



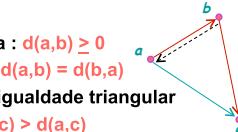
## Atributos Locais

### Conceito de Localidade

- Atributo local → do original *Local Feature*
  - *Feature* → detalhe peculiar, característica, propriedade ou atributo característico ou típico, detalhe importante
  - *Local feature* → enfatiza o aspecto de localidade
- Localidade → associado a uma **vizinhança** do pixel
  - É o conjunto dos pixels que se encontram a uma dada **distância** do pixel considerado

### Distância

- Distância → é um número que caracteriza a separação entre dois objetos
  - Seja  $S = \{\text{conjunto de objetos}\}$
  - Seja  $d: S \times S \rightarrow \mathbb{R}_+$
- Deve satisfazer os seguintes requisitos:
  - Sejam  $a, b, c \in S$ 
    - Deve ser não-negativa :  $d(a,b) \geq 0$
    - Deve ser comutativa:  $d(a,b) = d(b,a)$
    - Deve satisfazer à desigualdade triangular  
$$d(a,b) + d(b,c) \geq d(a,c)$$



### Distância x Topologia $S \times S$ VS $\mathbb{R}_+ \times \mathbb{R}_+$

- A distância foi definida em  $\mathbb{R}_+$ 
  - Portanto obedece a uma topologia contínua
  - Como ficam as distâncias entre objetos de um espaço discreto ?
- Há duas categorias de distâncias
  - As intrínsecas
    - Obedecem à topologia e curvatura do espaço dos objetos
  - As extrínsecas
    - Obedecem à topologia e curvatura do espaço das representações

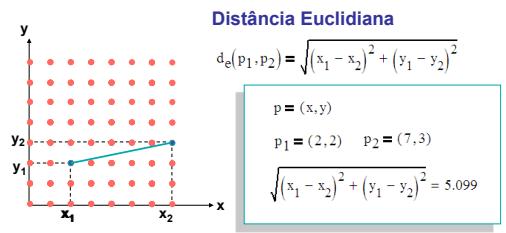
### Distância entre pixels

- Distâncias intrínsecas (espaço dos objetos)
  - $d: \text{função(caminho*) entre os pixels} \rightarrow \mathbb{Z}_+$   
\*caminho digital conexo
- Distâncias extrínsecas (espaço das representações)
  - $d: \text{função(coordenadas dos pixels)} \rightarrow \mathbb{R}_+$



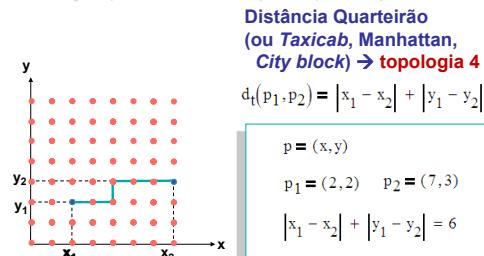
## Distância entre pixels

- Distâncias extrínsecas (espaço das representações)  
d: função(coordenadas dos pixels)  $\rightarrow \mathbb{R}_+$



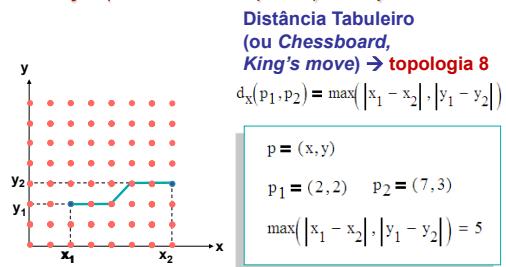
## Distância entre pixels

- Distâncias intrínsecas (espaço dos objetos)  
d: função(caminho entre pixels)  $\rightarrow \mathbb{Z}_+$



## Distância entre pixels

- Distâncias intrínsecas (espaço dos objetos)  
d: função(caminho entre pixels)  $\rightarrow \mathbb{Z}_+$



## Intrínsecas x Extrínsecas

- Qual delas devemos usar ?

- Depende do que se está representando
  - distância entre pontos do mundo físico representados pelos pixels  $\rightarrow$  distâncias extrínsecas
  - Distância entre pixels para se usar em processos computacionais  $\rightarrow$  distâncias intrínsecas

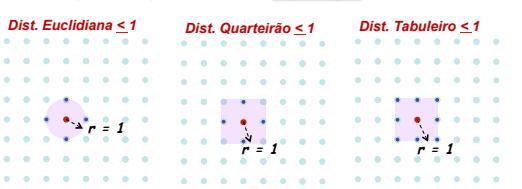


- Há outras distâncias que podem ser definidas, desde que obejam os requisitos: positividade, comutatividade, desigualdade triangular



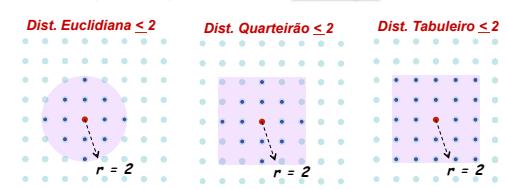
## Vizinhança de um pixel

- É o conjunto de pixels que se encontra a uma dada distância do mesmo
  - Depende da definição de distância usada
  - Depende, portanto, da topologia usada



## Vizinhança de um pixel

- É o conjunto de pixels que se encontra a uma dada distância do mesmo
  - Depende da definição de distância usada
  - Depende, portanto, da topologia usada



# Labeling

## Labeling

- Tarefa comum em processamento de imagens
- Atribui a cada componente conexa um rótulo  $\mathcal{L}$  diferente ( $\mathcal{L}$  pertencente aos naturais, diferente de 0)
- O **número de componentes** é o valor do maior rótulo
- Feita usualmente após a limiarização, ou após limiarização e filtragem

## Labeling - Algoritmo

- Atribui um rótulo em ordem crescente para cada componente conexa da imagem
- **Input:** Imagem binária  $f$ , K (conectividade)
- **Output:** Imagem em níveis de cinza  $g$ 
  - Como o número de componentes conexas pode ser muito grande, é usual que a saída seja unsigned int.

