

Wilhelm Burger · Mark J. Burge

Digital Image Processing

An algorithmic introduction using Java

Second Edition

2016

Springer

Berlin Heidelberg New York

HongKong London

Milano Paris Tokyo

Preface

This book provides a modern, self-contained introduction to digital image processing. We designed the book to be used both by learners desiring a firm foundation on which to build as well as practitioners in search of detailed analysis and transparent implementations of the most important techniques. This is the second English edition of the original German-language book, which has been widely used by:

- Scientists and engineers who use image processing as a tool and wish to develop a deeper understanding and create custom solutions to imaging problems in their field.
- IT professionals wanting a self-study course featuring easily adaptable code and completely worked out examples, enabling them to be productive right away.
- Faculty and students desiring an example-rich introductory textbook suitable for an advanced undergraduate or graduate level course that features exercises, projects, and examples that have been honed during our years of experience teaching this material.

While we concentrate on practical applications and concrete implementations, we do so without glossing over the important formal details and mathematics necessary for a deeper understanding of the algorithms. In preparing this text, we started from the premise that simply creating a recipe book of imaging solutions would not provide the deeper understanding needed to apply these techniques to novel problems, so instead our solutions are developed stepwise from three different perspectives: in mathematical form, as abstract pseudocode algorithms, and as complete Java programs. We use a common notation to intertwine all three perspectives—providing multiple, but linked, views of the problem and its solution.

Prerequisites

Instead of presenting digital image processing as a mathematical discipline, or strictly as signal processing, we present it from a practitioner's and programmer's perspective and with a view toward replacing many of the formalisms commonly used in other texts with constructs more readily understandable by our audience. To take full advantage of the *programming* components of this book, a knowledge of basic data structures and object-oriented programming, ideally in Java, is required. We selected Java for a number of reasons: it is the first programming language learned by students in a wide variety of engineering curricula, and professionals with knowledge of a related language, especially C# or C++, will find the programming examples easy to follow and extend.

The software in this book is designed to work with ImageJ, a widely used, programmer-extensible, imaging system developed, maintained, and distributed by the National Institutes of Health (NIH).¹ ImageJ is implemented completely in Java, and therefore runs on all major platforms, and is widely used because its “plugin”-based architecture enables it to be easily extended. While all examples run in ImageJ, they have been specifically designed to be easily ported to other environments and programming languages.

Use in research and development

This book has been especially designed for use as a textbook and as such features exercises and carefully constructed examples that supplement our detailed presentation of the fundamental concepts and techniques. As both practitioners and developers, we know that the details required to successfully understand, apply, and extend classical techniques are often difficult to find, and for this reason we have been very careful to provide the missing details, many gleaned over years of practical application. While this should make the text particularly valuable to those in research and development, it is not designed as a comprehensive, fully-cited scientific research text. On the contrary, we have carefully vetted our citations so that they can be obtained from easily accessible sources. While we have only briefly discussed the fundamentals of, or entirely omitted, topics such as hierarchical methods, wavelets, or eigenimages because of space limitations, other topics have been left out deliberately, including advanced issues such as object recognition, image understanding, and three-dimensional (3D) computer vision. So, while most techniques described in this book could be called “blind and dumb”, it is our experience that straightforward, technically clean implementations of these simpler methods are essential to the success of any further domain-specific, or even “intelligent”, approaches.

If you are only in search of a programming handbook for ImageJ or Java, there are certainly better sources. While the book includes many code examples, programming in and of itself is not our main focus. Instead Java serves as just one important element for describing each technique in a precise and immediately testable way.

Classroom use

Whether it is called signal processing, image processing, or media computation, the manipulation of digital images has been an integral part of most computer science and engineering curricula for many years. Today, with the omnipresence of all-digital work flows, it has become an integral part of the required skill set for professionals in many diverse disciplines.

Today the topic has migrated into the early stages of many curricula, where it is often a key foundation course. This migration uncovered a problem in that many of the texts relied on as standards

¹ <http://rsb.info.nih.gov/ij/>.

in the older graduate-level courses were not appropriate for beginners. The texts were usually too formal for novices, and at the same time did not provide detailed coverage of many of the most popular methods used in actual practice. The result was that educators had a difficult time selecting a single textbook or even finding a compact collection of literature to recommend to their students. Faced with this dilemma ourselves, we wrote this book in the sincere hope of filling this gap.

The contents of the following chapters can be presented in either a one- or two-semester sequence. Where feasible, we have added supporting material in order to make each chapter as independent as possible, providing the instructor with maximum flexibility when designing the course. Chapters 18–20 offer a complete introduction to the fundamental spectral techniques used in image processing and are essentially independent of the other material in the text. Depending on the goals of the instructor and the curriculum, they can be covered in as much detail as required or completely omitted. The following road map shows a possible partitioning of topics for a two-semester syllabus.

| Road Map for a 1/2-Semester Syllabus | Sem. | 1 | 2 |
|---|------|---|---|
| 1. Digital Images | ■ | □ | |
| 2. ImageJ | ■ | □ | |
| 3. Histograms and Image Statistics | ■ | □ | |
| 4. Point Operations | ■ | □ | |
| 5. Filters | ■ | □ | |
| 6. Edges and Contours | ■ | □ | |
| 7. Corner Detection | □ | ■ | |
| 8. The Hough Transform: Finding Simple Curves | □ | ■ | |
| 9. Morphological Filters | ■ | □ | |
| 10. Regions in Binary Images | ■ | □ | |
| 11. Automatic Thresholding | □ | ■ | |
| 12. Color Images | ■ | □ | |
| 13. Color Quantization | □ | ■ | |
| 14. Colorimetric Color Spaces | □ | ■ | |
| 15. Filters for Color Images | □ | ■ | |
| 16. Edge Detection in Color Images | □ | ■ | |
| 17. Edge-Preserving Smoothing Filters | □ | ■ | |
| 18. Introduction to Spectral Techniques | □ | ■ | |
| 19. The Discrete Fourier Transform in 2D | □ | ■ | |
| 20. The Discrete Cosine Transform (DCT) | □ | ■ | |
| 21. Geometric Operations | ■ | □ | |
| 22. Pixel Interpolation | ■ | □ | |
| 23. Image Matching and Registration | ■ | □ | |
| 24. Non-Rigid Image Matching | □ | ■ | |
| 25. Scale-Invariant Local Features (SIFT) | □ | ■ | |
| 26. Fourier Shape Descriptors | □ | ■ | |

