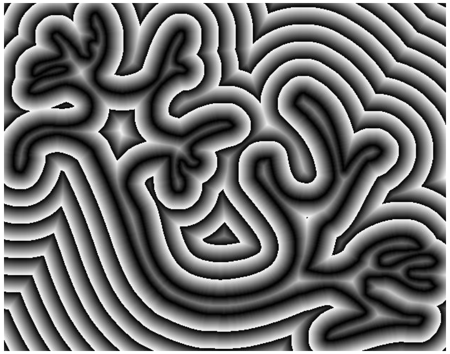
Comparação e Desenvolvimento de Algoritmos de Transformada de Distância Euclidiana e Aplicações



Aluno: Ricardo Fabbri
Orientador: Odemir Martinez Bruno
e-mail: rfabbri@if.sc.usp.br

Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo, Brasil.

FAPESP

08/2004

Plano da Apresentação

- Introdução
  - Definição de TD
  - Aplicações
- Algoritmos Euclidianos
  - Tipos de algoritmos
  - Principais algoritmos de cada tipo
- Metodologia
- Resultados
- Conclusões
- Referências

Introdução

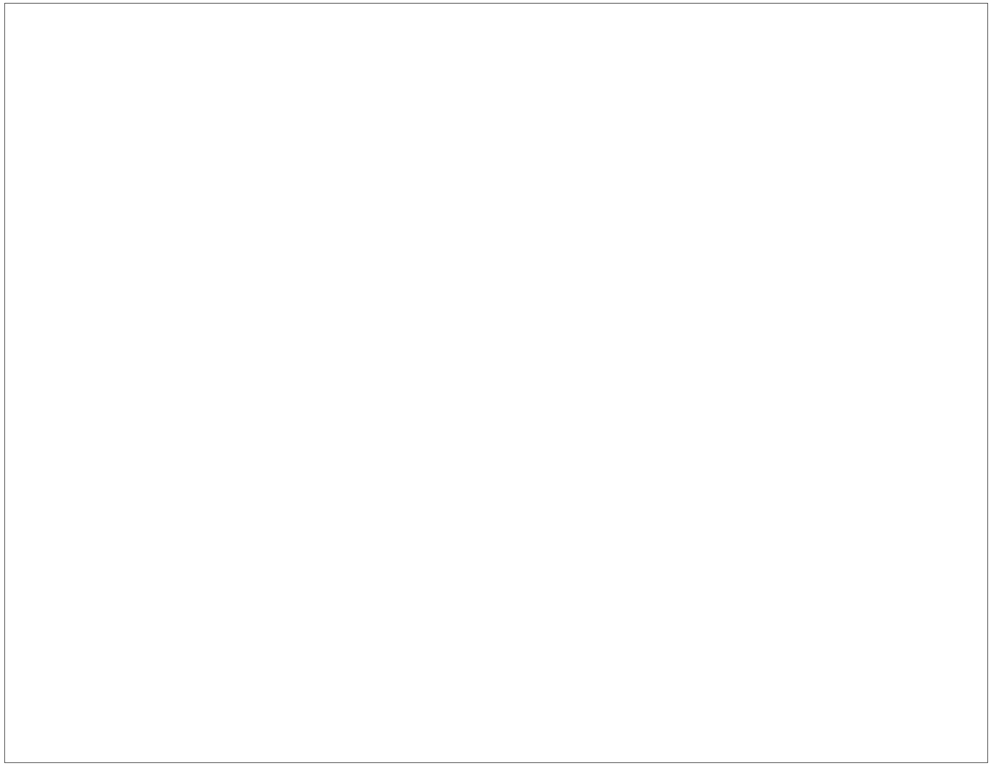
Transformada da Distância

- Para cada ponto de um domínio
  - Calcular a mínima distância ao conjunto de interesse

Transformada da Distância

- Para cada ponto de um domínio
  - Calcular a mínima distância ao conjunto de interesse

Domínio

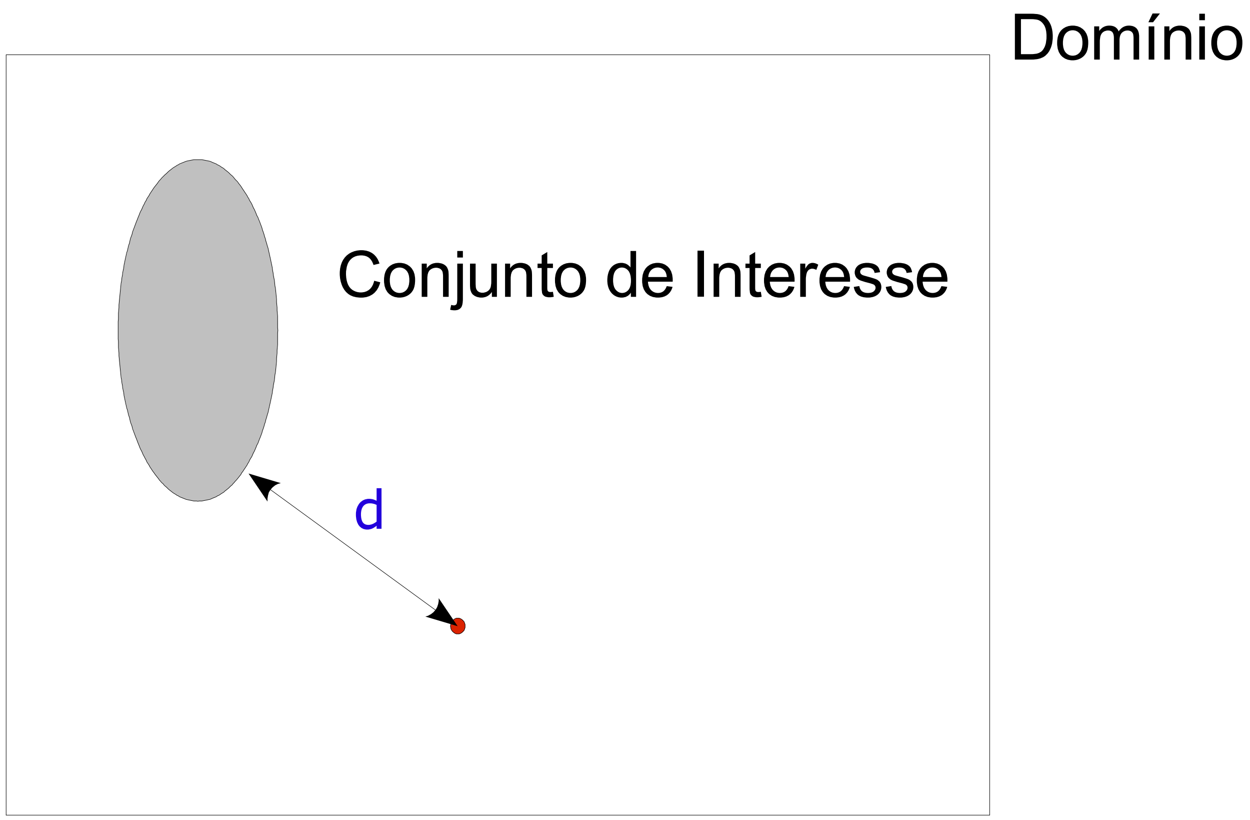


Transformada da Distância

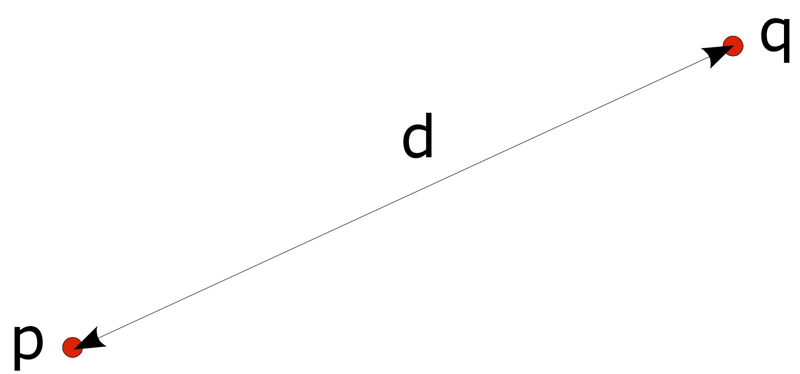
- Para cada ponto de um domínio
  - Calcular a mínima distância ao conjunto de interesse

Transformada da Distância

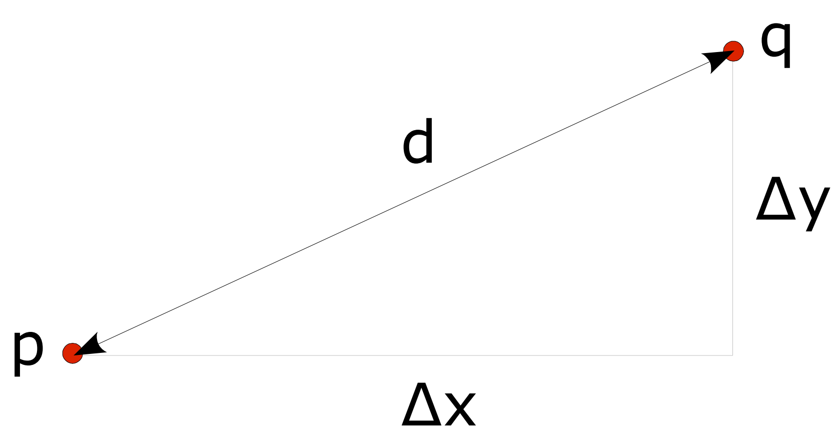
- Para cada ponto de um domínio
  - Calcular a mínima distância ao conjunto de interesse



Distância Pontual



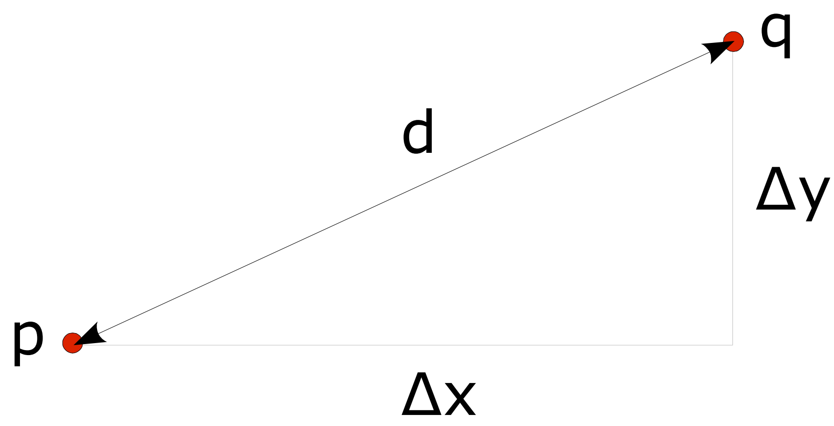
Distância Pontual



$$d = \sqrt{\Delta^2 x + \Delta^2 y}$$

(euclidiana)

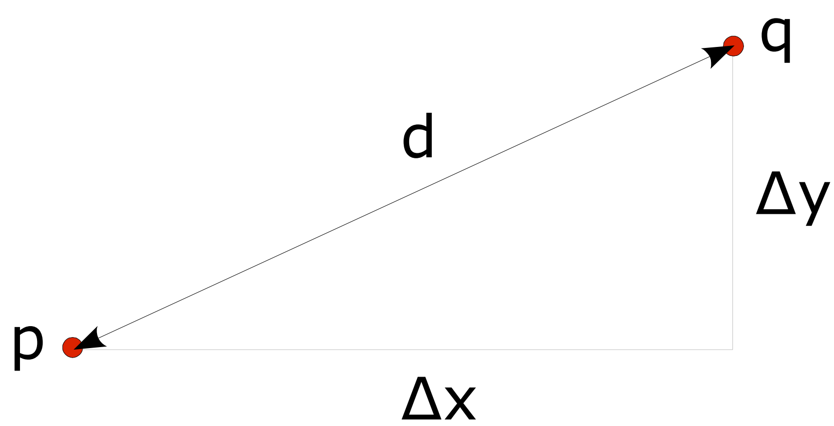
Distância Pontual



$$d = \sqrt{\Delta^2 x + \Delta^2 y} \qquad \text{(euclidiana)}$$

$$= |\Delta x| + |\Delta y| \qquad (\textit{cityblock})$$

Distância Pontual



$$d = \sqrt{\Delta^2 x + \Delta^2 y} \qquad \text{(euclidiana)}$$

$$= |\Delta x| + |\Delta y| \qquad (\textit{cityblock})$$

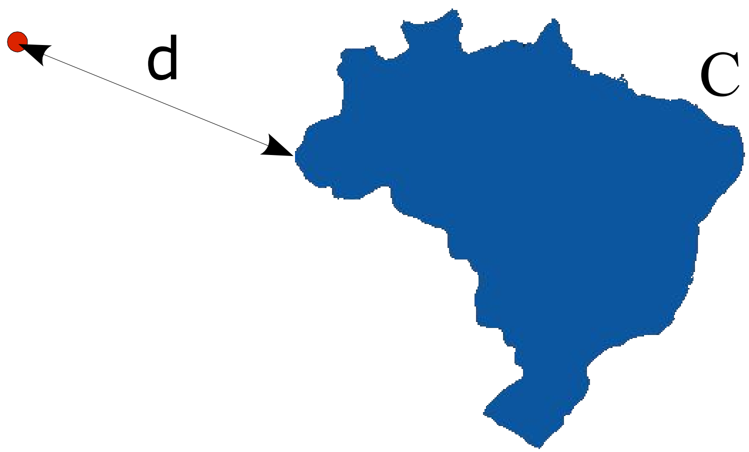
$$= \max\{|\Delta x|,\ |\Delta y|\} \qquad (\textit{chessboard})$$

Distância de Ponto a Conjunto

- Menor distância

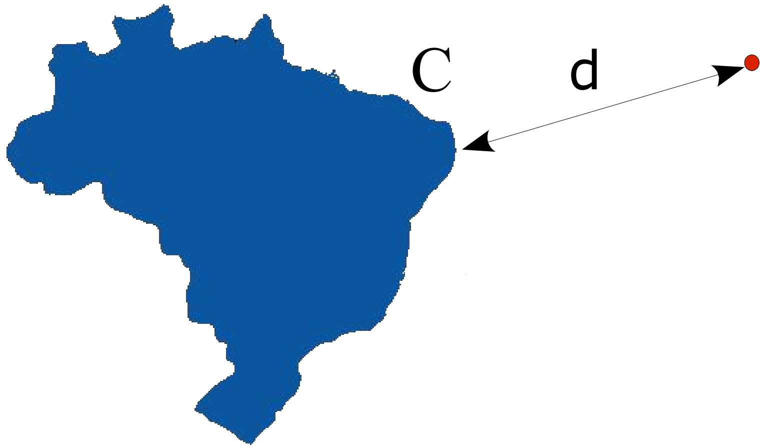
Distância de Ponto a Conjunto

- Menor distância

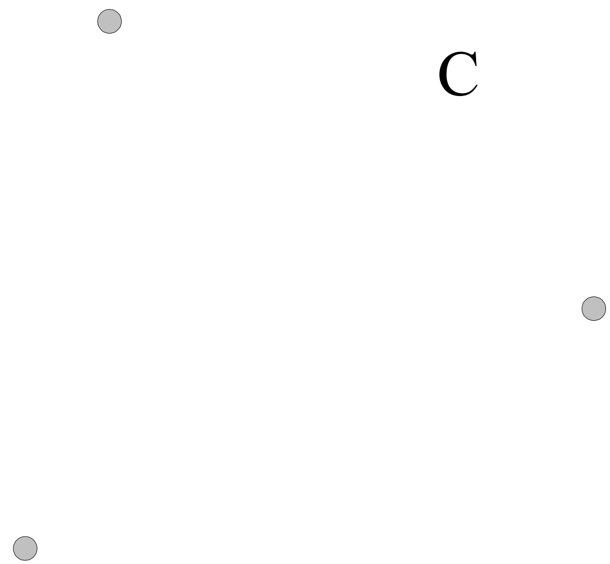


Distância de Ponto a Conjunto

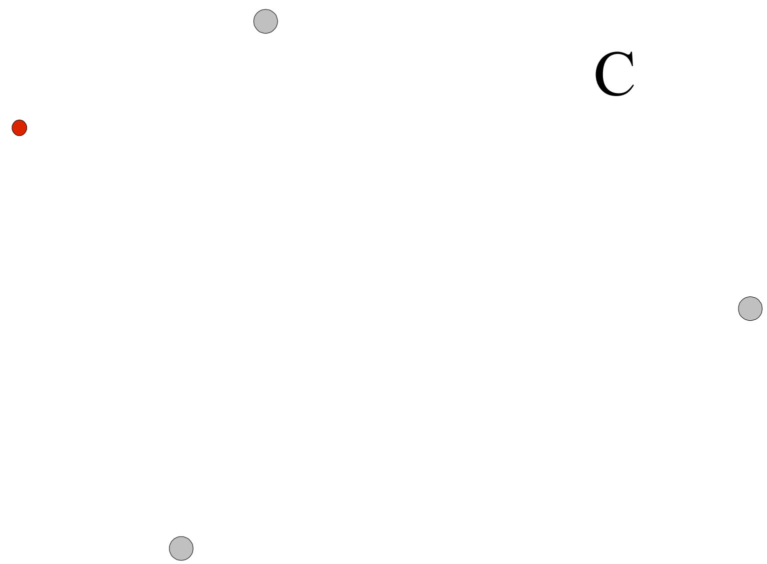
- Menor distância



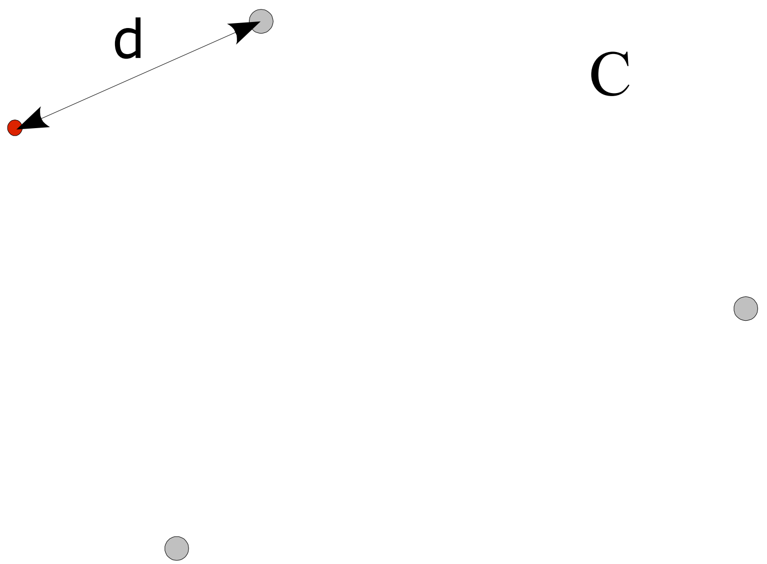
Distância de Ponto a Conjunto



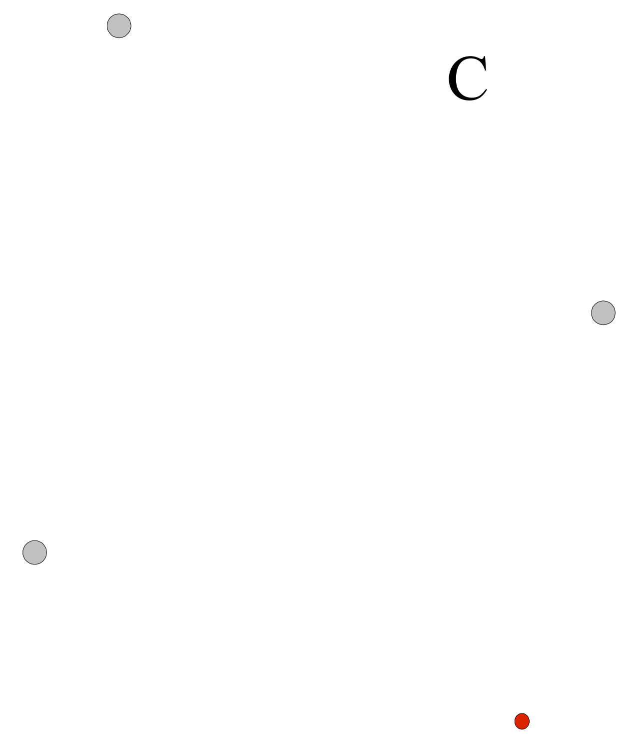
Distância de Ponto a Conjunto



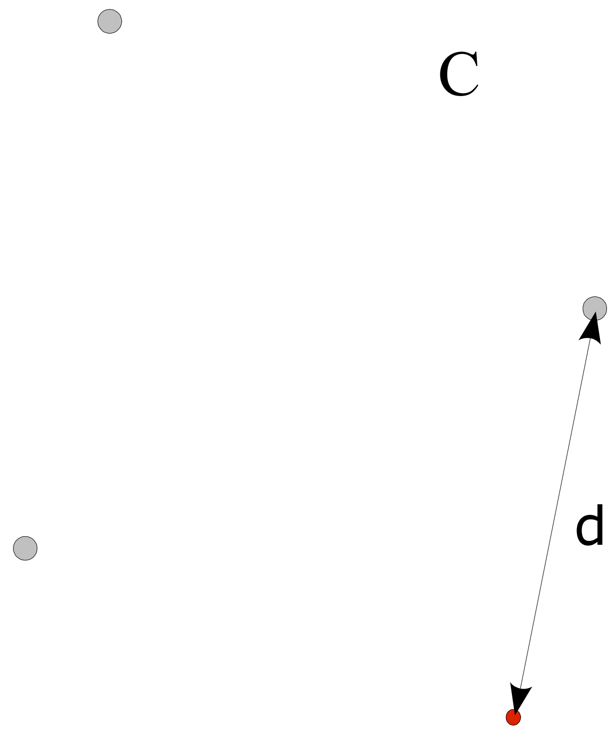
Distância de Ponto a Conjunto



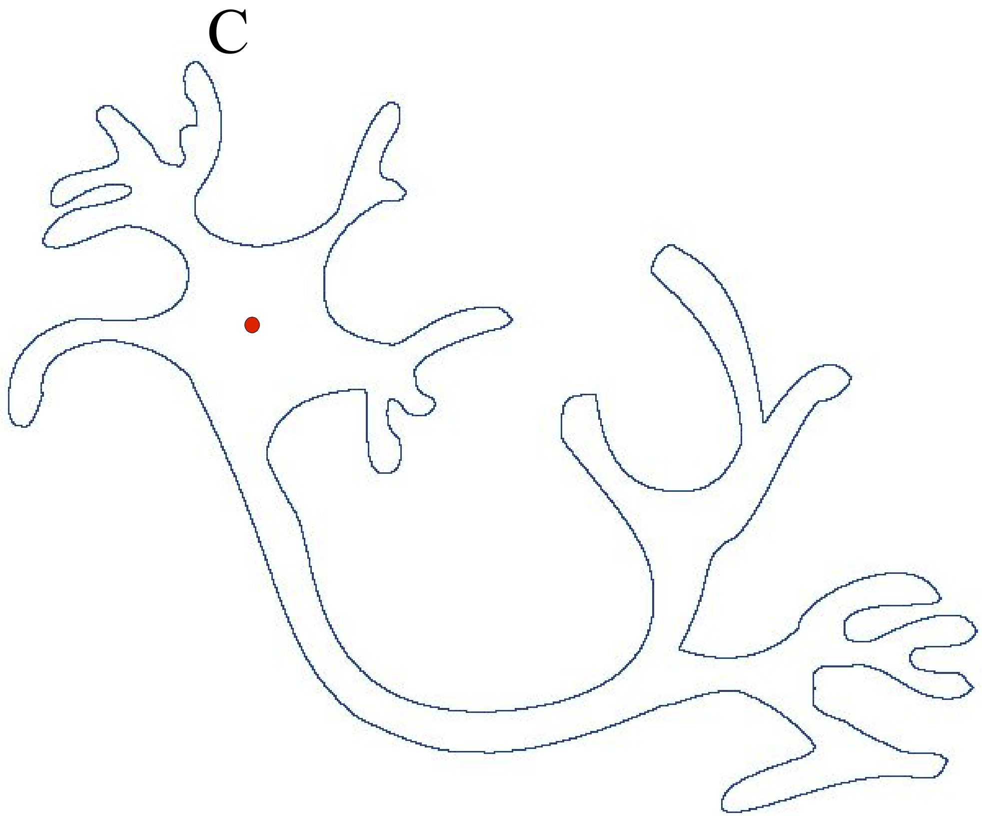
Distância de Ponto a Conjunto



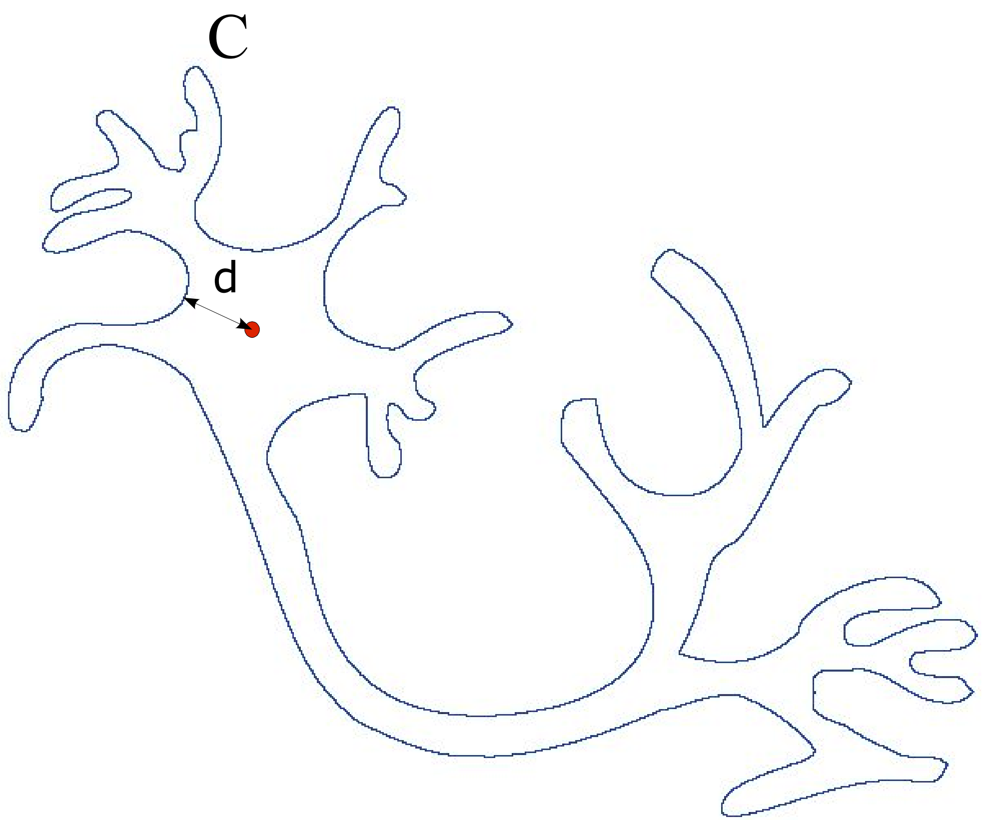
Distância de Ponto a Conjunto



Distância de Ponto a Conjunto



Distância de Ponto a Conjunto



TD para Imagens

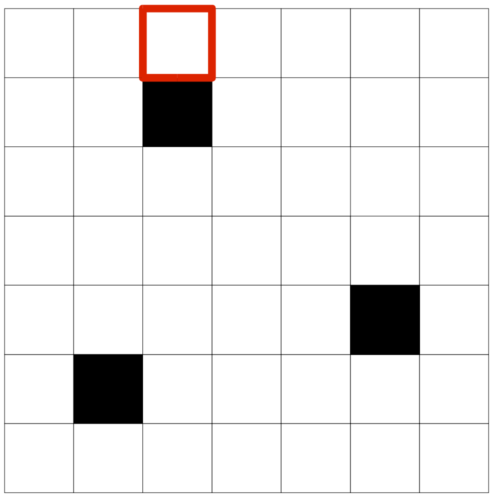
- Domínio
  - Imagem (grade discreta)
- Conjunto de interesse
  - Pixels com valor Zero (pixels pretos)
- Objetivo
  - Para cada pixel, calcular sua distância ao conjunto de interesse

