(s,0)(\mu(x,y))] &= \mathrm{FFT}\left[\int_{-\infty}^{\infty} \mu(x,0)\mathrm{d}x\right] = \int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty} \mu(x,0)\mathrm{d}x\right]\exp(-ivy)\mathrm{d}y \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mu(x,0)x\exp(-i(0u+vy))\mathrm{d}x\,\mathrm{d}y = \mathrm{M}(x,0)\end{aligned} \tag{2.8}$$

The theorem is true for all other angles θ as well because of the rotation property of the Fourier transform. Hence, projections of $R(s, \theta)[\mu(x, y)]$ for all angles θ are transformed into 1d frequency space. The coefficients for a projection $R(s, \theta)$ are mapped on the corresponding line with angle θ in 2d frequency space. The inverse Fourier transform is then applied for computing the reconstructed image.

There are two problems with this. Firstly, the Fourier transform from projections is defined in a polar coordinate system, whereas coefficients for carrying out the inverse transform are required in Cartesian coordinates. We need to account for the difference in the size of a unit area in polar and Cartesian coordinates. Otherwise, low-frequency components in the image would be overly emphasized (see Fig. 2.16). Correction is done by multiplying the Fourier coefficients with $1/r$, where r is the radial distance of a given location (u, v) to the origin. Hence, it is an application of a radial *ramp filter* (see Fig. 2.17 for a filtered reconstruction).

Secondly, the distribution of Fourier coefficients from polar coordinates becomes sparser in the high-frequency range of the Cartesian coordinate system. It may cause artifacts because noise often predominates in the high-frequency range.

Fig. 2.16 Projecting along angles between 0° and 180° produces a projection image (which is also called sinogram). Unfiltered reconstruction results in artifacts because the varying density of Fourier coefficients has not been accounted for

Fig. 2.17 Filtering the projection with a ramp filter that attenuates amplitudes with decreasing frequency removes artifacts from simple reconstruction

Hence, projections are usually low-pass filtered by filters such as the *Hamming window* or *Hamming filter* (see Sect. 4.3 on image enhancement).

Inversion of the Radon transform using the Central Slice Theorem does not really require transforming the projection data into frequency space. The Fourier transform is a linear operator. Adding terms for the Fourier transform or its inverse may be carried out in any order. One particular order would be to transform a line s of projection data, filter the result, and invert the transformation and then carry on with the next line. If we look at the process more closely, we see that this can be replaced by convolving the projection data of this line with a convolution function which is the spatial domain equivalent of the filter function, and then project the result perpendicular to s onto the image. This is to be done for every projection and gave the method its name *filtered backprojection* or *convolution backprojection reconstruction*.

The reconstruction result is a digital image of attenuation coefficients for each voxel with the thickness of the slice and a size along x and y according to the chosen in-plane spatial resolution. Common image sizes for CT images are 512×512 or 256×256. The size of the voxel along the x- and y-axes depends on the field of view (FOV) which depends on the opening angle of the cone beam

Table 2.1 Hounsfield units of different tissues

Air	Fat	Water	Blood	Muscle	White matter	Gray matter	CSF	Bone
−1000	−100	0	30–45	40	20–30	37–45	15	>150

Note Air, water, and bone are well differentiated, while contrast between different soft tissues is low

scanning the patient. For brain images, e.g., with a FOV of about 30 × 30 cm^2, in-plane voxel sizes for a 512 × 512 image are 0.5 × 0.5 mm^2.

Attenuation coefficients are normalized for making the result independent of imaging parameters such as beam energy. The scale is called *Hounsfield scale*. Normalization is based on the attenuation μ_{Water} of water and μ_{Air} of air:

$$\text{HU}(\mu) = 1000 \cdot \frac{\mu - \mu_{\text{Water}}}{\mu_{\text{Water}} - \mu_{\text{Air}}} \tag{2.9}$$

Thus, air has −1000 HU and water has 0 HU. Hounsfield units for different tissue types are given in Table 2.1. Hounsfield units are mapped to integers and usually represented in a range from −1000 to 3000. Hence, attenuation coefficients can be represented by two byte integers.

The display software of a scanner system lets the user choose an appropriate HU range to be mapped onto the displayable range of 256 gray values by specifying *window* (width of the range to be mapped) and *level* (value to be mapped onto gray value 128). Different window and level settings are used to emphasize different tissues. A bone window, for instance, will have a high level and a large window because attenuation for bone is high and extends over a wide range. A lung window, on the other hand, will have a much lower level, since most of the lung is filled with air.

Air, fat, water, and bone have significantly different attenuation, whereas differences between the various soft tissues are small (see Table 2.1). In fact, results from first experiments of CTs of the head were not overwhelming, and medical equipment companies were reluctant to build a CT scanner.[8] Scanners have much improved since then in terms of spatial resolution as well as in terms of suppressing noise and artifacts. CT is probably still the most often used digital image device in diagnosis and surgical intervention. Typical application areas for diagnostic imaging are bone CT, CT of the brain, and the parenchyma.

A number of artifacts may occur when creating the image by FBP, which have to be accounted for in a later image analysis:

- *Noise* (Fig. 2.18) is caused from scatter in the patient and at the detector. Since the likelihood of a photon of reaching the detector unscattered can be modeled as a Poisson process, the noise is often modeled by a Gaussian function (which is a good approximation of a Poisson distribution for large numbers of events).

[8]The first CT scanner was built by a record company (EMI).

Fig. 2.18 Noise and the partial volume effect (PVE) in CT imaging. The PVE causes blur at skin, fat, bone, and CSF boundaries. Noise amplitude in this image is in the range of the signal increase at the object boundary

- The *partial volume effect* (PVE, Fig. 2.18) occurs because attenuation within a voxel may not be uniform. The PVE within a slice is usually easy to recognize. It is most prominent between adjacent voxels with very different attenuation coefficients such as at the bone–CSF boundary. The reconstructed attenuation coefficient will be a value between that of the two tissues. The PVE occurs between slices as well and may be less obvious because the object causing the PVE may be in the slice above or below the currently displayed slice.
- *Metal* artifacts are caused by the very high difference between the attenuation of metal and tissue. This makes the reconstruction less robust against other influences such as noise and PVE.[9] It results in severe *streak* artifacts. Such artifacts can be observed at all high contrast boundaries but are particularly severe in the case of metal artifacts.
- *Motion* artifacts are the result of voluntary or involuntary patient movement (movement due to breathing, heartbeat, etc.) during acquisition of a single slice. Motion violates the assumption that attenuation at some location (x, y) is the same for every projection and causes blur in the image. Since the influence from motion may be different for different slices, it may cause the so-called *step-ladder artefact* between slices [visible in re-oriented views, see, e.g., Ghersin et al. (2006)].

[9]The problem of image reconstruction can be stated as an algebraic problem of finding unknowns —the attenuation—from linear equations—the projections—and it can be shown that the problem becomes more ill-conditioned when some very high attenuation regions exist.

Tomographic imaging using X-rays can also be carried out using X-ray equipment for projection images. Early attempts were the use of image intensifiers for rapid imaging of the heart [Mayo Clinic, Robb et al. (1982)]. Newer developments employ angiographic acquisition systems for reconstruction (Ning et al. 2000; Akpek et al. 2005). Since angiography uses a moveable and rotatable C-arm, projections can be reconstructed from programmed C-arm rotation. The receptor device is a TFT flat panel detector with up to 1000 × 1000 elements. Each projection simultaneously acquires data from 1000 slices with 1000 projections per line. Reconstructed images thus have a higher spatial resolution than ordinary CT images and produce all slices at the same time. It comes at a cost, however. Current systems are not fast enough to generate projections from as many angles as a conventional CT system. In consequence, artifacts due to an insufficient number of measurements increase. Images are noisier, and streak artifacts are more pronounced.

2.1.6 Contrast Enhancement in X-ray Computed Tomography

Contrast agent used in fluoroscopic imaging or angiography is used in X-ray CT as well. Instead of showing a time-varying function such as in fluoroscopy, the use of a contrast agent in X-ray CT enhances structures which are otherwise difficult to differentiate. A major application is the depiction of vessels [*CT angiography* or CTA, Dillon et al. (1993)].

CTA works similar to ordinary X-ray angiography with the difference that images are not projections (see Fig. 2.19a for an example). The 3d nature of CTA images allows quantitative analysis by direct measurement on the image. With the spatial resolution of today's scanners, the diagnosis of the extent of a stenosis becomes possible through CTA even for smaller vessels. It needs to be kept in mind, however, that CTA requires a higher exposure to X-rays than X-ray angiography. Compared to subtraction angiography (a reconstruction from DSA is depicted in Fig. 2.19b), CTA provides information about soft tissues not visible in DSA. On the other hand, the intensity of vessels filled with contrast agent is similar to that of bone, making it difficult to separate vessel structures close to the skull from bone structures.

2.1.7 Image Analysis on X-ray-Generated Images

Radiographs have a high spatial resolution which supports the detection of small lesions such as microcalcifications in mammography, potentially cancerous nodules in lung scans or small stenoses in angiography. The signal is bright for dense structures and dark for low attenuation objects.

As the image is a projection, no assignments between absolute brightness value and tissue type can be made. Distance measurements are not possible, except for

Fig. 2.19 CT angiography (**a**) shows contrast-enhanced vessels. While showing also tissue in the context, vessels are more difficult to detect than in reconstructed DSA images (**b**)

measurements of distance ratios between structures of which the approximate distance to the radiation source is known to the radiologist. Over- or underexposure (for digitized film) may reduce the contrast, and images may be blurred due to motion or the size of the focal spot.

Recognition of small objects may be hindered by the fact that structures hide other structures in the projection. Projection may also cause a very convoluted appearance of larger objects (such as complex bone fractures), which bears little resemblance to the actual shape of this object. Deducing the true geometry of such an object from its projection may be easy to the experienced radiologist but difficult to implement in an algorithm. It requires extensive context knowledge about projection direction, orientation of organs being imaged, and their relative position with respect to the projection direction.

Creating tomographic images reduces or removes some of the problems above, which is one of the reasons for the sudden increase in computer-assisted analysis methods in medicine with the appearance of CT imaging in the 1970s. CT images have a lower spatial resolution than radiographs, which is why radiographs may still be preferred if diagnosis requires high resolution images.

If noise, partial volume effects, and other artefact are disregarded, a mapping between tissue type and attenuation coefficient exists in a voxel of a CT. The normalized attenuation coefficients of the Hounsfield scale make mappings from different images created by different machines comparable. The value of this is not to be underestimated. Even though the mapping is only approximate and distorted by artifacts and even though it is certainly not invertible (i.e., a certain HU value is not uniquely associated with a certain type of tissue), it comes close to the primary goal of imaging, i.e., directly measuring and displaying tissue and tissue characteristics.

2.2 Magnetic Resonance Imaging

Protons and neutrons of the nucleus of an atom possess an angular momentum which is called *spin*. These spins cancel if the number of subatomic particles in a nucleus is even. Nuclei with an odd number exhibit a resultant spin which can be observed outside of the atom. This is the basis of *magnetic resonance imaging* (MRI) (Liang and Lauterbur 2000). In MRI, spins of nuclei are aligned in an external magnetic field. A high-frequency electromagnetic field then causes *spin precession* that depends on the density of magnetized material and on its molecular binding. The resonance of the signal continues for some time after this radio signal is switched off. The effect is measured and exploited to create an image of the distribution of the material.

The resonance effect has been used in MR spectroscopy for quite some time. A detailed description of MR spectroscopy and its clinical applications can be found in Salibi and Brown (1998). However, imaging, i.e., the computation of the spatial distribution of effects from magnetic resonance did not exist before the 1970s. Image generation with magnetic resonance is due to Paul C. Lauterbur and Peter Mansfield.[10]

Magnetic resonance imaging almost exclusively uses the response of the hydrogen nucleus which is abundant in the human body. Variation in hydrogen density and specifically its molecular binding in different tissues produces a much better soft tissue contrast than CT. MRI has some further advantages if compared with X-ray CT:

- MRI does not use ionizing radiation.
- Images can be generated with arbitrary slice orientation including coronal and sagittal views.
- Several different functional attributes can be imaged with MRI.

In summary, MRI is a remarkably versatile imaging technique justifying an extended look at the technique.

2.2.1 Magnetic Resonance

As mentioned above, nuclei with an odd number of protons or neutrons possess a spin. To produce a resonance image, spins of all nuclei of the body are aligned in a static magnetic field B_0. The strength of the magnetic field is measured in Tesla (T) or Gauss (10,000 Gauss = 1 T). Today's MR imaging devices for human full body imaging operate with field strengths between 1 and 3 T.

[10]Peter Mansfield is a British physicist who provided the basis for interpreting the signal from resonance. Paul C. Lauterbur is a chemist who first provided the methodology for turning the signal into an image. For their achievement, both received the Nobel Prize for Physiology and Medicine in 2003.

Fig. 2.20 Spins are aligned parallel or antiparallel to the B_0 field. They produce a signal induced by an RF field. Just a small excess amount of protons aligned parallel produces the signal, since signals from parallel and antiparallel aligned protons cancel

The static field causes spins to be aligned either parallel or antiparallel to the magnetic field. The strength of the measurable signal depends on the difference between these two types of alignment. This, in turn, depends on the type of atom, on the magnetic field strength and on the temperature at which the measurement takes place. The atom-specific sensitivity is highest for hydrogen.

A precession of spins around an axis parallel to the B_0 field can be induced by a radio signal. The effects of parallel and antiparallel precessing spins cancel. For hydrogen at normal room temperature and a magnetic field of 1.5 T, the ratio between parallel and antiparallel spins is approximately 500.000:500.001. Hence, the observed net signal from all spins will come from just one in a million protons (see Fig. 2.20).

Given that the hydrogen atom possesses the highest sensitivity of all atoms and that the temperature cannot be increased arbitrarily, the only way to increase the signal is to increase the magnetic field. This is the reason for the exceptionally high magnetic fields that are applied in MRI (for comparison, the strength of the earth magnetic field is 0.00003–0.00006 T).

If spins are aligned in the external B_0 field, their angular momentum, i.e., their spin frequency ω, depends on the strength of B_0 and an atom-specific *gyromagnetic constant* γ:

$$\omega = \gamma B_0 \tag{2.10}$$

The frequency ω is called *Larmor frequency*. The gyromagnetic constant for water is 42.58 MHz/T which translates into a spin frequency of 63.87 MHz for an MRI scanner with a static field with 1.5 T.

Fig. 2.21 Free induction decay experiment: spins of excited hydrogen atoms are projected into the *xy*-plane orthogonal to the B_0 magnetic field. After switching of the RF signal, the original *z*-magnetization is restored with time constant T_1. The observed signal decreases much faster since spins start to dephase

In order to produce resonance, additional energy is supplied causing the spins to precess around the direction of the B_0 field.[11] It is applied by an electromagnetic field which is perpendicular to B_0. The field effects proton spins only if it changes with the same frequency than the rotation frequency of the spins. Hence, if the strength of the B_0 field would be 1.5 T, the required frequency of the resonance triggering field would be 63.87 MHz. This high-frequency field (HF) radio wave is supplied by a radio antenna.

The radio signal tilts the spins by an angle $0° < \alpha < 180°$ from their rest direction. The magnitude of the angle depends on the energy of the HF field signal (see Fig. 2.21). After the signal is switched off, the spins slowly return into their rest direction parallel or antiparallel to the B_0 field while still carrying out the precession. This can be measured by an antenna (which could be the sender of the HF field acting as receiver). The observed decreasing signal is called *free induction decay* (*FID*). Its amplitude depends on several factors:

- The amplitude is proportional to the number of spins. Hence, the amplitude per unit volume is proportional to the *proton density*.
- Spins returning to their rest direction will deliver a signal of decreasing magnitude, which will cause the FID signal decay with time. The restoration of the original spin direction is called *longitudinal relaxation*. Magnetization $M_z(t)$ in *z*-direction (the direction of the B_0 field) at time *t* is

$$M_z(t) = M_0\left(1 - c \cdot \exp\left(-\frac{t}{T_1}\right)\right) \tag{2.11}$$

[11]The following discussion relates to effects observed in voxels containing a large enough number of photons so that quantum effects can be neglected.

with M_0 being the magnetization in M_0 before excitation. The time constant of the exponential relaxation is called *T_1-time*, *spin-lattice relaxation time*, or *longitudinal relaxation time*.

- Spinning protons act as magnets and influence the behavior of adjacent protons. Precession of spins will dephase with time, and the observable FID signal will vanish long before the original magnetization in z-direction is restored, since the antenna measures a vector sum of all spins (see Fig. 2.22). The signal decay is exponential as well and is described by a constant which is called *T_2-time*, *spin-spin relaxation time*, or *transverse relaxation time*. If $M_T(t)$ is the transverse magnetization perpendicular to z at time t, then

$$M_T(t) = M_T(0) \cdot \exp\left(-\frac{t}{T_2}\right) \tag{2.12}$$

T_1 and T_2 constants for the same proton differ depending on its molecular binding. Protons in water have a larger T_1 than in soft tissue, while T_2 is smaller in soft tissue. T_1 and T_2 in water are in the range of seconds.

Additionally to the tissue-specific attributes, an unwanted effect causes a much faster signal decay than T_2 relaxation. Signal generation is made under the assumption that the static B_0 field is constant everywhere in the imaged region. Even small field variations cause a much quicker dephasing of spins than dephasing by spin–spin relaxation. Signal decay due to an inhomogeneous magnetic field is exponential with a constant T_2*. The T_2* effect is unwanted, as it interrogates the homogeneity of the magnetic field but not the chemical composition of the probe to be imaged.

Fig. 2.22 Immediately after the RF signal is switched off, all spins are in phase giving the maximum signal. Locally different magnetization from neighboring atoms causes dephasing as does inhomogeneity of the B_0 field. The observed net magnetization decreases, since its magnitude is the magnitude of the vector sum of all protons in a voxel

The resonance signal allows analyzing the chemical composition of a probe in MR spectroscopy. Instead of exciting just hydrogen, the occurrence and density of other atoms with an odd number of protons or neutrons may be measured by applying a frequency band encompassing all the frequencies necessary to excite the respective atoms. Frequency analysis and measurement of decay times enables separation of the signal in constituents caused by the different materials. For imaging, however, additional measures need to be taken to localize the MR signal.

2.2.2 MR Imaging

Extending the MR technique for creating an image is due to the work of Mansfield and Lauterbur. The dependency of excitability of a proton on the frequency of the HF field is used for the necessary localization. It requires additional *gradient magnetic fields* (called x-, y-, or z-gradient according to the direction of increasing field strength).

If, before applying the HF-field, a linear gradient field in z-direction is overlaid to the B_0 field, the magnetic field strength will linearly increase in direction of z. Protons will spin with a frequency that depends on their location in z. If an HF field is applied whose frequency range corresponds to spin frequencies of protons in a slice with $z_1 < z < z_2$, only those protons will be excited and produce a resonance signal (see Fig. 2.23). The process is called *slice selection*. After excitation, the z-gradient is switched off and protons in the selected slice spin with a frequency as determined by the B_0-field.

In order to localize protons within a slice, a similar technique is applied while reading the signal. If a linear gradient field in x-direction is applied during observation of the resonance signal, the recorded signal will contain a range of frequencies (see Fig. 2.24). The process is called *frequency encoding*. The gradient is called *readout gradient*. Frequencies w_0 to w_{max} correspond to x-values x_0 to x_{max}.

Fig. 2.23 Slice selection is done by selecting an RF band that corresponds to a specific resonance frequency band given by the B_0 field and the slice selection gradient

Fig. 2.24 Frequency encoding of the signal is done by a gradient field in the *xy*-plane. It causes spins to rotate with different frequencies at readout time so that atoms can be associated with lines orthogonal to the gradient field

The resonance signal is then transformed into frequency space. The amplitude value at frequency $w_0 + (w_{\max} - w_0) \cdot (x_k - x_0)/(x_{\max} - x_0)$ corresponds to the response of all protons on the line $x = x_k$.

The technique produces a projection of spin densities along y onto the x-axis. A complete set of projections can be obtained by adding projections at measurements with gradient fields along different directions. Having obtained the data, an image could be reconstructed by filtered backprojection. This was the first technique for reconstructing MR images.

Today's scanners employ a different technique, which is called *k*-space imaging (*k*-space is another word for frequency space). *K*-space imaging generates the image function in frequency space directly through measurements. The first line in *k*-space is already the result of frequency encoding as described above. It produces the integral of spin densities along lines of constant y-value and can be interpreted as the first line of values of a Fourier transform in u-direction (containing the coefficients with frequency 0) assuming that the Fourier transform maps (x, y) to (u, v).

Let us now consider the signal before applying the frequency encoding gradient. If we apply another linear gradient in y-direction prior to making the measurement, spin frequency will increase along y and the spins will dephase. The gradient is called *phase encoding* gradient. It is applied just long enough in order to dephase spins between 0 and 2π for the range of y-values between y_0 and $y_{\max}$. After switching off phase encoding, frequency encoding is applied and the obtained signal is transformed into frequency space. The results are integrated spin densities along x weighted with a cosine wave of frequency 1 for each line of constant y. In other words, we have produced the real-valued part of the next line in *k*-space of the Fourier transform in u-direction (see Fig. 2.25). The imaginary part is computed by weighting the signal with the corresponding sine wave.

Fig. 2.25 Gradient field for each phase encoding causes a controlled dephasing which acts as a weighting of the observed signal with a cosine function. Phase encoding is done before readout, and the phase encoding gradient is switched off before readout

The measurement is repeated with phase encoding for the remaining lines in *k*-space. Once the *k*-space is filled, the image is generated by transforming it back into the spatial domain.

Image equipment looks similar to CT (see Fig. 2.26). However, the gantry in which the B_0 field is produced is usually smaller than a CT gantry. Image planes need not to be perpendicular to the direction of the B_0 field, since gradients may be generated in arbitrary directions. It is a major difference compared to CT imaging. Also, a variety of different images may be produced form the same patient showing

Fig. 2.26 Gantry of an MR imaging device (Magnetom Avanto, with kind permission from Siemens Sector Healthcare, Erlangen, Germany). Images can be produced at arbitrary angles with respect to the gantry by producing appropriate slice selection gradients

different aspects of the resonance signal. Three different parameters—spin density ρ, spin-lattice relaxation T_1, and spin–spin relaxation T_2—determine the resonance signal. Hence, different sequences can be developed for enhancing either of the parameters [some of the major sequences used in MR imaging will be discussed in the next section; a more detailed treatment can be found in Liang and Lauterbur (2000)]. It changes the appearance of different tissues in images (e.g., water and fat is bright in T_2 images and tissue is darker, while the opposite is true for a T_1 image). A normalized scale such as the Hounsfield units of CT does not exist.

2.2.3 Some MR Sequences

T_1 and T_2 time constants cannot be measured directly because signal strength is always influenced by proton density and because field inhomogeneities (the T_2* decay) hide the T_2 effect. T_2-enhanced images can be generated by the *spin echo* sequence. The sequence uses a clever mechanism to cancel out T_2* effects. It consists of a 90° impulse which tilts spins into the *xy*-plane followed by a sequence of 180° impulses producing spin echoes.

After the application of the 90° impulse, spins start to dephase due to the combined influence of spin–spin relaxation and field inhomogeneities. Spin–spin relaxation continues after the application of the 180° impulse since the relative spin orientation although turned by 180° remains the same. T_2* influence is inverted, however, because higher frequency spins, which had a positive phase difference, now have a negative phase difference with respect to spins with lower frequency. Spins will rephase with respect to field inhomogeneities and produce the first echo of T_2-relaxation (see Fig. 2.27). They dephase again until the next 180° impulse causes the next echo. The envelope containing the echoes decreases exponentially with T_2.

Fig. 2.27 Spin-echo sequence causes dephasing spins to rephase by flipping by a 180° impulse which reverses the direction of precession

Usually, a single echo will be taken as image. The time between 90° impulse and the echo impulse is called *echo time TE*. The time between two measurements is called *repetition time TR*. Short *TE* (20 ms) and long *TR* (2000 ms) will produce a proton-density-weighted image. Using a shorter repetition time (*TR* = 300–600 ms) will produce a T_1-weighted image because T_1 relaxation is generally longer than 200–600 ms. A long *TE* (>60 ms) and a long *TR* (2000 ms) produce a T_2-weighted image.

The *inversion recovery* sequence is another sequence used in MRI. It produces an image that is strongly influenced by the T_1 time constant. In inversion recovery, a 180° impulse is followed by a 90° impulse. The time between the two impulses is called *TI* or *inversion time*. The 180° impulse leads to saturation i.e., a new impulse would not produce a signal. However, after time *TI* protons have recovered their longitudinal magnetization to an extent which depends on their individual T_1 time. Hence, they are able to produce a resonance signal from the 90° impulse. An echo at echo time *TE* rephases spins which are then read out. Long inversion times (*TI* > 2000 ms) and short echo times (*TE* = 10 ms) produce a proton-density-weighted image. Shorter inversion times (400–800 ms) with the same echo time produce a T_1-weighted image, while choosing a long echo time (*TE* > 60 ms) will produce an image that is T_2-weighted.

MR images of the head created by imagining sequences like the ones above usually have a slice thickness of 1–3 mm and 256 × 256 voxels in a slice. Body MR images usually have 512 × 512 voxels per slice. Data acquisition times using either of the sequences are longer compared to CT imaging. 256 lines in *k*-space need to be measured for reconstructing an image of size 256 × 256 pixels. This requires 256 different measurements. A new measurement is only possible after effects from the previous measurement have decayed. With T_1-time constants in the range of a second, it results in repetition times in the range of several seconds. It leads to data acquisition times for the image in the range of 10–15 min. If image size doubles, acquisition time doubles as well. If measurements are repeated in order to increase the signal-to-noise ratio or if more than one measurement is desired, acquisition times for a full study may easily reach 30–60 min. The acquisition of a sequence of slices, however, does not necessarily take much longer. Since slice selection only influences protons of a given slice, signals from several slices may be acquired in an interleaved mode.

In the early 1980s, acquisition times have been a serious drawback. Since the 1990s, a wealth of fast imaging sequences has been developed. *Gradient echo imaging* was one of the first techniques to speed up imaging time. In gradient echo imaging, an excitation pulse of α (with $\alpha \leq 90°$) is given. However, instead of using the 180° impulse to produce a rephased spin echo, an inverse gradient is applied in order to cause the echo. Echo times between applying the second gradient and readout are much faster than echo times of the spin echo sequence. Pulses with α close to 90° produce a T_1-weighted image. Low angle pulses result in a T_2*-weighted image. The image is not T_2-weighted because, opposed to spin echo imaging, the gradient echo reverses dephasing from inhomogeneities and spin–spin relaxation in the same way.

Fig. 2.28 Main difference between normal encoding and a fast imaging sequence is that in fast imaging, several to all encodings are done during a single excitation of the resonance signal

Turbo spin echo sequences make several measurements at a time. This is done by applying a sequence of several phase shift gradients during a single echo and reading the signal after each phase shift (see Fig. 2.28). Several lines of *k*-space are filled at the same time, and image acquisition time decreases by a factor of the number of lines acquired per repetition. In its extreme, all lines are acquired from a single excitation (RARE—Rapid Enhancement with Relaxation Enhancement) so that an image can be acquired within time *TE*. RARE is a variant of an older sequence known as Echo Planar Imaging (EPI), a term which simply refers to acquiring the complete *k*-space in a single resonance experiment.

The treatment above provides just a short glimpse at the potential of producing different information from the same subject by varying imaging parameters in MRI. Readers interested in the details and potential of MR imaging are referred to texts with just this focus, such as Brown et al. (2014).

2.2.4 Artifacts in MR Imaging

Ultrafast sequences such as EPI produce images in less than 100 ms. There are problems, however. While making the measurements from a single echo, T_2* dephasing continues to occur. Different lines in *k*-space are measured with a different contrast with respect to T_2* decay. Moreover, the number of lines to be measured in *k*-space is usually restricted to 128 or less so that the spatial resolution of a single-shot image does not exceed 128 × 128 pixels. Furthermore, artifacts from chemical shift, ghosting, and shading may be particularly pronounced for fast imaging sequences.

The *chemical shift* of protons (and any other nuclei) is caused by magnetic shielding from electrons in the molecular environment. It depends on the molecular binding of protons and causes a material-dependent deviation of spin frequency. The difference is particularly apparent for protons in water, as compared to those in fat. Since frequency encoding interprets measured frequency as location, water protons will be reconstructed at an offset to fat protons in frequency encoding direction. Chemical shift is present in all non-shift-corrected images but is pronounced in EPI (RARE) images where the offset may amount to 8–10 pixels. At 3-mm voxel size, this is an offset of about 2.5 cm.

Ghosting appears because of inaccuracies in the phase encoding. If one or more lines in *k*-space are phase shifted, the corresponding waves in the spatial domain are shifted as well. Ghost images appear in this direction. Ghosting may also happen in regular imaging if the patient has moved in phase encoding direction between acquisitions. Motion in direction of frequency encoding causes the much less prominent blurring artifact similar to this kind of artifact in CT.

Shading is due to variation in attenuation of the RF signal and an inhomogeneous magnetic field. It causes differences in the resonance signal according to location. It is then turned into different intensities to be reconstructed for the same material at different locations.

Artifacts from noise and PVE are similar to CT imaging. Metal artifacts from paramagnetic materials causes signal deletion. Presence of ferromagnetic materials, such as implants, is a contraindication for MR imaging as is the presence of implanted electronic devices such as a pacemaker.

2.2.5 MR Angiography

MR angiography (MRA) exists with and without using contrast agents. Contrast-enhanced angiography uses *gadolinium*, an agent that causes a strong decrease in the T_1 relaxation time. Gadolinium-enhanced vessels can be imaged with a T_1-weighted sequence that saturates all other tissues while highlighting the vessels. The resulting contrast is so high that the images look similar to DSA images but without the necessity of subtracting a null image (see Fig. 2.29a). Vessels may be depicted even if they are smaller than a voxel because of the partial volume effect.

Fig. 2.29 Maximum intensity projections of gadolinium-enhanced MRA (*left*) and phase-contrast MRA (*right*). While using gadolinium, contrast agent produces images with better resolution, phase-contrast imaging is noninvasive and has the potential for computing vessel velocities (images from Siemens Sector Healthcare with permission)

MRA images come as a true 3d volume, but they are often displayed as *maximum intensity projection* images (MIP). This visualization technique projects the brightest voxel along the projection line on a pixel on the output screen. MIP is simple and fast and produces images similar to digital subtraction angiograms.

MR angiography without the use of a contrast agent exploits two different effects, which are called *flow void* and *phase-contrast angiography* (Dumoulin 1995).

A flow void occurs when a proton that has been excited by an HF impulse has moved out of the slice before the readout impulse is applied.

Phase-contrast angiography uses the motion artifact of moving blood for imaging. If a proton moves in direction of the phase encoding gradient, it carries its phase with it (see Fig. 2.30). Two different phase encoding gradients are applied at times t_1 and t_2. The second gradient is exactly opposite to the first gradient. Effects of the gradients on protons that have not moved between t_1 and t_2 will cancel. Protons that moved along the gradient direction will acquire a phase shift relative to static protons, which is proportional to the distance traveled between t_1 and t_2 (i.e., proportional to their speed). A 3d velocity vector for moving protons can be created by applying phase encoding in *x*-, *y*-, and *z*-direction. Velocity indicates the existence of a vessel (see Fig. 2.29b). Suggestions have been made that phase-contrast angiography may even enable analysis of velocity differences in blood vessels, e.g., at a stenosis or in an aneurism.

2.2.6 BOLD Imaging

Blood supplying the brain carries oxygen via hemoglobin. Oxygenated hemoglobin is diamagnetic such as all the other tissues, but deoxygenated hemoglobin is paramagnetic causing a small, local distortion of the magnetic field. The distortions change the measurable signal which can be made visible. The technique is called

Fig. 2.30 Principle of phase-contrast imaging: protons are phase-encoded by applying a gradient field. Motion in phase encoding direction leads to dephasing. If a rephasing gradient is applied, dislocated protons cause a signal loss since they are not completely rephased. The amount of signal loss depends on the motion in phase encoding direction

Blood Oxygen-Level Dependency (*BOLD*) Imaging (see Huettel et al. (2004) for a detailed look and Forster et al. (1998) for a short tutorial).

Since brain activity is associated with energy supply through oxygenated hemoglobin, BOLD imaging may be used to image brain activity. The ability for measuring functional activity truly differentiates MRI from CT imaging and has put MRI in competition with Nuclear Medicine Imaging techniques such as PET (described below). In order to produce an image from the BOLD effect, a sequence is chosen that is sensitive to local field inhomogeneities caused by the presence of deoxygenated hemoglobin. Gradient echo imaging with low angles presented in Sect. 2.2.3 is particularly sensitive to inhomogeneities and is often used in *functional MRI* (*fMRI*). It can be carried out using an EPI sequence by which an image is acquired with a single shot. Spin echo EPI can be used as well.

The BOLD effect is subtle. Reliable estimates for brain activity require experiments comprising a large number of images to cancel out noise effects. During image acquisition, a subject is asked to perform a task (e.g., listen to sound) for some time and then refrain from performing it. It is logged whether images are acquired during "action" or "no action" periods. The experiment is repeated several times. Potential correlation between intensity changes in the images and the "action"–"no action" sequence is computed for every voxel in a post-processing phase (see Fig. 2.31).

An interesting alternative to use the BOLD effect is resting-state BOLD imaging, where no task is given to the subject. The aim of this analysis is to detect potential correlations between different regions of the brain. It is hypothesized that different compartments are responsible for secondary processing of sensory input. Understanding connections between these regions is subject of current research, and it is assumed that correlating resting-state regions indicate potential connections. Resting-state analysis adds insight about functional correlation between brain regions to knowledge about the brain anatomy gained from fiber tracking in diffusion tensor images (see section below).

Fig. 2.31 A simple fMRI sequence consists of a sequence of images taking during a time in which a task is either on or off. Locations of which intensity changes correlate with the task design are then emphasized

Spatial resolution of a functional image is usually lower than that of anatomic images (128 × 128 voxels per slice). It may be registered with an anatomic image in order to relate function to anatomy.

Creating an fMRI image study is not easy. Severe distortions due to recording T_2* images for fMRI need to be corrected properly. Motion artifacts during the study may bias the correlation of task. Noise further reduces contrast and resolution of the fMRI images.

Several software packages exist to register anatomic and functional images, to carry out signal analysis, and to enable inter-subject studies. The most popular software, SPM, carries out statistic parametric mapping (as presented by Friston et al. (1995), Webpage http://www.fil.ion.ucl.ac.uk/spm/). Other commercial as well as non-commercial alternatives exist (e.g., BrainVoyager (Goebel et al. 1998), www.brainvoyager.com, FreeSurfer (Fischl et al. 1999), surfer.nmr.mgh.harvard.edu/). Despite of the complexity of processing the functional data, fMRI is popular and currently the only technique to measure cortical activity at a spatial resolution under 3 mm. Newer scanners such as a 7T scanner, now commercially available, carry the promise of pushing achievable resolution even further.

2.2.7 Perfusion Imaging

Gadolinium may be used to measure *perfusion* in MRI. Gadolinium not only reduces the T_1 time but also shortens the T_2 time (and with this the T_2* time). The change in T_2* is related to the amount of gadolinium. Since this is related to the amount of blood passing through the volume of tissue, Gadolinium-caused T_2* contrast predicts blood perfusion.

The main application of perfusion imaging is the depiction of parameters such as the *relative cerebral blood volume* (rCBV), relative cerebral blood flow (rCBF), and the mean transit time (MTT) in the brain (Gillard et al. 2004) (an example for the display of MTT is depicted in Fig. 2.32), perfusion imaging for tumor analysis in the female breast (Kuhl and Schild 2000), and perfusion imaging in cardiac imaging for rest and stress studies (Jahnke et al. 2007).

For observing the primary passage of the blood bolus containing the gadolinium through the tissue, a fast sequence is needed in order to produce sufficient temporal resolution. Echo planar imaging with a gradient echo sequence is one of the sequences that are fast enough and sensitive to T_2*.

An indication for performing cerebral perfusion imaging is the diagnosis of regions affected by a stroke. Diffusion imaging (see below) can demonstrate the central effect of a stroke on the brain, whereas perfusion imaging visualizes the larger “second ring” delineating blood flow and blood volume.

Fig. 2.32 Perfusion images are not interpreted as raw images. Diagnostically relevant measures such as the regional blood volume, regional blood flow, and mean transit time are computed and visualized. The two images here show the mean transit time of two slices of the brain of a normal volunteer

2.2.8 Diffusion Imaging

Molecules in a medium are in constant motion and will stray further away from an initial start location with time. The motion is neither directed nor deterministic. After a given time, it can only be said that—with certain likelihood—a molecule will be in some sphere around its initial position. The radius of the sphere depends on the time passed and on a diffusion coefficient. The latter is characteristic to the kind of molecule and the medium in which diffusion takes place. This is called *homogenous, isotropic diffusion* and requires molecules to be able to move unhindered. Diffusion will be lower if blocked by a cell boundary. It will be anisotropic if the shape of the cell restricts diffusion in some direction more than in others.

Measuring the diffusion coefficient of isotropic diffusion is called *diffusion imaging* (Gillard et al. 2004). A change of the value of the diffusion coefficient may indicate, for instance, the breakdown of cells in the brain after a stroke. *Diffusion tensor imaging* (DTI) relates to the measurement of a tensor that describes anisotropic diffusion. A diffusion tensor $\mathbf{D}(x, y, z)$ at some voxel (x, y, z) relates the flux $\mathbf{j}(x, y, z)$ to the gradient $\mathbf{D}$ of concentration $u(x, y)$ of some diffusing quantity u in the following way:

$$\mathbf{j}(x, y, z) = -\mathbf{D}(x, y, z) \times \nabla u(x, y, z). \tag{2.13}$$

The interesting part is the tensor itself. An eigenvector decomposition of $\mathbf{D}$ will produce an orthogonal system of diffusion directions together with the variance of diffusion (the eigenvalues) in each of the directions. Elongated cells such as a nerve fiber (an axiom) should produce maximum diffusion in one of these directions.

Fig. 2.33 Diffusion tensor of unbounded diffusion will have equal eigenvalues. If diffusion is bounded, eigenvalues will be lower and if the bounds are anisotropic, at least one eigenvalue will be lower than the others. Furthermore, the first eigenvector will point in the direction in which the bounded volume has its maximum extent

This is the direction of the fibers. Diffusion in the other two directions should be much lower (see Fig. 2.33).

Diffusion imaging by magnetic resonance employs a variant of a spin echo sequence. The purpose of the spin echo sequence is to cancel out effects of magnetic field inhomogeneities, given that the protons being imaged are static. Diffusion causes a slight loss of signal that cannot be recovered by the echo because diffusing protons move (as in phase-contrast imaging). A diffusion sequence measuring isotropic diffusion emphasizes this signal loss. The amount of diffusion enhancement depends on the strength and duration of the gradient impulses and on the time between the two impulses. The diffusion coefficient can be computed if these are known.

In the case of anisotropic diffusion, the response on the gradient impulse depends on the directions of the gradient field. Elements of the diffusion tensor are computed separately by applying a sequence of different gradient excitations.

The knowledge of local fiber direction from DTI is used for *fiber tracking* in the brain (Mori and van Zijl 2002), which—together with fMRI—may be used to infer configuration of regions of different brain functions (see Fig. 2.34). If two adjacent voxels belong to the same fiber bundle, they should exhibit linear diffusion and their main diffusion directions should be similar.

Reconstructing connections between different brain regions using fiber tracking is known as *MR tractography*. The result of a reconstruction of all connections between brain regions is part of the *connectome* of the human brain. The connectome is a map of the connections between different functional units of the human brain. Integrating information about the connectomes of different subjects in the attempt to analyze generic building blocks of humans is known as *connectionism*. It is presently an active research subject since MR diffusion imaging now produces images with a spatial resolution of about 1 mm (Urgubil et al. 2013; Koenig et al. 2013). In the Human Connectome Project, an interdisciplinary, multi-center research project, diffusion tensor imaging is combined with resting-

Fig. 2.34 Reconstructed fibers from diffusion MRI depicted on the right can be used with fMRI (depicted on the *left*) to infer configurations of functional regions in the brain (images from Magnetom Verio, with kind permission of Siemens Sector Healthcare, Erlangen, Germany)

state fMRI to infer functional connectivity and task-based MRI for inferring the parcellation of the brain into functional units in order to create the human connectome based on noninvasive MR imaging (see www.humanconnectomeproject.org). Figure 2.35 shows an example of the combined connections from 20 humans.

2.2.9 Image Analysis on Magnetic Resonance Images

There are a number of parallels between MRI and X-ray CT such as the reconstruction of a true 3d volume with good spatial resolution. Many analysis methods being developed for X-ray CT are in principle appropriate for analysis and measurement in MR as well. There are differences, however. The much better contrast for soft tissue enables separation of soft tissues by means of the reconstructed image intensity. However, there is no standardization of the image intensity with respect to the entity which has been measured. A T_2-weighted image may look different depending on the kind of sequence used, on the type of scanner, and on measurement parameters such as repetition time or echo time. And it certainly differs from the appearance of a T_1-weighted image. Computer-assisted analysis tools must take this into account either by incorporating scanning parameters into the method (which is seldom done), or by training appearance from sample images, or by requesting user information at the time of analysis.

Artifacts from shading in a study with known acquisition parameters hinder defining a mapping between tissue type and image brightness within this study. Automatic separation of gray matter and white matter in the brain, for instance, which have excellent contrast in some MR images, may need to deal with the different absolute brightness of the two materials at different locations.

Fig. 2.35 Combined fiber network from 20 subjects. Image adapted a color image published by user Andreashorn under Creative Commons license based on Horn et al. (2014). Fiber directions in the original contribution are color-coded and were transferred to gray level images for the printed version of this book

The wealth of different properties that can be imaged through magnetic resonance also increased interest in registration algorithms (discussed in Chap. 10). Sometimes, images showing different properties are already registered, e.g., if two echoes of a spin echo sequence are reconstructed. The first echo is dominated by proton density, and the second is T_2-weighted. Clustering and classification in multi-dimensional feature space may in such case provide further information about the entities being imaged. If registration is required, e.g., when combining functional images with a high-resolution T_1-weighted image of anatomy, rigid registration is usually insufficient. Registration has to account for deformation due to different effects of T_2* susceptibility in different imaging sequences.

Noise in MR imaging can be a problem if an analysis tool requires the mapping of a tissue type to a small range of brightness values, such as in direct volume rendering visualization or in threshold segmentation. Noise removal through the usual set of filters (to be discussed later in Chap. 4) helps, of course. It should be kept in mind, however, that most noise removal techniques operate under the hypothesis that the true image value is locally constant. Local homogeneity is not given if the size of image details reaches the limits of spatial resolution.

Accidental removal of such details by noise reduction is critical in MRI, since detection of small objects is often an objective of applying MRI because of its good soft tissue contrast.

2.3 Ultrasound

Sound waves will be reflected at boundaries between materials of different acoustic impedance. An ultrasound wave sent into the human body will be reflected at organ boundaries. The locus of reflection can be reconstructed if the speed of sound in the material through which the wave travels is known (Szabo 2004). For most soft tissues this speed is around 1500 m/s.

An ultrasound reflection signal is created using a transducer which acts as sender and receiver of ultrasound waves (Fig. 2.36 shows typical ultrasound equipment). Frequencies for diagnostic ultrasound range between 1 and 20 MHz. High-frequency waves attenuate faster than low-frequency waves and do not penetrate the body as good as low-frequency waves. High-frequency waves resolve smaller structures, however, since the size of a reflecting object has to be larger than the wavelength.

Fig. 2.36 Ultrasound equipment (Siemens Acuson, with kind permission of Siemens Sector Healthcare, Erlangen, Germany)

2.3.1 Ultrasound Imaging

An ultrasound *A-scan* sends a single wave with known direction into the body and records the amplitude of reflections as a function of travel time between sending and receiving the signal. It is a one-dimensional probe into the body showing tissue boundaries and other boundaries between regions with different acoustic impedance. *Ultrasound (US) images* (so-called *B-scans*, see Fig. 2.37) are created from a planar fan beam of differently rotated A-scans. Amplitudes are mapped to gray values for creating the image. They may also be acquired as 3d images with this fan beam rotating around a second axis perpendicular to the first axis of rotation.

Ultrasound imaging (also called *sonography*) happens in real time and is able to show motion of the organs being imaged. Ultrasound imaging of internal organs is only possible if they are not hidden by bone, since bone causes total reflection of incident sound waves. Organs to be imaged include liver, gallbladder, pancreas, kidneys, spleen, heart, and uterus. Heart imaging may also be carried out by putting the ultrasound device (a transducer sending and receiving sound waves) into the esophagus. Similarly, in transrectal ultrasound (TRUS) imaging, the device is placed into the rectum for imaging the prostate. Intravascular ultrasound (IVUS) is another technique for imaging organs from within. In this case, the ultrasound device is mounted on a catheter that is inserted into the artery for imaging and quantifying calcifications (plaque, see Fig. 2.38).

Doppler imaging is a specific technique using the Doppler effect for estimating the speed and direction of moving objects (such as blood) in the ultrasound image

Fig. 2.37 Ultrasound B-scan of the abdomen (image with kind permission of Siemens Sector Healthcare, Erlangen, Germany)

Fig. 2.38 Intravascular ultrasound image of the carotid artery clearly shows a difference between the elastic vascular wall and the plaque (although its automatic segmentation is not without challenges). Adaptation of a Creative Commons publication by user Ksheka in the English Wikipedia

(see Fig. 2.39). It is used for diagnosing effects of vessel blockages or changes of blood flow due to stenosis.

Ultrasound has also been used in image-guided therapy. If the sound waves are focused at some point, the energy will increase at this point (focused ultrasound—FUS, see also the website of the Focused Ultrasound Foundation, www.fusfoundation.org). The most obvious application of this effect is the thermal

Fig. 2.39 Doppler sonography uses the Doppler effect to depict blood velocity. The color-coded depiction of velocity differentiates between different flow directions and velocity (image with kind permission of Siemens Sector Healthcare, Erlangen, Germany)

ablation of tumors, where FUS is used to overheat tumor cells while leaving the remaining tissue intact. Temperature change induced by FUS can be visualized using MRI. Hence, planning commences with a scout MRI image after indicating the focus region by slight, non-lethal overheating. If region selection is verified, the tumor cells are destroyed by a short, focused ultrasound impulse. Besides tumor ablation, FUS may also be applied for inducing mechanical effects (e.g., moving stones in the gallbladder) or to stimulate growth. FUS has been applied in various body regions (e.g., brain, prostate, abdominal organs), but requires substantial support prior and during the intervention. Especially when FUS is required to penetrate bones (e.g., for thermal ablation in the brain), the increased energy absorption of bone structures has to be accounted for. If organs move (e.g., due to breathing or heartbeat) during intervention, this has to be modeled for proper guiding the operator by scout images.

A number of effects cause artifacts in an ultrasound image (see Fig. 2.40):

- Sound waves are attenuated just as electromagnetic waves in X-ray imaging.
- Absorption turns wave energy into heat.
- The wave may be scattered or refracted.
- Interference and a diverging wave cause further deterioration.

Absorption causes a decrease in amplitude with increasing depth. The decrease is exponential with an unknown absorption coefficient of the tissue. It is usually corrected by assuming constant absorption throughout the tissue.

Interference, scatter, and refraction of and between waves lead to the typical speckle artifacts in ultrasound images. It is a non-linear, tissue-dependent distortion of the signal.

Tissues and tissue boundaries that reflect or attenuate a high amount of the incoming sound energy produce an acoustic shadow behind the tissue. Materials

Fig. 2.40 Different effects influence the incident US wave of which only direct reflection is the wanted effect

that attenuate little of the incident energy lead to signal enhancement in tissues behind this material. This is for instance the case when imaged organs are behind a fluid-filled organ such as a filled bladder. The smaller absorption in the fluid contradicts the hypothetically assumed constant absorption and causes a higher than necessary absorption correction.

2.3.2 Image Analysis on Ultrasound Images

Ultrasound is noninvasive and inexpensive. Hence, it is widely used as a diagnostic tool. The artifacts mentioned in the previous section as well as the approximate nature of many of the underlying assumptions for imaging may adversely influence measurements in quantitative analysis:

- Localization in ultrasound imaging assumes that the speed of sound in the material is known. It is usually taken as a constant value of the average speed of sound in soft tissue and causes signal displacement depending on the deviation from this average.
- Refraction which has not been accounted for may lead to a further displacement error.
- Organ boundaries may cause mirror echoes or multiple echoes which appear as false boundaries in the image. Mirror echoes appear behind the true boundary. Multiple echoes appear between transducer and boundary.
- False, hyperbola-shaped boundaries may be caused by low-frequency lateral oscillation of the sound wave.
- Motion artifacts lead to wave-like distortions of boundaries.
- Acoustic shadowing may hide parts of tissues, and fluid-induced signal enhancement may lead to a position-dependent signal increase.
- Absorption decreases the signal-to-noise ratio with respect to the distance from the transducer.

Artifact removal through post-processing is only partially successful since their nonlinearity and non-stationarity defy common restoration techniques [although deconvolution in ultrasound images has been done, e.g., Jensen et al. (1993), Hokland and Kelly (1996)]. Using standard noise removal techniques to reduce speckle noise would seriously affect the image content since speckle is neither of high frequency nor stationary. Hence, image analysis often proceeds using the unaltered image and analysis methods have to account for this.

Tissue detection by means of their imaged intensity is impossible. However, the texture from reflection and speckle due to scatter and refraction within a tissue may be characteristic to certain tissue properties. This has been used for texture analysis of such tissues (Wagner et al. 1985; Wu et al. 1992). A variant of this technique is the analysis of the signal change of ultrasound over time (temporal enhanced ultrasound—teUS). This has been used to differentiate benign and malign tissues in the diagnosis of prostate cancer (Nahlawi et al. 2016).

Delineation of objects in ultrasound images has to account for inexact boundary depiction due to speckle, missing signal due to acoustic shadowing and for artifactual boundaries. Most boundary finding algorithms therefore include some knowledge about the expected course of a boundary.

Measurements of distance, size, and angles have to account for the fact that the slice direction of the scan is operator-directed and may vary between different images of the same patient.

2.4 Nuclear Imaging

Nuclear imaging measures the distribution of a radioactive tracer material and produces images of function in the human body. The tracer material is injected intravenously prior to image acquisition and will distribute through blood circulation. Distribution is indicative to perfusion of organs in the body. Examples for applications are measurements of brain activity, perfusion studies of the heart, diagnosis of inflammations due to arthritis and rheumatism, or the detection of tumor metastases due to increased blood circulation.

Images are created from measuring photons sent by the tracer material through the body. Spatial resolution in nuclear imaging is lower than for the procedures described above, since tracer concentration is very low as to not to interfere with the metabolism. Sensitivity of imaging techniques in nuclear medicine is high, since detectors are able to measure a signal from few photons. Major imaging techniques in nuclear medicine are as follows:

- *Scintigraphy*, which measures a projection of tracer distribution with a geometry similar to projection X-ray imaging.
- *SPECT* (*Single Photon Emission Computed Tomography*), which is a reconstruction from projections of tracer material producing a 3d material distribution.
- *PET* (*Positron Emission Tomography*), which is a tomographic technique as well but uses a different tracer materials that produce positrons. Radiation of positron–electron annihilation is measured and reconstructed.

2.4.1 Scintigraphy

For creating a scintigram, a molecule carrying the radioactive atom 99^{Tc} (Technetium-99) is applied. Photons emitted by tracer radiation are measured by a *gamma camera* [also written as *γ-camera* and sometimes called *Anger camera*, see Mettler and Guiberteau (2005)]. The camera consists of a collimator that restricts measurements of photons to those who hit the detector approximately at a 90° angle, a *scintillator crystal* that turns incident radiation into visible light, and *photomultipliers* for amplifying the signal.

The camera is usually mounted on a gantry that enables the camera to rotate (around various directions) around the patient. The *collimator* is a thick lead plate with cylindrical holes drilled whose axes are perpendicular to the scintillator crystal. Photons reaching the detector on a path perpendicular to the detector plane will reach the scintillator at a location that is given by the positioning of the detector hole through which it passes. Photons on a path with any other angle are reflected or attenuated by the lead collimator. If they reach the detector crystal through scattering, they have lost too much energy for being detected. Hence, the collimator causes the image to be an approximate parallel projection of photons from tracer material in the body onto the image (see Fig. 2.41).

The scintigram acquired by the gamma camera is a projection of activity weighted by the attenuation that photons experience on their path between emission and detection (see Fig. 2.42 for an example). Photons are attenuated by absorption and scatter. Absorption reduces the signal, while scatter reduces contrast and increases noise.

Collimator characteristics limit the spatial resolution and contrast of the scintigram (see Fig. 2.43). Photons reaching the detector do not exactly follow a path perpendicular to the detector plane. They originate from a cone-shaped volume whose size is determined by the diameter and the length of the cylindrical apertures in the collimator. Reducing the diameter and increasing the length will increase the spatial resolution, but it will also decrease the number of photons that reach the scintillator crystal hence reducing the signal.

2.4.2 Reconstruction Techniques for Tomography in Nuclear Imaging

Slice computation from multiple views of the gamma camera is done by applying tomographic reconstruction techniques. A sequence of images is acquired while the gamma camera rotates around the patient. If the axis, around which the camera rotates, is taken as y-axis of a device coordinate system, a slice may be reconstructed from lines with constant x-value of all projection images made by the camera.

Fig. 2.41 Photons reaching the scintillator to those on (approximately) parallel rays

Fig. 2.42 Bone scintigraphy (in this case a before- and after-treatment bone scintigraphy from Mado et al. (2006), published under Creative Commons license)

Fig. 2.43 Geometry of the collimator limits the spatial resolution of the scintigram. Resolution decreases with distance to the gamma camera

Slice reconstruction of nuclear images from projections for SPECT and PET is similar to reconstruction from X-ray projection but with an important difference which makes the FBP technique of X-ray CT less suitable for slice reconstruction although it is commonly used in commercial software. The aim of reconstruction is

to depict the spatial distribution $\mu(x, y)$ of radioactive tracer material. Reconstruction from projection would require projection of this distribution. The measurement, however, consists of projected, *attenuated* activity. Attenuation depends on the material between emission site and detector. This violates the assumption of the Radon transform, since the contribution of $\mu(x, y)$ to a line integral is dependent on the projection direction (see Fig. 2.44).

If attenuation is neglected, filtered backprojection may be used. However, it may lead to signal degradation which is unwanted in view of the already low signal. If attenuation were known for each voxel (x, y) in any projection direction (θ, s), it could be accounted for, provided that a suitable reconstruction algorithm exists.

Such technique is given by modeling reconstruction as a problem of estimating unknowns (the unknown densities $\mu(x, y)$) from a number of (linear) equations, which are the different projections. The oldest method based on this model, which originally was also used to reconstruct CT images, is the *algebraic reconstruction technique* [ART, Herman et al. (1978)]. For reconstructing the image, variables are rearranged so that all unknown densities μ are in a one-dimensional vector $\mathbf{x}$. Projections are arranged in another one-dimensional vector $\mathbf{y}$ (see Fig. 2.45). Now, the reconstruction requires finding an $\mathbf{x}$, for which

$$\mathbf{Mx} = \mathbf{y} \tag{2.14}$$

Elements m_{ij} of matrix $\mathbf{M}$ contain weights that represent how much of the volume in element x_i is measured in projection y_j ($m_{ij} = 0$, if the projection ray y_j does not pass x_i; otherwise, it is the relative volume of the voxel x_i, which is covered by a cylindrical ray y_j).

The matrix $\mathbf{M}$ can be huge, and it is not necessarily square. If an image with 256×256 elements shall be reconstructed from 180×350 projections, the matrix would have 65536×63000 elements. Even if it were a square matrix, it may be singular. The reconstruction is done numerically in an iterative fashion.

Fig. 2.44 Attenuation that depends on the attenuation coefficients along the path between emission site and detector causes different activity levels to be measured from the same site if the detection position is different

Fig. 2.45 Projection geometry for algebraic reconstruction. The value of m_{ij} is the ratio of the total area of pixel x_i that is covered by ray y_j

The reconstruction scheme is simple. Suppose that an estimate $\mathbf{x}^{(n)}$ of $\mathbf{x}$ is given. Then, an estimate $\mathbf{y}^{(n)} = \mathbf{M}\mathbf{x}^{(n)}$ of the projection vector can be computed. This estimate is compared with the true projection $\mathbf{y}$. The error $y_j - y_j^{(n)}$ for some projection y_j is then used to correct the equation $y_j = \sum_{j=0}^{N} m_{ij} x_j$ in the following way:

$$x_i^{(n+1)} = x_i^{(n)} + \left(y_j - y_j^{(n)}\right) \frac{m_{ij}}{\sum_{i=0}^{M} m_{ij}}. \tag{2.15}$$

This procedure is repeated for every equation in the system and—since corrections of $\mathbf{x}$ from later equation will have changed earlier corrections—it will be iterated until the residual error $\left\|\mathbf{y} - \mathbf{y}^{(n)}\right\|$ falls below some threshold. The first estimate $\mathbf{x}^{(0)}$ can be simply $\mathbf{x}^{(0)} = \mathbf{0}$, or it may be an estimate from some other reconstruction techniques (e.g., FBP).

Algebraic reconstruction takes much longer than FBP and is no longer applied for CT reconstruction. However, it is particularly easy to include effects such as attenuation into the problem. If α_{ij} is the attenuation of photons emitted at x_i and detected at y_j, all that is needed is to weight the m_{ij} (which may be interpreted as likelihood of an emitted photon at x_i to be detected at y_j) with α_{ij}.

Reconstruction in SPECT and in PET employs this strategy although a different reconstruction strategy is used. Photon detection is treated as a random process with known probability distribution. The method is called *maximum likelihood expectation maximization (MLEM) reconstruction*. For a detailed treatment, see Lange and Carson (1984) for an introductory treatment of common reconstruction techniques in nuclear imaging.

In order to model the problem, we assume that photon emission is a random process and that x_i is the mean number of photons emitted at location i which is detected at location y_j with probability m_{ij}. It results in the same equation as above, but with a slightly different meaning for $\mathbf{x}$ and $\mathbf{y}$. The number of photons detected at j depends on the probability that it is attenuated on its way, which in turn depends on the density, i.e., the mean number of photons $\bar{y}_j$ along the ray, which is

$$\bar{y}_j = \sum_{i=1}^{M} m_{ij} x_i. \tag{2.16}$$

It can be described by a Poisson process, whereby the probability of detecting y_j photons at j is

$$P(y_j) = \frac{\exp(-\bar{y}_j) \cdot (\bar{y}_j)^{y_j}}{y_j!}. \tag{2.17}$$

The measurements and thus the probabilities are independent. The conditional probability $P(\mathbf{y}|\mathbf{x})$ of observing measurement $\mathbf{y}$ given the tracer distribution $\mathbf{x}$ is the product

$$L(\mathbf{x}) = P(\mathbf{y}|\bar{\mathbf{x}}) = \prod_{j=1}^{N} P(y_j) = \prod_{j=1}^{N} \frac{\exp(-\bar{y}_j) \cdot (\bar{y}_j)^{y_j}}{y_j!}. \tag{2.18}$$

The maximum of L is computed by taking the derivative and setting it to zero. In order to simplify computation, the derivative is taken on the log-likelihood (maximizing the logarithm l of L also maximizes L):

$$l(\mathbf{x}) = \sum_{j=1}^{N} -\bar{y}_j + y_j \ln(\bar{y}_j) - \ln(y_j!). \tag{2.19}$$

Replacing $\bar{y}_j$ by the right-hand side of (2.16), we get

$$l(\mathbf{x}) = \sum_{j=1}^{N} \left(-\sum_{i=1}^{M} m_{ij} x_i + y_j \ln\left(\sum_{i=1}^{M} m_{ij} x_i \right) - \ln(y_j!) \right). \tag{2.20}$$

Taking the partial derivatives with respect to the x_i (using the chain rule and remembering that the derivative of $\ln x$ is $1/x$), we get

$$\frac{\partial l(\mathbf{x})}{\partial x_i} = -\sum_{i=1}^{M} m_{ij} + \sum_{j=1}^{N} \frac{y_j}{\sum_{i=1}^{M} m_{ij} \bar{x}_i} m_{ij} x_i = 0 \tag{2.21}$$

Multiplying both sides with x_i and resolving for x_i, we receive

$$x_i = -\frac{x_i}{\sum_{i=1}^{M} m_{ij}} \cdot \sum_{j=1}^{N} \frac{y_j}{\sum_{i=1}^{M} m_{ij} \bar{x}_i} m_{ij}, \tag{2.22}$$

which can be turned into the iteration step

$$x_i^{(n+1)} = -\frac{x_i^{(n)}}{\sum_{i=1}^{M} m_{ij}} \cdot \sum_{j=1}^{N} \frac{y_j}{\sum_{i=1}^{M} m_{ij}\bar{x}_i^{(n)}} m_{ij}. \tag{2.23}$$

This is the MLEM algorithm. Again, *attenuation correction* may be added by modifying the likelihoods m_{ij} with the attenuation weight. Other effects such as the loss of resolution with distance to the detector can be included in the weighting as well. Even scatter interaction can be included although this is not possible by simply adopting the weighting since interactions between sites have to be modeled (Shcherbinin et al. 2008).

A variant of the MLEM algorithm is *OSEM* (*ordered subset expectation maximization*), by which the order in which equations are updated is changed (Hudson and Larkin 1994). If projections are taken from, e.g., 96 angles, projection angles may be ordered in, say, 16 subsets with projections 1, 17, 33,… in the first subset, projections 2, 18, 34,… in the second subset, and so on. This reconstruction strategy has been shown to speed up convergence. The factor by which convergence increases is in the order of numbers of subsets.

Image reconstruction with EM increases noise in later iterations because noise characteristics are not accounted for. This means that it can be difficult to decide when to stop the algorithm. *MAP-EM (maximum* a posteriori *expectation maximization)* remedies this by including a priori knowledge about desired smoothness in the model. Different versions of MAP-EM exist. They have in common that smoothness is described by making constraints about activity differences in adjacent pixels. An algorithm by Green (1990), which is called the *OSL* (*one-step-late*) algorithm, for computing the MAP-EM estimate is similar to the EM algorithm. The term $\sum_{i=1}^{M} m_{ij}$ in Eq. (2.23) is replaced by

$$\sum_{i=1}^{M} m_{ij} + \beta \frac{\partial}{\partial x_i} U\left(\bar{x}_i^{(n)}\right), \tag{2.24}$$

where $\frac{\partial}{\partial x_i} U\left(\bar{x}_i^{(n)}\right)$ is an energy term enforcing smoothness, e.g.,

$$\frac{\partial}{\partial x_i} U\left(\bar{x}_i^{(n)}\right) = \sum_{k \in \mathrm{Nbs}(i)} w_{ik}\left(\bar{x}_i^{(n)} - \bar{x}_k^{(n)}\right) \tag{2.25}$$

and Nbs() refers to some neighborhood around a site i. The weighting term w_{ik} balances different types of neighboring pixels (e.g., neighbors at different distances).

2.4.3 Single Photon Emission Computed Tomography (SPECT)

SPECT uses projection images from the gamma camera in order to create an image of the radioactive tracer distribution (Mettler and Guiberteau 2005) (see Fig. 2.46 for an example of the image acquisition system used in cardiac SPECT). Images without attenuation correction can be reconstructed by FBP yielding a spatial resolution of approximately 3–6 mm side length of a pixel. Image sizes vary between 64 × 64 and 128 × 128 voxels per slice with 25–35 slices to be reconstructed. Using iterative reconstruction, attenuation correction and smoothness constraints may be included, leading to a better image quality at the expense of longer reconstruction times if compared to FBP (see Fig. 2.47). Attenuation maps can be generated from reconstruction of a transmission scan taken prior to imaging.

Acquisition of SPECT images can be carried out by a single rotating gamma camera. However, modern systems use *3-head cameras* for capturing three projections at a time. Acquisition time for a single projection is about 15–20 s which amounts, for a 3-head system, to total acquisition times between 5 and 10 min.

Scatter in SPECT decreases contrast and causes noise in the image. Due to the small number of photons measured, scatter may also cause artifacts, since scattering in dense materials with a high uptake of radioactive material may be falsely attributed to nearby regions. Scattered photons due to Compton scattering can be identified due to the energy loss of the scattered photon. Scattered photons may be

Fig. 2.46 Two-plane SPECT imaging system used in cardiac imaging (with kind permission from Siemens Sector Healthcare, Erlangen, Germany)

Fig. 2.47 SPECT image of the left ventricle. The two images are not smoothed in order to reveal their true spatial resolution. The *left image* was reconstructed without, and the image on the *right* with attenuation correction using the method of Celler et al. (2005) (with kind permission of Anna Celler, Department of Radiology, University of British Columbia, Vancouver)

removed by appropriate frequency filtering of the signal. Removal of scatter does reduce artifacts, but it cannot increase the signal-to-noise ratio.

Artifacts due to motion during image acquisition cause blurring of the data. Non-gated cardiac SPECT is not able to show the heart motion because of the long acquisition time, but it produces an average image over the complete heart cycle.

Major application fields for SPECT imaging are imaging of ventricular perfusion and ejection fraction of the heart, scans of lungs, kidneys, liver, and bone for tumor detection, and brain perfusion studies.

2.4.4 Positron Emission Tomography (PET)

PET uses positron emitters for producing the image (Mettler 2 and Guiberteau). Radioactive isotopes of atoms such as oxygen or fluoride emitting positrons are administered to the human body. If distributed in the body, emanating positrons annihilate if they meet an electron and produce two photons that are emitted in near-opposite direction. Photon energy is 511 keV. Events are measured by a *detector ring* and do not require collimators. An annihilation event is registered if two photons are detected at nearly the same time (within nanoseconds). The event is attributed to a location on a line connecting the two detection sites (see Fig. 2.48). This line is called *line of response* (*LOR*).

PET is an expensive technique if compared to SPECT because positron-emitting isotopes have a short half-life and need to be generated in a cyclotron in close neighborhood to the PET scanner. The scanning technique is demanding requiring a fixed detector ring that is capable of analyzing events according to synchronicity of measurement. Image quality of PET is better than SPECT. The number of

Fig. 2.48 Schematic view of a PET scanner (*left*) and resulting image of measured activity in the brain

attenuated photons decreases without collimation, the higher energy of the photons reduces attenuation loss in the body, and the near-parallelism of the path of the two protons focuses the ray better than the cylindrical aperture of the collimator in a gamma camera.

The spatial resolution of PET is in the range of 2–5 mm side length of a voxel. The signal-to-noise level is low due to the low number of counts. The true spatial resolution, i.e., the closest distance between two discernable objects, is often sacrificed in order to reduce noise by smoothing the data during or after reconstruction. PET, similar to SPECT, does not produce anatomic information. Metabolic function as imaged by PET is best evaluated if registered with some anatomic scan, e.g., from CT or MRI. Registration algorithms for registration of PET with anatomic imagery have been developed in the 1980s and 1990s (Pelizzari et al. 1989). Technical problems of registration at the resolution of PET images can be considered to be solved. Some of today's PET machines are combined with a multi-slice CT scanner for creating anatomic and functional images almost simultaneously and in close registration.

One of the many uses of functional imaging using PET is to observe brain activity using an oxygen isotope as a tracer. Being able to resolve cortex activity at resolutions of 5–10 mm stipulated much activity beginning in the late 1980s, because PET, for the first time, allowed observing activity at a resolution which is comparable to the size of major substructures—such as the visual cortex—in the brain (Senda et al. 2002). Presently, functional brain imaging through PET is challenged by fMRI, which is able to measure activity at an even higher spatial resolution than PET although being only an indirect indicator using the BOLD effect.

Further applications for PET imaging are the analysis of tumor metabolism in oncology or the tracing of labeled neuro-receptors in psychiatry.

2.4.5 Image Analysis on Nuclear Images

Images come as projection images as well as slice images sharing advantages and disadvantages of such kind of images, as explained in the section on computed tomography. Signal strength in the images depends on the amount of tracer given and the individual metabolism. Quantitative measurements based on intensity are usually a comparison of activity between two regions because the absolute intensity value at some location depends on external factors.

The image quality, i.e., resolution and signal-to-noise ratio, is poor because of the low number of photons contributing to an image and because of restrictions from image acquisition. This sometimes requires integrating quantitative measurements of relative activity over a larger region for arriving at reliable estimates.

Analysis often requires correlating the functional signal with anatomy. One should never be tempted to take the apparent agreement between activity distribution and expected anatomy as reason to derive anatomy from the functional signal. Anatomy and function may coincide, but a major purpose using a nuclear imaging technique is to identify and analyze regions where this is not the case. If analysis has to be carried out with respect to anatomy (e.g., "which part of the cardiac ventricle in a SPECT image is perfused normally?"), anatomical information has to be supplied by other means such as registering the data with anatomic images or an anatomic model of the organ to be imaged.

2.5 Other Imaging Techniques

Image modalities discussed so far make up the majority of images that are subjected to computer-assisted analysis. We will conclude this chapter with a brief discussion of some other methods that—although in part quite common in clinical practice—are not often subject to image analysis methods. Reasons are that at present most of them are diagnosed quite satisfactorily by inspection by a human expert.

2.5.1 Optical Coherence Tomography

The principle of optical coherence tomography (OCT) bears similarities to ultrasound imaging. A light wave (and not a sound wave) is sent into the probe, and reflections in the tissue are measured (see Fig. 2.49 for an example). Instead of the time delay (as in ultrasound), OCT uses the interference between the reflected wave and a wave reflected before the probe as measure to find the depth of reflection. The wave is generated by amplitude modification. The phase difference allows computing the depth for all depth differences smaller than the wavelength. Since waves are reflected at different positions, the reconstructed image carries information from various depths similar to the b-scan in ultrasound imaging. The spatial resolution is in the micrometer range and thus much higher than in ultrasound imaging.

Fig. 2.49 OCT of a muscle tumor (on the *right*). The original image was cropped. For the printed version, it has been mapped from color to gray values. Public domain. Image courtesy of Dr. Stephen Boppart, Biophotonics Imaging Laboratory, University of Illinois at Urbana-Champaign

Even resolutions in the submicrometer range have been reported (Frederici and Dubois 2015). The achievable depth depends on the attenuation and is specific to the material, but is generally in the range of 1–3 mm. A recent review on techniques and applications is done by Podoleanu (2014).

One of the main applications of OCT is ophthalmology, e.g., for analyzing the optic nerve and the retinal nerve fiber layer. A review on this is done by (Sakata et al. (2009). Another application field is dermatology. The review on using OCT of Wetzel (2001) states that OCT imaging enables assessment of inflammatory and bullous skin diseases and differential diagnosis for some tumors. There exist also catheter-mounted OCT systems to investigate vascular structures. This has been used, e.g., for diagnosing atherosclerotic plaques in coronary arteries (Jiang et al. 2005).

2.5.2 Photography

An example for diagnosis using photography is the depiction of vascular processes in retina photography (see Fig. 2.50). The retina is the only location in the human body where vessels are visible on the surface (Saine and Tyler 2002). Another application is the diagnosis and staging of skin tumors (Malvehy 2002) or burn scars (van Zuijlen et al. 2002). Photography in this field is sometimes replaced by dermoscopy (Bowling 2011) which produces microscopic images of the skin surface with a magnification factor of 1:10.

Fig. 2.50 Examples of retina photography (in the printed version, color of the original images has been mapped to gray levels): **a** retina of a normal and **b** degeneration of the macula (with kind permission of the National Eye Institute http://www.nei.nih.gov)

In treatment planning, photography may be used, e.g., for estimating the effects of plastic surgery. Digital photographs are usually high-quality and high-resolution color images.

A photographic image is a projection of some opaque surface, so that similar rules with respect to measurement and analysis apply as for X-ray projection images. If, for instance, distances on the retina surface shall be measured, the curvedness of the retina and the magnification factor due to the distance between camera and retina have to be considered. When using intensity or color in digital photographs for analysis, effects from external lighting, shading, or the camera position need to be accounted for.

2.5.3 Optical Microscopy

Optical microscopes in medical imaging are able to analyze living structures of sizes larger than approximately 200 nm. The magnification is generated through a system of lenses. Photosensitivity decreases with increased spatial resolution. Microscopic images are often used for diagnosis of pathology in tissue specimen on the cell level (see, e.g., Fig. 2.51). Images are color images or gray-level images with good contrast and signal-to-noise ratio. Microscopic images are used for cell counting, shape analysis of cells, and structural analysis of cells and cell distribution (Török and Kao 2008).

Cells are usually stained in order to better enhance certain structures. It is possible to stain cells with different dyes one after the other arriving at a stack of images showing different contrasts of the same cell culture. Staining may be toxic to the cells, however, which may alter the cell structure and which makes observation of the true dynamics of living cells difficult, e.g., if several images over time shall be taken.

Fig. 2.51 *Red*, *green,* and *blue* channels (mapped to grey levels) of a color-stained microscopic image of lung cells (image taken from Rogers (2004) published under Creative Commons license)

Confocal laser scanning microscopy (or simply confocal microscopy) is a variant of light microscopy that increases attainable spatial resolution and adds depth information to light microscopy. A pinhole is placed at the confocal plane of the lens and eliminates out-of-focus light. 3d images can be generated by changing the depth of the confocal plane.

A variant of cell microscopy is fluorescence microscopy. Instead of using reflection and absorption, the signal is generated by the fluorescence response of the living material to incident laser light. Since the response is at a different wavelength than the evoking laser light, influences from light reflection and absorption can be filtered so that the image only shows the fluorescence of the reacting material (Fig. 2.52 shows examples of fluorescence microscopy of the synaptic vesicles from drosophila larvae). Images are taken with different focus planes. In each image, the plane in focus is overlaid by blurred contributions from out-of-focus

Fig. 2.52 Dendritic spine (*left*) and fluorescence microscopy (*right*) of the synapses of drosophila larvae, see Bachmann et al. (2004). It is characteristic for fluorescence images that only fluorescent particles are visible while influence from the incident illumination is filtered out (the original images are in color showing different types of particles in different color channels). Hence, the dendrite is not visible in the image on the *right*

planes. Since the position of the focus plane and the simple PSF are known, a deblurred 3d image can be reconstructed by inverse filtering.

Creating fluorescence images over time may be subject to the *photo bleaching* artifact. Photo bleaching is the result of chemical damage to the fluorescent molecules from the electrons excited during fluorescence. It decreases the signal over time. This is particularly problematic in time-lapse fluorescence microscopy. Time-lapse microscopy refers to recording several frames of cell images over time and playing them at an enhanced speed to better convey cell dynamics. It may be applied to normal optical microscopy as well as to fluorescence microscopy and has been used, e.g., to measure cell cycle progression to understand drug treatment effects on cancer cells (Chen et al. 2006).

Fluorescence imaging in confocal laser scanning microscopy can be used to increase the resolution limit of optical microscopy. Diffraction limits the spatial resolution to the wavelength of light (about 200 nm). If the fluorescent structures are smaller than this limit, only a blurred version of it will be visible in the image. Near-field scanning optical microscopy increases the resolution by considering only the light that passes between a detector with very small aperture (e.g., 20 nm) and the probe, where the distance between the scanned probe and the detector is also small. The resolution is limited by the aperture of the detector.

If the original shape of the imaged object is simple enough to be approximated as a point source, it is just the point spread function of this point. The location of the point is the center of the blurred point. This has been used in subdiffraction confocal laser scanning microscopy using quantum dots to image function in living cells. Quantum dots are fluorescent molecules with sizes around 1 nm which can be attached to biologically active molecules to track them over time at a resolution of a few nanometers, as shown, e.g., by Sokoll et al. (2015) for analyzing vesicles in living nerve cells.

Microscopic images are good quality high- to very high-resolution images, which may suffer from blurring due to defocusing. Since color staining in cells is done on purpose, color-based segmentation and classification are often appropriate strategies for successful analysis. However, color processing often just determines a linear combination of colors that has the highest contrast using principal component analysis. All color vectors are then projected on the first principal component and subsequent processing is the same as for gray-level images.

Cell classification and detection of structural arrangement of cells may require quite elaborate segmentation techniques and a very good understanding of the often qualitative criteria to categorize cells.

2.5.4 Electron Microscopy

Electron microscopy uses the detection of electrons instead of light for image acquisition. Since their wavelength is much smaller than that of light, the achievable resolution is substantially higher (about 0.05 nm for transmission electron microscopy, approximately the size of a helium atom, as opposed to 200 nm for

most light microscopes). However, electron microscopy cannot be used to image the dynamics of living tissue since the probe needs to be immobilized (e.g., by freezing) to prepare it for the scanning procedure.

The first electron microscopes produced transmission images (*transmission electron microscopy*—TEM) where a beam of electrons is transmitted through an ultrathin specimen, interacting with the specimen as it passes through it. Similar to X-ray imaging for macroscopic structures, the image is created by detecting electrons on a CCD sensor or a photographic film with an appearance typical for transmission images.

Scanning electron microscopy (SEM) produces images of a sample by scanning it with a focused beam of electrons. The electrons interact with atoms in the sample, producing various signals that contain information about the sample's surface topography and composition. The electron beam is generally scanned in a raster scan pattern, and the beam's position is combined with the detected signal to produce an image. The achievable resolution is about 0.5–1 nm, which is lower than in TEM (Fig. 2.53 shows examples of TEM and SEM images). A similar technique than SEM is reflection electron microscopy that measures attributes of the reflection of the electron beam.

When analyzing the gray value of pixels from electron microscopy images, their origin has to be considered since an "intuitive" interpretation based on experience with images of macroscopic structures may be deceiving. In the treatment of X-ray radiation, we already saw that various types of scattering, absorption, and creation of secondary particles are the result of electrons interacting with material. This is even more true for electron microscopy where various types of scattering and backscattering may be used for imaging. This is an advantage as it allows adapting the image acquisition to probe for specific physical properties by using specific types of the signal for image generation. It does require detailed and in-depth expertise of the specific image acquisition, however, for developing computer-assisted analysis

Fig. 2.53 Transmission (*left*) and scanning (*right*) electron microscopies of cholera bacteriae. The resolution of the TEM is much higher than that of the SEM showing a single bacterium, whereas numerous bacteriae are depicted in the SEM. Images from the Dartmouth Electron Microscope Facility, Dartmouth College (public domain)

systems. A number of textbooks exist that describe the details for this kind of imaging technique, e.g., Reimer (2013) on TEM or Goldstein et al. (2012) on SEM.

2.5.5 EEG and MEG

For creating an *electroencephalogram* (*EEG*). a number of electrodes (16–25) are placed on the scalp to detect electrical impulses caused by brain activity. The impulses are amplified and represent an array of brain activity curves indicating function of the human brain. Brain activity happens in gray matter, which—for the most part—is close to the scalp. Hence, EEG provides a brain map of functional activity (Chambers and Sanei 2007).

Spatial resolution is poor, but temporal resolution of EEG is excellent (see Fig. 2.54 for an example of EEG time curves). Hence, EEG is a potential candidate to supplement methods with good spatial resolution and low temporal resolution of imaging (such as fMRI). An EEG measures neural activity directly, whereas other methods such as fMRI, SPECT or PET deduce brain activity from secondary signals (such as the blood oxygen level). Currently, the main activity in processing EEGs is signal analysis and pattern recognition of the temporal image. Image analysis plays a lesser role due to the poor spatial resolution of the signal.

A *magnetoencephalogram* (*MEG*) measures a similar effect than EEG through the magnetic field of neural brain activity (Hämäläinen et al. 1993; Papanicolaou 1995). MEG requires a system for recording very small magnetic fields (10^{-15} T)

Fig. 2.54 An EEG consists of a set of time signals that are acquired from different locations on the head surface. *Lines* of the original EEG were dilated in this picture for enhanced visibility (EEG entry of www.wikipedia.de from user "Der Lange," published under Creative Commons license)

and requires a special kind of recording magnet that is sensitive to such small fields (it is called *SQUID—superconducting quantum interference device*, see Fig. 2.55 for an example of the acquisition device). Measuring a magnetic field that is more than a billion times smaller than the earth magnetic field requires excellent shielding of the examination room. Compared to EEG, requirements for applying MEG are much higher. However, the achievable spatial resolution of about 2 mm is much better than in EEG. Temporal resolution is 1 ms and in the same range than EEG.

Compared to fMRI, MEG has several advantages. Since fMRI correlates signal change with an activity pattern over time, it is always acquired over a certain period of time. This is not the case for MEG, which means that not only it has a superior temporal resolution but also motion artifacts have not to be considered. Furthermore, since MEG measures absolute activity instead of activity relative to some paradigm, it may also be acquired in the absence of such paradigm (for instance while the subject is sleeping).

Spatially resolved MEG was mainly used for exact localization of a certain activity. Acquiring a surface map of the cortex is possible (Trujillo-Barreto et al. 2004), but the inverse problem (localizing the source from the measured signal) is severely underdetermined. Hence, regularizing the problem leads to a severe loss of spatial accuracy. The achievable spatial resolution is in the rage of several

Fig. 2.55 MEG acquisition, image courtesy of National Institute of Mental Health, National Institutes of Health, Department of Health and Human Services

millimeters, which is inferior to that of state-of-the-art fMRI images, which is in the submillimeter range. Currently, it is more of a complementary source of information, because of its excellent temporal resolution, than a competitor to fMRI.

2.6 Concluding Remarks

Image acquisition methods presented in this chapter can be grouped by acquisition technique into projective and non-projective methods, by the imaged parameters into physiological and functional imaging techniques, and by the physical effect that is measured. Each of the techniques produces information that is related to physiological and/or functional attributes in a known way. It can be assumed that the selection of an imaging technique is always intentional so that this relation can be exploited for subsequent analysis. Artifacts are often specific to an imaging technique so that this knowledge can and should be used when developing an analysis method.

The term "medical image" comprises a wide variety of different acquisition techniques and different signals to be acquired. Hence, understanding the semantics of the image will require a new effort every time that a new analysis task shall be solved with the help of a semiautomatically or automatically working algorithm. This is one of the major differences between processing medical images and other pictures. Another important aspect is that the depiction is usually not the projection of mostly opaque materials but either a projection of transparent materials (such as in X-ray imaging or in scintigraphy) or a full reconstruction of a 3d (or even 4d) scene. Recovery of distances or the treatment of partially hidden structures being in the focus of many computer vision algorithms is therefore not a major concern of the analysis of medical images. Main problems in medical image analysis are to deal with the incomplete differentiation of structures of interest, with artifacts from image acquisition and with the inaccessibility of the data to be analyzed.

2.7 Exercises

- Why is it impossible to measure distances in projection images?
- Why is angiography in cardiac imaging more popular than digital subtraction angiography?
- Explain the difference of the imaging technique between standard X-ray imaging and mammography.
- Explain the image acquisition process for digital radiography.
- What is meant by vignetting in radiographic imaging and what are the effects of this artifact?
- What is the partial volume effect and how does it affect the image?
- Why is it necessary to filter the projection data in filtered backprojection?

- Explain the backprojection procedure.
- What is the major difference between CT angiography and 3d reconstructed digital subtraction angiography?
- What is the purpose of filtering with a Hamming window filter when reconstructing CT images?
- What is meant by window and level in displaying CT images? Why is it needed?
- What is the typical spatial resolution of a body CT scan?
- Name three typical artifacts that occur in CT and MR imaging.
- What is the entity in the body that gives rise to the signal in MR imaging?
- Explain the meaning of the three components (ρ, t_1, t_2) that make up the MR signal.
- Explain the steps of slice selection, phase, and frequency encoding in MR imaging.
- What is the difference between t_2-relaxation and t_2^*-relaxation in MR imaging?
- How does the spin echo sequence in MR imaging avoid adverse effects from t_2^*-relaxation?
- Why is most MR image acquisition much slower than CT image acquisition?
- What is the principle of fast MR imaging techniques such as EPI imaging?
- Name and explain the differences between the different kinds of angiographic imaging in MRI. Which of the techniques can be used to recover velocity information?
- What is meant by ghosting in MR imaging and when does it occur?
- Which imaging technique uses Hounsfield units for normalization and what are the two reference values for normalization?
- What is the purpose of MR perfusion imaging? How is the perfusion information generated?
- What is the BOLD effect? How is it used in functional MRI?
- Why is it necessary to acquire a time sequence of images in functional MRI?
- What is measured in diffusion imaging? Why and how can this information be used for fiber tracking?
- What are A- and B-scans in ultrasound imaging?
- How is spatial information about the depth of a reflection generated in ultrasound imaging?
- Name some typical artifacts in ultrasound images.
- What are the similarities between UC imaging and OCT? What are the differences?
- What is the range of resolutions that can be achieved by OCT?
- Which imaging techniques are subsumed under the term “nuclear imaging”? Which of the techniques produces projection images?
- Explain the design of a gamma camera. What is the purpose of the collimator in this design?
- What are the adverse influences for reconstructing SPECT images?
- What are the reasons for SPECT having a low spatial resolution compared to, e.g., CT imaging?
- What are potential diagnostic questions that require a scintigraphy?

- What are advantages and disadvantage of MRI opposed to X-ray CT?
- Why is filtered backprojection not always appropriate to reconstruct SPECT images?
- Why is the spatial resolution of PET generally higher than the one of SPECT?
- What generates the signal in PET imaging? How does PET compare with functional MRI?
- What are applications for using photography in diagnosis?
- What are the major differences between light microscopy and raster electron microscopy?
- What are potential applications of using light microscopy?
- Why is it not possible to arbitrarily increase the spatial resolution of light microscopy?
- What are the two major kinds of electron microscopy? What are the differences regarding the image characteristics of these two kinds?
- Why is it not possible to image cell dynamics of living cells with electron microscopy?
- What is measured by an EEG? What is the spatial resolution of the signal?
- Name a potential application for acquiring an EEG?
- Which other imaging techniques produce a similar information than that generated by EEG?

References

Akpek S, Brunner T, Benndorf G, Strother C (2005) Three-dimensional imaging and cone beam volume CT in C-arm angiography with flat panel detector. Diagn Interv Radiol 11:10–13

Bacher K, Smeets P, Bonnarens K, De Hauwere A, Verstraete K, Thierens H (2003) Dose reduction in patients undergoing chest imaging: digital amorphous silicon flat-panel detector radiography versus conventional film-screen radiography and phosphor-based computed radiography. Am J Radiol 181:923–929

Bachmann A, Timmer M, Sierralta J, Pietrini G, Gundelfinger ED, Knust E, Thomas U (2004) Cell type-specific recruitment of Drosophila Lin-7 to distinct MAGUK-based protein complexes defines novel roles for Sdt and Dlg-S97. J Cell Sci 117(Pt 10):1899–1909

Bowling J (2011). Diagnostic dermoscopy: the illustrated guide. Wiley

Brown RW, Cheng YCN, Haacke EM, Thompson MR, Venkatesan R (2014). Magnetic resonance imaging: physical principles and sequence design. Wiley

Bushberg JT (1998) X-ray interaction. Radiographics 18(2):457–468

Bushberg JT, Seibert JA, Leidholdt EM (2002) The essential physics of medical imaging, 2nd edn. Lippincott Williams and Wilkins

Buzug TM (2008) Computed tomography—from photon statistics to modern cone-beam CT. Springer

Celler A, Dixon KL, Chang Z, Blinder S, Powe J, Harrop R (2005) Problems created in attenuation-corrected SPECT images by artefacts in attenuation maps: a simulation study. J Nucl Med 46(2):335–343

Chambers JA, Sanei S (2007) EEG signal processing. Wiley

Chan HP, Doi K, Galhotra S, Vyborny CJ, MacMahon H, Jokich HP (1987) Image feature analysis and computer-aided diagnosis in digital radiography. I. Automated detection of microcalcifications in mammography. Med Phys 14(4):538–548

Chen X, Zhou X, Wong ST (2006) Automated segmentation, classification, and tracking of cancer cell nuclei in time-lapse microscopy. IEEE Trans Biomed Eng 53(4):762–766

Cheng HD, Lui YM, Freimanis RI (1998) A novel approach to microcalcification detection using fuzzy logic technique. IEEE Trans Med Imaging 17(3):442–450

Dengler J, Behrens S, Desaga JF (1993) Segmentation of microcalcifications in mammograms. IEEE Trans Med Imaging 12(4):634–642

Dillon EH, van Leeuwen MS, Fernandez MA, Mali WP (1993) Spiral CT angiography. Am J Roentgenol 160(6):1273–1278

Dumoulin CL (1995) Phase contrast MR angiography techniques. Magn Reson Imaging Clin N Am 3(3):399–411

Federici A, Dubois A (2015) Full-field optical coherence microscopy with optimized ultrahigh spatial resolution. Opt Lett 40(22):5347–5350

Fink C, Hallscheidt PJ, Noeldge G, Kampschulte A, Radeleff B, Hosch WP, Kauffmann GW, Hansmann J (2002) Clinical comparative study with a large-area amorphous silicon flat-panel detector—image quality and visibility of anatomic structures on chest radiography. Am J Radiol 178:481–486

Fischl B, Sereno MI, Dale AM (1999) Cortical surface-based analysis. II. Inflation, flattening, and a surface-based coordinate system. Neuroimage 9:195–207

Forster BB, MacKay AL, Whittall KP, Kiehl KA, Smith AM, Hare RD, Liddle PF (1998) Functional magnetic resonance imaging: the basics of blood-oxygen-level dependent (BOLD) imaging. Can Assoc Radiol J 49(5):320–329

Friston KJ, Holmes AP, Worsley KJ, Poline JP, Frith CD, Frackowiak RSJ (1995) Statistical parametric maps in functional imaging: a general linear approach. Hum Brain Mapp 2(4): 189–210

Garmer M, Hennigs SP, Jäger HJ, Schrick F, van de Loo T, Jacobs A, Hanusch A, Christmann A, Mathias K (2000) Digital radiography versus conventional radiography in chest imaging—diagnostic performance of a large-area silicon flat-panel detector in a clinical CT-controlled study. Am J Radiol 174:75–80

Geluk RJ (1979) Transverse analogue tomography (TAT): a new method for cross-sectional imaging using X-rays. J Mod Opt 26(11):1367–1376

Ghersin E, Litmanovich D, Dragu R et al (2006) 16-MDCT coronary angiography versus invasive coronary angiography in acute chest pain syndrome: a blinded prospective study. Am J Roentgenol 186(1):177–184

Gillard JH, Waldman AD, Barker PB (2004) Clinical MR neuroimaging: diffusion, perfusion and spectroscopy. Cambridge University Press

Goebel R, Sefat DK, Muckli L, Hacker H, Singer W (1998) The constructive nature of vision: direct evidence from functional magnetic resonance imaging studies of apparent motion and motion imagery. Eur J Neurosci 10(5):1563–1573

Goldstein J, Newbury DE, Echlin P, Joy DC, Romig Jr AD, Lyman CE, Fiori C, Lifshin E (2012) Scanning electron microscopy and X-ray microanalysis: a text for biologists, materials scientists, and geologists. Springer

Green PJ (1990) Bayesian reconstructions from emission tomography data using a modified EM algorithm. IEEE Trans Med Imaging 9(1):84–93

Hämäläinen M, Hari R, Ilmoniemi RJ, Knuutila J, Lounasmaa OV (1993) Magnetoencephalography—theory, instrumentation, and applications to noninvasive studies of the working human brain. Rev Mod Phys 65(2):413–497

Herman GT, Lent A, Lutz PH (1978) Relaxation methods for image reconstruction. Commun ACM 21(2):152–158

Hokland JH, Kelly PA (1996) Markov models of specular and diffuse scattering in restoration of medical ultrasound images. IEEE Trans Ultrason Ferroelectr Freq Control 43(4):660–669

Horn A, Ostwald D, Reisert M, Blankenburg F (2014) The structural-functional connectome and the default mode network of the human brain. Neuroimage 102(Pt 1):142–151

Hudson HM, Larkin RS (1994) Accelerated image reconstruction using ordered subsets of projection data. IEEE Trans Med Imaging 13(4):601–609

Huettel SA, Song AW, McCarthy G (2004) Functional magnetic resonance imaging. Palgrave Macmillan

Jahnke C, Nagel E, Gebker R, Kokocinski T, Kelle S, Manka R, Fleck E, Paetsch I (2007) Prognostic value of cardiac magnetic resonance stress tests: adenosine stress perfusion and dobutamine stress wall motion imaging. Circulation 115:1769–1776

Jang IK, Tearney GJ, MacNeill B, Takano M, Moselewski F, Iftima N, Shishkov M, Houser S, Aretz HT, Halpern EF, Bouma BE (2005) In vivo characterization of coronary atherosclerotic plaque by use of optical coherence tomography. Circulation 111(12):1551–1555

Jensen JA, Mathorne J, Gravesen T, Stage B (1993) Deconvolution of in-vivo ultrasound B-mode images. Ultrason Imaging 15(2):122–133

Koenig KA, Sakaie KE, Lowe MJ, Lin J, Stone L, Bermel RA, Beall EB, Rao SM, Trapp BD, Phillips MD (2013) High spatial and angular resolution diffusion-weighted imaging reveals forniceal damage related to memory impairment. Magn Reson Imaging 31(5):695–699

Kuhl CK, Schild HH (2000) Dynamic image interpretation of MRI of the breast. J Magn Reson Imaging 12(6):965–974

Lange K, Carson R (1984) EM reconstruction algorithms for emission and transmission tomography. J Comput Assist Tomogr 8:306–316

Liang ZP, Lauterbur PC (2000) Principles of magnetic resonance imaging: a signal processing perspective. IEEE Press Series on Biomedical Engineering. Wiley

Mado K, Ishii Y, Mazaki T, Ushio M, Masuda H, Takayama T (2006) A case of bone metastasis of colon cancer that markedly responded to S-1/CPT-11 combination chemotherapy and became curable by resection. World J Surg Oncol 4:3

Malvehy J (2002) Follow-up of melanocytic skin lesions with digital total-body photography and digital dermoscopy: a two-step method. Clin Dermatol 20(3):297–304

McCollough CH (1997) X-ray production. Radiographics 17(4):967–984

McKetty MH (1998) X-ray attenuation. Radiographics 18(1):151–163

Meijering EHW, Niessen WJ, Viergever MA (1999) Retrospective motion correction in digital subtraction angiography: a review. IEEE Trans Med Imaging 18(1):2–21

Mettler M, Guiberteau MJ (2005) Essentials of nuclear medicine imaging, Saunders

Mori S, van Zijl PCM (2002) Fiber tracking: principles and strategies—a technical review. NMR Biomed 5(7–8):468–480

Nahlawi L, Imani F, Gaed M, Gomez JA, Moussa M, Gibson E, Fenster A, Ward AD, Abolmaesumi P, Shatkay H, Mousavi P (2016) Prostate cancer: improved tissue characterization by temporal modeling of radio-frequency ultrasound echo data. Intl Conf Medical Image Computing and Computer-Assisted Intervention. Springer LNCS 9900:644–652

Ning R, Chen B, Yu R, Conover D, Tang X, Ning Y (2000) Flat panel detector-based cone-beam volume CT angiography imaging: system evaluation. IEEE Trans Med Imaging 19(9):949–963

Papanicolaou AC (1995) An introduction to magnetoencephalography with some applications. Brain Cogn 27(3):331–352

Pelizzari CA, Chen GTY, Spelbring DR, Weichselbaum RR, Chen CT (1989) Accurate three-dimensional registration of CT, PET, and/or MR images of the brain. J Comp Ass Tomogr 13(1):20–26

Podoleanu AG (2014) Optical coherence tomography. Br J Radiol 12(4):285–306

Pooley RA, McKinney JM, Miller DA (2001) Digital fluoroscopy. Radiographics 21(2):512–534

Prince JL, Links J (2005) Medical imaging signals and systems. Prentice-Hall

Reimer L (2013) Transmission electron microscopy: physics of image formation and microanalysis, vol 36. Springer

Robb RA, Sinak LJ, Hoffman EA, Kinsey JH, Harris LD, Ritman EL (1982) Dynamic volume imaging of moving organs. J Med Syst 6(6):539–554

Rogers A (2004) T cells cause lung damage in emphysema. PLoS Med 1(1):e25
Saine PJ, Tyler ME (2002) Ophthalmic photography: retinal photography, angiography, and electronic imaging, 2nd edn. Butterworth-Heinemann
Sakata LM, DeLeon-Ortega J, Sakata V, Girkin CA (2009) Optical coherence tomography of the retina and optic nerve–a review. Clin Exp Ophthalmol 37(1):90–99
Salibi N, Brown MA (1998) Clinical MR spectroscopy: first principles. Wiley
Senda M, Kimura Y, Herscovitch P (2002) Brain imaging using PET. Academic Press
Shcherbinin S, Celler A, Belhocine T, Vanderwerf R, Driedger A (2008) Accuracy of quantitative reconstructions in SPECT/CT imaging. Phys Med Biol 53(17):4595–4604
Sokoll S, Prokazov Y, Hanses M, Biermann B, Toennies KD, Heine M (2015) Fast three-dimensional single-particle tracking in natural brain tissue. Biophys J 109(7):1463–1471
Szabo T (2004) Diagnostic ultrasound imaging: inside out. Academic Press Series in Biomedical Engineering
Török P, Kao FJ (2008) Optical imaging and microscopy. Techniques and advanced systems, Springer Series in Optical Sciences
Trujillo-Barreto NJ, Aubert-Vázquez E, Valdés-Sosa PA (2004) Bayesian model averaging in EEG/MEG imaging. Neuroimage 21(4):1300–1319
Uğurbil K, Xu J, Auerbach EJ, Moeller S, Vu AT, Duarte-Carvajalino JM et al (2013) Pushing spatial and temporal resolution for functional and diffusion MRI in the human connectome project. Neuroimage 80:80–104
van Zuijlen PPM, Angeles AP, Kreis RW, Bos KE, Middelkoop E (2002) Scar assessment tools: implications for current research. Plast Reconstr Surg 109(3):1108–1122
Wagner RF, Insana MF, Brown DG (1985) Unified approach to the detection and classification of speckle texture in diagnostic ultrasound. Intl Conf Speckle 556:146–152
Wang J, Blackburn TJ (2000) X-ray image intensifiers for fluoroscopy. Radiographics 20(5): 1471–1477
Welzel J (2001) Optical coherence tomography in dermatology: a review. Skin Res Technol 7(1):1–9
Wu CM, Chen YC, Hsieh KS (1992) Texture features for classification of ultrasonic liver images. IEEE Trans Med Imaging 11(2):141–152

Image Storage and Transfer

3

Abstract

Medical images are created, stored, accessed, and processed in the restricted environment of a hospital. The semantic of a medical image is driven by the particular purpose for creating it. This results in specific solutions for the archival of medical images. Transfer is different for medical images as well as driven by the technical specification of the various image acquisition systems and the particular requirements of users of these images. Archival and transfer of images is governed by two standards (HL7 and DICOM) that will be discussed in this chapter. The goal of the presentation is to enable the reader to understand the way images are stored and distributed in a hospital. It should further enable the reader to access images if some analysis method shall be applied to it and to decide how to implement such an analysis method in a clinical environment.

Concepts, Notions, and Definitions Introduced in this Chapter

- Information systems: the role of HIS, RIS, and PACS;
- Basic concepts of HL7;
- Introduction to the DICOM standard: information objects and services, establishing DICOM connectivity, the DICOM file format;
- Technical properties of medical images;
- Medical workstations.

It is likely that the first problem encountered when processing a digital medical image is how to access the image within the framework of information systems in the hospital. It will be difficult to set up a useful software module for post-processing images without some basic understanding as to how they are archived and transferred. Although experienced users in the hospital may be able to help, an image processing expert is generally assumed to be sufficiently knowledgeable with respect to databases and information systems.

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_3

Medical images differ from other images in several aspects. This has an impact on how they can be accessed. The most obvious of these aspects is that medical images receive their semantics only within the context in which they were created. Context information includes for instance demographic information about the person who was imaged, technical details about the image acquisition system, or the reason for the examination. Context information about medical images is much more extensive than for other images, where information about the size and quantization may suffice. For reasons of data integrity and data security, context needs to be firmly associated with the image.

A second important difference is that medical images are mappings of measurements of very different origins into a pictorial representation. Although photographs are used in diagnosis and treatment planning as well, medical imaging goes far beyond this, hence adding to the requirements for image meta-information. An unknown number of details about different image acquisition techniques have to be stored with the image. Storing meta-information describing the different parameters related to some image acquisition system efficiently leads to image formats that are different from conventional formats such as JPEG or PNG.

A third aspect which differentiates medical images from other pictures is that the use of medical images is highly constrained and regulated. Medical images contain sensitive personal information of which misuse must be prevented. Images have often been acquired for justifying quite invasive actions (decision in diagnosis or therapy with its consequences) that have to be executed with care and responsibility. A system where such images are created, stored, and accessed must be designed in a way that actively supports these goals. In consequence, images must not be accessed out of a framework that guarantees that the purpose of image access serves the intention for creating and keeping the image.

If a user intends to apply computer-based analysis techniques on a medical image, he or she should be aware of the points raised above. Hence, we will give an overview about storing and accessing medical images in a hospital. The chapter is not meant to be a comprehensive description about information systems in a hospital in general. We will mainly focus on topics that are directly related to images in the information system.

3.1 Information Systems in a Hospital

Images may be generated from patients either who are admitted to the hospital or who are sent in for outpatient examination. In either case, information is acquired that is necessary for the management of the actions performed while the patient is in the hospital. Hence, this information has to be accessible at various places (see Fig. 3.1). This includes information which is necessary:

Fig. 3.1 Schematic view of functions and locations in a hospital that deal with images or information related to images

- for administrating the patient's stay such as patient demographics and billing information;
- for performing the examination to which the images belong such as patient demographics, anamnesis, and reports; and
- for interpreting the images such as patient demographics, reports, imaging device information, and the images themselves.

The information is kept in different information systems in the departments of the hospital. Patient administration data is kept in the *hospital information system* (HIS). The HIS maps the internal structure of a hospital (its departments, clinics, and its access points for getting administrative information) into a database representation and governs access to this information. Ideally, a HIS would replace all paper files documenting a patient's stay. Any authorized person at any location in a hospital would have immediate access to this information. It requires that a modern hospital is equipped with the necessary network infrastructure, storage devices, and access points (such as PCs on the network). Such environment exists in many modern hospitals. Communication is often realized as a type of intranet solution in order to prevent unauthorized access. With increasing decentralization (outpatient care, outsourcing of administrative and technical support, teleradiology, etc.), secure and authorized access to this and other information systems from the outside world has become necessary and possible.

Data about radiological examinations is kept in a different system which is called *radiology information system* (RIS, see Fig. 3.2). The reason for this and other independent subsystems within the hospital is the complexity of the information structure and information flow. Modularization keeps the information where it is needed with well-defined interfaces governing the information exchange between

Fig. 3.2 The departmental radiology information system (*RIS*) and the picture archiving and communication system (*PACS*) are responsible for organizing, transferring, and archiving image data and image metadata

systems. It also helps to prevent unauthorized access. Department policy may prohibit, for instance, that an image created in a radiology department may be accessed without having a report from the radiology department associated with it.

Hospital information systems evolved from the need to create a paperless administration of in- and outpatients in a hospital. Development was initiated from the hospital administration and in many cases evolved from database systems for patient data administration. In early times, these systems were in-house adaptations of database systems or completely locally developed solutions. With information stored in such a system being more and more used for administrating and scheduling services performed in patient's stay, they have evolved into fully grown hospital information systems accessible by all departments of a hospital.

Radiology information systems are built to manage information about services that are connected with a radiological examination. This includes information about:

- examinations ordered,
- patient scheduling information,
- images created as part of an examination, and
- reporting.

RIS and HIS are similar in that they mainly cover administrative aspects of the patient's stay. If a patient is transferred to a radiology department, his or her demographic data is entered into the RIS together with all information about

services requested from the department. The RIS is then responsible to guide the patient's stay in the department. Examinations requested are scheduled, reports on examinations performed are kept and transferred to authorized requesters, services rendered are reported to the hospital information system, and so on.

The images themselves are not included in a RIS because of historical as well as practical reasons. Administrative data on the patient was digitized well before the majority of images became digital. Images were linked to the RIS by means of an identification code referring to pictures kept on film in some archive. Management of these pictures—e.g., ensuring that they actually were in the archive or, if not, that it was known where they could be found—was a non-digital and only partially computer-supported task.

The situation changed dramatically in the last three decades. In the 1970s, the majority of images created in a radiology department were still analog X-ray images. Thus, the preferred mode of access and display used analog means. With the advent of X-ray computed tomography in the 1970s and that of magnetic resonance imaging in the 1980s, images were generated whose primary representation was digital. Even X-ray images, still making up the bulk of images generated in a hospital, are more and more created digitally. In the foreseeable future, most, if not all, of the images created in the departments of a hospital will be digital.

It gave rise to the introduction of another image information system, which is called *picture archiving and communication system* (PACS, see Fig. 3.3). First, PACSs were introduced in the early 1980s. A PACS should archive and distribute

Fig. 3.3 In a PACS, the user has access to images from acquisition systems, workstations, and the archive. It usually requires a specific monitor configuration to render images, meta-information, and the interface to administer data access in a suitable way (with kind permission of Siemens Sector Healthcare, Erlangen, Germany)

pictures together with related information within radiology and to departments who ordered the images. Besides the images, this information includes patient demographics, information about reports, and technical information about the images such as the imaging device and its technical parameters.

RIS and PACS are the two information systems that a computer scientist is most likely to encounter when images in a hospital need to be accessed. The RIS contains information about the specific examination and associated previous examinations. This may also concern non-picture-based examinations. The brief description above already indicated that RIS and PACS share a lot of information (such as patient demographics). Hence, the RIS sometimes serves as primary access mechanism to archived images. It then governs access authorization and basic image manipulations such as retrieving or anonymizing images. The task may also be part of the PACS. In this case, access requires only communication with the PACS. In fact, the separation between RIS and PACS is much less obvious than that between HIS and RIS. For the latter, a clear hierarchy between hospital and departmental information system is reason for different systems, while for the former one, asynchronicity of development and type of data (text vs. images) is the main reason. In the future, PACS and RIS will probably fuse into a single departmental information system.

Existence of digital information systems throughout the hospital ensures that—in principle—authorized access to images is easy from any location in the hospital. Using the Internet, this can—again, in principle—easily be extended to virtually any place in the world. However, a necessary prerequisite is that these systems have export interfaces fitting to each other.

This is not self-evident. The different information systems were initially developed locally with local requirements in mind. With information systems more and more entering the clinical world, a need for standardized communication between information systems arose. It became clear that much of the information is needed multiple times (such as the patient demographics). Entering such information again every time a patient is admitted to a different department with its own departmental information system is tedious and error-prone.

Two different communication standards evolved: the HL7 messaging standard and DICOM. HL7 was developed for standardizing communication between clinical information systems, whereas DICOM was specifically targeted at standardizing communication of images. The former plays a role when connecting HIS, RIS, and PACS, and it will be briefly reviewed here. The latter mainly standardizes image communications between components of a PACS. As such, it will be a likely interface encountered by somebody wishing to access images stored in some picture archiving system. It will be reviewed in more detail in the next section.

HL7 is a standard developing organization which is accredited by the American National Standards Institute (ANSI). It is devoted to developing standards for communication in the healthcare business. The name HL7 stands for Health Level 7, and it refers to the application layer (the 7th layer) of the OSI (open systems interconnect, see Fig. 3.4) model.

Fig. 3.4 Layers of the OSI model

The OSI model is a framework for layered protocols in network communication. The application layer provides application-specific protocols for communication including information about authentication of communication partners, quality of service of the communication, and syntax of messages exchanged. HL7 is not a complete specification of such a protocol (such as DICOM, see below), but describes syntax and semantics of messages on this layer. For establishing communication, a layer 7 protocol is still to be agreed upon by which these messages are exchanged (e.g., FTP).

An HL7-conform system typically means that HL7-conform messages are created by an interface by which a system (archive, workstation, etc.) communicates with other systems. It generally does not mean that a system internally uses the HL7 standard to exchange information. This has consequences for accessing information in an HL7-conform system. Even if the module that uses the information is part of the system, it will need to be defined as an external partner communicating through the HL7 interface. If this is infeasible for reasons of performance, access through internal, often vendor-specific interfaces is needed.

HL7 messages are based on the HL7 *reference information model* (RIM), which is the main contribution of the HL7 organization. In a heroic undertaking, they succeeded in mapping most of the clinical information entities onto one big, generic, relational model. Within the RIM, six different stereotypes are specified, four of which refer to the nodes and two of which refer to the relations between nodes of the RIM:

- entity,
- act,
- role,
- participation,
- role relationship, and
- act relationship.

```
MSH|^~\&|EQUATORDX|Segments|ORDXTRAY:0.12.8 (Build 310)^L|HomeServer^1FFA8984-7166-4655-B195-7E
PID|1|4193101^^^Med[illegible]cts&7C3E3682-91F6-11D2-8F2C-444553540000&GUID^SR^HomeServer&1FFA89E
PV1|1|O||||||0191322W^MCINTYRE^ANDREW^^^^^^AUSHICPR^L^^^UPIN|0191322W^MCINTYRE^ANDREW^^^^^^AUSI
ORC|RE||7516A04B-77BC-4405-9138-034F84B4B132^HomeServer^1FFA8984-7166-4655-B195-7B4FFFD2F136^GU
OBR|1||7516A04B-77BC-4405-9138-034F84B4B132^HomeServer^1FFA8984-7166-4655-B195-7B4FFFD2F136^GUI
OBX|1|ST|MO_IMAGE_TITLE^^L|1|This is an ERCP Report||||||F
OBX|2|ED|MO_IMAGE^^L|1|^IM^JPEG^Base64^/9j/4AAQSkZJRgABAQAAAQABAAD/wgALCAIiAdwBAREA/9sAQwADAgIC
OBX|3|FT|MO_IMAGE_COMMENT^^L|1|This is a cholangiocarcinoma\.br\||||||F
OBX|4|FT|SIGNATURE_HEADER^^L|1|PKI Signed Message\.br\Patient: DEMO, Danny DOB:01.01.2005\.br\Re
OBX|5|ED|AUSETAV1^PKI Signature^L||AUSHICPKI^AP^Octet-stream^Base64^MIILUwYJKoZIhvcNAQcCoIILRD
```

Fields

Fig. 3.5 HL7 messages in versions 2.x are sequences of ASCII characters delimited by carriage return symbols

A specific HL7 message is an instance of a representation of this model. In HL7 version 2.x, it is a sequence of ASCII characters which consists of segments that in turn consist of composites (see Fig. 3.5). Segments are delimited by "carriage return" (\r) symbols; components are delimited by other delimiter symbols. Segments or components are either mandatory or optional accommodating a wide range of different messages while still conforming to the HL7 model. Segments may be repeated.

Choosing ASCII was intentional in order to make the messages readable by humans (although this may require some practice). The new HL7 version 3 standard replaced the somewhat outdated way of structuring the data. XML tags are used to tag information units, but the basic structure of the syntax remains the same (see Fig. 3.6).

The semantic of an HL7 message is determined by the first and mandatory identifying component. The current standard defines more than 100 different segments that may then follow. A specific implementation may still require a message part, which is not yet standardized. In order to let a message adhere to the standard nonetheless, the standardization committee allowed user-defined "Z" messages (named after the identification key). It is a potential source of incompatibility between two HL7-conform systems, as it for instance allows a vendor to hide non-mandatory information in a "Z" message. Other reasons of incompatibility are the use of different versions of the standard and various violations against the standard that are difficult to detect, such as missing fields (in versions 2.x, fields are identified by their position in the segments which means that missing, non-mandatory fields must be made identifiable by the delimiter symbol) or wrong data formats.

3.2 The DICOM Standard

When digital image acquisition systems such as X-ray computed tomography appeared in the clinics, the medium for data exchange and archival was film. Even though the data was created digitally, it was printed on film when sent away for reporting, for transferring it to the referring physician, or for archiving. Digital

```
<?xml version="1.0" encoding="ISO-8859-1"?>
 <message>
   <segment name="msh" >
    <field name="field_separator" len="1" >|</field>
    <field name="encoding_characters" len="4" >^~\&amp;</field>
    <field name="sending_application" len="180" >SENDAPP</field>
    <field name="sending_facility" len="180" >SENDFAC</field>
    <field name="receiving_application" len="180" >007</field>
    <field name="receiving_facility" len="180" >007</field>
    <field name="date_time_of_message" len="26" >200601212140</field>
    <field name="security" len="40" ></field>
    <field name="message_type" len="7" >ORU^R02</field>
    <field name="message_control_id" len="20" >HL7777</field>
    <field name="processing_id" len="3" >P</field>
    <field name="version_id" len="8" >2.3</field>
    <field name="sequence_number" len="15" >3434</field>
    <field name="continuation_pointer" len="180" ></field>
    <field name="accept_acknowledgement_type" len="2" ></field>
    <field name="application_acknowledgement_type" len="2" ></field>
    <field name="country_code" len="2" ></field>
    <field name="character_set" len="6" ></field>
    <field name="principal_language_of_message" len="60" ></field>
    </segment>
 <segment name="pid" ><field name="set_id" len="4" >1</field>
```

Fig. 3.6 In version 3 of the HL7 standard, XML tags are used to structure the message

images were stored digitally as well, but mainly for internal reasons related to the local infrastructure around the image acquisition system. There, digital images could be displayed on a local workstation, simple image analysis tools could be applied, or images could be selected that should be printed on film.

Initially, digital communication with other systems was not considered. Consequently, most vendors developed their own formats to store information deemed necessary for the purposes mentioned above. The decision about what information to store was dictated by the need to preserve semantically relevant information produced. This includes the following:

- Patient information: name and demographic information, identification in other information system in the hospital, etc.
- Examination information: referring clinic and/or physician, examination type, etc.
- Technical information: Many image acquisition systems require careful parameter selection controlling the acquisition process. A description of an X-ray CT, for instance, would include wavelength and amplitude of radiation, number and spacing of slices, spatial resolution within slices, reconstruction method, and reconstruction kernel.
- (Auxiliary) reporting information: Measurements and annotations being created during reporting, etc.
- The image or image sequence.

If sequences of images are generated, this information is often tagged to every single image of the sequence. The kind of information to be stored depends on the type, make, and version of the acquisition system.

With number and variety of digital imaging systems increasing, there was a growing need to manage images digitally. Preventing digital communication of images inhibited inclusion of them into the information system infrastructure of the hospital. This motivated the development of the DICOM 3.0 standard (and its predecessors ACR-NEMA 1.0 and 2.0). DICOM stands for *digital image communication in medicine* and is a full-fledged specification of the application layer of the OSI model. For communication of medical images, it replaces other file communication protocols such as FTP.

The DICOM standard and its two predecessors evolved from a joint effort of the American College of Radiology (ACR, http://www.acr.org) and the National Electrical Manufacturer's Association (NEMA, http://www.nema.org). Resources may be found on the DICOM home page at http://medical.nema.org.

Initially, DICOM was adopted only reluctantly by the industry. An open system standard supports multi-vendor environments in a hospital which automatically increases competition. It was also feared that a multi-purpose standard would be inferior to the highly specialized single-purpose interfaces of image acquisition systems. On the other hand, DICOM-conform machinery greatly simplifies connectivity between imaging components and the hospital information system. Today, it will be difficult to sell a major imaging device that does not conform to the standard.

DICOM is an interface standard similar to HL7. Internal communication is still vendor-specific. DICOM is designed to standardize communication between components such as imaging systems, printers, archives, and workstations. Formerly, these were often stand-alone system clusters communicating through specialized interfaces. This meant, for instance, that a CT image acquisition device of manufacturer X and an MRI device of manufacturer Y both required their own printer, archive, and workstation. Not only does this inhibit combined access to images from both devices, but also it results in an inefficient use of hardware resources.

At present, it seems that potential disadvantages from communication overhead generated by a multi-purpose standard are outweighed by the advantages of the open system architecture. It simplifies component sharing in a multi-vendor environment and ensures system compatibility when adding new components.

DICOM specifies a protocol for communicating objects between devices. Two different types of objects, *composite* and *normalized*, may be exchanged. An image is a typical composite object since it consists of several different entities, such as various text entities (e.g., the patient name), numerical entities (e.g., the number of pixels), and the image itself. Normalized objects, on the other hand, consist only of a single entity, such as a report. For each of the two types, the standard defines a number of services associated with it.

Services were first defined for composite objects, as DICOM was developed to support image communication. Four services, such as C-STORE, C-FIND, C-GET, and C-MOVE, are designed in order to exchange images. A service for updating an image is not provided in order to prevent intentional or unintentional alteration of an acquired image. In practice, this means that every change in an image, e.g., by some image enhancement procedure, will add a new image to a study.

Normalized services apply to single real-world entity. Four general services, such as N-CREATE, N_DELETE, N_SET, and N_GET, as well as the two specialized services such as N_ACTION and N_EVENT_NOTIFY were defined. The latter need to be specified within their specific context (e.g., printing a sheet of film followed by notifications of events that are caused by this request).

Service classes describe services that may be rendered to representations of information entities. Such an information entity could be, for instance, an MRI image with all its associated information (patient demographics, technical information about the acquisition, etc.). Classes of information entities in the DICOM world are templates that are called *information object description* (IOD, see Fig. 3.7).

An instance of an IOD is an *information object*. Information objects are uniquely identified. DICOM has adopted the ISO concept of the *unique identifier* (UID), which is a text string of numbers and periods with a unique root for each organization that is registered with ISO and various organizations that in turn register others in a hierarchical fashion.

For a given information object, the left-hand part of a UID is prespecified by the vendor. It may be decomposed into its hierarchical components. The right-hand part is generated when an information object is created. It is usually a combination of hospital, device, and patient identification together with a time stamp of the examination. This part is not intended to be parsed in order to regain the components. Its sole purpose is uniqueness. All relevant semantic information may be found in the information object itself.

Information objects are composed of sets of *modules*. Each module contains a specific set of data elements that is present or absent according to specific rules defined by the standard. Modularization supports the reuse of information

Fig. 3.7 An information object description (*IOD*) is defined for each information object. Different object classes may have different variants of the same modules. In the example above, for instance, the equipment module will contain different descriptions for CT and MR objects

specifications that are shared among different IODs. For example, a CT image information object contains, among others, a patient module, a general equipment module, a CT image module, and an image pixel module. An MR image information module would contain all of these, except the CT image module which would be replaced by an MR image module.

For a given IOD, several composite or normalized services may be useful. A service is defined as *DICOM message service element* (DIMSE) that invokes an operation or notification across the network. A DIMSE service group is a collection of DIMSEs applicable to an IOD (see Fig. 3.8). An information object description and the set of services operating on it are called a *service object pair* (SOP) class (see Fig. 3.8). An SOP class using composite services is called a *composite SOP class*, and an SOP class using normalized services is called a *normalized SOP class*. An instance of such a class is called a composite or normalized object (or, more correctly, a composite or normalized SOP instance). DICOM classes are static, which means that information entities (the data structures) and services (i.e., the methods) are provided as a template.

Communication follows the client–server paradigm. In the DICOM world, the server is called the *service class provider* (SCP) and a client is called the *service class user* (SCU). Communication between two components supporting the DICOM standard may be established in different ways between a server and a client. They have in common that messages are created as part of the external interface. A communication session between two components, which are called *application entities* (AE), is initiated by establishing a DICOM association (see Fig. 3.9). Basic understanding is established as to what information objects are to be exchanged and what services are to be invoked.

If communication is established such that full interoperability is ensured, components may exchange messages via DIMSEs. DICOM service classes support five different general application areas for communication as follows:

- network image management,
- network image interpretation management,
- network print management,

Fig. 3.8 A service–object pair combines an IOD (such as the CT object in this example) with services that are offered for this object. The services are called DICOM message service elements (DIMSE)

Fig. 3.9 Different application entities (*AE*) offer different services

- imaging procedure management, and
- off-line storage media management.

Of these five, the first application area is most relevant for accessing images in a hospital environment and thus will be explained in more detail. Network image management involves sending images between two devices. This could be, e.g., a scanner generating images and an external workstation or a digital image archive somewhere in the hospital network.

Two different kinds of communication are supported (see Fig. 3.10). In *push mode*, images are sent from one device to another device. This basic service would be appropriate, e.g., for sending images from a scanner to an archive. DICOM does not specify the timing behavior of the sending device, which means that the scanner could send images whenever the system is ready to do so. The receiver acts as listener that has to accept information at any time during which a DICOM association is established.

Pushing images may also be appropriate for communication to a workstation. This could be the case if the workstation serves as a reporting station where all images should be present. It could also be intended for sending images to a departmental workstation of a referring clinic.

Fig. 3.10 In push mode, images are transferred without request, while pull mode only transfers requested images

Often, however, the push service is not adequate. If a subset of images from an archive or from a scanner shall be processed on some workstation, it makes little sense to send all images over the network and to decide then which of those are actually to be used. Alternatively, the *pull mode* may be used for information exchange. It allows querying the sender first about its images. Selected images may then be pulled by the receiver.

The pull mode consists of two phases. First, the requesting component sends a query to the sending component. The prospective sender matches the keys of the query with images in its database and returns the number of information objects that match the keys together with their UIDs. The receiver then selects images from this list that he or she wishes to receive and requests them from the archive. Finally, these images are sent to the workstation.

The transfer of information is similar to that using other file transfer protocols such as FTP, however, with one important difference. Establishment of communication includes a common understanding between sender and receiver about basic properties of the kind of information exchanged such as keys to request in a pull service. Hence, sender and receiver know how to interpret such relevant information before a data transfer takes place. This enables organization and presentation of data according to such criteria. Instead of listing data by file names which, in case of the DICOM images, are their UIDs and difficult to interpret anyway, images can be presented in a more meaningful fashion. The result of a query may be structured by listing image data by patient name, patient ID, and study ID. Such organization by clinically relevant categories substantially enhances the use of transferred information.

DICOM network management services are specified in the service, query/retrieve, and storage commitment service classes. They are defined for composite objects only. The storage service class specifies the C-STORE service for pushing data to a client. The query/retrieve service class specifies C-MOVE, C-FIND, and C-GET services. With C-FIND, the sender is queried using a number of keys sent to him. With C-MOVE, a third party may invoke transferring images between two locations. A workstation, e.g., may query a scanner using the C-FIND service and then initiate a move of images from the scanner to an archive. C-GET is a service to retrieve images from a sending device.

For every information object, service classes have to be separately defined and supported. Standardization for almost all diagnostic imaging modalities exist, but all of the participating equipment has to conform to standards of all information objects that shall be shared among components in order to communicate. An archive, for instance, that supports the storage service class for MRI images will not be able to receive images from a CT scanner if it does not conform to the standard for CT images as well.

The storage commitment service class specifies the quality of storage provided by a device. Medical images are sensitive information, and it is to be assured that images are stored reliably for a given amount of time. Reliability means that images must not get lost in the time span to which a storage device has committed itself to. Such commitment was part of an analog archiving system as well. However, digital

image archiving required such commitment to be formalized so that it can be converted into a technical solution. Long-term storage devices such as long-term image archives commit to store an image permanently, while short-term storage devices such as a departmental image archive commit to a specified amount of time.

3.3 Establishing DICOM Connectivity

This book is not intended to serve as a reference to image networking in a hospital. However, a computer scientist working in medical image analysis in a hospital may encounter a situation where he or she has to participate actively in establishing communication for accessing images in the hospital network. Hence, we will give a short overview of aspects of transferring such images.

With DICOM as accepted standard, it would seem that the only question to ask is whether communication of a piece of equipment conforms to DICOM. Unfortunately, this is not true. DICOM needs to be highly modularized in order to accommodate the large variety of information objects to be exchanged in a hospital and the wide range of services requested. DICOM conformity without further specification just means that the piece of equipment conforms to communication according to at least one of the many parts of the standard. Somebody who has to access images from a DICOM-conform image acquisition device, a DICOM archive or a DICOM workstation, needs to know just to which part of the standard this device conforms. For this purpose, every piece of equipment that is said to conform to the DICOM standard has to provide a *DICOM conformance statement*.

A DICOM conformance statement consists of four parts: problem statement, application entity specifications, communication profiles, and specialization. In the *problem statement*, the vendor states the purpose of communication for his piece of equipment (e.g., a DICOM-conform CT scanner would state that its communication purpose is to transfer images to a storage device).

The major part of the conformance statement is the *application entity specification*. An application entity is a software module of the equipment implementing a specific application on the equipment. Several application entities may be implemented. An entity that is intended to communicate using the DICOM standard will offer one or more services for one or more information objects. Information objects and services to be provided are described here.

The *communication profile* then describes how services are communicated. Remember that DICOM specifies just the 7th layer of the ISO/OSI layered model. For successful communication, the lower layers have to be specified as well. The conformance statement will list and specify necessary details such as supported media access protocols (FDDI, Ethernet, etc.) or supported physical media (fiber, coaxial cable, etc.).

The *specialization* part of the conformance statement relates to extensions and vendor-specific specializations. DICOM is designed to be extensible in order to support new types of equipment or new uses of equipment without invalidating the

standard. This policy is implemented by allowing a certain degree of freedom in designing and filling templates of information objects. The subject will be discussed further when we explain the format of DICOM files, but it is of utmost importance that any of such extensions are described in detail in the specialization part of the conformance statement.

It should be possible to decide whether two pieces of DICOM-conform equipment can communicate based on the information in the conformance statements (see Fig. 3.11). If a CT system is SCP for C-STORE of CT information objects and some workstation is an SCU for the same service and information objects and if the two components communicate using the TCP/IP protocol and are connected by an Ethernet connection, then the scanner should be able to send images to the workstation.

However, while it is true that the conformance statement greatly simplifies installation of DICOM communication, it does not guarantee it. By comparing the conformance statements, it may tell the engineer whether communication is possible at all. If, for instance, a new CT scanner should send images to an already installed archive, this will only be possible if common intermediate layers 1–6 of the OSI model exist and if the conformance statements of the two devices both implement at least the storage service class for CT images. Communication may still be impossible or limited because of the several reasons as follows:

- Conformance has not been thoroughly tested and fails to establish in the specific environment.
- Information which is optional in the information object may be expected but is not present.
- Optional information may be present in specialized fields.
- Claimed conformance is erroneous.

Hence, compatibility of DICOM conformance statements does not ensure plug-and-play behavior of equipment.

If devices of two different vendors shall be connected, then it is sometimes difficult to establish who is violating the confirmations in the respective conformance statements. The search for necessary adjustments may be simpler if new equipment can be tested against some reference DICOM installations. Reference installations are open source software which do not only provide a standardized communication interface but also allow insight into the communication process

Fig. 3.11 It should be possible to decide whether two application entities are able to exchange information by comparing the desired service for the exchange with services offered by the AEs

itself. It may help an experienced user to determine and fix possible problems for DICOM-based communication.

Client and server software for various DICOM services may be found at the DICOM website of OFFIS (DICOM Test Kit DCMTK, http://dicom.offis.de) or at the Mallinckrodt Institute of Radiology (DICOM Test Node DTN, http://milan.wustl.edu/DICOM/). These two testing modules have been used for several years as vendor-free installations for various demonstration projects and are the base for many of the open source implementations of DICOM viewers.

3.4 The DICOM File Format

User-implemented image analysis methods are usually not directly integrated into software on a medical workstation communicating via DICOM with an archive or the image acquisition system. In most cases, images to be analyzed are accessed as files stored on some offline medium. Chapter 10 of the DICOM standard describes a file format standard for communication using offline media. A description is found on NEMA's DICOM pages (http://medical.nema.org).

Since the file format has to support the storage of many different types of information objects, it has to be highly variable. On the other hand, necessary effort for reading a file should be minimal. The two goals have been achieved by choosing a tagged format. Each tag relates to some data element (e.g., the patient name) of an information object. Its description can be found in a *data dictionary*.

The name of the DICOM file is its UID. Its content consists of a header of fixed length followed by a sequence of tagged data elements (see Fig. 3.12). The header contains the following:

- a 128 byte preamble, which is meant to support specific DICOM implementations. It does not have to follow the tagged structure of the DICOM file. If it is not used, it should be filled with zeroes.
- a 4-byte identification by the ASCII codes of the letters "D", "I", "C", and "M". Again, this information is not tagged.

Fig. 3.12 Structure of the DICOM file format. The preamble and the id are for DICOM-specific information only and do not contain any meta-information about the images. All the meta-information and the image itself are data elements

- a mandatory set of data elements containing meta-information about the file. Each of the data elements has to follow the tagged file format described below.

A data element consists of three parts: The *tag*, the *length* of the data element specified by the tag, and the *information* itself. The tag consists of a 4-byte *group number* and a 4-byte *element number*. The length of the tag is a 4-byte element following the tag and indicates the number of bytes reserved for the following information in the data field.

Data elements are characterized by their value representation (e.g., DS—decimal string, IS—integer string, and ST—short text), the maximum length of the data field, the set of acceptable symbols, the set of symbols that are explicitly not allowed, and the value multiplicity (a scalar has a multiplicity of "1", and a 3d coordinate vector has the multiplicity of "3"). Three different types of data elements exist:

- Type-1 data is mandatory and has to be filled with content,
- Type-2 data is mandatory but the data field may be empty, and
- Type-3 data is optional.

The data type may also be tagged by the letter *C*, such as 1*C* and 2*C*, which indicates a conditional type classification; e.g., it is type-1 only if specified conditions are fulfilled.

An information object must contain all mandatory data elements as specified in the standard for this type of object. The standard may also list a number of optional data elements. The information object may be encapsulated according to the meta-information in order to provide possibilities for encrypting and/or compressing the data. If it is neither encrypted nor compressed, it consists of a sequence of tagged data elements similar to the ones in the meta-information.

Tags are listed by their group and element number in the data dictionary. The data dictionary is part of the DICOM standard and can be found on NEMA's Web site. It also describes how data is represented (bytes may be interpreted as ASCII code or as various types of integers).

Odd group numbers are reserved for allowing vendor-specific adaptations or specializations. Hence, tags with odd group numbers will not appear in the data dictionary. These groups are also called *shadow groups*. Sometimes, shadow groups are used to represent non-mandatory data elements, which is an efficient way to hide this information. It is possible that the same tag of a shadow group may describe different information for files stemming from different systems, because group and element numbers of shadow groups do not need to be unique.

The semantic of tags from shadow groups must be explained in the conformance statement. Only then it will be possible to interpret all the information contained in transmitted or stored data. Even then, interpretation of the data may be difficult because files may be exchanged without sharing the conformance statement.

```
while not eof do begin
        int t_group = read_tag_group (file)
        int t_element = read_tag_element (file)
        int length = read_length (file)
        Interpret_Data (t_group, t_element, length, file)
endwhile

Interpret_Data()
        if t_group is even and exist(t_group, t_element) then
             interpret according to data dictionary
        else
            skip length bytes
```

Fig. 3.13 Tagged file format makes interpretation simple. Even if some data elements are unknown, others can be read, since unknown elements are simply skipped

Data elements of which the tag definition is missing may be skipped by a DICOM reader, since the length of information of a data element is part of the representation. This helps dealing with unknown shadow groups, and it enables reading data without a data dictionary (see Fig. 3.13 for a simple DICOM reader). For the latter, tag interpretation of only the most vital information for reading the data is hard-coded into the reader program. This is a solution of some software products that read the DICOM file format. The advantage is its simplicity although it is clearly insufficient for a DICOM reader within a PACS.

