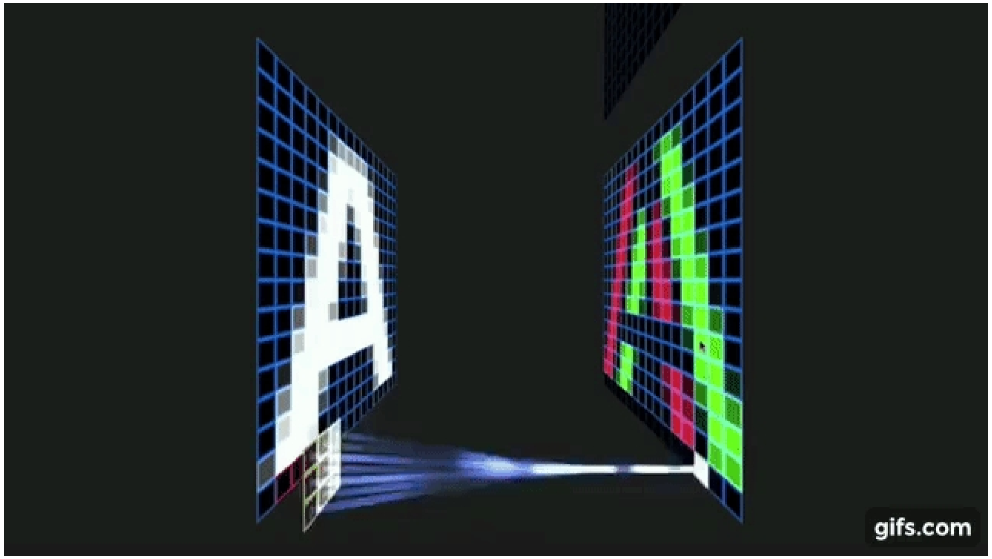
Realce/Filtragem de imagens

- Foi visto o uso de janela deslizante na rotulação de imagens para segmentação por área. Nesse caso a janela deslizante servia apenas como delimitadora da vizinhança como suporte à determinação da conexão entre regiões;

- Existem outras aplicações dessa estratégia de janela deslizante no processamento de imagens, por exemplo, no uso de convolução discreta em operações de filtragem.



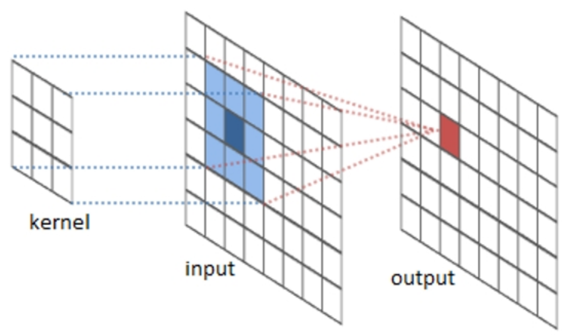
https://www.mobiquity.com/hs-fs/hubfs/CNN03.gif?width=640&name=CNN03.gif

- Convolução discreta

  - Várias operações sobre imagens são realizadas em vizinhanças locais com o emprego de janela deslizante, cada pixel na imagem é processado em uma operação limitada por uma espécie de “região de influência” determinada pela janela deslizante de *NxN* pixels;

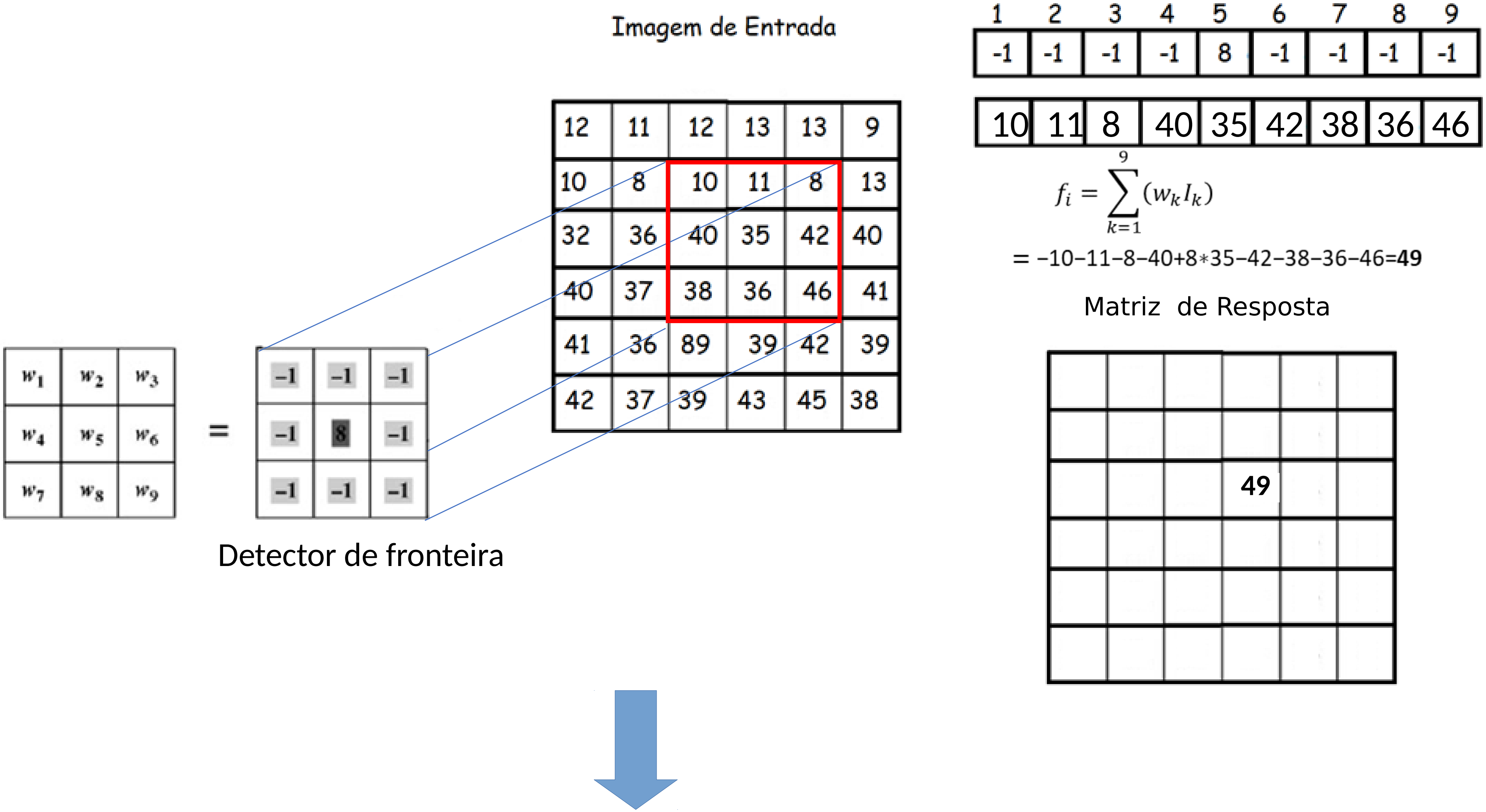
  - Em geral essa técnica é conhecida como convolução discreta (há o conceito de correlação que normalmente é confundido com convolução, os autores costumam misturar tudo e tratam ambos como a mesma coisa: convolução);

  - A janela *NxN* também é conhecida como *kernel* ou núcleo da transformação, normalmente uma matriz quadrada com *N* ímpar;