Transformada da Distância

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

Transformada da Distância

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



Transformada da Distância

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

		$\boxed{1}$				

Transformada da Distância

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

		1				

Transformada da Distância

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

		1				
5						

Transformada da Distância

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

5	2	1	2	5	10	17
4	1	0	1	4	9	10
5	2	1	2	5	4	5
5	4	4	5	2	1	2
2	1	2	4	1	0	1
1	0	1	4	2	1	2
2	1	2	5	5	4	5

Mapa de distâncias

Transformada da Distância

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

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0
0	0	1	2	4	2	1	0	0
0	0	1	4	8	4	1	0	0
0	0	1	2	4	2	1	0	0
0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0

Mapa de distâncias

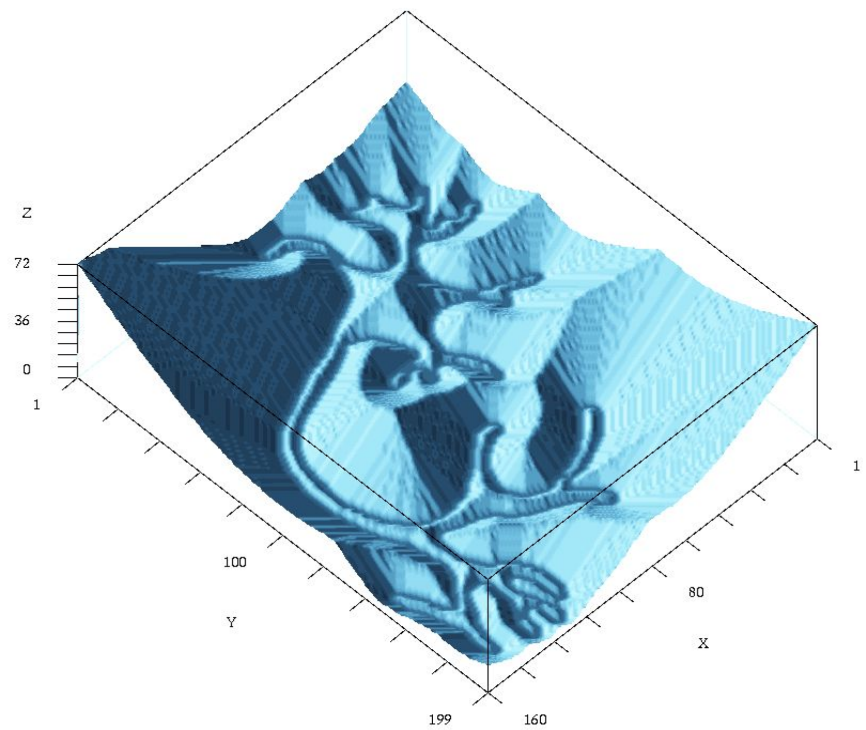
Transformada da Distância

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

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0
0	0	1	2	4	2	1	0	0
0	0	1	4	8	4	1	0	0
0	0	1	2	4	2	1	0	0
0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0

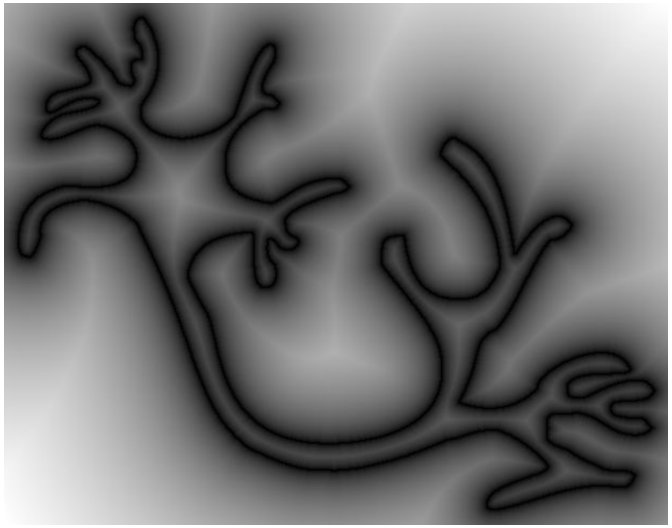
- Pontos de interesse = pixels pretos
- Em análise de formas:
  - Objeto = pixels brancos

Visualizando a TD



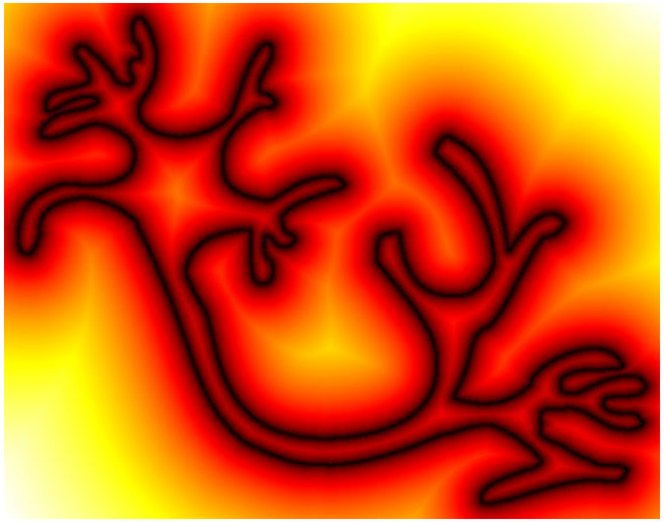
Altura ~ distância

Visualizando a TD



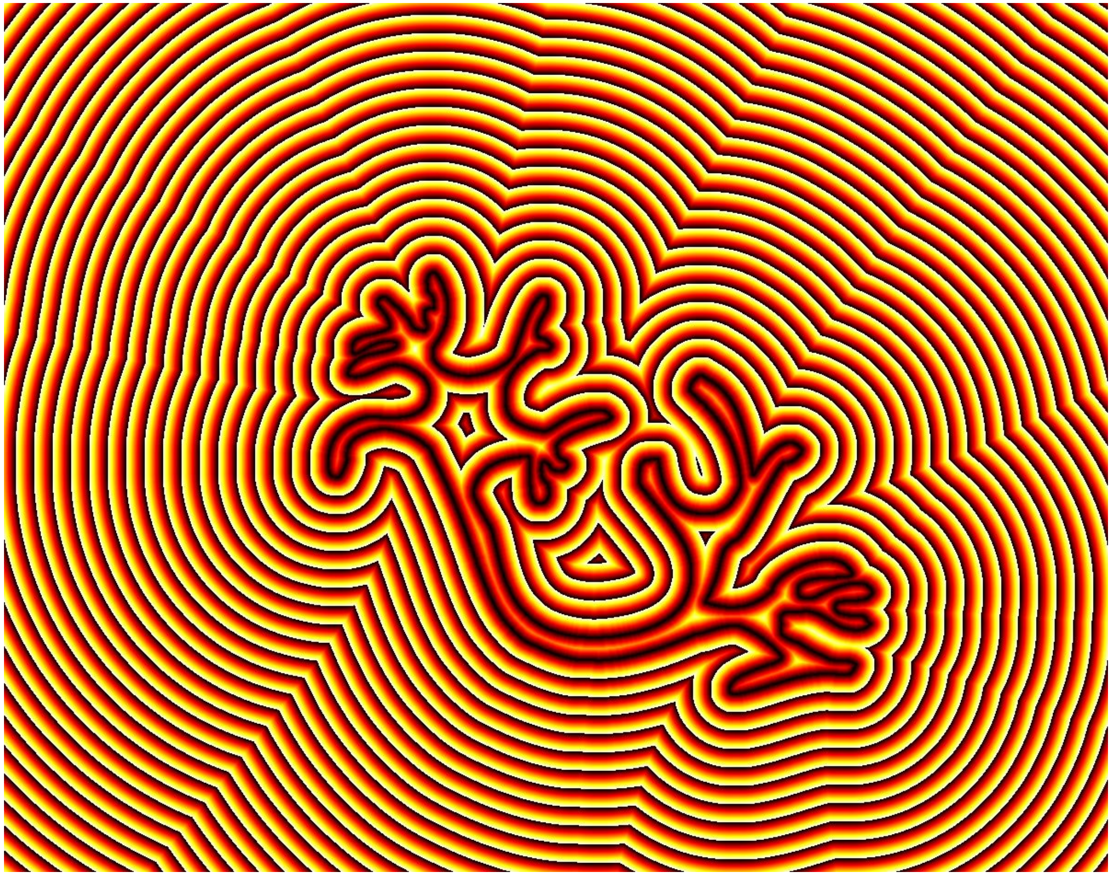
Brilho ~ distância

Visualizando a TD



Brilho ~ distância

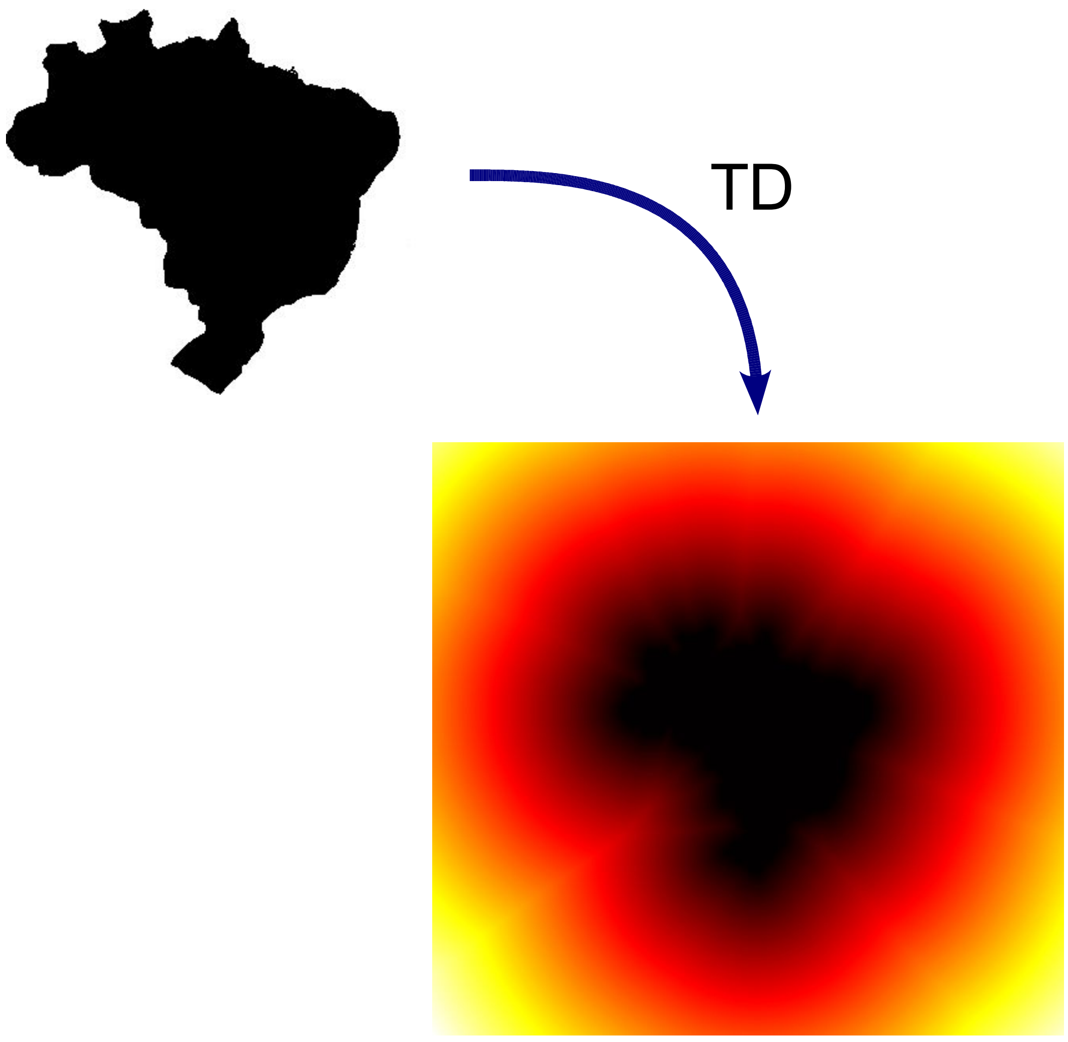
Visualizando a TD



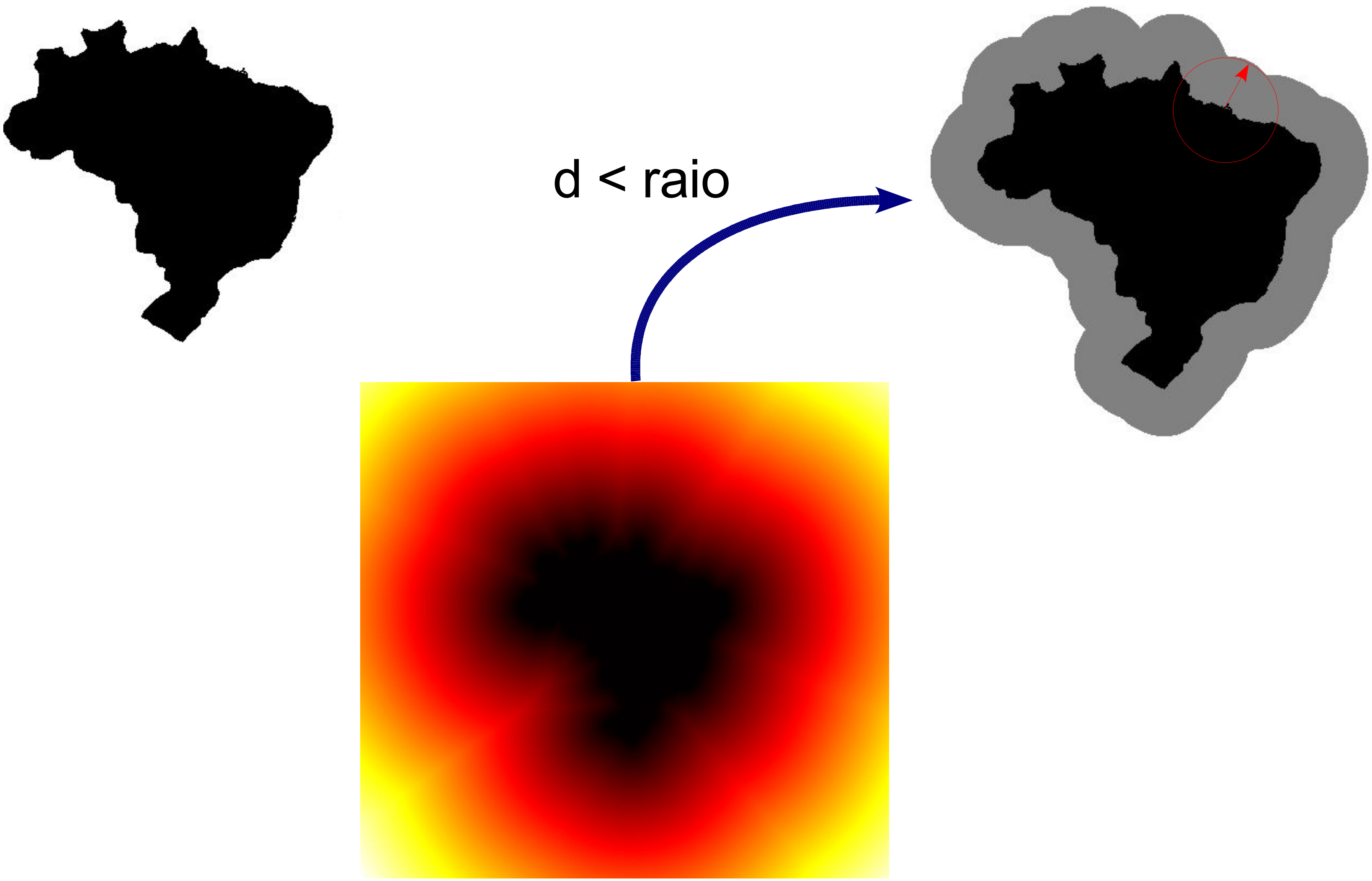
- Distâncias módulo n
- Curvas de nível ~ transições abruptas

Aplicações da TD

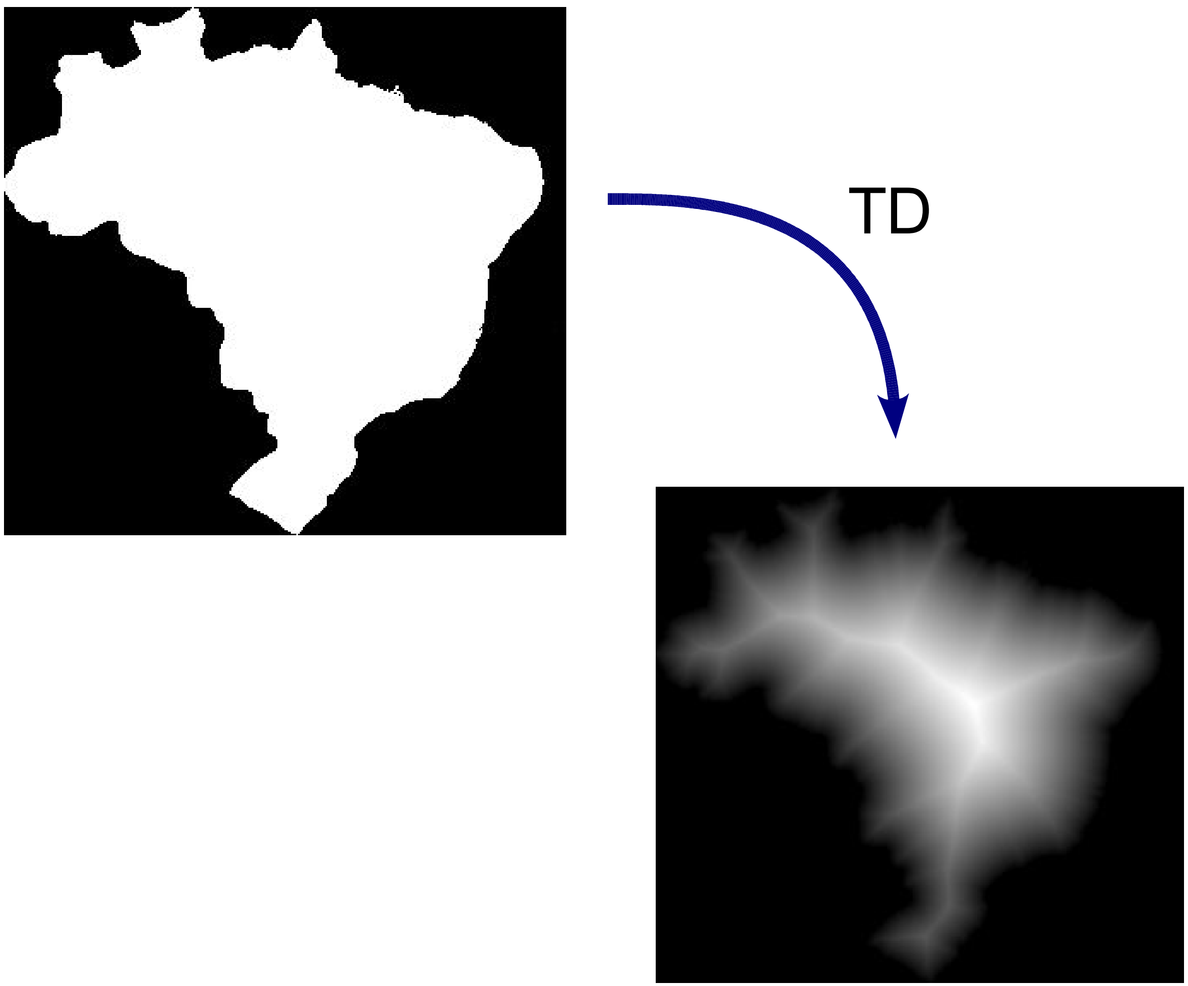
Dilatação



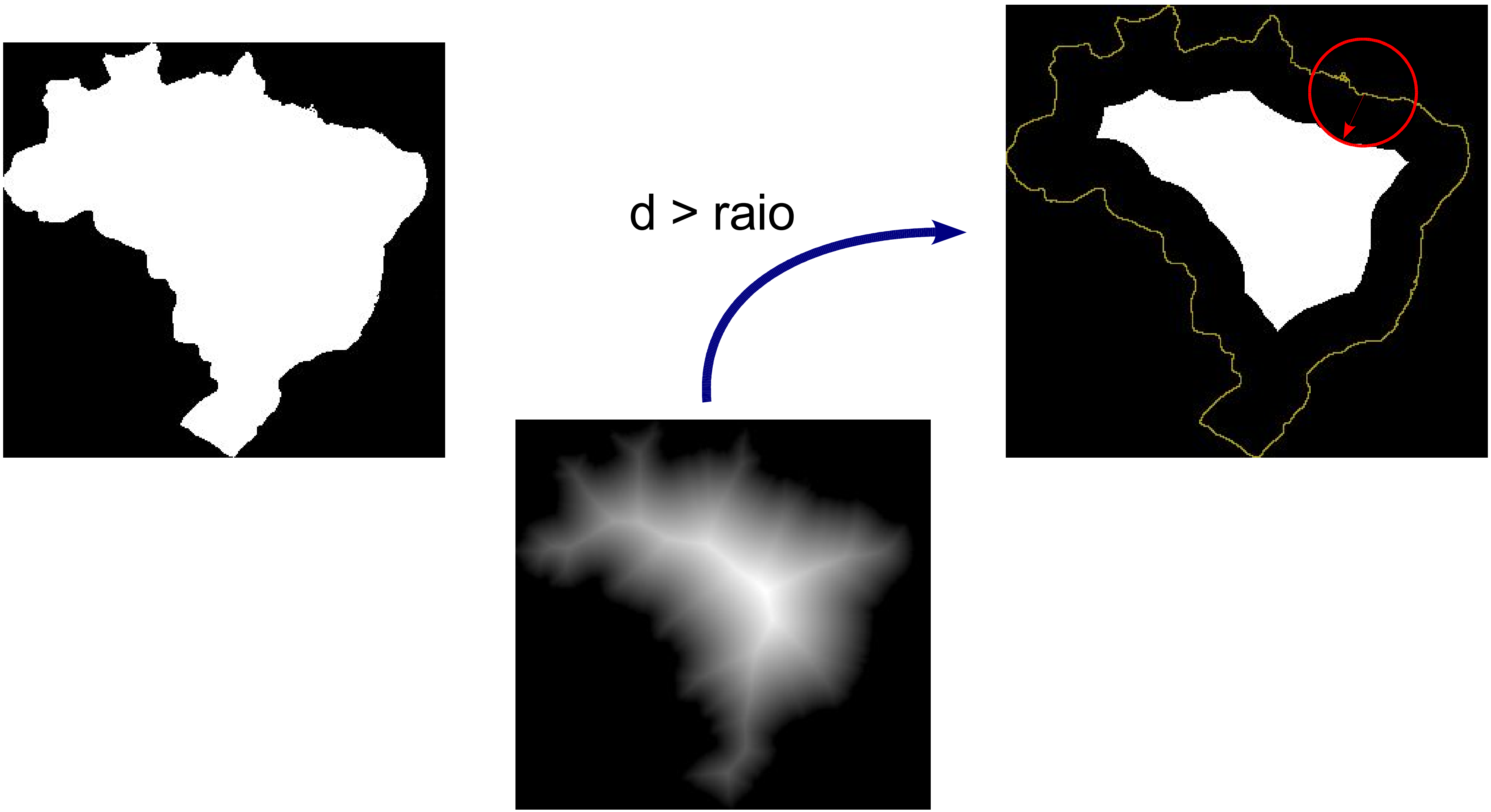
Dilatação



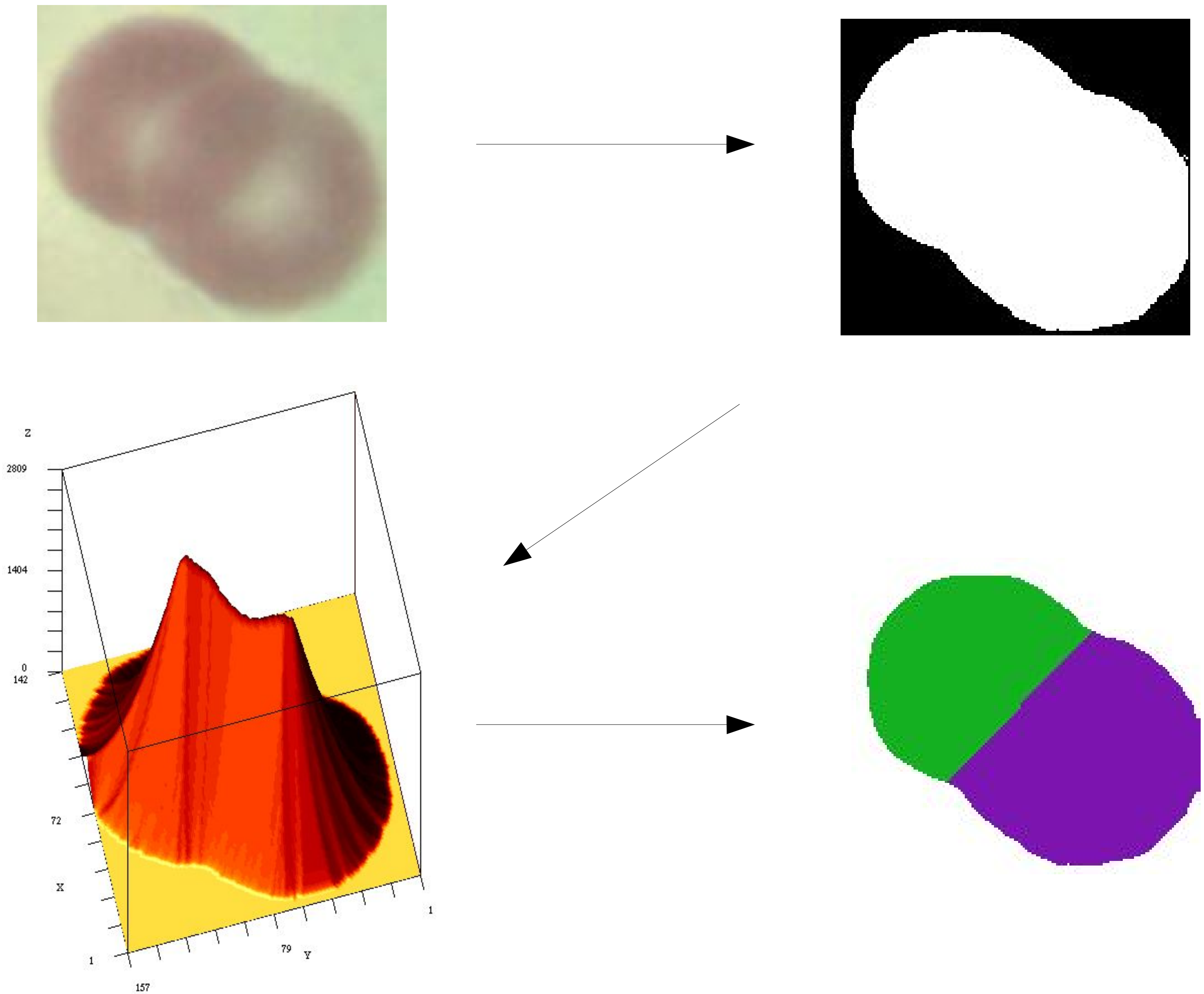
Erosão



Erosão



Separação de Objetos



Outras Aplicações

- Casamento de formas (*matching*)
- Navegação em Robótica
  - Caminhos mínimos
- *Image Registration*
- Imagens Médicas
- Esqueletos e Diagramas de Voronoi

Outras Aplicações

- Dimensão Fractal
- Medidas da forma
  - Largura máxima
- Classificação (*clustering*)
- Realce
- Ray-tracing
- Botânica
- Geologia

Motivação e Objetivos

Motivação

- Vários algoritmos de TDE recentes e complexos
- Não se sabe qual é o melhor
- Não se sabe ao certo quais são corretos
- Implementações pouco difundidas

Motivação

Causas:

- Algoritmos muito recentes e elaborados
- Descrições dos artigos é muito curta e abstrata
- Testes dos artigos são insuficientes e parciais
- Demonstrações não são tudo:
  - sempre podem conter erros sutis
- Pouca tradição de trabalhos de avaliação em P.I.