## Labeling - Algoritmo

- $\forall p \in E$ , percorrendo-se  $E$  em sentido raster,
- se  $f(p) = 1$  e  $g(p) = 0$ , atribua a  $g(p)$  o valor do próximo rótulo,  $I$ , e ponha  $p$  na fila
- enquanto a fila não estiver vazia,
- tire um elemento da fila, seja  $r$  tal elemento
- Para cada  $s$  de  $N_K(r)$ , se  $f(s) = 1$  e  $g(s) = 0$ ,
- faça  $g(s) = I$  e ponha  $s$  na fila

## Algoritmo de rotulação – $K = 4$

<b>0</b>	0 0 1 1 1	0 0 0 0 0
0 0 1 1 0 1	0 0 0 0 0	0 0 0 0 0
0 1 1 0 1 1	0 0 0 0 0	0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0	0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0	0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0	0 0 0 0 0
Input		Output

## Algoritmo de rotulação – $K = 4$

<b>0 0</b>	0 1 1 1	0 0 0 0 0 0
0 0 1 1 0 1	0 0 0 0 0	0 0 0 0 0 0
0 1 1 0 1 1	0 0 0 0 0	0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0	0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0	0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0	0 0 0 0 0 0
Input		Output

### Algoritmo de rotulação – K = 4

<b>0 0 0</b>	1 1 1	0 0 0 0 0 0
0 0 1	1 0 1	0 0 0 0 0 0
0 1 1	0 1 1	0 0 0 0 0 0
1 1 0	1 1 0	0 0 0 0 0 0
1 0 1	1 0 0	0 0 0 0 0 0
1 1 1	0 0 0	0 0 0 0 0 0

Input

Output

### Algoritmo de rotulação – K = 4

<b>0 0 0 1</b>	1 1 1	0 0 0 <b>0</b> 0 0
0 0 1	1 0 1	0 0 0 0 0 0
0 1 1	0 1 1	0 0 0 0 0 0
1 1 0	1 1 0	0 0 0 0 0 0
1 0 1	1 0 0	0 0 0 0 0 0
1 1 1	0 0 0	0 0 0 0 0 0

Input

Output

### Labeling - Algoritmo

- $\forall p \in E$ , percorrendo-se  $E$  em sentido raster,
  - se  $f(p) = 1$  e  $g(p) = 0$ , atribua a  $g(p)$  o valor do próximo rótulo,  $l$ , e ponha  $p$  na fila
  - enquanto a fila não estiver vazia,
  - tire um elemento da fila, seja  $r$  tal elemento
  - Para cada  $s$  de  $N_K(r)$ , se  $f(s) = 1$  e  $g(s) = 0$ ,
  - faça  $g(s) = l$  e ponha  $s$  na fila

### Algoritmo de rotulação – K = 4

<b>0 0 0 1</b>	1 1 1	0 0 0 <b>1</b> 0 0
0 0 1	1 0 1	0 0 0 0 0 0
0 1 1	0 1 1	0 0 0 0 0 0
1 1 0	1 1 0	0 0 0 0 0 0
1 0 1	1 0 0	0 0 0 0 0 0
1 1 1	0 0 0	0 0 0 0 0 0

Input

Output

### Labeling - Algoritmo

- $\forall p \in E$ , percorrendo-se  $E$  em sentido raster,
  - se  $f(p) = 1$  e  $g(p) = 0$ , atribua a  $g(p)$  o valor do próximo rótulo,  $l$ , e ponha  $p$  na fila
  - enquanto a fila não estiver vazia,
  - tire um elemento da fila, seja  $r$  tal elemento
  - Para cada  $s$  de  $N_K(r)$ , se  $f(s) = 1$  e  $g(s) = 0$ ,
  - faça  $g(s) = l$  e ponha  $s$  na fila

### Algoritmo de rotulação – K = 4

<b>0 0 0 1</b>	1 1 1	0 0 0 <b>1</b> 0 0
0 0 1	1 0 1	0 0 0 0 0 0
0 1 1	0 1 1	0 0 0 0 0 0
1 1 0	1 1 0	0 0 0 0 0 0
1 0 1	1 0 0	0 0 0 0 0 0
1 1 1	0 0 0	0 0 0 0 0 0

Input

Output

### Algoritmo de rotulação – K = 4

<b>0 0 0 1 1 1</b>	<b>0 0 0 1 1 0</b>
0 0 1 1 0 1	0 0 0 0 0 0
0 1 1 0 1 1	0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0 0

Input

Output

### Algoritmo de rotulação – K = 4

<b>0 0 0 1 1 1</b>	<b>0 0 0 1 1 0</b>
0 0 1 1 0 1	0 0 0 0 0 0
0 1 1 0 1 1	0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0 0

Input

Output

### Algoritmo de rotulação – K = 4

<b>0 0 0 1 1 1</b>	<b>0 0 0 1 1 0</b>
0 0 1 1 0 1	0 0 0 1 0 0
0 1 1 0 1 1	0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0 0

Input

Output

### Algoritmo de rotulação – K = 4

<b>0 0 0 1 1 1</b>	<b>0 0 0 1 1 0</b>
0 0 1 1 0 1	0 0 0 1 0 0
0 1 1 0 1 1	0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0 0

Input

Output

### Algoritmo de rotulação – K = 4

<b>0 0 0 1 1 1</b>	<b>0 0 0 1 1 1</b>
0 0 1 1 0 1	0 0 0 1 0 0
0 1 1 0 1 1	0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0 0

Input

Output

### Algoritmo de rotulação – K = 4

<b>0 0 0 1 1 1</b>	<b>0 0 0 1 1 1</b>
0 0 1 1 0 1	0 0 0 1 0 0
0 1 1 0 1 1	0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0 0

Input

Output

### Algoritmo de rotulação – K = 4

<b>Input</b>	<b>Output</b>
0 0 0 1 1 1	0 0 0 1 1 1
0 0 1 1 0 1	0 0 0 1 0 0
0 1 1 0 1 1	0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0 0

### Algoritmo de rotulação – K = 4

<b>Input</b>	<b>Output</b>
0 0 0 1 1 1	0 0 0 1 1 1
0 0 1 1 0 1	0 0 0 1 0 0
0 1 1 0 1 1	0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0 0

### Algoritmo de rotulação – K = 4

<b>Input</b>	<b>Output</b>
0 0 0 1 1 1	0 0 0 1 1 1
0 0 1 1 0 1	0 0 0 1 0 0
0 1 1 0 1 1	0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0 0

### Algoritmo de rotulação – K = 4

<b>Input</b>	<b>Output</b>
0 0 0 1 1 1	0 0 0 1 1 1
0 0 1 1 0 1	0 0 1 1 0 0
0 1 1 0 1 1	0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0 0

### Algoritmo de rotulação – K = 4

- Muitos passos depois .....

### Algoritmo de rotulação – K = 4

<b>Input</b>	<b>Output</b>
0 0 0 1 1 1	0 0 0 1 1 1
0 0 1 1 0 1	0 0 1 1 0 1
0 1 1 0 1 1	0 1 1 0 1 1
1 1 0 1 1 0	1 1 0 1 1 0
1 0 1 1 0 0	1 0 1 1 0 0
1 1 1 0 0 0	1 1 1 0 0 0

### Labeling - Algoritmo

- $\forall p \in E$ , percorrendo-se  $E$  em sentido raster,
  - se  $f(p) = 1$  e  $g(p) = 0$ , atribua a  $g(p)$  o valor do próximo rótulo,  $l$ , e ponha  $p$  na fila
  - enquanto a fila não estiver vazia,
    - tire um elemento da fila, seja  $r$  tal elemento
    - Para cada  $s$  de  $N_K(r)$ , se  $f(s) = 1$  e  $g(s) = 0$ ,
      - faça  $g(s) = l$  e ponha  $s$  na fila