3.5 Technical Properties of Medical Images

Medical images differ from ordinary images taken by a camera in several aspects because they are visualized measurement values (as explained in Chap. 2). This has consequences on the technical properties of these images. With technical properties, we mean attributes that result from the image acquisition technique and that are independent from the semantics of an image.

Medical images may come with two, three, or four dimensions. 2d images may be slices of the human body such as an ultrasound image or a single CT slice. They may also be projections of a 3d scene such as an X-ray image or a scintigram. 3d images are volumes of the human body, such as a 3d sequence from computed tomography, or time sequences of 2d images. 4d images are 3d volumes acquired over time. The DICOM file format in which images are stored often treats 2d images as an information unit even if they are part of a 3d or 4d sequence. A 3d data set is then treated as a sequence, and a 4d data set is treated as a study of several sequences.

Images may be differentiated into projection and slice images. In projection images, image attributes are integrated along rays and projected on a single pixel. The type of projection is important, if measurements in such images shall be made. For a cone beam projection, such as in X-ray radiography, an unambiguous

relationship between distance in the image and distance in imaged space cannot be established.

Images may be acquired at several signal bands (e.g., in MRI imaging, see Chap. 2). If done so, these bands are stored separately. Two different bands may even be stored as separate studies. Interpretation is only possible if the image file information about the semantic of each image with respect to the signal can be retrieved.

Image sizes given in the DICOM tags relate to the number of pixels per column or row. The true physical size of a pixel or voxel (in mm or cm) is a mandatory data element which can be found if the tag identification is known (either from the data dictionary or hard-coded in the image reading program).

Pixel values of medical images are quantized. The quantization often differs from digital photographs. The range may exceed 256 values when the information acquired by the imaging device justifies a bigger quantization range. Pixel values are stored as integers. The storage reserved for a single pixel is one or two bytes depending on the quantization range. For two-byte-per-pixel images, often only 12 bits are used. Sometimes, negative values are stored (e.g., to represent Hounsfield units which begin at −1000).

The user should be aware that there is no guarantee that different vendor represent values in the same fashion. This is especially true if the file format for data exchange is non-standard. Hounsfield units, e.g., may be represented on a scale from −1000 to 3000 or—shifted—as unsigned integers on a scale from 0 to 4000.

Transferring digital image files between systems may involve changing between big-endian and little-endian notation of the two bytes. This refers to whether the first or the last byte is the most significant byte in a 2-byte word. It should be no problem if communication is standardized, but needs to be considered otherwise. It is easily recognized when looking at the images (see Fig. 3.14). Endianness may also be different for the bit order in a byte.

Fig. 3.14 MR image with correct endianness (scaled from its original range 0–4000 to 0–255) and the same image with byte endianity reversed

Sometimes, pixels or voxels of images presented for post-processing are reduced to single bytes. This is the case when images are transferred via formats that do not support a wider range. The mapping from two to one byte may have been created expertly and intentionally in order to highlight specific details in an image. It may support human capabilities of recognizing important aspects in an image, but computer-based methods operate on very different assumptions than human vision. The reduction while being supportive for human perception may be counterproductive to computer-based image interpretation.

3.6 Displays and Workstations

Traditionally, many medical images—X-ray images, computer tomograms, and MR images—are viewed as hard copies. Hard copies are readily transportable and readily readable. No specific equipment is needed to perceive a hard copy radiograph although a light box system is required for professional reading.

For reading an image by a radiologist, the film is placed in front of a light box (see Fig. 3.15). Reporting or other evaluation tasks are carried out there. Film has a number of advantages. Backlit light boxes allow for a good control of perceived contrast. This is further supported by the fact that most light boxes are large and are at a fixed position where lighting conditions can be controlled. A modern light box system can easily carry several hundreds of films. On such a system, films are mounted beforehand. Several of the films may be presented simultaneously. They can be selected at the push of a button from the set of films mounted.

Replacing analog data transfer and display by a PACS has advantages. Most notably, digital data can be easily copied and transferred to any location reachable on the network. Transfer may also be extended to long distances in short time (teleradiology). Replacing the analog archive by a digital archive also makes localizing images easier. It has been reported that accessibility increased from 70 to 98% after switching from analog to digital archives. Another advantage is that image enhancement techniques such as magnification or contrast enhancement may support image interpretation.

Replacing film viewing by displaying images on a screen should preserve most of the advantages of a film-based reporting. The first problem to be solved relates to image transfer. The capacity of the "muscle net," i.e., carrying films around, can be surprisingly high and may compare favorably to digital transfer capacities if the network infrastructure is poor.

Although this is becoming a problem of the past with improvement of the network infrastructure, perceived transfer capacity may still be unsatisfactory. Conventionally transported radiographs are "invisibly" transferred to places; i.e., the user does not perceive transport time as waiting time (although he or she may get upset about pictures not or not yet delivered). Once pictures are present, all of them are immediately at his/her disposal.

Fig. 3.15 Light box in a traditional reading room: The system can be loaded with several hundreds of films. Films to be read are loaded immediately using a handheld or a foot control (with kind permission of Planilux, Warstein, Germany)

In a digital environment, the user initiates the image transfer. The time span between request and completion of an image transfer is perceived as waiting time. Although an 80% decrease in preparation time has been reported after switching to digital reporting, this includes time spent by technicians and archive support staff. The radiologist may still feel that the waiting time increased with introduction of a digital system. With increased data sizes—a CT study from a modern multi-slice CT scanner may contain more than 1000 slices—the situation may worsen as even a fast network may need several minutes to transfer the data. Faster networks and intelligent solutions similar to those preparing a conventional reading session can help to reduce this problem.

Another problem is the limited spatial and contrast resolution of a monitor. The contrast range of today's TFT monitors is quite satisfactory, but monitor placement and influence from external lighting can reduce perceived image contrast.

Fig. 3.16 A reading room as part of a PACS displays images on monitors instead of light boxes. Active and passive components of the depicted system control the lighting situation (with kind permission of Planilux, Warstein, Germany)

A professional reading system will allow controlling such influence (see Fig. 3.16 for example).

The spatial resolution of a conventional monitor is about 2 megapixels, whereas a digital radiograph may contain up to 4096 × 4096 pixels (=16 megapixels). There are few monitors that are able to display such resolution. Hence, a professional reading station is much more expensive than a conventional TFT monitor.

Furthermore, a reading room usually contains several light boxes that are able to display several images at the same time. Using several monitors instead is only a partial replacement, since there will never be enough monitor space to display as many images as can be displayed on a big light box (see Fig. 3.16 for an example and compare it to Fig. 3.15). A suitable user interface for switching between views has to make up for this deficiency.

There are no legal standards for the display of digital radiography, but the American College of Radiology (ACR) has developed some recommendations. The ACR distinguishes between images used for diagnosis (interpretation) and those used for other purposes (clinical review, teaching, etc.). According to ACR, the image quality should be sufficient to satisfy the needs of the clinical circumstances if images are meant for display use only. Their recommendations for display and display software are as follows:

- The luminance of grayscale monitor should be greater than or equal to 50 foot-lamberts.
- Controlled lighting should enable eliminating reflections in the monitor.
- The ambient light level should be as low as feasible.
- A capability for selection of image sequences should be provided.
- The software should be capable of associating the patient with the study and demographic characterizations with the images.

The rendering software for images should

- be capable of window and level adjustment,
- be capable of pan and zoom (magnification) function,
- be capable of rotating or flipping the images, provided that correct labeling of patient orientation is preserved,
- be capable of calculating and displaying accurate linear measurements and pixel values,
- be capable of displaying prior image compression ratio, processing or cropping, and
- have available the matrix size, the bit depth, and the total number of images acquired in the study.

Requirements for display consoles which are not used for interpretation are less stringent. This relates for instance to workstations that are used for computer-assisted procedures. There are two places where such work can take place:

- Workstations as part of the image acquisition system, and
- Independent workstations in the hospital network.

Most vendors sell workstations and workstation software for post-processing image data. These workstations are a part of the imaging system and as such may communicate with image acquisition devices in some non-standard fashion. Standardized communication is not really necessary unless such workstation is meant to be connected to the network. Methods for image processing are predefined by the vendor of the imaging device (a typical user interface for such workstation software is depicted in Fig. 3.17). They are generally not open, i.e., they are neither adaptable nor extendable. Software being delivered with such a workstation falls in six different groups:

- Image display: retrieval and display of images, setting window and level of the mapping between image values and rendered intensities, and printing of images.
- Image enhancement: magnification, contrast enhancement, and noise filtering.
- Image annotation.
- Image analysis: measurements of distances and angles, volume estimation or volume measurements, and simple segmentation techniques.

Fig. 3.17 Example of a user interface of DICOM viewer software. The software is able to interpret the information in the DICOM header and organizes images accordingly. Viewing software includes display and simple tools such as setting window and level or annotating the image (with kind permission of Siemens Sector Healthcare, Erlangen, Germany)

- 3d imaging: slice view in cine mode, maximum intensity projection, and surface and volume rendering.
- Specialized interpretation or support to a specific planning task (e.g., surgical and radio-therapeutical).

Workstations at imaging devices support the radiologist in interpreting the images. If results of such post-processing of images are exported, it is carried out by using the DICOM standard or by creating hard copies. Analysis results may not be mandatory data elements of the DICOM standard. Tags for their description may belong to the shadow groups.

Workstations elsewhere in the network and independent of the image acquisition system may be used for post-processing image data as well. Such workstations were first offered by vendors selling archiving equipment. An image archive is intentionally meant to make images accessible, and it was an obvious strategy to provide processing methods as well.

Images in the archive are associated with records in the RIS. Access to them may be realized through the RIS or PACS interface. Communication with such workstation is carried out using the DICOM standard and/or the RIS interface. Such workstation allows radiologists or clinicians anywhere in the hospital direct access to images, provided that they have been granted access to the RIS/PACS.

Images may also be accessed from other workstations connected to the network. DICOM client software such as contained in the DICOM Test Kit (see Sect. 3.3) is needed that pulls images from the archive, the imaging device, or some other workstation. A more comfortable solution is a DICOM viewer which is able to query other modalities and display the results in some organized fashion. The functionality of a DICOM viewer should consist of the following:

- an implementation of a query to the database of a DICOM sender (C-FIND),
- an implementation of the C-GET in order to retrieve images,
- an interface for displaying the result of a query and for selecting queried images to be fetched,
- a rendering module to display retrieved images, and
- a storage module that stores retrieved images on the local machine.

Free DICOM viewers can be found on the Internet. A good reference for searching a viewer is http://www.idoimaging.com, already mentioned in Chap. 1. Some of the viewers only support accessing DICOM files on a local storage device and need to be combined with suitable client software to transfer images to local storage. The DICOM viewer may be used for selecting images for further processing (which would have to be developed and implemented by the user). The viewer may contain further routines that support interpretation or computer-based analysis.

Selecting an appropriate viewer can be difficult. Some viewers are very good in reading and exporting images, while others have many useful modules to display and manipulate images. Our current favorites are as follows:

- microDicom (http://www.microdicom.com/) is sufficient to display images and view tags. Since it has been implemented as a 64-bit version, it is able to handle large data sets (large 3d studies, 4d data sets).
- ezdicom (http://sourceforge.net/projects/ezdicom) is fast and appropriate for displaying images. Tags are not always interpreted correctly. DICOM studies cannot be read.
- dicomWorks (http://dicom.online.fr) reads DICOM studies and is able to export images into other formats.
- mevislab (http://www.mevislab.de) is—as detailed in Chap. 1—much more than a DICOM viewer, but also displays images, interprets DICOM tags, and exports images.
- OsiriX (http://www.osirix-viewer.com) is software for MAC OS.

The suggestions have to be taken with a grain of salt, however. The selection reflects the current interest of our group when dealing with DICOM images. Furthermore, open-source software may improve rapidly. Some research and experimentation with software listed on the Web site mentioned above will always be necessary to find the current best software for a specific purpose.

3.7 Compression of Medical Images

DICOM supports data compression. Images may be compressed lossless or lossy. Lossless compression such as entropy encoding or run-length encoding typically results in compression rates of 1:2–1:3. Lossy compression achieves much higher compression rates in the range of 1:10–1:30. Anybody who ever created a JPEG image knows that most images contain a lot of psycho-visual redundancy, since data reduction not automatically reduces perceived image quality.

Regarding medical images, lossy compression is a difficult issue. Laws in most countries require that medical images are to be saved for a given time. The intention of these laws is that decisions which have been based on these images can be reviewed based on the original data. If data has been compressed for saving storage space, its original content must be retrievable, which prohibits lossy compression.

However, there are other reasons for image compression. If images are transferred in a teleradiology network, its purpose is to provide the person at the receiving end with additional information. This person may be a senior radiologist at home who has to decide whether attendance in the hospital is required. It may also be a transferring radiologist at a small clinic who receives images from one of his patients taken and interpreted in a medical center. In such cases, image compression may be lossy if the partners in the data exchange have agreed that this loss does not constitute a loss of information for the kind of purpose.

There are no general rules as to the quality of such images, since image interpretation is to be carried out or confirmed by reading the original images. A recommendation of the American College of Radiology states, however, that clinically diagnostic image quality must not be reduced. While this does not rule out lossy compression, it is not clear whether it can be established statistically by means of readability studies or whether it requires expert decisions for every individual image.

Compression is part of the DICOM standard. Among others, the JPEG and JPEG2000 standards have been adopted by the DICOM standardization committee. The DICOM committee does not make a decision about the type of compression, whether lossless or lossy. Not all DICOM viewers, however, can read compressed DICOM images. It may also be an obstacle to a self-written DICOM image interpreter when the data has first to be uncompressed before it can be interpreted.

3.8 Concluding Remarks

There is a plethora of resources regarding the two standards HL7 and DICOM, their development and their role in the clinical environment. Good sources for getting a grip on this topic are the Journal of the American Medical Informatics Association, the Journal of Digital Imaging, Radiology, and the European Journal of Radiology.

A short description of the HL7 protocol is given by (Hammond 1993). The new version 3 of HL7 is described in (Beeler 1999). The complete description is available at the HL7 Web site http://www.hl7.org.

An introductory text about requirements and current state of picture archiving and communication systems is (Bick and Lenzen 1999). Tutorial publications on the DICOM standard directed toward non-experts in information systems are the papers of Bidgood et al. (1997) and Horii (1997). Honeyman et al. (1996) and Mulvaney (2002) describe the current state of integrating DICOM-based PACS with HL7-based RISs. A guide to DICOM-targeted toward practitioners is Pianykh (2011), now its second edition. The annual SPIE conference medical imaging has a conference devoted to PACS and medical informatics, where numerous reports about introducing DICOM-based PACS in hospitals have been published. The same is true for the InfoRAD section of the annual RSNA meetings. While challenges and problems of introducing a PACS may be more of interest for scientists dealing with clinical information systems, knowledge about transfer and representation of images is vital for developing and using image processing methods. Manufacturers usually do not publish their format specifications, but Clunie (2005) maintains an extensive Web site about medical image formats. Detailed descriptions of several proprietary data formats as well as about the DICOM file format can be found at his site.

3.9 Exercises

- What is the difference between a HIS and a RIS? Why is it useful to have several different information systems in a hospital?
- Which information systems communicate via the HL7 standard?
- Which aspect of the 7th level of the ISO/OSI standard is not included in the HL7 standard?
- What is the main difference between the versions 2.x and 3 of HL7?
- Name the potential reasons for failure to connect via the HL7 standard.
- What is the purpose of a PACS? What is the difference to a RIS?
- What is an information object in the DICOM standard?
- What are the names for client and server in DICOM terminology?
- Where is information to be found being necessary to connect a new piece of equipment to a system that supports DICOM?
- What are potential reasons for failure to establish connectivity under DICOM?

- When is it useful to use DICOM test nodes and what kind of information can be gained from such use?
- Why is the DICOM file format a tagged format?
- Explain the structure of a DICOM file. What are the general components and how is the structure of a tagged information entity in the file?
- What does the type classification into type-1, type-2, and type-3 data for the data elements mean?
- Which information is contained in the data dictionary? How does this support reading DICOM files?
- How can DICOM files be interpreted without the use of a data dictionary? What are the disadvantages of this?
- What is meant by shadow groups? Why are they necessary and what are the disadvantages of this?
- What is the range of values of a CT image?
- What are the differences when reading film as opposed to reading digital images?
- What are the requirements for displays of a medical workstation?
- List a number of image manipulation methods that can be expected to be offered in a medical workstation.
- What are reasons for avoiding or even prohibiting lossy compression of medical images? When could it be used nonetheless?

References

Beeler GW (1999) Taking HL7 to the next level. M.D. Comput: Comput Med Pract 16(3):21–24

Bick U, Lenzen H (1999) PACS: the silent revolution. Eur J Radiol 9(6):1152–1160

Bidgood WD, Horii SC, Prior FW, van Syckle DE (1997) Understanding and using DICOM, the data interchange standard for biomedical imaging. J Am Med Inform Assoc 4(3):199–212

Clunie DA (2005) Medical image format FAQ. http://www.dclunie.com/medical-image-faq/html/

Hammond WE (1993) Health level 7. A protocol for the interchange of healthcare data. Stud Health Technol Inform 6:144–148

Honeyman JC, Frost MM, Moser R, Huda W, Staab EV (1996) RIS requirements to support a PACS infrastructure. Med Imag Proc SPIE 2711:120–125

Horii SC (1997) Primer on computers and information technology. Part 4: a non-technical introduction to DICOM. Radiographics 17(5):1297–1309

Pianykh OS (2011) Digital imaging and communications in medicine (DICOM): a practical introduction and survival guide, 2nd edn. Springer

Mulvaney J (2002) The case for RIS/PACS integration. Radiol Manage 24(3):24–29

Image Enhancement

4

Abstract

Two reasons exist for applying an image enhancement technique. Enhancement can increase perceptibility of objects in an image to the human observer or it may be needed as a preprocessing step for subsequent automatic image analysis. Enhancement methods differ for the two purposes. An enhancement method requires a criterion by which its success can be judged. This will be a definition of image quality, since improving quality is the goal of such method. Various quality definitions will be presented and discussed. Different enhancement techniques will be presented covering methods for contrast enhancement, for the enhancement of edges, and for noise reduction. Edge-preserving smoothing based on different assumptions about noise and edges in images will be presented in detail including the presentation of several popular methods for edge-preserving smoothing.

Concepts, Notions, and Definitions Introduced in this Chapter

- Measures of image resolution and contrast;
- The modulation transfer function;
- Signal-to-noise ratio;
- Contrast and resolution enhancement;
- Noise reduction and edge enhancement by linear filtering;
- Edge-preserving smoothing by median filtering, diffusion filtering, various approximations of diffusion filtering and Bayesian image restoration.

A seemingly simple operation on digital images is to enhance the image or features in the image. The main purpose is to map a given image to another image such that the content to be depicted is now easier to recognize. In medical imaging, image enhancement essentially enhances contrast by reducing any artefacts or noise in the image or by emphasizing differences between objects.

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_4

The reason for enhancement is to make structures easier detectable by a human observer. It may also serve as some necessary preprocessing step for further automatic analysis. While in the latter case, success or failure may be found by experiment (e.g., does some image processing task perform better with or without the enhancement step?), deciding on the effectiveness of the former can be difficult because it requires modeling the human observer.

Quantitative measures of image quality will help as they describe aspects of an image that are relevant to human or computer analysis independent of an observer. Improvement of quality measures is then evidence for success of an enhancement procedure. Some of the measures can also be used to construct a method that only improves this measure.

It should be noted that image quality for visual inspection by human or computer vision depends on many influences that require knowledge of the specific detection task. These include image content, observation task, visualization quality, and performance of the observer. Measures that can be computed a priori make assumptions about these aspects or exclude them from the definition of quality.

4.1 Measures of Image Quality

4.1.1 Spatial Resolution

Spatial and contrast *resolution* already being used in Chap. 2 to characterize images determine the smallest structure that can be represented in a digital image. These two measures are easily computable and relevant to digital image processing. Structures can only be analyzed (delineated, measured etc.), if they appear in the image. Spatial resolution describes this directly since the sampling theorem states that no detail with frequency less than twice the sampling distance can be represented without aliasing. The contrast resolution is an indirect measure of perceptibility of structures. The number of intensity levels has an influence on the likelihood with which two neighboring structures with similar but not equal appearance will be represented by different intensities.

Subsequent enhancement steps such as noise removal (described below) can reduce intensity differences further, which may have consequences for the recognizability of structures if the original intensity difference was small.

Technical measures of resolution do not, however, relate directly to the ability of humans to recognize a structure. Human visual processing involves a combination of various enhancement techniques such as smoothing, edge enhancement, contrast enhancement at edges and the like. Hence, perceived resolution, as opposed to technical resolution, cannot easily be reduced to a single cause such as spatial resolution.

Perceived resolution may be measured experimentally by treating the human visual system as a black box system with images as input and recognized objects determining resolution as output. The same kind of measure is also used when loss

Fig. 4.1 A test pattern for determining perceived resolution in line pairs per millimeter (lpmm). The number of line pairs per millimeter increases from *left* to *right* while the contrast decreases from *top* to *bottom*. It can be seen that perceived resolution depends on the contrast in the image and also that this relationship is non-linear

of resolution by transfer of information through a technical system shall be documented (such as creating a radiograph from a scene). The quantity that is measured is called *line pairs per millimeter* (lpmm), which refers to the thinnest pair of parallel black and white lines that can be differentiated (either by a human observer or by an image analysis algorithm). A sequence of parallel pairs of black and white lines with decreasing line thickness is displayed (see Fig. 4.1). Apparent resolution is $1/x$ lpmm if the thickness of the thinnest line pair is x mm. The measure is proportional to the frequency, but it is a more figurative expression and easier to understand.

Perceived resolution by a human is not independent of contrast. An object stands out less against the background, if the intensity difference between object and background is low. Perceived resolution may sometimes be even higher than technical resolution because decreasing contrast may be interpreted as PVE due to the subvoxel size of the object. For instance, vessels that are smaller than the voxel size are visible in contrast-enhanced MR angiography because of the PVE.

4.1.2 Definition of Contrast

Determining contrast requires knowledge about what is an object and what is background. Since this is unknown prior to analysis, a number of measures for calculating image contrast exist that make implicit assumptions about image content. Examples for object-independent contrast measures are global contrast, global variance, entropy, and contrast from the co-occurrence matrix.

The *Michelson contrast* $C_{\text{Michelson}}$ (Peli 1990) measures the utilization of the luminance range. The smallest luminance is $l_{\min} = 0$ (i.e., no luminance). An upper boundary cannot be given, since luminance may be arbitrarily high. The smallest

Fig. 4.2 Two images have the same global contrast C_{global}, while their local rms contrast C_{rms} differs by a factor of three [C_{rms} = 0.006 for (**a**) and C_{rms} = 0.018 for (**b**)]

displayed luminance is $f_{\min}$ and the highest displayed luminance is $f_{\max}$. Michelson contrast is then:

$$C_{\text{Michelson}} = \frac{f_{\max} - f_{\min}}{f_{\max} + f_{\min}}. \tag{4.1}$$

Michelson contrast ranges from 0 to 1. It assumes that the highest luminance is the highest displayed intensity $f_{\max}$. It is 1.0 if the smallest intensity $f_{\min} = l_{\min} = 0$. *Global contrast* C_{global} extends this concept by also assuming a maximum (displayable) luminance $l_{\max}$ (e.g., $l_{\min} = 0$ and $l_{\max} = 255$):

$$C_{\text{global}} = \frac{f_{\max} - f_{\min}}{l_{\max} + l_{\min}}. \tag{4.2}$$

The measure assumes a simple image in which the number of foreground pixels approximately equals that of the background pixels. *Global contrast* ranges from 0 to 1. It is 1.0 if the full range of intensity values is used and less than 1.0 otherwise. It is a useful measure to quantify inefficient usage of the available intensity range. However, it does not account for the distribution of intensities in the image (see Fig. 4.2a). An image could be highly underexposed with most of the pixels having intensities below some low threshold but having just one pixel with value $I_{\max}$, possibly caused by an artefact. The image would be perfect according to global contrast.

A somewhat better approach for measuring global contrast is the *root-mean-square* (*rms*) *contrast* (see Fig. 4.2b). Given an image $f(x, y)$ with intensities $l(x, y)$, the expected value of i is

$$\bar{l} = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} l(i,j), \tag{4.3}$$

and the rms contrast is

$$C_{\mathrm{rms}}(f) = \sqrt{\frac{1}{MN-1}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left(l(i,j)-\bar{l}\right)^2}. \tag{4.4}$$

The measure takes all pixels into account instead of just the pixels with maximum and minimum intensity value.

C_{rms} does not differentiate well between different intensity distributions. Assuming $l_{\min} = 0$, an image containing just two intensity levels $0.75 \cdot l_{\max}$ and $0.25 \cdot l_{\max}$ would have approximately the same variance than another one that contains all intensities between 0 and $l_{\max}$ equally distributed. If both are images of the same scene, the latter may show more details than the former.

Entropy as contrast measure includes histogram characteristics into the measure. It is computed from the normalized histogram of image intensities. A histogram $H(l)$ of an image $l(x, y)$ gives the frequency of occurrence for each intensity value. A *normalized histogram* $H_{\mathrm{norm}}(l)$ is computed from $H(l)$ by

$$H_{\mathrm{norm}}(l) = \frac{H(l)}{\sum_{k=l_{\min}}^{l_{\max}} H(k)}. \tag{4.5}$$

It gives the probability of l to appear in an image. If $H_{\mathrm{norm}}(20) = 0.05$, the probability is 0.05 that the gray value of a randomly picked pixel is 20.

Entropy is computed from H_{norm}. It is being used in information theory for determining the average information capacity of a pixel. Entropy is a convenient measure for estimating compression rates for images for a type of lossless compression,[1] but it may also be interpreted as representing the amount of information contained in an image. Increased entropy of an image would indicate enhanced contrast.

Information capacity is defined assuming that information $I(l)$ of a pixel with intensity l is inversely proportional to the probability of its occurrence. Thus,

$$I(l) = \left(H_{\mathrm{norm}}(l)\right)^{-1}. \tag{4.6}$$

If information is stored in a binary number, then the number of required digits would be

$$\mathrm{SI}(l) = \log_2 \frac{1}{H_{\mathrm{norm}}(l)} = -\log_2 H_{\mathrm{norm}}(l). \tag{4.7}$$

Distribution of intensities of all pixels in the histogram can be used to compute the lowest number of digits needed to encode the image

[1]Compression rates of a lossless compression method using signal compression are bounded by the entropy. These methods compress pixels independent of their neighborhood. It can be shown that the maximum compression rate is the ratio of bits per pixel in the image to bits per pixel as predicted by entropy. Lossless compression using other kinds of methods such as run-length encoding may achieve higher compression rates.

Fig. 4.3 Histograms of the pictures in Fig. 4.2 and entropy-based contrast measure

$$\mathrm{SI}_{\mathrm{total}}(H) = -\sum_{k=I_{\min}}^{I_{\max}} H_{\mathrm{norm}}(k)\log_2 H_{\mathrm{norm}}(k). \tag{4.8}$$

The entropy C_{entropy} is then the average signal length needed (see Fig. 4.3 for an example):

$$C_{\mathrm{entropy}}(H) = -\frac{1}{MN}\sum_{k=I_{\min}}^{I_{\max}} H_{\mathrm{norm}}(k)\log_2 H_{\mathrm{norm}}(k), \tag{4.9}$$

where MN is the number of pixels in f.

Entropy still does not account for the fact that contrast should measure intensity differences between some foreground and background object. This can be computed using a *gray-level co-occurrence matrix* (*GLCM*, Haralick 1979). Co-occurrence calculates the normalized rates of co-occurring intensity values in a given neighborhood. The neighborhood is defined by the distance and direction between the two pixels. Hence, co-occurrence $C_{\alpha,d}$ is a two-dimensional function of intensities l_1 and l_2. $C_{\alpha,d}(l_1, l_2)$ is the probability with which pixels with intensities l_1 and l_2 occur such that pixel l_1 and l_2 are d units apart at an angle of α with the x-axis. Co-occurrence matrices can be computed with different distances and different directions representing intensity changes between structures at different angles and with different sharpness at the edge.

For measuring contrast in a given image, co-occurrence is computed for a fixed distance (e.g., $d = 1$ pixel) and for arbitrary angles. $C_d(l_1, l_2)$ is then the co-occurrence of pixels with gray levels l_1 and l_2 at distance d with arbitrary angle (Fig. 4.4 shows an example for the two images in Fig. 4.2). For $d = 1$, this would be the four pixels of the 4-neighbourhood. Contrast C_{GLCM} is then defined as

$$C_{\mathrm{GLCM}}(i,j) = \frac{1}{I_{\max}^2}\sum_{i=0}^{I_{\max}}\sum_{j=0}^{I_{\max}} C_d(i,j)\left(1+(i-j)^2\right) - 1. \tag{4.10}$$

Contrast thus weights co-occurrences of two intensities by the difference between the two. Higher differences indicating edges receive higher weights.

Fig. 4.4 Co-occurrence matrices for the two pictures in Fig. 4.2 and entropy-based contrast measure C_{GLCM}

4.1.3 The Modulation Transfer Function

To a large extent, improvement of contrast and resolution are constrained by technical parameters of an imaging modality. Artefacts, noise, and approximations in the reconstruction may reduce contrast in the original measurement due to reconstruction. The degradation that an image suffers through reconstruction, transfer or any other process that changes contrast, is described by the *modulation transfer function* (MTF). An MTF is a function of frequency (or of resolution in lpmm), which describes the extent of signal loss (damping) with respect to frequency. It is used, for instance, to describe the loss of contrast when creating an X-ray image. It can also be used to describe the discrimination performance of human vision with respect to varying contrast. Curiously enough, the human observer MTF does not exhibit monotone decrease with frequency but peaks at some low frequency. In other words, contrast at low frequency needs to be higher for recognizing an object than in midrange. This justifies edge sharpening (described below) for enhancing perceived resolution.

4.1.4 Signal-to-Noise Ratio (SNR)

Noise in an image is another factor limiting the perceptibility of objects in an image. So far, all information in an image is assumed to be useful. Noise is an unwanted, image-corrupting influence. Noise $n(i, j)$ in an image is usually described as a random fluctuation of intensities with zero mean. If noise is assumed to be normally distributed with zero mean, variance $\sigma^2(n)$ or standard deviation $\sigma(n)$ characterizes the noise level. Object detection depends on the ratio of object-background contrast to noise variance. The former is called difference signal between object and background, and it is related to noise in the *signal-to-noise ratio* (*SNR*):

$$\mathrm{SNR}(f) = S(f)/\sigma(n). \tag{4.11}$$

Various specifications exist of what the difference signal should be leading to different definitions for measuring the signal-to-noise ratio in an image. In the simplest case, signal $S(f)$ is defined to be the largest intensity $f_{\max}$ (*peak SNR*) or the average intensity $E(f)$. The two measures take the maximum or average image contrast between all objects in the image into account. They can be computed without analysis as to which objects should be recognized in the image.

A common measure, which is measured in dB, is given by the logarithm of the ratio of signal and noise variance

$$\mathrm{SNR}_{\mathrm{dB}} = 10 \cdot \log_{10}\left[\frac{1}{MN}\frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left(f(i,j)-\bar{f}\right)^2}{\sigma^2}\right]. \tag{4.12}$$

Increase in SNR may indicate enhancement of the image. If SNR shall be used as an absolute quantity to determine the perceptibility of objects, the difference signal has to reflect the difference between object and background intensity.

None of the quantities listed covers the dependency of recognizing objects based on their size, shape, sharpness of edges, and texture as it would require prior detection of the object in the image. This should be kept in mind. Two images with equal contrast or noise characteristics may still be perceived as being of different quality (see Fig. 4.5).

The dependency of recognizability on the object shape exists sometimes for a computer algorithm as well. It is possible, for instance, to recognize an object with

Fig. 4.5 Four images have the same noise level, noise characteristics, and contrast. Object-dependent features such as the size of the object or the sharpness of boundaries still cause differences in the perceptibility of depicted objects

SNR < 1, i.e., where the noise exceeds the signal if the object is large enough. Human vision and also some computer-based enhancement techniques are able to reduce noise in such a case while leaving the signal more or less intact. The potential for such recognition ability cannot be measured or deduced from the image unless the kind of structures to be recognized is known.

4.2 Image Enhancement Techniques

Originally, image enhancement methods were meant to enhance the perceptibility of information. Hence, contrast or edge enhancement improves the image for inspection by a human observer. This should be kept in mind when considering an enhancement procedure. Although most of the methods are also valuable and necessary preprocessing steps for automatic analysis, some of them—such as contrast enhancement—are not, since they do not improve the relation between information and artefacts in the image.

4.2.1 Contrast Enhancement

Some of the contrast enhancement techniques can be directly related to contrast measures described in the previous section. The simplest method increases global contrast. If the range of possible intensity values $I_{\min}$ to $I_{\max}$ exceeds the range of intensities $f_{\min}$ to $f_{\max}$ in an image f, *linear contrast enhancement* is carried out creating new values g for every pixel by

$$g(f) = (f - f_{\min})\frac{I_{\max} - I_{\min}}{f_{\max} - f_{\min}} + I_{\min}. \tag{4.13}$$

The function to map f on g is called *transfer function*. Contrast enhancement in an arbitrary intensity window $w_{\min}$ to $w_{\max}$ with $I_{\min} < w_{\min} < w_{\max} < I_{\max}$ can be achieved with a similar transfer function. As there may be pixels with values f outside the window, Eq. (4.13) is changed to (see Fig. 4.6):

$$g(i,j) = \begin{cases} I_{\min}, & \text{if} \quad f(i,j) < w_{\min}, \\ (f(i,j) - w_{\min})\frac{I_{\max}}{w_{\max} - w_{\min}} + I_{\min}, & \text{if} \quad w_{\min} \le f(i,j) \le w_{\max}, \\ I_{\max}, & \text{if} \quad f(i,j) > w_{\max}. \end{cases} \tag{4.14}$$

Output intensity of structures with input intensities outside of $[w_{\min}, w_{\max}]$ is either $I_{\min}$ or $I_{\max}$ so that they are no longer recognizable. Contrast is improved for all other structures. This kind of enhancement is routinely used for mapping a 16-bit-per-pixel intensity image (such as CT or MRI) onto an 8-bit range. It is called *windowing* with window size $w_{\max} - w_{\min}$ and level $(w_{\max} + w_{\min})/2$.

Fig. 4.6 Linear contrast enhancement in a window (50, 120) for enhancing soft tissue differences in an MR image. The enhancement comes at the cost of reducing contrast in regions outside of the window (such as the water in the eyeballs)

Entropy is enhanced using *histogram equalization*. Assuming that intensities of some image f with histogram $H(f)$ are defined on a continuous domain, histogram equalization maximizes entropy by creating a new image g with a constant histogram. The transfer function is

$$g(i,j) = \int_{I_{\min}}^{g(i,j)} H_{\text{norm}}(g)\mathrm{d}g. \tag{4.15}$$

Intensities are quantized and, hence, histogram equalization is approximated by

$$g(i,j) = (I_{\max} - I_{\min} + 1) \sum_{g=I_{\min}}^{g(i,j)} H_{\text{norm}}(g) - 1. \tag{4.16}$$

The quantized version of histogram equalization of Eq. (4.16) can only decrease but not increase the entropy. Reordering intensity values in the histogram does not change the entropy. Mapping of two or more intensity values on the same new intensity value will decrease the entropy. Nevertheless, histogram equalization may increase the "perceived" information content by a human observer. This seems to contradict the assumption that high entropy indicates high information content. It can be explained by the limited capability of the human visual system to distinguish between intensity levels. Assuming that 64 shades of gray can be distinguished (a realistic assumption for a human observer), the histogram that represents the perceived information consists of 64 bins where sequences of four consecutive gray levels are summed up in a single bin. Even though the entropy of the 256-bin histogram decreases by histogram equalization, it may increase for a 64-bin histogram (this effect can be seen in Fig. 4.7, where histogram equalization was carried out on a CT with 1024 gray values and then mapped to 256 values for

Fig. 4.7 Even though histogram equalization decreases the entropy, the entropy may increase in a histogram after contrast resolution. In this example, histogram equalization was carried out on an image with 1024 intensity levels which was reduced to 256 intensity levels for display. The histogram of the contrast-resolution-reduced image is closer to a constant function than before histogram equalization

display). However, the increase in entropy refers only to the perception of the image by a human observer. For the computer, the decrease in entropy is definitely a loss of information. For using histogram equalization as preprocessing in computer-based image analysis, the gain in local contrast should be balanced against this information loss.

Even for perception by a human observer, histogram equalization not always achieves the desired effect. Using entropy as measure for information content implies that information of often occurring intensities should be increased by spreading them over a larger range. It comes at the cost of reduced information for intensities that rarely occur. Entropy does not consider any information about objects in an image and each pixel is equally important. For instance, the large and dark background region of the MR image in Fig. 4.8 is enhanced at the cost of reducing contrast in the foreground.

Fig. 4.8 Histogram equalization may produce unwanted effects since it enhances contrast on the assumption that often occurring gray values carry the most information. In the case depicted in this figure, it leads to the enhancement of background while reducing foreground contrast

Histogram equalization can be improved by making it locally adaptive. In *adaptive histogram equalization* (AHE), local histograms are computed separately for every pixel from the intensity distribution in some neighborhood around this pixel. The resulting mapping is carried out separately for each pixel. This definitely improves local contrast because a large region in some location of the image no longer influences the contrast of some small detail at another location. However, it does no longer guarantee that $f(i, j) \leq f(k, l)$ implies $g(i, j) \leq g(k, l)$ for all locations (i, j) and (k, l) in the image, which may confuse the observer and will confuse most computer algorithms as well.

Adaptive histogram equalization will also lead to an overamplification of noise in low contrast regions. If the gray value changes only little in a given neighborhood, the local histogram will have a sharp peak which will cause an extreme amplification of small gray level differences. *Contrast limited adaptive histogram equalization* (CLAHE) solves this problem by clipping the histogram at some level and redistributing the clipped area equally to all bins of the histogram. The clipping reduces the increase in the cumulative histogram and thus the contrast enhancement in regions with low contrast.

4.2.2 Resolution Enhancement

None of the sections above dealt with enhancing the spatial resolution, although contrast enhancement and noise removal will certainly improve the perceived resolution. Improving the spatial resolution within an image is often avoided because information needs to be added for up-sampling a given image.

Fig. 4.9 Preparations for shape-based interpolation between two binary images: Computation of the binary image from the original (in this case SPECT images from the left ventricle of the heart), computation of a distance transform in the foreground (positive distance values) and in the background (negative distance values)

Slice interpolation is a case where interpolation after reconstruction may be needed. The resolution within an image is sometimes much better than the inter-slice resolution, but most algorithms perform better on isotropic voxels.

In its simplest variant, interpolation is carried out as 1d linear or cubic interpolation in the direction of the z-axis. Interpolation is improved if structures to be interpolated are already segmented and the data is binary. Then, shape-based interpolation on the segment boundary can be carried out (Raya and Udupa 1990). Shape-based interpolation consists of three steps:

1. Creation of a *signed distance map* from every slice (see Fig. 4.9 for an example): For every voxel, a signed distance map contains a distance to the closest boundary. Voxels inside the object have a positive distance assigned to them. Voxels outside the object have a negative distance. The distance transform can be carried out, e.g., by using a morphological erosion operator on foreground and background voxels.
2. Linear (or cubic) *interpolation of distance maps*: Interpolation is carried out along the z-axis.
3. *Binarization* of interpolated slices (see Fig. 4.10): Voxels with negative distances are mapped to background and all others are mapped to foreground voxels.

Shape-based interpolation requires segmented data. It avoids many artefacts from partial volume effects under the assumption that objects in two adjacent slices only overlap if they share a common surface.[2]

For interpolation of non-segmented data, a model for continuity based on intensities (instead on boundaries such as in shape-based interpolation) between

[2]The actual definition is a bit more complex. It essentially says that only those objects in two adjacent slices should overlap for which the shortest distance on the surface between any two points on the two slices should not intersect any other slices.

Fig. 4.10 Result of interpolated intermediate slices from the binary images in Fig. 4.9

two slices is needed. A criterion that has been used is to assume that structure continuity between two slices may be derived from continuity of intensity. Interpolation would have to be carried out along the continuity direction. This is easier said than done because continuity of adjacency has to be preserved as well. A displacement field $\mathbf{d}(i, j) = (u(i, j)\ v(i, j))$ between two slices with images $f_1(i, j)$ and $f_2(i, j)$ is sought such that

$$\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left(f_1(i+u(i,j), i+v(i,j)) - f_2(i,j)\right)^2 = \min \tag{4.17}$$

and

$$\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left(u(i,j) - u_{\text{avg}}(i,j)\right)^2 + \left(v(i,j) - v_{\text{avg}}(i,j)\right)^2 = \min, \tag{4.18}$$

where $u_{\text{avg}}()$ and $v_{\text{avg}}()$ are the average values in some neighborhood around (i, j). The solution can be found using optical flow techniques. Interpolation is then carried out along displacement vectors. Using the quadratic terms in (4.18) for enforcing smoothness of the displacement field will create a smooth interpolation. This behavior is desired if slices are not too thick and a smooth variation of displacement within a slice is the norm. For thick slices, two neighboring pixels in one slice may be displaced to different sites in the next slice (e.g., if a new structure appears in the next slice between them). In such case, discontinuity-preserving constraints such as the ones presented in Sects. 4.3.2–4.3.7 may be more appropriate for achieving the desired results.

Approaches exist to apply shape-based interpolation to gray level images by inferring a shape gradient between slices. The shape gradient is defined as the direction between slices in which the smallest gray level change occurs. This can be computed for each voxel and defines a displacement field between two slices. If new slices are to be interpolated, three steps are carried out:

- Estimate displacement vectors for each voxel in the interpolated slice.
- Compute corresponding voxel in the two slices above and below the interpolated slice.
- Linearly interpolate voxel values in the interpolated slice along the interpolation direction.

The method may deliver inferior results if the input data is noisy, since the shape gradient and thus all subsequent computation depends on the reliability of the intensity information. An alternative is to use a super-resolution technique that is based on a global model (e.g., the interpolation of slices in dynamic MRI, Oktay et al. 2016). The image is assumed to be the result of a downsampling of an unknown high-resolution image. The information that has been lost by downsampling is modeled by a dictionary that is derived from training the information loss by a convolutional neural network from a large set of images. Features in the low-resolution image are transferred to the high resolution realm using an interpolated high-resolution image as spatial reference (Dong et al. 2016).

4.2.3 Edge Enhancement

Enhancing edges improves recognizing structures in images. Since automatic or interactive object delineation is a frequent task in image analysis, edge enhancement is often a prerequisite for tracking object boundaries.

Edges are closely associated with the *intensity gradient*, because the existence of an edge implies a local change in intensity. For a 2d image with continuous domain (x, y), the gradient is a vector (see Fig. 4.11)

$$\nabla f(x,y) = \left(\frac{\partial f}{\partial i}(x,y) \quad \frac{\partial f}{\partial j}(x,y) \right)^{\mathrm{T}}. \tag{4.19}$$

The *length* (the norm) of the gradient is the change in intensity in the direction of steepest ascent, hence characterizing the strength of the edge at (x, y). The gradient points in the *direction* of steepest ascent, which is perpendicular to the edge orientation at (x, y).

Fig. 4.11 Gradient is a vector that is always orthogonal to an edge. The length of the gradient depends on the strength of the edge

The gradient does not exist in the discrete domain (i, j) but can be approximated by differences

$$\nabla f(i,j) \approx \begin{pmatrix} f(i,j) - f(i-1,j) \\ f(i,j) - f(i,j-1) \end{pmatrix}. \tag{4.20}$$

Since edges and noise both have high frequency components, edge enhancement is often combined with smoothing. The two differences are computed by convolving the image with smoothing difference kernels

$$\nabla f(i,j) \approx \begin{pmatrix} [f * D_x](i,j) \\ [f * D_y](i,j) \end{pmatrix}. \tag{4.21}$$

Various kernels have been used. Examples are the *Sobel operator* with kernels $D_{\text{Sobel},x}$ and $D_{\text{Sobel},y}$ (see Fig. 4.12)

$$D_{\text{Sobel},x} = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \quad \text{and} \quad D_{\text{Sobel},y} = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}, \tag{4.22}$$

or the use of a Gaussian kernel. The partial derivatives of the Gaussian with standard deviation σ are

$$D_{\text{Gauss},x} = \frac{\partial f_{\text{Gauss}}}{\partial x}(x,y) = -\frac{x}{\sigma^3\sqrt{2\pi}} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \tag{4.23}$$

and

$$D_{\text{Gauss},y} = \frac{\partial f_{\text{Gauss}}}{\partial y}(x,y) = -\frac{y}{\sigma^3\sqrt{2\pi}} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right). \tag{4.24}$$

For computing the two filters $D_{\text{Gauss},x}$ and $D_{\text{Gauss},y}$, the derivatives are digitized and cut off using a suitable window size.

Edge enhancement using first derivatives produces information about strength and direction of edges, but it does not tell the location of an edge very well. This information is delivered from computing the second derivative. If edges are located at positions where the edge strength has a local maximum, the edge location is at the zero crossing of the second derivative.

A zero crossing is located between two adjacent pixels if one pixel has a second derivative with value 0 or if the second derivatives of the two pixels have different signs. Zero crossings are computed by the *Laplace operator* (or Laplacian). The Laplacian ∇^2 is the sum of all unmixed second derivatives,[3] i.e.,

[3]The Laplacian may also be computed as divergence of the gradient, denoted as $\nabla^2 = \nabla \cdot \nabla$, hence the symbol ∇^2.

Fig. 4.12 Gradient filters produce approximations of the partial derivatives in x- and y-direction (the two pictures in the *second row* show the result from applying the Sobel operator). The length of the gradient (*upper right*) can be used for computing edge features

$$\nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}. \tag{4.25}$$

For discrete images, it is approximated by differences, leading to a convolution with one of the following two kernels

$$D_{\text{Laplace_4}} = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix} \quad \text{or} \quad D_{\text{Laplace_8}} = \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}. \tag{4.26}$$

The former approximates (4.25) and the latter adds the two mixed derivatives to (4.25).

The Laplacian is often used in conjunction with smoothing kernel because it is very sensitive to noise. A well-known kernel is the *Laplacian of Gaussian* (LoG) filter, which computes the Laplacian of a Gaussian function

$$\text{LoG}(x, y) = -\frac{1}{\pi\sigma^4}\left(1 - \frac{x^2 + y^2}{2\sigma^2}\right)\exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right), \tag{4.27}$$

and then uses the result to convolve the image (see Fig. 4.13). The filter is sometimes called *Mexican hat filter* because of its shape. It is also known as *Marr-Hildreth filter* after David Marr and Ellen Hildreth, who showed that early edge enhancement in human vision can be modeled by an LoG filter.

The four second derivatives of a 2d image that are summed in $D_{\text{Laplace_8}}$ are components of the *Hessian* matrix

$$H = \begin{pmatrix} \frac{\partial^2 f}{\partial^2 x}(x, y) & \frac{\partial^2 f}{\partial x \partial y}(x, y) \\ \frac{\partial^2 f}{\partial y \partial x}(x, y) & \frac{\partial^2 f}{\partial^2 y}(x, y) \end{pmatrix}, \tag{4.28}$$

which may be used for enhancing corners. The value of determinant of the Hessian increases at corner points. The eigenvalues indicate whether a point in the image is a local maximum, minimum, or saddle point. If the two eigenvalues are positive, it is a local maximum; if they are negative, it is a local minimum. Otherwise, it is a saddle point.

Edges may also be enhanced in a directionally sensitive fashion by combining a difference operator with a directionally sensitive smoothing operator. A filter that has been shown to be a model for directionally sensitive cells in the primary visual cortex is the *Gábor filter*. In continuous space (x, y), it is defined as

$$G_{\sigma,\alpha,\gamma,\lambda,\psi}(x, y) = \exp\left(-\frac{s^2 + \gamma t^2}{2\sigma^2}\right)\cos\left(2\pi\frac{s}{\lambda} + \psi\right), \tag{4.29}$$

with

$$s = x\,\sin\alpha + y\,\cos\alpha, \quad t = x\,\cos\alpha - y\,\sin\alpha. \tag{4.30}$$

The exponential term is a smoothing function which is wider along s (i.e., in a direction α with respect to x) than along t perpendicular to s. The parameter

Fig. 4.13 Laplacian of Gaussian combines a Gaussian smoothing (*upper right*) with a Laplacian to arrive at the result depicted in the *lower left* image. Zero crossings of the Laplacian indicate edge locations in the image (shown in the *lower right* image)

γ determines the elongatedness of the function. The cosine term produces the difference of values along s. Its wavelength is given by λ. It controls the width of the range along which the difference is taken. The parameter ψ is an offset for the difference function.

Filter banks of Gábor filters can be used to enhance and group edges by direction, curvedness, and steepness.

4.3 Noise Reduction

Noise is usually modeled stationary, additive, and with zero mean. A noisy image g is related to the unknown noise-free image f through $g = f + n$, where n is the zero mean noise. Noise removal through linear filtering consists of estimating the expected value $E(g)$. Since $E(n) = 0$, we have

$$E(g) = E(f) + E(n) = E(f) = f, \tag{4.31}$$

because the deterministic function f has $E(f) = f$. Hence, noise reduction schemes try to reconstruct $E(g)$ from various assumptions about f:

- Linear filtering assumes f to be locally constant. E is then estimated by averaging over a local neighborhood.
- Median filtering assumes that noise is normally distributed, f is locally constant, except for edges, the signal at edges is higher than the noise, and edges are locally straight.
- Diffusion filtering and their approximations assume that f is locally constant, except at edges, and the properties of edges and noise can be differentiated by amplitude or frequency.
- Bayesian image restoration requires f to be locally smooth, except for edges. It further requires that in some local neighborhood edge, pixels are not the majority of all pixels in that neighborhood.

Since most of the assumptions are not true everywhere in the image, filtering results in various filter-specific artefacts.

4.3.1 Noise Reduction by Linear Filtering

If f is constant in some neighborhood around (i, j), $E(i, j)$ can be estimated by averaging over this neighborhood. The operation can be carried out in the spatial domain as a convolution with a *mean filter* (sometimes called *boxcar filter*) of size s:

$$f(i,j) \approx \left[g * c_{\text{mean},s}\right](i,j), \tag{4.32}$$

where "*" stands for the convolution operation. The convolution kernel $c_{\text{mean},s}$ is a square matrix of size $s \times s$ with s being odd and the filter being centered:

$$c_{\text{mean},s} = \frac{1}{s^2}\begin{pmatrix} 1 & 1 & \dots & 1 \\ 1 & & \dots & 1 \\ \dots & & & \dots \\ & & 1 & 1 \end{pmatrix}. \tag{4.33}$$

The estimate of $E(g(i, j))$ improves with the size of s, but the likelihood that f is constant in this neighborhood for all locations (i, j) decreases leading to increased blurring at edges.

Several other linear filters also deliver an estimate of $E(g)$. The *binomial filter* is a convolution kernel $c_{\text{binom},b}$ whose one-dimensional version contains the binomial coefficients of order b. A 2d kernel can be computed from convolving two 1d filters. A 1d filter of order b is created from repetitive convolution of a 1d filter of order 1:

$$c^{1D}_{\text{binom},b} = \frac{1}{2^b} c^{1D}_{\text{binom},b-1} * c^{1D}_{\text{binom},1}, \quad \text{with} \quad c^{1D}_{\text{binom},1} = [\,1 \quad 1\,]. \tag{4.34}$$

The 2d filter $c_{\text{binom},b}$ is computed by $c_{\text{binom},b} = c^{1D}_{\text{binom},1} \times \left(c^{1D}_{\text{binom},1}\right)^{\mathrm{T}}$, where T denotes the transpose of a matrix. Analysis of the binomial filter in frequency space shows that it produces fewer artefacts than the boxcar filter (see Fig. 4.14). It also emphasizes pixels in the neighborhood that are closer to the center pixel when estimating $E(g)$. This strategy is reasonable as the likelihood of f having the same value than $f(i, j)$ decreases with distance of a pixel to the center location (i, j).

With higher order, the binomial filter approaches a *Gaussian filter*. A Gaussian with given standard deviation σ can be used for noise filtering as well. The infinite support of the Gaussian function

$$c_{\text{Gauss},\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma^2}\right) \tag{4.35}$$

has to be transformed into a finite support region with some window size $|x| < x_{\max}$. Selecting $|x_{\max}| > 3\sigma$ ensures that the values $x > x_{\max}$ are sufficiently close to zero to avoid significant changes in the result. After cutting the function, it has to be normalized. Two-dimensional Gaussian kernels are created by multiplying a 1d kernel with its 1d transpose.

The separability of the Gaussian and also that of the binomial filters reduces computational costs when filter sizes are large. Instead of explicitly creating a 2d filter of size $s \times s$ and convolving the image with this filter, noise reduction is done by subsequent convolution with the two 1d filters. Computational costs per pixel are $O(s)$ instead of $O(s^2)$.

Filtering may be carried out by multiplication in the frequency domain. Noise in the frequency domain is modeled as a random process whose expected value for the amplitude is either constant (so-called white noise) or much slower decreasing with frequency than the amplitude of the signal (colored noise). In either case, noise dominates the high frequency range and noise reduction consists of applying a low-pass filter. The ideal low-pass filter with cut-off frequency $w_{\max}$, however, produces severe ringing artefacts.[4] Examples for filters that do not cause ringing are

[4]The reason for this is easily seen when the filter is transformed into the spatial domain in order to form the corresponding convolution kernel. The result is a sinc function which obviously causes the repetitions of edges after filtering, which is called ringing.

Fig. 4.14 If high frequency attenuation in frequency space by neighborhood averaging (*left*) or a binomial filter (*right*) is comparable, the results look very similar. However, attenuation in frequency space is highly anisotropic (*lower left*) while this is not the case for the binomial filter. It may cause anisotropic enhancement of edges which may be a source of problems for subsequent boundary detection steps

the Gaussian, the Butterworth filter, the Hamming or the Hann window (see Fig. 4.15 for a comparison of two filters). The latter two are often used for noise and artefact reduction in tomographic image reconstruction but are useful for additional a posteriori smoothing as well.

The Gaussian for filtering in frequency space corresponds to a Gaussian in the spatial domain with inverted standard deviation. The *Butterworth filter* attenuates noise proportional to frequency with cut-off frequency $w_{\max}$:

$$C_{\text{Butterworth},\omega_{\max}}(u,v) = \frac{1}{1 + \left((u^2+v^2)/w_{\max}^2\right)^k}. \tag{4.36}$$

Fig. 4.15 Filtering in frequency space with an ideal low-pass filter produces severe ringing artefacts (**a**) that can be avoided by a filter using gradual attenuation of high frequencies such as the Butterworth filter (**b**)

The Hamming and Hann window with cut-off frequency $w_{\max}$ are two similar filter functions. They are defined by

$$C_{\mathrm{Ha},\omega_{\max}}(u,v) = \begin{cases} \alpha - (1-\alpha)\cos\left(\frac{\sqrt{u^2+v^2}}{w_{\max}}\pi\right), & \text{if} \quad u^2+v^2 < w_{\max}^2 \\ 0, & \text{otherwise} \end{cases} \tag{4.37}$$

The filter is called *Hamming* window if $\alpha = 0.53836$. For $\alpha = 0.5$ it is called *Hann* window.

Low-pass filtering in frequency space has similar effects than approximating $E(g)$ in the spatial domain in that both methods blur edges. The reason is the same. The Fourier transform F of an image f containing an ideal step edge is a function with infinite support, i.e., there exists no cut-off frequency $w_{\max}$ for which $F(u, v) = 0$ if $u^2 + v^2 > w_{\max}^2$.

If the SNR is high and the noise is white, selecting a proper $w_{\max}$—or in spatial filtering selecting a proper filter size—is a compromise between the loss of signal at edges due to blurring and overall improvement of the SNR.

Noise reduction in medical images may be difficult because often spatial resolution must not be reduced. Small structures or structures with a very convoluted surface must still be visible after noise removal. If, e.g., voxels in an MR brain image have an edge length of 1.0 mm, it may be expected that small gyrii are recognizable which are only few millimeters apart. The assumption of a constant image function f over an area of several pixels for smoothing by linear filtering is violated in such regions. Smoothing makes detail less visible or may even make it to disappear.

Smoothing either requires a sufficiently high object-dependent contrast in the input image, a low level of spatial detail, or a combination of both. High contrast images are for instance MR soft tissue images, CT bone images, contrast-enhanced CT or MR angiograms, or some nuclear images. CT soft tissue images are difficult to smooth because of the low contrast between tissues. The same is true for ultrasound images because of the high ratio of low frequency noise and artefacts. Noise removal in nuclear images, such as SPECT, may also be inappropriate, since —despite of a good contrast—it may further reduce the already low spatial resolution.

Linear filtering will generally produce poor results if the SNR is low or if the ratio of low frequency noise is high. Successful noise reduction in such cases requires an edge model as integral part of the smoothing process. Such methods are called *edge-preserving smoothing*.

4.3.2 Median Filtering

Median filtering is a simple means to achieve edge-preserving smoothing. A *median filter* is a nonlinear rank order filter which selects the result at some location (i, j) from an ordered list of values. The values are taken from pixels in the neighborhood of (i, j). A filter is a median filter if the median rank is selected from the list. A 3×3 median filter would thus sort nine pixel values—the pixels in an 8-neighborhood of (i, j) plus (i, j) itself—and select the one ranked at position five. The neighborhood is usually square-shaped covering an odd number of pixels (e.g., 3×3, 5×5, or 7×7 neighborhoods). Since the median filter does not average over the neighborhood, it is particularly suitable to remove intensity outliers (impulse noise, sometimes called salt-and-pepper noise).

However, the median filter may also be used to reduce other noise. The result of a median filter approaches the expected value within the filtered neighborhood with increasing filter size if Gaussian noise can be assumed. Under such conditions, the median filter has similar noise reduction capabilities than linear filters presented in Sect. 4.3.1. However, there is more to the median filter. If the neighborhood region of a median filter contains an edge, then it preserves the edge under the following conditions (compare results in Fig. 4.16):

- The edge is straight within the neighborhood region.
- The signal difference of the two regions incident to the edge exceeds the noise amplitude.
- The signal is locally constant within each of the two regions.

Under these conditions, it can be shown that the filter result for a pixel (i, j) in some region r_1 at an edge separating r_1 from r_2 will always be chosen from r_1.

Although the filter is edge-preserving at such edges, it does not reduce noise, since the probability distribution of pixels $p \in r_1 \cup r_2$ is certainly not Gaussian. In

Fig. 4.16 Comparison between boxcar filter (**a**) and median filter (**b**) of the same size. It can be seen in the enlarged part that the median filter preserves sharp edges while the boxcar filter produces blur

Fig. 4.17 Median filtering on a simple test image. Corners are rounded by filters of any size, since no neighborhood size exists in which the boundary is locally straight at corners. Object details are removed if the filter size is larger than the detail

summary, the median filter is noise-reducing in neighborhoods with constant value and edge-preserving at edges.

The median filter does not work properly if the conditions listed above do not hold (see Fig. 4.17). Some of the problems that may arise are:

1. Noise reduction improves with the size of the filter under the condition that the function f is constant in this region. The usual compromise is that the neighborhood region is as large as possible while f may still be assumed to be constant. An extreme case is given, if the size of a region is a single pixel. Median filtering will remove the pixel. If it were noise (such as in impulse noise), this is a wanted behavior, but otherwise the structure would just vanish.
2. Boundaries are rarely straight. Again, the neighborhood size will be a compromise resulting in a neighborhood region just as large so that the boundary is (approximately) straight. At sharp corners where no neighborhood with straight boundary exists, median filtering removes corners.

3. The SNR at edges separating regions r_1 and r_2 may not be high enough to ensure that the noise amplitude is always lower than the signal difference between r_1 and r_2. Filtering a pixel in r_1 may lead to selection of a pixel from r_2. The course of the boundary would change. This is particularly unwanted because the median filter tends to enhance contrast between any two regions. The new boundary is visually more pleasing than the noisy boundary before filtering, but it is the wrong boundary nonetheless.
4. More than one edge may exist within the region covered by the filter. In such case, neighboring structures may be merged. This, again, is a serious alteration of the image.

All this can happen in medical images. Condition 1, 2, and 4 are often violated, because the image depicts structures or boundary details that are smaller than the filter size (consider again the detailed structure of the cortex surface in MR brain images). Condition 3 is often violated in CT soft tissue images as the contrast between different tissues is small compared to the noise level.

Since the likelihood of violating one of the conditions above increases with the size of the filter, variants exist that adapt the filter size to the local image statistics. This is fairly simple when impulse noise is expected, since the outliers vary so much from the expected data value that lower and upper thresholds $I_{\min}$ and $I_{\max}$ can be defined below, respectively, above which the pixel value is considered to be impulse noise. The filter size should be selected as small as possible but large enough to deliver a result within $I_{\min}$ and $I_{\max}$. Hence, for each location (i, j), the algorithm depicted in Fig. 4.18 is carried out.

Similar to linear filtering, pixels in the neighborhood may be given different weights. One could, for instance, give the central pixel a higher weight than other pixels so that it becomes less likely that at an edge a pixel value will be chosen from a different side of the edge than the central pixel. If the weights are positive integers, they count the number of repetitions for the weighted element in the list to be ordered. Hence, a one-dimensional weighted median filter for a 3-pixel neighborhood with weights (1 3 1) applied to a line (1 2 4) would result in an ordered list (1 2 2 2 4) of which the median would be 2. It should be clear that the sum of all weights should be an odd number if the median shall be computed.

```
set filter size fs=3
while fs=fmax do
        medfs = median(I,i,j,fs)
        if Imin<medfs<Imax then
                if Imin<I(I,j)<Imax then
                        return, I(i,j)
                else
                        return, medfs
        fs=fs+2
return, I(i,j)
```

Fig. 4.18 Adaptive filter sizes for impulse noise removal

The concept can be extended to arbitrary positive real-valued weights. Given a number of intensities $I_1, \ldots, I_N$ and corresponding weights $w_1, \ldots, w_N$, the filtering result I_{out} is given by

$$I_{\text{out}} = \arg\min_{I} \sum_{i=1}^{N} (I_i - I) w_i \tag{4.38}$$

The value for I_{out} is found by ordering intensity values into a list $I_{s(1)}, \ldots, I_{s(N)}$ so that $I_{s(i)} < I_{s(i+1)}$ for all i and searching for the index $s(m)$ such that

$$\sum_{i=s(1)}^{s(m)} w_i \geq \frac{1}{2} \sum_{i=s(1)}^{s(N)} w_i \tag{4.39}$$

The corresponding intensity $I_{s(m)}$ is the sought I_{out} that fulfils the condition of Eq. (4.38).

Since the median filter has a different behavior depending on the local distribution of intensities, it seems to be natural to adapt weights according to the local intensity statistics. This is called adaptive weighted median filter and has been used for speckle reduction in ultrasound images (Loupas et al. 1989).

The median filter—or any other edge-preserving filter—should be used with care. It may remove or alter edges that do not follow the (implicit) edge model. It can result in difficulties for later human or computer vision tasks relying on a faithful depiction of edges. This is especially true when the user is not aware of the implicit edge model.

4.3.3 Diffusion Filtering

The median filter has two disadvantages: It does not remove noise at edges even if the edge follows the implicit edge model, and it may alter edges in a random fashion that does not follow the edge model. Diffusion filtering is an alternative which enables smoothing at edges. It may also accommodate a wide range of edge models.

Diffusion filtering uses diffusion of a liquid or gaseous material as a model for noise reduction. Using homogeneous and inhomogeneous diffusion for data filtering has first been proposed by Perona and Malik (1990) and has since become very popular for edge-preserving smoothing. The popularity is probably due to the relative simplicity of concepts, the intuitive behavior of a diffusion process, and a fairly simple implementation. A good introduction can be found in Weickert (1998), now out of print but available online.

For applying a diffusion process as filter, image intensity is taken as material density. Noise is taken as density variation and diffusion is carried out iteratively. After an infinite number of iterations, *homogeneous diffusion* (Fig. 4.19a) levels any density inhomogeneity resulting in a noise-free image without edges. It can be

Fig. 4.19 **a** Homogeneous diffusion is only dependent on the density gradient, **b** inhomogeneous diffusion decreases at edges, **c** anisotropic diffusion decreases at an edge in edge normal direction

shown that the result from iterated homogeneous diffusion can be computed by filtering the image with a Gaussian with a standard deviation that depends on the (artificial) time passed for the diffusion process. Hence, homogeneous diffusion does not require iterative computation.

Diffusion across edges should be inhibited for edge enhancement. Since boundaries are unknown (otherwise, edge-preserving smoothing would be trivial), the edge response from edge enhancement is used to indicate potential boundary locations. *Inhomogeneous diffusion* (Fig. 4.19b) treats such boundary locations as a semi-permeable material. The process will still level densities (i.e., image intensities), but the process will be slower at potential edge locations. It will not be prohibited, however, because the gradient response may also be caused by noise. Hence, noise removal should stop after a number of iterations in order to prevent leveling the intensity difference between objects. The stopping criterion depends on image characteristics and on the parameters of the diffusion equation. Unlike homogeneous diffusion, computation of the diffusion result is not possible in a single step but requires iterative computation along the time axis.

Preserving boundaries may be improved if, instead of restricting any kind of diffusion at edges, it is only restricted across edges. This is called *anisotropic diffusion* (Fig. 4.19c). Allowing diffusion parallel to an edge enables noise removal by smoothing while inhibiting diffusion across an edge. Gradient direction is used as discriminative feature between noise and edges. Gradients between adjacent edge pixels tend to have similar directions while this is not true for the gradients of adjacent noise pixels. Diffusion in regions with noise pixels will be in random directions supporting homogenization, while it will be directed in boundary regions supporting contrast enhancement.

All three diffusion types can be described by the diffusion equation introduced in Sect. 2.2.8 on MR diffusion imaging. Diffusion in a 2d scene at some location (x, y) causes a flux $\mathbf{j}$ according to Fick's law

$$\mathbf{j}(x,y) = -\mathbf{D}(x,y) \times \nabla\mathbf{u}(x,y) = -\mathbf{D}(x,y) \times \begin{pmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{pmatrix}. \tag{4.40}$$

Fig. 4.20 Comparison of the different types of diffusion: **a** homogeneous diffusion, **b** inhomogeneous diffusion, **c** anisotropic inhomogeneous diffusion

The type of diffusion is described by the diffusion tensor **D**. It specifies for the density gradient $\nabla\mathbf{u}$ how much diffusion takes place in any direction.

Homogenous diffusion is independent of the strength and direction of the gradient (see Fig. 4.20a). The diffusion tensor is the product of a diffusion coefficient ε_0 and the identity matrix:

$$\mathbf{D}(x,y) = \varepsilon_0 \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{4.41}$$

Inhomogeneous diffusion depends on the gradient strength (see Fig. 4.20b). Perona and Malik, who presented diffusion filtering, suggested

$$\mathbf{D}(x,y) = \begin{pmatrix} \varepsilon\left(\|\nabla\mathbf{u}(x,y)\|^2\right) & 0 \\ 0 & \varepsilon\left(\|\nabla\mathbf{u}(x,y)\|^2\right) \end{pmatrix} \tag{4.42}$$

with

$$\varepsilon\left(\|\nabla\mathbf{u}(x,y)\|^2\right) = \varepsilon_0 \frac{\lambda^2}{\|\nabla\mathbf{u}(x,y)\|^2 + \lambda^2}. \tag{4.43}$$

Inhomogeneous diffusion depends on the location (x, y) and decreases with increasing gradient length $\|\nabla\mathbf{u}\|$. The parameter λ governs the influence of gradient length to diffusion. With λ approaching infinity, inhomogeneous diffusion turns into homogeneous diffusion.

The tensor is still a diagonal matrix imposing no constraints of the diffusion direction based on $\nabla\mathbf{u}$. This changes for anisotropic diffusion, where the diffusion tensor at each location (x, y) is chosen such that diffusion in gradient direction is

reduced while diffusion perpendicular to the gradient is not affected (see Fig. 4.20c for an example). As already mentioned in Sect. 2.2.8, the amount of anisotropic diffusion is given by eigen decomposition of **D**. The desired result for edge-preserving anisotropic diffusion is an eigenvector of which the eigenvalue decreases with increasing gradient $\nabla\mathbf{u}$ and another eigenvector perpendicular to $\nabla\mathbf{u}$ of which the eigenvalue does not depend on the strength of $\nabla\mathbf{u}$. Hence, the tensor is constructed from the desired eigen decomposition

$$\mathbf{D}(x,y) = \begin{pmatrix} e_{1,1}(x,y) & e_{2,1}(x,y) \\ e_{1,2}(x,y) & e_{2,2}(x,y) \end{pmatrix} \begin{pmatrix} \lambda(x,y)_1 & 0 \\ 0 & \lambda_2(x,y) \end{pmatrix} \begin{pmatrix} e_{1,1}(x,y) & e_{1,2}(x,y) \\ e_{2,1}(x,y) & e_{2,2}(x,y) \end{pmatrix}, \tag{4.44}$$

with

$$\begin{bmatrix} e_{1,1}(x,y) & e_{1,2}(x,y) \end{bmatrix} = \frac{\nabla u(x,y)}{\|\nabla u(x,y)\|}, \quad \lambda_1(x,y) = \varepsilon\Big(\|\nabla u(x,y)\|^2\Big), \tag{4.45a}$$

$$\begin{bmatrix} e_{2,1}(x,y) & e_{2,2}(x,y) \end{bmatrix} = \begin{bmatrix} e_{1,2}(x,y) & -e_{1,1}(x,y) \end{bmatrix}, \quad \lambda_2(x,y) = 1 \tag{4.45b}$$

Computing the diffusion requires solving a differential equation. It can be done analytically for homogenous diffusion. Homogenous diffusion at some time t equals convolution of the image with a Gaussian with variance depending on t.

The other two diffusion types are more interesting for edge-preserving smoothing. Here, diffusion is computed iteratively, assuming constant diffusion over fixed (small) time intervals. The image $f(x, y)$ is assumed to be the density distribution $u(x, y, 0)$ at time t_0. Given the density at time t_i, the density at time t_{i+1} is computed as

$$u(x,y,t_{i+1}) = u(x,y,t_i) + (t_{i+1} - t_i)\frac{\partial u(x,y,t_i)}{\partial t}, \tag{4.46}$$

where $\frac{\partial u(x,y,t_i)}{\partial t}$ indicates the change in density due to diffusion. It can be computed from the divergence of the flux

$$\frac{\partial u(x,y,t)}{\partial t} = -\operatorname{div}\mathbf{j}(x,y,t) \quad \text{with} \quad \operatorname{div}\mathbf{j} = \left(\frac{\partial j_x(x,y,t)}{\partial x} + \frac{\partial j_y(x,y,t)}{\partial y}\right). \tag{4.47}$$

Although it may look complex at first sight, implementation is straightforward. After specifying the appropriate diffusion tensor, an iterative scheme for updating the density according to (4.46) is implemented using the divergence of flux of (4.47).

Some care has to be taken on selecting appropriate parameters. The difference between time steps is critical, as it governs the approximation accuracy of solving an initial value problem by finite differences. However, if the step size is too small, noise reduction will progress slowly.

Selecting ε_0 influences the maximum diffusion per time step. Again, a small value causes slow progress, but if the value is too high, diffusion may cause oscillation of density values.

Finally, λ restricts diffusion at edges. An ideal value of λ would cause edge contrast to increase faster than contrast decreases due to diffusion. However, λ also influences the behavior within regions. For a low SNR image, there may be no value which is globally valid everywhere in the image.

Discrimination between boundaries and noise can be enhanced if gradient approximation is different for computing ε and for computing the flux due to density variations. The gradient in Eq. (4.42) could be replaced by some gradient $\nabla\mathbf{g}(x, y)$, which is intended to measure the likeliness of a pixel to be part of a boundary. This boundary gradient $\nabla\mathbf{g}(x, y)$ can be, for instance, approximated by applying a Gaussian kernel with some variance σ^2 to the density image before computing the differences. The noise gradient $\nabla\mathbf{u}(x, y)$ would still be computed without Gaussian smoothing. Details that are smaller than the variance of the Gaussian are then treated as noise and would be diffused with time, whereas larger details are treated as boundaries inhibiting diffusion.

Using anisotropic diffusion for noise removal is often preferred to inhomogeneous diffusion because resulting images look more pleasing to the observer. It should be noticed, however, that anisotropic diffusion may produce smooth but false boundaries if the true boundary contains details that are not recognized by the edge model (compare Figs. 4.21 and 4.22). This is also known as the *staircase effect* which in the limit may produce a cartoon-like appearance of the filtered image. It may be particularly misguiding in medical images since the ideal model for the imaged body is indeed a partially constant intensity function so that the staircase effect is not seen as a sign for potential artefacts. Sometimes, it is better to use an isotropic method that either leaves the noise unchanged or carries out a homogeneous smoothing. Both effects are easier recognized as an artefact than a false boundary.

Fig. 4.21 Diffusion filtering in a CT body image. While noise is reduced and contrast is enhanced, small structures start to disappear and at some locations (indicated by *arrows*) the course of the boundary starts to change

Fig. 4.22 Diffusion filtering in an MR brain image. Since structures are much smaller than in the CT image, the removal and change in boundaries due to diffusion is much more prominent than in the application depicted in Fig. 4.21

If kept going for a large number of iterations, then diffusion will almost always lead to a complete homogenization of the image. It rarely can be parameterized such that likeliness to boundary, i.e., contrast, everywhere in the image increases faster than noise is dissolved at boundaries. Hence, it is important to provide content- and parameter-specific thresholds at which diffusion stops. As such, diffusion filtering is not a suitable tool for the inexperienced user and will probably play a major role only in a research context where expert knowledge on the use and consequences of its application will be accessible.

4.3.4 Gradient Adaptive Smoothing

Gradient adaptive smoothing is an iterative process that bears similarities to diffusion filtering (Saint-Marc et al. 1991). At each iteration, a new image is generated by weighted filtering with weights depending on the local gradient.

Adaptive smoothing is defined as a series of filtering steps with a 3×3 filter kernel. Given an image I^t at iteration t, the new image at iteration $t + 1$ is computed as

$$I^{t+1}(m,n) = \frac{\sum_{i=-1}^{1}\sum_{j=-1}^{1} I^t(m+i,n+j)w^t(i,j)}{\sum_{i=-1}^{1}\sum_{j=-1}^{1} w^t(i,j)}. \tag{4.48}$$

The weighting function decreases with increased gradient length at $(m + i, n + j)$:

$$w^t(m,n) = \exp\left(-\frac{d^t(m,n)}{2k^2}\right) \quad \text{with} \quad d^t_{m,n}(i,j) = |\nabla I(m+i,n+j)| \tag{4.49}$$

The parameter k influences the amount of smoothing from adjacent pixels. The weighting causes pixels with high local gradient to influence the actual result less than pixels in homogeneous regions.

The filter can be seen as an extension of homogeneous diffusion since the latter would be achieved for constant weights w. By reducing the influence from pixels with high gradients, an inhomogeneous behavior is caused that tends to leave large intensity differences such as edges relatively intact. The version presented here does not allow anisotropic diffusion but may be extended by using the gradient direction or oriented wavelet responses to weight directionally sensitive filters (e.g., Carmona and Zhong 1998).

The method of Chen (2005) further develops adaptive smoothing by considering contextual discontinuities for defining the weights. The author states that using gradients to indicate boundaries may fail, if noise is high (as it is in many medical images). Hence, he adopts a method presented by Saha et al. (2000) to determine a local scale for each pixel and computes discontinuity based on this scale. For computing local scales, the largest circle (or sphere for 3d data) around a pixel is searched, in which 90% of the pixels vary by less than a threshold that is determined by the local variation in the immediate vicinity of this pixel. The weights for smoothing decrease with the discontinuities in the immediate vicinity and that on the contextual sphere. Since the latter is computed on a sphere that may touch but never cross a potential boundary, it is a more stable indicator to differentiate boundary gradients from noise gradients, since it usually averages over a much larger number of pixels than were used for the local gradient.

4.3.5 Bilateral Filtering

Anisotropic diffusion filtering and adaptive smoothing are iterative procedures. *Bilateral filtering* attempts to approximate this by a single-step procedure (Tomasi and Manduchi 1998). Since anisotropic diffusion restricts diffusion across potential segment boundaries, Gaussian smoothing is adapted such that the convolution kernel is additionally weighted by the intensity difference between central pixel and pixels in a predefined neighborhood (see Fig. 4.23).

It is just an approximation of anisotropic diffusion as not every pair of pixels in the set of neighbored pixels is adjacent. Hence, diffusion from one homogeneous sub-region to another sub-region is possible if the two sub-regions have similar intensities even if they are separated by another sub-region with different intensity.

The original Gaussian smoothing may be written as follows:

$$g(i,j) = 1/n_1 \sum_{(m,n)\in nb(i,j)} f(m,n) w_s\left(\left\|\begin{pmatrix} i \\ j \end{pmatrix} - \begin{pmatrix} m \\ n \end{pmatrix}\right\|\right) \tag{4.50}$$

where $w_s(x) = \exp(-x^2/\sigma_s^2)$ is a non-normalized Gaussian with variance σ_s^2, $nb(i, j)$ is a neighborhood around (i, j) including (i, j), and n_1 is a suitable normalization factor so that the weights add up to unity. The spatial kernel w_s weights the influence on the result with distance.

Fig. 4.23 Two weighting functions w_r and w_s for a pixel at position (i, j) are generated from the image function f in different ways. Whereas w_s is simply a Gaussian on the distance to (i, j), the value of w_r is given by a Gaussian on the difference to the intensity at (i, j)

For achieving a similar effect than in anisotropic diffusion, a second weighting function w_r is added that depends on the intensity difference between $f(i, j)$ and $f(m, n)$:

$$g(i,j) = \frac{1}{n_2(i,j)} \sum_{(m,n)\in nb(i,j)} f(m,n) w_s\left(\left\|\begin{pmatrix} i \\ j \end{pmatrix} - \begin{pmatrix} m \\ n \end{pmatrix}\right\|\right) w_r(\|f(i,j) - f(m,n)\|) \tag{4.51}$$

The function w_r is a non-normalized Gaussian with a different variance σ_r^2. The normalization function n_2 depends on the location (i, j) since the range kernel w_r depends on the intensity differences in $nb(i, j)$ which varies with (i, j).

By weighting the influence of neighboring pixels by their intensity difference, an effect similar to anisotropic diffusion is achieved, since anisotropic diffusion inhibits diffusion between pixels with dissimilar intensity (see Fig. 4.24 for a comparison). Diffusion is different nonetheless, since adjacency between pixels is not evaluated when computing w_r.

A similar solution was presented by Aurich and Weule (1995) who added an extra parameter η to control the amount of diffusion from neighboring pixels

Fig. 4.24 Comparison of an anisotropic diffusion filtering and a bilateral filter on an image with gray level range of 0–255, $\sigma_s = 2.0$, and $\sigma_r = 50$ (i.e., about 20% of the total range). The results are quite similar. The differences are probably mostly due to the fact that there is no exact mapping between the diffusion parameters and the parameters of the bilateral filter

$$g(i,j) = \frac{\eta}{n_2} \sum_{(m,n)\in nb(i,j)} f(m,n) w_s \left(\left\| \begin{pmatrix} i \\ j \end{pmatrix} - \begin{pmatrix} m \\ n \end{pmatrix} \right\| \right) w_r(\|f(i,j) - f(m,n)\|) \tag{4.52}$$

As in Gaussian smoothing, the size of the neighborhood should depend on the value of σ_s. The neighborhood should be sufficiently large to encompass all values of w_s substantially greater than 0. This is the case if the maximum distance between the center and the border of the neighborhood is greater than $2 \cdot \max(\sigma_s, \sigma_r)$.

The values of σ_s and σ_r hence determine the region over which smoothing takes place (for homogeneous diffusion, this represents the time during which diffusion happened). Choosing large values may reduce detail in the data while too small values would result in insufficient noise reduction (see Fig. 4.25 for examples).

The value for σ_r determines the amount of smoothing over edges (similar to the amount of diffusion across boundaries in anisotropic diffusion filtering). Hence, large values for σ_r may cause the dissolution of boundaries while too small values will leave noise relatively unchanged.

The bilateral filter has originally been presented as single-step method. A one-step filter treats intensity difference regardless of the location of the pixels in question. Hence, it cannot differentiate between single pixel intensity differences due to noise and those caused from two groups of pixels at an edge. This problem is solved to some extent by iterative bilateral filtering.

Iterative bilateral filtering first operates with small values for σ_s and large values for σ_r. Although a large σ_r would dissolve edges, the small value of σ_s produces a Gaussian smoothing over just a small neighborhood. Since noise usually has higher frequencies, it is more likely that intensity changes in such a small neighborhood

Fig. 4.25 Bilateral filtering of a 512 × 512 MRI with reduced intensity range (0–255) with varying values for σ_s and σ_r. Large values for σ_s lead to excessive smoothing boundaries while large values for σ_r will reduce detail in almost homogeneous regions

are caused by noise. Hence, the first step mainly reduces contrast in noisy regions without overly smoothing edges.

The value of σ_s is then gradually increased while σ_r is decreased. Each increase in σ_s increases the area over which smoothing is carried out. Hence, the low frequency components of the noise are now filtered. However, since σ_r is decreased at the same time, edges are kept more or less intact.

Bilateral filtering may produce similar staircase artefacts than anisotropic diffusion. Furthermore, it may result in gradient reversal artefacts if the gray level at an edge is unstable, e.g., if the noise level is high. Gradient reversal happens if the gray level at one side of the edge is closer to gray values of the other side. The range kernel does not take spatial neighborhood into account and the influence of spatial kernel for locations close to the edge does not change much, especially for large values of σ_s. This may lead to a replacement of gray values from the false side of the edge.

Even though the bilateral filter does not require iterations such as in anisotropic diffusion, it is slow. For each pixel, intensity differences to all other pixels in the neighborhood have to be computed before weighting takes place. Several approximations combined with efficient numerical schemes for computation have been presented to speed up the computation (see, e.g., Durand and Dorsey 2002; Elad 2002; Pham and van Vliet 2005; Paris and Durand 2006; Weiss 2006) so that interactive speed can be achieved.

4.3.6 Guided Filtering

Sometimes, the image data is just too noisy for serving as guide to limit smoothing. It will be difficult to find a good value for σ_r so that noise is effectively removed

without producing artefacts while edges are retained. In this case, instead of using the image itself for defining w_2, a guiding image f_g may be used to define object boundaries across which smoothing should not occur:

$$g(i,j) = \frac{1}{n_2} \sum_{(m,n)\in nb(i,j)} f(m,n) w_s\left(\left\|\begin{pmatrix} i \\ j \end{pmatrix} - \begin{pmatrix} m \\ n \end{pmatrix}\right\|\right) w_r\left(\left\|f_g(i,j) - f_g(m,n)\right\|\right). \tag{4.53}$$

If f_g is arbitrary, hence also including $f_g = f$, it is a generalization of the bilateral filter which is called the *joint bilateral filter* (Eisemann and Durand 2004).

Although f_g and f could be completely independent of each other, edges of f_g and f should coincide in order to achieve the desired edge-preserving smoothing. This has been used for improving color images, where the sharp intensity image is used to guide the restoration of edges in the hue canals. It also avoids gradient reversal if the guiding image reliably indicates edges.

The *guided filter*, presented in He et al. (2010), is a variant of this strategy. It treats the filtering process as an optimization problem to find parameters a_{ij}, b_{ij} of a linear transform

$$g(i,j) = a_{ij} f_g(m,n) + b_{ij} \tag{4.54}$$

such that

$$E(a_{ij}, b_{ij}) = \sum_{(m,n)\in nb(i,j)} \left[\left(a_{ij} f_g(m,n) + b_{ij} - f(m,n)\right)^2 + \varepsilon a_{ij}^2\right] \tag{4.55}$$

is minimal. The term εa_{ij} is a penalty term to keep the value of a_{ij} small. The model ensures that g has an edge only if f_g has an edge, since

$$\nabla g(i,j) = \nabla\left(a_{ij} f_g(m,n) + b_{ij}\right) = a_{ij} \nabla f_g(m,n). \tag{4.56}$$

The authors showed that the parameters can be computed by linear regression resulting in

$$\begin{aligned} a_{ij} &= \frac{\frac{1}{|nb|}\sum_{(m,n)\in nb(i,j)} f_g(m,n) f(m,n) - \mu_{ij}\bar{f}(i,j)}{\sigma_{ij}^2 + \varepsilon}, \\ b_{ij} &= \bar{f}(i,j) - a_{ij}\mu_{ij}. \end{aligned} \tag{4.57}$$

with μ_{ij} and σ_{ij}^2 being the mean and variance in the neighborhood nb, $|nb|$ the number of pixels in nb and $\bar{f}(i,j)$ the mean of f in the neighborhood nb.

For a given location (i, j), many windows overlap and hence there are many different locations (m, n) that produce different a_{ij}, b_{ij} for Eq. (4.57). To compute the final result, for each location (i, j), average coefficients

$$\bar{a}_{ij} = \frac{1}{|nb|} \sum_{(m,n)\in nb(i,j)} a_{mn},$$
$$\bar{b}_{ij} = \frac{1}{|nb|} \sum_{(m,n)\in nb(i,j)} b_{mn} \tag{4.58}$$

are computed resulting in a filtering result of

$$g(i,j) = \bar{a}_{ij} f_g(m,n) + \bar{b}_{ij}. \tag{4.59}$$

The guided filter can be computed in linear time and produces the exact result as opposed to approximate fast solutions to the bilateral filter.

If a guiding image does not exist and $f_g = f$ shall not be used since the original image is too noisy, a simple way to arrive at a suitable f_g is to adopt the strategy from anisotropic diffusion using different gradients for diffusion and edge detection presented in the previous section. Transferred to the realm of bilateral filters this means that f_g is a heavily smoothed and afterwards sharpened version of the original image f. Smoothing will remove most of the noise and some of the edges. Remaining edges are then restored by the sharpening step. Similar to the results in anisotropic diffusion, it may reduce boundary detail.

4.3.7 Bayesian Image Restoration

Linear filtering assumes existing prior knowledge about the noise but nothing about the image. Restoration in a probabilistic framework is able to incorporate smoothness constraints into the method. For this purpose, image characteristics are assumed to be representable by a Markov random field (MRF, see Sect. 14.1).

Determining an unknown image function f from some observation g—which not necessarily has to be an image but must be associated with the image in some known fashion—given some probabilistic knowledge about the nature of the mapping from f to g is a powerful tool based on a simple concept (see Fig. 4.26). Its application fields in image processing are image reconstruction—e.g., in the reconstruction of nuclear images (see Sect. 2.4.2)—noise removal, segmentation, and classification.

For simplifying the notation, we assume that pixels of an image are ordered as a vector. A vector $\mathbf{f}$ shall be restored from an observed vector $\mathbf{g}$. An individual pixel in the vector will be represented by f_i and g_i, respectively.

Noise reduction in a Bayesian framework searches for an image $\mathbf{f}$ that maximizes the conditional probability of observing a noisy image $\mathbf{g}$, given that the true image is $\mathbf{f}$. In Bayesian notation, we have

$$P(\mathbf{f}|\mathbf{g}) \propto P(\mathbf{g}|\mathbf{f}) \cdot P(\mathbf{f}). \tag{4.60}$$

Fig. 4.26 Image acquisition as a random process. The restoration aim is to find the most probable undistorted image given the observation and the parameter of the a priori knowledge

Fig. 4.27 Data knowledge and a priori domain knowledge about the images to be restored are the two components that define the probability of **f** having caused an observation **g**

The term $P(\mathbf{g}|\mathbf{f})$ is the data term, since it describes the dependency of the observation on the unknown undistorted data **f**. The term $P(\mathbf{f})$ comprises domain knowledge about **f** independent of an observation (e.g., that in most images neighboring scene elements have similar values, see Fig. 4.27).

The conditional probability of observing **g** given the image **f** is characterized by the type of noise. For the following, we assume zero-mean Gaussian noise with covariance matrix Σ:

$$P(\mathbf{g}|\mathbf{f}) = \frac{1}{Z_1}\exp\left[-\frac{1}{2}\left((\mathbf{f}-\mathbf{g})^{\mathrm{T}}\sum(\mathbf{f}-\mathbf{g})\right)\right]. \tag{4.61}$$

The factor Z_1 is the normalizing constant of the Gaussian distribution. If it is assumed that the probabilities of the individual pixels are independent of each other and that they all have the same variance σ^2, the above equation simplifies to

$$P(\mathbf{g}|\mathbf{f}) = \frac{1}{Z_1}\exp\left[-\sum_{i=0}^{N-1}\frac{(f_i - g_i)^q}{2\sigma^2}\right]. \tag{4.62}$$

The a priori probability $P(\mathbf{f})$ is modeled as an MRF among elements of the image $\mathbf{f}$. The probability of an MRF is

$$P(\mathbf{f}) = \frac{1}{Z_2}\exp(-U(\omega)), \tag{4.63}$$

where Z_2 is a normalization constant and $U(\omega)$ is the clique potential of all cliques in a neighborhood system. Neighborhood is defined as spatial adjacency, and cliques are possible configurations of scene elements in the neighborhood (see Fig. 4.28). Clique potentials are designed such that they enforce smoothness in the image. An example for clique energies has been given by Besag (1986),

$$U(\omega) = \beta\sum_{k=0}^{K-1} u_{ik}, \tag{4.64}$$

where K is the number of cliques and u_k is the number of pixels in clique k having the same intensity value than the center pixel i. Since this function increases the potential only for equal values, it will lead to strong edges.

If we now use (4.61) and take into account that the Markovian property ensures independence of $P(\mathbf{f})$ outside some neighborhood, we arrive at (omitting the normalization factors)

Fig. 4.28 A priori probability for some location in a grid depends on values in a freely definable neighborhood. Given a neighborhood, cliques are configurations of a subset of its elements

$$
\begin{aligned}
P(\mathbf{f}|\mathbf{g}) &\propto \exp\left(-\frac{1}{2}(\mathbf{f}-\mathbf{g})\sum(\mathbf{f}-\mathbf{g})^{\mathrm{T}}\right)\cdot\exp(-U(\omega)) \\
&= \prod_{i=0}^{N-1}\exp\left(-\frac{(f_i-g_i)^2}{2\sigma^2}\right)\cdot\exp\left(-\beta\sum_{k=0}^{K-1}u_{i,k}\right) \\
&= \prod_{i=0}^{N-1}\exp\left(-\frac{(f_i-g_i)^2}{2\sigma^2}-\beta\sum_{k=0}^{K-1}u_{i,k}\right) \\
&= \exp\left(-\sum_{i=0}^{N-1}\left(\frac{(f_i-g_i)^2}{2\sigma^2}-\beta\sum_{k=0}^{K-1}u_{i,k}\right)\right).
\end{aligned}
\tag{4.65}
$$

Maximization of $P(\mathbf{f}|\mathbf{g})$ means minimizing the exponent

$$
\sum_{i=0}^{N-1}\left(\frac{(f_i-g_i)^2}{2\sigma^2}-\beta\sum_{k=0}^{K-1}u_{i,k}\right), \tag{4.66}
$$

Finding the optimal estimate for $\mathbf{f}$ given the observation $\mathbf{g}$ is difficult as the number of configurations increases exponentially. Various techniques such as simulated annealing (Geman and Geman 1984), ICM (Besag 1986) or mean-field annealing (Zhang 1992) will be discussed in Sect. 14.1.

When Geman and Geman (1984) presented MRF Bayesian image restoration, the application example was a kind of relabeling of homogeneous regions. The definition of clique potentials and the optimization technique was geared towards restoration of a true labeling from a small set of labels. The greedy method of Besag (1986) was much more appropriate for dealing with a large number of potential labels (i.e., a large number of possible intensities). However, clique potentials defined by Geman and Geman (1984) already included an edge component supporting the creation of edges between homogeneously labeled regions. The a priori model consists of two components. One essentially smooths the data while the other supports creation of label discontinuities at region boundaries based on some line model.

Later approaches integrate the discontinuity condition parameter-free into the a priori model. In Geman and Reynolds (1992), the authors propose a constraint that enforces smoothness of some function with values known at some locations by producing the smallest possible number of intensity changes (see Fig. 4.29). The function defining the a priori probability of the MRF is

$$
\phi(u) = \frac{-1}{1+|u|^{\beta}}. \tag{4.67}
$$

This function is strictly concave for $0 < \beta < 2$ (in their experiments, the authors use $\beta = 1$), and it can be shown that for some 1d function f with values $f(x_1) = a$ and $f(x_2) = b$ with $x_1 < x_2$ and $a < b$ the result of minimizing $\int_{x_1}^{x_2}\phi(u)\mathrm{d}u$ is a function that is constant between x_1 and x_2, except for one location where

Fig. 4.29 If only similarity is required, the restored image will be blurred. The constraint from Eq. (4.67) also enforces as few intensity changes as possible, which enhances edges as well

f jumps from a to b. This is the opposite behavior to the use of interpolation functions such as $\phi(u) = u^2$, which would cause f to decrease gradually from x_1 to x_2. The behavior resulting from applying Eq. (4.67) achieves smoothness of data by maximizing the area of constant values while it enhances edges at the same time.

The two components of the logarithm of $P(\mathbf{f}|\mathbf{g}) \propto P(\mathbf{g}|\mathbf{f}) \cdot P(\mathbf{f})$ for two cliques finding first-order discontinuities, which consist of pairs of horizontally and vertically adjacent pixels, are now

$$\ln P(\mathbf{g}|\mathbf{f}) : \lambda \sum_{i=0}^{MN-1} (g_i - Kf_i)^2 \tag{4.68}$$

and

$$\ln P(\mathbf{f}) : \quad \sum_{c=1}^{C} \phi\left(\frac{\mathbf{f}_s^{(c)} - \mathbf{f}_t^{(c)}}{\Delta}\right) \tag{4.69}$$

where K describes operator blurring given by a point spread function (PSF). C is the number of cliques. The indices s and t stand for a reordering of elements in f such that $f_{s,i}^{(c)}$ and $f_{t,i}^{(c)}$ contain the two members of a clique c. The term to be minimized

$$f = \arg\min_f \sum_{c=1}^{C} \phi\left(\frac{\mathbf{f}_s^{(c)} - \mathbf{f}_t^{(c)}}{\Delta}\right) + \lambda \sum_{i=0}^{MN-1} (g_i - Kf_i)^2, \tag{4.70}$$

is not differentiable because of ϕ. For optimization, Geman and Reynolds (1992) suggest using simulated annealing. As the number of values that a pixel in f may take is large and images may be large as well, two shortcuts to the optimization are provided:

- At each iteration, a value at some site i is chosen from a range of values determined by the intensities of the four neighbors of f and from the value of g at

site i. The authors report no visible degradation if using a range of 6 values reducing the total number of states from 256 to 25 (4×6 plus the value of g_i). Although this probably has to be adopted to suit a larger range when dealing with medical images with a range of originally 4096 values, it does reduce the cost by a factor of 10.

- Annealing is not started with some random values for f but with a good guess (which could be a smoothed version of g which essentially is the maximum likelihood estimate for g without considering the Markovian prior).

It is noted that using cliques of pixels in horizontal and vertical direction may bias the result against diagonal edges. An extension to diagonal neighborhoods has been published later (Hurn and Jennison 1996).

Reducing noise based on the assumption that the image is locally constant may be overly restrictive, as most scenes may contain some smooth intensity variation within bounded regions. Hence, Geman and Reynolds (1992) extended the a priori term defining a second- and third-order version of ϕ that used the first and second-order derivative of f instead f itself.

For first-order derivative, three different cliques are defined: two cliques with three members s, t, u organized horizontally and vertically, respectively, and one clique with four members s, t, u, v organized in a square with length of two pixels (see Fig. 4.30). The function ϕ_2 is defined as

$$\phi_2\left(\frac{\mathbf{f}_s^{(c)} - 2\mathbf{f}_t^{(c)} + \mathbf{f}_u^{(3)}}{\Delta}\right) \quad \text{and} \quad \phi_2\left(\frac{\mathbf{f}_s^{(c)} - \mathbf{f}_t^{(c)} - \mathbf{f}_u^{(3)} + \mathbf{f}_v^{(3)}}{\Delta}\right), \tag{4.71}$$

depending on whether the clique consists of three or four members. For the second-order derivative, four cliques are defined. Two of the cliques have four members organized horizontally with

$$\phi_3\left(\frac{\mathbf{f}_s^{(c)} - 3\mathbf{f}_t^{(c)} + 3\mathbf{f}_u^{(3)} - \mathbf{f}_v^{(4)}}{\Delta}\right). \tag{4.72}$$

Fig. 4.30 Cliques used in Eqs. (4.70) and (4.71)

The other two cliques have two rows s_1, s_2, s_3 and t_1, t_2, t_3 of pixels organized either horizontally or vertically. The term is

$$\phi_3\left(\frac{\mathbf{f}_{s1}^{(c)} - 2\mathbf{f}_{s2}^{(c)} + 3\mathbf{f}_{s3}^{(3)} - \left(\mathbf{f}_{t1}^{(4)} - 2\mathbf{f}_{t2}^{(4)} + \mathbf{f}_{t3}^{(4)}\right)}{\Delta}\right). \tag{4.73}$$

Edge-preserving smoothing using the Bayesian approach requires that noise can be differentiated from edges based on local attributes, which requires a fairly good SNR. In the original paper, a mean SNR of approximately 15–20 has been used and restoration was able to restore the image in the presence of large blur artefacts. Higher noise may be accommodated when blurring is excluded.

It has been observed by the authors that using the second- or third-order constraints may lead to step artefacts because these constraints only approximate an intensity step edge.

A better result is achieved from consecutive application of constraints. The measured image is used as estimate for first-order discontinuity. The result serves as initial estimate for a model allowing continuous shading using the second-order constraint. This result is subjected to the third constraint resulting in smooth surfaces. The parameters λ and Δ need to be chosen carefully. Geman and Reynolds (1992) investigated restoration of a horizontal or vertical step edge for first-, second-, and third-order discontinuity. They showed that λ and Δ are dependent of each other and that, given Δ, the parameterization of λ depends on noise characteristics and on the order of the constraint.

Noise removal using Bayesian restoration has been used occasionally (Hu and Dennis 1991; Johnson et al. 1991; Garnier et al. 1995). It does play a more important role in image segmentation where the noise model is included into the segmentation task. Applications are tumor enhancement, detection and segmentation in mammography and radiography (Li et al. 1995; Baydush and Floyd 2000), in MRI (Bouman and Shapiro 1994; Marroquin et al. 2002; AlZubi et al. 2002), or in CT. Hence, we will return to the subject in Chap. 7.

Using MRFs in a Bayesian framework for edge-preserving smoothing is an elegant approach to solve the inverse problem to regain the original image from a distorted image given knowledge about attributes of noise and edges. Correct parameterization requires some understanding about the underlying methodology. As a tool, it should be preconfigured if it cannot be assured that expert knowledge is available. Similar to diffusion filtering, artefacts caused from parameterization errors may be difficult to interpret.

4.4 Concluding Remarks

Successful application of image enhancement methods requires that information content can be separated a priori from any artefactual influences. Enhancement techniques attempt to extract this information or to suppress the artefacts. Since the definition of information content will be different for every image and for every possible request for information from an image, enhancement methods are a compromise that trades generality for accuracy of this definition. Although most methods are parameterizable to accommodate enhancement of different kinds of images, the user should be aware of the extent to which he or she can adapt the method to some image class by changing parameters.

Whether application of a method makes sense and how parameters should be set is generally easy to decide if the method itself is simple (such as linear contrast enhancement or linear filtering). Efficient application of advanced methods such as diffusion filtering or Bayesian image restoration requires a good understanding of the various aspects and components of the method. Otherwise, a visually pleasing result may hide a degradation of the original information content.

Image enhancement is often a preprocessing step to segmentation. It is easier to find a segment boundary if it is clearly distinct in the image. There is a danger in this if edge-preserving smoothing has been applied. Since it is possible that false boundaries are created, segmentation results may be wrong as well. It can get difficult to detect the source of this problem when the enhanced image is sent to the segmenter without anybody checking the correctness of the enhancement step.

4.5 Exercises

- Name a difference in the goals of an enhancement technique applied to support human perception of an image as opposed to enhancement to support computer-assisted analysis.
- What contrast measure captures the local contrast at object boundaries?
- Name at least two problems when equating perceived contrast with global contrast.
- What kind of images do not profit from contrast enhancement by histogram equalization? What is the disadvantage when using adaptive histogram equalization instead?
- Why is histogram equalization not useful as preprocessing for an automatic image analysis method such as an automatic segmentation?
- How would filter masks for a Sobel filter look like, if partial differentials are approximated along the diagonal directions? How could the gradient be computed from the response of two such filters?
- How could linear filtering be used to sharpen edges? Would this be useful as preprocessing for a subsequent automatic image analysis? Explain why or why not.

- Why is it useful to combine a Laplace filter with a smoothing filter (such as a Gaussian)?
- Name the disadvantages of using a mean filter instead of a binomial filter.
- Why will the integral of all linear smoothing filters always sum up to unity?
- What kind of image degradation will always happen when using a linear low-pass filter?
- What kind of artefacts will be produced by median filtering? Under what conditions?
- Explain why the median filter will not reduce noise at edges.
- How will results of isotropic and anisotropic diffusion filtering differ from each other?
- How and where is the anisotropy of diffusion encoded in the diffusion process?
- Why will any diffusion filtering result in a completely homogeneous image provided that diffusion has been carried out long enough?
- What kind of artefact can be created by anisotropic diffusion that seriously affects perception of image content? Explain why this is inherent to the diffusion procedure.
- Approximating anisotropic diffusion by using a bilateral filter depends on two Gaussian density functions. Discuss the different effects caused by these two functions. Why is the approximation of anisotropic diffusion not perfect?
- Bilateral filtering was introduced as single step procedure but then extended to be an iterative filter. Why does it make sense to apply the filter iteratively and which parameter is changed during iteration?
- What is the purpose of the guidance image in guided filtering? How can a guidance image be generated from the image?
- What conditions need to be met so that a stochastic process is a Markov random field?
- What domain knowledge is represented by the a priori probability of an MRF?
- Why is it so difficult to find the optimal images given an observation, if observed values are an instantiation of an MRF?
- What is the role of a clique in an MRF and how does it relate to the representation of domain knowledge?

References

Al-Zubi S, Toennies KD, Bodammer N, Hinrichs H (2002). Fusing markov random fields with anatomical knowledge and shape based analysis to segment multiple sclerosis white matter lesions in magnetic resonance images of the brain. In: Proceedings of SPIE (medical imaging 2002), vol 4684, pp 206–215

Aurich V, Weule J (1995) Non-linear Gaussian filters performing edge preserving diffusion. Mustererkennung 1995:538–545

Baydush AH, Floyd CE (2000) Improved image quality in digital mammography with image processing. Med Phys 27(7):1503–1508

Besag J (1986) On the statistical analysis of dirty pictures. J R Stat Soc Ser B (Methodological) 48 (3):259–302

Bouman CA, Shapiro M (1994) A multiscale random field model for Bayesian image segmentation. IEEE Trans Image Process 3(2):162–177

Carmona R, Zhong S (1998) Adaptive smoothing respecting feature directions. IEEE Trans Image Process 7(3):353–358

Chen (2005) Adaptive smoothing via contextual and local discontinuities. IEEE Trans Pattern Anal Mach Intell 27(10):1552–1567

Dong C, Loy CC, He K, Tang X (2016) Image super-resolution using deep convolutional networks. IEEE Trans Pattern Anal Mach Intell 38(2):295–307

Durand F, Dorsey J (2002) Fast bilateral filtering for the display of high-dynamic-range images. ACM Trans Graphics (TOG) 21(3):257–266

Eisemann E, Durand F (2004) Flash photography enhancement via intrinsic relighting. ACM Trans Graphics (TOG) 23(3):673–678

Elad M (2002) On the origin of the bilateral filter and ways to improve it. IEEE Trans Image Process 11(10):1141–1151

Garnier SJ, Bilbro GL, Gault JW, Snyder WE (1995) Magnetic resonance image restoration. J Math Imaging Vis 5(1):7–19

Geman S, Geman D (1984) Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Trans Pattern Anal Mach Intell 6(6):721–741

Geman D, Reynolds G (1992) Constrained restoration and the recovery of discontinuities. IEEE Trans Pattern Anal Mach Intell 14(3):367–383

Haralick RM (1979) Statistical and structural approaches to texture. Proc IEEE 67(5):786–804

He K, Sun J, Tang X (2010) Guided image filtering. Comput Vis—ECCV:11–14

Hu Y, Dennis TJ (1991). MAP estimation in image restoration by a local search enhanced genetic algorithm. In: 6th international conference on digital processing of signals in communications, pp 123–128

Hurn M, Jennison C (1996) An extension of Geman's and Reynold's approach to constrained restoration and the recovery of discontinuities. IEEE Trans Pattern Anal Mach Intell 18 (6):657–662

Johnson VE, Wong WH, Hu X, Chen CT (1991) Image restoration using Gibbs priors: boundary modeling, treatment of blurring, and selection of hyperparameters. IEEE Trans Pattern Anal Mach Intell 13(5):413–425

Li HD, Kallergi M, Clarke LP, Jain VK, Clark RA (1995) Markov random field for tumor detection in digital mammography. IEEE Trans Med Imaging 14(3):565–576

Loupas T, McDicken WN, Allan PL (1989) An adaptive weighted median filter for speckle suppression in medical ultrasonic images. IEEE Trans Circ Syst 36(1):129–135

Marroquin JL, Vemuri BC, Botello S, Calderon E, Fernandez-Bouzas A (2002) An accurate and efficient Bayesian method for automatic segmentation of brain MRI. IEEE Trans Med Imaging 21(8):934–945

Oktay O, Bai W, Lee M, Guerrero R, Kamnitsas K, Cabellero J, Monteiro de Marvao AMS, Cook S, O'Regan D, Rueckert D (2016) Multi-input cardiac image super-resolution using convolutional neural networks. In: MICCAI 2016. Part III, LNCS, vol 9902, pp 246–254

Paris S, Durand F (2006). A fast approximation of the bilateral filter using a signal processing approach. Comput Vis—ECCV:568–580

Peli E (1990) Contrast in complex images. J Opt Soc America A 7(10):2032–2040

Perona P, Malik J (1990) Scale-space and edge detection using anisotropic diffusion. IEEE Trans Pattern Anal Mach Intell 12(7):629–639

Pham TQ, Van Vliet LJ (2005). Separable bilateral filtering for fast video preprocessing. In: IEEE international conference multimedia and expo, ICME 2005

Raya SP, Udupa JK (1990) Shape-based interpolation of multidimensional objects. IEEE Trans Med Imaging 9(1):32–42

Saha PK, Udupa JK, Odhner D (2000) Scale-based fuzzy connected image segmentation: theory, algorithms, and validation. Comput Vis Image Underst 77:145–174

Saint-Marc P, Chen JS, Medioni G (1991) Adaptive smoothing: a general tool for early vision. IEEE Trans Pattern Anal Mach Intell 6:514–529

Tomasi C, Manduchi, R (1998). Bilateral filtering for gray and color images. In: IEEE international conference computer vision (ICCV 1998), pp 839–846

Weickert J (1998) Anisotropic diffusion in image processing. Teubner, Stuttgart

Weiss B (2006) Fast median and bilateral filtering. ACM Trans Graphics (TOG) 25(3):519–526

Zhang J (1992) The mean field theory in EM procedures for fields. IEEE Trans Signal Process 40 (10):2570–2583

Feature Detection

5

Abstract

Image-based features such as key point locations or potential parts of object boundaries can be extracted from local image characteristics. Boundary parts are generated from results of an edge-enhancement step, while key point locations are local extrema of some local object property. Features may also be computed from samples of an object's boundary or interior. Potential object boundary parts are used for detecting or delineating objects in images. Key points may in some simple cases also be used to detect objects. In most cases, however, object characteristics are too complex to be captured by the attributes of a key point. They can be important attributes nonetheless. Key points define an object-dependent reference system in which they may be used to map objects of the same class onto each other. Region characteristics describe object boundaries just indirectly. They are often more robustly to compute as they often require only a rough estimate of the interior for their computation. Hence, they are more often used for object detection rather than for object delineation.

Concepts, Notions, and Definitions Introduced in this Chapter

- Edge tracking;
- Canny edge operator;
- Blobs and vesselness, Hough transform for lines;
- SIFT, SURF, BRIEF, and ORB features, superpixels, texture, MSER features, local shape context, texture features, HOG features, gist, and saliency

Image analysis aims at reducing information to a subset that is relevant to some analysis question. An example would be volumetry where the user wants a single number (the volume of a certain organ or pathological structure) from the image sequence. Other examples are the detection of metastases or the delineation of an organ boundary for radiotherapeutic treatment planning. Information reduction

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_5

often happens gradually with information being reduced until the desired result is extracted from the data. The first level of reduction computes local features that are assumed to pertain to objects of interest. Examples for local features are edges, blobs, or ridges in the image. Such features carry more information than a pixel, as they represent a first differentiation between object attributes and influences from image acquisition.

There are a number of reasons to apply a feature detection step in medical image analysis. In cases where the object of interest is simple, its attributes may be captured directly by the feature detector. This leads to object detection. More often, it is a preprocessing step. Feature locations and attributes may serve as an object-specific reference if the same object is captured by different images that shall be compared. While the content of the images may be different (e.g., if one of the images is CT and the other MRI), the object features such as edges or corners should be extractable in both images. Features may also help to define a region of interest that shall be further inspected. A blob detector, for instance, could find potential regions of interest that may comprise the blob-like lymph nodes. Further processing will then be restricted to these regions. Features may also help to guide a segmentation process in cases where the data is noisy or of low contrast. Edge features, for instance, could be generated that support segmentation if noisy data would produce too many spurious responses from applying a simple gradient operator. Finally, features may also be used to characterize deviations from an assumed norm of an anatomic structure. As an example, features to enhance tubular structures may highlight potential sites of aneurisms in the vascular system, since they deviate from the normal, tubelike shape of vessels.

The description of a feature is richer than that of a pixel. It comprises at least the feature location in image space and the feature type (e.g., edge or corner) and some measure of strength of the feature response at that location. It may include other information such as scale (e.g., size of a blob) or orientation (e.g., the direction of an edge).

5.1 Edge Tracking

Structures are meant to be detectable by a change of appearance between them and the background. Furthermore, the shape given by a structure's outline may be an important characteristic to differentiate it from other objects or to specify object-specific locations. Hence, edges are relevant features for several types of analysis tasks.

Edges can be those of intensity but may also separate different textures. The following section will deal with intensity edges. Most of the techniques can be applied to textures as well, provided that the texture measure can be mapped on a single scalar value.[1] Texture edges are then edges of this scalar value.

[1]This can be difficult, however, since texture measures are estimated from a local neighborhood that is assumed to have the same texture. This is not true at texture boundaries.

Detection of intensity edges uses the output of an edge-enhancement step. Edge strength is given by the length of the intensity gradient. The edge location is given by the zero-crossings of a Laplace operator (see Fig. 5.1). However, a large gradient does not automatically mean that there is an edge. Finding salient edges in an image can be difficult. Saliency can be based on many different attributes ranging from local smoothness and continuity constraints to high-level domain knowledge. Hence, edge detection is still an open problem in the general field of image analysis. An extensive survey of past and recent techniques can be found in Papari and Petkov (2011).

For edge detection in medical images, various, rather simple assumptions are used to separate edges from noise:

- The gradient for edges is often stronger than that caused by noise.
- The edge direction varies slowly along the edge.
- The edge strength varies slowly along the edge.

These assumptions are too simple to detect all edges defining an object boundary. They are, however, sufficient, since edges resulting from a tracking step are usually not the final result. Rather, these edges initialize some top-down analysis step to find boundaries of an organ, pathological process, or other structure of interest.

The *Canny edge operator* (Canny 1986) takes into account all three assumptions. It consists of an edge-enhancement step and an edge-tracking step. Edge enhancement is carried out by taking the maximum response from several one-dimensional edge filters, because—under an idealized edge model—the optimal response for an edge is created by a one-dimensional smoothing differential operator orthogonal to the yet unknown edge direction. The local maximum of the gradient length then specifies the edge location. In most applications, this step is

Fig. 5.1 Zero-crossings computed from applying an LoG operator (**a**) may be combined with information about edge strength from a gradient operator (**b**) to find edge locations and rate their importance based on edge strength (**c**)

replaced by a two-dimensional gradient operator (e.g., the derivatives of the Gaussian) combined with a non-maximum suppression step that reduces the edge response to a single pixel in gradient direction. Non-maximum suppression can be done, e.g., by computing zero-crossings of the second derivative.

Edge tracking is done by hysteresis thresholding. Two thresholds, t_1 and t_2 with $t_1 > t_2$, are defined and applied to the gradient length $|\mathbf{g}|$. If $|\mathbf{g}| > t_1$ for some pixel, this pixel always belongs to an edge. Pixels with $|\mathbf{g}| > t_2$ are edge pixels if they are adjacent to other edge pixels. The algorithm proceeds as follows (see also Fig. 5.2):

- Select the next pixel with $|\mathbf{g}| > t_1$ that is not yet assigned to an edge and assign it to a new edge.
- Track the edge as long as adjacent pixels are found with $|\mathbf{g}| > t_2$.

This process is repeated until no further pixels with $|\mathbf{g}| > t_1$ are found. The method finds connected edge segments. At intersections, it will track only one of the continuing curves. The other curve will be found as well if at least one of its edge pixels has a gradient larger than t_1. The value for t_1 should be high enough to make sure that none of the starting pixels is a noise pixel. However, since continuation of an edge is only found between neighboring pixels, the threshold t_2 should be low as not to hinder tracking (see the effect of different selections for t_1 and t_2 depicted in Fig. 5.3).

Fig. 5.2 The different steps of the Canny edge operator (the result from non-maximum suppression has been dilated for better visibility)

Fig. 5.3 Different choices for the two thresholds of the Canny edge detector lead to different results. Thresholds are given as percentage of the strongest gradient in the image

The Canny edge detector (or similar tracking approaches) will create edges as a subset of all edge locations with a certain strength. Since locations to be considered for an edge are local maxima of the gradient length, edges may be false if noise has distorted the course of the edge. Low-pass filtering before computing the gradient prevents some of these erroneous responses, but it may also lead to false local maxima locations if smoothing causes nearby edges to fuse.

Multi-resolution edge detection and multi-resolution edge tracking are two ways to get around this. In *multi-resolution edge detection*, the gradient response is calculated at different scales. For each location, an optimal scale is chosen to compute the response. Hence, instead of deciding for a single scale for the complete image—as it is done when selecting the variance in Gaussian smoothing—a different scale is selected for each location. The wavelet decomposition of an image into localized frequency components is a good way to arrive at such a multi-scale edge response. Mallat and Zhong (1992) have presented the appropriate wavelet transform. While wavelets provide for an (almost) redundancy-free representation of scale and frequency components, redundant decomposition schemes such as

using a sequence of Gaussians with different variances could be applied as well [a good survey on these techniques including multi-resolution tracking techniques is given in Basu (2002)].

In *multi-resolution edge tracking*, edges are searched at different scales. The process commences with the lowest resolution and then proceeds to the next higher resolution (see Fig. 5.4). At the higher resolution, edge locations from the previous resolution level are used to initiate the search for edges on the current level. In Williams and Shah (1990), e.g., three different scales are used by convolving the image with three different Gaussians with standard deviations σ, $\sqrt{2}\sigma$, and 2σ and computing gradients on each resolution level. The basic resolution σ has to be determined based on expected resolution and noise in the image. Edges are found in the lowest resolution image and are confirmed on the next higher resolution. Detected edge locations on the current resolution image replace edges from the lower resolution if they are close to an already found edge and have a similar direction. This way, edge detection on the coarser resolution suppresses spurious edges while at the same time allowing for corrections of accidentally fused edges or of an overly smoothed course of an edge.

If data is very noisy, e.g., in ultrasound images, none of the techniques mentioned above will produce useful edges. The amount of necessary smoothing would very likely interfere with proper edge localization. In such case, edge detection needs to be preceded by a restoration step, such as the edge-preserving smoothing techniques presented in Sects. 4.3.2–4.3.7. Filters are often adapted based on knowledge about the characteristics of noise or artifacts, such as the anisotropic diffusion filter for reflection images in Yu and Acton (2002).

Alternatively, an edge or contour model has to be created that accounts for the distortions in the data.

Contour models differ from edge models in that they assume that a set of open or closed contours are searched in the image in a top-down fashion, while edge models generate edges in a bottom-up fashion based on a (possibly very elaborate) model of what an edge is in this particular image. Contour models will be discussed in Chap. 9 (Active Contours and Active Surfaces).

Fig. 5.4 Multi-scale edge detection on image (**a**) allows to detect edge detail on a high-scale image (**b**) and to decide whether to keep the edge based on a low-resolution image (**c**)

An *edge model* is usually just a local template that is convolved with the image. The template represents the ideal edge. The response to the match can be exploited to determine edge locations as well as a measure of confidence for the edge to be present [see, e.g., Meer and Georgescu (2001)]. For ultrasound images, an example for such a template model would be that of a stick model presented by Czerwinski et al. (1993) who argued that boundaries in ultrasound are not visible as intensity changes but as reflection lines at tissue boundaries. Hence, they matched a line model with the image. The expected response for a location being part of an edge is then a threshold that is derived from known noise characteristics in the ultrasound images (Czerwinski et al. 1994).

5.2 Hough Transform

The *Hough transform* computes edge features by comparing image evidence with a very specific edge model. Given an image containing edge information of an unknown number of edges of a known kind, the Hough transform finds instances of this kind.

The Hough transform is a voting scheme that has been first presented to find straight lines in images and then has been extended to find arbitrary kinds of boundaries. Each location of a potential boundary—e.g., each location where the gradient length exceeds some threshold—votes for reference points in parameter space that are associated to certain shapes. Parameter combinations that receive the most votes describe likely object instances.

Being a voting system, the Hough transform keeps its ability to predict structure locations even if some votes are missing because of occlusion or a missing signal. The method is robust with respect to noise or artefactual edges that do not follow the edge model. Small variations from the predicted shape are tolerated although the Hough transform is less robust to variation of an object outline than template matching.

The Hough transform can be defined for any dimension but is most often applied in 2d because the number of parameters, i.e., the dimensionality of parameter space in which voting takes place, increases fast with increasing dimension of the images.

The transform for finding straight lines was presented in Hough (1962). A line in 2d space $\mathbf{x} = (x_1\ x_2)$ can be defined by the following variant of the line equation

$$\mathrm{d}(\alpha) = x_i \cos(\alpha) + y_i \sin(\alpha), \tag{5.1}$$

where α is the angle and $\mathrm{d}(\alpha)$ the distance to the origin of a line passing through a point $(x_i\ y_i)$. The Hough transform for a given potential boundary point $(x_i\ y_i)$ computes $d(\alpha)$. This point votes for all locations in parameter space $(\mathrm{d}(\alpha), \alpha)$ for which Eq. (5.1) is true (see Fig. 5.5). The parameter space is digitized into bins in which votes accumulate (called *accumulator cells*). The number of votes for a

Fig. 5.5 Each edge point (x_n, y_n) in image space is represented by a curve in Hough space. The curve describes all parameter combinations α, d(α) for lines passing through (x_n, y_n)

potential boundary point in an image depends on its probability of being part of a boundary. It could be, for instance, a function of the gradient length.

Lines in the image are those for which corresponding accumulator cells are local maxima and have received a sufficiently high number of votes. The QOF measure is a threshold on the number of votes (see Fig. 5.6 for an example).

A number of strategies increase the computation speed of the Hough transform and can be applied to most of its variants:

- The order in which votes are cast does not matter, which makes the method inherently parallelizable.
- If gradient directions in the edge images are reliable, the number of votes may be reduced by letting every edge point only vote for those solutions of Eq. (5.1) for which α is almost perpendicular to the gradient direction.
- If edge points are selected randomly from the image, intermediate results of the voting process may already be a good estimate for the final outcome.

There are also some strategies for increasing the robustness of the Hough transform with respect to noise, artifacts, and shape variation:

Fig. 5.6 Example of applying the Hough transform on an edge image. The predominance of edges at angles 45° and 135° with respect to the x-axis is visible as local maxima in Hough space

- A multi-scale strategy may be applied by computing an initial Hough transform only for large accumulator cells. The result is used as a prediction for ranges of parameters in Hough space that represent potential lines. Accumulation of votes at higher resolution is reduced to these ranges.
- Vote distribution in parameter space may be smoothed in order to take variation due to noise and artifacts into account.

The voting strategy of the Hough transform is not restricted to the search of straight-line segments. Any boundary structure that can be represented by a small set of parameters can be found. Hence, the Hough transform is suitable to represent shape information for searching object instances instead of object features in the image. This will be detailed in Chap. 11.

5.3 Corners

Corners in an image indicate specific locations pertaining to an object. Such locations are important if objects of the same kind are to be compared or if the same object shall be compared in different images (e.g., comparing brain data of a patient from CT and PET). Another reason to identify specific locations is to classify an unknown object by its characteristic outline as given by the corner point locations.

The identification of a characteristic location will always require further knowledge (e.g., anatomical knowledge about configuration or appearance), but the minimum requirement is that this location is identifiable as a specific point. This is what a *corner detector* does.

Corners belong to a class of local features that can be computed from images but represent attributes of depicted objects. This is of interest in many fields of computer vision. For more details, the reader is pointed to a survey about feature detectors in (Tuytelaars and Mikolajczyk 2007) who discusses purpose, requirements, methods, and applications of local image features. The characteristic that is exploited for most corner detectors is that the so-called aperture problem does not exist for a corner.

The *aperture problem* states that for at least one direction, the appearance of the image in a small window does not change if the window is moved in this direction. In homogeneous regions, this is true for arbitrary directions. At edges, it is true for the direction along the edge. For corners, this is not true for any direction.

The *Harris corner detector* (Harris and Stephens 1988) is based on this assumption and has been shown to have a good performance (Schmidt et al. 2000). The detector computes a quantity for a location $(x\ y)$ that depends on averaged intensity variations in arbitrary directions around $(x\ y)$. If this variation is high in almost all directions, a point of interest is found (see sketch in Fig. 5.7). The neighborhood across which this variation is averaged constitutes the scale of the corner detector. A second scale parameter is that of the image, given by the amount of smoothing prior to compute the intensity variation between pixel locations.

Fig. 5.7 The Harris corner detector measures changes of intensity of a region around a point. A corner causes significant changes for displacement in arbitrary direction, an edge only if displaced orthogonal to it

The corner feature is computed from the eigenvalue of the *Harris matrix*. It consists of the partial derivatives, weighted with some function w of the image function $I(x, y)$ in a neighborhood k around a direction $(x_0\ y_0)$ in which the image is displaced and then subtracted. It is defined as follows

$$H(x_0,\ y_0) = \sum_{x=x_0-k}^{x_0+k} \sum_{y=y_0-k}^{y_0+k} w(x,y) \begin{pmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) \\ \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) & \left(\frac{\partial I}{\partial y}\right)^2 \end{pmatrix}. \tag{5.2}$$

This matrix represents the change of intensity, if the image was shifted by $\pm k$ and then subtracted since it is derived from computing the weighted sum of squared differences in this neighborhood

$$S(x_0, y_0) = \sum_{x=x_0-k}^{x_0+k} \sum_{y_0-k}^{y_0+k} w(x - x_0, y - y_0)(I(x_0, y_0) - I(x, y))^2 \tag{5.3}$$

by approximating $I(x, y)$ using the Taylor expansion

$$I(x, y) = I(x_0, y_0) + (x_0 - x)\frac{\partial I(x_0, y_0)}{\partial x} + (y_0 - y)\frac{\partial I(x_0, y_0)}{\partial y} \tag{5.4}$$

around (x_0, y_0), which results in

$$S(x_0, y_0) = \sum_{x=x_0-k}^{x_0+k} \sum_{y_0-k}^{y_0+k} w(x - x_0, y - y_0) \left((x - x_0) \frac{\partial I(x, y)}{\partial x} + (y - y_0) \frac{\partial I(x, y)}{\partial y} \right)^2 . \tag{5.5}$$

Setting

$$\mathbf{H}_p = w(x - x_0, y - y_0) \begin{pmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) \\ \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) & \left(\frac{\partial I}{\partial y}\right)^2 \end{pmatrix} \tag{5.6}$$

and substituting $u = x_0 - x$ and $v = y_0 - y$, this can be written as

$$S(x_0, y_0) = \sum_{x=x_0-k}^{x_0+k} \sum_{y=y_0-k}^{y_0+k} (u \quad v) \mathbf{H}_p (u \quad v)^{\mathrm{T}} = (u \quad v) \mathbf{H} (u \quad v)^{\mathrm{T}}. \tag{5.7}$$

If the point in question is a corner, this difference should be large for arbitrary directions $(x_0 \; y_0)$. Hence, the two eigenvalues of $\mathbf{H}$ will be large. If just one eigenvalue is large, an edge has been found. There exists only one specific direction (the corresponding eigenvector orthogonal to the edge) in which the difference is large. In summary, three different cases can be derived from inspecting the eigenvalues λ_1 and λ_2 of $\mathbf{H}$:

- A corner has been found if λ_1 and λ_2 are large.
- An edge has been found if either λ_1 or λ_2 is large.
- The region is locally homogeneous if λ_1 and λ_2 are low.

The detection is not scale-invariant, however. The scale of the response is given by two parameters. The first is the amount of smoothing that is part of the approximation of the partial derivatives, and the second is the window size over which the Harris matrix is averaged. The corner detector can be made rotation-invariant by using a rotationally symmetric window function w (e.g., a Gaussian).

Another popular corner detector that performed well in comparison tests is *SUSAN* corner detector [smallest univalue segment assimilating nucleus (Smith and Brady 1997)]. The operator assumes

- that a point of interest has a different intensity than locations in a prespecified neighborhood,
- that this difference is larger than a threshold value g,
- that this point of interest is a local maximum of this difference.

First, an intensity-gradient-based response in some neighborhood N is computed as follows

$$R(x,y) = \sum_{(u,v)\in N} \exp\left(-\frac{[I(x,y) - I(u,v)]^6}{t}\right). \tag{5.8}$$

The exponent has been found experimentally. The value of t governs the strength with which the intensity difference is influencing R. The higher the average intensity difference is in N, the lower is the response. To compute the final response S_{SUSAN}, R is inverted and thresholded with g:

$$S_{\text{SUSAN}} = \begin{cases} g - R(x,y), & \text{if } g - R(x,y) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{5.9}$$

The value of S_{SUSAN} is a local maximum if (x, y) is a corner or a single point. The operator is faster to compute than the Harris corner detector but was found to be more sensitive to noise.

Features from accelerated segment test (*FAST*, Rosten and Drummond 2006) corner detector follow a similar strategy for interest point detection than SUSAN but use just the intensity values on a digitized circle of a given radius R (called FAST-R, the authors used $R = 3$, but in later applications, the value of R was adopted according user needs) around some pixel $\mathbf{p}$. Now, the length $L_{\mathbf{p}}$ of the longest path of pixels significantly brighter or darker than the intensity $I(\mathbf{p})$ at $\mathbf{p}$ is determined. Significance is determined by a minimum intensity difference $d_{\min}$ between the intensities at $\mathbf{p}$ and a pixel on the circle. The pixel $\mathbf{p}$ is a corner if $L_{\mathbf{p}}$ is greater than some threshold N.

Radius R, significance threshold $d_{\min}$ and length N are parameters of the corner detector that are specified according to the application. As in SUSAN, it is assumed that within a circle of radius R, there is at most one corner. Hence, R needs to be small enough as to not include more than one corner, but also large enough for not being affected from blur at corners. The value of N should be large enough for excluding edge pixels to be defined as corners. In Rosten and Dummond (2006), the authors suggested $N = 12$ for $R = 3$. Since a digitized circle of radius $R = 3$ contains 16 pixels, N is set to 75% of the pixels in the circle. A pixel $\mathbf{p}$ is a corner only if the angle of the two sides meeting at $\mathbf{p}$ is at most 90°.

For speeding up computation, first just four pixels are evaluated that are 90° apart on the circle. If no sequence of 3 pixels exists with intensities either all significantly above or below $I(\mathbf{p})$, the pixel is not investigated further, since any sequence containing at least 75% of the pixels can exist.

The authors noted that the speed and results depend on the selection of the first four pixels and presented a machine learning approach to select optimal pixel locations based on training images.

5.4 Template Matching

Template matching may be used to detect or highlight structures if their shape is known and simple. Two examples are blobness and vesselness filters.

Blobs are circular structures in the images. *Blob detection* either detects such structures (e.g., in counting microorganisms in cell microscopy) or highlights blobs as important features of an object to be detected, e.g., in the detection of lung nodules (Coppini et al. 2003; Schilham et al. 2003). Formally speaking, a blob is a local maximum or minimum of a radially symmetric intensity distribution.

Blobs are characterized by their size and location. If size is known, blobs can be enhanced by a matched filter of the same size and shape. The *Laplacian of Gaussian* with standard deviation σ according to the blob size is often used:

$$\mathrm{LoG}(x, y) = -\frac{1}{\pi\sigma^4}\left(1 - \frac{x^2 + y^2}{\sigma^2}\right)\exp\left(-\frac{x^2 + y^2}{\sigma^2}\right) \tag{5.10}$$

Alternatively, the *Difference of Gaussian* (DoG) filter may be used. The response of a DoG filter is computed by subtracting the filter response of the image with two Gaussians with different standard deviation from each other.

Potential blob sites are the local maxima (or minima, if the blob is darker than the background) of the convolution result (see Fig. 5.8 for an example). Blobs of size σ are detected by thresholding the result. If finding a suitable threshold is difficult, blob responses may be ordered by the strength of the response and the threshold is defined by selecting the n most prominent blobs.

Since the scale and hence the size of the blob in image coordinates are not known, blob detection can be extended to a scale-invariant detection including σ as a parameter in $\mathrm{LoG}(x, y, \sigma)$. In order to make the response independent of the size of the blob, the function needs to be normalized with σ yielding

Fig. 5.8 Result of a blob detection on a microscopy with blob-like microorganisms. Locations of organisms are the local maxima of the detector response map. The large and bright structure on the lower right (see *arrow*) produces a small response, because only the (barely visible) blobs in this structure are of the expected blob size. The blob conglomerate on the right (see *arrow*) is nicely resolved into its constituent blobs

$$\mathrm{LoG}_{\mathrm{norm}}(x,\ y,\ \sigma) = -\frac{\sigma}{\pi\sigma^4}\left(1 - \frac{x^2+y^2}{\sigma^2}\right)\exp\left(-\frac{x^2+y^2}{\sigma^2}\right) \tag{5.11}$$

The local minima or maxima in this space represent the blob location and the size of the blob.