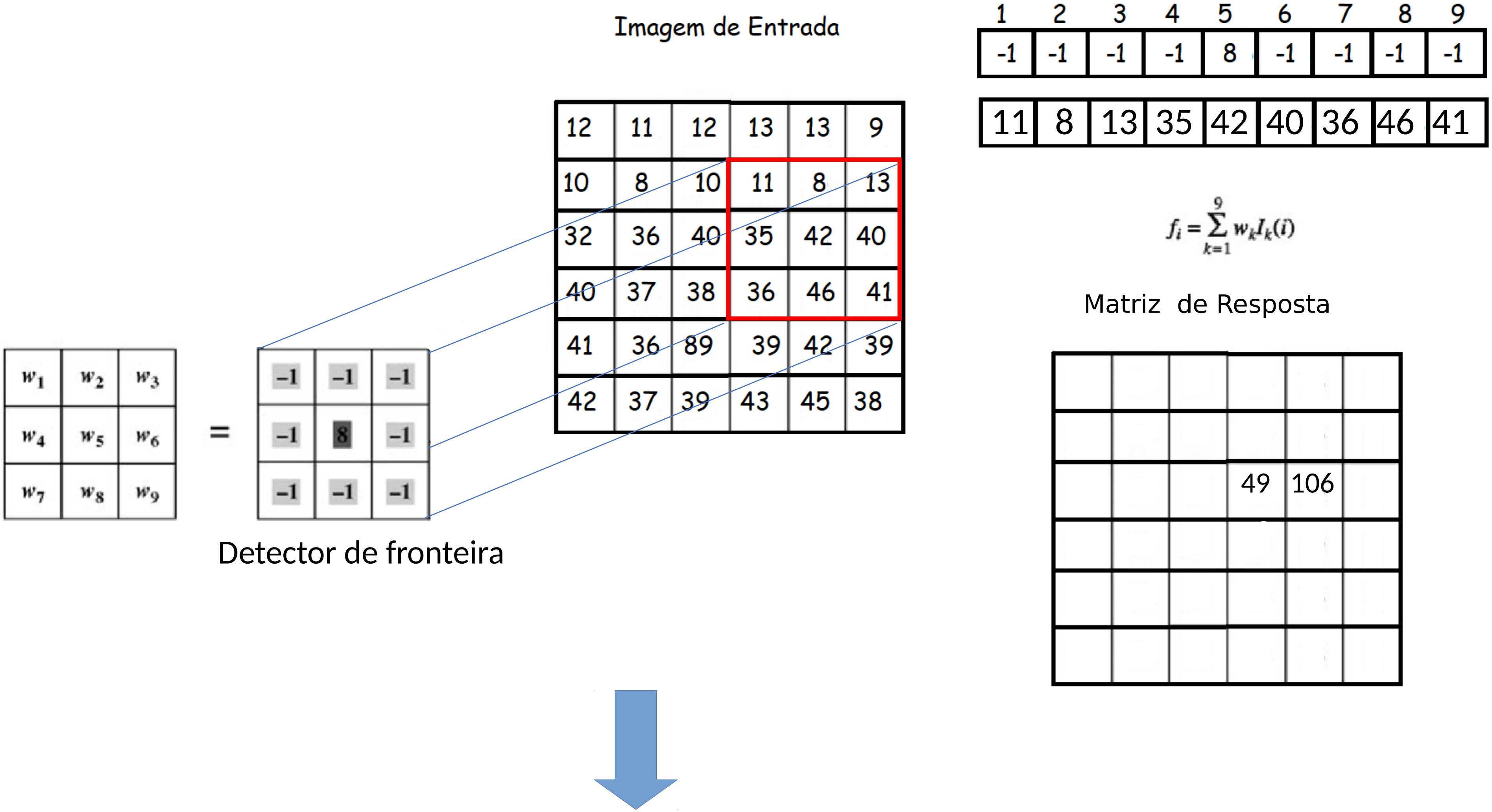


  - Dentre vários exemplos de uso da convolução discreta temos as operações de filtragem.

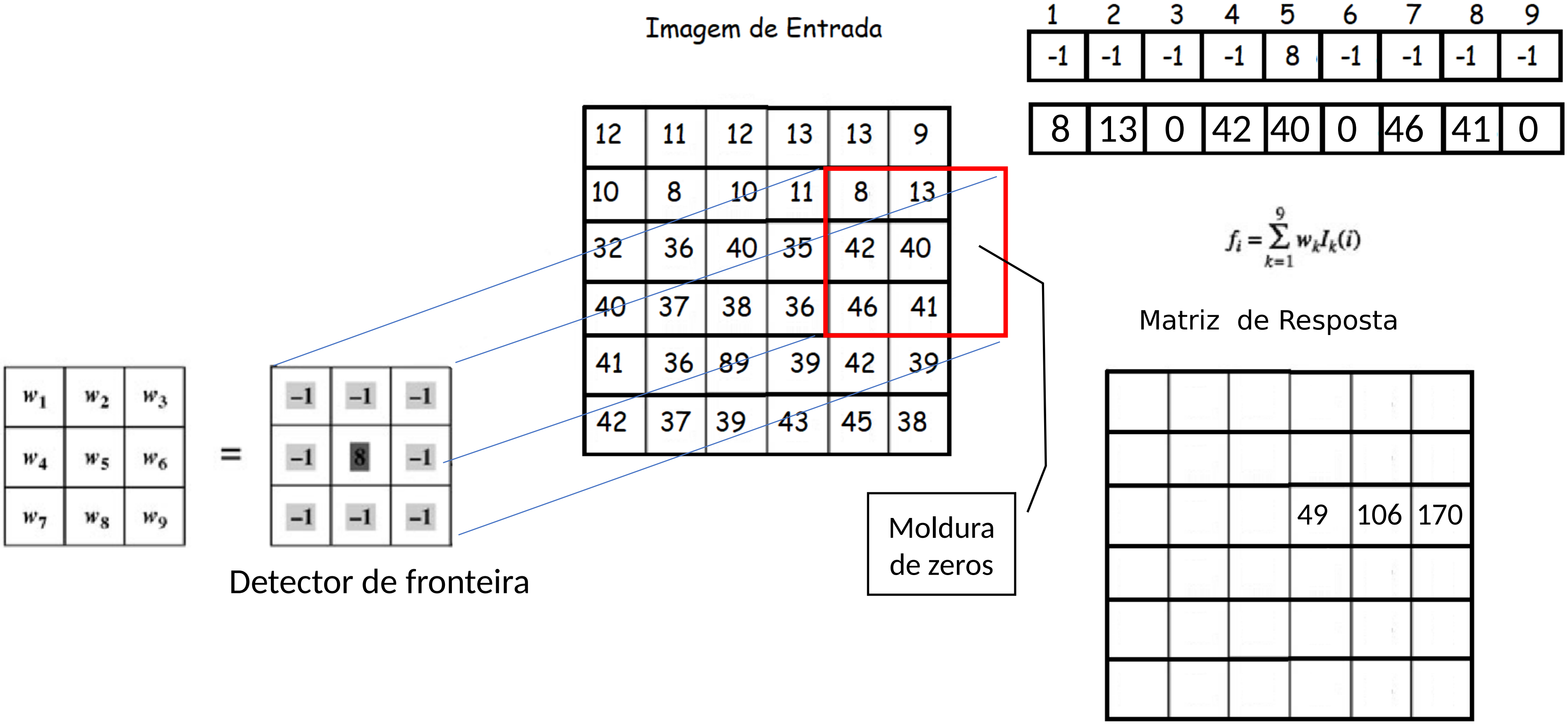
- Filtragem de imagens
  - Abaixo: uma janela com pesos (w) é utilizada em uma filtragem linear:



- Filtragem de imagens
  - … a janela é deslocada pixel a pixel e uma resposta é gerada na imagem de saída:



- Filtragem de imagens
  - … a janela é deslocada pixel a pixel e uma resposta é gerada na imagem de saída:



… esse processo continua até que toda a imagem de entrada tenha sido processada

Os passos na filtragem linear:

(1) Definir o janela/núcleo do filtro.

(2) Deslizar o núcleo sobre a imagem de modo que o pixel central do núcleo coincida com cada pixel-alvo na imagem.

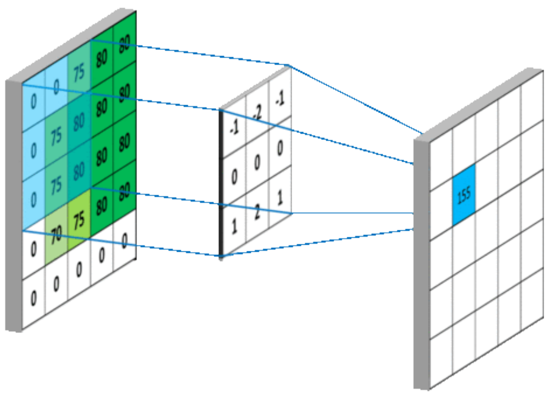
(3) Multiplicar os pixels sob o núcleo pelos correspondentes valores (pesos) no núcleo e somar os resultados.

