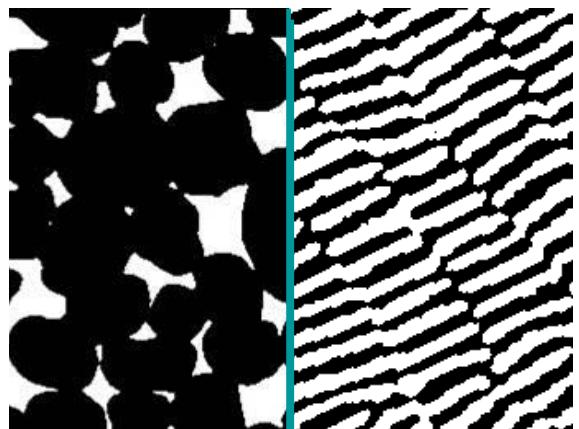
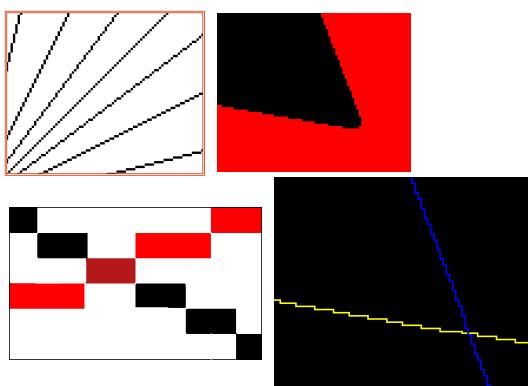
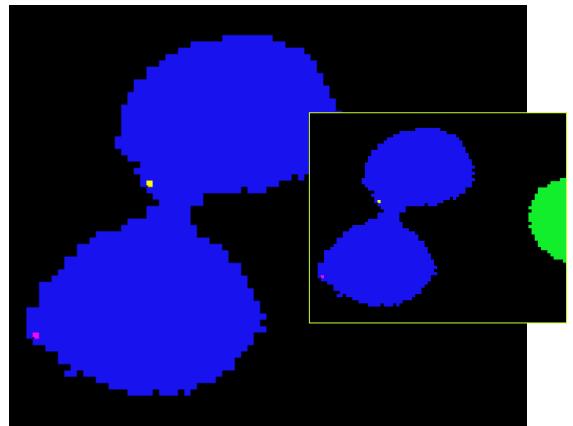
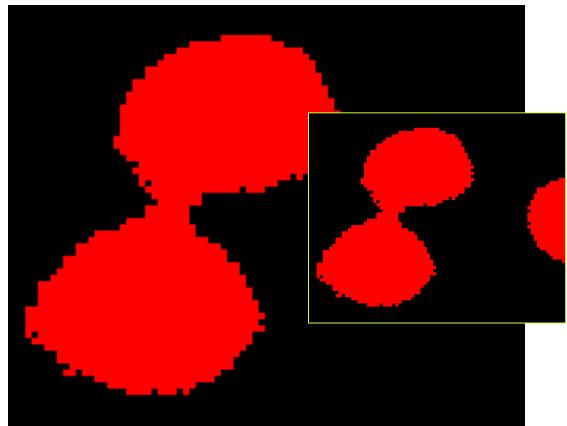




## Topologia digital Imagens Binárias



# Topologia Digital

## Topologia digital. É importante?

- Definição de **conexidade e componentes conexas** de um conjunto
- Classificação de **pontos interiores, exteriores e de borda**
- Formulação do teorema da **curva de Jordan** (um conjunto pode ser representado por suas bordas)
- Caracterização de **invariante topológico** (ex. número de Euler)
- Conservação da **homotopia** (ex. thinning)

## Noções de Topologia digital

- Objetos discretos, quando representam objetos contínuos, devem herdar certas características geométricas (propriedades geométricas).
- “O fosso cerca o castelo”
- Implicação geométrica: não dá para sair do castelo sem atravessar o fosso.