### Algoritmo de rotulação – $K = 4$

<b>0 0 0 1 1 1</b>	<b>0 0 0 1 1 1</b>
0 0 1 1 0 1	0 0 1 1 0 1
0 1 1 0 1 1	0 1 1 0 1 1
1 1 0 1 1 0	1 1 0 1 1 0
1 0 1 1 0 0	1 0 1 1 0 0
1 1 1 0 0 0	1 1 1 0 0 0

Input                      Output

### Algoritmo de rotulação – $K = 4$

<b>0 0 0 1 1 1</b>	<b>0 0 0 1 1 1</b>
0 0 1 1 0 1	0 0 1 1 0 1
0 1 1 0 1 1	0 1 1 0 1 1
1 1 0 1 1 0	1 1 0 1 1 0
1 0 1 1 0 0	1 0 1 1 0 0
1 1 1 0 0 0	1 1 1 0 0 0

Input                      Output

### Algoritmo de rotulação – $K = 4$

<b>0 0 0 1 1 1</b>	<b>0 0 0 1 1 1</b>
0 0 1 1 0 1	0 0 1 1 0 1
0 1 1 0 1 1	0 1 1 0 1 1
1 1 0 1 1 0	1 1 0 1 1 0
1 0 1 1 0 0	1 0 1 1 0 0
1 1 1 0 0 0	1 1 1 0 0 0

Input                      Output

### Algoritmo de rotulação – $K = 4$

1	2	62	63	58	
6	3	4	64	57	56
7	5	8	66	52	55
9	61				
12	10	11	49	47	48
13	27	28	46		
15	14	29	45	44	
18	16	32	30	31	43
19	17	26	33	41	42
20	24	25	34	35	38
21	22	23	37	36	40

4  
↑  
3 ← p → 1  
↓  
2

### Algoritmo de rotulação

- Este algoritmo é um bom exemplo de uso de estruturas de dados para melhorar a performance
- Existem outros algoritmos, por exemplo, o algoritmo de Rosenfeld e Pfaltz, que faz duas varreduras na imagem e dispensa o uso da fila.

## Transformada distância

### Transformada Distância

- Operador que atribui a cada ponto  $p$  de um objeto, a menor distância de  $p$  ao complemento do objeto
- Defini-se a distância de um ponto  $p$  a um conjunto  $X$  como:

$$d(p, X) = \min \{d(p, q), q \in X\}$$

### Transformada Distância – $d_{\text{chess}}$

0 0 0 0 0 0	0 0 0 0 0 0
0 0 1 1 1 0	0 0 1 1 1 0
0 1 1 1 1 0	0 1 1 1 2 → 1 0
0 1 1 1 1 0	0 1 1 1 1 0
0 0 1 1 0 0	0 0 1 1 0 0
0 0 0 0 0 0	0 0 0 0 0 0

### Transformada Distância – $d_{\text{taxicab}}$

0 0 0 0 0 0	0 0 0 0 0 0
0 0 1 1 1 0	0 0 1 1 1 0
0 1 1 1 1 0	0 1 2 2 → 1 0
0 1 1 1 1 0	0 1 2 2 1 0
0 0 1 1 0 0	0 0 1 1 0 0
0 0 0 0 0 0	0 0 0 0 0 0

### Transformada Distância

Formalmente, leva uma imagem binária em uma imagem em níveis de cinza

$$T_d : \wp(E) \rightarrow [0, k]^E$$

### Transformada Distância

$$T_d : \{0,1\}^E \rightarrow [0, k]^E$$

$$T_d(f)(p) = d(p, \{q \in E : f(q) = 0\})$$

### Algoritmo: transformada distância

- Você pensou em um algoritmo eficiente para a transformada distância?

### 4-,8-Vizinhos

- Lembrando:

	y-1	y	y+1
x-1		N <sub>2</sub> (p)	
x	N <sub>4</sub> (p)	p	N <sub>8</sub> (p)
x+1		N <sub>6</sub> (p)	

N<sub>4</sub>(p)

	y-1	y	y+1
x-1	N <sub>3</sub> (p)	N <sub>2</sub> (p)	N <sub>1</sub> (p)
x	N <sub>4</sub> (p)	p	N <sub>8</sub> (p)
x+1	N <sub>5</sub> (p)	N <sub>6</sub> (p)	N <sub>7</sub> (p)

N<sub>8</sub>(p)

### 4-,8-Vizinhos

- Vizinhos anteriores e posteriores

	y-1	y	y+1
x-1		N <sub>2</sub> (p)	
x	N <sub>4</sub> (p)	p	N <sub>8</sub> (p)
x+1		N <sub>6</sub> (p)	

N<sub>4</sub><sup>←</sup>(p) N<sub>4</sub><sup>→</sup>(p)

	y-1	y	y+1
x-1	N <sub>3</sub> (p)	N <sub>2</sub> (p)	N <sub>1</sub> (p)
x	N <sub>4</sub> (p)	p	N <sub>8</sub> (p)
x+1	N <sub>5</sub> (p)	N <sub>6</sub> (p)	N <sub>7</sub> (p)

N<sub>8</sub><sup>←</sup>(p) N<sub>8</sub><sup>→</sup>(p)

### Algoritmo: transformada distância

Entrada: Imagem binária f, conectividade G  
Saída: Imagem níveis de cinza f

Percorra E, em sentido raster,  $\forall p \in E$   
 se  $f(p) = 1$ ,  $f(p) = 1 + \min\{f(q) : q \in N_G^{\leftarrow}(p)\}$   
 Percorra E, em sentido anti-raster  $\forall p \in E$   
 se  $f(p) \neq 0$   
 $f(p) = \min\{f(p), 1 + \min\{f(q) : q \in N_G^{\rightarrow}(p)\}\}$

### Simulação – G = 4

0	0	0	0	0	0	0
0	0	1	1	1	0	
0	1	1	1	1	0	
0	1	1	1	1	0	
0	1	1	1	0	0	
0	0	0	0	0	0	

### Simulação – G = 4

0	0	0	0	0	0	0
0	0	1	1	1	0	
0	1	1	1	1	0	
0	1	1	1	1	0	
0	1	1	1	0	0	
0	0	0	0	0	0	

Simulação – G = 4

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

Simulação – G = 4

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

Simulação – G = 4

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

Simulação – G = 4

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

Simulação – G = 4

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

Simulação – G = 4

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

### Simulação – G = 4

0 0 0 0 0 0 0	0 0 0 0 0 0 0
0 0 1 1 1 1 0	0 0 1 1 1 1 0
0 1 1 1 1 1 0	0 1 2 2 2 2 0
0 1 1 1 1 1 0	0 1 0 0 0 0 0
0 1 1 1 1 0 0	0 0 0 0 0 0 0
0 0 0 0 0 0 0	0 0 0 0 0 0 0

### Simulação – G = 4

0 0 0 0 0 0 0	0 0 0 0 0 0 0
0 0 1 1 1 1 0	0 0 1 1 1 1 0
0 1 1 1 1 1 0	0 1 2 2 2 2 0
0 1 1 1 1 1 0	0 1 2 0 0 0 0
0 1 1 1 1 0 0	0 0 0 0 0 0 0
0 0 0 0 0 0 0	0 0 0 0 0 0 0

