Capítulo 2

Operadores Morfológicos

Uma forma elegante de resolver problemas de processamento de imagens é através da utilização de uma base teórica consistente. Uma destas teorias é a *morfologia matemática* criada na década de 60 por Jean Serra e George Matheron na *École Nationale Superiéure des Mines de Paris*, em Fontainebleau, França. Esta teoria diz que é possível fazer transformações entre reticulados completos[1], os quais são chamados de *operadores morfológicos*. Na morfologia matemática existem quatro classes básicas de operadores: dilatação, erosão, anti-dilatação e anti-erosão, chamadas de *operadores elementares*. Banon e Barrera [BB93] provaram que todos os operadores morfológicos invariantes por translação podem ser obtidos a partir de combinações de operadores elementares juntamente com as operações de união e intersecção. Além disso, quando um reticulado possui uma *família sup-geradora*, estes operadores podem ser caracterizados por *funções estruturantes*. Usando estes operadores elementares é possível construir uma *linguagem formal*, a *linguagem morfológica*, e sua implementação é chamada *máquina morfológica* [BB92]. Um exemplo de uma máquina morfológica é a *MMach* [BBL94].

[1]Um conjunto qualquer com uma relação de ordem é um reticulado completo se todo subconjunto não vazio tem um supremo e um ínfimo. Para detalhes da teoria dos reticulados veja [Bir67].

A teoria dos reticulados estudada em morfologia matemática é abrangente e o leitor interessado pode consultar, por exemplo, os trabalhos de Serra, Banon e Barrera [Ser88, BB94]. Banon e Barrera [BB94] também fizeram estudos de caracterização de operadores morfológicos por funções estruturantes. Zampirolli [Zam97] fez um estudo de *operadores baseados em grafos*. Porém as distâncias em um grafo estão relacionadas aos curtos das arestas ou vértices.

Serão apresentados neste capítulo estudos de casos da dilatação e da erosão caracterizadas por funções estruturantes nos reticulados das imagens binárias e em níveis de cinza.

2.1 Erosão e dilatação binária

Existem várias formas de escrever a erosão e a dilatação binária. Uma das mais antigas é a *subtração de Minkowski* e a *soma de Minkowski*, respectivamente. Seja $\mathbb{E}$ um grupo Abeliano[2]. Sejam X e B dois subconjuntos de $\mathbb{E}$. A *subtração de Minkowski* é um subconjunto de $\mathbb{E}$ definida da seguinte forma:

$$X \ominus B = \{x \in \mathbb{E} : B_x \subset X\}, \tag{2.1}$$

onde B_x é a *translação* de B por x [BB94]. Esta translação pode ser interpretada como a vizinhança B de x [Hei94]. Em morfologia matemática, o conjunto B é chamado de *elemento estruturante*. A notação da subtração de Minkowski também é particular, $\varepsilon_B(X)$ – lê-se erosão de X por B. Analogamente, a *soma de Minkowski* é definida da seguinte forma:

$$X \oplus B \;\; = \;\; \cup\{X_y : y \in B\}. \tag{2.2}$$

[2]Um conjunto com uma operação de adição e as propriedades comutativa, associativa, existência do elemento neutro e existência do oposto é um grupo Abeliano.

A notação usada em morfologia matemática para a soma de Minkowski é $\delta_B(X)$ – lê-se dilatação de X por B.

Seja X^c o *complemento* de X. Então a *subtração de Minkowski* pode ser obtida através da *soma de Minkowski* e vice-versa [WB88], isto é,

$$X \ominus B = (X^c \oplus B)^c.$$

Serão apresentadas a seguir algumas definições para deixar conceitualmente consistentes as implementações que serão descritas a partir do próximo capítulo e a teoria de morfologia matemática.

Sejam $\mathbb{E}$ e K dois conjuntos quaisquer finitos e não vazios. Um *operador* ψ entre $\mathbb{E}$ e K é definido como um mapeamento de $\mathbb{E}$ em K e denotado $\psi : \mathbb{E} \to K$ ou $\psi \in K^{\mathbb{E}}$.

O conjunto $\mathcal{P}(\mathbb{E})$ de todos os subconjuntos de $\mathbb{E}$, com a relação usual de inclusão $\subset$ entre subconjuntos, é um reticulado completo e é chamado de conjunto das partes de $\mathbb{E}$. As operações de ínfimo e supremo são, respectivamente, as operações de intersecção e união entre subconjuntos. Um elemento $X \subset \mathcal{P}(\mathbb{E})$ pode representar, por exemplo, uma *imagem binária* se $X \in \{0,1\}^{\mathbb{E}}$ – o valor de um elemento de $\mathbb{E}$ é 1 se ele pertence a X e 0, caso contrário.