A different approach is to use the determinant of the *Hessian matrix*. It evaluates the differential properties of the point-like shape of blobs. The Hessian is the matrix of partial derivatives of a function. For a 2d image $I(x,\ y)$, it is

$$\mathbf{H} = \begin{pmatrix} \frac{\partial I^2}{\partial^2 x} & \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} \\ \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} & \frac{\partial I^2}{\partial^2 y} \end{pmatrix} \tag{5.12}$$

Its determinant det(**H**) should be large if the image intensity peaks at some location, i.e., if this location is a blob. Hence, blob detection consists of detecting local maxima of the determinant. Again, the detector can be made scale-invariant by including the scale parameter into the function and by searching for local maxima in space and scale. Since the value of the determinant decreases with increasing blob size, the response is normalized by the blob size σ:

$$\mathrm{Blob}_{\mathrm{Hessian}}(x, y) = \sigma^2 \det(\mathbf{H}(x, y)). \tag{5.13}$$

A different approach is to use the shape of a blob directly by applying a kind of watershed transform (Lindeberg 1993). If a blob is a local maximum (or minimum), pixels surrounding the blob center should have lower (or higher) values. A standard watershed transform on such an image would segment the image into regions of which blobs that are darker than the background are a subset. The watershed transform treats intensity values as heights in a 3d landscape and computes watersheds by flooding the landscape. Flooding starts from local minima in the image (the sources in the landscape). A watershed is found when flooding water from different sources meets (for details on the watershed transform see Roerdink and Meijster (2001) or Sect. 6.7 in Chap. 6).

In order to separate blobs from background, flooding stops if an intensity level is reached beyond which everything counts as background. Blobs that are brighter than the background can be found as well by inverting the procedure, i.e., by flooding from the brightest pixel and again stopping at some threshold intensity.

The *vesselness filter* finds tubular structures. The basic idea is similar to the definition of the blobness filter. If an elongated structure can be discriminated in the image (brighter or darker than the background), then the preferred direction can be determined from an eigen decomposition of the Hessian matrix.

The vesselness filter was presented by Frangi et al. (1998) to find vessels in MRA images but may also be used to find other tubular structures in the human body. Frangi et al. (1998) derive a weighted combination of two geometric components and one intensity component as vesselness response from the eigenvalues

$\lambda_1 < \lambda_2 < \lambda_3$ of the Hessian matrix. The filter result is computed for a range of scales that represent the range of sizes expected for vascular structures. The maximum response over the scale range is taken as vesselness.

The three components represent different aspects of the vessel. For geometric components, the authors compute "non-blobbyness" and elongatedness along the direction of the tubular structure. Non-blobbyness R_a is the ratio of the smallest eigenvalue w.r.t. the two other eigenvalues:

$$R_a = \frac{|\lambda_1|}{\sqrt{|\lambda_2 \lambda_3|}}. \tag{5.14}$$

For blob-like structures, R_a should be close to 1 since the three eigenvalues will be similar. If the structure has a preferred direction, the eigenvalue that is associated with this direction should be small. Elongatedness along the direction of the tubular structure is measured as ratio between the second eigenvalue and third eigenvalue. For tubular structures, the two eigenvalues should be similar, whereas platelike structures should have a much higher λ_3 compared to λ_2. Hence, elongatedness is defined by

$$R_b = \frac{|\lambda_2|}{|\lambda_3|}. \tag{5.15}$$

Finally, the contrast of the tubular structure is given by a norm of the Hessian matrix. The authors used the Froebenius norm for an $N \times N$ matrix

$$R_c = \sqrt{\sum_{i=1}^{N} \lambda_i^2}. \tag{5.16}$$

The filter result is then

$$V = \left[1 - \exp\left(-\frac{R_a^2}{2a^2}\right)\right] \exp\left(-\frac{R_b^2}{2b^2}\right) \left[1 - \exp\left(-\frac{R_c^2}{2c^2}\right)\right] \tag{5.17}$$

with a, b, and c being weighting factors to stress the importance of each of the components. In the original paper, the authors set the result to 0, if λ_2 or λ_3 were negative, since they were only concerned with finding bright vessels on dark background.

The vesselness filter can be combined with diffusion filtering for achieving a smooth signal along the vessels (Manniesing and Niessen 2005). The authors use the directional information from the Hessian to initiate anisotropic diffusion along the vessel direction.

The vesselness filter behaves poorly at vessel crossings or bifurcations since it can determine just one prominent direction. For determining an arbitrary number of directions, a scoring scheme can be introduced on vesselness responses on a sequence of filtered images, where each filter acts as a matched filter for a specific direction (Hannink et al. 2014).

5.5 SIFT Feature and SURF

The *scale-invariant feature transform* (SIFT) has been developed by Lowe (1999, 2004) and patented by the University of British Columbia. SIFT generates and uses features to detect and identify objects in images. Local features are identified and represented in a descriptor. Objects are identified by comparing expected feature configurations with all possible subsets of configurations from features detected in an image. The object is detected if a sufficiently large correspondence has been found. The method proceeds in several steps:

- key point generation,
- key point reduction,
- feature computation, and
- key point matching.

In *key point generation*, rotation- and scale-invariant features are generated by searching for local extrema of a multi-scale blob detector based on a multi-scale Difference of Gaussian (DoG, see previous section and Fig. 5.9). The detector is insensitive with respect to noise by determining an optimal smoothing for each scale. This is done by computing a sequence of Gaussian smoothings $S(\sigma_i)$ with standard deviations $\sigma_i = \sigma_1, \sigma_2, \ldots, \sigma_n$ and computing a sequence of DoG $D(\sigma_i) = S(\sigma_i) - S(\sigma_{i-1})$, $i = 2, n$ (see Fig. 5.10). A local extremum is maximum or minimum not only in scale but also along the range of different smoothing scales σ_i.

Fig. 5.9 Key point responses are computed at different scales of an image pyramid. Potential key points are local extrema in scale space

Fig. 5.10 Additionally to being an extremum in scale space, it also has to be an extremum for differently smoothing DoG filters

Key point generation will create numerous responses from noise and artifacts. In *key point reduction*, the contrast at local extrema is used to remove low-contrast blob locations (Lowe 1999).

Responses are also removed if contrast is high, but the localization is unstable. This is the case along edges where the average contrast may be high because the feature is prominent across the edge, but the localization accuracy is low because the feature strength varies little along the edge. The selection criterion to remove edge responses uses corner detection schemes such as local curvature or the determinant of the Hessian (see previous section). In Lowe (2004), an improved version of this step was presented where the feature response function is interpolated and attributes of the interpolated function are used to remove unstable feature locations.

In *feature computation*, orientation attributes are determined and stored for each remaining key point. A histogram of gradient directions is computed for locations in the neighborhood of a key point (see Fig. 5.11). Gradient directions are computed at the scale that was determined for the key point in the first step. The maximum of this histogram indicates the local key point orientation. A threshold of 80% of this maximum is set, and it is tested whether any other local maximum in the histogram exceeds this threshold. If it is the case, multiple key points with the same location but different orientations are created. The key point orientation is then used to compute rotation-invariant features. A local window at the feature scale is defined around the point location. Gradients within this window are weighted with a Gaussian. Then, local gradient histograms in subwindows around

Fig. 5.11 Key point orientation is computed from a histogram of binned gradient directions in the vicinity of the key point

Fig. 5.12 The feature vector for a key point consists of binned histograms of normalized, relative gradient directions with respect to the key point orientation. Hence, key points are represented by a kind of rotation-invariant texture features

the key point location are used as key point features (see Fig. 5.12). The features are organized in a feature vector which is then normalized to make the feature independent from intensity variation. Hence, the feature vector describes the relative gradient length distribution for different gradient directions in the vicinity of the key point.

Key point features can then be used for *matching* model key points with key points extracted from the image. Although the objective for developing the SIFT procedure was to identify objects by key features, its main application in medical image analysis is to support feature-based registration algorithms (for registration methods see Chap. 10). Registration finds a transformation to map two images of the same object onto each other. The reason for using SIFT features is that it can be assumed that this mapping will only be successful if a sufficiently large number of features correspond in the two images. A registration is particularly easy if this correspondence is given by sets of pairs of corresponding feature locations. Since most medical image registration problems are 3d, the SIFT features have been extended to 3d as well (Allaire et al. 2008).

A faster variant to SIFT is SURF [*speeded-up robust features* (Bay et al. 2006)]. It uses essentially the same mechanism, except for the fact that the slow convolutions in SIFT are replaced by faster approximations. The methodology is patented in the USA and is claimed by the author of not only being faster but also more robust than the SIFT features.

5.6 Binary Key Point Descriptor and Detectors

Key point matching plays an important role in tracking, registration, and matching tasks as their invariance w.r.t. several image acquisition-related artifacts makes them good candidates for object-specific, local features. Several variants of this strategy have been presented for fast and robust determination of key points and key point attributes.

BRIEF (binary robust independent elementary features) is a binary descriptor that can be computed given a key point (Calonder et al. 2010). The key point could be generated, e.g., from a corner detector such as SUSAN or FAST or it could be the key points from a SIFT computation where the descriptor computation is omitted.

Similar to local binary patterns (see Sect. 5.10), BRIEF produces binary features from comparing intensities at pixel locations. For BRIEF, these are locations p_i and q_i in an $S \times S$ neighborhood around key point k (see Fig. 5.13). For a given number N of pairs (p_i, q_i) of sample locations in this neighborhood, the binary feature b_i for each pair (p_i, q_i) is computed from smoothed intensities I_σ (Gaussian smoothing with standard deviation σ) as follows

$$b_i = \begin{cases} 1, & \text{if } I_\sigma(p_i) < I_\sigma(q_i) \\ 0, & \text{otherwise} \end{cases} \tag{5.18}$$

The features are then concatenated in a bit string of length $N/8$ bytes. The Hamming distance is used for matching key point locations. It measures the difference between two strings of equal size by counting the number of different

Fig. 5.13 The BRIEF descriptor is constructed from a set of pixel pairs in a predefined neighborhood around each key point. Each pair is represented by a bit in a bit string that is then interpreted as integer

characters at the same position in the string. In our case, it counts the number of positions i in two feature vectors $B_1 = (b_{1,1}, b_{1,2}, \ldots)$ and $B_2 = (b_{2,1}, b_{2,2}, \ldots)$ with $b_{1,i} \neq b_{2,i}$.

The amount of smoothing necessary for computing (5.18) depends on the specific class of images. For selecting the sample positions, Calonder et al. (2010) investigated several schemes and found that an isotropic random Gaussian distribution of point pairs with $\sigma^2 = \frac{1}{25}S^2$ for an $S \times S$ neighborhood produced the best result for their matching.

In its original version, BRIEF is not rotation-invariant. If two images are rotated with respect to each other, as it would often be the case if key point matching is used for registering two scenes, the matching quality sharply decreases. Hence, Rublee et al. (2011) combined an orientation-aware FAST corner detector (see Sect. 5.3) and used the orientation to steer the determination of intensity pairs for computing (Eq. 5.18). They called this key point detector and descriptor *ORB* (oriented FAST and rotated BRIEF).

The FAST key point detector is made orientation sensitive by computing the offset between the detected corner and the center of gravity of the intensity distribution in the neighborhood of the key point (see Fig. 5.14). Since FAST detects corners, the center of gravity should be offset of the key point location in the direction of the corner. The orientation information is then used to rotate the BRIEF feature locations.

Fig. 5.14 The orientation of a corner point (here detected by FAST) can be computed as vector between the corner point and the center of gravity in a local neighborhood. The neighborhood should be large enough to enable reliable orientation computation, but small enough as to not to include other corners

5.7 Superpixel

If features relate to the interior of regions rather than to edges and corners, these regions have to be computed prior to feature computation. Pixels belong to the same region if some homogeneity criterion applies to them. Hence, a necessary step for computing region-based features is a segmentation of the image to find the regions. Different to the segmentation methods discussed in Chaps. 6–10 that deal with the separation of medically relevant structures (organs, pathological processes, etc.) from the images, the segmentation only needs to extract large enough segments that enable computation of region features.

Computation of superpixels is a means to arrive at such regions. Each superpixel consists of a set of connected pixels from the original image. Superpixels are non-overlapping and space-filling. The motivation for the computation of superpixels is twofold:

- Stable separation of information from noise is often impossible at the pixel level. Hence, the true resolution of the image, i.e., the level of detail of the information that is represented by the image, is lower than the actual spatial resolution.
- The searched level of detail, such as segment membership, is lower than the image resolution. Regional features may vary across a segment, but this variation may be captured much better by few superpixels than by the pixels themselves.

Given these two assumptions, superpixels can be generated by a data-driven segmentation technique such as the watershed transform or normalized graph cuts (the two methods are described in Sects. 6.7 and 8.1.5, respectively). These two methods produce a grouping of pixels to superpixels but have a different behavior. The watershed transform uses only the region homogeneity as criterion and will produce large superpixels in homogeneous regions and small ones otherwise. The normalized graph cut algorithm also enforces similar sizes of the regions.

A good overview on methods to generate superpixels is (Achanta et al. 2012) in their presentation of the *SLIC* (simple linear iterative clustering) method as a fast and simple alternative. SLIC carries out a local clustering in a feature space of pixels where each pixel is attributed to its location and its intensity (or color, if color images are to be processed). Superpixel centers are initially seeded evenly over the image. The spacing between seeds depends on the desired number of superpixels. Then, pixels are assigned to the most similar superpixel center given features f_{SLIC} of the superpixel center and those of the pixel (see Fig. 5.15):

$$f_{\mathrm{SLIC}} = (\,\lambda x \quad \lambda y \quad I(x,y)\,)^{\mathrm{T}}, \tag{5.19}$$

Fig. 5.15 The number of SLIC superpixels depends on the number of seeds that are evenly distributed over the image. All other pixels are assigned to their most similar seed. Similarity depends on intensity and distance between the pixel in question and the seed

where the pixel's location (x, y) is weighted by λ against its intensity $I(x, y)$. Hence, distance measures intensity and spatial differences at the same time. Superpixel centers are then updated by averaging over all pixels that are assigned to this superpixel. The process is repeated until convergence (the authors noted that stopping the method after 10 iterations was sufficient in all their tests).

The algorithm is fast, since it searches for the closest superpixel center only in the vicinity of a pixel. Irrespective of the number of seeds, the number of superpixel centers to be searched always remains fixed. If the number of iterations is fixed as suggested by the authors, the computational costs increase linearly with the number of superpixels.

Some care has to be taken regarding the distance measure. Intensity range and spatial range may be different, and their relation changes, e.g., if the image resolution changes. Hence, the two types of features are normalized independently before using them as criteria in the distance measure. The parameter λ is used to enforce the regularity of the superpixels. Small values of λ result in irregularly shaped, homogeneous superpixels. Since post-processing, e.g., texture-based segmentation of superpixels, often profits from regularly shaped and similarly sized superpixels, the weight will often be a user-specified compromise between homogeneity and regularity (see Fig. 5.16 for an example).

Fig. 5.16 Result of applying the SLIC algorithm with the same number of seeds and varying values for λ (0.1, 1.0, 10.0, 100.0). High values for λ (*lower row*) produce very regularly shaped superpixels, but segment boundaries may not always coincide with superpixel boundaries (see *arrows*). If such high regularity is required, the number of seeds needs to be increased

5.8 MSER Features

Locations, such as the center of gravity, generated from *maximally stable extremal regions* (MSER) have been presented by Matas et al. (2002). The key concept of this approach is to separate an image into local homogeneous regions with maximum contrast. Opposed to SIFT feature and SURF, this detector defines key points based on intensity and not on gradients.

The definition for an MSER can be found in Matas et al. (2002). It can be described informally as follows:

- A region r_t is any connected set of scene elements $\mathbf{s}$ with intensity $f(\mathbf{s}) > t$, where t is an arbitrary threshold.
- An instability measure $s_\Delta(r_t, t)$ (see Fig. 5.17) rates the size change of r_t when the threshold is varied by some value Δ.

Fig. 5.17 The instability measure rates the size change of a region for some threshold variation against the region size. It is minimal for a maximally stable extremal region

$$s_\Delta(r_t, t) = |r_{t+\Delta} - r_{t-\Delta}|/|r_t|, \tag{5.20}$$

where $|\cdot|$ indicates computing the volume of the respective region.

- A region r_t is maximally stable if $s_\Delta(r_t, t)$ is minimum for t.

Centers of maximally stable extremal regions serve as key points, and features of the regions can be used as key point attributes. Compared to the computation of SIFT features, the generation of MSER features is very fast [$O(n\log n)$, with n being the number of scene elements in Matas et al. (2002)]. Several ways for an even more efficient computation have been proposed (Donoser and Bischof 2006a; Nistér and Stewénius 2008).

Although MSER has first been presented for 2d applications, it easily extends to 3d since the definition does not depend on the dimension of the scene. Maximally stable regions in 3d have been used for segmentation of medical images (Donoser and Bischof 2006b). Since MSER features have been applied in feature-based correspondence analysis for stereovision (Matas et al. 2002), they could be used for registration as well although we are currently not aware of such an application in medical image analysis. In Forssen and Lowe (2007), MSER were combined with SIFT features. SIFT features were computed at the locations of maximally stable extremal region and provided a richer description than the original features.

5.9 Local Shape Context

Features mentioned so far are meant to characterize key point locations at boundaries or in regions in an image. However, features may also be generated from sampling the boundary. Similar to the generalized Hough transform, *local shape*

context (Belongie et al. 2002) describes an object by boundary features that need not be—and in most case will not be—key points. The purpose of using local shape context is to be able to match two structures based on the shape context information. Local shape context is defined on boundary points that stem from sampling boundaries of an object of interest that are generated by an edge detection procedure such as the Canny edge detector. The boundary need not to be closed. However, edge detection of two similar objects should result in similar sets of boundary parts. Sampling may be arbitrary, but in the absence of further knowledge, it should generate point locations distributed evenly over the boundary parts.

A structure is then represented by a sequence of points $P = \{p_1, \ldots, p_N\}$. For each point, shape context is computed by comparing its position to the position of all other points. It results in a set of direction vectors $D_i = \{d_{i,1}, \ldots, d_{i,N-1}\}$, where each direction is a vector from p_i to some other point p_j, $j \neq i$. The directions are binned in a log-polar coordinate system. In Belongie et al. (2002), five distance ranges and twelve sectors were used to determine the bin size. The histogram now gives the number of occurrences of other points with respect to the p_i per bin (see Fig. 5.18). The feature is translation-invariant as it defines the context with respect to the location of p_i. It can be made scale-invariant by normalizing the distances with the average distance of all points p_j to p_i. If necessary, it can be made

Fig. 5.18 Local shape context is represented by 2d histograms for points that contain frequency of occurrences of other points. The histogram is binned in a log-polar grid. It can be seen that shape contexts of similar point locations result in similar histograms

rotation-invariant by computing the histogram based on a coordinate system that is fixed to the tangent direction at p_i.

Given the shape context for every point p_i of P, a match to points q_j of some other figure Q is computed by finding a circular shift $j = \pi(i)$ such that the sum of local similarities $S(\pi) = \Sigma_{i=0,N} S(i, \pi(i))$ is maximal. The local similarity measure of Belongie et al. (2002) compares histograms h_i and $h_{\pi(i)}$ using the X^2 metric

$$S(i, \pi(i)) = \frac{1}{2} \sum_{k=1}^{K} \frac{\left(h_i(k) - h_{\pi(i)}(k)\right)^2}{h_i(k) + h_{\pi(i)}(k)}. \tag{5.21}$$

In a later publication (Moni et al. 2005), the authors used a slightly different feature descriptor where similarity was computed using the L_2-norm. For each bin, an average direction was computed from all directions falling into this bin. The components of each average direction (x- and y-component for a 2d image) are concatenated to form the feature vector.

The optimal permutation can be found in polynomial time using the Hungarian method (Papdimitriou and Stieglitz 1982) or, as in Belongie et al. (2002), the faster method of Jonker and Volgenant (1987).

Shape matching using local shape context can be made robust with respect to outliers if a certain number of dummy points are introduced in the shape representation. A dummy point of P has fixed matching costs for all points of Q (and vice versa for dummy points of Q). If similarity of all points in P to a point q_j in Q is worse than the matching cost for the dummy point, then q_j is matched to the next free dummy point. Hence, a fixed number of points in P and Q are allowed to have no counterpart.

Given the match, a transformation between the two scenes that contain P and Q can be interpolated so that not only P and Q but also the two scenes are registered with respect to each other (for further details on registration see Chap. 10).

5.10 Texture

All of the features presented so far mostly relate to the boundary of objects. However, the kind of key points and key point descriptors discussed in Sects. 5.5 and 5.6 may also describe properties of the interior of objects that may be used to discriminate by means of textures. Textures are difficult to define, as they are a kind of microstructure in a structured world (samples from the Brodatz textures in Fig. 5.19 show some of the variety of different textures). Just where an entity stops to be a structure and starts to be a microstructure is a matter of viewpoint. The fabric of a curtain, for instance, may be called a texture, but at close inspection, a thread of the fabric may be a structure itself having a texture from its constituting filaments.

Fig. 5.19 Samples from the Brodatz textures. It can be seen that it will be difficult to define feature sets that capture the characteristics of all these textures

Textures have two properties in common:

1. A texture has a repeating pattern. It may be an exact repetition as in manufactured surfaces or a repetition with a random component as in many textures of natural objects.
2. Computation of texture features requires a texture-specific, minimum size of the window in which all scene elements belong to the same texture.

Deterministic texture patterns on an object are usually found on manufactured objects. Representation of such textures is by their constituting, deterministic texture elements (texels). In medical imaging applications, textures on objects are caused by the structure of organs imaged. Constituting elements certainly exist, but they are usually much smaller than the imaging resolution. Visible textures are caused from interaction between the measured signal and these structures. The textures can be described statistically in the spatial or the frequency domain.

It has been shown that three properties are necessary for differentiating adjacent regions by texture:

1. Texture *orientation* can best be explained in frequency space. Textures possessing one or more pronounced orientations will have most of the energy in frequency space in sectors along those orientations.

2. Texture *periodicity* relates to the smallest region necessary to represent all properties of a texture.
3. Texture *complexity* relates to the composition of it. A texture which is composed of basic elements of the same size is less complex than one that consists of basic elements of different sizes. A texture which consists of white noise is less complex than one consisting of colored noise.

Textures of distinguishable, adjacent segments need to be different in at least one of the three aspects.

Segmentation by texture plays a minor role in medical image analysis, as most properties from differently textured organs are too subtle to be exploited. Examples for exemptions are as follows:

- MRI, where artifacts from tissue-specific magnetic field inhomogeneities contribute to noise.
- Ultrasound, where tissue-specific refraction artifacts influence the signal.
- Photographic images of the skin, where, e.g., scars exhibit macroscopic textures.
- Microscopic images, where cell textures can be spatially resolved.

A multitude of different texture measures exists, but in view of the rather limited use of texture as a feature we will only list some that are exemplary for different ways of measuring statistical properties of texture:

- *Haralick's features* of the co-occurrence matrix measure second-order statistical attributes of the gray level co-occurrence matrix (GLCM). GLCM features have been presented in Haralick et al. (1973). Later experiments confirmed that second-order statistical features are important for object discrimination (Julesz 1975, 1981).
- *Spectral features* are created from integrating amplitudes over a partitioning of the texture representation in the frequency domain. They are a direct measure of orientation and periodicity of a texture in the frequency or spatial domain (e.g., He and Wang 1991). Gabór filters are a specific variant of this as they combine spatial and frequency characteristics by a bank of differently oriented, windowed frequency transforms (Grigorescu et al. 2002).
- *Law's filter masks* are a set of orthogonal convolution kernels to measure periodicity and orientation of textures in the spatial domain.
- *Local binary patterns* (LBPs) compute for each pixel p in a given window the relative intensity w.r.t. neighboring pixels in an 8-neighborhood. Pixels in the neighborhood are ordered along a circle for the pixel in question (see Fig. 5.20). The local feature is a binary vector with eight entries (stored in a byte). Each entry is set to 1 if the respective pixel is brighter than p and it is set to 0 otherwise. The feature vector consists of all features of all pixel p in the texture window. This is the base version of LBP. A review discussing the many different variants of LBP can be found in Huang et al. (2011) on uses in face recognition.

Fig. 5.20 LBPs are generated by comparing the intensity at some location with intensities of its neighbors. If it is greater, a "1" is entered into the bit string; otherwise, a "0" is entered. With 8-neighbors, 256 different feature values are possible. The histogram of features of a texture patch constitutes the feature vector of the texture. Variants of this method use different neighborhoods, remove irrelevant features, or use differential features

Attributes of textures can be determined in many ways, and some research has been devoted to estimate the quality of such features (e.g., du Buf et al. (1990) who presented a framework for testing textures). A comparative study for determining the discriminative power of 15 types of texture measures including the above-mentioned three measures in Wagner (1999), which was carried out on six very different types of textures ranging from the Brodatz textures to textures in microscopic cell images, found no significant difference between the measures. It was found, however, that different measures excelled in the six classes. This corroborates a strategy of carefully selecting a texture measure based on the characteristics of texture to be differentiated.

Textures should not be computed across object boundaries if the goal is the determination of object characteristics. If the object is not segmented, it may be useful to split the image into superpixels and compute the texture in the superpixels.

5.11 Histogram of Oriented Gradients

Although shape context does not attempt to find key points, it still requires parts of object boundaries to be found. This may be unwanted if a boundary is difficult to detect because of insufficient image contrast or artifacts. The *histogram of gradients* (HOG) (Dalal and Triggs 2005) is a feature detector that does not rely on boundaries as it samples a dense gradient texture map and uses this to generate the features. Hence, HOG features are always computed from a gridded region of interest that is assumed to be mostly occupied by the structure to be analyzed.

A HOG cell is defined at each gridline intersection. The authors suggest two types of cells. R-HOG cells have a rectangular shape, while C-HOG cells have a circular shape. Gradients are computed for each pixel. Each pixel within a cell votes for a direction in a binned histogram of gradient directions. The vote is weighted with the strength of the gradient.

Fig. 5.21 HOG features computed from a microscopic image. The directional histogram in each HOG cell is given by the intensity of a line in the respective direction. Local orientation preferences are clearly recognizable

In order to make the response insensitive to intensity variation, cells are grouped into larger blocks. The gradient strength is the normalized with the average gradient in this block.

The authors used the HOG *features* for accurately detecting pedestrians in images using cell sizes of 4×4 to 12×12 pixels and block sizes of 1×1 to 4×4 cells with best results with block sizes between 2×2 and 3×3 and cell sizes between 6×6 and 8×8. They reported that the method was most successful when using the simplest gradient operator and no smoothing of the data, although this may be different when using HOG features on medical images with their higher noise levels.

LBP features have been combined with HOG features for pedestrian detection, and its superiority w.r.t. HOG features alone has been shown by Wang et al. (2009).

HOG features, either the original or variants, have been used for classification and detection of objects in medical and biological images (Fig. 5.21 shows an example of computing HOG features in a microscopic image). Examples are the lesion detector in lung CT of Song et al. (2012) who also combine LBP features with HOG features for classification or Sanroma et al. (2014) who use HOG features to classify the similarity of atlases to an image to be segmented in a multi-atlas segmentation approach.

5.12 Saliency and Gist

Saliency and gist are image features which are different to the features discussed above that pertain to structures in the image. Image features are not immediately useful for the analysis of objects. However, as they characterize the image, they

may be used to guide image analysis in application such as a feature-based search in an image database.

Saliency is an attribute that guides attention of human vision to certain locations in the image. Focusing on certain parts in an image is an integral part of the perception process of a human operator who searches, recognizes, and categorizes structures in an image as it allows the operator to single out subsets of the image and focus on attributes of this subset. A salient location in an image is a region where features differ significantly from features in the vicinity. Since saliency is a concept of biologically inspired computer vision, features are those perceived at early stages of visual perception. Examples are image intensity, image color, and local orientation. A biologically plausible method for computing saliency has been presented by Itti et al. (1998). Several other methods exist as well.

Computer vision techniques use saliency in the same manner for quickly directing attention to subparts of an image which may contain objects that are searched. Such rapid scene analysis is seldom the goal when working with medical images, but distribution, extent, and properties of salient locations characterize images in a way that can be useful for search and comparison tasks between images. This is even more so considering the fact that occlusion problems—which change such configuration—do not occur if images are 3d.

The *gist* of an image is an overall categorization of an image (e.g., by characterizing an image as a CT image of a head depicting a subdural hemorrhage). In human vision, deriving the gist of an image allows to pick a constraining category for further analysis. Being able to compute the gist of an image can substantially speed up image analysis as it reduces the number of possible explanations for depicted objects. Since gist summarizes information from all parts of the image to come up with an overall categorization, methods to compute gist are integrations from feature values everywhere in the image. Computer-based methods to compute gist sometimes assign to rather broad categories (Torralba et al. 2003), but combining gist and saliency computation has been shown to provide differentiation between quite similar but different categories (Sigian and Itti 2007).

Using gist features will probably be of little use for analyzing medical images where the gist is usually known. As with the case of saliency, gist features may help in the automatic categorization of images in a data basis of medical images. Here, meta-information may describe the gist of the image only in the context of the clinical or research question that gave rise to the production of the image.

5.13 Concluding Remarks

Feature detection is an intermediate step for generating semantics from an image. Computing such features changes attribute representation from an image-centered to an object-centered reference system. Since features are defined on image attributes, this has to be taken with a grain of salt, however. Feature computation often

assumes that object attributes can be defined based on local image attributes in an invariant fashion. In reality, object attributes visible in an image may be quite different for the same type of object.

Feature computation always consists of two parts. First, attributes are computed, and then, a criterion is applied to find locations where attributes are assumed to be relevant. The latter should be applied with care as well. Since the definition of feature attributes will not exactly separate object features from image artifacts, setting a strong relevancy criterion might remove necessary object features. This can be critical if the subsequent steps are automatic so that input to a later analysis module is not verified by user input (a search for an object boundary may fail if the edge-tracking step removed too many of the potential boundary parts).

5.14 Exercises

- What is the purpose of the two thresholds used in edge tracking by the Canny edge detector? What guidelines should be followed when setting the thresholds?
- Name situations or problems for which edge tracking is a useful preprocessing step.
- What would happen if the size of the accumulator cells of the Hough transform is made smaller? Under what circumstance would it be an advantage?
- Describe the steps of finding lines by the Hough transform.
- Explain strategies to speed up the Hough transform. Why is the Hough transform generally advantageous in a real-time environment?
- What are the underlying assumptions for a corner detector?
- What features of the Harris matrix are used for corner detection? How are they used for corner detection?
- What is the difference of features used by the SUSAN corner detector compared to the Harris corner detector?
- What filters can be used for blob detection?
- How is the blob detector made scale-invariant?
- Describe an algorithmic sketch for detecting blobs of unknown size in an image. What kind of prior information is needed to parametrize the method?
- Describe the steps that lead to SIFT key points.
- How are the SIFT features made rotation-invariant?
- What kind of features represent a SIFT key point?
- How are the SIFT made (approximately) invariant to illumination changes? Under what circumstance is invariance w.r.t. intensity changes no longer given? Why?
- How is the BRIEF descriptor computed? What is the difference to descriptors generated from SIFT or SURF?
- What is the difference between superpixels created by the watershed transform and those created by normalized graph cuts?
- Describe the different steps for computing superpixels by SLIC.

- Describe the condition that needs to be met for a point being the center of an MSER (maximally stable extremal region).
- Why can it be expected that MSER centers do not (not even approximately) coincide with SIFT key points?
- What are potential applications of using SIFT key points or MSER centers in a medical image analysis method? What would be their purpose?
- What are the differences between SIFT features and shape context features?
- How can HOG features be applied to detect an object in an image?
- Why should texture features not be computed across object boundaries? What can be done to avoid this if the image is not segmented?
- What are the similarities between BRIEF and LBP? What are the differences?
- Explain the strategy by which shape context is made insensitive to feature points that cannot be matched.

References

Achanta R, Shaji A, Smith K, Lucchi A, Fua P, Süsstrunk S (2012) SLIC superpixels compared to state-of-the-art superpixel methods. IEEE Trans Pattern Anal Mach Intell 34(11):2274–2282

Allaire S, Kim JJ, Breen SL, Jaffray DA, Pekar V (2008) Full orientation invariance and improved feature selectivity of 3D SIFT with application to medical image analysis. In: IEEE computer vision and pattern recognition workshops, CVPRW '08, pp 1–8

Basu M (2002) Gaussian-based edge-detection methods—a survey. IEEE Trans Syst Man Cybern Part C Appl Rev 32(3):252–259

Bay H, Tuytelaars T, van Gool L (2006) SURF: speeded up robust features. In: Proceedings of Europe conference on computer vision—ECCV 2006, LNCS, vol 3951, pp 404–417

Belongie S, Malik J, Puzicha J (2002) Shape matching and object recognition using shape contexts. IEEE Trans Pattern Anal Mach Intell 24(4):509–521

Calonder M, Lepetit V, Strecha C, Fua P (2010) Brief: binary robust independent elementary features. In: Proceedings of European conference on Computer vision—ECCV, pp 778–792

Canny J (1986) A computational approach to edge detection. IEEE Trans Pattern Anal Mach Intell 8(6):679–698

Coppini G, Diciotti S, Falchini M, Villari N, Valli G (2003) Neural networks for computer-aided diagnosis: detection of lung nodules in chest radiograms. IEEE Trans Inf Technol Biomed 7(4):344–357

Czerwinski RN, Jones DL, O'Brien WD (1993) An approach to boundary detection in ultrasound imaging. In: Proceedings on IEEE ultrasonics symposium, Baltimore

Czerwinski RN, Jones DL, O'Brien WD (1994) Edge detection in ultrasound speckle noise. In: Proceedings on IEEE international conference on image processing, Austin, pp 304–308

Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE computer society conference computer vision and pattern recognition (CVPR'05), 1:886–893

Donoser M, Bischof H (2006a) Efficient maximally stable extremal region (MSER) tracking. In: Proceedings of conference on computer vision and pattern recognition (CVPR), pp 553–560

Donoser M, Bischof H (2006b) 3d segmentation by maximally stable volumes (MSVs). In: Proceedings of international conference on pattern recognition (ICPR), pp 63–66

du Buf JMH, Kardan M, Spann M (1990) Texture feature performance for image segmentation. Pattern Recogn 23(3–4):291–309

Forssen PE, Lowe DG (2007) Shape descriptors for maximally stable extremal regions. In: IEEE 11th International conference on computer vision (ICCV), pp 1–8

Frangi AF, Niessen WJ, Vincken KL, Viergever MA (1998). Multiscale vessel enhancement filtering. In: Medical image computing and computer-assisted intervention—MICCAI'98, pp 130–137

Grigorescu SE, Petkov N, Kruizinga P (2002) Comparison of texture features based on Gabor filters. IEEE Trans Image Process 11(10):1160–1167

Hannink J, Duits R, Bekkers E (2014) Crossing-preserving multi-scale vesselness. In: Medical image computing and computer-assisted intervention—MICCAI 2014, pp 603–610

Haralick RM, Shanmugam K, Dinstein KI (1973) Textural features for image classification. IEEE Trans Systems Man Cybernet 3(6):610–621

Harris C, Stephens M (1988) A combined corner and edge detector. In: Alvey vision conference, pp 147–151

He DC, Wang L (1991) Texture features based on texture spectrum. Pattern Recogn 24(5):391–399

Hough PVC (1962) Method and means for recognizing complex pattern. U.S. patent no. 3.069.654

Huang D, Shan C, Ardabilian M, Wang Y, Chen L (2011) Local binary patterns and its application to facial image analysis: a survey. IEEE Trans Syst Man Cybern Part C (Appl Rev) 41(6):765–781

Itti L, Koch C, Niebur E (1998) A model of saliency-based visual attention for rapid scene analysis. IEEE Trans Pattern Recogn Mach Intell 20(11):1254–1259

Jonker R, Volgenant A (1987) A shortest augmenting path algorithm for dense and sparse linear assignment problems. Computing 38:325–340

Julesz B (1975) Experiments in the visual perception of texture. Sci Am 232(4):34–43

Julesz B (1981) Textons, the elements of texture perception, and their interactions. Nature 290:91–97

Lindeberg T (1993) Detecting salient blob-like image structures and their scales with a scale-space primal sketch: a method for focus-of-attention. Intl J Comput Vis 11(3):283–318

Lowe DG (1999) Object recognition from local scale-invariant features. Intl Conf Comput Vis ICCV 1999(2):1150–1157

Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Intl J Comput Vis 60(2):91–110

Mallat S, Zhong S (1992) Characterization of signals from multiscale edges. IEEE Trans Pattern Anal Mach Intell 14(7)

Manniesing R, Niessen W (2005) Multiscale vessel enhancing diffusion in CT angiography noise filtering. Inf Process Med Imaging 2005:138–149

Matas J, Chum O, Urba M, Pajdla T (2002). Robust wide baseline stereo from maximally stable extremal regions. In: Proceeding of British machine vision conference, pp 384–396

Meer P, Georgescu B (2001) Edge detection with embedded confidence. IEEE Trans Pattern Anal Mach Intell 23(12):1351–1365

Moni G, Belongie S, Malik J (2005) Efficient shape matching using shape contexts. IEEE Trans Pattern Anal Mach Intell 27(11):1832–1837

Nistér D, Stewénius H (2008) Linear time maximally stable extremal regions. In: Proceedings of Europe conference on computer vision ECCV 2008, LNCS vol 5303, pp 183–196

Papadimitriou C, Stieglitz K (1982) Combinatorial optimization: algorithms and complexity. Prentice-Hall

Papari G, Petkov N (2011) Edge and line oriented contour detection: state of the art. Image Vis Comput 29:79–103

Roerdink JBTM, Meijster A (2001) The watershed transform: definitions, algorithms and parallelization strategies. Fundam Informaticae 41:187–228

Rosten E, Drummond T (2006) Machine learning for high-speed corner detection. In: Computer vision—ECCV 2006, pp 430–443

Rublee E, Rabaud V, Konolige K, Bradski G (2011) ORB: an efficient alternative to SIFT or SURF. In: IEEE international conference on computer vision (ICCV 2011), pp 2564–2571

Sanroma G, Wu G, Gao Y, Shen D (2014) Learning to rank atlases for multiple-atlas segmentation. IEEE Trans Med Imaging 33(10):1939–1953

Schilham AMR, van Ginneken B, Loog M (2003) Multi-scale nodule detection in chest radiographs. In: Medical image computing and computer-assisted intervention—MICCAI 2003, LNCS 2878, pp 602–609

Schmid C, Mohr R, Bauckhage C (2000) Evaluation of interest point detectors. Intl J Comput Vis 37(2):151–172

Siagian C, Itti L (2007) Rapid biologically-inspired scene classification using features shared with visual attention. IEEE Trans Pattern Anal Mach Intell 27(2):300–312

Smith SM, Brady JM (1997) SUSAN—a new approach to low level image processing. Intl J Comput Vis 23(1):45–78

Song L, Liu X, Ma L, Zhou C, Zhao X, Zhao Y (2012). Using HOG-LBP features and MMP learning to recognize imaging signs of lung lesions. In: 25th international symposium on computer-based medical systems (CBMS), 2012, pp 1–4

Torralba A, Murphy KP, Freeman WT, Rubin MA (2003) Context-based vision system for place and object recognition. In: IEEE international conference on computer vision (ICCV), pp 1023–1029

Tuytelaars T, Mikolajczyk K (2007) Local invariant feature detectors: a survey. Found Trends in Comput Graph Vis 3(3):177–280

Wagner T (1999) Texture analysis. In: Jähne B, Haussecker H, Geisser P (eds) Handbook of computer vision and applications, Academic Press, pp 275–308

Wang X, Han TX, Yan S (2009) An HOG-LBP human detector with partial occlusion handling. IEEE 12th Intl Conf Computer Vision ICCV2009, pp 32–39

Williams DJ, Shah M (1990) Edge contours using multiple scales. Comput Vis Graph Image Process 51:256–274

Yu Y, Acton ST (2002) Speckle reducing anisotropic diffusion. IEEE Trans Image Process 11:1260–1270

Segmentation: Principles and Basic Techniques

6

Abstract

The purpose of image segmentation is to generate pixel agglomerations from an image that constitute parts of depicted objects. In medical imaging, segmentation often refers to the delineation of specific structures. Hence, it includes parts of classification as well. Segmentation strategies in medical imaging combine data knowledge with domain knowledge to arrive at the result. Data knowledge refers to assumptions about continuity, homogeneity, and local smoothness of image features within segments. Domain knowledge represents information about the objects to be delineated. In this chapter, basic strategies for integrating the two types of knowledge into the segmentation process will be discussed. We will also describe basic segmentation methods that are popular in medical image analysis.

Concepts, Notions, and Definitions Introduced in this Chapter

- Data features: intensity and texture;
- The role of homogeneity, smoothness and continuity in segmentation, using object localization and appearance as domain knowledge;
- The role of interaction;
- Basic segmentation techniques: thresholding, region merging techniques, region growing, watershed trans-form, live wire.

Quantitative analysis of a medical image requires objects or object features in the image to be identified and delineated. First, the image is segmented into regions that are the possible candidates of objects. This is followed by assigning meaning to these regions. For analyzing a digital photograph, this *segmentation* task would group pixels to regions that may belong to (parts of) objects based on the attributes of these regions. Hence, segmentation of images is similar to creating phonemes in speech or detecting syllables in a written text as it creates basic semantic entities from images.

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_6

Fig. 6.1 Finding a general solution for a segmentation task would be very difficult even if only the houses depicted in the four examples would need to be segmented correctly

However, it is difficult to apply domain knowledge about objects in an image to segmentation. The purpose of segmentation is to create semantic entities in the first place. Ideally, object-independent criteria help to partition an image into segments. After segmentation, every pixel is assigned to exactly one segment since every location in an image carries just one meaning.

There is a problem with this. Appearances of objects in some image—consider a picture of a landscape with a house in the foreground, some trees, and mountains in the background—may be very different within and between object classes (see, e.g., Fig. 6.1). A house may consist of visible surfaces of homogeneous color requiring a homogeneity criterion on color. A tree may be characterized by its natural texture and color, which requires texture to be included in the criterion. The depiction of mountains in the background is mostly characterized by intensity and the faded color of distant objects. A uniform segmentation criterion for all of these objects will be difficult to find. Regions found by a simple criterion such as homogeneous color may segment some regions correctly but may fail in other regions.

For the *segmentation of medical images*, the situation is somewhat less grave, as a medical image represents (ideally) the measurement of a diagnostically relevant entity that is measured in the same way everywhere in the body. Consider CT, for instance, where X-ray attenuation is measured on the normalized Hounsfield scale. Attenuation should not vary with location but only with density and atomic number.

External influences such as shading or object-specific signal degradation still add an object- or location-dependent component to the measured and reconstructed signal. Furthermore, measured values are not unique for a specific object class. Hence, data-driven segmentation will almost always result in a collection of regions that in some cases will separate an object of interest into several segments and in other cases will fuse different objects into a single segment.

6.1 Segmentation Strategies

There are several ways to deal with missing information without sacrificing the assumption that a low-level segmentation criterion is valid everywhere in the image:

- *Foreground segmentation* (Fig. 6.2) focuses on a single object in the image. Segmentation criteria create a good partitioning of foreground objects, whereas

Fig. 6.2 If only the liver needs to be separated from the CT, segmentation would be successful irrespective of any errors in regions outside the liver. Foreground segmentation that only incorporates information as to what differentiates liver from all other tissues would be sufficient

the quality of partitioning the background is irrelevant. Later analysis is carried out solely on foreground segments. The strategy requires some model knowledge to be applied after segmentation for separating foreground segments from the background. A simple way to introduce model knowledge would be to let the user point out foreground segments.

- *Hierarchical segmentation* (Fig. 6.3) applies a multi-resolution concept for gradual refinement. A first segmentation creates segments that are smaller than the smallest object. It is assumed that a common criterion (in most cases a homogeneity criterion) can be found at this scale. The result is sometimes called

Fig. 6.3 In hierarchical segmentation, the first segmentation step provides little more than superpixels. Two examples from segmenting (**a**) can be seen in (**b**) and (**c**), where slightly different homogeneity constraints lead to different sizes. The next hierarchy level attempts to combine superpixels with higher semantic units

over-segmentation. At the next level, some of these segments are merged into larger segments according to domain knowledge about object appearance. Successful application of this strategy requires that meaningful segments can be defined by a common criterion at a single but unknown scale. This scale is found by analyzing the levels of the segmentation hierarchy.

- *Multilayer segmentation* (Fig. 6.4) is another multi-resolution technique. It is assumed that a common segmentation criterion exists, but that its scale may vary throughout the image (like a structured texture of objects in a photograph of which the scale varies with distance of the object to the camera). Segmentation is carried out at different scales producing layers of segments. Later analysis will have to estimate local scales and patch segments according to it. It is more general than the previous strategy as the scale of a criterion often varies for different structures in an image. Analysis of segments is more difficult because an appropriate scale for every segment has to be established independently from other segments.

In medical imaging, different semantics stem from the intentional choice of the acquisition technique. The pixel value in a medical image is much more directly related to the diagnostic question than a pixel value of a photograph to the meaning of some outdoor scene. The acquisition technique has been chosen for the very reason that it is known to offer insight into some diagnostic question. This domain knowledge can be incorporated into the segmentation process. It becomes especially apparent for slice images. In some images, such as X-ray and CT, function value and membership to an organ class are related (see Fig. 6.5 and the Hounsfield Table 2.1 in Chap. 2). A segmentation criterion on pixel values may be sufficient for assigning class membership to a segment. Hence, segmentation and classification may mix in medical image analysis.

The search for occurrences of a specific structure by segmentation causes foreground segmentation to be more frequent in medical image analysis than in other image analysis tasks. Since it may involve a detection task, a model-driven

Fig. 6.4 In multilayer segmentation, segmentations at different levels of resolution are created and evaluated in parallel. In this example, the boundary between grey matter and CSF is captured quite well in the low-resolution segmentation, while the boundary between fat and bone is captured better in the high-resolution image

Fig. 6.5 In medical imaging, there is often a much more direct relationship between pixel intensity and semantics compared to photographic pictures (although it is by no means a unique mapping)

approach then discriminates the structure from background. Model knowledge may be integrated into the algorithm or supplemented interactively. Popular segmentation techniques, such as the various region growing techniques, the application of implicit or explicit active contours and surfaces, or active shape models, use such a model-driven approach.

The use of domain knowledge from the acquisition technique and the reduction in segmentation to the search of foreground objects helps to solve the segmentation problem to an extent which exceeds support of segmentation in general image processing. Furthermore, segmentation is often carried out in slice images with the consequence that occlusion has not to be dealt with. On the other hand, computer-assisted analysis of medical images is virtually impossible without segmentation.

6.2 Data Knowledge

Continuity in space and continuity in time are the two main data properties that are used for segmentation. An observable object is assumed to stand out in an image by some homogeneous intensity or texture in a region. Segmentation based on spatial continuity partitions a 2d or 3d image such that homogeneity within segments is larger than between adjacent segments. It assumes that the course of the segment boundary does not change abruptly (see, e.g., Fig. 6.6a).

Fig. 6.6 Continuity in space is used, for instance, when the fairly smooth course of the boundary of the hip bone is segmented using a spline curve (**a**), or when predicting vessel locations in the sequence of slices in (**b**) by using the location from the previous slice for initialization

Temporal continuity can be used in the same fashion by treating time as fourth dimension. Often, temporal continuity is exploited by computing a segmentation result at one time step and using it to constrain or initialize the segmentation at the next time step.

The strategy is sometimes applied as well for segmenting a sequence of 2d images of a 3d scene. Segmentation is carried out in a 2d slice. The result is then propagated into the next slice to initialize the segmentation in this slice (see Fig. 6.6b).

Propagation of segmentation constraints along the time axis or a spatial axis makes the result dependent on the initial segmentation. A developer should make sure that this initialization is as good as possible. This may be achieved

- by letting the user flip through slices or time steps for selecting the best starting point. Segmentation is then carried out from this initialization in both directions along the time or spatial axis.
- by starting the segmentation from several initialization points and selecting the best-rated segmentation from this. It requires a quality measure for segmentation results.
- by letting the process go several times back and forth through the data improving the segmentation until a stable state of segmentation is reached.

Carrying out n-dimensional segmentation by imposing continuity constraints between adjacent segmentations in $(n - 1)$-dimensional space reduces subsequent segmentation to a registration task if a one-to-one correspondence exists between segments in adjacent segmentations (see Fig. 6.6b). The task is simpler than registration in general, since continuity assumptions predict that corresponding segments are close to each other and have similar shape. Such situation often arises in 4d segmentation, where objects neither vanish nor are created between time steps. It is sometimes true for 3d segmentation from 2d slices as well, although this is not guaranteed. However, even if topology changes between slices, segments on the

previous slice can be used to constrain possible locations of an unknown number of segments with the same label in the current slice.

6.2.1 Homogeneity of Intensity

Spatial and temporal continuities can be characterized by homogeneous local appearance. This refers to the simplest case to homogeneity of intensity which is given by the intensity variance within a segment. Other approximations such as computing the difference between the brightest and the darkest pixels of a segment can be used as well.

Pixel or voxel intensities of a structure of interest often vary little throughout the segment. In consequence, intensity-based segmentation schemes are quite popular. However, a number of artefacts have to be accounted for. The best-known is noise. Segmentation in a noisy environment is more difficult than it seems, since noise reduction capabilities of the human visual system are quite effective. A human may see a visible difference between neighboring segments long before homogeneity-based segmentation would separate segments (see Fig. 6.7).

Noise is often modeled as Gaussian with zero mean value and a variance according to the SNR. Noise may be reduced during pre-processing, but this may cause small details to get removed (see Fig. 6.8). Noise reduction may be included into segmentation as well, e.g., by applying a multi-resolution strategy. A multi-resolution approach such as a *Gaussian pyramid* creates a sequence of images at different resolutions by repeated low-pass filtering and subsampling. Segmentation is carried out at a low resolution because of the noise-reducing effects of low-pass filter. Low-pass filtering also reduces boundary detail and may remove small segments. Hence, segmentation of a low-resolution image is transferred to the next higher resolution by applying the expand operation of the Gaussian pyramid to the segmentation (which is essentially an interpolation). The expanded result constrains

Fig. 6.7 Although contrast between tissues is good, the image contains a substantial amount of noise that can be seen in a plot of intensities of one line in the image

Fig. 6.8 Noise reduction prior to segmentation may cause loss of detail which is most critical if small structures are to be extracted by segmentation

segmentation at the higher resolution in much the same way than the spatial constraints imposed in sequential segmentation of temporal sequences (see Fig. 6.9).

Shading is another artefact that sometimes influences intensity-based segmentation. It usually stems from image acquisition. Examples are shading in an MRI due to field inhomogeneities or shading in ultrasound images due to acoustic shadows. If shading effects are less pronounced than segment homogeneity, it can be removed during pre-processing. Multiple sclerosis lesion segmentation in MRI presented in Chap. 1, for instance, estimates shading based on a segmentation of white matter. White matter is highly contrasted against the background and thus easy to segment without shading correction. As white matter should have constant

Fig. 6.9 Applying a multi-resolution strategy to segmentation: segmentation in a low-resolution image is used as initialization on the next higher resolution

intensity, any intensity variation is attributed to shading, which is then extrapolated for every location in the image. Alternatively, intensity-based segmentation may be carried out locally if intensity variation per unit distance is smaller than the contrast between segments. The image is partitioned into subregions, and segmentation is carried out separately for every subregion.

Some of the problems from shading can be resolved when resorting to boundary criteria instead of region criteria. If a segment is contrasted against neighboring segments, high-intensity gradients indicate boundaries between segments. Boundaries are either localized by tracking zero crossings of the second derivative of the intensity function or by computing local maxima of the gradient length. Gradient-based segmentation is simplest if the gradient length, i.e., the intensity difference between adjacent segments, is approximately equal throughout the image or if all non-zero gradient lengths are caused by segment boundaries. The former is rarely the case and the latter is never the case, since images are not noise-free. Hence, many edge-based segmentation schemes include a significance threshold on the gradient strength which needs to be exceeded for a pixel to be accepted as part of the segment boundary. It may remove parts of the true boundary that fall below the significance threshold. This information has to be added as part of the domain knowledge for creating a segmentation (e.g., by requiring closed boundaries or by enforcing a shape prior).

Compared to region-based segmentation, a major disadvantage of edge-based segmentation is the sensitivity to noise. Hence, noise reduction as part of gradient computation is mandatory. Using a multi-resolution strategy, such as in region-based segmentation, is possible. It requires registering boundaries from expanded images with edges in the original image.

6.2.2 Homogeneity of Texture

Continuity may also refer to the texture of structures. Texture features, described in Sect. 5.10, for a pixel can be computed from a region around this pixel. Their reliability depends on the size of this region. This is apparent for Haralick's GLCM features. The measure of co-occurrence in some neighborhood of a pixel just estimates the parameters of a probability distribution from samples. Its reliability depends on the number of samples and, hence, on the size of the neighborhood. The dependency on neighborhood size exists for other texture measures as well because the texture of a pixel is always defined by intensity variation in some neighborhood. Hence, regions from which textures are computed should be large. However, texture-based segmentation implicitly assumes that an image contains at least two different textures to be separated. Texture computation at unknown segment boundaries becomes unreliable if pixels of more than one segment contribute to the texture measure. This would point to selecting the smallest possible neighborhood for texture computation.

There are four strategies to deal with this problem as follows:

1. Texture measures are chosen that require only a small number of pixels for computing reliable features. Unreliability at segment boundaries is neglected or may even be used as characteristic for finding a boundary. An example is (Lorigo et al. 1998) who segmented MR knee images by a geodesic active contour, which uses local variance as a textural feature.
2. Segmentation is carried out iteratively. At initialization, the neighborhood around a pixel of a given texture is assumed to contain only pixels of that texture. In a second step, regions for texture computation are refined using edge information from the first step. An example is the work of Pitiot et al. (2002) on MR images. They use features of the co-occurrence matrix and estimate segment boundaries using Gabòr filters. Another example for iterative segmentation is the work of Reyes-Aldasoro and Bhalerao (2003) who used spectral features and combined it with a multi-resolution approach.
3. Segmentation is carried out in a multi-resolution framework. Low-reliability results computed at a low resolution initialize the segmentation at higher resolution. It requires that textures can be computed at different levels of resolution and that resolution-dependent reliability can be computed. An example is the work of Muzzolini et al. (1993) on texture-based segmentation of ultrasound images.
4. Texture-based segmentation is preceded by an initial segmentation step based on homogeneity, see e.g., Cheng et al. (2013). A strong homogeneity criterion is used that results in an over-segmentation. The resulting image consists of many segments containing only few pixels. The segments are called *superpixels*. Assuming that each superpixel lies in exactly one of the searched segments, a texture measure for each superpixel can be computed from all its pixels. Image-driven segmentation methods such as the watershed transform (Sect. 6.7) or normalized graph cuts (Sect. 8.1.4) may be used to produce the superpixels.

It has already been mentioned that texture-based segmentation includes components of classification. Texture-based classification without segmentation is more widespread in medical image analysis than texture-based segmentation. In classification, a sample region with homogeneous texture is already computed. The problem of computing regions with homogeneous texture does not exist.

6.3 Domain Knowledge About the Objects

Detecting an object requires additional information about what to segment. While the underlying model in data-driven segmentation is simple (homogeneity of intensity or texture), further knowledge may be entered through an explicit description of properties of boundaries in a segmentation or about a foreground object. Domain knowledge consists of information (see Fig. 6.10) as given below:

Fig. 6.10 Different kinds of domain knowledge can be used to guide an analysis process

1. The appearance of boundaries between segments;
2. The location of an object within an image;
3. The orientation and size of the object with respect to the scanner coordinate system;
4. Spatial relationships (location, orientation or relative sizes) of the object with respect to other objects in the image; and
5. The shape and appearance of the object.

Most of the attributes relate to foreground segmentation where an object shall be separated from the background. Just what information is used depends on the specific segmentation task. In order to be useful, domain knowledge for describing a foreground object should be

- *discriminative*, i.e., object and background must have different properties with respect to the feature;
- *generalizable*, i.e., the property must be true for all possible instances of the object of interest in all possible images within the field of intended use; and
- efficiently *computable* with sufficient reliability.

Efficiency of computation is not mandatory but domain knowledge should not contain unnecessary information. If, for instance, bone segmentation in CT images using a simple threshold for discriminating bone from the less-dense background materials delivers sufficient accuracy, it is not efficient to require an additional shape model for describing a specific type of bone.

6.3.1 Representing Domain Knowledge

Domain knowledge may be introduced to a segmentation method via an adjustable model that is included in the segmentation method or interactively at runtime. Using incorporated domain knowledge allows predicting the behavior of the method for a given data set.

If a method operates fully automatically based on data and a priori information, failure on part of the model indicates that the domain knowledge does not conform to the reality in the data. Validation will determine whether and to what extent segmentation errors may be attributed to incomplete or wrong assumptions about the objects. It will thus support improving a segmentation method by improving the descriptive power of the incorporated domain knowledge. This is not true if knowledge is introduced interactively, since human behavior cannot be exactly predicted.

Validation will be discussed further in Chap. 13, but it should be noted that the consequences of validation will be different if domain knowledge is supplied interactively as compared to automatic segmentation using an incorporated model.

Each of the five types of domain knowledge listed above can be represented by

- a *parameterized description* such as the representation of a segment orientation by a vector in scanner space;
- a *sampled description* such as the representation of shape by a sequence of boundary points; or
- an *implicit description* such as the representation of a shape by a function on image domain describing its boundary.

A parameterized description reduces a property to a set of parameters. The parameters can usually be represented with arbitrary precision. Other aspects corresponding to the same property are excluded. Describing the shape of an approximately oval object by center as well as length and orientation of its two major axes, for instance, fails to represent any deviation of the true object from the elliptical shape.

Properties described by samples require a sufficiently high sampling rate. The almost oval shape in the example above could be described by a finite number of boundary points. The number and distribution of these points determines the accuracy with which actual shapes are described.

Properties described implicitly as function of the image domain require a number of base functions on this domain which can be combined to describe the object boundary. The description accuracy depends on the number and complexity of base functions.

Property values of any of the descriptions may be combined in a property vector describing expected features of a segment.

6.3.2 Variability of Model Attributes

Domain knowledge about objects in images describes the properties of a class of objects. Incorporating it into a segmentation method has to include known variation among instances of a class. Variation is specified by a range of permissible property values in the property vector. Information about acceptable variance within a class may be obtained from expert information or from training.

The reliability of expert information depends on the expert and on the kind of knowledge. Many implicit knowledge representations use simple assumptions about the local smoothness of object boundaries which can be readily assumed for most structures occurring in medical imaging. Apart from few parameters governing the influence of intensities or their derivatives on the model, no further information has to be introduced prior to applying a model-driven segmentation technique. The strategy may have limited success in the case of artefacts or an insufficient relationship between image information and object appearance. Nonetheless, a very popular segmentation paradigm is based on implicit representation of local domain knowledge. An implicit representation, called *level set representation*, integrates many aspects of low-level knowledge about segments in a common concept and will be detailed in Chap. 9 on active contours and surfaces.

High-level knowledge characterizing class membership and variation in shape and appearance of an object is more difficult to acquire. An expert (a radiologist, a surgeon) certainly has a vast experience about possible appearances of structures which are of interest to him or her. This experience by far surpasses the capabilities of computer-based representation of it. However, the expert uses the knowledge as needed and may never have to express it a priori in an unambiguous fashion.

The type of domain knowledge also influences the ease with which a description may be created. It is, for instance, easy to establish the average size and position of the heart in x-ray CT. It is much more difficult to describe its expected shape and shape variation.

Resorting to training solves some of the problems mentioned above. Training samples are usually taken from the same kind of images that are to be segmented later. Transfer from human knowledge to the representation of domain knowledge in a segmentation method is taking place when a human operator prepares the training samples.

Training is time-consuming and may also suffer from lack of samples. If, e.g., the property vector of the parameterized description of oval shapes contains just five elements (x- and y-coordinates of the center, angle of the long axis with respect to the x-axis, lengths of long and short axes), training expected value, and variance in this 5D space from, say, 25 samples would not be very reliable. Asserting reliability gets worse if the size of the property vector increases.

Automated training may facilitate the acquisition of training samples. Still, a sufficient number of images must exist for training. Furthermore, an analysis module is needed in order to retrieve property values from these images. Developing the analysis module may require so much domain knowledge about objects to be segmented that it solves the segmentation problem altogether. Automatic

training from segmentation with an adjustable model component can still make sense because it may be used as a learning component, which causes performance to improve with continuous training.

Segmentation employs such descriptive a priori information by producing segments whose properties resemble properties as predicted by the model. Probability is sometimes computed using a Bayesian formulation where the domain knowledge is part of the a priori model. Often, probability is used much more informal and segments are chosen with properties that are in some defined way closest to an average appearance predicted by the model.

6.3.3 The Use of Interaction

Interactive incorporation of domain knowledge has the advantage of being flexible because an expert user decides about necessary input at segmentation time. If the system responds immediately to interaction, knowledge incorporation at run-time also delivers cues about potential input errors.

Interactive input may happen directly on the image or it may be an adjustment of segmentation parameters. The former is attractive to most users as it allows them to directly interact with the medium that they want to segment. However, interaction on the image can be tedious if

- it is required for a prolonged time;
- interaction devices are inappropriate; and
- the relation between interaction and consequence for the result is not intuitive.

This may affect the quality of segmentation or reduce the acceptance of the method. If tiresome interaction cannot be avoided, small input errors need to be tolerated and corrected automatically while large errors should be detected. Depicting structures instead of requiring their delineation is to be preferred because depiction is less sensitive to input errors than delineation. If delineation is necessary, cues from the image—often delineation should follow high gradient curves—and from the underlying assumption about the depicted object—boundaries are often smooth and not ragged—should be used to detect and correct potential errors.

Interaction by parameter setting is often more robust with respect to erroneous interaction, but effects on the segmentation are less intuitive. A way to cope with the semantic gap between abstract parameter setting and its effect is to produce an immediate result. The user learns dependencies between parameter settings and segmentation result from immediate experience.

In any case, an explanation of the mode of operation of a parameter should be available on request by the user. Seemingly complex behavior of parameter adjustment should be described by example and be available to the user as well (e.g., in a documentation or an online tutorial).

Interaction in a segmentation algorithm may come in one of the five varieties (see Fig. 6.11):

Fig. 6.11 Five different kinds of interaction that may happen during image analysis

1. In a priori *parameterization*, the user is requested to enter the parameter values of an adjustable segmentation algorithm. Once parameters are set—such as indicating the location of an object of interest or setting thresholds of the expected intensity range of the object—the segmentation algorithm proceeds automatically until it produces the result.
2. Through segmentation *guidance*, the user supports the segmentation until a satisfactory result is generated. An example would be the entering of boundary points in a user-guided delineation of an organ boundary.
3. *Feedback* happens after segmentation has been carried out. The user varies parameters or changes the way of guidance in order to improve the quality of the result. An example is the threshold variation in threshold segmentation. Feedback requires re-computation of the segmentation.
4. *Correction* changes the segmentation result according to user expectation. An example would be the removal of parts of a segment that do not belong to the object of interest. Correction usually implies positive confirmation.
5. *Confirmation* is the process by which the user accepts or rejects a result.

Parameterization is often required. Adapting parameters will be successful in applications which follow the underlying assumptions of the segmentation method and which can be characterized by a proper parameter setting. The limitation of parameterization is that interactively added knowledge can only specify but not extend or counteract knowledge implemented in the segmentation method.

Using guiding interaction is more powerful but at the cost of reduced control about the result. A very general and very simple segmentation tool using guidance would draw a boundary of a foreground object according to interactive mouse input from the user. This method would be able to segment any object that the user recognizes in the image. It also puts the burden of retrieving and using domain knowledge on the user.

Guidance may also counteract an inappropriate segmentation model. One could imagine, for instance, guiding an edge-tracking method in an ultrasound image. Since most of the high gradients stem from noise and artefacts, the user will constantly have to correct tracking errors which are due to the false hypothesis of high gradients being mainly caused by object boundaries (see Fig. 6.12).

Fig. 6.12 It depends on the image (and information and artefacts contained in it) whether a certain interaction component is useful and efficient or not

However, a similar edge tracking could be very suitable for some CT images with its high contrast. Tracking will only occasionally loose the boundary. It may be easier to let the user enter the information on the spot instead of providing a model that covers those rare cases. Hence, guidance can be an efficient and effective means to support segmentation if its use is justifiable. It should not be an excuse for a poor model of domain knowledge.

Feedback differs from guidance in that it does not interfere with the segmentation process itself but offers the user the chance to repeat the segmentation with refined parameters. It is less susceptible to misuse in case of an inappropriate segmentation method because feedback cannot change the segmentation model.[1] Advantageous use of feedback requires the expert to understand the underlying a priori model of the segmentation method. The expert, who knows how a successful segmentation should look like, can then decide based on the current results how parameterization should be changed.

Receiving feedback can be a frustrating experience, however, if the change in parameter values leads to unexpected behavior because the consequences on the segmentation result are not intuitive. Sometimes, understanding about proper parameterization can be gained by observing the behavior of a segmentation algorithm even if correspondence between segmentation result and parameter setting is not straightforward. Gaining experience from feedback of the method improves with the speed with which the results are computed.

Interaction through correction is another powerful tool because discrepancies between expected and computed results can be removed. However, correcting segmentation results hints at deficiencies of incorporated domain knowledge. An example for interaction through correction is the combination of thresholding with subsequent correction because object intensity does not differ everywhere from the background intensity.

[1]If the model is mainly based on its parameterization and requires frequent feedback, it again would indicate a poor design of the segmentation model. This should not happen in practice, because in such case the success of segmentation would vary substantially with different parameter settings making segmentation time-consuming and awkward to use.

Use of correction should be limited to cases where inclusion of the missing domain knowledge is inefficient because costs exceed benefits. Correction that is required regularly for creating a valid segmentation indicates an inappropriate segmentation method.

Interaction by confirmation does not extend the bandwidth of applications of a given segmentation method directly. It may be used, however, to adapt a segmentation method to some application. Every time that a user accepts or rejects a result, new information is given. Some segmentation algorithms are able to profit from confirmation by adapting automatic parameter setting accordingly. Confirmation also assures that the final decision as to whether a segmentation result is correct belongs to the expert user.

6.4 Interactive Segmentation

The algorithmically simplest way of segmentation is to completely rely on user guidance to outline the boundary of a foreground structure. Employing some input device such as a mouse, a trackball or an electronic stylus on a graphical tablet, the user traces the boundary on a rendition of a 2d scene. The method can be applied to arbitrary 2d scenes assuming that the user has all the necessary domain knowledge. Interactive segment delineation by an expert often provides a reference for a semi-automatic or automatic segmentation scheme. The underlying assumption is that a domain expert provides proper model knowledge.

The main problem of interactive segmentation is that it can neither be guaranteed that a user really possesses all the necessary information for a perfect segmentation of an image nor that this information is applied correctly during segmentation. Interactive segmentation as a validation tool hence requires some modeling of human error when segmenting an image. As no analytical model of this kind exists, it is estimated from an assumed error distribution based on input from several different observers.

If interactive delineation is used in routine segmentation, these accuracy problems may occur as well. Delineating boundaries in data is tedious and time-consuming. The performance of a user depends on individual experience, on stress, on tiredness, and on similar factors. Hence, interaction is often supported by low-level techniques borrowed from image enhancement or data-driven segmentation. Examples are as follows:

- Boundaries to be delineated are highlighted by displaying the intensity gradient instead of the original image. Edge contrast is increased by noise removal techniques such as edge-preserving smoothing.
- Just a small number of boundary points are provided by the user (see Fig. 6.13). They are connected by automatically generated line segments (straight lines or curves).

Fig. 6.13 Interactive delineation is supported if the user only has to click at few boundary points and if positions selected by user input are automatically corrected by displacing them toward the position with locally highest gradient

- User input is corrected automatically by offsetting the boundary orthogonal to its tangent towards the nearest highest gradient (see Fig. 6.13).
- The user is allowed to correct a delineated boundary. This is most efficient if the boundary does not consist of a sequence of neighboring pixels but of a sequence of points connected by curve segments. User correction of a single point then induces a corresponding correction of its two incident line segments.

Segmenting 3d data sets using interactive delineation requires successive delineation on a sequence of 2d slices. The process can be simplified by projecting results from the previous slice to the current slice and letting the user correct it.

6.5 Thresholding

Medical images are made on purpose. Often, the object of interest stands out by having intensity values higher or lower than the background. Hence, thresholding is an often-used tool in segmenting medical images. A survey of early thresholding methods can be found in Sahoo et al. (1988). Segmentation s of an image f by threshold t at a pixel or voxel $\mathbf{v}$ is given by:

$$s(\mathbf{v}) = \begin{cases} 1, & \text{if } f(\mathbf{v}) > t \\ 0, & \text{otherwise} \end{cases} \tag{6.1}$$

Thresholding produces a separation of image pixels into foreground pixels ($s = 1$) and background pixels ($s = 0$). If more than one threshold is chosen, separation into several different regions is possible as well (see Fig. 6.14).

Fig. 6.14 Thresholding can be done interactively and separates the image into different regions. Valleys in the histogram indicate potentially useful threshold values

Sometimes, the result already separates the object of interest from background, e.g., if thresholding was used on contrast-enhanced images such as digital subtraction images. If not all foreground regions are part of the object of interest, a connected component analysis (CCA) follows, which labels all foreground regions. Regions forming the object of interest are then identified by the user.

Thresholding can be done interactively. The user selects a threshold and segmented structures are displayed. The threshold is varied until the segmented structure meets the expectations of the user. This kind of interaction should be avoided if it is difficult to display the segmentation result (e.g., if data is 3d or 4d) or if it can be expected that the result will include too much user-caused variation.

Automatic thresholding techniques attempt to use the underlying assumptions of thresholding. Since it is assumed that foreground pixels have a different value than background pixels, the histogram of the image should have two peaks (a so-called *bimodal histogram*). The threshold should be somewhere between the two peaks. Since the distributions forming a bimodal histogram usually overlap, the threshold is found at a location between the peaks that produces the smallest number of wrong label assignments.

However, noise will cause the histogram to have several local maxima and minima. Otsu (1978) presented an iterative procedure to find an optimal threshold for a normalized histogram h. It produces a threshold t between two classes (0 and 1) so that the variance within each class (the within-class scatter q_w) is small and the variance between classes (the between-class scatter q_b) is large by maximizing the ratio $\lambda(t) = q_b(t)/q_w(t)$ (see Fig. 6.15). The within-class scatter $q_w(t)$ is computed from the grey-level variance $\sigma_0^2(t)$ for all pixels with grey value below t (the

Fig. 6.15 Otsu's thresholding methods seeks to find a threshold t that minimizes within-class scatter for the two classes characterized by expected values μ_0, μ_1 and variances σ_0^2 and σ_1^2 and between-class scatter characterized by the deviation of the class means from the overall expected value μ_T. Moving the threshold t toward the valley in this example would decrease within-class scatter since the high values just on the right of t are closer to the new μ_0 than to μ_1. It also reduces between-class scatter, since μ_0 would move by a higher amount towards μ_T than μ_1 moves away from it

0-class), and the grey-level variance $\sigma_1^2(t)$ from all pixels with grey value greater than or equal to t (the 1-class). The variance of the 0-class is

$$\sigma_0^2(t) = \sum_{g=0}^{t-1} p(g)(g - \mu_0(t))^2 \tag{6.2}$$

where μ_0 is the estimate of the expected value of all grey values below t.

The variance of the 1-class is computed similarly from all grey values greater or equal to t. The within-class scatter is then

$$q_w(t) = P_0(t) \cdot \sigma_0^2(t) + P_1(t) \cdot \sigma_1^2(t), \tag{6.3}$$

where $P(0) = \sum_{g=0}^{t-1} p(g)$ and $P(1) = 1 - P(0)$ are the a priori probabilities of a pixel to belong to one of the two classes.

The between-class scatter is then the weighted sum of variances of the expected values of the 0-class and the 1-class from the expected value μ_T of the histogram:

$$q_b(t) = P_0(t) \cdot (\mu_0(t) - \mu_T)^2 + P_1(t) \cdot (\mu_1(t) - \mu_T)^2 \tag{6.4}$$

The optimal threshold is found by exhaustive search through all possible values of t. The author also notes that the above-mentioned quality criterion is equivalent to two other criteria:

$$\kappa(t) = \sigma_T^2/q_w(t) \quad \text{and} \quad \eta(t) = q_b(t)/\sigma_T^2 \tag{6.5}$$

He suggested to maximize $\eta(t)$. Since the variance σ^2 of the histogram does not depend on t it only requires computation of the between-class scatter. The method has been extended to separate more than two classes by applying it recursively on the foreground or background pixels (Cheriet et al. 1998).

A similar approach uses the fuzzy set theory to iteratively determine thresholds (Tobias and Seara 2002). The authors define two subsets of grey values that belong to the foreground and the background and a third subset of grey values between the two of which membership is initially unknown. Grey values from this set are assigned fuzzy memberships that depend on the histogram of foreground and background pixels and on that of the non-classified region. The final threshold is then found at the location where the two fuzziness functions for foreground and background intersect.

Finding the threshold minimum can be difficult, if the number of foreground pixels is small compared to the number of background pixels. The peak for foreground pixels is then in the range of noise variation and may be impossible to be determined. A solution for finding the threshold in such case has been given by the algorithm of Zack et al. (1977) (see Fig. 6.16):

- Compute a line between the two local maxima of the two distributions.
- Compute the distance d$L(h)$ for each entry h of the histogram to this line.
- Convolve d_L with a smoothing kernel to remove effects from noise.
- The threshold is $h_{\max}$ for which $d_L(h_{\max})$ is maximum.

The algorithm of Zack is a good, albeit heuristic approximation of an optimal solution under a Bayesian assumption. Under this assumption, the histogram would be the combination of two probability distributions $P_f(\mathbf{v}|g)$ for $\mathbf{v}$ being a foreground pixel given its grey value g and $P_b(\boldsymbol{v}|g)$ for it being a background pixel weighted by their a priori probabilities (i.e., the two a posteriori distributions). The optimal threshold producing the smallest number of false classifications would then be the

Fig. 6.16 Algorithm of Zack defines the threshold as the location where the distance of a curve through the histogram from a line through the two maxima is maximum

intersection of the two distributions. This is not always the minimum of the histogram. If the variance of one of the two distributions is smaller than that of the other, the algorithm of Zack moves the threshold toward the former which is qualitatively similar to what happens for the optimal threshold under the Bayesian assumption.

Thresholding may not work, although the contrast between foreground and background is seemingly high, if underlying shading distorts the image. This happens in MR imaging when the coil used for inducing the imaging signal does not produce a constant signal within the scene (see Hou 2006; Vovk et al. 2007 for reviews). Microscopy is another field where shading complicates threshold-based segmentation (see Tomazevic et al. 2002 for a review). If shading correction is impossible during image generation, it can be done retrospectively if knowledge exists at least in parts of the image to separate shading effects from other intensity variations.

Another adverse influence stems from noise. Mainly at segment boundaries, many of the pixels may be assigned to the wrong segment. A simple way to deal with this is a morphological post-processing (opening or closing) to remove artefacts. Alternatively, Rosenfeld and Smith (1981) presented a variant of relaxation labeling which takes into account a measure of certainty of segment membership when correcting segment labels.

Initial reliabilities p and q are computed for each pixel $\mathbf{v}$ belonging to foreground or background based on some threshold t (see Fig. 6.17):

$$p(\mathbf{v}) = \frac{1}{2} + \frac{1}{2}\frac{g(\mathbf{v}) - t}{g_{\max} - t} \tag{6.6}$$

$$q(\mathbf{v}) = \frac{1}{2} + \frac{1}{2}\frac{t - g(\mathbf{v})}{t - g_{\min}} \tag{6.7}$$

Reliability values are iteratively updated based on foreground and background membership of surrounding elements $\mathbf{w}$. At some iteration k, support values S_f and S_b for $\mathbf{v}$ belonging to the foreground or background are then computed from the eight neighbor pixels $\mathbf{w} \in Nb(\mathbf{v})$

Fig. 6.17 For relaxation labeling, initial foreground and background reliabilities are computed by evaluating the distance of each pixel intensity from the threshold

Fig. 6.18 Example for computing support values in an 8-neighborhood

$$S_f(\mathbf{v}) = \frac{1}{8} \sum_{\mathbf{w} \in Nb(\mathbf{v})} C\left[p^k(\mathbf{v}) > q^k(\mathbf{w})\right] p^k(\mathbf{v}) - C(p^k(\mathbf{v}) < q^k(\mathbf{w})) q^k(\mathbf{w}) \tag{6.8}$$

$$S_b(\mathbf{v}) = \frac{1}{8} \sum_{\mathbf{w} \in Nb(\mathbf{v})} -C\left[p^k(\mathbf{v}) > q^k(\mathbf{w})\right] p^k(\mathbf{v}) + C(p^k(\mathbf{v}) < q^k(\mathbf{w})) q^k(\mathbf{w}), \tag{6.9}$$

where $C()$ is an indicator function that is 1 if the expression in the brackets is true and 0 otherwise (see Fig. 6.18).

Given the support from neighboring pixels, the segment membership reliabilities are now updated:

$$p^{k+1}(\mathbf{v}) = \frac{p^k(\mathbf{v})\left[1 + S_f^k(\mathbf{v})\right]}{p^k(\mathbf{v})\left[1 + S_f^k(\mathbf{v})\right] + q^k(\mathbf{v})\left[1 + S_b^k(\mathbf{v})\right]} \tag{6.10}$$

$$q^{k+1}(\mathbf{v}) = \frac{q^k(\mathbf{v})\left[1 + S_b^k(\mathbf{v})\right]}{p^k(\mathbf{v})\left[1 + S_f^k(\mathbf{v})\right] + q^k(\mathbf{v})\left[1 + S_b^k(\mathbf{v})\right]} \tag{6.11}$$

This process continues until change in reliability values falls below some pre-specified minimum.

Thresholding is more difficult, if multi-channel functions are to be segmented, since it requires specification of a region in multi-dimensional feature space. For more than two channels, the feature space needs to be projected on 2d space. Thresholds are defined interactively in several different projections. Backprojection of threshold boundaries from the different 2d spaces to the original feature space produces the multi-dimensional region boundary. Interacting with such a system for correcting initial guesses of thresholds can be very demanding. Each projection restricts the set of definable threshold boundaries. Finding good thresholds will require manipulation in the projections as well as manipulation of the projection directions.[2]

[2]Thresholding is essentially a classification in feature space as it finds a decision boundary. If the feature space is multi-dimensional, the user should resort to classification techniques that are described in Chaps. 7 and 12.

Thresholding is a simple technique which is fast, easy to implement, and easy to understand. It may be extended to separate several different classes in images with multi-modal histograms. Finding thresholds automatically may become unstable when it becomes difficult to find reliable criteria for differentiating between true and false local maxima in the histogram.

6.6 Homogeneity-Based Segmentation

Segmentation based on local intensity homogeneity does not require an absolute threshold but a local variance criterion that is valid for all pixels within a segment. The goal is to separate an image into the smallest number of segments, so that for each of the segments this criterion is fulfilled. Examples for homogeneity criteria are the variance of pixel values, maximum deviation between pixel intensities, or the probability that all pixels belong to the same probability distribution.

Two basic segmentation algorithms for this are region merging and the split-and-merge algorithm. Both are not guaranteed to produce an optimal result, as they only ensure the homogeneity criterion and not that the smallest number of segments is found.

The following are the steps for computing segmentation by *region merging* (see Fig. 6.19):

- Initially, each pixel is considered to be its own region.
- Map regions to a region adjacency graph (RAG) in which nodes represent regions and adjacent regions are connected by an edge.
- Compute the homogeneity value for each edge for a region that consists of the two regions connected by the edge.
- As long as there exists at least one edge of which the homogeneity value fulfils the homogeneity criterion

  – Merge the two most similar regions
  – Update the RAG accordingly

Fig. 6.19 Schematic view at the region merging process. At each iteration, the two most similar regions are merged until such merging would violate the similarity criterion for pixels belonging to the same region

The result is segmentation where each region fulfils the homogeneity criterion. It is not necessarily optimal because the greedy strategy always merges the most similar regions without considering the influence of this merge with respect to adjacent regions. Region merging should be implemented using efficient data structures for accessing RAG nodes and for sorting elements after a merge step. Otherwise, the method will be slow because the naïve approach has a $O(N^2)$ computation cost with N being the initial number of regions (which can be very large if for instance 3D data sets are segmented). There are a number of strategies to get around this:

- Compute an over-segmented image on which region merging is to be carried out so that N is small.
- Use the fact that only few changes in the sorting order take place when two regions are merged and use an appropriate data structure.
- Change the selection order in a way that does not require a new sort after each merge.

All of these strategies have been used. Using an already partially segmented image is, for instance, done when combining region merging with a prior region-splitting process by the *split-and-merge algorithm*. It starts with the complete image being a single region and keeps splitting the image until each region fulfils the homogeneity criterion. The split part has the following steps (see Fig. 6.20):

- Initially, the complete image is a single region
- As long as a region exists that does not fulfil the homogeneity criterion

Fig. 6.20 During the first phase of the split-and-merge algorithm, regions are split into subregions until each subregion fulfils the homogeneity criterion. The split procedure is documented in a quadtree

- Split this region in four quarters (for a 2d image) or eight subvolumes (for a 3d volume).
- Document the split in an appropriate data structure (quadtree or octtree).

If no further regions need to be split, the current segmentation is turned into an RAG on which a region merging is carried out to merge regions that were accidentally split. Even when using naïve representations and sorting techniques, the combination with the split step will reduce computation time substantially as splitting should result in much fewer regions than there are pixels or voxels in the image.

For computing initial regions, other segmentation techniques can be used as well. An example is the watershed transformation (WST) that is discussed in the next section. Although the WST criteria are different from the homogeneity criteria discussed here, the over-segmentation, which is usually the result of applying WST, results in homogeneous segments. Combining this with a merging step, hence, is more efficient than just using region merging alone.

In Haris et al. (1998), the method is sped up further by using a priority queue for keeping the ordered list of RAG edges. The gain in speed results from the fact that merging requires removing and adding only few edges from the list instead of having to sort the whole list again. Another way to increase the performance of the merging process is to avoid re-sorting altogether (Nock and Nielsen 2004). This is an approximate solution which is possible based on an external model about pixels belonging to the same region. The predicate for this criterion is computed and ordered only once and region merging proceeds by this order.

6.7 The Watershed Transform: Computing Zero Crossings

Intensity edges defining segment boundaries are somewhat complementary to homogeneity of intensity defining segments. There is a difference, however. Intensity edges may still be discernible and can be used for separating segments even if intensity variation in shading makes definition of a homogeneity criterion impossible. On the other hand, relying on edge information is inherently less robust with respect to noise, since computing derivatives of the intensity information also enhances noise. Hence, many edge-based segmentation schemes require that noise reduction takes place either in a pre-processing step or as part of the edge-based segmentation method.

The *watershed transform* (WST) is a popular way to use edge information as criterion to separate segments [an excellent review of WST techniques can be found in Roerdink and Meijster (2000)]. If the WST is carried out on the gradient lengths of the intensity gradient, it is equal to defining segment boundaries by the zero crossings of the Laplacian of the intensity function. For twice differentiable functions on a real domain, the locations of the zero crossings are closed, non-overlapping boundaries. For images defined on a discrete domain, this is only approximately true but it is a very nice parameter-free way of image segmentation

with the appeal that zero crossings have been shown to be the important features for analysis by human vision (Marr and Hildreth 1980). The watershed transform to compute the segments is defined as follows:

- The scene is treated as a landscape in which function values (i.e., the gradient length) represent height;
- Each local minimum in this landscape is a basin; and
- Watersheds in this landscape are boundaries in the terrain which separate regions that drain into different basins.

Given this description, an image will decompose into many segments as there are basins. If basins are local minima of the gradient length, every local minimum, maximum, or saddle point in the original intensity image will be the center of a segment. If the image contains noise, it will produce a severe over-segmentation. Using the WST for segmentation, therefore, requires either noise reduction before applying the WST, or noise reduction included into the WST, or intentional use as pre-segmentation to be further processed by some other process that merges WST segments.

Computing the watershed transform can be done by flooding where basins are treated as sources (see Fig. 6.21, and the sketch of the algorithm is shown in Fig. 6.22).

If noise in the image is not dealt with during pre-processing and if the number of segments shall be reduced, the WST can be applied in a hierarchical fashion (Beucher 1994). In this case, the segments of the first WST are superpixels for another WST. Gradients can be estimated by computing average intensity differences to neighboring segments weighted by the size of the neighboring segments. The hierarchical WST reduces the number of segments, but it also increases the size of the watershed (super) pixels. It can be avoided when watersheds are not pixels but pixel boundaries. The waterfall algorithm of Beucher does exactly this. Different basins are merged if their watersheds are lower than those to surrounding basins. The hierarchy of merging basins can be used to assign a saliency measure to the segments (Najman and Schmitt 1996). Saliency is higher when a merge happens later.

Fig. 6.21 Watershed transform can be carried out by flooding the scene from sources at local minima in the image

```
set the initial flood level for an image f to l=min_v (f(v))
while l<max_v(f(v)) do
    Detect all pixel v_l that are newly flooded at level l
    if a pixel v_l is connected to pixels which all have a segment label L then
            segment L is extended by v_l .
    if a pixel v_l is not connected to any pixel that has a segment label then
            this is the first pixel of a new segment: label it L_new.
    if a pixel v_l is connected to at least two pixels with different labels then
            label the pixel W (for watershed)
    l = l+1
end_while
```

Fig. 6.22 Sketch of the flooding algorithm for the watershed transform

It is not guaranteed that a hierarchical WST produces more meaningful segments than the original WST. The hierarchical WST transforms the segmentation into scale space, and the underlying assumption is that there is an optimal scale for the complete image. In reality, however, there is certainly an optimal scale for objects or parts of objects, but this scale varies within the image. Hence, for each object the optimal segment is found on a different scale. Unless the existence of an optimal scale can be predicted a priori, the only way to deal with this is to include expert knowledge into the segmentation that comprises information about the appropriate scale.

The watershed transform can be combined with user interaction by providing a picking component (Meyer and Beucher 1990; Vincent 1993). Over-segmentation from standard WST arises because the underlying model for segments (each local minimum in the map of gradient lengths is a segment) does not describe what is wanted, i.e., the separation of organs in a medical image. The *marker-based WST* (mWST) adds information about objects to be segmented by replacing local minima as source for flooding by pre-specified marker positions (see Fig. 6.23).

At least one marker needs to be placed into any foreground or background region to be differentiated. The watershed transform still increases the water level. However, regions that are not flooded from a marker position receive a provisional label "unknown." If a region with this label merges with a marker-labeled region, it receives the label from the latter. By providing markers, the number of segments is pre-defined (it cannot be larger than the number of markers). A region may be represented by more than one marker if they have the same label. Marker positions can be set manually or may be computed by a detection process (e.g., Grau et al. (2004) where marker positions are computed from skeletonized probability maps).

Fig. 6.23 Marker-based WST proceeds in a similar fashion than the original WST, except for the fact that flooding occurs only from marker positions. Regions that are not connected to a marker-labeled region receive a provisional label "unknown"

Fig. 6.24 Simple region growing algorithm

```
Algorithm Region_growing_segmentation (xseed)
begin
        Label_field[] = not_visited
        Region_grow(xseed)
end

Algorithm Region_grow(x)
begin
   if label_field[x] = not_visited and hom(f(x)) then
        begin
                label_field[x] = visited
                for i=1,nbs do
                        region_grow(neighbor(x,i))
        end
end
```

6.8 Seeded Regions

Region growing combines data-driven constraints with interactively added domain knowledge (Zucker 1976). The user points out an object that needs to be separated from the background by specifying a seed point in the object. A homogeneity criterion characterizes the appearance of the region to be segmented. Region growing detects all pixels or voxels that can be reached from the seed point by paths of adjacent pixels or voxels for which the homogeneity criterion applies.

Pointing out a structure in an image is much easier than delineating its boundary. It requires the user to recognize at least one location that surely belongs to the foreground object. The underlying assumption is that the homogeneity criterion is independent of the seed point location so that different seed point locations deliver the same result (this assumption needs to be tested in a later validation).

Fig. 6.25 Finding the best homogeneity criterion is often done by trial and error and can be difficult in region growing

In Fig. 6.24, a sketch of the region growing algorithm is given that is started on an image $f(\mathbf{x})$ with seed point "xseed" and homogeneity predicate "hom ($f(\mathbf{x})$)". The function "neighbor ($\mathbf{x}$,i)" is assumed to return the ith neighbor of $\mathbf{x}$.

Homogeneity needs to be computable for a single pixel or voxel. It is often the variance of the intensity function $f(\mathbf{x})$ around some expected value. This corresponds to the assumption that the foreground object can be separated by a lower threshold and an upper threshold from the background.

It can be challenging to define the homogeneity criterion. Often, it is assumed that the intensity of the seed point corresponds to the expected value in the foreground region. The variance is entered by the user and is found by trial and error (see Fig. 6.25).

While finding a good homogeneity criterion can be done interactively in 2d, controlling results in 3d by visual feedback is much more difficult. Homogeneity may then be found automatically in a two-pass process assuming that the object is well contrasted against the background (Pohle and Toennies 2001). In the first pass, a seed point is set and homogeneity is computed based on an estimate of expected value and variance. This estimate is based on the number of elements in the region already found and the current variance among these elements.

Initially, since the region has only few elements, the variance is assumed to be underestimated and corrected accordingly. With continuing growth, the homogeneity criterion approaches current estimates of expected value and variance of intensity. Growing terminates when no further scene elements can be included that follow the current estimate of the homogeneity criterion. The process is repeated with the same seed point but with the homogeneity criterion computed in the first pass.

The algorithm is successful if the estimate of the region characteristics is reliable before the region boundary is encountered for the first time. Hence, the user is asked to place the seed point in the center of the foreground object. The neighborhood order in the for-loop is changed at random producing a kind of random walk (see Fig. 6.26).

A variant to make the region growing process independent of parameters except for the seeds is *seeded region growing* that turns region growing into an mWST-like segmentation procedure (Adams and Bischof 1994). In seeded region growing, the user specifies a number of seeds that separate the image in as many segments as there are seeds. Initially, each seed forms its own segment and labels of all other pixels are unknown. Unlabeled neighbors of all labeled segments are ordered by some appropriate homogeneity criteria (e.g., intensity difference). The most similar

Fig. 6.26 Homogeneity criterion can be found in adaptive region growing by gathering information in a first path starting at the seed point. Region growing is then repeated with the same seed point when homogeneity has been determined

unlabeled pixel is then selected to be labeled. If this pixel is adjacent to labeled pixels which all have the same label, it receives this label. Otherwise, it receives the label of the segment to which it is most similar (see Fig. 6.27).

It is often wanted that inclusion criteria for region growing depend on features at the seed location as this allows to implicitly define homogeneity by selecting just the right location for placing a seed. It does mean, however, that results depend on the seed point location. If this is critical, *symmetric region growing* (Wan and Higgins 2003) may be used, which was shown by the authors to produce results independent of the seed point location as long as the seed is placed into the segment. The algorithm requires a symmetric inclusion criterion, i.e., if this criterion allows a region to grow from some point A to some other point B, then the same criterion should allow growing from B to A.

Fig. 6.27 In seeded region growing, each region receives a seed. Since always the most similar pixel is added to one of the regions, homogeneous sets of pixels will have been added to the regions before a pixel at the boundary of two regions is selected (e.g., the *white circle*). Pixels neighbored to two or more regions are added to the one region to which they are the most similar

Fig. 6.28 Interactively adding an artificial boundary is a simple way to prevent region growing to leak into parts that do not belong to the object to be segmented

Region growing sometimes leaks into the background because of noise, artefacts, or the approximating nature of the homogeneity criterion. Preventing leakage can be done either by interactively adding boundaries that must not be crossed by the growing process (see Fig. 6.28) or by using several seed points and creating the segment as union of regions from these seed points. The former is suitable if—because of noise or other objects close by with similar appearance—the leak happens only at a few isolated locations. The latter is appropriate when the image has a well-contrasted foreground object with shading. Selecting several seed points then assumes that the expected value of the intensity is at least locally constant.

6.9 Live Wire

If it is impossible to give a homogeneity criterion to separate an object from the background, it may still be possible to define its boundary given interactive input. There are a number of methods to do interactive edge tracking, but by far the most popular in medical image analysis is the *live wire* method (Mortensen et al. 1992; Barrett and Mortensen 1997; Falcão et al. 1998), also known as *intelligent scissors* (Mortensen and Barrett 1995). The concept is simple. The user selects a start point on the boundary; then minimum cost paths according to some optimality criterion are computed to all other points and the user selects the most appropriate end point among those (see Fig. 6.29). This becomes the start point for the next contour

Fig. 6.29 Live wire segmentation produces boundary segments as optimal paths between user-specified start and end points

segment. The procedure continues until the first boundary point is reached and the boundary around the object of interest is closed.

The methodology is popular because interaction is easy and the response to interaction is fast. In order to understand the behavior of the live wire algorithm, it is helpful to know some of its details. Several components govern the behavior of a live wire contour:

- The optimality criterion is a local criterion that can be evaluated at each pixel site and decreases with increasing likelihood of this site being part of the contour.
- The distance between start and end point indicates the maximum distance between sites where the optimality criterion overrides user input.
- Pre-processing separates artefactual influences from true edge information in the local optimality criterion.

Optimality is a combination of local pixel attributes and the boundary length. In other words, in the absence of local attributes the optimal boundary between two points is a straight line.

The image is turned into a graph by making nodes out of every pixel and by connecting two nodes by an edge if two pixels are neighbored (4- or 8-neighborhood). Non-negative local costs are attributed to the nodes of the graph. In Falcão et al. (1998), various attributes are computed for generating the cost function. They are related to the expected intensity, expected gradient length, or (for slice-wise segmentation) deviation from the result in the previous slice and subjected to selected feature functions (e.g. linear, Gaussian) that map on a range 0…1, so that 0 is the desired result. The user can train the attribute to segment-specific ranges (if, e.g., not the longest gradient but some intermediate range is searched).

A simple example for node costs $c(i, j)$ for a pixel (i, j) with function value $f(i, j)$ is

$$c(i,j) = (\|\nabla f(i,j)\| - \bar{g})^2, \tag{6.12}$$

where $\bar{g}$ is the expected gradient length for pixels belonging to the boundary. This cost function assumes that the average strength $\bar{g}$ along the boundary is known and penalizes deviations from it.

Given the cost function for the nodes, the optimal boundary between $(i_{\text{start}}, j_{\text{start}})$ and $(i_{\text{end}}, j_{\text{end}})$ can be computed as minimum cost path using Dijkstra's algorithm. This algorithm keeps a list of active nodes and associated path costs for reaching this node. Initially, it contains only the start node of the path. The path cost is the node cost. Then, the node with lowest path cost is selected and removed from the active node list. Path costs of nodes that are connected to the selected node by an edge are computed. These nodes are added to the active node list. The algorithm terminates when the end node is selected from the active node list. The minimum cost path is generated back-tracking the path from the end node to the start node.

The value of $\bar{g}$ needs to be pre-defined. It could be the largest gradient in the image. In this case, a boundary with the strongest gradients is searched. It could be the gradient at the start point which is then taken as model for the boundary (see Fig. 6.30). It could also be computed in a training phase, where representative boundaries are delineated manually and an average gradient is computed along these boundaries. The latter two solutions are especially useful when the optimal boundary does not contain the strongest edges in the image.

Given the cost function, optimal paths from the start pixels to all pixels can be computed at $O(N^2)$ costs, where N is the number of pixels. Backtracking from p_{end} can be done with cost $O(n)$, where n is the number of pixels along the boundary between p_{start} and p_{end}. Since n is small, this computation is possible in real time. Hence, after fixing the start point, path costs can be computed to every possible location in the image (the termination criterion is then that the list of active nodes is

Fig. 6.30 Choosing maximum gradient length for live wire segmentation would not produce the desired result for segmenting the liver boundary. Using the gradient at the start point as training sample should be more successful

Fig. 6.31 Live wire is an interactive procedure. **a** After selecting a start point path, costs are computed for arbitrary end points and the user selects a point on the boundary where the minimum cost path still follows the object boundary. **b** Even if the path follows the boundary, noise in the image causes a rugged path instead of the expected smooth object boundary

empty). The user may then move with the mouse over the image and the method will immediately produce the optimal boundary curve to the current mouse position (see Fig. 6.31). This makes interaction attractive as it allows simple selection of appropriate start and end points for computing boundary sections.

The method has some features that a user should be aware of:

- The path costs are a combination of a local cost (usually dependent on the gradient) and the path length. The latter may lead to unexpected behavior. A path may be preferred that consists of a small number of pixels although the gradients along the pixels do not resemble the expected gradient. An extreme case would be a start point that is also the endpoint of a boundary. The minimum cost path will always consist only of this point, no matter how well defined the boundary is.
- Delivering the shortest possible connection between start and end point is a useful property in cases where part of the boundary is missing (i.e., not recognizable by a pronounced gradient). Noise, however, which is present in most medical images and which also leads to nonzero gradients, may cause the path to follow the noise pattern (see Fig. 6.31b). This happens even if the noise amplitude is so small that it is not visible to the user. Setting closely spaced boundary points in such regions can be a frustrating experience. It is better to let the user switch to an alternative segmentation mode in low-gradient regions (in the simplest case, this were just straight lines between boundary points).
- Using gradient characteristics of the start point for defining node costs can lead to unexpected results if the user has not placed the start point directly on the boundary. There are two ways to avoid this:

- Start point selection may be followed by an automatic correction step that searches for a high-gradient location in the vicinity of the start point location.
- Alternatively, the user may be asked to select a model gradient from a choice of pre-computed boundary gradients of different strengths.

If the data are very noisy, it may not be possible to define a good gradient feature for the boundary search by live wire. If, furthermore, the boundary in search is close to other boundaries of similar appearance, segmentation by live wire may require a sequence of very closely set start points. Instead, the user may roughly sketch a corridor of pre-specified width around the boundary and restrict the search for the boundary to just this corridor. The strategy has been termed live lane by Falcão et al. (1998). It does not only avoid to find unwanted segment boundaries but also reduces computation time since only pixels in the lane become nodes of the graph.

Another interesting variant of the live wire paradigm is Xu et al. (2011). They add a smoothness constraint to the method for finding a double contour in retina images. The method uses a center line as input and computes the contours as orthogonal displacements above and below the center line. Smoothness of the contours is enforced because the change from orthogonal to the centerline is penalized.

Because of its simple and in most cases intuitive behavior, live wire segmentation is found in many software packages for medical workstations. Its main disadvantage is that—opposed to the mWST and to region growing—it is a 2d technique. It can be extended to 3d by using sequential processing. A 3D slice stack is segmented with results in previously segmented slices providing constraints for the segmentation in the current slice (Schenk et al. 2001). However, this makes only little use of shape information orthogonal to the slice orientation.

The work of Falcão and Udupa (2000) takes 3d shape information into account by defining a two-pass procedure on a sequence of slices. In the first path, boundaries are generated using live wire on a number of slices orthogonal to the original slice direction. The intersections of these boundaries with the original slices are then starting points for a boundary search in the slices. Although it extends the method to 3d, the separation into two passes makes it less intuitive compared to the original live wire method.

6.10 Concluding Remarks

Segmentation in medical image analysis often attempts to delineate specific objects of interest. Hence, segmentation methods require additional domain knowledge. It can be included by parameterization of a method. Since this may be too complex to be captured by a simple parameter estimation step, many of the basic segmentation techniques include an interactive component where a domain expert supplies missing information.

Basic segmentation techniques from general image analysis have been adapted accordingly, resulting in methods such as region growing or marker-based watershed transformation where the user points out the objects to be delineated. Other alternatives are the various live wire methods where the user directly indicates locations on the object boundary. While such interactively guided procedures can be very effective, its interaction should not be used to correct poorly applied model or data knowledge.

6.11 Exercises

- How can assumptions about continuity in time be used to support thresholding a time sequence of body CT images to segment the spine?
- How can 3D region growing based on a single seed point in a slice be realized? Discuss potential problems of such an approach and give possible solutions for those problems.
- Give examples for the five types of interaction when used for a threshold segmentation of bone in CT images.
- What are potential problems that could make homogeneity assumptions for intensity in a segment invalid?
- What is the difference between a hierarchical segmentation strategy and a multilayer segmentation strategy? How would an image look like where the later would be preferred?
- What is the main problem when using the homogeneity constraint to guide a texture-based segmentation?
- Which of the segmentation methods presented in this chapter uses knowledge about the appearance of object boundaries as domain knowledge? How is it used in this method?
- How can knowledge about the location of an object be integrated into threshold segmentation?
- Why is it necessary that domain knowledge must be generalizable?
- How could knowledge about the object size and acceptable size variance be gathered? How can it be used in threshold segmentation?
- Interactive boundary delineation of 3d data sets is tedious since it has to be repeated for every slice. What would be appropriate means to support the user? Discuss potential problems of the suggested method and how to deal with them.
- Discuss reasons for using Otsu's method for automatic threshold determination in threshold segmentation. What needs to be initialized and how is this done?
- What method could be used for threshold determination if the two local maxima of the bimodal distribution differ to a large degree? Why is that so?
- What is the reason for using relaxation labeling in threshold segmentation? What is the underlying assumption about the segments which must be true for relaxation labeling to succeed?

- How could a gradual refinement technique (hierarchical multi-scale segmentation) be used to support segmentation by region merging? What would be the benefits of such method?
- Explain why and under what circumstances segmentation by watershed transform is equal to segmentation by finding zero crossings of the second derivative in the image.
- How is the watershed transform extended to include a gradual refinement technique?
- Why is it not useful to maximize the sum of gradient lengths for a boundary as criterion in live wire segmentation?
- What are potential problems that a user may encounter when using the live wire method for boundary delineation?
- How can the live wire approach be extended to 3d? Explain the kind of user interaction in this case.

References

Adams R, Bischof L (1994) Seeded region growing. IEEE Trans Pattern Anal Mach Intell 16(6):641–647

Barrett WA, Mortensen EN (1997) Interactive live-wire boundary extraction. Med Image Anal 1(4):331–341

Beucher S (1994) Watershed, hierarchical segmentation and waterfall algorithm. In: Serra J, Soille P (eds) Mathematical morphology and its application to image and signal processing. Kluwer Academic Publishers, pp 69–76

Cheng J, Liu J, Xu Y, Yin F, Wong DWK, Tan NM, Tao D, Cheng CY, Aung T, Wong TY (2013) Superpixel classification based optic disc and optic cup segmentation for glaucoma screening. IEEE Trans Med Imag 32(6):1019–1032

Cheriet M, Said JN, Suen CY (1998) A recursive thresholding technique for image segmentation. IEEE Trans Image Process 7(6):918–921

Falcão AX, Udupa JK (2000) A 3d generalization of user-steered live-wire segmentation. Med Image Anal 4(4):389–402

Falcão AX, Udupa JK, Samarasekera S, Sharma S, Hirsch BE, de Lotufo AR (1998) User-steered image segmentation paradigms: live wire and live lane. Graph Models Image Process 60(4):233–260

Grau V, Mewes AUJ, Alcañiz M, Kikinis R, Warfield SK (2004) Improved watershed transform for medical image segmentation using prior information. IEEE Trans Med Imaging 23(4): 447–458

Haralick RM, Shanmugam K, Dinstein KI (1973) Textural features for image classification. IEEE Trans Syst Man Cybernet 3(6):610–621

Haris K, Efstratiadis SN, Maglaveras N, Katsaggelos AK (1998) Hybrid image segmentation using watersheds and fast region merging. IEEE Trans Image Process 7(12):1684–1699

Hou Z (2006) A review on MR image intensity inhomogeneity correction. Int J Biomed Imag: 1–11

Lorigo LM, Faugeras O, Grimson WEL, Keriven R, Kikinis R (1998) Segmentation of bone in clinical knee MRI using texture-based geodesic active contours. In: 1st international conference medical image computing and computer-assisted intervention—MICCAI 1998,

LNCS vol 1496, pp 1195–1204
Marr D, Hildreth E (1980) Theory of edge detection. Proc R Soc Lond B 207(1167):187–217
Meyer F, Beucher S (1990) Morphological segmentation. J Vis Commun Image Represent 1(1):21–46
Mortensen EN, Morse B, Barrett W, Udupa JK (1992) Adaptive boundary detection using 'live-wire' two-dimensional dynamic programming. Proc Comput Cardiol, 635–638
Mortensen EN, Barrett WA (1995) Intelligent scissors for image composition. In: Proceedings of the 22nd International Conference, SIGGRAPH 95, pp 191–198
Muzzolini R, Yang YH, Pierson R (1993) Multiresolution texture segmentation with application to diagnostic ultrasound images. IEEE Trans Med Imaging 12(1):108–123
Najman L, Schmitt M (1996) Geodesic saliency of watershed contours and hierarchical segmentation. IEEE Trans Pattern Anal Mach Intell 18(12):1163–1173
Nock R, Nielsen F (2004) Statistical region merging. IEEE Trans Pattern Anal Mach Intell 26(11):1452–1458
Otsu N (1978) A threshold selection method from grey-level histograms. IEEE Trans Syst Man Cybernet SMC-8:62–66
Pitiot A, Toga AW, Ayache N, Thompson P (2002) Texture based MRI segmentation with a two-stage hybrid neural classifier. In: Proceedings of the international joint conference neural networks (IJCNN'02), vol 3, pp 2053–2058
Pohle R, Toennies KD (2001) Segmentation of medical images using adaptive region growing. Proc SPIE (Med Imaging 2001) 4322:1337–1346
Reyes-Aldasoro CC, Bhalerao A (2003) Volumetric texture description and discriminant feature selection for MRI. In: Information processing in medical imaging, IPMI 2003, LNCS, vol 2732, pp 282–293
Roerdink JBTM, Meijster A (2000) The watershed transform: definitions, algorithms and parallelization strategies. Fundam Inf 41:187–228
Rosenfeld A, Smith RC (1981) Thresholding using relaxation. IEEE Trans Pattern Anal Mach Intell 3(5):598–606
Sahoo PK, Soltani S, Wong AKC, Chen YC (1988) A survey of thresholding techniques. Comput Vis Graph Image Process 41(2):233–260
Schenk A, Prause GPM, Peitgen HO (2001) Local-cost computation for efficient segmentation of 3D objects with live wire. Proc SPIE (Med Imaging 2001) 4322:1357–1364
Tobias OJ, Seara R (2002) Image segmentation by histogram thresholding using fuzzy sets. IEEE Trans Image Process 11(12):1457–1465
Tomazevic D, Likar B, Pernus F (2002) Comparative evaluation of retrospective shading correction methods. J Microsc 208(Pt 3):212–223
Vincent JL (1993) Morphological grayscale reconstruction in image analysis: applications and efficient algorithms. IEEE Trans Image Process 2:176–201
Vovk U, Pernus F, Likar B (2007) A review of methods for correction of intensity inhomogeneity in MRI. IEEE Trans Med Imaging 26(3):405–421
Wan SY, Higgins WE (2003) Symmetric region growing. IEEE Trans Image Process 12(9): 1007–1015
Xu X, Niemeijer M, Song Q, Sonka M, Garvin MK, Reinhardt JM, Abramoff MD (2011) Vessel boundary delineation on fundus images using graph-based approach. IEEE Trans Med Imaging 30(6):1184–1191
Zack G, Rogers W, Latt S (1977) Automatic measurement of sister chromatid exchange frequency. J Histochem Cytochem 25:741–753
Zucker SW (1976) Region growing: childhood and adolescence. Comput Graph Image Process 5(3):382–399

Segmentation in Feature Space

7

Abstract

Selection of an image acquisition technique is intentional in medical imaging. It can be assumed that pixel or voxel values in a medical image cover more semantics with respect to object class membership than intensity in a photograph. Hence, image segmentation can be done as classification in feature space where image intensities are the features. The dimensionality of feature space is usually low, and the number of samples characterizing object classes is high. Typical classifiers discussed in this chapter take this into account and estimate likelihood functions from samples. Classification is then done by computing a posteriori probabilities for each object class. Clustering in feature space will be discussed as well. Without requiring training, clustering may directly lead to segmentation. Even if this is not the case, clustering may be used to reduce the work load for producing the training data.

Concepts, Notions, and Definitions Introduced in this Chapter

- Classification of pixels and voxels;
- Bayesian classification;
- Computation of likelihood functions;
- Kernel density estimators;
- Gaussian mixture models;
- Clustering in dense, low-dimensional feature spaces;
- Association networks and Kohonen networks.

Thresholding presented in the previous chapter is actually a solution of the segmentation problem as a classification task in feature space (see Fig. 7.1). In thresholding, scene elements are assigned to one of the two classes—foreground or background—based on their intensity value. This assumes that some function exists

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_7

Fig. 7.1 Segmenting the image based on threshold implies underlying a posteriori probabilities for the different classes of which the histogram shows a non-normalized sum

that relates class membership probability to intensity of the scene element. This is part of a Bayesian formulation of a classification problem. Solving segmentation in this way is adequate for many kinds of medical images where a relationship between measured intensity value and tissue type can be assumed. Methods to solve the classification task will be discussed in this chapter.

Segmentation is rather specific compared to other classification tasks. It makes some methods more appropriate than others. Typically, the dimension of the feature vector is low. It may consist of a single scalar (such as the image intensity in Fig. 7.1), of low-dimensional multi-channel data (e.g., a triple of proton density, T_1 and T_2 relaxation in MRI imaging), or of a combination of intensity information with spatial information such as the coordinates of a scene element. Hence, the dimensionality of a feature vector is somewhere between one and five. The number of samples is high, since every scene element is a sample. The situation is almost the opposite to a regular classification task, where few samples in high-dimensional feature space represent the probability density functions that are needed to estimate the a posteriori probability of an unknown sample to belong to a certain class. In segmentation, methods are preferred that rely on parameter estimation to compute the a posteriori probability of a scene element to belong to some object class.

If the number of features per scene element is high, classification techniques for sparse sample distributions in high-dimensional space need to be employed. These techniques will be discussed in Chap. 12.

Fig. 7.2 Sometimes, the probability distributions intersect twice. Threshold t_1 is then disregarded, as it occurs at a location where probability for either of the two classes is low

7.1 Segmentation by Classification in Feature Space

Given an intensity $f(\mathbf{v})$ at some scene element $\mathbf{v}$ and two different classes—foreground fg and background bg—the optimal solution that minimizes the number of wrong decisions would be to assign the class with highest a posteriori probability $P(\mathbf{v}\in fg|f(\mathbf{v}))$ and $P(\mathbf{v}\in bg|f(\mathbf{v}))$, respectively to $\mathbf{v}$. According to the Bayesian theorem, the a posteriori probabilities can be computed from the a priori probabilities $P(\mathbf{v}\in fg)$ and $P(\mathbf{v}\in bg)$[1] and the likelihood functions $p(f(\mathbf{v})|\mathbf{v}\in fg)$ and $p(f(\mathbf{v})|\mathbf{v}\in bg)$:

$$P((\mathbf{v} \in fg|f(\mathbf{v}))) = \frac{p(f(\mathbf{v})|\mathbf{v} \in fg)P(\mathbf{v} \in fg)}{p(f(\mathbf{v})|\mathbf{v} \in fg)P(\mathbf{v} \in fg) + p(f(\mathbf{v})|\mathbf{v} \in bg)P(\mathbf{v} \in bg)}, \tag{7.1}$$

$$P((\mathbf{v} \in bg||f(\mathbf{v}))) = \frac{p(f(\mathbf{v})|\mathbf{v} \in bg)P(\mathbf{v} \in bg)}{p(f(\mathbf{v})|\mathbf{v} \in fg)P(\mathbf{v} \in fg) + p(f(\mathbf{v})|\mathbf{v} \in bg)P(\mathbf{v} \in bg)}. \tag{7.2}$$

The denominator is the same in both equations and can be omitted for finding the most likely class membership. Sometimes, the two a posteriori probabilities intersect only at one location. This is then the threshold value where the decision should switch from foreground to background. Otherwise, a priori probabilities and likelihood functions have to be evaluated for each value $f(\mathbf{v})$.

The two curves may intersect more than once (see Fig. 7.2). The extra intersection can be disregarded if it occurs at a location with low probability values. This would indicate a low reliability of the approximation of the true likelihood. It is for instance the case when Gaussian distributions with different means and variances approximate the likelihood function. The second intersection of such distributions is usually out of the range of permissible values for $f(\mathbf{v})$. Even if it is within the range

[1]The a priori probability is also called *marginal probability*, since P is marginalized over all possible feature values of $\mathbf{v}$.

of values for $f(\mathbf{v})$, this intersection is assumed to be due to an inaccurate approximation of the likelihood function by the Gaussian function.

A priori probability and likelihood function have to be generated from training data. The data should be representative for the problem so that its feature values reflect the true likelihood function. If appearance of foreground and background may change for different patients (or even for the same patient if imaged several times), training data should come from several different image sequences. Individual variation of tissue characteristics, image artifacts that vary with different patients such as shading artifacts or weight-dependent contrast, or the use of imaging devices from different manufacturers or with different parameterizations are reasons why such variation can be expected.

7.1.1 Computing the Likelihood Function

Given the classified training data, the estimate of the likelihood function for the two classes are normalized histograms h^n_{fg} and h^n_{bg} of the intensity values of scene elements belonging either to the foreground or the background. The histograms can be taken directly as likelihood functions (see Fig. 7.3) if the number of samples is high enough to cancel out effects from noise.

Using a *kernel density estimator* (KDE, also known as *Parzen window*) helps to reduce unwanted effects from undersampling (see Fig. 7.4). It is based on the assumption that each sample value $f(\mathbf{v})$ is representative for a range of values in the vicinity of $f(\mathbf{v})$. Hence, $f(\mathbf{v})$ is assumed to be the mean of some unknown density function $d()$. A new estimate for the likelihood function is then computed by convolving the normalized histograms with the density functions d_{fg} and d_{bg}:

$$p() = h^n_{fg} * d_{fg} \tag{7.3}$$

Fig. 7.3 Estimates for foreground and background probability can be generated from a representative sample. In the example above, vessels from 3d reconstructed DSA are to be separated from the background. It can be seen by inspection of the histograms that the background is well-approximated by a Gaussian while this is not true for the foreground

Fig. 7.4 **a** Noise in the sample data can be reduced by smoothing the histogram using a kernel density estimator. **b** The same technique can be used when the number of samples is too small for providing a reliable estimate for the probability distribution

and

$$p() = h_{bg}^{n} * d_{bg}. \tag{7.4}$$

Since d_{fg} and d_{bg} are usually unknown, they are approximated by Gaussian density functions (based on the central limit theorem). The variance of the Gaussian is set based on the average sampling density in the normalized histograms and noise characteristics. If the number of histogram bins is small compared to the number of samples, a low variance should be chosen while a higher variance should be selected otherwise.

It is well possible that different variances are appropriate for foreground likelihood and background likelihood functions. The number of foreground samples from the training data may be much smaller than that of background samples. Furthermore, the range of intensity values in the foreground may be different to that in the background.

It may be necessary to constrain the likelihood function even further. If it is assumed that in a noise-free environment the foreground or the background has homogeneous intensity, the only reason for variation of intensity values is noise. Noise is often assumed to be normally distributed with zero mean. The likelihood function is then a Gaussian with mean value representing the expected foreground or background intensity and variance representing measurement noise:

$$N(f; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{1}{2} \frac{(f-\mu)^2}{\sigma^2} \right) \tag{7.5}$$

A number of reasons may speak against using a Gaussian density function (compare the distributions of intensities for foreground and background in Fig. 7.3). The most notable non-Gaussian deviation is caused by the partial volume effect (PVE). If a bright foreground object is to be segmented from a dark background, some of the variation from the mean is not caused from normally distributed noise

but from PVE between bright and dark scene elements. The intensity is no longer normally distributed and not even symmetric with respect to the mean. Another example is a projection image when using contrast agent. In DSA imaging, the ideal image should have a constant white background with vessels of varying degrees of black depending on vessel thickness along the projection ray. Foreground variation from varying vessel thickness is non-Gaussian and a substantial cause for observed intensity variation.

Sometimes, it is known beforehand that the intensity distribution is actually a combination of several density functions. An example would be a foreground object that consists of different tissue types (see Fig. 7.5). The characteristic is captured by the normalized histogram with or without Parzen windowing. However, if the number of samples is too low for computing a reliable estimate without a model for the function, a *mixture of Gaussians* can be assumed with different means, variances, and a priori probabilities. It is a particular appropriate model if the class that is to be described consists of a mixture of several substances with different appearances. The individual likelihood functions are allowed to overlap to a large extent, since classification of the substances is not the goal.

A Gaussian mixture model Θ for one-dimensional distributions consists of a set of K Gaussian density functions $N(\mu_i, \sigma_i)$ with mean μ_i and standard deviation σ_i so that the probability of some event to have the value x_j is given by a weighted sum of the functions $N(\mu_i,\sigma_i)$:

$$P\left(x_j|\Theta\right) = \sum_{i=1}^{K} a_i N\left(x_j; \mu_i, \sigma_i\right) \tag{7.6}$$

The weights a_i are the a priori probabilities for each distribution. For our application, the values x_j are the intensity values of the scene element $\mathbf{v}$. The unknown mixture model needs to be estimated from samples. Obviously, the most fitting model for a set of sample values $X = \{x_j\}$ would be the one for which the probability $P(X|\Theta)$ is maximum, i.e.,

Fig. 7.5 Sometimes, the intensity distribution of a histogram has to be modeled as a mixture of several density functions with unknown parameters

$$\Theta_{\max} = \arg\max_{\Theta} P(X|\Theta). \tag{7.7}$$

There exists no analytical solution. However, a number of iterative techniques arrive at $\Theta_{\max}$ given some initial estimate. A popular approach is to use the *expectation–maximization algorithm* (EM). Expectation maximization is an iterative procedure to estimate a maximum a posteriori solution such as the one in Eq. (7.6). A detailed look at this technique can be found in McLachlan and Krishnan (1996).

Application of the EM algorithm in this case requires the number K of Gaussians to be known. It consists of two steps, which are repeated until convergence is reached. In the E-step, expectations y_{ij} of samples x_j belonging to distribution i are computed given the current estimates for μ_i, σ_i, and a_i:

$$y_{ij} = \frac{a_i N\left(x_j; \mu_i, \sigma_i\right)}{\sum_{i=1}^{K} a_i N\left(x_j; \mu_i, \sigma_i\right)} \tag{7.8}$$

In the M-step, these expectations are used to compute new estimates. The new estimate for the a priori probability is simply the average of expectations for this class:

$$a_i' = \frac{1}{N} \sum_{j=1}^{N} y_{ij}. \tag{7.9}$$

The estimates for the means and variances are then

$$\mu_i' = \frac{\sum_{j=1}^{N} y_{ij} x_j}{\sum_{j=1}^{N} y_{ij}} \tag{7.10}$$

and

$$\sigma_i' = \sqrt{\frac{\sum_{j=1}^{N} y_{ij}\left(x_j - \mu_i'\right)^2}{\sum_{j=1}^{N} y_{ij}}}. \tag{7.11}$$

The process converges to a local optimum. Since this may be far away from the global optimum, selecting good starting values is paramount. If these cannot be found easily, the search for the mixture model may be preceded by a clustering step (see Sect. 7.2), where K clusters are searched in the data. Cluster centers and within-class scatter are then initial estimates for mean and average deviation.

7.1.2 Multi-dimensional Feature Vectors

Computation does not change much if the feature is not a scalar but a vector. It is particularly easy if the features can be assumed to be independent of each other. The likelihood function for a feature vector **f** is then the product of the likelihood functions of the elements $f_1, \ldots, f_M$ of the vector:

$$p(\mathbf{f}|\mathbf{v} \in fg) = \prod_{m=1}^{M} p(f_m|\mathbf{v} \in fg). \tag{7.12}$$

Estimation of the likelihood functions can be done individually in the same way than for scalar features.

The likelihood function becomes a function of M values if independence cannot be assumed and probabilities need to be estimated in multi-dimensional feature space (see Fig. 7.6). This is again not different to the case discussed in the previous section, except for the density of samples. While in 1d the number of samples even from a single segmented image is high compared to the number of bins in the histograms, this changes dramatically if the feature vector has two, three, or more dimensions.

Let us assume that we have taken a single segmented slice with 512 × 512 pixels for training in order to illustrate this by an example. If 64.000 pixels of the total of 2^{18} pixels belong to the foreground and the intensity ranges from 0 to 4000, of which the foreground object occupies a range of 400 gray values, we have on average 160 samples (i.e., 64.000/400) per bin. Unless the image is very noisy, this would allow for using the sampled, normalized histogram directly as likelihood function. However, if we had the same situation, but the image consisted of three channels (e.g., T_1, T_2, and proton density in MRI) where the foreground object spanned the range of 400 gray levels in each channel, the average number of samples would be down to 0.001 per bin, which is clearly insufficient even for

Fig. 7.6 Feature space becomes multi-dimensional if more than one feature describes the data

applying a kernel density estimator. Even if we used sequences from 50 patients with each sequence having 20 slices in which the foreground object is visible, the average number of samples would still be only one per bin.

Parametric distribution functions are therefore often used for estimating a multi-dimensional likelihood function. Based on the central limit theorem, a normal distribution is used as model. Since features are not independent, the variances for each element of the feature vector have to be replaced by the covariances. The *covariance matrix* Σ for an M-dimensional feature vector consists of entries σ_{ij} that can be estimated from K sample features $\mathbf{f}^k$ by

$$\sigma_{ij} \approx \frac{1}{K-1} \sum_{k=1}^{K} \left(f_i^k - \mu_i\right)\left(f_j^k - \mu_j\right), \tag{7.13}$$

where $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots \mu_M)$ is the estimated mean for features $\mathbf{f}^k$. For real-valued feature values, the covariance matrix is symmetric and has the size K^2 for a K-dimensional feature vector (see Fig. 7.7). Given a 3d feature vector, in total 9 parameters (3 entries for the mean μ and 6 independent values of the 3×3 covariance matrix) have to be estimated. Hence, the 64.000 samples in this example would be more than sufficient.

The multi-dimensional normal distribution $N(\mathbf{f}; \boldsymbol{\mu}, \Sigma)$ for a feature vector $\mathbf{f}$ is then

$$N(\mathbf{f}; \boldsymbol{\mu}, \Sigma) = \frac{1}{\sqrt{(2\pi)^M \det(\Sigma)}} \exp\left(-\frac{1}{2}(\mathbf{f} - \boldsymbol{\mu})^{\mathrm{T}} \Sigma^{-1} (\mathbf{f} - \boldsymbol{\mu})\right), \tag{7.14}$$

where Σ^{-1} denotes the inverse of Σ and $()^{\mathrm{T}}$ the transpose of a vector.

Fig. 7.7 The covariance matrix C in 2d feature space is a 2×2 matrix. The high value of off-diagonal entry compared to the variances in the diagonal shows that there is a lot of covariance between the two features. The redundancy, which is causing the co-variance, is indicated by the *dashed line* in the 2d histogram

In principle, it is possible to compute a multi-dimensional threshold for a feature vector. Different methods will be presented in this chapter on classification, since it is particularly appropriate when sampling density is too sparse for computing reliable estimates for the parameters. In segmentation, the parameterized function can often be used directly. A posteriori probabilities of class membership (foreground and background) are computed from features of a scene element **v**, and **v** is then assigned to the class with highest probability.

7.1.3 Computing a Priori Probability

A priori probabilities represent knowledge about the objects, irrespective of the data. The simplest kind of a priori knowledge for a segmentation between foreground and background is an estimate of the ratio of foreground to background pixels. This can be computed from the training data that was used to compute the likelihood functions. It is important that this reflects the correct ratio. If, for instance, segmentation based on classification of scene elements is carried out as part of a process chain where first an approximate region of interest (ROI) is computed, ratios have to be computed from the ROIs.

A priori knowledge may also include neighbor information because segment membership is spatially correlated. The necessary model would predict that it is more likely that neighboring scene elements belong to the same class. This can be represented by a Markov random field (MRF). The model is similar to the one used in image enhancement (Chap. 4). If we assume a normal distribution for the likelihood function, this is done by replacing the likelihood function of Eq. (4.61) by the term derived in Eq. (7.5) resulting in

$$P(\mathbf{f}|\mathbf{g}) \propto P(\mathbf{g}|\mathbf{f}) \cdot P(\mathbf{f}) = \frac{1}{Z_1}\exp\left(-\frac{(f(\mathbf{v})-\mu)}{2\sigma^2}\right) \cdot \frac{1}{Z_2}\exp(-U(\omega)) \tag{7.15}$$

For segmentation, Geman and Geman (1984) suggested the generalized Ising model to model clique potentials V_c that make up $U(\omega)$

$$V_c = \begin{cases} \xi_c, & \text{if all sites of the clique have the same label} \\ -\xi_c, & \text{otherwise} \end{cases} \tag{7.16}$$

The potential ξ_c is specific to the clique type.

Instead of explicitly computing clique potentials U from all cliques in a given neighborhood, an approximation of U can be defined that describes desired segmentation properties. Since segmentation should produce as few segments as possible, the a priori probability can simply decrease with the number of different labels in the neighborhood of **v**. This has been used for MRF-based MRI brain segmentation by Held et al. (1997) (see Fig. 7.8).

Fig. 7.8 Examples for clique potentials according to the work of Held et al. (1997). The potential decreases sharply in regions with different labels in the neighborhood. Further optimization will change labels at these locations if the likelihood from image is low. This is true for the smaller structures cut by the line

$$P(\mathbf{f}) = \frac{1}{Z_2} \exp\left(- \sum_{\mathbf{w} \in Nb(\mathbf{v})} \delta\Big(f(\mathbf{v}) \neq f(\mathbf{w}) \Big) \right), \tag{7.17}$$

where δ is the delta function

$$\delta(x) = \begin{cases} 1, & \text{if } x = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{7.18}$$

Iterative optimization of the MRF is then done using simulated annealing or other means [e.g., ICM (Besag 1986), mean field optimization (Zhang 1992; Celeux et al. 2003), graph cuts (Boykov et al. 2001)] or by introducing a hidden measure model (Marroquin et al. 2003) (see also Sect. 14.1 on MRFs and their optimization). A good initialization, e.g., by optimizing the function without considering the a priori knowledge using an EM algorithm, helps to arrive at acceptable computation times.

Another option for fast MRF computation is to use multi-resolution MRFs. In Bouman et al. (1994) and Cheng and Bouman (2001), a system of Markov random fields is developed and parameterized, which defines neighbor systems on a multi-scale representation of the image. Results at each scale are included as a priori knowledge in the segmentation of the next scale. The segmentation is initialized by computing a labeling at the coarsest scale and then proceeding through the scales. It allows defining a small neighborhood system, since dependencies across larger ranges are captured at coarser scales. It also speeds up computation, since the initialization from the previous scale is a good predictor for the optimal result at the current scale.

7.1.4 Extension to More Than Two Classes

Likelihood functions and a priori probabilities can be computed for more than two classes if the scene shall be segmented into several different objects. Segmentation is then given by classification to the most probable class.

7.2 Clustering in Feature Space

So far, segmentation required the classes to be known to which scene elements should be assigned. This type of segmentation is actually a combination of segmentation and object detection, since scene elements are assigned a meaning. Segmentation in feature space, however, is possible without having to make this assumption. This is particularly useful if it is not known a priori into how many and which classes scene elements should be grouped.

The process is called *clustering*. The underlying assumption is that scene elements from the same object have more similar features than those that belong to different objects. Similar to classification, clustering is a generic methodology that is able to group features of any kind. It makes sense to differentiate between clustering techniques for low-dimensional densely populated features spaces and those for high-dimensional sparsely populated features spaces. Hence, we will discuss the topic twice (here and in Chap. 12).

Clustering for segmentation works in low-dimensional feature space. In 2d feature space (and with some difficulty in 3d as well), the simplest way of clustering is to do it interactively. The 2d distribution is displayed, and the user points out or delineates clusters. The resulting segmentation is then displayed, and clustering may then be corrected (see Fig. 7.9). However, since this process is highly subjective, it may be more appropriate for a quick first look at the data and should be replaced by some automatic clustering if a series of data sets is to be analyzed.

Fig. 7.9 If feature space is low-dimensional (in this case, $d = 2$), clustering may be done interactively in the histogram and results are immediately displayed

7.2.1 Partitional Clustering and K-means Clustering

K-means clustering is a variant of *partitional clustering*. The objective of a partitional clustering method is to find cluster centers CC = $\{c_1, \ldots, c_K\}$ in feature space such that the distance of all samples to their center is minimal:

$$\mathrm{CC}_{min} = \arg\min_{\mathrm{CC}} d_{\mathrm{CC}}(\mathbf{f}) = \arg\min_{\mathrm{CC}} \sum_{i=1}^{M} \|\mathbf{f}_i - \mathbf{c}(\mathbf{f}_\mathrm{i})\|, \tag{7.19}$$

where $\mathbf{c}(\mathbf{f}_i)$ delivers the cluster center $\mathbf{c}_k$ that is closest to $\mathbf{f}_i$.

A heuristic strategy is employed for finding optimal cluster centers. First, cluster centers are fixed and samples are assigned to clusters that minimize d_{CC} given the cluster centers. Then, a new center $\mathbf{c}_c$ is defined for each cluster c that minimizes $\sum_{f_i \in c_c} \|\mathbf{f}_i - \mathbf{c}_k\|$. The process is repeated until no further change of cluster assignment takes place (see Fig. 7.10).

Partitional clustering is initialized by assigning the *K* cluster centers to *K* arbitrary samples in the data set and by minimizing $\sum_{f_i \in c_c} \|\mathbf{f}_i - \mathbf{c}_k\|$ for the initial centers. The process is heuristic since assignment to clusters (the first step) and determination of new cluster centers (the second step) are optimized separately although they depend on each other. The procedure may not produce an optimal result and is not guaranteed to converge. Providing a good initial guess for the cluster centers is advisable.

Opposed to partitional clustering, *k-means clustering* (Duda et al. 2000) always terminates as it requires just two iterations. The objective remains the same. However, instead of assigning all samples to the current estimate for cluster centers before computing a new cluster center, the two objectives are interleaved by computing new cluster centers after each assignment of a feature to a cluster.

The method is initialized similar to partitional clustering by selecting *K* samples to act as initial cluster centers. Again, results will improve when the initial guess is close to the true location of the cluster centers. The process iterates once through the complete data set (including the samples that were chosen to represent initial cluster centers) by randomly removing samples from it. The selected sample is assigned to its closest cluster center. This cluster now has a new member so that its center has to be moved to the centroid of all samples (see Fig. 7.11).

Fig. 7.10 Partitional clustering starts with initial cluster centers to which samples are assigned according to distance. New cluster centers are computed after the assignment phase. The process is repeated until there are no changes in cluster membership

Fig. 7.11 First pass of the *k*-means clustering algorithm. Samples are selected at random and assigned to the nearest cluster center. After each assignment, cluster center locations are updated

The process is repeated until all samples have been chosen once. After that, final cluster center locations are assumed to have been found. Initial uncertainties stemming from a small number of samples determining a cluster center have been compensated by later addition of samples from the database. Uncertainty about cluster center locations at an early stage may have led to unreliable early cluster assignments. Hence, in a second iteration, cluster centers are kept constant and samples are assigned according to their distance to these centers (see Fig. 7.12).

Fig. 7.12 Second phase in *k*-means clustering. Given the estimates for cluster centers from the first step, all samples are analyzed according to their distance to the nearest cluster center and potentially re-assigned

Knowing the number of clusters and a good initialization for cluster center positions are necessary to employ k-means clustering successfully for segmentation. The number of clusters is often assumed to be larger than the number of different segment classes expected in the image. The main reason for this lies in the nature of clustering algorithms. Clusters are groups of samples that are close together in feature space. It is possible that a segment consists of different tissues with different appearances in the image. These different tissues may fall into different clusters. If the number of clusters is set to the expected number of segments, clusters separating different segments may be merged if they are closer to each other than clusters representing the different segments.

