

Grupo de Estudos de Visão Computacional

Davi de Jesus Teixeira
Caio Lucca Oliveira
Rian de Souza Santos

Apoio



Semana 3: Detecção de Bordas e Extração de Características

Goiânia
2024



Tópicos da Apresentação:

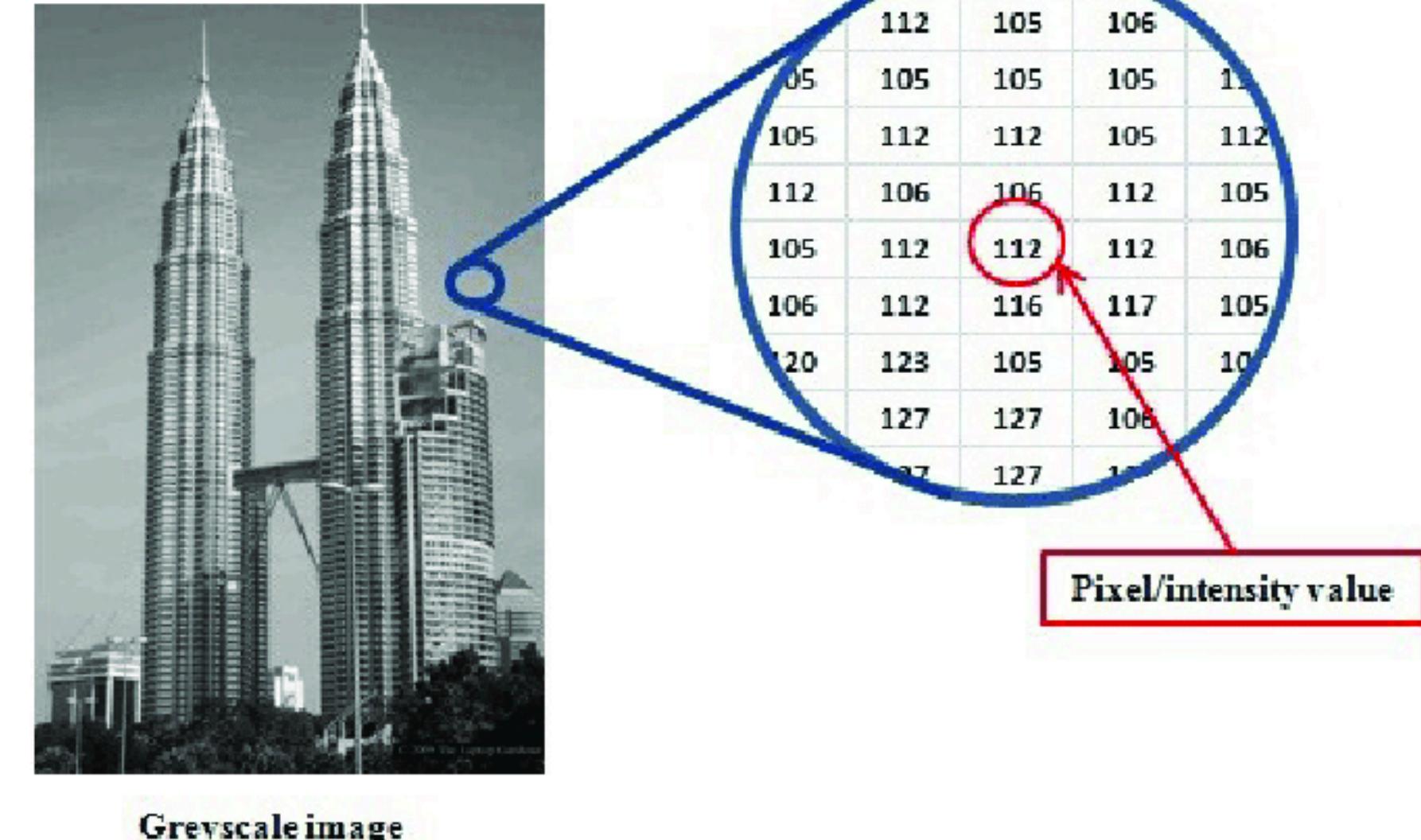
1. Intuição e Introdução
2. Edge Detection
3. Corner Detection
4. Feature Description
5. Prática

Recapitulando:

Imagen - Representação visual de algo.

Para o computador, uma imagem é representada como uma matriz de pixel N-dimensional:

- Imagem preta e branca - 2D
- Imagem “colorida” - 3D,
- Imagem “colorida” com canal de transparência - 4D



Recapitulando:

"Hoje em dia, apesar dos avanços significativos em visão computacional (Vision Transformers, Modelos Multi-modais etc.), que melhoraram bastante o desempenho em tarefas de visão computacional, como classificação de imagens, segmentação, e geração de descrições, as máquinas ainda estão longe de alcançar a compreensão profunda e causal que uma criança de dois anos tem ao observar uma imagem." -Richard Szeliski

A visão computacional é uma área complexa justamente porque é um **problema inverso**. Isso significa que, a partir de informações incompletas ou ambíguas, tentamos inferir aspectos complexos do mundo real, como a forma, a cor, a textura, e a posição tridimensional dos objetos.

Recapitulando:

Ou seja, além de passar as informações para um modelo no formato que ele espera (matricial), precisamos desenvolver técnicas para processar a imagem de forma com que ela se torne mais **significativa** para o computador

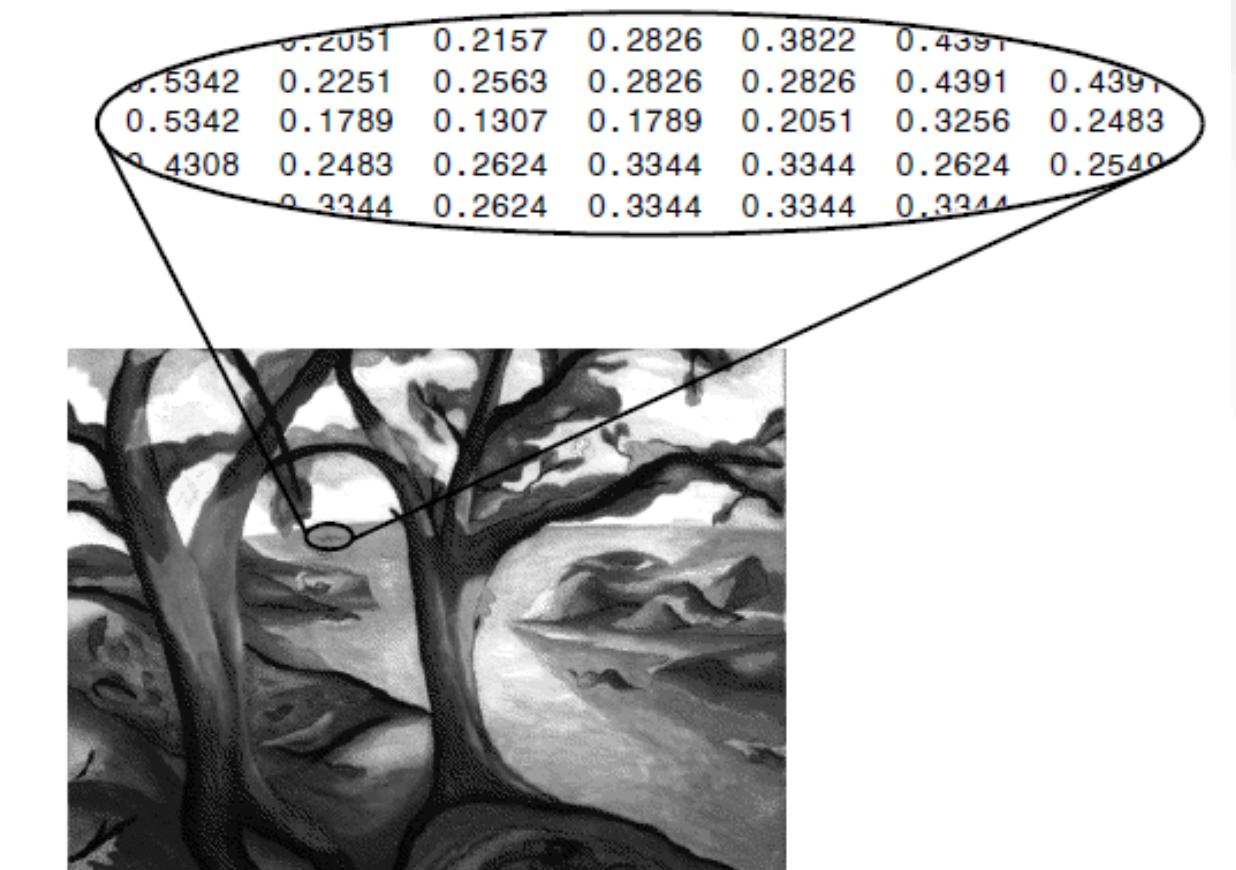
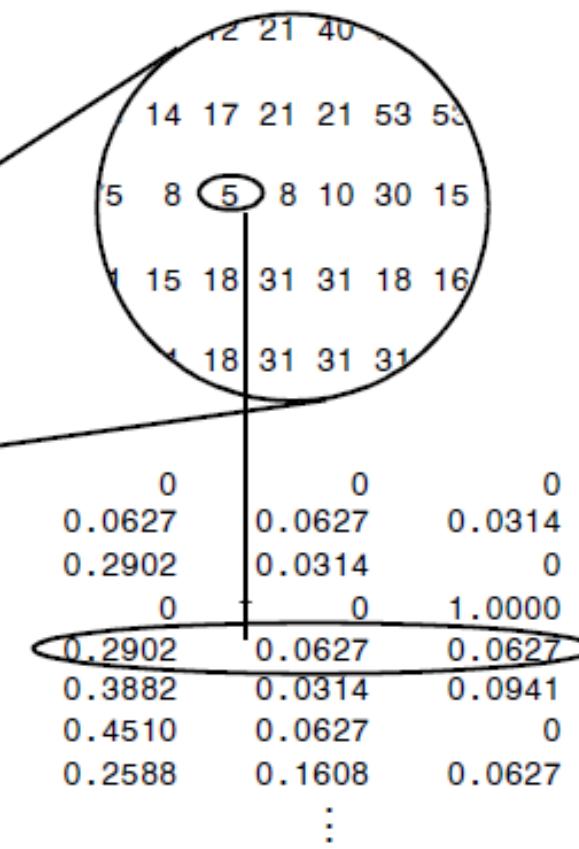
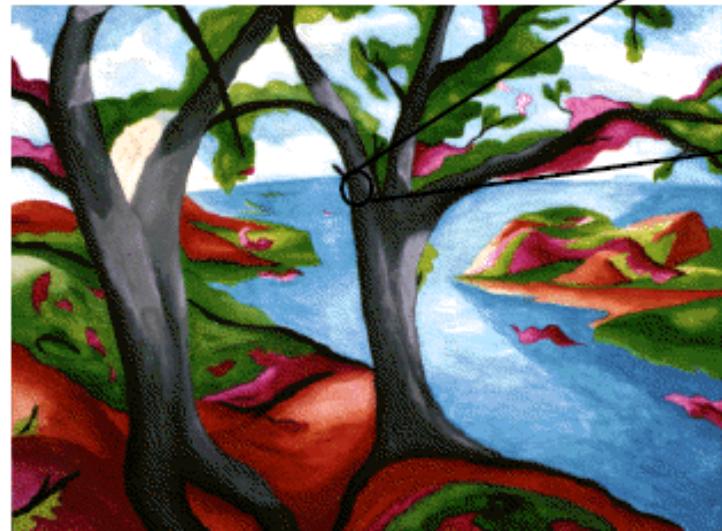
Imagens em RGB podem ser excessivas e desnecessárias para um modelo:

- Possuem muitos “dados” relacionados à coloração
- São mais pesadas por conta dessa característica
- Podem trazer ruído e ambiguidade ao modelo por questões de iluminação ou semelhança de cores

Imagens em Grayscale ainda são “abstratas” para o modelo :

- Não destacam as características importantes da imagem
- São mais leves que RGB, o que é bom para o processamento
- Muitas regiões “homogêneas” ou seja, regiões onde a intensidade da cor não muda muito (reduntante)

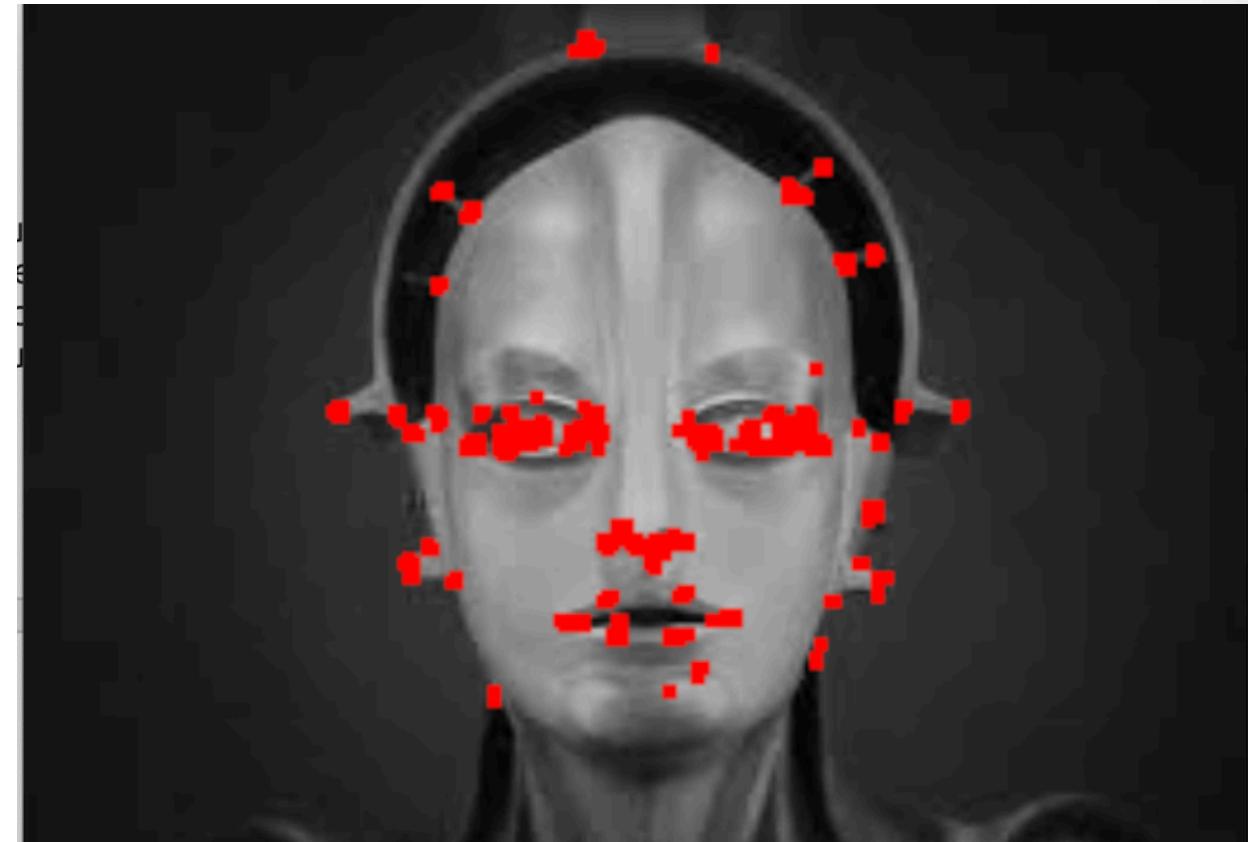
Recapitulando:



| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | |
| 0 | 0 | 0 | 0 | 1 | |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |

