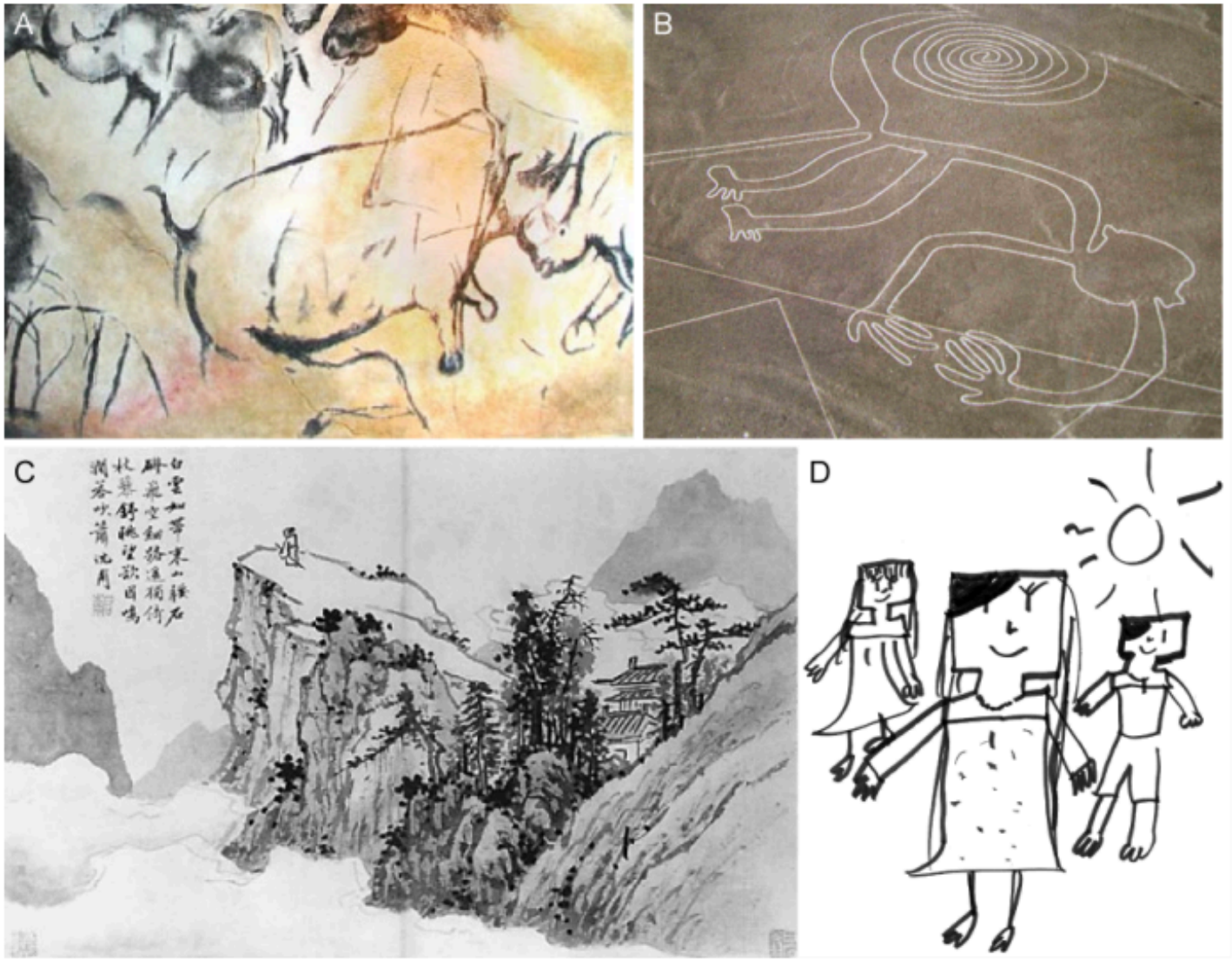
VISÃO COMPUTACIONAL

Encontrando Bordas

FILIPE ROLIM CORDEIRO (FILIPE.ROLIM@GMAIL.COM)

Contornos são importantes para nós



(A) Cave painting at Chauvet, France, about 30,000 B.C.;
(B) Aerial photograph of the picture of a monkey as part of the Nazca Lines geoglyphs, Peru, about 700 – 200 B.C.;
(C) Shen Zhou (1427-1509 A.D.): Poet on a mountain top, ink on paper, China;
(D) Line drawing by 7-year old L. Heras (2010 A.D.)

Contornos são importantes para os humanos

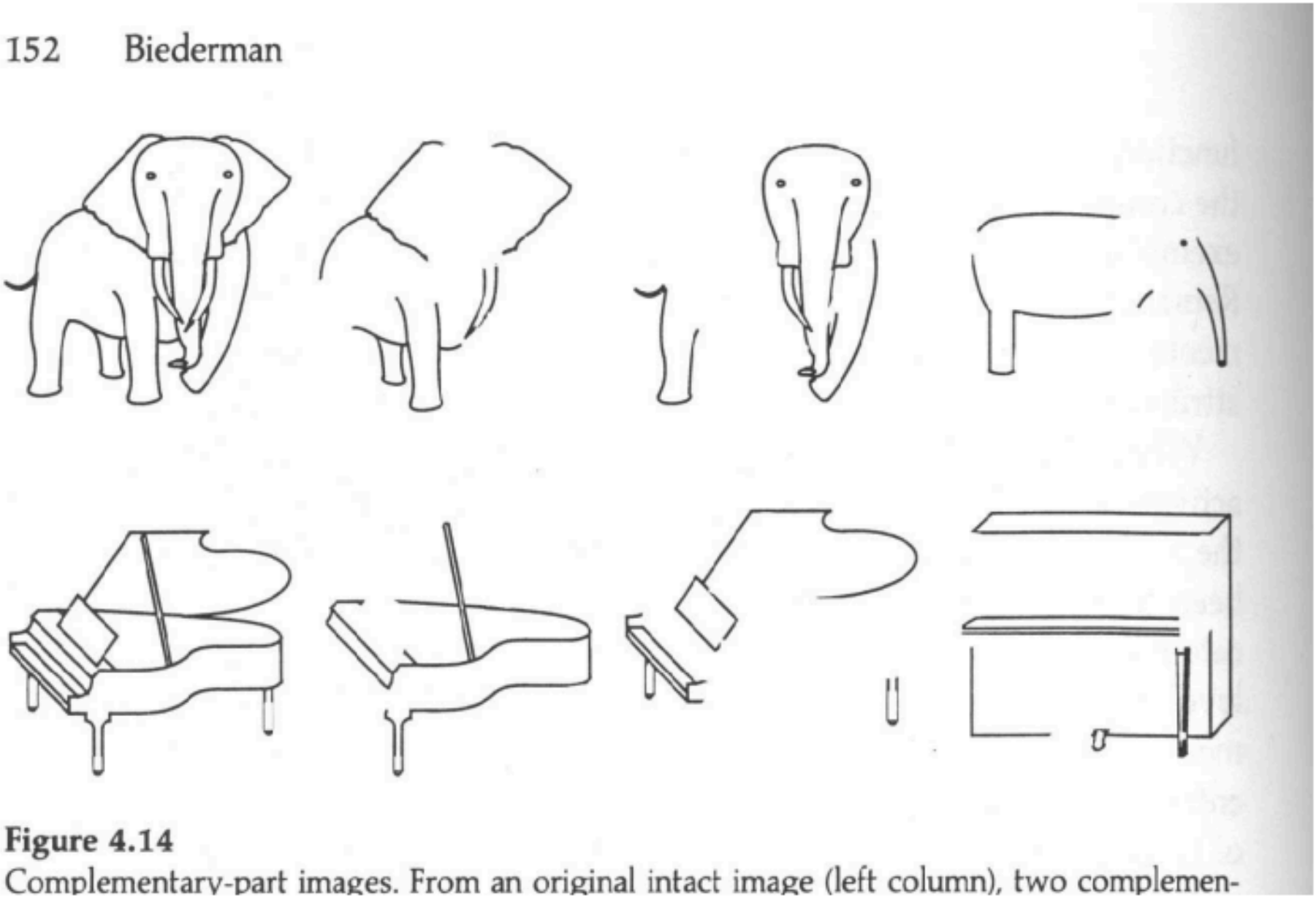
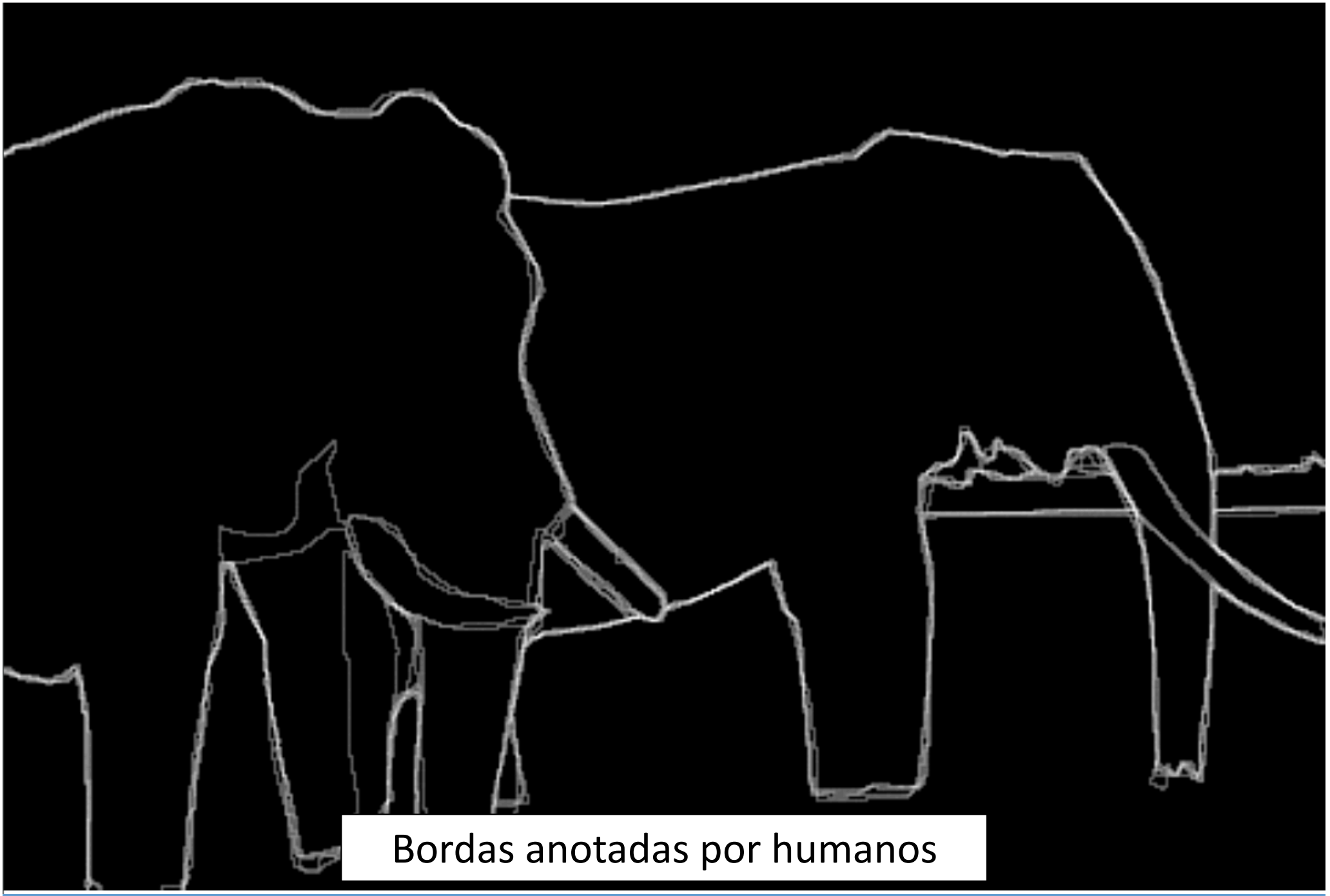
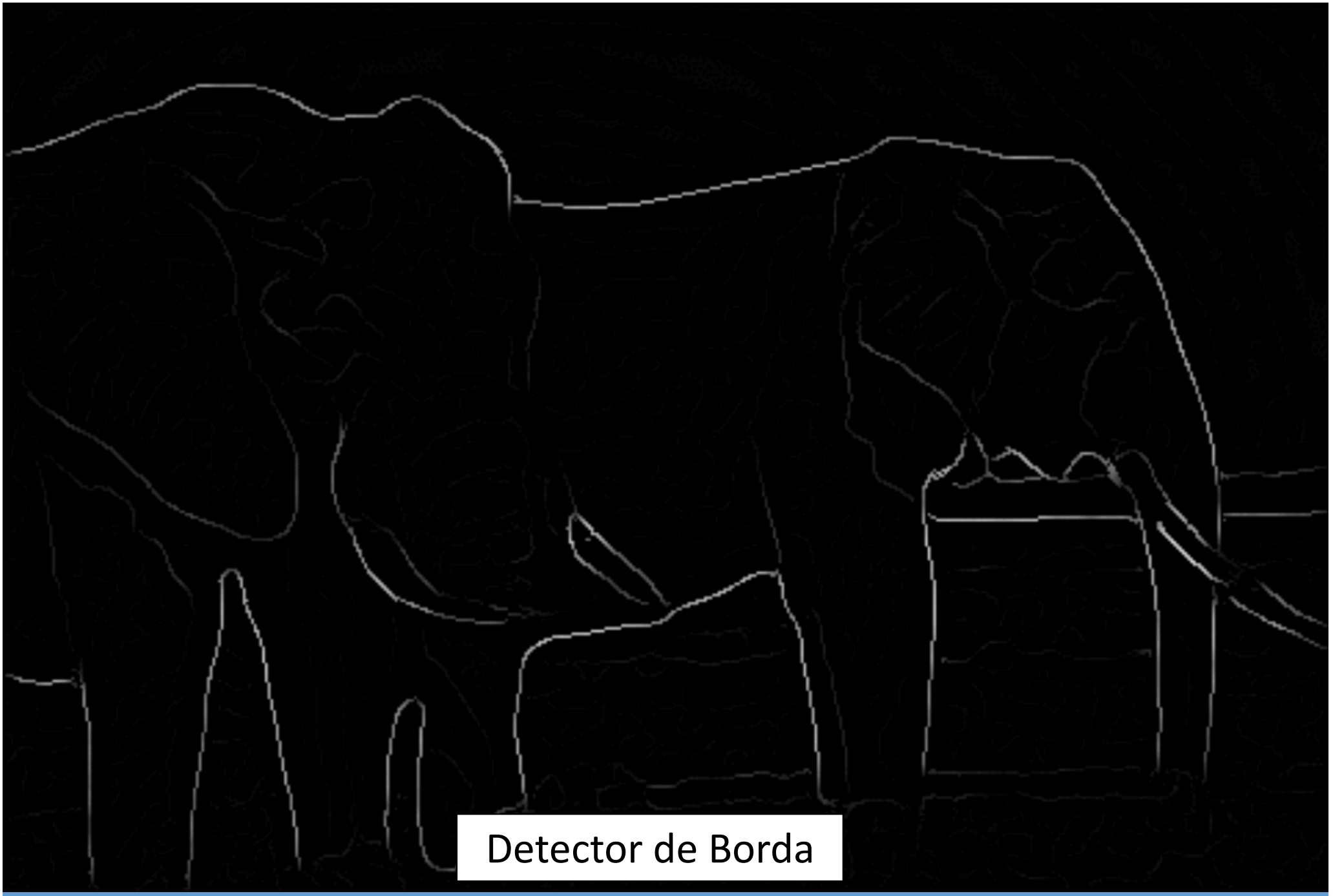


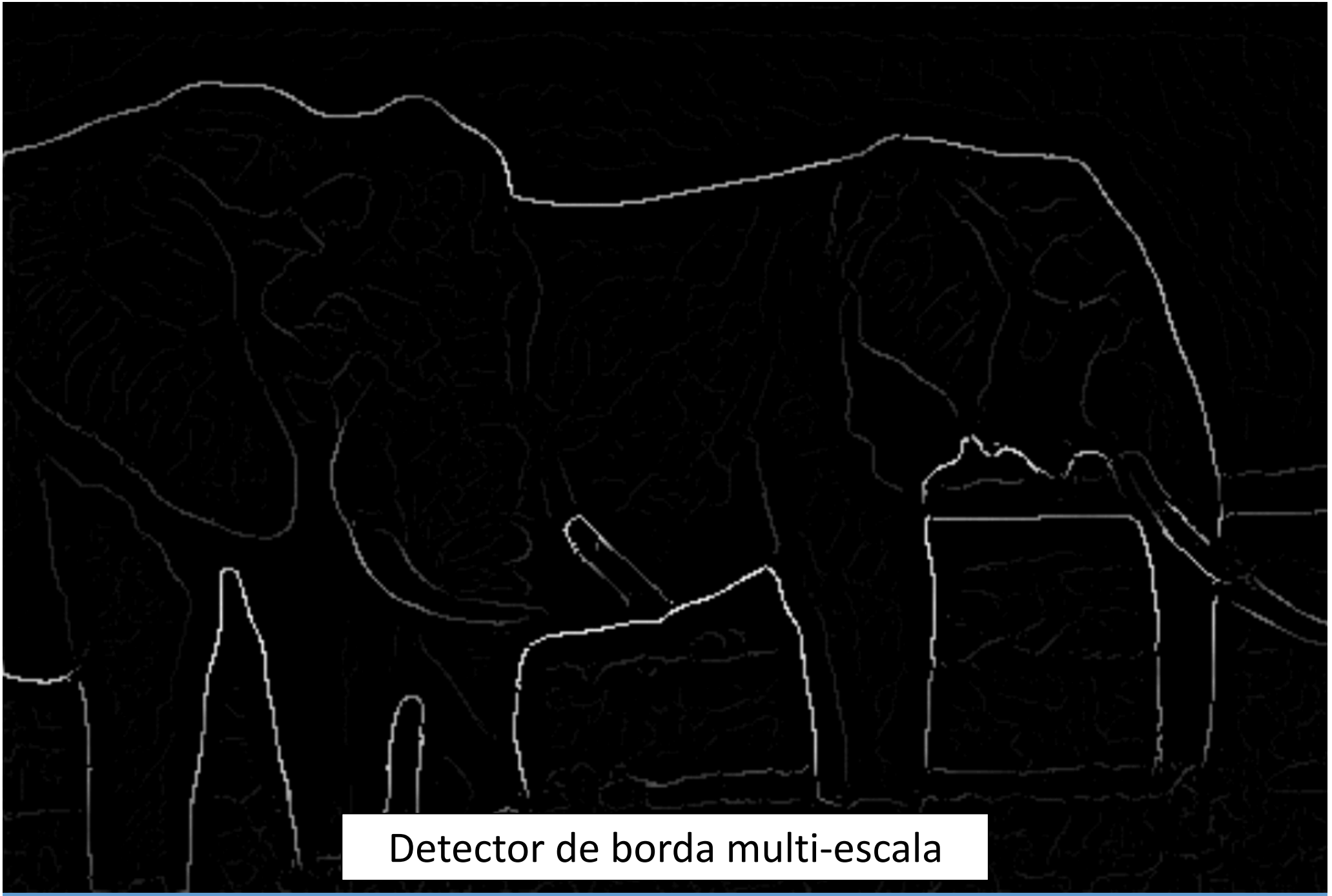
Figure 4.14
Complementary-part images. From an original intact image (left column), two complemen-

Definindo Bordas

Onde Estão as bordas do objeto?

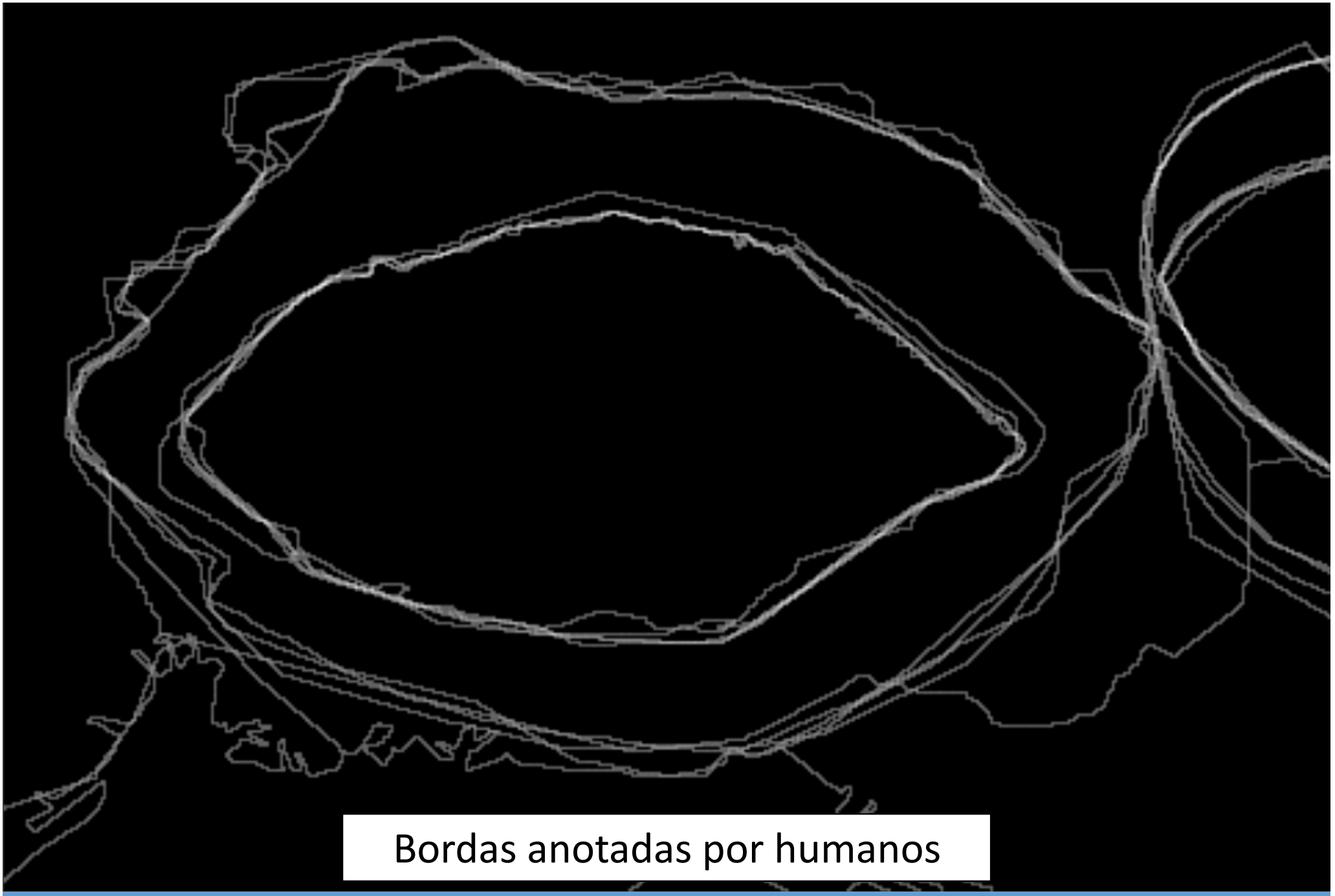

Bordas anotadas por humanos


Detector de Borda

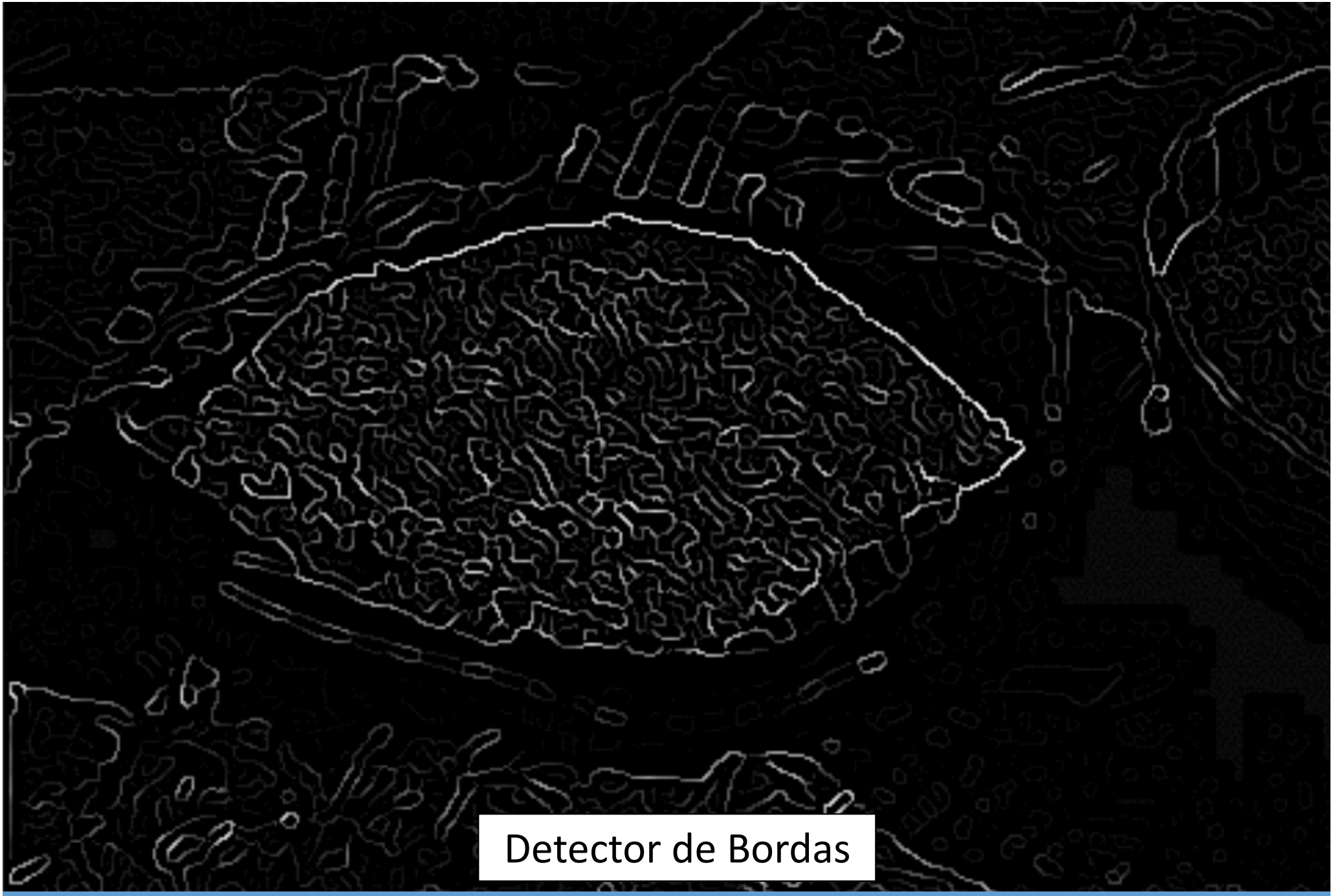

Detector de borda multi-escala


A força das bordas não necessariamente
correspondem à nossa percepção de bordas


Onde estão as bordas do objeto?



Bordas anotadas por humanos


Detector de Bordas

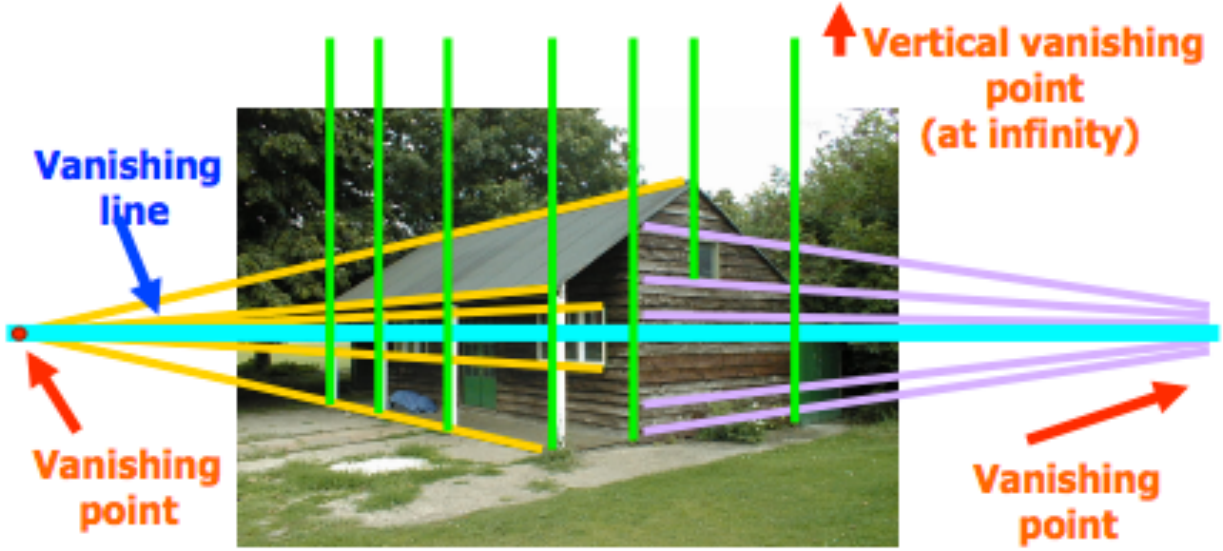

Definir bordas é difícil para nós

Detecção de bordas

- Objetivo
  - Identificar mudanças (discontinuidades) na imagem
    - Iformação semântica pode ser encontrada nas bordas
    - Mais compacto que pixels
- Ideal
  - Contorno definido por artista
    - Mas artista também usa conhecimento do objeto

Por que se importar com bordas?

- Extrair informação, reconhecer objetos
- Recuperar geometria e ponto de vista

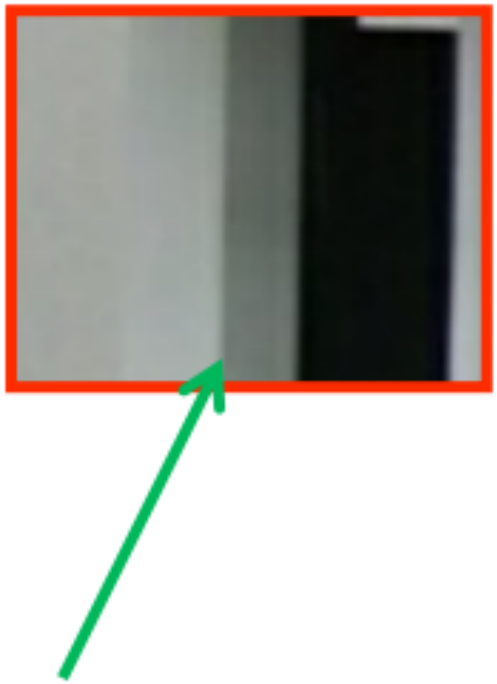


Origem das Bordas

- Descontinuidade de superfície
- Descontinuidade de profundidade
- Descontinuidade de cor de superfície
- Descontinuidade de iluminação

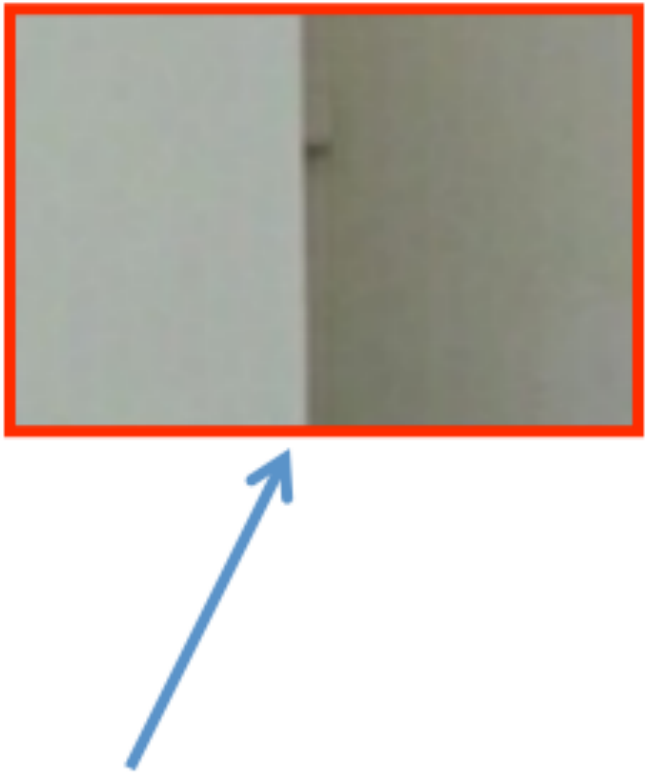
Origem das Bordas

- Descontinuidade de superfície



Origem das Bordas

- Descontinuidade de profundidade



Origem das Bordas

- Descontinuidade de cor

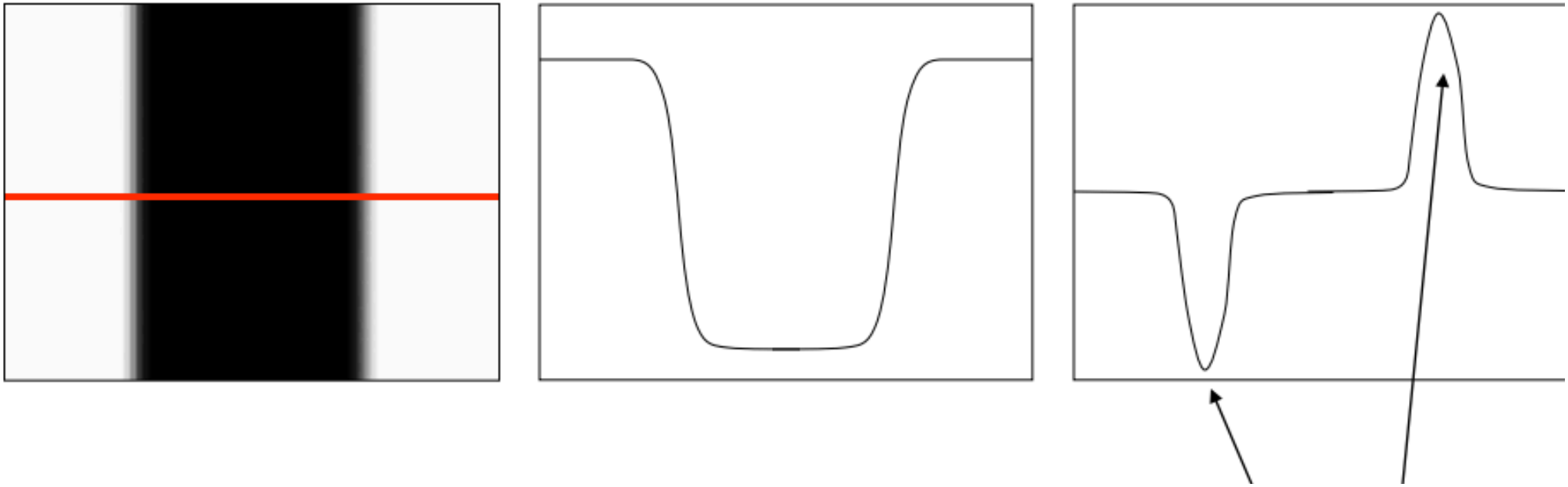
O que caracteriza uma borda

- Uma borda é uma rápida mudança na função de intensidade da imagem

imagem

Função de intensidade (ao longo da linha horizontal)

Primeira derivada



Bordas correspondem aos extremos da derivada


Original
Hello!
Edge Map
Hello!

Detecção de Bordas

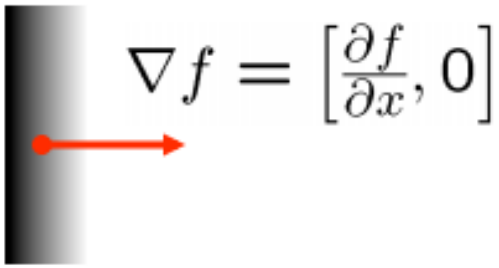
- Derivada de Primeira Ordem/ Métodos de Gradiente
  - Operadores de Roberts
  - Operadores de Sobel
  - Operadores de Prewitt
- Derivadas de Segunda Ordem
  - Laplaciano
  - Laplaciano de Gaussiana (LoG)
  - Diferença de Gaussianas (DoG)
- Detecção de Bordas Ótimo
  - Canny Edge Detection

Gradiente da Imagem

- Gradiente da imagem

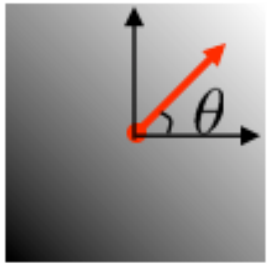
$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

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



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

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



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

Gradiente da Imagem

- A direção do gradiente é dada por

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

- A intensidade do gradiente é dada pela magnitude

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

Gradiente Discreto

- Como podemos calcular a derivada de uma imagem
  - Opção 1: reconstruir uma imagem contínua e então utilizar o gradiente
  - Opção 2: utilizar a derivada discreta (diferença finita)

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

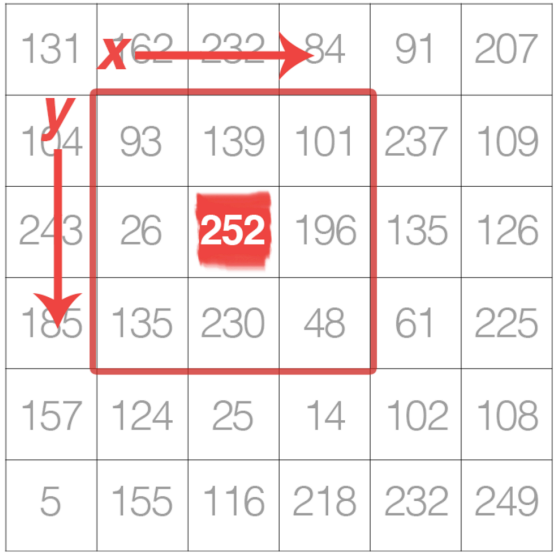
Gradiente Discreto

- Mudança vertical (eixo y)

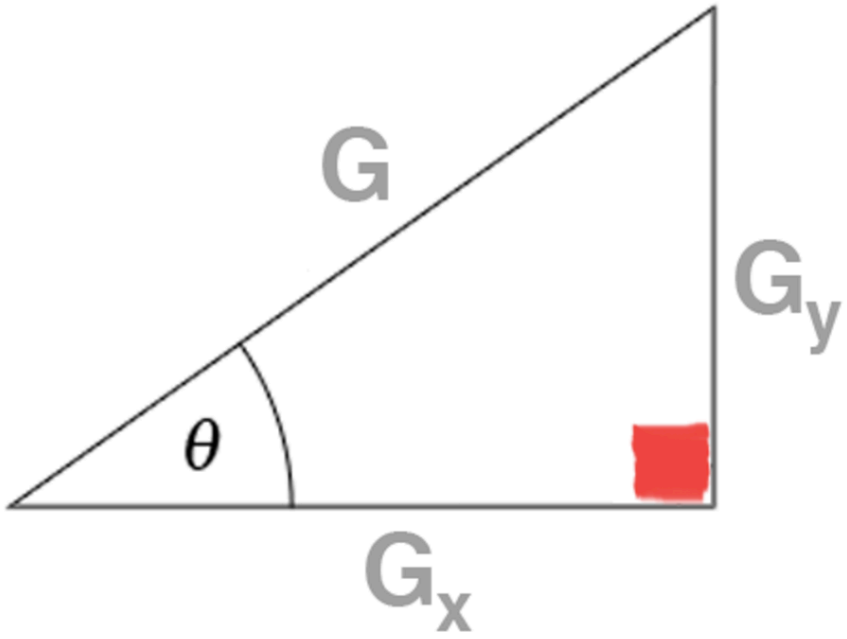
$$G_y = I(x, y-1) - I(x, y+1)$$

- Mudança horizontal (eixo x)

$$G_x = I(x+1, y) - I(x-1, y)$$



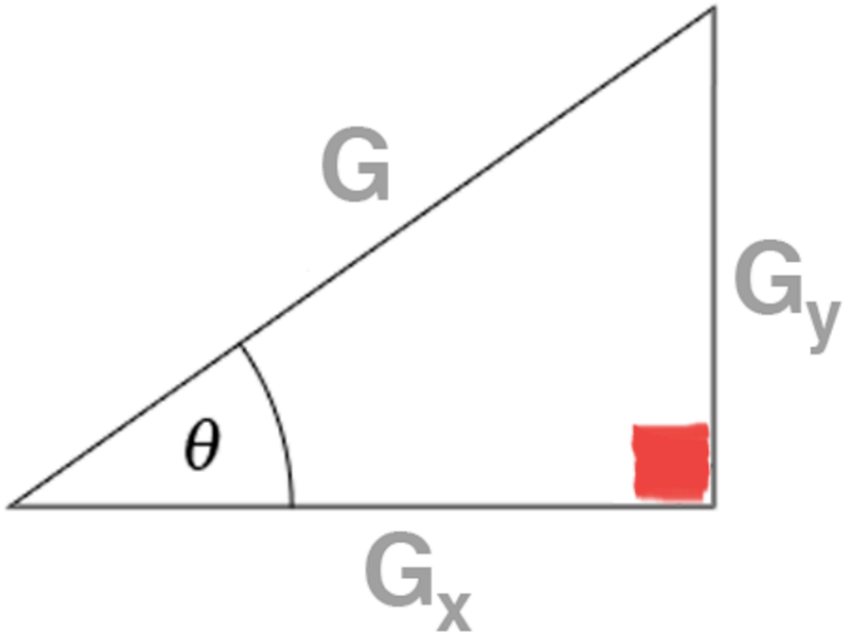
- Magnitude



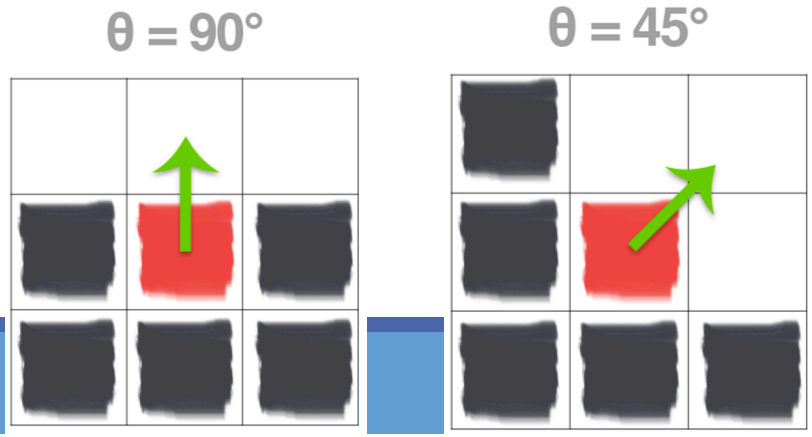
$$G = \sqrt{G_x^2 + G_y^2}$$

Gradiente Discreto

- Orientação



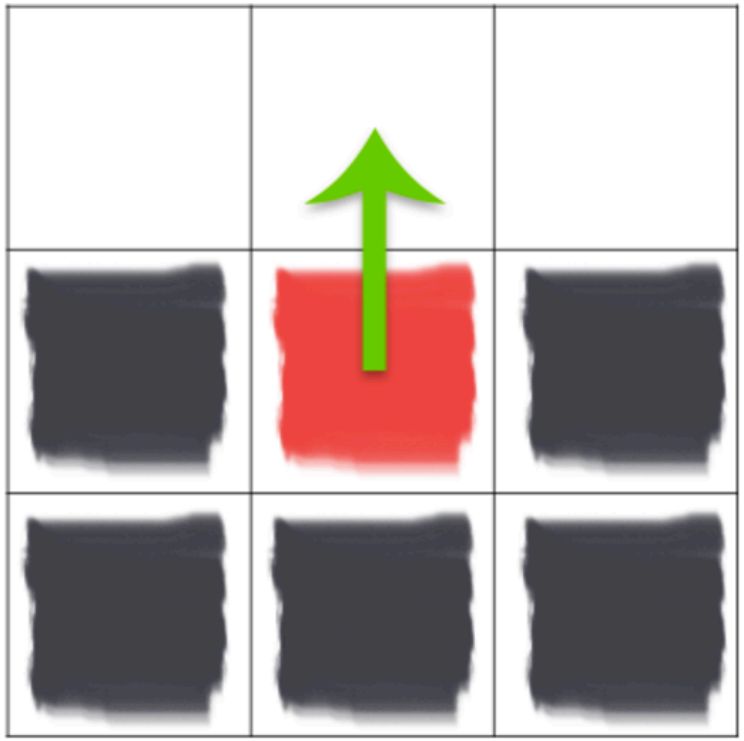
$$\theta = arctan2(G_y, G_x) \times (\frac{180}{\pi})$$



Exemplo

- Qual a magnitude e orientação do gradiente para o pixel da figura abaixo?