Motivação

- Desempenho dos Algoritmos Depende do Conteúdo
- Natureza dessa dependência é não-trivial
  - Número de pixels de interesse
  - Orientação
  - Espessura do objeto
  - Diversos fatores geométricos
  - Não se sabe ao certo quais fatores influenciam cada método!

Objetivos

- Estudar os recentes algoritmos de TDE
  - Organizar
  - Implementar
  - Comparar e Validar
    - Empiricamente e Teoricamente
- Principais Perguntas:
  - Quais os algoritmos mais rápidos?
  - Quais são exatos?
  - Qual a ordem de complexidade dos algoritmos?
  - Qual o algoritmo mais adequado a determinada tarefa?

Objetivos

- Pavimentar o caminho para:
  - Extensões a outras entidades
    - Diagramas de Voronoi / Esqueletos
    - Segmentação (outras métricas)
  - Correções, demonstrações
  - Novos algoritmos

Algoritmos

Força-Bruta

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

– Para cada pixel "1" da imagem, encontrar a mínima distância aos pixels "0"

Força-Bruta

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

Força-Bruta

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

Força-Bruta

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

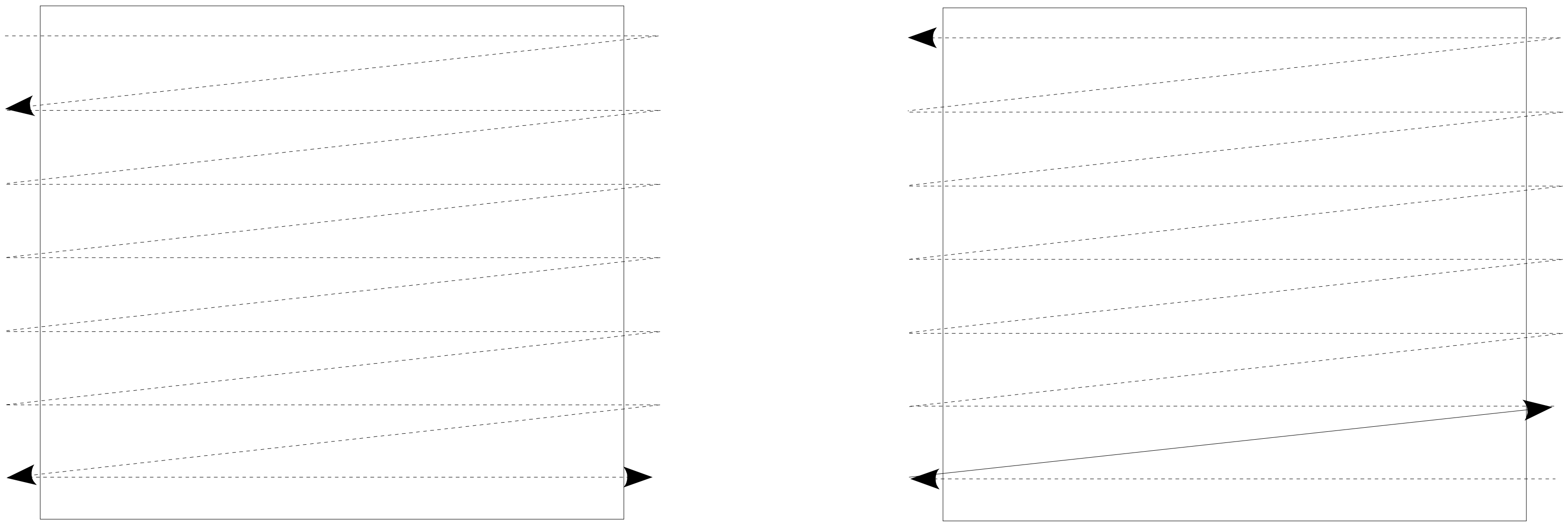
– Número de operações:
  - $\Omega(n^2)$ e $O(n^4)$

Otimizando a TD

- Aproveitar propriedades locais
  - Deduzir distâncias de um pixel a partir dos seus vizinhos
- Algoritmos $O(n^2)$ para métricas não-euclidianas
  - Desde 1966!
- Problema:
  - Métrica euclidiana x Domínio discreto
  - Algoritmos euclidianos $O(n^2)$
    - Apenas nos anos 90!

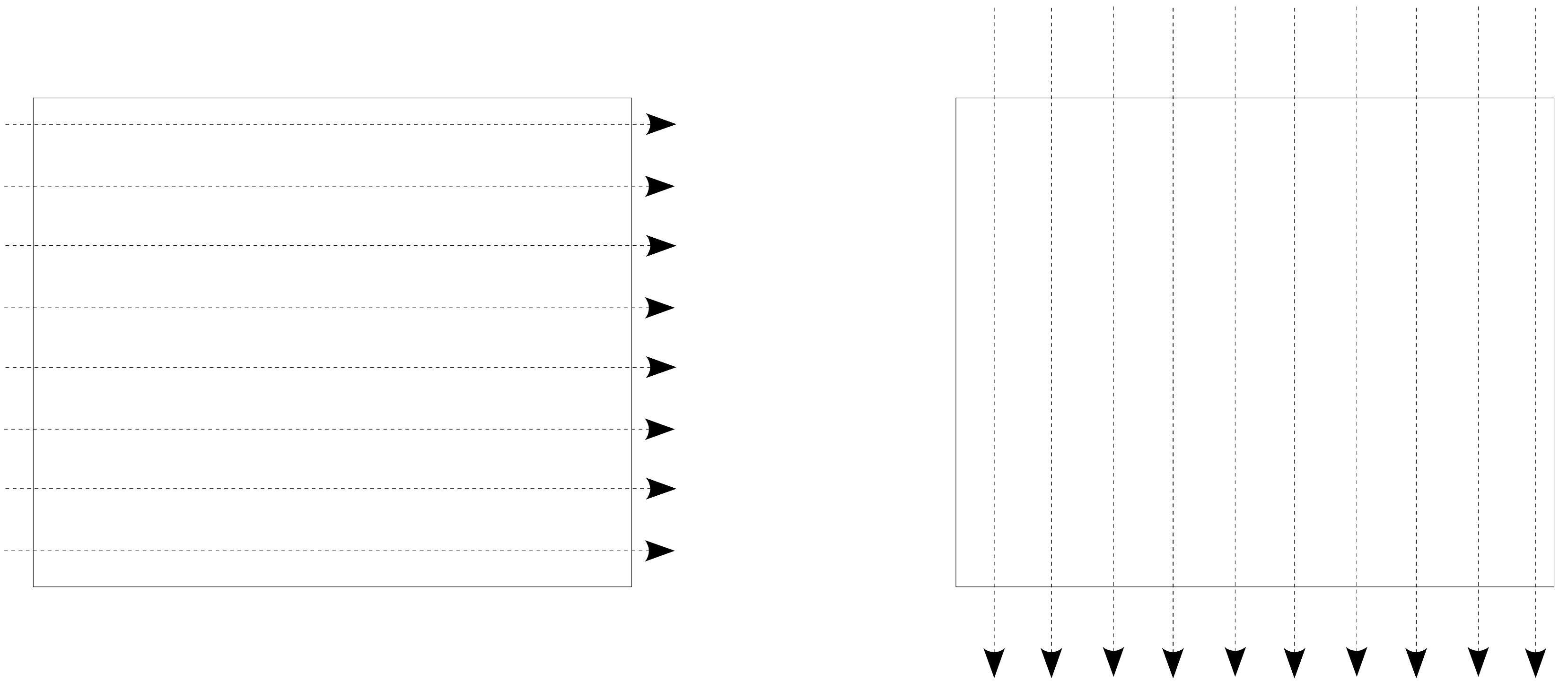
Tipos de Algoritmos

- Varredura Raster



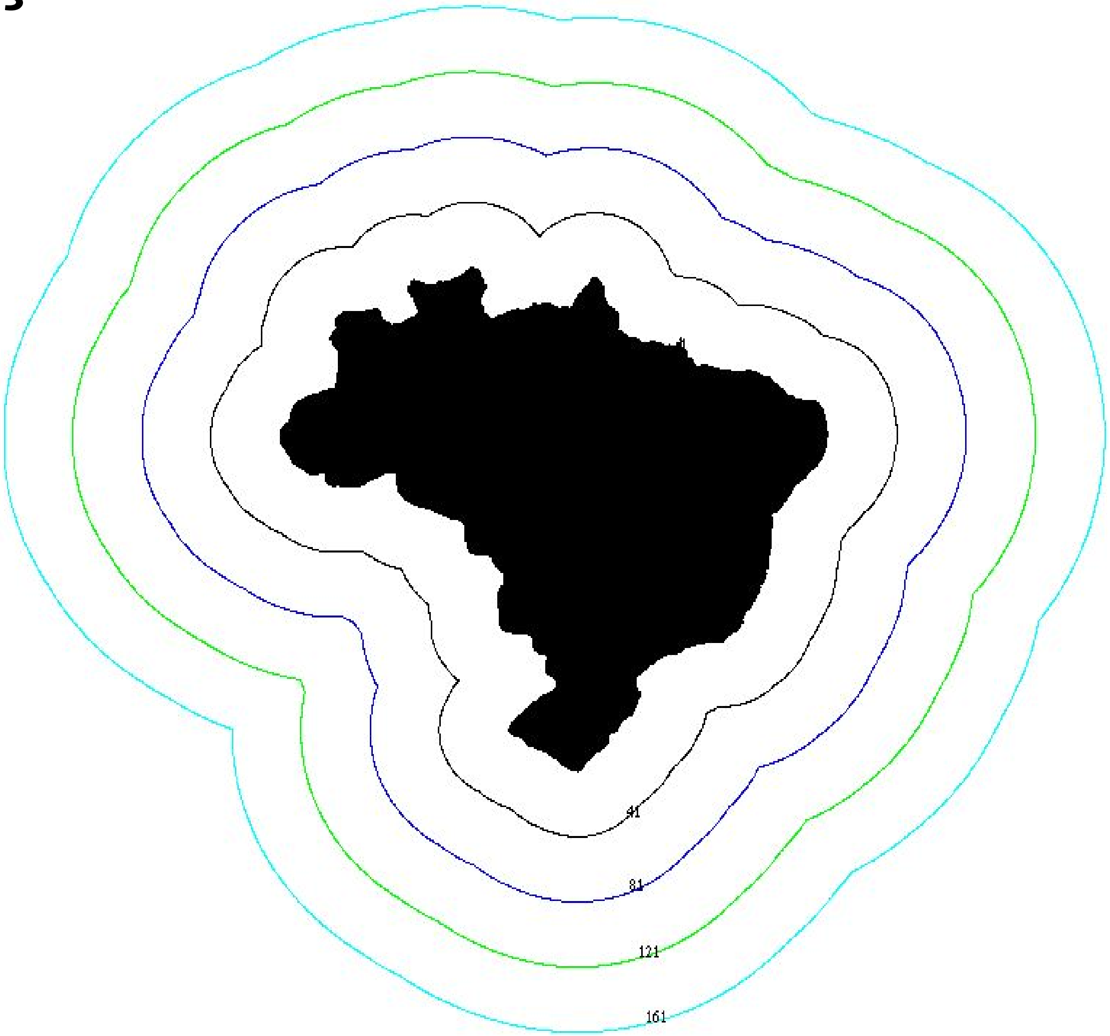
Tipos de Algoritmos

- Varredura Independente



Tipos de Algoritmos

- Propagação Ordenada



TDE por

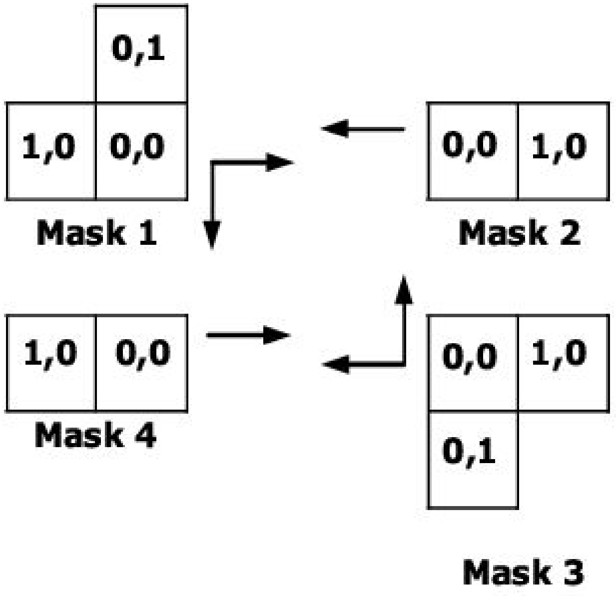
Varredura Raster

TDs Não Euclidianas

- Antepassados das TDEs eficientes
- Uso de máscaras de operação local
- Número fixo de passadas na imagem
- Rosenfeld 1966
  - *Cityblock*, *Chessboard*, Hexagonal, Octogonal
  - Úteis, porém muito distantes da Euclidiana
- Borgefors 1984
  - Métricas “Chamfer”
  - Pesos das máscaras escolhidos para aproximação ótima da TDE em 2 passadas

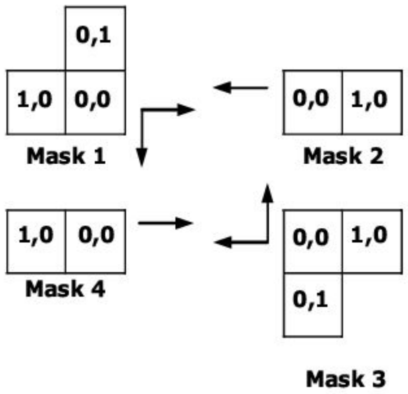
TDE de Danielsson

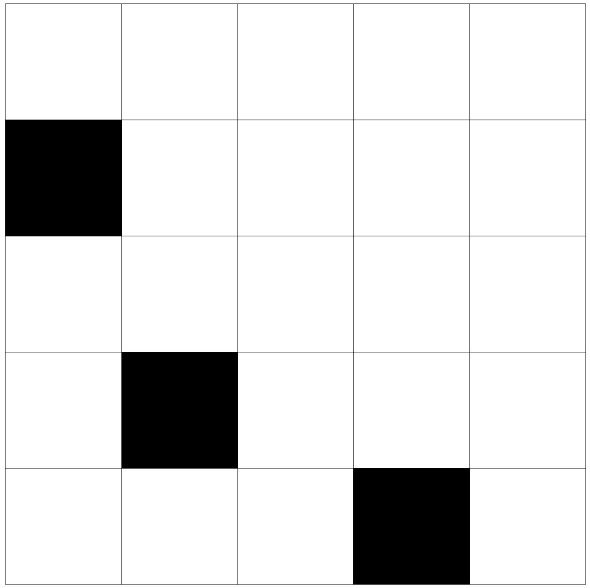
- Primeiro algoritmo eficiente (1980)
- O método euclidiano mais famoso
- Varredura Raster
- Trabalha com coordenadas (vetores)



