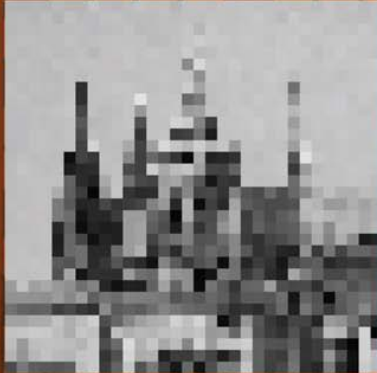




FOURTH EDITION



Image Processing, Analysis, and Machine Vision



Milan Sonka
Vaclav Hlavac
Roger Boyle

Image Processing, Analysis, and Machine Vision

Image Processing, Analysis, and Machine Vision

Fourth Edition

Milan Sonka

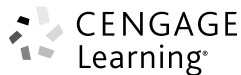
The University of Iowa, Iowa City

Vaclav Hlavac

Czech Technical University, Prague

Roger Boyle

Prifysgol Aberystwyth, Aberystwyth



Australia • Brazil • Japan • Mexico • Singapore • United Kingdom • United States

This is an electronic version of the print textbook. Due to electronic rights restrictions, some third party content may be suppressed. Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. The publisher reserves the right to remove content from this title at any time if subsequent rights restrictions require it. For valuable information on pricing, previous editions, changes to current editions, and alternate formats, please visit www.cengage.com/highered to search by ISBN#, author, title, or keyword for materials in your areas of interest.



**Image Processing, Analysis, and
Machine Vision, Fourth Edition**
Milan Sonka, Vaclav Hlavac, and
Roger Boyle

Publisher, Global Engineering:
Timothy L. Anderson
Senior Developmental Editor:
Hilda Gowans
Senior Editorial Assistant: Tanya Altieri
Art and Cover Direction, Production
Management: PreMediaGlobal
Compositor: Hrvoje Bugonovic
Rights Acquisition Director:
Audrey Pettengill
Rights Acquisition Specialist,
Text and Image: Amber Hosea
Text & Image Permissions Researcher:
Kristiina Paul
Manufacturing Planner: Doug Wilke
Cover Image: © Milan Sonka,
Vaclav Hlavac, and Roger Boyle

© 2015, 2008 Cengage Learning

WCN: 02-200-203

ALL RIGHTS RESERVED. No part of this work covered by the copyright herein may be reproduced, transmitted, stored or used in any form or by any means graphic, electronic, or mechanical, including but not limited to photocopying, recording, scanning, digitizing, taping, Web distribution, information networks, or information storage and retrieval systems, except as permitted under Section 107 or 108 of the 1976 United States Copyright Act, without the prior written permission of the publisher.

For product information and technology assistance, contact us at
Cengage Learning Customer & Sales Support, 1-800-354-9706

For permission to use material from this text or product,
submit all requests online at **www.cengage.com/permissions**
Further permissions questions can be emailed to
permissionrequest@cengage.com

Library of Congress Control Number: 2013947420

ISBN-13: 978-1-133-59360-7

ISBN-10: 1-133-59360-7

Cengage Learning
200 First Stamford Place, 4th Floor
Stamford, CT 06902
USA

Cengage Learning is a leading provider of customized learning solutions with office locations around the globe, including Singapore, the United Kingdom, Australia, Mexico, Brazil, and Japan. Locate your local office at **www.cengage.com/global**

Cengage Learning products are represented in Canada by
Nelson Education, Ltd.

To learn more about Cengage Learning, visit
www.cengage.com/engineering.

Purchase any of our products at your local college store or at our
preferred online store **www.cengagebrain.com**

Printed in the United States of America
1 2 3 4 5 6 7 17 16 15 14 13

Abbreviations

1D	one dimension(al)
2D, 3D, ...	two dimension(al), three dimension(al), ...
AAM	active appearance model
AGC	automatic gain control
AI	artificial intelligence
ART	adaptive resonance theory
ASM	active shape model
BBF	best bin first
BBN	Bayesian belief network
BRDF	bi-directional reflectance distribution function
B-rep	boundary representation
CAD	computer-aided design
CCD	charge-coupled device
CHMM	coupled HMM
CIE	International Commission on Illumination
CMOS	complementary metal-oxide semiconductor
CMY	cyan, magenta, yellow
CONDENSATION	CONditional DENsity propagATION
CRT	cathode ray tube
CSF	cerebro-spinal fluid
CSG	constructive solid geometry
CT	computed tomography
dB	decibel, 20 times the decimal logarithm of a ratio
DCT	discrete cosine transform
DFT	discrete Fourier transform
dof	degrees of freedom
DPCM	differential PCM
DWF	discrete wavelet frame
ECG	electro-cardiogram
EEG	electro-encephalogram
EM	expectation-maximization
FFT	fast Fourier transform
FLANN	fast library for approximate nearest neighbors
FOE	focus of expansion
GA	genetic algorithm

GB	Giga byte = 2^{30} bytes = 1,073,741,824 bytes
GIS	geographic information system
GMM	Gaussian mixture model
GRBF	Gaussian radial basis function
GVF	gradient vector flow
HDTV	high definition TV
HLS	as HSI
HMM	hidden Markov model
HOG	histogram of oriented gradients
HSI	hue, saturation, intensity
HSL	as HSI
HSV	hue, saturation, value
ICA	independent component analysis
ICP	iterative closest point algorithm
ICRP	iterative closest reciprocal point algorithm
IHS	intensity, hue, saturation
JPEG	Joint Photographic Experts Group
Kb	Kilo bit = 2^{10} bits = 1,024 bits
KB	Kilo byte = 2^{10} bytes = 1,024 bytes
KLT	Kanade-Lucas-Tomasi (tracker)
LBP	local binary pattern
LCD	liquid crystal display
MAP	maximum a posteriori
Mb	Mega bit = 2^{20} bits = 1,048,576 bits
MB	Mega byte = 2^{20} bytes = 1,048,576 bytes
MB, MB2	Manzanera-Bernard skeletonization
MCMC	Monte Carlo Markov chain
MDL	minimum description length
MJPEG	motion JPEG
MPEG	moving picture experts group
MRF	Markov random field
MRI	magnetic resonance imaging
MR	magnetic resonance
MSE	mean-square error
MSER	maximally stable extremal region
ms	millisecond

μs	microsecond
OCR	optical character recognition
OS	order statistics
PCA	principal component analysis
PDE	partial differential equation
p.d.f.	probability density function
PDM	point distribution model
PET	positron emission tomography
PMF	Pollard-Mayhew-Frisby (correspondence algorithm)
PTZ	pan-tilt-zoom
RANSAC	RANdom SAMple Consensus
RBF	radial basis function
RCT	reversible component transform
RGB	red, green, blue
RMS	root mean square
SIFT	scale invariant feature transform
SKIZ	skeleton by inference zones
SLR	single lens reflex
SNR	signal-to-noise ratio
STFT	short term Fourier transform
SVD	singular value decomposition
SVM	support vector machine
TLD	tracking-learning-detection
TV	television
USB	universal serial bus

Symbols

$\arg(x, y)$	angle (in radians) from x axis to the point (x, y)
$\operatorname{argmax}_i(\operatorname{expr}(i))$	the value of i that causes $\operatorname{expr}(i)$ to be maximal
$\operatorname{argmin}_i(\operatorname{expr}(i))$	the value of i that causes $\operatorname{expr}(i)$ to be minimal
div	integer division or divergence
mod	remainder after integer division
$\operatorname{round}(x)$	largest integer which is not bigger than $x + 0.5$
$\emptyset$	empty set
A^c	complement of set A
$A \subset B, B \supset A$	set A is included in set B
$A \cap B$	intersection between sets A and B
$A \cup B$	union of sets A and B
$A \setminus B$	difference between sets A and B
$\mathbf{A}$	(uppercase bold) matrices
$\mathbf{x}$	(lowercase bold) vectors
$\ \mathbf{x}\ $	magnitude (or modulus) of vector $\mathbf{x}$
$\mathbf{x} \cdot \mathbf{y}$	scalar product between vectors $\mathbf{x}$ and $\mathbf{y}$
$\tilde{x}$	estimate of the value x
$ x $	absolute value of a scalar
$\delta(x)$	Dirac function
Δx	small finite interval of x , difference
$\partial f / \partial x$	partial derivative of the function f with respect to x
$\nabla \mathbf{f}, \operatorname{grad} \mathbf{f}$	gradient of $\mathbf{f}$
$\nabla^2 \mathbf{f}$	Laplace operator applied to $\mathbf{f}$
$f * g$	convolution between functions f and g
$F.*G$	element-by-element multiplication of matrices F, G
D_E	Euclidean distance
D_4	city block distance
D_8	chessboard distance
F^*	complex conjugate of the complex function F
$\operatorname{rank}(A)$	rank of a matrix A
T^*	transformation dual to transformation T , also complex conjugate of T
$\mathcal{E}$	mean value operator
$\mathcal{L}$	linear operator
$\mathcal{O}$	origin of the coordinate system

#	number of (e.g., pixels)
$\tilde{B}$	point set symmetrical to point set B
$\oplus$	morphological dilation
$\ominus$	morphological erosion
$\circ$	morphological opening
$\bullet$	morphological closing
$\otimes$	morphological hit-or-miss transformation
$\oslash$	morphological thinning
$\odot$	morphological thickening
$\wedge$	logical and
$\vee$	logical or
trace	sum of elements on the matrix main diagonal
cov	covariance matrix
sec	secant, $\sec \alpha = 1/\cos \alpha$

Contents

List of algorithms	xxi
Preface	xxv
Possible course outlines	xxxi
1 Introduction	1
1.1 Motivation	1
1.2 Why is computer vision difficult?	3
1.3 Image representation and image analysis tasks	5
1.4 Summary	9
1.5 Exercises	10
1.6 References	10
2 The image, its representations and properties	11
2.1 Image representations, a few concepts	11
2.2 Image digitization	14
2.2.1 Sampling	14
2.2.2 Quantization	15
2.3 Digital image properties	16
2.3.1 Metric and topological properties of digital images	16
2.3.2 Histograms	23
2.3.3 Entropy	24
2.3.4 Visual perception of the image	25
2.3.5 Image quality	27
2.3.6 Noise in images	28
2.4 Color images	30
2.4.1 Physics of color	30
2.4.2 Color perceived by humans	32
2.4.3 Color spaces	36
2.4.4 Palette images	38
2.4.5 Color constancy	39
2.5 Cameras: An overview	40
2.5.1 Photosensitive sensors	40
2.5.2 A monochromatic camera	42
2.5.3 A color camera	44
2.6 Summary	45

xii **Contents**

2.7	Exercises	46
2.8	References	48
3	The image, its mathematical and physical background	50
3.1	Overview	50
3.1.1	Linearity	50
3.1.2	The Dirac distribution and convolution	51
3.2	Linear integral transforms	52
3.2.1	Images as linear systems	53
3.2.2	Introduction to linear integral transforms	53
3.2.3	1D Fourier transform	54
3.2.4	2D Fourier transform	59
3.2.5	Sampling and the Shannon constraint	62
3.2.6	Discrete cosine transform	65
3.2.7	Wavelet transform	66
3.2.8	Eigen-analysis	72
3.2.9	Singular value decomposition	73
3.2.10	Principal component analysis	74
3.2.11	Radon transform	77
3.2.12	Other orthogonal image transforms	78
3.3	Images as stochastic processes	79
3.4	Image formation physics	82
3.4.1	Images as radiometric measurements	82
3.4.2	Image capture and geometric optics	83
3.4.3	Lens aberrations and radial distortion	86
3.4.4	Image capture from a radiometric point of view	89
3.4.5	Surface reflectance	92
3.5	Summary	95
3.6	Exercises	97
3.7	References	98
4	Data structures for image analysis	100
4.1	Levels of image data representation	100
4.2	Traditional image data structures	101
4.2.1	Matrices	101
4.2.2	Chains	104
4.2.3	Topological data structures	106
4.2.4	Relational structures	107
4.3	Hierarchical data structures	108
4.3.1	Pyramids	108
4.3.2	Quadrees	109
4.3.3	Other pyramidal structures	111
4.4	Summary	112
4.5	Exercises	113
4.6	References	115
5	Image pre-processing	116
5.1	Pixel brightness transformations	117

5.1.1	Position-dependent brightness correction	117
5.1.2	Gray-scale transformation	117
5.2	Geometric transformations	120
5.2.1	Pixel co-ordinate transformations	121
5.2.2	Brightness interpolation	123
5.3	Local pre-processing	125
5.3.1	Image smoothing	125
5.3.2	Edge detectors	133
5.3.3	Zero-crossings of the second derivative	139
5.3.4	Scale in image processing	143
5.3.5	Canny edge detection	144
5.3.6	Parametric edge models	147
5.3.7	Edges in multi-spectral images	148
5.3.8	Local pre-processing in the frequency domain	148
5.3.9	Line detection by local pre-processing operators	155
5.3.10	Detection of corners (interest points)	156
5.3.11	Detection of maximally stable extremal regions	160
5.4	Image restoration	162
5.4.1	Degradations that are easy to restore	163
5.4.2	Inverse filtering	163
5.4.3	Wiener filtering	164
5.5	Summary	165
5.6	Exercises	167
5.7	References	174
6	Segmentation I	178
6.1	Thresholding	179
6.1.1	Threshold detection methods	181
6.1.2	Optimal thresholding	183
6.1.3	Multi-spectral thresholding	186
6.2	Edge-based segmentation	187
6.2.1	Edge image thresholding	188
6.2.2	Edge relaxation	190
6.2.3	Border tracing	191
6.2.4	Border detection as graph searching	196
6.2.5	Border detection as dynamic programming	206
6.2.6	Hough transforms	210
6.2.7	Border detection using border location information	217
6.2.8	Region construction from borders	218
6.3	Region-based segmentation	220
6.3.1	Region merging	221
6.3.2	Region splitting	224
6.3.3	Splitting and merging	225
6.3.4	Watershed segmentation	229
6.3.5	Region growing post-processing	232
6.4	Matching	232
6.4.1	Template matching	233
6.4.2	Control strategies of templating	235

6.5	Evaluation issues in segmentation	236
6.5.1	Supervised evaluation	237
6.5.2	Unsupervised evaluation	240
6.6	Summary	241
6.7	Exercises	245
6.8	References	248
7	Segmentation II	255
7.1	Mean shift segmentation	255
7.2	Active contour models—snakes	263
7.2.1	Traditional snakes and balloons	264
7.2.2	Extensions	267
7.2.3	Gradient vector flow snakes	268
7.3	Geometric deformable models—level sets and geodesic active contours	273
7.4	Fuzzy connectivity	280
7.5	Towards 3D graph-based image segmentation	288
7.5.1	Simultaneous detection of border pairs	289
7.5.2	Suboptimal surface detection	293
7.6	Graph cut segmentation	295
7.7	Optimal single and multiple surface segmentation— LOGISMOS	303
7.8	Summary	317
7.9	Exercises	319
7.10	References	321
8	Shape representation and description	329
8.1	Region identification	333
8.2	Contour-based shape representation and description	335
8.2.1	Chain codes	336
8.2.2	Simple geometric border representation	337
8.2.3	Fourier transforms of boundaries	341
8.2.4	Boundary description using segment sequences	343
8.2.5	B-spline representation	346
8.2.6	Other contour-based shape description approaches	348
8.2.7	Shape invariants	349
8.3	Region-based shape representation and description	353
8.3.1	Simple scalar region descriptors	353
8.3.2	Moments	358
8.3.3	Convex hull	360
8.3.4	Graph representation based on region skeleton	365
8.3.5	Region decomposition	370
8.3.6	Region neighborhood graphs	372
8.4	Shape classes	373
8.5	Summary	373
8.6	Exercises	375
8.7	References	379
9	Object recognition	385
9.1	Knowledge representation	386

9.2	Statistical pattern recognition	390
9.2.1	Classification principles	392
9.2.2	Nearest neighbors	393
9.2.3	Classifier setting	395
9.2.4	Classifier learning	398
9.2.5	Support vector machines	400
9.2.6	Cluster analysis	406
9.3	Neural nets	407
9.3.1	Feed-forward networks	409
9.3.2	Unsupervised learning	411
9.3.3	Hopfield neural nets	412
9.4	Syntactic pattern recognition	413
9.4.1	Grammars and languages	415
9.4.2	Syntactic analysis, syntactic classifier	417
9.4.3	Syntactic classifier learning, grammar inference	420
9.5	Recognition as graph matching	421
9.5.1	Isomorphism of graphs and subgraphs	421
9.5.2	Similarity of graphs	425
9.6	Optimization techniques in recognition	426
9.6.1	Genetic algorithms	427
9.6.2	Simulated annealing	430
9.7	Fuzzy systems	432
9.7.1	Fuzzy sets and fuzzy membership functions	432
9.7.2	Fuzzy set operators	434
9.7.3	Fuzzy reasoning	435
9.7.4	Fuzzy system design and training	438
9.8	Boosting in pattern recognition	439
9.9	Random forests	442
9.9.1	Random forest training	444
9.9.2	Random forest decision making	446
9.9.3	Random forest extensions	448
9.10	Summary	448
9.11	Exercises	452
9.12	References	459
10	Image understanding	464
10.1	Image understanding control strategies	466
10.1.1	Parallel and serial processing control	466
10.1.2	Hierarchical control	466
10.1.3	Bottom-up control	467
10.1.4	Model-based control	468
10.1.5	Combined control	469
10.1.6	Non-hierarchical control	472
10.2	SIFT: Scale invariant feature transform	474
10.3	RANSAC: Fitting via random sample consensus	477
10.4	Point distribution models	481
10.5	Active appearance models	492
10.6	Pattern recognition methods in image understanding	503