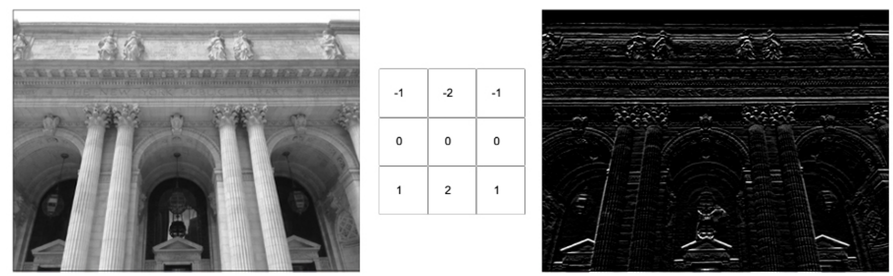
(4) Para cada pixel-alvo, copiar o valor resultante na mesma posição de uma nova imagem (filtrada).

Os passos da filtragem não linear são idênticos aos da linear, a única diferença é que os valores filtrados resultarão de alguma operação não linear, por exemplo:

$$f_i = \sum_{k=1}^{N} \left(w_{k_1} I_k^2(i) + W_{k_2} I_k(i) + w_{k_3} \right)$$

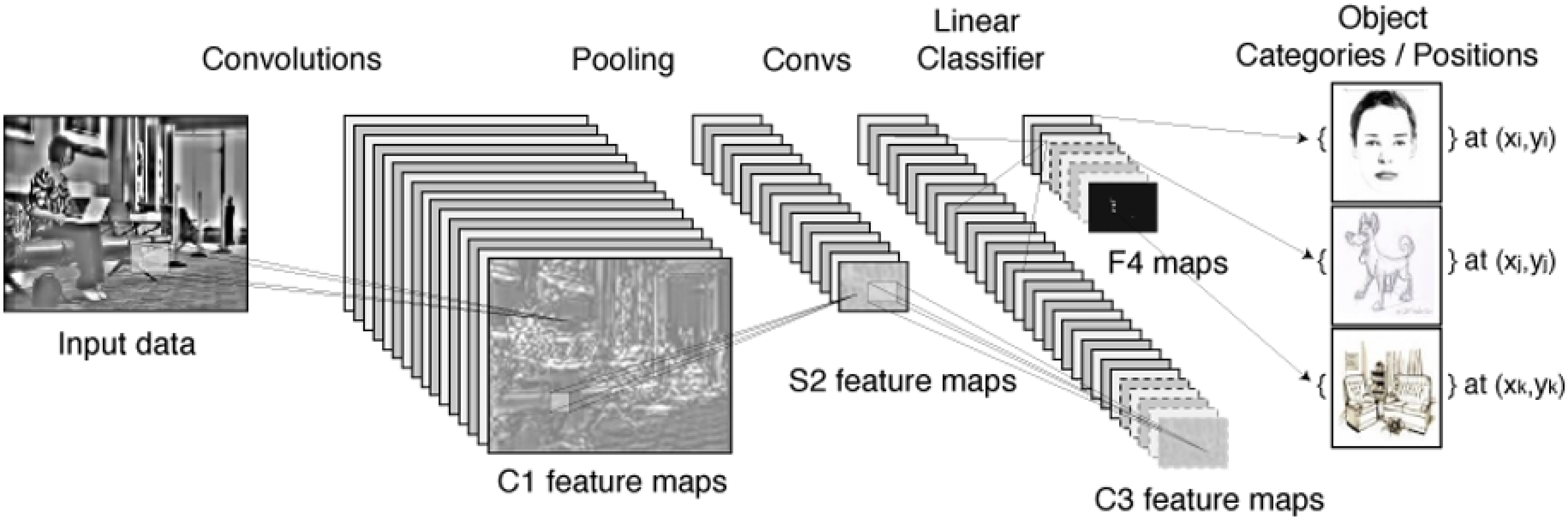
- Esse princípio da convolução discreta pode ser aplicado em diversas outras operações, abaixo tem-se o exemplo de um detector de fronteiras:





- Há ótimos tutorias com animações interessantes:
- https://www.mobiquity.com/insights/introduction-to-convolutional-neural-networks
- https://setosa.io/ev/image-kernels/

- A convolução discreta também está presente no fundamento da Rede Neural Convolucional



https://www.mobiquity.com/insights/introduction-to-convolutional-neural-networks

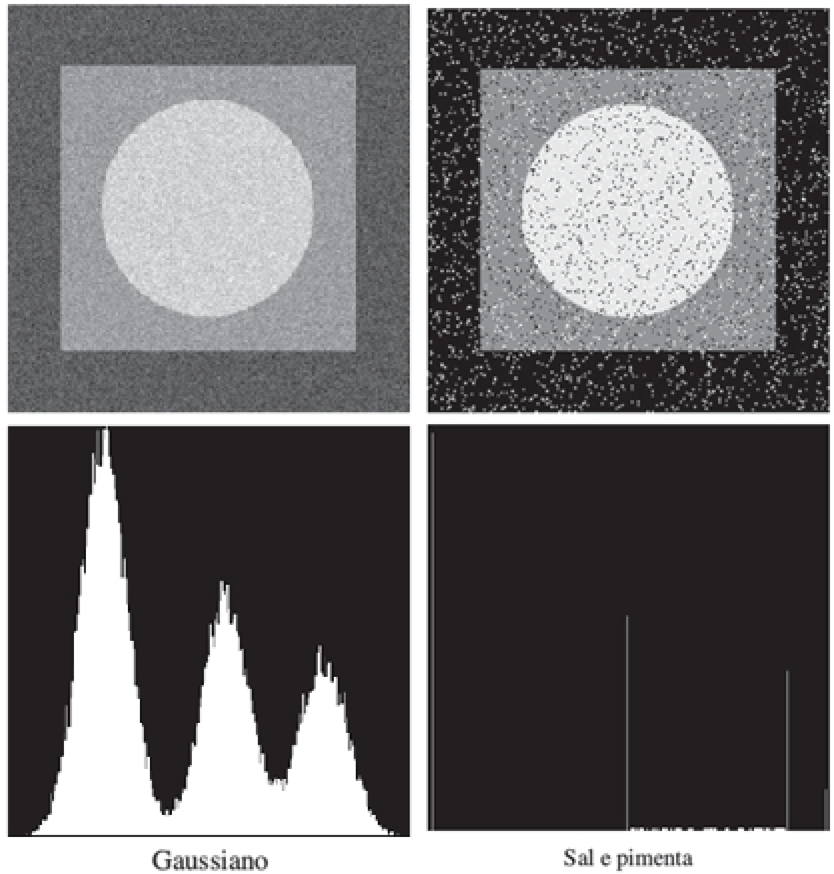
Aplicações de filtragem -- Tratamento de ruído:

- Ruído significa uma pequena variação (aleatória) sofrida pelo sinal em torno de seu verdadeiro valor devido a fatores externos ou internos durante o processamento da imagem (para maiores detalhes veja a seção 2.3.3 do Solomon).

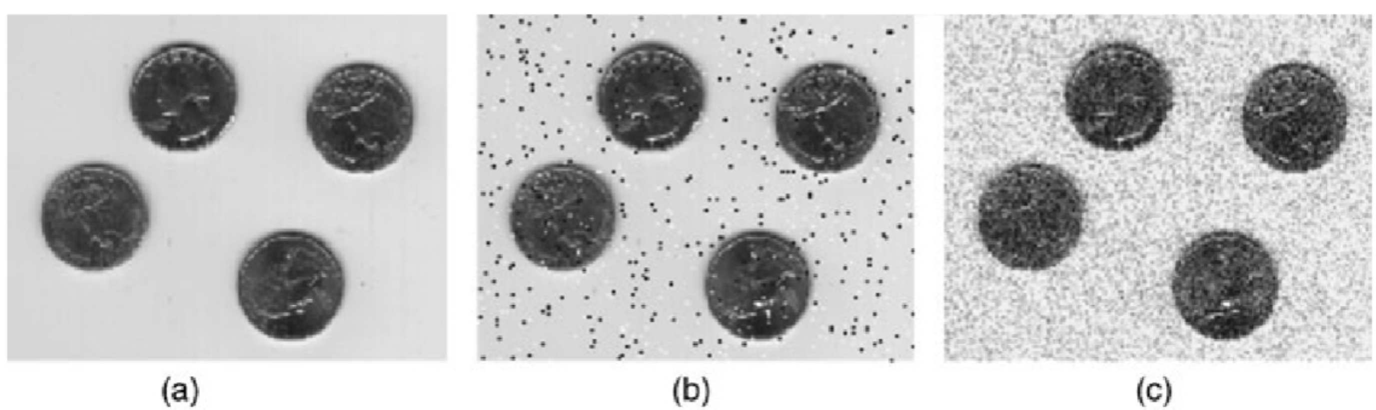
- Ruído é uma informação **espúria** que compromete a eficiência do processamento de imagens.