Addendum to the second edition

This second edition is based on our completely revised German third edition and contains both additional material and several new chap-

ters including: *automatic thresholding* (Ch. 11), *filters and edge detection for color images* (Chs. 15 and 16), *edge-preserving smoothing filters* (Ch. 17), *non-rigid image matching* (Ch. 24), and *Fourier shape descriptors* (Ch. 26). Much of this new material is presented for the first time at the level of detail necessary to completely understand and create a working implementation.

The two final chapters on SIFT and Fourier shape descriptors are particularly detailed to demonstrate the actual level of granularity and the special cases which must be considered when actually implementing complex techniques. Some other chapters have been rearranged or split into multiple parts for more clarity and easier use in teaching. The mathematical notation and programming examples were completely revised and almost all illustrations were adapted or created anew for this full-color edition.

For this edition, the *ImageJ Short Reference* and ancillary source code have been relocated from the Appendix and the most recently versions are freely available in electronic form from the book's website. The complete source code, consisting of the common `imagingbook` library, sample ImageJ plugins for each book chapter, and extended documentation are available from the book's SourceForge site.²

Online resources

Visit the website for this book

www.imagingbook.com

to download supplementary materials, including the complete Java source code for all examples and the underlying software library, full-size test images, useful references, and other supplements. Comments, questions, and corrections are welcome and may be addressed to

imagingbook@gmail.com

Exercises and solutions

Each chapter of this book contains a set of sample exercises, mainly for supporting instructors to prepare their own assignments. Most of these tasks are easy to solve after studying the corresponding chapter, while some others may require more elaborated reasoning or experimental work. We assume that scholars know best how to select and adapt individual assignments in order to fit the level and interest of their students. This is the main reason why we have abstained from publishing explicit solutions in the past. However, we are happy to answer any personal request if an exercise is unclear or seems to elude a simple solution.

Thank you!

This book would not have been possible without the understanding and support of our families. Our thanks go to Wayne Rasband at NIH for developing ImageJ and for his truly outstanding support of

² <http://sourceforge.net/projects/imagingbook/>.

the community and to all our readers of the previous editions who provided valuable input, suggestions for improvement, and encouragement. The use of open source software for such a project always carries an element of risk, since the long-term acceptance and continuity is difficult to assess. Retrospectively, choosing ImageJ as the software basis for this work was a good decision, and we would consider ourselves happy if our book has indirectly contributed to the success of the ImageJ project itself. Finally, we owe a debt of gratitude to the professionals at Springer, particularly to Wayne Wheeler and Simon Reeves who were responsible for the English edition.

Hagenberg / Washington D.C.
Fall 2015

Contents

| | | |
|----------|---|----|
| 1 | Digital Images | 1 |
| 1.1 | Programming with Images | 2 |
| 1.2 | Image Analysis and Computer Vision..... | 2 |
| 1.3 | Types of Digital Images | 4 |
| 1.4 | Image Acquisition..... | 4 |
| 1.4.1 | The Pinhole Camera Model | 4 |
| 1.4.2 | The “Thin” Lens | 6 |
| 1.4.3 | Going Digital | 7 |
| 1.4.4 | Image Size and Resolution | 8 |
| 1.4.5 | Image Coordinate System..... | 9 |
| 1.4.6 | Pixel Values..... | 9 |
| 1.5 | Image File Formats | 11 |
| 1.5.1 | Raster versus Vector Data | 12 |
| 1.5.2 | Tagged Image File Format (TIFF) | 12 |
| 1.5.3 | Graphics Interchange Format (GIF) | 13 |
| 1.5.4 | Portable Network Graphics (PNG) | 14 |
| 1.5.5 | JPEG | 14 |
| 1.5.6 | Windows Bitmap (BMP) | 18 |
| 1.5.7 | Portable Bitmap Format (PBM)..... | 18 |
| 1.5.8 | Additional File Formats | 18 |
| 1.5.9 | Bits and Bytes | 19 |
| 1.6 | Exercises | 21 |
| 2 | ImageJ | 23 |
| 2.1 | Software for Digital Imaging | 24 |
| 2.2 | ImageJ Overview | 24 |
| 2.2.1 | Key Features | 25 |
| 2.2.2 | Interactive Tools..... | 26 |
| 2.2.3 | ImageJ Plugins | 26 |
| 2.2.4 | A First Example: Inverting an Image..... | 28 |
| 2.2.5 | Plugin My_Inverter_A (using PlugInFilter) | 28 |
| 2.2.6 | Plugin My_Inverter_B (using PlugIn) | 29 |
| 2.2.7 | When to use PlugIn or PlugInFilter?..... | 30 |
| 2.2.8 | Executing ImageJ “Commands” | 32 |
| 2.3 | Additional Information on ImageJ and Java | 34 |
| 2.3.1 | Resources for ImageJ..... | 34 |
| 2.3.2 | Programming with Java | 34 |
| 2.4 | Exercises | 34 |

