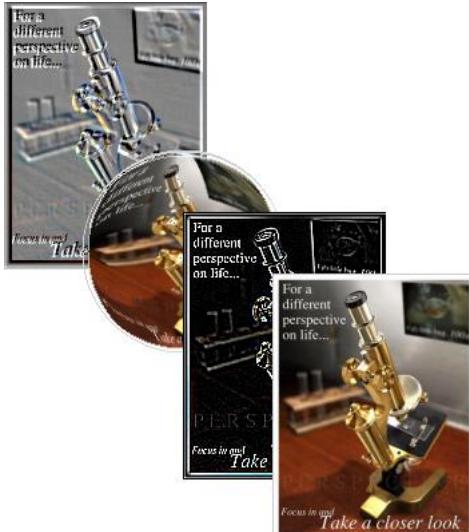


# Filtragem

Carlos Alexandre Barros de Mello  
CIn/UFPE



# Filtragem

- Melhoria nas Imagens
  - Mudança nas características das imagens
- Efeitos digitais

# Filtragem

- Domínio da Frequência
- Domínio Espacial
  - Ambos os tipos podem depender da aplicação de **convoluçãoes** para executarem a filtragem

# Filtragem

- A aplicação da filtragem em um domínio ou no outro depende, geralmente, do tipo de problema que se quer solucionar
- Em geral, qualquer um dos dois domínios pode ser usado (o espacial é mais “natural”), mas há situações onde um prevalece sobre o outro..

# Filtragem

- Exemplos de filtragem:



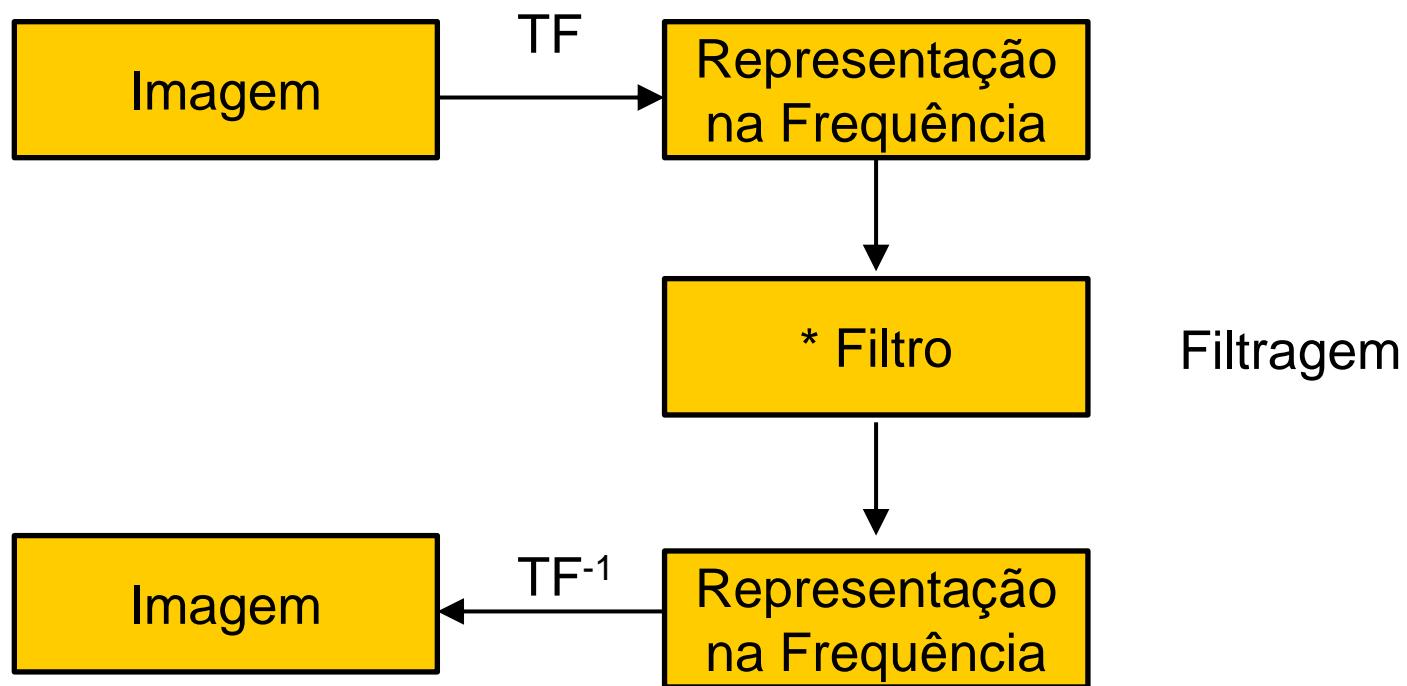
# Filtragem

- Exemplos de filtragem:



# Filtragem no Domínio da Frequência

- Uso da Transformada de Fourier



# Filtragem no Domínio da Frequência

- Uso da Transformada de Fourier
  - A Transformada de Fourier leva um sinal do domínio do tempo para o domínio da frequência
    - No caso de imagens, o domínio de origem é o espacial
  - Há uma transformação inversa que devolve o sinal ao domínio do tempo
  - DFT, FFT

# Filtragem no Domínio da Frequência

- Uso da DFT Bi-Dimensional

$$F(p, q) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) e^{-j(2\pi/M)pm} e^{-j(2\pi/N)qn}$$

$$f(m, n) = \frac{1}{MN} \sum_{p=0}^{M-1} \sum_{q=0}^{N-1} F(p, q) e^{j(2\pi/M)pm} e^{j(2\pi/N)qn}$$

# Filtragem no Domínio da Frequência

- Espectro de Fourier (Magnitude):

$$|F(u)| = [R^2(u) + I^2(u)]^{1/2}$$

- Fase:  $\phi(u) = \tan^{-1} \left[ \frac{I(u)}{R(u)} \right]$

- Espectro de Potência:

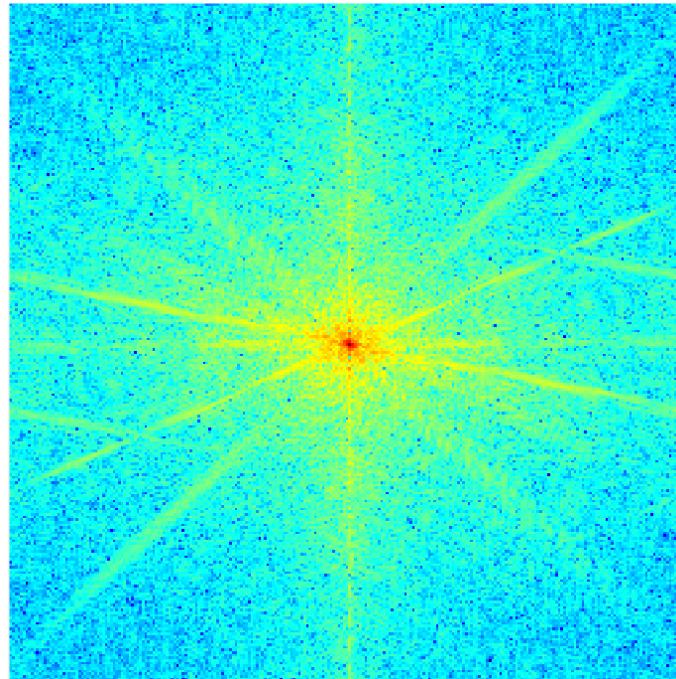
$$P(u) = |F(u)|^2 = R^2(u) + I^2(u)$$

# Filtragem no Domínio da Frequência

```
function img_fourier (nome, ext)
    nome_in = [nome '.' ext];
    im = imread(nome_in);
    figure, imshow (im);
    F = fft2(im);
    % Just for visualization
    F2 = fftshift(F);
    F2_2 = uint8(rescale (log(abs(F2)), 0, 255));
    figure, imshow(F2_2, colormap (jet(256)));
    figure, imshow(angle(F2), []); % Phase
```

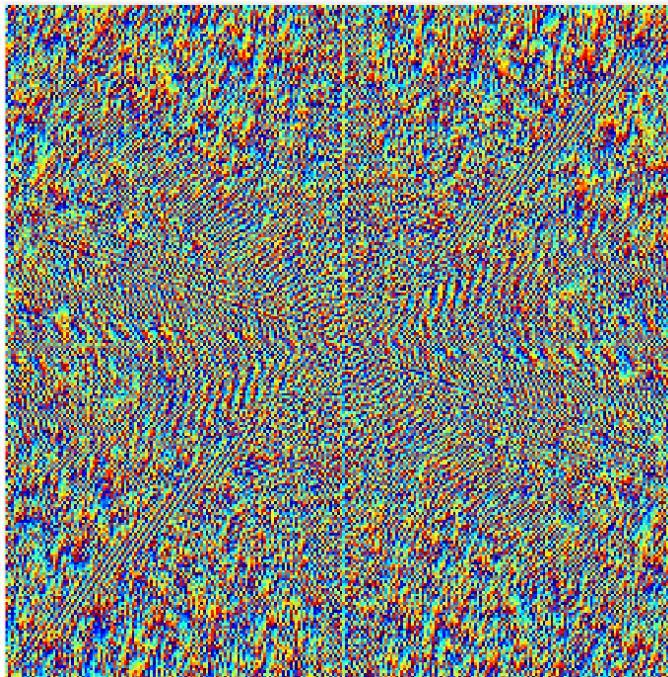


# Filtragem no Domínio da Frequência



Magnitude

# Filtragem no Domínio da Frequência



**Fase**

# Filtragem no Domínio da Frequência

Mudanças na imagem: consequências na transformada



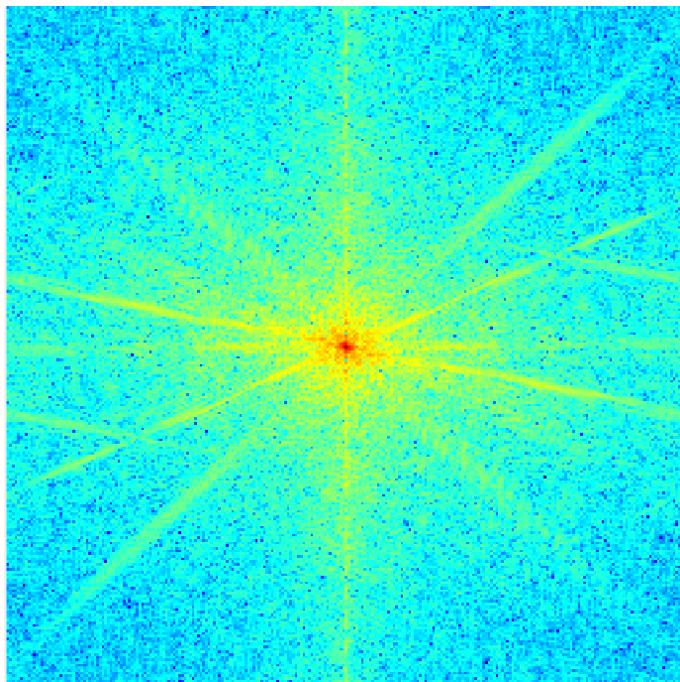
Imagen original



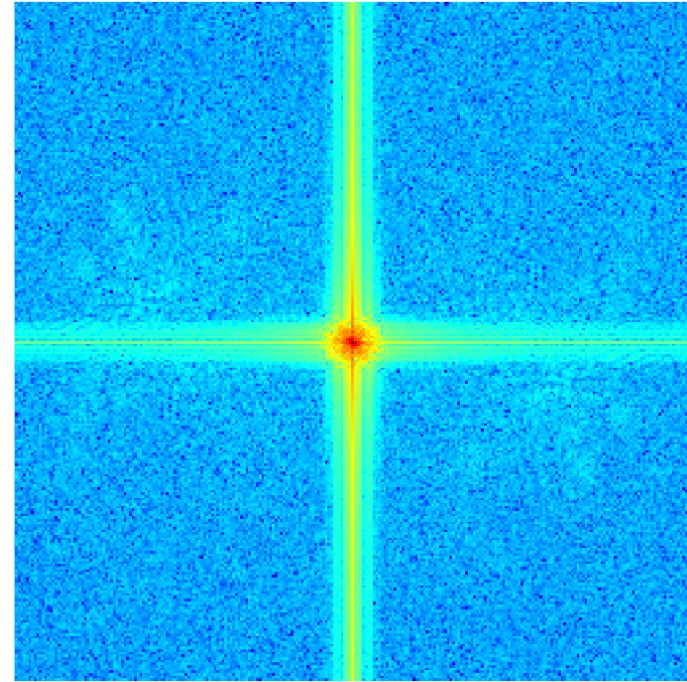
Imagen embaçada

# Filtragem no Domínio da Frequência

Mudanças na imagem: consequências na transformada



$F(\text{Imagem original})$

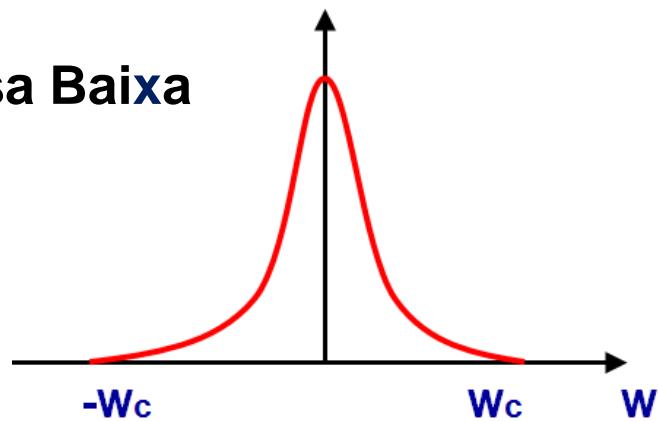


$F(\text{Imagem embaçada})$

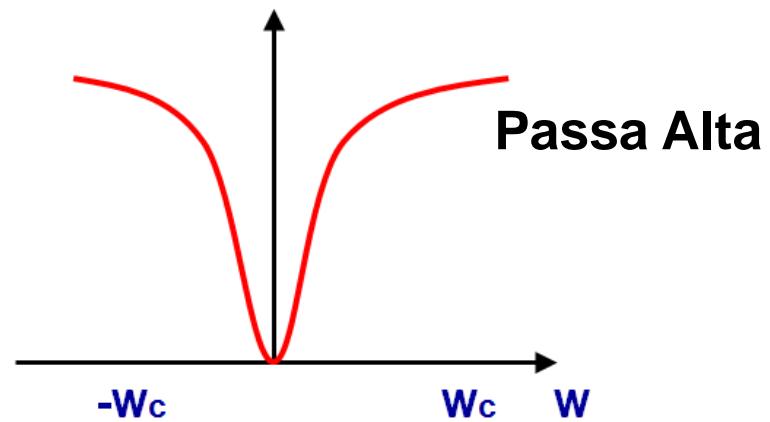
# Filtragem no Domínio da Frequência

- Filtros

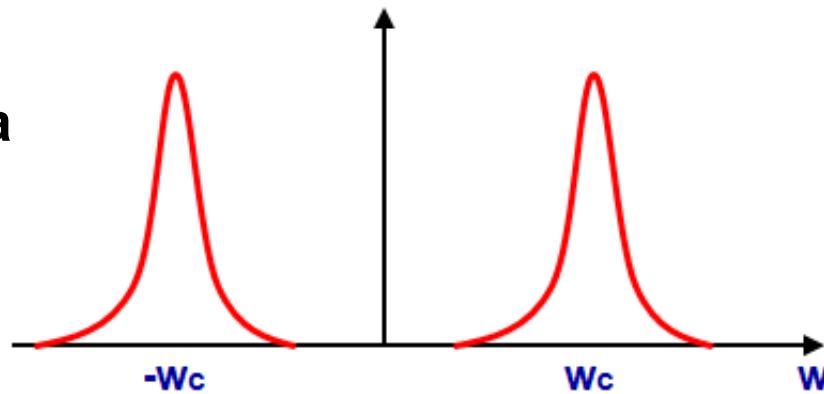
Passa Baixa



Passa Alta

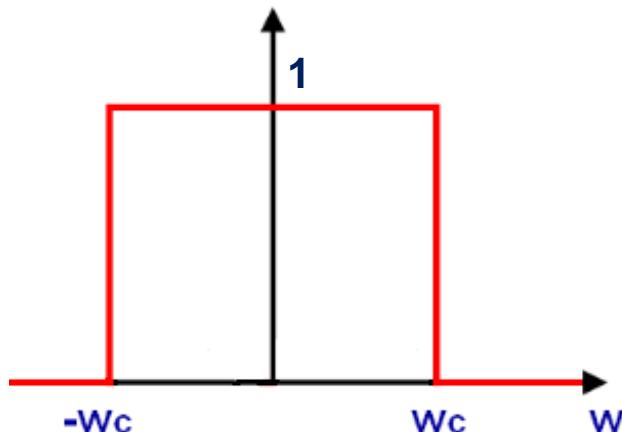


Passa Faixa



# Filtragem no Domínio da Frequência

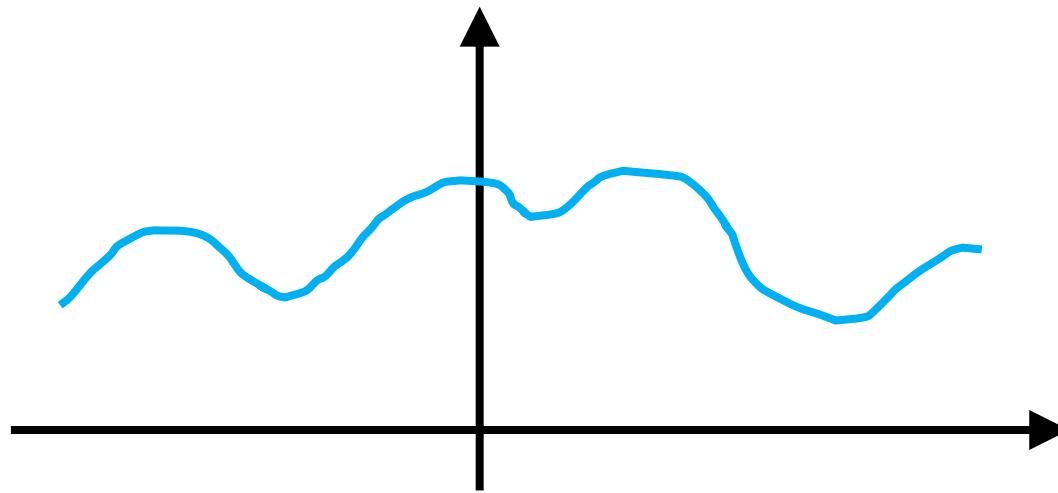
- Mas o que são filtros e qual é sua função?
  - Exemplo: Filtro passa baixa ideal



$$h[n] = \begin{cases} 1, & -w_c \leq w \leq w_c \\ 0, & \text{caso contrário} \end{cases}$$

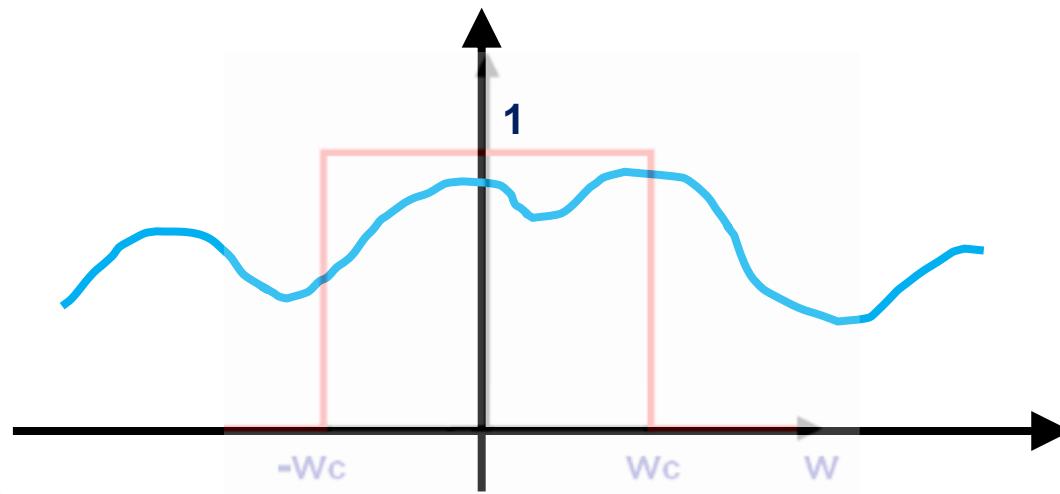
# Filtragem no Domínio da Frequência

- Mas o que são filtros e qual é sua função?
  - Qual o resultado da multiplicação do filtro anterior pelo *gráfico* abaixo?



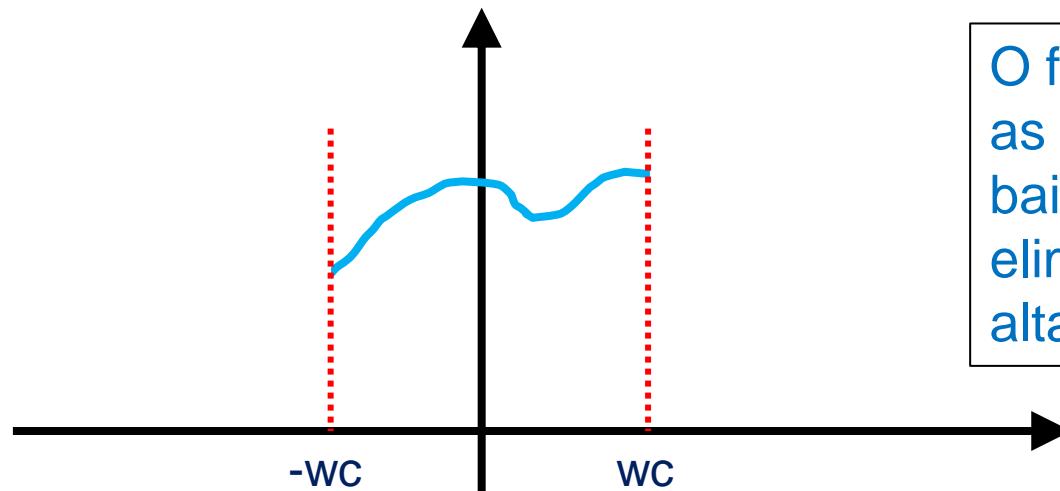
# Filtragem no Domínio da Frequência

- Mas o que são filtros e qual é sua função?
  - Qual o resultado da multiplicação do filtro anterior pelo gráfico abaixo?



# Filtragem no Domínio da Frequência

- Mas o que são filtros e qual é sua função?
  - Qual o resultado da multiplicação do filtro anterior pelo gráfico abaixo?



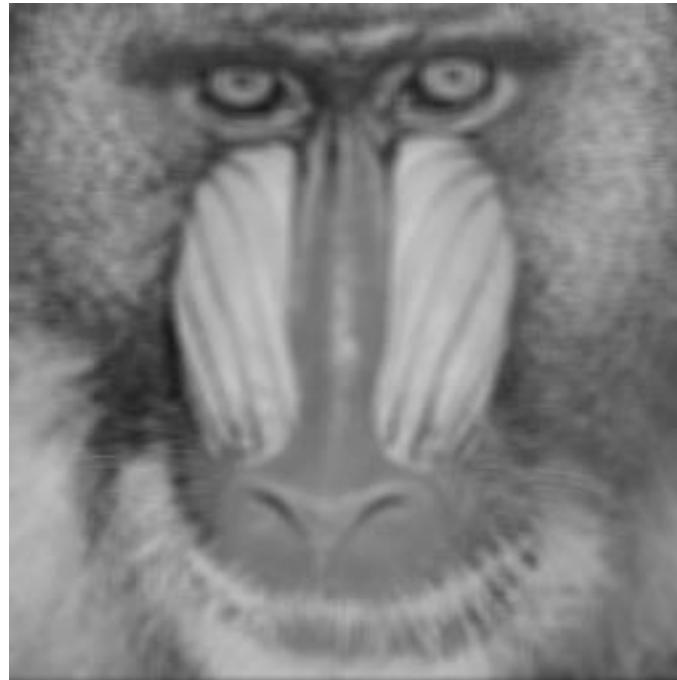
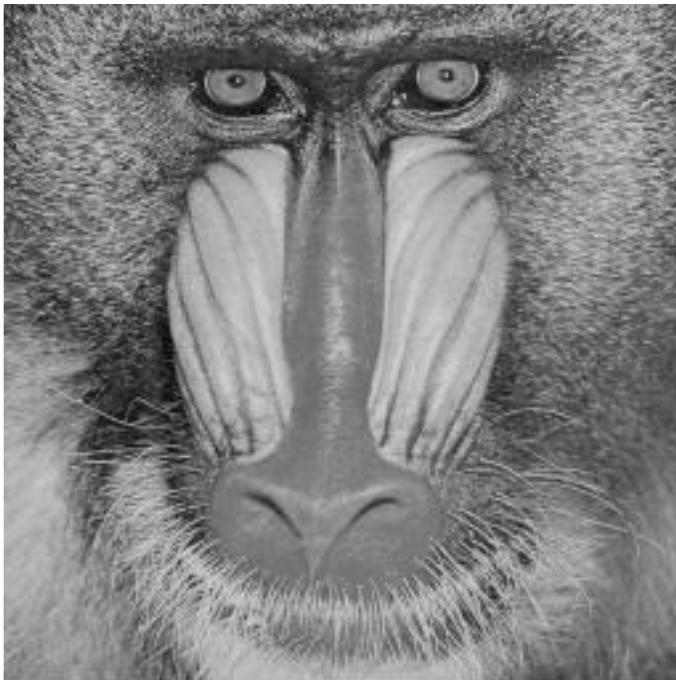
O filtro preserva as regiões de baixa frequência, eliminando as altas frequências.

