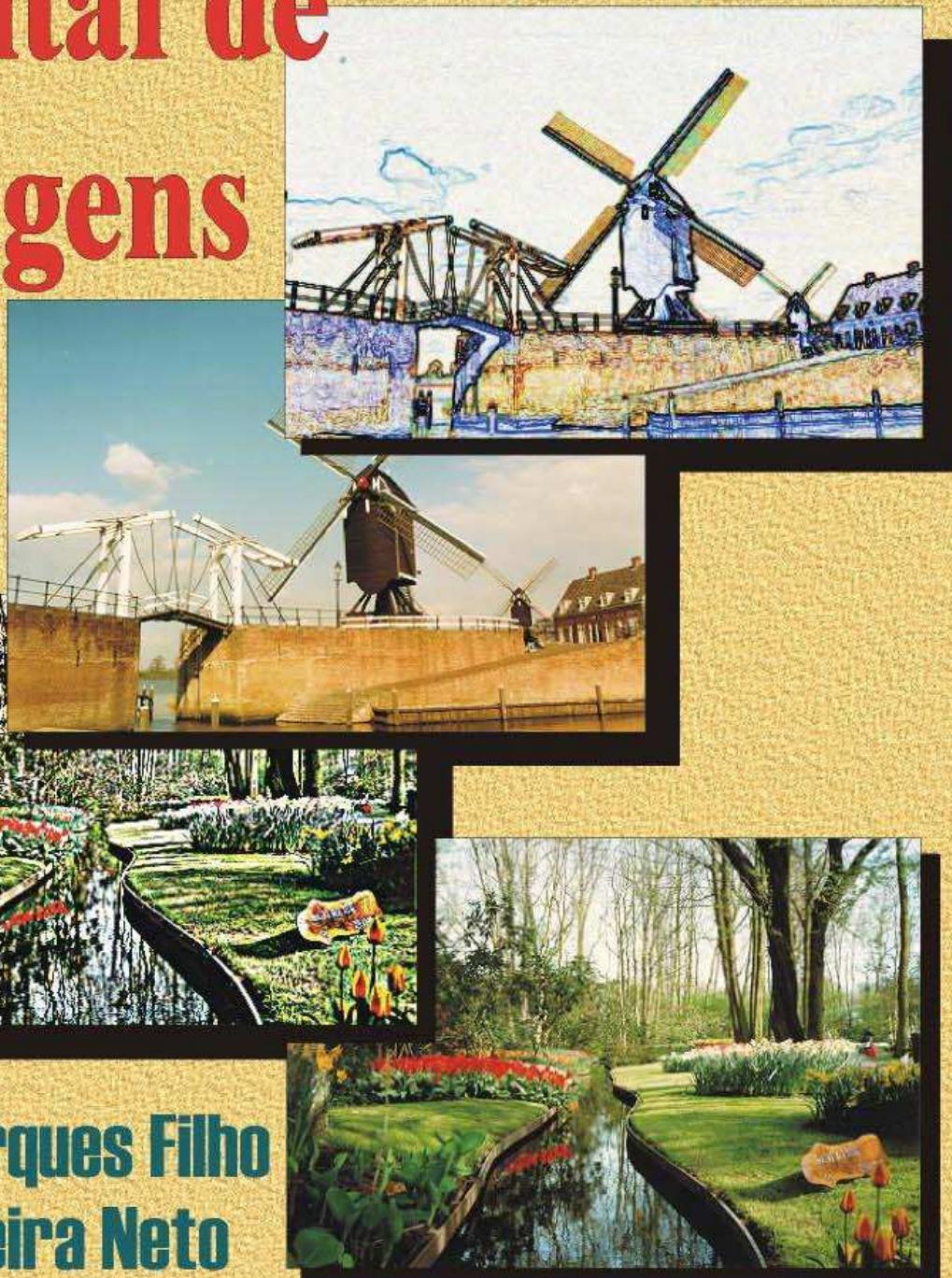


Processamento Digital de Imagens



Ogê Marques Filho
Hugo Vieira Neto



Processamento Digital de Imagens

*Ogê Marques Filho
Hugo Vieira Neto*

1999

À minha esposa Fabiana, pelo apoio incondicional e permanente.

Ogê Marques Filho

À minha família e meus verdadeiros amigos.

Hugo Vieira Neto

Agradecimentos

Inúmeras pessoas colaboraram, direta ou indiretamente para a concretização deste projeto e a elas dedicamos nossa eterna gratidão.

À Editora Brasport, em especial ao Sr. Sérgio Martins, pela confiança depositada na importância deste trabalho e na seriedade de seus autores.

Ao Sr. Joaquim Guerreiro, da Guerreiro Livros Técnicos de Curitiba, amigo de longa data, pela sua generosidade e pelo apoio decisivo para a materialização desta obra.

Aos familiares e verdadeiros amigos que apoiaram esta e outras iniciativas, nosso sincero reconhecimento.

Aos professores e orientadores do *Philips International Institute, Philips Research Laboratories* e *Delft University of Technology*, particularmente Bart de Greef, D. E. Boekee e Harald Ihle, por terem guiado nossos primeiros passos nesta área de estudo e pesquisa, nossa sincera gratidão.

Nosso agradecimento a todos os colegas docentes do Centro Federal de Educação Tecnológica do Paraná - CEFET-PR, em especial a Álvaro Stelle, Maria Gertrudes Te Vaarwerk, Paulo Roberto Brero de Campos e Walter Godoy Junior, e particularmente ao colega Humberto Remigio Gamba, pela paciente revisão dos originais.

A outros autores que nos inspiraram a trilhar o mesmo rumo, dentre eles: Borko Furht, Juarez do Nascimento, Raul Marques Pereira Friedmann e Volnei Antonio Pedroni.

Ao apoio das bibliotecárias Arlene de Oliveira Dias, Márcia Andreiko e Marilene do Rocio Veiga.

Agradecemos de forma especial aos alunos e ex-alunos que dedicaram seus esforços na pesquisa e implementação de técnicas de Processamento Digital de Imagens, dentre eles: Aderbal Paz, Alex Holztratner, Alsemiro Alves Junior, Augusto Serbena, Carlos Alberto Jayme, Carlos Alberto Zanella, Ciro de Carvalho Braga, Cláudio Navarro, Dalton Roberto Maran Salvatti, Daniel Góri Palka, Diego de Alves e Souza, Domingo Edmundo Saucedo, Edson Luis Morais, Eduardo Nascimento de Freitas, Eduardo Saito, Emanuel-Werner Kohlscheen, Emerson Kamogari, Emerson Luis Parolin, Fabiana Leskiu Marques, Fabio Luis Urú, Fábio Luiz de Andrade, Fábio Morais da Costa, Gilson Yukio Sato, Hélio Okuyama, Horst Lindner Junio, Ilídio Dinis Matola, Jefferson Osowsky, João Cadamuro Junior, Julio Fujisawa, Leonardo Carvalho Neto, Luiz Renato Quinalha, Manoel Garbuio de Souza, Marcelo Diogo dos Santos, Marcelo Mazzotti, Marcelo Monteiro, Marcos Alberto Lopes, Marcos Francisco Canali, Maurício Hadime Suzuki, Otávio Sugeno, Ricardo Schmidlin Imbiriba, Rodrigo Nastás Acras, Sacha Tadeu Branco, Sérgio Kubo, Sérgio L. Rocha Loures, Sergio Luis Resnauer, Silvio Cezar Bortolini, Simone Crocetti Pereira, Wilson Kawano e Zundir Buzzi Junior.

Sumário

Prefácio	xix
Capítulo 1 – Introdução	1
1.1 Processamento de Imagens: breve histórico e exemplos de aplicações	1
1.2 Um sistema de processamento de imagens e seus componentes	2
1.3 O sistema visual humano	5
1.4 Sistemas de Visão Artificial: fundamentos e desafios	7
1.5 Estrutura e escopo do livro	11
Exercícios Propostos	11
Na Internet	11
Bibliografia	15
Bibliografia Recomendada	16
Capítulo 2 - Fundamentos de Imagens Digitais	19
2.1 Aquisição e digitalização de imagens	19
2.2 Propriedades de uma imagem digital	25
2.3 Operações lógicas e aritméticas	28
2.4 Operações de convolução com máscaras	34
2.5 Transformações geométricas	42
Exercícios Propostos	48
No computador	51
Na Internet	51
Bibliografia	52
Capítulo 3 - Técnicas de Modificação de Histograma	55
3.1 Conceito de histograma	55
3.2 Transformações de intensidade	59
3.3 Equalização de histograma	61
3.4 Especificação direta de histograma	65
3.5 Outras técnicas	68
3.6 Limiarização (<i>Thresholding</i>)	71

Exercícios Propostos	77
No computador	79
Na Internet	79
Bibliografia	80
Capítulo 4 - Filtragem, Realce e Suavização de Imagens	83
4.1 Considerações iniciais	83
4.2 Suavização de imagens no domínio espacial	85
4.3 Realce de imagens no domínio espacial	95
4.4 Transformada de Fourier	99
4.5 Filtragem no domínio da freqüência	108
4.6 Processamento de imagens coloridas	118
4.7 Filtros adaptativos	126
Exercícios Propostos	133
No computador	134
Na Internet	135
Bibliografia	136
Capítulo 5 - Morfologia Matemática	139
5.1 Introdução	139
5.2 Dilatação e Erosão	139
5.3 Abertura e Fechamento	143
5.4 Transformação <i>hit-or-miss</i>	147
5.5 Algoritmos morfológicos básicos	148
Exercícios Propostos	163
No computador	164
Na Internet	164
Bibliografia	164
Capítulo 6 - Compressão e Codificação de Imagens	167
6.1 Fundamentos	167
6.2 Modelos de compressão de imagem	172
6.3 Elementos de Teoria da Informação	175
6.4 Compressão sem perdas	180

6.5 Compressão com perdas	192
6.6 Padrões de compressão de imagens	197
Exercícios Propostos	219
No computador	221
Na Internet	221
Bibliografia	222
Capítulo 7 - Aspectos práticos de hardware e software para processamento de imagens	225
7.1 O hardware	225
7.2 O software	232
Na Internet	239
Bibliografia	240
Apêndice A - Formatos de arquivos de imagens	243
A.1 Representação através de <i>bitmaps</i> e através de vetores	243
A.2 Formatos de Arquivos de Imagem	244
Na Internet	261
Bibliografia	262
Apêndice B - Roteiros de laboratório de processamento de imagens	263
B.1 Conceitos Introdutórios	263
B.2 Utilizando o MATLAB	263
B.3 Comandos e Funções da <i>Toolbox</i> de Processamento de Imagens	267
B.4 Roteiros de práticas de laboratório	271
Na Internet	298
Bibliografia	298
Glossário	299

Conteúdo

Prefácio	xix
Capítulo 1 – Introdução	1
1.1 Processamento de Imagens: breve histórico e exemplos de aplicações	1
1.2 Um sistema de processamento de imagens e seus componentes	2
1.2.1 Aquisição	2
1.2.2 Armazenamento	2
1.2.3 Processamento	3
1.2.4 Transmissão	4
1.2.5 Exibição	4
1.3 O sistema visual humano	5
1.4 Sistemas de Visão Artificial: fundamentos e desafios	7
1.4.1 Estrutura de um Sistema de Visão Artificial	9
1.4.2 Domínio do problema e resultado	9
1.4.3 Aquisição da imagem	9
1.4.4 Pré-processamento	9
1.4.5 Segmentação	10
1.4.6 Representação e Descrição	10
1.4.7 Reconhecimento e Interpretação	10
1.4.8 Base de Conhecimento	10
1.5 Estrutura e escopo do livro	11
Exercícios Propostos	11
Na Internet	11
Bibliografia	15
Bibliografia Recomendada	16
Capítulo 2 - Fundamentos de Imagens Digitais	19
2.1 Aquisição e digitalização de imagens	19
2.1.1 Aquisição	21
2.1.2 Digitalização	22
2.2 Propriedades de uma imagem digital	25

2.2.1 Vizinhança	25
2.2.2 Conectividade	26
2.2.3 Adjacência	27
2.2.4 Caminho	27
2.2.5 Medições de distância	27
Distância Euclidiana	27
Distância D_4 (<i>city-block</i>)	27
Distância D_8 (tabuleiro de xadrez)	27
2.3 Operações lógicas e aritméticas	28
2.3.1 Operações aritméticas pixel a pixel	29
2.3.2 Operações lógicas pixel a pixel	31
2.3.3 Operações orientadas a vizinhança	33
2.4 Operações de convolução com máscaras	34
2.4.1 Deteção de pontos isolados	37
2.4.2 Deteção de linhas	37
2.4.3 Deteção de bordas	37
2.5 Transformações geométricas	42
2.5.1 Ampliação e redução (<i>zoom</i>)	42
2.5.2 Alterações de dimensões (<i>scaling</i> e <i>sizing</i>)	42
2.5.3 Translação	44
2.5.4 Rotação	44
2.5.5 Espelhamento (<i>Flip</i>)	45
2.5.6 <i>Warping</i>	45
2.5.7 <i>Cropping, cutting e pasting</i>	47
Exercícios Propostos	48
No computador	51
Na Internet	51
Bibliografia	52
Capítulo 3 - Técnicas de Modificação de Histograma	55
3.1 Conceito de histograma	55
3.2 Transformações de intensidade	59
3.3 Equalização de histograma	61
3.4 Especificação direta de histograma	65

3.5 Outras técnicas	68
3.5.1 Hiperbolização	69
3.5.2 Hiperbolização quadrática	69
3.5.3 Expansão de histograma (<i>Input cropping</i>)	70
3.5.4 Compressão de histograma (<i>Output cropping</i>)	70
3.6 Limiarização (<i>Thresholding</i>)	71
3.6.1 Influência da iluminação	74
3.6.2 Limiarização pelas propriedades estatísticas da imagem	75
Exercícios Propostos	77
No computador	79
Na Internet	79
Bibliografia	80
Capítulo 4 - Filtragem, Realce e Suavização de Imagens	83
4.1 Considerações iniciais	83
4.1.1 Filtragem no domínio espacial	83
4.1.2 Filtragem no domínio da freqüência	84
4.2 Suavização de imagens no domínio espacial	85
4.2.1 Introdução	85
4.2.2 Filtro da média	86
4.2.3 Filtro da mediana	90
4.2.4 Outros filtros	93
Média de múltiplas imagens	93
Média dos k vizinhos mais próximos	94
4.3 Realce de imagens no domínio espacial	95
4.3.1 Filtro passa-altas básico	95
4.3.2 Realce por diferenciação	96
4.3.3 Filtragem <i>high-boost</i>	97
4.4 Transformada de Fourier	99
4.4.1 Transformada de Fourier para sinais unidimensionais (1-D) contínuos	99
4.4.2 Transformada de Fourier para sinais bidimensionais (2-D) contínuos	100
4.4.3 Transformada de Fourier para sinais unidimensionais (1-D) discretos	101

4.4.4 Transformada de Fourier para sinais bidimensionais (2-D) discretos	102
4.4.5 Propriedades da transformada de Fourier para sinais bidimensionais (2-D) discretos	103
Separabilidade	103
Translação	104
Periodicidade e simetria conjugada	104
Distributividade	105
Rotação	105
Escala	106
Valor médio	106
Laplaciano	107
Convolução	107
4.4.6 A Transformada Rápida de Fourier (FFT)	107
4.5 Filtragem no domínio da freqüência	108
4.5.1 Filtro passa-baixas (FPB)	108
Filtro passa-baixas ideal	108
Filtro passa-baixas Butterworth	111
4.5.2 Filtro passa-altas (FPA)	113
Filtro passa-altas ideal	114
Filtro passa-altas Butterworth	114
4.5.3 Filtragem homomórfica	115
4.6 Processamento de imagens coloridas	118
4.6.1 Conceitos básicos	119
4.6.2 Modelos de representação de cores	121
Modelo RGB	122
O modelo CMY	122
O modelo YIQ	122
O modelo HSI	122
4.6.3 Pseudocolorização	124
4.6.4 Processamento de imagens coloridas <i>full color</i>	125
4.7 Filtros adaptativos	126
4.7.1 Introdução	126
4.7.2 Aspectos Estatísticos	126

4.7.3 Alguns tipos de filtros adaptativos	128
Filtro de Erro Médio Quadrático Mínimo (MMSE - <i>Minimum Mean-Square Error</i>)	128
Filtro de média e mediana com dupla janela (DW-MTM - <i>Double Window-Modified Trimmed Mean</i>)	129
Filtro da Mediana Adaptativo (SAM - <i>Signal Adaptive Median</i>)	131
Exercícios Propostos	133
No computador	134
Na Internet	135
Bibliografia	136
Capítulo 5 - Morfologia Matemática	139
5.1 Introdução	139
5.2 Dilatação e Erosão	139
5.2.1 Definições básicas	140
5.2.2 Dilatação	140
5.2.3 Erosão	142
5.3 Abertura e Fechamento	143
5.3.1 Interpretação geométrica da abertura e do fechamento	144
5.3.2 Propriedades da abertura	144
5.3.3 Propriedades do fechamento	144
5.4 Transformação <i>hit-or-miss</i>	147
5.5 Algoritmos morfológicos básicos	148
5.5.1 Extração de contornos	148
5.5.2 Preenchimento de regiões (<i>Region filling</i>)	149
5.5.3 Extração de componentes conectados	150
5.5.4 Casco convexo (<i>Convex Hull</i>)	151
5.5.5 Afinamento (<i>Thinning</i>)	153
5.5.6 Espessamento (<i>Thickening</i>)	155
5.5.7 Esqueletos	155
5.5.8 Poda (<i>Pruning</i>)	158
Exercícios Propostos	163
No computador	164
Na Internet	164

Bibliografia	164
Capítulo 6 - Compressão e Codificação de Imagens	167
6.1 Fundamentos	167
6.1.1 Redundância de Codificação	168
6.1.2 Redundância Interpixel	170
6.1.3 Redundância Psicovisual	170
6.1.4 Critérios de Fidelidade	171
6.2 Modelos de compressão de imagem	172
6.2.1 O codificador e decodificador de fonte	172
6.2.2 O codificador e decodificador de canal	173
6.3 Elementos de Teoria da Informação	175
6.3.1 Medidas de informação	175
6.3.2 O canal de informação	176
6.3.3 Utilizando a Teoria da Informação	179
6.4 Compressão sem perdas	180
6.4.1 Códigos de palavra-código de comprimento variável	180
Código de Huffman	181
Código de Huffman Truncado	182
Codificação Aritmética	182
Codificação LZW (Lempel-Ziv-Welch)	185
6.4.2 Codificação <i>bit-plane</i>	188
Decomposição <i>bit-plane</i>	188
Codificação de áreas constantes	189
<i>Run-length</i> unidimensional	189
<i>Run-length</i> bidimensional	190
6.4.3 Codificação Preditiva sem Perdas	190
6.5 Compressão com perdas	192
6.5.1 Codificação Preditiva com Perdas	192
Modulação Delta (DM)	193
Modulação por Codificação Diferencial de Pulso (DPCM)	195
A etapa de quantização	195
6.5.2 Codificação por transformadas	196
Seleção de Transformadas	196

6.5.3 Outras técnicas	197
6.6 Padrões de compressão de imagens	197
6.6.1 Padrões CCITT para fac-símiles Grupo 3 e Grupo 4	198
Codificação unidimensional	198
Codificação bidimensional	198
6.6.2 JPEG	203
Características do JPEG	203
Codificador seqüencial	204
DCT (Transformada Discreta de Cossenos)	204
Quantização	205
Ordenação zig-zag	205
Codificador por entropia	205
Decodificador seqüencial	206
Compressão progressiva	207
Codificação seqüencial sem perdas	207
Outros aspectos do JPEG	208
6.6.3 H.261	209
Características do H.261	209
Estrutura de dados	210
Codificador	211
Decodificador	211
6.6.4 H.263	212
6.6.5 MPEG	213
Características do MPEG 1 e 2	214
Estrutura dos quadros MPEG	215
Codificação <i>interframe</i>	216
MPEG-4	218
MPEG-7	218
Exercícios Propostos	219
No computador	221
Na Internet	221
Bibliografia	222

Capítulo 7 - Aspectos práticos de hardware e software para processamento de imagens	225
7.1 O hardware	225
7.1.1 Sensores	225
Sensores a válvula	225
Sensores de estado sólido (linear e de área)	226
7.1.2 <i>Frame grabbers / frame buffers</i>	227
7.1.3 Arquiteturas	228
7.1.4 Dispositivos de saída	229
Monitores de vídeo	229
Impressoras	230
<i>Plotters</i>	231
7.1.5 Dispositivos de armazenamento	232
7.2 O software	232
7.2.1 Títulos disponíveis e classificação	233
Software para Aplicações Científicas	233
Software para Composição de Imagens Animadas	234
Software para Conversão de Formatos	234
Software para Manipulação de Imagens	235
Software para Visualização de Imagens	236
7.2.2 Linguagens e ambientes para desenvolvimento	237
Na Internet	239
Bibliografia	240
Apêndice A - Formatos de arquivos de imagens	243
A.1 Representação através de <i>bitmaps</i> e através de vetores	243
A.1.1 Comparações entre as formas de representação	243
A.1.2 Outras classes de representação	244
A.2 Formatos de Arquivos de Imagem	244
A.2.1 Arquivos de Imagens 2-D	244
A.2.2 Arquivos de Imagens 3-D	257
A.2.3 Arquivos de Animação e Vídeo	259
Na Internet	261
Bibliografia	262

Apêndice B - Roteiros de laboratório de processamento de imagens	263
B.1 Conceitos Introdutórios	263
B.2 Utilizando o MATLAB	263
Operadores matriciais	264
Operadores relacionais	264
Operadores lógicos	264
Caracteres especiais	264
Entrada de variáveis e matrizes	265
Principais funções	266
B.3 Comandos e Funções da <i>Toolbox</i> de Processamento de Imagens	267
B.4 Roteiros de práticas de laboratório	271
Prática 1 - Fundamentos da <i>toolbox</i> de processamento de imagens do MATLAB	272
Prática 2 - Operações lógicas, aritméticas e estatísticas com imagens	276
Prática 3 - Transformações geométricas e verificação de níveis de cinza de pixels	280
Prática 4 - Métodos ponto-a-ponto de realce e análise de imagens	284
Prática 5 - Filtragem no domínio espacial	287
Prática 6 - Filtragem no domínio da freqüência	291
Prática 7 - Morfologia Matemática	294
Na Internet	298
Bibliografia	298
Glossário	299

Prefácio

As áreas de processamento de imagens e visão por computador vêm apresentando expressivo desenvolvimento nas últimas décadas. Tal crescimento pode ser detetado na área acadêmica – onde o assunto é objeto de pesquisas, teses e dissertações nas mais importantes universidades brasileiras e mundiais –, na esfera industrial – onde a cada dia aumenta o número de empresas que produzem, comercializam e utilizam soluções de processamento eletrônico de imagens em seus processos – e na vida cotidiana, com a popularização dos computadores pessoais e das aplicações multimídia.

Profissionais das áreas de Engenharia, Informática, Matemática e Física, dentre outras, estão sendo reciclados para incorporarem os novos conhecimentos oriundos desta área e contam com pequeno número de referências em português sobre o assunto. Além disso, o crescimento espantoso do uso de microcomputadores em ambientes residenciais e a popularização da multimídia e da Internet colaboraram ainda mais para a difusão de informações com forte conteúdo visual e, consequentemente, despertaram também no chamado 'público leigo' a curiosidade de conhecer melhor as técnicas de processamento e manipulação de imagens disponíveis.

Foi por ver todo este crescente interesse em torno destes temas que este livro foi escrito. Com ele, esperamos poder atender as expectativas dos leitores ávidos por informações técnicas precisas e adequadas à exploração deste fantástico universo da imagem.

Sobre o conteúdo e filosofia do livro

Este livro é resultado de quase 10 anos de experiência na docência e pesquisa nas áreas de visão por computador e processamento de imagens e procura refletir o resultado desta experiência, suprindo o leitor com uma obra que aborda temas clássicos e obrigatórios relacionados a esta área de conhecimento – permitindo sua utilização como livro-texto em disciplinas de graduação e pós-graduação – bem como abrindo os horizontes para a exploração de assuntos inovadores de grande interesse tanto para aqueles que estão travando um primeiro contato com o assunto quanto para pesquisadores da área.

Procuramos estruturá-lo de forma didática, clara e agradável, incluindo grande quantidade de imagens ilustrativas das técnicas descritas, exemplos, exercícios resolvidos e propostos, práticas utilizando computador e o programa MATLAB® e sugestões de endereços na Internet para maiores informações sobre cada capítulo. Por filosofia, entendemos que este livro não deveria se limitar a compilar os principais aspectos da área de processamento de imagens, mas deveria ir um pouco além, e ser também um guia de estudo. Por esta razão foram empreendidos esforços para que o leitor perceba que o assunto não se esgota aqui e para guiá-lo na busca de informações adicionais sobre cada tópico. Exemplos destes esforços são as seções Leitura Complementar e Na Internet. Elas trazem indicações bibliográficas precisas para um aprofundamento do assunto assim como apresentam sugestões de *sites* na Internet diretamente relacionados aos temas considerados.

Ainda em função da filosofia do livro, entendemos ser oportuno fazer algumas ressalvas sobre seu conteúdo. As técnicas de processamento de imagens descritas neste livro trabalham fundamentalmente com imagens digitais, monocromáticas e estáticas, com raras exceções (fundamentos de imagens coloridas, apresentados no capítulo 4 e técnicas de compressão de imagens coloridas e seqüências de vídeo, no capítulo 6). Este livro não trata de análise de imagens bi- ou tridimensionais nem detalha técnicas óticas para processamento de imagens. As menções feitas a empresas e produtos de hardware e software são meramente ilustrativas e não têm qualquer caráter comercial. Os produtos mencionados neste livro são marcas registradas de propriedade dos seus respectivos fabricantes.

Finalmente, o livro não traz exemplos de código-fonte para a implementação dos algoritmos descritos, mas contém inúmeras referências bibliográficas e indicações de endereços na Internet para o leitor interessado.

A quem se destina

Este livro é naturalmente dedicado a estudantes de graduação e pós-graduação, professores e pesquisadores das áreas de Engenharia, Informática e correlatas, pelo conteúdo técnico e abordagem didática dos capítulos. Neste caso, houve uma preocupação em adequar seu conteúdo ao programa recomendado pela Sociedade Brasileira de Computação (SBC), que em documento datado de 1996, intitulado "Currículo de Referência da SBC para Cursos de Graduação Plena em Computação", menciona a disciplina "Processamento de Imagens", cujo currículo é completamente coberto pelo conteúdo desta obra.

Serve também a profissionais de diversas áreas, atraídos pela inevitável popularização do uso de técnicas e sistemas de processamento de imagens e pelo aspecto prático do livro, ressaltado particularmente pelos roteiros de experimentos utilizando microcomputador e pelas indicações de endereços na Internet.

Comentários, críticas, sugestões e colaborações para o contínuo aprimoramento de nosso trabalho são bem-vindos e podem ser feitos por e-mail para:

omarques@ieee.org

ou

hugo@daeln.cefetpr.br.

Ogê Marques Filho
Hugo Vieira Neto

Curitiba, Brasil, 1999.

Capítulo 1

Introdução

1.1 Processamento de Imagens: breve histórico e exemplos de aplicações

A área de processamento de imagens vem sendo objeto de crescente interesse por permitir viabilizar grande número de aplicações em duas categorias bem distintas: (1) o aprimoramento de informações pictóricas para interpretação humana; e (2) a análise automática por computador de informações extraídas de uma cena. Ao longo deste livro, reservaremos a expressão 'processamento de imagens' para designar a primeira categoria, adotando os termos 'análise de imagens', 'visão por computador' (ou 'visão computacional') e 'reconhecimento de padrões' para a segunda.

Uma das primeiras aplicações na primeira categoria remonta ao começo deste século, onde buscavam-se formas de aprimorar a qualidade de impressão de imagens digitalizadas transmitidas através do sistema Bartlane de transmissão de imagens por cabo submarino entre Londres e Nova Iorque. Os primeiros sistemas Bartlane, no início da década de 20, codificavam uma imagem em cinco níveis de intensidade distintos. Esta capacidade seria expandida, já em 1929, para 15 níveis, ao mesmo tempo em que era desenvolvido um método aprimorado de revelação de filmes através de feixes de luz modulados por uma fita que continha informações codificadas sobre a imagem.

Mas o grande impulso para a área de Processamento de Imagens viria cerca de três décadas mais tarde, com o advento dos primeiros computadores digitais de grande porte e o início do programa espacial norte-americano. O uso de técnicas computacionais de aprimoramento de imagens teve início no *Jet Propulsion Laboratory* (Pasadena, California - EUA)¹ em 1964, quando imagens da lua transmitidas por uma sonda Ranger² eram processadas por computador para corrigir vários tipos de distorção inerentes à câmera de TV acoplada à sonda. Estas técnicas serviram de base para métodos aprimorados de realce e restauração de imagens de outros programas espaciais posteriores, como as expedições tripuladas da série Apollo, por exemplo.

De 1964 aos dias atuais, a área de processamento de imagens vem apresentando crescimento expressivo e suas aplicações permeiam quase todos os ramos da atividade humana. Em Medicina, o uso de imagens no diagnóstico médico tornou-se rotineiro e os avanços em processamento de imagens vêm permitindo tanto o desenvolvimento de novos equipamentos quanto a maior facilidade de interpretação de imagens produzidas por equipamentos mais antigos, como por exemplo o de raio X. Em Biologia, a capacidade de processar automaticamente imagens obtidas de microscópios, por exemplo contando o número de células de um certo tipo presentes em uma imagem, facilita sobremaneira a execução de tarefas laboratoriais com alto grau de precisão e repetibilidade. O processamento e a interpretação automática de imagens captadas por satélites auxiliam os trabalhos nas áreas de Geografia, Sensoriamento Remoto, Geoprocessamento e Meteorologia, dentre outras. Técnicas de restauração de imagens auxiliam arqueologistas a recuperar fotos borradass de artefatos raros, já destruídos. O uso de robôs dotados de visão artificial em tarefas tais como controle de qualidade em linhas de produção aumenta a cada ano, num cenário de crescente automação industrial. Inúmeras outras áreas tão distintas como Astronomia, Segurança, Publicidade e Direito – para citar apenas algumas – vêm sendo beneficiadas com os avanços nas áreas de processamento de imagens e visão por computador.

¹ "<http://www.jpl.nasa.gov>"

² "<http://www.jpl.nasa.gov/missions/ranger/>"

Leitura complementar

Mascarenhas [Mascarenhas 1990] apresenta um abrangente texto introdutório aos assuntos abordados neste livro.

1.2 Um sistema de processamento de imagens e seus componentes

Os elementos de um sistema de processamento de imagens de uso genérico são mostrados na figura 1. Este diagrama permite representar desde sistemas de baixo custo até sofisticadas estações de trabalho utilizadas em aplicações que envolvem intenso uso de imagens. Ele abrange as principais operações que se pode efetuar sobre uma imagem, a saber: aquisição, armazenamento, processamento e exibição. Além disso, uma imagem pode ser transmitida à distância utilizando meios de comunicação disponíveis. Todas estas operações são descritas a seguir.

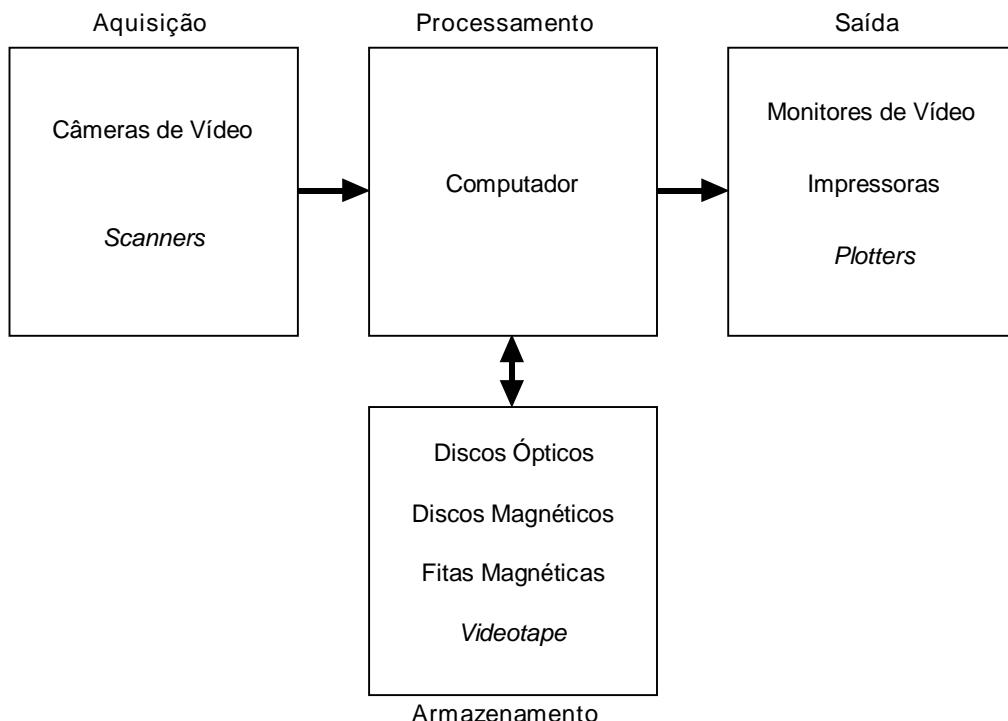


Figura 1 - Elementos de um sistema de processamento de imagens.

1.2.1 Aquisição

A etapa de aquisição tem como função converter uma imagem em uma representação numérica adequada para o processamento digital subsequente. Este bloco compreende dois elementos principais. O primeiro é um dispositivo físico sensível a uma faixa de energia no espectro eletromagnético (como raio X, ultravioleta, espectro visível ou raios infravermelhos), que produz na saída um sinal elétrico proporcional ao nível de energia detetado. O segundo – o digitalizador propriamente dito – converte o sinal elétrico analógico em informação digital, isto é, que pode ser representada através de bits 0s e 1s. Um módulo de aquisição de imagens é normalmente conhecido pelo nome de *frame grabber*. Os capítulos 2 e 7 deste livro trazem mais detalhes sobre os aspectos envolvidos na aquisição de imagens digitais.

1.2.2 Armazenamento

O armazenamento de imagens digitais é um dos maiores desafios no projeto de sistemas de processamento de imagens, em razão da grande quantidade de bytes necessários para tanto. Este armazenamento pode ser dividido em três categorias: (1) armazenamento de curta duração de uma imagem, enquanto ela é utilizada nas várias etapas do processamento, (2) armazenamento

de massa para operações de recuperação de imagens relativamente rápidas, e (3) arquivamento de imagens, para recuperação futura quando isto se fizer necessário. O espaço de armazenamento requerido é normalmente especificado em bytes (8 bits) e seus múltiplos: KB (kilobyte \approx 1000 bytes), MB (megabyte \approx 1 milhão de bytes), GB (gigabyte \approx 1 bilhão de bytes) e TB (terabyte \approx 1 trilhão de bytes). O capítulo 7 discute aspectos de armazenamento de imagens em maior profundidade.

Para o armazenamento de curta duração, a alternativa mais simples é utilizar parte da memória RAM do computador principal. Outra opção consiste no uso de placas especializadas, chamadas *frame buffers*, que armazenam uma ou mais imagens completas e podem ser acessadas a uma alta velocidade, tipicamente 30 imagens completas por segundo. O uso de *frame buffers* permite também que operações de *zoom* (ampliação ou redução para fins de visualização), *scroll* (rolagem na vertical) e *pan* (rolagem na horizontal) sejam executadas de forma praticamente instantânea. Placas *frame buffers* disponíveis no mercado atualmente apresentam capacidade de armazenamento na faixa de alguns MB de memória.

A segunda categoria de armazenamento normalmente requer o uso de discos magnéticos de no mínimo algumas centenas de MB e recentemente passou a utilizar também discos magneto-ópticos, por vezes agrupados em *jukeboxes* contendo de 30 a 100 discos. Nesta categoria o fator 'tempo de acesso' é tão ou mais importante que a capacidade (em bytes) do meio de armazenamento. Através de cálculos simples (n^o de pixels na horizontal x n^o de pixels na vertical x n^o de bits necessários para a escala de cinza / 8), pode-se estimar a quantidade de bytes necessários para armazenar uma imagem monocromática em disco. Este cálculo entretanto considera uma imagem representada como uma matriz, cujos elementos são os valores de tons de cinza dos respectivos pixels.³ Na prática, informações adicionais (tamanho da imagem e número de cores ou tons de cinza, no mínimo) são necessárias. Estas informações costumam ser colocadas em um cabeçalho (*header*) no início do arquivo. Infelizmente, não existe um único cabeçalho ou formato de armazenamento de imagens padronizados. Alguns dos formatos mais comuns são o BMP, PCX, TIFF, JPEG e GIF. Estes formatos de arquivos de imagem, além de muitos outros, são apresentados no Apêndice A.

Finalmente, o arquivamento de imagens é caracterizado por quantidades gigantescas de bytes contendo imagens cuja recuperação é esporádica. Nesta categoria, as fitas magnéticas estão dando lugar aos discos ópticos WORM (*Write-Once-Read-Many*), com capacidade que pode chegar a mais de 10 GB por disco, e que também podem ser agrupados em *jukeboxes*, com capacidade total de armazenamento superior a 1 TB.

1.2.3 Processamento

O processamento de imagens digitais envolve procedimentos normalmente expressos sob forma algorítmica. Em função disto, com exceção das etapas de aquisição e exibição, a maioria das funções de processamento de imagens pode ser implementada via software. O uso de hardware especializado para processamento de imagens somente será necessário em situações nas quais certas limitações do computador principal (por exemplo, velocidade de transferência dos dados através do barramento) forem intoleráveis.

A tendência atual do mercado de hardware para processamento de imagens é a comercialização de placas genéricas compatíveis com os padrões de barramento consagrados pelas arquiteturas mais populares de microcomputadores e estações de trabalho. O software de controle destas placas é que determinará sua aplicação específica a cada situação. As vantagens mais imediatas são: redução de custo, modularidade, reutilização de componentes de software em outra aplicação rodando sobre o mesmo hardware e independência de fornecedor. Convém notar, entretanto, que sistemas dedicados continuam sendo produzidos e comercializados para atender a tarefas específicas, tais como processamento de imagens transmitidas por satélites.

³ Para imagens coloridas, a situação é um pouco mais complexa. Normalmente estes números serão índices (endereços) de uma tabela de cores, denominada palheta ou *palette*. Para maiores detalhes, veja a seção 4.6 e o Resumo da Teoria da Prática de Laboratório nº 1.

1.2.4 Transmissão

Imagens digitalizadas podem ser transmitidas à distância utilizando redes de computadores e protocolos de comunicação já existentes. O grande desafio da transmissão de imagens à distância é a grande quantidade de bytes que se necessita transferir de uma localidade a outra, muitas vezes através de canais de comunicação de baixa velocidade e banda passante estreita. Este problema é ainda mais sério quando se deseja transmitir seqüências de vídeo (imagens em movimento com áudio associado) em tempo real, onde outros fatores, como por exemplo sincronização, devem ser considerados. Nestes casos, o uso de técnicas de compressão e descompressão de imagens, como as descritas no capítulo 6, é mandatório.

1.2.5 Exibição

O monitor de vídeo é um elemento fundamental de um sistema de processamento de imagens. Os monitores em uso atualmente são capazes de exibir imagens com resolução de pelo menos 640 x 480 pixels com 256 cores distintas. A tecnologia mais usual ainda é o TRC (Tubo de Raios Catódicos).

Um TRC para um sistema de processamento de imagens normalmente segue um padrão de vídeo. O padrão de vídeo mais comum para sistemas monocromáticos é o RS-170. Ele prevê 480 linhas horizontais entrelaçadas, isto é, a varredura de um quadro é feita em duas etapas, abrangendo primeiramente as linhas ímpares e posteriormente as linhas pares. Cada uma destas etapas é denominada campo. O tempo necessário para percorrer um campo é 1/60 s; consequentemente, o tempo total de um quadro é 1/30 s. As características de persistência visual do olho humano fazem com que, nesta velocidade, a varredura individual de cada campo não seja perceptível, bem como dão a impressão de que a seqüência de quadros explorados é perfeitamente contínua.

O padrão RS-170 especifica resolução vertical de 480 pixels, sendo a resolução horizontal determinada pelos circuitos eletrônicos do monitor e pelo tamanho dos pontos de fósforo na tela. Este número costuma ser 512, proporcionando imagens de 512 x 480 pixels.

A resolução espacial dos monitores é normalmente especificada em pontos por polegada (*dots per inch - dpi*). Um valor típico de resolução é 72 dpi, suficiente para exibir uma imagem de 1024 x 1024 pixels em um monitor de 19 polegadas ou uma imagem de 640 x 400 pontos em uma tela cuja diagonal meça 12 polegadas. A título de comparação, uma tela de TV tem resolução na faixa de 40 dpi.

Um TRC colorido difere radicalmente de seu antecessor monocromático, por apresentar três feixes eletrônicos, cada um correspondente a uma das três cores primárias (vermelho, verde e azul). A superfície interna da tela é constituída por três tipos de fósforo, dispostos de forma triangular, cada qual sensível a uma das cores primárias e excitado pelo respectivo canhão eletrônico. Isto significa dizer que, do ponto de vista construtivo, cada pixel é na verdade uma combinação de três pequenos pixels, um para cada cor primária.

A indústria de dispositivos de exibição vem apresentando sistematicamente novas tecnologias de fabricação de monitores de vídeo, dentre eles os monitores de cristal líquido (LCD), cada vez mais populares graças à disseminação dos computadores portáteis (*notebooks*).

Existem diversas formas de reprodução de imagens em papel. A melhor, e mais cara, é a reprodução fotográfica, onde o número de gradações de cinza é função da densidade dos grânulos de prata no papel. Outra possibilidade é o uso de papel sensível a temperatura, cuja composição química faz com que ele apresente coloração mais escura à medida que a temperatura aumenta. Este tipo de impressão ainda é o mais difundido em equipamentos de fax. Uma de suas desvantagens é o desvanecimento das imagens com o tempo. Nos últimos anos aumentou consideravelmente a oferta de impressoras térmicas coloridas no mercado. Estas impressoras baseiam-se na deposição de cera colorida sobre um papel especial para produzir a impressão. O capítulo 7 apresenta maiores informações sobre dispositivos de exibição e impressão de imagens.

Dispositivos periféricos de saída especializados na produção de cópias da imagem em forma de fotografias, slides ou transparências também estão se tornando cada vez mais usuais.

Uma alternativa às técnicas fotográficas consiste no uso de técnicas de *halftoning*. É o método usado por jornais e por impressoras convencionais (laser, matriciais ou a jato de tinta) para a impressão de imagens. Esta técnica consiste basicamente em imprimir pontos escuros de diferentes tamanhos, espaçados de tal maneira a reproduzir a ilusão de tons de cinza. À medida que a distância entre o observador e a imagem impressa aumentam, os detalhes finos vão desaparecendo e a imagem parece cada vez mais uma imagem contínua monocromática.

No jargão computacional, dá-se o nome de *dithering* ao processo de produção do efeito de *halftoning*, bem como a todas as técnicas de conversão de uma imagem para adaptá-la a resoluções menores, tanto para efeito de exibição como para impressão. Existem vários algoritmos de *dithering*, sendo o mais comum o de Floyd-Steinberg, que consiste de um processo adaptativo no qual o padrão de *dither* a ser atribuído a um pixel depende de seu tom de cinza e de seus vizinhos. Uma discussão mais detalhada destes algoritmos foge ao escopo deste livro.

Leitura complementar

O capítulo 7 de [Lindley 1991] e o capítulo 11 de [Rimmer 1993] trazem explicações teóricas e código-fonte em C para impressão de imagens monocromáticas e/ou coloridas utilizando *dithering*.

O capítulo 10 de [Dougherty 1994] é uma excelente referência para um estudo mais aprofundado dos conceitos e técnicas de *halftoning*.

1.3 O sistema visual humano

A figura 2 mostra um corte horizontal do olho humano. O globo ocular tem formato aproximadamente esférico e um diâmetro de cerca de 20 mm. Ele é envolvido por três membranas: a camada externa formada pela córnea e pela esclerótica, a coróide e a retina, que é a camada interna. A córnea é uma película transparente que cobre a parte anterior do olho. Dando continuidade à córnea, a esclerótica é uma membrana opaca que reveste o globo ocular.

A coróide está situada abaixo da esclerótica. Essa membrana contém uma rede de vasos sanguíneos que servem como a principal fonte de nutrição do olho. O revestimento da coróide é fortemente pigmentado, o que ajuda a reduzir a quantidade de luz que entra no olho. Ela é dividida em corpo ciliar e diafragma da íris, sendo este último responsável por controlar a quantidade de luz que deve penetrar no olho. O diâmetro da abertura central da íris (pupila) varia entre 2 mm e 8 mm. A parte frontal da íris contém o pigmento visível do olho, enquanto sua porção posterior possui um pigmento negro.

A membrana mais interna do olho é a retina, situada na sua parede posterior. Quando o olho focaliza uma cena, a imagem correspondente é projetada sobre a retina, na qual estão distribuídos dois tipos de receptores de luz discretos: os cones e os bastonetes. Os cones são em número de 6 a 7 milhões em cada olho e estão localizados na porção central da retina, chamada de fóvea. Eles são altamente sensíveis a cor e cada qual está conectado a uma terminação nervosa dedicada. O número de bastonetes é bastante maior, cerca de 75 a 150 milhões, distribuídos em toda a superfície da retina. Os bastonetes servem para dar uma visão geral da imagem captada no campo de visão. Eles não distinguem cores, mas são sensíveis a baixos níveis de iluminação.

A fóvea é uma reentrância circular na retina com aproximadamente 1,5 mm de diâmetro. De forma aproximada, podemos considerá-la um sensor de área quadrada de 1,5 mm por 1,5 mm. A densidade de cones nesta área da retina é de aproximadamente 150.000 elementos por mm². Baseando-se nessas aproximações, o número de cones na região de maior acuidade do olho é de aproximadamente 337.000 elementos. Para efeito comparativo, esta resolução é facilmente alcançada pela tecnologia atual, usando CCD de área não superior a 7 mm x 7 mm.

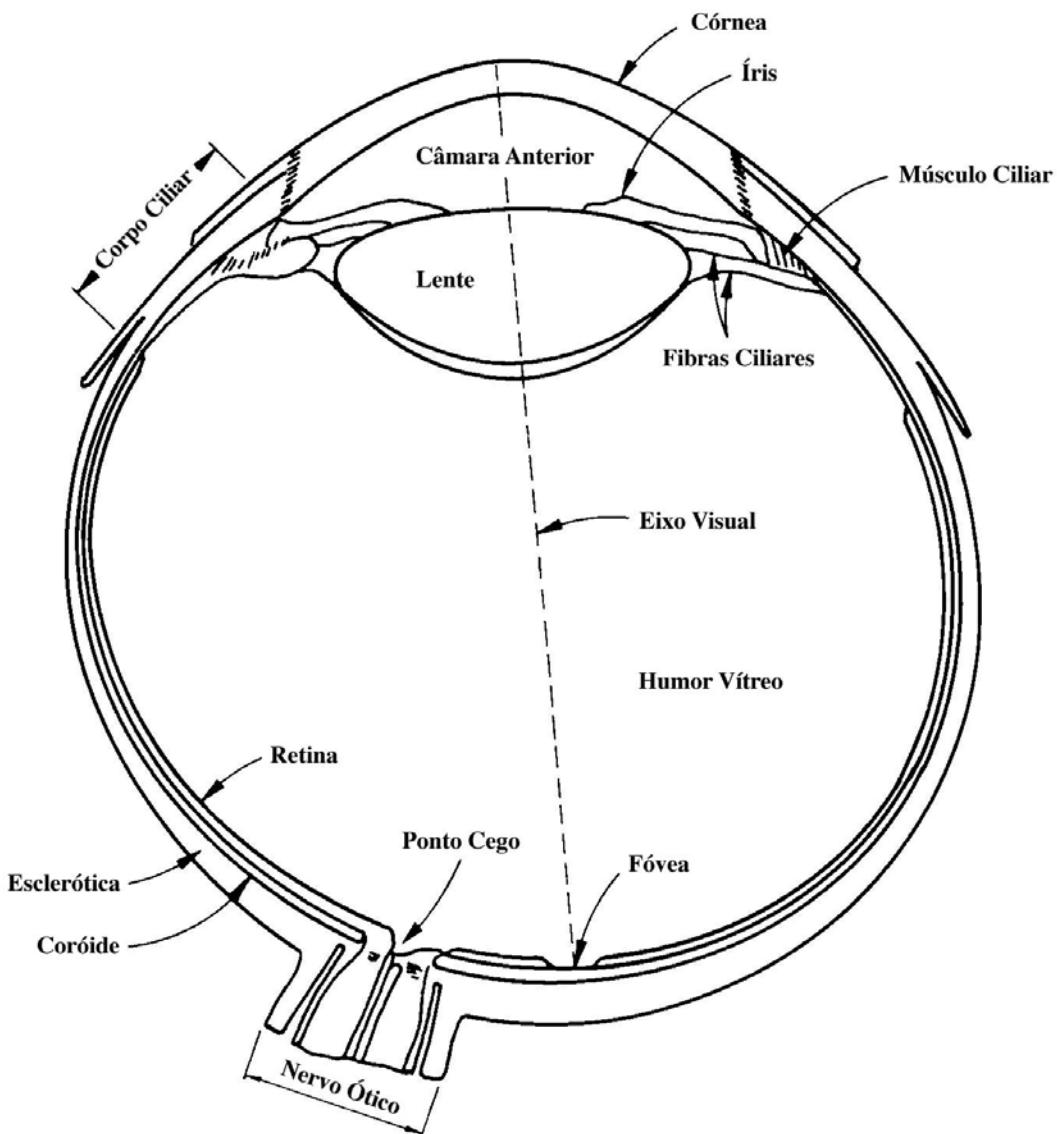


Figura 2 - Vista em corte do olho humano (adaptado de Gonzalez, R.C., Woods, R.E., *Digital Image Processing*, Addison-Wesley, 1992).

Dentre as características do sistema visual humano relevantes para o projeto de sistemas de visão artificial, podemos destacar a enorme faixa de valores de intensidade luminosa (da ordem de 10^{10}) aos quais o olho pode se adaptar e a relação aproximadamente logarítmica entre os níveis de intensidade luminosa presentes na cena e a sensação subjetiva de brilho, ilustrada na figura 3. Convém notar, entretanto, que o olho humano não pode operar sobre toda esta faixa simultaneamente. Ao invés disso, ele excursiona ao longo de toda esta faixa através de adaptações em sua sensibilidade global, um fenômeno conhecido como adaptação de brilho. A faixa total de valores de intensidade que o olho pode discriminar simultaneamente é comparativamente pequena em relação à faixa total de adaptação. Para um certo conjunto de condições, o nível de sensibilidade atual do sistema visual é chamado nível de adaptação de brilho, como por exemplo o valor B_a na figura 3. A pequena porção de curva que intercepta a curva principal representa a faixa de brilho subjetivo que o olho pode perceber quando adaptado a este nível. Esta faixa é bastante restrita, existindo um nível de brilho B_b abaixo do qual todos os estímulos são indistinguíveis. O trecho tracejado da curva não é, na verdade, restrito, mas se estendido além de um limite perde seu sentido, porque neste caso o olho humano buscaria outro nível de adaptação maior que B_a .

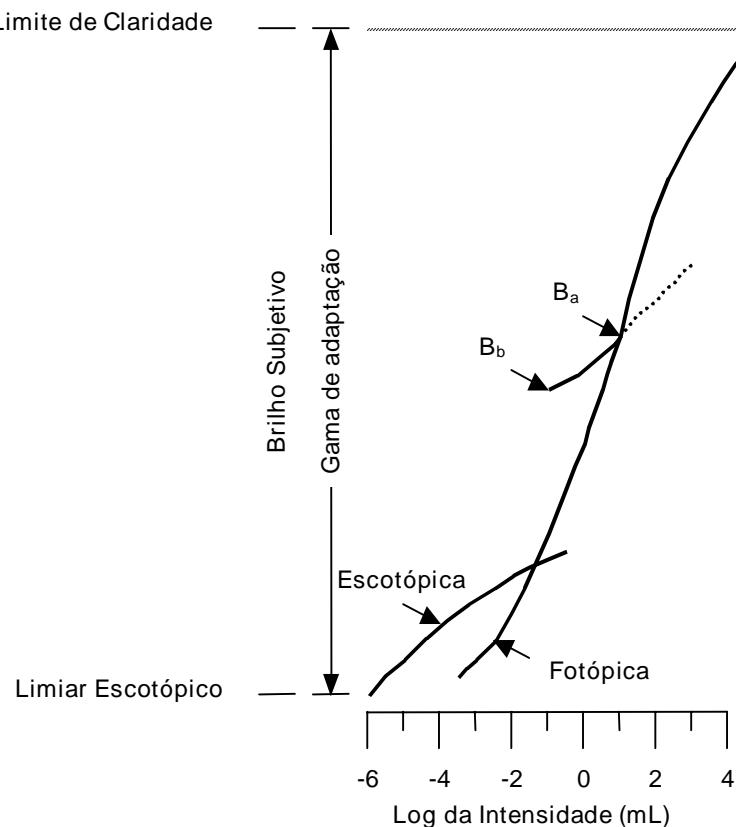


Figura 3 - Curva que relaciona a sensação subjetiva de brilho com o logaritmo da intensidade luminosa incidente sobre o olho humano.

A tabela 1 apresenta uma visão comparativa entre o sistema visual humano e um sistema de visão artificial.

Leitura complementar

Para maiores informações sobre aspectos do processo de percepção visual humana e suas implicações no projeto de sistemas de visão artificial, uma referência obrigatória é [Marr 1982].

A seção 2.1 de [Gonzalez e Woods 1992] apresenta interessantes fenômenos de percepção visual tais como contraste simultâneo, discriminação de brilho usando a razão de Weber e o padrão de faixas de Mach.

1.4 Sistemas de Visão Artificial: fundamentos e desafios

Nas primeiras páginas de seu livro *Digital Image Processing and Computer Vision*, Robert J. Schalkoff [Schalkoff 1989] formula uma sugestiva pergunta quando questiona: "O que estamos tentando fazer e por que isto é tão difícil?"

Uma resposta simplista à primeira parte da pergunta acima é: "Estamos tentando ensinar robôs a enxergar." Ao relacionarmos as dificuldades inerentes ao processo de dotar o computador de uma capacidade visual semelhante à dos seres humanos, deparamo-nos com três admiráveis características do processo de percepção visual humano, que são:

- uma base de dados muito rica;
- altíssima velocidade de processamento; e
- a capacidade de trabalhar sob condições muito variadas.

Os avanços na tecnologia de dispositivos de armazenamento de massa e o surgimento de novas CPUs e arquiteturas computacionais cada vez mais rápidas, com alto grau de paralelismo, nos induzem a crer que dispomos de condições cada vez melhores de modelar as duas primeiras características relacionadas acima. O grande desafio permanece sendo o de fazer com que os sistemas de visão artificial trabalhem em diferentes condições de luminosidade, contraste,

posicionamento relativo dos objetos em uma cena sem perder a capacidade de interpretar a cena, de forma análoga a nossa capacidade de reconhecer um amigo ou parente com relativa facilidade, independentemente de ele estar usando óculos ou não, ter deixado crescer a barba ou estar no carro ao lado do nosso em uma esquina num final de tarde, onde não dispomos de outra imagem senão a vista de perfil e onde as condições de luminosidade são bastante inferiores às que obteríamos ao meio-dia.

Tabela 1 - Comparação entre o sistema visual humano e um sistema de visão artificial.

	Sistema visual humano	Sistema de visão artificial
Espectro	Limitado à faixa de luz visível (300 nm a 700 nm) do espectro de ondas eletromagnéticas.	Pode operar em praticamente todo o espectro de radiações eletromagnéticas, dos raios X ao infravermelho.
Flexibilidade	Extremamente flexível, capaz de se adaptar a diferentes tarefas e condições de trabalho.	Normalmente inflexível, apresenta bom desempenho somente na tarefa para a qual foi projetado.
Habilidade	Pode estabelecer estimativas relativamente precisas em assuntos subjetivos.	Pode efetuar medições exatas, baseadas em contagem de pixels e, portanto, dependentes da resolução da imagem digitalizada.
Cor	Possui capacidade de interpretação subjetiva de cores.	Mede objetivamente os valores das componentes R, G e B para determinação de cor.
Sensibilidade	Capaz de se adaptar a diferentes condições de luminosidade, características físicas da superfície do objeto e distância ao objeto. Limitado na distinção de muitos níveis diferentes de cinza, simultaneamente.	Sensível ao nível e padrão de iluminação, bem como à distância em relação ao objeto e suas características físicas. Pode trabalhar com centenas de tons de cinza, conforme projeto do digitalizador.
Tempo de resposta	Elevado, da ordem de 0,1 s.	Dependente de aspectos de hardware, podendo ser tão baixo quanto 0,001 s.
2-D e 3-D	Pode executar tarefas 3-D e com múltiplos comprimentos de onda (dentro do espectro de luz visível) facilmente.	Executa tarefas 2-D com relativa facilidade, mas é lento e limitado em tarefas 3-D.
Percepção	Percebe variações de brilho em escala logarítmica. A interpretação subjetiva de brilho depende da área ao redor do objeto considerado.	Pode perceber brilho em escala linear ou logarítmica.

1.4.1 Estrutura de um Sistema de Visão Artificial

Definiremos um Sistema de Visão Artificial (SVA) como um sistema computadorizado capaz de adquirir, processar e interpretar imagens correspondentes a cenas reais. A figura 4 mostra esquematicamente um diagrama de blocos de um SVA. Suas principais etapas são explicadas a seguir, partindo da premissa de que um problema prático, por exemplo a leitura do Código de Endereçamento Postal (CEP) de um lote de envelopes, deve ser solucionado.

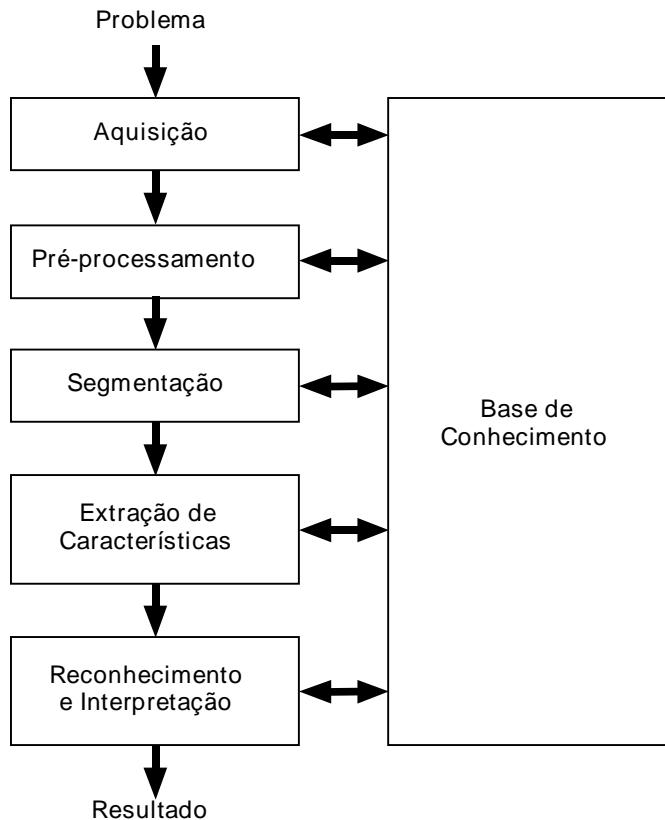


Figura 4 - Um Sistema de Visão Artificial (SVA) e suas principais etapas.

1.4.2 Domínio do problema e resultado

O domínio do problema, neste caso, consiste no lote de envelopes e o objetivo do SVA é ler o CEP presente em cada um deles. Logo, o resultado esperado é uma seqüência de dígitos correspondentes ao CEP lido.

1.4.3 Aquisição da imagem

O primeiro passo no processo é a aquisição de imagens dos envelopes. Para tanto são necessários um sensor e um digitalizador. O sensor converterá a informação óptica em sinal elétrico e o digitalizador transformará a imagem analógica em imagem digital.

Dentre os aspectos de projeto envolvidos nesta etapa, pode-se mencionar: a escolha do tipo de sensor, o conjunto de lentes a utilizar, as condições de iluminação da cena, os requisitos de velocidade de aquisição (assumindo que os envelopes passariam pela frente da câmera a uma certa velocidade), a resolução e o número de níveis de cinza da imagem digitalizada, dentre outros. Esta etapa produz à saída uma imagem digitalizada do envelope.

1.4.4 Pré-processamento

A imagem resultante do passo anterior pode apresentar diversas imperfeições, tais como: presença de pixels ruidosos, contraste e/ou brilho inadequado, caracteres (especialmente os dígitos do CEP) interrompidos ou indevidamente conectados etc. A função da etapa de pré-

processamento é aprimorar a qualidade da imagem para as etapas subsequentes. As operações efetuadas nesta etapa são ditas de baixo nível porque trabalham diretamente com os valores de intensidade dos pixels, sem nenhum conhecimento sobre quais deles pertencem aos dígitos do CEP, a outras informações impressas no envelope ou ao fundo. A imagem resultante desta etapa é uma imagem digitalizada de melhor qualidade que a original.

1.4.5 Segmentação

A tarefa básica da etapa de segmentação é a de dividir uma imagem em suas unidades significativas, ou seja, nos objetos de interesse que a compõem. Esta tarefa, apesar de simples de descrever, é das mais difíceis de implementar.

No caso específico do problema do CEP, é possível que o problema seja dividido em duas etapas: em um primeiro momento os algoritmos de segmentação tentarão localizar o CEP do restante das informações para posteriormente, trabalhando sobre esta subimagem, segmentar cada dígito individualmente. Segundo esta linha de raciocínio, este bloco produzirá à saída oito subimagens, cada qual correspondendo a um dígito do CEP.

1.4.6 Extração de Características

Esta etapa procura extrair características das imagens resultantes da segmentação através de descritores que permitam caracterizar com precisão cada dígito e que apresentem bom poder de discriminação entre dígitos parecidos, como o '5' e o '6'. Estes descritores devem ser representados por uma estrutura de dados adequada ao algoritmo de reconhecimento. É importante observar que nesta etapa a entrada ainda é uma imagem, mas a saída é um conjunto de dados correspondentes àquela imagem.

Para maior clareza, suponhamos que os descritores utilizados para descrever um caractere sejam as coordenadas normalizadas x e y de seu centro de gravidade e a razão entre sua altura e largura. Neste caso, um vetor de três elementos é uma estrutura de dados adequada para armazenar estas informações sobre cada dígito processado por esta etapa.

1.4.7 Reconhecimento e Interpretação

Nesta última etapa do sistema, denominamos reconhecimento o processo de atribuição de um rótulo a um objeto baseado em suas características, traduzidas por seus descritores. A tarefa de interpretação, por outro lado, consiste em atribuir significado a um conjunto de objetos reconhecidos. Neste exemplo, uma forma simples de interpretação seria a verificação do CEP em uma base de dados de CEPs válidos, para descobrir se o conjunto dos oito caracteres (cada qual reconhecido individualmente) faz sentido ou não.

1.4.8 Base de Conhecimento

Todas as tarefas das etapas descritas acima pressupõem a existência de um conhecimento sobre o problema a ser resolvido, armazenado em uma base de conhecimento, cujo tamanho e complexidade podem variar enormemente. Idealmente, esta base de conhecimento deveria não somente guiar o funcionamento de cada etapa, mas também permitir a realimentação entre elas. Por exemplo, se a etapa de representação e descrição recebesse 7 caracteres ao invés de 8, ela deveria ser capaz de realimentar a etapa de segmentação (provável responsável pela falha) para que esta procurasse segmentar novamente a subimagem 'suspeita' (aquele de maior largura), buscando dividi-la em duas. Esta integração entre as várias etapas através da base de conhecimento ainda é um objetivo difícil de alcançar e não está presente na maioria dos SVAs existentes atualmente.

Finalmente, cumpre observar que nem todos os SVAs possuem todos estes blocos e que a maioria das técnicas descritas neste livro estão delimitadas até o bloco 'pré-processamento'.

1.5 Estrutura e escopo do livro

Este livro está estruturado em sete capítulos, sendo os dois primeiros dedicados à apresentação de conceitos fundamentais que servirão de base para a compreensão dos capítulos seguintes. O capítulo 2 apresenta os principais aspectos das imagens digitais, caracterizando-as sob o ponto MARQUES FILHO, Ogê; VIEIRA NETO, Hugo. *Processamento Digital de Imagens*, Rio de Janeiro: Brasport, 1999. ISBN 8574520098.

de vista matemático, apresentando suas principais propriedades e ilustrando as principais operações lógicas, aritméticas e transformações geométricas que se pode efetuar sobre elas. O capítulo 3 define histograma de uma imagem e apresenta diversas técnicas de modificação de histograma.

Os capítulos 4, 5 e 6 tratam das técnicas de pré-processamento aplicadas a uma imagem. O capítulo 4 abrange as principais técnicas de filtragem de imagens no domínio espacial ou freqüencial, tanto com o objetivo de realçá-las como de remover ruído eventualmente presente sobre elas. Também são apresentados conceitos de processamento de imagens coloridas e de filtragem adaptativa. Já o capítulo 5 é inteiramente dedicado a técnicas de processamento e análise de imagens baseadas em morfologia matemática. Por sua vez, o capítulo 6 trata de um tema importante e atual que é a compressão de imagens, desde os fundamentos conceituais até os padrões mais utilizados atualmente.

O capítulo 7 serve como referência prática para o leitor interessado em implementar seu próprio sistema de processamento de imagens, abrangendo informações conceituais e práticas sobre o hardware e software disponível para tanto.

O livro contém ainda dois apêndices. O apêndice A apresenta uma visão resumida, porém abrangente, dos principais formatos de arquivos de imagens disponíveis atualmente. O apêndice B traz um conjunto de roteiros de práticas de laboratório utilizando a *toolbox* de Processamento de Imagens do software MATLAB®. Estas práticas são referenciadas nos capítulos correspondentes ao longo do livro.

Exercícios Propostos

1. Seja o diagrama da figura 4, destacando as principais etapas de um sistema de visão artificial. Supondo que se deseje utilizar os conceitos de análise de imagens para reconhecer placas de veículos à entrada e/ou saída de um estacionamento automatizado, comente quais as tarefas que cada bloco desempenhará na solução do problema, indicando os principais aspectos práticos envolvidos e seu impacto no projeto do sistema completo.

2. Em nosso estudo de visão por computador, vimos que as dificuldades em simular o sistema visual humano residem em três frentes:

- Base de dados muito extensa
- Velocidade de processamento muito alta
- Condições de trabalho muito variadas

Explique cada uma delas e comente o grau de dificuldade relativa de cada uma, bem como a relação entre os progressos tecnológicos em software e hardware e seu impacto na minimização destas dificuldades.

3. Qual a diferença entre os termos 'processamento de imagens' e 'reconhecimento de padrões'?

Na Internet⁴

Para maiores informações sobre alguns dos tópicos deste capítulo, sugerimos as páginas a seguir:

"<http://www.jpl.nasa.gov>"

NASA Jet Propulsion Laboratory

Home page do *Jet Propulsion Lab* da NASA.

⁴ Convém lembrar que a referência a informações disponíveis na Internet está sujeita a problemas decorrentes da natureza extremamente dinâmica da rede. É possível que alguns links indicados neste livro sejam removidos de seus servidores, transferidos para outros servidores ou estejam temporariamente fora do ar.

"<http://www.jpl.nasa.gov/missions/ranger/>"

Ranger: Mission to the Moon

Apresenta detalhes técnicos e históricos das missões Ranger.

Para aprofundar ou complementar outros assuntos tratados neste livro, existem inúmeras páginas úteis e interessantes relacionadas às áreas de processamento de imagens e visão por computador na *World Wide Web*. Relacionamos a seguir algumas delas, classificando-as em seis grupos, a saber:

- Top 10: dez páginas selecionadas pelos autores por apresentarem grande quantidade de informação e inúmeros *links* a outros *sites* de interesse.
- Grupos de pesquisa: páginas com *links* para grupos de pesquisa em visão computacional e processamento de imagens no Brasil e no exterior.⁵
- Publicações: neste grupo estão incluídas informações técnicas sobre processamento de imagens e tópicos correlatos, incluindo páginas relativas a periódicos científicos (*journals*), *sites* de revistas técnicas, páginas de perguntas mais freqüentes (FAQs), cursos interativos disponíveis na WWW, (trechos de) livros disponíveis *on-line*, notas de aula de professores da área, resenhas de livros etc.
- Eventos: contém *links* para as páginas dos principais simpósios e conferências mundiais na área.
- Imagens: *sites* contendo imagens para teste e ilustração de resultados de algoritmos.

Top 10

"<http://www.cs.cmu.edu/~cil/vision.html>"

Computer Vision Home Page

Excelente ponto de partida para pesquisas na WWW. Desdobra-se em várias páginas específicas, cada qual contendo inúmeros *links* de interesse.

"<http://www.ph.tn.tudelft.nl/PRInfo.html>"

Pattern Recognition Information

Página de referência para assuntos relacionados ao reconhecimento de padrões e temas correlatos.

"<http://www.lpac.ac.uk/SEL-HPC/Articles/VisionArchive.html>"

SEL-HPC Vision and Image Processing Archive

Repositório de artigos técnicos na área de processamento de imagens. Pode ser atualizado dinamicamente.

"<http://iris.usc.edu/Vision-Notes/bibliography/contents.html>"

USC Annotated Computer Vision Bibliography

Extensa e bem estruturada compilação de quase tudo o que já foi publicado na área. Um *bookmark* obrigatório.

"<http://ecvnet.lira.dist.unige.it>"

ECVNet

Relata atividades de um consórcio europeu na área de visão computacional e contém diversos *links* úteis.

⁵ Se o leitor for pesquisador da área, procure verificar se seu grupo de pesquisa está cadastrado nestes *sites*.

"<http://viswiz.gmd.de/MultimediaInfo>"

Multimedia Info & Resources

Excelente *site* para pesquisa de tópicos ligados a sistemas multimídia, processamento de sons, imagens e vídeo e temas correlatos.

"http://reality.sgi.com/employees/rchiang_esd/TI-ImageProc.html"

Technical Information - Image Processing

Outra ótima página repleta de *links* classificados de forma razoavelmente estruturada.

"<http://peipa.essex.ac.uk/>"

The Pilot European Image Processing Archive: Home Page

Repositório de arquivos de interesse nas áreas de processamento de imagens e visão por computador, mantido pela *University of Essex* (Inglaterra).

"<http://www.vision1.com/>"

The Vision and Imaging Technology Resource!

Guia técnico-comercial de produtos, serviços e recursos para desenvolvedores de soluções em visão computacional. Possui uma extensa biblioteca de *links*.

"<http://vision.arc.nasa.gov/VisionScience/VisionScience.html>"

Vision Science: The World-Wide Web Virtual Library

Guia de referência para pesquisa de temas relacionados a visão biológica (principalmente) e computacional na Web.

Grupos de pesquisa

"<http://www.cs.cmu.edu/~cil/v-groups.html>"

Computer Vision: Research Groups

Página com *links* para os principais grupos de pesquisa em processamento de imagens e visão por computador em todo o mundo.

"<http://www.cs.cmu.edu/~cil/txtv-groups.html>"

Computer Vision: Research Groups (text only)

Versão texto da página indicada acima, adequada para reduzir o tempo de carga da página de abertura na tela.

"<http://www.ph.tn.tudelft.nl/PRInfo/groups.html>"

Pattern Recognition Research Groups

Lista de grupos de pesquisa ativos na área, classificados por continente.

Publicações

"<http://www.lpac.ac.uk/SEL-HPC/Articles/VisionArchive.html>"

SEL-HPC Vision and Image Processing Archive

Repositório de artigos técnicos na área de processamento de imagens. Pode ser atualizado dinamicamente. Selecionado pelos autores deste livro como um dos dez *sites* mais relevantes na área.

"<http://www.nr.com/>"

Numerical Recipes in C

Referência obrigatória para programadores na área científica, este livro está agora disponível *on-line*. A versão PostScript de cada capítulo pode ser obtida gratuitamente no endereço acima.

"<http://www.cs.hmc.edu/~fleck/computer-vision-handbook/index.html>"

The Computer Vision Handbook

Ainda em construção, propõe-se a ser um grande livro-texto *on-line* sobre o assunto, com muitos *links* e referências bibliográficas.

"<http://www.khoral.com/dipcourse/dip17sep97/>"

Digital Image Processing (DIP) with Khoros 2

Pioneira e feliz iniciativa do Prof. Roberto de Alencar Lotufo da Unicamp, em parceria com o Prof. Ramiro Jordán da *University of New Mexico* (EUA), é um curso interativo completo de processamento de imagens na rede. Parte do curso exige a plataforma Khoros.

"<http://www.cogs.susx.ac.uk/users/davidy/teachvision/vision0.html>"

Sussex Computer Vision: Introduction to the HTML teach files

Curso *on-line* de visão por computador com exemplos de programas em POP-11 [Barrett et al. 1985].

"<http://www.ime.usp.br/mac/khoros/mmach.old/tutor/mmach.html>"

A Tutorial on Mathematical Morphology

Curso interativo de Morfologia Matemática desenvolvido pela Universidade de São Paulo sobre a plataforma Khoros 2.0. Possui opção para aqueles que desejarem uma visão geral do assunto de forma rápida e independente de hardware ou software.

"<http://www.cs.washington.edu/research/metip/metip.html>"

Mathematics Experiences Through Image Processing (METIP)

Home-page de inovador projeto desenvolvido pela *University of Washington* que se propõe a utilizar operações de processamento de imagens para motivar e facilitar o ensino de matemática. Permite o *download* gratuito dos títulos de software já desenvolvidos no âmbito do projeto.

"http://www.cm.cf.ac.uk/Dave/Vision_index.html"

MSc AI (and Engineering Application) Vision Systems Course Documentation

Curso abrangendo tópicos clássicos em formato eletrônico, porém pouco interativo e com pequeno número de imagens.

"<http://www.inforamp.net/~poynton/Poynton-T-I-Digital-Video.html>"

A Technical Introduction to Digital Video

Contém o índice do livro homônimo, bem como versões *on-line* dos capítulos 1 e 6.

"<http://www.eecs.wsu.edu/IPdb/title.html>"

Digital Image Processing Home Page

Curso de processamento de imagens em formato de hipertexto, ainda em construção, apresentando a teoria básica, exemplos de imagens e trechos de código-fonte em C.

Eventos

"<http://iris.usc.edu/Information/Iris-Conferences.html>"

Computer Vision Conference Listing from USC

Contém bem diagramada agenda de eventos e *links* para as páginas relacionadas a cada evento.

"<http://www.cs.cmu.edu/~cil/v-conf.html>"

Computer Vision: Conferences and Symposia

Lista de conferências e eventos na área, com *links* para as páginas de cada evento e para outras listas de eventos.

"<http://www.ph.tn.tudelft.nl/PRInfo/conferences.html>"

Pattern Recognition Related Conferences

Outra lista de conferências e eventos na área, com *links* para as páginas de cada evento.

"<http://afrodite.lira.dist.unige.it/confs/confs.html>"

Conferences Main Menu

Página de *links* para conferências e eventos, mantida pela ECVNet.

Imagens

"<http://www.ics.forth.gr/ecvnet/imageDB/index.html>"

ECVNet Image DataBases Page

Página contendo *links* para diversos repositórios de imagens na Internet.

"<http://www.cs.cmu.edu/~cil/v-images.html>"

Computer Vision: Test Images

Outra página contendo diversos *links* para diversos repositórios de imagens na Internet.

Bibliografia

[Barrett et al. 1985]

Barrett, R. et allii, *POP-11: a practical language for artificial intelligence*. Ellis Horwood Ltd., 1985.

[Dougherty 1994]

Dougherty, E.R. (ed.), *Digital Image Processing Methods*, Marcel Dekker, 1994.

[Gonzalez e Woods 1992]

Gonzalez, R.C. e Woods, R.E., *Digital Image Processing - Third Edition*, Addison-Wesley, 1992.

[Lindley 1991]

Lindley, C.A., *Practical Image Processing in C*, Wiley, 1991.

[Marr 1982]

Marr, D., *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*, W. H. Freeman and Co., 1982.

[Mascarenhas 1990]

Mascarenhas, N.D., "Introdução ao Processamento Digital de Imagens", *Anais da I Jornada EPUSP/IEEE de Computação Visual*, 1990, 387-420.

[Rimmer 1993]

Rimmer, S., *Bit-Mapped Graphics*, Windcrest Books, 1993.

[Schalkoff 1989]

Schalkoff, R.J., *Digital Image Processing and Computer Vision*, Wiley, 1989.

Bibliografia Recomendada

Cada capítulo deste livro contém ao seu final indicações bibliográficas referentes ao assunto abordado. Para uma visão mais abrangente das principais fontes de consulta na área, relacionamos a seguir alguns dos mais renomados livros e periódicos disponíveis atualmente.

Alguns dos principais periódicos científicos (*journals*) ligados às áreas de processamento de imagens e visão por computador são (em ordem alfabética):

- *Computer Vision, Graphics and Image Processing*
- *Graphical Models and Image Processing*
- *IEEE Computer Graphics and Applications*
- *IEEE Expert-Intelligent Systems and their Applications*
- *IEEE Multimedia*
- *IEEE Transactions on Image Processing*
- *IEEE Transactions on Information Theory*
- *IEEE Transactions on Medical Imaging*
- *IEEE Transactions on Multimedia*
- *IEEE Transactions on Pattern Analysis and Machine Intelligence*
- *IEEE Transactions on Signal Processing*
- *IEEE Transactions on Systems, Man and Cybernetics*
- *Image and Vision Computing*
- *Image Understanding, International Journal of Computer Vision*
- *Journal of Mathematical Imaging and Vision*
- *Machine Vision and Applications*
- *Pattern Recognition*
- *Proceedings of the IEEE.*
- *Real-Time Imaging*

Dentre os livros-texto consagrados na área, citamos e recomendamos (por ordem alfabética de autor):

- Castleman, K.R., *Digital Image Processing*, Prentice-Hall, 1995.
- Dougherty, E.R. e Giardina, C.R., *Matrix Structured Image Processing*, Prentice-Hall, 1987.
- Gonzalez, R.C. e Woods, R.E., *Digital Image Processing*, Addison-Wesley, 1992.
- Jain, A.K., *Fundamentals of Digital Image Processing*, Prentice-Hall, 1989.
- Jain, R.C., Kasturi, R., e Schunck, B.G., *Machine Vision*, McGraw-Hill, 1995.
- Lim, J.S., *Two-dimensional Signal and Image Processing*, Prentice-Hall, 1990.
- Lindley, C.A., *Practical Image Processing in C*, Wiley, 1991.
- Marr, D., *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*, W. H. Freeman and Co., 1982.
- Myler, H.R. e Weeks, A.R., *Computer Imaging Recipes in C*, Prentice Hall, 1993.
- Pavlidis, T., *Algorithms for Graphics and Image Processing*, Computer Science Press, 1982.
- Pratt, W. K., *Digital Image Processing*, Wiley Interscience, 1991.
- Russ, J. C., *The Image Processing Handbook*, CRC Press, 1995.
- Schalkoff, R.J., *Digital Image Processing and Computer Vision*, Wiley, 1989.
- Sonka, M., Hlavac, V. e Boyle, R., *Image Processing, Analysis and Machine Vision*, Chapman & Hall, 1993.

Capítulo 2

Fundamentos de Imagens Digitais

Este capítulo tem por objetivo apresentar as principais características das imagens digitais. A primeira seção caracteriza uma imagem do ponto de vista matemático e descreve o processo de aquisição de imagens e sua conversão para o formato digital. A seção 2.2 descreve as principais propriedades de uma imagem digital, padronizando a terminologia a ser utilizada no restante do livro. As principais operações lógicas e aritméticas sobre imagens são exemplificadas na seção 2.3. A seção 2.4 introduz o importante conceito de convolução com máscaras e fornece exemplos de máscaras úteis para enfatizar e/ou detetar propriedades de uma imagem. Finalmente, a seção 2.5 trata das transformações geométricas que podem ser aplicadas a imagens, fundamentando-as matematicamente e ilustrando-as com exemplos.

2.1 Aquisição e digitalização de imagens

Uma imagem monocromática pode ser descrita matematicamente por uma função $f(x,y)$ da intensidade luminosa, sendo seu valor, em qualquer ponto de coordenadas espaciais (x,y) , proporcional ao brilho (ou nível de cinza) da imagem naquele ponto. A figura 1 mostra uma imagem monocromática e a convenção utilizada neste livro para o par de eixos (x,y) ¹.

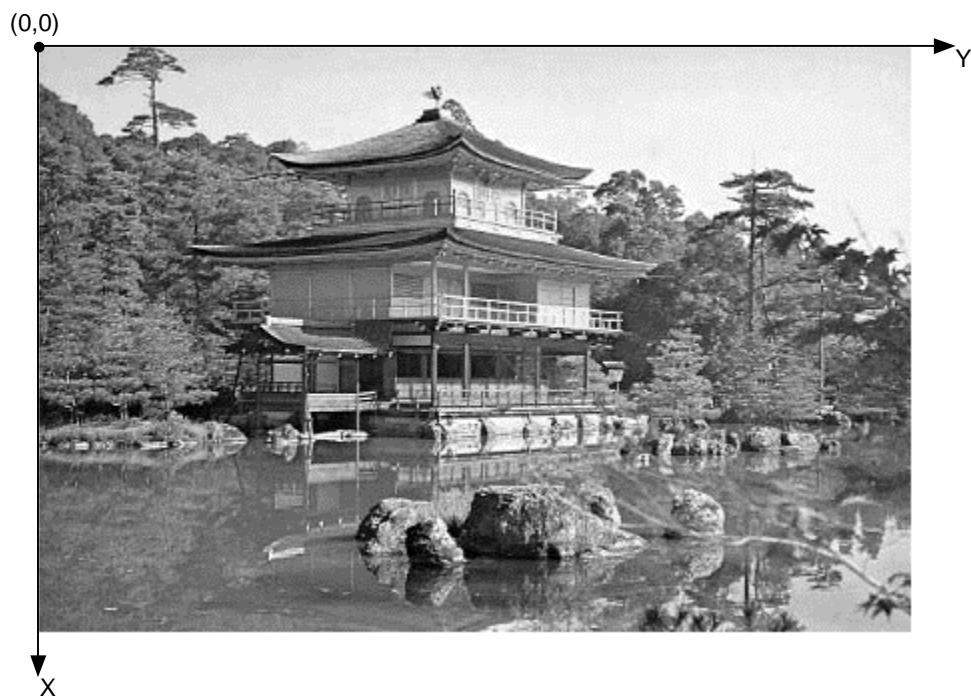


Figura 1 - Uma imagem monocromática e a convenção utilizada para o par de eixos (x,y) .

A função $f(x,y)$ representa o produto da interação entre a iluminância $i(x,y)$ – que exprime a quantidade de luz que incide sobre o objeto – e as propriedades de refletância ou de transmitância próprias do objeto, que podem ser representadas pela função $r(x,y)$, cujo valor

¹ Como o leitor deve ter notado, a posição e a direção dos eixos x e y são diferentes das utilizadas na Geometria Analítica. Em Processamento de Imagens, a notação (x,y) pode ser entendida como (*linha, coluna*). Convém observar que esta notação não está padronizada na literatura técnica da área.

exprime a fração de luz incidente que o objeto vai transmitir ou refletir ao ponto (x,y) . Estes conceitos estão ilustrados na figura 2. Matematicamente:

$$f(x, y) = i(x, y) \cdot r(x, y) \quad (2.1)$$

com:

$$0 < i(x, y) < \infty \quad \text{e}$$

$$0 < r(x, y) < 1$$

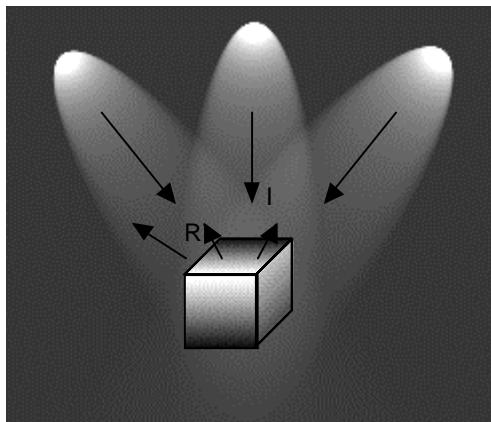


Figura 2 - Os componentes iluminância (I) e refletância (R) de uma imagem.

As tabelas 1 e 2 apresentam valores típicos de iluminância e refletância.

Tabela 1 - Exemplos de valores para $i(x,y)$ [em lux ou lúmen/m²]

$i(x,y)$	
900	dia ensolarado
100	dia nublado
10	iluminação média de escritório
0,001	noite clara de lua cheia

Tabela 2 - Exemplos de valores para $r(x,y)$

$r(x,y)$	
0,93	neve
0,80	parede branco-fosca
0,65	aço inoxidável
0,01	veludo preto

Ao longo deste livro, a intensidade de uma imagem monocromática f nas coordenadas (x,y) será denominada nível de cinza (ou tom de cinza) (L) da imagem naquele ponto. Este valor estará no intervalo:

$$L_{\min} \leq L \leq L_{\max}$$

sendo L_{\min} e L_{\max} valores positivos e finitos.

O intervalo $[L_{\min}, L_{\max}]$ é denominado escala de cinza da imagem. É comum deslocar este intervalo numericamente para o intervalo dos inteiros $[0, W]$, onde $L = 0$ significa pixel preto e $L = W-1$ representa pixel branco. Normalmente, W é uma potência inteira positiva de 2.

No caso de uma imagem que possui informações em intervalos ou bandas distintas de freqüência, é necessário uma função $f(x,y)$ para cada banda. É o caso de imagens coloridas padrão RGB, que são formadas pela informação de cores primárias aditivas, como o vermelho (*R - Red*), verde (*G - Green*) e azul (*B - Blue*). A seção 4.6 apresenta informações adicionais sobre imagens coloridas.

As técnicas de processamento de imagens descritas neste livro trabalham fundamentalmente com imagens monocromáticas, bidimensionais e estáticas. Para que uma imagem seja processada por alguma destas técnicas, é fundamental representar sua informação num formato adequado ao tratamento computacional, por exemplo, uma matriz de números inteiros não-negativos, cujos valores referenciam o brilho médio amostrado no ponto correspondente da cena.

Para converter uma cena real em uma imagem digitalizada, duas etapas são imprescindíveis: a aquisição da imagem e sua digitalização.

2.1.1 Aquisição

Chamaremos de aquisição de uma imagem o processo de conversão de uma cena real tridimensional em uma imagem analógica, ou seja, delimitaremos esta etapa ao processo de transdução optoeletrônica.

O primeiro passo na conversão de uma cena real tridimensional em uma imagem eletrônica é a redução de dimensionalidade. Assumiremos que uma câmera fotográfica, câmera de vídeo ou outro dispositivo converterá a cena 3-D em uma representação 2-D adequada, sem nos preocuparmos com as questões envolvidas nesta etapa.

O dispositivo de aquisição de imagens mais utilizado atualmente é a câmera CCD (*Charge Coupled Device*). Ela consiste de uma matriz de células semicondutoras fotossensíveis, que atuam como capacitores, armazenando carga elétrica proporcional à energia luminosa incidente. O sinal elétrico produzido é condicionado por circuitos eletrônicos especializados, produzindo à saída um Sinal Composto de Vídeo (SCV) analógico e monocromático.

Para a aquisição de imagens coloridas utilizando CCDs é necessário um conjunto de prismas e filtros de cor encarregados de decompor a imagem colorida em suas componentes R, G e B, cada qual capturada por um CCD independente. Os sinais elétricos correspondentes a cada componente são combinados posteriormente conforme o padrão de cor utilizado (NTSC (*National Television Standards Committee*) ou PAL (*Phase Alternating Line*), por exemplo). Uma câmera CCD monocromática simples consiste basicamente de um conjunto de lentes que focalizarão a imagem sobre a área fotossensível do CCD, o sensor CCD e seus circuitos complementares. A figura 3 mostra uma visão simplificada da aquisição de imagens com câmera CCD. O capítulo 7 traz maiores detalhes sobre sensores de imagem.

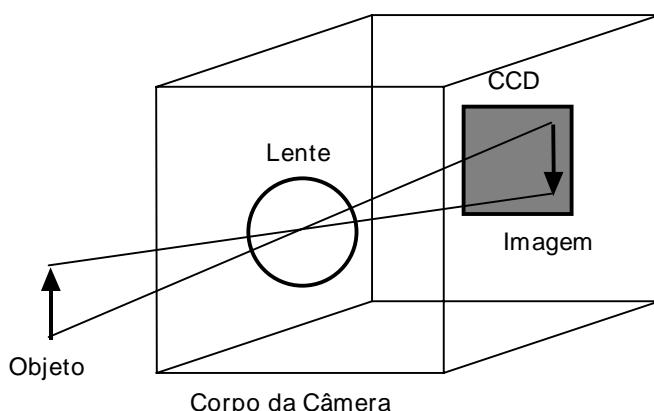


Figura 3 - Visão esquemática de uma câmera CCD.

2.1.2 Digitalização

O sinal analógico de vídeo obtido à saída do dispositivo de aquisição deve ser submetido a uma discretização espacial e em amplitude para tomar o formato desejável ao processamento computacional.

Chamaremos de amostragem o processo de discretização espacial e daremos o nome de quantização ao processo de discretização em amplitude.

Basicamente, a amostragem converte a imagem analógica em uma matriz de M por N pontos, cada qual denominado pixel (ou elemento de imagem):

$$f(x, y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0, N-1) \\ f(1,0) & f(1,1) & \dots & f(1, N-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1, N-1) \end{bmatrix} \quad (2.2)$$

Maiores valores de M e N implicam em uma imagem de maior resolução.

Por seu lado, a quantização faz com que cada um destes pixels assuma um valor inteiro, na faixa de 0 a $2^n - 1$. Quanto maior o valor de n , maior o número de níveis de cinza presentes na imagem digitalizada.

Do ponto de vista eletrônico, a digitalização consiste em uma conversão analógico-digital na qual o número de amostras do sinal contínuo por unidade de tempo indica a taxa de amostragem e o número de bits do conversor A/D utilizado determina o número de tons de cinza resultantes na imagem digitalizada.

Sob uma abordagem matemática formal, o processo de amostragem pode ser visto como uma divisão do plano xy em uma grade, com as coordenadas do centro de cada grade sendo uma dupla de elementos do produto cartesiano $Z \times Z$ (também escrito Z^2), o qual é o conjunto de todos os pares ordenados dos elementos (a, b) com a e b sendo números pertencentes a Z (conjunto dos inteiros). Portanto $f(x, y)$ é uma imagem digital se (x, y) forem números inteiros de $Z \times Z$ e f uma função que atribui um valor de nível de cinza (isto é, um número real do conjunto de números reais R) para cada par distinto de coordenadas, ou seja, f é o processo de quantização descrito anteriormente. Se os níveis de cinza resultantes forem também números inteiros (como geralmente é o caso), Z substitui R e uma imagem digital então se torna uma função bidimensional cujas coordenadas e valores de amplitude são números inteiros.

Na especificação do processo de digitalização deve-se decidir que valores de N , M e n são adequados, do ponto de vista de qualidade da imagem e da quantidade de bytes necessários para armazená-la. A tabela 3 fornece uma idéia estimativa do número de bytes necessários para armazenar uma imagem de $N \times N$ pixels com 2^n tons de cinza, calculados como: $N \times N \times n / 8$. Assume-se que um pixel estará inteiramente contido em um byte, mesmo que isto signifique que alguns bits de cada byte permaneçam vazios. Por exemplo, para $n = 5$, assume-se que cada pixel ocupa um byte, restando 3 bits sem utilização em cada byte.

Do ponto de vista qualitativo, poder-se-ia perguntar: quantos pontos e níveis de cinza serão necessários para que a versão digitalizada de uma imagem apresente qualidade comparável à imagem original? Parece evidente que quanto maiores os valores de M , N e n , melhor a imagem digital resultante. Mas sabendo que elevados valores de M , N e n implicarão em maiores custos de digitalização e armazenamento, deve existir uma forma de definir valores adequados à qualidade desejada. Convém observar ainda que 'qualidade de imagem' é um conceito altamente subjetivo, que também depende fortemente dos requisitos da aplicação dada.

Para que o leitor possa tirar suas próprias conclusões sobre a dependência entre qualidade subjetiva e resolução espacial, a figura 4(a) mostra uma imagem de 256 x 256 pixels, com 256 níveis de cinza. Mantendo constante o número de tons de cinza, as figuras 4 (b)-(d) mostram os resultados da redução espacial de $N = 256$ para $N = 128$, 64 e 32, respectivamente.

Tabela 3 - Número de bytes necessários para armazenar uma imagem digital $N \times N$ com 2^n níveis de cinza

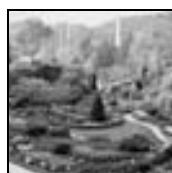
n	1	2	3	4	5	6	7	8
N								
32	128	256	512	512	1.024	1.024	1.024	1.024
64	512	1.024	2.048	2.048	4.096	4.096	4.096	4.096
128	2.048	4.096	8.192	8.192	16.384	16.384	16.384	16.384
256	8.192	16.384	32.768	32.768	65.536	65.536	65.536	65.536
512	32.768	65.536	131.072	131.072	262.144	262.144	262.144	262.144
1.024	131.072	262.144	393.216	524.288	655.360	786.432	917.504	1.048.576



(a)



(b)



(c)



(d)

Figura 4 - Efeito da resolução espacial na qualidade da imagem.

A figura 5 ilustra os efeitos da redução do número de níveis de cinza sobre a qualidade da imagem. Na figura 5(a) tem-se uma imagem de 442 x 299 pixels com 256 tons de cinza ($n = 8$). As figuras 5(b)-(h) foram obtidas reduzindo-se o número de bits de $n = 7$ até $n = 1$, enquanto a resolução espacial foi mantida constante em 442 x 299 pixels. A partir da imagem com 32 tons de cinza é perceptível o surgimento de uma imperfeição na imagem, conhecida como 'falso contorno' (*false contouring*).

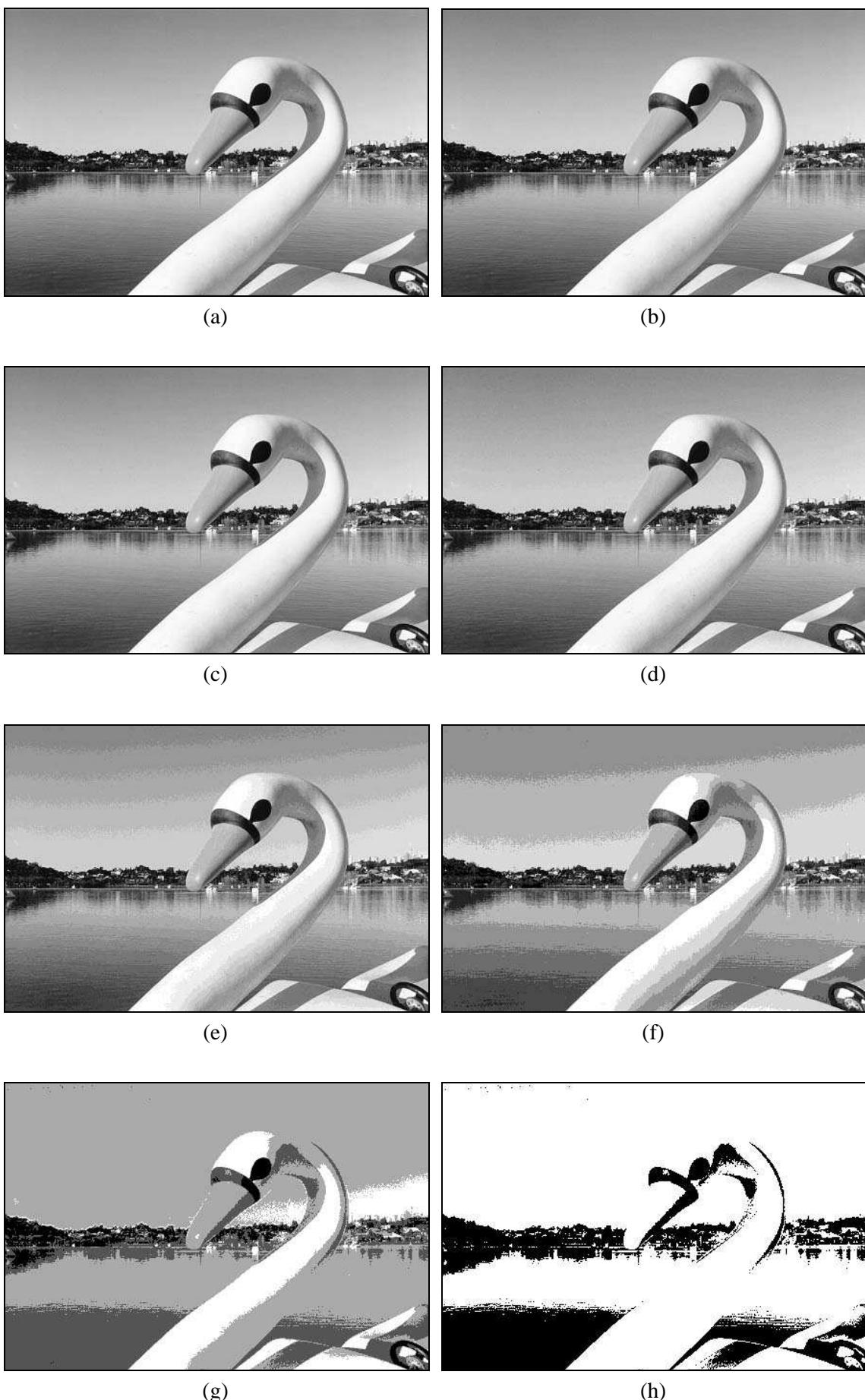


Figura 5 - Efeito do número de níveis de cinza na qualidade de uma imagem 442 x 299 com 256, 128, 64, 32, 16, 8, 4 e 2 níveis de cinza, respectivamente.

MARQUES FILHO, Ogé; VIEIRA NETO, Hugo. *Processamento Digital de Imagens*, Rio de Janeiro: Brasport, 1999. ISBN 8574520098.

Para obter uma imagem digital de qualidade semelhante a de uma imagem de televisão P&B, são necessários 512 x 512 pixels e 128 níveis de cinza. Em geral, 64 níveis de cinza são considerados suficientes para o olho humano. Apesar disto, a maioria dos sistemas de visão artificial utiliza imagens com 256 níveis de cinza.

Os processos de amostragem e quantização podem ser aprimorados usando técnicas adaptativas. Sob o aspecto da amostragem, a idéia básica é utilizar maior número de pontos em regiões de grande detalhe, em detrimento das regiões homogêneas de grandes dimensões, que poderiam ser amostradas com menor número de pixels. Sob o ângulo da quantização, uma vez que o olho humano não é capaz de perceber sutis diferenças de tons de cinza nas imediações de variações abruptas de intensidade, o objetivo seria utilizar poucos níveis de cinza nestas regiões. O principal obstáculo para a implementação destas técnicas é a necessidade de identificação prévia (ainda que aproximada) das regiões presentes na imagem e das fronteiras entre elas. No caso da quantização, entretanto, outra técnica adaptativa pode ser utilizada. Efetuando um levantamento da freqüência de ocorrência de todos os níveis de cinza permitidos, pode-se diminuir os degraus de quantização nas regiões da escala de cinza com maior concentração de ocorrência de pixels, aumentando-os nas demais regiões.

Leitura complementar

Para uma introdução à questão da conversão de uma cena tridimensional em uma imagem bidimensional, incluindo transformações de perspectiva, aspectos de calibração de câmeras e visão estéreo, sugerimos [Faugeras 1993], o capítulo 2 de [Schalkoff 1989] e a seção 2.5 de [Gonzalez e Woods 1992].

Para maiores detalhes sobre o funcionamento de câmeras CCD e fundamentos de sinais analógicos de vídeo recomendamos [Nince 1991].

Os capítulos 1 a 5 de [Lindley 1991] trazem uma descrição pormenorizada de um projeto de digitalizador de imagens (hardware e software).

Para uma análise dos efeitos produzidos na qualidade da imagem pela variação simultânea da resolução espacial e do número de níveis de cinza, ver o trabalho de Huang [Huang 1965], resumido na seção 2.3 de [Gonzalez e Woods 1992].

O capítulo 2 de [Pavlidis 1982] traz informações adicionais sobre os aspectos de amostragem e quantização.

Aos interessados em um aprofundamento matemático dos aspectos abordados nesta seção, recomendamos os capítulos 1, 4, 5 e 6 de [Pratt 1991].

2.2 Propriedades de uma imagem digital

Nesta seção consideraremos as principais relações entre pixels em uma imagem digital. Uma imagem digital é uma imagem $f(x,y)$ discretizada tanto espacialmente quanto em amplitude. Portanto, uma imagem digital pode ser vista como uma matriz cujas linhas e colunas identificam um ponto na imagem, cujo valor corresponde ao nível de cinza da imagem naquele ponto. Para efeito de notação, uma imagem digital será indicada por $f(x,y)$. Quando nos referirmos a um pixel em particular, utilizaremos letras minúsculas, tais como p e q . Um subconjunto de pixels de $f(x,y)$ será indicado por S .

2.2.1 Vizinhança

Um pixel p , de coordenadas (x,y) , tem 4 vizinhos horizontais e verticais, cujas coordenadas são $(x+1, y)$, $(x-1, y)$, $(x, y+1)$ e $(x, y-1)$. Estes pixels formam a chamada "4-vizinhança" de p , que será designada $N_4(p)$.

Os quatro vizinhos diagonais de p são os pixels de coordenadas $(x-1, y-1)$, $(x-1, y+1)$, $(x+1, y-1)$ e $(x+1, y+1)$, que constituem o conjunto $N_d(p)$.

A "8-vizinhança" de p é definida como:

$$N_8(p) = N_4(p) \cup N_d(p) \quad (2.3)$$

Os vários tipos de vizinhança estão ilustrados na figura 6.

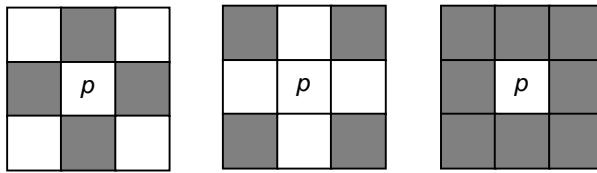


Figura 6 - Conceitos de 4-vizinhança, vizinhança diagonal e 8-vizinhança.

2.2.2 Conectividade

A conectividade entre pixels é um importante conceito usado para estabelecer limites de objetos e componentes de regiões em uma imagem. Para se estabelecer se dois pixels estão conectados, é necessário determinar se eles são adjacentes segundo algum critério e se seus níveis de cinza satisfazem a um determinado critério de similaridade. Por exemplo, em uma imagem binária, onde os pixels podem assumir os valores 0 e 1, dois pixels podem ser 4-vizinhos, mas somente serão considerados 4-conectados se possuírem o mesmo valor.

Seja V o conjunto de valores de tons de cinza utilizados para se definir a conectividade. Por exemplo, numa imagem binária, $V = \{1\}$ para a conexão de pixels com valor 1. Numa imagem de múltiplos tons de cinza, para a conexão de pixels com valores de intensidade na faixa de 32 a 64, $V = \{32, 33, \dots, 63, 64\}$. Conhecendo o conceito de vizinhança e dado o conjunto V , podemos definir os seguintes critérios de conectividade:

1. "4-conectividade": dois pixels p e q com valores de tom de cinza contidos em V , são "4-conectados" se $q \in N_4(p)$.
2. "8-conectividade": dois pixels p e q com valores de tom de cinza contidos em V , são "8-conectados" se $q \in N_8(p)$.
3. "m-conectividade (conectividade mista)": dois pixels p e q com valores de tom de cinza contidos em V , são "m-conectados" se:

- (i) $q \in N_4(p)$ ou
- (ii) $q \in N_d(p) \subseteq N_4(p) \cap N_4(q) = \emptyset$.

A conectividade mista é uma modificação da 8-conectividade e é introduzida para eliminar os múltiplos caminhos que geralmente surgem quando a 8-conectividade é usada. Por exemplo, seja o trecho de imagem da figura 7(a). Para $V = \{1\}$ os caminhos entre 8 vizinhos do pixel do centro são indicados por linhas contínuas na figura 7(b), onde se pode observar a existência de caminhos redundantes entre os pixels do centro e do canto superior esquerdo da figura. Esta redundância é resolvida utilizando-se a m-conectividade, que remove a conexão diagonal redundante, como mostra a figura 7(c).

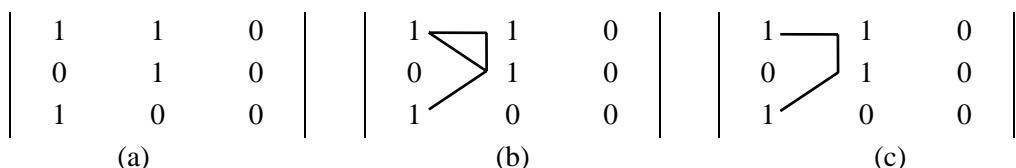


Figura 7 - (a) Segmento de imagem binária, (b) 8-vizinhos do pixel central, (c) m-vizinhos do pixel central.

2.2.3 Adjacência

Um pixel p é adjacente a um pixel q se eles forem conectados. Há tantos critérios de adjacência quantos são os critérios de conectividade. Dois subconjuntos de imagens, S_1 e S_2 , são adjacentes se algum pixel em S_1 é adjacente a algum pixel em S_2 .

2.2.4 Caminho

Um caminho (*path*) de um pixel p de coordenadas (x,y) a um pixel q de coordenadas (s,t) é uma seqüência de pixels distintos de coordenadas: $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$,

onde:

$$(x_0, y_0) = (x, y)$$

$$(x_n, y_n) = (s, t)$$

(x_i, y_i) é adjacente a (x_{i-1}, y_{i-1})

$$1 \leq i \leq n$$

n é denominado o comprimento do caminho.

2.2.5 Medições de distância

Dados os pixels p , q e z , de coordenadas (x,y) , (s,t) e (u,v) , respectivamente, define-se a função distância D , cujas propriedades são:

- (i) $D(p,q) \geq 0$ ($D(p,q) = 0$ se e somente se $p = q$)
- (ii) $D(p,q) = D(q,p)$
- (iii) $D(p,z) \leq D(p,q) + D(q,z)$

Distância Euclidiana

$$D_e(p,q) = \sqrt{(x-s)^2 + (y-t)^2} \quad (2.4)$$

Para esta medida de distância, os pixels com distância euclidiana em relação a (x,y) menor ou igual a algum valor r , são os pontos contidos em um círculo de raio r centrado em (x,y) .

Distância D_4 (*city-block*)

$$D_4(p,q) = |x - s| + |y - t| \quad (2.5)$$

onde $| . |$ denota módulo (ou valor absoluto).

Neste caso, os pixels tendo uma distância D_4 em relação a (x,y) menor ou igual a algum valor r formam um losango centrado em (x,y) . Os pixels com $D_4 = 1$ são os 4-vizinhos de (x,y) .

Distância D_8 (tabuleiro de xadrez)

$$D_8(p,q) = \max(|x - s|, |y - t|) \quad (2.6)$$

onde \max é um operador que devolve o maior valor dentre um conjunto de valores entre parênteses.

Neste caso os pixels com distância D_8 em relação a (x,y) menor ou igual a algum valor r formam um quadrado centrado em (x,y) . Os pixels com $D_8 = 1$ são os 8-vizinhos de (x,y) .

O conceito de distância pode estar relacionado ao conceito de conectividade. A distância D_m expressa a distância entre dois pontos m-conectados.

Exercício resolvido

Seja o trecho de imagem binária a seguir:

	p_3	p_4
p_1	p_2	
p		

Supondo que $V = \{1\}$, $p = p_2 = p_4 = 1$ e que p_1 e p_3 podem apresentar valores 0 ou 1, calcular a distância D_m entre p e p_4 para as seguintes situações:

a) Se $p_1 = p_3 = 0$.

Solução: a distância D_m vale 2, pois o caminho m entre p e p_4 é obtido unindo-se os pixels p , p_2 e p_4 .

b) Se p_1 ou p_3 valem 1.

Solução: a distância D_m vale 3, pois o caminho m entre p e p_4 será p, p_1, p_2, p_4 ou p, p_2, p_3, p_4 .

c) Se p_1 e p_3 valem 1.

Solução: a distância D_m vale 4, pois o caminho m entre p e p_4 será p, p_1, p_2, p_3, p_4 . |

Leitura complementar

As subseções 2.4.3 e 2.4.4 de [Gonzalez e Woods 1992] apresentam um método de atribuição de rótulos a aglomerados de pixels conectados de uma imagem e relacionam este procedimento aos conceitos matemáticos de relação binária, relação de equivalência e fecho transitivo.

2.3 Operações lógicas e aritméticas

Sabemos que após uma imagem ter sido adquirida e digitalizada, ela pode ser vista como uma matriz de inteiros e portanto pode ser manipulada numericamente utilizando operações lógicas e/ou aritméticas. Estas operações podem ser efetuadas pixel a pixel ou orientadas a vizinhança. No primeiro caso, elas podem ser descritas pela seguinte notação:

$$X \text{ opn } Y = Z$$

onde X e Y podem ser imagens (matrizes) ou escalares, Z é obrigatoriamente uma matriz e opn é um operador aritmético ($+, -, \times$ e $/$) ou lógico (AND, OR, XOR) binário².

Sejam duas imagens X e Y de igual tamanho. Estas imagens podem ser processadas pixel a pixel utilizando um operador aritmético ou lógico, produzindo uma terceira imagem Z , cujos pixels correspondem ao resultado de $X \text{ opn } Y$ para cada elemento de X e Y , conforme ilustra esquematicamente a figura 8.

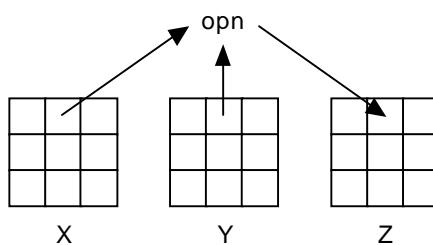


Figura 8 - Operações lógicas / aritméticas pixel a pixel.

² Aqui, o termo binário deve ser entendido como 'que requer dois operandos' e não tem qualquer relação com imagem binária.

2.3.1 Operações aritméticas pixel a pixel

Ao executarmos operações aritméticas sobre imagens, devemos tomar especial cuidado com os problemas de *underflow* ou *overflow* do resultado. A adição de duas imagens de 256 tons de cinza, por exemplo, pode resultar em um número maior que 255 para alguns pixels, ao mesmo tempo que a subtração de duas imagens pode resultar em valores negativos para alguns elementos. Para contornar estes problemas, existem basicamente duas alternativas: (1) manter os resultados intermediários em uma matriz na qual o espaço em memória alocado para cada pixel permita a representação de números negativos e/ou maiores que 255 e em seguida proceder a uma normalização destes valores intermediários; (2) truncar os valores maiores que o máximo valor permitido, bem como os valores negativos, igualando-os a 255 e 0, respectivamente. A decisão depende do objetivo que se tem em mente ao executar determinada operação. Efetivamente, a segunda alternativa é mais simples que a primeira.

Exercício resolvido

Dadas as matrizes X e Y a seguir, correspondentes a trechos 3 x 3 de imagens de 256 tons de cinza, adicioná-las e informar: (a) o resultado intermediário (sem considerações de *underflow* e *overflow*), (b) o resultado final utilizando normalização, (c) o resultado final utilizando truncamento.

$$X = \begin{bmatrix} 200 & 100 & 100 \\ 0 & 10 & 50 \\ 50 & 250 & 120 \end{bmatrix} \quad Y = \begin{bmatrix} 100 & 220 & 230 \\ 45 & 95 & 120 \\ 205 & 100 & 0 \end{bmatrix}$$

Solução:

$$(a) \begin{bmatrix} 300 & 320 & 330 \\ 45 & 105 & 170 \\ 255 & 350 & 120 \end{bmatrix}$$

(b) Fazendo com que a escala [45, 350] seja adequada ao intervalo [0, 255], utilizando-se a relação

$$g = \frac{255}{f_{max} - f_{min}}(f - f_{min}), \quad (2.7)$$

obtém-se:

$$\begin{bmatrix} 213 & 230 & 238 \\ 0 & 50 & 105 \\ 175 & 255 & 63 \end{bmatrix}$$

(c) Truncando os valores maiores que 255, obtém-se:

$$\begin{bmatrix} 255 & 255 & 255 \\ 45 & 105 & 170 \\ 255 & 255 & 120 \end{bmatrix}$$

As principais aplicações das operações aritméticas sobre imagens estão resumidas na tabela 4. Assim como Y foi implicitamente considerado até aqui como sendo uma matriz, ele também pode ser um escalar. A segunda coluna da tabela 4 avalia os efeitos qualitativos das operações aritméticas sobre imagens, abordando ambas as possibilidades. As figuras 9 a 12 mostram exemplos de cada operação aritmética.

Tabela 4 - Efeitos e aplicações das operações aritméticas sobre imagens

Operação	Efeito sobre a imagem	Aplicações
Adição	Z é o resultado da soma dos valores de intensidade de X e Y . Se Y for um escalar positivo, Z será uma versão mais clara de X ; o acréscimo de intensidade será o próprio valor de Y .	<ul style="list-style-type: none"> Normalização de brilho³ de imagens Remoção de ruídos (ver técnica da filtragem pela média de múltiplas imagens na subseção 4.2.4)
Subtração	Z é o resultado da diferença dos valores de intensidade de X e Y . Se Y for um escalar positivo, Z será uma versão mais escura de X ; o decréscimo de intensidade será o próprio valor de Y .	<ul style="list-style-type: none"> Deteção de diferenças entre duas imagens (eventualmente adquiridas de forma consecutiva) da mesma cena
Multiplicação	Z é o produto dos valores de intensidade de X e Y . Se Y for um escalar positivo, os valores de intensidade de Z serão diretamente proporcionais a X por um fator Y .	<ul style="list-style-type: none"> Calibração de brilho⁴
Divisão	Z é o razão dos valores de intensidade de X pelos valores correspondentes em Y . Se Y for um escalar positivo, os valores de intensidade de Z serão inversamente proporcionais a X por um fator Y .	<ul style="list-style-type: none"> Normalização de brilho

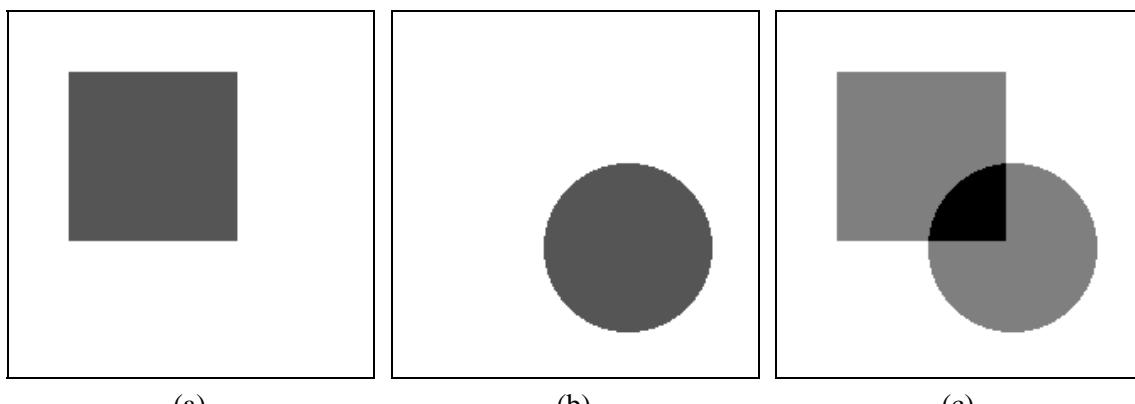


Figura 9 - Exemplo de adição de imagens monocromáticas: (a) X , (b) Y , (c) $X + Y$ (normalizado).