### Simulação – G = 4

0 0 0 0 0 0 0	0 0 0 0 0 0 0
0 0 1 1 1 1 0	0 0 1 1 1 1 0
0 1 1 1 1 1 0	0 1 2 2 2 2 0
0 1 1 1 1 1 0	0 1 2 2 0 0 0
0 1 1 1 1 0 0	0 0 0 0 0 0 0
0 0 0 0 0 0 0	0 0 0 0 0 0 0

### Simulação – G = 4

0 0 0 0 0 0 0	0 0 0 0 0 0 0
0 0 1 1 1 1 0	0 0 1 1 1 1 0
0 1 1 1 1 1 0	0 1 2 2 2 2 0
0 1 2 2 2 1 0	0 1 2 2 2 2 0
0 1 1 1 1 0 0	0 0 0 0 0 0 0
0 0 0 0 0 0 0	0 0 0 0 0 0 0

### Simulação – G = 4

0 0 0 0 0 0 0	0 0 0 0 0 0 0
0 0 1 1 1 1 0	0 0 1 1 1 1 0
0 1 1 1 1 1 0	0 1 2 2 2 2 0
0 1 2 2 2 2 0	0 1 2 2 2 2 0
0 1 1 1 1 0 0	0 0 0 0 0 0 0
0 0 0 0 0 0 0	0 0 0 0 0 0 0

### Simulação – G = 4

0 0 0 0 0 0 0	0 0 0 0 0 0 0
0 0 1 1 1 1 0	0 0 1 1 1 1 0
0 1 1 1 1 1 0	0 1 2 2 2 2 0
0 1 2 2 2 2 0	0 1 2 2 2 2 0
0 1 1 1 1 0 0	0 0 0 0 0 0 0
0 0 0 0 0 0 0	0 0 0 0 0 0 0

### Simulação – G = 4

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

### Simulação – G = 4

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

### Algoritmo: transformada distância

Entrada: Imagem binária f, conectividade G  
 Saída: Imagem níveis de cinza f

Percorra E, em sentido raster,  $\forall p \in E$   
 se  $f(p) = 1$ ,  $f(p) = 1 + \min\{f(q) : q \in N_G^+(p)\}$

Percorra E, em sentido anti-raster  $\forall p \in E$   
 se  $f(p) \neq 0$   
 $f(p) = \min\{f(p), 1 + \min\{f(q) : q \in N_G^-(p)\}\}$

### Simulação – G = 4

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

### Simulação – G = 4

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

### Simulação – G = 4

0 0 0 0 0 0 0	0 0 0 0 0 0 0
0 0 1 1 1 1 0	0 0 1 1 1 1 0
0 1 2 2 2 2 0	0 1 2 2 2 2 0
0 1 2 2 2 2 0	0 1 2 2 2 2 0
0 1 2 1 1 0 0	0 1 1 1 0 0
0 0 0 0 0 0 0	0 0 0 0 0 0 0

### Simulação – G = 4

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	1	1	0	0

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	1	1	0	0

### Simulação – G = 4

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	1	1	0	0

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	1	1	0	0

### Simulação – G = 4

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	1	1	0	0

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	1	1	0	0

### Simulação – G = 4

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	1	1	0	0

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	1	1	0	0

### Simulação – G = 4

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	2	2	2	0

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	2	2	2	0

### Simulação – G = 4

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	2	2	2	0

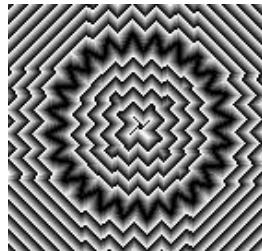
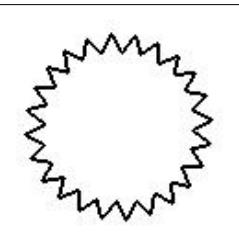
0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	2	0
0	1	2	2	2	0
0	1	2	2	2	0

Alguns slides depois...

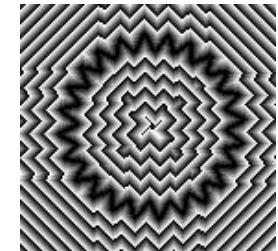
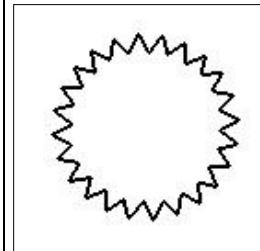
### Simulação – $G = 4$

0	0	0	0	0	0
0	0	1	1	1	0
0	1	2	2	1	0
0	1	2	2	1	0
0	1	1	1	0	0
0	0	0	0	0	0

### Transformada Distância

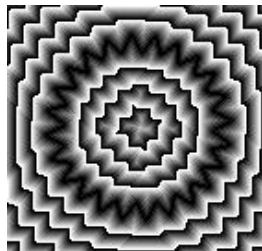
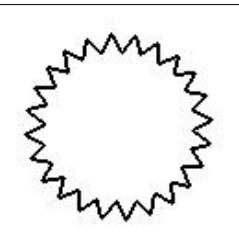


### Transformada Distância



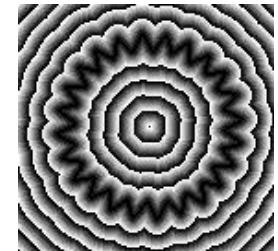
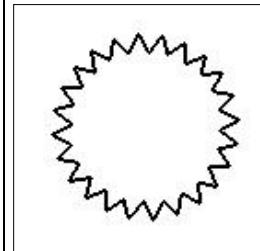
Transformação distância,  $d_r$

### Transformada Distância



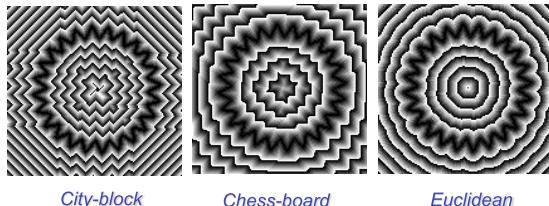
Transformação distância,  $d_x$

### Transformada Distância



Transformação distância,  $d_e$

## Transformada Distância



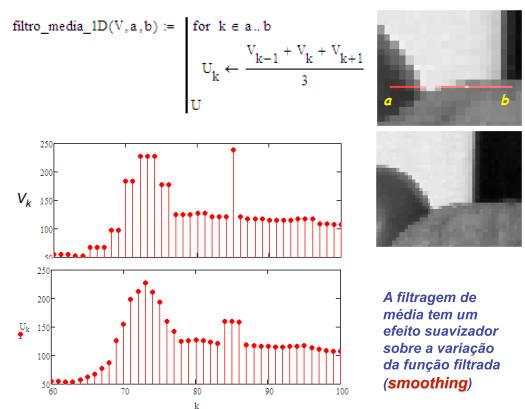
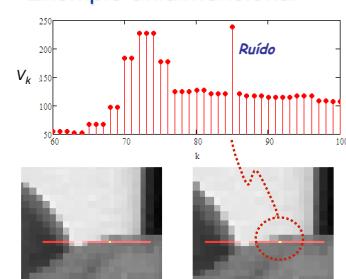
## Filtragem de média móvel