# Filtragem no Domínio da Frequência

- Observações importantes:
  - A multiplicação (que foi feita antes) em um domínio (tempo ou frequência) traz consequências para o outro domínio
  - O que representa *frequência* em imagens?
    - Isso equivale a entender o que se espera da aplicação de um filtro passa baixa, ou passa alta, ou qualquer outro, em uma imagem

# Filtragem no Domínio da Frequência

- Exemplo de aplicação de *um filtro passa baixa* em uma imagem:



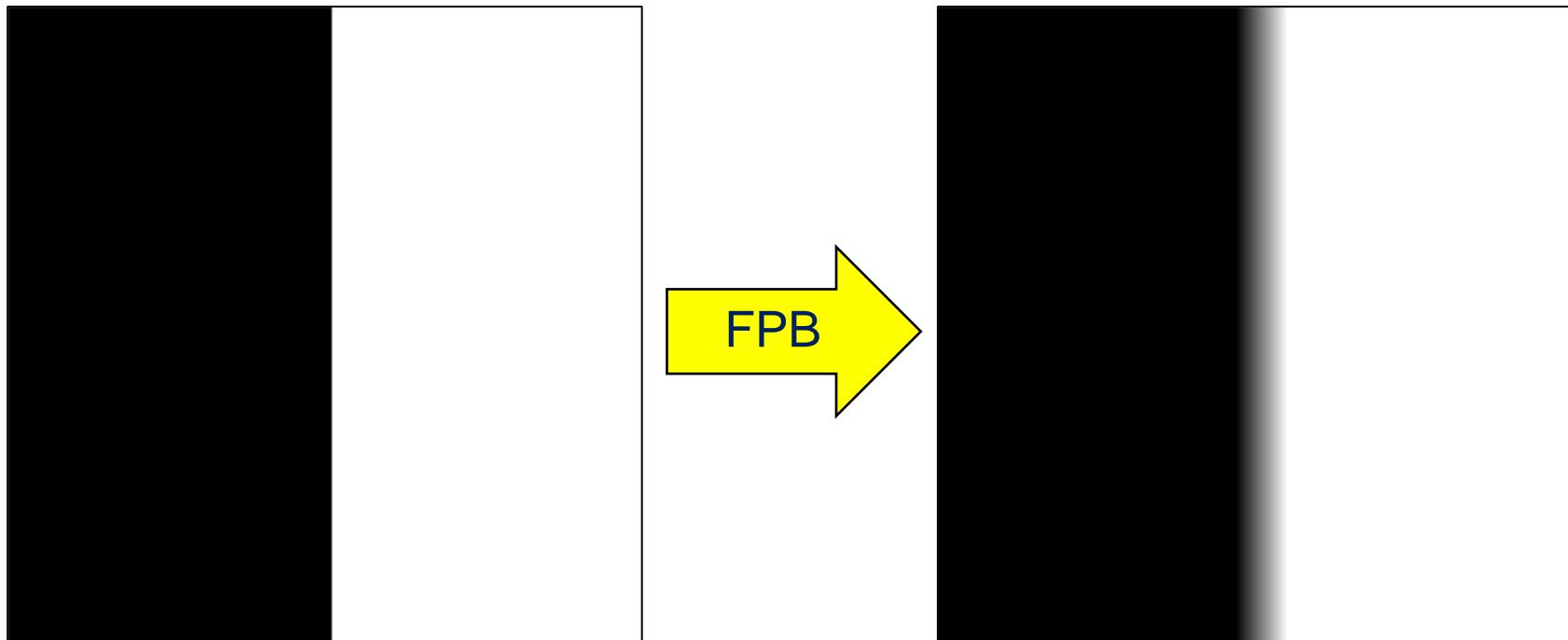
# Filtragem no Domínio da Frequência

---

- Um filtro passa baixa atenua as altas frequências; na imagem anterior, ele provocou um *borramento* na imagem
- Isso significa que as componentes alteradas diminuíram a percepção da separação entre os elementos da imagem

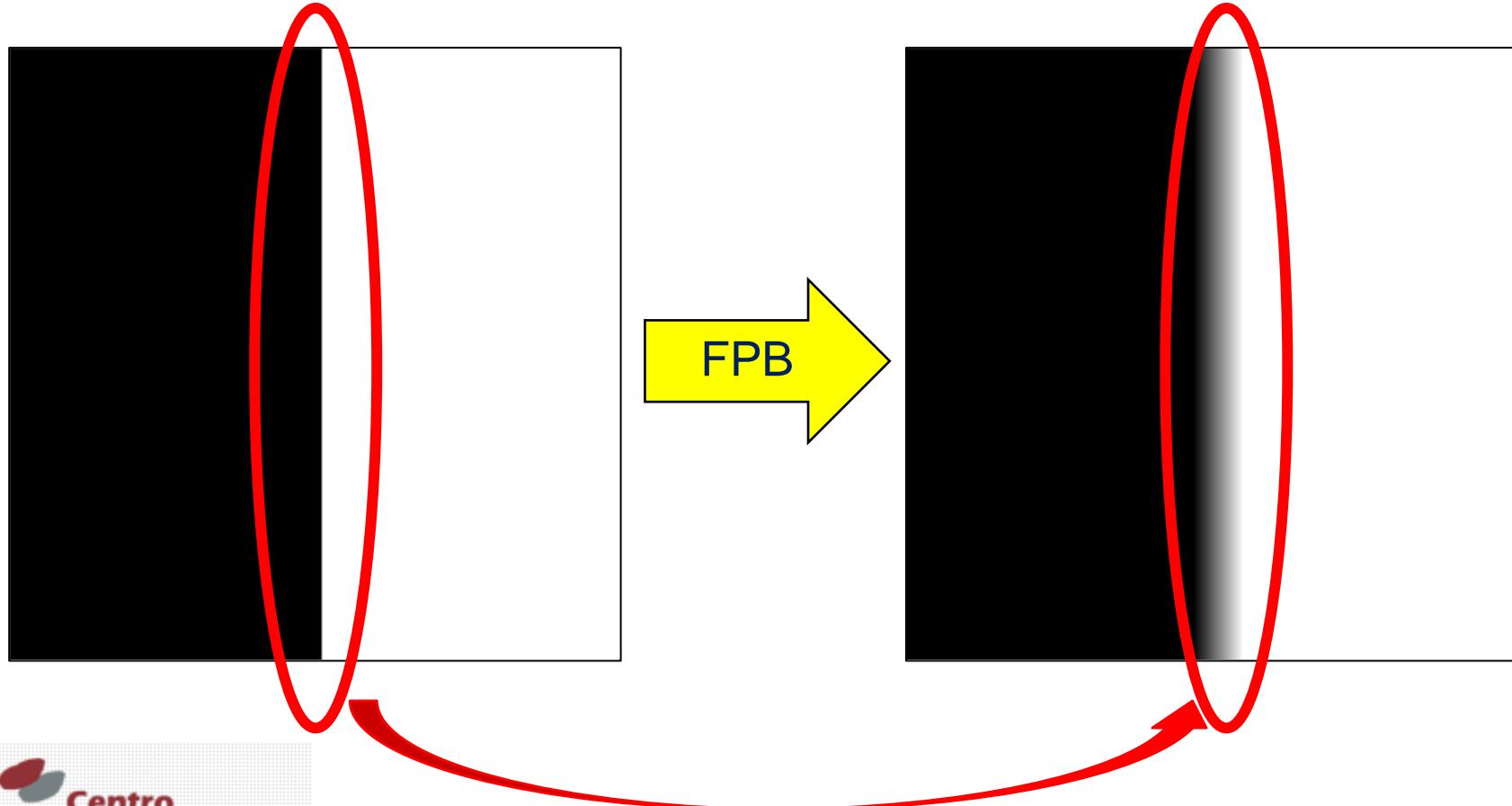
# Filtragem no Domínio da Frequência

- Filtro Passa Baixa



# Filtragem no Domínio da Frequência

- Filtro Passa Baixa



# Filtragem no Domínio da Frequência

- Filtros
  - Filtros passa-baixa (suavização, borramento)
    - Preserva as baixas frequências
    - Atenua as altas frequências
  - Filtros passa-alta (realce das bordas, aguçamento)
    - Preserva as altas frequências
    - Atenua as baixas frequências
  - Filtros passa-faixa (restauração de imagens)
    - Preserva frequências específicas
    - Atenua outras frequências
  - Baixa frequências: área de suavização (mudanças suaves entre tons)
  - Altas frequências: detalhes, como bordas e ruídos (mudanças bruscas entre tons)

# Filtragem no Domínio da Frequência

- Como a filtragem ocorre: *Convolução*
- Dadas duas funções  $f_1(t)$  e  $f_2(t)$ , define-se a convolução dessas funções como:

$$f_1(t) * f_2(t) = \int_{-\infty}^{\infty} f_1(\tau) f_2(t - \tau) d\tau$$

- Teorema da Convolução:
  - Se  $f_1(t) \leftrightarrow F_1(\omega)$  e  $f_2(t) \leftrightarrow F_2(\omega)$ , então:

$$f_1(t) * f_2(t) \leftrightarrow F_1(\omega) \cdot F_2(\omega)$$

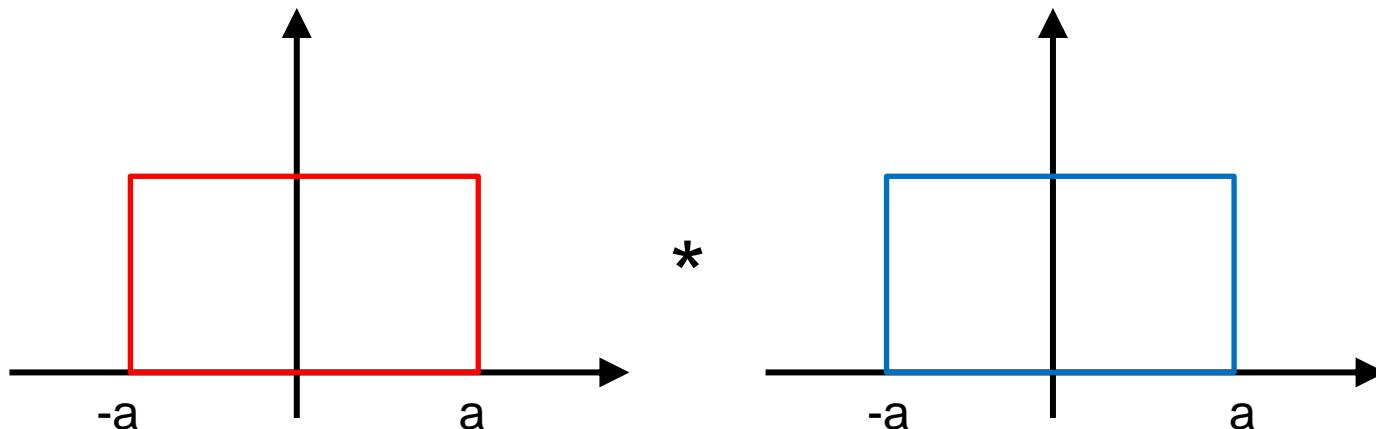
Importante!

$$f_1(t) \cdot f_2(t) \leftrightarrow (1/2\pi)[F_1(\omega)^* F_2(\omega)]$$

e

# Filtragem no Domínio da Frequência

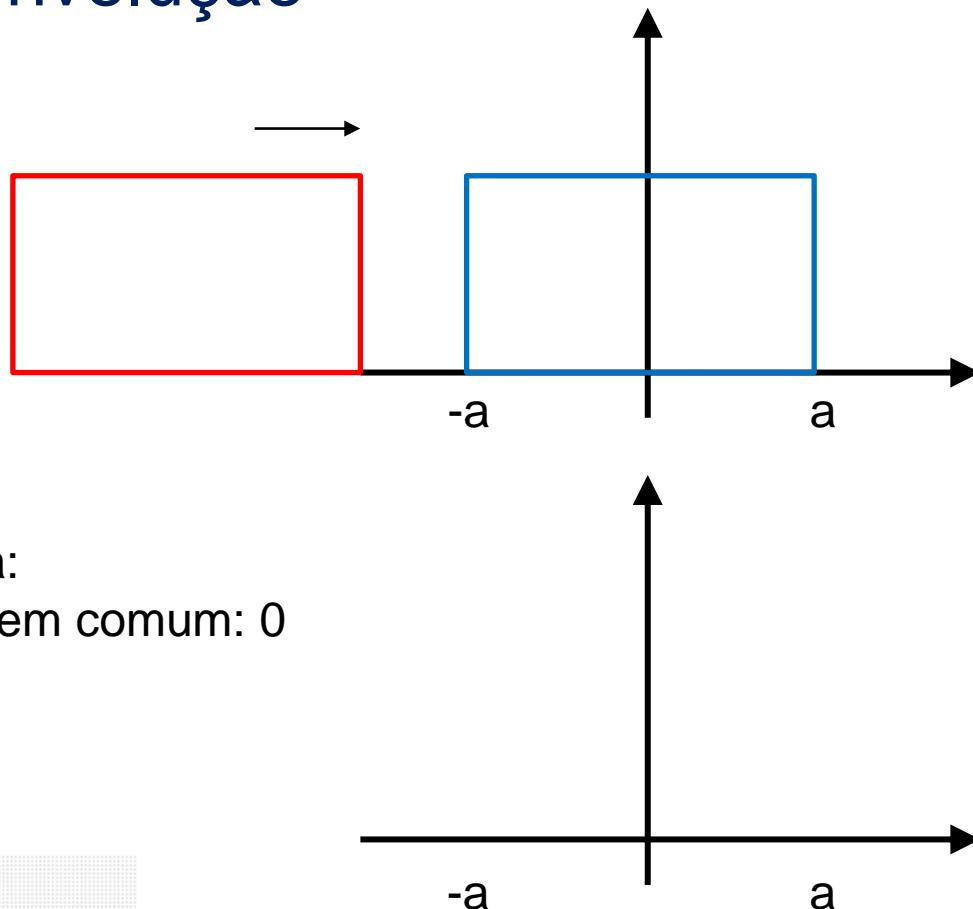
- Convolução
  - Exemplo: Convolução de duas funções “porta” iguais



Processo: Uma função fica fixa, a outra inverte (o que altera nada nesse caso) e se desloca através da outra. O resultado da convolução (a integral) é a área em comum. Observe o que acontece nesse caso....

# Filtragem no Domínio da Frequência

- Convolução

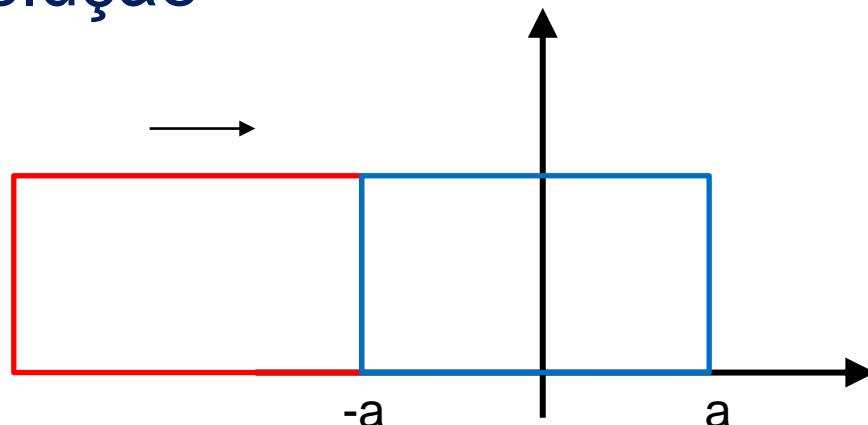


Saída:

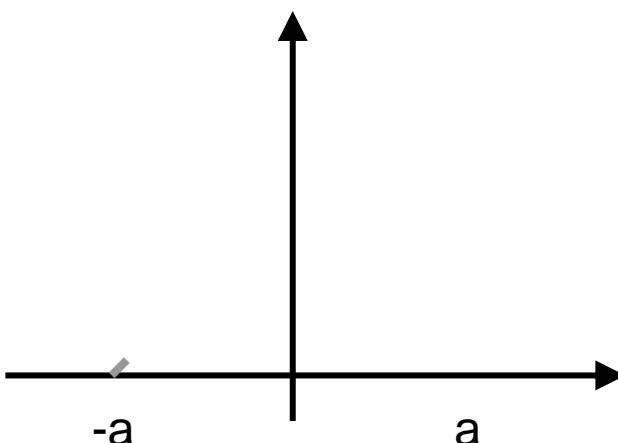
Área em comum: 0

# Filtragem no Domínio da Frequência

- Convolução

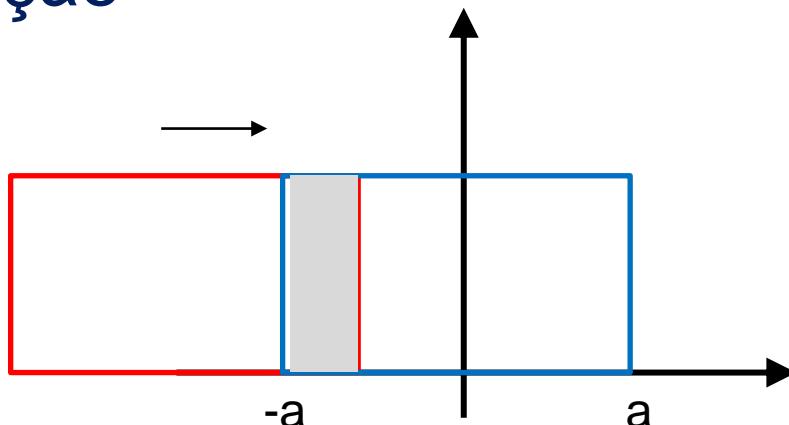


Começa a ter  
área em comum....

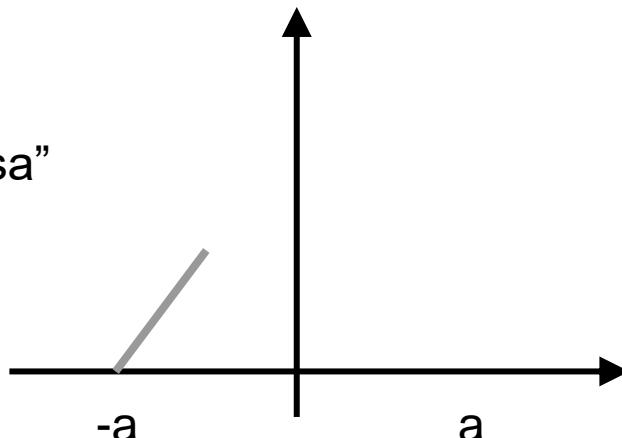


# Filtragem no Domínio da Frequência

- Convolução

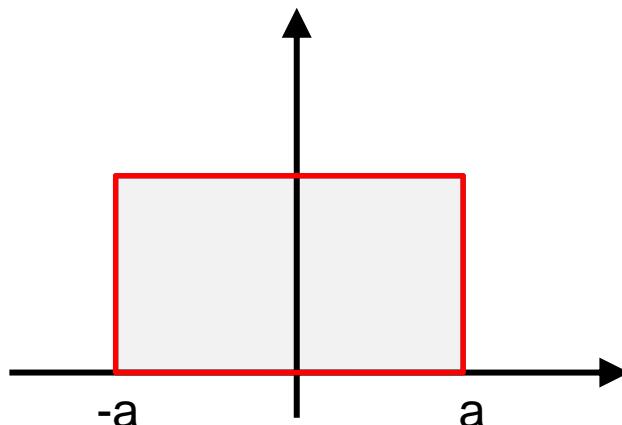


A área em comum vai crescendo, à medida que uma função “passa” pela outra....

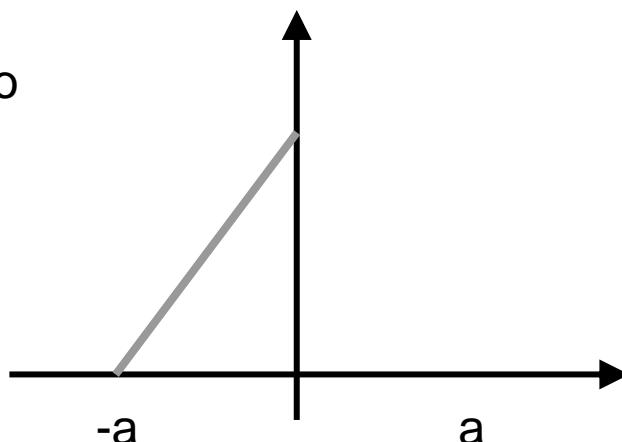


# Filtragem no Domínio da Frequência

- Convolução

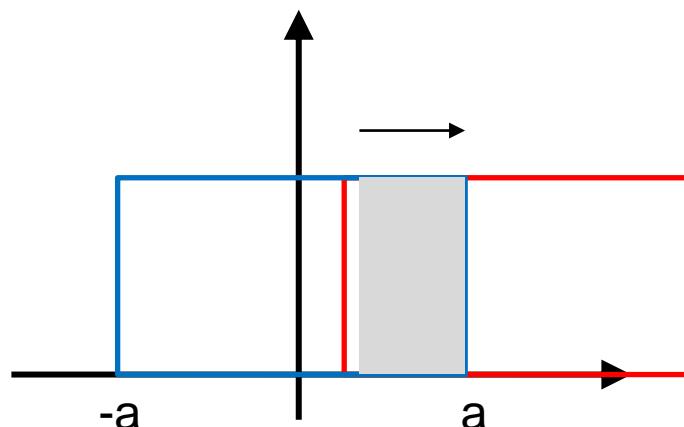


As duas funções estão  
sobrepostas....

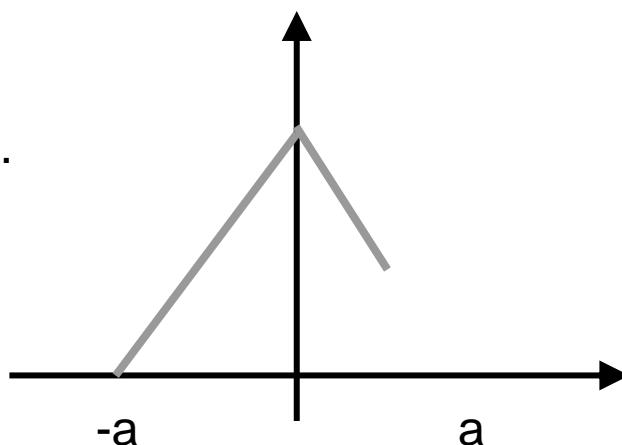


# Filtragem no Domínio da Frequência

- Convolução

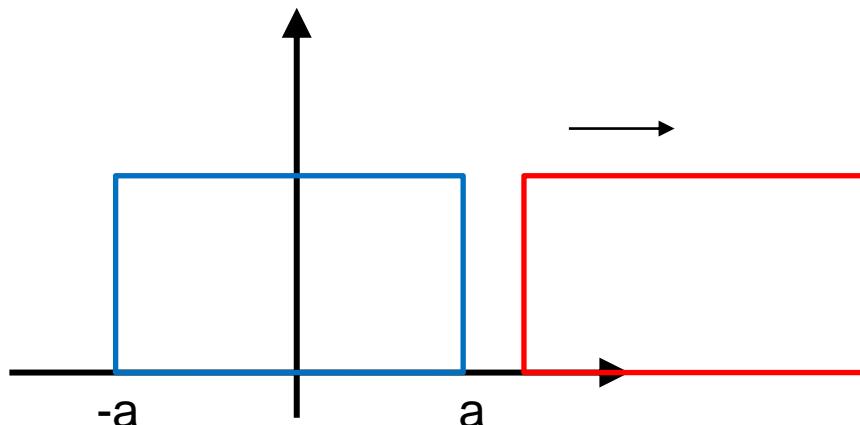


Continua o caminho....

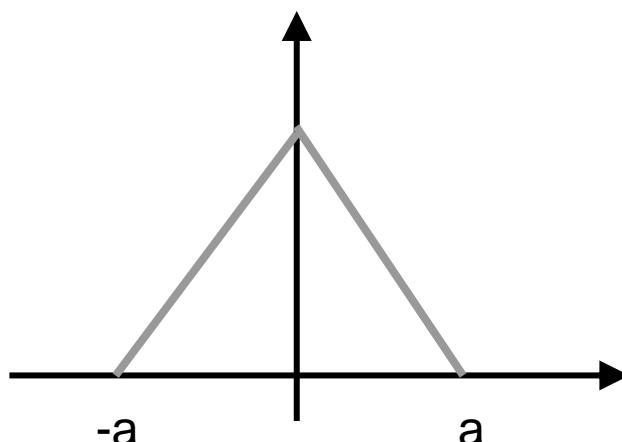


# Filtragem no Domínio da Frequência

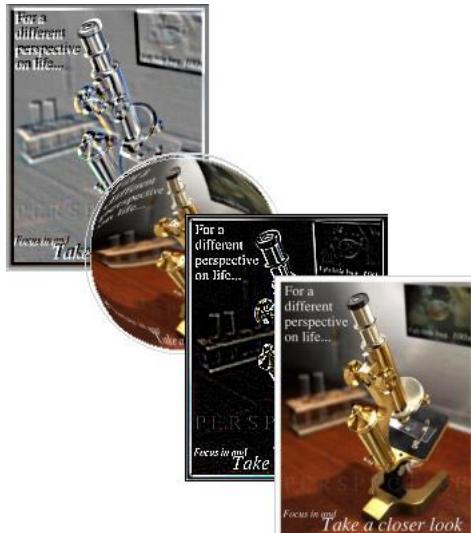
- Convolução



Até voltar a ter 0 de área em comum....