³ O processo de normalização de brilho consiste em adequar a faixa total de níveis de cinza a um intervalo pré-definido, de forma semelhante ao efetuado na parte (b) do Exercício Resolvido desta seção.

⁴ A calibração de brilho é um processo semelhante à normalização de brilho, mas que pode estar relacionado à adequação a diferentes valores de iluminância sobre uma mesma cena, por exemplo.

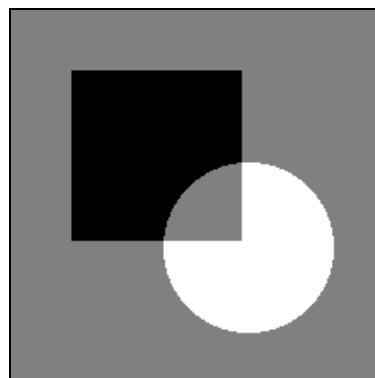


Figura 10 - Exemplo de subtração das imagens monocromáticas das figuras 9(a) e 9(b): $X - Y$ (normalizado).

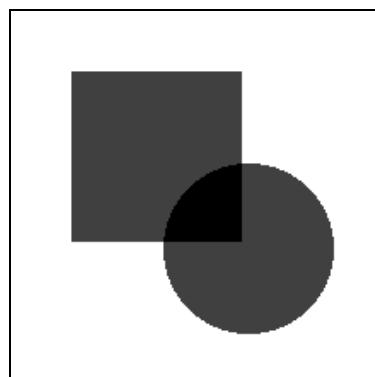


Figura 11 - Exemplo de multiplicação das imagens monocromáticas das figuras 9(a) e 9(b): XY (normalizado).

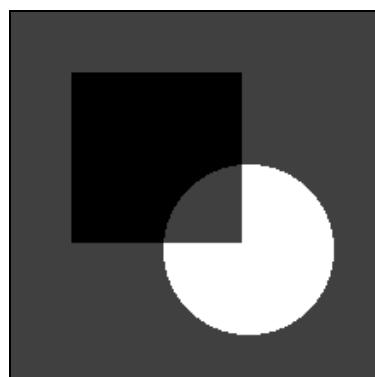
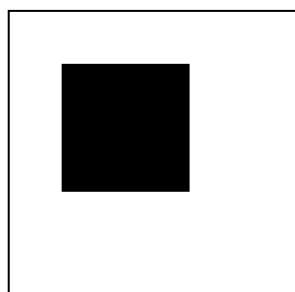


Figura 12 - Exemplo de divisão de imagens monocromáticas das figuras 9(a) e 9(b): X / Y (normalizado).

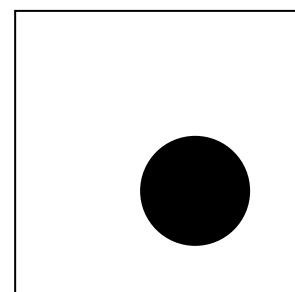
2.3.2 Operações lógicas pixel a pixel

Todas as operações lógicas (ou booleanas) conhecidas podem ser aplicadas entre imagens, inclusive a operação de complemento (NOT), que é uma operação unária (requer apenas um operando). Operações lógicas podem ser efetuadas em imagens com qualquer número de níveis de cinza mas são melhor compreendidas quando vistas em imagens binárias, como ilustra a figura 13. As figuras 14 a 17 ilustram as operações AND, OR, XOR e NOT aplicadas a imagens com múltiplos tons de cinza.



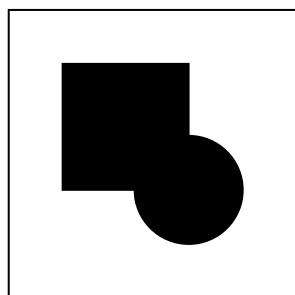
X

(a)



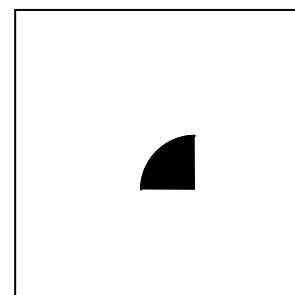
Y

(b)



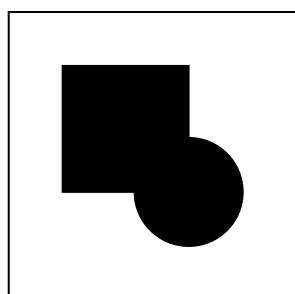
X and Y

(c)



X or Y

(d)



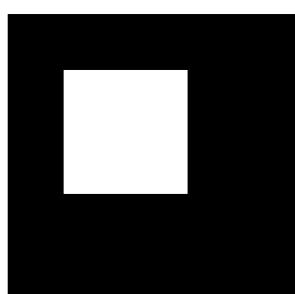
X xor Y

(e)



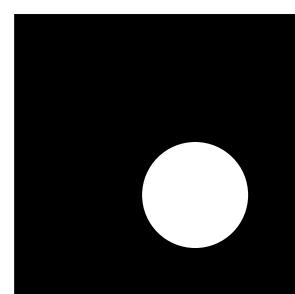
(not X) and Y

(f)



not X

(g)



not Y

(h)

Figura 13 - Exemplos de operações lógicas em imagens binárias.

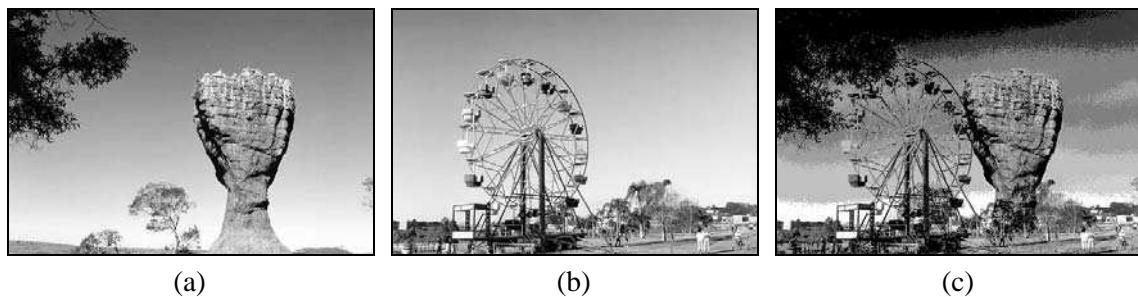


Figura 14 - Exemplo de operação AND entre imagens monocromáticas: (a) X , (b) Y , (c) $X \wedge Y$.

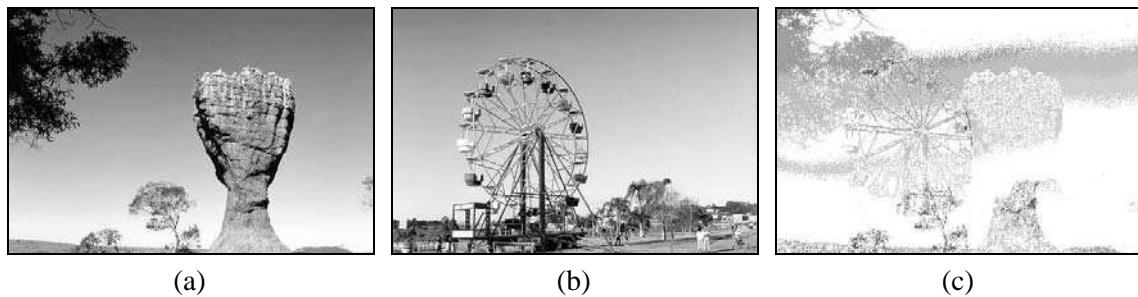


Figura 15 - Exemplo de operação OR entre imagens monocromáticas: (a) X , (b) Y , (c) $X \vee Y$.

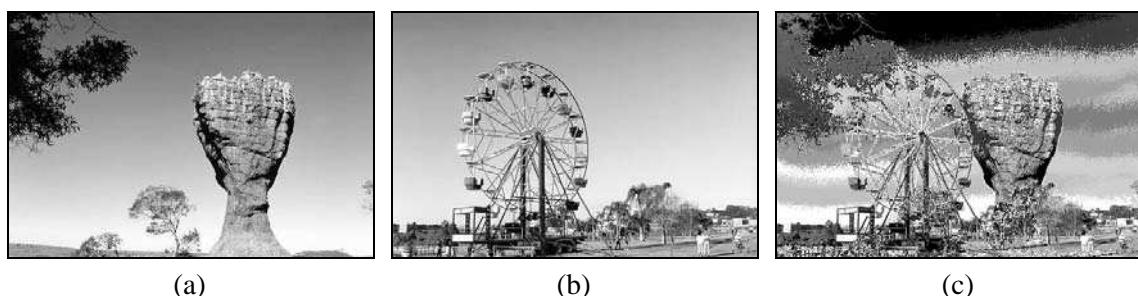


Figura 16 - Exemplo de operação XOR entre imagens monocromáticas: (a) X , (b) Y , (c) $X \oplus Y$.



Figura 17 - Exemplo de operação NOT sobre imagem monocromática: (a) X , (b) $\text{NOT } X$.

2.3.3 Operações orientadas a vizinhança

As operações lógicas e aritméticas orientadas a vizinhança utilizam o conceito de convolução com máscaras (ou janelas ou *templates*), que será introduzido a seguir e detalhado na próxima seção.

Seja uma sub-área de uma imagem:

Z_1	Z_2	Z_3
Z_4	Z_5	Z_6
Z_7	Z_8	Z_9

onde: Z_1, \dots, Z_9 são os valores de tons de cinza de cada pixel.

Seja uma máscara 3×3 de coeficientes genéricos W_1, \dots, W_9 :

W_1	W_2	W_3
W_4	W_5	W_6
W_7	W_8	W_9

A máscara acima percorrerá a imagem, desde o seu canto superior esquerdo até seu canto inferior direito. A cada posição relativa da máscara sobre a imagem, o pixel central da subimagem em questão será substituído, em uma matriz denominada 'imagem-destino', por um valor:

$$Z = \sum_{i=1}^9 W_i \cdot Z_i , \quad (2.8)$$

As operações de convolução com máscaras são amplamente utilizadas no processamento de imagens. Uma seleção apropriada dos coeficientes W_1, \dots, W_9 torna possível uma grande variedade de operações úteis, tais como redução de ruído, afinamento e deteção de características da imagem. Deve-se observar, entretanto, que a operação de convolução com máscaras exige grande esforço computacional. Por exemplo, a aplicação de uma máscara 3×3 sobre uma imagem 512×512 requer nove multiplicações e oito adições para cada localização de pixel, num total de 2.359.296 multiplicações e 2.097.152 adições. Por esta razão, aliada à relativa simplicidade de implementação de multiplicadores, somadores e registradores de deslocamento (*shift registers*), a literatura registra diversas implementações de convolução com máscaras em hardware.

Leitura complementar

O capítulo 11 de [Lindley 1991] apresenta, além das operações abordadas nesta seção, outras funções interessantes que operam pixel a pixel e podem ser aplicadas a imagens monocromáticas.

O capítulo 1 de [Dougherty e Giardina 1987] apresenta as operações básicas sobre imagens monocromáticas sob um enfoque exclusivamente matricial.

2.4 Operações de convolução com máscaras

Conforme antecipamos na seção anterior, inúmeras operações úteis em processamento de imagens são efetuadas a partir de um mesmo conceito básico, o de convolução com máscaras. Nesta seção abordaremos em mais detalhes o funcionamento das operações de convolução e apresentaremos alguns exemplos típicos de máscaras e os resultados que elas produzem quando aplicadas a imagens monocromáticas.

A operação de convolução unidimensional entre dois vetores A e B, denotada $A * B$, pode ser entendida como um conjunto de somas de produtos entre os valores de A e B, sendo que inicialmente o vetor B é espelhado e após cada soma de produtos é deslocado espacialmente de uma posição. Para ilustrar este conceito, mostraremos a seguir, passo a passo, a convolução do vetor $A = \{0, 1, 2, 3, 2, 1, 0\}$ com o vetor $B = \{1, 3, -1\}$.

1. Inicialmente, o vetor B é espelhado e alinhado com o primeiro valor de A. O resultado da convolução é $(0 \times (-1)) + (0 \times 3) + (1 \times 1) = 1$ (valores em branco assumidos como zero) e é colocado em A*B na posição correspondente ao centro do conjunto B.

A		0	1	2	3	2	1	0	
B	-1	3	1						
A*B		1							

2. O conjunto B é deslocado de uma posição. O resultado da convolução A*B é $(0 \times (-1)) + (1 \times 3) + (2 \times 1) = 5$.

A		0	1	2	3	2	1	0	
B		-1	3	1					
A*B		1	5						

3. O conjunto B é deslocado de uma posição. O resultado da convolução A*B é $(1 \times (-1)) + (2 \times 3) + (3 \times 1) = 8$.

A		0	1	2	3	2	1	0	
B			-1	3	1				
A*B		1	5	8					

4. O conjunto B é deslocado de uma posição. O resultado da convolução A*B é $(2 \times (-1)) + (3 \times 3) + (2 \times 1) = 9$.

A		0	1	2	3	2	1	0	
B				-1	3	1			
A*B		1	5	8	9				

5. O conjunto B é deslocado de uma posição. O resultado da convolução A*B é $(3 \times (-1)) + (2 \times 3) + (1 \times 1) = 4$.

A		0	1	2	3	2	1	0	
B					-1	3	1		
A*B		1	5	8	9	4			

6. O conjunto B é deslocado de uma posição. O resultado da convolução A*B é $(2 \times (-1)) + (1 \times 3) + (0 \times 1) = 1$.

A		0	1	2	3	2	1	0	
B					-1	3	1		
A*B		1	5	8	9	4	1		

7. O conjunto B é deslocado de uma posição. O resultado da convolução A*B é $(1 \times (-1)) + (0 \times 3) + (0 \times 1) = -1$. (valores em branco assumidos como zero)

A		0	1	2	3	2	1	0	
B						-1	3	1	
A*B		1	5	8	9	4	1	-1	

O conjunto $\{1, 5, 8, 9, 4, 1, -1\}$ é o resultado final da operação de convolução.

Este raciocínio pode ser expandido para o caso bidimensional, onde a imagem a ser processada é uma matriz bidimensional relativamente grande e corresponde ao conjunto A de

nosso exemplo anterior, enquanto uma matriz de pequenas dimensões (também chamada máscara ou janela) corresponde ao conjunto B. A máscara, após ter sido espelhada tanto na horizontal quanto na vertical, percorrerá todos os pontos da imagem deslocando-se ao longo de cada linha e entre as várias linhas, da direita para a esquerda, de cima para baixo, até ter processado o último elemento da matriz imagem. O resultado será armazenado em uma matriz de mesmas dimensões que a imagem original.

Seja a matriz A (imagem) dada por:

$$\begin{bmatrix} 5 & 8 & 3 & 4 & 6 & 2 & 3 & 7 \\ 3 & 2 & 1 & 1 & 9 & 5 & 1 & 0 \\ 0 & 9 & 5 & 3 & 0 & 4 & 8 & 3 \\ 4 & 2 & 7 & 2 & 1 & 9 & 0 & 6 \\ 9 & 7 & 9 & 8 & 0 & 4 & 2 & 4 \\ 5 & 2 & 1 & 8 & 4 & 1 & 0 & 9 \\ 1 & 8 & 5 & 4 & 9 & 2 & 3 & 8 \\ 3 & 7 & 1 & 2 & 3 & 4 & 4 & 6 \end{bmatrix}$$

e seja a matriz B (máscara) a seguir:

$$\begin{bmatrix} 2 & 1 & 0 \\ 1 & 1 & -1 \\ 0 & -1 & -2 \end{bmatrix}.$$

A operação de convolução bidimensional produzirá como resultado a matriz:

$$\begin{bmatrix} 20 & 10 & 2 & 26 & 23 & 6 & 9 & 4 \\ 18 & 1 & -8 & 2 & 7 & 3 & 3 & -11 \\ 14 & 22 & 5 & -1 & 9 & -2 & 8 & -1 \\ 29 & 21 & 9 & -9 & 10 & 12 & -9 & -9 \\ 21 & 1 & 16 & -1 & -3 & -4 & 2 & 5 \\ 15 & -9 & -3 & 7 & -6 & 1 & 17 & 9 \\ 21 & 9 & 1 & 6 & -2 & -1 & 23 & 2 \\ 9 & -5 & -25 & -10 & -12 & -15 & -1 & -12 \end{bmatrix}$$

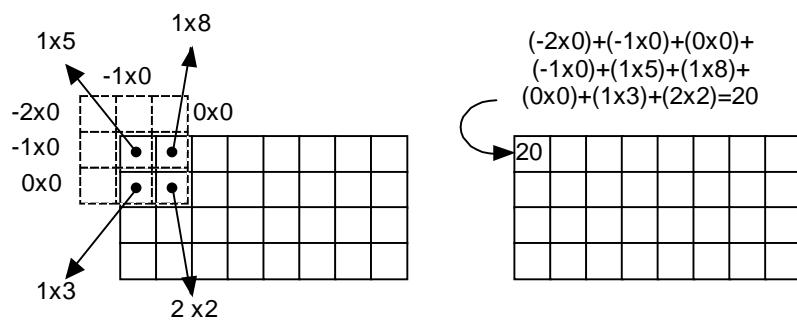


Figura 18 - Cálculo do primeiro valor da convolução de A por B.

A figura 18 ilustra em detalhes o cálculo do resultado correspondente ao pixel no canto superior esquerdo da imagem. Observar que a máscara B foi espelhada em relação a x e a y antes do cálculo das somas de produtos.

Para calcular os valores resultantes dos pixels próximos às bordas da imagem, podem ser adotadas diversas estratégias, dentre elas:

1. preencher com zeros o contorno da imagem, de maneira condizente com o tamanho da máscara utilizado, como ilustra a figura 18.

2. preencher o contorno da imagem com os mesmos valores da(s) primeira(s) e última(s) linha(s) e coluna(s).

3. prevenir a eventual introdução de erros nas regiões de bordas da imagem causados por qualquer um dos métodos acima, considerando na imagem resultante apenas os valores para os quais a máscara de convolução ficou inteiramente contida na imagem original.

A seguir, ilustraremos o uso do conceito de convolução com máscaras aplicado à deteção de características de imagens, particularmente pontos isolados, linhas e bordas.

2.4.1 Deteção de pontos isolados

A máscara a seguir é um exemplo de operador de convolução que, quando aplicado a uma imagem, destacará pixels brilhantes circundados por pixels mais escuros. Como será visto na seção 4.3, este operador corresponde a um filtro passa-altas.

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

2.4.2 Deteção de linhas

As máscaras a seguir podem ser utilizadas para a deteção de linhas horizontais e verticais (acima) e diagonais (abaixo).

$$\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

2.4.3 Deteção de bordas

O tema 'deteção de bordas' (*edge detection*) vem desafiando os pesquisadores da área de Processamento de Imagens há muitos anos e sobre ele continuam sendo experimentadas novas técnicas, cujos resultados são publicados ainda hoje nos mais conceituados periódicos científicos mundiais. Trata-se, portanto, de um tema em aberto, a deteção de bordas em cenas consideradas 'difíceis'.

Apenas a título de ilustração da operação de convolução com máscaras apresentamos a seguir alguns exemplos de máscaras que podem ser utilizadas para a tarefa de deteção de bordas.

Define-se borda (*edge*) como a fronteira entre duas regiões cujos níveis de cinza predominantes são razoavelmente diferentes. Pratt [Pratt 1991] define uma borda de luminosidade como uma descontinuidade na luminosidade de uma imagem. Analogamente, pode-se definir borda de textura ou borda de cor, em imagens onde as informações de textura ou

cor, respectivamente, são as mais importantes. Neste livro trataremos somente de bordas de luminosidade, às quais denominaremos simplesmente bordas.

Para a deteção e realce de bordas, aplicam-se habitualmente filtros espaciais lineares de dois tipos: (a) baseados no gradiente da função de luminosidade, $I(x,y)$, da imagem, e (b) baseados no laplaciano de $I(x,y)$.

Tanto o gradiente quanto o laplaciano costumam ser aproximados por máscaras de convolução ou operadores 3 x 3. Exemplos destas máscaras são os operadores de Roberts, Sobel, Prewitt e Frei-Chen, mostrados na tabela 5.

Tabela 5 - Operadores 3 x 3 utilizados para estimar a amplitude do gradiente através de uma borda.

Operador	Vertical	Horizontal
Roberts	$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Sobel	$\frac{1}{4} \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{4} \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$
Prewitt	$\frac{1}{3} \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{3} \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$
Frei-Chen	$\frac{1}{2+\sqrt{2}} \begin{bmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{2+\sqrt{2}} \begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ 1 & \sqrt{2} & 1 \end{bmatrix}$

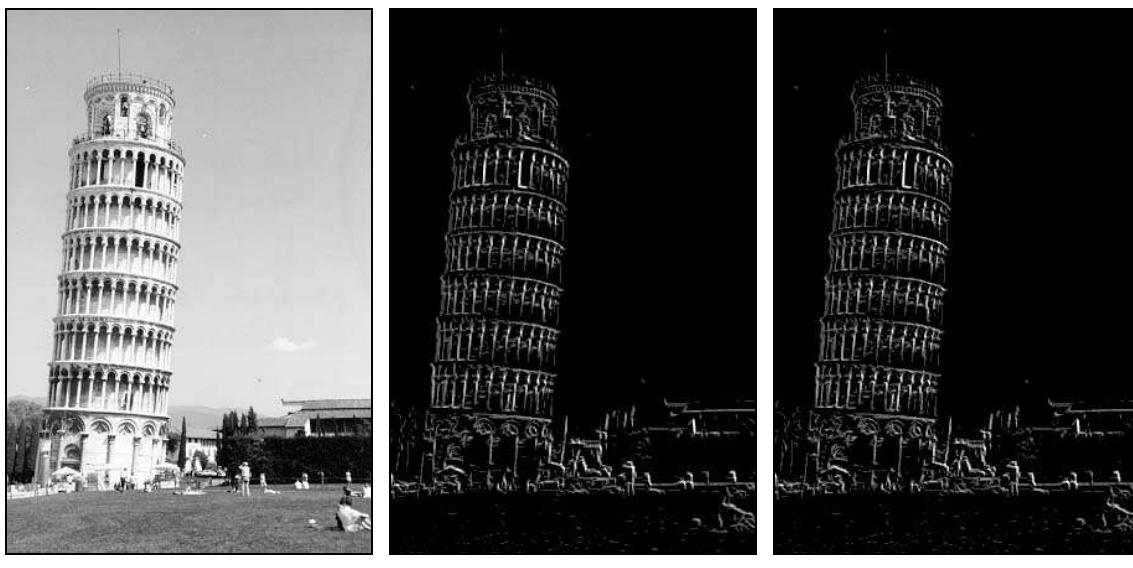


Figura 19 - Exemplo de realce e deteção de bordas. (a) imagem original, (b) realce de bordas utilizando os operadores de Prewitt horizontal e vertical, (c) realce de bordas utilizando os operadores de Sobel horizontal e vertical.

A figura 19 mostra os resultados da aplicação dos operadores de Prewitt e Sobel a uma imagem monocromática. Os resultados obtidos com a aplicação dos operadores verticais e

MARQUES FILHO, Ogê; VIEIRA NETO, Hugo. *Processamento Digital de Imagens*, Rio de Janeiro: Brasport, 1999. ISBN 8574520098.

horizontais foram combinados por meio de uma operação lógica OR. Notar que as diferenças são pouco perceptíveis.

O laplaciano é um operador definido como:

$$\nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad (2.9)$$

e que pode ser aproximado pelas máscaras da figura 20.

$$\begin{array}{c}
 \begin{matrix}
 0 & -1 & 0 \\
 -1 & 4 & -1 \\
 0 & -1 & 0
 \end{matrix}
 \quad
 \begin{matrix}
 -1 & -1 & -1 & -1 & -1 \\
 -1 & -1 & -1 & -1 & -1 \\
 -1 & -1 & 24 & -1 & -1 \\
 -1 & -1 & -1 & -1 & -1 \\
 -1 & -1 & -1 & -1 & -1
 \end{matrix}
 \quad
 \begin{bmatrix}
 -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
 -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
 -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
 -1 & -1 & -1 & 8 & 8 & 8 & -1 & -1 & -1 \\
 -1 & -1 & -1 & 8 & 8 & 8 & -1 & -1 & -1 \\
 -1 & -1 & -1 & 8 & 8 & 8 & -1 & -1 & -1 \\
 -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
 -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
 -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1
 \end{bmatrix} \\
 (a) \qquad \qquad (b) \qquad \qquad (c)
 \end{array}$$

Figura 20 - Máscaras para o cálculo do laplaciano: (a) 3 x 3, (b) 5 x 5, (c) 9 x 9.

A figura 21 mostra os resultados obtidos com cada uma das máscaras da figura 20 aplicadas a uma imagem monocromática.

Embora o laplaciano seja insensível à rotação, e portanto capaz de realçar ou detetar bordas em qualquer direção, seu uso é restrito devido a sua grande suscetibilidade a ruído.

A figura 22 mostra um exemplo de aplicação do laplaciano 3 x 3 acima a uma imagem monocromática com e sem ruído.

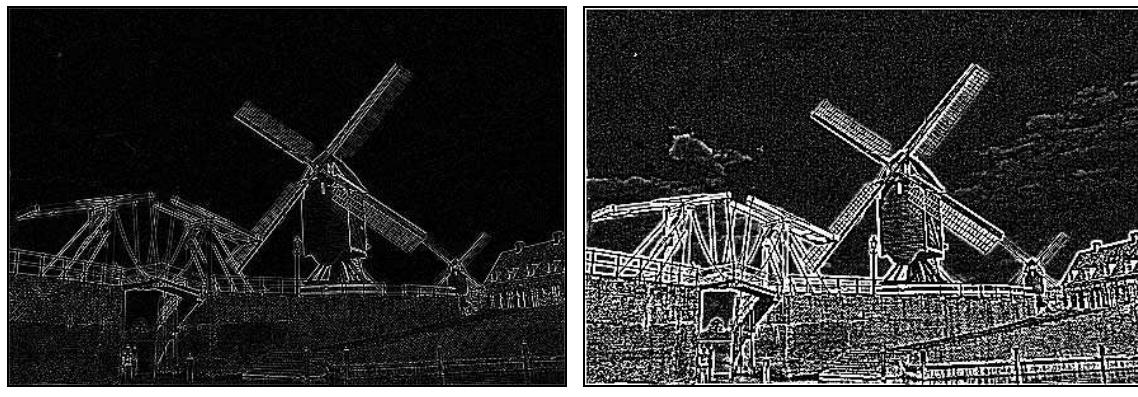
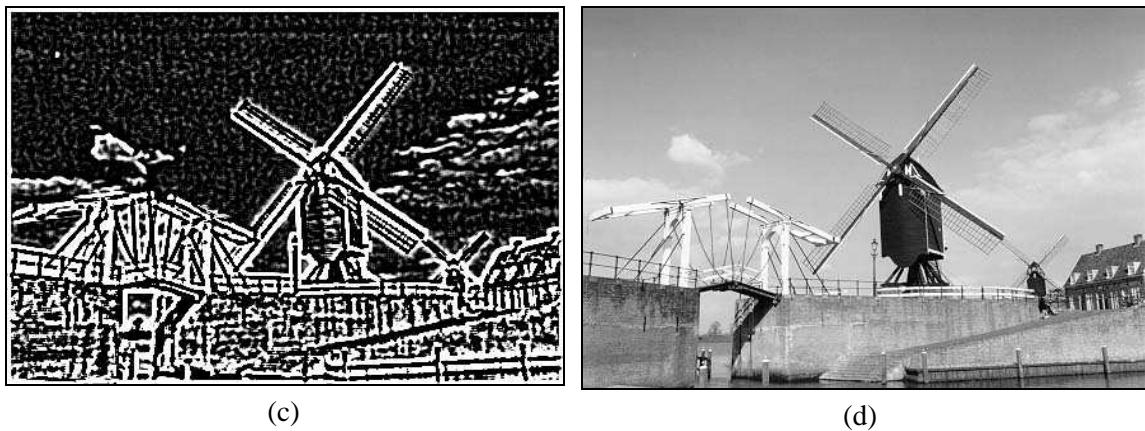


Figura 21 - Resultados da aplicação da máscara do laplaciano: (a) 3 x 3, (b) 5 x 5, (c) 9 x 9, (d) imagem original.



(c)

(d)

Figura 21 - Continuação.

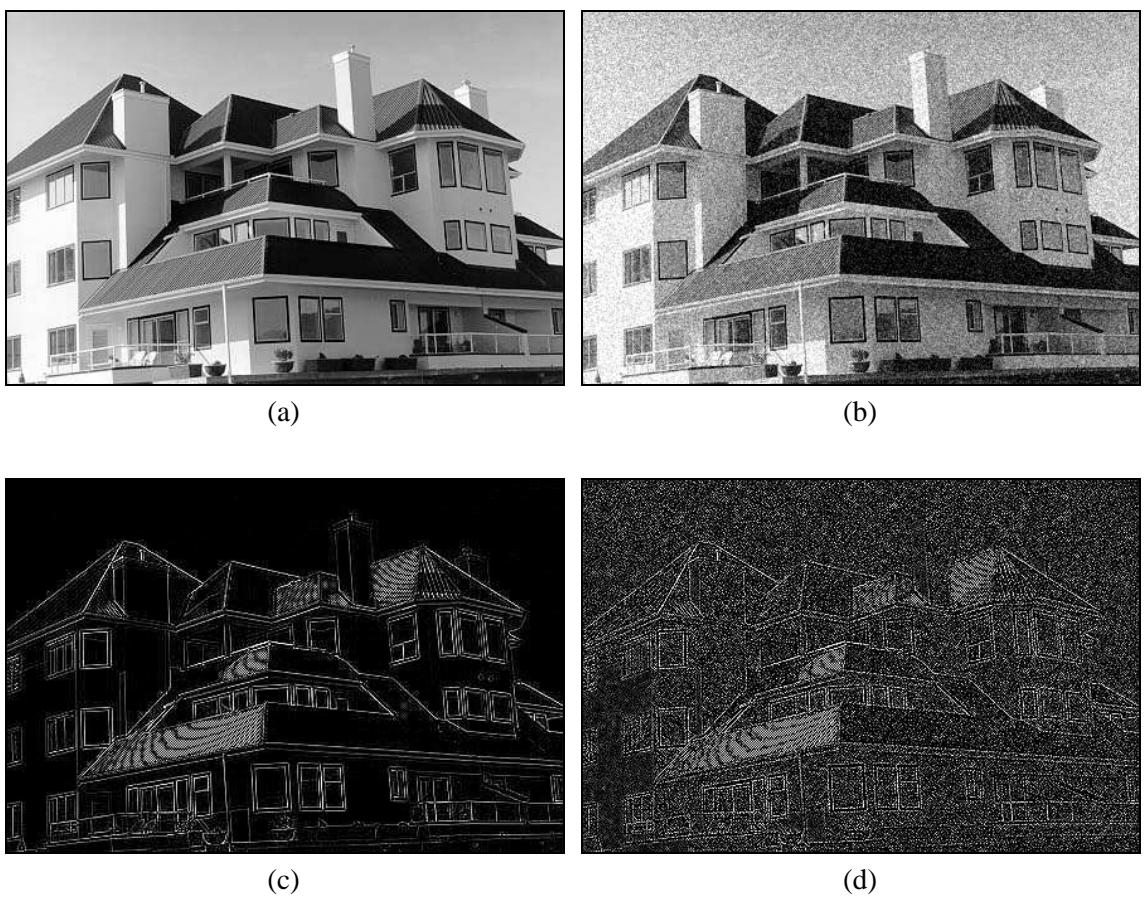


Figura 22 - Exemplo de utilização do laplaciano: (a) imagem original, (b) imagem ruidosa, (c) laplaciano sobre (a), (d) laplaciano sobre (b).

Existem outros operadores direcionais, que nada mais são que conjuntos de máscaras que representam aproximações discretas de bordas ideais em várias direções. Estes operadores incluem as máscaras direcionais introduzidas por Prewitt [Prewitt 1970], Kirsch [Kirsch 1971], e as máscaras simples de 3 e 5 níveis de Robinson [Robinson 1977]. A tabela 6 mostra estas máscaras com suas respectivas direções cardinais.

Tabela 6 - Máscaras de Prewitt, Kirsch e Robinson.

Direção da borda	Direção grad.	Prewitt	Kirsch	Robinson 3 níveis	Robinson 5 níveis
0	N	1 1 1 1 -2 1 -1 -1 -1	5 5 5 -3 0 -3 -3 -3 -3	1 1 1 0 0 0 -1 -1 -1	1 2 1 0 0 0 -1 -2 -1
1	NO	1 1 1 1 -2 -1 1 -1 -1	5 5 -3 5 0 -3 -3 -3 -3	1 1 0 1 0 -1 0 -1 -1	2 1 0 1 0 -1 0 -1 -2
2	O	1 1 -1 1 -2 -1 1 1 -1	5 -3 -3 5 0 -3 5 -3 -3	1 0 -1 1 0 -1 1 0 -1	1 0 -1 2 0 -2 1 0 -1
3	SO	1 -1 -1 1 -2 -1 1 1 1	-3 -3 -3 5 0 -3 5 5 -3	0 -1 -1 1 0 -1 1 1 0	0 -1 -2 1 0 -1 2 1 0
4	S	-1 -1 -1 1 -2 1 1 1 1	-3 -3 -3 -3 0 -3 5 5 5	-1 -1 -1 0 0 0 1 1 1	-1 -2 -1 0 0 0 1 2 1
5	SE	-1 -1 1 -1 -2 1 1 1 1	-3 -3 -3 -3 0 5 -3 5 5	-1 -1 0 -1 0 1 0 1 1	-2 -1 0 -1 0 1 0 1 2
6	E	-1 1 1 -1 -2 1 -1 1 1	-3 -3 5 -3 0 5 -3 -3 5	-1 0 1 -1 0 1 -1 0 1	-1 0 1 -2 0 2 -1 0 1
7	NE	1 1 1 -1 -2 1 -1 -1 1	-3 5 5 -3 0 5 -3 -3 -3	0 1 1 -1 0 1 -1 -1 0	0 1 2 -1 0 1 -2 -1 0
Fator de escala		1/5	1/15	1/3	1/4

Leitura complementar

Aos interessados em um aprofundamento matemático dos aspectos abordados nesta seção, recomendamos os capítulos 7 e 16 de [Pratt 1991].

A seção 6.4 de [Haralick e Shapiro 1992] trata dos temas convolução e correlação. A questão de deteção de bordas e linhas é vista no capítulo 7 do mesmo livro.

O capítulo 3 de [Dougherty e Giardina 1987] é inteiramente dedicado à deteção de bordas.

O artigo de Dawson [Dawson 1987] traz fragmentos de código em C para a convolução de imagens com máscaras 3 x 3 e explica sua possível utilização em processos de filtragem e deteção de bordas.

Prosise [Prosise 1994a] apresenta exemplos de máscaras de convolução úteis para produção de efeitos em imagens, tais como realce, borramento (*blurring*) e o efeito de baixo relevo (*emboss*).

Jain [Jain 1989] apresenta o conceito de gradientes estocásticos para resolver o problema da deteção de bordas em imagens ruidosas.

2.5 Transformações geométricas

Transformações geométricas são operações de processamento de imagens cujo principal efeito é a alteração da posição espacial dos pixels que a compõem. Elas costumam ser úteis em situações que vão desde a correção de distorções até a produção de efeitos artísticos sobre imagens.

2.5.1 Ampliação e redução (zoom)

As operações de ampliação e redução de imagens (em inglês, *zoom in* e *zoom out*, respectivamente) são processos pelos quais as dimensões de uma imagem são aumentadas ou diminuídas para efeito de visualização. A maneira mais simples de ampliar uma imagem é duplicar os valores dos pixels na direção X ou Y ou em ambas. Se o fator de ampliação não for o mesmo para as duas direções, a razão de aspecto (relação entre a dimensão horizontal e a vertical de uma imagem) da imagem será alterada.

Para expandir uma imagem por um fator 2, cada pixel é copiado 4 vezes na imagem resultante, conforme ilustra a figura 23. Convém notar que a resolução da imagem não é alterada, apenas seu tamanho para efeito de visualização.

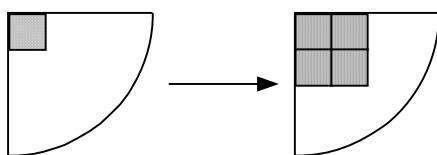


Figura 23 - Expansão de um pixel em 4 (*zoom 2x*)

Para reduzir as dimensões de uma imagem de um fator 2, basta utilizar o processo inverso, isto é converter cada agrupamento de quatro pixels novamente em 1 pixel. O problema neste caso é que normalmente estes pixels apresentarão valores diferentes de cinza, o que equivale a dizer que poderá haver perda de informação no processo de *zoom out*. Para minimizar este aspecto, uma técnica comum é substituir na imagem resultante o valor do pixel pela média dos quatro pixels equivalentes na imagem original.

Para um *zoom* de quatro vezes, utiliza-se uma vizinhança de dezesseis pixels e assim por diante. Para ampliar ou reduzir uma imagem de um fator fracionário, são necessários algoritmos de interpolação cujo detalhamento foge ao escopo deste livro.

A figura 24 mostra exemplos de *zoom in* e *zoom out* para imagens monocromáticas.

2.5.2 Alterações de dimensões (*scaling* e *sizing*)

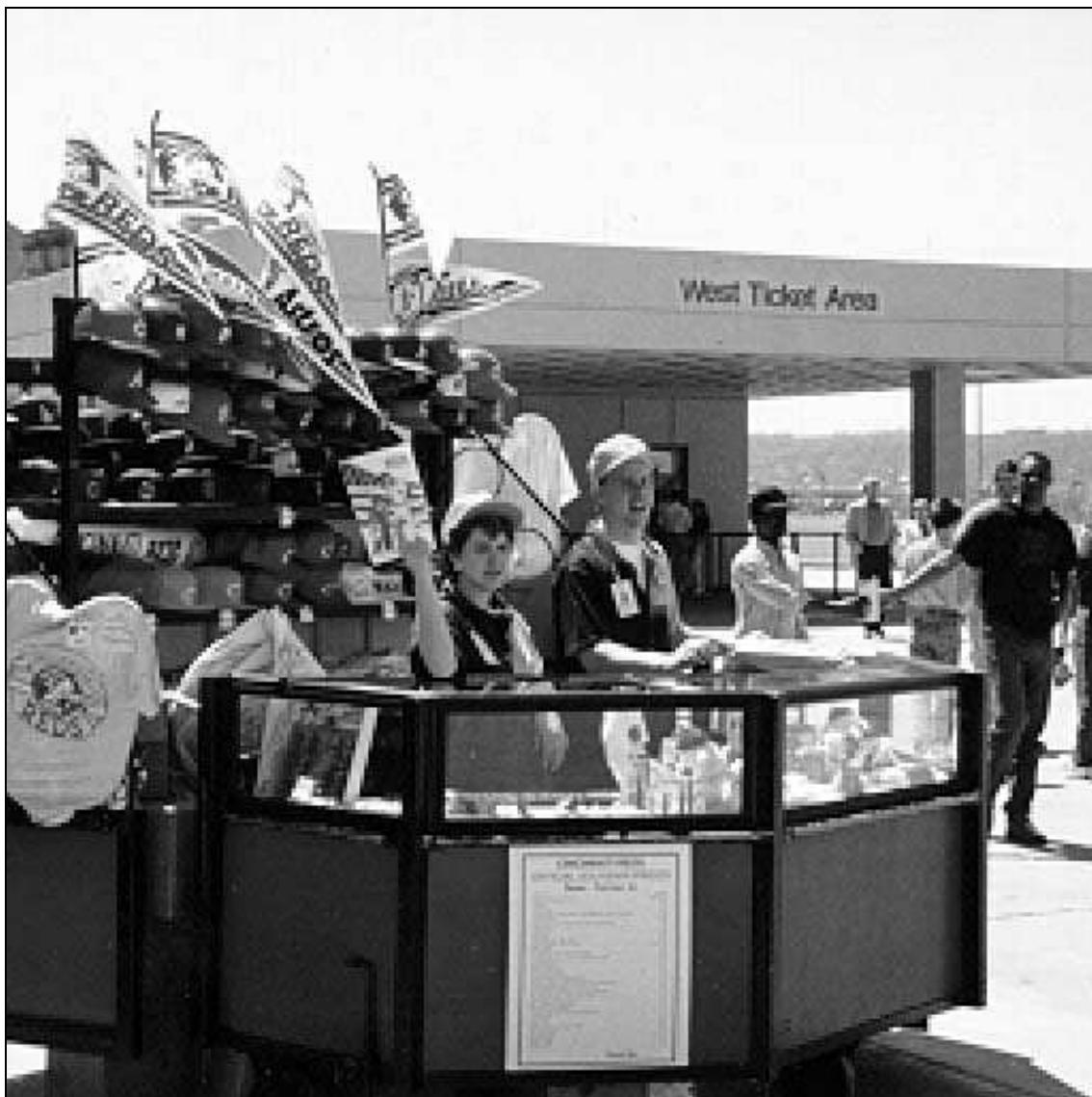
As modificações de uma imagem descritas anteriormente visam predominantemente facilitar a visualização e, via de regra, não representam alterações nas dimensões reais da imagem. Ou seja, quando dissemos que um pixel é multiplicado por 4, o leitor deve entender que um pixel da imagem que originalmente ocuparia um ponto do dispositivo de exibição passará a ocupar quatro pontos, sem afetar em nada as dimensões reais da imagem.

Caso nosso objetivo seja a alteração de dimensões da imagem, as técnicas descritas acima podem ser utilizadas, com a diferença de que a saída será o arquivo contendo a imagem alterada (ampliada/reduzida) e não o resultado visual de sua exibição em maior ou menor tamanho na tela. A literatura técnica de processamento de imagens por vezes distingue dois tipos de alterações de dimensões de uma imagem, embora tecnicamente idênticos:

1. o processo denominado *scaling* refere-se ao caso em que a imagem é ampliada ou reduzida por um fator (que pode ser igual para as dimensões horizontal e vertical – preservando a relação de aspecto original – ou não);
2. o nome *sizing* (algumas vezes *resizing*) é utilizado nos casos em que, ao invés de especificar o fator de ampliação / redução, o usuário especifica o novo tamanho que a imagem deve possuir.



(a)



(b)

Figura 24 - (a) imagem original, (b) imagem ampliada (*zoom in*) de 2 vezes; (c) imagem reduzida (*zoom out*) de 2 vezes.



(c)

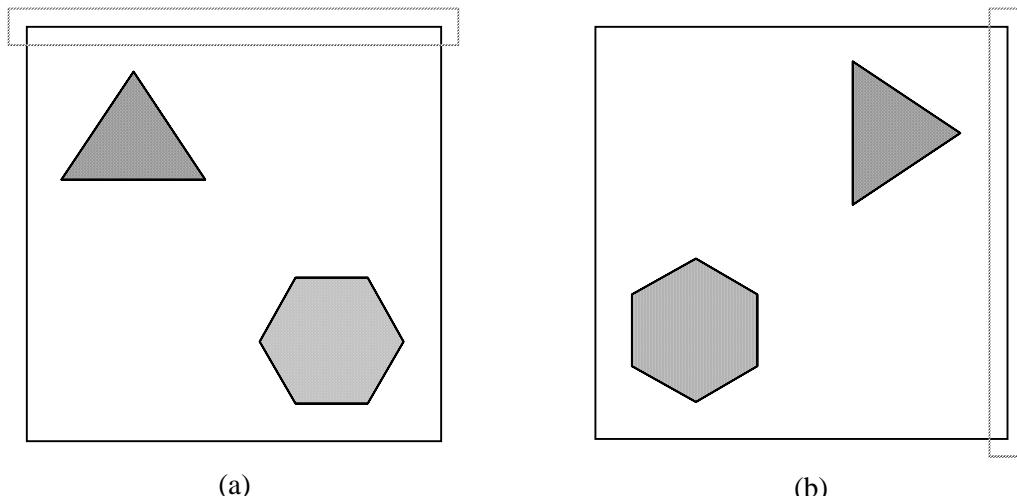
Figura 24 - Continuação.

2.5.3 Translação

A translação de uma imagem consiste basicamente no deslocamento linear de cada pixel de coordenadas (X, Y) na horizontal e/ou na vertical, mapeando para o ponto de coordenadas (X', Y') , calculadas como: $(X', Y') = (X + \Delta X, Y + \Delta Y)$, onde ΔX é o deslocamento vertical e ΔY é o deslocamento horizontal (em pixels).

2.5.4 Rotação

Uma imagem pode ser rotacionada de um ângulo arbitrário, tanto no sentido horário quanto no anti-horário. Rotações com ângulos múltiplos de 90° são mais simples de implementar, pois consistem na cópia de pixels que estão organizados em linhas, reordenando-os em colunas na direção em que se deseja rotacionar a imagem. A figura 25 ilustra o processo de rotação de 90° no sentido horário. A área tracejada destaca as primeiras linhas da imagem original, que são repositionadas em formas de colunas, da direita para a esquerda, na imagem rotacionada.

Figura 25 - Exemplo de rotação de 90° no sentido horário.

A rotação por ângulos quaisquer é uma tarefa mais complexa, que pode ser implementada usando as técnicas de *warping* descritas adiante. Matematicamente, a rotação de cada ponto (X, Y) de uma imagem por um ângulo arbitrário Ang , mapeará este ponto na localidade de coordenadas (X', Y') , onde X' e Y' são calculados pelas equações:

$$X' = X \cos(Ang) + Y \sin(Ang) \quad (2.10)$$

$$Y' = Y \cos(Ang) - X \sin(Ang) \quad (2.11)$$

O processo de rotação normalmente exige a correção de razão de aspecto da imagem resultante, tendo em vista que diversos modos de exibição de vídeo utilizam pixels não-quadrados.

2.5.5 Espelhamento (*Flip*)

O espelhamento (*flip*) é uma operação que combina a rotação por ângulos múltiplos de 90° com o cálculo de matriz transposta. Um *flip* horizontal nada mais é que uma rotação de 90° no sentido anti-horário (ou 270° no sentido horário) da versão transposta da imagem, enquanto um *flip* vertical é uma rotação de 90° no sentido horário (ou 270° no sentido anti-horário) da versão transposta da imagem. A figura 26 mostra exemplos de *flip* horizontal e vertical.

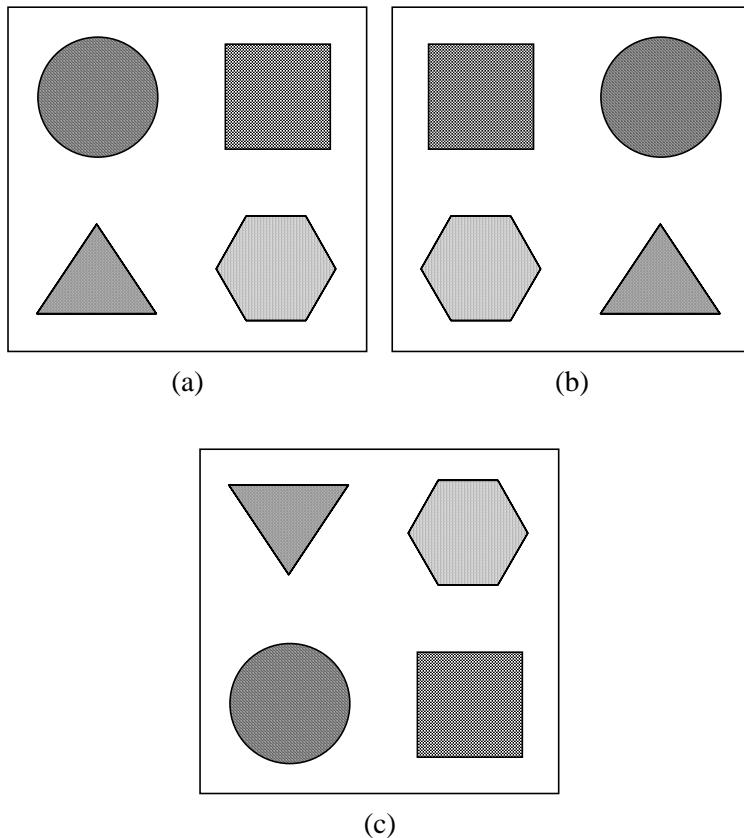


Figura 26 - (a) Imagem original, (b) *flip* horizontal, (c) *flip* vertical.

2.5.6 *Warping*

Warping é o nome dado ao processo de alteração de uma imagem de tal modo que a relação espacial entre seus objetos e características é alterada conforme outra imagem ou gabarito (*template*).

A transformação matemática mais comum é baseada na projeção afim (*affine projection*), dada pelas equações:

$$X' = \frac{aX + bY + c}{iX + jY + 1} \quad (2.12)$$

$$Y' = \frac{dX + eY + f}{iX + jY + 1} \quad (2.13)$$

onde X e Y são as coordenadas antigas e X' e Y' as novas. Os coeficientes a, b, c, d, e, f, i e j são determinados a partir de um conjunto de pontos de controle que correspondem à congruência

desejada entre as duas imagens ou entre a imagem original e o template selecionado. A figura 27 mostra um exemplo do processo de *warping* aplicado a uma imagem binária simples.

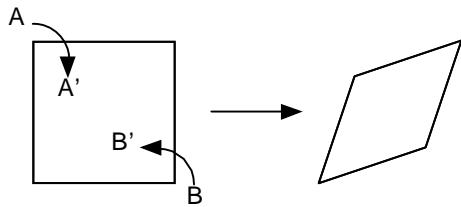


Figura 27 - Exemplo de *warping*.

Neste exemplo, o ponto A é movido para a posição A' e B é movido para B'. A projeção afim exige que sejam selecionados quatro pares de pontos de controle, para resultar um sistema de oito equações a oito incógnitas. Os pares A-A' e B-B' são duas escolhas óbvias. Os outros dois pontos escolhidos, neste exemplo, são os dois cantos restantes do quadrado (que permanecerão inalterados). Se tivéssemos selecionado mais de quatro pontos de controle, um ajuste por mínimos quadrados seria necessário para determinar os melhores valores para a transformação.

Na prática, um programa para a solução simultânea de um sistema de equações é utilizado para calcular os valores dos coeficientes. Então, entrando com as coordenadas X' e Y' da imagem destino, calcula-se os valores correspondentes de X e Y na imagem original. O nível de cinza do ponto de coordenadas (X,Y) é então atribuído à posição (X',Y') na imagem destino. Este processo de mapeamento pode ser facilmente executado em paralelo, pois cada ponto na imagem resultante depende de apenas um ponto da imagem original.

A figura 28 mostra um exemplo de *warping* utilizando imagem monocromática.

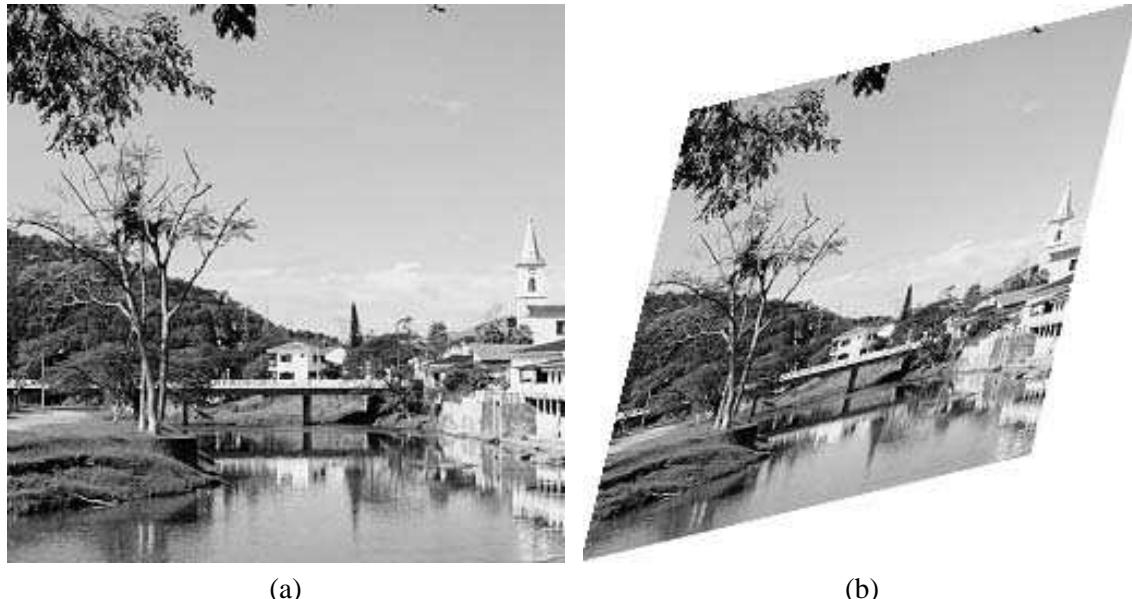


Figura 28 - Exemplo de *warping* de uma imagem monocromática utilizando padrão (*template*) em forma de losango.

Exercício resolvido

Baseando-se na figura 27, dadas as coordenadas originais dos vértices do quadrado e as coordenadas desejadas para o quadrado após o *warping*, indicadas na tabela a seguir e ilustradas na figura 29, calcular os valores dos coeficientes a, b, c, d, e, f, i e j correspondentes à transformação desejada.

Ponto de controle	Coordenadas originais (X, Y)	Coordenadas após <i>warping</i> (X', Y')
1	(0,0)	(2,2)
2	(4,4)	(3,3)
3	(4,0)	(4,0)
4	(0,4)	(0,4)

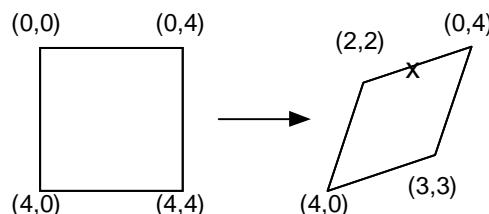


Figura 29 - Coordenadas dos pontos de controle antes e depois do *warping*.

Solução:

Entrando com estes valores para as coordenadas e resolvendo o sistema de equações resultante (eqs. 2.12 e 2.13), obtemos os seguintes valores para os coeficientes:

$$a = 0,75$$

$$b = -0,25$$

$$c = 1,00$$

$$d = -0,25$$

$$e = 0,75$$

$$f = 1,00$$

$$i = 0,00$$

$$j = 0,00$$

Para verificar se os coeficientes calculados estão corretos, podemos escolher um ponto na imagem modificada, por exemplo o ponto de coordenadas (0,5, 2,5), indicado com uma cruz na figura 29. Calculando os valores de X e Y correspondentes a este ponto na imagem original, obteremos o par (0, 2), como esperado.

2.5.7 Cropping, cutting e pasting

Recortar e colar trechos de imagens para compor novas imagens são operações corriqueiras de manipulação de imagens. Existem três formas de se recortar uma imagem. A primeira e mais simples consiste em utilizar uma região retangular, definida pelas coordenadas de dois de seus vértices. A segunda consiste em utilizar uma figura geométrica regular qualquer ou um polígono, regular ou não. A terceira e mais complexa consiste em se permitir delimitar a área de recorte 'à mão livre' utilizando o mouse ou dispositivo equivalente. O detalhamento de tais operações, por se enquadrarem mais no contexto de manipulação de imagens, foge ao objetivo do texto.

Leitura complementar

Para maiores detalhes sobre os aspectos de interpolação necessários à implementação de diversas transformações geométricas apresentadas nesta seção, recomendamos as subseções 4.3.2 e 14.5.1 de [Pratt 1991] e o capítulo 12 de [Lindley 1991].

Dawson [Dawson 1987] traz fragmentos de código em C para a execução de transformações geométricas sobre imagens, particularmente rotação e *resizing*. Em outro artigo

[Dawson 1989], ele apresenta trechos de código em C para operações de *cutting*, *pasting* e *warping*.

Prosise [Prosise 1994b] apresenta um programa (*AllPaper*) para redimensionar arquivos em formato BMP.

Exercícios Propostos

- Dados os dois subconjuntos de imagem S_1 e S_2 a seguir e sendo $V = \{1\}$, determinar se S_1 e S_2 estão: (a) 4-conectados, (b) 8-conectados, (c) m -conectados.

S_1						S_2			
0	1	0	0	0	1	0	0	0	0
1	0	0	1	0	0	0	0	1	0
1	0	1	1	0	1	1	1	0	1
1	0	1	1	1	0	0	0	1	0
0	0	0	1	0	0	1	0	0	0

- Dadas as matrizes X e Y a seguir, correspondentes a trechos 3×3 de imagens de 256 tons de cinza, efetuar a subtração $X-Y$ e informar: (a) o resultado intermediário (sem considerações de *underflow* e *overflow*), (b) o resultado final utilizando normalização, (c) o resultado final utilizando truncamento.

$$X = \begin{bmatrix} 200 & 100 & 100 \\ 0 & 10 & 50 \\ 50 & 250 & 120 \end{bmatrix} \quad Y = \begin{bmatrix} 100 & 220 & 230 \\ 45 & 95 & 120 \\ 205 & 100 & 0 \end{bmatrix}$$

- Considere o trecho de imagem a seguir, representado por uma matriz, onde cada elemento da matriz corresponde ao nível de cinza do pixel correspondente.

Seja $V = \{250, 251, 252, 253, 254, 255\}$. Calcular as distâncias D_4 , D_8 e D_m entre p e q.

p			
250	253	254	253
251	253	16	54
76	255	254	65
38	16	17	255

q

- Supondo que se deseja transmitir à distância uma imagem, utilizando um protocolo de comunicação em que a imagem é dividida em pacotes, onde cada pacote contém 1 bit de início (*start bit*), um byte (8 bits) de informação e 1 bit de término (*stop bit*), responder:
 - Qual o tempo necessário para se transmitir uma imagem de 512×512 pixels, com 256 níveis de cinza, à velocidade de 9600 bps ?
 - Qual seria o tempo de transmissão da mesma imagem à velocidade de 28800 bps ?
- Dada a imagem binária a seguir, onde os pixels marcados com 1 são pretos e os demais brancos, esboçar o resultado da aplicação passo a passo da técnica de suavização de imagens binárias descrita a seguir e comentar os resultados após cada etapa.



	1	1		1	1	1			
	1	1	1	1	1	1	1		
1		1	1	1		1			
	1	1	1	1	1	1			
	1	1	1	1	1	1			
	1	1	1	1	1	1			
			1					1	
1									

Imagen Original

Técnica: Substituição do valor do pixel de referência de uma janela 3 x 3, pelo resultado da aplicação sucessiva de 6 expressões booleanas aos pixels situados naquela janela, onde se utilizam as seguintes convenções:

a	b	c
d	p	e
f	g	h

As expressões são:

$$B_1 = p \vee b \wedge g \wedge (d \vee e) \vee d \wedge e \wedge (b \vee g)$$

$$B_2 = p \wedge [(a \vee b \vee d) \wedge (e \vee g \vee h) \vee (b \vee c \vee e) \wedge (d \vee f \vee g)]$$

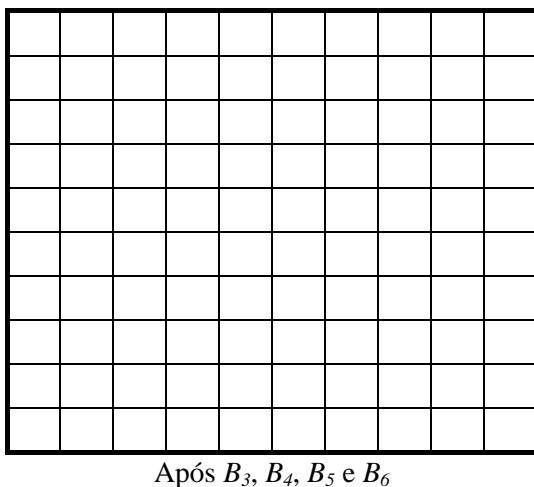
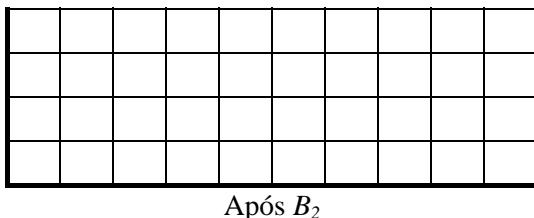
$$B_3 = \text{NOT}(p) \wedge (d \wedge f \wedge g) \wedge \text{NOT}(a \vee b \vee c \vee e \vee h) \vee p$$

$$B_4 = \text{NOT}(p) \wedge (a \wedge b \wedge d) \wedge \text{NOT}(c \vee e \vee f \vee g \vee h) \vee p$$

$$B_5 = \text{NOT}(p) \wedge (e \wedge g \wedge h) \wedge \text{NOT}(a \vee b \vee c \vee d \vee f) \vee p$$

$$B_6 = \text{NOT}(p) \wedge (b \wedge c \wedge e) \wedge \text{NOT}(a \vee d \vee f \vee g \vee h) \vee p$$

Após B_1



6. Demonstrar que a aplicação das máscaras de Prewitt sobre uma imagem equivale à aplicação de uma máscara de diferenciação do tipo [-1 0 1] (ou sua equivalente vertical) seguida de uma máscara do tipo [1 1 1] (ou sua equivalente vertical). Pede-se demonstrar para ambos os casos (Prewitt vertical e Prewitt horizontal).
7. Um pacote aplicativo de processamento de imagens utiliza um formato proprietário de arquivos de imagem, no qual os primeiros 32 bytes são reservados para o cabeçalho, onde estão, dentre outras, as informações das dimensões vertical e horizontal da imagem. Após o cabeçalho, a imagem é armazenada no arquivo na base de 1 byte por pixel, linha após linha, sem nenhum tipo de compactação. Sabendo que as imagens armazenadas neste formato são representadas em 256 tons de cinza, qual será o tamanho (em bytes) de um arquivo de imagem contendo 230 pixels na horizontal e 100 pixels na vertical?
8. Dadas as matrizes X e Y a seguir, correspondentes a trechos 3 x 3 de imagens de 256 tons de cinza, obter: (a) o resultado da operação lógica $X \vee Y$; (b) o resultado da operação lógica $X \wedge Y$; (c) o resultado da operação lógica $X \oplus Y$. Sugestão: converter os valores originais para seus equivalentes em base 2, efetuar as operações lógicas bit a bit e finalmente reconverter os valores resultantes para a base 10.

$$X = \begin{bmatrix} 200 & 100 & 100 \\ 0 & 10 & 50 \\ 50 & 250 & 120 \end{bmatrix} \quad Y = \begin{bmatrix} 100 & 220 & 230 \\ 45 & 95 & 120 \\ 205 & 100 & 0 \end{bmatrix}$$

No computador

Para consolidar os conhecimentos teóricos deste capítulo, recomendamos as práticas n^{os}. 1, 2 e 3 (Apêndice B).

Na Internet

Dentre as diversas referências disponíveis na WWW correlatas a este capítulo, destacamos:

"<http://www.cogs.susx.ac.uk/users/davidy/teachvision/vision2.html>"
Sussex Computer Vision: TEACH VISION2

"http://www.eecs.wsu.edu/IPdb/Enhancement/resolution_enhancement.html"
Resolution Enhancement

"<http://rfv.insa-lyon.fr/~jolion/Cours/cont.html>"
Contrast Analysis Demo

No curso interativo de processamento de imagens disponível no *site* da Unicamp, particularmente, encontram-se diversas páginas relacionadas ao conteúdo deste capítulo, cujos endereços e títulos aparecem a seguir:

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c9/s1/front-page.html>"
Edge Detection I

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c9/s2/front-page.html>"
Edge Detection II

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c6/s5/front-page.html>"
Correlation

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c6/s6/front-page.html>"
Image Enlargement

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c6/s7/front-page.html>"
Image Reduction

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c6/s1/front-page.html>"
Convolution Principles

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c2/s9/front-page.html>"
Translation, Rotation, Scaling / Geometric Transformations I

Bibliografia

- | | |
|---------------|---|
| [Dawson 1987] | Dawson, B.M., "Introduction to Image Processing Algorithms", <i>Byte</i> , Março 1987, 169-186. |
| [Dawson 1989] | Dawson, B.M., "Changing Perceptions of Reality", <i>Byte</i> , Dezembro 1989, 293-304. |

- [Dougherty e Giardina 1987] Dougherty, E.R. e Giardina, C.R., *Matrix Structured Image Processing*, Prentice-Hall, 1987.
- [Faugeras 1993] Faugeras, O.D., *Three-Dimensional Computer Vision*, MIT Press, 1993.
- [Gonzalez e Woods 1992] Gonzalez, R.C. e Woods, R.E., *Digital Image Processing - Third Edition*, Addison-Wesley, 1992.
- [Haralick e Shapiro 1992] Haralick, R.M. e Shapiro, L.G., *Computer and Robot Vision - Volume 1*, Addison-Wesley, 1992.
- [Huang 1965] Huang, T.S., "PCM Picture Transmission", *IEEE Spectrum*, 2, 12, 57-63.
- [Jain 1989] Jain, A.K., *Fundamentals of Digital Image Processing*, Prentice-Hall, 1989.
- [Kirsch 1971] Kirsch, R., "Computer determination of the constituent structure of biological images", *Computers and Biomedical Research* 4, 1971, 315-328.
- [Lindley 1991] Lindley, C.A., *Practical Image Processing in C*, Wiley, 1991.
- [Nince 1991] Nince, U. S., *Sistemas de Televisão e Vídeo*, LTC, 1991.
- [Pavlidis 1982] Pavlidis, T., *Algorithms for Graphics and Image Processing*, Computer Science Press, 1982.
- [Pratt 1991] Pratt, W. K., *Digital Image Processing*, Wiley Interscience, 1991. (2nd ed.)
- [Prewitt 1970] Prewitt, J.M., "Object enhancement and extraction" in B.S. Lipkin and A. Rosenfeld *Picture processing and psychopictorics*, Academic Press, 1970.
- [Prosise 1994a] Prosise, J., "Make Your Digital Images Shine", *PC Magazine*, 13 de Setembro de 1994, 319-322.
- [Prosise 1994b] Prosise, J., "Turn Wallpaper Into AllPaper", *PC Magazine*, 13 de Setembro de 1994, 350-360.
- [Robinson 1977] Robinson, G.S., "Edge detection by compass gradient masks.", *Computer Graphics and Image Processing* 6, 1977, 492-501.
- [Schalkoff 1989] Schalkoff, R.J., *Digital Image Processing and Computer Vision*, Wiley, 1989.

Capítulo 3

Técnicas de Modificação de Histograma

Este capítulo é inteiramente dedicado à definição e utilização do conceito de histograma. A Seção 3.1 conceitua histograma e dá exemplos de histogramas de imagens. Na Seção 3.2 uma técnica ponto a ponto de processamento de imagens, a transformação de intensidade, é apresentada. As seções 3.3 a 3.5 apresentam técnicas de modificação de histograma de imagens monocromáticas. Finalmente, a Seção 3.6 introduz o conceito de limiarização (*thresholding*) de imagens, como exemplo de utilização da informação contida em um histograma.

3.1 Conceito de histograma

O histograma de uma imagem é simplesmente um conjunto de números indicando o percentual de pixels naquela imagem que apresentam um determinado nível de cinza. Estes valores são normalmente representados por um gráfico de barras que fornece para cada nível de cinza o número (ou o percentual) de pixels correspondentes na imagem. Através da visualização do histograma de uma imagem obtemos uma indicação de sua qualidade quanto ao nível de contraste e quanto ao seu brilho médio (se a imagem é predominantemente clara ou escura).

Cada elemento deste conjunto é calculado como:

$$p_r(r_k) = \frac{n_k}{n} \quad (3.1)$$

onde:

$$0 \leq r_k \leq 1$$

$k = 0, 1, \dots, L-1$, onde L é o número de níveis de cinza da imagem digitalizada;

n = número total de pixels na imagem;

$p_r(r_k)$ = probabilidade do k -ésimo nível de cinza;

n_k = número de pixels cujo nível de cinza corresponde a k .

Exemplo

Os dados da tabela 1 correspondem a uma imagem de 128 x 128 pixels, com 8 níveis de cinza. O número de pixels correspondentes a um certo tom de cinza está indicado na segunda coluna, enquanto as respectivas probabilidades $p_r(r_k)$ aparecem na terceira coluna. A representação gráfica equivalente deste histograma é mostrada na figura 1.

Um histograma apresenta várias características importantes. A primeira delas é que cada $p_r(r_k)$ fornece, como sugere a notação, a probabilidade de um pixel da imagem apresentar nível de cinza r_k . Portanto, um histograma nada mais é que uma função de distribuição de probabilidades e como tal deve obedecer aos axiomas e teoremas da teoria de probabilidade. Por exemplo, é possível verificar que na tabela 1 a soma dos valores de $p_r(r_k)$ é 1, o que já era esperado.

Tabela 1 - Exemplo de histograma.

Nível de cinza (r_k)	n_k	$p_r(r_k)$
0	1120	0,068
1/7	3214	0,196
2/7	4850	0,296
3/7	3425	0,209
4/7	1995	0,122
5/7	784	0,048
6/7	541	0,033
1	455	0,028
Total	16384	1

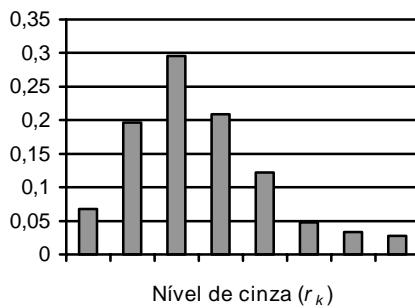


Figura 1 - Exemplo de histograma para imagem com oito níveis de cinza.

A figura 2 apresenta cinco exemplos de tipos de histogramas freqüentemente encontrados em imagens. O histograma da figura 2(a) apresenta grande concentração de pixels nos valores mais baixos de cinza, correspondendo a uma imagem predominantemente escura. Na figura 2(b) os pixels estão concentrados em valores próximos ao limite superior da escala de cinza, caracterizando uma imagem clara. Na parte (c) da figura, os pixels estão agrupados em torno de valores intermediários de cinza, correspondendo a uma imagem de brilho médio. Nas figuras (a), (b) e (c) a maioria dos pixels está concentrada em uma estreita faixa da escala de cinza, significando que as imagens correspondentes apresentam baixo contraste. A figura 2(d) corresponde a uma imagem com pixels distribuídos ao longo de toda a escala de cinza. É comum dizer que uma imagem com estas características apresenta um bom contraste. A figura 2(e) mostra um histograma tipicamente bimodal, isto é, apresentando duas concentrações de pixels, uma delas em torno de valores escuros e outra na região clara da escala de cinza. Pode-se dizer que a imagem correspondente apresenta alto contraste entre as duas concentrações, uma vez que elas se encontram razoavelmente espaçadas.¹

Para verificar a relação entre imagens e respectivos histogramas, a figura 3 mostra cinco imagens monocromáticas cujos histogramas são aqueles da figura 2.

¹ Convém observar que os conceitos de alto e baixo contraste neste caso somente estão relacionados ao espaçamento médio entre as raias do histograma. Já o termo 'bom contraste' deve ser utilizado com cautela, pois pode exprimir distribuição equitativa das raias ao histograma ao longo da escala de cinza – como foi utilizado neste caso – ou uma opinião subjetiva sobre a qualidade de uma imagem, que não poderia ser extraída somente da observação de seu histograma.

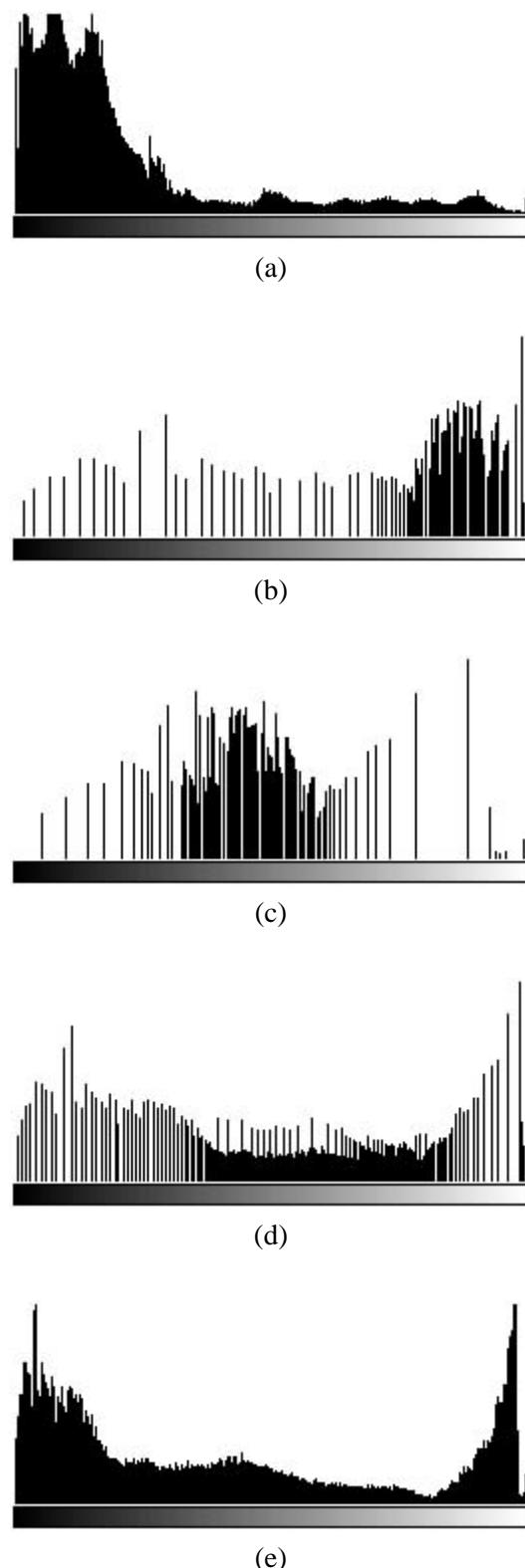


Figura 2 - Exemplos de histogramas.

O conceito de histograma também é aplicável a imagens coloridas. Neste caso, a imagem é decomposta de alguma forma (por exemplo, em seus componentes R, G e B) e para cada componente é calculado o histograma correspondente. A figura 4 (ver Seção *Figuras Coloridas*) mostra um exemplo de imagem colorida e seus histogramas R, G e B.

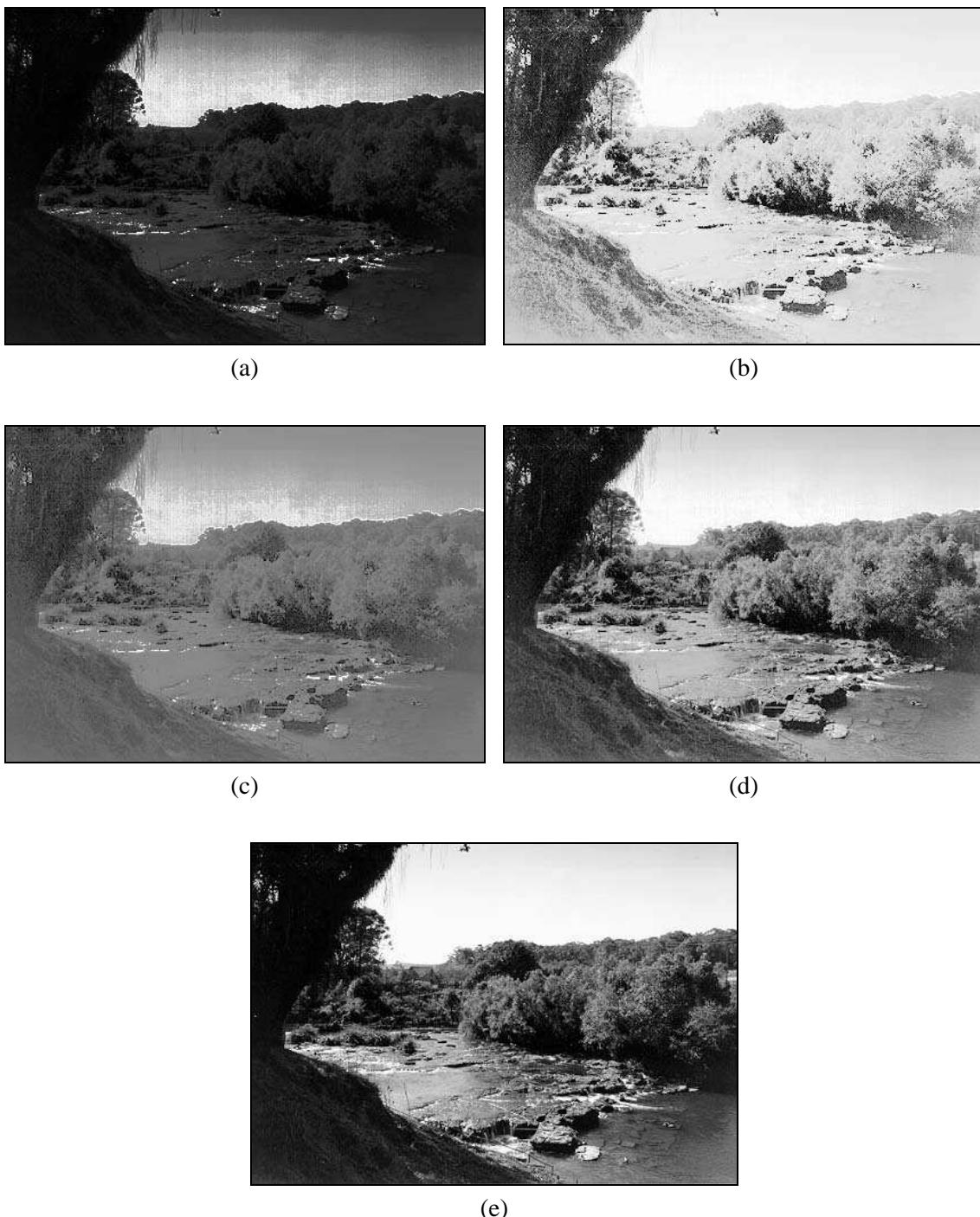


Figura 3 - Imagens correspondentes aos histogramas da figura 2.

Para computar o histograma de uma imagem monocromática, inicializa-se com zero todos os elementos de um vetor de L elementos, onde L é o número de tons de cinza possíveis. Em seguida, percorre-se a imagem, pixel a pixel, e incrementa-se a posição do vetor cujo índice corresponde ao tom de cinza do pixel visitado. Após toda a imagem ter sido percorrida, cada elemento do vetor conterá o número de pixels cujo tom de cinza equivale ao índice do elemento. Estes valores poderão ser normalizados, dividindo cada um deles pelo total de pixels na imagem.

Concluindo esta Seção, convém enfatizar que, embora o histograma de uma imagem forneça diversas informações qualitativas e quantitativas sobre ela (e.g. nível de cinza mínimo, médio e máximo, predominância de pixels claros ou escuros etc.), outras conclusões de caráter qualitativo (e.g. qualidade subjetiva global da imagem, presença ou não de ruído etc.) somente

podem ser extraídas dispondendo-se da imagem propriamente dita. Tal fato pode ser confirmado a partir de uma análise das figuras 2 e 3.

Leitura complementar

Para uma revisão da teoria básica de probabilidade, sugerimos [Ross 1994].

O capítulo 9 de [Lindley 1991] e o artigo de Dawson [Dawson 1987] apresentam código-fonte em C para cálculo e exibição de histogramas de imagens monocromáticas.

O capítulo 3 de [Pavlidis 1982] apresenta algoritmos para obtenção e equalização do histograma de uma imagem monocromática.

3.2 Transformações de intensidade

As técnicas de modificação de histograma são conhecidas como técnicas ponto-a-ponto, uma vez que o valor de tom de cinza de um certo pixel após o processamento depende apenas de seu valor original. Em contraste, nas técnicas de processamento orientadas a vizinhança, o valor resultante depende também, de alguma forma, dos pixels que circundam o elemento de imagem original.

Diversas técnicas de modificação da distribuição dos pixels na escala de cinza podem ser implementadas a partir do conceito de transformações de intensidade, apresentado formalmente a seguir.

Seja uma variável f , representando o nível de cinza dos pixels na imagem a ser processada. Por simplicidade, assumiremos inicialmente que a escala de cinza é normalizada, ou seja,

$$0 \leq f \leq 1,$$

onde $f = 0$ representa um pixel preto e $f = 1$ indica pixel branco.

Para qualquer f no intervalo $[0, 1]$, denominaremos transformações de intensidade as funções do tipo

$$g = T(f) \quad (3.2)$$

que mapearão cada pixel de tom de cinza f da imagem original em um novo tom de cinza, g , na imagem destino. Estas funções devem satisfazer duas condições:

- (i) devem retornar um único valor para cada valor distinto de f e devem crescer monotonicamente no intervalo $0 \leq f \leq 1$
- (ii) $0 \leq T(f) \leq 1$ para $0 \leq f \leq 1$.

Um exemplo de função que satisfaz estes critérios é dado na figura 5. O efeito desta transformação não-linear de intensidade sobre a imagem é um aumento de seu contraste.

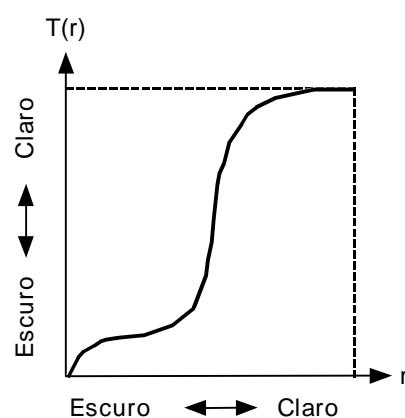


Figura 5 - Exemplo de transformação de intensidade.

As transformações de intensidade podem ser lineares ou não-lineares. As transformações lineares podem ser genericamente descritas pela equação:

$$g = c \cdot f + b \quad (3.3)$$

onde o parâmetro c controla o contraste da imagem resultante, enquanto b ajusta seu brilho. A figura 6 apresenta diversos exemplos de transformações lineares e seus respectivos valores de c e b .

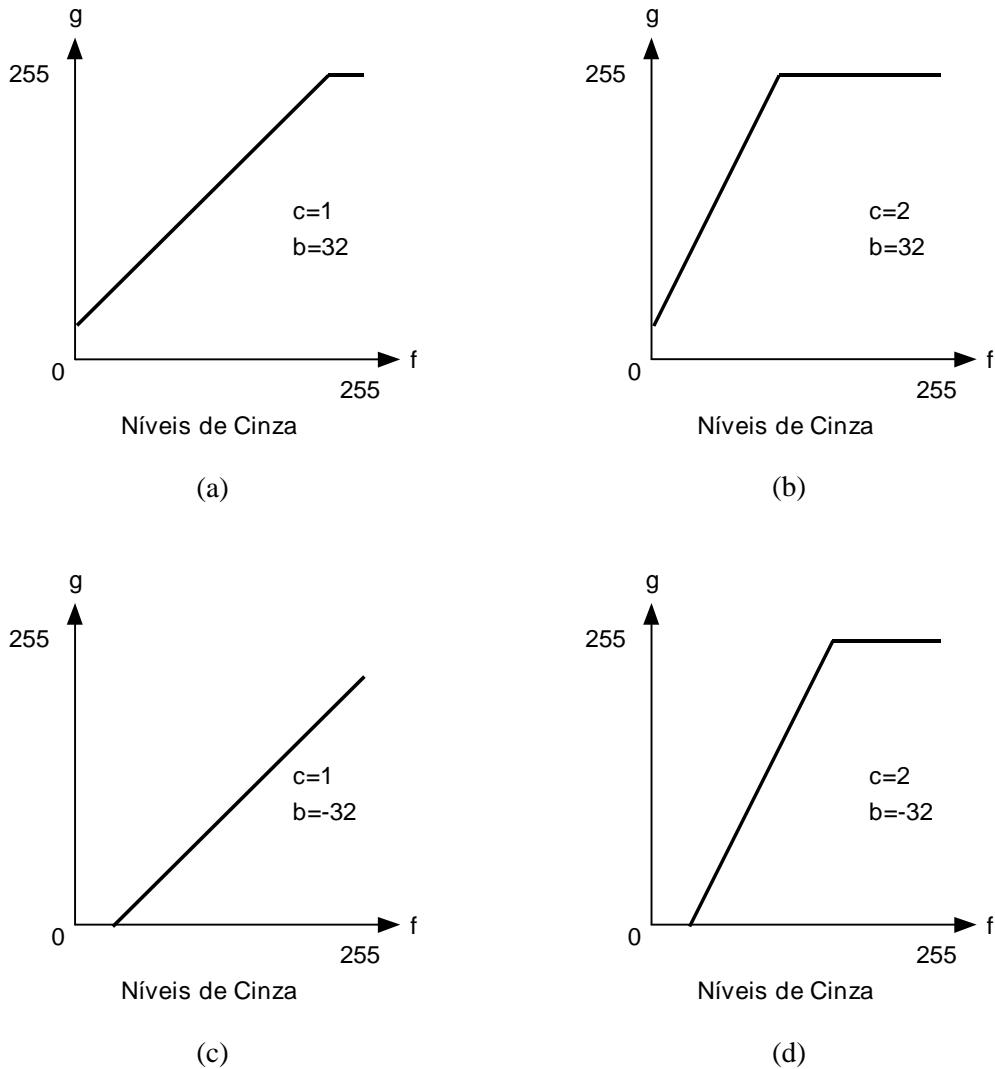


Figura 6 - Exemplos de transformações de intensidade lineares.

As transformações não-lineares podem ser descritas por equações tais como:

$$g = 31,875 \cdot \log_2(f + 1) \quad (3.4)$$

produzindo o resultado mostrado na figura 7. Nos aplicativos para processamento de imagens disponíveis atualmente, freqüentemente estas transformações são especificadas de forma interativa pelo usuário, utilizando o mouse ou dispositivo equivalente e 'desenhando' a curva desejada.

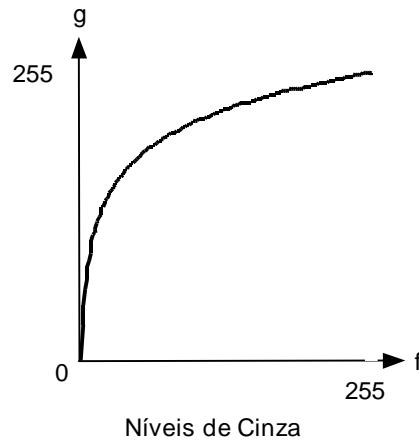


Figura 7 - Exemplo de transformação de intensidade não-linear.

O conceito de transformação de intensidade linear pode ser utilizado para implementar uma função que automaticamente expande a escala de tons de cinza de uma imagem para que ela ocupe todo o intervalo possível. Esta função recebe o nome de autoescala. Para um sistema que opera com imagens com 256 níveis de cinza, uma função de autoescala pode ser implementada calculando, para cada pixel com tom de cinza f , o nível de cinza resultante g , pela equação:

$$g = \frac{255}{f_{max} - f_{min}}(f - f_{min}), \quad (3.5)$$

onde f_{max} e f_{min} são, respectivamente, os níveis máximo e mínimo de cinza presentes na imagem original.

Leitura complementar

O capítulo 9 de [Lindley 1991] e o artigo de Dawson [Dawson 1987] apresentam código-fonte em C para cálculo de algumas transformações ponto-a-ponto discutidas nesta Seção.

O capítulo 5 de [Galbiati, Jr. 1990] contém inúmeros exemplos de funções de transformação de intensidade.

3.3 Equalização de histograma

A equalização de histograma é uma técnica a partir da qual se procura redistribuir os valores de tons de cinza dos pixels em uma imagem, de modo a obter um histograma uniforme, no qual o número (percentual) de pixels de qualquer nível de cinza é praticamente o mesmo. Para tanto, utiliza-se uma função auxiliar, denominada função de transformação. A forma mais usual de se equalizar um histograma é utilizar a função de distribuição acumulada (*cdf - cumulative distribution function*) da distribuição de probabilidades original, que pode ser expressa por:

$$s_k = T(r_k) = \sum_{j=0}^k \frac{n_j}{n} = \sum_{j=0}^k p_r(r_j) \quad (3.6)$$

onde:

$$0 \leq r_k \leq 1$$

$$k = 0, 1, \dots, L-1$$

A inversa desta função é dada por:

$$r_k = T^{-1}(s_k) \quad p/0 \leq s_k \leq 1 \quad (3.7)$$

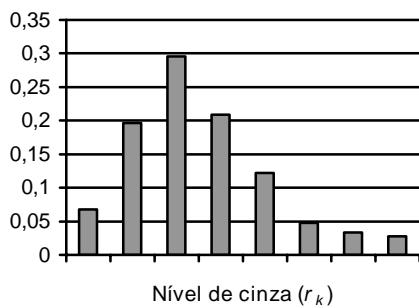
e embora ela não seja necessária no processo de equalização de histograma, será importante no método descrito na Seção seguinte. Convém notar que outras funções de transformação, que não a *cdf*, podem ser especificadas.

Exercício resolvido

Seja o histograma da tabela 1, ilustrado graficamente na figura 1, ambas reproduzidas a seguir para maior facilidade. Equalizá-lo utilizando a função de distribuição acumulada e plotar o histograma resultante.

Tabela 1 - Histograma original

Nível de cinza (r_k)	n_k	$p_r(r_k)$
0	1120	0,068
1/7	3214	0,196
2/7	4850	0,296
3/7	3425	0,209
4/7	1995	0,122
5/7	784	0,048
6/7	541	0,033
1	455	0,028
Total	16384	1



Solução:

Utilizando a *cdf* como função de transformação, calculamos:

$$\begin{aligned} s_0 &= T(r_0) = \sum_{j=0}^0 p_r(r_j) \\ &= p_r(r_0) \\ &= 0,068 \end{aligned}$$

De forma similar,

$$\begin{aligned} s_1 &= T(r_1) = \sum_{j=0}^1 p_r(r_j) \\ &= p_r(r_0) + p_r(r_1) \\ &= 0,264 \end{aligned}$$

e

$$\begin{array}{lll} s_2 = 0,560 & s_3 = 0,769 & s_4 = 0,891 \\ s_5 = 0,939 & s_6 = 0,972 & s_7 = 1. \end{array}$$

Esta função está plotada na figura 8.

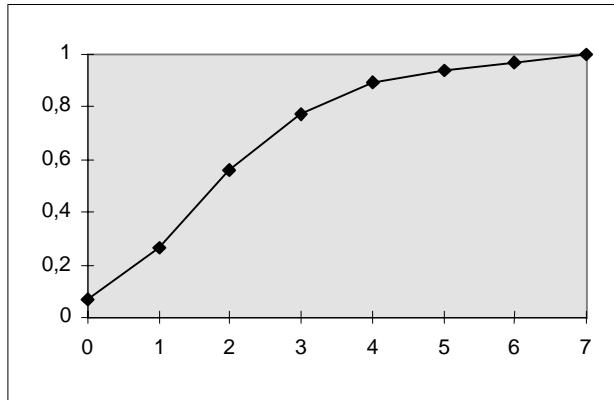


Figura 8 - Função de transformação utilizada para a equalização.

Como a imagem foi quantizada com apenas 8 níveis de cinza, cada valor s_j deverá ser arredondado para o valor válido (múltiplo de 1/7) mais próximo. Desta forma,

$$\begin{array}{llll} s_0 \approx 0 & s_1 \approx 2/7 & s_2 \approx 4/7 & s_3 \approx 5/7 \\ s_4 \approx 6/7 & s_5 \approx 1 & s_6 \approx 1 & s_7 \approx 1. \end{array}$$

Concluindo o mapeamento, verificamos que o nível original $r_0 = 0$ foi mapeado para $s_0 = 0$ e portanto a raia correspondente não sofreu alteração. Já os 3214 pixels que apresentavam tom de cinza 1/7 foram remapeados para $s_1 = 2/7$. Similarmente, os pixels com tom de cinza 2/7 foram modificados para 4/7, aqueles com $r = 3/7$ passaram a 5/7 e os de 4/7 mapearam em 6/7. Convém observar, entretanto, que as três raias correspondentes aos pixels com tons de cinza 5/7, 6/7 e 1 foram somadas em uma só raia, com tom de cinza máximo, isto é, 1.

Agrupando os resultados na tabela 2, teremos o histograma após a equalização, mostrado graficamente na figura 9.

Tabela 2 - Histograma equalizado

Nível de cinza (s_k)	n_k	$p_s(s_k)$
0	1120	0,068
1/7	0	0,000
2/7	3214	0,196
3/7	0	0,000
4/7	4850	0,296
5/7	3425	0,209
6/7	1995	0,122
1	1780	0,109
Total	16384	1

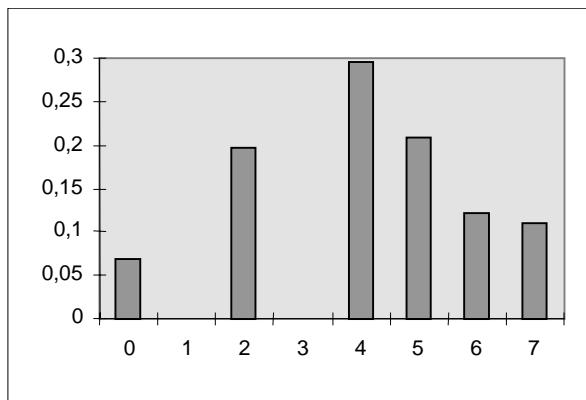


Figura 9 - Histograma equalizado.

Pode-se notar que o histograma equalizado, apesar de estar longe de ser perfeitamente plano, apresenta melhor distribuição de pixels ao longo da escala de cinza em relação ao original.

A figura 10 apresenta um exemplo de aplicação da técnica de equalização de histograma para aumentar o contraste de uma imagem 446 x 297 com 256 tons de cinza. A parte (a) apresenta a imagem original, cujo histograma é plotado na figura 10(c). A parte (d) mostra o histograma equalizado, correspondente à imagem da figura 10(b).

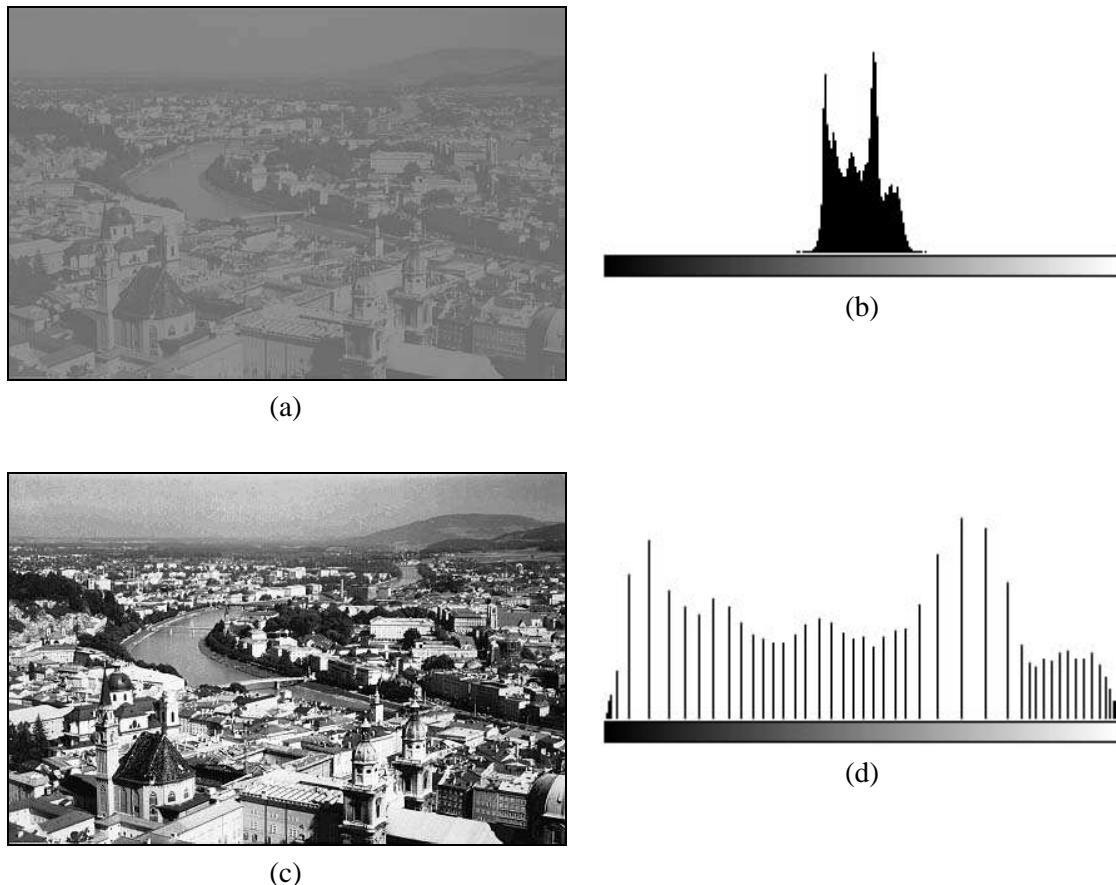


Figura 10 - Aplicação da equalização de histograma a imagens com baixo contraste.

As técnicas de obtenção e equalização de histogramas também podem ser aplicadas a trechos de imagens, por exemplo, janelas $n \times m$. Estas técnicas locais servem principalmente para realçar detalhes sutis de pequenas porções da imagem.

3.4 Especificação direta de histograma

Apesar de sua grande utilização em situações de aprimoramento de contraste de imagens, a equalização de histograma apresenta como principal limitação o fato de não permitir a especificação de nenhum parâmetro, a não ser a função de transformação, que, como vimos na Seção anterior, costuma ser a *cdf* da distribuição de probabilidade original. Existem situações, entretanto, em que seria desejável poder especificar que tipo de mudança se deseja sobre o histograma. Nestes casos, uma das possíveis técnicas é a especificação direta de histograma.

Dada uma imagem (e seu histograma original) e o novo histograma desejado, o procedimento da especificação direta de histograma consiste em:

- 1) equalizar os níveis da imagem original usando a *cdf* discreta:

$$s_k = T(r_k) = \sum_{j=0}^k \frac{n_j}{n} = \sum_{j=0}^k p_r(r_j) \quad (3.8)$$

- 2) equalizar a função densidade de probabilidade discreta (isto é, o histograma) desejada(o):

$$v_k = G(z_k) = \sum_{j=0}^k p_z(z_j) \quad (3.9)$$

- 3) aplicar a função de transformação inversa:

$$z = G^{-1}(s) \quad (3.10)$$

aos níveis obtidos no passo 1.

Exercício resolvido

Seja novamente o histograma da tabela 1. Deseja-se modificar este histograma de modo que a distribuição de pixels resultante seja aquela da tabela 3, a seguir.

Tabela 3 - Histograma desejado

Nível de cinza (z_k)	n_k	$p_z(z_k)$
0	0	0,000
1/7	0	0,000
2/7	0	0,000
3/7	1638	0,100
4/7	3277	0,200
5/7	6554	0,400
6/7	3277	0,200
1	1638	0,100
Total	16384	1

Solução:

O histograma após equalização já foi calculado em exercício resolvido anterior e seus resultados estão na tabela 2.

O próximo passo consiste em obter a *cdf* da distribuição de probabilidade desejada. Seguindo o mesmo raciocínio utilizado para o cálculo da *cdf* do histograma original, encontramos:

$$\begin{array}{llll} v_0 = 0 & v_1 = 0 & v_2 = 0 & v_3 = 0,1 \\ v_4 = 0,3 & v_5 = 0,7 & v_6 = 0,9 & v_7 = 1. \end{array}$$

O último passo – e o mais difícil de entender quando se estuda este assunto pela primeira vez – é a obtenção da inversa. Como estamos lidando com níveis discretos, a obtenção da função inversa consistirá basicamente em procurar, para cada valor de s_k , o valor de v_k que mais se aproxima dele. Por exemplo, o valor de v_k que mais se aproxima de $s_1 = 2/7 \approx 0,286$ é $G(z_4) = 0,3$ ou seja, $G^{-1}(0,3) = z_4$. Portanto, os pixels que após a equalização do histograma original foram reposicionados no tom de cinza s_1 serão mapeados para o tom de cinza z_4 . Em outras palavras, os 3214 pixels que apresentavam originalmente tom de cinza $1/7$ e que foram remapeados para $s_1 = 2/7$ devido à equalização, serão transladados novamente para $z_4 = 4/7$ por força da especificação direta de histograma. Procedendo de forma similar para os demais valores de s_k , teremos:

$$\begin{array}{ll} s_0 = 0 \rightarrow z_2 & s_1 = 2/7 \approx 0,286 \rightarrow z_4 \\ s_2 = 4/7 \approx 0,571 \rightarrow z_5 & s_3 = 5/7 \approx 0,714 \rightarrow z_5 \\ s_4 = 6/7 \approx 0,857 \rightarrow z_6 & s_5 = 1 \rightarrow z_7 \\ s_6 = 1 \rightarrow z_7 & s_7 = 1 \rightarrow z_7 \end{array}$$

Neste caso, assumimos que o algoritmo de cálculo da inversa, para um dado valor de s_k , percorreria os diversos valores de v_k , armazenando o índice do último valor que tenha resultado na menor diferença encontrada. Se o algoritmo possuir outra forma de solucionar 'empates', o nível s_0 poderá mapear em z_0 ou z_1 . A tabela 4 resume os histogramas original e desejado, suas respectivas *cdfs* e o processo de mapeamento descrito acima.

Tabela 4 - Resumo da especificação direta de histograma

k	$p_r(r_k)$	s_k	v_k	$p_z(z_k)$
0	0,068	0	0,00	0,000
1	0,196	2/7	0,00	0,000
2	0,296	4/7	0,00	0,000
3	0,209	5/7	0,10	0,100
4	0,122	6/7	0,30	0,200
5	0,048	1	0,70	0,400
6	0,033	1	0,90	0,200
7	0,028	1	1,00	0,100

A tabela 5 apresenta os valores obtidos para o histograma resultante. Para uma comparação visual entre o histograma desejado e o obtido, plotamos cada um deles nas figuras 11 e 12, respectivamente.

Tabela 5 - Histograma obtido

z_k	$p_z(z_k)$
0	0,000
1/7	0,000
2/7	0,068
3/7	0,000
4/7	0,196
5/7	0,505
6/7	0,122
1	0,109
Total	1

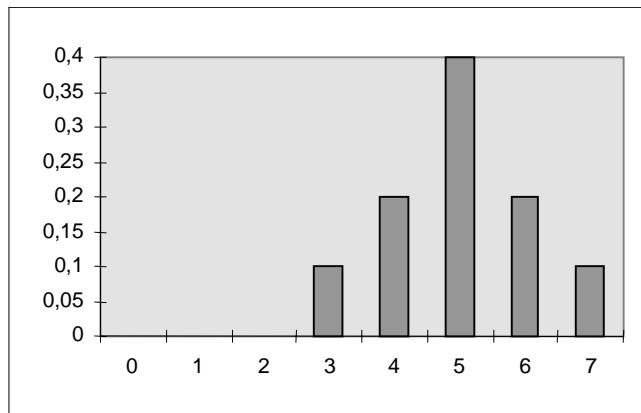


Figura 11 - Histograma desejado.

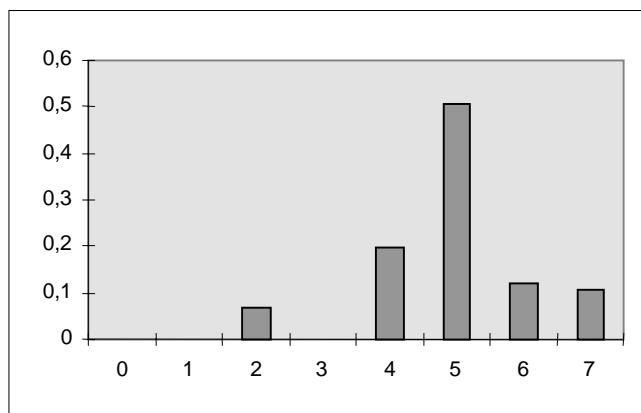


Figura 12 - Histograma obtido.

Pode-se notar que o histograma obtido aproxima-se, dentro do possível, do histograma desejado.

A figura 13 apresenta um exemplo de aplicação da técnica de especificação direta de histograma aplicada a uma imagem 443 x 298 com 256 tons de cinza. A parte (a) apresenta a imagem original, cujo histograma é plotado na figura 13(c). A parte (d) mostra o histograma desejado, enquanto a figura 13(e) mostra o histograma obtido, que corresponde à imagem da figura 13(b).

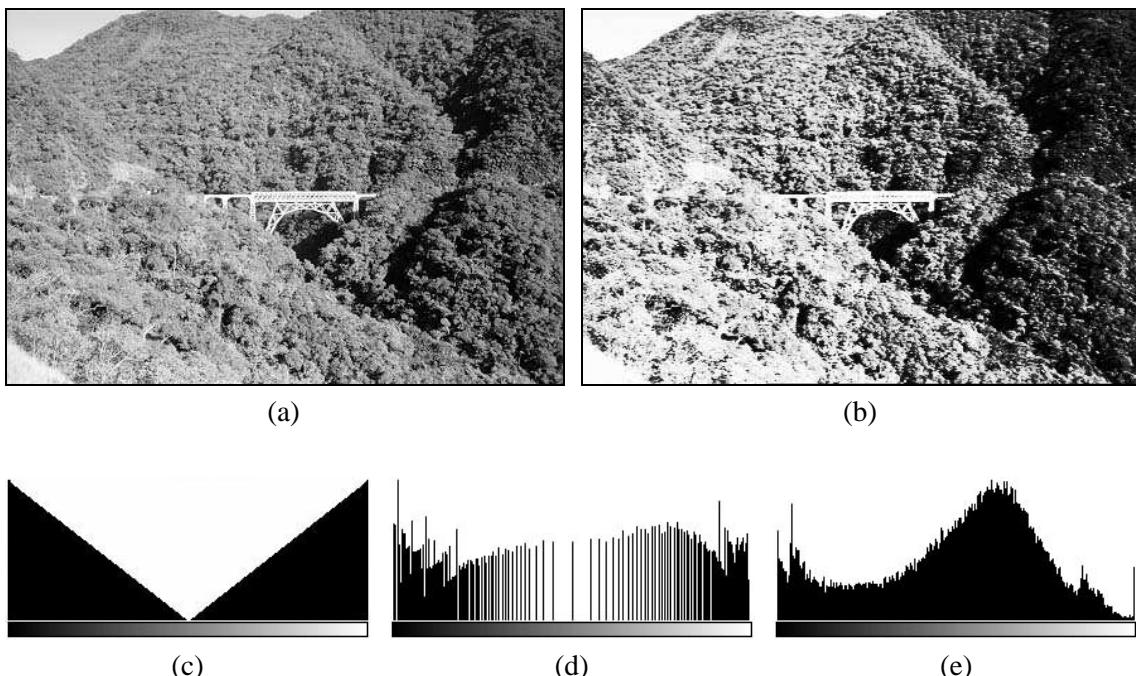


Figura 13 - Exemplo de aplicação da especificação direta de histograma.

Leitura complementar

Em [Woods e Gonzalez 1981] são descritas algumas técnicas de modificação de histograma, aplicadas em um sistema de realce de imagens em tempo real.

3.5 Outras técnicas

Existem inúmeras outras técnicas de processamento de imagens a partir de modificações de seus respectivos histogramas. Apresentamos a seguir os principais aspectos de algumas delas.

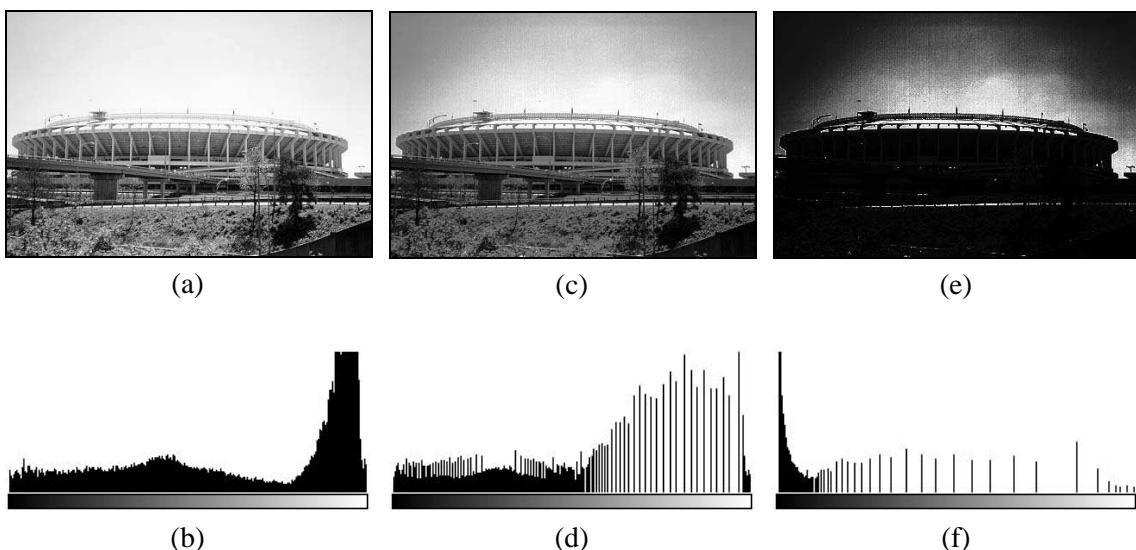


Figura 14 - Comparação entre equalização e hiperbolização de histograma: (a) imagem original, (b) histograma original, (c) imagem após equalização de histograma, (d) histograma equalizado, (e) imagem após hiperbolização, (f) histograma correspondente à imagem (e).

3.5.1 Hiperbolização

Esta técnica, proposta originalmente por Frei [Frei 1977], consiste na modificação da imagem original através de uma função de transferência tal que produza à saída uma imagem cujo histograma tenha forma hiperbólica. Ela é empregada principalmente para corrigir a equalização do histograma levando-se em conta a resposta visual do olho humano, que é considerada logarítmica. Ou seja, a equalização do histograma seria processada em duas etapas, sendo a primeira a aplicação da técnica de hiperbolização e a segunda executada pela retina. A figura 14 mostra um exemplo de aplicação desta técnica, comparando-a com a equalização de histograma.

3.5.2 Hiperbolização quadrática

Em [Cobra et al. 1992], Cobra, Costa e Menezes propõem uma nova abordagem à hiperbolização de histograma, à qual denominaram hiperbolização quadrática de histograma.

Esta técnica é baseada em um modelo do sistema visual periférico humano que leva em conta o fato de que o olho humano se acomoda à intensidade média da cena observada e não à intensidade dos pixels individuais, como subentende o modelo utilizado por [Frei 1977]. Como resultado, obtém-se uma distribuição mais espaçada dos níveis de cinza, com menor concentração na região escura do histograma.

A figura 15 ilustra o uso desta técnica, mostrando na parte (a) a imagem original, na parte (b) seu histograma, e nas figuras 15(g) e 15(h) o resultado da hiperbolização quadrática sobre a imagem e o histograma correspondente. Para efeito comparativo, também são apresentados os resultados da equalização de histograma – figuras 15(c) e 15(d) – e da hiperbolização – figuras 15(e) e 15(f).

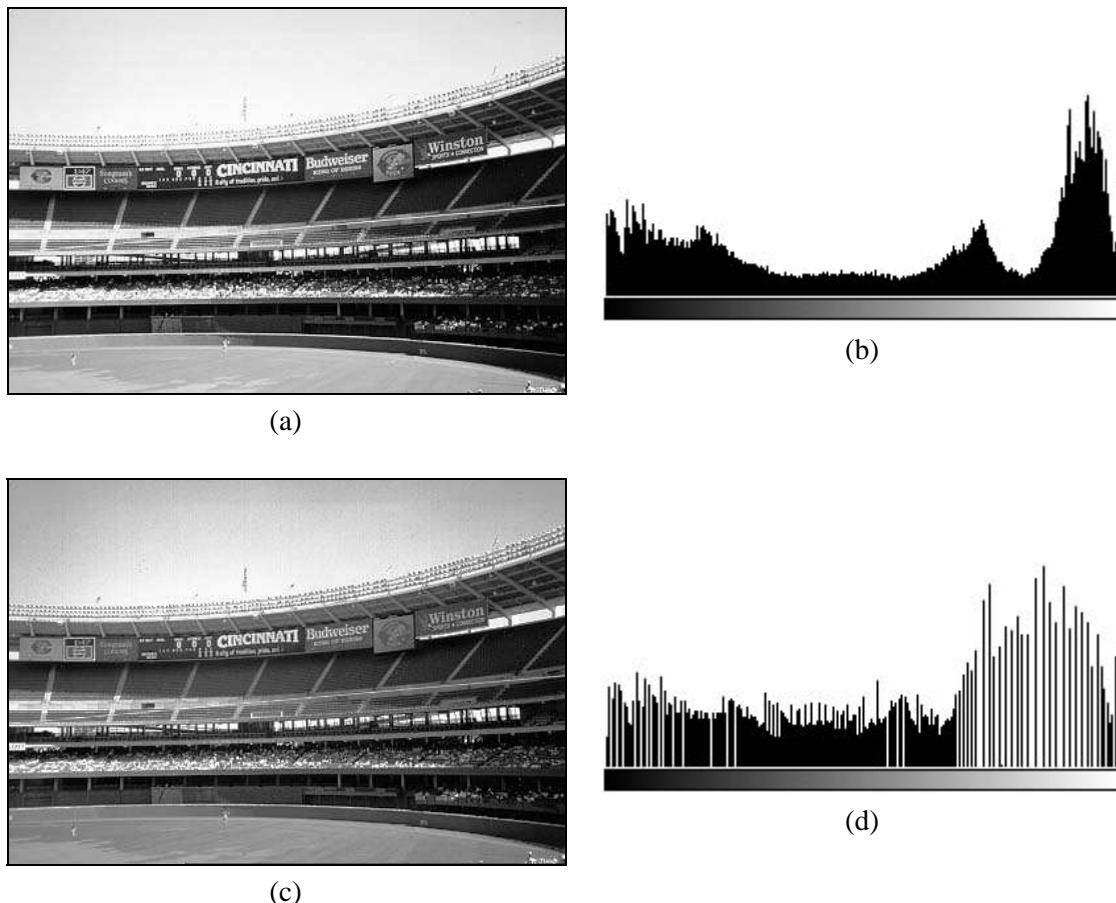


Figura 15 - Exemplo de hiperbolização quadrática de histograma.