| | | |
|----------|---|----|
| 3 | Histograms and Image Statistics | 37 |
| 3.1 | What is a Histogram? | 38 |
| 3.2 | Interpreting Histograms | 39 |
| 3.2.1 | Image Acquisition | 39 |
| 3.2.2 | Image Defects | 41 |
| 3.3 | Calculating Histograms | 43 |
| 3.4 | Histograms of Images with More than 8 Bits | 45 |
| 3.4.1 | Binning | 45 |
| 3.4.2 | Example | 45 |
| 3.4.3 | Implementation | 46 |
| 3.5 | Histograms of Color Images | 46 |
| 3.5.1 | Intensity Histograms | 47 |
| 3.5.2 | Individual Color Channel Histograms | 47 |
| 3.5.3 | Combined Color Histograms | 48 |
| 3.6 | The Cumulative Histogram | 49 |
| 3.7 | Statistical Information from the Histogram | 49 |
| 3.7.1 | Mean and Variance | 50 |
| 3.7.2 | Median | 51 |
| 3.8 | Block Statistics | 51 |
| 3.8.1 | Integral Images | 51 |
| 3.8.2 | Mean Intensity | 53 |
| 3.8.3 | Variance | 53 |
| 3.8.4 | Practical Calculation of Integral Images | 53 |
| 3.9 | Exercises | 54 |
| 4 | Point Operations | 57 |
| 4.1 | Modifying Image Intensity | 58 |
| 4.1.1 | Contrast and Brightness | 58 |
| 4.1.2 | Limiting Values by Clamping | 58 |
| 4.1.3 | Inverting Images | 59 |
| 4.1.4 | Threshold Operation | 59 |
| 4.2 | Point Operations and Histograms | 59 |
| 4.3 | Automatic Contrast Adjustment | 61 |
| 4.4 | Modified Auto-Contrast Operation | 62 |
| 4.5 | Histogram Equalization | 63 |
| 4.6 | Histogram Specification | 66 |
| 4.6.1 | Frequencies and Probabilities | 67 |
| 4.6.2 | Principle of Histogram Specification | 67 |
| 4.6.3 | Adjusting to a Piecewise Linear Distribution | 68 |
| 4.6.4 | Adjusting to a Given Histogram (Histogram Matching) | 70 |
| 4.6.5 | Examples | 71 |
| 4.7 | Gamma Correction | 74 |
| 4.7.1 | Why Gamma? | 75 |
| 4.7.2 | Mathematical Definition | 77 |
| 4.7.3 | Real Gamma Values | 77 |
| 4.7.4 | Applications of Gamma Correction | 78 |
| 4.7.5 | Implementation | 79 |
| 4.7.6 | Modified Gamma Correction | 80 |
| 4.8 | Point Operations in ImageJ | 82 |
| 4.8.1 | Point Operations with Lookup Tables | 82 |
| 4.8.2 | Arithmetic Operations | 83 |

| | | |
|----------|---|------------|
| 4.8.3 | Point Operations Involving Multiple Images . | 83 |
| 4.8.4 | Methods for Point Operations on Two Images | 84 |
| 4.8.5 | ImageJ Plugins Involving Multiple Images . . | 85 |
| 4.9 | Exercises | 86 |
| 5 | Filters | 89 |
| 5.1 | What is a Filter? | 89 |
| 5.2 | Linear Filters | 91 |
| 5.2.1 | The Filter Kernel | 91 |
| 5.2.2 | Applying the Filter | 91 |
| 5.2.3 | Implementing the Filter Operation | 93 |
| 5.2.4 | Filter Plugin Examples | 93 |
| 5.2.5 | Integer Coefficients. | 95 |
| 5.2.6 | Filters of Arbitrary Size | 96 |
| 5.2.7 | Types of Linear Filters | 97 |
| 5.3 | Formal Properties of Linear Filters | 99 |
| 5.3.1 | Linear Convolution | 100 |
| 5.3.2 | Formal Properties of Linear Convolution | 101 |
| 5.3.3 | Separability of Linear Filters | 102 |
| 5.3.4 | Impulse Response of a Filter | 104 |
| 5.4 | Nonlinear Filters. | 105 |
| 5.4.1 | Minimum and Maximum Filters | 105 |
| 5.4.2 | Median Filter | 107 |
| 5.4.3 | Weighted Median Filter | 109 |
| 5.4.4 | Other Nonlinear Filters | 111 |
| 5.5 | Implementing Filters | 112 |
| 5.5.1 | Efficiency of Filter Programs | 112 |
| 5.5.2 | Handling Image Borders | 113 |
| 5.5.3 | Debugging Filter Programs | 114 |
| 5.6 | Filter Operations in ImageJ | 115 |
| 5.6.1 | Linear Filters | 115 |
| 5.6.2 | Gaussian Filters | 115 |
| 5.6.3 | Nonlinear Filters | 116 |
| 5.7 | Exercises | 116 |
| 6 | Edges and Contours | 121 |
| 6.1 | What Makes an Edge? | 121 |
| 6.2 | Gradient-Based Edge Detection | 122 |
| 6.2.1 | Partial Derivatives and the Gradient | 123 |
| 6.2.2 | Derivative Filters | 123 |
| 6.3 | Simple Edge Operators | 124 |
| 6.3.1 | Prewitt and Sobel Operators | 125 |
| 6.3.2 | Roberts Operator | 127 |
| 6.3.3 | Compass Operators | 128 |
| 6.3.4 | Edge Operators in ImageJ | 130 |
| 6.4 | Other Edge Operators | 130 |
| 6.4.1 | Edge Detection Based on Second Derivatives | 130 |
| 6.4.2 | Edges at Different Scales | 130 |
| 6.4.3 | From Edges to Contours | 131 |
| 6.5 | Canny Edge Operator | 132 |
| 6.5.1 | Pre-processing | 134 |
| 6.5.2 | Edge localization | 134 |