### Filtragem de média

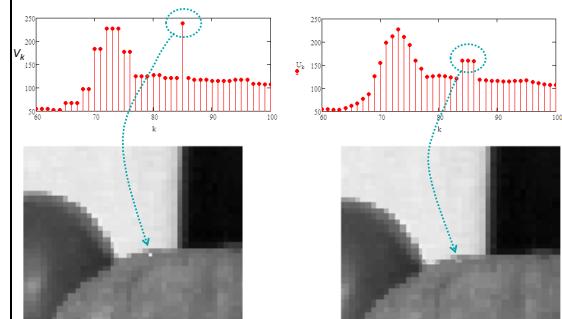
- Filtragem pode ser usada para aumentar a uniformidade das regiões
  - Eliminando irregularidades
  - Diminuindo o efeito do ruído
- A idéia consiste em substituir o valor de cada pixel pela média de seu valor com os vizinhos
  - Torna cada pixel mais semelhante aos seus vizinhos

### Filtragem de média

- Exemplo unidimensional

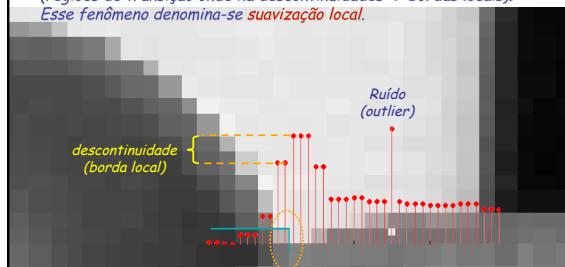


### Filtro de Média – exemplo unidimensional



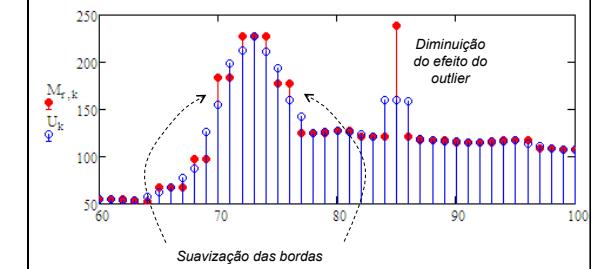
### Filtro de Média – exemplo unidimensional

A filtragem de média busca eliminar o ruído tornando o valor no pixel mais parecido com os vizinhos (suavização). Entretanto isso poderá ter efeitos indesejáveis sobre os contornos (regiões de transição onde há descontinuidades → bordas locais). Esse fenômeno denomina-se **suavização local**.



### Filtro de Média – exemplo unidimensional

Ocorre uma redução do contraste



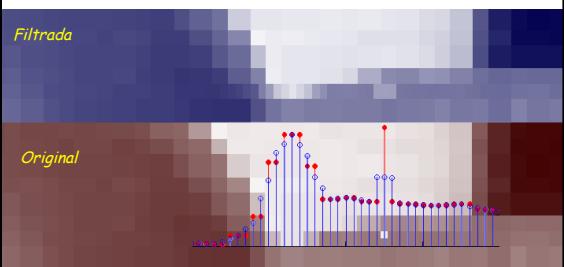
### Filtro de Média – exemplo unidimensional

Note a suavização das bordas locais.



### Filtro de Média – exemplo unidimensional

Lembrar que a filtragem foi realizada em apenas 1 linha da imagem

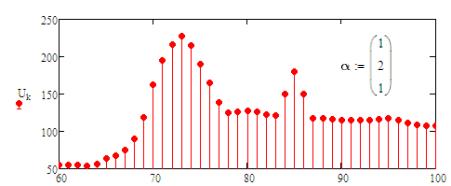


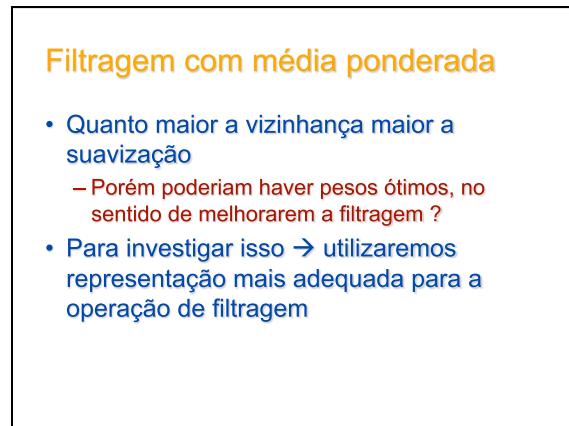
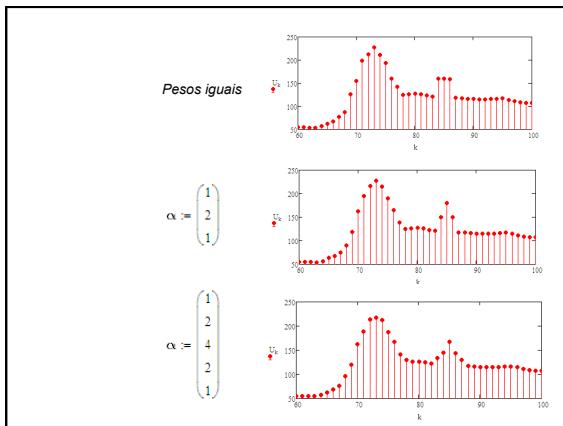
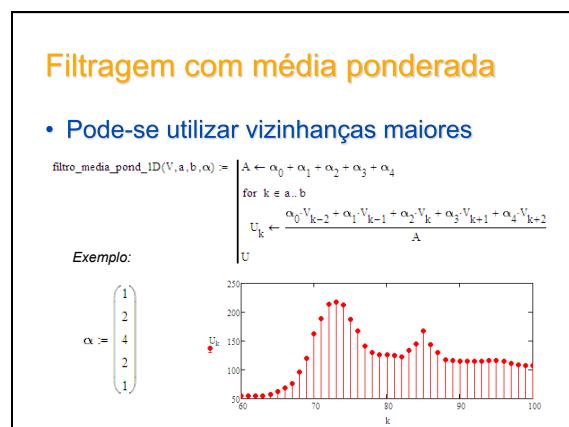
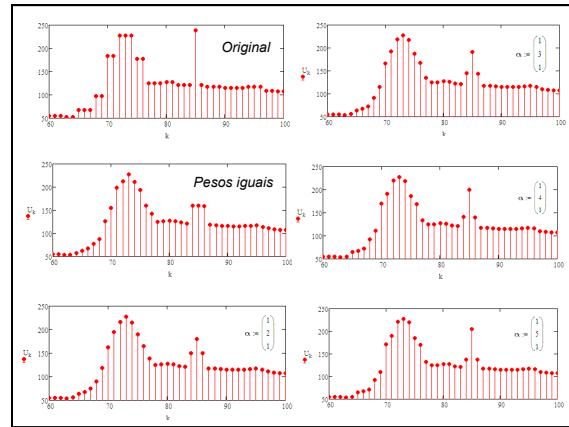
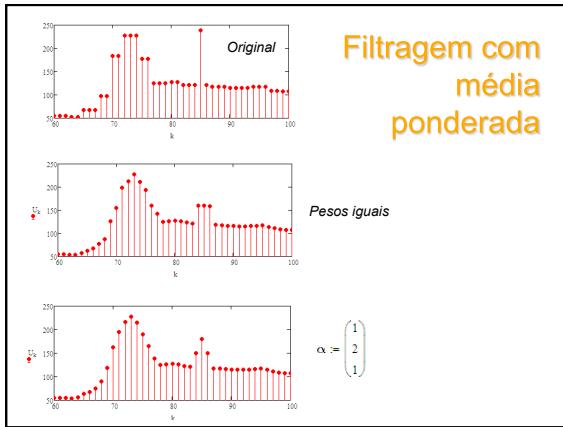
### Filtro de média