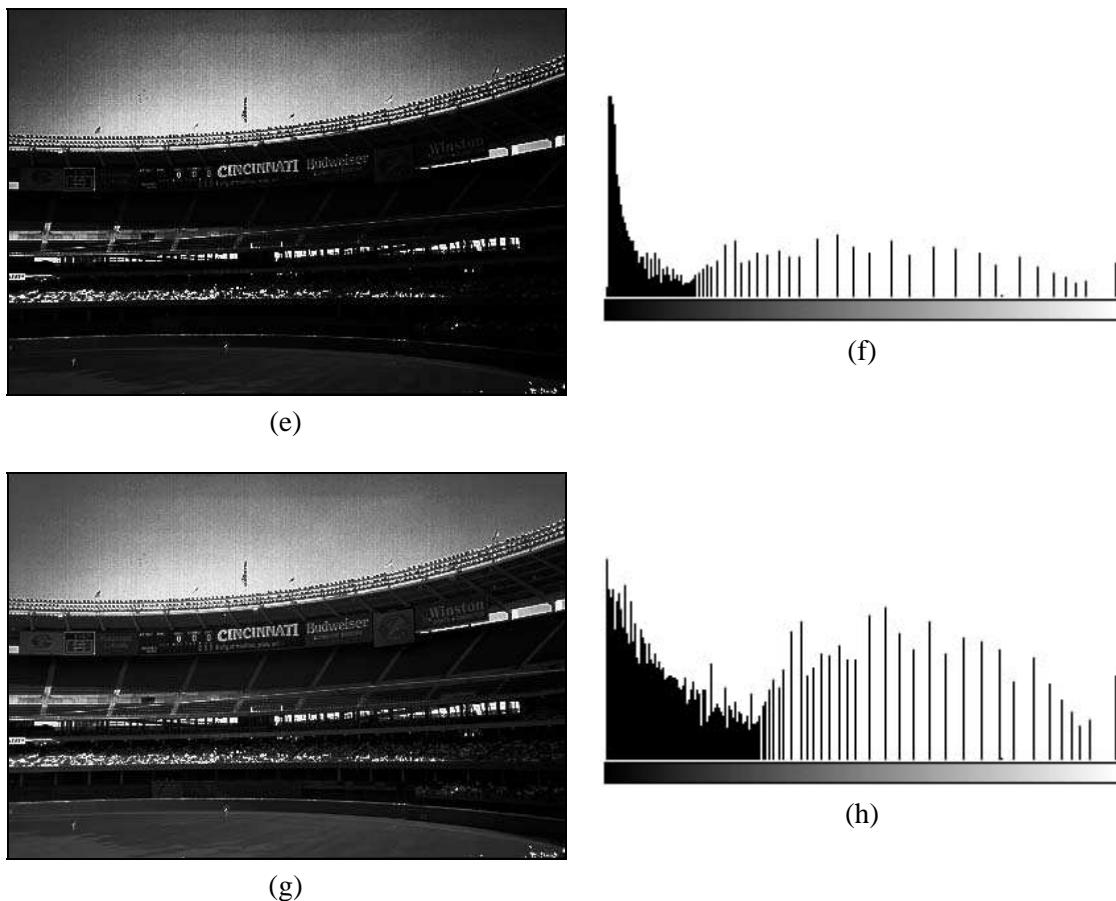


Figura 15 - Continuação.

3.5.3 Expansão de histograma (*Input cropping*)

Nesta técnica, o histograma original de uma imagem é modificado de tal forma que parte dele é expandida para ocupar toda a faixa de cinza da imagem. A figura 16 ilustra esquematicamente o processo e a figura 17 mostra um exemplo de utilização desta técnica para aprimoramento de contraste de uma imagem.

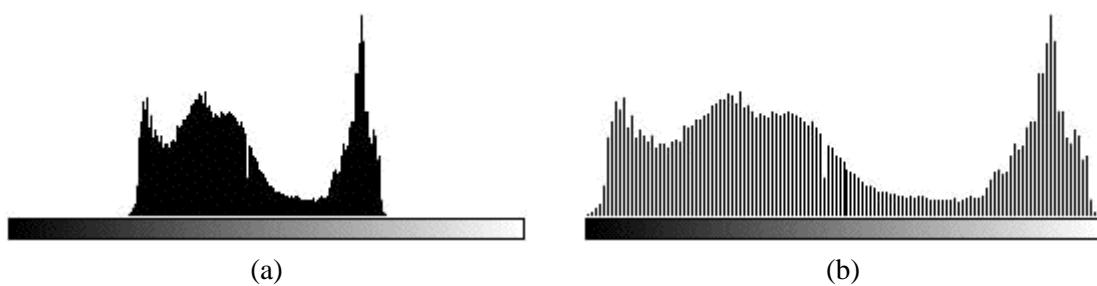


Figura 16 - Expansão de histograma.

3.5.4 Compressão de histograma (*Output cropping*)

A técnica de compressão de histograma, como o próprio nome sugere, modifica o histograma original de uma imagem de tal forma que suas raias passam a ocupar apenas um trecho da faixa total de cinza, produzindo como resultado uma redução de contraste na imagem. A figura 18 ilustra esquematicamente o processo enquanto a figura 19 mostra um exemplo de aplicação desta técnica a imagens monocromáticas.



Figura 17 - Exemplo de aplicação da técnica de expansão de histograma.

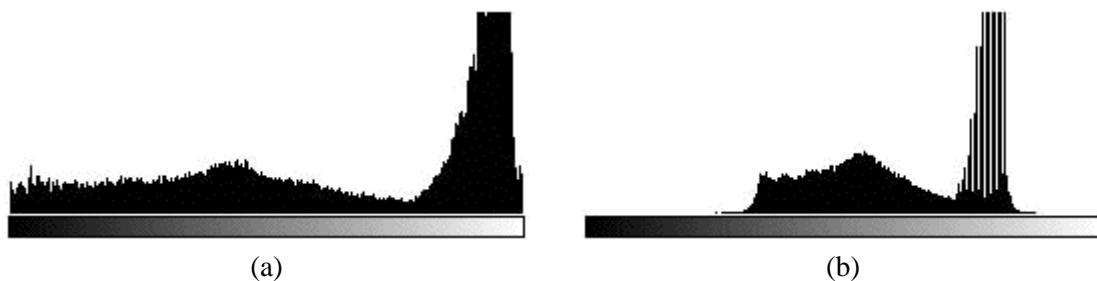


Figura 18 - Compressão de histograma.



Figura 19 - Exemplo de aplicação da técnica de compressão de histograma.

Leitura complementar

O capítulo 10 (seções 10.1 e 10.2) de [Pratt 1991] apresenta exemplos de diversas técnicas de modificação de histograma.

3.6 Limiarização (*Thresholding*)

O princípio da limiarização consiste em separar as regiões de uma imagem quando esta apresenta duas classes (o fundo e o objeto). Devido ao fato da limiarização produzir uma imagem binária à saída, o processo também é denominado, muitas vezes, binarização. A forma mais simples de limiarização consiste na bipartição do histograma, convertendo os pixels cujo tom de cinza é maior ou igual a um certo valor de limiar (T) em brancos e os demais em pretos, como ilustra a figura 20. No caso de níveis de cinza divididos basicamente em duas classes,

onde o histograma apresenta dois picos e um vale, a limiarização é trivial. Ainda assim, os efeitos decorrentes da escolha de um valor específico de limiar dentre os diversos pontos situados na região de vale podem ser analisados na figura 21.

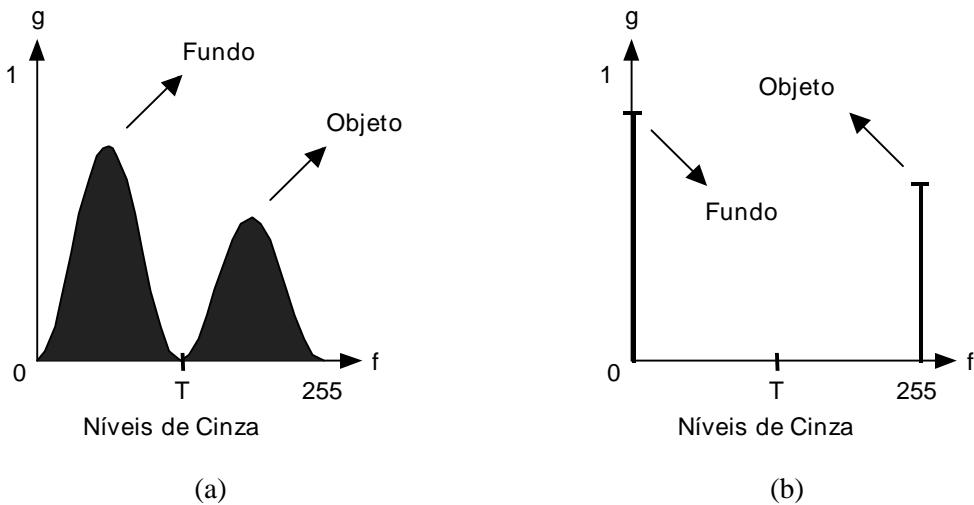


Figura 20 - Limiarização de uma imagem monocromática utilizando limiar T : (a) histograma original, (b) histograma da imagem binarizada.

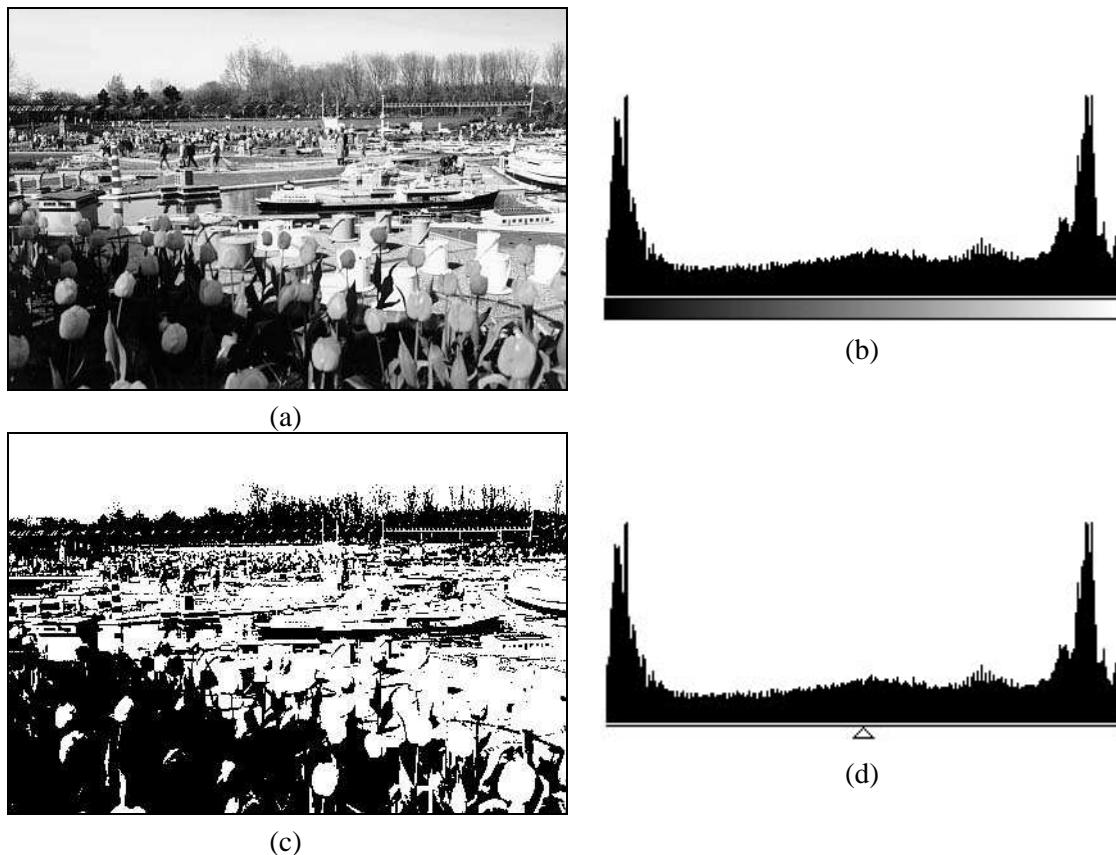


Figura 21 - Efeitos da escolha do valor de limiar na binarização de uma imagem de 256 tons de cinza. As imagens (c), (e) e (g) correspondem à bipartição dos histogramas (d), (f) e (h), respectivamente, nos limiares indicados, a saber: 128, 64 e 192.

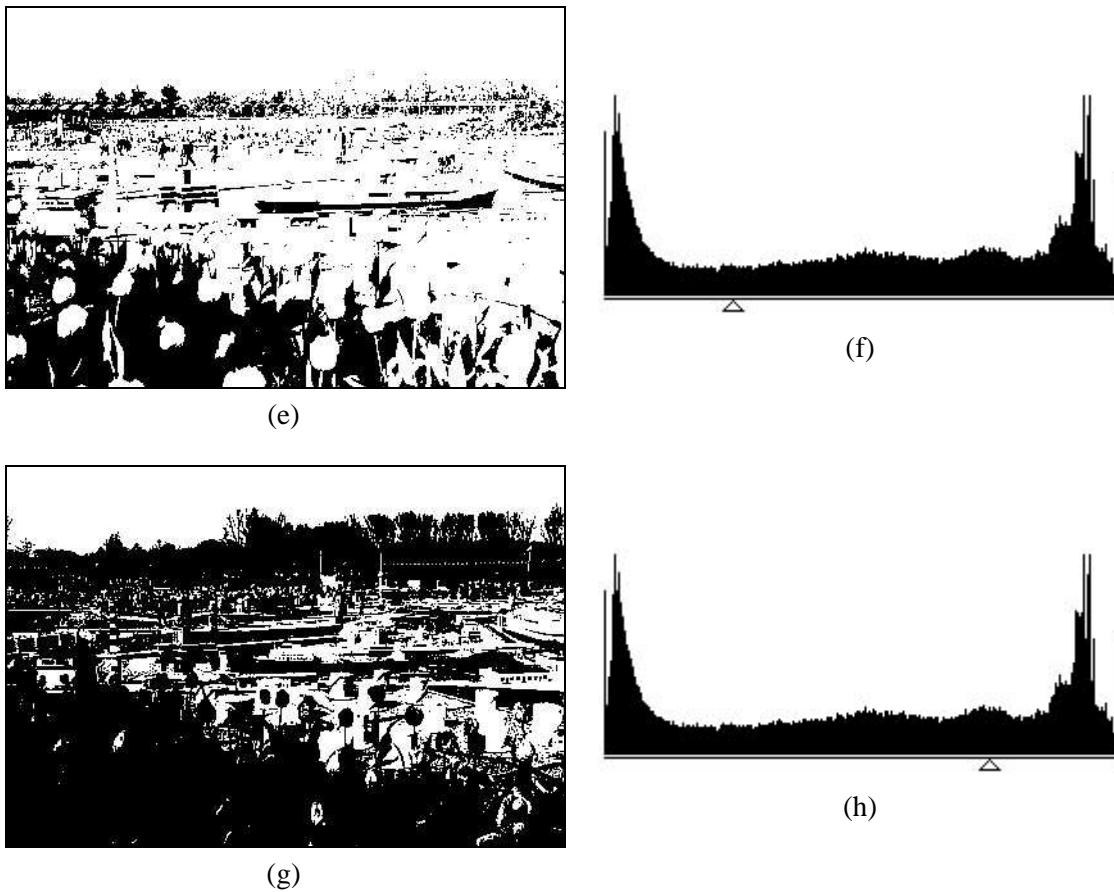


Figura 21 - Continuação.

Matematicamente, a operação de limiarização pode ser descrita como uma técnica de processamento de imagens na qual uma imagem de entrada $f(x,y)$ de N níveis de cinza produz à saída uma imagem $g(x,y)$, chamada de imagem limiarizada, cujo número de níveis de cinza é menor que N . Normalmente, $g(x,y)$ apresenta 2 níveis de cinza, sendo:

$$\begin{aligned} g(x,y) &= 1 \text{ se } f(x,y) \geq T \\ &= 0 \text{ se } f(x,y) < T \end{aligned} \quad (3.11)$$

onde os pixels rotulados com 1 correspondem aos objetos e os pixels etiquetados com 0 correspondem ao fundo (*background*) e T é um valor de tom de cinza pré-definido, ao qual denominamos limiar.

A figura 22(a) mostra um exemplo de histograma partitionado utilizando dois valores de limiar: $T_1 = 37$ e $T_2 = 233$. As figuras 22(b) e 22(c) mostram a imagem original e a imagem após a limiarização.

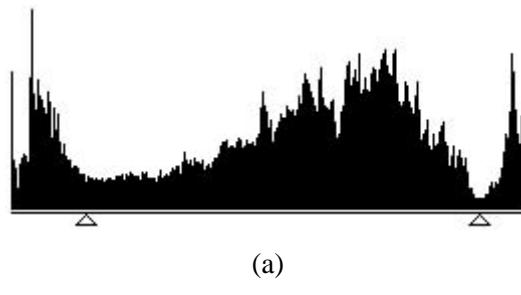


Figura 22 - Exemplo de utilização de múltiplos limiares.



Figura 22 - Continuação.

A limiarização pode ser vista como uma operação que envolve um teste com relação a uma função T do tipo $T = T[x, y, p(x,y), f(x,y)]$, onde $f(x,y)$ é o tom de cinza original no ponto (x,y) e $p(x,y)$ indica alguma propriedade local deste ponto, por exemplo a média de seus vizinhos. Quando T depende apenas de $f(x,y)$, o limiar é chamado global; quando T depende de $f(x,y)$ e de $p(x,y)$, o limiar é chamado local. Se, além disso, T depende das coordenadas espaciais de (x,y) , o limiar é chamado dinâmico ou adaptativo.

3.6.1 Influência da iluminação

A iluminação desempenha um papel significativo no processo de limiarização, uma vez que provoca alterações no histograma original da imagem, eventualmente eliminando uma região de vale entre dois picos, naturalmente propícia para a definição de um limiar global.

Pode-se provar [Papoulis 1965] que, sendo $f(x,y) = i(x,y) \cdot r(x,y)$ e sendo $z(x,y) = \ln f(x,y) = \ln i(x,y) + \ln r(x,y) = i'(x,y) + r'(x,y)$, onde $i'(x,y)$ e $r'(x,y)$ são variáveis aleatórias independentes, o histograma de $z(x,y)$ é dado pela convolução do histograma de $i'(x,y)$ com o de $r'(x,y)$.

Uma técnica comum utilizada para compensar a não uniformidade da iluminação consiste em projetar o padrão de iluminação em uma superfície refletora branca. Isto nos dá uma imagem $g(x,y) = K \cdot i(x,y)$, onde K depende da superfície utilizada. Deste modo, para qualquer imagem $f(x,y) = i(x,y) \cdot r(x,y)$ obtida com a mesma função iluminação, simplesmente divide-se $f(x,y)$ por $g(x,y)$, obtendo-se uma função normalizada:

$$h(x, y) = \frac{f(x, y)}{g(x, y)} = \frac{r(x, y)}{K} \quad (3.12)$$

Logo, se $r(x,y)$ pode ser limiarizada utilizando o limiar T , então $h(x,y)$ poderá ser segmentada usando um limiar T/K .

A figura 23 ilustra as alterações causadas por modificações no padrão de iluminação na imagem binarizada resultante. Na parte (a) é apresentada a imagem original, cujo histograma é exibido na figura 23(e). O resultado da limiarização desta imagem com limiar $T = 128$ é mostrado na figura 23(c). Na coluna da direita são mostradas a imagem com padrão de iluminação alterado (b), seu histograma (f) e o resultado da limiarização com o mesmo limiar utilizado anteriormente (d).

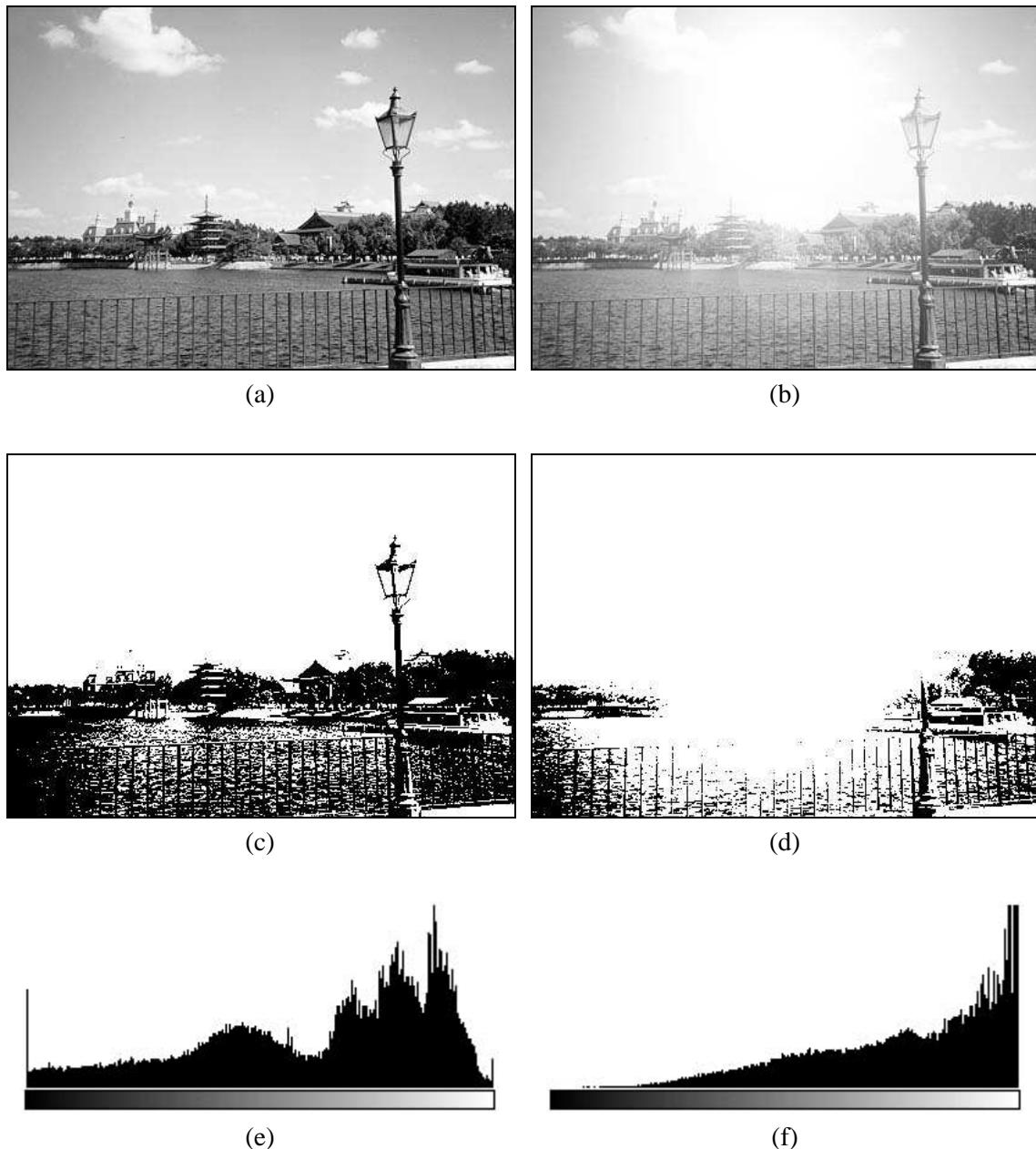


Figura 23 - Influência da iluminação no processo de limiarização.

3.6.2 Limiarização pelas propriedades estatísticas da imagem

Pelo exposto até aqui, assumiu-se que a escolha do valor de limiar é arbitrária e subjetiva. Sabendo que o histograma é uma representação gráfica da distribuição de probabilidade de ocorrência dos níveis de cinza em uma imagem, é lícito imaginar a possibilidade de uso de técnicas de cálculo do valor ótimo de limiar com base nas propriedades estatísticas da imagem.

Uma destas técnicas, denominada limiarização ótima, parte de uma imagem da qual se conhecem as principais propriedades estatísticas (supondo que sua distribuição de probabilidade é normal ou gaussiana), a saber:

μ_1 : média dos tons de cinza da região de interesse

μ_2 : média dos tons de cinza da região de fundo (background)

σ_1, σ_2 : desvios padrão

P_1, P_2 : probabilidade de ocorrência dos pixels pertencentes a esta ou aquela região.

Pode-se mostrar [Gonzalez e Woods 1992] que existe um valor ótimo de limiar, T , dado por uma das raízes da equação

$$AT^2 + BT + C = 0, \quad (3.13)$$

onde:

$$\begin{aligned} A &= \sigma_1^2 - \sigma_2^2 \\ B &= 2(\mu_1\sigma_2^2 - \mu_2\sigma_1^2) \\ C &= \mu_2^2\sigma_1^2 - \mu_1^2\sigma_2^2 + 2\sigma_1^2\sigma_2^2 \ln\left(\frac{\sigma_2 P_1}{\sigma_1 P_2}\right) \end{aligned} \quad (3.14)$$

Duas raízes reais e positivas indicam que a imagem pode requerer dois valores de limiar para obter uma solução ótima.

Se as variâncias forem iguais ($\sigma^2 = \sigma_1^2 = \sigma_2^2$), um único valor T é necessário:

$$T = \frac{\mu_1 + \mu_2}{2} + \frac{\sigma^2}{\mu_1 - \mu_2} \ln\left(\frac{P_2}{P_1}\right) \quad (3.15)$$

Se, além disso, as duas classes forem equiprováveis:

$$T = \frac{\mu_1 + \mu_2}{2} \quad (3.16)$$

o que está em acordo com o conceito intuitivo de que o valor ótimo de limiar quando as classes apresentam a mesma distribuição de probabilidade (os lóbulos são exatamente iguais) é o ponto médio entre as médias das classes.

Leitura complementar

O livro de Castleman [Castleman 1995] apresenta em detalhes outras alternativas de cálculo do valor ótimo de limiar.

Existem vários trabalhos científicos que abordam diferentes alternativas para a obtenção de uma melhor limiarização de uma imagem levando em conta seus parâmetros estatísticos. Estas técnicas pressupõem a determinação automática do melhor valor de limiar, ou seja, partem da premissa de que não haverá um operador humano que determine por tentativa e erro qual o valor de limiar mais adequado. Uma destas propostas, baseada na minimização da variância intra-grupo, encontra-se em [Otsu 1979] e está resumida em [Passariello e Mora 1995].

Em [Haralick e Shapiro 1992] encontra-se um resumo do método proposto por Kittler & Illingworth, o qual se baseia em assumir que o histograma é formado pela mistura de duas distribuições gaussianas, cujas médias e variâncias são conhecidas, no qual o objetivo é minimizar a chamada 'distância de informação de Kullback' [Passariello e Mora 1995]. De acordo com os resultados reportados em [Haralick e Shapiro 1992] com respeito a uma comparação entre o método de Otsu [Otsu 1979] e o método de Kittler-Illingworth, este último é o que produz melhores resultados.

Várias tentativas de estabelecimento de um valor adequado de limiar global (utilizando diversas técnicas de pré-processamento da imagem), sob os conceitos de precisão (*accuracy*) e reproduzibilidade, são mostradas em [Russ 1995].

Gómez-Allende [Gómez-Allende 1993] propõe um algoritmo original de limiarização baseado na busca de mínimos do histograma, no qual o histograma é submetido a uma filtragem

passa-baixas para reduzir as irregularidades causadas por objetos pouco relevantes e/ou ruído e, portanto, facilitar a deteção dos mínimos do histograma.

White e Rohrer [White e Rohrer 1983] descrevem um algoritmo de limiarização dinâmica implementado em hardware como parte de um sistema de Reconhecimento Óptico de Caracteres (OCR).

Mardia e Hainsworth [Mardia e Hainsworth 1988] propõem e comparam diversos algoritmos de limiarização espacial.

O capítulo 2 de [Haralick e Shapiro 1992], a Seção 7.3 de [Gonzalez e Woods 1992] e a Seção 5.1 de [Sonka et al. 1993] são dedicados ao problema da limiarização.

Exercícios Propostos

1. Que efeito uma transformação de intensidade $s = r^\gamma$ provocará em uma imagem monocromática com valores de níveis de cinza originais (r) normalizados na faixa de 0 a 1, caso $\gamma > 1$?

2. Assinalar V ou F conforme as proposições a seguir sejam verdadeiras ou falsas.

() A técnica de equalização de histograma aplicada a imagens digitais nunca produz à saída um histograma perfeitamente plano.

() Após um histograma ter sido equalizado, uma nova aplicação da técnica de equalização de histograma sobre a imagem não produzirá nela nenhuma alteração.

() A técnica de hiperbolização de histograma tem como principal vantagem sobre a equalização de histograma o fato de que a primeira leva em consideração as características não-lineares da curva de intensidade luminosa subjetiva versus intensidade luminosa física da imagem do olho humano.

() Ao recortar uma imagem em dois pontos quaisquer, o histograma da subcena resultante sempre será idêntico ao da imagem original, porque as raias verticais indicam a concentração de pixels em termos percentuais e, portanto, não dependem do número total de pixels da imagem.

3. Considere a imagem a seguir, representada por uma matriz 7×7 , onde cada elemento da matriz corresponde ao nível de cinza normalizado do pixel correspondente, sendo 0 = preto, 1 = branco.

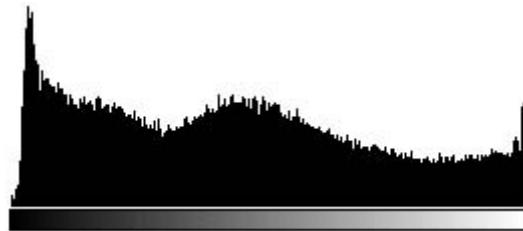
0	3/7	2/7	2/7	1/7	1/7	4/7
3/7	2/7	1/7	1/7	1/7	1/7	4/7
2/7	0	1	1/7	3/7	0	0
0	5/7	1/7	0	6/7	0	1/7
1/7	1/7	1/7	3/7	6/7	6/7	5/7
1/7	1/7	1/7	1/7	5/7	6/7	4/7
0	1	0	0	0	0	4/7

Pede-se:

- Calcular as probabilidades de cada nível de cinza e plotar seu histograma.
- Na imagem original predominam pixels claros ou escuros?

- c) Equalizar o histograma calculado utilizando a função de distribuição de probabilidade acumulada, informando os novos valores e plotando o histograma resultante (equalizado).

4. Dado o histograma a seguir, responder:



- a) a imagem correspondente ao histograma pode ser considerada de bom contraste? Por que?
 b) o histograma pode ser considerado equalizado? Por que?
 c) a imagem apresenta predominância de pixels claros ou escuros? Justificar.
 d) pelo histograma, é possível afirmar ou supor que a imagem apresenta ruído? Explicar.

5. Supor que uma imagem 256 x 256 com 8 níveis de cinza possui uma distribuição de tons de cinza dada pela tabela a seguir.

r_k	n_k	$p_r(r_k) = n_k/n$
$r_0 = 0$	2621	0,04
$r_1 = 1/7$	0	0,00
$r_2 = 2/7$	0	0,00
$r_3 = 3/7$	5243	0,08
$r_4 = 4/7$	7209	0,11
$r_5 = 5/7$	12452	0,19
$r_6 = 6/7$	24904	0,38
$r_7 = 1$	13107	0,20

Deseja-se transformar o histograma desta imagem aproximando-o do histograma correspondente à tabela a seguir.

z_k	$p_z(z_k)$
0	0,27
1/7	0,16
2/7	0,19
3/7	0,16
4/7	0,11
5/7	0,06
6/7	0,03
1	0,02

Pede-se:

- a) na imagem original predominam pixels claros ou escuros? Justifique.
 b) caso a modificação de histograma seja bem sucedida, qual o provável efeito desta modificação na imagem original?

- c) equalizar o histograma original, através da função $s = T(r)$;
- d) obter a função $v = G(z)$ e sua inversa;
- e) plotar o histograma original, o desejado, o equalizado e o histograma obtido ao final do processo;
- f) preencher a tabela abaixo com os valores finais de n_k e $p_z(z_k)$ para os 8 valores de z_k , comparar com os valores desejados e justificar as diferenças eventuais.

z_k	n_k	$p_z(z_k)$
0		
1/7		
2/7		
3/7		
4/7		
5/7		
6/7		
1		

No computador

Sugerimos o roteiro da prática nº 4 (Apêndice B) para complementar os aspectos teóricos abordados neste capítulo.

Na Internet

Dentre as diversas referências disponíveis na WWW correlatas a este capítulo, destacamos:

"http://www.eecs.wsu.edu/IPdb/Enhancement/hist_equalization.html"
Histogram Equalization

"http://www.eecs.wsu.edu/IPdb/Enhancement/hist_stretching.html"
Histogram Stretching

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c4/s2/front-page.html>"
Thresholding

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c4/s6/front-page.html>"
Logarithm Contrast Enhancement

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c4/s8/front-page.html>"
Exponential Contrast Enhancement

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c1/s3/front-page.html>"
Image Statistics

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c4/s4/front-page.html>"

Histogram Stretching Contrast Enhancement

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c4/s5/front-page.html>"

Histogram Equalization

Bibliografia

- [Castleman 1995] Castleman, K. R., *Digital Image Processing*, Prentice-Hall, 1995.
- [Cobra et al. 1992] Cobra, D.T.Q., Costa, J.A.D.D. e Menezes, M.F.B., "Realce de Imagens Através de Hiperbolização Quadrática do Histograma", *Anais do V SIBGRAPI*, Novembro 1992, 63-71.
- [Dawson 1987] Dawson, B.M., "Introduction to Image Processing Algorithms", *Byte*, Março 1987, 169-186.
- [Frei 1977] Frei, W., "Image Enhancement by Histogram Hyperbolization" *Computer Graphics and Image Processing* 6, 3, Junho 1977, 86-294.
- [Galbiati, Jr. 1990] Galbiati, Jr., L.J., *Machine Vision and Digital Image Processing Fundamentals*, Prentice-Hall, 1990.
- [Gómez-Allende 1993] Gómez-Allende, D. M., *Reconocimiento de Formas y Visión Artificial*, RA-MA Editorial, 1993.
- [Gonzalez e Woods 1992] Gonzalez, R.C. e Woods, R.E., *Digital Image Processing - Third Edition*, Addison-Wesley, 1992.
- [Haralick e Shapiro 1992] Haralick, R.M. e Shapiro, L.G., *Computer and Robot Vision - Volume 1*, Addison-Wesley, 1992.
- [Lindley 1991] Lindley, C.A., *Practical Image Processing in C*, Wiley, 1991.
- [Mardia e Hainsworth 1988] Mardia, K.V. e Hainsworth, T.J., "A Spatial Thresholding Method for Image Segmentation", *IEEE Trans. Pattern Analysis and Machine Intelligence*, 10, 6, Novembro 1988, 919-927.
- [Otsu 1979] Otsu, N., "A Threshold Selection Method from Grey-Level Histograms", *IEEE Transactions on Systems, Man and Cybernetics*, 9, 1, Janeiro 1979, 62-66.
- [Papoulis 1965] Papoulis, A., *Probability, Random Variables and Stochastic Processes*, McGraw-Hill, 1965.
- [Passariello e Mora 1995] Passariello, G. e Mora, F. (eds.), *Imágenes Médicas*, EQUINOCCIO - Ediciones de la Universidad Simón Bolívar, 1995.
- [Pavlidis 1982] Pavlidis, T., *Algorithms for Graphics and Image Processing*, Computer Science Press, 1982.

- [Pratt 1991] Pratt, W. K., *Digital Image Processing*, Wiley Interscience, 1991. (2nd ed.)
- [Ross 1994] Ross, S., *A First Course in Probability - 4th edition*, Macmillan, 1994.
- [Russ 1995] Russ, J. C., *The Image Processing Handbook - 2nd ed.*, CRC Press, 1995.
- [Sonka et al. 1993] Sonka, M., Hlavac, V. e Boyle, R., *Image Processing, Analysis and Machine Vision*, Chapman & Hall, 1993.
- [White e Rohrer 1983] White, J.M. e Rohrer, G.D., "Image Thresholding for Optical Character Recognition and Other Applications Requiring Character Image Extraction", *IBM J. Res. Develop.*, 27, 4, Julho 1983, 400-411.
- [Woods e Gonzalez 1981] Woods, R.E. e Gonzalez, R.C., "Real-Time Digital Image Enhancement", *Proceedings of the IEEE*, 69, 5, Maio 1981, 643-654.

Capítulo 4

Filtragem, Realce e Suavização de Imagens

O principal objetivo das técnicas de realce de imagens é processar uma certa imagem de modo que a imagem resultante seja mais adequada que a imagem original para uma aplicação específica. Desta afirmativa decorrem duas importantes conclusões:

1. A interpretação de que o resultado é mais adequado, ou não, normalmente é subjetiva e depende de conhecimento prévio do observador a respeito das imagens analisadas.
2. As técnicas de realce de imagens a serem estudadas neste capítulo são por natureza orientadas a um problema que se deseja resolver. Logo, não existem técnicas capazes de resolver 100% dos problemas que uma imagem digital possa apresentar, como também nem sempre uma técnica que produz bons resultados para imagens biomédicas adquiridas através de um tomógrafo computadorizado apresentará desempenho satisfatório se aplicada a uma imagem contendo uma impressão digital, por exemplo.

Os métodos de filtragem de imagens discutidos neste capítulo são normalmente classificados em duas categorias: as técnicas de filtragem espacial e as técnicas de filtragem no domínio da freqüência. Os métodos que trabalham no domínio espacial operam diretamente sobre a matriz de pixels que é a imagem digitalizada, normalmente utilizando operações de convolução com máscaras (Seção 2.3). Os métodos que atuam no domínio da freqüência se baseiam na modificação da transformada de Fourier (Seção 4.4) da imagem. Existem técnicas de filtragem que combinam ambas as abordagens.

A Seção 4.1 apresenta algumas considerações iniciais sobre filtragem de imagens no domínio espacial e no domínio freqüencial. As técnicas de suavização de imagens no domínio espacial utilizando operações orientadas a vizinhança são apresentadas na Seção 4.2. A Seção 4.3 trata das técnicas de realce (agudização) de imagens no domínio espacial. Na Seção 4.4 apresentamos a transformada de Fourier discreta bidimensional, ferramenta matemática indispensável para o projeto de filtros no domínio da freqüência, como os apresentados na Seção 4.5. A Seção 4.6 apresenta alguns fundamentos de imagens coloridas, bem como os conceitos de pseudocolorização e técnicas de processamento de imagens coloridas. Finalmente, a Seção 4.7 apresenta algumas técnicas de filtragem adaptativa.

4.1 Considerações iniciais

As técnicas de filtragem, realce e suavização apresentadas neste capítulo podem ser divididas em: técnicas no domínio espacial e técnicas no domínio freqüencial. O objetivo desta seção é destacar os princípios de funcionamento de cada uma destas abordagens.

4.1.1 Filtragem no domínio espacial

As técnicas de filtragem no domínio espacial são aquelas que atuam diretamente sobre a matriz de pixels que é a imagem digitalizada. Logo, as funções de processamento de imagens no domínio espacial podem ser expressas como:

$$g(x,y) = T [f(x,y)] \quad (4.1)$$

onde: $g(x,y)$ é a imagem processada, $f(x,y)$ é a imagem original e T é um operador em f , definido em uma certa vizinhança de (x,y) . Além disso, o operador T pode também operar sobre um conjunto de imagens de entrada, como será visto na Seção 4.2.4.

A vizinhança normalmente definida ao redor de (x,y) é a 8-vizinhança do pixel de referência, o que equivale a uma região 3×3 na qual o pixel central é o de referência, como indica a figura 1. O centro dessa região ou subimagem é movido pixel a pixel, iniciando no canto superior esquerdo da figura e aplicando a cada localidade o operador T para calcular o valor de g naquele ponto.

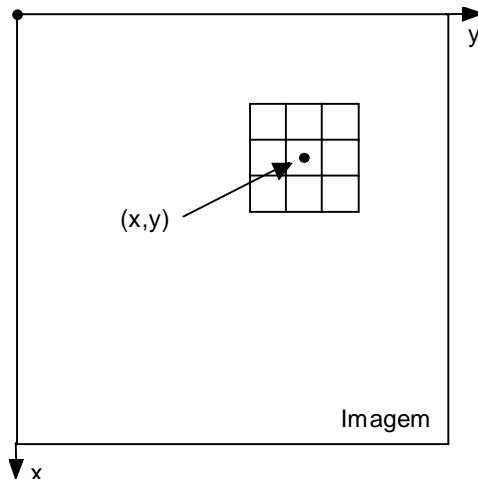


Figura 1 - Uma vizinhança 3×3 ao redor de um ponto de coordenadas (x,y) em uma imagem.

Nos casos em que a vizinhança é 1×1 , o operador T torna-se uma função de transformação (ou de mapeamento), do tipo:

$$s = T(r) \quad (4.2)$$

onde: r é o nível de cinza de $f(x,y)$ e s é o nível de cinza de $g(x,y)$ em um certo ponto. As técnicas de processamento de imagem pertencentes a este caso são freqüentemente denominadas técnicas ponto-a-ponto e já foram abordadas na Seção 3.2.

4.1.2 Filtragem no domínio da freqüência

A base matemática das técnicas de filtragem no domínio da freqüência é o teorema da convolução. Seja $g(x,y)$ a imagem formada pela convolução (denotada pelo símbolo $*$) da imagem $f(x,y)$ com um operador linear $h(x,y)$, ou seja,

$$g(x,y) = f(x,y) * h(x,y) \quad (4.3)$$

Então, pelo teorema da convolução (Seção 4.4), a seguinte relação no domínio da freqüência também é válida:

$$G(u,v) = F(u,v)H(u,v) \quad (4.4)$$

onde G , F e H são as transformadas de Fourier de g , f e h , respectivamente. Na terminologia de sistemas lineares, a transformada $H(u,v)$ é denominada função de transferência do filtro.

Inúmeros problemas de processamento de imagens podem ser expressos na forma da equação (4.4). Em uma aplicação de suavização de imagens, por exemplo, dada $f(x,y)$, o objetivo, após calcular $F(u,v)$, é selecionar $H(u,v)$ de tal maneira que a imagem desejada,

$$g(x,y) = \mathfrak{J}^{-1}[F(u,v)H(u,v)] \quad (4.5)$$

remova componentes de alta freqüência (possivelmente ruidosos) de $f(x,y)$. Isto poderia ser obtido usando um filtro Butterworth passa-baixas, por exemplo.

A equação (4.3) descreve um processo espacial análogo ao explicado na Seção 4.1.1 e por esta razão $h(x,y)$ é freqüentemente denominada máscara de convolução espacial. A conversão de filtros projetados no domínio da freqüência para o domínio espacial e vice-versa é matematicamente possível mas seu detalhamento foge ao escopo deste livro.

Leitura complementar

A Seção 4.5 de [Gonzalez e Woods 1992] detalha matematicamente o processo de geração de máscaras de convolução correspondentes a um filtro especificado no domínio da freqüência.

4.2 Suavização de imagens no domínio espacial

4.2.1 Introdução

O uso de máscaras espaciais no processamento de imagens é normalmente denominado filtragem espacial (em contraste com a expressão 'filtragem no domínio da freqüência', utilizada quando se opera sobre a transformada de Fourier da imagem original) e as máscaras são conhecidas como filtros espaciais. Nesta seção consideraremos filtros lineares e não-lineares aplicados ao processamento de imagens.

Os filtros lineares se baseiam no conceito de que a função de transferência de um sistema linear ($H(u,v)$) e sua função de resposta a impulso unitário ($h(x,y)$) estão relacionadas entre si através da transformada de Fourier, como ilustra a figura 2.

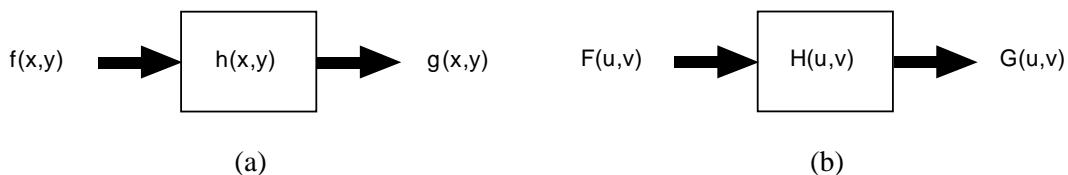


Figura 2 - Fundamentos de sistemas lineares. Na parte (a) (domínio espacial), a saída do sistema é obtida através da convolução de sua entrada com sua função de resposta a impulso unitário ($h(x,y)$). Em (b) (domínio da freqüência), a saída do sistema é o produto de sua função de transferência ($H(u,v)$) pela entrada.

Os filtros são denominados 'passa-baixas' quando atenuam ou eliminam as componentes de alta freqüência no domínio das transformadas de Fourier. Como as componentes de alta freqüência correspondem a regiões de bordas e/ou detalhes finos na imagem, o efeito da filtragem passa-baixas é a suavização da imagem, provocando um leve borramento na mesma. Já os filtros passa-altas atenuam ou eliminam os componentes de baixa freqüência e, em função disto, realçam as bordas e regiões de alto contraste da imagem. Os filtros passa-faixa, capazes de remover ou atenuar componentes acima de sua freqüência de corte superior e abaixo de sua freqüência de corte inferior, embora existam, são de pouca utilidade prática, com exceção de algumas tarefas específicas de restauração de imagens.

A figura 3 mostra as respostas em freqüência dos três principais tipos de filtros existentes e os respectivos filtros espaciais correspondentes.

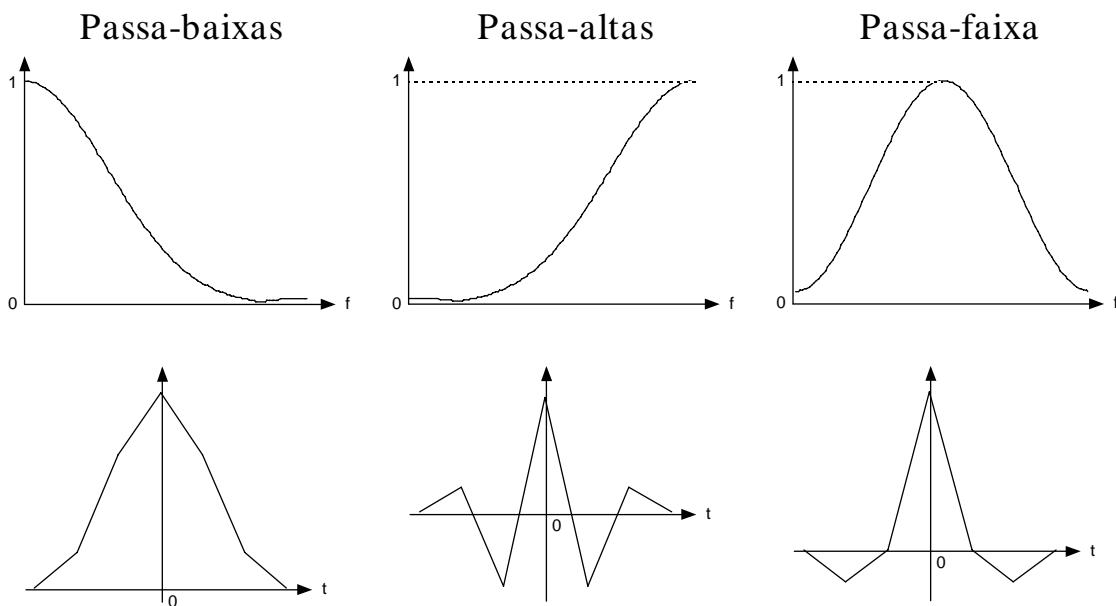


Figura 3 - (Acima) Resposta em freqüência dos principais tipos de filtros. (Abaixo) Filtros correspondentes no domínio espacial.

A suavização de imagens no domínio espacial baseia-se no uso de máscaras de convolução (ver Seção 2.4) adequadas para o objetivo em questão, normalmente o borramento da imagem (para eliminar detalhes que não são de interesse para as etapas subsequentes do processamento) ou a remoção de ruídos nela presentes. Dentre as técnicas mais conhecidas de suavização estão a filtragem pela média e o filtro da mediana, que serão detalhadas a seguir.

4.2.2 Filtro da média

Como se pode perceber na figura 3(a), a resposta ao impulso de um filtro passa-baixas indica que ele deve apresentar todos seus coeficientes positivos. A forma mais simples de implementar um filtro com tais características é construir uma máscara 3×3 com todos seus coeficientes iguais a 1, dividindo o resultado da convolução por um fator de normalização, neste caso igual a 9. Um filtro com esta característica é denominado filtro da média. A figura 4(a) mostra a máscara resultante, enquanto as figuras 4(b) e 4(c) ilustram o mesmo conceito, aplicado a máscaras de maiores dimensões. Na escolha do tamanho da máscara deve-se ter em mente que quanto maior a máscara, maior o grau de borramento da imagem resultante. A figura 5 mostra exemplos de máscaras de filtragem pela média de diferentes dimensões aplicadas a uma mesma imagem. As figuras 6 e 7 mostram exemplos de aplicação do filtro da média para remoção de ruídos em imagens monocromáticas.

$$\begin{array}{c}
 \text{(a)} \\
 \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \\
 \\
 \text{(b)} \\
 \frac{1}{25} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \\
 \\
 \text{(c)} \\
 \frac{1}{49} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}
 \end{array}$$

Figura 4 - Máscaras para cálculo do filtro da média: (a) 3×3 ; (b) 5×5 ; (c) 7×7 .

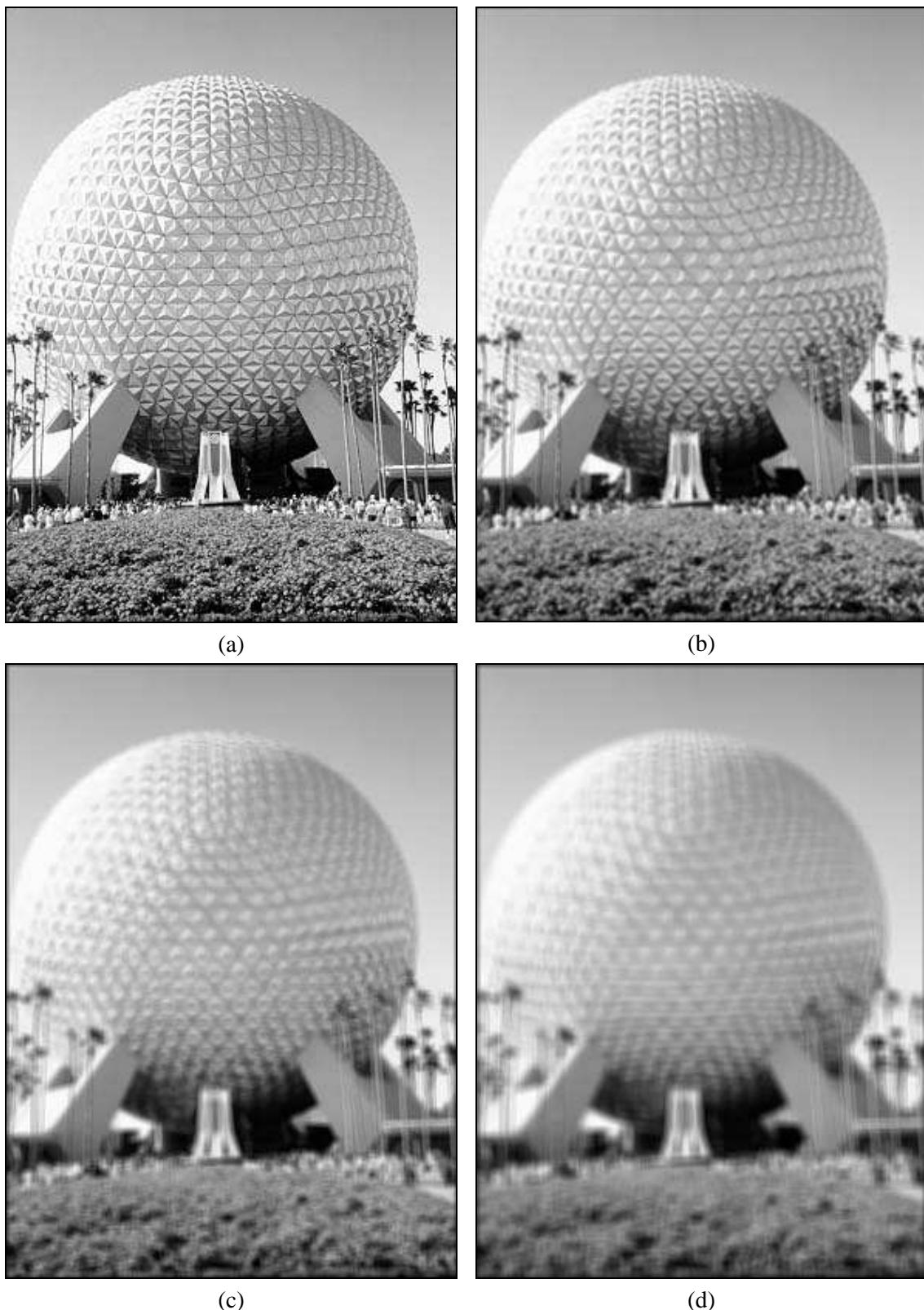
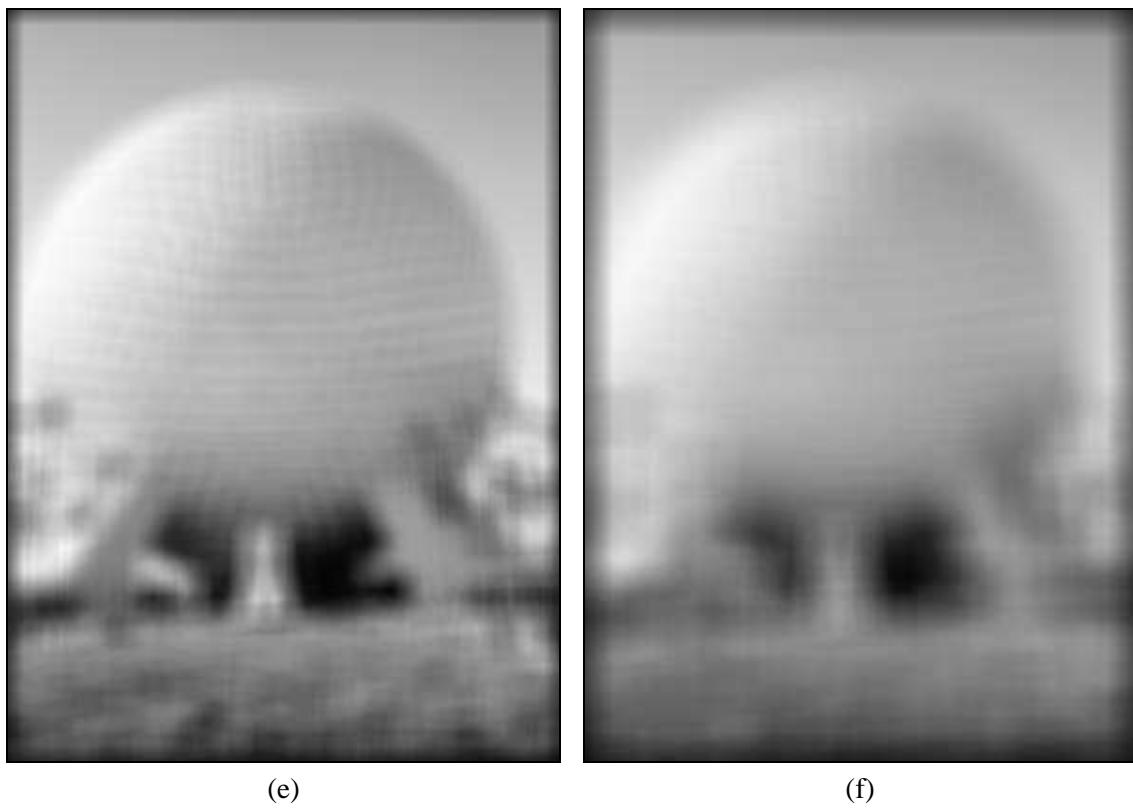


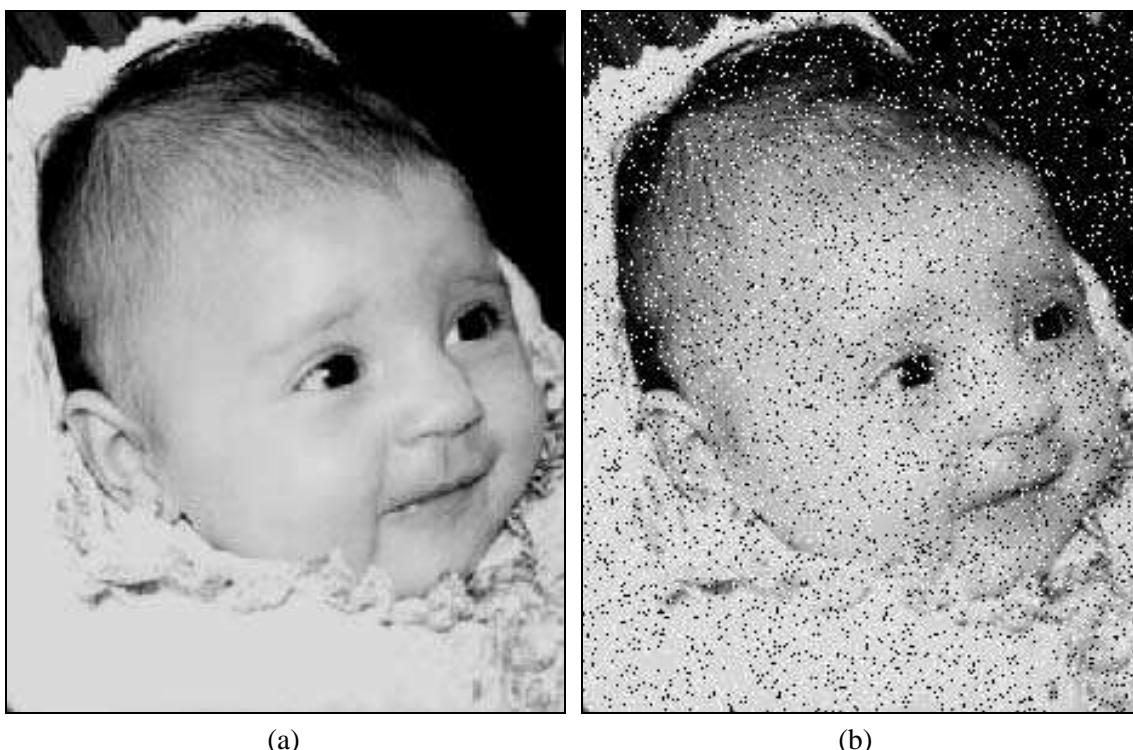
Figura 5 - (a) Imagem original; (b)-(f) resultados da aplicação do filtro da média com máscara de dimensões $n \times n$, $n = 3, 5, 7, 17, 31$.



(e)

(f)

Figura 5 – Continuação.



(a)

(b)

Figura 6 - (a) Imagem original; (b) imagem contaminada por ruído impulsivo (sal e pimenta); (c) resultado da filtragem pelo filtro da média com máscara 3×3 ; (d) resultado da filtragem pelo filtro da média com máscara 5×5 .

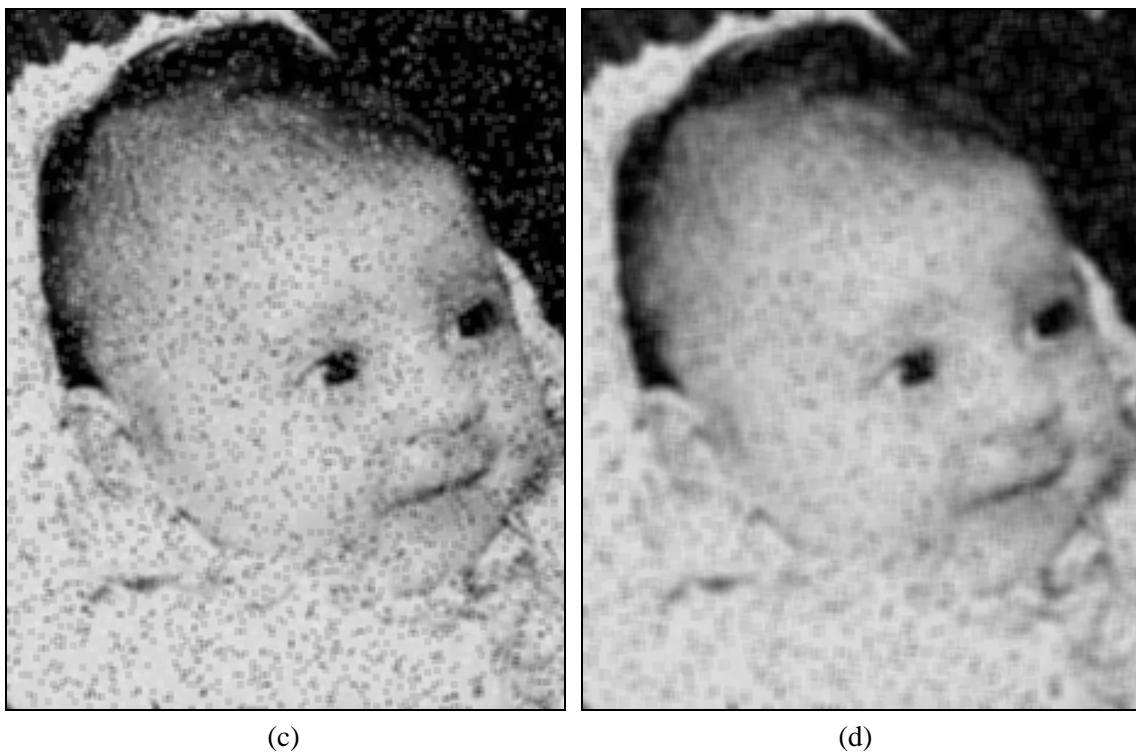


Figura 6 – Continuação

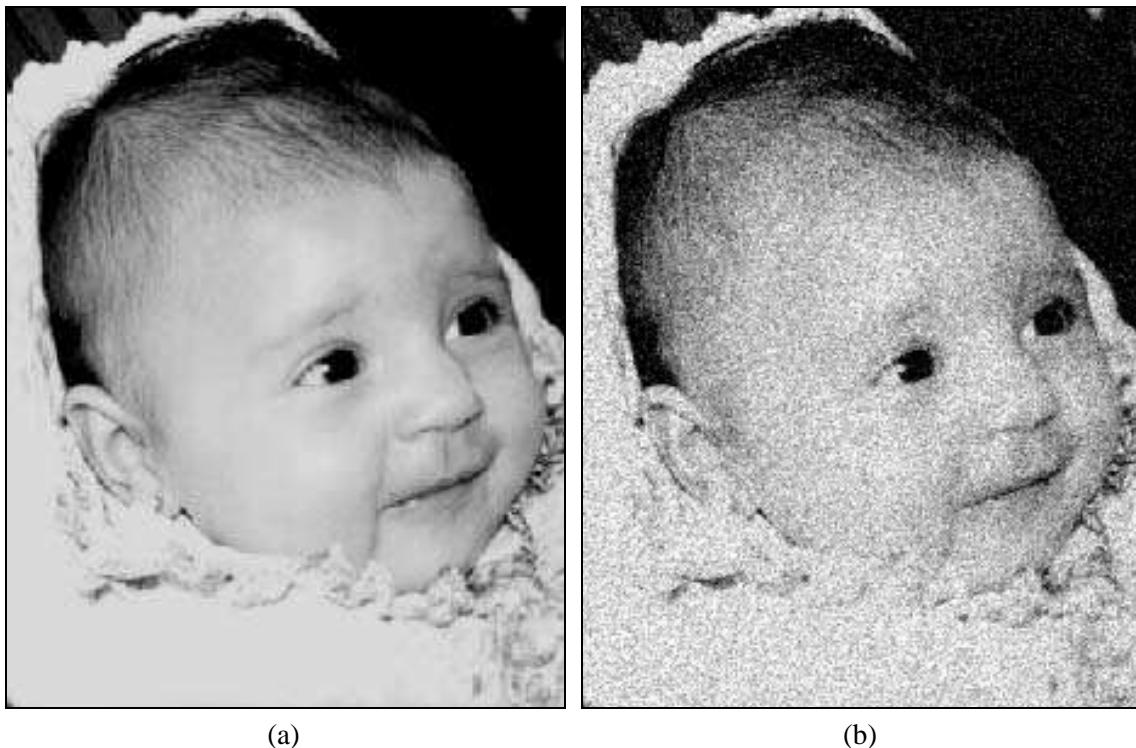


Figura 7 - (a) Imagem original; (b) imagem contaminada por ruído gaussiano; (c) resultado da filtragem pelo filtro da média com máscara 3x3; (d) resultado da filtragem pelo filtro da média com máscara 5 x 5.

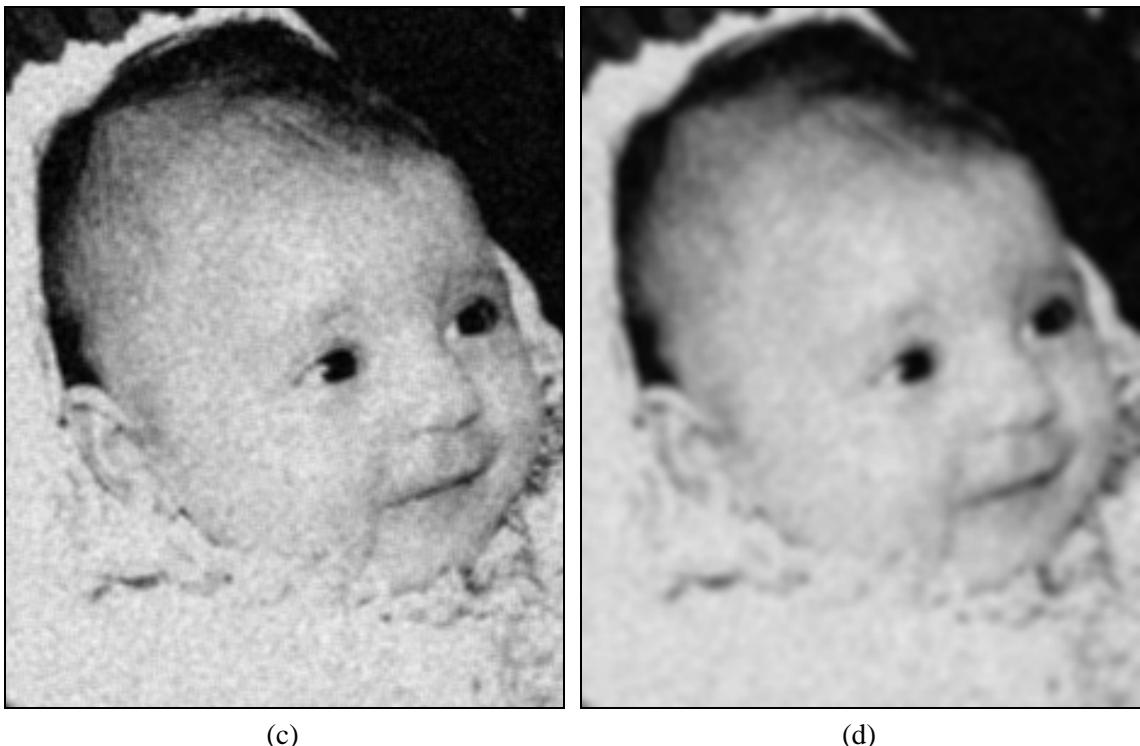


Figura 7 - Continuação.

O algoritmo básico de filtragem pela média pode ser alterado no sentido de minimizar a perda de definição na imagem resultante. Uma possível modificação consiste em incluir uma comparação do valor calculado com um limiar (T), antes de alterar seu tom de cinza. Se o valor absoluto da diferença entre o nível de cinza original do pixel ($f(x,y)$) e o valor calculado pela aplicação do filtro da média for menor que T , substitui-se o tom de cinza do pixel pelo valor calculado; caso contrário, mantém-se o valor de cinza original. O objetivo principal desta modificação é diminuir o efeito de suavização indesejável das bordas dos objetos presentes na imagem.

4.2.3 Filtro da mediana

Uma das principais limitações do filtro da média em situações onde o objetivo é remoção de ruídos em imagens está na sua incapacidade de preservar bordas e detalhes finos da imagem. Para contorná-la, uma técnica alternativa é o filtro da mediana. Nesta técnica, o nível de cinza do pixel central da janela é substituído pela mediana dos pixels situados em sua vizinhança.

Este método não-linear apresenta desempenho particularmente bom em situações nas quais a imagem é contaminada por ruído impulsivo (sal-e-pimenta), como ilustra a figura 8. Já para situações em que o ruído é do tipo gaussiano (figura 9), seu desempenho é apenas satisfatório, comparável ao do filtro pela média.

A mediana m de um conjunto de n elementos é o valor tal que metade dos n elementos do conjunto situem-se abaixo de m e a outra metade acima de m . Quando n é ímpar, a mediana é o próprio elemento central do conjunto ordenado. Nos casos em que n é par, a mediana é calculada pela média aritmética dos dois elementos mais próximos do centro. A ordenação (*sorting*) constitui uma etapa de tempo de processamento relativamente alto, apesar de inúmeros métodos eficientes existentes na literatura. Para reduzir o custo computacional do filtro da mediana, Pratt e outros [Pratt et al. 1984] propuseram um método alternativo, denominado filtro da pseudomediana, o qual estabelece que a pseudomediana de um conjunto de L elementos (S_L) pode ser computada como:

$$PMED\{S_L\} = \frac{MAXIMIN\{S_L\} + MINIMAX\{S_L\}}{2}, \text{ onde:}$$

$$\begin{aligned} MAXIMIN\{S_L\} &= MAX\{[MIN(S_1, \dots, S_M)], [MIN(S_2, \dots, S_{M+1})], \dots, \\ &[MIN(S_{L-M+1}, \dots, S_L)]\} \end{aligned} \quad (4.6)$$

$$\begin{aligned} MINIMAX\{S_L\} &= MIN\{[MAX(S_1, \dots, S_M)], [MAX(S_2, \dots, S_{M+1})], \dots, \\ &[MAX(S_{L-M+1}, \dots, S_L)]\} \end{aligned}$$

$$\text{para } M = \frac{L+1}{2}.$$

Leitura complementar

O capítulo 1 de [Dougherty 1994] trata do filtro da mediana com grande rigor matemático, abordando-o tanto sob o enfoque estatístico quanto algébrico.

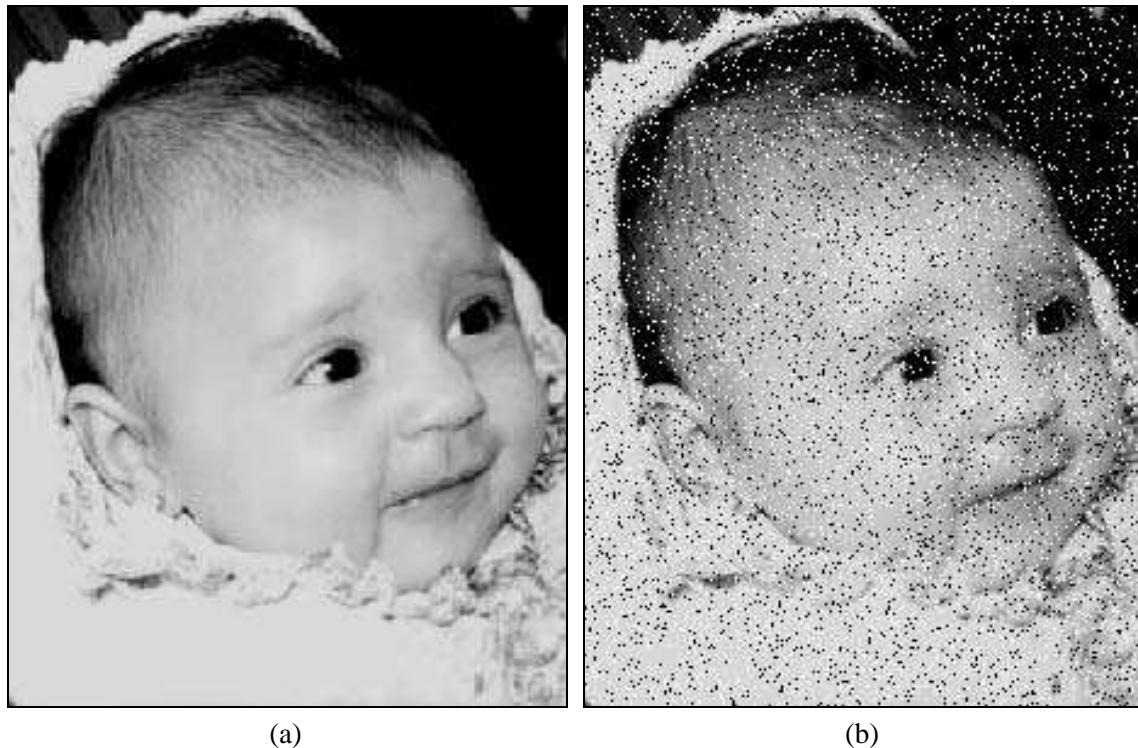


Figura 8 - (a) Imagem original; (b) imagem contaminada por ruído impulsivo (sal e pimenta); (c) resultado da filtragem pelo filtro da mediana com máscara 3x3; (d) resultado da filtragem pelo filtro da média com máscara 3 x 3.

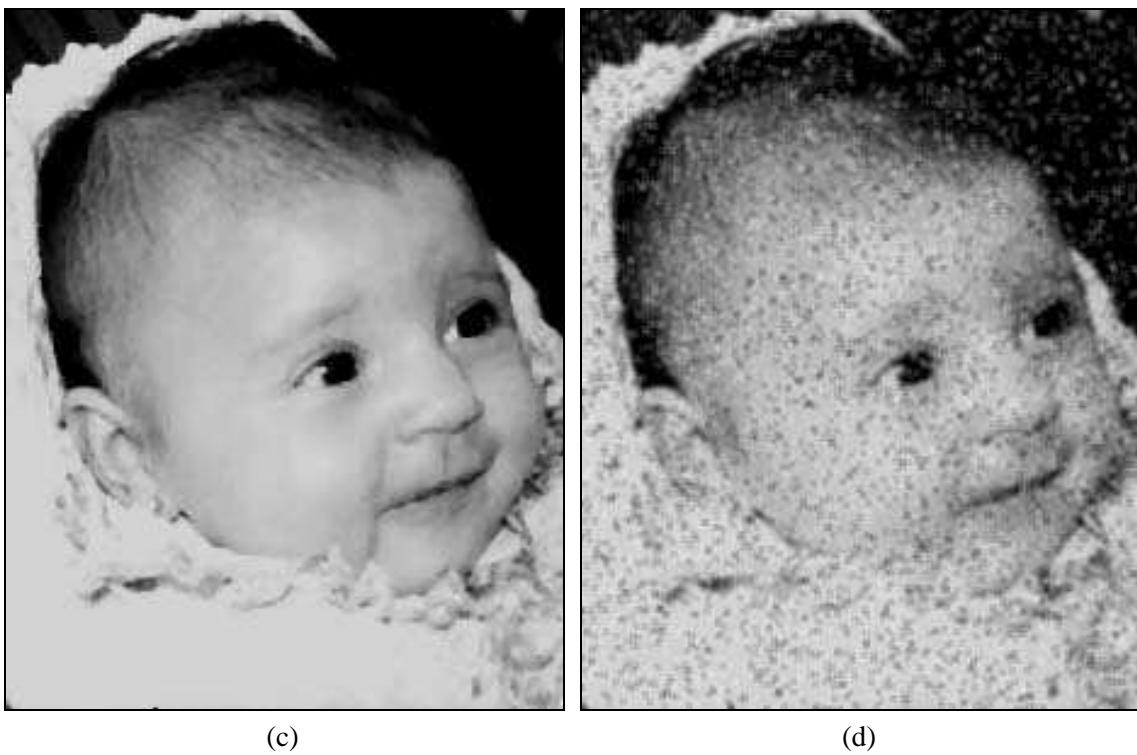


Figura 8 - Continuação.

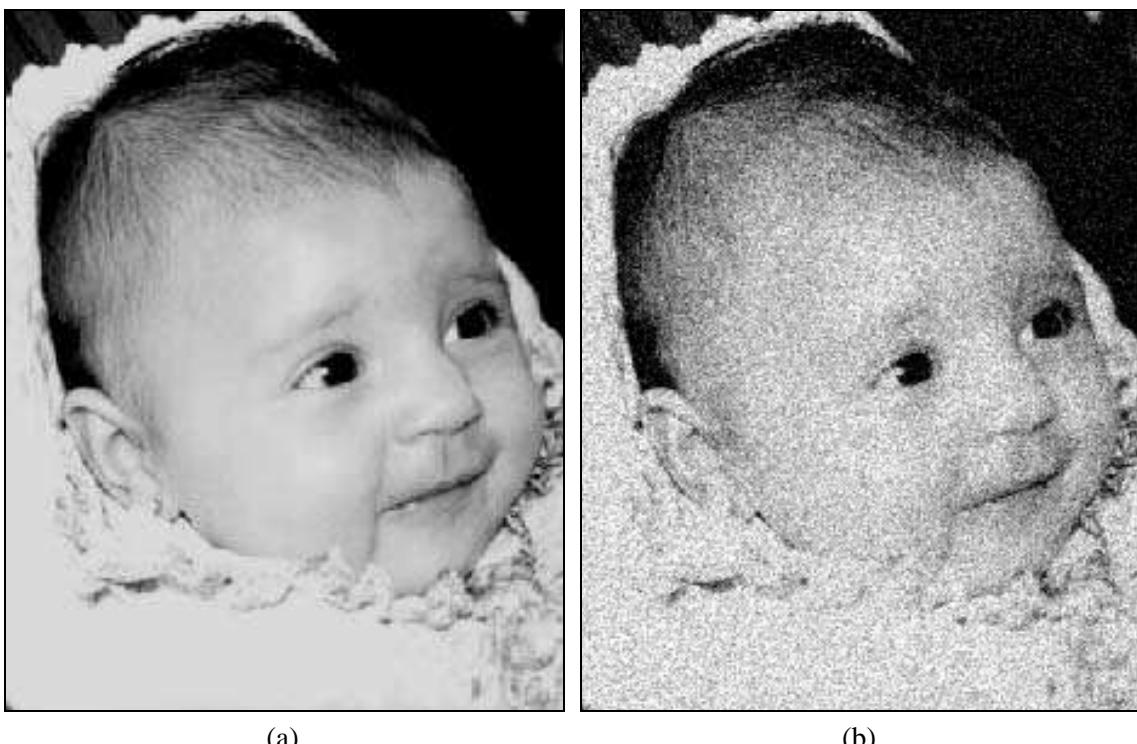


Figura 9 - (a) Imagem original; (b) imagem contaminada por ruído gaussiano; (c) resultado da filtragem pelo filtro da mediana com máscara 3x3; (d) resultado da filtragem pelo filtro da média com máscara 3 x 3.

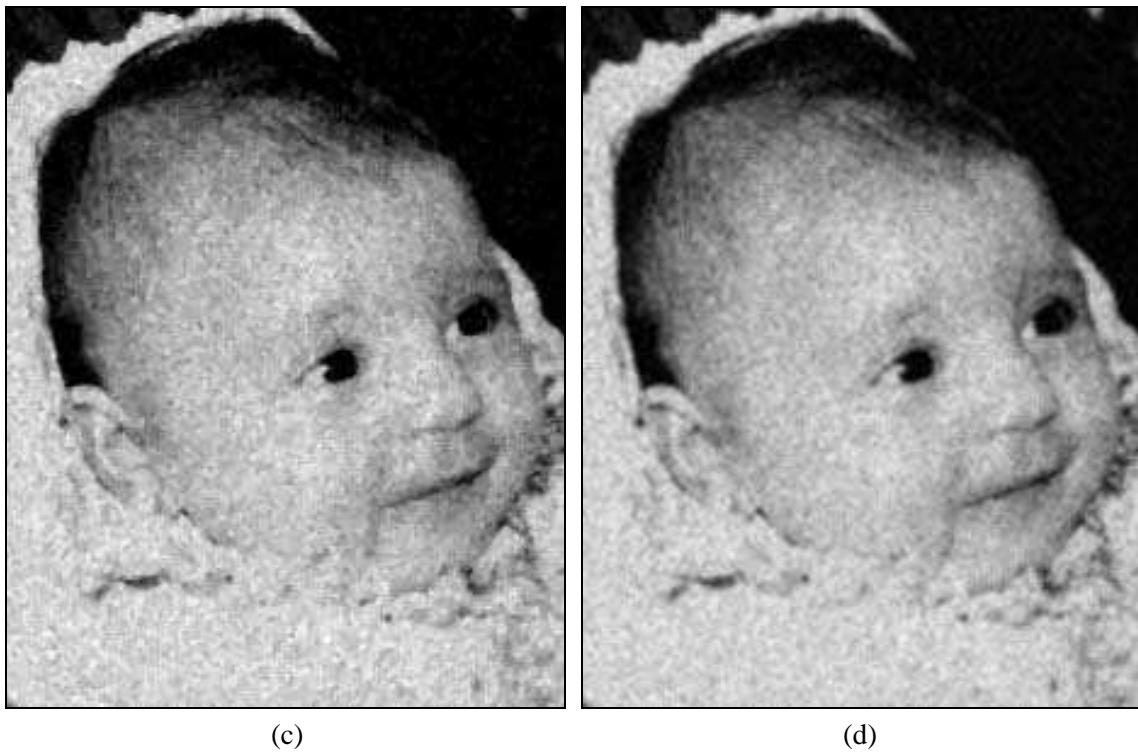


Figura 9 – Continuação.

4.2.4 Outros filtros

Existem diversos outros filtros espaciais para suavização de imagens, propostos e publicados por pesquisadores de todo o mundo nos últimos anos. Nesta seção destacaremos dois deles: o filtro da média de múltiplas imagens e o filtro da média dos k vizinhos mais próximos.

Média de múltiplas imagens

Seja uma imagem ruidosa $g(x,y) = f(x,y) + \eta(x,y)$ onde $f(x,y)$ é a imagem original e $\eta(x,y)$ é um padrão de ruído aditivo de média zero e descorrelacionado, que se sobrepõe à imagem. Supondo também a existência de M imagens ruidosas, cada qual adquirida em um instante diferente, pode-se calcular uma imagem média:

$$\bar{g}(x,y) = \frac{1}{M} \sum_{i=1}^M g_i(x,y) \quad (4.7)$$

na qual a influência do ruído terá sido minimizada.

Pode-se mostrar que:

$$E\{\bar{g}(x,y)\} = f(x,y) \quad (4.8)$$

$$\sigma_{\bar{g}(x,y)} = \frac{1}{\sqrt{M}} \sigma_{\eta(x,y)} \quad (4.9)$$

$$\sigma_{\bar{g}(x,y)}^2 = \frac{1}{M} \sigma_{\eta(x,y)}^2 \quad (4.10)$$

onde $E\{\bar{g}(x,y)\}$ é o valor esperado de $\bar{g}(x,y)$, $\sigma_{\bar{g}(x,y)}^2$ e $\sigma_{\eta(x,y)}^2$ são, respectivamente, as variâncias da imagem filtrada e do ruído aditivo, enquanto $\sigma_{\bar{g}(x,y)}$ e $\sigma_{\eta(x,y)}$ são seus respectivos desvios-padrão.

As equações (4.8) a (4.10) nos permitem concluir que quanto maior for o valor de M , menor a variância (e portanto o desvio padrão) dos pixels de $\bar{g}(x,y)$ e mais a imagem $\bar{g}(x,y)$ irá se aproximar de $f(x,y)$.

Esta técnica opera de forma igualmente satisfatória para ruído gaussiano ou aleatório, quando o número de imagens utilizadas no cálculo da imagem média é significativo, devido ao Teorema do Limite Central, que estabelece que a soma de um grande número de termos representando ruídos aleatórios tende a produzir um ruído resultante do tipo gaussiano e independente dos tipos dos ruídos incluídos naquela soma.

A figura 10 apresenta um exemplo de uso da técnica da média de múltiplas imagens para redução de ruído.

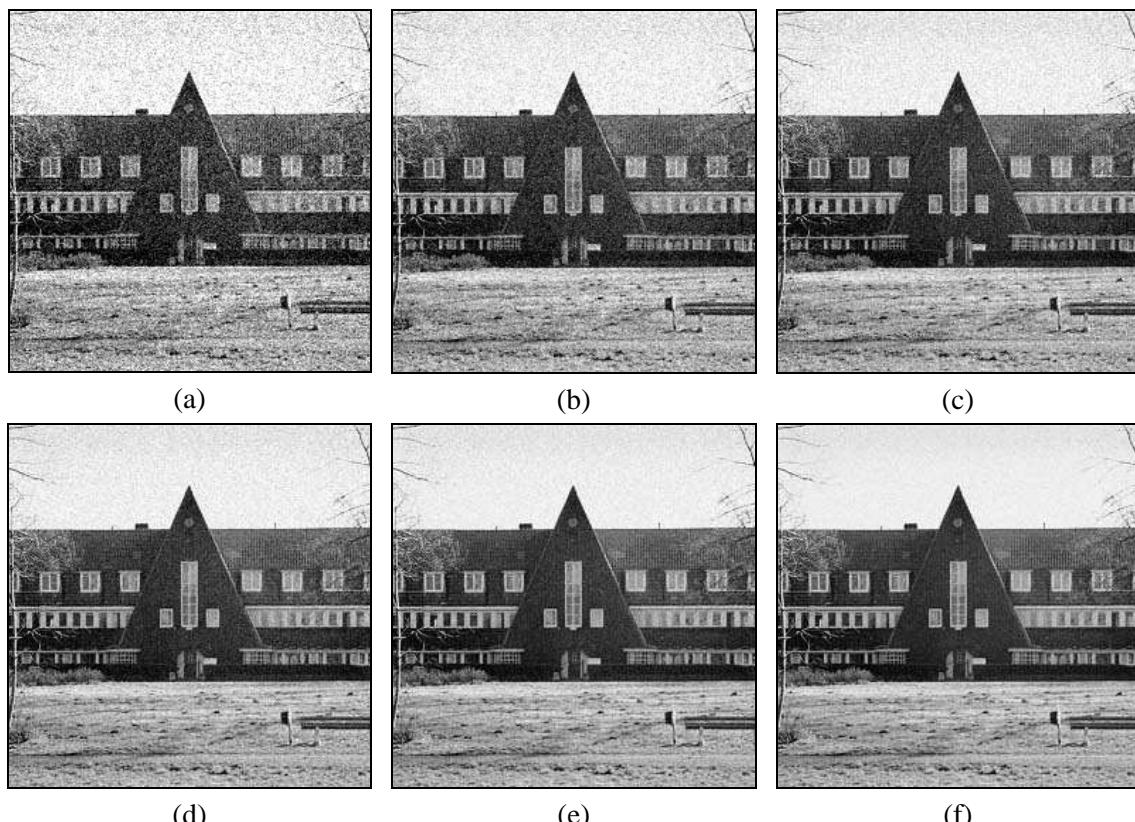


Figura 10 - Exemplo de redução de ruídos usando média de múltiplas imagens: (a) imagem ruidosa; (b)-(f) resultados para $M = 2, 4, 8, 16$ e 32 imagens.

Média dos k vizinhos mais próximos

Esta técnica, descrita em [Davis e Rosenfeld 1978], consiste em uma variação do método de filtragem pela média, na qual o pixel central da janela é substituído pela média dos k vizinhos cujas amplitudes mais se aproximam da amplitude do pixel central. Seu objetivo é deliberadamente evitar incluir no cálculo da média valores que possam estar sob a janela em decorrência de bordas ou regiões de alto contraste. Quanto maior o valor de k , mais o desempenho deste filtro se aproximará do filtro da média.

Exercício resolvido

Considere o trecho de imagem digital a seguir, representado por uma matriz 5 x 5. Seja o pixel central o pixel de referência. Forneça o valor resultante do pixel central caso a imagem seja processada:

- pelo algoritmo da filtragem pela mediana utilizando janela 3 x 3.
- pelo algoritmo da média utilizando janela 5 x 5.
- pela média dos k vizinhos mais próximos, utilizando janela 5 x 5, sendo $k = 9$.
- pelo algoritmo da pseudomediana utilizando janela 3 x 3.

121	20	198	84	4
87	188	189	99	8
88	115	134	49	19
16	18	187	98	9
12	103	15	176	38

Solução:

- Os 9 elementos sob a janela 3 x 3 são: {188, 189, 99, 115, 134, 49, 18, 187, 98}. Ordenando esta lista, temos: {18, 49, 98, 99, 115, 134, 187, 188, 189}. Logo, o elemento mediano é 115.
- Calculando a média dos 25 pixels sob a máscara, obtemos: 83.
- Os 9 vizinhos do pixel central cujos níveis de cinza mais se aproximam do valor 134 são: {121, 84, 87, 99, 88, 115, 98, 103, 176}. Calculando a média desses valores, obtemos: $107,889 \cong 108$.
- Neste caso, $L = 9$, logo $M = 5$. Portanto:

$$\text{MAXIMIN}\{S_9\} = \text{MAX}\{ \text{MIN}(188, 189, 99, 115, 134), \text{MIN}(189, 99, 115, 134, 49), \text{MIN}(99, 115, 134, 49, 18), \text{MIN}(115, 134, 49, 18, 187), \text{MIN}(134, 49, 18, 187, 98) \} = \text{MAX}(99, 49, 18, 18, 18) = 99$$

$$\text{MINIMAX}\{S_9\} = \text{MIN}\{ \text{MAX}(188, 189, 99, 115, 134), \text{MAX}(189, 99, 115, 134, 49), \text{MAX}(99, 115, 134, 49, 18), \text{MAX}(115, 134, 49, 18, 187), \text{MAX}(134, 49, 18, 187, 98) \} = \text{MIN}(189, 189, 134, 187, 187) = 134$$

$$\text{PMED}\{S_9\} = (99 + 134) / 2 = 116,5 \cong 117. \text{ (Resultado próximo ao obtido no item (a)).}$$

Leitura complementar

O capítulo 3 de [Pavlidis 1982] apresenta conceitos e algoritmos de filtragem (linear ou não-linear) no domínio espacial.

Diversas outras técnicas de filtragem no domínio espacial são resenhadas em [Araújo 1989].

4.3 Realce de imagens no domínio espacial

O principal objetivo das técnicas de realce é o de destacar detalhes finos na imagem. Nesta seção apresentaremos três métodos de realce de imagens no domínio espacial, a saber: filtro passa-altas básico, realce por diferenciação e ênfase em alta freqüência.

4.3.1 Filtro passa-altas básico

O formato da resposta ao impulso de um filtro passa-altas (figura 3(b)) deve ser tal que a máscara correspondente apresente coeficientes positivos nas proximidades de seu centro e negativos longe dele. No caso de uma máscara 3 x 3, isto significa projetar uma máscara com pixel central positivo e todos seus oito vizinhos negativos. Um exemplo de máscara com estas características é apresentado na figura 11. Pode-se notar que a soma algébrica dos coeficientes desta máscara é zero, significando que quando aplicada a regiões homogêneas de uma imagem, o resultado será zero ou um valor muito baixo, o que é consistente com o princípio da filtragem

passa-altas. A figura 12 mostra um exemplo de resultado de aplicação da máscara da figura 11 a uma imagem monocromática.

$$\frac{1}{9} \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Figura 11 - Exemplo de máscara para filtragem espacial passa-altas.



(a)

(b)

Figura 12 - (a) Imagem original; (b) imagem resultante após filtragem passa-altas com a máscara da figura 11.

4.3.2 Realce por diferenciação

Sabendo-se que o cálculo da média dos pixels em um trecho de imagem produz como efeito a remoção de seus componentes de alta freqüência e que o conceito de média é análogo à operação de integração, é razoável esperar que a diferenciação produza o efeito oposto e, portanto, enfatize os componentes de alta freqüência presentes em uma imagem. O método mais usual de diferenciação em aplicações de processamento de imagens é o gradiente. Em termos contínuos, o gradiente de $f(x,y)$ em um certo ponto (x,y) é definido como o vetor:

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \quad (4.11)$$

A magnitude deste vetor é dada por:

$$\nabla f = \text{mag}(\nabla f) = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \quad (4.12)$$

e é utilizada por várias técnicas de realce de imagens por diferenciação.

Para uma imagem digital, o gradiente pode ser aproximado por:

$$G[f(x, y)] \cong \{[f(x, y) - f(x+1, y)]^2 + [f(x, y) - f(x, y+1)]^2\}^{1/2} \quad (4.13)$$

ou por:

$$G[f(x, y)] \cong |f(x, y) - f(x+1, y)| + |f(x, y) - f(x, y+1)| \quad (4.14)$$

Outra aproximação, conhecida como gradiente de Roberts, utiliza as diferenças cruzadas, isto é, na diagonal:

$$G[f(x, y)] \cong \{[f(x, y) - f(x+1, y+1)]^2 + [f(x+1, y) - f(x, y+1)]^2\}^{1/2} \quad (4.15)$$

ou :

$$G[f(x, y)] \cong |f(x, y) - f(x+1, y+1)| + |f(x+1, y) - f(x, y+1)| \quad (4.16)$$

As equações (4.14) e (4.16) podem ser implementadas usando máscaras de tamanho 2 x 2, como as mostradas na figura 13, ou de dimensões 3 x 3, como por exemplo os operadores de Prewitt e Sobel, apresentados na Seção 2.4.

1	-1
0	0

1	0
-1	0

(a)

1	0
0	-1

0	1
-1	0

(b)

Figura 13 - Implementação do gradiente usando máscara 2 x 2: (a) gradiente convencional; (b) gradiente de Roberts.

4.3.3 Filtragem *high-boost*

A filtragem passa-altas também pode ser obtida subtraindo de uma imagem original uma versão filtrada por um filtro passa-baixas, ou seja:

$$\text{Passa-altas} = \text{Original} - \text{Passa-baixas} \quad (4.17)$$

O filtro *high-boost* ou técnica da ênfase em alta freqüência nada mais é que uma extensão da idéia original formulada na equação (4.17), na qual a imagem original é multiplicada por um fator de amplificação *A*:

$$\begin{aligned} \text{High-boost} &= (A) (\text{Original}) - \text{Passa-baixas} \\ &= (A - 1) (\text{Original}) + \text{Original} - \text{Passa-baixas} \\ &= (A - 1) (\text{Original}) + \text{Passa-altas}. \end{aligned} \quad (4.18)$$

Quando $A = 1$, o filtro se comporta de forma idêntica a um passa-altas. Nos casos em que $A > 1$, parte da imagem original é adicionada ao resultado, restaurando parcialmente os componentes de baixa freqüência. O resultado é uma imagem que se parece com a original, com um grau relativo de realce das bordas, dependente do valor de A . O processo genérico de subtração de uma imagem borrada da imagem original é conhecido na literatura como *unsharp masking*.

A ênfase em alta freqüência pode ser implementada utilizando a máscara da figura 14, na qual

$$w = 9A - 1 \quad (4.19)$$

com $A \geq 1$. A figura 15 mostra o efeito da variação de A no resultado final da filtragem.

$$\frac{1}{9} \begin{bmatrix} -1 & -1 & -1 \\ -1 & w & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Figura 14 - Máscara usada para filtragem *high-boost*.



Figura 15 - (a) imagem original; resultados da filtragem *high-boost* com a máscara da figura 14 para (b) $A = 1,1$, (c) $A = 1,15$ e (d) $A = 1,2$, respectivamente.

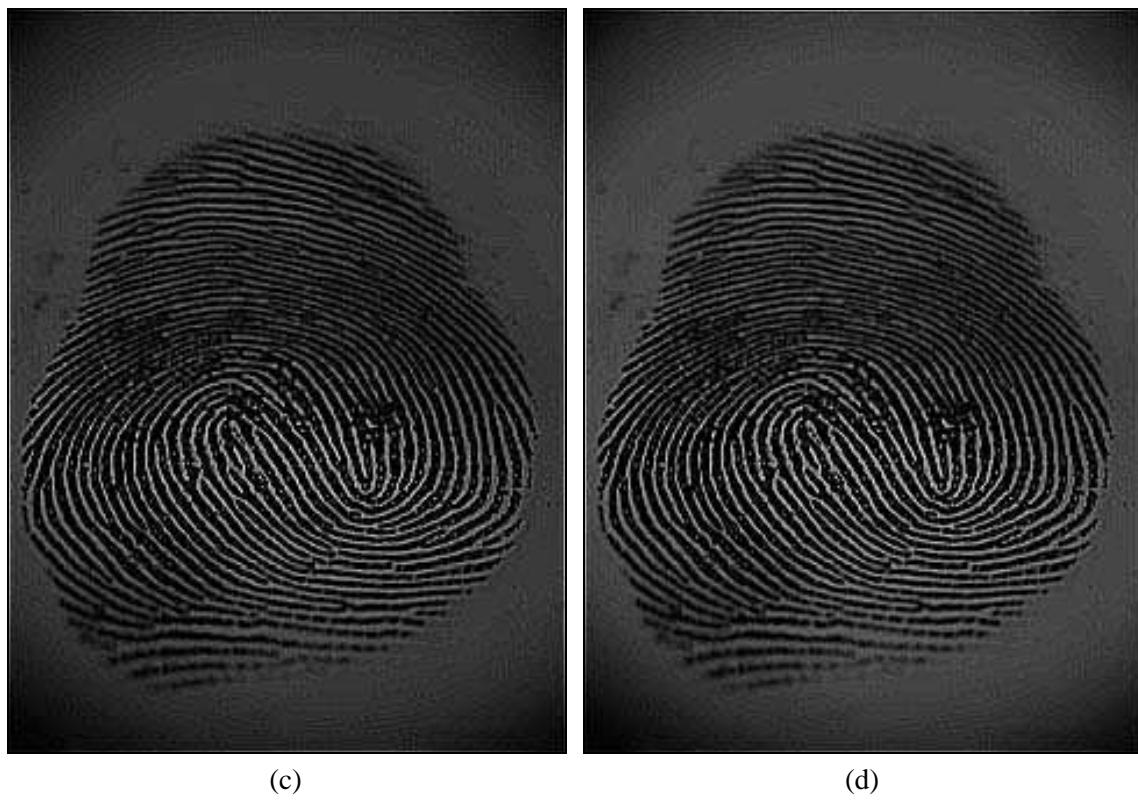


Figura 15 - Continuação.

4.4 Transformada de Fourier

Esta seção apresenta de forma sucinta os principais conceitos e propriedades da transformada de Fourier (FT), ferramenta matemática obrigatória para a especificação e projeto de filtros no domínio da freqüência.

4.4.1 Transformada de Fourier para sinais unidimensionais (1-D) contínuos

Seja $f(x)$ uma função contínua da variável real x . A transformada de Fourier (FT) de $f(x)$, indicada por $\mathfrak{F}\{f(x)\}$, é definida pela equação

$$\mathfrak{F}\{f(x)\} = F(u) = \int_{-\infty}^{\infty} f(x) \exp[-j2\pi ux] dx \quad (4.20)$$

onde $j = \sqrt{-1}$.

Dado $F(u)$, $f(x)$ pode ser obtida calculando-se a transformada inversa de Fourier (IFT)

$$\mathfrak{F}^{-1}\{F(u)\} = f(x) = \int_{-\infty}^{\infty} F(u) \exp[j2\pi ux] du \quad (4.21)$$

As equações (4.20) e (4.21), chamadas conjuntamente de 'par de Fourier', existem se $f(x)$ for contínua e integrável e $F(u)$ for integrável. Na prática, estas condições são quase sempre satisfeitas.

A FT de uma função real é, geralmente, um valor complexo¹, ou seja,

$$F(u) = R(u) + jI(u) \quad (4.22)$$

onde $R(u)$ e $I(u)$ são, respectivamente, os componentes reais e imaginários de $F(u)$. $F(u)$ também pode ser representada exponencialmente através da equação

$$F(u) = |F(u)|e^{j\phi(u)} \quad (4.23)$$

onde a componente de magnitude $|F(u)|$ é denominada 'espectro de Fourier' de $f(x)$ e $\phi(u)$ seu ângulo de fase. A figura 16 mostra um exemplo de função unidimensional simples (pulso retangular de amplitude A) e seu respectivo espectro de Fourier.

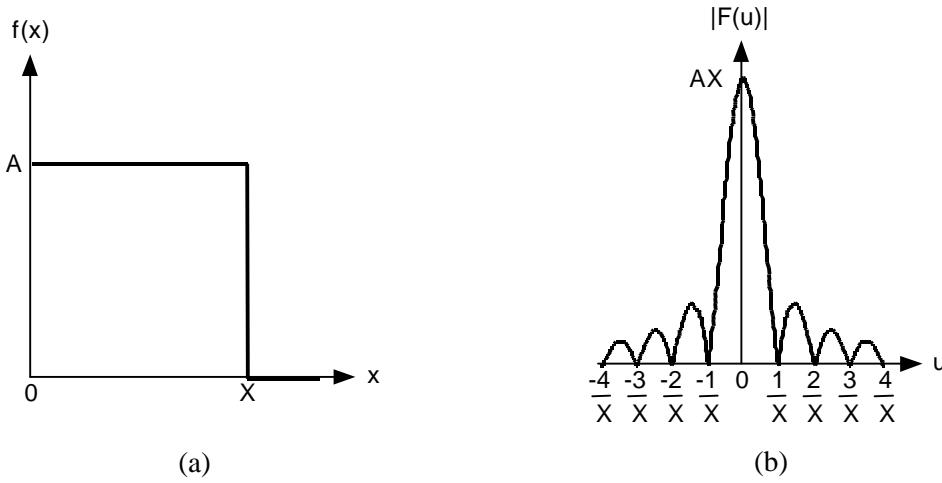


Figura 16 - Uma função unidimensional contínua simples e seu respectivo espectro de Fourier.

4.4.2 Transformada de Fourier para sinais bidimensionais (2-D) contínuos

O conceito de transformada de Fourier pode ser facilmente estendido para uma função de duas variáveis $f(x,y)$. Se $f(x,y)$ é contínua e integrável e $F(u,v)$ é integrável, então o par de Fourier a seguir existe:

$$\mathfrak{F}\{f(x,y)\} = F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) \exp[-j2\pi(ux+vy)] dx dy \quad (4.24)$$

e

$$\mathfrak{F}^{-1}\{F(u,v)\} = f(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u,v) \exp[j2\pi(ux+vy)] dx dy \quad (4.25)$$

Assim como no caso 1-D, $F(u,v)$ é complexa e sua amplitude, $|F(u,v)|$, é denominada espectro de Fourier.

A figura 17 mostra uma função bidimensional contínua e seu respectivo espectro de Fourier, representado de duas maneiras: em perspectiva tridimensional (figura 17(b)) e como

¹ Convém notar que uma imagem é um conjunto de números reais, porém a FT opera igualmente para conjuntos de números complexos.

uma função de intensidade, na qual o brilho é proporcional à amplitude de $|F(u,v)|$ (figura 17(c)).

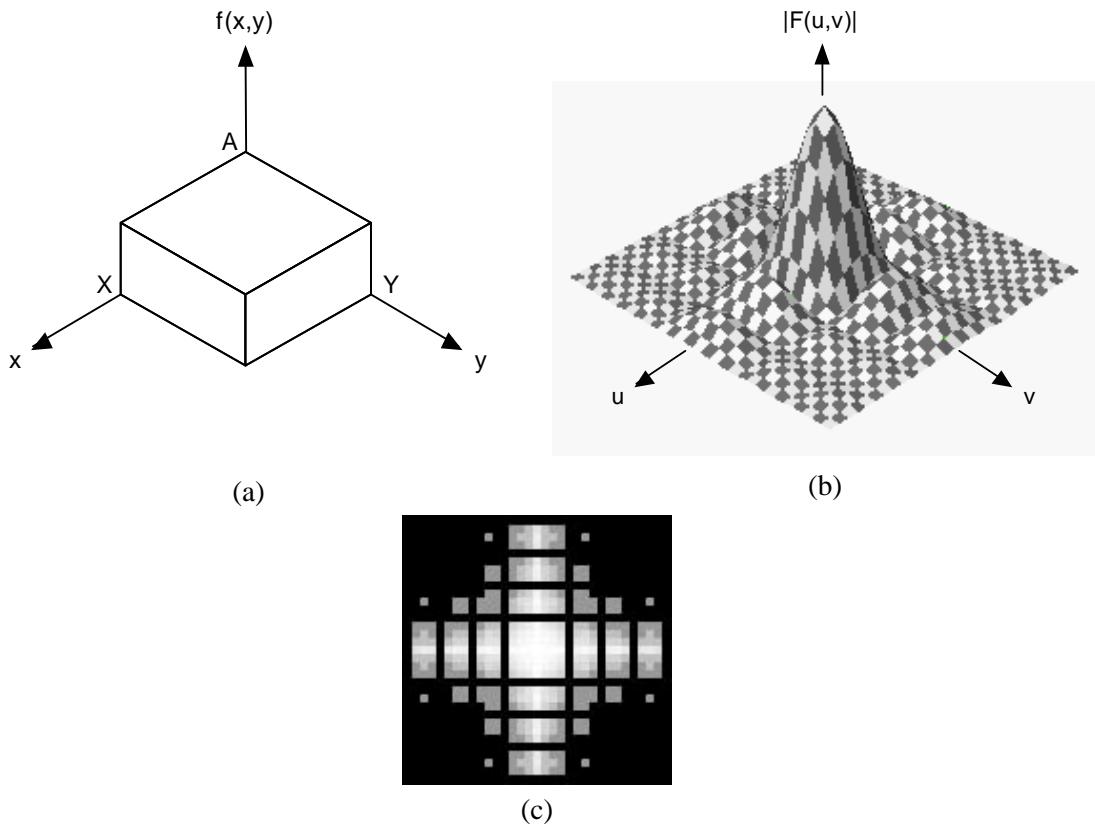


Figura 17 - Uma função bidimensional contínua (a), seu espectro de Fourier (b) e o espectro representado como uma função de intensidade (c).

4.4.3 Transformada de Fourier para sinais unidimensionais (1-D) discretos

Supondo que uma função contínua $f(x)$ seja discretizada, produzindo a seqüência

$$\{f(x_0), f(x_0 + \Delta x), f(x_0 + 2\Delta x), \dots, f(x_0 + [N-1]\Delta x)\}$$

através da amostragem de N pontos, espaçados entre si de Δx .

Denominando a função discreta $f(n)$,

$$f(n) = f(x_0 + n\Delta x) \quad (4.26)$$

onde n pode assumir os valores discretos 0, 1, 2, ..., N-1, podemos interpretar a seqüência $\{f(0), f(1), f(2), \dots, f(N-1)\}$ como qualquer seqüência de N amostras consecutivas do sinal original contínuo $f(x)$, uniformemente espaçadas.

A transformada de Fourier de $f(n)$ será:

$$F(u) = \frac{1}{N} \sum_{n=0}^{N-1} f(n) \exp\left[-j2\pi \frac{un}{N}\right] \quad (4.27)$$

para $u = 0, 1, 2, \dots, N-1$ e a transformada inversa de Fourier de $F(u)$ será:

$$f(n) = \frac{1}{N} \sum_{u=0}^{N-1} F(u) \exp\left[\frac{j2\pi un}{N}\right] \quad (4.28)$$

para $n = 0, 1, 2, \dots, N-1$.

Os valores $u = 0, 1, 2, \dots, N-1$ na transformada discreta de Fourier (eq. 4.27) correspondem a amostras dos valores da transformada do sinal contínuo nos pontos $0, \Delta u, 2\Delta u, \dots, (N-1)\Delta u$. Em outras palavras, $F(u)$ representa $F(u\Delta u)$. Portanto, os intervalos de espaçamento entre as amostras do sinal e de sua transformada estão relacionados através da expressão

$$\Delta u = \frac{1}{N\Delta x} \quad (4.29)$$

4.4.4 Transformada de Fourier para sinais bidimensionais (2-D) discretos

Seja agora uma função contínua bidimensional $f(x,y)$, discretizada em M amostras ao longo de x e N amostras ao longo de y . Neste caso, sua transformada discreta de Fourier será:

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \exp\left[-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)\right] \quad (4.30)$$

para $u = 0, 1, 2, \dots, M-1, v = 0, 1, 2, \dots, N-1$.

A transformada inversa é dada por:

$$f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) \exp\left[j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)\right] \quad (4.31)$$

O processo de amostragem da função bidimensional contínua $f(x,y)$ pode ser visto como uma grade 2-D, com impulsos unitários espaçados de Δx e Δy , nos eixos x e y respectivamente. A função $f(x,y)$ discretizada (utilizada nas eqs. 4.30 e 4.31) representa as amostras de $f(x,y)$ contínua original em pontos espaçados entre si de Δx e Δy , nos respectivos eixos. Os intervalos de amostragem nos domínios espacial e freqüencial estão relacionados entre si por:

$$\Delta u = \frac{1}{M\Delta x} \quad (4.32)$$

e

$$\Delta v = \frac{1}{N\Delta y} \quad (4.33)$$

Para o caso particular em que $M = N$, as equações (4.30) e (4.31) podem ser reescritas como:

$$F(u,v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \exp\left[-j2\pi\left(\frac{ux + vy}{N}\right)\right] \quad (4.34)$$

e

$$f(x, y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u, v) \exp\left[j2\pi\left(\frac{ux + vy}{N}\right)\right] \quad (4.35)$$

Notar que o termo $1 / MN$ foi desmembrado em dois termos $1 / N$, um em cada equação, uma vez que o agrupamento ou desmembramento destas constantes multiplicativas é arbitrário.

4.4.5 Propriedades da transformada de Fourier para sinais bidimensionais (2-D) discretos

Existem diversas propriedades da FT 2-D de grande interesse para o processamento de imagens. Muitas delas são derivações de propriedades semelhantes da FT 1-D e o leitor familiarizado com processamento de sinais unidimensionais irá facilmente reconhecê-las. Outras só fazem sentido no caso 2-D, como a propriedade da separabilidade.

Separabilidade

O par de Fourier das eqs. (4.34) e (4.35) pode ser decomposto em

$$F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} \exp\left[-\frac{j2\pi ux}{N}\right] \sum_{y=0}^{N-1} f(x, y) \exp\left[-\frac{j2\pi vy}{N}\right] \quad (4.36)$$

para $u, v = 0, 1, 2, \dots, N-1$, e

$$f(x, y) = \frac{1}{N} \sum_{u=0}^{N-1} \exp\left[\frac{j2\pi ux}{N}\right] \sum_{v=0}^{N-1} F(u, v) \exp\left[\frac{j2\pi vy}{N}\right] \quad (4.37)$$

para $x, y = 0, 1, 2, \dots, N-1$.

A principal vantagem desta decomposição é permitir que a FT ou a IFT 2-D possa ser obtida através de duas aplicações do algoritmo da FT ou IFT 1-D. Esta vantagem se torna evidente ao reescrevermos a eq. (4.36) da seguinte forma:

$$F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} F(x, v) \exp\left[-\frac{j2\pi ux}{N}\right] \quad (4.38)$$

onde

$$F(x, v) = N \left[\frac{1}{N} \sum_{y=0}^{N-1} f(x, y) \exp\left[-\frac{j2\pi vy}{N}\right] \right] \quad (4.39)$$

Para cada valor de x , a expressão entre colchetes da eq. (4.39) é uma transformada 1-D, com valores de freqüência $v = 0, 1, 2, \dots, N-1$. Portanto, a função 2-D $F(x, v)$ é obtida calculando-se a transformada ao longo de cada linha de $f(x, y)$ e multiplicando o resultado por N . O resultado final, $F(u, v)$ será obtido mediante uma nova aplicação da FT 1-D, desta vez ao longo das colunas do resultado intermediário $F(x, v)$, como indica a eq. (4.38). Este procedimento é ilustrado na figura 18. Sua principal vantagem prática é a possibilidade de aproveitar todas as otimizações já publicadas sobre o algoritmo da Transformada Rápida de Fourier (FFT - *Fast Fourier Transform*), aplicando seus resultados a problemas bidimensionais.

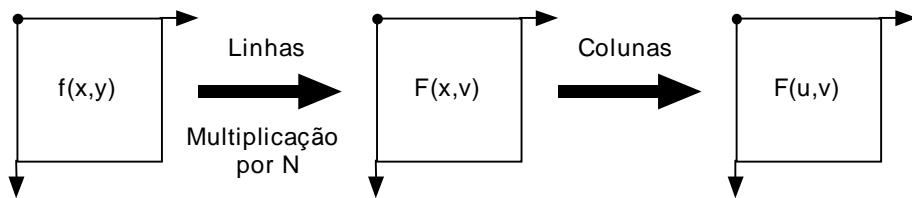


Figura 18 - Cálculo da transformada de Fourier 2-D a partir de duas aplicações do algoritmo da transformada 1-D.

Translação

As propriedades de translação do par de Fourier bidimensional são resumidas nas relações:

$$f(x, y) \exp\left[\frac{j2\pi(u_0x + v_0y)}{N}\right] \Leftrightarrow F(u - u_0, v - v_0) \quad (4.40)$$

e

$$f(x - x_0, y - y_0) \Leftrightarrow F(u, v) \exp\left[\frac{-j2\pi(ux_0 + vy_0)}{N}\right] \quad (4.41)$$

onde as setas duplas indicam a correspondência entre uma função e sua FT e vice-versa.

Para o caso particular em que $u_0 = v_0 = \frac{N}{2}$, a relação (4.40) se reduz a:

$$f(x, y)(-1)^{x+y} \Leftrightarrow F\left(u - \frac{N}{2}, v - \frac{N}{2}\right) \quad (4.42)$$

O deslocamento expresso na relação (4.42) é utilizado com bastante freqüência para uma melhor visualização do resultado da FT de uma imagem. Pode-se provar que tal deslocamento não altera a componente de magnitude da transformada resultante.

Periodicidade e simetria conjugada

A transformada discreta de Fourier e sua inversa são periódicas, com período N . Ou seja,

$$F(u, v) = F(u + N, v) = F(u, v + N) = F(u + N, v + N) \quad (4.43)$$

Se $f(x, y)$ é real, sua transformada de Fourier exibe também a propriedade conhecida como simetria conjugada:

$$F(u, v) = F^*(-u, -v) \quad (4.44)$$

ou

$$|F(u, v)| = |F(-u, -v)| \quad (4.45)$$

onde $F^*(u, v)$ é o conjugado complexo de $F(u, v)$.

A combinação das propriedades da translação e da periodicidade e a conveniência de sua utilização para fins de visualização podem ser ilustradas na figura 19.