Como tornar uma imagem mais significativa:

Uma idéia: detectando **bordas** e **cantos**



Borda: criada a partir da variação da intensidade da imagem em pontos específicos

Canto: criado a partir da identificação de pontos de intersecção de duas ou mais bordas

Recapitulando:

Detectores de Bordas:

- Gradient Operator
- Laplacian (leve variação do gradient operator)
- Sobel
- Canny-Edge

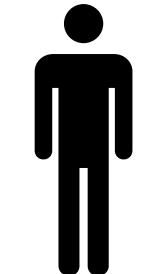
Detectores de Cantos (Corners):

- Harris Corner Detection
- Shi-Tomasi

E por fim, temos os **feature descriptors** que servem basicamente para criar uma representação matemática detalhada e única em torno de cada ponto de interesse (como um canto) para que ele possa ser comparado com pontos semelhantes em outras imagens, permitindo o reconhecimento do ponto em diferentes imagens, mesmo sob mudanças de escala, rotação e iluminação.

Antes de tudo, uma introdução a convolução:

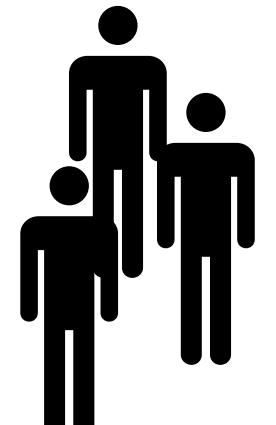
Pacientes distribuidos ao longo da semana:



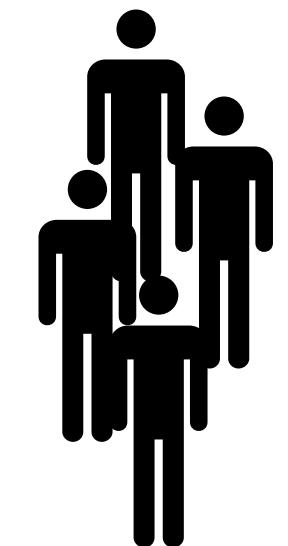
Segunda - 1



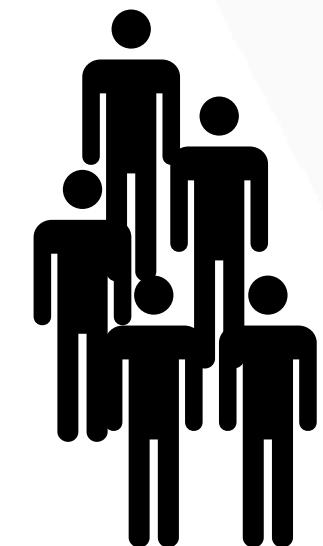
Terça - 2



Quarta - 3



Quinta - 4



Sexta - 5

Medicamento:

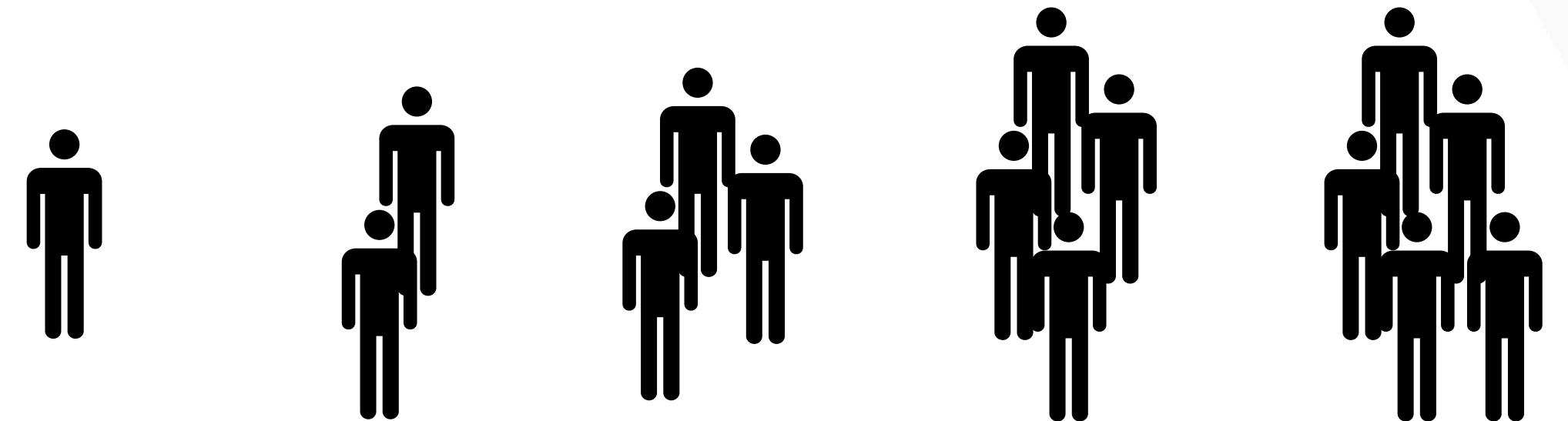


Dose única - 3 doses

Antes de tudo, uma introdução a convolução:

CASO 1: Quantas doses por semana seriam gastas discriminadas por dia:

Pacientes distribuidos ao longo da semana:



Segunda - 1

$$3 \times 1 = 3$$

Terça - 2

$$3 \times 2 = 6$$

Quarta - 3

$$3 \times 3 = 9$$

Quinta - 4

$$3 \times 4 = 12$$

Sexta - 5

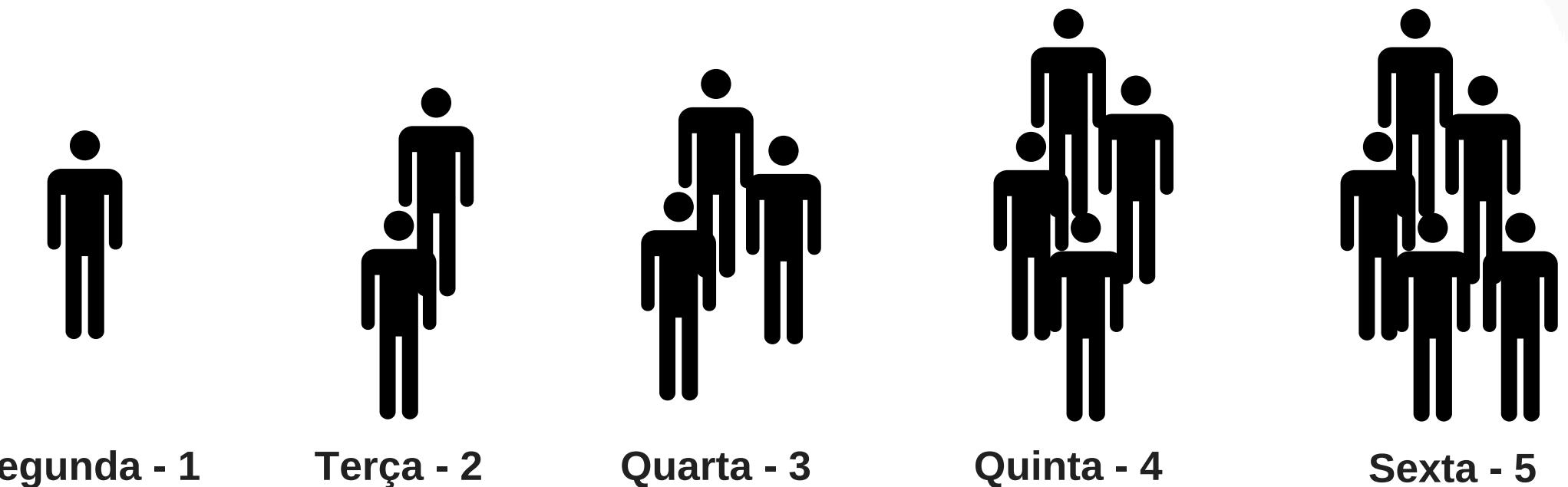
$$3 \times 5 = 15$$

TOTAL: 45 em 5 dias

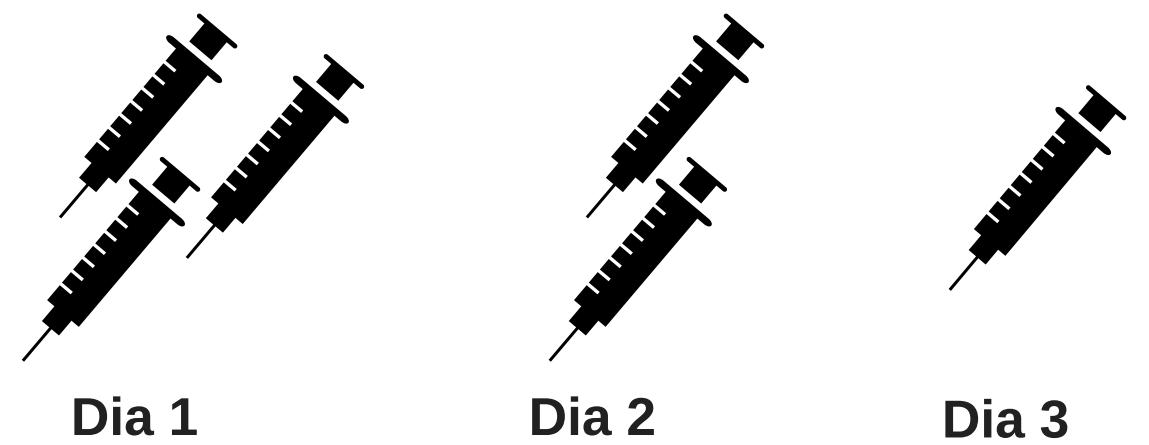
Antes de tudo, uma introdução a convolução:

CASO 2: O tratamento mudou. As doses agora são administradas durante 3 dias, no primeiro dia são 3, no segundo 2, e no último dia 1.

Pacientes distribuidos ao longo da semana:



Medicamento:



Antes de tudo, uma introdução a convolução:

Ficou bem mais complicado calcular, pois não se trata agora mais só de uma questão de multiplicação. Veja:

Pacientes ao longo da semana (cada índice é um dia): [1, 2, 3, 4, 5]

Plano de dosagem: [3, 2 , 1]

Dia 1: 3 comprimidos administrados ao paciente 1

Dia 2: 3 comprimidos administrados aos pacientes 2 ; 2 comprimidos administrados ao paciente 1

Dia 3: 3 comprimidos administrados aos pacientes 3; 2 comprimidos administrados aos pacientes 2; e 1 comprimido administrado ao paciente 1.

Dia 4:

Antes de tudo, uma introdução a convolução:

Uma idéia: Convoluir as funções

O nosso plano de dosagem é chamado de **kernel**.

Um kernel é em termos simples, uma função que usaremos para modificar o comportamento de uma outra função:

Vamos “flipar” a ordem do nosso kernel, pois vamos utilizar a ordem dele para percorrer o grupo de pacientes da direita para a esquerda e precisaremos manter essa ordem para que o primeiro paciente receba as 3 doses primeiro, então:

Doses (flipadas): [1, 2, 3]

Pacientes: [1, 2, 3, 4, 5]

Antes de tudo, uma introdução a convolução:

Convoluindo:

Segunda: Primeiro dia de doses, apenas o 1 tomaria a dose.