- O filtro de média pode envolver vizinhanças maiores
  - Quanto maior a vizinhança, maior a suavização (smoothing)
- A suavização observada é intensa porque o pixel tem o mesmo peso dos vizinhos no cálculo da média
- Pode-se usar média ponderada com peso maior no pixel em comparação com os pesos dos vizinhos

### Filtragem com média ponderada

```
filtro_media_pond_1D(V, a, b, α) := 
  A ← α0 + α1 + α2
  for k ∈ a..b
    Uk ←  $\frac{\alpha_0 \cdot V_{k-1} + \alpha_1 \cdot V_k + \alpha_2 \cdot V_{k+1}}{A}$ 
```





## Filtragem com média ponderada

- Tomemos um dos exemplos anteriores de cálculo da média ponderada:

$$U_k = \frac{\alpha_0 V_{k-2} + \alpha_1 V_{k-1} + \alpha_2 V_k + \alpha_3 V_{k+1} + \alpha_4 V_{k+2}}{A}$$

Onde:

$$A = \alpha_0 + \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4$$

Na expressão acima, é dada a sequência  $V_k$  que queremos filtrar, obtendo a sequência  $U_k$ . Os  $\alpha_k$  são os pesos. Essa expressão pode ser re-escrita na forma:

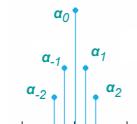
$$U_k = \frac{1}{A} \sum_{\xi=-\beta}^{\beta} (\alpha_\xi \cdot V_{k-\xi}) \quad \text{sendo } (2\beta+1) \text{ a largura do filtro e mudando-se os índices dos pesos.}$$

## Filtragem com média ponderada

$$U_k = \frac{1}{A} \sum_{\xi=-\beta}^{\beta} (\alpha_\xi \cdot V_{k-\xi}) \quad (2\beta+1) \rightarrow \text{largura do filtro.}$$

O filtro pode ser interpretado como uma outra sequência, formada pelos pesos que, combinada adequadamente com a sequência  $V_k$ , produz a sequência filtrada  $U_k$ :

$$\{a_{-2}, a_{-1}, a_0, a_1, a_2\}$$



Modificamos os índices dos filtros de modo a fazer com que o peso central tenha o índice zero.

## Filtragem com média ponderada

Se tomarmos a sequência  $\{a_{-2}, a_{-1}, a_0, a_1, a_2\}$

e a completarmos com valores zero de modo que ela fique do mesmo tamanho que a sequência  $\{V_k\}$  (essa operação chama-se "zero-padding"), então pode-se escrever:

$$U_k = \sum_{\xi} (\alpha_\xi \cdot V_{k-\xi}) = \sum_{\xi} (V_\xi \cdot \alpha_{k-\xi})$$

As duas expressões acima são simétricas. O índice  $\xi$  age como uma translação do filtro sobre os pontos da sequência  $V_k$ .

Diz-se que a média "move-se" sobre a sequência, suavizando os valores (smoothing). O filtro é dito também filtro de média móvel.

## Filtragem com média ponderada

$$U_k = \sum_{\xi} (\alpha_\xi \cdot V_{k-\xi}) = \sum_{\xi} (V_\xi \cdot \alpha_{k-\xi})$$

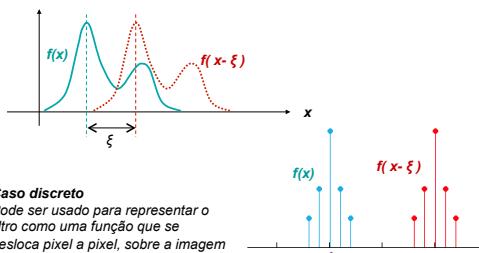
As expressões acima traduzem uma operação entre as sequências  $\{a_{-2}, a_{-1}, a_0, a_1, a_2\}$  e  $\{V_k\}$

Essa operação é linear e denomina-se **convolução**

$$U = V \otimes \alpha$$

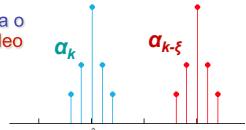
## Filtragem de média móvel

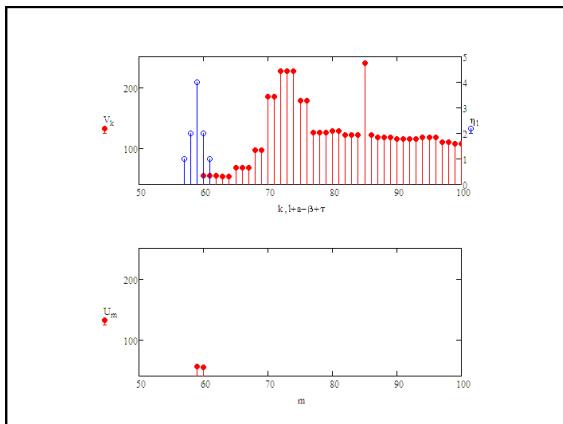
- Translação de uma função no domínio



## Filtragem de média móvel

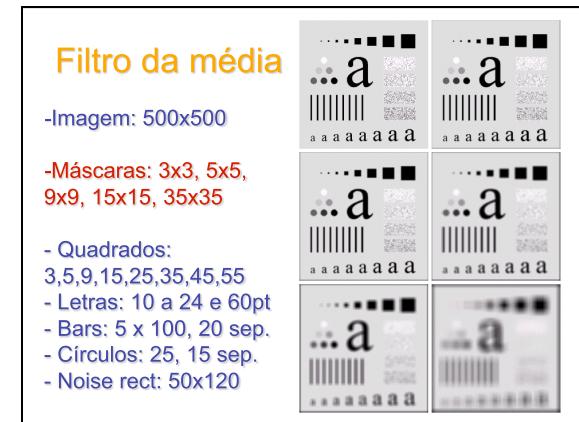
- O filtro de média ponderada com pesos  $a_k$ , com  $k = \{-\beta, \dots, 0, \dots, \beta\}$ , pode ser representado por uma função discreta  $a_{k-\xi}$ , com  $\xi = \{0, \dots, \text{tamanho da seq. } V\}$ , de modo que o filtro se desloca sobre toda a sequência  $V$ .
- A sequência  $a_k$ , que representa o filtro também se denomina **núcleo** ou **máscara** da operação de convolução.