| | | |
|----------|---|------------|
| 6.5.3 | Edge tracing and hysteresis thresholding | 135 |
| 6.5.4 | Additional Information | 137 |
| 6.5.5 | Implementation | 138 |
| 6.6 | Edge Sharpening | 139 |
| 6.6.1 | Edge Sharpening with the Laplacian Filter . . | 139 |
| 6.6.2 | Unsharp Masking | 142 |
| 6.7 | Exercises | 146 |
| 7 | Corner Detection | 147 |
| 7.1 | Points of Interest | 147 |
| 7.2 | Harris Corner Detector | 148 |
| 7.2.1 | Local Structure Matrix | 148 |
| 7.2.2 | Corner Response Function (CRF) | 149 |
| 7.2.3 | Determining Corner Points | 149 |
| 7.2.4 | Examples | 150 |
| 7.3 | Implementation | 152 |
| 7.3.1 | Step 1: Calculating the Corner Response Function | 153 |
| 7.3.2 | Step 2: Selecting “Good” Corner Points | 155 |
| 7.3.3 | Step 3: Cleaning up | 156 |
| 7.3.4 | Summary | 157 |
| 7.4 | Exercises | 158 |
| 8 | Finding Simple Curves: The Hough Transform . . . | 161 |
| 8.1 | Salient Image Structures | 161 |
| 8.2 | The Hough Transform | 162 |
| 8.2.1 | Parameter Space | 163 |
| 8.2.2 | Accumulator Map | 164 |
| 8.2.3 | A Better Line Representation | 165 |
| 8.3 | Hough Algorithm | 167 |
| 8.3.1 | Processing the Accumulator Array | 168 |
| 8.3.2 | Hough Transform Extensions | 170 |
| 8.4 | Java Implementation | 173 |
| 8.5 | Hough Transform for Circles and Ellipses | 176 |
| 8.5.1 | Circles and Arcs | 176 |
| 8.5.2 | Ellipses | 177 |
| 8.6 | Exercises | 179 |
| 9 | Morphological Filters | 181 |
| 9.1 | Shrink and Let Grow | 182 |
| 9.1.1 | Neighborhood of Pixels | 183 |
| 9.2 | Basic Morphological Operations | 183 |
| 9.2.1 | The Structuring Element | 183 |
| 9.2.2 | Point Sets | 184 |
| 9.2.3 | Dilation | 185 |
| 9.2.4 | Erosion | 186 |
| 9.2.5 | Formal Properties of Dilation and Erosion . . | 186 |
| 9.2.6 | Designing Morphological Filters | 188 |
| 9.2.7 | Application Example: Outline | 189 |
| 9.3 | Composite Morphological Operations | 192 |
| 9.3.1 | Opening | 192 |
| 9.3.2 | Closing | 192 |

| | | |
|-----------|--|------------|
| 9.3.3 | Properties of Opening and Closing | 193 |
| 9.4 | Thinning (Skeletonization) | 194 |
| 9.4.1 | Thinning Algorithm by Zhang and Suen | 194 |
| 9.4.2 | Fast Thinning Algorithm | 195 |
| 9.4.3 | Java Implementation | 198 |
| 9.4.4 | Built-in Morphological Operations in ImageJ | 201 |
| 9.5 | Grayscale Morphology | 202 |
| 9.5.1 | Structuring Elements | 202 |
| 9.5.2 | Dilation and Erosion | 203 |
| 9.5.3 | Grayscale Opening and Closing | 203 |
| 9.6 | Exercises | 205 |
| 10 | Regions in Binary Images | 209 |
| 10.1 | Finding Connected Image Regions | 210 |
| 10.1.1 | Region Labeling by Flood Filling | 210 |
| 10.1.2 | Sequential Region Labeling | 213 |
| 10.1.3 | Region Labeling—Summary | 219 |
| 10.2 | Region Contours | 219 |
| 10.2.1 | External and Internal Contours | 219 |
| 10.2.2 | Combining Region Labeling and Contour Finding | 220 |
| 10.2.3 | Java Implementation | 222 |
| 10.3 | Representing Image Regions | 225 |
| 10.3.1 | Matrix Representation | 225 |
| 10.3.2 | Run Length Encoding | 225 |
| 10.3.3 | Chain Codes | 226 |
| 10.4 | Properties of Binary Regions | 229 |
| 10.4.1 | Shape Features | 229 |
| 10.4.2 | Geometric Features | 230 |
| 10.5 | Statistical Shape Properties | 232 |
| 10.5.1 | Centroid | 233 |
| 10.5.2 | Moments | 233 |
| 10.5.3 | Central Moments | 234 |
| 10.5.4 | Normalized Central Moments | 234 |
| 10.5.5 | Java Implementation | 234 |
| 10.6 | Moment-Based Geometric Properties | 235 |
| 10.6.1 | Orientation | 235 |
| 10.6.2 | Eccentricity | 237 |
| 10.6.3 | Bounding Box Aligned to the Major Axis | 239 |
| 10.6.4 | Invariant Region Moments | 241 |
| 10.7 | Projections | 244 |
| 10.8 | Topological Region Properties | 244 |
| 10.9 | Java Implementation | 246 |
| 10.10 | Exercises | 246 |
| 11 | Automatic Thresholding | 253 |
| 11.1 | Global Histogram-Based Thresholding | 253 |
| 11.1.1 | Image Statistics from the Histogram | 255 |
| 11.1.2 | Simple Threshold Selection | 256 |
| 11.1.3 | Iterative Threshold Selection (Isodata Algorithm) | 258 |
| 11.1.4 | Otsu's Method | 260 |