10.6.1	Classification-based segmentation	503
10.6.2	Contextual image classification	505
10.6.3	Histograms of oriented gradients—HOG	509
10.7	Boosted cascades of classifiers	513
10.8	Image understanding using random forests	517
10.9	Scene labeling and constraint propagation	524
10.9.1	Discrete relaxation	525
10.9.2	Probabilistic relaxation	527
10.9.3	Searching interpretation trees	530
10.10	Semantic image segmentation and understanding	531
10.10.1	Semantic region growing	532
10.10.2	Genetic image interpretation	534
10.11	Hidden Markov models	543
10.11.1	Applications	548
10.11.2	Coupled HMMs	549
10.11.3	Bayesian belief networks	551
10.12	Markov random fields	553
10.12.1	Applications to images and vision	555
10.13	Gaussian mixture models and expectation–maximization	556
10.14	Summary	564
10.15	Exercises	568
10.16	References	572
11	3D geometry, correspondence, 3D from intensities	582
11.1	3D vision tasks	583
11.1.1	Marr’s theory	585
11.1.2	Other vision paradigms: Active and purposive vision	587
11.2	Basics of projective geometry	589
11.2.1	Points and hyperplanes in projective space	590
11.2.2	Homography	592
11.2.3	Estimating homography from point correspondences	594
11.3	A single perspective camera	598
11.3.1	Camera model	598
11.3.2	Projection and back-projection in homogeneous coordinates	601
11.3.3	Camera calibration from a known scene	602
11.4	Scene reconstruction from multiple views	602
11.4.1	Triangulation	603
11.4.2	Projective reconstruction	604
11.4.3	Matching constraints	605
11.4.4	Bundle adjustment	607
11.4.5	Upgrading the projective reconstruction, self-calibration	608
11.5	Two cameras, stereopsis	609
11.5.1	Epipolar geometry; fundamental matrix	610
11.5.2	Relative motion of the camera; essential matrix	612
11.5.3	Decomposing the fundamental matrix to camera matrices	613
11.5.4	Estimating the fundamental matrix from point correspondences	614
11.5.5	Rectified configuration of two cameras	615
11.5.6	Computing rectification	617

11.6	Three cameras and trifocal tensor	619
11.6.1	Stereo correspondence algorithms	621
11.6.2	Active acquisition of range images	627
11.7	3D information from radiometric measurements	630
11.7.1	Shape from shading	631
11.7.2	Photometric stereo	635
11.8	Summary	636
11.9	Exercises	637
11.10	References	639
12	Use of 3D vision	644
12.1	Shape from X	644
12.1.1	Shape from motion	644
12.1.2	Shape from texture	651
12.1.3	Other shape from X techniques	652
12.2	Full 3D objects	655
12.2.1	3D objects, models, and related issues	655
12.2.2	Line labeling	656
12.2.3	Volumetric representation, direct measurements	658
12.2.4	Volumetric modeling strategies	660
12.2.5	Surface modeling strategies	662
12.2.6	Registering surface patches and their fusion to get a full 3D model	663
12.3	2D view-based representations of a 3D scene	670
12.3.1	Viewing space	670
12.3.2	Multi-view representations and aspect graphs	670
12.4	3D reconstruction from an unorganized set of 2D views, and Structure from Motion	671
12.5	Reconstructing scene geometry	674
12.6	Summary	677
12.7	Exercises	678
12.8	References	680
13	Mathematical morphology	684
13.1	Basic morphological concepts	684
13.2	Four morphological principles	686
13.3	Binary dilation and erosion	687
13.3.1	Dilation	688
13.3.2	Erosion	689
13.3.3	Hit-or-miss transformation	692
13.3.4	Opening and closing	692
13.4	Gray-scale dilation and erosion	694
13.4.1	Top surface, umbra, and gray-scale dilation and erosion	694
13.4.2	Umbra homeomorphism theorem, properties of erosion and dilation, opening and closing	697
13.4.3	Top hat transformation	698
13.5	Skeletons and object marking	699
13.5.1	Homotopic transformations	699

xviii Contents

13.5.2	Skeleton, medial axis, maximal ball	699
13.5.3	Thinning, thickening, and homotopic skeleton	701
13.5.4	Quench function, ultimate erosion	704
13.5.5	Ultimate erosion and distance functions	706
13.5.6	Geodesic transformations	707
13.5.7	Morphological reconstruction	709
13.6	Granulometry	711
13.7	Morphological segmentation and watersheds	713
13.7.1	Particle segmentation, marking, and watersheds	713
13.7.2	Binary morphological segmentation	714
13.7.3	Gray-scale segmentation, watersheds	716
13.8	Summary	717
13.9	Exercises	718
13.10	References	720
14	Image data compression	722
14.1	Image data properties	723
14.2	Discrete image transforms in image data compression	724
14.3	Predictive compression methods	727
14.4	Vector quantization	730
14.5	Hierarchical and progressive compression methods	730
14.6	Comparison of compression methods	732
14.7	Other techniques	733
14.8	Coding	733
14.9	JPEG and MPEG image compression	734
14.9.1	JPEG—still image compression	734
14.9.2	JPEG—2000 compression	736
14.9.3	MPEG—full-motion video compression	738
14.10	Summary	740
14.11	Exercises	742
14.12	References	744
15	Texture	747
15.1	Statistical texture description	750
15.1.1	Methods based on spatial frequencies	750
15.1.2	Co-occurrence matrices	752
15.1.3	Edge frequency	754
15.1.4	Primitive length (run length)	755
15.1.5	Laws' texture energy measures	757
15.1.6	Local binary patterns—LBPs	757
15.1.7	Fractal texture description	762
15.1.8	Multiscale texture description—wavelet domain approaches	764
15.1.9	Other statistical methods of texture description	768
15.2	Syntactic texture description methods	769
15.2.1	Shape chain grammars	770
15.2.2	Graph grammars	772
15.2.3	Primitive grouping in hierarchical textures	773

	Contents	xix
15.3	Hybrid texture description methods	775
15.4	Texture recognition method applications	776
15.5	Summary	777
15.6	Exercises	779
15.7	References	782
16	Motion analysis	787
16.1	Differential motion analysis methods	790
16.2	Optical flow	794
16.2.1	Optical flow computation	794
16.2.2	Global and local optical flow estimation	797
16.2.3	Combined local–global optical flow estimation	800
16.2.4	Optical flow in motion analysis	801
16.3	Analysis based on correspondence of interest points	804
16.3.1	Detection of interest points	805
16.3.2	Lucas–Kanade point tracking	805
16.3.3	Correspondence of interest points	807
16.4	Detection of specific motion patterns	810
16.5	Video tracking	814
16.5.1	Background modeling	815
16.5.2	Kernel-based tracking	820
16.5.3	Object path analysis	826
16.6	Motion models to aid tracking	831
16.6.1	Kalman filters	831
16.6.2	Particle filters	837
16.6.3	Semi-supervised tracking—TLD	840
16.7	Summary	843
16.8	Exercises	846
16.9	References	848
Index		853

List of algorithms

2.1	Distance transform	20
2.2	Computing the brightness histogram	23
2.3	Generation of additive, zero mean Gaussian noise	28
4.1	Co-occurrence matrix $C_r(z, y)$ for the relation r	102
4.2	Integral image construction	103
5.1	Histogram equalization	119
5.2	Smoothing using a rotating mask	130
5.3	Efficient median filtering	131
5.4	Canny edge detector	146
5.5	Harris corner detector	159
5.6	Enumeration of extremal regions.	161
6.1	Basic thresholding	179
6.2	Otsu's threshold detection	184
6.3	Recursive multi-spectral thresholding	186
6.4	Non-maximal suppression of directional edge data	188
6.5	Hysteresis to filter output of an edge detector	189
6.6	Inner boundary tracing	191
6.7	Outer boundary tracing	193
6.8	Extended boundary tracing	195
6.9	Border tracing in gray-level images	196
6.10	A-algorithm graph search	198
6.11	Heuristic search for image borders	205
6.12	Boundary tracing as dynamic programming	208
6.13	Curve detection using the Hough transform	212
6.14	Generalized Hough transform	217
6.15	Region forming from partial borders	219
6.16	Region merging (outline)	221
6.17	Region merging via boundary melting	223
6.18	Split and merge	226
6.19	Split and link to the segmentation tree	227
6.20	Single-pass split-and-merge	228
6.21	Efficient watershed segmentation	230
6.22	Removal of small image regions	232
7.1	Mean shift mode detection	259
7.2	Mean shift discontinuity-preserving filtering	261
7.3	Mean shift image segmentation	262
7.4	Absolute fuzzy connectivity segmentation	282

xxii	List of algorithms	
7.5	Fuzzy object extraction	284
7.6	Fuzzy object extraction with preset connectedness	285
7.7	Graph cut segmentation	300
7.8	Optimal surface segmentation	304
7.9	Multiple optimal surface segmentation	311
8.1	4-neighborhood and 8-neighborhood region labeling	333
8.2	Region identification in run length encoded data	335
8.3	Quadtree region identification	335
8.4	Curvature estimation – HK2003 algorithm	338
8.5	Calculating area in quadtrees	354
8.6	Region area calculation from Freeman 4-connectivity chain code representation	354
8.7	Region convex hull construction	361
8.8	Simple polygon convex hull detection	363
8.9	Fully parallel skeleton by thinning – MB algorithm	366
8.10	Fully parallel skeleton by thinning – MB2 algorithm	367
8.11	Region graph construction from skeleton	370
9.1	Minimum distance classifier learning and classification	393
9.2	Nearest neighbor search with K-D trees	394
9.3	Learning and classification by estimating normal distribution probability densities	400
9.4	Support vector machine learning and classification	404
9.5	<i>K</i> -means cluster analysis	406
9.6	Back-propagation learning	410
9.7	Unsupervised learning of the Kohonen feature map	411
9.8	Recognition using a Hopfield net	413
9.9	Syntactic recognition	414
9.10	Graph isomorphism	424
9.11	Maximal clique location	425
9.12	Genetic algorithm	429
9.13	Simulated annealing optimization	431
9.14	Fuzzy system design	438
9.15	AdaBoost	440
9.16	AdaBoost-MH	441
10.1	Bottom-up control	467
10.2	Coronary border detection—a combined control strategy	471
10.3	Non-hierarchical control	472
10.4	Scale Invariant Feature Transform—SIFT	476
10.5	Random sample consensus for model fitting—RANSAC	478
10.6	Approximate alignment of similar training shapes	482
10.7	Fitting an ASM	487
10.8	AAM construction	492
10.9	Active appearance model matching	495
10.10	Contextual image classification	507
10.11	Recursive contextual image classification	508
10.12	HOG object detection and localization	510
10.13	Classifier generation for Viola-Jones face detection	514
10.14	Discrete relaxation	527

10.15	Probabilistic relaxation	529
10.16	Updating a region adjacency graph and dual to merge two regions	532
10.17	Semantic region merging	533
10.18	Genetic image segmentation and interpretation	537
10.19	Gaussian mixture parameters via expectation-maximization	559
10.20	Expectation-maximization (a generalization of Algorithm 10.19)	562
10.21	Baum-Welch training for HMMs (the forward-backward algorithm)	564
11.1	Image rectification	618
11.2	PMF stereo correspondence	626
11.3	Reconstructing shape from shading	633
12.1	Line labeling	657
12.2	Iterative closest reciprocal points	668
15.1	Auto-correlation texture description	750
15.2	Co-occurrence method of texture description	753
15.3	Edge-frequency texture description	754
15.4	Primitive-length texture description	756
15.5	Shape chain grammar texture synthesis	770
15.6	Texture primitive grouping	774
16.1	Relaxation computation of optical flow from dynamic image pairs	796
16.2	Optical flow computation from an image sequence	796
16.3	General Lucas–Kanade tracking	806
16.4	Velocity field computation from two consecutive images	808
16.5	Background maintenance by median filtering	816
16.6	Background maintenance by Gaussian mixtures	818
16.7	Kernel-based object tracking	823
16.8	Condensation (particle filtering)	837
16.9	Tracking-Learning-Detection—TLD	842

Preface

Image processing, analysis, and machine vision are an exciting and dynamic part of cognitive and computer science. Following an explosion of interest during the 1970s and 1980s, subsequent decades were characterized by a maturing of the field and significant growth of active applications; remote sensing, technical diagnostics, autonomous vehicle guidance, biomedical imaging (2D, 3D, and 4D) and automatic surveillance are the most rapidly developing areas. This progress can be seen in an increasing number of software and hardware products on the market—as a single example of many, the omnipresence of consumer-level digital cameras, each of which depends on a sophisticated chain of embedded consumer-invisible image processing steps performed in real time, is striking. Reflecting this continuing development, the number of digital image processing and machine vision courses offered at universities worldwide continues to increase rapidly.

There are many texts available in the areas we cover—a lot of them are referenced in this book. The subject suffers, however, from a shortage of texts which are ‘complete’ in the sense that they are accessible to the novice, of use to the educated, and up to date. Here we present the fourth edition of a text first published in 1993. We include many of the very rapid developments that have taken and are still taking place, which quickly age some of the very good textbooks produced in the recent past.

Our target audience spans the range from the undergraduate with negligible experience in the area through to the Master’s, Ph.D., and research student seeking an advanced springboard in a particular topic. The entire text has been updated since the third version (particularly with respect to most recent development and associated references). We retain the same Chapter structure, but many sections have been rewritten or introduced as new. Among the new topics are the Radon transform, a unified approach to image/template matching, efficient object skeletonization (MB and MB2 algorithms), nearest neighbor classification including BBF/FLANN, histogram-of-oriented-Gaussian (HOG) approach to object detection, random forests, Markov random fields, Bayesian belief networks, scale invariant feature transform (SIFT), recent 3D image analysis/vision development, texture description using local binary patterns, and several point tracking approaches for motion analysis. Approaches to 3D vision evolve especially quickly and we have revised this material and added new comprehensive examples. In addition, several sections have been rewritten or expanded in response to reader and reviewer comments. All in all, about 15% of this edition consists of newly written material presenting state-of-the-art methods and techniques that already have proven their importance in the field; additionally, the whole text has been edited for currency and to correct a small number of oversights detected in the previous edition.

In response to demand, we have re-incorporated exercises (both short-form questions, and longer problems frequently requiring practical usage of computer tools and/or

development of application programs) into this text. These re-use the valuable practical companion text to the third edition [Svoboda et al., 2008], but also cover material that was not present in earlier editions. The companion text provides Matlab-based implementations, introduces additional problems, explains steps leading to solutions, and provides many useful linkages to allow practical use: a Solution Manual is available via the Cengage secure server to registered instructors. In preparing this edition, we gratefully acknowledge the help and support of many people, in particular our reviewers Saeid Belkasim, Georgia State University, Thomas C. Henderson, University of Utah, William Hoff, Colorado School of Mines, Lina Karam, Arizona State University, Peter D. Scott, the University at Buffalo, SUNY and Jane Zhang, California Polytechnic State University. Richard W. Penney, Worcestershire, UK gave close attention to our third edition which has permitted the correction of many shortcomings. At our own institutions, Reinhard Beichel, Gary Christensen, Hannah Dee, Mona Garvin, Ian Hales, Sam Johnson, Derek Magee, Ipek Oguz, Kalman Palagyi, Andrew Rawlins, Joe Reinhardt, Punam Saha, and Xiaodong Wu have been a constant source of feedback, inspiration and encouragement.

This book reflects the authors' experience in teaching one- and two-semester undergraduate and graduate courses in Digital Image Processing, Digital Image Analysis, Image Understanding, Medical Imaging, Machine Vision, Pattern Recognition, and Intelligent Robotics at their respective institutions. We hope that this combined experience will give a thorough grounding to the beginner and provide material that is advanced enough to allow the more mature student to understand fully the relevant areas of the subject. We acknowledge that in a very short time the more active areas will have moved beyond this text.

This book could have been arranged in many ways. It begins with low-level processing and works its way up to higher levels of image interpretation; the authors have chosen this framework because they believe that image understanding originates from a common database of information. The book is formally divided into 16 chapters, beginning with low-level processing and working toward higher-level image representation, although this structure will be less apparent after Chapter 12, when we present mathematical morphology, image compression, texture, and motion analysis which are very useful but often special-purpose approaches that may not always be included in the processing chain.