# Filtragem no Domínio Espacial



Carlos Alexandre Barros de Mello  
CIn/UFPE



# Convolução Discreta de Sinais

- A filtragem espacial em imagens segue passos semelhantes aos usados no domínio da frequência
- Aqui, no entanto, a filtragem é aplicada na imagem mesmo sem que ela seja convertida para outro domínio
- Assim, o processo precisa de adaptações

# Convolução Discreta de Sinais

- Convolução 2D

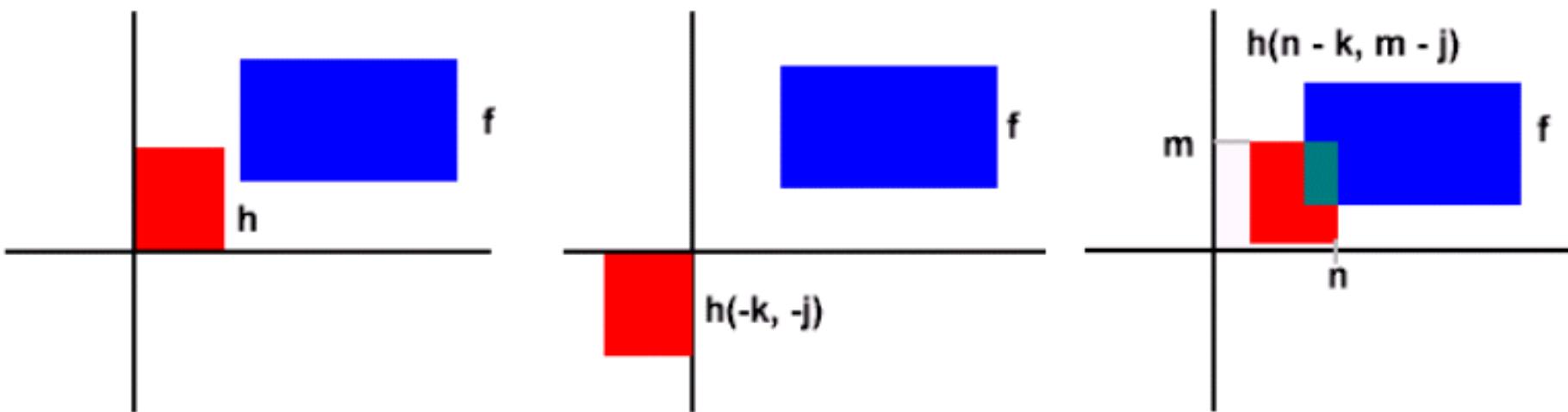
$$f(x, y) * h(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n)h(x - m, y - n)$$

- Teorema da convolução

$$f(x, y) * h(x, y) \Leftrightarrow F(u, v)H(u, v)$$

$$f(x, y)h(x, y) \Leftrightarrow F(u, v)*H(u, v)$$

# Convolução Discreta de Sinais



## Convolução Discreta

# Convolução Discreta de Sinais

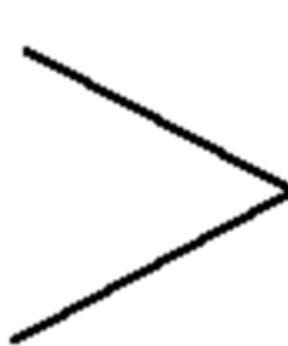
- Consideremos a convolução de sinais
  - Processo:

Matriz f

1	2	3
2	1	6

Filtro h

-1	2
2	-1



$f^*h$

-1	0	1	6
0	6	0	9
4	0	11	-6

Convolução Discreta em Sinais

# Convolução Discreta de Sinais

- Convolução discreta: como funciona...

1)

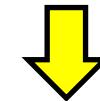
-1	2
2	-1

1	2	3
2	1	6

2)

-1	2		
2	-1	1	2
2	1	6	
2	1	6	

Área em comum:  
 $-1 \cdot 1 = -1$



-1	...	...
...	...	...

# Convolução Discreta de Sinais

## ■ Convolução discreta: como funciona...

3)

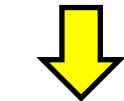
-1	2		
2	-1	2	3
1			

Área em comum:  
 $2 \cdot 1 + (-1) \cdot 2 = 0$

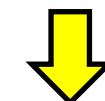
4)

-1	2		
1	2	-1	3
2	1	6	

Área em comum:  
 $2 \cdot 2 + (-1) \cdot 3 = 1$



-1	0	...
...	...	...
...	...	...



-1	0	1
...	...	...
...	...	...

# Convolução Discreta de Sinais

## ■ Convolução discreta: como funciona...

Área em comum:

$$2 \cdot 3 = 6$$

5)

-1	2
3	2

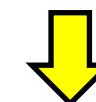
1	2	3	2	-1
2	1	6		



-1	0	1	6
...	...	...	...

6)

-1	2	1	2	3
2	-1	2	1	6



-1	0	1	6
0	...	...	...

Área em comum:  
 $2 \cdot 1 + (-1) \cdot 2 = 0$

# Convolução Discreta de Sinais

## ■ Convolução discreta: como funciona...

Área em comum:

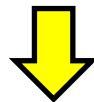
$$(-1).1 + 2.2 + 2.2 + (-1).1 = 6$$

7)

-1	1	2	2	3
2	2	-1	1	6

8)

1	-1	2	2	3
2	2	1	-1	6



-1	0	1	6
0	6	...	...



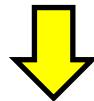
-1	0	1	6
0	6	0	...

# Convolução Discreta de Sinais

- Convolução discreta: como funciona...

9)

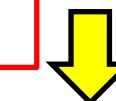
1	2	<b>3</b> <b>-1</b>	2
2	1	<b>6</b> <b>2</b>	-1



-1	0	1	6
0	6	0	9

10)

-1	<b>2</b> <b>2</b>	1	6
2	-1		



-1	0	1	6
0	6	0	9
4	...	...	...

.....

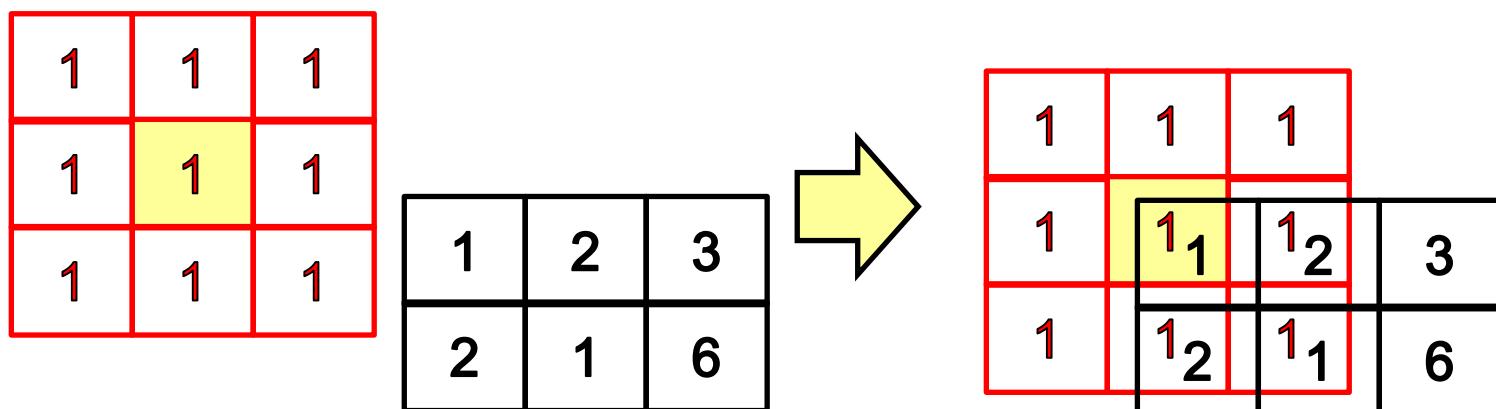
# Convolução Discreta de Sinais

- Convolução discreta: Problema:
  - Se, no exemplo anterior, temos uma imagem que você quer melhorar a qualidade, ela passa de 2 linhas e 3 colunas para um resultado final com 3 linhas e 4 colunas, o que não era esperado...
  - Assim, o processo precisa ser ajustado...

# Filtragem no Domínio Espacial

- Correlação:

- O casamento passa a ser do ponto central da máscara com o pixel a ser processado....



$$1.1 + 1.2 + 1.2 + 1.1 = 6$$

# Filtragem no Domínio Espacial

- Correlação:

- Mas, nesse exemplo, é como se tivéssemos uma borda zero ao redor da imagem – mas zero é preto!

1	0	1	0	1	0
1	0	1	1	1	2
1	1	2	3		

$$1.1 + 1.2 + 1.2 + 1.1 = 6$$

Na verdade, é:

$$1.0 + 1.0 + 1.0 + 1.0 + 1.0 + 1.1 + 1.2 + 1.2 + 1.1 = 6$$

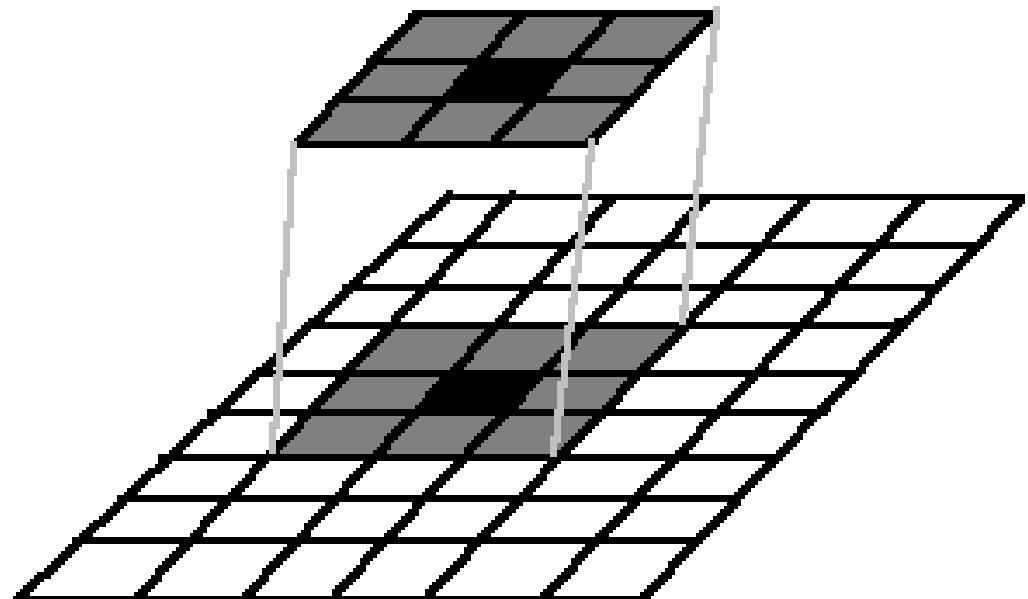
# Filtragem no Domínio Espacial

- Se o centro da máscara estiver numa posição  $(x,y)$  na imagem, o tom do pixel posicionado em  $(x,y)$  será substituído por  $R$
- A máscara é então movida para a próxima posição de pixel na imagem e o processo se repete
- É prática criar uma nova imagem para armazenar os valores de  $R$  em vez de mudar os valores de pixel no lugar
  - Evita o uso de pixels que tenham sido alterados na operação anterior

# Filtragem no Domínio Espacial

- O produto de convolução  $f^*h$  no pixel de coordenadas  $(m, n)$  é obtido colocando o centro da máscara acima do pixel  $(m,n)$ , multiplicando os elementos correspondentes na máscara e na imagem e somando os resultados

**Filtragem Espacial:  
Uso de máscaras**



# Filtragem no Domínio Espacial

- Seja a máscara 3x3

<b>W1</b>	<b>W2</b>	<b>W3</b>
<b>W4</b>	<b>W5</b>	<b>W6</b>
<b>W7</b>	<b>W8</b>	<b>W9</b>

- e sejam  $z_1, z_2, \dots, z_9$  a cor dos pixels sob a máscara
- O novo tom do pixel central será dado por
  - $R = W_1z_1 + W_2z_2 + \dots + W_9z_9$

# Aspectos Computacionais da Filtragem Discreta

- 1) Cor não realizável: cor resultante fora do espaço de cor do dispositivo.
  - Solução: Recorte para a cor mais próxima ou mudança de coordenada no espaço de cor.
  - Exemplo: Se o resultado de alguma operação der uma cor com valor menor que zero ou maior que 255, esses valores precisam ser corrigidos
    - Se cor < 0, então cor = 0
    - Se cor > 255, então cor = 255

# Aspectos Computacionais da Filtragem Discreta

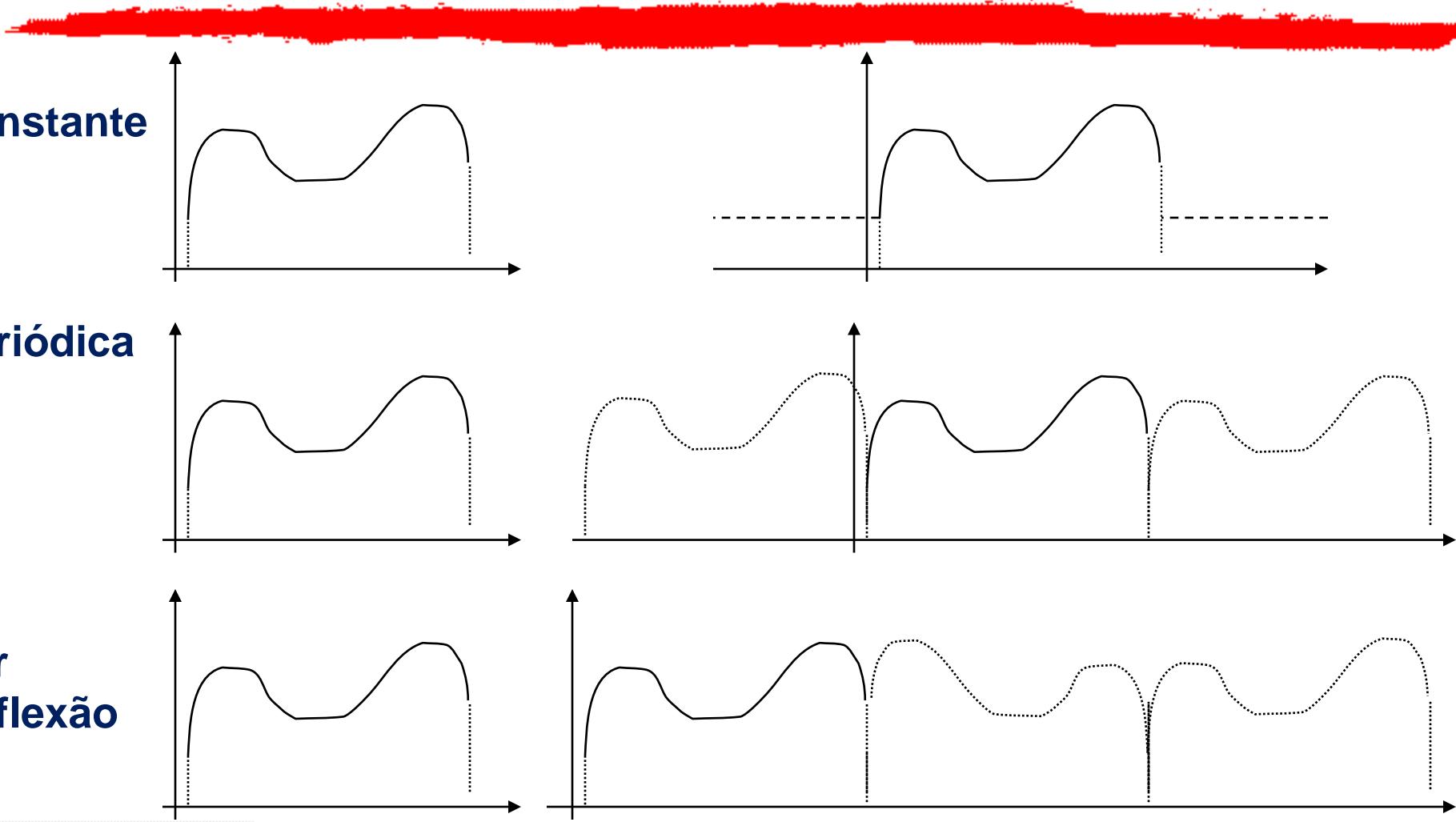
## ● 2) Eficiência Computacional:

- Solução: Dependente do problema
- Apenas percorrer a imagem já tem custo computacional  $O(N^2)$
- A melhoria da eficiência de seu programa passa por ter que percorrer a imagem a menor quantidade de vezes possível
  - Claro, continua  $O(N^2)$ , mas diminui a constante multiplicativa, o que implica em diminuição do tempo, na prática

# Aspectos Computacionais da Filtragem Discreta

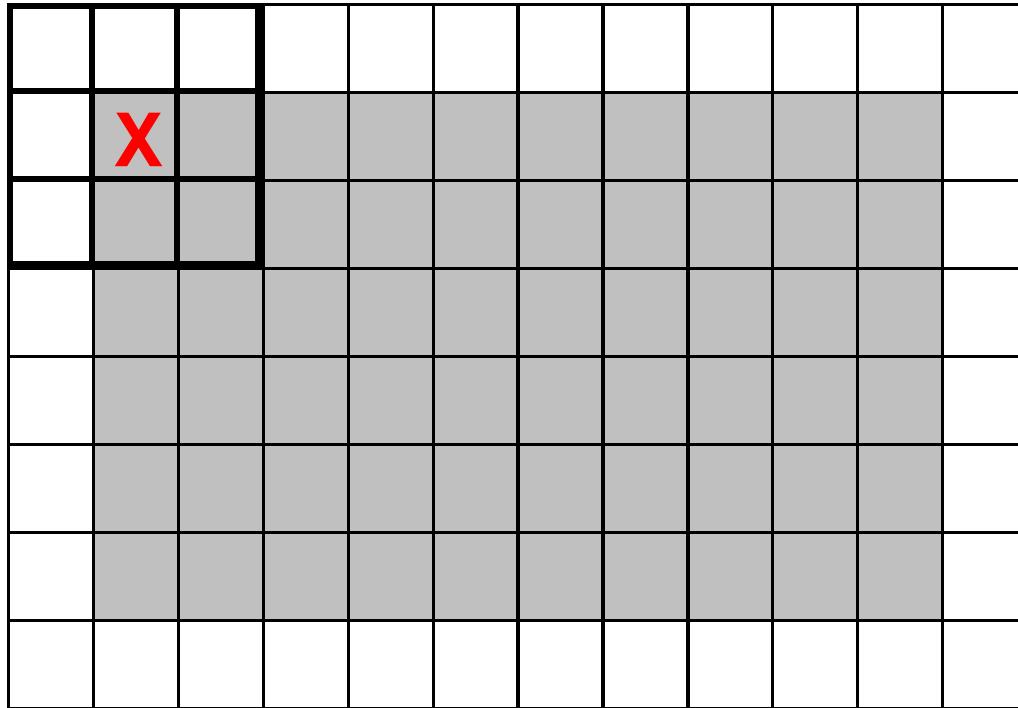
- 3) Extensão do Domínio da Imagem:
  - Solução:
    - Extensão Constante
      - Nula ou não
    - Extensão Periódica
    - Extensão por Reflexão
    - Não Extensão

# Extensão do Domínio da Imagem



# Convolução Discreta

- Extensões na Imagem

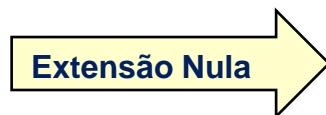


X – Pixel sendo processado

# Convolução Discreta

- Extensões na Imagem: Extensão constante ou nula

10	20	15
13	19	16
12	17	14



0	0	0	0	0
0	10	20	15	0
0	13	19	16	0
0	12	17	14	0
0	0	0	0	0

A extensão constante faz o mesmo com um valor diferente de zero.

# Convolução Discreta

- Extensões na Imagem: Extensão Periódica

19	16	13	19	16	13	19
17	14	12	17	14	12	17
20	15	10	20	15	10	20
19	16	13	19	16	13	19
17	14	12	17	14	12	17
20	15	10	20	15	10	20
19	16	13	9	16	13	19

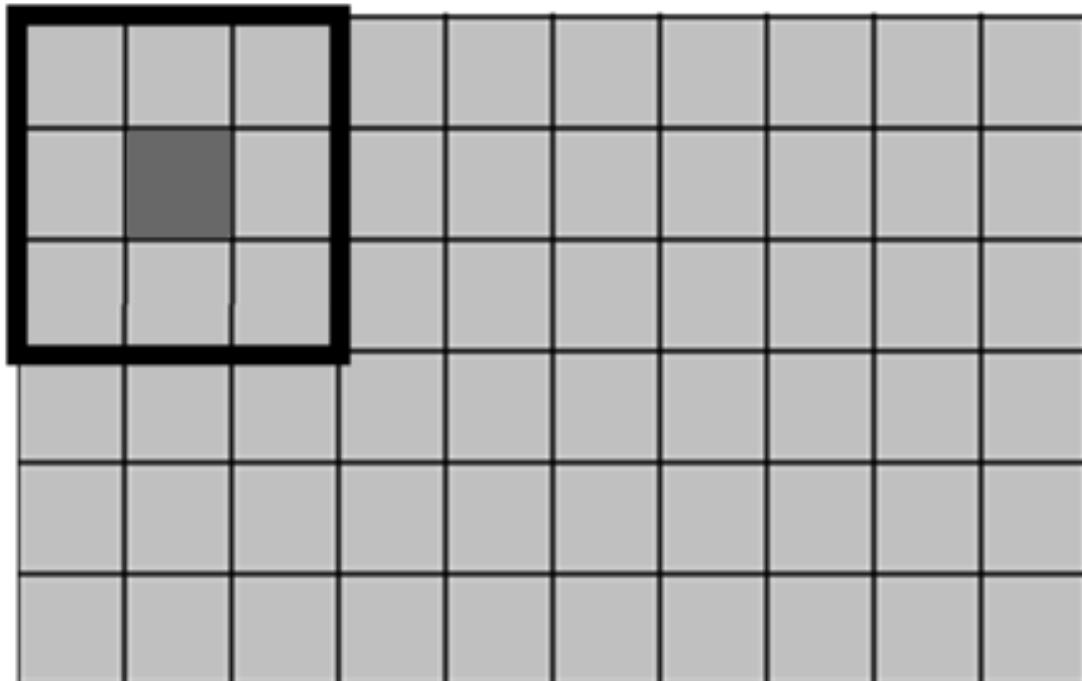
# Convolução Discreta

- Extensões na Imagem: Extensão por Reflexão

14	16	12	17	14	13	12
17	19	13	19	16	19	17
15	20	10	20	15	20	10
16	19	13	19	16	19	13
14	17	12	17	14	17	12
20	19	13	19	16	19	20
15	16	10	20	15	13	10

# Convolução Discreta

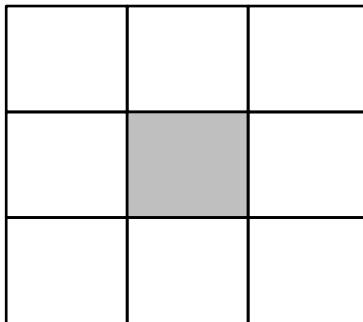
- Não Extensão da Imagem



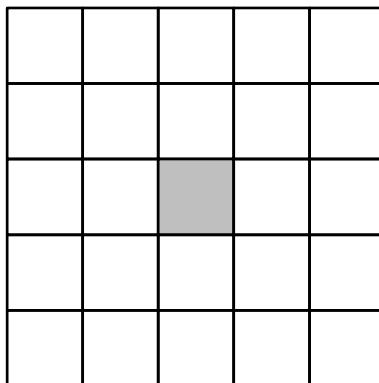
Nesse caso, o processamento começa de um pixel interno (pixel mais escuro na imagem ao lado) e as bordas são preservadas como na imagem original (sem qualquer mudança nos valores das cores).

# Convolução Discreta

- No caso de extensão, o quanto estender?



Filtro 3x3, processamento no pixel central: precisa estender uma linha para cima, uma para baixo, uma coluna para a esquerda e outra para a direita.



Filtro 5x5, processamento no pixel central: precisa estender duas linhas para cima, duas para baixo, duas coluna para a esquerda e duas para a direita.

E assim por diante....