| | | |
|-----------|--|------------|
| 11.1.5 | Maximum Entropy Thresholding | 263 |
| 11.1.6 | Minimum Error Thresholding | 266 |
| 11.2 | Local Adaptive Thresholding | 273 |
| 11.2.1 | Bernsen's Method | 274 |
| 11.2.2 | Niblack's Method | 275 |
| 11.3 | Java Implementation | 284 |
| 11.3.1 | Global Thresholding Methods | 285 |
| 11.3.2 | Adaptive Thresholding | 287 |
| 11.4 | Summary and Further Reading | 288 |
| 11.5 | Exercises | 289 |
| 12 | Color Images | 291 |
| 12.1 | RGB Color Images | 291 |
| 12.1.1 | Structure of Color Images | 292 |
| 12.1.2 | Color Images in ImageJ | 296 |
| 12.2 | Color Spaces and Color Conversion | 303 |
| 12.2.1 | Conversion to Grayscale | 304 |
| 12.2.2 | Desaturating RGB Color Images | 306 |
| 12.2.3 | HSV/HSB and HLS Color Spaces | 306 |
| 12.2.4 | TV Component Color Spaces—YUV, YIQ, and $Y C_b C_r$ | 317 |
| 12.2.5 | Color Spaces for Printing—CMY and CMYK | 320 |
| 12.3 | Statistics of Color Images | 323 |
| 12.3.1 | How Many Different Colors are in an Image? | 323 |
| 12.3.2 | Color Histograms | 324 |
| 12.4 | Exercises | 325 |
| 13 | Color Quantization | 329 |
| 13.1 | Scalar Color Quantization | 329 |
| 13.2 | Vector Quantization | 331 |
| 13.2.1 | Populosity Algorithm | 331 |
| 13.2.2 | Median-Cut Algorithm | 332 |
| 13.2.3 | Octree Algorithm | 333 |
| 13.2.4 | Other Methods for Vector Quantization | 336 |
| 13.2.5 | Java Implementation | 337 |
| 13.3 | Exercises | 337 |
| 14 | Colorimetric Color Spaces | 341 |
| 14.1 | CIE Color Spaces | 341 |
| 14.1.1 | CIE XYZ Color Space | 342 |
| 14.1.2 | CIE x, y Chromaticity | 342 |
| 14.1.3 | Standard Illuminants | 344 |
| 14.1.4 | Gamut | 345 |
| 14.1.5 | Variants of the CIE Color Space | 345 |
| 14.2 | CIELAB | 346 |
| 14.2.1 | CIEXYZ→CIELAB Conversion | 346 |
| 14.2.2 | CIELAB→CIEXYZ Conversion | 347 |
| 14.3 | CIELUV | 348 |
| 14.3.1 | CIEXYZ→CIELUV Conversion | 348 |
| 14.3.2 | CIELUV→CIEXYZ Conversion | 350 |
| 14.3.3 | Measuring Color Differences | 350 |
| 14.4 | Standard RGB (sRGB) | 350 |

| | | |
|-----------|---|------------|
| 14.4.1 | Linear vs. Nonlinear Color Components | 351 |
| 14.4.2 | CIEXYZ→sRGB Conversion | 352 |
| 14.4.3 | sRGB→CIEXYZ Conversion | 353 |
| 14.4.4 | Calculations with Nonlinear sRGB Values . . . | 353 |
| 14.5 | Adobe RGB | 354 |
| 14.6 | Chromatic Adaptation | 355 |
| 14.6.1 | XYZ Scaling | 355 |
| 14.6.2 | Bradford Adaptation | 356 |
| 14.7 | Colorimetric Support in Java | 358 |
| 14.7.1 | Profile Connection Space (PCS) | 358 |
| 14.7.2 | Color-Related Java Classes | 360 |
| 14.7.3 | Implementation of the CIELAB Color Space (Example) | 361 |
| 14.7.4 | ICC Profiles | 362 |
| 14.8 | Exercises | 365 |
| 15 | Filters for Color Images | 367 |
| 15.1 | Linear Filters | 367 |
| 15.1.1 | Monochromatic Application of Linear Filters | 368 |
| 15.1.2 | Color Space Considerations | 370 |
| 15.1.3 | Linear Filtering with Circular Values | 374 |
| 15.2 | Nonlinear Color Filters | 378 |
| 15.2.1 | Scalar Median Filter | 378 |
| 15.2.2 | Vector Median Filter | 378 |
| 15.2.3 | Sharpening Vector Median Filter | 382 |
| 15.3 | Java Implementation | 385 |
| 15.4 | Further Reading | 387 |
| 15.5 | Exercises | 388 |
| 16 | Edge Detection in Color Images | 391 |
| 16.1 | Monochromatic Techniques | 392 |
| 16.2 | Edges in Vector-Valued Images | 395 |
| 16.2.1 | Multi-Dimensional Gradients | 397 |
| 16.2.2 | The Jacobian Matrix | 397 |
| 16.2.3 | Squared Local Contrast | 398 |
| 16.2.4 | Color Edge Magnitude | 399 |
| 16.2.5 | Color Edge Orientation | 401 |
| 16.2.6 | Grayscale Gradients Revisited | 401 |
| 16.3 | Canny Edge Detector for Color Images | 404 |
| 16.4 | Other Color Edge Operators | 406 |
| 16.5 | Java Implementation | 410 |
| 17 | Edge-Preserving Smoothing Filters | 413 |
| 17.1 | Kuwahara-Type Filters | 414 |
| 17.1.1 | Application to Color Images | 416 |
| 17.2 | Bilateral Filter | 420 |
| 17.2.1 | Domain Filter | 420 |
| 17.2.2 | Range Filter | 421 |
| 17.2.3 | Bilateral Filter—General Idea | 421 |
| 17.2.4 | Bilateral Filter with Gaussian Kernels | 423 |
| 17.2.5 | Application to Color Images | 424 |
| 17.2.6 | Efficient Implementation by x/y Separation . | 428 |

