Processamento de Imagens

Prof. MSc. Daniel Menin Tortelli

e-mail: danielmenintortelli@gmail.com
Skype: daniel.menin.tortelli

Site: http://sites.google.com/site/danielmenintortelli/home

Propriedades de uma imagem digital

Propriedades de uma imagem digital

Convenções:

- 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**.

Medições de Distância

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 .

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.

Conectividade

- Seja ***V*** o conjunto de valores de tons de cinza utilizados para se definir a conectividade.

- Conhecendo o conceito de vizinhança e dado o conjunto ***V***, podemos definir os seguintes critérios de conectividade:

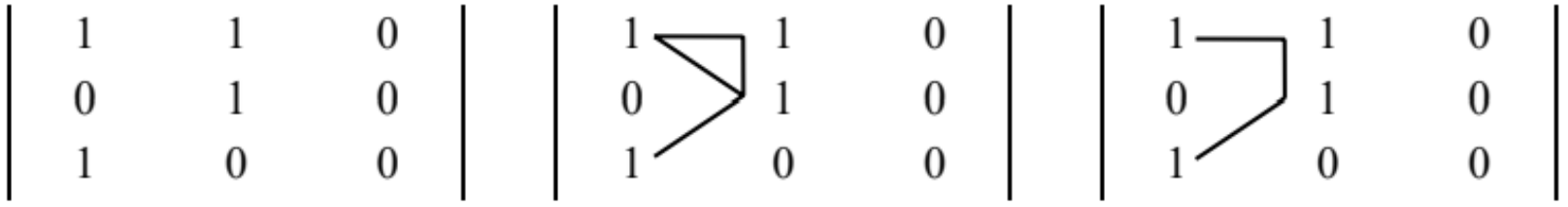
1. "4-conectividade": dois pixels p e q com valores de tom de cinza contidos em V, são "4-conectados" se $q \in N_4(p)$.
2. "8-conectividade": dois pixels p e q com valores de tom de cinza contidos em V, são "8-conectados" se $q \in N_8(p)$.
3. "m-conectividade (conectividade mista)": dois pixels p e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

   (i) $q \in N_4(p)$ ou

   (ii) $q \in N_D(p)$ e $N_4(p) \cap N_4(q) = \varnothing$.

Conectividade

- Por exemplo, seja o trecho de imagem da figura (a).
- Para *V = {1}* os caminhos entre 8 vizinhos do *pixel* do centro são indicados por linhas contínuas na figura (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 (c).



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

Caminho

- Um caminho (*path*) de um *pixel* ***p*** de coordenadas (*x,y*) a um *pixel* ***q*** de coordenadas *(s,t)* é uma sequência de *pixels* distintos de coordenadas: *(x0, y0), (x1 , y1), ... , (xn , yn),* onde:

(x0 , y0) = (x,y) *// Posição inicial é a posição de p(x,y)*

(xn , yn) = (s,t) *// Posição final é a posição de q(s,t)*

(xi , yi) é adjacente a *(xi-1, yi-1)*

1 <= i <= ***n***

n é denominado o comprimento do caminho.

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:

$$\text{(i) } D(p,q) \geq 0 \text{ } (D(p,q) = 0 \text{ se e somente se } p = q)$$

$$\text{(ii) } D(p,q) = D(q,p)$$

$$\text{(iii) } D(p,z) \leq D(p,q) + D(q,z)$$

- O conceito de distância pode estar relacionado ao conceito de conectividade.

- A distância D_m expressa a distância entre dois pontos m-conectados.

Distância Euclidiana

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

- 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*)

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

- 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*)

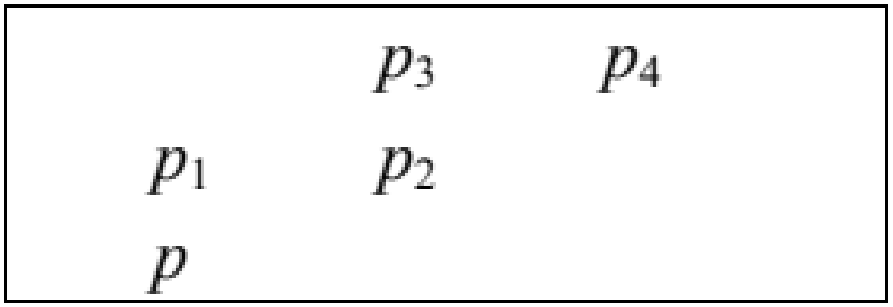
$$D_8(p,q) = max\left(|x-s|, |y-t|\right)$$

- 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).*

Exemplo de Distância

Seja o trecho de imagem binária a seguir:

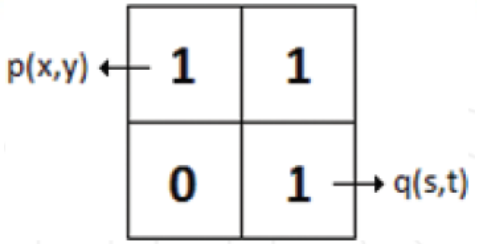


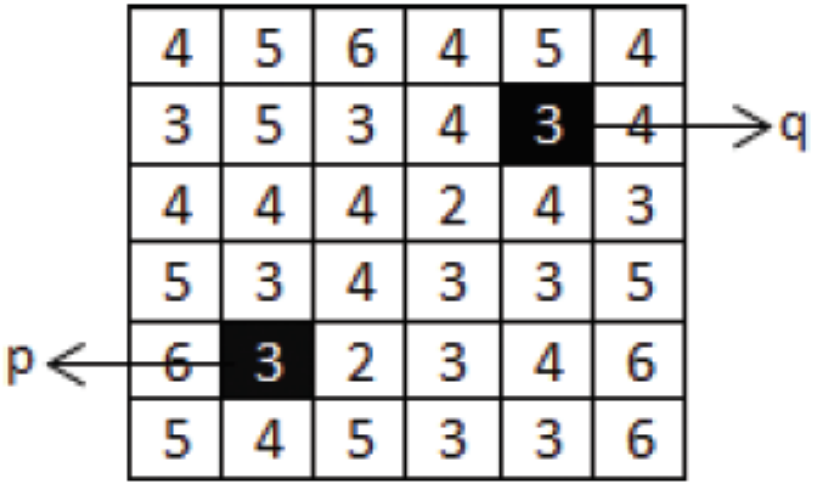
Supondo que $V = \{1\}$, $p = p_2 = p_4 = 1$ e que $\boldsymbol{p_1}$ e $\boldsymbol{p3}$ podem apresentar valores **0** ou **1**, calcular a distância $\mathbf{D_m}$ entre **p** e $\mathbf{p_4}$ para as seguintes situações:

a) Se $p_1 = p_3 = 0$.
b) Se p_1 ou p_3 valem 1.
c) Se p_1 e p_3 valem 1.

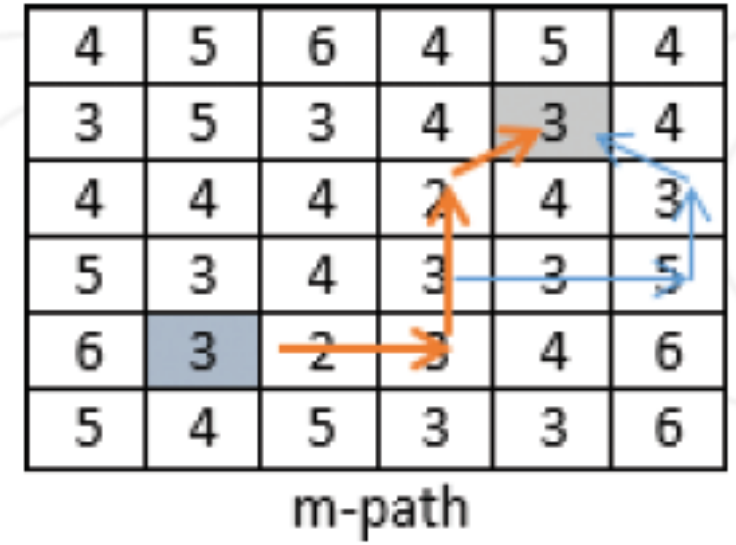
Exemplo de Distância

Distância entre os pixels **p** e **q**:





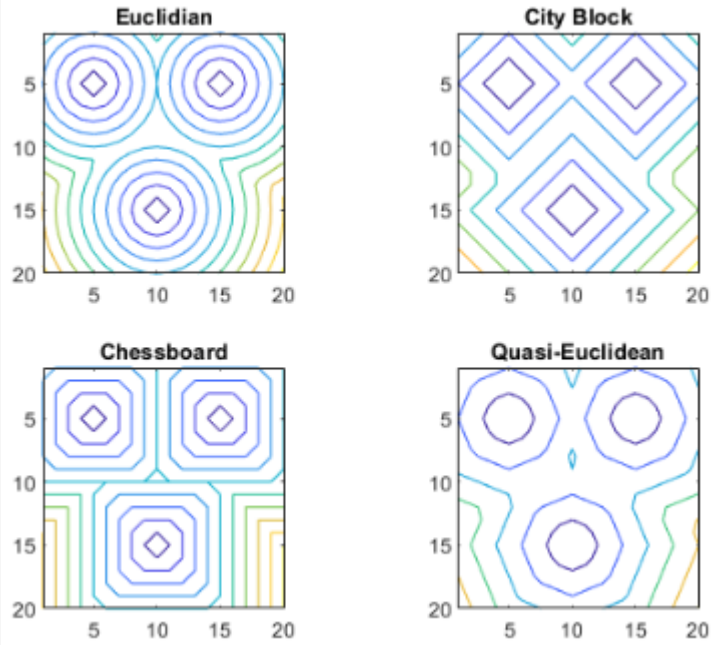
Qual o menor caminho entre **p** e **q**?



D_m é **5** porque o caminho laranja é mais curto que o caminho azul.

Exemplo de Distância

```
bw = zeros(20,20);
bw(5,5) = 1;
bw(5,15) = 1;
bw(15,10) = 1;

D1 = bwdist(bw, "euclidean");
D2 = bwdist(bw, "cityblock");
D3 = bwdist(bw, "chessboard");
D4 = bwdist(bw, "quasi-euclidean");

figure
subplot(2,2,1);
imagesc(mat2gray(D1)); imcontour(D1); title('Euclidian');
subplot(2,2,2);
imagesc(mat2gray(D2)); imcontour(D2); title('City Block');
subplot(2,2,3);
imagesc(mat2gray(D3)); imcontour(D3); title('Chessboard');
subplot(2,2,4);
imagesc(mat2gray(D4)); imcontour(D4); title('Quasi-Euclidean');
```



Resolução Espacial

Resolução Espacial

- A resolução espacial pode ser definida como o número de pixels por polegada.

- Diferentes resoluções espaciais da mesma imagem são mostradas na figura abaixo:

- A resolução espacial possui diferentes métodos de medição para dispositivos diferentes.

Resolução Espacial: DPI

Dot Per Inch (DPI):

- DPI é geralmente utilizado em monitores.
- As vezes é chamado de PPI (*Pixels Per Inch*).
- Entretanto, as duas expressões possuem uma diferença:
  - DPI é também usado para medir a resolução espacial de impressoras. Isso significa que DPI define quantos pontos de tinta por polegada na imagem impressa.

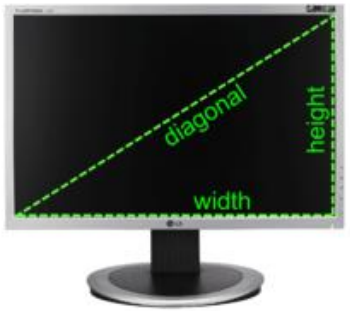
Resolução Espacial: DPI

Pixels Per Inch (PPI):

- PPI geralmente é usado em tablets, smartphones, etc.

- Se **a** e **b** são representam **altura** e **largura** na resolução de uma imagem, é possível calcular o valor dos PPI de um dispositivo através da equação:

$$PPI = \frac{\sqrt{a^2 + b^2}}{Diagonal\ Size\ of\ Devices}$$



Por exemplo: 1080x1920 *pixels*, 5.5 *inch* iPhone 6S Plus PPI:

$$PPI = \frac{\sqrt{1080^2 + 1920^2}}{5.5} \cong 401 \ (it\ is\ shown\ in\ apple\ web\ site)$$

Resolução Espacial: DPI

Lines Per Inch (LPI):

- LPI são linhas de pontos por polegada de impressoras.