- É usual aplicar procedimentos de filtragem de ruídos com etapa anterior ao processamento propriamente dito das imagens.

- As duas principais caracterizações de ruído usadas no processamento de imagens são o “Sal-e-pimenta” e o gaussiano, vistas no próximo slide.



Gonzalez e Woods

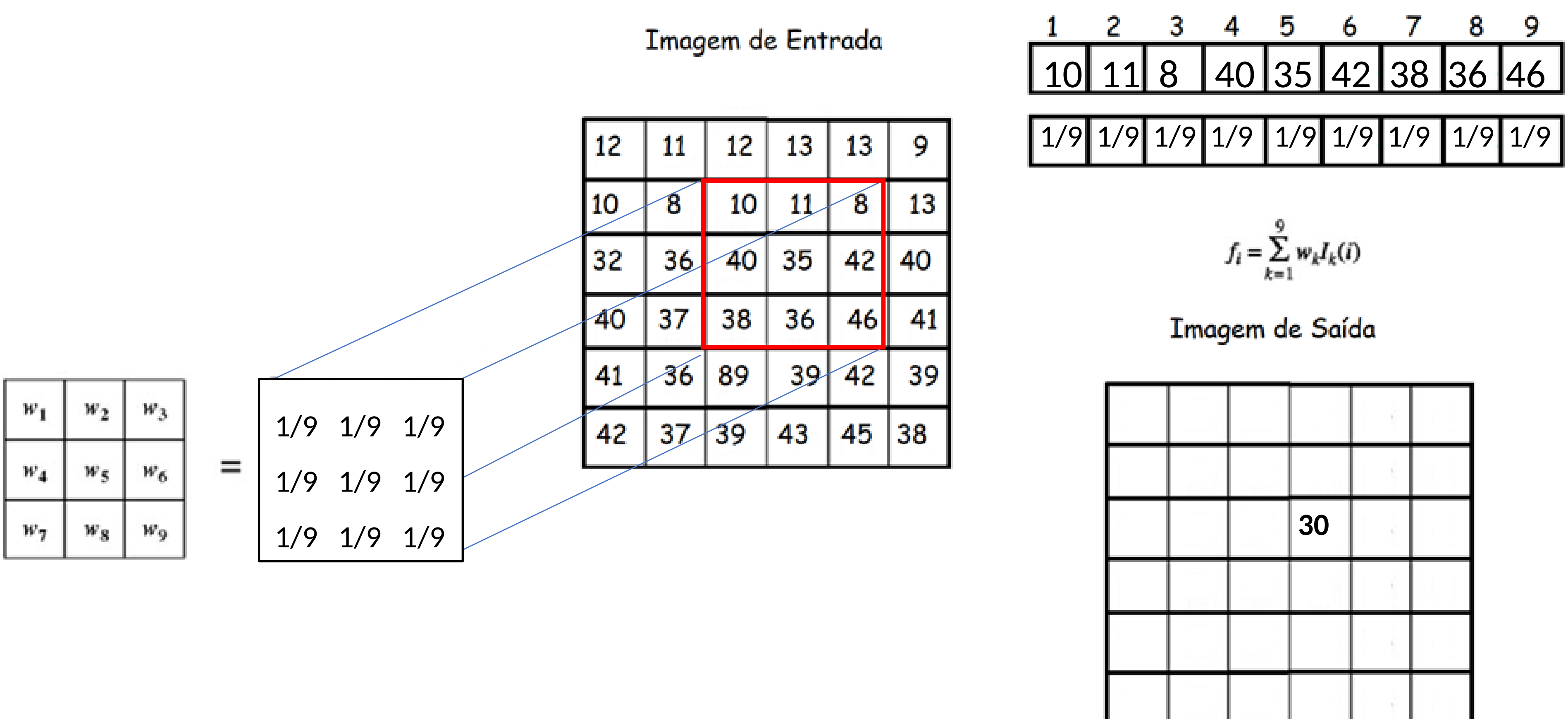


Solomon

Figura 4.3 (a) Imagem original com adição de (b) ruído 'sal e pimenta' e (c) ruído gaussiano.

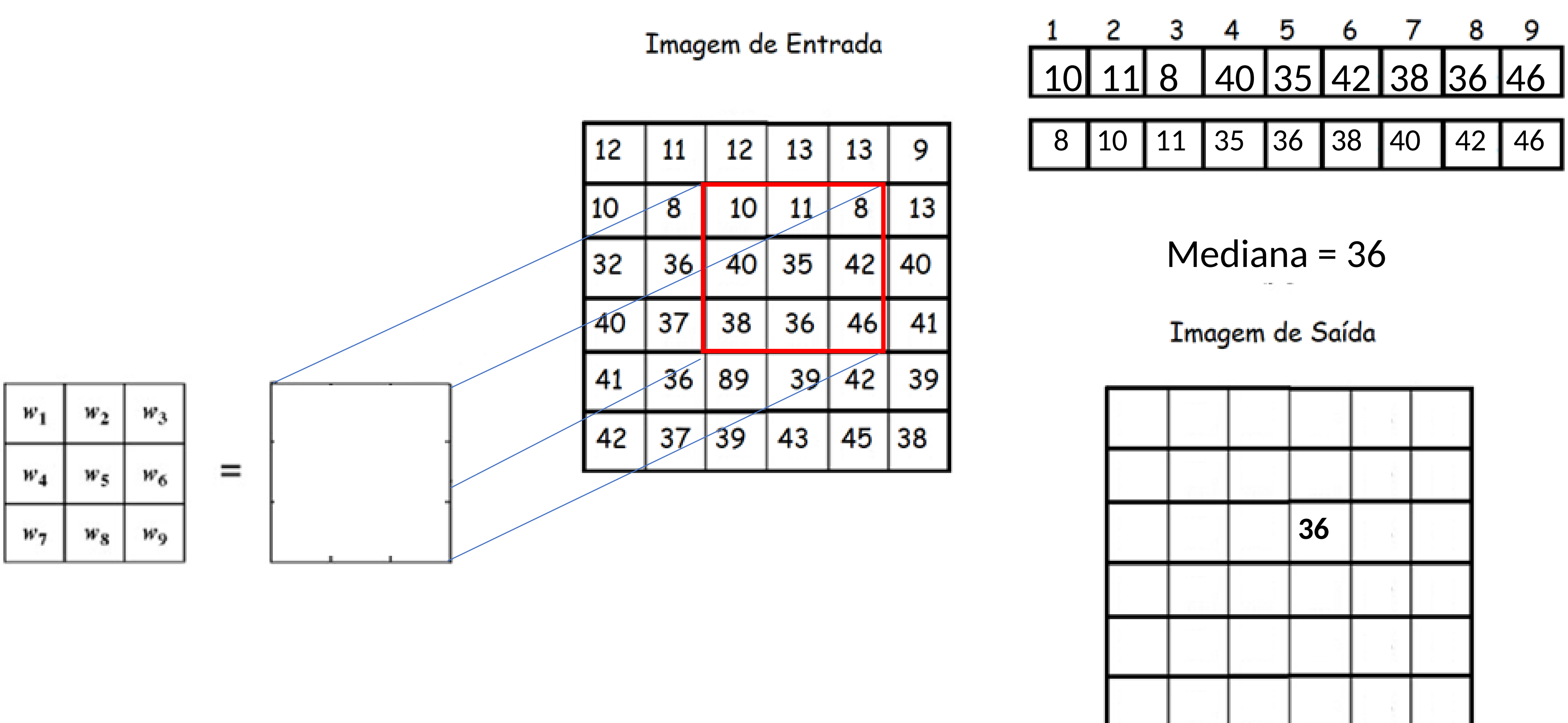
Aplicações de filtragem -- Tratamento de ruído:

- Filtragem pela média:
  - Produz um “borramento” na imagem tornando-a de aspecto mais suave;
  - Pode ser utilizado para tratar ruído, porém há opções mais eficientes porém computacionalmente mais custosas;
  - É implementado com um núcleo (janela deslizante) de tamanho NxM cujos pesos são W_k=1/NM



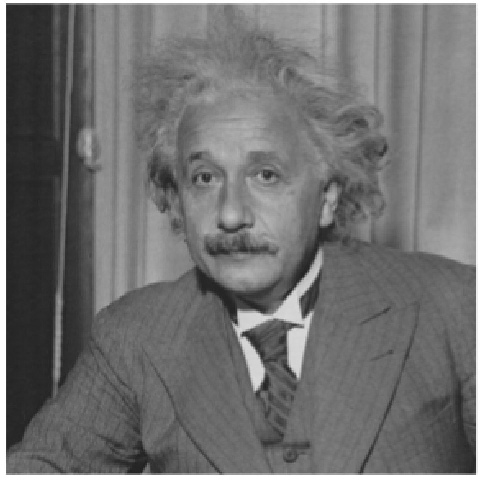
Aplicações de filtragem -- Tratamento de ruído:

- Filtragem pela mediana:
  - A filtragem pela mediana supera as principais limitações da filtragem pela média, às custas de maior gasto computacional;
  - O valor de cada pixel-alvo é substituído pela mediana estatística dos valores dos N x M pixels vizinhos.

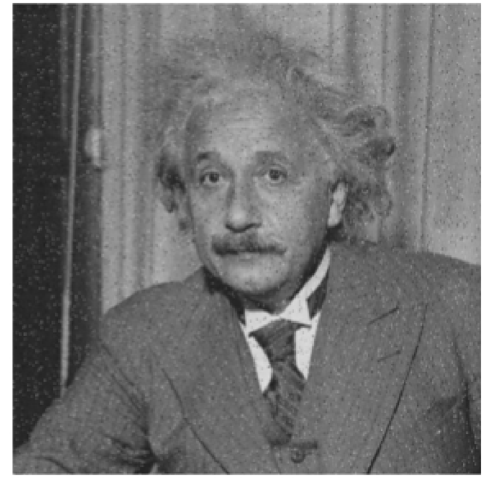


Aplicações de filtragem -- Tratamento de ruído:

- Ruído impulsivo ou sal-e-pimenta:

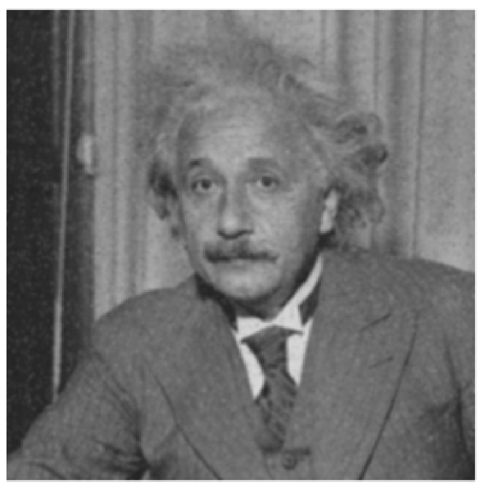


(a) imagem original

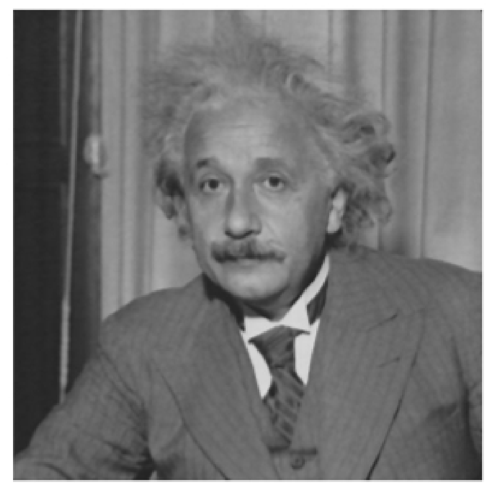


(b) com ruído impulsivo

Hélio Pedrini



(c) após filtro da média 5 $\times$5



(d) após filtro da mediana 5 $\times$5

Aplicações de filtragem -- Tratamento de ruído:

- Ruído impulsivo ou sal-e-pimenta:

Gonzalez e Woods

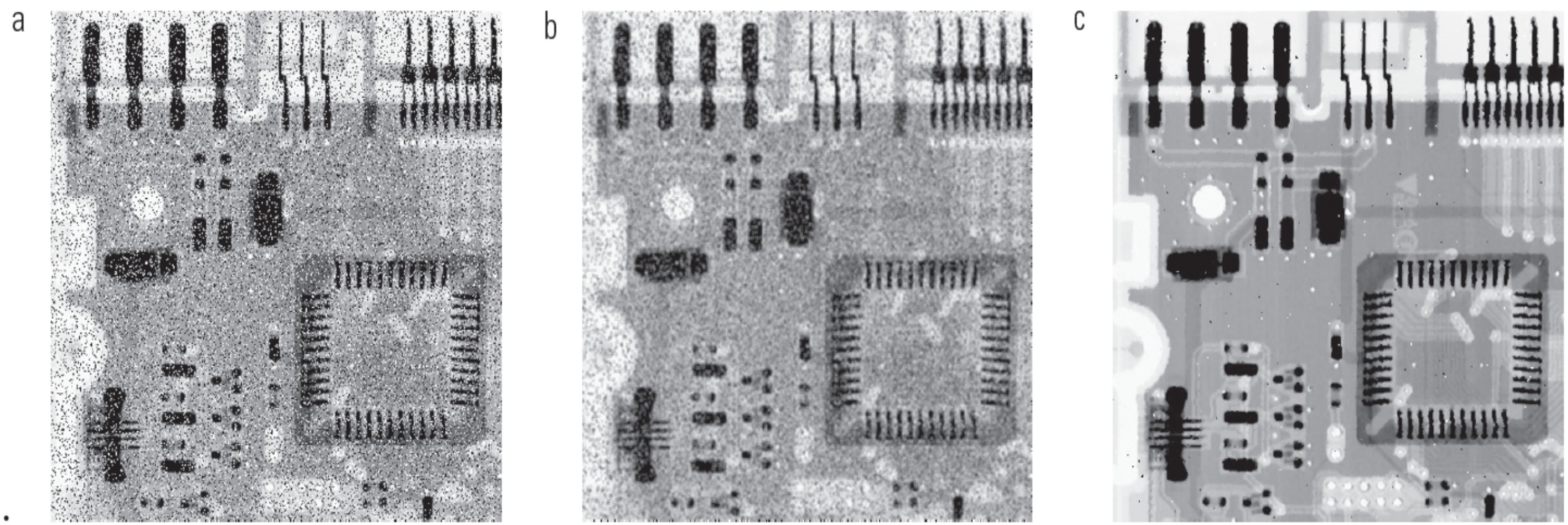


Figura 3.35 (a) Imagem de raios X de uma placa de circuito corrompida pelo ruído sal e pimenta. (b) Redução de ruído com um filtro de média 3×3. (c) Redução de ruído com um filtro de mediana 3×3. (Imagem original: cortesia do Sr. Joseph E. Pascente, Lixi, Inc.)

Aplicações de filtragem -- Tratamento de ruído:

- Filtragem gaussiana:
  - Também produz uma suavização da imagem, porém o grau de suavização é controlado pela escolha do parâmetro de desvio-padrão e não pelo valor absoluto das dimensões do núcleo (como no caso do filtro de média);
  - O núcleo gaussiano é uma aproximação discreta da função gaussiana contínua bidimensional clássica.