Since such characteristics are often unknown—otherwise, the more direct classification approach of Sect. 7.1 could be pursued—segmentation by k-mean clustering is often a trial-and-error procedure. The user starts with a first guess about the number of clusters, visualizes members of computed clusters, and then varies (increases or decreases) k until an acceptable solution is reached.

Results of k-means clustering depend on a good initialization for cluster centers. Since distribution in feature space is usually unknown, random selection is used to start the procedure. The simplest way is to separate the feature space into K compartments at random and compute centers of gravity from samples in each compartment. Peña et al. (1999) compare four methods and found that this simple random selection had a similar performance than a non-random procedure suggested by Kaufman and Rousseeuw (2005) that took distribution characteristics into account. The latter did, however, terminate less likely in a bad partitioning.

The method of Kaufman and Rousseeuw (2005) starts with sample locations, but it selects the samples according to a heuristic diversity criterion. Iteratively, a sample is selected as new cluster center that maximizes this diversity measure until K clusters have been selected. The diversity measure D rates the distance of the newly selected cluster center $\mathbf{C}_{\text{new}}$ to all already selected centers $\mathbf{C}_1, \ldots, \mathbf{C}_k$, $k < K$ with respect to distances of $\mathbf{C}_{\text{new}}$ to all non-selected samples $\mathbf{s}_i$ in the following way (see also Fig. 7.13)

Fig. 7.13 The diversity measure D rates the closest distance $d^c_{\min}$ of the new center to all other centers with respect to the furthest distance of the new center to any of the unassigned samples

$$d^c_{\min} = \min_k(\|\mathbf{C}_{\text{new}} - \mathbf{C}_k\|), \tag{7.20}$$

$$D = \max_i\left(0, d^c_{\min} - \|\mathbf{C}_{\text{new}} - \mathbf{s}_i\|\right). \tag{7.21}$$

A different strategy was presented by Bradley and Fayyad (1998) who used modes of an estimated density function for improving a given set of initial cluster centers. The samples in feature space are assumed to approximate a mixture of probability density functions (similar to the Gaussian mixture used in Sect. 7.1.1 to represent a likelihood function). The modes of this mixture are then the local maxima of the density and are assumed to be potential cluster centers. Searching for the modes is the subject of mean shift clustering described in the next section. The approach of Bradley and Fayyad (1998) does not require all modes to be detected but moves initially selected cluster centers closer to likely locations of the modes.

K-means clustering can be used as a preprocessing step to segmentation. The number of possible cluster centers is then set high to segment the scene into many regions (which may be thought of as a kind of superpixels). These are then post-processed for the final segmentation. The method is less sensitive to the selection of cluster centers compared to a direct attempt of doing classification via clustering. It has been used, for instance, by Ng et al. (2006) for providing a watershed transform with large enough regions so that noise removal can be done by intensity averaging in the superpixels.

7.2.2 Mean Shift Clustering

K-means clustering requires the number of clusters to be known. Misjudging the number of different clusters may lead to unwanted results. *Mean shift clustering* is an alternative that does not require this kind of information (Fukunaga and Hostetler 1975; Comaniciu and Meer 2002). Instead, mean shift clustering attempts to find all possible cluster centers in feature space. Samples are assumed to stem from a density function, which is a mixture of an unknown number of probability density functions. Each of the probability functions describes the probability of a sample to belong to one cluster. Ideal clustering would identify the probability functions of the mixture model and then assign cluster membership based on this probability.

Determining parameters of an unknown number of probability functions of an unknown type will be difficult or even impossible. In mean shift clustering, heuristics are employed to arrive at a feasible solution. It is assumed

- that the probability density functions of the mixture model have only one maximum and that this maximum represents the mean of the function.
- that combining the probability functions in the mixture model preserves the maxima, so that the local maxima of the mixture function represent the means of the underlying probability functions. The local maxima are the modes of the density function.

- that the local minima of the mixture model segment the feature space so that each segment contains only one of the local maxima.

Finding the local maxima of the mixture model under these assumptions produces the clusters in feature space. The mean shift algorithm proceeds as follows:

- For each location in feature space, a marker is shifted toward the next local maximum by applying a gradient ascent algorithm.
- If a local maximum has been found and it has no cluster label, it is labeled.
- Irrespective of whether the local maximum has been labeled at this step or previously, the location from which it has been found receives the label as cluster label.

Gradient ascent requires computing a gradient of the density function. Since the density function is only approximated by a distribution of samples, the gradient can only be approximated as well (see Fig. 7.14). This is done by a kernel window estimator. If the feature space is isotropic, the kernel is rotationally symmetric (Comaniciu and Meer 2002):

$$k(\mathbf{x}) = k\left(\|\mathbf{x}\|^2\right) \tag{7.22}$$

where k is a one-dimensional function on distance, such as the Gaussian $k(\mathbf{x}) = \exp(-\mathbf{x}^2/2)$. Kernel density estimation for some location $\mathbf{x}$ in d-dimensional feature space with samples $\mathbf{x}_1, \ldots, \mathbf{x}_N$ is then

$$f_{h,K}(\mathbf{x}) = \frac{c_{K,d}}{Nh^d} \sum_{i=1}^{N} k\left(\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{h}\right). \tag{7.23}$$

The value of h determines the width of the kernel and $c_{K,d}$ is a normalizing constant. If k is differentiable and g is the derivative of $-k$, the gradient is

Fig. 7.14 The gradient at some location in feature space is approximated by interpolation over a predefined neighborhood using a suitable kernel function

$$\nabla f_{h,K}(\mathbf{x}) = \frac{c_{K,d}}{Nh^{d+2}} \sum_{i=1}^{N} g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right) \left[\frac{\sum_{i=1}^{N} x_i g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)}{\sum_{i=1}^{N} g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)} - \mathbf{x}\right]. \tag{7.24}$$

The gradient consists of two parts with different meanings. The first part is a kernel estimator using the derivative g instead of the original function k, and the second represents the error between the actual position $\mathbf{x}$ and its estimate by the kernel estimator. The latter is called the mean shift $m_{h,G(\mathbf{x})}$ of $\mathbf{x}$.

The parts can be separated, if a new normalizing constant $c_{G,d}$ for g is introduced and the parts are rearranged:

$$\nabla f_{h,K}(\mathbf{x}) = \frac{c_{G,d}}{Nh^{d}} \sum_{i=1}^{N} g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right) \frac{c_{K,d}}{c_{G,d}h^2} \left[\frac{\sum_{i=1}^{N} x_i g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)}{\sum_{i=1}^{N} g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)} - \mathbf{x}\right] \tag{7.25}$$

The kernel estimator is then

$$\nabla f_{h,G}(\mathbf{x}) = \frac{c_{G,d}}{Nh^{d}} \sum_{i=1}^{N} g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right) \tag{7.26}$$

and the mean shift is

$$m_{h,G}(\mathbf{x}) = \frac{c_{K,d}}{c_{G,d}h^2} \left[\frac{\sum_{i=1}^{N} g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)}{\sum_{i=1}^{N} x_i g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)} - \mathbf{x}\right]. \tag{7.27}$$

Starting with some initial position $\mathbf{x}$, the algorithm proceeds by repeatedly changing the position by the mean shift until the gradient length is zero and a local maximum is reached. If this is done for all sample points and cluster labels are created and assigned as described above, it results in finding all modes in feature space and assigning each location in feature space to its corresponding mode (see Fig. 7.15).

In Comaniciu and Meer (2002), 2d images were segmented using color information and the pixel coordinates as features. Since the mean shift algorithm is a very general method, it has found many applications. Its biggest advantage is that it is parameter-free. Determination of the number of clusters and of the clustering itself does not depend on user input.

Such a data-driven decision has disadvantages, however. Segmentation by mean shift often results in oversegmented images, since every local maximum in feature space forms its own cluster (Tuzel et al. 2008). In Tuzel et al. (2009), additional constraints in feature space were used to override the decision of the mean shift procedure (so-called must-link constraints that forced cluster elements to stay bound together).

Fig. 7.15 The mean shift algorithm starts moving from every position in feature space toward the local maxima in the function

Computing the mean shift for each location in feature space is time-consuming if the dimension is high. However, this is not critical for segmentation with its usually low-dimensional feature space. For high-dimensional space, variants such as median shift (Shapira et al. 2009) have been developed.

7.2.3 Kohonen's Self-organizing Maps

Neural networks may be used for unsupervised clustering as well. Clustering is done by association of a feature vector to a model cluster vector using a similarity measure.

A well-known network of this kind is Kohonen's *self-organizing map* (*SOM* or Kohonen network) (Kohonen 1995). The network shares some similarities with mean shift clustering discussed in the previous section; in that, it attempts to cluster data based on some inherent attributes. In an SOM, this attribute is the similarity between feature vectors. Furthermore, an SOM also assumes some underlying structure between different clusters and attempts to find this structure as well. The structure information can be used to steer the clustering procedure.

The network consists of a single output layer that is fully connected to all nodes in the input layer (see Fig. 7.16). Each node in the input layer represents a feature in feature space. Each node in the output layer corresponds to a potential cluster.

The network is called self-organizing as it learns feature patterns and the organization of the feature distribution without supervision. Feature patterns are represented by weight vectors $\mathbf{w}_j = (w_{1j}\ w_{2j}\ \ldots\ w_{Nj})^{\mathrm{T}}$ leading to output nodes j. Organization of the feature distribution is represented by connecting output nodes to a one- or two-dimensional map and letting output at a node j influences adjacent nodes in some neighborhood $N_\delta(j)$. In its simplest form, the neighborhood between output nodes is $N_\delta(j) = 0$, i.e., there is no influence between output nodes. In this case, the network is a simple *association network*.

Fig. 7.16 Topology of a Kohonen network

Given that the feature vector has components $i = 1, \ldots, N$ to be assigned to $j = 1, \ldots, C$ clusters, $N \cdot C$ edges connect the input nodes with the output layer with weights w_{ij}. The activation signal $f_j(\mathbf{f})$ at an output node j is the norm of the difference between the feature vector and the vector of nodes $\mathbf{w}j = (w_{1j}\, w_{2j} \ldots w_{Nj})^{\mathrm{T}}$ connecting input nodes with the output node

$$f_j(\mathbf{f}) = \left\|\mathbf{f} - \mathbf{w}_j\right\| = \sqrt{\sum_{i=1}^{N} \left(f_i - w_{ij}\right)^2}. \tag{7.28}$$

The output of an association network is not f_j but the index of the node which has the highest value. If weights w_{ij} are set properly so that $\mathbf{w}_j$ is the center of a cluster $\mathbf{c}_j$, the result will be the index of that cluster.

Training the network seeks to move cluster centers $\mathbf{w}_j$ closer to the feature vectors. As training is unsupervised, the true cluster centers and the true clusters are not known. It is called *reinforcement learning* or *competitive learning*. Weights are initially set to random values. Feature vectors from samples are then fed to the network. The winning neuron in the output layer is determined. Weights leading to the winning neuron are adapted so that it becomes more similar to the feature vector:

$$\mathbf{w}_j^{(n+1)} = \mathbf{w}_j^{(n)} + \alpha\left(\mathbf{f} - \mathbf{w}_j\right), \tag{7.29}$$

where α is the learning rate. The value of α must be less than 1 in order to let the network memorize previous activations ($\alpha = 1$ would cause a perfect adaptation of the weight vector $\mathbf{w}_j$ to the pattern presented by $\mathbf{f}$, and the network "forgets" enforcements due to other vectors presented earlier). Initial values for α between 0.25 and 0.5 are quite common.

Another variant of it (the one that was originally introduced by Kohonen) requires normalization of the weights and gradually reduces the angle between $\mathbf{w}_j$ and $\mathbf{f}$:

$$\mathbf{w}_j^{(n+1)} = \frac{\mathbf{w}_j^{(n)} + \alpha\mathbf{f}}{\left\|\mathbf{w}_j^{(n)} + \alpha\mathbf{f}\right\|}. \tag{7.30}$$

Training is done for all samples. Since the neuron j wins for which $\mathbf{f}$ is most similar to $\mathbf{w}_j$, correction is done only for those weights that need the minimum

amount of change. If samples are clustered in feature space, different output neurons will win for different samples causing a gradual separation of the weights of the weight vectors leading to the different output neurons. Hence, the weight vectors learn feature patterns that are present in the data.

The process is repeated several times (called epochs) until a stable state is reached, i.e., weight change falls below some threshold. The learning rate decreases with time as to enable major corrections in the beginning of the process and increasing the influence of "memory" with time.

Several influences may affect the convergence of the system:

- The selection of the first samples has a much stronger influence on weight changes and cluster assignment than later training. If several artificial cluster centers for a single cluster are generated that are similar to each other, it leads to weak reinforcement of these clusters as later samples will change only one of these centers.
- If the number of cluster centers does not match the true number of centers, some of the weight vectors $\mathbf{w}_j$ will only receive weak enforcement either because superfluous centers divert reinforcement or because some cluster centers actually represent two or more true clusters.
- The initial random distribution of weights may inadvertently have produced patterns that misguide the pattern search, as this represents inappropriate a priori knowledge introduced into the system.

The remedy for slow convergence, convergence to a false optimum, or no convergence is to stop the training if improvement of clustering indicates problems and to repeat it with a new set of randomly selected weights and a new random order of selecting samples.

The success of clustering using an association network depends very much on a good initialization and a correctly selected number of clusters. By Kohonen's introduction of the *neighborhood activation*, a different representation of patterns in the data is achieved (see Fig. 7.17 for examples of neighborhood systems). It acknowledges possible relations between patterns of different clusters. A Kohonen network arranges output neurons in a line or in a regular grid so that every neuron is connected to two or four other neurons.

Reinforcement learning with the consideration of neighboring neurons does not vary much from the strategy stated above. The difference is that—besides updating the weights of the winning neuron j—also those of neurons in the neighborhood

Fig. 7.17 Different neighborhood activation schemes for the same number of nodes in the output layer make different assumptions about relations between the clusters

N_δ (j) of j are updated. If, e.g., $\delta = 5$, $N_\delta(j)$ contains all neurons that are less than 5 neurons away from j (using the maximum norm).

Training a Kohonen network produces a feature map that does not associate samples to cluster centers but maps out regions in the grid of output neurons. Hence, the number of output neurons in a Kohonen network must be larger than the number of clusters expected. Essentially, the Kohonen network detects clusters in high-dimensional feature space by mapping it to the low-dimensional space of node connectivity in the network.

Separation into clusters is a separate step after training the network. It requires some supervision. Representative samples from different clusters are mapped to the network. The response points out representative regions for the clusters. They are either found interactively by displaying the network response (which is much easier than displaying a potential high-dimensional feature distribution) or segmented automatically by observing the activation difference for samples from different clusters.

Training is not guaranteed to converge to an optimal clustering. Techniques to recognize and remedy convergence problems that were suggested for association networks apply here as well. However, introducing the concept of possible interdependency between clusters by connecting output neurons and being able to represent clusters by more than a single neuron simplifies the choice of the number of output nodes.

The number of nodes depends on the complexity of the cluster boundaries to be represented. Reduction of the dimensionality from feature space to a self-organizing map requires that all attributes that characterize a certain class in feature space must be representable by the map. Hence, if the map is too small, different patterns in feature space may not be separable in the map. On the other hand, the map also generalizes cluster attributes. A large map does not only require too much time to train, but it may also learn how to represent the noise in the sample data.

The neighborhood distance usually decreases with time. Initially, δ is set to a large value. It may comprise half the size of the output grid, since—in the beginning—the optimal dimensionality reduction for clusters in feature space to the Kohonen network is unknown. Neighborhood size is decreased with time because a large neighborhood inhibits the separation of regions in different clusters.

Kohonen networks have been used for similar applications than clustering approaches, e.g., for segmenting MR brain images using intensity with (Jiang et al. 2003) and without spatial features (Reddick et al. 1997).

7.3 Concluding Remarks

Segmentation as classification of scene elements solves two problems at the same time. It detects an object and it delineates its boundary. It requires discriminating features of the scene elements such as intensity or a vector of intensities from different imaging channels (e.g., color channels or T_1- and T_2-weighted images in

MRI). If discriminating features exist, classification is simple and leads to an automatic segmentation method. All necessary parameters can be learned from training data.

Compared to classification in general, the dimension of feature space is low and the sample density of training data is high. This allows estimation of the likelihood functions from training data and consequently classification by computing conditional a posteriori probabilities. Clustering techniques may even relieve the user from tedious manual delineation of objects in the training data.

Problems arise when the contrast between different structures (e.g., different organs) is low or if the noise level is too high. Training data should be selected carefully to assure its representativeness for the classification problem.

7.4 Exercises

- Why is it not always appropriate to assume a Gaussian distribution for the likelihood function? Give examples for cases where this is unjustified.
- What is the goal when using a kernel density estimator? What criteria influence the selection of the kernel? Why is it reasonable to consider different kernels for different classes?
- Consider the CT image in Fig. 7.1. How can training data be generated to separate the bone structures from the background for this kind of images?
- What are the requirements for computing the parameters of a Gaussian mixture model? Sketch a situation or give an example where a Gaussian mixture model is the appropriate means.
- What are the parameters that have to be computed for a multi-dimensional Gaussian distribution?
- What kind of information can be represented by the a priori probability term? How is it parameterized for the different kinds of information?
- What kind of knowledge is encoded when the segmentation is defined as probability of a Markov random field?
- How does clustering differ from classification? What roles can clustering play as part of image segmentation? Please give examples.
- What kind of criterion is attempted to be optimized by partitional clustering? Why is the result not necessarily optimal?
- How does *k*-means clustering differ from partitional clustering? What are the advantages of this difference?
- What needs to be initialized for *k*-means clustering? How can this initialization be carried out so that knowledge about the distributions of samples in feature space is included?
- What is the major difference between mean shift clustering and *k*-means clustering?
- What is meant by “mean shift” and how is this used in the clustering method?
- Explain why mean shift clustering is slow when feature space is high.

- What is the purpose of an association network and how can it be used for clustering?
- How is an association network trained?
- How does a Kohonen network differ from an association network? What are the consequences when the two network types are used for clustering?

References

Besag J (1986) On the statistical analysis of dirty pictures. J Roy Stat Soc Ser B (Methodol) 48 (3):259–302

Bouman CA, Shapiro M (1994) A multiscale random field model for Bayesian image segmentation. IEEE Trans Image Process 3(2):162–177

Boykov Y, Veksler O, Zabih R (2001) Fast approximate energy minimization via graph cuts. IEEE Trans Pattern Anal Mach Intell 23(11):1222–1239

Bradley PS, Fayyad UM (1998) Refining initial points for k-means clustering. In: Proceedings of 15th international conference on machine learning. pp 91–99

Celeux G, Forbes F, Peyrard N (2003) Procedures using mean field-like approximations for Markov model-based image segmentation. Pattern Recogn 36(1):131–144

Cheng H, Bouman CA (2001) Multiscale Bayesian segmentation using a trainable context model. IEEE Trans Image Process 10(4):511–525

Comaniciu D, Meer P (2002) Mean shift: a robust approach toward feature space analysis. IEEE Trans Pattern Anal Mach Intell 24(5):603–619

Duda RO, Hart PE, Stork DG (2000) Pattern classification: pattern classification pt.1, 2nd edn. Wiley

Fukunaga K, Hostetler LD (1975) The estimation of the gradient of a density function with applications in pattern recognition. IEEE Trans Inf Theory 21:32–40

Geman S, Geman D (1984) Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Trans Pattern Anal Mach Intell 6(6):721–741

Held K, Kops ER, Krause BJ, Wells WW III, Kikinis R, Muller-Gartner HW (1997) Markov random field segmentation of brain MR images. IEEE Trans Med Imaging 16(6):878–886

Jiang Y, Chen KJ, Zhou ZH (2003) SOM based image segmentation. Lecture notes in artificial intelligence, LNCS 2639:640–643

Kaufman L, Rousseeuw PJ (2005) Finding groups in data: an introduction to cluster analysis. Wiley

Kohonen T (1995) Self-organizing maps. Springer

McLachlan GJ, Krishnan T (1996) The EM algorithm and extensions. Wiley-Interscience

Marroquin JL, Santana EA, Botello S (2003) Hidden Markov measure field models for image segmentation. IEEE Trans Pattern Anal Mach Intell 25(11):1380–1387

Ng HP, Ong SH, Foong KWC, Goh PS, Nowinski WL (2006) Medical image segmentation using k-means clustering and improved watershed algorithm.In: IEEE Southwest symposium on image analysis and interpretation,pp 61–65

Peña JM, Lozano JA, Larrañaga P (1999) An empirical comparison of four initialization methods for the k-means algorithm. Pattern Recogn Lett 20(10):1027–1040

Reddick WE, Glass JO, Cook EN, Elkin TD, Deaton RJ (1997) Automated segmentation and classification of multispectral magnetic resonance images of brain using artificial neural networks. IEEE Trans Med Imaging 16(6):911–918

Shapira L, Avidan S, Shamir A (2009) Mode-detection via median-shift.In: International conference on computer vision (ICCV), pp 1909–1916

Tuzel O, Porikli F, Meer P (2008) Pedestrian detection via classification on Riemannian manifolds. IEEE Trans Pattern Anal Mach Intell 30(10):1713–1727

Tuzel O, Porikli F, Meer P (2009) Kernel methods for weakly supervised mean shift clustering. International conference on computer vision (ICCV 2009):59–68

Zhang J (1992) The mean field theory in EM procedures for Markov random fields. IEEE Trans Signal Process 40(10):2570–2583

Segmentation as a Graph Problem

8

Abstract

2d and 3d images can be mapped on a graph where scene elements are nodes and neighborhood is expressed by edges connecting the nodes. Assigning weights to edges that represent local properties of a good segmentation allows finding a segmentation using optimization methods on graphs. Two such techniques that have been used for segmentation are minimum cost graph cuts and minimum cost paths. Minimum cost paths are computed using variants of Dijkstra's algorithm. Optimal graph cuts have been computed using various different strategies (graph cuts, random walks, normalized cuts, image foresting transform). Methodology, parameterization, advantages, and problems for algorithms that are based on either of the two techniques are discussed in this chapter.

Concepts, Notions, and Definitions Introduced in this Chapter

- Interactive graph cuts;
- Graph cuts to optimize a Markov random field;
- Normalized graph cuts;
- Domain knowledge in graph cuts;
- Fuzzy connectedness and its mapping to a minimum cost path problem;
- Image foresting transform;
- Random walks;
- Similarities between the methods.

Pixels and voxels and the neighborhood relation among them can be represented as a graph. Pixels or voxels are nodes. Adjacency is represented by edges connecting the nodes. Each node carries local features such as intensity, gradient, or some texture measure as attribute. Segmentation then either produces a closed path that encloses a segment or it produces a labeling of the nodes so that all elements

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_8

belonging to a particular segment have the same label. Since there are many different closed paths and many different ways to label nodes, a selection criterion is required for finding the optimal segment boundary or the optimal segmentation. Given this criterion, also an efficient way to arrive at the optimum is required.

Among the many different labelings the one is selected that is optimal w.r.t. the node features and some a priori knowledge about what constitutes a good segmentation. A priori knowledge, also called domain knowledge, may be local relating to expected attributes adjacent nodes (e.g., that adjacent nodes of the same segment have similar attributes and those from different segments have different attributes) or it may be global e.g., expecting a specific shape of a segment.

A number of segmentation techniques that were discussed in Chap. 6, treat the image as a graph. Examples are region merging techniques, region growing or live wire segmentation. While the former two use a graph as convenient way to represent the segmentation process, live wire employs a simple graph theoretic concept to compute a result that is guaranteed to be optimal with respect to some criterion. Representing segmentation as an optimization task on a graph taps into the multitude of efficient methods for computing the desired result. The surveys of Peng et al. (2013) and Camilus and Govindam (2012) present good overviews on graph-based segmentation techniques. Two strategies emerged for computing a segmentation:

- Graph cuts are a way for automatic or semi-automatic segmentations based on region, boundary, and external attributes.
- Optimal paths on graphs can be used to associate scene elements to seeds representing different segments.

For the former, several different concepts have emerged of which the following three will be discussed in detail:

- *Graph cuts* have been presented in 2001 by Boykov and Jolly (2001). They are by far the most often used technique and produce a foreground to background segmentation which can be extended to multi-label segmentations. Pixels with the same label need not to be connected. The method may operate with seeds (but this is not necessary), but pixels of a labeled region do not have to be connected to seed pixels with the same label. Spatial closeness to seed pixels is necessary; however, when the data term of the other pixels contains no information about label membership or if label membership to different labels is similar.
- The image foresting transform (IFT) has been presented in 2004 by Falcao et al. (2004). It essentially assigns label membership to a pixel according to path costs to arrive at a seed from this pixel. The number of seeds with different labels is arbitrary (hence also the number of different regions). Path costs are computed by growing trees from each seed. Together with the fuzzy connectedness criterion (which is a maximum criterion) it requires pixels of a labeled region to be in a connected region that includes the seed of this label. Because of the specific fuzzy connectedness criterion (that rates connectivity along a path of connected

pixels by its weakest link) the segmentation is independent of the location of the seed within the region.

- The random walker has been presented by Grady (2006). It computes the probability of a pixel being reached from different seed. As for the IFT, the random walker requires pixels of a segment to be connected to its respective seed. However, since the connectivity criterion depends on the accumulated probabilities between seed and pixel, the location of the seed influences the segmentation result.

The three strategies will be presented one after the other, but they actually have a lot in common as will be explained in more detail at the end of this chapter.

All strategies define an optimal segmentation as minimization of a functional with *unary* and a *pairwise term*.

The unary term applies to each scene element and represents the connection between the observed data and the desired labeling at this scene element. It may be just intensity, color, or local texture, but it may also include high-level knowledge such as a shape-dependent expected label membership.

The pairwise term adds a priori knowledge about segment membership. It is a regularizer that enforces, for instance, homogenous intensity within the segments or short segment boundaries.

The optimization problem is solved using graph optimization techniques. Depending on the strategy, (graph cuts, random walker, or image foresting transform) it results in slightly different functionals that have emerged independently. The various methods that will be presented and discussed in the following sections apply to images of arbitrary dimension. However, for increased readability, we will restrict the discussion to examples of 2d images with pixels as scene elements.

8.1 Graph Cuts

A *graph cut* creates two or more disconnected subgraphs by removing edges from a connected graph. Graph cuts can be applied to many areas in image processing and image understanding of which segmentation is just one. Representing a segmentation of an image by a graph cut is straightforward (Boykov and Jolly 2001; Boykov et al. 2001). Pixels are represented by nodes, and pairs of nodes representing neighboring pixels are connected by edges. The graph cut consists of a set of edges that specify the segments. The general idea has been used earlier for segmentation, e.g., in Wu and Leahy (1993) who present a method for clustering on graphs or in Géraud et al. (1995) who presented a segmentation on a Markov random field on superpixels. However, the concept of Boykov et al. (2001) found widespread application because of its versatility and generality. It has been applied for numerous segmentation problems in recent years. Graph cuts can be computed for scenes of any dimension and—at least in principle—for arbitrary neighborhood systems defined between scene elements.

While cutting a graph into more than two subgraphs in an optimal manner is difficult, expensive and not yet fully understood, simple and fast algorithms exist for finding an optimal cut of a graph into two subgraphs. Each of these two subgraphs may represent several disconnected components in the image, but each of the components will be associated with the same label. Hence, graph cuts are well suited to solve a foreground segmentation problem. Nodes of the graph then represent pixels and edges represent adjacencies between pixels. If a scene shall be separated into subgraphs with more than two labels, graph cuts are either employed hierarchically by cutting a subgraph again or they are used to correct an existing multi-label segmentation that has been created by some other means.

8.1.1 Graph Cuts for Computing a Segmentation

A *minimum cost graph cut* consists of a set of edges so that the sum of edge weights in the cut is minimal for all possible cuts. In the case of segmentation, edge costs will be defined in a way as to let the minimum cost cut represent a good foreground segmentation. The criterion for an interactive graph cut will combine user knowledge ("this node belongs to the foreground") and local image attributes.

Finding the minimum cut is solved as a maximum flow problem (see Fig. 8.1). Each edge is assigned a flow capacity. All nodes $\mathbf{p} \in P$ representing pixels are connected to two special nodes, the *source S* and the *sink T*, via *terminal links* (*t*-links). Edges connecting neighboring pixels are called *neighbor links* (*n*-links). The set of all *n*-links is *N*. Let us now presume that water is flowing from the source through the graph to the sink.

If the segmentation would be known, weights for all links can be preset to describe the desired segmentation. Terminal links from the source to foreground pixels and from the sink to background pixels are attributed a "0", the remaining *t*-links are attributed a "1". Neighbor links connecting foreground or background

Fig. 8.1 Graph cuts for image segmentation: all pixels are connected to their neighbors via *n*-links and to a source and a sink via *t*-links. The cut separates the source from the sink. All *t*-links that are part of the cut indicate association of a pixel either to foreground (*t*-link to the source) or to the background

pixels are attributed a "1" and n-links connecting a foreground pixel with a background pixel receive a "0" as attribute.

The optimal cut consists of all t-links and n-links that have a value of "0", since the total cost of all edges in this case is 0. The set of edges of the cut consists of all edges connecting foreground pixels with the source, all edges connecting background pixels with the sink, and all edges at region boundaries between foreground and background pixels. Hence, edges to one of the terminal nodes in the cut define the segment label of the segmentation. Edges of the cut that are n-links represent the boundary between segments. Note that neither foreground nor background pixels need to be a single connected component.

Computing a segmentation that is already known is not particularly useful, but the strategy outlined above can be employed as well if the assignment of a pixel to foreground or background is less clear. Instead of attributing pixels with certainty of belonging to the foreground, $R_p(\text{"frg"}) = 0$, $R_p(\text{"bkg"}) = 1$, or background, $R_p(\text{"frg"}) = 1$ and $R_p(\text{"bkg"}) = 0$, probabilities of a pixel of belonging to foreground or background are used to assign weights to the t-links. If these probabilities P for some pixel $\mathbf{p}$ with coordinates $(p_1, p_2, \ldots)$ are $P(\mathbf{p} \in \text{"frg"})$ and $P(\mathbf{p} \in \text{"bkg"})$, then a possible measure suggested by the inventors of foreground graph cuts for segmentation (Boykov and Jolly 2001) is

$$R_p(\text{"frg"}) = -\ln P(\mathbf{p} \in \text{"frg"}) \text{ and } R_p(\text{"bkg"}) = -\ln P(\mathbf{p} \in \text{"bkg"}). \tag{8.1}$$

The probabilities P are specified based on domain knowledge (such as bone in CT ranges from 60 to 3000 HU) or they can be estimated from histogram analysis from training data. The value of this measure is close to zero if the probability of a pixel belonging to foreground or background is high and increases with decreasing probability (see Fig. 8.2).

If nothing else about segment membership is known, all n-links would receive a constant weight, and the application of a graph cut algorithm would be similar to histogram-based thresholding. The power of graph cut algorithms, however, stems from combining this local region attribute with boundary attributes.

Generally, segment boundaries are indicated by change of intensity, texture, or color. Given that a function $f(\mathbf{p})$ exists for each pixel $\mathbf{p}$ that combines these three attributes in a proper manner for the segmentation, the difference $\|f(\mathbf{p}) - f(\mathbf{q})\|$ for two pixels $\mathbf{p}$ and $\mathbf{q}$ being connected by an n-link may serve as weight for this link. The authors of Boykov and Jolly (2001) suggested the following measure (see Fig. 8.3):

$$B_{p,q} = \exp\left[-\frac{\|f(\mathbf{p}) - f(\mathbf{q})\|^2}{2\sigma^2}\right] \cdot \frac{1}{\|\mathbf{p} - \mathbf{q}\|}. \tag{8.2}$$

The function resembles a non-normalized Gaussian distribution. The value of σ is set so that it separates high-frequency and high-amplitude noise from true edge

Fig. 8.2 Flow values from source to pixels (foreground) and from pixels to sink (background) are inversely proportional to the probability of these pixels belonging to the foreground or background

Fig. 8.3 Boundary term depends on the local intensity difference and a smoothing parameter σ. A large value for σ means that small intensity differences are counted as noise and do not receive a low flow value

variation. The latter is assumed to have lower amplitude in the high-frequency range than the noise.

If the n-link weights are used with all t-links having the same weights, the method produces results similar to the watershed transform (which finds all boundaries at zero crossings). However, combining the two measures compensates deficiencies of one of the measures by the other measure. Weights for all the links are summarized in Table 8.1.

Table 8.1 Weights for t-links and n-links for computing a segmentation by graph cuts

Edge type	Weights	Edges
n-link	$B_{p,q} = \exp\left[-\frac{\lVert f(\mathbf{p})-f(\mathbf{q})\rVert^2}{2\sigma^2}\right] \cdot \frac{1}{\lVert \mathbf{p}-\mathbf{q}\rVert}$	all $(p, q) \in N$
t-link	$R(\mathbf{p}, S) = \lambda \cdot (-\ln P(\mathbf{p} \in \text{“}frg\text{”}))$	all (S, p), $p \in P$
t-link	$R(\mathbf{p}, T) = \lambda \cdot (-\ln P(\mathbf{p} \in \text{“}bkg\text{”}))$	all (T, p), $p \in P$

Finding a cut C separating S and T that minimizes the cost of all edges in C minimizes

$$E(C) = \lambda \cdot \left(\sum_{p \in S_c} R(\mathbf{p}, S) + \sum_{p \in T_{c_c}} R(\mathbf{p}, T) \right) + \sum_{(p,q) \in N_C} B_{p,q}. \tag{8.3}$$

Here, N_C is the subset of all edges in N that are part of the cut C, and S_C and T_C are subsets of edges $(\mathbf{p}, S)$ and $(\mathbf{p}, T)$, respectively, that are part of C. The value of λ governs the relative influence of the boundary term with respect to the region term. A small value for λ would be selected if the probability distributions of the region attribute overlap or if they are very flat (see Fig. 8.4). The segmentation could be still successful if a sufficiently high number of n-links on the segment boundary have low values. This behavior has been nicely demonstrated by an example presented in Boykov and Jolly (2001) (Fig. 8.5 shows a sketch of the behavior for different values of λ).

Fig. 8.4 Interactive input for foreground and background produces two distributions that overlap to such a large extent that a segmentation based on intensity alone will not produce the desired grouping into a single foreground object surrounded from background

Fig. 8.5 Intermediate range of values for λ results in the desired segmentation [sketch of results that can be achieved using the method presented in Boykov and Jolly (2001)]

Graph cuts have been extended further to include a priori knowledge about region membership. If foreground or background membership of some pixels is known, their t-link weights can be set accordingly as to ensure that the minimum cost cut does cut the corresponding link. The t-link weights for known foreground pixels $\mathbf{p}_{\text{frg}}$ and background pixels $\mathbf{p}_{\text{bkg}}$ are

$$R(\mathbf{p}_{\text{frg}}, S) = 0 \wedge R(\mathbf{p}_{\text{frg}}, T) = 1 + \max_{\mathbf{p} \in P} \left(\sum_{\mathbf{q}:(\mathbf{p},\mathbf{q}) \in N} B_{p,q} \right) \tag{8.4}$$

and

$$R(\mathbf{p}_{\text{bkg}}, T) = 0 \wedge R(\mathbf{p}_{\text{bkg}}, S) = 1 + \max_{\mathbf{p} \in P} \left(\sum_{\mathbf{q}:(\mathbf{p},\mathbf{q}) \in N} B_{p,q} \right). \tag{8.5}$$

It will always be costlier to cut the t-link from T to a $\mathbf{p}_{\text{frg}}$ or from S to a $\mathbf{p}_{\text{bkg}}$ than to cut any n-link leading to pixel $\mathbf{p}_{\text{frg}}$ and $\mathbf{p}_{\text{bkg}}$, respectively.

Introducing known foreground and background pixels enables user interaction at run-time. Histograms of interactively specified $f(\mathbf{p}_{\text{frg}})$ and $f(\mathbf{p}_{\text{bkg}})$ may be used to estimate region membership probabilities.

Computation of the minimum cut is done by computing the maximum flow between source and sink. The flow from source to sink will be limited by a set of edges which are saturated with flow. This set of edges forms a closed boundary separating the source from the sink[1] irrespective of the dimension of the scene that is represented by the graph.

The iterative Ford-Fulkerson algorithm computes the maximum flow. It augments flow until saturation (see Fig. 8.6). The algorithm keeps a copy G_{residual} of the graph $G = \{V, E\}$ with nodes $V = \{v_i\}$ and edges $E = \{e_j\}$. The weights $w_{\text{res}}(e_i)$ of edges in G_{residual} are the residual flow capacity, which, at initialization, is equal to the capacity $w(e_i)$ of edges in G.

At each iteration, the minimum cost path from S to T is determined in G_{residual} (see Fig. 8.7). Flow from S to T is augmented by the capacity of the edge with smallest weight $w_{\min}$ along this path. Residual capacity along the edges of the path is reduced by $w_{\min}$. Edges E with $w_{\text{res}}(e) = 0$ are removed and are part of the graph cut.[2] The process continues until T becomes unreachable from S. The set of edges removed from the residual capacity graph constitutes the minimum cost graph cut.

[1]If the boundary is not closed, at least one non-saturated edge between source and sink would exist. Flow could be increased until this edge is saturated as well.

[2]It is possible that edges between two neighboring pixels are removed by this procedure which may make a pixel unreachable from S and T. This corresponds to segmentation boundaries that are broader than a pixel in methods such as the watershed transform. If this is undesirable, similar strategies as in WST have to be followed for assigning these pixels either to foreground or background.

Fig. 8.6 Ford-Fulkerson algorithm finds a sequence of edges with maximum flow capacity at each step. Total flow is increased accordingly and remaining capacity is reduced

Fig. 8.7 Computation of the path with maximum remaining capacity is done by computing a minimum cost path on the capacity

The algorithm above is efficient in worst-case analysis for arbitrary graphs. Average performance for computing graph cuts for segmentation can be improved; however, if the specific structure of graphs from images is taken into account (i.e., that all pixels are connected by *t*-links to *S* and *T* and that all *n*-links are local) (Boykov and Komolgorov 2004).

The essence of Boykov's algorithm is the reuse of information from a path search between source and sink instead of starting the path search at each iteration after a flow augmentation from scratch. The disadvantage of this approach is that paths are not necessarily minimum cost paths. It does not change the optimality of the solution but increases computational costs in the worst case. However, on average, the method is faster than the Floyd-Fulkerson algorithm.

For the initial augmentation, Boykov's method searches a minimum cost path by simultaneously carrying out a breadth-first search from source *S* and sink *T*. If the two search trees meet, a minimum cost path is found and flow is augmented.

The method differs from the Floyd-Fulkerson algorithm in the next step. Instead of initiating a new minimum cost path search on the updated residual capacity graph G_{residual}, it reuses the net of the two search trees. Since augmentation will have caused at least one edge to be removed because of saturation, at least one of the search trees will be broken at this edge. Nodes that are no longer connected to *S* or *T* will be called orphan nodes.

Fig. 8.8 Adding a foreground node after a cut requires updating the capacities to this node from source and sink accordingly. Since this may result in edges with negative weight, a constant factor is added to all edges (including those removed because they are saturated)

In a restoration step, orphan nodes will be attempted to be reconnected to the one search tree to which they originally belonged. Hence, a new flow between source and sink has been restored although the minimum cost property is not necessarily given. Again, the bottleneck edge along this path is determined, flow is augmented, and G_{residual} is updated accordingly. The process terminates when no path can be determined, i.e., when orphan nodes remain unconnected in the restoration step.

The algorithms above assume that weights along edges are positive, which can be easily assured and which is true for the weighting scheme given by Eqs. (8.2)–(8.5).

For a given image, the computation of a graph cut needs to be repeated from scratch when weights are changed. This may happen, for instance, if the user adds foreground or background pixels because the current segmentation result is unsatisfactory.

The complete recomputation of maximum flow can be avoided using a simple trick. This will be demonstrated by the example of adding new foreground nodes $\mathbf{p}_{\text{frg},1}, \ldots, \mathbf{p}_{\text{frg},N}$. It changes the flow capacity at edges $(S, \mathbf{p}_{\text{frg},i})$ by the positive value $(K - \lambda) \cdot R_{\text{bkg}}(\mathbf{p}_{\text{frg},i})$. It also changes the flow capacity at edges $(T, \mathbf{p}_{\text{frg},i})$ by the negative value $-\lambda \cdot R_{\text{frg}}(\mathbf{p}_{\text{frg},i})$. Computing maximum flow can be continued using the previous result in the residual capacity graph G_{residual} instead of starting with G if residual capacities of edges $(S, \mathbf{p}_{\text{frg},i})$ and $(T, \mathbf{p}_{\text{frg},i})$ are changed accordingly. If one of these edges no longer exists in G_{residual} because it had been saturated, it has to be included again.

Updating the residual capacity graph as explained above may assign negative weights to some edges. This is remedied by computing $w_{\min} = \min_{e \in G_{\text{res}}} w(e)$ and subtracting it from all edge weights. It changes the maximum flow by a constant but does not change the set of edges in C (see Fig. 8.8).

8.1.2 Graph Cuts to Approximate a Bayesian Segmentation

The graph cut technique outlined above can also be used as part of a segmentation procedure that optimizes label classification based on a Markov random field model presented in Sect. 7.1.3. In Boykov et al. (2001), the problem is defined as an energy minimization task, which makes it conceptually similar to some of the active contour segmentation schemes presented in the next chapter:

- *Interaction costs* $V(f(\mathbf{p}), f(\mathbf{q}))$ represent what is the a priori probability term of the MRF formulation and what is called internal energy in active contours. Interaction costs are a smoothing term which is assumed to be minimal if labels $f(\mathbf{p})$ of a pixel $\mathbf{p}$ and those of its neighbors $\mathbf{q}$ are equal. Neighborhood for the set of neighbor pairs N can be defined in an arbitrary manner.
- *Data costs* $D(f(\mathbf{p}))$ represent what the likelihood function is in the MRF formulation and what is external energy in active contours. The value of $D(f(\mathbf{p}))$ is assumed to be minimal if features of pixels $\mathbf{p} \in G$ having some label are similar to feature values expected for this kind of segment.

The goal of segmentation is to find a labeling S_{opt} that minimizes a weighted combination of internal and external energy:

$$S_{\text{opt}} = \arg\min_{S} E(S) = \sum_{(\mathbf{p},\mathbf{q})\in N} V(f(\mathbf{p}), f(\mathbf{q})) + \gamma \sum_{\mathbf{p}\in G} D(f(\mathbf{p})) \tag{8.6}$$

The weight γ depends on the kind of image and has to be specified by the user. Data costs are weighted high if data is reliable (high contrast, low noise), while otherwise interaction costs are weighted higher. Data costs $D(f(\mathbf{p}))$ can be any non-negative function of $f(\mathbf{p})$. It could be, for instance, the difference of the actual value of $\mathbf{p}$ compared to an expected value $E_f(\mathbf{p})$ for pixels belonging to a segment labeled with f. Interaction costs need to be a metric or a semi-metric on the labels l. It is a reasonable constraint which requires non-negativity, symmetry, and $V(f(\mathbf{p}), f(\mathbf{p})) = 0$. If it is a metric, the triangle inequality must hold as well.

Automatic segmentation by graph cut requires some initial labeling that assigns a segment label to each pixel. In principle, this labeling can be arbitrary. There are two reasons why segmentation should be initiated with a good guess nonetheless:

- Most optimization schemes are iterative local relabeling procedures that are not guaranteed to terminate in the global minimum.
- It is not guaranteed that the global minimum really is the desired segmentation.

Hence, the initial labeling should be as close as possible to the desired final result. Such an initial segmentation can be created by first omitting the interaction term V. The initial label $f(\mathbf{p})$ for a scene element $\mathbf{p}$ is then the label of which the expected feature vector is closest to the feature vector observed at $\mathbf{p}$. It is a reasonable strategy since the interaction costs serve as smoothness constraint to reduce the effects of noise on the segmentation (see Fig. 8.9 for a schematic view). If omission of the smoothness constraint does not produce a good initial segmentation, the noise level is so high that the final segmentation is mainly influenced by the a priori knowledge. In such case, using such a simple model of a priori knowledge is inappropriate altogether and a different segmentation strategy should be employed.

Fig. 8.9 Automatic segmentation requires an initialization. This could be a segmentation based on the data term, since it can be assumed that this will be a noisy version of the unknown true segmentation

Given the initialization, graph cut optimizes the labeling according to the interaction term enforcing equal labels between neighboring scene elements and to the data term given by the likelihood function. Two different methods have been presented to achieve this goal (Boykov et al. 2001, see also Fig. 8.10):

- $\alpha - \beta$ *swap moves* find an optimal swap of labels between all scene elements having the labels α and β.
- α*-expansion moves* expand regions labeled with α into all regions with other labels.

If the scene is separated into two different regions (e.g., foreground and background), both methods produce optimal results. If the scene has more than two different segment types, it has been shown by Boykov et al. (2001) that the α-expansion moves produce a result of which the total energy is at most twice the minimum energy that is searched. No such assertion can be made for $\alpha - \beta$ swap moves.

A segmentation can be achieved by either iterating several times through all possible $\alpha - \beta$ swap moves for all label combinations (α, β) or by repeatedly carrying out α-expansion moves. A set of moves through all possible (α, β), combinations for $\alpha - \beta$ swaps or through all possible labels α for α-expansions is called a *cycle*. The iterative procedure stops when iterating through a cycle does not reduce the total energy $E(S)$. Termination after a fixed number of cycles is guaranteed, since D is non-negative and V is at least a semi-metric.

Although it sounds as if α-expansion moves should be preferred to swap moves, this should be taken with a grain of salt. Assuring that the energy of the final

Fig. 8.10 $\alpha - \beta$ swap finds an optimal label swap between all sites labeled with α and β. Other regions are not changed. An α-expansion expands the region labeled with α into all other regions

labeling is at most twice as high as the minimum energy tells little about the actual labeling error, i.e., the number and location of mis-labeled scene elements.

For computing each move (swap move or expansion move), a graph has to be created from the scene. Nodes in this graph represent a subset of the scene elements. Two nodes are connected by an edge if the corresponding scene elements are neighbors by some neighborhood definition. Neighborhood could be the usual 4- or 8-neighborhood for pixels (6- or 26-neighborhood for voxels), but in general, every kind of neighborhood is admissible.

The graphs for $\alpha - \beta$ swap moves and for α-expansion moves are different from each other. The graph for an $\alpha - \beta$ swap move consists of the subset of all nodes $\{\mathbf{p}_\alpha\} \cup \{\mathbf{p}_\beta\}$ representing scene elements labeled with $f(\mathbf{p}) = \alpha$ or $f(\mathbf{p}) = \alpha$ and two terminal nodes $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Nodes $\mathbf{p}, \mathbf{q} \in \{\mathbf{p}_\alpha\} \cup \{\mathbf{p}_\beta\}$ are connected by an edge $e(\mathbf{p}, \mathbf{q})$ if they are neighbors. All nodes $\mathbf{p}$ are connected with the two terminal links by edges $t(\mathbf{p}, \boldsymbol{\alpha})$ and $t(\mathbf{p}, \boldsymbol{\beta})$, respectively, (see Fig. 8.11). Table 8.2 lists the weights given to the edges.

This table has some similarities to the table for interactive graph cuts. Again, for all edges in the graph, interaction costs V are computed. The main difference is in the definition of the cost for the edges to the two terminals. Additionally to the data cost D of scene element $\mathbf{p}$, interaction costs are added from neighboring nodes that are labeled neither α nor β. This is necessary because a cut that results in a

Fig. 8.11 Graph of all nodes currently labeled α or β for an $\alpha - \beta$-swap move

Table 8.2 Weights for an $\alpha - \beta$ swap

Edge type	Weight	Edges
$e(\mathbf{p}, \mathbf{q})$	$V(f(\mathbf{p}), f(\mathbf{q}))$	all $(\mathbf{p}, \mathbf{q}) \in P_{\alpha,\beta,}(\mathbf{p}, \mathbf{q}) \in N$
$t(\mathbf{p}, \alpha)$	$\sum_{q \in N_p, q \notin P_{\alpha,\beta}} V(f(\mathbf{p}), f(\mathbf{q})) + \gamma D(f(\mathbf{p}))$	all $(\mathbf{p}) \in P_{\alpha,\beta}$
$t(\mathbf{p}, \beta)$	$\sum_{q \in N_p, q \notin P_{\alpha,\beta}} V(f(\mathbf{p}), f(\mathbf{q})) + \gamma D(f(\mathbf{p}))$	all $(\mathbf{p}) \in P_{\alpha,\beta}$

relabeling also changes potential relabeling costs to other nodes (the interaction cost between label α and another label γ may be different to the cost between labels β and γ). The set of edges in the graph cut contains edges connecting pixel nodes with terminal label nodes. Pixels are assigned the label of these terminal nodes. It has been shown in Boykov et al. (2001) that relabeling nodes according to the minimum cut on this graph using the edge costs from Table 8.2 maximally decreases the total energy with respect to the two labels. It is not the global optimum, however, since the swap does not consider costs from pixels with other labels γ.

The graph for an α-expansion consists of all nodes representing pixels in the scene and two terminal nodes $\boldsymbol{\alpha}$ and $\bar{\alpha}$. Auxiliary nodes $\mathbf{a}$ are introduced between each pair of nodes $\mathbf{p}$ and $\mathbf{q}$ that have different labels $f(\mathbf{p}) \neq f(\mathbf{q})$ (see Fig. 8.12). These auxiliary nodes are connected by edges to nodes $\mathbf{p}$ and $\mathbf{q}$ via edges $e(\mathbf{p}, \mathbf{a})$ and $e(\mathbf{a}, \mathbf{q})$ and to the terminal node $\bar{\alpha}$ via $t(\mathbf{a}, \bar{\alpha})$. Introduction of auxiliary nodes is necessary in order to represent the costs that incur when a label change takes place. Edge costs for all edges are listed in Table 8.3.

Similar to the $\alpha - \beta$ swap, it can be shown that computing a minimum cut using the graph with edge weights as given above and relabeling nodes according to the links to either $\boldsymbol{\alpha}$ or $\bar{\alpha}$ produces the maximum decrease in total energy that can be achieved by expanding the α-labels into all other pixels. The graph changes after every step, since number, placement, and weights of auxiliary nodes change.

Fig. 8.12 Graph for an α-expansion consists of all nodes whether being labeled α or not, plus auxiliary nodes between neighboring nodes with label α and some other label

Table 8.3 Weights for an α-expansion move

Edge type	Weight	Edge
$t(\mathbf{p},\bar{\alpha})$	∞	all $(\mathbf{p}) \in P_\alpha$
$t(\mathbf{p}, \bar{\alpha})$	$\gamma D(f(\mathbf{p}))$	all $(\mathbf{p}) \notin P_\alpha$
$t(\mathbf{p},\alpha)$	$\gamma D(\boldsymbol{\alpha})$	all $(\mathbf{p}) \in P$
$e(\mathbf{p}, \mathbf{a})$ $e(\mathbf{a}, \mathbf{q})$ $t(\mathbf{a}, \bar{\alpha})$	$V(f(\mathbf{p}), \boldsymbol{\alpha})$ $V(\boldsymbol{\alpha}, f(\mathbf{q}))$ $V(f(\mathbf{p}), f(\mathbf{q}))$	all $(\mathbf{p}, \mathbf{q}) \in N, f(\mathbf{p}) \neq f(\mathbf{q})$
$e(\mathbf{p}, \mathbf{q})$	$V(f(\mathbf{p}), \boldsymbol{\alpha})$	all $(\mathbf{p}, \mathbf{q}) \in N, f(\mathbf{p}) \neq f(\mathbf{q})$

8.1.3 Problem-Specific Adaptation of *t*-Link Weights

Graph cut segmentation has been used by numerous authors. Many of them presented adaptations of weights to a specific problem. The main reason for adapting *t*-link weights is that the Gaussian distribution of intensities in the foreground and background do not model the true intensity variation particularly well. The background often consists of many different objects, each with its own characteristic intensity and the foreground, e.g., a liver with vessels, cysts, and tumors, may not be as homogeneous as predicted by the Gaussian. The paper on multi-object segmentation with initial seeds stemming from a clustering step by Kéchichian et al. (2013) contains a good state-of-the-art discussion on using priors in graph cut segmentation.

In van der Lijn et al. (2008), the authors use a traditional Bayesian MAP estimator to compute the *t*-link weights from intensity-dependent probabilities for hippocampus segmentation in MRI. The probabilities for foreground and background are estimated from samples. The foreground is modeled as a Gaussian distribution. The background probability is modeled using kernel density estimation of the intensity histogram in order to account for the various objects with different intensities in the background. The unary term of the prior also contains a *spatial expectation term* of labels from multi-atlas registration.

Instead of using a kernel density estimator for complex probability densities, a Gaussian mixture model (GMM) may be used as well. In Zheng et al. (2013), the optical disk and cup in color images are segmented with a trained GMM for the *t*-link weights. Since the simple shape of the object to be segmented is known, deviation from a circle based on known circle center is added as directional penalty (see Fig. 8.13). This is a similar constraint than the 3d blob constraint of Funka-Lea et al. (2006) presented in the next section. A distance constraint between the two concentric circles representing the optical disk and the cup is included as well.

Data may also be preprocessed to enhance separability of segments based on the unary term as in Salah et al. (2011) who use kernels to represent transformation of input features into a space that optimizes differentiation of the segments by the unary term.

Fig. 8.13 Zheng et al. (2013) the optic disk and the cup are segmented to estimate the cup-to-disk ratio. Unary costs are modeled by GMMs to accommodate the gray value variation in foreground and background. A shape constraint has been added to the binary costs that favors circular boundaries around a given center (retina image adapted from Häggström (2014), public domain)

If the probability density function is complex or the number of samples is too low, parameter-free estimations such as random forests can be used to train the data term (Mahapatra et al. 2013).

Sometimes, local texture is more descriptive than the pixel or voxel intensity. In this case, *superpixels* (see Sect. 5.7) or *supervoxels* are used to compute the local descriptor. An example that also adds some shape information to the segmentation is Lucchi et al. (2012). The graph cut is carried out on SLIC supervoxels (Radkrishna et al. 2010, see also Sect. 5.7). Supervoxel features are the histogram and a ray descriptor, which computes features of the nearest edge along rays. Edges are created by applying the Canny edge detector.

Instead of pixel intensity, textures may also been used in graph cut segmentation. Textures could be those presented in Sect. 5.10. An example is Danesh et al. (2014) who use wavelet features for OCT segmentation. Textures may be computed at each scene element from local image characteristics. If misleading texture values at texture boundaries shall be avoided, the graph cut may be carried out on superpixels and textures are computed for each superpixel within its boundaries.

The *grab cut* algorithm of (Rother et al. 2004) applies the graph cut algorithm iteratively to find the best weights for the *t*-links. The user is expected to draw a rectangle around the object to be segmented and the inside of this rectangle is taken as first approximation for a foreground segmentation (see Fig. 8.14). Intensities inside and outside of the rectangle are used to estimate two GMMs to represent foreground and background intensity distributions, respectively. Graph cut segmentation is applied and the result is used as new segmentation estimate. The process is repeated until convergence. Since the initial guess excluded foreground pixels from the estimate of the background GMM and the foreground mask is usually dominated by the object to be segmented, the iterative procedure is very likely driven toward the best estimate for foreground and background intensities.

Fig. 8.14 GrabCut produces a series of improving segmentations. Starting from a rough first approximation, foreground, and background histograms are used to specify the unary term of the graph cut. The result is used as second approximation. It can be seen that with improved segment quality the difference between foreground and background histogram increases leading to a better separation of foreground and background based on the unary term

Segmentation errors may still remain because the two GMMs overlap. They may be reduced similar to the original interactive graph cut of Boykov and Jolly (2001) by adding brushes in the foreground and background as further seeds.

8.1.4 Adding Generic Domain Knowledge

Basic graph cut segmentation produces segments based on homogeneity in the segments and boundary length. Its performance can be enhanced if additional domain knowledge is integrated into the cost function.

A blob component has been used (Funka-Lea et al. 2006) to separate the heart from the background in CT images. The blob component is realized by adding a term to Eq. (8.6) that penalizes direction deviations of a line segment **pq** of a cut to a line from a prespecified blob center **c** to the location of **p** (see Fig. 8.15). This is a way to promote convex structures around a prespecified blob center (which in this case is a location in the center of the heart specified by the user). The approach has been generalized by Veksler (2008) who extended it to arbitrary star-convex shapes around a selected center. It allows including a variety of generic shape constraints into the graph cut formulation.

Fig. 8.15 Blob constraint penalizes cutting through edges between pixels the more the orientation varies from the orientation of a line between the blob center and the pixel. In this example, cutting the edge depicted in (**a**) is cheaper than cutting the edge depicted in (**b**)

Segmentation of line-like extrusions of a structure has been made possible by a method presented by Vicente et al. (2008). Such extrusions are often not left intact by regular graph cut segmentation, as it requires too many edges to be part of the cut. In Vicente et al. (2008), an additional user-specified constraint augments an existing segmentation, e.g., from graph cut segmentation, to add missing extrusions to the result (see Fig. 8.16). The user is required to place seed points into extremal points of the missing extrusion. The constraint then forces these points to be connected to the existing segmentation. Different kinds of conditions are given for this connection. A heuristic solution is presented that combines graph cut with a Dijkstra-like path searching algorithm to connect the seeds with the original segmentation.

Spatial connectivity to seed point locations cannot be guaranteed by graph cuts. This is usually a wanted feature. If connectivity is required, other methods such as the random walker or the image foresting transform (both described below) are probably better choices. Nonetheless, connectivity constraints have also been introduced to graph cuts, e.g., by Chen et al. (2012) who constructed a graph which enforces spatial connectivity for graph cuts.

Fig. 8.16 Without extra constraints, the dendritic spines in the microscopy above would not be segmented. In the work of Vicente et al. (2008) the graph cut is complemented by a path searching algorithm that connects user-specified seed points with the graph cut segmentation

Sometimes, two different image sources depict the searched object, for instance, if t_1- and t_2-weighted, MRI of the same subject have been created. The additional information may just be used for a multi-dimensional probability distribution. Alternatively, the images may be used to initiate a restricted co-segmentation solution. Co-segmentation in general attempts to find the same kind of object in two or more images under the assumption that the object has similar properties in these images (Vicente et al. 2011; Steffen et al. 2016). In our case, it is known that exactly the same object is positioned at exactly the same location but has different appearance in the two images. This may be modeled as a simultaneous graph cut on two graphs representing the two images. For reflecting dependencies between the two images, the two graphs need to be interlinked. An example for this is (Han et al. 2011), who co-segment rigidly registered PET and CT images by defining two separate graphs for graph cut for each of the two images and intermediating nodes for each CT-PET pixel. The edge costs for edges that connect the two graphs with the intermediating nodes represent costs incurring from pixels that receive different labels in PET and CT.

Priors can be even more specific than the ones here if a model predicts a specific shape. This will be discussed in Chap. 11 on the use of shape models.

8.1.5 Normalized Graph Cuts

Graph cuts sometimes produce undesired results because the minimum-cut-maximum-flow tends to cut as few edges as possible (see Fig. 8.17). In interactive graph cuts, it may produce segments with boundaries very close to the foreground or background seed points. This is particularly the case if the data terms for foreground and background have very different characteristics (e.g., when one of them has to allow for a much higher variation of feature values than the other). In such case, it can be helpful to add a size constraint into the equation that requires

Fig. 8.17 Graph cut segmentation may stop too early because the inhomogeneous foreground on the right results in a weak data constraint. Hence, smoothness overtakes, which causes a short boundary close to the initialization

segments to have similar size. It makes sense if the goal of segmentation is not to extract a specific object—of which the size may be very different from that of the background—but to produce agglomerations of pixels that shall be further processed for analysis. Normalizing the segmentation by segment size is comparable with similar segmentation methods such as the watershed transform or region merging, however, with the difference that it optimizes a criterion that includes both region and boundary terms.

In Shi and Malik (2000), a method is suggested that incorporates such a normalization criterion into the graph cut procedure. The *normalized graph cut* segments an image into regions with one of two labels **a** or **b** with minimal costs. Costs consist of the original graph cut costs *cut*(**a**, **b**) weighted by association costs *assoc* (**a**, **v**) and *assoc*(**b**, **v**) between nodes of a segment to all nodes **v** in the scene. The total cost of a normalized cut is then

$$NCut(\mathbf{a},\mathbf{b}) = \frac{cut(\mathbf{a},\mathbf{b})}{assoc(\mathbf{a},\mathbf{v})} + \frac{cut(\mathbf{a},\mathbf{b})}{assoc(\mathbf{b},\mathbf{v})} \tag{8.7}$$

Unfortunately, computing the optimal cut is NP-complete. Hence, Shi and Malik (2000) present an approximation of the result. The problem is mapped on a linear equation system representing the graph as a matrix and the unknown labeling by an indicator vector. The indicator vector **x** for nodes $\mathbf{N} = \{\mathbf{n}_1, \ldots, \mathbf{n}_M\}$ in the graph has elements x_i as follows:

$$x_i = \begin{cases} 1, & \text{if } \mathbf{n}_i \in A \\ -1, & \text{if } \mathbf{n}_i \in B \end{cases} \tag{8.8}$$

In $M \times M$ weight matrix, **W** denotes costs of the edge between nodes n_i and n_j. The value of an element w_{ij} is

$$w_{ij} = \begin{cases} e(i,j), & \text{if } \mathbf{n}_i \text{ and } \mathbf{n}_j \text{ are connected by an edge,} \\ 0, & \text{otherwise.} \end{cases} \tag{8.9}$$

Furthermore, a diagonal matrix **D** is created with entries d_{ii} representing association costs of a node $\mathbf{n}_i$ to all other nodes:

$$d_{ii} = \sum_{j=1,M} w_{ij}. \tag{8.10}$$

The cost function can then be rewritten as

$$NCut(A,B) = NCut(\mathbf{x}) = \frac{\sum_{\{(i,j)\},x_i>0,x_j<0} -w_{ij}x_ix_j}{\sum_{\{i\},x_i>0} d_{ii}} + \frac{\sum_{\{(i,j)\},x_i<0,x_j>0} -w_{ij}x_ix_j}{\sum_{\{i\},x_i<0} d_{ii}}. \tag{8.11}$$

Introducing a cost ratio k for the association costs of scene elements with label **a** to the total association cost

$$k = \frac{\sum_{\{i\},x_i>0} d_{ii}}{\sum_i d_{ii}} \tag{8.12}$$

and setting

$$\mathbf{y} = (\mathbf{1}+\mathbf{x}) - \frac{k}{1-k}(\mathbf{1}-\mathbf{x}) \tag{8.13}$$

it can be shown (Shi and Malik 2000) that the cost of a cut specified by an indicator vector **x** is given by

$$NCut(\mathbf{x}) = \frac{\mathbf{y}^T(\mathbf{D}-\mathbf{W})\mathbf{y}}{\mathbf{y}^T\mathbf{D}\mathbf{y}}. \tag{8.14}$$

Finding an optimal cut requires to find a vector **y** that minimizes $NCut(\mathbf{x})$. Solutions can be generated by solving the associated eigenproblem for

$$(\mathbf{D}-\mathbf{W})\mathbf{y} = \lambda\mathbf{D}\mathbf{y}. \tag{8.15}$$

The solution for the second largest eigenvalue, i.e., the associated eigenvector, produces the vector **y** and consequently the vector **x** to create the labeling. The result only approximates the optimal solution, since **x** may contain values that are neither 1 nor −1. In a final step, the real-valued vector is mapped onto an indicator vector.

The NCut algorithm can be extended to separate an image into more than two labels in two ways. Further separations can be generated by repeated subdivision if the NCut algorithm is applied again on a labeled scene. This is the most stable but a rather slow solution for multi-label segmentation. Another option is to use eigenvectors associated with smaller eigenvalues. Due to the way to approximate eigenvalues and due to the fact that the result is only an approximation of the true optimal decomposition, this is much less stable, however. Alternative ways to solve the multi-label segmentation problem were proposed, e.g., by Ng et al. (2002), Tolliver and Miller (2006), Rak et al. (2013).

In medical image analysis applications the advantage of the NCut algorithm being an unsupervised segmentation method may become a disadvantage, as object-specific information (e.g., the location or shape of an organ) is not part of the segmentation model. It has been used several times for vertebra segmentation in

MRI, see, e.g., (Gamio et al. 2004), but works better if a region of interest is separated containing a single vertebra prior to NCut segmentation (Huang et al. 2009).

8.2 Segmentation as a Path Problem

Instead of using graph cuts, segmentation can also be modeled as a minimum cost path problem. This is the underlying principle of the live wire method discussed in Sect. 6.9, where part of a segment boundary has been modeled as a path in a graph representing a 2d scene. It has been extended to segmentations based on regional attributes in several ways which will be presented below. Common to all approaches is that there is no restriction on the number of dimensions of the scene to be segmented. Another common aspect is that all approaches require seed elements similar to seeded region growing. Segmentation is then solved as a minimum cost path problem between each pixel and the seed elements. A pixel receives the label of a seed pixel which can be reached with the lowest cost.

8.2.1 Fuzzy Connectedness

Fuzzy connectedness is a concept to describe connectivity between sets that have a spatial component so that neighborhood between elements of the set can be defined. The concept has been exploited for image segmentation by relating connectedness to segmentation goals (Udupa and Samarasekara 1996). An exact but rather mathematical description can be found in Udupa and Saha (2003).

Fuzzy connectedness between two scene elements is given by *local affinity* values a of a path between the two elements. The affinity of a path is given by the smallest local affinity between neighboring elements along the path. Fuzzy connectedness μ between two elements $\mathbf{p}_j$ and $\mathbf{p}_k$ is then the minimum affinity of the strongest path between the elements (see Fig. 8.18):

$$\mu(\mathbf{p}_j, \mathbf{p}_k) = \max_{p(\mathbf{p}_j,\mathbf{p}_k)\in P(\mathbf{p}_j,\mathbf{p}_k)} \left[\min_{(\mathbf{p}_{s1},\mathbf{p}_{s2})\in p(\mathbf{p}_j,\mathbf{p}_k)} a(\mathbf{p}_{s1}, \mathbf{p}_{s2})\right], \tag{8.16}$$

where $P()$ is the set of all paths between two pixels, $p()$ is a path and $(\mathbf{p}_{s1}, \mathbf{p}_{s2})$ are adjacent pixels under some adjacency relationship.

The affinity value a between two pixels is a local property which depends on the distance between the two pixels, on the homogeneity of the feature values of the two pixels, and on the deviation of the feature values from expected values. Affinity decreases with increasing distance, inhomogeneity, and dissimilarity of the features to expected features.

In principle, any reasonable measure for distance, homogeneity, and dissimilarity to some expected value can be used to measure affinity. In Udupa and

Fig. 8.18 Fuzzy connectedness between two pixels is defined by the maximal local affinity of all paths between the two pixels

Saha (2003), however, an elegant way was presented to define homogeneity and dissimilarity to expected semantics. Details can be found in Udupa and Saha (2003) but the essence of the argumentation is that inhomogeneity can be caused by two effects: random fluctuation and the fact that the two pixels $\mathbf{p}_j$ and $\mathbf{p}_k$ belong to different objects. Since only the latter is relevant in terms of the affinity measure, the authors

- estimate homogenous regions of interest around the two scene elements,
- compute measures of deviation under two different hypotheses ($\mathbf{p}_j$ is brighter than $\mathbf{p}_k$ or $\mathbf{p}_k$ is brighter than $\mathbf{p}_j$),
- make the assumption that the higher of the two deviation measures ($d_{\max}$, $d_{\min}$) depends on random variation plus any variation caused by $\mathbf{p}_j$ and $\mathbf{p}_k$ belonging to different objects, while the other represents just variation by random noise (which simply means that the SNR should be better than 1),
- and finally, use the difference $d_{\max} - d_{\min}$ for computing a homogeneity measure.

A similar approach is used for computing the dissimilarity from expected feature values. The main difference is that the weighting does not depend on the relative difference in appearance at $\mathbf{p}_j$ and $\mathbf{p}_k$ but on a probability given some expected intensity distribution functions.

Given the affinity measure, fuzzy connectedness can be used to associate pixels to user-specified seed elements. In Herman and Carvalho (2001), it was shown that computing fuzzy connectedness for every pixel with respect to given seed pixels can be done efficiently by applying Dijkstra's algorithm to compute a minimum cost path.

Fuzzy connectivity for segmentation can be applied in an absolute and a relative manner (see Fig. 8.19). Absolute connectivity is specified if a threshold on fuzzy connectivity can be given that is required for all pixels of a segment. Given a seed pixel, all pixels are detected that fulfil this criterion with respect to the seed. This bears some similarity to region growing methods such as those presented in Chap. 6.

Fig. 8.19 Segmentation using absolute fuzzy connectedness puts the segment boundary between pixels of which fuzzy connectedness falls below some prespecified threshold. In relative fuzzy connectedness, connectedness for each pixel to a set of seed pixels is computed. Pixels receive the label of the seed to which the connectedness is maximal

However, different to those, fuzzy connectivity is not heuristic and guarantees to optimize a (fuzzy) membership function. Furthermore, region growing works only with a simple neighborhood relationship and homogeneity features cannot be defined. The latter leads to serious problems when the implicit homogeneity assumption of region growing (the region has a constant intensity value) is not true.

As a relative method, segmentation using fuzzy connectedness is even more versatile. In this case, fuzzy connectivity of a pixel to each of K different seed pixels is computed. The pixel receives the label of the one seed to which it has the highest fuzzy connectedness. The segmentation produces as many connected segments as there are seed pixels and does not require a predefined threshold on fuzzy connectedness.

8.2.2 The Image Foresting Transform

Using, the *image foresting transform* (IFT) (Falcao et al. 2004) for segmentation bears some similarity to segmentation by interactive graph cut segmentation. The IFT creates a segmented image by finding minimum cost paths in a graph with nodes representing pixels and edges representing neighborhood between pixels. The IFT allows separation of a scene into an arbitrary number of differently labeled segments. Minimum cost paths to each pixel may originate at arbitrary pixels, but specific seed locations may be selected by proper definition of the initial path costs at every pixel. The result is a forest of trees. Pixels being roots of a tree connect to all nodes of the graph that can be reached from this root by a minimum cost path (see Fig. 8.20).

The graph $G = \{V, E\}$, for which the IFT is computed, consists of nodes $V = \{\mathbf{p}_1, \mathbf{p}_2, \ldots\}$ representing pixels and edges $E = \{e_1, e_2, \ldots\}$ connecting adjacent pixels. Adjacency may be defined such that two pixels are adjacent if they share a common corner, edge, or surface. Other adjacency relationships are possible as well (similar to graph cuts). An edge e_j connecting $\mathbf{p}_{j,1}$ with $\mathbf{p}_{j,2}$ carries a cost $c(e_j) = c(\mathbf{p}_{j,1}, \mathbf{p}_{j,2})$. If region-based segmentation is the objective, this cost is related to the likelihood of $\mathbf{p}_{j,1}$ and $\mathbf{p}_{j,2}$ belonging to the same region. Furthermore, nodes $\mathbf{p}$ carry a cost $h(\mathbf{p})$, which is called the *node load*. In region-based segmentation, the node load reflects the likelihood of this node being a potential seed of a segment. Given a

Fig. 8.20 Image foresting transform computes the minimum cost path from seed pixels to all other pixels. Segmentation is then given by assigning labels from the seed which can be reached with lowest costs. The method bears similarity to segmentation using fuzzy connectedness. However, the length of the path influences computed costs

scene represented by a graph, the IFT will compute the shortest path from one of the seeds to every node in the graph. The method for computing the IFT is a modified and optimized version of Dijkstra's method for computing minimum cost paths.

Path costs $\pi(\mathbf{p}_{\text{root}}, \mathbf{p}_{\text{cur}})$ between a root node $\mathbf{p}_{\text{root}}$ and the current node $\mathbf{p}_{\text{cur}}$ are defined by an accumulation function $\pi_{acc}[\pi(\mathbf{p}_{\text{root}}, \mathbf{p}_{\text{prev}}), c(\mathbf{p}_{\text{prev}}, \mathbf{p}_{\text{cur}})]$ of the path cost $\pi(\mathbf{p}_{\text{root}}, \mathbf{p}_{\text{prev}})$ to the parent node $\mathbf{p}_{\text{prev}}$ of node $\mathbf{p}_{\text{cur}}$ and the edge cost $c(e) = c(\mathbf{p}_{\text{prev}}, \mathbf{p}_{\text{cur}})$ of an edge connecting $\mathbf{p}_{\text{prev}}$ with $\mathbf{p}_{\text{cur}}$ (see Fig. 8.21). Various definitions of the path cost function π_{acc} are possible, such as

$$\pi_{\text{acc}}[\pi(\mathbf{p}_{\text{root}}, \mathbf{p}_{\text{prev}}), c(\mathbf{p}_{\text{prev}}, \mathbf{p}_{\text{cur}})] = \pi(\mathbf{p}_{\text{root}}, \mathbf{p}_{\text{prev}}) + c(\mathbf{p}_{\text{prev}}, \mathbf{p}_{\text{cur}}), \tag{8.17}$$

which adds the edge cost to the cost of the path from $\mathbf{p}_{\text{root}}$ to $\mathbf{p}_{\text{prev}}$, or

$$\pi_{acc}(\pi(\mathbf{p}_{\text{root}}, \mathbf{p}_{\text{prev}}), c(\mathbf{p}_{\text{prev}}, \mathbf{p}_{\text{cur}})) = \max(\pi(\mathbf{p}_{\text{root}}, \mathbf{p}_{\text{prev}}), c(\mathbf{p}_{\text{prev}}, \mathbf{p}_{\text{cur}})) \tag{8.18}$$

which computes the maximum edge cost along the path. Any accumulation function that fulfils

$$\pi(\mathbf{p}_{\text{root}}, \mathbf{p}_{\text{cur}}) \geq \pi(\mathbf{p}_{\text{root}}, \mathbf{p}_{\text{prev}}). \tag{8.19}$$

is acceptable. If Eq. (8.19) is violated, a minimum cost path of finite length does not always exist.

Fig. 8.21 Sketch of path cost computation according to Eq. (8.17)

Using the concept for segmentation is straightforward. Each node is assigned a label and labels are propagated from the root of each tree in the forest to all children. After computing the IFT, labels are interpreted as segment membership.

Selecting appropriate node loads and path cost accumulation functions produces a variety of different segmentation methods. Additive path costs defined by Eq. (8.17) are useful for representing a seeded region growing by a minimum cost path search. In this case, a set of seeds s with labels l_s have to be predefined. Seed elements $\mathbf{s}$ receive a load $h(\mathbf{s})$ reflecting the homogeneity in a local neighborhood around $\mathbf{s}$. All other nodes $\mathbf{p}$ receive loads $h(\mathbf{p}) = \infty$, which excludes them from being roots of the IFT. Path costs between two nodes $(\mathbf{p}_1, \mathbf{p}_2)$ depend on local intensity differences or on the difference of $f(\mathbf{p}_2)$ to the function value at the root to which $(\mathbf{p}_1, \mathbf{p}_2)$ is added as path.

After carrying out the IFT, pixels have been labeled according to the root label of the minimum cost path leading to the corresponding node in the graph. The segmentation depends on the homogeneity along paths from the seed pixels to all pixels of a segment and on the length of the path.

The latter is often a desired behavior in interactive segmentation, as it allows the user to place seeds of different labels close to each other at subregions that belong—according to user knowledge—to different segments but where there is no visible difference in the appearance of the object. If the dependence on path length is unwanted, the accumulation function can be replaced by (8.18). In such case, the method resembles fuzzy connectedness computation, as the path cost depends on the most inhomogeneous part of the most homogeneous path between two pixels.

Equation (8.18) can also be used for computing the watershed transform. For a watershed transform with markers, the above-mentioned seeds play the role of markers. The cost function is the intensity at the seeds. The IFT will find a path to the marker with lowest intensity. For a WST without markers, loads for each pixel are set to infinity so that pixels become source of a water basin if they are local minima.

The IFT can be thought as a generalization of the concepts that underlie the live wire and the fuzzy connectedness approaches presented earlier. The authors have shown that many of the graph-oriented methods can be seen as special cases of the IFT. It is different from the graph cut strategy, as the assignment of seeds (whether explicitly or implicitly by being roots in the IFT forest) predetermines the topology of segments, whereas in graph cut segmentation pixels with the same label may belong to several different components. Whether graph cut or IFT is the appropriate strategy when representing a segmentation problem as path search in a graph depends on the domain knowledge about the desired kind of segmentation.

8.2.3 Random Walks

An effective and almost parameter-free variant to apply seeded segmentation on a graph are *random walks* (Grady 2006). Given seed pixels with labels $l_1, \ldots, l_K$, the algorithm computes for each unseeded pixel $\mathbf{p}$ the probability $P_k(\mathbf{p})$ that a random

Fig. 8.22 Random walk segmentation computes for each start pixel the probability of reaching each of the seeds given the probability w_{ij} of walking from pixel $\mathbf{p}_i$ to pixel $\mathbf{p}_j$

walk reaches the seed with label l_k first. Random walks define the desired segmentation similar to segmentation based on fuzzy connectedness or the image foresting transform (see Fig. 8.22). However, the search strategy goes from unlabeled pixels to the seeds instead of starting from the seeds. At first sight, this appears to be less efficient since the walk to the seed is governed by probabilities to walk along edges of the path between pixel and seed. Indeed, sampling random walks starting from every pixel would be unfeasible. Fortunately, the probability of reaching a seed from an unlabeled pixel can be directly computed without having to carry out the walk.

The walk is governed by probabilities to reach a pixel $\mathbf{p}_i$ from a neighboring pixel $\mathbf{p}_j$. This constitutes the data term. The probability is higher the more likely the two pixels belong to the same segment. A probability function suggested by Grady (2006) for the data term is

$$w_{ij} = \exp\left(-\beta\left(f(\mathbf{p}_i) - f(\mathbf{p}_j)\right)^2\right), \tag{8.20}$$

where f is the intensity of the pixel. The parameter β influences how quickly the probability decreases with increasing intensity differences. The function is similar to Boykov's term for interactive graph cuts (Eq. 8.2), except for the fact that the spatial distance between neighbors is not accounted for.

Segmentation by random walks first computes the probabilities for each pixel and each seed. Pixels are assigned a label which corresponds to the seed in which the random walk most likely ends. It is guaranteed that all pixels of a segment with label l_k are connected to the seed pixel with label l_k.

Computing the probabilities is associated with other equilibrium problems that can be calculated directly by solving a system of linear equations, as was shown by Grady (2006). The strategy to solve the problem is similar to that used in the computation of normalized cuts by expressing the optimization task on the graph as an optimization task on matrices. However, the optimized entity is completely

different. Opposed to the Normalized Graph Cut algorithm, the result is not an approximation, since its domain is the solution space.

In Grady (2006), it is shown that finding the probabilities for a walk starting at some pixel to arrive at some seed pixel is related to minimizing

$$D(\mathbf{x}) = \frac{1}{2}\sum_{e_{ij}\in E} w_{ij}\left(x_i - x_j\right) = \frac{1}{2}\mathbf{x}^T\mathbf{L}\mathbf{x}, \tag{8.21}$$

where $\mathbf{L}$ is a symmetric, positive semi-definite, sparse matrix with entries

$$L_{ij} = \begin{cases} d_i, & \text{if } i = j \\ -w_{ij}, & \text{if } v_\mathrm{i} \text{ and } v_j \text{ are adjacent nodes} \\ 0, & \text{otherwise.} \end{cases} \tag{8.22}$$

where d_i is the degree of the node v_i.

The nodes V are regrouped into a subset V_M of marked nodes and a subset V_U of unmarked nodes. The list of nodes is reordered so that the first nodes are the marked ones and the remaining ones are the unmarked nodes. The matrix $\mathbf{L}$ is reordered accordingly consisting now of two submatrices $\mathbf{L}_M$ and $\mathbf{L}_U$, which represent edge weights among marked and unmarked nodes, and a submatrix $\mathbf{B}$ containing weights between marked and unmarked nodes. The equation to minimize reads now (see Grady 2006 for details) is as follows:

$$\begin{aligned} D(\mathbf{x}) &= \frac{1}{2}\begin{pmatrix} \mathbf{x}_M^T & \mathbf{x}_U^T \end{pmatrix}\begin{pmatrix} \mathbf{L}_M & \mathbf{B} \\ \mathbf{B}^T & \mathbf{L}_U \end{pmatrix}\begin{pmatrix} \mathbf{x}_M \\ \mathbf{x}_U \end{pmatrix} \\ &= \frac{1}{2}\left(\mathbf{x}_M^T\mathbf{L}_M\mathbf{x}_M + 2\mathbf{x}_U^T\mathbf{B}^T\mathbf{x}_M + \mathbf{x}_U^T\mathbf{L}_U\mathbf{x}_U\right). \end{aligned} \tag{8.23}$$

Since this needs to be minimized with respect to the unknown $\mathbf{x}_U$, the equation is differentiated with respect to $\mathbf{x}_U$, yielding

$$\partial\left[\frac{1}{2}\left(\mathbf{x}_M^T\mathbf{L}_M\mathbf{x}_M + 2\mathbf{x}_U^T\mathbf{B}^T\mathbf{x}_M + \mathbf{x}_U^T\mathbf{L}_U\mathbf{x}_U\right)\right]/\partial\mathbf{x}_U = \frac{1}{2}\left(2\mathbf{B}^T\mathbf{x}_M + 2\mathbf{L}_U\mathbf{x}_U\right). \tag{8.24}$$

This is set to zero for finding the minimum of D, which results in a system of linear equations

$$\mathbf{B}^T\mathbf{x}_M + \mathbf{L}_U\mathbf{x} = 0 \;\Leftrightarrow\; \mathbf{L}_U\mathbf{x} = -\mathbf{B}^T\mathbf{x}_M. \tag{8.25}$$

For determining probabilities to reach some seed location $\mathbf{x}_s$, the potential for the marked nodes $\mathbf{x}_M$ is set to 1 for the location $\mathbf{x}_s$ and 0 for all other locations, resulting in a vector $\mathbf{m}^s$ with elements m_j^s

$$m_j^s = \begin{cases} 1, & \text{if node } j \text{ has the label } s \\ 0, & \text{otherwise} \end{cases} \tag{8.26}$$

The equation system to be solved for this label is then

$$\mathbf{L}_U \mathbf{x} = -\mathbf{B}^T \mathbf{m}^s. \tag{8.27}$$

Instead of solving for a single seed, the solution can be found for all seeds when a matrix $\mathbf{M}$ is created from all possible vectors $\mathbf{m}^s$ and a matrix $\mathbf{X}$ is created by duplicating $\mathbf{x}$ for every seed. Now, the equation system reads

$$\mathbf{L}_U \mathbf{X} = -\mathbf{B}^T \mathbf{M}. \tag{8.28}$$

The complete algorithm then consists of the following four steps

1. Create a graph from the image and compute edge weights based on the image intensities.
2. Generate a list of k seed nodes (interactively or by some automatic analysis).
3. Solve the equation system for the first $k-1$ seeds (since probabilities are computed, the probability $p(k)$ for the kth seed is $1 - \Sigma_{i=1\ldots k-1}\, p(i)$.
4. Assign a label to each pixel that belongs to the seed that is most likely reached from this pixel.

The code for this algorithm can be obtained from the web page of the author (www.cns.bu.edu/~lgrady/random_walker.matlab_code.zip).

Random walks have been applied to medical image segmentation for brain and cardiac MRI (Grady and Funka-Lea 2004) and have been shown to have a good performance on natural images (Duchenne et al. 2008). Similar to all other methods presented in this chapter, random walks are independent of the dimension of the data. A fast GPU implementation has been presented in Grady et al. (2005).

Automatic segmentation using a random walker requires seed locations to be automatically specified. An example for this is (Baudin et al. 2012) who present automatic seed point generation for muscle segmentation. Since there is little contrast between different muscles, substantial preprocessing is required. Seed points result from an annotated reference image that is rigidly registered to the data. Seed locations are sampled with low frequency and approximately mapped to the data. Seeds are not sampled close to the muscle boundaries. Hence, they approximate the interior of the different muscles so that the distance of seed locations to the boundary can be used as additional segmentation criterion. The idea is similar to atlas-based segmentation described in Chap. 11, but it is less sensitive to registration errors since just some locations inside the segments have to be found whereas atlas-based segmentation attempts to correctly register segment boundaries of the atlas with the image.

The random walker is an appropriate means here because, different to graph cuts, distance to seed points is intentionally part of the segmentation criterion. For the

optimized MRF, the unary term is label closeness and the pairwise term combines a direction and a distance criterion.

Another strategy for automatic seed localization uses trained appearance information to detect segments. It requires unique appearances of the different segments and has been applied, e.g., by Wighton et al. (2009) for skin lesion segmentation from color photographs. Gaussian mixture models are trained from color histograms for foreground and background regions.

Similar to graph cuts, random walkers have also been used to segment two registered images. An example is Bagci et al. (2012) for simultaneous segmentation on CT and PET. The corresponding optimization is derived from two observed features per voxel.

8.2.4 Other Methods Based on Seed Placements

The strategy of assigning segment membership based on initial seed labels dates back to the presentation of seeded region growing (Adams and Bischof 1994) and marker-based watershed transforms (Beucher and Meyer 1992). If weighted path length is added as segmentation criterion, it gets similar to the random walker. Such solution was presented by Kauffmann and Piché (2010) for fast multi-label segmentation based on a weighted graph. They compute k shortest paths from k seeds to each voxel and select the label which has the shortest path to all of the seeds. Weighting along the path penalizes inhomogeneities. Different to the random walker, however, this method computes the path costs directly on the graph instead of the computation as an eigenproblem.

The *GrowCut* algorithm of Vezhnevets and Konouchine (2005) use cellular automata to grow a segmentation from seeds. Initially, each pixel is assigned a label and a label strength. The latter reflects the confidence in the current label assignment. Then, labels and label strengths are iteratively updated. A sketch of the update mechanism of the algorithm is depicted in Fig. 8.23. A label changes if a similar pixel has a higher label strength than the current label. The similarity function is normalized and may be, for instance, the pixel similarity.

Fig. 8.23 Update step of the GrowCut algorithm. Label l_p at pixel p changes to label l_q if q is a neighbor of p and its label strength weighted by the similarity *sim*() between the pixels is greater than the label strength s_p

The method is fast and simple to implement and has thus been used as an alternative solution to the graph cut formulation. An example is the work of Hamanci et al. (2012) for brain tumor segmentation. They also showed that choosing an appropriate similarity function makes the method equivalent to the random walker. Grow cut may also be used to produce a rough estimate of the segmentation that is later refined such in Ryba et al. (2013) who segment the liver in CT images using level sets that are initialized by a grow cut segmentation.

The iterative fashion in which labels grow from the seed locations can be very slow, especially for large 3d data sets. In Zhu et al. (2014), it was observed that the labeling from grow cut can be achieved by turning it into a shortest path problem. The authors show that the labeling of any pixel is determined by the label of that seed that is connected to this pixel by the shortest path. The path length is given by the product of all neighbor similarities of pixels along the path weighted by the strength of the seed pixel. Applying a variant of Dijkstra's shortest path algorithm, the authors showed a speed gain of about 95% on a 3-d data set.

8.3 Similarities and Integrations of Graph-Based Strategies

Interestingly, graph cuts, level sets, image foresting transform, and random walkers have a lot in common. Several successful attempts have been made to show under what circumstances some or all of the methods are equivalent. In Sinop and Grady (2007), the authors showed that random walker and graph cut can be seen in a unified framework. This unification extends to a further algorithm that is based on similar concepts than fuzzy connectedness. The difference between random walker and graph cut is reduced to the metric used on the pairwise term (L_1 or L_2 norm). The presented extension uses L_∞-norm, which makes it similar to the image foresting transform. In the same year, (Allène et al. 2007) showed connections between watershed transform, spanning forests—which includes the image foresting transform although it was not mentioned—and graph cuts.

Power watersheds as the result of a unification of random walker, image foresting transform, graph cuts, and the watershed transform are presented by Couprie et al. (2009). Power watersheds allow adding a unary term to the watershed transform. Again, the difference is just the norm used in the unary and pairwise terms.

In Ciesielski et al. (2013), another unification has been presented, this time of the relative fuzzy connectedness (RFC) algorithm with graph cuts. As in Sinop et al. (2007), the authors find that the main difference lies in the metric used in the pairwise term (this time L_1 and L_∞). The advantage of fuzzy connectedness algorithms as opposed to graph cut is that it does not depend on seed placement and hence does also not cause too small segments around the seeds. While this can be achieved by normalized cuts as well, fuzzy connectedness is not as slow as the

NCut algorithm. For their own segmentation, the authors employ a linear-time graph cut for computing relative fuzzy connectedness (Ciesielski et al. 2012). According to a comment of the authors, the RFC seems to be equivalent to the power watershed algorithm of Couprie et al. (2009). The algorithm presented here combines RFC followed by graph cut. The RFC algorithm is fast and has the advantage that it is independent of the seed point location.

8.4 Concluding Remarks

Mapping an image on a graph is straightforward as are the definitions of global and local criteria for a good segmentation by node costs. The methods presented in this chapter show different ways to compute an optimal segmentation based on cost functions using (sometimes adapted) standard algorithms on graphs. The beauty of using graphs as representation lies in the independence from dimensionality and in the versatility to define different cost functions that represent different segmentation goals. The behavior of all segmentation methods presented here can be changed by parameterization. The basic algorithm always remains the same.

Selection of cost functions requires some insight into the underlying problem that is solved by the optimization procedure. A good example for this is the use of graph cuts to solve an MRF minimization problem. It shows common features between two segmentation strategies that at first sight appear to be very different. However, coming up with this solution will be difficult for a domain expert (e.g., a radiologist), unless he or she possesses also expert knowledge about the underlying principles of these two approaches. Hence, the potential versatility will be of more interest to the developer, who may generate a toolbox of different segmentation methods by reusing the representational power of a graph for characterizing the various attributes of different segmentation strategies. The paper of Kolmogorov and Zahbih (2004) is a helpful resource as far as graph cut segmentation is concerned, as it addresses energy functions and the graphs to construct for their minimization for binary problems such as foreground segmentation.

8.5 Exercises

- What kind of information is introduced by defining source and sink nodes in graph cut segmentation? How is it introduced?
- How is the influence of noise implicitly reduced in interactive graph cut segmentation?
- Explain the strategy that is used to avoid complete recomputation of the maximum flow when additional foreground or background nodes are specified in interactive graph cuts.

- How is an initial labeling created for graph-cut-based automatic segmentation? Why is it important that this labeling is a good approximation of the final result?
- What are the differences in using swap moves and expansion moves? Given a three-label segmentation, which of the two moves would be faster? Explain why.
- Why is it impossible to estimate the quality of a swap move optimization with respect to the unknown optimal solution?
- Explain how a circular shape prior could be added to the graph cut method.
- What would be the characteristics of a segmentation problem where regular graph cuts would be preferred over normalized graph cuts?
- What is meant by normalization in normalized graph cuts?
- At what step and how does it become apparent that the solution for finding an optimal normalized graph cut is approximate?
- Why is fuzzy connectedness an appropriate criterion to characterize a good segmentation? How is it used for this characterization?
- What is the difference between relative and absolute fuzzy connectedness? How is this reflected in segmentation methods based on fuzzy connectedness?
- Explain how the concept of fuzzy connectedness is related to the image foresting transform.
- What are similarities and what are the differences between image foresting transform and interactive graph cuts? Characterize a segmentation problem where the IFT would be preferred over interactive graph cuts.
- Explain how the foresting transform can be used to produce a watershed transform.
- How does a random walker differ from the image foresting transform?
- How is information about the smoothness of segment boundaries represented in a segmentation by a random walker?
- Graph cuts, image foresting transform, and random walker can be defined in a unified framework. What is the aspect in this framework, in which the methods differ?

References

Adams R, Bischof L (1994) Seeded region growing. IEEE Trans Pattern Anal Mach Intell 16 (6):641–647

Allène C, Audibert JY, Couprie M, Cousty J, Keriven R (2007) Some links between min-cuts, optimal spanning forests and watersheds. Math Morphol Appl Image Signal Process 253–264

Bagci U, Udupa JK, Yao J, Mollura DL (2012) Co-segmentation of functional and anatomical images. In: 15th international conference on medical image computing and computer assisted intervention MICCAI2012. Part III, LNCS, vol 7512, pp 459–467

Baudin PY, Azzabou N, Carlier P, Paragios N (2012) Automatic skeletal muscle segmentation through random walks and graph-based seed placement. In: International symposium medical imaging ISBI, pp 1036–1039

Beucher S, Meyer F (1992) The morphological approach to segmentation: the watershed transformation. Opt Eng 433–481

Boykov Y, Jolly MP (2001) Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In: 8th international conference on computer vision (ICCV'01), vol 1, pp 105–112

Boykov Y, Kolmogorov V (2004) An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans Pattern Anal Mach Intell 26(9):1124–1137

Boykov Y, Veksler O, Zabih R (2001) Fast approximate energy minimization via graph cuts. IEEE Trans Pattern Anal Mach Intell 23(11):1222–1239

Camilus LS, Govindam VK (2012) A review on graph based segmentation. Int J Image Graph Signal Process 5:1–13

Chen Y, Wang Z, Hu J, Zhao W, Wu Q (2012) The domain knowledge based graph-cut model for liver CT segmentation. Biomed Image Process Control 7:591–598

Ciesielski KC, Udupa JK, Falcao AX, Miranda PAV (2012) Fuzzy connectedness image segmentation in graph cut formulation: a linear-time algorithm and a comparative analysis. J Math Imaging Vis 44:375–398

Ciesielski KC, Miranda PAV, Falcao AX, Udupa JK (2013) Joint graph cut and relative fuzzy connectedness image segmentation algorithm. Med Image Anal 17:1046–1057

Couprie C, Grady L, Najman L, Talbot H (2009) Power watersheds: a new image segmentation framework extending graph cuts, random walker and optimal spanning forest. IEEE Int Conf Comput Vis ICCV 2009:731–738

Danesh H, Kafieh R, Rabbani H, Hajizadeh F (2014) Segmentation of choroidal boundary in enhanced depth imaging OCTs using a multiresolution texture based modeling in graph cuts. Comput Math Models Med 2014:8

Duchenne O, Audibert JY, Keriven R, Ponce J, Segonne F (2008) Segmentation by transduction. In: IEEE conference on computer vision and pattern recognition CVPR2008, pp 1–8

Falcao AX, Stolfi J, de Alencar Lotufo R (2004) The image foresting transform: theory, algorithms, and applications. IEEE Trans Pattern Anal Mach Intell 26(1):19–29

Funka-Lea G, Boykov Y, Florin C, Jolly MP, Moreau-Gobard R, Ramaraj R, Rinck D (2006) Automatic heart isolation for CT coronary visualization using graph-cuts. In: 3rd IEEE international symposium on biomedical imaging: nano to macro, pp 614–617

Gamio JC, Belongie S, Majumdar S (2004) Normalized cuts in 3d for spinal MRI segmentation. IEEE Trans Med Imaging 23(1):36–44

Grady L, Funka-Lea G (2004) Multi-label image segmentation for medical applications based on graph-theoretic electrical potentials. In: Computer vision and mathematical methods in medical and biomedical image analysis. LNCS, vol 3117, pp 230–245

Grady L (2006) Random walks for image segmentation. IEEE Trans Pattern Anal Mach Intell 28 (11):1768–1783

Grady L, Schiwietz T, Aharon S, Westermann R (2005) Random walks for interactive organ segmentation in two and three dimensions: implementation and validation. In: Medical image computing and computer-assisted intervention—MICCAI 2005. LNCS, vol 3750, pp 773–780

Géraud T, Mangin JF, Bloch I, Maître H (1995) Segmenting internal structures in 3d MR images of the brain by Markovian relaxation on a watershed based adjacency graph. In: IEEE international conference on image processing ICIP 1995, pp 548–551

Häggström M (2014) Medical gallery of Mikael Häggström 2014. Wikiversity J Med 1(2). doi:10.15347/wjm/2014.008

Hamamci A, Kucuk N, Karaman K, Engin K, Unal G (2012) Tumor-cut: segmentation of brain tumors on contrast enhanced MR images for radiosurgery applications. IEEE Trans Med Imaging 31(3):790–804

Han D, Bayouth J, Song Q, Taurani A, Sonka M, Buatti J, Wu X (2011) Globally optimal tumor segmentation in PET-CT images: a graph-based co-segmentation method. In: 22nd international conference information processing in medical imaging IPMI 2011. LNCS, vol 6801, pp 245–256

Herman GT, Carvalho BM (2001) Multiseeded segmentation using fuzzy connectedness. IEEE Trans Pattern Anal Mach Intell 23(5):460–474

Huang SH, Chu YH, Lai SH, Novak CL (2009) Learning-based vertebra detection and iterative normalized-cut segmentation for spinal MRI. IEEE Trans Med Imaging 28(10):1595–1605

Kauffmann C, Piché N (2010) Seeded ND medical image segmentation by cellular automaton on GPU. Int J Comput Assist Radiol Surg 5:251–262

Kéchichian R, Valette S, Desvignes M, Prost R (2013) Shortest-path constraints for 3d multiobject semiautomatic segmentation via clustering and graph cut. IEEE Trans Image Process 22 (11):4224–4236

Kolmogorov V, Zabih R (2004) What energy functions can be minimized via graph cuts? IEEE Trans Pattern Anal Mach Intell 26(2):147–159

Lucchi A, Smith K, Achanta R, Knott G, Fua P (2012) Supervoxel-based segmentation of mitochondria in EM image stacks with learned shape features. IEEE Trans Med Imaging 31 (2):474–486

Mahapatra D, Schüffler PJ, Tielbeek JAW, Vos FM, Buhmann JM (2013) Crohn‘s disease tissue segmentation from abdominal MRI using semantic information and graph cuts. In: IEEE 10th international symposium biomedical imaging: from nano to macro, pp 358–361

Ng A, Jordan M, Weiss Y (2002) On spectral clustering: analysis and an algorithm. In: Proceedings 14th annual conference advances in neural information processing systems, pp 849–856

Peng B, Zhang L, Zhang D (2013) A survey of graph theoretical approaches to image segmentation. Pattern Recognit 46:1020–1038

Radhakrishna A, Shaji A, Smith K, Lucchi A, Fua P (2010) Susstrunk S (2010). SLIC superpixels, EPFL Techn Rep

Rak M, König T, Toennies KD (2013) An adaptive subdivision scheme for quadratic programming in multi-label image segmentation. In: British machine vision conference BMVC 2013

Rother C, Kolmogorov V, Blake A (2004) Grabcut: interactive foreground extraction using iterated graph cuts. ACM Trans Graph (TOG) 23(3):309–314

Ryba T, Jirik M, Zelezny M (2013) An automatic liver segmentation algorithm based on grow cut and level sets. Pattern Recognit Image Anal 23(4):502–507

Salah MB, Mitiche A, Ayed IB (2011) Multiregion image segmentation by parametric kernel graph cuts. IEEE Trans Image Process 20(2):545–557

Shi J, Malik J (2000) Normalized cuts and image segmentation. IEEE Trans Pattern Anal Mach Intell 22(8):888–905

Sinop AK, Leo Grady (2007) A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In: IEEE 11th international conference on computer vision. ICCV 2007, pp 1–8

Steffen J, Rak M, König T, Toennies KD (2016) Information efficient automatic object detection and segmentation using cosegmentation, similarity based clustering and graph label transfer. In: 5th international conference pattern recognition, applications and methods, ICPRAM, pp 397–406

Tolliver DA, Miller GL (2006) Graph partitioning by spectral rounding: applications in image segmentation and clustering. In: IEEE computer society conference on computer vision and pattern recognition, vol 1, pp 1053–1060

Udupa JK, Samarasekera S (1996) Fuzzy connectedness and object definition: theory, algorithms, and applications in image segmentation. Graph Models Image Process 58(3):246–261

Udupa JK, Saha PK (2003) Fuzzy connectedness and image segmentation. Proc IEEE 91 (10):1649–1669

van der Lijn F, den Heijer T, Breteler MMB, Niessen WJ (2008) Hippocampus segmentation in MR images using atlas registration, voxel classification and graph cuts. Neuroimage 43:708–720

Veksler O (2008) Star shape prior for graph-cut image segmentation. In: Computer vision—ECCV 2008. LNCS, vol 5304, pp 454–467

Vezhnevets V, Konouchine V (2005) GrowCut: interactive multi-label ND image segmentation by cellular automata. Proc Graphicon 1:150–156

Vicente S, Kolmogorov V, Rother C (2008) Graph cut based image segmentation with connectivity priors. In: IEEE conference on computer vision and pattern recognition, CVPR 2008, pp 1–8

Vicente S, Rother C, Kolmogorov V (2011) Object cosegmentation. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 2217–2224

Wighton P, Sadeghi M, Lee TK, Atkins S (2009) A fully automatic random walker segmentation for skin lesions in a supervised setting. In: 12th international conference medical image computing and computer assisted intervention MICCAI 2009. Part II, LNCS, vol 5762, pp 1108–1115

Wu Z, Leahy R (1993) An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. IEEE Trans Pattern Recognit Mach Anal 15(11):1101–1113

Zheng Y, Stambolian D, O'Brien J, Gee JC (2013). Optic disk and cup segmentation from color fundus photograph using graph cut with priors. In: 16th international conference on medical image computing and computer assisted intervention MICCAI2013. LNCS, vol 8150, pp 75–82

Zhu L, Kolesov I, Gao Y, Kikinis R, Tannenbaum A (2014) An effective interactive medical image segmentation method using fast growcut. In: MICCAI workshop on interactive medical image computing

Active Contours and Active Surfaces

9

Abstract

Active contours and active surfaces are means of model-driven segmentation. Their use enforces closed and smooth boundaries for each segmentation irrespective of the image content. They are particularly useful if such properties cannot be derived everywhere from the data. We will discuss explicit and implicit active contours, their definition, parameterization, and properties. Different fitting methods for active contours will be presented in detail, since their understanding is necessary to understand parameterization and stability issues. We present methods to extend implicit active contours to multi-label segmentation and we discuss how to deal with intensity variations and inhomogeneous background intensity.

Concepts, Notions, and Definitions Introduced in this Chapter

- Explicit active contours and surfaces;
- Snakes;
- Intrinsic and extrinsic attributes of active contours;
- Optimization of explicit active contours;
- Constraints for the evolution of explicit active contours;
- Implicit active contours in the level set framework;
- Stationary and dynamic level sets;
- Level set evolution by front propagation;
- Geodesic active contours, variational level sets, multi-phase level sets.

The Bayesian formulation of the segmentation problem and its solution as a graph problem are examples for a data-driven segmentation. The underlying model for most of these methods is that objects in an image appear homogeneous. The use

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_9

Fig. 9.1 Finding the segment boundary between the two points will be very difficult if additional information about the smoothness and closedness of the course of the boundary cannot be included in the procedure

of active contours and active surfaces, i.e., active boundaries, is *model-driven*. The model predicts properties of an ideal segment, which are imposed on the data. In the case of active contours or surfaces, attributes of the outline of an ideal object boundary are predicted. The boundary is assumed to be smooth and closed. This kind of domain knowledge allows a successful segmentation of objects even if they are only partially contrasted against the background (see Fig. 9.1). Since the model does not predict the shape or appearance of an object, it may be applied to a wide variety of different objects (and not just, e.g., to livers in ultrasound images).

Using such model of an ideal segment is best suited for foreground segmentation. It requires representation of a deformable ideal segment that balances model attributes against data information. The importance of the model with respect to the data is usually parameterizable so that it can be set according to assumptions about the reliability of data and model. Since model-driven segmentation is a fitting process, the quality of a given estimate for the true segment boundary needs to be computable. Furthermore, a process is required to determine the best fit.

The model can be represented either explicitly or implicitly. An *explicit representation* describes the boundary as parameterizable curve or surface in space. Locations in space that are not on the boundary are not defined by this representation. An *implicit representation* describes the boundary as function of the space in which it is embedded. Hence, locations that are not part of the boundary are defined as well as inner or outer points.

An explicit representation is often more intuitive than an implicit representation because the searched boundary is explicitly described. It has the disadvantage that the behavior of an evolving instance of such representation, when being fitted to the data, is difficult to control. A location to which a boundary may move is defined in terms of the representation only when the boundary is placed on this location. This is different for an implicit representation. Each location in space is part of the representation at any time. Evolving boundaries may change their topology if necessary.

In the next section, explicit representations of active contours will be discussed. They are based on the snake model presented in Kass et al. (1988). This is followed by a discussion on implicit active contours presented in Osher and Sethian (1998).

9.1 Explicit Active Contours and Surfaces

Active contours were first presented by Kass et al. (1988) who called their model a *snake*. The snake is a curve that is fitted to the data while retaining certain smoothness properties. The goal is to find a contour that separates (part of) an object from the background. The course of the snake smoothly follows high-intensity gradients if the gradients reliably reflect the object boundary. Otherwise, a smooth boundary is generated bridging regions of noisy data or missing gradients. Such active contour is particularly well suited to segment an object instance in an image where the data is distorted by noise or artifacts.

The extent of regions with missing or distorted gradient information should be relatively small because otherwise the smooth boundary completion by the active contour may not reflect the true course of the object boundary.

9.1.1 Deriving the Model

An active contour is a deformable curve $\mathbf{r}(s)$, $0 \leq s \leq 1$ that is placed into a 2D image (see Fig. 9.2). The vector $\mathbf{r}(s)$ contains the (x, y) coordinates of points s on the curve. The model enforces smoothness by minimizing the first and second derivative along the curve as follows:

$$E_{\text{internal}}(\mathbf{r}(s)) = w_1 \left(\frac{\partial \mathbf{r}(s)}{\partial s} \right)^2 + w_2 \left(\frac{\partial^2 \mathbf{r}(s)}{\partial s^2} \right)^2 = \min, \quad w_1, w_2 > 0 \tag{9.1}$$

Fig. 9.2 A snake is a deformable curve $\mathbf{r}(s)$ with points $0 \leq s \leq 1$ that is placed into an image

The term E_{internal} models the *internal energy* of a linear elastic band. It behaves similar to a rubber band, except in extreme situations (a rubber band close to rupture no longer stretches linearly with the exerted force).

The first derivative in E_{internal} is the *elasticity* and the second derivative is the *stiffness* of the rubber band. The weights w_1 and w_2 rate stiffness against elasticity and balance this term against an external energy E_{external} from the image. *External energy* is often related to the image gradient for causing the active contour to follow high gradients in the image. An example is

$$E_{\text{external}}(\mathbf{r}(s)) = -\|\nabla f(\mathbf{r}(s))\|^2 = \min, \tag{9.2}$$

where $f(\mathbf{r}(s))$ is the image intensity at position $\mathbf{r}(s)$. Finding the optimal curve now requires minimization of the two energies along s:

$$\int_0^1 E_{\text{internal}}(\mathbf{r}(s)) + E_{\text{external}}(\mathbf{r}(s))\mathrm{d}s = \min. \tag{9.3}$$

Using variational calculus (see Sect. 14.2) to find a function $\mathbf{r}(s)$ that minimizes the functional in Eq. (9.3) leads to the following Euler–Lagrange equation

$$-w_1 \frac{1}{\partial s}\left(\frac{\partial \mathbf{r}(s)}{\partial s}\right) + w_2 \frac{1}{\partial s^2}\left(\frac{\partial^2 \mathbf{r}(s)}{\partial s^2}\right) + \nabla E_{\text{external}}(\mathbf{r}(s)) = 0. \tag{9.4}$$

The derivative of the external force, e.g., the gradient length as in Eq. (9.2), will be zero at the maxima of E_{external}. Hence, it is an influence vector field that points to the closest feature in the data. Minimizing (9.4) simulates placing an elastic band in this vector field. The band then moves to the closest features while trying to achieve an energy-minimal state.

In the absence of an external energy, this state is either a straight line or a point to which the curve shrinks. It is a straight line when the boundary conditions of Eq. (9.4) are given by two separate end points of the curve. It shrinks to a point when the boundary conditions are given by a closed curve, i.e., a periodic function $\mathbf{r}(s)$.

In practice, the curve $\mathbf{r}(s)$ is defined only at a finite number of locations $0 \leq s_1 < s_2 \cdots < s_n \leq 1$ (see Fig. 9.3). Derivatives through the curve are approximated by finite differences:

$$\frac{\partial \mathbf{r}(s_i)}{\partial s} \approx \frac{\mathbf{r}(s_i) - \mathbf{r}(s_{i-1})}{h}, \tag{9.5}$$

$$\frac{\partial^2 \mathbf{r}(s_i)}{\partial s^2} \approx \frac{\mathbf{r}(s_{i+1}) - 2\mathbf{r}(s_i) + \mathbf{r}(s_{i-1})}{h^2}, \; h = \|\mathbf{r}(s_i) - \mathbf{r}(s_{i-1})\|. \tag{9.6}$$

Differentiating this according to (9.4), we arrive at the following expressions for the terms of the Euler–Lagrange equation:

Fig. 9.3 The smooth snake curve defined everywhere on the curve is approximated by a finite sequence of curve points s_i with derivatives estimated from finite differences

$$\begin{aligned} -\left(w_1 \frac{\partial \mathbf{r}(s_i)}{\partial s}\right)' &\approx \frac{w_1}{h}\left((\mathbf{r}(s_i) - \mathbf{r}(s_{i-1})) - (\mathbf{r}(s_{i+1}) - \mathbf{r}(s_i))\right) \\ &= \frac{w_1}{h}\left(-\mathbf{r}(s_{i-1}) + 2\mathbf{r}(s_i) - \mathbf{r}(s_{i+1})\right), \end{aligned} \tag{9.7}$$

$$\left(\frac{\partial^2 \mathbf{r}(s_i)}{\partial s^2}\right)'' \approx \frac{w_2}{h^2}\begin{bmatrix} (\mathbf{r}(s_{i-2}) - 2\mathbf{r}(s_{i-1}) + \mathbf{r}(s_i)) \\ -2(\mathbf{r}(s_{i-1}) - 2\mathbf{r}(s_i) + \mathbf{r}(s_{i+1})) \\ +(\mathbf{r}(s_i) - 2\mathbf{r}(s_{i+1}) + \mathbf{r}(s_{i+2})) \end{bmatrix}. \tag{9.8}$$

The derivative of the external energy $\nabla E = [\,\partial E/\partial x \quad \partial E/\partial y\,]^{\mathrm{T}}$ is approximated by differences as well.

The results are two linear equation systems, one for each coordinate:

$$\mathbf{A}\mathbf{x} + \frac{\partial E(\mathbf{s})}{\partial x} = \mathbf{0}, \mathbf{x} = (\,x_1 \quad \ldots \quad x_N\,), \frac{\partial E(\mathbf{s})}{\partial x} = \begin{pmatrix} \frac{\partial E(s_1)}{\partial x} & \ldots & \frac{\partial E(s_N)}{\partial x} \end{pmatrix} \tag{9.9}$$

$$\mathbf{A}\mathbf{y} + \frac{\partial E(\mathbf{s})}{\partial x} = \mathbf{0}, \mathbf{y} = (\,y_1 \quad \ldots \quad y_N\,), \frac{\partial E(\mathbf{s})}{\partial y} = \begin{pmatrix} \frac{\partial E(s_1)}{\partial y} & \ldots & \frac{\partial E(s_N)}{\partial y} \end{pmatrix} \tag{9.10}$$

where $\mathbf{0}$ is the zero vector. The matrix $\mathbf{A}$ is a band matrix. It contains the weights from Eqs. (9.9) and (9.10):

$$\mathbf{A} = \begin{pmatrix} -2w_1 + 6w_2 & w_1 - 4w_2 & w_2 & \ldots & 0 \\ w_1 - 4w_2 & -2w_1 + 6w_2 & w_1 - 4w_2 & & \ldots \\ w_2 & w_1 - 4w_2 & -2w_1 + 6w_2 & & w_2 \\ \ldots & \ldots & \ldots & \ldots & w_1 - 4w_2 \\ 0 & 0 & 0 & \ldots & -2w_1 + 6w_2 \end{pmatrix} \tag{9.11}$$

Equations (9.9) and (9.10) can be solved analytically by inverting $\mathbf{A}$. It can be done in $O(N)$ time, since $\mathbf{A}$ is penta-diagonal. If weights w_1 and w_2 are kept constant, the matrix does not change and the inversion needs not to be repeated.

Given $\mathbf{A}$, an iterative scheme for the displacement of the active contour over time can be developed. If node positions $\mathbf{r}(s)$ vary with time, their location at time t is $\mathbf{r}^{(t)}(s)$ with $\mathbf{r}^{(t)}(s) = \begin{pmatrix} \mathbf{x}^{(t)} & \mathbf{y}^{(t)} \end{pmatrix}$. The active contour will stop moving if a local minimum of the energy function has been found. In the local minimum, $\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)} = \mathbf{0}$ and $\mathbf{y}^{(t+1)} - \mathbf{y}^{(t)} = \mathbf{0}$ must hold. This can be plugged into Eqs. (9.9) and (9.10), yielding:

$$\mathbf{A}\mathbf{x}^{(t)} + \frac{\partial E\left(\mathbf{s}^{(t)}\right)}{\partial x} = \gamma\left(\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}\right) \tag{9.12}$$

and

$$\mathbf{A}\mathbf{y}^{(t)} + \frac{\partial E\left(\mathbf{s}^{(t)}\right)}{\partial y} = \gamma\left(\mathbf{y}^{(t+1)} - \mathbf{y}^{(t)}\right) \tag{9.13}$$

Resolving this for the node locations at time $t + 1$ results in the following iteration step

$$\mathbf{x}^{(t+1)} = (\mathbf{A} - \gamma\mathbf{I})^{-1}\left(\mathbf{x}^{(t)} - \frac{\partial E\left(\mathbf{s}^{(t)}\right)}{\partial x}\right) \tag{9.14}$$

and

$$\mathbf{y}^{(t+1)} = (\mathbf{A} - \gamma\mathbf{I})^{-1}\left(\mathbf{y}^{(t)} - \frac{\partial E\left(\mathbf{s}^{(t)}\right)}{\partial y}\right), \tag{9.15}$$

where $\mathbf{I}$ is the identity matrix and γ is a parameter that controls the step size. The matrix $(\mathbf{A} + \gamma\mathbf{I})$ is still penta-diagonal and does not change for given values of w_1, w_2, and γ. Hence, it can be inverted with an $O(N)$ computational cost and does not need to be inverted again as long as the parameters do not change.

Unfortunately, the energy function has many local minima. It is not guaranteed that the iterative procedure fits the active contour to the desired object boundary. Hence, most active contour methods require the user to place the model contour sufficiently close to the object to be segmented (see Fig. 9.4).

9.1.2 The Use of Additional Constraints

Several variants of the active contour for boundary representation in 2d and 3d exist. The *balloon model* presented as an active contour in 2d (Cohen 1991) and in

Fig. 9.4 The energy function has many local minima. Hence, the snake will find the contour only if placed close enough to the boundary

Fig. 9.5 A balloon force is an additional force for a closed active contour that drives the contour outward or inward. In this example, an inward driving force was chosen that lets the contour move toward the boundary when placed around the structure of interest

3d (Cohen and Cohen 1993) adds an additional inflating force that causes a closed active contour or surface to move away from its initial position (see Fig. 9.5). The inflation force is included as additional external force

$$\mathbf{f}_{\text{external}}(\mathbf{r}(s)) = k_1\mathbf{n}(s) - k_2 \frac{\nabla E_{\text{external}}(\mathbf{r}(s))}{\|\nabla E_{\text{external}}(\mathbf{r}(s))\|}. \tag{9.16}$$

The vector $\mathbf{n}(s)$ is normal to the curve $\mathbf{r}(s)$ at a point s. The two external influences are weighted with respect to each other by k_1 and k_2 using application-specific weights. The inflation force causes the curve to move beyond spurious gradients caused by noise. Balloon forces have been used in many applications in medical image analysis (McInerney and Terzopoulos 1995a; Honea et al. 1999; Ladak et al. 2000), since their behavior is intuitive and often effective.

Another strategy to influence the behavior of an active contour is the introduction of *gradient vector flow*, as presented by Kichenassamy et al. (1995), Xu and Prince (1998). An introduction and examples can be found at the authors' Webpage at http://www.iacl.ece.jhu.edu/static/gvf/. The method is available as MATLAB code.

The authors observed that displacement, as defined by the gradient of external forces, vanishes if the active contour is too far away from image features (such as high gradients). They defined a static, smooth attraction field that is defined everywhere in the image. This field is called gradient vector flow (GVF), and it replaces the gradient of external energy $\nabla E_{\text{external}}$ in Eq. (9.4).

Fig. 9.6 Gradient vector flow (Xu and Prince 1998) drives a contour from everywhere in the image to the segment boundary

A gradient vector field $\mathbf{v}(\mathbf{x}) = \mathbf{v}(x_1, x_2) = [\mathrm{u}(x_1, x_2)\ \mathrm{v}(x_1, x_2)]$ in a 2D image with locations $\mathbf{x} = (x_1, x_2)$ is computed by minimizing (see Fig. 9.6 for an example)

$$E = \iint \lambda \left[\left(\frac{\partial u}{\partial x_1}\right)^2 + \left(\frac{\partial u}{\partial x_2}\right)^2 + \left(\frac{\partial v}{\partial x_1}\right)^2 + \left(\frac{\partial v}{\partial x_2}\right)^2 \right] + \|\nabla f\|^2 \|\mathbf{v} - \nabla f\|^2 \mathrm{d}x_1 \mathrm{d}x_2, \tag{9.17}$$

where f is the image function and $\nabla f(\mathbf{x})$ the desired image feature. A GVF function that minimizes ε produces values $\mathbf{v}(\mathbf{x})$ that are close to $\nabla f(\mathbf{x})$ if $\nabla f(\mathbf{x})$ is large. Otherwise, the first term of (9.17) enforces a smooth variation of $\mathbf{v}$ in space. The weight λ is application-specific. If the data contains noise, high values for λ will result in a smoothly varying gradient vector field that is not influenced by spurious gradients in the data.

The GVF needs to be computed only once. It is done by discretizing (9.17) and then arriving at an iterative procedure for estimating $\mathbf{v}$. At each location $\mathbf{x}$, the current estimate for $\mathbf{v}$ is adapted as to minimize the difference of $\mathbf{v}(\mathbf{x})$ to $\mathbf{v}(\mathbf{x}_{nb})$ in neighboring locations $\mathbf{x}_{nb}$. Simultaneously, the difference between the current estimates of $\mathbf{v}(\mathbf{x})$ to the image gradient $\nabla f(\mathbf{x})$ is minimized.[1]

The use of a GVF as opposed to inflation forces finds the true object boundary by image-driven means. It may still require initializing the active contour close to the object instance in the image because data information may be insufficient for guiding the contour evolution.

The two strategies to guide an active contour are probably the most often cited examples for including additional information in the process. Other methods are to restrict the search space by specifying end points of the curve (Cohen and Kimmel 1997), by adding a preceding edge detection step (Park and Keller 2001), or by

[1]The procedure is similar to optical flow computation by simultaneous minimization of the error for the Horn–Schunck constraint and the difference between neighboring displacement vectors.

Fig. 9.7 A T-snake moves as an explicit representation, but it includes an update step where an additional implicit representation is used to decide whether the explicit topology needs to be changed

introducing continuity constraints from already found structures (Davatzikos and Prince 1995; Neuenschwander et al. 1997).

9.1.3 T-Snakes and T-Surfaces

A constraint that sometimes inhibits a good segmentation is the preservation of topology. Although it is most welcome when the topology is known, it makes segmentation by active contours cumbersome when the segment consists of an unknown number of regions with smooth boundaries. In such case, topology-adaptive snakes (*T-snakes*) presented by McInerney and Terzopoulos (1995b, 2000) can be applied.

T-snakes receive their ability of topological adaptation by a combination of an explicit active curve representation with an implicit representation of the curve. Hence, they behave somewhat similar to implicit active contours to be described in the next section.

A T-snake is a closed active contour together with an implicit representation, by which space is separated into interior and exterior scene elements. The T-snake develops in a sequence of deformation steps. Within a deformation step, the contour behaves like a regular snake but additionally updates the implicit representation (see Fig. 9.7). The update step between two deformation steps uses the implicit representation for splitting the enclosed regions into two or more regions or for merging regions.

The snake evolution is governed by an inflation or deflation force. During a deformation step, the contour either shrinks or grows but may alternate between growing and shrinking from step to step.

The T-snake differs from a regular snake in the existence of the implicit representation and reparameterization after each deformation step. These two aspects are now described in more detail.

The implicit representation for a T-snake is a list of element features of a cell enumeration representation. A cell enumeration representation decomposes a finite space into a finite number of elements. Every point in space belongs to exactly one of the elements. Pixel and voxel representations are examples for 2D and 3D cell enumeration representations.

Fig. 9.8 For ease of computation the gridding is a simple grid. Grid elements are either interior, exterior, or boundary elements depending on the location of the simplex node inside or outside of the object

Although the T-snakes can be defined on a pixel or a voxel representation, it is easier to work with simplexes.[2] In 2D, this is a representation by triangles, in 3D it is a representation by tetrahedrons (the representation is then called a *T-surface*).

In order to make things not overly complicated, simplex cells are usually similar to each other and arranged in a simple grid (such as the one shown in Fig. 9.8). A simplex is fully inside or outside of an active contour if all its boundary nodes are interior or exterior nodes. The active contour is passing through the simplex if some of the simplex nodes are interior and some of them are exterior. Hence, the implicit representation consists of a list of simplexes s with labels f_s

$$f_s(s) = \begin{cases} -1 & \text{if } s \in \text{``exterior''} \\ 0 & \text{if } s \in \text{``boundary''} \\ 1 & \text{if } s \in \text{``interior''} \end{cases} \tag{9.18}$$

A similar function exists for all nodes n of the simplex:

$$f_n(n) = \begin{cases} -1 & \text{if } n \in \text{``exterior''} \\ 1 & \text{if } n \in \text{``interior''} \end{cases} \tag{9.19}$$

An edge connecting two nodes is interior or exterior if the two nodes are either interior or exterior. Otherwise, the T-snake must cross the edge: The edge is then a boundary edge.

The T-snake is initially defined by a vertex sequence with all vertices lying on boundary edges. The evolution of the T-snake will cause snake vertices no longer to lie on boundary nodes. Hence, intersection points of the evolved T-snake with grid edges are computed during reparameterization and denoted potential snake vertices. Given the T-snake, nodes of the cell representation are tested as to whether their label (interior or exterior) should change.

The update may be ambiguous if a T-snake intersects itself or is intersected by other T-snakes. In such case, a cell node may be exterior with respect to one part of

[2]The advantage over using a pixel or voxel grid is that unique surface representations for a surface passing the cell exist. This is important for an active contour because it is a surface in 2D or 3D. Using a pixel or voxel grid requires an additional mechanism to resolve ambiguities, which are known, for instance, from the marching cube algorithm on a voxel grid.

the T-snake and interior with respect to another part. The ambiguity is resolved by assuming that a T-snake either shrinks or grows. If it grows, then a node once being interior can never be declared exterior. The opposite is true for a shrinking T-snake.

After updating cell nodes, labels of cells and cell edges are updated accordingly.

In the next step, it is tested whether all vertices of the updated T-snake are still on boundary edges. If a part of the T-snake intersects interior edges because the T-snake intersected itself or touched other T-snakes, the respective vertices are removed and the T-snakes are merged. A T-snake vertex may also lie on an exterior edge. In this case, the vertices are removed resulting in two separate T-snakes.

If T-snakes are extended to T-surfaces on a 3D grid, the algorithm does not change, except for the fact that the grid now consists of nodes, edges, faces, and cells and the computation time increases with the amount of data.

9.2 The Level Set Model

T-snakes already show the advantage of an implicit boundary representation. However, T-snakes use different representations for different tasks. The implicit representation of a T-snake consists of an enumeration of function values in discrete space. Derivatives, curvatures, and other smoothness properties cannot be represented in this space. It requires changing between a discrete, implicit representation and a continuous, explicit representation during the evolution of the active contour.

The *level set model* integrates smoothness properties, correspondence to the data and evolution of a representation of interior and exterior points in a common, continuous implicit representation (Sethian 1998; Osher and Paragios 2003). The level set model is an implicit function of image space $\mathbf{x} = (x_1\ x_2\ \ldots\ x_N)$. Every location in space can be uniquely assigned a label "interior," "exterior," or "boundary." Self-intersections, which may happen in an explicit representation, are impossible. Topological changes during the adaptation of a level set representation to the data may happen. The goal of a level set representation in segmentation is to define a continuous, smooth representation of a segment boundary and let it evolve so that it adapts to boundary locations in the image in some optimal fashion. Just what is optimal is defined similarly to the criteria used for explicit active contours and surfaces.

The level set function can be thought of as a wave (see Fig. 9.9). Wave propagation guides a specific wave front to the segment boundary. In order to represent this propagation in time, the level set model embeds an implicit N-dimensional active contour in $(N + 1)$-dimensional space. The first N dimensions represent locations $\mathbf{x} = (x_1\ x_2\ \ldots\ x_N)$ in the image. The additional dimension is the time t.

Fig. 9.9 The level set model can be thought of a function representing waves moving over time. A specific wave front represents the current estimate of the segment boundary

An initial level set function $\phi(\mathbf{x}, t_0)$ at time t_0 is given by the user. Its level $\phi(\mathbf{x}, t_0) = k(t_0)$ describes the initial guess of the segment boundary. It usually does not coincide with the true object boundary. The level set function is then evolved toward the object while maintaining the smoothness of the wave front.

9.2.1 Level Sets

Wave propagation in the level set framework does not rely on the specific shape of a wave. Hence, a wave function ϕ is selected so that the position of a wave front is given by all points having a specific value (see Fig. 9.10). A k-level set of a

Fig. 9.10 The level set function ϕ is chosen in such a way that $\phi(\mathbf{x}, t) = k$ indicates the current estimate of the segment boundary

Fig. 9.11 Level sets are implicitly defined on *f*. Hence, topology of the explicit boundary may change (in the example above, this happens for $\phi(\mathbf{x}, t) = 2$)

function $\phi(\mathbf{x})$ consists of all points $\mathbf{x}$ for which $\phi(\mathbf{x}) = k$. For differentiable functions ϕ, which are defined on a real number domain, the level set consists of closed boundaries separating $\phi(\mathbf{x}) > k$ from $\phi(\mathbf{x}) < k$.

A level set is said to be the *interface* between two sets of points. If used to represent a segmentation, discretized values of $\mathbf{x}$ denote coordinates of scene elements (e.g., pixels, voxels) and $\phi(\mathbf{x}) = k$ is the segment boundary. Since the level k is arbitrary, it is usually the 0-level set that is evolved.

The level set method evolves the interface based on its intrinsic properties, extrinsic image properties, and domain knowledge. An intrinsic property could, for instance, enforce a smooth segment boundary. Extrinsic properties, e.g., related to the intensity gradient, cause the 0-level set to terminate on segment boundaries. An example for domain knowledge is the placement of the initial level set.

Level sets are in many respects similar to the snakes introduced in the previous section. Hence, the terms active contour and active surface apply to level sets as well. A number of differences should be noted, however:

- Snakes represent curves explicitly. Changing the topology during snake evolution is impossible unless the representation is transferred to an implicit representation (such as in T-snakes). The level set definition is an implicit representation by some function $\phi(\mathbf{x})$ without having this restriction (see Fig. 9.11).
- Snake-like representations can be defined for arbitrary dimensions only if the explicit representation is adapted accordingly (and possibly also the optimization scheme). The level set formulation is independent of the dimension.
- Snakes are discretized in terms of the explicit representation which may require reparameterization if adjacent nodes become too far apart in image space. This is not necessary for level sets, as they are defined—and will be evolved—in continuous space.

These are good reasons to consider an implicit active contour for segmentation. It does not mean, however, that explicit active contours should be disregarded in any case. If, for instance, the topology of a segmentation is known, this is already represented in an explicit formulation.

The great advantage of the level set representation is its generality. It enables solutions for many different applications under the same concept.

Before applying level sets to segmentation, several questions need to be answered:

- What is an appropriate function ϕ for representing the interface?
- How can interface motion be represented in a parameterizable fashion so that different evolving interfaces can be generated by changing parameters?
- Which parameters are useful to let the interface evolve toward the segment boundaries?

The level set function and the evolution methods will be discussed in the next section. This is then followed by a treatment of techniques to speed up the computation. Finally, we will explain strategies to parameterize the level set method.

9.2.2 Level Sets and Wave Propagation

The evolution of a level set is the propagation of a wave ϕ. The current estimate of segment boundaries at some time t consists of all locations $\mathbf{x}$ for which $\phi(\mathbf{x}, t) = 0$. The wave ϕ can be defined as the level set function:

$$\phi(\mathbf{x}, t) = \begin{cases} \mathrm{d}(\mathbf{x}, C(t)), & \text{if } \mathbf{x} \text{ is inside the segment} \\ -\mathrm{d}(\mathbf{x}, C(t)), & \text{otherwise} \end{cases} \tag{9.20}$$

The function $C(t)$ is the set of points in the 0-level set at time t, and $\mathrm{d}(\mathbf{x}, C(t))$ is the shortest distance of $\mathbf{x}$ to this curve. In other words, the level set function is a signed distance function (see Fig. 9.12).

The wave is propagated orthogonal to the wave front. Domain knowledge about expected segment properties and a data term influence the propagation speed at each point. Ideally, the interface stops (at least it slows down) at the segment boundary. Interface motion can be of two kinds (see Fig. 9.13):

- A *stationary level set* is a function of arrival times for a level set for each location $\mathbf{x}$. Being a function, a point $\mathbf{x}$ in image space never gets reached twice. Hence, the interface is moving to either the inside or the outside starting from the initial level set. A segmentation is then a set of points which are reached at the same time.
- A *dynamic level set* defines the level set as function of $\mathbf{x}$ and time t. The interface may change direction during its evolution and may cross a point

Fig. 9.12 Given some initial position for the level set, the level set function can be generated as distance transform for the distance from this initial boundary

Fig. 9.13 Level sets propagate orthogonal to the wave front. Stationary level sets move either outward or inward. Dynamic level sets may change the propagation direction

x several times. A segmentation is the location of the interface at some point in time.

Computing the propagation of stationary level sets is simple. Given a speed F, the arrival time $T(\mathbf{x})$ at some location $\mathbf{x}$ uses the fact that traveled distance $\mathrm{d}x$ during time interval $\mathrm{d}T$ equals speed multiplied by $\mathrm{d}T$. In one dimension, this gives

$$F \cdot \mathrm{d}T = \mathrm{d}x \;\Leftrightarrow\; F \cdot \frac{\mathrm{d}T}{\mathrm{d}x} = 1, \tag{9.21}$$

and for spaces of arbitrary dimensions, it is

$$F \cdot |\nabla T| = 1, \text{with } T = 0 \text{ on the initial segment boundary } \Gamma. \tag{9.22}$$

Fig. 9.14 Level set evolution of a dynamic level set over time

The gradient ∇T is inverse to the velocity vector. Computing the level set evolution requires finding arrival times T for every scene element such that the condition in Eq. (9.21) is fulfilled. The speed F is a function of locations $\mathbf{x}$. $F(\mathbf{x})$ will be based on internal and external properties. Internal properties guarantee, for instance, the smoothness of the interface. External properties cause the interface to slow down at potential segment boundaries. Examples for different speed functions will be given in Sect. 9.2.6. If the speed function is defined properly, the segment boundary is found if progression per time interval falls below some threshold.