## Noções de Topologia digital

- O **plano digital** pode ser modelado como o conjunto  $Z \times Z$ , i.e., todos os pontos do plano real  $R \times R$  com coordenadas inteiras.
  - Em processamento de imagens, o plano digital pode ser usado como um modelo para as imagens P&B (objetos são representados por subconjuntos do plano digital)

## Noções de Topologia digital

$Z \times Z$	$X \subset Z \times Z$
○ ○ ○ ○ ○ ○	○ ○ ○ ● ● ●
○ ○ ○ ○ ○ ○	○ ○ ● ● ○ ●
○ ○ ○ ○ ○ ○	○ ● ● ○ ● ●
○ ○ ○ ○ ○ ○	● ● ○ ● ● ○
○ ○ ○ ○ ○ ○	● ○ ● ● ○ ○
○ ○ ○ ○ ○ ○	● ● ● ○ ○ ○

## Representação formal de imagem

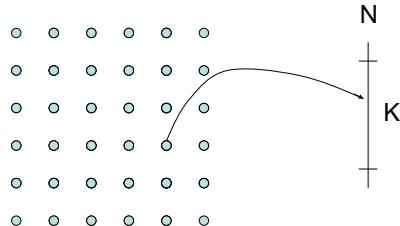
- Uma **imagem digital** pode ser modelada como uma aplicação de um subconjunto finito  $E$  de  $Z \times Z$  num subconjunto limitado dos naturais, ou num subconjunto limitado dos racionais (com representação binária).

$$f : E \rightarrow K, K \subset N$$

- Denotamos a imagem usualmente por  $f$  e o conjunto de todas as imagens de  $E$  em  $K$  por  $K^E$

## Representação formal

$$f : E \rightarrow K, K \subset N$$



## Representação formal

- Uma **imagem digital binária** é definida por:

$$f : E \rightarrow K, K = \{0, k\}$$

usualmente,  $k = 1$ , ou 255.

- Uma **imagem digital em níveis de cinza** é definida por:

$$f : E \rightarrow K, K = [0, k], k \in N$$

usualmente,  $k = 255$ .

## Exemplo

Exemplo de uma imagem binária  $6 \times 6$

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

## Comparação conjunto X função

Comparação de representações

0	0	0	•	•	•	0	0	0	1	1	1
0	0	•	•	•	0	•	0	0	1	1	0
0	•	•	•	0	•	•	0	1	1	0	1
•	•	0	•	•	•	0	1	1	0	1	1
•	0	•	•	0	0	0	1	0	1	1	0
•	•	•	0	0	0	0	1	1	1	0	0

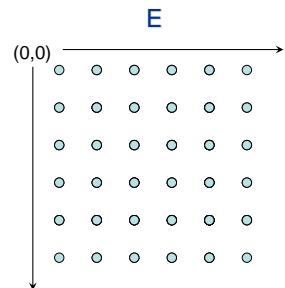
## Exemplo

Exemplo de uma imagem níveis de cinza  $5 \times 5$

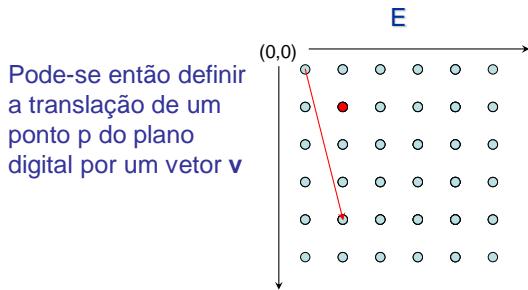
0	0	0	0	0
250	255	0	0	110
205	200	181	100	104
0	190	195	205	179
0	0	0	203	0

## Noções de Topologia digital

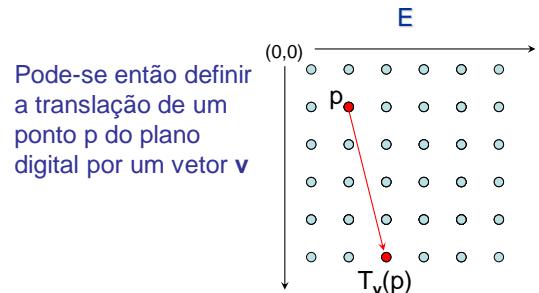
Coordenadas: em geral, coloca-se o centro no canto superior esquerdo de E e os eixos são orientados desta forma:



## Noções de Topologia digital



## Noções de Topologia digital



## Noções de Topologia digital

– **4-vizinhos:** dado um ponto  $p \in Z \times Z$ ,  $p = (x,y)$ , os 4-vizinhos de  $p$  são os **vizinhos diretos**, acima, abaixo, à direita e à esquerda.