255	255	255
0		0
0	0	0



Exemplo

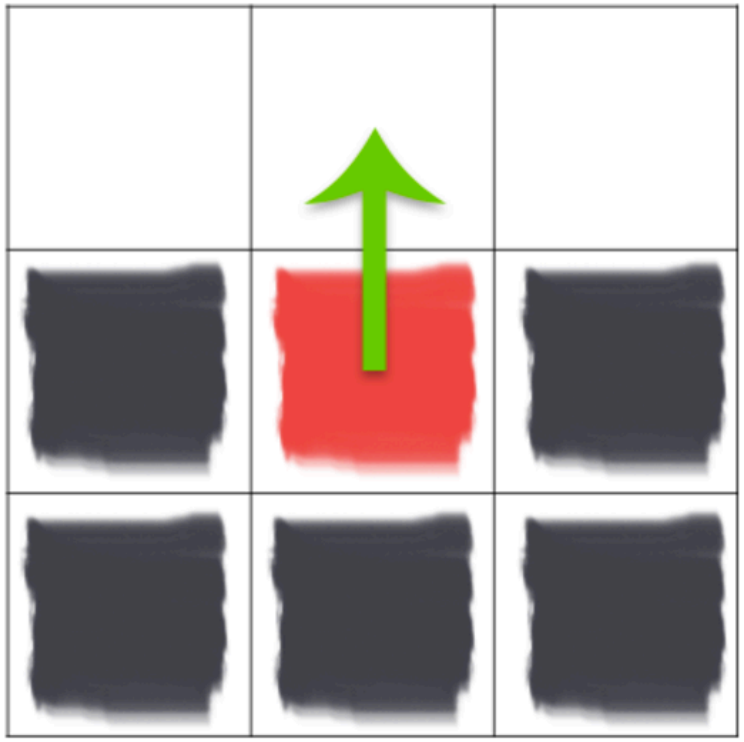
$G_x = 0 - 0 = 0$

$G_y = 255 - 0 = 255$

$G = \sqrt{0^2 + 255^2} = 255$

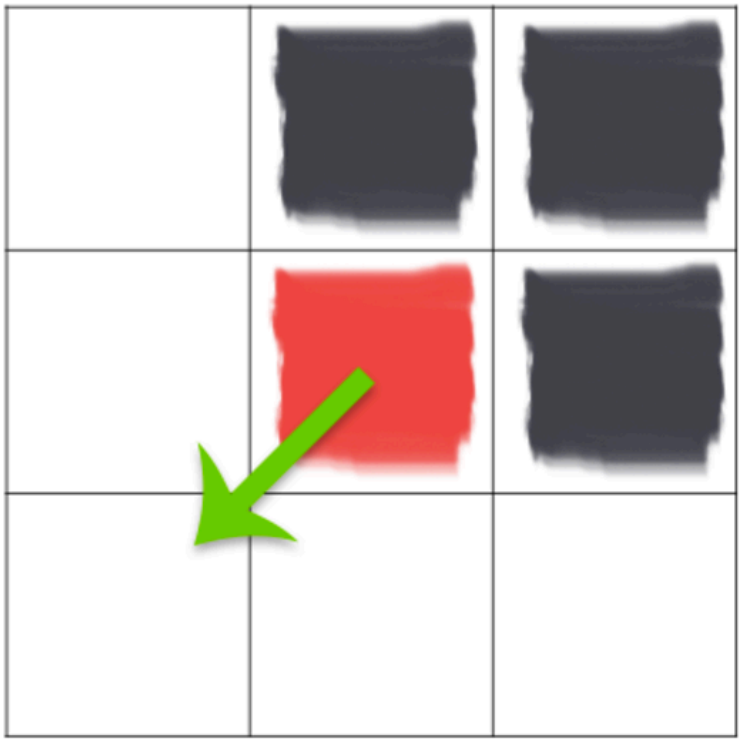
$\theta = arctan2(255,0) \times (\frac{180}{\pi})$=90

255	255	255
0		0
0	0	0



Exemplo 2

255	0	0
255		0
255	255	255



Exemplo 2

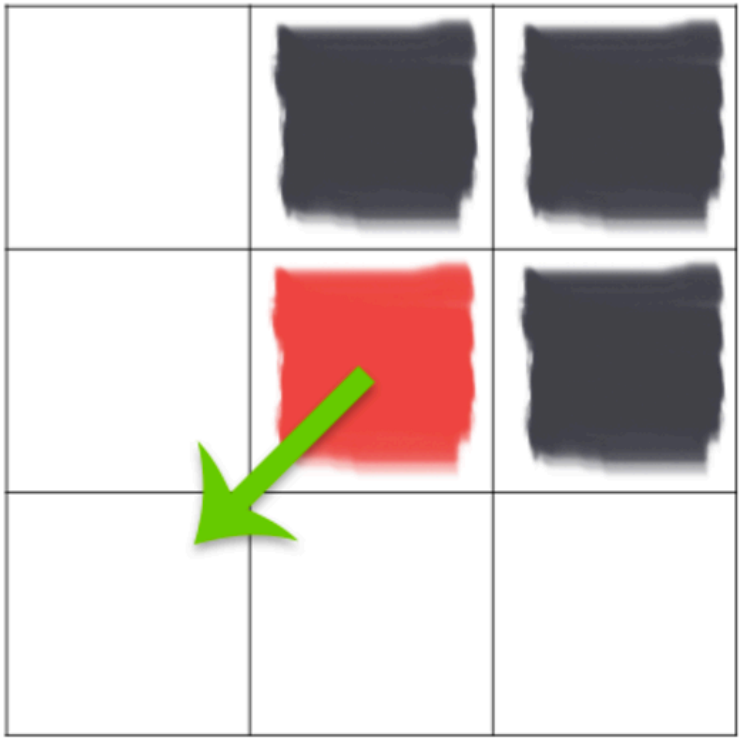
$G_x = 0 - 255 = -255$

$G_y = 0 - 255 = -255$

$G = \sqrt{255^2 + 255^2} = 360{,}2$

$\theta = arctan2(-255, -255) \times (\frac{180}{\pi})$=-135

255	0	0
255		0
255	255	255

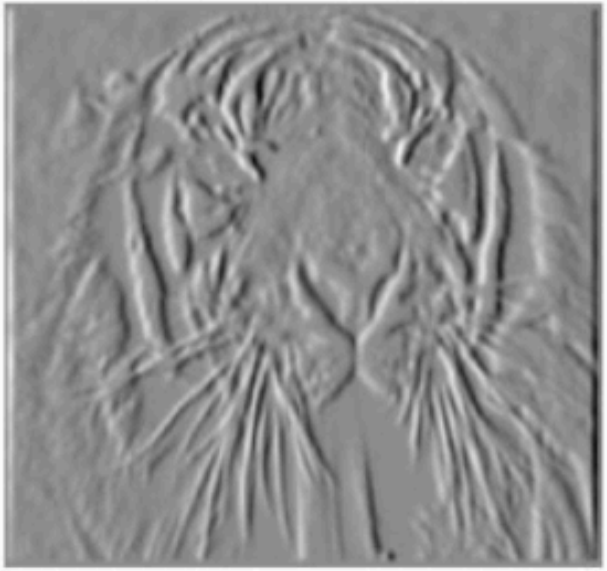


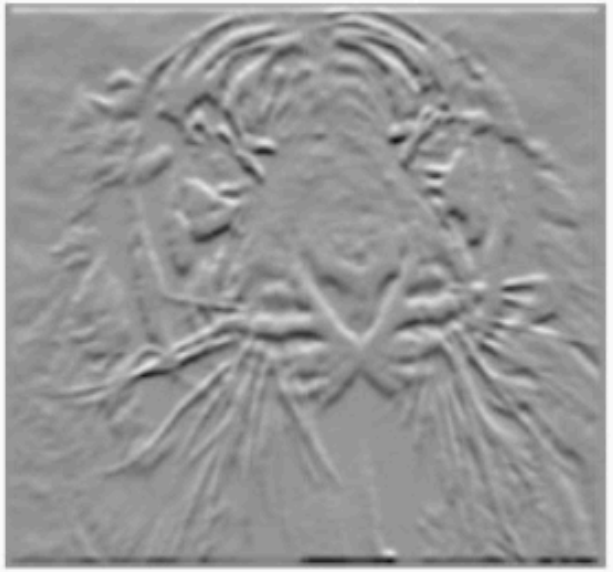
Gradientes

Imagem Original

Magnitude do Gradiente

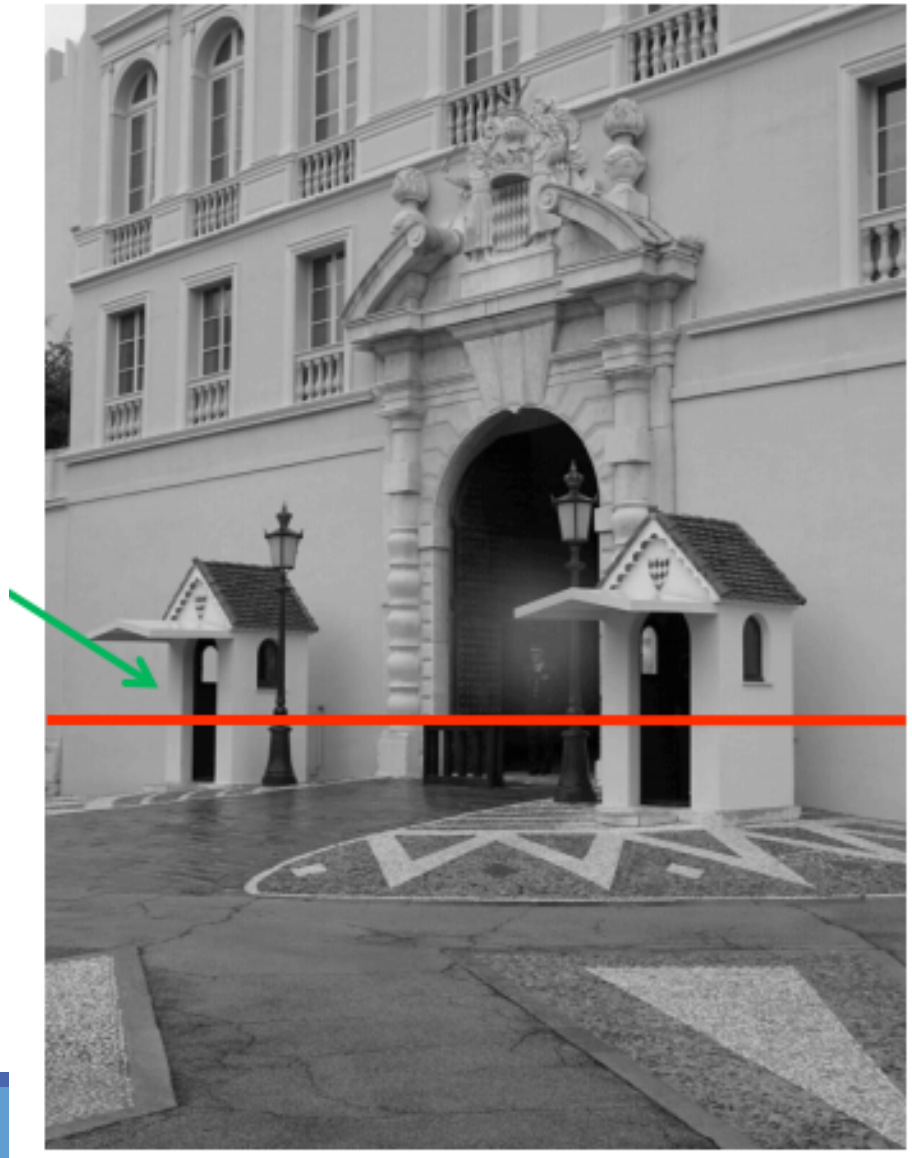
Eixo x

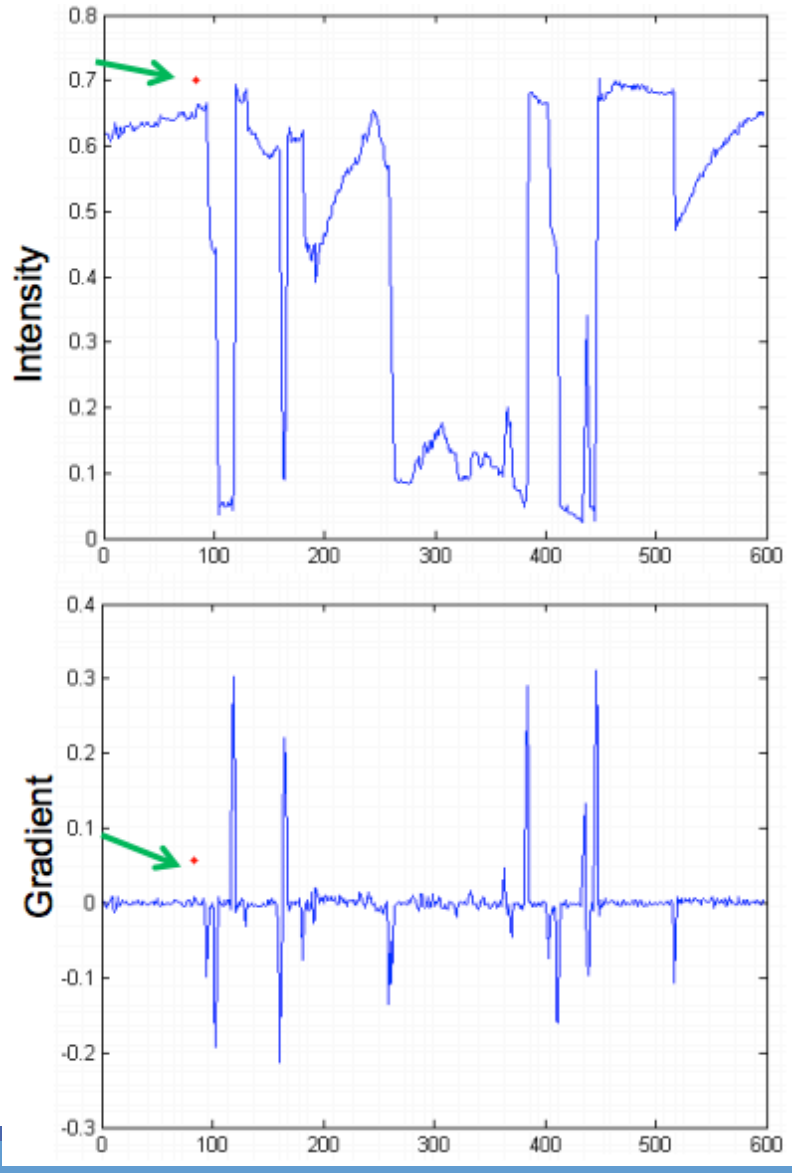




Eixo y

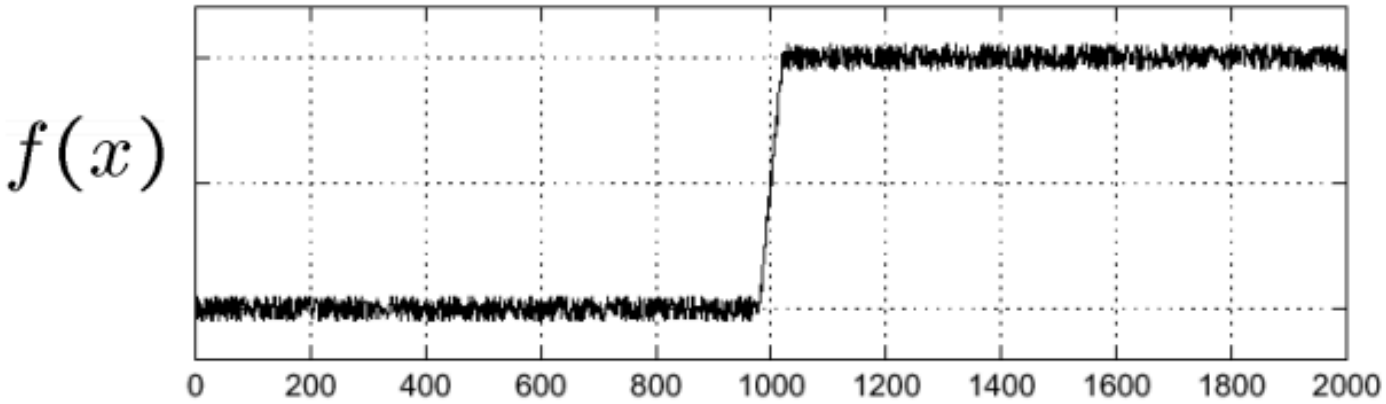
Variação de Intensidade

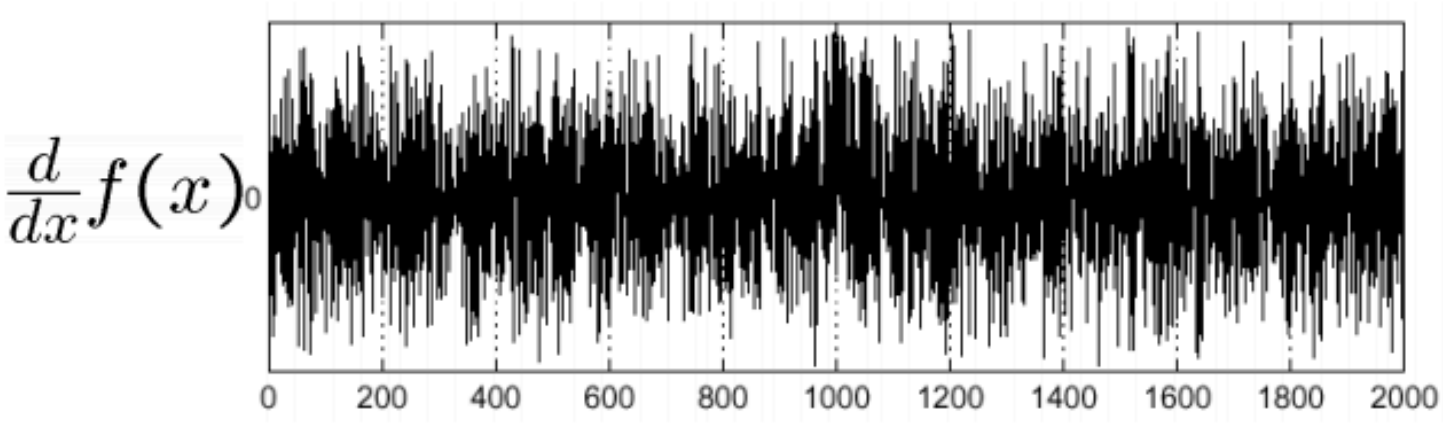




Efeitos do ruído

- Considere uma linha ou coluna da imagem





Onde está a borda?

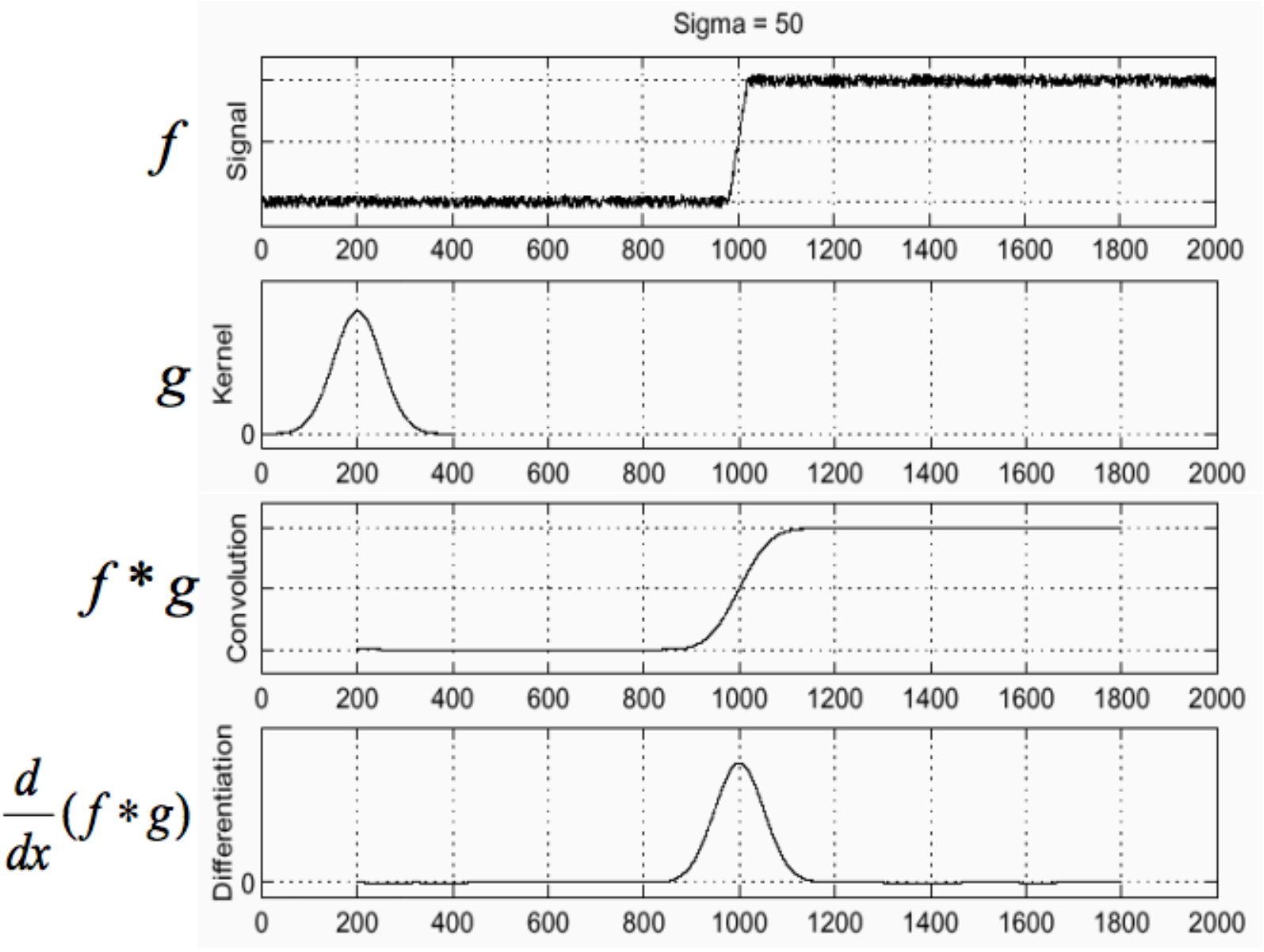
Efeitos do ruído

- Imagens com ruído resultam em pixels que parecem muito diferentes de seus vizinhos
- O que pode ser feito?

Efeitos do ruído

- Imagens com ruído resultam em pixels que parecem muito diferentes de seus vizinhos
- O que pode ser feito?
  - Suavizar a imagem, forçando pixels diferentes dos vizinhos (ruídos?) se parecerem mais com vizinhos

Solução: Suavizar primeiro



Teorema derivativo da convolução

- Propriedade:

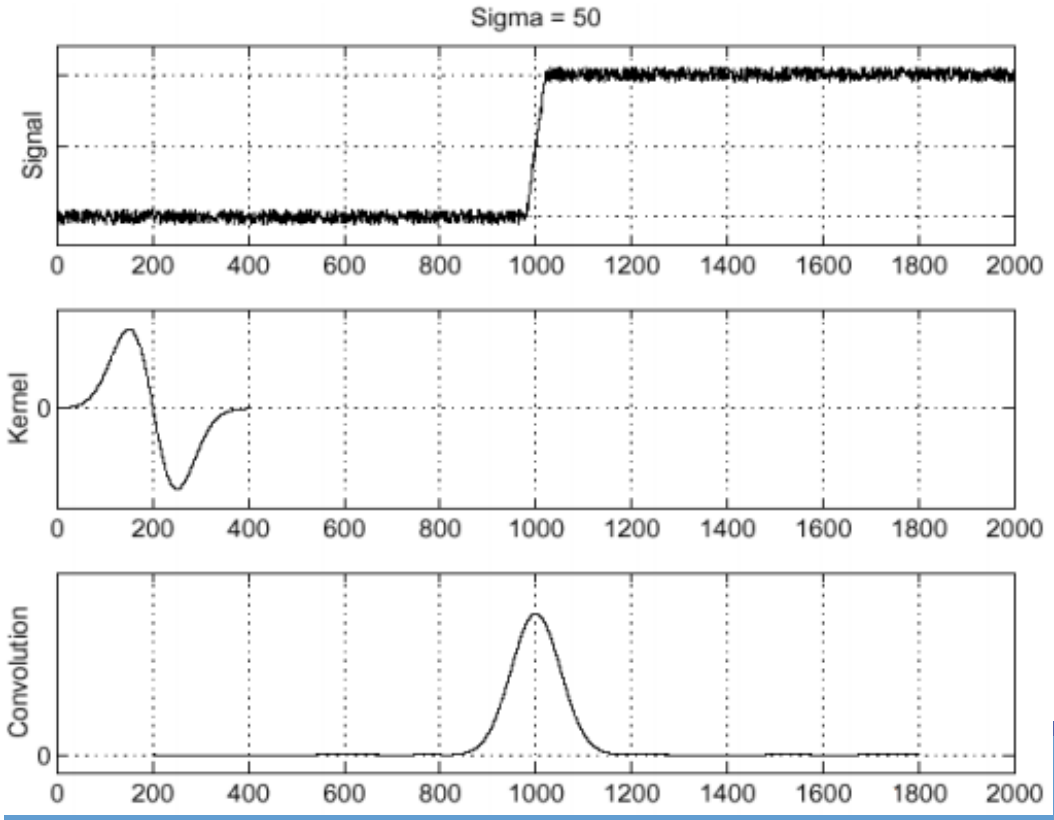
$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$

- Isso salva uma operação

f

$\frac{d}{dx} g$

$f * \frac{d}{dx} g$



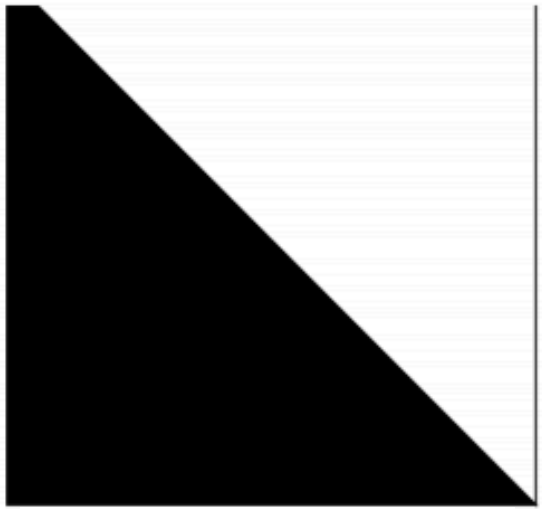
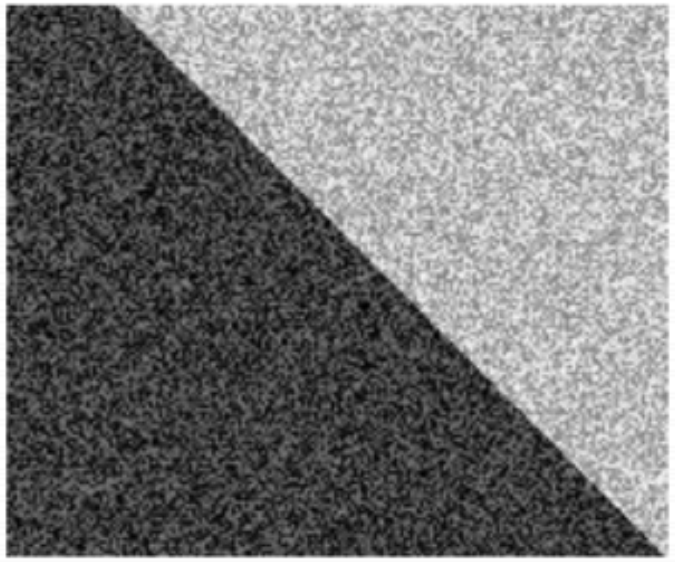
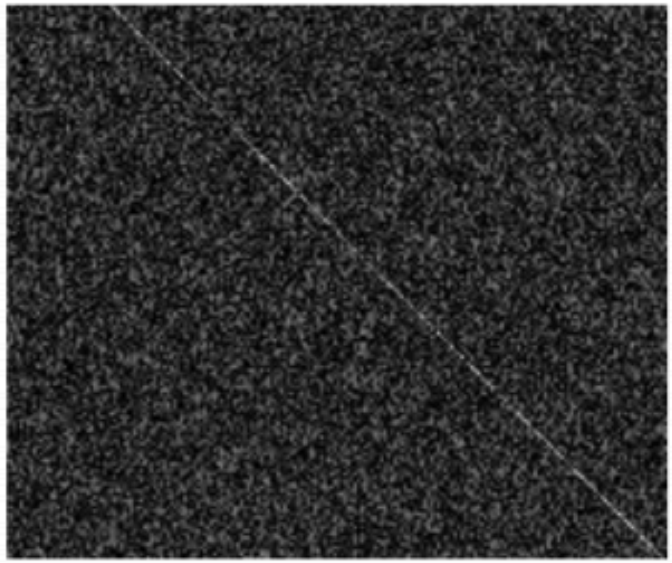


Imagem com borda

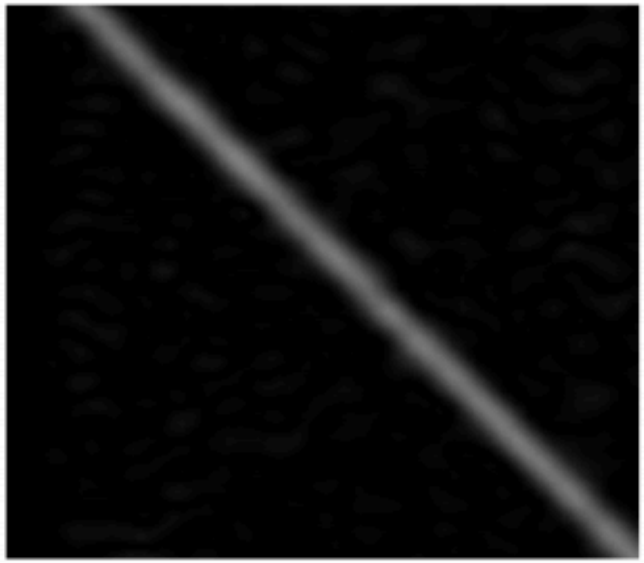
Localização da borda



Imagem+ruído



Derivadas detectam borda e ruído

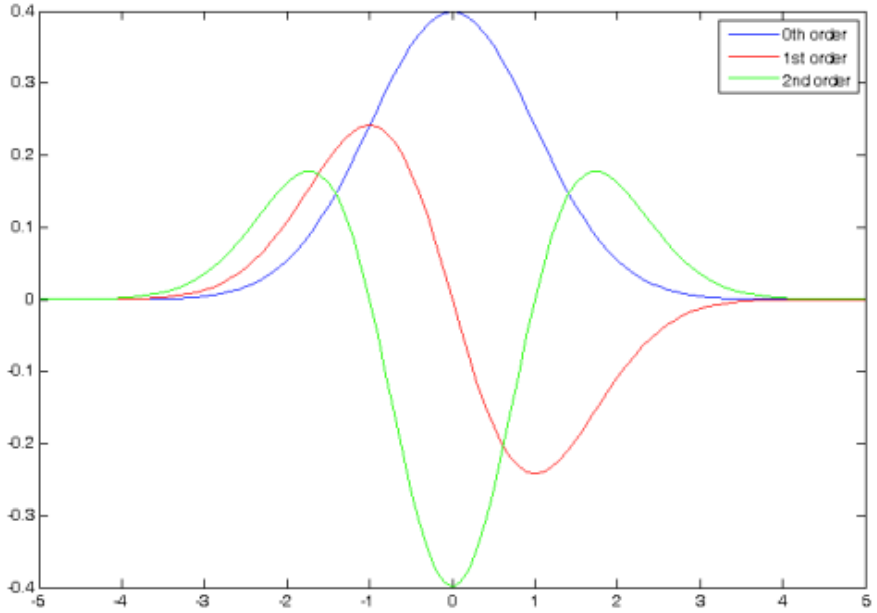


Derivada suavizada remove ruído, mas borra borda

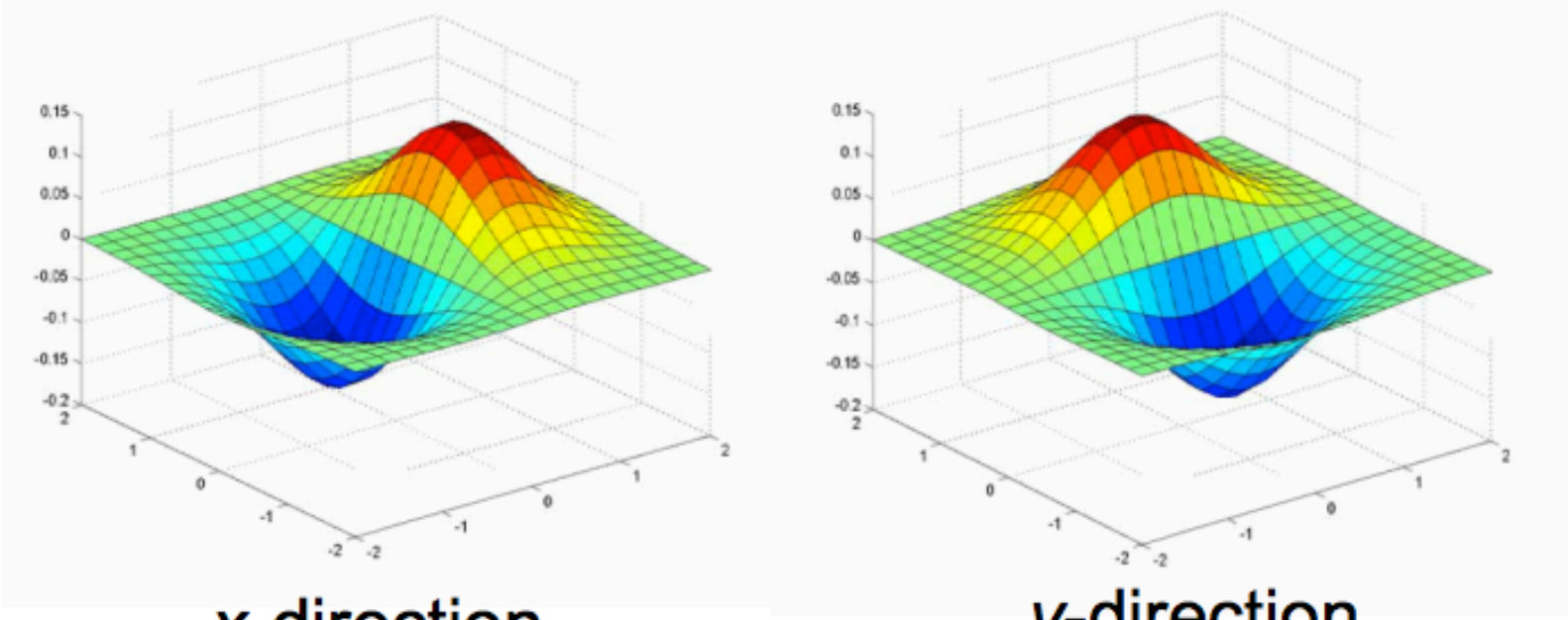
Gaussiana 1D e sua derivada

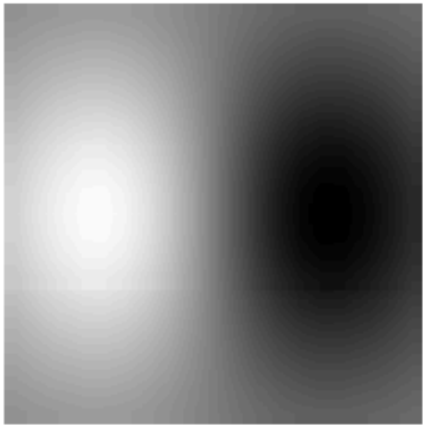
$$G_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