# Tipos de Filtros Digitais

- Estatísticos ou Determinísticos
  - Estatísticos utilizam propriedades estatísticas da imagem para determinar seu valor em cada pixel
    - Exemplo: moda, mediana

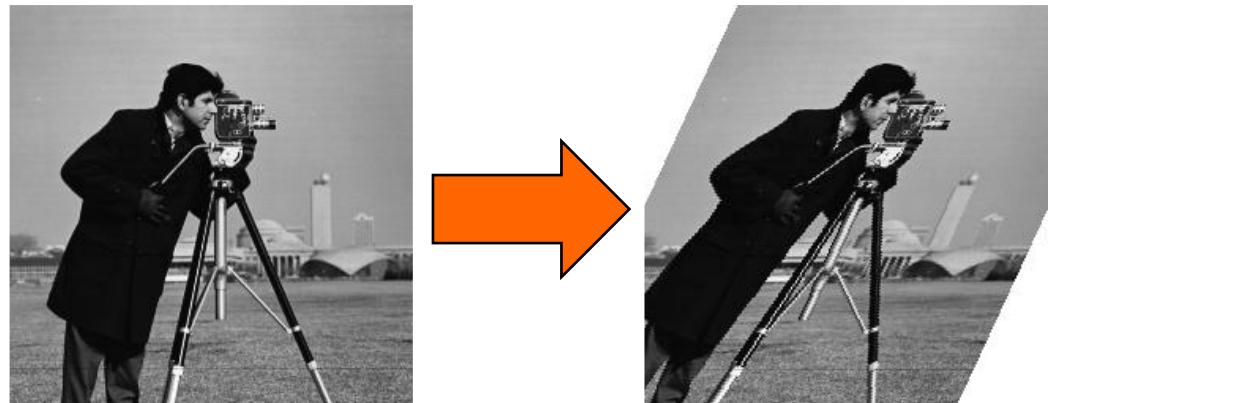
# Tipos de Filtros Digitais

- Topológicos ou de Amplitude
  - Topológicos: atuam no conjunto suporte da imagem
    - Exemplo: *Warping*
  - de Amplitude: atuam no espaço de cor
    - Exemplo: Correção Gama

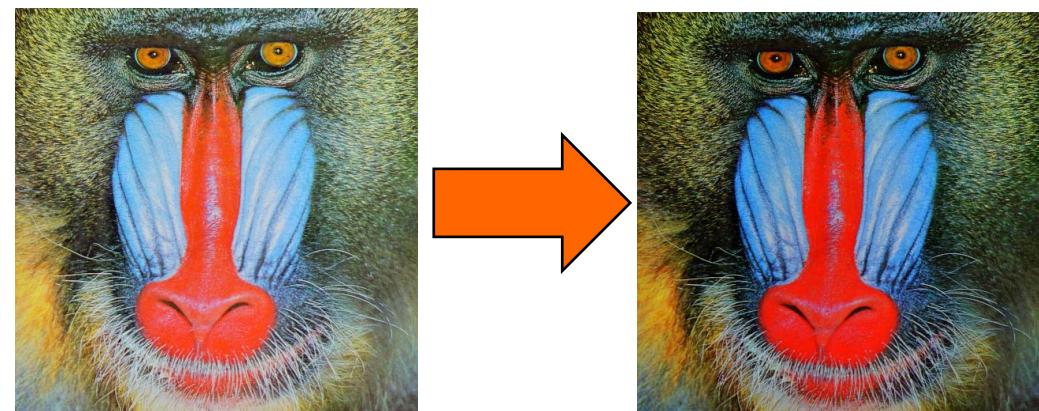
# Tipos de Filtros Digitais

- Topológicos ou de Amplitude

- Warping



- Correção Gama



# Tipos de Filtros Digitais

- Lineares ou Não-lineares
  - Lineares: não geram novas frequências na imagem
  - Obedecem ao teorema da sobreposição:
    - $T(\lambda f) = \lambda T(f)$
    - $T(f + g) = T(f) + T(g)$
  - Exemplo: Filtro Box (linear)

# Tipos de Filtros Digitais

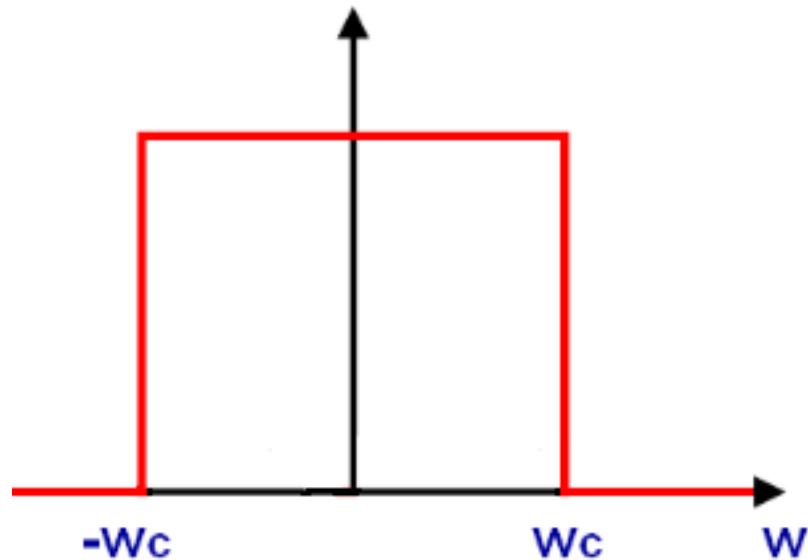
- Tipos de Filtros *Lineares*

- Filtro Passa-Baixa
- Filtro Passa-Alta
- Filtro Passa-Faixa

# Tipos de Filtros Lineares

- Filtro Passa-Baixa (*Low-Pass Filter*)

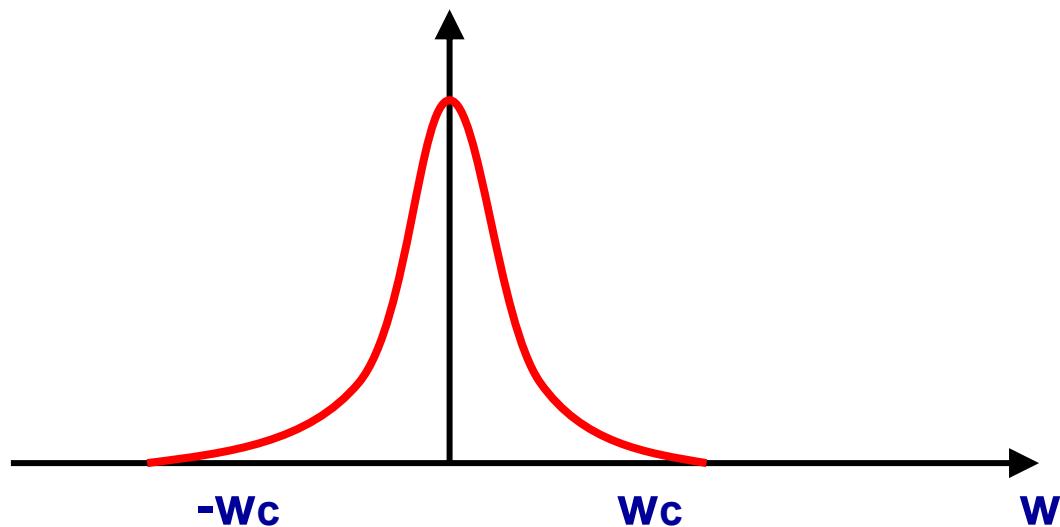
Permite passar as baixas frequências de uma imagem e atenua as altas



LPF ideal (não-realizável)

# Tipos de Filtros Lineares

- Filtro Passa-Baixa Real



# Tipos de Filtros Lineares

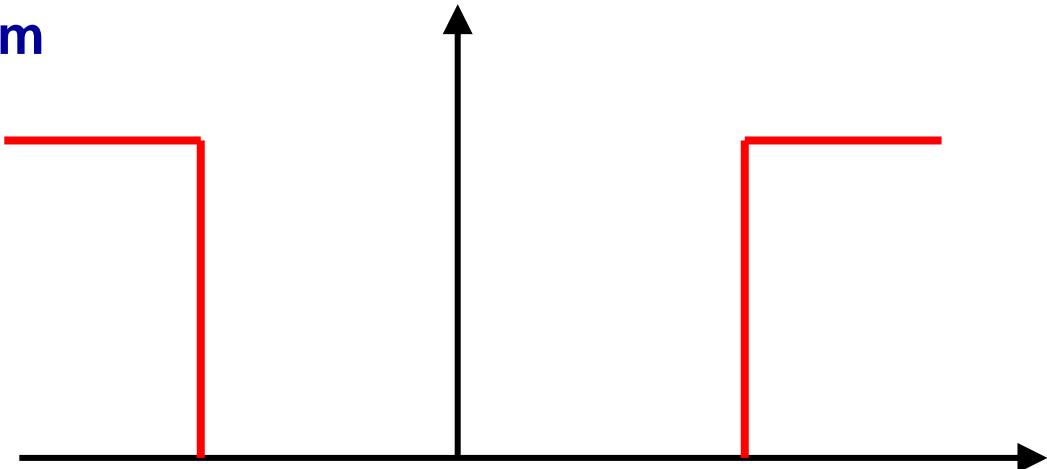
## ● Filtro Passa-Baixa

- Componentes de alta frequência caracterizam bordas ou outros detalhes finos de uma imagem
- O efeito resultante de um LPF é o embaçamento da imagem

# Tipos de Filtros Lineares

- Filtro Passa-Alta (*High-Pass Filter*)

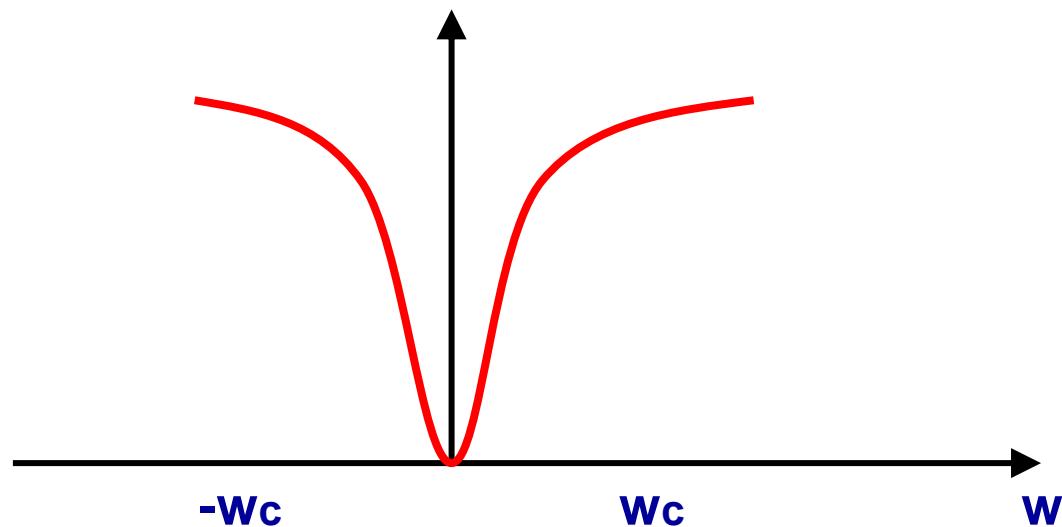
Permite passar as altas freqüências de uma imagem e atenua as baixas



HPF ideal (não-realizável)

# Tipos de Filtros Lineares

- Filtro Passa-Alta Real



# Tipos de Filtros Lineares

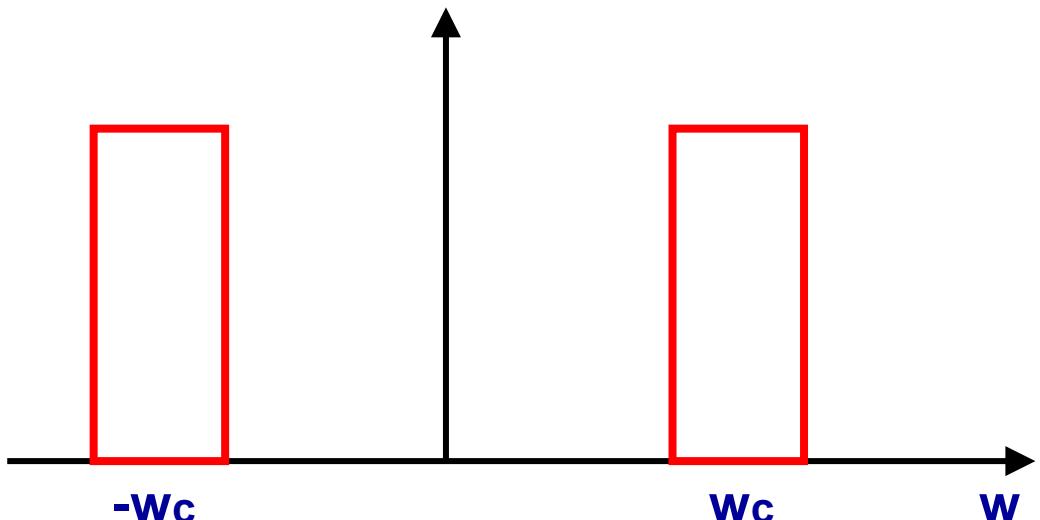
## ● Filtro Passa-Alta

- Redução de características que variam lentamente em uma imagem como o contraste e a intensidade média
- Efeito de intensificação das bordas e de detalhes finos na imagem

# Tipos de Filtros Lineares

- Filtro Passa-Faixa (*Band-Pass Filter*)

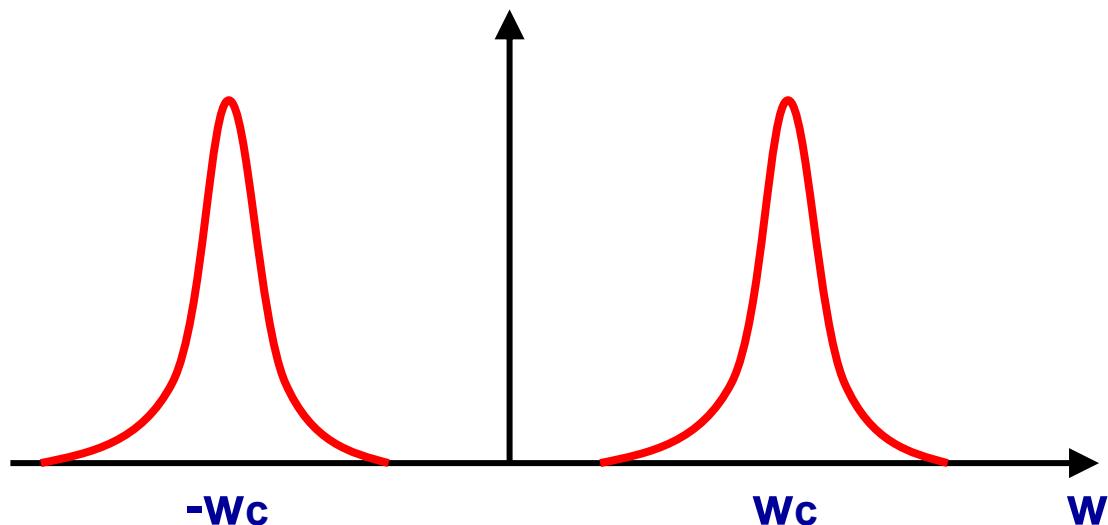
- Permite passar faixas específicas de uma imagem
- Removem regiões selecionadas
- Usados em restauração de imagens



BPF ideal (não-realizável)

# Tipos de Filtros Lineares

- Filtro Passa-Faixa Real



# Filtros de derivada

## Filtro de Prewitt - BPF

$h =$

$$\begin{matrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{matrix}$$



# Filtros de derivada

## Filtro de Prewitt2 - BPF

$h =$

$$\begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix}$$



# Filtros de derivada

## Filtro de Sobel1 - BPF

$h =$

$$\begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix}$$

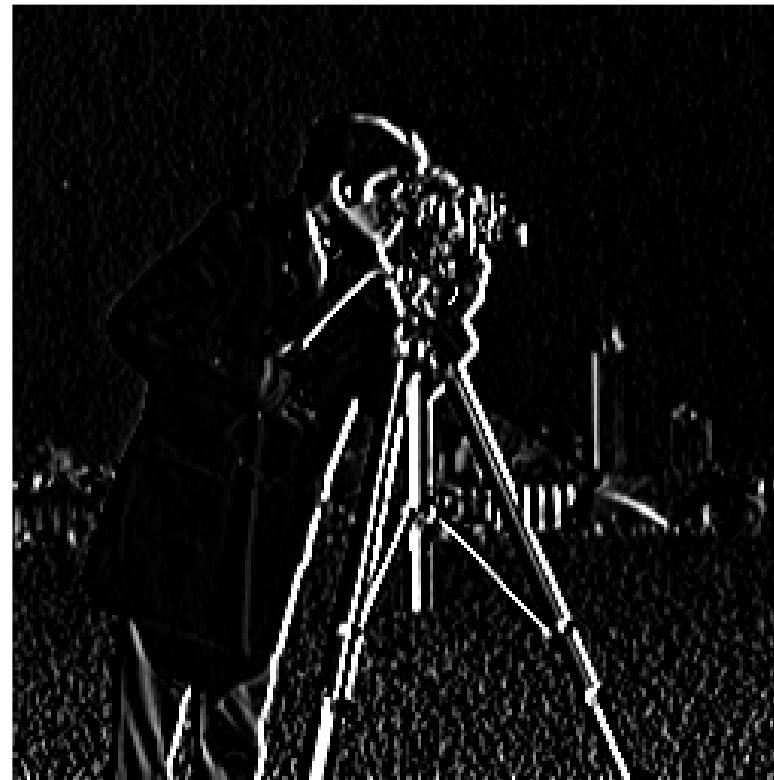


# Filtros de derivada

## Filtro de Sobel2 - BPF

$h =$

$$\begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$$



# Filtragem no MatLab

## Exemplo de Filtragem:

```
>> I = imread ('blood1.tif');  
>> h = fspecial ('average',5);  
>> I2 = filter2(h, I))  
>> imshow(I,[]);  
>> figure, imshow(I2, [])  
>> h  
h =  
0.0400 0.0400 0.0400 0.0400 0.0400  
0.0400 0.0400 0.0400 0.0400 0.0400  
0.0400 0.0400 0.0400 0.0400 0.0400  
0.0400 0.0400 0.0400 0.0400 0.0400  
0.0400 0.0400 0.0400 0.0400 0.0400
```

Cria uma matriz 5x5, só com 1, dividida pela soma dos 1's (25).

Implementa convolução 2D



# Filtragem no MatLab

- Exemplo (cont.):

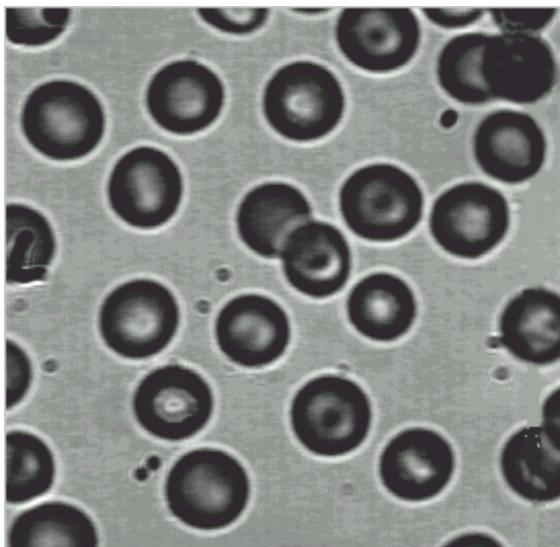


Imagen Original

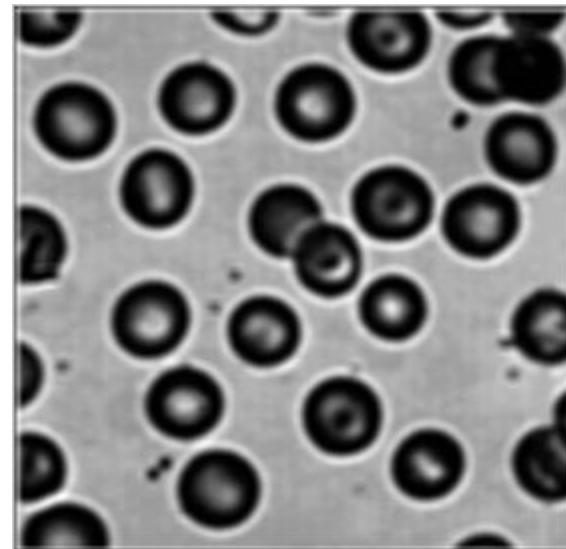


Imagen Filtrada

# Filtragem no MatLab

- Exemplo : Filtro Passa-Baixa

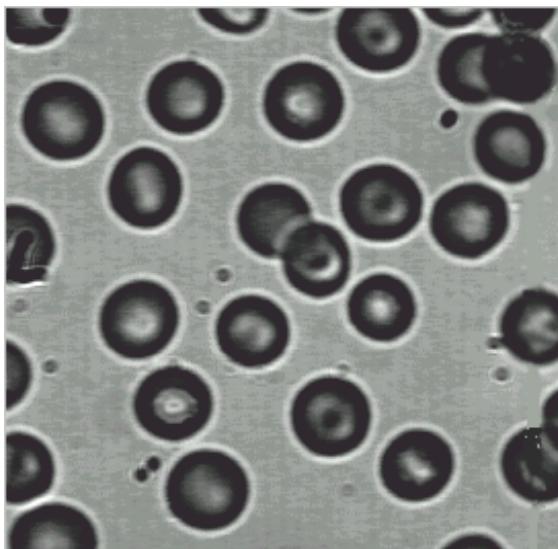


Imagen Original

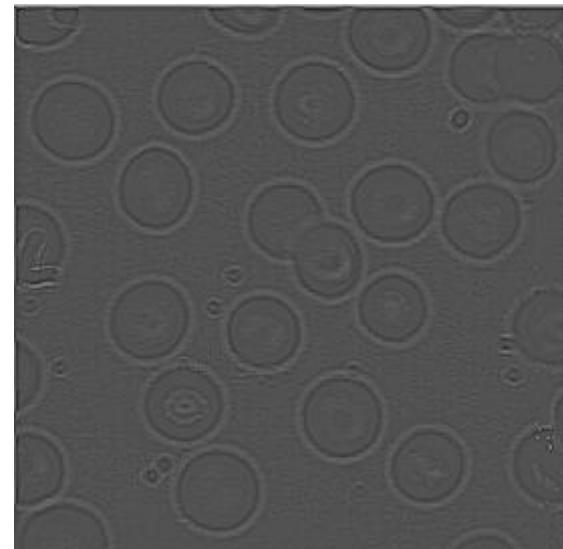


Imagen Filtrada

$h =$

$$\begin{matrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{matrix}$$

```
>> I = imread('blood1.tif');
>> h = [0 -1 0; -1 4 -1; 0 -1 0];
>> I2 = filter2(h, I);
>>
```



# Filtragem no MatLab

- Exemplo : Filtro Passa-Alta

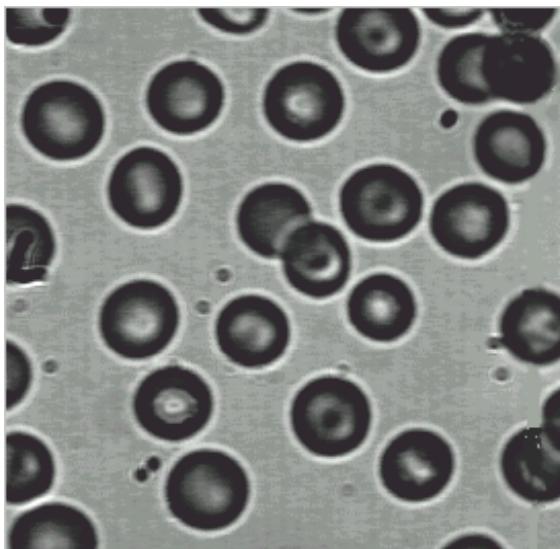


Imagen Original

$h =$

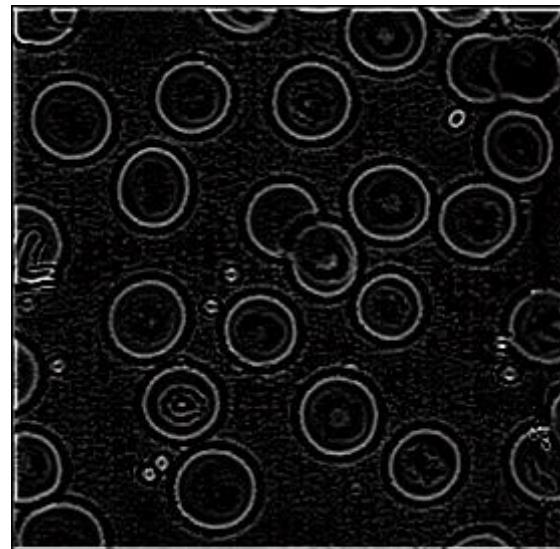


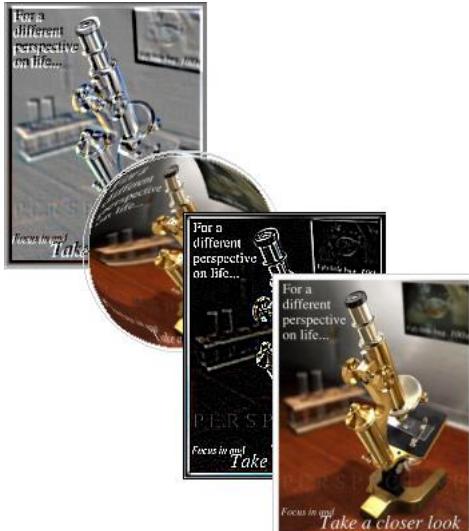
Imagen Filtrada

$$\begin{matrix} 0 & 1 & 1 & 0 \\ 1 & -2 & -2 & 1 \\ 1 & -2 & -2 & 1 \\ 0 & 1 & 1 & 0 \end{matrix}$$