TDE de Danielsson

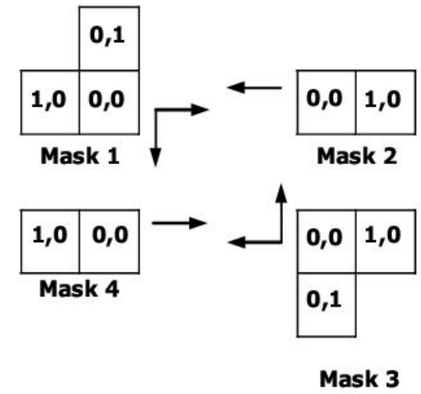
- Varredura Raster

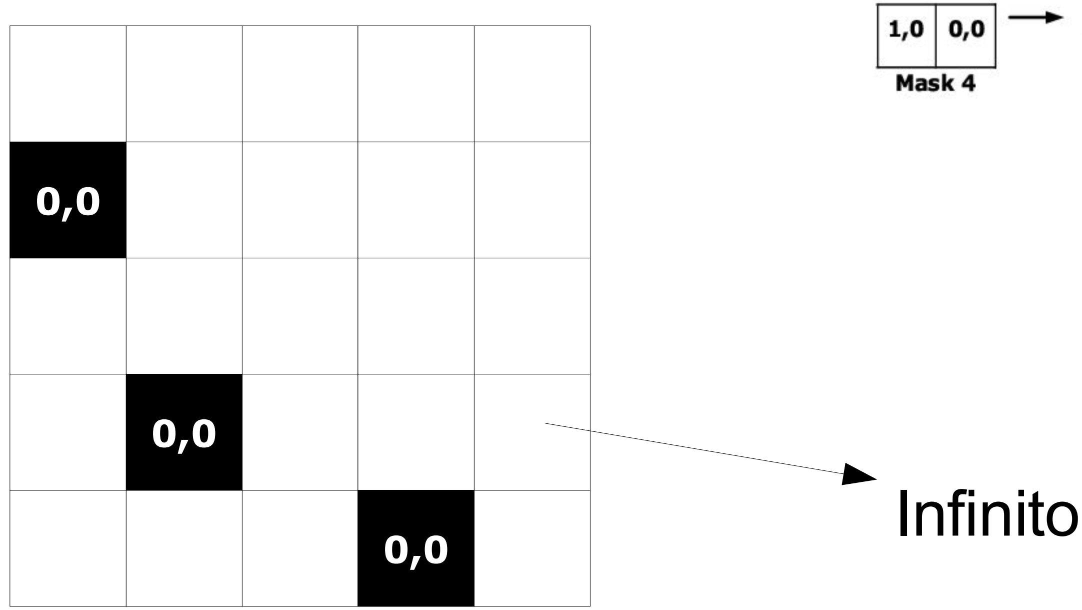




TDE de Danielsson

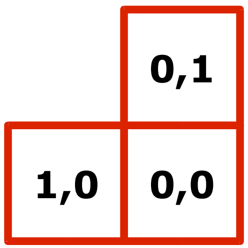
- Varredura Raster

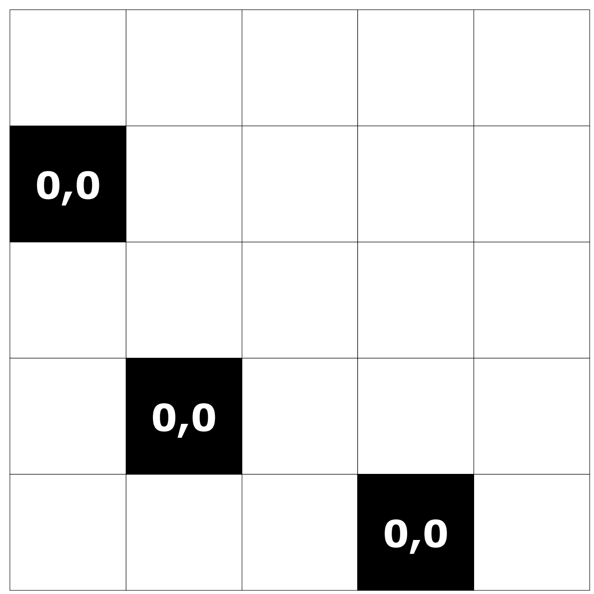


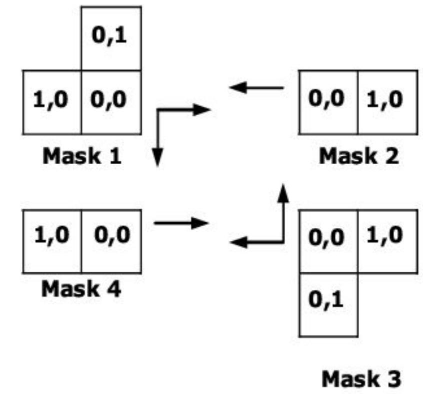


TDE de Danielsson

- Varredura Raster

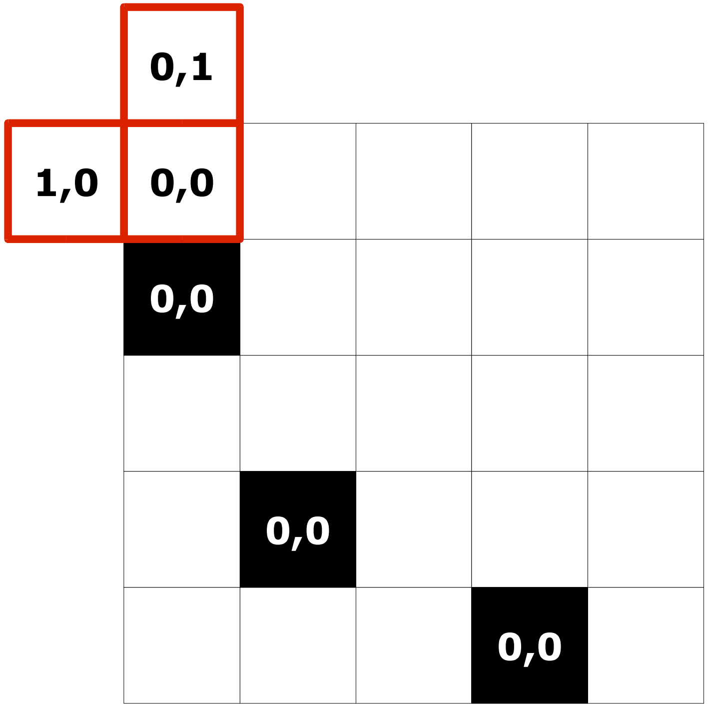


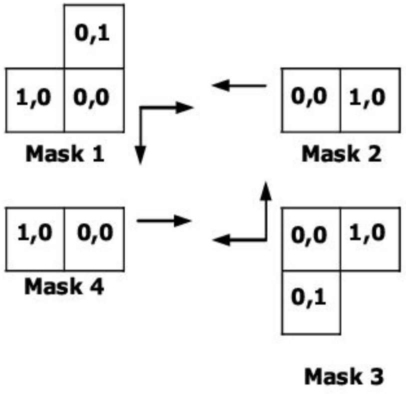




TDE de Danielsson

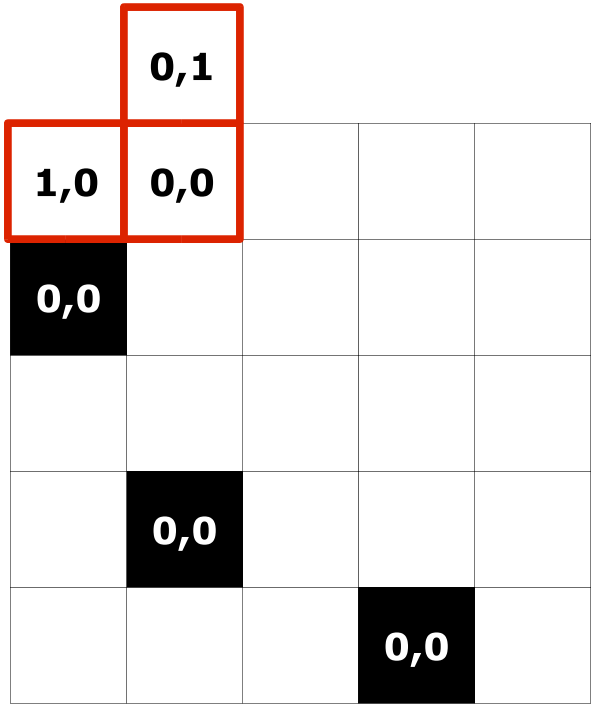
- Varredura Raster

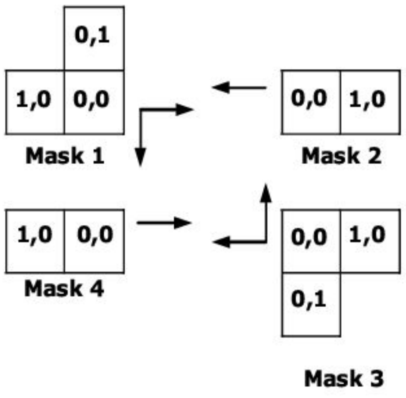




TDE de Danielsson

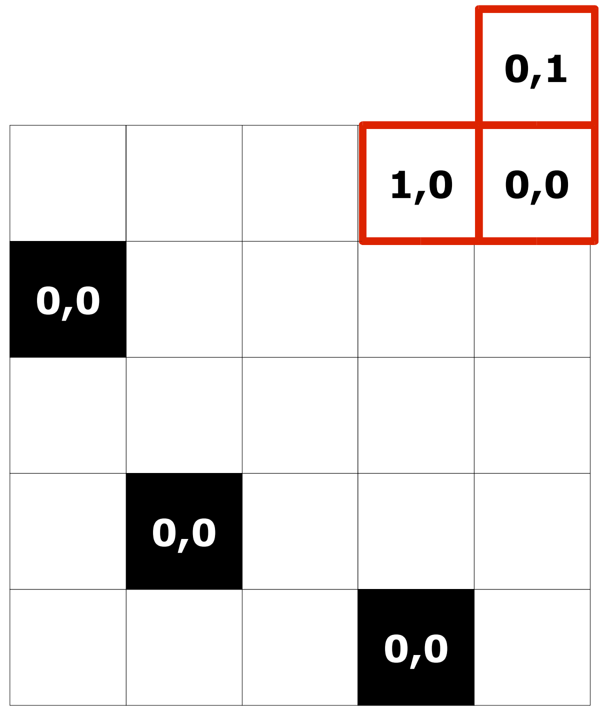
- Varredura Raster

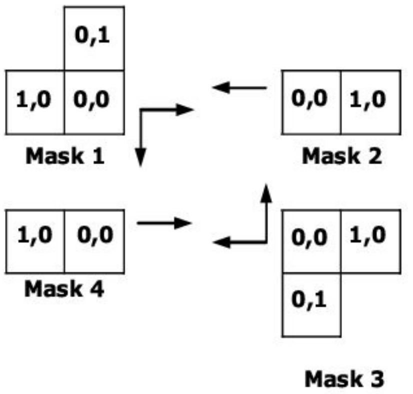




TDE de Danielsson

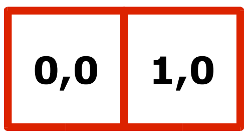
- Varredura Raster

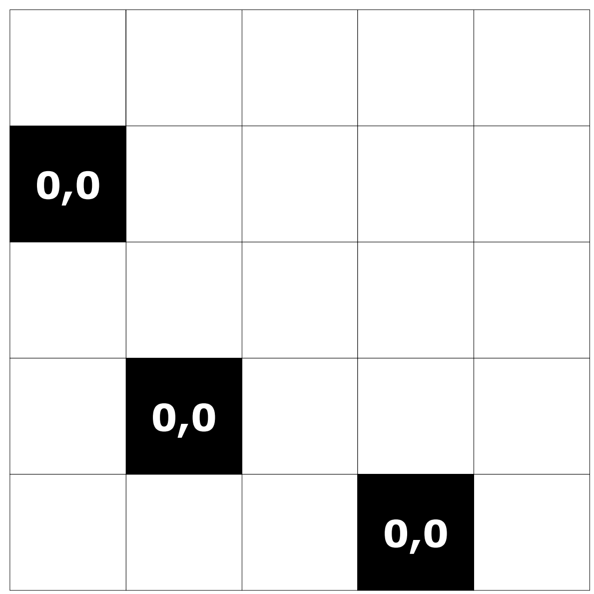


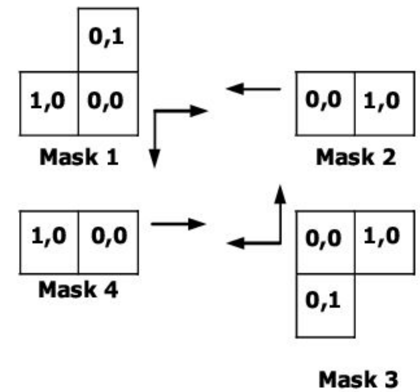


TDE de Danielsson

- Varredura Raster

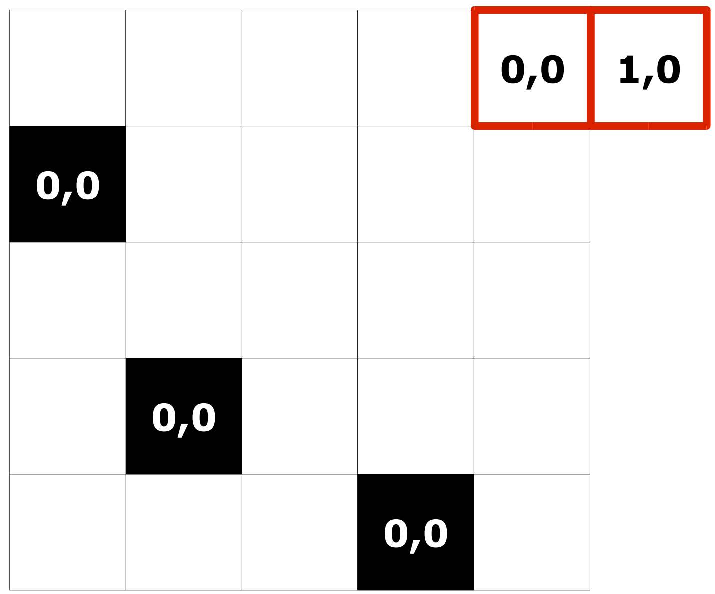


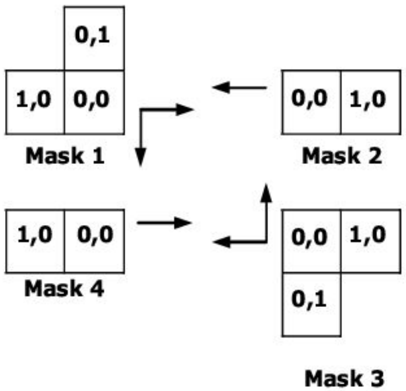




TDE de Danielsson

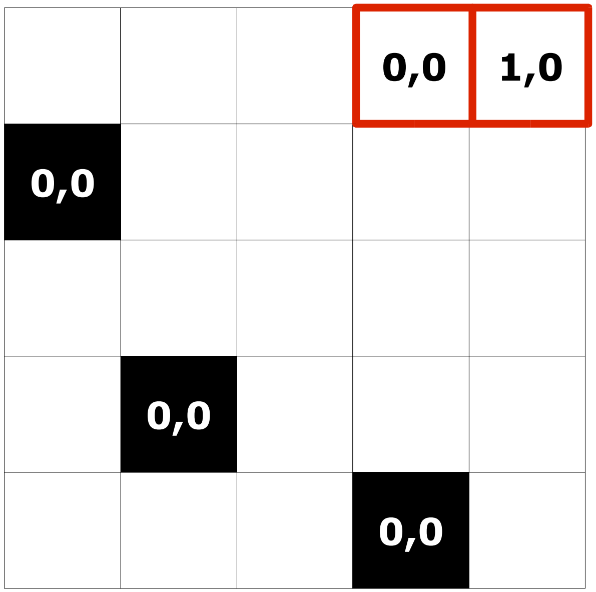
- Varredura Raster

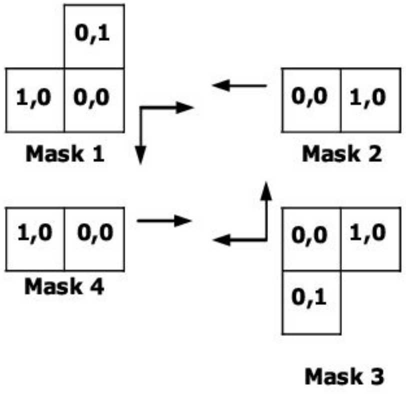




TDE de Danielsson

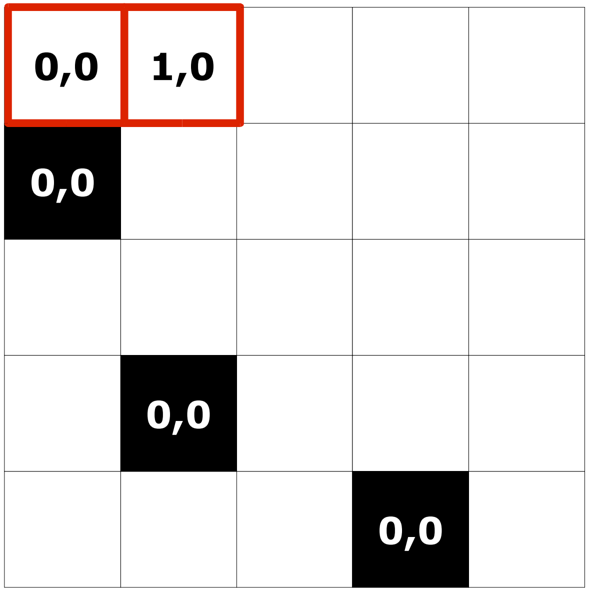
- Varredura Raster

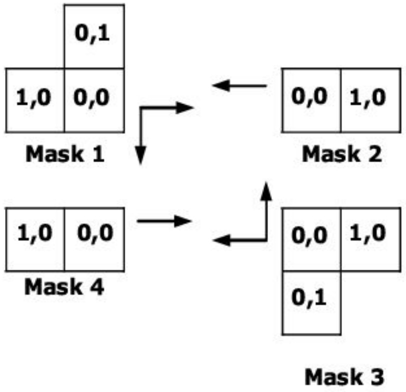




TDE de Danielsson

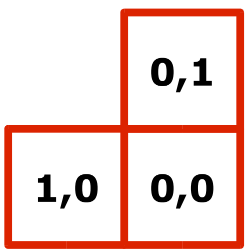
- Varredura Raster

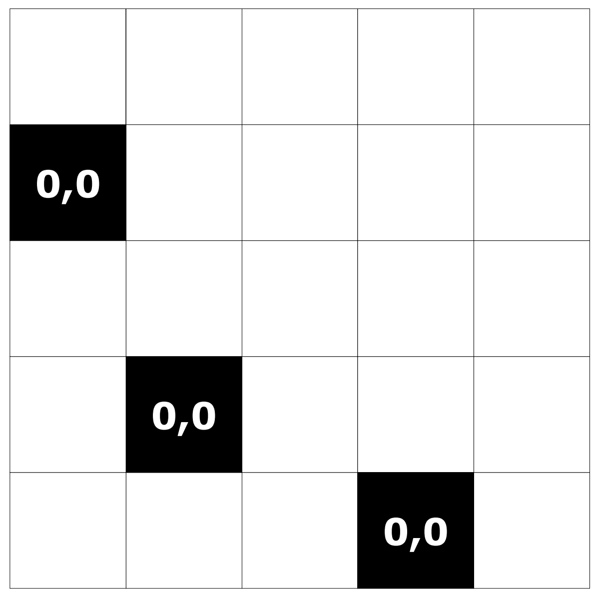


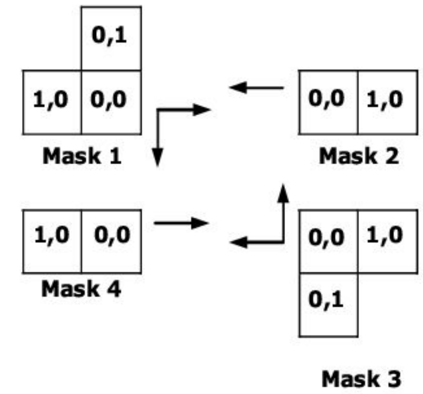


TDE de Danielsson

- Varredura Raster

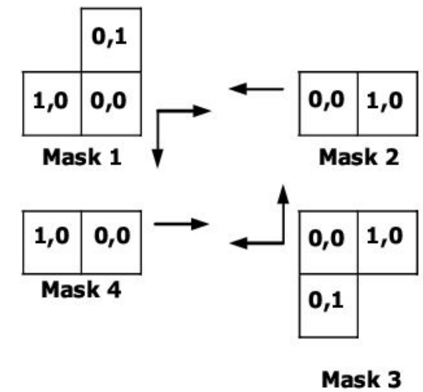






TDE de Danielsson

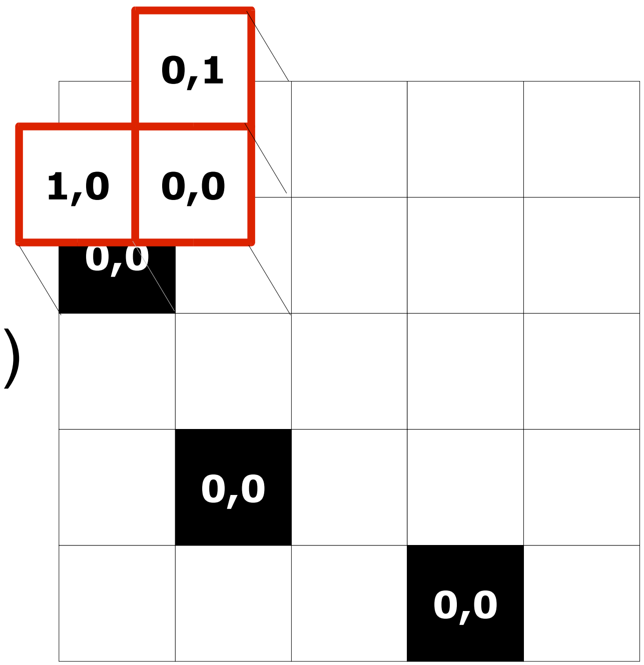
- Varredura Raster



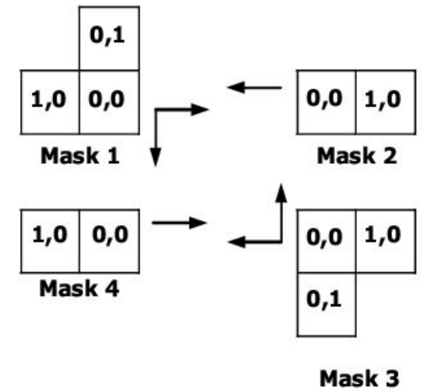
- Soma e guarda o menor valor

TDE de Danielsson

- Varredura Raster

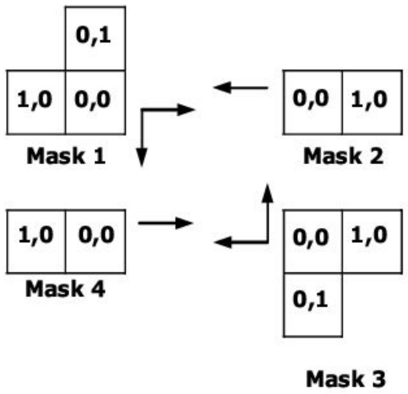


(1,0) + (0,0) = (1,0)

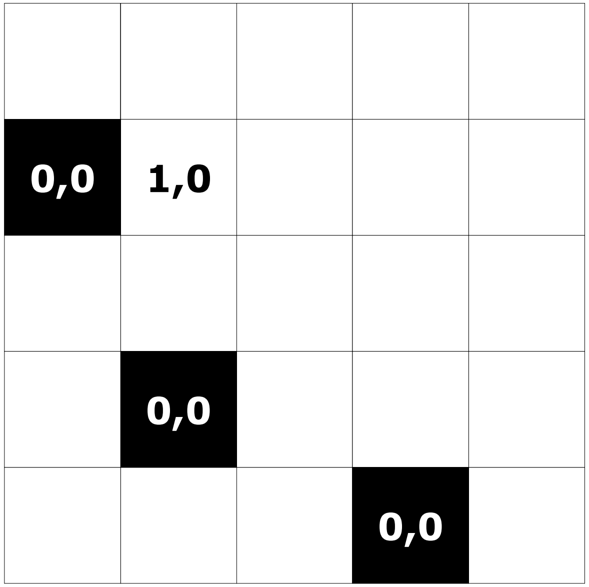


TDE de Danielsson

- Varredura Raster



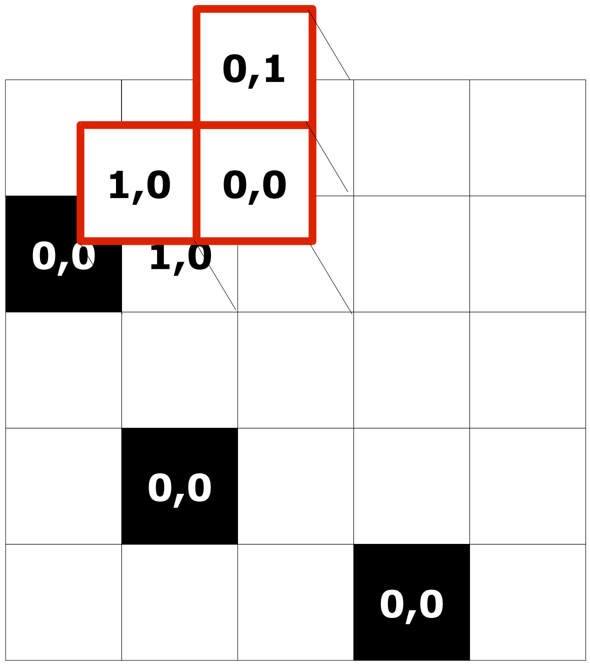
$(1,0) + (0,0) = (1,0)$

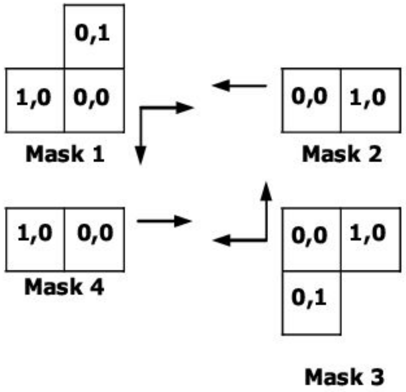


TDE de Danielsson

- Varredura Raster

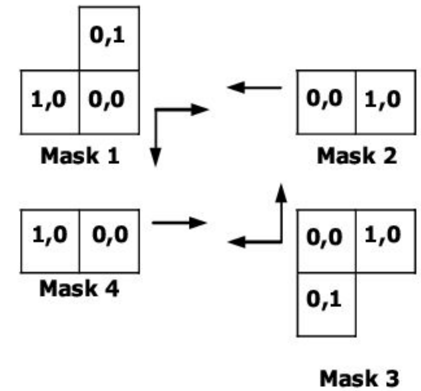
(1,0) + (1,0) = (2,0)



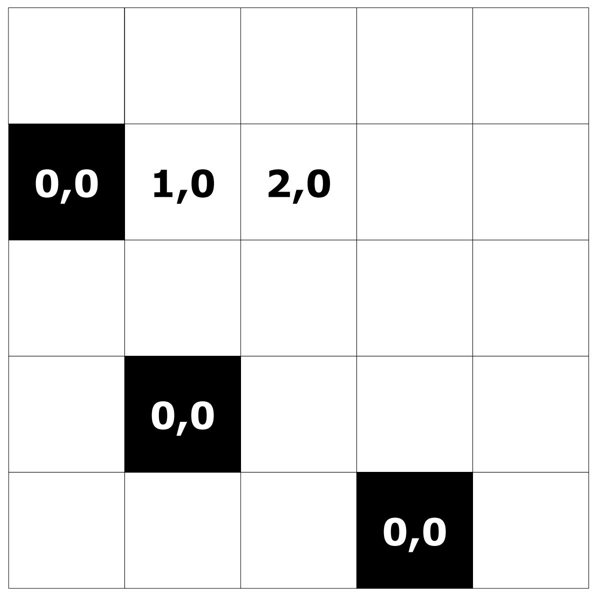


TDE de Danielsson

- Varredura Raster

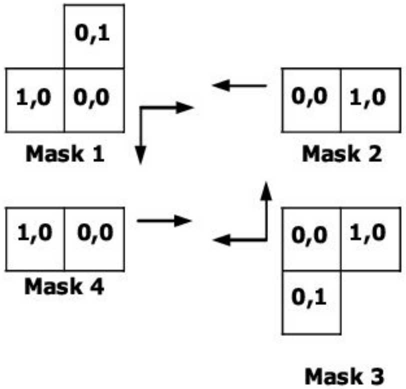


$(1,0) + (1,0) = (2,0)$

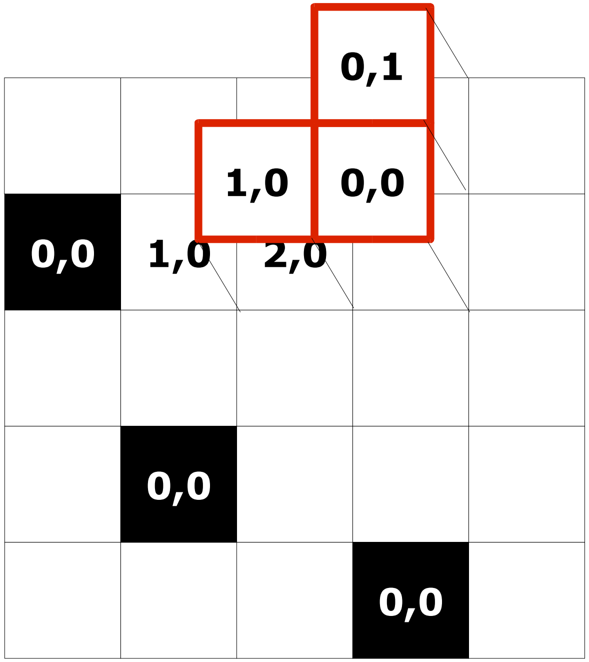


TDE de Danielsson

- Varredura Raster

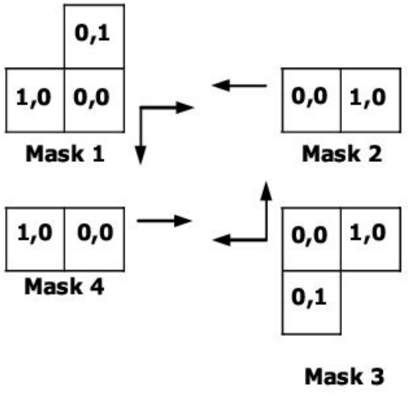


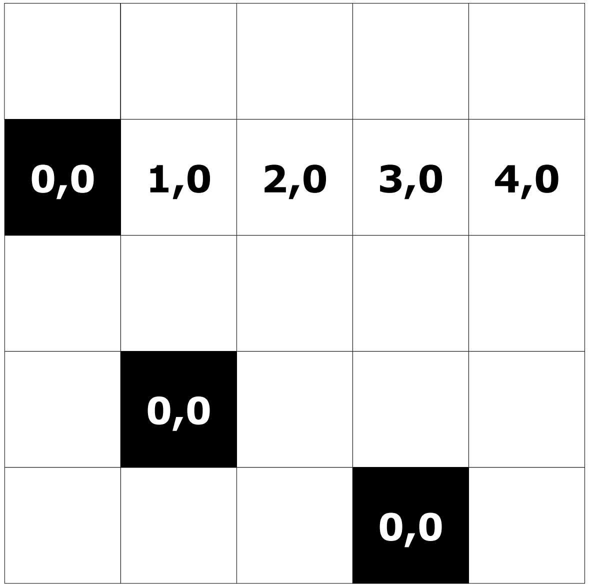
(2,0) + (1,0) = (3,0)