- A impressora possui diferentes valores de LPI, conforme mostrado na tabela abaixo:

Printer	LPI value
Screen printing	45–65 LPI
Laser printing (300 dpi)	65 LPI
Laser printing (600 dpi)	85–105 LPI

Detecção de Bordas

Detecção de Bordas

- A detecção de bordas é basicamente um método de segmentar uma imagem em regiões, com base em descontinuidades, ou seja, permite que o usuário observe as características de uma imagem onde há mudança mais ou menos abrupta no nível de cinza ou na textura, indicando o fim de uma região na imagem e o início de outra.

- A detecção de bordas faz uso de operadores diferenciais para detectar alterações nos gradientes de níveis de cinza ou de cores em uma imagem.

- A diferenciação é uma operação linear e uma aproximação discreta de um filtro diferencial que pode, então, ser implementada pelo método de núcleo (máscara de convolução).

Detecção de Bordas

- Uma condição muito importante que devemos impor sobre o núcleo de um filtro desse tipo é que a resposta seja zero em regiões completamente suaves.

- Essa condição pode ser forçada assegurando que a soma dos pesos na máscara do núcleo seja zero.

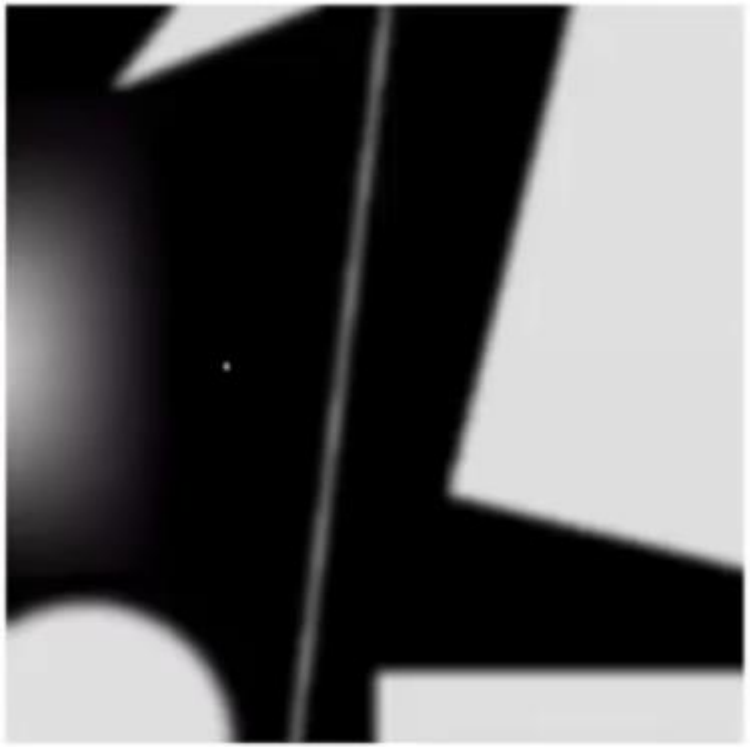
- A detecção de bordas (que é a principal aplicação de filtros diferenciais) recebe auxílio de um estágio inicial suavizador (na maioria das vezes, gaussiano) para supressão de ruído.

- A detecção de bordas é dividida em duas categorias principais:
  - **Detecção de bordas de primeira ordem;**
  - **Detecção de bordas de segunda ordem**.

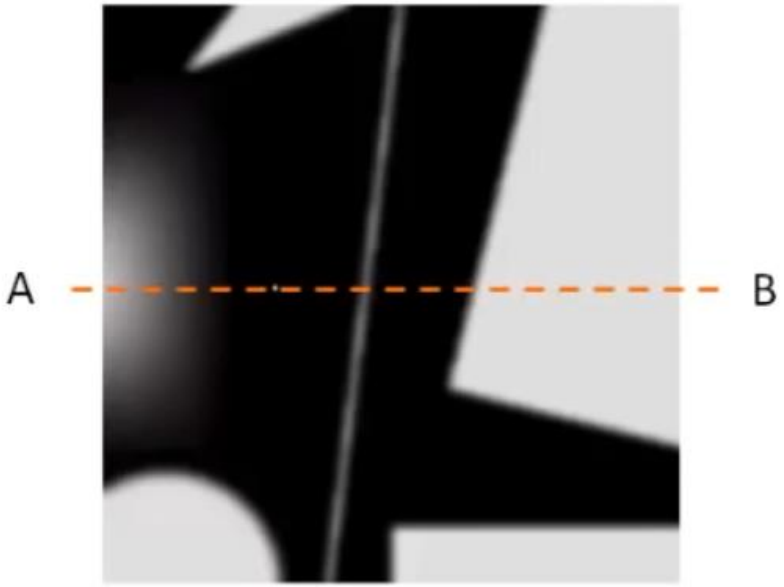
Detecção de Bordas

- A detecção de bordas utiliza o conceito de **filtros passa-alta**, que deixam passar as altas frequências da imagem (bordas e detalhes na imagem onde ouve uma transição abrupta nos níveis de intensidade do pixel), enquanto suprimem as baixas frequências.

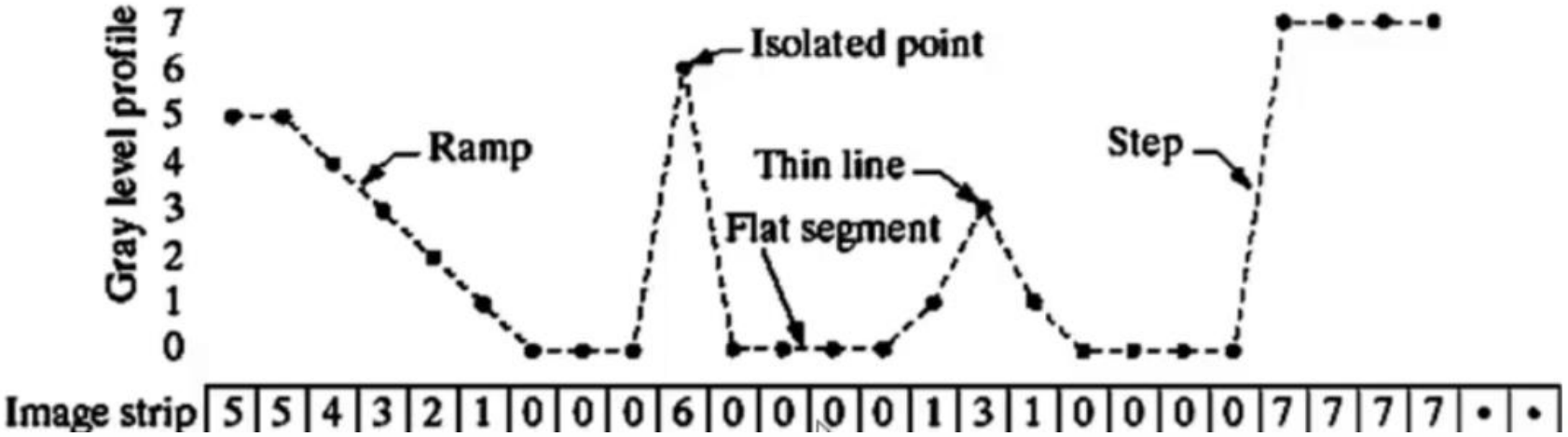
- A figura mostra uma imagem com vários pontos onde ocorrem transições abruptas de intensidade:



Detecção de Bordas



- Exemplo: Se a imagem for composta por intensidades que variam entre 0-7, e analisarmos uma linha da matriz da imagem, podemos definir a variação das intensidades dos pixels como:



Cálculo da Primeira Derivada

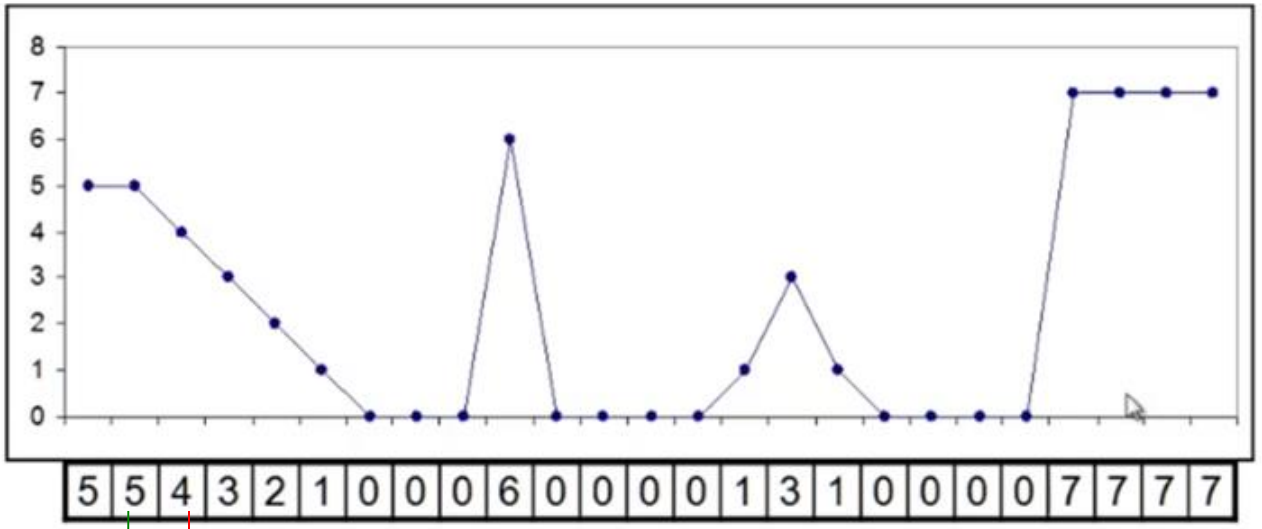
- Uma vez que as intensidades são conhecidas, é possível calcular a primeira derivada da função ***f*** com relação a ***x*** com a fórmula:

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

- Isso mostra a diferença entre os valores subsequentes e calcula a taxa de mudança da função em relação a ***x***.

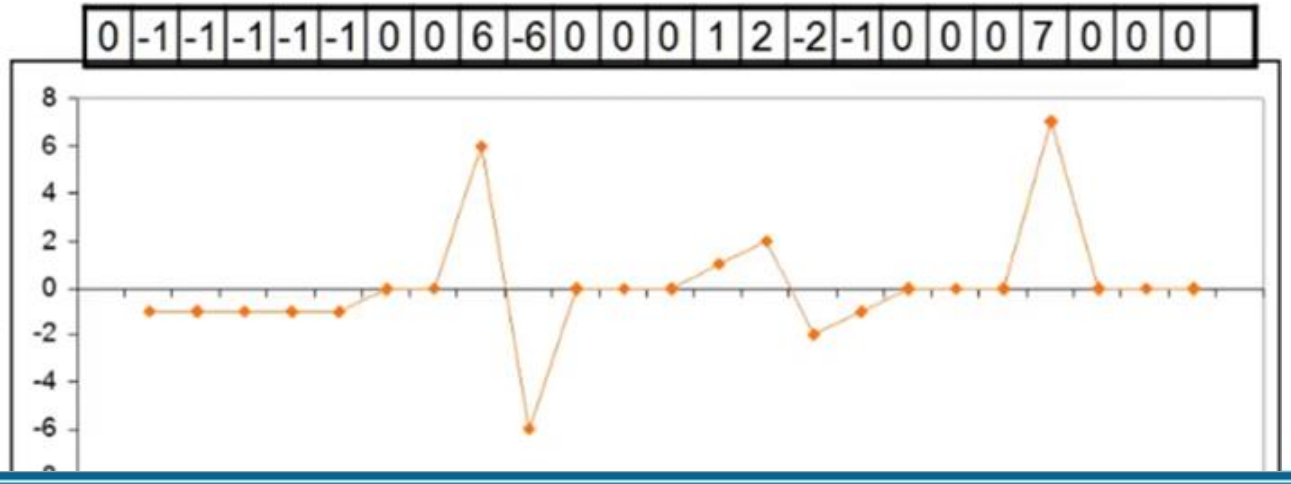
- A primeira derivada é sensível a mudanças de intensidades nos pixels da imagem.