# Filtros Estatísticos

Carlos Alexandre Mello  
CIn/UFPE



# Filtro Estatístico da Mediana

- Dada uma matriz de pixels (imagem), o filtro da **mediana** de ordem  $n$  varre a imagem, analisando regiões de tamanho  $nxn$
- Os pixels da região terão os seus valores organizados em ordem crescente de intensidade para calcular a mediana, o valor encontrado no centro da distribuição
- Todos os pixels da região serão substituídos pelo valor da mediana.
- O filtro da mediana elimina valores discrepantes, suavizando a imagem

# Filtro Estatístico da Mediana

- Admitamos ter um filtro de mediana de ordem 3
- Vamos analisar os 9 pixels em uma região 3x3
- Se os valores dos pixels encontrados é:

6 8 5 2 9 0 2 6 4

6	8	5
2	9	0
2	6	4

- A ordenação dos valores em ordem crescente é:
- 0 2 2 4 5 6 6 8 9
- A mediana da distribuição acima é 5
  - Todos os 9 pixels da região passarão a ter o valor da mediana, ou seja, 5

5	5	5
5	5	5
5	5	5

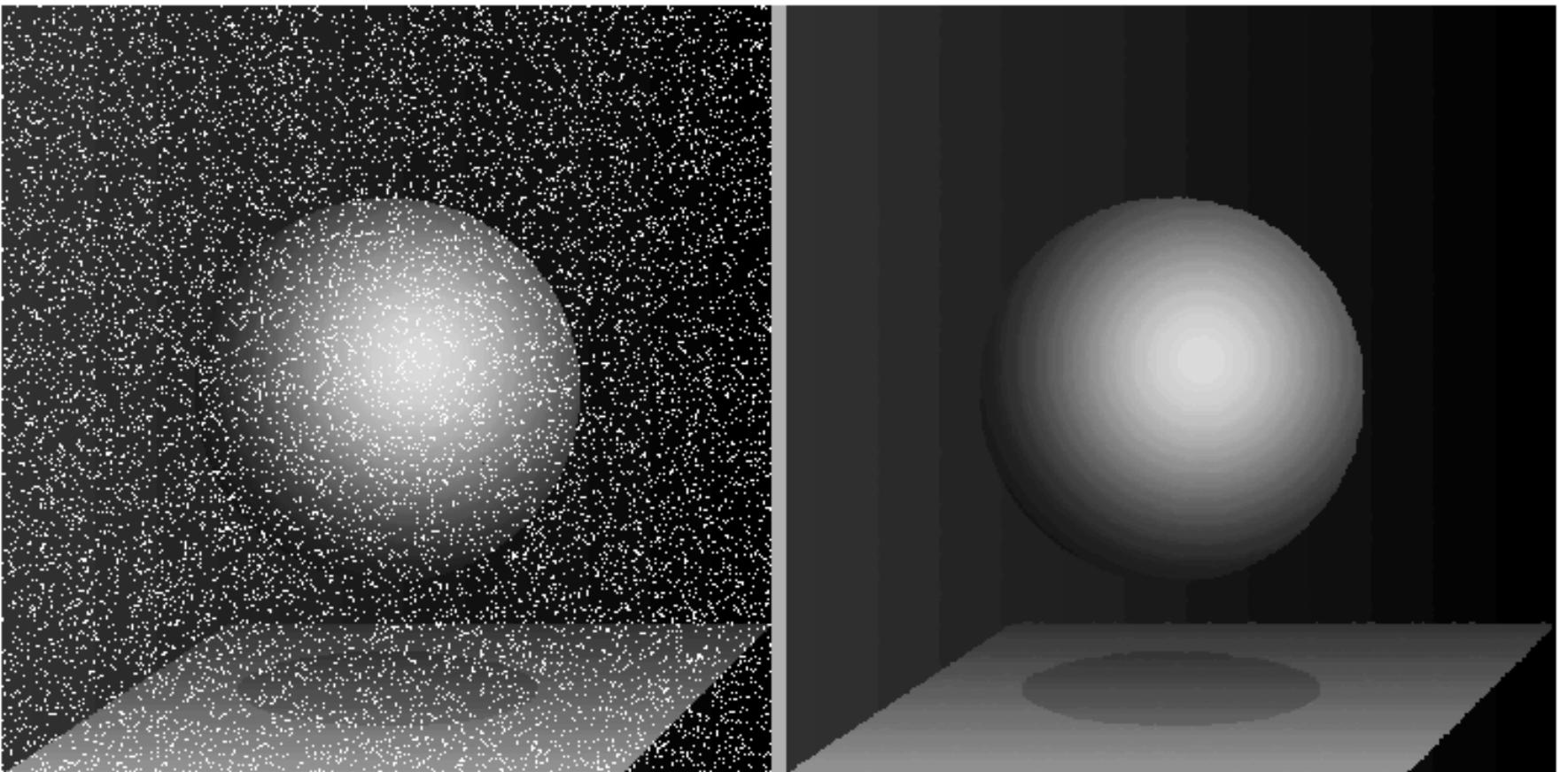
# Exemplo de Filtragem



# Exemplo de Filtragem: Mediana



# Exemplo de Filtragem: Mediana

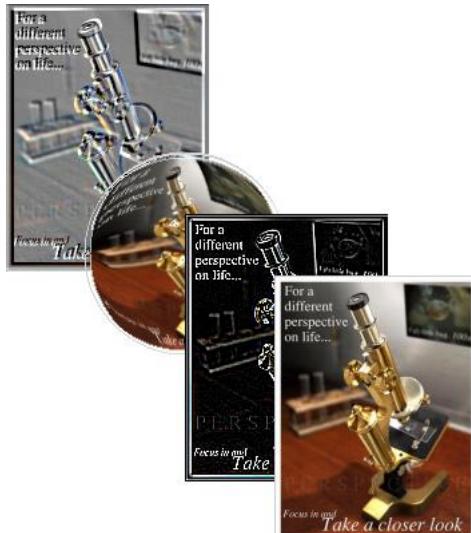


# Filtro Estatístico da Moda

- Dada uma matriz de pixels (imagem), o filtro da **moda** de ordem  $n$  varre a imagem, analisando regiões de tamanho  $nxn$
- Os pixels da região terão os seus valores lidos e sua moda é o valor do pixel que mais se repete
- Todos os pixels da região serão substituídos pelo valor da moda
- O filtro da moda elimina valores discrepantes, suavizando a imagem
- Os filtros da Mediana e da Moda são locais e, em geral, não-lineares

# Filtros de Amplitude

Carlos Alexandre Mello  
CIn/UFPE



# Filtros de Amplitude

- Modificam a **cor** do pixel
  - Mudam o sistema de cor da imagem ou fazem ajustes nas cores dos pixels
- Exemplos:
  - Filtros de Moda e Mediana
  - Correção Gama
  - Transformada de Gamute
  - Recorte de Cor

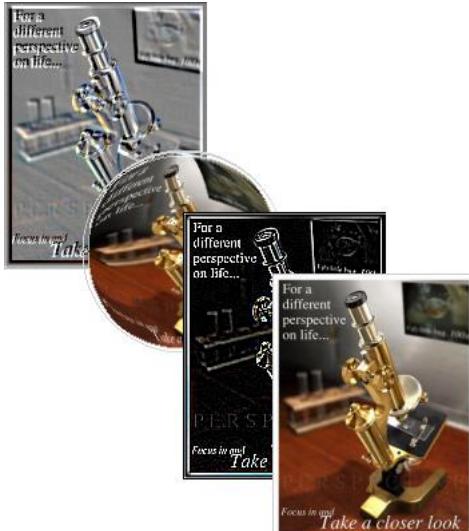
# Exemplo de Filtragem



# Filtro de Saturação em Red=0.60



# Filtros Topológicos



Carlos Alexandre Mello  
CIn/UFPE

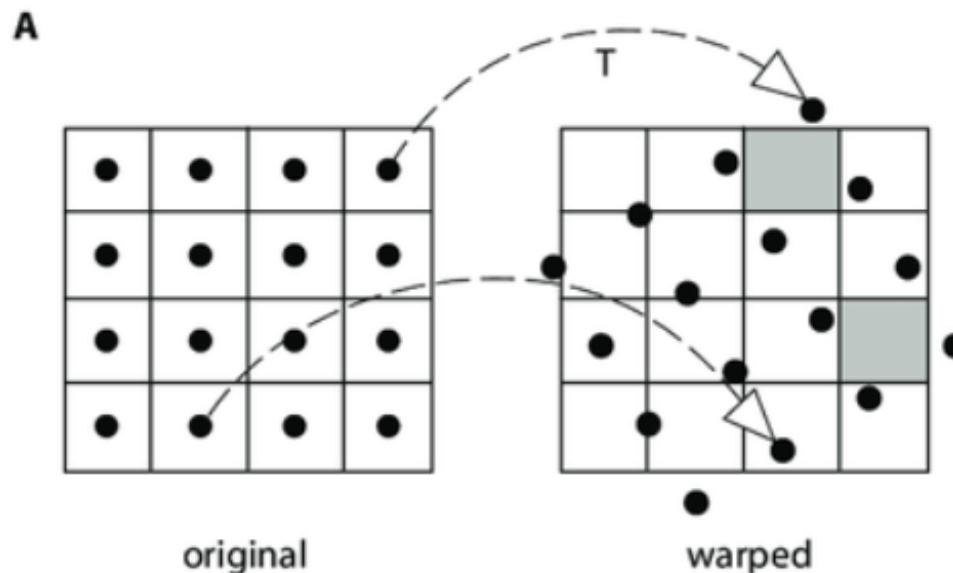


# Filtros Topológicos

- Modificam a estrutura dos objetos
- Exemplos:
  - Filtros de Warping
  - Filtros de Morphing
  - Rotação de imagens

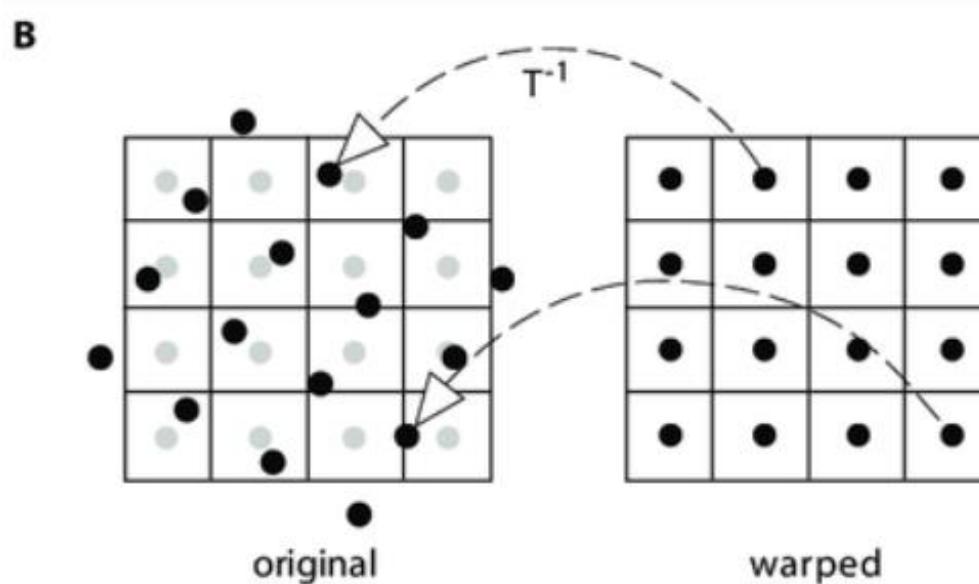
# Filtros Topológicos

- As operações de *warping* nada mais são do que o mapeamento de uma coordenada da imagem original para outra na imagem resultante
- A estrutura será mudada de acordo com esse mapeamento entre as coordenadas



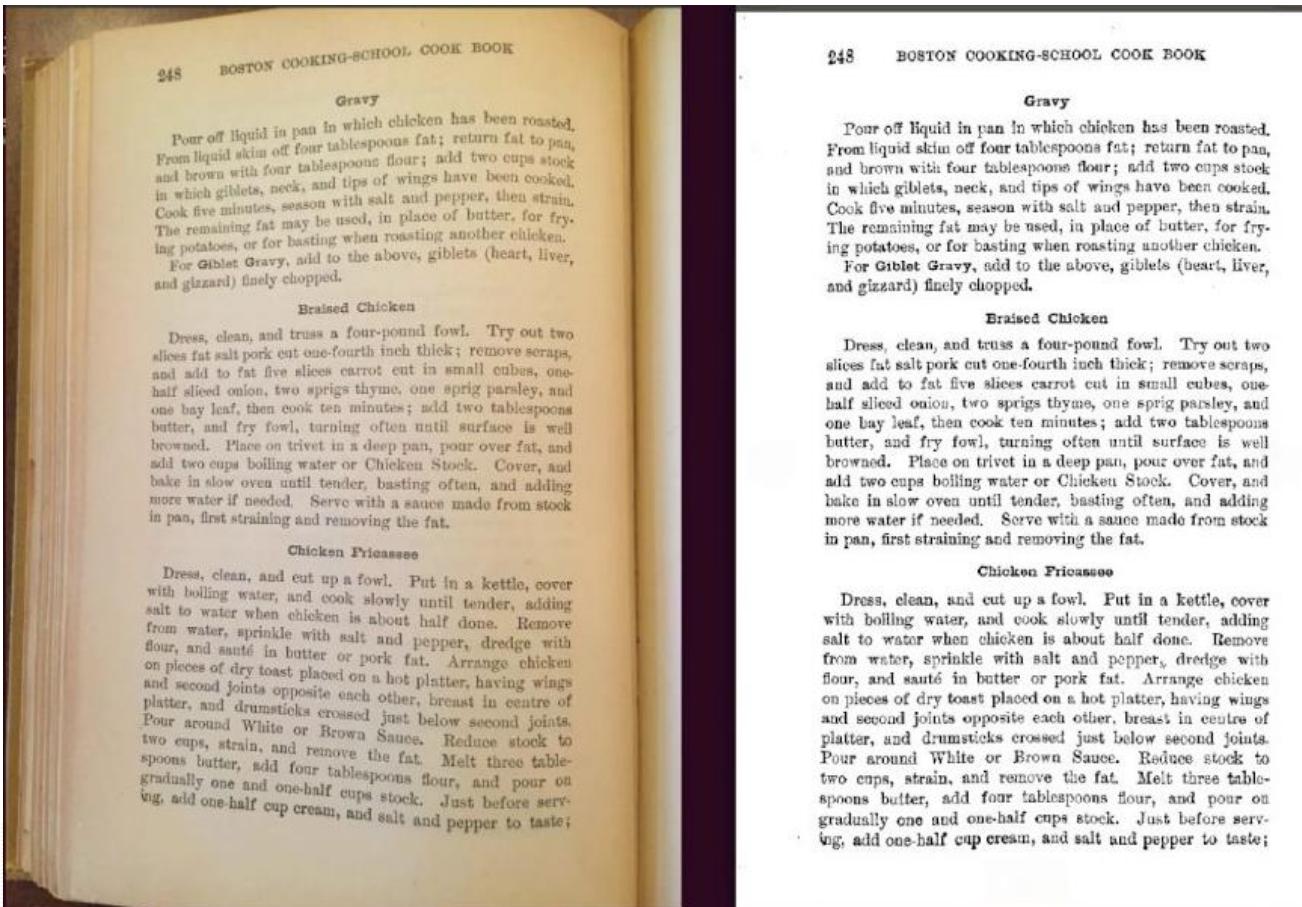
# Filtros Topológicos

- A transformação inversa é conhecida como *dewarping*



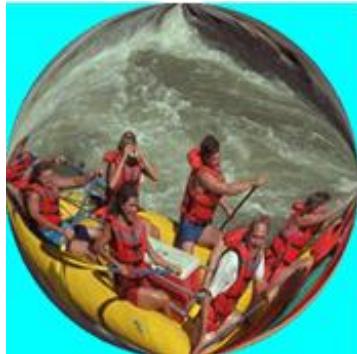
# Filtros Topológicos

- Exemplo de aplicação de *dewarping*



# Warping

Esfera



Pentágono



Cilindro Horizontal



Cilindro Vertical

# Warping



Deslocamento

# Warping



Perspective Horizontal e Vertical

Skew



# Warping



Flip



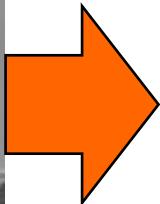
Mirror

# Warping

- Exemplo: Warping em uma onda....
  - $\text{im\_out}(u, v) = \text{im}(u + 20 * \sin(2\pi v / 128), v)$ 
    - im = Imagem de entrada
    - im\_out = imagem de saída
    - Observe que todo o processamento é apenas nas coordenadas

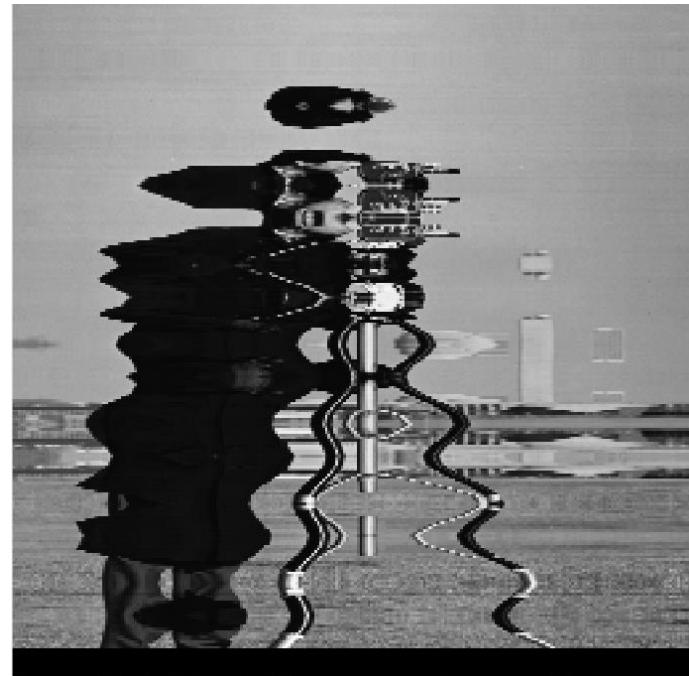
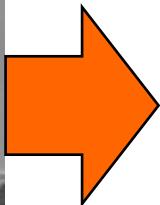
# Warping

- Exemplo: *Warping* em uma onda....



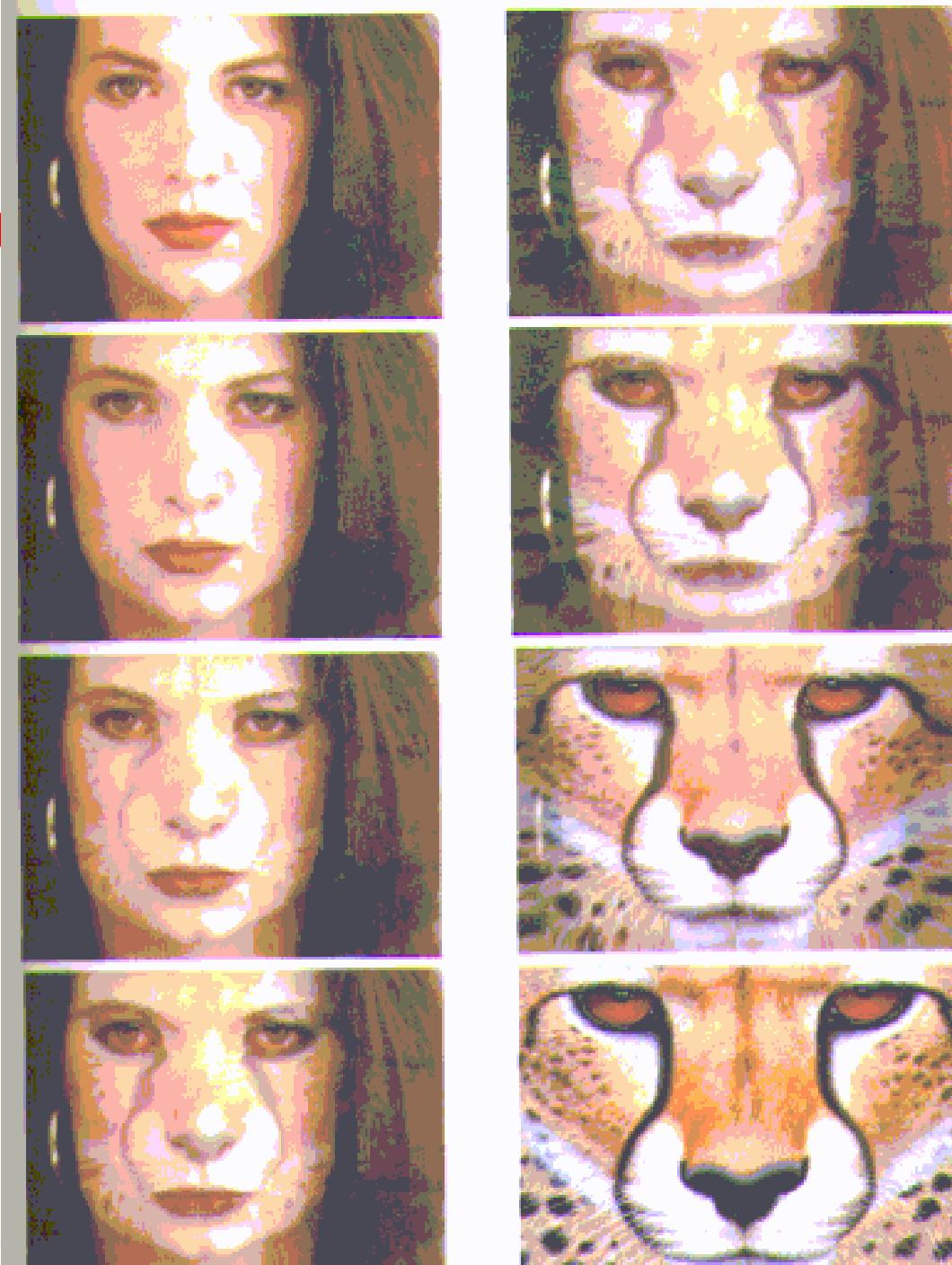
# Warping

- Exemplo: *Warping* em uma onda....



- $\text{im\_out}(u, v) = \text{im}(u + 20 * \sin(2\pi u / 128), v)$

# Filtro Topológico de Morphing



Processamen

# Filtro Morfológico de Morphing

- Transformação de uma imagem em outra

Cross  
Dissolve

$$I(t) = (1-t)*S+t*T$$

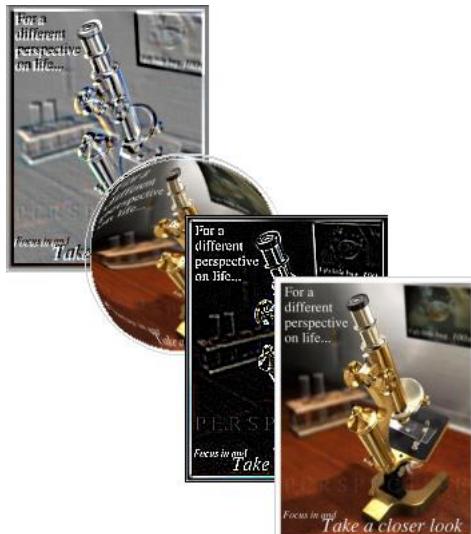


Mesh  
based



Non-Uniform Metamorphosis

# Filtros Lineares para Imagens



Carlos Alexandre Barros de Mello  
CIn/UFPE



# Filtros Lineares

- $T(k f) = k \cdot T(f)$
- $T(f + g) = T(f) + T(g)$

onde:       $k$  é um escalar (constante)

$f$  e  $g$  são vetores (imagens)

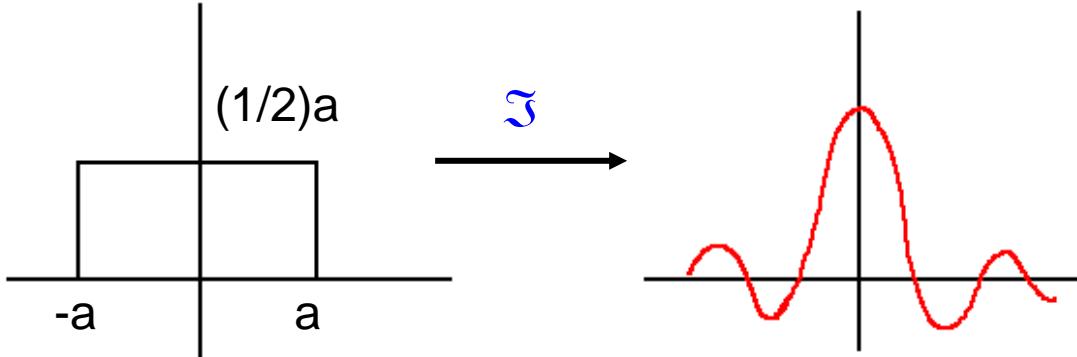
- Não geram novas frequências na imagem

# Exemplo de Filtros Lineares

- Filtros:

- Box
- Bartlett
- Gaussiano
- ...