However, using stationary level sets for segmentation severely restricts their applicability. They behave similar to region growing. Dynamic level sets are required if the active contours should be able to reverse the propagation direction when, for instance, stretching modeled by internal energies causes a currently expanding front to shrink.

The wave front of a dynamic level set is represented by embedding the level set function for an N-dimensional image into an $N + 1$-dimensional space with time being the additional dimension. The 0-level set $\phi(\mathbf{x}, t) = 0$ represents the current segment boundary at time t. Level set evolution traces changes $\mathrm{d}\phi/\mathrm{d}t$ of ϕ over time (see Fig. 9.14).

The equation-governing dynamic level set evolution bears some similarity to Eq. (9.21) for stationary level sets. It is

$$\phi_t = -F|\nabla\phi|, \tag{9.23}$$

where ϕ is the derivative of ϕ at any point $\mathbf{x}$ over time.

Evolution of an initial level set function ϕ consists of repeatedly solving an initial value problem using the current level set function $\phi(\mathbf{x}, t)$ as initial value and solving for $\phi(\mathbf{x}, t + \Delta t)$ using the derivative in Eq. (9.23). It requires some care in the selection of appropriate differences to approximate the differential and in the selection of the step size. This will be detailed in the next section. First, however,

Fig. 9.15 Definitions for the level set curve C and its evolution over time

we will need to derive Eq. (9.23) in order to understand the meaning of its components.

For arriving at Eq. (9.23), an abstract formulation of all points on the level set is defined, plugged into the level set term, and then differentiated with respect to time t (see Fig. 9.15):

- The level set function at time t is $\phi(\mathbf{x}, t)$
- The set of points $\mathbf{p}$ on the level set interface C at time $t = 0$ is $C(\mathbf{p}, 0) = \{\mathbf{x} \mid \phi(x, 0) = 0\}$
- The set of points $\mathbf{p}$ on the level set interface C at time t is $C(\mathbf{p}, t) = \{\mathbf{x} \mid \phi(\mathbf{x}, t) = t\}$
- All points on the interface at time t are $C(t) = \phi^{-1}(t)$.

The definitions are used to differentiate the level set equation $\phi(\mathbf{x}, t) = 0 \Leftrightarrow \phi(C(t), t) = 0$:

$$\frac{\mathrm{d}\phi(C(t),t)}{\mathrm{d}t} = 0. \tag{9.24}$$

Applying the chain rule[3] results in

$$\frac{\partial\phi(C(t),t)}{\partial t} = \frac{\partial\phi(C(t),t)}{\partial C(t)}\frac{\partial C(t)}{\partial t} + \frac{\partial\phi(\mathbf{x},t)}{\partial t}. \tag{9.25}$$

The different terms on the left-hand side describe the propagation of the interface. Since $C(t)$ are the points on the interface at time t, the expression

[3]The chain rule for a function g of several functions $f_1, f_2, \ldots, f_n$ is $\frac{\mathrm{d}[g(f_1(t),f_2(t),\ldots,f_n(t))]}{\mathrm{d}t} = \frac{\mathrm{d}g}{\mathrm{d}f_1}\frac{\mathrm{d}f_1}{\mathrm{d}t} + \frac{\mathrm{d}g}{\mathrm{d}f_2}\frac{\mathrm{d}f_2}{\mathrm{d}t} + \cdots + \frac{\mathrm{d}g}{\mathrm{d}f_n}\frac{\mathrm{d}f_n}{\mathrm{d}t}$.

$\partial\phi(C(t), t)/\partial C(t)$ is the partial derivative of the level set function with respect to the interface. This is just the gradient of the level set function. It is abbreviated by

$$\nabla\phi(\mathbf{x}, t) = \frac{\partial\phi(C(t), t)}{\partial C(t)}. \tag{9.26}$$

The reference to the domain is often omitted. The term is referred to as $\nabla\phi$.

The term $\partial C(t)/\partial t$ is the derivative of the interface with respect to time. It describes the (intended) propagation speed of the evolving interface. This term will be the container of curve- and image-dependent information that governs interface evolution.

Finally, the term $\partial\phi(\mathbf{x}, t)/\partial t$ describes the change of the level set function over time. Since this entity needs to be computed to evolve the level set, the equation is rearranged accordingly. With setting $\phi_t = \partial\phi(\mathbf{x}, t)/\partial t$ (again omitting the domain variable), the evolution equation is compactly written as follows:

$$\phi_t = -\nabla\phi\frac{\partial C(t)}{\partial t}. \tag{9.27}$$

Level set evolution is orthogonal to the interface. With unit vector $\nabla\phi/|\nabla\phi|$ which is orthogonal to C, the speed F of the curve evolution $\partial C(t)/\ \partial t$ in this direction is

$$F = \frac{\partial C(t)}{\partial t}\frac{\nabla\phi}{|\nabla\phi|} \Leftrightarrow \frac{\partial C(t)}{\partial t} = F\frac{|\nabla\phi|}{\nabla\phi}. \tag{9.28}$$

Hence, speed F can be included in Eq. (9.27) arriving at the formulation $\phi_t = -F|\nabla\phi|$ of Eq. (9.23).

Given an appropriate speed function F, Eq. (9.23) is the base for computing a level set segmentation. Internal and external properties will be defined for F so that the interface slows down at the segment boundaries. The decrease in speed can be used for terminating the evolution.

The formulation of the level set evolution in Eqs. (9.21) and (9.23) is attractive because it separates considerations for stable computation (how to set Δt for approximating the update of the level set function, how to approximate $\nabla\phi$) from definitions of segmentation constraints in F. Hence, given an appropriate implementation of the approximation scheme, a user can change segmentation characteristics without having to worry about stability issues. It is worthwhile to understand the approximation nonetheless, since the selection of a speed function F may influence the speed of interface progression.

9.2.3 Schemes for Computing Level Set Evolution

Level set evolution can be computed explicitly or implicitly. Although implicit computation is by far the most often applied solution, we will begin with a short characterization of explicit level set evolution. It is very similar to the evolution of explicit active contours and shares its advantages and disadvantages. The method is called *Lagrangian solution* or *marker particle solution*. The latter describes the approach pretty well (see Fig. 9.16). The current location of the interface is discretized by uniform sampling. Each sample point is a marker particle that is tracked over time. Since evolution is orthogonal to the interface, normals are estimated and evolved according to the speed function. The solution is easy to implement but has some problems:

- Changes of topology are difficult to deal with. The Lagrangian solution defines topology explicitly by initial adjacencies between marker particles. An extra rule is needed that produces change of topology.
- Particle tracking requires regular resampling, since evolution of the interface will change the density of the distribution of marker particles along the level set.

Hence, the marker particle solution is only appropriate when the initial level set placement is already close to the expected final result, so that neither of the two problems arises.

The implicit solution (called *Eulerian solution*) computes the value of the level set function ϕ at each point in time at each location $\mathbf{x}$ in the scene. Hence, the level set function is represented instead of the interface. Sampling depends on image

Fig. 9.16 In the marker particle solution, markers on the 0-level set are tracked over time

Fig. 9.17 Example of a simple wave $f(x)$ propagating with constant speed from *left* to *right*

resolution and not on marker locations on the interface. The topology of the interface is a result of evaluating $\phi(\mathbf{x}, t) = 0$.

A detailed derivation of the Eulerian solution is given in Sethian (1998). We will highlight aspects that relate to the robustness of the evolution process.

Given the level set equation $\phi_t = -F|\nabla\phi|$, the evolution process for $\phi(t)$ is

$$\phi(t+\Delta t) = \phi(t) + \Delta t \cdot \phi_t = \phi(t) - \Delta t \cdot F|\nabla\phi| \tag{9.29}$$

An appropriate step size for the time step Δt needs to be found, and an appropriate approximation for the gradient $\nabla\phi$ has to be computed. It will turn out that the gradient approximation depends on the local propagation of the interface resulting in a so-called *upwind scheme*. The step size will depend on the fastest propagation speed in the domain.

In order to illustrate this, we will follow the explanation in Sethian (1998) and discuss the solution in a 1d domain. We will begin with a simple 1d wave $u(x, t)$ that travels with constant unity speed in direction of the positive x-axis (see Fig. 9.17). Given an initialization $u(x, 0) = f(x)$, the solution for this propagation is

$$u(x,t) = f(x-t) \tag{9.30}$$

and the representation by a differential equation is

$$u_x = -u_t \;\Leftrightarrow\; u_x + u_t = 0, \tag{9.31}$$

where the subscripts denote differentiation with respect to x and t, respectively.

If $u(x, 0) = f(x)$ is the initial placement of a level set, the speed function for this process would be $F = 1$. For computing the evolution, the two differentials have to be approximated (see Fig. 9.18). There is one possible solution for u_t:

$$u_t \approx D^{+t}u \equiv u_t(x,t) = \frac{u(x,t+k) - u(x,t)}{k}, \tag{9.32}$$

where k is the step size along the time axis.

Fig. 9.18 For computing the level set propagation, differentials need to be approximated by differences in x- and t-direction

Since the function is evolved along t, the approximation of the differential in x can be in arbitrary direction, resulting in the following three possible solutions:

$$u_x \approx D^{+x}u \equiv u_x(x,t) = \frac{u(x+h,t) - u(x,t)}{h} \tag{9.33}$$

$$u_x \approx D^{-x}u \equiv u_x(x,t) = \frac{u(x,t) - u(x-h,t)}{h} \tag{9.34}$$

$$u_x \approx D^{0x}u \equiv u_x(x,t) = \frac{u(x+h,t) - u(x-h,t)}{2h} \tag{9.35}$$

where h is the step size along the x-axis.

It depends on the wave propagation which of the three differences should be taken. The wave in this example does not change its shape. Hence, the differential at time t travels along the propagation direction to another position at time $t + k$. Estimating the differential by a difference should therefore interpolate at time t over an interval that contains the location of the differential $u_x(x, t + k)$. D^{+x} must be chosen if the wave propagates from right to left, and D^{-x} otherwise (see Fig. 9.19 for an example).

If the waves were real ocean waves, this direction would be against the direction of the wind that drives the waves. Hence, it is called an *upwind scheme*. Taking the downwind difference instead would always extrapolate the differential, which would make the solution instable.

Choosing the upwind scheme does not automatically guarantee that the approximation is always interpolating. Extrapolation is avoided only if a step size k is chosen such that the wave propagation does not exceed the propagation h of the wave everywhere on the line x (see Fig. 9.20).

In the example of Eq. (9.31), the direction toward the origin is upwind, since the wave moves to the right. Thus, D^{-x} is the difference to select. Since the wave is moving with constant speed everywhere, the appropriate step size must ensure that $k \leq h$. The evolution of the wave $u_x + u_t = 0$ is then approximated by

Fig. 9.19 In this example, D^{-x} is the difference in upwind direction. It would produce the correct estimate $u(5, t + k) = 4$. Choosing D^{+x} would produce $u(5, t + k) = 5$ and choosing D^{0x} would result in $u(5, t + k) = 4.5$

Fig. 9.20 The step size for the time steps has to be chosen so that estimates of derivatives in upwind direction result in an interpolation. In the example, step k_1 would be unstable, while k_2 and k_3 would result in a stable estimate

$$\frac{u(x,t+k)-u(x,t)}{k}+\frac{u(x,t)-u(x-h,t)}{h}=0 \tag{9.36}$$

leading to the following iteration step

$$u(x,t+k)=u(x,t)-kD^{-x}u(x,t)=u(x,t)-\frac{k}{h}[u(x,t)-u(x-h,t)]. \tag{9.37}$$

There is not much change if propagation is a function of x or $u(x)$ that may be positive at some locations x and negative at others (see Fig. 9.21). For an arbitrary propagation $a(x)$, selecting a switch function

$$\nabla u=\max(0,a(x))D^{-x}u-\min(0,a(x))D^{+x}u \tag{9.38}$$

always chooses the upwind direction.

Fig. 9.21 If the direction of propagation depends on the location, the upwind direction has to be selected accordingly

9.2.4 Computing Stationary Level Set Evolution

Stationary level set evolution uses the upwind scheme for stable estimates of arrival times. It will be explained for 2d level sets but easily extends to 3d. Arrival times $T(x, y)$ need to be computed at every location (x, y) given an initial level set based on Eq. (9.21). Resolving (9.21) with respect to arrival times results in

$$|\nabla T(x,y)| = \frac{1}{F(x,y)} \Rightarrow \sqrt{(\nabla T(x,y))^2} = \frac{1}{F(x,y)} \tag{9.39}$$

for a given speed function F. The approximation $|\nabla T|$ by upwind differences is then

$$\left[\begin{array}{c} \max(D^{-x}T(x,y),0)^2 + \min(D^{+x}T(x,y),0)^2 \\ + \max(D^{-y}T(x,y),0)^2 + \min(D^{+y}T(x,y),0)^2 \end{array}\right]^{\frac{1}{2}} = \frac{1}{F(x,y)}. \tag{9.40}$$

This is a set of nonlinear equations where $T(x, y)$ are the unknowns. It is solved by the following iterative algorithm:

- Initialize T with $T(x, y) = 0$ for all locations on the initial boundary Γ and $T(x, y) = \infty$ for all other locations.
- Correct all equations of the equation system iteratively.
- Terminate the process when the change of all T's falls below some threshold.

The method requires several runs, since equations are dependent on each other via the difference operators D (hence, correcting one equation will change T-values in other equations).

Computing arrival times in this fashion is stable but very slow. In the first runs, only pixels in the immediate vicinity of the boundary Γ will converge quickly to their true values. Far away pixels correct T-values based on equally wrong T-values in their vicinity.

It is more efficient to restrict updates to those pixels that are neighbored by pixels of which the correct arrival time is known. This is done by the *fast marching method* of Sethian (1996). The initial level set Γ can be an infinitesimal circle indicating the location of the structure to be segmented or it may already roughly approximate the true object boundary.

Since the level set function is stationary, the speed can be computed in advance for each location $\mathbf{x}$. Shortest arrival times are then computed by turning the problem into a shortest path computation. Local costs at a location $\mathbf{x}$ depend on the speed with which the moving front passes $\mathbf{x}$.

This is similar to Dijkstra's minimum cost path algorithm used, e.g., for the image foresting transform. However, Dijkstra's algorithm computes shortest connections along edges between two locations $\mathbf{x}_1$ and $\mathbf{x}_2$. This is not necessarily the shortest connection in the underlying continuous Euclidean space of the level set function (see Fig. 9.22). Thus, the fast marching algorithm computes arrival times by propagating the front in Euclidean space.

The fast marching algorithm for computing arrival times from a set of start pixels $\mathbf{X}_0 = \{\mathbf{x}_{0,i}\}$ keeps (see Fig. 9.23)

- a list of *known* nodes for which arrival times have already been computed and which are completely surrounded by other known nodes.
- a list of *far* nodes to which arrival times have not yet been computed.
- a list of *trial* nodes to which arrival times have been computed and which are adjacent to at least one far node.

Fig. 9.22 The major difference between fast marching and Dijkstra's minimum cost path algorithm is that the latter always proceeds along edges, whereas the former estimates costs in the correct direction of propagation

Fig. 9.23 The fast marching method proceeds by extending the wave front from nodes with known arrival times into trial nodes that are adjacent to known nodes

Fig. 9.24 The two different cases for computing arrival time: **a** the two closest nodes are opposite from each other. **b** The two closest nodes are at a 90° angle

Initially, all nodes except those in $\mathbf{X}_0$ are far nodes. Nodes in $\mathbf{X}_0$ are known nodes if they are completely surrounded by other nodes in $\mathbf{X}_0$. Otherwise, they are trial nodes. The algorithm iteratively selects the trial node $\mathbf{x}_{\min}$ with shortest arrival time, removes it from the trial set, puts it into the known-node set, and computes the propagation into the adjacent nodes of $\mathbf{x}_{\min}$.

Computing arrival times for a node $\mathbf{x}$ neighboring $\mathbf{x}_{\min}$ requires the estimation of the front passing known nodes in the vicinity of $\mathbf{x}_{\min}$. Two cases have to be differentiated (see Fig. 9.24):

1. The two nodes $\mathbf{x}_{n1}$ and $\mathbf{x}_{n2}$ with lowest arrival time $u(\mathbf{x}_{n1})$ and $u(\mathbf{x}_{n2})$ of all four neighbors of $\mathbf{x}$ are opposite to each other. In this case, the front is moving from $\mathbf{x}_{\text{closest}} = \text{argmin}(u(\mathbf{x}_{n1}), u(\mathbf{x}_{n2}))$. The arrival time at $\mathbf{x}$ is $u(\mathbf{x}) = u(\boldsymbol{x}_{\text{closest}}) + c(\mathbf{x})$.
2. The two nodes $\mathbf{x}_{n1}$ and $\mathbf{x}_{n2}$ are at a 90° angle with respect to $\mathbf{x}$. The front has passed through these nodes. The propagation direction is computed under the assumptions that

- the front moves orthogonal to its boundary;
- that this boundary can be approximated by a straight line between $\mathbf{x}_{n1}$ and $\mathbf{x}_{n2}$,; and
- that the speed does not accelerate between $\mathbf{x}_{\min}$ and $\mathbf{x}$.

Iterative propagation is stopped when the current level set best approximates the object boundary. The stopping criterion depends on the evolution of the front. Since local propagation speed (the speed term) is usually defined in a way so that the speed is low when the gradient is high, the front should move slowly when reaching the object boundary. A slow propagation speed is indicated by a fast increase of arrival times. This is a criterion for stopping the propagation.

9.2.5 Computing Dynamic Level Set Evolution

The upwind scheme for computing dynamic level set evolution represented by $\phi_t = -F|\nabla\phi|$ needs to consider that the sign of ϕ is unknown. Hence, the switch has to be applied first to select the correct upwind direction depending on the sign of ϕ. For the simple case of $F = 1$, Eq. (9.23) simplifies to $\phi_t + |\nabla\phi| = 0 \Rightarrow \phi_t + \sqrt{(\nabla\phi)^2} = 0$ resulting in the following iteration step for a 1D level set

$$\phi^{t+1}(x) = \phi^t(x) + \Delta t\sqrt{\max(D^{-x}\phi^t(x), 0)^2 + \min(D^{+x}\phi^t(x), 0)^2} \tag{9.41}$$

If $F \neq 1$ and its sign is unknown, the switch has to be applied again, resulting in the final upwind scheme for 1D level sets with arbitrary F

$$\phi_{t+1}(x) = \phi^t(x) + \Delta t\left[\begin{array}{l}\max(F(x), 0)\sqrt{\max(D^{-x}\phi^t(x), 0)^2 + \min(D^{+x}\phi^t(x), 0)^2} \\ + \min(F(x), 0)\sqrt{\max(D^{+x}\phi^t(x), 0)^2 + \min(D^{-x}\phi^t(x), 0)^2}\end{array}\right]. \tag{9.42}$$

Changing the dimension does not change much in this scheme, except that the upwind direction has to be selected based on the gradient direction $\nabla\phi$ in multi-dimensional space. For a 3D space, for instance, the upwind scheme is

$$\phi_{t+1}(x, y, z) = \phi_t(x, y, z) + \Delta t\left[\max(F(x, y, z), 0)\sqrt{\nabla^+} + \min(F(x, y, z), 0)\sqrt{\nabla^-}\right] \tag{9.43}$$

with

$$\nabla^{+} = \begin{bmatrix} \max(D^{-x}\phi^{t}(x,y,z),0)^{2} + \min(D^{+x}\phi^{t}(x,y,z),0)^{2} \\ +\max(D^{-y}\phi^{t}(x,y,z),0)^{2} + \min(D^{+y}\phi^{t}(x,y,z),0)^{2} \\ +\max(D^{-z}\phi^{t}(x,y,z),0)^{2} + \min(D^{+z}\phi^{t}(x,y,z),0)^{2} \end{bmatrix} \tag{9.44}$$

and

$$\nabla^{-} = \begin{bmatrix} \max(D^{+x}\phi^{t}(x,y,z),0)^{2} + \min(D^{-x}\phi^{t}(x,y,z),0)^{2} \\ +\max(D^{+y}\phi^{t}(x,y,z),0)^{2} + \min(D^{-y}\phi^{t}(x,y,z),0)^{2} \\ +\max(D^{+z}\phi^{t}(x,y,z),0)^{2} + \min(D^{-z}\phi^{t}(x,y,z),0)^{2} \end{bmatrix} \tag{9.45}$$

Given appropriate time differences Δt, this scheme can be repeatedly applied to update the level set function. However, computation would be very slow, since the equation has to be evaluated for every time step and every location. Furthermore, since the step size is restricted by the fastest progress anywhere in the scene, progress can be slow when the interface slowly reaches the segment boundary but ϕ is changing very fast somewhere else in the image.

The *narrowband method* (Adalsteinsson and Sethian 1995) makes use of the fact that the state of ϕ is only of interest close to the 0-level set. The time-consuming computation of the level set everywhere is replaced by a method where ϕ is computed only for positions $\mathbf{x}$ that are close to the current level set (i.e., where $\phi(\mathbf{x}, t) = 0$, see Fig. 9.25). Computation is still slower than fast marching because ϕ has to be updated in a band $|\phi(\mathbf{x}, t_0)| < k_{\max}$ around the initial level set $\phi(\mathbf{x}, t_0) = 0$ instead of just propagating a front $\phi(\mathbf{x})$. It surpasses fast marching capabilities in that it computes a dynamic level set in the band, hence enabling a front that shrinks and grows according to the data information as long as the level set $\phi(\mathbf{x}, t) = 0$ does not exceed the initial band $|\phi(\mathbf{x}, t_0)| < k_{\max}$.

Fig. 9.25 The narrowband method computes the level set function only in a narrow band around the 0-level set. The boundary locations of the narrow band are labeled "land mines." Hitting a land mine with the 0-level set requires recomputing the level set function in the narrow band around the current 0-level set

Pixel sites in the narrow band are labeled *alive*, since their correct level set value is known and can be used for level set propagation. Sites at the border of the narrow band are called *land mines*. If the level set hits a land mine at some time t_e, the level set function $\phi(\mathbf{x}, t_e) = k(t_e)$ has to be computed for all values $k(t_e)$, since the function ϕ has not been evolved outside the band and derivatives there are not valid. The time t_e is constant for $\phi(\mathbf{x}, t_e) = k(t_e)$ and may be rewritten as $\phi_{te}(\boldsymbol{x}) = k_{te}$. Since $\phi_{te}(\mathbf{x})$ does not depend on t, it can be computed using the fast marching method. Then, a new band $|\phi(\mathbf{x}, t_e)| < k(t_e)$ is defined and the algorithm proceeds.

9.2.6 Segmentation and Speed Functions

Initial placement of the level set at time t_0 introduces position information into the segmentation. A boundary described by a level set is placed somewhere close to an object in an image. The object may consist of several segments. If forces attract the level set toward an object boundary and push it away from interior or exterior points (e.g., by using intensity gradient information), the level set will evolve into several closed boundaries following the gradients in the vicinity of the initial placement. Using the curvature term lets the level set disregard small regions with high gradients caused by noise.

The level set approach can also be used for a non-localized search. The initial level set (possibly representing boundaries of several disconnected objects) is then placed in such a way that it encloses the whole image. The level set will still evolve in a set of smooth boundaries close to high-gradient regions.

The *speed function* is the only parameterizable entity which influences the result of a level set segmentation. It encodes interface properties and data-driven properties. A simple speed function is

$$F = F_{\text{img}}(v - \varepsilon\kappa) \tag{9.46}$$

The term v is an advection force. It drives the interface either to the outside (positive advection, $v > 0$) or to the inside (negative advection, $v < 0$). It is a necessary component if segmentation shall be started from some seed element, which is so far away from the segment that the data term cannot guide the evolution of the interface.

The second term relates level set evolution to curvature κ weighted by ε (see Fig. 9.26). It keeps the interface from developing local high-curvature details if the curvature term increases with negative curvature. High-curvature details may eventually develop into sharp corners (see Fig. 9.27). Since this is unwanted in most applications in the medical field (there are few structures in the human body with sharp corners), the curvature term is used almost always in this fashion.

Curvature in 2D is the second derivative of the curve. The solution is more difficult in higher dimensions since infinitely many curves pass a point on a surface. An average curvature for interfaces in arbitrary dimensions can be defined using the divergence of surface normals. Divergence $\nabla \cdot \mathbf{f}$ of a vector field $\mathbf{f}$ describes the

Fig. 9.26 Curvature is the change of surface orientation per unit length. Curvature of the boundary can be positive (in locally convex parts) or negative (in locally concave parts)

Fig. 9.27 A negative curvature term in the speed function will smoothen the boundary and keep it from developing sharp corners

degree of dispersion of the field at source (positive divergence) and the degree of concentration at sinks (negative divergence) (see Fig. 9.28). Divergence is

$$\nabla \cdot f = \sum_{i=1}^{n} \frac{\partial}{\partial x} f_i \tag{9.47}$$

The divergence of the normal of a 0-level set is positive at convex locations and negative at concave locations (see Fig. 9.29). Using the definition above, the divergence for a 2D level set is

$$\kappa_{2d} = \nabla \cdot \mathbf{n} = \nabla \frac{\nabla \phi}{|\nabla \phi|} = \frac{\partial \left[\frac{\phi_x}{\left(\phi_x^2 + \phi_y^2\right)} \right]}{\partial x} + \frac{\partial \left[\frac{\phi_y}{\left(\phi_x^2 + \phi_y^2\right)} \right]}{\partial y} = -\frac{\phi_{xx} - 2\phi_x \phi_y \phi_{xy} + \phi_{yy}}{\left(\phi_x^2 + \phi_y^2\right)^{\frac{3}{2}}}. \tag{9.48}$$

Computing curvature in higher dimensions follows the same structure adding derivatives of the derivative of the level set function with respect to all directions.

To some extent, the curvature term may even be used to fill in missing information. If salient image features are missing in some parts of the image but the

Fig. 9.28 Divergence characterizes the flow of a vector field

Fig. 9.29 The divergence of the normal of the 0-level set is the curvature of this level set

shape of objects is known and simple, curvature can represent this shape. An example would be the segmentation of circles or spheres of unknown size. The curvature of a sphere is constant and the reciprocal of its radius. Penalizing deviation from this expectation will cause a level set to evolve into circles or spheres if no image information is present. Hence, if such a circle is partially visible by having high gradients only at parts of its boundary, the level set will evolve into a boundary following these high-gradient regions as long as the curvature is close to the expected curvature. If this is not true because the gradients do not represent the circle boundary, the level set will follow the circular shape instead.

In principle, this can be extended to describe arbitrary expected shapes. However, including shape information becomes increasingly difficult when shapes become complex. In such case, it is easier to integrate explicit shape knowledge as prior information into the model (Leventon et al. 2000; Rousson and Paragios 2002; Paragios 2003).

Data knowledge is introduced by the last term F_{img}. Since the interface should stop at segment boundaries, a force that is inversely proportional to the intensity gradient is a good candidate for F_{img}. Examples suggested in the literature are

$$F_{\text{img}}(\mathbf{x}) = \frac{1}{1 + \alpha|\nabla(G_\sigma * I(x))|} \tag{9.49}$$

and

$$F_{\text{img}}(\mathbf{x}) = \exp(-\alpha|\nabla(G_\sigma * I(x))|). \tag{9.50}$$

In both cases, α controls the strength with which the speed is influenced by this term, and G_σ denotes a convolution of the intensity function I with a Gaussian with standard deviation σ.

On inspection, it becomes obvious that the image term will never be 0 and consequently the propagation according to (9.46) will never be 0 as well. Keeping the image term from stopping the propagation is intentional, as it is usually not known beforehand what the exact gradient strength is at a boundary and whether this is a unique feature for the segment boundary (otherwise segmentation would be trivial).

Since the level set evolution will not stop automatically, a termination criterionhas to be given. Commonly used criteria are to stop evolution if the average or maximum evolution speed falls below some threshold or if an evolution step does not change the current pixel labeling (i.e., no foreground pixels become background or vice versa).

9.2.7 Geodesic Active Contours

The desired level set segmentation can be represented as minimum energy of an energy functional of the level set function. Calculus of variation is then used to define a gradient descent on this functional given an initial level set function. Different to the solution in the previous section, where an initial value problem is solved without ensuring convergence, an explicit optimum is defined that can be reached by the iterative gradient descent. The optimization cannot be separated from the definition of constraints for the level set. This is different to front propagation where constraints are defined in the speed function but the upwind scheme for propagating the front remains the same for all suitable speed functions.

The minimizationproblem for computing a snake can be formulated in the level set framework as a variational problem. It is independent of the dimension of the active contours and allows topology changes during evolution of the active contour.

The approach received its name because the problem is formulated as finding a locally shortest contour in a non-Euclidean space. For doing this, the original snake is altered by removing the second derivative from the minimization equation. The authors of Caselles et al. (1997) showed that this does not substantially influence the smoothness constraint of the curve.

The condition for an ideal curve $C(p)$ is then to minimize

$$E(C) = \int_0^1 |C'(p)|^2 \mathrm{d}p + \lambda \int_0^1 g\left(|\nabla I(C(p))|^2\right) \mathrm{d}p = E_{\text{internal}}(C) + \lambda E_{\text{external}}(C) \tag{9.51}$$

where $p = p(\mathbf{x})$ are locations on the curve, I is the image function, ∇I its gradient, g is an arbitrary function that strictly decreases with the gradient, and λ is a

Fig. 9.30 If the strength of the gradient is used as 3rd dimension for the embedding of a 2D object in 3D space, locally shortest curves on this surface (the geodesics) are the segment boundaries that are found by geodesic active contours (which may change topology if needed)

parameter that controls the influence of the internal energy versus the external energy of C.

Caselles et al. showed that this corresponds to a search for a minimum length curve in a Riemannian space that is induced by the image information (see Fig. 9.30). Riemannian space is a curved space that may be thought as being embedded in a higher-dimensional Euclidean space. Distances are computed on the curved manifold. Hence, they depend on the distance traveled along the manifold and on the local curvature. This may sound a bit complicated, but representing the energy minimization problem of active contours as a shortest path problem in well-defined non-Euclidean space provides a simple solution to the optimization problem. It enables us to make use of theorems about shortest distances in this space. It can be shown that the solution to Eq. (9.51) is to evolve the curve in time t in the following manner[4]

$$\frac{\partial C(t)}{\partial t}(p) = [g(I(p))\kappa - (\nabla g(p) \cdot \mathbf{n})]\mathbf{n} \tag{9.52}$$

where κ is the curvature of the curve and $\mathbf{n}$ is normal to the curve pointing inward. Hence, the curve will move toward high-gradient areas, where g is small, while attempting to smooth out its course by minimizing the curvature.

Explicit evolution of C, however, may pose a problem when optimization requires a change of topology. Hence, the optimization is formulated using the level set approach. The curve $C(t)$ is taken as the 0-level set of a level set function $\phi(\mathbf{x}, t)$. Optimization is not by wave propagation but by using variational calculus (see Sect. 14.2) to minimize (9.51). Hence, level sets of this kind are referred to as *variational level sets*.

[4]It is a solution under some specializing assumptions that are necessary to define a unique optimum because the formulation contains a free parameter. Details are described in Caselles et al. (1997). It does not restrict the use of this formalism to find an optimal active contour.

The Euler–Lagrange equation

$$\frac{\mathrm{d}E}{\mathrm{d}f} = \frac{\mathrm{d}}{\mathrm{d}x}\frac{\partial F}{\partial f'} - \frac{\partial F}{\partial f} = 0 \tag{9.53}$$

for a functional of the form $F(f(x), f'(x), x)$ is created if we substitute $f(x)$ by $\phi(\mathbf{x})$ and consequently $f'(x)$ by $\nabla\phi(\mathbf{x})$. In Casselles et al. (1997), the authors showed in Appendix B and C that this derivative results in the following evolution of $\phi(\mathbf{x}(t), t)$ with respect to time

$$\begin{aligned}\frac{\partial\phi}{\partial t}(\mathbf{x}) &= |\nabla\phi|\mathrm{div}\left(g(I(\mathbf{x}))\frac{\nabla\phi(\mathbf{x})}{|\nabla\phi(\mathbf{x})|}\right)\\ &= g(I(\mathbf{x}))|\nabla\phi(\mathbf{x})|\kappa(\mathbf{x}) + \nabla g(I(\mathbf{x}))\nabla\phi(\mathbf{x})\end{aligned} \tag{9.54}$$

The function $\phi(\mathbf{x}(0), 0)$ at $t = 0$ is the signed distance function, which is evolved according to Eq. (9.54). It has been shown that the choice of the initial function does not influence the result. Given its initial placement, the level set will evolve in a shortest distance curve in a Riemannian space that is defined by image attributes. The function g is

$$g(\mathbf{x}) = \frac{1}{1 + |\nabla[I(\mathbf{x}) * s(\mathbf{x})]|^p}, \tag{9.55}$$

where s is a smoothing kernel with which I is convolved and $p = 1, 2$.

The energy E_{internal} in (9.51) is a smoothness term that makes the active contour insensitive with respect to noise. In Westin et al. (2000), a variant of the geodesic active contour has been presented that can be made—based on prior knowledge—locally adaptive to enforce smoothness in some parts of the image while following image detail in other parts.

The energy E_{external} is intended to attract the geodesic active contour toward salient image features. Paragios et al. (2004) showed that using gradient vector flow (see Sect. 9.1.2) as external energy improves the speed of convergence.

9.2.8 Level Sets and the Mumford–Shah Functional

TheMumford–Shah functionaldescribes a segmentation based on regional and boundary image attributes (Mumford and Shah 1989). Segmentation by minimizing the Mumford–Shah functional is based on reasonable generic attributes for an optimal segmentation. Hence, numerous schemes have been developed to define a segmentation for various specializations of this functional. It has been shown that level set segmentation can be defined in a way that optimizes a specific variant of the Mumford–Shah functional.

Fig. 9.31 An optimal segmentation according to the Mumford–Shah functional is a description of an image by a non-overlapping set of smooth functions with local support and minimum boundary length. It essentially creates a cartoon-like mapping of the original image with smoothed boundaries

In foreground segmentation, a segmentation is a function $f_{\text{seg}}(\mathbf{x})$ which is 1 for scene elements $\mathbf{x}$ that belong to the segment and 0 otherwise. Since many functions follow this definition, additional constraints based on desired attributes of a segment are needed. The Mumford–Shah functional

$$E(c_1, c_2, \Gamma) = \alpha \int_{\Omega_i} (u(\mathbf{x}) - c_1)^2 \mathrm{d}\mathbf{x} + \beta \int_{\Omega_{ei}} (u(\mathbf{x}) - c_2)^2 \mathrm{d}\mathbf{x} + \gamma \int_{p} |\Gamma(p)| \mathrm{d}p \quad (9.56)$$

provides these constraints (Tsai et al. 2001). The equation is a specialization of the general functional of Mumford and Shah (1989) that allows for an arbitrary number of segments (see Fig. 9.31).

The first two terms of (9.56) describe the deviation of intensity $u(\mathbf{x})$ of interior elements $\mathbf{x} \in \Omega_i$ and exterior elements $\mathbf{x} \in \Omega_e$ from expected (average) intensities c_1 and c_2 in the interior and exterior regions, respectively. Similar to the Bayesian formulation in Sect. 7.1, they ensure that the optimal segmentation separates the scene into interior and exterior elements so that they closely follow the expected characteristics of foreground and background elements.

The last term is the derivative of the curve Γ at points p along the curve. It enforces smoothness of the boundary between interior and exterior points. The effect of this term is similar to that of the internal energy terms of an active contour. It takes into account that the observable boundary in the image may be distorted by noise or may be partially missing. The parameters α, β, and γ govern the influence of the different terms on the desired optimal segmentation.

9.2.8.1 The Level Set Terms

Finding a foreground segmentation requires to find a function that minimizes (9.56) given an image. Including the region attributes of (9.56) in a level set form has first been presented by Chan and Vese (2001). The method is known as *active contours*

without edges, since the data term controlling the level set evolution is based on region attributes and not on edges.

For separating foreground from background, the two regions Ω_i and Ω_e have to be found. This can be conveniently expressed by a level set function $\phi(\mathbf{x})$, which is defined in the complete domain of $\mathbf{x}$, if the Heaviside function

$$H(z) = \begin{cases} 1, & z \geq 0 \\ 0, & z < 0 \end{cases} \tag{9.57}$$

is used. Foreground elements are now all elements for which $H(\phi(\mathbf{x})) = 1$, and background elements are those for which $H(\phi(\mathbf{x})) = 0$. With these definitions, the Mumford–Shah functional simplifies to (now using the weighting variables λ_1, λ_2, and v of Chan and Vese 2001)

$$\begin{aligned} E(\phi, c) = \lambda_1 &\int_R (u(\mathbf{x}) - c_1)^2 H(\phi(\mathbf{x}))\mathrm{d}\mathbf{x} \\ &+ \lambda_2 \int_R (u(\mathbf{x}) - c_2)^2 (1 - H(\phi(\mathbf{x})))\mathrm{d}\mathbf{x} \\ &+ v \int_R |\nabla H(\phi(\mathbf{x}))| \end{aligned} \tag{9.58}$$

The optimal segmentation is a function $\phi(\mathbf{x})$ that minimizes E (see Fig. 9.32 for an example). Similar to geodesic active contours, the optimal level set is computed using variational calculus (see Sect. 14.2). It is a matter of applying standard differentiation rules (and remembering that the derivative of the Heaviside function is the delta function δ) to arrive at the following Euler–Lagrange equation for the level set minimization

$$\frac{\partial E}{\partial \phi} = \delta(\mathbf{x}) \left[\lambda_1 (u(\mathbf{x}) - c_1)^2 - \lambda_2 (u(\mathbf{x}) - c_2)^2 - v \nabla \frac{\nabla \phi(\mathbf{x})}{|\nabla \phi(\mathbf{x})|} \right] \tag{9.59}$$

which gives the update rule

$$\frac{\partial \phi}{\partial t} = \delta(\mathbf{x}) \left[-\lambda_1 (u(\mathbf{x}) - c_1)^2 + \lambda_2 (u(\mathbf{x}) - c_2)^2 + v \nabla \frac{\nabla \phi(\mathbf{x})}{|\nabla \phi(\mathbf{x})|} \right]. \tag{9.60}$$

Since the Heaviside function and its derivative, the δ-function, are inherently unstable, they are replaced by a differentiable approximation such as the sigmoid function.

In Eq. (9.58), c_1 and c_2 approximate the appearance in the foreground and background region. This could be, for instance, the average intensity of the enclosed regions. If domain knowledge exists about the expected appearance in

Fig. 9.32 Active contours without edges for the segmentation of vessels in a projection image: segmentation is fairly independent of the initial level set. The result is influenced by the homogeneity term. This can be seen for the smaller vessels at the periphery where the signal difference is too small for a good segmentation

foreground and background, this could be used as well. The two parameters λ_1 and λ_2 represent the expected reliability of the approximation by c_1 and c_2. The parameter ν enforces short and therefore smooth boundaries.

The regional terms determine the behavior of the level set. If, for instance, the level set is in the background, the variation of $(u(\mathbf{x}) - c_1)^2$ in the current foreground should be larger than that of $(u(\mathbf{x}) - c_2)^2$ in the current background. Since the two terms are minimized, the level set will move inward toward the true foreground-to-background boundary.

The evolution of variational level sets will always result in the minimum if the functional to be optimized is convex (such as the one above). Hence, the result is independent from the initial level set as long as the initial boundary has nonzero length (see, e.g., Fig. 9.32).

Also, the narrowband technique for speeding up computation is implicitly part of the scheme, since multiplication with the (regularized) δ-function requires updates only in the vicinity of the current 0-level set.

Active contours without edges have later been extended to multi-phase level sets. The differentiation of 2^n segments requires n separate level set functions ϕ_1, …, ϕ_n. For example, four different regions can be separated, if two level set functions ϕ_1 and ϕ_2 are defined (see Fig. 9.33). The regions are then given by

Fig. 9.33 Four different segments can be differentiated, if two level set functions ϕ_1 and ϕ_2 are used

$$
\begin{aligned}
&H(\phi_1(\mathbf{x}))H(\phi_2(\mathbf{x})),\\
&H(\phi_1(\mathbf{x}))(1-H(\phi_2(\mathbf{x}))),\\
&(1-H(\phi_1(\mathbf{x})))H(\phi_2(\mathbf{x})),\\
&(1-H(\phi_1(\mathbf{x})))(1-H(\phi_2(\mathbf{x}))).
\end{aligned}
$$

Alternatively, separate level set functions $r = 1, \ldots, n - 1$ can be assigned to segment an object into $n - 1$ regions plus a background region (Yazdanpanah et al. 2011). A pixel belongs to region r if it has a positive distance to the rth 0-level set. Introducing an extra repulsion term avoids overlapping segments (see Fig. 9.34).

If the functional is convex, fast schemes exist that produce the optimal result independent of initialization. For a non-convex functional, the final segmentation will depend on the initial level set function. This dependence is often desired to add a position constraint to the segmentation.

The level set formulation may also be used to define homogeneity constraints based on the vectors (Chan et al. 2000). Multiple input channels were used by Klann et al. (2011) who reconstructed SPECT images from two independent types

Fig. 9.34 Two separate level set functions may be used as well for generating a multi-label segmentation. However, level set functions may overlap and the overlap region, where both functions are positive, needs to have a semantic definition or needs to be avoided by using repulsion forces between level set functions

of information. The attenuation of γ-rays in tissue and the activity distribution of the tracer are jointly computed by maximizing local contrast and searching for short segment boundaries.

Sometimes, the average intensity is not a very good predictor to determine foreground or background membership of a scene element. Knowledge about the location of the segment boundary—included in active contour approaches—cannot be included in the original formulation of Chan and Vese (2001). If this information is present, the approach can be extended to an active geodesic regions method (Paragios 2002). Given probabilities P_{int} and P_{ext} of a scene element belonging to the interior and exterior of a segment, the level set evolution is

$$\frac{\partial \phi}{\partial t} = \alpha\left(\log\left(\frac{P_{\text{int}}}{P_{\text{ext}}}\right)\right)|\nabla\phi| + (1-\alpha)(g(P_C)\kappa|\nabla\phi| + \nabla\phi\nabla g(P_C)), \tag{9.61}$$

where g decreases with the probability P_C of points of a curve C belonging to the true boundary.

9.2.8.2 Dealing with Intensity Variations

Objects in medical images should have an almost homogeneous intensity because the imaging procedure has intentionally been chosen to discriminate a specific object by its intensity. Sources of intensity variation are zero-mean noise and partial volume effects at the boundary between objects. The former does not pose a problem for in region-based level set methods if the object contrast is higher than the noise level. Boundary-based level set methods require noise gradients that are smaller than boundary gradients. The curvature term counteracts potential boundary errors from noise at the object boundary.

Partial volume effects may be problematic for a region-based level set method if the intensity variation in the background is higher than in the foreground object. This often happens because the background contains many different objects each of which having a different average intensity. The variation due to the partial volume effect may then be attributed to expected background intensity variation causing an underestimation of the extent of the foreground object. The problem can be avoided if the level set formulation contains terms that account for the non-homogeneous background. Boundary-based level sets should not be affected by partial volume effects unless the effect is so strong with respect to the object contrast that it overly reduces gradient strength.

Some imaging devices are also subject to a third source for intensity variation. Measured signal strength varies because the signal-exciting source varies locally—an example is shading in MR imaging—or because the sent signal varies depending on location—examples are ultrasound images and again MR images. The effect can be modeled by a *bias field* that is overlaid to the undistorted signal.

Since intensity variation due to shading often vary slowly, its major components are found in the low-frequency range. Bias field variation should have little to no

Fig. 9.35 Shading in MRI is accommodated by separating the image into blocks and computing the data term in each block separately. Computed features are interpolated using results from neighboring blocks in order to avoid discrepancies at the block boundaries

influence on the image gradient because of this low-pass characteristic. Hence, it is more critical for region-based level sets than for boundary-based level sets.

The original Chan–Vese algorithm (Eq. 9.58) already includes some kind of bias field correction by using the average value in the current foreground and background regions to determine c_1 and c_2 for the next iteration. The gradient descent on the functional penalizes deviation of the image intensity from these averages combined with a model term that emphasizes short boundaries. The method works well if gradient strength varies (e.g., if partial volume effects cause weak gradients), but requires a constant intensity within the segments.

Intensity variationdue to shading can also be accounted for by space partitioning so that the intensity in each partition can be taken as approximately constant. An example is Sokoll et al. (2008) who segment gray matter, white matter, and cerebrospinal fluid in MR brain images. The data is subdivided into partitions of $20 \times 20 \times 20$ voxels and the level set evolution is estimated in each partition separately. Weighted features from a 6-neighborhood of each partition are used for feature computation to avoid discrepancies at partition boundaries (see Fig. 9.35).

If a smoothly varying bias function is part of the level set equation, the discrepancy problem can be avoided altogether. An example is the method of Li et al. (2011a) who compute the data term for the energy functional from the difference

Fig. 9.36 The 0-level set $\phi(\mathbf{x}) = 0$ of a level set function ϕ defines a current segmentation. The local binary fitting energy constrains the level set function. The weighted difference between optimal average foreground and background intensities f_1^{opt} and f_2^{opt} and actual intensity $I(\mathbf{y})$ is minimized. The weight function decreases with distance to $\mathbf{x}$

between image intensity and the product of the expected average intensity and a local bias field. The bias field is generated from local averages in a predefined neighborhood in which the bias field is assumed to be almost constant. This is an adequate assumption for the application to the segmentation in MR images. Three functions—level set function, local bias field and bias-corrected average values in the segment—are then iteratively updated by keeping all but one function constant. Similar solutions are Wang et al. (2010) and Zhang et al. (2010a).

The local binary fitting energy of Li et al. (2007) also considers shading under the assumption that it is locally constant. The algorithm assumes that the local neighborhood at the object boundary is always representable by a binary function. The local binary fitting energy integrates for each point $\mathbf{x}$ in the image differences between functions $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ and the intensities in the neighborhood of $\mathbf{x}$. The neighborhood is defined by a Gaussian kernel that decreases with distance to $\mathbf{x}$. Optimization repeatedly updates the current level set function given f_1^{opt} and f_2^{opt} as local intensity models for foreground and background and then finds new optimal values f_1^{opt} and f_2^{opt} for a given level set (see Fig. 9.36). Since the model intensities vary locally (f_1 and f_2 are functions of $\mathbf{x}$), the method will find boundaries even if the local contrast between foreground and background varies. A variant that can be computed more efficiently is Zhang et al. (2010b) who use a truncated Gaussian kernel.

9.2.8.3 Modeling an Inhomogeneous Background

Background in a medical image is often inhomogeneous and the often implicitly assumed homogeneity for foreground–background segmentation may be an inappropriate representation (see Fig. 9.37).

The simplest way is to ignore the background as in Kosic et al. (2010). The authors drop the background term of Eq. (9.58). Their optimization moves the level set only in a local neighborhood and thus avoids its termination in a trivial minimum (a foreground segment consisting of a single pixel or voxel). Background may

Fig. 9.37 The background is usually less homogeneous than the foreground as it may contain many different objects such as the other abdominal organs in this CT image

also be modeled to consist of many different, homogeneous background objects using a multi-phase model as in Yazdanpanah et al. (2011). However, it makes the segmentation overly complicated if this separation of the background into several segments is not required.

Background may also be represented by a Gaussian mixture model as suggested by Lee et al. (2012) although, finally, they modeled it by a single Gaussian. Different weights of foreground and background terms account for the different accuracy of the homogeneity assumption in the foreground and in the background. This background model becomes weak compared to the foreground model and may be inappropriate. Deviations from the assumed average intensity are mainly meant to account for the influence from 0-mean noise. Gaussian mixtures have been used for background representation by Park et al. (2011) for segmentation of color photographs of flowers.

9.2.9 Avoiding Reinitialization

Level set evolutions using the narrowband technique require reinitialization at the latest when the 0-level set hits the boundary of the narrow band, since everything outside the band has not been updated at all. However, even if the level set function is updated everywhere, it will deform with time. The distance transform is carried out at regular time intervals for avoiding numerical problems.

Efficient schemes for reinitialization exist (Sussman and Fatemi 1999) but the deformed level set function will behave differently than the original distance function. Furthermore, each reinitialization costs time. Instead, the unit length of the gradient of the level set function everywhere can be included in the level set equation. It may even be used to deform an arbitrary, differentiable function into a distance transform (Li et al. 2005). In their variational formulation of the level set

evolution, the following additional term, properly weighted, was added to the evolution:

$$\zeta(\phi) = \Delta\phi - \operatorname{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right). \tag{9.62}$$

It is minimal if $|\nabla\phi| = 1$ holds everywhere which is only true if ϕ is a signed distance function.

In Li et al. (2011b), this term has been used to let the level set function start from a smooth version of a binary function that has the value $-C$ outside the segment and C inside of it. The level set function then automatically deforms into a signed distance function.

The authors of Li et al. (2005) later extended their approach to achieve a more stable behavior. Instead of requiring a level set function with $|\nabla\phi| = 1$ everywhere, they suggested in Li et al. (2010) a function that has unit length gradients only in the vicinity of the 0-level set and nonzero gradient length elsewhere.

9.2.10 Noise Reduction and Boundary Enhancement

If the noise level is too high compared to the local contrast or if local intensity variation is present but has not been accounted for, the computed object boundary may become too ragged. Edge-preserving smoothing then uses a simple edge model for removing noise and creating smooth boundaries prior to level set segmentation. Examples for such preprocessing are soft tissue segmentation in CT of Suzuki et al. (2010) or kidney segmentation in MRI of Gloger et al. (2015). A more elaborate approach is to use gradient vector flow, described in Sect. 9.1.2, which already contains a smoothing regularizer to generate smooth boundaries at each point in time during level set evolution (Lee et al. 2012).

Successful noise reduction may allow removing the smoothness term from the level set equation (Shyu et al. 2012). This results in a greatly simplified update rule for the level set function as the model term on the boundary length is removed from Eq. (9.58). If the average values c_1 and c_2 in Eq. (9.58) are known and constant, optimization can be done in a single step based on the histogram of data values $I(\mathbf{x})$. The optimal 0-level set would consist of all locations $\mathbf{x}$ with

$$I(\mathbf{x}) = \frac{1}{2}(\alpha_1 c_1 + \alpha_2 c_2) \tag{9.63}$$

and the level set function would be generated from this. If c_1 and c_2 represent average values of the current foreground and background segments, optimization reduces to computation of current averages

$$c_1 = \frac{\int_\Omega H(\phi(\mathbf{x}))I(\mathbf{x})\mathrm{d}\mathbf{x}}{\int_\Omega H(\phi(\mathbf{x}))\mathrm{d}\mathbf{x}} \text{ and } c_2 = \frac{\int_\Omega (1 - H(\phi(\mathbf{x})))I(\mathbf{x})\mathrm{d}\mathbf{x}}{\int_\Omega (1 - H(\phi(\mathbf{x})))\mathrm{d}\mathbf{x}}, \tag{9.64}$$

followed by computing a new 0-level set using (9.63) that is then used to build the new level set function.

However, decoupling the constraint for short boundaries from the remainder of the level set equation will result in a different segmentation. For instance, edge-preserving smoothing may retain small connected components in the foreground or background that would be removed if the short boundary constraint is part of the level set optimization.

9.2.11 Initial Level Sets and Initial Regions of Interest

Many level set formulations do not guarantee that a global minimum is reached and the final result depends on the initialization. This is often intended in medical imaging because it is a simple way to add domain knowledge to the segmentation. The search is now guided by the location, orientation, and possibly even shape of the initial level set and may also be restricted to a region of interest.

Still, interaction may be tedious and it could introduce operator bias. If the object and background have homogeneous and distinct intensities or colors, initial level sets can also be generated from clustering in feature space. An example is Qi et al. (2012) to segment cells in histopathological specimen microscopies. They also include some knowledge about the spatial configuration of the different segments into their method. Stained cells are darker than the background. Cell centers are found by a mean-shift-like procedure where votes are transported along intensity gradients. Subsequent segmentation evolves separate level sets from each center, letting level sets compete for the free space.

Clustering has also been extended to include local smoothness of the intensity function for defining the initial level set function. In Bunyak et al. (2011), the authors separate color images from cell microscopy into several different regions represented by different clusters. The objective function of the fuzzy c-means algorithm (see Sect. 12.7.2) is complemented by a weighted neighborhood term N. It penalizes differences in membership degree u_{ik} to cluster i of pixels p_k in a neighborhood Ω around pixel p_j (m is the fuzzification):

$$N_\Omega(i,j) = \left(u_{ij}\right)^m \exp\left(-\sum_{k\in\Omega}\left(u_{ik}\right)^m\right) \tag{9.65}$$

Three different structures are separated (nucleus, pseudo-lumen, and epithelial cytoplasm) from the background using multi-phase level sets. Initial 0-level sets are given by the cluster boundaries. The images show fairly homogeneous foreground and background regions.

If several structures share their appearance with the object of interest, the clustering methods described above will not allow their unique separation, since 0-level sets will be generated for all objects with the same appearance. The problem

becomes worse when the homogeneity assumption is not true and feature values of different clusters overlap.

A possible solution can be to single out a region of interest (ROI) that removes similarly looking objects from further consideration. An example is the work of Lin et al. (2012) to segment lesions in dental X-rays. First, teeth are found in noise-reduced contrast-enhanced images by thresholding followed by connected component analysis. The level set method is then initialized using trained thresholds on the result. Level set segmentation is needed to introduce an indirect smoothness constraint via the level set equation.

A similar strategy was followed by Gloger et al. (2012) for kidney segmentation. They applied shape constraints on threshold-segmented MR images to find an initial kidney region. Thresholds were determined from training sets.

Extensive preprocessing was used by Kohlberger et al. (2011) to detect multiple objects to segment abdominal organs in MRI. ROI detection is based on trained knowledge about the expected shape and appearance of each organ of interest as well as on the relative positions of each organ with respect to other organs. The initial segmentation defines the number of necessary level sets (one for each organ). The final level set segmentation resolves small inaccuracies such as overlapping boundaries of different segments. The level sets compete for space, as segments should fill the space but not overlap.

9.2.12 Topologically Constrained Level Sets

Compared to parametrically defined active contours presented in Sect. 9.1, level sets have the advantage that their topology may change during evolution. An initial level set may be a simple, closed curve. This curve will split and merge during evolution according to image properties.

This may be unwanted, if the topology of an object is known a priori. Level sets can be adapted to prohibit such change of topology. Han et al. (2003) suggested additional constraints based on digital topology to control the level set evolution. Digital topology defines closeness and connectedness for a discrete sampling of continuous space. A connected component of a discrete scene consists of all scene elements between which a path exists. A path exists between two scene elements s_0 and s_N if a sequence $s_0, s_1, s_2, \ldots, s_N$ exists so that every pair s_i, s_{i+1} is adjacent and a given homogeneity criterion is true for all pairs.

Adjacency depends on the dimension of the scene. Pixels in a 2D scene can be 4-adjacent if they share a common edge, or 8-adjacent if they share a common point. Voxels may be 6-, 18-, or 26-adjacent. Homogeneity in its simplest version requires equal function values but can be any other attribute that is definable on pairs of scene elements. For topologically constrained level sets, a pair of scene elements fulfills the homogeneity criterion, if the pair is completely inside or outside of the boundary defined by the current 0-level set.

When adding or removing a scene element, it is of interest whether this action changes the topology:

Fig. 9.38 Topology-preserving level set evolution stops before the front moves over non-simple points

- Removing a scene element may split a connected component into several components.
- Adding a scene element may merge two or more components.

If a label of a scene element can be switched between foreground and background without altering the topology, it is called a *simple point* (see Fig. 9.38). Whether a point is simple or not changes after each label change and has to be continuously monitored.

The speed term of the evolving level set is now supplemented by a term that prevents a front from moving into non-simple points. The evolving front is mapped on the discrete image after each iteration step. Subsequently, points of the discrete scene are classified into simple and non-simple points. Speed is reduced to zero if a non-simple point is to be passed.

9.3 Concluding Remarks

Active contours fit a parameterizable model of a segment to the data by finding an optimal location for a model instance. Since global and local constraints of the active contour are always asserted, they are particularly well suited if data alone does not provide this information.

The major difference between the two types of active contours is the representation. While an explicit representation is inherently topology-preserving and fast to optimize, implicit representations are defined independent of dimension and operate in continuous space. Preference of one of the two kinds should be given to the methodology that best fits an actual segmentation problem.

From a methodological point of view, implicit formulations are very attractive as the enable describing the wanted segmentation as a property of the image where desired attributes of the segmentation can be added in the model terms. Some examples have been presented in this chapter. Examples to include higher-level knowledge such as shape and appearance will be presented in Chap. 11. A review on including domain knowledge in the level set framework is found in Toennies et al. (2016).

9.4 Exercises

- Sketch a segmentation problem where active contours (defined explicitly or implicitly) would be the appropriate segmentation strategy. Explain the reasons why it would be appropriate.
- What kind of information is represented by the internal energy term of an explicit active contour? What could be possible adverse effects if this term is weighted too strongly in the equation?
- Why is it important to initialize an active contour close to the searched segment boundary?
- How can balloon forces be used for relieving the user from having to place the active contour close to its final position? Discuss the disadvantages of the strategy.
- What are the features that are used to compute the gradient in gradient vector flow?
- What would be image properties that will cause the gradient vector flow to fail leading an active contour to the segment boundary?
- How can active contours be constructed so that topological changes are accommodated?
- How is the current segment boundary represented in the level set framework?
- Explain why it is not important how the level set function looks like.
- What is the difference for the wave propagation if represented by stationary versus dynamic level sets?
- Why is it necessary for dynamic level sets to embed the n-dimensional level set function into $(n + 1)$-dimensional space?
- What is the meaning of the speed function? Explain the three different components of the speed function.
- How is boundary smoothness represented in the level set framework?
- What is meant by an "upwind scheme" and why is it necessary to use an upwind scheme?
- What conditions have to be met for guaranteeing stable computation of the level set evolution?
- Explain why the fast marching algorithm is less time-consuming than naive computation of arrival times for stationary level set.

- What is the difference between the fast marching approach and Dijkstra's shortest path algorithm?
- Why is the narrowband technique faster than regular propagation of a dynamic level set? What are potential disadvantages of this technique?
- Why is using the narrowband technique also an advantage regarding the step size of the time step compared to regular propagation of the level set function?
- What are the properties that determine the size of the band in the narrowband technique?
- What is minimized when optimizing geodesic active contours under the level set framework?
- Explain how region information can be included in level set segmentation for "active contours without edges."
- What are the assumptions that are made so that variational level sets optimize the Mumford–Shah functional?
- Why is it necessary to replace the Heaviside function for variational level sets? What are the desired properties of the function that replaces the Heaviside function?
- Why is it preferable to run the level set evolution on a signed distance function? How can frequent reinitialization of the level set function be avoided?
- Which kind of medical images may have shading artifacts (represented by a bias field)? How can bias field correction be included into the level set evolution?
- Explain why the need to choose a good initialization for the level set evolution can be an advantage.
- How can the level set evolution be modified to constrain topology? Compare this strategy with the introduction of topological variability for explicit active contours. What do the two approaches have in common?

References

Adalsteinsson D, Sethian JA (1995) A fast level set method for propagating interfaces. J Comput Phys 118(2):269–277

Bunyak F, Hafiane A, Palaniappan K (2011) Histopathology tissue segmentation by combining fuzzy clustering with multiphase vector level sets. In: Arabnia HR, Tran Q-N (eds) Software tools and algorithms for biological systems. Springer, Berlin, pp 413–424

Caselles V, Kimmel R, Sapiro G (1997) Geodesic active contours. Int J Comput Vis 21(1):61–79

Chan TF, Vese LA (2001) Active contours without edges. IEEE Trans Image Process 10(2):266–277

Chan TF, Sandberg BY, Vese LA (2000) Active contours without edges for vector-valued images. J Vis Commun Image Represent 11(2):130–141

Cohen LD (1991) On active contour models and balloons. CVGIP Image Underst 53(2):211–218

Cohen LD, Cohen I (1993) Finite-element methods for active contour models and balloons for 2-d and 3-d images. IEEE Trans Pattern Recognit Mach Intell 15(11):1131–1147

Cohen LD, Kimmel R (1997) Global minimum for active contour models: a minimal path approach. Intl J Comput Vis 24(1):57–78

Davatzikos CA, Prince JL (1995) An active contour model for mapping the cortex. IEEE Trans Med Imaging 14(2):65–80

Gloger O, Toennies KD, Liebscher V, Kugelmann B, Laqua R, Völzke H (2012) Prior shape level set segmentation on multistep generated probability maps of MR datasets for fully automatic kidney parenchyma volumetry. IEEE Trans Med Imaging 31(2):312–325

Gloger O, Toennies K, Laqua R, Völzke H (2015) Fully automated renal tissue volumetry in MR volume data using prior-shape-based segmentation in subject-specific probability maps. IEEE Trans Biomed Eng 62(10):2338–2351

Han X, Xu C, Prince JL (2003) A topology preserving level set method for geometric deformable models. IEEE Trans Pattern Anal Mach Intell 25(6):755–768

Honea DM, Snyder WE, Hanson KM (1999) Three-dimensional active surface approach to lymph node segmentation. Proc SPIE 3661(2):1003–1011

Kass M, Witkins A, Terzopoulos D (1988) Snakes: active contour models. Int J Comput Vision 1(4):321–331

Kichenassamy S, Kumar A, Olver P, Tannenbaum A, Yezzi A (1995) Gradient flows and geometric active contour models. In: 5th international conference on computer vision (ICCV'95), pp 810–817

Klann E, Ramlau R, Ring W (2011) A Mumford-Shah level-set approach for the inversion and segmentation of SPECT/CT data. Inverse Prob Imaging 5(1):137–166

Kohlberger T, Sofka M, Zhang J, Birkbeck N, Wetzl J, Kaftan J, Declerck J, Zhou SK (2011) Automatic multi-organ segmentation using learning-based segmentation and level set optimization. In: Medical image computing and computer-assisted intervention–MICCAI 2011, pp 338–345

Kosic N, Weber S, Büchler P, Lutz C, Reimers N, Gonzales Ballester MA, Reyes M (2010) Optimisation of orthopaedic implant design using statistical shape analysis based on level set. Med Image Anal 14(3):265–275

Ladak HM, Milner JS, Steinman DA (2000) Rapid three-dimensional segmentation of the carotid bifurcation from serial MR images. J Biomech Eng 122(1):96–99

Leventon ME, Grimson WEL, Faugeras O (2000) Statistical shape influence in geodesic active contours. In: IEEE computer society conference on computer vision and pattern recognition (CVPR'00), vol 1, pp 1316–1323

Lee M, Cho W, Kim S, Park S, Kim JH (2012) Segmentation of interest region in medical volume images using geometric deformable model. Comput Biol Med 42(5):523–537

Li C, Xu C, Gui C, Fox MD (2005) Level set evolution without re-initialization: a new variational formulation. In: IEEE conference on computer vision and pattern recognition CVPR 2005, vol 1, pp 430–436

Li CM, Kao C, Gore JC, Ding Z (2007) Implicit active contours driven by local binary fitting energy. In: IEEE conference on computer vision and pattern recognition CVPR 2007, pp 1–7

Li C, Xu C, Gui C, Fox MD (2010) Distance regularized level set evolution and its application to image segmentation. IEEE Trans Image Process 19(12):3243–3254

Li CM, Huang R, Gatenby JC, Metaxas DN, Gore JC (2011a) A level set method for image segmentation in the presence on intensity inhomogeneities with applications to MRI. IEEE Trans Image Process 20(7):2007–2016

Li BN, Chui CK, Chang S, Ong SH (2011b) Integrating spatial fuzzy clustering with level set methods for automated medical image segmentation. Comput Biol Med 41:1–10

Lin PL, Huang PY, Huang PW (2012) An automatic lesion detection method for dental X-ray images by segmentation using variational level set. In: International conference on machine learning and cybernetics (ICMLC), vol 5, pp 1821–1825

McInerney T, Terzopoulos D (1995a) A dynamic finite element surface model for segmentation and tracking in multidimensional medical images with application to cardiac 4D image analysis. J Comput Med Imaging Graph 19(1):69–83

McInerney T, Terzopoulos D (1995b) Topologically adaptable snakes. In: International conference on computer vision ICCV, pp 840–845

McInerney T, Terzopoulos D (2000) T-snakes: topologically adaptable snakes. Med Image Anal 4(1):73–91

Mumford D, Shah J (1989) Optimal approximations by piecewise smooth functions and associated variational problems. Commun Pure Appl Math 42(5):577–685

Neuenschwander WM, Fua P, Iverson L, Székely G, Kübler O (1997) Ziplock snakes. Int J Comput Vis 25(3):191–201

Osher S, Sethian J (1988) Fronts propagating with curvature-dependent speed—algorithms based on Hamilton-Jacobi formulations. J Comput Phys 79:12–49

Osher S, Paragios N (2003) Geometric level set methods in imaging, vision, and graphics. Springer, Berlin

Paragios N (2002) Geodesic active regions and level set methods for supervised texture segmentation. Int J Comput Vis 46(3):223–247

Paragios N (2003) A level set approach for shape-driven segmentation and tracking of the left ventricle. IEEE Trans Med Imaging 22(6):773–776

Paragios N, Mellina-Gottardo O, Ramesh V (2004) Gradient vector flow fast geodesic active contours. IEEE Trans Pattern Anal Mach Intell 26(3):402–407

Park J, Keller JM (2001) Snakes on the watershed. IEEE Trans Pattern Anal Mach Intell 23 (10):1201–1205

Park SC, Kim SH, Lee GS, Lee CW, Na IS, Le AVH, Do TY (2011) Flower area segmentation by a level set based on GMM initialization. In: KSII the 3rd international conference on internet (ICONI) 2011, pp 17–21

Qi X, Xing F, Foran DJ, Yang L (2012) Robust segmentation of overlapping cells in histopathology specimens using parallel seed detection and repulsive level set. IEEE Trans Biomed Eng 59(3):754–765

Rousson M, Paragios N (2002) Shape priors for level set representations. In: 7th European conference on computer vision ECCV 2002. LNCS, vol 2351, pp 416–418

Sethian JA (1996) A fast marching level set method for monotonically advancing fronts. Appl Math 93(4):1591–1595

Sethian JA (1998) Level set and fast marching methods. Cambridge University Press, Cambridge

Shyu KK, Pham VT, Tran TT, Lee PL (2012) Global and local fuzzy energy-based active contours for image segmentation. Nonlinear Dyn 67(1559):1578

Sokoll S, Rink K, Toennies KD, Brechmann A (2008) Dynamic segmentation of the cerebral cortex in MR data using implicit active contours. In: 12th annual conference on medical image understanding and analysis MIUA 2008, pp 184–188

Sussman M, Fatemi E (1999) An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. SIAM J Sci Comput 20(4):1165–1191

Suzuki K, Kohlbrenner R, Epstein ML, Obajuluwa AM, Xu J, Hori M (2010) Computer-aided measurement of liver volumes in CT by means of geodesic active contour segmentation coupled with level-set algorithm. Med Phys 37(5):2159.2166

Toennies KD, König T, Gloger O (2016) Domain knowledge for level set segmentation in medical imaging. Biomed Image Segmentation Adv Trends 51–74

Tsai A, Yezzi A, Willsky AS (2001) Curve evolution implementation of the Mumford-Shah functional for image segmentation, denoising, interpolation, and magnification. IEEE Trans Image Process 10(8):1169–1186

Wang XF, Huang DS, Xu H (2010) An efficient local Chan-Vese model for image segmentation. Pattern Recognit 43(3):603–618

Westin CF, Lorigo LM, Faugeras O, Grimson WEL, Dawson S, Norbash A, Kikinis R (2000) Segmentation by adaptive geodesic active contours. In: Medical image computing and computer-assisted intervention—MICCAI 2000. LNCS, vol 1935, pp 266–275

Yazdanpanah A, Hamarneh G, Smith BR, Sarunic MV (2011) Segmentation of intra-retinal layers from optical coherence tomography images using an active contour approach. IEEE Trans Med Imaging 30(2):484–496

Xu X, Prince JL (1998) Snakes, shapes, and gradient vector flow. IEEE Trans Image Process 7(3):359–369

Zhang K, Zhang L, Zhang S (2010a) A variational multiphase level set approach to simultaneous segmentation and bias correction. In: 17th IEEE international conference on image processing ICIP 2010, pp 4105–4108

Zhang K, Song H, Zhang L (2010b) Active contours driven by local image fitting energy. Pattern Recognit 43(4):1199–1206

Registration and Normalization 10

Abstract

Information about an object from different sources can be combined if a transformation allows mapping data from one source to data of the other source. In medical imaging, the two sources are image acquisition systems. If the two sources depict the same subject, this process is called registration. If they depict different subjects, it is called normalization. The mapping is a geometric transformation that accounts for different positioning of a patient in two image acquisition systems. Determining a registration or normalization transformation requires redundant information in the two images, a suitable restriction of acceptable transformations, and, for iterative schemes, a criterion that rates the quality of a given transformation. Various ways to compute a registration or normalization transformation from medical images will be discussed in this chapter.

Concepts, Notions, and Definitions Introduced in this Chapter

- Rigid and non-rigid registration;
- Reference and moving image;
- Normalization;
- Similarity criteria: average distance, Procrustes distance, intensity difference, correlation and covariance, stochastic and deterministic sign change, mutual information;
- Analytic solution for rigid registration;
- Constraints for iterative registration: elasticity, viscosity, splines, similarity of displacement vectors;
- Multi-resolution registration;
- Joint segmentation and registration.

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_10

Fusing information from different sources produces synergy by spatially relating data from the sources with respect to each other. Registration and normalization are the processes to find an appropriate mapping for the fusion.

Registration computes a transformation between two or more pictorial representations of the same subject. The transformation is carried out on one image, the *moving image*, so that it optimally fits to the other image, the *reference image* (sometimes also called target image or fixed image). If the two scenes have the same dimensionality and dimensions correspond—e.g., when comparing a 3d MRI data set with a 3d CT data set (see Fig. 10.1)—topology will often be preserved by the transformation. Hence, adjacent locations prior to registration will remain adjacent. It is a useful constraint for determining the registration transformation.

Normalization is the operation to compare images from two or more different subjects. It differs from registration in that topology is not always preserved. The use of the term normalization refers to the mapping of images (the moving images) of different subjects onto some common norm image (which in this case is the reference image). In the medical sciences, it probably originates from functional imaging where normalization is routinely used for group studies. Normalization also happens when a geometric model of shape and appearance is mapped to an image. This is also called *matching* referring to a use in general image analysis (e.g., template matching). It is used extensively to describe atlas-based segmentation. One of the first publications was R. Bajcy's work on 2d and 3d elastic matching (Bajcy et al. 1983; Bajcy and Kovacic1989).

Registration and normalization are sometimes subsumed under the common notation registration, but different requirements and constraints justify the discrimination by name even though many normalization algorithms borrow from registration. Surveys on registration algorithms in all areas of digital image processing and analysis have been published by Brown (1992) and Zitova and Flusser (2003) (the latter is intended to be an update of the former). The two surveys may also serve as tutorials on basic requirements and characteristics in registration. Characterization of methods used in medical applications can be found in the extensive survey of Maintz and Viergever (1998). A good introduction into numerical issues of registration is Modersitzki (2003). A newer review targeted at medical image applications is Oliveira and Tavares (2014).

Finding the appropriate transformation between two scenes f and g depends on four aspects (Brown 1992):

- The *feature space* determines features F in f and g that will be used for correspondence.
- The *similarity criterion* S defines what is meant by correspondence between two registered scenes.
- The *search space* contains all possible transformations T among which an optimal transformation is searched.
- The *search strategy* tells how—given some transformation producing some similarity—the next transformation is found.

Fig. 10.1 Registration finds a transformation to map two different images of the same patient onto each other

Finding correspondence between two scenes f and g requires at least some redundancy between image features $F(f)$ and $F(g)$ so that a sufficiently large number of locations $\mathbf{x}_1$ and $\mathbf{x}_2$ in the two images exist, for which local feature similarity $s(F(f(\mathbf{x}_1)), F(g(T(\mathbf{x}_2)))$ can be defined (see Fig. 10.2). In order to make the search for the correct registration function a minimization task, the similarity measure is often defined such that it is low when feature locations correspond given some transformation T.

In this chapter, we will discuss techniques and their limitations to solve the problems based on registration. The concluding sections will address differences of normalization and matching methods with respect to registration.

10.1 Feature Space and Correspondence Criterion

Establishingcorrespondences between reference image f and moving image g requires finding a measure by which equivalence between locations in the two scenes is quantified. Two points in the two images are assumed to be equivalent if they have the same location with respect to some unknown patient coordinate system. The relation between the known scanner coordinate system and the patient coordinate system is influenced

- by different positioning of the patient in the scanner.
- by distortions caused by the imaging system.
- by any movements of organs due to physiology.
- by changes caused from interventions taking place between creating f and g.

Equivalent locations in the two images are assumed to have similar appearance. This does not imply that unlike locations have different appearances. Correspondence criteria will often only apply to a subset of locations in f and g where similarity either uniquely identifies pairs of correspondent points in patient coordinates (this is the case for labeled landmarks) or at least the number of possible correspondence pairs is reduced (this is the case when, e.g., landmark curves are used).

Fig. 10.2 For finding the registration transformation, redundant information must exist that defines the mapping at least at some locations (such as the landmark positions indicated in this example)

Features commonly used for measuring correspondence in medical image registration are as follows:

- Extrinsic markers.
- Intrinsic features: landmark points, curves, or regions.
- Voxel intensities or gradients.

Extrinsic or intrinsic features are *model-based features* (see Fig. 10.3), whereas voxel intensities or gradients are *image-based features* (see Fig. 10.4). Model-based features are selected intentionally according to the assumption that they provide essential information for computing the registration transformation. Using image-based features assumes that low-level operations on an image suffice for producing the necessary correspondence information. The number of locations $\mathbf{x}$ for which correspondence may be established—i.e. the domain for which local correspondence c is defined—is generally smaller in model-based registration than in image-based registration, but their accuracy is often better.

10.1.1 Correspondence Criteria for Extrinsic and Intrinsic Markers

Extrinsic and intrinsic markers are assumed to be describable by a finite set of feature points which are distinguishable in the two scenes. Local correspondence may be computed only for feature points in the two scenes. A feature function assigns the label 1 to feature points and the label 0 to all other locations in the image. Feature similarity c_c simply notes the incidence of coinciding feature points with the same label

Fig. 10.3 Examples for intrinsic and extrinsic markers. Markers do not have to be points, but the same structure should be visible in two scenes to be registered

Fig. 10.4 Image-based features are generated by some filtering process from the image. Since the two images to be registered are usually not equal, image-based features do not necessarily have direct correspondence. It is assumed, however, that most features are visible in both scenes

$$c_c[F(f(\mathbf{x}_1)), F(g(\mathbf{x}_2)), T] = \begin{cases} 0, & \text{if } F(f(\mathbf{x}_1)) = F(g(\mathbf{x}_2)) > 0 \text{ and } \mathbf{x}_1 = T(\mathbf{x}_2) \\ 1, & \text{otherwise.} \end{cases} \tag{10.1}$$

The similarity S between scenes f and g after transforming g by some transformation T would be the sum of local correspondences

$$S_c(F,f,g,\mathbf{x}_1,\mathbf{x}_2,T) = \frac{1}{|M|}\sum_{(\mathbf{x}_1,\mathbf{x}_2)\in M} c_c[F(f(\mathbf{x}_1)),F(g(\mathbf{x}_2)),T], \tag{10.2}$$

where M is the set of feature point pairs in f and g with $\mathbf{x}_1 = T(\mathbf{x}_2)$, and $|M|$ is the number of elements of M. Normalization by $|M|$ makes transformations T with different numbers of corresponding markers comparable to each other. Removing the normalization constant may make sense if transformations with few matching feature points shall be penalized.

The mapping is not unique, i.e., more than one transformation T may exist that minimizes S. A unique mapping is achieved if F assigns a unique nonzero label to every feature point such that corresponding features have equal labels.

The similarity criterion above counts the number of matched feature points. Its underlying assumption is that computation of feature points is exact and a suitable transformation exists with $\mathbf{x}_1 = T(\mathbf{x}_2)$ for all feature point pairs $(\mathbf{x}_1, \mathbf{x}_2)$ of the registration transformation T.

It is possible and even likely that this is not true because marker positions have not been determined accurately or they do not match exactly. Local correspondence can be relaxed by introducing some minimum distance ε that must not be exceeded for declaring two feature point positions to be equal:

$$c_\varepsilon[F(f(\mathbf{x}_1)),F(g(\mathbf{x}_2)),T] = \begin{cases} 0, & \text{if } F(f(\mathbf{x}_1)) = F(g(T(\mathbf{x}_2))) > 0 \wedge |\mathbf{x}_1 - T(\mathbf{x}_2)| < \varepsilon \\ 1, & \text{otherwise.} \end{cases} \tag{10.3}$$

Using c_ε still counts the number of correspondences between feature points. The similarity criterion for a transformation T is

$$S_\varepsilon(F,f,g,\mathbf{x}_1,\mathbf{x}_2,T) = \frac{1}{|M|}\sum_{M=\{(\mathbf{x}_1,\mathbf{x}_2)|\mathbf{x}_1=CP(T(\mathbf{x}_2))\}} c_\varepsilon[F(f(\mathbf{x}_1)),F(g(\mathbf{x}_2)),T], \tag{10.4}$$

where $CP()$ ensures that $T(\mathbf{x}_2)$ is the closest of all marker points to $\mathbf{x}_1$. This is necessary for ensuring that any feature point in f may only be mapped to exactly one feature point in g.

A unique relationship between feature points in the two scenes may not exist so that counting correspondences are not suitable. If, for instance, feature points in the two images describe the same structure, e.g., the skin surface of the head, but no specifically labeled positions on the skin surface are identified, similarity must attempt to minimize the average distance between two sets of feature points. A suitable definition for local correspondence between feature points of f and g could be the distance between two matching feature points $\mathbf{x}_1$ and $\mathbf{x}_2$ such that the transformed $\mathbf{x}_2$ is closest to $\mathbf{x}_1$:

Fig. 10.5 The Procrustes distance between two sets of points is the distance from points of one of the sets to closest counterparts in the other set

$$c_d[F(f(\mathbf{x}_1)), F(g(\mathbf{x}_2)), T] = \begin{cases} \|\mathbf{x}_1 - T(\mathbf{x}_2)\|, & \text{if } F(f(\mathbf{x}_1)) > 0 \wedge F(g(T(\mathbf{x}_2))) > 0 \\ & \wedge\ \mathbf{x}_1 = CP(T(\mathbf{x}_2)) \\ 1, & \text{otherwise.} \end{cases} \tag{10.5}$$

The registration similarity S_d is computed similarly than S_e:

$$S_d(F, f, g, \mathbf{x}_1, \mathbf{x}_2, T) = \frac{1}{|M|} \sqrt{\sum_{M=\{(\mathbf{x}_1,\mathbf{x}_2)|\mathbf{x}_1=CP(T(\mathbf{x}_2))\}} c_d[F(f(\mathbf{x}_1)), F(g(\mathbf{x}_2)), T]} \tag{10.6}$$

S_d is called the *Procrustes distance*[1] (see Fig. 10.5). Local similarity using the Procrustes distance between markers can also be used for matching markers by requiring $F(f(\mathbf{x}_1)) = F(g(T(\mathbf{x}_2))) > 0$ instead of $F(f(\mathbf{x}_1)) > 0 \wedge F(g(T(\mathbf{x}_2))) > 0$.

Similarity can be computed for feature points from extrinsic and intrinsic markers. *Extrinsic markers* are artificially attached markers to the body such as skin markers or markers on a stereotactic frame. Marker materials must be clearly visible in the two images. Appropriate materials depend on the imaging technique. Plastic, for instance, is barely visible in CT and not visible at all in MR. Any material visible that does not emit photons will not be detected by nuclear imaging techniques.

10.1.2 Requirements for Extrinsic and Intrinsic Markers

Markers should be small for simplifying computation of the feature point location from a segmented marker region. They should be opaque with respect to the imaging technique and produce a high-contrast signal. The accuracy with which

[1]Sometimes, the square root is not taken.

Fig. 10.6 Minimizing the Procrustes distance for a set of points produces the optimal transformation only at these points. Transformations at other locations can be arbitrary unless the set of permissible transformations is suitably restricted. Any of the three regions depicted on the right optimize the Procrustes distance in the same way

feature point locations can be determined varies with the type of marker and the noise level and different spatial resolution of the imaging device. Hence, computation of a registration transformation from feature point locations may be inaccurate.

A registration transformation based on matching feature point positions is valid only at these positions (see Fig. 10.6). Registration of other positions in the scene requires the space of acceptable transformations to be suitably constrained so that matching feature point positions uniquely determine a transformation for the complete image. Even for finding a rigid transformation, this can be difficult because the relation between extrinsic markers and the patient may be non-rigid. Skin markers, for instance, may move in some non-rigid fashion with respect to the bone.

Intrinsic landmarks are feature points of anatomical or geometrical nature. They are identifiable locations in the scenes to be registered and may be used in the same fashion than extrinsic markers. *Anatomical landmarks* are locations used in anatomy for providing spatial reference information. Examples are the anterior and posterior commissura (AC and PC) used for inter-subject normalization of human brains (see Fig. 10.7). Anatomic landmarks are often identified interactively. An expert user will know their definition and how to find them in the images. The name of an anatomic landmark may serve as a unique label for the feature point.

Geometrical landmarks are feature points that can be found geometrically but do not necessarily have an anatomic label (see Fig. 10.7). An example would be the set of local curvature maxima on some defined surface. The landmarks can be computed automatically if their geometric definition is computable. Geometrical landmarks are generally not unique, as the geometric definition of a landmark is non-unique (e.g., there will be many local curvature maxima). However, using geometric features, such as SIFT, SURF, MSER, or local shape context, presented in Chap. 5 provides a rich description that allows substantial reduction of potential counterpart of feature locations in the other image.

Fig. 10.7 Anatomic landmarks such as the anterior and posterior commissural (ac, pc) or geometric landmarks such as locations of high curvature on object boundaries can be used for registration

Still, noise, different spatial resolution between image scenes, and other artifacts, can cause an identifiable geometric landmark in one scene not to be detected in the other scene.

Computing a registration transformation from anatomic or geometric landmarks requires similar constraints than the use of extrinsic markers. The spatial relationship between landmarks and the scenes to be registered need to be known and should be simple for inferring a scene registration transformation from landmark registration. If, for instance, the vertebra in a preoperative CT scan shall be registered with an intra-operative scan, the transformation will be rigid. If, however, some of the landmarks were on other vertebrae, the relation between the object of interest and the landmarks may no longer be rigid and too complex for being accounted for.

Inaccurate interactive depiction or automatic computation of anatomical or geometrical landmarks results in displacement errors, which affects the accuracy of the transformation between landmark positions.

Geometrical landmarks may also be curves or regions, e.g., the ridges on some surface or the surface itself (see Fig. 10.8). The mapping will be non-unique because there is no unique identification of locations within such curves or regions. Using curve or region points requires a similarity criterion such as S_d, which does not require point-to-point correspondence. The criterion needs to be adapted so that points in one scene that do not have a counterpart are not included into the similarity measure. This can be done by using the ε of the c_ε measure to exclude feature points for which $|\mathbf{x}_1 - CP(T(\mathbf{x}_2))| > \varepsilon$.

Intrinsic features define the mapping for a selected subset of locations in the image similar to extrinsic features. Computing the scene registration transformation again requires a suitably constrained transformation space. However, registering two scenes does not always require equal accuracy everywhere in the scene. Often,

Fig. 10.8 If ridges are used as landmarks, the distance between two ridges is an average of distances of points on one ridge to closest points on the other ridge

some region of interest needs to be analyzed. Intrinsic landmarks are then placed closely to this region which reduces adverse influences from placement inaccuracies there.

10.1.3 Similarity for Image-Based Features

If the number of landmark locations is small, if their localization is unreliable, or if the search space for a transformation is complex, such as in non-rigid registration, landmark-based registration may not be suitable. *Image-based registration features* can provide a denser map of corresponding locations $\mathbf{x}$. Features do not require prior identification. They are computed using low-level image processing techniques. Features may not be reliable everywhere in the scene, but on average the number of reliable feature locations is larger than in landmark-based feature computation. Image-based features are not unique, though, because values of low-level features cannot serve as unique label.

In the simplest case, intensity may serve as feature. Using intensities requires equal objects having equal brightness in the two images. An example is the registration of two images from the same patient made at different times with the same modality under the same protocol. The local correspondence criterion $c_{\text{intensity}}$ is the intensity difference between voxel locations $\mathbf{x}$

$$c_{\text{intensity}}(\mathbf{x}, f, g; T) = \|f(\mathbf{x}) - T \circ g(\mathbf{x})\| \tag{10.7}$$

$T \circ g(\mathbf{x})$ indicates that function g was transformed by T prior to computing the difference. Similarity of two registered scenes is then the sum of all local correspondences. Using the intensity difference may be hampered by noise, especially if the contrast in the image is low.

Computing sign changes instead is a more stable criterion (see Fig. 10.9). The criterion was introduced by Venot and Leclerc (1984) and has been used for 2d image registration [a newer application is Maglogiannis (2004)]. The two images $f(\mathbf{x})$ and $T \circ g(\mathbf{x})$ are subtracted from each other to yield $d(\mathbf{x}) = f(\mathbf{x}) - T \circ g(\mathbf{x})$.

If images are in perfect registration and contain zero-mean noise, the number of sign changes between neighboring pixels in d depends only on the noise characteristics. The number of sign changes decreases if images are misregistered. A similarity criterion $S_{\text{sign_change}}$ to be minimized that is based on sign changes is

$$S_{\text{sign_change}}(T) = -\sum_{\mathbf{x}} \sum_{\mathbf{xn} \in Nbs(\mathbf{x})} |\text{sgn}(\text{d}(\mathbf{x}) - \text{d}(\mathbf{xn}))|, \tag{10.8}$$

where sgn() is the signum function, and $Nbs(\mathbf{x})$ contains the neighbors of $\mathbf{x}$. The authors called the criterion *stochastic sign change criterion.* It requires the images to contain noise. Its success is based on the assumption that the noise level is lower than the signal and that spatial frequency of noise is higher than that of the signal. High-frequency details of objects will not contribute to the registration success but lead to its deterioration. If images are noise-free, a synthetic noise-like pattern may be added to both images prior to their subtraction. This is called *deterministic sign change criterion*.

If voxel intensities for the same object are different in the two scenes f and g, and if no simple mapping for making them equal exists, the measures above cannot be used. However, if at least some of the objects are visible in the two images, commonness between features still relates to the boundary between objects. This can be exploited in several ways.

A straightforward strategy is to use the intensity gradient as feature. Feature computation requires an extra segmentation step for identifying boundary pixels by means of the gradient properties (see Fig. 10.10). Boundary segments are treated as landmark curves or landmark regions for which the average distance between

Fig. 10.9 Sign changes increase if two images (in this case 1d functions) are in registration in (**a**) compared to the non-registered images in (**b**)

Fig. 10.10 Gradient information from the images can be exploited for segmentation if an additional edge detection step uses gradient strength for finding edges (in the case above by using the Canny edge detector)

closest points is to be minimized. Gradients, however, are sensitive to noise and artifacts. Artifact edges may erroneously be treated as boundary edges, localization of edges may contain displacement errors, and boundary edges may not be found in both images.

Gradient information is employed indirectly if a multiple threshold segmentation of the two images is created, which is then registered using the deterministic sign criterion. Multiple thresholds produce segments in which some of the boundaries should coincide with object boundaries. Summing sign changes over the image reduces the influence of noise and artifacts from segmentation boundaries that are not object boundaries.

If segmentation is not suitable as preprocessing step, the influence of noise at edges can be reduced by employing a better edge model. Assuming ideal step edges, the *correlation coefficient* in a δ-neighborhood $N_\delta(\mathbf{x})$ around some location $\mathbf{x}$ can be used as local correspondence criterion c_{edge}:

$$\begin{aligned} c_{\text{edge}}(\mathbf{x}, f, g; T) &= cc_\delta(f(x), T \circ g(x)) \\ &= \frac{\frac{1}{|N_\delta(\mathbf{x})|} \sum_{\mathbf{x}_i \in N_\delta(\mathbf{x})} (f(\mathbf{x}_i) - \bar{f}(\mathbf{x}))(T \circ g(\mathbf{x}_i) - \bar{g}(\mathbf{x}_i))}{\sum_{\mathbf{x}_i \in N_\delta(\mathbf{x})} (f(\mathbf{x}_i) - \bar{f}(\mathbf{x}))^2 \sum_{\mathbf{x}_i \in N_\delta(\mathbf{x})} (T \circ g(\mathbf{x}_i) - \bar{g}(\mathbf{x}_i))^2}, \end{aligned} \tag{10.9}$$

where $\bar{f}(\mathbf{x})$ and $\bar{g}(\mathbf{x})$ are estimates of the expected value in f and g in $N_\delta(\mathbf{x})$. If contrast in the two images to be registered may be reversed, i.e., a dark object in f may be bright in g, the absolute value of c_{edge} is taken. Scene similarity S_{edge} is computed by summing up voxel correspondences at all boundary locations.

If explicit edge detection has not been carried out, scene similarity may be computed by summing over all voxels in the scene. It reduces sensibility to registration errors since correlation between two voxels with constant brightness in their neighborhood is always maximal. The problem can be remedied to some extent by weighting local similarity with the negative gradient length.

Using local correlation takes advantage of the fact that the local neighborhood parallel to an edge should be smooth while this is not true for noise. Another strategy to make use of this is would be to use low-pass filtering before computing the gradient. This has been done, for instance, by Bajcy et al. (1983), but it is sensitive to the gradient strength at an edge. Solutions are preferred where edges of equal strength are mapped onto each other requiring contrast in f and g to be similar.

An alternative for using the correlation that is not sensitive to contrast differences between f and g is the computation of the *mutual information* between two scenes. Similar to correlation, mutual information measures brightness correspondence in the two images. It is high when brightness changes in the two registered images coincide. The measure of coincidence is less restrictive than correlation, as it does not require the pixel-wise linear relationship to be constant for every intensity level.

Mutual information $I(f, g)$ of a pair of images f and g is related to their *joint entropy* $E(f, g)$ in the following way

$$I(f,g) = E(f) + E(g) - E(f,g), \tag{10.10}$$

where $E(f)$ and $E(g)$ are the entropies of f and g as defined in Sect. 4.1.2. The joint entropy $E(f, g)$ is given by

$$E(f,g) = \sum_{i=0}^{N-1} \sum_{j}^{N-1} p(i,j) \log_2 p(i,j). \tag{10.11}$$

The quantity $p(i, j)$ is the joint probability of intensities i and j occurring at the same location $\mathbf{x}$ in f and g. It is estimated from the normalized joint histogram of f and g. The intensity of the two images is assumed to range from 0 to $N - 1$. Mutual information measures how good intensity values in image g at some location $\mathbf{x}$ may be predicted by intensity values in image f at the same location. If the two images are in perfect registration, this predictability is at its maximum if at least some of the information in f is also depicted in g (see Fig. 10.11 for an example for translation and Fig. 10.12 for an example for rotation).

Mutual information can be used as a local correspondence criterion at any location $\mathbf{x}$ as well by computing it from some predefined neighborhood region around $\mathbf{x}$. Small local neighborhoods, however, may produce unreliable estimates of I from the underlying estimates of the probability distributions p.

If applied as global estimate, the different domains of the two images have to be considered. If transformation T is applied to g, some locations $\mathbf{x}$ for which f is defined may become undefined for the transformed g and vice versa. Since the region in $\mathbf{x}$ for which f and g are defined changes with T, the similarity measure

Fig. 10.11 Mutual information for differently shifted MR images (proton density and T2-weighted). Although the two images are quite dissimilar, mutual information shows a pronounced minimum if registration is perfect

Fig. 10.12 Mutual information for differently rotated versions of the images from Fig. 10.11. Again, there is a pronounced maximum if registration is perfect

needs to be adapted in order to account for the different number of locations from which the underlying joint probabilities are estimated.

Mutual information has been used in numerous registration schemes (for a survey see Pluim et al. 2003). Its great popularity is probably due to its ability to include all image information into the similarity and its simplicity of use. Furthermore, an extensive investigation about the accuracy of registration methods has found mutual information to deliver the best results (West et al. 1997; Fitzpatrick et al. 1998). Since the quality of the measure deteriorates with increasing noise, images are often preprocessed for noise removal.

10.2 Rigid Registration

Finding a registration transformation means to minimize the similarity criterion. The registration problem has a number of properties that have to be accounted for:

- In most cases, the similarity measure is an estimate based on data with noise and artifacts and a model that overly simplifies reality.
- An infinitely large number of transformations exist, but only a finite number of information pieces constrain the transformation.
- The optimization functions listed in the previous chapter have an image-dependent number of local minima.

Rigid registration extremely reduces the search space by assuming that differences between patient and scanner coordinates can be represented by rotation and translation only. A number of cases exist, notably registration of cranial 3d images, where the difference between the two scenes to be registered can be represented by a rigid transformation.

10.2.1 Registration Using Paired Landmarks

Rigid registration models patient geometry as a rigid body and accounts for different positioning of the patient in the scanner (see Fig. 10.13). Comparing CT, MR and PET images of the head, which may be treated as a rigid body, were among the first applications for rigid registration. Registration of other bone structures such as the vertebrae may be represented by rigid registration as well.

Registration usually does not account for scaling (which happens when transforming from scanner coordinates to image coordinates), as most 3d imaging devices provide information about the size of a voxel.

The unknown transformation can be described by

Fig. 10.13 Rigid registration accounts for a different position of the patient in the scanner, assuming that there exists a patient coordinate system in which semantically equivalent points in the two scenes to be registered have the same coordinates. Transformation from scanner coordinates in image coordinates involves scaling and potentially mirroring, both of which are usually known before

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \mathbf{R} \times \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \mathbf{t} \quad \text{with } \mathbf{R} = \begin{pmatrix} r_{11} & r_{21} & r_{31} \\ r_{12} & r_{22} & r_{32} \\ r_{13} & r_{23} & r_{33} \end{pmatrix} \quad \text{and} \quad \mathbf{t} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix}, \tag{10.12}$$

where $\mathbf{R}$ is a rotation matrix and $\mathbf{t}$ a translation vector. Solving the task amounts to solving linear equations with a total of 12 unknowns (9 for the rotation and 3 for the translation). A straightforward solution would be to determine pairs of four extrinsic or intrinsic markers $\{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4\}$ and $\{\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{q}_4\}$ such that $\mathbf{p}_i$ should map onto $\mathbf{q}_i$ for $i = 1, \ldots, 4$ (see Fig. 10.14). Using the coincidence correspondence criterion $c_{c,}$ the registration transformation can be found by equating corresponding landmarks via the transformation in Eq. (10.12) and solving for the unknowns.

The result would be unsatisfactory because landmark positions will not be exact. A better solution would be to require a larger number of corresponding landmarks and solving the overdetermined equation system using a minimum norm strategy. It corresponds to using average minimum distance similarity S_d.

This would still not be satisfactory because the computed matrix $\mathbf{R}$ may not be a rotation matrix introducing unwanted shear and scaling into the registration. The nine coefficients of the rotation matrix are not independent of each other, as a rotation has only three degrees of freedom: two angles for describing an axis around which rotation takes place and the rotation angle around this axis. A matrix is a rotation matrix if it is orthonormal with a determinant of 1.

A closed-form solution for a large number of correspondence pairs of landmarks that ensures orthonormality of $\mathbf{R}$ has been presented by Arun et al. (1987). Two sets $\mathbf{p}_i$, and $\mathbf{q}_i$, $i = 1, N$ of corresponding pairs of points—called *paired landmarks*—shall be registered with respect to each other using rotation and translation. The

Fig. 10.14 The naive registration algorithm would compute the transformation by solving a linear equation system from coordinates (x, y, z) from four pairs of landmarks ($\mathbf{p}$, $\mathbf{q}$)

Procrustes distance of the similarity criterion S_d is to be used for minimizing the average distance between corresponding locations after registration.

In the first step, the translation vector is found by computing expected values for the two sets of points

$$\bar{\mathbf{p}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{p}_i, \quad \bar{\mathbf{q}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{q}_i \tag{10.13}$$

The translation is accounted for by forming new point sets $\mathbf{p}'$ and $\mathbf{q}'$ with $\mathbf{p}' = \mathbf{p} - \bar{\mathbf{p}}$ and $\mathbf{q}' = \mathbf{q} - \bar{\mathbf{q}}$.

Finding the rotation $\mathbf{R}$ between $\mathbf{p}'$ and $\mathbf{q}'$ that minimizes the average distance between registered landmarks is then a minimization task:

$$\mathbf{R}_{\text{opt}} = \arg\min_{\mathbf{R}} \sum_{i=1}^{N} \left\|\mathbf{p}_i' - \mathbf{q}_i' \cdot \mathbf{R}\right\|^2 = \sum_{i=1}^{n} \mathbf{p}_i'^{\mathrm{T}} \cdot \mathbf{p}_i' + \mathbf{q}_i'^{\mathrm{T}} \cdot \mathbf{q}_i' - 2 \cdot \left(\mathbf{p}_i'^{\mathrm{T}} \cdot \mathbf{R} \cdot \mathbf{q}_i'\right). \tag{10.14}$$

The term $\sum_{i=1}^{N} \left\|\mathbf{p}_i' - \mathbf{q}_i' \cdot \mathbf{R}\right\|^2$ is sometimes called *fiducial registration error*. The landmarks to be matched are called *fiducial markers*.

Minimizing (10.14) means maximizing $\sum_{i=1}^{N} \mathbf{p}_i'^{\mathrm{T}} \cdot \mathbf{R} \cdot \mathbf{q}_i'$, since the other terms do not depend on $\mathbf{R}$. Arun showed that this rotation matrix can be found from a singular value decomposition of $\mathbf{D} = \mathbf{U}\Lambda\mathbf{V}^{\mathrm{T}} = \sum_{i=1}^{N} \mathbf{p}_i' \cdot \mathbf{q}_i'^{\mathrm{T}}$. The rotation matrix is $\mathbf{R} = \mathbf{V}\mathbf{U}^{\mathrm{T}}$. The matrix is orthonormal but may have a determinant with value -1 in which case the transformation is a rotation that is mirrored at the origin. This should be checked because it indicates severe displacement errors of landmarks or a distribution of landmarks that allow for several different mappings with similar similarity values.

Horn (1987) presented another closed-form solution for registering point sets **p** and **q** using *unit quaternions* for representing the rotation. A *quaternion* has one real and three imaginary parts and is denoted by $q = r + a \cdot i + b \cdot j + c \cdot k$ with i, j, k being the three imaginary parts. Quaternions may also be written as vectors $\mathbf{q} = (r\ a\ b\ c)$. Unit quaternions, i.e., quaternions $\mathbf{q}_r$ with $\|\mathbf{q}_r\| = 1$, have only three independent variables and can be used for representing rotations in 3d space. A 3d rotation of a vector $(x_1\ x_2\ x_3)$ in quaternion algebra requires the vector to be represented as quaternion $(0\ x_1\ x_2\ x_3)$. Rotation around a vector **r** with angle α is then

$$\begin{pmatrix} 0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} \cdot \begin{pmatrix} \cos\ \alpha/2 \\ r_1 \sin\ \alpha/2 \\ r_2 \sin\ \alpha/2 \\ r_3 \sin \alpha/2 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ -x_1 \\ -x_2 \\ -x_3 \end{pmatrix}. \tag{10.15}$$

For given sets of points $\mathbf{p}'$ and $\mathbf{q}'$ that are corrected for translation and scaling, Horn showed that the optimal rotation minimizing the average displacement error has components of the eigenvector corresponding to the largest eigenvalue of a 4×4 matrix **Q** that is computed as follows

$$\mathbf{Q}(\mathbf{S}_{pq}) = \begin{pmatrix} tr(\mathbf{S}_{pq}) & \Delta_{pq}^{\mathrm{T}} \\ \Delta_{pq} & \mathbf{S}_{pq} + \mathbf{S}_{pq}^{\mathrm{T}} - tr(\mathbf{S}_{pq}) \cdot \mathbf{I}_{3\times 3} \end{pmatrix}, \tag{10.16}$$

where

$$\mathbf{S}_{pq} = \begin{pmatrix} s_{11} & s_{21} & s_{31} \\ s_{12} & s_{22} & s_{32} \\ s_{13} & s_{23} & s_{33} \end{pmatrix} = \sum_{i=1}^{n} \mathbf{p}_i' \cdot \mathbf{q}_i'^{\mathrm{T}} \quad \text{and} \quad \Delta_{pq} = (\, s_{23} \quad s_{31} \quad s_{12}\,)^{\mathrm{T}}. \tag{10.17}$$

and $tr()$ is the trace of a matrix. $\mathbf{I}_{3\times 3}$ is a 3×3 identity matrix.

The use of quaternions avoids the ambiguity of orthonormal rotation matrices that may or may not include a mirroring at the origin. It has the disadvantage that it can only be applied for 3d scenes.

10.2.2 Registration with Unpaired Landmarks

The methods of Arun and Horn require paired landmarks. If this cannot be guaranteed, another strategy using principal component analysis (PCA, see Sect. 14.3) can be used for producing average minimum distance between registered landmarks (Toennies et al. 1990). The idea is quite simple. If landmarks represent a structure of a suitably irregular shape, the origin and principal axes of the landmark distribution represent a unique patient-specific coordinate system with respect to some

Fig. 10.15 Principal axes generated from landmarks can act as a patient coordinate system, if the landmark distribution is sufficiently anisotropic. Otherwise, the axes orientation may depend on small changes of landmark positions (which may be caused from localization errors)

common world coordinate system (in this case, the scanner coordinate system, see Fig. 10.15). The registration transform is given by first translating the landmarks into the origin of the world coordinate system and then rotating them according to the principal axes.

The translation between the two systems is given as in the previously described method of Arun et al. (1987). Given a matrix of eigenvectors $\mathbf{E}_p$ and $\mathbf{E}_q$ from the PCA of the two point sets $\mathbf{p}$ and $\mathbf{q}$, the rotation is then

$$\mathbf{R}_{\text{opt}} = \mathbf{E}_p \mathbf{E}_q, \tag{10.18}$$

assuming that eigenvalues in $\mathbf{\Lambda}_p$ and $\mathbf{\Lambda}_q$ correspond. This should be the case if the point distributions of $\mathbf{p}$ and $\mathbf{q}$ represent the same shape.

Using the PCA, decomposition does not require paired landmarks, which makes it also applicable in cases where landmark curves or landmark surfaces shall be registered. It comes at a price, however. PCA-based registration will fail if the shape of the point distribution is approximately spherical. In such case, all three eigenvalues will be approximately equal and orientation from eigenvectors will mainly be governed by noise and artifacts.

The strategies described above rely on two sets of corresponding points among which a transformation shall be found. If the number of points is small, noise and outliers may affect the result. If paired landmarks are to be registered, the influence of noise may be estimated by computing the average and maximum distances between corresponding points of the two data sets after registration. If the PCA is used, noise influence can be estimated from differences of corresponding eigenvalues.

10.2.3 Outlier Detection

Outliers, i.e., landmarks with grossly wrong locations, should also be accounted for. They may remain undetected in automatic landmark computation as their detection

Fig. 10.16 Potential outliers can be detected by a variant of the leaving-one-out technique. The predicted position of the left-out landmark with respect to all other landmarks is measured and compared with the true position. If the difference is too large, the landmark pair is marked as potential outliers

requires the two scenes to be registered. Outliers may be found by comparing distances between registered landmark pairs. The distance between corresponding landmarks, of which one is an outlier, should substantially vary from the distance between other pairs of landmarks. Outlier pairs should be removed.

Outlier detection for paired landmarks can also be done using a variant of the leaving-one-out technique in classification. A pair of landmarks is removed from the two landmark distributions, and a registration transformation is computed without it. The resulting transformation is applied to the removed pair of landmarks, and the distance between the two landmarks after registration is measured (see Fig. 10.16). It is marked for removal if the distance exceeds some threshold. After repeating this procedure for all pairs, those marked for removal are eliminated.

Outlier detection for unpaired landmarks can also be done using the leaving-one-out technique. In this case, single landmarks are removed before registration and the registration transformation is applied to this landmark. Its distance to the closest point in the other set of landmarks is computed. The landmark is marked as outlier, if the distance exceeds some maximum distance. The process is less reliable if distances between adjacent landmarks within a set of landmarks vary widely.

Outlier detection and removal should be iterated until no further outliers are detected or the number of landmarks is too small for a reliable registration.

10.2.4 Registration in the Frequency Domain

If it is known that objects in the two scenes f and g to be registered have the same intensity, e.g., when images from the same scanner taken at different times shall be registered, rigid registration for 2d scenes may also be carried out in the frequency domain (Reddy and Chatterji 1996). Translating an object changes the phase but not the spectrum (amplitude) of the signal in the frequency domain. Rotation around the origin in the spatial domain corresponds to a rotation in the frequency domain (see Fig. 10.17). 2d registration in the frequency space thus first estimates the rotation component. The signal is transferred into polar coordinates. Rotation in Cartesian

Fig. 10.17 The image on the *right* has been rotated by 20° and translated by 20 pixels in *x*-direction. Only the rotation shows up in the spectrum in frequency space

coordinates around the origin corresponds to a translation along the angular axis in polar coordinates. The rotation angle is found by 1d cross-correlation along this axis. If objects have different intensities in the two images but are visible in both, the approach also works if the gradient length image instead of the intensity image is transformed.

Computing the translation is then done either by comparing phases after rotationally registering the images in the frequency domain or by just computing displacements of intensity or gradient length in the spatial domain.

10.2.5 The Head-Hat Registration

The previous methods solved the registration problem in one step. They do provide an optimal solution based on the landmarks and within the model that they use. Pelizarri's registration method for registering CT, MRI, and PET images is an alternative which registers surfaces with respect to each other based on point distances (Pelizarri et al. 1989). Compared with registering patient-specific coordinate systems from principal axes of point distributions, the method has the advantage that local point distances directly influence the result. The method requires segmentation of a surface that is visible in scenes to be registered and which is related to the objects to be registered in a rigid fashion.

In its original version, the method was applied to register cranial images using the skin surface of the head. The method is also known as *head-hat registration*. The skin of the two scenes f and g to be registered is segmented using thresholding. The skin surface of the first scene is represented by a sequence of boundary curves in slices. It is taken as "head" of the registration algorithm. The skin of the other surface (the "hat") is represented by a dense map of surface points to be fitted to the head.

Fitting is done iteratively using Powell's method (Press et al. 1992). Given some initial registration transformation, a new transformation is found by iteratively minimizing the distances between surface points of g to the surface f. Distances are computed by intersecting rays from points in g to the center of inertia of the surface f with f.

Powell's method is known to get stuck in local minima. Several strategies can be used to overcome this. A simple solution would be to provide a rough registration interactively. This can be done by interactively aligning 2d projections of the skin surfaces of f and g into the xy-, xz-, and yz-plane (Grimson et al. 1996). There have also been several approaches to replace Powell's method by another minimization method which is less sensitive to the initial transformation (Xu and Dony 2004).

10.2.6 The ICP Algorithm

The *iterative closest point* (*ICP*) *algorithm* of Besl and McKay (1992) is more general than the method described in the previous subsection (see Fig. 10.18). Rigid registration can be applied to point sets, implicit and parametric curves and surfaces, as well as to facetted surfaces. The two scenes f and g may be surfaces or volumes. The scene f is called the data to which the model g has to be fitted. Fitting minimizes distances of data points to the model that may be described by a variety of different representations.

Algorithms for computing the distance of a data point to the model need to be given for various model representations. If the model is a point, it is simply the Euclidian distance; if it is a cloud of points, then it is the distance to its closest point. If it is a parametric surface (e.g., a spline), distance computation depends on the type of surface. A closed-form solution may no longer exist. The distance for some implicit surfaces may be computed by inserting the point coordinates into the

Fig. 10.18 The ICP algorithm repetitively computes point correspondences based on distances and then a registration transformation based on pairs of corresponding points. In this example, one of the correspondences is wrong at the first iteration resulting in a small registration error, which is then corrected in the second iteration where all correspondences are correct

surface equation. For each of the measures, a method has to be given to return the closest point $CP(\boldsymbol{p}_i, \boldsymbol{q}_j)$ in the model component q_j to a data point p_i.

Given $CP(\boldsymbol{p}_i, \boldsymbol{q}_j)$, an initial transformation $T^{(0)}$ has to be defined. The iterative loop proceeds as follows:

- Given a transformation $T^{(n)}$ at iteration n, this transformation is applied to the data points $\mathbf{p}_i$.
- Compute the set of all $q_i = CP\left(\boldsymbol{p}_i, T^{(n)} \cdot \boldsymbol{q}_j\right)$ for the transformed data points.
- Find optimal rotation and translation of $T^{(n+1)}$ using either the method of Arun et al. (1987) or Horn (1987) above.

The algorithm terminates when improvement in the result quality falls below some preset threshold. The quality of the result depends on the initial transformation and on the distribution of the landmark points or regions. Even though Arun et al. (1987) and Horn (1987) provide optimal solutions, this is only true for paired landmarks. The ICP algorithm iterates through different configurations of correspondence pairs, and it is not guaranteed that it terminates with a correct pairing. If landmarks are distributed on some irregular surface or if the initial transformation is close to the desired result, it is less likely that the process terminates in a local minimum.

The ICP algorithm for iterative rigid registration is the most general of all registration algorithms presented so far, and numerous adaptations using different similarity criteria, optimization criteria, and preprocessing routines have been published. A survey about the various adaptations can be found in Rusinkiewicz and Levoy (2001).

10.3 Registration of Projection Images to 3d Data

Although the majority of computer-assisted image analysis is carried out on data with three or more dimensions, the majority of images created in clinical routine are still projection images. They become particularly important in image-guided interventions. Often, planning of the intervention has been done on 3d data (CT or MRI), but the surgeon or interventional radiologist has the chance to control his or her operation using radiographs, since the apparatus for generating the images does not require much space in the operating theater and the images are generated quickly. Registering such a 2d fluoroscopy taken during a surgical intervention to a 3d X-ray CT scan from preoperative planning poses some additional problems because of the information reduction from 3d to 2d (a recent review on 2d–3d registration methods can be found in Markelj et al. 2012).

Fig. 10.19 Registration of a projection image to a 3d scene requires finding a projection of the 3d scene that matches the projection image

10.3.1 Registration Using Digitally Reconstructed Radiographs

As there is no information about localization along the projection ray in a projection image, the registration attempts to find a projection $P(f)$ of the 3d image f such that $P(f)$ registers with the projection image g. Hence, the projection image is always the reference image.

The artificial projection $P(f)$ is called *digitally reconstructed radiograph*(DRR). All possible DRRs can be assigned to locations on a sphere. This location can be specified by two angles (θ, τ) (see Fig. 10.19). Registration has to find a location $(\theta, \tau)_{\min}$ such that $\mathrm{DRR}((\theta, \tau)_{\min})$ produces the most similar image to g. Similarity criteria can be the Procrustes distance between landmark points or curves in $P(f)$ and g or an intensity criterion.

Using landmarks such as the curves of Lavallée and Szelinski (1995) has the advantage that it is fast. Only segmented curves or points need to be projected instead of creating the complete DRR. Furthermore, the similarity measure is independent of the appearance of a projected point. It does, however, require segmentation which may be tedious or even impossible to integrate into the workflow of a surgical intervention.

Using similarities based on intensities such as intensity correlation, gradient correlation, mean absolute difference, or mutual information relieves the user from providing segmented scenes.[2] If the two scenes f and g are created using the same kind of measurement signal (e.g., CT and fluoroscopy), all of the criteria mentioned above should produce good results (see Fig. 10.20 for an example of registering DSA with 3d reconstructed digital angiography). However, two influences may contribute to the degradation of the criterion:

- Noise will affect gradient-based criteria more than intensity averaging criteria such as mutual information or intensity correlation.

[2]Different similarity measures in 2d to 3d registration have been compared in Penney et al. (1998).

Fig. 10.20 Two views of registering 3d reconstructed DSA with projection DSA. The registration was necessary to map flow information from dynamic projection images to static 3d images (Hentschke et al. 2010)

- Depiction of surgical instruments in an intervention scenario will typically have a high contrast. As these instruments may not have been present at the preoperative scan, they are artifacts to the similarity criterion. Moreover, as they may move during intervention, they cannot be easily accounted for by some preoperative calibration step.

If images are from different types of acquisition systems, e.g., when comparing an MRI with a fluoroscopic image, criteria should be based on intensity edges rather than on intensities. It may still be unsuccessful if the two scenes do not possess enough common object boundaries. A pattern-based criterion, which covers the matching quality only in a region of interest, may solve this problem partially if this region is visible in the two scenes. Such criterion was presented by Weese et al. (1997) who assumed that a segmentation in a preoperative scan—in their application, it was a CT scan of the vertebrae—has been carried out that separated a structure of interest—a vertebra in their case—from the background. The DRR is created from the structure of interest. The matching criterion is applied only to this region. Given a region $(x, y) \in O$ of a projected structure and a neighborhood N_r with radius r around every pixel, the pattern intensity criterion is

$$S_{\text{pattern}} = \sum_{(x,y)\in O} \sum_{(u,v)\in N} \frac{\sigma^2}{\sigma^2 + \left([P(f)(x,y) - g(x,y)] - [P(f)(u,v) - g(u,v)]\right)^2}. \tag{10.19}$$

The criterion measures the amount of variation in the difference image $P(f)(x, y)$ in some neighborhood N_r. According to the authors, a neighborhood radius of three to five pixels was a good compromise between suppressing noise and saving computation time. The parameter σ^2 controls the influence of local contrast on the

result. It should be higher than the noise variance and lower than the contrast between edges and texture within the structure of interest.

Creating the projection images from the 3d scene is time-consuming, although this has improved with the use of efficient light field techniques borrowed from computer graphics (Russakoff et al. 2003).

10.3.2 Optimization of the Similarity Criterion

Searching the parameter space (θ, τ) may be difficult because closeness of projection parameters not necessarily implies similar values of the similarity criterion. This applies for the projection of curves as well as for intensity projections. On the other hand, widely different projection angles may produce similar images. This is the case when objects have a regular shape such as a sphere or when projections along opposite directions are compared. The former requires the initial guess of the projection direction to be close to the true projection direction. The latter may cause a projection direction not to be found even if the initial guess is good.

Several strategies exist in order to deal with this problem. A multi-resolution semi-exhaustive search has been applied by Cyr et al. (2000). They borrow from aspect graph techniques used for 3d object recognition from 2d views. Projections of f are treated as aspects of f of which one registers with g. Initially, the sphere of projection direction is sampled at a coarse resolution, and projections are computed for each of the sampled directions. The projection images are compared with g, and directions along which projections are in good agreement with g are sampled at a finer rate. The process terminates when no further improvement is observed.

Computing a matching quality based on intensity is only suitable if an assumed projection direction is close to the searched projection direction. Otherwise, the quality criterion may not hint toward the kind of change necessary for improving the registration transformation. Selecting landmarks solves this problem, but it requires their segmentation in f and g. This may be difficult for the projection image g. The solution of Cyr et al. (2000) is based on concepts of shape recognition. They use the shock graphs of segmented regions in the projection images $P(f)$ and g. A shock graph is a kind of medial axis representation where region boundaries are described by shock waves originating at the axis. A hierarchy of shocks of increasing order encodes various levels of detail of the region boundary. Similarity is computed by comparing shock graphs derived from $P(f)$ and g. It is not a simple process, as it involves assigning shocks to each other in some optimal fashion and has been solved by a stochastic optimization technique similar to simulated annealing described in Sect. 14.1.

10.4 Search Space and Optimization in Non-rigid Registration

If rigidity cannot be assumed, the search space for a possible registration increases drastically. The problem is virtually unsolvable if nothing can be assumed about the transformation, except that a displacement field $\mathbf{d}(\mathbf{x})$ exists for every point $\mathbf{x}$ that maps $f(\mathbf{x})$ to its corresponding $\mathbf{g}(\mathbf{x} + \mathbf{d}(\mathbf{x}))$. Even if some similarity criterion exists for confirming correspondence, it does not uniquely identify a displacement field. Surveys about non-rigid registration can be found in Sotiras et al. (2013), Dawant (2002), Lester and Arridge (1999). A survey that is more directed toward physicians is Crum et al. (2014).

The necessary information for solving a non-rigid registration problem comes from the reasonable assumption that topology is preserved by the displacement field, i.e., that the mapping is diffeomorphic. A diffeomorphism requires an invertible, continuously differentiable mapping between the scenes to be registered. It introduces a smoothness constraint for computing $\mathbf{d}$ (see Fig. 10.21). Various types of smoothness have been introduced:

- A linear elastic model enforces smoothness of displacement that is proportional to the displacement (Bajcy et al. 1983; Rexilius et al. 2001).
- A viscous fluid model enforces smoothness that depends on the evolution of displacement (Lester et al. 1999; Thirion 1998).

Fig. 10.21 For non-rigid registration, displacement vectors d are computed at selected locations. A smoothness constraint enforces similarity between neighboring displacement vectors and allows interpolating displacement at locations where no vector has been computed

- Registration can be modeled as freeform deformation of different kinds of splines (Rueckert et al. 1999; Chui and Rangarajan 2003).
- Displacement vector similarity globally minimizes an arbitrary penalizing term for local similarity (Brox et al. 2004).

10.4.1 Elastic Registration

In *elastic registration*, the scene is modeled as elastic surface or space that may be deformed but not cut by external forces. Deforming a scene by attraction from some external force is counteracted by forces toward the rest configuration of the elastic scene. External forces impose local similarity constraints between image f and the deformed registered image g. Elastic matching in 2d and 3d has been proposed for atlas-based segmentation by Bajcy et al. (1983) and Bajcy and Kovacic (1989) and has been used in various registration applications. The elastic model balances the image force $\mathbf{i}_{\text{force}}$ against displacement $\mathbf{d}$ depending on the gradient of the divergence of $\mathbf{d}$ and the local Laplacian ∇^2 of $\mathbf{d}$:

$$\mu \cdot \nabla^2 \mathbf{d}(\mathbf{x}) + (\mu + \beta) \cdot \nabla(\nabla \cdot \mathbf{d}(\mathbf{x})) = \mathbf{i}_{\text{force}}(\mathbf{x}). \tag{10.20}$$

The parameters μ and β are Lamé's constants representing the elasticity and stiffness with high values for μ indicating strong elasticity.

10.4.2 Using a Viscous Fluid Model

A *viscous fluid model* constrains deformation differently allowing for inhomogeneous displacement changes. It is particularly appropriate if some deformation should be varying, e.g., when non-rigid registration should model organ displacement due to surgical intervention. In the viscous fluid model, current velocity $\mathbf{v}(\mathbf{x})$ is driving the deformation:

$$\mu \cdot \nabla^2 \mathbf{v}(\mathbf{x}) + (\mu + \beta) \cdot \nabla(\nabla \cdot \mathbf{v}(\mathbf{x})) = \mathbf{i}_{\text{force}}(\mathbf{x}) \tag{10.21}$$

Current velocity at some time t has to be computed from the change of displacement vectors making the computation more costly than that of the elasticity model.

A *diffusion model* has been introduced with the demon algorithm of Thirion (1998) and its variants. The author adapts a thought experiment of Maxwell about the exchange of hot and cold particles through a semipermeable interface to registration. In the original experiment, the interface is intelligent. Demons on the interface decide for each particle whether it may path to the other side. Hot particles may only path in one direction, and cold particles may only path in the other direction. Contrary to the second law of thermodynamics that predicts increasing

Fig. 10.22 Demons are placed on the boundary of the fixed image and attract locations in the moving image

entropy (hence hot and cold particles would mix evenly after some time), this causes all hot particles to gather at one side and all cold particles to gather at the other side of the interface (the seeming contradiction to the principle that entropy never decreases was later solved since the "work" of the demons causes the overall entropy to increase nonetheless).

If applied to registration, the motion of particles would be replaced by the displacement of the moving image w.r.t. the reference image. Demons may be placed at each point in the fixed image or (if such information exists) they may only be placed at characteristic points, e.g., on the surface of a segmented object in the fixed image or on edge points computed by a suitable detector.

The demon algorithm then iteratively moves locations (see Fig. 10.22). In principle, each demon may attract each location in the moving image based on distance and the similarity criterion. The closest and most similar demon location would then cause the largest force. This would be too costly, however, since at each iteration and for each location in the moving image a resultant force from all demons has to be computed.

Hence, a given location in the moving image is influenced only by the closest demon. For a dense grid of demons, Thirion (1998) used optical flow to describe the force. It automatically searches just a local displacement field. For demons sitting on surfaces, Thirion (1998) presented a variant for rigid and affine transformations. The demon force is based on an estimated probability of the point of the moving image to be inside or outside. Other variants of the demon algorithm exist as well. In all cases, the results are regularized by subsequent smoothing (e.g., Gaussian smoothing). The diffeomorphic demons of Vercauteren et al. (2008) extend this concept by constraining the transformation to be invertible and continuously differentiable. The inversion is particularly important for normalization problems (see Sect. 10.5). Here, the goal is to map images on a common norm image. If the transformation is invertible and the inverse transformation is also continuously differentiable, it allows mapping back possible analysis results in the norm image to the original images.

Using either of the physical models has the advantage that optimization can be done by solving the differential equation iteratively. For a given displacement field and a given model, the force at every point $\mathbf{x}$ is derived from the mismatch of internal and external forces. Their difference is a vector along which a point $\mathbf{x}$ is moving. The iterative procedure is continued until equilibrium is reached.

10.4.3 Free-Form Deformations

Using *free-form deformation* employs a smoothness model given by a differentiable free-form surface or volume that may deform under an external influence from the image. Model smoothness is given by minimizing the second derivative across the surface. Image influence is a similarity criterion S such as mutual information which is defined locally around the control points of the free-form surface or volume. Given a smoothness constraint D, the registration problem is to find a displacement field $\mathbf{d}_{\min}$ so that

$$\mathbf{d}_{\min} = \arg\min_{\mathbf{d}} S(f(\mathbf{x}), g(\mathbf{x} + \mathbf{d}(\mathbf{x}))) + \gamma D(\mathbf{x}) \tag{10.22}$$

The parameter γ governs the influence of the smoothness constraint $D(\mathbf{x})$. Minimizing the function is done iteratively starting from a good initial guess usually provided by a preceding rigid registration.

10.4.4 Neighborhood Similarity

Neighborhood similarity of displacement vectors (Brox et al. 2004) does not depend on a physical model. Apart from this, it is similar to the previous strategy. It minimizes an arbitrary function penalizing non-smoothness. This can be an interpolating function, such as in the classical optical flow formulation of Horn (1986), where the square of the differences between adjacent displacements is minimized, or a strictly non-interpolating function such as Eq. (4.51) is presented in Sect. 4.3.4. Optical flow techniques may be supported by local heuristics such as the block matching proposed in Ourselin et al. (2000).

In principle, displacement vectors can be computed for every point in discrete space. In order to speed up the process, similarity computation may be constrained to a subset of locations (see Fig. 10.23). These may be either created from a subsampling of the data [as it was done in early methods like (Bajcy and Kovacic 1989)] or selected at locations of interest such as edges. The transformation between sample locations is then predicted from sample point transformations and the particular smoothness model.

Optimizing Eq. (10.22) as well as optimization from applying a physical model suffers from the many local minima of the function as well as from the fact that the smoothness term may not adequately reflect the true displacement variation in the vicinity of some point. It is advisable to begin the registration with a rigid registration that produces an initial estimate for $\mathbf{d}_{\min}(\mathbf{x})$, which is done by most authors.

Parameters for elasticity or viscosity in the methods described above are homogenous throughout the scene. This may be adequate if displacements are small. Additional information is needed to model inhomogeneous displacement if larger displacements between f and g are expected. This has been done for modeling displacement fields in preoperative and intra-operative MR scans in

Fig. 10.23 Displacement vector computation is usually restricted to some locations such as equally or randomly sampled node locations in a prespecified grid in (**a**) or feature locations (**b**)

computer-assisted surgery (e.g., Lester et al. 1999; Rexilius et al. 2001) and for registering 2d images from different modalities (duBois et al. 2005). Information about organ boundaries can be used to assign an inhomogeneous material field as input for registration.

10.4.5 Multi-scale Registration

Iterative registration methods may benefit from a multi-resolution strategy, since their convergence speed depends on the initialization. It is faster to carry out a non-rigid registration on a down-sampled version than on the original image. It may also increase the capture range for non-rigid registration which is often limited to a few pixels or voxels. Furthermore, registration on lower resolutions is less affected by noise, since the down-sampling step includes low-pass filtering of the image.

For multi-resolution image registration, a pyramidal resolution reduction scheme is employed that creates a sequence of images reducing the resolution by a factor of 2 in each direction. The pyramid usually consists of just 2–4 levels depending on the original resolution in order to preserve enough image detail at the coarsest level for generating a good initialization for the representation on the next level. The resulting displacement vectors at a given level are then up-sampled and interpolated to serve as initialization for the next higher resolution level.

Convergence speed of most iterative optimization schemes will heavily depend on how close the initial displacement vector field is to the final result. Hence, the additional work of producing the image pyramid, carrying out the registration several times, and transferring results from one resolution level to the next is

outweighed by the substantial gain that results from the reduction of the number of input scene elements by a factor of 4 (for 2d images) or 8 (for 3d images). Nevertheless, processing time can be decreased further, if similarity criteria and optimization schemes are carefully selected (Maes et al. 1999; Thévenaz and Unser 2000).

10.5 Normalization

Registration as described in the previous sections refers to the comparison of information of the same subject obtained from different modalities or taken at different times. Smoothness and similarity constraints imposed on the registration transformation are justified, since topology can be assumed to be preserved. This is no longer true if images of different subjects are to be matched. The two main reasons necessitating inter-subject matching are as follows:

- Images are taken from a group of subjects, and statistical analysis shall be carried out on variation of appearance in the images of some medically relevant feature.
- A scene f shall be mapped onto an atlas g containing model information for interpreting f.

The term normalization stems from the first reason. In order to carry out spatially resolved statistical analysis across patients, the anatomy is mapped onto some "norm anatomy" which provides a reference system. Atlas mapping can be seen as a normalization task as well with the difference that the norm is already given (the atlas) and mapping has to be found by which a scene can be mapped onto the norm anatomy.

The reasonable assumptions in registration that corresponding points exist in the two scenes, that correspondence is unique and complete, and that topology is preserved do not apply to normalization. It may even be difficult to establish what is meant by correspondence. The purpose of normalization in inter-subject studies is to study functional behavior in itself and with respect to anatomy. Hence, the same function should be performed at corresponding locations in f and g. However, in functional brain imaging, which is one of the major applications of normalization, it is known that anatomy is quite different for different individuals. Defining corresponding locations by means of the morphology and appearance of anatomy is not necessarily successful. Even if there would be common architecture of anatomy in all humans, its appearance is related to the cell level which cannot be depicted by most imaging devices. Regarding the functional architecture of the brain, it has been found that correspondences at the cell level do not necessarily infer shape correspondence on the level of sulci and gyrii (Roland et al. 1997).

Fig. 10.24 The Talairach coordinate system is defined by the ac-pc line and the mid-sagittal plane that separates the two brain hemispheres (the picture on the *left* uses a version of plate 720 of Gray's Anatomy as published in Wikimedia under Creative Commons license)

Normalization of human brain anatomy based on similarity of appearance usually neglects this knowledge assuming that function and morphology coincide at a coarse level of spatial resolution. Under this assumption, normalization is a transformation into *Talairach coordinates*. The coordinate system (depicted in Fig. 10.24)—also called *stereotaxic coordinate system*—was established by Talairach and Tournoux (1988) for a normalized localization of functional Brodmann areas. The origin of the Talairach coordinate system is the anterior point of the brain commissura (AC point). The *y*-axis of the system points to the posterior point of the commissura, its *x*-axis from left to right, and its *z*-axis from ventral to dorsal (up-down) direction. Transformation into stereotaxic coordinates can be carried out by an affine transformation for which identification of origin, axes, and length of axes is sufficient.

The Talairach transformation is only accurate at a low spatial resolution. Human brain mapping of Brodmann areas using Talairach coordinates has been used for localization of functional areas in the brain as part of stereotactic surgery planning as well as for acquiring statistical information about human brain function from PET images.

The method has been used for normalizing functional MRI images as well. However, with increased spatial resolution, it is no longer deemed adequate as sole means of normalization. In order to improve the accuracy of normalization, three different strategies are followed:

- Probabilistic Talairach coordinates comprise the correspondence uncertainty of the affine Talairach transformation (Maziotta et al. 1995).
- Application of non-rigid registration methods increases the range of acceptable transformations.
- ROI-based registration reduces the normalization to a region of interest.

The use of probabilistic coordinates acknowledges the uncertainty in the Talairach transformation, but it does not offer a solution. It will improve statistical analysis, however, if the statistical base for estimated probabilities is less biased or more reliable than data from experiments on a possibly small number of subjects.

Using non-rigid registration methods based on a smoothness model about the displacement field would truly solve the normalization problem. This is the method of choice when normalization is used for atlas-based classification and segmentation. The atlas provides a spatial label distribution which is then mapped on the patient data (it is a kind of backward normalization).

However, non-rigid registration requires that a correspondence criterion is established and that the smoothness constraint on the displacement field reflects the true displacement characteristics. Both conditions are difficult to assert for normalization. Describing correspondence and smoothness in atlas matching is probably easier than for functional brain mapping, as the atlas consists of a relatively uniform distribution of labels. It may not matter to what location a certain boundary point in the atlas is mapped in the image as long as it is mapped on the boundary of the same object. The goal is not to map equally functioning cells onto each other, but only to assign correct labels and delineate correct boundaries. Any of the criterions based on intensities discussed in the previous sections should be sufficient, provided that the general conditions such as a good first guess are met. Hence, image analysis based on matched atlas data applied non-rigid registration techniques quite successfully (Sandor and Leahy 1997; Cootes et al. 1999).

Using non-rigid registration for normalizing functional brain images has been attempted as well and is known as brain warping (Toga 1999). The accuracy of applying a criterion which is based on image intensity and smoothness to normalize brain structures is, however, limited. Findings of Roland et al. (1997) and others suggested that model-based constraints are required since cytoarchitecture is not adequately reflected in the images. Such a model would be difficult to create in view of the still limited knowledge about the functional architecture of the brain. Hence, the third strategy is to restrict normalization to a region of interest for which such a model might be easier to establish and use. ROI-based normalization is often carried out on surface maps of the boundary between gray matter and white matter because surfaces may be displayed for letting the user enter the model knowledge interactively. Methods for automatic finding of functional regions of interest have been published as well (Engel et al. 2005) (see Fig. 10.25). The advantage of selecting a region of interest is that expert knowledge about a single region is often more readily available—often with the very same people carrying out the investigation—than generalizable knowledge about the functional architecture of the whole brain.