Imagem de Entrada

12	11	12	13	13	9
10	8	10	11	8	13
32	36	40	35	42	40
40	37	38	36	46	41
41	36	89	39	42	39
42	37	39	43	45	38

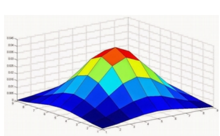
1	2	3	4	5	6	7	8	9
10	11	8	40	35	42	38	36	46
8	12	8	12	20	12	8	12	8

$$f_i = \frac{1}{100} * \sum_{k=1}^{9} (w_k I_k)$$

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

=

8	12	8
12	20	12
8	12	8

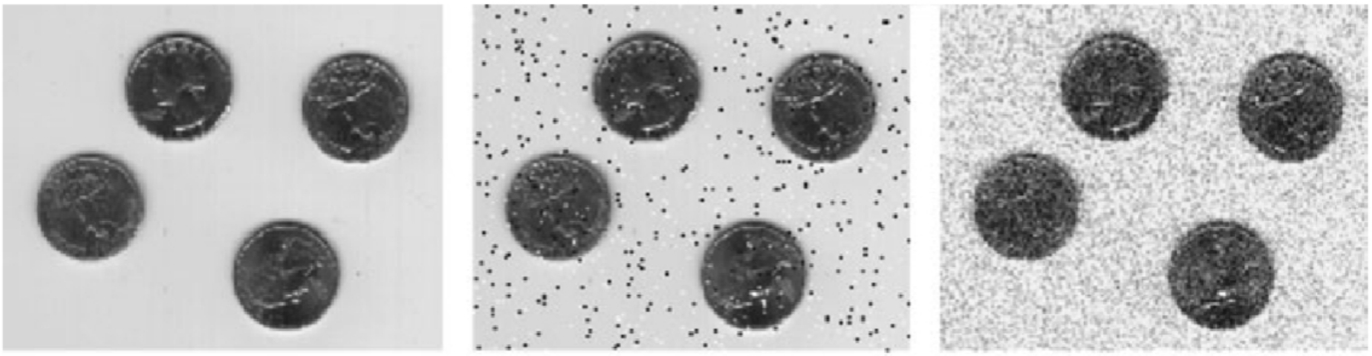


Normalização

Imagem de Saída

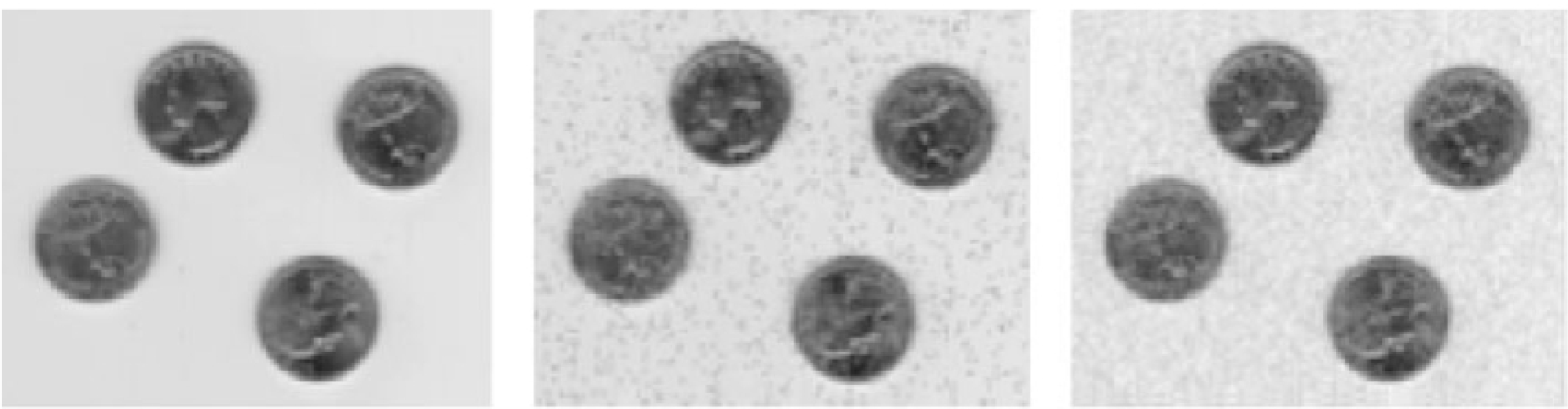
			31		

Aplicações de filtragem -- Tratamento de ruído:



(a) (b) (c)

Figura 4.3 (a) Imagem original com adição de (b) ruído 'sal e pimenta' e (c) ruído gaussiano.



(a) (b) (c)

Pedrini

Figura 4.8 Filtro gaussiano (5×5, com $\sigma = 2$) aplicado às imagens (a) original, (b) com ruído 'sal e pimenta' e (c) com ruído gaussiano.

Aplicações de filtragem: Aguçamento/reforço de nitidez

Normalmente um filtro gaussiano

$$m(x,y) = f(x,y) - \bar{f}(x,y)$$

$$g(x,y) = f(x,y) + c * m(x,y)$$

c >= 1
Valores muito altos de ‘c’ podem demandar ajuste de valores de pixels para [0,255]

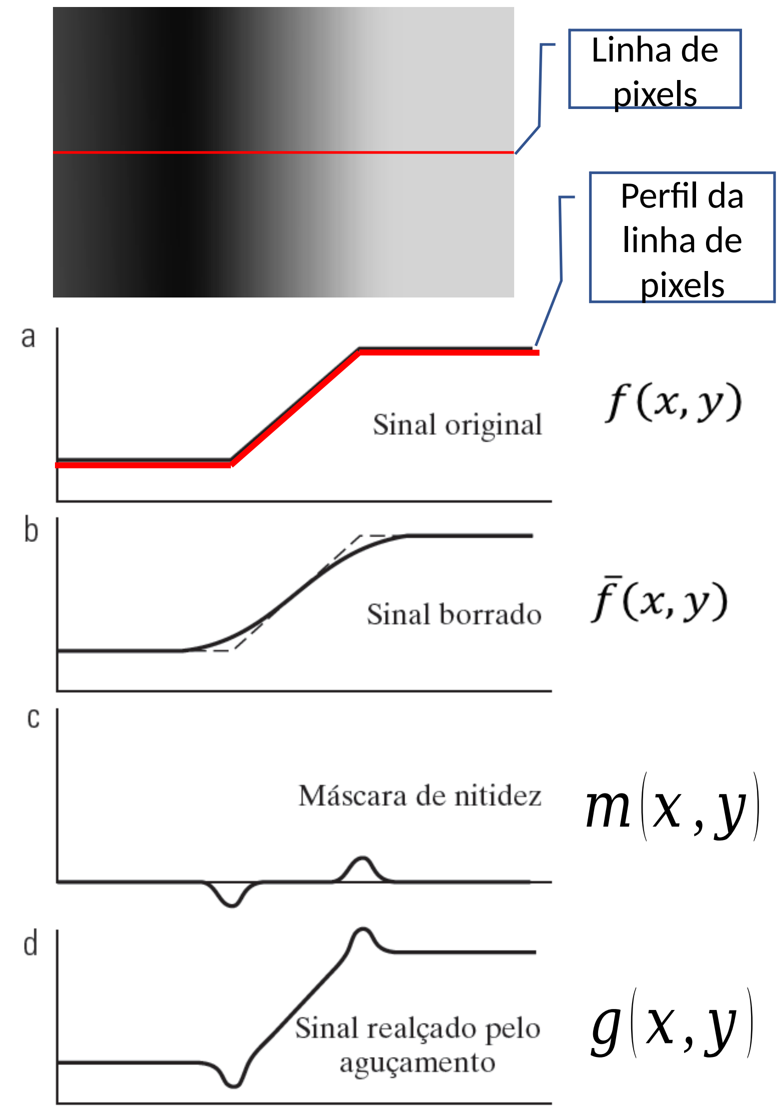
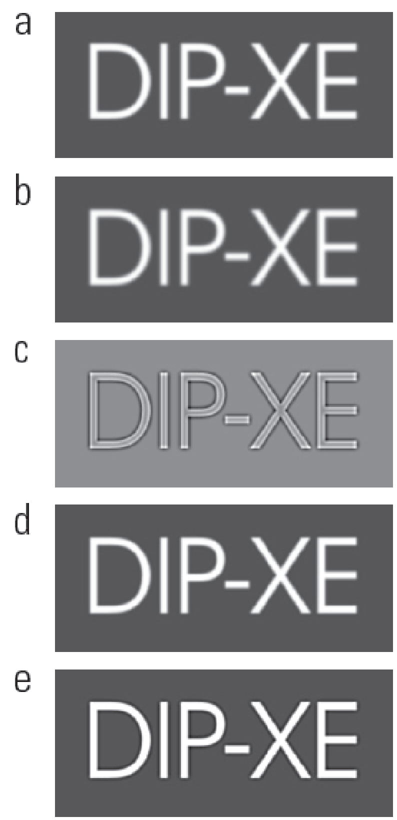


Figura 3.39 Ilustração unidimensional do funcionamento da máscara de nitidez. (a) Sinal original. (b) Sinal borrado com o original tracejado para referência. (c) Máscara de nitidez. (d) Sinal realçado pelo aguçamento obtido pelo acréscimo de (c) a (a). Gonzalez e Woods

Aplicações de filtragem: Aguçamento/reforço de nitidez

$$b)\; m(x,y) = f(x,y) - \bar{f}(x,y)$$

$$g(x,y) = f(x,y) + c * m(x,y)$$



Após operar filtro gaussiano com σ = 3 e janela 3x3

Máscara de nitidez m(x,y)

g(x,y) com c=1

g(x,y) com c=4,5 (high-boost))

Gonzalez e Woods

Figura 3.40 (a) Imagem original. (b) Resultado do borramento com um filtro gaussiano. (c) Máscara de nitidez. (d) Resultado da utilização de uma máscara de nitidez. (e) Resultado da filtragem *high-boost*.