	$y-1$	$y$	$y+1$
$x-1$		$N_2(p)$	
$x$	$N_4(p)$	$p$	$N_8(p)$
$x+1$		$N_6(p)$	

## Noções de Topologia digital

– **8-vizinhos:** os 8-vizinhos de  $p$  os vizinhos diretos e os indiretos, isto é, são todos os pontos com coordenadas inteiras  $(k,l)$  tais que:

	$y-1$	$y$	$y+1$
$x-1$	$N_3(p)$	$N_2(p)$	$N_1(p)$
$x$	$N_4(p)$	$p$	$N_8(p)$
$x+1$	$N_5(p)$	$N_6(p)$	$N_7(p)$

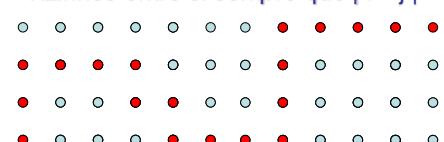
## Noções de Topologia digital

– O conjunto dos 8-vizinhos do ponto  $p$  união com ele mesmo é denotado usualmente por  $\mathcal{N}_8(p)$ .

– O conjunto dos 4-vizinhos do ponto  $p$  união com ele mesmo é denotado usualmente por  $\mathcal{N}_4(p)$ .

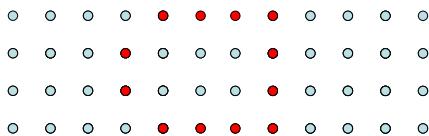
## Topologia digital

– Seja  $K \in \{4,8\}$  e  $I = \{0, 1, \dots n\}$ . Um  **$K$ -caminho digital**, ou simplesmente **caminho**  $\mathcal{P}$  é uma seqüência  $\{p_i\}_{i \in I}$  de pontos em  $Z \times Z$  tais que  $p_i$  e  $p_j$  são  $K$ -vizinhos entre si sempre que  $|i - j| = 1$ .



## Topologia digital

- Um ponto é dito **terminal** se seu índice é 0, ou  $n$ , e ele tem apenas um vizinho pertencente ao caminho.
- Um caminho tal que  $p_0 = p_n$  é dito **caminho fechado**.



## Teorema da curva de Jordan (versão digital)

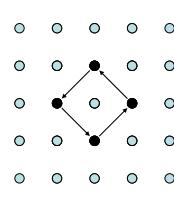
- Dada uma **curva fechada simples** (sem cruzamentos)  $\mathcal{P}$  em um plano digital  $Z \times Z$ , então,  $Z \times Z \setminus \mathcal{P}$  consiste em exatamente dois conjuntos conexos.
- Um dos conjuntos é limitado e é chamado de **interior** com respeito a  $\mathcal{P}$  e o outro é ilimitado e é chamado **exterior** com respeito a  $\mathcal{P}$ .

## Topologia digital

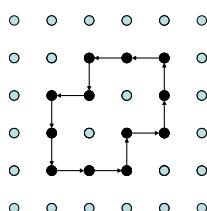
- Um subconjunto  $S$  de  $Z \times Z$  é dito um conjunto  **$K$ -conexo** se por quaisquer dois pontos  $p$  e  $q$  de  $S$  houver um caminho  $P$  que passa por  $p$  e  $q$  e totalmente contido em  $S$ .
- Uma **componente conexa** de um conjunto  $S$  de  $Z \times Z$  é um conjunto conexo maximal de  $S$ .

## Topologia digital

Segundo problema: **Paradoxo da conectividade**



1 componente 8-conexa



3 componentes 4-conexas

## Topologia digital

Primeiro problema:



8-conexo



4-conexo

- Uma 8-curva fechada simples deve ter no mínimo 4 pontos
- Uma 4-curva fechada simples deve ter no mínimo 8 pontos

## Topologia digital

- Em 1979, Rosenfeld provou que o teorema da curva de Jordan é verdadeiro se a curva e seu complemento são equipados com topologias diferentes.