| | | |
|-----------|--|------------|
| 17.2.7 | Further Reading | 432 |
| 17.3 | Anisotropic Diffusion Filters | 433 |
| 17.3.1 | Homogeneous Diffusion and the Heat Equation | 434 |
| 17.3.2 | Perona-Malik Filter | 436 |
| 17.3.3 | Perona-Malik Filter for Color Images | 438 |
| 17.3.4 | Geometry Preserving Anisotropic Diffusion .. | 441 |
| 17.3.5 | Tschumperlé-Deriche Algorithm | 444 |
| 17.4 | Java Implementation | 448 |
| 17.5 | Exercises | 450 |
| 18 | Introduction to Spectral Techniques | 453 |
| 18.1 | The Fourier Transform | 454 |
| 18.1.1 | Sine and Cosine Functions | 454 |
| 18.1.2 | Fourier Series Representation of Periodic Functions | 457 |
| 18.1.3 | Fourier Integral | 457 |
| 18.1.4 | Fourier Spectrum and Transformation | 458 |
| 18.1.5 | Fourier Transform Pairs | 459 |
| 18.1.6 | Important Properties of the Fourier Transform | 460 |
| 18.2 | Working with Discrete Signals | 464 |
| 18.2.1 | Sampling | 464 |
| 18.2.2 | Discrete and Periodic Functions | 469 |
| 18.3 | The Discrete Fourier Transform (DFT) | 469 |
| 18.3.1 | Definition of the DFT | 469 |
| 18.3.2 | Discrete Basis Functions | 472 |
| 18.3.3 | Aliasing Again! | 472 |
| 18.3.4 | Units in Signal and Frequency Space | 475 |
| 18.3.5 | Power Spectrum | 477 |
| 18.4 | Implementing the DFT | 477 |
| 18.4.1 | Direct Implementation | 477 |
| 18.4.2 | Fast Fourier Transform (FFT) | 479 |
| 18.5 | Exercises | 479 |
| 19 | The Discrete Fourier Transform in 2D | 481 |
| 19.1 | Definition of the 2D DFT | 481 |
| 19.1.1 | 2D Basis Functions | 481 |
| 19.1.2 | Implementing the 2D DFT | 482 |
| 19.2 | Visualizing the 2D Fourier Transform | 485 |
| 19.2.1 | Range of Spectral Values | 485 |
| 19.2.2 | Centered Representation of the DFT Spectrum | 485 |
| 19.3 | Frequencies and Orientation in 2D | 486 |
| 19.3.1 | Effective Frequency | 486 |
| 19.3.2 | Frequency Limits and Aliasing in 2D | 487 |
| 19.3.3 | Orientation | 488 |
| 19.3.4 | Normalizing the Geometry of the 2D Spectrum | 488 |
| 19.3.5 | Effects of Periodicity | 489 |
| 19.3.6 | Windowing | 490 |
| 19.3.7 | Common Windowing Functions | 491 |
| 19.4 | 2D Fourier Transform Examples | 492 |

| | | |
|-----------|---|------------|
| 19.5 | Applications of the DFT | 496 |
| 19.5.1 | Linear Filter Operations in Frequency Space | 496 |
| 19.5.2 | Linear Convolution and Correlation | 499 |
| 19.5.3 | Inverse Filters | 499 |
| 19.6 | Exercises | 500 |
| 20 | The Discrete Cosine Transform (DCT) | 503 |
| 20.1 | 1D DCT | 503 |
| 20.1.1 | DCT Basis Functions | 504 |
| 20.1.2 | Implementing the 1D DCT | 504 |
| 20.2 | 2D DCT | 504 |
| 20.2.1 | Examples | 506 |
| 20.2.2 | Separability | 507 |
| 20.3 | Java Implementation | 509 |
| 20.4 | Other Spectral Transforms | 510 |
| 20.5 | Exercises | 510 |
| 21 | Geometric Operations | 513 |
| 21.1 | 2D Coordinate Transformations | 514 |
| 21.1.1 | Simple Geometric Mappings | 514 |
| 21.1.2 | Homogeneous Coordinates | 515 |
| 21.1.3 | Affine (Three-Point) Mapping | 516 |
| 21.1.4 | Projective (Four-Point) Mapping | 519 |
| 21.1.5 | Bilinear Mapping | 525 |
| 21.1.6 | Other Nonlinear Image Transformations | 526 |
| 21.1.7 | Piecewise Image Transformations | 528 |
| 21.2 | Resampling the Image | 529 |
| 21.2.1 | Source-to-Target Mapping | 530 |
| 21.2.2 | Target-to-Source Mapping | 530 |
| 21.3 | Java Implementation | 531 |
| 21.3.1 | General Mappings (Class Mapping) | 532 |
| 21.3.2 | Linear Mappings | 532 |
| 21.3.3 | Nonlinear Mappings | 533 |
| 21.3.4 | Sample Applications | 533 |
| 21.4 | Exercises | 534 |
| 22 | Pixel Interpolation | 539 |
| 22.1 | Simple Interpolation Methods | 539 |
| 22.1.1 | Ideal Low-Pass Filter | 540 |
| 22.2 | Interpolation by Convolution | 543 |
| 22.3 | Cubic Interpolation | 544 |
| 22.4 | Spline Interpolation | 546 |
| 22.4.1 | Catmull-Rom Interpolation | 546 |
| 22.4.2 | Cubic B-spline Approximation | 547 |
| 22.4.3 | Mitchell-Netravali Approximation | 547 |
| 22.4.4 | Lanczos Interpolation | 548 |
| 22.5 | Interpolation in 2D | 549 |
| 22.5.1 | Nearest-Neighbor Interpolation in 2D | 550 |
| 22.5.2 | Bilinear Interpolation | 551 |
| 22.5.3 | Bicubic and Spline Interpolation in 2D | 553 |
| 22.5.4 | Lanczos Interpolation in 2D | 554 |
| 22.5.5 | Examples and Discussion | 555 |

