Filtragem espacial

Métodos que usam informação de uma **vizinhança** do píxel da imagem de entrada para calcular o novo valor do correspondente píxel na imagem de saída — **operações locais**

- **Filtragem espacial**
  - **Suavização**
    - Filtro de **Média**
    - Filtro de **Mediana**
    - Filtro **Gaussiano**
    - …
  - **Detecção de arestas** (ou **orlas**)
    - **Gradiente** (op. Roberts, op. Sobel, …)
    - Operador **Laplaciano**
    - Operador **LoG** (*Laplacian of Gaussian*)
    - …
  - **Realce da imagem** (*image sharpening*)

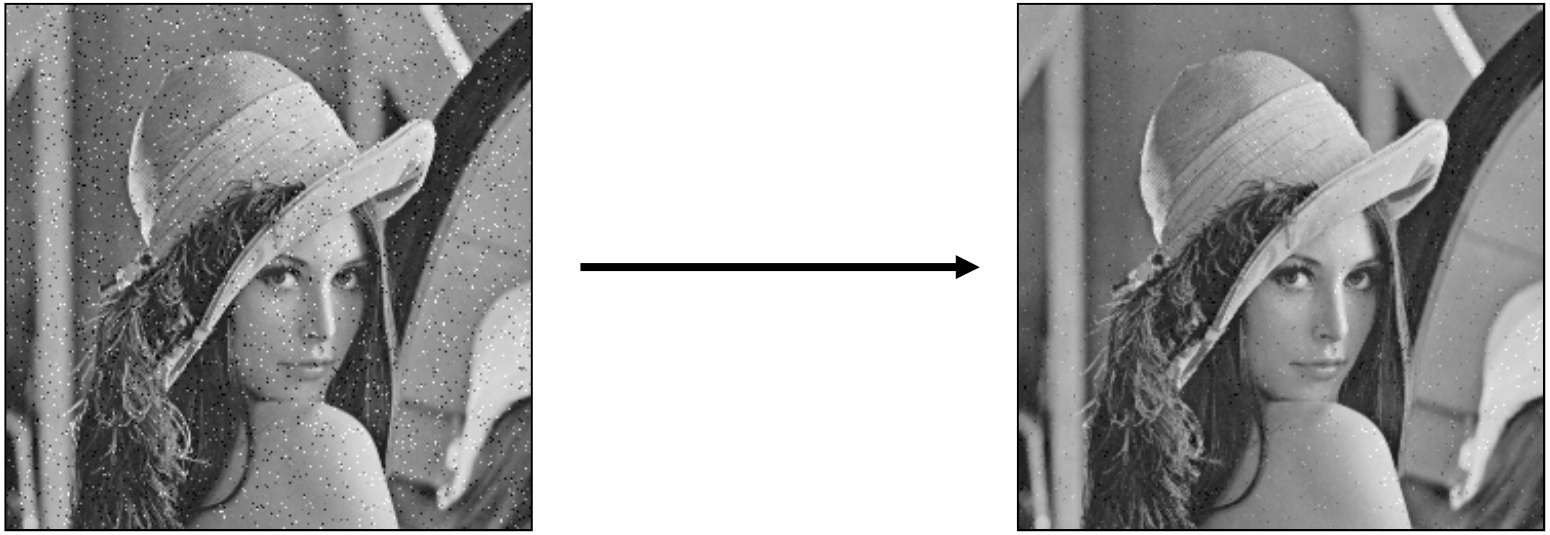
Filtragem espacial

Operadores de suavização:

Têm como objectivo a **eliminação de ruído** ou de outras pequenas perturbações que afectem o sinal imagem. São equivalentes aos **filtros passa-baixo** no domínio da Transformada de Fourier.

Como desvantagem, **esbatem a imagem**, suavizando as arestas (orlas) dos objectos presentes.

Exemplo de um operador de suavização



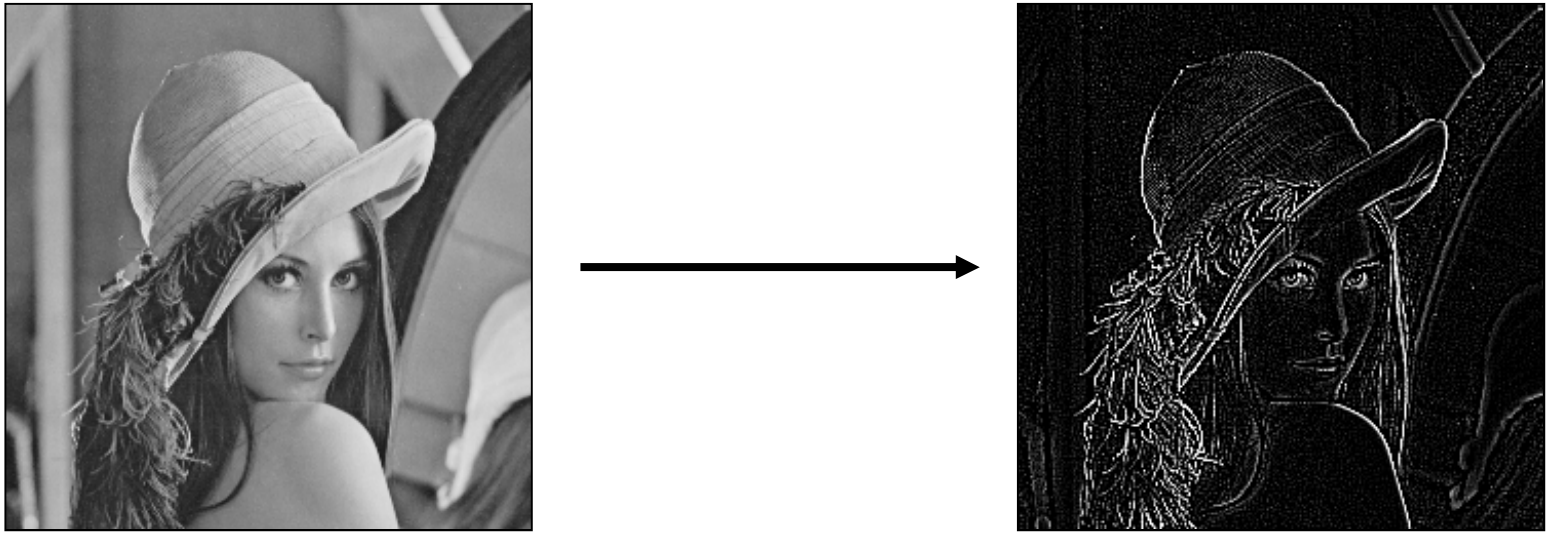
Filtragem espacial

Operadores de detecção de arestas:

Têm como objectivo **realçar as arestas** (orlas) da imagem. São equivalentes aos **filtros passa-alto** no domínio da Transformada de Fourier.

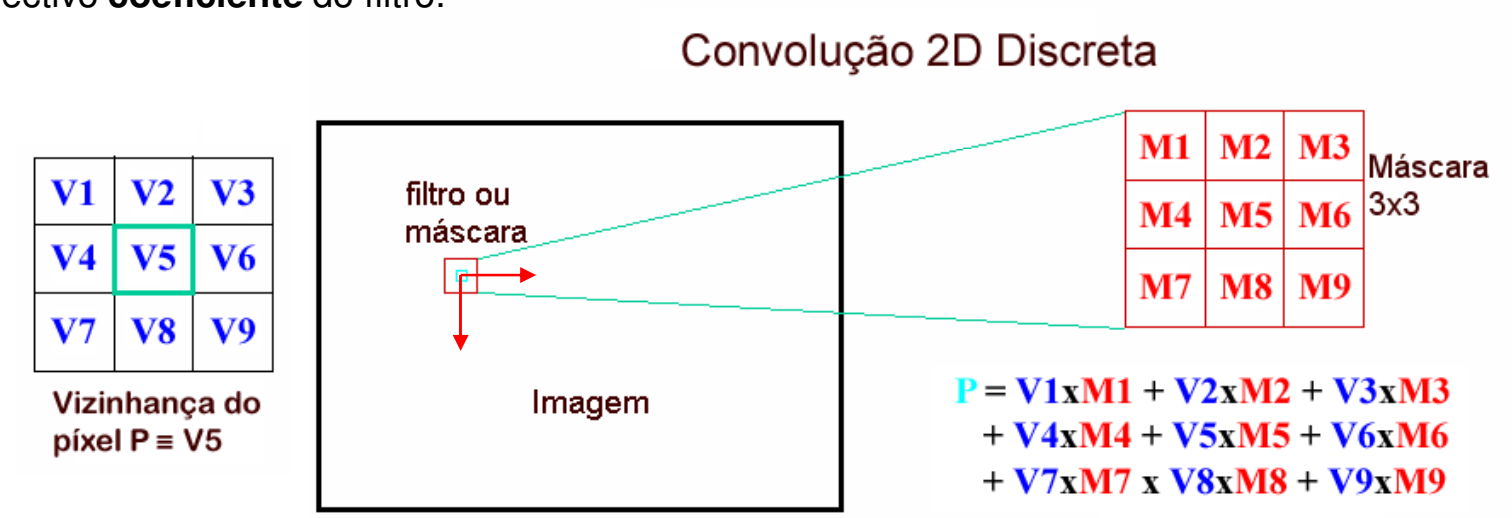
Como desvantagem, **realçam (aumentam) o ruído**, que normalmente é de alta frequência.