Decimal section numbering is used, and equations and figures are numbered within each chapter. Each chapter is supported by an extensive list of references and exercises. A selection of algorithms is summarized formally in a manner that should aid implementation. Not all the algorithms discussed are presented in this way (this might have doubled the length of the book); we have chosen what we regard as the key, or most useful or illustrative, examples for this treatment. Each chapter further includes a concise Summary section, Short-answer questions, and Problems/Exercises.

Chapters present material from an introductory level through to an overview of current work; as such, it is unlikely that the beginner will, at the first reading, expect to absorb all of a given topic. Often it has been necessary to make reference to material in later chapters and sections, but when this is done an understanding of material in hand will not depend on an understanding of that which comes later. It is expected that the more advanced student will use the book as a reference text and signpost to current activity in the field—we believe at the time of going to press that the reference list is full in its indication of current directions, but record here our apologies to any work we have overlooked. The serious reader will note that the reference list contains citations

of both the classic material that has survived the test of time as well as references that are very recent and represent what the authors consider promising new directions. Of course, before long, more relevant work will have been published that is not listed here.

This is a long book and therefore contains material sufficient for much more than one course. Clearly, there are many ways of using it, but for guidance we suggest an ordering that would generate five distinct modules:

Digital Image Processing I, an undergraduate course.

Digital Image Processing II, an undergraduate/graduate course, for which Digital Image Processing I may be regarded as prerequisite.

Computer Vision I, an undergraduate/graduate course, for which Digital Image Processing I may be regarded as prerequisite.

Computer Vision II, a graduate course, for which Computer Vision I may be regarded as prerequisite.

Image Analysis and Understanding, a graduate course, for which Computer Vision I may be regarded as prerequisite.

The important parts of a course, and necessary prerequisites, will naturally be specified locally; a suggestion for partitioning the contents follows this Preface.

Assignments should wherever possible make use of existing software; it is our experience that courses of this nature should not be seen as ‘programming courses’, but it is the case that the more direct practical experience the students have of the material discussed, the better is their understanding. Since the first edition was published, an explosion of web-based material has become available, permitting many of the exercises we present to be conducted without the necessity of implementing from scratch. We do not present explicit pointers to Web material, since they evolve so quickly; however, pointers to specific support materials for this book and others may be located via the designated book web page, <http://www.cengage.com/engineering>.

In addition to the print version, this textbook is also available online through **MindTap**, a personalized learning program. If you purchase the *MindTap* version of this book, you will obtain access to the book’s *MindTap Reader* and will be able to complete assignments online. If your class is using a *Learning Management System* (such as *Blackboard*, *Moodle*, or *Angel*) for tracking course content, assignments, and grading, you can seamlessly access the *MindTap* suite of content and assessments for this course. In *MindTap*, instructors can:

- Personalize the learning path to match the course syllabus by rearranging content, hiding sections, or appending original material to the textbook content.
- Connect a *Learning Management System* portal to the online course and Reader.
- Customize online assessments and assignments.
- Track student progress and comprehension with the Progress application.
- Promote student engagement through interactivity and exercises.

Additionally, students can listen to the text through *ReadSpeaker*, take notes and highlight content for easy reference, as well as self-check their understanding of the material.

The book has been prepared using the L^AT_EX text processing system. Its completion would have been impossible without extensive usage of the Internet computer network and electronic mail. We would like to acknowledge the University of Iowa, the Czech Technical University, the Department of Computer Science at Prifysgol Aberystwyth, and the School of Computing at the University of Leeds for providing the environment in which this book was born and re-born.

Milan Sonka is Director of the Iowa Institute for Biomedical Imaging, Professor/Chair of Electrical & Computer Engineering, and Professor of Ophthalmology & Visual Sciences and Radiation Oncology at the University of Iowa, Iowa City, Iowa, USA. His research interests include medical image analysis, computer-aided diagnosis, and machine vision. Václav Hlaváč is Professor of Cybernetics at the Czech Technical University, Prague. His research interests are knowledge-based image analysis, 3D model-based vision and relations between statistical and structural pattern recognition. Roger Boyle very recently retired from the School of Computing at the University of Leeds, England, where he had been Head. His research interests are in low-level vision and pattern recognition, and he now works within the UK National Phenomics Centre at Prifysgol Aberystwyth, Cymru.

All authors have contributed throughout—the ordering on the cover corresponds to the weight of individual contribution. Any errors of fact are the joint responsibility of all.

Final typesetting has been the responsibility of Hrvoje Bogunović at the University of Iowa. This fourth collaboration has once more jeopardized domestic harmony by consuming long periods of time; we remain very happy to invest more work in this text in response to readers' comments.

References

Svoboda T., Kybic J., and Hlavac V. *Image Processing, Analysis, and Machine Vision: A MATLAB Companion*. Thomson Engineering, 2008.

Milan Sonka

*The University of Iowa
Iowa City, Iowa, USA*

milan-sonka@uiowa.edu

<http://www.engineering.uiowa.edu/~sonka>



Václav Hlaváč

*Czech Technical University
Prague, Czech Republic*

hlavac@cmp.felk.cvut.cz

<http://cmp.felk.cvut.cz/~hlavac>



Roger Boyle

*Prifysgol Aberystwyth
United Kingdom*

rogerdboyle@gmail.com

<http://users.aber.ac.uk/rob21/>



Possible course outlines

Here, one possible ordering of the material covered in the five courses proposed in the Preface is given. This should not, of course, be considered the only option—on the contrary, the possibilities for organizing Image Processing and Analysis courses are practically endless. Therefore, what follows shall only be regarded as suggestions, and instructors shall tailor content to fit the assumed knowledge, abilities, and needs of the students enrolled.

Figure 1 shows course pre-requisite dependencies of the proposed ordering. Figure 2 shows the mapping between the proposed course outlines and the material covered in the individual chapters and sections.

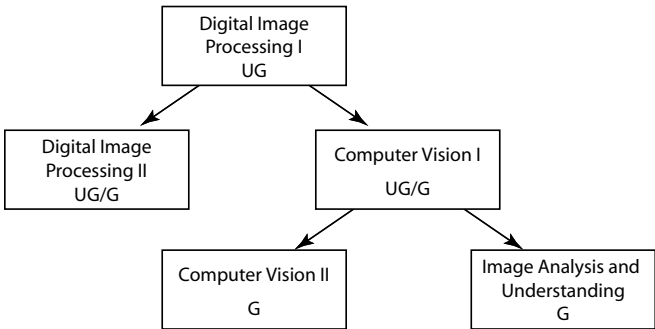


Figure 1: Pre-requisite dependencies of the proposed five courses. UG = undergraduate course, G = graduate course. © Cengage Learning 2015.

	chapter															
course	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DIP I	■	■	■	■	■	■								■	■	
DIP II	■		■	■		■	■						■	■		
CV I	■	■				■	■	■	■	■	■				■	
CV II			■	■		■					■	■	■			■
IAU			■				■		■	■	■					■

Figure 2: Mapping between the proposed course outlines and material covered in individual chapters and sections. See course outlines for details. © Cengage Learning 2015.

Digital Image Processing I (DIP I)

An undergraduate course.

- 1 Introduction
- 2 The image, its representation and properties
 - 2.1 Image representations
 - 2.2 Image digitization
 - 2.3 Digital image properties
- 4 Data structures for image analysis
- 5 Image pre-processing
 - 5.1 Pixel brightness transformations
 - 5.2 Geometric transformations
 - 5.3 Local pre-processing (except 5.3.6–5.3.7, 5.3.9–5.3.11, limited coverage of 5.3.4, 5.3.5)
 - 5.4 Image restoration (except 5.4.3)
- 6 Segmentation I
 - 6.1 Thresholding (except 6.1.3)
 - 6.2 Edge-based segmentation (except 6.2.4, 6.2.5, 6.2.7, 6.2.8)
 - 6.3 Region growing segmentation (except 6.3.4)
 - 6.4 Matching
 - 6.5 Evaluation issues in segmentation
- 3 The image, its mathematical and physical background
 - 3.2 Linear integral transforms (3.2.1–3.2.4, 3.2.6 only)
- 14 Image data compression (except wavelet compression, except 14.9)
Practical image processing projects

Digital Image Processing II (DIP II)

An undergraduate/graduate course, for which Digital Image Processing I may be regarded as prerequisite.

- 1 Introduction (brief review)
- 2 The image, its representation and properties
 - 2.4 Color images
 - 2.5 Cameras
- 3 The image, its mathematical and physical background (except 3.2.8–3.2.10)

- 5 Image pre-processing
 - 5.3.4 Scale in image processing
 - 5.3.5 Canny edge detection
 - 5.3.6 Parametric edge models
 - 5.3.7 Edges in multi-spectral images
 - 5.3.8 Pre-processing in frequency domain
 - 5.3.9 Line detection
 - 5.3.10 Corner detection
 - 5.3.11 Maximally stable extremal regions
 - 5.4 Image restoration
- 6 Segmentation I
 - 6.1 Thresholding – considering color image data
 - 6.2.1 Edge image thresholding – considering color image data
 - 6.3.1–3 Region-based segmentation – considering color image data
 - 6.4 Matching – considering color image data
- 14 Image compression
 - 14.2 Discrete image transforms in image compression
 - 14.9 JPEG and MPEG
- 13 Mathematical morphology
 - Practical image processing projects

Computer Vision I (CV I)

An undergraduate/graduate course, for which Digital Image Processing I may be regarded as prerequisite.

- 1 Introduction (brief review)
- 2 The image, its representation and properties (brief review)
- 6 Segmentation I
 - 6.2.4 Border detection as graph searching
 - 6.2.5 Border detection as dynamic programming
 - 6.2.7 Border detection using border location information
 - 6.2.8 Region construction from borders
 - 6.3.4 Watershed segmentation
- 7 Segmentation II

xxxiv Possible course outlines

7.1 Mean shift segmentation

7.2 Active contour models

8 Shape representation and description

9 Object recognition

9.1 Knowledge representation

9.2 Statistical pattern recognition (except 9.2.5)

9.3 Neural networks

9.4 Syntactic pattern recognition

10 Image understanding

10.1 Image understanding control strategies

10.2 SIFT

10.3 RANSAC

10.6 Pattern recognition methods in image understanding

10.9 Scene labeling

10.10 Semantic image segmentation and understanding

15 Texture

Practical computer vision projects

Computer Vision II (CV II)

A graduate course, for which Computer Vision I may be regarded as prerequisite.

2 The image, its representation and properties

2.4 Color images

2.5 Cameras

3 The image, its mathematical and physical background

3.4 Image formation physics

5 Image pre-processing

5.3.4 Scale in image processing

5.3.5 Canny edge detection

5.3.6 Parametric edge models

5.3.7 Edges in multi-spectral images

5.3.9 Line detection

5.3.10 Corner detection

5.3.11 Maximally stable extremal regions

11 3D Vision, geometry and radiometry**12** Use of 3D vision**16** Motion analysis

Practical 3D vision projects

Image Analysis and Understanding (IAU)

A graduate course, for which Computer Vision I may be regarded as prerequisite.

7 Segmentation II (except 7.1, 7.2)**9** Object recognition

9.2.5 Support vector machines

9.5 Recognition as graph matching

9.6 Optimization techniques in recognition

9.7 Fuzzy systems

9.8 Boosting in pattern recognition

9.9 Random forests

3 The image, its mathematical and physical background

3.2.8 Eigen analysis

3.2.9 Singular value decomposition

3.2.10 Principal component analysis

10 Image understanding

10.1 Image understanding control strategies

10.4 Point distribution models

10.5 Active appearance models

10.7 Boosted cascade of classifiers

10.8 Image understanding using random forests

10.11 Hidden Markov models

10.12 Markov random fields

10.13 Gaussian mixture models and expectation maximization

16 Motion analysis

Practical image understanding projects

Chapter 1

Introduction

1.1 Motivation

Vision allows humans to perceive and understand the world surrounding them, while computer vision aims to duplicate the effect of human vision by electronically perceiving and understanding an image. Books other than this one would dwell at length on this sentence and the meaning of the word ‘duplicate’—whether computer vision is *simulating* or *mimicking* human systems is philosophical territory, and very fertile territory too.

Giving computers the ability to see is not an easy task—we live in a three-dimensional (3D) world, and when computers try to analyze objects in 3D space, the visual sensors available (e.g., TV cameras) usually give two-dimensional (2D) images, and this projection to a lower number of dimensions incurs an enormous loss of information. Sometimes, equipment will deliver images that are 3D but this may be of questionable value: analyzing such datasets is clearly more complicated than 2D, and sometimes the ‘three-dimensionality’ is less than intuitive to us ... terahertz scans are an example of this. Dynamic scenes such as those to which we are accustomed, with moving objects or a moving camera, are increasingly common and represent another way of making computer vision more complicated.

Figure 1.1 could be witnessed in thousands of farmyards in many countries, and serves to illustrate just some of the problems that we will face.



Figure 1.1: A frame from a video of a typical farmyard scene: the cow is one of a number walking naturally from right to left. *Courtesy of D. R. Magee, University of Leeds.*

2 Chapter 1: Introduction

There are many reasons why we might wish to study scenes such as this, which are attractively simple *to us*. The beast is moving slowly, it is clearly black and white, its movement is rhythmic, etc.; however, automated analysis is very fraught—in fact, the animal’s boundary is often very difficult to distinguish clearly from the background, the motion of the legs is self-occluding and (subtly) the concept of ‘cow-shaped’ is not something easily encoded. The application from which this picture was taken¹ made use of many of the algorithms presented in this book: starting at a low level, moving features were identified and grouped. A ‘training phase’ taught the system what a cow might look like in various poses (see Figure 1.2), from which a model of a ‘moving’ cow could be derived (see Figure 1.3).

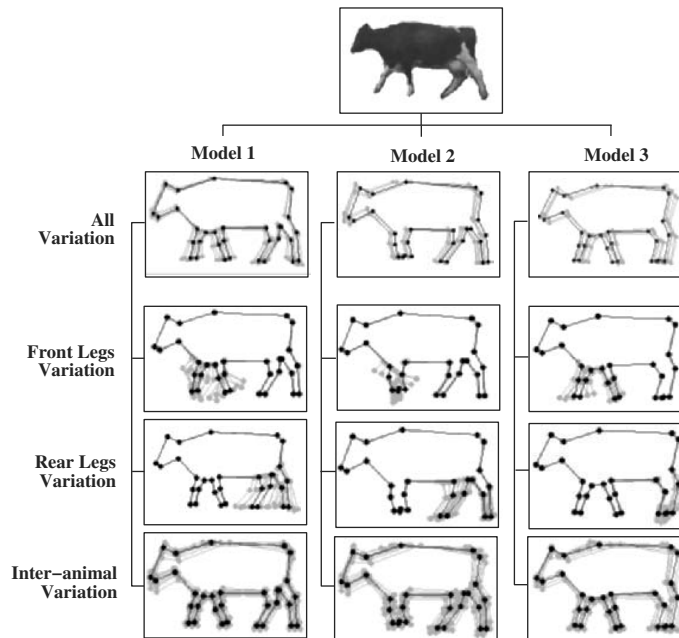


Figure 1.2: Various models for a cow silhouette: a straight-line boundary approximation has been learned from training data and is able to adapt to different animals and different forms of occlusion. *Courtesy of D. R. Magee, University of Leeds.*

These models could then be fitted to new (‘unseen’) video sequences. Crudely, at this stage anomalous behavior such as lameness could be detected by the model failing to fit properly, or well.

Thus we see a sequence of operations—image capture, early processing, segmentation, model fitting, motion prediction, qualitative/quantitative conclusion—that is characteristic of image understanding and computer vision problems. Each of these phases (which may not occur sequentially!) may be addressed by a number of algorithms which we shall cover in due course.

¹The application was serious; there is a growing need in modern agriculture for automatic monitoring of animal health, for example to spot lameness. A limping cow is trivial for a human to identify, but it is very challenging to do this automatically.



Figure 1.3: Three frames from a cow sequence: notice the model can cope with partial occlusion as the animal enters the scene, and the different poses exhibited. *Courtesy of D. R. Magee, University of Leeds.*

This example is relatively simple to explain, but serves to illustrate that many computer vision techniques use the results and methods of mathematics, pattern recognition, artificial intelligence (AI), psycho-physiology, computer science, electronics, and other scientific disciplines.

Why is computer vision hard? As an exercise, consider a single gray-scale (monochromatic) image: put the book down and before proceeding write down a few reasons why you feel automatic inspection and analysis of it may be difficult.

1.2 Why is computer vision difficult?

This philosophical question provides some insight into the complex landscape of computer vision. It can be answered in many ways: we briefly offer six—most of them will be discussed in more detail later in the book.

Loss of information in 3D $\rightarrow$ 2D is a phenomenon which occurs in typical image capture devices such as a camera or an eye. Their geometric properties have been approximated by a pinhole model for centuries (a box with a small hole in it—a ‘camera obscura’ in Latin). This physical model corresponds to a mathematical model of perspective projection; Figure 1.4 summarizes the principle. The projective transformation maps points along rays but does not preserve angles and collinearity.