– Denotemos por:

- $K' = 4$  se  $K = 8$
- $K' = 8$  se  $K = 4$

## Teorema da curva de Jordan (segunda versão digital)

- Dada uma  $\mathcal{K}$ -curva fechada simples  $\mathcal{P}$  em um plano digital  $Z \times Z$ , então,  $Z \times Z \setminus \mathcal{P}$  consiste em exatamente dois

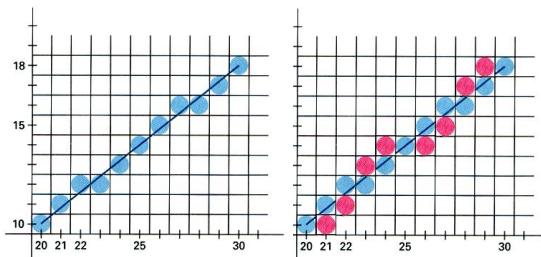
$\mathcal{K}$ -conjuntos conexos.

- Um dos conjuntos é limitado e é chamado de **interior** com respeito a  $\mathcal{P}$  e o outro é ilimitado e é chamado **exterior** com respeito a  $\mathcal{P}$ .

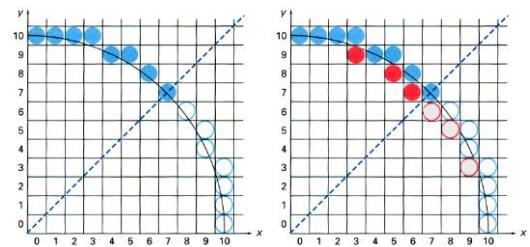
## Topologia Digital

- A topologia da imagem determina como conectar os pixels
- Ao separar regiões da imagem, devemos rotular seus pixels, dizendo a qual região pertencem
- A regra de conectividade derivada da topologia (4 ou 8) determinará se devemos incluir, ou não, um pixel na vizinhança de outros
- Serve também para definir as regras de construção de modelos a serem ajustados a contornos e regiões

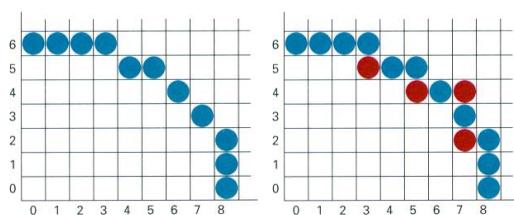
## Exemplo 1 - Reta



## Exemplo 2 - Circunferência



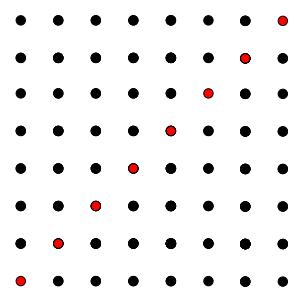
## Exemplo 3 - Elipse



Qual a conectividade a ser usada para as curvas (contornos) ?

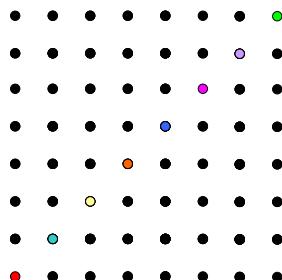
Na figura ao lado tem-se 1 contorno separando 2 regiões.

Como deve ser a conectividade ?



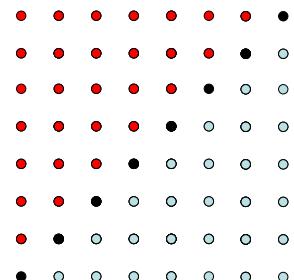
## Qual a conectividade?

Se escolhemos 4 para os contornos e 8 para as regiões, teremos 8 componentes conexas no contorno e 1 componente nas regiões da figura ao lado



## Qual a conectividade?

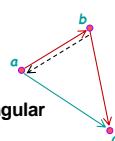
Se escolhemos 8 para os contornos e 4 para as regiões, teremos 2 componentes conexas para as regiões na figura ao lado e 1 para o contorno



## Atributos Locais

### Distância

- Distância → é um número que caracteriza a separação entre dois objetos
  - Seja  $S = \{\text{conjunto de objetos}\}$
  - Seja  $d: S \times S \rightarrow \mathcal{R}_+$
- 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  $d(a,b) + d(b,c) \geq d(a,c)$



## Conceito de Localidade

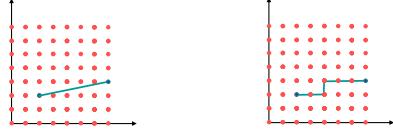
- Atributo local → do original *Local Feature*
  - *Feature* → detalhe peculiar, característica, propriedade ou atributo característico ou típico, detalhe importante
  - *Local feature* → enfatiza o aspecto de localidade
- Localidade → associado a uma **vizinhança** do pixel
  - É o conjunto dos pixels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