Total: 3 un

1 2 3 4 5

1 2 3

Terça: Agora 3 pacientes tomarão a dose, 2-3 e 1-2:

Total: 8 un

1 2 3 4 5

1 2 3

Quarta: A ideia prossegue a mesma, 6 pacientes tomarão a dose: 3-3, 2-2, 1-1

Total: 14 un

1 2 3 4 5

1 2 3

Quinta: O paciente 1 já foi completamente dosado

Total: $12+6+2 = 20$ un

1 2 3 4 5

0 1 2 3

Sexta: Continua

Total: $15+8+3 = 26$ un

1 2 3 4 5

0 0 1 2 3

Sábado: Continua

Total: $10+4 = 14$ un

1 2 3 4 5

0 0 0 1 2 3

Domingo: Último dia, grupo 5 recebe a última dose

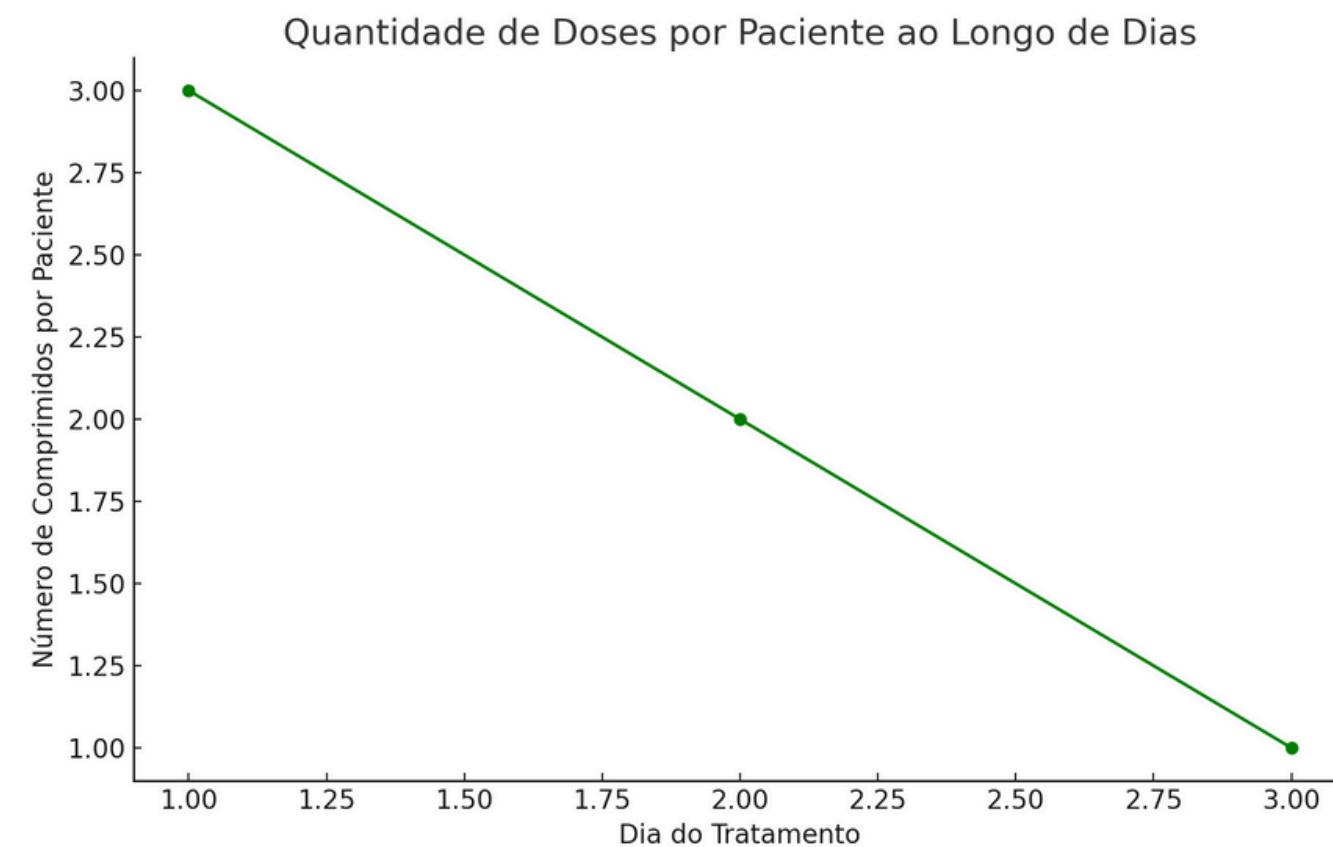
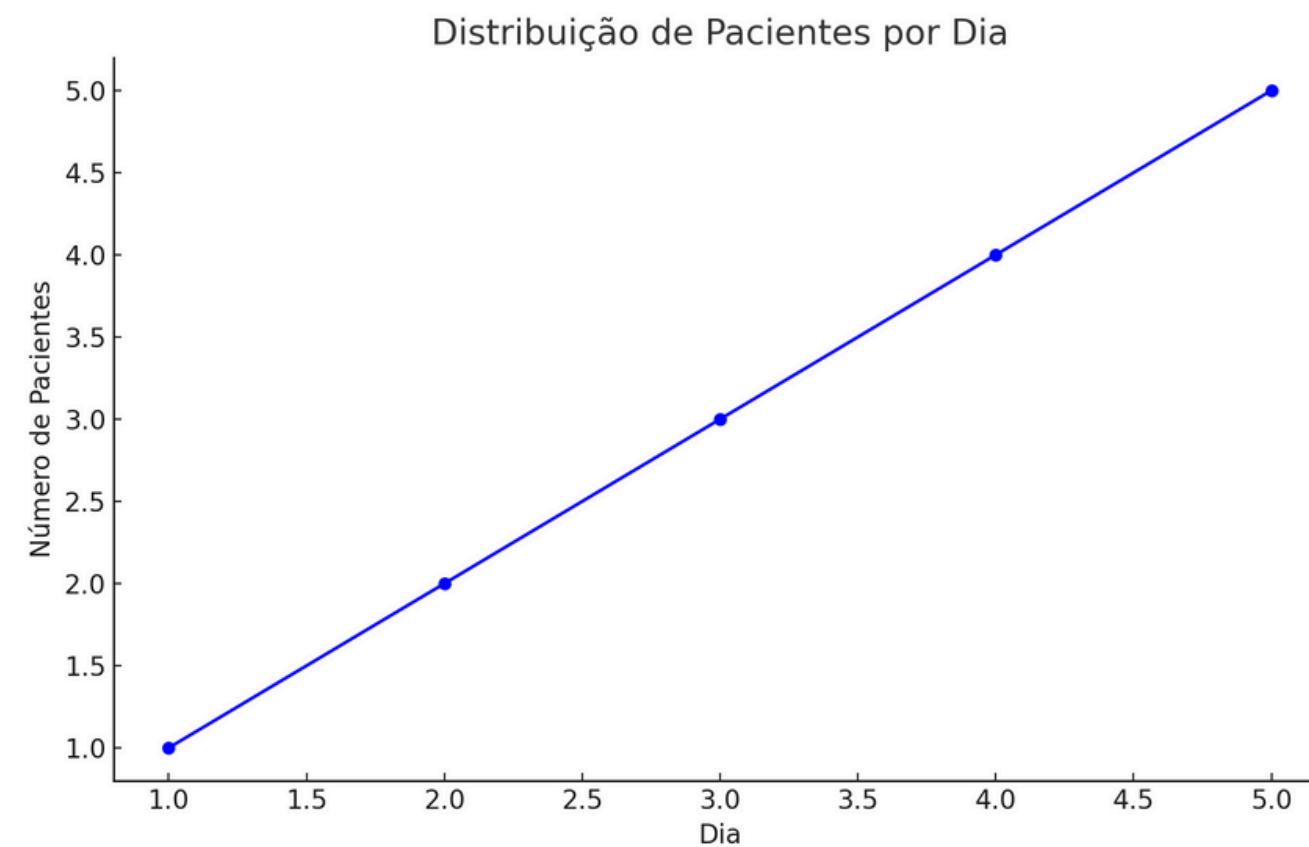
Total: $5 \times 1 = 5$ un

1 2 3 4 5

0 0 0 0 1 2 3

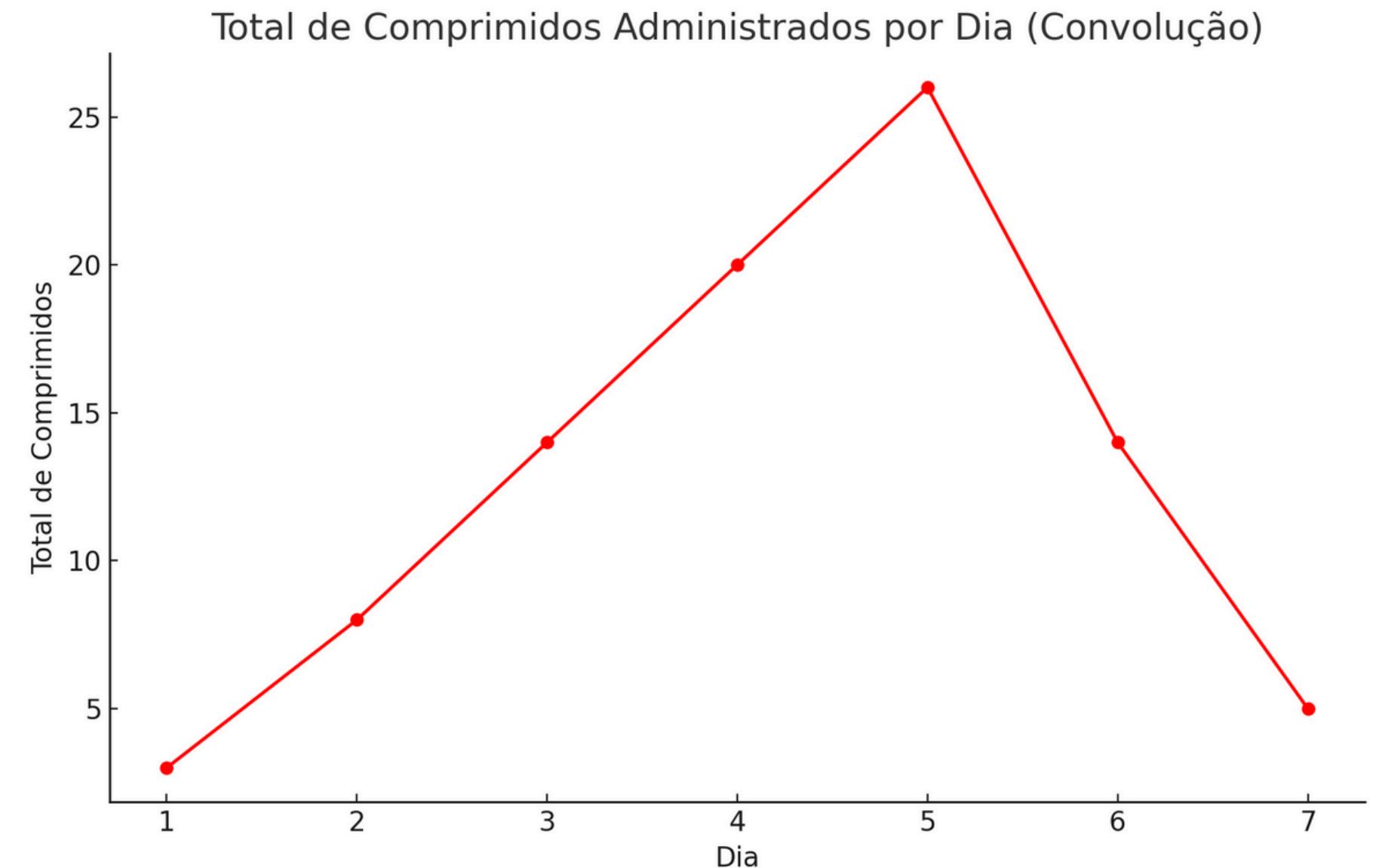
Antes de tudo, uma introdução a convolução:

Convoluindo:



Antes de tudo, uma introdução a convolução:

Convoluindo:



Antes de tudo, uma introdução a convolução:

Resumindo: A convolução é uma operação matemática que combina duas funções para gerar uma terceira, mostrando como a forma de uma função é modificada pela outra.

A convolução pode ser definida da seguinte forma:

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)g(t - \tau) d\tau$$

To convolve a kernel with an input signal:
flip the signal, move to the desired time,
and accumulate every interaction with the kernel

Edge Detection

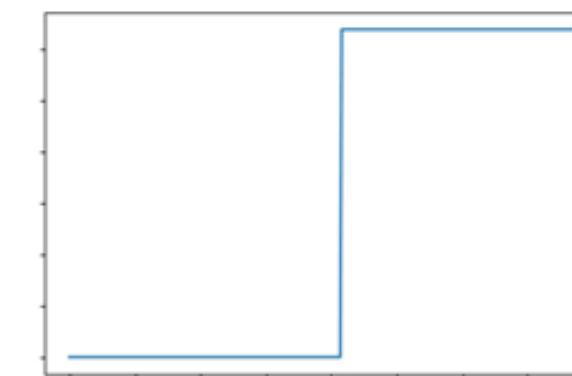
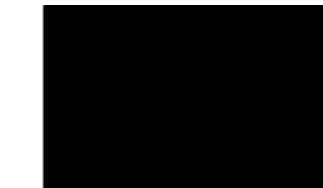
O que é uma borda?

Resposta curta: uma borda pode ser definida como uma mudança abrupta (descontinuidade) de intensidade dentro de uma determinada região da imagem.