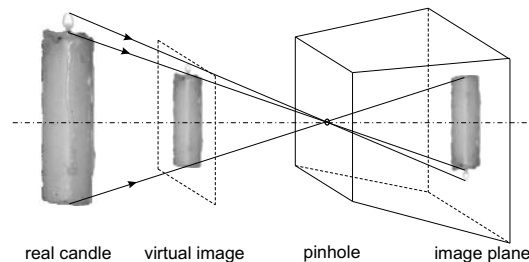


Figure 1.4: The pinhole model of imaging geometry does not distinguish size of objects. © Cengage Learning 2015.

The main trouble with the pinhole model and a single available view is that the projective transformation sees a small object close to the camera in the same way as

4 Chapter 1: Introduction

a big object remote from the camera. In this case, a human needs a ‘yardstick’ to guess the actual size of the object which the computer does not have.

Interpretation of image(s) is a problem humans solve unwittingly that is the principal tool of computer vision. When a human tries to understand an image then previous knowledge and experience is brought to the current observation. Human ability to reason allows representation of long-gathered knowledge, and its use to solve new problems. Artificial intelligence has worked for decades to endow computers with the capability to understand observations; while progress has been tremendous, the practical ability of a machine to understand observations remains very limited.

From the mathematical logic and/or linguistics point of view, image interpretation can be seen as a mapping

$$\text{interpretation: image data} \longrightarrow \text{model}.$$

The (logical) model means some specific world in which the observed objects make sense. Examples might be nuclei of cells in a biological sample, rivers in a satellite image, or parts in an industrial process being checked for quality. There may be several interpretations of the same image(s). Introducing interpretation to computer vision allows us to use concepts from mathematical logic, linguistics as syntax (rules describing correctly formed expressions), and semantics (study of meaning). Considering observations (images) as an instance of formal expressions, semantics studies relations between expressions and their meanings. The interpretation of image(s) in computer vision can be understood as an instance of semantics.

Practically, if the image understanding algorithms know into which particular domain the observed world is constrained, then automatic analysis can be used for complicated problems.

Noise is inherently present in each measurement in the real world. Its existence calls for mathematical tools which are able to cope with uncertainty; an example is probability theory. Of course, more complex tools make the image analysis much more complicated compared to standard (deterministic) methods.

Too much data. Images are big, and video—increasingly the subject of vision applications—correspondingly bigger. Technical advances make processor and memory requirements much less of a problem than they once were, and much can be achieved with consumer level products. Nevertheless, efficiency in problem solutions is still important and many applications remain short of real-time performance.

Brightness measured in images is given by complicated image formation physics. The radiance ($\approx$ brightness, image intensity) depends on the irradiance (light source type, intensity and position), the observer’s position, the surface local geometry, and the surface reflectance properties. The inverse tasks are ill-posed—for example, to reconstruct local surface orientation from intensity variations. For this reason, image-capture physics is usually avoided in practical attempts at image understanding. Instead, a direct link between the appearance of objects in scenes and their interpretation is sought.

Local window vs. need for global view. Commonly, image analysis algorithms analyze a particular storage bin in an operational memory (e.g., a pixel in the image) and its local neighborhood; the computer sees the image through a keyhole; this makes it very difficult to understand more global context. This problem has a long tradition in

artificial intelligence: in the 1980s McCarthy argued that formalizing context was a crucial step toward the solution of the problem of generality. It is often very difficult to interpret an image if it is seen only locally or if only a few local keyholes are available. Figure 1.5 illustrates this pictorially. How context is taken into account is an important facet of image analysis.

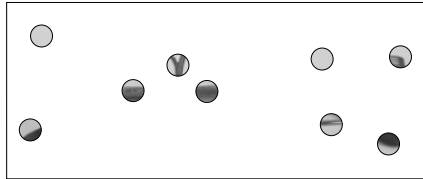


Figure 1.5: Illustration of the world seen through several keyholes providing only a local context. It is very difficult to guess what object is depicted; the complete image is shown in Figure 1.6. © Cengage Learning 2015.

1.3 Image representation and image analysis tasks

Image understanding by a machine can be seen as an attempt to find a relation between input image(s) and previously established models of the observed world. Transition from the input image(s) to the model reduces the information contained in the image to relevant information for the application domain. This process is usually divided into several steps and several levels representing the image are used. The bottom layer contains raw image data and the higher levels interpret the data. Computer vision designs these intermediate representations and algorithms serving to establish and maintain relations between entities within and between layers.

Image representation can be roughly divided according to data organization into four levels, see Figure 1.7. The boundaries between individual levels are inexact, and more detailed divisions are also proposed in the literature. Figure 1.7 suggests a bottom up approach, from signals with almost no abstraction, to the highly abstract description needed for image understanding. Note that the flow of information does not need to be unidirectional; often feedback loops are introduced which allow the modification of algorithms according to intermediate results.

This hierarchy of image representation and related algorithms is frequently categorized in an even simpler way—*low-level* image processing and *high-level* image understanding.

Low-level processing methods usually use very little knowledge about the content of images. In the case of the computer knowing image content, it is usually provided by high-level algorithms or directly by a human who understands the problem domain. Low-level methods may include image compression, pre-processing methods for noise filtering, edge extraction, and image sharpening, all of which we shall discuss in this book. Low-level image processing uses data which resemble the input image; for example, an input image captured by a TV camera is 2D in nature, being described by an image function $f(x, y)$ whose value, at simplest, is usually brightness depending on the co-ordinates x, y of the location in the image.

If the image is to be processed using a computer it will be digitized first, after which it may be represented by a rectangular matrix with elements corresponding to the brightness at appropriate image locations. More probably, it will be presented in color, implying (usually) three channels: red, green and blue. Very often, such a data set will

6 Chapter 1: Introduction

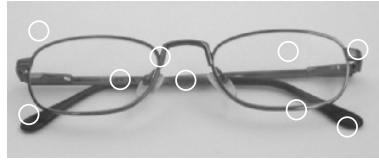


Figure 1.6: It is easy for humans to interpret an image if it is seen globally: compare to Figure 1.5.
© Cengage Learning 2015.

be part of a video stream with an associated frame rate. Nevertheless, the raw material will be a set or sequence of matrices which represent the inputs and outputs of low-level image processing.

High-level processing is based on knowledge, goals, and plans of how to achieve those goals, and artificial intelligence methods are widely applicable. High-level computer vision tries to imitate human cognition (although be mindful of the health warning given in the very first paragraph of this chapter) and the ability to make decisions according to the information contained in the image. In the example described, high-level knowledge would be related to the ‘shape’ of a cow and the subtle interrelationships between the different parts of that shape, and their (inter-)dynamics.

High-level vision begins with some form of formal model of the world, and then the ‘reality’ perceived in the form of digitized images is compared to the model. A match is attempted, and when differences emerge, partial matches (or subgoals) are sought that overcome them; the computer switches to low-level image processing to find information needed to update the model. This process is then repeated iteratively, and ‘understanding’ an image thereby becomes a co-operation between top-down and bottom-up processes. A feedback loop is introduced in which high-level partial results create tasks for low-level image processing, and the iterative image understanding process should eventually converge to the global goal.

Computer vision is expected to solve very complex tasks, the goal being to obtain similar results to those provided by biological systems. To illustrate the complexity of these tasks, consider Figure 1.8 in which a particular image representation is presented—

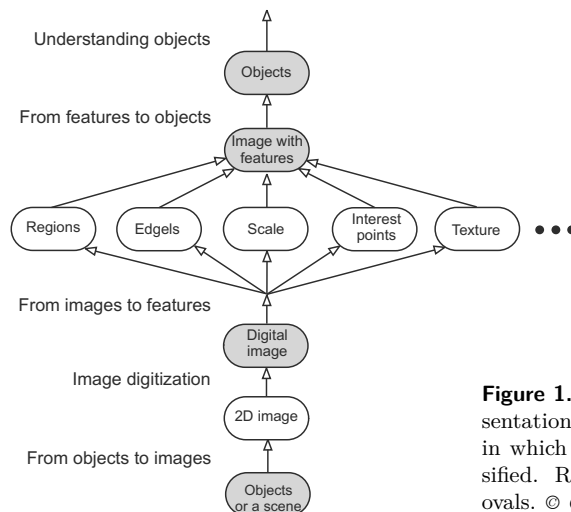


Figure 1.7: Four possible levels of image representation suitable for image analysis problems in which objects have to be detected and classified. Representations are depicted as shaded ovals. © Cengage Learning 2015.

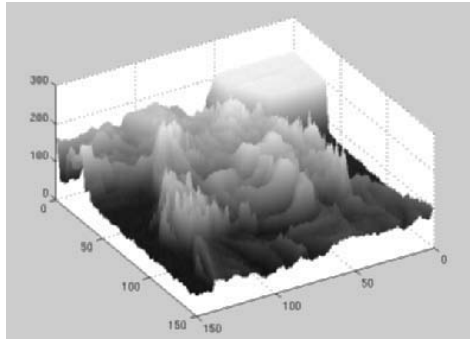


Figure 1.8: An unusual image representation. © R.D. Boyle 2015.

the value on the vertical axis gives the brightness of its corresponding location in the [gray-scale] image. Consider what this image might be before looking at Figure 1.9, which is a rather more common representation of the same image.

Both representations contain exactly the same information, but for a human observer it is very difficult to find a correspondence between them, and without the second, it is unlikely that one would recognize the face of a child. The point is that a lot of a priori knowledge is used by humans to interpret the images; the machine only begins with an array of numbers and so will be attempting to make identifications and draw conclusions from data that to us are more like Figure 1.8 than Figure 1.9. Increasingly, data capture equipment is providing very large data sets that do *not* lend themselves to straightforward interpretation by humans—we have already mentioned terahertz imaging as an example. Internal image representations are not directly understandable—while the computer is able to process local parts of the image, it is difficult for it to locate global knowledge. General knowledge, domain-specific knowledge, and information extracted from the image will be essential in attempting to ‘understand’ these arrays of numbers.

Low-level computer vision techniques overlap almost completely with digital image processing, which has been practiced for decades. The following sequence of processing steps is commonly seen: An image is captured by a sensor (such as a camera) and digitized; then the computer suppresses noise (image pre-processing) and maybe enhances some object features which are relevant to understanding the image. Edge extraction is an example of processing carried out at this stage.

Image segmentation is the next step, in which the computer tries to separate objects from the image background and from each other. Total and partial segmentation may be distinguished; total segmentation is possible only for very simple tasks, an example being the recognition of dark non-touching objects from a light background. For example, in analyzing images of printed text (an early step in optical character recognition, OCR) even this superficially simple problem is very hard to solve without error. In more complicated problems (the general case), low-level image processing techniques handle the partial segmentation tasks, in which only the cues which will aid further high-level processing are extracted. Often, finding parts of object boundaries is an example of low-level partial segmentation.

Object description and classification in a totally segmented image are also understood as part of low-level image processing. Other low-level operations are image compression, and techniques to extract information from (but not *understand*) moving scenes.



Figure 1.9: Another representation of Figure 1.8.
© R.D. Boyle 2015.

Low-level image processing and high-level computer vision differ in the data used. Low-level data are comprised of original images represented by matrices composed of brightness (or similar) values, while high-level data originate in images as well, but only those data which are relevant to high-level goals are extracted, reducing the data quantity considerably. High-level data represent knowledge about the image content—for example, object size, shape, and mutual relations between objects in the image. High-level data are usually expressed in symbolic form.

Many low-level image processing methods were proposed in the 1970s or earlier: research is trying to find more efficient and more general algorithms and is implementing them on more technologically sophisticated equipment, in particular, parallel machines (including GPU's) are being used to ease the computational load. The requirement for better and faster algorithms is fuelled by technology delivering larger images (better spatial or temporal resolution), and color.

A complicated and so far unsolved problem is how to order low-level steps to solve a specific task, and the aim of automating this problem has not yet been achieved. It is usually still a human operator who finds a sequence of relevant operations, and domain-specific knowledge and uncertainty cause much to depend on this operator's intuition and previous experience.

High-level vision tries to extract and order image processing steps using all available knowledge—image understanding is the heart of the method, in which feedback from high-level to low-level is used. Unsurprisingly this task is very complicated and computationally intensive. David Marr's book [Marr, 1982], discussed in Section 11.1.1, influenced computer vision considerably throughout the 1980s; it described a new methodology and computational theory inspired by biological vision systems. Developments in the 1990s moved away from dependence on this paradigm, but interest in properly understanding and then modeling human visual (and other perceptual) systems persists—it remains the case that the only known solution to the 'vision problem' is our own brain!

Consider *3D vision problems* for a moment. We adopt the user's view, i.e., what tasks performed routinely by humans would be good to accomplish by machines. What is the relation of these 3D vision tasks to low-level (image processing) and high-level (image analysis) algorithmic methods? There is no widely accepted view in the academic community. Links between (algorithmic) components and representation levels are tailored to the specific application solved, e.g., navigation of an autonomous vehicle. These applications have to employ specific knowledge about the problem solved to be competitive with tasks which humans solve. Many researchers in different fields work on related

problems and research in ‘cognitive systems’ could be the key which may disentangle the complicated world of perception which includes also computer vision.

Figure 1.10 depicts several 3D vision tasks and algorithmic components expressed on different abstraction levels. In most cases, the bottom-up and top-down approach is adopted to fulfill the task.

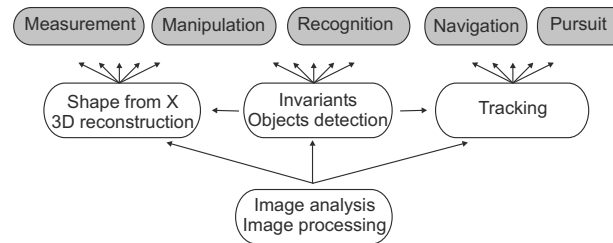


Figure 1.10: Several 3D computer vision tasks from the user’s point of view are on the upper line (filled). Algorithmic components on different hierarchical levels support it in a bottom-up fashion. © Cengage Learning 2015.

1.4 Summary

- Human vision is natural and seems easy; computer mimicry of this is difficult.
- We might hope to examine pictures, or sequences of pictures, for quantitative and qualitative analysis.
- Many standard and advanced AI techniques are relevant.
- ‘High’ and ‘low’ levels of computer vision can be identified.
- Processing moves from digital manipulation, through pre-processing, segmentation, and recognition to understanding—but these processes may be simultaneous and co-operative.
- An understanding of the notions of heuristics, a priori knowledge, syntax, and semantics is necessary.
- The vision literature is large and growing; books may be specialized, elementary, or advanced.
- A knowledge of the research literature is necessary to stay up to date with the topic.
- Developments in electronic publishing and the Internet are making access to vision simpler.

1.5 Exercises

Short-answer questions

- S1.1 What is the difference between image analysis (or computer vision) on one side and computer graphics on the other side?
- S1.2 Walking around your local town, locate five instances in which digital images are now publicly visible or in use where they would not have been 10–15 years ago.
- S1.3 For some or all of the instances identified in Exercise S1.2, estimate the economics of the application—the size of market implied and the cost of the unit.
- S1.4 Determine the current cost of domestic video cameras and storage, and image-capture hardware for home computers. Try to determine the cost of comparable equipment 1, 5, and 10 years ago. Make some predictions about the power and price of comparable equipment in 1, 5, and 10-year time.
- S1.5 In your own, or a local, university, determine how long image processing, image analysis, or computer vision have been taught. Try to determine the major syllabus or course offering changes over the last 1, 5, and (if possible) 10 years.
- S1.6 Locate a topic area of computer science outside vision in which *heuristics* are commonly used.
- S1.7 Locate a topic area of computer science outside vision in which *a priori information* is commonly used.
- S1.8 Locate a topic area of computer science outside vision in which *syntax* is commonly used.
- S1.9 Locate a topic area of computer science outside vision in which *semantics* are commonly used.
- S1.10 In a technical or academic library, locate the section on computer vision. Examine the texts there, noting the date of publication and chapter titles; compare them with those of this text. Draw some conclusions about the topics (or, at least, their titles) that are static, and those which have developed recently.
- S1.11 Select *at random* some journal references from the listings at the end of each chapter in this book. Construct a histogram of publication year.
- S1.12 Locate in a library some of the references selected in Exercise S1.11; study the publication dates of references listed there, and draw another histogram of years. What do this exercise and Exercise S1.11 tell you about the development of computer vision?
- S1.13 Make yourself familiar with solved problems and **Matlab** implementations of selected algorithms provided in the corresponding chapter of the **Matlab Companion** to this text [Svoboda et al., 2008]. The **Matlab Companion** homepage <http://visionbook.felk.cvut.cz> offers images used in the problems, and well-commented **Matlab** code is provided for educational purposes.
- S1.14 Use the **Matlab Companion** [Svoboda et al., 2008] to develop solutions to additional exercises and practical problems provided there. Implement your solutions using **Matlab** or other suitable programming languages.

1.6 References

- Marr D. *Vision—A Computational Investigation into the Human Representation and Processing of Visual Information*. Freeman, San Francisco, 1982.
- Svoboda T., Kybic J., and Hlavac V. *Image Processing, Analysis, and Machine Vision: A MATLAB Companion*. Thomson Engineering, 2008.