Seja uma *imagem digital*, ou simplesmente *imagem*, como sendo uma função do reticulado $K^{\mathbb{E}}$. Assim, se f é uma imagem então $f \in K^{\mathbb{E}}$. O conjunto $\mathbb{E}$ representa o *domínio da imagem* podendo ser *unidimensional* se $\mathbb{E} \subset \mathbb{Z}$, onde $\mathbb{Z}$ representa o conjunto dos números inteiros e *bidimensional*, se $\mathbb{E} \subset \mathbb{Z} \times \mathbb{Z}$. Existem também imagens tridimensionais, imagens com domínio em uma grade hexagonal, imagens representadas por grafos, etc. Sem perda de generalidade, seja a origem de uma imagem definida no pixel zero no caso unidimensional (pixel mais à esquerda) e $(0,0)$ no caso bidimensional (topo esquerdo da imagem). O conjunto K representa o *contra-domínio da imagem* ou os valores que os pontos da imagem podem assumir. Para as imagens binárias considere $K = \{0, k\}$, onde k é um valor diferente de 0, por exemplo,

$k = 1$ ou $k = \infty$. Para as imagens em níveis de cinza considere o intervalo $K = [0, k]$, onde $k = 255$ para imagens com níveis de cinza representadas no tipo *byte*. Para representação computacional será definida uma imagem em uma matriz com domínio em $\mathbb{E}$ e o tipo desta matriz representa os níveis de cinza [Vin92]. Observe que esta representação se refere às imagens tratadas neste texto, ou seja, imagens unidimensionais ou bidimensionais, dos tipos binários ou em níveis de cinza. Para imagens representadas por grafos, por exemplo, a estrutura de dados é mais complexa. Apesar desta variedade de possibilidades de tipos de imagens, todos os algoritmos apresentados nesta tese são genéricos quanto ao domínio $\mathbb{E}$. O que muda são as implementações nas linguagens de programação de propósito geral, como *C* e *MATLAB*.

2.1.1 Decomposição de elementos estruturantes

As seguintes propriedades das operações de Minkowski produz um método para a *decomposição de elemento estruturante* [WB88]:

$$\begin{aligned} \varepsilon_{A\oplus B}(X) &= \varepsilon_A(\varepsilon_B(X)) \quad e \\ \delta_{A\oplus B}(X) &= \delta_A(\delta_B(X)). \end{aligned}$$

Por exemplo, uma dilatação por um elemento estruturante 3×3 é equivalente a realizar duas dilatações unidimensionais 1×3 e 3×1. Outro exemplo é obter o elemento estruturante *octagonal* a partir de dois elementos estruturantes 3×3. Como ilustrados, respectivamente, nas duas somas de Minkowski da Figura 2.1.

Como conseqüência desta decomposição, seja a *soma de Minkowski generalizada* definida como segue:

$$B_G = B_1 \oplus \cdots \oplus B_k,$$

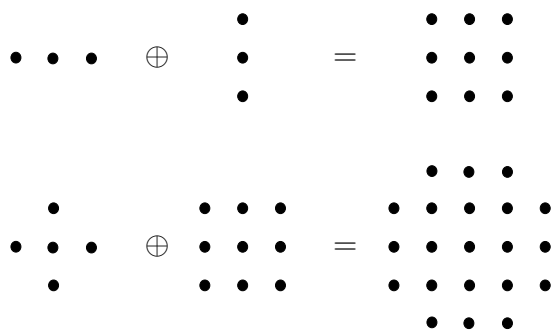

Figura 2.1: Exemplos de decomposições de elementos estruturantes.

onde $\{B_1, \cdots, B_k\}$ são elementos que decompõem B_G. Assim,

$$\varepsilon_{B_G}(f) = \varepsilon_{B_k}(\cdots(\varepsilon_{B_1}(f))\cdots) \quad e \tag{2.3}$$

$$\delta_{B_G}(f) = \delta_{B_k}(\cdots(\delta_{B_1}(f))\cdots). \tag{2.4}$$

Quando existe um único B que decompõe B_G, escrevemos,

$$B_G = \underbrace{B \oplus \cdots \oplus B}_{k \ vezes},$$

ou simplesmente $B_G = kB$.