TDE de Danielsson

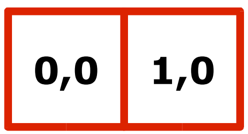
- Varredura Raster

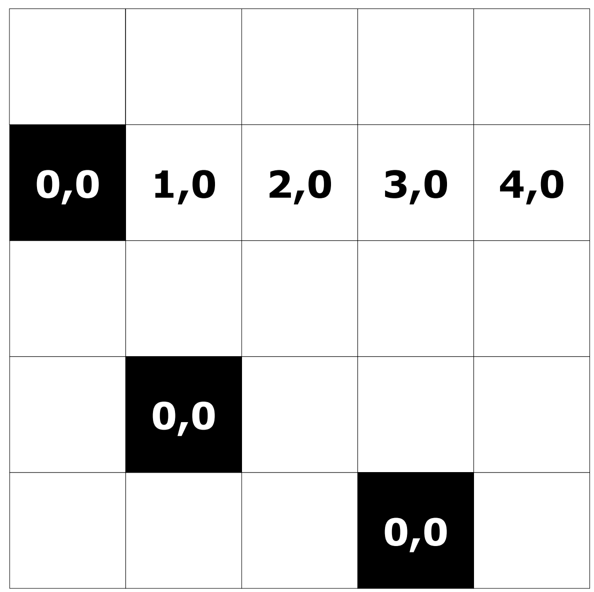


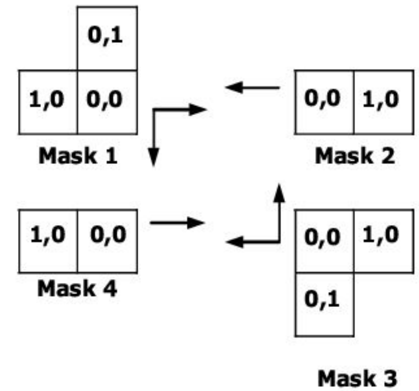


TDE de Danielsson

- Varredura Raster

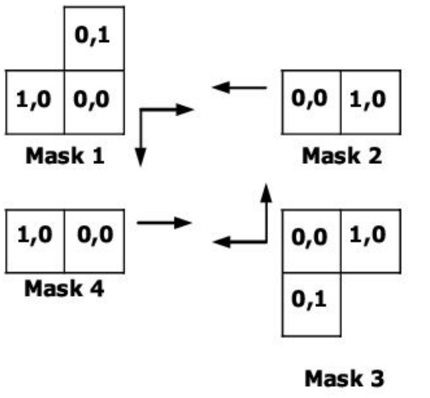


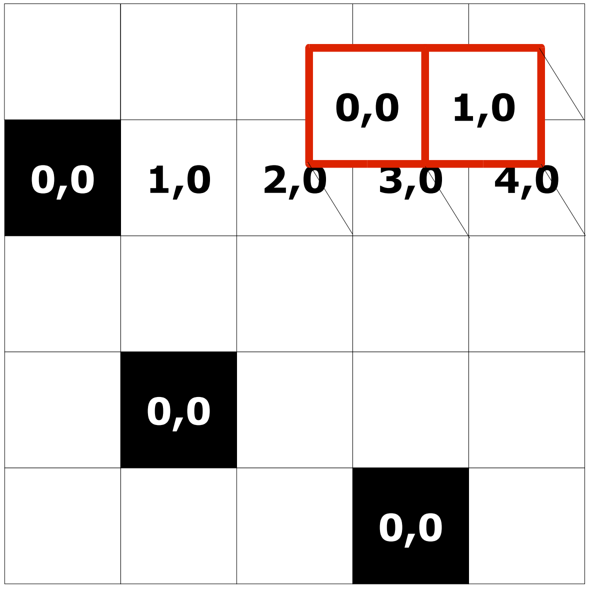




TDE de Danielsson

- Varredura Raster

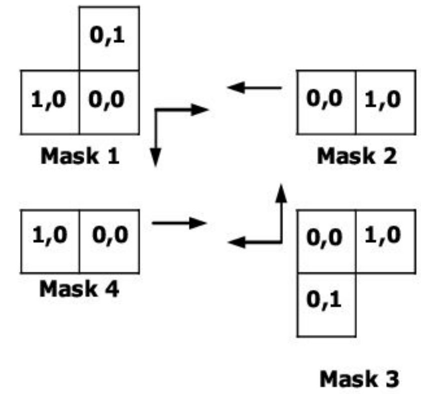


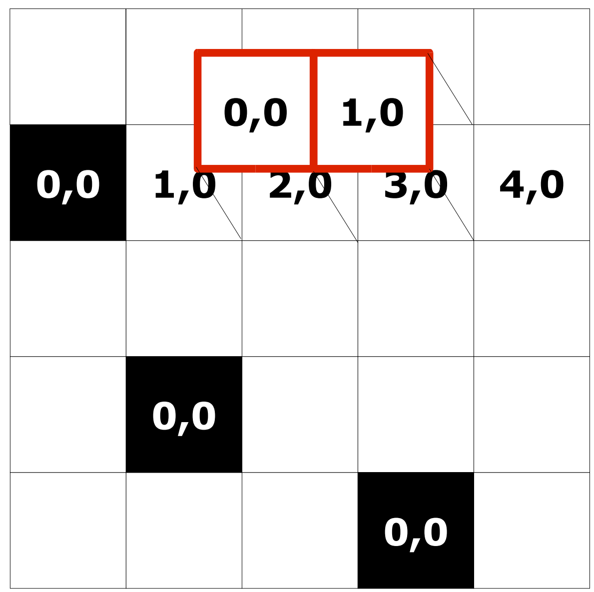


$||(3,0)|| < ||(1,0) + (4,0)||$

TDE de Danielsson

- Varredura Raster

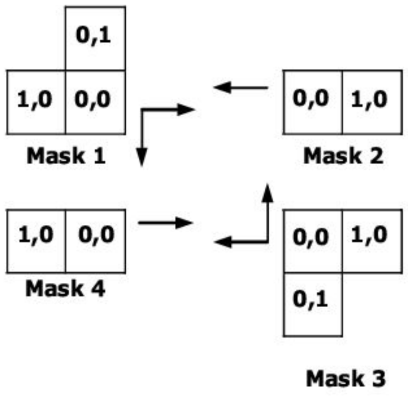


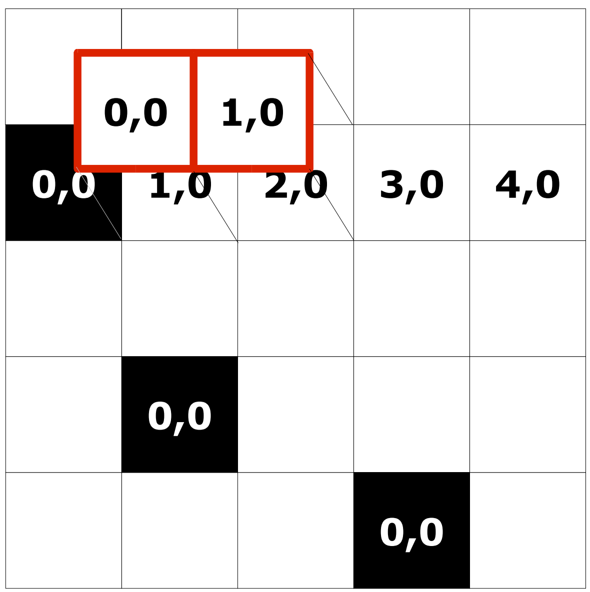


||(2,0)|| < ||(1,0) + (3,0)||

TDE de Danielsson

- Varredura Raster

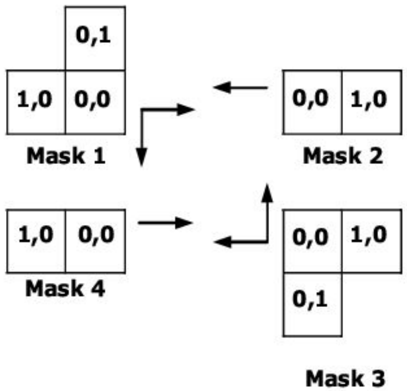




||(1,0)|| < ||(1,0) + (2,0)||

TDE de Danielsson

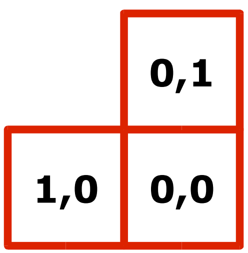
- Varredura Raster

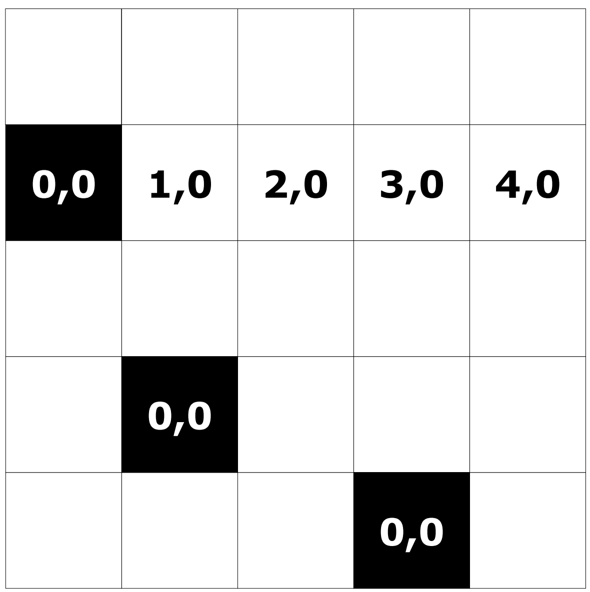


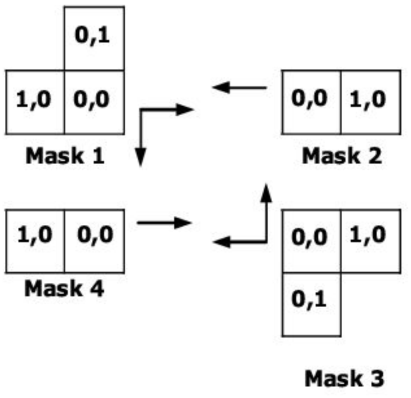
||(0,0)|| < ||(1,0) + (1,0)||

TDE de Danielsson

- Varredura Raster

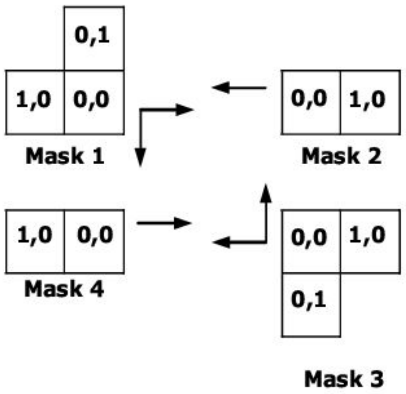


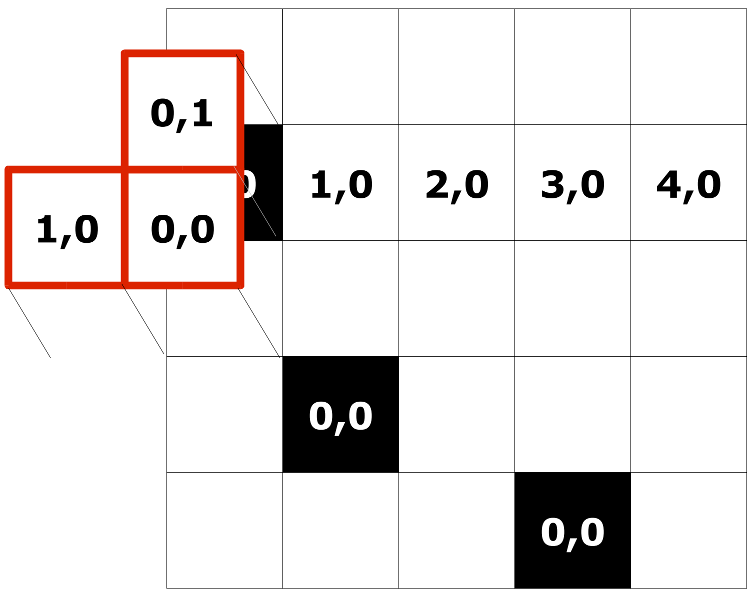




TDE de Danielsson

- Varredura Raster

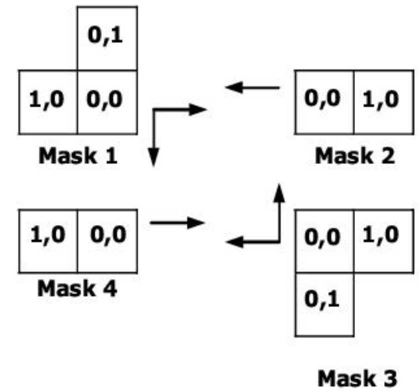


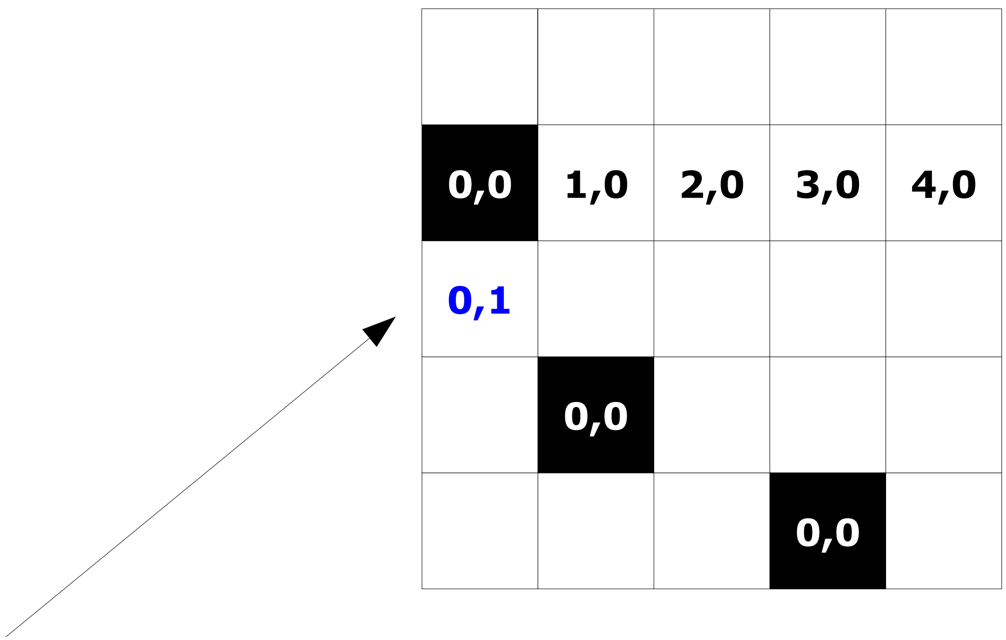


$$\|(0,1) + (0,0)\| < \|(0,0) + (\text{inf},\text{inf})\|$$

TDE de Danielsson

- Varredura Raster

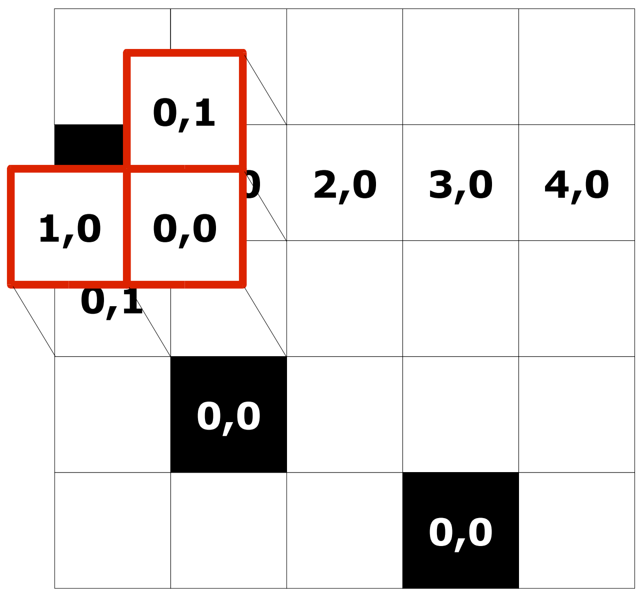


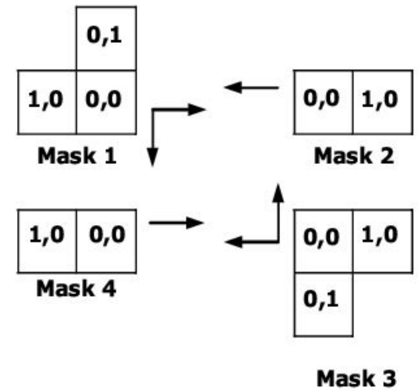


||(0,1)|| < infinito

TDE de Danielsson

- Varredura Raster

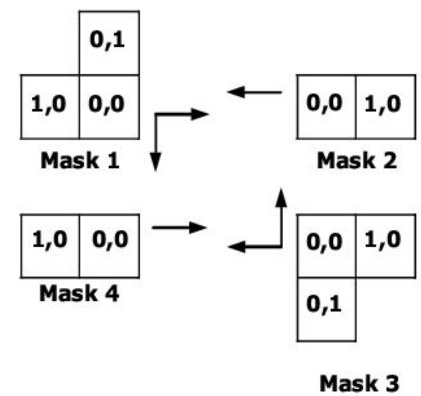




(1,0) + (0,1) = (1,1)

TDE de Danielsson

- Varredura Raster

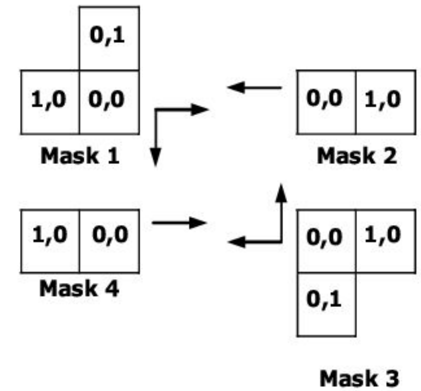


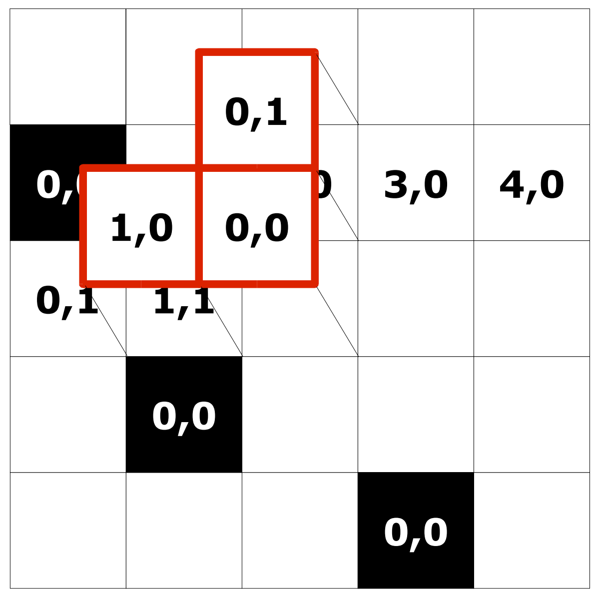
0,0	1,0	2,0	3,0	4,0
0,1	1,1			
	0,0			
			0,0	

(1,0) + (0,1) = (1,1)

TDE de Danielsson

- Varredura Raster

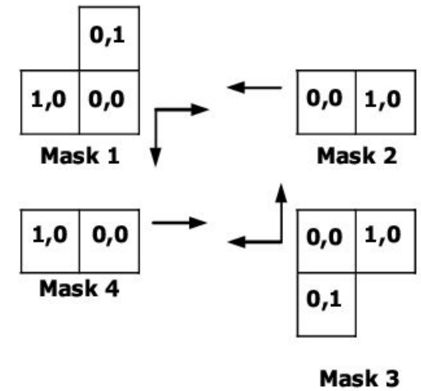




(1,0) + (1,1) = (2,1)
(0,1) + (2,0) = (2,1)

TDE de Danielsson

- Varredura Raster

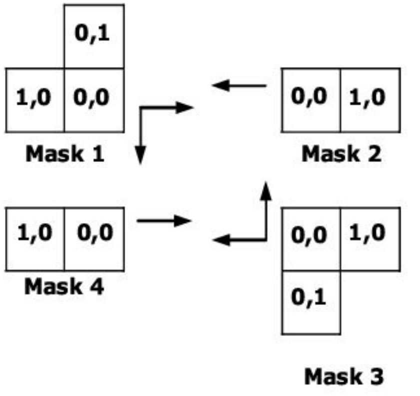


0,0	1,0	2,0	3,0	4,0
0,1	1,1	2,1		
	0,0			
			0,0	

(1,0) + (1,1) = (2,1)
(0,1) + (2,0) = (2,1)

TDE de Danielsson

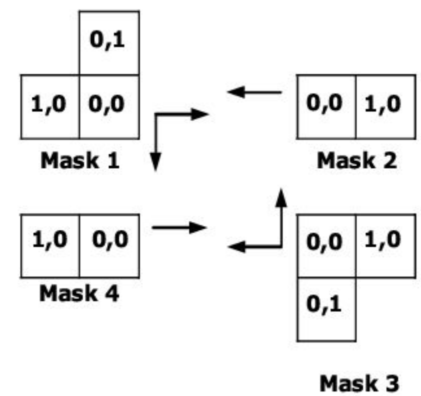
- Varredura Raster



0,0	1,0	2,0	3,0	4,0
0,1	1,1	2,1	3,1	4,1
	0,0			
			0,0	

TDE de Danielsson

- Varredura Raster

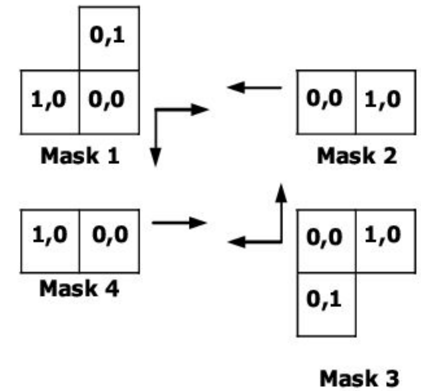


0,0	1,0	2,0	3,0	4,0
0,1	1,1	2,1	3,1	4,1
0,2	0,0			
			0,0	

(0,1) + (0,1) = (0,2)

TDE de Danielsson

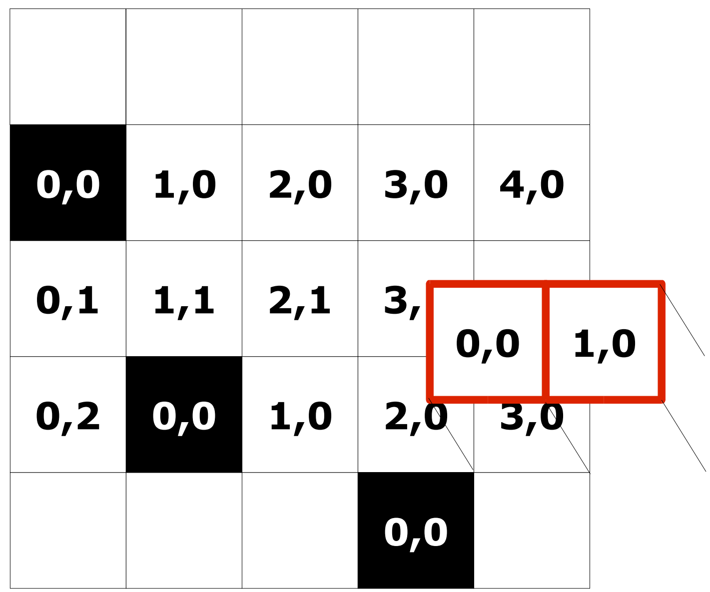
- Varredura Raster

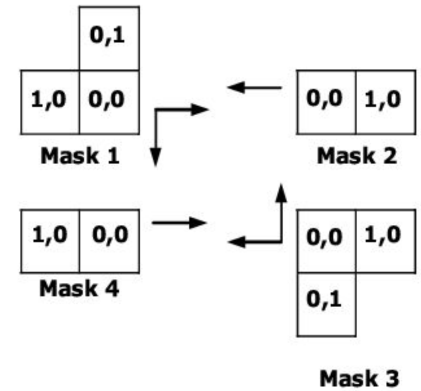


0,0	1,0	2,0	3,0	4,0
0,1	1,1	2,1	3,1	4,1
0,2	0,0	1,0	2,0	3,0
			0,0	

TDE de Danielsson

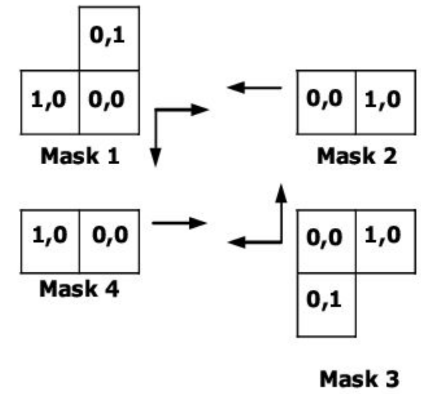
- Varredura Raster

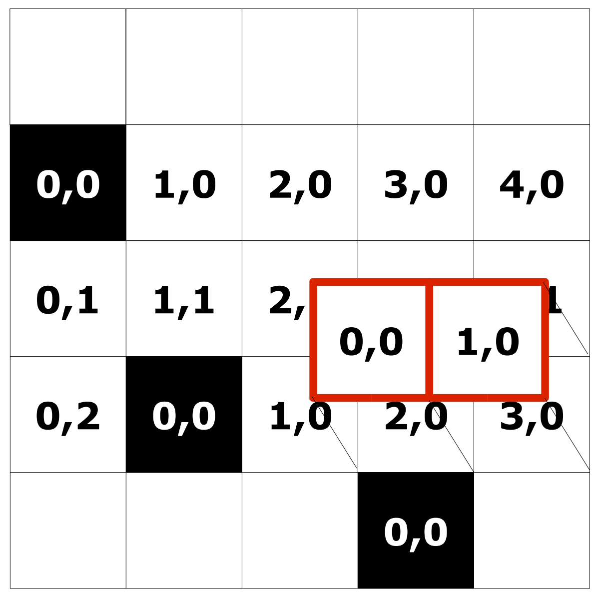




TDE de Danielsson

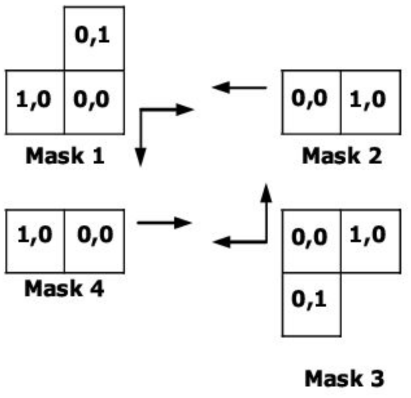
- Varredura Raster

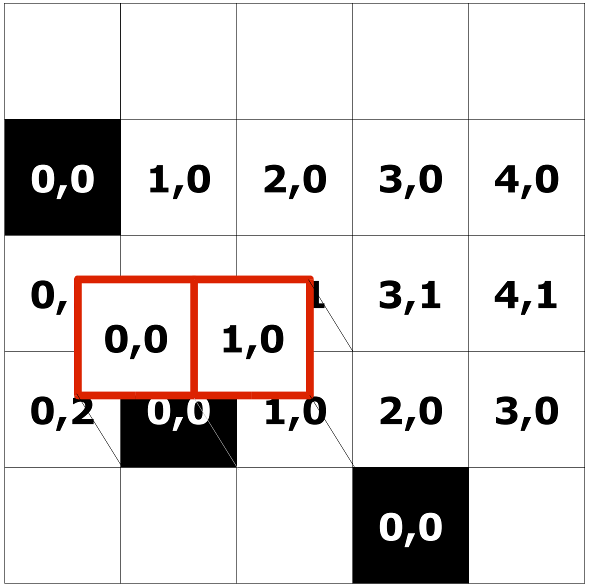




TDE de Danielsson

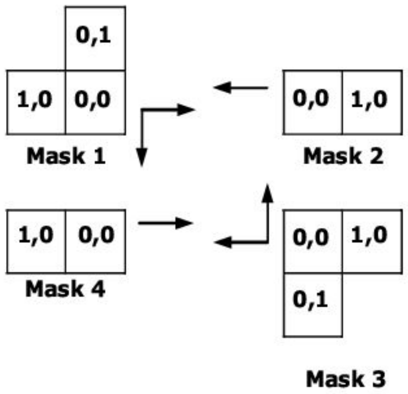
- Varredura Raster

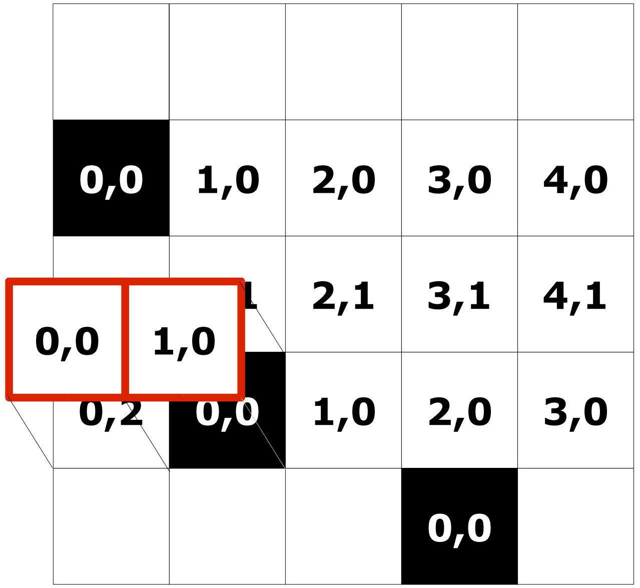




TDE de Danielsson

- Varredura Raster

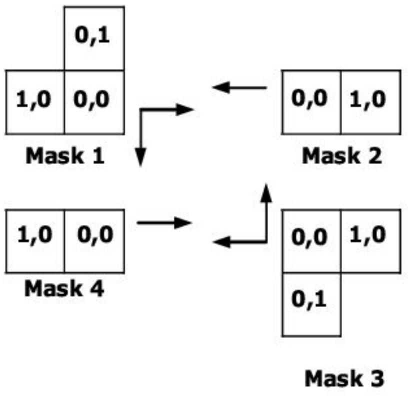




||(0,0) + (0,2)|| < ||(1,0) + (0,0)||

TDE de Danielsson

- Varredura Raster



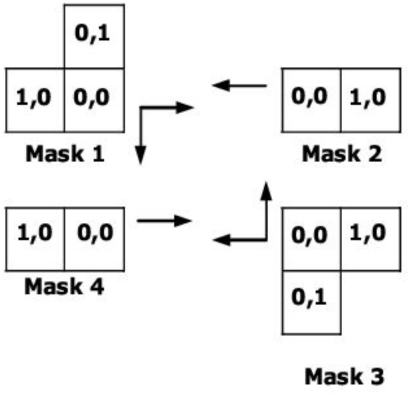
0,0	1,0	2,0	3,0	4,0
0,1	1,1	2,1	3,1	4,1
1,0	0,0	1,0	2,0	3,0
			0,0	

||(0,2)|| < ||(1,0)||

TDE de Danielsson

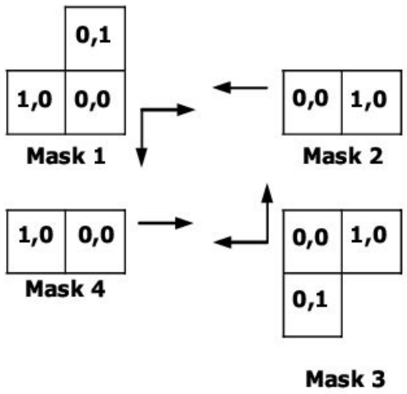
- Varredura Raster

0,0	1,0	2,0	3,0	4,0
0,1	1,1	2,1	3,1	4,1
1,0	0,0	1,0	2,0	3,0
1,1	0,1	1,1	0,0	1,0



TDE de Danielsson

- Varredura Raster

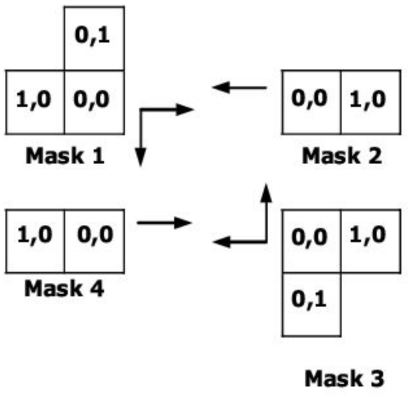


0,0	1,0	2,0	3,0	4,0
0,1	1,1	2,1	3,1	4,1
1,0	0,0	1,0	2,0	3,0
1,1	0,1	1,0	0,0	1,0

TDE de Danielsson

- Máscaras 3 e 4 de baixo para cima

0,1	1,1	2,1	3,1	4,1
0,0	1,0	2,0	3,0	1,3
0,1	0,1	1,1	0,2	1,2
1,0	0,0	1,0	0,1	1,1
1,1	0,1	1,0	0,0	1,0



TDE de Danielsson

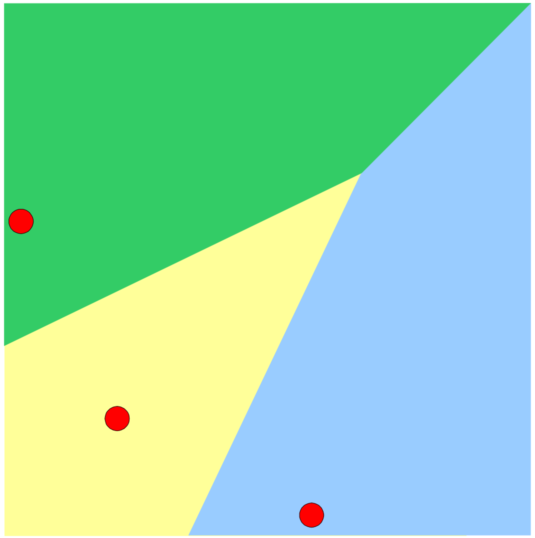
- $d^2 = x^2 + y^2$

1	2	5	10	17
0	1	4	9	10
1	1	2	4	5
1	0	1	1	2
2	1	1	0	1

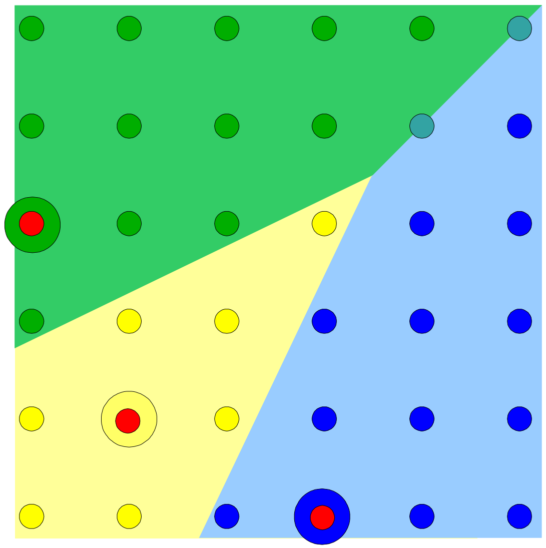
TDE de Danielsson

- 4 passadas em n^2 pixels
  - Complexidade $O(n^2)$
- Problema:
  - Resultado Inexato!