Chapter 2

The image, its representations and properties

This chapter and the next introduce concepts and mathematical tools which are widely used in image analysis, and will be used throughout this book. We have separated this material into essential basics (this chapter) and more intense mathematical theory (the next). This division is intended to help the reader to start practical work immediately; mathematical details may be skipped in favor of concentrating on the intuitive meaning of the basic concepts while the next chapter provides a thorough anchoring to a mathematical background. Such a division can never be perfect, and this chapter contains some forward references and dependencies on its successor.

2.1 Image representations, a few concepts

Mathematical models are often used to describe images and other signals. A signal is a function depending on some variable with physical meaning; it can be one-dimensional (e.g., dependent on time), two-dimensional (e.g., an image dependent on two co-ordinates in a plane), three-dimensional (e.g., describing a volumetric object in space), or higher-dimensional. A scalar function might be sufficient to describe a monochromatic image, while vector functions may be used to represent, for example, color images consisting of three component colors.

Functions we shall work with may be categorized as **continuous**, **discrete**, or **digital**. A continuous function has continuous domain and range; if the domain set is discrete, then we have a discrete function; if the range set is also discrete, then we have a digital function. Many of these functions will be linear, and correspondingly simple to deal with.

We shall take the usual intuitive definition of **image**—an example might be the image on the human retina, or captured by a video camera. This can be modeled by a continuous (image) function of two variables $f(x, y)$ where (x, y) are co-ordinates in a plane, or perhaps three variables $f(x, y, t)$, where t is time. This model is reasonable in the great

majority of applications that we encounter, and which are presented in this book. Nevertheless, it is worth realizing that an ‘image’ may be acquired in many ways. We shall note often that color is the norm, even when algorithms are presented for monochromatic images, but we do not need to constrain ourselves to the visible spectrum. Infra-red cameras are now very common (for example, for night-time surveillance). Other parts of the electro-magnetic [EM] spectrum may also be used; microwave imaging, for example, is becoming widely available. Further, image acquisition outside the EM spectrum is also common: in the medical domain, datasets are generated via magnetic resonance (MR), X-ray computed tomography (CT), ultrasound etc. All of these approaches generate large arrays of data requiring analysis and understanding and with increasing frequency these arrays are of 3 or more dimensions. We imply a study of all these modalities in the title of this book.

The continuous image function

The (gray-scale) image function values correspond to brightness at image points. The function value can express other physical quantities as well (temperature, pressure distribution, distance from the observer, etc.). **Brightness** integrates different optical quantities—using brightness as a basic quantity allows us to avoid the complicated process of image formation which will be discussed in Section 3.4.

The image on the retina or on a camera sensor is intrinsically two-dimensional (2D). We shall call such an image bearing information about brightness points an **intensity image**. The 2D image on the imaging sensor is commonly the result of projection of a three-dimensional (3D) scene. The simplest mathematical model for this is a pin-hole camera (see Figure 1.4).

The 2D intensity image is the result of a **perspective projection** of the 3D scene, which is modeled by the image captured by a pin-hole camera illustrated in Figure 2.1. In this figure, the image plane has been reflected with respect to the XY plane in order

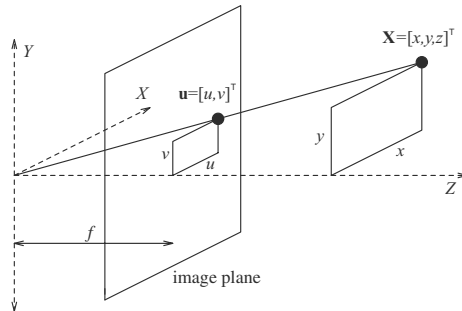


Figure 2.1: Perspective projection geometry.
© Cengage Learning 2015.

not to get a mirrored image with negative co-ordinates. The quantities x , y , and z are co-ordinates of the point $\mathbf{X}$ in a 3D scene, and f is the distance from the pinhole to the image plane. f is commonly called the focal length because in lenses it has a similar meaning. The projected point $\mathbf{u}$ has co-ordinates (u, v) in the 2D image plane, which can easily be derived from similar triangles

$$u = \frac{x f}{z}, \quad v = \frac{y f}{z}. \quad (2.1)$$

A non-linear perspective projection is often approximated by a linear **parallel** (or **orthographic**) **projection**, where $f \rightarrow \infty$. Implicitly, $z \rightarrow \infty$ says that the orthographic projection is a limiting case of the perspective projection for faraway objects.

When 3D objects are mapped into the camera plane by perspective projection, a lot of information disappears because such a transform is not one-to-one. Recognizing or reconstructing objects in a 3D scene from one image is an ill-posed problem. In Chapter 11, we shall consider more elaborate representations that attempt to recapture information about the original scene that an image depicts. As may be expected, this is not a simple task and involves intermediate representations that try to establish the **depth** of points in the image. The aim is to recover a full 3D representation such as may be used in computer graphics—that is, a representation independent of the viewpoint, and expressed in the co-ordinate system of the object rather than of the viewer. If such a representation can be recovered, then any intensity image view of the object(s) may be synthesized by standard graphics techniques.

Recovering information lost by perspective projection is only one, mainly geometric, problem of computer vision—another is understanding image brightness. The only information available in an intensity image is the brightness of the appropriate pixel, which is dependent on a number of independent factors such as object surface reflectance properties (given by the surface material, microstructure, and marking), illumination properties, and object surface orientation with respect to viewer and light source—see Section 3.4. It is a non-trivial and again ill-posed problem to separate these components when trying to recover the 3D geometry of an object from the intensity image.

Some applications work with 2D images directly—for example, an image of a flat specimen viewed by a microscope with transparent illumination, a character drawn on a sheet of paper, the image of a fingerprint, etc. Many basic and useful methods used in digital image analysis do not therefore depend on whether the object was originally 2D or 3D. Much of the material in this book restricts itself to the study of such methods—the problem of 3D understanding is addressed explicitly in Chapters 11 and 12.

Image processing often deals with **static** images, in which time is constant. A monochromatic static image is represented by a continuous image function $f(x, y)$ whose arguments are co-ordinates in the plane. Most images considered in this book will be presented as monochromatic and static, but we will frequently note that they are taken from color or dynamic applications, and extensions of the techniques we develop will be obvious.

Computerized image processing uses digital image functions which are usually represented by matrices, so co-ordinates are natural numbers. The domain of the image function is a region R in the plane

$$R = \{(x, y), 1 \leq x \leq x_m, 1 \leq y \leq y_n\}, \quad (2.2)$$

where x_m, y_n represent the maximal co-ordinates. The function has a limited domain— infinite summation or integration limits can be used, as the image function value is zero outside the domain R . The customary orientation of co-ordinates in an image is in the normal Cartesian fashion (horizontal x -axis, vertical y -axis, origin bottom-left), although the (*row*, *column*, origin top-left) orientation used in matrices is also often used.

The range of image function values is also limited; by convention, in monochromatic images the lowest value corresponds to black and the highest to white. Brightness values bounded by these limits are **gray-levels**.

The quality of a digital image grows in proportion to the spatial, spectral, radiometric, and time resolutions. The **spatial resolution** is given by the proximity of image samples in the image plane; **spectral resolution** is given by the bandwidth of the light frequencies captured by the sensor; **radiometric resolution** corresponds to the number of distinguishable gray-levels; and **time resolution** is given by the interval between time samples at which images are captured. The question of time resolution is important in dynamic image analysis, where time sequences of images are processed.

Images $f(x, y)$ can be treated as deterministic functions or as realizations of stochastic processes. Mathematical tools used in image description have roots in linear system theory, integral transforms, discrete mathematics, and the theory of stochastic processes. Mathematical transforms usually assume that the image function $f(x, y)$ is ‘well-behaved’, meaning that the function is integrable, has an invertible Fourier transform, etc. A comprehensive explanation of the mathematical background useful for representing and treating image functions is given in [Bracewell, 2004; Barrett and Myers, 2004].

2.2 Image digitization

An image to be processed by computer must be represented using an appropriate discrete data structure, for example, a matrix. An image captured by a sensor is expressed as a continuous function $f(x, y)$ of two co-ordinates in the plane. Image digitization means that the function $f(x, y)$ is **sampled** into a matrix with M rows and N columns. Image **quantization** assigns to each continuous sample an integer value—the continuous range of the image function $f(x, y)$ is split into K intervals. The finer the sampling (i.e., the larger M and N) and quantization (the larger K), the better the approximation of the continuous image function $f(x, y)$ achieved.

Image function sampling poses two questions. First, the sampling period should be determined—this is the distance between two neighboring sampling points in the image. Second, the geometric arrangement of sampling points (sampling grid) should be set.

2.2.1 Sampling

Clearly, there is a relationship between the density of digital sampling and the detail that the image will contain; the theoretical aspects of this (in particular, Shannon’s theorem) are given in Section 3.2.5—the reader is strongly encouraged to understand at least the implications of this important result. It is worth glancing ahead to Figure 3.11 to see a clear illustration of this intuitive issue.

For now, it is sufficient to appreciate that if quality comparable to an ordinary (standard) television image is required, sampling into a 512×512 grid is used (768×576 for PAL format and 640×480 for NTSC format using a rectangular capture window); this is the reason many image digitizers use this (or higher) resolution. Resolution of high-definition television (HDTV) is up to 1920×1080 pixels. Such a resolution turns out to be adequate for a very wide range of practically useful tasks. Yet, much higher image resolutions are routinely available – for example provided by digital cameras or smartphones – with $10,000 \times 7,096$ pixels being the highest resolution available from a CMOS chip in 2013 (79 Megapixels, see also Section 2.5.1).

A continuous image is digitized at **sampling points**. These sampling points are ordered in the plane, and their geometric relation is called the **grid**. The digital image

is then a data structure, usually a matrix. Grids used in practice are usually square (Figure 2.2a) or hexagonal (Figure 2.2b). It is important to distinguish the grid from the raster; the **raster** is the grid on which a neighborhood relation between points is defined.¹

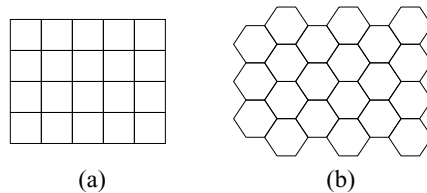


Figure 2.2: (a) Square grid. (b) Hexagonal grid.
© Cengage Learning 2015.

One infinitely small sampling point in the grid corresponds to one picture element also called a **pixel** or **image element** in the digital image; in a three-dimensional image, an image element is called a **voxel** (volume element). The set of pixels together covers the entire image; however, the pixel captured by a real digitization device has finite size (since the sampling function is not a collection of ideal Dirac impulses but a collection of limited impulses—see Section 3.2.5). The pixel is a unit which is not further divisible² from the image analysis point of view. We shall often refer to a pixel as a ‘point’.

2.2.2 Quantization

A value of the sampled image $f_s(j \Delta x, k \Delta y)$ is expressed as a digital value in image processing. The transition between continuous values of the image function (brightness) and its digital equivalent is called **quantization**. The number of quantization levels should be high enough to permit human perception of fine shading details in the image. Most digital image processing devices use quantization into k equal intervals. If b bits are used to express the values of the pixel brightness then the number of brightness levels is $k = 2^b$. Eight bits per pixel per channel (one each for red, green, blue) are commonly used although systems using other numbers (e.g., 16) can be found. An efficient computer representation of brightness values in digital images requires that eight bits, four bits, or one bit are used per pixel, meaning that one, two, or eight pixel brightnesses can be stored in one byte.

The main problem in images quantized with insufficient brightness levels is the occurrence of false contours which effect arises when the number of brightness levels is lower than that which humans can easily distinguish. This number is dependent on many factors—for example, the average local brightness—but displays which avoid this effect will normally provide a range of at least 100 intensity levels. This problem can be reduced when quantization into intervals of unequal length is used; the size of intervals corresponding to less probable brightnesses in the image is enlarged. These gray-scale transformation techniques are considered in Section 5.1.2.

Figures 3.11a and 2.3a-d demonstrate the effect of reducing the number of brightness levels in an image. An original image with 256 brightness levels (Figure 3.11a) has its

¹E.g., if 4-neighborhoods are used on the square grid, the square raster is obtained. Similarly, if 8-neighborhoods are used on the same square grid, then the octagonal raster is obtained. These 4-neighborhood and 8-neighborhood concepts are introduced in Section 2.3.1.

²In some case, the properties of an image at subpixel resolution can be computed. This is achieved by approximating the image function by a continuous function.

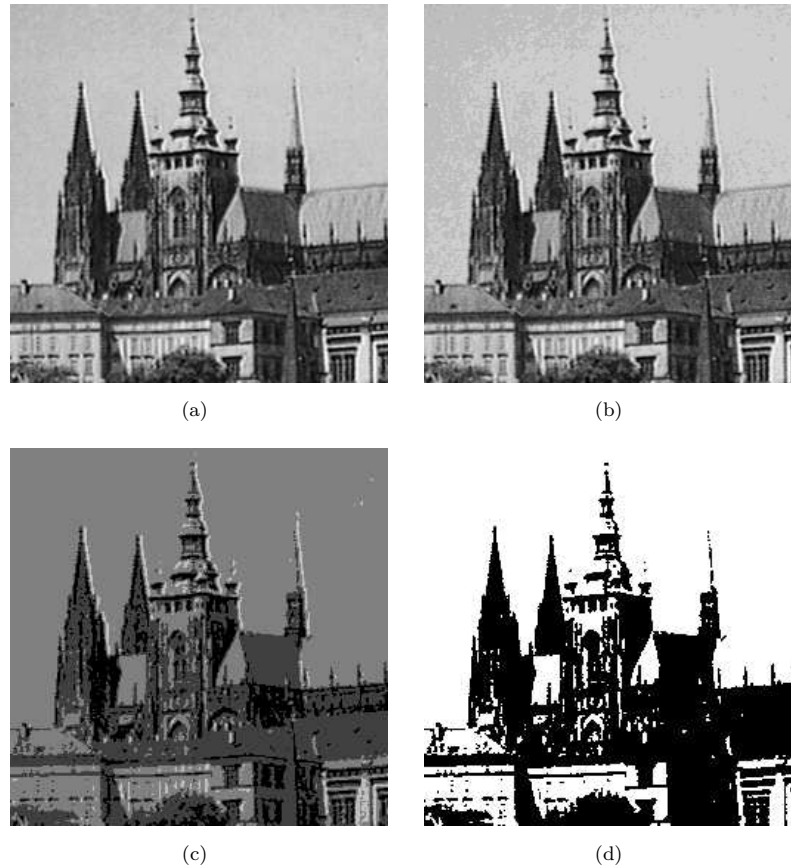


Figure 2.3: Brightness levels. (a) 64. (b) 16. (c) 4. (d) 2. © Cengage Learning 2015.

number of brightness levels reduced to 64 (Figure 2.3a), and no degradation is perceived. Figure 2.3b uses 16 brightness levels and false contours begin to emerge, and this becomes clearer in Figure 2.3c with four brightnesses and in Figure 2.3d with only two.

2.3 Digital image properties

A digital image has several properties, both metric and topological, which are somewhat different from those of continuous two-dimensional functions. Human perception of digital images is a frequent aspect, since judgment of image quality is also important.

2.3.1 Metric and topological properties of digital images

A digital image consists of picture elements with finite size—these pixels carry information about the brightness of a particular location in the image. Usually (and we assume this hereafter) pixels are arranged into a rectangular sampling grid. Such a digital im-

age is represented by a two-dimensional matrix whose elements are natural numbers corresponding to the quantization levels in the brightness scale.

Some intuitively clear properties of continuous images have no straightforward analogy in the domain of digital images. **Distance** is an important example. Any function D holding the following three condition is a ‘distance’ (or a *metric*)

$$\begin{aligned} D(\mathbf{p}, \mathbf{q}) &\geq 0; D(\mathbf{p}, \mathbf{q}) = 0 \text{ if and only if } \mathbf{p} = \mathbf{q}, && \text{identity,} \\ D(\mathbf{p}, \mathbf{q}) &= D(\mathbf{q}, \mathbf{p}), && \text{symmetry,} \\ D(\mathbf{p}, \mathbf{r}) &\leq D(\mathbf{p}, \mathbf{q}) + D(\mathbf{q}, \mathbf{r}), && \text{triangular inequality.} \end{aligned}$$

The distance between points with co-ordinates (i, j) and (h, k) may be defined in several different ways.

The **Euclidean distance** D_E known from classical geometry and everyday experience is defined by

$$D_E((i, j), (h, k)) = \sqrt{(i - h)^2 + (j - k)^2}. \quad (2.3)$$

The advantage of Euclidean distance is that it is intuitively obvious. The disadvantages are costly calculation due to the square root, and its non-integral value.