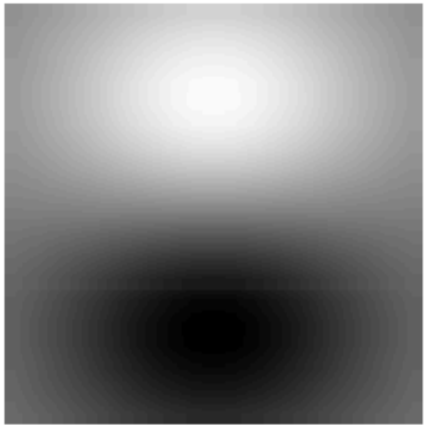
$$G'_\sigma(x) = \frac{d}{dx} G_\sigma(x) = -\frac{1}{\sigma}\left(\frac{x}{\sigma}\right) G_\sigma(x)$$



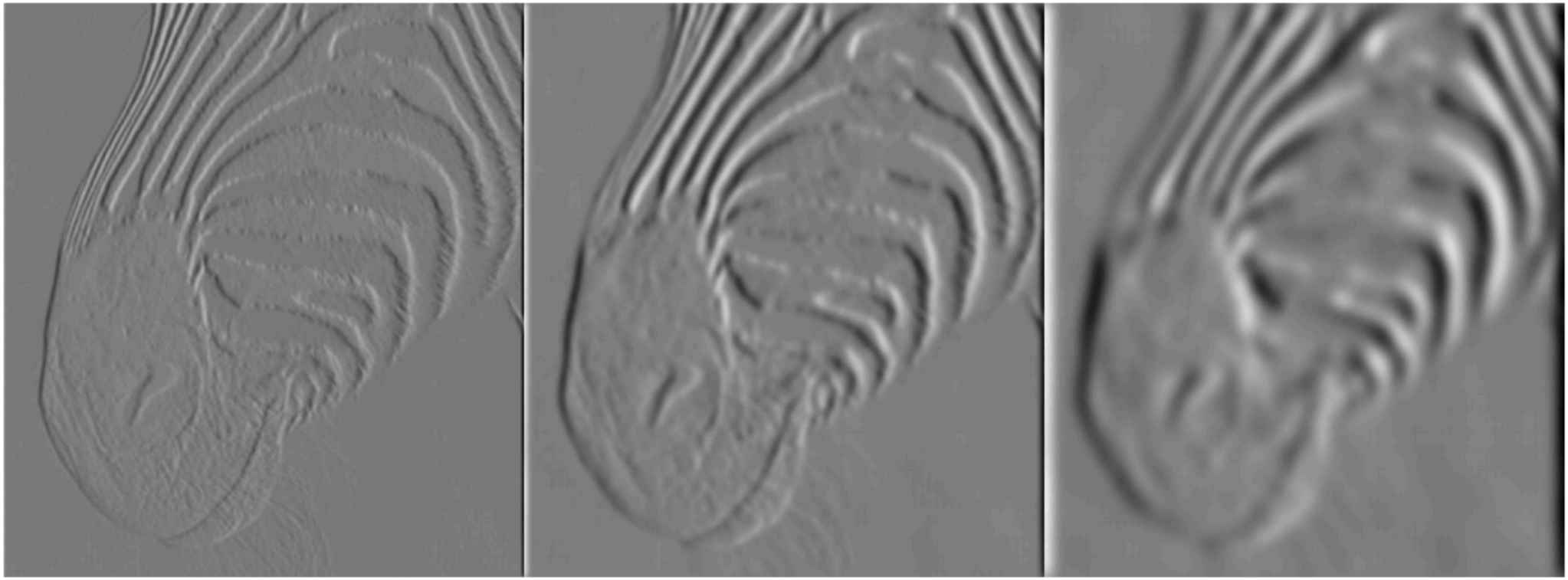
Derivada do filtro Gaussiano







Tradeoff entre suavização e localização



1 pixel 3 pixels 7 pixels

- Derivada suavizada remove ruído, mas embaça borda. Também encontra borda em diferentes `escalas`

O operador Sobel

- Aproximação da derivada da Gaussiana
  - Máscara:

$\frac{1}{8}$

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

s_x

$\frac{1}{8}$

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

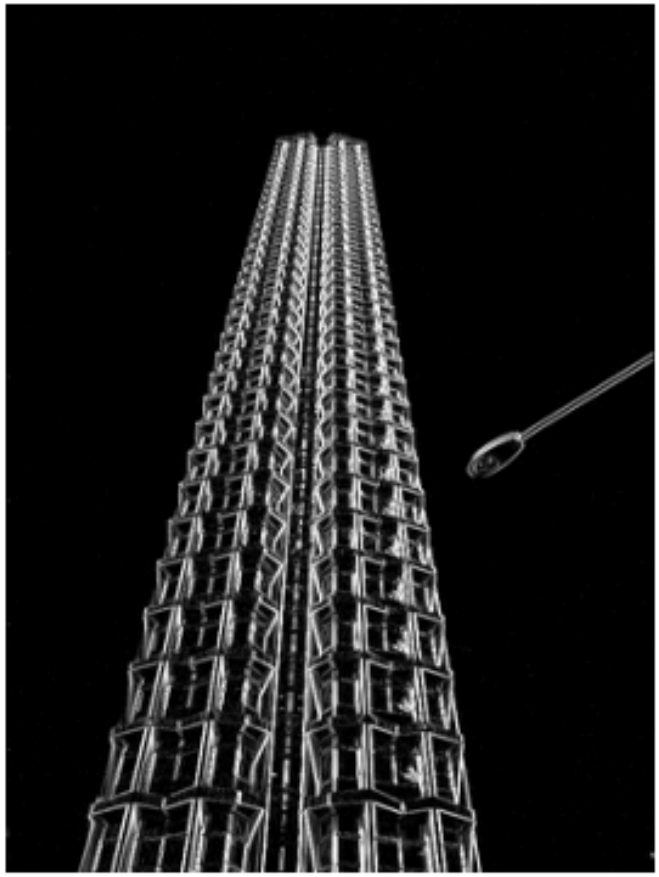
s_y

  - A definição padrão omite o termo 1/8
    - Não faz diferença para detecção
    - O termo 1/8 é para dar a magnitude de gradiente correta

Sobel: exemplo





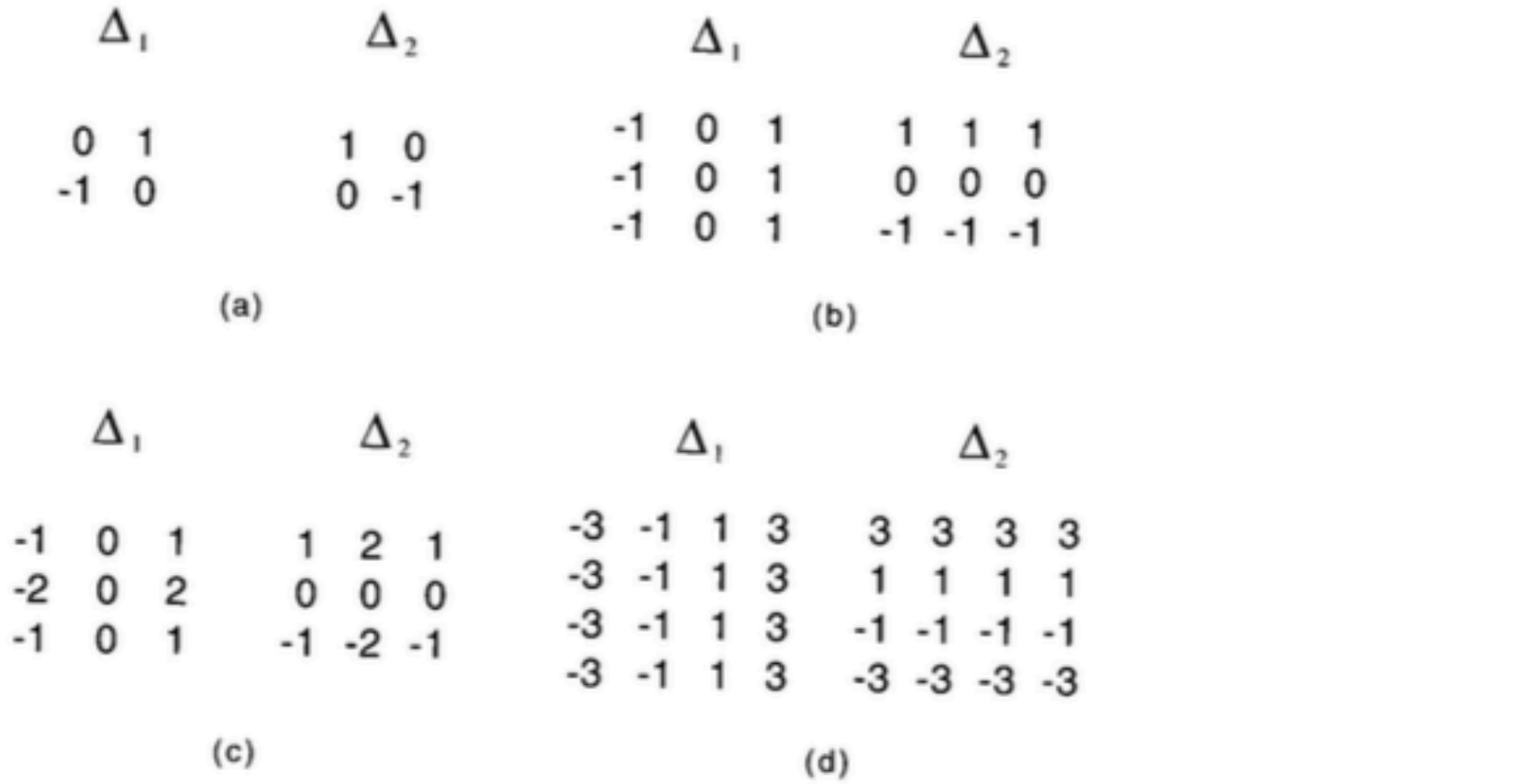


Derivative in X direction

Derivative in Y direction

Visualize with scaled absolute value

Outros Operadores



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

Detecção de Linhas

-1	-1	-1
2	2	2
-1	-1	-1

Horizontal

-1	-1	2
-1	2	-1
2	-1	-1

+45°

-1	2	-1
-1	2	-1
-1	2	-1

Vertical

2	-1	-1
-1	2	-1
-1	-1	2

-45°

Intensifica regiões
horizontais em uma imagem

Detecção de Bordas

- Operadores de Gradiente (Sobel)

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

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

Detecção de Bordas

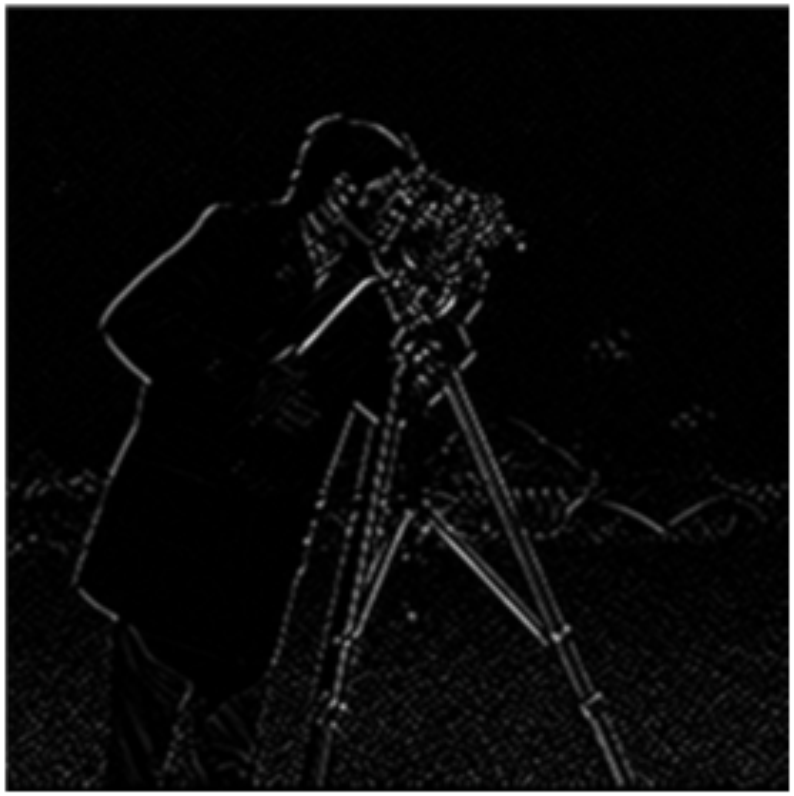
- Máscaras Ortogonais (Frei e Chen)

0	1	0
-1	0	-1
0	1	0

Detecção de Bordas

- Máscaras Ortogonais (Frei e Chen)

-1	0	1
0	0	0
1	0	-1



Detecção de Bordas

- Máscaras Ortogonais (Frei e Chen)

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

Detecção de Bordas

- Máscaras Ortogonais (Frei e Chen)

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

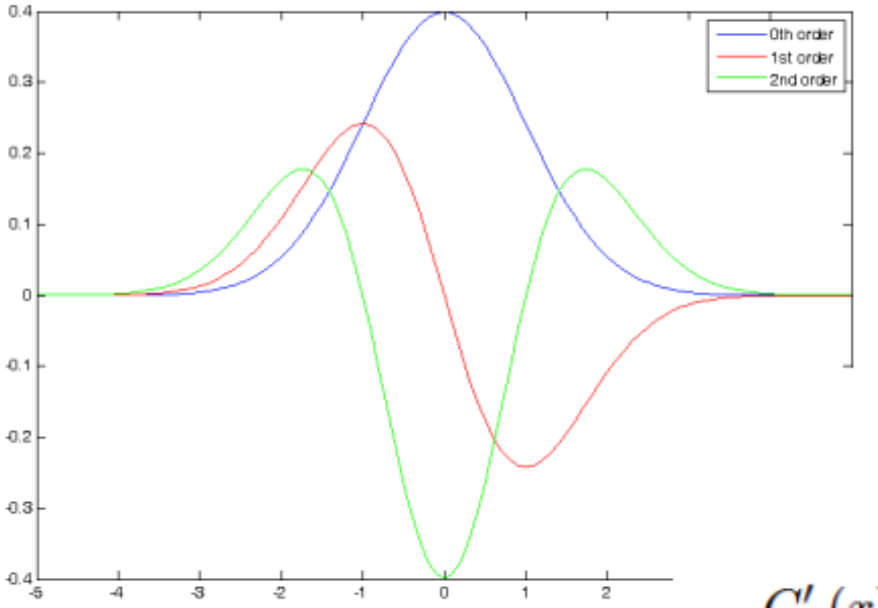
Detecção de Bordas

- Compare a saída do operador de Sobel com a de Roberts
  - O ruído aparece com menor intensidade em Roberts
  - O operador de Roberts perdeu algumas bordas
  - O operador de Sobel detecta bordas mais finas.

Sobel

Roberts

Gaussiana 1D suas derivadas



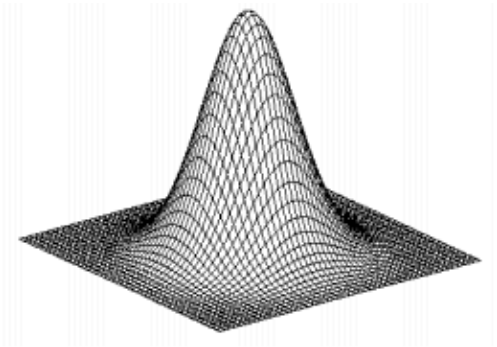
$$G_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

$$G'_\sigma(x) = \frac{d}{dx} G_\sigma(x) = -\frac{1}{\sigma}\left(\frac{x}{\sigma}\right) G_\sigma(x)$$

$$G''_\sigma(x) = \frac{d^2}{dx^2} G_\sigma(x) = \frac{1}{\sigma^2}\left(\frac{x^2}{\sigma^2} - 1\right) G_\sigma(x)$$

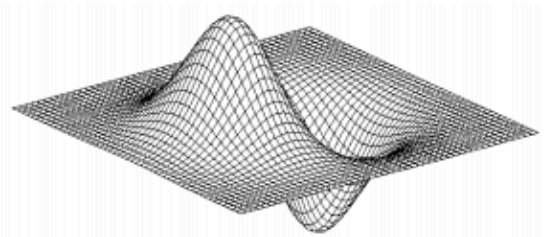
$G'_\sigma(x)$'s maxima/minima occur at $G''_\sigma(x)$'s zeros. And, we can see that $G'_\sigma(x)$ is an odd symmetric function and $G''_\sigma(x)$ is an even symmetric function.

Operador Laplaciano



Gaussian

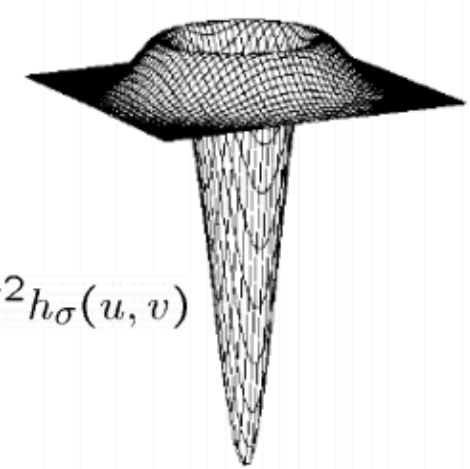
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Laplacian of Gaussian



$$\nabla^2 h_\sigma(u,v)$$

∇^2 is the **Laplacian** operator:

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

2D Laplace filter

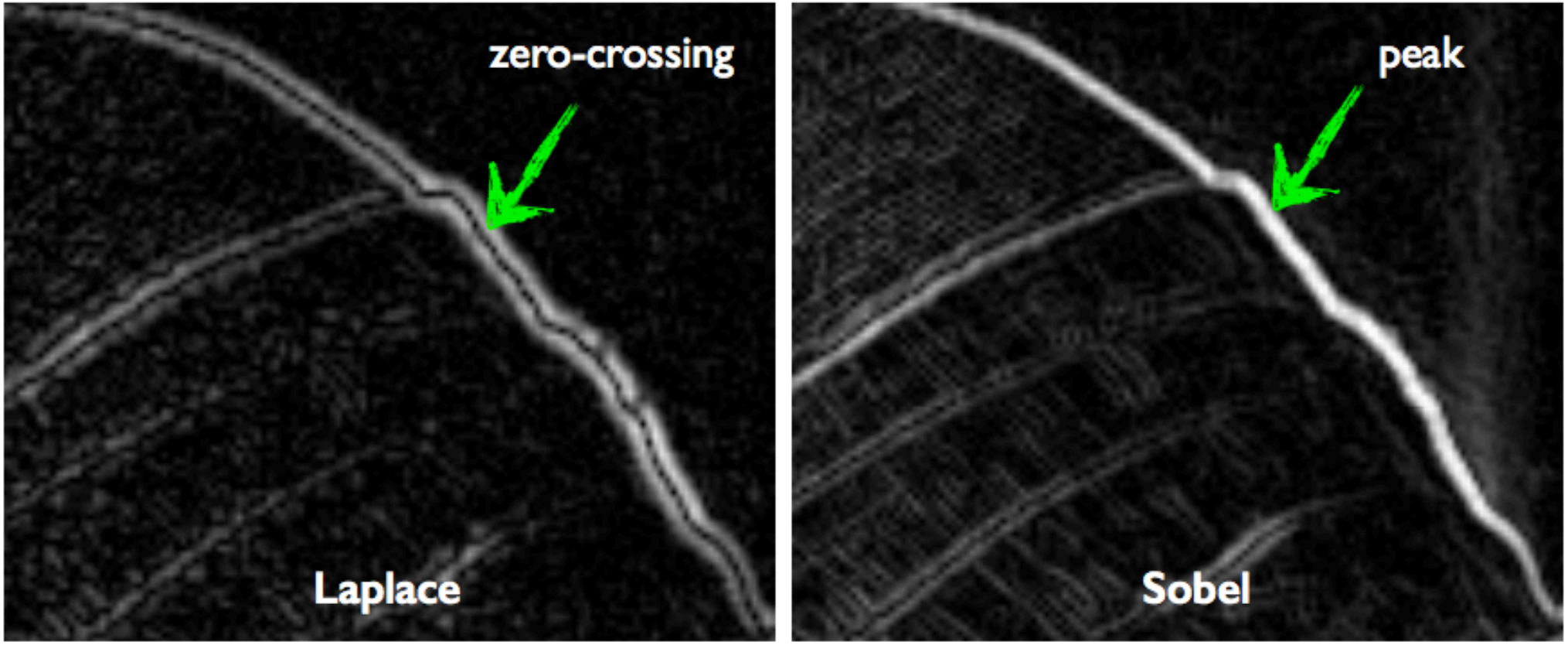
1	-2	1

1D Laplace filter

0	1	0
1	-4	1
0	1	0

2D Laplace filter

- Qual a diferença entre os resultados?



Zero crossings are more accurate at localizing edges

(but not very convenient)

Derivada de Segunda Ordem

- Laplaciano de Gaussiana

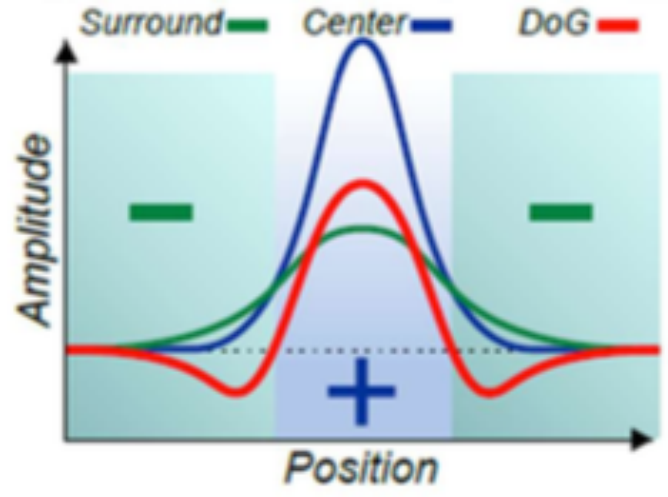
Derivada de Segunda Ordem

- Laplaciano de Gaussiana (Binarizada)

Derivada de Segunda Ordem

- Diferença de Gaussianas (DoG)

```
g1 = cv2.getGaussianKernel(5,0.3)
g2 = cv2.getGaussianKernel(5,0.5)
dog = g1-g2
final = cv2.filter2D(img, cv2.CV_8U, dog)
```



(a) Spatial Plot

Exemplo

- Código

Derivada de Segunda Ordem

- Diferença de Gaussianas

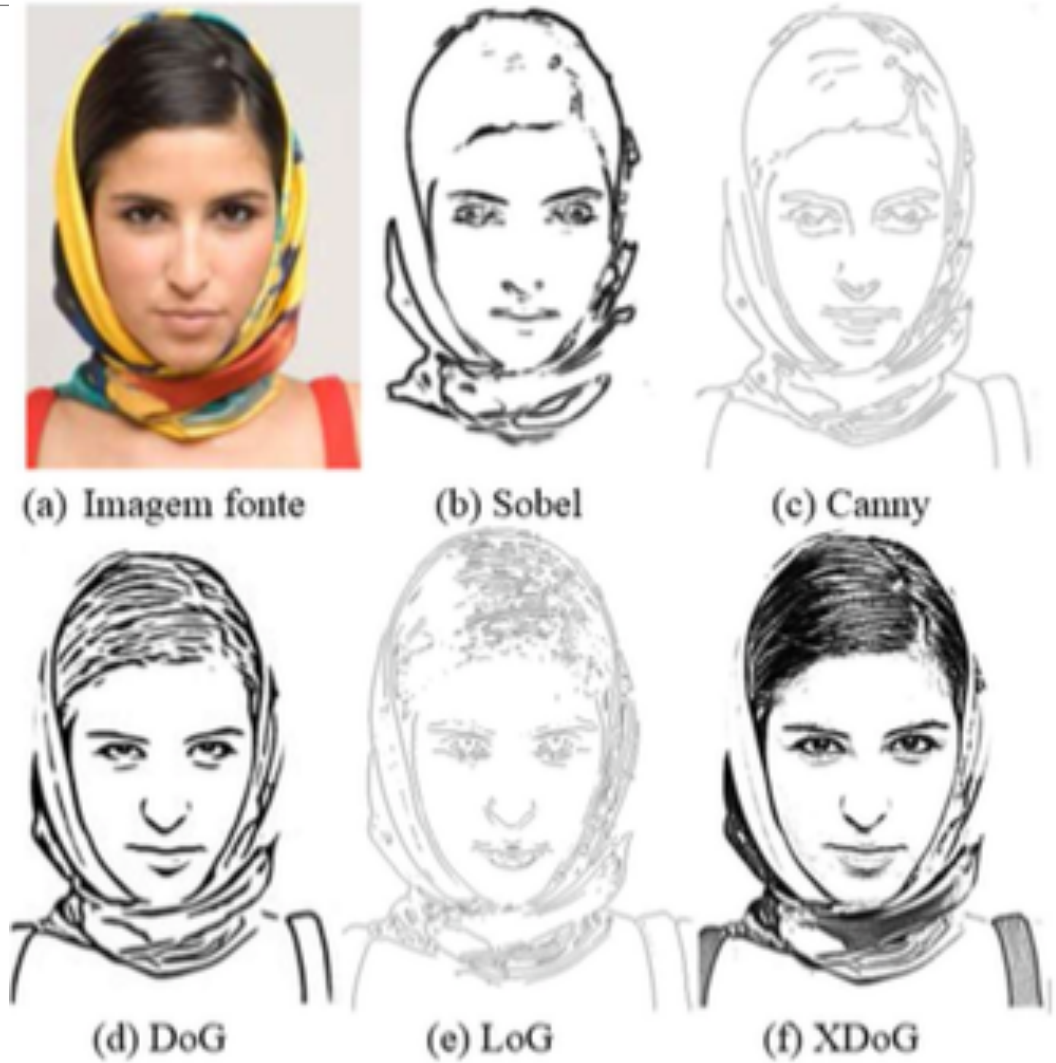
Derivada de Segunda Ordem

- Diferença de Gaussianas (Binarizada)

Derivada de Segunda Ordem

- Diferença de Gaussianas (enhanced)

Diferença de Gaussianas Estendida



Fonte: H.Winnemöller, "XDoG: Advanced Image Stylization with eXtended Difference-of-Gaussians"

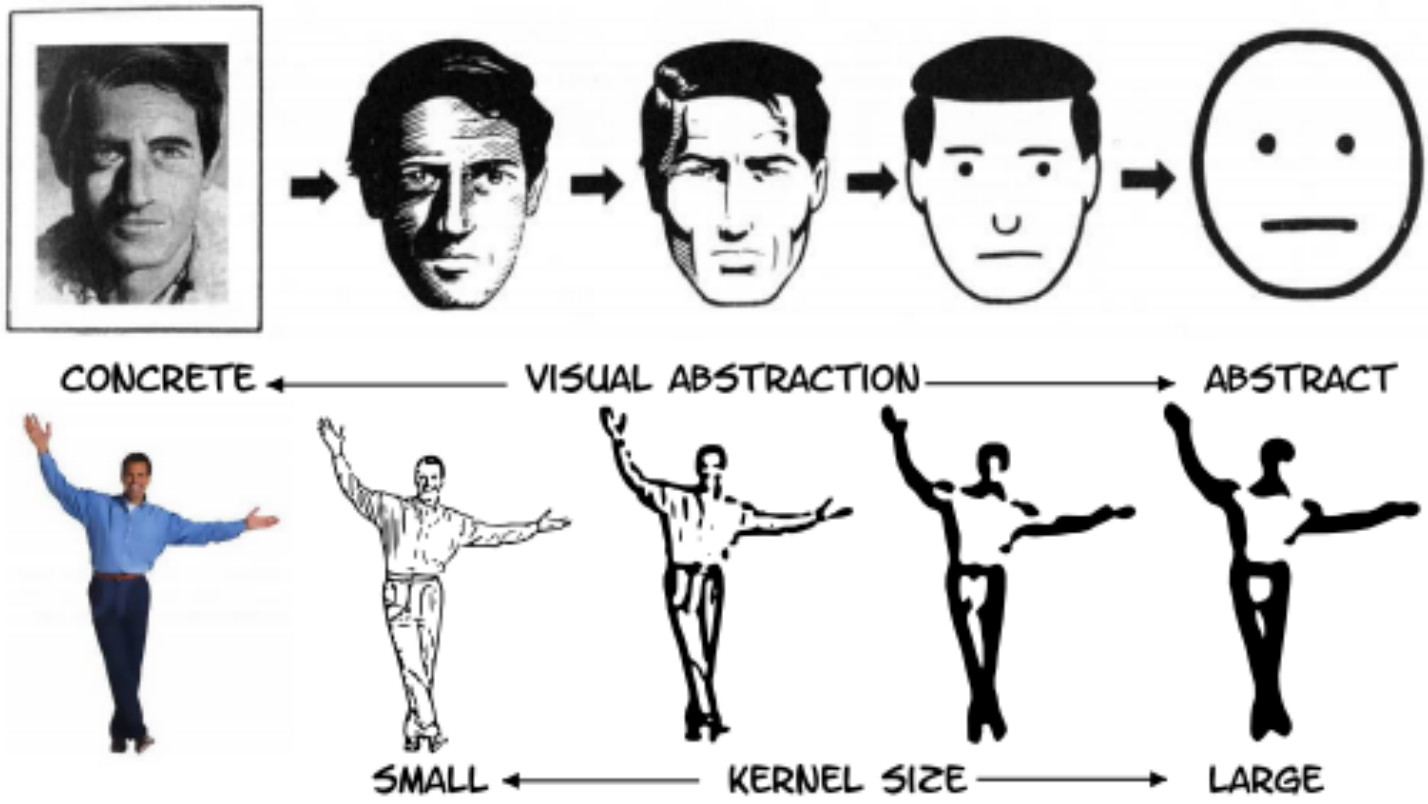
XDoG

$$T_{\varepsilon,\varphi}(u) = \begin{cases} 1 & u \geq \varepsilon \\ 1 + \tanh(\varphi \cdot (u - \varepsilon)) & \text{otherwise.} \end{cases}$$

$$u = S_{\sigma,k,\tau} * Imagem$$

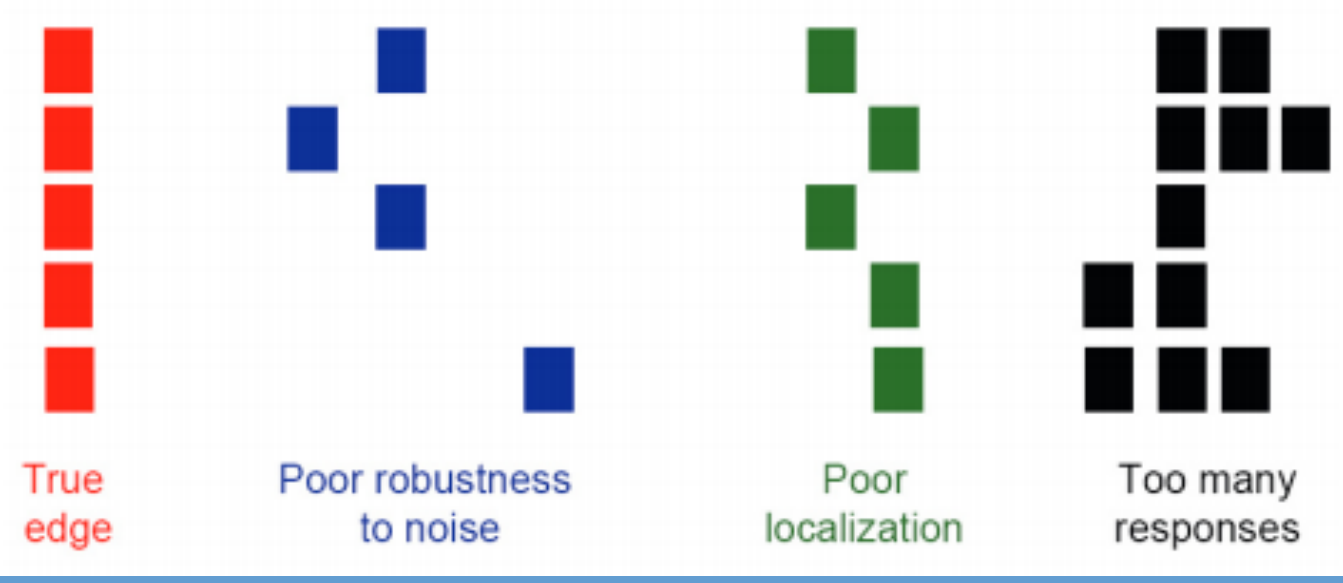
$$S_{\sigma,k,p}(x) = (1 + p) \cdot G_{\sigma}(x) - p \cdot G_{k\sigma}(x)$$

XDoG


CONCRETE
VISUAL ABSTRACTION
ABSTRACT
SMALL
KERNEL SIZE
LARGE

Desenvolvendo um detector de borda

- Critérios par detector de borda `ótimo`:
  - **Boa detecção**: o detector ótimo deve minimizar a probabilidade de falsos positivos, assim como falso negativos
  - **Boa localização**: as bordas detectadas devem ser a mais próximas possíveis das bordas verdadeiras
  - **Resposta única:** o detector deve retornar apenas um ponto para cada ponto real de borda. Isto é, minimizar o número de píxels em volta da borda real.



Detector de Canny

- Um dos mais usados detectores de borda em visão computacional

Detector de Canny

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

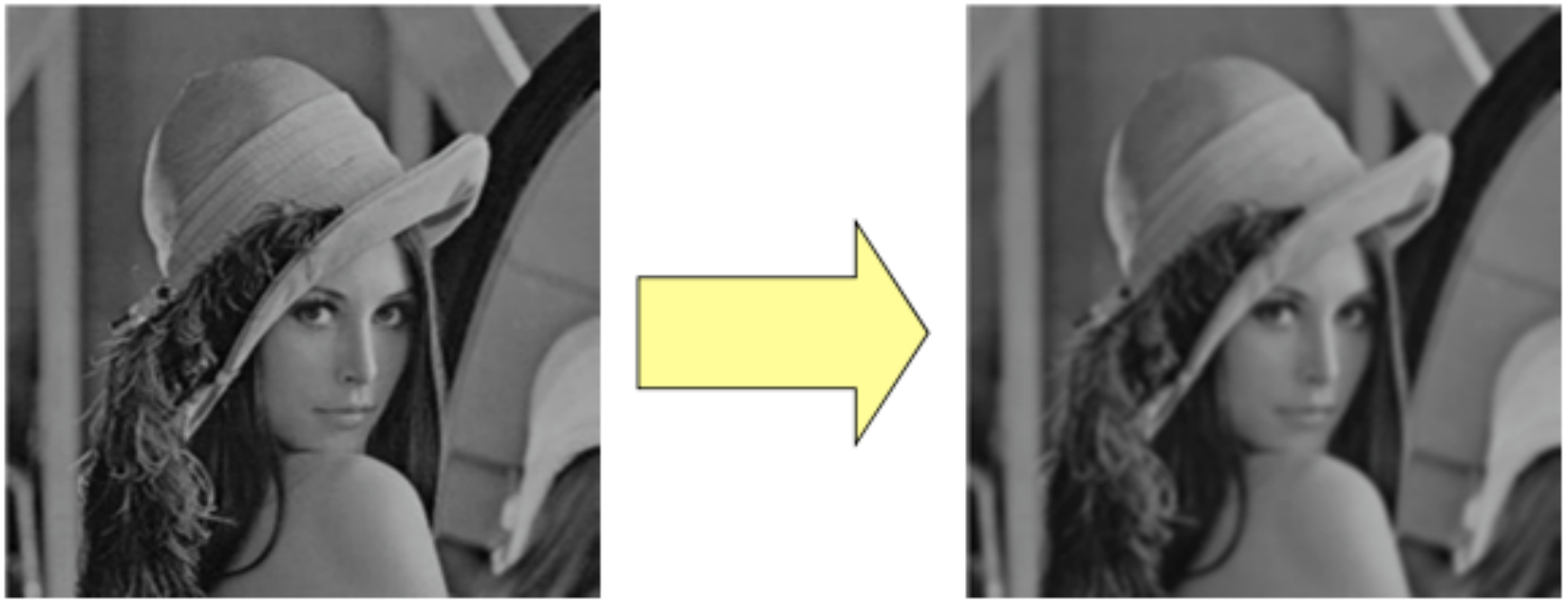
Detector de Canny

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

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

Detector de Canny

- Algoritmo (na prática)
  - Suavização



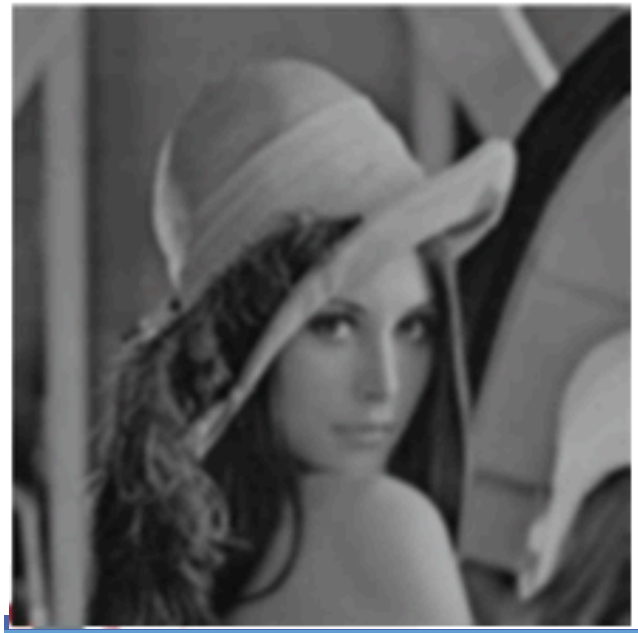
Detector de Canny

- Algoritmo (na prática)
  - Busca por gradientes

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

Gx

Gy



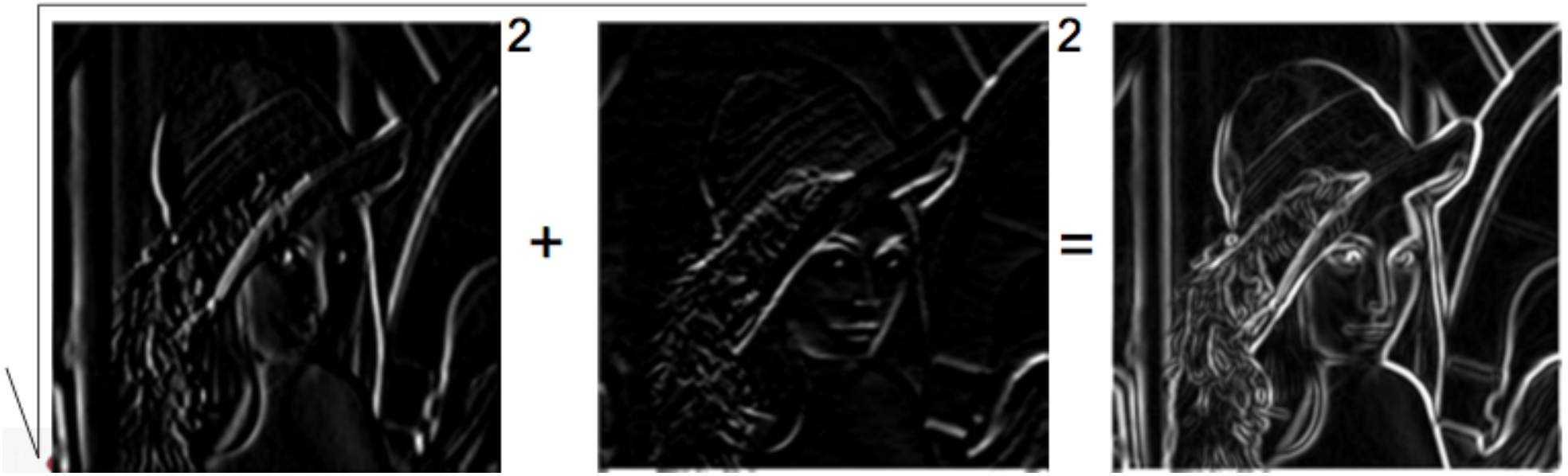
Detector de Canny

- Algoritmo (na prática)
  - A magnitude do gradiente pode ser encontrada como

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

ou

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

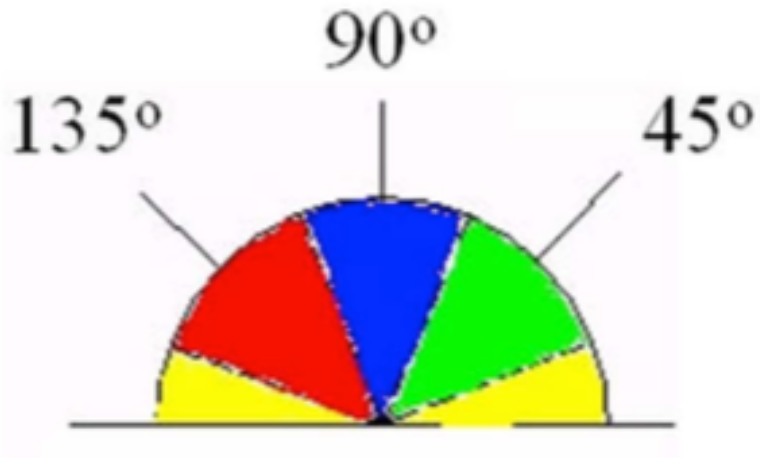


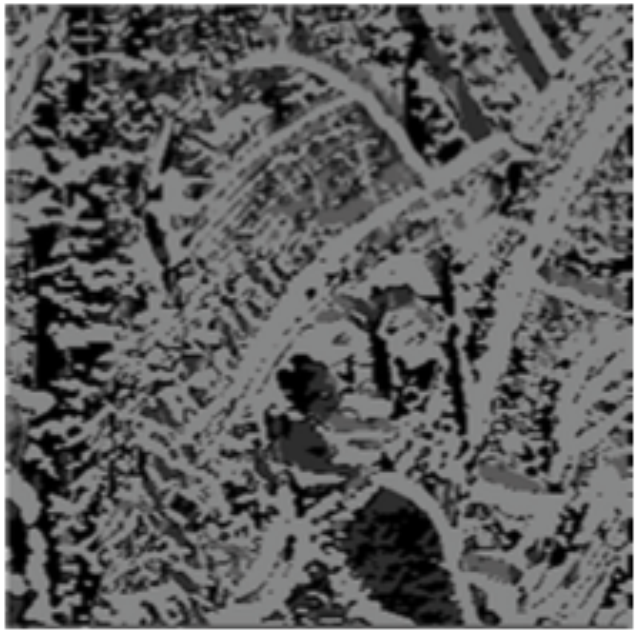
Detector de Canny

- Algoritmo (na prática)
  - A direção das arestas é calculada e armazenada:

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

  - E aproximada para



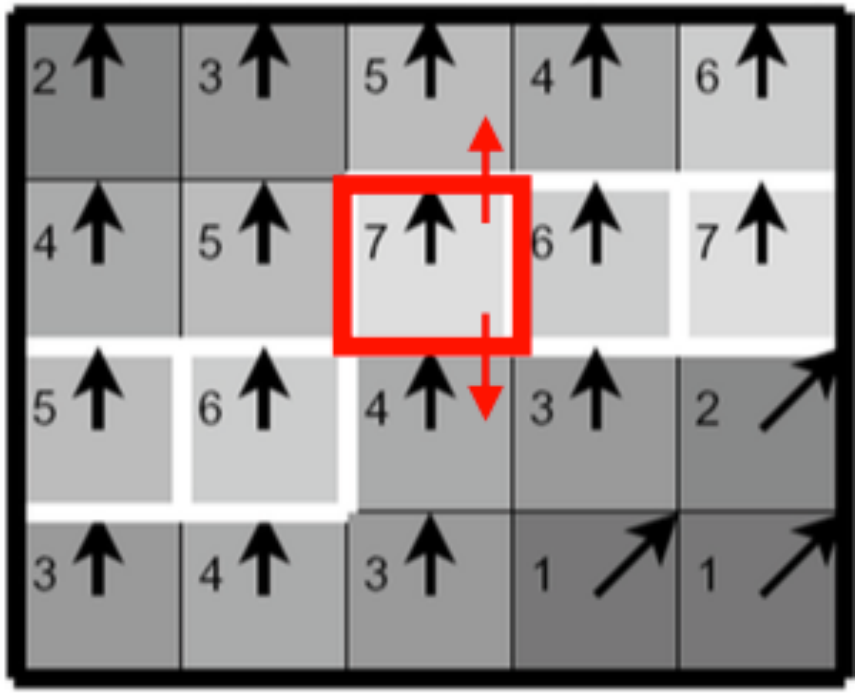


Detector de Canny

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

Detector de Canny

- Algoritmo (na prática)
  - Non-Maximum Suppression:
    - Exemplo: pixels apontando para o norte são comparados com pixel abaixo e acima



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

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

Antes do Non-max Suppression

Depois do non-max suppression

Detector de Canny

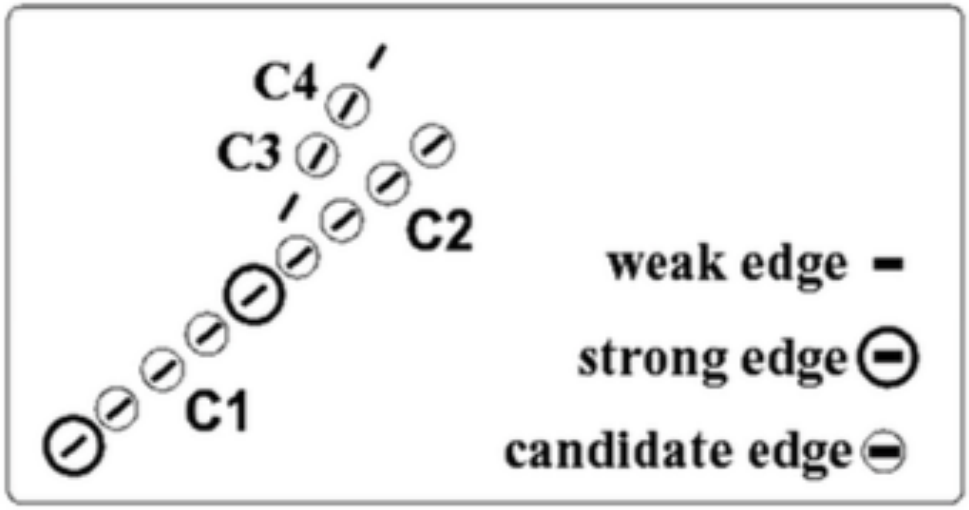
- Algoritmo (na prática)
  - Hysteresis thresholding
    - Dois limiares: um alto e um baixo
    - Qualquer pixel acima do maior limiar é convertido para branco. Os pixels ao redor são então analisados: se seus valores são maiores que o menor limiar, eles são convertidos para branco
      - Um pixel (x,y) é chamado forte se cor(x,y) > Th_alto
      - Um pixel (x,y) é chamado fraco se cor(x,y)<=Th_baixo
      - Todos os outros pixels são chamados de candidatos

Detector de Canny

- Algoritmo (na prática)
  - Hysteresis Thresholding + Edge Tracking
    - Em cada posição (x,y), descarta o pixel (x,y) se ele é fraco; permanece na imagem de saída se for forte
    - Se um pixel é um candidato, segue a cadeia de máximos locais conectados em ambas as direções na direção da aresta, enquatno cor(i,j) > Th_baixo
    - Se um pixel candidato inicial está conectado a um pixel forte, ele não é suprimido; do contrário, é.

Detector de Canny

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



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

Canny Edges

Detector de Canny

- Exemplo: Variações nos parâmetros

Gaussiana: 5x5, $\sigma = 1.4$
Thresholding:
Th_{baixo} = 10% do máximo
Th_{alto} = 30% do máximo

Detector de Canny

- Exemplo: Variações nos parâmetros

Gaussiana: 5x5, $\sigma = 5$
Thresholding:
Th_{baixo} = 10% do máximo
Th_{alto} = 30% do máximo

Detector de Canny

- Exemplo: Variações nos parâmetros

Gaussiana: 5x5, $\sigma = 1.4$
Thresholding:
Th_{baixo} = 10% do máximo
Th_{alto} = 20% do máximo

Detector de Canny

- Exemplo: Variações nos parâmetros

Gaussiana: 5x5, $\sigma = 1.4$
Thresholding:
Th_{baixo} = 5% do máximo
Th_{alto} = 10% do máximo

Detector de Canny

- Exemplo: Variações nos parâmetros

Gaussiana: 5x5, $\sigma = 0.5$
Thresholding:
Th_{baixo} = 5% do máximo
Th_{alto} = 10% do máximo

Detector de Canny

- Exemplo: Variações nos parâmetros

Gaussiana: 5x5, $\sigma = 0.5$
Thresholding:
Th_{baixo} = 10% do máximo
Th_{alto} = 30% do máximo

Detector de Canny

- Exemplo: Variações nos parâmetros

Gaussiana: 5x5, $\sigma = 0.5$
Thresholding:
Th_{baixo} = 10% do máximo
Th_{alto} = 20% do máximo

Detector de Canny

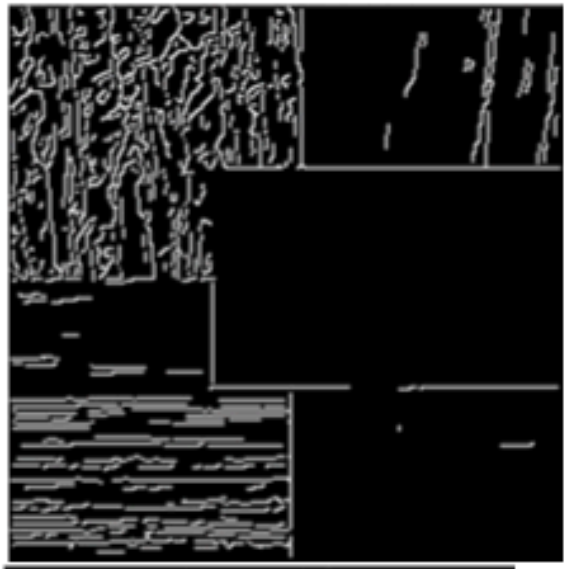
- Exemplo:

Gaussiana: 5x5, $\sigma = 0.5$
Thresholding:
Th_{baixo} = 10% do máximo
Th_{alto} = 20% do máximo

Detector de Canny

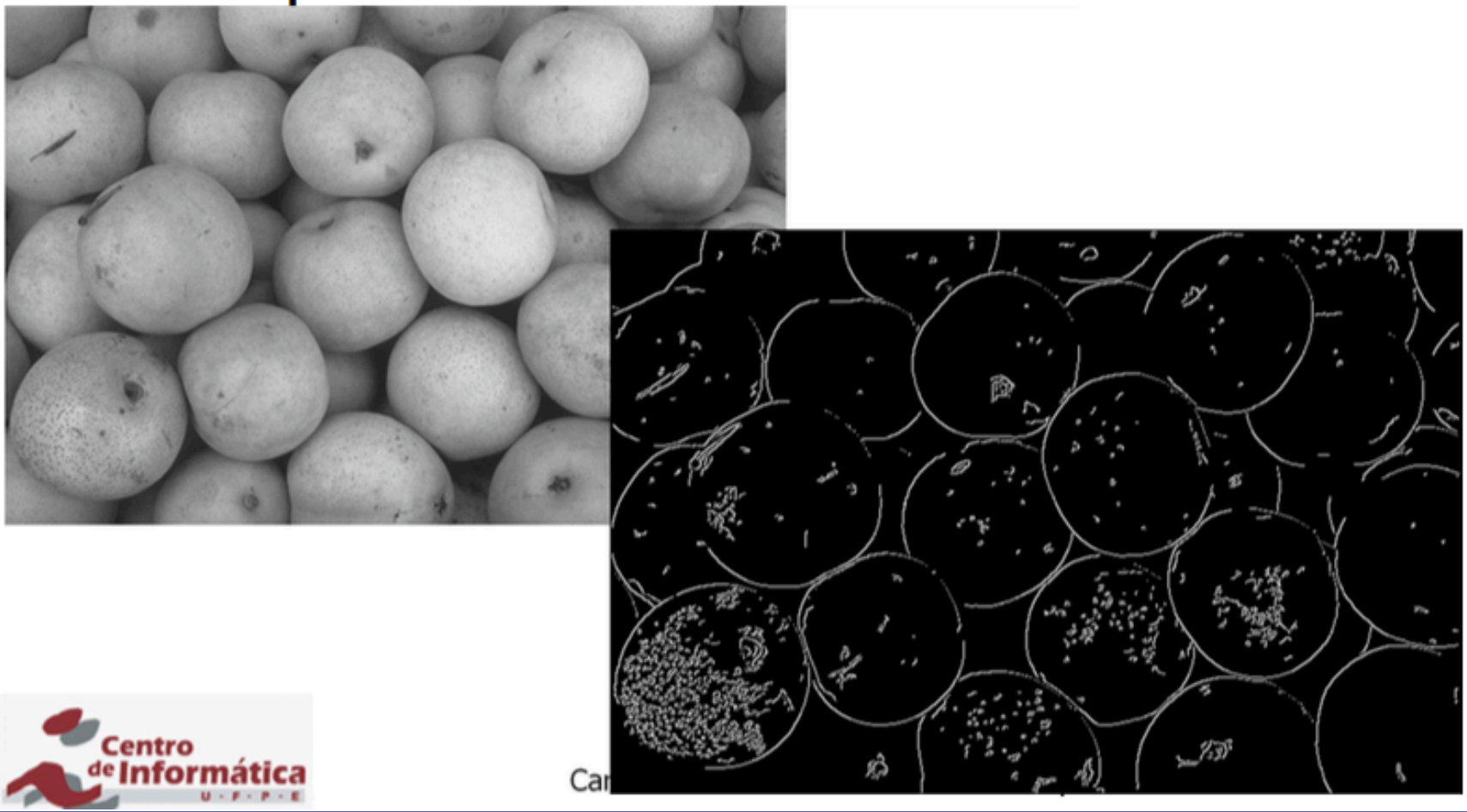
- Exemplo:

Gaussiana: 5x5, $\sigma = 0.5$
Thresholding:
Th_{baixo} = 10% do máximo
Th_{alto} = 20% do máximo



Detector de Canny

- Exemplo:



Centro de Informática U·F·P·E

Detector de Borda

Detector de Borda

Prewitt

Detector de Borda

Roberts

Detector de Borda

Sobel

Detector de Borda

Canny

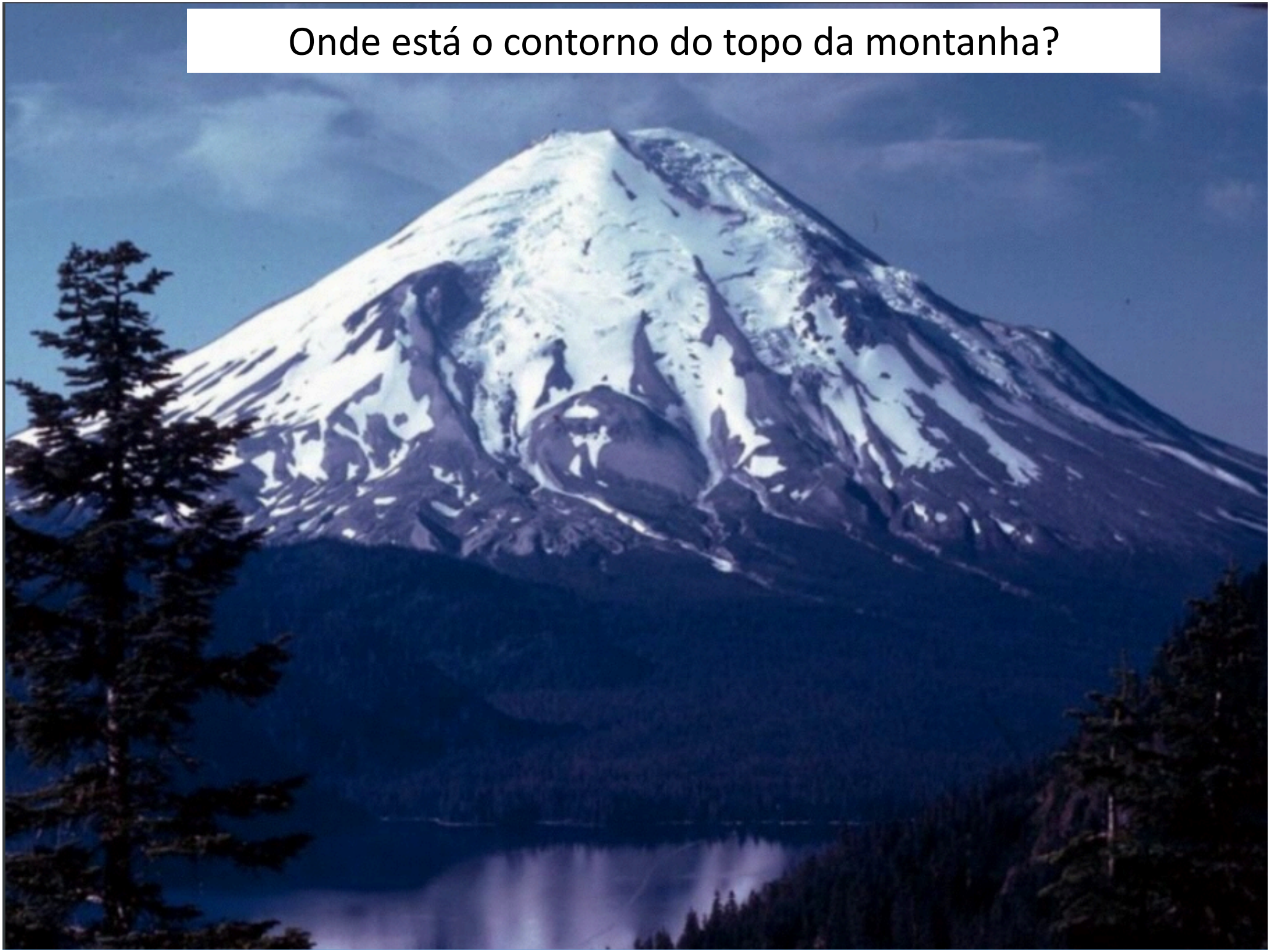
Detector de Borda

DoG

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

Extraindo Linhas

Onde está o contorno do topo da montanha?



Linhas são difíceis de encontrar

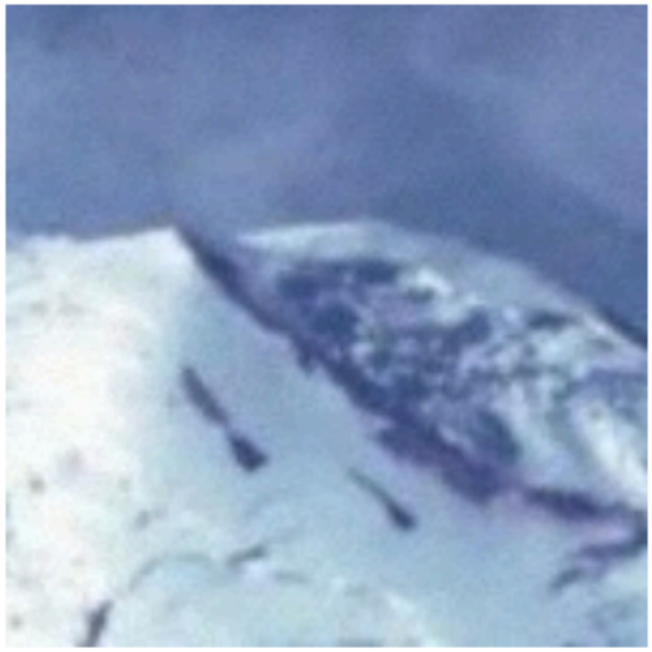
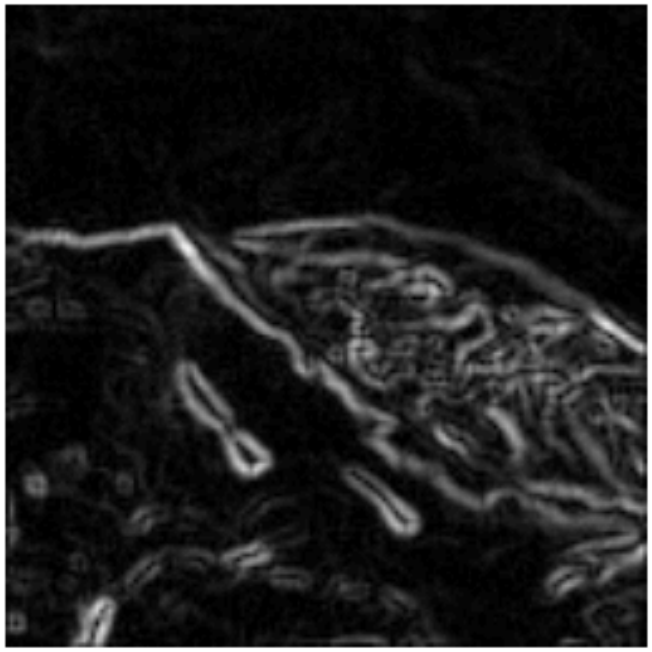
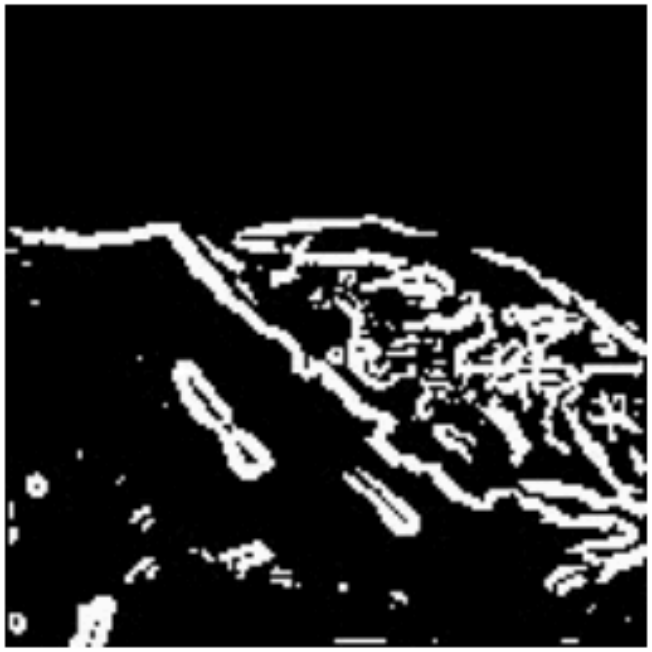


Imagem Original



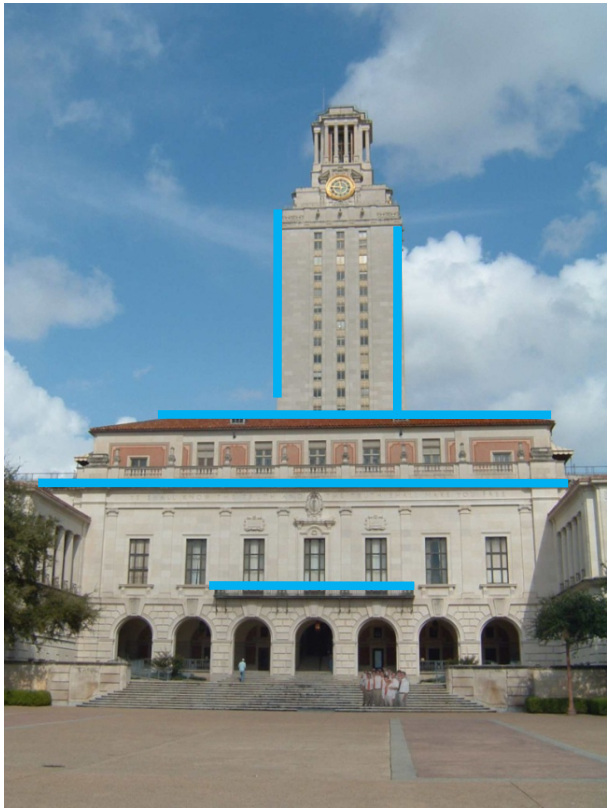
Detector de Borda

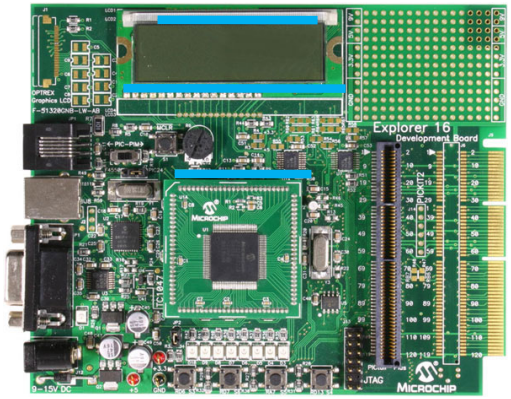


Thresholding

Encontrando linhas

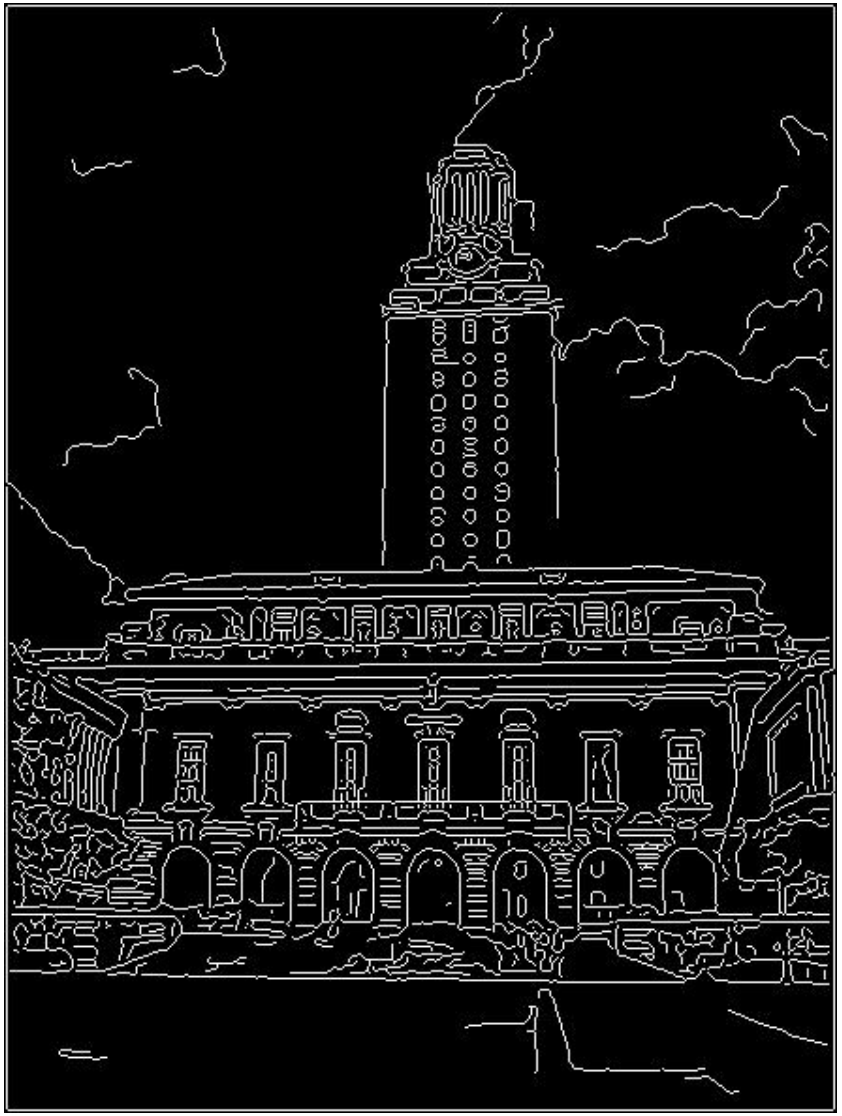
- Por que encontrar linhas?
  - Muito objetos são caracterizados pela presença de linhas



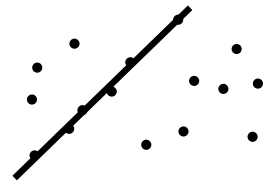


- Por que não são feitas utilizando detectores de borda?

Dificuldades de encontrar linahs

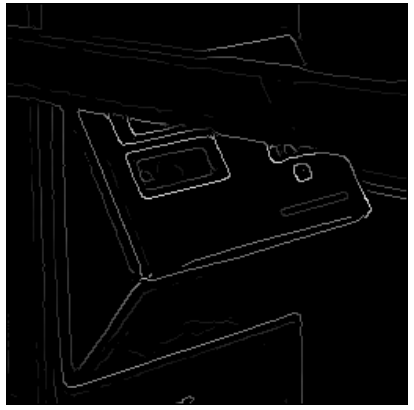


- **Muitos pontos de borda:**
  - múltiplos modelos
  - Quais pontos em quais linhas
- Algumas partes da linha faltando
- **Ruído**:



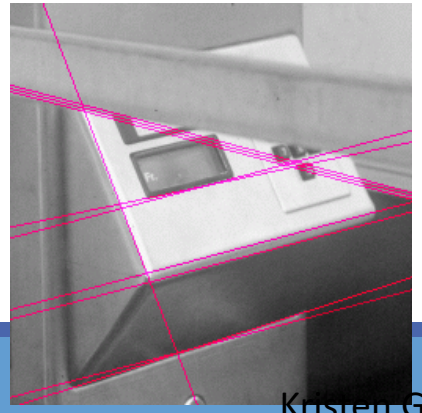
Ajuste de linhas: Transformada de Hough

- Dado pontos que pertencem à uma linha, qual a linha?
- Quantas linhas existem?
- Quais pontos pertencem a quais linhas?



- **Hough Transform** é uma técnica de votação
  - Ideia:

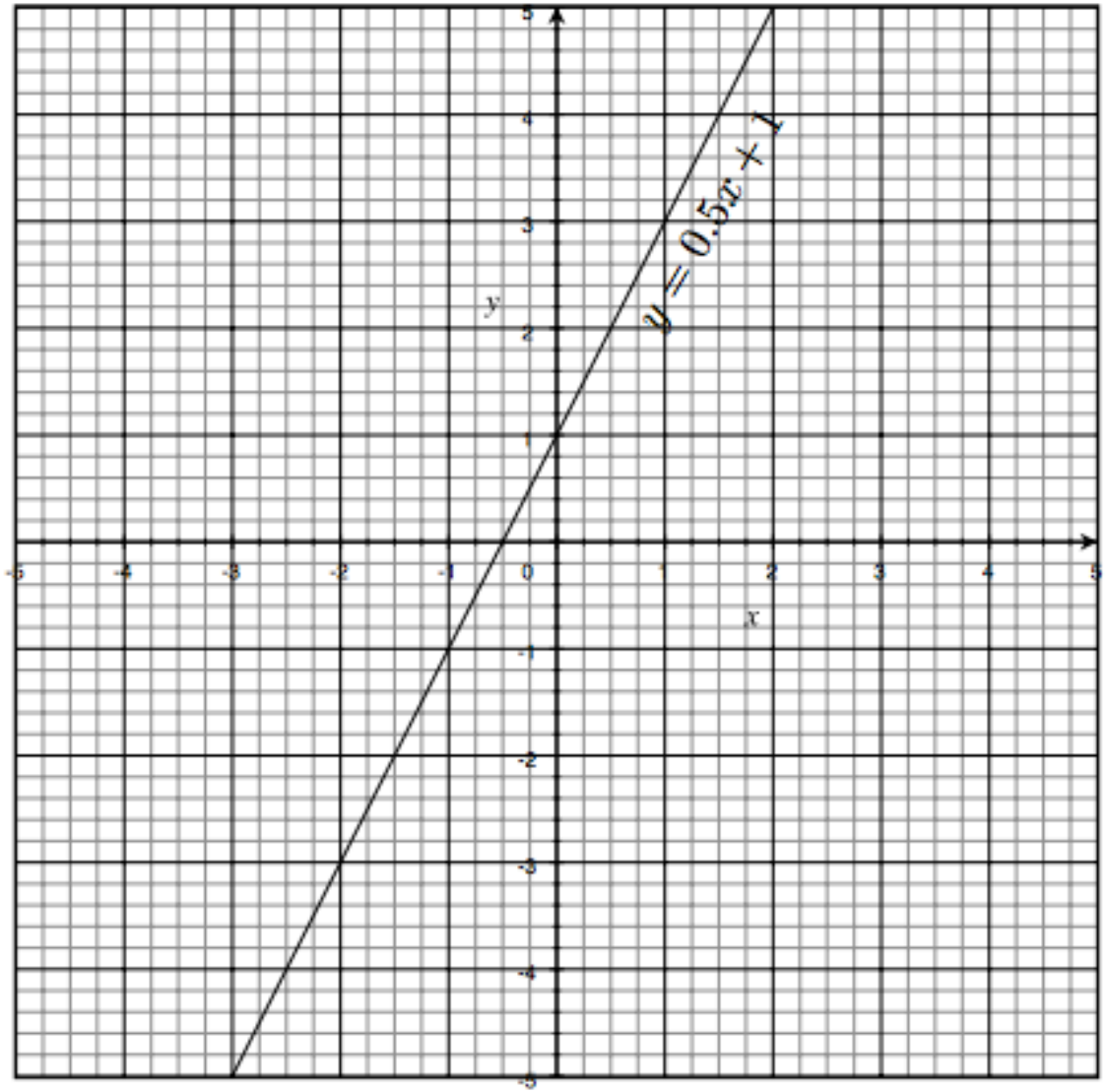
1. Armazena voto para cada linha possível onde cada ponto da borda está.
2. Procura por linhas que possuem muitos votos.



Slope intercep form

$$y = mx + b$$

slope y-intercept



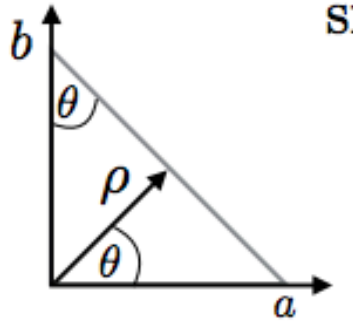
Normal Form

$$x\cos\theta + y\sin\theta = \rho$$

Derivation:

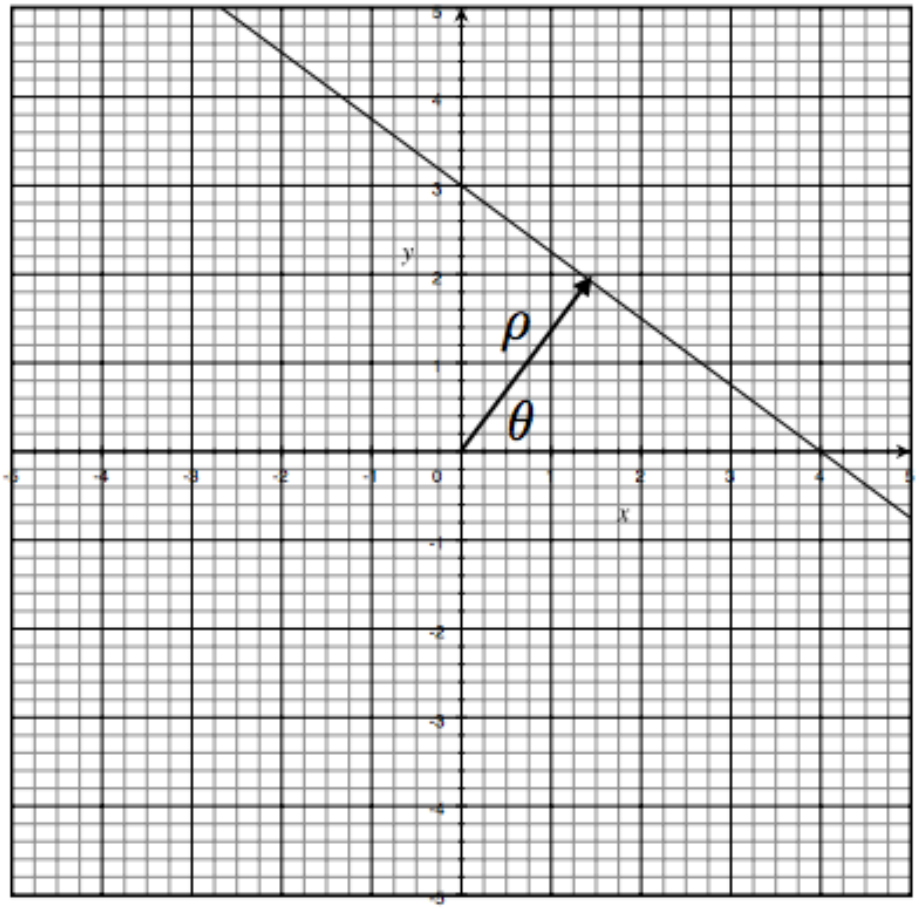
$$\cos\theta = \frac{\rho}{a} \rightarrow a = \frac{\rho}{\cos\theta}$$

$$\sin\theta = \frac{\rho}{b} \rightarrow b = \frac{\rho}{\sin\theta}$$



plug into: $\frac{x}{a} + \frac{y}{b} = 1$

$$x\cos\theta + y\sin\theta = \rho$$



Transformada de Hough

- Framework genérico para detecção de forma/objeto
- Bordas não precisam estar conectadas
- Linhas podem estar omitidas
- Ideia: votação de bordas para possíveis modelos

Imagem e Espaço de Parâmetros

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

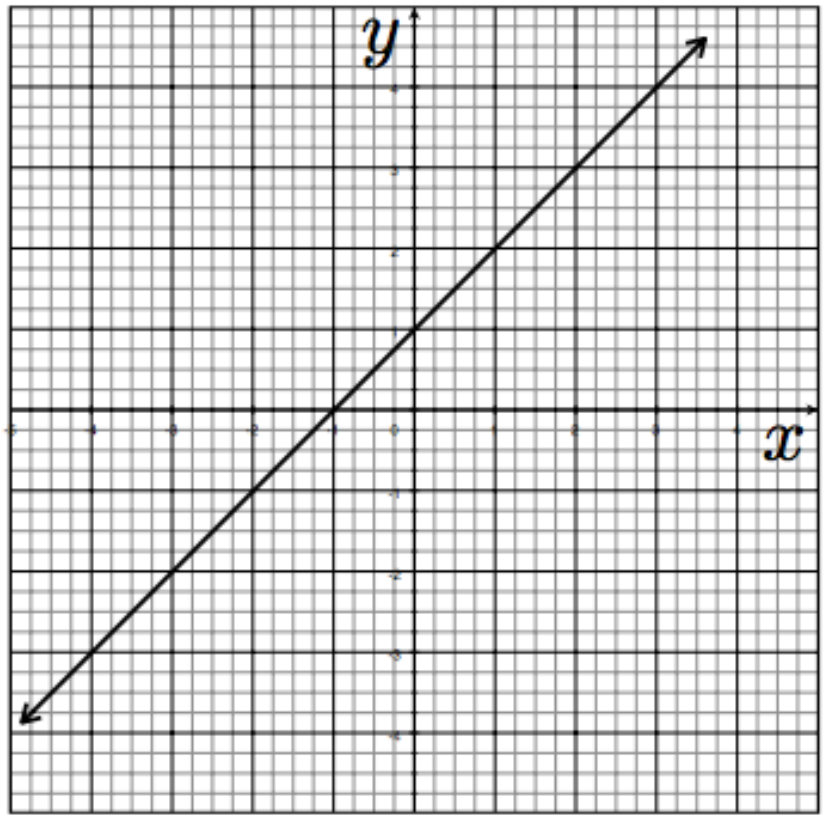
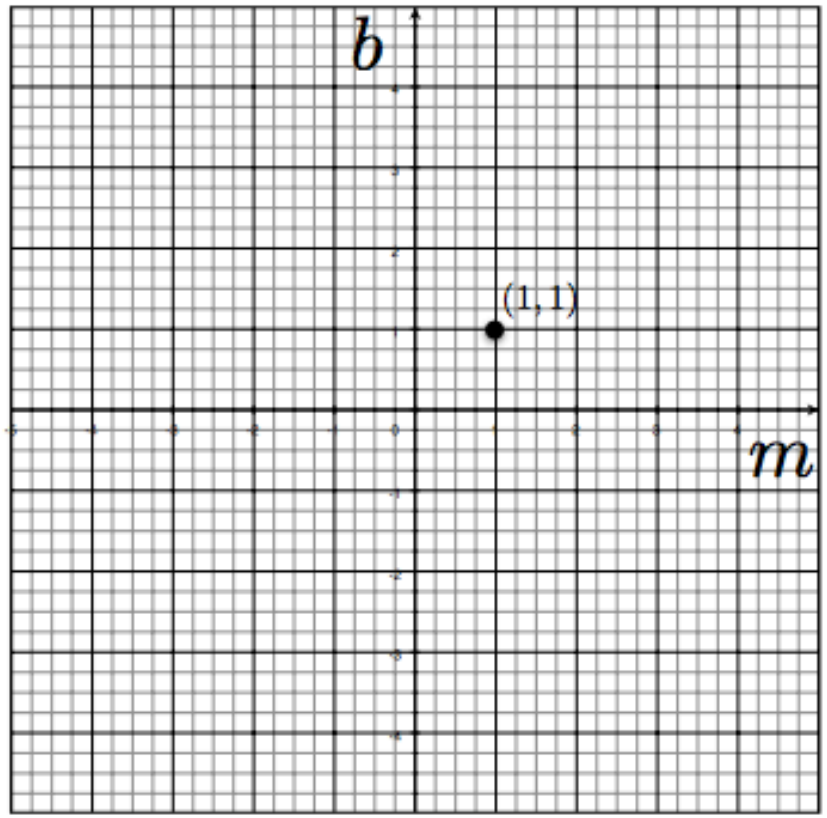