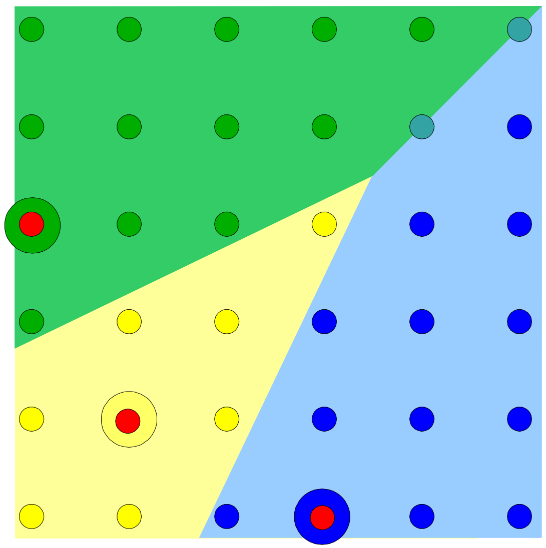
Erros das TDEs



Erros das TDEs

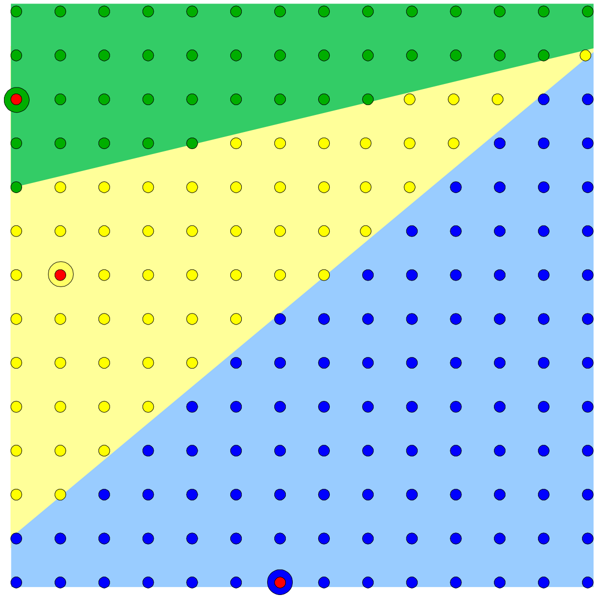


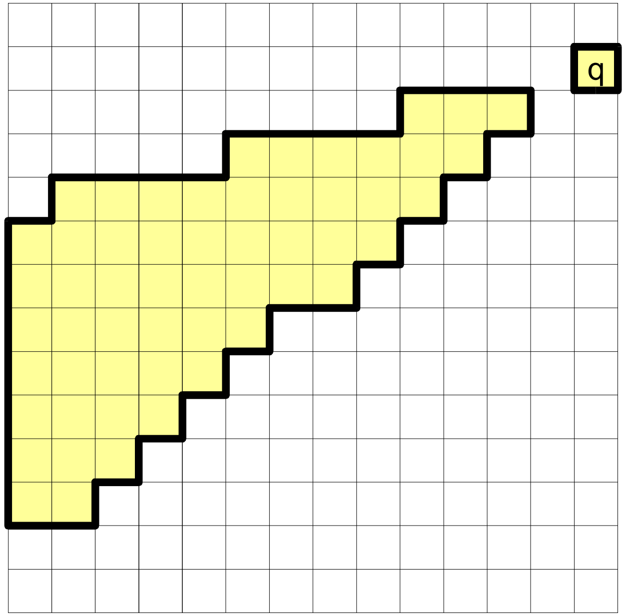
Erros das TDEs



4	5	8	13	20	29
1	2	5	10	17	20
p1	1	4	q	10	13
1	1	2	4	5	8
1	p2	1	1	2	5
2	1	1	p3	1	4

Erros das TDEs





TDEs Exatas por Varredura Raster

- **1999: Cuisenaire**
- Correções sobre Danielsson
- Restaura conectividade das RVs
- Não foi provado se é exato ou linear
- Apenas alguns testes empíricos

TDEs Exatas por Varredura Raster

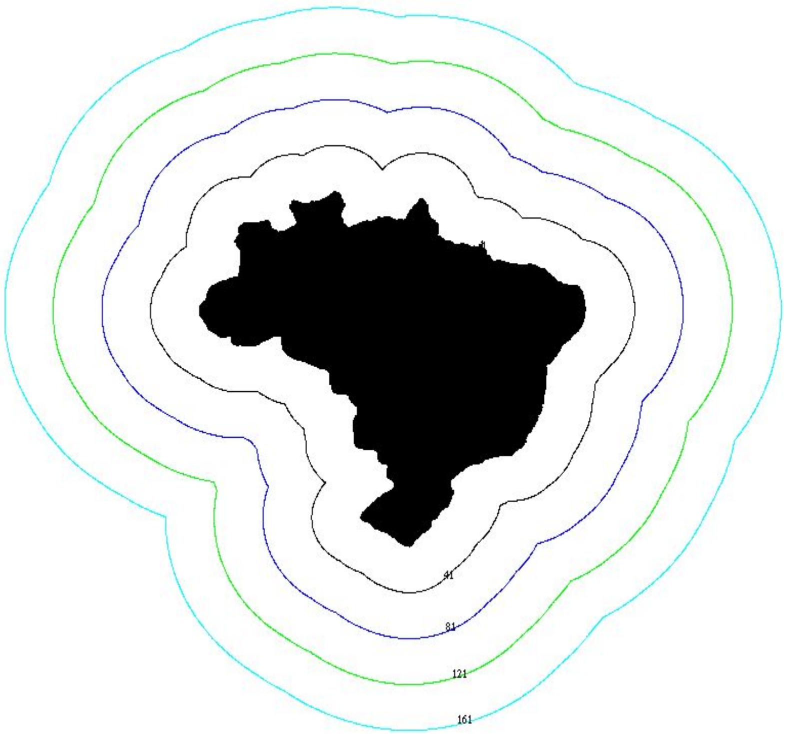
- **2004: Shih**
- Máscara 3x3
- 2 passadas
- Demonstrações de corretude e complexidade
- Nenhum teste empírico
- Nenhuma comparação com outros métodos

Algoritmos de

Propagação

TDEs por Propagação

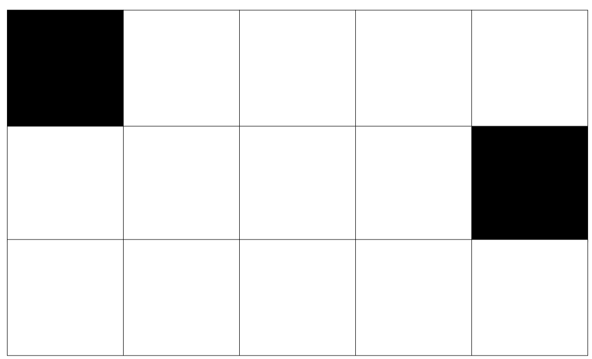
- A partir dos pixels pretos, propagar distâncias
- Base da Implementação:
  - Algoritmo de Dijkstra



Algoritmo Básico

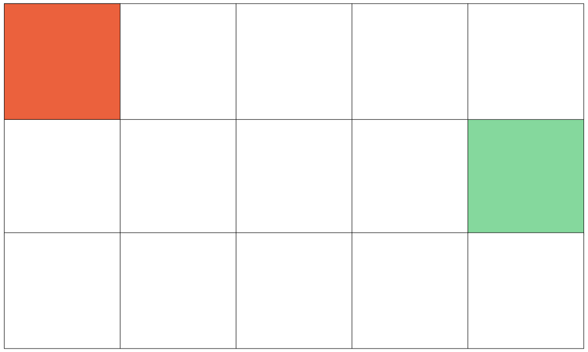
1. Inicialize a distância de todo pixel branco para um valor suficientemente alto.
2. Inicialize um conjunto auxiliar de pixels, denominado *Conjunto de Contorno*, para armazenar os pixels de fronteira.
3. Enquanto o Conjunto de Contorno não está vazio, faça:
   (a) Remova um pixel do Conjunto de Contorno, denominado *pixel central*.
   (b) Para cada vizinho branco do pixel central, faça:
      i. Calcule uma nova distância para o vizinho, baseando-se na distância do pixel central.
      ii. Se esta distância nova for menor que a distância corrente do vizinho:
         - atualize sua distância corrente como sendo a menor.
         - coloque esse vizinho no Conjunto de Contorno.

Algoritmo Básico

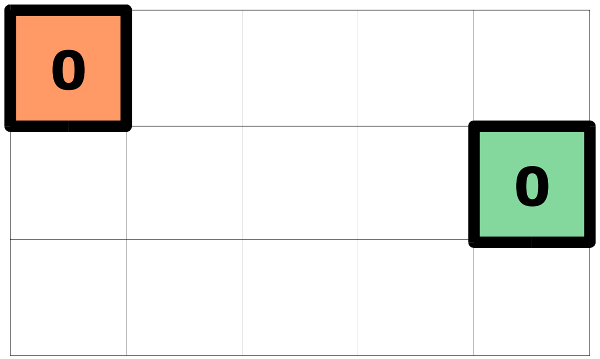


Algoritmo Básico

- Cada pixel fonte identificado por uma cor
  - Didática apenas



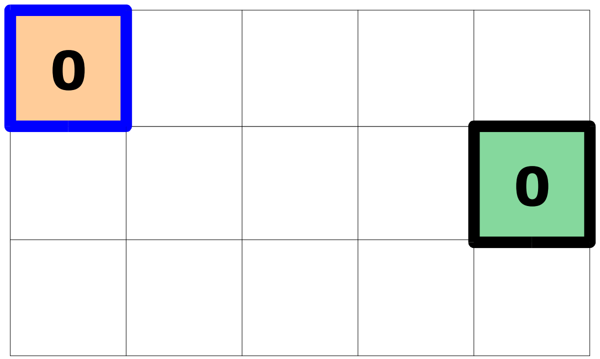
Algoritmo Básico



- Inserir pixels de interesse em uma fila com custo 0
- Demais pixels têm custo infinito

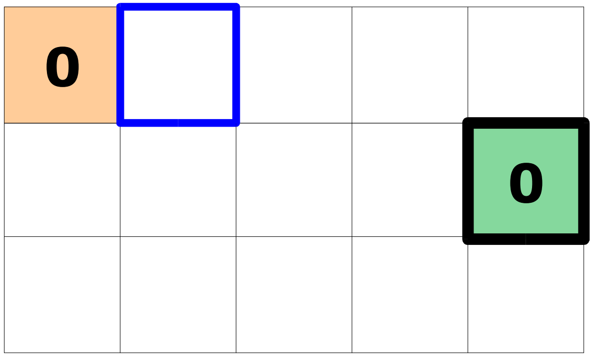
Algoritmo Básico

- Remova um pixel p da fila

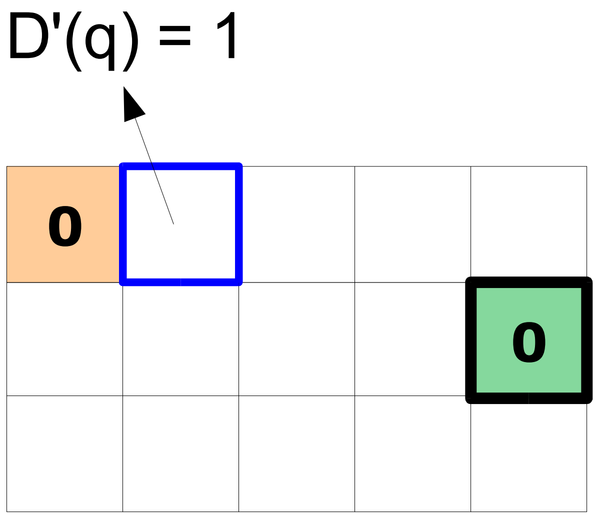


Algoritmo Básico

- Remova um pixel p da fila
- Para cada vizinho q

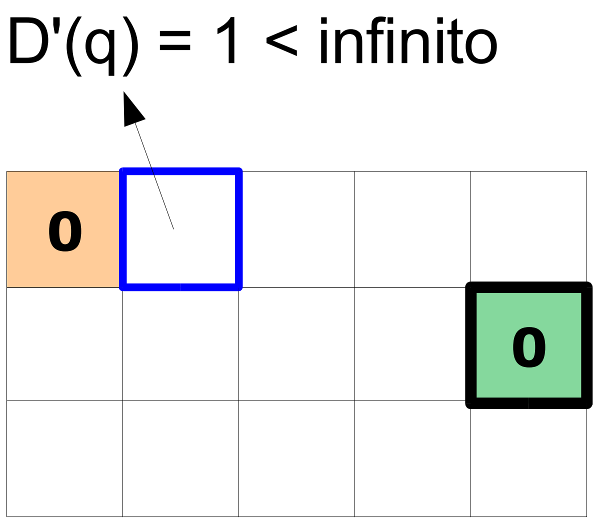


Algoritmo Básico



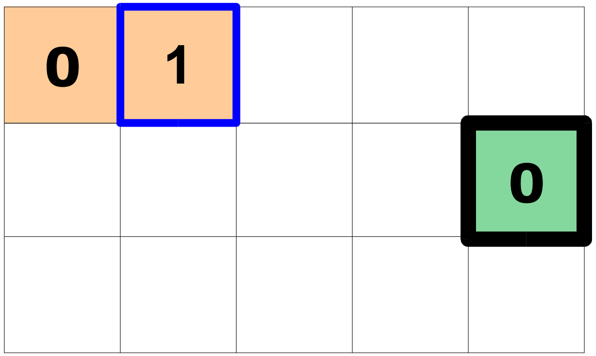
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)

Algoritmo Básico



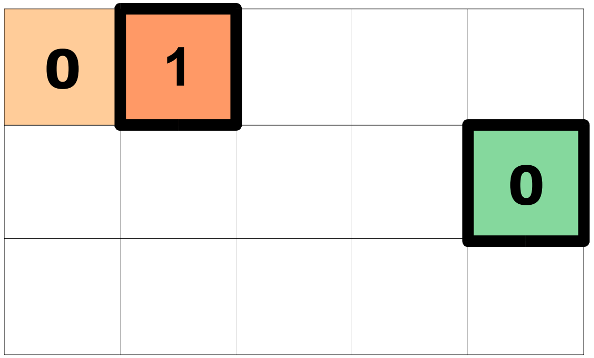
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



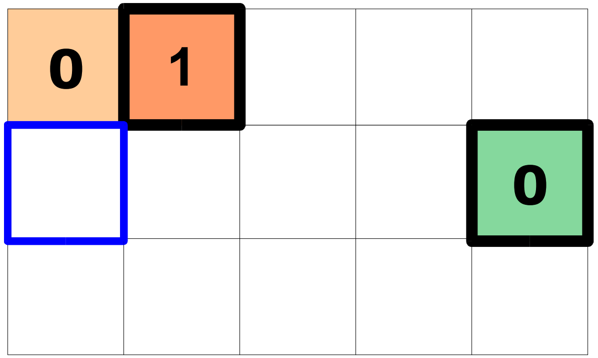
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



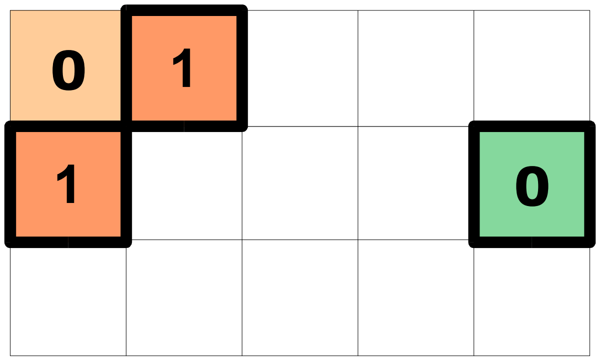
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



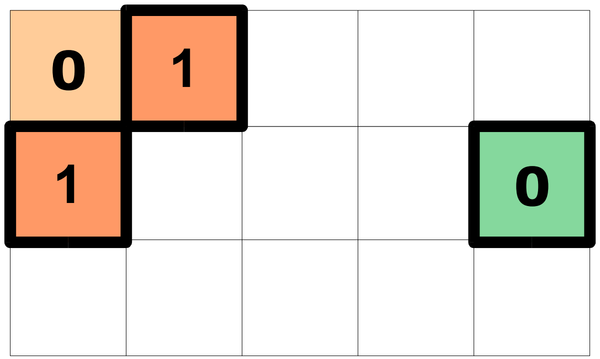
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



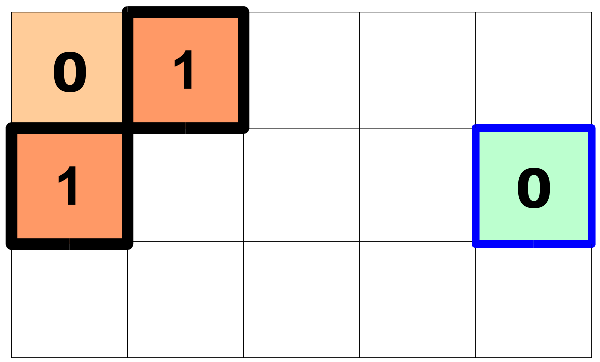
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



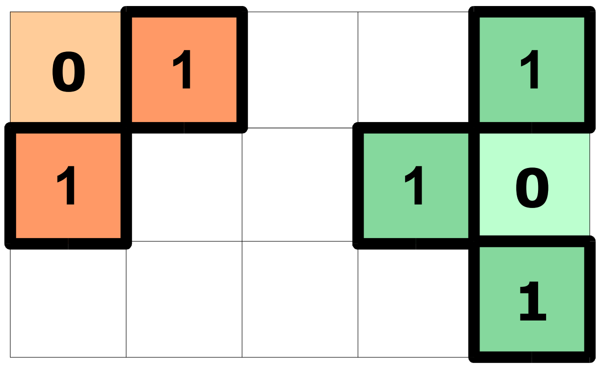
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



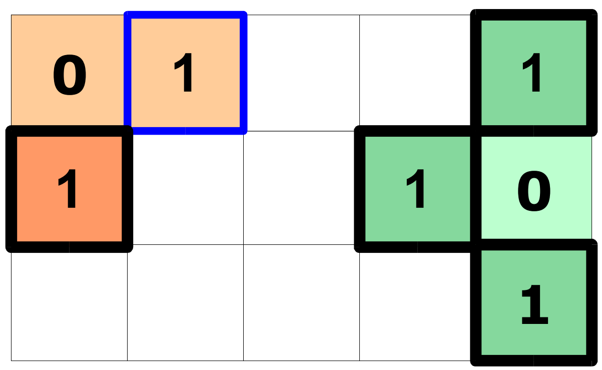
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



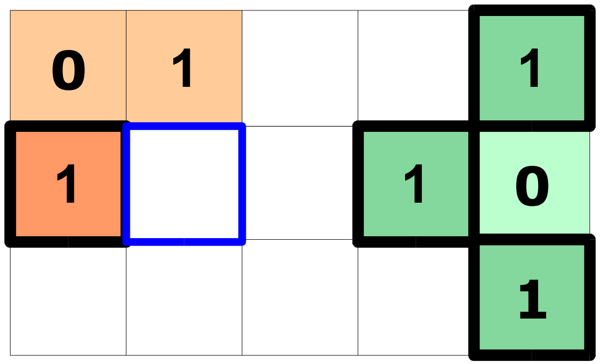
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



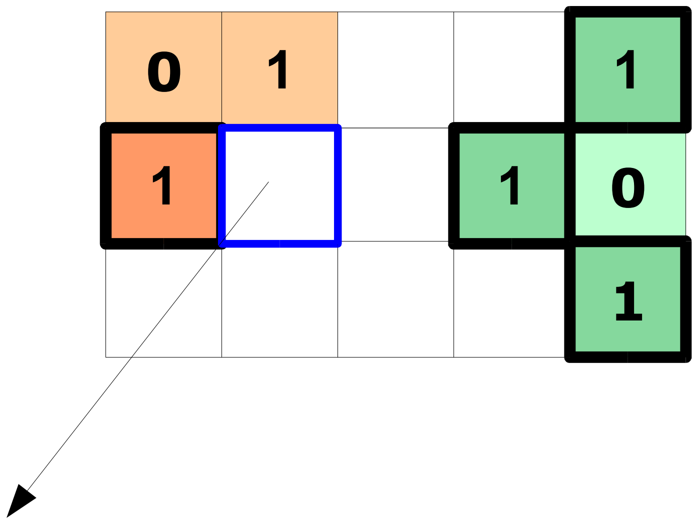
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

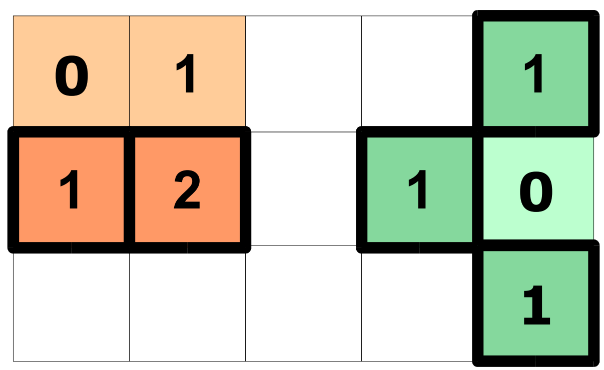
Algoritmo Básico



$D'(q) = 1^2 + 1^2 = 2$

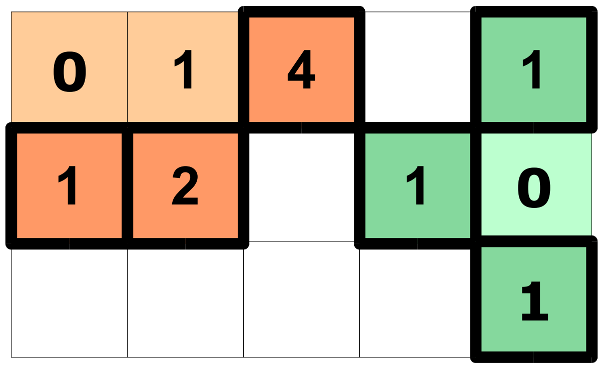
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



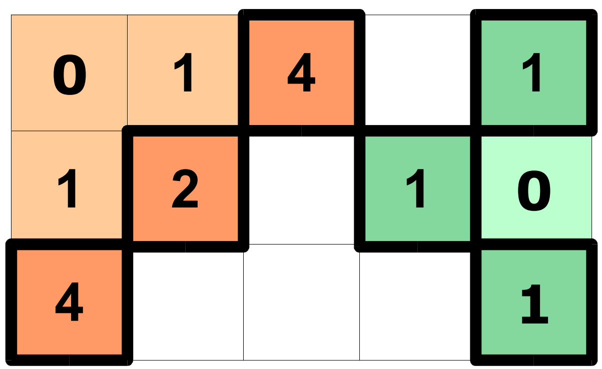
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



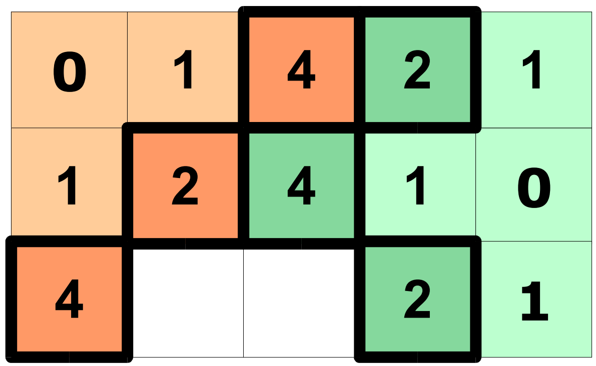
- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico



- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico

0	1	4	2	1
1	2	4	1	0
4	5	5	2	1

- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico

0	1	4	2	1
1	2	4	1	0
4	5	5	2	1

- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

Algoritmo Básico

0	1	4	2	1
1	2	4	1	0
4	5	5	2	1

- Remova um pixel p da fila
- Para cada vizinho q
  - Deduza D'(q) usando D(p)
  - Se D'(q) < D(q)
    - Atualize D(q)
    - Insira q na fila

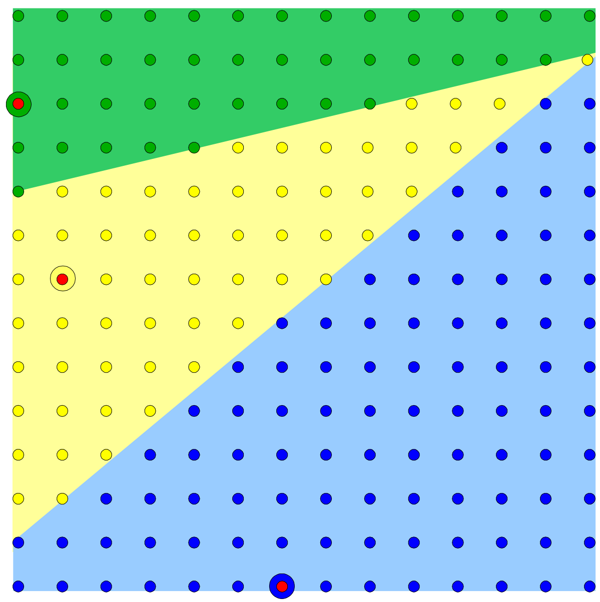
TDEs por Propagação

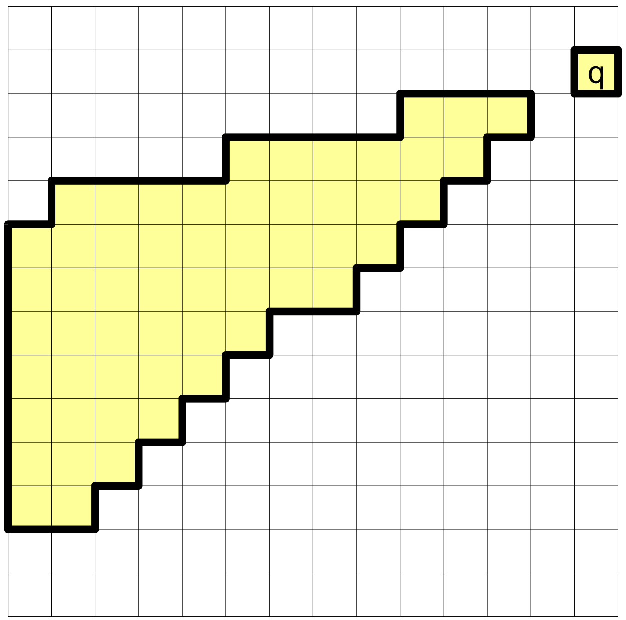
- Gargalo:
  - Escolher pixel de menor custo da lista
- Em imagens, os custos são:
  - Inteiros positivos
  - Limitados
- Solução:
  - Utilizar bucket sort
- Denota-se PSN:
  - Essa TDE com vizinhança fixa

TDEs por Propagação

- Problema:
  - Uso de visinhança fixa acarreta erros!
- Solução Eficiente:
  - Usar visinhanças maiores onde necessário
    - Método de Cuisenaire, 1999

TDEs por Propagação





Método de Cuisenaire

- Problemas:
  - Eficiência não demonstrada
    - Acredita-se: $O(n^2)$
  - Exatidão não demonstrada
- Foram realizados testes empíricos
  - Porém insuficientes
- Detalhes omissos ou errados no artigo

Outros Métodos de Propagação

- **Image Foresting Transform** (IFT)
  - 2002
- Abstração de vários problemas de imagens usando grafos
- Resolvidos eficientemente por Dijkstra
- TDE por IFT
  - Atualmente é o PSN
  - Logo, é inexato
- Vantagem: Teoria sólida e código eficiente e unificado

Outros Métodos de Propagação

- **Eggers 1998**
- Utiliza duas listas
- $O(n^3)$
  - Mas pode ser rápido em média
- Exatidão demonstrada
- Testes empíricos
  - Apenas uma idéia geral
  - Seriam necessários mais testes

TDE por

Varredura Independente

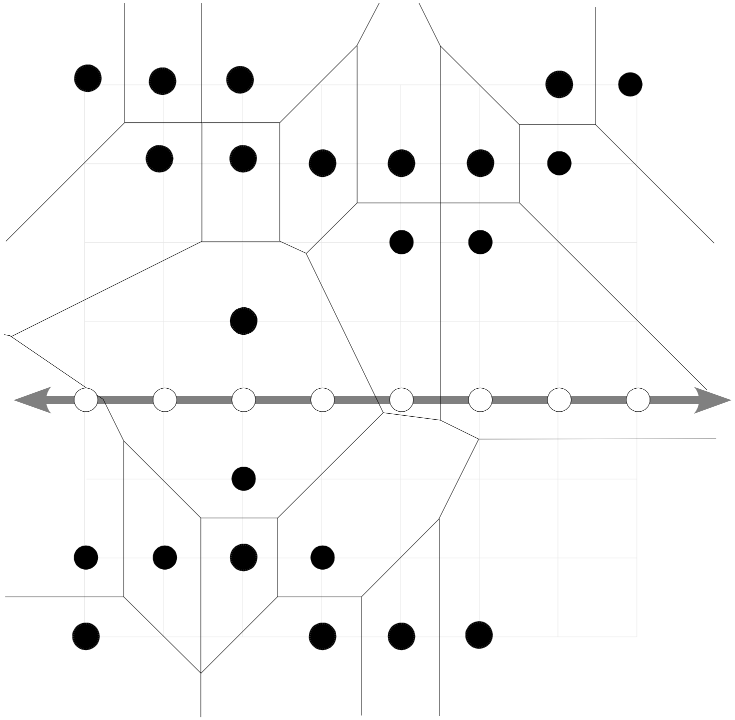
Método de Maurer

- Primeiro passo: TDE 1D

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0
0	0	1	4	4	4	1	0	0
0	0	4	9	9	9	4	0	0
0	0	1	4	4	4	1	0	0
0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0