| | | |
|-----------|--|------------|
| 22.6 | Aliasing | 556 |
| 22.6.1 | Sampling the Interpolated Image | 557 |
| 22.6.2 | Low-Pass Filtering | 558 |
| 22.7 | Java Implementation | 560 |
| 22.8 | Exercises | 563 |
| 23 | Image Matching and Registration | 565 |
| 23.1 | Template Matching in Intensity Images | 566 |
| 23.1.1 | Distance between Image Patterns | 566 |
| 23.1.2 | Matching Under Rotation and Scaling | 574 |
| 23.1.3 | Java Implementation | 574 |
| 23.2 | Matching Binary Images | 574 |
| 23.2.1 | Direct Comparison of Binary Images | 576 |
| 23.2.2 | The Distance Transform | 576 |
| 23.2.3 | Chamfer Matching | 580 |
| 23.2.4 | Java Implementation | 582 |
| 23.3 | Exercises | 583 |
| 24 | Non-Rigid Image Matching | 587 |
| 24.1 | The Lucas-Kanade Technique | 587 |
| 24.1.1 | Registration in 1D | 587 |
| 24.1.2 | Extension to Multi-Dimensional Functions .. | 589 |
| 24.2 | The Lucas-Kanade Algorithm | 590 |
| 24.2.1 | Summary of the Algorithm | 593 |
| 24.3 | Inverse Compositional Algorithm | 595 |
| 24.4 | Parameter Setups for Various Linear Transformations | 598 |
| 24.4.1 | Pure Translation | 598 |
| 24.4.2 | Affine Transformation | 599 |
| 24.4.3 | Projective Transformation | 601 |
| 24.4.4 | Concatenating Linear Transformations | 601 |
| 24.5 | Example | 602 |
| 24.6 | Java Implementation | 603 |
| 24.6.1 | Application Example | 605 |
| 24.7 | Exercises | 607 |
| 25 | Scale-Invariant Feature Transform (SIFT) | 609 |
| 25.1 | Interest Points at Multiple Scales | 610 |
| 25.1.1 | The LoG Filter | 610 |
| 25.1.2 | Gaussian Scale Space | 615 |
| 25.1.3 | LoG/DoG Scale Space | 619 |
| 25.1.4 | Hierarchical Scale Space | 620 |
| 25.1.5 | Scale Space Structure in SIFT | 624 |
| 25.2 | Key Point Selection and Refinement | 630 |
| 25.2.1 | Local Extrema Detection | 630 |
| 25.2.2 | Position Refinement | 632 |
| 25.2.3 | Suppressing Responses to Edge-Like Structures | 634 |
| 25.3 | Creating Local Descriptors | 636 |
| 25.3.1 | Finding Dominant Orientations | 637 |
| 25.3.2 | SIFT Descriptor Construction | 640 |
| 25.4 | SIFT Algorithm Summary | 647 |
| 25.5 | Matching SIFT Features | 648 |

| | | |
|-----------|--|------------|
| 25.5.1 | Feature Distance and Match Quality | 648 |
| 25.5.2 | Examples | 654 |
| 25.6 | Efficient Feature Matching | 657 |
| 25.7 | Java Implementation | 661 |
| 25.7.1 | SIFT Feature Extraction | 662 |
| 25.7.2 | SIFT Feature Matching | 663 |
| 25.8 | Exercises | 663 |
| 26 | Fourier Shape Descriptors | 665 |
| 26.1 | Closed Curves in the Complex Plane | 665 |
| 26.1.1 | Discrete 2D Curves | 665 |
| 26.2 | Discrete Fourier Transform (DFT) | 667 |
| 26.2.1 | Forward Fourier Transform | 668 |
| 26.2.2 | Inverse Fourier Transform (Reconstruction) | 668 |
| 26.2.3 | Periodicity of the DFT Spectrum | 670 |
| 26.2.4 | Truncating the DFT Spectrum | 672 |
| 26.3 | Geometric Interpretation of Fourier Coefficients | 673 |
| 26.3.1 | Coefficient G_0 Corresponds to the Contour's Centroid | 673 |
| 26.3.2 | Coefficient G_1 Corresponds to a Circle | 674 |
| 26.3.3 | Coefficient G_m Corresponds to a Circle with Frequency m | 675 |
| 26.3.4 | Negative Frequencies | 676 |
| 26.3.5 | Fourier Descriptor Pairs Correspond to Ellipses | 676 |
| 26.3.6 | Shape Reconstruction from Truncated Fourier Descriptors | 679 |
| 26.3.7 | Fourier Descriptors from Unsampled Polygons | 682 |
| 26.4 | Effects of Geometric Transformations | 687 |
| 26.4.1 | Translation | 687 |
| 26.4.2 | Scale Change | 688 |
| 26.4.3 | Rotation | 688 |
| 26.4.4 | Shifting the Sampling Start Position | 689 |
| 26.4.5 | Effects of Phase Removal | 690 |
| 26.4.6 | Direction of Contour Traversal | 691 |
| 26.4.7 | Reflection (Symmetry) | 691 |
| 26.5 | Transformation-Invariant Fourier Descriptors | 692 |
| 26.5.1 | Scale Invariance | 693 |
| 26.5.2 | Start Point Invariance | 694 |
| 26.5.3 | Rotation Invariance | 696 |
| 26.5.4 | Other Approaches | 697 |
| 26.6 | Shape Matching with Fourier Descriptors | 700 |
| 26.6.1 | Magnitude-Only Matching | 700 |
| 26.6.2 | Complex (Phase-Preserving) Matching | 701 |
| 26.7 | Java Implementation | 704 |
| 26.8 | Discussion and Further Reading | 708 |
| 26.9 | Exercises | 709 |
| A | Mathematical Symbols and Notation | 713 |
| A.1 | Symbols | 713 |
| A.2 | Set Operators | 717 |
| A.3 | Complex Numbers | 717 |