É evidente que tais procedimentos de decomposição são úteis com respeito a implementações da erosão e da dilatação [Hei94], como serão apresentados nos próximos capítulos.

2.2 Tipos de elementos estruturantes

Observe que não foi definido o domínio da imagem nas Equações 2.1 e 2.2. Para tratar deste caso, serão definidas as *transformações infinitas*, onde o domínio da imagem e o domínio da transformação não são necessariamente os mesmos. Uma transformação muito utilizada na prática são

as *transformações limitadas*, onde a imagem e o resultado da transformação pertencem ao mesmo domínio $\mathbb{E}$.

Uma aplicação de *transformação infinita* são os operadores *invariantes por translação* (i.t.), isto é, ψ é invariante por translação se e somente se

$$\psi(B_x) = (\psi(B))_x,$$

onde $B_x = B + x = \{y + x,\ y \in B\}$ é a *translação* de B por $x \in \mathbb{E}$.

Uma *função estruturante* b será definida por $b \in \mathbb{Z}^B$. Assim, em morfologia matemática também é possível aplicar as Equações 2.1 e 2.2 em funções estruturantes em vez de elementos estruturantes. Porém, é comum usar *elemento estruturante* para transformações morfológicas em imagens binárias (ou conjuntos) e *função estruturante* para transformações morfológicas em imagens em níveis de cinza. Observe que um elemento estruturante pode ser visto como o domínio da função estruturante, pois $b \in \mathbb{Z}^B$, e as transformações morfológicas binárias podem ser vistas como um caso particular de transformações em níveis de cinza ($\{0,1\}^{\mathbb{E}} = K^{\mathbb{E}}$).

Algumas possíveis escolhas para funções estruturantes, considerando $x \in \mathbb{E}$ e $B \subseteq \mathbb{E} \oplus \mathbb{E}$, são [BB94]:

I. *transformação infinita*

$$b(x) = B_x;$$

II. *transformação limitada*

$$b(x) = B_x \cap \mathbb{E}.$$

Veja na Figura 2.2 um exemplo de dilatação binária limitada.

As funções estruturantes *quase invariantes por translação* (q.i.t.) são exemplos de transformações limitadas cujo domínio é $\mathbb{E}$. Mais detalhes de funções estruturantes i.t. e q.i.t. podem ser encontrados em [BB94].

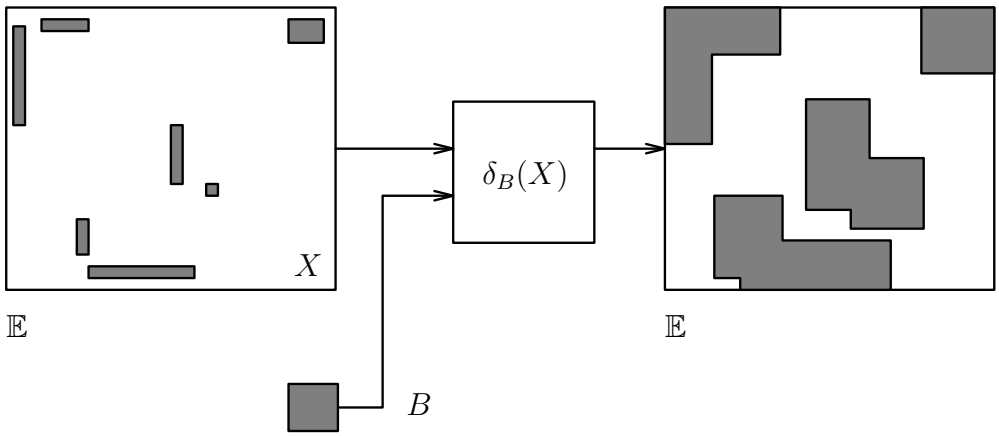

Figura 2.2: Dilatação binária limitada de X por B.

2.3 Erosão e dilatação em níveis de cinza

Nesta seção serão apresentadas a erosão e a dilatação no reticulado das imagens em níveis de cinza $K^{\mathbb{E}}$.