The distance between two points can also be expressed as the minimum number of elementary steps in the digital grid which are needed to move from the starting point to the end point. If only horizontal and vertical moves are allowed, the ‘**city block**’ **distance** D_4 is obtained (also called the L_1 metric or Manhattan distance, because of the analogy with the distance between two locations in a city with a rectangular grid of streets):

$$D_4((i, j), (h, k)) = |i - h| + |j - k|. \quad (2.4)$$

If moves in diagonal directions are allowed in the digitization grid, we obtain the distance D_8 , or ‘**chessboard**’ **distance**. D_8 is equal to the minimal number of king-moves on the chessboard from one part to another:

$$D_8((i, j), (h, k)) = \max \{ |i - h|, |j - k| \}. \quad (2.5)$$

These distance definitions are illustrated in Figure 2.4.

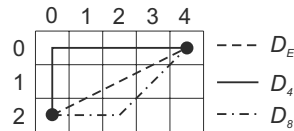


Figure 2.4: Distance metrics D_e , D_4 , and D_8 . © Cengage Learning 2015.

Pixel **adjacency** is another important concept in digital images. Two pixels $(\mathbf{p}, \mathbf{q})$ are called **4-neighbors** if they have distance $D_4(\mathbf{p}, \mathbf{q}) = 1$. Analogously, **8-neighbors** have $D_8(\mathbf{p}, \mathbf{q}) = 1$ —see Figure 2.5.

It will become necessary to consider important sets consisting of several adjacent pixels—**regions** (in set theory, a region is a connected set). More descriptively, we can define a **path** from pixel P to pixel Q as a sequence of points $A_1, A_2, \dots, A_n$, where $A_1 = P$, $A_n = Q$, and A_{i+1} is a neighbor of A_i , $i = 1, \dots, n - 1$; then a region is a set of pixels in which there is a path between any pair of its pixels, all of whose pixels also belong to the set.

If there is a path between two pixels in the set of pixels in the image, these pixels are called **contiguous**. Alternatively, we can say that a region is a set of pixels in which

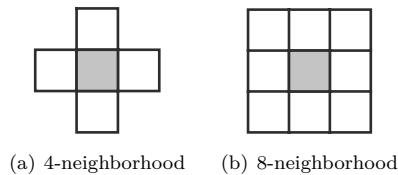


Figure 2.5: Neighborhood of the representative pixel (gray filled pixel in the middle). © Cengage Learning 2015.

each pair of pixels is contiguous. The relation ‘to be contiguous’ is reflexive, symmetric, and transitive and therefore defines a decomposition of the set (the image in our case) into equivalence classes (regions). Figure 2.6 illustrates a binary image decomposed by the relation ‘contiguous’ into three regions.

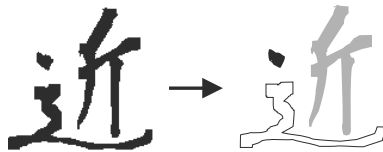


Figure 2.6: The relation ‘to be contiguous’ decomposes an image into individual regions. The Japanese Kanji character meaning ‘near from here’ decomposes into 3 regions. © Cengage Learning 2015.

Assume that R_i are disjoint regions in the image which were created by the relation ‘to be contiguous’, and further assume (to avoid special cases) that these regions do not touch the image bounds. Let R be the union of all regions R_i ; R^C be the set complement of R with respect to the image. The subset of R^C which is contiguous with the image bounds is called the **background**, and the remainder of the complement R^C is called **holes**.³ A region is called **simple contiguous** if it has no holes. Equivalently, the complement of a simply contiguous region is contiguous. A region with holes is called **multiple contiguous**.

Note that the concept of region uses only the property ‘to be contiguous’. Secondary properties can be attached to regions which originate in image data interpretation. It is common to call some regions in the image **objects**; a process which determines which regions in an image correspond to objects in the world is a part of image **segmentation** and is discussed in Chapters 6 and 7.

The brightness of a pixel is a very simple property which can be used to find objects in some images; if, for example, a pixel is darker than some predefined value (threshold), then it belongs to the object. All such points which are also contiguous constitute one object. A hole consists of points which do not belong to the object and are surrounded by the object, and all other points constitute the background. An example is the black printed text on this white sheet of paper, in which individual letters are objects. White areas surrounded by the letter are holes, for example, the area inside the letter ‘o’. Other white parts of the paper are the background.

These neighborhood and contiguity definitions on the square grid create interesting paradoxes. Figure 2.7a shows two digital line segments with 45° slope. If 4-connectivity is used, the lines are not contiguous at each of their points. An even worse conflict with intuitive understanding of line properties is also illustrated; two perpendicular lines do intersect in one case (upper right intersection) and do not intersect in another case (lower left), as they do not have a common point (i.e., their set intersection is empty).

³Some literature does not distinguish holes and background, and calls both of them background.

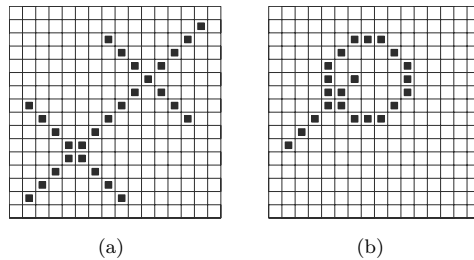


Figure 2.7: Paradoxes of crossing lines.
© Cengage Learning 2015.

It is known from Euclidean geometry that each closed curve (e.g., a circle) divides the plane into two non-contiguous regions. If images are digitized in a square grid using 8-connectivity, we can draw a line from the inner part of a closed curve into the outer part which does not intersect the curve (Figure 2.7b). This implies that the inner and outer parts of the curve constitute only one region because all pixels of the line belong to only one region, giving another paradox. One possible solution to contiguity paradoxes is to treat objects using 4-neighborhoods and background using 8-neighborhoods (or vice versa). A more exact treatment of digital images paradoxes and their solution for binary and other images can be found in [Klette and Rosenfeld, 2004].

These problems are typical on square grids—a hexagonal grid (see Figure 2.2), however, solves many of them. Any point in the hexagonal raster has the same distance to all its six neighbors. There are some problems peculiar to the hexagonal raster as well (for example, it is difficult to express a Fourier transform on it). For reasons of simplicity and ease of processing, most digitizing devices use a square grid despite the known drawbacks.

An alternative approach to the connectivity problems is to use discrete topology based on cellular complexes [Kovalevsky, 1989]. This approach develops a complete strand of image encoding and segmentation that deals with many issues we shall come to later, such as the representation of boundaries and regions. The idea, first proposed by the German mathematician Riemann in the nineteenth century, considers families of sets of different dimensions; points, which are 0-dimensional sets, may then be assigned to sets containing higher-dimensional structures (such as pixel arrays). This approach permits the removal of the paradoxes we have seen.

The **distance transform**—also called the **distance function** or **chamfering algorithm** or simply **chamfering**—is a simple application of the concept of distance. The idea is important as it provides the basis of several fast algorithms that will be seen multiple times in this book. The distance transform provides the distance of pixels from some image subset (perhaps describing objects or some features). The resulting ‘image’ has pixel values of 0 for elements of the relevant subset, low values for close pixels, and then high values for pixels remote from it—the appearance of this array gives the name to the technique.

For illustration, consider a binary image, in which ones represent the objects and zeros the background. The input image is shown in Figure 2.8 and the result of the D_4 distance transform is illustrated in Figure 2.9.

To calculate the transform, a two-pass algorithm has been suggested [Rosenfeld and Pfaltz, 1966, 1968] for distances D_4 and D_8 . The idea is to traverse the image by a small local mask. The first pass starts from the top-left of the image and moves horizontally

0	0	0	0	0	0	1	0
0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0
0	1	1	0	0	0	1	0
0	1	0	0	0	0	0	1
0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0

Figure 2.8: Input image: gray pixels correspond to objects and white to background. © Cengage Learning 2015.

5	4	4	3	2	1	0	1
4	3	3	2	1	0	1	2
3	2	2	2	1	0	1	2
2	1	1	2	1	0	1	2
1	0	0	1	2	1	0	1
1	0	1	2	3	2	1	0
1	0	1	2	3	3	2	1
1	0	1	2	3	4	3	2

Figure 2.9: Result of the $[D_4]$ distance transform. © Cengage Learning 2015.

left to right until it reaches the bounds of the image and then returns to the beginning of the next row. The second pass goes from the bottom-right corner in the opposite bottom-up, right to left direction using a different local mask. The effectiveness of the algorithm comes from propagating the values of the previous image investigation in a ‘wave-like’ manner. The masks used in calculations are shown in Figure 2.10.

AL	AL		BR
AL	P	P	BR
AL		BR	BR

Figure 2.10: Pixel neighborhoods used in distance transform calculations— $\mathbf{p}$ is the central one. The neighborhood on the left is used in the first pass (top-down, left to right); that on the right is used in the second (bottom-up, right to left). © Cengage Learning 2015.

Algorithm 2.1: Distance transform

1. Choose a distance N_{max} that exceeds the image dimension with respect to the chosen distance metric (D_4 or D_8). Initialize an image F of the same dimension as the input: set pixels corresponding to the subset(s) to be chamfered to 0, and all others to N_{max} .

2. (Refer to Figure 2.10.) Pass through image pixels from top to bottom and left to right. For a given pixel, consider neighbors above and to the left and set

$$F(\mathbf{p}) = \min_{\mathbf{q} \in AL} (F(\mathbf{p}), D(\mathbf{p}, \mathbf{q}) + F(\mathbf{q})) .$$

3. (Refer to Figure 2.10.) Pass through image pixels from bottom to top and right to left. For a given pixel, consider neighbors above and to the left and set

$$F(\mathbf{p}) = \min_{\mathbf{q} \in BR} (F(\mathbf{p}), D(\mathbf{p}, \mathbf{q}) + F(\mathbf{q})) .$$

4. If any pixels remain still set at N_{max} , go to (2).

5. The array F now holds a chamfer of the chosen subset(s).

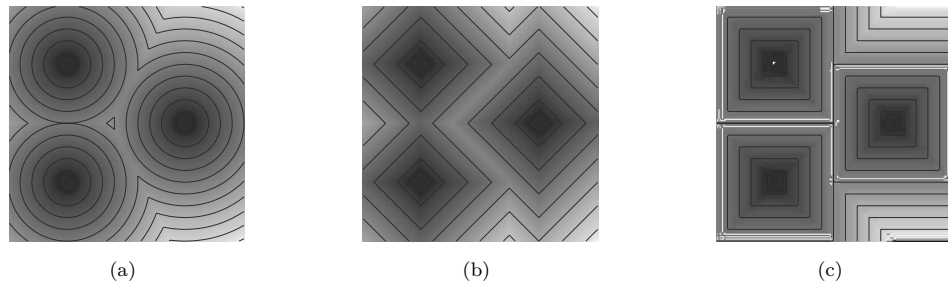


Figure 2.11: Three distances used often in distance transform calculations—the input consists of three isolated ‘ones’. Output distance is visualized as intensity; lighter values denote higher distances. Contour plots are superimposed for better visualization. (a) Euclidean distance D_E . (b) City block distance D_4 . (c) Chessboard distance D_8 . © Cengage Learning 2015.

This algorithm needs obvious adjustments at image boundaries, where the sets AL and BR are truncated. It is open to various improvements by using different distance calculations [Montanari, 1968; Barrow et al., 1977; Borgefors, 1986; Breu et al., 1995; Maurer et al., 2003]. Distance transform performance for D_E , D_4 , D_8 on an input consisting only of three distinct ‘ones’ is shown in Figure 2.11.

The distance transform has many applications, e.g., in discrete geometry, path planning and obstacle avoidance in mobile robotics, finding the closest feature in the image, and skeletonization (discussed with mathematical morphology methods in Section 13.5.5).

An **edge** is a further important concept used in image analysis. This is a local property of a pixel and its immediate neighborhood—it is a vector given by a magnitude and direction which tells us how fast the image intensity varies in a small neighborhood of a pixel. Images with many brightness levels are used for edge computation, and the gradient of the image function is used to compute edges. The edge direction is perpendicular to the gradient direction which points in the direction of the fastest image function growth. Edges are considered in detail in Section 5.3.2.

The related concept of the **crack edge** creates a structure between pixels in a similar manner to that of cellular complexes. However, it is more pragmatic and less mathematically rigorous. Four crack edges are attached to each pixel, which are defined by its relation to its 4-neighbors. The direction of the crack edge is that of increasing brightness, and is a multiple of 90° , while its magnitude is the absolute difference between the brightness of the relevant pair of pixels. Crack edges are illustrated in Figure 2.12 and may be used in considering image segmentation (Chapter 6).

The **border** (boundary) of a region is another important concept in image analysis. The border of a region R is the set of pixels within the region that have one or more neighbors outside R . The definition corresponds to an intuitive understanding of the border as a set of points at the bound of the region. This definition is sometimes referred to as the **inner border** to distinguish it from the **outer border**, that is, the border of the background (i.e., its complement) of the region. Inner and outer borders are illustrated in Figure 2.13. Due to the discrete nature of the image, some inner border elements which would be distinct in the continuous case coincide in the discrete case, as can be seen with the one-pixel-wide line at the right of Figure 2.13.

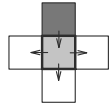


Figure 2.12: Crack edges.
© Cengage Learning 2015.

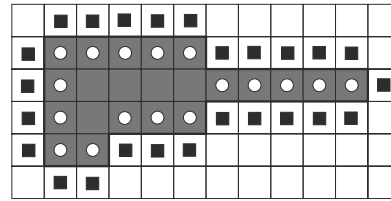


Figure 2.13: Region inner borders shown as white circles and outer borders shown as black squares (using 4-neighborhoods). © Cengage Learning 2015.

While edges and borders are related, they are not the same thing. ‘Border’ is a global concept related to a region, while ‘edge’ expresses local properties of an image function. One possibility for finding boundaries is chaining the significant edges (points with high gradient of the image function). Methods of this kind are described in Section 6.2.

A region is described as **convex** if any two points within it are connected by a straight line segment, and the whole line lies within the region—see Figure 2.14. The property of convexity decomposes all regions into two equivalence classes: convex and non-convex.



Figure 2.14: A convex region (left) and non-convex region (right). © Cengage Learning 2015.

A **convex hull** of a region is the smallest convex region containing the input (possibly non-convex) region. Consider an object whose shape resembles the letter ‘R’ (see Figure 2.15). Imagine a thin rubber band pulled around the object; the shape of the rubber band provides the convex hull of the object. Convex hull calculation is described in Section 8.3.3.



Figure 2.15: Description using topological components: An ‘R’ object, its convex hull, and the associated lakes and bays. © Cengage Learning 2015.

Topological properties are not based on the distance concept. Instead, they are invariant to *homeomorphic* transforms which can be illustrated for images as **rubber sheet transforms**. Imagine a small rubber balloon with an object painted on it; topological properties of the object are those which are invariant to arbitrary stretching of the rubber sheet. Stretching does not change contiguity of the object parts and does not change the number of holes in regions. We use the term ‘topological properties’ of the region to describe its qualitative properties invariant to small changes (e.g., the property of being convex), even though an arbitrary homeomorphic transformation can change a convex region to a non-convex one and vice versa. Considering the rubber sheet analogy, we mean that the stretching of the sheet is only gentle. Further properties of regions that are not rubber sheet invariant are described in Section 8.3.1.

An object with non-regular shape can be represented by a collection of its topological components, Figure 2.15. The set inside the convex hull which does not belong to an object is called the **deficit of convexity**. This can be split into two subsets: **lakes** (dark gray) are fully surrounded by the object; and **bays** (light gray) are contiguous with the border of the convex hull of the object.

The convex hull, lakes, and bays are sometimes used for object description; these features are used in Chapter 8 (object description) and in Chapter 13 (mathematical morphology).

2.3.2 Histograms

The **brightness histogram** $h_f(z)$ of an image provides the frequency of the brightness value z in the image—the histogram of an image with L gray-levels is represented by a one-dimensional array with L elements.

Algorithm 2.2: Computing the brightness histogram

1. Assign zero values to all elements of the array h_f .
2. For all pixels (x, y) of the image f , increment $h_f(f(x, y))$ by 1.

The histogram provides a natural bridge between images and a probabilistic description. We might want to find a first-order probability function $p_1(z; x, y)$ to indicate the probability that pixel (x, y) has brightness z . Dependence on the position of the pixel is not of interest in the histogram; the function $p_1(z)$ is of interest and the brightness histogram is its estimate. The histogram is often displayed as a bar graph, see Figure 2.16.

The histogram is usually the only global information about the image which is available. It is used when finding optimal illumination conditions for capturing an image, gray-scale transformations, and image segmentation to objects and background. Note

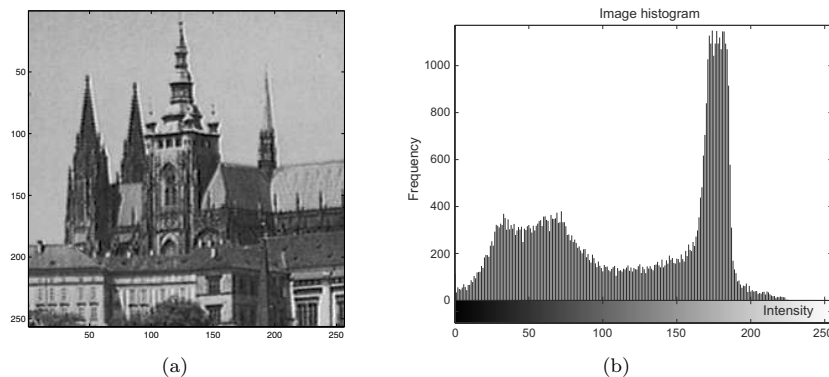


Figure 2.16: Original image (a) and its brightness histogram (b). © Cengage Learning 2015.

that one histogram may correspond to several images; for instance, a change of the object position on a constant background does not affect it.