Exemplo de um operador de detecção de arestas



Filtragem espacial

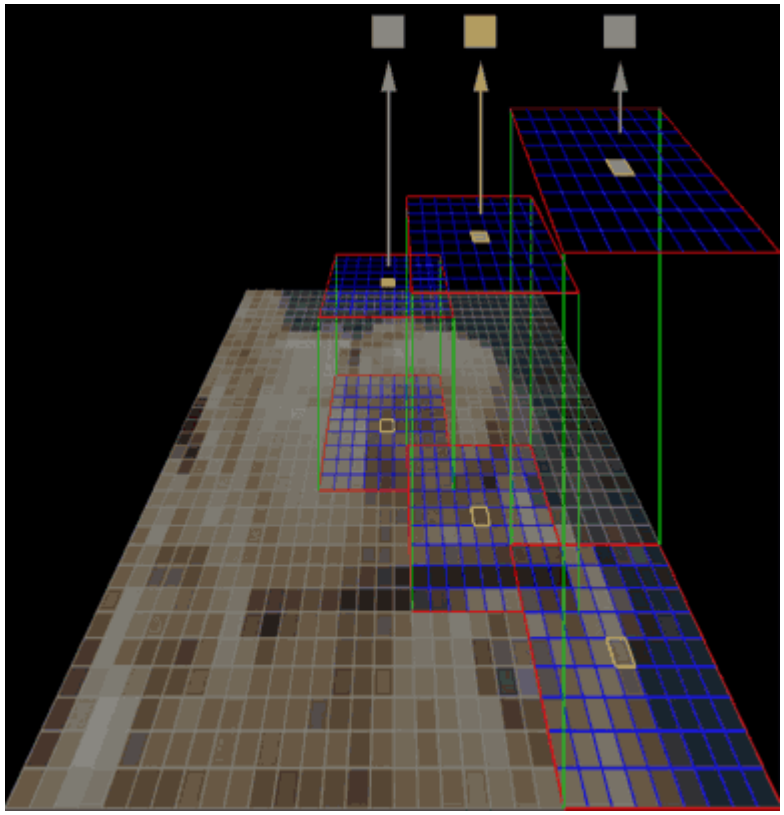
A filtragem espacial é realizada através da **convolução discreta** entre uma imagem de entrada (I_{in}) e um **filtro** ou **máscara** (h), resultando uma imagem de saída (I_{out}): o valor de cada píxel na imagem de saída é calculado através de uma **combinação linear** dos valores da **vizinhança** do píxel (na qual ele próprio se inclui) na imagem de entrada; a contribuição de cada píxel vizinho é determinada pelo respectivo **coeficiente** do filtro.



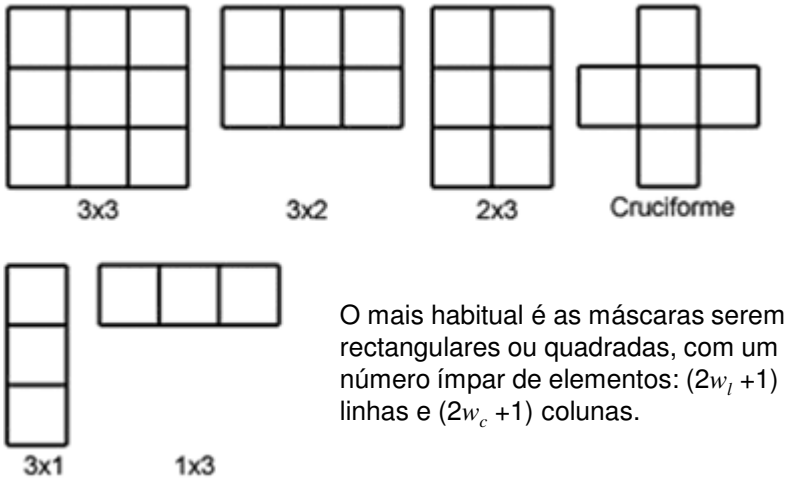
$$I_{out}(i,j) = \sum_{l=-w_l}^{w_l} \sum_{c=-w_c}^{w_c} I_{in}(i+c, j+l).h(w_c+c, w_l+l)$$

Filtragem espacial

A máscara é deslocada sobre toda a imagem (razão pela qual é também designada **janela deslizante**) e em cada posição os valores correspondentes da máscara e dos píxeis da imagem de entrada são multiplicados e somados, sendo o resultado atribuído ao respectivo píxel da imagem de saída.



Existem várias máscaras-padrão para aplicações específicas, onde o tamanho e os coeficientes do filtro determinam as características da operação.



O mais habitual é as máscaras serem rectangulares ou quadradas, com um número ímpar de elementos: $(2w_l + 1)$ linhas e $(2w_c + 1)$ colunas.

Operações de suavização

Filtro de Média

O valor de cada píxel na imagem de saída é calculado pela média dos seus **vizinhos**, **incluindo-se também a ele próprio**.

$$I_{out}(i,j) = \frac{1}{N_V} \sum_{m,n \in V} I_{in}(m,n)$$

$N_V = (2w_l + 1) \times (2w_c + 1)$
(n.º de píxeis na vizinhança)

$$M_8 = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \frac{1}{9} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

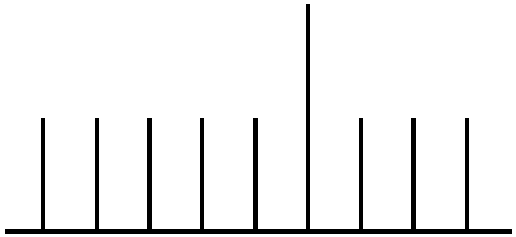
Algumas **características**:

- **Atenua as variações** de intensidade do píxel central em relação aos seus vizinhos.
- **Esbate (desfoca) a imagem** *(blur)* — quanto maior a janela, maior o esbatimento resultante.
- A máscara pode ter dimensões 3x3, 5x5, ou até maiores.
- Tempo de processamento reduzido.

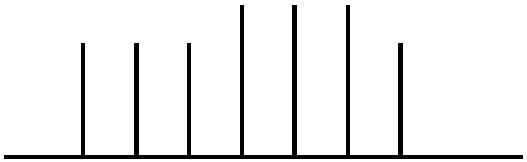
$$M_4 = \frac{1}{5} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Operações de suavização

Exemplo 1D: Filtro de média com 3 elementos

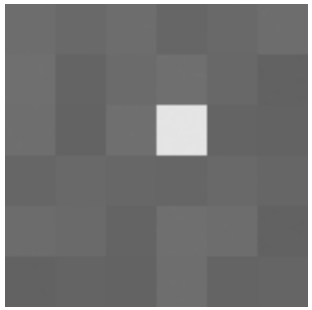


Original (ruído isolado)



Filtrado

Exemplo 2D: Filtro de média 3x3



109	105	109	102	105	109
111	100	107	112	104	98
110	99	112	229	100	99
102	105	103	102	105	109
108	106	100	112	109	98
98	100	99	110	100	101

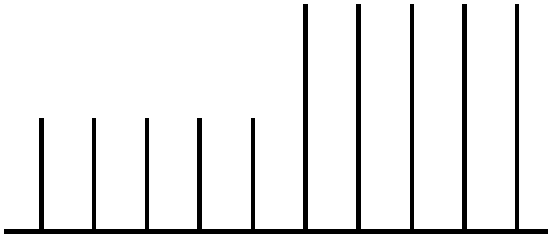
Original

109	105	109	102	105	109
111	107	119	120	118	98
110	105	119	119	118	99
102	105	119	119	118	109
108	102	104	104	105	98
98	100	99	110	100	101