Image space

a line becomes a point



Parameter space

variables

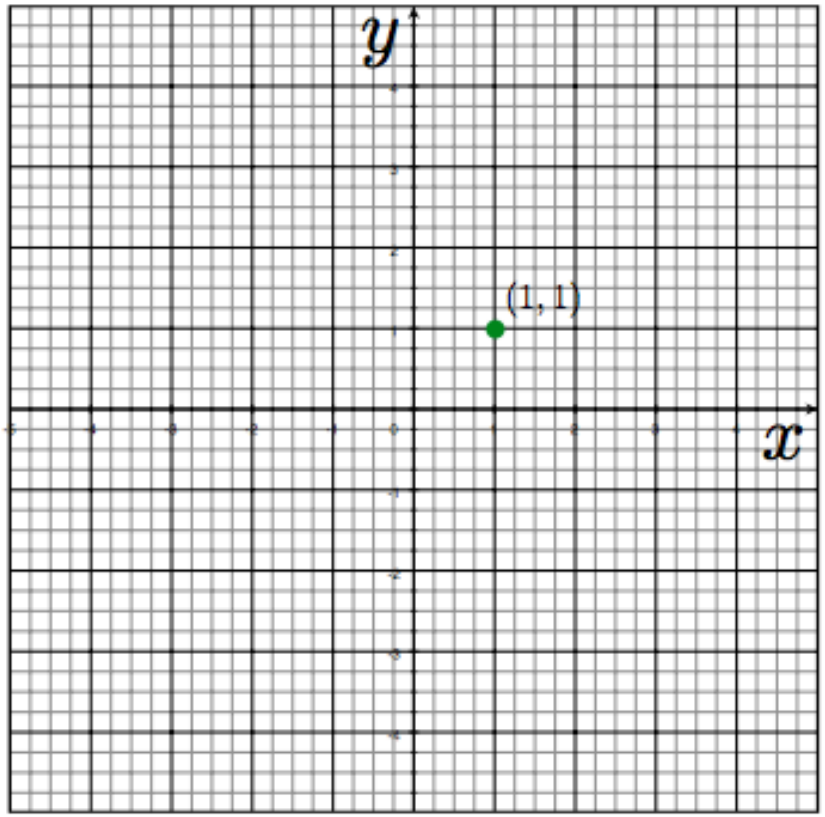
$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters



a point becomes a line

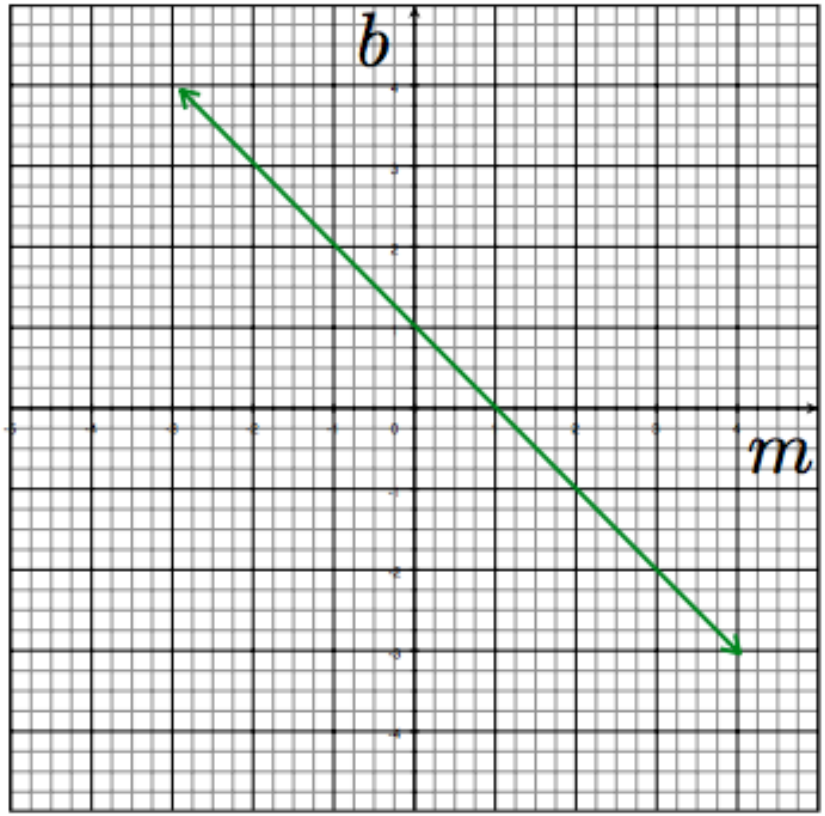


Image space

Parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

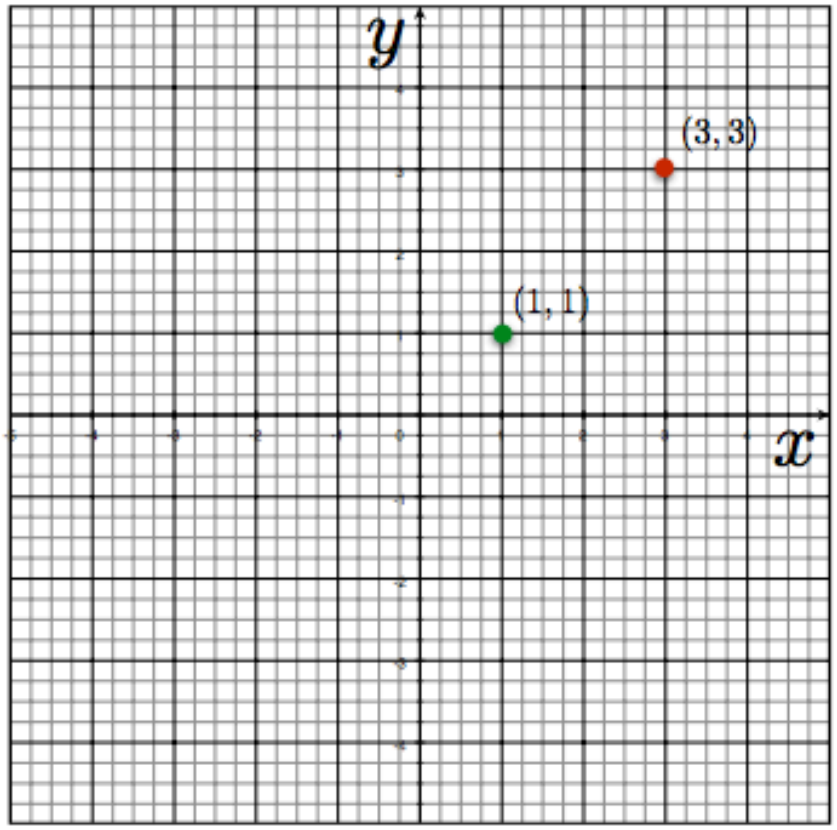
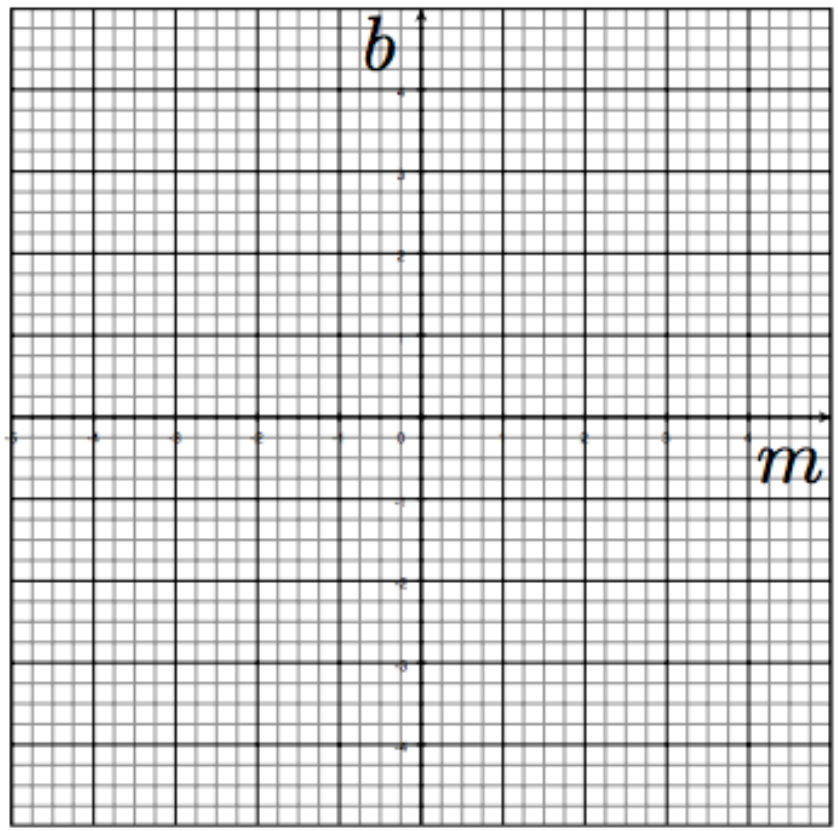


Image space

two points become ?



Parameter space

variables

$$y = mx + b$$

parameters

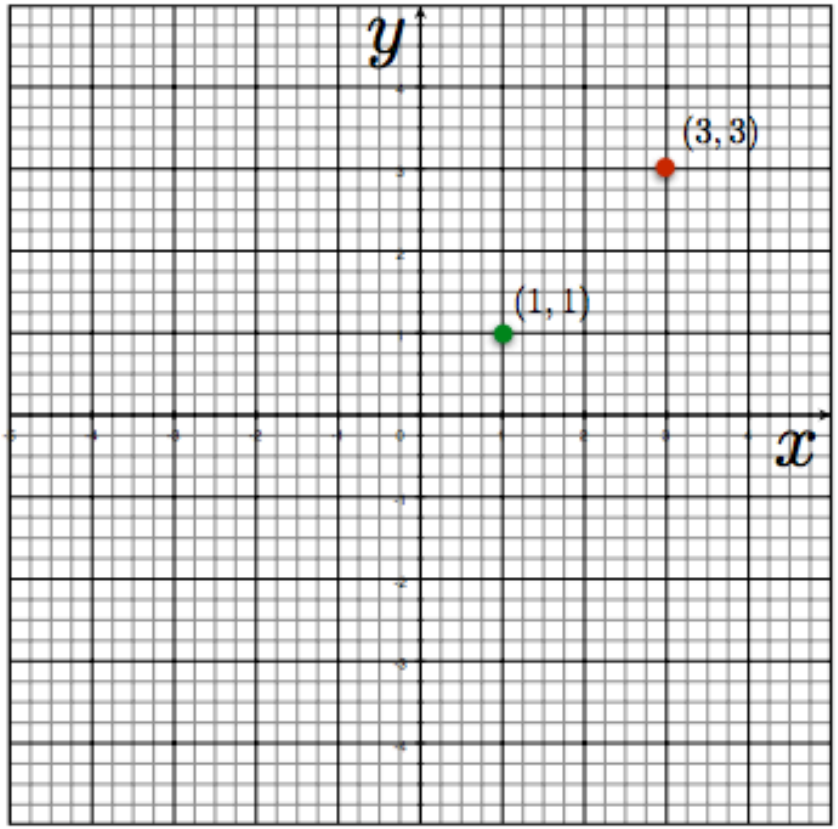


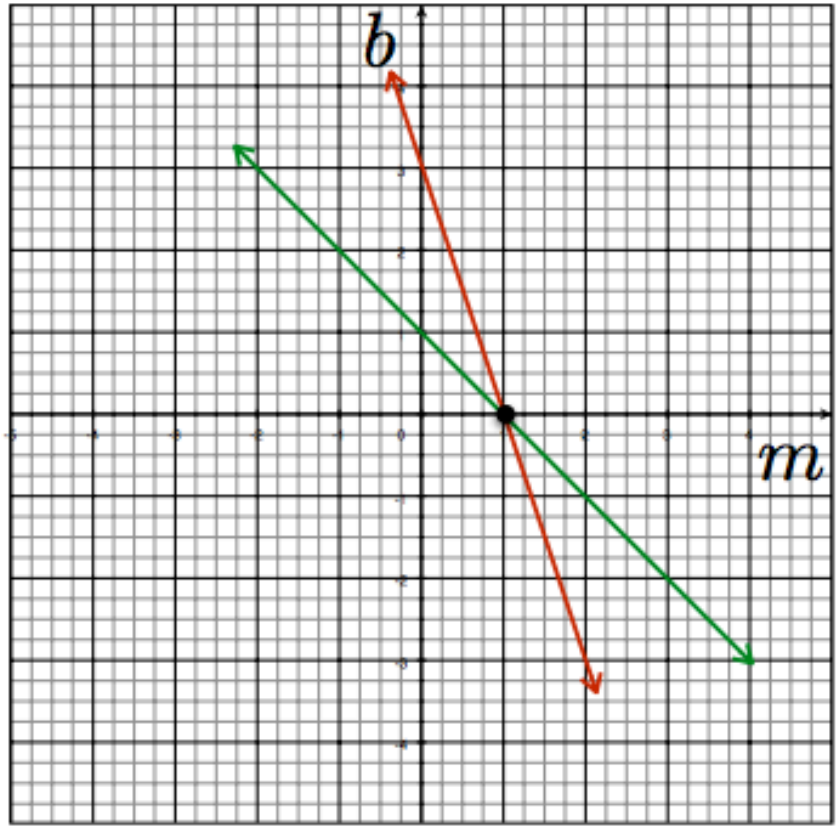
Image space

two points become ?

variables

$$y - mx = b$$

parameters



Parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

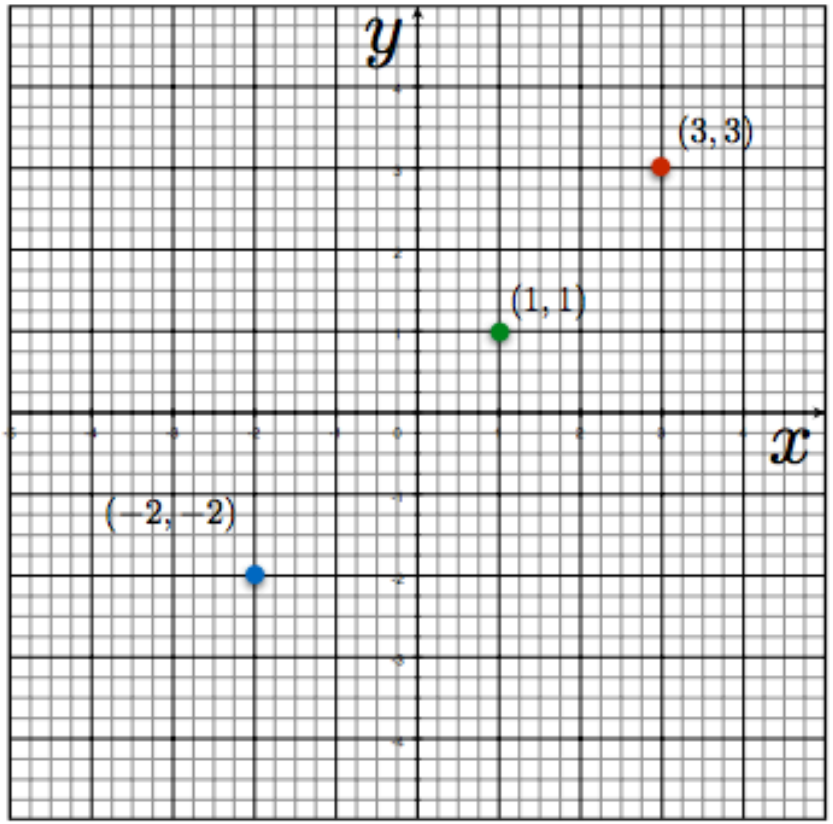
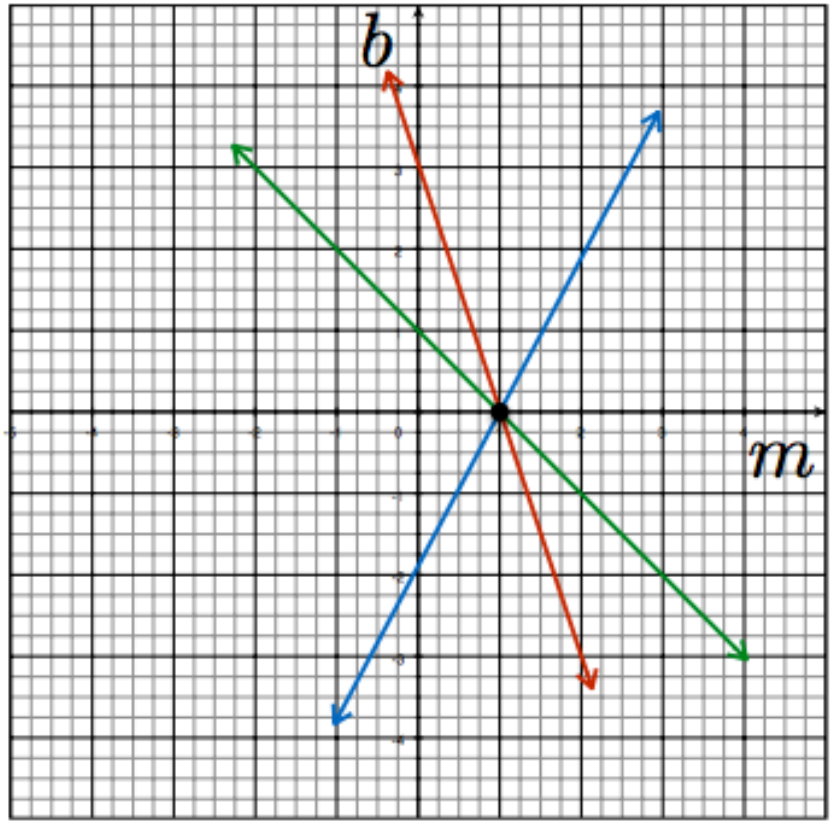


Image space

three points become ?



Parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

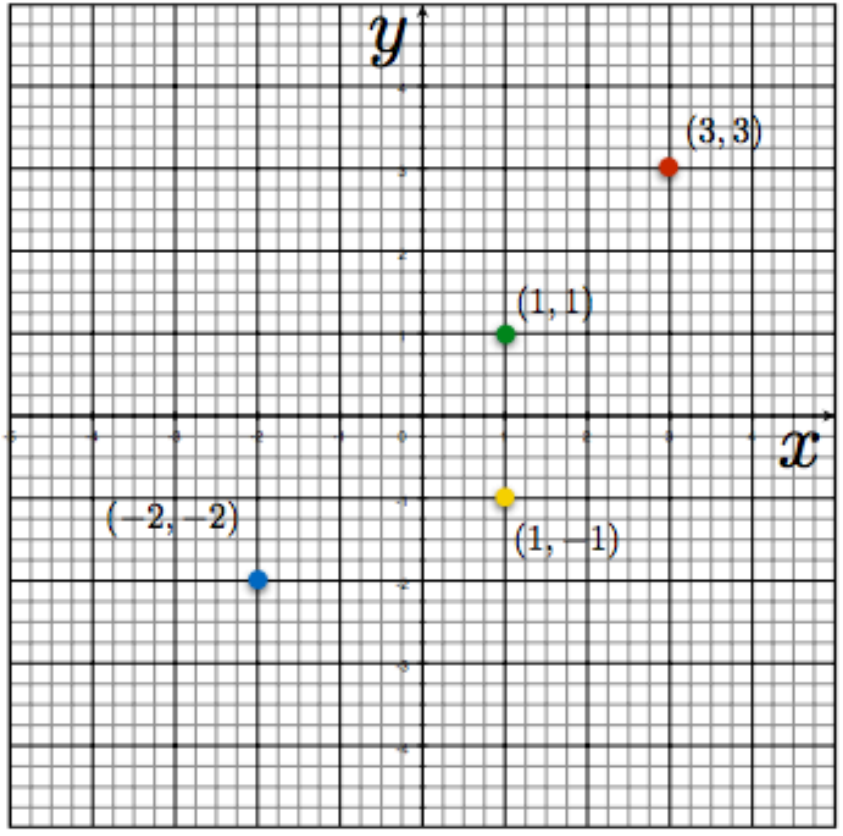
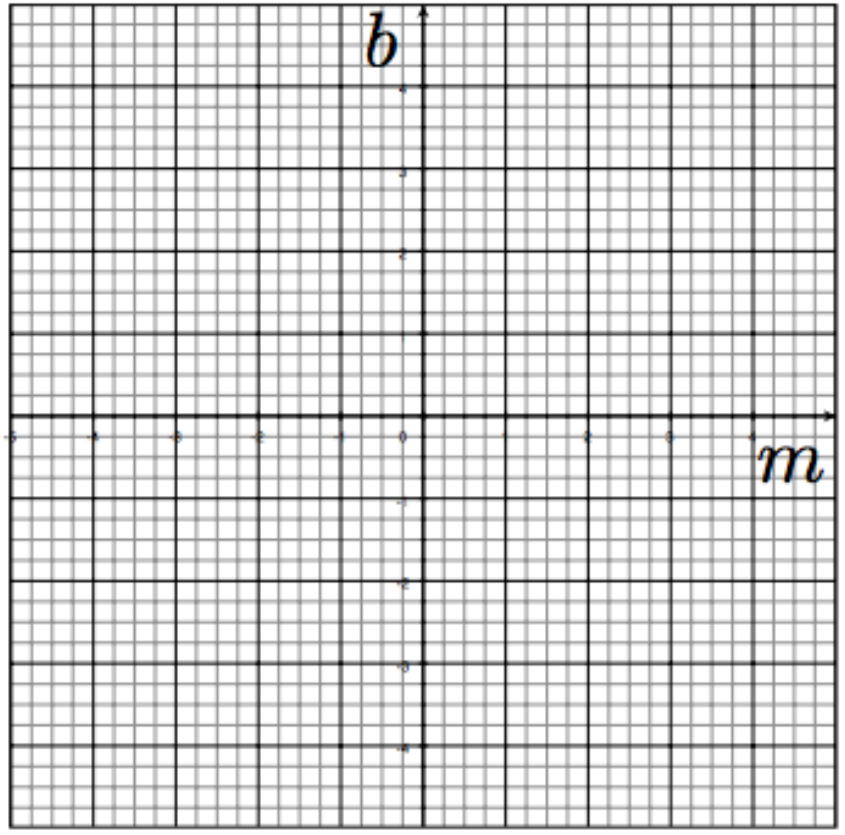


Image space

four points become ?



Parameter space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

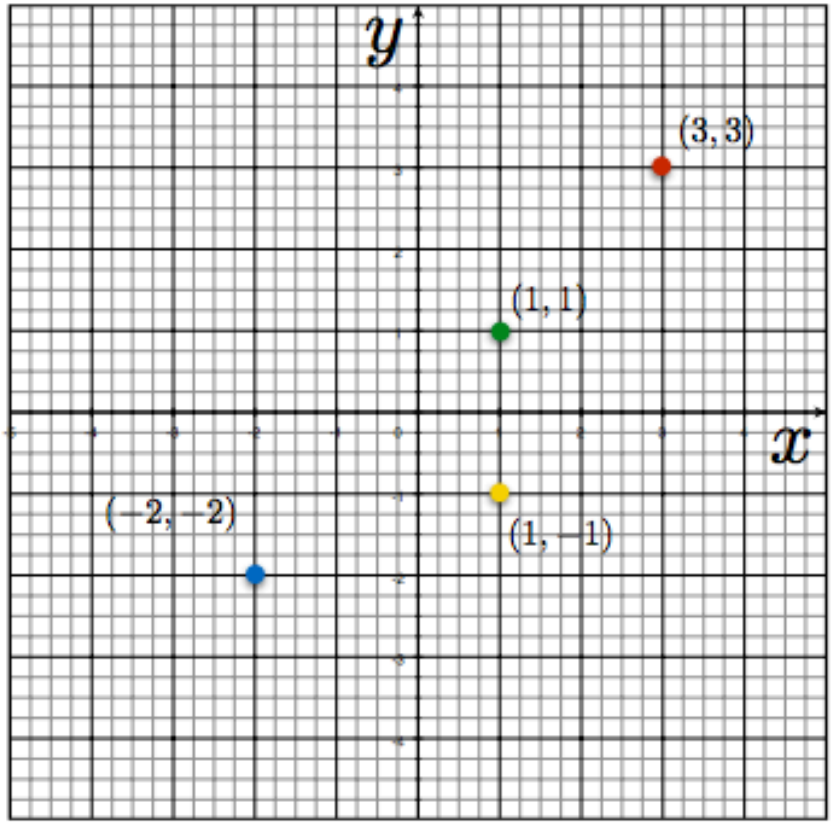
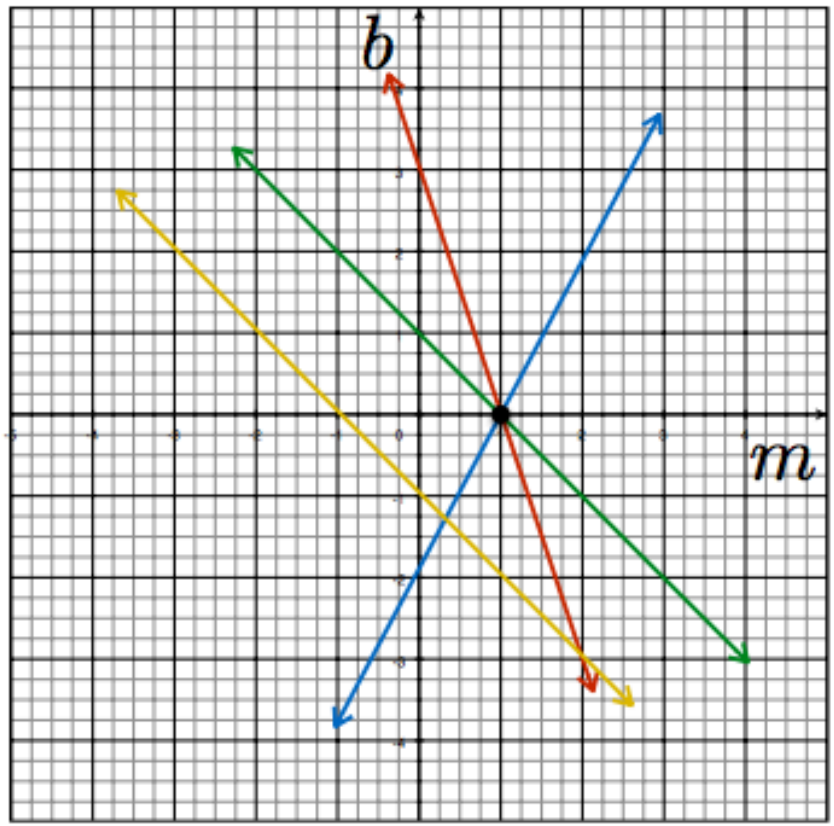


Image space

four points become ?



Parameter space

Detecção de linha pela transformada de Hough

Algorithm:

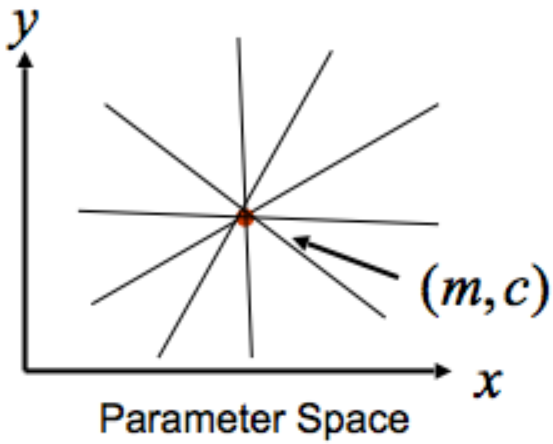
1. Quantize Parameter Space (m,c)

2. Create Accumulator Array $A(m,c)$

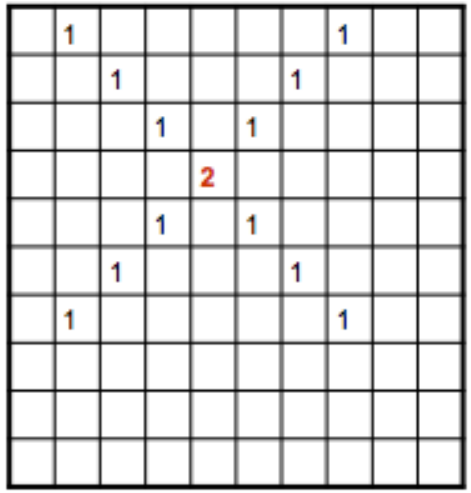
3. Set $A(m,c)=0 \quad \forall m,c$

4. For each image edge (x_i,y_i)
   For each element in $A(m,c)$
   If (m,c) lies on the line: $c=-x_i m+y_i$
   Increment $A(m,c)=A(m,c)+1$

5. Find local maxima in $A(m,c)$



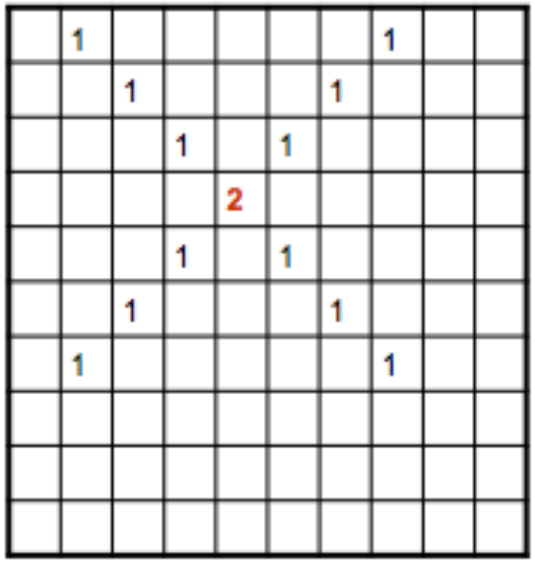
$A(m,c)$



Problemas com parametrização

- O que há de errado com a parametrização (m,c)?
- Quão grande o acumulador precisa ser?

$A(m,c)$



O espaço de m é muito grande!

O espaço de c é muito grande!

Melhor Parametrização

Use normal form:

$$x \cos\theta + y \sin\theta = \rho$$

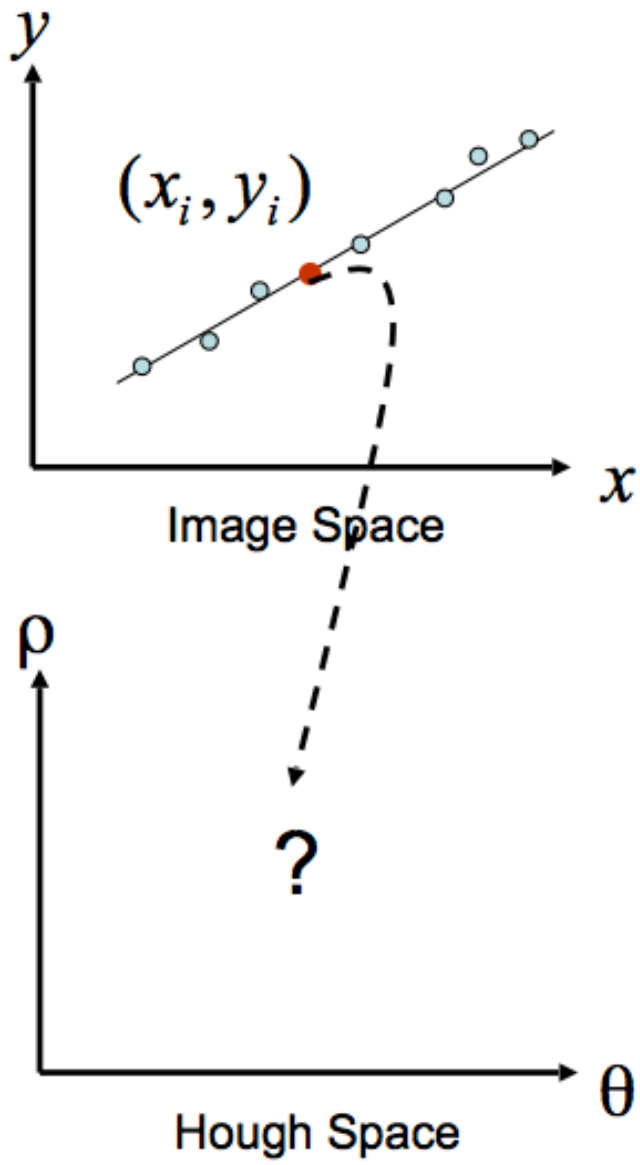
Given points (x_i, y_i) find (ρ, θ)

Hough Space Sinusoid

$$0 \leq \theta \leq 2\pi$$

$$0 \leq \rho \leq \rho_{max}$$

(Finite Accumulator Array Size)



Espaços de imagem em parâmetro

variables

$$y = mx + b$$

parameters

parameters

$$x\cos\theta + y\sin\theta = \rho$$

variables

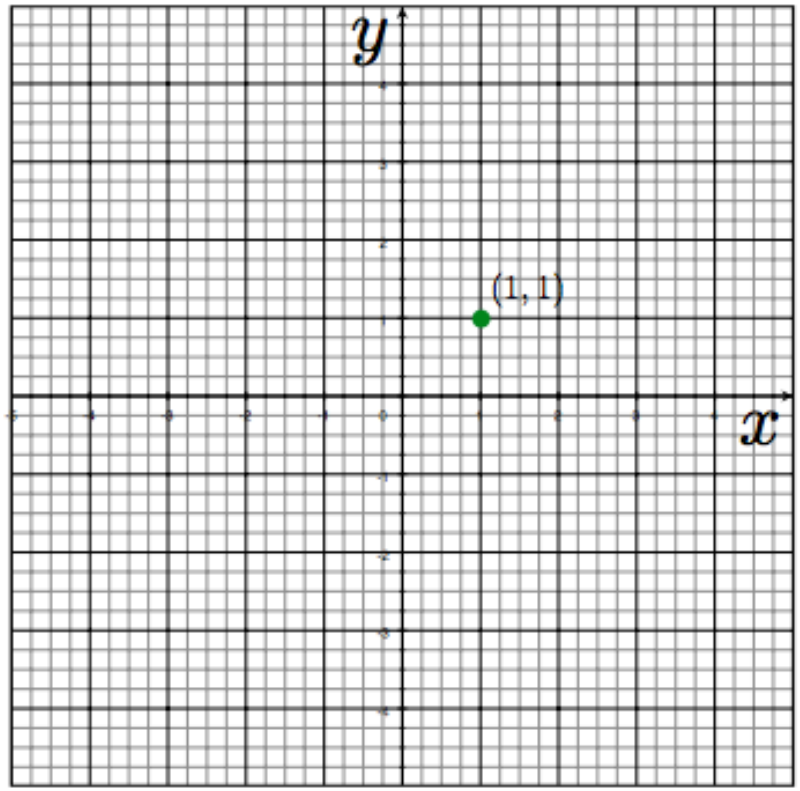


Image space

a point becomes a wave

Parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

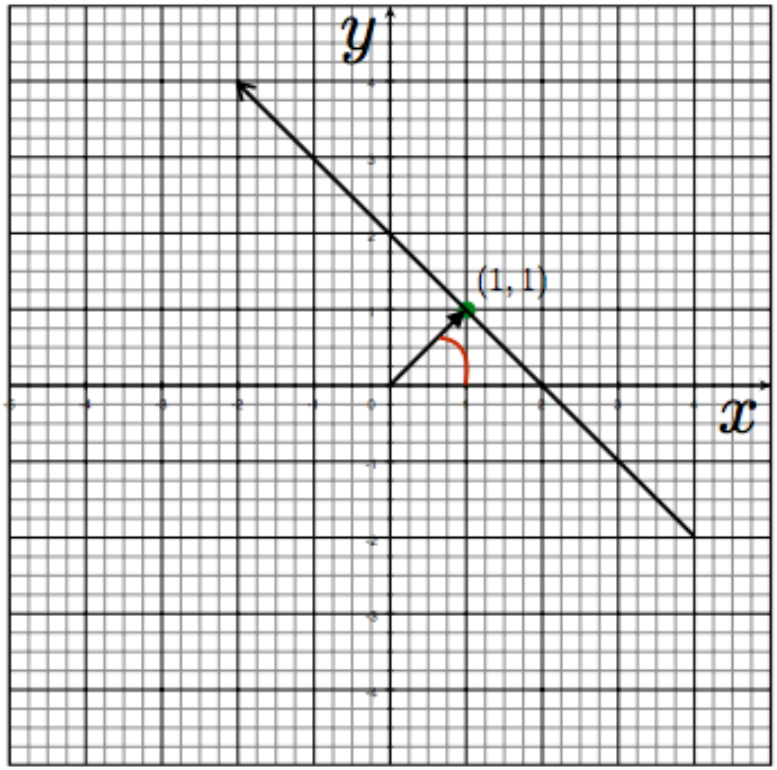
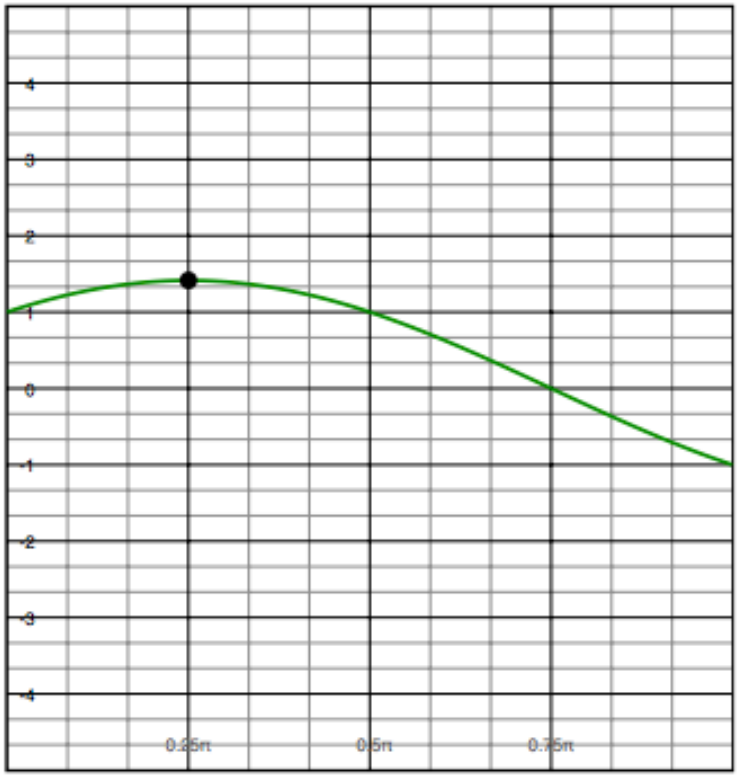