Fig. 10.25 Partial normalization of Heschl's gyrus based on non-rigid registration with a deformable model (see Engel et al. 2005)

10.6 Joint Segmentation and Registration

Registration presented so far may be carried out on the original images or on segmented images. The latter is less likely to terminate in a local minimum as the restriction to segmented boundaries substantially reduces the potential for such minima. The reliance of a good registration on sufficient and trustworthy redundant information between the two scenes to be registered has resulted in several attempts to combine segmentation with registration. If the same segment has been identified in the fixed and the moving image, it provides ample redundancy for registration. On the other hand, combining registered images may improve information necessary to carry out the segmentation. For optimizing registration transformation and segmentation simultaneously, a probability is maximized to create a specific segmentation result given a transformation.

The problem can be formulated as an MRF in order to represent domain knowledge about expected spatial continuity for each image and between registered locations. Examples can be found in Flach et al. (2002), who describe their approach from an application perspective, and in Wyatt and Noble (2003) and Pohl et al. (2006), who develop generic models that are then demonstrated by means of specific applications. The last two papers discuss the performance of the strategy and the validity of incorporated a priori knowledge as well as potential dependencies between a priori knowledge, optimization performance, and quality of the results.

Segmentation in an MRF assigns segment labels to scene elements based on a unary term that represents data knowledge and a pairwise term that enforces similarity between neighboring scene elements. Joint segmentation and registration defined as an MRF optimization problem requires defining the registration problem

in a similar fashion as energy minimization task. The pairwise term V_{reg} enforces smoothness of displacement vectors Δ, and the unary term D_{reg} represents the data knowledge:

$$E_{\text{reg}}(\Delta) = \sum_{\mathbf{s}\in P} D_{\text{reg}}(\Delta_{\mathbf{s}}) + \lambda \cdot \sum_{(\mathbf{s},\mathbf{t})\in N} V_{\text{reg}}(\Delta_{\mathbf{s}}, \Delta_{\mathbf{t}}) \tag{10.23}$$

In this equation, P is the set of all scene elements with coordinate $\mathbf{s}$ and N is the set of all neighboring scene element pairs. The pairwise term V_{reg} computes the dissimilarity between neighboring displacement vectors. This could be, e.g., the cosine between the two displacement vectors

$$V_{\text{reg}}(\Delta_{\mathbf{s}}, \Delta_{\mathbf{t}}) = \frac{|\Delta_{\mathbf{s}} \cdot \Delta_{\mathbf{t}}|}{\|\Delta_{\mathbf{s}}\| \cdot \|\Delta_{\mathbf{t}}\|} \tag{10.24}$$

The unary term D_{reg} is the local similarity between the two scenes after transforming one of them by the current displacement field. A formulation of the unary term that has been presented by Glocker et al. (2008) is as follows:

$$D_{\text{reg}}(\Delta_{\mathbf{s}}) = \int_{\Omega} \eta(\|\mathbf{s} - \mathbf{x}\|)\rho(I_1, T_{\Delta}(I_2))\mathrm{d}\mathbf{x}, \tag{10.25}$$

where η is an influence function that decreases with distance between $\mathbf{s}$ and $\mathbf{x}$, I_1 and I_2 are the two images to be registered, T_{Δ} is the current non-rigid transformation applied to I_2, and ρ is a similarity function, such as the absolute intensity difference or mutual information. Analogous to the non-rigid registration methods presented in Sect. 10.4, similarity may be evaluated at points of a grid that is overlaid the scene or at each scene element.

The unary and pairwise terms can now be combined with a formulation of segmentation as an MRF optimization problem. For a two-label segmentation with foreground label L_f and background label L_b, the unary term for the segmentation part will consist of the probability of a pixel $\mathbf{s}$ to belong to foreground or background. The probabilities $p(L = L_f|I(\mathbf{s}))$ and $p(L = L_b|I(\mathbf{s}))$ of a pixel $\mathbf{s}$ with intensity $I(\mathbf{s})$ can be estimated from trained a posteriori probabilities. Given intensities $I(\mathbf{s})$ of the reference image and $I(\mathbf{s} + \Delta)$ of the floating image, the unary term may be the product of the probabilities of the current labels in the reference and the floating image:

$$D_{\text{seg}}(\mathbf{s}) = 1 - p(L = L(\mathbf{s})|I(\mathbf{s})) \cdot p(L = L(\mathbf{s}+\Delta)|I(\mathbf{s}+\Delta)). \tag{10.26}$$

The pairwise term for the segmentation part then measures label similarity between neighboring pixels, e.g.,

$$V_{\text{seg}}(\boldsymbol{s},\boldsymbol{t}) = \begin{cases} 0, & \text{if } L(\boldsymbol{s}) = L(\boldsymbol{t}) \\ A, & \text{otherwise} \end{cases} \tag{10.27}$$

where A is a large positive number. Other models that, e.g., integrate V_{seg} and V_{reg} so that acceptable registration difference depends on label equality (Mahapatra and Sun 2010) or that work with fuzzy or probabilistic label assignments (Parisot et al. 2012) have been proposed as well.

Having the pairwise and unary terms for segmentation and registration, they are combined to arrive at

$$\begin{aligned} E(\Delta, L) = &\sum_{\mathbf{s}\in P} \alpha D_{\text{reg}}(\Delta_{\mathbf{s}}) + (1-\alpha) D_{\text{seg}}(\mathbf{s}) \\ &+ \lambda \cdot \sum_{(\mathbf{s},\mathbf{t})\in N} \alpha V_{\text{reg}}(\Delta_{\mathbf{s}}, \Delta_{\mathbf{t}}) + (1-\alpha) V_{\text{seg}}(\mathbf{s},\mathbf{t}). \end{aligned} \tag{10.28}$$

The parameter α controls the influence of segmentation and registration, and the parameter λ controls the influence of the regularizing pairwise term on the result.

The labels of the MRF consist of all possible combinations of displacement labels $\Delta = \{\Delta_1, \Delta_2, \ldots, \Delta_n\}$ and segment labels $L = \{L_1, L_2, \ldots, L_m\}$. Compared to a segmentation alone, the number of labels can get very large. If a displacement of $(-3, -2, \ldots, +3)$ scene elements is allowed per direction in a 3d registration task and images shall be segmented into two regions, then $2 \cdot 7 \cdot 7 \cdot 7 = 686$ different labels are to be assigned. Optimization can be done efficiently using graph cuts (e.g., with α-expansion moves, see Sect. 8.1.2) using efficient computation schemes such as the one presented in Jamriška et al. (2012). Registration can be further sped up if a multi-resolution approach is used (see Sect. 10.4.5).

10.7 Non-rigid Discontinuous Registration

The regularization term for non-rigid registration enforces smoothness of the displacement field. This may not always be appropriate, since displacement may be discontinuous, e.g., when respiratory motion lets the lung move differently to other organs such as the diaphragm, when a catheter slides into an organ, or when a surgical procedure cuts into an organ. In all cases, the smoothness constraint would smooth displacement vectors at locations of discontinuity.

There are different ways to deal with this problem, all of which requiring modeling the discontinuity interface between regions with different displacement. The simplest solution requires segmentation of all objects that move differently and then computes a registration transformation independently for each segment (e.g., Wu et al. 2008). However, it has the disadvantage that not just the smoothness constraint but all aspects of the registration procedure are independent for each

region. It becomes difficult to ensure non-overlapping segments and similar global characteristics that should be guaranteed irrespective of the discontinuity at the interface.

A more elegant solution requires segmentation of discontinuity interfaces but uses global registration with a locally adaptable regularization term. An example is to use anisotropic diffusion to regularize the displacement field in order to accommodate a sliding motion at the discontinuity interface (Pace et al. 2013). Based on a segmented discontinuity interface (which is in this case the lung boundary), diffusion filtering imposes a smoothing constraint on the non-rigid registration. Simple homogeneous diffusion would achieve similar results than other global smoothing terms such as using regularizing splines. Hence, anisotropic smoothing is used to restrict smoothing at the interface.

Different to anisotropic diffusion for edge-preserving smoothing presented in Sect. 4.3.3, a displacement vector field (and not a scalar field) is smoothed. For sliding motion at the interface, the only kind of discontinuity that is allowed is a different motion along the interface (see Fig. 10.26). Hence, the amount of diffusion is defined based on the local orientation of the interface w.r.t. the direction of the translation vectors. Hence, given the orientation of the interface at some point $\mathbf{x}$ on the interface, the displacement vector is separated into displacement tangential to the interface and orthogonal to it by projecting it on the normal of the interface and on the tangent at this point. Then, a diffusion tensor is built with two components (Pace et al. 2013):

$$\|P\nabla u_i(\mathbf{x})\|^2 + w\left(\mathbf{n}^{\mathrm{T}}\nabla u_i^{\perp}(\mathbf{x})\right)^2 \tag{10.29}$$

The normal $\mathbf{n}$ is that of the closest point on the interface to the location $\mathbf{x} = (x_1\ x_2\ x_3)$, u_i is a component of the displacement vector $\mathbf{u}$ ($i = 1, 2, 3$), and $u_i^{\perp}$ is the ith component of a vector that is the projection of $\mathbf{u}$ onto the normal (it is the part of $\mathbf{u}$ in direction of $\mathbf{n}$). The eigenvectors and eigenvalues of $P = I - w\mathbf{n}\mathbf{n}^{\mathrm{T}}$ allow diffusion only tangential to $\mathbf{n}$. The weighting w decreases exponentially from 1 to 0 with increasing distance of $\mathbf{x}$ to the interface.

The first term of Eq. (10.29) allows diffusion tangential to the discontinuity interface. Hence, it diffuses both the tangential and the normal component of $\mathbf{u}$ along the interface. It smooths the displacement field along the interface. The second term allows also diffusion normal to the tangent plane but only for the component of $\mathbf{u}$ in the direction of the normal $\mathbf{n}$. Hence, it inhibits tearing the two segments at the interface while still allowing a sliding motion of the two segments w.r.t. each other along the interface.

The function is minimized for all components of $\mathbf{u}$ at all locations $\mathbf{x}$ in the image.

The method presented above still requires segmentation of the discontinuity interface. If this is not possible or inefficient, the interface can be inferred from the data similar to the inference of an edge in edge-preserving smoothing. Inference is always approximate, i.e., most or all locations will be declared part of the discontinuity interface with uncertainty (otherwise the solution would be trivial since

Fig. 10.26 Sliding motion is defined as motion discontinuity along the interface. Other kinds of varying motion that are not allowed are variations within the tissue (which are also suppressed by the global regularizer) and variations orthogonal to the interface (as it would tear the two tissues leaving an undefined void between them)

the segmentation could be derived directly from the data). Consequently, the quality of results deteriorates when the implicit model for probable interfaces does not match the true, but unknown model.

A straightforward way of integrating a data-inferred presence of a discontinuity interface into optical flow-based registration is to change the error norm of the regularizer. Optical flow can be used for registration when the assumption about the data term in optical flow computation that moving pixels keep their local characteristics is true. The regularizer minimizes the total variation of the displacement vectors. Object characteristics can be anything that should be similar according to the registration task (in the simplest case, it is just the intensity), whereas total variation is given by integrating the L_2-norm of displacement variation over the image. Since using the L_2-norm will smooth the displacement field everywhere, it will produce errors at the discontinuity interface. Hence, it needs to be replaced by a norm, so that it strongly enforces smoothness at most places in the image. Thus, a generalization of the regularized L_1-norm has been used for segmentation-free, discontinuity-preserving registration (Heinrich et al. 2010):

$$\Psi_p(r) = \left(r^2 + \varepsilon^2\right)^{\frac{p}{2}} \tag{10.30}$$

Adding the regularizer ε produces a differentiable function for arbitrary values of $p \neq 0$. For $p = 1$, it approximates the L_1-norm (which also has been used for discontinuity-preserving registration, e.g., Kiriyanthan et al. 2016) by a function that is differentiable everywhere. Variational calculus can then be used to arrive at an iterative registration scheme.

Changing the norm in a total variation approach is rather indirect for introducing a model inferring potential locations of interfaces. It is also possible to adapt the segmentation-based strategy using anisotropic diffusion instead. In this case, the interface is replaced by a (weaker) model for potential interface locations. The authors in Papiez et al. (2014) adapt a method filtering the displacement field by anisotropic diffusion at each iteration. The original method uses the image structure tensor to guide anisotropic diffusion to infer a potential interface. This is replaced

by a bilateral filter. As above, the displacement field and not the intensity is filtered at each iteration.

The main problem here is that the regional range of the bilateral filter is difficult to select globally. It may lead to oversegmentation, if chosen too small, or to overly smooth translation fields, if chosen too large. A similar problem occurs in anisotropic diffusion, since the desired effect happens only if the implicitly chosen (weak) boundary model is close to the true (unknown) underlying boundary model. In Papiez et al. (2014), the authors weighted the filter kernel with a Gaussian that depends on the local dissimilarity of the displacement field. It is somewhat similar to using the structure tensor as boundary model as it assumes that locally varying displacement mainly occurs at boundaries with sliding motion.

10.8 Concluding Remarks

Registration generates synergy from fusing two or more 2d or 3d images of the same subject that show different attributes. It computes a mapping transformation for making image fusion possible. The amount of information gained from the fused images increases with decreasing redundancy between them. Redundant information is necessary, however, for computing the registration transformation.

Missing correspondence information may be replaced by domain knowledge, but care should be taken to make sure that domain knowledge does not predominate the registration result. Often, domain knowledge enforces similar appearance between registered images. Hence, the result may look visually pleasing but does not reflect the true spatial relation between locations in the two registered scenes.

10.9 Exercises

- Consider registering CT and MR images of the head from the same patient. What kind of registration transformation would be necessary? What similarity criterion could be used and what could be used to evaluate it on the data?
- How can similarity between curves be measured, if curves are used for measuring correspondence of two scenes to be registered?
- Why is it unwise to assume that landmarks are in perfect correspondence after successful registration? Please give a similarity criterion that allows relaxing this condition.
- What is meant by Procrustes distance and how is it computed?
- What are the advantages of using anatomical landmarks instead of geometrical landmarks? What are the disadvantages?
- Why should a rigid registration transform not be computed by taking the minimum number of corresponding points and solving the associated linear equation system?

- What is the advantage of using quaternions for computing a rigid registration transformation? What is a disadvantage?
- How can point correspondence be created from two sets of landmarks of which pairwise correspondence is not known? What are the assumptions that have to be met so that this approach is successful?
- How can a rigid registration be computed from two sets of landmarks without requiring their pairwise correspondence? When and why does this method fail?
- Describe a method to detect outliers for landmark-based registration.
- What feature describes the mutual information criterion? How does it differ from the stochastic sign change criterion?
- What kind of constraints can be used for making non-rigid registration solvable? What assumption is made about the data by these constraints? Describe a situation where this assumption is not true.
- What role do demons play in the demon registration algorithm?
- How can the iterative non-rigid registration be initialized with a good guess?
- What is the result of a registration of a projection image (such as X-ray) with a 3d scene?
- Usually, registering a projection image with a 3d scene requires a good initial guess. What could be reasons for this?
- What is the goal of normalization? How does it differ from registration?
- Which problem may arise if a non-rigid registration technique is applied for a normalization task? How could this problem be dealt with?
- Explain reasons why one would like to combine registration with segmentation.
- How are labels of an MRF defined for carrying out a joint segmentation and registration?

References

Arun KS, Huang TS, Blostein SD (1987) Least-squares fitting of two 3-D point sets. IEEE Trans Pattern Anal Mach Intell 9(5):698–700

Bajcy R, Lieberson R, Reivich M (1983) A computerized system for the elastic matching of deformed radiographic images to idealized atlas images. J Comput Assist Tomogr 7(4): 618–625

Bajcy R, Kovacic S (1989) Multiresolution elastic matching. Comput Vision Graph Image Process 46(1):1–21

Besl PJ, McKay ND (1992) A method for registration of 3-d shapes. IEEE Trans Pattern Anal Mach Intell 14(2):239–256

Brown LF (1992) A survey of image registration techniques. ACM Comput Surv 24(4):325–376

Brox T, Bruhn A, Papenberg N, Weickert J (2004) High accuracy optical flow estimation based on a theory for warping. In: Proceedings of 8th European conference computer vision ECCV 2004. LNCS, vol 3024(4), pp 25–36

Chui H, Rangarajan A (2003) A new point matching algorithm for non-rigid registration. Comput Vis Image Underst 89:114–141

Cootes TF, Beeston C, Edwards GJ, Taylor CJ (1999) A unified framework for atlas matching using active appearance models. In: Proceedings of 16th international conference information processing in medical imaging IPMI'99. LNCS, vol 1613, pp 322–333

Crum WR, Hartkens T, Hill DLG (2014) Non-rigid image registration: theory and practice. Br J Radiol 77(Suppl. 2):S140–S153

Cyr CM, Kamal AF, Sebastian TB, Kimia BB (2000) 2d-3d registration based on shape matching. In: IEEE workshop math methods in biomedical image analysis, pp 198–203

Dawant B (2002) Nonrigid registration of medical images: purpose and methods, a short survey. In: Proceedings of IEEE international symposium biomedical imaging, pp 465–468

du Bois d'Aische A, Craene MD, Geets X, Gregoire V, Macq B, Warfield SK (2005) Efficient multi-modal dense field non-rigid registration: alignment of histological and section images. Med Image Anal 9(6):538–546

Engel K, Brechmann A, Toennies KD (2005) A two-level dynamic model for the representation and recognition of cortical folding patterns. In: IEEE international conference image processing ICIP2005, I, pp 297–300

Fitzpatrick JM, West JB, Maurer CR Jr (1998) Predicting error in rigid-body point-based registration. IEEE Trans Med Imaging 17(5):694–702

Flach B, Kask E, Schlesinger D, Skulish A (2002) Unifying registration and segmentation for multi-sensor images. In: Pattern recognition 2002,.LNCS, vol 2449, pp 190–197

Glocker B, Komodakis N, Tziritas G, Navab N, Paragios N (2008) Dense image registration through MRFs and efficient linear programming. Med Image Anal 12(6):731–741

Grimson WEL, Ettinger GJ, White SJ, Lozano-Pérez T, Wells WM III, Kikinis R (1996) An automatic registration method for frameless stereotaxy, image guided surgery, and enhanced reality visualization. IEEE Trans Med Imaging 15(2):129–140

Heinrich MP, Jenkinson M, Brady M, Schnabel JA (2010) Discontinuity preserving regularization for the variational optical-flow registration using the modified L^p norm. In: Medical image analysis for the clinic—a grand challenge, MICCAI workshop, pp 185–194

Hentschke CM, Serowy S, Janiga G, Rose G, Toennies KD (2010) Estimating blood flow by re-projection of 2d-DSA to 3d-RA data sets for blood flow simulations. Intl J Comput Assist Radiol Surg (CARS) 5(1):342–343

Horn BKP (1986) Robot vision. MIT Press, Cambridge

Horn BKP (1987) Closed form solution of absolute orientation using unit quaternions. J Optics Soc Am A 4:629–642

Jamriška O, Sýkora D, Hornung A (2012) Cache-efficient graph cuts on structured grids. In: IEEE conference on computer vision and pattern recognition (CVPR 2012), pp 3673–3680

Kiriyanthan S, Fundana K, Majeed T, Cattin PC (2016) Discontinuity preserving image registration through motion segmentation: a primal-dual approach. Comput Math Methods Med ID 9504949

Lavallée S, Szelinski R (1995) Recovering the position and orientation of free-form objects from image contours using 3d distance maps. IEEE Trans Pattern Anal Mach Intell 17(4):378–390

Lester H, Arridge SR (1999) A survey of hierarchical non-linear medical image registration. Pattern Recogn 32(1):129–149

Lester H, Arridge SR, Jansons KM, Lemieux L, Hajnal JV, Oatridge A (1999) Non-linear registration with the variable viscosity fluid algorithm. In: Proceedings of 16th international conference on information processing in medical imaging IPMI'99. LNCS, vol 1613, pp 238–251

Maes F, Vandermeulen D, Suetens P (1999) Comparative evaluation of multiresolution optimization strategies for multimodality image registration by maximization of mutual information. Med Image Anal 3(4):373–386

Maglogiannis I (2004) Automatic segmentation and registration of dermatological images. J Math Model Anal 2(3):277–294

Mahapatra D, Sun Y (2010) Joint registration and segmentation of dynamic cardiac perfusion images using MRFs. In: International conference on medical image computing and computer-assisted intervention MICCAI 2010, pp 493–501

Maintz JBA, Viergever MA (1998) A survey of medical image registration. Med Image Anal 2(1):1–36

Markelj P, Tomaževič D, Likar B, Pernuš F (2012) A review of 3D/2D registration methods for image-guided interventions. Med Image Anal 16(3):642–661

Mazziotta JC, Toga AW, Evans A, Fox P, Lancaster J (1995) A probabilistic atlas of the human brain: theory and rationale for its development. Neuroimage 2(2):89–101

Moderisitzki J (2003) Numerical methods for image registration. Oxford University Press, Oxford

Oliveira FP, Tavares JMR (2014) Medical image registration: a review. Comput Methods Biomech Biomed Eng 17(2):73–93

Ourselin S, Roche A, Prima S, Ayache N (2000) Block matching: a general framework to improve robustness of rigid registration of medical images. In: Medical image computing and computer-assisted intervention—MICCAI 2000. LNCS, vol 1935

Pace DF, Aylward SR, Niethammer M (2013) A locally adaptive regularization based on anisotropic diffusion for deformable image registration of sliding organs. IEEE Trans Med Imaging 32(11):2114–2126

Papiez BW, Heinrich MP, Fehrenbach J, Risser L, Schnabel JA (2014) An implicit sliding-motion preserving regularization via bilateral filtering for deformable image registration. Med Image Anal 18:1299–1311

Parisot S, Duffau H, Chemouny S, Paragios N (2012) Joint tumor segmentation and dense deformable registration of brain MR images. In: International conference on medical image computing and computer-assisted intervention, MICCAI 2012, pp 651–658

Pelizzari CA, Chen GT, Spelbring DR, Weichselbaum RR, Chen CT (1989) Accurate three-dimensional registration of CT, PET, and/or MR images of the brain. J Comput Assist Tomogr 13(1):20–26

Penney GP, Weese J, Little JA, Desmedt P, Hill DLG, Hawkes DJ (1998) A comparison of similarity measures for use in 2-d–3-d medical image registration. IEEE Trans Med Imaging 17(4):586–595

Pluim JBW, Maintz JBA, Viergever MA (2003) Mutual-information based registration of medical images: a survey. IEEE Trans Med Imaging 22(8):986–1004

Pohl KM, Fisher J, Grimson WEL, Kikinis R, Wells WM (2006) A Bayesian model for joint segmentation and registration. NeuroImage 31:228–239

Press WH, Flannery BP, Teukolsky SA, Vetterling WT (1992) Numerical recipes in C: the art of scientific computing, 2nd edn. Cambridge University Press, Cambridge

Reddy BS, Chatterji BN (1996) An FFT-based technique for translation, rotation, and scale-invariant image registration. IEEE Trans Image Process 5(8):1266–1271

Rexilius J, Warfield SK, Guttmann CRG, Wei X, Benson R, Wolfson L, Shenton M, Handels H, Kikinis K (2001) A novel nonrigid registration algorithm and applications. In: Proceedings of 4th international conference on medical image computing and computer-assisted intervention —MICCAI 2001. LNCS, vol 2208, pp 923–931

Roland PE, Geyer S, Amunts K, Schormann T, Schleicher A, Malikovic A, Zilles K (1997) Cytoarchitectural maps of the human brain in standard anatomical space. Hum Brain Mapp 5:222–227

Rueckert D, Sonoda LI, Hayes C, Hill DLG, Leach MO, Hawkes DJ (1999) Nonrigid registration using free-form deformations: application to breast MR images. IEEE Trans Med Imaging 18 (8):712–721

Rusinkiewicz S, Levoy M (2001) Efficient variants of the ICP algorithm. In: 3rd international conference on 3-d imaging and modeling (3DIM'01), pp 145–152

Russakoff DB, Rohlfing T, Maurer CR Jr (2003) Fast intensity-based 2d-3d image registration of clinical data using light fields. In: Proceedings of 9th international conference on computer vision (ICCV2003), pp 416–422

Sandor S, Leahy R (1997) Surface-based labeling of cortical anatomy using a deformable atlas. IEEE Trans Med Imaging 16(1):41–54

Sotiras A, Davatzikos C, Paragios N (2013) Deformable medical image registration: a survey. IEEE Trans Med Imaging 32(7):1153–1190

Talairach J, Tournoux P (1988) Co-planar stereotaxic atlas of the human brain: 3-dimensional proportional system—an approach to cerebral imaging. Thieme Medical Publishers, Stuttgart

Thévenaz P, Unser M (2000) Optimization of mutual information for multiresolution image registration. IEEE Trans Image Process 9(12):2083–2099

Thirion JP (1998) Image matching as a diffusion process: an analogy with Maxwell's demons. Med Image Anal 2(3):243–260

Toga AW (1999) Brain warping. Academic Press, Massachusetts

Toennies KD, Udupa JK, Herman GT, Wornum IL III, Buchman SR (1990) Registration of 3-d objects and surfaces. IEEE Comput Graph Appl 10(3):52–62

Venot A, Leclerc V (1984) Automated correction of patient motion and gray values prior to subtraction in digitized angiography. IEEE Trans Med Imaging 3(4):179–186

Vercauteren T, Pennec X, Perchant A, Ayache N (2008) Symmetric log-domain diffeomorphic registration: A demons-based approach. In: International conference on medical image computing and computer-assisted intervention, pp 754–761

Weese J, Penney GP, Buzug TM, Fassnacht C, Lorenz C (1997) 2D/3D registration of pre-operative CT images and intra-operative X-ray projections for image guided surgery. In: Proceedings international symposium computer assisted radiology and surgery (CARS97), pp 833–838

West J, Fitzpatrick JM et al (1997) Comparison and evaluation of retrospective intermodality brain image registration techniques. J Comp Ass Tomogr 21(4):554–568

Wu Z, Rietzel E, Boldea V, Sarrut D, Sharp GC (2008) Evaluation of deformable registration of patient lung 4d CT with subanatomical region segmentation. Med Phys 35:775–781

Wyatt PP, Noble JA (2003) MAP MRF joint segmentation and registration of medical images. Med Image Anal 7:539–552

Xu X, Dony RD (2004) Differential evolution with Powell's direction set method in medical image registration. In: IEEE international symposium biomedical imaging, I, pp 732–735

Zitova B, Flusser J (2003) Image registration methods: a survey. Image Vis Comput 21(11): 977–1000

11 Shape, Appearance and Spatial Relationships

Abstract

Object detection in medical image analysis can be modelled as a search for an object model in the image. The model describes attributes such as shape and appearance of the object. The search consists of fitting instances of the model to the data. A quality-of-fit measure determines whether one or several objects have been found. Generating the model for a structure of interest can be difficult. It has to include knowledge about acceptable variation of attributes within an object class while remaining suitably discriminative. Several techniques to generate and use object models will be presented in this chapter. Information about acceptable object variation in these models is either generated from training or is introduced via modeling. In either case, an efficient representation is needed with (relatively) few parameters yet being able to represent variation of shape and appearance between subjects. Applying shape (and appearance) models to the data may produce the object segmentation or they may be used as additional constraint in a subsequent segmentation process. This chapter closes with a discussion on how shape models can be used to augment data-driven segmentation by a shape model component.

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_11

Concepts, Notions, and Definitions Introduced in this Chapter

- Representation of shape and appearance;
- Template matching and Hough transform;
- Quadrics and super-ellipsoids;
- Medial axis representation;
- Active shape and active appearance models;
- Mass-spring models;
- Finite element models: part-based models;
- Atlas-based segmentation;
- Graphical models.

Instances of an object may be searched explicitly if a model of the object's outline or appearance exists. The search may be carried out automatically. Found instances are reported together with a quality of fit between model representation and data. User interaction may be allowed or required for model creation as well as for the correction of search results. Ideally, a model is defined such that interaction supporting the search can be incorporated into the model.

The power of a model to describe variants of a class of objects increases with the number of degrees of freedom that exist for describing shape or appearance. Parameters need to be specified, however, before the model can be used for searching object instances. For simplicity, we will call all these models *shape models*. It should be understood that shape comprises the outline of a figure as well as the expected appearance of the region that is enclosed by the outline.

A model in the engineering sciences abstracts or approximates a real world entity. Its purpose is to describe or simulate this entity in a more accessible fashion. In image processing and computer vision the use of the term model is similar, although its purpose is slightly different. It is used for searching, detecting, or identifying objects depicted in an image. Models in computer vision share some similarities with geometric models used in computed geometry and computer-aided design. In computer-aided geometry, a *model* is a system of rules and entities to describe rigid or non-rigid objects. A *representation* is a specific way to describe such a model. An instance of a model representation (or short: *model instance*) is a parameterized representation. We will use the term *shape model* to describe a model and its representation and *model instance* to describe a specific instance.

A shape model is a powerful constraint for foreground segmentation as well. Fitting model instances to the data singles out locations where a foreground object of known outline and appearance may be found (see Fig. 11.1). The accuracy with which this happens depends on the kind of model. Simple models only report a

Fig. 11.1 A shape pattern as model can be used to search for possible occurrences of this pattern in an image by placing an instance of this pattern at arbitrary positions in the image and comparing the correlation of the pattern with the local appearance at this location

potential site of an object instance and make a rough approximation of its actual shape. A post-processing step is needed for segmentation that uses this as spatial constraint for a local boundary search. More advanced models include shape variation and may deliver a fairly accurate description of the true object boundary. Using such model is sometimes the only automatic way to achieve a segmentation, since noise, missing or false boundaries in the data can make it impossible to generate a data-driven delineation of object boundaries.

11.1 Shape Models

Using shape models has a long history in image analysis. They help to search instances of a class of objects described by shape, and they support classification of segments based on outline and appearance attributes. Surveys on deformable shape models in image analysis are Terzopoulos and Fleischer (1988) and Cheung et al. (2002).

The two different goals do not necessarily lead to the same kind of model

- If object instances are searched, regions containing the shape need to be separated from those that do not. The number of different object instances is unknown. The data needs to be expressed in terms of the model. A *quality-of-fit measure* (*QoF*) rates correspondence between model instance and data. It guides the search and decides on acceptable deviations in the data from predictions by the model. Stopping criteria are required in order to decide whether a search has been successful (*shape detection*) and in order to decide to which extent the model instance is similar to the instance in the image (*shape deviation*). The two criteria may be different.
- If an object shall be classified by means of its shape, a descriptor of a detected object instance is generated. Perceived similarity within classes should map to represented similarity in the descriptor. The model and instances of its representation do not need to describe the exact shape or appearance of the object as long as the representation suffices for differentiation between shape classes.

Some models and representations such as the *active shape model* (ASM) (Cootes and Taylor 1992) may be applicable for both of the two purposes. However, most models may be categorized into one of the two classes listed above.

Within this chapter, we will focus on the first kind and its use for segmentation and object detection. Object recognition that can also be carried out using a shape model will be discussed only briefly. Object recognition techniques in general are surveyed in Edelman (1997). The review in Riesenhuber and Poggio (2000) contains a brief description of underlying theories for model-based object recognition.

There is potential for confusion in the use of the terms *segmentation* and *registration* when discussing the use of shape models. Using a shape model in segmentation requires the model instance to be registered with an object instance in the image. The registered model instance segments the scene because it provides an estimate of the object's boundary. Hence, registration is the method to find an object instance in the scene resulting in a foreground segmentation if the detection has been successful. Shape models used for search and delineation of object boundaries will be discussed in this chapter under the term segmentation.

For successful shape matching, attributes of an object instance in an image should be characteristic in order to make it different from other regions in the image. The shape representation needs to describe these characteristics and permissible ranges of variation. Shape representations do this in different ways:

- A constant shape representation (such as the one in Fig. 11.1) describes the expected appearance of an object instance. Locations of object instances are found by correlating the representation with the image. Object variation is captured by the correlation measure. Criteria for shape detection and shape deviation are not separable, since both have an impact on the correlation. Segmentation requires post-processing since the representation has usually too few parameters for exactly representing the object's outline. Models of this kind are presented in Sect. 11.2.
- A deformable representation includes some or all of the object-specific variations (see Fig. 11.2). Representations of such models can differentiate between foreground and background with potentially arbitrary accuracy. Deformable models and representations may be differentiated by the way how variation is represented:

  - *Implicit models* describe objects as functions of the domain in which the image is defined. Variation is implicitly given by variation of the function parameters. It enables an arbitrarily exact adaptation of the function to the object's shape and appearance, given that the parameter set of the function is rich enough. Implicit models will be described in Sect. 11.3.
  - *Explicit models* describe objects by a finite number of object details. Variation is defined explicitly on these details and is directly related to variation in the object's appearance. It may be difficult to find a finite number of attributes, which capture object appearance and its variation sufficiently well. Various kinds of explicit models will be described in Sects. 11.4–11.6.

Fig. 11.2 A variable shape representation includes (constrained) variation of the object shape

From a Bayesian point of view, shape representations are estimates of the a priori term in the Bayesian equation. The estimate is particularly simple for a constant representation. The a priori probability of an object instance is 1 for objects that have similar appearance and boundary than the model instance. The threshold for similarity is user-specified. Otherwise, it is 0.

A better approximation of the a priori term is achieved by a deformable shape representation. Shape deformation incurs a cost that should be closely related to the usually unknown probabilities. Several strategies exist for establishing the link between cost of variation and a priori probability. For ASMs described in Sect. 11.5 this is straightforward because probability distributions are learnt from samples. Proper training of the distribution, however, may require too many samples. Separating shape variation in a predefined high-level part and a trainable low-level part is a strategy for keeping probability distributions simple. An example for this is the medial axis representation (Sect. 11.4.2) where only local variation needs to be trained from samples. Variation may also be derived from the composition of a shape model from simple sub-units without further training. This is the case for physically-based shape models such as a *finite element model* (=FEM, Sect. 11.6.2).

All strategies for acquiring a priori knowledge need to deal with the problem that sufficient reliable information is seldom present. Again, there are a number of ways to deal with this:

- *Shape decomposition*: This strategy relates to the Geon theory (Biedermann 1985; Rivlin et al. 1995) (see Fig. 11.3) and the fact that objects are often recognized based on context (Toussaint 1978). In short, the theory states that objects are recognized by their composition rather than by deformation of the constituting components. In consequence, a proper, user-defined decomposition of a shape into sub-units provides a representation with few parameters that are easy to estimate (see Fig. 11.4 for an example).
- *Scale space*: This relates to a theory that different levels of importance for object features are assumed to be represented at different levels of spatial resolution

Fig. 11.3 Geon theory states that the semantics of a depicted structure mainly stems from the decomposition of it into its constituent parts

Fig. 11.4 Application of a shape decomposition strategy: providing a simplified structure of expected image content will enable identification of structures based on their configuration. The model does not have to be accurate as long as the spatial relationship between shape components is correct

[e.g., Mokhtarian and Mackworth (1986), Jackway and Deriche (1996) for an application for shape detection and Lindeberg (1994) for an introduction and review]. The shape representation is scaled and applied to an image that is scaled accordingly. Examples for scale space operators are the Gaussian pyramid (see Fig. 11.5) or the wavelet transform. The segmentation at a given resolution may be used as domain knowledge for a subsequent segmentation at a higher resolution.

- *Shape hierarchy*: The strategy is also rooted in the Geon theory. It is similar to shape decomposition and may be combined with it. The underlying assumption is the idea of a semantic scale space, i.e., that shape components can be ordered in a way that the most important shape attributes are described at the highest level.

Fig. 11.5 Three stages of a multi-scale representation (up-sampled to the original size). Representation of semantics will be different at different scales. In this example, the fact that the ventricles consist of a *left* and a *right* ventricle is representable only at scales depicted in **a**, **b** while the fact that the ventricles are in the *center* of the brain is representable at the lowest scale depicted in **c**

Less important detail may be neglected for segmentation. Defining semantic scales requires user intervention since specifying importance is highly application-dependent and requires high-level knowledge.

Some or all of these strategies have been employed by most of the representations being described in the following sections.

11.2 Simple Models

If shape can be described by few parameters with values in a finite range, finding an object instance can be done by sampling the parameter space and evaluating a quality-of-fit measure. Two often used models are rigid templates and the Hough transform.

11.2.1 Template Matching

A rigid *template* $t(\mathbf{x})$ is defined on a domain $0 \leq \mathbf{x} \leq \mathbf{x}_{\max}$. Values of t describe outline and appearance of an object in an image $f(\mathbf{x})$ by assuming that a QoF measure $d_{\mathrm{QoF}}(f(\mathbf{x}), t(\mathbf{x} + \Delta\mathbf{x}))$ exists that is minimal if an object instance can be found at position $\Delta\mathbf{x}$ in f. Often used quality-of-fit measures are the mean square distance

$$d_{\mathrm{MSD}}(f, t, \Delta\mathbf{x}) = \sqrt{\sum_{\{\mathbf{x}|0 \leq \mathbf{x} \leq x_{\max}\}} (f(\mathbf{x} + \Delta\mathbf{x}) - t(\mathbf{x}))^2} \tag{11.1}$$

or the correlation coefficient

$$d_{\text{corr}}(f,t,\Delta\mathbf{x}) = 1 - \frac{\sigma_{f,t}}{\sigma_f^2\sigma_t^2}$$
$$= 1 - \frac{\sum_{\{x|0\le x\le x_{\max}\}}(f(\mathbf{x}+\Delta\mathbf{x})-\bar{f})(t(\mathbf{x})-\bar{t})}{\sum_{\{\mathbf{x}|0\le \mathbf{x}\le x_{\max}\}}(f(\mathbf{x}+\Delta\mathbf{x})-\bar{f})^2\sum_{\{x|0\le x\le x_{\max}\}}(t(\mathbf{x})-\bar{t})^2}, \tag{11.2}$$

where $\bar{f}$ is the expected value of f in the region overlapped by the template and $\bar{t}$ is the expected value of the template (see Fig. 11.6 for an example).

The QoF function in Eq. (11.1) requires intensities of f and t to be equal if the template overlaps an object instance. The QoF function in Eq. (11.2) only requires the template intensities to be linearly correlated to the intensities of an object instance in f. If covariance d_{cov} is used instead of correlation, template matching can be computed fast in the frequency domain

$$d_{\text{cov}}(f,t,\Delta\mathbf{x}) = \mathbf{FT}^{-1}(\mathbf{FT}(f)\cdot(\mathbf{FT}(t))^*)(\Delta\mathbf{x}), \tag{11.3}$$

where $\mathbf{FT}$ and $\mathbf{FT}^{-1}$ are the Fourier transform and its inverse and $\mathbf{FT}(t)^*$ is the conjugate complex of $\mathbf{FT}(t)$. The method requires t to be defined over the domain of f. Since the template domain is usually much smaller, unknown values for t outside its original domain are filled with zeros and the template is created such that its expected value is $\bar{t} = 0$. The difference between the zeros in the template and the true intensities in f degrade the matching result.

Template matching assumes that exact outline, appearance, orientation and scale of an object are known. Small variations may be tolerated as long as the response of the QoF function in background regions is still different from that of a true match. Defining the threshold can be difficult.

Fig. 11.6 Template matching with two different circular templates: the smaller template generates strong responses for the small objects in the image while the larger template generates the strongest response for the large object on the lower left in the original image although its shape is not circular

Variation of an object's orientation and scale quickly degrades the QoF measure. In this case, a bank of templates at different scales and orientations needs to be applied to the scene. The result of the matching is the optimum response from all these templates.

Template matching may be accelerated by introducing a multi-scale strategy. Templates are applied first at a coarser resolution. Acceptance of potential object sites is generous at this stage and serves to restrict the search space at finer resolutions.

11.2.2 Hough Transform

The *Hough transform* introduced in Sect. 5.2 can be extended to arbitrarily parameterizable boundaries without changing the principles of the method. A particularly simple variant is the search for circular objects (Davies 1988). The parametric equation is

$$r(c_1, c_2) = (x_1 - x_{c1})^2 + (x_2 - x_{c2})^2. \tag{11.4}$$

The 3d parameter space (c_1, c_2, r) requires every edge point to vote for all centers and radii of circles containing this edge point as boundary point. If the radius is known, the problem becomes 2d with every edge point voting for potential circle centers (see Fig. 11.7).

With increasing dimension of parameter space the average number of votes per accumulator cell may become so small that the voting result is no longer reliable. If too many parameters are required to describe a shape or if no parametric description of the shape exists, the parametric Hough transform can be replaced by the *generalized Hough transform* (GHT), which uses lookup tables to represent shape instead of an analytic equation. A review of the different ways to compute the GHT can be found in Kassim et al. (1999).

Fig. 11.7 The Hough transform for *circles* votes for *circles* of a given size

Fig. 11.8 Tabulation of (r, α) values with respect to ϕ: If the shape is concave, there may be several entries for the same value of ϕ

For the GHT, an arbitrary reference point $\mathbf{x}_r = (\mathbf{x}_{r,1}\ \mathbf{x}_{r,2})$ is chosen that represents a 2d shape described by boundary points $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$ with coordinates $\mathbf{x}_i = (x_{i,1}\ x_{i,2})$. The spatial relation of each boundary point to the reference point is given by

$$x_{r,1} = x_{i,1} + r_i \sin\alpha_i, \quad x_{r,2} = x_{i,2} + r_i \cos\alpha_i, \tag{11.5}$$

where α_i is the angle of a line from $\mathbf{x}_i$ to $\mathbf{x}_r$ with the $\mathbf{x}_1$-axis (see Fig. 11.8). A boundary point is characterized by its orientation as given by the angle ϕ of the tangent at $\mathbf{x}_i$ with the $\mathbf{x}_1$-axis. Pairs ($\mathbf{r}$, α) are now tabulated with respect to ϕ.

For carrying out the Hough transform in an edge image, first the angle ϕ of the gradient with the $\mathbf{x}_2$-axis is computed. This angle corresponds to the angle between edge tangent and $\mathbf{x}_1$-axis. Then, ($\mathbf{r}$, α)-values corresponding to this angle are taken from the $\mathbf{r}$-table and are used to compute potential locations of reference points. The edge point votes for all these locations. There will be more than one entry in the $\mathbf{r}$-table for a given angle ϕ if the shape of the model is concave.

The GHT, as described above, is actually a special case of the GHT since orientation and scale of the shape need to be known. If rotation around some angle θ and scaling are unknown, the dimension of the Hough space is increased by these two variables. A vote for an edge at location $(x_{i,1}\ x_{i,2})$ is given for the cells $H(x_{r,1}, x_{r,2}, \theta, s)$, fulfilling

$$\begin{aligned} x_{r,1} &= x_{i,1} + s \cdot r_i \left(x_{i,1}\cos\theta + x_{i,2}\sin\theta\right), \\ x_{r,2} &= x_{i,2} + s \cdot r_i \left(-x_{i,1}\sin\theta + x_{i,2}\cos\theta\right) \end{aligned} \tag{11.6}$$

The angle α of the orientation of the edge gradient is no longer needed because the rotation between model and object instance in the image is unknown.

11.3 Implicit Models

An implicit model is a function $m(\mathbf{x}) = 0$ of image space describing the outline of object instances that it may contain. Implicit models are commonly restricted to boundary descriptions although appearance models are possible as well. Simple implicit models contain only few parameters restricting the range of representable objects in a similar way than models described in the previous sections. An examples for a simple implicit model describing object boundaries in 3d is the *quadric* $\mathbf{Q}(\mathbf{x})$

$$Q(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} = 0 \;\text{ with }\; \mathbf{Q} = \begin{pmatrix} a & e & f & g \\ e & b & h & j \\ f & h & c & k \\ g & j & k & d \end{pmatrix}, \tag{11.7}$$

where $\mathbf{x} = (x_1\ x_2\ x_3\ 1)$ is represented in homogeneous coordinates and $\mathbf{Q}$ is the parameter matrix of the quadratic equation

$$ax_1^2 + bx_2^2 + cx_3^2 + 2ex_1x_2 + 2fx_1x_3 + 2hx_2x_3 + 2gx_1 + 2jx_2 + 2kx_3 + d^2 = 0 \tag{11.8}$$

A quadric may describe a large number of regular boundaries which are all symmetric with respect to the origin of the coordinate system. Used as shape model, parameter ranges describe expected object shapes.

Fitting requires a QoF measure that connects the model with the data. Since the model describes boundaries, the QoF measures the probability of locations $\mathbf{x}$ of being boundary points for which $\mathbf{Q}(\mathbf{x}) = 0$ holds. This can be done by computing the average gradient strength in an image weighted by the inverse distance to the boundary $\mathbf{Q}(\mathbf{x}) = 0$:

$$\mathrm{QoF}(Q_i) = \sum_{\mathbf{x}\in D(\mathbf{x})} \frac{\nabla f(\mathbf{x})}{|Q_i(\mathbf{x})|^n + 1}, \tag{11.9}$$

where $D(\mathbf{x})$ is the domain of $\mathbf{x}$ and n controls how the quality decreases with increasing distance to the expected boundary. Optimization may be carried out by gradient ascent in parameter space that includes transformation parameters and the shape parameters of the quadric.

A different implicit representation are the *super-ellipsoids* presented by Barr (1992), which are defined by

$$\left(|x_1|^{\frac{2}{e}} + |x_2|^{\frac{2}{e}}\right)^{\frac{e}{k}} + |x_3|^{\frac{2}{k}} = 1. \tag{11.10}$$

Fig. 11.9 Various shapes can be generated by varying the two parameters of the super-ellipsoid equation

For $e = k = 1$, Eq. (11.10) describes a sphere, which is inscribed in a cube with sides parallel to the axes of the coordinate system. Increasing the value of parameters e and k creates soft edges parallel to these axes (Fig. 11.9 shows several variants that can be created by varying e and k). The curvature of edges increases with n ($k \rightarrow \infty$ lets the boundaries approach the enclosing cube). The ratio e/k determines the ratio of curvatures parallel to and orthogonal to the x_3-axis. Using the super-ellipsoid equation, scaling along the three axes has to be determined from optimizing the QoF measure.

Implicit shape representations as simple as the ones described above are not suited for describing the rather intricate shapes of many anatomic objects. Hence, the representation is either combined with an explicit shape model, which uses the fitted implicit model for initialization, or it is replaced by a model offering more degrees of freedom for shape variation.

Complex shapes can be constructed by combining super-ellipsoids (Chevalier et al. 2001), but this is probably more appropriate if artificial objects shall be represented. For biological structures, the implicit model can be combined with an explicit model by adding free-form deformations of an adapted super-ellipsoid (Terzopoulos and Metaxas 1991; Bardinet et al. 1995; Gong et al. 2004). First, a super-ellipsoid is fitted to the data. The resulting shape is then enclosed by a box

that is aligned with the axes of the fitted super-ellipsoid. Control points $\mathbf{p}_{i,j,k}$ for a spline representation of the ellipsoidal surface are defined on the box surface

$$\mathbf{f}(u, v, w) = \sum_{i,j,k} \mathbf{p}_{i,j,k} B_i(u) B_j(v) B_k(w), \tag{11.11}$$

where points $\mathbf{f}(u, v, w)$ are surface locations on the super-ellipsoid. The location and number of control points may be changed in a subsequent optimization process that adapts the model instance to potential surface locations in the image.

11.4 The Medial Axis Representation

Using the *medial axis* of an object for shape representation is motivated by investigations about carriers of semantics in shape (Blum 1967). The notion of a shape descriptor consisting of a supporting curve-like structure in the center of a shape that may possibly contain branches was among others adopted from research results in human vision by Marr (1983). He claimed that humans recognize the nature and shape of a 3d figure from supporting center curves, their branches and a hypothesis about circles with varying radius perpendicular to the center curves. Circle radii are derived from apparent radii of the visible silhouette of the figure. Hence, a suitable representation of shape that efficiently separates important shape features from negligible features, noise, and artefacts should be a representation by a graph of curves with varying radii [called generalized cylinders, see Binford (1987)].

11.4.1 Computation of the Medial Axis Transform

The *medial axis transform* presented by Blum (1967) creates such a curve from a given shape. It provides the means to analyses given shapes and to synthesize new shapes. The medial axis is computed from local data information, yet resulting in a high-level representation of relevant shape detail. This is of course only true only in cases where the desired high-level shape description is buried in the outline of the object in question. A major part of research regarding medial axis-based representations has been devoted to establish such a relationship between an object outline and a meaningful medial axis (see, e.g., Giblin and Kimia 2004).

The medial axis consists of a set of points that are centers of maximal hyperspheres (circles for 2d shapes, spheres for 3d shapes) filling a bounded object (see Fig. 11.10). A hypersphere is maximal if it is fully contained in the object and if no hypersphere with the same center exists that has a larger radius and is still fully contained in the object. The point set forms connected curves if objects are defined in continuous space. For discrete space objects, several approximations of the medial axis transform exist that use various definitions of connectivity between pixels or voxels and various distance metrics.

Fig. 11.10 The medial axis is the set of points that are centers of maximal circles inscribed in the figure

Fig. 11.11 Computation of the medial axis from a binary image. Pruning attempts to remove small branches from the axis but may also remove much of the central axis

Computation of the medial axis is done by a thinning algorithm [see Fig. 11.11, for a discussion of thinning algorithms see Lam et al. (1992)]. These algorithms initially treat the complete figure as medial axis. Scene elements (pixels or voxels) are then iteratively removed until no element can be removed without violating the connectivity constraint. A simple thinning algorithm is the iterative application of a series of hit-or-miss operators.

A *hit-or-miss operator* (HOM operator) finds known figures in an image by combining morphological erosion on a binary image with one on the inverse image. The two erosion operators model the shape of the figure and its background. The HOM operator combines the result of these two operations by a Boolean "and" and returns locations where the figure on the expected background is found in the scene.

Thinning by the hit-or-miss operator is carried out by searching boundary elements in the scene that can be removed without disconnecting the remaining figure. Boundary elements can be grouped by their angle with respect to the image coordinate system. Eight different types (0°, 45°, 90°, …, 315°) exist. The two HOM operators for 0° and 45° boundary elements in the usual notation for such operators[1] are

[1]The notation is the usual shorthand notation for HOM operators combining the two structuring elements in a single operator. The '0's represent the erosion structuring element that is applied to the inverted binary image and the '1's represent the erosion structuring element that is applied to the original binary image.

$$\begin{pmatrix} 0 & 0 & 0 \\ X & 1 & X \\ 1 & 1 & 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 0 & 0 & X \\ 0 & 1 & 1 \\ X & 1 & 1 \end{pmatrix}.$$

The remaining HOM operators are generated by rotating these two operators by multiples of 90°.

Thinning is carried out by applying all eight operators to the binary image, marking elements found by the HOM operator, and removing them. The process ends when no further elements can be removed. The remaining elements are the medial axis (sometimes also called the *skeleton*) of the figure. A distance transform is applied to the original image. Distances at medial axis elements are the radii of the maximal hyperspheres centered at those elements.

11.4.2 Shape Representation by Medial Axes

The medial axis does not always describe the shape appropriately for object detection. The main problem stems from the fact that every intrusion or protrusion of the boundary spawns a new branch of the medial axis system. Although some of these branches may be important shape attributes, others represent detail pertaining to a specific object instance but not to the object class in general.

There are several ways to deal with this. Application of the multi-scale theory is one of it (Pizer et al. 1987). Medial axes are created in scale space, e.g., computed on images of a Gaussian pyramid. If scale space correlates with importance of figure detail, the scale space representation can be used for shape analysis. If figure details, which are relevant for the intended application, manifest themselves at a known scale, the medial axis at this scale will be chosen for representation. The strategy often helps to separate noise effects from true shape details.

The hierarchy induced by scale space does not always coincide with a semantic hierarchy. Spatial scale for some detail of an object may vary within an object. If, for instance, palm and fingers of the hand are assumed to be at the same level of a semantic hierarchy, a higher spatial resolution may be required to represent the fingers than the hand. In such case, it is more appropriate to define a semantic scale space by imposing hierarchy levels by the user, as it has been suggested recently for using a medial axis representation (called *m-rep*) in segmentation and registration (Pizer et al. 1999; Joshi et al. 2002). The authors suggest that a hierarchy of representations is to be created that consists of four different levels:

1. *Object level*. Primitives at this level are subfigures. Each subfigure has its own medial axis representation. It may interact with other subfigures by changing relative position, orientation or scale with respect to each other. Subfigures are ordered hierarchically and represented as an acyclic, directed graph. Relations may be differentiated into those between subfigures at the same level (typically different details of a shape or group of shapes) and those between a (sub)figure

and its descendent child figures (typically the relationship between two subsequent levels in the scale space defined by the subfigure hierarchy).

2. *Figural level*. Primitive at this level are medial axis representations describing the specific shape of a subfigure.
3. *Medial axis primitive level*. Interactions between elements of the medial axis are represented at this level. The medial axis representation consists of a sequence of nodes representing a sampling of the medial axis. Nodes are attributed with their positional relation to neighboring axis nodes, and with orientation and distance to their two closest boundary points of the subfigure that they represent.
4. *Boundary primitive level*. The medial axis primitives are meant to represent shape variation of a class of shapes. The position of boundary points of an actual object instance may differ from this because of influences from data acquisition or within-class shape variation.

Hence, the m-rep implements a *part-based model* at all its different hierarchy levels, since it does not only represent shape attributes but also relations between different shapes. If the m-rep is applied to segmentation, it provides domain knowledge. Deviation of a model instance m from the expected shape is permitted but incurs a penalty $C(m)$. The penalty term is assumed to be related to the a priori likelihood of the model to take this particular shape. The authors of the m-rep present a geometric term for estimating C but claim that replacing this by a trained probability distribution is possible.

$C(m)$ is combined with a cost term $C(x, m)$ related to the likelihood of obtaining an image x given the model instance m. In Joshi et al. (2002), this is again a geometric term based on the distance between expected and data appearance. If nodes are boundary nodes, this is related to the gradient length in the image.

Segmentation using an m-rep is carried out by placing the model in the image and letting it deform to maximize $C(m) + C(x, m)$. It produces a MAP estimate of an optimal segmentation if $C(m)$ and $C(x, m)$ are assumed to be the logarithms of the a priori probability $P(m)$ and the likelihood function $P(x|m)$, respectively.

11.5 Active Shape and Active Appearance Models

The *active shape model* (ASM, also called *statistical shape model*, SSM) represents the shape of an object by an expected shape and permissible variations (see Fig. 11.12). It was presented by Cootes and Taylor (1992), Cootes et al. (1995) and has found numerous applications since then. It is a *point distribution model* (PDM) representing attributes of boundary points. The distribution is estimated from feature vectors of sample instances. The concept is simple and very attractive. Variation is directly related to the Bayesian formulation of image segmentation. Observed variation from a training phase is used to predict variation for unknown objects.

An active shape model describes a K-dimensional shape that is represented by L boundary points in a *shape feature vector* $\mathbf{s} = (s_0, s_1, \ldots, s_N) = (x_{1,1}\ x_{1,2}\ \ldots\ x_{1,L}\ x_{2,1}\ x_{2,2}\ \ldots\ x_{2,L}, \ldots, \mathrm{x}_{K,1}, \ldots\ \mathrm{x}_{K,L})$, where $x_{k,l}$ is the kth component of the lth

Fig. 11.12 Principle of an active shape model: a shape and acceptable variation taken from a sufficiently large training data set is used to find instances of the object in an image

boundary point $\mathbf{x}_l$. Elements of the shape vector are treated as random variables. The density function for each random variable is assumed to be Gaussian with unknown mean and variance.

The concept can be easily applied to any dimension K. However, increasing the dimension of the shape vector will require more samples in feature space in order to compute a reliable estimate of the distribution. The dimension of feature space is large, even for 2d objects. An outline in 2d that is represented by 50 boundary points spans a 100-dimensional feature space. If the features were truly independent, it would require a very large number of samples for computing reliable estimates for mean and variance of the Gaussians than it is usually available.[2] Fortunately, it can be safely assumed that point locations are highly correlated. Samples will occupy a much lower dimensional subspace. Decorrelating the data using principal component analysis will result in a small number of uncorrelated components of which parameters can be estimated independently of each other.

11.5.1 Creating an ASM

The probability density function of the ASM is meant to reflect deformation within an object class. The position or orientation of an object instance in some external reference system such as a world coordinate system shall not be represented. This would only make sense if this reference system is somehow object-specific (such as a patient coordinate system). Training from samples hence requires proper alignment of training shapes.

[2] A rule of thumb borrowed from statistical pattern recognition for estimating likelihood functions from samples predict for a 100-dimensional feature space that at least 2^{100} samples would be needed.

Computing the density functions of an active shape model from training samples consists of several steps:

- In *landmark selection*, a number of points on the object boundary are to be identified in all samples. Cootes and Taylor (1995) suggested a hierarchy of landmarks:

  – *Primary landmarks* are anatomical landmarks (see Fig. 11.13). Semantic equivalence between different exemplars of the training set is assured. However, it may be difficult to detect primary landmark locations based on local image characteristics.
  – *Secondary landmarks* are locations of extrema of some local image features such as curvature (see Fig. 11.13). Since they are defined on local image features, it is possible to detect them automatically. However, semantic equivalence cannot be assured. A secondary landmark may not even be present in all data sets of the training data. Hence, the validity of the assumption needs to be tested that a secondary landmark indicates semantically equivalent locations in different exemplars of the training data.
  – *Tertiary landmarks* are equally spaced landmarks that fill the gap between uniquely identifiable primary or secondary landmarks (see Fig. 11.14). Their purpose is to represent curvature properties of regions without other landmarks.

- Landmarks are *aligned* with respect to a common, shape-specific coordinate system by minimizing distances between corresponding landmarks (Procrustes analysis).

Fig. 11.13 Primary landmarks are named anatomical locations, while secondary landmark are geometrically identifiable locations such as ridge intersections

Fig. 11.14 Tertiary landmarks are used to represent curvature of the shape boundary. Sampling may depend on local curvature (*left*) or it may be an even sampling (*right*)

- The estimated covariance matrix is *de-correlated* by applying the PCA transform. New, uncorrelated features $\mathbf{s}' = \mathbf{\Phi s}$ are generated from the matrix $\mathbf{\Phi}$ containing the eigenvectors of the covariance matrix.
- The feature space is reduced by removing axes for which corresponding eigenvalues indicate insignificant feature variance. The selection of significant *modes of variation* is commonly done by thresholding accumulated variance. Modes are kept as long as the sum of variances is below some percentage of the total variance in the training data.

Several aspects should be kept in mind when creating the probabilistic shape model. They can influence the performance since landmark selection assumes that a set of semantically equivalent locations can be identified on each shape. Apart from the fact that it is difficult to define what is meant by semantic equivalence, it is often impossible to find a sufficient number of them. Interactive landmark selection requires human interaction which is seldom feasible in a medical environment. Consequently, landmark detection is based on local attributes such as curvature (Brett and Taylor 1999) or by introducing high-level knowledge such as registering the shape with an atlas containing known landmarks (Frangi et al. 2001). Geometric shape features such as the ones described in Chap. 5 could be used as well. These attributes may not always capture the full meaning of semantic equivalence. Using tertiary landmarks may fail to describe equivalent positions, if shape variation between primary or secondary landmarks is large.

Alignment is not simple because neither the mean shape nor the shape-specific coordinate system is known in advance. Finding the mean shape is commonly done iteratively. A sample $\mathbf{s}_0$ is selected from the training data and declared to be the mean shape $\bar{\mathbf{s}}$. All other samples are aligned to this shape. Then, a new estimate for the mean shape is computed. The process is repeated if the new mean deviates from the old mean by more than some threshold value. This is known as generalized Procrustes analysis.

The transformation between world and shape coordinate system is assumed to be rigid. Hence, shapes are translated with their center of gravity into the center of gravity of the mean shape. A rotation is then searched that minimizes the Procrustes

Fig. 11.15 Alignment based on invalid assumptions about the object-specific intrinsic coordinate system may lead to wrongly estimated shape variation

distance $\|\mathbf{s} - \bar{\mathbf{s}}\|$. For taking scaling into account, Cootes et al. (1995) suggested to normalize all shape vectors $\mathbf{s}$ such that $\|\mathbf{s}\| = 1$.

Problems from alignment arise when the assumptions about the mean shape and the transformation are not true (see Fig. 11.15). Using centers of gravity for finding the translation vector implies that these centers of gravity are semantic landmarks. It may not be true, if object instances of a class vary widely. Accounting for scaling, whether by normalization of the shape vector or by other means, may be inappropriate as well, if part of the scaling is due to object-specific variation. Hence, often some of the variation in the model will be due to alignment errors.

The often limited number of samples for training from which the probability distribution is estimated may decrease the significance of the estimate. As pointed out above, the relatively small number of samples used for training is justified by the assumption that the actual number of uncorrelated components describing shape variation is much smaller than the number of elements in the shape vector. This is certainly true, but it is just unknown how many uncorrelated components (modes) describe the shape variation. It is computed from a covariance estimate from few samples in the original, high-dimensional space.

11.5.2 Using ASMs for Segmentation

The ASM approach is useful for image segmentation because many of the problems listed above relate to classification issues rather than to matching issues. Deforming an ASM instance that is placed in the vicinity of some object instance so that it takes its most probable shape will often deliver a good approximation of the true

boundary even though the a priori probability of the deformed ASM instance in the image may not reflect the true a priori probability.

Using an ASM in segmentation requires a method to align and deform the shape such that it fits a potential shape instance in the image. Given a truncated $M \times N$ matrix $\mathbf{\Phi}'$ consisting of the first M eigenvectors (i.e., the largest M modes of variation) of $\mathbf{\Phi}$, boundary points of a shape are approximated by

$$\tilde{\mathbf{s}}(\mathbf{b}) = \bar{\mathbf{s}} + \mathbf{\Phi}'\mathbf{b}, \tag{11.12}$$

where $\mathbf{b}$ is a feature vector that describes deviation from the mean shape $\bar{\mathbf{s}}$. Since modes are uncorrelated, the a priori probability to belong to a shape class can be approximated from multiplying the underlying feature probabilities

$$P_{\mathbf{s}}(\mathbf{b}) \approx \prod_{i=1}^{M} \frac{1}{2\pi\sigma_i} \exp\left(-\frac{b_i}{\sigma_i^2}\right). \tag{11.13}$$

Given potential boundary points in a scene (e.g., from edge detection), the objective is to find transformation parameters and values for $\mathbf{b}$ that maximize this probability given the data information.

The search for pose parameters is independent of the search for shape features $\mathbf{b}$, since transformation parameters were excluded during generation of the ASM. Hence, the mean shape with $\mathbf{b} = \mathbf{0}$ is registered with the image using a rigid registration algorithm (see Chap. 10 and Fig. 11.16).

The result is a pose estimate $\mathbf{T}$ for which a deformation vector $\mathbf{b}$ is computed that maximizes $\log P_{\mathbf{sT}}(\mathbf{b}) + \log P_{\mathbf{sT}}(f|\mathbf{b})$ given the data (see Fig. 11.17). The log-likelihood function $\log P_{\mathbf{sT}}(f|\mathbf{b})$ can be computed by summing likelihoods of boundary points belonging to the true boundary. Similar to optimization for m-reps, this can be done by measuring the gradient length at presumed boundary locations.

Fig. 11.16 The first step of segmentation with an ASM is to register the shape model with the data accounting for attributes not captured by the ASM such as position, orientation and scale of the model instance

Fig. 11.17 Local deformation is done after the average shape has been registered

The deformed shape is used as a new estimate for computing the next pose estimate. The process is repeated until convergence.

11.5.3 The Active Appearance Model

The ASM can be extended to an *active appearance model* (AAM) comprising intensity and texture information at points of the PDM (Cootes et al. 1998). A shape vector $\mathbf{s}$, which after decorrelation is reduced to $M_{\mathbf{s}}$ modes of variation from a mean shape $\bar{\mathbf{s}}$, is concatenated with an appearance vector $\mathbf{a}$. The latter may contain intensity or texture values for each point of the PDM being measured in the vicinity of the corresponding point.

Appearances are initially trained separately from shape features. They are decorrelated as well resulting in $M_{\mathbf{a}}$ modes describing appearance variation. The two aspects of an object are then combined by

$$\hat{\mathbf{b}} = \begin{pmatrix} \mathbf{b}_a \mathbf{W}_a \\ \mathbf{b}_s \end{pmatrix}, \tag{11.14}$$

where $\mathbf{W}_a$ is a diagonal matrix which may contain weights to rate the importance of appearance features. The resulting feature vector is de-correlated again in order to remove any linear correlation that may exist between shape and appearance features. This results in the final feature vector $\mathbf{b}$ that represents mode variations for integrated shape and appearance. The AAM is matched with the image in a similar way than the ASM.

11.5.4 Deficiencies of Statistical Shape Models

Besides criticisms regarding the acquisition of training data as mentioned above, problems may also result from the selection of the parametric model and from its application to particular cases. The reader should keep in mind that a model (the shape model or any other a priori assumption) is only necessary because the data information is insufficient for arriving at the correct decision, i.e., in our case for arriving at a correct segmentation. Hence, information from data and model will contradict each other (otherwise the model would not be necessary) and the segmentation algorithm will have to define how to deal with these contradictions. In the example on how to apply the ASM to segmentation, this was solved by a weighting that has to be determined a priori by the developer or the user based on the trustworthiness of the data and the model.

The assumption of normally distributed point locations in the canonical coordinate system is often not true. A simple example is the rotating point in Fig. 11.18. If a Gaussian distribution is assumed, the resulting estimate correctly predicts that all locations with the same distance to the fixed point have the same probability, but it makes a completely wrong prediction about most probable location of the moving point.

A more complex scenario is a shape that is equally likely one of two kinds, e.g., it may either be L-shaped or F-shaped. The mean shape would be E-shaped which would be predicted as the most likely shape assuming that the distribution is Gaussian. Such different subtypes may occur when anatomical objects are modelled. If it is known beforehand, different shape models may be trained for the subtypes. All subtype models would be used to search for a shape instance in the data. However, if the existence of different subtypes is unknown, the trained unimodal Gaussian distribution would not predict the correct occurrence frequency of the different subtypes.

Even if the shape and appearance distribution function is estimated correctly, a second problem remains. For a particular shape found in an image, the most likely

Fig. 11.18 Example of a shape representation of shape that is not adequately modeled by a Gaussian density function. The shape information in this simple example is given by the coordinates of a point that changes its position relative to another, fixed point. All observed positions lie on a *circle* at a non-zero distance to the fixed point. The distribution function, reconstructed from the samples, predicts, however, that the position of the moving point is the more likely the close it is to the fixed point

shape is not necessarily the best guidance. It becomes apparent if the support from data (such as edge information) is weak. Shape-driven segmentation would then drive the segment boundary towards the most probably shape although the particular shape might deviate substantially from this. This is only acceptable if the data information is weak everywhere in the image. In such case, the most probable shape may be the best guess given the circumstances, although information that the result is mainly driven by the model information and not much supported by the data should be given to the user. However, driving the segmentation towards the mean shape is unacceptable, if the data is weak in just few places, e.g., in places where two adjacent organs with similar appearance reduce the contrast at the organ boundary. The data information may be sufficient to determine just where in the shape probability distribution this particular shape is located. The shape model should complement the data by searching the most probable continuation in this region of weak data information and not the most probable shape (see Fig. 11.19).

Several strategies within the ASM concept or outside of it have been presented to deal with the two problems, which will be discussed in the following subsections.

Fig. 11.19 A priori probability of the expected shape is not always a good guidance for segmenting a particular image. It may lead to unexpected results when the data is unreliable at just a few locations and the shape does not vary much in other places. Hence, the outcome at the *lower right* of the picture is more likely because the prior is more likely although the data supports a solution such as the one depicted at the *lower left*

11.5.5 Changing the Parametric Model

Cootes and Taylor (1999) already presented a solution that addresses the problem of having a multimodal distribution function. Before feature reduction, a multivariate normal distribution (as in the original ASM approach) is assumed. After feature reduction by selecting the modes of variation, a Gaussian mixture model (GMM, Sect. 7.1.1) is fitted to the data. The authors note that using a kernel density estimator would be an even better approximation from the samples, since initialization for iteratively estimating the GMM from samples can be difficult, but resort to the GMM approach for its greater efficiency (a justification that was certainly correct at that time).

Choosing a GMM does not solve the problem of always preferring the most probable shape if data information is weak. A partial solution to this is to use a weighted attraction of the model instance during fitting. For information about the confidence in the data at different locations, a confidence-weighted fitting of the shape model can be used (Cootes et al. 1994). It tends to accept low-probability displacements in regions of the image where the confidence in the data is high.

11.5.6 Using a Part-Based Model

Instead of using a more complex distribution function such as the GMM, it may sometimes be simpler to separate the shape into the different parts and represent variations within and between parts separately. Furthermore, for object detection, spatial relationships between parts may be more decisive for finding an object than its shape. Examples where a part-based model is more appropriate are highly structured anatomic objects such as the spine or objects whose position is defined by nearby objects, such as the pancreas that lies between spleen and the intestines.

A hierarchical decomposition, similar to that used in the m-rep hierarchy described in Sect. 11.4.2, simplifies the representation of inter-subject variability by an ASM at the expense of user-supplied knowledge. The shape is decomposed into a group of subshapes that are represented as ASMs. Relations between subshapes such as relative orientation and scale are defined and modeled as a high-level ASM (AlZubi and Toennies 2003).

The model consists of several layers. On the first layer, parts are described by ASMs. Relations between parts are described by a layered, directed acyclic graph on top of this. In each layer, spatial relationships between elements of the previous layer are represented by edges connected to a node of this layer (see Fig. 11.20). Elements of the previous layer may have more than one relationship.

Relations between parts are of two kinds. *Pose parameters* describe rigid transformations between two parts and *shape parameters* describe potential dependencies between two parts. Shape dependencies between parts are defined on the second layer using selected modes of variations from the previous layer. First, modes from all parts that are connected to the same node on the second layer are combined in a common feature vector. Then, training data is used to determine

Fig. 11.20 In a part model, a master shape (in this case the palm of the hand) determines relations of other shapes. Variation of each part is described by an ASM. Variation of part relations are represented by a 2nd level ASM

linearly uncorrelated features by a second layer PCA (this is similar to the decorrelation of shape and appearance features in the AAM).

An object search is initiated by a single part, which has to be present and detectable in the image. Using the part relationships of this part, position and shape of related parts are inferred using the second layer statistics of the ASSM. The ASSM has been applied to biometric sketch recognition (Al-Zubi et al. 2003) and to recognition of online drawings (Al-Zubi and Toennies 2003).

11.5.7 Using a Non-parametric Model

A parametric model represents shape (and appearance) variation by functions with just as many parameters that are necessary to represent all necessary variations. A *non-parametric model* uses the samples themselves to represent shape and appearance of an unknown object by the most similar samples in the training set. It solves the two problems—the chosen parametric representation may be too simple for a given shape distribution and a given shape with a low probability may not be represented well by the distribution—at the same time. This is similar to the search for a local linear embedding (Roweis and Saul 2000) where an embedded non-linear manifold is searched in feature space that contains all samples and where shape similarity is given by Euclidean distances on this manifold.

Given shape and appearance features of an image, the set of training samples is searched for the most similar shape or shapes of potential objects in the image. This shape is then used for segmentation. This is the underlying method of Zhang et al. (2012) who use a sparse shape dictionary generated from training samples to find the most similar representation of the object in the image.

Given a shape repository $\mathbf{D} = (\mathbf{d}_1, \ldots, \mathbf{d}_K)$ where each shape $\mathbf{d}_i$ is represented by a sequence of vertex coordinates (as in the original ASM approach described above), and shape information $\mathbf{v}$ generated from the image, the method finds a

shape $\mathbf{d}_s$ that is a linear combination of a subset of shapes from $\mathbf{D}$ most similar to $\mathbf{v}$. The size of the subset is restricted by a user-specified threshold.

The task is formulated as optimization task to find parameters ($\mathbf{x}$, $\mathbf{e}$, β) that define this shape:

$$(\mathbf{x}, \mathbf{e}, \beta) = \underset{\mathbf{x},\mathbf{e},\beta}{\operatorname{argmin}} \|T(\mathbf{v}, \beta) - \mathbf{D}\mathbf{x} - \mathbf{e}\|_2^2 \quad \text{with} \quad \|\mathbf{x}\|_0 \leq k_1, \|\mathbf{e}\|_0 \leq k_2. \tag{11.15}$$

The transformation T with parameters β transforms the image features into the canonical shape reference coordinate system. The image features are coordinates generated from edge locations in the image such that the number of coordinates matches those of shapes in the shape data base. The vector $\mathbf{x}$ is the sought weight vector that produces the best fitting weighted combination of shapes from $\mathbf{D}$ by $\mathbf{d}_s = \mathbf{D}\mathbf{x}$. By restricting the L_0-norm the number of non-zero entries of $\mathbf{x}$ is bounded by k_1. A residual error $\mathbf{e}$ accounts for errors from at most k_2 outliers.

The NP-hard problem can be solved approximately if the L_0 norm is replaced by the L_1 norm arriving at

$$\mathbf{d}_s = \underset{\mathbf{x},\mathbf{e},\beta}{\operatorname{argmin}} \|T(\mathbf{v}, \beta) - \mathbf{D}\mathbf{x} - \mathbf{e}\|_2^2 + \lambda_1 \|\mathbf{x}\|_1 + \lambda_2 \|\mathbf{e}\|_1 \tag{11.16}$$

The computational cost for solving Eq. (11.16) depends on the number of samples in $\mathbf{D}$ and the number of features of each sample. Since both can be large, Zhang et al. (2012) suggest using dictionaries to represent shape samples and features of each sample. For computing the shape dictionary, a new data base $\mathbf{D}_{\min}$ of size $n \times k$ with $k \ll K$ is generated with

$$(\mathbf{D}_{\min}, \mathbf{X}) = \underset{\mathbf{D}_{\min}, \mathbf{X}}{\operatorname{argmin}} \|\mathbf{D} - \mathbf{D}_{\min} \times \mathbf{X}\| \tag{11.17}$$

The matrix $\mathbf{X}$ of size $k \times K$ contains K weight vectors that produce the most similar $\mathbf{D}$ from $\mathbf{D}_{\min}$. Hence, $\mathbf{D}_{\min}$ consists of k linear combinations of the samples in $\mathbf{D}$.

For reducing the number of features in each sample given a reduced subset $\mathbf{v}_{sub}$ from the data, Eq. (11.16) is extended by

$$(\mathbf{x}, \mathbf{e}, \beta) = \underset{\mathbf{x},\mathbf{e},\beta}{\operatorname{argmin}} \|T(\mathbf{v}_{\text{sub}}, \beta) - \mathbf{S}\mathbf{D}_{\min}\mathbf{x} - \mathbf{S}\mathbf{e}\|_2^2 \quad \text{with} \quad \|\mathbf{x}\|_0 \leq k_1, \|\mathbf{e}\|_0 \leq k_2. \tag{11.18}$$

where $\mathbf{S}$ is a diagonal matrix with entries on the diagonal set to "1" if the feature is in $\mathbf{v}_{\text{sub}}$ and to "0" otherwise. For selecting vertices for $\mathbf{v}_{\text{sub}}$ the authors apply a spatially local clustering algorithm on the input vertices.

Optimization is then carried out on the reduced vertex set and/or samples by iteratively updating β and ($\mathbf{x}$, $\mathbf{e}$) in Eq. (11.18).

The method has the advantage that it always selects the best linear combination of training shapes to represent the data. Hence, it will not drive the segmentation towards the expected value of the shape distribution but towards shapes that are most similar to the shape in the data. By accommodating outliers as indicated by the error term it is insensitive to a small number of wrongly detected vertices in the data.

The method relies on the detection of vertices in the data before the shape model can be applied. This can be difficult especially if the initial placement of the shape is far from perfect and the image contains many false edges due to noise or other artefacts. This can be avoided if shapes are matched to the complete image instead to vertices. This is the basis for shape selection in the work of Gloger et al. (2015). The authors match all shapes from the training data base to pre-processed MR images of the abdomen for segmenting the kidney. Pre-processing used an intensity-based a priori model to create a kidney probability map where each voxel value represents the probability to be part of the kidney. Kidney shapes from a training data base with K shapes are then matched to the pre-processed image data to determine the $k < K$ best-fitting shapes. Matching is inexpensive as it is carried out by correlation in frequency space.

The number k is kept variable since the number of similar shapes varies with their a priori probability (shapes close to the expected value will occur more often in the data base than others). The authors chose $k_{\min} < k < k_{\max}$ shapes with highest correlation coefficient under the condition that the correlation coefficient of each of the selected shapes was higher than some threshold. If fewer than $k_{\min}$ shapes were found the shape constraint was rejected.

Selected shapes were then used as a constraint in a subsequent levelset segmentation.

11.6 Physically-Based Shape Models

It has been noted by the authors of the ASM that the amount of training data is sometimes insufficient for providing a reliable estimate for the shape model. They suggested applying an FEM for generating sufficient samples from its deformation (Cootes and Taylor 1995). As the number of training samples increases, the influence of the FEM part should decrease. FEMs are an example of a physically-based *deformable shape model* (DSM) in its own right. The main difference to ASMs is that deformation is not trained but uses domain knowledge.

A DSM is a non-rigid, explicit model with a priori constrained shape variation. Physically-based constraints have been presented, e.g., by Terzopoulos et al. (1987) and Pentland and Sclaroff (1991) and have been used in modeling anatomical objects, e.g., by Hamarneh et al. (2001). It should be noted that using a DSM for computer vision purposes poses two problems that do not exist if it is used for simulation of a true physical entity such as, e.g., the modeling of cloth described in Provot (1995):

- If used for the search of object instances, a DSM needs to be attracted by image features. Attraction forces have to be defined based on the image data. They can be thought as a gravitational force in some strange world. The developer of such a model needs to ensure that the behavior of a model instance in this world is still perceived as being somewhat natural.
- If model deformation shall capture variation between different object instances, this needs to be parameterized as part of the model based on domain knowledge. Parameterization may be difficult because variation among shapes of the same class is not caused by a physically induced deformation (consider a model of a person: shape differences of different persons are not due to some force exerted on a basic shape).

In view of the inaccuracies to be expected in model parameterization, applying a DSM for registering a model instance with an object instance requires sufficient redundancy between model and data. Inaccuracies in the model are then overridden by data information. This is required for any deformation model. It can never be assured that a model with a finite number of elements, components, or parameters captures all details that characterize an object in an image. Even if these details were captured, their exact description by the model representation cannot be guaranteed.

If DSMs are so difficult to use, it is fair to ask why they should be used at all. A major reason is that such models allow explicit specification of shape variation of an object without prior training. It restricts the efficient use[3] of such model to those objects whose variation is mainly described by decomposition into simpler model components. Adding the ability for variation without training is difficult—as it would be for a statistical model as well—but in a first approximation it can be done by overly relaxing variation constraints (for a statistical model this would translate into assuming a flat distribution function). Low image forces would cause large object deformations. Such a model is successful if the image information is "strong" enough in order to counteract overly relaxed variation constraints.

Some user involvement will always be necessary because user input extends the generalization ability of the model. A fully automatically acting model requires definition and parameterization that describes shape and appearance variation of all possible object instances completely and unambiguously. If this is possible at all, such a representation would be very specific to some application.

User involvement, as in all vision applications, should not merely accept or correct a result. Each interaction can be interpreted as additional information to the a priori knowledge about the object and should be included into the model.

Correction of a wrong result as well as training or teaching the model can be very straightforward in a DSM. A physically-based DSM, despite of the strangeness of the world of forces created by an image, reacts in a way that is familiar to the viewer.

[3]Each of the models presented here can be essentially extended such that it can replace any of the other models. However, each of the models has been built with some idea about the objects to be described and the way necessary knowledge is to be gathered. It works most efficiently if objects or scenes to which it is applied follow this idea.

If a result is not satisfactory or if the behavior while searching for the result is not correct, this familiarity allows the user to pinpoint elements of the model that should behave differently.

11.6.1 Mass-Spring Models

A mass spring model is a DSM that receives its shape by physical constraints from springs between mass points [used for instance for shape modeling in surgical simulation (Paloc et al. 2002) or for searching object instances in an image (Bergner et al. 2004), see Fig. 11.21a]. The DSM can be represented as a graph. Nodes with coordinates specified by vectors $\mathbf{n}_i$ are assumed to have masses m_i. Its edges are represented by springs $\mathbf{s}_{ij}$ connecting two mass points $\mathbf{n}_i$ and $\mathbf{n}_j$. The model—if placed in an image—moves towards an object of interest while keeping its shape controlled by external forces $\mathbf{f}_i^{\text{ext}}$ and internal forces $\mathbf{f}_i^{\text{int}}$. External forces attract the model towards image features while internal forces act between nodes and cause the model to retain its shape and size.

The net force acting on every node $\mathbf{n}_i$ is a weighted combination of external and internal forces:

$$\mathbf{f}_i = \alpha \mathbf{f}_i^{\text{ext}} + (1 - \alpha)\mathbf{f}_i^{\text{int}}. \tag{11.19}$$

The weight $0 < \alpha < 1$ indicates the importance of shape preservation and it is application-dependent. The behavior of a model instance over time t can be computed if node locations are made dependent on t. At time $t = 0$, every node has an initial location $\mathbf{n}_i(0)$ with velocity $\mathbf{v}_i(0) = 0$. Usually, initial node locations are set in such a way that internal forces are zero. The shape described by such configuration is the *prototype* of the model instance and represents the average shape of the object instance in the scene.

Given node locations $\mathbf{n}_i(t)$ and velocities $\mathbf{v}_i(t)$, node locations at time $t + \Delta t$ are

$$\mathbf{n}_i(t + \Delta t) = \mathbf{n}_i(t) + \Delta t \cdot \mathbf{v}_i(t). \tag{11.20}$$

Acceleration $\mathbf{a}_i$ at node $\mathbf{n}_i$ is computed from node mass m_i and force $\mathbf{f}_i$ as $\mathbf{a}_i = \mathbf{f}_i / m_i$. New velocities can now be computed:

$$\mathbf{v}_i(t + \Delta t) = \mathbf{v}_i(t) + \Delta t \cdot \mathbf{a}_i(t). \tag{11.21}$$

The time step Δt is application-specific. Small time steps increase the accuracy of the approximation of the differential equation. They also increase the number of iterations needed for the model to approach the object of interest.

In order to prevent oscillations, acceleration of nodes may be multiplied by a damping factor d which decreases with time t such as $d(t) = \exp(-\Delta t)$.

External forces are directed towards attraction points in an image. They may be thought as gravitational forces towards inert masses. Attraction forces acting on a

Fig. 11.21 **a** A mass-spring model consists of masses connected by springs. **b** Placed in an image, masses are attracted by image forces while springs are responsible for controlling the shape deformation

node of the DSM should decrease with increasing distance of the node from the attraction point (see Fig. 11.21b).

Every location $\mathbf{x}$ in an image may attract a model instance. Whether and to which extent it will be attracted by a particular location in the image depends on the image attribute $a(\mathbf{x})$ at this location. The length of the intensity gradient can be such an attribute, if model nodes shall be attracted by the object boundaries.

The force acting at $\mathbf{x}$ can be pre-computed by convolving $a(\mathbf{x})$ with a force kernel of which the magnitude increases with decreasing distance, e.g.,

$$k(\mathbf{x}) = \begin{cases} \frac{w}{\|\mathbf{x}\|^2}, & \text{if } \|\mathbf{x}\| \neq 0 \\ 0, & \text{otherwise.} \end{cases} \tag{11.22}$$

The value of w is a scale factor. The force vector for the external force $\mathbf{f}^{\text{ext}}$ exerted at location $\mathbf{x}$ is computed by convolving the force kernel with the attribute map and taking the gradient:

$$\mathbf{f}^{\text{ext}}(\mathbf{x}) = \nabla(a(\mathbf{x}) * k(\mathbf{x})). \tag{11.23}$$

The gradient vector field presented by Kichenassamy et al. (1995) and Xu and Prince (1998) detailed in Sect. 9.1.2 could be used as well.

The vector field can be pre-computed for every attribute field. The external force that pulls a node $\mathbf{n}_i$ is now (see Fig. 11.22):

$$\mathbf{f}_i^{\text{ext}} = \mathbf{f}^{\text{ext}}(n_i). \tag{11.24}$$

If a model instance is placed into a scene, each node will start to move towards its attractors. Since a moving mass possesses kinetic energy, it will pass the main

Fig. 11.22 A force field is a vector field that is computed from the gradient of image features. Masses of a model instance placed in the image will be attracted in direction of the force field vectors

Fig. 11.23 Spring forces and torsion forces govern the behavior of a mass-spring model. Spring forces act along the springs. They mainly restrict the change of scale. Torsion forces are angular forces with respect to a local, global or external coordinate system and restrict the deformation of the model instance

attraction zone and will move away from it until the external forces will start acting against kinetic forces pulling the mass back towards the object.

Moving further than the attraction point is intentional as it enables the model instance to find potentially better suited locations in the image. The damping factor mentioned above will cause the model instance to reach equilibrium after some oscillations.

Two kinds of internal forces govern the behavior of a stable mass-spring model: *spring forces* and *torsion forces* (see Fig. 11.23). The former mainly controls size changes of the model (Terzopoulos et al. 1987) and the latter restricts shape variation (Dornheim et al. 2005).

For computing spring forces, a rest length s_{ij}^0 needs to be assigned to each spring $\mathbf{s}_{ij}$. It is the length of a spring if the shape model takes its prototypical shape. If the distance between two nodes $\mathbf{n}_i$ and $\mathbf{n}_j$ along the spring $\mathbf{s}_{ij}$ deviates from the rest length, it creates a spring force $\mathbf{f}_{ij}^s$ acting in the direction of $\mathbf{s}_{ij}$ (this is under the assumption that $m_i = 1$ for each mass point):

$$\mathbf{f}_{ij}^{s} = \left[k_{ij}\left(s_{ij}^{0} - \left\|\mathbf{n}_i - \mathbf{n}_j\right\|\right)\right]\frac{\mathbf{s}_{ij}}{\left\|\mathbf{s}_{ij}\right\|}. \tag{11.25}$$

The parameter k_{ij} is the spring constant which describes the spring's stiffness. An effective net force acting on a node $\mathbf{n}_i$ is a sum of forces from all $K(i)$ springs $\mathbf{s}_{ik}$, $k = 1, K(i)$ that are connected to $\mathbf{n}_i$:

$$\mathbf{f}_{i}^{s} = \sum_{k=1}^{K(i)} \mathbf{f}_{ik}^{s}. \tag{11.26}$$

Mass-spring models using spring forces to control the model shape do not necessarily prevent deformations. A mass-spring model usually has many stable states of which the prototype shape is just one. Hence, the kinetic energy of a moving model may cause a model instance to converge to any of the other stable states. Adding stabilizing and almost rigid springs can solve this problem but will result in a heavily connected model whose behavior will not be easily predictable.

Stiffness control can be explicitly added as additional force to the node network. The force inhibits the relative rotation between nodes and controls the spatial relationship between all springs connected to a single node. It is called *torsion force* and restricts the twisting movement by rotating nodes into the rest location (see Fig. 11.24). This local force is implicitly defined by the shape prototype.

Introduction of the torsion force reduces the number of stable states of the model to just the one which corresponds to the prototype shape. In order to compute and apply the torsion force, springs connected to a node $\mathbf{n}_i$ need to be defined in a node coordinate system that may be one of three kinds (see Fig. 11.23):

- A fixed *external coordinate system* keeps the orientation of the model stable. This is simply the world coordinate system which is assigned to every node.
- A *global model coordinate system* keeps the shape of the object stable but allows for object rotations. It needs to be rotated appropriately after each iteration.

Fig. 11.24 Model deformation without and with torsion forces [based on the work of Dornheim et al. (2005)]

- A *local point coordinate system* has similar properties than a global model coordinate system but allows for local variation of acceptable twisting of the model instance. It is computed from spring directions that are incident to a mass point.

Given a node coordinate system, a spring vector $\mathbf{s}_{ij}$ in world coordinates can also be represented as a vector $\mathbf{s}_{ij}^{i}$ in the node coordinates of node $\mathbf{n}_i$. If the model has its prototypical shape, all spring directions in node coordinates attain their rest directions $\mathbf{s}_{ij}^{i,0}$. If the shape is deformed, actual spring directions $\mathbf{s}_{ij}^{i}$ may vary from their rest directions. The torsion force acting on node $\mathbf{n}_i$ is then a rotational force relative to the node $\mathbf{n}_j$ that attempts to rotate spring $\mathbf{s}_{ij}$ around $\mathbf{n}_j$ back to its rest orientation. It is defined as:

$$\mathbf{f}_{ij}^{t} = t_{ij} \cdot \frac{\Delta \mathbf{s}_{ij}^{i}}{\left\| \Delta \mathbf{s}_{ij}^{i} \right\|}, \tag{11.27}$$

where t_{ij} depends on the angle between rest position and actual position of the spring and $\Delta \mathbf{s}_{ij}^{i}$ determines the direction of the force. The value of t_{ij} is:

$$t_{ik} = t_i \cdot \angle\left(\mathbf{s}_{ik}^{i}, \mathbf{s}_{ik}^{i,0}\right). \tag{11.28}$$

The factor t_i is a torsion constant governing the local stiffness at node $\mathbf{n}_i$. The direction in which the force is acting is given by:

$$\Delta s_{ij}^{i} = s_{ij}^{i,0} - s_{ij}^{i} \left(\frac{s_{ij}^{i} \cdot s_{ij}^{i,0}}{\left\| s_{ij}^{i} \right\| \cdot \left\| s_{ij}^{i,0} \right\|} \right), \tag{11.29}$$

where "•" is the inner product.

In order to compute node displacement due to torsion, the sum of torsion forces of all springs $k = 1, K(i)$ that are connected to $\mathbf{n}_i$ has to be computed:

$$\mathbf{f}_{i}^{t} = \sum_{k=1}^{K(i)} \mathbf{f}_{ik}^{0}. \tag{11.30}$$

Since $\mathbf{s}_{ij}^{i,0}$ remains constant with respect to the local coordinate system of node $\mathbf{n}_i$, but varies with respect to the world coordinate system, it has to be computed after each iteration. The rotation of the node coordinate system with respect to the world coordinate system can be determined by averaging the rotations caused by all springs that are connected to this node. Combining spring forces and torsion forces results in an effective internal force $\mathbf{f}_{i}^{\text{int}}$ acting on each node i:

$$\mathbf{f}_i^{\text{int}} = (1-\beta)\mathbf{f}_i^s + \beta\mathbf{f}_i^t. \tag{11.31}$$

The factor β with $0 \leq \beta \leq 1$ governs the relative influence of spring and torsion forces.

External forces attracting a model instance to object boundaries create a complex world of attractions because non-zero intensity gradients exist almost everywhere in the image. A model instance may only be attracted to the "correct" location (i.e., by the object instance in search) if it is placed close to this location. For automatic segmentation by a DSM the most likely location of the object instance needs to be determined beforehand.

If shape or appearance of other objects differs substantially from that of the object instance, a correlation measure between average model instance and image may be used. Otherwise, a stochastic search may be enacted where model instances are placed at several locations in the image. Decisions as to whether such instance has found an object will be based on a quality-of-fit measure of the fitted model instance. This could be, for instance, the magnitude of the sum of internal forces in the model (i.e., a deformation cost) or the sum of gradient lengths at node locations (i.e., the support by the image), or a combination of the two. Successful candidates of the stochastic search may spawn new model instances in their vicinity until the average QoF of the population of model instances no longer increases.

11.6.2 Finite Element Models

Finite element models [FEMs, see, e.g., Zienkiewics et al. (2005), Petyt (1998) for an introductory treatment] are an alternative way to solve the equilibrium problem when placing a deformable shape model instance into an image and letting it deform according to external forces (Sclaroff and Pentland 1995). The conceptual difference between a mass-spring model and an FEM is that the FEM is a continuous space model that may be defined for each location $\mathbf{x}$ in the model, whereas the mass-spring model is only defined at the mass points (see Fig. 11.25).

Finite element models are often used in the material sciences. They approximate the behavior of a continuous physical entity by means of a finite number of elements. Each element is bounded by nodes. The behavior of the represented entity under the influence of forces is reduced to the influence at these nodes. Elements of an FEM may be 1d, 2d or 3d and they may be embedded in 2d or 3d space. It should be kept in mind that the FEM controls the degrees of freedom of shape deformation only within its elements. If, for instance, the boundary of a 3d object is represented by a 2d FEM, this restricts the shape of the bounded object only in a very indirect fashion.[4] However, if the dimensionality of the elements equals that of the space in which they are embedded, the FEM represents physical behavior at any

[4]A FEM may be defined in a similar fashion than the mass-spring model by letting 1d springs being the elements. In such case, a bounded 2d or 3d object may be represented by a dense mesh of springs restricting shape variation.

Fig. 11.25 A mass-spring model transfers forces only along springs, while an FEM is a continuous space model where forces are defined everywhere in the elements

Fig. 11.26 **a** Shape functions $N(x, y)$ describe the influence of forces acting at some node on all other places in the element. The example on the left shows a linear shape function for node n_1. The deformation of the element is then computed from the influence of forces acting on all nodes

location **x** in a bounded region covered by the elements. Shape deformation is controlled by the decomposition into elements and their physical attributes.

FEMs have been used to represent a deformable object boundary (Delingette et al. 1992; Mandal et al. 1998) and to describe deformable object shapes (Ferrant et al. 2000; Engel and Toennies 2010). The former is a topological constraint restricting the number of closed boundaries, while the latter constrains object shapes. We will restrict our discussion to FEMs constraining the object shape.

Given an element of the FEM and external forces acting on it, it deforms until potential energy of the element (strain and stress) and external forces balance. Strain and stress within the element are interpolated by *shape functions* such that they result in a continuous displacement field for this element (see Fig. 11.26). Interpolation guarantees that displacement at node locations does not depend on other nodes. This is necessary for a continuous displacement field of a mesh of connected elements.

Potential energy usually describes the element as elastically deformable object. This imposes a smoothness constraint on element deformation. Together with the continuity of a displacement field, it ensures a kind of stability of the shape representation that is similar to that achieved by torsion forces in the mass-spring model.

The following is a more detailed explanation of this application using a triangular grid as example. A triangular grid would be appropriate to represent shape and its variations of a 2d figure. The triangle as basic element of this FEM consists

of three bounding nodes $\mathbf{n}_1$, $\mathbf{n}_2$, $\mathbf{n}_3$ with coordinates $(x_{i,1}, x_{i,2})$, $i = 1,3$. A force acting on a node $\mathbf{n}_i$ is represented by a 2d vector $\mathbf{f}_i = (f_{2i}, f_{2i+1})$. It causes displacement of $\mathbf{n}_i$ by $\mathbf{u}_i = (u_{2i}, u_{2i+1})$ until internal forces from elasticity of the material and stress caused by the deformation balance the external force.[5] This can be expressed by the following relationship for an element e:

$$\mathbf{u}^{(e)}\mathbf{K}^{(e)} + \mathbf{f}^{(e)} = 0 \tag{11.32}$$

The (e) customarily indicates that we are talking about a single element of an FEM.

The matrix $\mathbf{K}^{(e)}$ is called stiffness matrix. It summarizes material-specific elasticity and stress components. The equation above only defines a displacement for nodes. Displacement for all other locations within the element is interpolated using shape functions $\boldsymbol{\phi}(\mathbf{x})$ for which $\boldsymbol{\phi}(\mathbf{n}_i) = \mathbf{u}_i$ must be true.

If finite elements are combined to a mesh, a common stiffness matrix has to be assembled from the element matrices. The following two conditions must hold for assemblage of any two elements of a mesh to keep displacements within the mesh continuous:

1. The displacements at a common node of two elements must be the same for each element.
2. The sum of all forces exerted from elements meeting at a common node must balance the external force at this node.
3. The interpolated displacement at common element boundaries must be C_0-continuous.

The two conditions can be guaranteed by the following, simple assemblage step. If components of all N nodes in an assembled mesh are indexed from 1 to $2N$ and node features of an element e of this mesh have indices $n_1, n_2, \ldots, n_6$ with a corresponding 6×6 stiffness matrix $\mathbf{K}^{(e)}$, an extended stiffness matrix $\mathbf{K}^{(\text{ext})}$ of size $2N \times 2N$ is created with

$$K_{k,l}^{(\text{ext})} = \begin{cases} K_{i,j}^{(e)} & \text{if } k = n_i \wedge l = n_j \wedge i,j = 1,2,3,4,5,6 \\ 0 & \text{otherwise.} \end{cases} \tag{11.33}$$

This is done for all element matrices (see Fig. 11.27). The extended matrices are added. The result is an assembled stiffness matrix $\mathbf{K}$ for the mesh[6] fulfilling the two conditions above.

[5]Node values and force values of all nodes of an element (and later of the complete FEM mesh) are combined in a single vector. Hence, values for a node with index i in 2-d have indices $2i$ and $2i + 1$ in the displacement vector $\mathbf{u}$ and the external force vector $\mathbf{f}$.

[6]This kind of assemblage becomes costly for the sparse matrix $\mathbf{K}$ if N is large. Faster methods to carry out this operation exist but the operation itself stays the same. For application in medical image analysis, however, the size of N is usually small (i.e., $N \ll 100.000$) and model creation does not happen often.

Fig. 11.27 Example for an assemblage. A common matrix of the size of the total number of nodes is created and entries from the element matrices are set at the appropriate positions in the common matrix

Fig. 11.28 A dynamic FEM deforms depending on external image forces towards the sought object

The displacement of nodes of an FEM mesh requires the solution of the linear equation system $\mathbf{uK} = \mathbf{f}$ for the displacement vector $\mathbf{u}$. Additional constraints are needed, however, because the mesh may still freely move in the 2d space. These constraints are called *essential boundary conditions* and *natural boundary conditions* in the FEM literature. Essential boundary conditions restrict the displacement vector $\mathbf{u}$ (e.g., by stating that $(u_{2i}, u_{2i+1}) = 0$ for some nodes $\mathbf{n}_i$). Natural boundary conditions restrict the force vector. Given a sufficiently constrained problem, the displacement of an FEM mesh under an external force can be computed by solving the linear equation system for $\mathbf{u}$.

So far, the FEM is static. The concept is extended to include dynamics for modeling external image forces attracting a model instance to depicted objects (see Fig. 11.28). The influence of external forces depends on the current placement of the model in the scene. Since it changes while the model instance moves, displacement $\mathbf{u}$ and forces $\mathbf{f}$ are made time-dependent. External forces in the dynamic FEM attract the nodes to image regions of relevance, and are balanced against the topology-preserving internal mesh forces. This is represented by the linear dynamic model

$$\mathbf{M}\ddot{\mathbf{u}}(t) + \mathbf{D}\dot{\mathbf{u}}(t) + \mathbf{K}\mathbf{u}(t) + \mathbf{f}(t) = 0. \tag{11.34}$$

The nodal displacements $\mathbf{u}(t)$ depend on time t. The terms $\dot{\mathbf{u}}$ and $\ddot{\mathbf{u}}$ are the first and second derivative of the displacement with respect to time (i.e. velocity and acceleration). Acceleration depends on a mass-specific inertia represented by the mass matrix $\mathbf{M}$. The matrix $\mathbf{D}$ models velocity-dependent damping.[7] If the viscosity of the modeled material is unknown, the damping matrix $\mathbf{C}$ can be taken as *Raleigh damping* $\mathbf{D} = \alpha\mathbf{M} + \beta\mathbf{K}$. $\mathbf{K}$ is the stiffness matrix already known from static FEMs.

The mass and stiffness matrices restrict the variation of the DSM. Hence, these material properties encode the domain knowledge about the object variance. Using an FEM to describe inter-subject variation of organs and structures does not require the description of a specific material since this is not a simulation that describes real deformation processes. Material properties represent an abstract deformation concept.

Stiffness $\mathbf{K}^{(e)}$ of an element e is computed from

- a stress matrix $\mathbf{B}^{(e)}$,
- an elasticity matrix $\mathbf{C}^{(e)}$,
- the element thickness k,
- the area A of the element.

For an element of constant thickness (which is usually assumed when using finite elements in object detection and segmentation), stiffness is

$$\mathbf{K}^{(e)} = kA\left(\mathbf{B}^{(e)}\right)^{\mathrm{T}}\mathbf{C}^{(e)}\mathbf{B}^{(e)}. \tag{11.35}$$

Stress represents the deformation under (initial) forces such as a shrinking due to a change in temperature. The influence of stress can be computed for arbitrary directions by a stress tensor that describes translation and shearing forces. For a 2d element, this is

$$\varepsilon = S\mathbf{u} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \begin{pmatrix} u \\ v \end{pmatrix} \approx S\mathbf{N}\tilde{u}^{(e)} = \mathbf{B}^{(e)}\tilde{u}^{(e)}. \tag{11.36}$$

Hence, the stress matrix results from the shape functions in $\mathbf{N}$ and the shape of the elements. Since shape functions are usually fixed and simple when using FEM to model a DSM, the component is not used to vary deformation constraints for the DSM. This is different for elasticity. Elasticity describes the deformation of the element under some external force. For a linear-elastic homogeneous body, elasticity can be described by the elasticity modulus E and the Poisson ratio ν. It is

[7] A dynamic system can be modeled without damping but damping prevents oscillation.

$$\mathbf{C}^{(e)} = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & (1-\nu)/2 \end{bmatrix}. \tag{11.37}$$

The Poisson ratio is the ratio $\nu = -e/e_{\perp}$ by which an object expands or shrinks orthogonal to the direction of the impacting force (see Fig. 11.29). In other words, it tells how much of the acting force is transferred sideways. The Poisson ratio may vary between −1 and 0.5. It is positive for most materials (a negative Poisson ratio would mean that a force causing a shrinking in one direction would also cause shrinking in directions orthogonal to it).

The elasticity modulus E describes how much permanent deformation is caused by an acting force. A sufficient description for isotropic materials is the use of Young's modulus which is a single scalar value and assumes that the deformation is independent of the direction in which the force acts on the material.

The mass matrix $\mathbf{M}$ is the second component that describes the behavior of the FEM. It describes the inertia of the system with respect to external forces. The mass matrix $\mathbf{M}^{(e)}$ of an element e depends on the material density ρ and the shape functions $\mathbf{N}$

$$m_{i,j} = \int_{\Omega_e} \mathbf{N}_i^{\mathrm{T}} \rho \mathbf{N}_j \mathrm{d}\Omega_e. \tag{11.38}$$

The mass matrix for a 2d element with area A, thickness k, material density ρ, and linear shape functions is

$$\mathbf{M}^{(e)} = \frac{\rho k A}{12}\begin{pmatrix} 2 & 0 & 1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 1 & 0 & 1 \\ 1 & 0 & 2 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 & 0 & 1 \\ 1 & 0 & 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 1 & 0 & 2 \end{pmatrix}. \tag{11.39}$$

Fig. 11.29 The Poisson ratio ν describes the deformation orthogonal to the direction of the incident force

Given mass, damping and stiffness matrices concatenated from their respective element matrices, external forces cause the model to move and deform. The global force vector $\mathbf{f}(t)$ contains dynamic loads that attract the FEM towards image features. Similar to the use of mass-spring models for segmentation and registration, these features need to be attributes $a(\mathbf{x})$ that are correlated to features of sought objects. Examples are the image gradient or image intensity. The external force at time t exerted on nodes $\mathbf{n}_i$, $i = 1, \ldots, N$ with current position $\mathbf{x}_i(t)$ is:

$$\mathbf{f}_i(\mathbf{x}, t) = \lambda(t) \nabla a[\mathbf{x}(t)], \tag{11.40}$$

where $\lambda(t)$ is a weighting factor that prevents oscillation similarly to the damping factor in a mass-spring model.

Dynamic displacement can be computed by solving the system of partial differential equations iteratively, given that initial displacement and displacement speed are known. A standard solution technique is the *Newmark-β algorithm*. The general principles behind this algorithm are

- Differentials are approximated by differences in a sequence $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n, \mathbf{u}_{n+1}, \ldots$ that are Δt apart.
- The state of the system at time t_{n+1} is given by a Taylor approximation of $\mathbf{u}$ at time t_{n+1}.
- The remainder term of the Taylor series is replaced by a weighted difference between differentials at t_n and t_{n+1} (the weight is the name-giving β).
- The second derivative of $\mathbf{u}$ is computed first, then it is used to compute the first derivative, which in turn is used to compute $\mathbf{u}$ at time t_{n+1}.

In order to derive the Newmark-β algorithm we follow the argumentation of Zienkiewics et al. (2005) and start with a simplified FEM without mass matrix $\mathbf{M}$ (which describes a so-called transient field used, e.g., to describe heat conduction). The solution will then be extended to a complete FEM without detailed derivation. A transient field is given by

$$\mathbf{D}\dot{\mathbf{u}}(t) + \mathbf{K}\mathbf{u}(t) + \mathbf{f}(t) = 0. \tag{11.41}$$

Approximating $\mathbf{u}$ by a Taylor approximation (see Fig. 11.30) results in the finite difference solution $\mathbf{u}_{n+1} = \mathbf{u}_n + \Delta t \cdot \dot{\mathbf{u}}_n$. However, this extrapolates the function based on an extrapolation of the derivative. The error from omitting the remainder term of the Taylor series is not accounted for. If the function $\mathbf{u}(t)$ is sufficiently differentiable, a ratio between $\dot{\mathbf{u}}_n$ and $\dot{\mathbf{u}}_{n+1}$ exists that exactly represents the average derivative between t_n and t_{n+1}. Hence, interpolation of this value produces a more stable solution than extrapolating from t_n (see Fig. 11.31). The value of $\mathbf{u}_{n+1}$ is now approximated by

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \Delta t[(1-\beta)\dot{\mathbf{u}}_n + \beta\dot{\mathbf{u}}_{n+1}], \tag{11.42}$$

Fig. 11.30 The finite difference solution is inherently instable, as it extrapolates the displacement $\mathbf{u}_{n+1}$ from known values $\mathbf{u}_n$ at time t_n

Fig. 11.31 Using derivatives at time t_n and t_{n+1} produces a more stable solution for estimating $\mathbf{u}_{n+1}$

where β is a weighting parameter that has to be set. Approximating Eq. (11.41) results in

$$\begin{aligned}
&\mathbf{D}\dot{\mathbf{u}}_{n+1} + \mathbf{K}\mathbf{u}_{n+1} + \mathbf{f}_{n+1} = 0 \\
&\mathbf{K}\mathbf{u}_{n+1} = \mathbf{K}(\mathbf{u}_n + \Delta t[(1-\beta)\dot{\mathbf{u}}_n + \beta\dot{\mathbf{u}}_{n+1}]) \\
&\Rightarrow \mathbf{D}\dot{\mathbf{u}}_{n+1} + \mathbf{K}(\mathbf{u}_n + \Delta t[(1-\beta)\dot{\mathbf{u}}_n + \beta\dot{\mathbf{u}}_{n+1}]) + \mathbf{f}_{n+1} = 0.
\end{aligned} \tag{11.43}$$

Except for $\mathbf{f}_{n+1}, \dot{\mathbf{u}}_{n+1}$ only depends on quantities that are known at time t_n. If $\mathbf{f}_{n+1}$ is approximated by forward differencing, $\mathbf{u}_{n+1}$ is given by rearranging the terms and resolving for $\dot{\mathbf{u}}_{n+1}$:

$$\dot{\mathbf{u}}_{n+1} = -(\mathbf{D} + \Delta t\beta\mathbf{K})^{-1}[\mathbf{K}(\mathbf{u}_n + \Delta t(1-\beta)\dot{\mathbf{u}}_n) + \mathbf{f}_{n+1}]. \tag{11.44}$$

From $\dot{\mathbf{u}}_{n+1}$ the value for $\mathbf{u}_{n+1}$ can be computed based on Eq. (11.42).

The system is not self-starting, i.e., $\mathbf{D}\dot{\mathbf{u}}_0 + \mathbf{K}\mathbf{u}_0 + \mathbf{f}_0 = 0$ needs to be guaranteed. The values of the external forces $\mathbf{f}_{n+1}$ need to be computable (e.g., by another Taylor approximation).

The choice of β influences the behavior of the algorithm. Choosing $\beta = 0$ results in a finite difference approximation. In the absence of domain knowledge $\beta = 0.5$ is a usual choice. Since stability of the solution cannot be guaranteed without

knowledge about the difference between the chosen β and the unknown optimal β, the time step Δt should be rather small.

The Newmark-β algorithm can be extended to FEMs with mass matrix. The derivation strategy is the same except for the fact that the expression gets longer. The resulting scheme for computing $\mathbf{u}$ and its derivatives at time t_{n+1} is

$$\ddot{\mathbf{u}}_{n+1} = \mathbf{A}^{-1}\left(\mathbf{D}[\dot{\mathbf{u}}_n + (1-\beta_1)\Delta t\ddot{\mathbf{u}}_n] + \mathbf{K}\left[\mathbf{u}_n + \Delta t\dot{\mathbf{u}}_n + \frac{1}{2}(1-\beta_2)\Delta t^2\ddot{\mathbf{u}}_n\right] + \mathbf{f}_{n+1}\right) \tag{11.45}$$

with

$$\mathbf{A} = \mathbf{M} + \beta_1\Delta t\mathbf{D} + \frac{1}{2}\beta_2\Delta t^2\mathbf{K} \tag{11.46}$$

$$\dot{\mathbf{u}}_{n+1} = \dot{\mathbf{u}}_n + (1-\beta_1)\Delta t\ddot{\mathbf{u}}_n + \beta_1\Delta t\ddot{\mathbf{u}}_{n+1} \tag{11.47}$$

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \Delta t\dot{\mathbf{u}}_n + \frac{1}{2}(1-\beta_2)\Delta t^2\ddot{\mathbf{u}}_n + \frac{1}{2}\beta_2\Delta t^2\ddot{\mathbf{u}}_{n+1}. \tag{11.48}$$

An alternative to the direct solution is to decouple the equations by transforming them from the space of displacements $\mathbf{u}$ into the space of *free vibration modes* $\mathbf{v}$. Free vibration modes are the solution of a free vibration problem without damping. In such case

$$\mathbf{M}\ddot{\mathbf{u}}(t) + \mathbf{K}\mathbf{u}(t) = 0 \tag{11.49}$$

for $t > 0$. The equation system describes observed motion of an FEM mesh that is put into motion by an impulse at $t = 0$. Being a homogeneous linear differential equation system, solutions must be a sum of weighted periodic functions (see Fig. 11.32):

$$\mathbf{u}(t) = \sum_{j=1}^{2N}\mathbf{v}_j\exp\left(i\omega_j t\right). \tag{11.50}$$

Replacing $\mathbf{u}(t)$ in (11.49) by (11.50) and computing the second derivative $\ddot{\mathbf{u}}$ results in

$$\begin{aligned} &\mathbf{M}\sum_{j=1}^{2N} -\omega_i^2\mathbf{v}_j\exp\left(i\omega_j t\right) + \mathbf{K}\sum_{j=1}^{2N}\mathbf{v}_j\exp\left(i\omega_j t\right) = 0 \\ \Leftrightarrow &\sum_{j=1}^{2N}\mathbf{K}\mathbf{v}_j\exp\left(i\omega_j t\right) = \sum_{j=1}^{2N} -\omega_i^2\mathbf{M}\mathbf{v}_j\exp\left(i\omega_j t\right). \end{aligned} \tag{11.51}$$

Fig. 11.32 Free vibration of an FEM

Comparing factors on both sides and noting that the periodic term $\exp(i\omega_i t)$ is always non-zero, it can be seen that $2N$ solutions of pairs ($\boldsymbol{\omega}$, $\mathbf{v}$) to the following equation

$$\mathbf{K}\mathbf{v} = \boldsymbol{\omega}^2\mathbf{M}\mathbf{v} \Leftrightarrow \mathbf{M}^{-1}\mathbf{K}\mathbf{v} = \boldsymbol{\omega}^2\mathbf{v}. \tag{11.52}$$

are searched. Hence, the $2N$ solutions are the eigenvectors of $\mathbf{M}^{-1}\mathbf{K}$ and the corresponding eigenvalues in $\boldsymbol{\omega}^2$.

This does not only solve the free vibration problem of Eq. (11.49). It also provides the means to solve Eq. (11.34) by a set of decoupled partial differential equations, if Raleigh damping $\mathbf{D} = \alpha\mathbf{M} + \beta\mathbf{K}$ is assumed. In this case, diagonalization of $\mathbf{M}$ and $\mathbf{K}$ also diagonalizes all linear combinations.

The equation is transformed into the space of vibration modes. A matrix $\boldsymbol{\Omega}$, containing eigenvectors that are normalized such that $\boldsymbol{\Omega}^{\mathrm{T}}\mathbf{M}\boldsymbol{\Omega} = \mathbf{I}$, is created. Eigenvalues ω_i^2 are the values of a diagonal matrix $\boldsymbol{\Lambda}$. Transformation in the vibration modes produces a new system of decoupled equations

$$\widehat{\mathbf{M}}\ddot{\mathbf{v}}(t) + \widehat{\mathbf{D}}\dot{\mathbf{v}}(t) + \widehat{\mathbf{K}}\mathbf{v}(t) = \mathbf{Q}(t) \tag{11.53}$$

with the diagonal matrices

$$\widehat{\mathbf{M}} = \boldsymbol{\Omega}^{\mathrm{T}}\mathbf{M}\boldsymbol{\Omega} = \mathbf{I} \tag{11.54}$$

$$\widehat{\mathbf{K}} = \boldsymbol{\Omega}^{\mathrm{T}}\mathbf{K}\boldsymbol{\Omega} = \boldsymbol{\Lambda} \tag{11.55}$$

$$\widehat{\mathbf{D}} = \alpha\mathbf{I} + \beta\boldsymbol{\Lambda} \tag{11.56}$$

and the transformed force vector $\mathbf{Q}(t) = \boldsymbol{\Omega}^{\mathrm{T}}\mathbf{f}(t)$.

The now decoupled equations can be brought in the following standard form for differential equations

$$\ddot{v}_i + 2\omega_i\varsigma_i\dot{v}_i + \omega_i^2 v_i + f_i = 0 \tag{11.57}$$

by dividing each equation by the transformed mass factor $\hat{m}_i$ and letting

$$2\omega_i\varsigma_i = \frac{\hat{c}_i}{\hat{m}_i}, \quad \omega_i^2 = \frac{\hat{k}_i}{\hat{m}_i}, \quad f_i = -\frac{q_i}{\hat{m}_i}. \tag{11.58}$$

The analytical solution for this differential equation is

$$\begin{aligned} y_i(t) =& \exp(-\xi_i\omega_i t)\left[\frac{\dot{y}_i(0) + \xi_i\omega_i y_i(0)}{\bar{\omega}_j}\sin\bar{\omega}_i t + y_i(0)\cos\bar{\omega}_i t\right] \\ &+ \frac{1}{\bar{\omega}_i}\int_0^t \exp(-\xi_i\omega_i(t-\tau))\sin\bar{\omega}_i(t-\tau)f_i(\tau)d\tau \end{aligned} \tag{11.59}$$

with

$$\bar{\omega}_j = \omega_j\sqrt{1-\xi_j^2} \ (\xi \text{ is a damping constant}) \tag{11.60}$$

and initial conditions $y_i(0) = \mathbf{v}_i^{\mathrm{T}}\mathbf{M}\mathbf{u}_i(0)$ and $\dot{y}_i(0) = \mathbf{v}_i^{\mathrm{T}}\mathbf{D}\underline{\mathbf{u}}_i(0)$.

The solution does not contain derivatives and the approximation of the integrals can be done using numerically stable methods. Hence, this solution is preferred if instabilities can be expected because computing time constraints requires long time intervals or if the system is meant to be simulated over a long time.

Eigenvectors and eigenvalues carry semantics similar to ASM. An eigenvector is called a *mode of vibration* and represents a generalized symmetry axis of shape deformation. The major difference is that *variation modes* are derived from trained variation, whereas *vibration modes* result from material properties and decomposition of a shape.

The overall deformation is a weighted sum of the generalized deformations due to the transformed forces $\mathbf{\Omega}^{\mathrm{T}}\mathbf{f}(t)$. It is possible to reduce the number of modes in order to remove small information details and reduce computational costs. Eigenvalues characterize the amount of force necessary to cause displacement. Assuming ascending order of eigenvalues, the first $N!$ eigenvalues of an N-dimensional FEM will be zero and represent rigid transformation (rotation and translation). The following, intermediate eigenvalues are the most relevant modes representing shape deformation.

Rigid transformation is still part of the model, which is different to ASM where the trained model is normalized with respect to rotation and translation and where this information has to be incorporated into the fitting process by means of an additional registration step. This causes problems, however, since deformation constrained by the stiffness matrix $\mathbf{K}$ is defined as strain caused by a directed force on an infinitesimal line at this point. Since this line is defined in the given coordinate system, rotation of the FEM instance would make the stiffness matrix $\mathbf{K}$ dependent on the current rotation, hence dependent on time t. This is unwanted since it would require re-computation of $\mathbf{K}$ and of the vibration modes at every new iteration step. Neglecting it would cause some part of the rotation to be interpreted

Fig. 11.33 If the rotation of the reference coordinate system for the stiffness matrix is neglected, serious distortions are the result. Deformation modes of a stick model without (*upper rows*) and with warping

as deformation possibly leading to serious distortions (see Fig. 11.33). This is only acceptable if we assume that local fitting of the model instance comprises only little rotation. Otherwise, warping techniques, as presented, e.g., by Müller and Gross (2004) or Choi and Hyeong-Seok (2005) can be used to circumvent the problem.

The basic idea of warping is to include the current rotation into the representation. Therefore, forces are applied to the un-rotated model and the result is rotated back. If warping is applied on the node level, the current rotation $\mathbf{R}_i$ is computed for all nodes i. They are concatenated into an orthogonal matrix $\mathbf{R}$ of which its inverse is applied to the current displacement $\mathbf{u}(t)$ as well as to its derivatives. Deformation is then applied to these vectors and the result is rotated back (the impact of this correction can be seen in Fig. 11.33). It can be shown that this can be applied in the spectral basis as well (Choi and Hyeong-Seok 2005). This essentially means that the current vibration modes are "rotated" versions of the original modes.

The current rotation can be computed by several methods. A fast way is to search for the rotation $\mathbf{R}_i$ at node coordinates $\mathbf{x}_i$ that optimally registers difference vectors $\mathbf{d}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ between $\mathbf{x}_i$ and adjacent nodes $\mathbf{x}_j$ relative to difference vectors $\mathbf{d}_{ij0}$ at $t = 0$. The resulting optimization problem

$$\mathbf{R}_i = \arg\min_{\mathbf{R}(t)} \sum_{j=1}^{M} \left\| \mathbf{d}_{ij}(t) - \mathbf{R}(t)\, \mathbf{d}_{ij0} \right\| \tag{11.61}$$

for each node can be computed fast using a method proposed in Byers and Xu (2008).

Material properties are often set constant throughout the figure described by the FEM. As already mentioned earlier, the FEM can be combined with an ASM (Cootes and Taylor 1995). A shape model is constructed as FEM that "produces" training data for an ASM. Thus, the modes of variation of the initial model are actually modes of vibration of the FEM. If the model is used for some application, each confirmed segmentation result is considered to be a training sample. The FEM behavior is gradually replaced by the trained statistics of an ASM.

Vibration modes are object-specific symmetries that represent intrinsic properties of the object. This was used by Sclaroff and Pentland (1995) to create an object-specific intrinsic coordinate system for registering instances of the same object class. Vibration modes are ordered according to their variance (the eigenvalues) and separated into three groups. The highest-variance modes represent rigid transformations (since the largest total displacement from a given external force is that by a rigid transformation). In the case of 2d models these are the first three modes representing translation in x and y as well as rotation around the origin. They do not carry information about object-specific deformation.[8] The next highest set of modes is then used to represent intrinsic object-class variation. Similar to the strategy in ASM, vibration modes with low eigenvalues are not considered. Most likely they are severely influenced by inaccuracies from model generation.

Differences between two shapes that are represented by different deformations of the same FEM can now be modelled by modal displacement in the truncated eigenspace (i.e. without the rigid transformation components and low-order modes). Corresponding locations for producing the match in the original space can be reconstructed by projecting the modal amplitudes back using the truncated matrix of eigenvectors.

Similar to ASMs, the deformation in modal amplitudes provides an estimate of the similarity of an object instance with the model. In object detection, the quality of fit of an FEM instance is defined by the deformation. Together with a data term that rates the local appearance in the vicinity of an FEM instance in the image, a weighted QoF measure can be computed.

This can be used for a stochastic search and detection algorithm. FEM instances are placed with arbitrary pose parameters into the image. They are allowed to deform and displace under image forces. The QoF is computed after the instance no longer moves. Pose parameters of the best fitting instances are varied and the process is repeated until the average quality-of-fit of the population no longer improves. Instances with QoF values exceeding some threshold are then detections of object instances. In conjunction with a hierarchical model (see below), this

[8]These modes are not contained in an ASM, since influence from rotation and translation is removed during normalization of the training data.

approach has been shown to be very successful for object detection of single objects (Engel et al. 2011) or an unknown number of objects (Engel and Toennies 2009).

In classification, a similar strategy is employed. Instead of using a single model, FEMs for different classes are used for the search. Classification is then done by comparing the QoF values of different FEMs.

Almost any kind of shape variation under external forces can be modeled by appropriately varying material attributes of the underlying physical model, but this puts a heavy burden on the developer of the model. Furthermore, the model has to be adapted for every object class. Hence, the original objective for introducing a physically-based model—to replace training by a less costly procedure—is no longer achieved.

If expected shape variation is too complex, a solution is to resort to shape hierarchies (Engel et al. 2011). This is another kind of part-based model such as the ASSM described in Sect. 11.5.1. A shape is decomposed into sub-units and each sub-unit is modeled by an FEM mesh. Relationships among sub-units are modeled by a higher-level FEM. This FEM describes sub-units by a number of nodes that are sufficient to represent relationships among sub-units (see Fig. 11.34).

If, for instance, two sub-units are related to each other by an expected average distance between them, a single node per sub-unit is sufficient to represent the location of the sub-unit. If relationships are more complex, e.g., if sub-units are supposed to be nearly parallel to each other with an expected distance between them, this can be represented as well. High-level nodes representing sub-units are related to the FEM representation at the lower level via essential boundary conditions (i.e., the low-level FEM restricts the displacement of its high-level representation).

Different to the ASSM, the hierarchical FEM does not require an initial part from which the existence, pose and shape of the other parts is inferred. Randomly distributed instances of the complete model are placed in the image and adapt locally

Fig. 11.34 Different kinds of second-layer FEMs represent different kinds of spatial relationships between components (**a** distance, **b** approximate parallelism)

according to image forces (as for the FEM model described above). The best-fitting instances spawn new instances in their vicinity until no further improvement of the QoF measure is observed. The local adaptation happens by message passing. Part FEMs of the first level adapt to the image. Forces, induced by their displacement, are applied as natural boundary conditions to the second-level FEM. The resulting elastic deformation then cause node displacements on the second level that are passed as essential boundary conditions to the first level. The method has later been extended to 3d and the message passing mechanism was replaced by integration of the FEMs on the two or more levels into a single FEM formulation which combines different warping models in an efficient framework (Rak et al. 2013; Toennies et al. 2014).

11.7 Shape, Appearance and Pose Priors for Segmentation

Shape models, such as ASM or other deformable models, may be used for segmentation if part of the data knowledge is misleading. Ideally, the boundary of a fitted model instance already coincides with the segment boundary. In Heiman et al. (2006) and Okada et al. (2007) two different uses of an active shape model to segment the liver in CT are presented. However, the result may be unsatisfactory. A shape model will have just as many degrees of freedom as are necessary to describe the object characteristics. Object detail may not be representable with sufficient accuracy. Increasing the number of adaptable parameters is not a good solution, since the restriction is intentional in order to train or set shape variation from limited information. A more flexible deformable model may be able to adapt to the object boundary but it fails to fulfil the original goal for using a shape model. The model is no longer able to predict object shape and adapts to image detail irrespectively of whether this detail is reliable or not.

Hence, shape information is often used as prior knowledge that guides an otherwise data-driven segmentation method such as those described in the previous sections. The data-driven part should be able to correct the over-simplistic model at locations where the information from the data is strong.

11.7.1 Atlas-Based Segmentation

A simple means to use shape information for segmentation is to use a digital anatomical atlas. Ideally, the atlas is an image of the same kind than the image to be segmented (e.g., a t_2-weighted MRI if just these images are to be segmented). Each pixel in the atlas has a segmentation label associated to it. Hence, the atlas contains shape and appearance information of all structures covered by the atlas. It also

contains the relative pose information of the different organs with respect to each other (see Cuadra et al. 2015; Iglesias and Sabuncu 2015 for a detailed treatment).

Atlases are often used in segmentation of brain data. This structure varies relatively little compared to other regions of the body and many sub-structures in the brain are difficult to segment based on a data-driven strategy (Thompson and Toga 1997; Kelemen et al. 1999).

The atlas is first rigidly registered to the data. Individual shape variation is then accounted for by non-rigid registration. Segment labels of the atlas are—within the degrees of freedom of the deformation transformation—individualized for the particular image. If the segmentation after mapping of the labels is inaccurate but the image contains sufficient information for local refinement, the labels can be used as additional data term in a subsequent refinement step. The purpose is similar to slice-sequential approaches using shapes from previous slices to initialize the current slice, however, with the difference that initialization is automatic and driven by prior information about the shape.

If just a single structure needs to be described, shape variation may also be trained from samples. If the training data base is sufficiently large and representative for the object shape, it produces a representation that discriminates better between the target object and other objects than an elastically deforming atlas. However, the relationship to neighboring structures, which is present in atlases and which may be an important discriminator, is not present in most shape models. An example for the use of a trained shape model is Hu et al. (2011) where segmentation takes place in registered eigenshape space. The authors compute an active appearance model from training data to segment hippocampus and amygdala in brain MRI. The appearance terms are then used in the data term predicting the expected intensity while the shape terms constrain the propagation of the level set.

This idea to train shape variation from samples applied in the ASM-like strategy as detailed above may also be used for atlas-based segmentation. In multi-atlas segmentation, several atlases generated from several image samples represent anatomical variability. Atlases can either be applied separately to the image (Artaechevarria et al. 2009; Wang et al. 2013) or as a combined, fuzzy or probabilistic atlas (Mazziotta et al. 1995; Park et al. 2003).

If they are applied separately, each atlas is registered to the image. A label assigned by one of the atlases to a pixel in the image votes for this particular segment. Decisions are made, e.g., by majority vote or by weighted votes of all or of the best fitting atlases. The QoF-function of the registration method may be used to estimate the fitting quality.

If, instead, atlases are merged to a single atlas before segmentation, each pixel in the atlas has several labels assigned to it. Each label is assigned a weight that depends on how many of the atlases assign just this label to this pixel. Mapping the atlas to the image to be segmented requires a decision step just which of the labels is assigned to each pixel. This can be done, e.g., based on local appearance or by letting the result be the input for a subsequent segmentation process such as level set segmentation that uses the assigned weights as additional input.

Fig. 11.35 Shape priors can be used in data with very strong artefacts if the final segmentation method (active contours in this case) is optimized only locally (Engel and Toennies 2008)

If artefacts in images are too strong and the image does not contain enough different structures, global optimization based on the shape prior may still fail, since the data contains too many false object boundaries. Using a shape instance as prior still makes sense, since it can initialize a local search for the segment boundary. This was done by Engel and Toennies (2008), who used a shape model to detect an object of interest using a global shape search followed by a locally restricted adaptation of an active contour for segmenting the substantia nigra in transcranial ultrasound images (see Fig. 11.35).

11.7.2 Combining Shape Information with Level Set Segmentation

Shape information may also be integrated in level set segmentation. The anatomic atlases described above are particularly suitable for this purpose. They contain information about the shape of depicted organs and about spatial relations between organs. After registration with the image they serve as an extra data term for the level set evolution, see, e.g., Duan et al. (2010), Wang et al. (2014).

Shape samples from manually segmented training data may be used as well. The samples are aligned with respect to each other and a signed distance transform is carried out. The signed distance map of each sample is then re-organized into a shape feature vector. Similar to active shape models, principal component analysis is carried out on the set of all N samples and a subset of n variation modes is selected, see, e.g., Leventon et al. (2000), Cremers and Rousson (2007).

Fig. 11.36 Kernel density estimates may better represent the true shape distribution in shape space. In the example above, the Gaussian function predicts a most likely shape that is far from the four sample shapes

Segmentation is restricted to shapes that can be represented in this n-dimensional sub-space:

$$\phi(\alpha, \mathbf{h}, \boldsymbol{\theta}) = \bar{\phi}(\mathbf{R}_{\boldsymbol{\theta}}\mathbf{x} + \mathbf{h}) + \sum_{i=1}^{n} \alpha_i \psi_i(\mathbf{R}_{\boldsymbol{\theta}}\mathbf{x} + \mathbf{h}) \tag{11.62}$$

Here, $\mathbf{R}_{\boldsymbol{\theta}}$ is the rotation matrix parameterized with rotation angles $\boldsymbol{\theta} = \{\theta_1, \theta_2, \theta_3\}$ and $\mathbf{h}$ is a translation vector to account for the pose parameters. The function ψ_i maps the ith eigenmode back to image space. The α_i are the corresponding weights (similar to the $\mathbf{b}$-vector in ASMs). Hence, $\phi(\boldsymbol{\alpha}, \mathbf{h}, \boldsymbol{\theta})$ represents the finite sub-space in the infinite-dimensional shape space in which the optimal segmentation is searched.

In Cremers and Rousson (2007), the authors use kernel density estimates (see Fig. 11.36) instead as this better captures the true shape variation than assuming a Gaussian distribution of Leventon et al. (2000). They optimize the logarithm of the a posteriori probability of the shape and pose parameters given the image. It results in an energy term similar to the active contours without edges (Chan and Vese 2001) where the shape parameters (the a priori probability) are an additional summand that replaces the regularizer in Chan and Vese (2001):

$$\begin{aligned} E(\alpha, h, \theta) = &- \int_{\Omega} H_e(\phi(\mathbf{x}))\log p_{fg}(I) + (1 - H_e(\phi(\mathbf{x})))\log p_{bg}(I)\mathrm{d}x \\ &- \log\left(\frac{1}{N\sigma}\sum_{i=1}^{N} K\left(\frac{\boldsymbol{\alpha} - \boldsymbol{\alpha}_i}{\sigma}\right)\right) \end{aligned} \tag{11.63}$$

H_e is the regularized Heaviside function. The functions p_{fg} and p_{bg} are probabilities of intensity I belonging to either the foreground or background, that are used instead of the squared intensity difference from reference intensities in Chan and Vese (2001), and K is a Gaussian kernel function on the difference of the current weight vector and the weight vector α_i of the ith sample. The standard deviation σ

regulates the probability with which differences from a reference shape are tolerated. The energy E to be minimized is the negative log-likelihood of the probability of the current level set to describe the searched segment given the foreground and background probabilities and the shape prior.