Cálculo da Primeira Derivada



4 - 5 = -1

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$



Cálculo da Primeira Derivada

- A **magnitude** do vetor é dado por:

$$\nabla f = mag(\nabla \mathrm{f})$$
$$= \left[G_x^2 + G_y^2\right]^{1/2}$$
$$= \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{1/2}$$

Prewitt Kernels

-1	0	1
-1	0	1
-1	0	1

Gx (X derivative)

-1	-1	-1
0	0	0
1	1	1

Gy (Y derivative)

$V = (Gx^2 + Gy^2)^{0.5}$

- Simplificando...

$$\nabla f \approx |G_x| + |G_y|$$

Detecção de bordas de primeira ordem

- Três núcleos de filtros detectores de bordas de primeira ordem (Primeira Derivada) mais comuns são ***Roberts***, ***Prewitt*** e ***Sobel***.

- Os três são implementados como a combinação de dois núcleos: um para a derivada em relação a **x** e outro para a derivada em relação a **y**, derivadas das equações mostradas anteriormente.

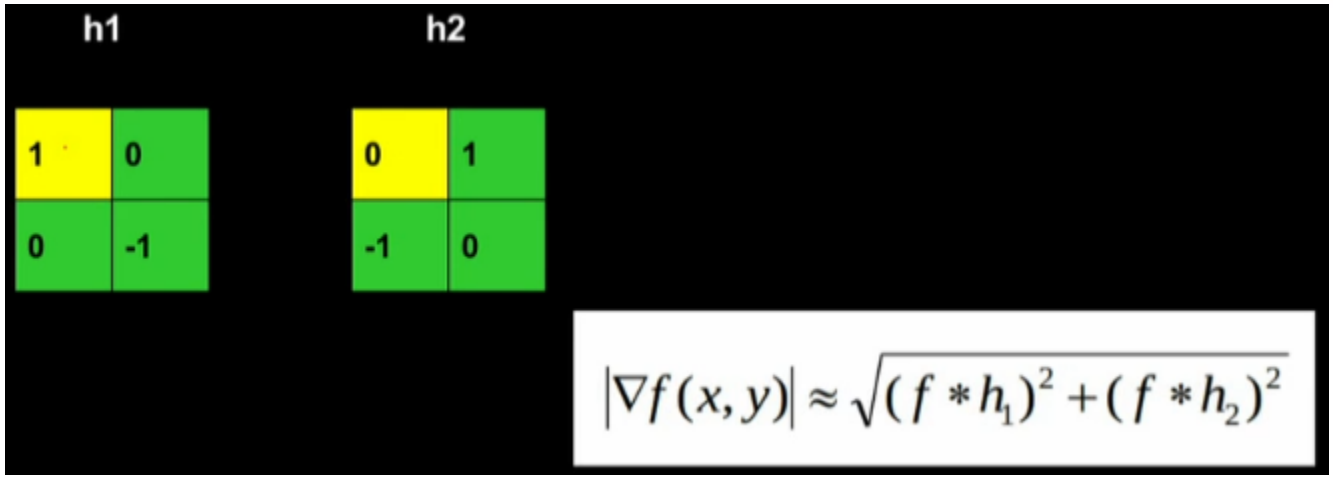
- A principal diferença entre os operadores de *Sobel* e de *Prewitt* é que o núcleo de *Sobel* implementa diferenciação em uma direção e aplica média (aproximadamente) gaussiana na outra (núcleo gaussianos).

Detecção de bordas de primeira ordem

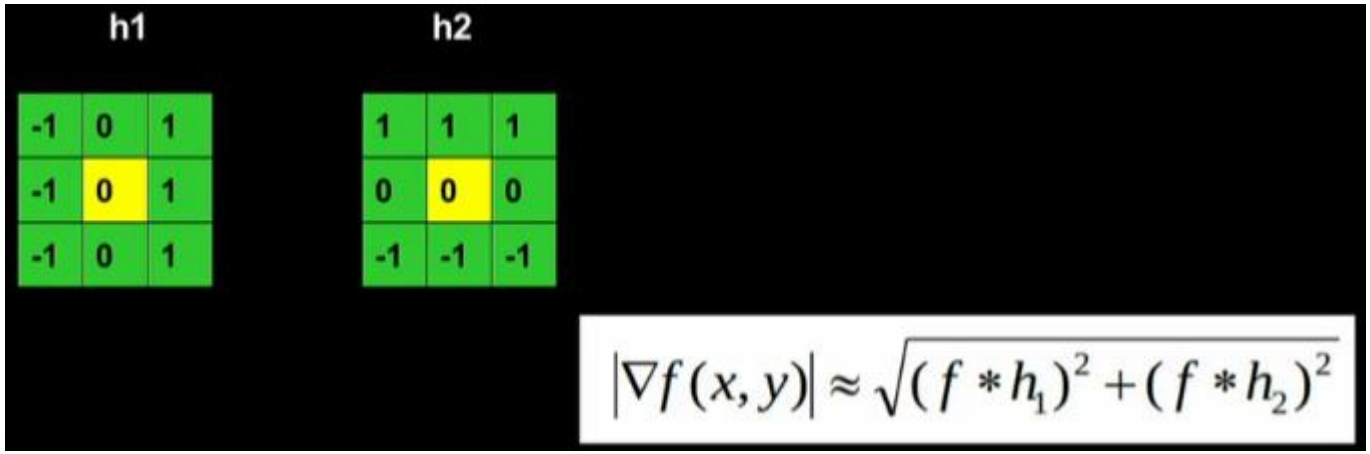
Roberts			Prewitt			Sobel		
0	–1	Derivada em relação a X	1	0	–1	1	0	–1
1	0		1	0	–1	2	0	–2
			1	0	–1	1	0	–1
–1	0	Derivada em relação a Y	1	1	1	1	2	1
0	1		0	0	0	0	0	0
			–1	–1	–1	–1	–2	–1

Filtros detectores de bordas de primeira ordem.

Detector de Bordas Gradiente Cruzado de *Roberts*



Detector de Bordas *Prewitt*



Detecção de bordas de primeira ordem

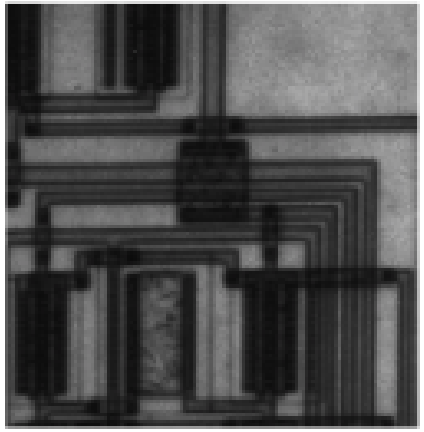
- ***Para encontrar a borda de objetos em uma imagem, usa-se os dois filtros independentemente e, após, soma-se os resultados.***

- Os algoritmos mais comuns para detecção de bordas são: *Sobel*, *Canny*, *Prewitt*, *Roberts*, etc.

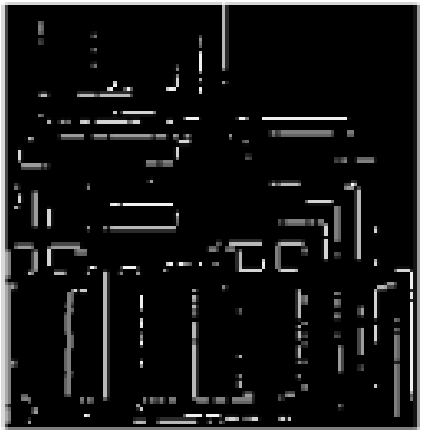
```
I = imread('circuit.tif');
Prewitt = edge(I, 'Prewitt');
Canny = edge(I, 'Canny');
Sobel = edge(I, 'Sobel');

subplot(2,2,1); imshow(I); title("Imagem Original");
subplot(2,2,2); imshow(Prewitt); title("Prewitt");
subplot(2,2,3); imshow(Canny); title("Canny");
subplot(2,2,4); imshow(Sobel); title("Sobel");
```

Detecção de bordas de primeira ordem

Imagem Original



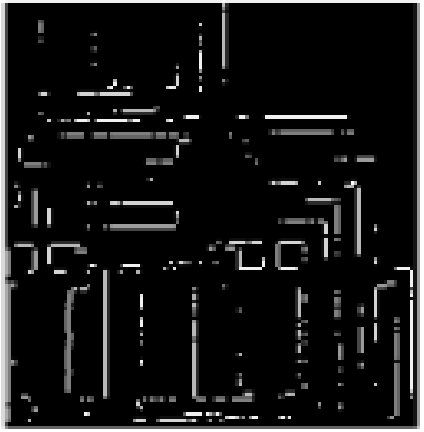
Prewitt



```
>> f = fspecial('prewitt')

f =

     1     1     1
     0     0     0
    -1    -1    -1

>> f = fspecial('prewitt')'

f =

     1     0    -1
     1     0    -1
     1     0    -1
```

Canny

Sobel



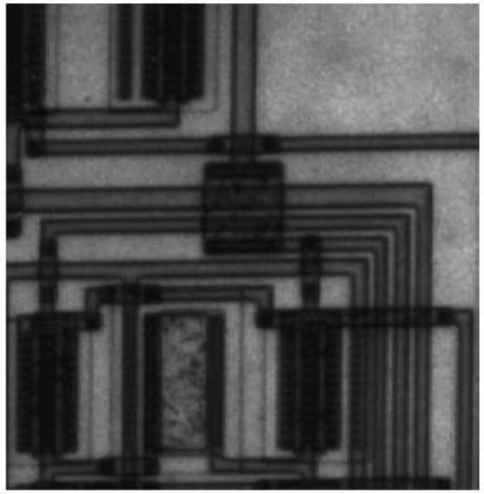
```
>> f = fspecial('sobel')

f =

     1     2     1
     0     0     0
    -1    -2    -1

>> f = fspecial('sobel')'

f =

     1     0    -1
     2     0    -2
     1     0    -1
```

Detecção de bordas de primeira ordem (Sobel)

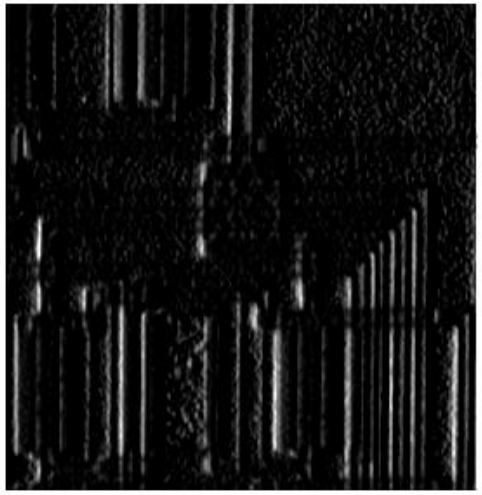
```
I = imread('circuit.tif'); % Lê a imagem

% Filtros para suavização
kMedia = ones(3,3)/9;
kGaussiano = fspecial('gaussian', [5 5], 2);

% Filtros de Sobel
k1 = [1 0 -1;
      2 0 -2;
      1 0 -1];

k2 = [1  2  1;
      0  0  0;
     -1 -2 -1];

 SobelV = filter2(k1, I); % Aplica filtro com relação a Y
 SobelH = filter2(k2, I); % Aplica filtro com relação a X
 SobelEdge = sqrt(SobelV.^2 + SobelH.^2); % Combina o resultado dos dois filtros

 SobelEdgeBW = imbinarize(SobelEdge/255); % Converte a imagem para preto/branco
 SobelEdgeBW_Filtered = imfilter(SobelEdgeBW, kMedia); % Remove ruído da imagem

 % Exibe resultados
 figure, imshow(I); title('Imagem Original');
 figure, imshow(SobelV/255); title('Sobel Vertical');
 figure, imshow(SobelH/255); title('Sobel horizontal');
 figure, imshow(SobelEdge/255); title('Sobel Edge');
 figure, imshow(SobelEdgeBW); title('Sobel Edge BW');
```

Detecção de bordas de primeira ordem (Sobel)