# Filtro Box



- Média aritmética dos pixels na vizinhança de um dado pixel
- Atenua as altas frequências da imagem
- No caso bidimensional:
  - $\text{Box}(x,y) = \text{Box}(x).\text{Box}(y)$
  - Na frequência:  $\text{sample}(x,y) = \text{sample}(x).\text{sample}(y)$

# Filtro Box

$$\text{Box}(x) = \begin{cases} 1/2a & \text{se } -a \leq x \leq a \\ 0 & \text{se } |x| > a \end{cases}$$

Box 3x3

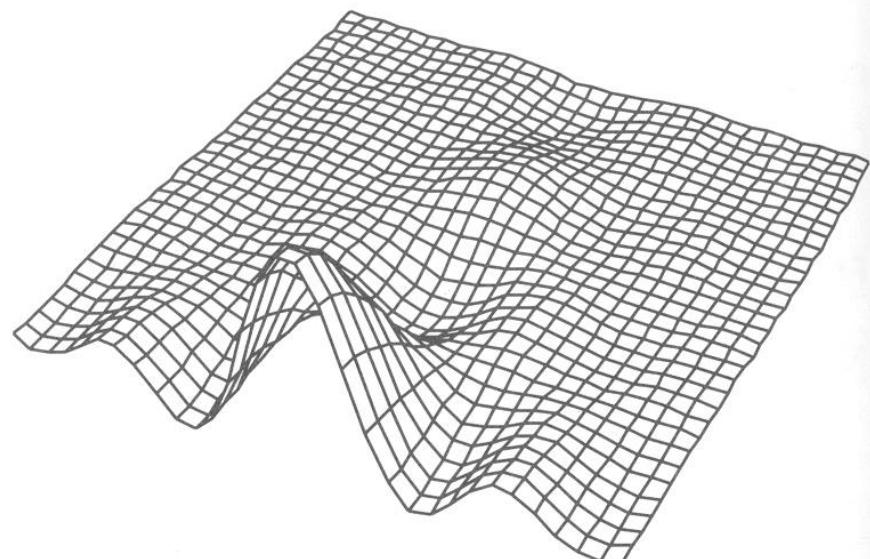
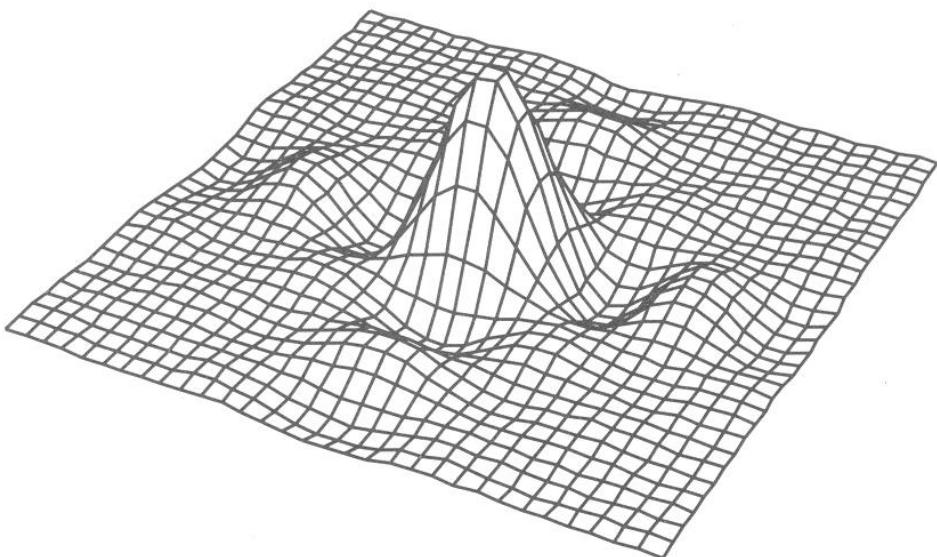
1	1	1
1	1	1
1	1	1

Fator de  
Normalização



# Filtro Box

## Transformada de Fourier do filtro Box



# Um pouco mais sobre filtros....

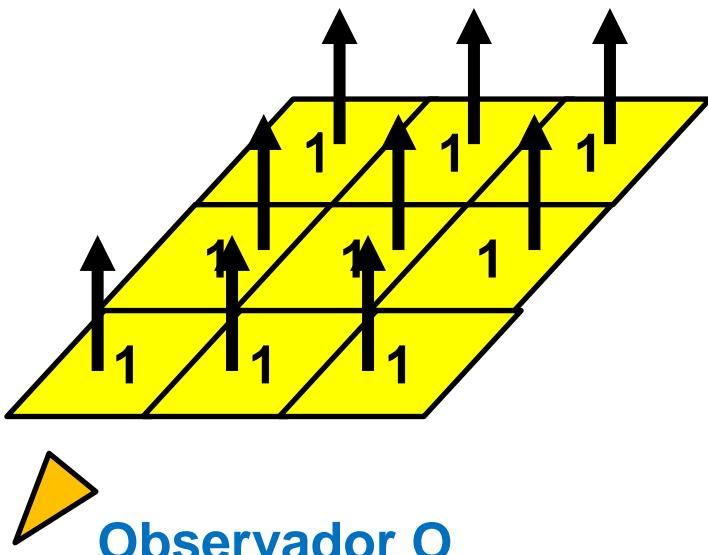
- Máscaras e sua interpretação ou cálculo
  - No domínio da frequência, os filtros são determinados por equações relacionadas com a frequência de corte e as características do filtro (ser *flat*, etc)
  - E no domínio espacial? Observe o filtro abaixo.... Que tipo de filtro é esse?

1	1	1
1	1	1
1	1	1

**OBS:** Para simplificar a explicação, o fator de normalização será suprimido, mas ele **sempre** deve ser aplicado.

# Um pouco mais sobre filtros....

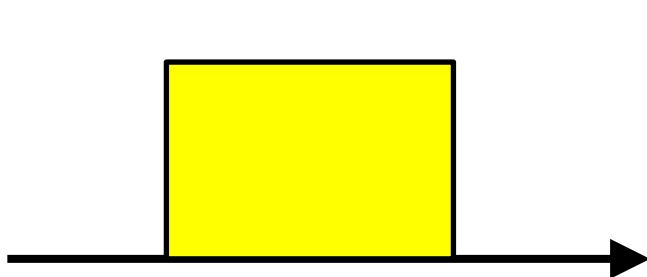
- Máscaras e sua interpretação ou cálculo
  - Considere a matriz como sendo um objeto tridimensional



Cada célula tem o mesmo valor 1. Entenda esse valor como sendo um vetor de mesma magnitude. Como um objeto tridimensional, formamos uma caixa. Considere o observador O; qual sua visão?

# Um pouco mais sobre filtros....

- Máscaras e sua interpretação ou cálculo
  - O filtro pode ser visto como um filtro passa baixa....

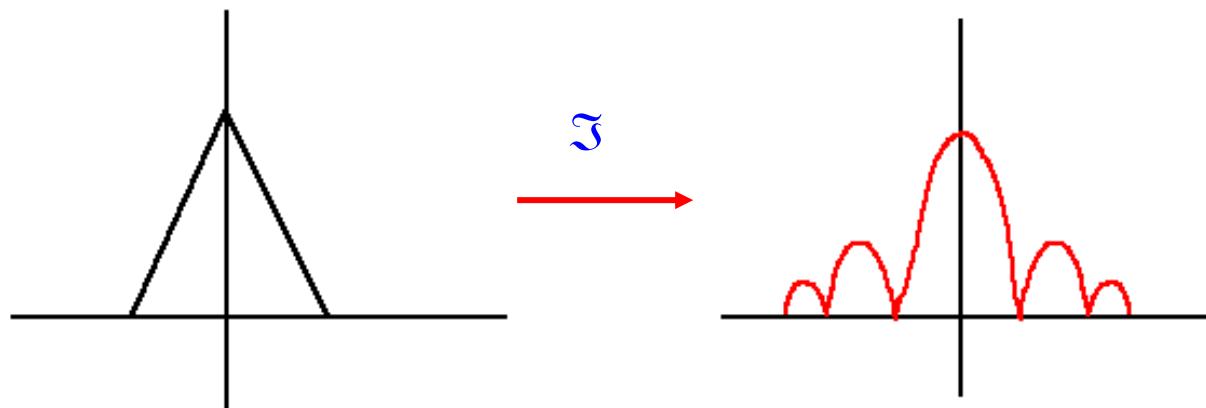


Visão do observador:  
filtro passa baixa (filtro  
box)

○ Mas um filtro passa baixa deve ser linear. Do jeito que está, ele é não linear. Para garantir a linearidade, a matriz deve ser multiplicada por 1/9... Isso garantirá que novas frequências não serão criadas....

# Filtro de Bartlett

- Filtro Triangular (ou filtro de Bartlett)



# Filtro de Bartlett

- Atenua as altas frequências
- $h(t) = \text{box}(t) * \text{box}(t)$ 
  - $*$  = operação de convolução
- $h(x,y) = h(x).h(y) \xrightarrow{\mathcal{F}} \text{sample}^2(x,y)$
- Atenuação mais acentuada que o Box

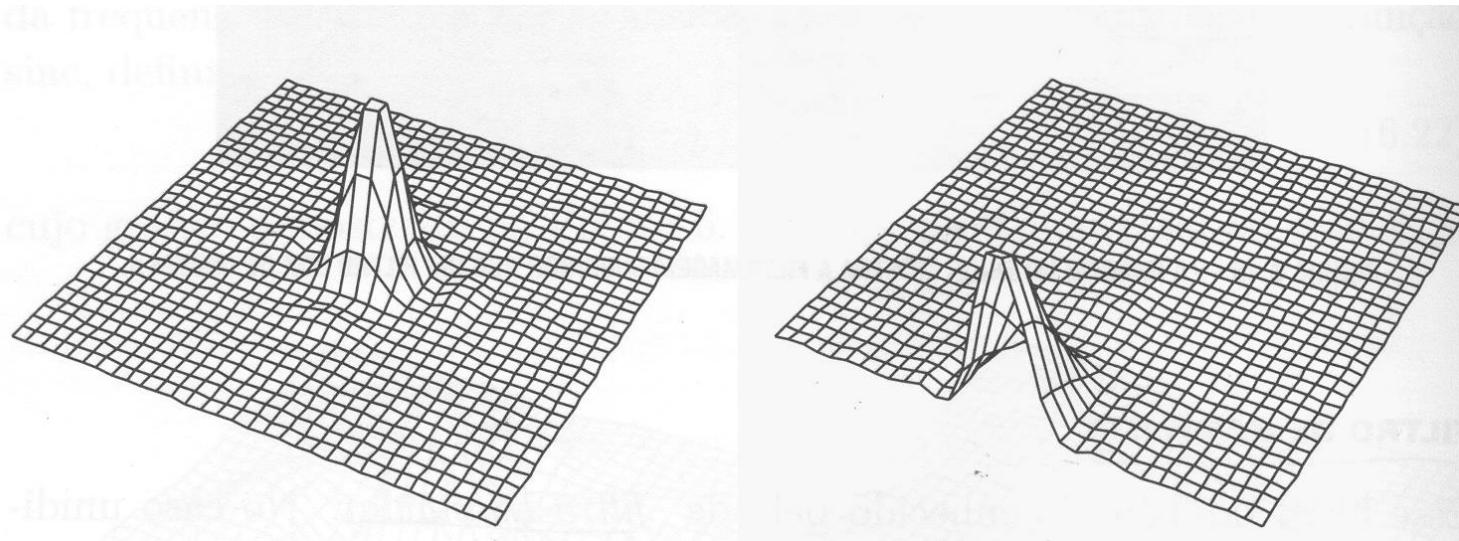
# Filtro de Bartlett

$$h(x) = \begin{cases} 1 - |x|, & \text{se } |x| \leq 1 \\ 0, & \text{se } |x| \geq 1 \end{cases}$$

$$\frac{1}{81} \begin{array}{|c|c|c|c|c|} \hline 1 & 2 & 3 & 2 & 1 \\ \hline 2 & 4 & 6 & 4 & 2 \\ \hline 3 & 6 & 9 & 6 & 3 \\ \hline 2 & 4 & 6 & 4 & 2 \\ \hline 1 & 2 & 3 & 2 & 1 \\ \hline \end{array}$$


# Filtro de Bartlett

Função de Transferência Bidimensional do Filtro Triangular



# Um pouco mais sobre filtros lineares....

- Máscaras e sua interpretação ou cálculo
  - Convolução de dois filtros passa baixa box:

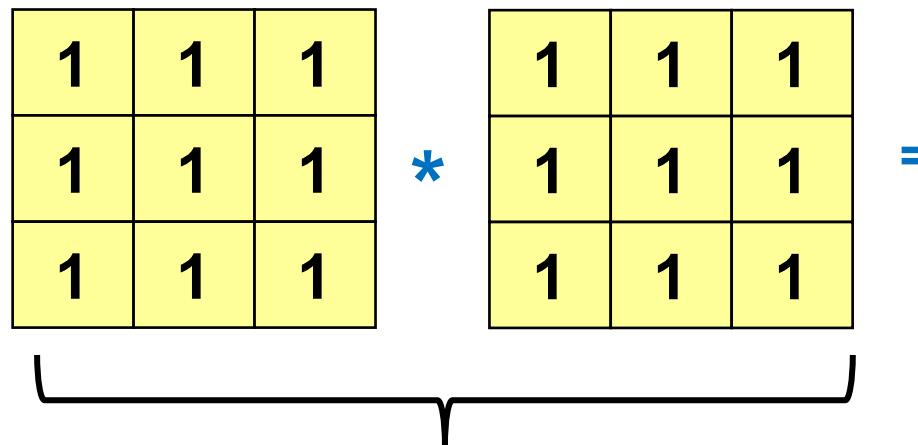
$$\frac{1}{9} \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix} * \frac{1}{9} \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

É uma convolução discreta entre sinais, não entre imagens!

# Um pouco mais sobre filtros lineares....

- Máscaras e sua interpretação ou cálculo
  - Convolução de dois filtros passa baixa box:

$$\frac{1}{9} \cdot \frac{1}{9} = \frac{1}{81}$$



Convolução Discreta

# Um pouco mais sobre filtros lineares....

- Máscaras e sua interpretação ou cálculo
  - Convolução de dois filtros passa baixa box:

The diagram illustrates the convolution of two 3x3 box filters. On the left, a 3x3 input matrix and a 3x3 mask are shown. The mask has a central value of 11, which is highlighted with a yellow box. An equals sign follows the input matrix. To the right is a 5x5 output matrix where every element is also highlighted with a yellow box, indicating the result of the convolution operation.

1	1	1
1	1	1
1	1	11

=

1	...	...	...	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...

# Um pouco mais sobre filtros lineares....

- Máscaras e sua interpretação ou cálculo
  - Convolução de dois filtros passa baixa box:

1	1	1	
1	1	1	
1	<b>11</b>	<b>11</b>	1
1	1	1	1
1	1	1	1

=

1	2	...	...	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...

# Um pouco mais sobre filtros lineares....

- Máscaras e sua interpretação ou cálculo
  - Convolução de dois filtros passa baixa box:

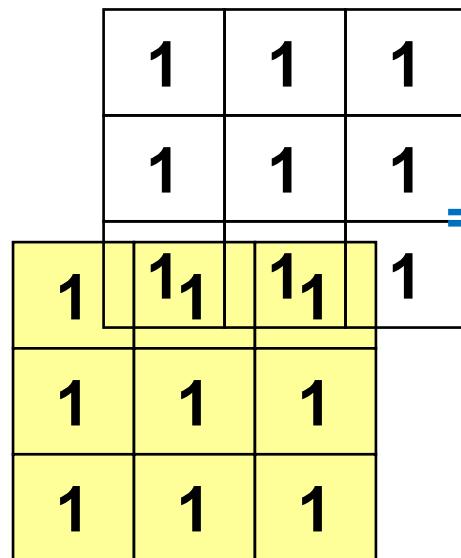
1	1	1
1	1	1
<b>11</b>	<b>11</b>	<b>11</b>
1	1	1
1	1	1

=

1	2	3	...	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...

# Um pouco mais sobre filtros lineares....

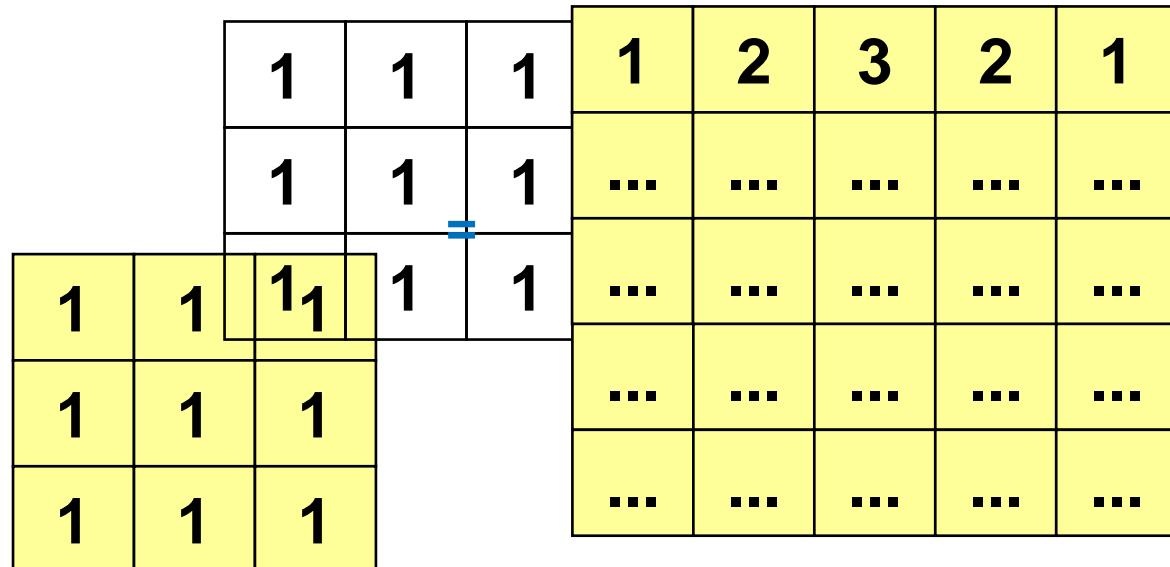
- Máscaras e sua interpretação ou cálculo
  - Convolução de dois filtros passa baixa box:



1	2	3	2	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...
...	...	...	...	...

# Um pouco mais sobre filtros lineares....

- Máscaras e sua interpretação ou cálculo
  - Convolução de dois filtros passa baixa box:



# Um pouco mais sobre filtros lineares....

- Máscaras e sua interpretação ou cálculo
  - Convolução de dois filtros passa baixa box:

1  
81

1	2	3	2	1
2	4	6	4	2
3	6	9	6	3
2	4	6	4	2
1	2	3	2	1

Que é a máscara do filtro de Bartlett.  
Observe que a convolução de duas funções porta (2D) iguais é um sinal triangular....

# Filtro Gaussiano

$$G_{\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-x^2/2\sigma^2)$$

Média = 0

Variância =  $\sigma^2$

# Filtro Gaussiano

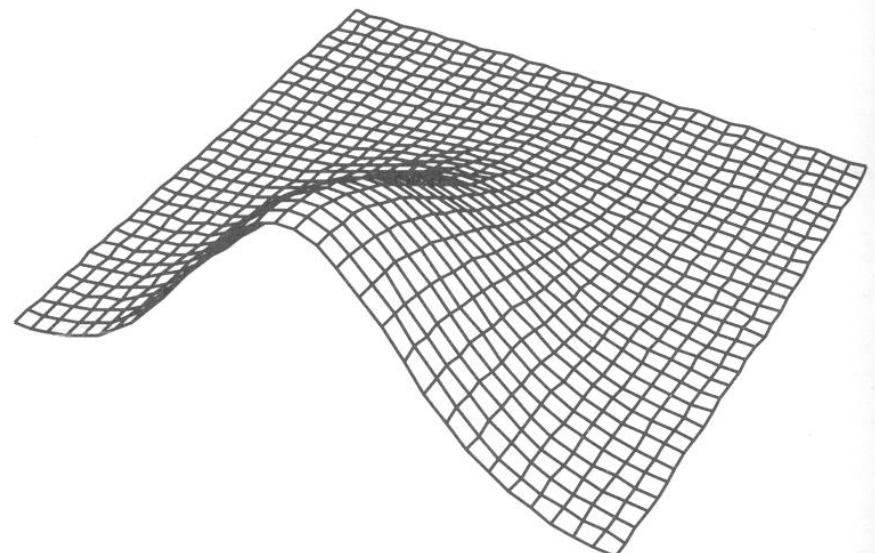
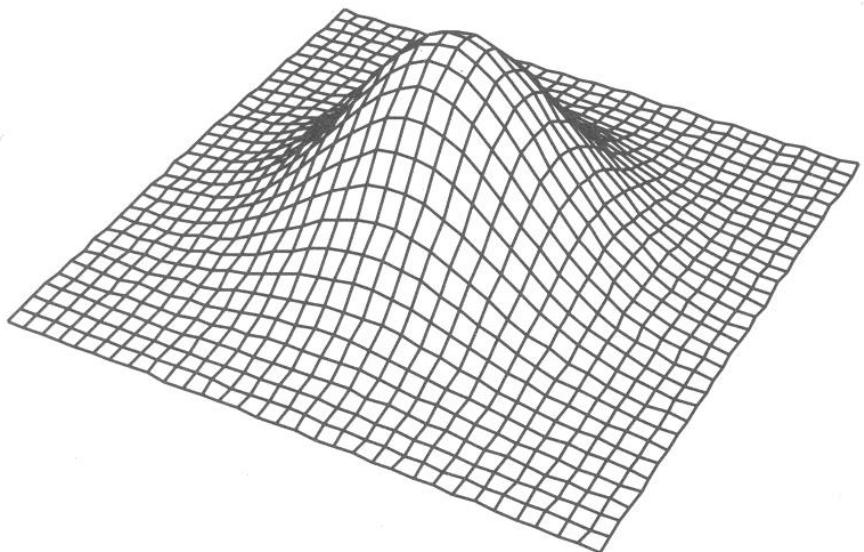
- Filtro Passa-Baixa
- $h(x) = \text{box}_1(x) * \text{box}_2(x) * \dots * \text{box}_n(x) \rightarrow$  filtro gaussiano

$$\frac{1}{256} \left( \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 3 & 2 & 1 \\ 2 & 4 & 6 & 8 & 6 & 4 & 2 \\ 3 & 6 & 9 & 12 & 9 & 6 & 3 \\ 4 & 8 & 12 & 16 & 12 & 8 & 4 \\ 3 & 6 & 9 & 12 & 9 & 6 & 3 \\ 2 & 4 & 6 & 8 & 6 & 4 & 2 \\ 1 & 2 & 3 & 4 & 3 & 2 & 1 \end{array} \right) \longrightarrow \text{Máscara Binomial}$$

Observação: O filtro de Bartlett é um filtro gaussiano com poucas amostras ( $n = 2$ ).