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

- A distância foi definida em  $\mathcal{R}_+$ 
  - 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 **dos objetos**
  - As extrínsecas
    - Obedecem à topologia e curvatura do **espaço das representações**

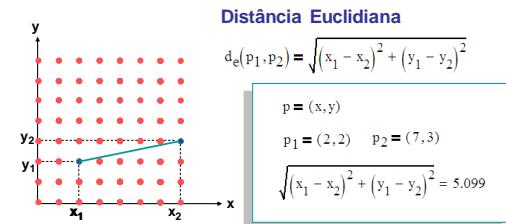
## Distância entre pixels

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



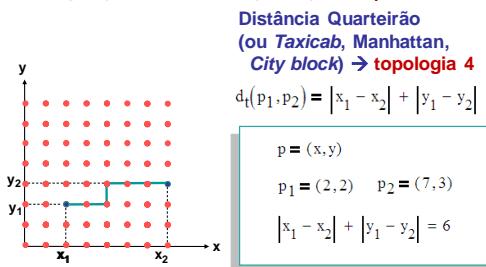
## Distância entre pixels

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



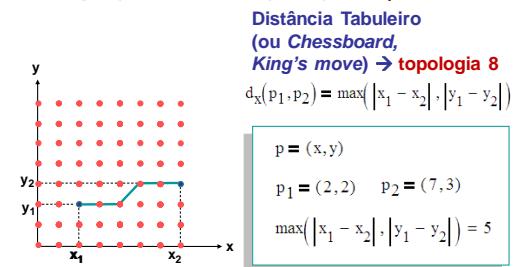
## Distância entre pixels

- 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 obedeçam aos requisitos: positividade, comutatividade, desigualdade triangular



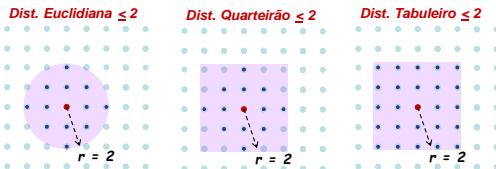
## Vizinhança de um pixel

- 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 – $K = 4$

Input	Output
0 0 0 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

### Algoritmo de rotulação – K = 4

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

Input

Output

### Algoritmo de rotulação – K = 4

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

Input

Output

### Algoritmo de rotulação – K = 4

<b>0 0 0 1</b>	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,  $I$ , 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 0 0 1</b>	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,  $I$ , 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 0 0 1 1 1</b>	<b>0 0 0 1 0 0</b>
0 0 1 <b>1 0 1</b>	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 <b>1 0 1</b>	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 <b>1 0 1</b>	0 0 0 <b>0 0 0</b>
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 <b>1 0 1</b>	0 0 0 <b>1 0 0</b>
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 <b>1 0 1</b>	0 0 0 <b>1 0 0</b>
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 <b>1 0 1</b>	0 0 0 <b>1 0 0</b>
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 <b>1</b> 0 1	0 0 0 <b>1</b> 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 <b>1</b> 0 1	0 0 0 <b>1</b> 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 <b>1</b> 1 0 1	0 0 0 <b>1</b> 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 <b>1</b> 1 0 1	0 0 <b>0</b> <b>1</b> 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 <b>1</b> 1 0 1	0 0 <b>0</b> <b>1</b> 1 0 0
0 1 1 0 1 1	0 0 0 0 0 0 0
1 1 0 1 1 0	0 0 0 0 0 0 0
1 0 1 1 0 0	0 0 0 0 0 0 0
1 1 1 0 0 0	0 0 0 0 0 0 0

Input                      Output

### 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  $s \in N_K(r)$ , se  $f(s) = 1$  e  $g(s) = 0$ ,
- faça  $g(s) = l$  e ponha  $s$  na fila

### Algoritmo de rotulação – K = 4

<b>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

### 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

### 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

### Algoritmo de rotulação – K = 4

1	2	62	63	58
6	3	4	64	57
7	5	8	66	52
9	61	50	51	53
12	10	11	49	47
13	27	28	46	
15	14	29	45	44
18	16	32	30	31
19	17	26	33	41
20	24	25	34	35
21	22	23	37	36
				40