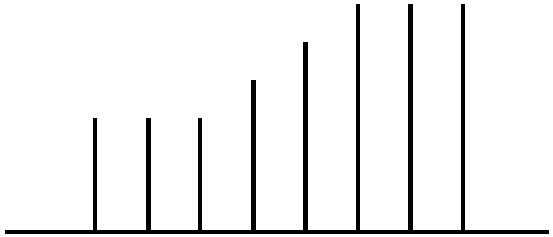
Filtrado

Operações de suavização

Exemplo 1D: Filtro de média com 3 elementos

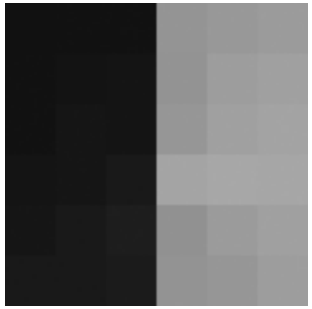


Original (fundo/objecto)



Filtrado

Exemplo 2D: Filtro de média 3x3

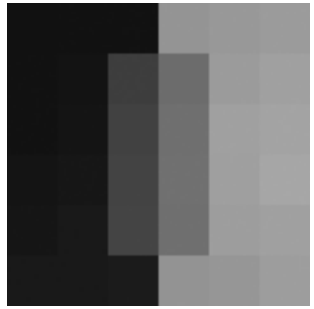


17	18	18	148	152	155
18	19	20	147	157	160
18	21	20	150	160	163
20	22	25	165	168	167
22	25	30	145	155	160
25	25	27	147	150	157

Original

17	18	18	148	152	155
18	19	62	108	155	160
18	20	65	112	160	163
20	23	67	113	159	167
22	25	68	112	157	160
25	25	27	147	150	157

Filtrado



Operações de suavização

Filtro de Média Ponderada

Atribui **maior peso aos píxeis mais próximos** do píxel central do que àqueles que se encontram na periferia da vizinhança → **preserva melhor as arestas** da imagem do que o filtro de média simples.

Exemplos:

$$M_8 = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} = \frac{1}{16}\begin{bmatrix}1\\ 2\\ 1\end{bmatrix}\begin{bmatrix}1 & 2 & 1\end{bmatrix}$$

$$M_4 = \frac{1}{6}\begin{bmatrix}0 & 1 & 0\\ 1 & 2 & 1\\ 0 & 1 & 0\end{bmatrix}$$

Operações de suavização

Filtro de Média condicionado pelos dados

- Procura **evitar o esbatimento** da imagem.
- Só afecta o valor de pontos da imagem que respeitem determinado **critério**.
- Previne o cálculo da média entre píxeis que pertençam a **regiões diferentes** da cena.

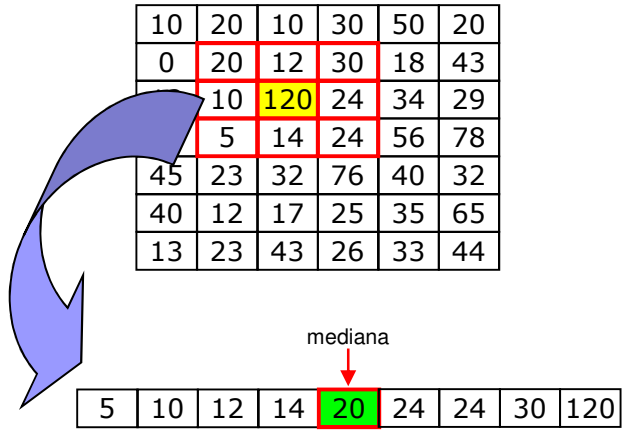
$$I_{out}(i,j) = \begin{cases} \overline{m}_V = \dfrac{1}{M} \displaystyle\sum_{m,n \in V} I_{in}(m,n) & , \quad \left| I_{in}(i,j) - \overline{m}_V \right| \leq T \\ I_{in}(i,j) & , \quad \ldots \end{cases}$$

- **Evita grandes variações** entre os valores dos píxeis das imagens de entrada e de saída.
- Grandes variações entre píxeis vizinhos são atribuídas à presença de **arestas** ou **contornos**, que são assim **preservados**.

Operações de suavização

Filtro de Mediana

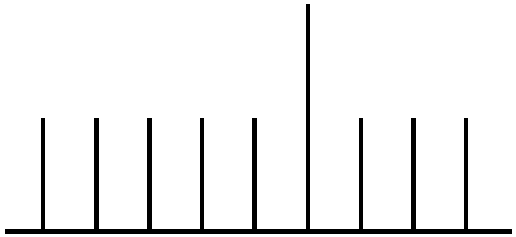
- Filtro **não linear**: o valor de saída não é combinação linear dos valores da vizinhança.
- Considera uma vizinhança em torno de cada píxel, atribuindo-lhe na imagem de saída o **valor central** do vector de **valores ordenados** da vizinhança considerada.
- **Eficaz** a remover pontos de **ruído isolado** com intensidades muitos díspares («**ruído sal e pimenta**»).
- **Preserva arestas e contornos** da imagem: um valor pouco representativo na vizinhança não altera a mediana.



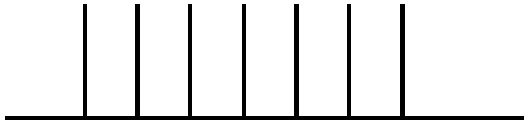
- O valor do píxel é substituído por um existente na vizinhança (na imagem de saída não surgem valores que não constem na imagem inicial).
- Tempos de cálculo superiores ao filtro de média.

Operações de suavização

Exemplo 1D: Filtro de mediana com 3 elementos

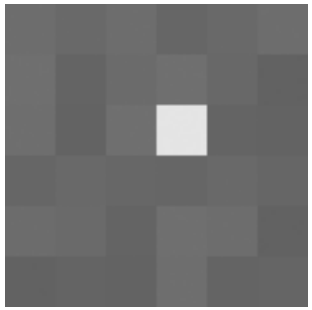


Original (ruído isolado)



Filtrado

Exemplo 2D: Filtro de mediana 3x3

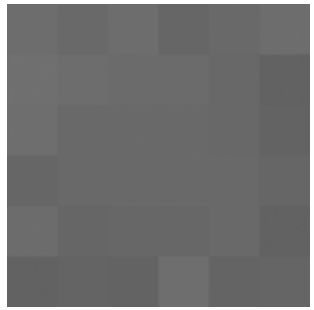


109	105	109	102	105	109
111	100	107	112	104	98
110	99	112	229	100	99
102	105	103	102	105	109
108	106	100	112	109	98
98	100	99	110	100	101

Original

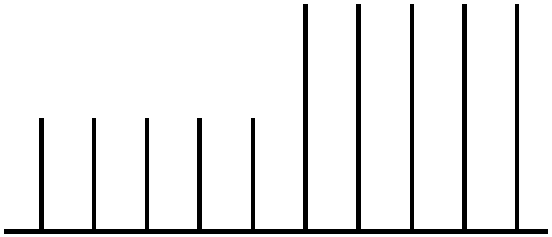
109	105	109	102	105	109
111	109	107	107	104	98
110	105	105	105	104	99
102	105	105	105	105	109
108	102	103	103	105	98
98	100	99	110	100	101

Filtrado

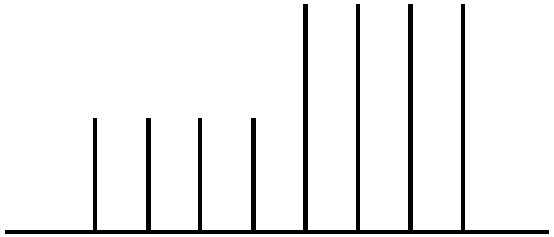


Operações de suavização

Exemplo 1D: Filtro de mediana com 3 elementos

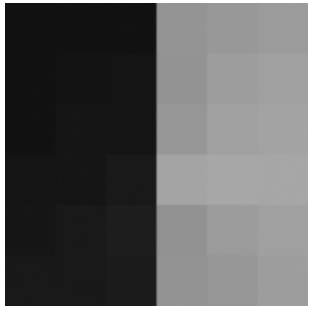


Original (fundo/objecto)



Filtrado

Exemplo 2D: Filtro de mediana 3x3

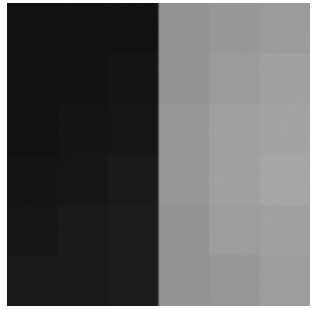


17	18	18	148	152	155
18	19	20	147	157	160
18	21	20	150	160	163
20	22	25	165	168	167
22	25	30	145	155	160
25	25	27	147	150	157

Original

17	18	18	148	152	155
18	18	20	148	155	160
18	20	22	150	160	163
20	22	25	150	160	167
22	25	27	147	157	160
25	25	27	147	150	157

Filtrado



Operações de suavização

Comparativo Média/Mediana: Ruído isolado («sal e pimenta»)

Filtro de média 3x3

Filtro de média 5x5

Filtro de mediana 3x3

Operações de suavização

Comparativo Média/Mediana: Ruído gaussiano

Filtro de mediana 3x3

Filtro de média 3x3

Operações de suavização

Filtro de Moda

Para uma dada vizinhança, o valor de cada píxel (na imagem de saída) é o do seu **vizinho mais comum** (considerando os valores da imagem de entrada).