The variational derivatives of the pose parameters $\mathbf{h}$ and $\boldsymbol{\theta}$ only depend on the image data while the prior term $\boldsymbol{\alpha}$ is also influenced by its deviation from the shape vectors of the training samples:

$$\frac{\mathrm{d}\mathbf{h}}{\mathrm{d}t} = \int_{\Omega} \delta_e\left(\phi_{\boldsymbol{\alpha},\mathbf{h},\boldsymbol{\theta}}(\mathbf{x})\right)\nabla\phi_{\boldsymbol{\alpha},\mathbf{h},\boldsymbol{\theta}}\log\frac{p_{bg}(I)}{p_{fg}(I)}\mathrm{d}x \tag{11.64}$$

$$\frac{\mathrm{d}\boldsymbol{\theta}}{\mathrm{d}t} = \int_{\Omega} \delta_e\left(\phi_{\boldsymbol{\alpha},\mathbf{h},\boldsymbol{\theta}}(\mathbf{x})\right)\nabla\phi_{\boldsymbol{\alpha},\mathbf{h},\boldsymbol{\theta}}\nabla_{\boldsymbol{\theta}},\mathbf{R}\mathbf{x}\log\frac{p_{bg}(I)}{p_{fg}(I)}\mathrm{d}x \tag{11.65}$$

$$\frac{\mathrm{d}\boldsymbol{\alpha}}{\mathrm{d}t} = \int_{\Omega} \delta_e\left(\phi_{\boldsymbol{\alpha},\mathbf{h},\boldsymbol{\theta}}(\mathbf{x})\right)\psi(\mathbf{R}_{\boldsymbol{\theta}}\mathbf{x}+\mathbf{h})\log\frac{p_{bg}(I)}{p_{fg}(I)}\mathrm{d}x + \frac{1}{\sigma^2}\frac{\sum_{i=1}^{N}(\boldsymbol{\alpha}_i-\mathbf{a})K(\boldsymbol{\alpha}_i-\mathbf{a})}{\sum_{i=1}^{N}K(\boldsymbol{\alpha}_i-\mathbf{a})} \tag{11.66}$$

Each derivative is weighted with the regularized Dirac-δ function δ_e and, hence, has to be evaluated only in the vicinity of the 0-level-set. The system aligns intrinsically to the shapes in the image provided that the intensity-based foreground and background probabilities are sufficiently discriminative and the process is initialized sufficiently close to the object to be segmented.

If the shape can be represented as local attributes of the level set, its integration becomes much simpler. For sphere-like structures, Lim et al. (2013) introduced Willmore flow as indirect shape prior. The Willmore energy of a surface decreases with its average curvature. Hence, sphere-like structures would be preferred.

Willmore energy can be combined with Cremer's shape model as it was done in Lim et al. (2013) to segment the vertebrae. This would not be necessary if the shape models were sufficiently accurate to predict the individual shape. However, the shape model in Lim et al. (2013) is not based on a sufficiently large sample data base to make such accurate predictions. Either the finite-dimensional shape sub-space of Cremers and Rousson (2007) has too few dimensions or the kernel density estimate is too inaccurate. Hence, the shape model locks in the level set at places where boundary information is sufficient, whereas the Willmore flow provides missing information at locations where neither data information nor the shape model can predict where to find the boundary.

Another simple shape prior has been presented by Yazdanpanah et al. (2011). They introduce a circular shape prior parameterized by center and radius for segmenting retina structures in OCT. For separating the different layers of the retina, they defined as many level sets as they have layers to segment. The layers form concentric circles in the 2d image domain. Defining shape priors of all level sets to

Fig. 11.37 The front propagation of Kohlberger et al. (2009) tracks key points on an approximately aligned liver shape. The propagation speed is controlled by expected local features for these key points

have the same center further constrains the shape (by determining a common location of all centers). The different segments are ordered by the size of the radius. The region for each segment is defined by the inside of the respective level set that is outside of all segments with smaller radius.

Instead of predicting a global shape as in the previous two examples, specific shape variation may be trained locally if a reference to a global shape can be given. In Kohlberger et al. (2009) the authors train expected local features, such as intensities at the organ boundary and curvature of the boundary, from livers and kidney in CT images. Key points on an approximately aligned reference shape are tracked in a front propagation approach and local features associated to these key points are then used to control the evolution of the level set (see Fig. 11.37). The local features are combined with a global, explicitly defined shape prior that is integrated with the implicit level set representation using the method presented by Farzinfar et al. (2008).

Pose priors need to be treated extra since the shape of a single object does not tell much about its relation to other objects in the image. The simplest way to integrate this is by interactive specification of the initial 0-level set. Since most level set methods find only a local optimum, results depend on the initialization. This apparent disadvantage remedies the drawback of the ambiguous definition of the object and its appearance by the level set model. It also makes the narrowband technique advantageous, as it reduces levels sets to a kind of advanced region growing from user-specified start regions.

An example that derives location and appearance information from initialization is Taheri et al. (2010) on tumor segmentation. The 0-level set propagates from an initial position fully inside the tumor. The evolution is driven by intensity-based

tumor probabilities of the voxels. Parameters of a Gaussian distribution are estimated from the region inside the initial 0-level set and updated during level set evolution.

In Rink and Toennies (2007) information about the current shape of the 0-level set is used to guide its evolution. Their *bridging force* to segment dendritic spines in electron microscopy is based on the current evolution direction and speeds up evolution at high curvature locations if another part of the 0-level set is sufficiently close. Hence, it is able to bridge small gaps in longish structures that are due to insufficient support by the data.

Spatial relationships between different segments are included if coupled level sets are used where one level set influences the other level set. This has been presented by Zeng et al. (1999), who use two active surfaces for the segmentation of the cortex into white matter (wm), grey matter (gm) and cerebro-spinal fluid (csf). While each of the surfaces may evolve according to different features within the data, they have to stay in a predefined normal range to each other given by the estimated cortical thickness. Similar techniques have been used by Sokoll et al. (2008), who derived and adapted features during level set evolution (see Fig. 11.38), and by Wang et al. (2014), who introduce a distance term between csf-gm interface and gm-wm interface that penalizes deviations from the acceptable thickness range of grey matter.

An example that uses coupling in different ways at several steps of a multi-step segmentation is Duan et al. (2010). Their goal is to segment the inner and outer wall of the bladder in MR images. Since the inner wall is much better visible, it is segmented in the first step and serves as initialization for the level set function representing the outer wall. Since the appearance of the bladder wall is different to the background, the data term in the Chan-Vese model is modified so that only a small ring outside of the inner wall is evaluated for computing the difference between average intensity in the background and the pixel intensity. After having an indirect definition of the outer wall based on the current estimate for the inner wall, a new level set function is defined that includes both walls that are then evolved together.

Fig. 11.38 The coupling energy in Sokoll et al. (2008) penalizes deviations from the expected thickness of the *grey* matter. The two 0-level-sets for the *grey* matter/*white* matter boundary and for the *grey* matter/csf boundary are coupled. A additional coupling energy penalizes distances that exceed the expected thickness of the *grey* matter

11.7.3 Solutions Based on Graph Cuts

Shape information can also be added to graph cut segmentation (Vu and Manjunath 2008; Freedman and Zhang 2005). A simple way to make shape distance exploitable to graph cuts is to assign the shape deviation penalty to pixel sites. This is realized in Freedman and Zhang (2005) by generating an unsigned distance function $\bar{\phi}$ of the reference shape and by using this to adapt the n-link energy of Boykov et al. (2001), see Sect. 8.1.2, by a shape distance (see Fig. 11.39)

$$d^2(\mathbf{p}, \mathbf{q}) = \bar{\phi}\left(\frac{\mathbf{p}+\mathbf{q}}{2}\right). \tag{11.67}$$

This term needs to be weighed against all other terms in the equation, which in this case is the n-link energy $B_{\mathbf{p},\mathbf{q}}$ that decreases with increasing gradient between the two pixels $\mathbf{p}$ and $\mathbf{q}$. The effects will be similar to the application of shape priors in level set segmentation. In the absence of strong gradients, the boundary will be defined by the shape prior. This approach just weights the distance but not the shape similarity to the model. For enforcing shape similarity, the gradient of the distance function can be used (Nakagomi et al. 2013) if the angle between the direction between adjacent pixels and gradient of the distance transform of registered model shapes are used to weight the n-links.

Shape may also be incorporated via t-links. This was done by Ali et al. (2007) for kidney segmentation. Shapes from trained kidney shapes resulted in a probabilistic kidney distance map which was directly assigned to the corresponding t-links. It can be extended to incorporate intensity information as well. In this case, an AAM is registered with the image and shape and appearance costs are assigned to each pixel in the t-links, which depend on the distance from the registered shape and the intensity difference of the expected appearance (Chen et al. 2013). Information from a registered atlas has been used to supplement the intensity value in the t-links by Song et al. (2006) for carrying out a multilabel segmentation. They use the Lorentzian error norm as robust error measure for computing homogeneity on n-links. The Lorentzian error norm of intensity difference d_{ij} between two nodes i and j is defined as

Fig. 11.39 The difference between shape prior and actual shape may be integrated into a graph cut algorithm by weighting the n-links between neighboring pixels or by penalizing t-links between pixels and source or sink node

Fig. 11.40 The length of the gradient of the robust Lorentzian error norm decreases with increased error (which in our case is the intensity difference between pixels)

$$L_\sigma(d_{ij}) = \log\left(1 + \frac{1}{2}\left(\frac{d_{ij}}{\sigma}\right)^2\right), \tag{11.68}$$

where σ is a parameter that regulates the influence of increasing differences on the error measure. The Lorentzian error measure is robust w.r.t. outliers since the gradient of the error function decreases with increasing intensity difference (see Fig. 11.40). Inhomogeneity is iteratively corrected by estimating label-specific average intensities creating an inhomogeneity field.

In Vu and Manjunath (2008) an overlap difference between actual shape and reference shape has been used as shape distance. The shape prior predicts whether a pixel belongs to the foreground or to the background. A weighted penalty is added to each t-link that depends on whether pixels should be in the foreground or in the background (see Fig. 11.31). This will increase cost of the cut when the difference between the reference shape and the produced segmentation gets larger.

11.7.4 Graph Cuts on Pre-segmented Images

The application of graph cuts to pre-segmented images was already suggested by Boykov et al. (2001) for computing the optimal labeling in an MRF formulation of the segmentation problem.

A strategy that is based on a preliminary segmentation has been suggested by Li et al. (2004). The initial segmentation defines the topology of the final result and the graph is built on this segmentation to produce a final result that fits better to the data. The method has later been called LOGISMOS (layered optimal graph image segmentation of multiple objects and surfaces) by the authors (Yin et al. 2010).

First, a shape model, e.g., an ASM or an elastically deformable model is registered with the object to be segmented. A surface mesh is built and columns are created orthogonal to the surface at each mesh point. Locations along the columns are potential true segment boundary locations. These locations are nodes of a

Fig. 11.41 The graph of the LOGISMOS algorithm is built on a surface mesh that is created from a pre-segmented boundary. Columns are created orthogonal to the surface and nodes are created along the columns. Inter-column arcs and intra-columns arcs restrict the course of the potential final segment boundary

directed graph with edges within the columns, between columns and also potentially between columns of different objects in close spatial vicinity (see Fig. 11.41). The final segmentation will consist of a subset of these nodes so that exactly one node per column is selected.

The solution is found by a graph cut. In its initial version, the method was restricted to star-convex surfaces but later publications showed also solutions that deal with the intersection problem that may arise for concave surfaces.

Smoothness is enforced by several factors. The height of the columns restricts the variation of the searched surface from the registered model instance. Increasing or decreasing the distance from the initial surface along the column is penalized to ensure smoothness. Column graphs are interconnected and used to introduce a further smoothness constraint. The graph is constructed such that the searched segmentation boundary is given by the surface of the minimum closed set of this graph given the weights. Edges are directed with back-direction with infinite capacity to avoid that the same column is cut twice (Li et al. 2012).

Various applications of the original methods have been published Liu et al. (2013), Xia et al. (2014). In Li et al. (2006) the method is extended to search a double surface. Each surface is represented by its own graph with intra-column weights containing the desired surface attributes for each of the surfaces. Columns of the two graphs are connected in a way that guarantees that a minimum distance between the two surfaces is always preserved. This is a hard constraint, opposed, e.g., to the double contours by level sets where the distance is just an additional term in the optimization equation. The method has later been applied to other tubular structures such as the upper airway tree (Yin et al. 2010; Liu et al. 2013).

11.8 Using a Graphical Model for Detecting Object Parts

Graphical models describe dependencies between random variables by a graph. Its nodes are random variables or random processes. Two nodes are connected by an edge if the variables depend on each other. Graphical models may be used for the probabilistic representation of a phenomenon that depends on a set of random variables. The graph is usually not fully connected, since random variables in the graph do depend on a small subset of the other random variables in the graph (see Fig. 11.42).

Graphical models may have directed or undirected edges. An undirected graphical model is a Markov Random Field and represents local dependencies between nodes of the MRF. Directed graphical models represent causalities between nodes. A directed edge from node n_i to node n_j represents the dependency of the value at n_j on the value n_i but not vice versa. Hence, an event at n_i causes, with a certain probability, an event at n_j.

In computer vision, a graphical model may be used to deduce an underlying condition, e.g., the presence of a pedestrian, from observations such as global and local image features. A number of methods exist for the deduction, of which sampling the posterior distribution and (loopy) belief propagation are probably the most often used strategies.

A sampling strategy repeatedly takes samples from the posterior distribution. The goal is to find the most probable explanation for the observations (simulated annealing, discussed in Sect 14.1, is an example for this strategy). The method is slow but has few restrictions regarding the underlying graphical model. It is often used in undirected graphical models.

Belief propagation, see, e.g., Yedidia et al. (2003), is a well-established method that guarantees to converge to the marginal probabilities $P(x_i)$ for random variables $X_i = \{x_i\}$ for each node n_i in acyclic directed graphs. It builds on a message passing strategy where each node repeatedly receives a message for each possible state from all neighboring parent nodes (see Fig. 11.43). This may be seen as prediction of the node state based on the states of the parent nodes. For computing probabilities using

Fig. 11.42 A graphical model represents dependencies between random variables. The example shows an acyclic graphical model. Graphical models may be directed or undirected. Directed graphical models may be cycle-free (chains, trees, …) or may contain cycles

Fig. 11.43 Update step for message passing in belief propagation. Message $m_{ij}^{t+1}(x_j)$ from x_i to x_j at iteration $t + 1$ is generated by summing up all conditional probabilities for x_j given all possible states x_i of random variable X_i weighted by the current probability estimate $h(x_i)$ of x_i. The value of $h(x_i)$ is given by the current prior probability estimate $p^t(x_i)$ which is weighted by all incoming messages

belief propagation, initially all beliefs are set to 1 or to some random value. The update step then passes the node probabilities through the graph eventually producing the exact conditional probabilities and thus also the exact marginal probabilities for each state of each node. This can then be used to compute the probability for a certain state configuration.

Reaching the most probable state in all nodes by belief propagation is not guaranteed if the graph contains cycles (also called loops). However, even loopy belief propagation (belief propagation in graphs with cycles) has been shown to lead to good results. Hence it may also be applied to undirected graphical models.

Alternatively, the graph may be separated into all its cliques. Cliques with more than one node constitute a cycle and are transformed into supernodes. Probability computation in the cliques is now separated and belief propagation is carried out on a new graph consisting of supernodes and single-node cliques.

Graphical models can be used to represent part-relationships. In this case, the observed information consists of local image attributes that indicate potential presences of parts. The part relationships are modeled as dependencies between parts and represented by edges of a graphical model. Since the structure of the graphical model is usually known, the only information needed are the conditional probabilities of observing image features if a part is present at some place. A trained graphical model can then be used to predict the existence and position of an object to be searched. The result may then be used for a subsequent local segmentation of the parts.

Opposed to the part model described above (m-rep, ASSM, part-based FEM), it is simple to deal with missing parts. Since part-to-part dependencies are always probabilistic, a part relationship does not imply certainty as to the existence of another part.

Graphical models to model part relationships have been applied repeatedly to detect the spine in medical images (Rak and Toennies 2016a). In this case, it even turned out that the structure is such a strong constraint that prior probabilities

needed only roughly be estimated and required little to no training (Rak and Toennies 2016b). If the structure of part-relationships is not such a strong predictor for the presence of the object in search, the necessary probabilities can be trained from interpreted images, since usually all states are observable.

If not all states are observable, expectation maximization can be used to estimate expected values for unobservable nodes. The expected values are treated as if they were observations resulting in new estimates of expected values. Since expectation maximization is a local optimization procedure, it may terminate in a local minimum and thus requires a good initialization.

If graphical models are meant to represent part-relationships, the structure of the graph can be assumed to be known and needs not to be found from samples. This is not true for other applications of graphical models. For the sake of completeness it should thus be mentioned that the structure of graphical models may also be learned from observations. If the number of nodes is known and states of each node are observable, the necessary figure-of-merit for finding the optimal graph may be generated from the marginal probabilities of all parameters. This penalizes graphs with many edges, since marginal probabilities of loosely dependent variables would increase the total of marginal probabilities. It is an application of the principle of Occam's Razor since a graph is searched that explains the observed states with as few dependencies as possible. Unfortunately, the search space, i.e., the number of edge configurations, increases super-exponentially with the number of nodes. Hence, sub-optimal strategies are applied, e.g., by commencing with a fully connected graph and always removing those edges that produce the smallest improvement of the figure of merit. The problem becomes more complex if not all states are observable. The missing marginal probabilities then need to be estimated, e.g., by expectation maximization.

11.9 Concluding Remarks

Models of object attributes are a powerful constraint system to find and segment objects when discriminating among objects is impossible based on data alone. This may be the case if the data is severely distorted by artefacts (e.g., in ultrasound) or if adjacent, different objects have the same intensity in the image (which is the case for many soft tissues in MR or CT images). The object model introduces the necessary a priori knowledge for finding and extracting the object nonetheless.

It is mandatory that shape attributes are part of the object model, since low-contrast or high-noise images require shape information to separate the object from surrounding structures. Appearance information supports detection and it may also help to stabilize shape information at object boundaries.

The biggest problem using object models is to capture and represent acceptable variation within an object class. Training from samples and user-specified decomposition of objects into simpler parts are two strategies to support

acquisition of this information. Object variation can be represented implicitly by accepting a certain degradation of the quality-of-fit measure, or explicitly by including variation into the model.

Although a shape model can be necessary to produce a segmentation at all, restrictions of its deformation may require post-processing that allows a fitting of segment boundaries to the data at locations where image information is reliable.

11.10 Exercises

- When would a shape model be advisable for supporting a segmentation?
- How is an active shape model used if an object is to be segmented? How could one deal with the fact that the ASM probably would not represent enough shape detail for the segment delineation?
- How could template matching be used to estimate the kidney position (of a healthy kidney) in MRI? Please explain what information is needed and how it could be gathered and represented by the template.
- Assume that the Hough transform is used to find the spherical microorganisms in Fig. 11.7. What would be a strategy to differentiate between bad fits (but microorganisms are found) and wrong detections? Please explain under which assumption your criterion works.
- How could the GHT be used to find the typical shape of the ventricles in Fig. 11.4, given that some shape and size variance has to be expected?
- What is the difference between using normalized cross-correlation (Eq. 11.2) and covariance (Eq. 11.3) for template matching? What would be a situation where Eq. 11.2 should be preferred?
- Why does the medial axis transform not always represent major shape attributes properly? With what kind of strategy can the problem be solved?
- What is the advantage of the hierarchical structure of the medial axis representation? Please explain how the domain knowledge for each of the levels could be acquired if the m-rep were used to detect vertebrae in CT image sequences.
- How is variability encoded in an ASM and how is this information acquired?
- Why could it produce problems when training data for a 2d ASM consisting of 50 boundary points consists of 100 samples?
- What kind of landmarks are used for representing an ASM? What kind of information about the shape is encoded by the different types of landmarks?
- What influence on the trained ASM does it have when alignment of the training samples was incorrect? What is the influence if landmark positions are inexact?
- Please explain how alignment is carried out. Under which assumptions will this alignment be successful?
- Why can it safely be assumed that the true dimension of the feature space of an ASM is much lower than the number of degrees of freedom given by the boundary points?

- Why should the dimensionality be further reduced than the dimension that is spanned by the features from the samples?
- What are the stages necessary to fit an ASM to the image?
- How can the ASM be extended to carry appearance information? Why is shape and appearance first trained separately?
- How is shape variability represented by a mass-spring model? And how can this variability be changed if necessary?
- What is the purpose of introducing torsion forces in the mass-spring model?
- How is a mass-spring model fitted to the data? How is the influence from the data represented?
- What is the difference of a shape representation by a mass-spring model and a finite element model?
- How can appearance information be represented in a mass-spring model or a finite element model?
- What role do Poisson ratio and elasticity modulus play when defining a finite element model for detecting objects in an image?
- What is meant with natural and essential boundary condition of a finite element model? Why are such conditions needed?
- What is the advantage of computing free vibration modes for an FEM when computing motion of a dynamic finite element model? When would it be preferred instead of a direct computation of the displacement?
- How can the transformation into free vibration modes be used to compute a quality-of-fit measure?
- What are similarities and what are the differences between vibration modes of finite element models and variation modes of ASMs?
- How can finite element models be used to support training of an ASM? Why can this be assumed to be successful (what does it say about the data)?
- What is the difference between integration of a shape constraint into the model term and into the data term in level set segmentation?
- Why did the shape-based level sets of Cremers et al. use a kernel density estimate for representing shape variation?
- How can a shape constraint be integrated into a graph cut solution?
- How is a shape constraint integrated in graph cuts using the LOGISMOS algorithm? What kind of pre-processing is necessary for this approach?
- How can part-relationships be represented using a graphical model? What kind of graphical model would be appropriate?
- Explain how probabilities in a graphical model are computed using belief propagation. What is the condition for the graph under which belief propagation produces the optimum?

References

Ali AM, Farag AA, El-Baz AS (2007) Graph cuts framework for kidney segmentation with prior shape constraints. In: MICCAI 2007, Part I. LNCS, vol 4791, pp 384-392

Al-Zubi S, Toennies KD (2003) Generalizing the active shape model by integrating structural knowledge to recognize hand drawn sketches. In: Proceedings of CAIP 2003. LNCS, vol 2756, pp 320–328

Al-Zubi S, Brömme A, Toennies K (2003) Using an active shape structural model for biometric sketch recognition. In: Joint Pattern Recognition Symposium, pp 187–195

Artaechevarria X, Munoz-Barrutia A, Ortiz-de-Solórzano C (2009) Combination strategies in multi-atlas image segmentation: application to brain MR data. IEEE Trans Med Imaging 28 (8):1266–1277

Bardinet E, Cohen LD, Ayache N (1995) Tracking medical 3D data with a parametric deformable model. In: Proceedings of IEEE international symposium computer vision, pp 299–304

Barr AH (1992). Rigid physically based superquadrics. In: Kirk D (ed) Graphics gems III. Academic Press, pp 137–159

Bergner S, Al-Zubi S, Toennies KD (2004) Deformable structural models. In: Proceedings of the IEEE international conference image processing ICIP, pp 1875–1878

Biederman I (1985) Human image understanding: recent research and a theory. Comput Vis Graph Image Process 32:29–73

Binford T (1987) Generalized cylinder representation encyclopedia of artificial intelligence. Wiley, New York, pp 321–323

Blum H (1967) A transformation for extracting new descriptors of shape. In: Proceedings of a symposium models for the perception of speech and visual form, pp 362–380

Boykov Y, Veksler O, Zabih R (2001) Fast approximate energy minimization via graph cuts. IEEE Trans Pattern Anal Mach Intell 23(11):1222–1239

Brett AD, Taylor CJ (1999) A framework for automated landmark generation for automated 3d statistical model construction. In: Proceedings of 16th international conference on information processing in medical imaging IPMI'99. LNCS, vol 1613, pp 376–381

Byers R, Xu H (2008) A new scaling for Newton's iteration for the polar decomposition and its backward stability. SIAM J Matrix Anal Appl 30(2):822–843

Chan TF, Vese LA (2001) Active contours without edges. IEEE Trans Image Process 10(2):266–277

Chen X, Udupa JK, Alavi A, Torigan DA (2013) GC-ASM: synergistic integration of graph cut and active shape model strategies for medical image segmentation. Comput Vis Image Underst 117:513–524

Cheung KW, Yeung DY, Chin RT (2002) On deformable models for visual pattern recognition. Pattern Recognit 35(7):1507–1526

Chevalier L, Jaillet F, Baskurt A (2001) 3D shape coding with superquadrics. In: Proceedings of IEEE international conference image processing ICIP, II, pp 93–96

Choi MG, Hyeong-Seok K (2005) Modal warping: real-time simulation of large rotational deformation and manipulation. IEEE Trans Vis Comput Graph 11(1):91–101

Cootes TF, Taylor CJ (1992) Active shape models—'smart snakes'. In: Proceedings of British machine vision conference

Cootes TF, Taylor CJ (1995) Combining point distribution models with shape models based on finite-element analysis. Image Vis Comput 13(5):403–409

Cootes TF, Taylor CJ (1999) A mixture model for representing shape variation. Image Vis Comput 17(8):567–573

Cootes TF, Hill A, Taylor CJ, Haslam J (1994) The use of active shape models for locating structures in medical images. Image Vis Comput 12(6):355–366

Cootes TF, Taylor CJ, Cooper DH, Graham J (1995) Active shape models—their training and application. Comput Vis Image Underst 61(1):38–59

Cootes TF, Edwards GJ, Taylor CJ (1998) Active appearance models. In: 5th European conference on computer vision ECCV1998. LNCS, vol 1407, pp 484–498

Cremers D, Rousson M (2007) Efficient kernel density estimation of shape and intensity priors for level set segmentation. In: Deformable models. Springer, New York, pp 447–460

Cuadra MB, Duay V, Thiran JP (2015) Atlas-based segmentation. In: Handbook of biomedical imaging. Springer, New York, pp 221–244

Davies ER (1988) A modified Hough scheme for general circle location. Pattern Recognit 7(1):37–43

Delingette H, Hebert M, Ikeuchi K (1992) Shape representation and image segmentation using deformable surfaces. Image Vis Comput 10(3):132–145

Dornheim L, Toennies KD, Dornheim J (2005) Stable dynamic 3d shape models. In: IEEE international conference on image processing ICIP, III, pp 1276–1279

Duan Z, Liang S, Bao H, Zhu S, Wang G, Zhang JJ, Chen HLu (2010) A coupled level set framework for bladder wall segmentation with application to MR cystography. IEEE Trans Med Imaging 29(3):903–915

Edelman S (1997) Computational theories in object recognition. Trends Cognit Sci 1:296–304

Engel K, Toennies KD (2008) Segmentation of the midbrain in transcranial sonographies using a two-component deformable model. In: 12th annual conference medical image understanding and analysis, pp 3–7

Engel K, Toennies KD (2009) Hierarchical vibrations: a structural decomposition approach for image analysis. In: Energy minimization methods in computer vision and pattern recognition. LNCS, vol 5681, pp 317–330

Engel K, Toennies KD (2010) Hierarchical vibrations for part-based recognition of complex objects. Pattern Recognit 43(8):2681–2691

Engel K, Toennies KD, Brechmann A (2011) Part-based localisation and segmentation of landmark-related auditory cortical regions. Pattern Recognit 44(9):2017–2033

Farzinfar M, Xue Z, Teoh EK (2008) Joint parametric and non-parametric curve evolution for medical image segmentation. In: Europe conference computer vision (ECCV 2008), pp 167–178

Ferrant M, Macq B, Nabavi A, Warfield SK (2000) Deformable modeling for characterizing biomedical shape changes. In: 9th international conference discrete geometry for computer imagery DGCI 2000. LNCS, vol 1953, pp 235–248

Frangi AF, Rueckert D, Schnabel J, Niessen WJ (2001) Automatic 3d ASM construction via atlas-based landmarking and volumetric elastic registration. In: Proceedings of 17th international conference information processing in medical imaging IPMI 2001. LNCS, vol 2082, pp 78–91

Freedman D, Zhang T (2005) Interactive graph cut based segmentation with shape priors. In: IEEE computer society conference on computer vision and pattern recognition (CVPR 2005), vol 1, pp 755–762

Giblin P, Kimia BB (2004) A formal classification of 3d medial axis points and their local geometry. IEEE Trans Pattern Recognit Mach Intell 26(2):238–251

Gloger O, Toennies KD, Mensel B, Völzke H (2015) Fully automatized renal parenchyma volumetry using a support vector machine based recognition system for subject-specific probability map generation in native MR volume data. Phys Med Biol 60(22):8675

Gong L, Pathak SD, Haynor DR, Cho PS, Kim Y (2004) Parametric shape modeling using deformable superellipses for prostate segmentation. IEEE Trans Med Imaging 23(3):340–349

Hamarneh G, McInerney T, Terzopoulos D (2001) Deformable organisms for automatic medical image analysis. In: Medical image computing and computer-assisted intervention MICCAI 2001. LNCS, vol 2208, pp 66–76

Heimann T, Wolf I, Meinzer HP (2006) Active shape models for a fully automated 3d segmentation of the liver—an evaluation on clinical data. In: Medical image computing and computer-assisted intervention MICCAI 2006. LNCS, vol 4191, pp 41–48

Hu S, Coupé P, Pruessner JS, Collins DL (2011) Appearance-based modeling for segmentation of hippocampus and amygdala using multi-contrast MR imaging. Neuroimage 58(2):549–559

Iglesias JE, Sabuncu MR (2015) Multi-atlas segmentation of biomedical images: a survey. Med Image Anal 24(1):205–219

Jackway PT, Deriche M (1996) Scale-space properties of the multiscale morphological dilation-erosion. IEEE Trans Pattern Anal Mach Intell 18(1):38–51

Joshi S, Pizer SM, Fletcher PT, Yushkevich P, Thall A, Marron JS (2002) Multiscale deformable model segmentation and statistical shape analysis using medial descriptions. IEEE Trans Med Imaging 21(5):538–550

Kassim AA, Tan T, Tan KH (1999) A comparative study of efficient generalized Hough transform techniques. Image Vis Comput 17(10):737–748

Kelemen A, Székely G, Gerig G (1999) Elastic model-based segmentation of 3-D neuroradiological data sets. IEEE Trans Med Imaging 18(10):828–839

Kichenassamy S, Kumar A, Olver P, Tannenbaum A, Yezzi A (1995) Gradient flows and geometric active contour models. In: 5th international conference computer vision (ICCV'95), pp 810–817

Kohlberger T, Uzubas MG, Alvino C, Kadir T, Slosman D, Funka-Lea G (2009) Organ segmentation with level sets using local shape and appearance priors. Medical image computing and computer-assisted intervention–MICCAI 2009. Springer, Berlin, pp 34–42

Lam L, Lee SW, Suen CY (1992) Thinning methodologies—a comprehensive survey. IEEE Trans Pattern Anal Mach Intell 14(9):869–885

Leventon ME, Grimson WEL, Faugeras O. (2000) Statistical shape influence in geodesic active contours. In: IEEE conference computer vision and pattern recognition (CVPR 2000), vol 1, pp 316–323

Li K, Wu X, Chen DZ, Sonka M (2006) Optimal surface segmentation in volumetric images—a graph-theoretic approach. IEEE Trans PAMI 28(1):119–134

Li K, Wu X, Chen DZ, Sonka M (2004) Efficient optimal surface detection: theory, implementation and experimental validation. In: Proceedings of SPIE international symposium medical imaging: image processing, pp 620–627

Li X, Chen X, Yao J, Zhang X, Yang F, Tian J (2012) Automatic renal cortex segmentation using implicit shape registration and novel multiple surfaces graph search. IEEE Trans Med Imaging 31(10):1849–1860

Lim PH, Bagci U, Bai L (2013) Introducing Willmore flow into level set segmentation of spinal vertebrae. IEEE Trans Biomed Eng 60(1):115–122

Lindeberg T (1994) Scale-space theory: a basic tool for analysing structures at different scales. J Appl Stat 21(2):225–270

Liu X, Chen DZ, Tawhai MH, Wu X, Hoffman EA, Sonka M (2013) Optimal graph search based segmentation of airway double surfaces across bifurcations. IEEE Trans Med Imaging 32 (3):493–510

Mandal C, Vemuri BC, Qin H (1998) A new dynamic FEM-based subdivision surface model for shape recovery and tracking in medical images. In: Medical image computing and computer-assisted intervention MICCAI'98. LNCS, vol 1496, pp 753–760

Marr D (1983) Vision. Henry Holt & Company

Mazziotta JC, Toga AW, Evans A, Fox P, Lancaster J (1995) A probabilistic atlas of the human brain: theory and rationale for its development: the international consortium for brain mapping (ICBM). Neuroimage 2(2):89–101

Mokhtarian F, Mackworth A (1986) Scale-based description and recognition of planar curves and two-dimensional objects. IEEE Trans Pattern Anal Mach Intell 8(1):34–43

Müller M, Gross M (2004) Interactive virtual materials. In: Proceedings of graphics interface GI'04, pp 239–246

Nakagomi K, Shimizu A, Kobatake H, Yakami M, Fujimoto K (2013) Multi-shape graph cuts with neighbor prior constraints and its application to lunge segmentation from chest CT volume. Med Image Anal 17:62–77

Okada T, Shimada R, Sato Y, Hori M, Yokota K, Nakamoto M, Chen YW, Nakamura H, Tamura S (2007) Automated segmentation of the liver from 3d CT images using probabilistic atlas and multi-level statistical shape model. In: medical image computing and computer-assisted intervention MICCAI 2007. LNCS, vol 4791, pp 86–93

Paloc C, Bello F, Kitney R, Darzi A (2002) Online multiresolution volumetric mass spring model for real time soft tissue deformation. In: Proceedings of 5th international conference medical image computing and computer-assisted intervention MICCAI 2002. LNCS, vol 2489, pp 219–226

Park H, Bland PH, Meyer CR (2003) Construction of an abdominal probabilistic atlas and its application in segmentation. IEEE Trans Med Imaging 22(4):483–492

Pentland AP, Sclaroff S (1991) Closed-form solutions for physically-based modeling and reconstruction. IEEE Trans Pattern Anal Mach Intell 13(7):715–729

Petyt M (1998) Introduction to finite element vibration analysis. Cambridge University Press

Pizer SM, Oliver WR, Bloomberg SH (1987) Hierarchical shape description via the multiresolution symmetric axis transforms. IEEE Trans Pattern Anal Mach Intell 9(4):505–511

Pizer SM, Fritsch DS, Yushkevich PA, Johnson VE, Chaney EL (1999) Segmentation, registration, and measurement of shape variation via image object shape. IEEE Trans Med Imaging 18(10):851–865

Provot X (1995) Deformation constraints in a mass model to describe rigid cloth behavior. In: Graph Interface, pp 147–154

Rak M, Toennies KD (2016a) On computerized methods for spine analysis in MRI: a systematic review. Intl J Comput Assist Radiol Surg 11(8):1445–1465

Rak M, Toennies KD (2016b) A learning-free approach to whole spine vertebra localization in MRI. In: Medical image computing and computer-assisted intervention MICCAI 2016

Rak M, Engel K, Toennies KD (2013) Closed-form hierarchical finite element models for part-based object detection. In: Vision, Modelling, and Visualization VMV 2013, pp 137–144

Riesenhuber M, Poggio T (2000) Models of object recognition. Nat Neurosci Suppl 3:1190–1204

Rink K, Toennies KD (2007) A level set bridging force for the segmentation of dendritic spines. Computer analysis of images and patterns CAIP 2007. Springer, Berlin, pp 571–578

Rivlin E, Dickinson SJ, Rosenfeld A (1995) Recognition by functional parts. Comput Vis Image Underst 62(2):164–176

Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500):2323–2326

Sclaroff S, Pentland AP (1995) Modal matching for correspondence and recognition. IEEE Trans Pattern Anal Mach Intell 17(6):545–561

Sokoll S, Rink K, Toennies KD, Brechmann A (2008) Dynamic segmentation of the cerebral cortex in MR data using implicit active contours. In: 12th annual conference medical image understanding and analysis MIUA 2008, pp 184–188

Song Z, Tutison N, Avants B, Gee BC (2006) Integrated graph cuts for brain MRI segmentation. Proceedings of the international conference on medical image computing and computer-assisted intervention MICCAI 2006, LNCS 4191, pp 831–838

Taheri S, Ong SH, Chong VFH (2010) Level-set segmentation of brain tumors using a threshold-based speed function. Image Vis Comput 28(1):26–37

Terzopoulos D, Fleischer K (1988) Deformable models. Vis Comput 4(6):306–331

Terzopoulos D, Metaxas D (1991) Dynamic 3D models with local and global deformations: deformable superquadrics. IEEE Trans Pattern Anal Mach Intell 13(7):703–714

Terzopoulos D, Platt J, Barr A, Fleischer K (1987) Elastically deformable models. Proc SIGGRAPH Comput Graph 21(4):205–214

Thompson PM, Toga AW (1997) Detection, visualization and animation of abnormal anatomic structure with a deformable probabilistic brain atlas based on random vector field transformations. Med Image Anal 1(4):271–294

Toennies KD, Rak M, Engel K (2014) Deformable part models for object detection in medical images. Biomed Eng Online 13(1):1

Toussaint GT (1978) The use of context in pattern recognition. Pattern Recognit 10(3):189–204
Vu N, Manjunath BS (2008) Shape prior segmentation of multiple objects with graph cuts. In: IEEE computer society conference computer vision pattern recognition (CVPR 2008), pp 1–8
Wang H, Suh JW, Das SR, Pluta JB, Craige C, Yushkevich PA (2013) Multi-atlas segmentation with joint label fusion. IEEE Trans Pattern Anal Mach Intell 35(3):611–623
Wang L, Shi F, Li G, Gao Y, Lin W, Gilmore JH, Shen D (2014) Segmentation of neonatal brain MR images using patch-driven level sets. Neuroimage 84:141–158
Xia Y, Chandra SS, Engstrom C, Strudwick MW, Crozier S, Fripp J (2014) Automatic hip cartilage segmentation from 3D MR images using arc-weighted graph searching. Phys Med Biol 59(23):7245
Xu C, Prince JL (1998) Snakes, shapes, and gradient vector flow. IEEE Trans Image Process 7 (3):359–369
Yazdanpanah A, Hamarneh G, Smith BR, Sarunic MV (2011) Segmentation of intra-retinal layers from optical coherence tomography images using an active contour approach. IEEE Trans Med Imaging 30(2):484–496
Yedidia JS, Freeman WT, Weiss Y (2003) Understanding belief propagation and its generalizations. Explor Artif Intell New Millenn 8:236–239
Yin Y, Zhang X, Williams R, Wu X, Anderson DD, Sonka M (2010) LOGISMOS—layered optimal graph image segmentation of multiple objects and surfaces: cartilage segmentation in the knee joint. IEEE Trans Med Imaging 29(12):2023–2037
Zeng X, Staib LH, Schultz RT, Duncan JS (1999) Segmentation and measurements of the cortex from 3-d MR images using coupled-surfaces propagation. IEEE Trans Med Imaging 18 (10):927–937
Zhang S, Zhan Y, Metaxas DN (2012) Deformable segmentation via sparse representation and dictionary learning. Med Image Anal 16(7):1385–1396
Zienkiewics OC, Taylor RL, Zhu JZ (2005) The finite element method: its basis & fundamentals, 6th edn. Elsevier

Classification and Clustering

12

Abstract

Assigning semantics to segments is required if segmentation has not been combined with object detection. Classification is then based on evaluating segment attributes such as shape and appearance. The dimension of feature space is often high (>10) and the number of samples to train a classifier or to deduce a clustering is low. Methods are different compared to classification or clustering of pixels or voxels. For the most part, likelihood functions are not estimated and the classification criterion is directly based on the training data. Feature reduction techniques, classifiers, and clustering methods that focus on analysis in sparse feature spaces are the topic of this chapter. Among the different methods treated in this chapter, deep convolutional neural networks stand out, as their variable applicability has been found widespread use that

Concepts, Notions, and Definitions Introduced in this Chapter

- Feature reduction: PCA, ICA, Fisher's discriminant analysis, auto-correlation;
- Distance-based classification: minimum distance classifier, kNN classifier;
- Decision boundaries: linear decision boundaries, backpropagation networks, support vector machines;
- (Deep) convolutional neural networks;
- Discriminant functions;
- Agglomerative clustering;
- Fuzzy clustering;
- Decision trees;
- Random forests;
- Multiple instance learning;
- Bagging and boosting.

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_12

encompasses applications on a pixel basis. The methods presented here complement the methodology presented in Chap. 7.

Structures of interest in an image may be divided into different classes based on their segment properties. In medical image analysis, classification and segmentation sometimes mix, since class membership may also be assigned based on pixel attributes (see Chap. 7). *Classification* discussed in this chapter relates to the assignment of class membership to samples—which could be segments—based on their features. The major difference in classification integrated in segmentation is that the number of samples is often smaller and the number of features per sample is often larger. This has consequences for the classification strategy. If pixels are treated as samples, their large number and the low dimension of feature space result in a fairly accurate estimate of underlying probabilities. Training becomes more difficult when the dimensionality of feature space is high and few training samples exist. Classification resorts to strategies that do not estimate the probability distribution functions for the complete feature space, but

- compute a posteriori probabilities only for those feature values of samples to be classified, or
- compute the boundary at which two a posteriori probability functions have the same values, i.e., the boundary where class label assignment changes.

This will be discussed in the following sections. It will be followed by a treatment of clustering techniques that merely detect patterns in the data that may be influential for classification. Often, clustering is a first step to analyze the significance of features.

12.1 Features and Feature Space

A segment may be characterized by different types of attributes that make up the feature space:

- Average gray values or—as it is the case of some MR images—a vector of average gray values from different channels represent a (possibly known) relationship between measurement and membership to an object class.
- Second or higher order statistics such as gray level variance within the segment represent possible object-specific variations of the measurement value within an object class or object-specific measurement artefacts.
- Texture features of the gray level distribution in the segment represent tissue characteristics. They may also be important attributes for differentiating between different states of an organ's class (such as healthy organs vs. pathologic variations).
- Shape features such as size and elongation of a segment or ASM mode variation may typify a segment as well. If the segment represents the complete object (and not a part of it), shape features may help to determine the type of object.

The first three types of features may be reliable, even if the segment comprises only a part of the object to be classified as long as it is mainly located within the object. Shape features are only reliable if the segment boundary closely follows the object boundary.

In general, any feature that contributes to the characterization of a segment may be included in the feature vector. Not all of the features may be necessary, and some redundancy will be included in such a representation.

12.1.1 Linear Decorrelation of Features

Features $f_1, \ldots, f_N$ of a vector $\mathbf{f} = (f_1, f_2, \ldots, f_N)$ in N-dimensional feature space are initially assumed to be independent of each other. The dimensionality of feature space is often much higher than the two or three dimensions that can easily be rendered by some visualization technique. Thus, interactive techniques that display a feature distribution of samples and ask the user to identify classes based on visible clustering are difficult to implement. Sample distributions in feature space could be projected on 2D subspaces and then displayed, but assessing distributions in N dimensions from a series of 2D projections is very difficult for $N \gg 2$.

The high dimensionality of feature space complicates automatic classification as well, because most techniques rely on applying the Bayesian Theorem. The necessary probability distributions are often not known in advance and need to be estimated from classified samples. This training of the classifier requires a certain density of sample distribution for delivering reliable estimates (see Fig. 12.1).

For a given number of samples in the training database, the density decreases with increasing number of feature dimensions. In such case, estimates from a feature histogram may be difficult to generate (see Fig. 12.2). However, initial feature selection often accumulates all computable attributes of a segment that may

Fig. 12.1 If the number of samples per feature is high, it is easy to reliably estimate a probability distribution from samples

Fig. 12.2 It is difficult to approximate an unknown distribution from samples if the histogram shows few occurrences in a high-dimensional feature space

Fig. 12.3 Feature reduction from original space **a** can be done by simply removing features with the least discriminative power **b** or by first de-correlating the data and then applying reduction in the de-correlated space **c**

contribute to the identification of the object class. The initial assumption of uncorrelated features may not be true.[1]

The simplest way to reduce the dimensionality of the feature space is a repeated removal of features that produces the least increase in the classification error. Feature reduction is more efficient if features are decorrelated first and the dimension of feature space is reduced afterwards (see Fig. 12.3).

Linear decorrelation and subsequent feature reduction can be carried out by *principal component analysis* (*PCA*) in feature space [see Sect. 14.3, a detailed treatment can be found in Abdi and Williams (2010)]. If features are correlated, the distribution of samples in feature space actually occupies a lower-dimensional subspace. The PCA produces an orthogonal transformation in feature space such that all covariance values between features are zero. Coordinate axes after transformation are aligned or orthogonal to this subspace (see Fig. 12.4). Features corresponding to orthogonal axes to the subspace can be identified and removed.

[1]Consider, for instance, Haralick's texture features from the co-occurrence matrix. Many of the features measure quantities which may behave similar for a class of segments. The same may be true for shape features. E.g., the size of a segment may be closely related to its elongatedness if segments are cells in a microscopic image of two types of which the small ones are mostly circular while the larger ones are not.

Fig. 12.4 PCA produces an axis system that decorrelates the data. It is oriented along the data distribution in feature space. Features of this new coordinate system may be removed if projection on the remaining axes produces only a small error

12.1.2 Linear Discriminant Analysis

Using PCA for feature reduction can be tricky because the decomposition does not tell anything about class separability after reduction. This would require knowledge on class membership from a set of classified training samples. A simple strategy is to select features such that the within-class scatter is minimized and the between-class scatter is maximized for some distribution of samples in feature space. The technique is known as *linear discriminant analysis* (LDA) or *Fisher's discriminant analysis* (Fisher 1936).

The *within-class scatter matrix* $\mathbf{\Sigma}_W$ is defined as (see Fig. 12.5)

$$\mathbf{\Sigma}_W = \sum_{k=1}^{K} P(k) \cdot \mathbf{\Sigma}_k, \tag{12.1}$$

where $P(k)$ is the a priori probability of a sample belonging to class k and $\mathbf{\Sigma}_k$ is the covariance matrix for the distribution of samples of class k in feature space. The a

Fig. 12.5 Within-class scatter matrix is computed from adding covariance matrices of the different classes weighted by their a priori probability

Fig. 12.6 Between-class scatter matrix is computed from the vectors from class centers $\bar{\mathbf{f}}_i$ to the center $\bar{\mathbf{f}}$ of all classes

priori probabilities can be estimated from the ratio of samples belonging to class k to the number of all samples. The covariance matrix can be estimated as above, but now including only feature values of samples of class k.

The *between-class scatter matrix* $\mathbf{\Sigma}_B$ is defined as (see Fig. 12.6)

$$\mathbf{\Sigma}_B = \sum_{k=1}^{K} P(k)\left(\bar{\mathbf{f}}_k - \bar{\mathbf{f}}\right)\left(\bar{\mathbf{f}}_k - \bar{\mathbf{f}}\right)^{\mathrm{T}}, \quad \mathbf{f} = \sum_{k=1}^{K} P(k)\mathbf{f}_k. \tag{12.2}$$

It measures the scatter of average feature vectors $\bar{\mathbf{f}}_k$ for the different classes k with respect to the average feature vector $\mathbf{f}$ of the whole data set.

Feature reduction should produce new features $\mathbf{f}'$ with improved class separability by maximizing the between-class scatter and minimizing the within-class scatter. A new set of orthogonal feature axes is selected so that the combined variances of $\mathbf{\Sigma}_B$ and the inverse of $\mathbf{\Sigma}_W$ given by the sum of diagonal elements of their product $\mathrm{tr}\left(\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B\right)$ (the *trace* of the matrix) are maximal. Hence, using a similar argumentation than for computing the PCA, eigenvectors and eigenvalues of $\mathbf{\Sigma}_W^{-1}\mathbf{\Sigma}_B$ are computed. Eigenvectors are the new basis onto which features $\mathbf{f}$ are projected yielding $\mathbf{f}'$. Features with large eigenvalues are selected for subsequent classification.

12.1.3 Independent Component Analysis

Transformation in feature space, as discussed above, mainly attempts to reduce redundancy in the data for the purpose of assigning classes to samples. Sometimes, however, each sample may contain a mixture of two or more components. No class assignment is desired. Instead, the ratio of this mixture shall be determined. This scenario may arise, e.g., in time signal analysis describing organ function. Functional units are often much smaller than the image resolution, and information carriers would be subpixels. As they cannot be measured, analysis attempts to

Fig. 12.7 Opposed to PCA, ICA produces a system of linear independent features. No prediction about the value of one feature can be made based on the value of another feature

decompose the measured signal into its unknown components. This is possible if the different components are independent and some additional assumptions hold. The procedure is called *independent component analysis* (*ICA*) (Hyvärinen and Oja 2000; Hyvärinen et al. 2009), see Fig. 12.7.

Opposed to PCA, a closed-form solution for computing the independent components does not exist. ICA methods are iterative optimization methods that are not guaranteed to find the global optimum. Independent component analysis assumes that

- a feature vector $\mathbf{f}$ consists of a number of independent components $\mathbf{f}_{c1}, \ldots, \mathbf{f}_{cn}$.
- the number of components is known.
- their distribution is non-Gaussian.

The procedure consists of two steps. First, independent components are computed from the feature distribution in some data set. Then, each feature vector is projected on the independent components in order to determine the composition in every sample.

Computation of independent components from observed feature values f_i is done by finding feature axes $\boldsymbol{f'}$ that minimize the Gaussianity of the decomposed data set. The underlying assumption is a general observation about probability. A probability distribution of some stochastic process approaches a Gaussian distribution if the number of independent causes for the process increases.[2] In reverse, a distribution is less Gaussian if it consists of fewer independent components (see Fig. 12.8). ICA estimates Gaussianity of the current set of axes and rotates one of the axes so that

[2]The central limit theorem states that the probability density function of a sum of independent random variables having the same but unknown distribution characteristics will approximately be a Gaussian function if the variance for each variable is finite.

Fig. 12.8 Distribution of dependent components **a** resembles a Gaussian, whereas truly independent data **b** does not

Gaussianity is minimized. This process is repeated until convergence. Gaussianity can be measured, for instance, by computing the absolute value of *kurtosis excess* γ_2 (Hyvärinen and Oja 2000)

$$\gamma_2 = \frac{E\left((f-\bar{f})^4\right)}{E\left((f-\bar{f})^2\right)^2} - 3, \tag{12.3}$$

which measures the peakedness of a distribution. A distribution with a high peak is called leptokurtic and has values of $\gamma_2 < 0$, one with a flat peak is called platykurtic with $\gamma_2 > 0$. The Gaussian distribution is mesokurtic with $\gamma_2 = 0$. Maximizing $|\gamma_2|$ therefore minimizes Gaussianity. Another option for computing Gaussianity is to use the negative entropy, since entropy is maximal for a Gaussian distribution. Further measures to be used in ICA can be found, e.g., in Blanco and Zazo (2003).

12.2 Bayesian Classifier

Given features of a sufficient number of samples, the Bayesian Theorem can be directly used for computing class membership. Stated in the Bayesian framework, the conditional probability $P(c_k|\mathbf{f})$ that a sample belongs to class c_k given its feature values $\mathbf{f}$ depends on the likelihood $P(\mathbf{f}|c_k)$ of obtaining $\mathbf{f}$ for a member of class c_k and the a priori probability $P(c_k)$ of a member of class c_k being present:

$$P(c_k|\mathbf{f}) = \frac{P(f|c_k)P(c_k)}{P(\mathbf{f})}. \tag{12.4}$$

The normalizing probability $P(\mathbf{f})$, also called marginal probability, may be dropped for classification, since $P(\mathbf{f})$ is independent of c_k. Classification of a sample with feature vector $\mathbf{f}$ requires choosing the class c_{opt} with

$$c_{\text{opt}} = \arg\max_{c_k} P(c_k|\mathbf{f}) = \arg\max_{c_k} P(f|c_k)P(c_k). \tag{12.5}$$

Likelihood function and the a priori probability have to be estimated from training data or specified using expert knowledge.

The normalized multi-dimensional histogram of binned feature values generated from classified training samples may be used to estimate $P(\mathbf{f}|c_k)$. A rule of thumb says that the number b of bins along an axis in N-dimensional feature space should relate to the number of available samples M by

$$b = M^{\frac{1}{N+1}}. \tag{12.6}$$

Even if feature space is reduced, the number of classified samples may still not be adequate for a reliable estimate of the likelihood function.

Various techniques exist for constraining the range of possible likelihood functions without being overly restrictive. If the number of classified samples is rather large, it can be assumed that every sample represents the expected value of a Gaussian distribution of known variance in feature space. This corresponds to convolving the histogram with a Gaussian kernel with zero mean and this variance. This is a special case of a *kernel density estimator* (Rosenblatt 1956; Sarle 1994), which computes a likelihood function $f_h(x)$ from samples x_i, $i = 1, \ldots, N$ that are drawn from this distribution by convolving the samples with a kernel function K

$$f_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right). \tag{12.7}$$

In the case of the Gaussian kernel (see Fig. 12.9), this function is given by

$$K\left(\frac{x - x_i}{h}\right) = \frac{1}{\sqrt{2\pi h}}\exp\left(-\left(\frac{x - x_i}{h}\right)^2\right). \tag{12.8}$$

If it is known that features are uncorrelated, e.g., after carrying out PCA analysis and transforming the feature space,[3] estimation of the likelihood function becomes

[3]It should be remembered that the PCA is based on an estimate of the covariance matrix. The reliability of this estimate depends on the dimensionality of feature space and the number of samples.

Fig. 12.9 Convolving the histogram of a feature distribution with Gaussian kernels with increasing variance produces an increasingly smoother estimate of the underlying distribution

much easier. In this case, $P(f|c_k) = \prod_{i=1}^{N} P(f_i|c_k)$ and the 1D likelihood functions can be computed separately for each feature.

Often, however, the likelihood function is constrained even more by assuming that it is a multivariate Gaussian distribution. Variation of feature values can often be assumed to be caused by a number of independent influences with appropriate properties so that we can make use of the Central Limit Theorem again. Assuming Gaussian distributions reduces necessary estimations to the computation of the expected value $\bar{\mathbf{f}}$ and the covariance matrix $\mathbf{\Sigma}$ of $\mathbf{f}$. The problem is further simplified if features are uncorrelated so that only their variances have to be estimated.

Computing the a priori probabilities $P(c_k)$ is usually easier. If it can be assumed that the samples in the training data set are representative for the population, the ratio of the number of samples of class c_k to the total number of samples estimates $P(c_k)$.

12.3 Classification Based on Distance to Training Samples

The number of training samples may not suffice for a good estimate of the feature likelihood function. On the other hand, in most regions of the feature space, the difference between the most probable class assignment and its next competitor is very large. Estimation errors will not affect the classification result. Hence, several methods directly estimate the a posteriori probability using some simplifying assumptions.

Minimum distance classification requires univariate Gaussian feature likelihood functions with equal variances for each class. The a posteriori probability of a sample with feature vector $\mathbf{f}$ belonging to some class c_{opt} is maximized by selecting the class with average feature vector $\bar{\mathbf{f}}_k$ to which it is the closest (see Fig. 12.10):

$$c_{\text{opt}} = \arg\min_{c_k} \left\| \mathbf{f} - \bar{\mathbf{f}}_k \right\|. \tag{12.9}$$

Computing the minimum distance classifier is slightly more costly than Bayesian decision making as the computational cost increases linearly with the number of class centers. However, it does not require estimates of the likelihood function. The estimates for average values $\bar{\mathbf{f}}_k$ may be improved when newly classified samples are added to the training database.

Fig. 12.10 Minimum distance classifier assigns a class to an unknown sample, of which the distance to the class center (the expected value) is closest

Requiring univariate Gaussians with equal variances is rather restrictive, however, considering scatter plots of most distributions. A more accurate estimate for the a posteriori probability can be generated when the actual distribution of training samples is taken into account. In order to approximate the probability density function for some class c_k at a location **f** in feature space, the function is assumed to be locally constant in some vicinity f_δ around **f**. Using f_δ as bin size, the normalized histogram serves as a estimate for $P(c_k|\mathbf{f})$.

Samples may be distributed so sparsely that the size of f_δ needs to be large, and constancy can no longer be assumed. Hence, classification schemes adapt neighborhood size according to local density of training samples.

The *nearest neighborhood classifier* uses the smallest possible neighborhood size (see Fig. 12.11). The neighborhood around some sample **f** is the smallest hypersphere that contains the closest training sample to **f**. If the class of the nearest

Fig. 12.11 Nearest neighborhood classifier computes the closest classified sample to a sample with unknown class and classifies the sample accordingly

sample is c_{near}, the estimate of $P(c_{\text{near}}|\mathbf{f})$ is 1 and $P(c|\mathbf{f}) = 0$ for all other classes c. Class c_{near} is assigned to the sample with features $\mathbf{f}$.

Using the nearest neighbor is a fairly rough estimate of the true probabilities and may deliver wrong results if the likelihood functions for the different classes have a complex shape. Hence, the adaptive neighborhood can be increased to include the first k samples. This is the *k-nearest neighborhood* (*kNN*) *classifier* (see Fig. 12.12). Given that l_k of the k training samples belong to class c_i, the estimate of its probability is $P(c_i|\mathbf{f}) = l_k/k$. The kNN classifier is a voting mechanism where each of the k samples votes for a certain class. The class receiving the most votes is assigned to the sample with feature vector $\mathbf{f}$.

The quality of the estimate improves with the number of samples n and the number k of voting samples, if n increases faster than k. If k and the ratio n/k approach infinity the classification error approach that of Bayesian classification with known a priori probabilities and known feature likelihood functions.

Computing the result of a kNN classifier can be time-consuming. In order to find the nearest neighbors to a given sample, distances to all samples in the training database have to be computed. Many efficient schemes exist [e.g., the branch-and-bound strategy in Jiang and Zhang (1993), the pre-computation of a distance transform in Warfield (1996) or the approximation in the wavelet domain in Hwang and Wen (1998)], but the computational cost still increases with the size of the training database. This can be critical if the accuracy of the computation is to be improved by including classified samples into the database. The error rate may drop with time as the average density of samples in feature space increases, but the computation time increases. This is particularly disturbing when the user is not aware of the underlying reasons, since the system seems to slow down with age.

Continued training can be achieved at lesser cost if density is increased selectively. The strategy is based on the observation that accuracy does not need to be equal everywhere in feature space. A sample is included in the training database, but it is marked as passive if its classification was determined by unanimous vote

Fig. 12.12 kNN classifier classifies an unknown sample according to the class memberships of the k nearest neighbors (shown here for $k = 3$)

Fig. 12.13 **a** Newly classified samples are only added to the database if they are close to the class boundary. **b** Votes can be weighted with distance in order to account for their reliability with respect to the classification decision

(see Fig. 12.13a). The kNN classifier only uses active training samples. The strategy increases the sampling density at decision boundaries between classes where votes are seldom unanimous. Passive samples have to be reviewed from time to time after new samples have been added to the active training database.

As the kNN classifier uses a distance metric to compute the number of votes for a class, it may be necessary to normalize feature values with their standard deviation before computing nearest neighbors. However, if a high variance of a feature value indicates a higher importance of this feature, normalization should not be carried out.

Classification based on feature values may be unreliable if

- the data values are too small,
- the vote difference is too small, or
- the sample is too far away from any classified samples in the training data.

This reliability can be measured, and if it is too low, a sample may be rejected. Votes can also be weighted with the distance to the sample which automatically causes reliability to be low if a sample is too far away from any classified sample (see Fig. 12.13b).

12.4 Decision Boundaries

We already observed that estimates for a posteriori probabilities have to be most accurate at decision boundaries. In fact, the decision boundary alone, i.e., the curve of equal a posteriori probability for at least two classes suffices if a method exists

for determining on which side of such boundary the feature vector of an unknown sample is located. Techniques for finding and applying decision boundaries will be discussed in this section. Most of the discussion will be based on a so-called *two-class problem*, i.e., the assignment of an unknown sample to one of two classes. Extensions to more than two classes will be discussed where appropriate.

All of the methods described below have in common that the search for a decision boundary is formulated as an optimization task that is usually solved iteratively. In general, the solution strategy does not guarantee that the global minimum, i.e., the optimal decision boundary, is found. This makes the use of decision boundaries sometimes inferior to the methods described in the previous section. The major advantages of decision boundaries are

- that their computation considers the need for increased accuracy at the boundary between two class regions.
- that the classification is faster requiring only constant computational cost once the decision boundary is defined.

12.4.1 Adaptive Decision Boundaries

If classes are well-separated in feature space, linear *adaptive decision boundaries* may serve as a classifier. A single linear decision boundary solves a two-class problem. It is represented by a linear equation in feature space

$$D(\mathbf{f}, \mathbf{w}, w_0) = \mathbf{w}^{\mathrm{T}}\mathbf{f} + w_0, \tag{12.10}$$

where $\mathbf{w} = (w_1, w_2, \ldots, w_N)$ and w_0 are weights determining the position of the hyperplane $D(\mathbf{f}, \mathbf{w}, w_0) = 0$ that separates the two classes. The constant w_0 is called the *bias* term. It is proportional to the distance of $\mathbf{f}$ to the hyperplane if $D(\mathbf{f}, \mathbf{w}, w_0) \neq 0$. $D(\mathbf{f}, \mathbf{w}, w_0)$ is positive if $\mathbf{f}$ is on the side of the hyperplane to which its normal points. Otherwise, it is negative.

Classification into two classes requires weights to be found such that classified training samples are separated by the hyperplane. The iterative algorithm depicted in Fig. 12.14 will compute the hyperplane if it exists, i.e., if the classes are linearly separable (see Fig. 12.15 for an illustration).

Restricting the number of iterations irrespective of the current classification error is necessary because of several reasons:

- The samples may not be linearly separable. In this case, no hyperplane exists, and the algorithm will only stop after the maximum number of iterations is reached.
- The correction factors c and k are set too high. The hyperplane oscillates around the correct solution because most correction steps are pushing the hyperplane too far into the opposite direction.

initialize weights $w_0,\ldots,w_M$ with small random values

repeat until classification is perfect, average error is smaller than ε_{min} **or** maximum number of steps is executed

select sample **f** from training set and compute $D(\mathbf{f},\mathbf{w},w_0)$

if class(**f**) $\neq$ $sgn(D(\mathbf{f},\mathbf{w},w_0))$ **then** ; with $c,k > 0$

$w_i := w_i + c \cdot class(\mathbf{f}) \cdot f_i$

$w_0 := w_0 + c \cdot class(\mathbf{f}) \cdot k$

end_if

end_repeat

class(**f**) returns -1 if the sample belongs to the first and +1 if it belongs to the second class
sgn() is the signum function.

Fig. 12.14 Sketch of the loop for computing weights of a linear decision boundary

Fig. 12.15 If a selected sample is not classified correctly, the decision boundary is moved towards the incorrectly classified sample

- The correction factors are set too low. The convergence to the solution is too slow and may not reach convergence after the maximum number of iterations.

Finding c and k is critical, and it is even more so the narrower the range of acceptable hyperplanes is, i.e., the closer the two classes are to each other.

If classes are not linearly separable, they may be separated by a nonlinear boundary. However, computing distances to a nonlinear boundary is much more complex than computing the distance to the hyperplane. It gets simpler if the nonlinear boundary can be embedded in a higher-dimensional space in such a way that the problem becomes linear. An example is a quadratic decision surface (see Fig. 12.16) in two-dimensional feature space where the embedding is

Fig. 12.16 A quadratic decision boundary may separate classes that are not linearly separable

$$D(\mathbf{f}, \mathbf{w}, w_0) = w_0 + w_1 u_1 + w_2 u_2 + w_3 u_3 + w_4 u_4 + w_5 u_5, \tag{12.11}$$

with

$$u_1 = f_1, \quad u_2 = f_2, \quad u_3 = f_1 f_2, \quad u_4 = f_1^2, \quad u_5 = f_2^2. \tag{12.12}$$

Similar embeddings can be created for other polynomial surfaces, but the dimensionality of feature space increases rapidly. Hence, embeddings are usually restricted to a low-dimensional feature space and polynomials of low order.

Decision boundaries can solve multiple class problems. A decision boundary D_{ij} needs to be trained for every pair of classes i and j. An unknown sample is inserted into all equations. It is classified as the one class that is not ruled out by any of the equations. The number of decision boundaries increases with the square of the number of classes. Furthermore, regions exist where all classes are ruled out. Samples in such region are either classified as *undecided* or submitted to a second classifier (e.g., by computing the distance to the closest outruling class boundary).

12.4.2 The Multilayer Perceptron

Backpropagation neural networks deal with nonlinear decision boundaries and with multiple class problems more efficiently than the decision boundaries described above. They are also called *multilayer perceptrons* (MLP) as they consist of a sequence of perceptrons.

Perceptrons were introduced early in computer science (Rosenblatt 1958). They are a single layer neural network with all input nodes connected to a single output node. They model a linear decision boundary. Nevertheless, they introduce the concept of a neural network and will be described in more detail before continuing to the discussion of multilayer networks.

Any neural network can be represented by a graph with nodes—the cell body of the artificial neural network—and edges connecting the nodes—the dendrites. Two

great introductions into neural networks in general and their use for pattern recognition can be found in Bishop (1995) and Duda et al. (2001).

12.4.2.1 Structure of the Multilayer Perceptrons

A neural network is hierarchical if it does not allow cycles in the graph. A hierarchical network is *layered* if nodes are organized in layers so that each node of a given layer is only connected to nodes of the previous layer—from which it receives input—and to nodes of the next layer—to which it gives input (see Fig. 12.17a). The input nodes of a perceptron are connected to a single output neuron.

Nodes "fire", i.e., they give a signal through all their outgoing edges to nodes of the next layer based on input that they receive and based on a node-specific activation function f (see Fig. 12.17b).

In order to mimic the behavior of a nerve cell, this firing can be implemented as a threshold on the input. The input is produced by a gathering function g that integrates input received from other nodes. Input originates from firing nodes in the previous layer. The signal is weighted by edge-specific weights connecting the firing nodes with the receiving node. If we label nodes by layer l and their position k in the layer and if we assume a fully connected network, gathering is

$$g\left(n_{l,k}\right) = \sum_{i=0}^{K(l-1)} w_{l-1,i} f\left(n_{l-1,i}\right). \tag{12.13}$$

The number of nodes at layer $l - 1$ is $K(l - 1)$. The input from nodes $n_{l-1,0}$ is a constant representing the bias. Firing is

$$f\left(n_{l,k}\right) = \begin{cases} 1, & \text{if } g\left(n_{l,k}\right) > t \\ 0, & \text{otherwise,} \end{cases} \tag{12.14}$$

Fig. 12.17 **a** Structure of a layered neural network. **b** Nodes fire if the gathered signal exceeds some threshold defined by the activation function

where t is a threshold that determines when to fire. Node layers are counted from 1 to L with L being the output layer. The layer $L = 0$ receives input from the feature vector $\mathbf{f}$

$$g(n_{1,k}) = \sum_{i=0}^{K(0)} w_{0,i} f_i, \tag{12.15}$$

where f_i, $i = 1$, $K(0)$ is the i-th feature of $\mathbf{f}$ with length $K(0)$ and $\mathbf{f}_0 = 1$ is the input bias.

It is now easy to see that, with firing threshold $t = 0$, a perceptron is a linear decision boundary. It is trained in the same fashion and solves the same kind of classification problems.

If a sequence of perceptrons is coupled, the capability of representing various decision boundaries increases dramatically (see Fig. 12.18). If the first layer feeds in several nodes and a second layer is added with a single node, the decision surface at the second layer is a convex, not necessarily closed polyhedron. Each of its bounding surfaces is determined by the output of one of the nodes in the previous layer. By increasing the number of nodes in the previous layer, the bounding polyhedron can approximate any convex surface with arbitrary precision. If several classes are to be separated, extra nodes per class representing their own decision boundary are created in the second layer.

Arbitrary concave decision boundaries can be represented by adding a third layer. By addition and subtraction of the convex figures from the previous layer, any decision boundary can be approximated with arbitrary precision, provided that the number of nodes in the first and second layer is arbitrarily large.

The middle layer of a three-layer perceptron is called *hidden layer*. Although it can be shown that three layers suffice to approximate any decision boundary, multilayer perceptrons may contain more than one hidden layer. This accommodates different views on the data, since each new layer adds a level of abstraction to the description of the decision boundary. The first layer separates space by hyperplanes, the second layer combines hyperplanes to convex figures, the third combines these to convex or concave figures, and the fourth combines concave

Fig. 12.18 **a** A single layer enables linear separation in feature space. **b** Adding nodes in an additional (hidden) layer, arbitrary convex decision boundaries can be represented. **c** With two hidden layers, convex figures can be combined to represent arbitrary concave decision boundaries

figures, and so on. Increasing the number of nodes within a layer adds capabilities to describe details of the decision boundary at the current level of abstraction. If a problem is known to be separable by a sequence of decisions, it is often advisable to choose a number of layers that corresponds to the number of hierarchy levels of this sequence.

Let us assume that the $K(L)$ nodes of the last layer L of a multilayer perceptron represent classes to which samples in feature space shall be assigned. These nodes could code the classification result in any convenient way. One could, for instance, use a binary code so that class membership to some class k would be a node pattern of the output nodes, which is the binary code of k. A simpler way would be that the desired output for a training sample belonging to class k is a vector $\mathbf{y}$ with $K(L)$ elements y_i such that $y_i = 1$ if $i = k$ and $y_i = 0$ otherwise. If the network is untrained, the output from an input feature vector $\mathbf{x}$ will differ from the desired pattern $\mathbf{y}$. The task is to find weights $w_{l,k}$ that minimize $\|\mathbf{y} - \mathbf{x}\|^2$ for all samples of the training data set.

12.4.2.2 Training of a Multilayer Perceptron

Nothing has been said so far about the training of a multilayer perceptron. The simple algorithm for linear decision boundaries is not applicable. In turns out, however, that a generic method exists that computes a gradient descent on the weights for an arbitrary multilayer perceptron. It requires that the error function is differentiable. Hence, the thresholding in the activation function is replaced by the *sigmoid function* (see Fig. 12.19)

$$f(n_{l,k}) = \frac{1}{1 + \exp(c \cdot g(n_{l,k}))}. \tag{12.16}$$

It introduces fuzzyness into the decision process. The function asymptotically approaches 0 for $f() \to -\infty$ and it approaches 1 for $f() \to \infty$. The rate of change is fastest at the origin. The steepness at the origin is controlled by c. The derivative of the sigmoid function is

$$f'(n_{l,k}) = f(n_{l,k})(1 - f(n_{l,k})). \tag{12.17}$$

It has been shown that the partial derivatives of the error function with respect to the weights can be computed by *backpropagating* the error from the output nodes into the network.[4] Iterative minimization starts with a feedforward step at each iteration n. It produces the output vector $\mathbf{x}^{(n)}$ using the current weights.

[4]The proof of this is a bit lengthy and will not be shown here. For a detailed treatment of backpropagation networks see Bishop (1995). The reasons why backpropagation produces a gradient descent are, briefly, the following: At each node, the gathering function sums up weighted results from the previous node and the derivative of a sum of functions is the sum of its derivatives. Fortunately, for computing the derivative of some weights, all previous weights can be treated as constants in the derivation. Applying the activation function to the sum is the application of a function to another function so that the chain rule applies to its derivative.

Fig. 12.19 Replacing a threshold-based activation function by the differentiable sigmoid function enables training of the backpropagation by gradient descend

Backpropagation then distributes the error correction, beginning at the output layer, back into the network. First, the derivative of the error $\|\mathbf{x}^{(n)} - \mathbf{y}\|$ with respect to the weights between previous and last layer is computed. These weights are corrected accordingly and used to propagate the error into the previous layer. Error correction at this layer is again done by computing and applying partial derivatives of the error. Weight update of the current iteration terminates when the input layer is reached. For all but the last layer, computation of the derivative requires computation of derivatives in the higher layers. A sketch of the algorithm is depicted in Fig. 12.20.

The *learning rate* λ controls the speed with which learning converges. Large values for λ accelerate learning, but the probability of oscillation around a solution increases as well. Since gradient descent may be stuck in a local minimum, starting with a good guess for the weights improves chances of convergence to a satisfactory solution. In the absence of such knowledge, random values for the weights should be chosen in order to avoid unwanted bias.

Initialize all weights $w_{l,k}^{(0)}$ to small random numbers

choose a positive constant λ (the learning rate)

for all samples **do**

set $f_{0,1},\ldots,f_{0,K(0)}$ to feature values of selected sample

compute output vector $x_1,\ldots,x_{K(L)}$ at layer L with weights $w_{l,k}^{(n)}$

compute partial derivatives at L: $\delta_{L,k}^{(n)} = x_k^{(n)}\left(1 - x_k^{(n)}\right)\left(x_k^{(n)} - y_k\right)$

for all layers l=L-1,...,1 **do**

$$\delta_{L,k}^{(n)} = f_{l,k}^{(n)}\left(1 - f_{l,k}^{(n)}\right)\sum_{i=1}^{K(l)} \delta_{l,i}^{(n)} w_{l,i}^{(n)}$$

end_for

set $w_{l,k}^{(n+1)} = w_{l,k}^{(n)} - \lambda \cdot \delta_{l,k}^{(n)} f_{l,k}^{(n)}$ for all l,k

until weight change is insignificant

Fig. 12.20 Sketch of the backpropagation algorithm that realizes a gradient descent on the weights in the network

Convergence may be slow, especially if the network consists of many layers. Error distribution at every layer spreads the error more and more uniformly. Adding a *momentum* term to the correction may improve convergence speed:

$$w_{l,k}^{(n+1)} = w_{l,k}^{(n)} - \lambda \cdot \delta_{l,k}^{(n)} f_{l,k}^{(n)} + \alpha \left[w_{l,k}^{(n)} - w_{l,k}^{(n-1)} \right]. \tag{12.18}$$

The momentum term $\left[w_{l,k}^{(n)} - w_{l,k}^{(n-1)} \right]$ weighted by α pushes the result towards the direction of change from the previous iteration.

There are three kinds of failures that may happen when a backpropagation network is trained:

- The data is not separable because the number of layers and nodes is too small for an appropriate approximation of the decision boundary.
- A local minimum may be found that does not optimally classify the data. The reason could be
  - an inadequate network topology (relation of the number of layers to the number of nodes in the layers).
  - a learning rate that is too high or too low.
  - an initialization that is too far from the global optimum.
- The decision boundary overadapts to the training data (see Fig. 12.21). It perfectly separates training samples but fails to separate the test data. Reasons are
  - too many nodes and layers so that the network does not properly generalize.
  - too many iterations in the optimization so that the boundaries approximate the training samples too closely.

Iterative optimization is stopped based on the average error on the test data set. The test error usually starts to increase while the training error still decreases which signifies loss of generalization (see Fig. 12.22).

Fig. 12.21 If training is carried out for a prolonged time, the large number of degrees of freedom of a backpropagation may cause the system to over-adapt to the training data

Fig. 12.22 Criterion for stopping the learning should be chosen based on the classification error on the test data

In order to find just the right number of nodes and layers for a proper generalization, an initial network may be repeatedly changed by removing nodes. Convergence of an optimally generalizing network can be further improved when edges are removed that carry very low weights. These edges mostly contribute noise to the classification result.

12.4.2.3 Using the MLP as Auto-encoder

Auto-encoders attempt to reconstruct the features themselves instead of producing a classification result. An MLP trained as auto-encoder would have as many output nodes as input nodes (see Fig. 12.23). It would be trained by comparing the output of the network with the input features. Learning is unsupervised since it does not require any class labels or other desired output values assigned to the samples.

An auto-encoder can be used for efficient feature reduction. In this case, not the output layer, but one of the hidden layers contains the desired result. The hidden layer is designed such that is has fewer nodes than the input and output layers. The trained network is then cut at this hidden layer whose nodes serve as output for the decoder and as input for the encoder. Sparsity can be further enhanced by removing nodes from the hidden layer with low activation probability.

Fig. 12.23 An auto-encoder is an MLP that first reduces the number of features by several fully connected layers and then expands the result to the dimension of the original feature space. It is trained by seeking a decoding-encoding network that minimizes the difference between input and output

Fig. 12.24 Different, perfect separations of the training data lead to different classifications of unknown samples

12.4.3 Support Vector Machines

More than one boundary may separate the samples of a classification problem. *Support vector machines* (SVM) are able to select among several possibilities the one boundary that maximizes the distance to the samples (see Fig. 12.24 for the principle). An introduction to SVMs is Steinwart and Christmann (2008), a detailed in-depth view by two of the important contributors in the field is Schölkopf and Smola (2002).

Maximizing sample distances from the decision boundary is a useful property considering that training data only samples the true distribution. It reduces the chance of misclassification.

Linear support vector machines solve a two-class problem. Consider the classification problem for linear decision boundaries in Eq. (12.10). The classifier can be rewritten such that

$$D(\mathbf{f}, \mathbf{w}, w_0) \cdot d > 0, \quad d = \text{class}(\mathbf{f}) \in \{-1, 1\}. \tag{12.19}$$

This can be reformulated as

$$d \cdot \left(\mathbf{w}^{\mathrm{T}}\mathbf{f} + w_0\right) \geq 1, \tag{12.20}$$

since weights for strictly separable sample distributions can always be properly scaled to make the minimum greater than one. The distance r of some sample $\mathbf{f}$ from the hyperplane is

$$r = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{f} + w_0}{\|\mathbf{w}\|}. \tag{12.21}$$

The minimum distance for all samples $\mathbf{f}_i$ is then

$$\mu_L = \min_i \left(\frac{|\mathbf{w}^{\mathrm{T}}\mathbf{f}_i + w_0|}{\|\mathbf{w}\|}\right). \tag{12.22}$$

The training task is to find a hyperplane that maximizes μ_L. Given such hyperplane, some samples $\mathbf{f}_b$ will have exactly this distance μ_L to the hyperplane, i.e.

$$r_s = \begin{cases} \frac{1}{\|\mathbf{w}_s\|}, & \text{if } d_b = +1 \\ \frac{-1}{\|\mathbf{w}_s\|}, & \text{if } d_b = -1 \end{cases} \tag{12.23}$$

These vectors $\mathbf{f}_b$ are called the *support vectors* (see Fig. 12.25). Hence, no samples are in a corridor of width $2/\|\mathbf{w}\|$ implying that $\frac{\|\mathbf{w}\|}{2} = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$ subject to the constraint

$$\left(\mathbf{w}^{\mathrm{T}}\mathbf{f}_i + w_0\right)d_i \geq 1 \Rightarrow \left(\mathbf{w}^{\mathrm{T}}\mathbf{f}_i + w_0\right)d_i - 1 \geq 0 \tag{12.24}$$

has to be minimized for finding the optimal decision boundary.

Using Lagrange multipliers $\boldsymbol{\alpha}$ for solving a constrained optimization task delivers

$$J(\mathbf{w}, w_0, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} - \sum_{i=1}^{N} \alpha_i \left(d_i \left[\mathbf{w}^{\mathrm{T}}\mathbf{f}_i + w_0\right] - 1\right). \tag{12.25}$$

A series of operations is needed to arrive at this expression. Details can be found in literature about support vector machines, e.g., Schölkopf and Smola (2002). The goal is to find a saddle point for $J(\mathbf{w}, w_0, a)$. It is done by minimizing J with respect to $\mathbf{w}$ and w_0 keeping α fixed, hence, computing partial derivatives with respect to $\mathbf{w}$ and w_0 and setting them to 0. Equating the derivatives of $\mathbf{w}$ to zero delivers

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i d_i \mathbf{f}_i. \tag{12.26}$$

The value of $\mathbf{w}$ is a weighted sum of the feature vectors. Actually, most of the weights are zero and only the support vectors influence the results.

Fig. 12.25 Support vector machines maximize the distance from the decision boundary. The samples that define the course of the decision boundary are the support vectors

After computing $\mathbf{w}$ and w_0, the values for α are determined. It requires minimization of

$$Q(\alpha) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j\left((\mathbf{f}_i)^{\mathrm{T}}\mathbf{f}_j\right) + \sum_{i=1}^{N}\alpha_i \tag{12.27}$$

subject to the constraint that $\sum_{i=1}^{N}\alpha_i d_i = 0$ (resulting from minimization for w_0) and $\alpha_i \geq 0$. This can be done using methods of linear algebra on constrained optimization.

The result is then inserted into (12.26) for delivering $\mathbf{w}$. The bias term w_0 is computed by choosing a support vector $\mathbf{f}_b$ and computing

$$\mathbf{w}^{\mathrm{T}}\mathbf{f}_s + w_0 = 1 \Leftrightarrow w_0 = 1 - \mathbf{w}^{\mathrm{T}}\mathbf{f}_s. \tag{12.28}$$

Although this sounds difficult, it is nothing but application of standard linear algebra techniques. Linear support vector machines have the advantage over linear decision boundaries that they assert an optimal quality of the solution. This is probably the reason why they are popular especially if few training samples exist (as it is often the case in medical image analysis).

Nonlinear decision boundaries can be computed similar to the strategy discussed earlier by embedding the nonlinear space in a higher dimensional linear space. However, the beauty of using support vector machines lies in Eq. (12.27) for computing the Lagrange multipliers. If the original feature space is mapped on some higher dimensional space $\Phi(\mathbf{f}) = \mathbf{u}$, the new equation to be optimized becomes

$$\begin{aligned} Q(\alpha) &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j\left((\mathbf{u}_i)^{\mathrm{T}}\mathbf{u}_j\right) + \sum_{i=1}^{N}\alpha_i \\ &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j\left(\Phi(\mathbf{f}_i)^{\mathrm{T}}\Phi(\mathbf{f}_j)\right) + \sum_{i=1}^{N}\alpha_i. \end{aligned} \tag{12.29}$$

In other words, just the dot product $\Phi(\mathbf{f}_i)^{\mathrm{T}}\Phi(\mathbf{f}_j)$ has to be generated instead of spanning the space itself. If a simple function $K(\mathbf{f}_i, \mathbf{f}_j) = \Phi(\mathbf{f}_i)^{\mathrm{T}}\Phi(\mathbf{f}_j)$ exists, it replaces the dot product in Eq. (12.29). This function is called a *kernel function*. It enables to find a nonlinear decision boundary by embedding $\mathbf{f}$ in a higher dimensional space without actually carrying out the optimization in this space. This application of Mercer's theorem is sometimes called the *kernel trick* (Müller et al. 2001).

A number of such kernel functions exist and are used for nonlinear support vector machines. A polynomial surface with degree d is optimized using

$$K_{\leq d}(x, y) = \Phi_{\leq d}(\mathbf{x}) \cdot \Phi_{\leq d}(\mathbf{y}) = \left(1 + \mathbf{x}^{\mathrm{T}}\mathbf{y}\right)^d \tag{12.30}$$

Hence, the quadratic would be given by

$$\begin{aligned} K_{\leq 2}((x_1 \quad x_2), (y_1 \quad y_2)) &= \left(1 + \mathbf{x}^{\mathrm{T}}\mathbf{y}\right)^2 \\ &= (1 + x_1 y_1 + x_2 y_2)^2 \\ &= 1 + x_1 y_1 + x_2 y_2 + x_1^2 y_1^2 + 2x_1 x_2 y_1 y_2 + x_2^2 y_2^2 \end{aligned} \tag{12.31}$$

The dot product does not even have to be defined in Euclidean metrics. The following kernel (also known as *radial base functions* with variance σ^2)

$$K_\sigma(x, y) = \Phi_\sigma(\mathbf{x}) \cdot \Phi_\sigma(\mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2}\right) \tag{12.32}$$

uses the angle between support vectors as metrics. Samples are classified according to their similarity to the support vectors. Results are somewhat similar to that of discriminant functions discussed in the next section and to association networks used for clustering, discussed in Sect. 7.2.3.

Originally, classification by support vector machines requires that training data sets are separable. Otherwise, no decision boundary exists can be optimized. If some of the data is distorted so that features overlap in feature space, *slack variables* ξ_i can be introduced that make the problem separable again and therefore solvable.

Slack variables push the feature locations away from the boundary. This is done by modifying Eq. (12.24) to

$$\left(\mathbf{w}^{\mathrm{T}}\mathbf{f}_i + w_0\right) d_i \geq 1 - \xi_i. \tag{12.33}$$

It was shown by Cortes and Vapnik (1995) that the introduction of slack variables adds an extra term to Eq. (12.25), but does not change the kernel property of Eq. (12.27).

12.5 Convolutional Neural Networks and Deep Learning

Convolutional neural networks (CNN) have found much attention as they may be used to model deep learning concepts inspired by hypotheses about the processing of sensor input in biological systems. A concise introduction into CNNs is Zeiler and Fergus (2014).

Deep learning is a branch of machine learning that assumes a layered processing of sensor input where the most important information is filtered out from the processed input layer by layer. A CNN with a large number of layers (15–20 or even more) is called a *deep CNN* (dCNN) or *deep neural network* (since not all of

the layers are convolutional layers). It has been shown that layers far from the input layer are indeed capable to learn abstract concepts about the input that has been presented to the network.

Similar to multilayer perceptrons (MLP), dCNNs also process input data through several layers of nodes to arrive at the desired output. However, the processing in each convolutional layer is different in two aspects, *locality* and *shared weights*. Instead of connecting all nodes from the previous layer to all nodes in the next layer, input is processed only locally and several different processing nodes are trained for each layer. The concept has been derived from research in animal and human vision and models the function of receptive fields in early vision. Weights on connections between layers are shared, i.e., they are adapted for each node receiving input from its receptive field in the same manner. Results are then down-sampled in a pooling step and transferred to the next layer.

The localization at each step implements knowledge about the spatially local structure of most objects in images. This hypothesis of locality of information is not new as it has been exploited, for instance, when modeling information dependencies by Markov random field. An adaptive neural network that implements this idea was the neocognitron, presented by Fukushima (1980) which has later been improved by LeCun et al. (1989) for feature learning in handwritten digit recognition.

A convolutional neural network requires much fewer nodes than an MLP and much fewer weights per layer to be trained. As such, CNNs can be thought as an implementation of a local, multi-scale strategy with motivation similar to those of Gabór filter banks, multiresolution models, and the like. Using this strategy as part of a trainable network produces a powerful classifier. It was shown to outperform many traditional classification systems that often consist of trainable components which are carefully designed to the problem.

CNNs can be trained similar to MLPs, although especially for deep CNNs with many layers the amount of training necessary and therefore the cost of training can be substantial (hardware-based solutions result in acceptable training times nonetheless).

12.5.1 Structure of a CNN

A dCNN has a *feature extraction part* to compute local features and a feature-based *classification part* that trains relationships between features (see Fig. 12.26). Classification is carried out by a fully-connected multilayer perceptron. The preceding feature extraction part consists of a sequence of*building blocks* that compute and reduce feature maps.Each building block has one or several *convolution layers* with several feature maps followed by a*pooling layer*. A feature map in a convolution layer has approximately the same size than the input map. If it is part of the first hidden layer, the input map is simply the input image. Otherwise, it is the output of the previous layer. Each node of the feature map is connected by edges to a local neighborhood (this is the so-called *receptive field* of the CNN) in the input

Fig. 12.26 Structure of a dCNN contains a local feature analysis step, where convolution and pooling layers in several building blocks are used to find the most relevant local features. In the feature-based classification part, these features are then classified by a fully-connected MLP to produce the output result. In principle, dCNNs can be seen as extending the MLP by convolutional layers that produce optimal features from the input data

map. The size of the receptive field is a parameter to be specified by the developer. Recently, developers seem to prefer small neighborhoods (3×3, 5×5) as they were found to be effective and reduce the number of trainable parameters in the system. The local neighborhood is equal for all nodes. Adjacent nodes are connected to adjacent neighborhoods and neighborhoods overlap (see Fig. 12.27). The amount of overlap is called the*stride* of the convolutional neural network. A stride

Fig. 12.27 A building block of the feature extraction part of a dCNN consists of two types of layers, the convolution layer(s) and the pooling layer. Nodes in the convolution layer are connected to a local neighborhood (the receptive field) in the previous layer. The convolution layer contains several sublayers (three in our example) which learn different convolutions. The number of sublayers is called depth of the convolution layer. More than one convolution layer may exist in a building block. The pooling layer reduces the output before passing it to the next layer

of n would mean an offset of n between neighboring nodes in the output layer. Missing nodes at the boundary of the input map are usually been padded, although they may also be defined not to have the full neighborhood.

As in MLPs, the input is weighted by edge weights and a bias weight is added, then summed and subjected to an activation function (which could be the sigmoid function as in MLPs). Training of the network is subject to the additional restriction that weights are shared between all neurons in the receptive field. This means that corresponding edges of two nodes always have the same weights. It explains the name of the CNN as this implements a trainable convolution in each feature map. It has been shown that the gradient descent through backpropagation can be adapted, so that it can optimize network weights under this restriction. As in MLPs, training of the network consists of forward propagating sample features, i.e., images, to the output layer, back propagating the error, and correcting the weights during backpropagation.

For each convolution layer, several different feature maps are trained. Since training commences with randomly chosen weights and gradient descent finds just local minima, different feature maps will develop different features during training. For the description of CNNs the two-dimensional feature maps (for a 2d image) are stacked along an additional dimension which is called the *depth* dimension of the CNN. This is a parameter to be specified by the developer (depth values between 40 and 500 are common).

The final step of a CNN building block is called *pooling* and reduces the resolution of the output to serve as input to the next layer. Pooling is done in a separate layer following the convolution layer(s). Non-overlapping neighborhood regions in the pooling layer are connected to a single output node. A popular method to generate the output result is *maxpooling* where only the maximum response is submitted to the output of the pooling layer. Maxpooling introduces a certain amount of local translational invariance (in the range of the pooling region) into the network since the maximum response in the neighborhood region does not depend on location within this neighborhood. However, selection of the pooling method is a free parameter of the system. Other pooling methods such as applying the *softmax* function (Bishop 2006)

$$s_i = \frac{\exp(x_i)}{\sum_{k=1}^{N} x_k} \tag{12.34}$$

to an input node vector $\mathbf{x} = (x_1, \ldots, x_N)$ of nodes $1, \ldots, N$ or applying the L_2-norm to create a weighted output have been used as well.

An alternative to pooling is to replace it by an additional convolutional layer with a stride >1 as it has been proposed by Springenberg et al. (2014) who found that such a pure convolutional network produces similar results than a network using pooling. It is similar to average pooling, where the average response is passed to the next layer.

12.5.2 What Does a CNN Learn

A CNN consists of several (sometimes very many) building blocks as described above. It can be used, for instance, to detect objects in a region of interest (ROI). Different to a fully-connected MLP, the connection of nodes in the receptive field substantially reduces the number of parameters to be trained. Consider a ROI of 100×100 pixels to search for a certain cell event such as mitosis in microscopy images. Given a hidden layer with as many nodes and a receptive field of 5×5 pixels and a depth of 40 features (which are common values for the size of the receptive field and depth of the CNN), just 1000 parameters are needed to be trained for the first building block of the CNN, while the first layer of an MLP of the same size would have 10^8 parameters!

So what is lost? Layers of CNNs and MLPs can be seen as layers of abstraction. Abstraction by a layer increases with distance of this layer from the input layer. The first layer of a CNN trains convolution kernels of the image. After training, it may reflect local features in the image that are important for the decision task. If, e.g., vertical lines are characteristic for a classification task, the network would ideally produce matched filters for this. In fact, for most CNNs, several layers are needed for this since each layer represents a convolution with a single, small (e.g., 3×3) kernel. However, a sequence of several convolution layers, possibly interleaved with several pooling layers will learn complex local features.

An example on feature learning and classification by a dCNN is depicted in Figs. 12.28 and 12.29. Figure 12.28 shows the structure of the VGG16 net (also known as OxfordNet) of Simonyan and Zisserman (2014) that has won the ImageNet Large Scale Visual Recognition Competition 2014 for recognizing 1000 image categories. The depth of building blocks increases with decreasing size of the feature maps. Figure 12.29 shows some output results immediately before pooling in three building blocks. The first building block produces local responses at edges. This is expected because the small receptive field (3×3) can only perceive and learn characteristics that stem from dependencies in this small neighborhood. Each

Fig. 12.28 Schematic view of the structure of the OxfordNet of Simonyan and Zisserman (2014). Each building block consists of several convolutional layers computing 3×3 kernels and a pooling step (not shown) that reduces the resolution of the feature maps by a factor of two (stride 2). The depth of the building layers increases with the decreasing resolution of the feature maps

Fig. 12.29 Application of the dCNN depicted in Fig. 12.28 to an image (image cropped from picture "Ouzo the Persian Cat" by user Andrew Bone published under CC BY 2.0). It shows feature maps of the last convolutional layer before pooling. It is clearly visible that the local neighborhood increases from building block A to C. While the features of block A mainly react to different edge characteristics the features of block C characterize more global dependencies. The last line shows the average of responses from all feature maps of a building block

pooling step increases the size of the neighborhood in the original input image from which features are computed which can clearly be seen in results from later building blocks.

The feature-extraction part of a CNN cannot represent global dependencies between image components. For detecting relevant features in images, however, this is a good thing since small image detail is usually just locally correlated. It is the reason for analysis strategies such as multi-resolution Gaussian pyramids or for the modeling of low level information by MRFs. Hence, a CNN nicely reflects basic assumptions about how information is encoded in images. An MLP, that should represent similar abstractions of the input, would need extra training learn just this principle of local dependencies.

On a higher level, this locality principle not necessarily applies. Thus, the convolution layers of CNN extend only to a certain level, where the output (trained features) is fed to the fully connected MLP. The MLP part of the network is able to learn and represent arbitrary relationships between local feature responses at the convolution output of the network. A deep CNN, therefore, adds a preprocessing

step to a MLP that replaces user-specified decisions about relevant image features by a learning process.

A convolutional neural network can also be used as auto-encoder. Because of the local structure of the network, it is capable of learning local features from the input data and has been used as a feature detector.

12.5.3 Applications in Medical Image Analysis

Two major problems exist for application of deep CNNs in medical imaging. First, many existing, trained networks are 2D whereas applications in medical imaging are often 3D. Second, the required large amount of training data is not always present. A good overview of current applications gives a special issue from IEEE Transactions on Medical Imaging on this topic (Greenspan et al. 2016).

Interesting applications for dCNNs are detection problems where numerous small structures shall be detected such as mitosis detection in cells, lung nodules, or small aneurysms (Cireşan et al. 2013; van Ginneken et al. 2015; Nemoto et al. 2016). The dCNN may be applied to a small ROI with fewer nodes in the first and all subsequent layers. Furthermore, even few data sets provide a relatively large number of training samples since many instances of the object exist in a single image. If the images are 2D, such as microscopies and digital X-ray, the 3D problem does not exist as well. This is also true, if 3D images are processed slice by slice.

For training from a 3D data set or if the ROI is rather large, the size of the training data may still be too small. In this case, additional training data may be created by *data augmentation*. Data augmentation attempts to use existing data to interpolate new unseen data by perturbing the original data. Perturbation could for instance cause small displacements, rotations or apply any other affine transformation to the data. It can be often justified, since invariance w.r.t. these transformations is a desired property of the analysis system. However, it may tilt the learned a priori knowledge towards these influences and may neglect other, equally or even more important sources of variation such as variation due to patient-specific differences of anatomy. Hence, it has been suggested to apply nonlinear deformations to the data for better representing patient-specific anatomical variation. The strategy has limitations as well, since the augmentation can only be as good as is the quality with which the interpolation from existing data represents the unseen data.

Another strategy to deal with the insufficient amount of training data is to apply concepts of *transfer learning* to dCNNs. The general idea of transfer learning is that many concepts apply to more than one situation. If a computer has learned, for instance, to differentiate animals from plants in video sequences, many of the differentiating concepts (e.g., animals move and plants do not) may also apply to the different task to separate humans from plants. Applied to dCNNs, this leads to re-using parts of a trained network that have been trained for a specific analysis task in order to solve a new similar task. Local features, for instance, that are necessary to segment a liver in CT may also be similarly important to segment the kidney in CT.

This may extend even further. Features, which are trained on a very large number of images to classify objects in pictures in one domain, could very well capture the essence in pictures of a completely different domain. The feature extraction part of the already mentioned OxfordNet of Simonyan and Zisserman (2014) that was trained on a very large number of images to differentiate 1000 categories of real-world objects, for instance, has been frequently been employed to serve as feature reduction steps in other domains where sufficient samples for training the CNN were not available. The rationale behind this is the assumption that features on a local level are not very specific to some classification task and features trained on a large database are still better than user-selected features such as the responses to a bank of Gábor filters.

An advantage of medical images that may reduce necessary training stems from image acquisition. Imaging protocols usually restrict the pose of the patient data w.r.t. the scanner coordinate system. Hence, the position of the patient in the scanner is not arbitrary and global spatial relationships between organs may often be represented by local relationships of these organs to specific scanner coordinates. In consequence, some global features can be learned locally.

Deep CNNs can be applied to segmentation tasks. A simple way to do so is to treat the segmentation as ROI-based classification task (e.g., to assign foreground and background labels). In this case, a ROI around a scene element (pixel or voxel) is defined, which is large enough to comprise all information necessary to decide on foreground or background classification. The ROI constitutes the input layer for the network, and it is trained on all possible ROI positions in a number of pre-segmented images. The segmentation decision is then carried out for each scene element separately. The depth of the CNN depends on the size of the ROI. If the labeling decision can be based on few neighboring scene elements, the resulting networks may not be particularly deep as, e.g., in Zhang et al. (2015) who use 13×13 patches in MRI slices and a network that comprised just three convolutional layers.

Another strategy to compute a segmentation by a deep CNN is to use a decoder–encoder structure similar to auto-encoders (see Fig. 12.30). The decoder consists of

Fig. 12.30 If used for segmentation, a dCNN first reduces the image data for determining relevant features for the segmentation. Then, the result is expanded to map segmentation labels to each pixel or voxel

a number of convolution and pooling layers and the encoder consists of deconvolution and unpooling layers. Instead of using the input map as desired output as in an auto-encoder, the label map of segmented training images is learned (see, e.g., Brosch et al. 2016). The size of the input required for segmentation may comprise the complete scene (which would potentially be able to learn global relationship between segments), but may also be restricted to a smaller ROI. Selecting this size will be a compromise between increased training needed for a large ROI and the capability of expressing large scale relations between organs depicted by the image.

12.6 Classification by Association

Adaptive linear discriminant functions bear similarities to the minimum distance classifier in that their classification criterion is based on the correspondence to a model feature vector given by weights $w_{k,0}$, …, $w_{k,N}$ for a class k. Discriminant functions have to be defined for each class

$$D_k(\mathbf{f}) = w_{k,0} + w_{k,1}f_1 + w_{k,2}f_2 + \cdots + w_{k,N}f_N. \tag{12.35}$$

Classification of an unknown sample with feature vector $\mathbf{f}$ happens by inserting $\mathbf{f}$ in each of the $\mathbf{K}$ discriminant functions and selecting the class k for which $D_k(\mathbf{f})$ is maximal (see Fig. 12.31).

Training of a set of discriminant functions is similar to the training of decision boundaries. Discriminant functions are evaluated for classified training samples. If the incorrect class is assigned to a training sample, its corresponding discriminant function is changed to deliver smaller values and the discriminant function of the true class is changed to deliver higher values. A sketch of the algorithm is depicted in Fig. 12.32.

Fig. 12.31 Adaptive discriminant functions can be thought of as vectors in a feature space, where an additional feature f_0 has been added to represent the bias term. Classification is then by association to the most similar discriminant function

```
initialize weights w0,...,wM with small random values
repeat
    select sample f from training set
    compute all Dk(f)
    if the sample of class Ci is erroneously classified as Cj then
            for all features m=1,...,M do
                    wi,m:=wi,m+c·fi,m
                    wi,0:=wi,0+c·k
                    wj,m:=wj,m-c·fi,m
                    wj,0:=wj,0-c·k                    ; with c,k > 0
            end_for
    end_if
until classification is perfect          or average error < emin
                                         or steps > max_steps
```

Fig. 12.32 Sketch of the algorithm to compute weights from training features for adaptive discriminant functions

The similarity to the minimum distance classifier is due to the fact that computing the discriminant function is similar to measuring a non-Euclidean distance between features $\mathbf{f}$ and the weight vector $\mathbf{w}_k = (w_{k,1}, w_{k,2}, \ldots, w_{k,N})$, since

$$D_k(\mathbf{f}) = (\mathbf{w}_k)^{\mathrm{T}}\mathbf{f} + w_{k,0} = \|\mathbf{w}_k\| \cdot \|\mathbf{f}\| \cdot \cos(\angle \mathbf{w}_k, \mathbf{f}) + w_{k,0}, \tag{12.36}$$

with $\cos(\angle \mathbf{w}_k, \mathbf{f})$ being the distance measure. The bias can be integrated into the feature vector and the weight vector by setting $\mathbf{f}' = (1, f_1, f_2, \ldots, f_N)$ and $\mathbf{w}_k' = (w_{k,0}, w_{k,1}, w_{k,2}, \ldots, w_{k,N})$.

The discriminant functions can be turned into a distance classifier if feature vectors and initial weights are normalized by letting $\mathbf{f}_{\mathrm{norm}} = \mathbf{f}'/\|\mathbf{f}'\|$ and $\mathbf{w}_{k,\mathrm{norm}} = \mathbf{w}_k'/\|\mathbf{w}_k'\|$. In this case, weights need to be normalized after each update. The trained classifier then decides based on the angle between $\mathbf{w}_{k,\mathrm{norm}}$ and $\mathbf{f}_{\mathrm{norm}}$.

The result is similar to a decision on distance between $\mathbf{f}$ and a class center of the minimum distance classifier presented earlier, except that the discriminant vector $\mathbf{w}_{k,\mathrm{norm}}$ is trained based on a sample distribution. It also bears similarities to a support vector machine using radial base functions as explained in the previous section.

A discriminant function may also be used in a different way. If some order with respect to class membership can be assumed to be reflected in feature values of its samples (e.g., classes that represent stages of some process), a single function may suffice to discriminate between classes.

The approach is simple. Given V samples with N features, where the true class of a sample $\mathbf{f}_i = (f_{i1}, f_{i2}, \ldots, f_{iN})$ is represented by some number d_i, a function is searched that minimizes the distance $(D(\mathbf{f}_i) - d_i)^2$ for all samples:

$$\mathbf{w}_{\text{opt}} = \arg\min_{\mathbf{w}} E(\mathbf{w}, \mathbf{f}) = \arg\min_{\mathbf{w}} \sum_{i=1}^{V} (D(\mathbf{f}_i, \mathbf{w}) - d_i)^2 \tag{12.37}$$

with

$$D(\mathbf{f}_i, \mathbf{w}) = \mathbf{w}^{\mathrm{T}}\mathbf{f}_i + w_0. \tag{12.38}$$

Since the function is quadratic and non-negative, parameters can be computed directly without requiring an iterative method and the result is guaranteed to be the global minimum. The desired weights w_i are those for which the partial derivative of the error function E is 0. The classifier is called *minimum square error discriminant function*.

It may seem that, if a single linear discriminant function solves the classification problem, it should be the eigenvector with maximal eigenvalue of a PCA. This is not necessarily the case because the PCA does not maximize class separability. However, if PCA indicates that reduction of a high-dimensional feature space to one or two dimensions covers most of the variance in the samples, it makes sense to investigate whether a minimum square error discriminant function can be applied.

12.7 Clustering Techniques

Clustering is sometimes called *class detection* since it analyzes sample distribution in feature space. Cluster members may be inspected after clustering, and this may help to detect classes. Many different clustering techniques exist (Jain and Dubes 1988; Xu and Wunsch 2005), of which some popular techniques for medical data will be described in the following sections.

Clustering of data is carried out in a top-down or a bottom-up fashion on unclassified samples. Examples of top-down clustering schemes have been presented in Sect. 7.2. The *k*-means clustering described in Sect. 7.2.1 can also be applied to high-dimensional sparse feature spaces, while mean-shift clustering (Sect. 7.2.2) gets very slow with increased dimension of feature space.

In this chapter, we will describe two further clustering techniques that are particularly suitable if the number of samples is comparatively low.

If clustering is applied for class detection, it is sometimes called *unsupervised classification* and the process is called *unsupervised training*, as it does not require assignment or checking the class membership of samples.

12.7.1 Agglomerative Clustering

Bottom-up clustering is also called *agglomerative clustering*. Initially, every sample is its own cluster. Pairs of closest clusters are then merged until the expected

number of clusters is reached. If this is unknown, agglomeration continues until a single cluster is generated. Merging is documented by a binary tree with leaves being the initial clusters (see Fig. 12.33). A parent node is created from two nodes in the tree if their corresponding clusters are merged.

Storing the history of agglomeration in the tree enables later analysis of generated clusters. Some classes may fall into more than one cluster. Going up and down in the agglomeration tree and selecting and inspecting members of clusters help to understand the true distribution of class members in clusters.

Agglomerative clustering requires that the distance $d(C_1, C_2)$ between two clusters C_1 and C_2 can be computed. Different measures have been used (see Fig. 12.34):

- *Single linkage* defines the distance between two classes by the distance between its closest members.
- *Complete linkage* defines the distance by that of its furthest members.
- *Average linkage* defines the distance by the distance between centroids of two clusters.

Agglomerative clustering not always delivers good results as $d(M(C_1, C_2), C_3) \neq d(C_1, M(C_2, C_3))$, where $M()$ indicates the merging of two clusters. Hence, final clustering depends on the early decisions.

Fig. 12.33 Agglomerative clustering initially treats each sample as its own cluster. It then merges always the two closest clusters until all clusters are merged. The merging process is represented in a tree structure. The user then selects the desired number of clusters

Fig. 12.34 Different kinds of distance measures between clusters: **a** single linkage, **b** complete linkage, **c** average linkage

12.7.2 Fuzzy c-Means Clustering

Since clustering for class detection is provisional, it may be appropriate to model membership in a fuzzy way. Again, several different algorithms exist (see, e.g., Baraldi and Blonda (1999) for a survey) of which we present the base method.

A sample may belong to different clusters with different degrees of membership. The method is known as *fuzzy c-means clustering*. The degree of membership is given by the partition matrix **U**, where each entry u_{ij} determines the degree of membership of a sample i of N samples with feature vector $\mathbf{f}_i$ to cluster j of C clusters (see Fig. 12.35). Membership degree is normalized to a range of [0, 1] so that the following two conditions must hold for the values u_{ij} in the partition matrix

- Total membership for each sample to clusters 1 to C: $\sum_{j=1}^{C} u_{ij} = 1$.
- No empty clusters: $0 < \frac{1}{N}\sum_{i=1}^{N} u_{ij} < N$.

The objective is to create a good partitioning with compact clusters based on a compactness definition (see Fig. 12.36). Computing compactness requires a

Fig. 12.35 Example for a partition matrix. A membership degree is assigned to each sample for each class

Fig. 12.36 The goal of fuzzy clustering is to find a partitioning with compact clusters

distance metric $d(\mathbf{f}_i, \mathbf{f}_j)$ to exist (e.g., the Euclidean distance). Compact clusters are created if the average distance between fuzzy members of a cluster is minimal:

$$\mathbf{U}_{\text{opt}} = \arg\min_{\mathbf{U}} \sum_{j=1}^{C} \frac{\sum_{k,l=1}^{N} u_{kj}^2 u_{lj}^2 d(\mathbf{f}_k, \mathbf{f}_l)}{2\sum_{i=1}^{N} u_{ij}^2}. \tag{12.39}$$

Computing a fuzzy clustering is particularly simple if compactness is defined by the distance of a sample to its cluster center. A cluster center $\mathbf{c}_j$ for a given partition matrix $\mathbf{U}$ is the weighted sum of sample locations in features space. The weighting $w(u_{ij})$ is a function of the membership degree u_{ij} of a sample i to a cluster j:

$$\mathbf{c}_j = \frac{1}{\sum_{i=1}^{N} f\left(u_{ij}\right)} \sum_{i=1}^{N} w\left(u_{ij}\right)\mathbf{f}_i. \tag{12.40}$$

The iterative process starts with an initial partition matrix that could be generated, e.g., from smoothing a hard partitioning by a Gaussian kernel. The objective is to find $\mathbf{U}_{\text{opt}}$, such that the sum of weighted distances $u_{ij}\|\mathbf{f}_i - \mathbf{c}_j\|$ between features $\mathbf{f}_i$ and cluster centers $\mathbf{c}_j$ is minimal. The function to be minimized is now

$$\mathbf{U}_{\text{opt}} = \arg\min_{\mathbf{U}} \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^m \left\|\mathbf{f}_i - \mathbf{c}_j\right\|^2, \quad 2 \le m < \infty. \tag{12.41}$$

The parameter m controls the fuzziness of the distribution. With m approaching infinity, the algorithm turns into hard clustering.

The matrix $\mathbf{U}_{\text{opt}}$ can be found by gradient descent in the space of the u_{ij}. The step for updating u_{ij} at iteration n is

$$u_{ij}^{(n+1)} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\left\|\mathbf{f}_i - \mathbf{c}_i^{(n)}\right\|}{\left\|\mathbf{f}_i - \mathbf{c}_k^{(n)}\right\|}\right)^{\frac{2}{m-1}}}, \tag{12.42}$$

where $\mathbf{c}_j^{(n)}$ is a cluster center at iteration (n) defined by weighted clustering

$$\mathbf{c}_j^{(n)} = \frac{1}{\sum_{i=1}^{N} \left(u_{ij}^{(n)}\right)^m} \sum_{i=1}^{N} \left(u_{ij}^{(n)}\right)^m \mathbf{f}_i. \tag{12.43}$$

Optimization terminates if the change between partition matrices of subsequent iterations falls below some threshold. A matrix norm such as the maximum norm on matrices may be used to compute the stopping criterion:

$$\left\|\mathbf{U}^{(n)} - \mathbf{U}^{(n+1)}\right\| < \varepsilon \quad \text{with} \quad \left\|\mathbf{U}^{(n)} - \mathbf{U}^{(n+1)}\right\| = \max_{ij} \left\|u_{ij}^{(n)} - u_{ij}^{(n+1)}\right\|. \tag{12.44}$$

Fuzzy c-mean clustering is often intended to lead to a (fuzzy) classification of the samples. Hence, the number of clusters usually corresponds to the number of classes to be differentiated. Using gradient descent for optimization requires the initial partition matrix to be close to the final result.

12.8 Bagging and Boosting

Classification based on training from examples can be difficult if the decision boundary between classes is complex. The number of training samples may be insufficient to generate an appropriate classifier. *Bagging* and *boosting* are two techniques to arrive at a good classifier nonetheless by applying a kind of divide-and-conquer strategy. Instead of training a strong classifier that correctly classifies all samples, a set of weak classifiers is searched where each sample in the training set is classified by the majority of the classifiers. The advantage of combining weak classifiers lies in the fact that the absolute accuracy that is requested from the strong classifier is replaced by a relative accuracy of a combination of weak classifiers. A comparison of the performance of bagging and boosting algorithms can be found in Bauer and Kohavi (1999).

The main difference between bagging and boosting is that bagging generates a voting scheme, where each classifier votes for a class and the class with the majority of votes wins, while boosting generates a sequence of classifiers, where each classifier learns from the previous classifier and votes are weighted according to classifier quality. Both methods have in common that a given classification algorithm (e.g., kNN classification) is trained on a number of subsets of the training data producing the different classifiers.

Bagging (Bootstrap aggregating), presented by Breiman (1996), repeatedly selects subsets S_i from the training set and trains a classifier C_i based on this subset. The final classifier tries then all classifiers C_i on the unknown sample and assigning the class to it that receives the most votes.

Boosting has been introduced by Schapire (1990) and been extended to AdaBoost (Adaptive Boosting) in Freund and Schapire (1996). AdaBoost, used in the context of medical image analysis, e.g., by Pujol et al. (2003), Ochs et al. (2007), and Takemura et al. (2010), trains each classifier on the complete training data. A sketch of the algorithm is depicted in Fig. 12.37.

Different classifiers are generated by weighting the samples based on the training error after a classifier has been trained. The weighted sample set is the submitted to the next classifier and weighted again. Samples receive higher weights for the next classifier if they were not classified correctly by the current classifier. Hence, the next classifier will be trained to remove errors not corrected sufficiently by all previous classifiers. The final classifier combines then, as in the case of bagging, the weak classifiers and decides by majority vote. However, different to bagging, classifier votes are weighted by their error rate.

While bagging may be carried out in parallel, AdaBoost is inherently sequential since its classifiers learn from their predecessors. Other boosting algorithms exist that define the search for the final classifier as an optimization task that can be

Input: training set S of labeled pairs (s_i,l_i), i=1,N, classification scheme CS, number of trials T

Output: combined classifier C^*

$S'=S$
initialize weights $w(s_i)$=1.0, i=1,N
for i=1,T **do begin** ; create weak classifiers
ε_i=1.0
k=0
while ε_i>0.5 and k<25 **do begin** ; classifiers must have a minimum quality
; try up to 25 times to achieve this
C_i = computeClassifier(S',CS)
$$\varepsilon_i = \frac{1}{N}\sum_{s_j \in S':C_i(s_j)\neq l_j} w(s_j)$$
k=k+1
endwhile
$\beta_i=\varepsilon_i/(1.0-\varepsilon_i)$
for j=1,N **do begin** ; error-based sample weighting
if $C_i(x_j)=l_j$ **then** $w(x_j)=w(x_j)/(2\varepsilon_i)$
else $w(x_j)=w(x_j)/(2(1.0-\varepsilon_i))$
endfor
endfor
$$C^*(x) = \arg\max_{l\in L} \sum_{i:C_i(x)=l} \log\frac{1}{\beta_i}$$; final classifier

Fig. 12.37 Sketch of the AdaBoost algorithm (adapted from Freund and Schapire 1996 and Bauer and Kohavi 1999)

solved in parallel. An example is LP Boosting (Demirez et al. 2002) that defines boosting for linear decision boundaries as a linear programming task and that has been used, e.g., to analyzes cortical thickness from MRI for autism diagnosis (Singh and Chung 2008).

12.9 Decision Trees

Decision trees represent a sequence of decisions based on the input to arrive at the final decision. The sample to be classified is fed to the root of the tree. Each node of the tree requests a decision based on the features of the sample. If, for example, a binary decision tree is used, the decision would request either *yes* or *no* as answer. The sample is the passed to a leaf of the tree according to the answers. The leaf contains the final decision, i.e., the classification.

12.9.1 Creating a Decision Tree

Decision trees can be created by an expert who decides which questions on which features are needed to reach a decision. This option may be appropriate for decisions in a well-known environment (e.g., for deciding about part-use in manufacturing), but often the reasons for a decision are too complex and dependencies between them are unknown. In this case, the tree is trained from labeled samples.

Training a decision tree is done top-down. The training data is repeatedly separated into disjoint subsets. Each separation splits a node of the decision tree generating child nodes. At the end of the process each of the subsets contains only samples with the same label (see Fig. 12.38).

In practice, a certain error rate will be accepted, in order to avoid over-fitting to the training data. This is called the *pruning step*. If pruning is carried out beforehand (so-called pre-pruning), this is done by restricting the number of layers of the decision tree. It is a simple pruning strategy, but has the disadvantage that the pruning is based on a prior assumption about the depth of the tree. Pre-pruning cannot take into account the full information about the optimal separation of the sample data set. If post-pruning is used, the tree is fully developed and then pruned. Pruning starts at the leaves by collapsing those parts of the tree which produce the smallest increase in the error rate. Post-pruning is more costly than pre-pruning, but it takes the local characteristics of sample distributions in feature space into account.

Decision trees can be used for classification (where the output is one of a number of labels) or for regression (where the output is the real-valued point on the regression function).

A commonly used splitting criterion for generating a tree from samples is the entropy of the value distributions in the subsets. An ideal subset would contain only one label. Hence, its entropy would be minimal. A set of samples would be split

Fig. 12.38 A decision tree is grown by repeatedly subdividing the feature space. In this example, a simple linear decision boundary is used for always finding a subdivision with as few erroneously classified examples as possible. This is clearly insufficient for the complete data set (*first row*), but further subdivisions improve the classification quality. Classification of an unknown sample is then done by passing it to the decision tree down to a leaf and assigning the label of this leaf. The example also shows the danger of overfitting the tree to the training data

into those subsets that have the smallest entropy of all possible separations in disjoint subsets. CART (Classification and Regression Trees) algorithm and its successors (ID3, C4.5, and C5) are methods to generate binary decision trees under these assumptions.

12.9.2 Random Forest

A random forest consists of a set of trained decision trees. An unknown sample is fed to each of the trees and if used for classification, labels are assigned by majority vote. Each tree is trained on a different subset of all training samples. Hence, random forests apply the bagging strategy described in the previous section to decision trees.

The algorithm presented by Breiman (2001) addresses the fact that these trees may become correlated if the data contains several strong predictors among the features of the samples. In this case, most of the selected sample subsets would allow predicting sample labels based on these strong predictors. The entropy-based splitting procedure would favor splits based on these features. It would counteract one of the biggest advantages of bagging strategies which is the ability to accept weak classifiers that perform well for few samples that are substantially different to the set of samples. They may well be just those samples for which the aforementioned strong predictors among the features do not produce a good result.

In order to avoid this, Breiman (2001) added a second randomization step at the splitting stage. Each time when a node is split, a randomly selected subset of features is used for computing the optimal split. Hence, some splits may be based on the strong predictors, but others will not. It results in a more diverse forest of decision trees which is less biased by the prediction capability of single features.

The number of trees in a random forest is a free parameter that needs to be determined depending on the application problem. It may also be computed by cross-validation. In this case, the number of trees is iteratively increased until a certain quality of classification is reached. Each decision tree is used to predict the labels of all samples unseen by this tree (i.e., not used for training of this tree). The average prediction error for all trees serves as a measure of quality for the current tree ensemble.

12.10 Multiple Instance Learning

Although the scenario of multiple instance learning (MIL) is attractive and it is being investigated by many groups, it has currently few applications in analysis tasks in medical imaging such as computer-aided diagnosis, see Fung et al. (2006), Bi and Liang (2007), Xu et al. (2014). MIL is a weakly supervised learning technique that will be briefly described in the following.

The first application domain for MIL was drug design (Dietterich et al. 1997; Maron and Lonzano-Péres 1997). It was investigated whether certain molecules bind or not. Molecules may have different spatial configurations. The binding ability depends not only on the molecule, but also on the configuration. Hence, when observing relevant features of a molecule by some imaging technique, they may vary to a great extent for the same kind of molecule.

Each of the observations is treated as an instance of the same type of molecule which belongs to one of two classes ("binding" or "not binding"). Assigning this class membership to the molecule type in order to create a training set would result in distribution functions that overlap to a large extent as it is not part of the model that only some of the instances of a binding molecule may actually bind. The problem is represented better if all the instances of a molecule are treated as a bag of samples that is either labeled "binding" or "not binding". The former is known to contain at least one binding instance while the latter does not contain any binding instance. The task is to extract a decision boundary from a set of training bags that separates instances of the one class from those of the other class.

MIL in medical image analysis can be used, for instance, to reduce operator work in supervised training. The work of Xu et al. (2014) is an example for this. Their goal was to create a classifier for detecting tumorous cells in microscopes. One part of the work was to provide operator-labeled training data. Instead of letting the operator delineate and indicate each tumor cell in the image, they segmented the images into patches (the instances) and requested from the operator to indicate images that contained at least one tumor patch. Hence, each of the training images is a bag of samples where some of the bags contain tumor instance and some not.

The problem can be put in a more abstract form. Two types of bags B exist in MIL: Positive bags $B = 1$ contain at least one sample s with class(s) = 1 and negative bags $B = -1$ contain only samples s with class(s) = −1. Having positive and negative bags is actually the key to solve the MIL problem. An optimal decision boundary will separate all samples from negative bags from positive samples such that at least one member of all positive bags is separated from the negative bags.

Since the positive samples in positive bags are unknown, different strategies can be and are used to arrive at a solution under various a priori assumptions (see Fig. 12.39). A decision boundary enclosing as many positive samples and as few

Fig. 12.39 In multiple instance learning, positive bags (*circles*) contain at least one positive instance, while negative bags (*squares*) contain only negative instances. Finding a decision boundary requires additional assumptions. Examples are to find the rectangle (*dashed lines*) that encloses the most instance of positive bags without containing instances from negative bags or the smallest region that contains at least one instance of each positive bag (*gray region*)

negative samples as possible is searched when positive bags are assumed to be largely made up of positive samples. A non-overfitting decision boundary enclosing no negative samples, but as many positive samples as possible is searched if negative samples are reliable. A decision boundary enclosing dense clusters of positive samples is searched when it is assumed that only few feature value combinations constitute the unknown positive range (which is the case in searching for binding configurations).

12.11 Bag of Visual Words

When features are used to detect objects they need to be combined to some kind of a super-structure, since the semantic of a single feature is usually insufficient for representing the object characteristics. A simple way to arrive at a higher level semantic is to just collect features (points, edges, textures) from a region that shall be recognized. Feature collection is done by picking regularly or randomly distributed patches from the region and computing feature attributes in these region (Nowak et al. 2006). The bag of feature approach is attractive for search in image databases and has been successfully applied to solve content-based image retrieval tasks for medical images, e.g., by Caicedo et al. (2009), Wang et al. (2011). It has also been used to assess the quality of images (Farzi et al. 2016).

Each patch as assumed to be a "visual word" making up the visual sentence describing the image. It is a *bag of visual words* that represents the (main) meaning of the region from which the features are taken. The strategy has been adapted from the Bag of Words (BoW) method that has been successfully applied to analyze text categories, e.g., to detect spam e-mails. The basic idea is that a text is characterized by the frequency of word occurrences in it. Even though words receive parts of their meaning as parts of sentences or paragraphs it has been shown that comparison of word occurrences suffices to assign the messages to broad categories such as spam and no-spam messages. Hence, a dictionary of key words occurring in e-mails is generated that contains buzzwords used in spam as well as frequently occurring words in regular emails. A decision is then made by comparing the histogram of an unknown e-mail with those of typical "spam" and "no spam" e-mails.

A similar strategy is followed in the bag-of-visual-word classification. Since the content of the bag is not spatially ordered, the representation by visual words is invariant with respect to translation and if the feature values are invariant to rotation or scale, invariant with respect to these two transformations as well. For feature-based analysis, the histograms are compared with histograms of class prototypes.

Different to a BoW strategy, the set of words is not known in advance, from which the dictionary of "visual words" can be created. Hence, dictionary creation is a necessary pre-processing step. Features (e.g., texture features) are defined that are

assumed to carry the semantics of the visual word. Patches are then taken from sample images and their feature vectors are computed. Since meaningful feature vectors should occur more often than accidentally or unlikely values, visual words should form clusters in feature space. Clustering is applied to detect the words in feature space (see Fig. 12.40). The number of clusters is chosen in a way that the average error of expressing the training data by their closest cluster center is below some threshold.

Selecting the number of visual words can be critical for the success. Too few words in the dictionary may represent the unknown words not correctly. This is especially so because distance measures in feature space may not coincide well with semantic distances between different visual words. Too many visual words, on the other hand, may make distinctions between feature vectors that are meaningless in a semantic space of visual words. It also makes the classification inefficient. Hence, it makes sense to carefully select the feature space in which visual words are represented, potentially apply feature reduction techniques to reduce redundancies and normalize the feature space before creating the dictionary.

For image classification of an unknown image, patches from the image are assigned to the one word, whose cluster center is closest to the feature vector of this patch. The distance to the cluster centers may be used to determine whether the feature vector of an unknown word does not exist in the dictionary. It may also be used to weigh the reliability of the assignment to a word in the dictionary.

Fig. 12.40 Visual words are defined from clusters of feature vectors from patches taken from training images. It is well possible that feature vectors are typical for several classes, hence patches from images that belong to different classes may be mapped to similar locations in feature space. This is not critical as long as the frequency of occurrence is characteristic for an image class

Given a good dictionary, classification is done by carrying out the following steps:

- Arbitrarily select patches (=visual words) from the image.
- Compute descriptor (feature) from the batch of visual words.
- Classify according to characteristics of the patch collection.

The last step consists of a histogram comparison of the word occurrences of the unknown object with the histograms of the different object classes (see Fig. 12.41).

Fig. 12.41 Given a dictionary of visual words, unknown samples are classified by comparing their histogram of visual words with that of prototypical class members of each class

12.12 Dealing with Imbalanced Data

If classes are not equally represented in the training data set, the data set is called imbalanced. Imbalance may be the result of non-representative sampling, for instance if training data has been gathered in a hospital with a bias towards sick people, but the method shall be applied in a population study where the majority of subjects are healthy. Imbalance may also be inherent to the classification task. If, for instance, n (normal) and p (pathological) are the two classes of a classification problem, the n class would occur much more often than the p class.

Training a classifier based on average classification error would bias the results towards the majority class. If in our example 95% of all subjects were classified as normal, then classification based on the a priori probability (all samples are classified normal irrespective of their features) would produce a classification error of just 5% which in many applications would be an acceptable error margin.

In terms of the quality measure (overall prediction accuracy) this is correct unless the imbalance is due to improper sampling. It assumes, however, that all errors are equally important, which may not be true. In the example above, the consequences of overlooking a pathological case may be substantially more serious than erroneously labeling a normal case as pathological.In Bayesian decision making, this can be considered by including a *cost of error* L into the equation. A posteriori probabilities are weighted with L and the decision between two classes c_1 and c_2 is now governed by

$$L(c_1|c_2)P(c_1|\mathbf{f}) > L(c_2|c_1)P(c_2|\mathbf{f}) \tag{12.45}$$

where $L(c_i|c_j)$ the cost of error is for erroneously assigning a member of class c_i the class label c_j. Hence, even if the a posteriori probability of a sample for class c_1 is higher than that for c_2, it may receive the label c_2 nonetheless if the cost of error to assign a member of class c_2 erroneously to class c_1 is higher than the cost for the opposite case.

For non-parametric decision making this may not be as easy and needs to be simulated by manipulating the training data. Either the majority class is under-sampled or artificial samples are generated for the minority class. The latter is preferred since the former reduces the size of the training data set.

The simplest way to create new samples is to randomly select samples from the minority class and create duplicates until the desired size of the training set is reached. Since this may introduce bias towards the original samples, alternatively novel samples can be generated by perturbing the features of existing samples. A popular algorithm for this purpose is the *synthetic minority over-sampling technique* (SMOTE) of Chawla et al. (2002). For generating new samples, a kNN-tree is created for each sample in feature space that links the k nearest neighbors to this sample. New samples are generated on the line connecting the sample with one of its neighbors at a randomly chosen distance between sample and neighbor (see Fig. 12.42) assuming a uniform distribution. The value of k depends

Fig. 12.42 SMOTE classifier first constructs a network of kNN trees and then generates new samples along the edges that connect the root of the tree with the nearest neighbors (we chose $k = 3$ in this example). If an edge is member of more than one tree, samples can be generated for each tree of which this edge is part (if chosen by the random selection)

on the characteristics of the data set and the required number of new samples to be generated. It should be small enough to restrict sample generation to a local neighborhood around the original sample. A value of $k = 5$ was suggested in the original paper. The extent of oversampling is a number $l \leq k$ which means that new samples are generated on a subset of lines randomly chosen from the k lines linking a sample with its nearest neighbors.

SMOTE can be seen as an approximation of creating additional samples with distribution characteristic that has been created using a kernel density estimator. Actually creating this density function would be impractical, however, since the feature space may be high dimensional and data is just too sparsely distributed. Instead a manifold over the feature space is created on which projected samples are uniformly distributed. New samples are then created that do not change this characteristic. To understand this, let us assume that the original distribution is already uniformly distributed. If new samples are generated randomly between the existing samples with a uniform distribution, the distribution function remains uniform.

In order to decide where new samples are generated between existing samples, each sample location is connected by k lines with its k closest neighbors and samples are created on a randomly chosen subset of these lines. If lines are selected randomly and positions on the lines are generated randomly as well, both with uniform distribution, bias from choosing the neighborhood system and from the feature values of the original samples are avoided. Exactly the same happens with SMOTE, except that the neighborhood is defined by embedding samples in a higher

Fig. 12.43 By embedding samples in a higher dimensional space in which they are uniformly distributed, the creation of new samples uniformly distributed between neighboring samples approximate selection of samples according to the (unknown) true distribution as it could be approximated by kernel density estimation

dimensional space in which the arbitrarily distributed samples are uniformly distributed (see Fig. 12.43).

The developer of a classification method should have a clear understanding of what happens when the training data is manipulated in this way. While this is relatively clear when introducing a cost of error, up- and down-sampling techniques are sometimes applied based on a vague impression that the imbalanced data is somehow "not correct" or would somehow hinder the training of the classifier. The former is not true as the data is truly imbalanced (as in the example above with a normal and a pathological class). Hence, balancing the classes remedies a different problem, namely an inappropriate training criterion such as the average error over all samples. Deciding the extent of oversampling of the minority class therefore indirectly introduces a cost of error such as in Eq. (12.45). The developer should take this into account, especially if classifiers are used that approximate the a posteriori probability (such as kNN) or that are sensitive w.r.t. changes in the a posteriori probability of conditional class membership.

Instead of increasing the sample size, changing the quality measure for training may work equally well. Measures, used for validation (see Chap. 13), that differentiate between false positive and false negative results such as the F-score could be a useful alternative to oversampling in feature space.

Training of the classifier may indeed be impeded, nonetheless, since the minority class may contain too few samples to define a good estimate of a decision boundary between classes in feature space. In this case, oversampling would serve as a kind of "discrete kernel density estimator" since it can be seen as a convolution of the samples with an appropriate kernel function followed by sampling the result.

12.13 Concluding Remarks

Classification in sparse feature space mostly attempts to create a decision criterion without estimating likelihood functions. The strategy is successful when suitable features cluster well. Hence, errors in computing class membership will only result from samples close to the decision boundary.

Since reliability and speed of most classification methods increase with decreasing dimensionality of feature space, feature reduction should precede classification. This can be carried out with and without considering the distribution of classified samples in feature space.

A classification technique should be chosen based on knowledge about distribution characteristics in feature space. Simple techniques such as linear decision boundaries or linear support vector machines generalize well and are less dependent on initialization. Methods allowing for complex decision boundaries help to deal with data that is difficult to classify. They may, however, have the tendency to over-adapt to training data.

Convolutional neural networks have their own means to counteract over-adaption. The reduction of the input at each layer to local receptive field is a strong restriction which nonetheless seems not to impede the wide applicability of these networks. It allows producing much deeper trees than those in MLPs, and they appear to produce much better abstractions than MLPs.

There are a number of software packages that let the developer access most of the methods (and several more) that have been presented in this chapter. Two well-known libraries are R and WEKA. The software environment and the programming language R is used for statistical computing. It is used as interpreter language, i.e., the user may call functions of R in command line mode (similar to Matlab and IDL, mentioned in the introduction). The R project is an open source project; the software can be downloaded from https://www.r-project.org/. WEKA (Waikato Environment for Knowledge Analysis, Hall et al. 2009) is library of machine learning software written in Java that has been developed at the University of Waikato. It can be downloaded at http://www.cs.waikato.ac.nz/ml/weka/.

12.14 Exercises

- Why is it difficult to derive a likelihood function from a sparsely populated feature space? Why does it help when features are independent of each other?
- When is it a disadvantage to use PCA for feature reduction? Please explain how training data can be used to arrive at a better feature reduction. Does this have limitations? Which?
- How can independent component analysis be used for classification? What conditions must be met so that independent components can be found?