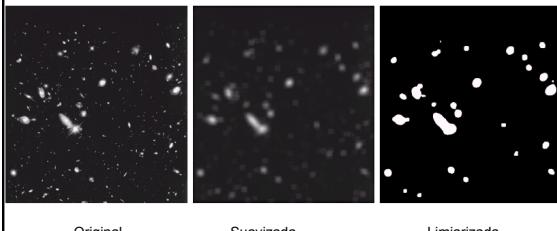


## Filtragem em imagens

- Podemos generalizar facilmente os métodos desenvolvidos para o caso unidimensional
- Os núcleos ou máscaras que compõem os filtros apresentam-se na forma de matrizes quadradas de ordem ímpar
  - Em geral apresentam alguma simetria em relação ao elemento central



### Suavização com filtro da média



### Filtragem convolucional

### Filtragem convolucional

- É o nome generalizado da filtragem de média móvel (ponderada)
- Os pesos são os elementos que constituem o núcleo ou máscara de convolução
- A filtragem convolucional é uma operação linear

### Filtros convolucionais

- De acordo com os valores dos pesos os filtros realizarão diferentes tipos de ações sobre a imagem
  - Suavização
  - Filtragem de ruído
  - Realce de contraste
  - Detecção de bordas
  - Detecção de atributos locais

### Suavização e filtragem de ruído

- Emprega métodos de média ponderada
- Há diversos tipos de filtros possíveis
  - As diferenças estão nos valores dos elementos do núcleo, ou máscara
  - Demonstra-se que os filtros ótimos constituem uma categoria denominada de *funções prolatas* (Torre e Poggio - 1984)
  - Dentre elas a mais utilizada é a máscara gaussiana (Marr e Hildreth – 1979)

### Filtragem com máscara gaussiana

- O núcleo ou máscara correspondem a valores tomados sobre um perfil gaussiano
 
$$G(\mu, \sigma)_{i,j} = \frac{1}{2\pi\sigma} \cdot e^{-\frac{(i-j)^2}{2\sigma^2}}$$
- O núcleo gaussiano é separável, isto é:
  - O núcleo  $\mathcal{K}$  pode ser decomposto no produto de dois vetores:
 
$$[\mathcal{K}] = \mathcal{K}_x \cdot \mathcal{K}_y^T$$
- Como a máscara é separável, o filtro gaussiano  $G(0,1)$  pode ser obtido facilmente através do cálculo de coeficientes binomiais ou do triângulo de Pascal

## Triângulo de Pascal

0:	1
1:	1 1
2:	1 2 1
3:	1 3 3 1
4:	1 4 6 4 1
5:	1 5 10 10 5 1
6:	1 6 15 20 15 6 1
7:	1 7 21 35 35 21 7 1
8:	1 8 28 56 70 56 28 8 1

## Coeficientes binomiais

$$C(m, n) := \text{if } [m \geq n, \frac{m!}{n!(m-n)!}, 0] \quad m \geq n \geq 0$$

$$A_{i,j} := C(i,j)$$

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 3 & 3 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 \\ 1 & 5 & 10 & 10 & 5 & 1 & 0 & 0 & 0 \\ 1 & 6 & 15 & 20 & 15 & 6 & 1 & 0 & 0 \\ 1 & 7 & 21 & 35 & 35 & 21 & 7 & 1 & 0 \\ 1 & 8 & 28 & 56 & 70 & 56 & 28 & 8 & 1 \end{pmatrix}$$

Exemplo:

$$K = \begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix}$$

## Exemplo

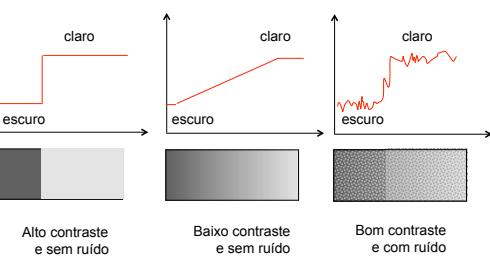
Filtragem da imagem corrompida com 5% de ruído gaussiano monocromático usando o filtro gaussiano de ordem 5

$$K = \begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix}$$

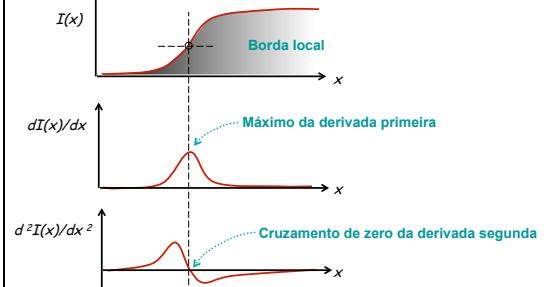


## Realce de contraste

## Princípio da detecção de contorno pelo gradiente local



## Princípio da detecção de contorno pelo gradiente local



## Derivadas parciais discretas

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

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

## Derivadas parciais discretas

- Deve ser zero em segmentos de nível de cinza constante
- Deve ser diferente de zero nas bordas das regiões cujo nível de cinza é crescente, ou decrescente
- Deve ser diferente de zero em regiões cujo nível de cinza é crescente, ou decrescente

## Derivada de primeira versus segunda ordem

- Derivadas de primeira ordem produzem bordas mais grossas
- Derivadas de primeira ordem respondem fortemente a inclinações
- Derivadas de segunda ordem respondem mais fortemente a impulsos pequenos
- Derivadas de segunda ordem produzem uma resposta dobrada a mudanças de níveis de cinza

## Laplaciano

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

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

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

## Laplaciano

$$\begin{aligned} \nabla^2 f &= f(x+1, y) + f(x-1, y) - 2f(x, y) \\ &\quad + f(x, y+1) + f(x, y-1) - 2f(x, y) \end{aligned}$$

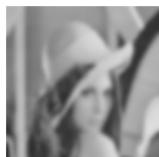
$$\begin{aligned} \nabla^2 f &= f(x+1, y) + f(x-1, y) + f(x, y+1) \\ &\quad + f(x, y-1) - 4f(x, y) \end{aligned}$$

## Realce de contraste

- Utilizando-se pesos negativos no núcleo em torno do valor central positivo, ou vice-versa, resulta na ênfase das diferenças entre o pixel central e seus vizinhos
  - Se a ênfase for suficientemente pronunciada, de modo que, para uma região uniforme da imagem, o resultado do realce seja nulo, ocorrerá a detecção dos contornos
  - Caso contrário, ocorrerá a ênfase do contraste ou dos contornos, de acordo com o sinal do elemento central
- Não se faz a normalização dos coeficientes neste caso

## Exemplo – realce de contornos

$$K := \begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$



## Simplificação

- Se subtrairmos de  $f$  o seu laplaciano (realce de contraste), teremos uma transformação que realça a imagem.