Filtro dos k vizinhos mais próximos

Para uma dada vizinhança, o novo valor de cada píxel, $I_{out}(i,j)$, será igual à **média** dos k **vizinhos** cujo valor está **mais próximo** do seu valor original, $I_{in}(i,j)$.

(k é um **parâmetro ajustável**; p. ex., para uma vizinhança 3x3 um valor habitual é k = 6.)

Filtro Sigma

Para uma dada vizinhança, $I_{out}(i,j)$ será igual à média dos **vizinhos** cujos valores não diferem mais do que T unidades do valor original do píxel considerado, $I_{in}(i,j)$.

(O parâmetro T pode ser derivado do **desvio padrão** dos píxeis da imagem, donde o nome do filtro.)

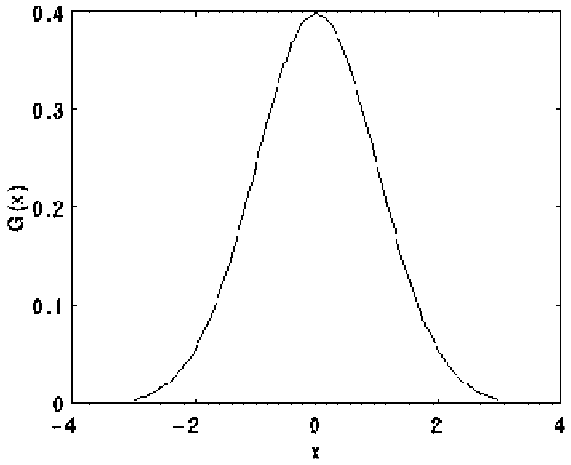
Operações de suavização

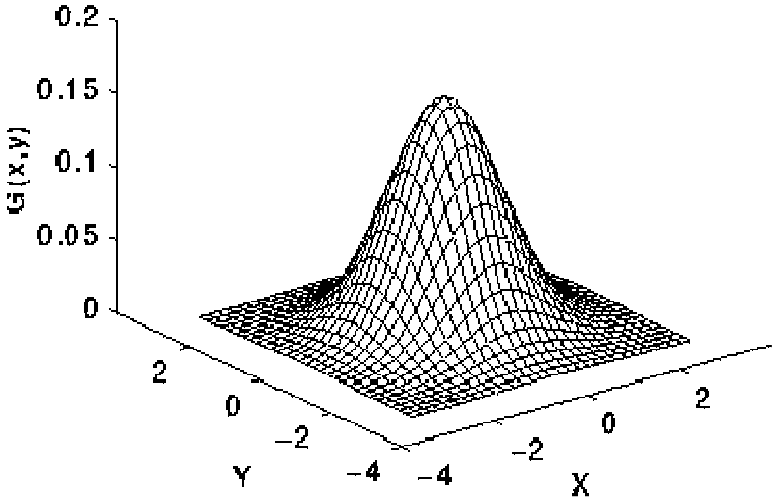
Filtro Gaussiano

- Filtro linear cujos coeficientes são determinados por uma **função gaussiana** com **desvio padrão σ**.

$$g_\sigma(x) = k.e^{-\frac{x^2}{2\sigma^2}}$$

$$g_\sigma(x, y) = k.e^{-\frac{x^2+y^2}{2\sigma^2}}$$





Operações de suavização

Aproximação discreta do Gaussiano 2D

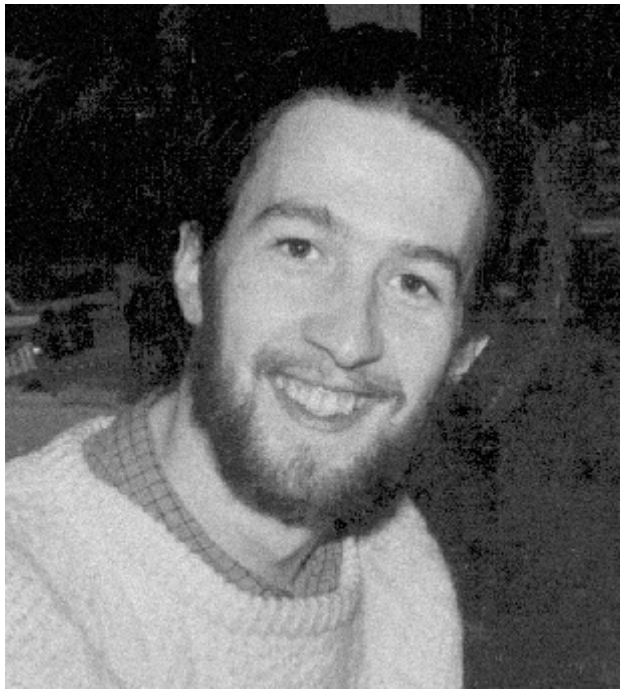
- **Eficaz** na remoção de ruído que siga uma distribuição gaussiana (**ruído gaussiano**).
- Atribui **maior peso aos píxeis mais próximos** do píxel central do que àqueles que se encontram na periferia da vizinhança → **preserva melhor as arestas** da imagem.
- O **grau de suavização** é controlado pelo valor do **desvio padrão**: quanto maior o desvio padrão, maior a largura do filtro e maior o seu poder de suavização.
- **Filtro separável**: pode ser implementado através de duas convoluções ortogonais com um filtro 1D → **eficiente** mesmo com filtros de grandes dimensões.

$$G_\sigma = \frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix} = \frac{1}{273}\begin{bmatrix} 1 \\ 4 \\ 6{,}35 \\ 4 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 4 & 6{,}35 & 4 & 1 \end{bmatrix}$$

Coeficientes do filtro gaussiano 5x5 com desvio padrão igual a 1

Operações de suavização

Exemplo: Ruído gaussiano



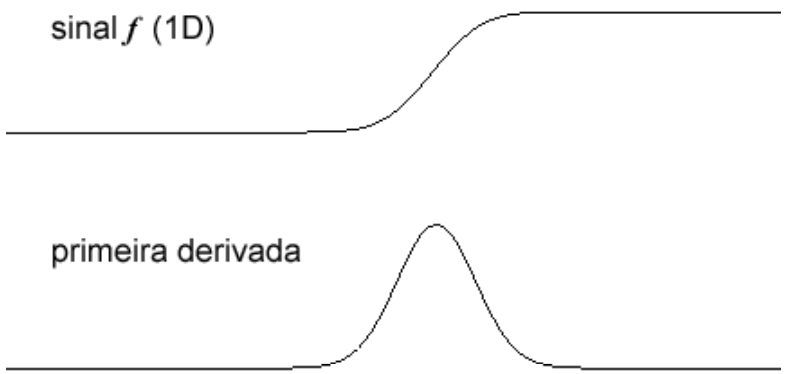
Filtro gaussiano 5x5
($\sigma = 1$)

Operações de detecção de arestas

Arestas ou orlas (*edges*):

Zonas onde ocorrem **fortes variações** nos valores da vizinhança de um píxel.

Uma aresta está associada a um **máximo local** na **primeira derivada** do sinal da imagem.



A detecção de arestas **realça as zonas** da imagem onde ocorrem essas variações significativas.