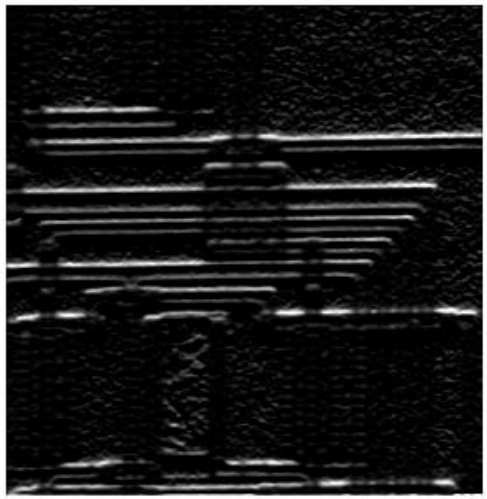
Imagem Original



Sobel Vertical



Sobel horizontal



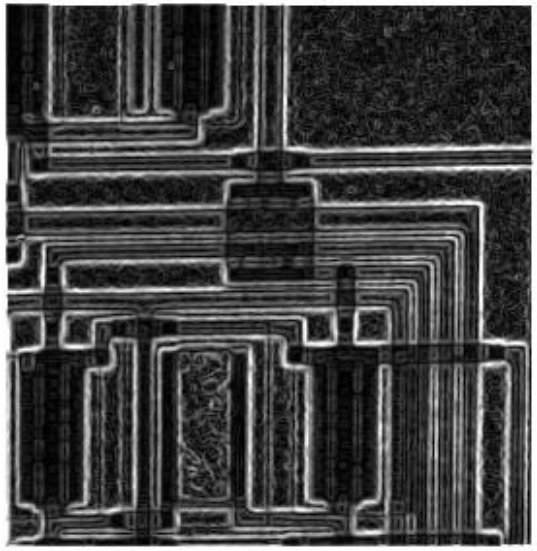
Sobel

1	0	–1
2	0	–2
1	0	–1

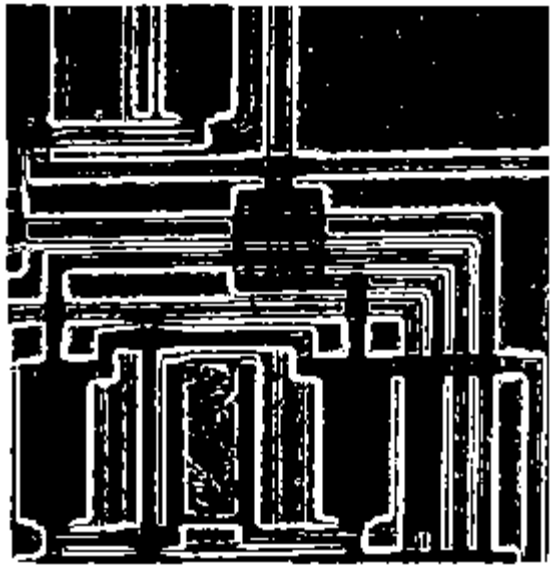
$V = (Gx^2 + Gy^2)^{0.5}$

1	2	1
0	0	0
–1	–2	–1

Sobel Edge



Sobel Edge BW



Detecção de bordas de segunda ordem

- Em geral, filtros de bordas de primeira ordem não são de uso muito comum no realce de imagens.

- *Seu principal uso reside no processo de detecção de bordas, como um estágio no procedimento de segmentação de imagem.*

- Uma abordagem muito mais comum para o realce de imagens é o uso de um operador diferencial de segunda ordem: o **laplaciano**.

Detecção de bordas de segunda ordem

- Um operador diferencial de segunda ordem muito popular é o **laplaciano**:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

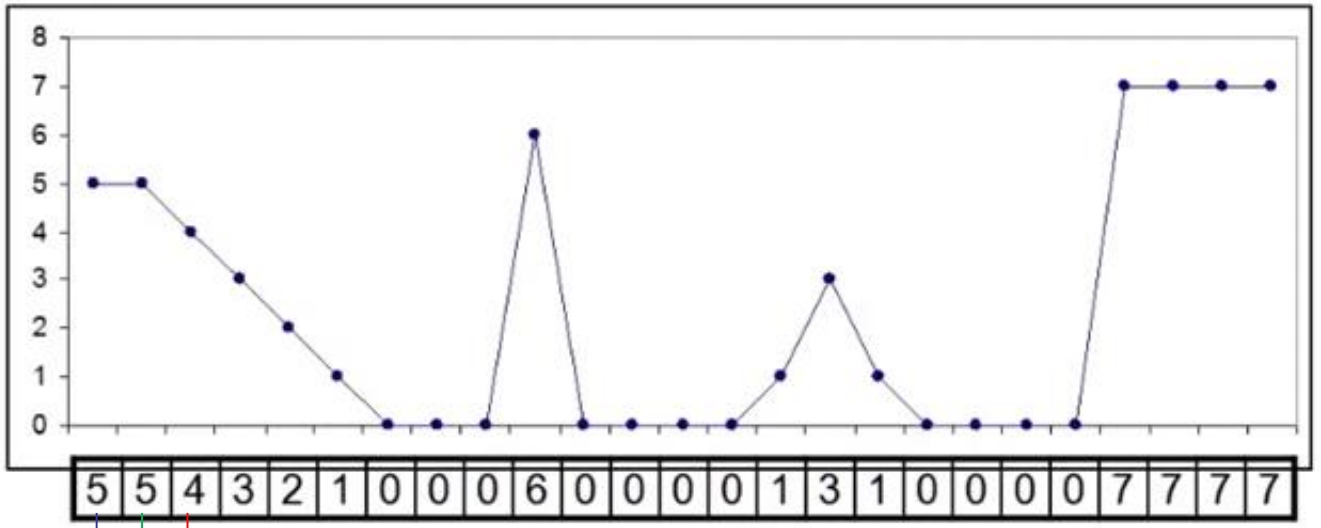
- Onde a derivada parcial de primeira ordem na direção de **x** é definida como:

$$\frac{\partial^2 f}{\partial^2 x} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

- Na direção de **y** é definida como:

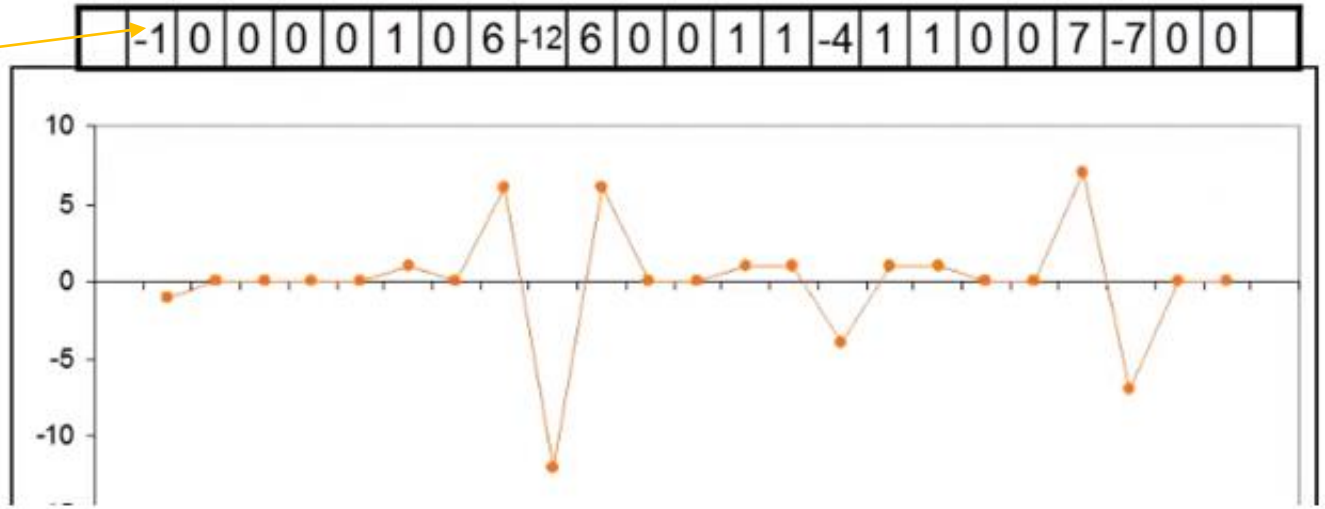
$$\frac{\partial^2 f}{\partial^2 y} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

Cálculo da Segunda Derivada



$$4 + 5 - 2 \times 5 = -1$$

$$\frac{\partial^2 f}{\partial^2 x} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$



Detecção de bordas de segunda ordem

- Na forma discreta do Laplaciano, esse operador é dado como:

$$\nabla^2 f = f(x+1,\ y) + f(x-1,\ y) + f(x,\ y+1) + f(x,\ y-1) - 4f(x,y)$$

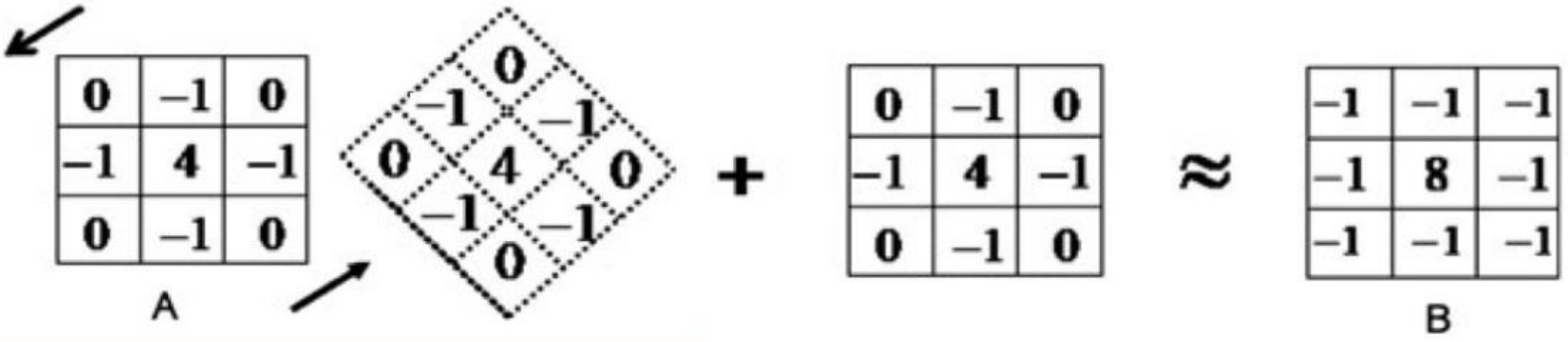
- Isso pode ser facilmente implementado em um núcleo de filtro 3 x 3, como mostrado na figura A:

0	1	0
1	−4	1
0	1	0

Detecção de bordas de segunda ordem

- Uma das possíveis limitações da aplicação da máscara na forma dada na figura A é a intensidade relativa de características da imagem posicionadas aproximadamente em direções diagonais em relação aos eixos da imagem.

- Se imaginarmos a rotação dos eixos em 45° e superpusermos o laplaciano após rotação ao original, poderemos construir um filtro que seja invariante em relação a múltiplas rotações de 45°, figura B:



Detecção de bordas de segunda ordem

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

- O Laplaciano de um função ***f*** é soma da segunda derivada parcial de ***f*** em relação a **X**, com a segunda derivada parcial de ***f*** em relação a **Y**.

- Máscaras para o filtro Laplaciano:
  - **Centro Negativo**: remove bordas exteriores.
  - **Centro positivo**: remove bordas interiores.

- O valor do pixel central da máscara está relacionado com os pixels da vizinhança utilizados nos cálculos.

a

0	1	0
1	−4	1
0	1	0

b

1	1	1
1	−8	1
1	1	1

c

0	−1	0
−1	4	−1
0	−1	0

d

−1	−1	−1
−1	8	−1
−1	−1	−1

(a) Máscara de filtragem utilizada para implementar a Equação

(b) Máscara utilizada para implementar uma extensão dessa equação que inclui os termos diagonais.

(c) e (d) Duas outras implementações do laplaciano frequentemente encontradas na prática.