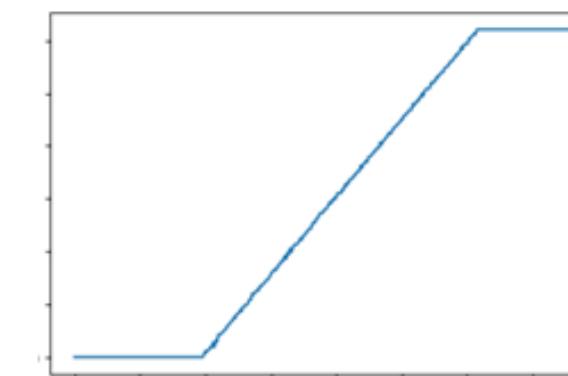


2. Edge Detection

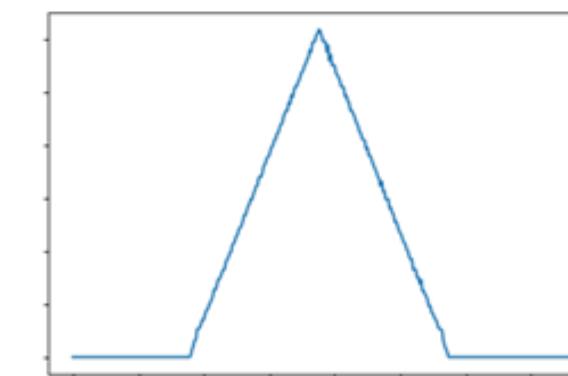
Tipos de Borda



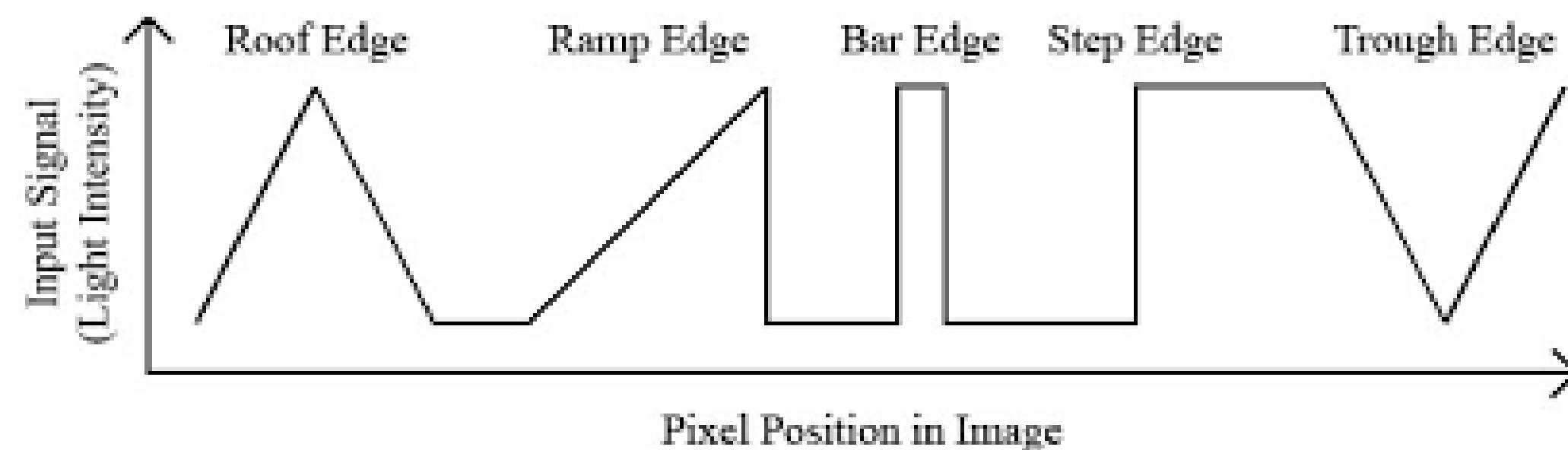
step edge



ramp edge



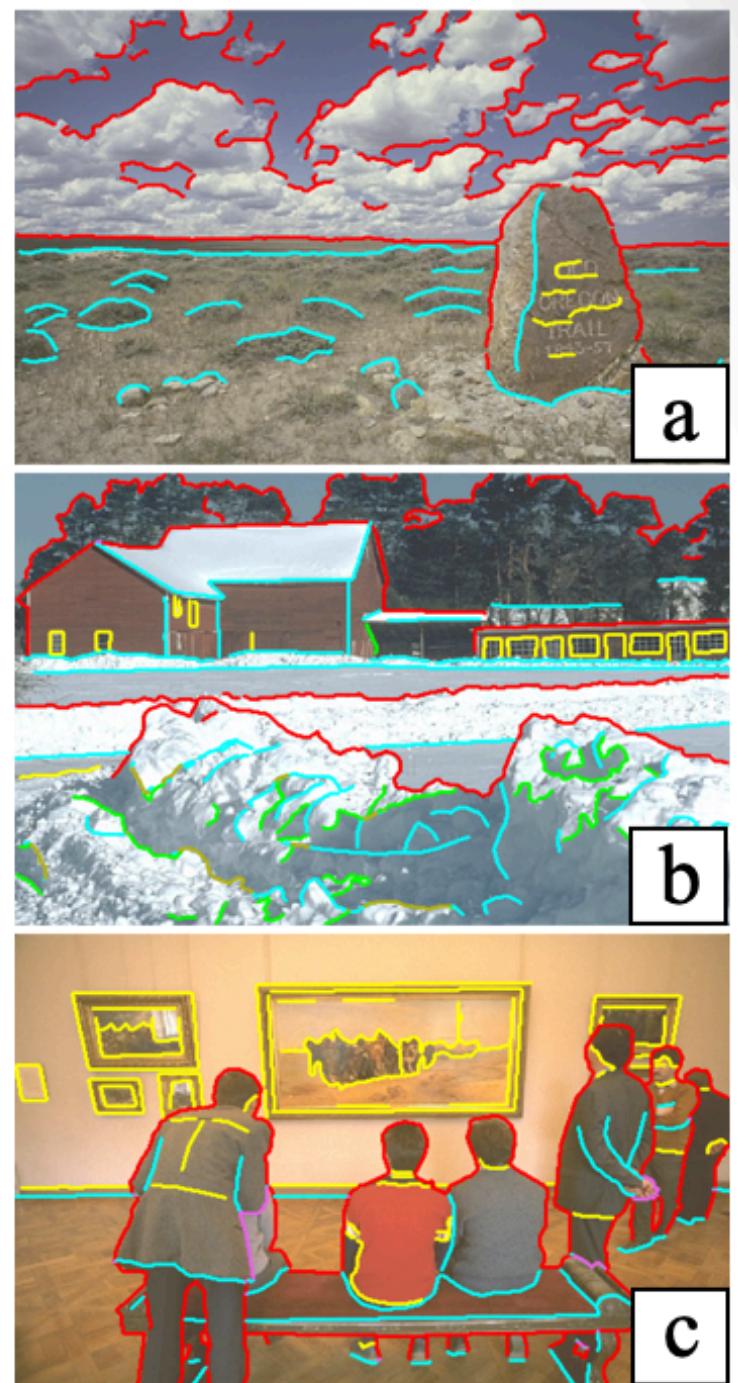
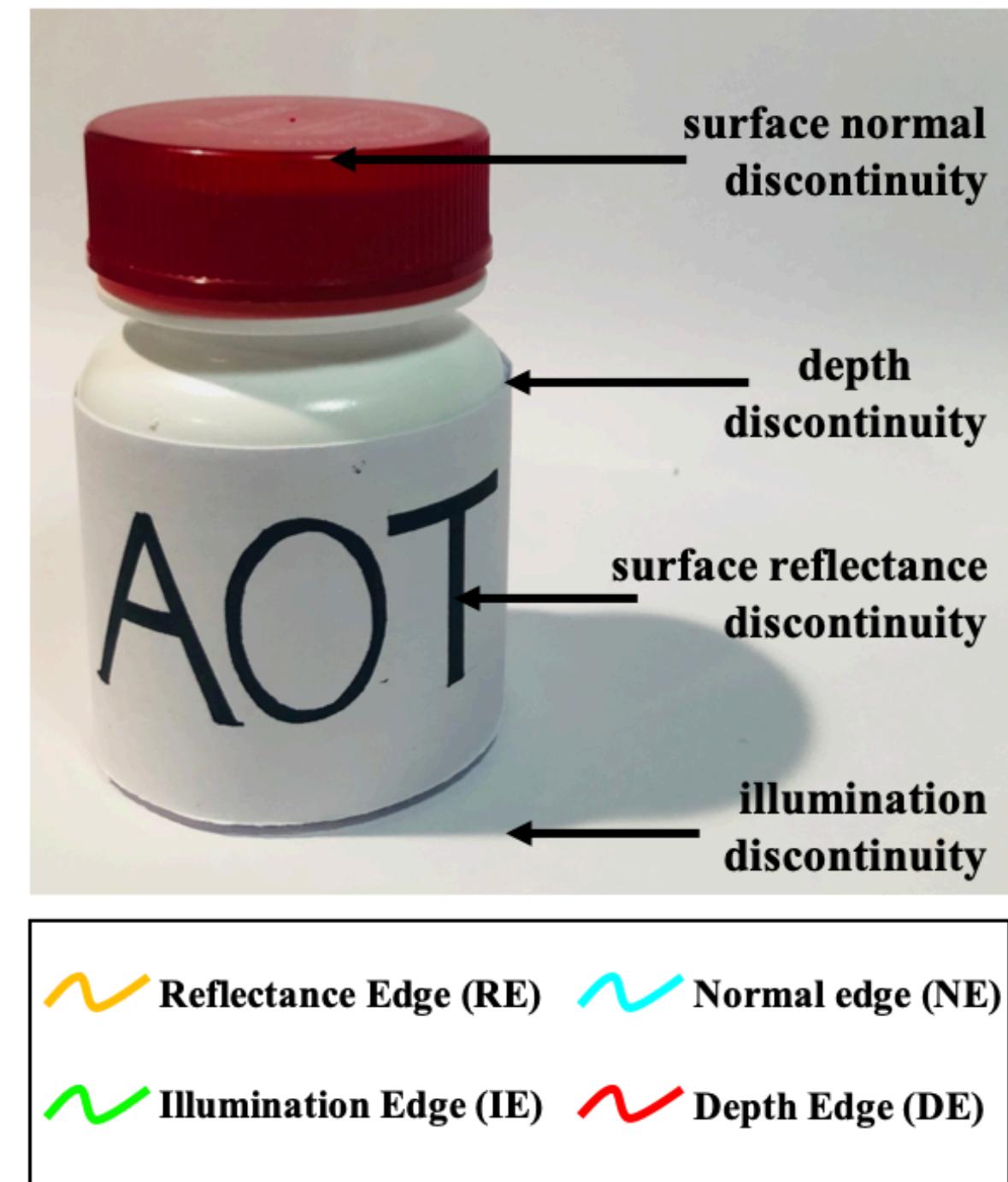
roof edge



2. Edge Detection

O quê causa uma borda?

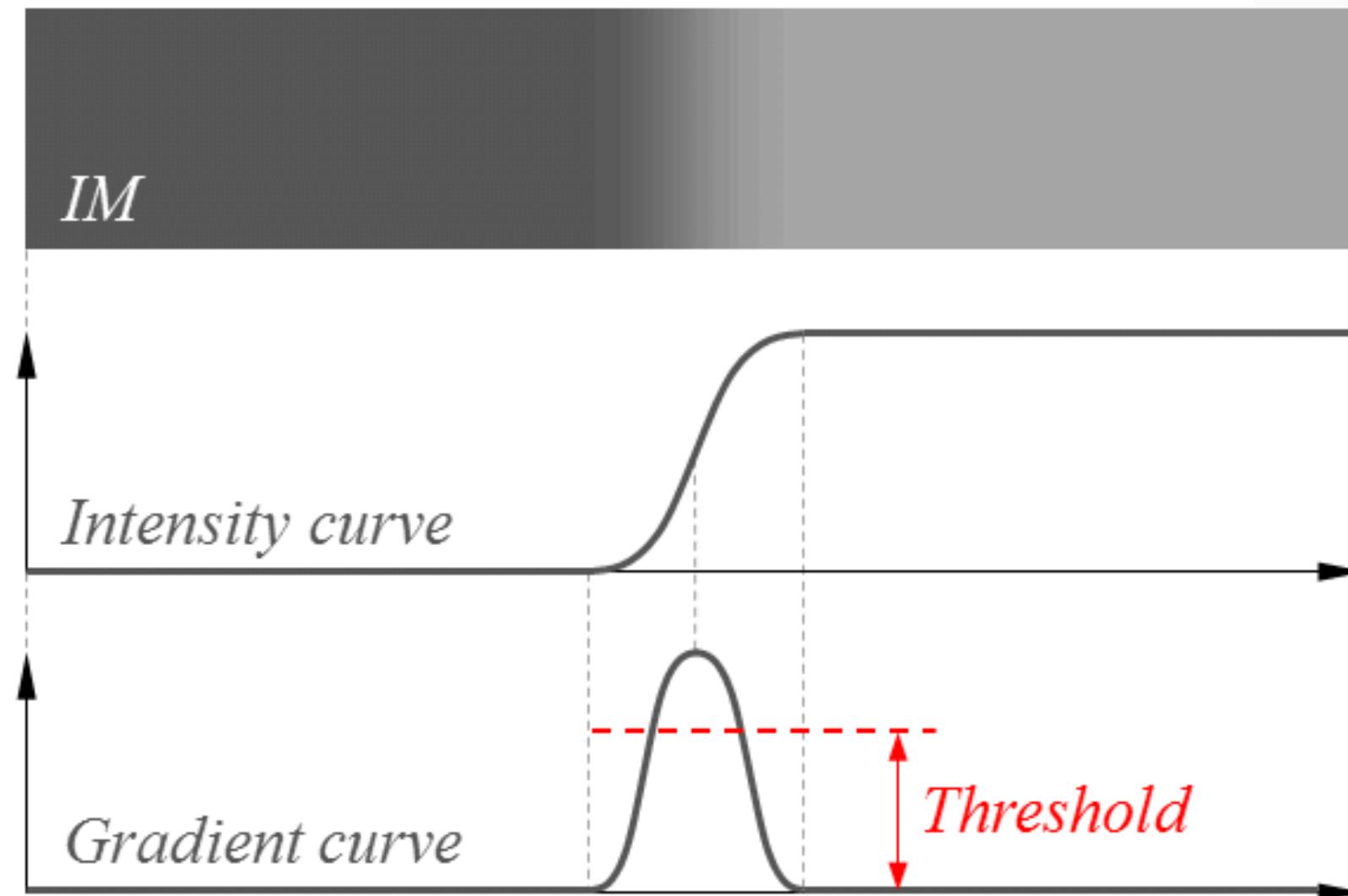
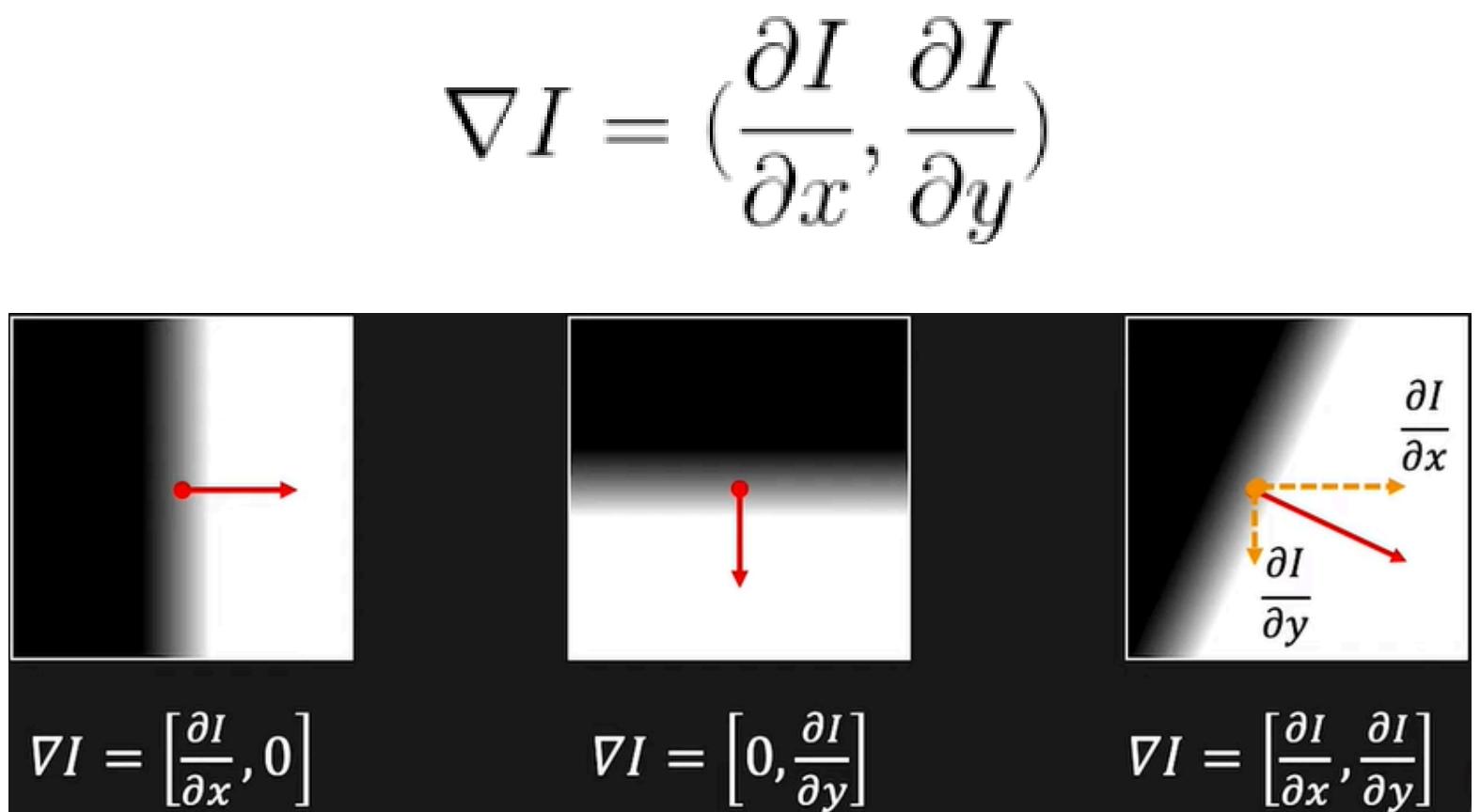
- *Surface Normal Discontinuity*
- *Depth Descontinuity*
- *Surface Reflection Discontinuity*
- *Illumination Descontinuity*