Gradiente

O gradiente é o **equivalente 2D da primeira derivada** do sinal, sendo definido pelo **vector**:

$$\nabla I(x,y) = \begin{bmatrix} \nabla_x \\ \nabla_y \end{bmatrix} = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix}$$

→ **Gradiente** segundo a direcção x

→ **Gradiente** segundo a direcção y

Operações de detecção de arestas

Propriedades do Gradiente:

1. A **magnitude** do gradiente é igual à taxa de variação máxima de $I(x,y)$ por unidade de distância e segundo a direcção de ∇.

$$\nabla_m[I(x,y)] = mag[\nabla] = \sqrt{\nabla_x^2 + \nabla_y^2} \approx |\nabla_x| + |\nabla_y|$$

2. O vector gradiente aponta na **direcção** segundo a qual se dá a maior variação do sinal $I(x,y)$.

$$\nabla_\theta[I(x,y)] = \theta(x,y) = \tan^{-1}\left(\frac{\nabla_y}{\nabla_x}\right)$$

3. A direcção do gradiente é **perpendicular** à direcção da **aresta**.

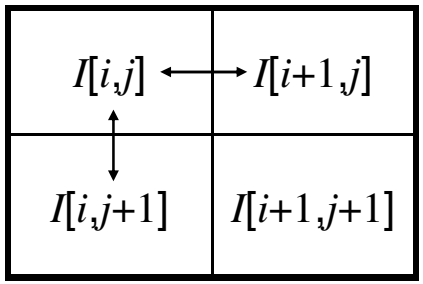
Operações de detecção de arestas

Aproximação numérica do Gradiente

Em PDI, as derivadas são aproximadas por **diferenças**.

$$\frac{\partial I}{\partial x} \equiv \nabla_x \approx I[i+1, j] - I[i, j]$$

$$\frac{\partial I}{\partial y} \equiv \nabla_y \approx I[i, j+1] - I[i, j]$$

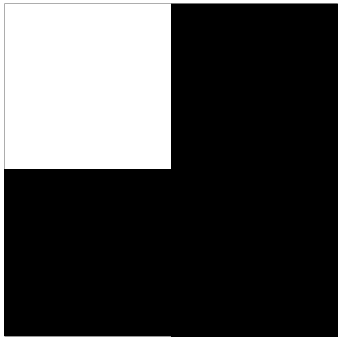


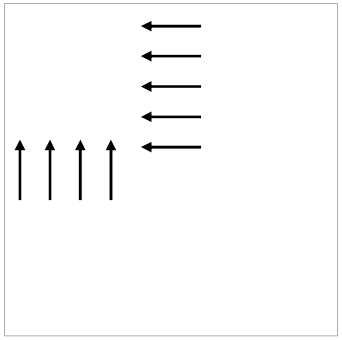
$\nabla_x =$ | −1 | 1 |

Máscara para detecção de arestas verticais

$\nabla_y =$ | −1 | / | 1 |

Máscara para a detecção de arestas horizontais



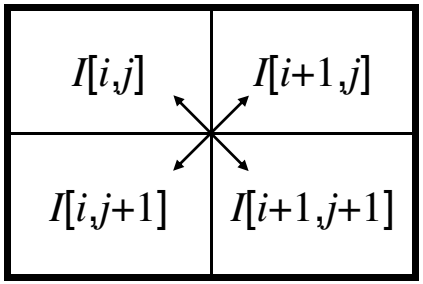


Operações de detecção de arestas

Operadores de Roberts

Operadores não orientados segundo os eixos x e y, mas funcionalmente similares aos anteriores.

$$\nabla_1 \approx I[i, j] - I[i+1, j+1]$$
$$\nabla_2 \approx I[i+1, j] - I[i, j+1]$$



$$\nabla_1 = \begin{array}{|c|c|}\hline 1 & 0 \\ \hline 0 & -1 \\ \hline \end{array} \qquad \nabla_2 = \begin{array}{|c|c|}\hline 0 & 1 \\ \hline -1 & 0 \\ \hline \end{array}$$

Estas máscaras geram valores centrados em **pontos interpolados** ([i+½, j] e [i, j+½], ou [i+½, j+½], respectivamente). Para obter valores centrados em [i, j], é frequente usar pares de máscaras com um número ímpar de elementos e simétricas em relação a [i, j]:

$$\nabla_x \approx I[i+1, j] - I[i-1, j]$$
$$\nabla_y \approx I[i, j+1] - I[i, j-1]$$

$$\nabla_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline \end{array}$$

Máscara para detecção de arestas verticais

$$\nabla_y = \begin{array}{|c|}\hline -1 \\ \hline 0 \\ \hline 1 \\ \hline \end{array}$$

Máscara para a detecção de arestas horizontais

Operações de detecção de arestas

É comum o uso de máscaras 3x3, por exemplo:

Operadores de Prewitt

–1	0	1
–1	0	1
–1	0	1

∇_x

–1	–1	–1
0	0	0
1	1	1

∇_y

Operadores de Sobel

–1	0	1
–2	0	2
–1	0	1

∇_x

–1	–2	–1
0	0	0
1	2	1

∇_y

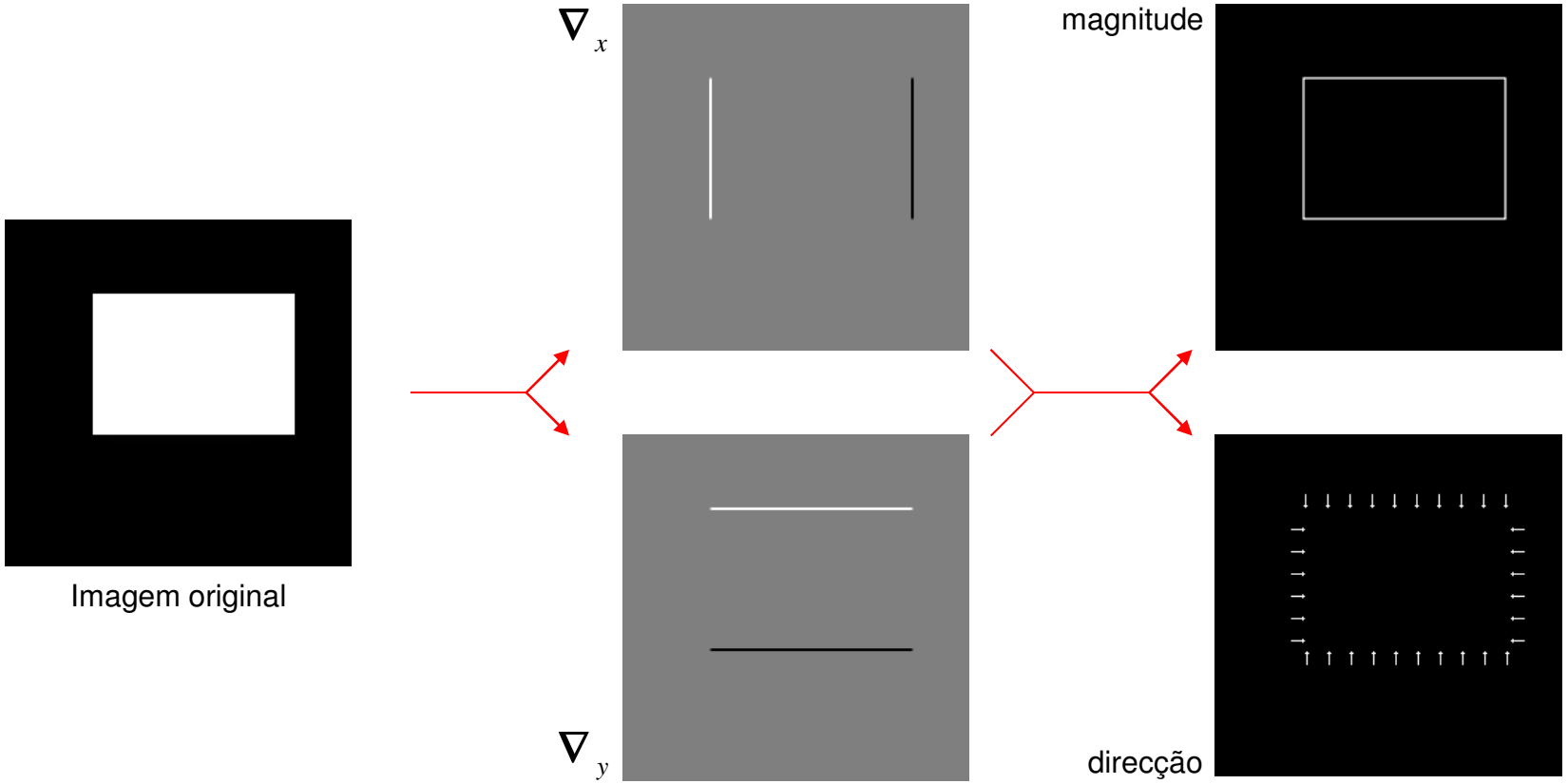
Ou até maiores:

–3	–4	0	4	3
–4	–6	0	6	4
–6	–12	0	12	6
–4	–6	0	6	4
–3	–4	0	4	3

–3	–4	–6	–4	–3
–4	–6	–12	–6	–4
0	0	0	0	0
4	6	12	6	4
3	4	6	4	3

Operações de detecção de arestas

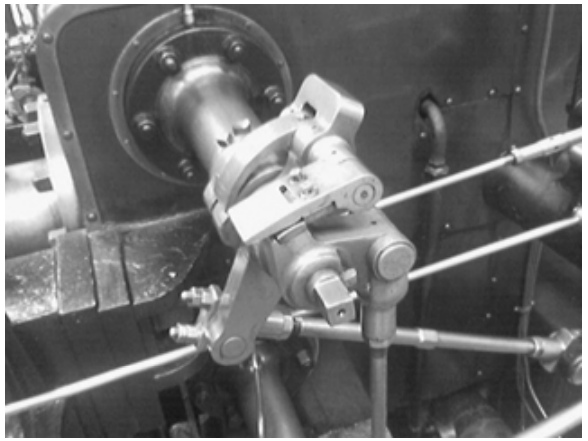
Princípio da detecção de arestas

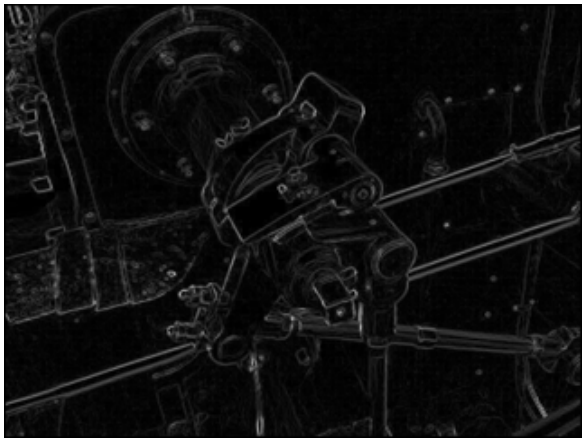


(operadores de Sobel)

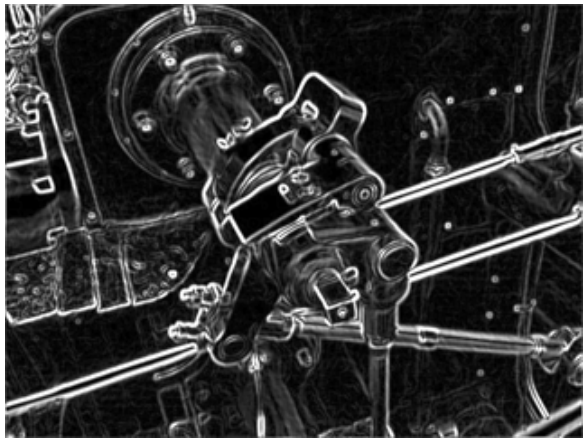
Operações de detecção de arestas

Imagem original





Roberts (magnitude do gradiente)



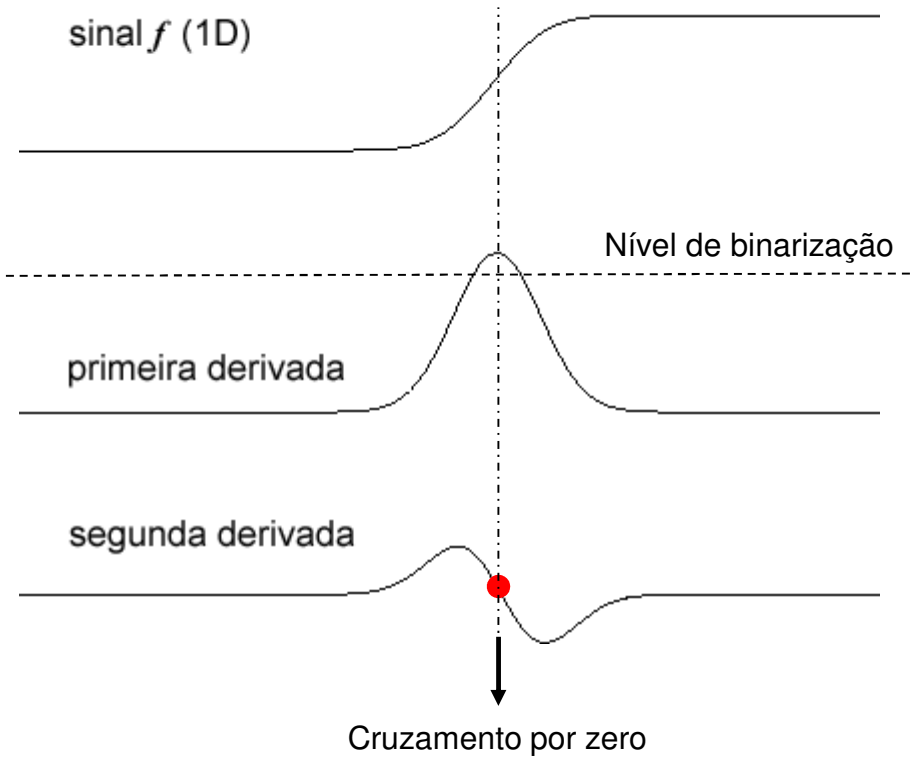
Sobel (magnitude do gradiente)

Operações de detecção de arestas

Operadores de segunda derivada

Problema: Usando detectores de arestas baseados no cálculo da 1.ª derivada do sinal, sempre que o valor resultante da filtragem estiver acima de um limiar, considera-se estar-se na presença de uma aresta — daqui resulta um **elevado número de píxeis pertencentes à aresta**.

Solução: Encontrar máximos locais do gradiente (pontos com valores elevados na 1.ª derivada e com um **cruzamento por zero na 2.ª derivada**); as arestas corresponderão aos pontos onde estes cruzamentos por zero ocorrem.



Operações de detecção de arestas

Laplaciano

O Laplaciano é o **equivalente 2D da segunda derivada** do sinal:

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

As segundas derivadas segundo x e y podem ser **aproximadas** usando as equações das diferenças:

$$\begin{aligned}\frac{\partial^2 I}{\partial x^2} = \frac{\partial \nabla_x}{\partial x} &= \frac{\partial (I[i+1, j] - I[i, j])}{\partial x} = \frac{\partial I[i+1, j]}{\partial x} - \frac{\partial I[i, j]}{\partial x} \\ &\approx (I[i+2, j] - I[i+1, j]) - (I[i, j+1] - I[i, j]) \\ &= I[i+2, j] - 2I[i+1, j] + I[i, j]\end{aligned}$$

Operações de detecção de arestas

A aproximação anterior está centrada no píxel $[i+1, j]$. Substituindo i por $i-1$ (e trocando os sinais*):

$$\frac{\partial^2 I}{\partial x^2} = -I[i+1, j] + 2I[i, j] - I[i-1, j]$$

Analogamente para y:

$$\frac{\partial^2 I}{\partial y^2} = -I[i, j+1] + 2I[i, j] - I[i, j-1]$$

Combinando as duas equações num único operador, obtém-se a seguinte máscara, usada na **aproximação do Laplaciano** (3x3):

∇^2 :

0	–1	0
–1	4	–1
0	–1	0

* O que quer dizer que, em todo o rigor, trabalhamos com o ***negativo*** do Laplaciano matemático.

Operações de detecção de arestas

Outras possíveis aproximações do Laplaciano 3x3:

0	–4	0
–4	20	–4
0	–4	0

1	–2	1
–2	4	–2
1	–2	1

–1	–1	–1
–1	8	–1
–1	–1	–1

O Laplaciano assinala a **presença de uma aresta** quando na sua saída se verifica uma transição por zero (***zero crossing***):

0	0	0	6	–6	0	0	0

zero crossing

Operações de detecção de arestas

Laplaciano do Gaussiano (LoG)

Problema: A **segunda derivada** das intensidades da imagem é **muito sensível ao ruído**.

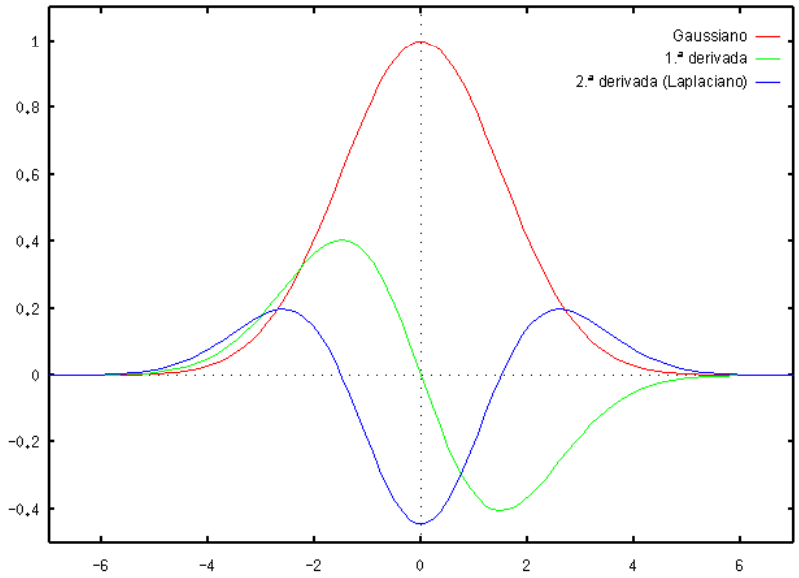
Solução: Primeiro **suavizar** a imagem e só depois **realçar** as arestas — Operador LoG.