Cálculo das Derivadas

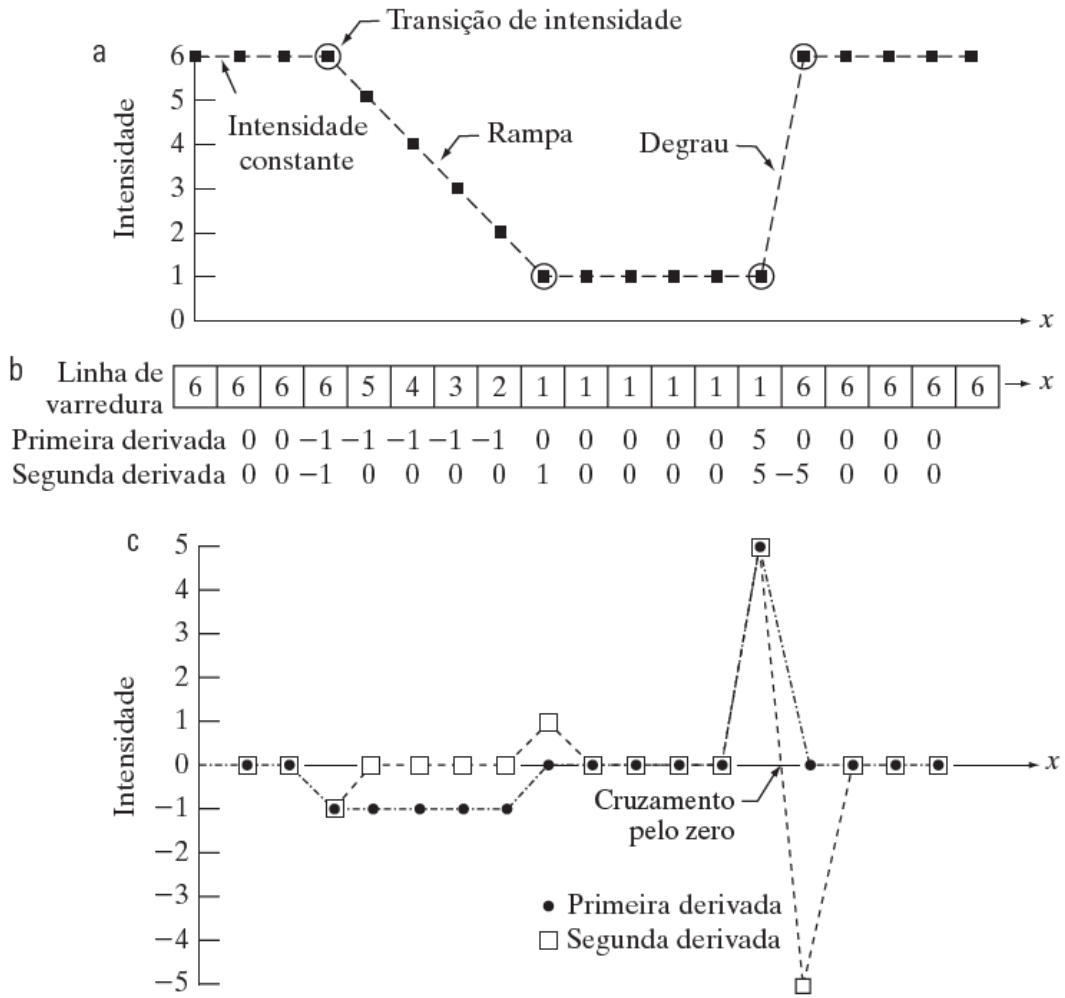
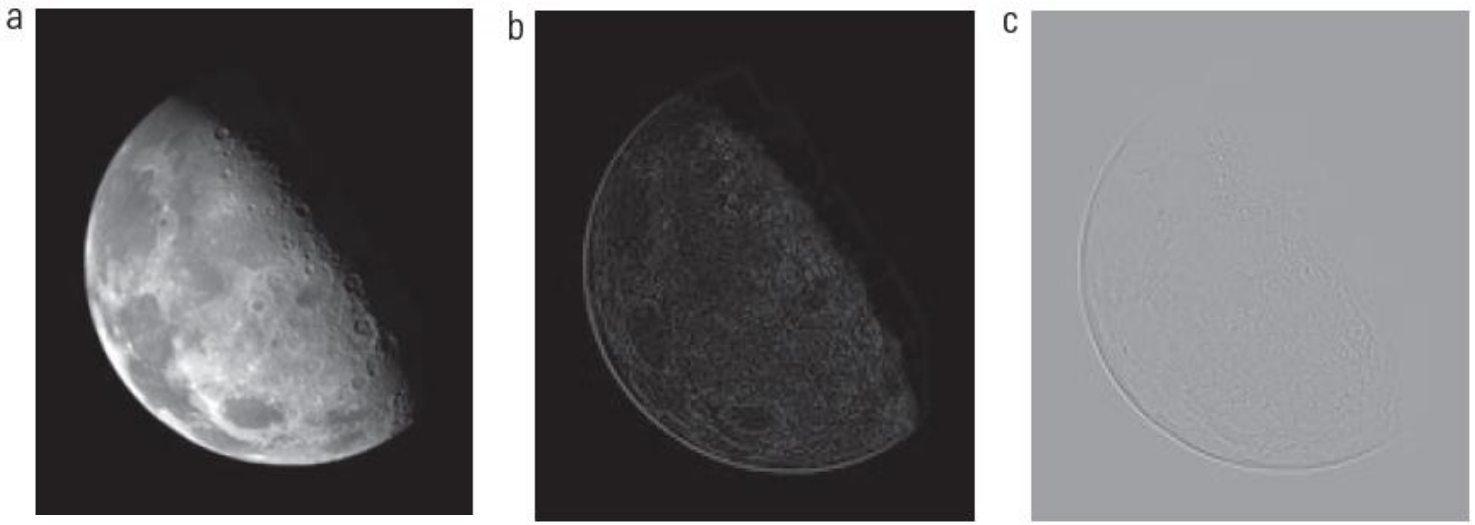
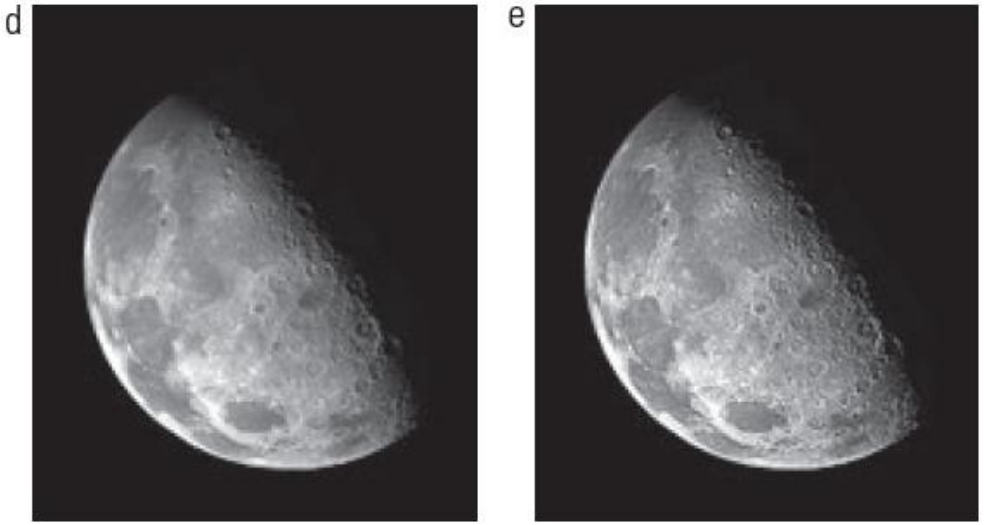


Ilustração do primeiro e do segundo derivativo de uma função digital unidimensional representando uma seção de um perfil de intensidade horizontal de uma imagem. Em (a) e (c), os pontos de dados são ligados por linhas tracejadas para facilitar a visualização.

Detecção de bordas de segunda ordem





(a) Imagem borrada do polo norte da Lua.
(b) Laplaciano sem ajuste.
(c) Laplaciano com ajuste.
(d) Imagem aguçada utilizando a máscara

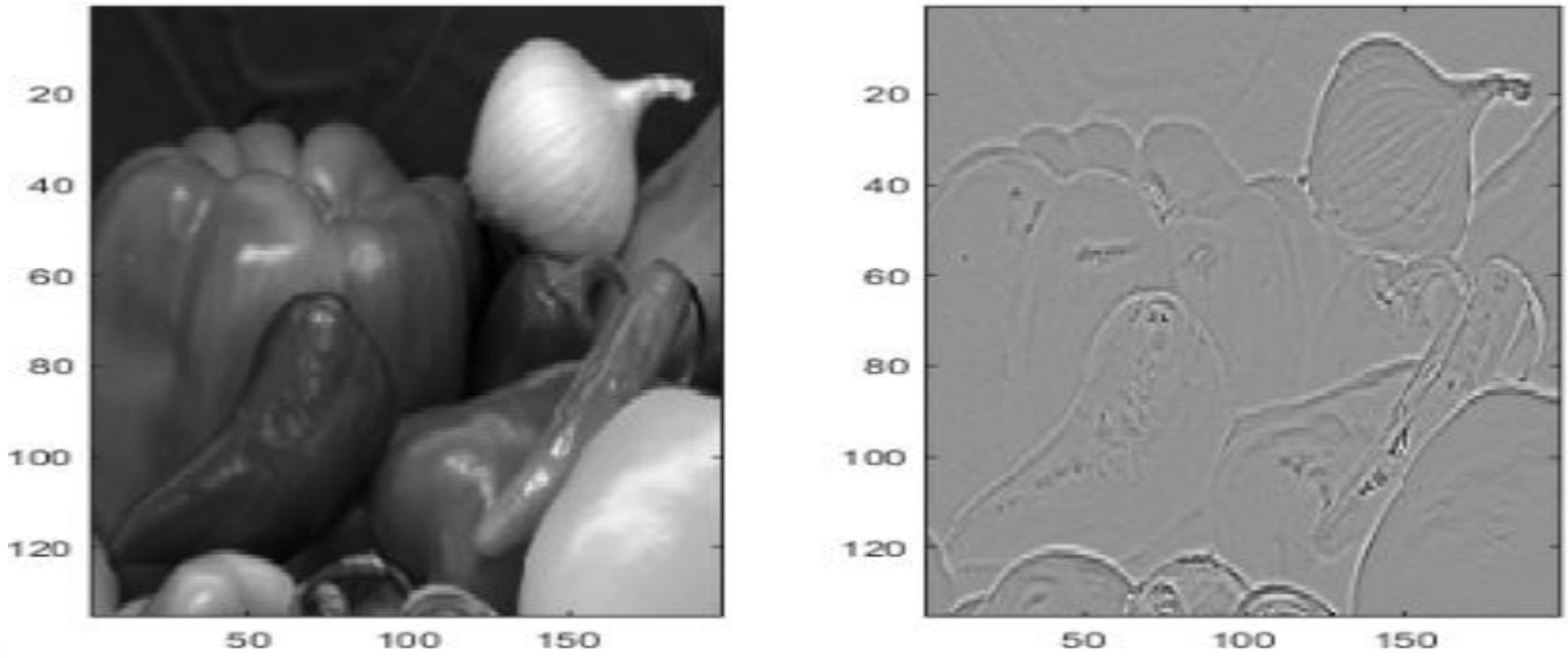
0	1	0
1	−4	1
0	1	0

(e) Resultado da utilização da máscara

1	1	1
1	−8	1
1	1	1

Detecção de bordas de segunda ordem

```
I = rgb2gray(imread('onion.png')); % Lê a imagem (em escala de cinza)
k = fspecial('laplacian'); % Cria o filtro laplaciano
IEl = imfilter(double(I), k, 'symmetric'); % Bordas com filtro laplaciano

% Exibe os resultados
subplot(1,2,1), imagesc(I);
subplot(1,2,2), imagesc(IEl);
colormap('gray');
```



Detecção de bordas de segunda ordem

- Neste exemplo, primeiro, foi construído o filtro laplaciano e, então, foi aplicado à imagem usando a função ***imfilter()***.

- Repare no uso direto da função ***rgb2gray()*** para carregar a imagem em questão (em cores) em escala de cinza.

- Ademais, foi realizada a operação do laplaciano na versão da imagem de entrada em ponto flutuante (função ***double()***);

- Como o operador laplaciano retorna valores positivos e negativos, foram usadas as funções ***imagesc()*** e ***colormap()*** para adequar a escala e exibir a imagem.