Seja $\mathbb{E}$ um retângulo finito de $\mathbb{Z} \times \mathbb{Z}$, onde $\mathbb{Z}$ é o conjunto dos números inteiros e $K \subset \mathbb{Z}$ é o conjunto finito dos níveis de cinza. Então $K^{\mathbb{E}}$ pode representar todas as possíveis *imagens em níveis de cinza*. A coleção $K^{\mathbb{E}}$ com a relação de ordem parcial $\leq$ dada por, $\forall f_1, f_2 \in K^{\mathbb{E}}$, $f_1 \leq f_2 \Leftrightarrow f_1(x) \leq f_2(x)$ e $\forall x \in \mathbb{E}$, é um reticulado completo. As operações de ínfimo e supremo são as operações de mínimo e máximo usuais.

Para tornar viável as implementações das transformações morfológicas em níveis de cinza, serão definidos os operadores $\dot{-}$ e $\dot{+}$ que limitam as transformações no intervalo K da seguinte forma [Hei91]:

Definição 2.1 *Seja K um intervalo $[0,k]$ de $\mathbb{Z}$ com $k>0$. Sejam v e t inteiros, então $t \to t\dot{-}v$ em K é definido por*

$$t\dot{-}v = \begin{cases} 0 & se\ t<k\ e\ t-v\leq 0, \\ t-v & se\ t<k\ e\ 0\leq t-v\leq k, \\ k & se\ t<k\ e\ t-v>k, \\ k & se\ t=k. \end{cases}$$

Analogamente, o operador $t \to t\dot{+}v$ em K é definido por

$$t\dot{+}v = \begin{cases} 0 & se\ t=0, \\ 0 & se\ t>0\ e\ t+v\leq 0, \\ t+v & se\ t>0\ e\ 0\leq t+v\leq k, \\ k & se\ t>0\ e\ t+v>k. \end{cases}$$

◇

Existem várias formas de escrever a erosão e a dilatação em níveis de cinza. Serão definidas a seguir a erosão e a dilatação por uma função estruturante que serão as mais usadas no decorrer do texto.

A erosão e a dilatação podem ser descritas, respectivamente, pelas expressões [Hei91]: $\forall f \in K^{\mathbb{E}}$, $\forall x \in \mathbb{E}$ e $b \in \mathbb{Z}^B$,

$$\varepsilon_b(f)(x) = \min\{f(y)\dot{-}b(y-x) : y \in B_x \cap \mathbb{E}\} \quad e \tag{2.5}$$

$$\delta_b(f)(x) = \max\{f(y)\dot{+}b(x-y) : y \in B_x \cap \mathbb{E}\}, \tag{2.6}$$

onde B_x é a *translação* de B por x. Veja na Figura 2.3 uma ilustração da dilatação em níveis de cinza.

2.4 Diferentes formas de escrever a erosão

Serão apresentados a seguir três exemplos diferentes de escrever a erosão morfológica. Estas erosões estão classificadas no padrão paralelo, descrito

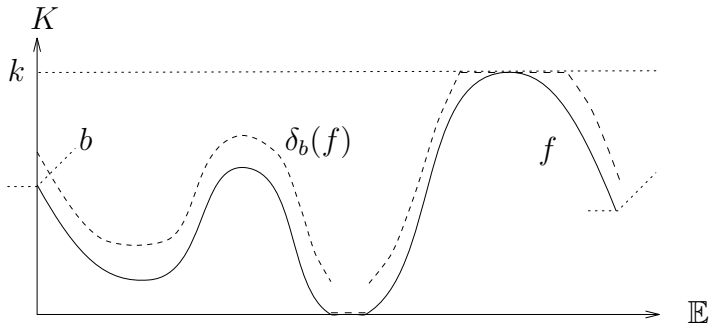

Figura 2.3: Dilatação de uma imagem em níveis de cinza f por uma função estruturante b.

com detalhes no próximo capítulo.

Exemplo 1

Na Equação 2.5, é apresentada a primeira forma de escrever a erosão morfológica aplicada em imagens em níveis de cinza. Esta definição é a mais conhecida na literatura e é ilustrada na Figura 2.4. O valor da erosão no ponto x é o mínimo de $f(y)\dot{-}b(y-x)$ na janela dada pelo elemento estruturante centrado em x.

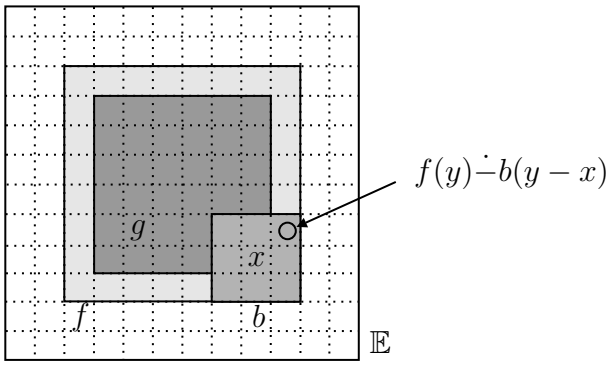

Figura 2.4: Primeiro exemplo de erosão.