| | | |
|----------|--|-----|
| B | Linear Algebra | 719 |
| B.1 | Vectors and Matrices | 719 |
| B.1.1 | Column and Row Vectors | 720 |
| B.1.2 | Length (Norm) of a Vector | 720 |
| B.2 | Matrix Multiplication | 720 |
| B.2.1 | Scalar Multiplication | 720 |
| B.2.2 | Product of Two Matrices | 721 |
| B.2.3 | Matrix-Vector Products | 721 |
| B.3 | Vector Products | 722 |
| B.3.1 | Dot (Scalar) Product | 722 |
| B.3.2 | Outer Product | 723 |
| B.3.3 | Cross Product | 723 |
| B.4 | Eigenvectors and Eigenvalues | 723 |
| B.4.1 | Calculation of Eigenvalues | 724 |
| B.5 | Homogeneous Coordinates | 726 |
| B.6 | Basic Matrix-Vector Operations with the <i>Apache Commons Math</i> Library | 727 |
| B.6.1 | Vectors and Matrices | 727 |
| B.6.2 | Matrix-Vector Multiplication | 728 |
| B.6.3 | Vector Products | 728 |
| B.6.4 | Inverse of a Square Matrix | 728 |
| B.6.5 | Eigenvalues and Eigenvectors | 728 |
| B.7 | Solving Systems of Linear Equations | 729 |
| B.7.1 | Exact Solutions | 730 |
| B.7.2 | Over-Determined System (Least-Squares Solutions) | 731 |
| C | Calculus | 733 |
| C.1 | Parabolic Fitting | 733 |
| C.1.1 | Fitting a Parabolic Function to Three Sample Points | 733 |
| C.1.2 | Locating Extrema by Quadratic Interpolation | 734 |
| C.2 | Scalar and Vector Fields | 735 |
| C.2.1 | The Jacobian Matrix | 736 |
| C.2.2 | Gradients | 736 |
| C.2.3 | Maximum Gradient Direction | 737 |
| C.2.4 | Divergence of a Vector Field | 737 |
| C.2.5 | Laplacian Operator | 738 |
| C.2.6 | The Hessian Matrix | 738 |
| C.3 | Operations on Multi-Variable, Scalar Functions (Scalar Fields) | 739 |
| C.3.1 | Estimating the Derivatives of a Discrete Function | 739 |
| C.3.2 | Taylor Series Expansion of Functions | 740 |
| C.3.3 | Finding the Continuous Extremum of a Multi-Variable Discrete Function | 743 |
| D | Statistical Prerequisites | 749 |
| D.1 | Mean, Variance, and Covariance | 749 |
| D.1.1 | Mean | 749 |
| D.1.2 | Variance and Covariance | 749 |
| D.1.3 | Biased vs. Unbiased Variance | 750 |

| | | | |
|----------|--|-----|----------------|
| D.2 | The Covariance Matrix | 750 | <hr/> CONTENTS |
| D.2.1 | Example | 751 | |
| D.2.2 | Practical Calculation | 752 | |
| D.3 | Mahalanobis Distance | 752 | |
| D.3.1 | Definition | 752 | |
| D.3.2 | Relation to the Euclidean Distance | 753 | |
| D.3.3 | Numerical Aspects | 753 | |
| D.3.4 | Pre-Mapping Data for Efficient Mahalanobis Matching | 754 | |
| D.4 | The Gaussian Distribution | 756 | |
| D.4.1 | Maximum Likelihood Estimation | 756 | |
| D.4.2 | Gaussian Mixtures | 758 | |
| D.4.3 | Creating Gaussian Noise | 758 | |
| E | Gaussian Filters | 761 | |
| E.1 | Cascading Gaussian Filters | 761 | |
| E.2 | Gaussian Filters and Scale Space | 761 | |
| E.3 | Effects of Gaussian Filtering in the Frequency Domain | 762 | |
| E.4 | LoG-Approximation by the DoG | 763 | |
| F | Java Notes | 765 | |
| F.1 | Arithmetic | 765 | |
| F.1.1 | Integer Division | 765 | |
| F.1.2 | Modulus Operator | 766 | |
| F.1.3 | Unsigned Byte Data | 767 | |
| F.1.4 | Mathematical Functions in Class <code>Math</code> | 768 | |
| F.1.5 | Numerical Rounding | 769 | |
| F.1.6 | Inverse Tangent Function | 769 | |
| F.1.7 | Classes <code>Float</code> and <code>Double</code> | 770 | |
| F.1.8 | Testing Floating-Point Values Against Zero .. | 770 | |
| F.2 | Arrays in Java | 771 | |
| F.2.1 | Creating Arrays | 771 | |
| F.2.2 | Array Size | 771 | |
| F.2.3 | Accessing Array Elements | 771 | |
| F.2.4 | 2D Arrays | 772 | |
| F.2.5 | Arrays of Objects | 775 | |
| F.2.6 | Searching for Minimum and Maximum Values | 775 | |
| F.2.7 | Sorting Arrays | 776 | |
| | References | 777 | |
| | Index | 791 | |