$$I_{out}(x,y) = \nabla^2 [G_\sigma(x,y) * I_{in}(x,y)]$$

$$= [\nabla^2 G_\sigma(x,y)] * I_{in}(x,y)$$

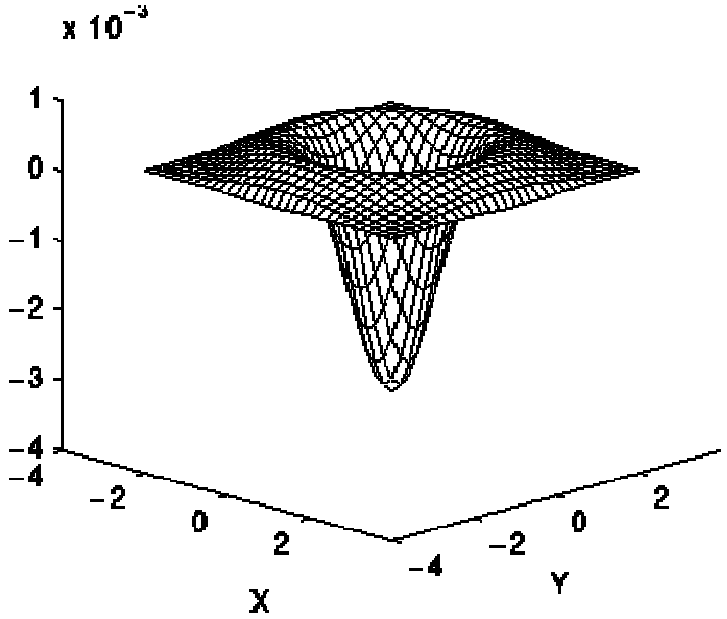
$$= \left[\frac{1}{\sqrt{2\pi\sigma^2}} \frac{x^2+y^2-2\sigma^2}{\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}} \right] * I_{in}(x,y)$$

2.ª derivada do Gaussiano
(Laplaciano do Gaussiano)



Gaussiano, 1.ª derivada e 2.ª derivada (funções 1D)

Operações de detecção de arestas



Laplaciano do Gaussiano (2D)

$LoG \approx$

0	0	–1	0	0
0	–1	–2	–1	0
–1	–2	16	–2	–1
0	–1	–2	–1	0
0	0	–1	0	0

aproximação do LoG 5x5
(após inversão dos sinais)

O operador LoG é também conhecido como operador de **Marr-Hildreth** ou, mais familiarmente, como **"chapéu mexicano"** (referência à representação 3D do operador 2D).

Operações de detecção de arestas

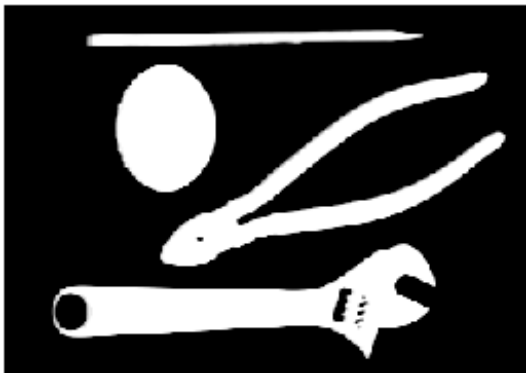
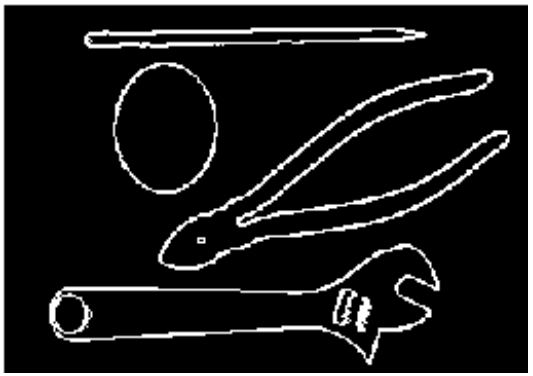


Imagem de entrada



Zero crossings do resultado do LoG

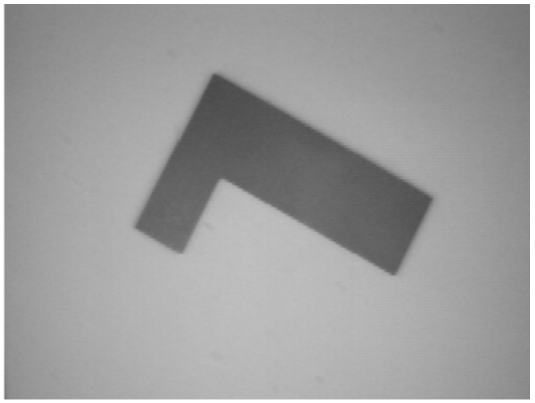
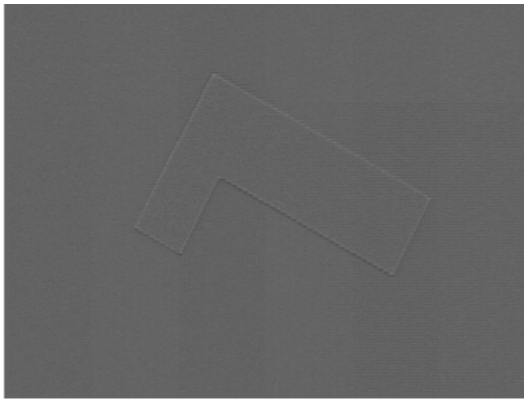
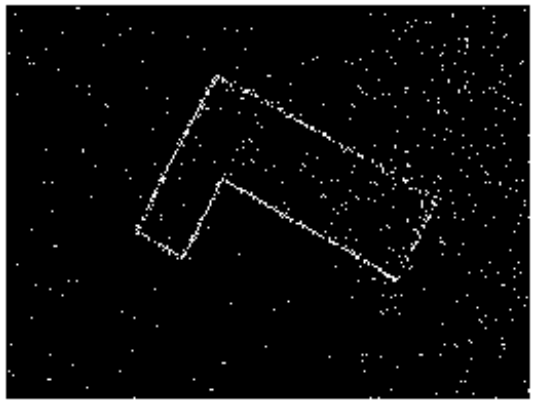


Imagem de entrada



Resultado do LoG



Zero crossings do resultado do LoG validados pelo gradiente

Operações de detecção de arestas

Operador de Canny

O operador LoG tem duas limitações: não raro detecta **arestas "falsas"** e **erra na localização** de arestas verdadeiras que sejam curvas. Uma solução mais avançada é, p. ex., o algoritmo de Canny, que procede à **validação das arestas** com base em **máximos locais do gradiente**.

1. **Redução do ruído**: Filtragem com Gaussiano.
2. **Detecção de arestas**: Utilizar quatro filtros (**horizontal**, **vertical** e **diagonais**, por exemplo, Sobel + Roberts), aproximando as direcções a 0º, 45º, 90º e 135º.
3. **Detecção dos máximos locais**: Considerar estar perante um máximo local se o píxel actual tiver um gradiente com magnitude superior à dos seus vizinhos segundo a direcção perpendicular à do seu gradiente. (Por exemplo, se a direcção do seu gradiente é 0º, comparar a sua magnitude com a dos seus vizinhos nas direcções Norte e Sul.)
4. **Traçado das arestas** usando **binarização com histerese**: Considerar como arestas certas os máximos locais que tiverem um valor maior que um certo limiar (*threshold*) superior; a partir destas, aceitar também como arestas os píxeis conexos que, na direcção da aresta, têm gradiente com magnitude maior do que um certo limiar inferior.