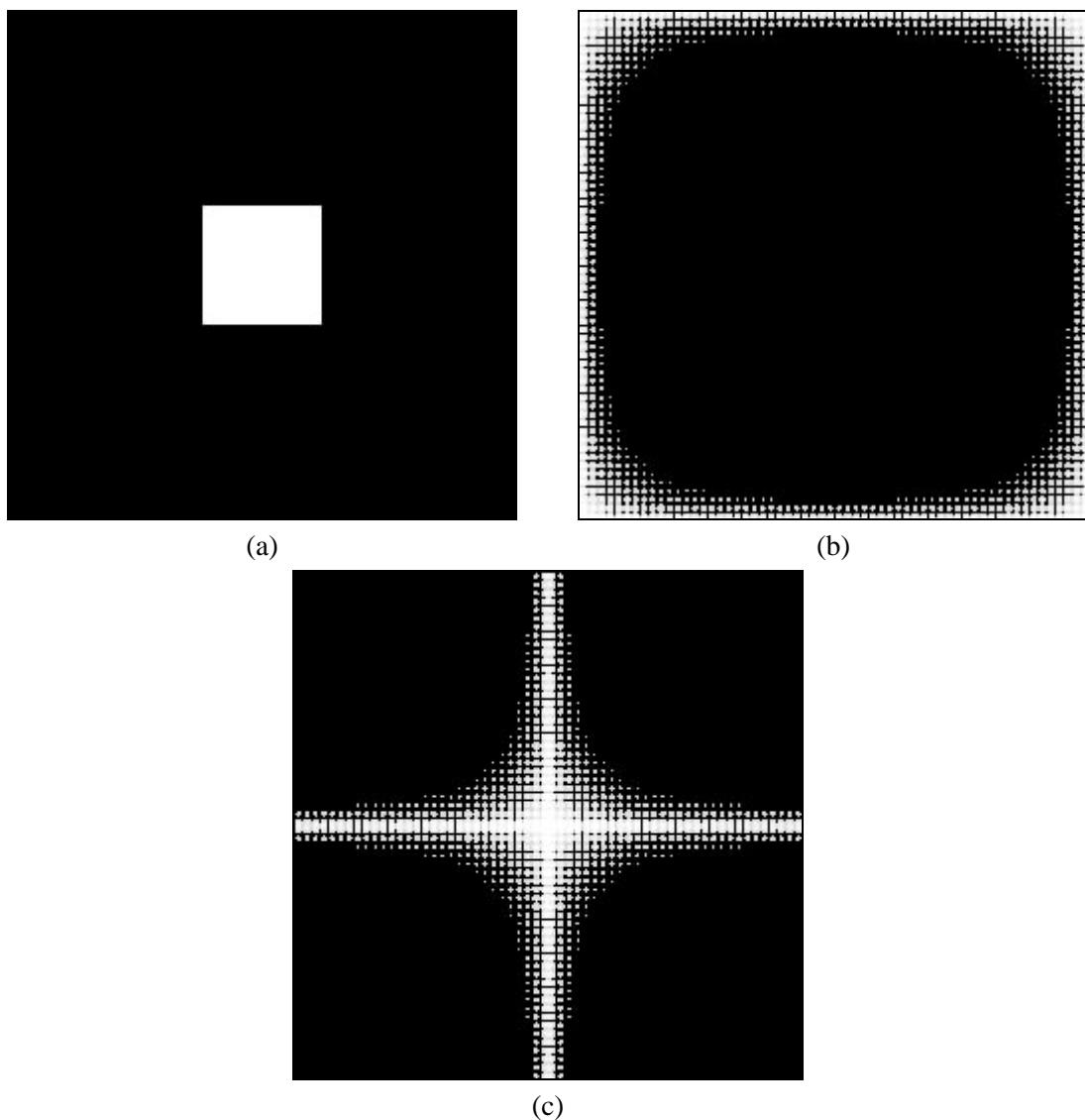


Figura 19 - (a) Imagem simples; (b) FT sem deslocamento; (c) FT após deslocamento para o centro do retângulo de referência.

Distributividade

A FT obedece à propriedade distributiva para a adição, mas não para a multiplicação, ou seja:

$$\Im\{f_1(x, y) + f_2(x, y)\} = \Im\{f_1(x, y)\} + \Im\{f_2(x, y)\} \quad (4.46)$$

e, em geral,

$$\Im\{f_1(x, y) \cdot f_2(x, y)\} \neq \Im\{f_1(x, y)\} \cdot \Im\{f_2(x, y)\} \quad (4.47)$$

Rotação

Em poucas palavras, a propriedade da rotação estabelece que, se uma imagem $f(x, y)$ for rotacionada de um certo ângulo θ_0 , sua transformada, $F(u, v)$, será rotacionada do mesmo ângulo. A figura 20 ilustra este conceito.

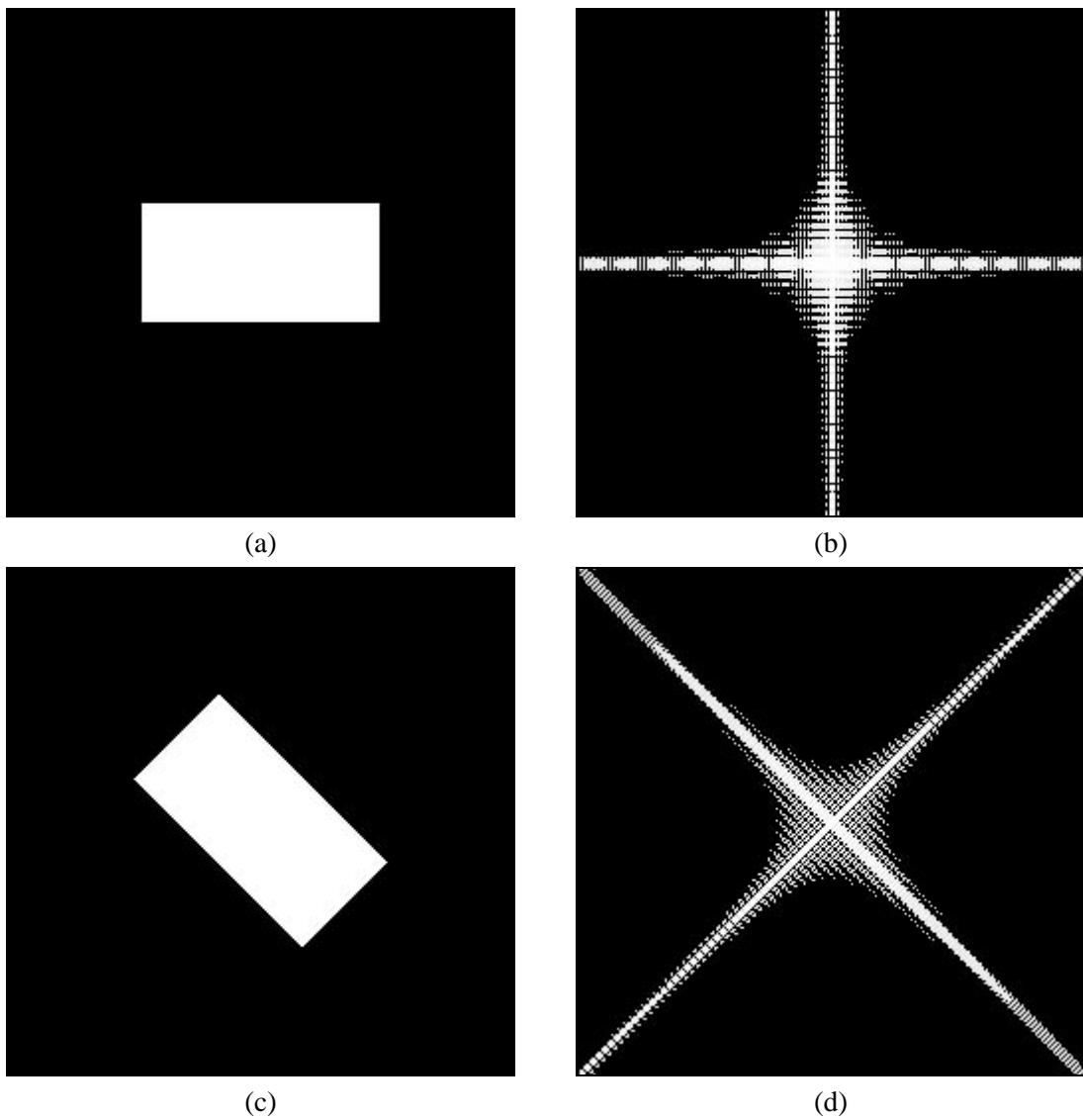


Figura 20 - (a) imagem original; (b) FT de (a); (c) imagem rotacionada; (d) FT de (c).

Escala

Sejam dois escalares a e b . Pode-se mostrar que:

$$af(x, y) \Leftrightarrow aF(u, v) \quad (4.48)$$

e

$$f(ax, by) \Leftrightarrow \frac{1}{|ab|} F\left(\frac{u}{a}, \frac{v}{b}\right) \quad (4.49)$$

Valor médio

O valor médio de uma função bidimensional $f(x, y)$ é dado por:

$$\bar{f}(x, y) = \frac{1}{N^2} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) \quad (4.50)$$

Substituindo $u = v = 0$ na eq. (4.36), obtemos

$$F(0,0) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y). \quad (4.51)$$

Logo, o valor médio de uma função 2-D está relacionado à sua FT através da relação

$$\bar{f}(x,y) = \frac{1}{N} F(0,0). \quad (4.52)$$

Laplaciano

O laplaciano de uma função de duas variáveis $f(x,y)$ é definido como:

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad (4.53)$$

A FT do laplaciano de uma função bidimensional é:

$$\Im\{\nabla^2 f(x,y)\} \Leftrightarrow -(2\pi)^2 (u^2 + v^2) F(u,v) \quad (4.54)$$

O laplaciano é um operador útil no processo de deteção de bordas, como indicado na Seção 2.4.

Convolução

O teorema da convolução, que no caso de funções unidimensionais pode ser resumido pelos pares de Fourier das eqs. (4.55) e (4.56), também pode ser estendido ao caso bidimensional, conforme indicado nas eqs. (4.57) e (4.58). Nestas equações, a operação de convolução é denotada por um asterisco.

$$f(x) * g(x) \Leftrightarrow F(u)G(u) \quad (4.55)$$

$$f(x)g(x) \Leftrightarrow F(u)*G(u) \quad (4.56)$$

$$f(x,y) * g(x,y) \Leftrightarrow F(u,v)G(u,v) \quad (4.57)$$

$$f(x,y)g(x,y) \Leftrightarrow F(u,v)*G(u,v) \quad (4.58)$$

4.4.6 A Transformada Rápida de Fourier (FFT)

Trata-se de um algoritmo cujo principal objetivo é reduzir o custo computacional do cálculo da FT de N pontos, substituindo o processo convencional de cálculo, no qual o número de multiplicações e adições é proporcional a N^2 por um engenhoso arranjo que combina diversas transformadas parciais, cada qual com pequeno número de pontos, em que o número de adições e multiplicações é proporcional a $N \log_2 N$. Para se poder apreciar a diferença em velocidade entre os algoritmos, pode-se supor $N = 512$ pontos, verificando que neste caso a FFT é mais de 56 vezes mais rápida. O detalhamento do algoritmo da FFT e seus diversos aprimoramentos foge ao escopo deste livro. O leitor interessado encontrará boas referências logo a seguir.

Leitura complementar

A transformada de Fourier é assunto suficientemente extenso para merecer ser abordado em um livro inteiro, como é o caso de [Papoulis 1962]. Ao leitor interessado em mais detalhes sobre a FT em uma ou duas dimensões indicamos, além do livro citado, o livro de Brigham [Brigham 1974], os capítulos 2 a 6 de [Enden e Verhoeckx 1989], os capítulos 4 e 5 de [Oppenheim et al. 1983], o capítulo 3 de [Gonzalez e Woods 1992] e os capítulos 1 e 3 de [Lim 1990].

O capítulo 2 de [Pavlidis 1982] apresenta algoritmo para cálculo da Transformada Rápida de Fourier (FFT) 1-D. Maiores considerações algorítmicas sobre a FFT para uma ou mais dimensões são encontradas no capítulo 12 de [Press et al. 1994].

4.5 Filtragem no domínio da freqüência

Conforme antecipamos na Seção 4.1.2, a idéia básica dos filtros no domínio da freqüência está em computar a FT da imagem a ser filtrada, multiplicar este resultado pela função de transferência do filtro e extrair a IFT do resultado.

4.5.1 Filtro passa-baixas (FPB)

Sendo $F(u,v)$ a transformada de Fourier da imagem a ser processada e sendo $G(u,v)$ a transformada de Fourier da imagem que se deseja obter à saída (com os componentes de alta freqüência atenuados), a filtragem passa-baixas consiste em encontrar um $H(u,v)$ tal que:

$$G(u,v) = F(u,v)H(u,v) \quad (4.59)$$

Filtro passa-baixas ideal

Um filtro passa-baixas 2-D ideal é aquele cuja função de transferência satisfaz a relação

$$H(u,v) = \begin{cases} 1 & \text{se } D(u,v) \leq D_0 \\ 0 & \text{se } D(u,v) > D_0 \end{cases} \quad (4.60)$$

onde D_0 é um valor não-negativo (análogo à freqüência de corte de um filtro 1-D), e $D(u,v)$ é a distância do ponto (u,v) à origem do plano de freqüência; isto é,

$$D(u,v) = (u^2 + v^2)^{1/2} \quad (4.61)$$

A figura 21 mostra a resposta em freqüência de um filtro passa-baixas 2-D ideal, $H(u,v)$, tanto em perspectiva 3-D (a) quanto em corte (b).

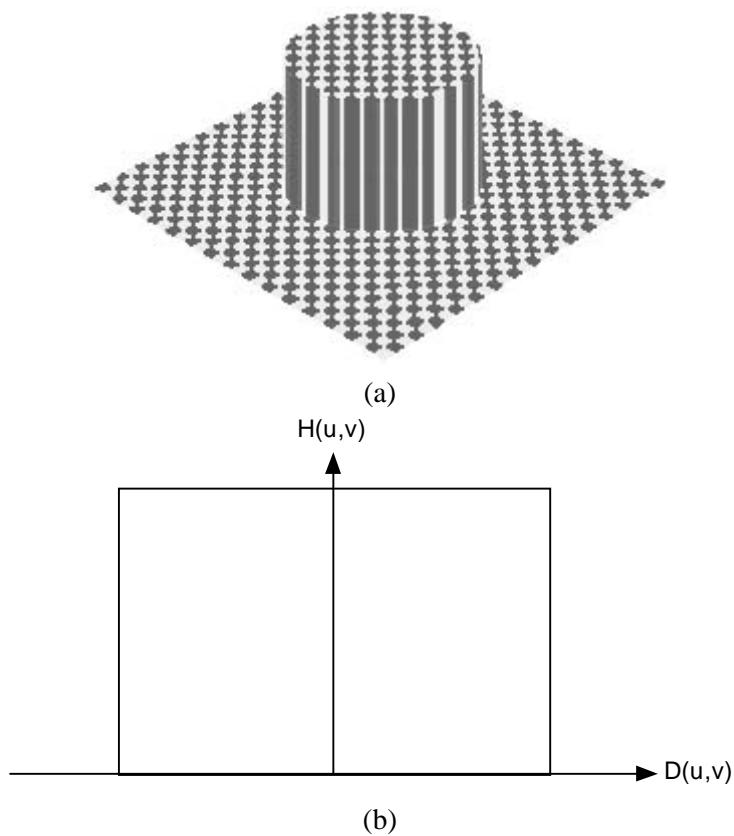


Figura 21 - Resposta em freqüência de um filtro passa-baixas ideal.

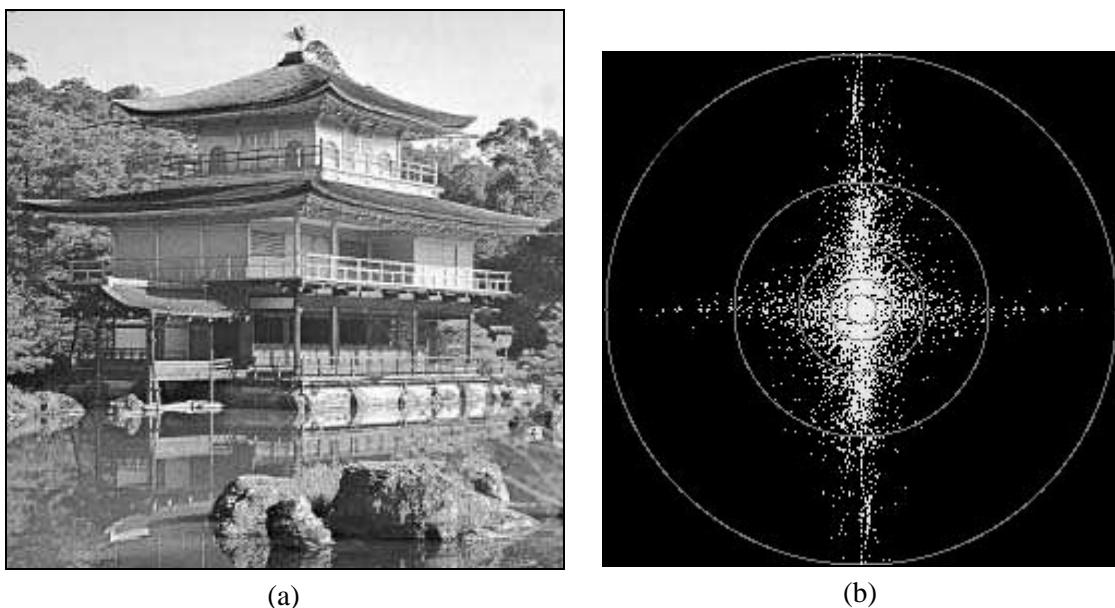


Figura 22 - (a) Imagem 512 x 512 e (b) seu respectivo espectro de Fourier. Os anéis sobrepostos ao espectro indicam as freqüências de corte dos filtros passa-baixas correspondentes.

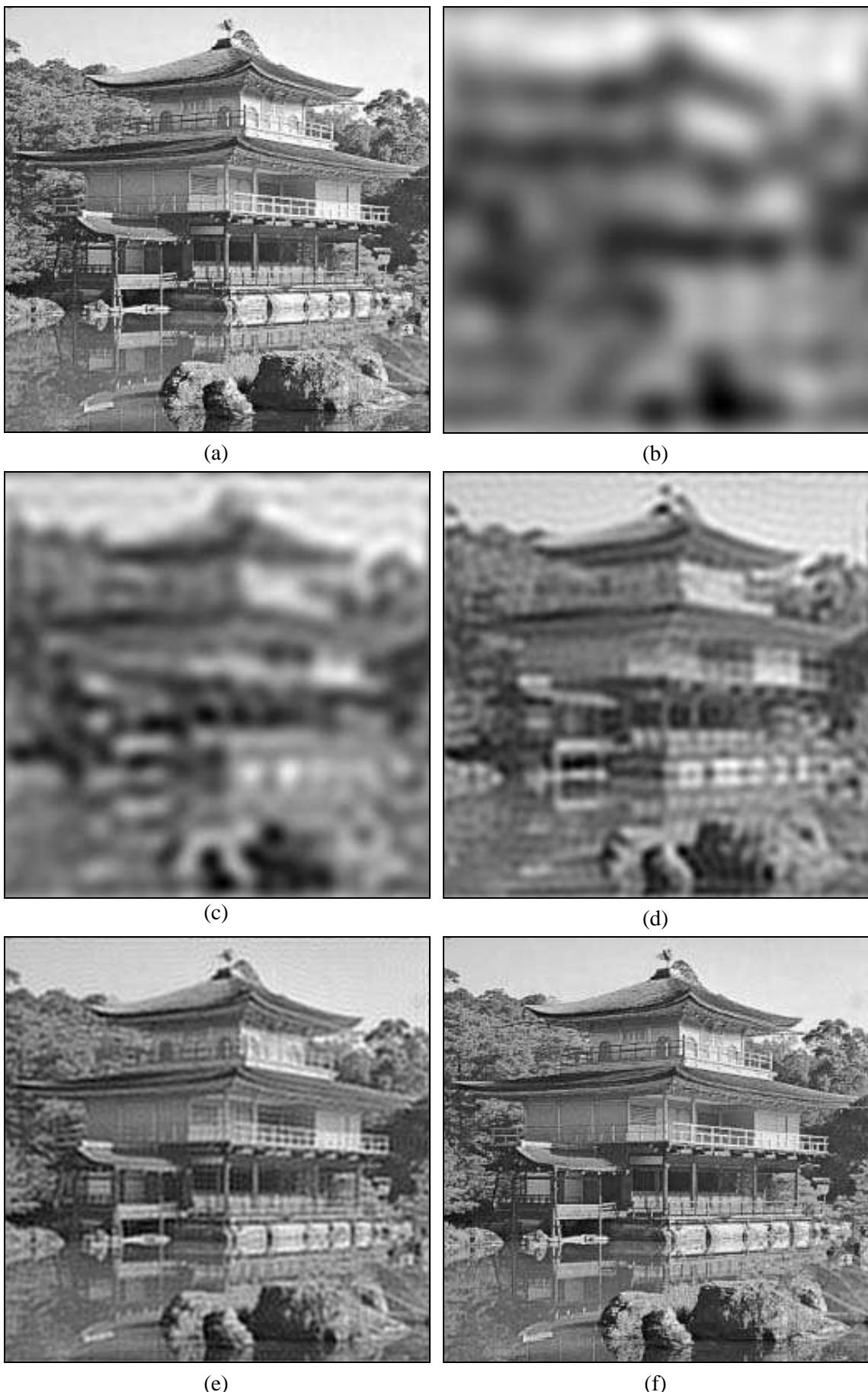


Figura 23 - (a) Imagem original; (b) - (f) resultados da aplicação de filtro passa-baixas ideal com freqüência de corte correspondente aos raios indicados na figura 22(b), a saber: 8, 16, 32, 64 e 128 pixels.

Muito embora a abrupta transição entre banda de passagem e banda de rejeição do filtro passa-baixas ideal não possa ser implementada fisicamente utilizando componentes eletrônicos, ela pode certamente ser simulada por computador. A figura 22 mostra uma imagem de 256 x 256 pixels e seu espectro de Fourier, indicando sobre ele diversos anéis cujos raios são proporcionais às freqüências de corte dos filtros passa-baixas correspondentes. Os raios escolhidos neste caso foram: 8, 16, 32, 64 e 128 pixels, correspondendo respectivamente a 7,8%, 13,5%, 23,9%, 45,9% e 90,9% da informação contida no espectro original. Quanto menor o raio, menor a freqüência de corte e, portanto, maior o grau de borramento da imagem resultante. A figura 23 apresenta exemplos de filtros passa-baixas de diferentes freqüências de corte aplicados a uma mesma imagem original.

Filtro passa-baixas Butterworth

Um filtro passa-baixas realizável em hardware é o filtro Butterworth. Um filtro Butterworth de ordem n e com freqüência de corte a uma distância D_0 da origem possui função de transferência dada pela equação

$$H(u, v) = \frac{1}{1 + [D(u, v) / D_0]^{2n}} \quad (4.62)$$

onde $D(u, v)$ é dado pela eq. (4.61). A figura 24 mostra a resposta em freqüência de um filtro passa-baixas 2-D Butterworth, $H(u, v)$, tanto em perspectiva 3-D (a) quanto em corte (b).

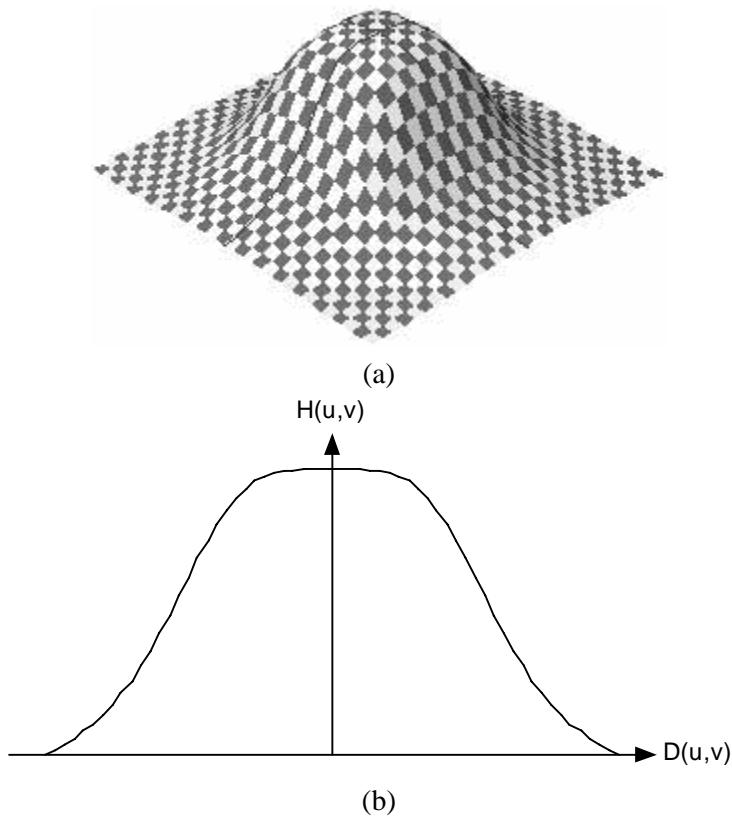


Figura 24 - Resposta em freqüência de um filtro passa-baixas Butterworth.

Ao contrário do filtro passa-baixas ideal, o filtro de Butterworth não possui uma transição abrupta entre banda de passagem e banda de rejeição e, portanto, é necessário estabelecer alguma convenção para determinar o valor exato da freqüência de corte do filtro. Um valor comumente usado para determinar quando $D(u, v) = D_0$ é 0,707 do valor máximo de $H(u, v)$.

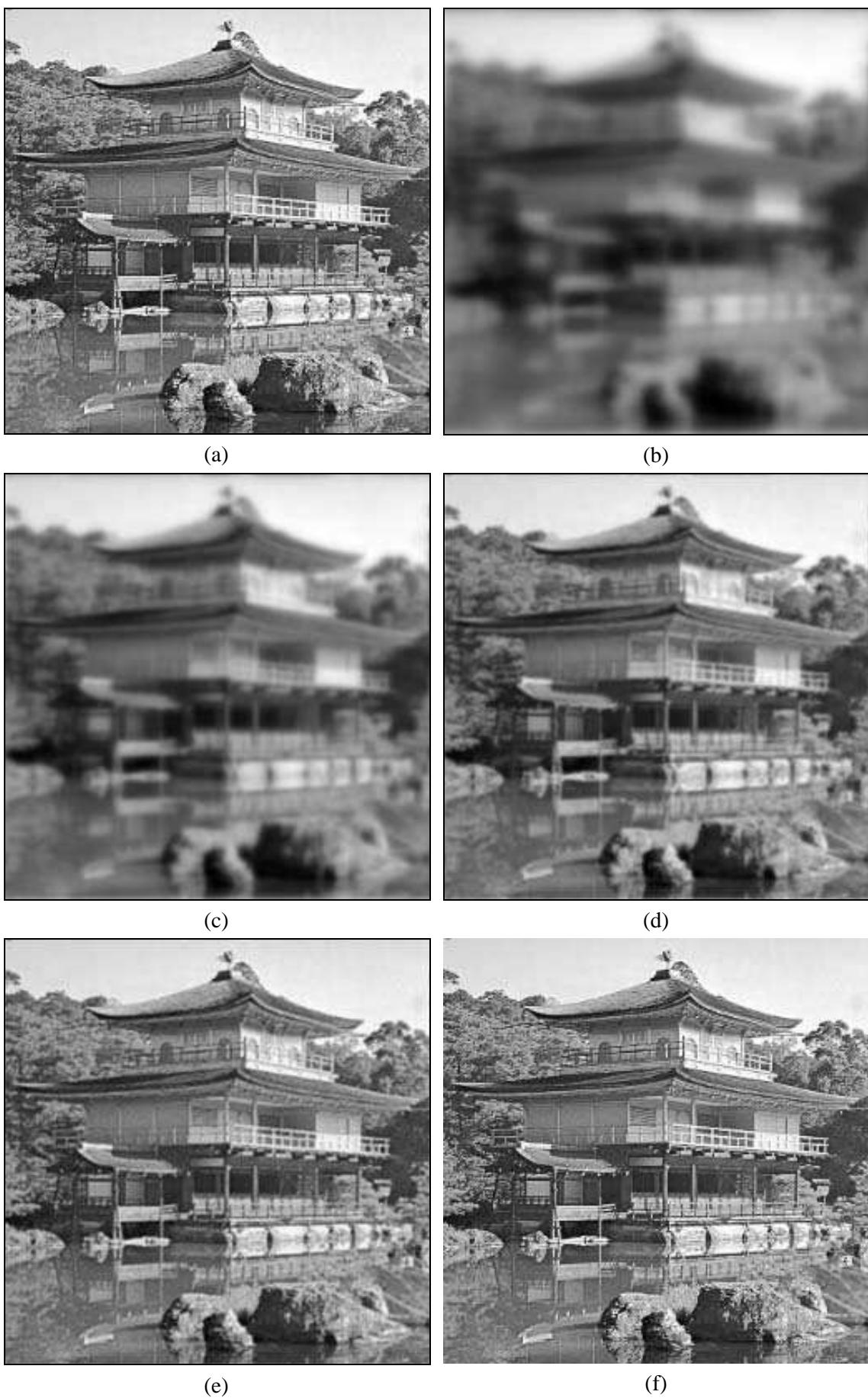


Figura 25 - (a) Imagem original; (b) - (f) resultados da aplicação de filtro passa-baixas Butterworth com freqüência de corte correspondente aos raios indicados na figura 22(b), a saber: 8, 16, 32, 64 e 128 pixels.

A figura 25 apresenta exemplos de filtros passa-baixas Butterworth de diferentes freqüências de corte aplicados a uma mesma imagem original. Através dela é possível perceber que a redução de informação obtida com um filtro Butterworth é significativamente menor que aquela obtida com um filtro ideal de mesmo raio.

A figura 26 mostra um exemplo de utilização de filtro passa-baixas Butterworth de 2^a ordem para redução do efeito indesejável de falsos contornos (ver Seção 2.1). Já a figura 27 ilustra o uso de um FPB Butterworth semelhante, desta vez para remoção de ruído.

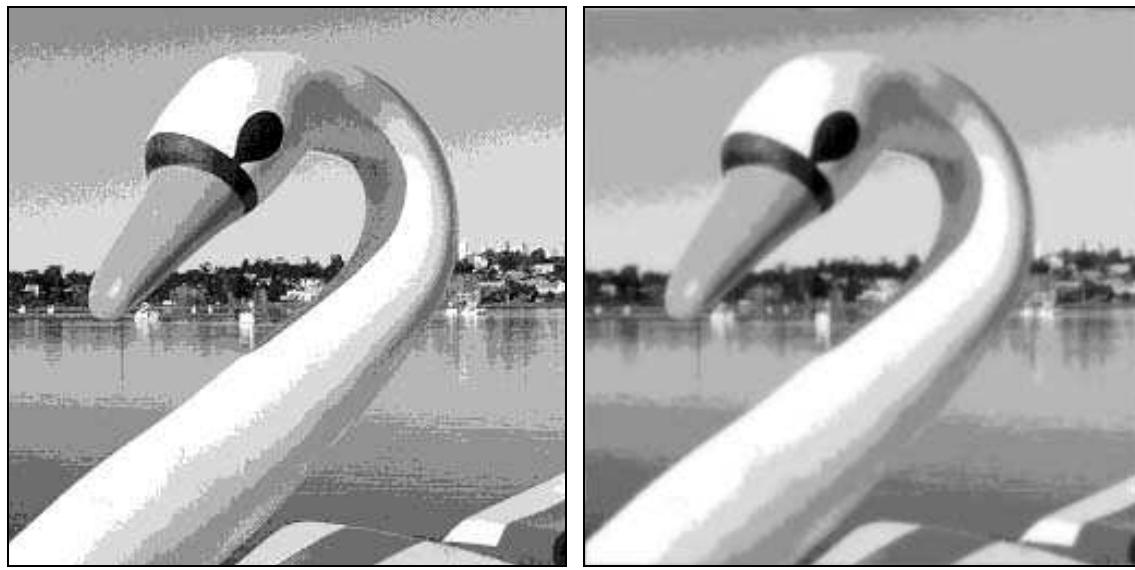


Figura 26 - Exemplo de aplicação de filtragem passa-baixas para suavização do efeito de falsos contornos.

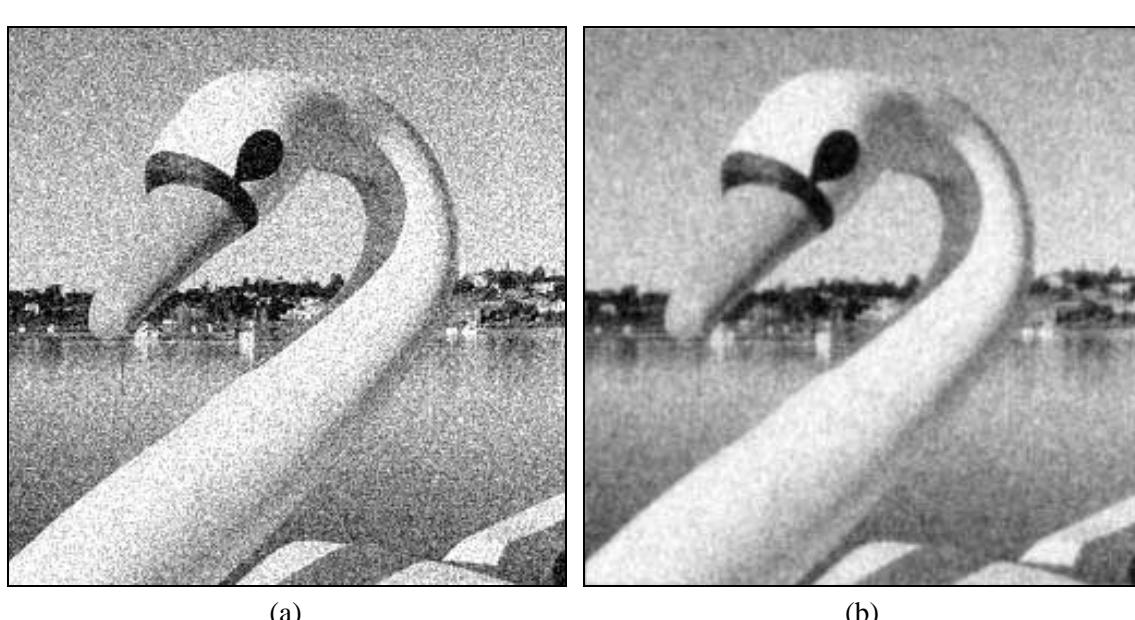


Figura 27 - Exemplo de aplicação de filtragem passa-baixas para redução de ruído em imagens.

4.5.2 Filtro passa-altas (FPA)

O objetivo do uso de filtros passa-altas em imagens é o realce de suas regiões de alta freqüência, tais como bordas e/ou texturas ricas em variações abruptas de níveis de cinza. Para o projeto de

filtros passa-altas no domínio da freqüência, aplicam-se as mesmas considerações feitas para os filtros passa-baixas, com a exceção, óbvia, do comportamento em freqüência desejado.

Filtro passa-altas ideal

Um filtro passa-altas 2-D ideal é aquele cuja função de transferência satisfaz a relação

$$H(u, v) = \begin{cases} 0 & \text{se } D(u, v) \leq D_0 \\ 1 & \text{se } D(u, v) > D_0 \end{cases} \quad (4.63)$$

onde D_0 é a 'distância de corte' do filtro e $D(u, v)$ é a distância do ponto (u, v) à origem do plano de freqüência, dada pela eq. (4.61).

A figura 28 mostra a resposta em freqüência de um filtro passa-altas 2-D ideal, $H(u, v)$, tanto em perspectiva 3-D (a) quanto em corte (b). Assim como o FPB ideal, o FPA ideal não é fisicamente realizável.

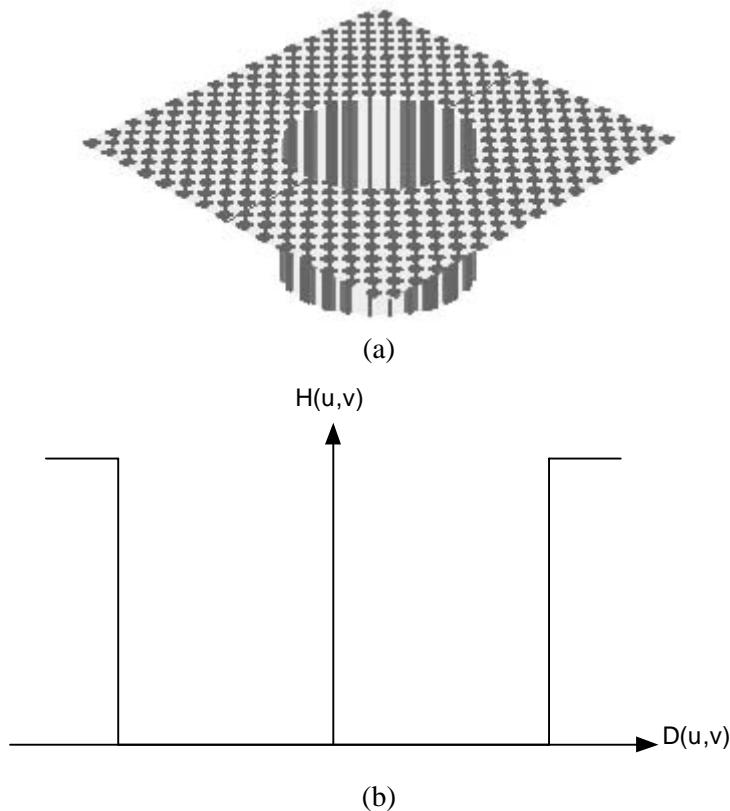


Figura 28 - Resposta em freqüência de um filtro passa-altas ideal.

Filtro passa-altas Butterworth

Um FPA Butterworth de ordem n e com freqüência de corte a uma distância D_0 da origem possui função de transferência dada pela equação

$$H(u, v) = \frac{1}{1 + [D_0 / D(u, v)]^{2n}} \quad (4.64)$$

onde $D(u, v)$ é dado pela eq. (4.61). A figura 29 mostra a resposta em freqüência de um FPA 2-D Butterworth, $H(u, v)$, tanto em perspectiva 3-D (a) quanto em corte (b). A exemplo do FPB

Butterworth, um valor comumente usado para determinar a freqüência de corte de um FPA Butterworth é 0,707 do valor máximo de $H(u,v)$.

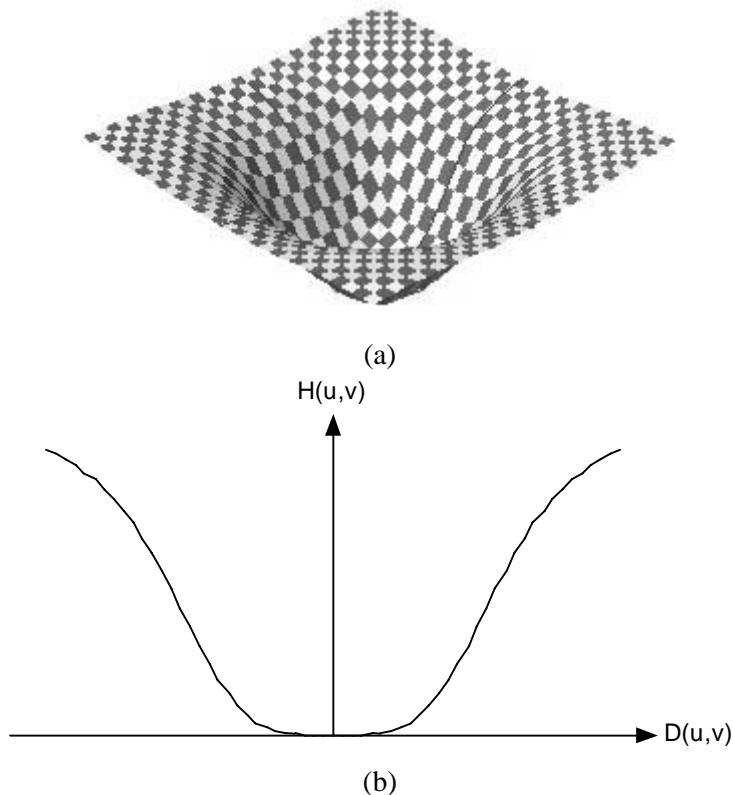


Figura 29 - Resposta em freqüência de um filtro passa-altas Butterworth.

A filtragem passa-altas usando um filtro Butterworth apresenta como desvantagem a excessiva atenuação dos componentes de baixa freqüência. Este problema é solucionado com a técnica denominada 'ênfase em alta freqüência', que consiste basicamente em adicionar uma constante à função de transferência do filtro passa-altas de modo a preservar os componentes de baixa freqüência (ver Seção 4.3). A técnica de ênfase em alta freqüência sozinha não produz um resultado muito melhor que o da filtragem passa-altas convencional; porém, a aplicação da ênfase em alta freqüência seguida da equalização de histograma pode produzir melhores resultados, como ilustra a figura 30.

4.5.3 Filtragem homomórfica

O modelo iluminância-refletância apresentado na Seção 2.1 pode ser usado como base para uma técnica de filtragem no domínio da freqüência que é útil para aprimorar a qualidade de uma imagem através da compressão da faixa dinâmica de brilho simultaneamente com o aumento de contraste.

A formulação matemática dos filtros homomórficos parte da equação que relaciona uma imagem $f(x,y)$ com suas componentes de iluminância e refletância:

$$f(x,y) = i(x,y)r(x,y) \quad (4.65)$$

Utilizando propriedades de logaritmos podemos definir uma função $z(x,y)$ dada por:

$$\begin{aligned} z(x,y) &= \ln f(x,y) \\ &= \ln i(x,y) + \ln r(x,y). \end{aligned} \quad (4.66)$$

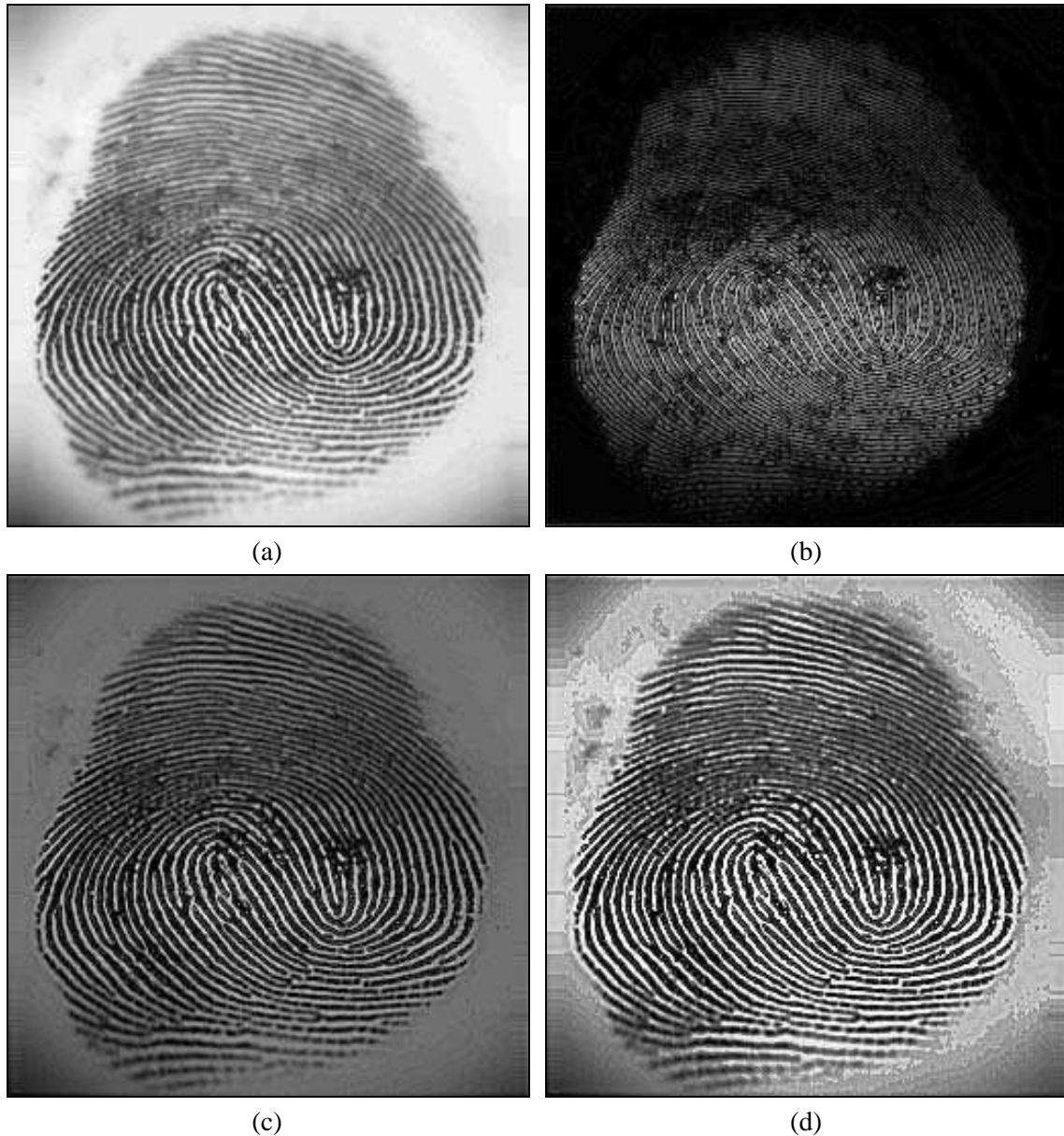


Figura 30 - Exemplo de filtragem passa-altas: (a) imagem original; (b) imagem processada com filtro Butterworth passa-altas; (c) resultado da ênfase em alta-freqüência; (d) ênfase em alta freqüência seguida de equalização de histograma.

Então, aplicando a propriedade distributiva da FT:

$$\begin{aligned}\Im\{z(x, y)\} &= \Im\{\ln f(x, y)\} \\ &= \Im\{\ln i(x, y)\} + \Im\{\ln r(x, y)\}.\end{aligned}\tag{4.67}$$

ou

$$Z(u, v) = I(u, v) + R(u, v).\tag{4.68}$$

onde $I(u, v)$ e $R(u, v)$ são as FTs de $\ln i(x, y)$ e de $\ln r(x, y)$, respectivamente.

Se $Z(u, v)$ for processada por um filtro de função de transferência $H(u, v)$, a transformada de Fourier do resultado, $S(u, v)$, será:

$$\begin{aligned} S(u,v) &= H(u,v)Z(u,v) \\ &= H(u,v)I(u,v) + H(u,v)R(u,v). \end{aligned} \quad (4.69)$$

No domínio espacial,

$$\begin{aligned} s(x,y) &= \mathfrak{J}^{-1}\{S(u,v)\} \\ &= \mathfrak{J}^{-1}\{H(u,v)I(u,v)\} + \mathfrak{J}^{-1}\{H(u,v)R(u,v)\}. \end{aligned} \quad (4.70)$$

Denominando

$$i'(x,y) = \mathfrak{J}^{-1}\{H(u,v)I(u,v)\} \quad (4.71)$$

e

$$r'(x,y) = \mathfrak{J}^{-1}\{H(u,v)R(u,v)\}. \quad (4.72)$$

podemos representar a eq. (4.70) sob a forma

$$s(x,y) = i'(x,y) + r'(x,y). \quad (4.73)$$

Finalmente, como $z(x,y)$ foi obtida extraíndo-se o logaritmo natural da imagem original $f(x,y)$, a operação inversa fornecerá à saída a imagem filtrada $g(x,y)$. Este método de filtragem está resumido na figura 31.

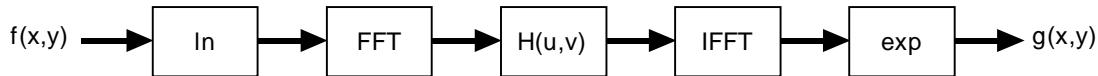


Figura 31 - Diagrama em blocos ilustrativo da filtragem homomórfica.

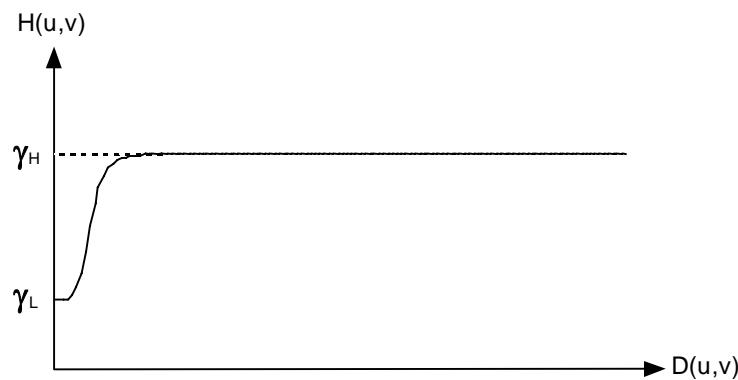


Figura 32 - Vista em corte da função de transferência $H(u,v)$ de um filtro homomórfico simétrico.

Na escolha da função de transferência $H(u,v)$ costuma-se buscar uma resposta em freqüência que atenue as baixas freqüências (associadas à componente de iluminância) e realce as altas freqüências (associadas à componente de refletância). A figura 32 mostra em corte um exemplo de função de transferência $H(u,v)$ em função de $D(u,v)$ (distância a partir da origem do par de coordenadas (u,v)). A especificação completa de $H(u,v)$ é obtida rotacionando a seção

transversal 360° ao redor do eixo vertical. Se os parâmetros γ_H e γ_L forem escolhidos de tal maneira que $\gamma_H > 1$ e $\gamma_L < 1$, o filtro resultante tenderá a atenuar as componentes de baixa freqüência e realçar as de alta freqüência, provocando uma compressão da faixa dinâmica de brilho simultaneamente com o aumento de contraste. O resultado da aplicação de um filtro homomórfico a uma imagem é ilustrado na figura 33.



Figura 33 - Exemplo de filtragem homomórfica: (a) imagem original; (b) resultado da filtragem homomórfica com raio 8/128, $\gamma_H = 1,3$ e $\gamma_L = 0,7$.

Leitura complementar

Diversos outros filtros no domínio da freqüência podem ser encontrados em [Lim 1990], [Pratt 1991] e [Jain 1989].

4.6 Processamento de imagens coloridas

O uso de cores em processamento digital de imagens decorre de dois fatores motivantes principais:

1. Na análise automática de imagens (reconhecimento de padrões), a cor é um poderoso descritor das propriedades de um objeto, que pode simplificar sua identificação e segmentação.
2. Na análise de imagens com intervenção humana, o olho humano pode discernir milhares de nuances de cores de diferentes matizes e intensidades, enquanto sua capacidade de distinguir diferentes tons de cinza não passa de algumas poucas dezenas de tons diferentes.

O processamento de imagens coloridas pode ser dividido em duas frentes principais: o processamento *full color*, onde as imagens já são adquiridas através de sensores em cores e a *pseudocolorização*, processo pelo qual são atribuídas cores diferentes a distintas regiões da escala de cinza de uma imagem monocromática.

O uso de técnicas de processamento de imagens coloridas *full color* é relativamente recente e sobre ele ainda há comparativamente pouca bibliografia. Trata-se, porém, de área importante e promissora para os próximos anos.

4.6.1 Conceitos básicos

Embora o processo psicofisiológico de percepção de cor pelo sistema nervoso central humano ainda não seja totalmente compreendido, os aspectos físicos da cor vêm sendo estudados há muitos anos por inúmeros cientistas e engenheiros, constituindo hoje um sólido conjunto de conhecimentos teóricos.

Em 1666, Sir Isaac Newton descobriu que um prisma de vidro atravessado pela luz branca é capaz de decompô-la em um amplo espectro de cores que vão do violeta, num extremo, ao vermelho, no outro. Este espectro, com alguns valores representativos de comprimento de onda, está ilustrado na figura 34. Como se pode perceber a partir desta figura, o chamado 'espectro de luz visível' ocupa uma faixa muito estreita do espectro total de radiações eletromagnéticas.

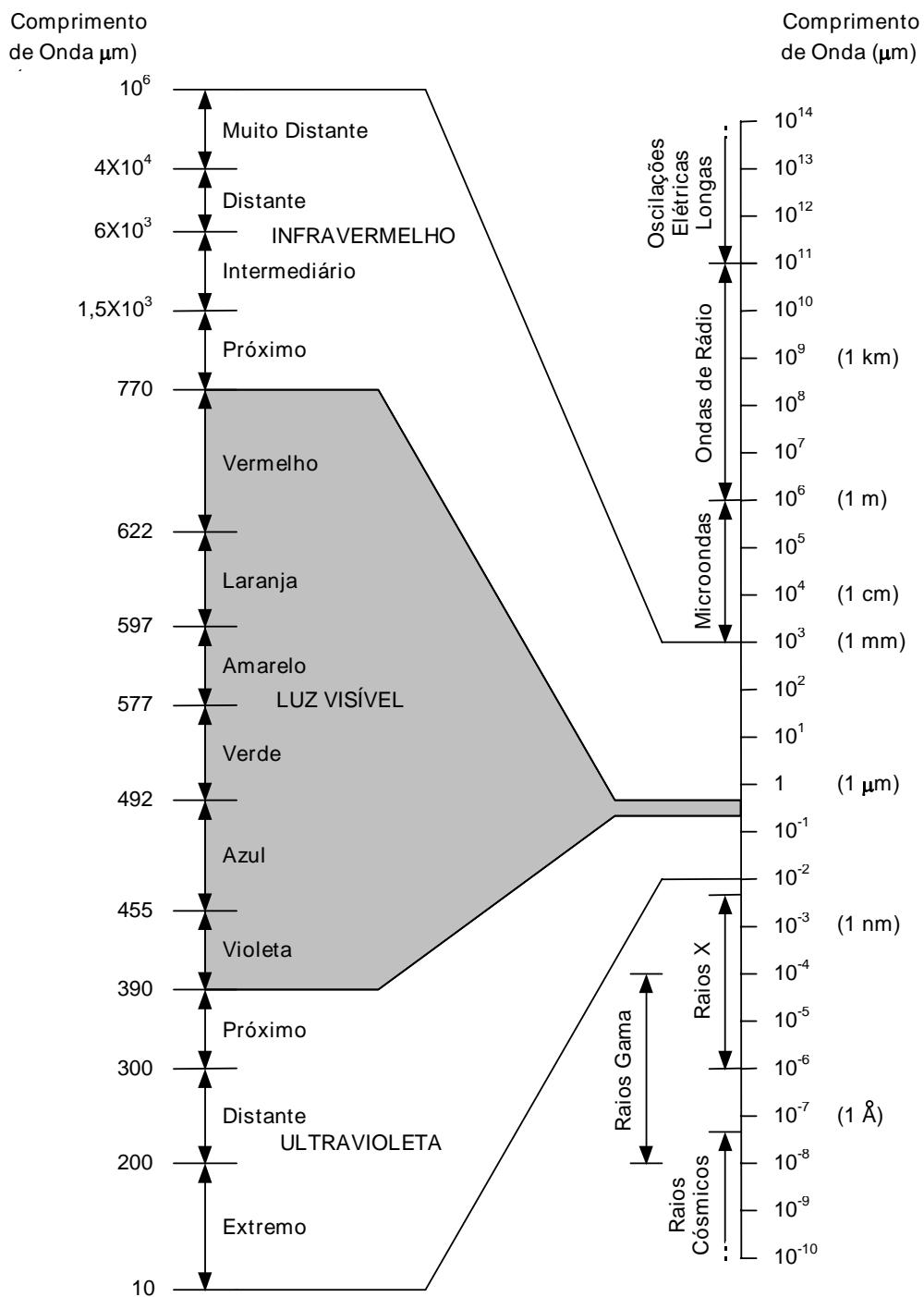


Figura 34 - Espectro eletromagnético, com destaque para as subdivisões da região de luz visível.

Para a cor ser vista, é necessário que o olho seja atingido por energia eletromagnética. Vemos um objeto através da luz refletida por ele. Se ele parece verde à luz do dia é porque, embora seja banhado pela luz branca do sol, ele reflete somente a parte verde da luz para os nossos olhos; o restante do espectro é absorvido. A teoria de percepção cromática pelo olho humano baseia-se em hipótese formulada por Young em 1801, que estabelece que os cones (células fotossensíveis que compõem a retina juntamente com os bastonetes) se

subdividem em três classes, com diferentes máximos de sensibilidade situados em torno do vermelho (*R - Red*), do verde (*G - Green*) e do azul (*B - Blue*). Desta forma, todas as sensações de cor percebidas pelo olho humano são na verdade combinações das intensidades dos estímulos recebidos por cada um destes tipos de cones. Estas três cores são denominadas cores primárias aditivas, pois é possível obter qualquer outra cor a partir de uma combinação aditiva de uma ou mais delas, em diferentes proporções. Para efeito de padronização, o CIE (*Commission Internationale de l'Eclairage* - Comissão Internacional de Iluminação) atribuiu, em 1931, os seguintes comprimentos de onda a estas cores primárias: azul = 435,8 nm, verde = 546,1 nm, vermelho = 700 nm. (1 nm = 10^{-9} m)

As combinações de cores consideradas até o momento pressupõem a emissão de radiações coloridas situadas dentro do espectro de luz visível, as quais combinadas aditivamente produzem um efeito final correspondente à soma dos efeitos individuais. A mistura das cores primárias, duas a duas, produz as chamadas cores secundárias, que são: magenta (*R+B*), amarelo (*R+G*) e ciano ou turquesa (*G+B*). A mistura das três cores primárias ou de uma secundária com sua cor primária 'oposta' produz a luz branca, como se pode ver na figura 35(a) (ver Seção *Figuras Coloridas*). Há uma outra classe de combinação de cores, usada por exemplo em impressoras coloridas, onde as cores primárias estão associadas aos pigmentos magenta, ciano e amarelo, que combinados de forma subtrativa produzem as cores secundárias vermelho, verde e azul. Estas combinações são chamadas subtrativas porque cada pigmento, ao ser depositado em fundo branco, subtrai parte da luz branca incidente, refletindo apenas a cor correspondente ao pigmento. Ao contrário da combinação aditiva, na subtrativa a união das três cores primárias ou de uma secundária com sua primária oposta produz o preto, como se vê na figura 35(b) (ver Seção *Figuras Coloridas*). Deste ponto em diante, somente faremos referência a cores primárias e combinações aditivas.

Um exemplo clássico de dispositivo que opera sob o princípio da combinação aditiva de cores é o monitor de vídeo, que possui em sua superfície pontos triangulares compostos de fósforos sensíveis a cada uma das cores primárias. Cada tipo de fósforo de cada ponto da tela é bombardeado por um feixe eletrônico cuja intensidade é proporcional à quantidade de vermelho, verde ou azul naquele ponto da imagem que se deseja representar. As componentes de *R*, *G* e *B* de cada tríade de fósforo são 'adicionadas' pelos cones do olho humano e a cor correspondente é então percebida.

As três características normalmente utilizadas para distinguir as cores entre si são: brilho (*B - brightness*), matiz (*H - hue*) e saturação (*S - saturation*). O brilho representa a noção de intensidade luminosa da radiação, o matiz é uma propriedade associada ao comprimento de onda predominante na combinação das várias ondas visíveis, enquanto a saturação expressa a pureza do matiz ou, em outras palavras, o grau de mistura do matiz original com a luz branca. Cores como o rosa e o vermelho, por exemplo, têm o mesmo matiz, mas apresentam diferentes graus de saturação.

O matiz e a saturação costumam ser denominados conjuntamente de cromaticidade, o que nos permite dizer que uma cor pode ser definida pelo seu brilho e por sua cromaticidade. Os percentuais de vermelho, verde e azul presentes em uma cor recebem o nome de coeficientes tricromáticos e são dados pelas equações:

$$r = \frac{R}{R + G + B} \quad (4.74)$$

$$g = \frac{G}{R + G + B} \quad (4.75)$$

e

$$b = \frac{B}{R + G + B} \quad (4.76)$$

onde R , G e B representam a quantidade de luz vermelha, verde e azul, respectivamente, normalizada entre 0 e 1. Logo, a soma dos três coeficientes tricromáticos é:

$$r + g + b = 1 \quad (4.77)$$

Os coeficientes tricromáticos exatos de cada cor são computados a partir do Diagrama de Cromaticidade publicado pelo CIE.

4.6.2 Modelos de representação de cores

O objetivo dos modelos de cores é permitir a especificação de cores em um formato padronizado e aceito por todos. Em linhas gerais, um modelo de cores é uma representação tridimensional na qual cada cor é representada por um ponto no sistema de coordenadas 3-D. A maioria dos modelos em uso atualmente é orientada ao hardware (impressoras ou monitores coloridos, por exemplo) ou a aplicações que utilizam manipulação de cores (como os vários títulos de software comentados no capítulo 7). Os modelos mais utilizados para representação de cores são: *RGB* (*red, green, blue*), *CMY* (*cyan, magenta, yellow*), *CMYK* (variante do modelo *CMY*, onde *K* denota *black*), *YCbCr* (padrão normalizado pela recomendação ITU-R BT.601 e utilizado em várias técnicas de compressão de vídeo), *YIQ* (padrão NTSC de TV em cores) e *HSI* (*hue, saturation, intensity*), às vezes também denominado *HSV* (*hue, saturation, value*). Apresentamos a seguir mais detalhes sobre os padrões *RGB*, *CMY*, *YIQ* e *HSI*.

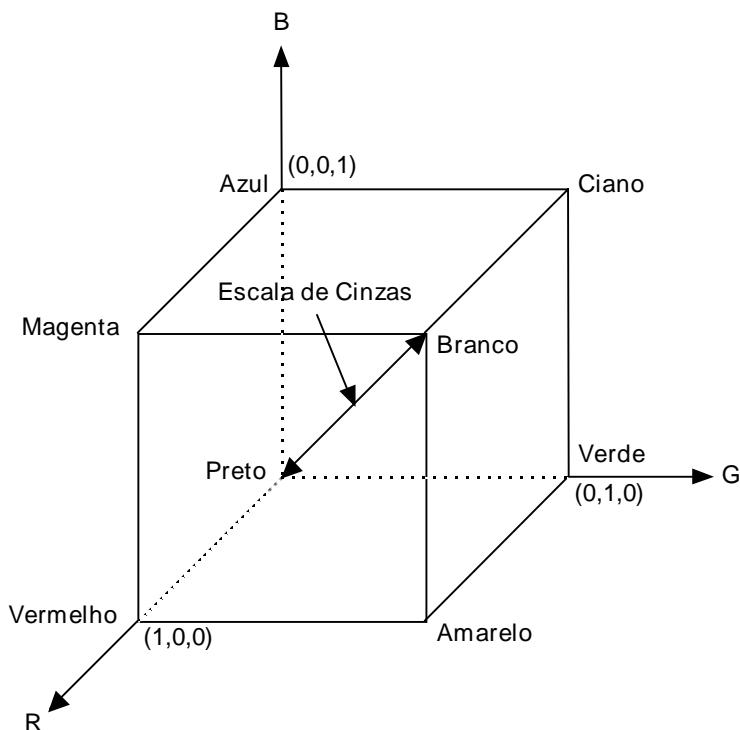


Figura 36 - Modelo *RGB*.

Modelo *RGB*

O modelo *RGB* é baseado em um sistema de coordenadas cartesianas, que pode ser visto como um cubo onde três de seus vértices são as cores primárias, outros três as cores secundárias, o vértice junto à origem é o preto e o mais afastado da origem corresponde à cor branca, conforme ilustra a figura 36. Neste modelo, a escala de cinza se estende através de uma linha (a diagonal do cubo) que sai da origem (preto) até o vértice mais distante dela (branco). Por conveniência, geralmente assume-se que os valores máximos de R , G e B estão normalizados na faixa de 0 a 1. O modelo *RGB* é o mais utilizado por câmeras e monitores de vídeo.

O modelo CMY

Este modelo é baseado nos pigmentos primários ciano, magenta e amarelo. A maioria dos dispositivos que opera sob o princípio da deposição de pigmentos coloridos em papel (como impressoras ou fotocopiadoras coloridas, por exemplo) requer uma conversão interna do formato *RGB* para o formato *CMY*. Esta conversão é simples e consiste na equação:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (4.78)$$

onde todos os valores envolvidos estão normalizados no intervalo [0, 1]. A eq. (4.78) permite a dedução da relação oposta, que, contudo, não é de interesse prático.

O modelo *YIQ*

O modelo *YIQ* é utilizado no padrão NTSC de TV em cores. Basicamente, o modelo *YIQ* foi desenvolvido sob o princípio da dupla compatibilidade, que norteou os projetos de TV colorida para garantir a convivência entre o sistema colorido e o sistema preto e branco (P&B) já existente. A componente *Y* (luminância) contém a informação necessária para um receptor P&B reproduzir a imagem monocromática correspondente, enquanto as componentes *I* e *Q* codificam as informações de cromaticidade. A conversão de *RGB* para *YIQ* pode ser obtida pela equação:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0,299 & 0,587 & 0,114 \\ 0,596 & -0,275 & -0,321 \\ 0,212 & -0,523 & 0,311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (4.79)$$

A principal vantagem do modelo *YIQ* é sua capacidade de permitir a separação entre a componente de brilho (*Y*) e as componentes de cromaticidade (*I* e *Q*). Uma técnica como a equalização de histograma (ver Seção 3.3), por exemplo, pode ser aplicada à componente *Y* de uma imagem colorida, com o intuito de aprimorar seu contraste sem distorcer as informações de matiz e saturação da imagem original.

O modelo *HSI*

O modelo *HSI* é de grande interesse, uma vez que permite separar as componentes de matiz, saturação e intensidade da informação de cor em uma imagem, da forma como o ser humano as percebe. Sua utilização é mais intensa em sistemas de visão artificial fortemente baseados no modelo de percepção de cor pelo ser humano, como por exemplo um sistema automatizado de colheita de frutas, em que é preciso determinar se a fruta está suficientemente madura para ser colhida a partir de sua coloração externa. Geometricamente, o modelo *HSI* pode ser visto como um sólido, indicado na figura 37(b), cujos cortes horizontais produzem triângulos (figura 37(a)) nos quais os vértices contêm as cores primárias e o centro corresponde à combinação destas cores em iguais proporções. Esta combinação estará mais próxima do preto ou do branco, conforme a altura em que o corte tenha sido efetuado.

A conversão entre os modelos *RGB* e *HSI* utiliza equações razoavelmente mais complexas, cuja dedução foge ao escopo desta obra. Em resumo, a conversão de *RGB* para *HSI* pode ser obtida através das equações:

$$I = \frac{1}{3}(R + G + B) \quad (4.80)$$

$$S = 1 - \frac{3}{R+G+B} [\min(R, G, B)] \quad (4.81)$$

e

$$H = \cos^{-1} \left\{ \frac{\frac{1}{2}[(R-G)+(R-B)]}{[(R-G)^2 + (R-B)(G-B)]^{1/2}} \right\} \quad (4.82)$$

onde, se $(B/I) > (G/I)$, deve-se fazer $H = 360^\circ - H$. A fim de normalizar a faixa de matiz, deve-se fazer $H = H/360^\circ$.

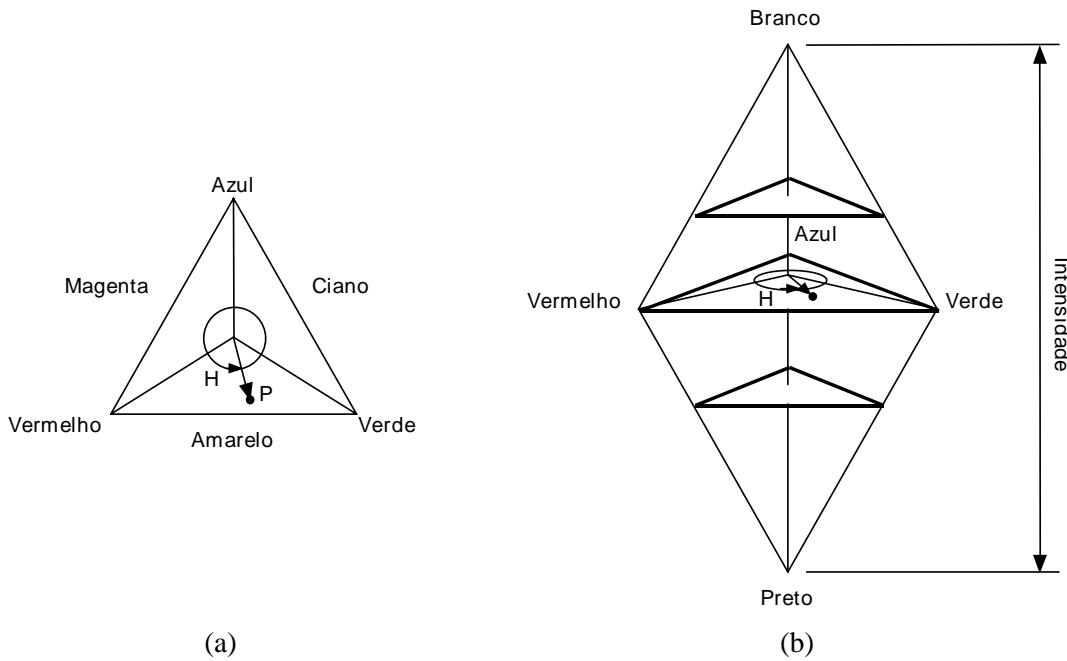


Figura 37 - Modelo HSI.

A conversão de H , S e I para r , g e b é mais complexa por depender do intervalo de valores de H , conforme as equações a seguir:

Para $0^\circ < H \leq 120^\circ$:

$$b = \frac{1}{3}(1-S) \quad (4.83)$$

$$r = \frac{1}{3} \left[1 + \frac{S \cos H}{\cos(60^\circ - H)} \right] \quad (4.84)$$

$$g = 1 - (r + b) \quad (4.85)$$

Para $120^\circ < H \leq 240^\circ$:

$$H = H - 120^\circ \quad (4.86)$$

$$r = \frac{1}{3}(1 - S) \quad (4.87)$$

$$g = \frac{1}{3} \left[1 + \frac{S \cos H}{\cos(60^\circ - H)} \right] \quad (4.88)$$

e

$$b = 1 - (r + g) \quad (4.89)$$

Para $240^\circ < H \leq 360^\circ$:

$$H = H - 240^\circ \quad (4.90)$$

$$g = \frac{1}{3}(1 - S) \quad (4.91)$$

$$b = \frac{1}{3} \left[1 + \frac{S \cos H}{\cos(60^\circ - H)} \right] \quad (4.92)$$

e

$$r = 1 - (b + g) \quad (4.93)$$

Os valores obtidos de r , g e b podem ser convertidos em R , G e B conforme as equações (4.74) a (4.76).

A figura 38 (ver Seção *Figuras Coloridas*) mostra um exemplo de imagem colorida decomposta em suas componentes R , G e B . A mesma imagem aparece decomposta em H , S e I na figura 39 (ver Seção *Figuras Coloridas*) e em suas componentes Y , I e Q na figura 40 (ver Seção *Figuras Coloridas*).

4.6.3 Pseudocolorização

É a técnica através da qual se atribuem cores a imagens monocromáticas com base na distribuição de níveis de cinza da imagem original. A técnica mais simples e difundida de pseudocolorização é conhecida na literatura como *intensity* (ou também *density*) *slicing* e pode ser entendida com o auxílio da figura 41. Interpretando a imagem monocromática original como uma função de intensidade 2-D, este método define planos de corte que interceptam ('fatiam') a imagem original em diferentes pontos acima do plano xy . Na figura 41 mostramos o caso específico de um plano de corte (fatia) situado à altura l_i em relação ao plano da imagem. Cada lado do plano mostrado receberá uma cor diferente. O resultado será uma imagem de duas cores cuja aparência pode ser interativamente controlada movendo-se o plano de corte para cima ou para baixo.

Extrapolando-se o raciocínio para M planos e definindo os níveis l_1, l_2, \dots, l_M , onde l_0 representa o preto na imagem original [$f(x,y) = 0$] e l_L o branco [$f(x,y) = L$], podemos interpretar o processo de fatiamento como sendo a divisão da escala de cinza da imagem original em $M + 1$ regiões ($0 < M < L$), nas quais a atribuição de cor é feita segundo a relação

$$f(x, y) = c_k \quad \text{se } f(x, y) \in R_k \quad (4.94)$$

onde c_k é a cor associada à k -ésima região R_k .

A pseudocolorização também pode ser interpretada no domínio bidimensional como sendo uma função de mapeamento em forma de escada, onde cada degrau corresponde a uma gama de valores de tons de cinza na imagem original que mapeiam em uma determinada cor na imagem pseudocolorizada.

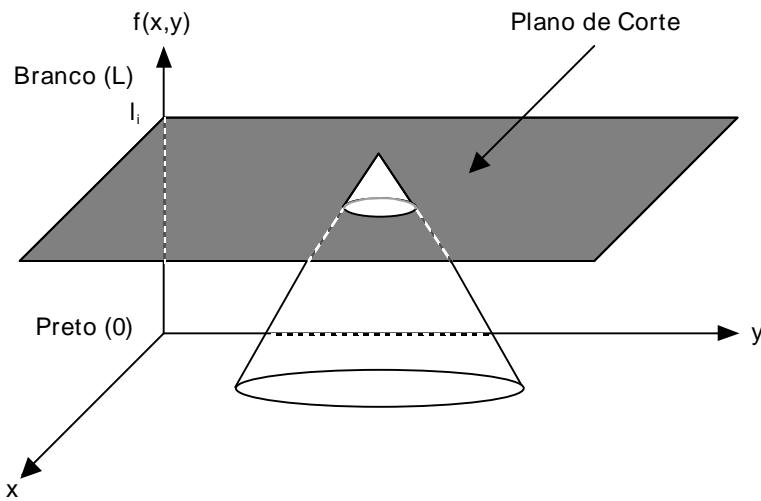


Figura 41 - *Intensity slicing*.

4.6.4 Processamento de imagens coloridas *full color*

Conforme indicamos no início desta seção, o processamento de imagens coloridas ainda é algo relativamente recente e que ganha interesse a cada dia, na medida em que a disponibilidade de melhor hardware a menor custo passa a viabilizar implementações que até poucos anos atrás poderiam ser proibitivas do ponto de vista financeiro e/ou computacional.

Os modelos de cores mais adequado para o processamento *full color* são aqueles que decompõem a imagem colorida de tal maneira que a informação de luminância esteja contida em um dos componentes. É o caso do modelo *YIQ*, onde a componente *Y* contém a informação de luminância e do modelo *HSI*, onde a informação de brilho está toda contida na componente *I*. Este último modelo é ainda mais útil em aplicações que procuram reproduzir o mecanismo de percepção cromática do olho humano, conforme antecipamos na Seção 4.6.2.

A partir da decomposição da imagem colorida nas componentes adequadas, diversas técnicas existentes para imagens monocromáticas podem ser aplicadas com sucesso a imagens coloridas, como por exemplo a equalização de histograma, ilustrada na figura 42 (ver Seção *Figuras Coloridas*), que melhora o contraste da imagem da parte (a) da figura sem distorcer sua informação de cromaticidade.

Leitura complementar

Para o leitor interessado nas deduções das equações de conversão do modelo *RGB* para *HSI* e vice-versa, sugerimos o capítulo 4 de [Gonzalez e Woods 1992].

4.7 Filtros adaptativos

4.7.1 Introdução

No decorrer deste capítulo, vimos inúmeras técnicas de filtragem de imagens onde o objetivo principal é a remoção de ruído. Cada um dos filtros estudados pode operar de forma local ou global, no domínio espacial ou da freqüência. Apesar das diferenças de comportamento entre eles, existe uma característica comum a todos os filtros estudados até aqui: suas características permanecem constantes ao longo de toda a imagem que está sendo processada. Existem situações, porém, em que seria desejável que um filtro mudasse suas características conforme o trecho de imagem que estivesse percorrendo ou ainda de acordo com o tipo de ruído nela presente. Por exemplo, se o ruído presente em uma imagem tiver uma distribuição uniforme, este ruído será melhor filtrado aplicando-se um filtro da média, havendo porém uma inevitável perda de detalhes na imagem. Por outro lado, se o ruído for impulsivo, haverá maior eficácia em se aplicar um filtro da mediana. Estes fatores constituem a grande motivação para o estudo e a implementação de filtros adaptativos bidimensionais.

O filtro ideal para se usar em uma imagem é aquele que muda suas características de forma adaptativa, dependendo do conteúdo de imagem presente em uma janela local, reduzindo o ruído presente na imagem e ao mesmo tempo preservando seu conteúdo. Por exemplo, se na região percorrida por uma janela houver apenas informação de bordas, então um filtro da mediana poderá ser usado, por suas propriedades de preservação de detalhes destas bordas. Se, por outro lado, a janela estiver posicionada sobre uma região de fundo uniforme, então o filtro deveria mudar suas características de forma a atuar como um filtro da média.

O projeto de filtros adaptativos pode ser dividido em duas etapas. A primeira consiste no processo de decisão usado para determinar o tipo de filtro a ser usado, que pode ser tão simples quanto um detetor de borda ou tão elaborado quanto a determinação dos parâmetros estatísticos do ruído presente na imagem. A segunda consiste na determinação do melhor filtro a ser usado para o problema específico em questão.

4.7.2 Aspectos Estatísticos

A presença de ruído em uma imagem requer o uso de técnicas estatísticas para caracterizá-lo, comparando o histograma do ruído contido na imagem com histogramas teóricos conhecidos.

Por definição, considera-se ruído qualquer tipo de informação indesejada que obstrui a aquisição e o processamento da informação desejada. Existem muitos tipos de ruídos que podem estar presentes em imagens e estes tipos podem ser determinados pelo formato do histograma do ruído. Um tipo de ruído que comumente aparece em imagens é o ruído distribuído uniformemente, ou seja que possui um histograma uniforme. A probabilidade de um valor de ruído tendo tons de cinza entre a e b é $1/(b-a)$ e fora desta faixa é 0. Por exemplo, se $b = 200$ e $a = 100$, então o ruído uniforme estará na faixa de 100 a 200, com cada valor de tom de cinza tendo a probabilidade de 0,01 (ou 1%).

Outro tipo muito comum é o ruído com distribuição gaussiana. Este ruído é freqüentemente usado para modelar ruídos desconhecidos, devido ao Teorema do Limite Central, que estabelece que a soma de um grande número de termos representando ruídos aleatórios tende a produzir um ruído resultante do tipo gaussiano e independente dos tipos dos ruídos incluídos naquela soma. O ruído gaussiano é muito comum em imagens devido ao ruído eletrônico presente nas câmeras de vídeo. Em uma distribuição gaussiana, a probabilidade de um ruído ocorrer em um determinado tom de cinza decresce à medida que os valores de tons de cinza divergem do valor do tom de cinza presente no pico central m . A variável σ determina a largura do histograma e é conhecida como desvio padrão, enquanto que a variável m é conhecida como média.

Outro tipo comum de ruído presente em imagens que são iluminadas por laser é o ruído com distribuição exponencial negativa. Este ruído aparece porque as superfícies iluminadas por laser são geralmente irregulares comparadas com o comprimento de onda do laser. O pico do histograma está no tom de cinza igual a zero e a variável a determina quão rapidamente este histograma cai a zero.

Finalmente, existe ainda o ruído sal e pimenta, que normalmente ocorre devido a defeitos no sistema de geração da imagem. O ruído sal e pimenta contém dois níveis de cinza localizados em a e b , com probabilidade de ocorrência igual a p . A probabilidade total do ruído sal e pimenta é a soma das probabilidades para cada ruído e é dada por $2p$. Os pixels ruidosos brancos são chamados sal, enquanto os pixels de ruído preto são chamados pimenta.

A figura 43 ilustra os histogramas típicos dos quatro tipos de ruídos mais comuns em imagens digitais.

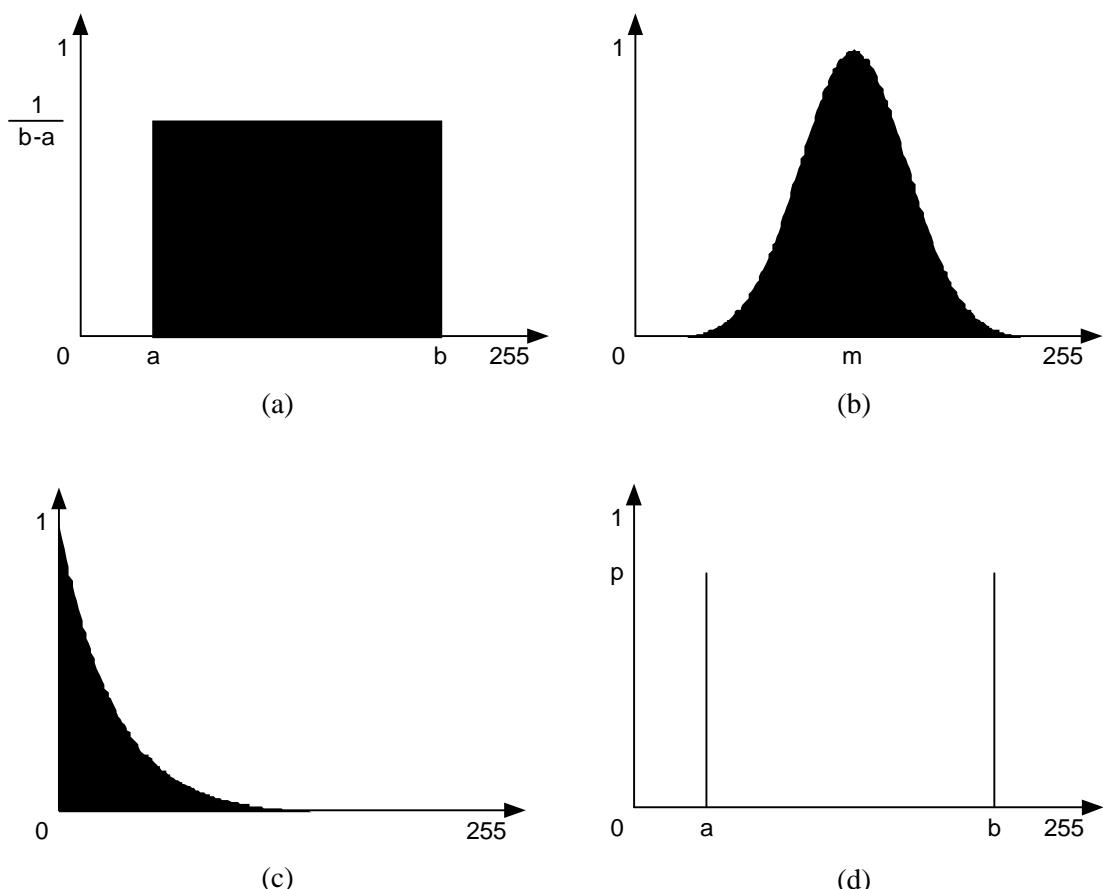


Figura 43 - Histogramas dos principais tipos de ruído: (a) ruído uniforme; (b) ruído gaussiano; (c) ruído exponencial negativo; (d) ruído sal e pimenta.

Existem duas maneiras pelas quais uma imagem pode ser corrompida por ruído. A primeira é chamada ruído aditivo, na qual simplesmente é adicionado algum tipo de ruído a uma imagem até então livre de ruído. A segunda maneira é denominada ruído multiplicativo, que consiste em multiplicar cada pixel da imagem por um termo de ruído randômico. Existem técnicas de filtragem adaptativa relatadas na literatura que operam com somente um ou outro tipo de ruído.

O objetivo de se descobrir o tipo de ruído que está presente em uma imagem é determinar o melhor filtro para reduzir este ruído. A forma de obter informações estatísticas sobre o ruído presente na imagem é normalmente um grande desafio e dele pode depender a maior ou menor eficácia do método de filtragem. A maneira mais usual de fazê-lo é procurar isolar regiões homogêneas na imagem ruidosa e comparar o histograma desta região com os vários histogramas teóricos dos tipos de ruídos mais comuns em imagens, como os mostrados na figura 43. Uma vez que o tipo de ruído tenha sido determinado, comparando-se o formato do

histograma da região ruidosa com os vários histogramas teóricos, pode-se utilizar a tabela 1 para determinar os principais parâmetros teóricos do histograma, que são seus momentos de primeira e segunda ordem, conhecidos respectivamente por média e variância.

Tabela 1 - Parâmetros teóricos dos principais tipos de histogramas de ruídos
(G_i representa o tom de cinza de um pixel)

Histograma	Nome do ruído	Momento de 1 ^a ordem	Momento de 2 ^a ordem
$h_i = \begin{cases} \frac{1}{b-a} & \text{para } a \leq G_i \leq b \\ 0 & \text{caso contrario} \end{cases}$	uniforme	$\frac{a+b}{2}$	$\frac{(a-b)^2}{12} + \frac{(a+b)^2}{4}$
$h_i = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[\frac{-(G_i - m)^2}{\sigma^2}\right]$ para $-\infty \leq G_i \leq \infty$	gaussiano	m	$\sigma^2 + m^2$
$h_i = \frac{1}{a} \exp\left[-\frac{G_i}{a}\right]$ para $0 \leq G_i \leq \infty$	exponencial negativo	a	$2a^2$

4.7.3 Alguns tipos de filtros adaptativos

Filtro de Erro Médio Quadrático Mínimo (MMSE - Minimum Mean-Square Error)

O filtro adaptativo MMSE faz uso do conhecimento da variância local para determinar se o filtro da média deve ou não ser aplicado na região onde se encontra a máscara. Este filtro apresenta melhores resultados se o ruído for do tipo aditivo. A eq. (4.95) mostra uma imagem com ruído aditivo $g(x,y)$ em função da imagem original livre de ruído $f(x,y)$ e do termo ruidoso $n(x,y)$.

$$g(x,y) = f(x,y) + n(x,y). \quad (4.95)$$

O filtro MMSE utiliza a variância do ruído juntamente com a variância local para calcular o novo valor do pixel de referência da janela, segundo a eq. (4.96).

$$r(x,y) = \left(1 - \frac{\sigma_n^2}{\sigma_1^2}\right)g(x,y) + \frac{\sigma_n^2}{\sigma_1^2}K \quad (4.96)$$

onde $r(x,y)$ é a imagem filtrada, $g(x,y)$ é a imagem ruidosa, σ_n^2 é a variância do ruído, σ_1^2 é a variância local em relação ao pixel (x,y) e K é o resultado da aplicação de um filtro da média local.

Na região de fundo de uma imagem, as variações no valor dos pixels são devidas somente ao ruído. Portanto, a variância local nesta região será aproximadamente igual à variância do ruído. Logo, o primeiro termo da eq. (4.96) será aproximadamente igual a zero e o resultado do filtro MMSE será o mesmo que seria produzido pelo filtro da média, isto é: $r(x,y) = K$.

Se a máscara for movida para uma área da imagem que contém bordas, a variância local se tornará bem maior que a variância do ruído ($\sigma_n^2 \ll \sigma_1^2$). Logo, o segundo termo da eq. (4.96)

será aproximadamente igual a zero e o resultado do filtro MMSE será o valor original do pixel, isto é: $r(x,y) = g(x,y)$.

Os casos considerados acima são casos extremos do filtro MMSE. Para os casos intermediários, uma parcela proporcional da imagem original e da saída do filtro da média local são adicionados para produzir a saída do filtro MMSE, conforme a eq. (4.96).

O filtro MMSE é bastante eficaz na redução de uma boa parcela do ruído presente em uma imagem, sem suavizar suas bordas. A figura 44 mostra exemplos de utilização do filtro MMSE a imagens contaminadas por diferentes tipos de ruído aditivo. Através dela é possível comprovar que este filtro é mais eficiente quando o ruído $n(x,y)$ é do tipo uniforme ou gaussiano e não apresenta bom desempenho frente a ruídos tipo impulsivo e sal e pimenta.

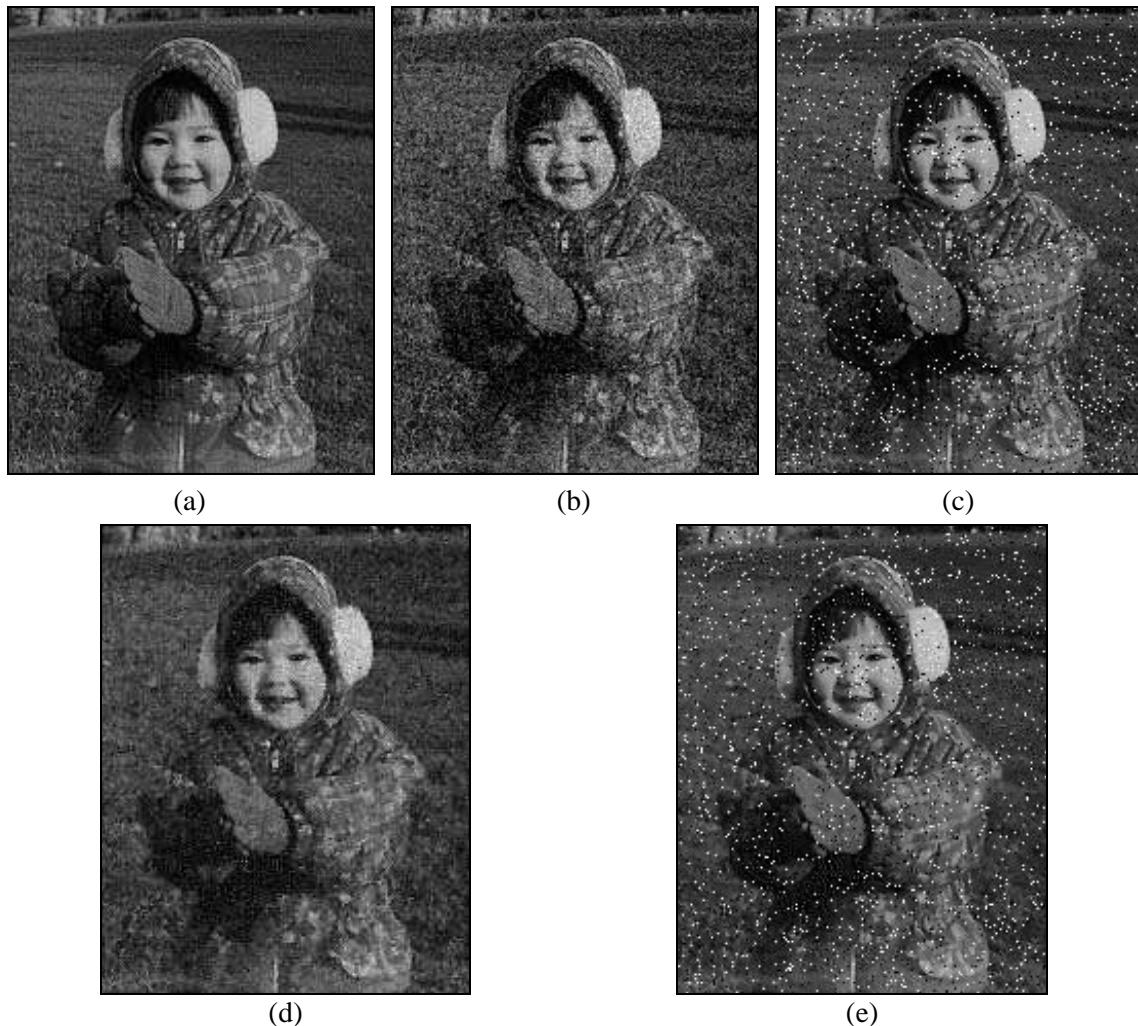


Figura 44 - (a) Imagem original; (b) imagem contaminada por ruído gaussiano; (c) imagem contaminada por ruído sal e pimenta; (d) resultado da aplicação do filtro MMSE sobre a imagem (b); (e) resultado da aplicação do filtro MMSE sobre a imagem (c).

Filtro de média e mediana com dupla janela (*DW-MTM - Double Window-Modified Trimmed Mean*)

O filtro adaptativo DW-MTM utiliza duas janelas de dimensões diferentes: primeiramente uma janela menor, na qual é aplicado o algoritmo da mediana; depois, outra janela, de maiores dimensões, onde se aplica o algoritmo da filtragem pela média apenas levando em conta os pixels situados próximos do valor da mediana anteriormente calculado.

Este filtro trabalha igualmente bem com os ruídos uniforme e gaussiano, bem como com os ruídos impulsivo e sal e pimenta.

O algoritmo para o filtro adaptativo DW-MTM é descrito a seguir. Dado um pixel localizado na posição (x,y) dentro da imagem, um filtro de mediana ($MED[g(x,y)]$) é computado dentro de uma região local de $n \times n$ ao redor da posição (x,y) . O valor da mediana computado para este filtro (MED) é usado para estimar o valor da média da área local de $n \times n$. Na seqüência, uma janela maior, igualmente centrada na posição (x,y) , de tamanho $q \times q$, é usada para calcular o valor da média, levando em conta na janela $q \times q$ somente os pixels dentro da faixa de tom de cinza situada entre $(MED - C)$ e $(MED + C)$, onde C é uma constante escolhida em função de um fator arbitrário K e do desvio padrão do ruído presente na imagem, segundo a equação:

$$C = K \cdot \sigma_n \quad (4.97)$$

A faixa típica de valores para K é de 1,5 a 2,5. Para $K = 0$, o filtro DW-MTM reduz-se a um filtro da mediana $n \times n$. Para valores muito grandes de K , o filtro DW-MTM reduz-se a um filtro da média $q \times q$. Portanto, conforme K decresce, o filtro DW-MTM filtra melhor ruídos impulsivos, mas não funciona bem na filtragem dos ruídos uniforme e gaussiano.

A figura 45 mostra exemplos de aplicação do filtro DW-MTM em imagens ruidosas.

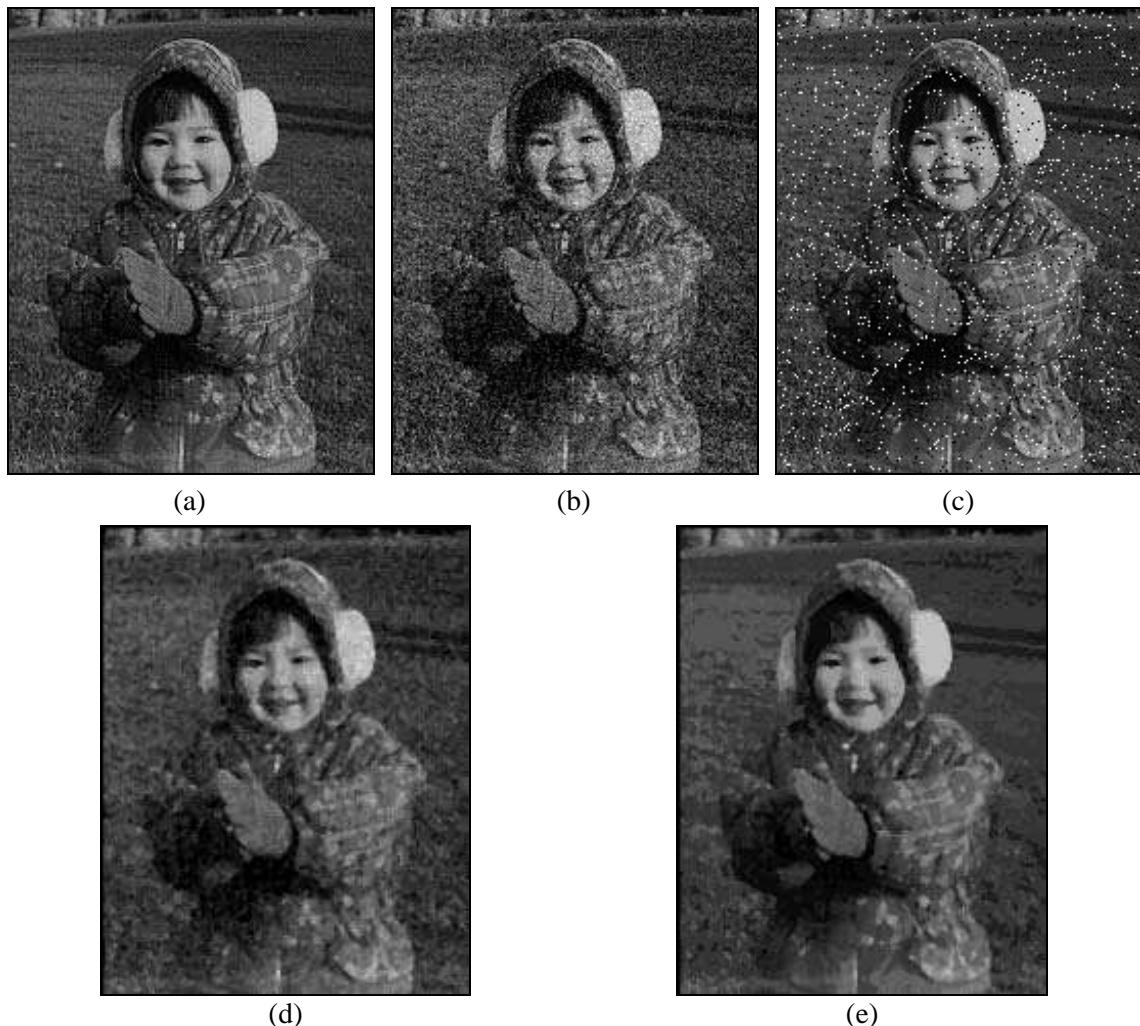


Figura 45 - (a) Imagem original; (b) imagem contaminada por ruído gaussiano; (c) imagem contaminada por ruído sal e pimenta; (d) resultado da aplicação do filtro DW-MTM sobre a imagem (b); (e) resultado da aplicação do filtro DW-MTM sobre a imagem (c).

Filtro da Mediana Adaptativo (SAM - Signal Adaptive Median)

O filtro adaptativo SAM utiliza o fato de que uma região de fundo uniforme de uma imagem contém pouca informação de baixa freqüência e que a maioria das informações de alta freqüência estão contidas em bordas e impulsos. Partindo desta premissa, pode-se decompor uma imagem $g(x,y)$, em suas componentes de baixa e alta freqüência:

$$g(x,y) = g_{lf}(x,y) + g_{hf}(x,y) \quad (4.98)$$

A figura 46 mostra esquematicamente o processo de decomposição de $g(x,y)$, através da aplicação de um filtro passa-altas e de um filtro passa-baixas convencional.

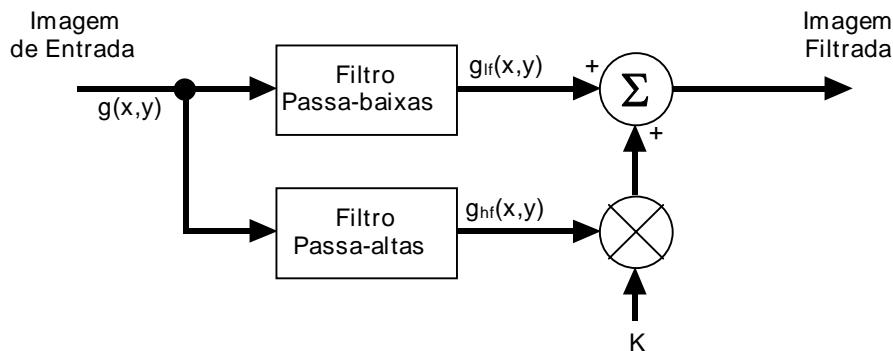


Figura 46 - Decomposição de uma imagem em suas componentes de alta e baixa freqüência.

Uma vez separadas, apenas as componentes de baixa freqüência são usadas como saída do filtro se ele está posicionado sobre uma região de fundo da imagem. Nas regiões contendo bordas, ambas as componentes de baixa e alta freqüência são utilizadas. Um parâmetro de controle (K) determina o quanto de componentes de alta freqüência aparecem na saída do filtro.

Sabendo que a componente de alta freqüência também pode ser determinada a partir da imagem original não filtrada e da imagem filtrada de baixa freqüência, através da eq. (4.17), o filtro SAM pode ser implementado conforme o diagrama da figura 47. Neste diagrama, o filtro da mediana é usado para permitir a filtragem passa-baixas e para obter também as componentes de alta freqüência. Um detetor de bordas e impulsos determina o tamanho da janela (máscara) e ainda o valor de K . O filtro começa com um determinado tamanho de janela $n \times n$ e, se uma borda é detectada, seu tamanho é reduzido para $n-2 \times n-2$. Este processo se repete até que se obtenha uma janela sem a presença de bordas. Neste momento o valor de K é escolhido com base na variância local calculada dentro da janela e em uma estimativa da variância do ruído.

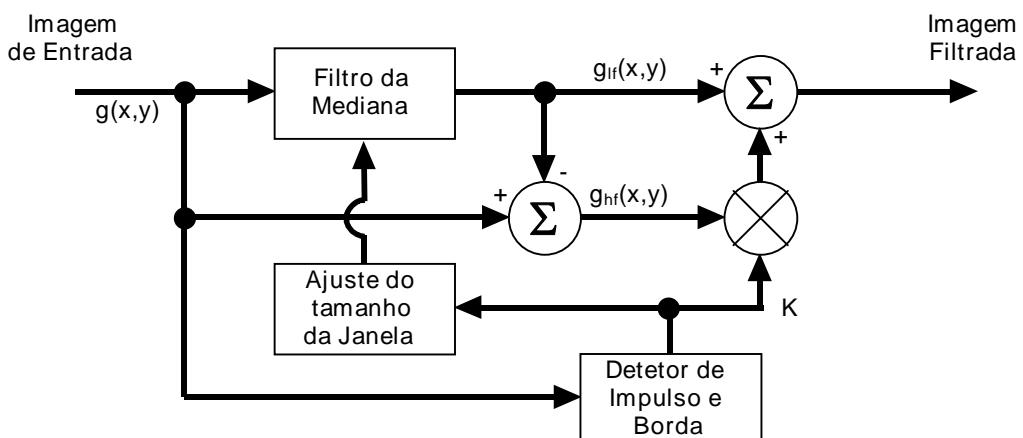


Figura 47 - Diagrama em blocos do filtro SAM.

O valor de K é determinado segundo o processo de decisão mostrado na eq. (4.99):

$$K = \begin{cases} 0 & \text{para } C \cdot \sigma_n^2 \geq \sigma_1^2 \\ 1 - C \cdot \frac{\sigma_n^2}{\sigma_1^2} & \text{para as demais situações} \end{cases} \quad (4.99)$$

Se a variância local for menor que o produto de uma constante C pela variância do ruído, então a saída do filtro SAM será igual à saída do filtro da mediana. Caso contrário, uma parcela das componentes de alta frequência são adicionadas à saída do filtro SAM. Esta situação normalmente indica a presença de uma borda. A constante C é usada para ajustar a sensibilidade do filtro a bordas.

A figura 48 mostra exemplos de aplicação do filtro SAM em imagens ruidosas.

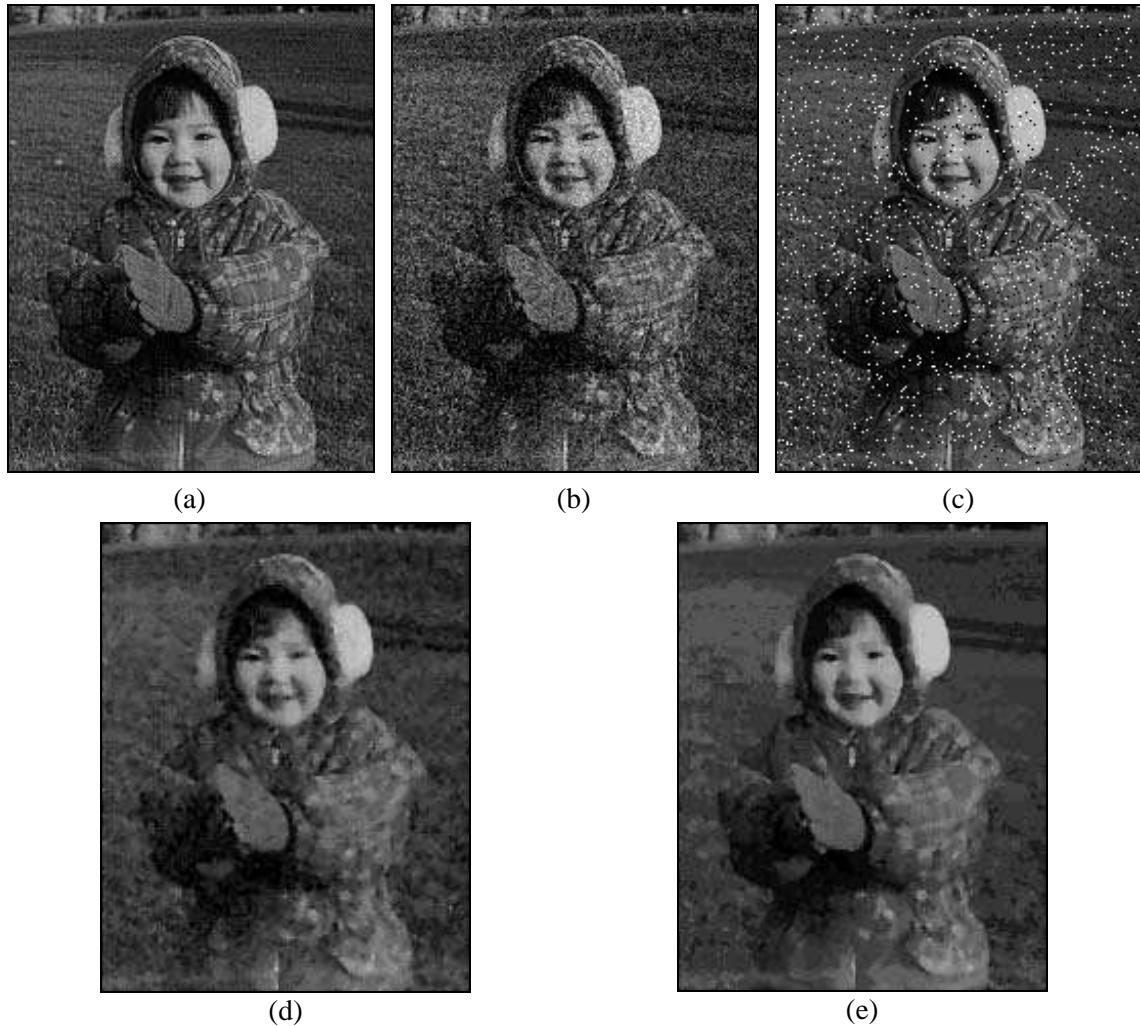


Figura 48 - (a) Imagem original; (b) imagem contaminada por ruído gaussiano; (c) imagem contaminada por ruído sal e pimenta; (d) resultado da aplicação do filtro SAM sobre a imagem (b); (e) resultado da aplicação do filtro SAM sobre a imagem (c).

Leitura complementar

O capítulo 11 de [Myler e Weeks 1993] é inteiramente dedicado a filtros adaptativos bidimensionais.

Para um maior detalhamento de distribuições de probabilidade, sugerimos [Ross 1994].

Mais detalhes sobre os filtros MMSE, SAM e DW-MTM, inclusive com as respectivas funções implementadas utilizando MATLAB®, podem ser encontradas em [Marques e da Costa 1996].

Exercícios Propostos

1. Considere o trecho de imagem a seguir, representado por uma matriz 7×7 , onde cada elemento da matriz corresponde ao nível de cinza do pixel correspondente. Sabe-se que na quantização desta imagem foram utilizados 8 bits. Seja o pixel central o pixel de referência. Forneça o valor resultante do pixel central caso a imagem seja processada:

a) pelo algoritmo da filtragem pela mediana utilizando janela 3×3 .

b) pelo algoritmo da filtragem pela mediana utilizando janela em forma de cruz, isto é considerando no cálculo da mediana apenas os pixels de coordenadas: (x, y) (pixel de referência), $(x-1, y)$, $(x+1, y)$, $(x, y-1)$ e $(x, y+1)$.

c) por um algoritmo adaptativo que funciona da seguinte maneira: primeiramente aplica-se um filtro da mediana em uma janela 3×3 ao redor do pixel de referência, calculando-se MED . Depois disto, aplica-se um filtro da média utilizando uma janela 5×5 , levando em consideração apenas os pixels cujo tom de cinza esteja dentro da faixa entre $MED - C$ e $MED + C$, inclusive os extremos. Assumir que $C = 22$.

d) pelo algoritmo da média utilizando janela 7×7 .

e) pelo algoritmo da pseudomediana utilizando janela 3×3 .

0	3	221	220	198	84	4
3	23	187	188	189	99	8
9	9	188	115	134	49	9
0	5	176	18	187	98	9
15	15	123	103	165	76	9
14	12	156	188	188	98	9
9	8	190	190	190	90	0

2. Assinalar V ou F conforme as proposições a seguir sejam verdadeiras ou falsas.

() A técnica da filtragem pela mediana é sempre melhor que a filtragem pela média quando o objetivo é reduzir ruído presente em uma imagem.

() A propriedade da Transformada de Fourier que permite que a FFT 2-D seja computada a partir de duas aplicações do algoritmo da FFT 1-D é a separabilidade.

() A remoção de ruídos usando a filtragem pela média é de grande utilidade quando se deseja processar imagens contendo pontos ruidosos isolados, cuja amplitude (tom de cinza) é

bem maior do que os tons de cinza de seus vizinhos. Nestes casos, a filtragem pela média parece remover um a um (como se utilizasse uma pinça) os pixels ruidosos.

() A técnica de remoção de ruídos usando a média de múltiplas imagens tem aplicabilidade limitada pois somente pode ser aplicada com sucesso em situações nas quais se disponha de diversas versões da mesma imagem capturadas em instantes de tempo diferentes e sobre as quais exista um ruído de valor médio zero e descorrelacionado.

() A técnica de realce de imagens conhecida como ênfase de alta freqüência é uma modificação da filtragem passa-altas, cujo efeito é o de diminuir a atenuação dos componentes de baixa freqüência da imagem, o que corresponde a obter um bom realce dos detalhes de alto contraste sem sacrificar exageradamente as áreas de menor riqueza de detalhes.

() A aplicação do algoritmo da filtragem da média com os K vizinhos mais próximos, onde o nível de cinza do pixel de referência de uma janela é substituído pelo valor médio dos K vizinhos de p (pixel de referência), cujos níveis de cinza mais se aproximam do nível de cinza de p , causará maior redução do ruído quanto menor o valor de K .

() Na aplicação do filtro da mediana sobre uma imagem utilizando máscara 3×3 , não são gerados novos valores de tons de cinza na imagem resultante, ao contrário do filtro da média, que pode, eventualmente, gerá-los.

() Toda técnica de realce de imagens pressupõe uma certa subjetividade, já que ao final do processo um ser humano dirá se a imagem resultante é melhor ou pior que a original.

3. Sabe-se que para computar uma FFT de N pontos são necessárias $N \log_2 N$ adições e $0.5N \log_2 N$ multiplicações. Quantas adições e multiplicações são necessárias para se computar a FFT bidimensional de uma imagem $M \times N$? Justifique.

4. Considere a expressão genérica da filtragem pela média dada a seguir:

$$g(x,y) = \frac{1}{M} \sum_{(n,m) \in S} f(n,m)$$

Supondo que $M = 4$, o que corresponde a computar a média dos quatro vizinhos imediatos de (x, y) , excluindo o próprio (x, y) , obter o filtro equivalente $H(u, v)$ no domínio da freqüência e mostrar que este filtro é do tipo passa-baixas.

5. Verificar a validade da eq. (4.17) utilizando os conceitos de convolução com máscaras e as máscaras das figuras 4(a) e 11.

No computador

Executar os roteiros das práticas 5 e 6 do Apêndice B para sedimentar os conteúdos teóricos do capítulo.

Na Internet

Dentre as diversas referências disponíveis na WWW correlatas a este capítulo, destacamos:

["http://www.eecs.wsu.edu/IPdb/Enhancement/averaging.html"](http://www.eecs.wsu.edu/IPdb/Enhancement/averaging.html)

Averaging Filter

"<http://www.eecs.wsu.edu/IPdb/Enhancement/median.html>"

Median Filter

"http://www.eecs.wsu.edu/IPdb/Enhancement/butterworth_low.html"

Butterworth Lowpass Filter

"http://www.eecs.wsu.edu/IPdb/Enhancement/unsharp_masking.html"

Unsharp Masking

"http://www.eecs.wsu.edu/IPdb/Enhancement/butterworth_high.html"

High Frequency Emphasis

"http://www.eecs.wsu.edu/IPdb/Enhancement/homo_filtering.html"

Homomorphic Filtering

"<http://ai.bpa.arizona.edu/~chrisy/cip.html>"

Color Image Processing

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c9/s3/front-page.html>"

Median Filtering

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c6/s4/front-page.html>"

Image Sharpening

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c4/s9/front-page.html>"

Pseudocolor Applications

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c5/s3/front-page.html>"

DFT: Properties

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c5/s4/front-page.html>"

DFT of Simple Images

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c5/s7/front-page.html>"

DFT: Filtering in the Frequency Domain

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c2/s7/front-page.html>"

Color Models Concepts

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c2/s8/front-page.html>"

RGB Image Manipulation

Bibliografia

- [Araújo 1989] Araújo, A.A., *Filtragem Espacial - Técnicas de Realce para Imagem*, IX Congresso da SBC, VIII Jornada de Atualização em Informática, 16-21 Julho 1989, Uberlândia-MG.
- [Brigham 1974] Brigham, E.O., *The Fast Fourier Transform*, Prentice-Hall, 1974.
- [Davis e Rosenfeld 1978] Davis, L.S. e Rosenfeld, A., "Noise Cleaning by Iterated Local Averaging", *IEEE Transactions on Systems, Man and Cybernetics*, 7, 705-710.
- [Dougherty 1994] Dougherty, E.R. (ed.), *Digital Image Processing Methods*, Marcel Dekker, 1994.
- [Enden e Verhoeckx 1989] Enden, A.W.M. van den e Verhoeckx, N.A.M., *Discrete-time signal processing: an introduction*, Prentice-Hall, 1989.
- [Gonzalez e Woods 1992] Gonzalez, R.C. e Woods, R.E., *Digital Image Processing - Third Edition*, Addison-Wesley, 1992.
- [Jain 1989] Jain, A.K., *Fundamentals of Digital Image Processing*, Prentice-Hall, 1989.
- [Lim 1990] Lim, J.S., *Two-dimensional Signal and Image Processing*, Prentice-Hall, 1990.
- [Marques e da Costa 1996] Marques, F.A.L. e da Costa, F.M., "Filtragem de Imagens Usando Filtros Adaptativos", Relatório Técnico, CEFET-PR, Curitiba-PR, 1996.
- [Myler e Weeks 1993] Myler, H.R. e Weeks, A.R., *Computer Imaging Recipes in C*, Prentice Hall, 1993.
- [Oppenheim et al. 1983] Oppenheim, A.V., Willsky, A.S. e Young, I.T., *Signals and systems*, Prentice-Hall, 1983.
- [Papoulis 1962] Papoulis, A., *The Fourier integral and its applications*, McGraw-Hill, 1962.
- [Pavlidis 1982] Pavlidis, T., *Algorithms for Graphics and Image Processing*, Computer Science Press, 1982.
- [Pratt 1991] Pratt, W. K., *Digital Image Processing*, Wiley Interscience, 1991. (2nd ed.)
- [Pratt et al. 1984] Pratt, W.K., Cooper, T.J. e Kabir, I., "Pseudomedian Filter", *Proc. SPIE Conference*, Los Angeles, CA, Janeiro 1984.
- [Press et al. 1994] Press, W.H. et al., *Numerical Recipes in C - 2nd Edition*, Cambridge University Press, 1994.
- [Ross 1994] Ross, S., *A First Course in Probability - 4th edition*, Macmillan, 1994.

Capítulo 5

Morfologia Matemática

Este capítulo tem por objetivo apresentar os principais conceitos, operações e algoritmos de uma importante área de suporte do processamento de imagens, que é a Morfologia Matemática. A Seção 5.1 traz considerações históricas e antecipa uma visão geral do que é morfologia matemática. Na Seção 5.2 são apresentadas as operações de dilatação e erosão. A Seção 5.3 descreve e exemplifica as importantes operações de abertura e fechamento. A Seção 5.4 é inteiramente dedicada ao conceito de transformação *hit-or-miss*, dadas suas diversas aplicações em reconhecimento de padrões. O capítulo é concluído com a apresentação de diversos algoritmos morfológicos básicos, compilados na Seção 5.5.

5.1 Introdução

Assim como na biologia, onde a expressão morfologia se refere ao estudo da estrutura dos animais e plantas, a morfologia matemática, elaborada inicialmente por Georges Matheron e Jean Serra [Serra 1982], concentra seus esforços no estudo da estrutura geométrica das entidades presentes em uma imagem. A morfologia matemática pode ser aplicada em várias áreas de processamento e análise de imagens, com objetivos tão distintos como realce, filtragem, segmentação, deteção de bordas, esqueletização, afinamento, dentre outras.

O princípio básico da morfologia matemática consiste em extrair as informações relativas à geometria e à topologia de um conjunto desconhecido (uma imagem), pela transformação através de outro conjunto completamente definido, chamado elemento estruturante. Portanto, a base da morfologia matemática é a teoria de conjuntos. Por exemplo, o conjunto de todos os pixels pretos em uma imagem binária descreve completamente a imagem (uma vez que os demais pontos só podem ser brancos). Em imagens binárias, os conjuntos em questão são membros do espaço inteiro bidimensional Z^2 , onde cada elemento do conjunto é um vetor 2-D cujas coordenadas são as coordenadas (x,y) do pixel preto (por convenção) na imagem. Imagens com mais níveis de cinza podem ser representadas por conjuntos cujos elementos estão no espaço Z^3 . Neste caso, os vetores têm três elementos, sendo os dois primeiros as coordenadas do pixel e o terceiro seu nível de cinza.