2.1 Gradient Edge Detection

Como funciona uma detecção de borda usando o Gradient?

- Análise de variação pixel a pixel através de um vetor
- Geração de gráfico a partir de uma máscara convolucional
- Análise da primeira derivada
- Ajuste de threshold (limite)



2.1 Gradient Edge Detection

Magnitude e Orientação

- Magnitude: intensidade da borda

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

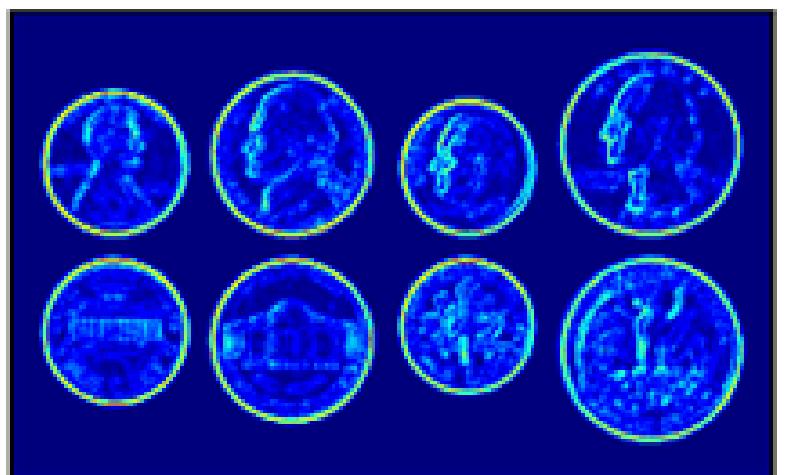
- Orientação: orientação da borda em relação aos eixos

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

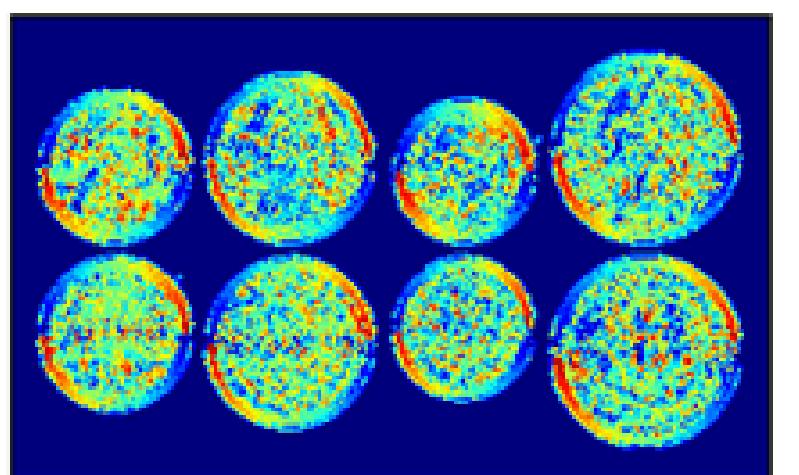
Grayscale



Gradient Magnitude



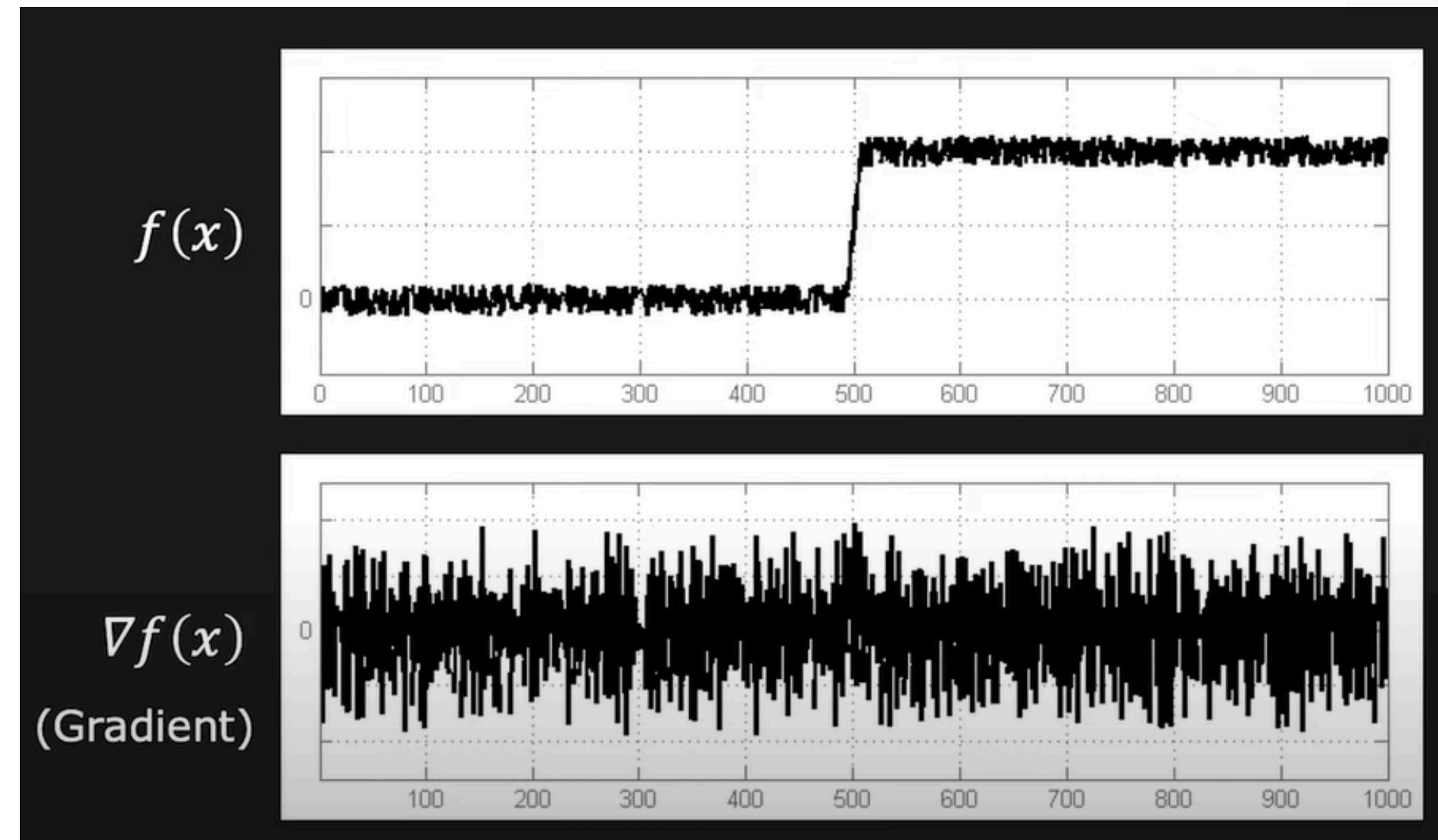
Gradient Orientation [0, 180]



2.1 Gradient Edge Detection

Ruídos

- Cria picos de intensidade que não condizem com a imagem
- Atrapalham no processo de análise de derivada
- Cria falsos positivos



Filtros de Detecção de Bordas

O que é: O filtro de detecção de bordas é como um primeiro passo para calcular a derivada da intensidade de uma imagem.

Para entender melhor:

Derivada: Na matemática, a derivada representa a taxa de variação de uma função em um determinado ponto.

Bordas: As bordas em uma imagem são justamente os pontos onde essa variação é mais acentuada, ou seja, onde a derivada é maior.

Filtro: O filtro de detecção de bordas é um operador matemático que calcula uma aproximação da derivada da imagem em cada pixel.



Filtros de Detecção de Borda

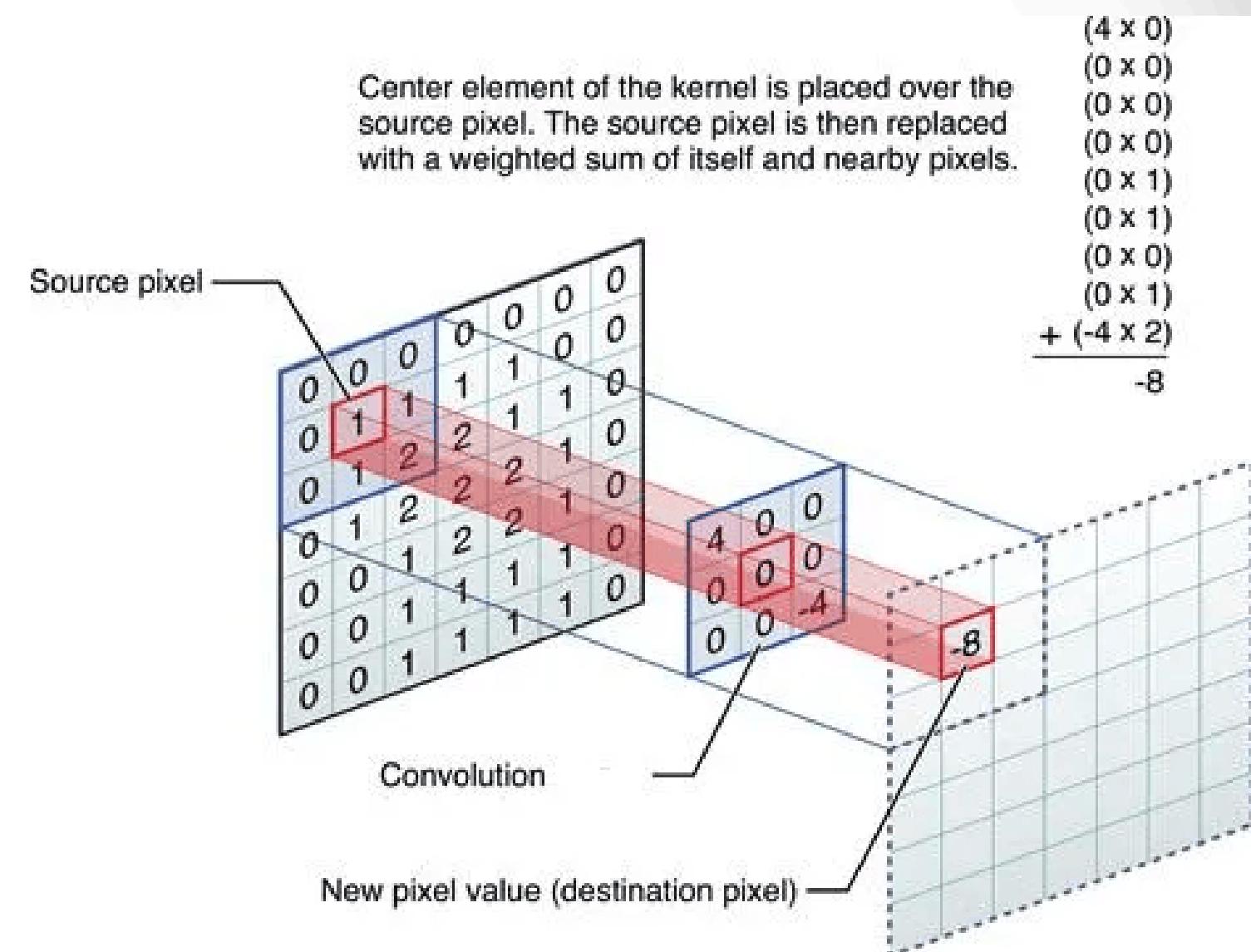
Máscara de Gradiente:

| Gradient | Roberts | Prewitt | Sobel (3x3) | Sobel (5x5) |
|---------------------------------|---|--|--|---|
| $\frac{\partial I}{\partial x}$ | $\begin{matrix} 0 & 1 \\ -1 & 0 \end{matrix}$ | $\begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix}$ | $\begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$ | $\begin{matrix} -1 & -2 & 0 & 2 & 1 \\ -2 & -3 & 0 & 3 & 2 \\ -3 & -5 & 0 & 5 & 3 \\ -2 & -3 & 0 & 3 & 2 \\ -1 & -2 & 0 & 2 & 1 \end{matrix}$ |
| $\frac{\partial I}{\partial y}$ | $\begin{matrix} 1 & 0 \\ 0 & -1 \end{matrix}$ | $\begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{matrix}$ | $\begin{matrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{matrix}$ | $\begin{matrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 3 & 5 & 3 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ -2 & -3 & -5 & -3 & -2 \\ -1 & -2 & -3 & -2 & -1 \end{matrix}$ |