# Filtro Gaussiano

Função Gaussiana de Média 0 e Variância 2



# Exemplo de Filtragem

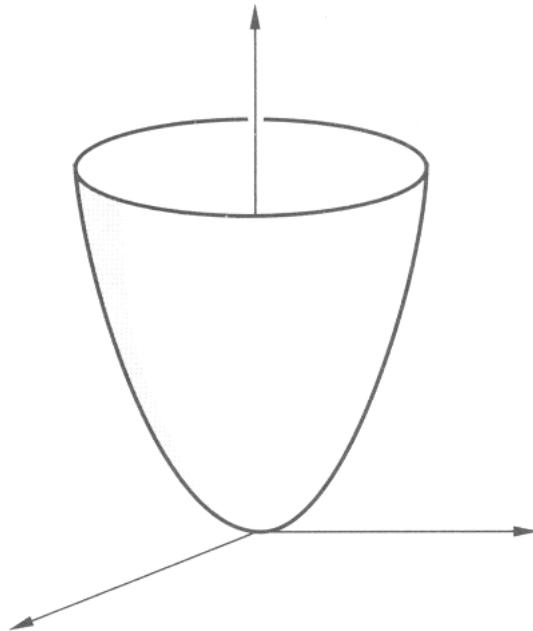


# Filtragem Gaussiana



# Filtro Laplaciano

- Função de Transferência do Filtro Laplaciano  
(filtro Passa Alta)



# Filtro Laplaciano

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

Atenua as Baixas Frequências

Acentua as Altas Frequências



# Filtro Laplaciano

0	1	0
1	-4	1
0	1	0

Máscara 3x3



# Exemplo de Filtragem



# Filtro Passa-Alta



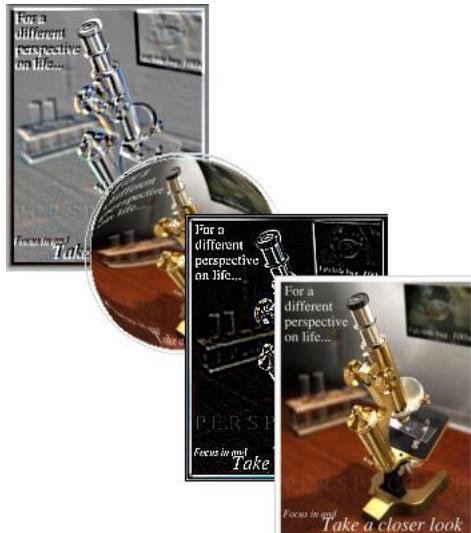
# Exemplo de Filtragem



# Filtro Passa-Alta Mais Forte



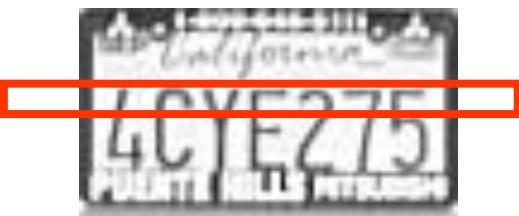
# *Edge Detection - Detecção de Bordas*



**Carlos Alexandre Barros de Mello**  
**CIn/UFPE**

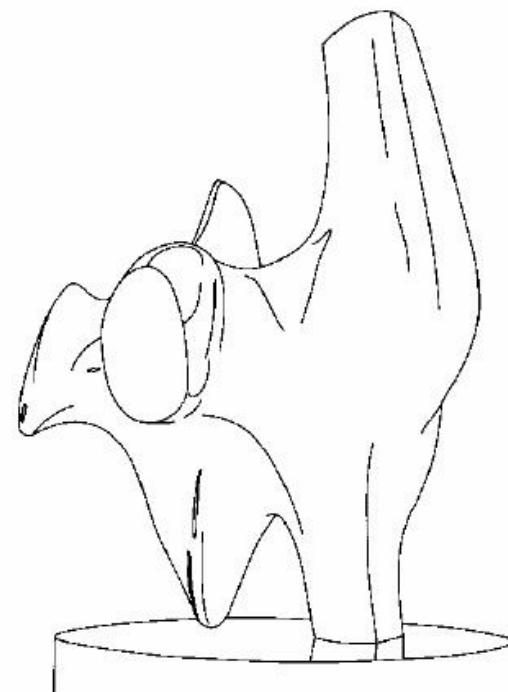
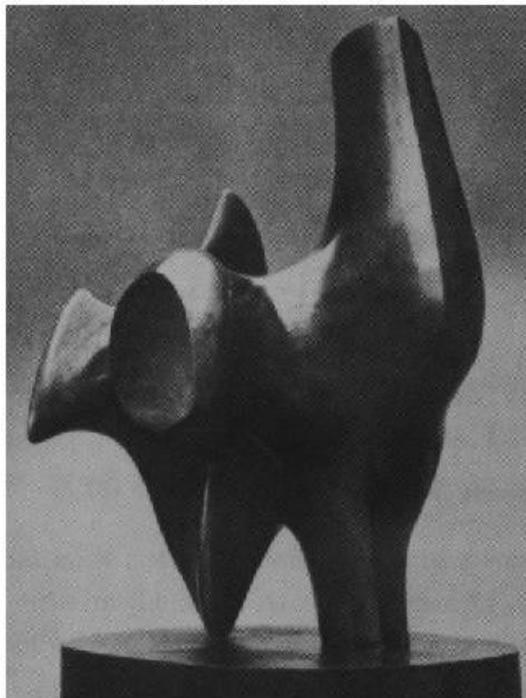


# Edge Detection – Detecção de Bordas

- Principais algoritmos
    - Baseados em Descontinuidade
      - A imagem é particionada, tomando como base mudanças bruscas em níveis de cinza
    - Baseados em Similaridades
      - Baseada em limiarização, crescimento de regiões e divisão e fusão de regiões
- 
- Mudanças entre tons claros e escuros

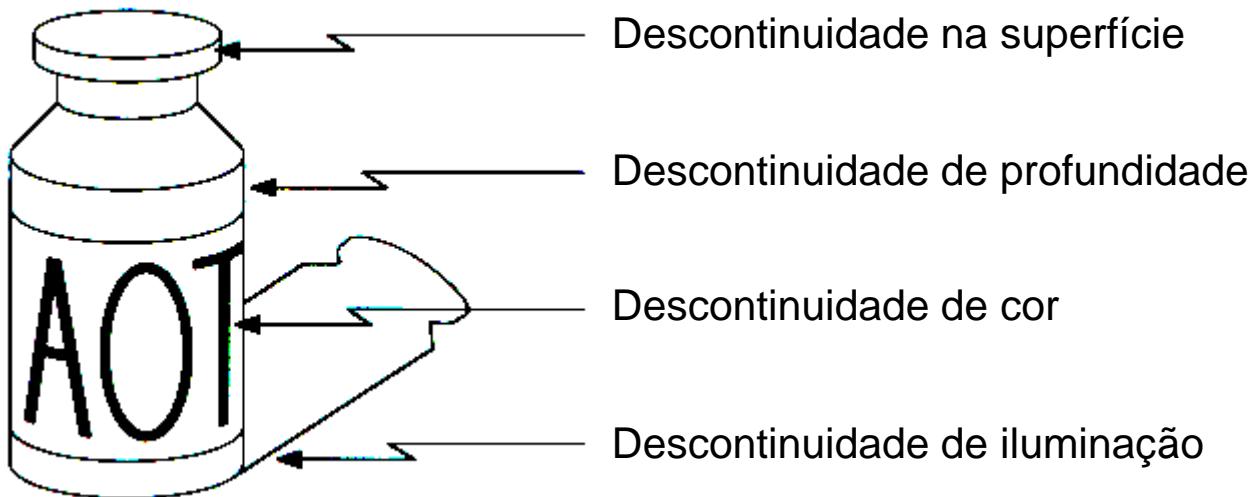
# Edge Detection – Detecção de Bordas

- Converte uma imagem 2D em um conjunto de curvas
  - Extrai características salientes da cena



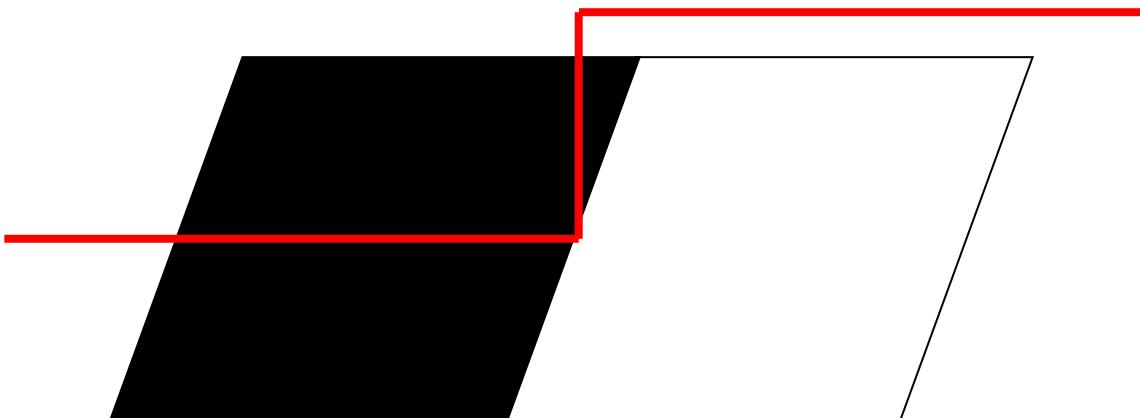
# Edge Detection – Detecção de Bordas

- As bordas são causadas por uma variedade de fatores



# Edge Detection – Detecção de Bordas

- As bordas ocorrem junto com mudanças



# Edge Detection – Detecção de Bordas

- Derivada de Primeira Ordem / Métodos de Gradiente
  - Operadores de Roberts
  - Operadores de Sobel
  - Operadores de Prewitt
- Derivada de Segunda Ordem
  - Laplaciano
  - Laplaciano de Gaussiana (LoG)
  - Diferença de Gaussianas (DoG)
- Detecção de Bordas Ótimo
  - Canny Edge Detection

# Edge Detection – Detecção de Bordas

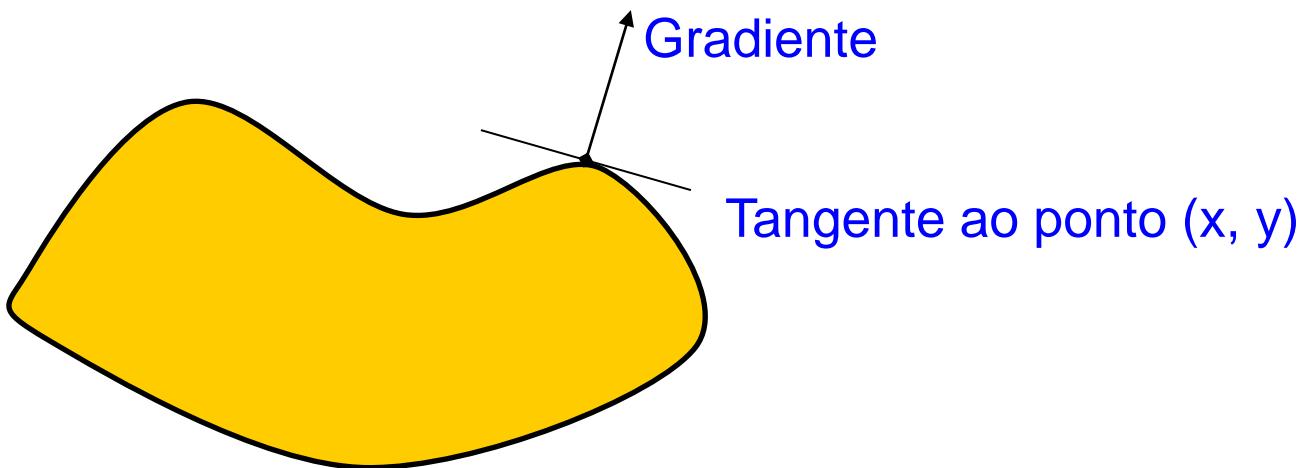
## ● Gradiente

- A detecção de bordas é essencialmente uma operação de identificação de mudanças locais significativas nos tons de uma imagem
- Essas mudanças podem ser descritas através de **derivadas**
- Como uma imagem depende de duas coordenadas espaciais, as bordas podem ser expressas por meio de derivadas parciais
- **Uso do Gradiente:** um vetor cuja direção indica os locais nos quais os níveis de cinza sofram maior variação

# Edge Detection – Detecção de Bordas

- **Gradiente**

- O gradiente tem direção sempre perpendicular à tangente da borda



# Edge Detection – Detecção de Bordas

- **Gradiente**

- Para detecção de bordas, duas medidas são importantes:

- A magnitude do vetor gradiente

- $\nabla f = \text{mag}(\nabla f) = (G_x^2 + G_y^2)^{1/2} = ((\partial f / \partial x)^2 + (\partial f / \partial y)^2)^{1/2}$
    - ou
    - $\nabla f \approx |G_x| + |G_y|$

- A direção do vetor gradiente

- Seja  $\theta(x, y)$  o ângulo da direção do vetor  $\nabla f$  na posição  $(x, y)$ , então  $\theta(x, y) = \tan^{-1}(G_y/G_x)$

# Edge Detection – Detecção de Bordas

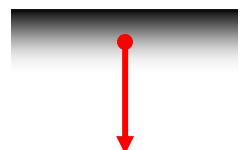
- Gradiente de uma imagem

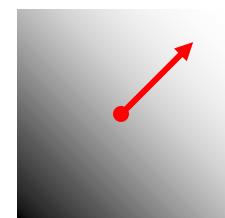
- O gradiente de uma imagem

$$\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

- O gradiente aponta na direção da mudança mais rápida de intensidade


$$\nabla f = \left[ \frac{\partial f}{\partial x}, 0 \right]$$


$$\nabla f = \left[ 0, \frac{\partial f}{\partial y} \right]$$


$$\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

# Edge Detection – Detecção de Bordas

- **Gradiente Discreto**

- Como podemos calcular a derivada de uma imagem  $f(x, y)$ ?
  - Opção 1: reconstruir uma imagem contínua e então tomar o gradiente (não é razoável)
  - Opção 2: tomar a derivada discreta (diferença finita)

$$\frac{\partial f}{\partial x}[x, y] \approx f[x + 1, y] - f[x, y]$$

# Edge Detection – Detecção de Bordas

## ● Gradiente Discreto

- Uma mudança de intensidade pode ser detectada pela diferença entre os valores de pixels adjacentes
- Bordas verticais podem ser detectadas pela diferença horizontal entre pontos, enquanto bordas horizontais podem ser detectadas pela diferença vertical entre pontos adjacentes da imagem

# Edge Detection – Detecção de Bordas

- Gradiente Discreto

$f(x-1,y-1)$	$f(x-1,y)$	$f(x-1,y+1)$
$f(x,y-1)$	$f(x,y)$	$f(x,y+1)$
$f(x+1,y-1)$	$f(x+1,y)$	$f(x+1,y+1)$

- Uma outra abordagem é o cálculo aproximado do gradiente por diferenças cruzadas:

$$\nabla f \approx \sqrt{[f(x,y) - f(x+1,y+1)]^2 + [f(x+1,y) - f(x,y+1)]^2}$$

# Edge Detection – Detecção de Bordas

- Gradiente Discreto

$f(x-1, y-1)$	$f(x-1, y)$	$f(x-1, y+1)$
$f(x, y-1)$	$f(x, y)$	$f(x, y+1)$
$f(x+1, y-1)$	$f(x+1, y)$	$f(x+1, y+1)$

- O uso de valores absolutos:

$$\nabla f \approx |f(x, y) - f(x+1, y+1)| + |f(x+1, y) - f(x, y+1)|$$

# Edge Detection – Detecção de Bordas

- Gradiente Discreto

- Essa aproximação por valores absolutos pode ser implementada por máscaras
- Toma-se o valor absoluto das imagens filtradas pelas duas máscaras e soma-se os resultados:

$$G_x =$$

1	0
0	-1

$$G_y =$$

0	-1
1	0

# Edge Detection – Detecção de Bordas

- Operadores de Sobel (1990)

- Aproxima a magnitude do gradiente como a diferença de valores ponderados dos níveis de cinza como:

- $G_x \approx [f(x - 1, y + 1) + 2f(x, y + 1) + f(x + 1, y + 1)] - [f(x - 1, y - 1) + 2f(x, y - 1) + f(x + 1, y - 1)]$
- $G_y \approx [f(x + 1, y - 1) + 2f(x + 1, y) + f(x + 1, y + 1)] - [f(x - 1, y - 1) + 2f(x - 1, y) + f(x - 1, y + 1)]$

$f(x-1,y-1)$	$f(x-1,y)$	$f(x-1,y+1)$
$f(x,y-1)$	$f(x,y)$	$f(x,y+1)$
$f(x+1,y-1)$	$f(x+1,y)$	$f(x+1,y+1)$

$$G_x = \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$$

$$G_y = \begin{array}{|c|c|c|} \hline -1 & -2 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

# Edge Detection – Detecção de Bordas

- Operadores de Sobel (1990)
  - Operadores mais comumente usados:

$G_x$			$G_y$		
$\frac{1}{p}$	-1	0	1	-1	$2-p$
	$2-p$	0	$p-2$	0	0
	-1	0	1	1	$p-2$

	Prewitt	Sobel	Isotropic
$p$	3	4	$2 + \sqrt{2}$

- A definição padrão dos operadores omite o  $1/p$ 
  - Não faz diferença para a detecção de bordas
  - O termo  $1/p$  é necessário para ter o valor correto do gradiente

# Edge Detection – Detecção de Bordas

- Outros Operadores

$$\Delta_1$$

$$\begin{matrix} 0 & 1 \\ -1 & 0 \end{matrix}$$

$$\Delta_2$$

$$\begin{matrix} 1 & 0 \\ 0 & -1 \end{matrix}$$

$$\Delta_1$$

$$\begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix}$$

$$\Delta_2$$

$$\begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{matrix}$$

(a)

(b)

$$\Delta_1$$

$$\begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$$

$$\Delta_2$$

$$\begin{matrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{matrix}$$

$$\Delta_1$$

$$\begin{matrix} -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \end{matrix}$$

$$\Delta_2$$

$$\begin{matrix} 3 & 3 & 3 & 3 \\ 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \\ -3 & -3 & -3 & -3 \end{matrix}$$

(c)

(d)

- (a): operador de Roberts (b): 3x3 operador de Prewitt  
(c): operador de Sobel (d) 4x4 operador de Prewitt

# Edge Detection – Detecção de Bordas

- Detecção de Bordas
  - Operadores de Gradiente (Sobel)



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

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



# Edge Detection – Detecção de Bordas

- Detecção de Bordas
  - Máscaras Ortogonais (Frei e Chen)



0	1	0
-1	0	-1
0	1	0



# Edge Detection – Detecção de Bordas

- Detecção de Bordas
  - Máscaras Ortogonais (Frei e Chen)



-1	0	1
0	0	0
1	0	-1



# Edge Detection – Detecção de Bordas

- Detecção de Bordas
  - Máscaras Ortogonais (Frei e Chen)



1	-2	1
-2	4	-2
1	-2	1



# Edge Detection – Detecção de Bordas

- Detecção de Bordas
  - Máscaras Ortogonais (Frei e Chen)



-2	1	-2
1	4	1
-2	1	-2



# Operador de Roberts - Exemplo

- Pontos espúrios indicam que o operador é suscetível a ruído



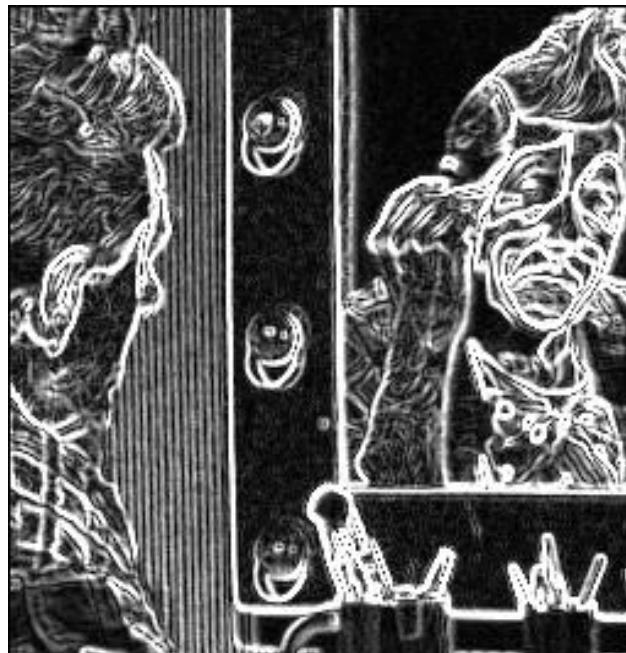
Original



Roberts

# Operador de Sobel - Exemplo

- Compare a saída do operador de Sobel com a de Roberts:
  - O ruído ainda está presente mas com menor intensidade
  - O operador de Roberts perdeu algumas bordas
  - O operador de Sobel detecta bordas mais grossas



Sobel



Roberts

# Edge Detection – Detecção de Bordas

- Derivada de Segunda Ordem
  - Laplaciano



0	1	0
1	-4	1
0	1	0

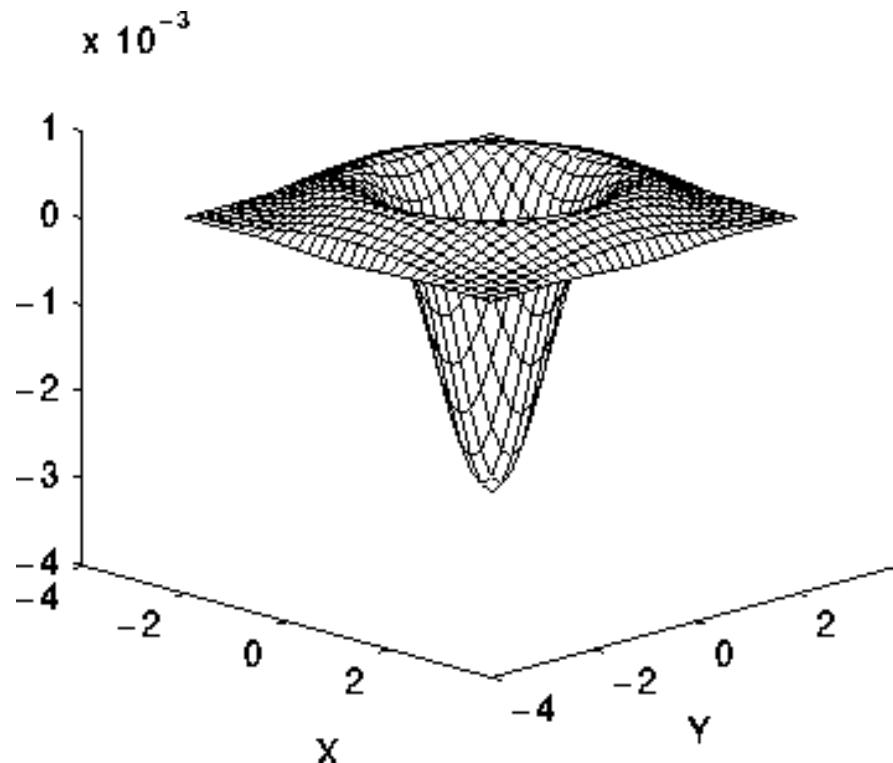


# Edge Detection – Detecção de Bordas

- Derivada de Segunda Ordem
- Laplaciano da Gaussiana
  - Também chamado de **Marr-Hildreth Edge Detector**
- Passos
  - Suavizar a imagem usando filtro Gaussiano
  - Intensificar as bordas usando o operador Laplaciano

# Edge Detection – Detecção de Bordas

- Laplaciano de Gaussiana
  - Também chamado de operador de Chapéu Mexicano



# Edge Detection – Detecção de Bordas

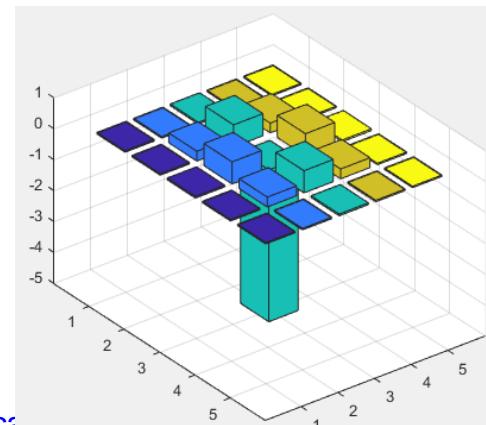
- Derivada de Segunda Ordem
  - Laplaciano de Gaussiana

```
>> h = fspecial ('log')
```

```
h =
```

```
0.0448    0.0468    0.0564    0.0468    0.0448  
0.0468    0.3167    0.7146    0.3167    0.0468  
0.0564    0.7146   -4.9048    0.7146    0.0564  
0.0468    0.3167    0.7146    0.3167    0.0468  
0.0448    0.0468    0.0564    0.0468    0.0448
```

```
>> im_filt = imfilter (im, h);
```



# Edge Detection – Detecção de Bordas

- Derivada de Segunda Ordem
  - Laplaciano de Gaussiana

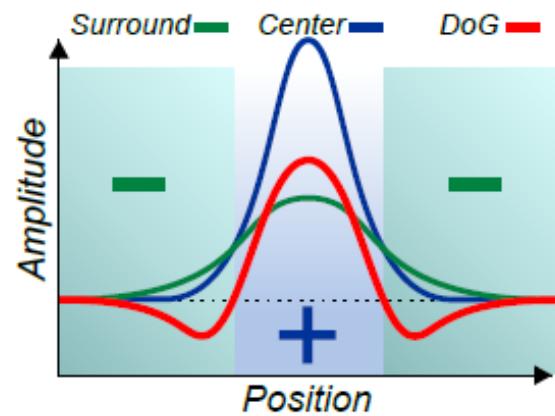


# Edge Detection – Detecção de Bordas

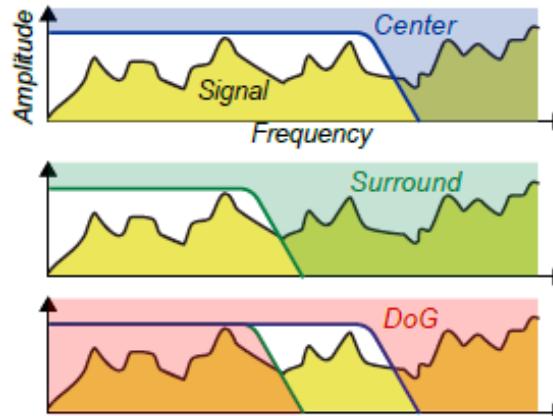
- Derivada de Segunda Ordem
- Diferença de Gaussianas (DoG)



```
>> h1 = fspecial ('gaussian', 5, 0.5);  
>> h2 = fspecial ('gaussian', 5, 0.3);  
>> h = h1 - h2;  
>> im_filt = imfilter (im, h);
```