4  
↑  
 $p$   
↓  
3 ← → 1

## 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	1	1	1	1	0	0	1	1	1	1	0
0	0	1	1	0	0	0	0	1	1	1	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>6</sub> (p)
x+1		N <sub>8</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	0
0	0	1	1	1	1	0	0	0	1
0	1	1	1	1	1	0	0	1	1
0	1	1	1	1	1	0	0	1	1
0	1	1	1	1	0	0	0	1	1
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 0	0 0 1 0 0 0
0 1 1 1 1 0	0 0 0 0 0 0 0
0 1 1 1 1 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 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 0	0 0 1 1 0 0
0 1 1 1 1 0	0 0 0 0 0 0 0
0 1 1 1 1 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 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 0	0 0 1 1 1 0
0 1 1 1 1 0	0 0 0 0 0 0 0
0 1 1 1 1 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 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 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
0 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 0	0 0 1 1 1 0
0 1 -1 1 1 0	0 1 2 0 0 0
0 1 1 1 1 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 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 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
0 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 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 0

Simulação – G = 4

0 0 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 1 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 0

Simulação – G = 4

0 0 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 1 2 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 0

Simulação – G = 4

0 0 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 2 1 0	0 1 2 1 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 0

Simulação – G = 4

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

Simulação – G = 4

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

## Simulação – G = 4

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

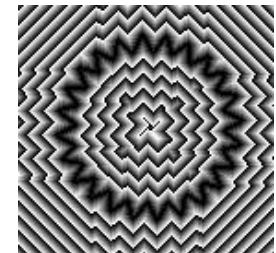
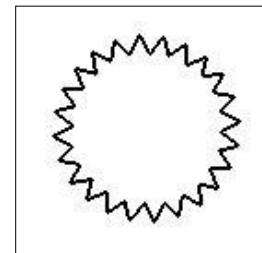
0	0	0	0	0	0	0
0	0	1	1	1	0	
0	1	2	2	1	0	
0	1	2	2	1	0	
0	1	1	1	0	0	
0	0	0	0	0	0	0

Alguns slides depois...

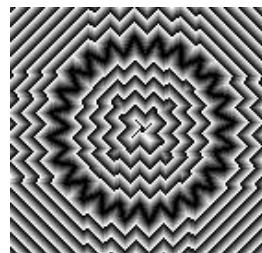
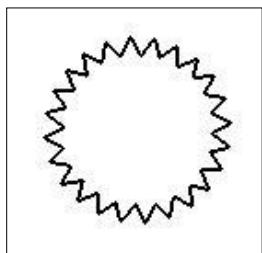
## Simulação – $G = 4$

0	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	0

## Transformada Distância

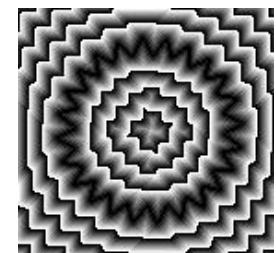
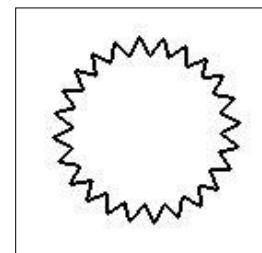


## Transformada Distância



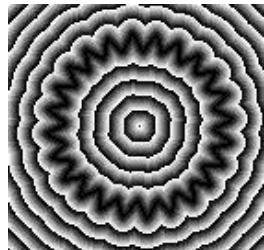
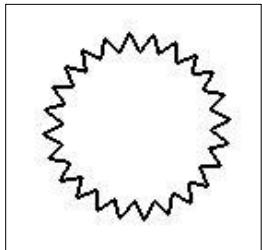
Transformação distância,  $d_t$

## Transformada Distância



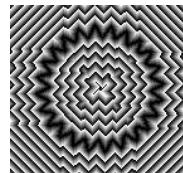
Transformação distância,  $d_x$

### Transformada Distância

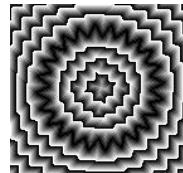


Transformação distância,  $d_e$

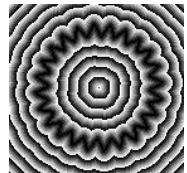
### Transformada Distância



City-block



Chess-board



Euclidean