Operações Morfológicas

Operações Morfológicas

- A palavra morfologia denota um ramo da biologia que lida com a forma e a estrutura de animais e plantas.
- A morfologia matemática serve como ferramenta para extrair componentes da imagem (estrutura e forma) que são úteis para a descrição e representação.
- Além disso, a morfologia matemática pode ser aplicada para pré e pós-processamento de imagens.
- Operações morfológicas podem ser aplicadas a imagens de todos os tipos, mas o principal uso de morfologia é no processamento de **imagens binárias**.

Operações Morfológicas

- *Morfologia digital* ou *matemática* é uma modelagem destinada à descrição ou análise da forma de um objeto digital.

- As operações básicas da morfologia digital são:
  - A *erosão*, a partir da qual são removidos da imagem os pixels que não atendem a um dado padrão;

$$Erosão: E(A, B) = A \ominus B = \{x \in E \mid Bx \subseteq A\}$$

  - A *dilatação*, a partir da qual uma pequena área relacionada a um pixel é alterada para um dado padrão.

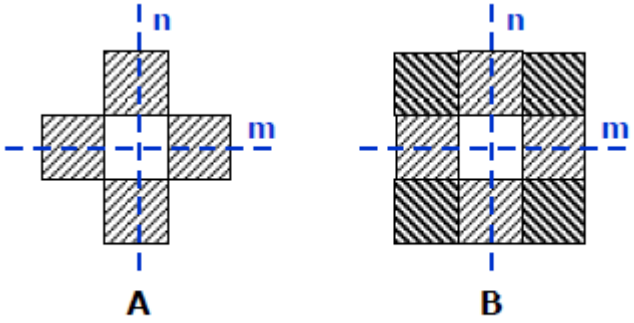
$$Dilatação: D(A, B) = A \oplus B = A \oplus B = \{x \in E \mid Bx \cap A \neq \varnothing\}$$

Operações Morfológicas

- Tanto o conjunto ***A*** quanto o conjunto ***B*** podem ser considerados como sendo imagens.

- Todavia, ***A*** costuma ser considerado com sendo a imagem sob análise e ***B*** como o *elemento estruturante*, o qual está para a morfologia como a *máscara* (*mask*, *template* ou *kernel*) está para teoria de filtragem linear.

- Os *elementos estruturantes* mais comuns são os conjuntos 4-conexões e 8-conexões, ***N4*** e ***N8****:*



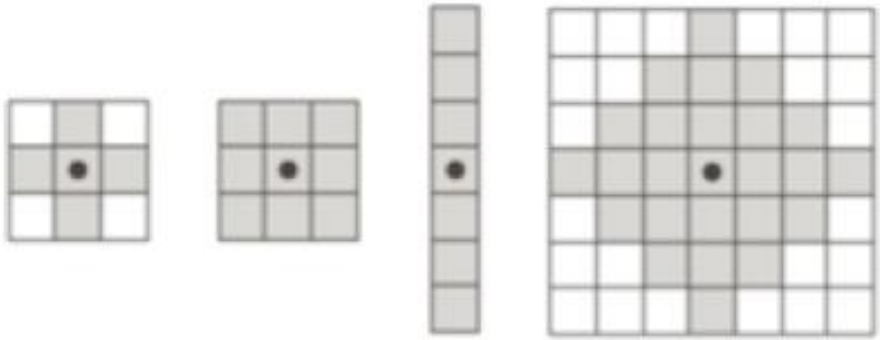
Elementos *estruturantes*: (A) padrão N_4; e (B) padrão N_8.

Elementos Estruturantes (SE)

- Uma operação morfológica binária é determinada a partir da vizinhança examinada ao redor do ponto central.

- Essa vizinhança é definida por um conjunto de pixels com um formato, chamado de **elemento estruturante**.

- Um elemento estruturante é definido pelos pixels que o constituem e uma forma que possui diferentes tamanhos (3x3, 4x4, 5x5, etc) e formatos:



Operações Morfológicas: Dilatação

- **Dilatação** é o processamento morfológico para crescimento (expansão) de um objeto em uma imagem.
- A dilatação, em geral, faz com que o objeto cresça no tamanho. Buracos menores do que o elemento estruturante são eliminados e o número de componentes pode diminuir.

$$C = A \oplus B$$

onde:

C é a nova imagem;

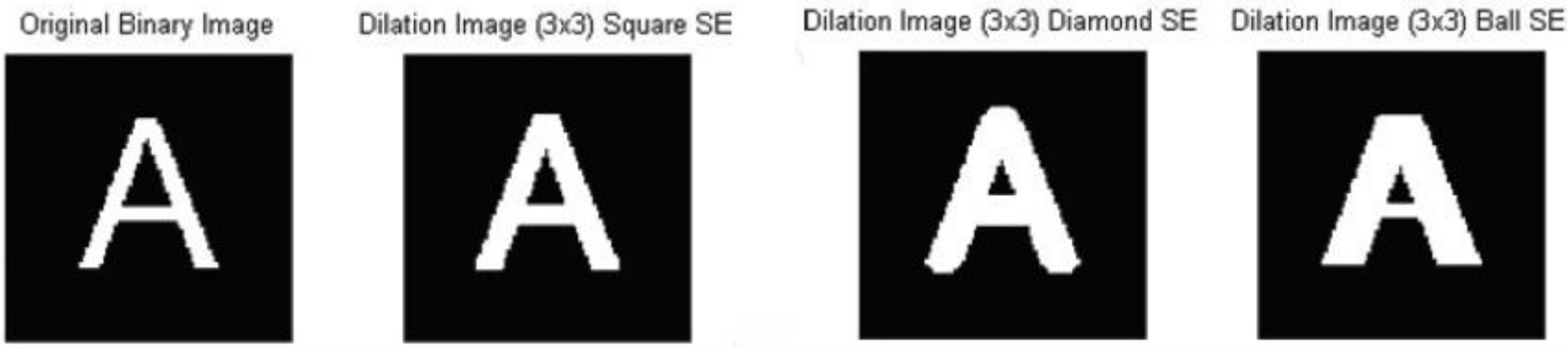
A é a imagem original;

B é o elemento estruturante.

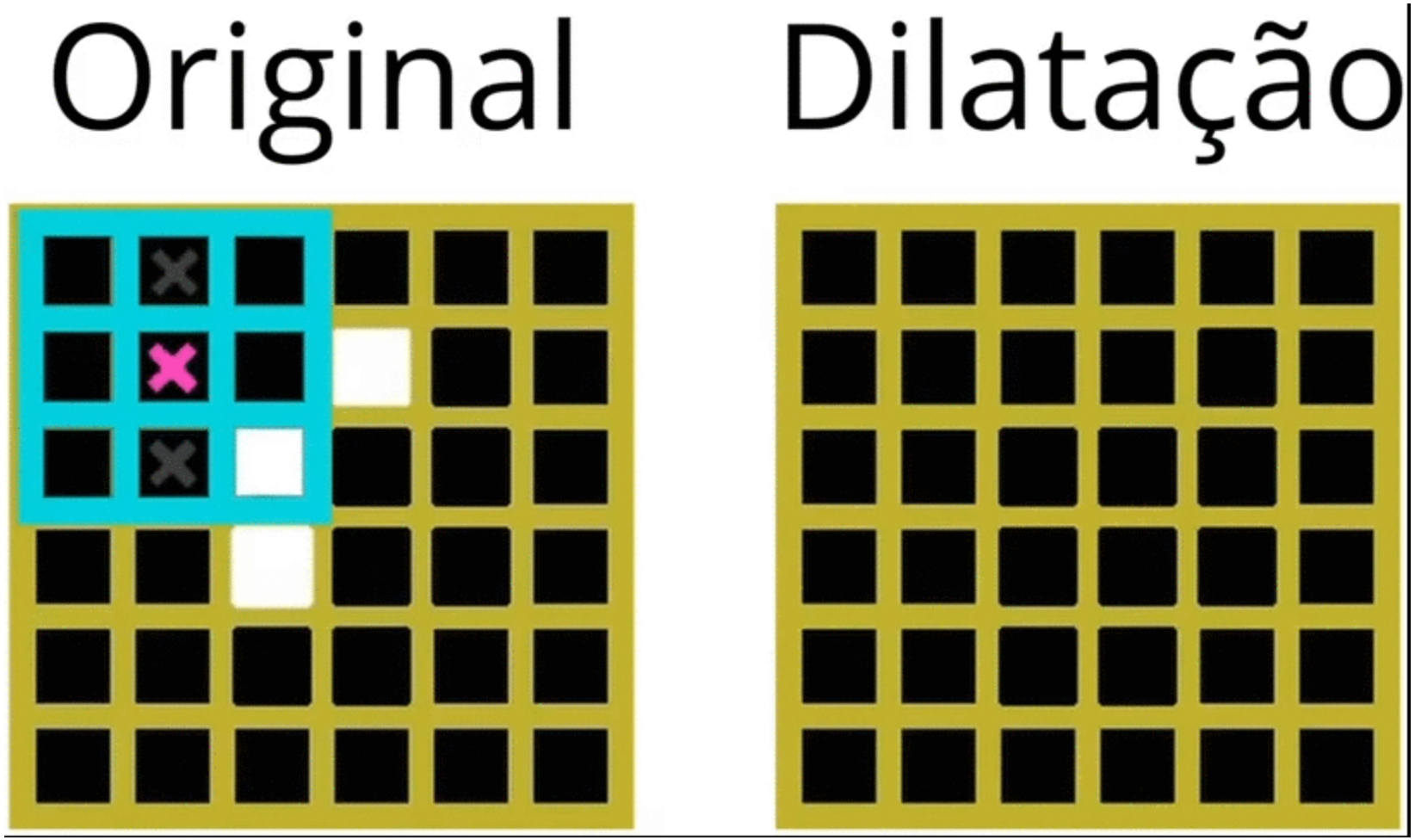
Operações Morfológicas: Dilatação

$$Dilatação: D(A, B) = A \oplus B = A \oplus B = \{x \in E \mid Bx \cap A \neq \varnothing\}$$

- Passar o elemento estruturante por todos os pixels da imagem original:
  - Se o valor do pixel sob o elemento central for diferente de 0, copie todos os valores 1 do elemento estruturante para a imagem resultante.



Operações Morfológicas: Dilatação



Operações Morfológicas: Dilatação

```
% Lê uma imagem binária
I = imread("J.bmp");

% Define um elemento estruturante do tipo quadrado 3x3
SE_square = strel('square', 3);
% Aplica a convolução da imagem original (I) com o elemento estruturante (SE)
dilationsq = imdilate(I, SE_square);

% Define um elemento estruturante do tipo diamante 3x3
SE_diamond = strel('diamond', 3);
% Aplica a convolução da imagem original (I) com o elemento estruturante (SE)
dilationdm = imdilate(I, SE_diamond);

% Define um elemento estruturante do tipo linha
SE_line = strel('line', 9, 0);
% Aplica a convolução da imagem original (I) com o elemento estruturante (SE)
dilationl = imdilate(I, SE_line);

% Exibe resultados
subplot(2,2,1); imshow(I); title("Imagem Original");
subplot(2,2,2); imshow(dilationsq); title("Dilatação com Square SE 3x3");
subplot(2,2,3); imshow(dilationdm); title("Dilatação com Diamond SE 3x3");
subplot(2,2,4); imshow(dilationl); title("Dilatação com Line SE");
```

Operações Morfológicas: Dilatação

Imagem Original

Dilatação com Square SE 3x3

Dilatação com Diamond SE 3x3

Dilatação com Line SE

Operações Morfológicas: Erosão

- **Erosão** é o processamento morfológico para diminuição (contração) de um objeto em uma imagem.
- A erosão reduz as dimensões do objeto. Objetos menores do que o elemento estruturante são eliminados e o número de componentes pode aumentar.

$$C = A \ominus B$$

onde:

C é a nova imagem;

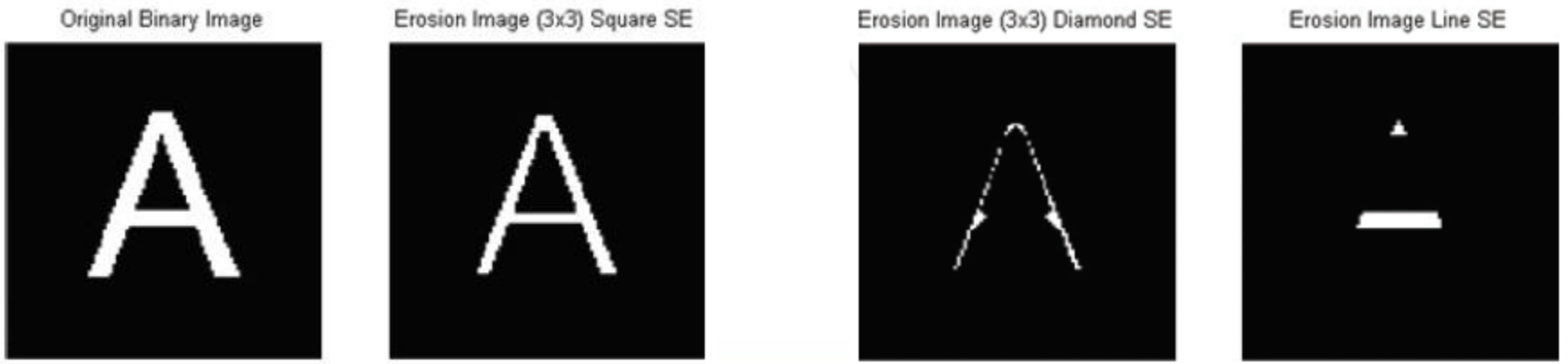
A é a imagem original;

B é o elemento estruturante.