Operações de detecção de arestas

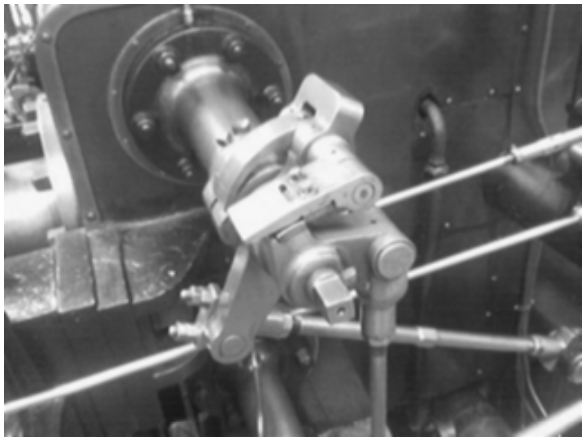
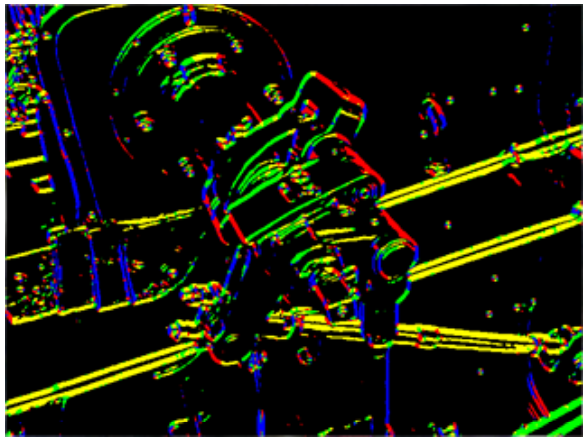
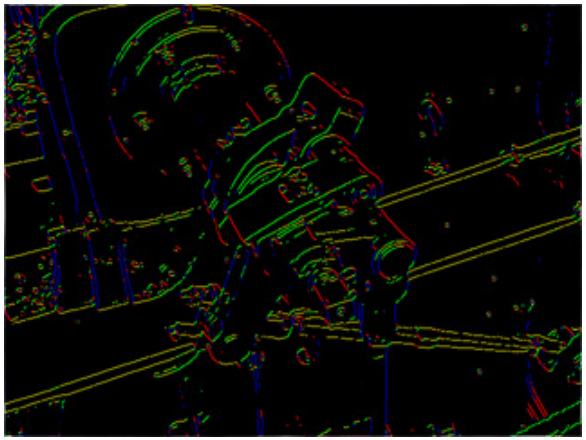


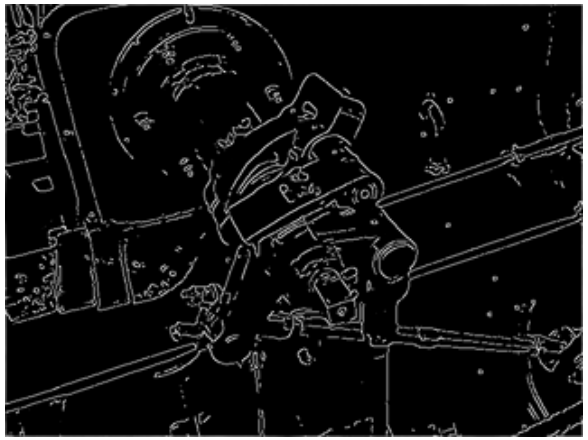
Imagem filtrada com Gaussiano



Arestas detectadas (cor indica direcção)

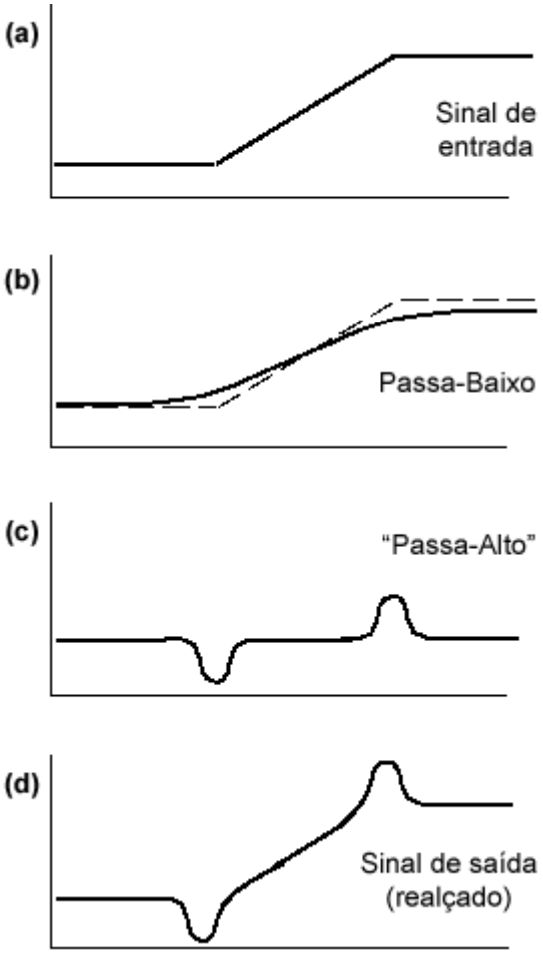


Arestas que são máximos locais



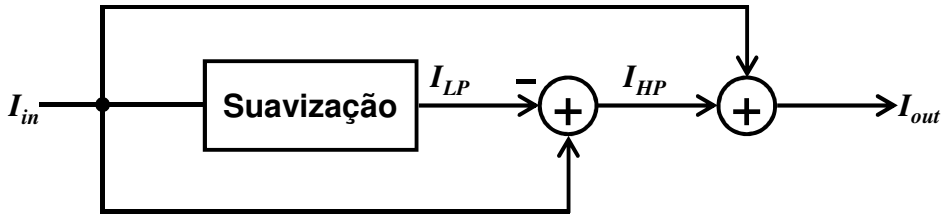
Resultado final do detector de Canny

Operações de realce da imagem



Unsharp masking

Os operadores de realce da imagem (***image sharpening***) têm como objectivo **realçar os detalhes finos** que estão esbatidos (*blurred*) na imagem original. Um exemplo é o *unsharp masking*, versão digital de uma velha técnica fotográfica.



(c) $I_{HP} = I_{in} - I_{LP}$

(d) $I_{out} = I_{in} + k.I_{HP}$

Operações de realce da imagem

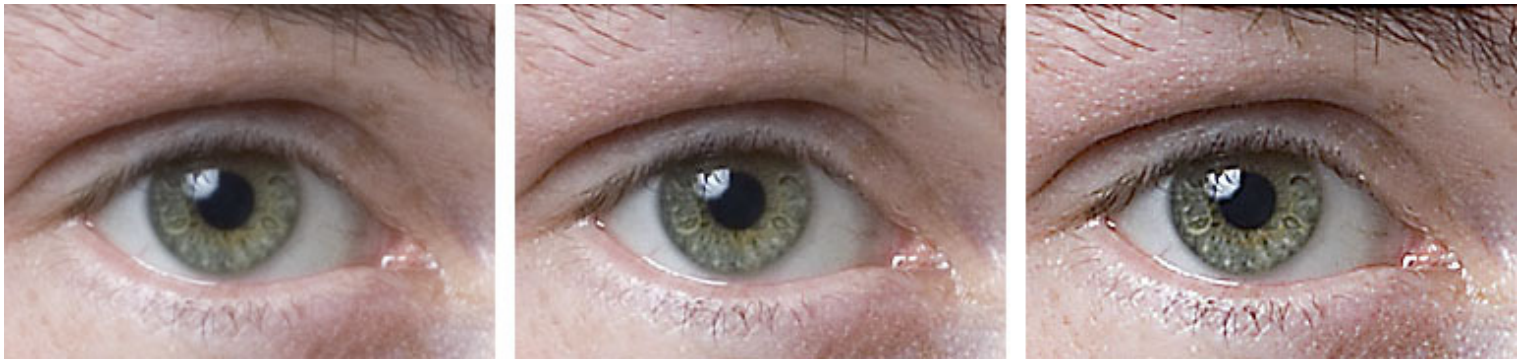
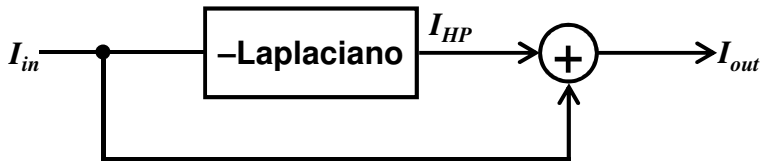


Imagem original e após *unsharp masking* (dois exemplos com graus de realce diferentes)

A metade inferior da espiral foi realçada por *unsharp masking*.

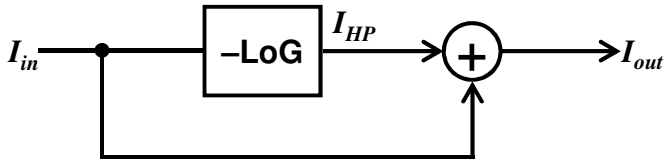
Operações de realce da imagem

Em PDI, uma maneira mais comum de implementar uma *"unsharp mask"* é usar o **Laplaciano*** para extrair a informação passa-alto num só passo:



$$I_{out} = I_{in} - k.\underbrace{\nabla^2 I_{in}}_{I_{HP}}$$

Para contornar o problema da **sensibilidade ao ruído** de que sofre o Laplaciano, uma alternativa é substituí-lo pelo operador **LoG***:



$$I_{out} = I_{in} - k.\underbrace{\nabla^2 G_\sigma * I_{in}}_{I_{HP}}$$

* De facto, conforme visto antes, usamos os *negativos* destes operadores.

Operações de realce da imagem

Imagem original

Unsharp masking com Laplaciano

Unsharp masking com LoG