Image space

a line becomes a point



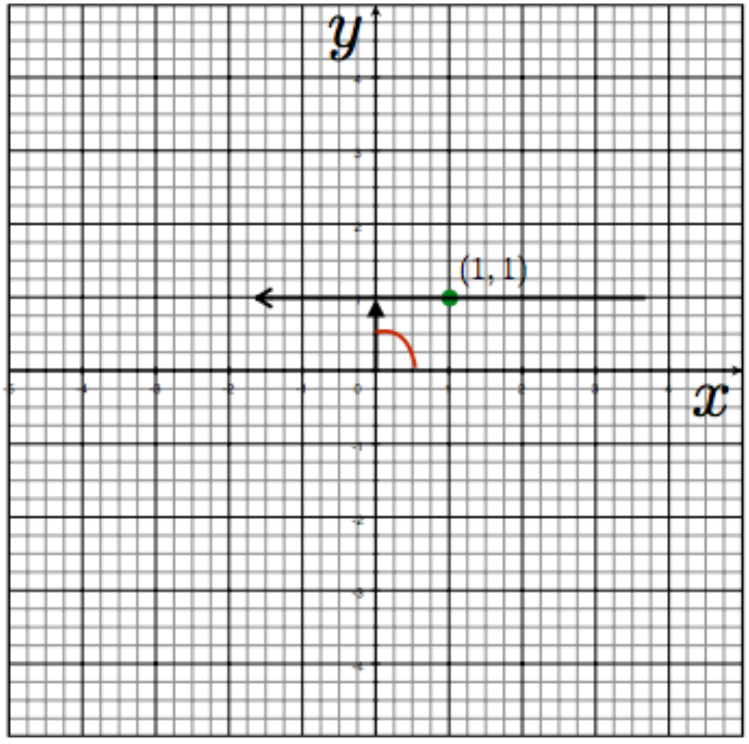
Parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$



a line becomes a point

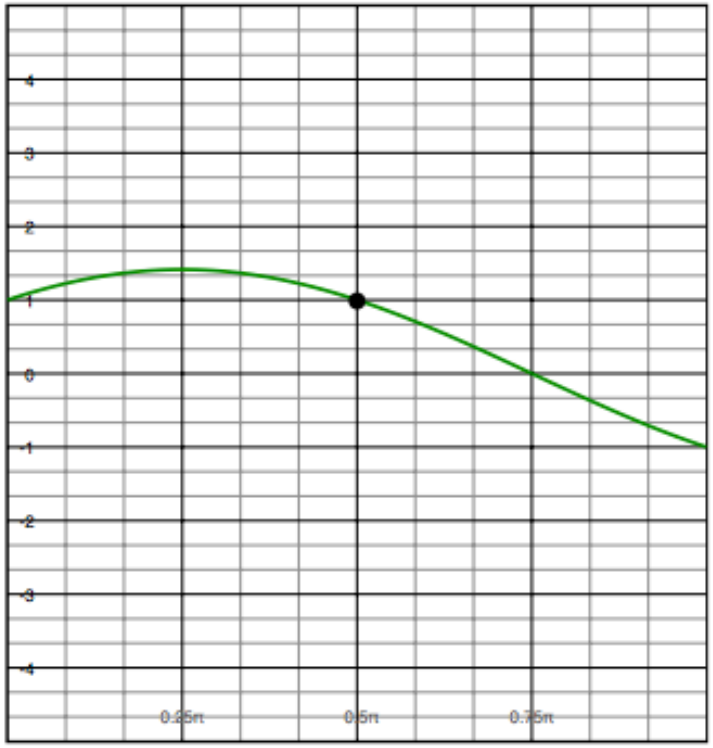


Image space

Parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

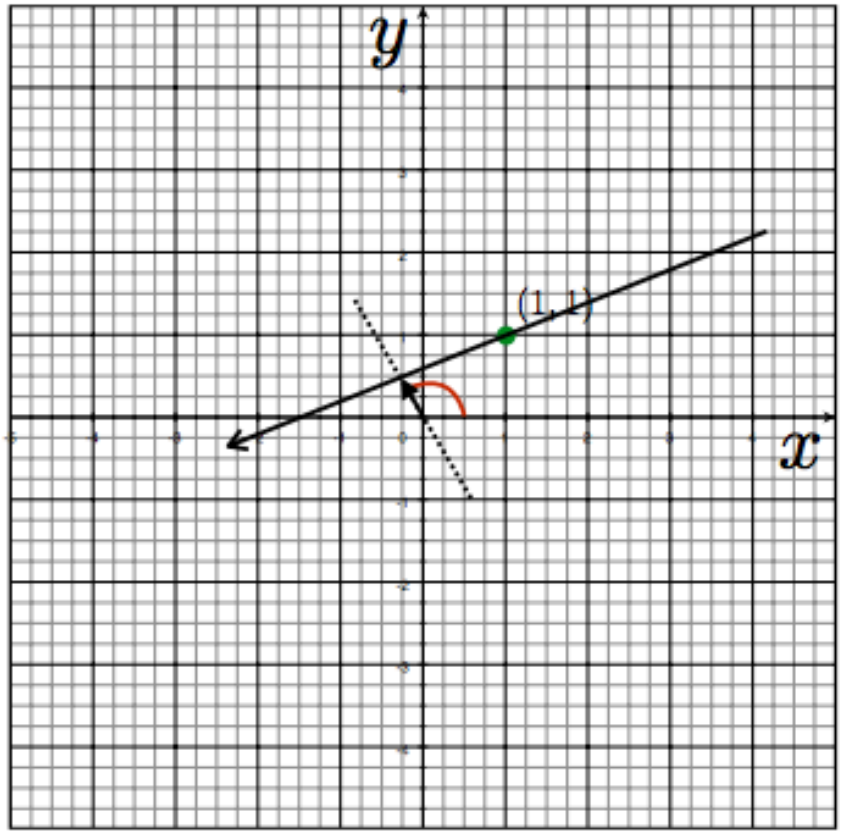
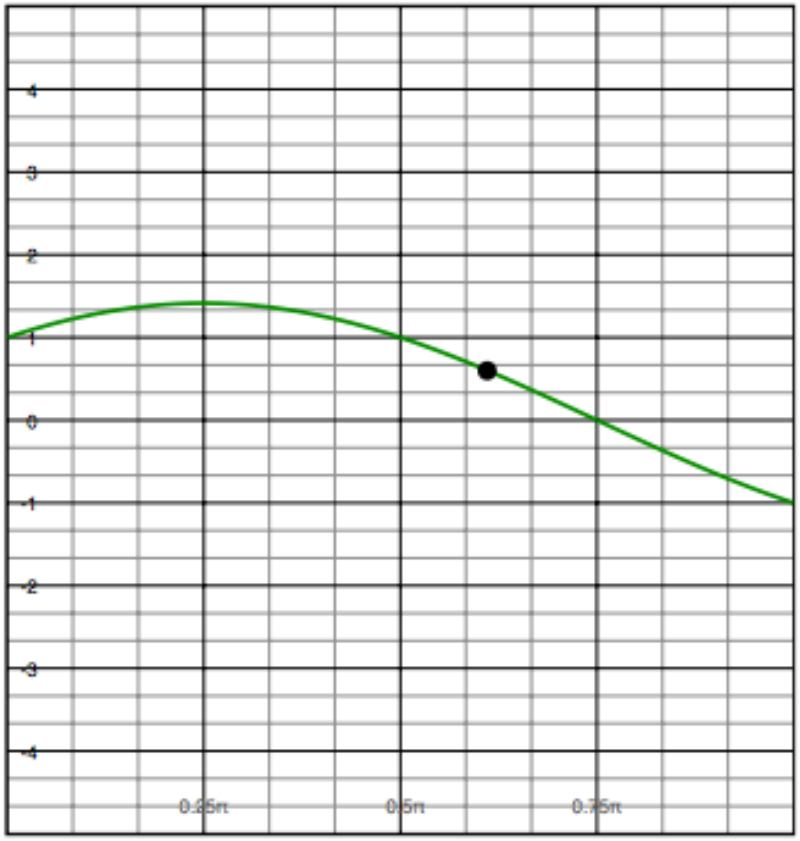


Image space

a line becomes a point



Parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

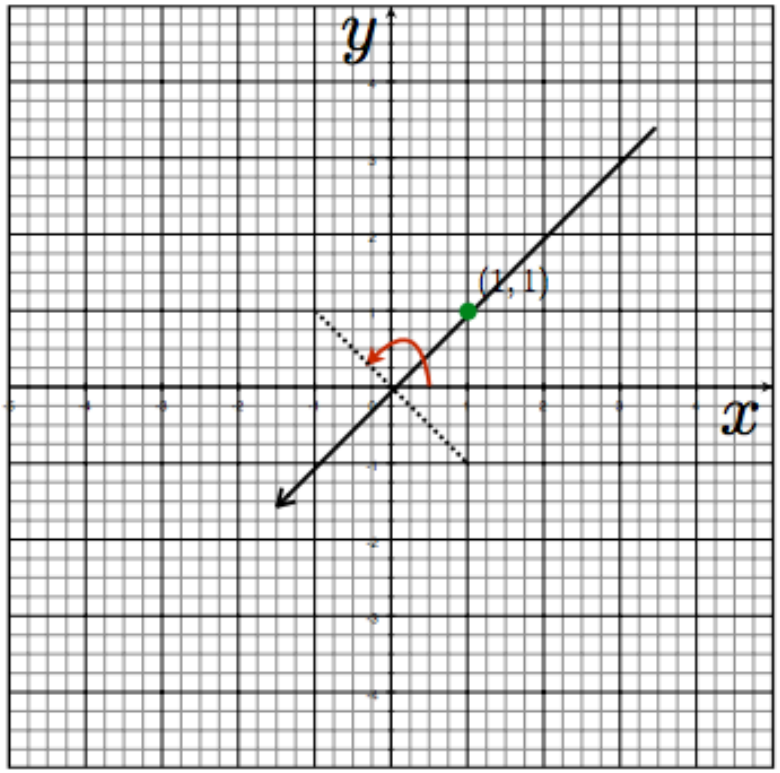
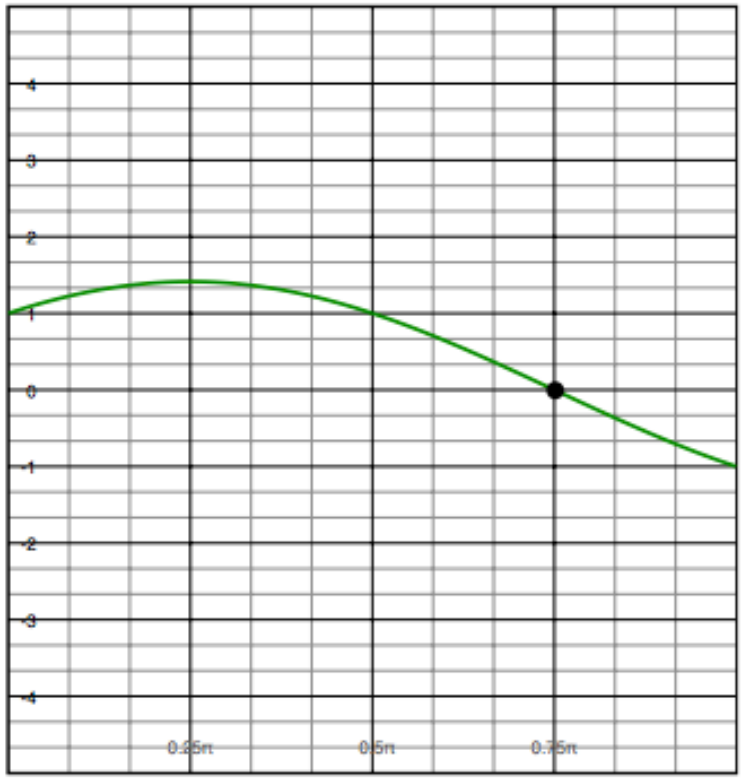


Image space

a line becomes a point



Parameter space

variables

$$y = mx + b$$

parameters

$$x\cos\theta + y\sin\theta = \rho$$

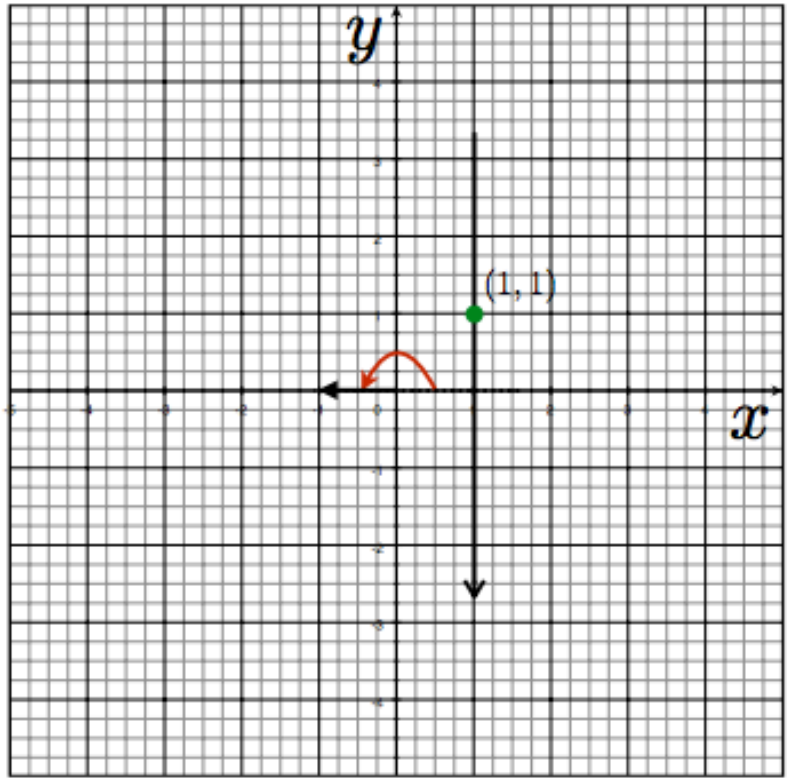
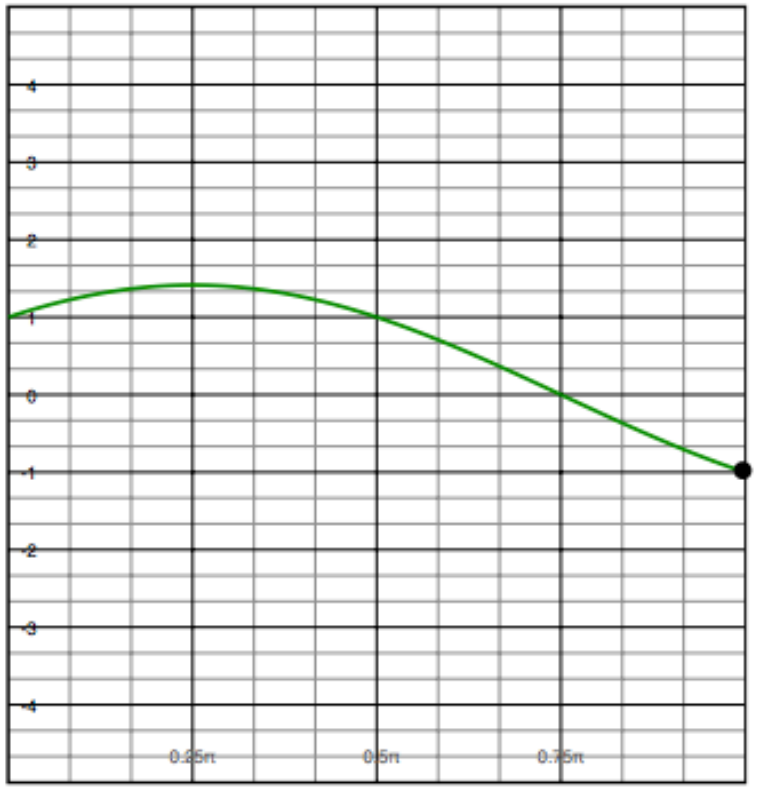


Image space

a line becomes a point



Parameter space

variables

$$y = mx + b$$

parameters

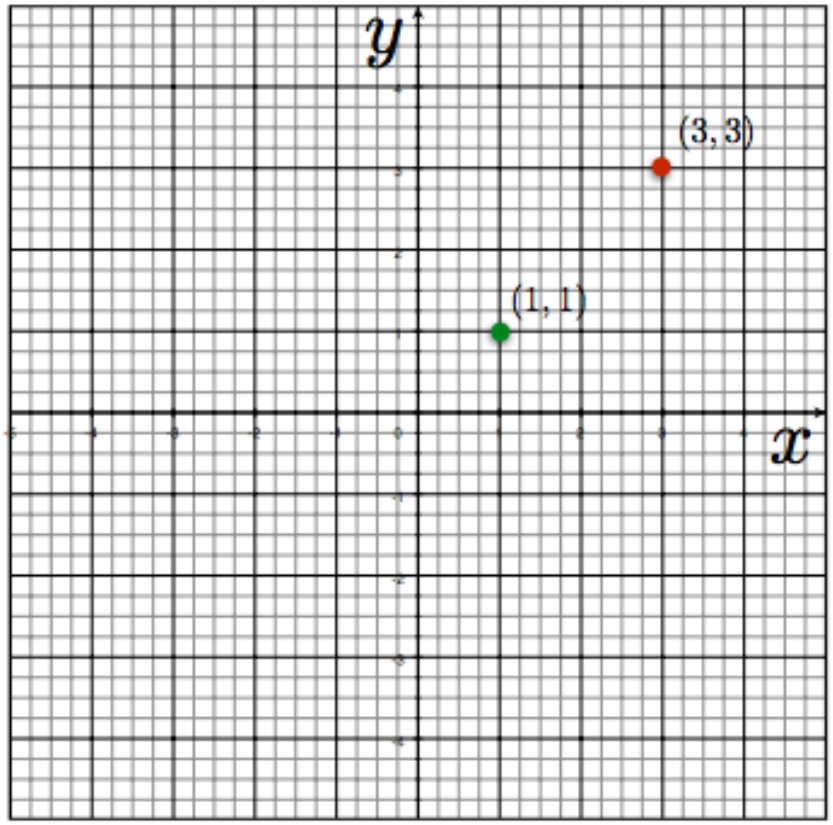
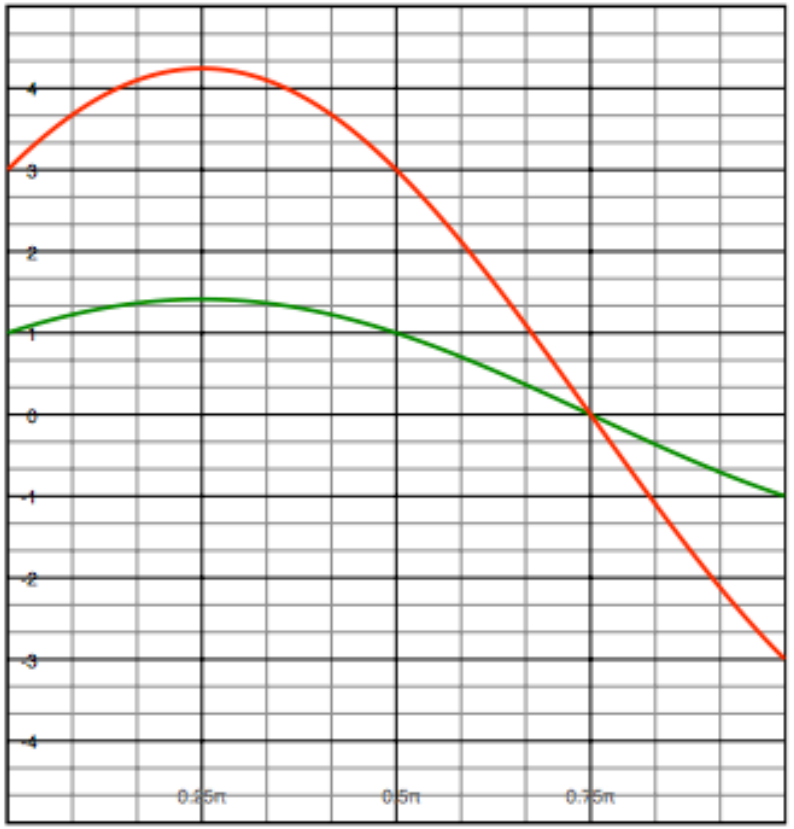


Image space

two points become ?



Parameter space

variables

$$y = mx + b$$

parameters

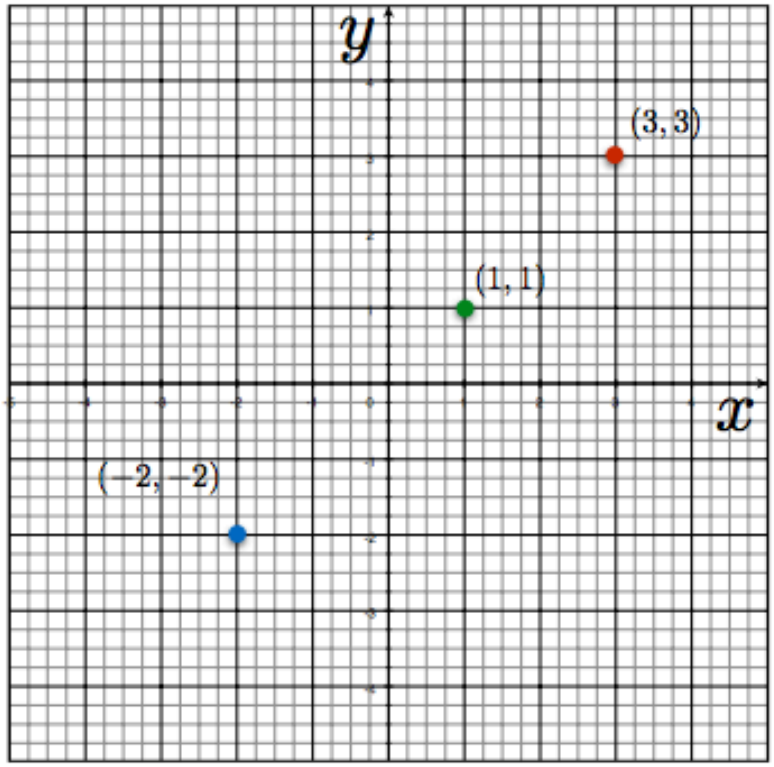
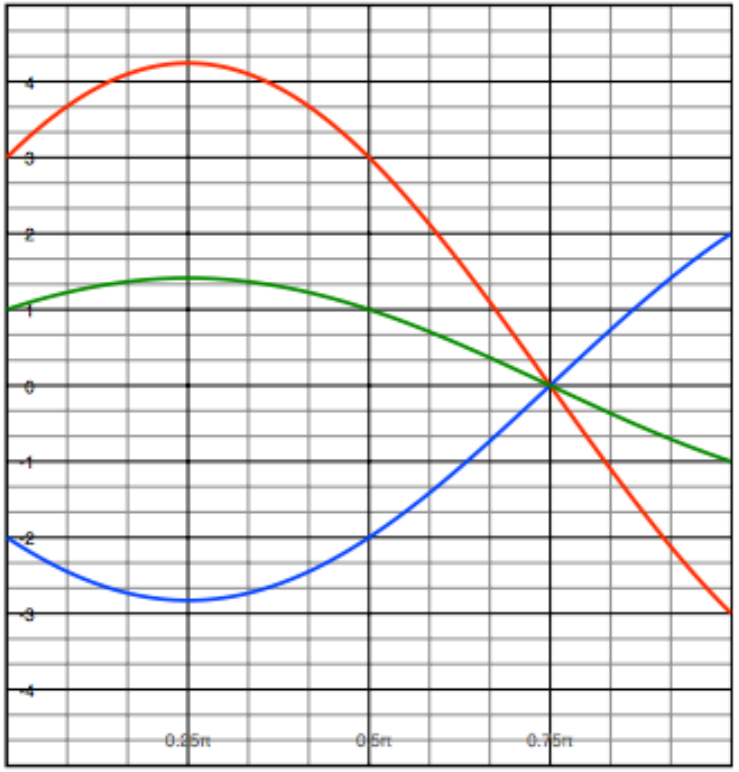


Image space

three points become ?



Parameter space

variables

$$y = mx + b$$

parameters

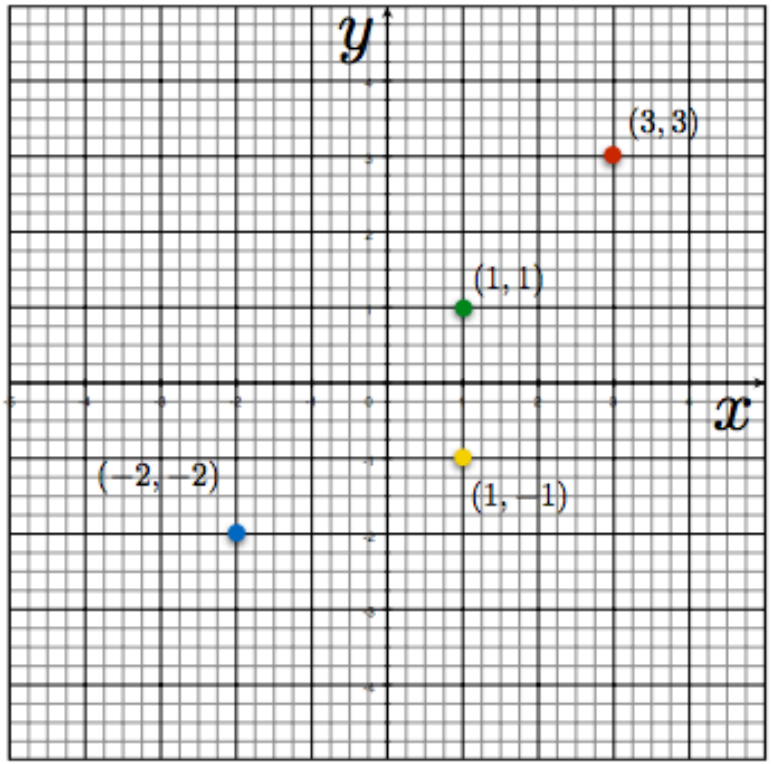
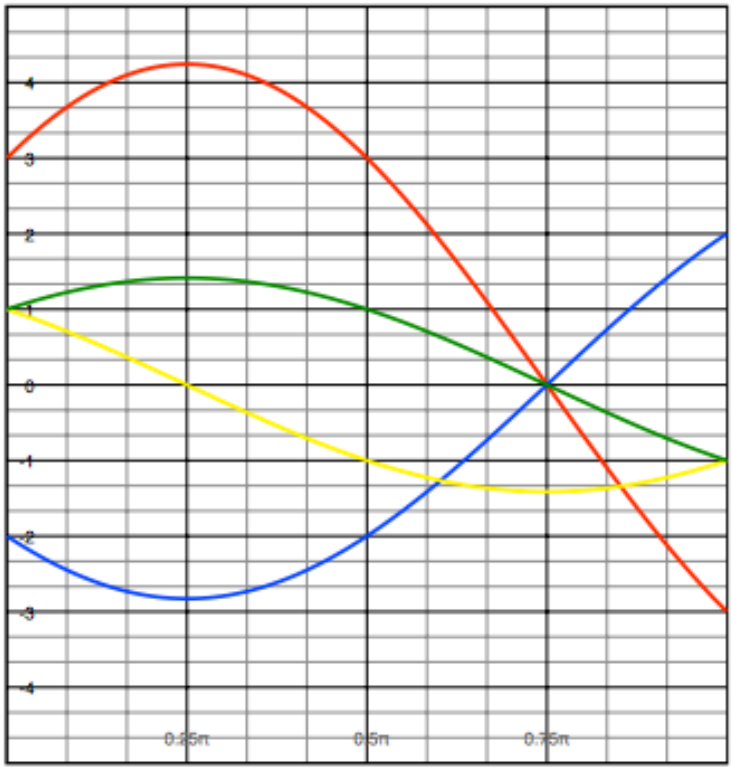


Image space

four points become ?



Parameter space

Demonstração

- http://www.dis.uniroma1.it/~iocchi/slides/icra2001/java/hough.html

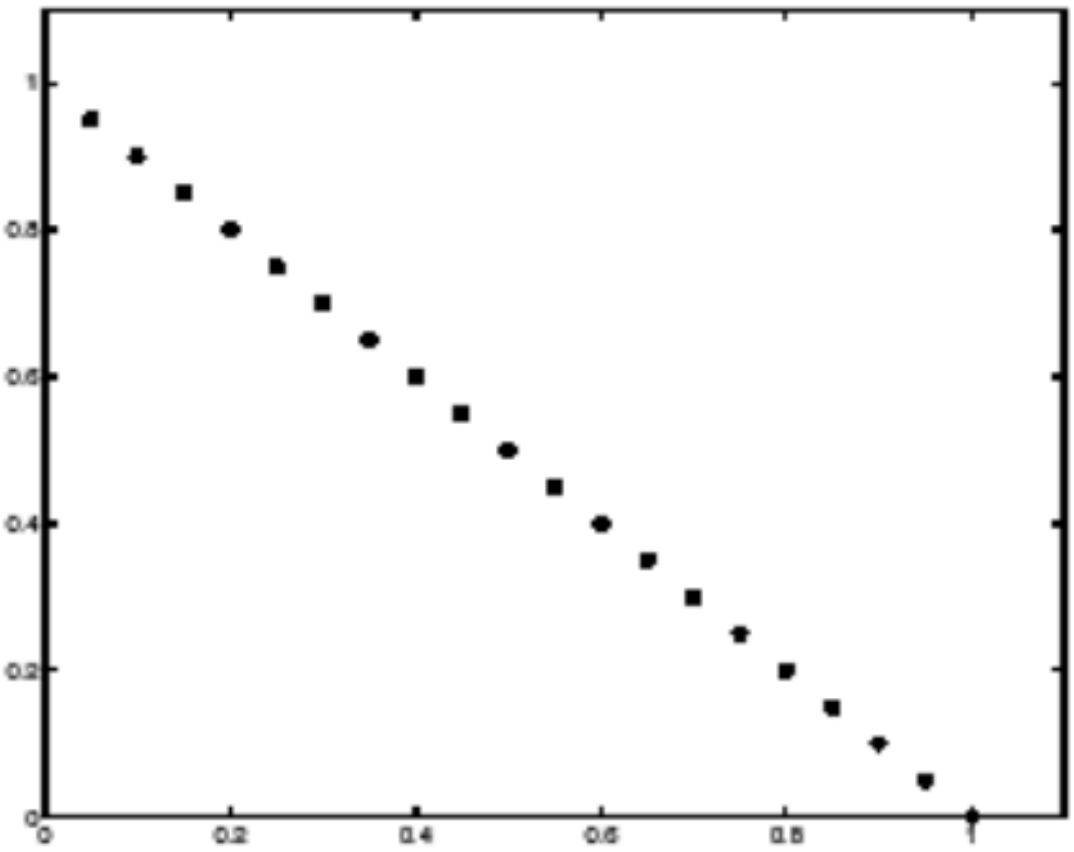
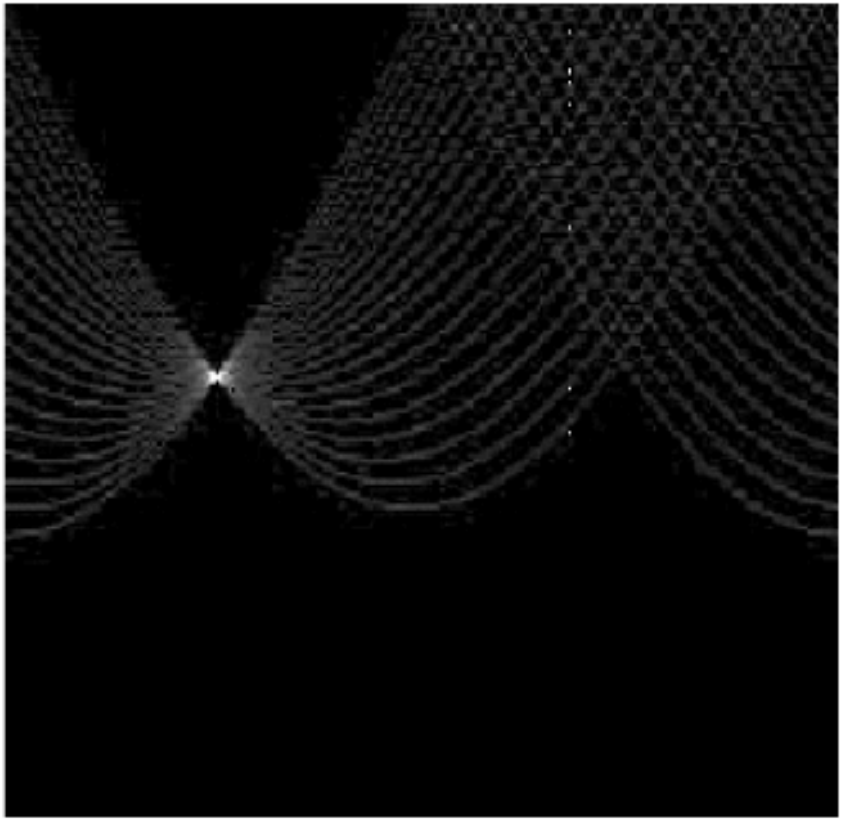
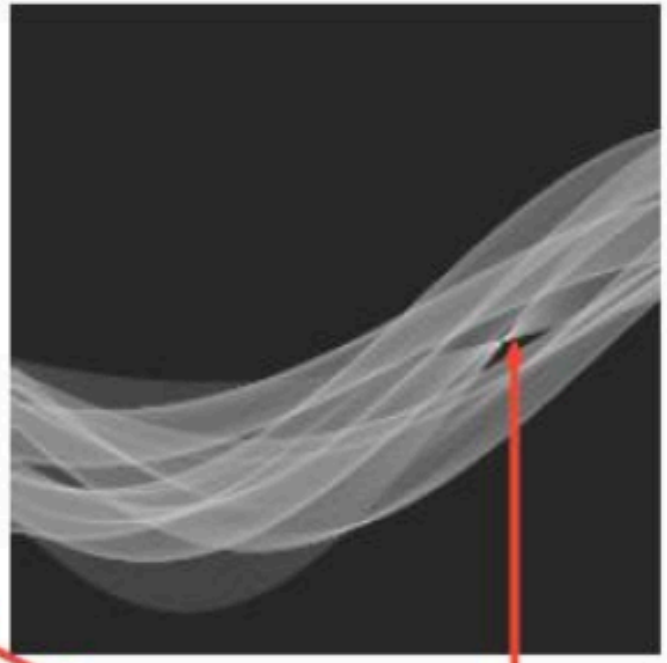
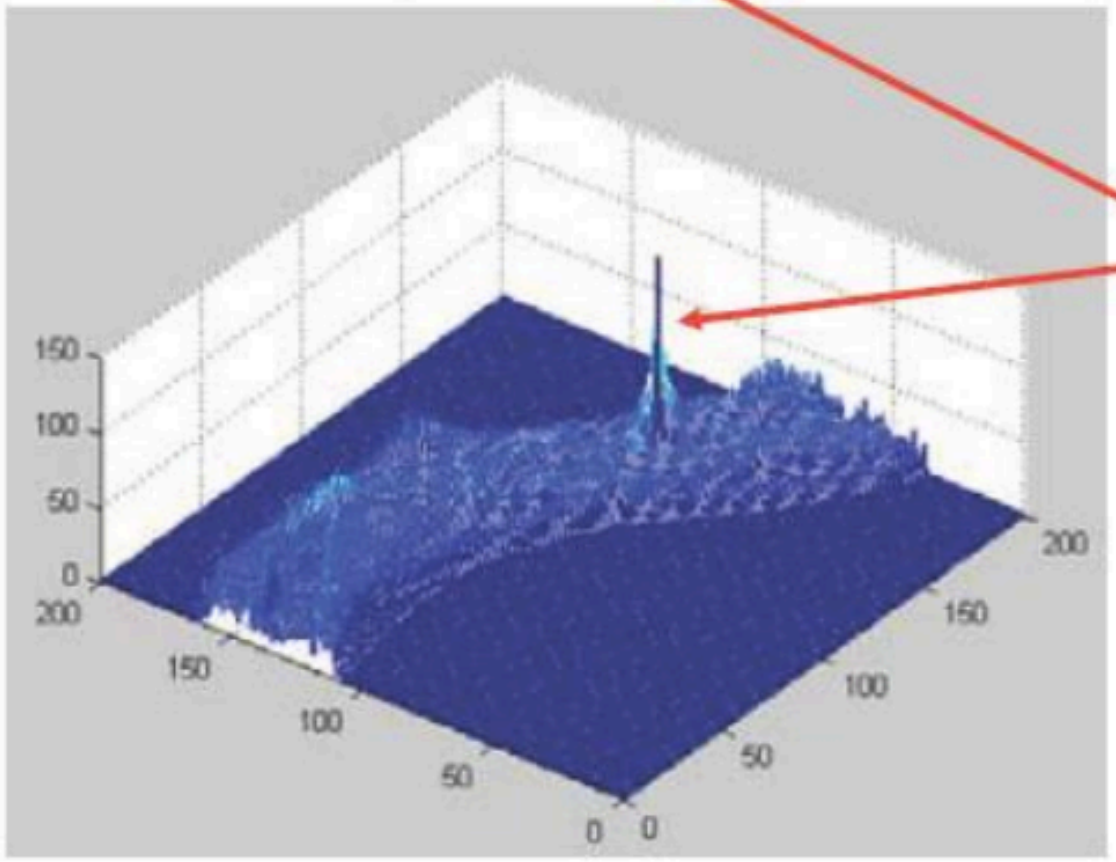


Image space



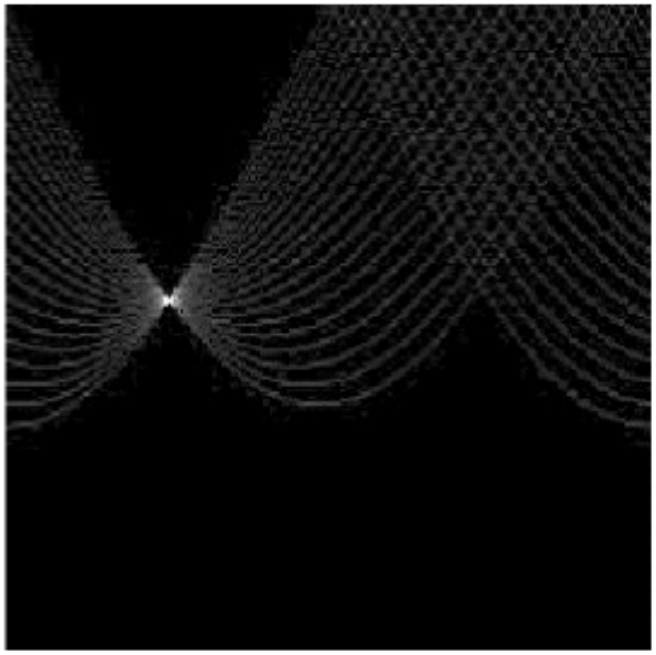
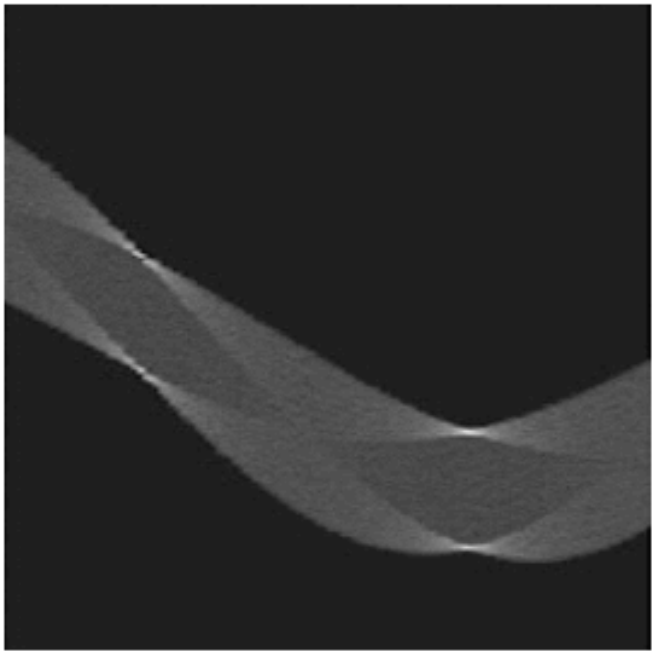
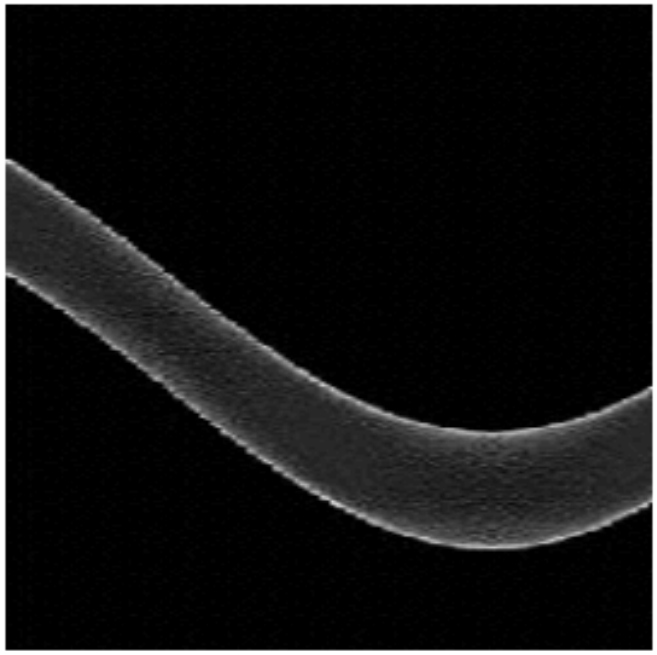
Votes





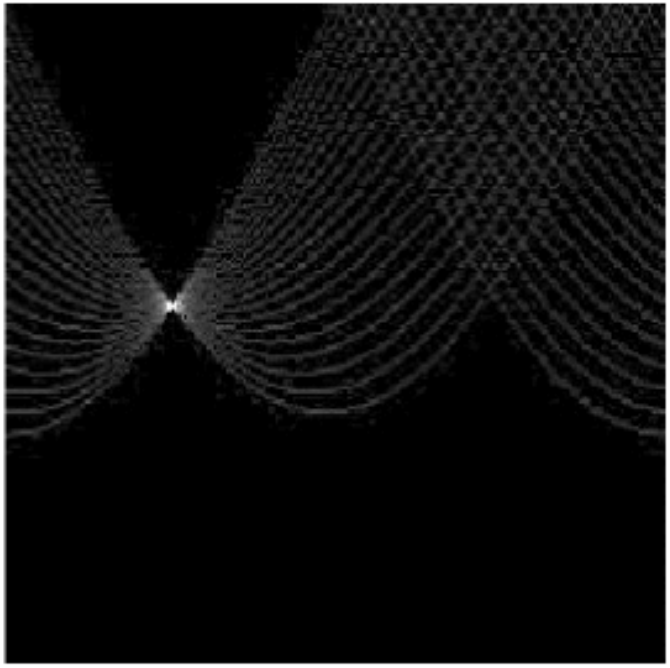
Picos no espaço paramétrico correspondem à linha

Formas básicas (no espaço polar)

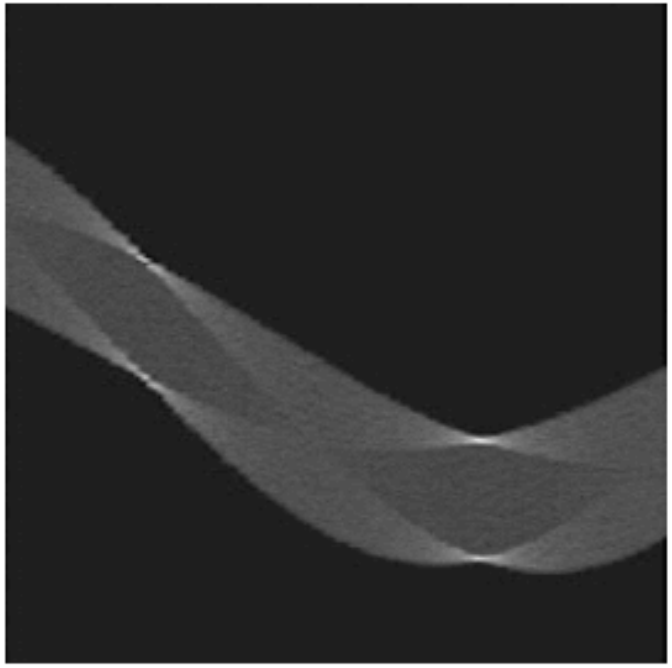




Quais são os formatos?