Leitura complementar

Dois livros clássicos, imprescindíveis para quem deseja se aprofundar no tema, são [Serra 1982] e [Serra 1988].

O livro de Facon [Façon 1996] é uma das poucas referências em português inteiramente dedicadas ao assunto.

Para uma revisão da teoria de conjuntos, indicamos [Ross e Wright 1992].

Este capítulo concentra-se em conceitos e exemplos de morfologia matemática aplicada a imagens binárias. Para uma introdução à extensão destes conceitos para imagens com mais níveis de intensidade sugerimos o capítulo 12 de [Serra 1982], a Seção 5.5 de [Haralick e Shapiro 1992] e a Seção 8.4.5 de [Gonzalez e Woods 1992].

5.2 Dilatação e Erosão

Iniciaremos nossa discussão de operações morfológicas pelas duas operações básicas: dilatação e erosão. Para bem compreendê-las, inicialmente apresentaremos algumas definições úteis da teoria de conjuntos.

5.2.1 Definições básicas

Sejam A e B conjuntos em Z^2 , cujos componentes são $a = (a_1, a_2)$ e $b = (b_1, b_2)$, respectivamente. A translação de A por $x = (x_1, x_2)$, denotada $(A)_x$, é definida como:

$$(A)_x = \{c \mid c = a + x, \text{ para } a \in A\}. \quad (5.1)$$

A reflexão de B , denotada \hat{B} , é definida como:

$$\hat{B} = \{x \mid x = -b, \text{ para } b \in B\}. \quad (5.2)$$

O complemento do conjunto A é:

$$A^c = \{x \mid x \notin A\}. \quad (5.3)$$

Finalmente, a diferença entre dois conjuntos A e B , denotada $A - B$, é definida como:

$$A - B = \{x \mid x \in A, x \notin B\} = A \cap B^c. \quad (5.4)$$

A figura 1 ilustra geometricamente as definições apresentadas, onde pontos pretos identificam a origem do par de coordenadas. A figura 1(a) mostra o conjunto A . A parte (b) mostra a translação de A por $x = (x_1, x_2)$. O conjunto B é exibido na parte (c), enquanto a figura 1(d) mostra sua reflexão em relação à origem. Finalmente, a parte (e) apresenta o conjunto A e seu complemento, enquanto a figura 1(f) mostra a diferença entre este conjunto A e o conjunto B .

5.2.2 Dilatação

Sejam A e B conjuntos no espaço Z^2 e seja \emptyset o conjunto vazio. A dilatação de A por B , denotada $A \oplus B$, é definida como:

$$A \oplus B = \{x \mid (\hat{B})_x \cap A \neq \emptyset\}. \quad (5.5)$$

Portanto, o processo de dilatação consiste em obter a reflexão de B sobre sua origem e depois deslocar esta reflexão de x . A dilatação de A por B é, então, o conjunto de todos os x deslocamentos para os quais a interseção de $(\hat{B})_x$ e A inclui pelo menos um elemento diferente de zero. Com base nesta interpretação, a equação anterior pode ser escrita como:

$$A \oplus B = \{x \mid [(\hat{B})_x \cap A] \subseteq A\}. \quad (5.6)$$

O conjunto B é normalmente denominado elemento estruturante.

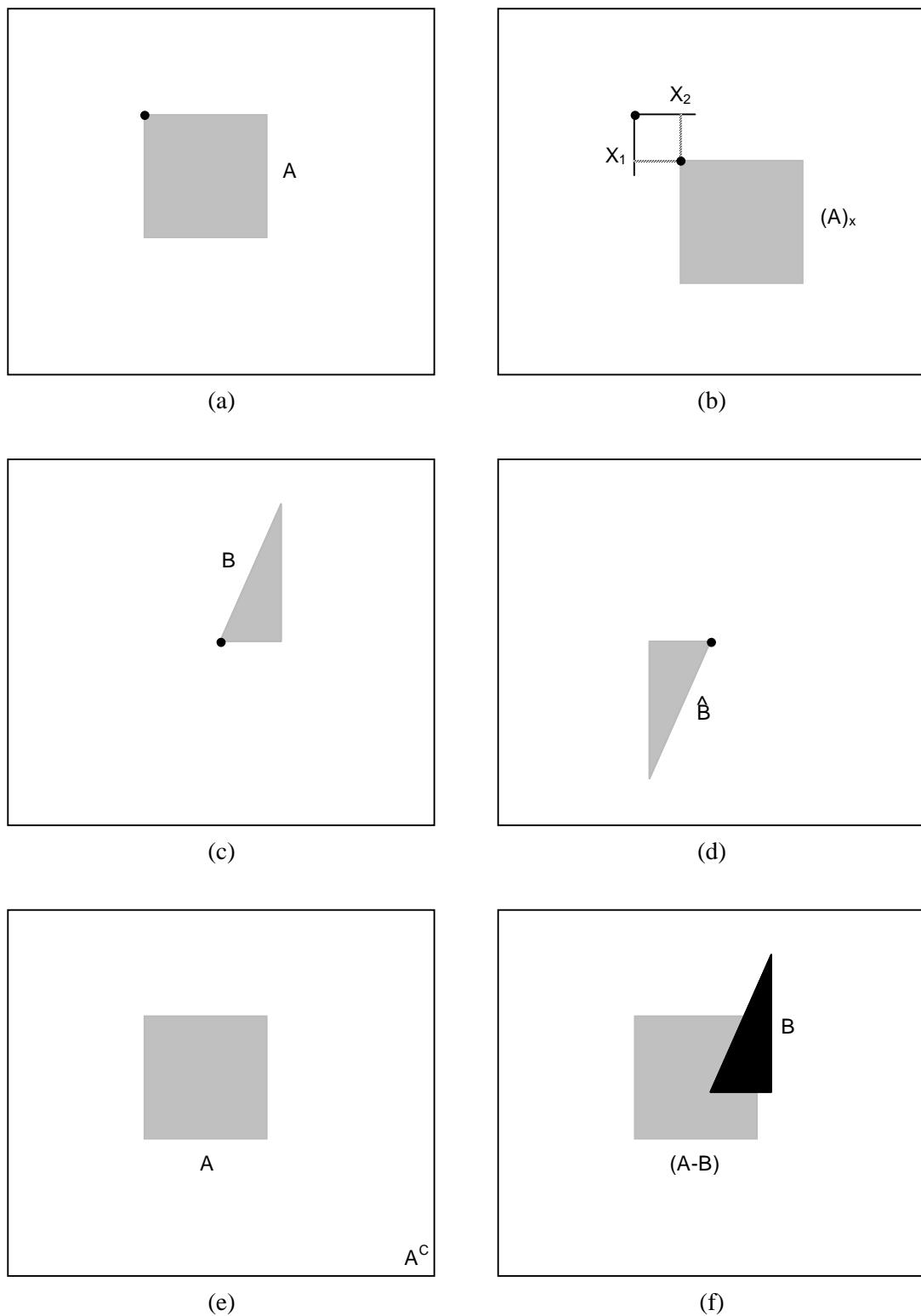


Figura 1 - Exemplos de operações básicas sobre conjuntos.

A figura 2 mostra os efeitos da dilatação de um conjunto A usando três elementos estruturantes (B) distintos. Observar que as operações morfológicas são sempre referenciadas a um elemento do conjunto estruturante (neste caso, o elemento central).

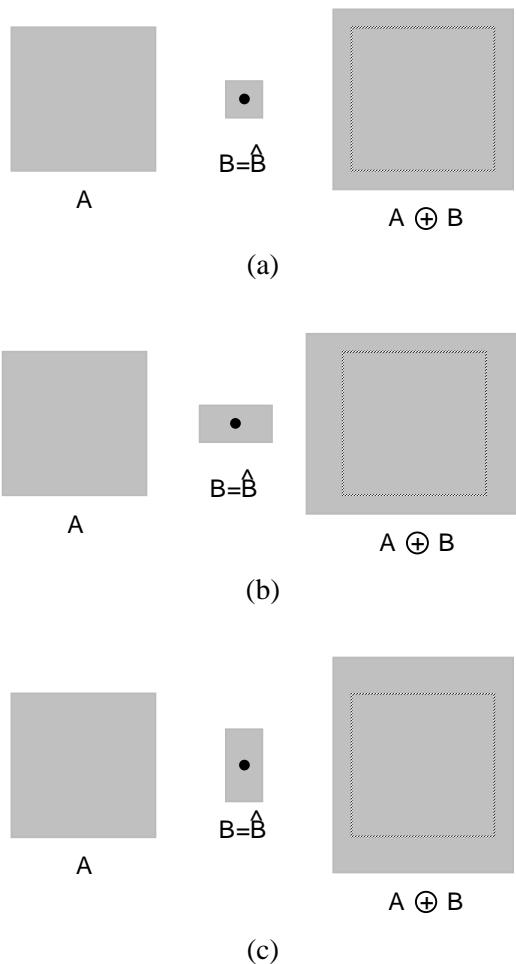


Figura 2 - Dilatação.

5.2.3 Erosão

Sejam A e B conjuntos no espaço Z^2 . A erosão de A por B , denotada $A \Theta B$, é definida como:

$$A \Theta B = \{x | (B)_x \subseteq A\} \quad (5.7)$$

o que, em outras palavras significa dizer que a erosão de A por B resulta no conjunto de pontos x tais que B , transladado de x , está contido em A .

A figura 3 mostra os efeitos da erosão de um conjunto A usando três elementos estruturantes (B) distintos.

A dilatação e a erosão são operações duais entre si com respeito a complementação e reflexão. Ou seja,

$$(A \Theta B)^c = A^c \oplus \hat{B}. \quad (5.8)$$

A prova desta dualidade está demonstrada a seguir:

Partindo da definição de erosão, temos:

$$(A \Theta B)^c = \left\{x \mid (B)_x \not\subseteq A\right\}^c. \quad (5.9)$$

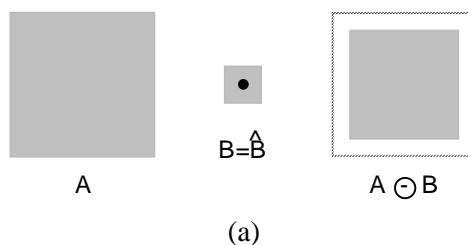
Se o conjunto $(B)_x$ está contido no conjunto A , então $(B)_x \cap A^c = \emptyset$. Portanto, a equação anterior torna-se:

$$(A \Theta B)^c = \left\{ x \mid (B)_x \cap A^c = \emptyset \right\}^c. \quad (5.10)$$

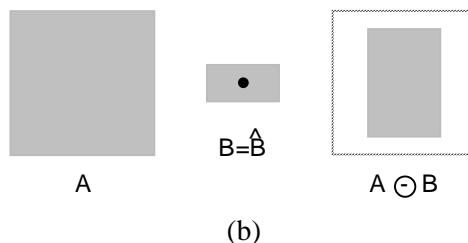
Porém, o complemento do conjunto dos x 's que satisfazem $(B)_x \cap A^c = \emptyset$ é o conjunto dos x 's tais que $(B)_x \cap A^c \neq \emptyset$. Logo,

$$\begin{aligned} (A \Theta B)^c &= \left\{ x \mid (B)_x \cap A^c \neq \emptyset \right\} \\ &= A^c \oplus \hat{B} \end{aligned} \quad (5.11)$$

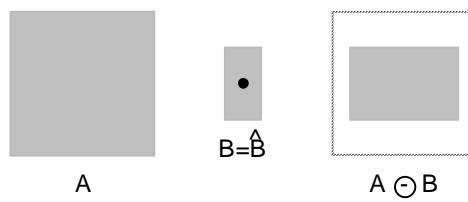
q.e.d.



(a)



(b)



(c)

Figura 3 - Erosão.

Leitura complementar

Os conceitos de dilatação e erosão são vistos no capítulo 2 de [Serra 1982], no capítulo 2 de [Facon 1996] e na Seção 5.2 de [Haralick e Shapiro 1992].

5.3 Abertura e Fechamento

Como vimos nas figuras da seção anterior, a dilatação expande uma imagem enquanto a erosão a encolhe. Nesta seção discutiremos duas outras importantes operações morfológicas: a abertura e o fechamento.

A abertura em geral suaviza o contorno de uma imagem, quebra istmos estreitos e elimina proeminências delgadas. O fechamento, por sua vez, funde pequenas quebras e alarga golbos estreitos, elimina pequenos orifícios e preenche *gaps* no contorno.

A abertura de um conjunto A por um elemento estruturante B , denotada $A \circ B$, é definida como:

$$A \circ B = (A \Theta B) \oplus B \quad (5.12)$$

o que equivale a dizer que a abertura de A por B é simplesmente a erosão de A por B seguida de uma dilatação do resultado por B .

O fechamento do conjunto A pelo elemento estruturante B , denotado $A \bullet B$, é definido como:

$$A \bullet B = (A \oplus B) \Theta B \quad (5.13)$$

o que nada mais é que a dilatação de A por B seguida da erosão do resultado pelo mesmo elemento estruturante B .

A figura 4 mostra exemplos de operações de abertura e fechamento utilizando um elemento estruturante circular. A parte (a) da figura mostra a operação de abertura, indicando no alto o conjunto original A , na linha intermediária a etapa de erosão e na linha inferior o resultado da operação de dilatação aplicada ao conjunto resultante da erosão. Na figura 4(b) são detalhadas as operações de dilatação do conjunto original A e subsequente erosão do resultado.

5.3.1 Interpretação geométrica da abertura e do fechamento

A abertura e o fechamento podem ser interpretados geometricamente de maneira simples. Suponha-se, por exemplo, que o elemento estruturante circular B da figura 4 como um disco plano. A fronteira de $A \circ B$ é composta pelos pontos da fronteira de B que se distanciam mais para dentro da fronteira de A à medida que B é girado em torno da parte interna desta fronteira. Esta propriedade geométrica de 'encaixe' da operação de abertura pode ser expressa em termos da teoria de conjuntos como:

$$A \circ B = \bigcup \{(B)_x \mid (B)_x \subset A\} \quad (5.14)$$

A figura 5 mostra este conceito com um elemento estruturante de outro formato.

De maneira similar, a operação de fechamento pode ser interpretada geometricamente, supondo que o disco desliza pela parte externa da fronteira de A . Geometricamente, um ponto z é um elemento de $A \bullet B$ se e somente se $(B)_z \cap A \neq \emptyset$ para qualquer translação de (B) que contenha z . A figura 6 mostra esta propriedade.

Assim como no caso da dilatação e erosão, a abertura e o fechamento são duais, ou seja:

$$(A \bullet B)^c = (A^c \circ \hat{B}). \quad (5.15)$$

5.3.2 Propriedades da abertura

- (i) $A \circ B$ é um subconjunto (subimagem) de A .
- (ii) Se C é um subconjunto de D , então $C \circ B$ é um subconjunto de $D \circ B$.
- (iii) $(A \circ B) \circ B = A \circ B$.

5.3.3 Propriedades do fechamento

- (i) A é um subconjunto de $A \bullet B$.
- (ii) Se C é um subconjunto de D , então $C \bullet B$ é um subconjunto de $D \bullet B$.
- (iii) $(A \bullet B) \bullet B = A \bullet B$.

Estas propriedades auxiliam na interpretação dos resultados obtidos quando as operações de abertura e fechamento são utilizadas para construir filtros morfológicos. Para um exemplo de filtro morfológico, na figura 7 apresentamos uma imagem de um objeto retangular com ruído à qual se aplica o filtro $(A \circ B) \bullet B$. Após a operação de abertura, os pontos ruidosos externos ao objeto já foram removidos. A etapa de fechamento remove os pixels ruidosos do interior do objeto. Convém observar que o sucesso desta técnica depende do elemento estruturante ser maior que o maior aglomerado de pixels ruidosos conectados presente na imagem original.

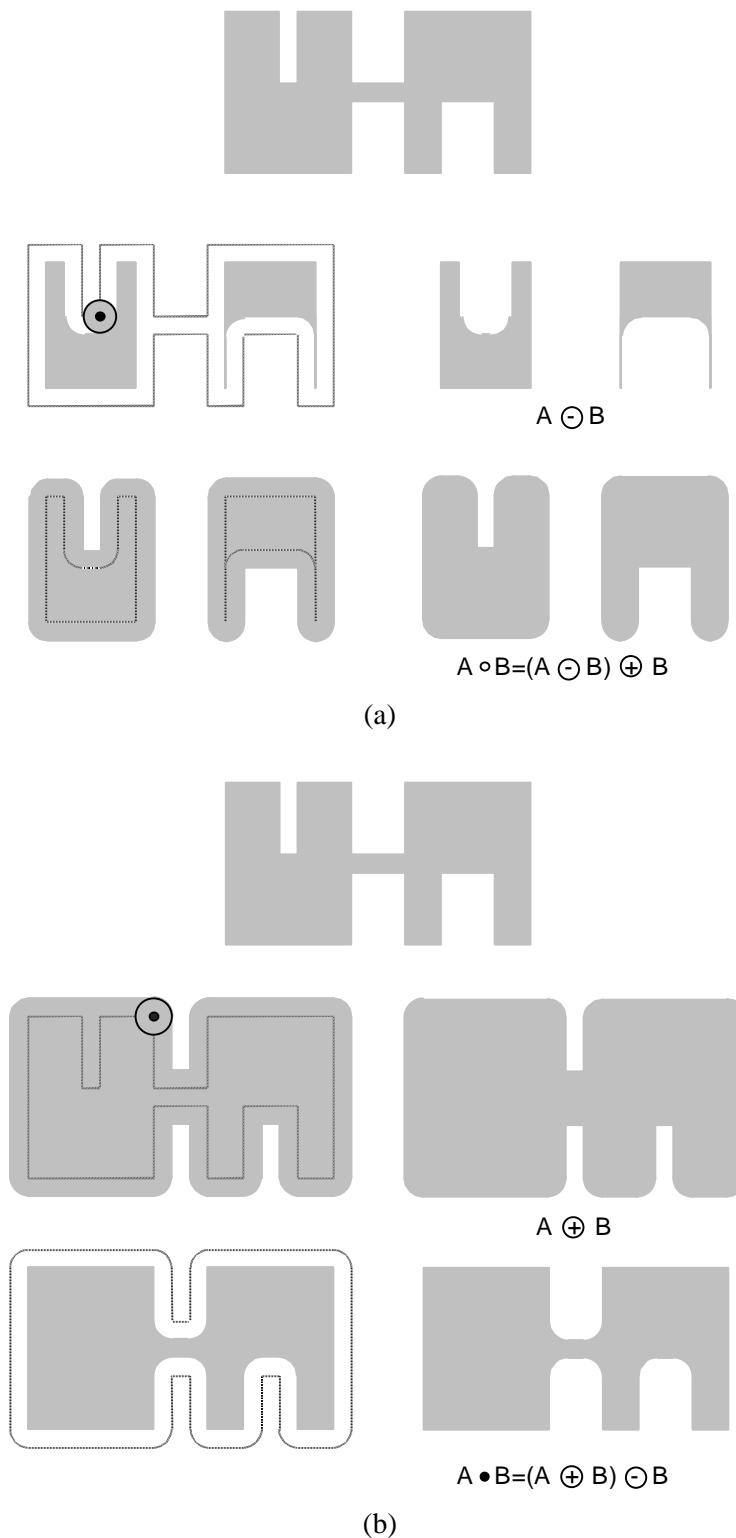


Figura 4 - Exemplos de abertura e fechamento.

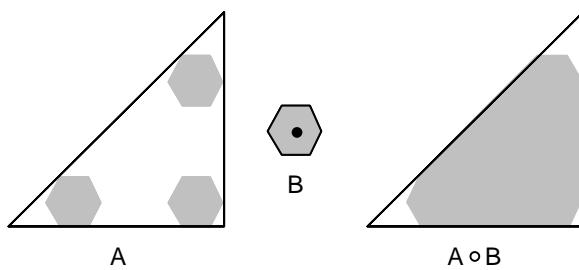


Figura 5 - Propriedade de 'encaixe' da abertura.

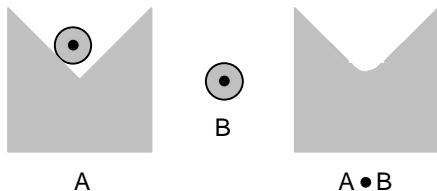


Figura 6 - Interpretação geométrica do fechamento.

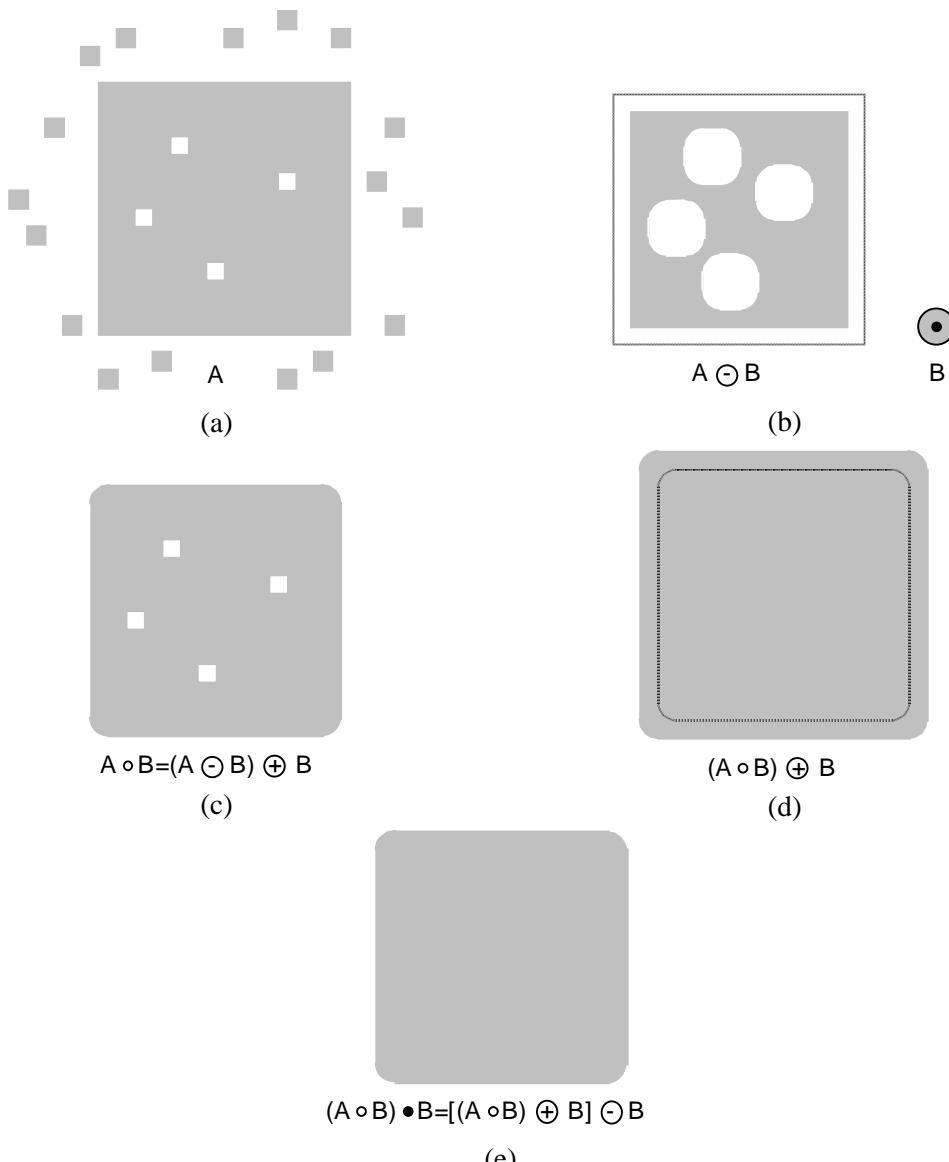


Figura 7 - Filtro morfológico: (a) imagem original, ruidosa; (b) resultado da erosão; (c) abertura de A; (d) resultado de uma operação de dilatação aplicada à imagem (c); (e) resultado final.

Leitura complementar

O capítulo 1 de [Dougherty 1994] apresenta outros exemplos de aplicação de filtros morfológicos.

O capítulo 5 de [Haralick e Shapiro 1992] mostra os conceitos de abertura e fechamento e do uso de técnicas morfológicas para redução de ruídos em imagens.

O capítulo 2 de [Serra 1982] apresenta os conceitos de abertura e fechamento.

Os capítulos 2 e 3 de [Dougherty 1993] abordam as propriedades estatísticas e apresentam estratégias de projeto de filtros morfológicos.

5.4 Transformação *hit-or-miss*

A transformação morfológica *hit-or-miss* é uma ferramenta básica para o reconhecimento de padrões. Na figura 8 se vê um conjunto A que consiste de três padrões (subconjuntos), X , Y e Z . O sombreado das partes (a)-(c) indica os conjuntos originais, enquanto que as áreas sombreadas das partes (d) e (e) da figura indicam os resultados das operações morfológicas. Seja o objetivo: buscar a localização de um dos objetos de A , por exemplo, Y .

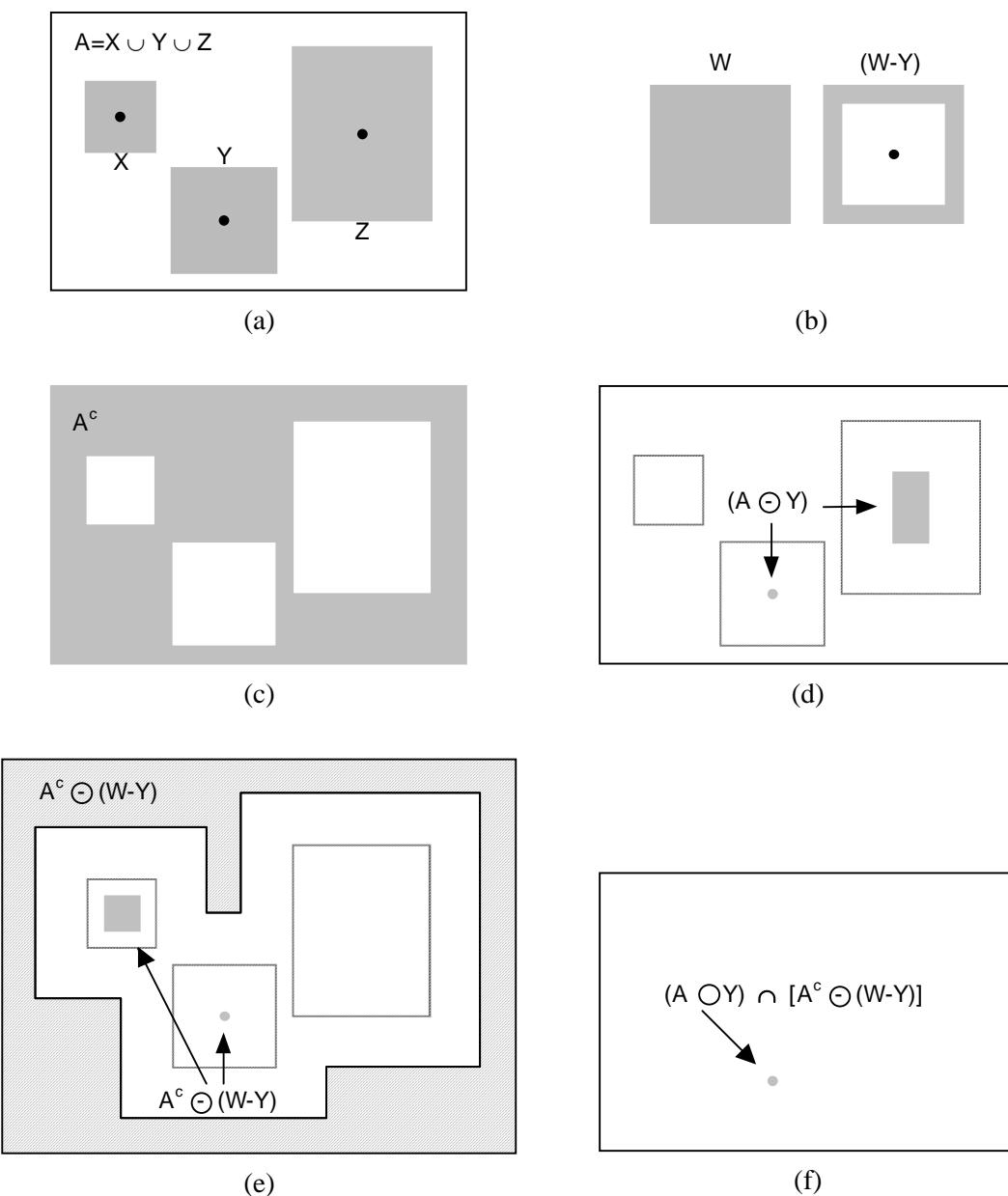


Figura 8 - Transformação *hit-or-miss*.

Seja a origem de cada forma localizada em seu centro de gravidade. Se circundarmos Y com uma pequena janela W , o 'fundo local' de Y com respeito a W será o conjunto diferença ($W - Y$) mostrado na parte (b). A figura 8(c) mostra o complemento de A , que será necessário mais adiante. A parte (d) mostra a erosão de A por Y . A figura 8(e) mostra a erosão do complemento de A pelo conjunto fundo local ($W - Y$); a região sombreada externa é parte da erosão. Notar que o conjunto dos lugares para os quais Y cabe exatamente dentro de A é a interseção da erosão de A por Y e a erosão de A^c por ($W - Y$), como mostra a parte (f). Esta interseção é precisamente o lugar que se está buscando. Em outras palavras, se B denota o conjunto composto por X e seu fundo, o encaixe de B em A , denotado $A \text{ hom } B$, é:

$$A \text{ hom } B = (A \Theta Y) \cap [A^c \Theta (W - Y)]. \quad (5.16)$$

Generalizando, pode-se fazer $B = (B_1, B_2)$, onde B_1 é o conjunto dos elementos de B associados com um objeto e B_2 o conjunto dos elementos de B associados com o fundo correspondente. No caso anterior, $B_1 = Y$ e $B_2 = (W - Y)$. Com esta notação, pode-se escrever:

$$A \text{ hom } B = (A \Theta B_1) \cap (A^c \Theta B_2). \quad (5.17)$$

Usando a definição de diferenças de conjuntos e a relação dual entre erosão e dilatação podemos escrever:

$$A \text{ hom } B = (A \Theta B_1) - (A \oplus \hat{B}_2). \quad (5.18)$$

Logo, o conjunto $A \text{ hom } B$ contém todos os pontos para os quais, simultaneamente, B_1 encontrou uma correspondência (ou um '*hit*') em A e B_2 encontrou uma correspondência em A^c .

Leitura complementar

O capítulo 2 de [Serra 1982] e a Seção 5.2 de [Haralick e Shapiro 1992] apresentam conceitos e exemplos de transformada *hit-or-miss*.

5.5 Algoritmos morfológicos básicos

Começaremos agora a tratar dos usos práticos da morfologia matemática em processamento de imagens. Quando se está trabalhando com imagens binarizadas, a principal aplicação da morfologia é extrair componentes da imagem que sejam úteis na representação e descrição de formatos. A seguir, serão apresentados algoritmos de extração de contornos, extração de componentes conectados, delimitação do casco convexo de um objeto e esqueletização de uma região. Também são apresentados algoritmos úteis para as etapas de pré- ou pós-processamento, tais como os de afinamento (*thinning*), preenchimento de regiões (*region filling*), espessamento (*thickening*) e poda (*pruning*).

5.5.1 Extração de contornos

É possível extrair o contorno de um conjunto A , denotado por $\beta(A)$, executando a erosão de A por B e então calculando a diferença entre A e sua erosão. Isto é,

$$\beta(A) = A - (A \Theta B) \quad (5.19)$$

onde B é um elemento estruturante adequado.

A figura 9 mostra a mecânica da extração de contornos. Na parte (a) tem-se o conjunto original, na parte (b) o elemento estruturante, em (c) o resultado da erosão e finalmente em (d) o resultado da diferença, que corresponde ao contorno de A .

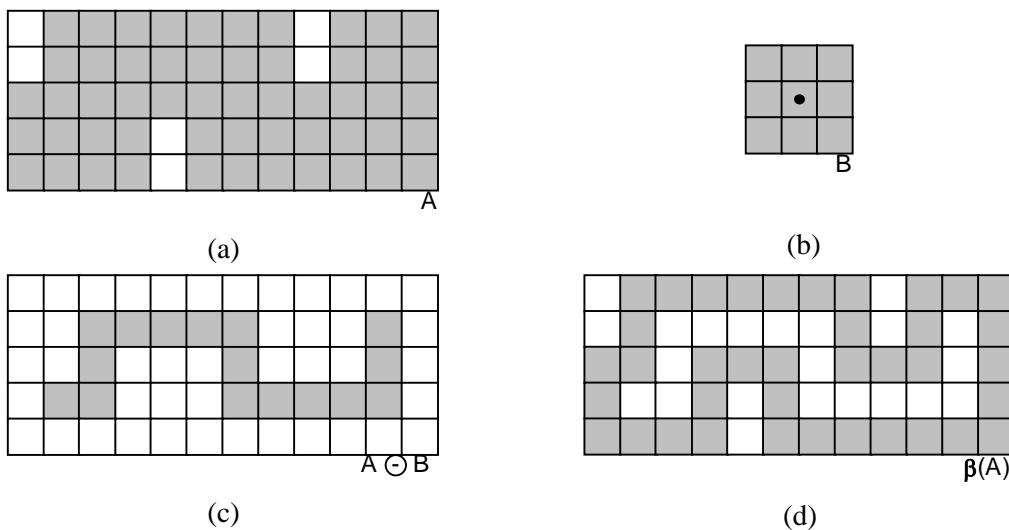


Figura 9 - Extração de contornos.

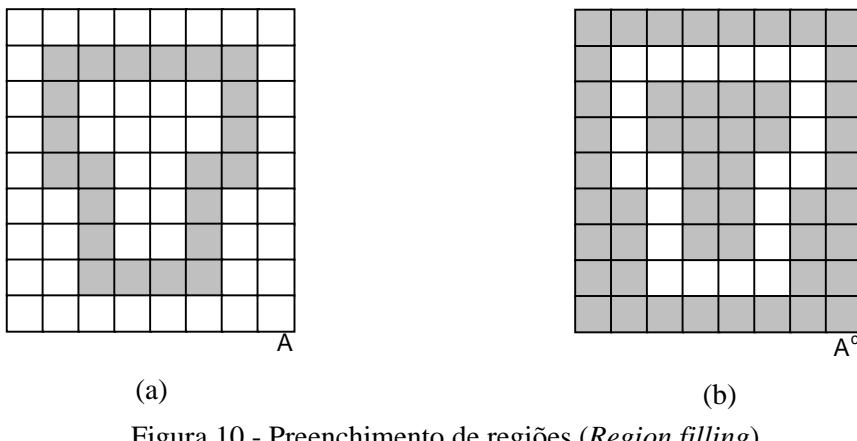
5.5.2 Preenchimento de regiões (*Region filling*)

Seja um contorno fechado A , que pode ser expresso como um conjunto contendo um subconjunto cujos elementos são pontos do contorno 8-conectados. Partindo de um ponto p situado dentro do contorno, o que se deseja é preencher o interior de esta região com 1's.

Assumindo que todos os pontos que não estão sobre a fronteira estão rotulados como 0, atribuímos o valor 1 a p para iniciar o procedimento. O procedimento a seguir preenche a região com 1's:

$$X_k = (X_{k-1} \oplus B) \cap A^c \quad k = 1, 2, 3, \dots \quad (5.20)$$

onde $X_0 = p$ e B é o elemento estruturante simétrico mostrado na parte (c) da figura 10. O algoritmo termina na k -ésima iteração se $X_k = X_{k-1}$. O conjunto união de X_k e A contém a fronteira e os pontos internos a ela.

Figura 10 - Preenchimento de regiões (*Region filling*).

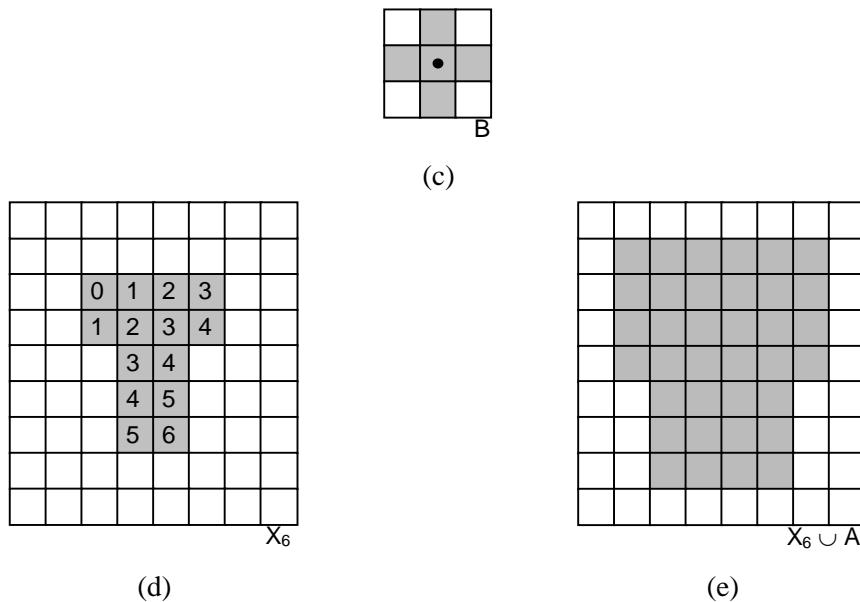


Figura 10 – Continuação.

Este procedimento é ilustrado na figura 10. Na parte (a) tem-se o conjunto original A , cujo complemento é mostrado em (b). A figura 10(c) mostra o elemento estruturante utilizado. A parte (d) indica o resultado obtido após a sexta iteração (a última que ainda produziu alguma diferença em relação à iteração anterior), em que os números indicam que iteração contribuiu para o surgimento de quais pixels no resultado parcial. Finalmente, o resultado da união do conjunto da figura 10(d) com o conjunto original é mostrado na parte (e).

5.5.3 Extração de componentes conectados

Seja Y um componente conectado contido em um conjunto A e suponha-se que um ponto p de Y é conhecido. Então, a expressão iterativa abaixo provê todos os elementos de Y :

$$X_k = (X_{k-1} \oplus B) \cap A \quad k = 1, 2, 3, \dots \quad (5.21)$$

onde $X_0 = p$ e B é o elemento estruturante adequado mostrado na parte (b) da figura 11, que ilustra a mecânica da equação acima. O algoritmo converge quando $X_k = X_{k-1}$. O valor final de X_k será atribuído a Y .

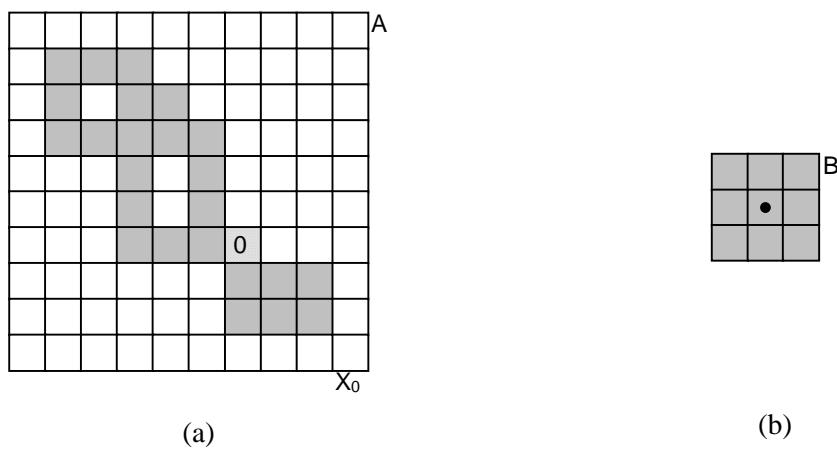


Figura 11 - Extração de componentes conectados.

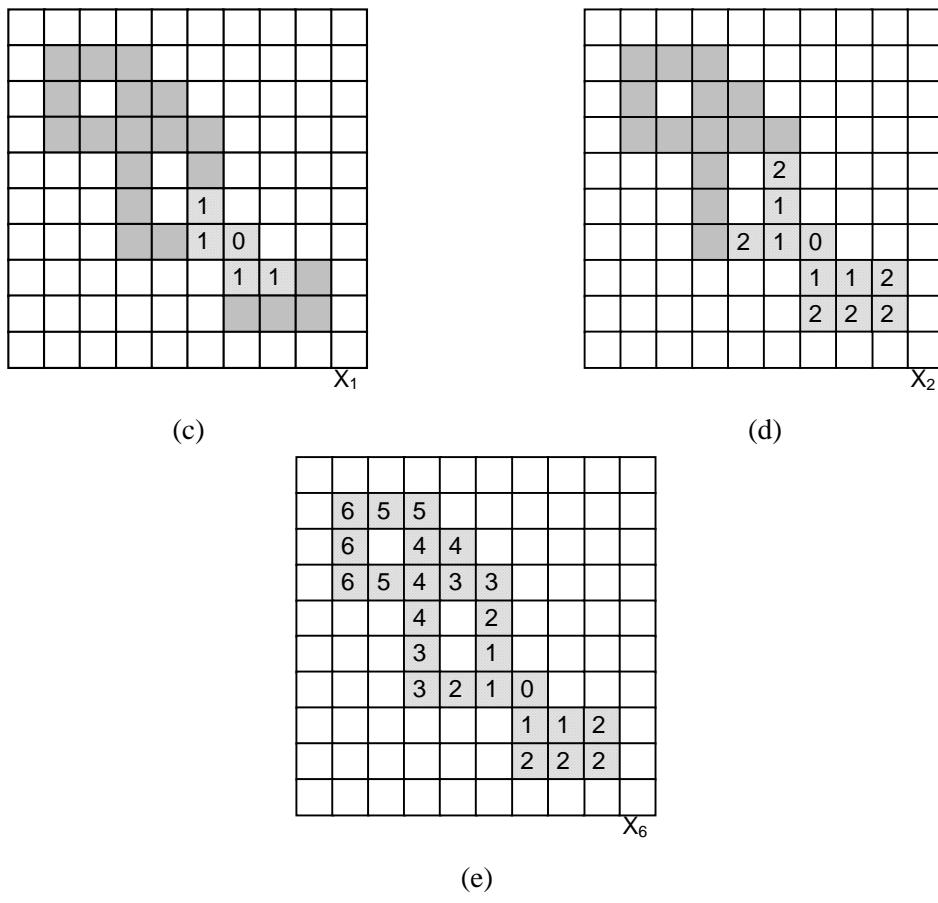


Figura 11 - Continuação.

A figura 11 mostra um exemplo de extração de componentes conectados. Em sua parte (a) são mostrados o conjunto original A e o pixel de partida, indicado pelo número 0. O elemento estruturante utilizado está na figura 11(b). As partes (b) e (c) mostram, respectivamente, os resultados após a primeira e segunda iterações. O resultado final (após 6 iterações) é mostrado na figura 11(e).

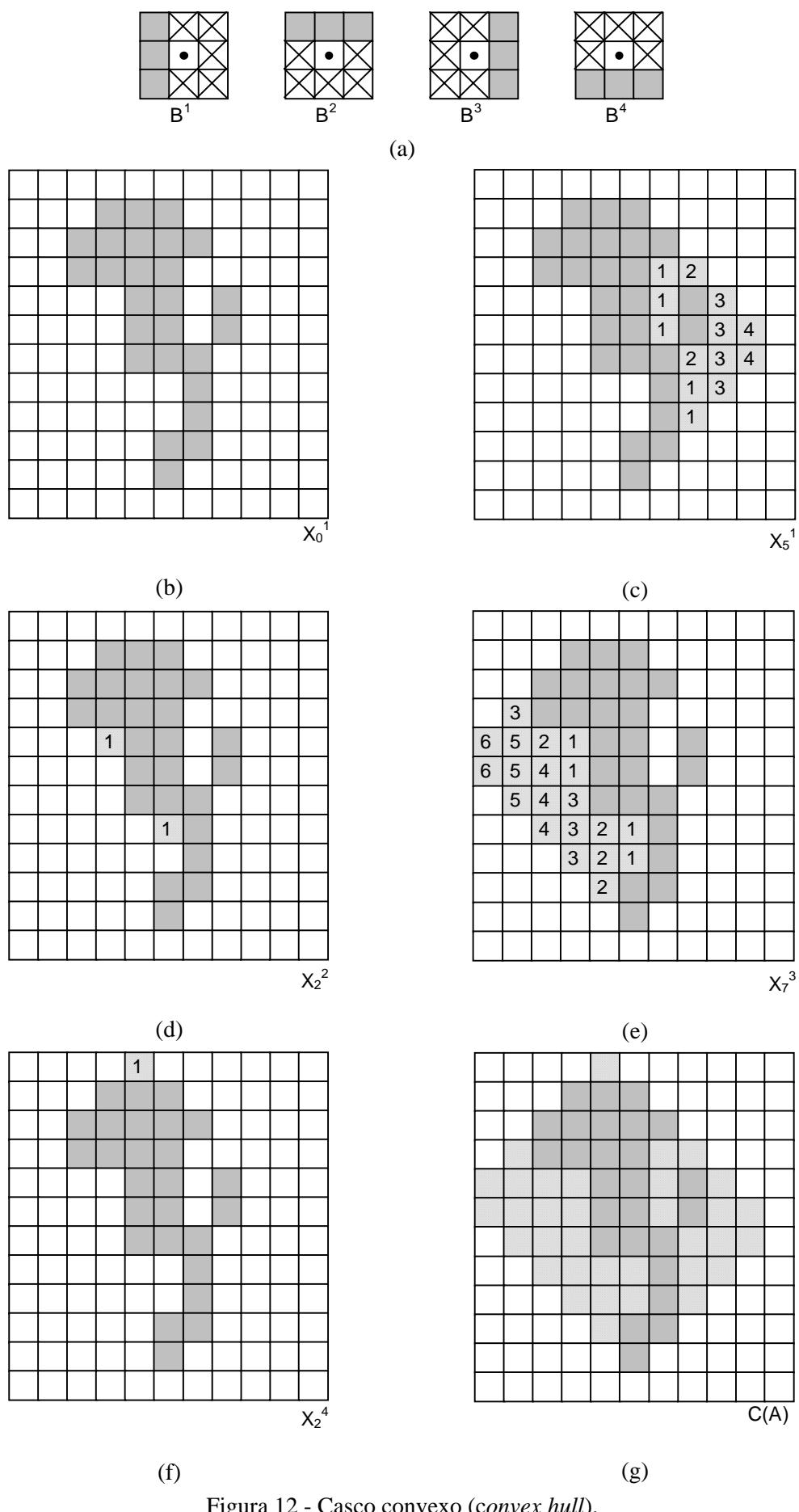
5.5.4 Casco convexo (*Convex Hull*)

Define-se casco convexo H de um conjunto arbitrário S como o menor conjunto convexo que ainda contém S . Apresentaremos a seguir um algoritmo baseado em morfologia matemática para a obtenção do casco convexo $C(A)$ de um conjunto A . Seja B^i , $i = 1, 2, 3, 4$, representando quatro elementos estruturantes. Notar que estes elementos possuem pontos indicados com X que significam uma condição '*don't care*', quer dizer, o pixel naquela posição pode ter valor 0 ou 1. O procedimento consiste em implementar a equação:

$$X_k^i = (X \text{ hom } B^i) \cup A \quad i = 1, 2, 3, 4 \quad \text{e} \quad k = 1, 2, 3, \dots \quad (5.22)$$

com $X_0^i = A$. Agora, seja $D^i = X_{\text{conv}}^i$, onde o subscrito 'conv' indica convergência no sentido que $X_k^i = X_{k-1}^i$. Então, o casco convexo de A é:

$$C(A) = \bigcup_{i=1}^4 D^i. \quad (5.23)$$

Figura 12 - Casco convexo (*convex hull*).

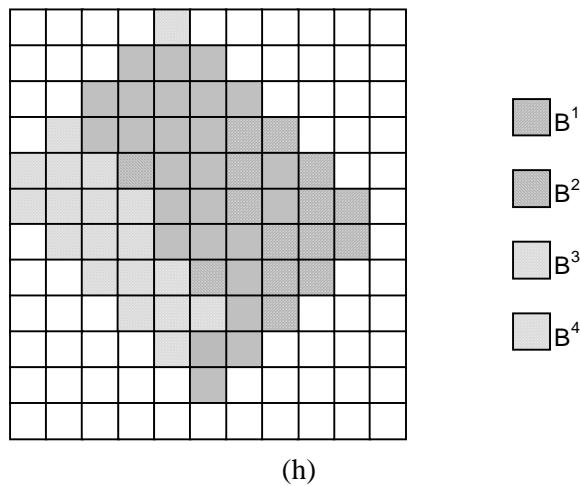


Figura 12 - Continuação.

Em outras palavras, o procedimento consiste em se aplicar iterativamente a transformada *hit-or-miss* sobre A com B^1 ; quando não houverem mais mudanças, executa-se a união com A e dá-se ao resultado o nome D^1 . O procedimento é repetido com B^2 até que não existam outras mudanças e assim sucessivamente. A união dos quatro D 's resultantes constitui o casco convexo de A .

A figura 12 mostra as etapas deste procedimento, iniciando pela exibição dos quatro elementos estruturantes utilizados, na parte (a). A parte (b) mostra o conjunto original A . As figuras 12(c), (d), (e) e (f) mostram o resultado final do processamento para cada elemento estruturante, indicando numericamente a contribuição de cada iteração no resultado final para aquele elemento. O resultado final aparece na parte (g) e é detalhado na parte (h), que ilustra a contribuição de cada elemento estruturante para o resultado final. Nesta figura, a notação X_k^i indica a k -ésima iteração (aquele em que não houve mudança em relação a X_{k-1}^i).

5.5.5 Afinamento (*Thinning*)

O afinamento de um conjunto A por um elemento estruturante B , denotado $A \otimes B$, pode ser definido com a ajuda da transformada *hit-or-miss*:

$$\begin{aligned} A \otimes B &= A - (A \text{ hom } B) \\ &= A \cap (A \text{ hom } B)^c. \end{aligned} \quad (5.24)$$

Outra expressão para o afinamento de A é baseada em uma seqüência de elementos estruturantes:

$$\{B\} = \{B^1, B^2, B^3, \dots, B^n\} \quad (5.25)$$

onde B^i é uma versão rotacionada de B^{i-1} . Usando este conceito, define-se o afinamento por uma seqüência de elementos estruturantes como:

$$A \otimes \{B\} = (((((A \otimes B^1) \otimes B^2) \dots) \otimes B^n)). \quad (5.26)$$

Em outras palavras, o processo consiste em afinar A por um passo com B^1 , então afinar o resultado com um passo de B^2 e assim sucessivamente até que A seja afinado com um passo de B^n . O processo todo é repetido até que não ocorram outras mudanças.

A figura 13 mostra todas as etapas de afinamento de uma imagem usando oito elementos estruturantes, indicados no topo da figura. A partir do conjunto original A , são ilustrados os resultados parciais mais relevantes, até a situação em que nenhum elemento estruturante consiga remover nenhum outro pixel da imagem. Como este resultado ainda possui conexões diagonais redundantes entre pixels pretos, estas são removidas utilizando o conceito da m -conectividade (ver capítulo 2). Notar que nesta figura, todos os pixels não representados explicitamente podem ser considerados brancos e que os números dentro de cada pixel indicam o elemento estruturante utilizado e não a iteração (como foi o caso na figura 12).

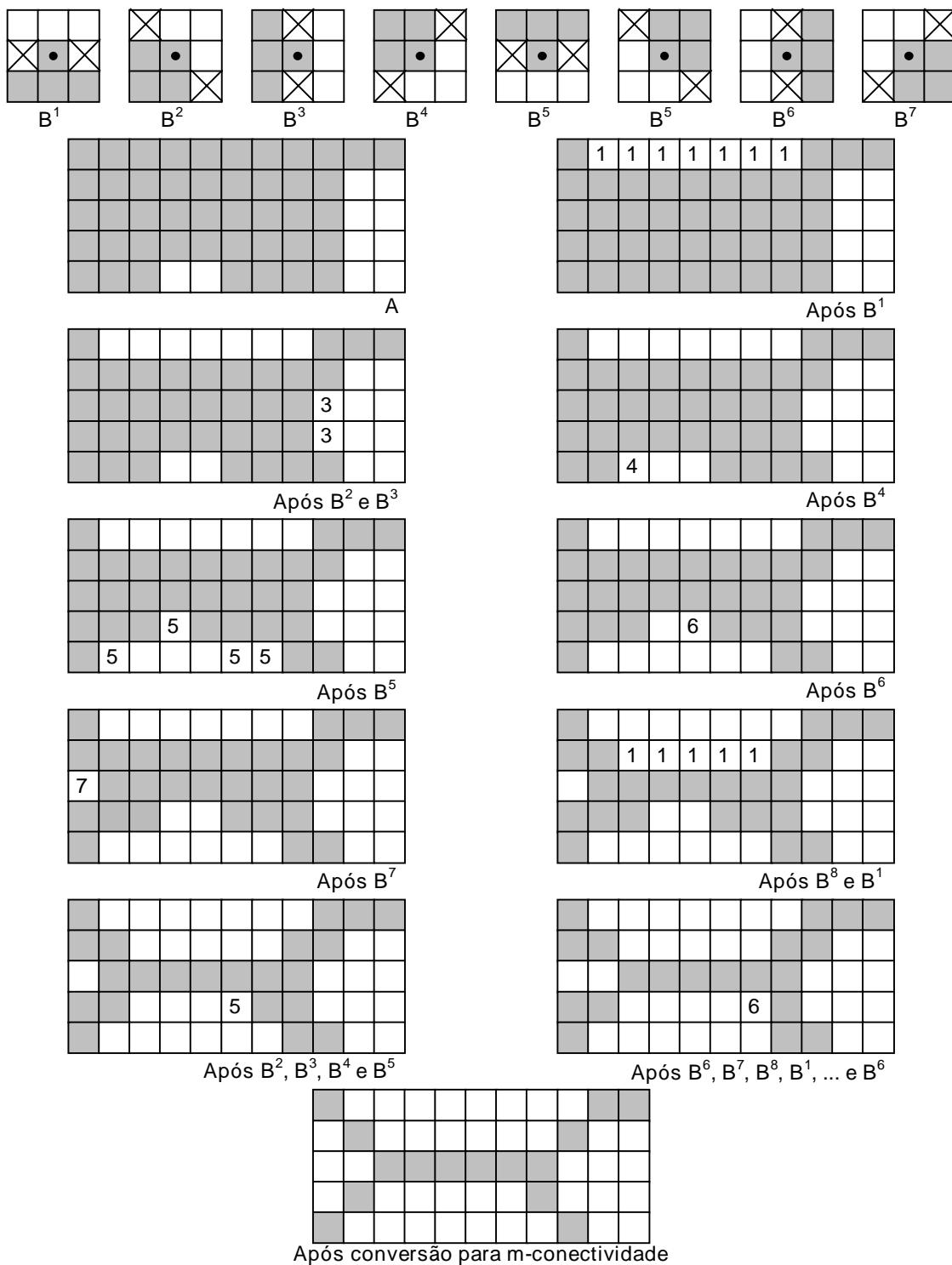


Figura 13 - Afinamento (*Thinning*).

5.5.6 Espessamento (*Thickening*)

O espessamento é o dual morfológico do afinamento e pode ser definido como:

$$A \text{ thi } B = A \cup (A \text{ hom } B) \quad (5.27)$$

onde B é um elemento estruturante adequado. O espessamento também pode ser definido como uma operação seqüencial:

$$A \text{ thi } \{B\} = (((A \text{ thi } B^1) \text{ thi } B^2) \dots) \text{ thi } B^n. \quad (5.28)$$

Os elementos estruturantes usados para o espessamento têm o mesmo formato dos usados para afinamento (ver figura 13), porém com os 1's e 0's intercambiados. Entretanto, um algoritmo separado para espessamento raramente é usado. O procedimento usual é afinar o fundo do conjunto e complementar o resultado. Ou seja, para espessar o conjunto A , faz-se $C = A^c$, afina-se C , e então obtém-se C^c . A figura 14 mostra o processo de espessamento, obtido pelo afinamento do complemento de A , onde a parte (a) representa a imagem original, a parte (b) seu complemento e a parte (c) a versão afinada do complemento de A . Na parte (d) da figura observa-se que este procedimento resulta em alguns pixels isolados, que são removidos em uma etapa complementar de processamento, como mostra a parte (e).

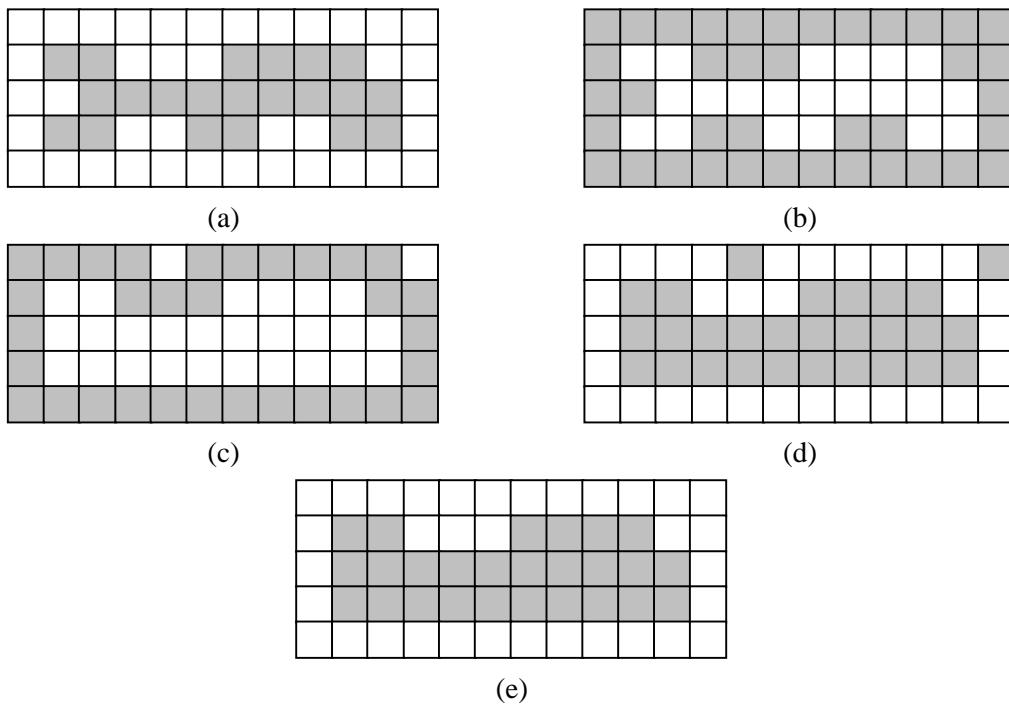


Figura 14 - Espessamento (*thickening*) através do afinamento do fundo.

5.5.7 Esqueletos

Uma abordagem importante para representar a forma estrutural de uma região plana é reduzi-la a um grafo. Esta redução pode ser implementada obtendo o esqueleto da região através de um algoritmo de afinamento (também denominado esqueletização).

O esqueleto de uma região pode ser definido usando a Transformação do Eixo Médio (MAT - *Medial Axis Transformation*), proposta por Blum [Blum 1967] e definida como:

"A MAT de uma região R com fronteira B é obtida da seguinte forma: para cada ponto p em R , encontra-se seu vizinho mais próximo em B . Se p tem mais de um vizinho à mesma

distância mínima, diz-se que p pertence ao eixo médio (esqueleto) de R^n . A figura 15 mostra exemplos de MAT usando a distância euclidiana.

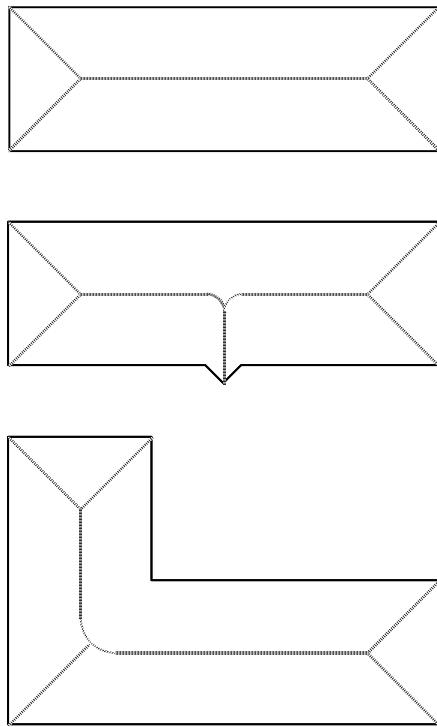


Figura 15 - Exemplos de esqueletos obtidos através do conceito da MAT para três regiões simples.

A obtenção do esqueleto de um objeto plano também é possível através de técnicas morfológicas. Lantuéjoul [Lantuéjoul 1980] demonstrou que o esqueleto de um conjunto (região) A pode ser expresso em termos de erosões e aberturas. Isto é, sendo $S(A)$ o esqueleto de A , pode-se provar que:

$$S(A) = \bigcup_{k=0}^K S_k(A) \quad (5.29)$$

com

$$S_k(A) = \bigcup_{k=0}^K \left\{ (A \Theta kB) - [(A \Theta kB) \circ B] \right\} \quad (5.30)$$

onde B é um elemento estruturante, $(A \Theta kB)$ indica k erosões sucessivas de A ; ou seja,

$$(A \Theta kB) = ((\dots (A \Theta B) \Theta B) \dots) \Theta B \quad (5.31)$$

k vezes, e K é o último passo iterativo antes de A resultar, por erosão, em um conjunto vazio. Em outras palavras,

$$K = \max \left\{ k \mid (A \Theta kB) \neq \emptyset \right\}. \quad (5.32)$$

Estas equações indicam que $S(A)$, o esqueleto de A , pode ser obtido pela união dos subconjuntos esqueletos $S_k(A)$. Pode-se mostrar, além disso, que A pode ser reconstruído a partir destes subconjuntos utilizando a equação

$$A = \bigcup_{k=0}^K (S_k(A) \oplus kB) \quad (5.33)$$

onde $(S_k(A) \oplus kB)$ denota k dilatações sucessivas de $S_k(A)$; ou seja,

$$(S_k(A) \oplus kB) = ((\dots(S_k(A) \oplus B) \oplus B) \dots) \oplus B \quad (5.34)$$

k vezes, sendo o limite K de uniões o mesmo utilizado anteriormente.

A figura 16 mostra estes conceitos. Nesta figura, a imagem original está na primeira posição da primeira coluna e sua versão afinada na última posição da quarta coluna. Notar que o resultado não satisfaz os requisitos de um algoritmo de afinamento, que são:

- não remover pontos terminais de um segmento;
- não violar a conectividade da imagem original;
- não causar erosão excessiva da região.

A imagem resultante tem dois graves problemas: é mais espessa do que deveria ser e, pior ainda, remove a conectividade dos elementos da imagem original. Por estes motivos, apesar da elegante formulação matemática dos algoritmos morfológicos, os algoritmos heurísticos, como os descritos em [Zhang e Suen 1984], [Naccache e Shingal 1984] — comentado em [Fu et al. 1987] — e [Perrotti e Lotufo 1992], muitas vezes produzem melhores resultados. A interseção da última linha com a última coluna contém o resultado da reconstrução do conjunto A .

$k \setminus$	$A \ominus kB$	$(A \ominus kB) \circ B$	$S_k(A)$	$\bigcup_{k=0}^K S_k(A)$	$S_k(A) \oplus kB$	$\bigcup_{k=0}^K S_k(A) \oplus kB$
0						
1						
2						

Figura 16 - Esqueletos.

Leitura complementar

O capítulo 4 de [Dougherty 1994] apresenta um bom resumo dos temas afinamento e esqueletização.

O capítulo 9 de [Pavlidis 1982] é inteiramente dedicado a algoritmos de afinamento.

5.5.8 Poda (*Pruning*)

Os métodos de poda são complementos essenciais dos algoritmos de afinamento, uma vez que estes, em geral, deixam componentes parasitas que devem ser removidos em uma etapa de pós-processamento. O conceito de poda será apresentado através de um exemplo. Seja um caractere cujo esqueleto contém pontos espúrios causados por diferentes espessuras nos traços do caractere original (antes do afinamento), mostrado na figura 17(b). Assumindo que os componentes parasitas têm comprimento menor ou igual a três pixels, neste caso podem ser definidos dois elementos estruturantes (cada qual com suas versões rotacionadas de 90°) projetados para detetar pontos terminais, mostrados na parte (a). Notar que quatro destes elementos têm duas quadrículas indicadas com X que significam uma condição '*'don't care'*', quer dizer, o pixel naquela posição pode ter valor 0 ou 1. A primeira etapa será afinar o conjunto A com os elementos estruturantes B, obtendo o conjunto X₁, isto é:

$$X_1 = A \otimes \{B\} \quad (5.35)$$

onde {B} denota a seqüência dos oito elementos estruturantes.

Aplicando a equação (5.35), por exemplo, três vezes,¹ produz-se o resultado X₁, mostrado na figura 17(c). O passo seguinte é “restaurar” o caractere a sua forma original, porém com os ramos parasitas removidos. Para fazê-lo, inicialmente se constrói o conjunto X₂, contendo todos os pontos terminais de X₁ (figura 17(d)):

$$X_2 = \bigcup_{k=1}^8 (X_1 \text{ hom } B^k) \quad (5.36)$$

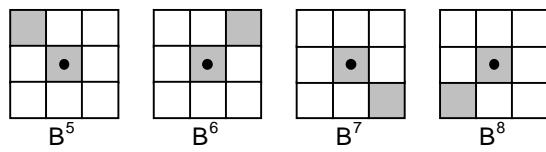
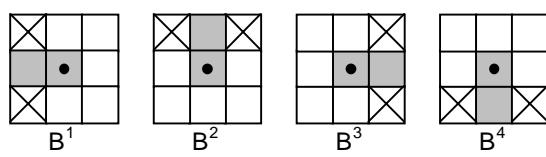
onde os B_k são os mesmos detetores de pontos terminais usados anteriormente. O passo seguinte é a dilatação dos pontos terminais três vezes, usando o conjunto A como delimitador:

$$X_3 = (X_2 \oplus H) \cap A \quad (5.37)$$

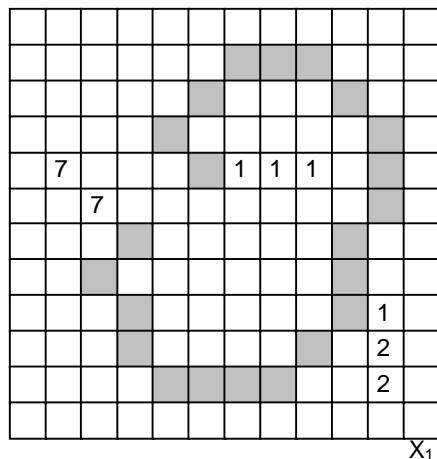
onde H é um elemento estruturante de 3 x 3 composto por 1's. Este tipo de dilatação previne a criação de pontos de valor 1 fora da região de interesse, como se pode comprovar na parte (e) da figura 17. Finalmente, a união de X₃ e X₁ dá o resultado final (figura 17(f)):

$$X_4 = X_1 \cup X_3 \quad (5.38)$$

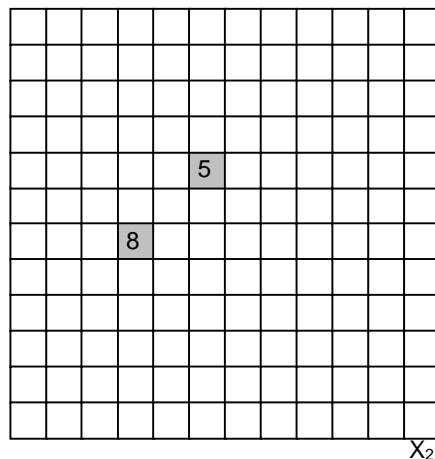
¹ Neste caso, o algoritmo é aplicado três vezes porque se está supondo que os ramos parasitas têm comprimento menor ou igual a 3 pixels.



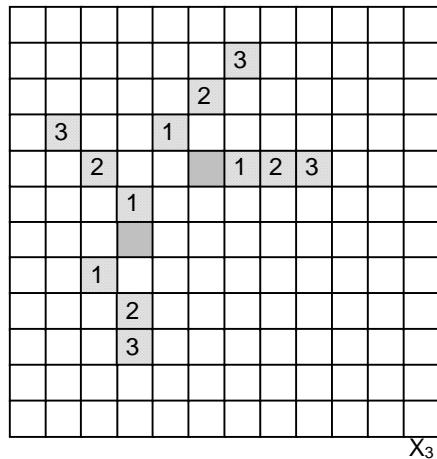
(a)

 X_1

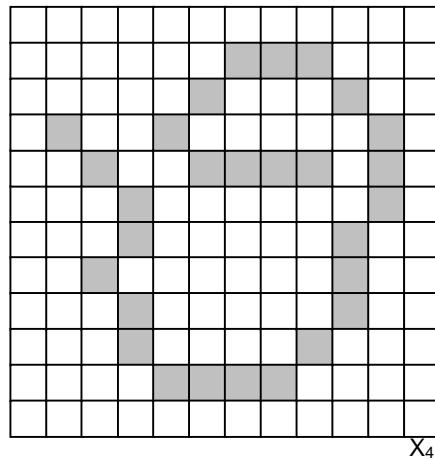
(b)

 X_2

(c)

 X_3

(d)

 X_4

(e)

(f)

Figura 17 - Poda (*pruning*).

Outros problemas práticos típicos de processamento de imagens podem ser resolvidos através da combinação dos algoritmos morfológicos básicos aqui apresentados. A tabela 1 apresenta um resumo das operações morfológicas vistas neste capítulo e seus resultados.

Tabela 1 - Sumário dos algoritmos morfológicos e seus resultados (adaptado de [Gonzalez e Woods 1992])

Operação	Equação	Comentários ²
Translação	$(A)_x = \{c \mid c = a + x, \text{ para } a \in A\}$	Translada a origem de A para o ponto x .
Reflexão	$\hat{B} = \{x \mid x = -b, \text{ para } b \in B\}$	Reflete todos os elementos de B em relação à origem deste conjunto.
Complemento	$A^c = \{x \mid x \notin A\}$	Conjunto de pontos que não estão em A .
Diferença	$A - B = \{x \mid x \in A, x \notin B\} = A \cap B^c$	Conjunto de pontos que pertencem a A porém não a B .
Dilatação	$A \oplus B = \{x \mid (\hat{B})_x \cap A \neq \emptyset\}$	'Expande' a fronteira de A . (I)
Erosão	$A \ominus B = \{x \mid (B)_x \subseteq A\}$	'Contraí' a fronteira de A . (I)
Abertura	$A \circ B = (A \ominus B) \oplus B$	Suaviza contornos, quebra istmos estreitos e elimina pequenas ilhas e picos agudos. (I)
Fechamento	$A \bullet B = (A \oplus B) \ominus B$	Suaviza contornos, une quebras estreitas e golfos longos e delgados e elimina pequenos orifícios. (I)
Transformada <i>hit-or-miss</i>	$A \text{ hom } B = (A \ominus B_1) \cap (A^c \ominus B_2)$ $= (A \ominus B_1) - (A \ominus \hat{B}_2)$	O conjunto de pontos (coordenadas) nos quais, simultaneamente, B^1 encontrou uma correspondência (<i>hit</i>) em A e B^2 encontrou uma correspondência em A^c .
Extração de contorno	$\beta(A) = A - (A \ominus B)$	Conjunto de pontos sobre a fronteira do conjunto A . (I)

²Os números em algarismos romanos entre parênteses referem-se aos elementos estruturantes usados no processo morfológico. Estes elementos estruturantes estão na figura 18.

Operação	Equação	Comentários ²
Preenchimento de uma região (Region filling)	$X_k = (X_{k-1} \oplus B) \cap A^c;$ $X_0 = p$ e $k = 1, 2, 3, \dots$	Preenche uma região em A , dado um ponto p na região. (II)
Componentes conectados	$X_k = (X_{k-1} \oplus B) \cap A$ $X_0 = p$ e $k = 1, 2, 3, \dots$	Encontra um componente conectado E em A , dado um ponto p em E . (I)
Casco convexo (Convex hull)	$X_k^i = (X_{k-1}^i \text{ hom } B^i) \cup A;$ $i = 1, 2, 3, 4, \dots k = 1, 2, 3, \dots,$ $X_0^i = A$, e $D^i = X_{conv}^i.$ $C(A) = \bigcup_{i=1}^4 D^i$	Obtém o casco convexo $C(A)$ do conjunto A , onde $conv$ indica convergência no sentido de que $X_k^i = X_{k-1}^i$. (III)
Afinamento (Thinning)	$A \otimes B = A - (A \text{ hom } B)$ $= A \cap (A \text{ hom } B)^c$ $A \otimes \{B\} = (((A \otimes B^1) \otimes B^2) \dots) \otimes B^n)$	Afina o conjunto A . As duas primeiras equações fornecem a definição morfológica básica de afinamento. As duas últimas mostram o afinamento através de uma seqüência de elementos estruturantes. Na prática, normalmente este último método é usado. (IV)
Espessamento (Thickening)	$A \text{ thi } B = A \cup (A \text{ hom } B)$ $A \text{ thi } \{B\} = (((A \text{ thi } B^1) \text{ thi } B^2) \dots) \text{ thi } B^n)$	Espessa o conjunto A . (Ver os comentários anteriores sobre seqüências de elementos estruturantes). Usa (IV) com os 0's e 1's intercambiados.
Esqueletos	$S(A) = \bigcup_{k=0}^K S_k(A)$ $S_k(A) = \bigcup_{k=0}^K \{(A \Theta kB) - [(A \Theta kB) \circ B]\}$ $A = \bigcup_{k=0}^K (S_k(A) \oplus kB)$	Busca o esqueleto $S(A)$ do conjunto A . A última equação indica que A pode ser reconstruído a partir dos subconjuntos $S_k(A)$. Em todas as três equações, K é o valor do passo iterativo depois do qual o conjunto A resulta, por erosão, em um conjunto vazio. A notação $(A \Theta kB)$ denota a k -ésima iteração de erosão sucessiva. (I)

Operação	Equação	Comentários ²
Poda (<i>Pruning</i>)	$X_1 = A \otimes \{B\}$	
	$X_2 = \bigcup_{k=1}^8 (X_1 \text{ hom } B^k)$	
	$X_3 = (X_2 \oplus H) \cap A$	
	$X_4 = X_1 \cup X_3$	X_4 é o resultado da poda aplicada ao conjunto A . O número de vezes que a primeira equação deve ser aplicada para obter X_1 deve ser especificada. Os elementos estruturantes (V) são utilizados para as duas primeiras equações. A terceira equação usa o elemento estruturante (I).

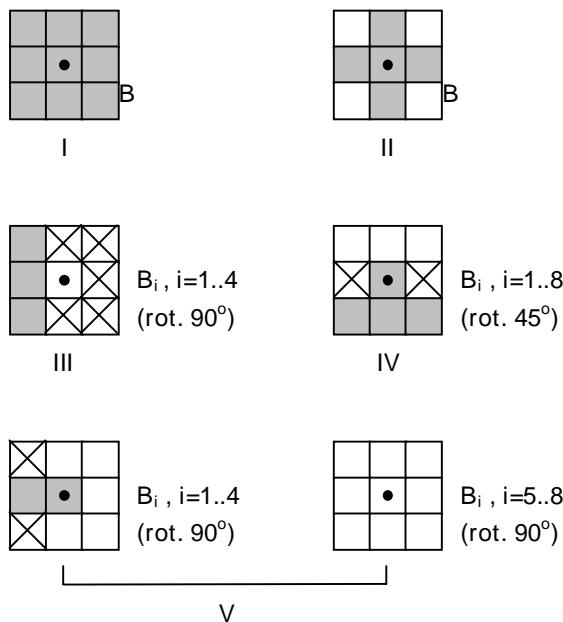


Figura 18 - Os cinco elementos estruturantes utilizados.

Leitura complementar

A aplicação dos conceitos de morfologia matemática a problemas de visão por computador segue sendo um tema de investigação em nível mundial. Os periódicos especializados contêm inúmeros trabalhos científicos desenvolvidos a partir das idéias básicas resumidas neste capítulo.

O capítulo 11 de [Serra 1982] apresenta algoritmos morfológicos de esqueletização, obtenção do MAT, afinamento, espessamento e extração de componentes conectados.

Para aqueles que pretendem desenvolver software usando os conceitos aqui apresentados, também são de grande interesse os livros de Giardina e Dougherty [Giardina e Dougherty 1988] e Dougherty [Dougherty 1992].

O capítulo 4 de [Dougherty e Giardina 1987] é exclusivamente dedicado à Morfologia Matemática.

Os artigos de Haralick et al. [Haralick et al. 1987] e Maragos [Maragos 1987] servem de referência para o estudo de algoritmos morfológicos aplicados a Processamento de Imagens.

O capítulo 8 de [Dougherty 1993] e o capítulo 4 de [Facon 1996] são inteiramente dedicados a algoritmos morfológicos.

O artigo de Jang e Chin [Jang e Chin 1990] aborda em detalhes o processo de afinamento usando Morfologia Matemática, apresentando e comparando dois algoritmos distintos para esta tarefa.

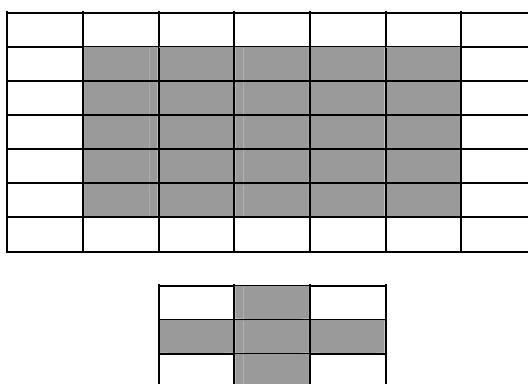
No capítulo 8 de [Gonzalez e Woods 1992] encontramos um exemplo de aplicação dos algoritmos morfológicos básicos às etapas de pré-processamento de um sistema de reconhecimento óptico de caracteres (OCR) capaz de ler o código postal norte-americano (*zip code*) em um envelope.

Em [Jain 1989] encontramos uma aplicação do processamento morfológico em uma aplicação que trata da inspeção de placas de circuito impresso. A imagem original é limiarizada, reduzida a um pixel de espessura usando um algoritmo morfológico de afinamento e o resultado é submetido a uma etapa de poda.

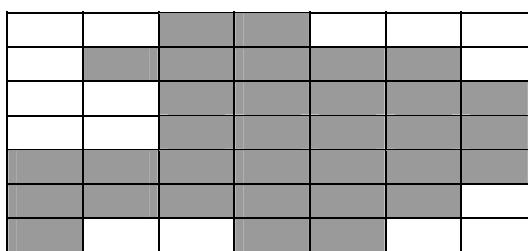
Recomendamos ainda a leitura do capítulo 7 de [Russ 1995], que traz muitos outros exemplos de aplicação das operações e algoritmos morfológicos básicos apresentados neste capítulo.

Exercícios Propostos

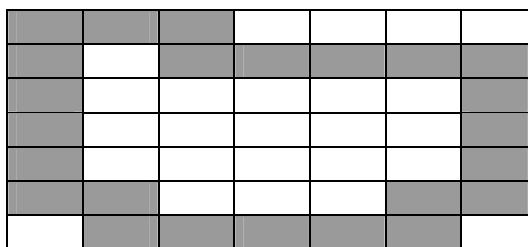
- Seja o algoritmo de esqueletização usando técnicas morfológicas descrito na Seção 5.5.7. Dada a imagem binária a seguir e o elemento estruturante abaixo dela, representar os resultados intermediários de $(A \Theta kB)$, $(A \Theta kB)^oB$, $S_k(A)$ e $\bigcup_{k=0}^K S_k(A)$ e explicar porque neste caso $K=2$.



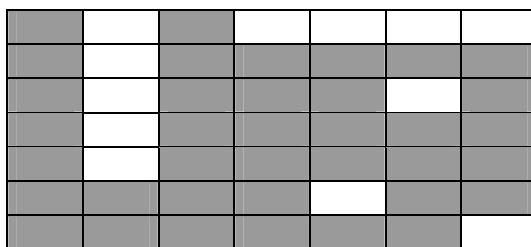
- Extrair o contorno do objeto a seguir usando o algoritmo morfológico descrito na Seção 5.5.1, utilizando uma matriz 3 x 3 de pixels pretos como elemento estruturante e indicando os resultados intermediários e final.



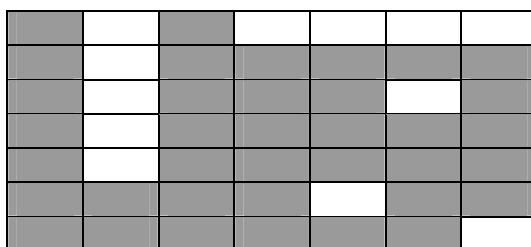
- Preencher o interior do objeto a seguir usando o algoritmo morfológico descrito na Seção 5.5.2, utilizando uma matriz 3 x 3 em forma de cruz como elemento estruturante e indicando os resultados intermediários e final.



4. Realizar a abertura do objeto a seguir utilizando uma matriz 3 x 3 com todos os pixels pretos como elemento estruturante e indicando os resultados intermediários e final.



5. Realizar o fechamento do objeto a seguir utilizando uma matriz 3 x 3 com todos os pixels pretos como elemento estruturante e indicando os resultados intermediários e final.



No computador

Para complementar o conteúdo deste capítulo, sugerimos a prática de laboratório nº 7 (Apêndice B).

Na Internet

["http://www.ime.usp.br/mac/khoros/mmach.old/tutor/mmach.html"](http://www.ime.usp.br/mac/khoros/mmach.old/tutor/mmach.html)

A Tutorial on Mathematical Morphology

Tutorial interativo elaborado pela USP.

["http://www.khoral.com/dipcourse/dip17sep97/html-dip/c9/s4/front-page.html"](http://www.khoral.com/dipcourse/dip17sep97/html-dip/c9/s4/front-page.html)

Dilation, Erosion, Opening, Closing

Página com explicações e ilustrações dos conceitos de dilatação, erosão, abertura e fechamento.

["http://www.khoral.com/dipcourse/dip17sep97/html-dip/c9/s5/front-page.html"](http://www.khoral.com/dipcourse/dip17sep97/html-dip/c9/s5/front-page.html)

Contours

Descreve o processo de extração de contornos usando morfologia matemática.

Bibliografia

[Blum 1967]

Blum, H., "A Transformation for Extracting New Descriptors of Shape" in Wathen-Dunn, W. (ed.), *Models for the Perception of Speech and Visual Form*, MIT Press, 1967.

[Dougherty 1992]

Dougherty, E.R., *An Introduction to Morphological Image Processing*, SPIE Press, 1992.

- [Dougherty 1993] Dougherty, E.R. (ed.), *Mathematical Morphology in Image Processing*, Marcel Dekker, 1993.
- [Dougherty 1994] Dougherty, E.R. (ed.), *Digital Image Processing Methods*, Marcel Dekker, 1994.
- [Dougherty e Giardina 1987] Dougherty, E.R. e Giardina, C.R., *Matrix Structured Image Processing*, Prentice-Hall, 1987.
- [Facon 1996] Facon, J., *Morfologia Matemática: Teoria e Exemplos*, Editora Universitária Champagnat, PUC-PR, 1996.
- [Fu et al. 1987] Fu, K.S., Gonzalez, R. e Lee, C., *Robotics: Control, Sensing, Vision and Intelligence*, McGraw-Hill, 1987.
- [Giardina e Dougherty 1988] Giardina, C.R. e Dougherty, E.R., *Morphological Methods in Image and Signal Processing*, Prentice-Hall, 1988.
- [Gonzalez e Woods 1992] Gonzalez, R.C. e Woods, R.E., *Digital Image Processing – Third Edition*, Addison-Wesley, 1992.
- [Haralick e Shapiro 1992] Haralick, R.M. e Shapiro, L.G., *Computer and Robot Vision - Volume 1*, Addison-Wesley, 1992.
- [Haralick et al. 1987] Haralick, R.M., Sternberg, S.R. e Zhuang, X., “Image Analysis Using Mathematical Morphology”, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 9, 4, 532-550.
- [Jain 1989] Jain, A.K., *Fundamentals of Digital Image Processing*, Prentice-Hall, 1989.
- [Jang e Chin 1990] Jang, B.K. e Chin, R.T., "Analysis of Thinning Algorithms Using Mathematical Morphology", *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12, 6, Junho 1990, 541-551.
- [Lantuéjoul 1980] Lantuéjoul, C., “Skeletonization in Quantitative Metallography” in Haralick, R.M. & Simon, J.C. (eds.), *Issues of Digital Image Processing*, Sijthoff and Noordhoff, 1980.
- [Maragos 1987] Maragos, P., “Tutorial on Advances in Morphological Image Processing and Analysis”, *Optical Engineering*, 26, 7, 623-632.
- [Naccache e Shingal 1984] Naccache, N.J. e Shingal, R., “SPTA: A Proposed Algorithm for Thinning Binary Patterns”, *IEEE Transactions on Systems, Man and Cybernetics*, 14, 3, 409-418.
- [Pavlidis 1982] Pavlidis, T., *Algorithms for Graphics and Image Processing*, Computer Science Press, 1982.
- [Perrotti e Lotufo 1992] Perrotti, F.A. e Lotufo, R.A., “Um novo algoritmo paralelo de afinamento”, *Anais do V SIBGRAPI*, Novembro 1992, 285-293.
- [Ross e Wright 1992] Ross, K.A. e Wright, C.R.B., *Discrete Mathematics - 3rd ed.*, Prentice-Hall, 1992.

- [Russ 1995] Russ, J. C., *The Image Processing Handbook - 2nd ed.*, CRC Press, 1995.
- [Serra 1982] Serra, J., *Image Analysis and Mathematical Morphology* Academic Press, 1982.
- [Serra 1988] Serra, J., *Image Analysis and Mathematical Morphology - vol. 2* Academic Press, 1988.
- [Zhang e Suen 1984] Zhang, T.Y e Suen, C.Y. , “A Fast Parallel Algorithm for Thinning Digital Patterns”, *Communications of the ACM*, 27, 3, 236-239.

Capítulo 6

Compressão e Codificação de Imagens

Um dos maiores desafios do processamento de imagens é contornar o problema da grande quantidade de bytes necessários para armazenar ou transmitir à distância uma imagem digitalizada. Tal problema é agravado substancialmente quando se trabalha com imagens em movimento (vídeo digital) e em ambos os casos requer a ação de algoritmos e técnicas que permitam a redução da quantidade de dados necessária para representar uma imagem ou seqüência de imagens.

As técnicas de compressão de imagens buscam atingir tal redução partindo de uma premissa básica, que é a remoção de informação redundante. O interesse pelas técnicas de compressão de imagens remonta a quase meio século atrás (na época, utilizando técnicas analógicas) e é cada vez maior, graças à popularização da multimídia, sua extensão a sistemas geograficamente distribuídos, e a inúmeros novos inventos que necessitam da tecnologia de compressão de imagens para se tornarem viáveis, como por exemplo: a videoconferência, a TV de alta definição (HDTV), a TV interativa e o vídeo sob demanda (*video on demand - VOD*).

Neste capítulo, abordaremos aspectos teóricos e práticos de compressão e codificação de imagens. As Seções 6.1 a 6.3 apresentam os fundamentos teóricos necessários para a compreensão dos princípios e técnicas de compressão e codificação de dados de qualquer natureza e que, portanto, são também aplicáveis à situação em que estes dados são imagens. A Seção 6.1 concentra-se em explicar o conceito de redundância e como ela pode ser explorada pelos algoritmos de compressão de imagens. A Seção 6.2 apresenta os modelos de compressão e descompressão de imagens, enquanto a Seção 6.3 revisa conceitos e teoremas fundamentais da Teoria da Informação. As Seções 6.4 a 6.6 são dedicadas aos aspectos mais práticos das técnicas de compressão de imagens. Estas técnicas costumam ser divididas em dois grandes grupos: aquelas nas quais toda informação original é preservada (chamadas de técnicas de compressão sem perdas) e aquelas em que ocorre uma perda tolerável de informação, com o objetivo de atingir maiores taxas de compressão (técnicas com perdas). Na Seção 6.4 são resenhadas técnicas de compressão sem perdas, enquanto a Seção 6.5 apresenta diversas técnicas de compressão com perdas. Finalmente, a Seção 6.6 apresenta os fundamentos de alguns dos mais conhecidos padrões de compressão adotados mundialmente.

6.1 Fundamentos

O termo 'compressão de dados' refere-se ao processo de redução do montante de dados exigidos para representar uma dada quantidade de informação. Deve-se esclarecer que denominamos 'dados' aos meios pelos quais uma informação é transmitida. Várias quantidades de dados podem ser usadas para representar a mesma quantidade de informação. Tal pode ser o caso, por exemplo, de uma pessoa prolixo e uma outra que vai direto ao assunto, que ao final de suas falas tenham relatado a mesma história. Neste caso, a informação que interessa é a história; as palavras são os dados utilizados para relatar as informações. Se as duas pessoas utilizarem um número diferente de palavras para relatar a mesma história, basicamente tem-se duas diferentes versões e pelo menos uma delas inclui dados não essenciais. Em outras palavras, ela contém dados (ou palavras) que podem tanto não fornecer informações relevantes, como simplesmente reafirmar o que já é sabido, ou seja, o que se costuma denominar 'redundância de dados'.

A redundância de dados é um aspecto essencial no estudo de compressão de imagens digitais. Para quantificá-lo matematicamente, suponhamos que n_1 e n_2 representam o número de unidades portadoras de informações em dois conjuntos de dados que representem a mesma informação. Neste caso, a 'redundância relativa' (R_D) do primeiro conjunto de dados (aquele representado por n_1) poderá ser definida como

$$R_D = 1 - \frac{1}{C_R} \quad (6.1)$$

onde o parâmetro C_R , comumente chamado de 'razão (ou taxa) de compressão', é

$$C_R = \frac{n_1}{n_2} \quad (6.2)$$

No caso em que $n_2 = n_1$, $C_R = 1$ e $R_D = 0$, podemos concluir que o primeiro conjunto de dados não contém nenhum dado redundante em relação ao segundo. Quando $n_2 \ll n_1$, $C_R \rightarrow \infty$ e $R_D \rightarrow 1$, haverá significativa compressão de dados altamente redundantes. Finalmente, no caso em que $n_2 > n_1$, $C_R \rightarrow 0$ e $R_D \rightarrow -\infty$, podemos concluir que o segundo conjunto de dados contém muito mais dados do que o primeiro, representando, obviamente, o caso de expansão de dados, normalmente indesejado. Em geral, C_R e R_D situam-se nos intervalos abertos $(0, \infty)$ e $(-\infty, 1)$, respectivamente. Uma razão de compressão comum na prática, como 10 (ou 10:1) significa que o primeiro conjunto de dados tem 10 unidades de informação (p. ex. bits) para cada unidade no segundo conjunto de dados (comprimido). A redundância correspondente (neste caso, 0,9) significa que 90% dos dados no primeiro conjunto de dados são redundantes.

Na compressão de imagens digitais, três redundâncias básicas de dados podem ser identificadas e exploradas: redundância de codificação, redundância interpíxel, e redundância psicovisual. A compressão de dados é efetivamente obtida quando uma ou mais dessas redundâncias são reduzidas ou eliminadas.

6.1.1 Redundância de Codificação

No Capítulo 3 apresentamos os conceitos de aprimoramento da qualidade de uma imagem através da modificação de seu histograma, partindo da premissa de que os níveis de cinza de uma imagem são quantidades aleatórias. Mostramos que uma grande quantidade de informação sobre a aparência de uma imagem poderia ser obtida a partir de um histograma de seus níveis de cinza. Nesta seção, utilizaremos uma formulação matemática similar para mostrar como o histograma de níveis de cinza de uma imagem também pode auxiliar na elaboração de códigos para reduzir a quantidade de dados usada para representá-la.

Consideremos, mais uma vez, que uma variável aleatória discreta r_k no intervalo $[0, 1]$ representa os níveis cinza de uma imagem e que cada r_k ocorre com probabilidade $p_r(r_k)$. Como no Capítulo 3,

$$p_r(r_k) = \frac{n_k}{n} \quad k = 0, 1, 2, \dots, L - 1 \quad (6.3)$$

onde L é o número dos níveis cinza, n_k é o número de vezes que o nível cinza k aparece na imagem, e n é o número total de pixels na imagem. Se o número de bits utilizado para representar cada valor de r_k é $l(r_k)$, a quantidade média de bits exigida para representar cada pixel é

$$L_{avg} = \sum_{k=0}^{L-1} l(r_k) p_r(r_k) \quad (6.4)$$

Em resumo, o comprimento médio das palavras-código atribuídas aos diversos valores de tom de cinza é calculado através da soma do produto do número de bits utilizados para representar cada nível de cinza pela probabilidade de ocorrência daquele nível. Assim, o número total de MARQUES FILHO, Ogê; VIEIRA NETO, Hugo. *Processamento Digital de Imagens*, Rio de Janeiro: Brasport, 1999. ISBN 8574520098.

bits exigido para codificar uma imagem de dimensões $M \times N$ é MNL_{avg} . A representação dos níveis de cinza de uma imagem com um código binário natural de m bits reduz o lado direito da eq. (6.4) para m bits. Em outras palavras, $L_{avg} = m$ quando $l(r_k)$ for substituído por m na eq. (6.4).

Exemplo

Seja uma imagem monocromática de 8 tons de cinza, distribuídos conforme a tabela 1. A representação gráfica de seu histograma é mostrada na figura 1.

Supondo que cada tom de cinza desta imagem seja codificado por um código natural de 3 bits (código 1), o resultado poderia ser aquele indicado na terceira coluna da tabela 1, para o qual L_{avg} é igual a 3 bits. Porém, se utilizássemos o código 2, indicado na quinta coluna da tabela 1, o número médio de bits necessário para codificar cada pixel da imagem seria reduzido para:

$$\begin{aligned} L_{avg} &= \sum_{k=0}^7 l_2(r_k) p_r(r_k) \\ &= 2(0,26) + 2(0,18) + 2(0,22) + 3(0,15) + 4(0,08) \\ &\quad + 5(0,06) + 6(0,03) + 6(0,02) \\ &= 2,69 \text{ bits.} \end{aligned}$$

Utilizando a eq. (6.2) podemos calcular a razão de compressão obtida, C_R , como $3/2,69 = 1,115$. Através da eq. (6.1), calculamos a redundância como:

$$R_D = 1 - \frac{1}{1,115} = 0,103.$$

Da tabela 1, pode-se facilmente extrair a relação de proporcionalidade inversa entre a probabilidade de um certo tom de cinza e o comprimento da palavra-código correspondente no código 2.

Tabela 1 - Exemplo de codificação de imagens usando palavras-código de comprimento variável.

Nível de cinza (r_k)	$p_r(r_k)$	Código 1	$l_I(r_k)$	Código 2	$l_2(r_k)$
$r_0 = 0$	0,26	000	3	01	2
$r_1 = 1/7$	0,18	001	3	11	2
$r_2 = 2/7$	0,22	010	3	10	2
$r_3 = 3/7$	0,15	011	3	001	3
$r_4 = 4/7$	0,08	100	3	0001	4
$r_5 = 5/7$	0,06	101	3	00001	5
$r_6 = 6/7$	0,03	110	3	000001	6
$r_7 = 1$	0,02	111	3	000000	6

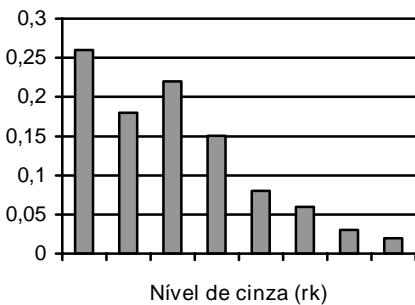


Figura 1 - Representação gráfica do histograma da imagem.

6.1.2 Redundância Interpixel

Existem situações em que uma imagem apresenta pixels fortemente correlacionados, sendo estas correlações decorrentes de uma relação estrutural ou geométrica entre os objetos que a compõem. Pelo fato de o valor de qualquer pixel de uma imagem poder ser razoavelmente predito a partir dos valores de seus vizinhos, a informação contida por pixels individuais é relativamente pequena. A maior parte da contribuição visual de um simples pixel para uma imagem é redundante; ela poderia ter sido predita com base dos valores de seus vizinhos. As expressões 'redundância espacial', 'redundância geométrica', e 'redundância entre quadros (*interframe*)' são utilizadas para indicar estas dependências. Nós as denominaremos pelo termo 'redundância interpixel', que engloba todos os casos particulares.

A fim de reduzir as redundâncias interpixel em uma imagem, o arranjo bidimensional de pixels, normalmente utilizado para a visualização e interpretação, deve ser transformado em um formato mais eficiente (mas, geralmente, 'não visualizável'), por exemplo, utilizando as diferenças entre os pixels adjacentes para representar uma imagem. Transformações capazes de remover a redundância interpixel são conhecidas como mapeamentos. Estes mapeamentos são ditos reversíveis se os elementos da imagem original puderem ser reconstruídos a partir do conjunto de dados transformados.

6.1.3 Redundância Psicovisual

Existem inúmeras experiências capazes de comprovar o fato de que o olho humano não responde com igual sensibilidade a toda informação visual que recebe. Certas informações possuem menor importância relativa do que outras no processo visual normal. Estas informações menos importantes podem ser consideradas redundantes do ponto de vista psicovisual, e, portanto, podem ser eliminadas sem prejudicar significativamente a qualidade da imagem percebida pelo sistema visual humano.

A redundância psicovisual é fundamentalmente diferente das redundâncias anteriormente discutidas. Ao contrário das redundâncias de codificação e interpixel, a redundância psicovisual é associada a informações visuais quantificáveis ou reais. Sua eliminação é possível apenas pelo fato de a informação propriamente dita não ser essencial para o processamento visual normal.

Considerando que a eliminação de dados psicovisualmente redundantes resulta em uma perda de informação quantitativa, a mesma é comumente chamada de quantização. Esta terminologia é consistente com o uso normal da palavra, a qual geralmente significa o mapeamento de uma ampla faixa de valores de entrada para um número limitado de valores de saída. Como esta é uma operação irreversível (a informação visual é perdida), a quantização resulta em uma compressão de dados com perdas.

6.1.4 Critérios de Fidelidade

A necessidade de obtenção de maiores taxas de compressão aliada à exploração adequada de limitações e peculiaridades do sistema visual humano permite a elaboração de técnicas de compressão de imagens nas quais ocorre uma perda de informação visual quantitativa ou real. Considerando que informações de interesse podem ser perdidas, torna-se desejável quantificar a natureza e a extensão da perda de informação. Dois grupos gerais de critérios são utilizados como base para tal análise: (1) critérios de fidelidade objetiva e (2) critérios de fidelidade subjetiva.

Quando o nível de perda de informação puder ser expresso como uma função da imagem original, ou imagem de entrada, e da imagem de saída, comprimida e descomprimida subsequente, diz-se que este baseia-se em um critério de fidelidade objetiva. Um bom exemplo é o erro médio quadrático (rms) entre a imagem original e a processada. Seja $f(x,y)$ a imagem de entrada e seja $f'(x,y)$ a estimativa ou aproximação de $f(x,y)$ resultante de sua compressão e subsequente descompressão. Para qualquer valor de x e y , o erro $e(x,y)$ entre $f(x,y)$ e $f'(x,y)$ pode ser definido como

$$e(x,y) = f'(x,y) - f(x,y) \quad (6.5)$$

de forma que o erro total entre as duas imagens é

$$\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f'(x,y) - f(x,y)]$$

sendo as imagens de tamanho $M \times N$. O erro médio quadrático, e_{rms} , entre $f(x,y)$ e $f'(x,y)$ pode ser obtido por:

$$e_{rms} = \left[\frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f'(x,y) - f(x,y)]^2 \right]^{1/2} \quad (6.6)$$

Outro critério de fidelidade objetiva possível é a relação sinal-ruído rms (SNR_{rms}) entre a imagem comprimida e a descomprimida, dada por

$$SNR_{rms} = \sqrt{\frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f'(x,y)^2}{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f'(x,y) - f(x,y)]^2}} \quad (6.7)$$

Embora os critérios de fidelidade objetiva ofereçam mecanismos simples e convenientes para se avaliar a perda de informação decorrente da compressão, na maioria das vezes as imagens comprimidas e descomprimidas são observadas por seres humanos ao final do processo. O sistema visual humano possui características peculiares, a ponto de duas imagens contendo a mesma quantidade de erro rms poderem ser avaliadas como se possuíssem qualidade visual completamente distinta. Uma destas características é a maior sensibilidade a erros em áreas escuras da imagem e nas regiões de bordas. Conseqüentemente, a medição da qualidade de uma imagem por meio de avaliações subjetivas de um observador humano é freqüentemente mais apropriada.

Para a avaliação subjetiva da qualidade de uma imagem podem ser usados critérios absolutos (como o proposto pela *Television Allocations Study Organization*, que classifica uma

imagem em: excelente, ótima, aceitável, marginal, inferior e imprestável) ou relativos, usando por exemplo comparações sucessivas entre pares de imagens.

6.2 Modelos de compressão de imagem

Nesta seção, examinaremos as características principais de um sistema de codificação e compressão de imagens e desenvolveremos um modelo geral para representá-lo.

Um sistema de codificação / compressão genérico (figura 2) consiste de dois blocos estruturais distintos: um codificador e um decodificador. O codificador parte de uma imagem de entrada $f(x,y)$, a partir da qual cria um conjunto de símbolos. Após a transmissão através do canal, o sinal codificado é aplicado ao bloco decodificador, onde uma imagem de saída reconstruída $f'(x,y)$ é produzida. A imagem recebida, $f'(x,y)$, poderá ou não ser uma réplica exata de $f(x,y)$. Em caso positivo, o sistema é dito imune a erros, ou seja, capaz de preservar a informação; em caso negativo, haverá um certo nível de distorção presente na imagem reconstruída.

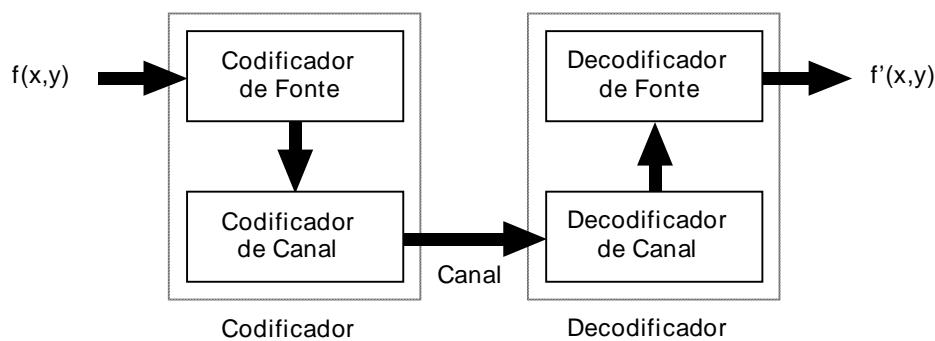


Figura 2 - Um sistema genérico de codificação e compressão de dados.

Tanto o codificador como o decodificador apresentados na figura 2, consistem de dois sub-blocos relativamente independentes. O codificador é composto por um codificador de fonte, o qual remove as redundâncias recebidas, e um codificador de canal, o qual aumenta a imunidade a ruídos do sinal produzido à saída do codificador de fonte. Analogamente, a etapa decodificadora inclui um decodificador de canal seguido por um decodificador de fonte. Se o canal entre o codificador e o decodificador for imune a ruídos, as etapas codificadora e decodificadora de canal serão omitidas.

6.2.1 O codificador e decodificador de fonte

O codificador de fonte é responsável pela redução ou eliminação de qualquer redundância (de codificação, interpixel, ou psicovisual) presente na imagem de entrada. Sua função normalmente pode ser modelada por uma série de três operações independentes. Como mostra a figura 3(a), cada operação está projetada para reduzir uma das três redundâncias descritas na Seção 6.1. A figura 3(b) representa o decodificador de fonte correspondente.

No primeiro estágio do processo de codificação de fonte, o mapeador transforma os dados de entrada em um formato (geralmente não visível) projetado para reduzir as redundâncias interpíxel da imagem de entrada. Esta operação geralmente é reversível e pode, ou não, reduzir diretamente a quantidade de dados exigidos para representar a imagem. A codificação por comprimento de cadeia (*Run-length encoding*) é um exemplo de um mapeamento que resulta diretamente na compressão de dados. A representação de uma imagem por um conjunto de coeficientes de transformadas matemáticas, por outro lado, é um exemplo de caso em que o mapeador transforma a imagem em uma série de coeficientes, cujas redundâncias são mais acessíveis aos estágios posteriores do processo de codificação.

O segundo estágio, ou bloco quantizador na figura 3(a), reduz a precisão de saída do mapeador de acordo com alguns critérios de fidelidade preestabelecidos. Este estágio reduz as redundâncias psicovisuais da imagem de entrada e as alterações que promove no sinal são

irreversíveis. Portanto, este bloco deve ser omitido quando se desejar a compressão livre de erros.

No terceiro e último estágio do processo de codificação de fonte, aparece o codificador de símbolos, responsável por produzir uma palavra-código de comprimento fixo ou – na maioria dos casos – variável para representar cada saída do quantizador. Ao final desta etapa, a imagem de entrada não deve apresentar qualquer tipo de redundância. Convém ressaltar, finalmente, que nem todos os blocos indicados na figura 3(a) devem obrigatoriamente estar presentes num codificador de fonte, bem como alertar para o fato de que em muitos sistemas de compressão um mesmo algoritmo executa as funções correspondentes a mais de um bloco.

O decodificador de fonte apresentado na figura 3(b) contém apenas dois componentes: um decodificador de símbolos e um mapeador inverso. Estes blocos realizam, em seqüência oposta, as operações inversas dos blocos mapeadores e codificadores de símbolo do codificador de fonte. Como a quantização resulta em perda de informação irreversível, um bloco quantizador inverso não aparece no modelo genérico do decodificador de fonte apresentado na figura 3(b).

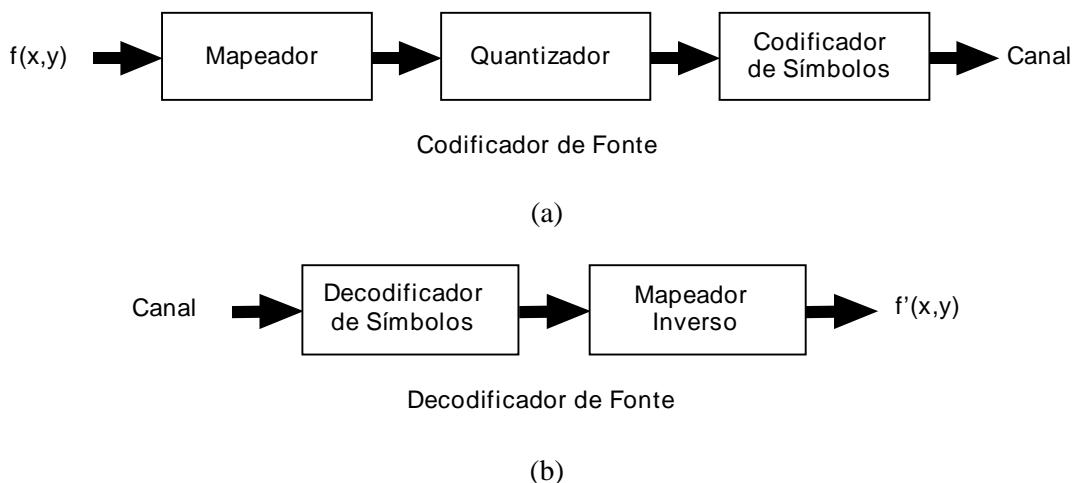


Figura 3 - Diagramas em blocos: (a) codificador de fonte; (b) decodificador de fonte.

6.2.2 O codificador e decodificador de canal

O codificador e o decodificador de canal exercem uma importante função no processo global de codificação e decodificação nos casos em que o canal da figura 3 estiver contaminado por ruído ou sujeito a erro. Eles são projetados para reduzir o impacto do ruído do canal através da inserção de uma forma controlada de redundância nos dados provenientes do codificador de fonte.

Uma das técnicas mais úteis e conhecidas de codificação de canal foi concebida por R.W. Hamming [Hamming 1950]. Esta técnica consiste em se adicionar bits suficientes aos dados que estiverem sendo codificados, a fim de assegurar que um número mínimo de bits deve variar entre as palavras-código válidas. Hamming mostrou, por exemplo, que se 3 bits de redundância forem adicionados a uma palavra de 4 bits, de forma que a distância¹ entre duas palavras-código válidas seja 3, todos os erros que atinjam somente um bit podem ser detectados e corrigidos. (Através da atribuição de bits de redundância adicionais, erros em múltiplos bits podem ser detectados e/ou corrigidos). O código Hamming (7,4) possui palavras-código de 7 bits de $h_1 h_2 \dots h_6 h_7$ associadas aos números binários de 4 bits $b_3 b_2 b_1 b_0$ através das expressões lógicas:

¹ A 'distância' entre duas palavras-código é definida como o número de bits que devem ser modificados em uma palavra-código, de modo a resultar na outra. Por exemplo, a distância entre as palavras-código 01010101 e 11011100 é 3. A distância mínima de um código é a menor distância entre duas de suas palavras-código.

$$\begin{aligned}
 h_1 &= b_3 \oplus b_2 \oplus b_0 & h_3 &= b_3 \\
 h_2 &= b_3 \oplus b_1 \oplus b_0 & h_5 &= b_2 \\
 h_4 &= b_2 \oplus b_1 \oplus b_0 & h_6 &= b_1 \\
 && h_7 &= b_0
 \end{aligned} \tag{6.8}$$

onde \oplus denota a operação XOR (ou-exclusivo). Observe que os bits h_1 , h_2 e h_4 são bits de paridade par para os conjuntos de bits $b_3 b_2 b_0$, $b_3 b_1 b_0$ e $b_2 b_1 b_0$, respectivamente.

Para se decodificar uma palavra-código codificada por Hamming, o decodificador de canal deve verificar a paridade da palavra-código recebida, para saber se ela é coerente com a convenção de paridade previamente estabelecida. Se tiver havido erro em um único bit, a palavra de paridade $c_4 c_2 c_1$ será diferente de zero. Os valores dos bits individuais desta palavra são obtidos pelas seguintes expressões lógicas:

$$\begin{aligned}
 c_1 &= h_1 \oplus h_3 \oplus h_5 \oplus h_7 \\
 c_2 &= h_2 \oplus h_3 \oplus h_6 \oplus h_7 \\
 c_4 &= h_4 \oplus h_5 \oplus h_6 \oplus h_7
 \end{aligned} \tag{6.9}$$

Se um valor diferente de zero for encontrado, o decodificador simplesmente complementa a posição da palavra-código indicada pela palavra de paridade. O valor binário decodificado é, em seguida, extraído da palavra-código corrigida como $h_3 h_5 h_6 h_7$.

Exercício resolvido

Seja uma imagem de 256 tons de cinza quantizada pelo método IGS (*Improved Gray-Scale*) [Bisignani 1966] de modo a resultar em uma imagem de 16 tons de cinza, na qual cada pixel é codificado usando uma palavra-código de 4 bits, dentre as palavras-código mostradas na segunda coluna da tabela 2. Pede-se:

- Projetar um código Hamming (7,4) capaz de proteger as informações codificadas contra erros em um de seus bits.
- Calcular a razão de compressão obtida, levando-se em conta somente a codificação de fonte.
- Calcular a razão de compressão obtida, considerando o *overhead* introduzido pela codificação de canal.

Tabela 2 - Palavras-código para uma imagem quantizada pelo método IGS.

Pixel	Código IGS	Código de Hamming (7,4)
i	0110	1100110
i + 1	1001	0011001
i + 2	1000	1110000
i + 3	1111	1111111

Solução

- Utilizando as relações lógicas (6.8) para a primeira palavra-código, obtemos:

$$\begin{aligned}
 h_1 &= b_3 \oplus b_2 \oplus b_0 = 0 \oplus 1 \oplus 0 = 1 \\
 h_2 &= b_3 \oplus b_1 \oplus b_0 = 0 \oplus 1 \oplus 0 = 1 \\
 h_4 &= b_2 \oplus b_1 \oplus b_0 = 1 \oplus 1 \oplus 0 = 0 \\
 h_3 &= b_3 = 0 \\
 h_5 &= b_2 = 1
 \end{aligned}$$

$$h_6 = b_1 = 1$$

$$h_7 = b_0 = 0.$$

Logo, a palavra-código de Hamming equivalente é: 1100110₂.

Procedendo de forma semelhante para as demais palavras-código, obteremos as palavras-código indicadas na terceira coluna da tabela 2.

b) A razão de compressão obtida após a quantização (codificação de fonte) é obtida dividindo o número de bits originalmente necessário para representar cada pixel da imagem (8 bits) pelo comprimento da palavra-código IGS (4 bits). Logo, $C_R = 2$.

c) Devido à utilização do código de Hamming, cada pixel utilizará 7 bits para ser representado. Portanto, a razão de compressão após a codificação de canal será: $C_R = 8/7 = 1,14$.

6.3 Elementos de Teoria da Informação

Uma vez que o objetivo das técnicas de compressão de imagens é reduzir tanto quanto possível a quantidade de dados utilizados para representar uma imagem, surge naturalmente a questão: Quantos dados são realmente necessários para representar a imagem? Em outras palavras, existe uma quantidade mínima de dados considerada suficiente para descrever completamente a imagem sem perda de informação? A teoria da informação fornece o embasamento matemático para a resposta desta e de outras perguntas relacionadas ao tema.

6.3.1 Medidas de informação

A premissa fundamental da teoria da informação é que a geração de informação pode ser modelada como um processo probabilístico, no qual um evento aleatório E , que ocorre com probabilidade $P(E)$ contém

$$I(E) = \log \frac{1}{P(E)} = -\log P(E) \quad (6.10)$$

unidades de informação. A quantidade $I(E)$ é freqüentemente denominada 'informação própria' de E . Em linhas gerais, a quantidade de informação própria atribuída ao evento E é inversamente proporcional à probalidade de ocorrência de E . Se $P(E) = 1$ (isto é, o evento ocorre sempre), $I(E) = 0$ e nenhuma informação é atribuída a ele. Isto significa dizer que, pelo fato de não existir nenhuma incerteza associada ao evento, nenhuma informação precisaria ser transferida comunicando-se que o evento ocorreu. Entretanto, se $P(E) = 0,99$, comunicar que E ocorreu transmitiria uma pequena quantidade de informação. Por outro lado, a comunicação de que E não ocorreu conteria uma quantidade maior de informação, pois este resultado é menos provável.

A base do logaritmo na equação (6.10) determina a unidade utilizada para medir a informação. Se utilizarmos um logaritmo de base r , a medida conterá r unidades. Se a base 2 for selecionada, a unidade resultante de informação é chamada bit. Observe que se $P(E) = \frac{1}{2}$, $I(E) = -\log_2 \frac{1}{2}$, ou 1 bit. Ou seja, 1 bit corresponde à quantidade de informação transmitida quando um dos dois eventos, igualmente possíveis, ocorre. Um exemplo clássico de tal situação é o ato de atirar uma moeda honesta e comunicar o resultado (cara ou coroa).