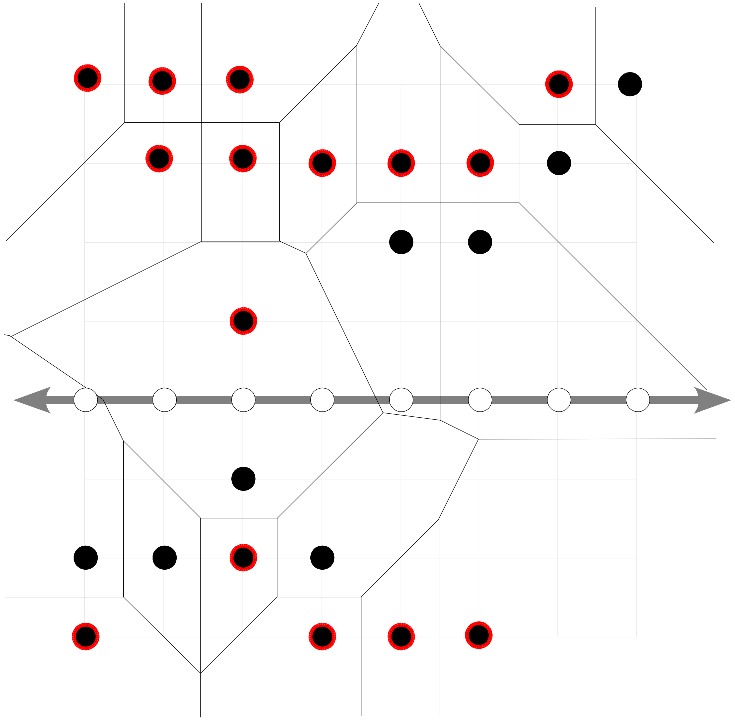
Método de Maurer

- Segredo: 2o. passo



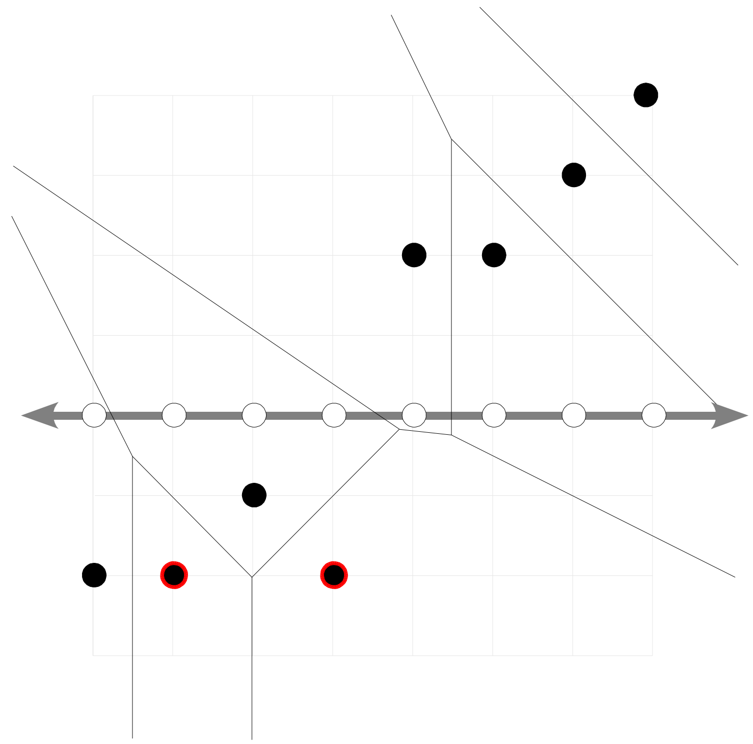
Método de Maurer

- Segredo: 2o. passo



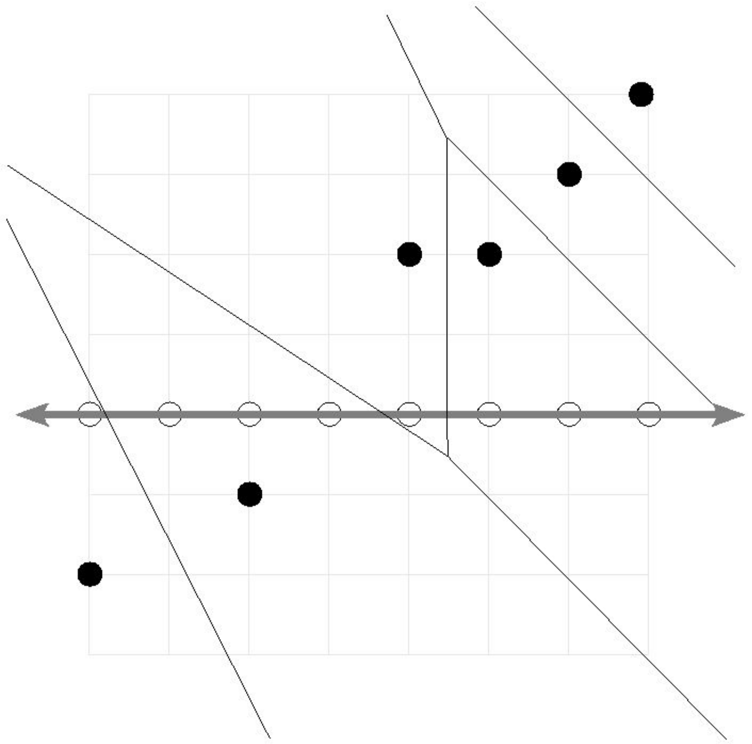
Método de Maurer

- Segredo: 2o. passo



Método de Maurer

- Segredo: 2o. passo



Método de Maurer

- Provado ser linear
- Análise Empírica
  - Relatada brevemente
  - Nada mostrado
- Extensível a n-D
- Paralelizável

Varredura Independente:

Outros algoritmos

- **Saito 1994**
- 2a etapa:
  - Restringe pixels a serem buscados usando TDE 1D da 1a etapa
  - Propriedades baseadas em interseção de parábolas restringem ainda mais a busca
- Fácil de implementar
- Extensível a n-D
- $O(n^3)$, mas parece ser rápido na média

Varredura Independente:

Outros algoritmos

- **Lotufo-Zampirolli, 2001**
- Morfologia Matemática
- Decomposição de elemento estruturante euclidiano em elementos 1D
- Erosões eficientes usando filas 1D
- Acredita-se:
  - $O(n^3)$
- Testes publicados são extremamente positivos

Avaliação Comparativa

Metodologia

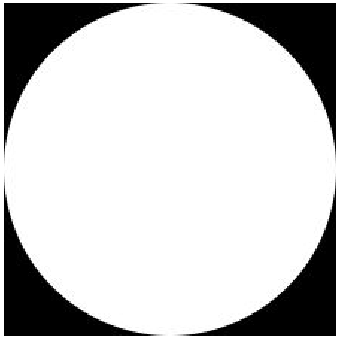
Algoritmos testados

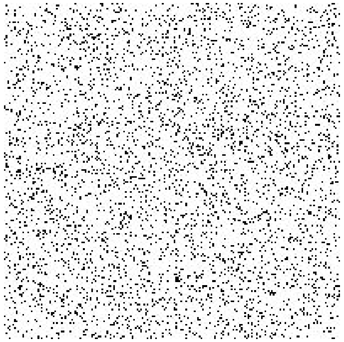
- Varredura Independente
  - Maurer - 2003
  - Saito - 1994
  - Lotufo-Zampirolli - 2001
- Propagação Ordenada
  - PMN de Cuisenaire - 1999
  - Eggers - 1998

Metodologia

Imagens Teste

- Um pixel no canto da imagem
  - Maior distância possível
- Círculo branco inscrito
- Imagem meia-preenchida
- Pixels pretos aleatórios
  - Desempenho relativo ao nro. de pontos de interesse

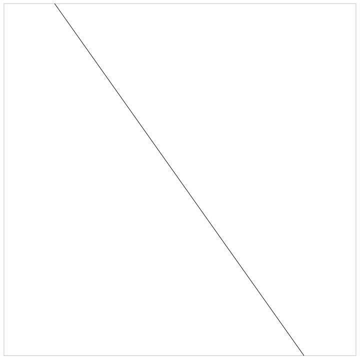




Metodologia

Imagens Teste

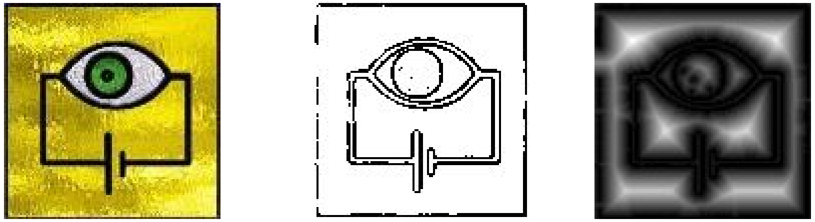
- Uma linha girando de 0° a 90°



Metodologia

Imagens Teste

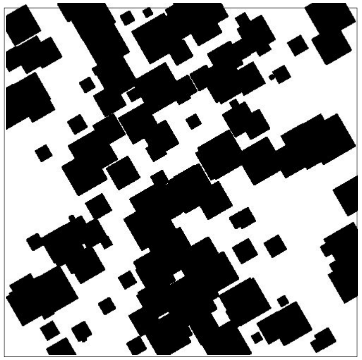
- Uma linha girando de 0° a 90°
- Imagens Binarizadas de Objetos Reais
  - Bordas binarizadas

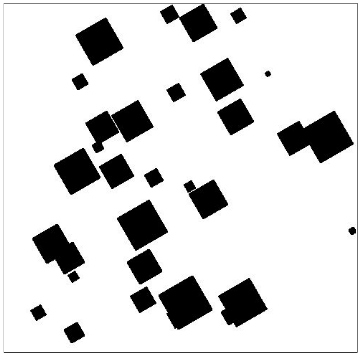


Metodologia

Imagens Teste

- Quadrados Aleatórios. Exemplo:
  - Porcentagem p de quadrados pretos
  - Rotacionados de um ângulo θ
  - Tamanhos dos quadrados sorteados em um intervalo

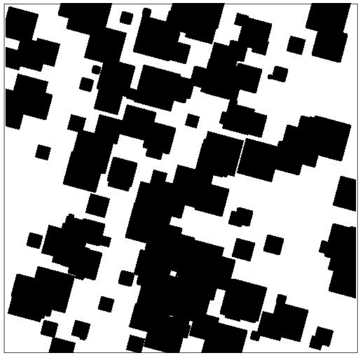
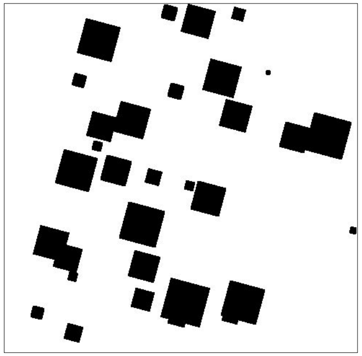




Metodologia

Imagens Teste

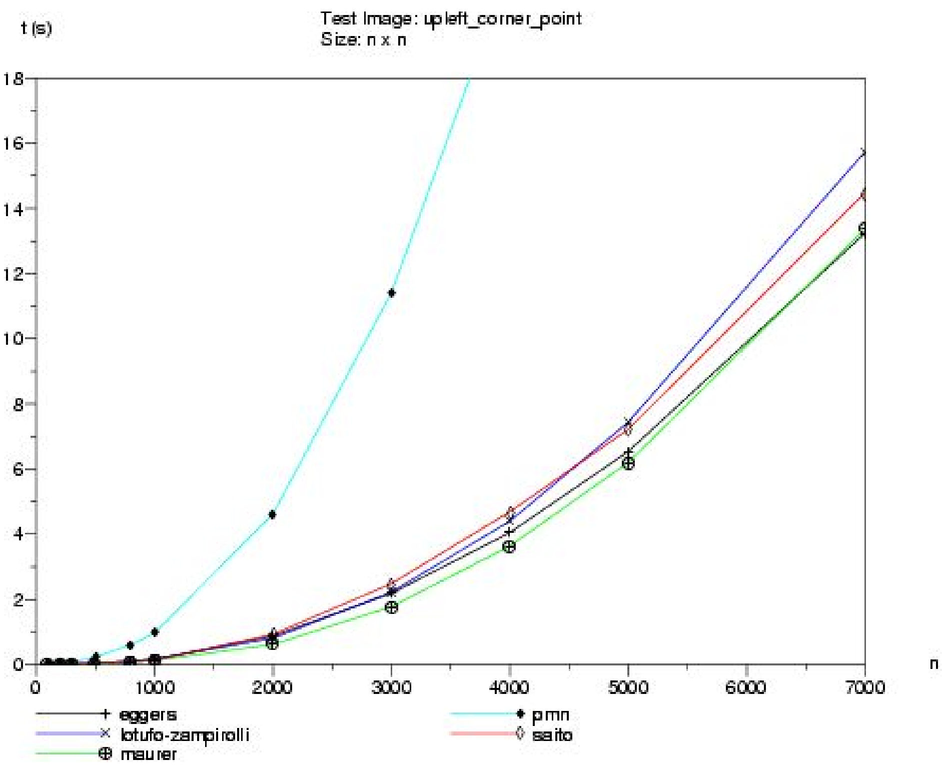
- Quadrados Aleatórios. Exemplo:
  - Porcentagem p de quadrados pretos
  - Rotacionados de um ângulo θ
  - Tamanhos dos quadrados sorteados em um intervalo

Resultados

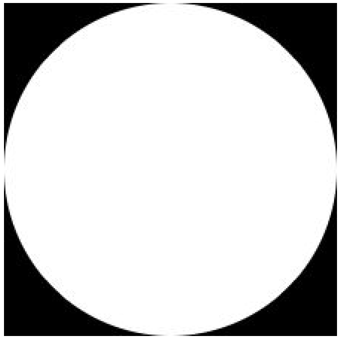
Pixel no canto

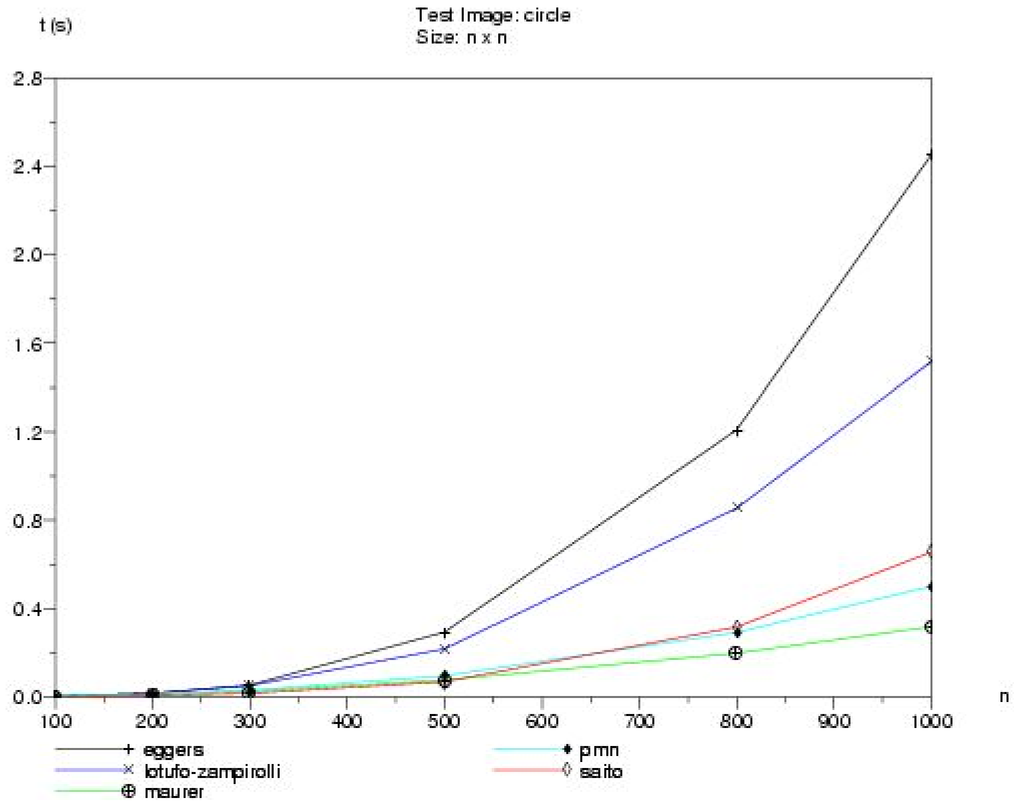
- Quadrados Aleatórios. Exemplo:
  - Surpresa: Cuisenaire