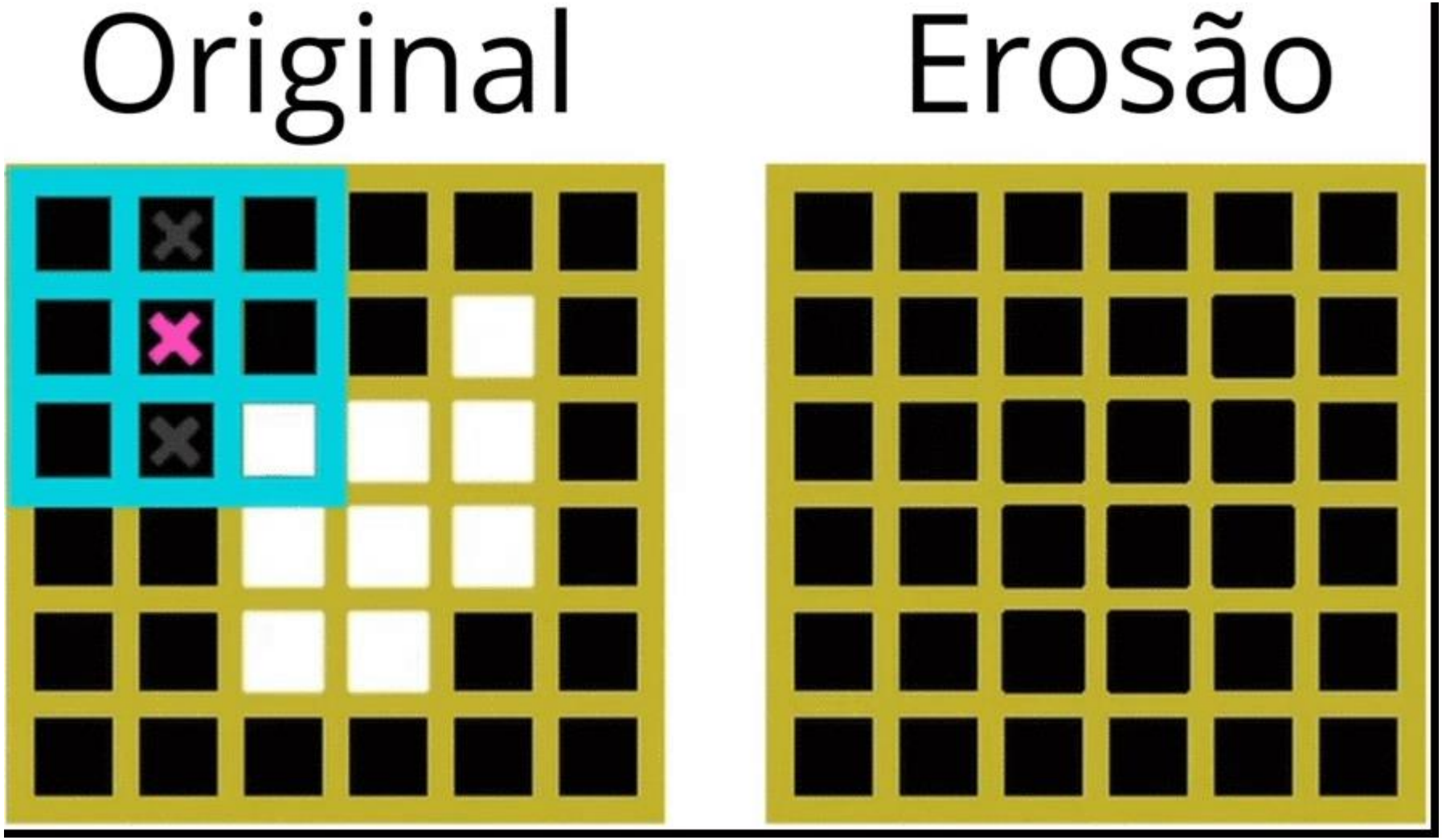
Operações Morfológicas: Erosão

$$Erosão: E(A, B) = A \ominus B = \{x \in E \mid Bx \subseteq A\}$$

- Passar o elemento estruturante por todos os pixels da imagem original:
  - Se nenhum valor dos pixels da imagem sob os valores não nulos do elemento estruturante for 0, ponha 1 no resultado.



Operações Morfológicas: Erosão



Operações Morfológicas: Erosão

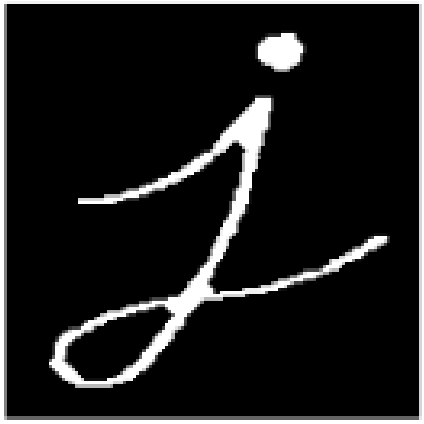
```
% Lê uma imagem binária
I = imread("J.bmp");

% Define um elemento estruturante do tipo quadrado 3x3
SE_square = strel('square', 3);
% Aplica a convolução da imagem original (I) com o elemento estruturante (SE)
erodesq = imerode(I, SE_square);

% Define um elemento estruturante do tipo diamante 3x3
SE_diamond = strel('diamond', 3);
% Aplica a convolução da imagem original (I) com o elemento estruturante (SE)
erodedm = imerode(I, SE_diamond);

% Define um elemento estruturante do tipo linha
SE_line = strel('line', 9, 0);
% Aplica a convolução da imagem original (I) com o elemento estruturante (SE)
erodel = imerode(I, SE_line);

% Exibe resultados
subplot(2,2,1); imshow(I); title("Imagem Original");
subplot(2,2,2); imshow(erodesq); title("Erosão com Square SE 3x3");
subplot(2,2,3); imshow(erodedm); title("Erosão com Diamond SE 3x3");
subplot(2,2,4); imshow(erodel); title("Erosão com Line SE");
```

Operações Morfológicas: Erosão

Imagem Original

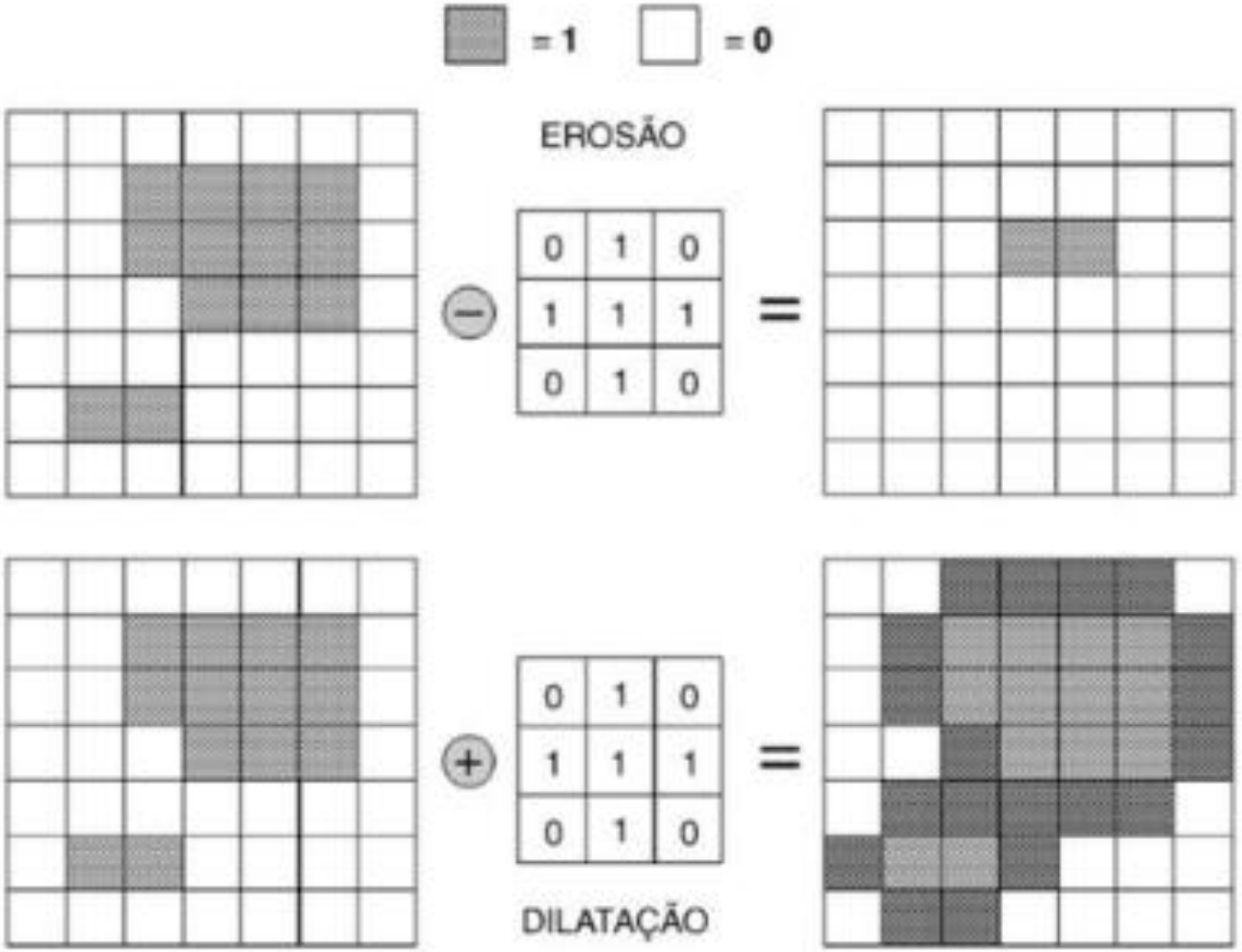
Erosão com Square SE 3x3

Erosão com Diamond SE 3x3



Erosão com Line SE

Operações Morfológicas



Operações Morfológicas: Abertura

- **Abertura** pode ser usada para suavizar o contorno de uma imagem, quebrar istmos estreitos, eliminar proeminências delgadas, abrir buracos e eliminar ruídos na imagem.