6.3.2 O canal de informação

Quando a informação própria é transferida entre uma fonte de informação e um usuário daquela informação, diz-se que a fonte de informação está conectada ao usuário de informação por um canal de informação. O canal de informação é o meio físico que conecta a fonte ao usuário. Pode ser uma linha telefônica, um meio de propagação de ondas eletromagnéticas, ou um cabo

entre dois computadores. A figura 4 mostra um diagrama em blocos simples para um sistema de informação discreto. Aqui, o parâmetro de particular interesse é a capacidade do sistema, definida como sendo sua habilidade em transferir informação.

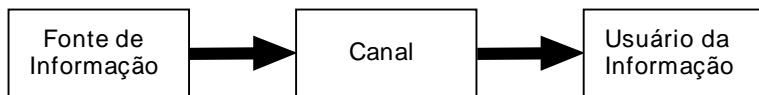


Figura 4 - Diagramas em blocos de um sistema de informação discreto.

Suponhamos que a fonte de informação da figura 4 gere uma seqüência aleatória de símbolos a partir de um conjunto de símbolos possíveis. Em outras palavras, a saída da fonte é uma variável aleatória discreta. O conjunto de símbolos de fonte $\{a_1, a_2, \dots, a_J\}$ é chamado de alfabeto da fonte A, e os elementos do conjunto, denotados por a_j , são chamados de símbolos ou letras. A probabilidade de a fonte vir a produzir o símbolo a_j é $P(a_j)$, e

$$\sum_{j=1}^J P(a_j) = 1 \quad (6.11)$$

O vetor $z = [P(a_1), P(a_2), \dots, P(a_J)]^T$, de dimensões $J \times 1$, representa o conjunto de todas as probabilidades dos símbolos da fonte $\{P(a_1), P(a_2), \dots, P(a_J)\}$. O conjunto finito (A, z) descreve completamente a fonte de informação.

A probabilidade de a fonte discreta emitir o símbolo a_j é $P(a_j)$, de forma que a informação própria gerada pela produção de um único símbolo de fonte é, de acordo com a equação (6.10), $I(a_j) = -\log P(a_j)$. Se k símbolos forem gerados, a lei dos grandes números estipula que, para um valor consideravelmente alto de k , o símbolo a_j sairá (em média) $kP(a_j)$ vezes. Logo, a informação própria média obtida das k saídas é

$$-k(Pa_1)\log P(a_1) - kP(a_2)\log P(a_2) - \dots - kP(a_J)\log P(a_J)$$

ou

$$-k \sum_{j=1}^J P(a_j) \log P(a_j).$$

A informação média por saída de fonte, denotada por $H(z)$, é

$$H(z) = -\sum_{j=1}^J P(a_j) \log P(a_j) \quad (6.12)$$

e é chamada de incerteza ou entropia de fonte. Ela define a quantidade média de informação (em unidades r por símbolo) obtida através da observação de uma simples saída de fonte. À medida que sua magnitude aumenta, mais incerteza e, consequentemente mais informação é associada à fonte. Se os símbolos de fonte forem equiprováveis, a entropia da equação (6.12) será maximizada e a fonte fornecerá a maior média possível de informações por símbolo.

Após termos modelado a fonte de informação, podemos agora desenvolver a função de transferência do canal de informação com razoável facilidade. Como modelamos a entrada para o canal na figura 4 como uma variável aleatória discreta, a informação transferida para a saída do canal será, também, uma variável aleatória discreta. Assim como a variável aleatória de fonte, ela assume valores pertencentes ao conjunto de símbolos $\{b_1, b_2, \dots, b_K\}$, chamado de

alfabeto do canal B. A probabilidade do símbolo b_k ser apresentado para o usuário de informação é $P(b_k)$. O conjunto finito (B, v) , onde $v = [P(b_1), P(b_2), \dots, P(b_K)]^T$, descreve completamente a saída do canal e, por conseguinte, a informação recebida pelo usuário.

A probabilidade de uma determinada saída de canal $P(b_k)$ e a distribuição de probabilidade da fonte z estão relacionadas pela expressão

$$P(b_k) = \sum_{j=1}^J P(b_k | a_j) P(a_j) \quad (6.13)$$

onde $P(b_k | a_j)$ é a probabilidade condicional de que a saída b_k será recebida, considerando-se que o símbolo de fonte a_j foi gerado. Se as probabilidades condicionais mencionadas na equação (6.13) forem dispostas em uma matriz Q de dimensões K x J, de forma que

$$Q = \begin{bmatrix} P(b_1 | a_1) & P(b_1 | a_2) & \dots & P(b_1 | a_J) \\ P(b_2 | a_1) & \vdots & \dots & \vdots \\ \ddots & \ddots & \dots & \ddots \\ P(b_K | a_1) & P(b_K | a_2) & \dots & P(b_K | a_J) \end{bmatrix} \quad (6.14)$$

a distribuição de probabilidade do alfabeto de saída completo pode ser computada a partir da expressão

$$v = Qz. \quad (6.15)$$

A matriz Q, com elementos $q_{kj} = P(b_k | a_j)$, é conhecida como matriz de transição direta do canal ou simplesmente matriz do canal.

Para que se possa determinar a capacidade de um canal de informação com uma matriz de transição direta de canal Q, a entropia da fonte de informação deve ser primeiramente calculada considerando-se que o usuário da informação observa uma saída particular b_k . A equação (6.13) define uma distribuição de símbolos de fonte para qualquer b_k observado, de forma que cada b_k possua uma função de entropia condicional. Com bases nos passos que levam à equação (6.12), esta função de entropia condicional, denotada por $H(z|b_k)$, pode ser escrita como

$$H(z|b_k) = -\sum_{j=1}^J P(a_j | b_k) \log P(a_j | b_k) \quad (6.16)$$

onde $P(a_j | b_k)$ é a probabilidade do símbolo a_j ter sido transmitido pela fonte, levando-se em conta que o usuário tenha recebido b_k . O valor esperado (ou valor médio) desta expressão sobre todos os b_k é

$$H(z|v) = \sum_{k=1}^K H(z|b_k) P(b_k) \quad (6.17)$$

a qual, após a substituição de $H(z|b_k)$ pela expressão à direita na equação (6.16) e alguns pequenos rearranjos, pode ser escrita como

$$H(z|v) = - \sum_{j=1}^J \sum_{k=1}^K P(a_j, b_k) \log P(a_j | b_k) \quad (6.18)$$

onde $P(a_j, b_k)$ é a probabilidade conjunta de a_j e b_k . Ou seja, $P(a_j, b_k)$ é a probabilidade de a_j ser transmitido e de b_k ser recebido.

O termo $H(z|v)$ é chamado de ambigüidade de z para com v . Representa a informação média de um símbolo de fonte, considerando-se a observação do símbolo de saída que resultou de sua geração. Pelo fato de $H(z)$ ser a informação média de um símbolo de fonte, considerando nenhum conhecimento do símbolo de saída resultante, a diferença entre $H(z)$ e $H(z|v)$ é a informação média recebida durante a observação de um único símbolo de saída. Esta diferença, denotada por $I(z, v)$ e conhecida como informação mútua de z e v , é

$$I(z, v) = H(z) - H(z|v). \quad (6.19)$$

Substituindo-se os valores de $H(z)$ e $H(z|v)$ dados pelas equações (6.12) e (6.18), e lembrando-se que $P(a_j) = P(a_j, b_1) + P(a_j, b_2) + \dots + P(a_j, b_K)$ obtemos

$$I(z, v) = \sum_{j=1}^J \sum_{k=1}^K P(a_j, b_k) \log \frac{P(a_j, b_k)}{P(a_j)P(b_k)} \quad (6.20)$$

a qual, após posterior manipulação, pode ser escrita como

$$I(z, v) = \sum_{j=1}^J \sum_{k=1}^K P(a_j) q_{kj} \log \frac{q_{kj}}{\sum_{i=1}^J P(a_i) q_{ki}}. \quad (6.21)$$

Assim, a informação recebida durante a observação de uma única saída do canal de informação é uma função da distribuição de probabilidade dos símbolos de entrada (ou símbolos de fonte) z e da matriz de canal Q . O valor mínimo possível de $I(z, v)$ é zero e ocorre quando os símbolos de entrada ou saída são estatisticamente independentes. Neste caso, $P(a_j, b_k) = P(a_j)P(b_k)$ e o termo logarítmico na equação (6.20) é 0 para todo j e k . O valor máximo de $I(z, v)$ dentre todas as escolhas possíveis de distribuição de fonte z é a capacidade C do canal descrito pela matriz de canal Q . Ou seja,

$$C = \max_z [I(z, v)] \quad (6.22)$$

onde o máximo é obtido sobre todas as distribuições de entrada possíveis. A capacidade do canal define a taxa máxima (em unidades de informação r por símbolo de fonte) pela qual a informação pode ser transmitida seguramente através do canal. Além disso, a capacidade de um canal não depende das probabilidades de entrada da fonte (ou seja, como o canal é utilizado), mas é uma função das probabilidades condicionais do canal.

6.3.3 Utilizando a Teoria da Informação

Conforme antecipamos no início desta seção, a teoria da informação fornece as ferramentas necessárias para representar e manipular informações de forma direta e quantitativa. O exemplo a seguir mostra a utilização destas informações para o propósito de compressão de imagens.

Exemplo

Seja a imagem 4 x 8 de 256 tons de cinza a seguir:

22	22	22	95	167	234	234	234
22	22	22	95	167	234	234	234
22	22	22	95	167	234	234	234
22	22	22	95	167	234	234	234

Suponhamos que se deseja estimar o conteúdo de informação (ou entropia) da imagem acima. Para tanto, três alternativas podem ser adotadas.

Como primeiro caso, assumimos que a imagem foi produzida por uma fonte de informação imaginária, capaz de emitir seqüencialmente pixels (i.e. seus valores de tons de cinza, codificados em 8 bits) estatisticamente independentes, de acordo com uma lei de probabilidade pré-definida. Nesta situação, os símbolos da fonte são os níveis de cinza e o alfabeto é composto por 256 símbolos possíveis. Se a distribuição de probabilidade dos símbolos for conhecida (e.g. gaussiana), a entropia de cada pixel na imagem poderá ser calculada pela eq. (6.12). No caso de uma distribuição uniforme, por exemplo, os símbolos da fonte são equiprováveis e a entropia da fonte é de 8 bits/pixel. Em outras palavras, a informação média por símbolo de fonte (pixel) é 8 bits. Portanto, a entropia total da imagem acima é de $4 \times 8 \times 4 \times 8 = 256$ bits. Esta imagem em particular é apenas uma das $2^{8 \times 4 \times 8}$, ou $2^{256} (\sim 10^{77})$ imagens 4 x 8 equiprováveis que podem ser produzidas pela fonte.

Um segundo método, conhecido como estimativa de primeira ordem, consiste na construção de um modelo baseado na freqüência relativa de ocorrência de cada símbolo na imagem sob consideração. Ou seja, consideraríamos a imagem analisada como uma amostra do comportamento da fonte que a gerou. Levantando as probabilidades de cada nível de cinza na imagem proposta, teríamos:

Nível de cinza	Nº de ocorrências	Probabilidade
22	12	3/8
95	4	1/8
167	4	1/8
234	12	3/8

Entrando com as probabilidades obtidas na eq. (6.12), obtemos uma entropia de 1,81 bits/pixel, o que representa uma entropia total da fonte de aproximadamente 58 bits.

Uma terceira forma de calcular a entropia da fonte seria examinar a freqüência relativa de blocos de 2 pixels na imagem. Assumindo que a imagem em questão é conectada de linha a linha e do final ao início, as freqüências relativas de ocorrência dos pares de pixels podem ser computadas, conforme a tabela a seguir:

Par de tons de cinza	Nº de ocorrências	Probabilidade
(22, 22)	8	1/4
(22, 95)	4	1/8
(95, 167)	4	1/8
(167, 234)	4	1/8
(234, 234)	8	1/4

A estimativa de entropia resultante do uso da eq. (6.12) será $2,5/2 = 1,25$ bits/pixel, onde a divisão por 2 é uma consequência de estarmos considerando dois pixels de cada vez. Esta estimativa é denominada estimativa de segunda ordem da entropia da fonte. Usando raciocínio semelhante, poderíamos computar a entropia de terceira, quarta, ..., n-ésima ordem da fonte, o que se tornaria computacionalmente lento.

Os valores obtidos no exemplo anterior fornecem interessantes interpretações. A estimativa de primeira ordem da entropia pode ser entendida como o limite mínimo de representação de um pixel usando um certo número de bits (i.e. a taxa máxima de compressão) que se pode atingir explorando apenas a redundância de codificação. Já os resultados mais baixos obtidos com a estimativa de segunda ordem sugerem que taxas ainda maiores podem ser obtidas, se explorarmos a redundância interpíxel. No exemplo analisado, a quantidade de bits necessária para representar um pixel poderia cair de 1,81 bits/pixel para 1,25 bits/pixel se a redundância interpíxel e de codificação fossem, ambas, exploradas.

6.4 Compressão sem perdas

Apresentaremos a seguir algumas das principais técnicas de compressão de dados sem perdas. Em alguns textos técnicos a expressão 'compactação' é também utilizada para estes casos, reservando-se a expressão 'compressão' especificamente para as técnicas que introduzem perdas.

No caso específico de compressão de imagens, convém notar que, em diversas circunstâncias práticas, a compressão deve obrigatoriamente ser sem perdas. Tal é o caso quando se aplicam técnicas de compressão a imagens médicas ou de documentos para fins de arquivamento, onde eventuais perdas são indesejáveis e, muitas vezes, legalmente proibidas.

As técnicas a seguir relatadas são aplicáveis a imagens monocromáticas com dois ou mais tons de cinza e costumam permitir a obtenção de taxas de compressão na faixa de 2 a 10. Elas geralmente consistem de duas etapas principais: (1) elaboração de um método alternativo de representação da imagem, a fim de reduzir as redundâncias interpíxel; e (2) codificação do resultado desta nova representação. Estes passos correspondem aos blocos 'mapeador' e 'codificador de símbolos' da figura 3.

6.4.1 Códigos de palavra-código de comprimento variável

A maneira mais simples de se obter uma compressão de imagens sem perdas é trabalhar na redução apenas da redundância de codificação. Para tanto, pode-se codificar os valores de tons de cinza utilizando códigos de comprimento variável, que atribuem palavras-código mais curtas aos símbolos mais prováveis. Na prática, os símbolos de fonte a serem codificados podem ser os valores de tons de cinza da imagem ou a saída de uma operação de mapeamento (e.g. diferenças entre pixels consecutivos, *run-lengths* etc.)

Código de Huffman

A técnica mais popular de codificação para remoção de redundância é o código de Huffman [Huffman 1952]. Quando aplicado à codificação de cada símbolo da fonte, individualmente, o código de Huffman fornece o menor número inteiro possível de unidades de informação (bits) por símbolo de fonte.

O primeiro passo no algoritmo de Huffman consiste na criação de uma série de reduções na fonte original, através da ordenação das probabilidades de ocorrência dos símbolos sob consideração, combinando os (dois) símbolos de menor probabilidade em um único símbolo que irá substituí-los na próxima etapa de redução da fonte. A figura 5 ilustra este processo para o caso de codificação binária. À esquerda, aparecem os símbolos originais da fonte hipotética de informação, ordenados em ordem decrescente de probabilidade de ocorrência. Na primeira redução, os dois símbolos de menor probabilidade (a_3 com prob. = 0,06 e a_5 com prob. = 0,04) são combinados, formando um 'símbolo composto' cuja probabilidade é $0,06 + 0,04 = 0,1$. Este

'símbolo composto' e sua respectiva probabilidade são posicionados na coluna correspondente à primeira redução de fonte de forma que todos os valores da coluna estejam em ordem decrescente. O processo é então repetido até atingirmos uma fonte reduzida com apenas dois símbolos.

Fonte Original		Reduções de fonte			
Símbolo	Probabilidade	1	2	3	4
a ₂	0,4	0,4	0,4	0,4	0,6
a ₆	0,3	0,3	0,3	0,3	0,4
a ₁	0,1	0,1	0,2	0,3	
a ₄	0,1	0,1	0,1		
a ₃	0,06	0,1			
a ₅	0,04				

Figura 5 - Reduções de fonte no algoritmo de Huffman.

O segundo passo no algoritmo de Huffman consiste em codificar cada fonte reduzida, iniciando pela menor fonte e caminhando em direção à fonte original. O menor código binário possível para uma fonte de 2 símbolos é, obviamente, formado pelos símbolos 0 e 1. Como a figura 6 ilustra, estes valores são atribuídos aos dois símbolos da direita (neste caso, segundo a convenção 'probabilidade maior recebe bit 0'). Como o símbolo de probabilidade 0,6 foi gerado a partir da combinação de dois outros símbolos na fonte reduzida à sua esquerda, o 0 usado para codificá-lo é agora atribuído a ambos os símbolos que lhe deram origem, colocando-se um 0 ou 1 à direita de cada um (segundo a mesma convenção) para distingui-los. O processo é repetido para cada fonte reduzida até se retornar à fonte original. O código resultante aparece na terceira coluna da figura 6. O comprimento médio do código é:

$$\begin{aligned} L_{\text{avg}} &= (0,4)(1) + (0,3)(2) + (0,1)(3) + (0,1)(4) + (0,06)(5) + (0,04)(5) \\ &= 2,2 \text{ bits/símbolo.} \end{aligned}$$

A entropia da fonte, calculada pela eq. (6.12), é 2,14 bits/símbolo. A eficiência do código de Huffman, calculada como a razão entre a entropia da fonte e o comprimento médio do código, é de 0,973.

O algoritmo de Huffman permite a criação de um código ótimo para um dado conjunto de símbolos e respectivas probabilidades, com a ressalva de que os símbolos devem ser codificados um de cada vez. O código é denominado 'código de bloco instantâneo e unicamente decodificável', porque cada símbolo da fonte é mapeado em uma seqüência fixa (bloco) de bits, cada palavra-código pode ser decodificada instantaneamente, ou seja, sem fazer referência a símbolos subsequentes e porque não há mais de uma forma de decodificar uma string de 0s e 1s, ou seja, nenhuma palavra-código é prefixo de nenhuma outra.

A principal desvantagem prática do código de Huffman é a necessidade de se armazenar ou transmitir a tabela de símbolos da fonte e respectivas probabilidades juntamente com os dados codificados.

Fonte Original			Reduções de fonte			
Símbolo	Prob.	Código	1	2	3	4
a ₂	0,4	1	0,4 1	0,4 1	0,4 1	0,6 0
a ₆	0,3	00	0,3 00	0,3 00	0,3 00	0,4 1
a ₁	0,1	011	0,1 011	0,2 010	0,3 01	
a ₄	0,1	0100	0,1 0100	0,1 011		
a ₃	0,06	01010	0,1 0101			
a ₅	0,04	01011				

Figura 6 - Atribuição de palavras-código no algoritmo de Huffman.

Leitura complementar

Diversos livros apresentam implementações do código de Huffman em linguagem C, dentre eles [Tenenbaum et al. 1990].

Código de Huffman Truncado²

Quando o número de símbolos a serem codificados é muito grande, a construção do código de Huffman torna-se uma tarefa não trivial do ponto de vista computacional. Além disso, aos símbolos menos prováveis poderão ser atribuídas palavras-código proibitivamente longas. Uma possível modificação sobre o código de Huffman original consiste em se codificar somente os K símbolos mais prováveis, dentre os N símbolos da fonte. Para os demais símbolos, utiliza-se uma palavra-código de prefixo seguida de um código de comprimento fixo adequado. Esta modificação do algoritmo original de codificação por Huffman é denominada 'código de Huffman truncado'.

A tabela 3 ilustra a obtenção do código de Huffman truncado para o caso em que $N = 21$ e $K = 12$. Neste caso, os símbolos a_1 a a_{12} foram codificados por Huffman enquanto os símbolos a_{13} a a_{21} utilizam um prefixo de 2 bits (10) seguido de um código de comprimento fixo e igual a 4 bits. O comprimento médio de uma palavra-código, neste caso, será 4,24 bits/símbolo, valor ligeiramente maior que aquele que seria obtido utilizando Huffman tradicional (4,05 bits/símbolo), mas ainda bastante próximo do limite teórico dado pela entropia da fonte, que é de 4,0 bits/símbolo.

Codificação Aritmética

Na codificação por Huffman existe uma correspondência biunívoca entre as palavras-código e os símbolos (ou seqüências de símbolos) da fonte. A técnica de codificação aritmética, cuja concepção é atribuída a Elias (ver [Abramson 1963]), é uma técnica orientada a bloco, na qual uma palavra-código aritmética é atribuída a uma seqüência de símbolos de entrada. A palavra-código em si define um intervalo de números reais entre 0 e 1. À medida que o número de símbolos na mensagem aumenta, o intervalo usado para representá-la se torna menor e o número de bits utilizados para representá-lo se torna maior.

A figura 7 ilustra o processo básico de codificação aritmética. Neste caso, deseja-se codificar uma seqüência de 5 símbolos, $a_1a_2a_3a_3a_4$, obtida a partir de uma fonte de quatro símbolos. No início do processo, assume-se que a mensagem ocupa todo o intervalo [0,1). Como indica a tabela 4, este intervalo está inicialmente subdividido em 4 regiões, de acordo com a probabilidade de cada símbolo. Como o símbolo a_3 ocorre mais vezes, ele ocupa um intervalo maior que os demais símbolos. Ao codificarmos a mensagem, o primeiro símbolo a ser codificado será a_1 , o que provocará um estreitamento inicial do intervalo da mensagem para o intervalo [0, 0,2). Este intervalo reduzido é então subdividido de acordo com as probabilidades dos símbolos que compõem a fonte e o processo continua com o próximo símbolo a ser codificado, neste caso a_2 . A codificação de a_2 reduz o intervalo a [0,04, 0,08), a codificação de a_3 provoca uma nova redução, desta vez para [0,056, 0,072) e assim por diante. O símbolo final da mensagem, que deve ser reservado como um indicador especial de fim de mensagem (*EOM - End-Of-Message*), reduzirá a faixa ao intervalo [0,06752, 0,0688). Naturalmente, qualquer número real neste intervalo, por exemplo 0,068, poderá ser utilizado para representar a mensagem.

² Diversos autores dão a esta alternativa o nome de Código de Huffman Modificado (MHC).

Reservaremos este nome para a modificação no código de Huffman encontrada na codificação de imagens P&B no padrão G3 dos sistemas fac-símile.

Tabela 3 - Codificação utilizando código de Huffman truncado.

Símbolo da fonte	Probabilidade	Huffman Truncado
a ₁	0,2	11
a ₂	0,1	011
a ₃	0,1	0000
a ₄	0,06	0101
a ₅	0,05	00010
a ₆	0,05	00011
a ₇	0,05	00100
a ₈	0,04	00101
a ₉	0,04	00110
a ₁₀	0,04	00111
a ₁₁	0,04	01000
a ₁₂	0,03	01001
a ₁₃	0,03	10 0000
a ₁₄	0,03	10 0001
a ₁₅	0,03	10 0010
a ₁₆	0,02	10 0011
a ₁₇	0,02	10 0100
a ₁₈	0,02	10 0101
a ₁₉	0,02	10 0110
a ₂₀	0,02	10 0111
a ₂₁	0,01	10 1000

Na mensagem codificada aritmeticamente da figura 7, três dígitos decimais são necessários para representar uma mensagem de cinco símbolos, numa média de 0,6 dígito decimal/símbolo. Pode-se mostrar que este resultado se aproxima da entropia da fonte, que é de 0,58 dígito decimal/símbolo. Quanto maior o comprimento da mensagem a ser codificada, mais o resultado se aproximará do limite teórico dado pelo Teorema de Shannon.

Existem dois problemas práticos principais, que limitam a eficiência da codificação aritmética: (1) a adição de um indicador de fim de mensagem, necessário para separar uma mensagem da seguinte; e (2) a necessidade de se utilizar aritmética de precisão finita. As duas formas mais usuais de se contornar o segundo problema são a estratégia de escala e a estratégia de arredondamento. A primeira renormaliza cada subintervalo para a faixa [0, 1) antes de subdividi-lo de acordo com as probabilidades dos símbolos. A segunda garante que os truncamentos associados à precisão finita não comprometem a precisão de representação dos subintervalos de codificação.

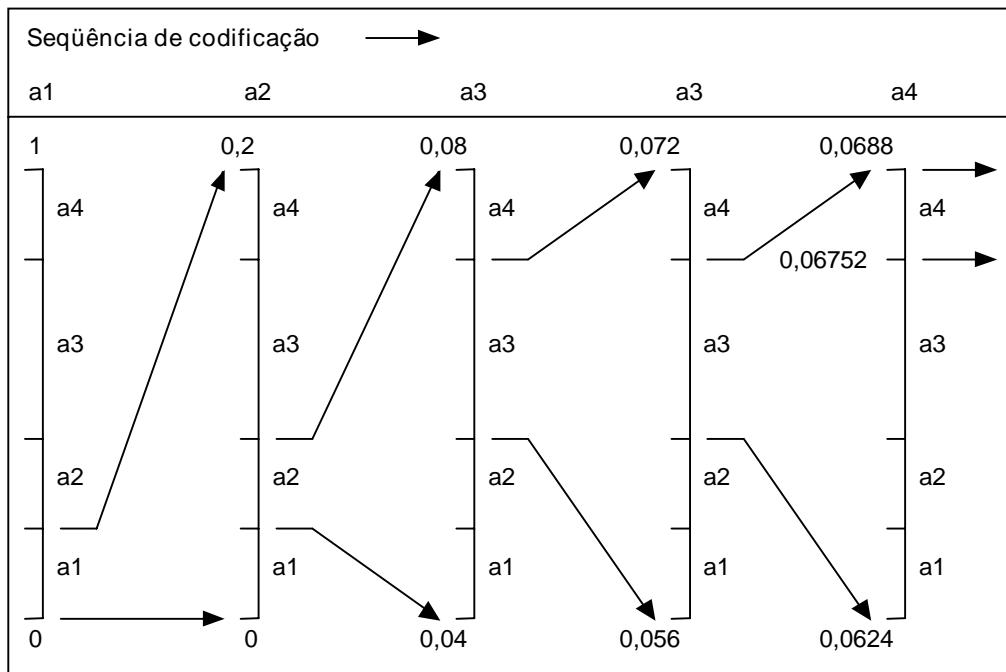


Figura 7 - Codificação aritmética.

Tabela 4 - Exemplo de codificação aritmética.

Símbolo	Probabilidade	Subintervalo inicial
a_1	0,2	[0,0, 0,2)
a_2	0,2	[0,2, 0,4)
a_3	0,4	[0,4, 0,8)
a_4	0,2	[0,8, 1,0)

Exercício resolvido

Dada a mensagem 'MAC ADDICT' codificá-la e decodificá-la utilizando codificação aritmética. Por simplicidade, desprezar o indicador de fim de mensagem (EOM).

Solução:

O primeiro passo é levantar as probabilidades de cada caractere presente na mensagem, resultando nas duas primeiras colunas da tabela a seguir. Na terceira coluna, indicamos o intervalo de valores para cada caractere.

Caractere	Probabilidade	Intervalo
Espaço	0,1	[0,0 , 0,1)
A	0,2	[0,1 , 0,3)
C	0,2	[0,3 , 0,5)
D	0,2	[0,5 , 0,7)
I	0,1	[0,7 , 0,8)
M	0,1	[0,8 , 0,9)
T	0,1	[0,9 , 1,0)

O processo de codificação consiste em adicionar um dígito a cada símbolo correspondente na mensagem original. Iniciando pelo primeiro símbolo – 'M' –, sabemos que o número resultante ficará na faixa entre 0,8 e 0,9. Prosseguindo com o símbolo 'A', reduziremos esta faixa ao

intervalo entre 0,81 e 0,83. As sucessivas reduções de intervalo estão resumidas na tabela a seguir. O resultado final será o número 0,8160903936.

Caractere lido	Limite inferior	Limite superior
M	0,8	0,9
A	0,81	0,83
C	0,816	0,820
Espaço	0,8160	0,8164
A	0,81604	0,81612
D	0,816080	0,816096
D	0,8160880	0,8160912
I	0,81609024	0,81609056
C	0,816090336	0,816090400
T	0,8160903936	0,8160904000

O processo de decodificação está resumido na tabela a seguir, que mostra as várias reduções do número inicial até resultar em 0,0 (fim da decodificação).

Número	Símbolo decodificado	Limite inferior	Limite superior	Largura do intervalo
0,8160903936	M	0,8	0,9	0,1
0,160903936	A	0,1	0,3	0,2
0,30451968	C	0,3	0,5	0,2
0,0225984	Espaço	0,0	0,1	0,1
0,225984	A	0,1	0,3	0,2
0,62992	D	0,5	0,7	0,2
0,6496	D	0,5	0,7	0,2
0,748	I	0,7	0,8	0,1
0,48	C	0,3	0,5	0,2
0,9	T	0,9	1,0	0,1
0,0				

Leitura complementar

O capítulo 5 de [Nelson e Gaily 1996] é inteiramente dedicado à codificação aritmética e contém exemplo de programa em C para implementá-la.

Codificação LZW (Lempel-Ziv-Welch)

Todas as técnicas de codificação para redução de redundância vistas até aqui pressupõem a necessidade de se levantar as probabilidades dos símbolos da fonte. Apresentaremos a seguir o código LZW, considerado por muitos autores o 'algoritmo universal de codificação', por não requerer um conhecimento *a priori* das estatísticas da fonte.

O método de codificação LZW baseia-se na construção progressiva de uma tabela (dicionário) contendo as strings de símbolos de vários comprimentos encontradas na seqüência de símbolos da fonte. Como este dicionário é criado a partir da seqüência a ser codificada, ele reflete com precisão as estatísticas daquela fonte de informação sendo codificada.

Na codificação, a tabela (dicionário) de strings é inicializada com todos os símbolos individuais. A seqüência de entrada é então examinada, símbolo a símbolo, e a mais longa string para a qual existe uma entrada correspondente na tabela é extraída, e a palavra-código para esta string é então transmitida. A string extraída é estendida de mais um símbolo da seqüência de

entrada, formando uma nova string, que é então adicionada à tabela. Esta string recebe um número (endereço) único na tabela e passa a ficar disponível para uso futuro. O processo de codificação continua procurando sempre extrair a mais longa string possível da seqüência de entrada, estendendo esta string de mais um símbolo, adicionando-o ao dicionário, e assim por diante.

Na decodificação, inicialmente é criada uma tabela com os símbolos individuais da fonte. A partir daí, cada palavra-código recebida é traduzida, através desta tabela, em uma string original. Exceto para o caso do primeiro símbolo, cada vez que uma palavra-código é recebida, o dicionário é atualizado da seguinte forma: após a palavra-código ter sido traduzida, seu primeiro símbolo de fonte é adicionado à string anterior para acrescentar uma nova string ao dicionário. Desta forma, o decodificador, incrementalmente, reconstrói a mesma tabela usada no codificador.

Para melhor esclarecer o funcionamento do algoritmo LZW, consideremos o seguinte exemplo. Seja uma fonte ternária, cujos símbolos possíveis são *A*, *B* e *C*, e considere a seqüência de entrada:

ABAAAAAAACAABAAAAACABAAAAB ...

O dicionário inicialmente conterá os três símbolos da fonte, *A*, *B* e *C*. Por simplicidade, assumiremos que o tamanho máximo da tabela é 16 (4 bits são usados para representar o endereço de uma posição da tabela). Após a etapa de codificação, o dicionário da tabela 5 terá sido criado.

A seqüência de palavras-código geradas será:

0 - 1 - 0 - 5 - 6 - 2 - 5 - 4 - 6 - 0 - 8 - 10 - 15 ...

No início do processo de codificação, o método LZW é um tanto ineficiente, mas à medida que a tabela atinge um tamanho razoável, muitos símbolos podem ser representados por uma única palavra-código.

Tabela 5 - Codificação LZW

String de entrada	Palavra-código
<i>A</i>	0
<i>B</i>	1
<i>C</i>	2
<i>AB</i>	3
<i>BA</i>	4
<i>AA</i>	5
<i>AAA</i>	6
<i>AAAC</i>	7
<i>CA</i>	8
<i>AAB</i>	9
<i>BAA</i>	10
<i>AAAA</i>	11
<i>AC</i>	12
<i>CAB</i>	13
<i>BAAA</i>	14
<i>AAB...</i>	15

Na prática, os endereços da tabela costumam ocupar 12 bits. Para resolver o problema do limite máximo de strings do dicionário ser atingido, existem estratégias de atualização adaptativa do dicionário, de modo a refletir as estatísticas de fonte mais recentes.

Exercício resolvido

Dada a mensagem 'BABABABABABAB' produzida por uma fonte capaz de gerar os símbolos {A, B, C}, codificá-la e decodificá-la utilizando o método LZW.

Solução:

1) Codificação

O processo de codificação parte do dicionário já inicializado com três símbolos, A, B e C, cujas palavras-código são, respectivamente, 0, 1 e 2.

O algoritmo de codificação gerará a seqüência de saída:

1-0-3-5-4-7-1

e produzirá o dicionário a seguir:

Palavra-código	String de entrada
0	A
1	B
2	C
3	BA
4	AB
5	BAB
6	BABA
7	ABA
8	ABAB

2) Decodificação

O processo de decodificação também inicia com o dicionário já inicializado com três símbolos, A, B e C, cujas palavras-código são, respectivamente, 0, 1 e 2.

A cada palavra-código de entrada, ele procurará o símbolo correspondente no dicionário e efetuará a decodificação. Além de decodificar o último símbolo enviado, o decodificador também atualiza seu dicionário com strings geradas pelo codificador porém ainda não utilizadas, como por exemplo a string 'BABA' (palavra-código 6).

O detalhe mais sutil do algoritmo de decodificação surge quando se tenta decodificar uma palavra-código que ainda não foi utilizada pelo dicionário da etapa receptora. Neste caso (como veremos adiante para as palavras-código 5 e 7), o decodificador é capaz de deduzir a string correspondente através da concatenação da última string decodificada com o primeiro símbolo desta mesma string.

Pode-se provar que a etapa decodificadora recupera sem erros a informação original e constrói uma tabela idêntica à da etapa codificadora.

Detalhando passo a passo a seqüência de decodificação, temos:

1. Recebe a palavra-código 1.
2. Localiza a string correspondente no dicionário ('B') e decodifica.
3. Recebe a palavra-código 0.
4. Localiza a string correspondente no dicionário ('A') e decodifica.
5. Acrescenta a string formada pela concatenação da penúltima string decodificada com o primeiro símbolo da última string codificada ('B' + 'A' = 'BA') ao dicionário (palavra-código 3).
6. Recebe a palavra-código 3.
7. Localiza a string correspondente no dicionário ('BA') e decodifica.
8. Acrescenta a string formada pela concatenação da penúltima string decodificada com o primeiro símbolo da última string codificada ('A' + 'B' = 'AB') ao dicionário (palavra-código 4).
9. Recebe a palavra-código 5.

10. Como a palavra-código 5 ainda não tem correspondência no dicionário, o decodificador deduz a que string ela corresponde, concatenando a última string decodificada com o primeiro símbolo desta mesma string ('BA' + 'B' = 'BAB').
 11. Acrescenta a string recém-formada ao dicionário (palavra-código 5) e decodifica a palavra-código 5.
 12. Recebe a palavra-código 4.
 13. Localiza a string correspondente no dicionário ('AB') e decodifica.
 14. Acrescenta a string formada pela concatenação da penúltima string decodificada com o primeiro símbolo da última string codificada ('BAB' + 'A' = 'BABA') ao dicionário (palavra-código 6).
 15. Recebe a palavra-código 7.
 16. Como a palavra-código 7 ainda não tem correspondência no dicionário, o decodificador deduz a que string ela corresponde, concatenando a última string decodificada com o primeiro símbolo desta mesma string ('AB' + 'A' = 'ABA').
 17. Acrescenta a string recém-formada ao dicionário (palavra-código 7) e decodifica a palavra-código 7.
 18. Recebe a palavra-código 1.
 19. Localiza a string correspondente no dicionário ('B') e decodifica.
 20. Acrescenta a string formada pela concatenação da penúltima string decodificada com o primeiro símbolo da última string codificada ('ABA' + 'B' = 'ABAB') ao dicionário (palavra-código 8).
 21. Fim da decodificação.
-

Leitura complementar

O artigo de Nelson [Nelson 1989] apresenta os conceitos básicos da compressão LZW e inclui código-fonte em C.

6.4.2 Codificação *bit-plane*

Após termos examinado alguns dos principais métodos de remoção de redundância de codificação, consideraremos a seguir uma das várias técnicas de compressão de imagens que busca explorar suas redundâncias interpixel. Este método, denominado codificação *bit-plane*, é baseado no conceito de decomposição de uma imagem de múltiplos tons de cinza em uma série de imagens binárias, comprimindo a seguir cada uma delas utilizando um dos inúmeros métodos de compressão de imagens binárias. Nesta seção explicaremos uma possível forma de decomposição e analisaremos alguns dos mais populares métodos de compressão das imagens binárias resultantes.

Decomposição *bit-plane*

Os níveis de cinza de uma imagem monocromática podem ser representados na forma de um polinômio de base 2

$$a_{m-1}2^{m-1} + a_{m-2}2^{m-2} + \dots + a_12^1 + a_02^0. \quad (6.23)$$

Com base nesta propriedade, uma forma simples de decompor uma imagem em uma coleção de imagens binárias consiste em separar os m coeficientes do polinômio em m *bit planes* de 1 bit. O *bit plane* de ordem 0 será obtido a partir dos coeficientes a_0 de cada pixel, enquanto o *bit plane* de ordem ($m - 1$) conterá os coeficientes a_{m-1} . A desvantagem desta abordagem é que pequenas variações de tom de cinza na imagem original poderão produzir significativas variações de intensidade nos *bit planes* correspondentes. Por exemplo, se um pixel de intensidade 127 (01111111) estiver ao lado de outro, de intensidade 128 (1000000), cada *bit plane* conterá uma transição de 0 para 1 (ou de 1 para 0).

Para reduzir este problema, uma alternativa é decompor a imagem original usando o código de Gray para m bits. O código de Gray para m bits $g_{m-1} \dots g_2g_1g_0$ correspondente ao polinômio (6.23) pode ser calculado como:

$$\begin{aligned} g_i &= a_i \oplus a_{i+1} \quad 0 \leq i \leq m-2 \\ g_{m-1} &= a_{m-1}. \end{aligned} \quad (6.24)$$

onde o símbolo \oplus indica a operação ou-exclusivo. A principal propriedade do código de Gray é que duas palavras-código consecutivas diferem em apenas um bit. No caso dos valores 127 e 128, por exemplo, suas representações binárias equivalentes serão 11000000 e 01000000, respectivamente.

Codificação de áreas constantes

Uma forma simples e eficiente de comprimir uma imagem binária ou *bit plane* consiste na utilização de palavras-código especiais para designar grandes regiões de pixels 1 (brancos) ou 0 (pretos) contíguos. Nesta técnica, denominada 'codificação de áreas constantes' (CAC), a imagem é dividida em blocos de $m \times n$ pixels, os quais são classificados como: totalmente brancos, totalmente pretos ou mistos. A categoria mais provável recebe a palavra-código de um bit 0, enquanto as demais categorias recebem as palavras-código 10 e 11. A compressão é obtida porque os mn bits que seriam normalmente necessários para codificar o bloco são substituídos por apenas 1 ou 2 bits. Evidentemente, a palavra-código correspondente aos blocos mistos é usada apenas como prefixo do bloco codificado. Este conceito pode ser extrapolado em subdivisões subsequentes de cada bloco. Nos casos em que cada bloco ocupa um quarto da área total da imagem, a estrutura de dados equivalente à divisão sucessiva da imagem será uma árvore quaternária.

Uma pequena variação desta técnica consiste em codificar as áreas brancas com 0 e todos os outros blocos (inclusive os formados somente por pixels pretos) com um bit 1 seguido do padrão de bits do bloco. Esta abordagem, denominada *White Block Skipping* (WBS), apresenta bons resultados em imagens obtidas a partir de textos digitalizados, cujas características estruturais são favoráveis ao seu uso.

Uma modificação adicional na técnica WBS original consiste em codificar a imagem linha a linha, designando pela palavra-código 0 uma linha totalmente branca e codificando as linhas que contêm um ou mais pixels pretos por um bit 1 seguido do padrão de bits da linha.

Run-length unidimensional

Uma interessante alternativa à codificação de blocos constantes é a representação de cada linha de uma imagem ou *bit plane* através de uma seqüência de valores de comprimento, que representam os comprimentos das cadeias de 0's e 1's. Nesta técnica, denominada codificação *run-length*, a idéia básica é codificar cada grupo de 0's e 1's contíguos encontrados em uma varredura da esquerda para a direita através de seu comprimento, segundo uma convenção pré-estabelecida. Uma convenção comum é admitir que cada linha comece com uma seqüência de pixels brancos, que corresponde ao primeiro valor numérico encontrado. Caso a linha em questão comece com um pixel preto, codifica-se uma seqüência de brancos de comprimento zero.

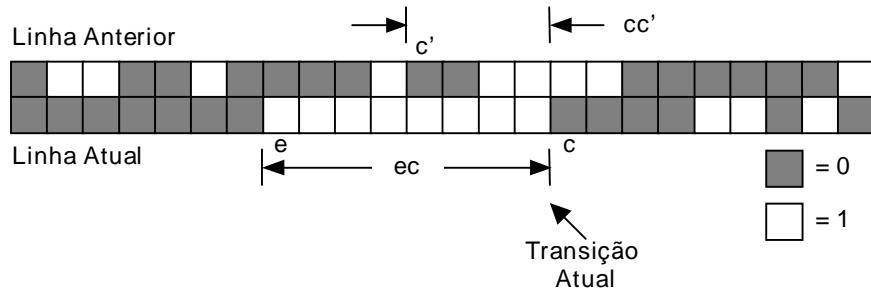
Embora a compressão *run-length* seja, por si só, um método eficaz de compressão de imagens, é possível obter uma compressão adicional, codificando as seqüências obtidas utilizando palavras-código de comprimento variável.

Run-length bidimensional

Os conceitos da codificação *run-length* unidimensional são extensíveis ao caso 2D. Uma das técnicas mais conhecidas a utilizar tal extensão é a codificação por endereço relativo (*relative address coding - RAC*), que se baseia na codificação das transições de branco para preto e vice-versa, levando em conta a linha atual e a imediatamente anterior. A figura 8 ilustra o método. Nela, a distância ec representa a distância entre a transição atual (c) e a transição anteriormente

ocorrida na mesma linha (e). Já cc' representa a distância entre c (na linha atual) e a transição similar da linha anterior que ocorre à direita da transição e da linha atual (c'). Se $ec \leq cc'$, a distância a ser codificada pelo método RAC, d , é considerada igual a ec e é utilizada para representar a transição atual em c . Caso contrário ($cc' < ec$), a distância d a ser codificada é igual a cc' .

A exemplo do *run-length* unidimensional, a codificação RAC requer a adoção de uma convenção que inclui o estabelecimento de transições imaginárias no início e fim de cada linha, bem como uma linha de início imaginária, totalmente branca, precedendo a primeira linha real. Além disso, a codificação RAC também costuma utilizar códigos de comprimento variável para codificar as distâncias (d) encontradas, como ilustra a figura 8(b).



(a)

Distância medida	Distância codificada	Código	Faixa de distância	Código $h(d)$
cc'	0	0	1 - 4	0 xx
ec ou cc' (esq)	1	100	5 - 20	10 xxxx
cc' (dir)	1	101	21 - 84	110 xxxxxx
ec	d ($d > 1$)	$111 h(d)$	85 - 340	1110 xxxxxxxx
cc' (c' à esq)	d ($d > 1$)	$1100 h(d)$	341 - 1364	11110 xxxxxxxx
cc' (c' à dir)	d ($d > 1$)	$1101 h(d)$	1365 - 5460	111110 xxxxxxxxx

(b)

Figura 8 - Exemplo ilustrativo do método RAC.

6.4.3 Codificação Preditiva sem Perdas

Para concluir nossa abordagem de técnicas de compressão sem perdas, apresentaremos agora uma técnica de codificação, cuja idéia básica é a remoção de redundância entre pixels próximos, extraíndo e codificando apenas a informação nova trazida por cada pixel. Esta informação 'nova' é normalmente definida como a diferença entre o valor real do pixel e o valor predito para aquele pixel. Por esta razão, esta técnica recebe o nome de codificação preditiva.

A figura 9 mostra os principais componentes de um codificador preditivo sem perdas e de seu respectivo decodificador. Em ambos os blocos, aparece um elemento-chave, que é o estágio preditor. À medida que cada pixel da imagem de entrada, indicado pela notação f_n , é introduzido no codificador, o preditor gera um número, que é o valor previsto para aquele pixel, com base em entradas anteriores. A saída do preditor é então arredondada para um valor inteiro, a que denominaremos \hat{f}_n , utilizado para calcular o 'erro de previsão', dado por

$$e_n = f_n - \hat{f}_n \quad (6.25)$$

que é então codificado por um código de comprimento variável (pelo 'codificador de símbolos' do estágio codificador), gerando o próximo elemento do conjunto de dados comprimidos. O decodificador da figura 9(b) reconstrói e_n a partir das palavras-código de comprimento variável recebidas e executa a operação inversa

$$f_n = e_n + \hat{f}_n. \quad (6.26)$$

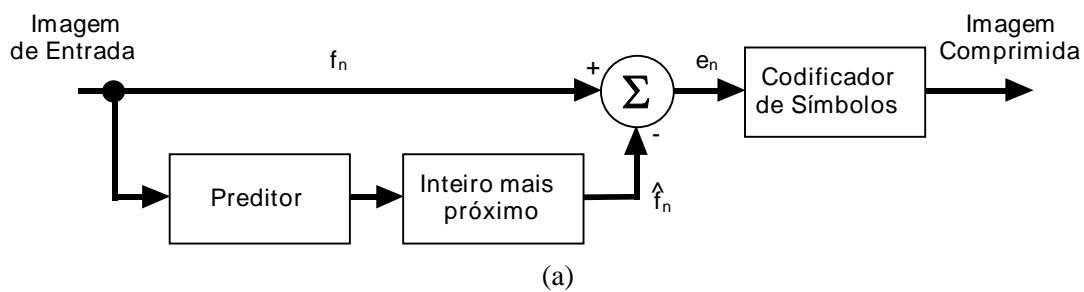
Diversos métodos locais, globais e adaptativos podem ser usados para gerar \hat{f}_n . Na maioria dos casos, entretanto, a predição é formada por uma combinação linear dos m pixels anteriores. Ou seja,

$$\hat{f}_n = \text{round} \left[\sum_{i=1}^m \alpha_i f_{n-i} \right] \quad (6.27)$$

onde m é a ordem do preditor linear, *round* é uma função utilizada para indicar a operação de arredondamento e α_i para $i = 1, 2, \dots, m$ são os coeficientes de predição. O subscrito n equivale a um índice espacial ou temporal de ocorrência do pixel. No caso particular da codificação preditiva linear 1-D, por exemplo, a eq. (6.27) pode ser reescrita como

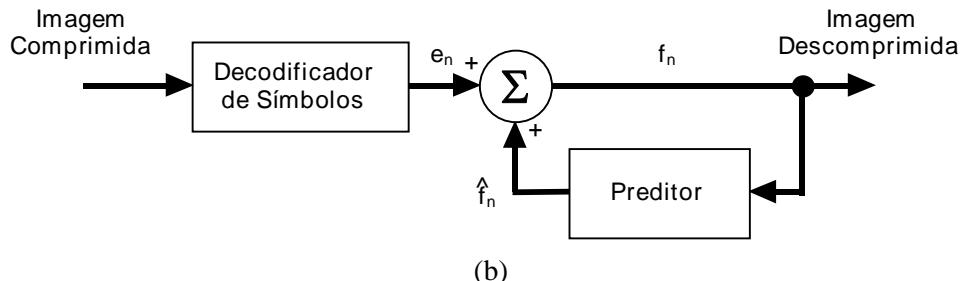
$$\hat{f}_n(x, y) = \text{round} \left[\sum_{i=1}^m \alpha_i f(x, y-i) \right] \quad (6.28)$$

onde cada variável está explicitamente relacionada a suas coordenadas espaciais x e y . Convém notar que na predição linear 1-D, $\hat{f}(x, y)$ é função apenas dos pixels anteriores da mesma linha. No caso 2-D, ele também depende de pixels que tenham aparecido em linhas anteriores, enquanto que no caso 3-D, também são levados em conta os pixels de *frames* anteriores. Por fim, convém notar que a eq. (6.28) não pode ser avaliada para os primeiros m pixels de uma linha, que portanto terão de ser codificados por outro método (e.g. Huffman), introduzindo um pequeno *overhead* no processo.



(a)

Figura 9 - Um modelo de codificação preditiva: (a) codificador; (b) decodificador.



(b)

Figura 9 - Continuação.

6.5 Compressão com perdas

Diferentemente das técnicas livres de erro apresentadas na seção anterior, a codificação com perdas baseia-se no conceito de comprometimento da precisão da imagem reconstruída em troca de uma maior compressão. Se a distorção resultante (que poderá ou não ser aparentemente visível) puder ser tolerada, o aumento na compressão poderá ser bastante significativo (de taxas na faixa de 3:1 até razões de compressão maiores ou iguais a 100:1). Conforme antecipamos na Seção 6.2, a principal diferença entre as técnicas de compressão com perdas e sem perdas é a presença ou ausência do bloco quantizador da figura 3.

6.5.1 Codificação preditiva com perdas

Duas das técnicas de compressão preditiva com perdas mais conhecidas são a modulação delta (DM) e a Modulação por Codificação Diferencial de Pulso (DPCM). Para analisá-las, acrescentaremos um quantizador ao modelo introduzido na Seção 6.4.3. e examinaremos o compromisso resultante entre precisão na reconstrução da imagem e desempenho da etapa de compressão. Como mostra a figura 10, o quantizador (que absorve a função de arredondamento para o número inteiro mais próximo do codificador sem erros) é inserido entre o codificador de símbolos e o ponto no qual o erro de predição é calculado. Ele mapeia o erro de predição em uma faixa limitada de saída, denotada \dot{e}_n , a qual estabelece a quantidade de compressão e distorção associados à codificação preditiva com perdas.

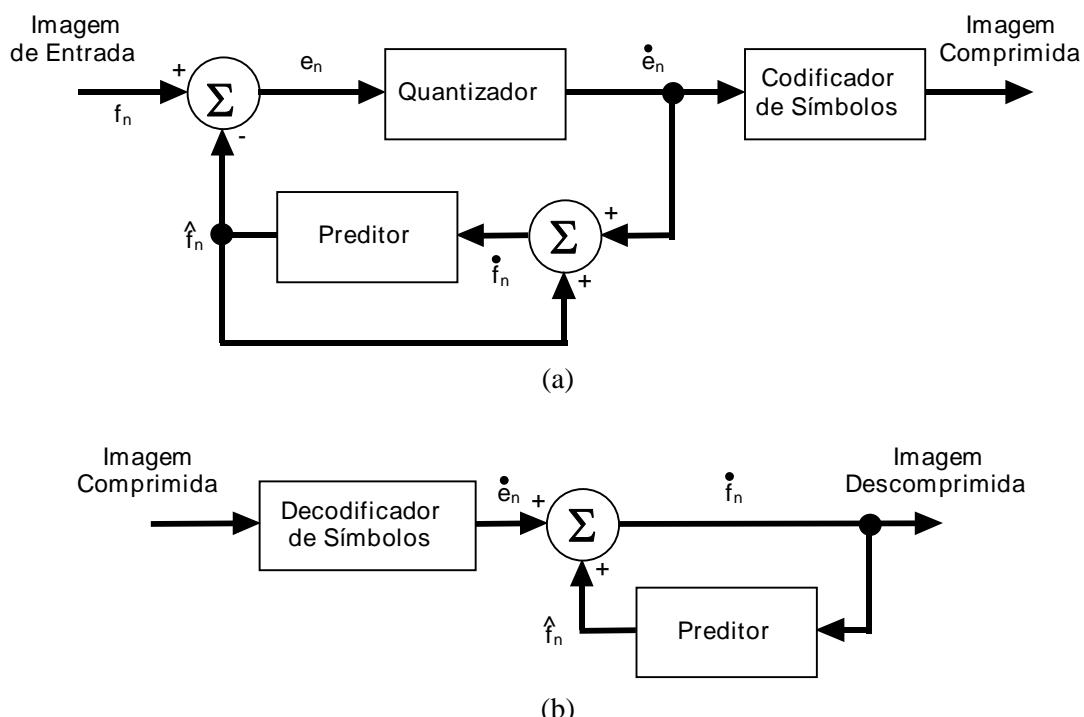


Figura 10 - Um modelo de codificação preditiva com perdas: (a) codificador; (b) decodificador.

A fim de acomodar a inserção do estágio de quantização, o codificador livre de erros da figura 9(a) deve ser alterado de tal maneira que as previsões geradas pelo codificador e decodificador sejam equivalentes. Como mostra a figura 10(a), isto é obtido colocando-se o predictor dentro de um laço de realimentação, onde sua entrada, denotada \dot{f}_n , é gerada como uma função de previsões anteriores e dos erros quantizados correspondentes. Ou seja,

$$\dot{f}_n = \dot{e}_n + \hat{f}_n \quad (6.29)$$

onde \hat{f}_n aparece como definido na Seção 6.4.3. Esta configuração de malha fechada previne o surgimento de erros na saída do decodificador. Da figura 10(b) pode-se deduzir que a saída do decodificador é também fornecida pela equação (6.29).

Modulação Delta (DM)

A modulação Delta (DM) constitui um exemplo simples e bastante conhecido de codificação preditiva com perdas no qual o preditor e o quantizador são definidos como

$$\hat{f}_n = \alpha \dot{f}_{n-1} \quad (6.30)$$

e

$$\dot{e}_n = \begin{cases} +\zeta & \text{para } e_n > 0 \\ -\zeta & \text{caso contrario} \end{cases} \quad (6.31)$$

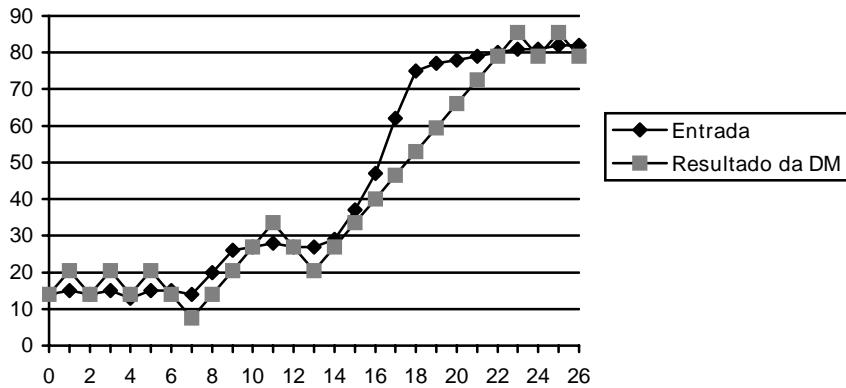
onde α é um coeficiente de predição (normalmente menor que 1) e ζ é uma constante positiva. A saída do quantizador, \dot{e}_n , pode ser representada por um único bit, portanto o codificador de símbolos da figura 10(a) pode utilizar código de comprimento fixo e igual a 1 bit. O código DM resultante utiliza 1 bit/pixel. A tabela 6 ilustra a mecânica do processo de modulação delta, bem como os cálculos necessários para comprimir e reconstruir a sequência de entrada {14, 15, 14, 15, 13, 15, 15, 14, 20, 26, 27, 28, 27, 29, 37, 47, 62, 75, 77, 78, 79, 80, 81, 81, 82, 82} com $\alpha = 1$ e $\zeta = 6.5$.

O processo se inicia com a transferência do primeiro pixel de entrada para o decodificador. Uma vez estabelecida a condição inicial $\dot{f}_0 = f_0 = 14$, tanto no codificador quanto no decodificador, as saídas subsequentes podem ser calculadas através das equações (6.30), (6.25), (6.31) e (6.29). Logo, quando $n = 1$, por exemplo, $\hat{f}_1 = (1)(14) = 14$, $e_1 = 15 - 14 = 1$, $\dot{e}_1 = +6,5$ (porque $e_1 > 0$), $\dot{f}_1 = 6,5 + 14 = 20,5$ e o erro de reconstrução é $(15 - 20,5)$ ou -5,5 níveis de cinza.

A figura 11 ilustra graficamente os dados da tabela 6. Nela podem ser observados os dois principais problemas da técnica DM: o ruído granular na região em que a imagem de entrada apresenta valores praticamente uniformes, entre a 1^a e a 8^a amostra (porque ζ é muito grande para acompanhar estas pequenas flutuações), e a dificuldade em acompanhar uma transição abrupta nos valores de entrada (*slope overload*), presente no intervalo entre a 14^a e a 19^a amostra, porque neste caso ζ é muito pequeno para acompanhar as variações de entrada. Esta limitação da modulação Delta provocaria uma suavização das bordas da imagem comprimida, enquanto o ruído granular distorceria regiões homogêneas da imagem. Estas limitações são comuns a todas as formas de compressão com perdas e algumas formas de minimizá-las são o projeto de preditores e quantizadores ótimos e a utilização de técnicas adaptativas.

Tabela 6 - Codificação DM

Entrada		Codificador			Decodificador		Ero	
n	f	\hat{f}	e	\dot{e}	\dot{f}	\hat{f}	\dot{f}	$[f - \dot{f}]$
0	14	—	—	—	14,0	—	14,0	0,0
1	15	14,0	1,0	6,5	20,5	14,0	20,5	-5,5
2	14	20,5	-6,5	-6,5	14,0	20,5	14,0	0,0
3	15	14,0	1,0	6,5	20,5	14,0	20,5	-5,5
4	13	20,5	-7,5	-6,5	14,0	20,5	14,0	-1,0
5	15	14,0	1,0	6,5	20,5	14,0	20,5	-5,5
6	15	20,5	-5,5	-6,5	14,0	20,5	14,0	1,0
7	14	14,0	0,0	-6,5	7,5	14,0	7,5	6,5
8	20	7,5	12,5	6,5	14,0	7,5	14,0	6,0
9	26	14,0	12,0	6,5	20,5	14,0	20,5	5,5
10	27	20,5	6,5	6,5	27,0	20,5	27,0	6,5
11	28	27,0	1,0	6,5	33,5	27,0	33,5	-5,5
12	27	33,5	-6,5	-6,5	27,0	33,5	27,0	0,0
13	27	27,0	0,0	-6,5	20,5	27,0	20,5	6,5
14	29	20,5	8,5	6,5	27,0	20,5	27,0	2,0
15	37	27,0	10,0	6,5	33,5	27,0	33,5	3,5
16	47	33,5	13,5	6,5	40,0	33,5	40,0	7,0
17	62	40,0	22,0	6,5	46,5	40,0	46,5	15,5
18	75	46,5	28,5	6,5	53,0	46,5	53,0	22,0
19	77	53,0	24,0	6,5	59,5	53,0	59,5	17,5
20	78	59,5	18,5	6,5	66,0	59,5	66,0	18,5
21	79	66,0	13,0	6,5	72,5	66,0	72,5	13,0
22	80	72,5	7,5	6,5	79,0	72,5	79,0	7,5
23	81	79,0	2,0	6,5	85,5	79,0	85,5	2,0
24	81	85,5	-4,5	-6,5	79,0	85,5	79,0	-4,5
25	82	79,0	3,0	6,5	85,5	79,0	85,5	3,0
26	82	85,5	-3,5	-6,5	79,0	85,5	79,0	-3,5

Figura 11 - Representação gráfica dos valores originais e após codificação, ilustrando os problemas de ruído granular e *slope overload*.

Modulação por Codificação Diferencial de Pulsos (DPCM)

A técnica DPCM, proposta por [Cutler 1952], é a mais conhecida técnica de codificação preditiva. Ela utiliza um preditor ótimo, cujo objetivo é minimizar o erro médio quadrático de predição, assumindo que o erro de quantização é desprezível (ou seja, $\hat{e}_n \approx e_n$) e que o valor predito de um pixel depende de uma combinação linear de m pixels anteriores, isto é,

$$\hat{f}_n = \sum_{i=1}^m \alpha_i f_{n-i} . \quad (6.32)$$

O número de pixels considerados na predição, m , é normalmente conhecido como 'ordem do preditor' e estudos realizados sobre imagens de TV e de radiografias demonstraram que não há ganho significativo em se utilizar ordem superior a 3.

O conjunto de coeficientes do preditor ótimo normalmente é obtido modelando a imagem como uma fonte de Markov 2-D, cuja função de auto-correlação é

$$\cdot E\{f(x, y)f(x - i, y - j)\} = \sigma^2 \rho_v^i \rho_h^j \quad (6.33)$$

resultando no preditor linear de 4^a ordem

$$\cdot \hat{f}(x, y) = \alpha_1 f(x, y - 1) + \alpha_2 f(x - 1, y - 1) + \alpha_3 f(x - 1, y) + \alpha_4 f(x - 1, y + 1) \quad (6.34)$$

cujos coeficientes ótimos são

$$\cdot \alpha_1 = \rho_h \quad \alpha_2 = -\rho_v \rho_h \quad \alpha_3 = \rho_v \quad \alpha_4 = 0 \quad (6.35)$$

onde ρ_h e ρ_v são os coeficientes de correlação horizontal e vertical da imagem, respectivamente.

Leitura complementar

A Seção 6.4 de [Lynch 1985] é totalmente dedicada à modulação DPCM, bem como sua variante adaptativa (ADPCM).

Para uma revisão dos conceitos de modelos de Markov e suas aplicações em estratégias de predição sugerimos a Seção 1.10 de [Williams 1991].

A etapa de quantização

Um quantizador é em essência um bloco com função de transferência em forma de escada, que mapeia todos os possíveis valores de entrada em um menor número de níveis de saída. Desta forma, o número de símbolos a serem codificados se reduz, às custas de um maior erro na imagem reconstruída. A quantização individual de cada valor do sinal de entrada é denominada quantização escalar (QE), enquanto a quantização conjunta de um bloco de valores do sinal de entrada recebe o nome de quantização vetorial (QV). Para um mesmo esquema de codificação, pode-se garantir que a QV produz resultados iguais ou superiores que a QE. Porém, em alguns casos, o ganho em termos de compressão não compensa a complexidade adicional de implementação. A técnica de QE mais conhecida é o quantizador de Lloyd-Max.

Leitura complementar

A Seção 2.5 de [Lynch 1985] é totalmente dedicada a conceitos e métodos de quantização.

6.5.2 Codificação por transformadas

As técnicas discutidas na Seção 6.5.1 operam diretamente nos pixels de uma imagem e por isso são chamadas de métodos de domínio espacial. Nesta seção, consideraremos as técnicas de compressão baseadas na modificação da transformada de uma imagem. Na codificação por transformadas, utiliza-se uma operação matemática linear reversível para mapear a imagem dentro de um conjunto de coeficientes, os quais em seguida são quantizados e codificados. Para a maioria das imagens naturais, um número significativo de coeficientes têm pequena magnitude e podem, portanto, ser quantizados (ou mesmo descartados), causando pouca distorção na imagem decodificada. Diversas transformadas matemáticas conhecidas podem ser utilizadas para transformar os dados da imagem. Quanto maior a capacidade da transformada de compactar informação em poucos coeficientes, melhor ela será para fins de compressão.

A figura 12 mostra um sistema de codificação por transformadas típico. O decodificador implementa a seqüência inversa dos estágios (com exceção da função de quantização) do codificador, o qual realiza quatro operações principais: decomposição da imagem original em subimagens, cálculo da transformada direta, quantização e codificação. O objetivo da transformada direta é descorrelacionar os pixels de cada subimagem e reunir o maior número de informações possível no menor número de coeficientes. O estágio de quantização, a seguir, elimina seletivamente, ou quantiza mais grosseiramente, os coeficientes que carregam o menor número de informações. Estes coeficientes têm o menor impacto sobre a qualidade da subimagem reconstruída. O processo termina com a codificação (normalmente utilizando-se palavras-código de comprimento variável) dos coeficientes quantizados. Alguns ou todos os estágios da codificação por transformadas podem ser adaptados ao conteúdo local da imagem, ao que se denomina codificação adaptativa por transformadas.

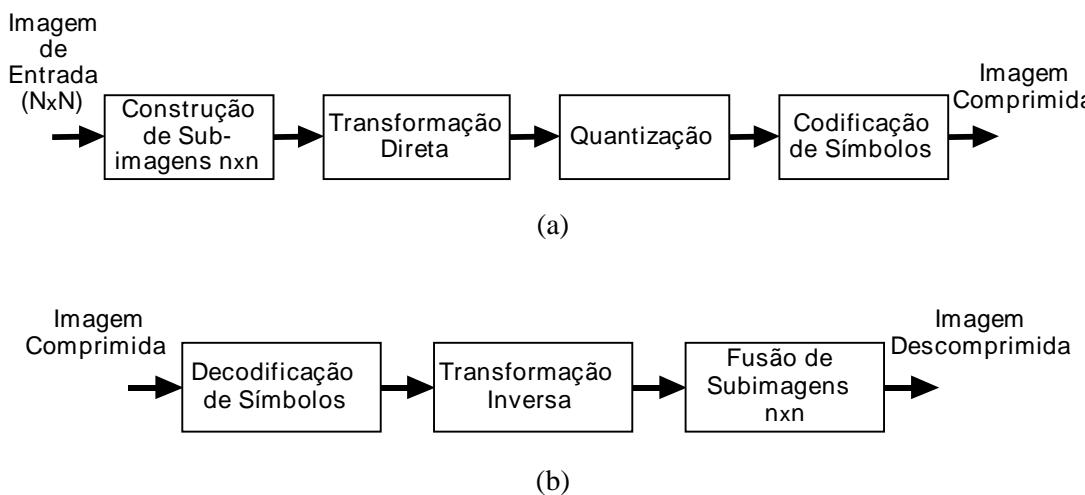


Figura 12 - Codificação por transformadas: (a) codificador; (b) decodificador.

Seleção de Transformadas

A escolha da melhor transformada matemática para uma dada aplicação depende de vários parâmetros, dentre eles a quantidade de erro de reconstrução tolerável e os recursos computacionais disponíveis. A maior parte da compressão é obtida durante a quantização dos coeficientes transformados (e não durante o estágio de transformação propriamente dito). Uma das transformadas mais utilizadas é a transformada discreta de cossenos (DCT), cuja formulação matemática é apresentada a seguir.

A DCT direta 1-D é definida como:

$$C(u) = \alpha(u) \sum_{x=0}^{N-1} f(x) \cos \left[\frac{(2x+1)u\pi}{2N} \right] \quad (6.36)$$

Para $u = 0, 1, 2, \dots, N-1$.

Similarmente, a DCT inversa 1-D é definida como:

$$f(x) = \sum_{u=0}^{N-1} \alpha(u) C(u) \cos \left[\frac{(2x+1)u\pi}{2N} \right] \quad (6.37)$$

Para $x = 0, 1, 2, \dots, N-1$. Em ambas equações (6.36) e (6.37), α é:

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & p/u = 0 \\ \sqrt{\frac{2}{N}} & p/u = 1 \end{cases} \quad (6.38)$$

A DCT direta 2-D é dada por:

$$C(u,v) = \alpha(u)\alpha(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cos \left[\frac{(2x+1)u\pi}{2N} \right] \cos \left[\frac{(2y+1)v\pi}{2N} \right] \quad (6.39)$$

para $u, v = 0, 1, 2, \dots, N-1$.

Enquanto a DCT inversa 2-D pode ser calculada como:

$$f(x,y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \alpha(u)\alpha(v) C(u,v) \cos \left[\frac{(2x+1)u\pi}{2N} \right] \cos \left[\frac{(2y+1)v\pi}{2N} \right] \quad (6.40)$$

para $x, y = 0, 1, 2, \dots, N-1$. Onde α é dada pela eq.(6.38).

Nos últimos anos a DCT tem-se tornado uma das ferramentas matemáticas mais utilizadas em padrões de compressão de imagens, como o JPEG, o H.261 e o MPEG, que serão abordados na Seção 6.6.

6.5.3 Outras técnicas

Inúmeras outras técnicas de compressão de imagens vêm sendo testadas, em alguns casos com resultados bastante promissores. Dentre elas destacamos o uso de fractais [Barnsley e Sloan 1988], a transformada Wavelet [Mallat 1989] [Rioul e Vetterli 1991], a codificação por sub-bandas [Vetterli 1984] e codificação preditiva com árvore binária (BTPC - *Binary Tree Predictive Coding*).

6.6 Padrões de compressão de imagens

A adoção de padrões de compressão de imagens traz vários benefícios, dentre os quais relacionamos: (1) facilita o intercâmbio de imagens comprimidas entre vários dispositivos e aplicações; (2) permite o uso do mesmo hardware e software em uma ampla gama de produtos, reduzindo custos e encurtando o ciclo de desenvolvimento de novos produtos; e (3) provê referenciais de qualidade esperada para imagens comprimidas. Os esforços de padronização de algoritmos de compressão de imagens estão divididos em três tipos: imagens binarizadas, imagens estáticas (monocromáticas ou coloridas) e seqüências de imagens (vídeo digital). Nesta seção abordaremos o padrão adotado pelo CCITT para transmissão de imagens binarizadas através de fac-símile grupos 3 e 4, o padrão JPEG de compressão de imagens estáticas e os padrões H.261, H.263 e MPEG de compressão de vídeo.

6.6.1 Padrões CCITT para fac-símiles Grupo 3 e Grupo 4

O padrão CCITT para fac-símile Grupo 3 e Grupo 4 é o mais conhecido e utilizado padrão de compressão de imagens binárias. O padrão adotado para o fax Grupo 3 utiliza uma codificação *run-length* 1-D não-adaptativa na qual $K - 1$ linhas de cada conjunto de K linhas (para $K = 2$ ou 4) podem ser opcionalmente codificadas por um algoritmo de exploração da redundância entre linhas consecutivas (MREAD). Já no fax Grupo 4, apenas a codificação 2-D é implementada.

Codificação unidimensional

Nesta técnica, cada linha completa é lida e convertida, para fins de transmissão, em uma seqüência de segmentos de linha (*run lengths*) alternadamente brancos e pretos. Assume-se que todas as linhas começam com um segmento branco para garantir que o receptor mantenha sincronismo de cor. Caso a linha lida comece com um segmento preto, então é inserido um segmento branco de comprimento zero no início da transmissão da linha.

Cada segmento da linha é então codificado, segundo o seu comprimento e cor, através das tabelas 7 e 8. São utilizados códigos separados para representar segmentos de linha brancos e pretos. Este tipo de codificação é conhecido como 'Código de Huffman Modificado' (MHC - *Modified Huffman Code*). Através dele pode-se representar um segmento de linha até o máximo comprimento admissível para uma linha completa, que é de 1728 pixels.

Existem duas categorias de palavras-código, denominadas 'palavras-código de terminação' (PCT) (tabela 7) e 'palavras-código de composição' (PCC) (tabela 8). Segmentos de linha com comprimento entre 64 e 1728 pixels são codificados por uma PCC seguida por uma PCT. A PCC representa um valor de comprimento $64 \times N$ (onde N é um inteiro entre 1 e 27) que é igual a, ou menor que, o valor do comprimento a ser codificado. A PCT seguinte especifica a diferença entre a PCC e o valor real do comprimento a ser codificado.

Por exemplo, suponhamos a codificação de um segmento de linha branco com 200 pixels. Como este comprimento é maior do que 63, é necessário compor o código com uma PCC e uma PCT. O maior valor de PCC inferior ao valor a ser codificado é igual a 192 ($200 / 64 = 3$ com resto 8, portanto $N = 3$). A PCT codificará então a diferença entre 200 e 192 que é igual a oito. Consultando-se as tabelas 7 e 8, obteremos os códigos 010111 e 10011 para a PCC e a PCT, respectivamente, resultando na palavra-código 01011110011. Neste segmento, em particular, foram utilizados apenas 11 bits para codificar 200 pixels.

A codificação de uma linha termina quando todos os segmentos de linha, perfazendo um total de 1728 pixels, forem transmitidos. Cada linha codificada é seguida pela palavra-código de fim de linha (EOL - *End Of Line*), codificada como 000000000000. A palavra-código EOL é uma seqüência única que não pode ocorrer dentro de uma linha válida de dados codificados. Ela pode ser detectada independentemente de como o receptor divide a linha codificada em palavras-código. Na verdade, o EOL representa uma informação redundante pois, após contar 1728 pixels, o receptor já sabe que a linha terminou. Esta redundância existe para tornar o sistema mais tolerante a falhas. Assim, se uma falha de transmissão corromper alguns dos dados codificados na linha lida, este erro não impedirá o verdadeiro EOL de ser detetado.

Codificação bidimensional

A técnica de codificação bidimensional adotada pelo CCITT tanto para o fax Grupo 3 quanto para o Grupo 4 recebe o nome de *Modified Relative Element Address Designate* (MREAD). Esta técnica é um esquema de codificação seqüencial linha-a-linha, onde se transmite apenas a informação nova (em relação a linha anterior) que está contida na linha sendo codificada. A linha anterior é chamada de linha de referência. No fax Grupo 4, a linha de referência para a primeira linha da imagem é uma linha imaginária composta somente de pixels brancos. Já na codificação padronizada para o Grupo 3, para obtermos a primeira linha de referência, já que não existe linha anterior, é utilizada a codificação unidimensional com o código MHC. Caso haja uma falha na transmissão dos dados codificados de uma linha, o erro resultante vai se propagar pelas linhas seguintes. Para evitar esse efeito, é transmitida regularmente uma linha codificada no modo unidimensional. A técnica MHC é aplicada então à primeira linha de cada K sucessivas linhas, de modo a confechar a área danificada em caso de erros devidos a falhas de

transmissão. As subsequentes ($K - 1$) linhas são codificadas linha-a-linha, pela técnica bidimensional MREAD.

Tabela 7 - Código de Huffman Modificado - Códigos de terminação (PCT)

<i>Run-length</i>	Palavra-código Branco	Palavra-código Preto	<i>Run-length</i>	Palavra-código Branco	Palavra-código Preto
0	00110101	0000110111	32	00011011	000001101010
1	000111	010	33	00010010	000001101011
2	0111	11	34	00010011	000011010010
3	1000	10	35	00010100	000011010011
4	1011	011	36	00010101	000011010100
5	1100	0011	37	00010110	000011010101
6	1110	0010	38	00010111	000011010110
7	1111	00011	39	00101000	000011010111
8	10011	000101	40	00101001	000001101100
9	10100	000100	41	00101010	000001101101
10	00111	0000100	42	00101011	000011011010
11	01000	0000101	43	00101100	000011011011
12	001000	0000111	44	00101011	000001010100
13	000011	00000100	45	00000100	000001010101
14	110100	00000111	46	00000101	000001010110
15	110101	000011000	47	00001010	000001010111
16	101010	0000010011	48	00001011	000001100100
17	101011	0000011000	49	01010010	000001100101
18	0100111	0000001000	50	01010011	000001010010
19	0001100	00001100111	51	01010100	000001010011
20	0001000	00001101000	52	01010101	000000100100
21	0010111	00001101100	53	00100100	000000110111
22	0000011	00000110111	54	00100101	000000111000
23	0000100	00000101000	55	01011000	000000100111
24	0101000	00000010111	56	01011001	000000101000
25	0101011	00000011000	57	01011010	000001011000
26	0010011	000011001010	58	01011011	000001011001
27	0100100	000011001011	59	01001010	000000101011
28	0011000	000011001100	60	01001011	000000101100
29	00000010	000011001101	61	00110010	000001011010
30	00000011	000001101000	62	00110011	000001100110
31	00011010	000001101001	63	00110100	000001100111

Tabela 8 - Código de Huffman Modificado - Códigos de composição (PCC)

<i>Run-length</i>	Palavra-código Branco	Palavra-código Preto
64	11011	0000001111
128	10010	000011001000
192	010111	000011001001
256	010111	0000001011011
320	00110110	0000000110011
384	00110111	0000000110100
448	01100100	0000000110101
512	01100101	00000001101100
576	01101000	00000001101101
640	01100111	00000001001010
704	011001100	00000001001011
768	011001101	00000001001100
832	011010010	00000001001101
896	011010011	00000001110010
960	011010100	00000001110011
1024	011010101	00000001110100
1088	011010110	00000001110100
1152	011010111	00000001110110
1216	011011000	00000001110111
1280	011011001	00000001010010
1344	011011010	00000001010011
1408	011011011	00000001010100
1472	010011000	00000001010101
1536	010011001	00000001011010
1600	010011010	00000001011011
1664	0110000	00000001100100
1728	010011011	00000001100101

A idéia básica da técnica MREAD é codificar a posição de cada transição de preto para branco ou de branco para preto em relação a um elemento de referência a_0 , situado na linha atual.

A figura 13 ilustra esquematicamente o procedimento de codificação MREAD em um fluxograma. Através dela pode-se notar que inicialmente é necessário localizar os elementos de referência para o processo de codificação, que são:

- a_0 : pixel de início na linha atual, que se torna o pixel de referência;
- a_1 : pixel onde ocorre a primeira transição à direita de a_0 na linha atual;
- a_2 : pixel onde ocorre a primeira transição à direita de a_1 na linha atual;
- b_1 : pixel de cor oposta a a_0 , que indica a primeira transição à direita de a_0 na linha anterior;
- b_2 : pixel onde ocorre a primeira transição à direita de b_1 na linha anterior.

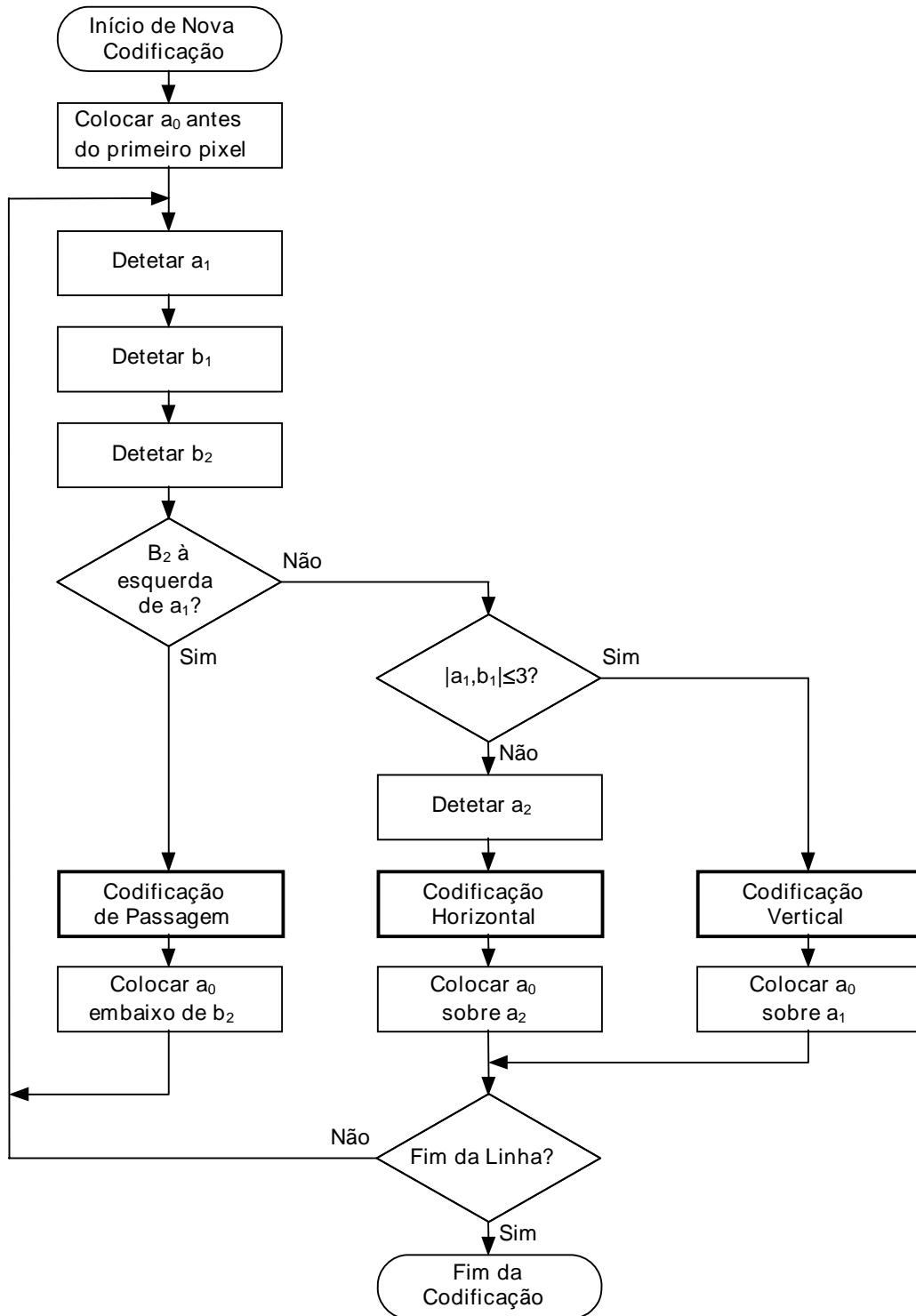


Figura 13 - Procedimento de codificação MREAD. A notação $|a_1, b_1|$ indica o valor absoluto da distância entre os elementos a_1 e b_1 .

Se algum destes pixels não puder ser detetado porque a correspondente transição não existe, o valor correspondente será atribuído a um pixel imaginário à direita do último pixel da linha em questão. A figura 14 ilustra duas situações possíveis e os correspondentes elementos de referência detetados.

Após a identificação dos elementos de referência são efetuados dois testes simples para determinar um dos três modos de codificação possíveis, que são: modo de passagem, modo vertical e modo horizontal. O teste inicial compara a localização de b_2 em relação a a_1 . O segundo calcula a distância (em pixels) entre a_1 e b_1 e compara com o valor 3. De acordo com os MARQUES FILHO, Ogê; VIEIRA NETO, Hugo. *Processamento Digital de Imagens*, Rio de Janeiro: Brasport, 1999. ISBN 8574520098.

resultados destes testes, um dos três modos de codificação 2-D será utilizado, um novo elemento de referência a_0 será determinado e o procedimento recomeça, conforme indica o fluxograma da figura 13.

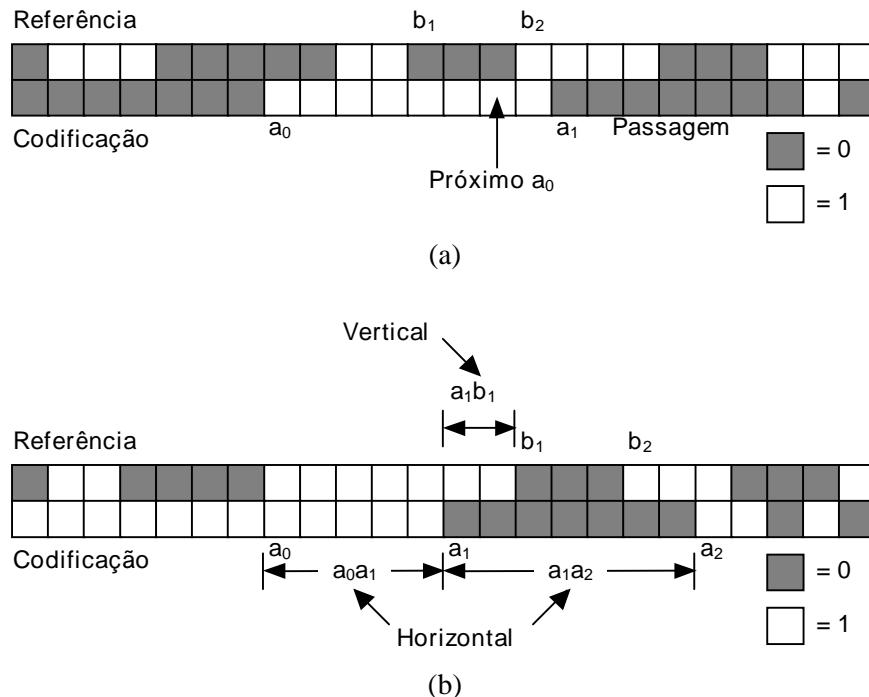


Figura 14 - Parâmetros de codificação 2-D para os casos: (a) modo de passagem; (b) modos horizontal e vertical.

Tabela 9 - Palavras-código para codificação CCITT 2-D

Modo	Elementos a serem codificados	Notação	Palavra-Código	
Passagem	b_1, b_2	P	0001	
Horizontal	$a_0 a_1, a_1 a_2$	H	$001 + M(a_0 a_1) + M(a_1 a_2)$	
	a_1 sob b_1	$a_1 b_1 = 0$	V(0)	1
	a_1 1 pixel à direita de b_1	$a_1 b_1 = 1$	$V_D(1)$	011
	a_1 2 pixels à direita de b_1	$a_1 b_1 = 2$	$V_D(2)$	000011
Vertical	a_1 3 pixels à direita de b_1	$a_1 b_1 = 3$	$V_D(3)$	0000011
	a_1 1 pixel à esquerda de b_1	$a_1 b_1 = 1$	$V_E(1)$	010
	a_1 2 pixels à esquerda de b_1	$a_1 b_1 = 2$	$V_E(2)$	000010
	a_1 3 pixels à esquerda de b_1	$a_1 b_1 = 3$	$V_E(3)$	0000010

Nota: A codificação $M(.)$ do modo horizontal significa que as distâncias indicadas entre parênteses são codificadas usando as palavras-código das tabelas 7 e 8.

A tabela 9 define os códigos utilizados para cada um dos três modos possíveis. No modo de passagem, que especificamente exclui o caso em que b_2 está diretamente acima de a_1 , apenas a palavra-código 0001 basta. Como se pode ver na figura 14(a), este modo identifica seqüências de pixels brancos ou pretos na linha de referência que não sobrepõem as seqüências de branco ou preto da linha atual. No modo horizontal, as distâncias de a_0 até a_1 e de a_1 até a_2 devem ser codificadas usando os códigos MHC das tabelas 7 e 8 e os resultados concatenados ao padrão 001. Este procedimento está indicado na tabela 9 usando a notação $001 + M(a_0 a_1) + M(a_1 a_2)$.

$M(a_1 \ a_2)$. Finalmente, no modo vertical, uma das sete palavras-código correspondentes à distância entre a_1 e b_1 é atribuída.

Convém notar que embora a figura 14(b) indique os modos de codificação horizontal e vertical para fins ilustrativos, ela reflete uma situação em que a codificação será no modo vertical, uma vez que b_2 está à direita de a_1 e a distância entre a_1 e b_1 é menor que 3. Logo, a situação ilustrada é $V_E(2)$, implicando no uso da palavra-código 000010. Na próxima iteração a_0 será movido para a posição onde até então estava a_1 .

Leitura complementar

O leitor interessado em um estudo comparativo de diversas técnicas de codificação e compressão de imagens binárias, desenvolvidas no final da década de 70, quando o CCITT estudava qual(is) delas utilizar no padrão G3 dos equipamentos fac-símile, encontrará em [Allens et al. 1980] uma boa resenha em português sobre o tema. Dentre outros artigos relevantes na área de compressão e codificação de imagens binárias, podemos indicar: [Hunter 1980], [Ting 1980], [Kawaguchi 1980], [Yasuda 1980], [Arps 1980] e [Usubuchi 1980].

[Pratt et al. 1980] apresentam um interessante sistema de compressão de dados híbrido, em que os trechos de um documento a ser transmitido por fax que forem reconhecidos como texto passam pela ação de um software de reconhecimento de caracteres (OCR) e são representados por seu código ASCII, enquanto os demais trechos da imagem (linhas, figuras, informação ilegível) são codificados por técnicas *run-length*.

Jayme [Jayme 1992] [Jayme et al. 1993] propõe um método para compactação de imagens de assinaturas digitalizadas com dois tons de cinza.

6.6.2 JPEG

Sigla de *Joint Photographic Experts Group*, comitê formado pelo ISO, ITU-T e IEC que originou o padrão. Estabelecido em 1991, foi projetado para comprimir imagens naturais coloridas ou monocromáticas com até 65536 x 65536 pixels.

Características do JPEG

O JPEG é otimizado para fotografias, figuras e imagens naturais (imagens em tom contínuo de cores ou níveis de cinza, sem bordas abruptas). No caso de imagens com poucas cores (e.g. linhas, figuras simples, caracteres etc.), seu desempenho é inferior, por exemplo, ao padrão GIF (*Graphics Interchange Format*). O JPEG não deve ser usado com imagens bitonais (preto e branco), sendo necessário existir, no mínimo, 16 níveis de cinza.

Quanto à resolução de cores, o JPEG toma como base uma imagem com 8 bits/amostra, ou seja, 24 bits/pixel para imagens em cores ou 8 bits/pixel para tons de cinza. No caso de compressão sem perdas é permitido usar qualquer valor entre 2 e 16 bits/amostra.

Apesar de existirem implementações para compressão sem perdas, o JPEG é fundamentalmente uma técnica de compressão com perdas baseado na DCT (Transformada Discreta de Cossenos). O algoritmo explora as limitações do olho humano, notadamente o fato de que variações de cor são menos perceptíveis que variações de brilho.

O grau de perda pode ser variado ajustando-se parâmetros de compressão. Para imagens coloridas as taxas de compressão podem variar de 10:1 a 20:1 sem perda visível. Taxas de 30:1 a 50:1 podem ser atingidas com pequenas a moderadas distorções. Para imagens de baixa qualidade (do tipo *preview*) taxas de até 100:1 são praticáveis.

Imagens em tons de cinza não podem ser comprimidas a taxas tão elevadas porque, como já foi citado, o olho humano é mais sensível a variações de brilho do que a variações de cor. Perdas visíveis podem surgir quando imagens monocromáticas são comprimidas a taxas maiores que 5:1.

O JPEG tem quatro modos de operação:

- seqüencial: a imagem é codificada em uma única varredura (da esquerda para a direita, de alto a baixo);
- progressiva: a imagem é codificada em múltiplas varreduras, aumentando a qualidade e resolução a cada nova varredura;

- hierárquica: a imagem é codificada em múltiplas resoluções;
- sem perda.

A vantagem das técnicas progressiva e hierárquica é permitir ao usuário selecionar um nível de qualidade variável para a imagem. Por exemplo, num meio de transmissão lento pode ser desejável transmitir uma imagem de menor qualidade. Numa aplicação tipo *browser*, o usuário poderia escolher uma imagem específica dentre diversas imagens de baixa resolução para então solicitá-la em maiores detalhes.

É importante lembrar que o JPEG não é um formato de arquivo, mas apenas uma família de algoritmos de compressão. Inicialmente o comitê não estabeleceu nenhum padrão para o formato de arquivo a ser utilizado. O que normalmente se conhece como 'arquivo JPEG' é um formato de arquivo chamado JFIF (*JPEG File Interchange Format*) definido pela C-Cube Microsystems e que se tornou o padrão *de facto* na Internet. Existem outros formatos de arquivos, inclusive o SPIFF (compatível com o JFIF), que foi definido posteriormente pelo comitê JPEG.

Codificador seqüencial

A codificação pode ser dividida em uma seqüência de operações, apresentada na figura 15: divisão da imagem em blocos 8 x 8, cálculo dos coeficientes da DCT, quantização, reordenação dos coeficientes em zig-zag e codificação baseada em entropia.

DCT (Transformada Discreta de Cossenos)

A imagem é dividida em blocos não sobrepostos de 8 x 8 pixels. Cada um dos 64 elementos de um bloco apresentará um valor no intervalo [0, $2^p - 1$]. Estes valores são deslocados para o intervalo [- (2^{p-1}) , $(2^{p-1}) - 1$]. Numa imagem monocromática na qual $p = 8$, por exemplo, os valores no intervalo [0, 255] seriam deslocados para [-128, 127].

A DCT codificará estes valores, transportando-os do domínio espacial para o domínio de freqüências segundo a equação:

$$F(u, v) = \frac{1}{4} C(u) C(v) \sum_{x=0}^7 \sum_{y=0}^7 f(x, y) \cos \frac{\pi u(2x+1)}{16} \cos \frac{\pi v(2y+1)}{16} \quad (6.41)$$

onde

$$C(u), C(v) = \frac{1}{\sqrt{2}} \text{ para } u, v = 0 \quad (6.42)$$

$$C(u), C(v) = 0 \text{ para } u, v > 0 \quad (6.43)$$

Dos valores resultantes, $F(0,0)$ é chamado coeficiente DC e os demais 63 valores são denominados coeficientes AC.

Numa imagem típica, muitos dos coeficientes terão valor zero ou próximo de zero. Estes componentes serão descartados no processo de compactação de dados.

Note-se que a DCT em si não compacta os dados. A análise dos valores resultantes da DCT é que permite escolher os dados que podem ser descartados sem perda visível de qualidade da imagem.

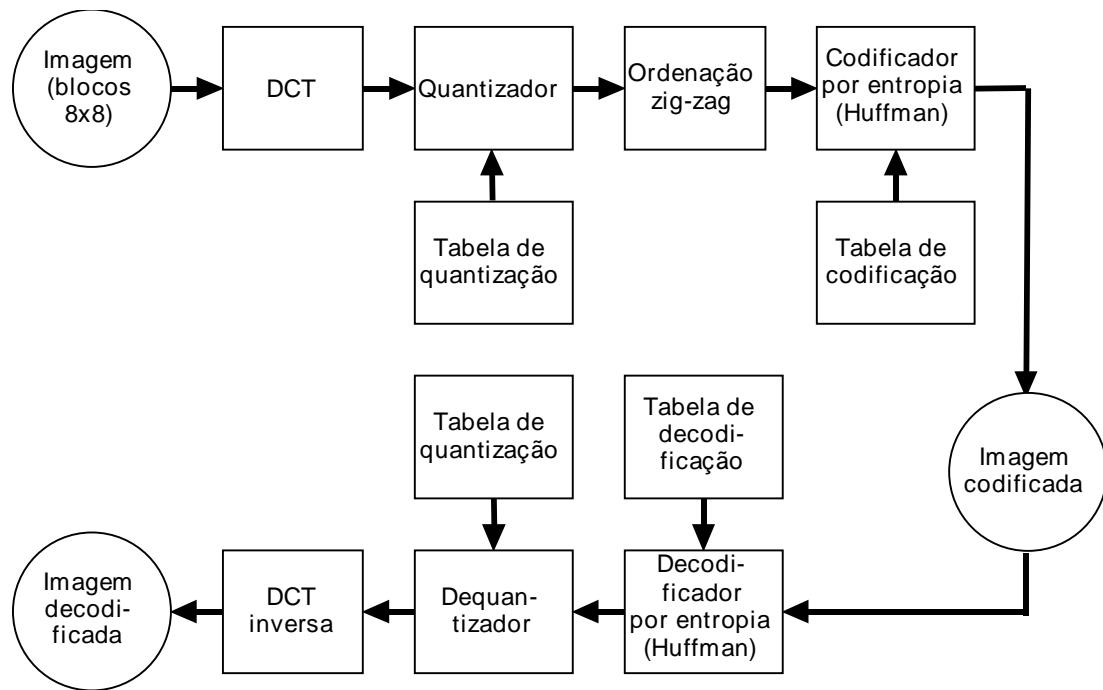


Figura 15 - Esquema básico de codificação e decodificação JPEG seqüencial.

Quantização

Esta etapa aumenta o número de coeficientes com valor zero, valendo-se da redução da amplitude dos coeficientes que contribuem pouco para a qualidade de imagem. A quantização também descarta informação que não é visualmente significativa.

Os coeficientes resultantes da etapa de DCT são transformados de acordo com a fórmula:

$$F_{Q}(u,v) = \text{round} \left[\frac{F(u,v)}{Q(u,v)} \right] \quad (6.44)$$

onde os valores $Q(u,v)$ constituem a tabela de quantização. Cada um dos elementos $Q(u,v)$ é um número inteiro no intervalo 1 a 255. O padrão permite até quatro tabelas de quantização.

Ordenação zig-zag

Os 63 coeficientes AC são reordenados numa seqüência zig-zag (figura 16(a)) visando facilitar a etapa de codificação por entropia. A reordenação coloca os coeficientes de baixa freqüência, que têm maior probabilidade de serem diferentes de zero, antes dos coeficientes de alta freqüência. Já os coeficientes DC, que representam os valores médios dos blocos de 64 pixels, são codificados por meio de técnicas preditivas (figura 16(b)) devido à forte correlação existente entre os coeficientes DC de blocos adjacentes.

Codificador por entropia

O JPEG especifica dois métodos que podem ser usados para esta etapa de codificação: algoritmo de Huffman ou codificação aritmética. Descrevemos a seguir o funcionamento do codificador Huffman.

Na saída do ordenador zig-zag encontram-se muitos coeficientes com valor zero devido às etapas de DCT+quantização. Seqüências de coeficientes com valor zero seguida de um coeficiente com valor diferente de zero são codificadas em *run-length* segundo o formato chamado símbolo intermediário:

Quantidade de coeficientes com valor 0	Número de bits necessários para codificar o coeficiente diferente de 0	Valor do coeficiente diferente de 0
<i>Run-length</i>	<i>Size</i>	<i>Amplitude</i>

Por exemplo, a seqüência de coeficientes 0, 0, 0, 0, 0, 0, 0, 476 será codificada como (6,9) (476). Onde 6 é a quantidade de zeros consecutivos, 476 o valor do coeficiente diferente de zero que segue a seqüência de zeros, 9 é o número de bits necessário para codificar 476.

O termo *Run-length* usa 4 bits de maneira que podemos representar seqüências de até 15 zeros. Caso existam mais do que 15 zeros usamos o símbolo intermediário (15,0) como indicador de 16 zeros. Podemos ter até três (15,0) consecutivos. Por exemplo a seqüência de símbolos intermediários (15,0) (15,0) (7,4) (12) corresponde a uma seqüência de $16+16+7 = 39$ coeficientes zero seguido de um coeficiente igual a 12.

O termo *Size* usa 4 bits para representar valores de 0 a 10. 10 é o número máximo de bits necessário para representar o termo *Amplitude* cujo valor está no intervalo [-1024, 1023]. O símbolo (0,0) significa fim do bloco 8x8 (EOB, *end of block*).

Já os coeficientes DC são codificados na forma:

Número de bits necessário para codificar o coeficiente DC	Valor do coeficiente DC
---	-------------------------

Cada símbolo intermediário é então transformado numa seqüência binária de comprimento variável de acordo com o algoritmo de Huffman: aos símbolos com maior probabilidade de ocorrência são atribuídas seqüências binárias mais curtas e aos de menor probabilidade atribuem-se seqüências binárias mais longas. As tabelas de Huffman para a codificação seqüencial JPEG são encontradas em [Pennenbaker e Mitchell 1993].

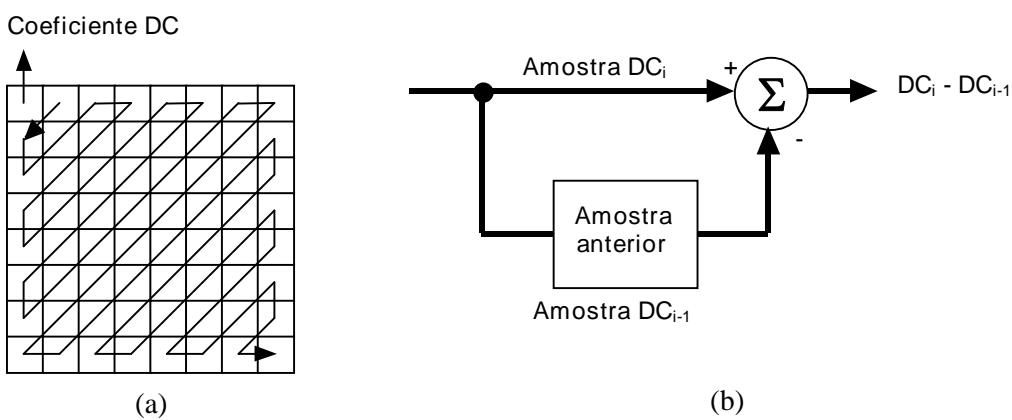


Figura 16 - (a) Ordenação zig-zag; (b) codificador preditivo dos coeficientes DC.

Decodificador seqüencial

A seqüência binária da imagem codificada é primeiro convertida em coeficientes DCT através do decodificador por entropia. Estes coeficientes são então dequantizados de acordo com a fórmula:

$$F(u, v) = Fq(u, v)Q(u, v) \quad (6.45)$$

Os coeficientes dequantizados são transportados do domínio da freqüência para o domínio espacial através da DCT inversa:

$$f(x, y) = \frac{1}{4} \sum_{u=0}^7 \sum_{v=0}^7 C(u)C(v)F(u, v) \cos \frac{\pi u(2x+1)}{16} \cos \frac{\pi v(2y+1)}{16} \quad (6.46)$$

onde

$$C(u), C(v) = \frac{1}{\sqrt{2}} \quad \text{para } u, v = 0 \quad (6.47)$$

$$C(u), C(v) = 1 \quad \text{para } u, v > 0 \quad (6.48)$$

Após a DCT inversa, os valores decodificados são deslocados para o intervalo $[0, 2^P - 1]$.

Compressão progressiva

Em algumas aplicações o tamanho da imagem e/ou a velocidade do canal de transmissão pode tornar o processo de codificação-transmissão-decodificação muito lento. Pode então ser vantajoso, ou mesmo necessário, ter acesso rapidamente à imagem completa mesmo que ela seja de baixa qualidade. No modo progressivo a codificação JPEG é feita por uma seqüência de varreduras da imagem, onde cada varredura gera um subconjunto dos coeficientes DCT. Já na primeira varredura temos a imagem completa, de menor qualidade, mas em menor tempo que no modo seqüencial.

A operação pode ser executada por seleção espectral, aproximação sucessiva ou por uma combinação de ambas. Na seleção espectral os coeficientes da DCT são agrupados em diversas faixas espetrais: coeficientes que representam baixas freqüências são enviados primeiro. Por exemplo, para uma seqüência de 4 faixas espetrais podemos ter:

Faixa 1: coeficientes DC;

Faixa 2: coeficientes AC₁ e AC₂;

Faixa 3: coeficientes AC₃...AC₆;

Faixa 4: coeficientes AC₇...AC₆₃

Na técnica de aproximação sucessiva todos os coeficientes da DCT são enviados inicialmente com baixa precisão e, posteriormente, os valores são refinados a cada nova varredura.

Codificação seqüencial sem perdas

A compressão sem perdas é baseada em codificação preditiva atingindo taxas de cerca de 2:1. Em lugar de se codificar o valor da amostra, codifica-se a diferença entre o valor real e o valor previsto através de algoritmos por entropia.

Considerando-se que o pixel X tem como vizinhos acima e à esquerda os pixels A, B e C:

C	B
A	X

o valor previsto pode ser calculado por uma das seguintes fórmulas:

0	sem predição
1	X = A
2	X = B
3	X = C

4	$X = A + B - C$
5	$X = A + (B - C)/2$
6	$X = B + (A - C)/2$
7	$X = (A + B)/2$

Outros aspectos do JPEG

JPEG x GIF

O JPEG não substitui o GIF. O GIF é superior ao JPEG quando se trata de codificar imagens com poucas cores ou grandes áreas com o mesmo valor de pixel. O JPEG, ao contrário do GIF, distorce imagens que contenham bordas bem definidas, isto é, transições abruptas nos valores dos pixels (o exemplo extremo são caracteres pretos sobre fundo branco). Nestes casos a imagem JPEG apresenta-se borrada. O bom desempenho do JPEG é obtido com imagens *full-color* (até 8 bits/amostra) de tom contínuo, sem transições bruscas de cores. Já o GIF é limitado a um mapa de cores (palheta) de 256 cores. Informações adicionais sobre os formatos GIF e JPG são fornecidas no Apêndice A.

Parâmetros de qualidade

Os compressores JPEG permitem trabalhar com os parâmetros qualidade de imagem x tamanho de arquivo através da seleção de um nível de qualidade. No entanto, os níveis de qualidade não são padronizados:

- a Apple costumava usar uma escala de 0 a 4, tendo alterado para uma escala 0-100 nos softwares mais recentes;
- o Paint Shop Pro usa uma escala de 100 invertida, isto é, quanto menor o valor numérico, maior a qualidade da imagem;
- o Adobe Photoshop permite escolher entre os níveis *high*, *medium* e *low*.

Desta forma, dizer que um arquivo JPEG tem 'qualidade 75' não tem significado a menos que se especifique também o software utilizado na compressão. Esta falta de padronização, no entanto, não impede o intercâmbio de arquivos JPEG.

Note-se que mesmo ajustando o nível de qualidade para o valor máximo, a compressão será com perdas. O algoritmo para JPEG sem perdas é completamente diferente do JPEG normal, tendo como característica principal não usar a DCT.

Pixel transparente

Alguns tipos de arquivos de imagem, como o GIF, permitem escolher um valor de pixel não usado como pixel transparente (sem cor). No JPEG, devido às perdas inerentes do algoritmo, isto não é possível: um pixel não tem necessariamente o mesmo valor inicial uma vez que pequenos erros são permitidos como parte do processo de compressão.

Acumulação de perdas

A finalidade do JPEG é ser um padrão para armazenamento e transmissão de imagens. Para a manipulação de imagens deve-se primeiro converter o arquivo JPEG para algum formato *full-color* sem perdas (e.g. o TIFF), fazer as alterações na imagem e então reconverte-la para JPEG. Deve-se atentar para o fato de que sucessivas conversões JPEG → outro formato → edição → JPEG introduzem perdas que se acumulam a cada nova reconversão.

M-JPEG

Apesar do JPEG ter sido concebido como um padrão de compressão para imagens estáticas, muitos fabricantes aplicaram o JPEG para seqüências de imagens de vídeo tratando cada quadro como uma imagem isolada, dando origem ao que se denomina M-JPEG (*motion* JPEG). Infelizmente, na falta de um padrão estabelecido, cada fabricante implementou a técnica à sua maneira.

O padrão reconhecido para compressão de imagens em movimento é o MPEG (*Moving Pictures Experts Group*), que além de comprimir a imagem de um quadro isolado como o JPEG, vale-se das redundâncias existentes entre quadros sucessivos (redundância temporal ou *interframe*) para obter maiores taxas de compressão. Por este motivo o MPEG tem taxa de compressão cerca de 3 vezes superior ao M-JPEG. A codificação *interframe*, no entanto, dificulta a edição de imagens quadro-a-quadro, motivo este que tornou o M-JPEG muito popular nos equipamentos de edição de vídeo.

Leitura complementar

Para uma descrição completa e formal do padrão JPEG, sugerimos [Pennenbaker e Mitchell 1993] e [Wallace 1991].

O artigo de Furht [Furht 1995a] descreve didaticamente as etapas de codificação e decodificação JPEG e compara o desempenho do algoritmo sobre diferentes imagens.

Aos leitores interessados nos aspectos de custo computacional do algoritmo JPEG, sugerimos [Monnes e Furht 1994].

6.6.3 H.261

Estabelecido em 1991, o H.261 faz parte de um conjunto de padrões do ITU-T para serviços audiovisuais em telecomunicações. Além do H.261, que trata da codificação/decodificação de sinais de vídeo, temos o H.221 (estrutura de quadros), H.230 (controle de sincronismo de quadro), H.242 (comunicação entre terminais audiovisuais) e H.320 (equipamentos de sistema e terminais). Codificadores/decodificadores de áudio são especificados por outros padrões (o G.725, por exemplo).

Características do H.261

O padrão também é conhecido como $p \times 64$ porque pode operar nas diversas capacidades de um canal ISDN (*Integrated Service Digital Network*) (taxas de $p \times 64$ kbps, para $p = 1, \dots, 30$). Para $p = 1$ e 2, a limitada largura de banda disponível, permite somente comunicação face-a-face (videofone). Para $p \geq 6$ imagens mais complexas podem ser transmitidas permitindo aplicações de videoconferência. Os formatos de imagem permitidos são CIF (*Common Intermediate Format*) e QCIF (*Quarter CIF*).

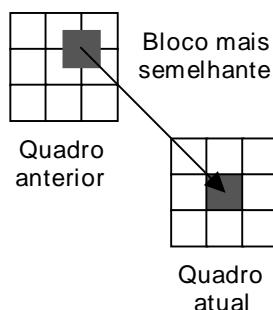


Figura 17 - Princípio da codificação *interframe*

O H.261 tem muitas semelhanças com o JPEG: ambos utilizam as técnicas de dividir a imagem em pequenos blocos e nelas aplicar a DCT, quantização e codificação por comprimento variável. A maior diferença é que o JPEG codifica cada imagem (ou quadro) individualmente (codificação *intraframe*) enquanto que o H.261, além da codificação *intraframe*, usa codificação *interframe* (figura 17): o quadro anterior é usado para prever o quadro atual. Como resultado, somente a diferença entre eles (quadro anterior e previsto), geralmente de pequena magnitude, é transmitida. Com esta técnica as taxas de compressão atingidas variam entre 100:1 a 2000:1.

Entretanto, as técnicas de predição de movimento são utilizadas de forma limitada: só o quadro imediatamente anterior é considerado. A razão desta limitação fundamenta-se no fato de que o padrão destina-se à comunicação em tempo real onde é necessário reduzir o atraso de

processamento. Também deve-se levar em conta que as aplicações do H.261, videofone e videoconferência, não são intensivas em movimento.

O tipo de aplicação também dirigiu o padrão para um equilíbrio entre a complexidade do codificador e do decodificador, uma vez que ambos são necessários para comunicação em tempo real. A estrutura de codificação e seus parâmetros foram escolhidos visando aplicações com baixa taxa de dados.

O H.261 é a especificação de um conjunto de protocolos que todo fluxo de dados (*bitstream*) codificado tem que seguir e também um conjunto de operações que todo decodificador compatível com o padrão tem que ser capaz de executar.

Estrutura de dados

O H.261 tem um fluxo de dados com a seguinte estrutura hierárquica: Imagens, Grupo de Blocos (GOB), Macro Blocos (MB) e Blocos. Um MB (macro bloco) é composto de 4 blocos 8 x 8 de luminância (Y) e dois blocos 8 x 8 de crominância (Cr e Cb). Um GOB (grupo de blocos) é composto de 3 x 11 MBs. Uma imagem CIF contém 12 GOBs e uma imagem QCIF, 3 GOBs. Estes conceitos estão ilustrados na figura 18.

O cabeçalho da camada imagem contém:

- PSC (*picture start code*): 20 bits;
- TR (*temporal reference*): 5 bits, número do quadro entrante;
- PTYPE (*type information*): CIF ou QCIF;
- bits do usuário;

segue um número de GOB.

O cabeçalho da camada GOB contém:

- GBSC (*group of blocks start code*): 16 bits;
- GN (*group number*): 4 bits, endereço do GOB;
- GQUANT (*quantizer information*): tamanho do passo do quantizador (entre 1 e 31);
- bits do usuário;

segue um número de MB.

O cabeçalho do MB contém:

- MBA (*macroblock address*): MB previamente codificado;
- MTYPE (*type information*): existem 10 tipos;
- MQUANT (*quantizer*): passo de quantização normalizado;
- MVD (*motion vector data*);
- CBP (*coded block pattern*): indica a localização do bloco codificado;

No bloco temos:

- coeficientes da transformada quantizados;
- EOB (*end of block*);

O tipos de MB são essencialmente quatro: intra, inter, inter com compensação de movimento e inter com compensação de movimento com filtro.

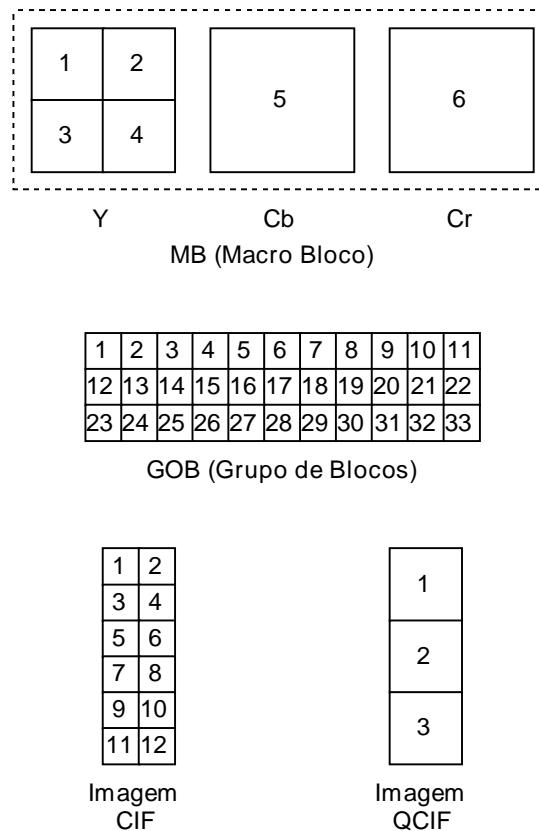


Figura 18 - Estrutura hierárquica dos dados H.261.

Codificador

O algoritmo de codificação H.261 (figura 19) consiste em:

- compressão *intraframe* baseada em DCT;
- predição *interframe* baseado em DPCM e compensação de movimento.

O algoritmo inicia com uma codificação *intraframe* usando DCT e quantização e enviando o resultado para o multiplex de vídeo. O novo quadro é descompactado via dequantizador e DCT inversa, armazenado na memória de quadro para uso na codificação *interframe*. Na codificação *interframe* a predição baseada no algoritmo DPCM é usada para comparar cada macro bloco (MB) do quadro atual com os MBs do quadro anterior. As diferenças são calculadas, como valores de erro, codificados via DCT e quantização, enviadas para o multiplex de vídeo com ou sem vetores de movimento. Na etapa final utiliza-se codificação por métodos entrópicos (Huffman, por exemplo).

Decodificador

A figura 20 ilustra esquematicamente o processo de decodificação H.261. Os coeficientes, exceto os coeficientes DC *intraframes*, são recuperados de acordo com uma tabela de dequantização. Os coeficientes DC *intraframe* são uniformemente quantizados com passo fixo de 8 e codificados com 8 bits.

O padrão requer uma DCT inversa próxima da DCT inversa ideal de 64 bits. Para evitar erros de DCT inversa e propagação de erros introduzidos pelo canal, pelo menos 1 em cada 132 quadros tem que ser codificado *intraframe*.

Leitura complementar

Maiores detalhes sobre o H.261 podem ser vistos em [Aranvid et al. 1993] e [Furht 1995b].