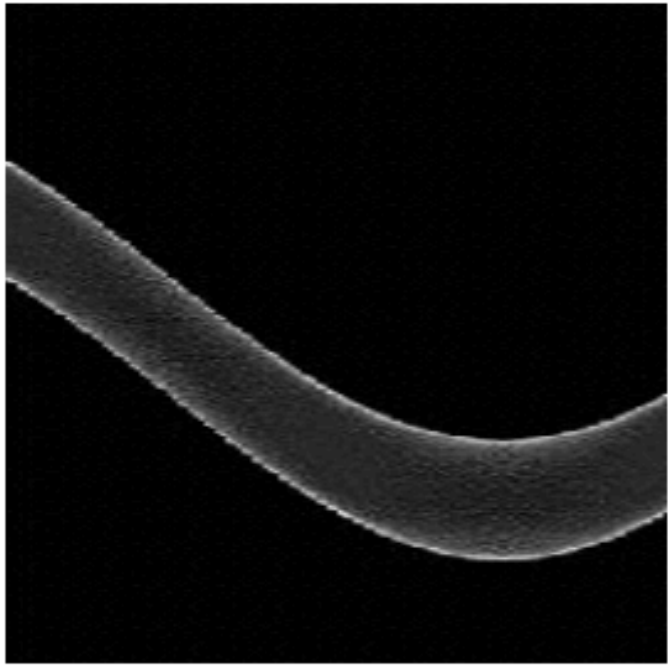
Formas básicas (no espaço polar)



linha

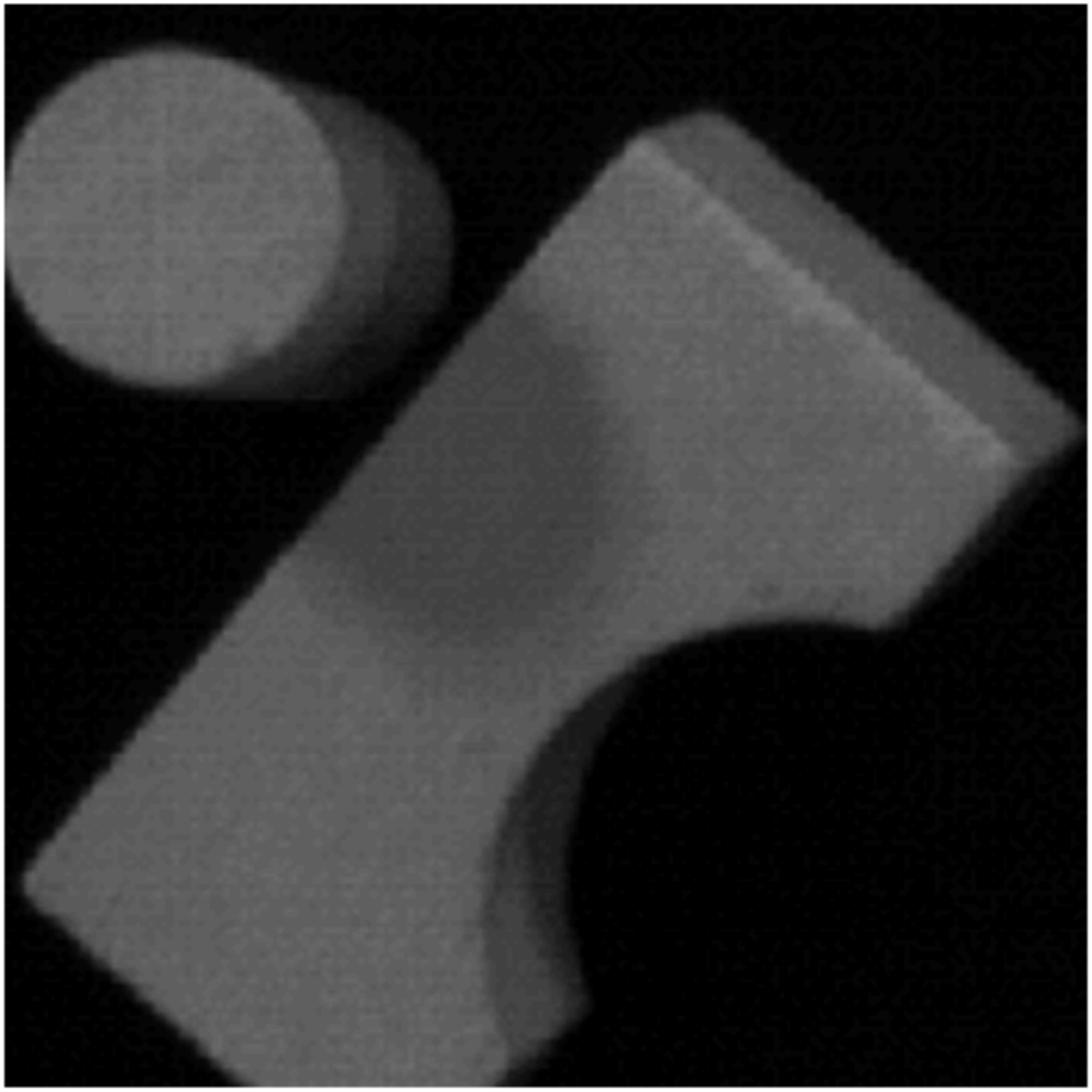


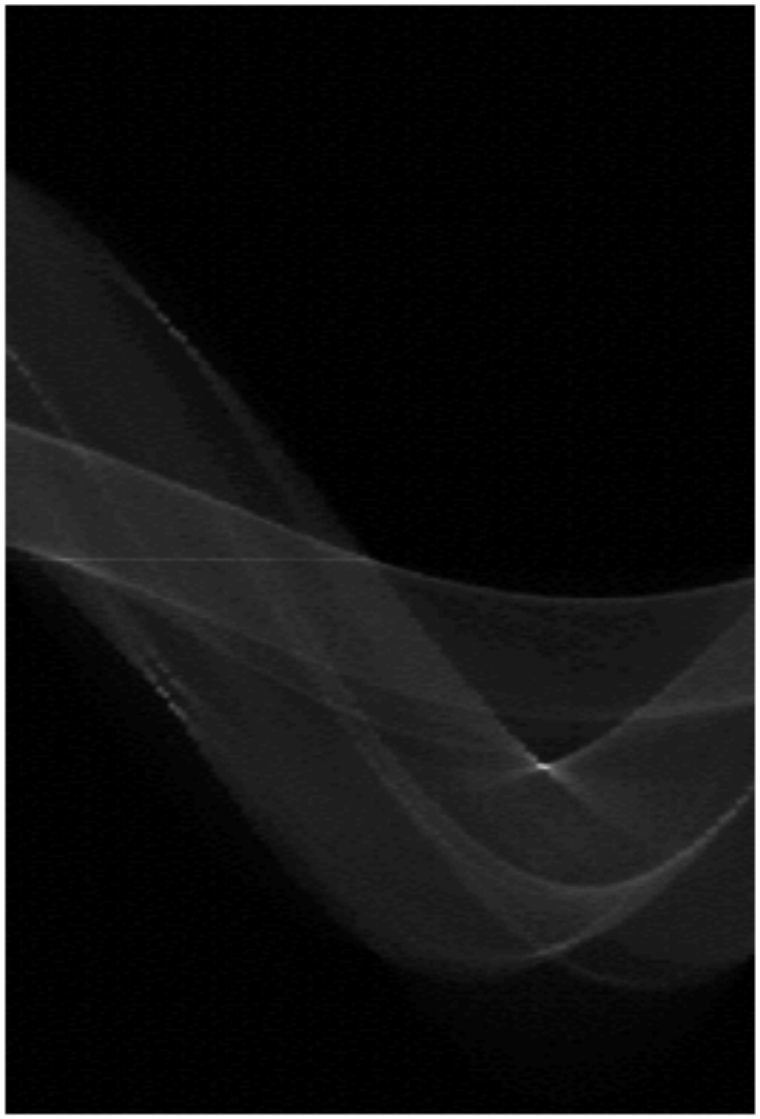
Retângulo



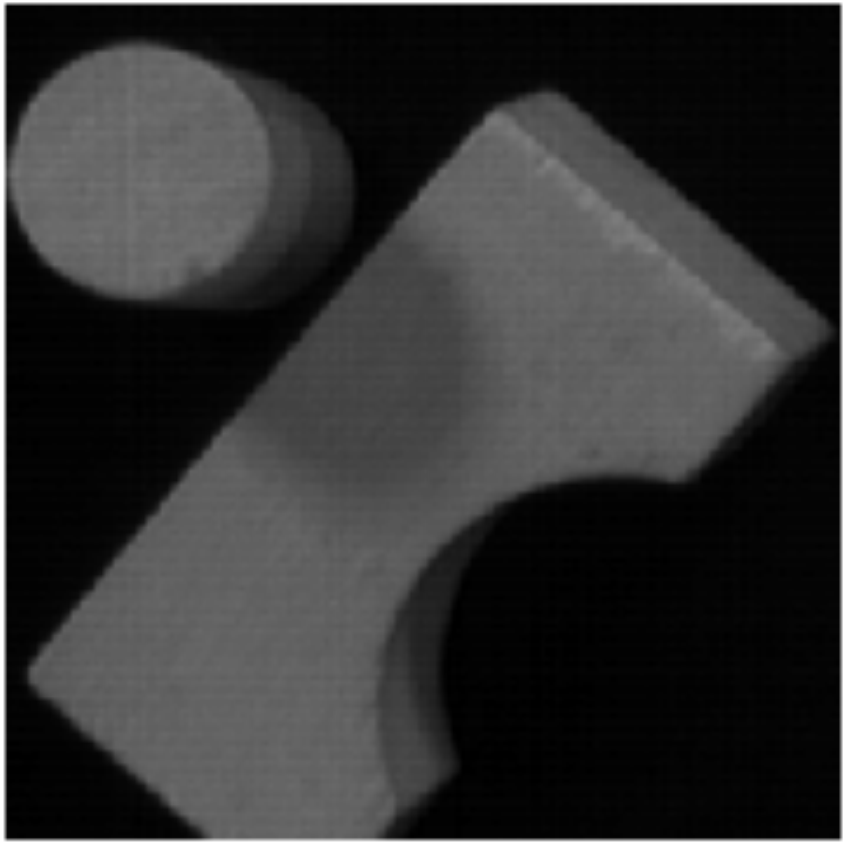
Círculo

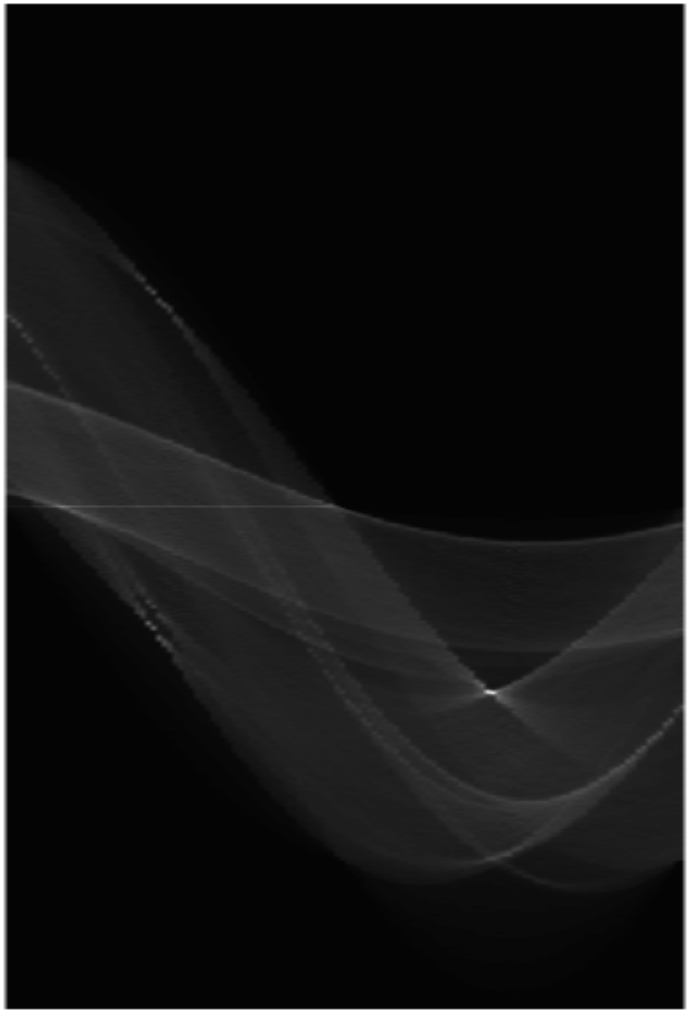
Formas básicas





Exemplo

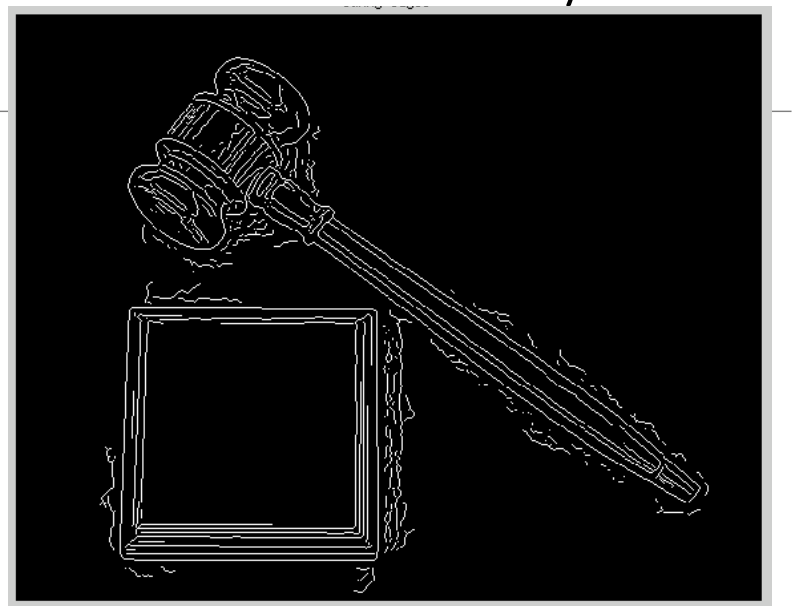




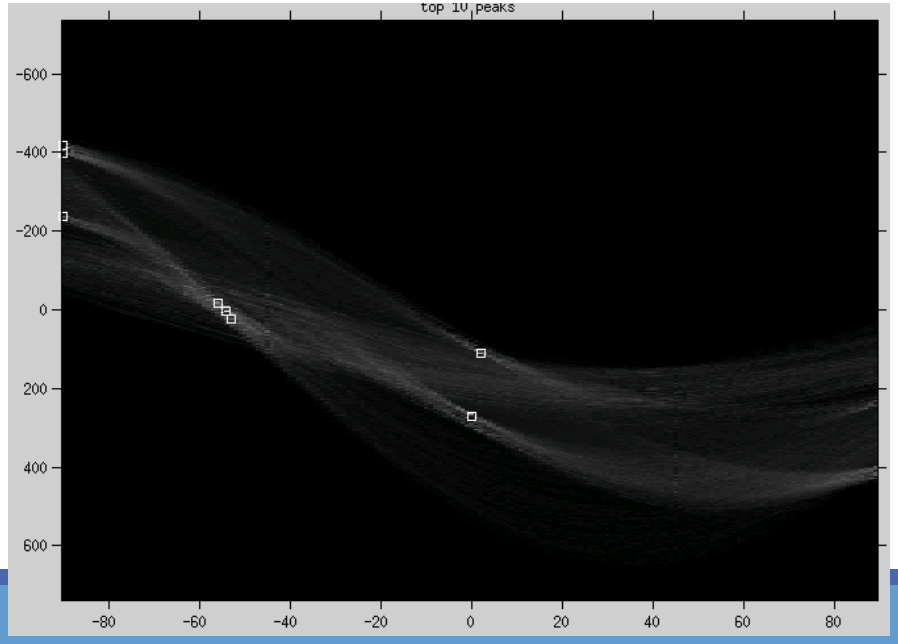
Qual linha gerou o ponto mais brilhante?