Filtros de Detecção de Bordas

Convolução:

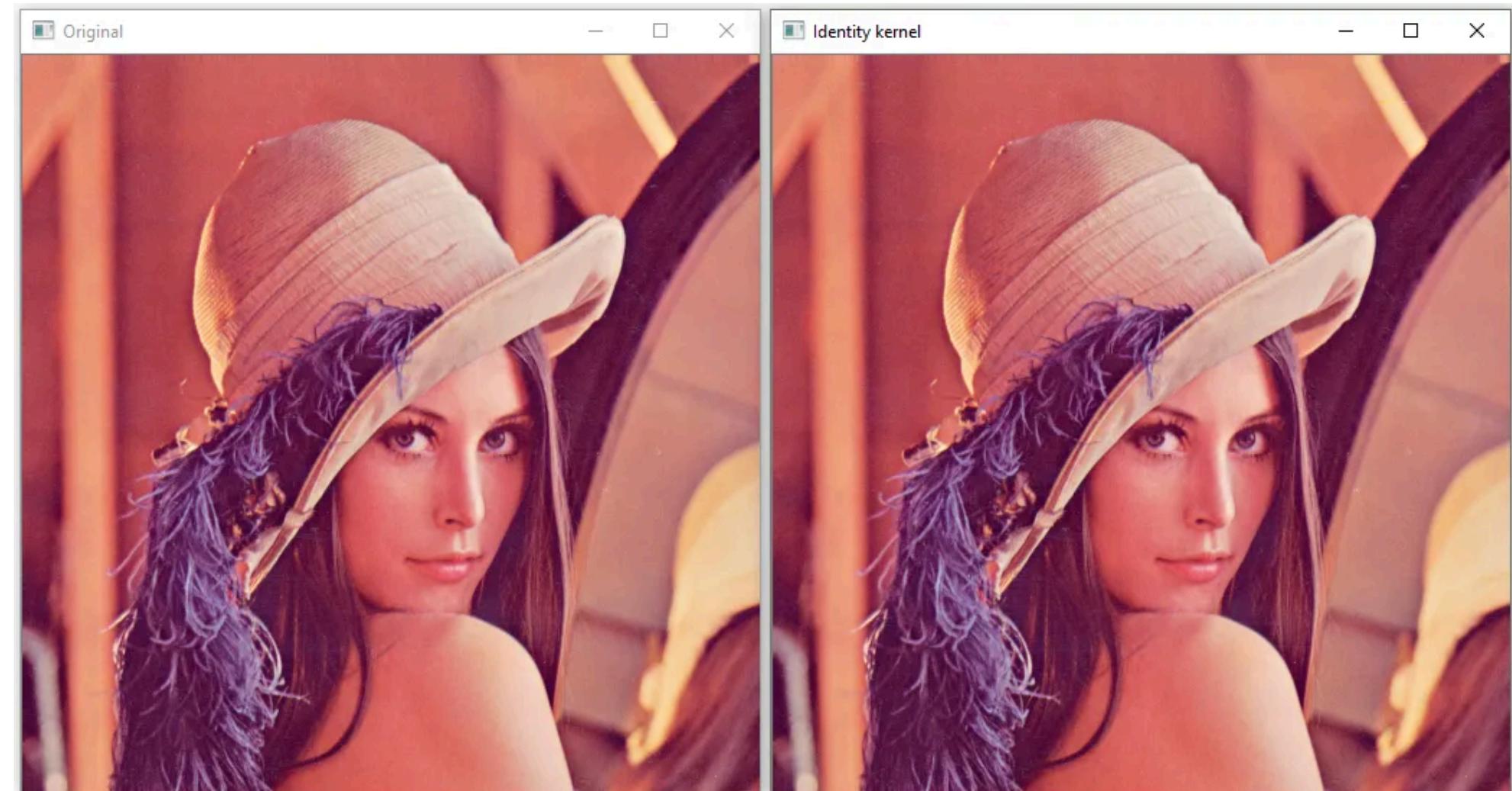
a convolução é uma operação matemática que combina duas funções para gerar uma terceira. Essa nova função representa uma espécie de "média ponderada" das duas funções originais, onde os pesos são determinados por uma função chamada kernel.



Filtros de Detecção de Bordas

A saída do processo de convolução muda com os valores de kernel em mudança.
Por exemplo, um Identity Kernel mostrado abaixo, quando aplicado a uma imagem por meio de convolução, não terá efeito na imagem resultante. Cada pixel manterá seu valor original.

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$



Filtros de Detecção de Bordas

Um Sharpen Kernel como este, quando aplicado a uma imagem por meio de convolução, terá um efeito de nitidez de imagem na imagem resultante. Os valores precisos podem ser personalizados para níveis variados de nitidez, conforme mostrado na figura a seguir.

```
[ [-1 -1 -1]
  [-1  9 -1]
  [-1 -1 -1] ]
```

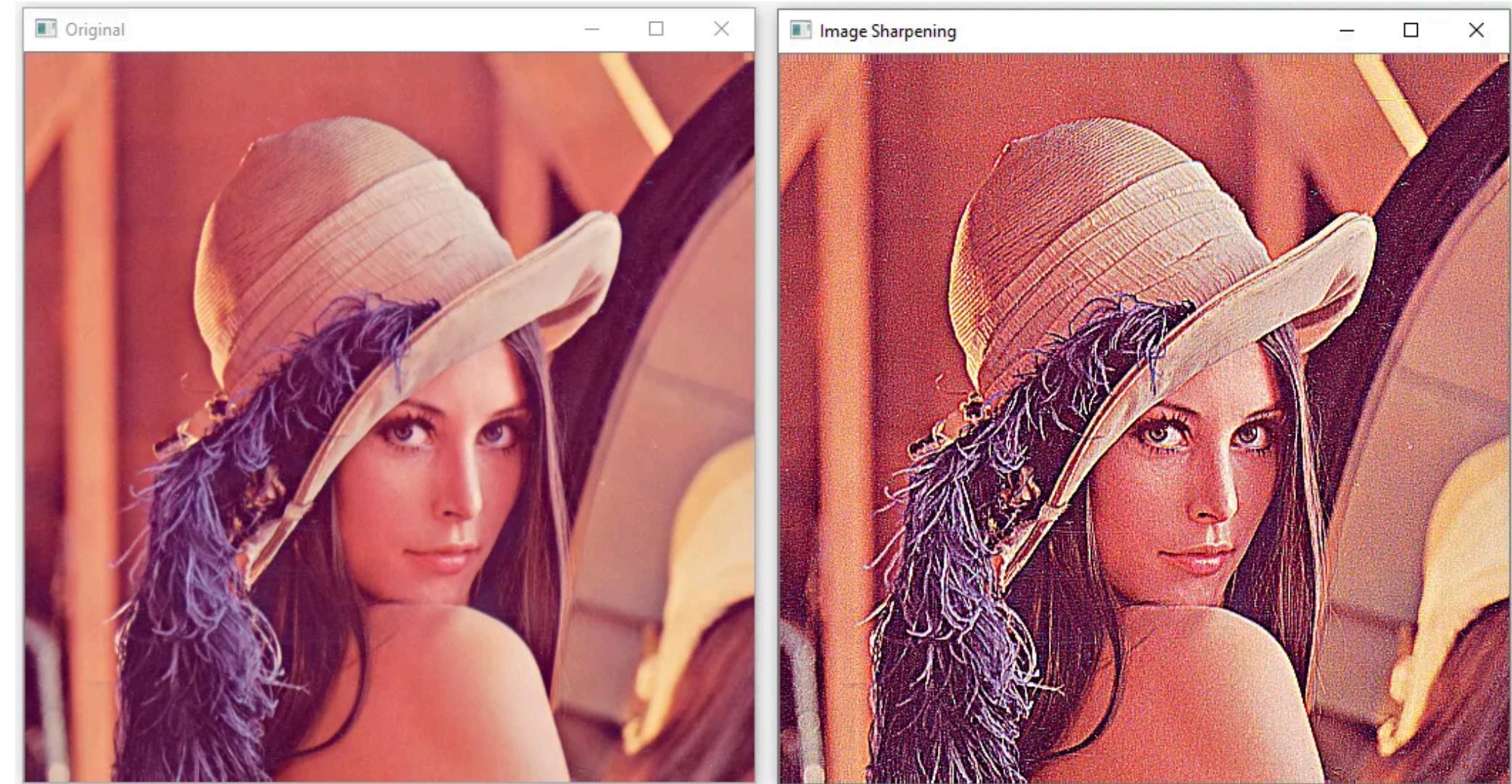


Imagen original (esquerda) e imagen após aplicar filtro de nitidez de tamanho 3x3 (direita)

Filtros de Detecção de Bordas

O Kernel de Desfoque Gaussiano, quando aplicado a uma imagem por meio de convolução, aplicará um efeito de Desfoque Gaussiano à imagem resultante.

$$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Or

$$\frac{1}{256} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Kernel de Desfoque Gaussiano

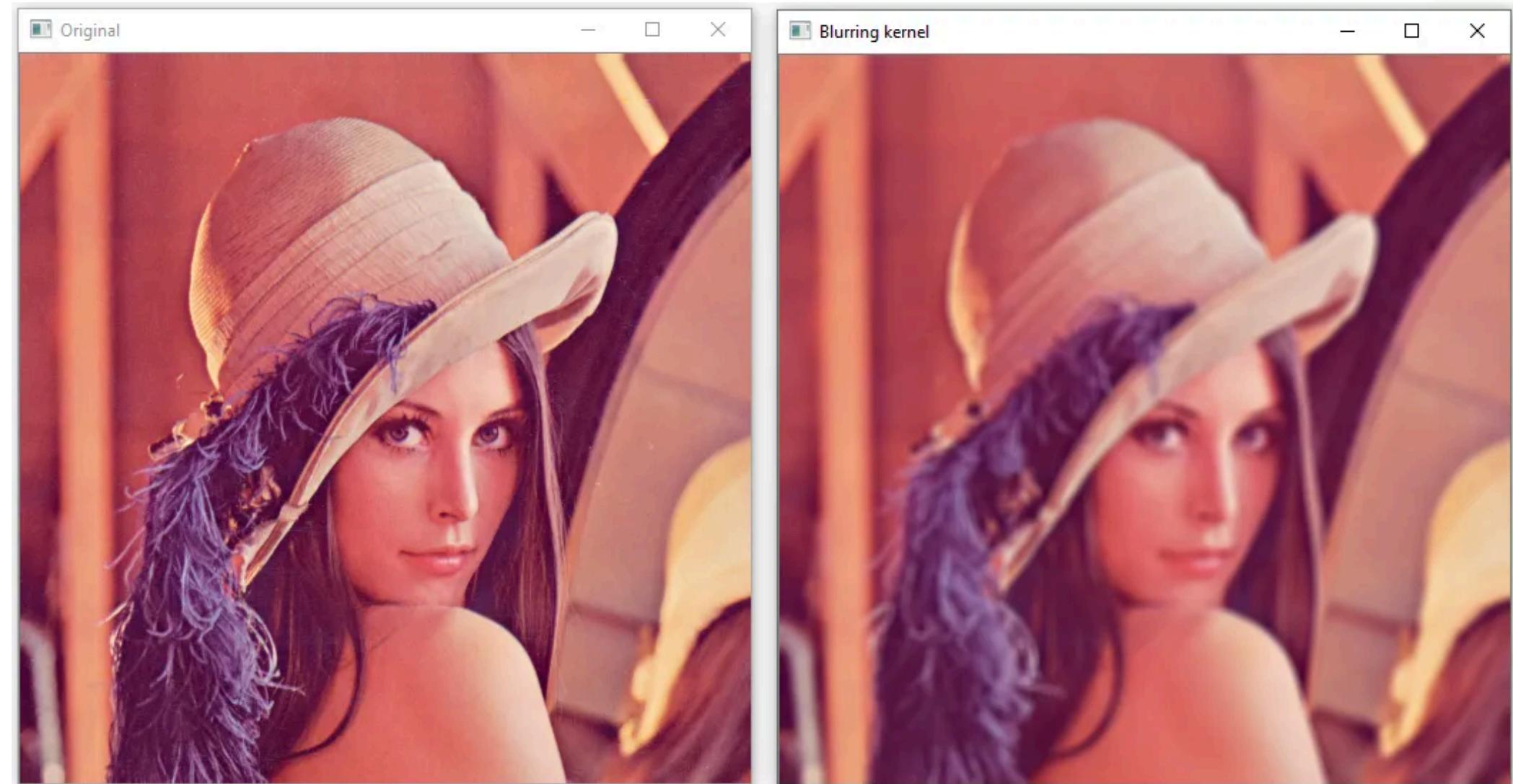


Imagen original (esquerda) e imagem após aplicar filtro de desfoque de tamanho 7x7 (direita)

Filtro Sobel

O que o Filtro Sobel faz?

Calcula o gradiente em duas direções (horizontal e vertical) e combina os resultados para obter a magnitude do gradiente.

Matematicamente, este operador utiliza duas matrizes 3×3 que são convoluídas com a imagem original para calcular aproximações das derivadas - uma para as variações horizontais e uma para as verticais. Sendo:

$$\mathbf{G}_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * \mathbf{A} \quad \text{e} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

Portanto a magnitude, \mathbf{G} , e a direcção, do gradiente são dados por:

$$\mathbf{G} = \sqrt{\mathbf{G}_x^2 + \mathbf{G}_y^2} \quad \Theta = \arctan\left(\frac{\mathbf{G}_y}{\mathbf{G}_x}\right)$$

Filtro Sobel

O filtro irá calcular um valor para cada pixel, indicando a magnitude da borda naquele ponto. Esse valor é proporcional ao gradiente da imagem.



Aplicação do filtro em uma imagem:

Filtro Sobel

Resultados:



G(x)



G(y)

Filtro Sobel

Resultados:



Imagen obtida através de magnitudes das coordenadas polares dos vetores de intensidade

Filtro Canny

É um dos algoritmos mais sofisticados, que combina vários estágios para obter uma detecção de bordas robusta e precisa.