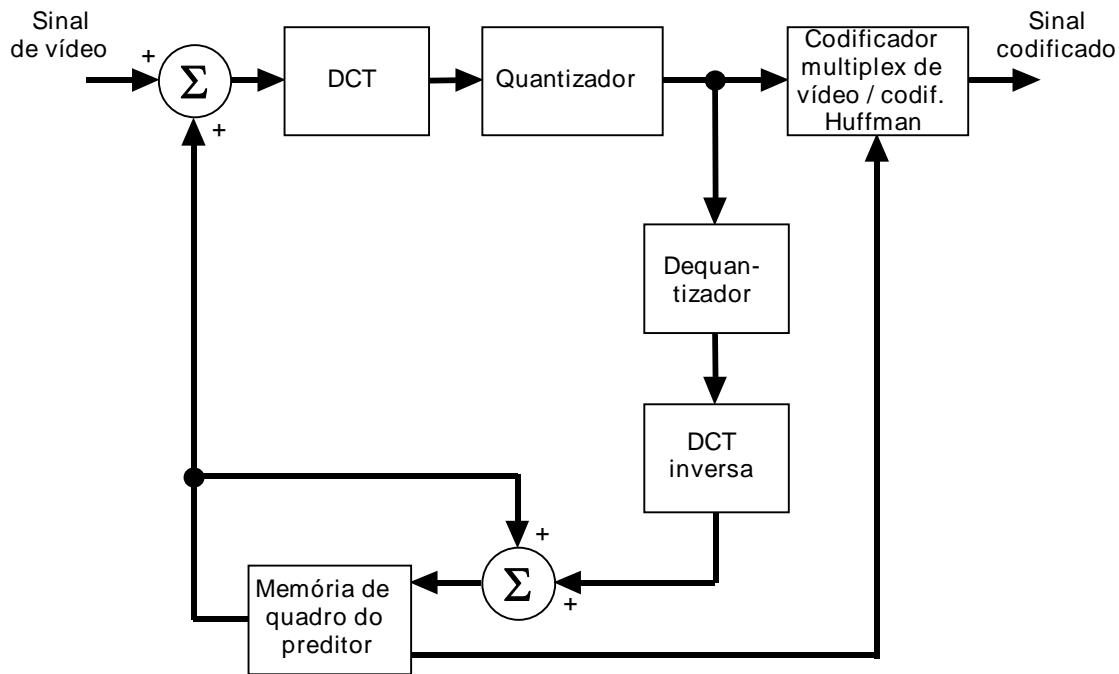


Figura 19 - Codificador H.261.

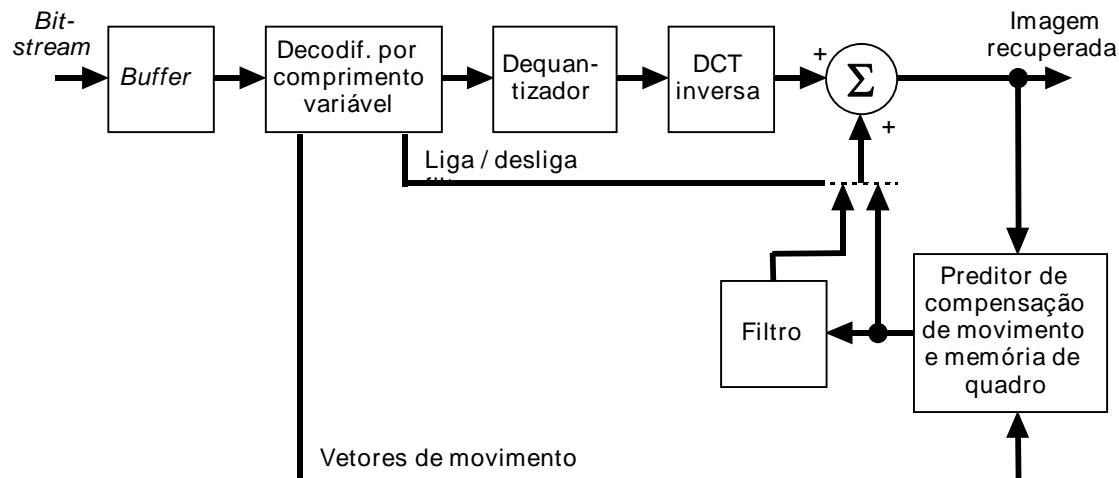


Figura 20 - Decodificador H.261.

6.6.4 H.263

O H.263 é um padrão para comunicação visual recentemente estabelecido pelo ITU-T capaz de operar a baixa taxas de dados. O padrão foi desenvolvido tendo em vista a baixa capacidade da rede telefônica convencional e das comunicações sem fio.

Com os métodos de codificação atuais é possível conseguir razoável qualidade de imagem (para algumas aplicações) a taxas de 64 kbps. Abaixo disto a transmissão só é possível a baixa taxa de quadros além de surgirem distorções na imagem. Tendo em vista estas limitações, o ITU-T desenvolveu um conjunto de especificações que permitem a utilização de canais de baixa capacidade na comunicação multimídia. Para a codificação de vídeo dois algoritmos foram desenvolvidos: H.263 e H.263/L.

O H.263 foi baseado em tecnologia existente em 1995 e desenvolvido para oferecer qualidade de imagem significativamente superior ao H.261. No desenvolvimento do padrão foi considerada a máxima capacidade de transmissão da rede telefônica convencional, à época 28,8

kbps. A taxas de dados tão baixas é necessário manter o mais baixo possível o *overhead* na informação transmitida.

Dentre as características do H.263, destacamos:

- baixa complexidade (e custo);
- interoperabilidade e/ou coexistência com outros padrões de comunicação visual (e.g., H.261);
- robustez quanto a erros introduzidos pelos canais de comunicação;
- flexibilidade para extensões futuras como operação a taxas de dados mais elevadas;
- parâmetros de qualidade de serviços tais como resolução de imagem, atraso de sinal, taxa de quadros, qualidade de cores etc.;
- a exemplo do H.261, a codificação é um processo híbrido de predição de movimento *interframe* com codificação DCT do erro de predição;
- estrutura do GOB mais simples que no H.261;
- o uso da compensação de movimento é opcional no codificador;
- diversos parâmetros podem ser variados para controlar a taxa de dados: processamento do sinal de vídeo prioritário à codificação do sinal da fonte, escala do quantizador, seleção de modos e taxa de quadros. O decodificador pode informar as escolhas feitas quanto a resolução temporal x resolução espacial por meios externos (protocolo H.245);
- quatro métodos de codificação avançados: vetor de movimento irrestrito, predição avançada, quadros P-B e codificação aritmética baseada em sintaxe;
- o codificador pode operar com cinco formatos de imagem: sub-QCIF, QCIF, CIF, 4CIF E 16CIF. Os decodificadores têm que operar, no mínimo, com sub-QCIF e QCIF. Para os codificadores somente um dos formatos, sub-QCIF ou QCIF, é obrigatório.

Leitura complementar

Outros detalhes sobre o H.263 podem ser vistos em [Rijske 1996] e [Herman 1996].

6.6.5 MPEG

O comitê MPEG (*Moving Pictures Experts Group*) foi estabelecido pelo ISO e o IEC em 1988 com o objetivo de desenvolver padrões de codificação de vídeo e áudio associado para armazenamento em mídia digital. O objetivo inicial do MPEG foi estabelecer padrões para a codificação de vídeo (e áudio) a três taxas de dados: 1,5, 10 e 40 Mbps, conhecidos como MPEG-1, 2 e 3, respectivamente.

Tabela 10 - Tipos de MPEG

MPEG	Ano	Aplicação típica	Taxa de dados típica
1	1992	vídeo CD	1,5 Mbps
2	1994	vídeo com qualidade <i>broadcast</i>	4-100 Mbps
4	versão 1 aprovada em outubro de 1998 versão 2 prevista para dezembro de 1999	comunicação multimídia	-
7	previsto para julho de 2001	interface de descrição de conteúdo multimídia	-

O MPEG-1 visava aplicações como o vídeo-CD (armazenamento de vídeo e áudio digital com qualidade similar a do VHS). O MPEG-2 era destinado a aplicações de maior qualidade e resolução (televisão *broadcast*). O MPEG-3 foi abandonado em julho de 1993

quando se verificou que as funcionalidades do MPEG-2 permitiam-no abranger o tipo de aplicação que o MPEG-3 pretendia atingir: a televisão de alta definição (HDTV).

Atualmente o comitê trabalha na definição dos MPEG-4 e 7 que não pretendem se restringir a aplicações de vídeo, mas a padronizar a codificação (MPEG-4) e a descrição do conteúdo de aplicações multimídia (MPEG-7). A tabela 10 resume os vários tipos de MPEG.

Características do MPEG 1 e 2

O MPEG é primariamente uma especificação para a sintaxe que um fluxo de dados (*bitstream*) compatível com o padrão deve seguir. Também é especificado um processo de decodificação típico que auxilia na interpretação da sintaxe do *bitstream*. Esta abordagem permite o intercâmbio de dados mas não restringe inovações, ou a implementação de sistemas proprietários, na criação (codificação) e na decodificação do *bitstream* MPEG. O padrão consiste de três partes: sincronização de áudio e vídeo, vídeo e áudio, ilustrados esquematicamente na figura 21.

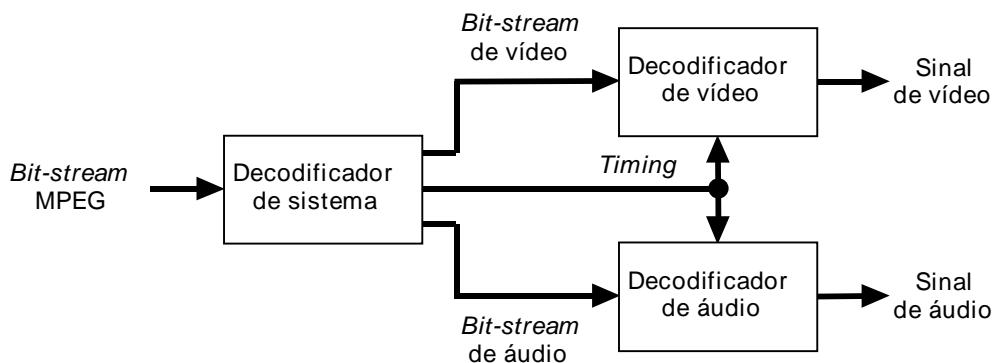


Figura 21 - Esquema genérico da decodificação MPEG.

A compressão de sinais de vídeo pode atingir taxas de até 200:1. Ambos os tipos de aplicações, simétricas e assimétricas, são atendidas pelo MPEG. Aplicações assimétricas caracterizam-se pelo uso frequente da decodificação enquanto que o processo de codificação é realizado uma só vez, como por exemplo o vídeo sob demanda e o ensino à distância. Aplicações simétricas requerem igual uso tanto da codificação quanto da decodificação. As aplicações de tempo real como a videoconferência são exemplos típicos.

Na elaboração do MPEG houve preocupação em se prever suporte para uma série de características tipicamente encontradas em equipamentos de gravação e transmissão de vídeo: acesso aleatório, busca rápida em avanço (*forward search*) e em retrocesso (*reverse search*), playback reverso, sincronismo entre os sinais de áudio e vídeo, capacidade de tratar erros, editabilidade e flexibilidade de formatos.

MPEG-1

Originalmente desenvolvido para estabelecer um padrão para o armazenamento de áudio e vídeo em mídia digital, o MPEG-1 é otimizado para operar a taxas de 1,5 Mbps (padrão do CD de áudio). Tipicamente, o sinal de áudio toma 192 kbps, o sinal de vídeo 1,15 Mbps e o restante é usado para os dados do sistema. Apenas o modo de varredura seqüencial (não entrelaçado) é suportado. Suas aplicações incluem multimídia interativa e vídeo-CD. O formato de imagem típico é de 320 x 240 pixels. Diferentemente dos padrões para videoconferência, o MPEG-1 preocupa-se mais em estabelecer parâmetros para se obter determinados níveis de qualidade do que em conseguir transmitir informação a uma certa taxa de dados.

MPEG-2

É compatível com o MPEG-1 mas inclui extensões para abranger uma maior variedade de aplicações. O MPEG-2 foi concebido inicialmente como um padrão para a transmissão de vídeo digital (a taxas de 4-9 Mbps) com qualidade equivalente à da televisão comercial (*broadcast*). Entretanto o MPEG-2 é eficiente também para outras aplicações a taxas de dados e amostragem

mais altas como a HDTV. O aperfeiçoamento mais significativo em relação ao MPEG-1, e que é essencial para a TV *broadcast*, foi a inclusão da capacidade de codificação de vídeo entrelaçado. Dentre os exemplos de aplicação do MPEG-2, citamos os sistemas de TV a cabo, o consórcio americano para HDTV e o DVD.

Devido à generalidade do MPEG-2, que permite taxas de até 400 Gbps e imagens de até 16000 x 16000 pixels, um sistema de perfis (*profiles*) e níveis (*levels*) foi definido para colocar limites práticos nos muitos parâmetros de uma aplicação real. Um perfil é um subconjunto da sintaxe do *bitstream*. Por exemplo, a sintaxe permite operação escalonável S/R (sinal/ruído) ou espacial, mas os perfis *main* e *simple* não usam este recurso do MPEG-2. Um nível, por sua vez, restringe os parâmetros dentro de uma sintaxe permitida.

Estrutura dos quadros MPEG

São três os tipos de quadros usados pelo MPEG: I (*intraframe*), P (preditivo) e B (bidirecional).

O quadro I é codificado de maneira similar ao JPEG usando a informação de uma única imagem em particular, sem considerar outras imagens, prévias ou futuras. Os quadros I são chamados referências temporais e utilizados como pontos de acesso aleatório dentro do fluxo de dados MPEG. Sua taxa de compressão é a menor dentre todos os tipos de quadros.

O quadro P usa predição, isto é, o quadro atual é codificado com referência a um quadro prévio que pode ser do tipo I ou P. O processo é similar à codificação preditiva do H.261, com a diferença de que o quadro prévio nem sempre é o quadro imediatamente anterior como acontece no H.261. A taxa de compressão do quadro P é significativamente maior que a do quadro I.

Por último, o quadro B é codificado usando-se dois quadros como referência: um quadro anterior (passado) e outro posterior (futuro). Os quadros anterior e posterior podem ser do tipo I ou P. Quadros B proporcionam as taxas de compressão mais altas.

Na figura 22, o quadro P (5) é codificado com referência ao quadro prévio I (1). Já os três primeiros quadros B (2,3,4) são codificados pela combinação de dois quadros de referência: quadro prévio I (1) e quadro futuro P (5). Note-se que os quadros P podem propagar erros por serem obtidos com referência a um quadro prévio e servirem também como referência aos quadros B.

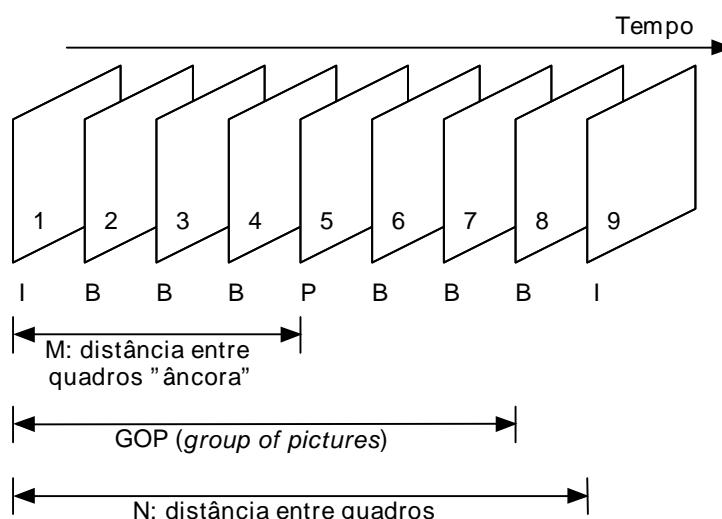


Figura 22 - Seqüência de quadros MPEG.

Devido à existência dos quadros B, a ordem de decodificação diferirá da ordem de codificação: o quadro P (5) tem que ser decodificado antes dos quadros B (2, 3 e 4); o quadro I (9) tem que ser decodificado antes dos quadros B (6, 7 e 8). A seqüência de transmissão, para uma correta decodificação, seria, portanto, {1, 5, 2, 3, 4, 9, 6, 7, 8}.

A aplicação é que determinará os tipos e a seqüência de quadros utilizada. Por exemplo, se houver necessidade de acesso aleatório rápido, a solução mais simples seria codificar toda a

seqüência de vídeo com quadros I (caso em que o MPEG ficaria idêntico ao M-JPEG). A seguinte seqüência provou ser eficaz para um grande número de aplicações práticas:

(I B B P B B P B B) (I B B P B B P B B) ...

Para sistemas operando a 30 quadros/s, quadros I são enviados a cada 400 ms aproximadamente (um quadro I a cada 10 a 12 quadros). Os quadros I além de permitirem o acesso aleatório no fluxo MPEG, conforme especificado, também garantem a qualidade da imagem porque os quadros P e B são baseados neles. Portanto, é importante que os quadros I sejam transmitidos com maior confiabilidade que quadros P ou B.

Os quadros B tornam a imagem mais suave e consomem menos largura de banda. O problema é que para utilizá-los o decodificador necessita armazenar quadros P para calculá-los, elevando a complexidade e o custo do sistema.

Ainda com relação à figura 22, definimos um GOP (*group of pictures*) como uma seqüência que se inicia com um quadro I e se estende até o quadro imediatamente anterior ao próximo quadro I. O GOP mostrado na figura é dito aberto: o último quadro do GOP usa o primeiro quadro do próximo GOP como referência. O segundo tipo de GOP é o fechado, onde um quadro P fecha o grupo e, portanto, não tem vínculos com o próximo GOP. A figura 23 apresenta a divisão do GOP em sucessivas unidades menores até chegarmos ao elemento básico da imagem MPEG, o bloco.

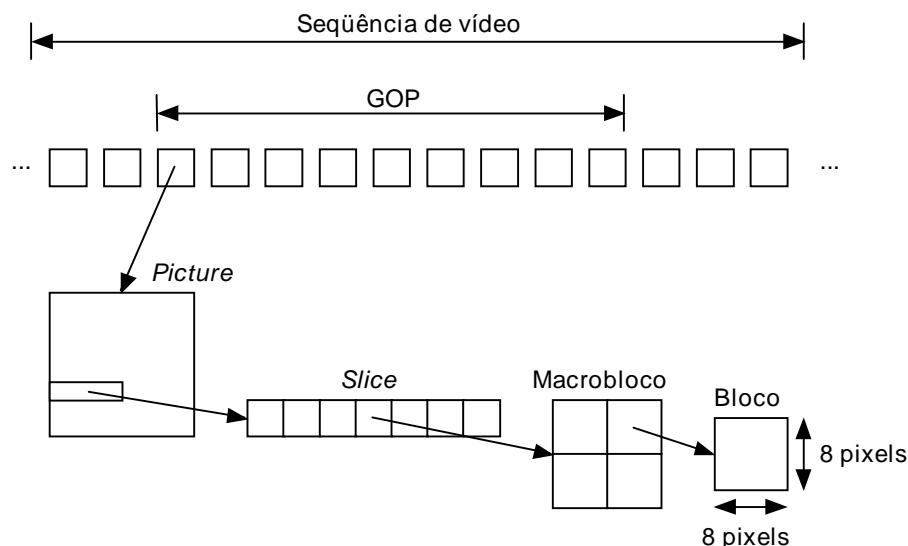


Figura 23 - Estrutura hierárquica da seqüência de imagens MPEG.

Codificação *interframe*

O princípio da codificação *interframe* é muito simples: numa seqüência de imagens de vídeo há uma grande probabilidade de que o quadro atual seja muito semelhante ao anterior e que o quadro futuro também se assemelhe ao atual. Codificando apenas a diferença existente entre os quadros atual e prévio, a quantidade de informação a ser transmitida/armazenada seria menor do que codificando-se cada quadro isoladamente. A técnica pode ser refinada se em lugar de utilizarmos o quadro prévio diretamente, levarmos em conta o movimento dos objetos na cena que pode haver entre um quadro e outro. O mecanismo da codificação interframe funciona como se fosse criado um novo quadro a partir do prévio onde os objetos se movimentaram de acordo com uma previsão (estimativa) de movimento (*motion estimation*). A diferença entre este quadro hipotético transformado e o atual é ainda menor que a diferença dele em relação ao quadro atual.

A figura 24 ilustra a técnica. Neste exemplo, imaginemos que o quadro prévio já foi codificado (e transmitido/armazenado). Não é necessário codificar o quadro atual, mas apenas transmitir a informação sobre o movimento do objeto ocorrido de um quadro a outro (vetor de movimento) além da informação nova existente no quadro atual (erro de predição).

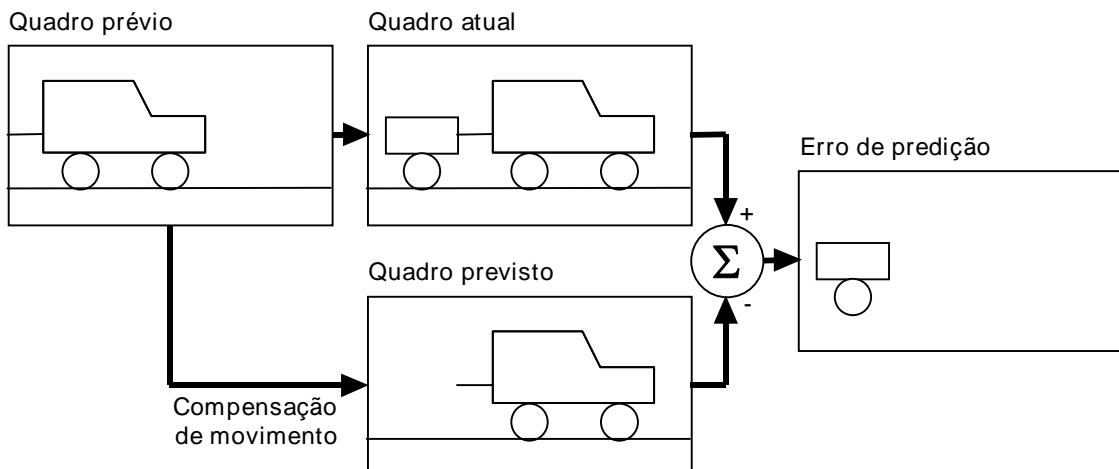


Figura 24 - Ilustração da previsão de movimento entre quadros.

Apesar de conceitualmente simples, a implementação desta técnica apresenta duas dificuldades práticas:

- é necessário identificar objetos com formatos diversificados num quadro;
- é preciso determinar o deslocamento do objeto ocorrido de um quadro a outro.

O MPEG não especifica nenhuma técnica para a predição de movimento. No entanto, é prática comum utilizar-se a técnica de semelhança entre blocos (*block matching*), que faz algumas simplificações em relação ao esquema anteriormente descrito:

- não são identificados objetos reais na imagem. A previsão de movimento é feita sobre os macroblocos (16 x 16 pixels), como se cada um deles fosse um objeto;
- a busca é limitada a uma área de 28 x 28 pixels.

Se o macrobloco M é encontrado na mesma posição relativa em S, o vetor de movimento é zero; caso contrário, um vetor de movimento diferente de zero é codificado em lugar do macrobloco. Os vetores de movimento são obtidos pela minimização de uma função de custo. Furht [Furht 1995b] apresenta as funções de custo mais conhecidas na literatura e detalha alguns algoritmos para a minimização destas funções de custo. O artigo de [Pirsch et al. 1995] considera a implementação destas técnicas em circuitos VLSI.

Quadros P sempre usam a predição em avanço (*forward prediction*), isto é, a área de busca do macrobloco fica num quadro posterior ao atual. Já os quadros B podem usar um quadro anterior (*backward prediction*) ou posterior (*forward prediction*). Neste caso tanto o quadro P como o B geram um vetor de movimento para cada macrobloco. O quadro B pode também usar dois quadros de referência, um anterior e outro posterior, gerando dois vetores de movimento.

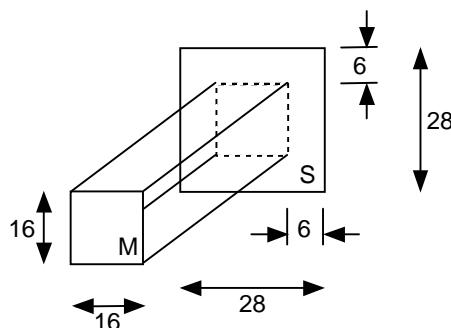


Figura 25 - Predição de movimento: área de busca do macrobloco.

MPEG-4

Iniciado em julho de 1993, originalmente o MPEG-4 tinha como objetivo estabelecer um padrão para videoconferência/videofone a baixas taxas de dados. Devido, em parte, ao estabelecimento do H.324 (H.263) pelo ITU-T como padrão para este tipo de aplicação, a abrangência do MPEG-4 foi muito ampliada. A primeira versão do MPEG-4 foi aprovada em outubro de 1998 e a segunda versão está prevista para dezembro de 1999..

O objetivo do MPEG-4 é ser um padrão para codificar diferentes formas de dados (objetos audiovisuais de origem natural ou sintetizada) provendo meios para representar, integrar e intercambiar estes objetos. O padrão deverá oferecer:

- um novo tipo de interatividade com objetos dinâmicos: peças individuais de informação dentro de uma cena chamados objetos AV (audiovisuais);
- integração de material audiovisual natural (filmes, fotos, etc) e sintetizado (gráficos, animação gerados por computador, etc);
- possibilidade de o usuário interferir no modo como o material audiovisual será apresentado (capacidade de composição de uma cena);
- acesso aleatório mais eficiente aos componentes de uma sequência audiovisual;
- qualidade audiovisual subjetivamente superior aos outros padrões de codificação a taxas de dados comparáveis;
- habilidade em codificar múltiplas vistas/trilhas sonoras de uma cena explorando a redundância entre as diferentes vistas e com suficiente sincronismo entre elas;
- reusabilidade de ferramentas e dados;
- independência da aplicação em relação às camadas de mais baixo nível (uso de API - *application interface*);
- capacidade de os receptores realizarem *downloading* dos softwares de aplicação;
- uso simultâneo de material vindo de diferentes fontes;
- integração de informação de tempo real com informação armazenada em uma apresentação;
- robustez a erros mesmo na transmissão em canais ruidosos e de baixa capacidade;
- compatibilidade com MPEG-1 e 2.

MPEG-7

O crescimento da disponibilidade de informação audiovisual distribuída por diversos locais em todo o mundo torna cada vez mais difícil encontrar a informação desejada. Atualmente existem maneiras de se encontrar na WWW informação sobre um determinado assunto através de consulta no modo textual. Não existe, no entanto, nenhuma maneira eficiente de se encontrar informação visual através da descrição de seu conteúdo.

O MPEG-7 (interface para descrição de conteúdo multimídia) é o padrão MPEG que visa solucionar este problema. O padrão especificará uma descrição padronizada dos vários tipos de informação multimídia (imagens estáticas, gráficos, áudio, vídeo, e informação sobre como estes elementos são combinados numa apresentação). Esta descrição será associada ao conteúdo multimídia para permitir acesso rápido e eficiente ao material e poderá ser feita em vários níveis semânticos. O nível de abstração mais baixo pode ser a descrição do formato, tamanho, textura, cor e composição; o nível mais alto uma descrição textual do tipo 'esta cena contém tais e tais elementos...'.

Alguns exemplos de uso da interface incluem:

- tocar algumas notas num teclado musical e obter uma lista de peças musicais contendo a melodia produzida ou algo próximo dela;
- desenhar algumas linhas na tela e obter um conjunto de imagens contendo símbolos, logotipos, ideogramas, etc, que se assemelhem ao que foi desenhado;
- digitar algumas palavras e obter uma lista de textos que contenham as mesmas palavras e/ou traduções correlacionadas.

O estabelecimento deste padrão proporcionará utilização mais eficiente dos sistemas multimídia atuais e abrirá novas possibilidades de uso beneficiando aplicações como bibliotecas digitais, educação, medicina, entretenimento, etc. Espera-se que o padrão esteja concluído até julho de 2001.

Leitura complementar

Para uma visão abrangente e introdutória dos padrões e técnicas de compressão de imagens estáticas e vídeo, incluindo um maior detalhamento do MPEG-4, sugerimos [Kawano 1997].

Exercícios Propostos

1. Qual a diferença fundamental entre as técnicas de compressão de imagens com perdas (*lossy*) e sem perdas (*lossless*) do ponto de vista da exploração da redundância?
2. Considere a imagem a seguir, representada por uma matriz 7×7 , onde cada elemento da matriz corresponde ao nível de cinza normalizado do pixel correspondente, sendo 0 = preto, 1 = branco. Pede-se:
 - a) Calcular as probabilidades de cada nível de cinza e esboçar seu histograma.
 - b) Calcular a entropia da fonte (em bits / pixel).
 - c) Codificar cada nível de cinza utilizando Huffman, indicando as palavras-código obtidas na coluna apropriada.
 - d) Calcular o comprimento médio das palavras-código atribuídas no passo (c) e a eficiência do código de Huffman utilizado.
 - e) Calcular a taxa de compressão obtida em relação à utilização de um código de comprimento fixo e igual a 3 bits e a redundância relativa correspondente.

0	3/7	2/7	2/7	1/7	1/7	4/7
3/7	2/7	1/7	1/7	1/7	1/7	4/7
2/7	0	1	1/7	3/7	0	0
0	5/7	1/7	0	6/7	0	1/7
1/7	1/7	1/7	3/7	6/7	6/7	5/7
1/7	1/7	1/7	1/7	5/7	6/7	4/7
0	1	0	0	0	0	4/7

Nível de cinza (r_k)	$p_r(r_k)$	Código de Huffman	$l(r_k)$
$r_0 = 0$			
$r_1 = 1/7$			
$r_2 = 2/7$			
$r_3 = 3/7$			
$r_4 = 4/7$			
$r_5 = 5/7$			
$r_6 = 6/7$			
$r_7 = 1$			

3. Seja a codificação de canal usando Hamming exemplificada na tabela 2. Decodifique a palavra-código original e determine se há algum bit errado (e, em caso positivo, qual é ele), caso a palavra-código recebida seja:

- a) 1100111
- b) 1100110
- c) 1100010.

4. Calcule todas as (16) probabilidades (não se esqueça das probabilidades condicionais e conjuntas) associadas ao canal de informação em que $A = \{0, 1\}$, $B = \{0, 1\}$, $z = [0,75, 0,25]^T$ e

$$Q = \begin{bmatrix} 2/3 & 1/3 \\ 1/10 & 9/10 \end{bmatrix}.$$

5. Decodificar a mensagem 0.23355, codificada aritmeticamente segundo o modelo a seguir. Interpretar o símbolo '!' como EOM (fim de mensagem).

Símbolo	Probabilidade
<i>a</i>	0.2
<i>e</i>	0.3
<i>i</i>	0.1
<i>o</i>	0.2
<i>u</i>	0.1
!	0.1

6. Uma imagem binarizada de 64 x 64 pixels foi codificada usando o código WBS 1-D utilizando blocos de quatro pixels. O código WBS obtido para uma linha da imagem foi: 0110010000001000010010000000, onde 0 significa pixel preto. Pede-se:

- a) Decodificar a linha.
- b) Elaborar um procedimento WBS 1-D iterativo que começa buscando linhas totalmente brancas (um bloco de 64 pixels) e reduz o tamanho dos blocos que contêm um ou mais pixels pretos à metade, sucessivamente, até chegar a blocos de quatro pixels.
- c) Usar o algoritmo elaborado no passo (b) para codificar a linha decodificada previamente. O resultado deste item ocupa mais ou menos bits que o padrão mencionado no enunciado? Por que?

7. Utilize o algoritmo de compressão CCITT Grupo 3 (e 4) para codificar a segunda linha do segmento de duas linhas abaixo:

```
0 1 1 0 0 1 1 1 0 0 1 1 1 1 1 1 1 1 0 0 0 0 1
1 1 1 1 1 1 0 0 0 1 1 1 0 0 0 0 1 1 1 1 1 1
```

Assumir que o elemento de referência inicial a_0 está localizado no primeiro pixel da segunda linha.

8. Por que os coeficientes resultantes do cálculo da DCT direta no algoritmo JPEG, após a quantização, são lidos em zig-zag?

9. O que são e em que se baseiam as técnicas preditivas de compressão de imagens? Qual a diferença básica entre técnicas preditivas com perdas e sem perdas?

No computador

Sugerimos executar o programa *dctdemo*, que acompanha a *toolbox* de processamento de imagens do MATLAB. Este programa permite comprovar interativamente a influência do número de coeficientes da DCT utilizados na reconstrução de uma imagem na sua qualidade subjetiva.

Na Internet

Existe uma quantidade imensa de *sites* direta ou indiretamente ligados à codificação e compressão de imagens. Dentre eles, relacionamos:

"<http://www.mpeg.org/index.html>"

MPEG . ORG - MPEG Pointers and Resources

Ponto de partida ideal para a área de compressão de imagens e vídeo, particularmente o padrão MPEG. Contém *links* para novidades, programas, empresas, FAQs etc. ligados ao assunto. Um *bookmark* obrigatório.

"<http://www.vol.it/MPEG/>"

MPEG Moving Picture Expert Group Information

Informações sobre o padrão MPEG

"<http://www.visiblelight.com/mpeg/index.htm>"

MPEG Plaza - The Source For MPEG

Contém informações sobre produtos, empresas, software e dados técnicos sobre o padrão MPEG, agrupados de forma bem estruturada.

"<http://www.cis.ohio-state.edu/hypertext/faq/usenet/jpeg-faq/top.html>"

JPEG image compression: Frequently Asked Questions

Respostas a questões mais comuns sobre JPEG.

"<http://www.khoral.com/dipcourse/dip17sep97/html-dip/c4/s12/front-page.html>"

Bit Plane Slicing

Descreve e exemplifica o processo de converter uma imagem monocromática de 8 bits/pixel em oito imagens binárias, cada qual correspondendo a um bit do byte original.

"<http://www.deakin.edu.au/~agoodman/scc308/topic7.html>"

Topic 7: File formats and image compression

Capítulo de tutorial *on-line* dedicado a formatos de arquivos de imagem e algumas técnicas de compressão utilizadas nestes formatos.

"<http://www.engr.mun.ca/~john/btpc.html>"

Binary Tree Predictive Coding

Descrição completa e bem documentada de uma nova proposta de algoritmo de codificação de imagens estáticas, com vantagens em relação ao LZW e ao JPEG.

"<http://act.by.net/act.html>"

Archive Comparison Test (A.C.T.)

Comparativo de desempenho de programas de compressão de dados para diversas plataformas. Atualizado periodicamente.

"<http://www.internz.com/compression-pointers.html>"

Compression Pointers

Lista de *links* interessantes e úteis na área de compressão de dados e imagens.

"<http://drogo.cselt.stet.it/mpeg/>"

The Moving Picture Experts Group (MPEG) Home Page

Página oficial do comitê encarregado da padronização MPEG.

Bibliografia

- [Abramson 1963] Abramson, N., *Information Theory and Coding*, McGraw-Hill, 1963.
- [Allens et al. 1980] Alens, N. et al., "Tópicos sobre fac-símile", Relatório Técnico RT-73, Contrato Telebrás 139/76, Unicamp, Junho 1980.
- [Aranvid et al. 1993] Aranvid, R. et al., "Image and video coding standards", *AT&T Technical Journal*, Jan/Fev 1993, 67-89.
- [Arps 1980] Arps, R.B., "Bibliography on Binary Image Compression", *Proceedings of the IEEE*, 68, 7, Julho 1980, 922-924.
- [Barnsley e Sloan 1988] Barnsley, M.F. e Sloan, A.D., "A better way to compress images", *Byte*, Janeiro 1988, 215-223.
- [Bisignani 1966] Bisignani, W.T., Richards, G.P. e Whelan, J.W., "The Improved Grey Scale and Coarse-Fine PCM Systems: Two New Digital TV Bandwidth Reduction Techniques", *Proc. IEEE*, 54, 3, 376-390.
- [Cutler 1952] Cutler, C.C., *Differential Quantization of Communication Signals*, U.S. Patent 2.605.361, Julho 1952.
- [Furht 1995a] Furht, B., "A survey of multimedia compression techniques and standards - part I: JPEG standard.", *Real-Time Imaging Journal*, 1, 1, 1995, pp. 49-67.
- [Furht 1995b] Furht, B., "A survey of multimedia compression techniques and standards - part II: video compression.", *Real-Time Imaging Journal*, 1, 5, 1995, pp. 319-337.
- [Hamming 1950] Hamming, R.W., "Error Detecting and Error Correcting Codes" *Bell Sys. Tech. Journal*, 29, 147-160.
- [Herman 1996] Herman, M., "The Fundamentals of H.324 Desktop Video Conferencing", *Electronic Design*, 14 de Outubro, 1996, 114-128.
- [Huffman 1952] Huffman, D.A., "A Method for the Construction of Minimum Redundancy Codes", *Proc. IRE*, 40, 10, 1098-1101.
- [Hunter 1980] Hunter, R. e Harry Robinson, A., "International Digital Facsimile Coding Standards", *Proceedings of the IEEE*, 68, 7, Julho 1980, 854-867.
- [Jayme 1992] Jayme, C.A., "Proposta de um Método para Compactação de Imagens de Assinaturas", Dissertação de Mestrado, CPGEI, CEFET-PR, Abril 1992.

- [Jayme et al. 1993] Jayme, C.A., Marques Filho, O. e Godoy Jr., W., "Proposta de um Método para Compactação de Imagens de Assinaturas, Aplicável à Automação Bancária", Congresso do Mercosoft 93, Curitiba-PR, Maio 1993.
- [Kawaguchi 1980] Kawaguchi, E. e Endo, T., "On a Method of Binary-Picture Representation and Its Application to Data Compression", *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2, 1, Janeiro 1980, 27-35.
- [Kawano 1997] Kawano, W., "Técnicas e Padrões de Compressão de Vídeo para Sistemas Multimídia Distribuídos", Monografia - Curso de Especialização em Teleinformática, Centro Federal de Educação Tecnológica do Paraná, 1997.
- [Lynch 1985] Lynch, T.J., *Data compression techniques and applications*, Van Nostrand Reinhold, 1985.
- [Mallat 1989] Mallat, S.G., "A theory for multiresolution signal decomposition: the wavelet representation", *IEEE Transactions Pattern Analysis and Machine Intelligence*, PAMI-11(7), 674-693.
- [Monnes e Furht 1994] Monnes, P. e Furht, B., "Parallel JPEG Algorithms for Still Image Compression", *Proceedings of Southeastcom '94*, Abril 1994, 375-379.
- [Nelson 1989] Nelson, M., "LZW Data Compression", *Dr. Dobb's Journal*, Outubro 1989.
- [Nelson e Gaily 1996] Nelson, M. e Gaily, J.L., *The Data Compression Book 2nd ed.* M&T Books, 1996.
- [Pennenbaker e Mitchell 1993] Pennenbaker, W.B. e Mitchell, J.L., *JPEG still image data compression standard*, Van Nostrand Reinhold, 1993.
- [Pirsch et al. 1995] Pirsch, P. et al., "VLSI architectures for video compression: a survey", *Proceedings of the IEEE*, Fevereiro 1995, 220-246.
- [Pratt et al. 1980] Pratt, W.K. et al., "Combined Symbol Matching Facsimile Data Compression System", *Proceedings of the IEEE*, 68, 7, Julho 1980, 786-796.
- [Rijske 1996] Rijske, K., "H.263: video coding for low-bit-rate communication", *IEEE Communications Magazine*, Dezembro 1996, 42-45.
- [Rioual e Vetterli 1991] Rioual, O. e Vetterli, M., "Wavelets and Signal Processing", *IEEE Signal Processing Magazine*, Outubro 1991, 14-38.
- [Tenenbaum et al. 1990] Tenenbaum, A.M., Langsam, Y. e Augenstein, M.J., *Data Structures Using C*, Prentice-Hall, 1990.

- [Ting 1980] Ting, D. e Prasada, B., "Digital Processing Techniques for Encoding of Graphics", *Proceedings of the IEEE*, 68, 7, Julho 1980, 757-769.
- [Usubuchi 1980] Usubuchi, T. et al., "Adaptive Predictive Coding for Newspaper Facsimile", *Proceedings of the IEEE*, 68, 7, Julho 1980, 807-813.
- [Vetterli 1984] Vetterli, M., "Multi-dimensional sub-band coding: some theory and algorithms", *Signal Processing*, 6, 97-112.
- [Wallace 1991] Wallace, G., "The JPEG still picture compression standard", *Communications of the ACM*, 34, 30-44.
- [Williams 1991] Williams, R.N., *Adaptive data compression*, Kluwer Academic Publishers, 1991.
- [Yasuda 1980] Yasuda, Y., "Overview of Digital Facsimile Coding Techniques in Japan", *Proceedings of the IEEE*, 68, 7, Julho 1980, 830-845.

Capítulo 7

Aspectos Práticos de Hardware e Software para Processamento de Imagens

Os assuntos abordados nos capítulos anteriores constituem, em sua maioria, um conjunto de conhecimentos teóricos clássicos das principais técnicas de processamento digital de imagens. Este capítulo procura contemplar os leitores interessados em implementar ou testar na prática algumas destas técnicas. Para tanto, contém maiores detalhes sobre hardware específico para processamento de imagens (Seção 7.1), desde os sensores utilizados para aquisição, passando por placas de processamento (*frame grabbers*) e armazenamento (*frame buffers*) e chegando até os principais dispositivos de armazenamento, exibição e impressão disponíveis atualmente.

Na Seção 7.2 encontram-se compilados diversos títulos de software – classificados segundo a sua finalidade – e ambientes para desenvolvimento de aplicativos. Muitos destes títulos encontram-se disponíveis em repositórios de *shareware* na Internet, indicados no final do capítulo.

7.1 O hardware

Para adquirir imagens digitais, são necessários equipamentos especiais. Normalmente, a imagem a ser adquirida é primeiramente convertida em sinal elétrico analógico através de sensores ópticos. Posteriormente, esse sinal analógico é convertido em sinal digital através de circuitos eletrônicos específicos chamados *frame grabbers* (dispositivos de captura de quadro), tornando possível a interpretação por computadores. Uma vez digitalizada a imagem, essa pode ser adequadamente processada, muitas vezes fazendo uso de arquiteturas especiais para agilizar o processo. Por fim, utilizam-se dispositivos de saída, como monitores e impressoras – para a visualização dos resultados – e dispositivos de armazenamento – para a preservação dos mesmos.

7.1.1 Sensores

O processo de gerar dados a partir de imagens nada mais é do que a conversão da intensidade luminosa em sinais elétricos distribuídos espacialmente. Entre os diversos tipos de sensores existentes, os de maior destaque são os sensores a válvula e os sensores de estado sólido.

Sensores a válvula

Os primeiros dispositivos sensores de imagens foram os tubos de câmera utilizados nas primeiras câmeras analógicas de televisão. Seu funcionamento consiste na varredura, através de um feixe eletrônico, de uma superfície fotossensível à base de fósforo. A intensidade de corrente do feixe eletrônico é então proporcional à intensidade luminosa incidente no ponto da superfície fotossensível em questão. A varredura é responsável pelo mapeamento espacial da imagem, ou seja, o valor do sinal elétrico obtido em determinado instante de tempo corresponde a um ponto específico da superfície fotossensível.

A varredura é realizada da esquerda para a direita e de cima para baixo, até que um quadro seja completado, conforme ilustra a figura 1. A varredura é entrelaçada, isto é, primeiramente são varridas as linhas ímpares e posteriormente as linhas pares. Para facilitar a reprodução em um tubo de imagem, são acrescentados ao sinal de vídeo referências de sincronismo horizontal e vertical, gerando o chamado sinal composto de vídeo (SCV).

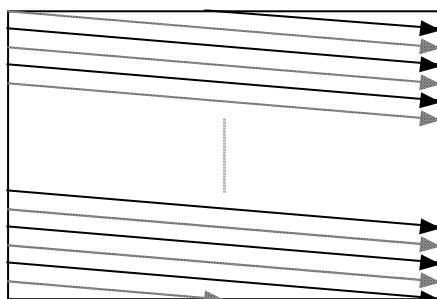


Figura 1 - Varredura da superfície fotossensível pelo feixe de elétrons

Entre os tubos de câmera mais comuns estão o Vidicon, o Saticon e o Plumbicon. Esses dispositivos apresentam a desvantagem de serem relativamente volumosos, difíceis de ser ajustados por se tratar de válvulas eletrônicas, além de desvantagens com respeito a consumo de energia e aquecimento, que tendem a ser eliminadas com o uso de dispositivos de estado sólido. Entretanto, a varredura dos tubos de câmera, capaz de converter uma imagem bidimensional em um sinal elétrico unidimensional variante no tempo, foi responsável pela criação de um formato padrão de transmissão serial de dados que continua sendo significativo nos dias atuais: o padrão RS-170.

Sensores de estado sólido (linear e de área)

Existem diversas alternativas para conversão de imagens em sinais elétricos utilizando-se dispositivos de estado sólido. Entre os principais dispositivos estão as matrizes de fotodiódos, dispositivos de injeção de carga (CID) e dispositivos de carga acoplada (CCD). Focalizaremos nossa atenção no tipo de dispositivo mais comumente utilizado na atualidade: o CCD.

A tecnologia CCD baseia-se em pastilhas semicondutoras com determinado número de recipientes capazes de armazenar carga elétrica, que possuem um determinado mecanismo de transferência entre elas. A quantidade de carga elétrica armazenada nos recipientes corresponde a valores analógicos, o que equivale a dizer que erros de quantização são praticamente nulos. Dessa maneira, o CCD equivale a uma memória analógica, cujos dados são acessados serialmente através da transferência de carga entre os recipientes.

A utilização de materiais fotossensíveis permite a construção de dispositivos CCD cujos recipientes de carga constituem também elementos sensíveis à luz. Assim, a carga armazenada em cada recipiente é proporcional à intensidade luminosa incidente sobre o mesmo. Posteriormente, a leitura seqüencial dos recipientes pode gerar um sinal elétrico variante no tempo nos moldes do padrão gerado por tubos de câmera.

Normalmente, a formação de imagens envolve a utilização de sensores CCD de área, que consistem em matrizes bidimensionais de elementos fotossensíveis. Sensores de área são amplamente utilizados em câmeras de vídeo digitais e contribuem para o reduzido tamanho desses equipamentos. Outra abordagem, particularmente importante para aplicações que demandam alta resolução, emprega sensores CCD lineares na técnica conhecida por varredura matricial linear. Os sensores lineares utilizados são matrizes que contêm os elementos fotossensíveis alinhados em uma única coluna ou linha de alta resolução espacial. A informação luminosa da imagem bidimensional é então convertida em sinais elétricos através de varredura espacial perpendicular à linha sensora, ou seja, através do deslocamento relativo do sensor em relação à imagem em questão. Esta é a técnica empregada nos equipamentos de fac-símile e scanners de documentos.

Os sensores CCD oferecem diversas vantagens sobre os tubos de câmera. Entre elas estão: o tamanho reduzido; o menor consumo de energia; a menor suscetibilidade a efeitos de espalhamento (*blooming*); a melhor resposta a mudanças na iluminação incidente; e o bom desempenho em condições de baixa iluminação [Schalkoff 1989].

Entretanto, a tecnologia de fabricação dos sensores CCD é um tanto dispendiosa financeiramente e, além disso, não permite a inclusão de circuitos de controle na mesma pastilha semicondutora. Tentando contornar esses problemas, um grupo de pesquisadores da Universidade de Edinburgo desenvolveu um novo tipo de sensor, utilizando a tecnologia

CMOS. A abordagem escocesa permite que sejam incluídos na mesma pastilha circuitos de controle, conversores analógico/digital, memórias digitais e circuitos com funções específicas para o processamento de imagens, além da matriz sensora propriamente dita. É possível, desse modo, integrar um sistema completo de visão por computador numa única pastilha de baixo custo. Outra vantagem da tecnologia CMOS é o consumo de energia, que é reduzido a aproximadamente um quinto do consumo da tecnologia CCD [Vellacott 1994].

Sensores de imagens coloridas necessitam de três sensores monocromáticos operando em conjunto com filtros de cores, a fim de gerar os sinais das três cores primárias aditivas: o vermelho (R), o verde (G) e o azul (B). Para efeito de geração de sinal de vídeo, as três componentes R, G e B são combinadas aritimeticamente para gerar o sinal de luminância e codificadas e moduladas para gerar a componente de crominância. O sinal de luminância equivale ao sinal de vídeo monocromático da imagem, sobre o qual é superposta a informação de cor.

Leitura complementar

Maiores detalhes sobre a formação do sinal composto de vídeo podem ser encontradas no capítulo 8 de [Schalkoff 1989] e nos capítulos 7 e 8 de [Grob 1989].

Ainda em [Grob 1989], em seu capítulo 3, encontram-se mais informações sobre tubos de câmera.

Sobre o processo de funcionamento de dispositivos CCD, também recomenda-se a leitura do capítulo 8 de [Schalkoff 1989].

7.1.2 *Frame grabbers / frame buffers*

Para que seja possível o processamento de imagens através de algoritmos computacionais é preciso que os dados estejam disponíveis na forma digital. Como explicado anteriormente, é comum que os sensores de imagens forneçam sinais elétricos analógicos variantes no tempo representando a imagem de entrada, como é o caso das câmeras de vídeo. Faz-se necessária, então, uma conversão do padrão analógico de representação da imagem para o formato binário de representação, utilizado pelos computadores.

O dispositivo responsável pela conversão do sinal analógico de vídeo para uma matriz de dados digitais contendo informações sobre a imagem é conhecido como dispositivo de captura de quadro (*frame grabber*).

Cabe a esse dispositivo detetar as informações de sincronismo horizontal e vertical do sinal composto de vídeo, determinando os limites de informação da imagem. Uma vez detetado um pulso de sincronismo vertical, tem início a captura de um quadro. O sinal analógico a partir de então é amostrado no tempo, passa por uma conversão analógico/digital e é armazenado em memória. Dá-se o nome de memória de quadro (*frame buffer*) ao sistema de armazenamento volátil de imagens digitalizadas.

É importante ressaltar que a resolução horizontal da imagem obtida depende do número de amostras realizadas numa linha, ou seja, entre dois pulsos de sincronismo horizontal do sinal composto de vídeo. A quantidade de tons de cinza (ou de cores) depende da capacidade de quantização do(s) conversor(es) analógico/digital empregado(s).

O processo se repete até que seja detetado o final do quadro, ou seja, o próximo pulso de sincronismo vertical. Cabe dizer aqui que a resolução vertical da imagem digital obtida é limitada pelo número de linhas imposto pelo padrão de sinal de vídeo utilizado. Os pulsos de sincronismo horizontal e vertical servem de referência para a montagem da matriz de dados da imagem, juntamente com as informações de quantidade de amostras por linha (resolução horizontal) e quantidade de linhas propriamente dita (resolução vertical).

Para a captura de imagens digitais coloridas são necessários três circuitos de aquisição independentes para cada um dos sinais de cor R, G e B. O sistema de detecção de informações de sincronismo é comum aos três circuitos de aquisição, porém é necessária a existência de um sistema complementar para a decodificação do sinal de crominância nas três componentes de cor.

A construção de dispositivos de aquisição de imagens através de sinal de vídeo apresenta diversos desafios. A velocidade de conversão analógico/digital talvez seja a maior dificuldade a ser enfrentada. O tempo de duração da informação de uma linha de vídeo no sistema PAL-M, por exemplo, é de aproximadamente $53,3 \mu\text{s}$ ($53,3 \times 10^{-6}$ segundos). Desse modo, para se obter 512 amostras nesse tempo é necessária a utilização de um conversor analógico/digital capaz de realizar a conversão em menos de 104 ns (104×10^{-9} segundos), equivalente a aproximadamente 9,6 milhões de amostras por segundo. Dessa forma, torna-se necessária a utilização dos velozes e custosos conversores analógico/digitais do tipo *flash*. Outra dificuldade envolve a alta quantidade de memória envolvida no processo, que também exige taxas de transferência de dados superiores a 7,5 megabytes por segundo [Schalkoff 1989].

Devido a essas complicações técnicas, tais dispositivos de aquisição normalmente possuem um custo alto, principalmente quando possibilitam a obtenção de elevadas resoluções.

7.1.3 Arquiteturas

Tendo em vista o exposto nos capítulos anteriores, pode-se concluir que as técnicas de processamento de imagens geralmente exigem um grande poder de processamento. O esforço computacional exigido supera a capacidade de processamento de muitas arquiteturas de computador que fazem uso de um único microprocessador. Por mais que se consiga acelerar a velocidade de um microprocessador em termos de freqüência de operação, em algum momento um limite será atingido.

Por outro lado, nos capítulos anteriores foram apresentadas técnicas que em sua grande maioria podem ser realizadas paralelamente. Isto significa dizer que diversas etapas independentes de um algoritmo podem ser calculadas ao mesmo tempo, para depois os resultados parciais serem combinados gerando o resultado final. Para fazer uso dessa característica tão freqüente dos algoritmos de processamento de imagens pode-se implementar computadores especiais, utilizando arquiteturas multiprocessadas. Um requisito essencial para bem aproveitar o poder computacional de arquiteturas com múltiplos microprocessadores é o desenvolvimento de programas e algoritmos dedicados, que possibilitem a execução paralela das tarefas.

Um conceito importante, o qual deve-se ter em mente, é o de que o desempenho não aumenta linearmente com o aumento de processadores empregados. Isso significa dizer que se utilizarmos dois processadores em paralelo não será obtido um desempenho duas vezes superior ao apresentado por um único processador e sim um desempenho sensivelmente inferior a duas vezes. Esse fato se deve a perdas de tempo devidas à troca de informações entre os processadores, não computadas pelos algoritmos em si.

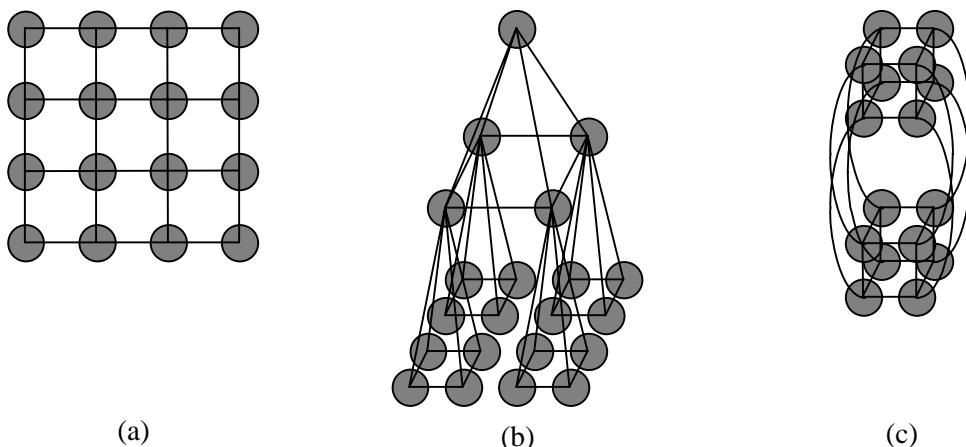


Figura 2 - Estruturas de conexão de arquiteturas multiprocessadas: (a) estrutura em malha; (b) estrutura em pirâmide; (c) estrutura em hipercubo.

As arquiteturas multiprocessadas geralmente são classificadas segundo a taxonomia de Flynn, baseando-se na multiplicidade de fluxo de instruções e dados. São possíveis, segundo

essa classificação, quatro arquiteturas diferentes: SISD (*Single Instruction-Single Data*), SIMD (*Single Instruction-Multiple Data*), MISD (*Multiple Instruction-Single Data*) e MIMD (*Multiple Instruction-Multiple Data*). A estrutura de conexão entre os processadores também constitui uma forma de classificação: em malha, em pirâmide, em hipercubo, etc., como ilustrado na figura 2.

Foge ao escopo deste capítulo a completa exposição do assunto, por se tratar de tema bastante extenso e específico. O leitor interessado encontrará maiores informações nas indicações bibliográficas.

Leitura complementar

É sugerida a leitura do capítulo 9 de [Dougherty 1994] para maiores detalhes sobre as diversas arquiteturas multiprocessadas.

Para considerações sobre algoritmos paralelos, consultar o capítulo 8 de [Schalkoff 1989].

O capítulo 8 de [Pearson 1991] trata de arquiteturas para processamento de imagem.

Ainda em [Pearson 1991], no seu capítulo 10, existem maiores detalhes sobre arquiteturas paralelas para processamento de imagens e redes neurais artificiais.

7.1.4 Dispositivos de saída

De pouco adianta realizar operações com imagens se não pudermos visualizar o resultado. Com poucas exceções, como em alguns casos de reconhecimento de padrões, o resultado do processamento retorna também na forma de imagem.

Entre os dispositivos de saída mais comumente encontrados, temos os monitores de vídeo, seguidos pelas impressoras e finalmente pelos *plotters*. Esses dispositivos têm função exatamente oposta à dos sensores, ou seja, transformar dados digitais em informação visível.

Monitores de vídeo

Representam a classe de dispositivos de saída mais veloz, entretanto volátil. Os circuitos de um monitor buscam realizar exatamente o oposto de uma câmera de vídeo, isto é, a partir do sinal composto de vídeo efetuar a reconstrução da imagem em termos visuais.

Em monitores que se utilizam de tubos de imagem (tubos de raios catódicos - TRC) para apresentar resultados, o sistema de geração de imagens é similar aos sistemas sensores que se utilizam de tubos de câmera. Da mesma forma, é realizada uma varredura da tela através de um feixe de elétrons. A superfície da tela é revestida por um material capaz de emitir luz quando atingido pelo feixe de elétrons, em intensidade luminosa proporcional à intensidade de corrente elétrica do feixe. Os pulsos de sincronismo presentes no sinal composto de vídeo são utilizados para orientar a varredura da tela, reproduzindo a informação visual adequadamente.

Monitores coloridos possuem três canhões de elétrons, correspondentes aos três sinais R, G e B que compõem a informação de cor. O feixe de elétrons de cada canhão atinge pontos específicos na tela, os quais não podem ser atingidos pelos outros canhões. Esses pontos, correspondentes às três informações de cor são dispostos muito próximos uns dos outros como ilustra a figura 3, a composição final da cor é realizada pelo olho. Tais equipamentos apresentam o inconveniente de, além de serem volumosos devido ao tamanho do tubo de câmera empregado, apresentarem também dificuldades de ajustes.



Figura 3 - Disposição dos pontos R, G e B na tela de um monitor.

Existem também monitores que utilizam a tecnologia de cristal líquido (LCD), já bastante desenvolvida nos dias atuais. O cristal líquido possui uma propriedade física que possibilita o seu uso em dispositivos de geração de imagens: quando não excitado eletricamente

se apresenta opaco, e quando convenientemente excitado suas moléculas se orientam de maneira a permitir a passagem de luz. Sendo assim, é possível a construção de matrizes de elementos de cristal líquido capazes de apresentar imagens. Mais uma vez, para a apresentação de imagens coloridas é necessária a implementação de matrizes para cada um dos sinais componentes R, G e B de cor.

Os monitores de cristal líquido estão para os sensores CCD assim como os tubos de imagens estão para os tubos de câmera. A tecnologia de cristal líquido permite a obtenção de monitores mais compactos, cujo consumo de energia é bastante reduzido, possibilitando o seu uso em computadores portáteis. Possuem, no entanto, a desvantagem de apresentarem diferentes tonalidades conforme o ângulo visual formado com a tela.

Impressoras

São dispositivos que apresentam imagens definitivas em sua saída, normalmente em papel. Muitas tecnologias de impressão estão disponíveis atualmente, sendo as mais conhecidas a matricial, a jato de tinta e a laser.

As primeiras impressoras gráficas que surgiram faziam uso da tecnologia matricial, a qual utiliza-se de uma cabeça de impressão composta de pequenas agulhas que são disparadas contra uma fita entintada com o objetivo de marcar o papel. O número de agulhas pode variar de 7 a 24, dispostas em linha vertical. Para a obtenção de imagens bidimensionais, a cabeça de impressão é deslocada em relação ao papel na direção horizontal, como mostrado na figura 4. Depois de impressa uma linha completa, o papel é avançado verticalmente em relação à cabeça de impressão.

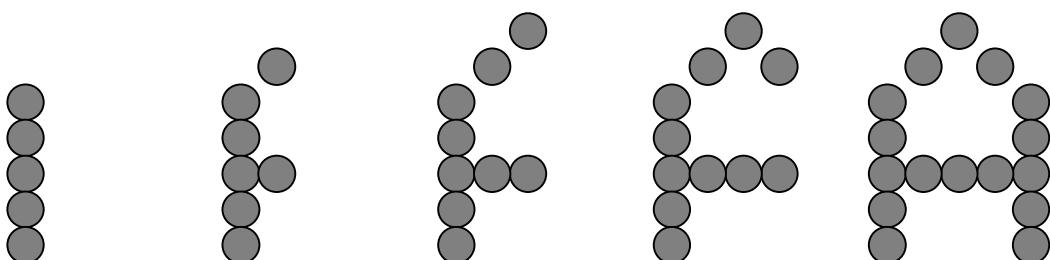


Figura 4 - Seqüência de impressão matricial do caracter “A”.

Diferentemente do que ocorre nos monitores de vídeo, para que seja possível a geração de imagens coloridas em impressoras, são utilizadas as cores primárias subtrativas: o ciano (C), o magenta (M), o amarelo (Y) e o preto (K). Portanto, no caso das impressoras matriciais, é necessária a utilização de uma fita de impressão colorida contendo faixas dessas quatro cores. A composição da imagem final é feita imprimindo-se cada cor separadamente.

As impressoras matriciais são as mais acessíveis financeiramente, porém são demasiadamente ruidosas e não possibilitam a reprodução de imagens de alta resolução.

A tecnologia a jato de tinta possibilita a obtenção de imagens monocromáticas ou coloridas de alta resolução de maneira silenciosa e a preços razoáveis. A cabeça de impressão desse tipo de impressora é constituída de elementos capazes de espirrar tinta líquida sobre o papel. O ponto produzido por esses elementos é bastante localizado e pequeno a ponto de ser possível o alinhamento de diversos deles verticalmente na cabeça de impressão. De maneira semelhante à impressão matricial, a cabeça é deslocada horizontalmente em relação ao papel, o qual é avançado verticalmente ao término da impressão da linha. A diferença está na resolução possível de ser obtida com essa tecnologia e, consequentemente, na qualidade da imagem impressa.

Para a impressão de imagens coloridas em impressoras jato de tinta é preciso o uso de cartuchos contendo tintas das cores C, M, Y e K. Normalmente utilizam-se dois cartuchos: um contendo a tinta preta e outro contendo as demais tintas coloridas. Impressoras mais acessíveis exigem que o usuário alterne os cartuchos manualmente durante a impressão de uma imagem colorida, outras um pouco mais custosas permitem a comodidade de utilizar os dois cartuchos simultaneamente. Da mesma forma que nas impressoras matriciais coloridas, imagens coloridas

são obtidas através da combinação das impressões de cada uma das cores primárias, realizadas separadamente.

As impressoras a laser utilizam o mesmo princípio de funcionamento de máquinas fotocopiadoras, radicalmente diferente das tecnologias matricial e a jato de tinta. O elemento de impressão utilizado nessas impressoras é chamado toner e consiste num pó capaz de ser carregado eletrostaticamente. Portanto, o toner quando carregado é capaz de ser atraído por objetos de carga oposta.

Para gerar a imagem a ser impressa utiliza-se um cilindro eletrostático, que é carregado eletrostaticamente com carga oposta à carga com que é carregado o toner. Um feixe de laser é então utilizado para varrer o cilindro, apagando as áreas onde o toner não deve aderir. Isso é possível graças a uma propriedade física do material utilizado para revestir o cilindro eletrostático, a qual permite que as áreas atingidas pelo feixe de laser sejam descarregadas eletricamente. O processo de varredura pode ser visto esquematicamente na figura 5.

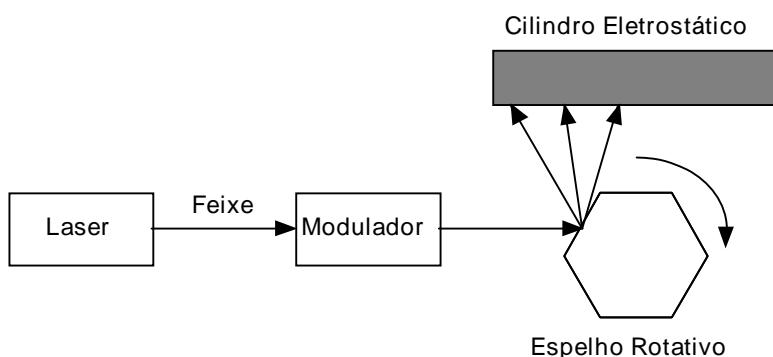


Figura 5 - Varredura do cilindro eletrostático pelo feixe de laser.

Uma vez desenhada a imagem sobre o cilindro, permite-se que o toner entre em contato com o mesmo, aderindo sobre as áreas carregadas. Feito isso, passa-se o papel entre o cilindro eletrostático e um segundo cilindro auxiliar que exerce pressão sobre o papel. Desse modo o toner é transferido para o papel, deixando a imagem impressa. Finalmente, o papel passa por uma unidade de fixação que faz uso de alta temperatura para fixar o toner ao papel. É possível a obtenção de imagens de boa resolução através da impressão a laser.

Existem impressoras a laser coloridas, cujo custo ainda é inacessível aos usuários comuns. Isso se deve ao fato do processo de geração de imagens coloridas a laser ser bastante complexo na prática, apesar de seguir as mesmas linhas mestras da geração de imagens monocromáticas. Novamente, é necessária a utilização de toner nas cores C, M, Y e K para a geração das imagens coloridas.

Plotters

Esses equipamentos permitem a reprodução de imagens em papel. São mais adequados para imagens de desenhos gerados por aplicativos CAD, com predominância de linhas e formas geométricas.

Um *plotter* literalmente desenha sobre o papel, utilizando-se para isso de uma ou mais canetas (ou penas) coloridas. As canetas são controladas para abaixarem ou subirem, tocando ou não o papel. O desenho das linhas é obtido através do deslocamento relativo entre as canetas e o papel, nas direções horizontal e vertical. Normalmente as canetas são deslocadas numa direção e o papel na outra, perpendicularmente.

Os *plotters* operam com diversos tamanhos de papel, podendo chegar ao tamanho A0. Muitas vezes são utilizados para desenhos de alta precisão, como por exemplo, desenhos de placas de circuito impresso. Em geral são dispositivos pouco acessíveis aos usuários comuns.

Leitura complementar

Para obter mais informações sobre o funcionamento e a formação de imagens em tubos de imagens, aconselha-se a leitura do capítulo 4 de [Grob 1989] e do capítulo 3 de [Rubinstein 1988]. Esse último também contém maiores informações sobre dispositivos de impressão.

7.1.5 Dispositivos de armazenamento

O armazenamento de imagens também constitui um fator de extrema importância em diversas aplicações do processamento de imagens. A coleta de imagens astronômicas para efeito de análise e comparações é um exemplo bem claro dessa importância.

Os dispositivos de armazenamento de massa sofreram grandes avanços nos últimos anos. Grandes quantidades de informação que antes eram possíveis de serem armazenadas apenas em fita magnética ou em gigantescos discos rígidos agora podem ocupar cartuchos removíveis. Os chamados Zip Drives são capazes de armazenar 100MB em apenas um cartucho. Indo bem mais além, os Jaz Drives possuem capacidade de armazenamento de 1GB em apenas um cartucho removível.

Completando o leque de opções de armazenamento estão os dispositivos de armazenamento óptico e magneto-óptico. A tecnologia de discos compactos (CD) encontra-se atualmente bastante popularizada e acessível aos diversos campos de aplicação. O CD-ROM e o Photo CD, desenvolvido pela Kodak, representam duas boas alternativas para armazenamento definitivo de imagens em mídia óptica, com capacidades aproximadas de 600MB. Os cartuchos magneto-ópticos oferecem ainda a possibilidade de apagamento e regravação de dados, tornando-os ainda mais flexíveis quanto ao uso. Para armazenamento de imagens de vídeo, um novo padrão de CD está emergindo: o vídeo-disco digital (DVD), o qual permite o armazenamento de imagens em movimento de alta qualidade, além de trilha sonora de qualidade superior à do CD de áudio convencional. A especificação do DVD permite até 17GB de dados armazenados em um único CD, porém atualmente esse limite situa-se em torno de 4GB.

O armazenamento em fita continua sendo, no entanto, uma solução viável para o armazenamento de enormes quantidades de dados, apresentando a desvantagem da lentidão na procura de informações e a deterioração da mídia magnética com o passar do tempo. Os dispositivos ópticos possuem a vantagem de não sofrerem deterioração com o tempo, como acontece com os dispositivos que fazem uso de mídia magnética.

7.2 O software

Diferentes aplicações exigem diferentes programas de processamento e análise de imagens. Existem diversos aplicativos comerciais para edição e manipulação de imagens fotográficas, os quais permitem diversas operações de filtragem e retoques. Outras aplicações mais sofisticadas, como as científicas, exigem operações morfológicas, transformações matemáticas e outros recursos.

O software para aplicações específicas normalmente aparece na forma de bibliotecas de funções para programação, permitindo a integração do processamento e análise de imagens a outras técnicas computacionais, como a inteligência artificial e o reconhecimento de padrões.

A quantidade de títulos de *software* disponíveis atualmente é imensa. Alguns deles estão classificados a seguir.

7.2.1 Títulos disponíveis e classificação

Software para Aplicações Científicas

Global Lab Image

O software Global Lab Image foi criado pela Data Translation para uso científico do processamento de imagens. Ele permite efetuar diversas operações, tais como: realce, manipulação de histograma, filtragem morfológica, análise freqüencial, contagem, medição e classificação automática de objetos, etc.

O Global Lab Image possui ainda uma biblioteca de funções que possibilita o desenvolvimento de outros aplicativos para a plataforma PC em ambiente MS-Windows.

Uma versão demonstrativa do Global Lab Image está disponível na Internet no endereço "http://www.datx.com/tech/global_lab_img.htm". O endereço da Data Translation é "<http://www.datx.com/>".

LATIM

O Laboratório de Tratamento de Imagens (LATIM) é um software para processamento de imagens, cujo caráter é didático. Engloba as principais técnicas de manipulação e processamento de imagens, tendo como principais objetivos: oferecer uma ferramenta versátil capaz de auxiliar na geração e teste de novas técnicas e algoritmos; e permitir demonstrações de cunho didático das principais técnicas de processamento de imagens.

Desenvolvido em 1989 pela Universidade Federal do Rio de Janeiro (UFRJ) e escrito em linguagem Pascal, o LATIM trabalha com um formato de armazenamento de imagens proprietário. Inclui rotinas de exibição de imagens, manipulação de histograma, convolução (filtragem) e operações lógicas e aritméticas, entre outras. Uma característica importante é a possibilidade de expansão, que possibilita ao usuário incorporar ao sistema as suas próprias rotinas desenvolvidas em Pascal.¹

PC_IMAGE

O software para análise de imagens PC_IMAGE, desenvolvido pela empresa Foster Findlay Associates, possui uma poderosa gama de operações para processamento de imagens. Permite realizar operações aritméticas, limiarização, manipulação de histograma, filtragens linear e não-linear, morfologia matemática binária e em níveis de cinza, pseudocolorização, entre outras.

O PC_IMAGE foi desenvolvido para o MS-Windows a partir da biblioteca de funções C_IMAGES. Rotinas para aplicações especiais podem ser desenvolvidas utilizando-se a biblioteca C_IMAGES e chamadas em uma janela do PC_IMAGE.

Uma versão demonstrativa do PC_IMAGE está disponível na Internet, no endereço "<http://www.demon.co.uk/ffaltd/ffaftp.html>". Maiores informações sobre o software podem ser obtidas no endereço "<http://www.demon.co.uk/ffaltd/pcimage.html>". O endereço da Foster Findlay Associates é "<http://www.demon.co.uk/ffaltd/index.html>".

SITIM

O extinto Sistema de Tratamento de Imagens (SITIM), da empresa Engespaço, contava com uma biblioteca de software desenvolvida pelo Instituto Nacional de Pesquisas Espaciais (INPE). Dirigia-se a aplicações em sistemas geográficos de informações, sensoriamento remoto e microscopia.

Dentro de cada aplicação específica, eram possíveis operações lógicas e aritméticas, deteção de bordas, realce, filtragem, segmentação, extração de características, classificação supervisionada e não supervisionada, etc.

Algumas poucas informações históricas sobre o SITIM podem ser encontradas na Internet, na página do Laboratório de Tratamento de Imagens Digitais do INPE, cujo endereço é "http://www.ltid.inpe.br/html/desc_2.html".

TIMWIN

¹ Apesar de ter sido escrito em 1989 para os padrões da época (PC-XT com 640 kB de RAM, sem disco rígido, monitor CGA monocromático, etc.), o LATIM possui "importância histórica" por ter permitido viabilizar as aulas da disciplina de "Processamento de Imagens" do curso de Engenharia Industrial Elétrica do CEFET-PR quando os únicos recursos computacionais disponíveis eram exatamente estes.

É um software para processamento de imagens e medições de propriedades de objetos para a plataforma PC desenvolvido por TEA and DIFA Measuring Systems.

Mais de 200 operações de processamento de imagens estão disponíveis, divididas nos seguintes grupos: operações orientadas a pixel, operações orientadas a vizinhança, operações em planos de bits, morfologia matemática binária, operações geométricas, gráficos, operações de controle e transformada de Fourier, entre outras.

Apenas o formato TIFF é suportado, além do formato proprietário do aplicativo.

Maiores informações e também uma versão demonstrativa do TIMWIN estão disponíveis na Internet, no endereço "<http://www.ph.tn.tudelft.nl/Software/TimWin/timwww2.html>".

Software para Composição de Imagens Animadas

GifBuilder

É um *freeware* para criação de arquivos de imagens animadas no formato GIF, utilizando a plataforma Macintosh. Como entrada pode-se utilizar arquivos nos formatos GIF, PICT, PSDTIFF e TIFF ou ainda no formato QuickTime. A saída é um arquivo GIF de múltiplas imagens. Algumas opções disponíveis são a seleção de bits por pixel, palheta de cores, entrelaçamento, transparência, tempo de atraso entre os quadros, repetição de quadros, etc.

O GifBuilder pode ser obtido na Internet através do endereço "<http://www.shareware.com/>", bastando digitar o seu nome no mecanismo de busca.

GifCon

O *shareware* GIF Construction Set (GifCon), da empresa Alchemy Mindworks, permite a montagem de imagens animadas no formato GIF na plataforma PC. É possível a inclusão de elementos de controle do tempo de exposição de cada quadro. As imagens de cada quadro, no entanto, devem ser criadas com o auxílio de outro aplicativo que possibilite a gravação no formato GIF.

O endereço da Alchemy Mindworks na Internet é "<http://www.mindworkshop.com/alchemy/alchemy.html>". O software está disponível em "<http://www.mindworkshop.com/alchemy/gifcon.html>", onde existem também maiores informações.

Software para Conversão de Formatos

GraphicConverter

O *shareware* GraphicConverter para Macintosh foi desenvolvido por Thorsten Lemke com o objetivo inicial de converter os diversos formatos de arquivos de imagens entre si. Posteriormente foram adicionadas algumas ferramentas de manipulação de imagens, incluindo alguns filtros. Diversos formatos de arquivos são suportados, incluindo: BMP, EPSF, GIF, HPGL, IFF, IMG, JPEG, MacPaint, PBM, PCX, PIC, PICT, PNG, PSD, RAW, SUN, TGA, TIFF, WMF, XBM, etc. O GraphicConverter permite também a composição de imagens animadas em arquivos GIF e QuickTime Movie.

Graphic Workshop

O *shareware* Graphic Workshop, da empresa Alchemy Mindworks, existe em versões para MS-DOS e MS-Windows. Permite a leitura e conversão dos mais variados formatos de arquivos de imagens, entre eles: ART, BMP, CUT, HRZ, IFF, IMG, JPEG, LBM, MAC, MSP, PIC, PCX, RAS, RLE, TGA, TIFF e WPG. Permite ainda algumas operações de transformação e manipulação simples de imagens.

Maiores informações e o próprio software estão disponíveis em "<http://www.mindworkshop.com/alchemy/gifcon.html>", na Internet. A Alchemy

Mindworks pode ser contatada através da Internet pelo endereço "<http://www.mindworkshop.com/alchemy/alchemy.html>".

Hijaak Graphics Suite

O software HiJaak, da Quarterdeck Corporation, é um conjunto de aplicativos para manipulação e processamento de imagens e conversão de formatos de arquivos. É composto pelos módulos HiJaak Browser, HiJaak Smuggler, HiJaak Paint, HiJaak Draw e HiJaak PRO.

O conjunto de aplicativos suporta inúmeros formatos de arquivos de imagens, incluindo entre outros: BMP, CGM, DXF, EPS, GIF, IFF, IMG, JPEG, MacPaint, MSP, PCD, PCX, PGL, PICT, TGA, TIFF, WMF, WPG, além de formatos de arquivos multimídia, tais como AVI, FLI, MIDI, QuickTime e WAV. É possível a catalogação de arquivos e a conversão entre formatos, edição de imagens, criação de efeitos visuais, realce, filtragem, manipulação de histograma, etc.

Software para Manipulação de Imagens

Adobe PhotoShop

Desenvolvido pela Adobe Systems para as plataformas Macintosh e PC, o PhotoShop é um dos programas para manipulação de imagens fotográficas mais populares existentes no mercado

O PhotoShop permite uma série de operações, incluindo transformações geométricas, realce, filtragem e até mesmo alguns efeitos especiais. A estrutura do programa permite a inclusão de módulos de processamento de terceiros, chamados *plug-ins*, tornando o PhotoShop um aplicativo bastante versátil e com características expansíveis.

Há suporte para os formatos de arquivos BMP, EPS, GIF, IFF, JPEG, MacPaint, PCX, PICT, PSD, RAW, TGA e outros.

Aldus PhotoStyler

O PhotoStyler, desenvolvido pela Aldus Corporation, destina-se ao processamento de imagens fotográficas. É um programa bastante popular entre os programas de manipulação de imagens existentes.

Os formatos de arquivos de imagem suportados pelo PhotoStyler são: BMP, EPS, GIF, JPEG, MacPaint, PCD, PICT, PCX, PSD, RLE, TGA e TIFF. São possíveis operações de transformação geométrica, realce, filtragem e também alguns efeitos especiais.

Corel Photo-Paint

Incluído nos pacotes de software de editoração gráfica da Corel, está o aplicativo Photo-Paint, destinado ao processamento de imagens fotográficas. Possibilita algumas operações de filtragem e efeitos especiais, operações de realce e transformações geométricas.

O Photo-Paint possui suporte para os seguintes formatos de arquivos: BMP, GIF, JPEG, PCD, PCX, TGA e TIFF.

Paintbrush e Paint

O Paintbrush e o Paint são os programas de manipulação de imagens da Microsoft incluídos nas versões 3.x e 95 do MS-Windows, respectivamente. Oferecem pouco suporte a formatos de arquivos de imagem: apenas BMP, MSP e PCX. Oferecem também poucos recursos para processamento, porém alguns recursos para desenho.

PhotoFinish

O software PhotoFinish da ZSoft Corporation permite o processamento de imagens fotográficas através de operações de transformação geométrica, realce, filtragem e diversos efeitos especiais. Suporta os formatos BMP, GIF, JPEG, MSP, PCD, PCX, TGA e TIFF.

Software para Visualização de Imagens

JPEGView

O *postcardware*² JPEGView para Macintosh foi desenvolvido por Aaron Giles para apresentação de imagens no formato JPEG. Entretanto, há suporte também para os formatos BMP, GIF, MacPaint, TIFF e PICT. O JPEGView permite ainda a visualização de imagens animadas no formato GIF.

LView

Desenvolvido por Leonardo Haddad Loureiro para ambiente MS-Windows, o *freeware* LVIEW permite a visualização de arquivos de imagens nos formatos BMP, GIF, JPEG e TGA. É possível ainda realizar algumas operações geométricas e alguns ajustes nas imagens.

PreVue

O PreVue é um *shareware* desenvolvido por Marvin Gozum que permite a apresentação de arquivos de imagens nos formatos BMP, DCX, EPS, GIF, JPEG PICT, PCX e TGA. Permite ainda a visualização de imagens animadas nos formatos AVI, FLI e FLC, além de ler arquivos de som MIDI.

WinJPEG

O *shareware* WinJPEG, desenvolvido por Norman Yee e Ken Yee, permite pequenos ajustes e visualização de imagens nos formatos BMP, GIF, JPEG, PCX, TGA e TIFF.

CompuShow

O CompuShow (CSHOW) é um *shareware* da empresa Canyon State Systems and Software para visualização de imagens nos formatos BMP, GIF, IMG, MacPaint, MSP, PCX, TGA, TIFF, entre outros.

WinLab

A empresa Ph.D. Software desenvolveu o *shareware* Winlab, que permite a leitura de arquivos de imagens nos formatos BMP, GIF, IMG, PCX, RAS, RAW, TGA e TIFF. Além da simples apresentação das imagens, o WinLab é capaz de realizar equalização de histograma e filtragem, entre outras operações.

7.2.2 Linguagens e ambientes para desenvolvimento

Biblioteca de Subrotinas Aurora

Desenvolvida pela empresa Data Translation em linguagem C, a biblioteca de subrotinas Aurora possui diversas funções para operações lógicas e aritméticas, convolução (filtragem), operações orientadas a vizinhança, operações estatísticas e outras.

Outras informações podem ser obtidas diretamente com o fabricante através da Internet, no endereço "<http://www.datx.com/>".

Biblioteca de Subrotinas para Processamento de Imagens DT-Iris

Similarmente à biblioteca Aurora, a biblioteca DT-Iris da Data Translation possui funções escritas em linguagem C para operações lógicas e aritméticas, convolução (filtragem), manipulação de histograma, etc.

Maiores informações são obtidas diretamente com o fabricante no endereço "<http://www.datx.com/>", através da Internet.

² O curioso termo *postcardware* foi cunhado pelo próprio autor do programa, para enfatizar ao usuário que a 'taxa' cobrada para a regularização do software é o envio de um cartão postal.

C_IMAGES

A biblioteca C_IMAGES, desenvolvida pela empresa Foster Findlay Associates, contém diversas rotinas de processamento e análise de imagens. Escrita em linguagem C, a biblioteca é independente de plataforma de *hardware*.

O módulo principal permite filtragens linear e não-linear, limiarização, manipulação de histograma, operações de morfologia matemática binária, operações de medida, operações de transformação geométrica, operações lógicas e aritméticas, entre outras.

Alguns módulos opcionais incluem operações de conversão de formatos de arquivos de imagens, morfologia matemática em níveis de cinza e processamento de imagens coloridas. Recentemente foi desenvolvido também um módulo especial para o processamento de imagens tridimensionais.

Uma versão demonstrativa está disponível na Internet, no endereço "<http://www.demon.co.uk/ffaltd/ffafpt.html>". Maiores informações sobre a biblioteca podem ser obtidas no endereço "<http://www.demon.co.uk/ffaltd/cimages.html>". O endereço da Foster Findlay Associates é "<http://www.demon.co.uk/ffaltd/index.html>".

Caixa de Ferramentas (Toolbox) para Processamento de Imagens do MATLAB

Trata-se de uma biblioteca de rotinas desenvolvidas para o software MATLAB, específicas para o processamento de imagens. É possível a criação de programas (arquivos M) para o MATLAB, utilizando-se dessas funções e possivelmente criando outras.

Existe suporte para os formatos BMP, GIF, HDF, PCX, TIFF e XWD. As funções permitem operações de manipulação e realce de imagens, transformações geométricas, morfologia matemática binária, filtragem, transformada de Fourier bidimensional, extração de dados estatísticos, etc.

O endereço da MathWorks, fabricante do MATLAB, na Internet, é "<http://www.mathworks.com/>".

DADiSP

O software DADiSP foi especialmente desenvolvido para uso de engenheiros e cientistas pela empresa DSP Development Corporation.

Disponível para diversas plataformas, o DADiSP consiste numa interface gráfica de processamento e análise de sinais e imagens. Possui excelente capacidade de apresentação de gráficos, além de permitir operações com matrizes, transformada de Fourier bidimensional, análise estatística, filtragem digital, etc.

O endereço na Internet da DSP Development Corporation é "<http://www.dadisp.com/>", onde estão disponíveis maiores informações e versões demonstrativas do DADiSP.

Khoros

O Khoros é um ambiente de desenvolvimento para processamento de imagens, gráficos e visualização. É composto de bibliotecas de programação, exemplos de código e ferramentas de programação, projetadas especificamente para diversas áreas de aplicação. É possível o processamento de vastas bases de dados, graças ao suporte ao processamento distribuído. O Khoros é considerado independente de plataforma, tendo sido portado para diversos modelos diferentes de estações de trabalho e computadores pessoais.

O Consórcio Khoros, responsável pelo desenvolvimento, distribui o software em regime de sistema aberto. As contribuições de pesquisadores e membros do Consórcio, atuando independentemente, são as responsáveis pelos melhoramentos e extensões do Khoros.

O site da Khoral Research na Internet contém maiores informações sobre o Khoros: "<http://www.khoral.com/>".

OPTIMAS

A empresa Optimas Corporation criou uma linguagem analítica para o processamento de imagens, baseada na sintaxe da linguagem C. O software OPTIMAS, de mesmo nome da empresa fabricante, permite a criação de macros específicas para cada caso de processamento e análise de imagens.

A biblioteca de funções disponíveis possui funções de conversão entre os formatos de arquivos IMG, GIF, PCX, PICT e TIFF, entre outros. Possui também funções de manipulação de histograma, limiarização, deteção de bordas, filtragem, morfologia matemática, transformada de Fourier bidimensional, operações de medida e coleta de dados estatísticos, etc. O OPTIMAS também permite que as operações sejam realizadas manualmente através de menus e ícones.

Na Internet, o endereço da Optimas Corporation é "<http://www.optimas.com/>". Uma versão demonstrativa do software e maiores informações sobre o mesmo podem ser obtidas no endereço "<http://www.optimas.com/opdesc.htm>".

SPRING

O Sistema de Processamento de Informações Geográficas (SPRING), desenvolvido pelo Instituto Nacional de Pesquisas Espaciais (INPE), é orientado ao processamento de imagens geográficas. Possui recursos para filtragem, operações aritméticas, transformação HSI-RGB, segmentação, classificação, estatística, restauração, eliminação de ruído, manipulação de histograma, etc.

Alguns dos objetivos do Projeto SPRING são: tornar amplamente acessível à comunidade brasileira um GIS de rápido aprendizado; fornecer um ambiente unificado de Geoprocessamento e Sensoriamento Remoto para aplicações urbanas e ambientais; produzir um sistema multiplataforma, para ambientes Windows, Linux, Solaris, SunOS, HP-UX, IRIX, DEC-OSF/1 e AIX; e dispor de uma biblioteca de classes em C++ que suporte o desenvolvimento de estudos e projetos em GIS.

Maiores detalhes sobre o SPRING podem ser obtidos através do endereço "<http://www.inpe.br/spring/>", na Internet.

VISILOG

O pacote VISILOG, desenvolvido pela empresa NOESIS, fornece uma vasta biblioteca de algoritmos para processamento de imagens. É um programa modular, composto de um núcleo e uma série de extensões opcionais.

O núcleo é estruturado em três principais sub-sistemas: o gerente de entrada e saída, a caixa de ferramentas para análise de imagens e a interface com o usuário. Escrito em linguagem C, numa abordagem orientada a objetos, o VISILOG foi concebido com ênfase na independência de plataforma de hardware. A caixa de ferramentas para análise de imagens suporta operações ponto a ponto, operações orientadas a vizinhança, operações geométricas, operações de medida, deteção de bordas, transformada de Fourier bidimensional, entre outras.

Algumas das extensões opcionais disponíveis são: o módulo de morfologia matemática, o módulo de segmentação de imagens, o módulo de reconhecimento de padrões, o módulo de reconhecimento de caracteres, o módulo de processamento de cores e outros módulos de processamento de imagens tridimensionais.

Aplicações típicas do VISILOG incluem: controle de qualidade (inspeção automática), metalografia, robótica, geração de imagens médicas e análise microscópica, sensoriamento remoto, etc.

Informações mais detalhadas sobre o Visilog estão disponíveis na Internet, no endereço "<http://www.noesisvision.com/prod01.htm>". O endereço da Noesis é "<http://www.noesisvision.com/>".

Na Internet

Os endereços mencionados ao longo do capítulo encontram-se agrupados a seguir.

["http://www.mindworkshop.com/alchemy/alchemy.html"](http://www.mindworkshop.com/alchemy/alchemy.html)

Alchemy Mindworks

MARQUES FILHO, Ogê; VIEIRA NETO, Hugo. *Processamento Digital de Imagens*, Rio de Janeiro: Brasport, 1999. ISBN 8574520098.

"<http://www.demon.co.uk/ffaltd/cimages.html>"
C_IMAGES

"<http://www.datx.com/>"
Data Translation

"<http://www.dadisp.com/>"
DSP Development Corporation

"<http://www.demon.co.uk/ffaltd/index.html>"
Foster Findlay Associates - Home Page

"<http://www.mindworkshop.com/alchemy/gifcon.html>"
GIF Construction Set for Windows

"http://www.datx.com/tech/global_lab_img.htm"
Global Lab Image

"<http://www.mindworkshop.com/alchemy/gww.html>"
Graphic Workshop for Windows

"<http://www.mathworks.com/>"
MathWorks

"<http://www.khoral.com/>"
Khoral Research

"<http://www.noesisvision.com/>"
Noesis

"<http://www.optimas.com/>"
Optimas Corporation

"<http://www.optimas.com/opdesc.htm>"
OPTIMAS

"<http://www.demon.co.uk/ffaltd/pcimage.html>"
PC_IMAGE

"<http://www.shareware.com/>"
Shareware

"<http://www.inpe.br/spring/>"
SPRING

"<http://www.ph.tn.tudelft.nl/Software/TimWin/timwww2.html>"
TIMWIN

"<http://www.noesisvision.com/prod01.htm>"

Visilog

Além dos *sites* acima, outros endereços de interesse são:

"<http://www.cs.uwa.edu.au/robvis/VIP.html>"

VIP

Descreve biblioteca de funções em C desenvolvida pela *University of Western Australia*.

"<http://www.vision1.com/cameras.html>"

CCD Cameras

Guia técnico-comercial de fabricantes e modelos de câmeras.

"<http://www.cmpcmm.com/cc/standards.html>"

Computer and Communication Standards

Descreve padrões de vídeo, como por exemplo o RS-170.

"<http://www.vision1.com/products.html>"

Machine Vision Product Information

Diretório de informações práticas sobre hardware e software para visão computacional.

"<http://www.tucows.com/>"

Welcome to TUCOWS

Repositório de software para *download*. Inclui diversos programas de visualização e manipulação de imagens.

Bibliografia

- | | |
|-------------------|---|
| [Dougherty 1994] | Dougherty, E.R. (ed.), <i>Digital Image Processing Methods</i> , Marcel Dekker, 1994. |
| [Grob 1989] | Grob, B., <i>Televisão e Sistemas de Vídeo</i> , Guanabara, 1989. |
| [Pearson 1991] | Pearson, D. (ed.), <i>Image Processing</i> , McGraw-Hill, 1991. |
| [Rubinstein 1988] | Rubinstein, R., <i>Digital typography</i> , Addison-Wesley, 1988. |
| [Schalkoff 1989] | Schalkoff, R.J., <i>Digital Image Processing and Computer Vision</i> , Wiley, 1989. |
| [Vellacott 1994] | Vellacott, O., "CMOS in Camera", <i>IEE Review</i> , Maio 1994. |

Apêndice A

Formatos de Arquivos de Imagens

Este apêndice contém informações técnicas básicas sobre os principais formatos de arquivos de imagens disponíveis atualmente. Seu principal objetivo é fornecer uma visão comparativa entre os vários formatos, tanto para arquivos de imagens estáticas (2-D ou 3-D) ou dinâmicas (animação e vídeo). Para o leitor interessado em maiores aprofundamentos em qualquer um destes formatos, são sugeridos bibliografia e endereços relevantes na Internet.

A.1 Representação através de *bitmaps* e através de vetores

Dois modos de representação básicos podem ser utilizados para compor imagens: (1) através de *bitmaps* (mapas de bits), também conhecidos como *pixel maps* (mapas de pixels) ou *raster* (varredura), e (2) através de vetores. Alguns formatos de arquivos de imagens podem empregar uma composição de ambos os métodos de representação, que diferem significativamente entre si.

A representação através de *bitmaps* constitui a forma mais simples de implementação e funciona para qualquer imagem, dentro de determinados limites. Os *bitmaps* nada mais são do que o conjunto de pixels da imagem digital, onde o valor de cada pixel representa as suas características de luminosidade e cor. Desse modo, os dados mapeiam a imagem, dando origem ao nome *bitmap*.

Bitmaps funcionam bem para imagens com variações complexas em suas formas e cores, tais como quadros (*frames*) de vídeo e fotografias digitalizadas. As imagens das telas de computadores são criadas no formato *bitmap* e portanto são mais facilmente gravadas do mesmo modo. As imagens *bitmap* também são adequadas para reprodução em impressoras, cujo método de formação de imagens é semelhante à varredura de um monitor de vídeo.

Na representação através de vetores, descreve-se uma imagem através dos parâmetros das formas geométricas que a compõem. Em outras palavras, os pontos, linhas, polígonos, círculos, elipses e demais formas geométricas complexas, preenchidas ou não, são representados através de parâmetros e coeficientes matemáticos. Normalmente os arquivos de imagens representados através de vetores se parecem com listagens de programas que contêm comandos e dados em formato ASCII. Por exemplo, um retângulo com vértice superior esquerdo situado em (10,20) e com vértice inferior direito situado em (60,50) no plano cartesiano, poderia ser hipoteticamente representado pelo comando RECTANGLE(10,20,60,50).

A representação através de vetores é adequada para imagens com predominância de linhas, constituídas de formas geométricas e preenchimentos simples, tais como diagramas, gráficos simples e desenhos gerados em programas CAD. Imagens representadas por vetores são particularmente úteis para reprodução em *plotters*, cujo processo de formação de imagens baseia-se no desenho de linhas contínuas.

As técnicas de representação por *bitmaps* e vetores podem ser combinadas, compondo o chamado *metafile* (meta-arquivo), no qual costuma predominar a informação vetorial em relação aos *bitmaps*.

A.1.1 Comparações entre as formas de representação

Bitmaps podem representar qualquer tipo de imagem, uma vez que toda imagem pode ser digitalizada. No entanto, imagens do tipo *bitmap* apresentam alguns problemas.

Um dos problemas práticos existentes é o tamanho da imagem, que pode demandar vários megabytes para armazenamento e processamento, no caso de uma imagem colorida de alta resolução. É por esse motivo que as técnicas de compressão de dados (ver capítulo 6) são importantes na representação de imagens através de *bitmaps*. Outro problema de ordem prática é MARQUES FILHO, Ogé; VIEIRA NETO, Hugo. *Processamento Digital de Imagens*, Rio de Janeiro: Brasport, 1999. ISBN 8574520098.

o alto poder de processamento requerido para manipular imagens do tipo *bitmap*. A resolução fixa é também um problema, resultando em qualidade visual inferior sempre que se procura ampliar a imagem original e em perda de resolução quando se armazena uma versão reduzida em tamanho.

A representação através de vetores possui maior limitação no que tange ao que pode ser representado efetivamente em relação ao que pode ser representado por *bitmaps*. Por exemplo, embora muito adequada para representar um projeto arquitetônico, ela não é apropriada para representar uma foto digitalizada. Porém, existe uma maior flexibilidade quanto à resolução obtida e também quanto à manipulação das formas geométricas da imagem, as quais podem ser tratadas como objetos independentes.

A.1.2 Outras classes de representação

Existem ainda métodos de representar imagens tridimensionais que fazem uso de modelos matemáticos complexos. Esses modelos incluem informações sobre fontes de luz, câmeras e objetos da cena.

Para aplicações em multimídia, há formatos de animação e vídeo, capazes de armazenar uma seqüência de imagens. A diferença básica entre os dois é que o formato de vídeo armazena uma trilha sonora juntamente com a seqüência de imagens.

A.2 Formatos de Arquivos de Imagem

A.2.1 Arquivos de Imagens 2-D

BMP / DIB

- Nome: Microsoft Windows Device Independent Bitmap.
- Proprietário: Microsoft Corporation.
- Tipo de Arquivo: Bitmap.
- Características: Suporta cores com até 24 bits. Cores com até 8 bits são armazenadas na forma de mapa de cores. Pode ser compressão RLE ou nenhuma compressão.
- Plataformas: PC e Macintosh.
- Aplicações: Armazenamento de imagens para uso no Microsoft Windows.
- Vantagens: Bem suportado no Microsoft Windows.
- Desvantagens: Pouco suportado em outros sistemas.

CGM

- Nome: Computer Graphics Metafile.
- Proprietário: American National Standards Institute.
- Tipo de Arquivo: Metafile.
- Características: Possui 3 diferentes codificações: binária de 8 bits, binária de 16 bits e texto. Consiste numa seqüência de comandos gráficos.
- Plataformas: PC e estações de trabalho UNIX.
- Aplicações: Armazenamento e troca de imagens.
- Vantagens: É o único padrão gráfico oficial até o momento.
- Desvantagens: Difícil de ser implementado e validado, possuindo 3 codificações incompatíveis entre si.

DXF

- Nome: Drawing Interchange Format.
- Proprietário: Autodesk, Inc.
- Tipo de Arquivo: Vetor binário e ASCII (imagens bi- e tridimensionais).
- Características: Constitui mais uma linguagem gráfica que um formato de imagem propriamente dito. É capaz de representar modelos tridimensionais.
- Plataformas: PC, Macintosh e estações de trabalho UNIX.
- Aplicações: Projeto Assistido por Computador (CAD).
- Vantagens: Largamente suportado em aplicativos CAD.
- Desvantagens: Apresenta a vantagem de descrever vetores tridimensionais.
É ineficiente para armazenamento.
Implementar um leitor completo para o formato DXF requer muito esforço, pois o mesmo deve ser capaz de desenhar e manipular fontes e formas geométricas complexas, e ainda representar bidimensionalmente formas tridimensionais.

FIF

- Nome: Fractal Image Format
- Proprietário: Iterated Systems
- Tipo de Arquivo: Bitmap.
- Características: Suporta cores com até 24 bits. Utiliza técnica de compressão de dados baseada em fractais, a qual possibilita que a taxa de compressão seja estipulada pelo usuário.
- Aplicações: Artes gráficas e editoração eletrônica.
- Vantagens: Apresenta altas taxas de compressão sem degradação notável na imagem.
- Desvantagens: O método de compressão é matematicamente complexo e geralmente exige hardware especial.
A documentação técnica do formato não está disponível publicamente.

FITS

- Nome: Flexible Image Transport System.
- Proprietário: Grupo de Trabalho da Comissão 5 da União Internacional de Astronomia.
- Tipo de Arquivo: Bitmap.
- Características: Os dados do formato FITS básico consistem normalmente em matrizes de dimensão N. Em arquivos que contêm imagens, essa matriz é geralmente bidimensional (imagem em níveis de cinza) ou tridimensional (conjunto de imagens em níveis de cinza).
- Plataformas: Estações de trabalho UNIX e PC.
- Aplicações: Armazenamento e troca de imagens astronômicas.
- Vantagens: Permite a inclusão de dados descritivos sobre a imagem.
É um formato bastante portátil e bem padronizado.
- Desvantagens: Extremamente orientado para aplicações astronômicas.
Não utiliza nenhuma técnica de compressão de dados.

GIF

- Nome: Graphics Interchange Format.
- Proprietário: CompuServe, Inc.
- Tipo de Arquivo: Bitmap.
- Características: Suporta cores de até 24 bits numa palheta de até 256 cores em imagens de até 65536 por 65536 pixels. Utiliza compressão de dados pela técnica LZW. Permite o armazenamento de múltiplas imagens num mesmo arquivo, possibilitando animações.
- Plataformas: A maioria dos computadores pessoais e algumas estações de trabalho UNIX.
- Aplicações: Artes gráficas, editoração eletrônica. Apresentação de imagens na Internet.
- Vantagens: É um formato excelente para troca de dados entre diferentes plataformas com boas taxas de compressão.
A sua popularidade é ainda mais aumentada graças ao seu uso como formato padrão de imagens utilizado na Internet, juntamente com o padrão JPEG, e à distribuição gratuita da sua documentação pela CompuServe.
- Desvantagens: Não apresenta possibilidade de armazenamento de tabelas de tons de cinza nem de correção de cor.
Também não possibilita representação dos dados nos modelos CMYK e HSI.
Até o momento é possível armazenar somente uma palheta de 256 cores de 24 bits.

HPGL

- Nome: Hewlett-Packard Graphics Language.
- Proprietário: Hewlett-Packard Co.
- Tipo de Arquivo: Vetor.
- Características: Constitui a linguagem de comandos para os plotters HP. Consiste quase que completamente de caracteres ASCII, tornando-se fácil de produzir e corrigir.
- Plataformas: Plotters HP e compatíveis e impressoras a laser.
- Aplicações: Controle de plotters e atualmente impressoras a laser.
- Vantagens: Amplamente utilizado.
Independente do tamanho do papel, porém imagens muito grandes podem exigir que o desenho seja feito em partes.
- Desvantagens: Como um formato de imagem constitui um nível muito baixo de representação.

IFF

- Nome: IFF Interleaved Bitmap (ILBM).
- Proprietário: Electronic Arts, Inc.
- Tipo de Arquivo: Bitmap.
- Características: Tipicamente as imagens no formato ILBM apresentam largura de 320 ou 640 pixels (modos de vídeo do Amiga). As imagens com largura de 640 pixels suportam cores com 4 bits e as imagens com 320 pixels de largura suportam cores com 5 bits. Cores com 6 bits são suportadas em modos de vídeo exclusivos do Amiga. No entanto, os mapas de cor são armazenados em 8 bits. Também é possível armazenar informações das coordenadas de um ponto principal, caso a imagem seja de um cursor. Utiliza compressão pela técnica RLE, ou nenhuma compressão.
- Plataformas: Amiga e, restritamente, Macintosh e PC.
- Aplicações: Multimídia.
- Vantagens: É um formato bem padronizado e extensível.
Oferece possibilidade de uso de características exclusivas do hardware de vídeo do Amiga.
- Desvantagens: Pelo fato de ser extensível, podem existir extensões incompatíveis entre si.
Oferece pouca compressão de dados.

IMG

- Nome: GEM IMG.
- Proprietário: Originalmente, Digital Research. Atualmente, Novell.
- Tipo de Arquivo: Bitmap.
- Características: Suporta imagens monocromáticas, em tons de cinza e a cores. Imagens coloridas são armazenadas em 4 planos diferentes (R, G, B e W). Utiliza método de compressão em blocos.
- Plataformas: Atari e PC.
- Aplicações: Artes gráficas e editoração gráfica em ambiente gráfico GEM.
- Vantagens: É suportado pelos aplicativos do ambiente gráfico GEM.
- Desvantagens: Pouca compressão, sem possibilidade de uso de mapas de cor.
Pouca documentação disponível.

JPEG

- Nome: JPEG.
- Proprietário: Joint Photographic Experts Group.
- Tipo de Arquivo: Bitmap comprimido.
- Características: Permite o uso de diversas técnicas de compressão, sendo que a maioria delas apresenta perdas, o que significa que a imagem original não será exatamente idêntica à imagem recuperada após a descompressão. A taxa de compressão pode ser determinada pelo usuário. Suporta cores com até 24 bits.
- Plataformas: Macintosh, PC e estações de trabalho UNIX.
- Aplicações: Armazenamento digital de fotografias. Apresentação de imagens na Internet.
- Vantagens: Oferece a maior taxa de compressão existente para imagens fotográficas. Ao lado do padrão GIF, constitui um dos padrões para arquivos de imagens apresentadas na Internet.
Permite compressão através de hardware específico.
- Desvantagens: O padrão ainda está em desenvolvimento e existem algumas opções incompatíveis entre si.
A compressão e descompressão por software é um tanto lenta.

MAC

- Nome: MacPaint (PNTG).
- Proprietário: Apple Computer, Inc.
- Tipo de Arquivo: Bitmap binário.
- Características: Suporta apenas imagens binárias (preto e branco) em apenas um único tamanho de 576 por 720 pixels. Utiliza compressão do tipo PackBits.
- Plataformas: Macintosh e PC.
- Aplicações: Uso geral em aplicativos Macintosh e em alguns aplicativos PC.
- Vantagens: É amplamente suportado por aplicativos Macintosh.
É compacto e simples de ser implementado.
- Desvantagens: Extremamente limitado quanto ao tamanho e ao número de tons da imagem (permite escala de cinzas apenas através da técnica de dithering).

MSP

- Nome: Microsoft Paint.
- Proprietário: Microsoft Corporation.
- Tipo de Arquivo: Bitmap.
- Características: Suporta apenas bitmaps monocromáticos (preto e branco). Utiliza a técnica de compressão RLE.
- Plataformas: PC.
- Aplicações: Artes gráficas e editoração eletrônica.
- Vantagens: É um formato compacto.
- Desvantagens: É muito limitado e pouco suportado.

PBM

- Nome: Portable Bitmap Utilities.
- Proprietário: Jef Poskanzer (autor do aplicativo PBM Utilities).
- Tipo de Arquivo: Bitmap.
- Características: Possui três subformatos: Portable Bitmap (para bitmaps monocromáticos), Portable Gray Map (para bitmaps em tons de cinza) e Portable Pixel Map (para bitmaps em cores). Cada subformato pode assumir duas variantes: codificação binária e codificação em código ASCII.
- Plataformas: Estações de trabalho UNIX e PC.
- Aplicações: Conversão de formatos de arquivos de imagem.
- Vantagens: Esse formato é simples de ser escrito e lido, através de codificação por texto.
- Desvantagens: Resulta em arquivos muito grandes para servir como formato de armazenamento.
As versões binárias suportam apenas cores com até 8 bits. É basicamente suportado apenas em UNIX.

PCD

- Nome: Kodak Photo CD.
- Proprietário: Kodak.
- Tipo de Arquivo: Bitmap.
- Características: Suporta cores com até 24 bits, utilizando um modelo especial para reprodução e impressão em alta qualidade. A técnica de compressão de dados utilizada é proprietária.
- Plataformas: Macintosh, PC e estações de trabalho UNIX.
- Aplicações: Armazenamento de imagens fotográficas em CD.
- Vantagens: Possui amplo suporte em programas de editoração.
- Desvantagens: O formato pode ser gerado somente por equipamentos profissionais da Kodak.

PCX

- Nome: PCX.
- Proprietário: Zsoft Corporation.
- Tipo de Arquivo: Bitmap.
- Características: Suporta cores com até 24 bits em imagens de até 65536 por 65536 pixels. Pode utilizar compressão de dados pela técnica RLE ou nenhuma compressão.
- Plataformas: PC e Macintosh.
- Aplicações: Artes gráficas e editoração eletrônica.
- Vantagens: É um dos formatos mais antigos e portanto é suportado pela maioria dos aplicativos PC.
- Desvantagens: Não apresenta possibilidade de armazenamento de tabelas de tons de cinza nem de correção de cor.
Também não possibilita representação dos dados nos modelos CMYK e HSI.
A técnica RLE de compressão de dados não é muito eficiente para imagens complexas, tais como fotos.
Devido às muitas implementações possíveis, alguns aplicativos não são capazes de ler todas os tipos existentes.

PCL

- Nome: Hewlett-Packard Printer Control Language.
- Proprietário: Hewlett-Packard Co.
- Tipo de Arquivo: Seqüência de comandos de impressora a laser.
- Características: É constituído de comandos de impressão, possibilitando a inclusão de imagens bitmap monocromáticas. Utiliza as técnicas de compressão RLE, PackBits, modulação Delta entre linhas e adaptativa, ou ainda, nenhuma compressão.
- Plataformas: Impressoras a laser HP e compatíveis.
- Aplicações: Controle de impressoras a laser.
- Vantagens: Constitui a linguagem de comandos para impressoras a laser mais suportada e uma das mais compactas para imagens bitmap monocromáticas.
- Desvantagens: Suporta apenas imagens monocromáticas.
Recuperar a imagem armazenada significa na maioria das vezes simular o modelo de geração de imagens da impressora.

PCT

- Nome: QuickDraw Picture Format (PICT).
- Proprietário: Apple Computer, Inc.
- Tipo de Arquivo: Metafile (linguagem binária de descrição de página).
- Características: É constituído de uma seqüência de comandos gráficos, os quais podem conter dados vetoriais ou imagens bitmap. Suporta apenas bitmaps monocromáticos de até 32KB com resolução fixa de 72 dpi na sua versão 1 (QuickDraw). Suporta até 256 cores utilizando uma paleta de 48 bits na sua versão 2 (Color QuickDraw). Não permite armazenamento de informações de correção gama.
- Plataformas: Primeiramente Macintosh, atualmente também PC e estações de trabalho UNIX.
- Aplicações: Formato de apresentação de gráficos QuickDraw no Macintosh.
- Vantagens: Constitui um dos formatos gráficos mais suportados no Macintosh. Bitmaps monocromáticos são armazenados através da eficiente técnica de compressão PackBits.
- Desvantagens: Apesar de permitir maior profundidade de cores, está limitado ao sistema Color QuickDraw, que lê e escreve os arquivos PICT no Macintosh.

PIC

- Nome: PIC.
- Proprietário: Lotus Development Corp.
- Tipo de Arquivo: Vetor.
- Características: Seqüência de comandos gráficos.
- Plataformas: PC, estações de trabalho.
- Aplicações: Arquivo intermediário entre a planilha de cálculo Lotus 1-2-3 e aplicativos de impressão gráfica.
- Vantagens: Simples de ser gerado e lido.
- Desvantagens: Muito inflexível.

PS

- Nome: PostScript.
- Proprietário: Adobe Systems, Inc.
- Tipo de Arquivo: Metafile (linguagem de descrição de página).
- Características: Suporta cores com até 36 bits. Permite padronização e correção de cores, imagens dos tipos bitmap e vetor, fontes do tipo vetor e transformações lineares em imagens. Pode ser armazenado em ASCII ou dados binários. Possui 4 variantes: Level 1, Level 2, Encapsulated e Display PostScript.
- Plataformas: Primeiramente Macintosh, atualmente também PC e estações de trabalho UNIX.
- Aplicações: Edição eletrônica.
- Vantagens: Constitui o padrão absoluto para edição eletrônica.
- Desvantagens: É geralmente armazenado em ASCII, fato que torna arquivos de imagens bitmap grandes e a sua leitura e apresentação um tanto lenta.
Constitui um formato de difícil interpretação.

PSD

- Nome: Adobe Photoshop.
- Proprietário: Adobe Systems, Inc.
- Tipo de Arquivo: Bitmap.
- Características: Suporta cores com até 24 bits. Suporta camadas de cores e canais Alfa. Utiliza a técnica de compressão de dados RLE.
- Plataformas: Macintosh e PC.
- Aplicações: Artes gráficas e edição eletrônica.
- Vantagens: É um formato popular e amplamente suportado, pelo fato de possibilitar o uso de múltiplas camadas de cores e canais Alfa.
- Desvantagens: A técnica RLE não oferece grandes taxas de compressão.

RAS

- Nome: Sun Rasterfiles.
- Proprietário: Sun Microsystems.
- Tipo de Arquivo: Bitmap.
- Características: Suporta cores com até 24 bits ou mapas de cores. As formas de compressão utilizadas são RLE ou nenhuma compressão, podendo comportar os formatos TIFF ou IFF.
- Plataformas: Estações de trabalho Sun.
- Aplicações: Armazenamento de imagens.
- Vantagens: Bem suportado nas estações de trabalho Sun.
- Desvantagens: Pouco suportado em outros sistemas.

TGA

- Nome: Targa.
- Proprietário: Truevision, Inc.
- Tipo de Arquivo: Bitmap.
- Características: Suporta cores com até 32 bits, com ou sem mapa e tabela de correção de cores. Utiliza a técnica RLE de compressão de dados ou nenhuma compressão.
- Plataformas: PC e Macintosh.
- Aplicações: Captura de imagens de vídeo.
- Vantagens: É um formato que permite diversas anotações sobre a imagem.
- Desvantagens: Possui muitos subformatos, nem todos suportados por todos os aplicativos.

TIFF

- Nome: Tag Image File Format (TIFF).
- Proprietário: Aldus Corporation.
- Tipo de Arquivo: Bitmap.
- Características: Suporta cores com até 48 bits ou uma paleta de 65536 cores. Permite dados de transparência e opacidade. O tipo de compressão utilizado varia com a versão do formato (RLE, LZW, PackBits, Huffmann Modificada, Fac-símile Grupos 3 e 4 ou nenhuma).
- Plataformas: Macintosh, PC e estações de trabalho UNIX.
- Aplicações: Artes gráficas, editoração eletrônica.
- Vantagens: É suportado por diversas plataformas de hardware, sendo especialmente útil para troca de dados entre plataformas diferentes.
É um formato adequado para vários tipos de aplicação e é muito bem documentado.
Apresenta boas taxas de compressão.
- Desvantagens: A versatilidade do TIFF promove algumas dificuldades, devidas às inúmeras possibilidades de criação de extensões do formato.
No entanto, a versão 6.0 do formato TIFF especifica uma linha mestra de capacidades, visando melhorar sua funcionalidade para troca de dados entre aplicativos.

UNIX Plot Format

- Nome: UNIX Plot Format.
- Proprietário: UNIX System Labs.
- Tipo de Arquivo: Vetor.
- Características: Seqüência de comandos gráficos.
- Plataformas: Estações de trabalho UNIX.
- Aplicações: Formato comum para aplicativos de desenho para o sistema UNIX.
- Vantagens: Suporte universal de baixo nível em sistemas baseados em UNIX.
- Desvantagens: Muito baixo nível, suporte limitado.
Apresenta problemas quanto a ordem de armazenamento de bytes em computadores incompatíveis entre si.

WMF

- Nome: Microsoft Windows Metafile.
- Proprietário: Microsoft Corporation.
- Tipo de Arquivo: Lista de funções.
- Características: Armazena uma lista de chamadas a funções gráficas do Microsoft Windows. Cada chamada contém um tamanho, um número de função e alguns argumentos. Muitas das chamadas possuem uma referência de cor como argumento, possibilitando tanto cores com 24 bits quanto mapas de cores.
- Plataformas: PC.
- Aplicações: Armazenamento e troca de imagens entre aplicativos no Microsoft Windows.
- Vantagens: Possibilita tamanhos de arquivos bem menores que os correspondentes bitmaps, devido a descrições de características de alto nível.
É um formato de arquivo bem estruturado.
- Desvantagens: Fortemente relacionado ao modelo de geração de imagens do Microsoft Windows.
Os arquivos são relativamente complexos.

WPG

- Nome: WordPerfect Graphics.
- Proprietário: Originalmente, WordPerfect. Atualmente, Novell.
- Tipo de Arquivo: Metafile.
- Características: Pode armazenar tanto bitmaps quanto seqüências de comandos gráficos. Não suporta mapas de cores e requer que as primeiras 16 cores da palheta utilizada sejam as cores do padrão VGA. As demais 16 cores da palheta deverão ser tons de cinza, partindo do preto até o branco. Utiliza compressão de dados do estilo PackBits.
- Plataformas: Macintosh, PC e estações de trabalho UNIX.
- Aplicações: Edição eletrônica.
- Vantagens: Bem suportado no WordPerfect e aplicativos correlatos.
- Desvantagens: Pouco suportado em outros aplicativos.

XBM

- Nome: X Window Bitmaps.
- Proprietário: MIT X Consortium.
- Tipo de Arquivo: Bitmap monocromático.
- Características: É constituído de código fonte em C, a ser compilado por aplicativos do sistema X Windows. Prevê o armazenamento de um ponto central da imagem, caso esta seja referente a um cursor.
- Plataformas: Estações de trabalho UNIX.
- Aplicações: Cursores e ícones para o sistema X Windows.
- Vantagens: É lido diretamente pelo sistema X Windows.
- Desvantagens: É específico apenas ao sistema X Windows.

XWD

- Nome: X Window Dump.
- Proprietário: MIT X Consortium.
- Tipo de Arquivo: Bitmap.
- Características: Permite um grande número de subformatos. Suporta diversas configurações de cores, através de mapas ou não. Não utiliza compressão de dados.
- Plataformas: Estações de trabalho UNIX.
- Aplicações: Armazenamento de imagens no sistema X Windows.
- Vantagens: É suportado por muitos aplicativos do sistema X Windows.
- Desvantagens: Não possui suporte significativo fora do sistema X Windows.
Gera arquivos pouco eficientes em termos de tamanho.