Aplicações de filtragem: Aguçamento/reforço de nitidez

$$m(x,y) = f(x,y) - \bar{f}(x,y)$$

$$g(x,y) = f(x,y) + c * m(x,y)$$

Valores muito altos de ‘c’ podem demandar ajuste dos pixels de g(x,y) para [0,255]:

Ajuste de g(x,y)

100	**765**	**510**	100
200	200	**300**	**300**

$$g_m(x,y) = g(x,y) - \min(g(x,y))$$

0	675	410	0
100	100	200	200

$$g_a(x,y) = 255\left[\frac{g_m(x,y)}{\max(g_m(x,y))}\right]$$

0	255	155	0
38	38	76	76

Em Python a convolução discreta via janela deslizante pode ser apenas uma matriz numpy NxN com N ímpar e cujo centro é "deslizado" sobre cada coordenada *pixel(i,j)* da imagem (dentro de dois laços aninhados, por exemplo)

Para cada cujas *pixel(i,j)* as dimensões da janela deslizante são utilizadas pera determinar uma região no entorno do *pixel(i,j)*

Essa região corresponde a coordenadas no entorno do *pixel(i,j)* na matriz de pixels (slice da matriz de pixels)

Implemente essa função de "deslizamento".

Exercícios utilizando a imagem lua1_gray.jpg (no Moodle)
a) Implemente o método que executa o operador de média sobre uma imagem em tons de cinza, o núcleo da transformada encontra-se abaixo:

$$I_{saida} = media(I_{entrada}).$$

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

1/9	1/9	1/9	1/9	1/9
1/9	1/9	1/9	1/9	1/9
1/9	1/9	1/9	1/9	1/9
1/9	1/9	1/9	1/9	1/9
1/9	1/9	1/9	1/9	1/9

https://www.imageeprocessing.com/2014/04/gaussian-filter-without-using-matlab.html

b) Implemente o método que executa o operador gaussiano uma imagem em tons de cinza $I_{saida} = gaussiano(I_{entrada})$.
O núcleo 3x3 da transformada pode ser calculado conforme abaixo. Utilize σ=0,6 e σ=1,0

c) Execute testes com os operadores implementados em a e b e compare esteticamente os resultados.

$$kernel(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

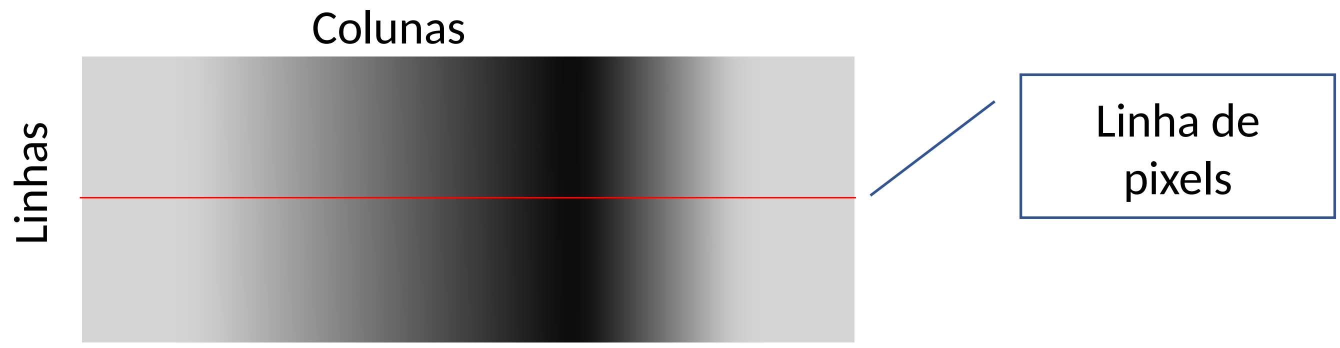
$$X = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} Y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

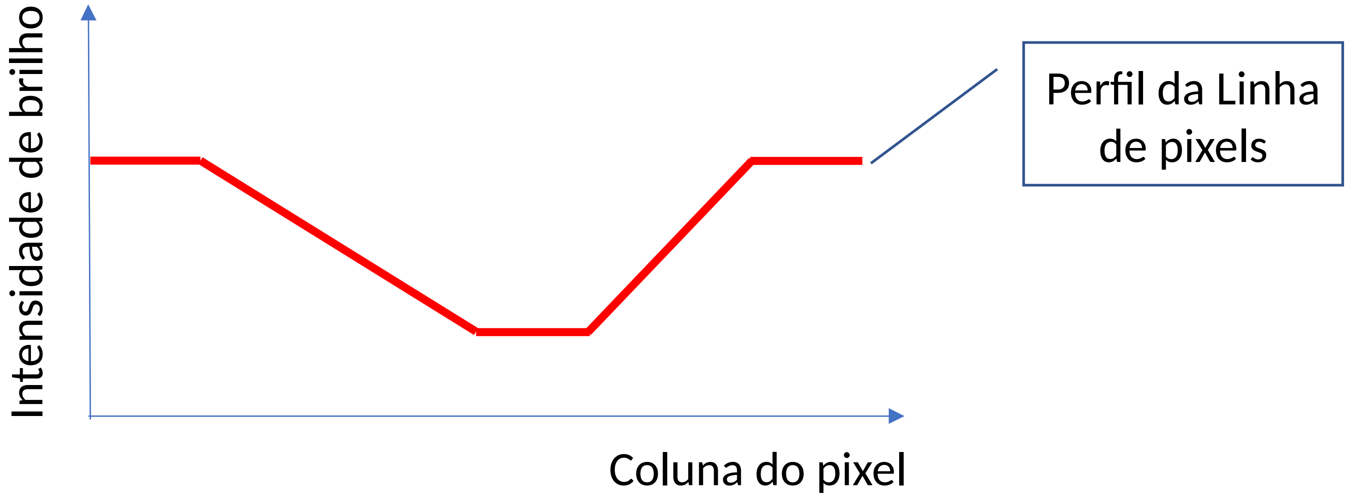
Qual a utilidade e como é possível calcular a derivada de uma imagem (domínio discreto)?

- Existe uma categoria de filtros que é baseada no conceito de derivada

Filtragem por derivadas de primeira e segunda ordem:

- Derivadas têm comportamento importante em áreas planas, rampas e pontos de descontinuidade (degraus, início e fim de rampas).