Exemplo 2

Outra forma de escrever a erosão de uma imagem f por uma função estruturante b é apresentada na equação a seguir e ilustrada na Figura 2.5: $\forall f \in K^{\mathbb{E}}$ e $b \in \mathbb{Z}^B$,

$$\varepsilon_b(f) \quad = \quad \min\{(f_y \cap \mathbb{E}) \dot{-} b(y) : y \in B\}. \tag{2.7}$$

A principal diferença deste exemplo para o exemplo anterior é relativa ao parâmetro f_y, que tem como retorno uma imagem transladada por y.

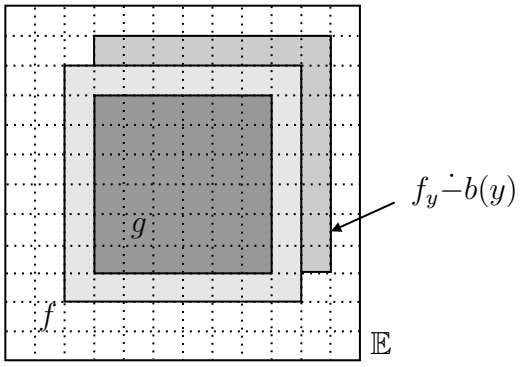

Figura 2.5: Segundo exemplo de erosão.

Exemplo 3

Neste último exemplo, a erosão de cada pixel x da imagem g é obtida fazendo a redução através da operação de *mínimo* de todas as sub-imagens de f subtraídas de b (veja também Figura 2.6): $\forall f \in K^{\mathbb{E}}$ e $b \in \mathbb{Z}^B$, $\forall x \in \mathbb{E}$

$$\varepsilon_b(f)(x) \quad = \quad \min\{(f(x)_{y:\ \forall y \in B} \cap \mathbb{E}) \dot{-} b\}. \tag{2.8}$$

Observe neste exemplo que $f' = f(x)_{y,\ \forall y \in B} \cap \mathbb{E}$ retorna uma sub-imagem de f com centro no pixel x tendo o mesmo domínio de b.

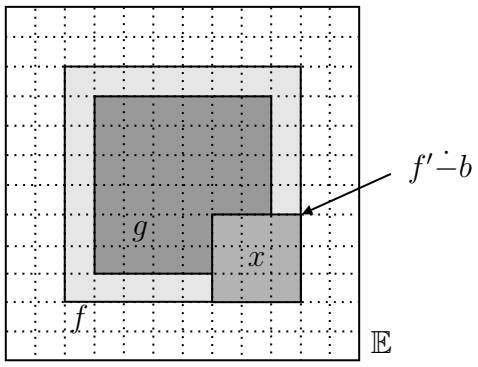

Figura 2.6: Terceiro exemplo de erosão.

2.5 Notação

Será apresentada nesta seção a notação utilizada para descrever os algoritmos neste documento a partir do próximo capítulo. Esta notação foi inspirada na *notação Z* e foi usada para gerar código de forma automática, como descrito no Apêndice A. Z é utilizada para a especificação de problemas em engenharia de software através de expressões matemáticas [Pre02].

Para declarar uma variável pertencente a um tipo será utilizado o formato:

$nome\_variavel : tipo$;

onde *nome_variavel* pode ser letras ou números. Outra possibilidade é fazer

$nome\_variavel \in tipo$;

Para declarar variáveis do mesmo tipo seja:

$nome\_variavel\_1, \ldots, nome\_variavel\_n : tipo$; // declarações

onde n é um número natural e os comentários seguem "//". Para declarar uma imagem f seja:

$f : \mathbb{E} \longrightarrow K$ *ou* $f \in K^{\mathbb{E}}$.

Para definir uma transformação será utilizada a notação apresentada no Algoritmo 1. Onde s define um símbolo para representar o algoritmo; D_i representa o domínio da transformação, com entrada $e_i \in D_i,\ i = 1, \ldots, n$; I_i representa o retorno, com $s_i \in I_i,\ i = 1, \ldots, m$, onde m e n são inteiros positivos; e *expressões* são as expressões matemáticas implementadas.