$$g = f - \nabla^2 f$$

## Simplificação

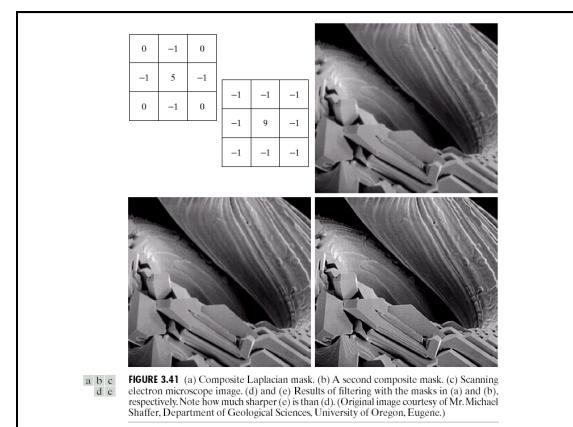
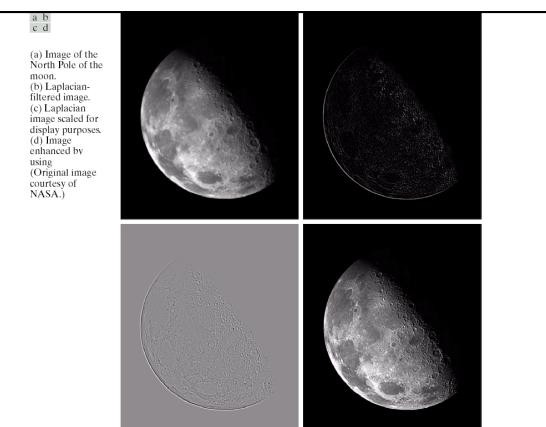
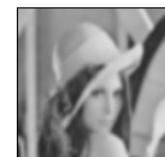
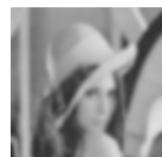
- Se subtrairmos de  $f$  o seu laplaciano (realce de contraste), teremos uma transformação que realça a imagem.

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

$$g = 5f(x, y) - f(x+1, y) - f(x-1, y) - f(x, y+1) - f(x, y-1)$$

## Exemplo – realce de contraste

$$K := \begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$



## Filtro high-boost

- Parte-se de uma operação de mascaramento não-afiado. Denotemos por  $f$  uma imagem em níveis de cinza, como usualmente e  $\bar{f}$  a imagem  $f$  transformada por algum operador de borramento

$$f_s = f - \bar{f}$$

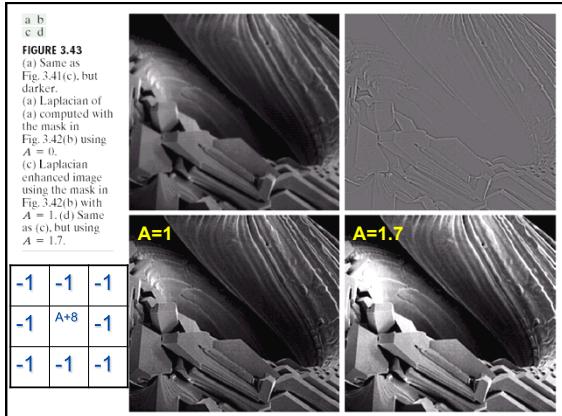
## Filtro high-boost

- Seja  $A$  um real positivo, maior ou igual a 1

$$f_{hb} = Af - \bar{f}$$

- Quando usamos um filtro laplaciano

$$f_{hb} = Af - \nabla^2 f$$



## Filtros “Gradiente”

- O gradiente de  $f$  é definido por:

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- O módulo do gradiente é definido por:

$$\|\nabla f\| = \left[ \left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2 \right]^{\frac{1}{2}}$$

## Filtros “Gradiente”

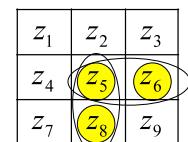
- O módulo do gradiente é aproximado por:

$$\|\nabla f\| \approx \sqrt{\left| \frac{\partial f}{\partial x} \right|^2 + \left| \frac{\partial f}{\partial y} \right|^2}$$

## Filtros “Gradiente”

$$\frac{\partial f}{\partial x} = (z_8 - z_5)$$

$$\frac{\partial f}{\partial y} = (z_6 - z_5)$$



### Gradiente cruzado de Roberts

$$\frac{\partial f}{\partial x} = (z_9 - z_5)$$

$$\frac{\partial f}{\partial y} = (z_8 - z_6)$$

$z_1$	$z_2$	$z_3$
$z_4$	$\text{z}_5$	$z_6$
$z_7$	$z_8$	$\text{z}_9$

### Gradiente cruzado de Roberts

$$\frac{\partial f}{\partial x} = (z_9 - z_5)$$

$$\frac{\partial f}{\partial y} = (z_8 - z_6)$$

$z_1$	$z_2$	$z_3$
$z_4$	$z_5$	$\text{z}_6$
$z_7$	$\text{z}_8$	$z_9$

$$\nabla f \approx |z_9 - z_5| + |z_8 - z_6|$$

### Gradiente cruzado de Roberts

$$\nabla f \approx |z_9 - z_5| + |z_8 - z_6|$$

-1	0
0	1

0	-1
1	0

### Operadores de Prewitt, 3x3

$$\nabla f \approx |(z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)| + |(z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)|$$

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

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

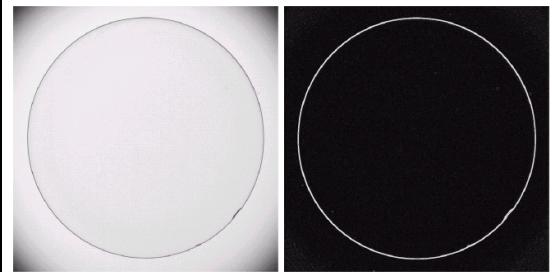
### Operadores de Sobel, 3x3

$$\nabla f \approx |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)|$$

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

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

### Operadores de Sobel



## Operadores de Sobel

a b

**FIGURE 10.10**  
 (a) Original image.  
 (b)  $|G_x|$ , component of the gradient in the x-direction.  
 (c)  $|G_y|$ , component in the y-direction.  
 (d) Gradient image,  $|G_x| + |G_y|$ .



## Prewitt, Sobel – realce diagonais

Prewitt

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

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

Sobel

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

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

## Realce de diagonais - Sobel



## Aplicação de Sobel, suavização

- Em alguns casos, podemos não querer que todas as bordas sejam realçadas
- Principalmente em casos que a imagem tem boa resolução
- Neste caso, aplica-se alguma suavização na imagem antes do operador gradiente

## Aplicação de Sobel, suavização

a

b

c

d

**FIGURE 10.11**  
 Same sequence as in Fig. 10.10, but with the original image smoothed with a  $5 \times 5$  averaging filter.



## Aplicação mais elaborada

- A aplicação a seguir é um exemplo de abordagem heurística para realce de imagem usando:
  - Operador laplaciano
  - Operador gradiente (Sobel)
  - Transformação de LUT