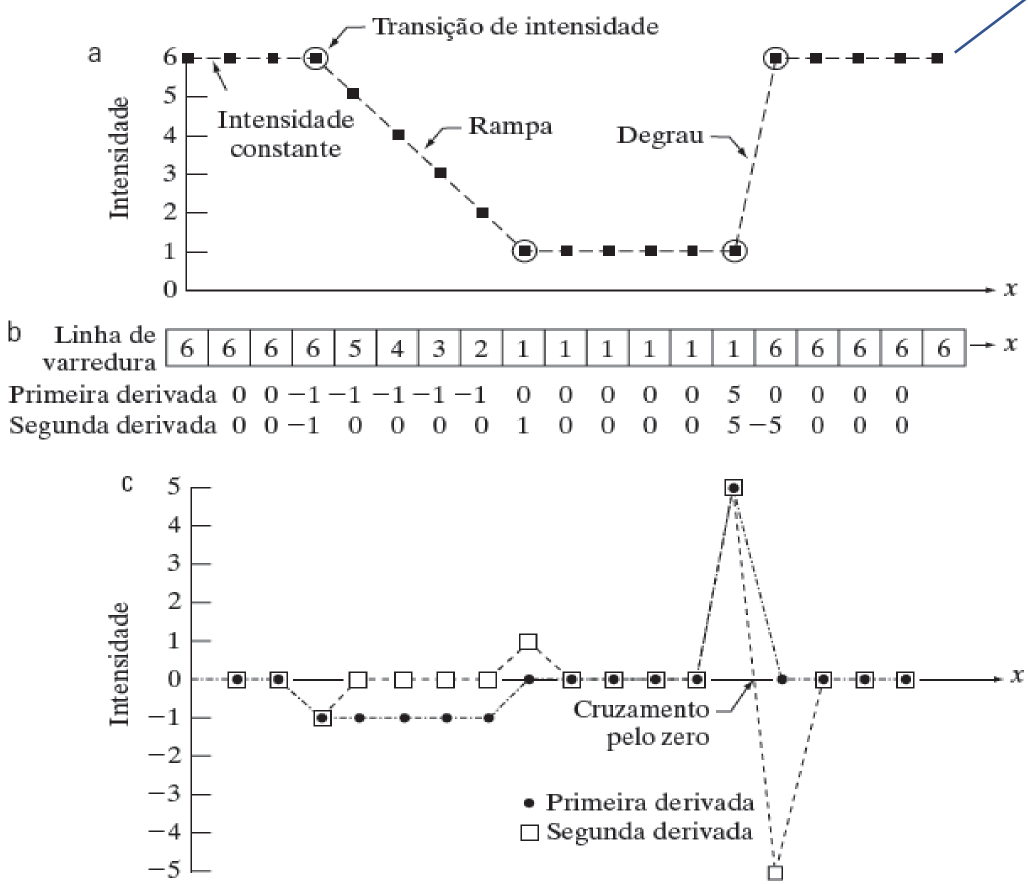




Filtragem por derivadas de primeira e segunda ordem:

- Derivadas têm comportamento importante em áreas planas, rampas e pontos de descontinuidade (degraus, início e fim de rampas).

Linha de pixels



Gonzalez e Woods

Figura 3.36 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.

Laplaciano: filtragem por derivadas de segunda ordem em duas dimensões:

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

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

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

Portanto teremos:

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

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

Essa expressão define os pesos do kernel 3 x 3 do operador
Laplaciano convencional (em cruz)

Laplaciano: filtragem por derivadas de segunda ordem em duas dimensões:

Laplaciano:

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

Gonzalez e Woods

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

Figura 3.37 (a) Máscara de filtragem utilizada para implementar a Equação 3.6-6. (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.

Aplicações de filtragem: Aguçamento/reforço de nitidez

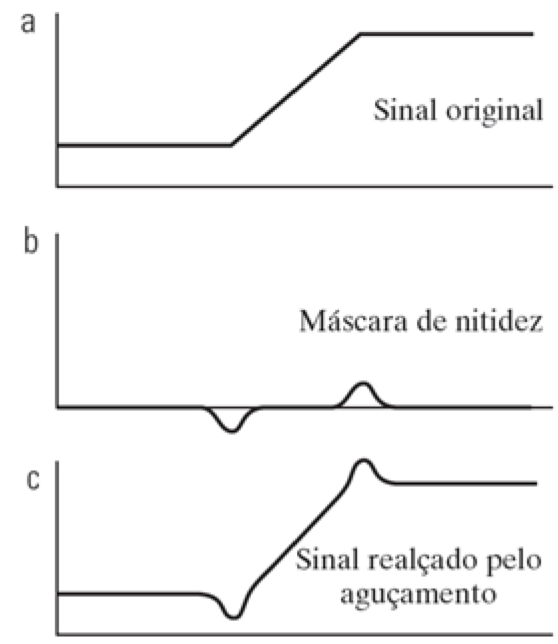
Imagem de saída

Imagem de entrada

$$g(x,y) = f(x,y) + c[\nabla^2 f(x,y)]$$

Laplaciano

Constante de ganho cujo sinal varia conforme o filtro



Gonzalez e Woods

Aplicações de filtragem: Aguçamento/reforço de nitidez

$$b)\ g(x,y) = f(x,y) + c[\nabla^2 f(x,y)]$$

0	1	0
1	−4	1
0	1	0

c) Ajuste de g(x,y)

$$g_m(x,y) = g(x,y) - \min(g(x,y))$$

$$g_a(x,y) = 255\left[\frac{g_m(x,y)}{\max(g_m(x,y))}\right]$$

$$g_{final}(x,y) = f(x,y) + g_a(x,y)$$

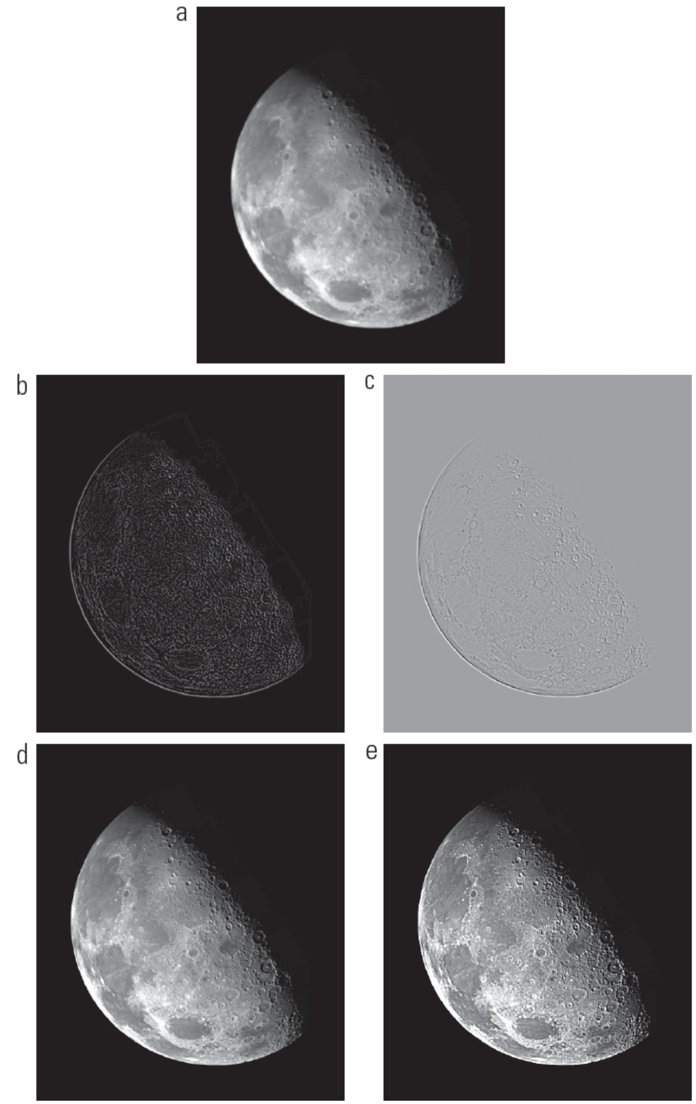


Figura 3.38 (a) Imagem borrada do polo norte da Lua. (b) Laplaciano sem ajuste. (c) Laplaciano com ajuste. (d) Imagem aguçada utilizando a máscara da Figura 3.37(a). (e) Resultado da utilização da máscara da Figura 3.37(b). (Imagem original: cortesia da Nasa.)

Exercício utilizando a imagem 11_test.png (no Moodle):

No livro do Gonzalez e Woods, leia a seção de título *Utilizando segunda derivada para para o aguçamento de imagens – Laplaciano*.

Implemente o filtro de aguçamento utilizando o operador laplaciano nas figuras Figs 3.37-a e 3.37-b. Compare esteticamente os resultados.

Aguçamento: $g(x,y) = f(x,y) + c[\nabla^2 f(x,y)]$

Atenção: verifique que o sinal da constante *c*, contida na expressão do aguçamento (abaixo), depende de características do kernel utilizado (como vimos, existem variações de pesos para núcleos do mesmo operador laplaciano).

Veremos na próxima aula o operador gradiente. Uma outra aplicação de operadores derivativos.