Algoritmo 1 *Especificação de transformação*

$s : D_1 \times \cdots \times D_n \longrightarrow I_1 \times \cdots \times I_m$
$\quad s(e_1, \ldots, e_n) = (s_1, \ldots, s_m)$

expressões;

Por exemplo, o Algoritmo 2 apresenta a soma de duas imagens com varreduras explícitas, isto é, pixel-por-pixel. Esta operação é a soma usual sem a preocupação com o tratamento de saturação.

Algoritmo 2 Soma duas imagens

$\_ + \_ : K^{\mathbb{E}} \times K^{\mathbb{E}} \longrightarrow \mathbb{Z}^{\mathbb{E}}$
$\qquad +(f_1, f_2) = f_1 + f_2$

$\forall x \in \mathbb{E}$
$\quad (f_1 + f_2)(x) = f_1(x) + f_2(x)$;

Outra forma de fazer um algoritmo para somar duas imagens com saturação em $K = [0, 255]$ é definida no Algoritmo 3, onde $\wedge$ calcula o mínimo entre $f1(x) + f2(x)$ e 255.

As notações deste documento para representar as transformações de morfologia matemática são as usadas na literatura e estão descritas na tabela de símbolos, no início deste texto. Nesta tabela também estão definidos símbolos próprios para representar algoritmos como a *erosão por propagação*, ou ε^p. Além disso, quando uma função estruturante b é aplicada em uma imagem f é comum utilizar $\varepsilon_b^p(f)$. Para deixar clara a seqüência de argumentos deste

Algoritmo 3 Soma duas imagens com saturação

$\_\bar{+}\_ : K^{\mathbb{E}} \times K^{\mathbb{E}} \longrightarrow \mathbb{Z}^{\mathbb{E}}$
$\quad \bar{+}(f_1, f_2) = f_1 \bar{+} f_2$

$\forall x \in \mathbb{E}$
$\quad (f_1 \bar{+} f_2)(x) = (f_1(x) + f_2(x)) \wedge 255;$

operador também utilizamos $\varepsilon^p(f, b)$. Por outro lado, quando um algoritmo com uma grande quantidade de argumentos junto ao símbolo é especificado, usamos simplesmente uma letra para representar este operador. Por exemplo, em vez de usar $f_1 \bar{+} f_2$ no Algoritmo 3 pode ser usado simplesmente g.

Referências Bibliográficas

[BB92] J. Barrera e G.J.F. Banon. Expressiveness of the morphological language. No *Image Algebra and Morphological Image Processing III*, páginas 264–274, San Diego, California, 1992. SPIE.

[BB93] G.J.F. Banon e J. Barrera. Decomposition of mappings between complete lattices by mathematical morphology, Part I: general lattices. *Signal Processing*, 30:299–327, 1993.

[BB94] G.J.F. Banon e J. Barrera. *Bases da morfologia matemática para analise de imagens binárias*. IX Escola de Computação, Recife, Brasil, 1994.

[BBL94] J. Barrera, G.F. Banon e R.A. Lotufo. A mathematical morphology toolbox for the KHOROS system. No *Image Algebra and Morphological Image Processing V*, páginas 241–252, Bellingham, Julho 1994. SPIE.

[Bir67] G. Birkhoff. *Lattice Theory*. American Mathematical Society, Providence, Rhode Island, 1967.

[Hei91] H.J.A.M. Heijmans. Theoretical aspects of gray-level morphology. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):568–581, Junho 1991.

[Hei94] H.J.A.M. Heijmans. *Morphological Image Operators*. Academic Press, Boston, 1994.

[Pre02] R.S. Pressman. *Engenharia de Software*. McGraw-Hill, Rio de Janeiro, 5ed. edition, 2002.

[Ser88] J. Serra, editor. *Image Analysis and Mathematical Morphology - Volume II: Theoretical Advances*. Academic Press, London, 1988.

[Vin92] L. Vincent. Morphological algorithms. No E. R. Dougherty, editor, *Mathematical Morphology in Image Processing*, capítulo 8, páginas 255–288. Marcel Dekker, New York, 1992.

[WB88] X. Wang e G. Bertrand. An algorithm for a generalized distance transformation based on Minkowski operations. *9th ICPR*, páginas 1163–1167, 1988.

[Zam97] F.A. Zampirolli. Operadores morfológicos baseados em grafos de vizinhanças – uma extensão da MMach toolbox. Dissertação de Mestrado, Unversidade de São Paulo, São Paulo, Brasil, Abril 1997.