O primeiro passo da execução do Canny Edge Detector é a execução do filtro Gaussiano

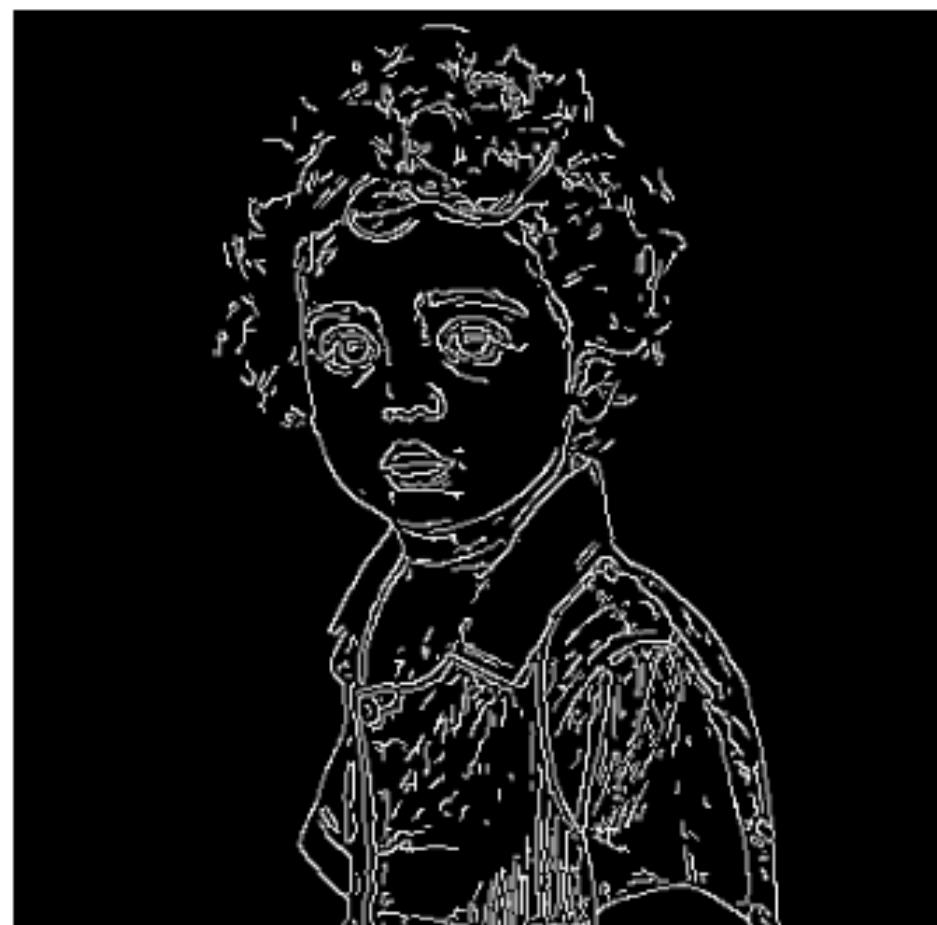
O segundo passo é a obtenção das intensidades e gradientes da imagem que justamente é o produto da execução do filtro Sobel.

O terceiro passo é conhecido como Non-maximum Suppression, que consiste em remover pixels não desejados para o processamento, causando um afinamento das bordas

Por último, há um algoritmo que precisa ser executado para remover bordas remanescentes por um processo de corte de intensidades, conhecido como Hysteresis Thresholding.



Non-max Suppression



Final Output

Filtro Canny

Filtro gaussiano

Um filtro que borre a imagem pode parecer contra-intuitivo no contexto em que é necessário obter detalhes presentes nela. Entretanto a vantagem deste tipo de filtro aqui é a de remover detalhes excessivos, os ruídos.



Comparativo entre imagens. (a) Acima a imagem original. (b) Abaixo, à esquerda, após realizar o Canny, sem borramento. (c) Abaixo, à direita, com o filtro de borramento gaussiano

Filtro Canny

supressão de não-máximos

A supressão de não-máximos é uma etapa essencial no processamento de bordas em algoritmos de detecção, como o filtro Canny. O objetivo dessa técnica:

Afinar as bordas;

Reter apenas os pixels que representam um máximo local ao longo da direção do gradiente;

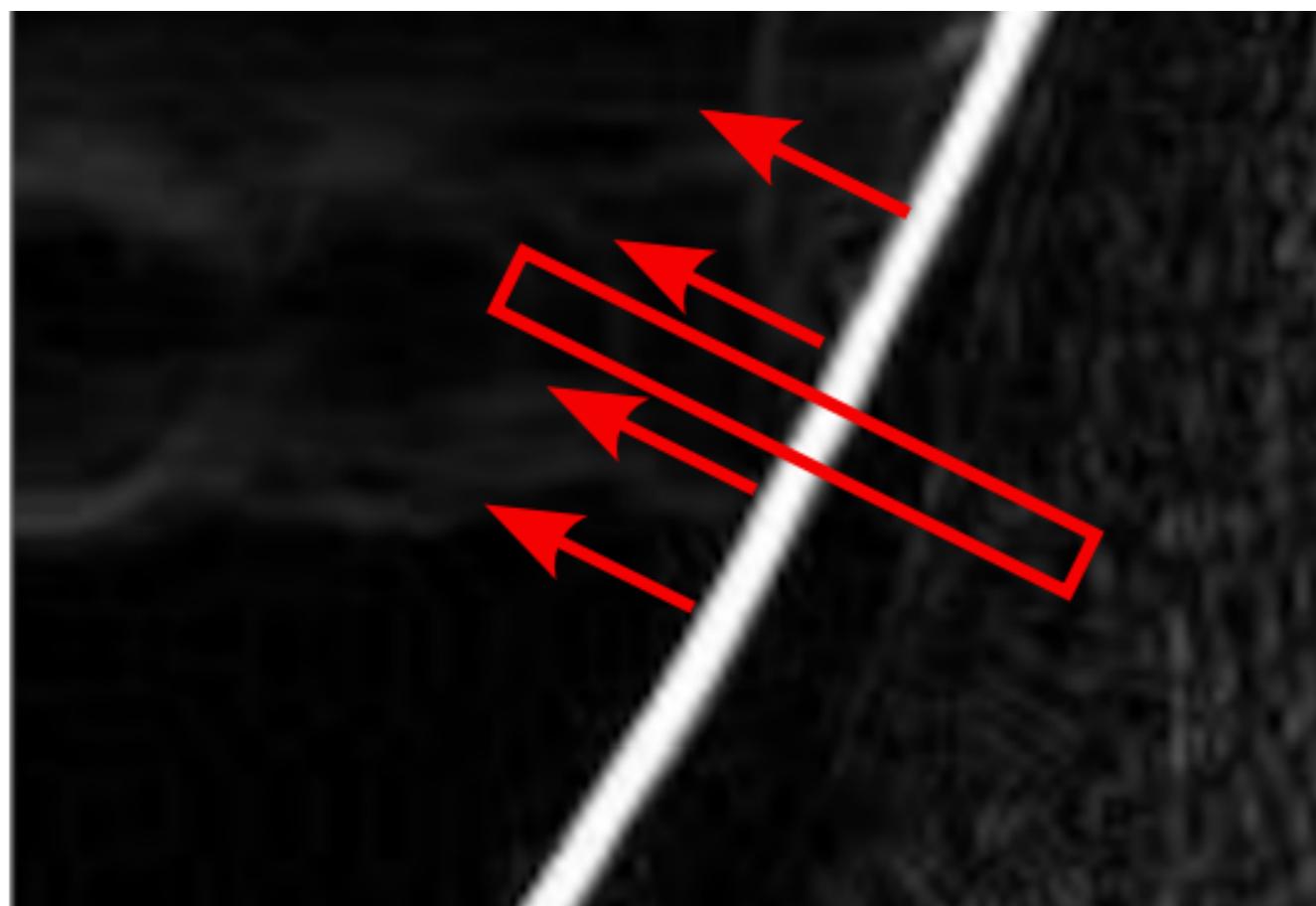
Eliminar espessuras indesejadas e a definir contornos mais precisos.

Após calcular a magnitude e a direção do gradiente para cada pixel, essa etapa verifica se cada pixel representa um pico de intensidade (máximo local) ao longo da linha do gradiente.

Filtro Canny

supressão de não-máximos

Para visualizar o efeito da supressão, imagine um gradiente de intensidades ao longo de uma borda inclinada. Sem a supressão de não-máximos, a borda apareceria "espessa", pois vários pixels ao redor têm intensidades próximas. Com a supressão, apenas os pixels nos pontos de máximo local são retidos, resultando em uma borda mais fina e precisa.



Seleção de pixels segundo direção do gradiente

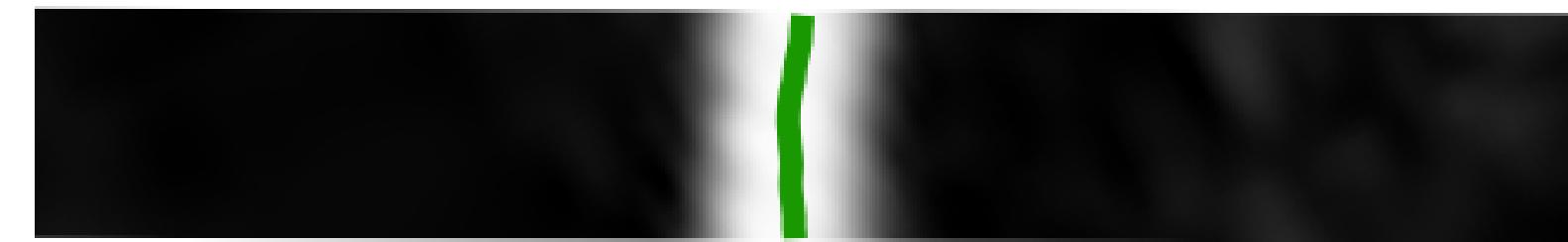


Imagen extraída da figura a esquerda. Linha de seleção para pixels mais intensos em verde