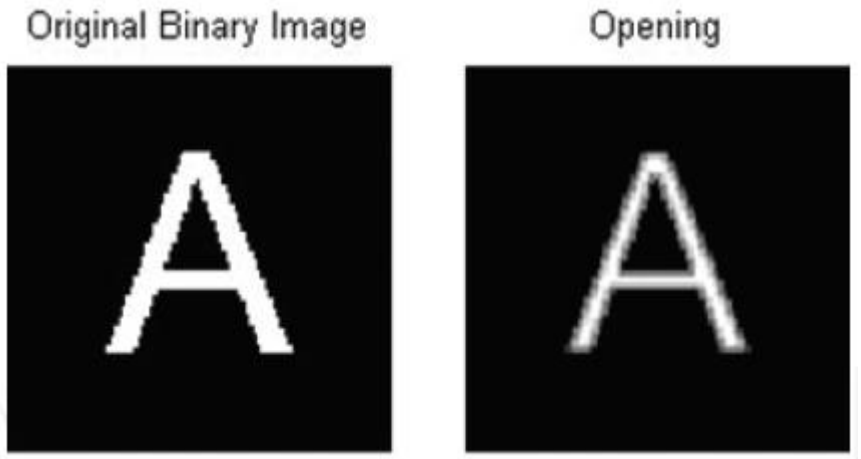
$$C = (A \ominus B) \oplus B$$

onde:

C é a nova imagem;

A é a imagem original;

B é o elemento estruturante.



Operações Morfológicas: Abertura

```
% Lê uma imagem binária
I = imread("J_Open.bmp");

% Define um elemento estruturante do tipo diamante com raio 3
SE = strel("diamond", 3);

% Aplica a convolução da imagem original (I) com o elemento estruturante (SE)
open = imopen(I, SE);

% Exibe resultados
subplot(2,2,1); imshow(I); title("Imagem Original");
subplot(2,2,2); imshow(open); title("Abertura");
```

Imagem Original

Abertura

Operações Morfológicas: Fechamento

- **Fechamento** pode ser usado para fundir pequenas quebras, alargar golfos estreitos e fechar buracos na imagem.

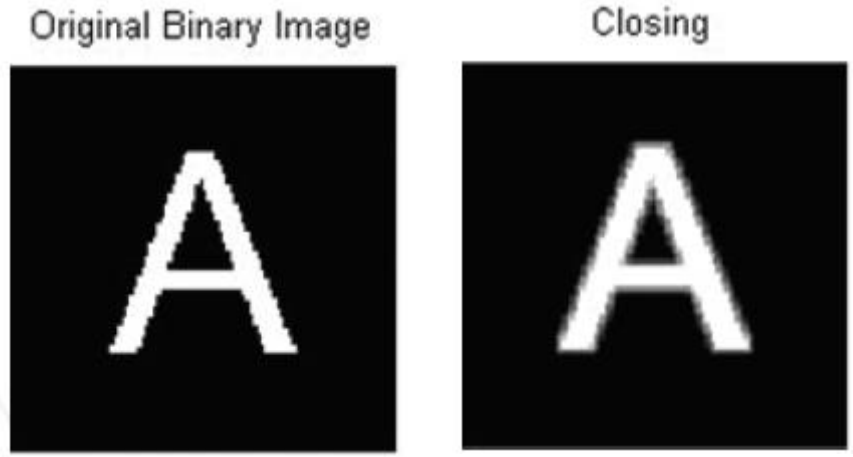
$$C = (A \oplus B) \ominus B$$

onde:

C é a nova imagem;

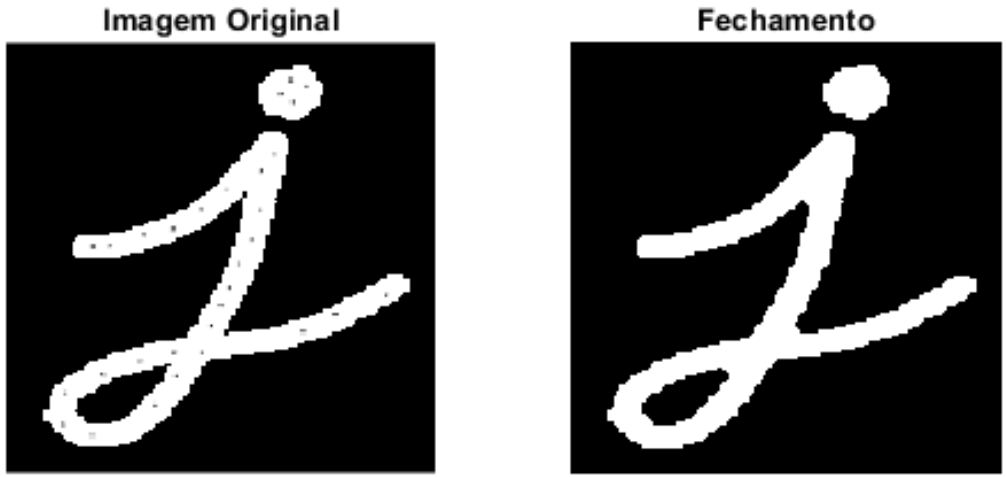
A é a imagem original;

B é o elemento estruturante.



Operações Morfológicas: Fechamento

```
% Lê uma imagem binária
I = imread("J_Close.bmp");

% Define um elemento estruturante do tipo disco com raio 2
SE = strel('disk', 2, 0);

% Aplica a convolução da imagem original (I) com o elemento estruturante (SE)
close = imclose(I, SE);

% Exibe resultados
subplot(1,2,1); imshow(I); title("Imagem Original");
subplot(1,2,2); imshow(close); title("Fechamento");
```



Operações Morfológicas: Abertura e Fechamento

```
% Lê uma imagem binária
I = imread("J.bmp");

% Define um elemento estruturante do tipo disco com raio 4
SE = strel('disk', 4, 0);

% Aplica a convolução da imagem original (I) com o elemento estruturante (SE)
open = imopen(I, SE);
close = imclose(I, SE);
open_close = imopen(imclose(I, SE), SE);

% Exibe resultados
subplot(2,2,1); imshow(I); title("Imagem Original");
subplot(2,2,2); imshow(open); title("Abertura");
subplot(2,2,3); imshow(close); title("Fechamento");
subplot(2,2,4); imshow(open_close); title("Abertura e Fechamento");
```

Operações Morfológicas: Abertura e Fechamento

Imagem Original

Abertura

Fechamento

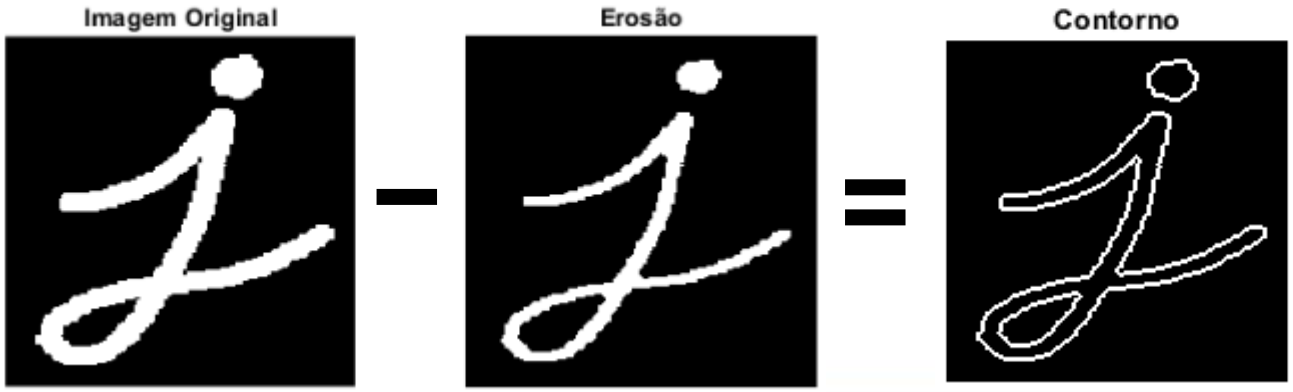
Abertura e Fechamento

Contorno

- Podemos definir o contorno (fronteira ou perímetro) de um objeto erodindo-o com um elemento estruturante apropriado e pequeno e, em seguida, subtraindo o resultado da imagem original. Assim, para uma imagem binária **A** e um elemento estruturante **B**, o contorno $\mathbf{A_p}$ é definido como:

$$A_P = A - A \ominus B$$

```
% Lê uma imagem binária
I = imread("J.bmp");

% Define um elemento estruturante do tipo disco com raio 2
SE = strel('disk', 2, 0);

% Aplica a convolução da imagem original (I) com o elemento estruturante (SE)
erode = imerode(I, SE);

% Subtrai a imagem original pela erosão para gerar o contorno
I_Contorno = I - erode;

% Exibe resultados
subplot(1,2,1); imshow(I); title("Imagem Original");
subplot(1,2,2); imshow(erode); title("Erosão");
figure; imshow(I_Contorno);  title("Contorno");
```



Operações Morfológicas

- O modo e a magnitude da expansão ou redução da imagem dependem necessariamente do *elemento estruturante* **B**.

- A aplicação de uma transformação de dilatação ou erosão a uma imagem sem a especificação de um *elemento estruturante*, não produzirá nenhum efeito.

- Dilatações e erosões são usadas para a criação de transformações mais sofisticadas, as quais conduzem a vários resultados relevantes quanto à análise de imagens, dentre os quais se citam:
  - os filtros morfológicos,
  - o preenchimento de buracos,
  - a extração de contornos,
  - o reconhecimento de padrões.

Lista de Exercícios

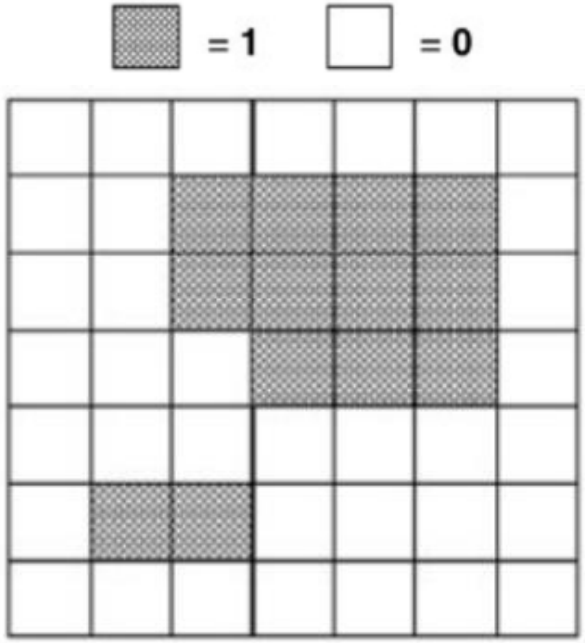
Exercício 1

Considerando os detectores de bordas de *Prewitt* e de *Sobel*, aplique detecção de bordas às imagens dos três canais RGB (por exemplo, com as imagens '*onion.png*' e '*football.jpg*').

Exiba os resultados como uma imagem em cores de três canais e como canais individuais de cor (um por figura).

Exercício 2

Considere a imagem original e o elemento estruturante abaixo e:



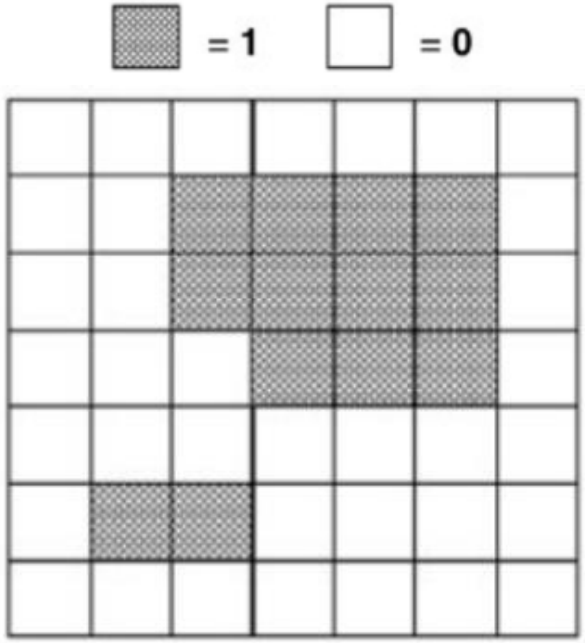
1	0	0
0	1	0
0	0	1

Esboce o resultado da **erosão** com este elemento estruturante.

Esboce o resultado da **dilatação** com este elemento estruturante.

Exercício 3

Considere a imagem original e o elemento estruturante abaixo e:



1	1	1
1	1	1
1	1	1

Qual é o resultado de, primeiro, erodir a imagem com este elemento estrutural e, em seguida, dilatá-la com o mesmo elemento estrutural (operação denominada fechamento)?

Qual é o resultado de, primeiro, dilatar a imagem com este elemento estrutural e, em seguida, erodi-la com o mesmo elemento estrutural (operação denominada abertura)?