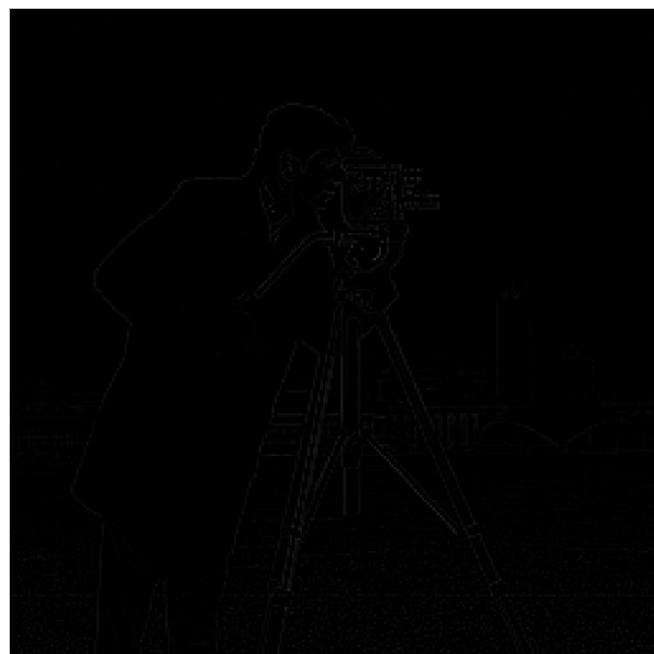
(a) Spatial Plot



(b) Frequency Plots

# Edge Detection – Detecção de Bordas

- Derivada de Segunda Ordem
  - Diferença de Gaussianas



# Edge Detection – Detecção de Bordas

- Derivada de Segunda Ordem
  - Diferença de Gaussianas (*enhanced*)



# Detecção de Borda Ótimo: Canny

- John Canny, 1986
- Um bom detector deve se preocupar com:
  - Taxa de erro: O detector deve responder apenas a bordas, encontrando todas
  - Localização: A distância entre os pixels de borda encontrados e as bordas reais deve ser a menor possível
  - Resposta: O detector não deve identificar múltiplos pixels de borda onde só existir uma borda

# Detecção de Borda Ótimo: Canny

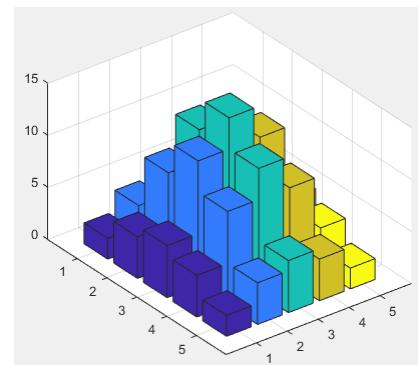
- Algoritmo:
  - 1. Suavização da imagem
    - Objetivo: Eliminar ruído
  - 2. Busca por gradientes
    - Intensifica regiões com maiores derivadas
  - 3. Encontra a direção das bordas
  - 4. Aproxima as direções para 0, 45, 90 ou 135 graus
  - 5. *Non-maximum Suppression*
    - Apenas máximos locais são considerados bordas
  - 6. Limiarização por histerese + Edge Tracking
    - Bordas finais são determinadas, suprimindo as bordas que não estão conectadas a bordas “fortes”

# Detecção de Borda Ótimo: Canny

- Algoritmo (na prática)
  - Suavização
    - Aplicação de um filtro Gaussiano
    - Exemplo: desvio = 1,4:

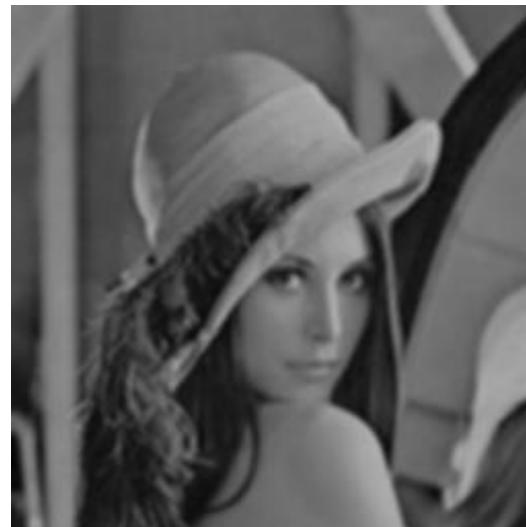
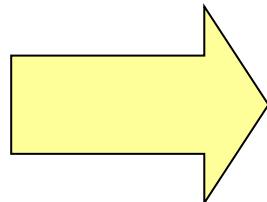
$$B = \frac{1}{159} \cdot \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

Fator de Normalização



# Detecção de Borda Ótimo: Canny

- Algoritmo (na prática)
  - Suavização



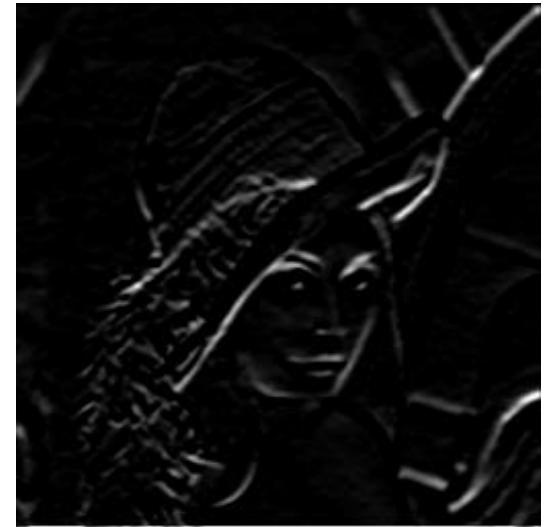
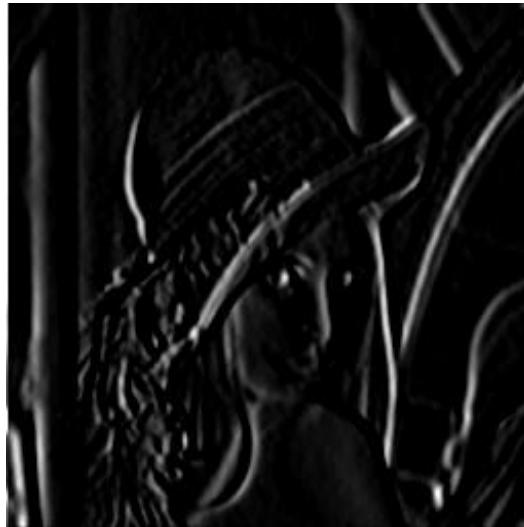
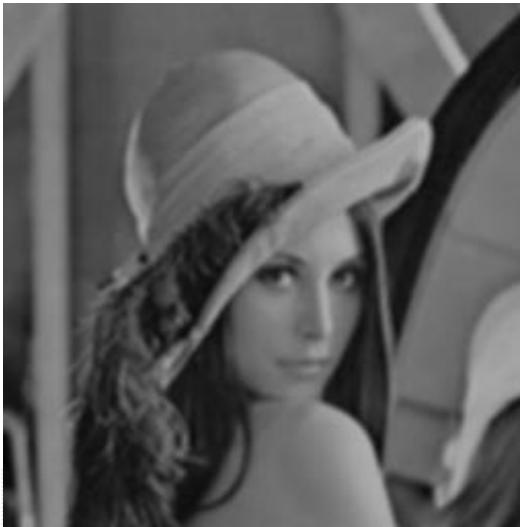
# Detecção de Borda Ótimo: Canny

- Algoritmo (na prática)

- Busca por gradientes

$$K_{GX} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad K_{GY} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Gx    Gy



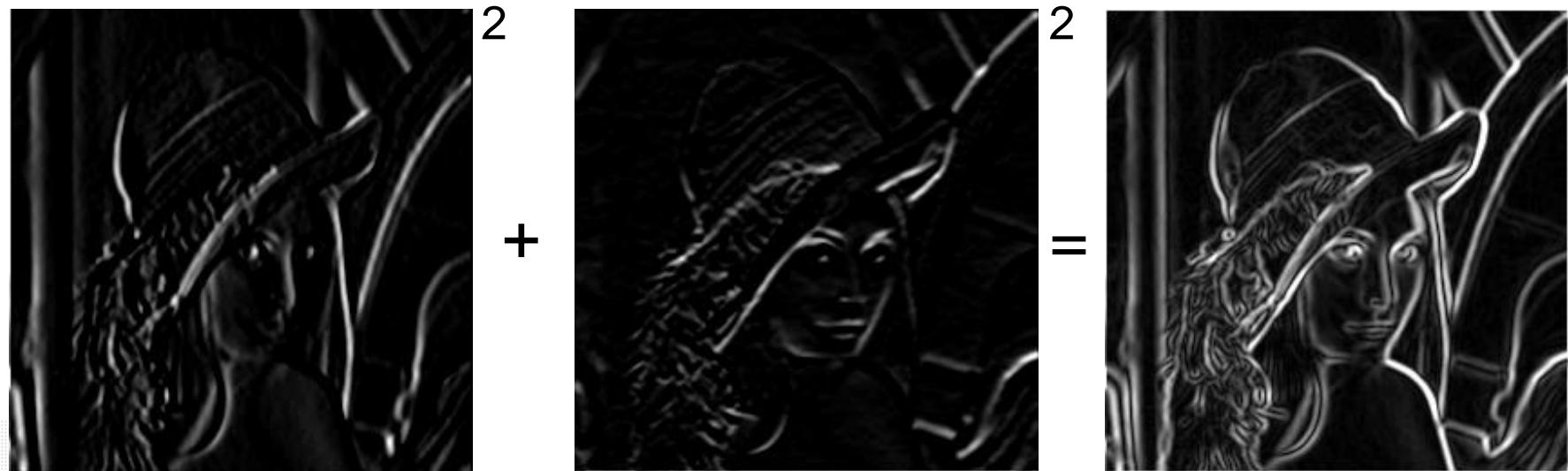
# Detecção de Borda Ótimo: Canny

- Algoritmo (na prática)

- A magnitude do gradiente pode ser encontrada como:

ou  $|G| = \sqrt{G_x^2 + G_y^2}$

$$|G| = |G_x| + |G_y|$$



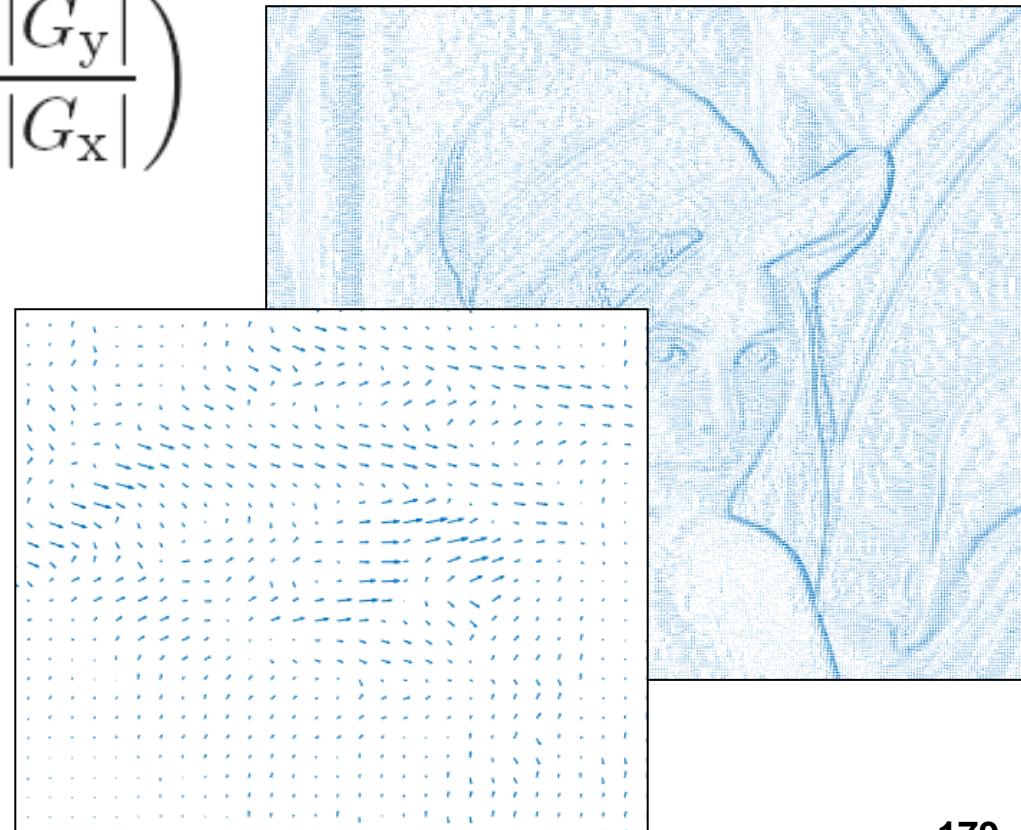
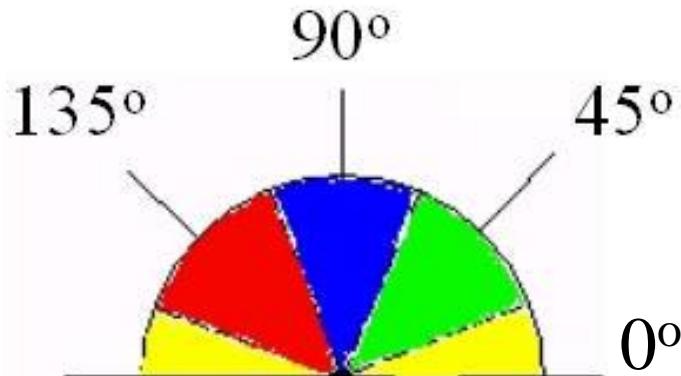
# Detecção de Borda Ótimo: Canny

- Algoritmo (na prática)

- A direção das arestas é calculada e armazenada:

$$\theta = \arctan \left( \frac{|G_y|}{|G_x|} \right)$$

- e aproximada para:



# Detecção de Borda Ótimo: Canny

- Algoritmo (na prática)

- *Non-Maximum Suppression:*

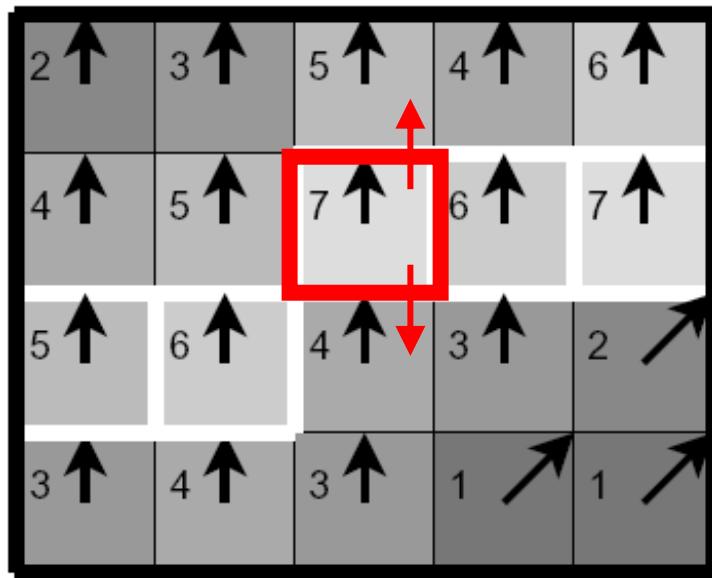
- O propósito deste passo é converter arestas suavizadas na imagem da magnitude em arestas mais “sharp”
    - Isso é feito preservando os máximos locais na imagem do gradiente e removendo o resto
    - A comparação é feita na direção do ângulo
      - Se a direção for de  $0^\circ$ , compara o elemento anterior e posterior da mesma linha. Se o valor do pixel for menor em um dos dois casos, torna-se zero (é suprimido)
      - Faz algo semelhante nas outras direções

# Detecção de Borda Ótimo: Canny

- Algoritmo (na prática)

- Non-Maximum Suppression:

- Exemplo: pixels apontando para o norte são comparados com pixels abaixo e acima



Os que são preservados são os que estão nos quadrados brancos, pois são os maiores (nessa direção); os outros são removidos.

**Exemplo:** compara o 7 com os valores acima e abaixo; como ele é maior permanece.

# Detecção de Borda Ótimo: Canny

- Algoritmo (na prática)

- Hysteresis thresholding

- Dois limiares: um alto e um baixo
    - Qualquer pixel acima do maior limiar é convertido para branco. Os pixels ao redor são então analisados: se seus valores são maiores que o menor limiar, eles são convertidos para branco
      - Um pixel  $(x, y)$  é chamado forte se  $\text{cor}(x, y) > \text{Th}_{\text{alto}}$
      - Um pixel  $(x, y)$  é chamado fraco se  $\text{cor}(x, y) \leq \text{Th}_{\text{baixo}}$
      - Todos os outros pixels são chamados de candidatos

# Detecção de Borda Ótimo: Canny

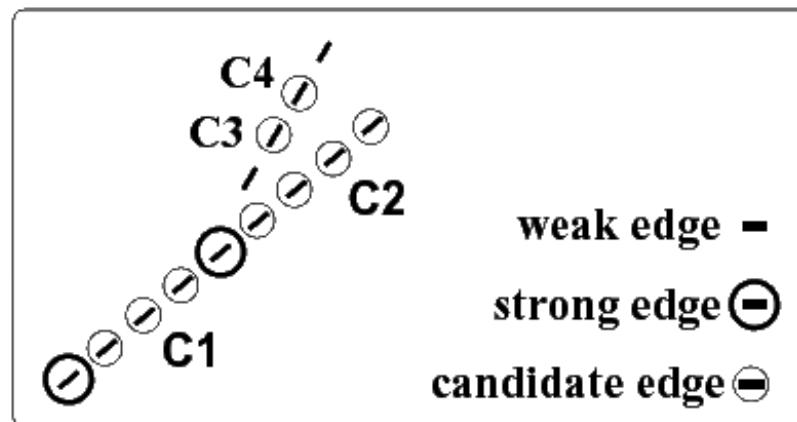
- Algoritmo (na prática)

- Hysteresis thresholding + Edge Tracking

- Em cada posição  $(x, y)$ , descarta o pixel  $(x, y)$  se ele é fraco; permanece na imagem de saída se for forte
    - Se um pixel é um candidato, segue a cadeia de máximos locais conectados em ambas as direções na direção da aresta, enquanto  $\text{cor}(i, j) > \text{Th}_{\text{baixo}}$
    - Se um pixel candidato inicial está conectado a um pixel forte, ele não é suprimido; do contrário, é

# Detecção de Borda Ótimo: Canny

- Algoritmo (na prática)
  - Hysteresis thresholding + Edge Tracking



As arestas candidatas C1 e C2 são preservadas na saída, enquanto as arestas C3 e C4 são suprimidas.

# Detecção de Borda Ótimo: Canny

- Exemplo: Variações nos parâmetros



Gaussiana:  $5 \times 5$ ,  $\sigma = 1.4$

Thresholding:

$Th_{baixo} = 10\%$  do máximo

$Th_{alto} = 30\%$  do máximo

# Detecção de Borda Ótimo: Canny

- Exemplo: Variações nos parâmetros



Gaussiana:  $5 \times 5$ ,  $\sigma = 5$

Thresholding:

$Th_{baixo} = 10\%$  do máximo

$Th_{alto} = 30\%$  do máximo

# Detecção de Borda Ótimo: Canny

- Exemplo: Variações nos parâmetros



Gaussiana:  $5 \times 5$ ,  $\sigma = 1.4$

Thresholding:

$Th_{baixo} = 10\%$  do máximo

$Th_{alto} = 20\%$  do máximo

# Detecção de Borda Ótimo: Canny

- Exemplo: Variações nos parâmetros



Gaussiana:  $5 \times 5$ ,  $\sigma = 1.4$

Thresholding:

$Th_{baixo} = 5\%$  do máximo

$Th_{alto} = 10\%$  do máximo

# Detecção de Borda Ótimo: Canny

- Exemplo: Variações nos parâmetros



Gaussiana:  $5 \times 5$ ,  $\sigma = 0.5$

Thresholding:

$Th_{baixo} = 5\%$  do máximo

$Th_{alto} = 10\%$  do máximo

# Detecção de Borda Ótimo: Canny

- Exemplo: Variações nos parâmetros



Gaussiana:  $5 \times 5$ ,  $\sigma = 0.5$   
Thresholding:  
 $Th_{baixo} = 10\%$  do máximo  
 $Th_{alto} = 30\%$  do máximo

# Detecção de Borda Ótimo: Canny

- Exemplo: Variações nos parâmetros



Gaussiana:  $5 \times 5$ ,  $\sigma = 0.5$   
Thresholding:  
 $Th_{baixo} = 10\%$  do máximo  
 $Th_{alto} = 20\%$  do máximo

# Detecção de Borda Ótimo: Canny

- Exemplo:



Gaussiana:  $5 \times 5$ ,  $\sigma = 0.5$   
Thresholding:  
 $Th_{baixo} = 10\%$  do máximo  
 $Th_{alto} = 20\%$  do máximo

# Detecção de Borda Ótimo: Canny

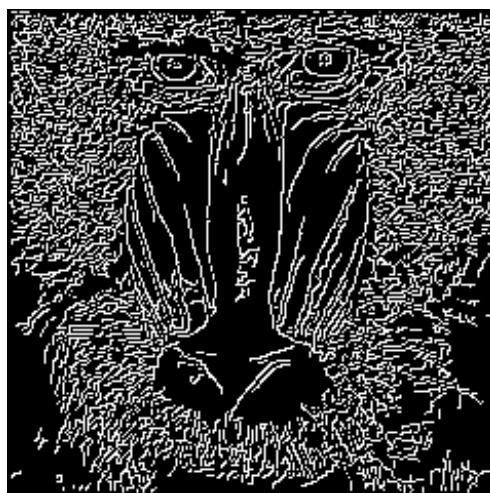
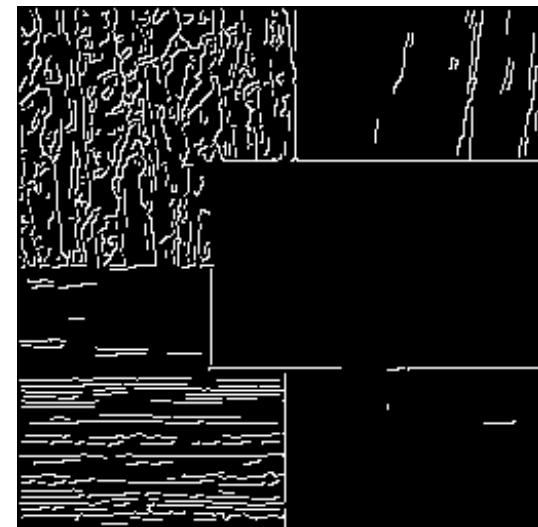
- Exemplo:

Gaussiana: 5x5,  $\sigma = 0.5$

Thresholding:

$Th_{baixo} = 10\%$  do máximo

$Th_{alto} = 20\%$  do máximo



# Detecção de Borda



# Detecção de Borda



Prewitt

# Detecção de Borda



Roberts

# Detecção de Borda



Sobel

# Detecção de Borda



Canny

# Detecção de Borda



DoG

```
>> h1 = fspecial ('gaussian', [5 5], 0.5);
>> h2 = fspecial ('gaussian', [5 5], 0.7);
>> h3 = h2 - h1;
>> im2 = imfilter (im, h3);
```

# Detecção de Bordas

- Recomendações de leitura:

- An overview of edge and object contour detection. Daipeng Yang, Bo Peng, Zaid Al-Huda, Asad Malik, Donghai Zhai, Neurocomputing, 2022
- Survey of Image Edge Detection, Rui Sun, Tao Lei, Qi Chen, Zexuan Wang, Xiaogang Du, Weiqiang Zhao and Asoke K. Nandi, Frontiers in Signal Processing, 2022
- Recent advances on image edge detection: A comprehensive review, Junfeng Jing, Shenjuan Liu, Gang Wang, Weichuan Zhang, Changming Sun, Neurocomputing, 2022
- Edge detection on noisy images using Prewitt operator and fractional order differentiation, Saeed Balochian & Hossein Baloochian, Multimedia Tools and Applications, 2022
- EDTD: Edge Detection with Transformer, Mengyang Pu, Yaping Huang, Yuming Liu, Qingji Guan, Haibin Ling, CVPR, 2022
- Color edge detection by learning classification network with anisotropic directional derivative matrices, Ou Li, Peng-Lang Shui, Pattern Recognition, 2021

# Links - Colab

- Filtragem no Colab:

- Domínio espacial:

- <https://colab.research.google.com/drive/1wN6m-XZnLtrNAjmZs4-YuFHqBhtsyTx0?authuser=1>

# Links – Colab

- Detecção de Bordas no Colab:
  - Canny:
    - <https://colab.research.google.com/drive/1yKBMshxy2ZmrvrNjB-LMKVUIpUv8keml#scrollTo=HIK9amGCaxZj>