Imagem Original

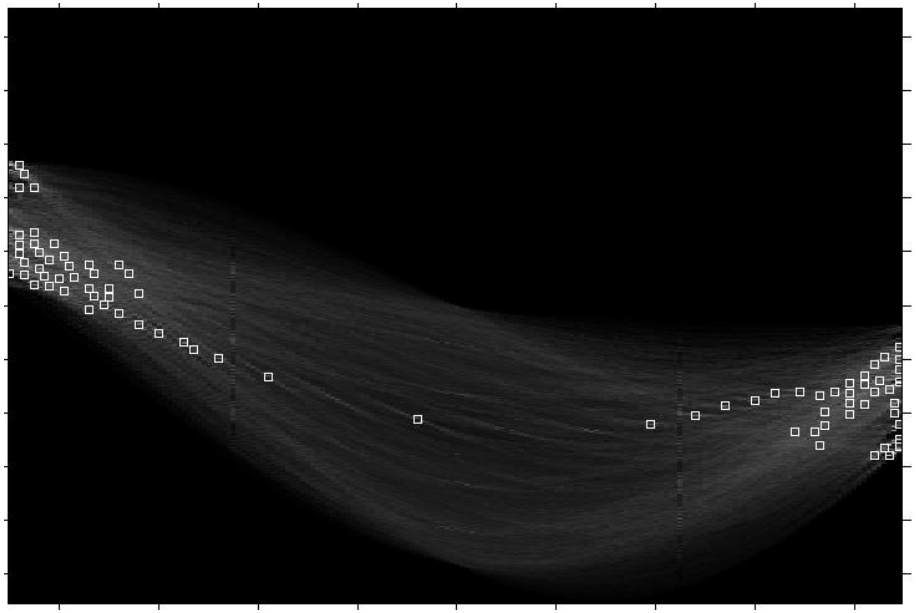
Bordas de Canny

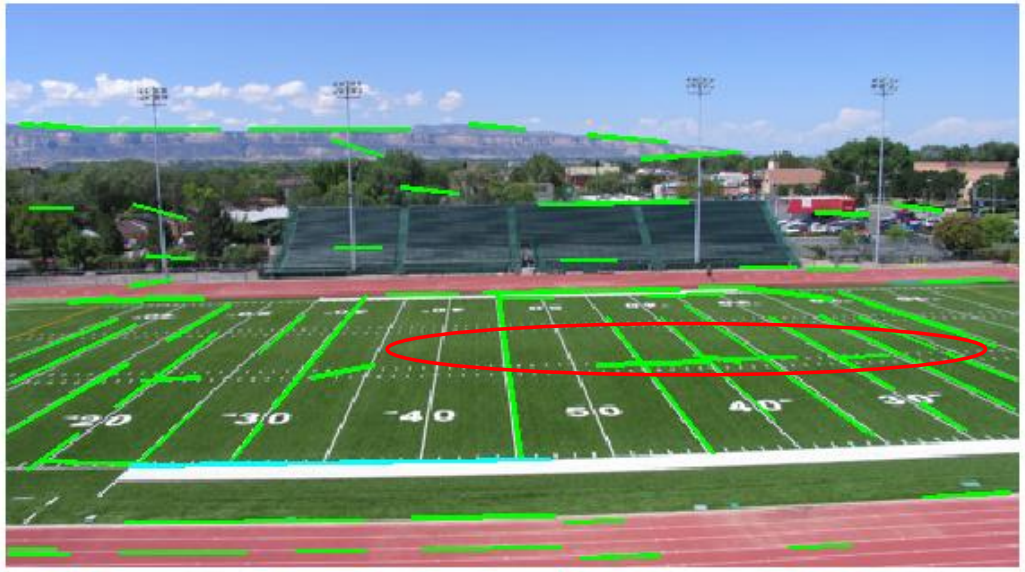


Espaço de votação



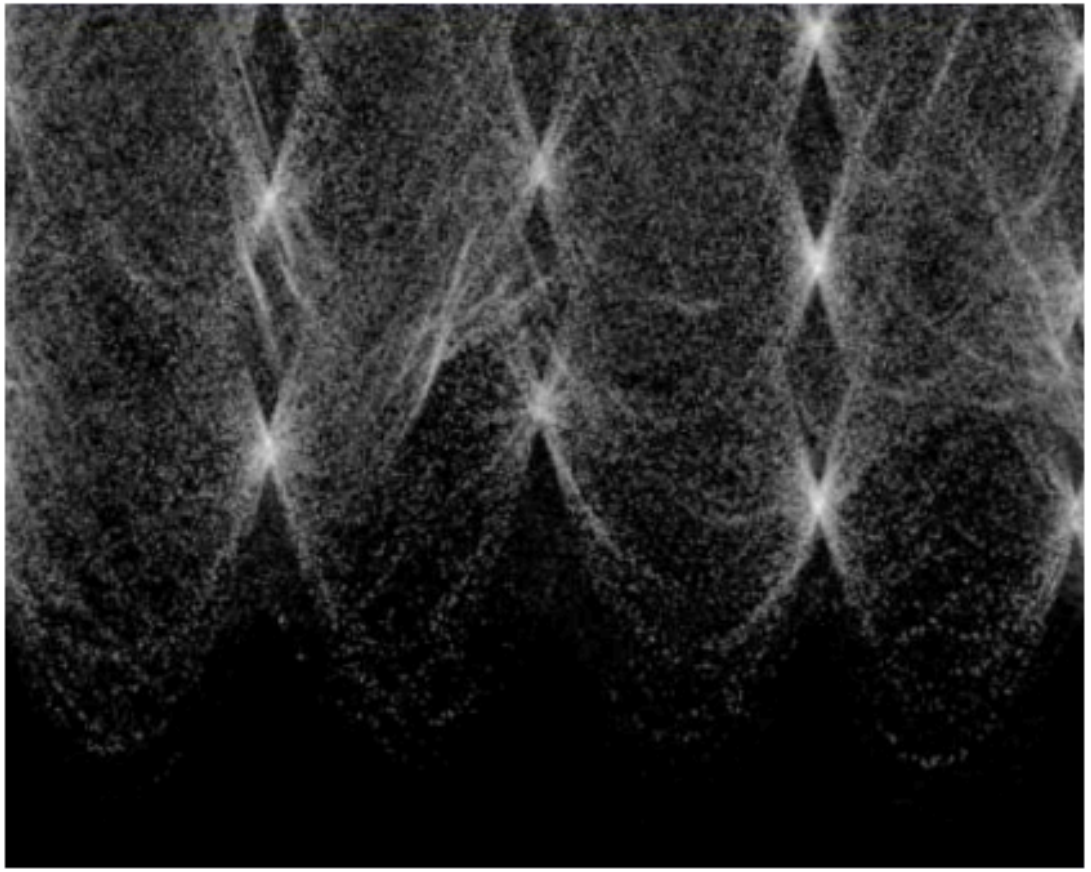
Slide credit: Kristen Grauman



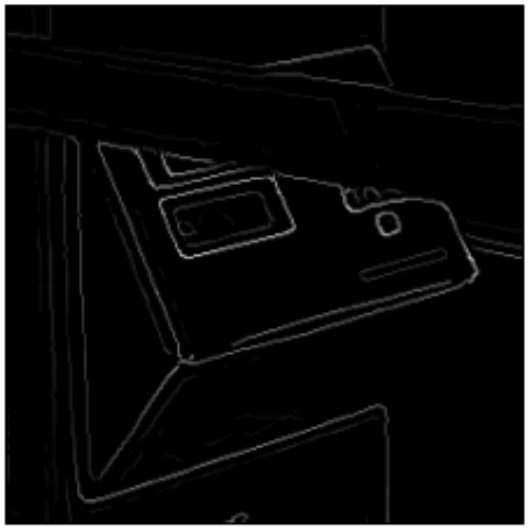


Slide credit: Kristen Grauman

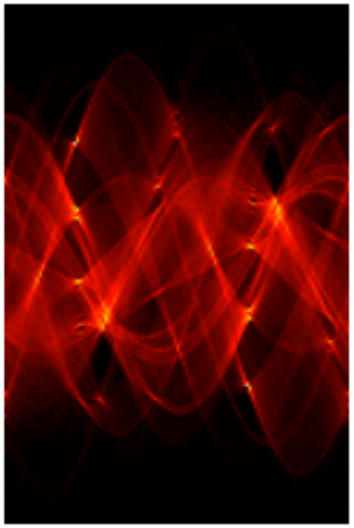
Imagens mais complexas



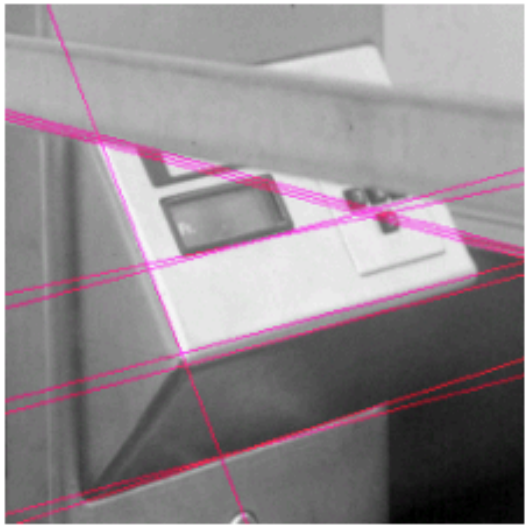
Original



Edges



parameter space



Hough Lines

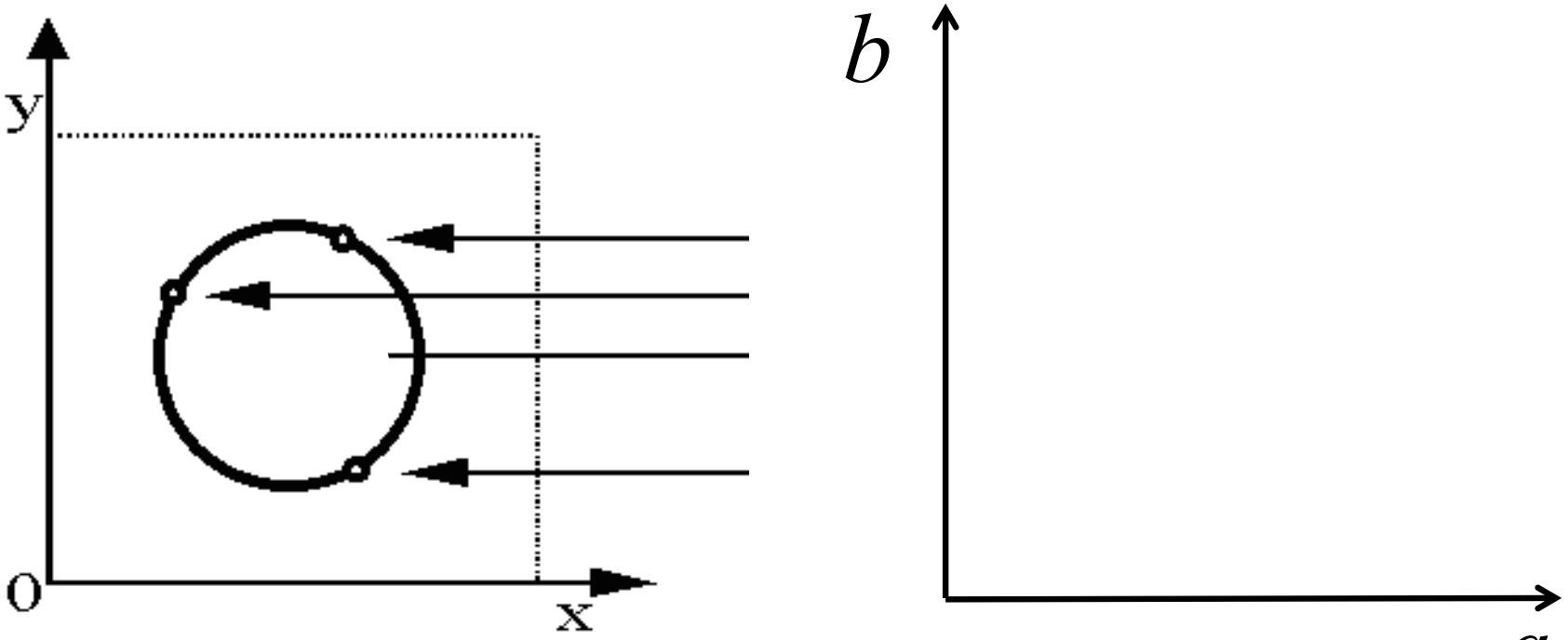
Generalized Hough Transform

Tranformada de Hough para círculos

- Círculo: centro (a,b) e raio r

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

▷Para um raio r fixo, direção do gradiente desconhecida



Espaço da Imagem

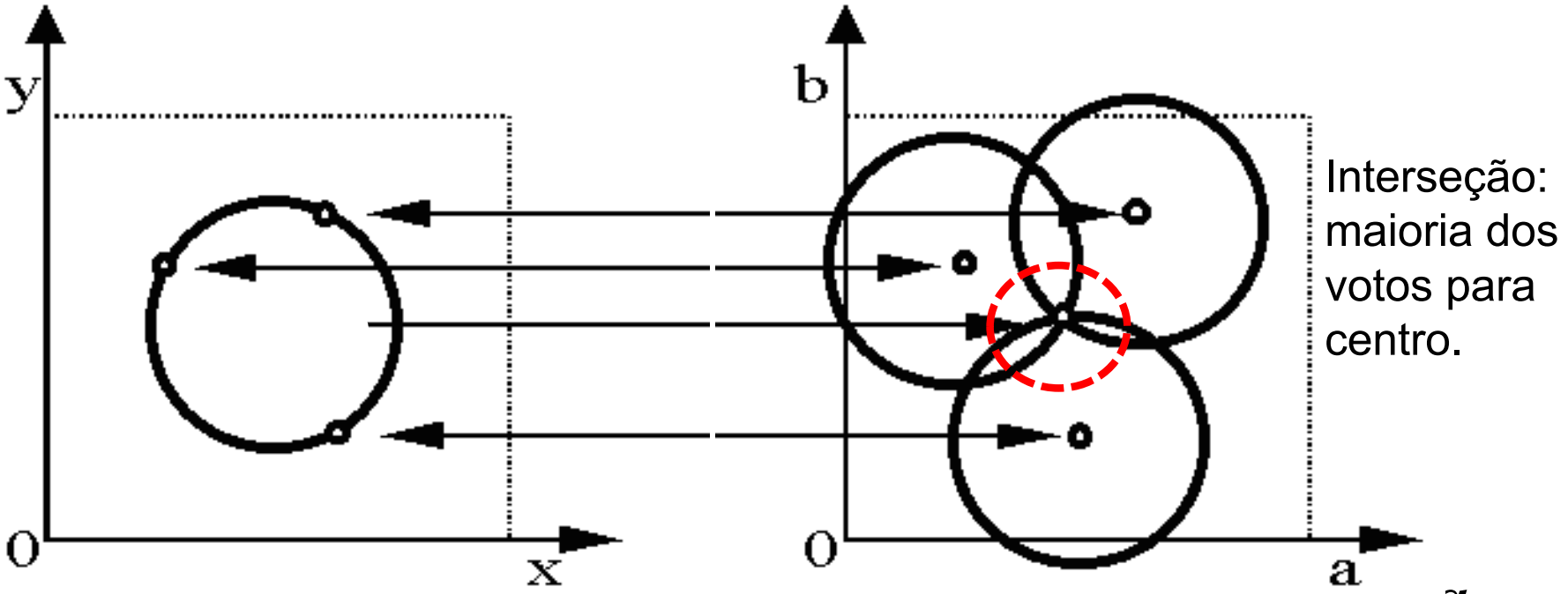
Espaço de Hough

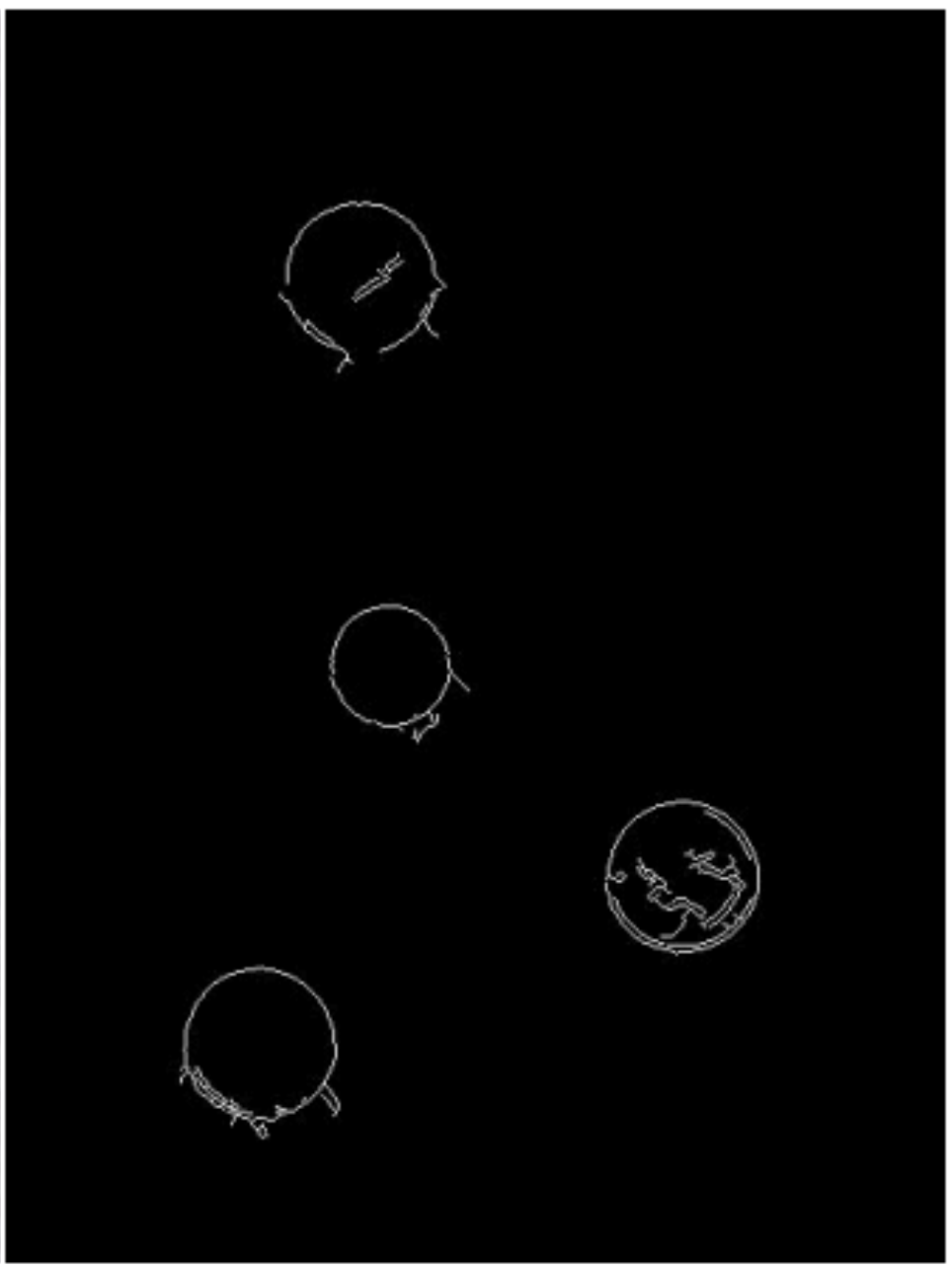
Tranformada de Hough para círculos

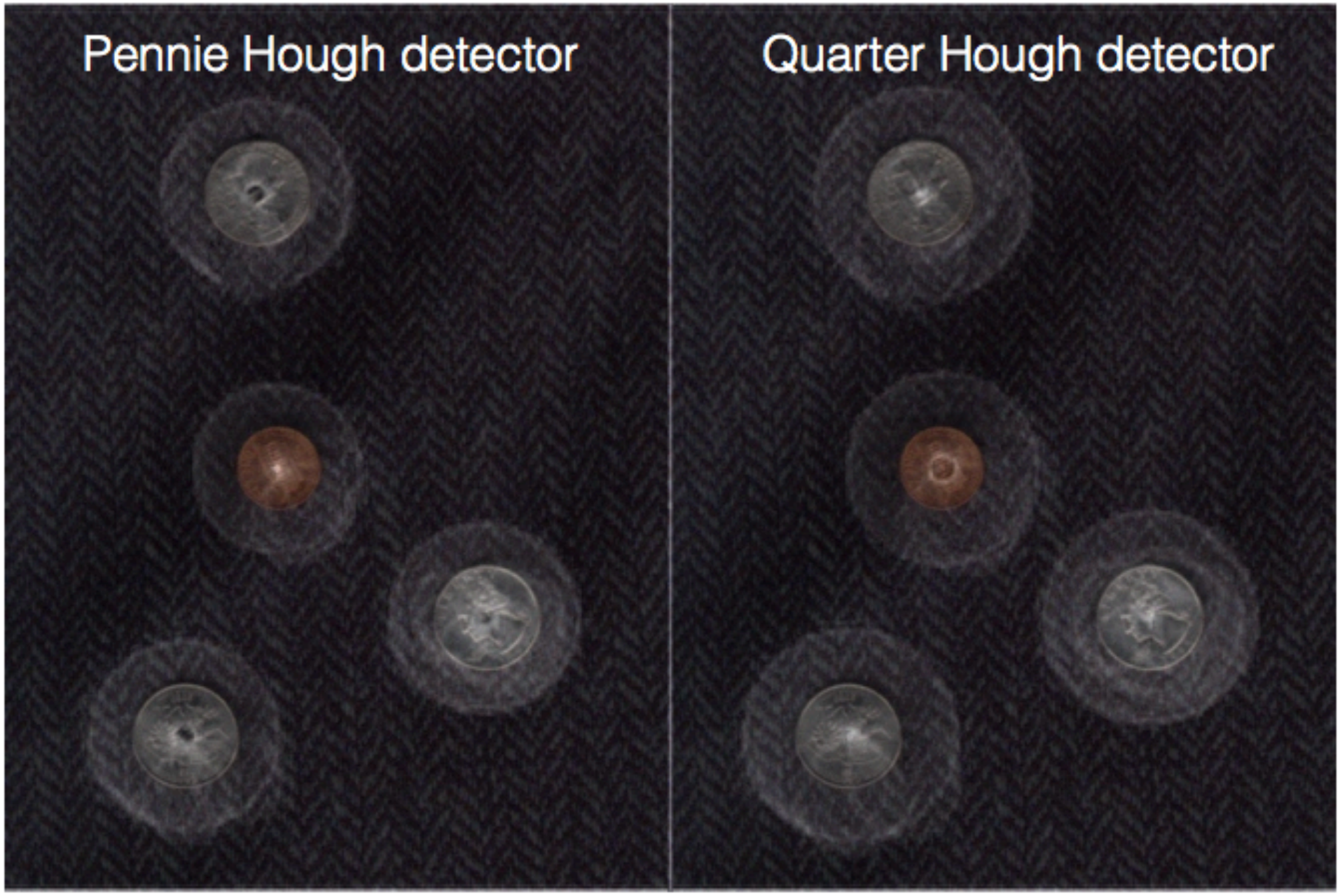
- Círculo: centro (a,b) e raio r

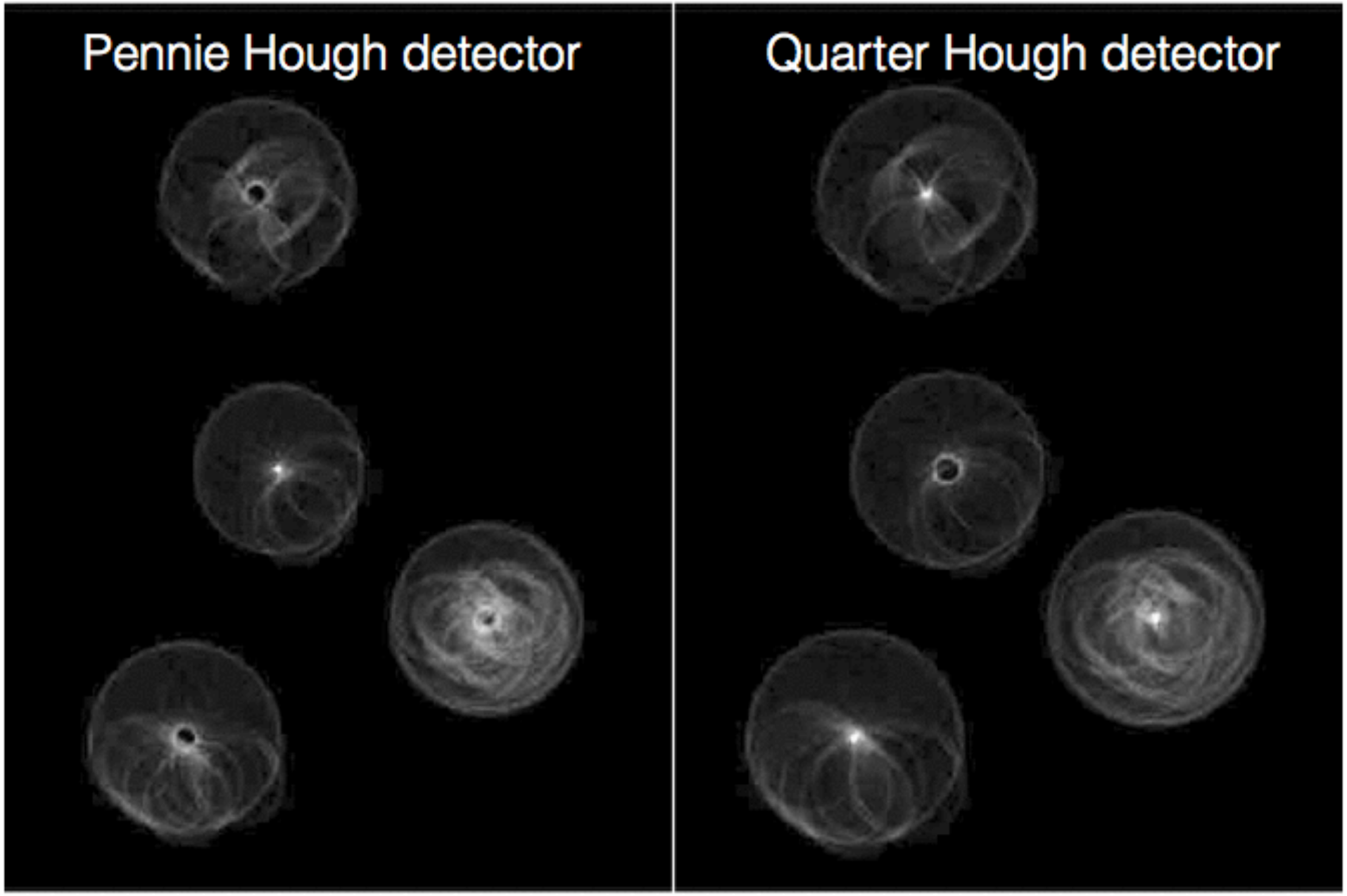
$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

▷Para um raio r fixo, direção do gradiente desconhecida



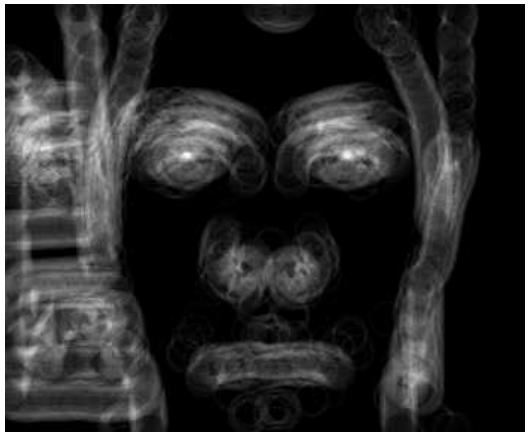



Pennie Hough detector
Quarter Hough detector


Pennie Hough detector
Quarter Hough detector

Example: iris detection

Gradient+threshold



Hough space
(fixed radius)

Max detections

Example: iris detection

Figure 2. Original image

Figure 3. Distance image

Figure 4. Detected face region

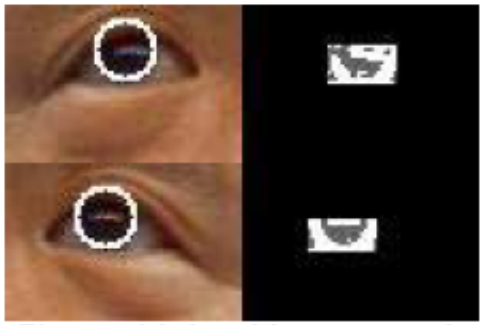


Figure 14. Looking upward

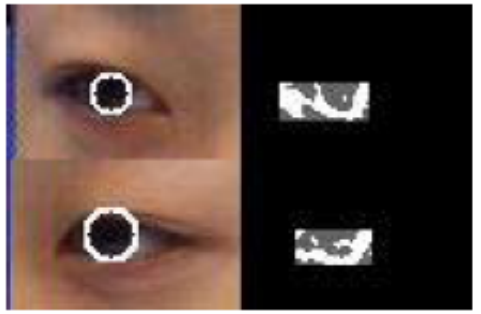


Figure 15. Looking sideways

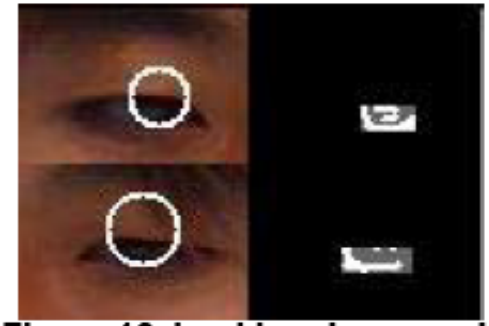
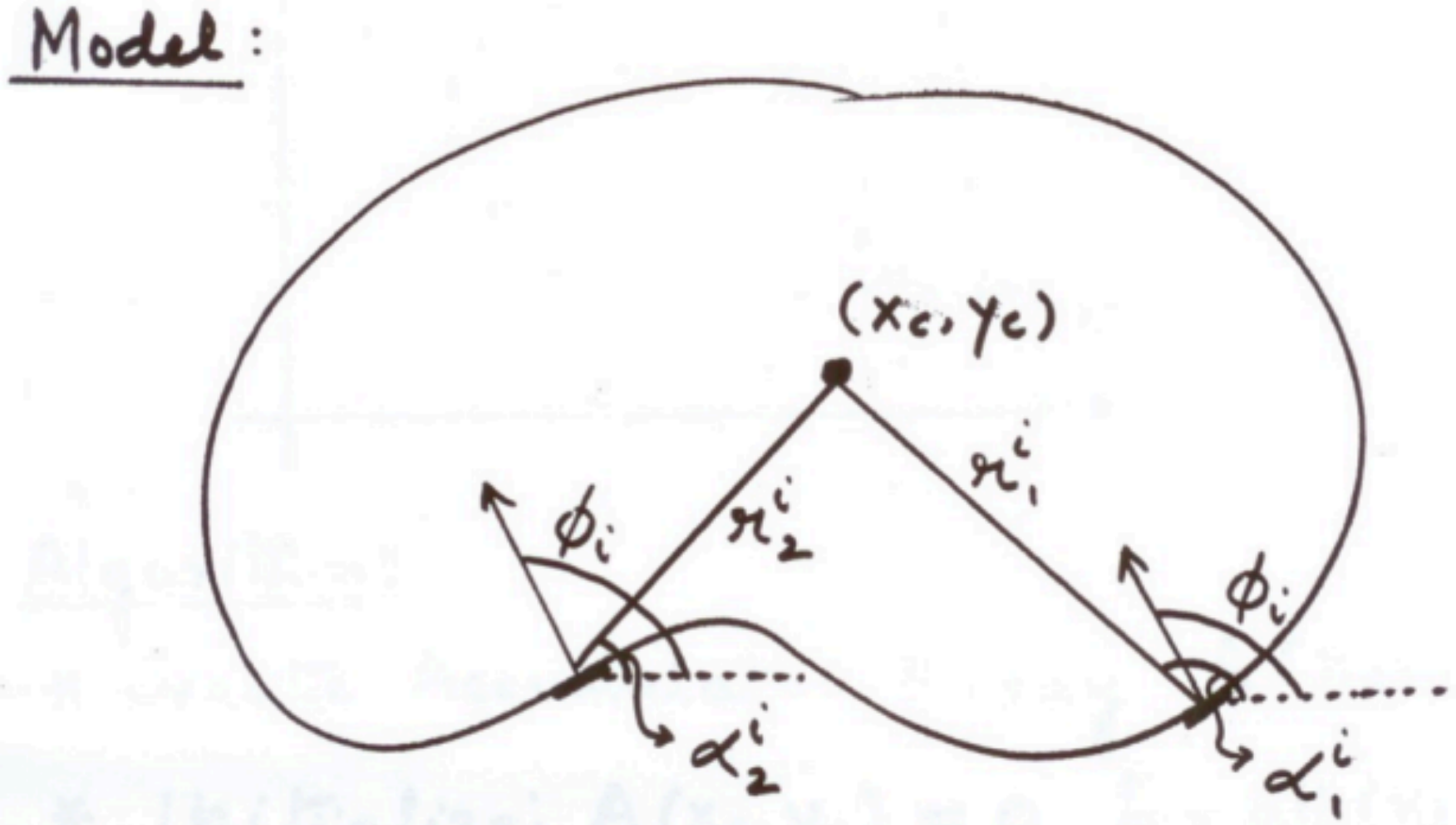


Figure 16. Looking downward

- An Iris Detection Method Using the Hough Transform and Its Evaluation for Facial and Eye Movement, by Hideki Kashima, Hitoshi Hongo, Kunihito Kato, Kazuhiko Yamamoto, ACCV 2002.

Transformada de Hough Generalizada



A. Train phase:

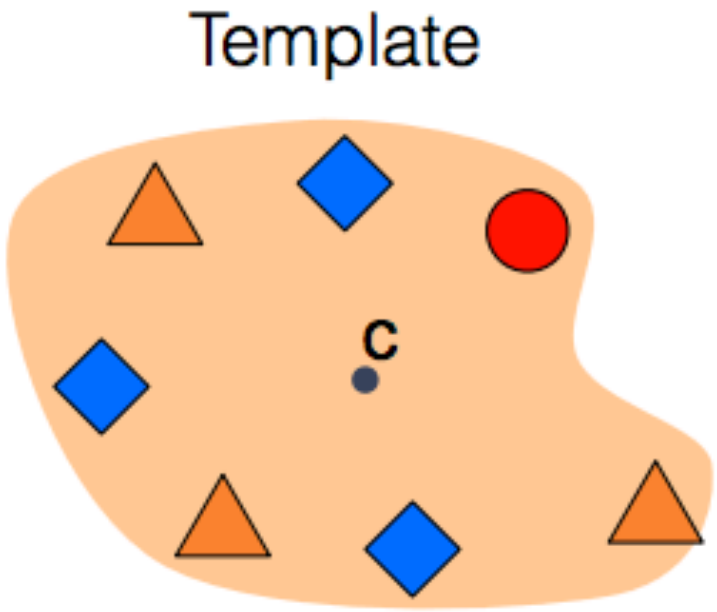
1. Get features

2. Store all displacements of feature from center

B. Test phase:

1. Get features & lookup displacements

2. Vote for center location



A. Train phase:

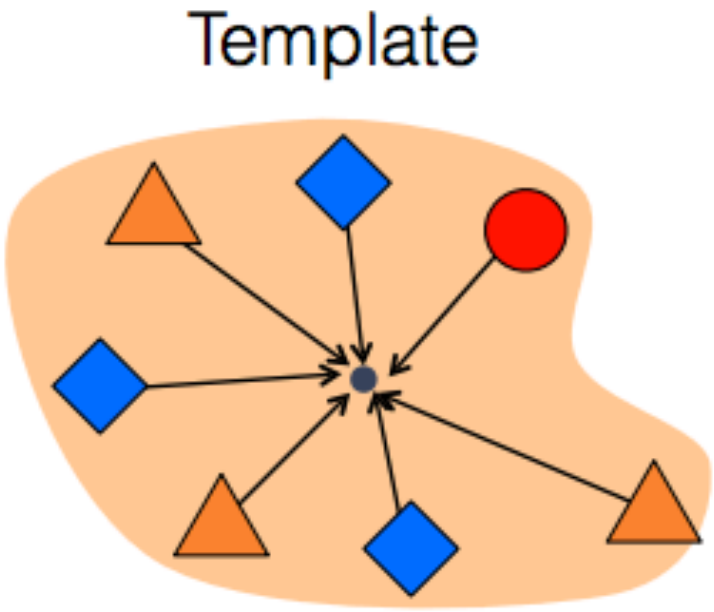
1. Get features

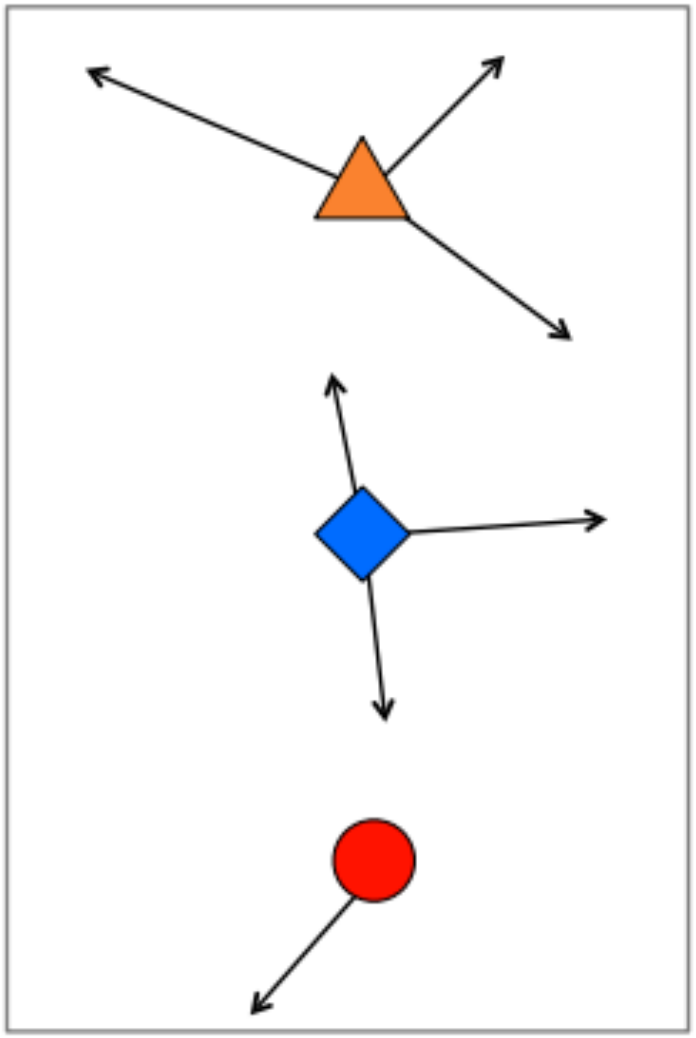
2. Store all displacements of feature from center

B. Test phase:

1. Get features & lookup displacements

2. Vote for center location





A. Train phase:

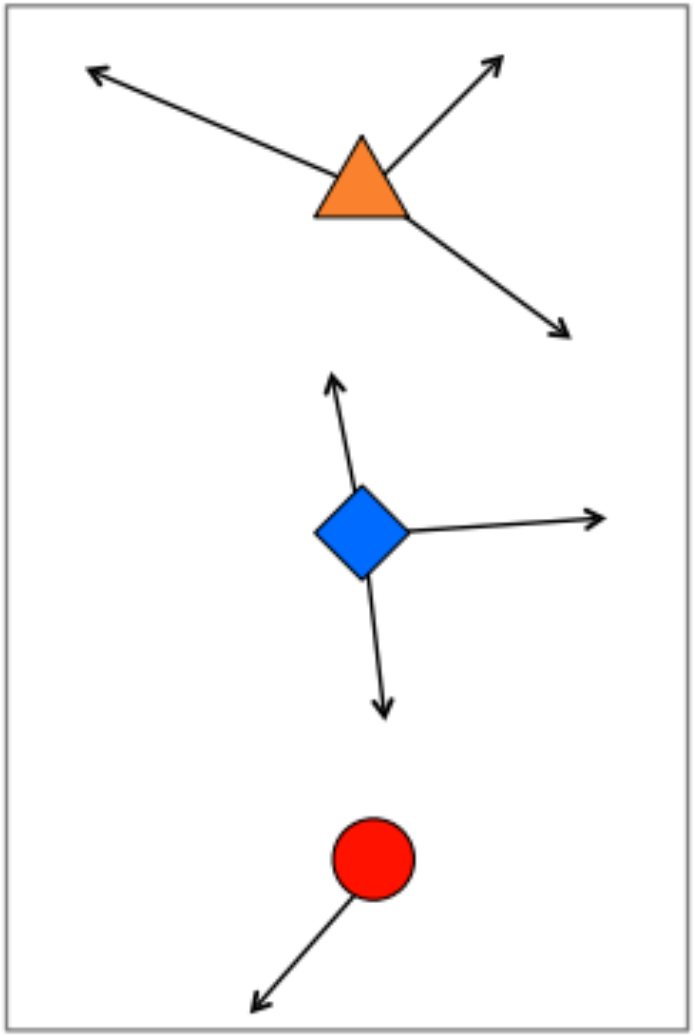
1. Get features

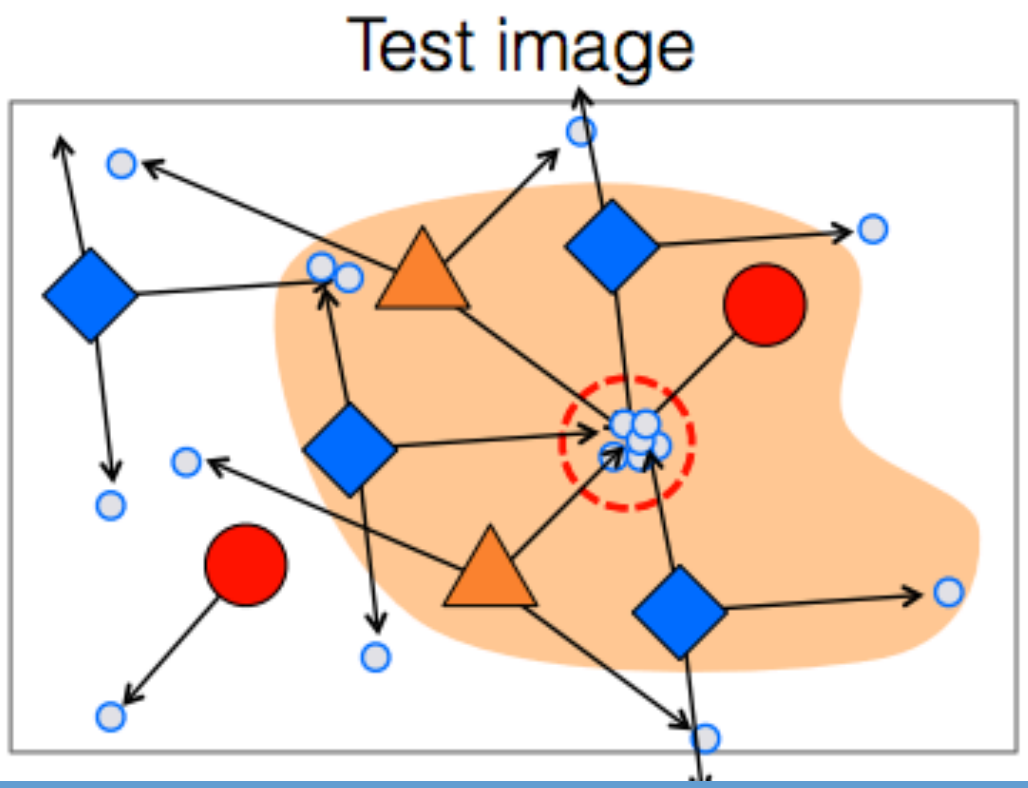
2. Store all displacements of feature from center

B. Test phase:

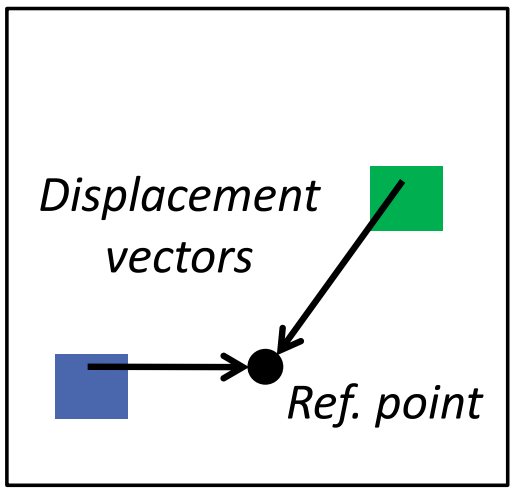
1. Get features & lookup displacements

2. Vote for center location

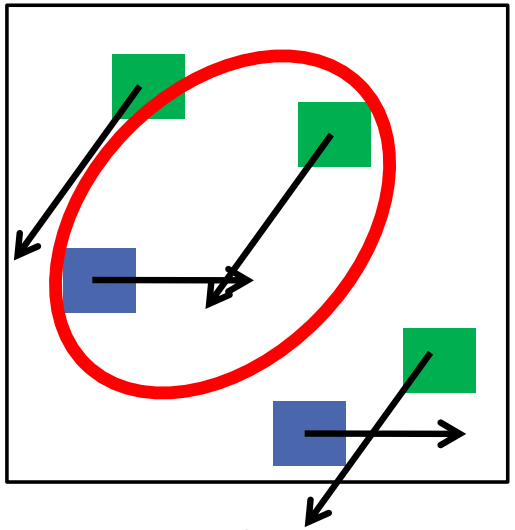




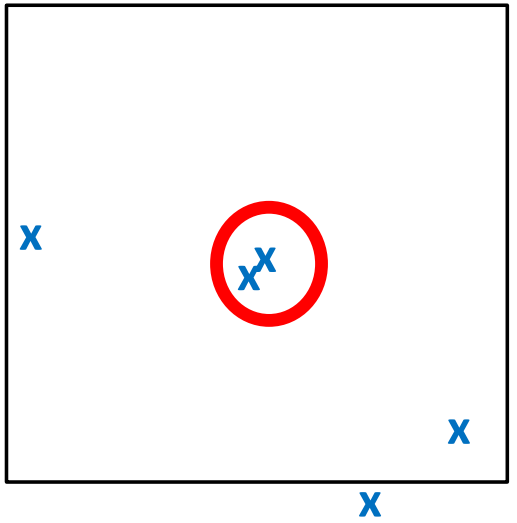
Transformada de Hough Generalizada



Model image



Novel image

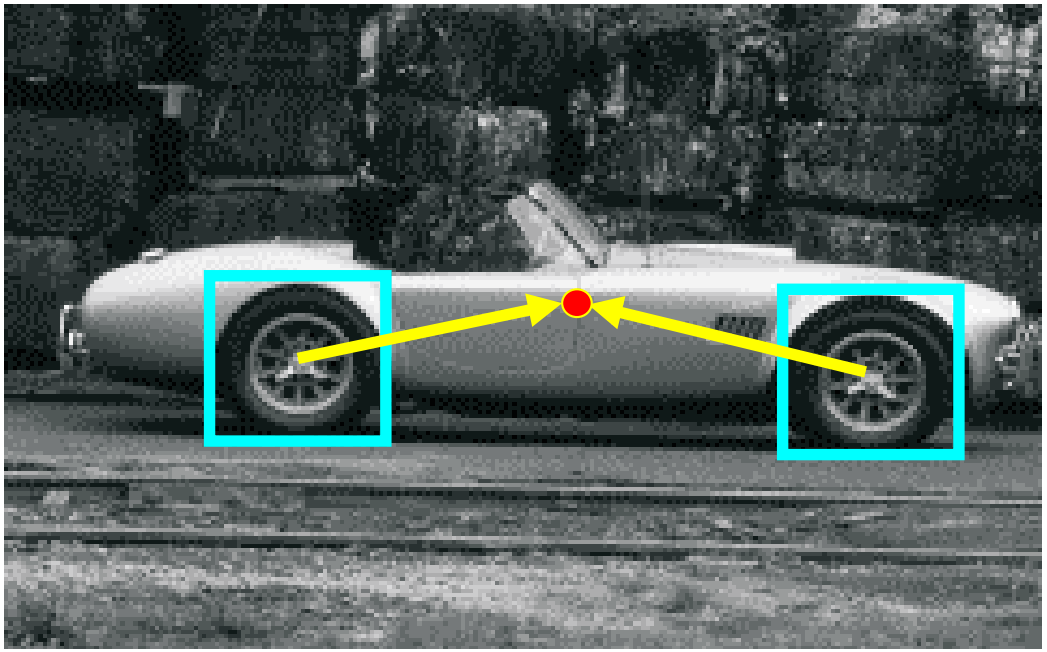


Vote space

Aplicações das Transformadas de Hough

Generalized Hough for object detection

- Instead of indexing displacements by gradient orientation, index by matched local patterns.



training image

“visual codeword” with displacement vectors

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Source: L. Lazebnik

Generalized Hough for object detection

- Instead of indexing displacements by gradient orientation, index by “visual codeword”

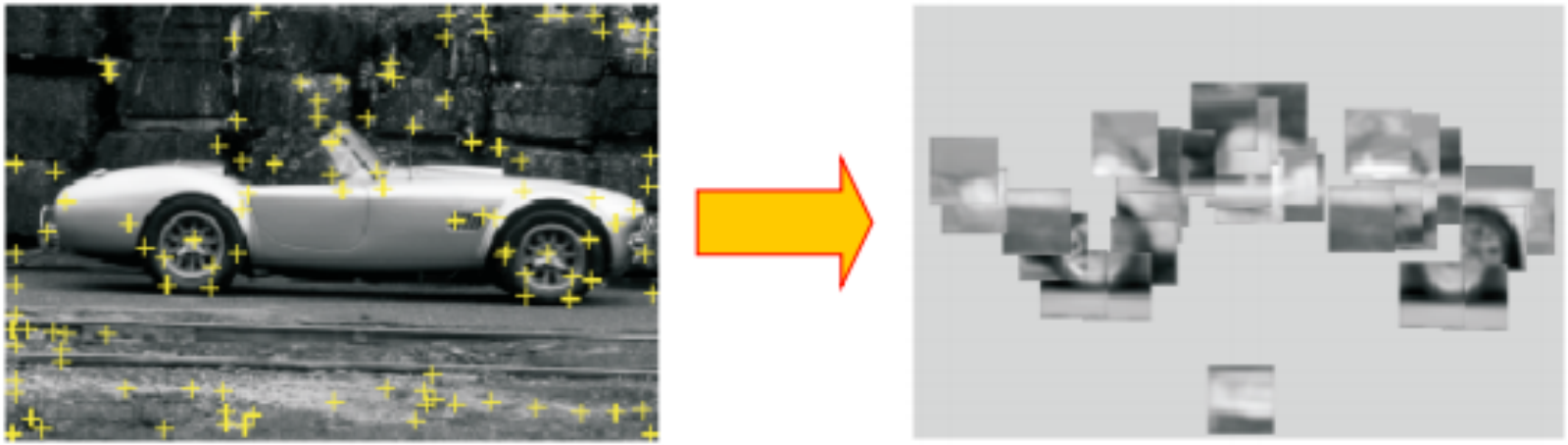
test image

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Source: L. Lazebnik

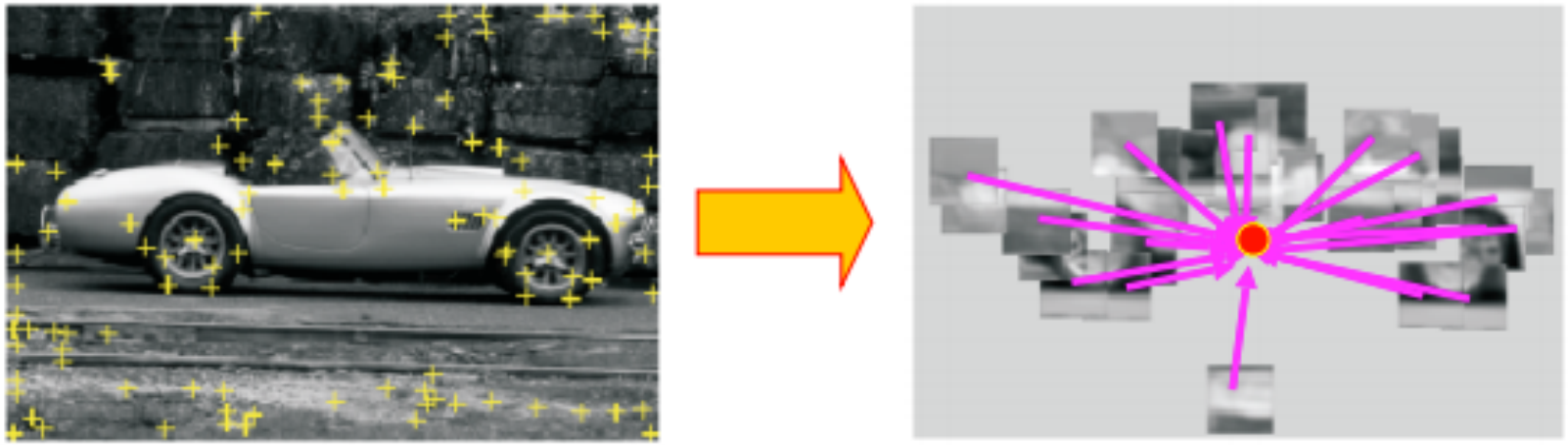
Fase de Treinamento

- 1. Adquirir características

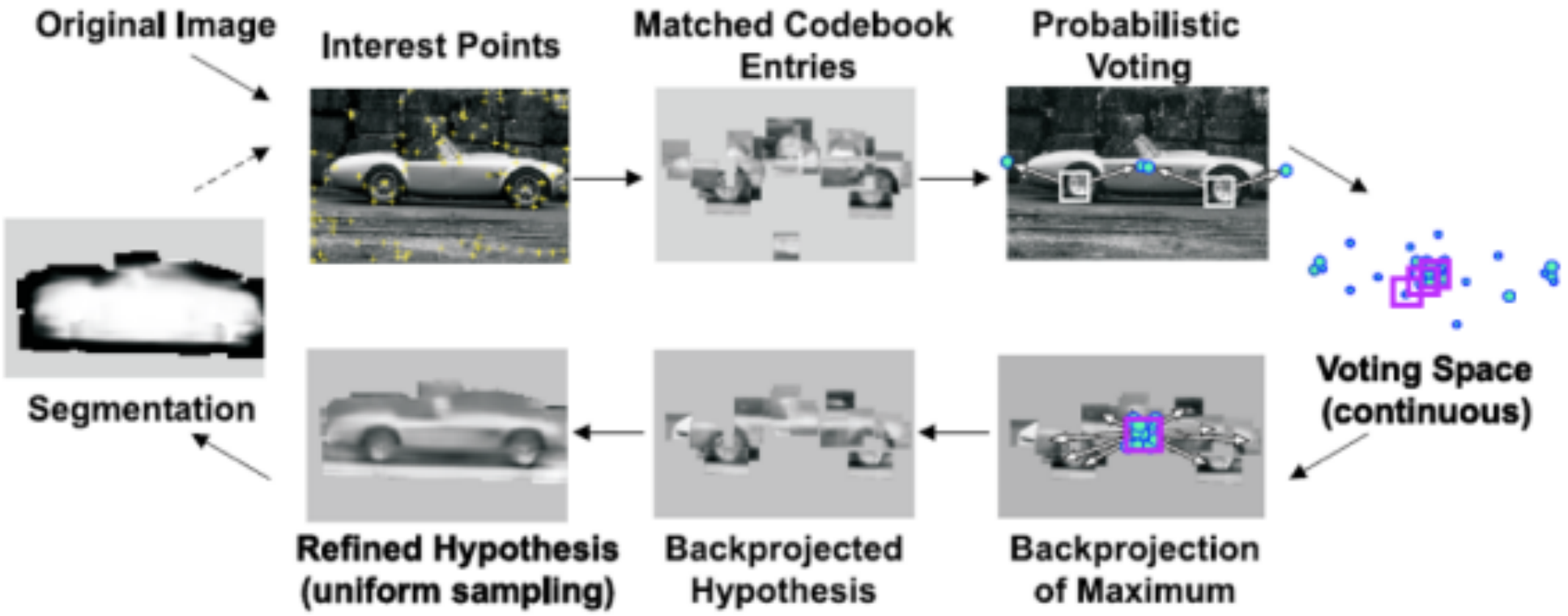


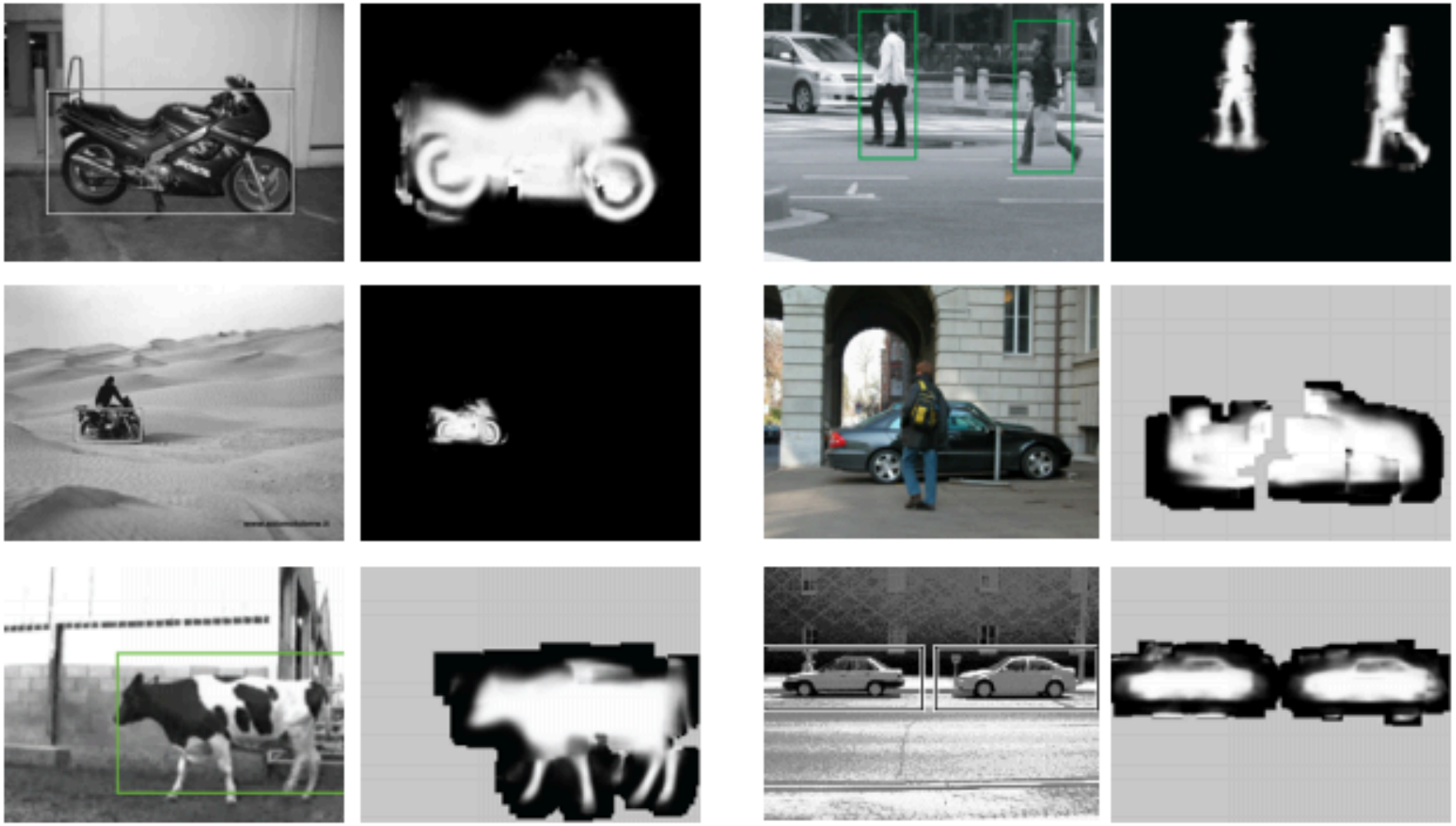
Fase de treinamento

- 2. store displacements



Fase de teste





Transformada de Hogh...

Lida bem com oclusão? 

Detecta múltiplas instâncias? 

Robusto a ruído? 

Baixa complexidade computacional? 

Fácil ajustar parâmetros?