The histogram of a digital image typically has many local minima and maxima, which may complicate its further processing. This problem can be avoided by local smoothing of the histogram; this may be done, for example, using local averaging of neighboring histogram elements as the base, so that a new histogram $h'_f(z)$ is calculated according to

$$h'_f(z) = \frac{1}{2K+1} \sum_{j=-K}^K h_f(z+j), \quad (2.6)$$

where K is a constant representing the size of the neighborhood used for smoothing. This algorithm would need some boundary adjustment, and carries no guarantee of removing all local minima. Other techniques for smoothing exist, notably Gaussian blurring; in the case of a histogram, this would be a one-dimensional simplification of the 2D Gaussian blur, equation (5.47), which will be introduced in Section 5.3.3.

2.3.3 Entropy

If a probability density p is known then image information content can be estimated regardless of its interpretation using **entropy** H . The concept of entropy has roots in thermodynamics and statistical mechanics but it took many years before it was related to information. The information-theoretic formulation comes from Shannon [Shannon, 1948] and is often called **information entropy**.

An intuitive understanding of information entropy relates to the amount of uncertainty about an event associated with a given probability distribution. The entropy can serve as an measure of ‘disorder’. As the level of disorder rises, entropy increases and events are less predictable.

Entropy is formally defined assuming a discrete random variable X with possible outcomes (called also states) $x_1, \dots, x_n$. Let $p(x_k)$ be the probability of the outcome x_k , $k = 1, \dots, n$. Then the entropy is defined as

$$H(X) \equiv \sum_{k=1}^n p(x_k) \log_2 \left(\frac{1}{p(x_k)} \right) = - \sum_{k=1}^n p(x_k) \log_2 p(x_k). \quad (2.7)$$

The entropy of the random variable X is the sum, over all possible outcomes k of X , of the product of the probability of outcome x_k with the logarithm of the inverse of the probability of x_k .

The base of the logarithm in this formula determines the unit in which entropy is measured. If this base is two then the entropy is given in bits. Recall that the probability density $p(x_k)$ needed to calculate the entropy is often estimated using a gray-level histogram in image analysis, Section 2.3.2.

Entropy measures the uncertainty about the realization of a random variable. For Shannon, it served as a proxy capturing the concept of information contained in a message as opposed to the portion of the message that is strictly determined and predictable by inherent structures. For example, we shall explore entropy to assess redundancy in an image for image compression (Chapter 14).

2.3.4 Visual perception of the image

Anyone who creates or uses algorithms or devices for digital image processing should take into account the principles of human image perception. If an image is to be analyzed by a human the information should be expressed using variables which are easy to perceive; these are psycho-physical parameters such as contrast, border, shape, texture, color, etc. Humans will find objects in images only if they may be distinguished effortlessly from the background. A detailed description of the principles of human visual perception can be found in [Bruce et al., 1996; Palmer, 1999]. Human perception of images provokes many illusions, the understanding of which provides valuable clues about visual mechanisms. Some of the better-known illusions will be mentioned here—the topic is covered exhaustively from the point of view of computer vision in [Frisby, 1979].

The situation would be relatively easy if the human visual system had a linear response to composite input stimuli—i.e., a simple sum of individual stimuli. A decrease of some stimulus, e.g., area of the object in the image, could be compensated by its intensity, contrast, or duration. In fact, the sensitivity of human senses is roughly logarithmically proportional to the intensity of an input signal. In this case, after an initial logarithmic transformation, response to composite stimuli can be treated as linear.

Contrast

Contrast is the local change in brightness and is defined as the ratio between average brightness of an object and the background. Strictly speaking, we should talk about luminance⁴ instead of brightness if our aim is to be physically precise. The human eye is logarithmically sensitive to brightness, implying that for the same perception, higher brightness requires higher contrast.

Apparent brightness depends very much on the brightness of the local surroundings; this effect is called conditional contrast. Figure 2.17 illustrates this with five circles of the same size surrounded by squares of different brightness. Humans perceive the brightness of the small circles as different.



Figure 2.17: Conditional contrast effect. Circles inside squares have the same brightness and are perceived as having different brightness values. © Cengage Learning 2015.

Acuity

Acuity is the ability to detect details in an image. The human eye is less sensitive to slow and fast changes in brightness in the image plane but is more sensitive to intermediate changes. Acuity also decreases with increasing distance from the optical axis.

Resolution in an image is firmly bounded by the resolution ability of the human eye; there is no sense in representing visual information with higher resolution than that of

⁴Luminance describes the amount of light that passes through or is emitted from a particular area, and falls within a given solid angle. Luminance is given candela per square meter [cd/m^2].

the viewer. Resolution in optics is defined as the inverse value of a maximum viewing angle between the viewer and two proximate points which humans cannot distinguish, and so fuse together.

Human vision has the best resolution for objects which are at a distance of about 250 mm from an eye under illumination of about 500 lux; this illumination is provided by a 60 W bulb from a distance of 400 mm. Under these conditions the distance between two distinguishable points is approximately 0.16 mm.

Some visual illusions

Human perception of images is prone to many illusions. For a comprehensive treatment of the subject, see [Palmer, 1999].

Object borders carry a lot of information for humans. Boundaries of objects and simple patterns such as blobs or lines enable adaptation effects similar to conditional contrast, mentioned above. The Ebbinghaus illusion is a well-known example—two circles of the same diameter in the center of images appear to have different diameters (Figure 2.18).

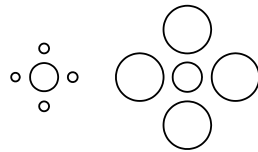


Figure 2.18: The Ebbinghaus illusion. © Cengage Learning 2015.

Perception of one dominant shape can be fooled by nearby shapes. Figure 2.19 shows parallel diagonal line segments which are not perceived as parallel. Figure 2.20 contains rows of black and white squares which are all parallel. However, the vertical zigzag squares disrupt our horizontal perception.

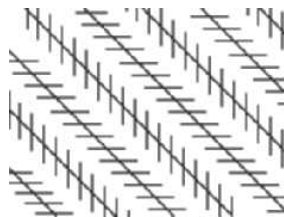


Figure 2.19: Disrupted parallel diagonal lines. © Cengage Learning 2015.

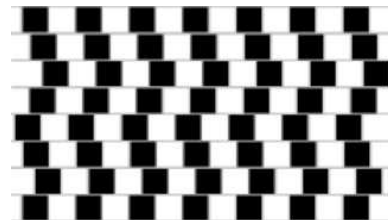


Figure 2.20: Horizontal lines are parallel, although not perceived as such. © Cengage Learning 2015.

Perceptual grouping

Perceptual grouping [Palmer, 1999] is a principle used in computer vision to aggregate elements provided by low-level operations such as edgels, which are small blobs to bigger chunks having some meaning. Its roots are in Gestalt psychology first postulated by

Wertheimer in 1912 [Brett King and Wertheimer, 2005]. Gestalt psychology proposes that the operational principle of the mind and brain is holistic, parallel, and with self-organizing tendencies.

Gestalt theory was meant to have general applicability; its main tenets, however, were induced almost exclusively from observations on visual perception. The overriding theme of the theory is that stimulation is perceived in organized or configuration terms. Gestalt in German means configuration, structure or pattern of physical, biological, or psychological phenomena so integrated to constitute a functional unit with properties not derivable by summation of its parts. Patterns take precedence over elements and have properties that are not inherent in the elements themselves.

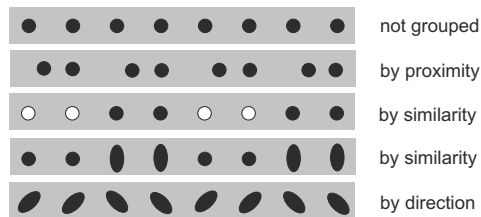


Figure 2.21: Grouping according to properties of elements. © Cengage Learning 2015.

The human ability to group items according to various properties is illustrated in Figure 2.21. Perceived properties help people to connect elements together based on strongly perceived properties as parallelism, symmetry, continuity and closure taken in a loose sense as illustrated in Figure 2.22.

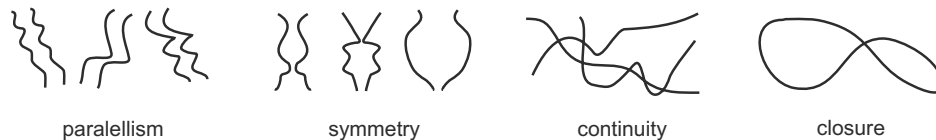


Figure 2.22: Illustration of properties perceived in images which allow humans to group together elements in cluttered scenes. © Cengage Learning 2015.

It has been demonstrated that mimicking perceptual grouping in machine vision system is a plausible technique. It permits the creation of more meaningful chunks of information from meaningless outcomes of low-level operations such as edge detection. Such grouping is useful in image understanding. This principle will be used in this book mainly for image segmentation.

2.3.5 Image quality

An image might be degraded during capture, transmission, or processing, and measures of image quality can be used to assess the degree of degradation. The quality required naturally depends on the purpose for which an image is used.

Methods for assessing **image quality** can be divided into two categories: subjective and objective. Subjective methods are often used in television technology, where the ultimate criterion is the perception of a selected group of professional and lay viewers. They appraise an image according to a list of criteria and give appropriate marks. Details about subjective methods may be found in [Pratt, 1978].

Objective quantitative methods measuring image quality are more interesting for our purposes. Ideally such a method also provides a good subjective test, and is easy to apply; we might then use it as a criterion in parameter optimization. The quality of an image $f(x, y)$ is usually estimated by comparison with a known reference image $g(x, y)$ [Rosenfeld and Kak, 1982], and a synthesized image is often used for this purpose. One class of methods uses simple measures such as the mean quadratic difference $\sum (g - f)^2$, but this does not distinguish a few big differences from many small differences. Instead of the mean quadratic difference, the mean absolute difference or simply maximal absolute difference may be used. Correlation between images f and g is another alternative.

Another class measures the resolution of small or proximate objects in the image. An image consisting of parallel black and white stripes is used for this purpose; then the number of black and white pairs per millimeter gives the resolution.

2.3.6 Noise in images

Real images are often degraded by some random errors—this degradation is usually called **noise**. Noise can occur during image capture, transmission, or processing, and may be dependent on, or independent of, the image content.

Noise is usually described by its probabilistic characteristics. Idealized noise, called **white noise** is often used. White noise has a constant power spectrum (explained in Section 3.2.3), meaning that all noise frequencies are present and have the same intensity. For example, the intensity of white noise does not decrease with increasing frequency as is typical in real-world signals. White noise is frequently employed to model the worst approximation of degradation, the advantage being that its use simplifies calculations.

A special case of white noise is **Gaussian noise**. A random variable with a Gaussian (normal) distribution has its probability density function given by the Gaussian curve. In the 1D case the density function is

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \quad (2.8)$$

where μ is the mean and σ the standard deviation of the random variable. Gaussian noise is a very good approximation to noise that occurs in many practical cases.

When an image is transmitted through some channel, noise which is usually independent of the image signal occurs. This signal-independent degradation is called **additive noise** and can be described by the model

$$f(x, y) = g(x, y) + \nu(x, y), \quad (2.9)$$

where the noise ν and the input image g are independent variables. Algorithm 2.3 will generate zero mean additive Gaussian noise in an image—this can often be of use in testing or demonstrating many algorithms in this book which are designed to remove noise, or to be noise resistant.

Algorithm 2.3: Generation of additive, zero mean Gaussian noise

1. Suppose an image has gray-level range $[0, G - 1]$. Select $\sigma > 0$; low values generate less noise.

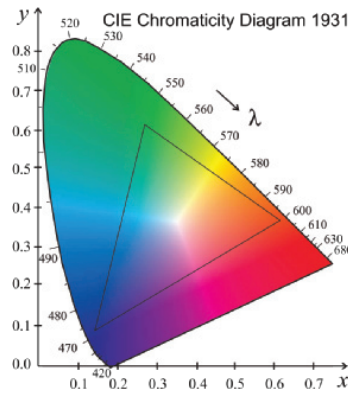


Plate 1: © Cengage Learning 2015. Page 35, Figure 2.30.

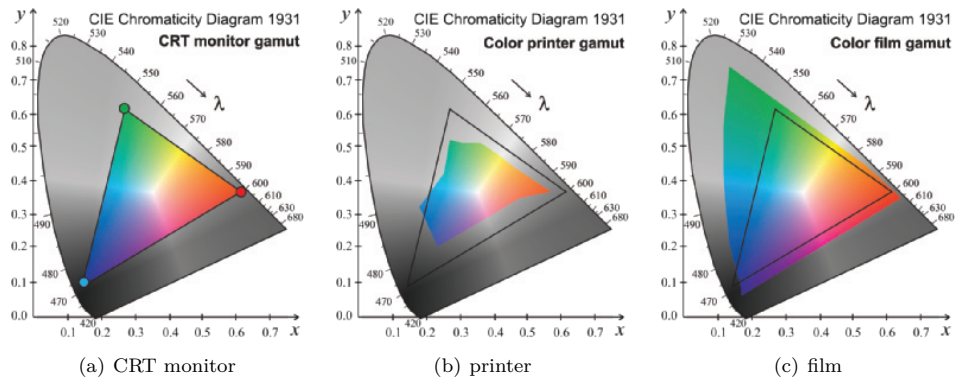


Plate 2: © Cengage Learning 2015. Page 35, Figure 2.31.

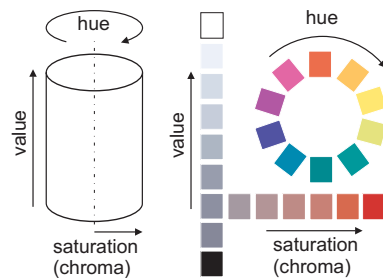


Plate 3: © Cengage Learning 2015. Page 37, Figure 2.33.

B Color inset

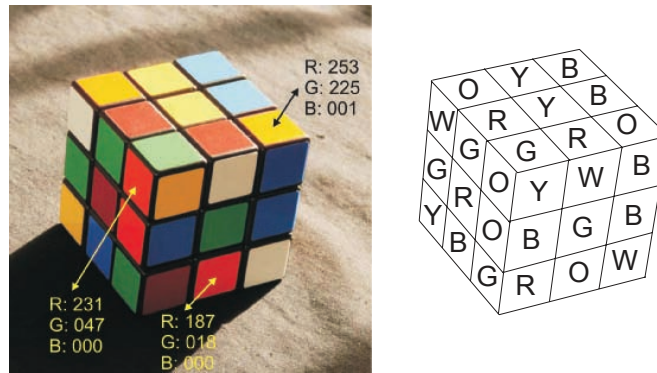


Plate 4: © Cengage Learning 2015. Page 39, Figure 2.34.

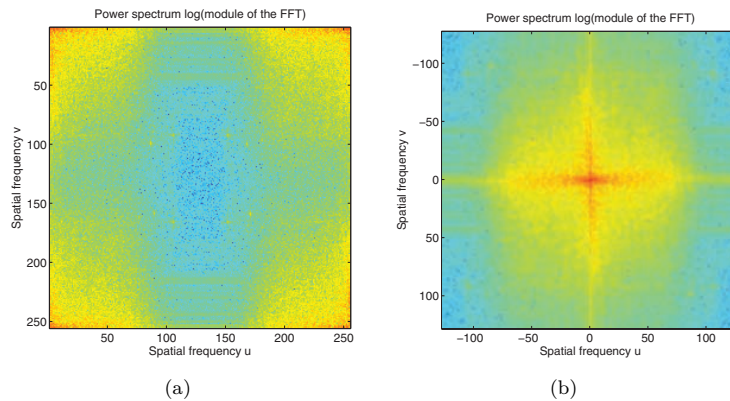


Plate 5: © Cengage Learning 2015. Page 61, Figure 3.7.

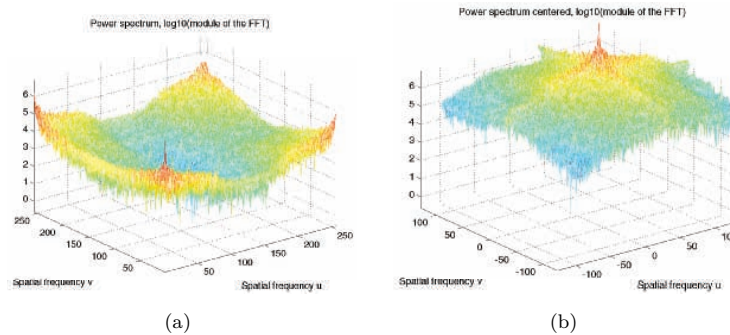


Plate 6: © Cengage Learning 2015. Page 62, Figure 3.9.