A.2.2 Arquivos de Imagens 3-D**3DS**

- Nome: 3D Studio.
- Proprietário: Autodesk, Inc.
- Tipo de Arquivo: Modelo de imagens tridimensionais.
- Características: Suporta modelos tridimensionais, normas e atributos de superfície e animação.
- Plataformas: PC.
- Aplicações: Modelagem e animação tridimensionais.
- Vantagens: É suportado por diversos aplicativos na plataforma PC.
- Desvantagens: É um formato restrito à plataforma PC.

OBJ

- Nome: Wavefront Object.
- Proprietário: Wavefront Technologies.
- Tipo de Arquivo: Modelo de imagens tridimensionais.
- Características: Suporta modelos tridimensionais, normas e atributos de superfície e animação. não utiliza nenhuma técnica de compressão. Possui dois subformatos: texto e binário.
- Aplicações: Modelagem e animação tridimensionais.
- Vantagens: O formato texto possui estrutura aberta.
- Desvantagens: O formato binário é proprietário do fabricante.

POV

- Nome: POV Raytracer.
- Proprietário: Persistence of Vision.
- Tipo de Arquivo: Modelo de imagens tridimensionais.
- Características: Suporta modelos tridimensionais, normas e atributos de superfície e animação. não utiliza nenhuma técnica de compressão..
- Aplicações: Modelagem e animação tridimensionais.
- Vantagens: Consiste numa linguagem de descrição simples baseada em cenas tridimensionais.
- Desvantagens: É necessário conhecimento de programação para se escrever um arquivo de descrição de cena.

RIB

- Nome: Renderman Interface Bytestream.
- Proprietário: Pixar.
- Tipo de Arquivo: Descrição de cena.
- Características: Consiste numa linguagem altamente especializada para representação em alta qualidade, tipicamente em formato ASCII, mas também possivelmente em formato binário. Aceita modelos tridimensionais de dados, incluindo informações sobre a câmera, luzes e outras variáveis como opacidade dos objetos. Opcionalmente pode ser dividido em dois ou mais quadros, permitindo a representação de seqüências de animação.
- Plataformas: Macintosh, PC e estações de trabalho UNIX.
- Aplicações: Representação de cenas tridimensionais, contendo informações sobre a câmera, luzes e demais componentes.
- Vantagens: É um dos formatos mais sofisticados, capaz de criar imagens bidimensionais realísticas a partir de informações tridimensionais de uma cena.
É considerado o maior passo dado em direção a um padrão da indústria para esse fim.
- Desvantagens: Leitores do formato RIB são aplicativos muito complexos de serem desenvolvidos.
Não é um formato desenvolvido para atender a outras necessidades mais simples, como armazenar modelos tridimensionais de dados.
Não possibilita a inclusão de imagens bidimensionais do tipo bitmap ou vetor.

A.2.3 Arquivos de Animação e Vídeo

AVI

- Nome: Video for Windows.
- Proprietário: Microsoft Corporation.
- Tipo de Arquivo: Bitmap.
- Características: Suporta cores com até 24 bits. Possibilita taxas de compressão variáveis.
- Plataformas: Principalmente PC, ocasionalmente Macintosh.
- Aplicações: Vídeo digital.
- Vantagens: Proporciona vídeo e áudio de alta qualidade
- Desvantagens: A qualidade de vídeo oferecida não é tão alta quanto a do formato QuickTime.

FLI / FLC

- Nome: Autodesk “Flick” Formats.
- Proprietário: Autodesk, Inc.
- Tipo de Arquivo: Bitmap.
- Características: O formato FLI suporta imagens com apenas 64 cores, tamanho de até 320 por 240 pixels e até 4000 quadros por arquivo. O formato FLC permite imagens com tamanho de até 1280 por 1024 pixels, com até 256 cores. Utilizam compressão RLE (orientada a byte no formato FLI e orientada a word no formato FLC) e diferenciação quadro a quadro. O primeiro quadro é armazenado integralmente na forma de bitmap comprimido. Os quadros seguintes contêm apenas informações sobre os pixels que diferem do quadro anterior.
- Plataformas: PC.
- Aplicações: Animação de imagens bitmap.
- Vantagens: As técnicas de compressão empregadas são simples de ser decodificadas e codificadas, sendo adequadas para animações breves.
- Desvantagens: Um número crescente de aplicativos vem suportando o formato.
- Desvantagens: Ambos os formatos não suportam áudio.
- Desvantagens: A palheta suporta apenas 256 cores.
- Desvantagens: A compressão RLE não é indicada para imagens que fazem uso da técnica de *dithering*.

MPEG

- Nome: MPEG.
- Proprietário: Moving Pictures Expert Group - International Organization for Standardization (ISO).
- Tipo de Arquivo: Bitmap em movimento.
- Características: Utiliza compressão DCT baseada em blocos e compressão intra-quadros. Possui dois subformatos: MPEG-1 e MPG-2, sendo que o MPEG-1 constitui um subconjunto do MPEG-2.
- Plataformas: PC, Macintosh e estações de trabalho UNIX.
- Aplicações: Compressão e descompressão de vídeo em tempo real, com áudio síncrono, para aplicações multimídia.
- Vantagens: Constitui o padrão da indústria, capaz de reduzir capacidades de armazenamento e transmissão de vídeo de alta qualidade.
- Desvantagens: É um formato muito complexo, geralmente implementado em hardware somente para aplicações em tempo real.
- Desvantagens: Requer grande poder de processamento quando realizado por software.

QT

- Nome: QuickTime Animation Format.
- Proprietário: Apple Computer, Inc.
- Tipo de Arquivo: Bitmap.
- Características: Utiliza compressão RLE orientada a byte e a word, combinada com compressão por diferenciação de quadros.
- Plataformas: Principalmente Macintosh, ocasionalmente PC e estações de trabalho UNIX..
- Aplicações: Animação de imagens bitmap.
- Vantagens: Muito bem suportado na plataforma Macintosh.
Constitui um formato bastante compacto.
- Desvantagens: A compressão oferecida não é tão eficiente quanto a MPEG.

Leitura complementar

Existem diversos livros inteiramente dedicados a formatos de arquivos de imagens, dentre os quais destacamos [Kay e Levine 1994].

O capítulo 6 de [Lindley 1991] trata dos formatos PCX e TIFF, incluindo código-fonte em C para sua manipulação (abertura, exibição e gravação de imagens nestes formatos). O apêndice 2 deste livro transcreve a especificação completa do formato TIFF versão 5.0.

Os artigos de Furht, [Furht 1995a] e [Furht 1995b], descrevem em detalhes os padrões JPEG e MPEG, respectivamente.

Na Internet

"<http://www.mindworkshop.com/alchemy/alchemy.html>"

Alchemy Mindworks

Home-page dos criadores do *shareware* utilitário Graphic Workshop.

"http://www.cc.iastate.edu/olc_answers/packages/graphics/file.formats.faq.html"

Graphic File Formats FAQ

Questões normalmente levantadas sobre formatos de arquivos de imagens, divididas em quatro partes: questões gerais sobre formatos de arquivos; programas para visualização e conversão de formatos de arquivos; onde obter especificações de formatos de arquivos; e dicas e truques.

"<http://www.deakin.edu.au/~agoodman/scc308/topic7.html>"

Topic 7: File formats and image compression

Capítulo de tutorial *on-line* dedicado a formatos de arquivos de imagem e algumas técnicas de compressão utilizadas nestes formatos.

"<http://www.cs.sfu.ca/undergrad/CourseMaterials/CMPT479/material/notes/Chap3/Chap3.2/Chap3.2.html>"

Graphic/Image File Formats

Capítulo de curso online sobre Sistemas Multimídia mantido pela *Simon Fraser University* (Canadá).

"<http://member.aol.com/royalef/gifabout.htm>"

All About GIF89a

Detalha o formato GIF89a.

Bibliografia

- [Furht 1995a] Furht, B., "A survey of multimedia compression techniques and standards – part I: JPEG standard.", *Real-Time Imaging Journal*, 1, 1, 1995, pp. 49-67.
- [Furht 1995b] Furht, B., "A survey of multimedia compression techniques and standards – part II: video compression.", *Real-Time Imaging Journal*, 1, 5, 1995, pp. 319-337.
- [Kay e Levine 1994] Kay, D. e Levine, J., *Graphics File Formats - 2nd ed.*, Windcrest / McGraw-Hill, 1994.
- [Lindley 1991] Lindley, C.A., *Practical Image Processing in C*, Wiley, 1991.

Apêndice B

Roteiros de Laboratório de Processamento de Imagens

Este apêndice tem por objetivos apresentar alguns aspectos fundamentais do MATLAB e sua *toolbox* (caixa de ferramentas) para processamento de imagens. Inicialmente, são apresentados alguns conceitos introdutórios sobre o ambiente MATLAB, especialmente compilados para o leitor sem contato prévio com este software. Na sequência, apresentamos um resumo das principais funções disponíveis na *toolbox* de processamento de imagens. Finalmente, sugerimos sete roteiros de práticas de laboratório utilizando o MATLAB e sua *toolbox*, recomendados ao longo dos capítulos 2 a 5 deste livro.

B.1. Conceitos Introdutórios

O MATLAB (abreviação de 'laboratório de matrizes' - *MATrix LABoratory*) é um sistema para cálculos matemáticos e matriciais, o qual pode ser imaginado como uma espécie de linguagem de programação. Todas as variáveis são tratadas como matrizes pelo MATLAB, com uma característica especial: são dimensionadas automaticamente, fato que facilita sobremaneira a implementação de algoritmos matriciais. Outra vantagem do uso do MATLAB é o seu extenso conjunto de rotinas de representação gráfica.

É possível a criação de programas com as funções do MATLAB para implementar algoritmos mais complexos. Esses programas são conhecidos como arquivos-M ou *scripts*.

Neste apêndice procuraremos introduzir os conceitos básicos de utilização do MATLAB, orientados ao processamento digital de imagens. É sempre adequada a advertência de que muitos outros aspectos úteis e importantes do MATLAB não serão abordados aqui.

B.2. Utilizando o MATLAB

Normalmente o MATLAB é utilizado no modo comando, ou seja, os comandos são processados imediatamente após a sua entrada, exibindo os resultados na tela. Porém, também é possível a criação de seqüências de comandos (*scripts*) armazenadas em arquivos denominados arquivos-M, como já havia sido antecipado. Essa possibilidade é bastante útil para seqüências de comandos comumente repetidas e também para a criação de novas funções específicas.

Uma característica bastante útil e prática do MATLAB é a de que as suas variáveis não precisam ser dimensionadas antes de serem usadas. As variáveis são geradas e dimensionadas automaticamente ao serem referenciadas pela primeira vez em uma atribuição de valores, permanecendo na memória de trabalho até que esta seja limpa.

Para limpar integralmente a memória de trabalho é utilizado o comando `clear` ou `clear all`. Para apagar apenas uma variável, utiliza-se o comando em conjunto com o nome da variável, da seguinte forma: `clear <nome-da-variável>`. Para se obter uma listagem das variáveis existentes em memória, utiliza-se o comando `whos`.

As variáveis presentes na memória de trabalho podem ser armazenadas em disco utilizando-se o comando `save` e posteriormente é possível recuperá-las através do comando `load`. O comando `save` também pode ser usado para gravar apenas algumas variáveis específicas.

Algumas variáveis do MATLAB possuem papel ou valores específicos. A variável `ans` contém o resultado da última operação realizada sem atribuição a nenhuma variável. As variáveis `i` e `j` contêm a unidade imaginária (raiz quadrada de -1). O valor de infinito (∞) é armazenado na variável `inf`, enquanto o valor de pi (π) é armazenado na variável `pi`. Ao ocorrer

um erro como uma divisão por zero, por exemplo, o resultado é do tipo NaN (não-número - *not-a-number*).

O formato de apresentação dos dados pelo MATLAB é determinado pelo comando `format <tipo>`: `format long` apresenta os números reais com 15 dígitos; `format long e` apresenta os números reais com 15 dígitos em notação científica; `format short` apresenta os números reais com 5 dígitos (formato padrão do MATLAB); e `format short e` apresenta os números reais com 5 dígitos em notação científica.

O MATLAB suporta extensamente representações gráficas. A limpeza da janela gráfica é realizada pelo comando `clc` e para se traçar um gráfico linear em coordenadas cartesianas, é utilizado o comando `plot`. O comando `hold on` mantém o gráfico corrente na janela gráfica (geralmente utilizado para a superposição de gráficos). Pode-se também fazer uso do comando `grid` para se desenhar uma grade reticulada no gráfico em questão. Os comandos `xlabel` e `ylabel` são usados para definir os rótulos dos eixos x e y, respectivamente. O título de um gráfico é definido a partir do comando `title` e os seus eixos a partir do comando `axis`.

Existem ainda alguns outros comandos e funções de uso geral. O comando `clock` permite a obtenção do ano, mês, dia, hora, minuto e segundo do sistema, na forma de um vetor contendo valores decimais. A obtenção somente da data é possível através do comando `date`. A função `computer` retorna o tipo de computador no qual o MATLAB está sendo executado, sendo útil para programas (*scripts*) que devam ser executados de maneira distinta em diferentes plataformas.

Para sair do ambiente do MATLAB, utiliza-se o comando `exit` ou o comando `quit`.

Alguns dos principais operadores e caracteres especiais do MATLAB são dados abaixo:

Operadores matriciais:

- + Adição
- Subtração
- *
- Multiplicação
- ^ Potenciação
- ' Transposta conjugada

Operadores relacionais:

- < Menor que
- <= Menor ou igual a
- > Maior que
- >= Maior ou igual a
- == Igual
- ~= Diferente

Operadores lógicos:

- & AND
- | OR
- ~ NOT

Caracteres especiais:

- [e] Formação de vetores e matrizes
- (e) Determinação da precedência de operadores em expressões lógicas e aritméticas
- ,
- Separação de subscritos e argumentos de funções
- ;
- Encerramento de linhas e supressão da impressão de resultados
- :
- Subscrição de conjuntos e geração de vetores ordenados
- !
- Execução de comandos do sistema operacional

% Introdução de comentários

Quanto aos caracteres especiais, enfatizamos o uso de três deles:

O caractere % é utilizado para inserir comentários nos programas, para sua maior clareza e entendimento. Portanto, linhas de programa precedidas pelo caractere % serão ignoradas pelo MATLAB.

O ponto-e-vírgula (;) é usado para suprimir a apresentação de resultados na tela. Se o último caractere de uma linha de comando for um ponto-e-vírgula, o comando será apenas executado e não apresentará o resultado na tela. Essa característica é especialmente útil dentro de programas, onde nem sempre os resultados intermediários são de interesse. Outro papel desempenhado pelo ponto-e-vírgula é o de separar linhas de elementos dentro de matrizes. Quando é dada a entrada de uma matriz no MATLAB, o ponto-e-vírgula delimita o final de cada linha de elementos.

Outro caractere muito importante é o dois pontos (:), o qual pode ser usado para especificar iterações do comando for, criar vetores ordenados ou ainda subscrever matrizes. Exemplificando: a declaração 1:3 corresponde ao vetor [1 2 3]; X(:,n) corresponde à n -ésima coluna da matriz X; e X(n,:) corresponde à n -ésima linha da matriz X.

Entrada de variáveis e matrizes:

Para entrar com variáveis no MATLAB, basta digitar o nome da variável, igualando-a ao seu valor, como por exemplo:

a = 1

Uma ressalva importante é que o MATLAB faz distinção entre letras maiúsculas e minúsculas, portanto as variáveis a e A são diferentes.

Para entrar com um vetor linha procede-se da mesma maneira, delimitando-o com colchetes e utilizando e espaços (ou vírgulas) para separar seus elementos:

b = [1 2 3 4 5]

O mesmo é válido para vetores coluna, com a diferença de que os elementos são separados por ponto-e-vírgula:

c = [1;2;3;4;5]

Para matrizes, utiliza-se espaços (ou vírgulas) para separar os elementos de uma mesma linha e ponto-e-vírgula para separar as linhas, da seguinte forma:

d = [1 2 3;
4 5 6;
7 8 9]

Uma vez entrados os dados, pode-se realizar as operações desejadas, como por exemplo uma soma:

A = B + C

Pode-se também utilizar-se de funções, como por exemplo a que fornece a matriz inversa da matriz usada como argumento (B):

$$A = \text{inv}(B)$$

Os conceitos básicos sobre o MATLAB até aqui apresentados serão suficientes para a execução das práticas de laboratório de processamento de imagens propostas mais a frente. O leitor, no entanto, não deve se limitar ao exposto neste apêndice, realizando as suas próprias experiências e criando maior intimidade com o ambiente do MATLAB.

A seguir serão dadas algumas descrições simples das principais funções e comandos do MATLAB. Maiores detalhes podem ser obtidos através do comando `help`, que fornece uma lista de funções e operadores pré-definidos para os quais há informações de auxílio disponíveis. O comando `help <nome-da-função>` fornece informações sobre a função especificada. Trata-se de uma facilidade de auxílio interativo bastante útil quando se deseja saber sobre o funcionamento de funções específicas.

Principais Funções

<code>abs</code>	valor absoluto ou módulo de um número complexo.
<code>angle</code>	ângulo de fase de um número completo.
<code>atan</code>	arco tangente
<code>conj</code>	conjugado complexo.
<code>conv</code>	convolução.
<code>corrcoef</code>	coeficiente de correlação entre duas matrizes.
<code>cos</code>	coseno.
<code>cosh</code>	coseno hiperbólico.
<code>cov</code>	covariância.
<code>deconv</code>	deconvolução.
<code>det</code>	determinante de uma matriz.
<code>diag</code>	matriz diagonal.
<code>eig</code>	autovalores e autovetores de uma matriz.
<code>exp</code>	exponenciação na base natural (e).
<code>expm</code>	exponenciação de matriz.
<code>eye</code>	matriz identidade.
<code>filter</code>	implementação de filtro digital.
<code>imag</code>	parte imaginária de um número complexo.
<code>inv</code>	matriz inversa.
<code>length</code>	comprimento de um vetor.
<code>log</code>	logaritmo na base natural (e).
<code>logm</code>	logaritmo de matriz.
<code>log10</code>	logaritmo na base 10.
<code>max</code>	valor máximo de um vetor ou matriz*.
<code>mean</code>	valor médio de um vetor.
<code>median</code>	valor da mediana de um vetor.
<code>min</code>	valor mínimo de um vetor ou matriz*.
<code>ones</code>	matriz cujos elementos possuem todos valor 1.
<code>prod</code>	produto de elementos de matrizes.
<code>rand</code>	geração de números e matrizes com valores aleatórios.
<code>rank</code>	posto de uma matriz.

<code>real</code>	parte real de um número complexo.
<code>rem</code>	resto de divisão.
<code>sign</code>	função sinal.
<code>sin</code>	seno.
<code>sinh</code>	seno hiperbólico.
<code>size</code>	dimensões de uma matriz (linhas e colunas).
<code>sqrt</code>	raiz quadrada.
<code>sqrtm</code>	raiz quadrada de uma matriz.
<code>std</code>	desvio padrão.
<code>sum</code>	soma dos elementos de um vetor ou matriz*.
<code>tan</code>	tangente.
<code>tanh</code>	tangente hiperbólica.
<code>trace</code>	traço de uma matriz.
<code>zeros</code>	matriz cujos elementos possuem todos valor 0.

* as funções max, min e sum quando utilizadas com matrizes, retornam um vetor linha contendo os resultados das operações individuais em cada coluna da matriz.

B.3. Comandos e Funções da Toolbox de Processamento de Imagens

Entrada e saída:

<code>bmpread</code>	lê arquivo BMP (Microsoft Windows Bitmap) do disco.
<code>bmpwrite</code>	escreve arquivo BMP (Microsoft Windows Bitmap) para o disco.
<code>gifread</code>	lê arquivo GIF (Graphics Interchange Format) do disco.
<code>gifwrite</code>	escreve arquivo GIF (Graphics Interchange Format) para o disco.
<code>hdfpeek</code>	lista pares de objetos tag/ref em arquivo HDF.
<code>hdfread</code>	lê dados de arquivo HDF.
<code>hdfwrite</code>	escreve dados para arquivo HDF.
<code>pcxread</code>	lê arquivo PCX (ZSoft Paint Format) do disco.
<code>pcxwrite</code>	escreve arquivo PCX (ZSoft Paint Format) para o disco.
<code>tiffread</code>	lê arquivo TIFF (Tagged Image File Format) do disco.
<code>tiffwrite</code>	escreve arquivo TIFF (Tagged Image File Format) para o disco.
<code>xwdread</code>	lê arquivo XWD (X window dump) do disco.
<code>xwdwrite</code>	escreve arquivo XWD (X window dump) para o disco.

Utilitários:

<code>getimage</code>	obtém dados da imagem a partir dos eixos.
<code>isbw</code>	verdadeiro para imagens em preto e branco.
<code>isgray</code>	verdadeiro para imagens em níveis de cinza.
<code>isind</code>	verdadeiro para imagens indexadas.

Operações em cores:

<code>brighten</code>	clareia ou escurece mapa de cores (biblioteca do MATLAB).
<code>cmunique</code>	encontra cores de mapas distintos e imagem correspondente.
<code>cmpermute</code>	permute posições de mapas de cores.
<code>cmgamma</code>	correção Gamma de mapas de cores.
<code>cmgamdef</code>	tabela de correção Gamma pré-definida.

dither	<i>dithering</i> pelo método de Floyd-Steinberg.
hsv2rgb	converte valores HSV para espaço de cores RGB (biblioteca do MATLAB).
imadjust	ajusta e amplia intensidade de imagem.
imapprox	aproxima imagem indexada para image com menor quantidade de cores.
ntsc2rgb	converte valores NTSC para o espaço de cores RGB.
rgb2gray	converte valores RGB para cinza.
rgb2hsv	converte valores RGB para o espaço de cores HSV(biblioteca do MATLAB).
rgb2ntsc	converte valores RGB para o espaço de cores NTSC.
rgbplot	desenha componentes do mapa de cores RGB (biblioteca do MATLAB).

Operações geométricas:

imcrop	recorta imagem.
imresize	redimensiona imagem.
imrotate	roda imagem.
truesize	redimensiona figura de modo que a imagem possua o tamanho real.
imzoom	ampliação e redução de uma imagem ou desenho 2-D

Melhoramento e análise:

brighten	clareia ou escurece mapa de cores (biblioteca do MATLAB).
grayslice	mapeamento por densidade (intensidade).
histeq	equalização de histograma.
imadjust	ajusta e amplia intensidade de imagem.
imapprox	aproxima imagem indexada para image com menor quantidade de cores.
imhist	histograma de imagens.
impixel	cor de um pixel.
improfile	modelo de intensidade.
interp2	interpolação bidimensional de dados (biblioteca do MATLAB).

Estatística:

mean2	média de uma matriz.
corr2	coeficiente de correlação bidimensional.
std2	desvio padrão bidimensional.

Operações morfológicas:

bwarea	área de objetos em imagem binária.
dilate	dilatação (espessamento) de imagem binária.
erode	erosão (afinamento) de imagem binária.
edge	extração de bordas.
bweuler	número de Euler.
bwmorph	operadores morfológicos.
bwperim	perímetro de objetos em imagem binária.

Projeto de filtros FIR:

fsamp2	projeto de filtros FIR 2-D através de amostragem em freqüência.
fspecial	filtros 2-D especiais.
ftrans2	projeto de filtros FIR 2-D através de transformação de freqüência.
fwind1	projeto de filtros FIR 2-D FIR utilizando janelas 1-D.
fwind2	projeto de filtros FIR 2-D FIR utilizando janelas 2-D.

`imnoise` ruído em imagem.

Resposta em freqüência:

`freqspace` espaçamento de freqüência para respostas em freqüência 2-D.
`freqz2` resposta em freqüência bidimensional.

Filtragem:

`colfilt` filtragem não-linear local por colunas.
`conv2` convolução bidimensional (biblioteca do MATLAB).
`filter2` filtragem bidimensional (biblioteca do MATLAB).
`medfilt2` filtro da mediana bidimensional.
`mfilter2` filtro mascarado.
`nlfilter` filtragem não-linear local.
`wiener2` filtro adaptativo de Wiener 2-D.

Processamento em blocos:

`bestblk` melhor tamanho de bloco para processamento em blocos.
`blkproc` processa uma imagem em blocos.
`col2im` reordena blocos de colunas distintas ou deslizantes para formar imagem.
`colfilt` filtragem não-linear local por colunas.
`im2col` reordena blocos distintos ou deslizantes para formar colunas.

Região de interesse (ROI):

`mfilter2` filtro mascarado.
`roipoly` define região de interesse poligonal.
`roiolor` define região de interesse por cor.

Transformadas:

`dct2` transformada do cosseno discreto bidimensional.
`fft2` transformada rápida de Fourier bidimensional (biblioteca do MATLAB).
`fftshift` move componente de ordem zero para o centro (biblioteca do MATLAB).
`idct2` transformada do cosseno discreto bidimensional inversa.
`ifft2` transformada rápida de Fourier 2-D inversa (biblioteca do MATLAB).
`radon` transformada de Radon.

Conversões:

`dither` *dithering* pelo método de Floyd-Steinberg.
`gray2ind` converte imagem em níveis de cinza para imagem indexada.
`hsv2rgb` converte valores HSV para espaço de cores RGB (biblioteca do MATLAB).
`im2bw` converte imagem para preto e branco por limiarização.
`imslice` obtém/coloca *slices* de imagem em um *deck* de imagens.
`ind2gray` converte imagem indexada para imagem em níveis de cinza.
`ind2rgb` converte imagem indexada para imagem RGB.
`mat2gray` converte matriz para imagem em níveis de cinza.
`ntsc2rgb` converte valores NTSC para o espaço de cores RGB.
`rgb2gray` converte valores RGB para cinza.
`rgb2HSV` converte valores RGB para o espaço de cores HSV(biblioteca do MATLAB).
`rgb2ind` converte imagem RGB para imagem indexada.

`rgb2ntsc` converte valores RGB para o espaço de cores NTSC.

Apresentação:

<code>colorbar</code>	apresenta barra de cores (escala de cores).
<code>colormap</code>	define ou obtém a tabela de consulta de cores (biblioteca do MATLAB).
<code>gray</code>	mapa de cores linear de níveis de cinza (biblioteca do MATLAB).
<code>hsv, hot, jet</code>	mapas de cores. Digite <code>help color</code> para outros (biblioteca do MATLAB).
<code>image</code>	apresenta imagem indexada (biblioteca do MATLAB).
<code>imagesc</code>	ajusta dados e apresenta como imagem (biblioteca do MATLAB).
<code>imcontour</code>	contorno da imagem.
<code>imovie</code>	faz um filme de um <i>deck</i> de imagens.
<code>imshow</code>	apresenta todos os tipos de imagens.
<code>montage</code>	apresenta um <i>deck</i> de imagens como uma montagem retangular.
<code>subimage</code>	apresenta múltiplas imagens.
<code>warp</code>	realiza <i>warp</i> da imagem sobre uma superfície.

Demonstrações:

<code>imdemo</code>	demonstração geral de processamento de imagens.
<code>dctdemo</code>	demonstração de compressão de imagens através da DCT 2-D.
<code>firdemo</code>	demonstração de filtragem FIR 2-D.
<code>nlfldemo</code>	demonstração de filtragem não-linear 2-D.

Funções privativas:

<code>cumsum3d</code>	soma cumulativa em matriz 3-D acomodada em matriz 2-D.
<code>dct</code>	transformada do cosseno discreto 1-D.
<code>dctmtx2</code>	matriz de transformação DCT 2-D unitária.
<code>ditherc</code>	arquivo MEX para <i>dithering</i> .
<code>elem3d</code>	posições de elementos de matriz 3-D acomodada em matriz 2-D.
<code>getline</code>	rastreio de movimento do mouse com linha elástica.
<code>getpts</code>	rastreio de movimento do mouse com pontos visíveis.
<code>getrect</code>	rastreio de movimento do mouse com retângulo elástico.
<code>gif</code>	comprime dados em formato GIF.
<code>hdfreadc</code>	arquivo MEX para ler arquivos HDF.
<code>hdfpeekc</code>	arquivo MEX para listar conteúdo de arquivos HDF.
<code>hdfwc</code>	arquivo MEX para escrever arquivos HDF.
<code>idct</code>	transformada do cosseno discreto 1-D inversa.
<code>im2gray</code>	converte imagens para níveis de cinza.
<code>imhistc</code>	arquivo MEX para cálculo de histograma de imagens.
<code>ndx3d</code>	índice de matriz 3-D acomodada em matriz 2-D.
<code>rgb2im</code>	converte imagens RGB para imagens indexadas ou em níveis de cinza.
<code>rle</code>	comprime dados pelo método RLE.
<code>size3d</code>	tamanho da matriz 2-D para acomodar matriz 3-D.
<code>tiff</code>	comprime dados em formato TIFF RLE.
<code>ungif</code>	descomprime dados em formato GIF.
<code>unrle</code>	descomprime dados pelo método RLE.
<code>untiff</code>	descomprime dados em formato TIFF RLE.
<code>vmquant</code>	arquivo M de interface para o arquivo MEX para quantização de cor.

`vmquantc` arquivo MEX para quantização de cor.

`waitbar` apresenta barra de progresso.

Arquivos MAT:

`bwmorph.mat` tabelas de consulta para a função `bwmorph.m`.

`forest.mat` foto digitalizada: Carmanah Old Growth Forest.

`mri.mat` imagens de ressonância magnética da cabeça de um homem.

`trees.mat` imagem digitalizada: "Trees with a view" Susan Cohen.

B.4. Roteiros de práticas de laboratório

Os roteiros de práticas de laboratório que se seguem são sugestões de utilização da *toolbox* de processamento de imagens do MATLAB para reforço e compreensão dos conceitos teóricos relacionados.

Índice das Práticas

1. Fundamentos de operação da *toolbox* de processamento de imagens do MATLAB.
2. Operações lógicas, aritméticas e estatísticas com imagens.
3. Transformações geométricas e verificação de níveis de cinza de pixels.
4. Métodos ponto-a-ponto de realce e análise de imagens.
5. Filtragem no domínio espacial.
6. Transformadas de Fourier (FFT) e filtragem no domínio da freqüência.
7. Morfologia Matemática.

Os scripts das práticas acima relacionadas estão disponíveis na Internet nos seguintes endereços:

["http://www.cse.fau.edu/~omarques/PDI/"](http://www.cse.fau.edu/~omarques/PDI/)

["http://www.daeln.cefetpr.br/~hugo/PDI/"](http://www.daeln.cefetpr.br/~hugo/PDI/)

Prática 1

Fundamentos da *toolbox* de processamento de imagens do MATLAB

Objetivos:

1. Conhecer os aspectos básicos da ferramenta de trabalho para estas práticas.
2. Conhecer os tipos de imagens com que trabalha o programa.
3. Familiarizar-se com os sistemas de coordenadas existentes.
4. Conhecer as técnicas de exibição de imagens disponíveis no MATLAB.

Resumo da teoria:

A *toolbox* do MATLAB permite trabalhar com 4 tipos de imagens:

- imagens indexadas
- imagens de intensidade
- imagens binarizadas e
- imagens RGB.

As imagens indexadas requerem duas matrizes: uma delas tem as dimensões da imagem e cada ponto desta matriz especifica um índice que serve para pesquisar em uma segunda matriz, que contém o mapa de cores, quais são os componentes R (Vermelho - *Red*), G (Verde - *Green*) e B (Azul - *Blue*) de cada pixel.

As imagens de intensidade contêm toda a informação sobre a imagem em uma única matriz. Cada elemento desta matriz representa o nível de intensidade do pixel, em uma faixa normalizada de 0 (preto) a 1 (branco).

Imagens binarizadas são um caso particular de imagem de intensidade, no qual cada pixel somente pode assumir o valor 0 (preto) ou 1 (branco).

Imagens RGB são compostas por três matrizes separadas, cada qual contendo os valores dos componentes R, G e B (normalizados em uma faixa de 0 a 1) de cada pixel.

O MATLAB também permite trabalhar com coleções de imagens relacionadas entre si, denominadas *Image Decks*. Cada imagem dentro de um *Image Deck* é chamada *Image Slice*. Todas as imagens em um *deck* devem ter o mesmo tamanho.

O MATLAB oferece várias funções para converter entre duas formas de representação de imagens. Desta forma, pode-se transformar uma imagem de um certo formato em qualquer outro, assim como também é possível transformar uma matriz qualquer em uma imagem de intensidade. Estas funções estão resumidas a seguir:

Para converter...	Em...	Use...
Imagen indexada	Imagen binarizada	<code>roicolor, roipoly, im2bw</code>
Imagen indexada	Imagen de intensidade	<code>ind2gray</code>
Imagen indexada	Imagen RGB	<code>ind2rgb</code>
Imagen de intensidade	Imagen binarizada	<code>edge, im2bw, roicolor, roipoly</code>
Imagen de intensidade	Imagen indexada	<code>grayslice, gray2ind</code>
Imagen de intensidade	Imagen RGB	a matriz original de intensidade para todos os 3 componentes R, G e B
Imagen RGB	Imagen indexada	<code>rgb2ind</code>
Imagen RGB	Imagen de intensidade	<code>rgb2gray</code>
Imagen RGB	Imagen binarizada	<code>im2bw</code>
Imagen binarizada	Imagen indexada	<code>gray2ind</code>
Matriz	Imagen de intensidade	<code>mat2gray</code>
<i>Deck</i>	Imagen (<i>subdeck</i>)	<code>imslice</code>

Além disso, o programa permite importar e exportar imagens nos formatos: GIF, TIFF, HDF, BMP, XWD ou PCX.

Conforme observado na seção 2.1, a representação de coordenadas de um pixel em uma imagem não é padronizada, existindo pelo menos três sistemas de coordenadas importantes:

- O cartesiano (eixo x horizontal, eixo y vertical orientado para cima e origem no ponto de coordenadas (0,0))
- O sistema de coordenadas de matrizes (eixo j horizontal, eixo i vertical orientado para baixo e origem no ponto de coordenadas (1,1)). Este é o sistema de coordenadas usado pela *toolbox* de processamento de imagens do MATLAB.
- O sistema de coordenadas de pixels, que não tem uma notação padrão (na convenção adotada neste livro, eixo y horizontal, eixo x vertical orientado para baixo e origem no ponto de coordenadas (0,0)).

Para exibir imagens, utiliza-se a função `imshow` com parâmetros que dependem do tipo de imagem que se está exibindo. Para exibir *decks*, utiliza-se a função `montage`.

Para exibir múltiplas imagens, pode-se usar o comando `subplot`, fazendo os ajustes de mapa de cores eventualmente necessários.

Procedimento:

1. Iniciar o MATLAB.
2. Criar uma matriz `A`, de tamanho 4 x 5, cujos valores são:

```
[1   1   2   1   3  
 1   1   2   3   1  
 2   2   3   2   2  
 1   3   2   1   1]
```

3. Convertê-la em uma imagem com três níveis de cinza, usando:

```
I = mat2gray(A)
```

4. Exibi-la com três níveis de cinza¹ e refletir sobre o resultado obtido:

```
imshow(I, 3)
```

5. Criar um mapa de cores através de uma matriz `mapa`:

```
mapa = [.4 .4 .4; 0 .6 1; 1 0 0]
```

6. Exibir a imagem indexada com o mapa de cores criado e interpretar o resultado obtido, usando o comando:

```
imshow(A, mapa)
```

7. Criar uma imagem de intensidade em forma de faixas de diferentes tons de cinza, desde o branco até o preto:

```
F = (0:15)/15  
G = [F; F; F]
```

8. Verificar os valores dos pixels da imagem e interpretá-los.
9. Exibir a imagem com 16 níveis de cinza, fazendo:

```
imshow(G, 16)
```

¹ Apesar do valor 3 não ser uma potência inteira positiva de 2, a função `imshow` aceita-o como parâmetro.

10. Carregar uma imagem indexada colorida já disponível na *toolbox* e exibi-la. Notar que após ter sido carregada, sua matriz de índices será armazenada na variável *x*, enquanto seu mapa de cores (palheta) estará na variável *map*.

```
load trees
imshow(X,map)
```

11. Usando o comando *whos* verificar o tamanho da matriz *map* e concluir a respeito.

12. Verificar o conteúdo da matriz *map*, observando que nenhuma linha é igual a outra e que cada uma delas corresponde a uma combinação das componentes R, G e B, normalizadas na faixa de 0 a 1.

13. Converter a imagem *trees* para imagem de intensidade e exibi-la com 128 níveis de cinza.

```
J = ind2gray(X,map);
imshow(J,128)
```

14. Criar uma imagem binarizada:

```
BN = [0 0 1 0 0; ...
      0 1 1 1 0; ...
      1 1 1 1 1; ...
      0 1 1 1 0; ...
      0 0 1 0 0]
```

15. Exibi-la usando:

```
imshow(BN,2)
```

16. Converter a imagem *trees* de indexada para RGB e exibi-la:

```
[R,G,B] = ind2rgb(X,map);
imshow(R,G,B)
```

17. Verificar o conteúdo de R, G e B do pixel de coordenadas (5,5).

Opção 1 (utilizando o conceito de imagem indexada):

- Verificar o conteúdo de *x*(5,5)
 - Verificar resposta: *ans* = 106
 - Pesquisar a fila 106 da matriz *map*:
- ```
map(106, 1:3)
```
- Aparecerá a resposta:

```
ans = 0.5490 0.7412 0.9059
```

Opção 2 (utilizando as variáveis R, G e B):

Pesquisar os valores de *R*(5,5), *G*(5,5) e *B*(5,5).

18. Para exemplificar o conceito de *decks* de imagens, carregar a imagem *mri* e exibi-la usando a seqüência<sup>2</sup>:

```
clear all
load mri
```

---

<sup>2</sup> O último passo requer grande quantidade de memória disponível. Caso seja impossível executá-lo na sua configuração de equipamento, passar ao item 19.

```
colormap(map)
montage(D,siz)
```

19. Para exibir uma única imagem, utilizar a função `imslice` para extraí-la do `deck`, armazenando-a em uma variável separada. Posteriormente, exibi-la usando `imshow`:

```
S3 = D(imslice(siz,3));
imshow(S3,map)
```

20. Testar a exibição de múltiplas imagens com o mesmo mapa de cores, com a seqüência de passos a seguir:

```
clear all
load trees
subplot(1,2,1), imshow(X, map), title ('Antes de rotacionar')
subplot(1,2,2), imshow(imrotate(X,35,'crop'),map),title('Depois')
```

21. Testar a exibição de múltiplas imagens com diferentes mapas de cores, com a seqüência de passos a seguir:

```
clear all
load trees
subplot(1,2,1), imshow(X,map), colormap(map)
load kids
subplot(1,2,2), imshow(X+size(colormap,1), [colormap;map])
```

## Prática 2

### Operações lógicas, aritméticas e estatísticas com imagens

#### Objetivos:

1. Executar operações lógicas e aritméticas entre imagens.
2. Obter parâmetros estatísticos de imagens.

#### Resumo da teoria:

Imagens são matrizes. Portanto, as propriedades de aritmética e álgebra matricial também são válidas para imagens e todas as operações que se pode efetuar com matrizes também podem ser efetuadas com imagens.

Dentre as operações aritméticas que se pode realizar com imagens, veremos as seguintes:

- Adição
- Subtração
- Multiplicação
- Divisão
- Diferença absoluta

Dentre as operações lógicas que se pode realizar com imagens, veremos:

- AND
- OR
- XOR

Para uma visão ampla das aplicações destas operações sobre imagens binarizadas sugerimos o capítulo 7 de [Russ 1995].

Dentre os parâmetros estatísticos que se podem extrair de uma imagem, veremos:

- Média
- Desvio padrão

#### Procedimento:

1. Iniciar o MATLAB.
2. Para a primeira parte desta prática trabalharemos com três imagens de mesmo tamanho. Em decorrência disto, inicialmente utilizaremos a função `imcrop` para criar três imagens de mesmo tamanho (100 x 100), a partir das imagens `trees`, `kids` e `forest` e utilizaremos as funções `ind2gray` e `im2bw` para gerar suas versões monocromáticas e binarizadas, respectivamente.

```

rect = [5, 5, 100, 100];
load trees
Y1 = X;
map1 = map;
Y1 = imcrop(Y1,rect);
M1 = ind2gray(Y1, map1);
BW1= im2bw(Y1,map1,.5);

load kids
Y2 = X;
map2 = map;
Y2 = imcrop(Y2,rect);
M2 = ind2gray(Y2, map2);
BW2=im2bw(Y2,map2,.3);

```

```

load forest
Y3 = X;
map3 = map;
Y3 = imcrop(Y3,rect);
M3 = ind2gray(Y3, map3);
BW3=im2bw(Y3,map3,.5);

```

3. Para ilustrar a operação de adição, vamos mostrar em uma única janela as imagens BW1, BW2 e BW3 juntamente com os resultados das adições entre elas (BW1+BW2, BW2+BW3, BW1+BW3):

```

subplot(2,3,1), imshow(BW1,2), title('BW1')
subplot(2,3,2), imshow(BW2,2), title('BW2')
subplot(2,3,3), imshow(BW3,2), title('BW3')
subplot(2,3,4), imshow((BW1+BW2),2), title('BW1+BW2')
subplot(2,3,5), imshow((BW1+BW3),2), title('BW1+BW3')
subplot(2,3,6), imshow((BW2+BW3),2), title('BW2+BW3')

```

4. Examinar atentamente os resultados, maximizando a janela que contém as seis imagens, e concluir a respeito da adição de imagens binarizadas.

5. Analogamente, para ilustrar a operação de subtração, vamos mostrar em uma única janela as imagens BW1, BW2 e BW3 juntamente com os seis possíveis resultados das subtrações entre elas (BW1-BW2, BW2-BW3, BW1-BW3, BW3-BW2, BW2-BW1, BW3-BW1):

```

subplot(3,3,1), imshow(BW1,2), title('BW1')
subplot(3,3,2), imshow(BW2,2), title('BW2')
subplot(3,3,3), imshow(BW3,2), title('BW3')
subplot(3,3,4), imshow((BW1-BW2),2), title('BW1-BW2')
subplot(3,3,5), imshow((BW2-BW1),2), title('BW2-BW1')
subplot(3,3,6), imshow((BW3-BW1),2), title('BW3-BW1')
subplot(3,3,7), imshow((BW1-BW3),2), title('BW1-BW3')
subplot(3,3,8), imshow((BW2-BW3),2), title('BW2-BW3')
subplot(3,3,9), imshow((BW3-BW2),2), title('BW3-BW2')

```

6. Examinar atentamente os resultados, maximizando a janela que contém as nove imagens, e concluir a respeito.

7. Para exemplificar a adição de imagens monocromáticas, repetiremos o passo 3, agora com as imagens M1, M2 e M3.

```

subplot(2,3,1), imshow(M1,256), title('M1')
subplot(2,3,2), imshow(M2,256), title('M2')
subplot(2,3,3), imshow(M3,256), title('M3')
subplot(2,3,4), imshow((M1+M2),256), title('M1+M2')
subplot(2,3,5), imshow((M1+M3),256), title('M1+M3')
subplot(2,3,6), imshow((M2+M3),256), title('M2+M3')

```

8. Examinar atentamente os resultados, maximizando a janela que contém as seis imagens e concluir a respeito.

9. Analogamente, efetuar agora as seis subtrações possíveis entre as imagens M1, M2 e M3 e exibir os resultados obtidos, concluindo a respeito.<sup>3</sup>

Solução:

---

<sup>3</sup> Para garantir que os valores resultantes da subtração são coerentes com a teoria e estão contidos no intervalo normalizado [0, 1] utilizamos uma operação de truncamento dos valores negativos, zerando-os.

```

subplot(3,3,1), imshow(M1,256), title('M1')
subplot(3,3,2), imshow(M2,256), title('M2')
subplot(3,3,3), imshow(M3,256), title('M3')
subplot(3,3,4), imshow((M1-M2).*((M1-M2)>=0),256), title('M1-M2')
subplot(3,3,5), imshow((M2-M1).*((M2-M1)>=0),256), title('M2-M1')
subplot(3,3,6), imshow((M3-M1).*((M3-M1)>=0),256), title('M3-M1')
subplot(3,3,7), imshow((M1-M3).*((M1-M3)>=0),256), title('M1-M3')
subplot(3,3,8), imshow((M2-M3).*((M2-M3)>=0),256), title('M2-M3')
subplot(3,3,9), imshow((M3-M2).*((M3-M2)>=0),256), title('M3-M2')

```

10. Nesta etapa examinaremos a multiplicação entre imagens monocromáticas. Em geral, quando os níveis de cinza das imagens não estão normalizados, deve-se especificar um fator ( $<1$ ) para multiplicar pelo resultado da operação de multiplicação, de tal maneira que a imagem resultante possua valores dentro da faixa de níveis de cinza desejada. No MATLAB, entretanto, isto não é necessário, pois as imagens já estão normalizadas em uma faixa de 0 a 1.

Notar que a operação utilizada no MATLAB é o produto elemento-por-elemento (denotado pelo símbolo ' $\cdot \ast$ '), pois é isto o que se deseja, não um produto matricial.

```

subplot(2,3,1), imshow(M1,256), title('M1')
subplot(2,3,2), imshow(M2,256), title('M2')
subplot(2,3,3), imshow(M3,256), title('M3')
subplot(2,3,4), imshow((M1.*M2),256), title('M1*M2')
subplot(2,3,5), imshow((M1.*M3),256), title('M1*M3')
subplot(2,3,6), imshow((M2.*M3),256), title('M2*M3')

```

11. Examinar os resultados, maximizando a janela que contém as seis imagens, e concluir a respeito da multiplicação de imagens monocromáticas.

12. Analogamente ao passo 10, efetuaremos agora a divisão entre os elementos das matrizes M1, M2 e M3, dois a dois. O operador utilizado será ' $\cdot /$ ' pelos mesmos motivos mencionados antes para a multiplicação.

```

subplot(3,3,1), imshow(M1,256), title('M1')
subplot(3,3,2), imshow(M2,256), title('M2')
subplot(3,3,3), imshow(M3,256), title('M3')
subplot(3,3,4), imshow((M1./M2),256), title('M1/M2')
subplot(3,3,5), imshow((M2./M1),256), title('M2/M1')
subplot(3,3,6), imshow((M3./M1),256), title('M3/M1')
subplot(3,3,7), imshow((M1./M3),256), title('M1/M3')
subplot(3,3,8), imshow((M2./M3),256), title('M2/M3')
subplot(3,3,9), imshow((M3./M2),256), title('M3/M2')

```

Notar que o MATLAB poderá emitir mensagens de alerta por possíveis tentativas de divisão por zero, que não prejudicam a conclusão da operação ou sua interpretação qualitativa.

13. Examinar atentamente os resultados, maximizando a janela que contém as nove imagens e concluir acerca da divisão de imagens monocromáticas.

14. Para finalizar nosso estudo de operações aritméticas, ilustraremos o uso da função `abs` para obtermos a diferença absoluta entre duas imagens, denominando-a pelo nome simbólico DIF.

```

subplot(3,3,1), imshow(M1,256), title('M1')
subplot(3,3,2), imshow(M2,256), title('M2')
subplot(3,3,3), imshow(M3,256), title('M3')
subplot(3,3,4), imshow(abs(M1-M2),256), title('M1 DIF M2')
subplot(3,3,5), imshow(abs(M2-M3),256), title('M2 DIF M3')
subplot(3,3,6), imshow(abs(M3-M1),256), title('M3 DIF M1')

```

```
subplot(3,3,7), imshow(abs(M2-M1),256), title('M2 DIF M1')
subplot(3,3,8), imshow(abs(M3-M2),256), title('M3 DIF M2')
subplot(3,3,9), imshow(abs(M1-M3),256), title('M1 DIF M3')
```

Os valores das janelas 4 e 7, 5 e 8, 6 e 9, são (ou parecem ser) idênticos dois a dois? Por quê?

15. Para exemplificar as operações lógicas entre imagens, tornaremos a utilizar as imagens binarizadas BW1, BW2 e BW3 e os operadores lógicos & (and), | (or) e xor, disponíveis no MATLAB. Faremos isto mostrando seis imagens por janela (as três originais mais os três resultados do uso de um certo operador).

Inicialmente para o operador and:

```
subplot(2,3,1), imshow(BW1,2), title('BW1')
subplot(2,3,2), imshow(BW2,2), title('BW2')
subplot(2,3,3), imshow(BW3,2), title('BW3')
subplot(2,3,4), imshow((BW1&BW2),2), title('BW1 and BW2')
subplot(2,3,5), imshow((BW1&BW3),2), title('BW1 and BW3')
subplot(2,3,6), imshow((BW2&BW3),2), title('BW2 and BW3')
```

16. Examinar atentamente os resultados, maximizando a janela que contém as seis imagens e concluir a respeito.

17. Repita os passos 15 e 16, desta vez para o operador or.

Solução:

```
subplot(2,3,4), imshow((BW1|BW2),2), title('BW1 or BW2')
subplot(2,3,5), imshow((BW1|BW3),2), title('BW1 or BW3')
subplot(2,3,6), imshow((BW2|BW3),2), title('BW2 or BW3')
```

18. Repita os passos 15 e 16, desta vez para o operador xor.

Solução:

```
subplot(2,3,4), imshow((xor(BW1,BW2)),2), title('BW1 xor BW2')
subplot(2,3,5), imshow((xor(BW1,BW3)),2), title('BW1 xor BW3')
subplot(2,3,6), imshow((xor(BW2,BW3)),2), title('BW2 xor BW3')
```

19. Deixamos como exercício efetuar operações lógicas sobre imagens com múltiplos níveis de cinza, uma vez que o MATLAB não nos permite fazê-lo diretamente (veja a descrição dos operadores &, | e xor no arquivo de ajuda do software).

20. O MATLAB permite extrair algumas informações estatísticas sobre o conteúdo de uma imagem. A seguir, verificaremos a média e o desvio padrão de cada uma das três imagens monocromáticas M1, M2 e M3, armazenando os resultados nas variáveis m1, m2, m3, d1, d2 e d3, respectivamente. Anotar os resultados obtidos e concluir a respeito.

```
m1=mean2(M1)
m2=mean2(M2)
m3=mean2(M3)
d1=std2(M1)
d2=std2(M2)
d3=std2(M3)
```

### Prática 3

#### Transformações geométricas e verificação de níveis de cinza de pixels

##### Objetivos:

1. Conhecer as operações geométricas básicas sobre imagens: interpolação, rotação, *cropping*, *zooming*, *flip* horizontal e vertical e *resizing*.
2. Verificar as informações de níveis de cinza ao longo de uma linha (*intensity profile*) da imagem.
3. Inspecionar os valores de intensidade de um pixel ou grupo de pixels.

##### Resumo da teoria:

Dada uma imagem, é bastante comum desejar modificá-la, alterando suas dimensões, rotacionando-a, eliminando parte da imagem e permanecendo somente com outra parte, obtendo uma versão espelhada dela etc.

A este tipo de operação sobre imagens denominamos transformações geométricas, pois em geral envolvem princípios básicos de geometria analítica aplicados a matrizes, que neste caso são imagens.

A maior parte das operações geométricas utiliza um processo conhecido como interpolação, que é uma técnica utilizada pelo programa para determinar valores entre pixels definidos. Por exemplo, se alterarmos as dimensões de uma imagem de tal maneira que ela contenha o dobro do número de pixels original, o programa obterá os valores para os novos pixels através de interpolação. A Toolbox de Processamento de Imagens do MATLAB provê três métodos de interpolação:

- Vizinho mais próximo
- Interpolação bilinear
- Interpolação bicúbica

A interpolação pelo método do vizinho mais próximo ajusta uma superfície constante através dos valores de intensidade. O valor de um pixel interpolado é o valor do pixel mais próximo.

A interpolação bilinear ajusta uma superfície linear sobre os valores já existentes. O valor de um pixel interpolado é uma combinação dos valores de seus 4-vizinhos.

A interpolação bicúbica ajusta uma superfície cúbica sobre os valores já existentes. O valor de um pixel interpolado é uma combinação dos 16 pixels mais próximos.

Algumas funções geométricas permitem que se especifique o método de interpolação desejado (*nearest*, *bilinear* ou *bicubic*) como um de seus argumentos. Em geral, o primeiro método é o mais apropriado para imagens indexadas, enquanto as interpolações bilineares ou bicúbicas são as mais recomendadas para imagens de intensidade ou RGB.

Finalmente, outra tarefa bastante comum é inspecionar os valores de intensidade de um pixel, um grupo de pixels ou ao longo de uma linha da imagem. Nesta prática veremos como fazê-lo usando o MATLAB.

##### Procedimento:

1. Iniciar o MATLAB.
2. Carregar a imagem *trees* em memória, rotacioná-la de um ângulo de 25°, armazenando o resultado na variável *Y* e exibindo a imagem resultante:

```
load trees
Y = imrotate(X,25);
imshow(Y,map)
```

3. A função `imrotate` permite que se especifique o método de interpolação desejado, sendo o método do vizinho mais próximo o *default*. Além disso, permite também especificar um quarto parâmetro, '`crop`', que faz com que a função retorne a porção central do resultado, cujas dimensões são as mesmas da imagem original. Para ver o efeito destas opções, sugerimos a seqüência a seguir:

```
Z=imrotate(X,25,'bilinear');
whos
figure
imshow(Z,map)
W=imrotate(X,25,'bicubic');
figure
imshow(W,map)
V= imrotate(X,25,'crop');
whos
figure
imshow(V,map)
```

Nota: O comando `whos` serve neste caso para verificar as dimensões das variáveis: X é uma imagem de (258 x 350), enquanto Y ou Z são de (385 x 429); a imagem V, obtida com a opção '`crop`' tem tamanho (258 x 350).

Verificar que os métodos utilizados para gerar Z ou W são ainda mais lentos que o método original (a interpolação pelo vizinho mais próximo), o qual era efetivamente o melhor método para este caso. Explicar porque os resultados com o método original são claramente melhores.

4. Outra operação comum é a operação de *cropping*, implementada pela função `imcrop`, ilustrada na seqüência a seguir, na qual a área de *cropping* é definida por uma matriz `rect` (1 x 4), cujo conteúdo representa, respectivamente, a coordenada da coluna onde se deseja o corte, a linha onde o corte deve começar, a largura da imagem cortada e sua altura.

```
clear all
load trees
Y=imcrop(X,[71,107,92,95]);
subplot(2,1,1), imshow(X,map)
subplot(2,1,2), imshow(Y,map)
```

5. Outra opção para definir a área de *cropping* é especificá-la usando o mouse. Para fazê-lo, basta chamar `imcrop` sem argumentos, mover o mouse sobre a imagem (que já deve estar disponível em outra janela), pressionar o botão esquerdo para definir o canto superior esquerdo e arrastar através da imagem o mouse até o canto inferior direito. Ao soltar o botão do mouse, aparecerá, no lugar da imagem original, a imagem já cortada. A seqüência é:

```
clc
imshow(X,map)
imcrop
```

6. Para alterar o tamanho de uma imagem utiliza-se a função `imresize`, cujos parâmetros incluem a possibilidade de especificar o método de interpolação desejado. Para verificar os efeitos de `imresize`, recomendamos a seqüência a seguir:

```
close
clear
load trees
Y=imresize(X,2);
whos
figure
imshow(Y,map)
Z=imresize(X,.7);
```

```

whos
figure
imshow(Z,map)
W=imresize(X, [100, 200]);
whos
figure
imshow(W,map)

```

Sobre os resultados vistos, pergunta-se:

- Foi possível perceber alguma diferença apreciável entre as imagens X, Y e Z, ao visualizá-las?
- Quais foram as consequências de se especificar o novo tamanho para a imagem W como sendo (100 x 200)?

7. Muitas vezes deseja-se ver com mais detalhes o conteúdo de uma imagem (ou de uma região da imagem), sem alterar suas dimensões. Para isto se utiliza um *zoom in*. Em outros casos, a imagem é muito grande para caber na tela. Nestas ocasiões se utiliza o *zoom out*. No MATLAB, o comando `imzoom on` habilita o uso do *zoom* com o mouse. Neste caso, basta pressionar o botão esquerdo do mouse sobre uma imagem para ampliá-la (*zoom in*) ou pressionar o botão direito para reduzir sua tamanho na tela (*zoom out*). O centro da região para o *zoom* será o ponto onde se pressionou o botão do mouse. Para desligar o modo de *zoom*, escreva `imzoom off`. Teste o uso do *zoom* com uma imagem dentre as já conhecidas (tree, kids ou forest).

8. Outra opção bastante comum em programas de processamento de imagens é o *flip*, que produz uma imagem espelhada na horizontal ou vertical. O MATLAB não provê nenhuma função *flip*, porque o *flip* nada mais é que uma combinação de transposição e rotação de matrizes. Assim, para obter o efeito do *flip*, sugerimos a seqüência abaixo:

- Para o *flip* horizontal:

```

clear
load trees
U = X'; % calcula a matriz transposta de X
W = imrotate(U,270);
subplot(2,2,1), imshow(X,map)
subplot(2,2,2), imshow(W,map)

```

- Para o *flip* vertical:

```

V = imrotate(U,90);
subplot(2,2,3), imshow(V,map)

```

9. Para visualizar graficamente as variações de níveis de cinza ao longo de uma linha sobre a imagem, pode-se utilizar a função `improfile`. Chamando `improfile` sem argumentos, podemos definir com o botão esquerdo do mouse onde começa a linha de interesse e com o direito onde ela termina.<sup>4</sup> Executar a seqüência a seguir, lembrando-se de usar o mouse depois da ultima linha de comando abaixo:

```

clear
load spine
G=ind2gray(X,map);
clf
imshow(G,64)
improfile

```

---

<sup>4</sup> Pode-se definir mais de uma linha, usando o botão esquerdo do mouse seguidas vezes e pressionando o botão direito para concluir o processo.

10. Outra opção consiste em passar parâmetros para a função `improfile`, onde os parâmetros são as coordenadas dos pontos inicial e final da linha. A seqüência de passos abaixo ilustra o uso da função `improfile` com parâmetros:

```
X = [123 233]
Y = [132 290]
subplot(1,2,1), imshow(G,64)
hold on
plot(X,Y)
subplot(1,2,2), improfile(G,X,Y)
```

11. Finalmente, veremos como obter informações sobre um pixel ou grupo de pixels, através da função `impixel`:

Para imagens indexadas, pode-se armazenar em uma variável o conteúdo das componentes R, G e B de um pixel (dadas suas coordenadas) ou de um grupo de pixels (dadas duas listas de coordenadas destes pixels). Por exemplo:

```
load trees
v1 = impixel(X,map,21,151)
x1 = [2 3 4 5 7]
y1 = [7 8 9 10 12]
v2 = impixel(X,map,x1,y1)
```

Pergunta-se:

- a. Quais são as dimensões e o conteúdo das variáveis v1 e v2?
- b. Qual o significado de cada linha da matriz v2 e como estas linhas estão relacionadas às variáveis x1 e y1?

12. Outra alternativa é utilizar `impixel` interativamente com o mouse, pressionando o botão esquerdo do mouse sobre cada ponto sobre o qual se deseja conhecer os valores de R, G e B. Quando todos os pontos tiverem sido selecionados, pressione *Enter*. Escolher uma imagem e alguns pontos de interesse e relatar as conclusões obtidas.

## Prática 4

### Métodos ponto-a-ponto de realce e análise de imagens

#### Objetivos:

1. Obter e interpretar o histograma de uma imagem.
2. Conhecer o efeito da aplicação de funções de transformação de intensidade a imagens.
3. Aprimorar o contraste de uma imagem através da equalização de seu histograma.
4. Destacar pixels de interesse em uma imagem monocromática através de técnicas de pseudocolorização.
5. Adequar os valores de níveis de cinza de uma imagem, de tal maneira que se situem dentro de uma faixa de valores de interesse.

#### Resumo da teoria:

##### Histograma de uma imagem

O histograma de uma imagem é um gráfico que mostra a quantidade (ou o percentual) de pixels correspondentes a cada nível de cinza em uma imagem. A função `imhist` mostra o histograma de uma imagem.

##### Transformações de intensidade

Os pixels de uma imagem podem ser submetidos a uma função de transformação ponto-a-ponto, da forma  $y = x^\gamma$ , onde um fator  $\gamma$  positivo e maior que 1 produzirá uma imagem resultante mais escura que a imagem original, enquanto um fator  $\gamma$  entre 0 e 1 produzirá uma imagem resultante mais clara que a imagem original. No MATLAB, estas funções de transformação de intensidade estão disponíveis através da função `imadjust`.

##### Equalização de histograma

O histograma de uma imagem fornece informações sobre o contraste da cena correspondente. Para aprimorar o contraste de uma imagem, uma das técnicas mais comuns é a equalização de seu histograma. Esta equalização reorganiza os valores de intensidade, produzindo histogramas mais lineares.

No MATLAB, a função `histeq` implementa o conceito de equalização de histograma.

##### Pseudocolorização (*Density slicing*)

A técnica de pseudocolorização cria uma imagem colorida a partir de uma imagem de intensidade através do mapeamento de faixas de valores de intensidade a cores diferentes.

No MATLAB, a função `grayscale` implementa o conceito de pseudocolorização. Ela requer uma imagem e um parâmetro que descreva como distribuir as faixas de níveis de cinza da imagem original nas cores desejadas.

##### Modificação de histograma

Além de permitir a equalização de histograma, o MATLAB permite outras formas de modificação de histograma, como por exemplo a compressão e expansão, disponíveis através da função `imadjust`.

#### Procedimento:

1. Iniciar o MATLAB.
2. Carregar a imagem `clown`, convertê-la para imagem monocromática e exibir seu histograma, usando a função `imhist`.

```
load clown
I=ind2gray(X,map);
subplot(2,1,1), imhist(I,128)
```

```
 subplot(2,1,2), imshow(I,128)
```

3. Repetir o passo 2 para as imagens `trees` e `forest` e interpretar os resultados obtidos.

4. Verificar o efeito de aplicar transformações de intensidade a uma imagem com valores de  $\gamma$  entre 0 e 1:

```
load trees
I = ind2gray(X,map);
J = imadjust(I, [], [], .5);
subplot(2,2,1), imshow(I,128)
subplot(2,2,3), imshow(J,128)
subplot(2,2,2), imhist(I,128)
subplot(2,2,4), imhist(J,128)
```

5. Verificar o efeito de aplicar transformações de intensidade a uma imagem com valores de  $\gamma$  maiores que 1:

```
load trees
I = ind2gray(X,map);
J = imadjust(I, [], [], 1.5);
subplot(2,2,1), imshow(I,128)
subplot(2,2,3), imshow(J,128)
subplot(2,2,2), imhist(I,128)
subplot(2,2,4), imhist(J,128)
```

6. Equalizar o histograma da imagem `forest` para 128 níveis de cinza.

```
load forest
I= ind2gray(X,map);
J= histeq(I,128);
subplot(2,2,1), imshow(I,128)
subplot(2,2,2), imhist(I,128)
subplot(2,2,3), imshow(J,128)
subplot(2,2,4), imhist(J,128)
```

7. A função `histeq` também permite que o histograma seja equalizado para um número menor de níveis de cinza. Para verificar o efeito desta opção, sugere-se a seqüência:

```
load forest
I= ind2gray(X,map);
J= histeq(I,16);
subplot(2,2,1), imshow(I,16)
subplot(2,2,2), imhist(I,16)
subplot(2,2,3), imshow(J,16)
subplot(2,2,4), imhist(J,16)
```

8. Repetir os passos 6 e 7 para a imagem `trees`.

9. Para ilustrar o conceito de pseudocolorização, dividiremos a faixa total de níveis de cinza da imagem `spine` em 6 segmentos e atribuindo a cada um deles uma cor distinta, pré-definida no mapa de cores `prism`:

```
load spine
I = ind2gray(X,map);
imshow(I,128)
Y = grayslice(I,6);
imshow(Y,prism(6)); colorbar
```

10. Para verificar o conceito de expansão de histograma (*input cropping*), sugerimos a seqüência:

```
load clown
I = ind2gray(X,map);
J = imadjust(I,[0 0.5],[],[]);
subplot(2,2,2), imhist(I,128)
subplot(2,2,1), imshow(I,128)
subplot(2,2,4), imhist(J,128)
subplot(2,2,3), imshow(J,128)
```

11. Repetir o passo 10 para a imagem *forest* e concluir a respeito.

12. Para verificar o conceito de compressão de histograma (*output cropping*), sugerimos a seqüência:

```
load forest
I = ind2gray(X,map);
J = imadjust(I,[],[0.5 1.0],[]);
subplot(2,2,2), imhist(I,128)
subplot(2,2,1), imshow(I,128)
subplot(2,2,4), imhist(J,128)
subplot(2,2,3), imshow(J,128)
```

## Prática 5

### Filtragem no domínio espacial

#### Objetivos:

1. Verificar o funcionamento de filtros passa-baixas utilizando as técnicas de filtragem pela média, mediana e média de múltiplas imagens.
2. Verificar o funcionamento de filtros passa-altas no domínio espacial.
3. Verificar o funcionamento de filtros de realce utilizando a técnica de *unsharp masking*.
4. Verificar o funcionamento de filtros passa-altas *high-boost*.

#### Resumo da teoria:

Uma das aplicações mas comuns das técnicas de processamento de imagens está nas áreas de remoção de ruídos e aprimoramento da qualidade de uma imagem. Nesta prática abordaremos três técnicas de remoção de ruído, que equivalem a filtros passa-baixa, que são:

- Filtragem pela média: nesta técnica, o pixel central de uma janela (normalmente 3 x 3) é substituído pela média dos valores de intensidade de seus 8 vizinhos.
- Filtragem pela mediana: nesta técnica, o pixel central de uma janela (normalmente 3 x 3) é substituído pela mediana dos valores de intensidade de seus 8 vizinhos.
- Filtragem pela média de múltiplas imagens: nesta técnica, assumindo-se que existem  $n$  versões ruidosas de uma mesma imagem e que o ruído é aleatório e descorrelacionado, calcula-se a média das várias imagens ruidosas.

Esta prática também apresenta quatro formas de filtragem passa-altas no domínio espacial, que são:

- Realce pela função *emboss*: nesta técnica, utilizando máscaras para deteção de bordas, obtém-se um efeito de baixo relevo sobre a imagem original.
- Realce utilizando máscaras recomendadas na literatura: neste caso, executamos a convolução de uma imagem com uma máscara projetada para realçar os componentes de alta freqüência da imagem original.
- Realce utilizando a técnica de *unsharp masking*: nesta técnica, subtrai-se de uma imagem uma versão suavizada dela, que se pode obter, por exemplo, calculando a média dos 4-vizinhos de um pixel.
- Realce por filtragem *high-boost*: equivale a um filtro passa-altas, porém com maior preservação da informação de baixa freqüência da imagem.

#### Procedimento:

1. Iniciar o MATLAB.
2. Criar a matriz ‘h’ para filtragem passa-baixa pela técnica de filtragem da média.

```
h = [1 1 1
 1 1 1
 1 1 1] / 9
```

3. Carregar a imagem *kids*, convertê-la para imagem monocromática e filtrá-la usando a técnica de filtragem pela média. Exibir na tela as duas imagens e concluir a respeito.

```
load kids
I=ind2gray(X,map);
B=filter2(h,I);
subplot(1,2,1);imshow(I,128)
subplot(1,2,2);imshow(B,128)
```

4. Repetir o passo 3 para a imagem `trees`.

```
load trees
I=ind2gray(X,map);
B=filter2(h,I);
subplot(1,2,1);imshow(I,128)
subplot(1,2,2);imshow(B,128)
```

5. A seguir, aplicaremos a mesma técnica a versões ruidosas das imagens `kids` e `trees`. Verificar os resultados obtidos e concluir a respeito.

```
load kids
I=ind2gray(X,map);
J=imnoise(I,'salt & pepper');
B=filter2(h,J);
subplot(2,2,1);imshow(I,128)
subplot(2,2,2);imshow(J,128)
subplot(2,2,3);imshow(B,128)

load trees
I=ind2gray(X,map);
J=imnoise(I,'salt & pepper');
B=filter2(h,J);
subplot(2,2,1);imshow(I,128)
subplot(2,2,2);imshow(J,128)
subplot(2,2,3);imshow(B,128)
```

6. Utilizaremos na seqüência a função `medfilt2`, que implementa a técnica de filtragem pela mediana, aplicando-a inicialmente a uma imagem sem ruído.

```
load kids
I=ind2gray(X,map);
K=medfilt2(I,[3 3], [50 50]);
subplot(1,2,1);imshow(I,64)
subplot(1,2,2);imshow(K,64)
```

7. Repetir o passo anterior para uma versão da imagem `kids` contaminada com ruído sal e pimenta.

```
% load kids
% I=ind2gray(X,map);
J=imnoise(I,'salt & pepper');
K=medfilt2(J,[3 3], [50 50]);
subplot(2,2,1);imshow(I,64)
subplot(2,2,2);imshow(J,64)
subplot(2,2,3);imshow(K,64)
```

8. A seqüência a seguir executa a filtragem passa-baixas através da técnica da média de múltiplas imagens.

```
% load kids
% I=ind2gray(X,map);
J1=imnoise(I,'salt & pepper');
J2=imnoise(I,'salt & pepper');
J3=imnoise(I,'salt & pepper');
J4=imnoise(I,'salt & pepper');
J=(J1+J2+J3+J4)/4;
subplot(2,3,1);imshow(I,64)
subplot(2,3,2);imshow(J,64)
subplot(2,3,3);imshow(J1,64)
subplot(2,3,4);imshow(J2,64)
```

```
subplot(2,3,5);imshow(J3,64)
subplot(2,3,6);imshow(J4,64)
```

9. Carregar a imagem `trees`, convertê-la para monocromática e aplicar a ela um filtro obtido através da criação de uma máscara de convolução por meio da função `fspecial` com a opção `sobel`. O efeito produzido sobre a imagem original é conhecido como *emboss*.

```
load trees
clg
I=ind2gray(X,map);
h=fspecial('sobel');
A=filter2(h,I);
imshow(mat2gray(A),64)
```

10. Utilizar a seqüência de passos abaixo para realçar uma imagem utilizando máscara 3 x 3. Convém observar que o resultado deixa a desejar do ponto de vista qualitativo, mas certamente ilustra o realce das componentes de alta freqüência.

```
clear
A=[0 -1 0; -1 5 -1; 0 -1 0];
load kids
N = 255;
J=ind2gray(X,map);
K=round(J*N);
L=filter2(A,K);
[lin, col] = size(L);
for i=1:lin, for j=1:col, if L(i,j)<0 L(i,j)=0; end, end, end
for i=1:lin, for j=1:col, if L(i,j)>N L(i,j)=N; end, end, end
L = L/N;
subplot(1,2,1), imshow(J,256)
subplot(1,2,2), imshow(L,256)
```

11. Repetir o passo 10 utilizando outras máscaras sugeridas na literatura.

12. Aplicar os conceitos de *unsharp masking*, executando a seqüência de comandos a seguir.

```
% clear
% load kids
% J = ind2gray(X,map);
h = fspecial('unsharp')
clg
N = 255;
K=round(J*N);
L=filter2(h,K);
[lin, col] = size(L);
for i=1:lin, for j=1:col, if L(i,j)<0 L(i,j)=0; end, end, end
for i=1:lin, for j=1:col, if L(i,j)>N L(i,j)=N; end, end, end
L = L/N;
subplot(1,2,1), imshow(J,256)
subplot(1,2,2), imshow(L,256)
```

13. Filtrar uma imagem pela técnica de ênfase em alta freqüência (também denominada filtragem *high-boost*) utilizando três diferentes valores para o parâmetro que especifica a quantidade de informação de baixa freqüência da imagem original que se deseja preservar. Observamos mais uma vez que os resultados são pouco agradáveis visualmente, mas coerentes com os fundamentos teóricos que se deseja verificar.

```
% clear
% load kids
% J = ind2gray(X,map);
```

```

p1 = 1.1;
p2 = 1.15;
p3 = 1.2;
w1 = 9*p1 - 1;
w2 = 9*p2 - 1;
w3 = 9*p3 - 1;
m1 = [-1 -1 -1; -1 w1 -1; -1 -1 -1];
m2 = [-1 -1 -1; -1 w2 -1; -1 -1 -1];
m3 = [-1 -1 -1; -1 w3 -1; -1 -1 -1];
clg
% N = 255;
% K=round(J*N);
L1=filter2(m1,K);
[lin, col] = size(L1);
for i=1:lin, for j=1:col, if L1(i,j)<0 L1(i,j)=0; end, end, end
for i=1:lin, for j=1:col, if L1(i,j)>N L1(i,j)=N; end, end, end
L1 = L1/N;
L2=filter2(m2,K);
[lin, col] = size(L2);
for i=1:lin, for j=1:col, if L2(i,j)<0 L2(i,j)=0; end, end, end
for i=1:lin, for j=1:col, if L2(i,j)>N L2(i,j)=N; end, end, end
L2 = L2/N;
L3=filter2(m3,K);
[lin, col] = size(L3);
for i=1:lin, for j=1:col, if L3(i,j)<0 L3(i,j)=0; end, end, end
for i=1:lin, for j=1:col, if L3(i,j)>N L3(i,j)=N; end, end, end
L3 = L3/N;
subplot(2,2,1), imshow(J,256)
subplot(2,2,2), imshow(L1,256)
subplot(2,2,3), imshow(L2,256)
subplot(2,2,4), imshow(L3,256)

```

## Prática 6

### Filtragem no domínio da freqüência

**Objetivos:**

1. Verificar algumas das propriedades da transformada de Fourier direta e inversa aplicada a imagens.
2. Implementar filtros passa-altas e passa-baixas no domínio da freqüência.
3. Implementar a filtragem *high-boost* no domínio da freqüência.
4. Verificar o funcionamento de um filtro homomórfico.

**Resumo da teoria:**

**Transformada de Fourier (FFT)**

Uma imagem pode ser convertida do domínio espacial para o domínio da freqüência através de várias transformadas, sendo uma delas a transformada de Fourier bidimensional.

A FFT bidimensional está disponível no MATLAB através da função `fft2`, que é normalmente utilizada em conjunto com a função `fftshift` (que reorganiza a saída da função `fft2`, deslocando o componente de freqüência zero para o centro do arranjo bidimensional resultante).

**Filtragem passa-baixas no domínio da freqüência**

Nesta técnica, utilizam-se métodos de projeto de filtros FIR conhecidos da teoria de processamento digital de sinais 1-D, adaptando estes filtros ao caso 2-D, obtendo as máscaras de convolução correspondentes para eliminar os componentes de mais alta freqüência de uma imagem ruidosa.

**Filtragem passa-altas no domínio da freqüência**

Nesta técnica, utilizam-se métodos semelhantes ao caso passa-altas, mas com o propósito de realçar os componentes de mais alta freqüência de uma imagem.

**Procedimento:**

1. Iniciar o MATLAB.
2. Criar uma imagem 32 x 32 binária, contendo um quadrado de pixels brancos de dimensões 16 x 16 circundado por pixels pretos:

```
BN = zeros(32);
BN(9:24, 9:24) = ones(16);
```

3. Obter a FFT da imagem BN e comparar o resultado com aquele previsto teoricamente, visualizando-o tanto em 2-D quanto em 3-D:

```
F = fftshift(fft2(BN));
imshow(BN, 2)
figure
colormap(jet(64)), imagesc(log(1 + abs(F)));
colorbar
figure
mesh(log(1 + abs(F)));
```

4. Na sequiência, carregar a imagem `forest`, convertê-la de imagem indexada para imagem de intensidade, aplicar a transformada de Fourier à imagem monocromática resultante e exibir seus resultados.

```
load forest
I = ind2gray(X, map);
F = fftshift(fft2(I));
```

```
imshow(I,256)
figure
colormap(jet(64)),imagesc(log(1 + abs(F))); colorbar
```

5. Repetir o item 4 para uma porção da imagem `kids` de tamanho 128 x 128.

```
load kids
X = imcrop(X,[0, 0, 100, 100]);
I = ind2gray(X,map);
F = fftshift(fft2(I));
imshow(I,256)
figure
colormap(jet(64)),imagesc(log(1 + abs(F))); colorbar
```

6. Girar a imagem de 45 graus, recalcular a FFT e comparar os resultados, verificando a propriedade de rotação da FFT.

```
Y = imrotate(I, 45);
G = fftshift(fft2(Y));
figure
imshow(Y,256)
figure
colormap(jet(64)),imagesc(log(1 + abs(G))); colorbar
```

7. Para exemplificarmos a filtragem passa-baixas no domínio da freqüência, projetaremos um filtro passa-baixas utilizando janela de Hamming, que gerará uma máscara de convolução 11 x 11 equivalente (`h`), e aplicaremos o filtro 2-D resultante a duas imagens ruidosas, verificando os resultados. A máscara equivalente no domínio espacial é mostrada usando o comando `mesh`.

```
clear
clc
load trees
I=ind2gray(X,map);
J1=imnoise(I,'gaussian');
J2=imnoise(I,'salt & pepper');
[f1, f2] = freqspace([11,11]);
[x, y] = meshgrid(f1, f2);
Hd = zeros(size(x));
d = find(sqrt(x.*x + y.*y)<.3);
Hd(d) = ones(size(d));
h = fwirl1(Hd, hamming(11))
mesh(h)
K1 = filter2(h,J1);
K2 = filter2(h,J2);
figure
subplot(2,2,1);imshow(J1,64)
subplot(2,2,2);imshow(K1,64)
subplot(2,2,3);imshow(J2,64)
subplot(2,2,4);imshow(K2,64)
```

8. Na seqüência, ilustraremos a filtragem passa-altas no domínio da freqüência. Para tanto, projetaremos um filtro passa-altas utilizando janela de Hamming, que gerará uma máscara de convolução 11 x 11 equivalente (`h`), e aplicaremos o filtro 2-D resultante a duas imagens, verificando os resultados. O comando `mesh` é utilizado para exibir a curva de resposta em freqüência desejada. O resultado obtido é equivalente do ponto de vista qualitativo àquele obtido com a opção `sharpen` em diversos programas de processamento de imagens.

```
clear
clc
load trees
J=ind2gray(X,map);
[f1, f2] = freqspace([11,11]);
[x, y] = meshgrid(f1, f2);
Hd = ones(size(x));
d = find(sqrt(x.*x + y.*y)>.4);
Hd(d) = 2 .* ones(size(d));
mesh(Hd)
figure
h = fwind1(Hd, hamming(11))
K = filter2(h,J);
subplot(2,1,1);imshow(J,64)
subplot(2,1,2);imshow(K.*(K>=0),64)
```

9. Repetir o passo 8 com a imagem kids.

## Prática 7

### Morfologia Matemática

**Objetivos:**

1. Verificar o funcionamento das operações morfológicas básicas: dilatação, erosão, abertura e fechamento.
2. Verificar o funcionamento de algoritmos morfológicos para operações de processamento de imagens tais como: afinamento, preenchimento de regiões, extração de contorno etc.
3. Implementar um filtro para remoção de ruídos de imagens binarizadas utilizando os conceitos e operações morfológicas.

**Resumo da teoria:**

Para uma visão introdutória do tema 'Morfologia Matemática' aplicada a imagens binárias, sugerimos o capítulo 5 deste livro.

**Procedimento:**

1. Iniciar o MATLAB.

2. Carregar a imagem `circbw` e criar uma matriz SE (elemento estruturante) de acordo com o exemplo abaixo. Utilizar o comando `dilate` para dilatar a imagem com o elemento estruturante SE e mostrar na tela o resultado<sup>5</sup>.

```
load circbw;
SE = [1 0; 0 1];
C = dilate(BW,SE);
clf
subplot(2,2,1), imshow(~BW,2)
subplot(2,2,2), imshow(~C,2)
```

3. Verificar que o resultado obtido não é exatamente o esperado, porque a função `dilate` foi executada com o método `default` (`thicken`), que evita conectar objetos e preserva o número de Euler da imagem<sup>6</sup>. Para obter uma dilatação semelhante à estudada na teoria básica de operações morfológicas, especifique a opção `dilate`, executando a seqüência a seguir:

```
D = dilate(BW,'dilate');
subplot(2,2,3), imshow(~D,2)
```

Nota: Esta seqüência, bem como a do passo seguinte, não especifica o elemento estruturante. Nestes casos, o comando `dilate` adiciona pixels à 8-vizinhança de cada ponto localizado nas fronteiras dos objetos da imagem.

4. Utilizaremos agora a opção `fatten` da função `dilate`, que produz resultado semelhante ao da opção `thicken`, porém não garante que o número de Euler da imagem permaneça inalterado.

```
E = dilate(BW,'fatten');
subplot(2,2,4), imshow(~E,2)
```

<sup>5</sup> Devido à convenção utilizada no capítulo 5 ser oposta à utilizada pelo MATLAB, as imagens binarizadas devem ser invertidas quando de sua exibição.

<sup>6</sup> O número de Euler de uma imagem binária é obtido calculando-se a diferença entre o números de componentes conectados na imagem e o número de orifícios (*holes*) nela presentes.

5. Criar uma nova matriz SI (abaixo) para ser o elemento estruturante e aplicar uma erosão (função `erode`) à imagem.

```
SI = [0 1; 1 0];
F = erode(BW,SI);
clf
subplot(2,2,1), imshow(~BW,2)
subplot(2,2,2), imshow(~F,2)
```

6. Verificar que, novamente, o resultado obtido não é exatamente o esperado, porque o comando `erode` foi executado com o método *default* (*thin*), que executa a erosão sobre os objetos até um limite para o qual eles estarão reduzidos a linhas (não chegando a transformá-los em pontos), e preserva o número de Euler da imagem. Para obter uma erosão semelhante à estudada na teoria básica de operações morfológicas, especifique a opção `erode`, executando a seqüência:

```
G = erode(BW,'erode');
subplot(2,2,3), imshow(~G,2)
```

Nota: Esta seqüência, bem como a do passo seguinte, não especifica o elemento estruturante. Nestes casos, o comando `erode` remove pixels da 8-vizinhança de cada ponto localizado nas fronteiras dos objetos da imagem.

7. Executar mais uma vez o comando `erode`, agora com a opção `shrink`, que executa a erosão até que os objetos se transformem em pontos, preservando o número de Euler da imagem original.

```
H = erode(BW,'shrink');
subplot(2,2,4), imshow(~H,2);
```

8. Tanto o comando `erode` quanto o comando `dilate` permitem executar múltiplas operações em um único comando. Executar a seqüência a seguir para verificar este aspecto.

```
I = erode(BW,'thin',5);
clf
subplot(2,2,1), imshow(~BW,2)
subplot(2,2,2), imshow(~I,2)
J = dilate(BW,'thicken',5);
subplot(2,2,3), imshow(~J,2)
K = erode(J,'thin',5);
subplot(2,2,4), imshow(~K,2)
```

9. A seqüência seguinte exemplifica a operação de abertura utilizando a função `bwmorph` com a opção `open`, assim como a partir das opções `erode` e `dilate`, correspondentes às operações de erosão e dilatação.

```
clf
L = bwmorph(BW, 'erode');
M = bwmorph(L, 'dilate');
N = bwmorph(BW, 'open');
subplot(2,2,1), imshow(~BW,2)
subplot(2,2,2), imshow(~L,2)
subplot(2,2,3), imshow(~M,2)
subplot(2,2,4), imshow(~N,2)
```

10. A próxima seqüência exemplifica a operação de fechamento utilizando a função `bwmorph` com a opção `close`, bem como a partir das opções `erode` e `dilate`, correspondentes às operações de erosão e dilatação.

```

clg
P = bwmorph(BW, 'dilate');
Q = bwmorph(P, 'erode');
R = bwmorph(BW, 'close');
subplot(2,2,1), imshow(~BW,2)
subplot(2,2,2), imshow(~P,2)
subplot(2,2,3), imshow(~Q,2)
subplot(2,2,4), imshow(~R,2)

```

11. Para verificar o funcionamento do algoritmo de extração de contornos descrito na subseção 5.5.1, utilizaremos a seqüência:

```

load logo
BW = im2bw(X,map,.5);
BW = ~BW;
B = [1 1 1 ; 1 1 1 ; 1 1 1];
R1 = erode(BW, B, 'erode');
R2 = BW - R1;
subplot(2,2,1), imshow(~BW,2)
subplot(2,2,2), imshow(~R1,2)
subplot(2,2,3), imshow(~R2,2)

```

12. Para extrair o contorno dos objetos de uma imagem de maneira mais simples, podemos utilizar a função bwperim.

```

P1 = bwperim(BW);
subplot(2,2,3), imshow(~P1,2);

```

Comparar os resultados dos dois métodos (imagens 3 e 4).

13. No MATLAB, várias aplicações da transformação *hit-or-miss* já estão implementadas como parâmetros da função bwmorph. A partir de agora, veremos algumas destas aplicações, começando pelo preenchimento de regiões (*region filling*).

```

subplot(1,2,1), imshow(BW,2)
subplot(1,2,2), imshow(~bwmorph(BW,'fill',5),2)

```

14. Para verificar a opção thin da função bwmorph, que implementa o afinamento, utilize a seqüência:

```

clg
subplot(1,2,1), imshow(~BW,2)
subplot(1,2,2), imshow(~bwmorph(BW,'thin',5),2)

```

15. A operação de espessamento (*thickening*) pode ser implementada através da opção thicken, conforme a seqüência:

```

clg
subplot(1,2,1), imshow(~BW,2)
subplot(1,2,2), imshow(~bwmorph(BW,'thicken',5),2)

```

16. O esqueleto de um objeto pode ser obtido através da opção skel de bwmorph ou através da opção skeleton da função erode, conforme a seqüência:

```

clg
subplot(2,2,1), imshow(~BW,2)
subplot(2,2,2), imshow(~bwmorph(BW,'skel',5),2)
subplot(2,2,3), imshow(~erode(BW,'skeleton',5),2)

```

Comparar os resultados (2 e 3), verificando se são idênticos.

17. Verificar as opções bridge, clean, majority, remove e spur da função bwmorph através das seqüências:

```

clc
subplot(2,2,1), imshow(~BW, 2)
subplot(2,2,2), imshow(~bwmorph(BW, 'bridge', 5), 2)
subplot(2,2,3), imshow(~bwmorph(BW, 'remove', 5), 2)
subplot(2,2,4), imshow(~bwmorph(BW, 'majority', 5), 2)

BWR = imnoise(BW, 'salt & pepper');
subplot(1,2,1), imshow(~BWR, 2)
subplot(1,2,2), imshow(~bwmorph(BWR, 'clean', 5), 2)

clc
subplot(2,2,1), imshow(~BW, 2)
SK = ~bwmorph(BW, 'skel', 5);
subplot(2,2,2), imshow(SK, 2)
subplot(2,2,3), imshow(~bwmorph(~SK, 'spur', 5), 2)

```

18. Finalmente, verificaremos o funcionamento de um filtro morfológico usando a seqüência:

```

clear
clc
S = [0 0 0 0 0
 0 1 1 1 0
 0 1 1 1 0
 0 1 1 1 0
 0 1 1 1 0
 0 1 1 1 0
 0 1 1 1 0
 0 0 0 0 0];
B = imresize (S, [80 50], 'nearest');
C = ~B;
CR = imnoise(C, 'salt & pepper');
subplot(2,3,1), imshow(C,2)
subplot(2,3,2), imshow(CR,2)
EE = [1 1 1 ; 1 1 1 ; 1 1 1];
R1 = erode(~CR, EE);
subplot(2,3,3), imshow(~R1,2)
R2 = dilate(R1, EE);
subplot(2,3,4), imshow(~R2,2)
R3= dilate (R2, EE);
subplot(2,3,5), imshow(~R3,2)
R4 = erode(R3, EE);
subplot(2,3,6), imshow(~R4,2)

```

### Na Internet

Para pesquisas sobre os aspectos teóricos de cada prática, sugerimos consultar as indicações de endereço ao final dos capítulos 1 a 5. Para maiores informações sobre a MathWorks (fabricante do MATLAB), a *toolbox* de processamento de imagens e outros aspectos práticos relacionados a este apêndice, recomendamos os endereços a seguir:

"<http://www.mathworks.com/>"

The MathWorks Web Site

"<http://www.mathworks.com/products/image/>"

MathWorks: Image Processing Toolbox

"<http://education.mathworks.com/>"

Welcome to MATLAB In Education

### Bibliografia

[Russ 1995]

Russ, J. C., *The Image Processing Handbook - 2nd ed.*, CRC Press, 1995.

# Glossário

---

4-conectividade - um par de pixels vizinhos é dito 4-conectado se eles possuem um lado em comum.

8-conectividade - um par de pixels vizinhos é dito 8-conectado se eles possuem um lado ou um canto em comum.

Abertura - operação morfológica que suaviza o contorno geométrico dos objetos contidos em uma imagem. Essa operação é composta de uma operação de erosão, seguida de uma operação de dilatação.

Amostragem - método utilizado para a digitalização de um sinal analógico, no qual são retiradas amostras do sinal analógico em determinada freqüência. Ver Teorema de Nyquist.

Borda - mudança nos valores dos pixels (excedendo algum limiar) entre duas regiões de valores relativamente uniformes. Bordas correspondem a mudanças no brilho da imagem, as quais podem corresponder a uma descontinuidade na orientação ou refletância da superfície, ou ainda na iluminação.

Brilho - valor do nível de cinza de um pixel de uma imagem. Quanto maior é o valor do nível de cinza do pixel, maior é seu brilho.

CCD - Dispositivo de Carga Acoplada (*Charge Coupled Device*). É o elemento fotossensível utilizado em câmeras de estado sólido.

Codificação de Huffman - técnica de codificação que calcula a probabilidade de ocorrência dos valores em um conjunto de dados e atribui códigos de comprimento menor para os valores mais prováveis.

Codificação LZW (Lempel-Ziv-Welch) - método de codificação semelhante à codificação de Huffman, no qual as probabilidades são recalculadas quando o desempenho se altera.

Codificação *PackBits* - variante de implementação da codificação RLE utilizado nos formatos MacPaint e TIFF.

Codificação RLE - técnica simples de codificação de redundâncias, a qual consiste em pares de números. Um número representa um valor de pixel e o outro o número de vezes que esse valor se repete na sequência da imagem.

Compressão de Imagens - operação que preserva toda ou quase toda a informação da imagem ao mesmo tempo em que reduz a quantidade de memória necessária para armazená-la ou o tempo necessário para transmiti-la.

Conjunto - coleção de elementos que possuem alguma característica em comum.

Conjunto Nulo - conjunto que não possui elementos.

Contorno - linha que delimita objetos contidos em uma imagem.

Contraste - grau de variação dos níveis de cinza em uma imagem.

Conversor Analógico/Digital (A/D) - dispositivo eletrônico capaz de converter sinais analógicos contínuos em informação digital discreta.

Convolução Discreta - processo no qual duas imagens são combinadas através de operações de deslocamento, multiplicação e adição. Normalmente, uma das imagens é muito menor que a outra, sendo chamada de janela ou máscara de convolução. Máscaras podem ser projetadas para realizar uma ampla gama de funções de filtragem.

Correlação - correspondência entre os pixels de uma imagem e outra de referência.

Córnea - superfície transparente externa do olho humano, a qual realiza o processo inicial de focalização.

Cursor - objeto gráfico utilizado para identificar a localização de um dispositivo apontador, tal qual um mouse.

Deteção de Bordas - técnica de determinação dos contornos dos objetos contidos em uma imagem.

Digitalizador - dispositivo eletrônico capaz de converter sinais analógicos contínuos em informação digital discreta. Ver Conversor Analógico/Digital (A/D).

Dilatação - operação morfológica que aumenta o tamanho geométrico de objetos contidos em uma imagem.

Discreto - referente a sinais ou dados divididos em amostras ou quantidades fixas.

Dispositivos de Estado Sólido - componentes eletrônicos feitos de material semicondutor cuja composição é inteiramente sólida, diferentemente das válvulas eletrônicas, as quais fazem uso de vácuo para o seu funcionamento.

Distância Euclidiana - distância medida através da raiz quadrada da soma dos quadrados das diferenças entre as coordenadas dos pontos em questão.

Distância  $D_4$  - distância medida através da soma dos módulos das diferenças entre as coordenadas dos pixels em questão.

Distância  $D_8$  - distância medida através da soma do máximo valor entre os módulos das diferenças entre as coordenadas dos pixels em questão.

*Dithering* - termo utilizado para descrever algoritmos que simulam representações em níveis de cinza em dispositivos cuja saída é binária, tais como impressoras monocromáticas.

Elemento Estruturante - conjunto de pixels utilizado para descrever a função estruturante utilizada em operações morfológicas de erosão, dilatação e operações derivadas.

Entrelaçamento - processo de varredura que salta uma linha a cada linha varrida, em métodos de sensoriamento ou apresentação de imagens. Cada quadro é dividido em dois campos: um de linhas ímpares e outro de linhas pares, com o objetivo de reduzir a cintilação que ocorre com o processo de varredura não entrelaçada.

Equalização de Histograma - processo que procura converter o histograma de uma imagem numa distribuição uniforme. O efeito da equalização de histograma é a melhoria do contraste da imagem.

Erosão - operação morfológica que reduz o tamanho geométrico de objetos contidos em uma imagem.

Escala de Cinza - faixa de níveis de cinza correspondentes aos valores dos pixels em uma imagem monocromática.

Espectro - conjunto ordenado de raias que descrevem o conteúdo em frequência de um conjunto de dados (ou sinal). Ver Transformada Discreta de Fourier.

Espectro Eletromagnético - faixa de comprimentos de onda de energia conhecidos e seus respectivos nomes.

Espectro Visível - porção do espectro eletromagnético que é visível ao olho humano.

Esqueletização - processo de obtenção da forma estrutural básica de um objeto em uma imagem, de apenas um pixel de espessura.

Fechamento - operação morfológica que suaviza o contorno geométrico dos objetos contidos em uma imagem. Essa operação é composta de uma operação de dilatação, seguida de uma operação de erosão.

Filtragem Homomórfica - processo no qual um filtro espacial é aplicado ao logaritmo da imagem original e o resultado final é obtido pela exponenciação da filtragem.

Filtros Adaptativos - filtros que mudam suas características conforme a imagem à qual são aplicados e ao tipo de ruído nela presente.

Filtro Espacial - filtro bidimensional que opera na distribuição espacial de valores de pixels em uma pequena vizinhança. Apesar dos filtros de frequência espacial operarem em distribuições espaciais de pixels, o termo filtro espacial é geralmente reservado para denominação de convoluções discretas, enquanto o termo anterior é utilizado para filtros derivados de transformadas matemáticas.

Filtro da Média - filtro que substitui o pixel central da máscara de convolução pela média dos valores de nível de cinza sob a máscara.

Filtro de Máximo - filtro que substitui o pixel central da máscara de convolução pelo máximo dos valores de nível de cinza sob a máscara.

Filtro da Mediana - filtro que substitui o pixel central da máscara de convolução pela mediana dos valores de nível de cinza sob a máscara.

Filtro de Mínimo - filtro que substitui o pixel central da máscara de convolução pelo mínimo dos valores de nível de cinza sob a máscara.

Filtro Passa-Altas - filtro espacial linear que atenua as freqüências espaciais mais baixas de uma imagem e acentua as mais altas. Normalmente é utilizado para destacar pequenos detalhes, bordas e linhas.

Filtro Passa-Baixas - filtro espacial linear que atenua as freqüências espaciais mais altas de uma imagem e acentua as mais baixas. Normalmente é utilizado para remover pequenos detalhes indesejados, eliminar algum tipo de ruído ou suavizar imagens.

Filtro Passa-Faixa - filtro espacial linear que atenua as freqüências espaciais de uma imagem que estão fora da sua faixa de atuação e acentua as que estão dentro. Possui pouca utilização prática no processamento de imagens.

*Frame buffer* - memória digital projetada para armazenar uma imagem ou um conjunto de imagens que foram capturadas por um digitalizador ou frame grabber.

*Frame grabber* - circuito eletrônico que converte (digitaliza) sinal analógico de vídeo em uma imagem digital.

Freqüência - medida da periodicidade de um conjunto de dados, ou seja, quanto freqüentemente padrões são repetidos com respeito a uma determinada medida como tempo ou distância.

Freqüência Espacial - medida da periodicidade de um conjunto de dados com respeito a uma medida de distância. Mudanças periódicas em valores de brilho em uma imagem são definidos em termos de freqüência espacial.

*Gamma* - medida básica de contraste. Na terminologia eletrônica, *gamma* é a inclinação da curva de distribuição de brilho de um dispositivo de saída como um monitor. Um valor *gamma* alto significa uma inclinação elevada e, consequentemente, alto contraste.

*Halftoning* - técnica para proporcionar efeito de escala de cinza em dispositivos de saída binária, tais quais impressoras monocromáticas. Ver dithering.

Histograma - distribuição dos valores dos níveis de cinza. É um gráfico do número de pixels em cada nível de cinza possível em uma imagem, constituindo a distribuição de probabilidade dos valores dos pixels. Pode ser processado através de técnicas estatísticas, as quais resultam em mudanças no brilho e contraste da imagem, independentes da distribuição espacial dos pixels.

Iluminação - fonte externa de energia que ilumina uma cena ou imagem.

Iluminância - medida de intensidade luminosa incidente em determinado ponto ou região.

Imagen - projeção de uma cena em um plano, normalmente representada como uma matriz de valores de brilho.

Imagen Binária - imagem na qual os pixels assumem apenas dois valores, geralmente 0 ou 1.

Imagen Digital - obtida pela partição da área da imagem em uma matriz bidimensional finita, cujas células (pixels) recebem valores correspondentes à intensidade luminosa naquela região.

Imagen Monocromática - imagem cujos pixels podem assumir uma faixa de valores variando do preto ao branco (nível de cinza).

Imagen Padrão - 512 x 512 pixels, com quantização de 8 bits por pixel.

Interseção - região comum a dois objetos ou conjuntos.

Irradiância - medida de intensidade luminosa irradiada por uma fonte de luz.

LCD - Display de Cristal Líquido (*Liquid Crystal Display*). Dispositivo que faz uso do efeito de atenuação de luz, apresentado por cristais amorfos, para criar saída visível em um monitor.

Limiarização - técnica de segmentação dos níveis de cinza em duas regiões diferentes, também chamada de binarização de uma imagem. Determina-se um valor de limiar e todos os valores dos pixels menores ou iguais a esse valor são mapeados em 0, enquanto os demais são mapeados em 1.

Limiarização Ótima - processo de determinação do melhor valor de limiar para uma imagem particular, com base em suas propriedades estatísticas.

m-conectividade - abreviação de conectividade mista. Consiste em uma modificação da 8-conectividade, visando eliminar possíveis duplicidades de conexão entre dois pixels.

Máscara de Convolução - matriz pequena, normalmente de tamanho 3x3 a 7x7, usada como filtro em operações de convolução.

Média - resultado da soma de um conjunto de valores dividida pelo total de itens somados.

Mediana - valor central de um conjunto de valores ordenados.

Monitor - dispositivo utilizado para apresentar imagens ou dados computacionais.

Monocromático - termo utilizado para descrever imagens em branco e preto ou em níveis de cinza.

Morfologia Matemática - área do processamento de imagens que leva em consideração a análise do formato dos objetos contidos em uma imagem.

Nervo Ótico - nervo que transporta informações do olho para o cérebro humano.

Nível de Cinza - valor de um pixel numa imagem monocromática.

NTSC - Comitê Nacional de Padrões de Televisão dos EUA (*National Television Standards Committee*). Termo utilizado para descrever o sistema de televisão em cores americano.

Números Complexos - sistema numérico representado por um par de valores reais a e b, na forma  $a+jb$ , onde a é chamado de parte real (Re) e b é chamado de parte imaginária (Im). O termo imaginário é utilizado devido ao segundo valor, b, ser multiplicado pelo operador imaginário  $j$ , equivalente à raiz quadrada de -1. É simplesmente uma convenção que permite fácil representação de funções dependentes de freqüência. A utilidade dos números complexos é revelada quando espectros de fase e magnitude são obtidos dos resultados da Transformada de Fourier.

PAL - Linha de Fase Alternante (*Phase-Alternating Line*). Termo utilizado para descrever o sistema de televisão em cores europeu e brasileiro.

Pixel - abreviação de elemento de imagem (*picture element*). Denominação técnica para o menor elemento constituinte de uma imagem digital. São dispostos em linhas e colunas para compor um quadro. Também denominado pel.

Periódico - um conjunto de dados (ou sinal) é chamado periódico quando contém padrões que se repetem ao longo do tempo. Período é o nome dado ao subconjunto que contém o padrão que se repete.

Plumbicon - tubo de câmera à vácuo, utilizado para aquisição de imagens pela varredura de uma tela fotossensível através de um feixe de elétrons.

Processamento de Imagens - processo de transformação de uma imagem em uma outra com propriedades mais desejáveis, tais como menor ruído, menor distorção geométrica, maior nitidez, etc.

Processador Matricial - computador especificamente projetado para realizar cálculos matriciais (ou em imagens) com eficiência.

Pupila - abertura do olho humano para a entrada de luz.

Quadro (*Frame*) - termo utilizado para descrever uma imagem, normalmente no contexto de conjunto: uma imagem dentro de uma seqüência.

Quantização - processo de mapeamento de uma ampla faixa de valores de entrada para um número limitado de valores de saída.

Realce de Imagens - uso de técnicas de processamento de imagens para acentuar certas propriedades e melhorar a qualidade de informação recebida de uma imagem.

Relação de Aspecto - relação entre altura e largura de uma imagem capturada por uma câmera ou exibida em um monitor.

Reconstrução de Imagens - algoritmos que buscam construir imagens bidimensionais a partir de dados unidimensionais ou imagens tridimensionais a partir de dados bidimensionais.

Recortar e Colar (*Cut and Paste*) - processo de delimitação de uma área de uma imagem, removendo-a (recortando-a) ou adicionando-a (colando-a) à mesma imagem ou a outra imagem diferente.

Refletância - percentual de intensidade luminosa incidente que é refletida pelos objetos em uma imagem.

Região - conjunto de pixels conectados com alguma propriedades em comum, tal como a mesma faixa de nível de cinza em uma imagem.

Resolução - a menor característica de uma imagem que pode ser percebida por um sistema de aquisição de imagens. É geralmente dependente do número de pixels presentes na imagem: quanto maior o número de pixels, maior a resolução.

Restauração de Imagens - algoritmos que buscam remover degradações em imagens (ruído, falta de foco, borrados, etc.) baseados em critérios objetivos que buscarão recuperar informações presumivelmente presentes na imagem original.

Retina - área sensora de luz do olho humano.

RS-170 - padrão de transmissão do sinal composto de vídeo, o qual inclui sinais de sincronismo horizontal e vertical. Possui 525 linhas, entrelaçadas em 1/30 de segundo.

Ruído Gaussiano - tipo de ruído cujo histograma possui a forma gaussiana (sino).

Ruído Sal e Pimenta - tipo de ruído que contém apenas dois valores, um próximo ao valor máximo (sal) e o outro próximo ao valor mínimo (pimenta) da escala de cinza.

Ruído Uniforme - tipo de ruído cujo histograma é uniforme.

Saticon - tubo de câmera à vácuo, utilizado para aquisição de imagens pela varredura de uma tela fotossensível através de um feixe de elétrons.

SECAM - Memória Seqüencial (*Sequential à Memoire*). Sistema de televisão em cores francês.

Segmentação - processo de divisão da imagem em um número definido de regiões individuais, ou segmentos.

Sinal Analógico - sinal representado através de uma função contínua.

Sinal Composto de Vídeo (SCV) - sinal elétrico variante no tempo, cuja amplitude representa o brilho da imagem em determinado ponto da tela, cuja localização é determinada pela varredura. O SCV inclui sinais de sincronismo horizontal e vertical para o correto funcionamento do sistema de varredura.

Suavização de imagens - qualquer filtragem espacial que produz em sua saída uma imagem com menos detalhes que a imagem original. O processo de suavização tende a eliminar pequenos detalhes, destacando grandes estruturas da imagem.

*Template* - subimagem utilizada para operações de correlação ou comparação. Algumas vezes o termo é usado para descrever uma máscara de convolução. Ver convolução discreta.

Teorema de Nyquist - teorema da amostragem, o qual determina que um sinal deve ser amostrado a uma taxa de amostragem pelo menos duas vezes superior à maior componente de freqüência presente no sinal. Quando a taxa de amostragem de Nyquist é utilizada, todas as componentes do sinal amostrado serão adequadamente representadas, garantindo que o sinal contínuo poderá ser corretamente recuperado a partir de sua versão amostrada, posteriormente.

Transformada Discreta de Fourier (DFT - *Discrete Fourier Transform*) - transformação matemática realizada em dados discretos, a qual fornece informações sobre as componentes senoidais dos dados sob análise, sob a forma de conteúdo espacial de freqüência.

Transformada Rápida de Fourier (FFT - *Fast Fourier Transform*) - formulação especial da transformada discreta de Fourier que faz uso de fórmulas repetitivas para aumentar a eficiência dos cálculos.

Tubo de Raios Catódicos (TRC ou CRT) - nomenclatura técnica para o popularmente chamado tubo de imagem de um monitor, que permite mostrar imagens e gráficos através do posicionamento eletrônico de um feixe de elétrons em uma tela fotossensível.

União - processo de combinação de dois conjuntos em apenas um.

Variância - valor médio do quanto difere um conjunto de dados da média do próprio conjunto. Formalmente é o valor médio dos quadrados dos desvios da média.

Vídeo - sinal analógico que carrega informações de imagens. Ver Sinal Composto de Vídeo.

Vidicon - tubo de câmera à vácuo, utilizado para aquisição de imagens pela varredura de uma tela fotossensível através de um feixe de elétrons.

Vizinhança - região que circunda um pixel.

*Warping* - algoritmo utilizado para realizar uma operação de distorção geométrica em uma imagem

*Zoom* - algoritmo utilizado para ampliar uma imagem para efeito de visualização.



# **Figuras Coloridas**

---

## Capítulo 3

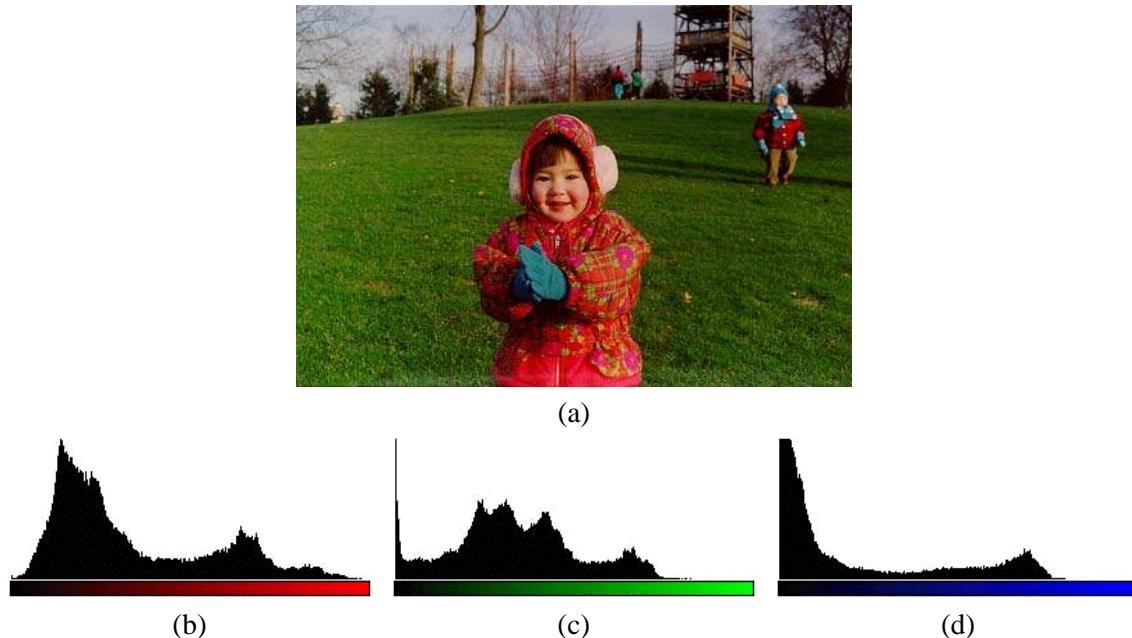


Figura 4 - (a) Imagem colorida e histogramas de seus componentes: (b) R, (c) G e (d) B.

## Capítulo 4

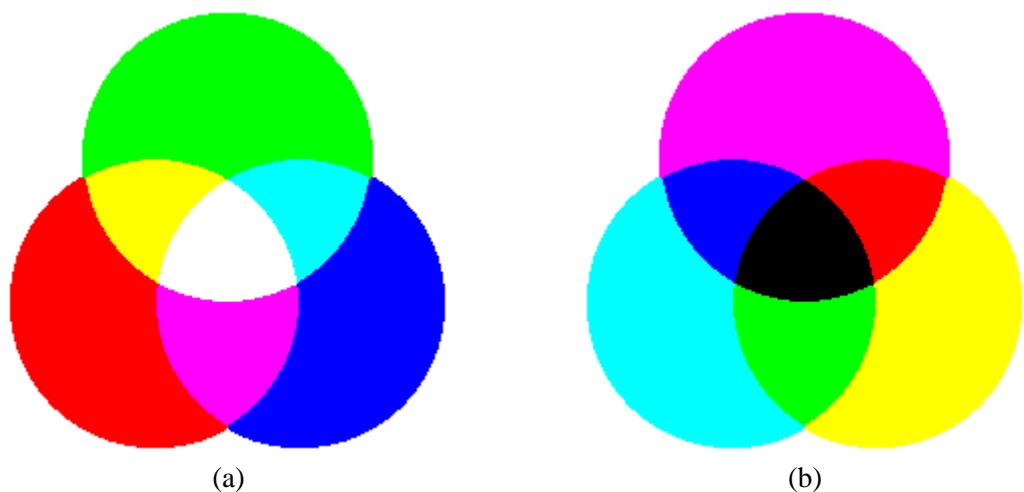


Figura 35 - Mistura de cores primárias e secundárias: (a) mistura aditiva; (b) mistura subtrativa.



Figura 38 - (a) Imagem colorida; (b) componente  $R$ ; (c) componente  $G$ ; (d) componente  $B$ .



Figura 39 - (a) Imagem colorida e seus componentes: (b)  $H$ , (c)  $S$  e (d)  $I$ .

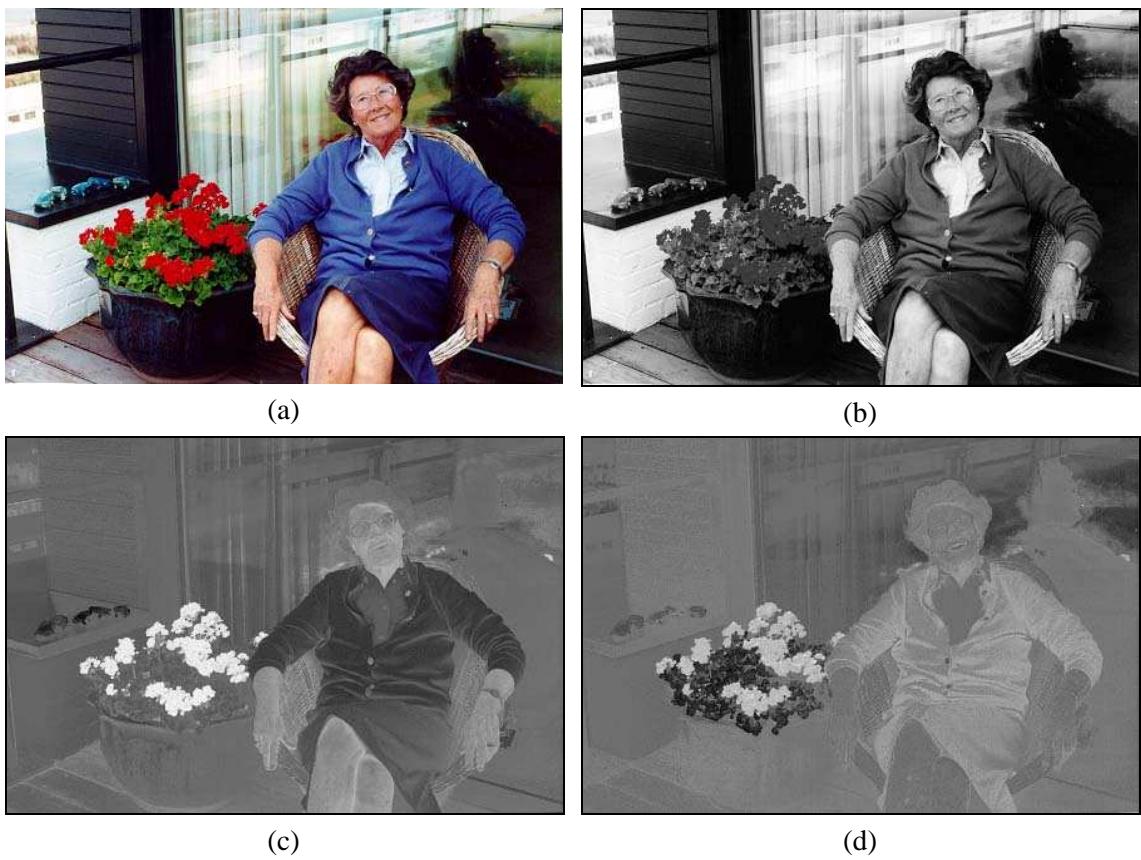


Figura 40 - (a) Imagem colorida e seus componentes: (b)  $Y$ , (c)  $I$  e (d)  $Q$ .



Figura 42 - Equalização de histograma aplicada a imagens coloridas.

# Processamento Digital de Imagens

As áreas de Processamento e Análise de Imagens, Visão por Computador e correlatas vêm apresentando expressivo desenvolvimento na última década. Tal crescimento pode ser detectado na área acadêmica, onde o assunto é objeto de pesquisas, teses e dissertações nas mais importantes universidades brasileiras e estrangeiras; na esfera industrial, onde a cada dia aumenta o número de empresas que produzem, comercializam e utilizam soluções de Processamento Digital de Imagens em seus processos; e na vida cotidiana, com a popularização dos micro-computadores pessoais e das aplicações multimídia.

Este livro é resultado de quase dez anos de experiência na docência e pesquisa nas áreas de Visão por Computador e Processamento de Imagens e aborda temas clássicos e obrigatórios relacionados a esta área de conhecimento, bem como assuntos inovadores de grande interesse, tanto para aqueles que estão travando um primeiro contato com o assunto quanto para estudiosos da área. Contém exercícios resolvidos e propostos, roteiros de laboratório e sugestões de endereços na Internet para maiores informações sobre os assuntos de cada capítulo. Os apêndices trazem informações complementares de grande interesse para leitores iniciantes e para aqueles que dispõem de micro-computador e desejam implementar e testar rotinas de Processamento Digital de Imagens, complementando o material teórico de cada capítulo.



Ogé Marques Filho é Engenheiro Eletrônico, formado pelo Centro Federal de Educação Tecnológica do Paraná (CEFET-PR) e Mestre em Engenharia Eletrônica pelo Philips International Institute (Eindhoven, Holanda). Possui mais de 15 anos de experiência acadêmica em Instituições de Ensino brasileiras e atualmente é docente do Departamento de Ciência e Engenharia da Computação da Florida Atlantic University (Boca Raton, Florida - USA).



Hugo Vieira Neto é Engenheiro Eletrônico, formado pelo CEFET-PR em 1995 e é Mestre em Engenharia Elétrica e Informática Industrial, também pelo CEFET-PR. Atualmente é docente do Departamento de Eletrônica e Telecomunicações desta mesma instituição.



BRASPORT