- Develop a situation where a kNN classifier would be preferred over a linear decision boundary. Discuss the attributes of this classification problem that lead to the preference.
- How can a kNN classifier "learn" from confirmed results? What is a potential disadvantage of this? And how can this be dealt with?
- What is the assumption under which a minimum distance classifier delivers the correct Bayesian decision (i.e., delivers the same result than that from evaluating the a posteriori probability)?
- Why is it useful to reject samples from classification by a kNN classifier? Which samples are rejected?
- How is classification by linear decision boundary related to classification based on the Bayesian theorem?
- How can the linear decision boundary approach be extended to nonlinear boundaries? What are the limitations of this approach?
- What kind of decision boundaries can be generated by a MLP with two layers (hence, without hidden layer) if the number of node per layer is not limited?
- Why is it necessary to use the sigmoid function as activation function in an MLP instead of thresholding? What effect does this have on classification?
- What is an auto-encoder and how could it be used in classification?
- What is the major difference between an MLP and a CNN?
- What are the purposes of the two sub-layers of a CNN layer?
- It is possible to build a CNN with very few layers (e.g., 4–6). Explain, why such a CNN would not be able to learn high level concepts. Why could it still be useful?
- What is the optimization goal when training a support vector machine? What are the support vectors?
- What is meant by the "kernel trick"? How does it extend the capabilities of a support vector machine?
- How can a support vector machine be made to simulate an association network?
- How can an SVM be extended to deal with overlapping training samples?
- Explain, how a decision tree can be trained from labeled samples.
- Explain, why a random forest "implements" a boosting strategy for decision trees.
- What could be a problem for a random forest if some of the features are strong predictors for the labels to be learned? How could this be avoided?
- Which strategy should be followed if the number of clusters is not known for agglomerative clustering? How can clusters then be identified?
- How is fuzzyness encoded in fuzzy clustering? What is the goal of finding an optimal fuzzy clustering?
- What is the advantage of using bagging instead of a single trained classifier? In what kind of distribution in feature space does bagging do particularly well?
- What is the main difference between bagging and boosting? Why should this be advantageous for using boosting?
- What is meant by unbalanced data in a classification task? What kind of problem can arise if training data is unbalanced? Explain, why.
- Explain the processes that are applied to balance data using SMOTE.

References

Abdi H, Williams LJ (2010) Principal component analysis. Wiley Interdiscip Rev Comput Stat 2(4):433–459

Baraldi A, Blonda P (1999) A survey of fuzzy clustering algorithms for pattern recognition. Part I + II. IEEE Trans Syst Man Cybern B Cybern 29(6):778–801

Bauer E, Kohavi R (1999) An empirical comparison of voting classification algorithms: bagging, boosting and variants. Mach Learn 36:105–142

Bi J, Liang J (2007) Multiple instance learning of pulmonary embolism detection with geodesic distance along vascular structure. In: IEEE conference on computer vision and pattern recognition CVPR'07, pp 1–8

Bishop CM (1995) Neural networks for pattern recognition. Oxford University Press, Oxford

Bishop CM (2006) Pattern recognition and machine learning, computational statistics and data analysis. Springer, Berlin, p 198

Blanco Y, Zazo S (2003) New Gaussianity measures based on order statistics: application to ICA. Neurocomputing 51:303–320

Breiman L (1996) Bagging predictors. Mach Learn 24:123–140

Breiman L (2001) Random forests. Machine Learning 45(1):5–32

Brosch T, Tang LY, Yoo Y, Li DK, Traboulsee A, Tam R (2016) Deep 3D convolutional encoder networks with shortcuts for multiscale feature integration applied to multiple sclerosis lesion segmentation. IEEE Trans Med Imag 35(5):1229–1239

Caicedo JC, Cruz A, Gonzalez FA (2009) Histopathology image classification using bag of features and kernel functions. Artif Intell Med LNCS 5651:126–135

Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002). SMOTE: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357

Cireşan DC, Giusti A, Gambardella LM, Schmidhuber J (2013) Mitosis detection in breast cancer histology images with deep neural networks. In: International conference on medical image computing and computer-assisted intervention MICCAI 2013, pp 411–418

Cortes C, Vapnik V (1995) Support vector networks. Mach Learn 20(3):273–297

Demiriz A, Bennett KP, Shawne-Taylor J (2002) Linear programming boosting via column generation. Mach Learn 46(1–3):225–254

Dietterich TG, Lathrop RH, Lozano-Péres T (1997) Solving the multiple-instance problem with axis-parallel rectangles. Artif Intell J 89(1–2):31–71

Duda RO, Hart PE, Stork DG (2001) Pattern classification, 2nd edn. Wiley, Hoboken

Farzi M, Pozo JM, McCloskey EV, Wilkinson JM, Frangi AF (2016). Automatic quality control for population imaging: a generic unsupervised approach. International conference on medical image computing and computer-assisted intervention MICCAI 2016, pp 291–299

Fisher RA (1936) The use of multiple measurements in taxonomic problems. Ann Eugenics 7:179–188

Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm. In: Machine learning: proceedings of the 13th international conference, pp 148–156

Fukushima K (1980) Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol Cybern 36(4):193–202

Fung G, Dundar M, Krishnapuram B, Bharat Rao R (2006) Multiple instance learning for computer aided diagnosis. In: Proceeding of the 19th annual conference advances in neural information processing, pp 425–432

Greenspan H, van Ginneken B, Summers RM (2016) Guest editorial deep learning in medical imaging: overview and future promise of an exciting new technique. IEEE Trans Med Imag 35(5):1153–1159

Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH (2009) The WEKA data mining software: an update. SIGKDD Explor 11(1):10–18

Hwang WJ, Wen KW (1998) Fast kNN classification algorithm based on partial distance search. Electron Lett 34(21):2062–2063

Hyvärinen A, Oja E (2000) Independent component analysis: algorithms and applications. Neural Netw 13(4–5):411–430

Hyvärinen A, Hurri J, Hoyer PO (2009) Independent component analysis. Natural Image Stat Comput Imag Vis 39(II):151–175

Jain AK, Dubes RC (1988) Algorithms for clustering data. Prentice-Hall, Upper Saddle River

Jiang Q, Zhang W (1993) An improved method for finding nearest neighbors. Pattern Recogn Lett 14:531–535

Le Cun Y, Jackel LD, Boser B, Denker JS, Graf HP, Guyon I, Henderson D, Howard AE, Hubbard W (1989) Handwritten digit recognition: applications of neural network chips and automatic learning. IEEE Commun Mag 27(11):41–46

Maron O, Lozano-Péres T (1997) A framework for multiple instance learning. In: Proceeding of conference on advances in neural information processing systems (NIPS), pp 570–576

Müller KR, Mika S, Ratsch G, Tsuda K, Schölkopf B (2001) An introduction to kernel-based learning algorithms. IEEE Trans Neural Netw 12(2):181–202

Nemoto M, Hayashi N, Hanaoka S, Nomura Y, Miki S, Yoshikawa T, Ohtomo K (2016) A primitive study of voxel feature generation by multiple stacked denoising autoencoders for detecting cerebral aneurysms on MRA. SPIE Med Imag 2016:97852S

Nowak E, Jurie F, Triggs B (2006) Sampling strategies for bag-of-features image classification. European Conference on Computer Vision ECCV 2006, LNCS 3954, pp 490–503

Ochs RA, Goldin JG, Abtin F, Kim HJ, Brown K, Batra P, Roback D, McNitt-Gray MF, Brown MS (2007) Automated classification of lung bronchovascular anatomy in CT using AdaBoost. Med Image Anal 11(3):315–324

Pujol O, Rosales M, Radeva P, Nofrerias-Fernández E (2003) Intravascular ultrasound images vessel characterization using AdaBoost. Funct Imag Model Heart LNCS 2674:242–251

Rosenblatt M (1956) Remarks on some non-parametric estimates of a density function. Ann Math Stat 27:832–837

Rosenblatt F (1958) The perceptron: a probabilistic model for information storage and organization in the brain. Psychol Rev 65(6):386–408

Sarle WS (1994) Neural networks and statistical models. In: Proceedings of the 19th annual SAS Users Group international conference, pp 1–13

Schapire RL (1990) The strength of weak learnability. Mach Learn 5(2):197–227

Schölkopf B, Smola AJ (2002) Learning with kernels—support vector machines: regularization, optimization and beyond. MIT Press, Massachusetts

Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556

Singh ML, Chung MK (2008) Cortical surface thickness as a classifier: boosting for autism classification. Med Image Comput Comput Assist Interv MICCAI 2008 LNCS 5241:999–1007

Springenberg JT, Dosovitskiy A, Brox T, Riedmiller M (2014) Striving for simplicity: the all convolutional net. arXiv preprint arXiv:1412.6806

Steinwart I, Christmann A (2008) Support vector machines. Springer, Berlin

Takemura A, Shimizu A, Hamamoto K (2010) Discrimination of breast tumors in ultrasonic images using an ensemble classifier based on the AdaBoost algorithm with feature selection. IEEE Trans Med Imag 29(3):598–609

van Ginneken B, Setio AA, Jacobs C, Ciompi F (2015) Off-the-shelf convolutional neural network features for pulmonary nodule detection in computed tomography scans. In: IEEE 12th international symposium on biomedical imaging (ISBI 2015), pp 286–289

Wang J, Li Y, Zhang Y, Wang C, Xie H, Chen G, Gao X (2011) Bag-of-features based medical image retrieval via multiple assignment and visual words weighting. IEEE Trans Med Imaging 30(10):1996–2011

Warfield S (1996) Fast k-NN classification for multichannel image data. Pattern Recogn Lett 17(7):713–721

Xu R, Wunsch D II (2005) Survey of clustering algorithms. IEEE Trans Neural Netw 16(3): 645–678

Xu Y, Mo T, Feng Q, Zhong P, Lai M, Eric I, Chang C (2014) Deep learning of feature representation with multiple instance learning for medical image analysis. In: 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 1626–1630

Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. Eur Conf Comput Vis ECCV 2014:818–833

Zhang W, Li R, Deng H, Wang L, Lin W, Ji S, Shen D (2015) Deep convolutional neural networks for multi-modality isointense infant brain image segmentation. NeuroImage 108:214–224

Validation 13

Abstract

Structures to be analyzed in medical images are usually inaccessible. Sufficiency and correctness of employed domain knowledge cannot be proven. Hence, validation of an analysis method estimates correctness of results from tests on a limited number of samples. For carrying out the validation, suitable samples need to be selected, comparison measures have to be chosen that reflect the quality of the result, and a norm is required against which the method is tested. These aspects are realized differently for a delineation task, a detection task, or a registration task. Requirements, means, and limitations of validation will be discussed in this chapter.

Concepts, Notions, and Definitions Introduced in this Chapter

- Overlap and outlier measures for delineation tasks: over- and under-segmentation, Dice and Jaccard coefficient, Hausdorff distance;
- The ROC curve;
- Success in detection: type I and type II errors, sensitivity and specificity, precision and recall rates;
- Measuring registration errors;
- Ground truth: manual delineation, hardware and software phantoms;
- Training and test data;
- Significance: *t*-test and Welsh test;
- The *p*-value.

An analysis method needs to be tested with respect to the quality with which results are achieved. Unfortunately, a direct way to prove the quality does not exist, since the subjects of analysis are not accessible, except by indirect means. It results in a number of characteristics of a validation procedure:

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_13

- Validation is carried out in a statistical fashion, because the quality of an analysis procedure is tested on a number of sample cases. The outcome is assumed to be representative for all cases.
- Validation is relative, as it gives information about the current method with respect to some other way to generate the analysis results.
- Validation is indirect by comparing features that a correct analysis method should produce instead of comparing analysis methodologies.

Even if validation shows success, underlying assumptions of the validation must not be forgotten. Validation by statistical means, for instance, requires a representative set of data. It may be difficult to show to what extent a chosen set represents the totality of all data sets. A validation with respect to a reference method may be inconclusive, if this method is inferior to other methods for the same analysis task. Finally, validation based on some criterion, e.g., the volume of a segment may not reflect attributes that are needed for the application of the method. Hence, documentation of the validation scenario is an integral part of any validation. It gives a potential user of a method the chance to judge whether the validation is appropriate for his or her purpose.

For image-guided surgery that requires segmentation, registration as well as object detection, Jannin et al. (2002) presented a number of characteristics that should be addressed and documented:

- *Accuracy* measures the deviation of results from known ground truth.
- *Precision and reproducibility* measure the extent to which equal or similar input produces equal or similar results.
- *Robustness* characterizes the change of analysis quality if conditions deviate from assumptions made for analysis (e.g., when noise level increases or if object appearance deviates from prior assumptions).
- *Efficiency* describes the effort necessary to achieve an analysis result.
- *Fault detection* is the ability to detect potential false results during application of an analysis method.

Except for the last, these attributes are also listed by Udupa et al. (2006) who subsume precision and robustness under the same heading.

Accuracy is measured by computing a measure of the quality (Udupa et al. (2006) coined the phrase *figure of merit* = FOM for this) comparing analysis results with some ground truth. Useful measures of quality will be discussed in the next section followed by ways to generate ground truth data.

Quality measures are required for estimating reproducibility as well. However, ground truth is not necessarily required, since only deviations of results for the same or slightly deviating input shall be determined. High reproducibility does not differentiate between correct or wrong results.

Robustness and reliability characterize reproducibility. They are similar in that both judge dependencies of results on changes in the input. The difference is in the causes. A reliable method produces valid results within the range of expected input variation (e.g., acceptable variation of any initialization, expected variation of

appearance of objects etc.). A robust method is insensitive to variations outside of the expected range of variation (e.g., unexpected levels of noise, wrong parameterization, unexpected pathological variation, etc.). While reliability should be a given attribute of a method, robustness will always be limited. Data representing degrees of acceptable variation used for computing reliability should be extended to represent degrees of unacceptable variation (e.g., by adding unusual amounts of noise or by starting with an inacceptable initialization). Reliability and robustness estimates depend on representative data, a question that will be discussed in an own section.

Efficiency describes the effort that is necessary to produce an analysis result. It seems to be easy to compute as it usually refers to the computational cost. However, most of the methods discussed in the previous chapters require some kind of interaction. Hence, Udupa et al. (2006) suggested including human operator time into the efficiency estimate. Even this may by insufficient as the kind of human interaction may influence other quality measures. For instance, expecting some difficult delineation from the human observer at irregular time intervals will lead to a different kind of operator fatigue than boredom from having to wait for giving some simple input. Its assessment requires setting up a user's study which is often too expensive for being done. Still, as part of the validation, the kind of information requested through interaction, the kind of knowledge assumed from the human expert, and the occasions when interaction is requested, should be documented.

Fault detection is a highly desirable feature. It requires a method to estimate the reliability of its own results. Fault detection can be realized by detecting faulty input, faulty intermediate results or faulty output, each of which requiring a model of faultiness at the respective stage of processing. If ranges of faulty behavior can be given, validation of fault detection capabilities requires corresponding ground truth data.

The documentation of the validation scenario builds on the description of the analysis method as mentioned in Chap. 1. Given the intended use of an analysis method and assumptions about the data, the description of a validation scenario should contain the following information:

- Description of the data on which the validation is to be carried out
- Description and justification of what is assumed to be the ground truth
- Criteria by which the quality is to be measured
- Definition and justification of what constitutes a successful validation.

Results from validation are samples nonetheless. Hence, this chapter is concluded with a section on the computation of significance of a validation result.

13.1 Measures of Quality

Quality depends on the kind of analysis that has been carried out on the data. If an object is delineated, it will be a correspondence measure between the delineated object and some reference segmentation. If the task was object detection, it will be a

ratio between correct and incorrect decisions. If it has been a registration, it will be the deviation from the correct registration transformation. Since an analysis task may involve combinations of these methods, the kind of validation procedure will have to be selected based on the intended application.

Let us illustrate this with an example: Delineation of an object boundary may be done by registering a shape model to the image data which is then deforming according to local image information. In this case, the quality of the desired result should be measured by the correspondence of the deformed model instance with the expected segmentation. The correctness of the registration transformation in itself plays only a minor role and will probably not be an interesting part of the validation for the user. However, since an incorrect registration will influence the performance of the subsequent model adaptation to the image data, measuring the registration error will tell something about the robustness of the method (the method may fail completely if the registration part fails). This may be helpful for a developer who wants to adapt this method.

In the following sections, measures for delineation tasks, object detection tasks and image registration tasks will be listed separately although the necessity for combining them should not be forgotten.

13.1.1 Quality for a Delineation Task

Measuring quality for a delineation task requires a measure of correspondence between delineated structure f and some known, true reference delineation g (see Fig. 13.1). Generating ground truth will be discussed in detail in the next section. For now, let us assume that ground truth data exists. Hence, for an image consisting of voxels or pixels $\mathbf{v}$ a function g is known with

$$g(\mathbf{v}) = \begin{cases} 1, & \text{if} \quad \mathbf{v} \text{ belongs to the delineated object} \\ 0, & \text{otherwise.} \end{cases} \tag{13.1}$$

Fig. 13.1 Validation of a delineation requires a measure of comparison between some reference (the often so-called ground truth) and the delineated object

Correspondence can be measured in different ways:

- Volumetric measurements compute volume or area differences between f and g.
- Overlap measures compute the overlap between object and background elements in f and g.
- Distance measurements compute the deviation between the boundaries of f and g.
- Outlier measures compute the maximum deviation between f and g.

It is easily seen that these measures capture different aspects of correspondence. A volumetric measurement only compares a property derived from the delineation. An overlap measure computes an element-wise correspondence. Distance measurements do the same with corresponding boundary points. They may return substantially different results than an overlap measure, since a large change of boundary detail not necessarily results in an equally large change of volume.

Finally, outlier measurements capture singular deviations between f and g. Selecting a type of quality measure depends on the application. If a general quality shall be established, e.g., for promoting a method for other applications apart from that for which it has been developed, all types of quality measures should be used.

Volumetric measurements are easy to compute. They simply count the number of elements $|F|$ and $|G|$ with $F = \{\mathbf{v}|f(\mathbf{v}) = 1\}$ and $G = \{\mathbf{v}|g(\mathbf{v}) = 1\}$ weighted by the area or volume covered by each scene element (pixel or voxel) $\mathbf{v}$ (see Fig. 13.2). They are also the most unreliable measure, since equal volume does not mean that the same object has been delineated. However, these or similar measures such as expected diameter or the size of a bounding box enclosing the object are sometimes the only way to get a quality measure at all when the exact delineation of the object is unknown.

Overlap measures compute the amount of over-segmentation, under-segmentation, or a combination of the two (see Fig. 13.3). *Over-segmentation* is simply the number of elements $\mathbf{v}$, for which $g(\mathbf{v}) = 0$ and $f(\mathbf{v}) = 1$, while *under-segmentation* is the

Fig. 13.2 The volume difference is the simplest FOM for comparing a delineation with the ground truth. It does, however, not account for shape differences

Fig. 13.3 Over- and under-segmentation as well as a combination of the two represent differences in shape between ground truth and the segmentation to be tested

opposite, i.e., $g(\mathbf{v}) = 1$ and $f(\mathbf{v}) = 0$. In order to make these numbers comparable for objects of different size, the amounts are usually given as percent over- or under-segmentation,

$$O = 100 \cdot \frac{|\{\mathbf{v}|g(\mathbf{v}) = 0 \wedge f(\mathbf{v}) = 1\}|}{|G|} \tag{13.2}$$

and

$$U = 100 \cdot \frac{|\{\mathbf{v}|g(\mathbf{v}) = 1 \wedge f(\mathbf{v}) = 0\}|}{|G|}. \tag{13.3}$$

It makes sense to evaluate the two measures separately when the criticality of over-segmentation is different than that of under-segmentation. However, for compactness of presentation, results from the two measures may be combined. Two often used quality measures are the *Dice coefficient* and the *Jaccard coefficient* (Crum et al. 2006). They are borrowed from statistics where they are used to rate the similarity of data sets. The Dice coefficient (Dice 1945) is defined as

$$d = \frac{2|F \cap G|}{|F| + |G|}, \tag{13.4}$$

where $F \cap G$ is the set of all elements $\mathbf{v}$ with $f(\mathbf{v}) = 1$ and $g(\mathbf{v}) = 1$. The coefficient is 1, if the correspondence is perfect, and smaller than 1 otherwise. The Dice coefficient was found to agree well with perceived variability of results of a segmentation in Zou et al. (2004).

The Jaccard index (Jaccard 1912) is given by

$$j = \frac{|F \cap G|}{|F \cup G|}, \tag{13.5}$$

where $F \cup G$ is the set of all elements $\mathbf{v}$ with $f(\mathbf{v}) = 1$ or $g(\mathbf{v}) = 1$. Again, this coefficient is 1, if the correspondence is perfect, and decreases otherwise. The Jaccard index is also known as *Tanimoto coefficient* on sets.

Dice coefficient and Jaccard index are interchangeable, since

$$\begin{aligned} d &= \frac{2 \cdot |F \cap G|}{|F| + |G|} = \frac{2 \cdot |F \cap G|}{|F \cup G| + |F \cap G|} \\ &\Leftrightarrow \frac{1}{d} = \frac{1}{2}\left(\frac{1}{j} + 1\right) \Leftrightarrow d = 2\left(\frac{j}{j+1}\right) \Leftrightarrow j = \frac{d}{2-d} \end{aligned} \tag{13.6}$$

Hence, it is a matter of preference which of the two measures is used, although it appears that in medical image analysis literature the Dice coefficient is slightly more popular than the Jaccard index. The equations above show that the Dice coefficient for the same evaluation is always closer to 1.0 than the Jaccard index.

Outliers cannot be measured by the criteria listed above although they may sometimes be critical. An example would be a task where organ boundaries are to be delineated as part of access planning in minimally invasive surgery. In such case, the maximum deviation of the delineated boundary from the true boundary is an important quality. It can be measured by the *Hausdorff distance* between the two data sets F and G (used, e.g., in Chalana and Kim (1997), Gerig et al. (2001), see Fig. 13.4). The Hausdorff distance h is the maximum of all minimal distances d between points in F and points in G:

$$h = \max\left(\inf_{f \in F} d(f, G), \inf_{g \in G} d(g, F)\right) \tag{13.7}$$

It describes the largest distance of all smallest distances of points in one of the data sets to all points of the other data set. Since this measure is very sensitive to artefacts, a modified version of the Hausdorff distance is sometimes used that averages distances of the largest outliers. The for the quantile h^q computes histograms of distances $d(f, G)$ and $d(g, F)$. For each of the histograms, thresholds $t_q(d(f, G))$ and $t_q(d(g, F))$ are computed so that $q\%$ of all distances are smaller than t_q. The quantile Hausdorff distance is then (see Fig. 13.5)

Fig. 13.4 The Hausdorff distance is the maximum of all shortest distances between points of one set to the other set

Fig. 13.5 The quantile Hausdorff distance for some quantile is computed from a quantile of a histogram of distances from F to G and from G to F

$$h^q = \max\left(t_q(d(f,G)), t_q(d(g,F))\right) \tag{13.8}$$

If the ground truth data consists of the boundaries of delineated segments, the Hausdorff distance may still be computed. Using the Hausdorff distance is based on the assumption that semantically corresponding points are determined by computing shortest distances between boundary points of f and g. If the quality of a delineation result is not appropriately characterized by this, average distances between corresponding locations of f and g can be computed. Point-to-point correspondences can be established using a shape registration algorithm (Chalana and Kim 1997).

13.1.2 Considering Interaction in a Segmentation Task

Most segmentation algorithms do not work fully automatic. Interaction with the experienced user helps to integrate his or her domain knowledge into the delineation even though it is not part of the segmentation model. A semi-automatic approach is sometimes preferred because necessary domain knowledge is difficult to come by or because it would be overly complicate to include this knowledge into the segmentation algorithm.

Besides accuracy, as it has been measured by the methods described above, Udupa and Herman (1999) suggested that *repeatability* and *efficiency* should be evaluated in order to assess the quality of an interactive segmentation method.

An automatic method will always produce the same result if applied several times to the same data. The performance of humans, however, varies based on many factors such as experience, attentiveness to the task or the (perceived)

importance of the decision. This could lead to changes of the interactively given input and it may not depend solely on domain knowledge about the task at hand. Even variations caused by different domain knowledge of the expert user might be critical, since it may make comparisons between data sets impossible that were segmented by different persons (e.g., when tumor segmentation in pre- and post-operatively acquired images was carried out by different operators). Hence, evaluating intra- and inter-observer variability should be part of assessing interactive segmentation. However, although the experimental setup for such an evaluation is simple—let several experts segment several data sets several times—it may nevertheless be difficult to measure repeatability. Experts may simply not be available for the time needed for evaluation.

Efficiency of interaction relates to the efficiency with which the system lets the expert user introduce the domain knowledge into the process. In its simplest form, efficiency is measured by the interaction time required for a user to carry out a segmentation. Other aspects to be measured are, for instance,

- the learning curve, i.e., how much time is needed to understand and apply the required interaction.
- the level of required background knowledge, i.e., how much domain expertise is needed to carry out the interaction.
- the error and correction rate, i.e., how often does the user need to correct his or her own input.
- the correction efficiency, i.e., how much effort is needed for a correction w.r.t. the extent of correction.

Most of these points require a carefully set-up usability study with the intended user group. Again, lack of expert time is a problem for the evaluation.

To summarize, testing the interaction-related aspects of a semi-automatic segmentation method is needed because it provides the future user with an estimate of the required accuracy and work load associated with this method. It may not be carried out nonetheless, because the workload for making the evaluation is deemed to be unacceptable.

13.1.3 Quality for a Detection Task

Quality measures for a detection task are simpler than those for a delineation task. An object to be detected can be found or not and a detection may be correct or not. Given two sets T and F of objects in the ground truth data, where T consists of objects to be detected and F consists of all other objects, and two types p and n of decisions of the detector, where p is a positive decision that the object is detected and n is the negative decision, four different cases arise (see Fig. 13.6):

- *True positive* detections TP are those belonging to T, which have been rightly detected by a positive decision p.

Fig. 13.6 A detection task may result in two different kinds of error

	object present	object not present
object found	True Positive	False Positive Type I Error
object not found	False Negative Type II Error	True Negative

- *True negative* detections TN are those belonging to F, which are rightly resulted in a negative decision n.
- *False positive* results FP are those that do belong to F but resulted in a positive decision p.
- *False negative* results FN are those belonging to T but were classified as n.

Listing detection results organized as in Fig. 13.6 produces the *confusion matrix* for a two-class classification problem. This is also a common means to convey classification results in multi-label detection tasks, as it quickly reveals important characteristics of the tested detector. High values in the diagonal indicate good detection quality, single, high off-diagonal values indicate that a pair of labels is often confused and high values in singles rows or columns of the matrix indicate labels that are particularly difficult to detect correctly. If the number of labels and hence the confusion matric becomes large, it is sometimes rendered as a grey level picture where matrix elements (i, j) are pixels and the grey level represents the number of test cases where the true label i became assigned the label j.

A good detection method would produce as many TP and TN as possible. However, since false positive results (e.g., a tumor is detected while no tumor is present) and false negative results (e.g., a tumor is overlooked) may have wildly different consequences, the two types of error are measured separately. The former is called a *type-I error* and the latter is a *type-II error*. Since the absolute numbers of FP or FN detections do not carry much information, they are normalized with the number of cases tested. This is expressed by the *sensitivity* and *specificity* of a method. Sensitivity Sv is defined as

$$\mathrm{Sv} = \frac{\mathrm{TP}}{T} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{13.9}$$

It represents the rate of positive detections with respect to all elements of T. Sensitivity tells how likely it is to miss a detection by the analysis method (in the information retrieval community it is also called *recall rate Rc*). The specificity Sp is defined as

$$\mathrm{Sp} = \frac{\mathrm{TN}}{\mathrm{N}} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}. \tag{13.10}$$

Fig. 13.7 The ROC curve rates sensitivity versus specificity for different parameter settings of a detection method. The method described by the curve on the right is inferior to the method described by the curve on the *left*, since specificity decreases faster with increased sensitivity

It represents the rate of negative decisions with respect to all elements of N. Specificity tells how likely it is that the detection method produces a false alarm. In the information retrieval community this is replaced by a different measure, called *precision rate* Pr of the retrieval

$$\mathrm{Pr} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}. \tag{13.11}$$

It reflects a different view at the detection task as it measures the amount of "noise" (the false positives) which is generated by the detection algorithm.

Sensitivity and specificity as well as precision and recall are separate measures of the type-I and the type-II error. If the quality of a result shall be described by a single scalar, they can be combined. An often used measure for this is the F_β-score. It is defined as

$$F_\beta = (1 + \beta^2) \cdot \frac{\mathrm{Pr} \cdot \mathrm{Rc}}{\beta^2 \cdot \mathrm{Pr} + \mathrm{Rc}} = \frac{(1 + \beta^2) \cdot \mathrm{TP}}{(1 + \beta^2) \cdot \mathrm{TP} + \beta^2 \cdot \mathrm{FN} + \mathrm{FP}}. \tag{13.12}$$

The parameter β controls the impact of the false positives and the false negatives. Although $\beta = 1$ is usually taken when the F-score is computed, the value may be varied to account for different impacts of false negatives and false positives (e.g., if it is more costly if a tumor is not detected than if a region is falsely declared to be a tumor). If used as a quality measure for detector training it may be used to bias the result accordingly.

Often, a detection task can be parameterized to be more specific or more sensitive. Hence, this trade-off itself is an important characteristic for evaluating a method (Popovics et al. 2007). A simple example would be a threshold classification where decreasing the threshold would increase the number of positive detections. This would increase the sensitivity, since more elements of T would be

included in TP but it also decreases the specificity, since more elements of F would falsely be classified as belonging to T. The trade-off is specific to the detection method and independent of the threshold. Since it measures the performance of a parametrizable detector, it is also used as a quality measure for determining its performance. It is called *receiver operator characteristic* (*ROC*) and measures the ratio of sensitivity versus specificity for each parameter setting α:

$$\mathrm{ROC}(\alpha) = (\mathrm{Sv}(\alpha) \quad 1 - \mathrm{Sp}(\alpha)). \tag{13.13}$$

A curve can be drawn in a graph with axes Sv and Sp where different parameterizations α produce curve locations (see Fig. 13.7). This *ROC curve* represents the performance of the detector independent of the parameterization α. Hence, it is a way to compare two different, parametrizable detectors. An ideal ROC curve would produce sensitivity and specificity of 100% irrespective of the value of α. Since the diagram is normalized with Sv and $1 - \mathrm{Sp}$ ranging from 0 to 1, the area under this ROC curve would be 1.0. The worst result would be an ROC curve, where an increase of Sv would cause an equal increase of $1 - \mathrm{Sv}$. This would mean that increasing the number of correct answers would increase the number of false answers in the same fashion. In this case, the ROC curve would be a diagonal and the area under the curve would be 0.5. Computing the *area under the ROC curve* allows comparing two different, parameterizable detectors.

Sometimes, the ROC curve cannot be generated for the ground truth. This would be the case when the ground truth is provided by human operator decision (it will be difficult to expect from an operator to be gradually more stringent in his or her decisions). A ROC curve can be estimated if several different, independent detection tasks were solved on the same data with the same detection system (e.g., by asking several human operators to solve the same detection task). Each of the solutions provides one point in the ROC diagram. Fitting a curve through these points gives an estimate of the ROC curve for this detector class (see Fig. 13.8).

Fig. 13.8 A ROC curve can also be generated to measure human operator performance when several operators performed the same task

Sensitivity, specificity and the ROC curve have been used to measure the quality of a delineation task as well. In this case, the sets of T, F, p and n are sets of pixels or voxels inside and outside of the delineated object in the ground truth data and the data coming from the delineation method.

13.1.4 Quality for a Registration Task

The purpose of registration is to find a transformation that maps an n-dimensional image onto another m-dimensional image. If dimensions n and m are different, the transformation includes a projection step of the scene with higher dimension onto a space with dimensions of the scene with lower dimension.

Quality of a registration task is measured by direct or indirect measurement of the difference between true and computed registration transformation. Direct comparison requires the true transformation parameters to be known (which is sometimes possible when phantom data is used for providing the ground truth). Average deviation of transformation parameters plus detection of outliers is then the appropriate means to describe how well the registration method succeeded. For non-rigid registration, this transformation is given by a vector field defining local displacements. Differences in the true and the computed vector field are then used as quality measure. For rigid transformations, differences in global rotation and translation are used as quality measure. Since rotation and translation represent different kinds of transformation, they should be measured separately.

If the true transformation is unknown, an indirect way to compute a quality measure is to exchange moving and still image (see Fig. 13.9). The transformation should be the inverse and any deviations are taken as inaccuracies in the computation of the registration transformation. This way of validation should be used with care, however. It implicitly assumes that criteria to find a registration transformation from image A to image B are independent to that of finding a registration from

Fig. 13.9 If the still and the moving image are exchanged for registration, the resulting transformations should be the inverse of each other

Fig. 13.10 Using fiducial markers to validate a registration may yield unexpected results if the markers are far away from the objects to be registered, if they have localization errors, or if they are too few given the number of degrees of freedom for a registration. The optimal fit for fiducial markers does then not automatically mean that the registration itself is optimal

B to *A*. If this is not the case, the quality measure does not tell much about the inherent quality of the registration method.

If the data sets to be registered are of the same kind (e.g., when comparing images of the same patient using the same imaging device over time), the second image can be generated from the first image by transforming and re-sampling the data. This transformation is then compared with that of the registration transformation. Again, this has to be used with care, since neither deviation between the two images to be registered nor re-sampling artefacts are considered.

An indirect way, which is often used for computing validity of a registration transformation, is to compute deviations of point locations after registration. If a number of point pairs are known that represent semantically equivalent locations in the two images, applying the registration transformation to these point locations in one image should map them exactly on locations of their counterparts in the other image. The point pairs are called *fiducial markers*. Of course, the point pair correspondence must not have been used for computing the registration transformation.

Using fiducial markers has the advantage that it evaluates a property that is usually the reason for computing the registration transformation in the first place. One has to be aware, however, that the quality of this mapping is evaluated only at the locations of the fiducial markers (see Fig. 13.10). This may become a problem, if markers are placed in locations that are very different from those where a high accuracy is required (e.g., when only using skin surface markers to test a registration of brain images), or if the registration transformation has many degrees of freedom and only few marker positions are used (e.g., when evaluating a non-rigid registration transformation with few markers).

13.2 The Ground Truth

All measures to estimate the quality of an analysis procedure require comparison of the analysis result with the true information. The truth is difficult to come by, since the reason for producing images in the first place was to gather information about the human body that cannot be accessed otherwise.

Ground truth is the true analysis result of data to which the analysis method is applied. The data can be real or artificial. As we will see below, each of the strategies is tainted, i.e., it is never completely sure whether the selected data really represents the ground truth. An interesting suggestion to deal with this problem came from the validation committee within the ITK consortium (Yoo et al. 2000). They suggested a blind test. Independent observers evaluate analysis results of different data experts (which also could be different established computer analysis methods) with respect to a new method. The new method would be judged as accurate when no significant difference can be found. For automated analysis, the observers will be replaced by an automatic scoring system. Under the assumption that data experts produce unbiased variations of an unknown ground truth (an assumption which also gave rise to the STAPLE procedure, see Sect. 13.2.1), the method is able to rate a new method correctly even if the exact ground truth is unknown.

13.2.1 Ground Truth from Real Data

Ground truth from real data results from applying the currently established best method to it. This is often difficult to determine, however. If an established method exists at all, it may well be a different method for different kinds of image data, for different image acquisition protocols, and for different analysis questions. Hence, even if such a standard exists (e.g., to use active surfaces for liver segmentation in CT or to use mutual information and spline-based non-rigid registration for registering MR brain images), the developer will need to show that the conditions, under which this standard is applied, are comparable to the ones under which this is an established standard.

Another problem is that the implementation of the established state-of-the-art method is not always available. This is changing, because many researchers make their code available and many of the state-of-the-art methods for segmentation and registration have been included into the freely available ITK package (www.itk.org, see Chap. 1).

If a currently best method does not exist, analysis by a human expert is an option to produce ground truth data. It requires some effort on the developer's side and a lot more effort on the data expert's side. The developer needs to provide an appropriate interactive interface for analyzing the data. The required effort from the expert's side is much higher. First, the expert has to carry out the analysis interactively on several data sets. In fact, the more data sets are analyzed, the easier it will be to differentiate between significant and insignificant results. Significance will increase further, if the expert analyses the data several times. *Intra-observer variability* can be estimated that describes variation of judgment by the same individual. A high variability may indicate that the knowledge on which decisions are based is either not very clear, not sufficient, or cannot be applied easily. This is a reason to provide a good user interface, since intra-observer variability should not rate the quality of the input component.

Different experts may have different opinions about the content in the data. Hence, *inter-observer variability* should be measured as well by asking several experts to analyze the data. A good result for the automatic analysis method would be one which is close to a result on which experts agree and within the range of variation among experts. This requires definition of what is meant by agreement among experts. It can be done using a voting system where each expert has a vote (Warfield et al. 1995). Agreement can be reached by majority vote, by requiring total agreement, or by something in between. The situation is less than ideal, since each of the choices for defining agreement makes assumptions about the expertise of the voters without having the chance to validate these assumptions.

STAPLE (simultaneous truth and performance level estimation) (Warfield et al. 2004) is an elegant way to solve this problem for delineation and object detection tasks. The ground truth is assumed to be a set of labeled voxels and the input is a labeling from different experts (or segmentation algorithms) whose reliability is unknown. STAPLE defines the problem probabilistically and treats expert segmentations as samples that deviate in an unknown fashion from the ground truth. It reconstructs the most likely ground truth data given the samples. This, in turn, defines the reliability of each sample. Hence, STAPLE is an application of a maximum likelihood (ML) algorithm to the sample results.

The STAPLE method for a two-label segmentation (e.g., foreground vs. background) presumes that a number of R segmentations exist that assign a binary label to each of N scene elements. It estimates the true segmentation $\mathbf{t} = (t_1, \ldots, t_N)$ from the R segmentations $\mathbf{d}_j$, $j = 1, R$ and $\mathbf{d}_j = (d_{j,1}, \ldots, d_{j,N})$. It further estimates sensitivity $\mathbf{p} = (p_1, \ldots, p_R)$ and specificity $\mathbf{q} = (q_1, \ldots, q_R)$ for each of the R segmentations. Sensitivity and specificity of STAPLE relates to the number of classified voxels or pixels. If one of the R segmentations is new and shall be tested against the other segmentations, sensitivity and specificity of the new method can be used to rate it against the other methods. Other quality measures such as the Dice coefficient can be computed based on $\mathbf{t}$.

Given an initial estimate of the sensitivities and specificities and an estimate for the a priori probability $P(t_i)$ of t_i being a foreground element, Warfield et al. (2004) showed how to use an EM algorithm for optimizing $\mathbf{p}$, $\mathbf{q}$ and $\mathbf{t}$. The E-step computes the current expected probability $w_i^{(k)}$ of an element i to have the label $t_i = 1$ by

$$a_i^{(k)} = P(t_i = 1) \prod_{j \in \{j | d_{i,j}=1\}} p_j^{(k)} \prod_{j \in \{j | d_{i,j}=0\}} \left(1 - p_j^{(k)}\right), \tag{13.14}$$

$$b_i^{(k)} = (1 - P(t_i = 1)) \prod_{j \in \{j | d_{i,j}=0\}} q_j^{(k)} \prod_{j \in \{j | d_{i,j}=1\}} \left(1 - q_j^{(k)}\right), \tag{13.15}$$

$$w_i^{(k)} = \frac{a_i^{(k)}}{\left(a_i^{(k)} + b_i^{(k)}\right)}. \tag{13.16}$$

The terms in the products in Eqs. (13.14) and (13.15) represent conditional probabilities of each segmentation j to arrive at a decision $d_{i,j}$ given that the true segmentation is either $d_i = 1$ or $t_j = 0$, and the current estimates of sensitivity and specificity. Together with the a priori probability $P(t_i = 1)$ and $P(t_i = 0) = 1 - P(t_i = 1)$, these are the right-hand sides of the Bayesian theorem for computing the a posteriori probability for $t_i = 1$ and $t_i = 0$, respectively, given the decisions $\mathbf{d}_j$ and the current estimates for $\mathbf{p}$ and $\mathbf{q}$.

Computing the $w_i^{(k)}$ requires that the different segmentations are independent of each other, since otherwise computing conditional probabilities by multiplying probabilities from each segmentation j is not permissible. A priori probabilities for segmentations where the result at some site i is independent of all other sites, are the ratio of foreground elements to all scene elements. This assumes that segmentations j are also unbiased.

If a priori assumptions include information of spatial dependencies between scene elements, this can be represented by a Markov random field (see Sect. 14.1). Optimization for the general case can be slow (e.g., simulated annealing) or may require a good initial estimate (ICM or graph cuts). However, as shown in Chap. 9, the graph cut solution is an efficient way to find the exact optimum, if the number of labels is two.

Given the result of the E-step, the M-step of the EM algorithm updates sensitivity and specificity values by

$$p_j^{(k+1)} = \frac{\sum_{\{i|d_{i,j}=1\}} w_i^{(k)}}{\sum_{i=1}^{N} w_i^{(k)}}, \tag{13.17}$$

$$q_j^{(k+1)} = \frac{\sum_{\{i|d_{i,j}=0\}} \left(1 - w_i^{(k)}\right)}{\sum_{i=1}^{N} \left(1 - w_i^{(k)}\right)}. \tag{13.18}$$

The initial values for $\mathbf{p}$ and $\mathbf{q}$ use existing information about the performance of the segmentation methods to be tested. In the absence of such information, the authors of Warfield et al. (2004) suggest to start the system with very high values p_j, $q_j < 1$.

The STAPLE procedure can be extended to process multi-label segmentations, however, at the cost that the optimization of a spatially correlated segmentation via an MRF prior will become either slow or suboptimal.

If comparison is carried out with the currently best method, the burden of detecting the truth has been passed to the validation of this method (be it an automatic method or expert analysis). This is even true when elaborate reliability checks such as STAPLE are applied, since they still rely on some basic assumptions (in this case: no biased observer and a good initial estimate).

13.2.2 Ground Truth from Phantoms

Phantoms will exhibit varying degrees of realism. The more realistic the phantom is, the less accessible is the information represented by the phantom. Selecting the appropriate phantom is always a compromise. Based on the degree of realism, phantoms fall into four groups:

- Cadaver phantoms (human or animal).
- Artificial hardware phantoms constructed of material that is known to produce a similar image signal than real data.
- Software phantoms representing the imaged measurement distribution.
- Software phantoms representing the reconstructed image.

In their survey on image segmentation methods in medical imaging, (Pham et al. 2000) classified phantoms of the first two kinds as *physical phantoms* and the latter two as *computational phantoms*. The main difference is that a physical phantom is imaged by the same imaging device than the patient data, while for the latter influences from imaging have to be simulated.

A cadaver or an animal specimen has similar properties than real patient data. Since image acquisition is equal to that of generating patient data, the following attributes of the imaging procedure are represented by the phantom:

- Material properties
- Measurement properties
- Influences from image reconstruction
- Shape properties of the imaged object.

In order to make the phantom data useful for validation analysis, results have to be generated in the phantom. That means,

- for a detection task that number and locations of objects to be detected have to be specified.
- for a registration task that fiducial markers have to be implanted that are visible in the images, or that the phantom is fixed to a reference frame from which transformation parameters can be deduced.

Using a real phantom for a delineation task requires the object to be delineated in the phantom in a way that makes it visible in the images. If only indirect quality measures such as the volume of an organ are needed for validation, real phantoms can be used and the volume is measured afterwards (requiring separation of the anatomic object from the rest of the phantom). Otherwise, delineation information has to be supplied from some other source. An example for this is the visible human project (http://www.nlm.nih.gov/research/visible/visible_human.html) where a male and a female cadaver were imaged by CT and MRI and specimen slices were

then photographed (see Fig. 13.11). The information has been used to delineate anatomical structures in the images (Toh et al. 1996; Spitzer et al. 1996).

Some problems may arise when using such a phantom for validation:

- Attributes of the image phantom may still vary in an unknown fashion from that of the real data.
- Anatomical variability (normal as well as pathological) is not captured by the phantom.

Artificial hardware phantoms capture all but the material properties. Material properties are simulated by selecting material that produces a similar signal than the tissues to be imaged (see Fig. 13.12). Since the artificial structure is built on purpose, aspects for validation (extent of an object, specific landmarks, delineation of object boundaries) are usually known and accessible. Simple hardware phantoms just capture local properties to be measured by the imaging system (e.g., containers of some fluid that emit photons to test nuclear imaging systems). They are still appropriate to validate analysis methods if knowledge about appearance and shape of objects or features of interest as well as information about spatial relationships to surrounding tissues has not been used for the analysis. More realistic phantoms exist, e.g., for simulating imaging of vascular structures. Phantoms are usually single instances so that anatomical variation cannot be modeled. However, they are well-suited to compare the quality of different methods applied to the same data, such as the comparison of different image reconstruction algorithms (see Fig. 13.13).

Fig. 13.11 CT and MRI slices and the imaged cytographic slices from the Visible Human Project (created using a browser of from the Visible Human page of the University of Michigan at http://vhp.med.umich.edu/)

Fig. 13.12 Example of an artificial phantom. The picture shows a hardware phantom for the simulation of SPECT images of the heart. The phantom consists of several bottles of photon-emitting fluids that have a similar characteristic than the organs that are simulated. The shape of the bottles resembles (at the spatial resolution of SPECT imaging) the shape of the organs (with kind permission of Anna Celler, Department of Radiology, University of British Columbia, Vancouver)

Software phantoms representing the measured image signal are the next level of abstraction. The phantom consists of a 3d distribution of the signal to be measured by the image acquisition system (e.g., X-ray absorption for an X-ray CT phantom). The 3d signal distribution may be generated from interpreted image material [e.g., the MRI BrainWeb phantom (Collins et al. 1998)]. Creating the phantom requires knowledge about original material coefficients (e.g., T_1, T_2 and proton density for the BrainWeb phantom). This information is taken from experiments. It is often idealized, e.g., by assuming that the material is homogeneous within an organ. Noise and artefact models of image acquisition are required for simulating the measurement. Effects from image reconstruction and shape properties of objects in the image are part of the phantom. The advantage of a software phantom compared to a hardware phantom is that it easily allows inclusion of known anatomical variation by creating several different shape phantoms. Also, the original properties of the objects to be represented are easily accessible. A number of software phantoms of this kind exist:

- The BrainWeb phantom, www.bic.mni.mcgill.ca/brainweb/, (Cocosco et al. 1997; Collins et al. 1998; Kwan et al. 1999) was created from a single subject. Although not representing inter-patient variability, it gives a realistic representation of the typical appearance of brain structures (see Fig. 13.14). A simple

Fig. 13.13 Different reconstruction algorithms from data acquired from the phantom depicted in Fig. 13.12. The *top row* show the measured emission and transmission data from the phantom, the middle and the *bottom row* shows reconstruction results using the OSEM algorithm (see Chap. 2) including various restoration techniques (*AC* attenuation correction, *RR* resolution recovery, *SC* scatter correction) presented in Shcherbinin et al. (2008) compared to standard reconstruction technique of the GE evolution scanner (with kind permission of Anna Celler, Department of Radiology, University of British Columbia, Vancouver)

user interface allows various parameterizations to produce different simulations of MRI images.

- The Field II ultrasound simulation program (see Fig. 13.15), to be found at http://server.electro.dtu.dk/personal/jaj/field/, (Jensen and Svendsen 1992; Jensen 1996, 2004) simulates the creation of ultrasound images from various input data. It is available as Matlab software.
- The group of Segars et al., http://www.bme.unc.edu/~wsegars/index.html and http://dmip1.rad.jhmi.edu/xcat/, see Segars et al. (1999), Segars and Tsui (2009) developed several phantoms for CT and Nuclear Medicine such as the dynamic MCAT heart phantom simulating a moving heart which is based on normal anatomy and spline-based interpolation of dynamics, the NCAT phantom that has been extended to also include the upper airway tree and the lung lobes and to simulate normal and pathological variations (Garrity et al. 2003; Veress et al.

Fig. 13.14 Images generated from the BrainWeb software phantom (Collins et al. 1998) look very realistic. The simulation was done using the phantom that can be downloaded from www.bic.mni.mcgill.ca/brainweb/

Fig. 13.15 Depiction of an ultrasound phantom and the simulated ultrasound image. The simulated image was created using the Field II ultrasound simulation program of (Jensen and Svendsen 1992, 1996)

2006), and its extension, the XCAT. MCAT and XCAT are created from parameterizable geometric primitives that model patient-specific anatomic variation. The phantoms are used to validate SPECT and PET data (see Fig. 13.16).

Fig. 13.16 Emission and transmission image of the NCAT phantom. Projections can be reconstructed to simulate CT, SPECT or PET imaging. In this case, different reconstruction algorithms for SPECT images are compared integrating attenuation correction (AC) and resolution recovery (RR) in the OSEM reconstruction algorithm (see Shcherbinin et al. (2008) for details, with kind permission of Anna Celler, Vancouver General Hospital)

Since material and measurement properties are often idealized, images may differ from images of real objects even though they look realistic. Promising results on phantom data can then be misleading, since they do not seem to agree with the perceived performance on real data.

The highest level of abstraction is the use of a software phantom that models the appearance of an imaged object. Only the shape of the object is part of the phantom, while all other aspects that influence the appearance in the image need to be simulated. The shape information either stems from interpreted images (e.g., the MNI brain atlas of Collins et al. (1998) to represent variation of the human brain anatomy) or—in an even more abstract fashion—it reflects just those shape properties that are assumed to be important for the analysis task. Artefacts from material, measurement or reconstruction are simulated. Observed distortions in the image are modeled by the simulation model. The following are examples for artifacts that are modeled:

- Detector and measurement noise is usually combined as zero-mean Gaussian noise.
- Partial volume effects and effects from the detector point spread function are modeled by smoothing the data.
- Variation in signal strength is modeled by including artificial shading.

The artefact model may be approximate, even if the images look similar to true images.

In summary, the phantoms listed above produce ground truth only to the extent to which influences on the image generation have correctly been accounted for. Choosing a phantom is a compromise between realism of the images generated and accessibility to the ground truth. Ideally, the phantom itself would be validated, i.e., it would be investigated to what extent the phantom data reflects the reality. Since this is usually not done, using a phantom for validating an analysis procedure should include information as to what influences on the image content were simulated, on what information the simulated properties were based, and on what kind of information the analysis procedure relies. This gives others the chance to decide on the validity of arguments for their own purposes.

13.3 Representativeness of Data

Validation will be based on statistical arguments (such as, "in 99.2% of all cases, volume deviated by less than 5% from the ground truth"). Hence, besides trusting ground truth and the appropriateness of the quality measures, representativeness is another issue for validation. Using representative data means that all data properties potentially influencing the performance of the analysis method are reflected in the test data. A couple of strategies strengthen the argument of representativeness:

- Separation between test and training data
- Identification of sources of variation
- Identification of outliers
- Investigation of robustness with respect to parameter variation.

13.3.1 Separation Between Training and Test Data

The separation between training data and test data is self-evident if classification is the goal of the analysis. However, other analysis tasks include training as well, for instance, when optimal parameters have to be determined. Parameters are anything that needs to be fixed prior to carrying out the analysis. This could be a stopping criterion for an iterative algorithm or a threshold for a decision component. As in classification, it is not acceptable to validate a parameter-dependent segmentation on ground truth data which has been used to compute the optimal parameter value.

Often, the set of data that can be used for training and testing is very small. Separating it into two even smaller subsets would further reduce the significance of results. A workaround is to use the *leaving-one-out technique* (also called *jackknife technique*, see Fig. 13.17 for an example). A data set of N elements, for which the ground truth is known, is separated in a training set of $N - 1$ elements and a test set

Fig. 13.17 Leaving-one-out technique for a classification task. Repeatedly, a sample is left out from training and the resulting decision boundary is tested on this sample. The test is to be carried out using every sample as left-out sample

consisting of just one element. Parameter estimation is done on the training set and the quality is then measured on the single test element. This is done for all N subsets of $N - 1$-element training sets and 1-element test sets. The overall quality is then computed by combining the N different test results. The leaving-one-out technique is a special case of the leaving-n-out technique, where n instead of just one sample is left out.

A more relaxed version is n-fold cross validation which does not use all possible subsets of n elements. The data is arbitrarily separated into n different, disjoint subsets. At each of n tests one of the subsets is used to test the classifier against the data from all the other subsets.

13.3.2 Identification of Sources of Variation and Outlier Detection

Representativeness means that typical sources of variation in the data are covered by the ground truth data. In order to test the behavior in extreme cases, outliers should be identified. If data consists of patient data that has been analyzed by the current best method or by a human expert, sources of variation and outliers can be found with the help of an expert. The expert will be able to tell whether the images reflect the "usual appearance" of the objects to be analyzed, but it will require some discussion between data expert and method expert, nonetheless. The reason is that analysis methodology will act differently for different sources of variation. A detection method that mainly relies on the detection of objects in the vicinity of other objects will be insensitive to object variations, but very sensitive to variations in appearance of neighboring objects. A data expert will only perceive this as important if it is a criterion for the expert's decision as well.

Another source of misunderstanding regarding variation and outliers is the difference of the task for the expert and for a computer method. The expert often analyses single cases with correct output expected for this case while the computer-assisted analysis method has to work in general for all potential cases.

The expert judges the single case using additional information from anamnesis and other diagnostic imaging or non-imaging information. The notions of "usual" or "outlier" may be based on all this information and not on the image alone. In order to prevent this kind of misunderstanding, it should be made very clear that the discussion is about the appearance of an object in the image.

Discussion is usually easier for the engineer or computer scientist when phantoms are the ground truth. The phantom is generated with keeping variation with respect to appearance in the image in mind. Furthermore, if variable phantoms have been created, the different sources of variation (noise, artifacts, patient-specific changes in shape and appearance) have been identified and ranges of normal and abnormal variation have been defined. It may be difficult to establish whether all sources of variation are covered, however. This is true for technical as well as for anatomical sources of variation. For the variation of technical parameters or models, it cannot often be proven that the model captures exactly the reality (e.g., when assuming a certain kind of noise) and it is left to the developer to decide whether the model still produces a good validation scenario. For anatomical variation, certain simple parameters, e.g., size or length-to-breadth ratio, are often measured, but it cannot be guaranteed that these parameters reflect everything that is necessary for producing a representative set of phantoms. Furthermore, pathological variations are often not captured at all.

Given the remarks above, it seems that estimating normal and extraordinary variation is impossible. This is fortunately untrue, since some sources of variation can be captured with some accuracy. Using this information is preferable to using the argumentation above for just choosing arbitrary data for validation. Disclosing assumptions about sources of variation helps others to rate the validity of these arguments.

Outliers can also be detected based on their very nature. If they are so different from the rest of the cases, computable properties used as a priori knowledge for the analysis method will exist that are different for these data sets as well. Hence, it will be impossible to predict the property of an outlier from non-outlier data. This can be used in *cross-validation*.

Given N data sets, cross-validation computes predictions of these properties from all subsets of $N - 1$ elements and applies this prediction to the remaining data set. The $N - 1$ elements would be a sufficiently high number to generate an estimate of a likelihood function. If the property value of the left-out Nth element has a low probability according to this likelihood function, the element is probably an outlier and should be removed. Outlier removal continues until no further outliers can be found, or if the data set becomes too small. In the latter case, the outliers actually characterize the data. There are a number of reasons for this:

- The properties used as a priori knowledge are inappropriate for the analysis method because parameter settings have to be adapted for every single case.
- The data falls into several classes for which separate methods need to be found.

- The acquisition of ground truth data failed as the data is apparently not representative for the totality of all data (if it were representative, outliers would not occur this often).

13.3.3 Robustness with Respect to Parameter Variation

Given the argumentation above, it can be assumed that a developer or user will never be completely sure that test data was representative. Any parameterization from a training stage will have been generated from (slightly) non-representative data. Hence, parameters would be preferred where small changes do not cause a large change in the analysis result. It simply means that the main power of the analysis is in the method and in its combination of sub-methods rather than in the parameterization.

Finding parameters and testing the method with respect to changes of parameter values can be costly because every possible combination of parameters needs to be tested. It may become unfeasible, if the number of parameters is large. Hence, it is worthwhile trying to detect parameters that are most likely independent of each other. Doing this by simply testing would be unfeasible again. However, if the analysis methodology has been developed with exact documentation about information that is exploited for the analysis and about ways how this is used for carrying out the analysis, it is usually possible to hypothesize which parameters may be independent.

This is especially the case when the method is organized as a sequence of processes where each process has its own set of parameters and produces a well-defined intermediate result. If the intermediate results can be predicted for the ground truth data, parameters can be optimized and tested for each process independently. Even if subsequent processes are able to correct erroneous intermediate results, the optimal result for each step can be defined independent of the other steps.

13.4 Significance of Results

Results of an analysis may be satisfactory, but there is always the question whether the number of samples was sufficiently high for having a predictive value for the universe. Given a data set $S = \{s_1, \ldots, s_N\}$ reflecting the ground truth represented by a quality value $q_{\mathrm{G}}(s)$ and test results $q_{\mathrm{T}}(s)$, the question is how likely they describe the same entity except for some error $E(q_{\mathrm{G}} - q_{\mathrm{T}})$ Given the ground truth, an established method T_{e} and a new method T_{n}, the question is whether T_{n} really produces better results.

Fig. 13.18 The purpose of the student *t*-test is to determine how likely two distributions of observations belong to the same observation. It is often tested by estimating how likely the expected value of one of the two distributions belongs to the other distribution

Significance of a test result can be computed by estimating the probability that it arose by chance. This percentage is the *p*-value that is used to describe the significance of the outcome of an experiment. The statement "the results are significant with $p < 0.01$" simply means that the probability that the results arose by chance is less than 1%.

Intuitively, significance depends on the number of samples and on the amount of similarity or dissimilarity between the two populations (ground truth vs. test or method *A* vs. method *B*). Significance can be tested by the *Student's t-test* (also called *t*-test). It computes the probability whether quality measurements q_{new} from a new method could be the result of some natural fluctuation of measurements q_{old} that were computed with an earlier method (see Fig. 13.18).

In order to formulate this as a problem of computing probabilities, several aspects have to be specified. First, a *null hypothesis* is formulated. A null hypothesis assumes that nothing has happened. In our case, this means that the new method produces the same kind of results than the old method. A probability has to be computed that they have been drawn from the same probability distribution.

The new method is significantly different if this is not likely. It is significantly better if it is significantly different and results are better. If probability characteristics of quality measures of both the new and the old method were known, this would be easy. One could simply compute the probability with which the expected value of the distribution of the new method is a value from the distribution of quality measures of the old measure.

Probability distributions are usually unknown and can only be estimated from the sample results from q_{new} and q_{old}. The reliability of this computation depends on the number of samples for these estimates. This is captured by the Student's *t* test. Many different variants of this *t*-test exist and the interested reader is referred to (Aron et al. 2004) for further details. Here, we will present two versions that are often appropriate for computing significance of the performance of an analysis method compared to the performance of some previous method. They are called the un-paired and paired *t*-test. It will be presented for normal distributed data where mean and variance are unknown.

Let us assume that we have samples $X_1, X_2, \ldots, X_N$ (e.g., some quality measure with respect to ground truth) with mean μ and variance σ^2. The average of the samples is

$$\overline{X}_N = (X_1 + \cdots + X_N)/N. \tag{13.19}$$

The estimated variance of the samples is then

$$S_N^2 = \frac{1}{N-1} \sum_{i=1}^{N} \left(X_i - \overline{X}_N\right)^2 \tag{13.20}$$

The following parameter is then normal-distributed with zero mean and normalized variance

$$Z = \frac{\overline{X}_N - \mu}{\sigma/\sqrt{N}}. \tag{13.21}$$

Now let us assume that we have an unknown sample w. How likely is it that this sample is the mean of a distribution from which we have the N samples X_i? For this, a *confidence interval* is computed. It is

$$\int_{-a}^{a} Z(w)\mathrm{d}w = p \tag{13.22}$$

and defines an interval $[-a, a]$ so that the probability for w is p. E.g., the 90% confidence interval

$$\int_{-a}^{a} Z(w)\mathrm{d}w = 0.90 \tag{13.23}$$

defines an interval bounded by $[-a, a]$ for which the probability is 90% that w belongs to Z. In other words, if w is outside the interval, the probability is only 10% that w belongs to Z. If we restrict a to be positive (e.g., if our value of w represents a change in quality and we are only interested in positive changes), then the same a characterizes a 95% confidence interval.

In order to compute the confidence interval from the samples Z, w is transformed into the normalized normal distribution:

$$
\begin{aligned}
&P(-a<w<a)=p \\
&\Leftrightarrow P\left(-a<\frac{\overline{X}_N-\mu}{\sigma/\sqrt{N}}<a\right)=p \\
&\Leftrightarrow P\left(\overline{X}_N-a\frac{\sigma}{\sqrt{N}}<\mu<\overline{X}_N+a\frac{\sigma}{\sqrt{N}}\right)=p
\end{aligned}
\tag{13.24}
$$

This allows us to compute the value for a given the standard deviation σ. Since σ is unknown, the distribution is replaced by the estimate from the samples (this is the *t-distribution*):

$$
\begin{aligned}
&P(-a<w<a)=p \\
&\Leftrightarrow P\left(-a<\frac{\overline{X}_N-\mu}{S_N/\sqrt{N}}<a\right)=p \\
&\Leftrightarrow P\left(\overline{X}_N-a\frac{S_N}{\sqrt{N}}<\mu<\overline{X}_N+a\frac{S_N}{\sqrt{N}}\right)=p
\end{aligned}
\tag{13.25}
$$

Hence, $\overline{X}_N \pm a\frac{S_N}{\sqrt{N}}$ is the confidence interval with probability p that w is in Z. The values for a are tabulated and tables and Excel sheets exist for given probabilities p. When computing the t-test, one-sided and two-sided tests have to be differentiated. A one-sided t-test assumes an interval $[-\infty, a]$ and w is outside the interval if $w > a$. Hence, the 90% confidence interval of a two-sided t-test results in the same value of a than the 95% confidence interval of the one-sided t-test.

In the t-test above, pairings between results from the old method and those from the new method were not considered. This allows comparison between methods even if the samples were generated from different data sets, as long as they can be assumed to be equally representative for the problem. However, since this cannot always be shown (or even made plausible), tests are often carried out on the same data set. Pairs of results (X_1,Y_1), …, (X_N,Y_N) exist where X is the quality of the old method and Y the result of the new method.

A *paired t-test* accounts for this using the mechanism above, but now with the following distribution function

$$
t=\frac{\overline{P}_D-\mu}{S_D/\sqrt{N}}, \tag{13.26}
$$

$$
\overline{P}_D=\frac{1}{N}\sum_{i=1}^{N}(X_i-Y_i), \tag{13.27}
$$

$$
S_D=\frac{1}{N-1}\sum_{i=1}^{N}\left[(X_i-Y_i)-\overline{P}_D\right]^2. \tag{13.28}
$$

The difference between results of the old and new method is taken as sample value. The null hypothesis would be $\mu = 0$.

Since the t-test essentially tests the probability of the expected value of a distribution to be a value of the distribution of the null hypothesis, it implicitly assumes that the variances of the two distributions are equal. If this cannot be assured, the *Welch test* (Welch 1947) is a variant that should be applied. The difference between the t-test and the Welch test is that the t-distribution is now generated from the two different sample distributions X_1 and X_2 by normalizing them with the estimated variances s_1 and s_2 from both distributions, leading to

$$t = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}} \tag{13.29}$$

This is then used in the same way for computing significance than the unpaired t-test for the Student's t-distribution.

13.5 Concluding Remarks

Validation of an analysis method is a necessary prerequisite for introducing any new analysis method to the scientific community, since it cannot be verified that domain knowledge is sufficient and correct for solving the analysis problem. Validation is difficult because sufficiency and correctness have to be corroborated from results on representative test data. An acceptable validation needs to show to what extent tests are estimates for such a generalization. Furthermore, it has to be made plausible that quality measures and comparison data really support the argumentation.

Validation is often a hen-and-egg problem. Hence, a detailed description of the validation scenario and a well-founded justification as to why which measures were selected, is mandatory. Even though this is still not a formal verification, it enables the reader and potential future user of a method to investigate the validity of the arguments used for setting up the validation scenario.

13.6 Exercises

- Why is it necessary to document the validation scenario and not just the results from validation?
- What are the major components of a validation scenario and what should be documented?

- Describe a situation where volume computation would be an appropriate criterion for measuring the quality of a delineation task. When should it not be used?
- Under what conditions does it make sense to differentiate between over- and under-segmentation?
- What information about delineation quality is revealed by the Hausdorff distance? Please describe a scenario where this measure is important to rate a delineation method.
- What information is captured by sensitivity and specificity? For what reason is specificity needed to rate a detection method? Give an example.
- Imagine a registration method that has been developed to register pre- and post-operative CT scans. What are the options to be used as ground truth? Discuss advantages and disadvantages of the different options.
- What needs to be made sure when selecting test data for ground truth?
- Why is it necessary to carry out manual segmentation several times by different and by the same person if it shall be used for ground truth? How is the information that is gained from these multiple segmentations used for rating the performance of an algorithm?
- What is the difference between hardware phantoms and software phantoms? What are the advantages of using a software phantom?
- How can which kinds of influences from image acquisition be represented in a software phantom?
- What are the limitations when representing anatomic variations by a software phantom?
- What is meant by robustness with respect to parameter variation? Why is it important that a method is robust with respect to parameter variation?
- Give an example from segmentation where it is necessary to differentiate between training and test data.
- What is the basic idea of STAPLE? What advantage does this have when validating a method?
- What is the meaning of "*p*-value"?
- Assume that results of a method A on data AT and results of method B on data BT have been generated. It can be assumed that AT and BT are samples from the same distribution of test data. What would be an appropriate null hypothesis for a *t*-test?

References

Aron A, Aron EN, Coups E (2004) Statistics for the behavioral and social sciences: a brief course. Prentice-Hall, Upper Saddle River

Chalana V, Kim Y (1997) A methodology for evaluation of boundary detection algorithms on medical images. IEEE Trans Med Imaging 16(5):642–652

Cocosco CA, Kollokian V, Kwan RKS, Pike GB, Evans AC (1997) BrainWeb: online interface to a 3d MRI simulated brain database. Neuroimage 5:452

Collins D, Zijdenbos A, Kollokian V, Sled J, Kabani N, Holmes C, Evans A (1998) Design and construction of a realistic digital brain phantom. IEEE Trans Med Imaging 17(3):463–468

Crum WR, Camara O, Hill DLG (2006) Generalized overlap measures for evaluation and validation in medical image analysis. IEEE Trans Med Imaging 25(11):1451–1461

Dice LR (1945) Measures of the amount of ecologic association between species. Ecology 26(3):297–302

Garrity JM, Segars WP, Knisley SB, Tsui BMW (2003) Development of a dynamic model for the lung lobes and airway tree in the NCAT phantom. IEEE Trans Nucl Sci 50(3):378–383

Gerig G, Jomier M, Chakos M (2001) Valmed: a new validation tool for assessing and improving 3D object segmentation. In: Proceedings of 4th international conference on medical image computing and computer-assisted intervention (MICCAI 2001), LNCS, vol 2208, pp 516–523

Jaccard P (1912) The distribution of flora in the alpine zone. New Phytol 11:37–50

Jensen JA (1996) Field: a program for simulating ultrasound systems. In: 10th Nordic-Baltic conference on biomedical imaging, medical and biological engineering and computing, vol 34(1), pp 351–353

Jensen JA, Svendsen NB (1992) Calculation of pressure fields from arbitrarily shaped, apodized, and excited ultrasound transducers. IEEE Trans. Ultrason Ferroelectr Freq Control 39:262–267

Jensen JA (2004) Simulation of advanced ultrasound systems using Field II. In: IEEE international symposium on biomedical imaging nano macro, vol I, pp 636–639

Jannin P, Fitzpatrick JM, Hawkes DJ, Pennec X, Shahidi R, Vannier MW (2002) Validation of medical image processing in image-guided therapy. IEEE Trans Med Imaging 21(12): 1445–1449

Kwan RKS, Evans AC, Pike GB (1999) MRI simulation-based evaluation of image-processing and classification methods. IEEE Trans Med Imaging 18(11):1085–1097

Pham DL, Xu C, Prince JL (2000) Current methods in medical image segmentation. Annu Rev Biomed Eng 2:315–337

Popovic A, de la Fuente M, Engelhardt M, Radermacher K (2007) Statistical validation metric for accuracy assessment in medical image segmentation. Intl J Comput Assist Radiol Surg 2(3–4):169–181

Segars WP, Lalush DS, Tsui BMW (1999) A realistic spline-based dynamic heart phantom. IEEE Trans Nucl Sci 46(3):503–506

Segars WP, Tsui BMW (2009) MCAT to XCAT: evolution of 4-d computerized phantoms for imaging research. Proc IEEE 97:1954–1968

Shcherbinin S, Celler A, Belhocine T, Vanderwerf A, Driedger A (2008) Accuraccy of quantitative reconstructions in SPECT/CT imaging. Phys Med Biol 53(17):4595–4604

Spitzer V, Ackerman MJ, Scherzinger AL, Whitlock D (1996) The visible human male: a technical report. J Am Med Inf Assoc 3(2):118–130

Toh MY, Falk RB, Main JS (1996) Interactive brain atlas with the Visible Human Project data: development methods and techniques. RadioGraphics 16:1201–1206

Udupa JK, Herman GT (eds) (1999) 3D imaging in medicine. CRC press, Boca Raton

Udupa JK, LeBlanc VR, Zhuge Y, Imielinska C, Schmidt H, Currie LM, Hirsch BE, Woodburn J (2006) A framework for evaluating image segmentation algorithms. Comput Med Imaging Graph 30:75–87

Veress AI, Segars WP, Weiss JA, Tsui BMW, Gullberg GT (2006) Normal and pathological NCAT image and phantom data based on physiologically realistic left ventricle finite-element models. IEEE Trans Med Imaging 25(12):1604–1616

Warfield SK, Dengler J, Zaers J, Guttmann CR, Wells WM III, Ettinger GJ, Hiller J, Kikinis R (1995) Automatic identification of grey matter structures from MRI to improve the segmentation of white matter lesions. J Image Guided Surg 1(6):326–338

Warfield SK, Zou KH, Wells WM III (2004) Simultaneous truth and performance level estimation (STAPLE): an algorithm for the validation of image segmentation. IEEE Trans Med Imaging 23(7):903–921

Welch BL (1947) The generalization of "Student's" problem when several different population variances are involved. Biometrika 34(1–2):28–35

Yoo TS, Ackerman MJ, Vannier MW (2000) Toward a common validation methodology for segmentation and registration algorithms. In: Medical image computing and computer-assisted intervention—MICCAI 2000, LNCS, vol 1935, pp 217–225

Zou KH, Warfield SK, Bharatha A, Tempany CMC, Kaus MR, Haker SJ, Wells WM III, Jolesz FA, Kikinis R (2004) Statistical validation of image segmentation quality based on a spatial overlap index. Acad Radiol 11(2):178–189

Appendix 14

Abstract

The appendix contains a detailed discussion of selected mathematical aspects that are necessary for many of the methods presented in this book. In its three sections, Markov random fields and their optimization, a derivation of the solution of a variational problem for a function of a single variable, and a description of the principal component analysis including a solution that is robust with respect to outliers in the sample are presented.

Concepts, Notions, and Definitions Introduced in this Appendix

- Markov random fields, neighborhood systems, simulated annealing, mean field annealing, iterated conditional modes;
- Variational calculus, Euler–Lagrange equation, principal component analysis, robust PCA, outliers and outlyingness.

14.1 Optimization of Markov Random Fields

14.1.1 Markov Random Fields

If a vector $\mathbf{g}$ is an observation of a random process with mean $\mathbf{f}$, the relation between $\mathbf{g}$ and $\mathbf{f}$ is given by the likelihood function $p(\mathbf{g}|\mathbf{f})$ of observing $\mathbf{g}$, if the mean were $\mathbf{f}$. A priori knowledge about the nature of $\mathbf{f}$ is given by the a priori probability $P(\mathbf{f})$ of observing $\mathbf{f}$. If the random process is assumed to be an artifact of the observation (which is the case when noise is distorting a deterministic $\mathbf{f}$), restoration aims to recover the expected value $\mathbf{f}$ given the observation $\mathbf{g}$. This is expressed by the Bayesian theorem

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_14

$$P(\mathbf{f}|\mathbf{g}) = \frac{1}{P(\mathbf{g})} p(\mathbf{g}|\mathbf{f}) \cdot P(\mathbf{f}), \tag{14.1}$$

where $P(\mathbf{g})$ is the a priori probability of $\mathbf{g}$ and acts as a normalizing factor.

If $\mathbf{f}$ and $\mathbf{g}$ represent images, the a priori knowledge about $\mathbf{f}$ is often related to local homogeneity (e.g., when predicting that it is likely that neighboring pixels have the same gray value). Hence, neighborhood has to be defined among elements of $\mathbf{f}$. The probability $p(\mathbf{f})$ must take this neighborhood information into account. It can be expressed by defining the probability as a *Markov random field* (*MRF*). The following is an overview about Markov random fields in their use for restoration and segmentation. Further details and a detailed discussion of other types of MRFs can be found in Li (2009). Details on the optimization of MRFs are also found in the chapter on stochastic optimization in Duda et al. (2001).

A set of random variables $\mathbf{f} = \{f_1, f_2, \ldots, f_i, \ldots, f_k, \ldots\}$ is an MRF, if the conditional probability of an element f_i only depends on the probabilities of elements f_k in some neighborhood of i. Neighborhood can be arbitrary but must be finite. Variables and the neighborhood system constitute a graph with f_i being the nodes and neighboring nodes being connected by an edge.

For regular lattices representing images, neighborhood systems can be ranked by the distance of neighboring elements to the center. A neighborhood of rank 1 of a 2D image would comprise the 4-neighbors, a rank-2 neighborhood would include the 8-neighbors plus pixels in x- and y-direction that are 2 pixels away and so on (see Fig. 14.1).

For a given neighborhood system, dependencies between nodes are defined by clique potentials. A *clique* in a graph is a collection of nodes that is fully connected (every node is connected to every other node, see Fig. 14.2 for rank-1 cliques). The potential V determines the dependency between members of a clique.

Fig. 14.1 Neighbor sizes on a regular grid can be differentiated by the distance to the furthest pixel, which is indicated by the rank

Fig. 14.2 In a neighborhood of rank 1 several 1st, 2nd, and 3rd order cliques exist

The Hammersley–Clifford theorem states that for random variables ω, possessing MRF quality, the probability distribution is a Gibbs distribution depending on clique potentials

$$\begin{aligned} P(\mathbf{f} = \omega) &= \frac{1}{\sum_{\omega} \exp(-U(\omega))} \exp(-U(\omega)) \\ &= \frac{1}{\sum_{\omega} \exp(-U(\omega))} \exp\Big(-\sum\nolimits_{c \in C} V_c(\omega)\Big), \end{aligned} \tag{14.2}$$

where the potential $U(\omega)$ is the sum of all clique potentials $V_c(\omega)$ of cliques $c \in C$ for a given neighborhood system. The normalization factor sums potentials over all possible configurations. If we assume that the likelihood function $p(\mathbf{f}|\mathbf{g})$ is a Gaussian with mean vector $\mathbf{f}$

$$p(\mathbf{g}|\mathbf{f}) = \frac{1}{Z} \exp\left(-\frac{1}{2}(\mathbf{f} - \mathbf{g})\Sigma(\mathbf{f} - \mathbf{g})^{\mathrm{T}}\right), \tag{14.3}$$

this is the energy U for a clique of size 1. It can be integrated into the a priori probability term so that the probability to maximize is

$$\begin{aligned} p(\mathbf{f}|\mathbf{g}) &= \frac{1}{P(\mathbf{g})} p(\mathbf{g}|\mathbf{f}) \cdot P(\mathbf{f}) \\ &= \frac{1}{\sum_{\omega} \exp(-U(\omega))} \exp\Big(-\sum\nolimits_{c \in C} V_c(\omega)\Big). \end{aligned} \tag{14.4}$$

For maximizing $p(\mathbf{f}|\mathbf{g})$ using a maximum likelihood approach, it is necessary to proceed from some configuration $\omega^{(n)}$ to the next configuration $\omega^{(n+1)}$ so that the a posteriori probability of the configuration increases. It requires computation of the normalizing constant in (14.4), which is called the *partition function*. This is unfeasible, however, since it sums over all possible configurations and increases exponentially with the number of nodes.

The partition function can be neglected if $U(\omega)$ is fixed. Given an appropriate definition of the potentials V, maximizing (14.4) then means minimizing

$$e = \sum_{c \in C} V_c(\omega). \tag{14.5}$$

Optimization is still not simple, since elements of **f** and **g** are dependent on each other through the covariance matrix Σ and the neighborhood dependencies contained in the definition of the clique potentials. Optimization is further complicated by the fact that the landscape produced by the energy function has many local maxima. Finding the global maximum cannot be done by a simple gradient ascent.

Several methods have been presented to find an optimal **f** based on equation (14.5) of which some of the established methods will be discussed in the following sections.

14.1.2 Simulated Annealing

Optimization for solving a restoration task by *simulated annealing* has been presented by Geman and Geman (1984). It is a stochastic optimization technique that iterates through configurations of **f**. The term of the method refers to the simulation of an annealing process of heated metal that—starting from an arbitrary configuration—takes its energy-minimal configuration during the cooling process. Locally and globally, minimal configurations are the local maxima of a parameterized Gibbs distribution where a temperature parameter T represents the actual state of cooling. Equation (14.2) reads now

$$\begin{aligned} P(\mathbf{f} = \omega) &= \frac{1}{\sum_{\omega} \exp(-U(T,\omega))} \exp\left(-\frac{U(\omega)}{T}\right) \\ &= \frac{1}{\sum_{\omega} \exp(-U(T,\omega))} \exp\left(-\frac{\sum_{c \in C} V_c(\omega)}{T}\right) \end{aligned} \tag{14.6}$$

If the temperature is very high, the distribution function is rather flat (see Fig. 14.3). Moving through the configuration space by iterative maximization is easy, since energy differences between configurations are small. Determining the

Fig. 14.3 The energy function is rather flat at high temperatures T. With decreasing temperature the minima get more pronounced

global maximum is difficult, however, since moving away from the maximum is just as easy.

The maxima get more pronounced with decreasing temperatures. Finding a local maximum becomes simpler, while moving out of a local maximum becomes less likely. When the temperature approaches $T = 0$, only the local maxima have non-zero probability.

The annealing process is simulated by gradually decreasing the temperature. At each temperature level, the process is allowed to search for the optimal configuration for some time. The search is stochastic. Randomly chosen new configurations are accepted unconditionally if the total energy decreases. The probability of accepting an energy increase depends on the current temperature and the ratio between energy before and after selecting the new configuration. Since the temperature remains constant while iterating through the configurations, the partition function is constant as well and cancels out when computing the ratio. Hence, it needs not to be computed. A sketch of the algorithm can be found in Fig. 14.4.

The authors of Geman and Geman (1984) showed that the algorithm produces the optimal solution regardless of the initial configuration, if the following annealing schedule is applied

$$T^{(n)} \geq \frac{m \cdot \Delta}{\ln(t+1)}, \tag{14.7}$$

where m is the number of sites and Δ is the energy difference between the configuration with highest energy and the one with lowest energy. Since m is usually large and Δ (if it is known at all) is large as well, the procedure converges very slowly. Hence, approximate schedules are often used such as Geman and Geman (1984)

```
set temperature T=T0
set initial configuration ω_cur
while T>ε do
    while sites have not been polled several times do
        select site i randomly from all sites
        compute perturbed version ω_new of the current estimate ω_cur
                    by selecting new site label L_new(i)≠L(i)
        compute change of of potential between ΔU=U(ω_cur)-U(ω_new)
        if ΔU>0 then
            ω_cur = ω_new
        else
            if exp(-ΔU/T) > random(0,1) then ω_cur = ω_new
    end_while
    reduce T according to cooling schedule
end_while
```

Fig. 14.4 Sketch of the annealing algorithm

$$T^{(n)} = \frac{C}{\ln(t+1)}, \tag{14.8}$$

where the parameter C is chosen based on domain knowledge, or (Kirkpatrick 1983)

$$T^{(n)} = \kappa T^{(t-1)} \tag{14.9}$$

with parameter κ chosen based on domain knowledge. Selection of C or κ depends on knowledge as to how close the initial configuration is to the optimal result.

14.1.3 Mean Field Annealing

Mean field annealing (Peterson and Soderberg 1989; Zhang 1992) uses a different approach to deal with the fact that computing the partition function is unreasonable. Instead of iterating through configurations by single state changes, it tracks the mean field

$$\langle \omega \rangle_T = \sum_{\omega} \omega_i P_T(\omega) = \sum_{\omega} \omega_i \frac{1}{\sum_{\omega} \exp(-U(T,\omega))} \exp\left(-\frac{U(\omega)}{T}\right). \tag{14.10}$$

The mean approaches the optimal solution when the temperature reaches 0. The advantage of this approach is that the mean reaches the optimal solution much faster than by going through stochastic changes at single sites.

For computing the a posteriori probability, the mean from neighboring sites is used to estimate the a priori probability. Equation (14.6) is replaced by

$$P(\mathbf{f} = \omega) \approx \prod_i \frac{1}{\sum_{\omega} \exp(-U^{\mathrm{MF}}(T,\omega))} \exp\left(-\frac{U^{\mathrm{MF}}(\omega_i)}{T}\right), \tag{14.11}$$

where U^{MF} is the mean field local energy from neighboring sites. It is assumed to be known.

Since the energy from interaction with neighboring sites is only an estimate based on the current configuration, mean field annealing has to be iterated in a similar fashion than simulated annealing. However, since the U^{MF} estimates are constant for the current iteration, all sites can be updated simultaneously.

Estimates from the mean interaction with neighboring sites are converging faster to the true means than updates from stochastic interaction. Hence, the temperature can be lowered faster than in simulated annealing. There is a heuristic component involved, however, since minima in the mean field are only equal to those in the original energy function if the temperature is zero.

14.1.4 Iterative Conditional Modes

Optimization is much faster using a strategy suggested by Besag (1986) which is called *iterative conditional modes* (*ICM*). ICM is a greedy strategy that only accepts improvements of the overall potential of a configuration. Hence, it requires a good initial guess for **f**. As such, ICM is similar to simulated annealing with $T = 0$. The initial guess is readily available, if a maximum likelihood estimate is generated from the distorted image without considering neighborhood dependencies of the MRF.

Optimal values for **f** are found by changing pixel values at one site f_i at a time. Given an image $\mathbf{f}^{(n)}$ at iteration n, the next image at iteration $(n + 1)$ is created by maximizing

$$\begin{aligned} P\left(f_i^{(n+1)} \middle| f_{S-i}^{(n)}, \mathbf{g}\right) &= p(\mathbf{f}|\mathbf{g}) \cdot P(f_i|f_{\mathrm{Nbs}}) \\ &= p(f_i|g_i) \cdot P(f_i|f_{\mathrm{Nbs}}) \\ &= \exp\left(-\frac{(f_i - g_i)^2}{2\sigma^2} - \beta \sum_{k=0}^{K-1} u_{ik}\right), \end{aligned} \tag{14.12}$$

where $f_{S-i}^{(k)}$ includes all pixels except f_i. Maximizing the probability means minimizing the sum of clique potentials

$$f_i^{(n+1)} = -\arg\min_{f_i} \left[\frac{\left(f_i^{(n)} - g_i\right)^2}{2\sigma^2} + \beta \sum_{k=0}^{K-1} u_{ik}^{(n)}\right]. \tag{14.13}$$

If the neighborhood influence is 0, it turns into a maximum likelihood estimator.

ICM computation can be parallelized, if pixels are grouped into sets of non-adjacent pixels.

14.2 Variational Calculus

The goal of variational calculus is to find a minimizing function for a functional (i.e., a function F of a function f). The strategy is similar to the minimization of a function $f(x)$. While for the latter a value x is searched, where $f(x + \Delta x) > f(x)$ for arbitrary changes Δx, in variational calculus a function $f(x)$ is searched of which arbitrarily small changes $\varepsilon \cdot \eta(x)$ cause an increase of the value of the functional F (see Fig. 14.5). The functional is a function of f and its derivatives. How such test functions are applied to solve a general variational problem can be found in textbooks on the calculus of variations, e.g., Weinstock (1974) or Gelfand and Fomin (2003).

Fig. 14.5 The goal of variational calculus is to find a function $f(x)$ that minimizes a functional of f. Such function is found when any variation by an added test function $\varepsilon\eta(x)$ increases the value of the functional

We will present the derivation of the solution of a simple variational problem of the kind

$$E(f) = \int_a^b F(f(x), f'(x), x)\mathrm{d}x, \tag{14.14}$$

where $f(x)$ is twice differentiable on the interval [a,b]. It covers the variational problems for methods presented in this book. The strategy in comparison with minimizing a function is depicted in Table 14.1.

In order to find a function $f(x)$ that minimizes (14.14), small variations $f(x) + \varepsilon \cdot \eta(x)$ are introduced. The function $\eta(x)$ is twice differentiable on $[a, b]$ with boundary conditions

$$\eta(a) = 0, \quad \eta(b) = 0 \tag{14.15}$$

Otherwise, $\eta(x)$ is arbitrary. Hence, a function is added to $f(x)$ such that $f(a) + \varepsilon\eta(a) = f(a)$ and $f(b) + \varepsilon\eta(b) = f(b)$ for arbitrary functions $\eta(x)$ and arbitrary values ε. The value of ε is now taken as variation parameter

$$J_{f(x)}(\varepsilon) = \int_a^b F(f(x) + \varepsilon\eta(x),\ f'(x) + \varepsilon\eta'(x),\ x)\mathrm{d}x \tag{14.16}$$

If $f(x)$ is a minimum, any $\varepsilon \neq 0$ will increase $J_{f(x)}(\varepsilon)$. Hence, a necessary condition for $f(x)$ that minimizes E is

$$\left.\frac{\mathrm{d}J_{f(x)}(\varepsilon)}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} = \int_a^b \frac{\mathrm{d}}{\mathrm{d}\varepsilon}[F(f(x) + \varepsilon\eta(x),\ f'(x) + \varepsilon\eta'(x),\ x)]\mathrm{d}x = 0 \tag{14.17}$$

Table 14.1 Comparison between the minimization of a function and that of a functional

Given	$f(x)$	$F(f(x), f'(x), x)$
Searched	x, for which $f()$ is minimal	$f(x)$, for which $_{f}F()$ is minimal
Solution	x with $f'(x) = 0$	$f(x)$ with $\frac{\mathrm{d}E}{\mathrm{d}f} = \frac{\mathrm{d}}{\mathrm{d}x}\frac{\partial F}{\partial f'} - \frac{\partial F}{\partial f} = 0$
Iteration	$x^{(n+1)} = x^{(n)} - f'(x^{(n)})$	$f^{(n+1)}(x) = f^{(n)}(x) - \mathrm{d}E/\mathrm{d}f$

Applying the chain rule (and dropping the arguments for clarity) results in

$$\left.\frac{\mathrm{d}J_{f(x)}(\varepsilon)}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} = \int_a^b \frac{\mathrm{d}F}{\mathrm{d}f}\eta + \frac{\mathrm{d}F}{\mathrm{d}f'}\eta' \mathrm{d}x = 0. \tag{14.18}$$

For using the Lagrangian symbolic of the solution, the following variable substitution is made

$$\delta(x) = \varepsilon\eta(x). \tag{14.19}$$

Multiplying (14.18) with ε on both sides and making the substitution result in

$$\delta J = \varepsilon \left.\frac{\mathrm{d}J_{f(x)}(\varepsilon)}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} = \int_a^b \frac{\partial F}{\partial f}\delta + \frac{\partial F}{\partial f'}\delta' \mathrm{d}x = 0. \tag{14.20}$$

Partial integration[1] of the second term in (14.18) delivers

$$\delta J = \left[\frac{\partial F}{\partial f'}\delta\right]_a^b + \int_a^b \left[\frac{\partial F}{\partial f} - \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial F}{\partial f'}\right)\right]\delta \ \mathrm{d}x = 0. \tag{14.21}$$

The first term vanishes, since the boundary conditions require $\delta(a) = \delta(b) = 0$ resulting in

$$\int_a^b \left[\frac{\partial F}{\partial f} - \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial F}{\partial f'}\right)\right]\delta \mathrm{d}x = 0 \tag{14.22}$$

for arbitrary functions δ. The fundamental lemma of calculus of variations states that

$$\int_a^b f(x) \cdot \delta(x) \mathrm{d}x = 0 \tag{14.23}$$

[1]Partial integration of an integral of the kind $_{\int a \ldots b}f'g$ uses the multiplication rule from differentiation to arrive at $_{\int a \ldots b}f'g = [fg]_{a \ldots b} - {}_{\int a \ldots b}fg'$. In the case above $f' := \delta'$ and $g := \partial F/\partial f'$.

for a continuous function f and arbitrary, twice differentiable functions δ obeying the boundary conditions $\delta(a) = \delta(b) = 0$. This implies $f(x) = 0$ on $[a, b]$.

The sketch of the proof is as follows: Since δ is arbitrary, it can be selected as $r(x) \cdot f(x)$ with $r(x)$ positive on $]a, b[$, twice differentiable and $r(a) = r(b) = 0$ (e.g., $r(x) = (x - a)(x - b)$). It results in

$$\int_a^b [f(x)]^2 \cdot r(x)\, \mathrm{d}x = 0. \tag{14.24}$$

Since $r > 0$ on $]\, a, b\, [$, it is easily seen that $f(x) = 0$ must hold on $]\, a, b\, [$ for (14.24) to hold. Since f is continuous, $f(x) = 0$ must hold on $[a, b]$ as well. Hence, (14.22) is only true if

$$\frac{\partial F}{\partial f} - \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial F}{\partial f'}\right) = 0 \tag{14.25}$$

This condition is called the *Euler–Lagrange equation* giving rise to the iterative solution scheme listed in Table 14.1.

14.3 Principal Component Analysis

Principal component analysis (PCA) is a way for linear decorrelation in feature space (Abdi et al. 2010). If many of the original features are correlated, the distribution of samples in feature space actually occupies a lower-dimensional subspace. The PCA produces an orthogonal transformation in feature space such that all covariance values between features are zero. Coordinate axes after transformation are aligned or orthogonal to this subspace (see Fig. 14.6). Features corresponding to axes orthogonal to the subspace can be identified and removed.

14.3.1 Computing the PCA

The covariance matrix $\mathbf{C}$ of the original feature space can be estimated from sample covariances

$$\begin{aligned} \mathbf{C} &= \begin{pmatrix} c_{11} & c_{21} & c_{N1} \\ c_{12} & & \\ c_{1N} & & c_{NN} \end{pmatrix}, \\ c_{ij} &\approx \frac{1}{K-1}\sum_{k=1}^{M}\left(f_{ik} - \bar{f}_i\right)\left(f_{jk} - \bar{f}_j\right), \quad \bar{f}_i \approx \frac{1}{K}\sum_{k=1}^{K} f_{ik}, \end{aligned} \tag{14.26}$$

Fig. 14.6 PCA produces an axis system that decorrelates the data. It is oriented along the data distribution in feature space. Features of this new coordinate system may be removed if projection on the remaining axes produces only a small error

where K is the number of samples available, and f_{ik} is the ith feature of the kth feature vector in the set of samples.

If features were (linearly) uncorrelated, all off-diagonal elements should be zero. If features are uncorrelated and occupy only a lower-dimensional subspace that is aligned with features axes, some of the variances in the diagonal should be zero as well. Any location of a sample in feature space can still be exactly represented if these features were removed.

Covariance between features usually exists. Hence, $\mathbf{C}$ will contain nonzero off-diagonal elements. The PCA will create a set of new features $\mathbf{f}'$ of which their elements are linear combinations of features of $\mathbf{f}$

$$f'_j = \sum_{i=1}^{N} f_i e_{ij} \tag{14.27}$$

so that the covariance matrix of the sample distribution $\mathbf{f}'$ no longer has nonzero off-diagonal elements. The beauty of the method is that a closed-form solution exists for computing weights e_{ij}, which we will describe below (see also Fig. 14.7).

Given an estimate of $\mathbf{C}$, a new orthogonal system of feature axes $f'_1, \ldots, f'_N$ will be computed with covariance matrix $\mathbf{C}'$

$$\mathbf{C}' = \begin{pmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \sigma_2^2 & \\ 0 & & \sigma_N^2 \end{pmatrix}, \tag{14.28}$$

with σ_i^2 being the variance of feature f_i. Computing eigenvalues λ_i and eigenvectors $\mathbf{e}_i = (e_{i1} \quad e_{i2} \ \ldots \ e_{iN})$ for $\mathbf{C}$ with

Fig. 14.7 The PCA solves an eigenproblem for the covariance matrix C. The eigenvectors e_i are the axes of the new coordinate system, and the eigenvalues λ_i are variances σ_i^2 along these axes (of which standard deviations σ_i are shown in this picture)

$$\mathbf{C}\mathbf{e}_i = \lambda_i \mathbf{e}_i \Rightarrow \mathbf{C}\mathbf{E} = \mathbf{E}\boldsymbol{\Lambda} \Leftrightarrow \mathbf{E}^{\mathrm{T}}\mathbf{C}\mathbf{E} = \boldsymbol{\Lambda},$$
$$\mathbf{E} = (\,\mathbf{e}_1 \quad \mathbf{e}_2 \quad \ldots \quad \mathbf{e}_N\,) = \begin{pmatrix} e_{11} & e_{21} & & e_{N1} \\ e_{12} & e_{22} & & \\ & & \ldots & \\ e_{1N} & & & e_{NN} \end{pmatrix} \tag{14.29}$$

produces the decomposition immediately because

$$\begin{aligned} \boldsymbol{\Lambda} &= \mathbf{E}^{\mathrm{T}}\mathbf{C}\mathbf{E} \approx \mathbf{E}^{\mathrm{T}}\left[\frac{1}{K}\sum_{k=1}^{K}\left(\mathbf{f}_k - \bar{\mathbf{f}}\right) \times \left(\mathbf{f}_k - \bar{\mathbf{f}}\right)^{\mathrm{T}}\right]\mathbf{E} \\ &= \frac{1}{K}\sum_{k=1}^{K}\left[\mathbf{E}^{\mathrm{T}}\left(\mathbf{f}_k - \bar{\mathbf{f}}\right)\right] \times \left[\left(\mathbf{f}_k - \bar{\mathbf{f}}\right)^{\mathrm{T}}\mathbf{E}\right] = \frac{1}{K}\sum_{k=1}^{K}\mathbf{f}'_k \times \left(\mathbf{f}'_k\right)^{\mathrm{T}} \end{aligned} \tag{14.30}$$

The matrix $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigenvalues of $\mathbf{C}$. They correspond to the feature variances of the covariance matrix $\mathbf{C}' = \boldsymbol{\Lambda}$ in a transformed system where a new feature f'_{ik} is computed by projecting the feature vector $\mathbf{f}_k$ on the ith eigenvector:

$$f'_{ik} = (\mathbf{e}_i)^{\mathrm{T}}\left(\mathbf{f}_k - \bar{\mathbf{f}}\right). \tag{14.31}$$

Feature reduction can now be carried out by investigating the variances, i.e., the eigenvalues of $\mathbf{C}$. A feature f'_i can be removed if its corresponding eigenvalue λ_i is zero. Eigenvalues close to zero indicate high linear correlation. Corresponding features may be removed as well, since the uncorrelated contribution of those features is often due to noise.

For determining which features are to be removed, features $\mathbf{f}'$ are ordered according to their variance. Accumulated variance $\sigma^2_{\text{accum}}(n) = \sum_{i=1}^{n} \sigma_i^2$ is used to determine the amount of feature reduction. A value of $n < N$ is chosen such that the percentage of $p_{\text{var}}(n) = \sigma^2_{\text{accum}}(n)/\sigma^2_{\text{accum}}(N)$ of the total variance exceeds some threshold (e.g., $p_{\text{var}}(n) > 0.95$ signifying that the first n features explain 95% of the variance in feature space).

14.3.2 Robust PCA

Principal component analysis is often applied when the number of samples is low and the dimension of the feature space is high. However, covariances in the matrix $\mathbf{C}$ are estimated from the samples. Its reliability depends on the size K of the sample set and the dimension of the feature vector. If the PCA is carried out, for instance, in 100-dimensional feature space using just 50 samples, the subspace spanned by the samples will be at most 50-dimensional. Covariance estimates can then be unreliable because of two reasons (as it has been already noted in Sect. 11.5):

- There is no redundant information about the variance. Hence, any influence from measurement noise in the feature values is interpreted as legal covariance. This is unwanted when reduced features shall describe some class-specific variation.
- Any outlier in the feature values directly influences the variance estimates of the data (see Fig. 14.8). This is unwanted, as the low probability of measuring an outlier value is not reflected in the covariance estimation.

Several ways exist for a robust PCA that attempt to solve these issues (Wang and Karhunen 1996; Li et al. 2002; Skočaj et al. 2002; Hubert et al. 2005) by assuming that the number of degrees of freedom, i.e., the number of dimensions needed to characterize class attributes by features, is substantially lower than the number of samples. It is furthermore assumed that outliers can be detected by measuring some distance from non-outlying feature vectors. The main characteristic of a covariance

Fig. 14.8 Error in the PCA due to an outlier in the samples

matrix, which is not influenced by outliers, is therefore a low but nonzero determinant. All of the methods work in an iterative fashion.

In the following, the robust PCA by Hubert et al. (2005) will be described which combines several strategies of previous attempts. The method is available as "robpca" in MATLAB. It proceeds in three stages.

In the *first stage*, the dimension of the data is reduced to the subspace that is spanned by the samples. If the number of samples is n and the original dimension of the data is d, the subspace is $d_0 \leq \min(n, d)$. In other words, if—what is often the case—the number of samples of n is lower than d, the dimension d_0 can be at most n (and may be lower, if some of the samples are linear dependent on each other). The result of this stage is the largest subspace in which variances can be estimated from the sample data.

In the *second stage*, the h least outlying samples are searched. In Hubert et al. (2005) it is shown that h must be larger than $[(n + d_0 + 1)/2]$ and low enough as to not include any outliers. In the absence of knowledge about the number of outliers and of d_0, the authors suggested to select

$$h = \max\left(\alpha n, \frac{n + d_{\max} + 1}{2}\right), \tag{14.32}$$

where $d_{\max}$ is the maximum number of principal components that will be selected as features and α is a control parameter with $\alpha =]0,1]$.

Selecting α and $d_{\max}$ depends on domain knowledge about the data. Hence, it is not advisable to use any default values from a given implementation. If $d_{\max}$ is too small, it may be that too much of the natural class-specific variance in the data is not accounted for, and if it is set too high, outliers may not be removed. The value of α controls this behavior as well. A high value overrides a too conservatively chosen maximum of dimensions, while a lower value will increase the robustness.

Given the number non-outlying samples h, outlyingness will be computed for each sample. This is essentially a distance measure between the sample in question and all other samples. The main component of the measure (details can be found in Hubert et al. (2005), see also Fig. 14.9) is a normalized sample distance to all lines through pairs of other samples. Since distances are computed with respect to other samples, they are independent on an external coordinate system. The outlyingness measure is a relative measure that rates this distance to the average normalized distance of all samples for all possible directions. Only the h samples with lowest outlyingness are kept for further processing. The first estimate of a robust covariance matrix $\mathbf{C}^{(0)}$ and a robust mean $\boldsymbol{\mu}^{(0)}$ is computed from these h samples.

In the *third stage*, $\mathbf{C}^{(0)}$ and $\boldsymbol{\mu}^{(0)}$ are used for computing a covariance matrix with smallest determinant. Two different, non-optimal methods for this computation are applied. The covariance matrix that has the lowest determinant of the two is chosen.

The first method iteratively computes Mahalanobis distances for each sample to the current mean using the current covariance matrix estimate. The h samples with the smallest distance are selected to compute the next estimate. The process is repeated until convergence. It may happen that during the process the determinant

Fig. 14.9 The outlyingness for some sample x_i is computed by comparing a distance measure to arbitrary directions v with distances of all other samples x_j to this direction v

becomes zero indicating that the h samples span a subspace of the d_0-dimensional space. In such case, d_0 is reduced to this subspace and the process is continued. The result of this first method is a covariance matrix $\mathbf{C}^{(1)}$ and the associated mean $\boldsymbol{\mu}^{(1)}$.

The second method starts with the number of dimensions $d_1 \leq d_0$ that is the result of the first method and repeatedly and randomly selects $(d_1 + 1)$ samples and computes a covariance and mean estimate from this subset. If the determinant is nonzero and smaller than the current estimate, the new covariance matrix and mean are kept. The result are estimates $\mathbf{C}^{(2)}$ and $\boldsymbol{\mu}^{(2)}$. If $\det(\mathbf{C}^{(2)}) < \det(\mathbf{C}^{(1)})$, then the current estimate $\mathbf{C}^{(3)}$ is set to $\mathbf{C}^{(2)}$. Otherwise $\mathbf{C}^{(1)}$ is selected as current estimate $\mathbf{C}^{(3)}$. The means are selected accordingly.

The covariance matrix $\mathbf{C}^{(3)}$ is then used to compute distances between each sample and the estimated mean $\boldsymbol{\mu}^{(3)}$. Samples are weighted by the distances. The final covariance matrix is then computed from the weighted samples. In Hubert et al. (2005), a hard rejection (weights are 0 or 1) is applied depending on this distance but soft weighting would be possible as well if domain knowledge indicates that outlier characterization by distance may not be reliable.

References

Abdi H, Williams LJ (2010) Principal component analysis. Wiley Interdiscip Rev Comput Stat 2 (4):433–459

Besag J (1986) On the statistical analysis of dirty pictures. J R Stat Soc Ser B (Methodol) 48 (3):259–302

Duda RO, Hart E, Stork DG (2001) Pattern classification, 2nd edn. Wiley Interscience

Gelfand IM, Fomin SV (2003) Calculus of variations. Dover Publishing

Geman S, Geman D (1984) Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Trans Pattern Anal Mach Intell 6(6):721–741

Hubert M, Rousseeuw PJ, Branden KV (2005) ROBPCA: a new approach to robust principal component analysis. Technometrics 47(1):64–79

Kirkpatrick S, Gelatt CD Jr, Vecchi MP (1983) Optimization by simulated annealing. Science 220 (4598):671–680

Li SZ (2009) Markov random field modeling in image analysis, 3rd edn. Springer

Li Y, Xu LQ, Morphett J, Jacobs R (2002) An integrated algorithm of incremental and robust PCA. In: Proceedings of IEEE international conference image processing ICIP 2003, vol I, pp 245–248

Peterson C, Soderberg B (1989) A new method for mapping optimization problems onto neural networks. Intl J Neural Syst 1(1):3–22

Skočaj D, Bischof H, Leonardis A (2002) A robust PCA algorithm for building representations from panoramic images. In: European conference computer vision—ECCV 2002, LNCS, vol 2353, pp 81–85

Wang L, Karhunen J (1996) A unified neural bigradient algorithm for robust PCA and MCA. Intl J Neural Syst 7(1):53–67

Weinstock R (1974) Calculus of variations. Dover Publishing

Zhang J (1992) The mean field theory in EM procedures for Markov random fields. IEEE Trans Signal Process 40(10):2570–2583

Erratum to: Guide to Medical Image Analysis

Klaus D. Toennies

Erratum to:
K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5

In the original version of the book, in copyright page, the sentence "The author(s) has/have asserted their right(s) to be identified as the author(s) of this work in accordance with the Copyright, Design and Patents Act 1988." is to be added. The erratum book has been updated with the change.

The updated original online version of this book frontmatter can be found at http://dx.doi.org/10.1007/978-1-4471-7320-5

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5_15

Index

0–9
0-level set, 323
3D slicer, 10

Symbols
α-β swap move, 286
α-expansion move, 286

A
AAM. *See* Active appearance model
Absolute connectivity, 297
Absorption edge, 31
Accuracy, 530
Active appearance model, 426
  combining shape and appearance, 426
Active contour, 313
  external energy, 314
  geodesic, 341
  internal energy, 314
  optimization, 314
  without edges, 345
Active geodesic regions, 348
Active shape model, 420
  alignment of landmarks, 423
  combination with FEM, 449
  de-correlation, 423
  Gaussian mixture model, 429
  landmark selection, 422
  modes of variation, 423
  non-parametric model, 430
  segmentation, 424
  training, 421
Active structural shape model, 429
AdaBoost, 513
Adaptive decision boundary, 486
  bias term, 486
  multiple class problem, 488
  nonlinear, 487
Adaptive linear discriminant function, 506
Advection force, 338
Agglomerative clustering, 508
Algebraic Reconstruction Technique (ART), 74
Analysis software, 9
Anatomical landmark, 368
Anger camera, 71
Angiography, 35
Anisotropic diffusion filtering, 152
Annealing schedule, 567
Anterior commissura, 368, 393
Aperture problem, 181
A posteriori probability, 251
  direct estimation, 482
Application entity, 106
A priori probability, 251
  Bayesian image restoration, 165
  estimation, 258
Artificial hardware phantom, 547
ASM. *See* Active shape model
Assemblage (FEM), 441
ASSM. *See* Active structural shape model
Association cost, 294
Association network, 267
Atlas-based segmentation, 453
Atlas mapping, 392
Attenuation
  SPECT, 74
  ultrasound imaging, 69
  X-ray, 32
Attenuation coefficient, 39
Attenuation correction, 77
Auto-encoder, 494
Average linkage, 509
Average pooling (CNN), 501

B
Backpropagation, 491
Backpropagation network. *See* Multilayer perceptron
Bagging, 512

© Springer-Verlag London Ltd. 2017

K.D. Toennies, *Guide to Medical Image Analysis*, Advances in Computer Vision and Pattern Recognition, DOI 10.1007/978-1-4471-7320-5

Bag of visual words, 518
Balloon model, 316
  inflation force, 317
Bayesian classifier, 480
Bayesian image restoration, 162
Bayesian theorem, 251, 563
Beam hardening, 30
Belief propagation, 463
Between-class scatter matrix, 478
Bias field correction, 349
Bilateral filtering, 157
  joint, 161
Binomial filter, 145
Blackening curve, 34
Blob detection, 185
Blood oxygen level dependency, 60
BOLD imaging, 60
Boundary condition
  essential, 442
  natural, 442
Boxcar filter, 144
BrainWeb, 548
Bremsstrahlung, 28
BRIEF descriptor, 191
Butterworth filter, 146

C

Canny edge operator, 175
CAT. *See* Computed tomography
Cathode ray tube, 28, 31
Central slice theorem, 41
Chemical shift, 57
Classification
  adaptive decision boundary, 486
  adaptive linear discriminant function, 506
  bagging, 512
  Bayesian classifier, 480
  Bayesian theorem, 251
  convolutional neural network, 498
  imbalanced data, 521
  kNN classifier, 484
  minimum distance classifier, 482
  multilayer perceptron, 488
  nearest neighbourhood classifier, 483
  support vector machine, 495
Clinical study, 6
Clinical workflow, 6
Clique, 564
Clique potential, 564
Clustering, 260, 508
  agglomerative, 508
  fuzzy c-means, 510
  interactive, 260
  k-means, 261
  mean shift, 264
  partitional, 261
  self-organising map, 267
CNN. *See* Convolutional neural network
Cohort study, 6
Competitive learning, 268
Complete linkage, 509
Composite object, 104
Composite service, 105
Composite SOP class, 106
Compton scatter, 30
Computational phantom, 546
Computed Tomography (CT), 38
Computer-aided detection, 7
Computer-aided diagnosis, 7
Computer-assisted surgery, 7
Confidence interval, 557
Confocal laser scanning microscopy, 84
Confusion matrix, 538
Connectome, 63
Contrast, 127
  GLCM, 130
  global, 128
  Michelson, 127
  root-mean-square (rms), 128
Contrast enhancement, 133
  histogram equalization, 134
  linear, 133
  windowing, 133
Contrast resolution, 126
Convolutional neural network, 498
  building block, 499
  structure, 499
  training, 501
  transfer learning, 504
Convolution backprojection, 43
Corner detector, 181
  FAST, 184
  Harris, 181
  SUSAN, 183
Correlation coefficient, 372, 412
Correspondence criterion, 363
  anatomical and geometrical landmarks, 364
  image-based features, 370
Cost of error, 521
Coupled level sets, 459
Covariance, 412
Covariance matrix, 257, 572
Cross-validation, 554
  n-fold, 553
CRT. *See* Cathode ray tube
CTA. *See* CT angiography

CT angiography, 46
CT artifacts
  metal artefact, 45
  motion, 45
  noise, 44
  partial volume effect, 45
  step-ladder-artefact, 45
  streak artifacts, 45
Curvature
  from divergence, 338

D
Damping matrix, 443
Data augmentation (CNN), 504
Data dictionary, 111
Data element (DICOM)
  type, 112
Decision boundary, 485
Decision tree, 514
  pruning, 514
  splitting, 514
  training, 514
Deep CNN, 498
Deep neural network, 498
Deformable curve, 313
Deformable model, 408
Deformable Shape Model (DSM), 432
Demon algorithm, 388
Dermoscopy, 82
Deterministic sign change criterion, 371
Diagnosis support, 7
Dice coefficient, 534
DICOM, 102
  application entity, 106
  compression, 121
  conformance statement, 109
  connectivity, 109
  file format, 111
  message service element, 106
  service class, 105
  service class provider, 106
  service class user, 106
  tag elements, 112
  viewer, 120
Diffeomorphism, 387
Difference of Gaussians, 185
Diffusion filtering, 151
Diffusion imaging, 62
Diffusion tensor, 153
Diffusion Tensor Imaging (DTI), 62
Digitally Reconstructed Radiograph (DRR), 384
Digital mammography, 38
Digital Subtraction Angiography (DSA), 36
DIMSE. *See* DICOM message service element
Discriminant function
  minimum square error, 508
Display
  ACR recommendations, 117
Divergence, 338
DoG. *See* Difference of Gaussians
Domain knowledge, 8, 19
  in MRF, 163
  in segmentation, 210
  medial axis representation, 420
  representation, 220
  segmentation, 218
  training, 221
  variability, 221
Doppler imaging, 67
Dynamic level set, 324
  evolution, 326
  stopping criterion, 341
  upwind scheme, 336

E
Edge detection
  Canny, 175
  multi-resolution, 177
Edge enhancement, 139
Edge-preserving smoothing
  Bayesian image restoration, 162
  bilateral filtering, 157
  diffusion filtering, 151
  gradient adaptive smoothing, 156
  guided filtering, 160
Edge tracking, 174
  multi-resolution, 178
Efficiency, 531
  interactive segmentation, 537
Elasticity, 314, 443
  modulus, 443
Elasticity matrix, 443
Elastic registration, 388
Electroencephalogram (EEG), 87
Electromagnetic wave, 26
Electron microscopy, 85
Electron volt, 26
Element matrix, 441
  assemblage, 441
Endianness, 114
Entropy, 129
Epidemiology, 6
Essential boundary condition, 442
Eulerian solution, 329
Euler-Lagrange equation, 572
  active contour, 314
  variational level sets, 343

Evaluation, 8, 19
Expansion move, 286
  graph, 288
Expectation maximisation algorithm, 255
Explicit model, 408
External energy, 314
Extrinsic marker, 367

F
False negative, 538
False positive, 538
FAST corner detector, 184
Fast marching algorithm, 334
Fault detection, 531
FBP. *See* Filtered backprojection
Feature
  linear decorrelation, 475
Feature reduction, 475, 574
  auto-encoder, 494
  by principal component analysis, 476
  interactive, 476
Feature similarity, 364
Feature space
  active shape model, 421
  classification, 474
  in registration, 363
Feature vector
  multi-dimensional, 256
FEM. *See* Finite element model
F-factor, 27
Fiber tracking (DTI), 63
Fiducial marker, 377, 542
Fiducial registration error, 377
Field II ultrasound simulation, 549
Field of View (FOV), 43
Figure of Merit (FOM), 530
Filtered backprojection, 41
Finite element model, 439
  acceleration, 443
  assemblage, 441
  combination with ASM, 449
  dynamic, 442
  external force, 445
  modal warping, 450
  searching model instances, 451
  shape function, 440
  static, 440
  velocity, 443
Fisher's discriminant analysis. *See* Linear discriminant analysis
Flat panel detector, 34
Floyd-Fulkerson algorithm, 282
Fluorescence microscopy, 84
Fluoroscopy, 35
fMRI. *See* Functional MRI
Focal spot (CRT), 32
Focused ultrasound, 68
Foreground segmentation, 210
Free-form deformation, 390
Free Induction Decay (FID), 50
Free vibration mode, 447
Froebenius norm, 187
F score, 539
Functional MRI, 59
FUS. *See* Focused ultrasound
Fuzzy c-means clustering, 510
Fuzzy connectedness, 296
  connectivity, 297

G
Gábor filter, 142
Gamma camera, 71
  collimator, 72
Gamma ray, 27
Gantry
  CT, 40
  MRI, 54
Gaussian filter, 145
Gaussianity, 480
Gaussian mixture model, 254, 289
  ASM, 429
  expectation maximisation algorithm, 255
Generalised cylinder, 417
Generalised Hough Transform (GHT), 413
Geometrical landmark, 368
Ghosting, 58
Gibbs distribution, 565
Gist, 203
GLCM. *See* Grey-level co-occurrence matrix
Global contrast, 128
Grab cut, 290
Gradient adaptive smoothing, 156
Gradient echo imaging, 56
Gradient histogram
  HOG, 201
  SIFT, 189
Gradient magnetic fields, 52
Gradient Vector Flow (GVF), 317
Graph cut, 276, 277
  a priori knowledge, 282
  connecting extrusions, 292
  data cost, 285
  expansion move, 286
  initialization, 285
  interaction cost, 285
  LOGISMOS, 461
  MRF optimisation, 284
  normalized, 293

shape model, 460
shape prior, 291, 460
sink, 278
source, 278
spatial connectivity, 292
swap move, 286
weights, 279
Graphical model, 463
Gray-level co-occurrence matrix, 130, 200
Ground truth, 542
from human expert, 543
from phantoms, 546
from real data, 543
STAPLE, 544
GrowCut, 304
Guided filter, 161
Gyromagnetic constant, 49

H

Hamming distance, 191
Hamming filter, 43, 147
Hann filter, 147
Harris corner detector, 181
Harris matrix, 182
Hausdorff distance, 535
quantile, 535
Head-hat registration, 381
Heaviside function, 345
Hessian matrix, 142
determinant, 186
Hidden layer, 490
Hierarchical FEM, 452
Hierarchical segmentation, 211
Hierarchical watershed transform, 235
Histogram, 129
Histogram equalization, 134
adaptive, 136
contrast limited adaptive, 136
Histogram of gradients, 201
Hit-or-miss operator, 418
HL7, 100
reference information model, 101
HOG. *See* Histogram of gradients
Homogeneous diffusion filtering, 151
Hospital Information System (HIS), 97
Hough transform
accumulator cell, 179
circles, 413
generalised, 413
straight lines, 179
Hounsfield Unit (HU), 44

I

ICP algorithm, 382
Ideal low pass filter, 145
IDL, 10
IFT. *See* Image foresting transform
Image-based features, 370
Image foresting transform, 276, 298
node load, 298
path cost, 299
watershed transform, 300
Image intensifier, 33
pincushion distortion, 34
S-distortion, 34
vignetting, 33
Imbalanced data, 521
Implicit model, 408, 415
Incompatibility
DICOM, 110
HL7, 102
Independent Component Analysis (HCA), 478, 479
Information object, 105
Information object description, 105
Inhomogeneous diffusion filtering, 152
Intelligent scissors, 240
Intensity gradient, 139, 175
in speed functions, 340
Interaction
confirmation, 223
correction, 223
feedback, 223
guidance, 223
guidelines, 222
parameterisation, 223
Interactive delineation, 225
Interactive graph cut, 284
Internal energy, 314
elasticity, 314
stiffness, 314
Inter-observer variability, 544
Intra-observer variability, 543
Intravenous ultrasound, 67
Intrinsic landmark, 368
Inversion recovery sequence, 56
IOD. *See* Information object description
Ionising radiation, 27
Ising model, 258
Iterative closest point algorithm, 382
Iterative Conditional Modes (ICM), 569
ITK, 13
IVUS. *See* Intravenous ultrasound

J

Jackknife technique, 552
Joint bilateral filter, 161
Joint entropy, 373

K
Kernel density estimator, 252, 481
Kernel function
  support vector machine, 497
Kernel trick, 497
Key point
  SIFT, 188
  SURF, 190
K-means clustering, 261
  diversity criterion, 263
  initialisation, 263
K-nearest-neighbourhood (KNN) classifier, 484
  active and passive samples, 484
  sample normalisation, 485
Kohonen network, 267
  clustering, 270
  neighbourhood activation, 269
  training, 268
K-space imaging, 53
Kurtosis excess, 480

L
Lagrangian solution, 329
Laplace operator, 140
Laplacian of Gaussian, 142, 185
Larmor frequency, 49
LDA. *See* Linear discriminant analysis
Leaving-one-out technique, 552
Level (display), 44
Level set, 321
  avoiding re-initialization, 351
  bias field, 348
  bias field correction, 349
  inhomogeneous background, 350
  initialization, 353
  intensity variation, 348
  interface, 323
  multi-phase, 346
  multiple segments, 346
  noise removal, 352
  topologically constrained, 354
  variational, 343
Level set evolution
  computation, 329
  step size, 331
  upwind scheme, 330
Level set function, 321
Level set initialization
  by clustering, 353
  region of interest, 354
Level set segmentation
  pose prior, 458
  shape model, 455
  Willmore flow, 457
Light box, 115
Likelihood function, 251
  estimation, 252
  partial volume effect, 253
Linear discriminant analysis, 477
Linear discriminant function, 506
Linear support vector machine, 495
Line of response, 79
Line Pairs per Millimeter (LPMM), 127
Live lane, 244
Live wire, 240
  cost function, 241
  3d, 244
  noise, 243
  optimality criterion, 241
Local affinity, 296
Local binary fitting energy, 350
Local Binary Pattern (LBP), 200
Local shape context, 197
LoG. *See* Laplacian of Gaussian
LOGISMOS algorithm, 461
Longitudinal relaxation, 50
Loopy belief propagation, 464
Lorentzian error, 460

M
Magnetic resonance, 48
  diffusion imaging, 62
  imaging techniques, 52
  longitudinal relaxation, 50
  perfusion imaging, 61
  proton density, 50
  T2* effect, 51
  transverse relaxation time, 51
Magnetic resonance imaging, 48
  data acquisition times, 56
  echoplanar imaging (EPI), 57
  frequency encoding, 52
  RARE sequence, 57
  readout gradient, 52
  slice selection, 52
Magnetoencephalogram (MEG), 87
Mammography, 37
  digital, 38
Marginal probability, 251
Marker-based Watershed Transform (mWST), 236
Marker particle solution, 329
Markov random field, 162, 563
  graph cut optimisation, 284
  neighbourhood, 564

Marr-Hildreth filter, 142
Mass matrix, 443
Mass spring model, 434
  external force, 435
  internal force, 436
  node coordinate system, 437
Matching, 362
MATLAB, 10
Maximally Stable Extremal Regions (MSER), 195
Maximum a Posteriori Expectation Maximization (MAP-EM), 77
Maximum flow, 282
Maximum Likelihood Expectation Maximization (MLEM) reconstruction , 75
Maxpooling (CNN), 501
MCAT heart phantom, 549
Mean field annealing, 568
Mean filter, 144
Mean shift clustering, 264
  mean shift, 266
  mode, 264
Mean square distance, 411
Mean transit time, 61
Medial axis, 417
  representation, 419
  scale space, 419
  transform, 418
Median filter, 148
  adaptive filter size, 150
  artefacts, 149
  weighted, 150
Medical workstation, 115
  software, 118
Mercer's theorem, 497
MevisLab, 9
Mexican hat filter, 142
Michelson contrast, 127
Microscopy, 83, 85
  electron, 85
  fluorescence, 84
Mincut maxflow, 282
Minimum cost graph cut, 278
Minimum distance classifier, 482
Mixture of Gaussians, 254
MLP. *See* Multilayer perceptron
Model, 406
  explicit representation, 312
  implicit representation, 312
  instance, 406
Model-driven segmentation
  appearance model, 426
  by active contours, 311
  shape model, 407
Modes of variation, 423
Modulation Transfer Function (MTF), 131
Moving image, 362
MRA. *See* MR angiography
MR angiography, 58
  flow void, 59
  gadolinium-enhanced, 58
  phase-contrast, 59
MR artefact
  chemical shift, 57
  ghosting, 58
  noise, 58
  partial volume effect, 58
  shading, 58
M-rep, 419
MRF. *See* Markov random field
MRI. *See* Magnetic resonance imaging
MR tractography, 63
Mulitlayer perceptron
  momentum term, 493
Multi-atlas segmentation, 454
Multilayer perceptron, 488
  auto-encoder, 494
  for classification, 491
  learning rate, 492
  overadaptation, 493
Multilayer segmentation, 212
Multi-phase level set, 346
Multiple Instance Learning (MIL), 516
Multi-resolution edge detection, 177
Multi-resolution edge tracking, 178
Multi-resolution MRF, 259
Multi-scale registration, 391
Mumford-Shah functional, 343
Mutual information, 373

N

Narrow band method, 337
Natural boundary condition, 442
NCAT phantom, 549
NCut algorithm, 295
Nearest neighbourhood classifier, 483
Near-field scanning optical microscopy, 85
Neighbourhood similarity, 390
Neighbour link, 278
Newmark-β algorithm, 445
  computation, 447
N-fold cross validation, 553
N-link, 278
Noise, 131
  reduction, 144
Nonlinear decision boundary, 487
  support vector machine, 497

Non-rigid registration, 387
demon algorithm, 388
diffusion model, 388
discontinuities, 397
initialisation, 390
smoothness constraint, 387
viscous fluid model, 388
Normalization, 362, 392
ROI-based, 394
Normalized graph cut, 293
approximation, 294
association cost, 294
Normalized histogram, 129
Normalized object, 104
Normalized service, 105
Normalized SOP class, 106
Nuclear imaging, 71
Null hypothesis, 556

O
OCT. *See* Optical coherence tomography
Octave, 11
One-sided t-test, 558
OpenCV, 13
Optical coherence tomography, 81
Optical microscopy, 83
Ordered Subset Expectation Maximization (OSEM), 77
OSI, 100
OSL algorithm, 77
Otsu's method, 227
Outlier detection, 554
paired landmarks, 379
PCA, 575
unpaired landmarks, 379
Outlyingness, 576
Overlap measure, 533
Over-segmentation, 212, 235, 533

P
Paired landmarks, 376
Paired t-test, 558
Pair production, 31
Part-based model, 420, 429, 452
graphical model, 464
Partial Volume Effect (PVE), 45
Partitional clustering, 261
Partition function, 565
Parzen window, 252
Pattern intensity criterion, 385
PCA. *See* Principal component analysis
Peak SNR, 132
Perceptron, 488
Perfusion imaging, 61
PET. *See* Positron emission tomography
Phantom, 546
Photobleaching, 85
Photoelectric absorption, 30
Photon, 26
Physical phantom, 546
Picture Archiving and Communication System (PACS), 99
Pincushion distortion, 34
Point Distribution Model (PDM), 420
Poisson ratio, 443
Pooling layer (CNN), 499
Positron emission tomography, 79
artifacts, 80
Posterior commissura, 368, 393
Precision rate, 539
Primary landmarks, 422
Principal axis, 379
Principal component analysis, 476, 572
feature reduction, 574
for registration, 378
Probabilistic Talairach coordinates, 393
Procrustes analysis, 422
generalized, 423
Procrustes distance, 367
Proton density, 50
Pull mode, 108
Push mode, 107
P-value, 556

Q
Quadric, 415
Quality
delineation task, 532
detection task, 537
registration task, 541
Quality of Fit (QOF), 407
Quantile Hausdorff distance, 535
Quaternion, 378

R
R (programming language), 524
Radiation
characteristic, 28
excitation, 28
monochrome, 28
polychrome, 28
Radiation absorbed dose, 27
Radiation exposure, 27
Radiology Information System (RIS), 97
Radiomics, 7
Radon transform, 39

Raleigh damping, 443, 448
Random forest, 516
Random walk, 277, 300
  algorithm, 303
  computation, 302
  for segmentation, 301
  probability function, 301
Rayleigh scatter, 30
Recall rate, 538
Receiver operator characteristic, 540
Receptive field (CNN), 499
Reference image, 362
Reference information model, 101
Region Adjacency Graph (RAG), 232
Region growing, 237
  homogeneity, 238
  leaking, 240
  seeded, 238
  symmetric, 239
  two pass, 238
Region merging, 232
Registration, 362
  components, 362
  digitally reconstructed radiograph, 384
  features, 364
  head-hat, 381
  multiscale, 391
  pattern intensity criterion, 385
  projection Image, 383
  with quaternions, 378
  with segmentation, 395
Reinforcement learning, 268
Relative cerebral blood flow, 61
Relative cerebral blood volume, 61
Relative connectivity, 298
Relaxation labelling, 230
Reliability, 530
Repeatability
  interactive segmentation, 536
Resolution enhancement, 136
Retina photography, 82
Rigid registration, 375
  from paired landmarks, 376
  in frequency domain, 380
Ringing artefact, 145
Robustness, 530, 555
Robust PCA, 575
ROC curve, 540
  area under the ROC curve, 540
Rotation
  matrix, 376
  with quaternions, 378

S
Saliency, 202
Scale-invariant feature transform. *See* SIFT
Scale space, 409
Scanning electron microscopy, 86
Scintigraphy, 71
S-distortion, 34
Secondary landmarks, 422
Seeded region growing, 238
Segmentation
  active contour, 313
  by classification, 251
  clustering, 260
  data knowledge, 213
  deep CNN, 505
  domain knowledge, 210, 218
  finite element model, 439
  fuzzy connectedness, 296
  Gaussian pyramid, 215
  geodesic active contours, 341
  graph cut, 284
  image foresting transform, 298
  influence from noise, 215
  Interactive, 225
  interactive graph cut, 278
  kinds of interaction, 222
  level sets, 338
  live wire, 240
  Mass spring model, 434
  normalized graph cut, 293
  random walks, 300
  region growing, 237
  region merging, 232
  shading, 216
  split and merge, 233
  texture, 217
  thresholding, 226
  variational level set, 343
  watershed transform, 234
  with active shape models, 424
  with shape prior, 453
Self-organising map, 267
  clustering, 270
  neighbourhood activation, 269
  training, 268
SEM. *See* Scanning electron microscopy
Sensitivity, 538
Service Class Provider (SCP), 106
Service Class User (SCU), 106
Service Object Pair (SOP), 106
Shading, 58
Shadow groups, 112

Shape-based interpolation, 137
Shape context, 197
Shape decomposition, 409
Shape descriptor, 417
Shape detection, 407
Shape deviation, 407
Shape feature vector, 420
Shape function, 440
Shape hierarchy, 410
Shape matching
  dummy points, 198
  with local shape context, 198
Shape model, 406
  graph cut segmentation, 460
  level set segmentation, 455
Shape prior, 291, 453
Shape representation, 408
SIFT, 188
  feature computation, 189
  key point generation, 188
  key point reduction, 189
  matching, 190
Sigmoid function, 491
Signal-to-Noise Ratio (SNR), 131
Signed distance function
  level sets, 324
Significance, 555
Simple linear iterative clustering, 193
Simple point, 355
Simulated annealing, 166, 566
  annealing schedules, 567
  optimal solution, 567
Single linkage, 509
Single Photon Emission Computed Tomography (SPECT), 78
Skeleton. *See* Medial axis
Slack variable, 498
Slice interpolation, 137
SLIC superpixel, 193
Smoothness constraint, 387
SMOTE, 521
Snake, 313
Sobel operator, 140
Softmax pooling (CNN), 501
Software phantom, 548
SOM. *See* Self-organising map
Sonography. *See* Ultrasound imaging
Spatial resolution, 126
Specificity, 538
Speckle artefact, 69
Speeded-up Robust Features (SURF), 190
Speed function, 328, 337
  advection force, 338
  curvature, 338
  data knowledge, 340
  intensity gradient, 340
Spin (of an atom), 48
Spin echo sequence, 55
Spin-lattice-relaxation time, 51
Spin precession, 50
Spin-spin-relaxation time, 51
Spiral scanner, 40
Split and merge, 233
Spring force, 436
SSM. *See* Statistical shape model
Staircase effect, 155
STAPLE, 544
  sensitivity and specificity estimation, 544
Stationary level set, 324
  arrival time, 333
  evolution, 325
  upwind scheme, 333
Statistical shape model, 420
Stereotaxic coordinate system, 393
Stiffness, 314
Stiffness matrix, 441
Stochastic sign change criterion, 371
Stopping criterion
  dynamic level set, 341
  stationary level set, 336
Stress matrix, 443
Stride (CNN), 500
Student's t-test, 556
Subdiffraction confocal laser scanning microscopy, 85
Super-ellipsoid, 415
  with free-form deformation, 416
Superpixel, 193, 290
Support vector, 496
Support vector machine, 495
  linear, 495
  nonlinear decision boundary, 497
  radial base functions, 498
  slack variables, 498
SUSAN corner detector, 183
SVM. *See* Support vector machine
Swap move, 286
  graph, 287
Symmetric region growing, 239

T

Talairach coordinates, 393
Tanimoto coefficient, 535
T-distribution, 558
T2* effect, 51
TEM. *See* Transmission electron microscopy
Template matching, 411
Terminal link, 278

Tertiary landmarks, 422
Texture, 198
  Haralick's features, 200
  Law's filter masks, 200
  local binary pattern, 200
  spectral features, 200
Thinning algorithm, 418
Thresholding, 226
  connected component analysis, 227
  Otsu's method, 227
  relaxation labelling, 230
  shading correction, 230
  using fuzzy memberships, 229
  Zack's algorithm, 229
Time lapse microscopy, 85
T-link, 278
  weight adaptation, 289
Topologically constrained level sets, 354
Torsion force, 437
Transfer function, 133
Transfer learning (CNN), 504
Transmission electron microscopy, 86
Transrectal ultrasound, 67
Transverse relaxation, 51
Treatment planning, 7
True negative, 538
True positive, 537
TRUS. *See* Transrectal ultrasound
T-snake, 319
  evolution, 319
T-surface, 320
T-test, 556
T1-time, 51
T2-time, 51
Turbo spin echo sequence, 57
Two-sided t-test, 558

U
Ultrasound artifacts, 69
Ultrasound imaging, 66
  A-scan, 67
  B-scan, 67
Under-segmentation, 533
Unique identifier (UID), 105
Upwind scheme, 330

V
Valence electron, 28
Validation
  documentation, 530
  interactive segmentation, 536
  quality measures, 531
  robustness, 555
  sources of variation, 553
Variational calculus, 569
Variational level sets, 343
  active contours without regions, 345
  active geodesic regions, 348
  Euler-Lagrange equation, 343
  geodesic active contour, 342
Vesselness filter, 186
Vibration mode, 447
  object-specific symmetry, 451
Viewer software, 9
Vignetting, 33
Viscous fluid model, 388
Visible human, 546

W
Watershed transform, 234
  by image foresting transform, 300
  flooding, 235
  hierarchical, 235
  marker-based, 236
  over-segmentation, 235
Wave propagation, 324
WEKA, 524
Welch-test, 559
Window (display), 44
Within-class scatter matrix, 477
WST. *See* Watershed transform

X
XCAT phantom, 550
X-ray, 26
  attenuation, 30
  contrast, 31
  generation, 28
  imaging, 31
  tube, 29

Y
Young's modulus, 444

Z
Zack's algorithm, 229
Z message (HL7), 102