Plate 7: *Courtesy of M. Urban, Czech Technical University, Prague. Page 159, Figure 5.34.*

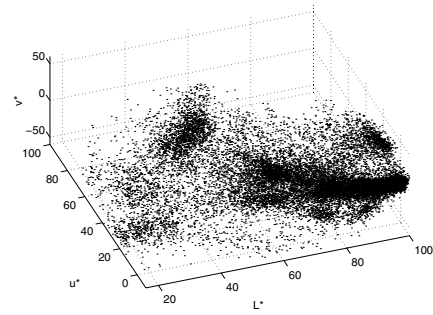


Plate 8: *Courtesy of J. Matas, Czech Technical University, Prague. Page 160, Figure 5.35.*

D Color inset

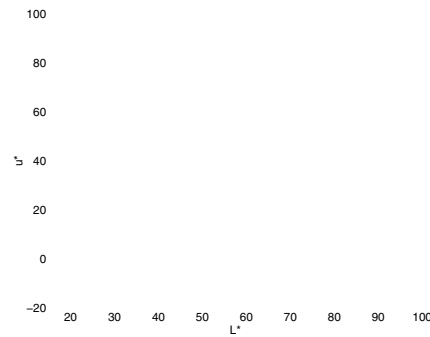


(a)

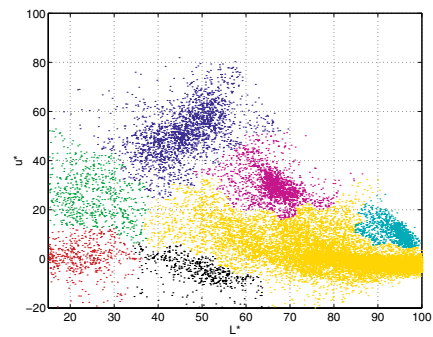


(b)

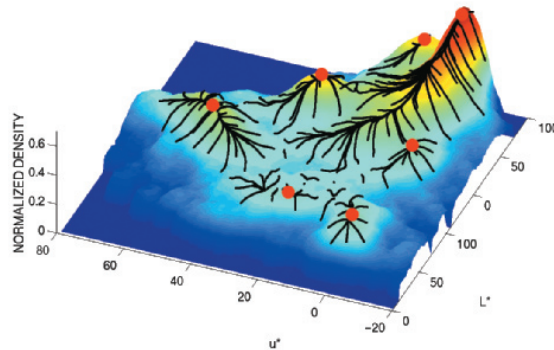
Plate 9: © 2002 IEEE. Reprinted, with permission, from D. Comaniciu, P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, pp. 603-619, 2002. Page 259, Figure 7.2.



(a)



(b)



(c)

Plate 10: © 2002 IEEE. Reprinted, with permission, from D. Comaniciu, P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, pp. 603-619, 2002. Page 260, Figure 7.3.

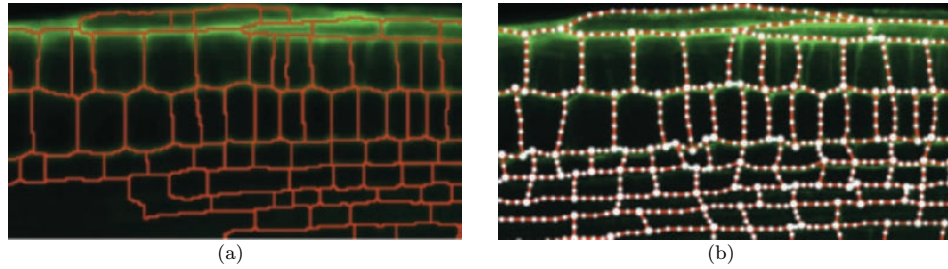


Plate 11: Republished with permission of American Society of Plant Biologists, from *The Plant Cell*, Pound *et al.*, "CellSeT: Novel Software to Extract and Analyze Structured Networks of Plant Cells from Confocal Images," 1989; permission conveyed through Copyright Clearance Center, Inc. Page 268, Figure 7.11.

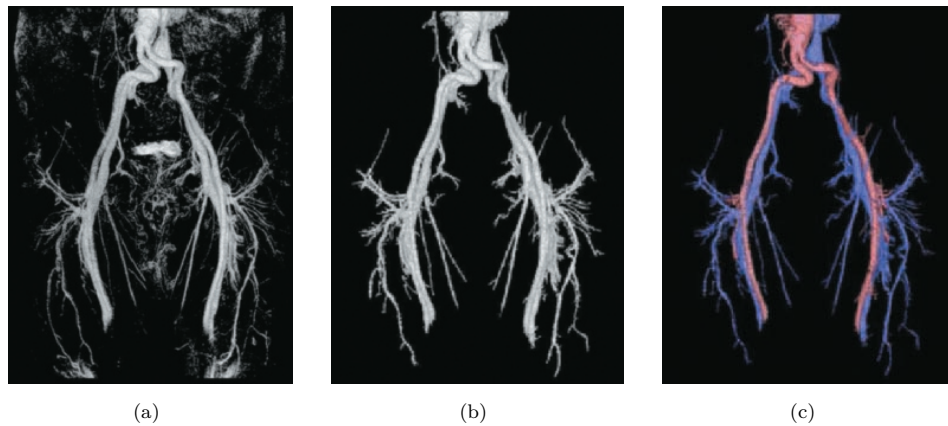


Plate 12: Courtesy of J. K. Udupa, University of Pennsylvania. Page 287, Figure 7.28.

F Color inset

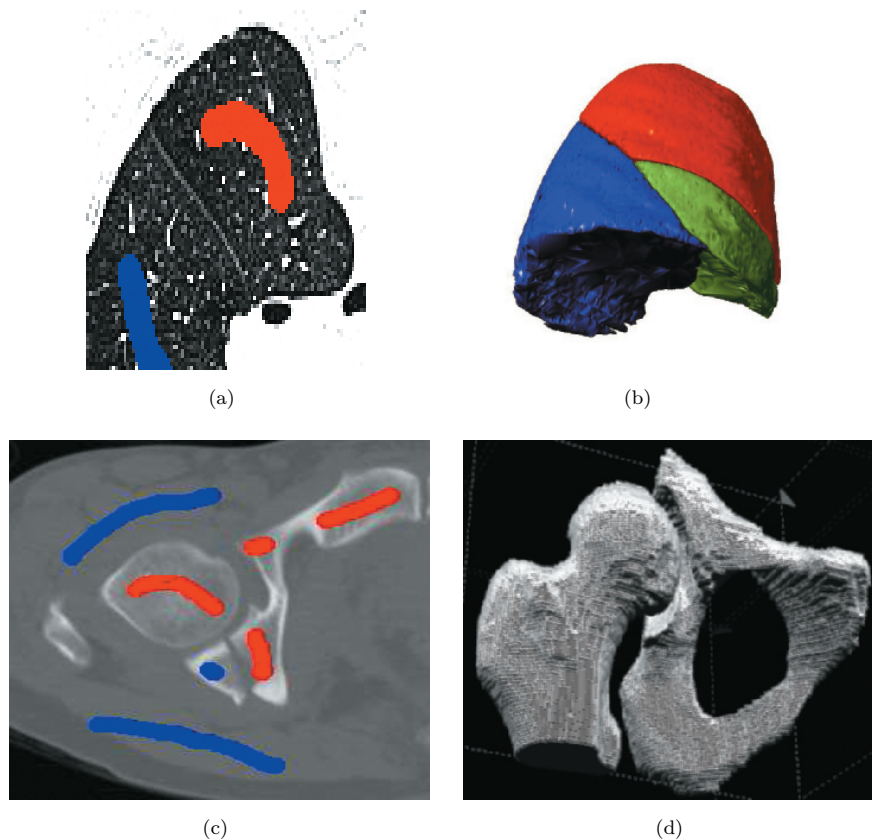


Plate 13: Courtesy of Y. Boykov, University of Western Ontario and B. Geiger, Siemens Research.
 Page 302, Figure 7.38.

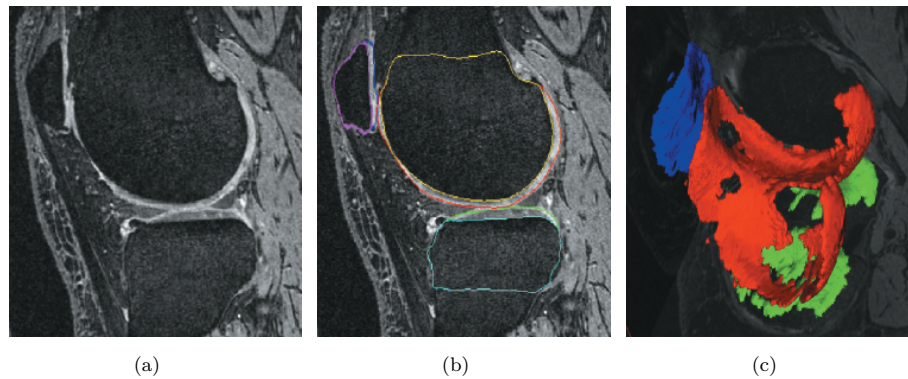


Plate 14: Yin, Y., Zhang, X., Williams, R., Wu, X., Anderson, D.D., Sonka, M., “LOGISMOS-Layered Optimal Graph Image Segmentation of Multiple Objects and Surfaces: Cartilage Segmentation in the Knee Joint,” *Medical Imaging, IEEE Transactions, Volume: 29, Issue: 12, Digital Object Identifier: 10.1109/TMI.2010.2058861, Publication Year: 2010, Page(s): 2023-2037. Page 316, Figure 7.48.*

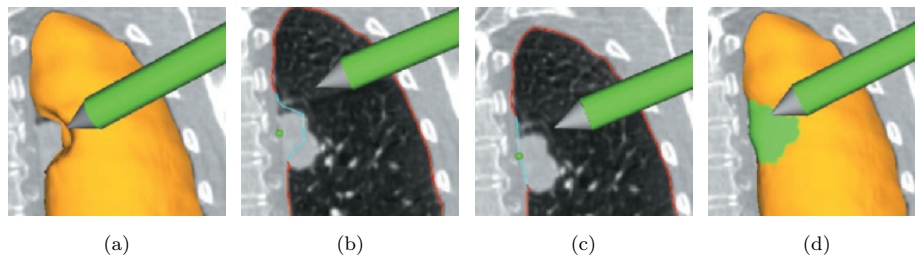


Plate 15: Sun S., Sonka M., and Beichel R., “Lung segmentation refinement based on optimal surface finding utilizing a hybrid desktop/virtual reality user interface,” *Computerized Medical Imaging and Graphics*, 37(1):15-27, 2013b. Page 316, Figure 7.49.

Decision tree structure

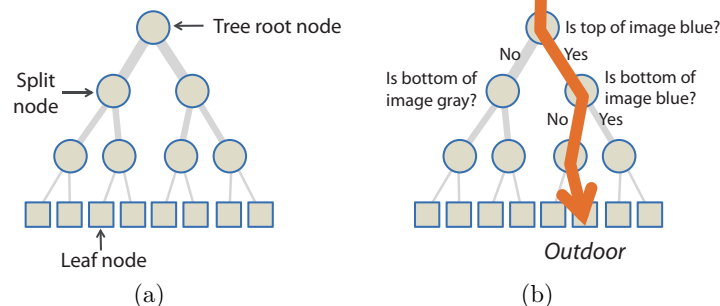


Plate 16: Based on [Criminisi et al., 2011]. Page 443, Figure 9.35.

H Color inset

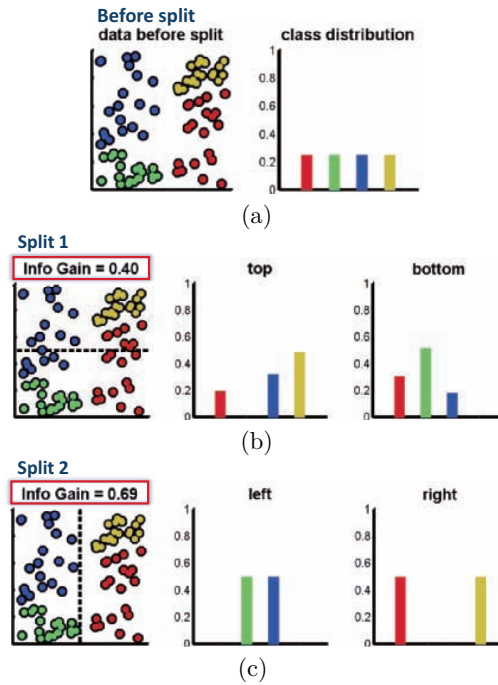


Plate 17: Based on [Criminisi et al., 2011]. Page 444, Figure 9.36.

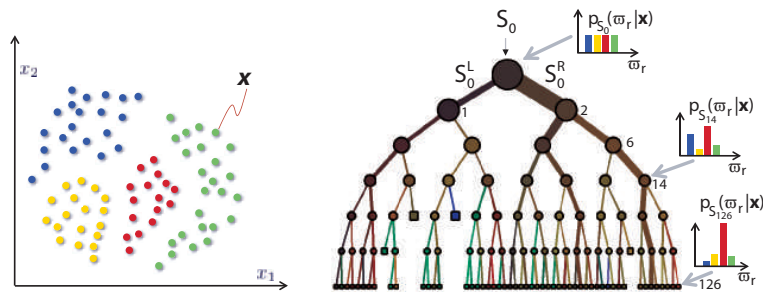


Plate 18: Based on [Criminisi et al., 2011]. Page 445, Figure 9.37.

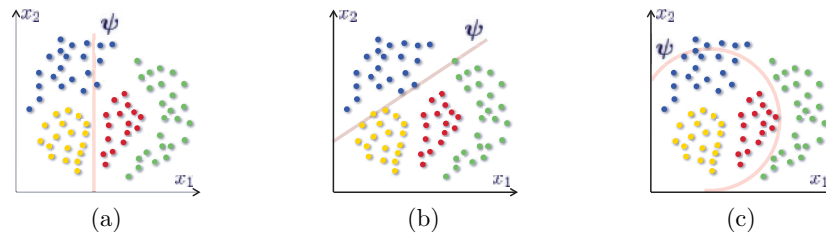


Plate 19: Based on [Criminisi et al., 2011]. Page 446, Figure 9.38.

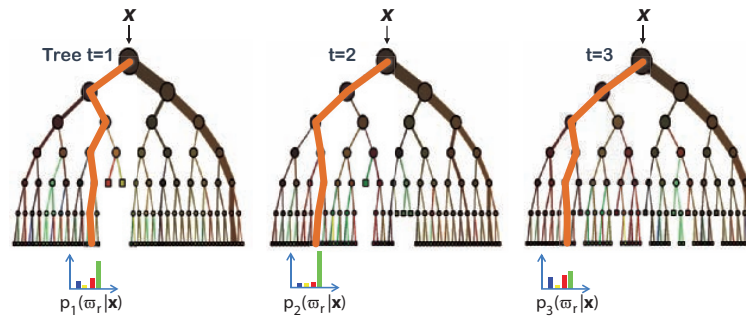


Plate 20: Based on [Criminisi et al., 2011]. Page 447, Figure 9.39.

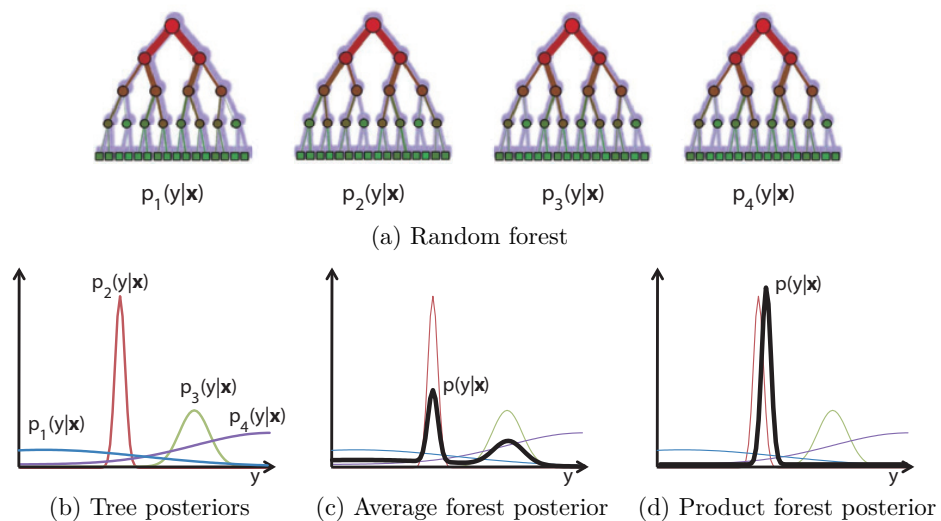


Plate 21: Based on [Criminisi et al., 2011]. Page 447, Figure 9.40.