Resultados

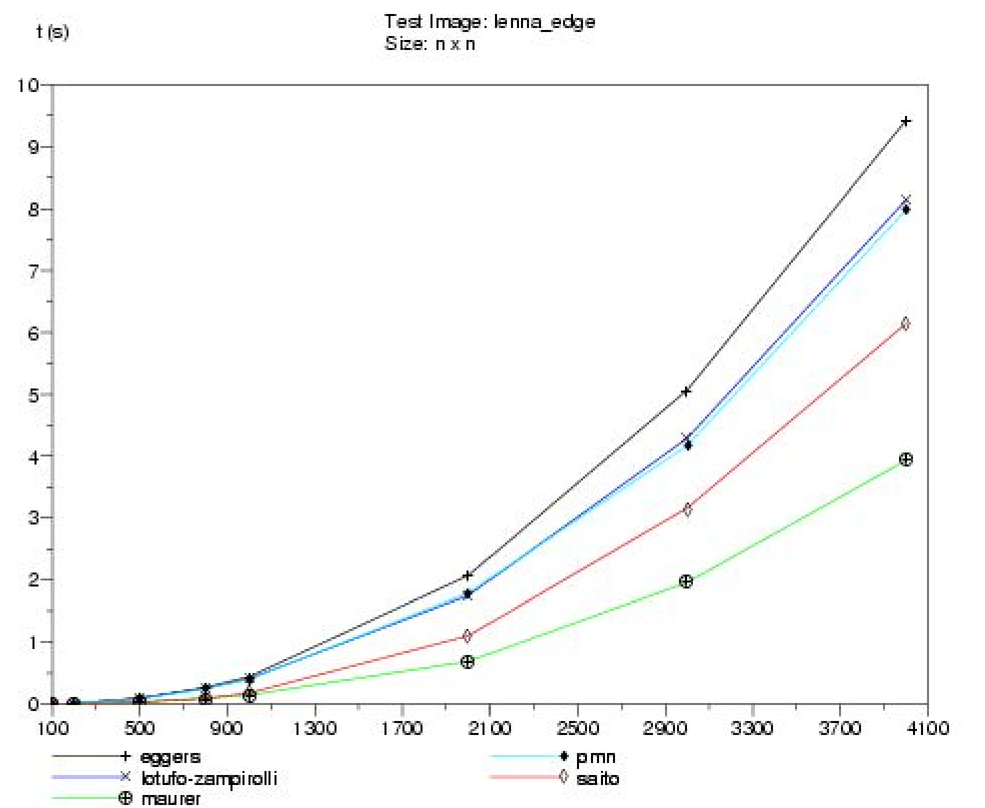
Círculo





Resultados

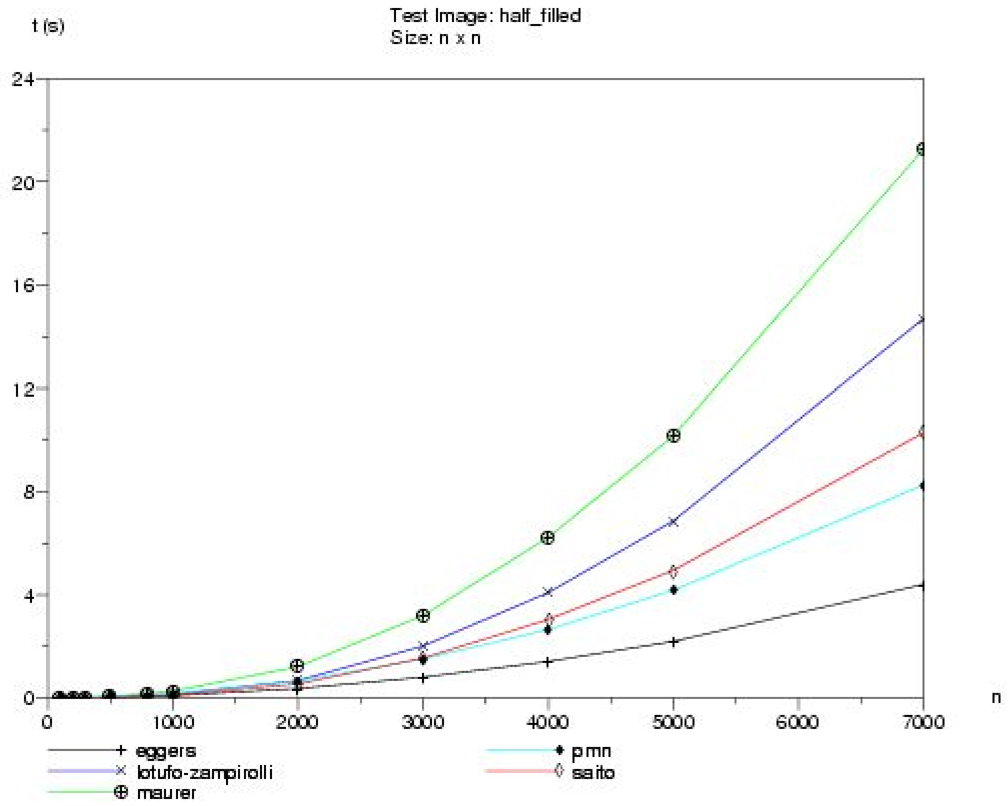
Bordas de Lenna



Resultados

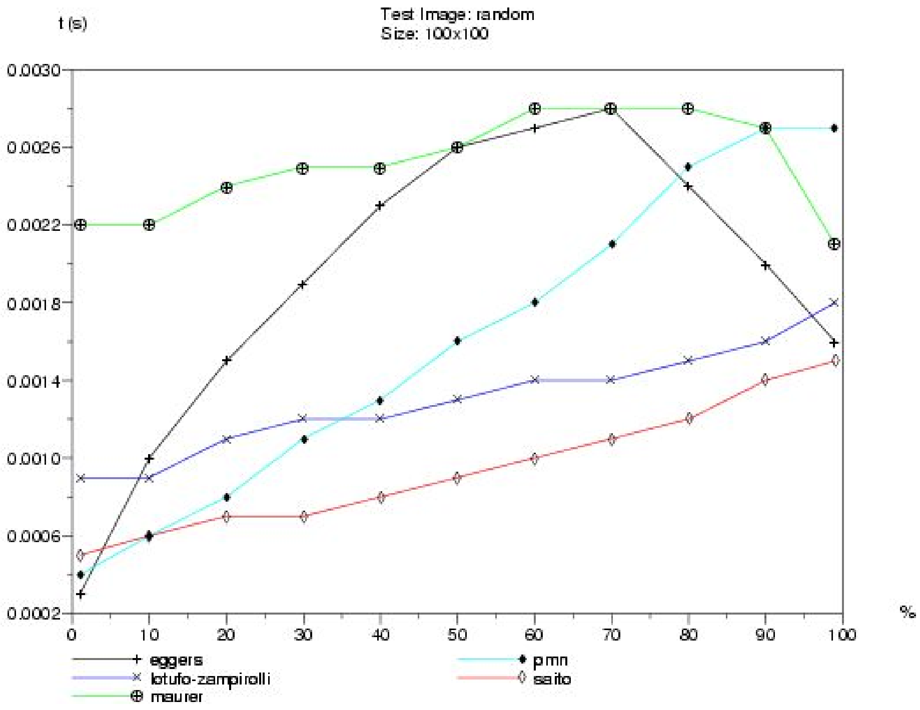
Imagem meia-preenchida

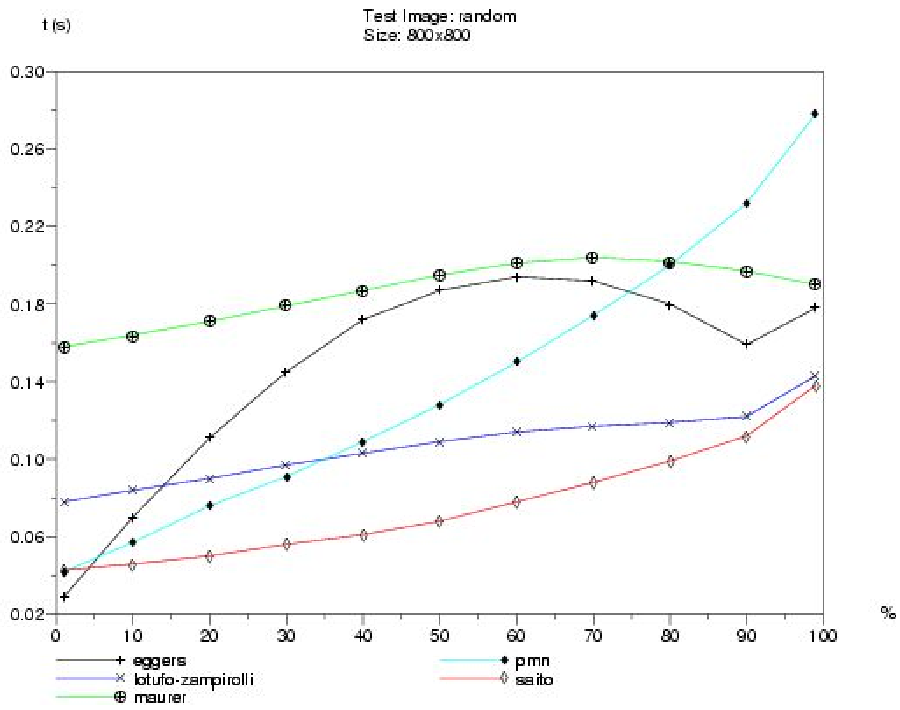
- Surpresa:
  - Maurer



Resultados

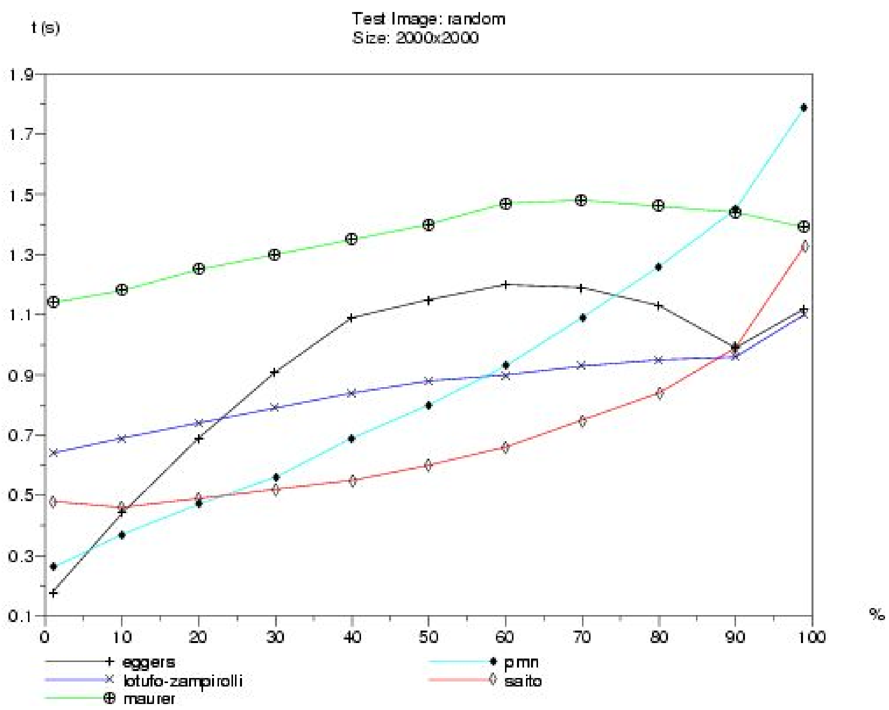
Pixels Aleatórios

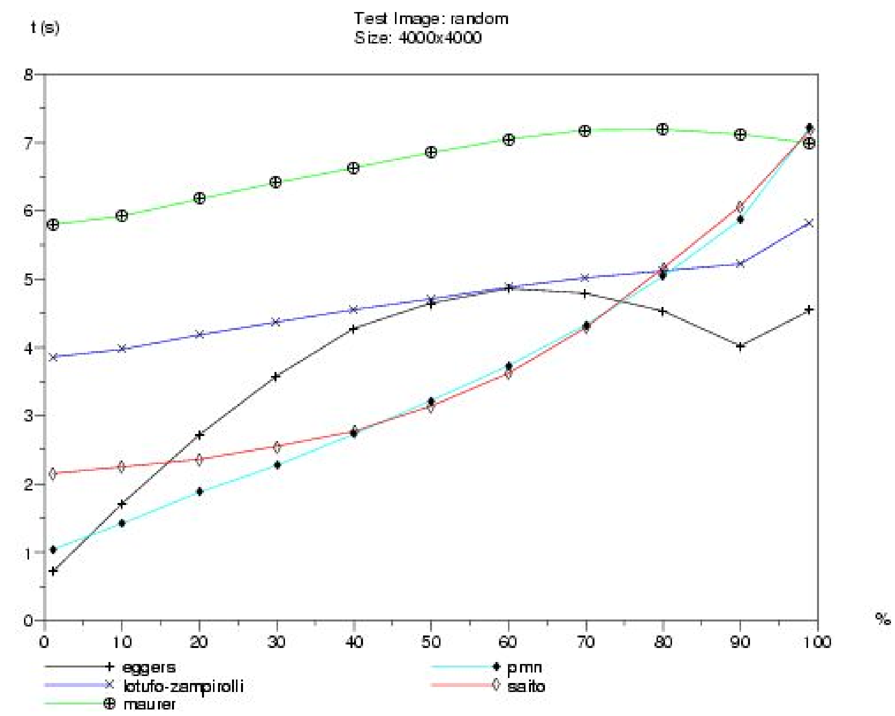




Resultados

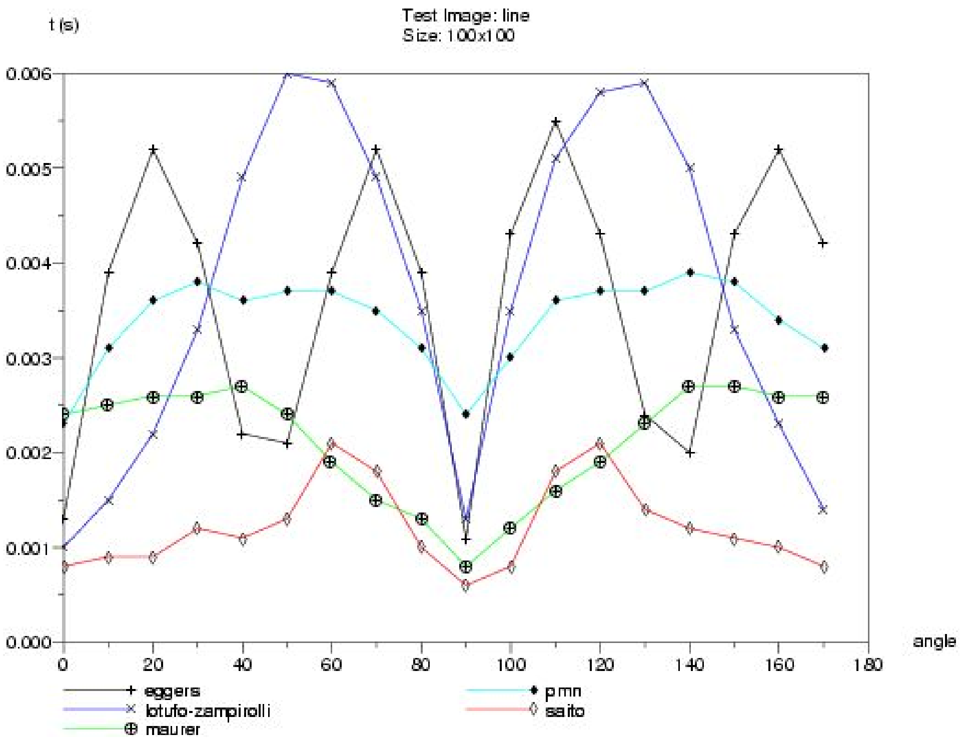
Pixels Aleatórios

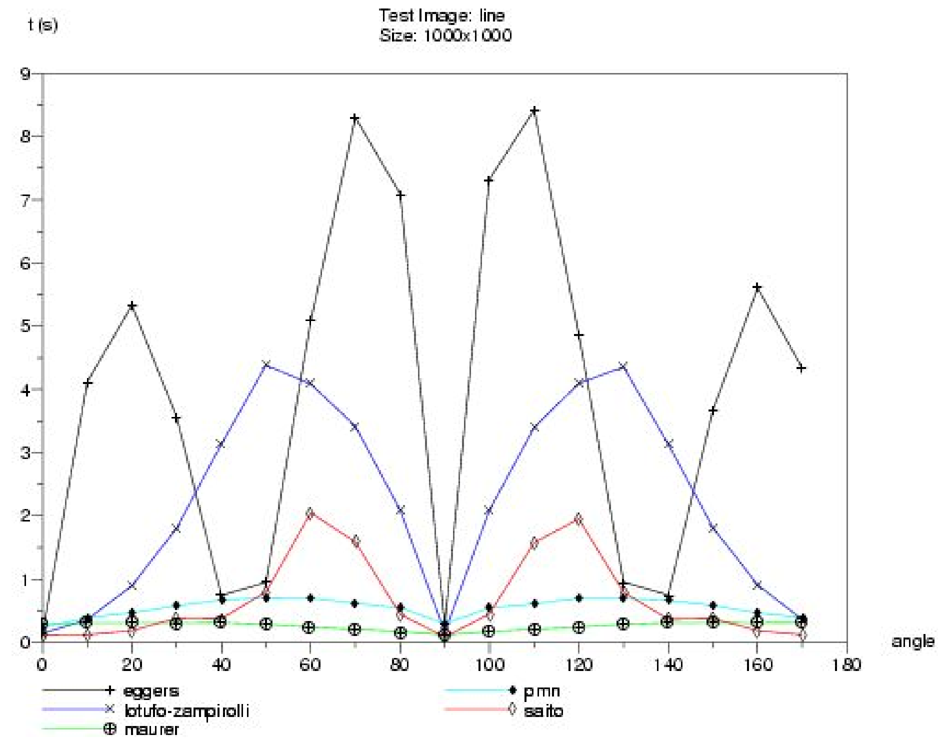




Resultados

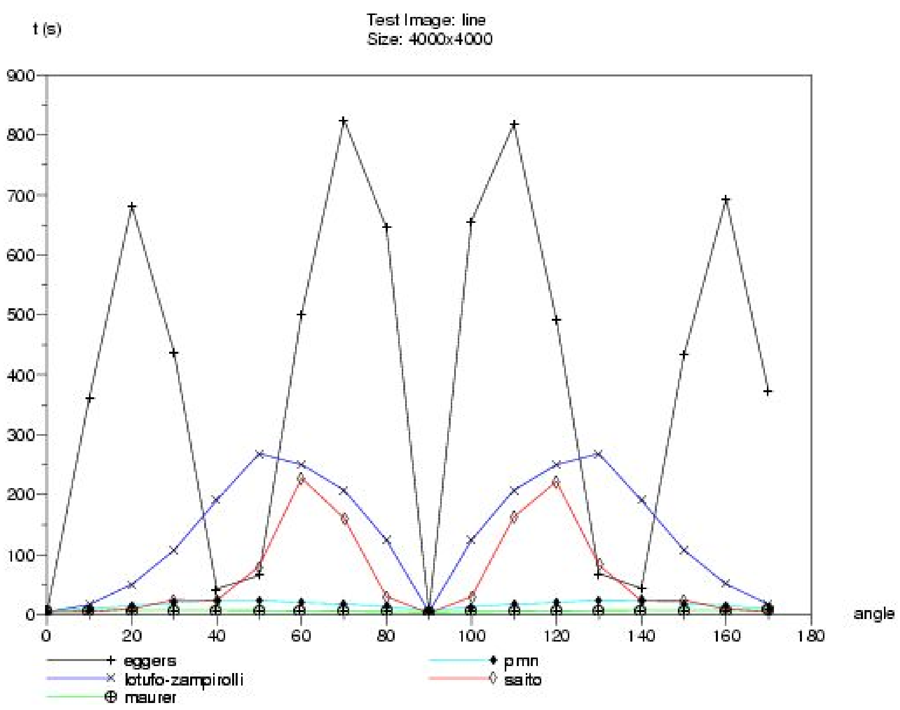
Linha giratória

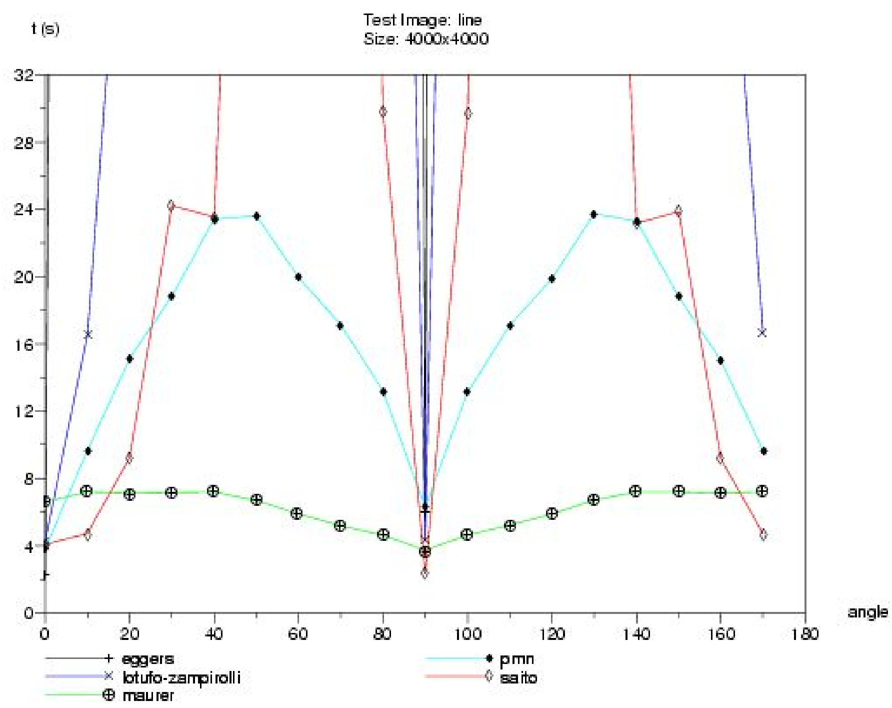




Resultados

Linha giratória

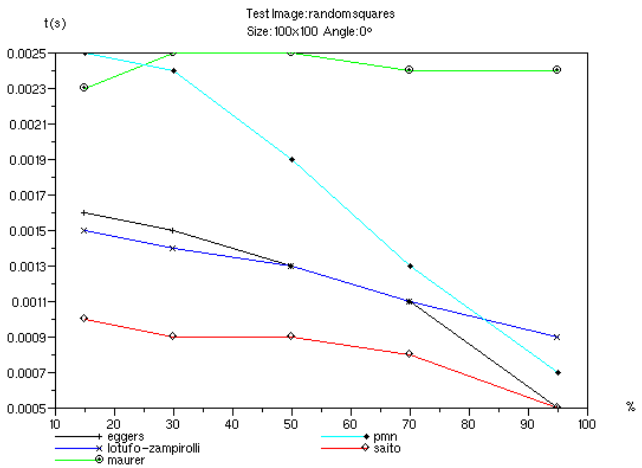


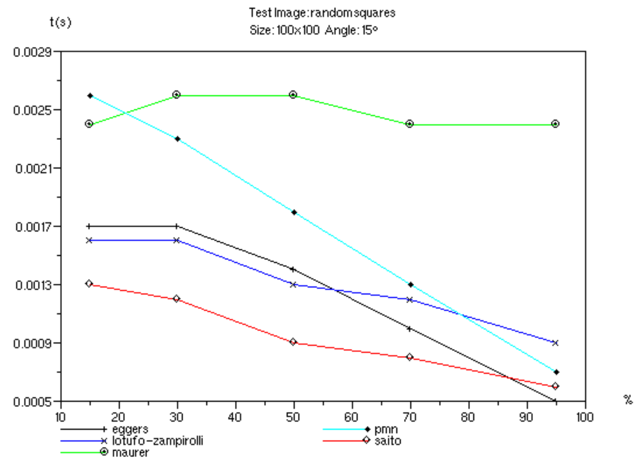


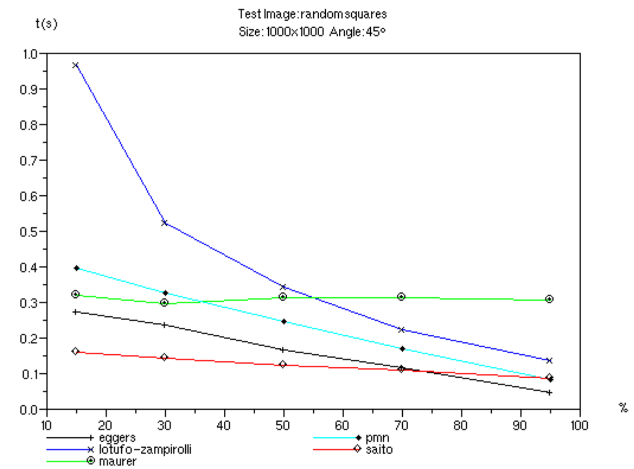
Resultados

Quadrados Aleatórios

- Ângulos fixos
- 100x100



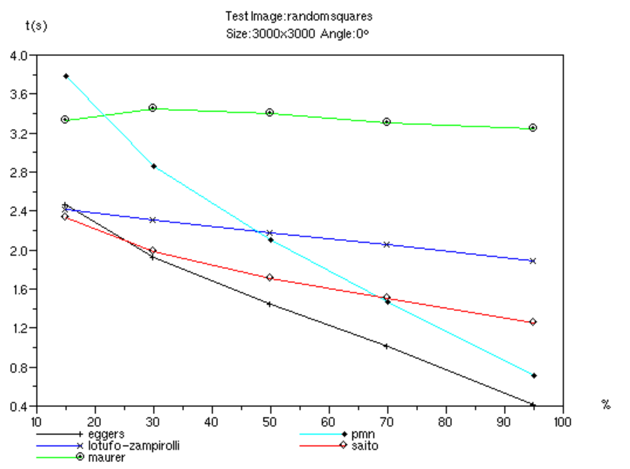


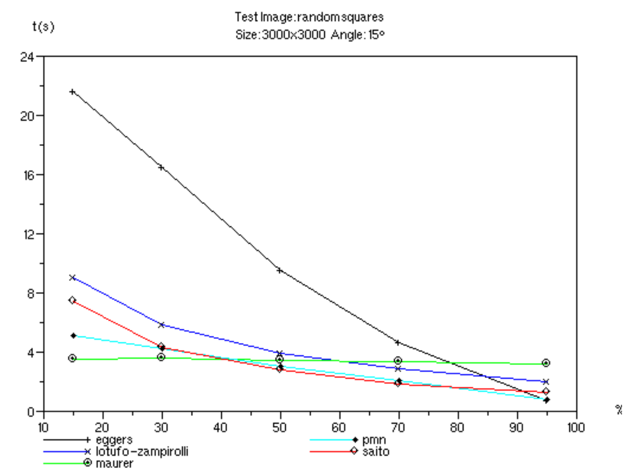


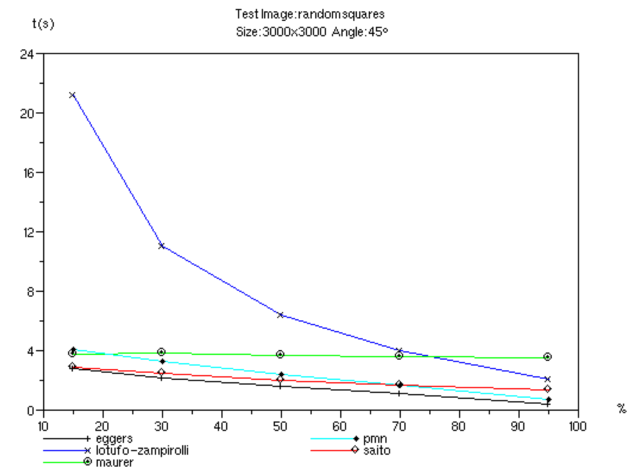
Resultados

Quadrados Aleatórios

- Ângulos fixos
- 3000x3000



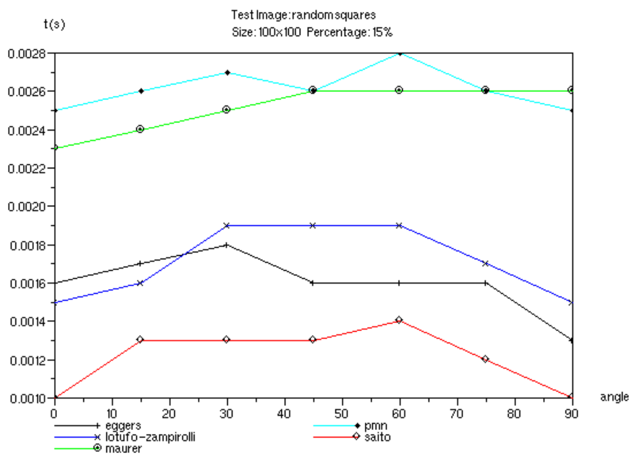


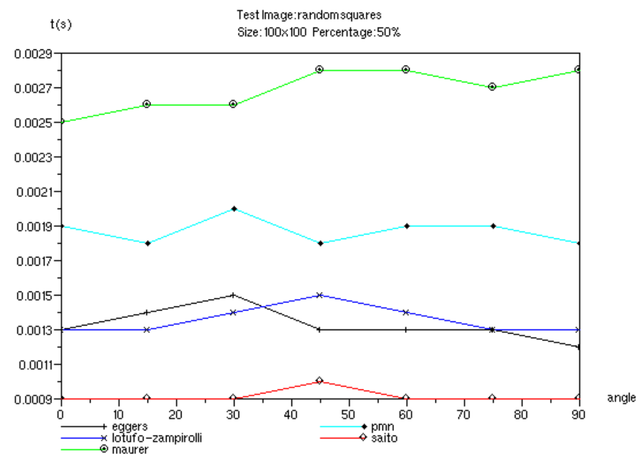


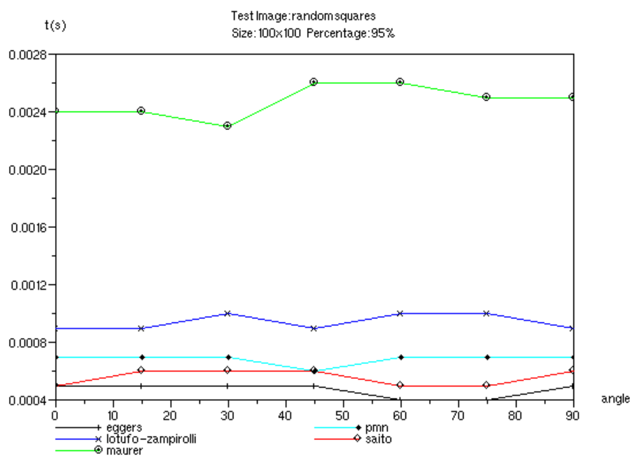
Resultados

Quadrados Aleatórios

- Porcentagem fixa
- 100x100



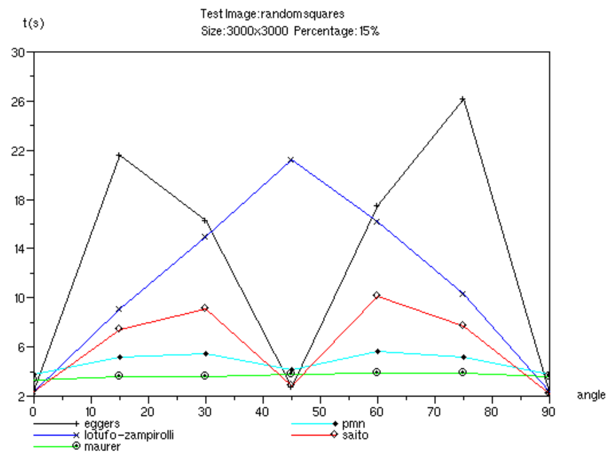


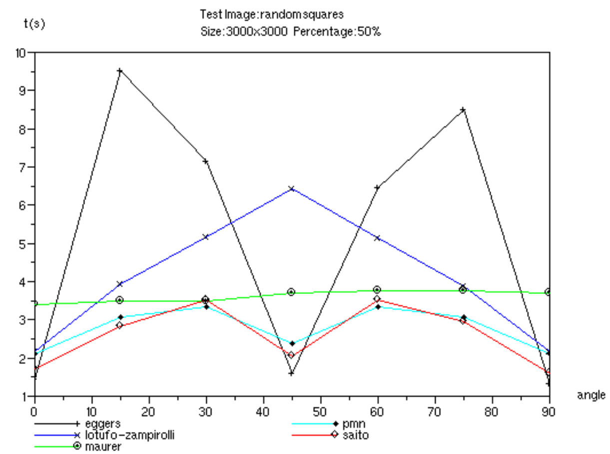


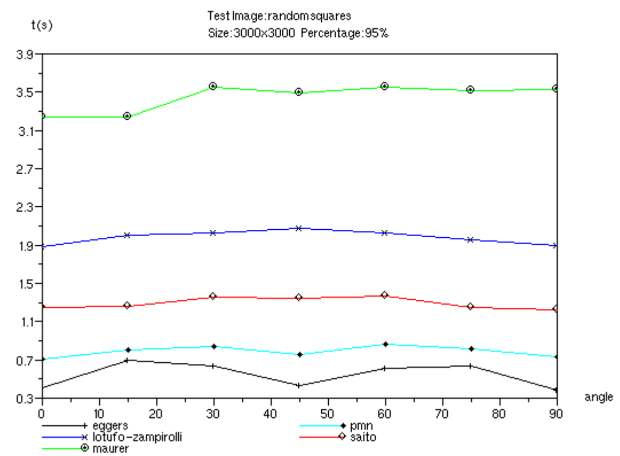
Resultados

Quadrados Aleatórios

- Porcentagem fixa
- 3000x3000







Exatidão

- Único inexato: Cuisenaire
  - Quadrados aleatórios
  - Círculo 300x300
  - Lenna a partir de 500x500
  - Reta giratória (exceto ângulo reto)
- Ainda não determinamos se o erro está na teoria ou na implementação

Observações

- Comportamento bastante variado
- Velocidade proporcional à quantidade de pixels de interesse
- Também depende muito da geometria
- Exemplo:
  - Imagens com 50% de pixels
    - Comportamentos bem diferentes!

Desempenho de

Cuisenaire

- Pior caso para imagens com muitos pixels brancos
- O método mais dependente do número de pixels de interesse
- Relativamente estável à inclinação
- Linear relativo ao tamanho da imagem para imagem da reta e quadrados

Desempenho de
Maurer

- O mais estável relativo ao conteúdo
- Mais rápido para a maioria dos casos
- Relativamente lento para imagens com muitos pixels de interesse
  - Mas nunca ficou mais que 6 vezes mais lento que Saito

Desempenho de

Saito

- Bem na média
- Nunca foi o mais lento
- O mais fácil de implementar
- Pior caso: reta inclinada a 60º
  - 40 vezes mais lento que Maurer
- Bastante dependente ao conteúdo

Desempenho de

Lotufo-Zampirolli

- Desempenho mediano
- Não foi o melhor para nenhuma imagem testada
- Estável ao número de pixels de interesse
- Depende muito da orientação

Desempenho de

Eggers

- O mais dependente do conteúdo
- Foi o melhor para:
  - Imagem meia-preenchida
  - Imagem de quadrados 50% a 0º
  - Um ponto no canto
- Foi o pior para:
  - Bordas de Lenna
  - Círculo inscrito
  - Algumas porcentagens e inclinações das outras imagens

Conclusões e Perspectivas

Principais conclusões

- Maurer e Saito parecem ser os melhores algoritmos
  - Saito mais fácil que Maurer
  - Maurer se mostrou linear e estável, Saito não
- A implementação de Cuisenaire utilizada não é linear nem exata
- Eggers e Lotufo-Zampirolli, no geral, mostraram desempenho relativo inferior e uma dependência grande ao conteúdo
- O desempenho dos algoritmos avançados de TDE dependem muito do conteúdo da imagem

Contribuições

- Levantamento bibliográfico atualizado e organizado
- Descrição inédita dos métodos
  - Uniformidade
  - Ênfase nos conceitos-chave
  - Elucida passagens obscuras dos originais
- Validação dos algoritmos de TDE
- Implementação confiável e acessível
  - Será disponibilizada em software livre
  - Extensivamente testada
  - Interface com Scilab

Contribuições

- Algoritmos mais confiáveis comprovadamente eficientes
- Potencial para futuros avanços teóricos
  - Experimentos forneceram evidência para propriedades a serem provadas futuramente
  - *Insight* para novos algoritmos
- Metodologia aplicável a outros problemas correlatos

Contribuições

- Validações futuras facilitadas
- Utilidade do trabalho é ampla
  - A TDE é base de diversos outros operadores, técnicas e aplicações

Trabalho Futuro

- Extensão dos métodos para outros problemas
  - Eixos mediais multi-escala
  - Diagramas de Voronoi
- Tratar outros domínios
  - 3D
  - Domínios não-convexos
- Extensão para outras métricas
  - Segmentação
- Incluir métodos baseados em EDPs
- Estudo teórico aprofundado

Trabalho Futuro

- Incorporar mais imagens teste
- Analisar formalmente o método de Shih2004

Referências

A. Rosenfeld and J. Pfaltz, "Sequential operations in digital picture processing," *Journal of the ACM*, vol. 13, no. 4, 1966.

G. Borgefors, "Distance transformations in arbitrary dimensions," *Computer Vision, Graphics, and Image Processing*, vol. 27, pp. 321–345, 1984.

G. Borgefors, "Distance transformations in digital images," *Computer Vision, Graphics, and Image Processing*, vol. 34, pp. 344–371, 1986.

P.-E. Danielsson, "Euclidean distance mapping," *Computer Graphics and Image Processing*, vol. 14, pp. 227–248, 1980.

D. W. Paglieroni, "Distance transforms: Properties and machine vision applications," *Graphical Models and Image Processing*, vol. 54, no. 1, pp. 56–74, 1992.

T. Saito and J. Toriwaki, "New algorithms for Euclidean distance transformations of an n-dimensional digitised picture with applications," *Pattern Recognition*, vol. 27, no. 11, pp. 1551–1565, 1994.

H. Eggers, "Two fast Euclidean distance transformations in $\mathbb{Z}^2$ based on sufficient propagation," *Computer Vision and Image Understanding*, vol. 69, pp. 106–116, jan 1998.

Referências

O. Cuisenaire, *Distance Transformations: Fast Algorithms and Applications to Medical Image Processing*. PhD thesis, Université Catholique de Louvain, Belgique, oct 1999.

O. Cuisenaire and B. Macq, "Fast Euclidean distance transformation by propagation using multiple neighborhoods," *Computer Vision and Image Understanding*, vol. 76, no. 2, pp. 163–172, 1999.

O. Cuisenaire and B. Macq, "Fast and exact signed Euclidean distance transformation with linear complexity," in *ICASSP99 - IEEE Intl Conference on Acoustics, Speech and Signal Processing*, vol. 6, (Phoenix, USA), pp. 3293–3296, mar 1999.

C. Maurer, R. Qi, and V. Raghavan, "A linear time algorithm for computing the Euclidean distance transform in arbitrary dimensions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 265–270, feb 2003.

Referências

F. Y. Shih and Y.-T. Wu, "Fast Euclidean distance transformation in two scans using a 3×3 neighborhood," *Computer Vision and Image Understanding*, vol. 93, pp. 195 – 205, feb 2004.

R. Lotufo and F. Zampirolli, "Fast multi-dimensional parallel Euclidean distance transform based on mathematical morphology," in *Proccedings of SIBGRAPI, XIV Brazilian Symposium on Computer Graphics and Image Processing*, pp. 100–105, IEEE Computer Society, 2001.

R. A. Lotufo and F. A. Zampirolli, "Multidimensional parallel EDT using 1d erosions by propagation," march 2003. submetido.

F. A. Zampirolli, *Transformada de Distância por Morfologia Matemática.* PhD thesis, UNICAMP, Campinas, Brasil, jun 2003.

A. Falcao, J. Stolfi, and R. A. Lotufo, "The image foresting transform: theory, algorithms, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 19–29, jan 2004.

Referências

Implementações

- SIP- Scilab Image Processing Toolbox
  - http://siptoolbox.sourceforge.net
- Animal – An Imaging Library
  - http://animal.sourceforge.net

Futuramente:

- Referências categorizadas:
  - www.hotreference.com
- Resultados completos:
  - http://cyvision.if.sc.usp.br/~rfabbri/edt