Resultado do afinamento

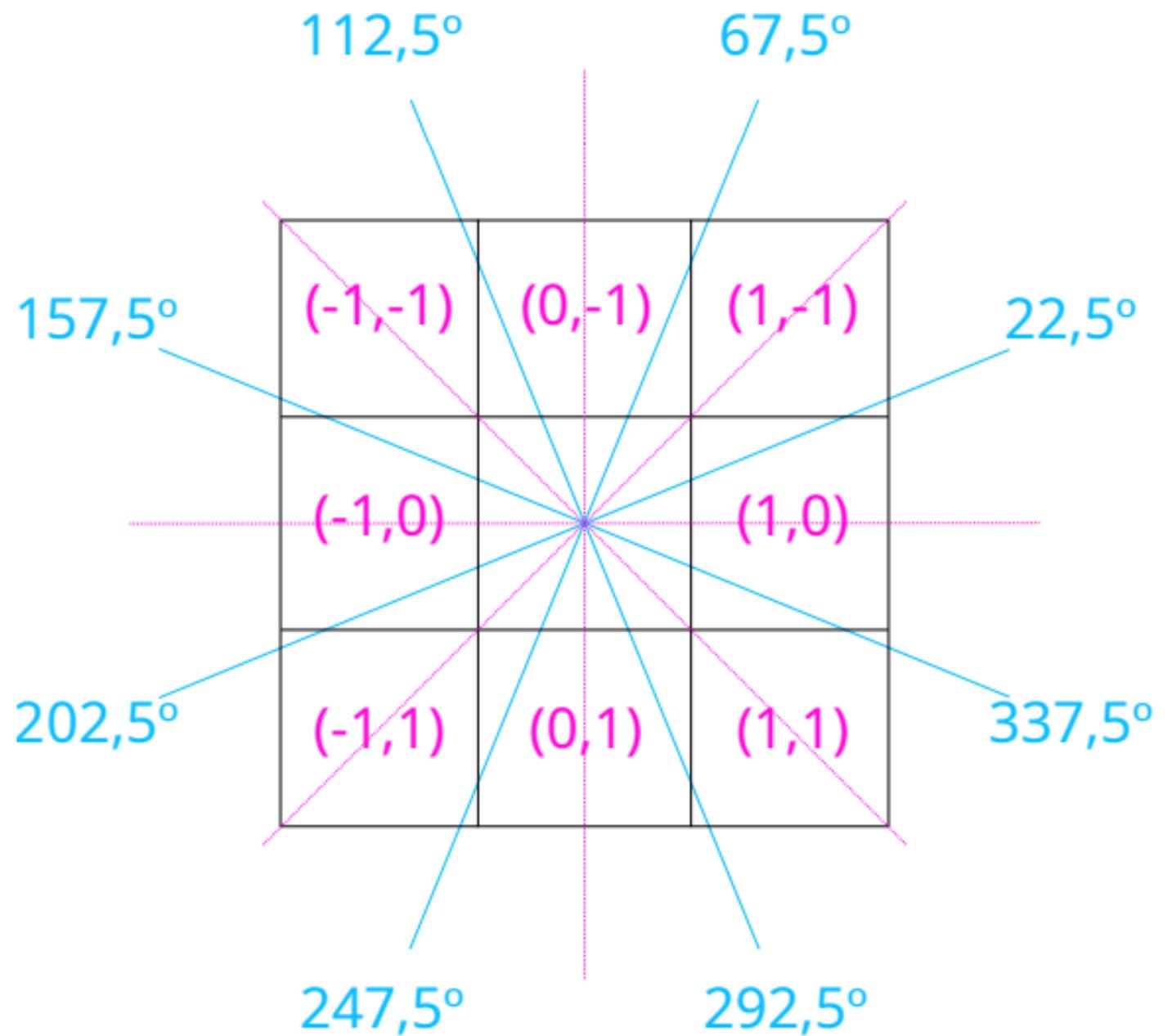
Filtro Canny

supressão de não-máximos

Cálculo da Direção e Magnitude do Gradiente

Interpolação Direcional

Comparação com os Vizinhos



Filtro Canny

Supressão de não-máximos

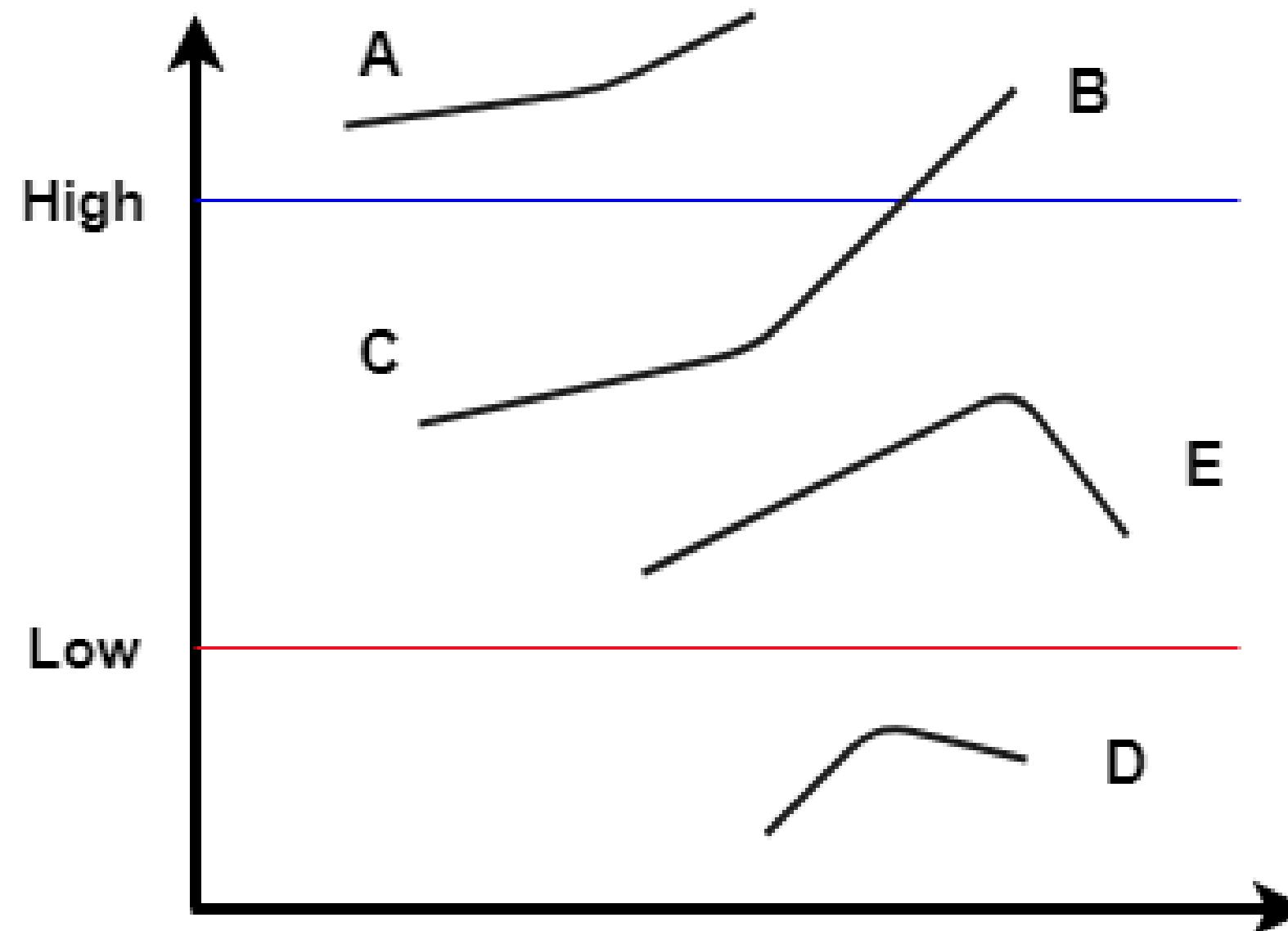


Resultado do afinamento na imagem

Filtro Canny

Hystereses thresholding

Existe uma questão a ser considerada: como limpar a imagem de modo que apenas as bordas mais intensas, ou seja, as bordas de interesse, permaneçam, enquanto todo o resto é removido?



Filtro Canny

Hystereses thresholding



Imagen proveniente da etapa de afinamento, após multiplicação das intensidades por 50, antes do filtro Hysteresis Thresholding

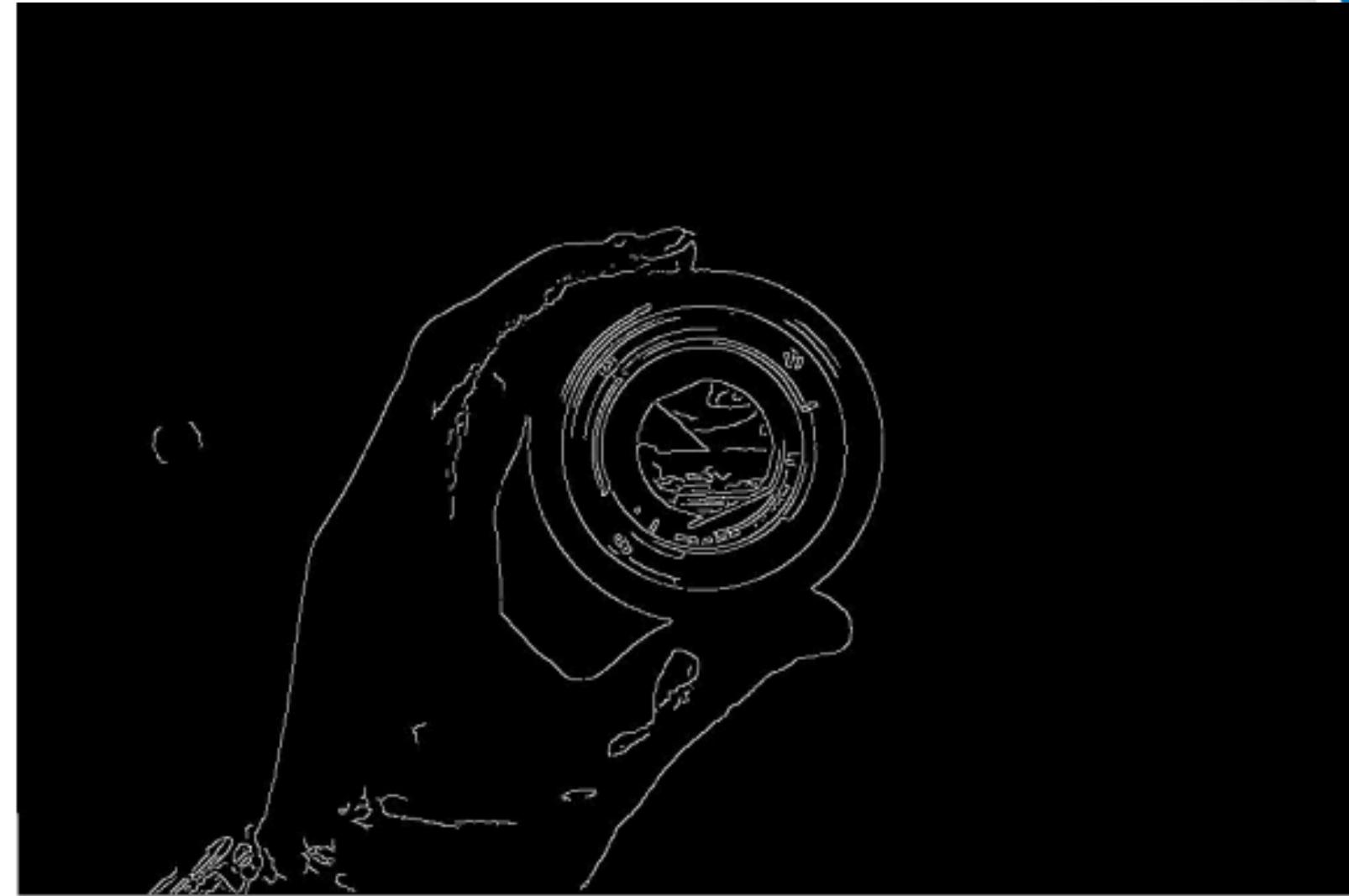


Imagen após filtro Hysteresis Thresholding e com multiplicação das intensidades por 50 realizada

Corner Detection

Algoritmos de detecção de cantos. Onde um canto é dado como o ponto de intersecção de N-bordas, ou seja é um ponto onde há múltiplas variações na intensidade da imagem em direções diferentes por conta de bordas que se encontram.

Feature points



Corner Detection - Características

A importância dos cantos na imagem está no fato de que eles são **muito específicos** na imagem. Isso quer dizer que **há muita variação em torno deles** (proveniente das bordas), e portanto, são muito fáceis de serem **rastreados e diferenciados**.

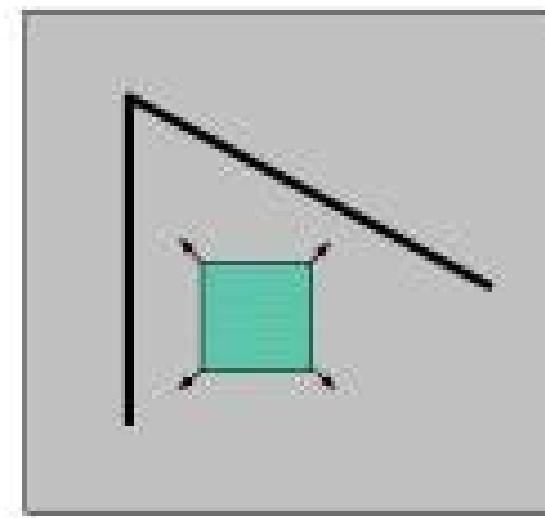
Outras características importantes dos cantos nas imagens:

- Robustos à transformações (como mudança de iluminação, rotações, mudanças na escala)
- Invariância a pequenas distorções e ruídos
- Amplamente utilizados em algoritmos de correspondência e reconhecimento, como em aplicações de visão computacional que exigem rastreamento de objetos, reconstrução 3D e mapeamento visual.

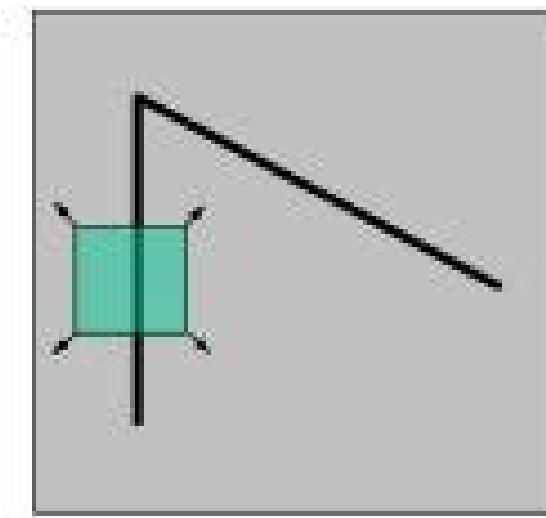
Corner Detection - Harris Corner

Baseado na análise das mudanças nos gradientes de intensidade, tornando-se útil para identificar regiões onde há uma variação acentuada em duas direções ortogonais.

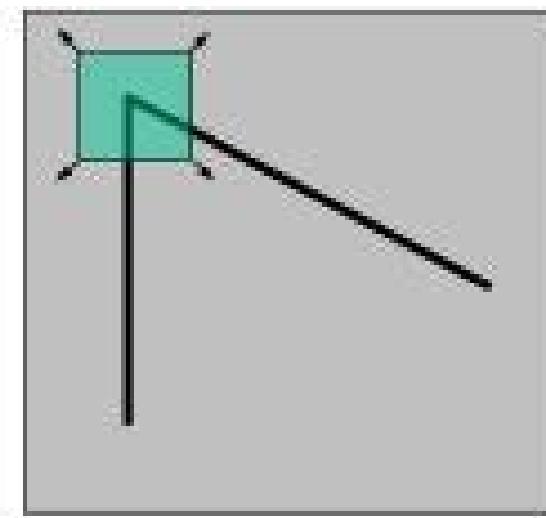
A ideia é **detectar regiões onde há uma mudança significativa na intensidade em todas as direções ao redor de um ponto** — essas regiões correspondem aos "cantos" ou pontos de alta curvatura.



"flat":
no change in all
directions



"edge":
no change along the
edge direction



"corner":
significant change in
all directions

Corner Detection - Harris Corner

Etapas:

1. Primeiro, para cada pixel da imagem, calculamos os **gradientes de intensidade** nas **direções x e y**. Esses gradientes indicam como a intensidade da imagem muda em cada direção e são fundamentais para identificar variações acentuadas em duas direções.

Os gradientes **I_x** e **I_y** podem ser calculados usando **filtros de Sobel**, I é a intensidade da imagem

4. Após calcular R para cada pixel, obtemos uma matriz de respostas que destaca potenciais cantos. No entanto, a detecção inicial pode incluir **múltiplos cantos próximos**, então é preciso aplicar a **supressão de não-máximos** para manter apenas os pontos de maior valor R localmente.

2. Para cada pixel, o algoritmo constrói uma matriz **M de autocorrelação** que encapsula as informações dos gradientes locais ao redor do pixel.

Essa matriz é utilizada para **quantificar a variação da intensidade** nos arredores do pixel em todas as direções.

3. Com a matriz M calculada, precisamos **definir uma resposta** que indique se um ponto é um canto. Essa resposta é calculada pela **Resposta de Harris (R)**

A interpretação da resposta de Harris R é:

R alto: ponto com alta curvatura em ambas as direções, indicando um canto.

R baixo ou próximo de zero: ponto que não é uma borda.

R negativo: ponto onde há uma região plana.

Corner Detection - Harris Corner

Etapas:

3. To detect corners in an image, the Harris corner detector can be used. Fig. 3 shows an image $I(x, y)$.
 $z = 180 + a + b + c + d$.

| | | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|-----|-----|-----|-----|-----|-----|
| y | 0 | 100 | 100 | 100 | 100 | 100 | 100 |
| | 1 | 100 | 100 | 100 | 100 | 100 | 100 |
| | 2 | 100 | 100 | z | z | 100 | 100 |
| | 3 | 100 | 100 | z | z | 100 | 100 |
| | 4 | 100 | 100 | z | z | 100 | 100 |
| | 5 | 100 | 100 | 100 | 100 | 100 | 100 |

Fig. 3

- (a) Using the operators, $S_x = [-1 \ 0 \ 1]$ and $S_y = [-1 \ 0 \ 1]^T$, compute the horizontal gradients I_x and the vertical gradients I_y of the image $I(x, y)$.
- (b) Determine the Hessian matrix H , as given below, at the pixel positions (1, 1) and (1, 3).

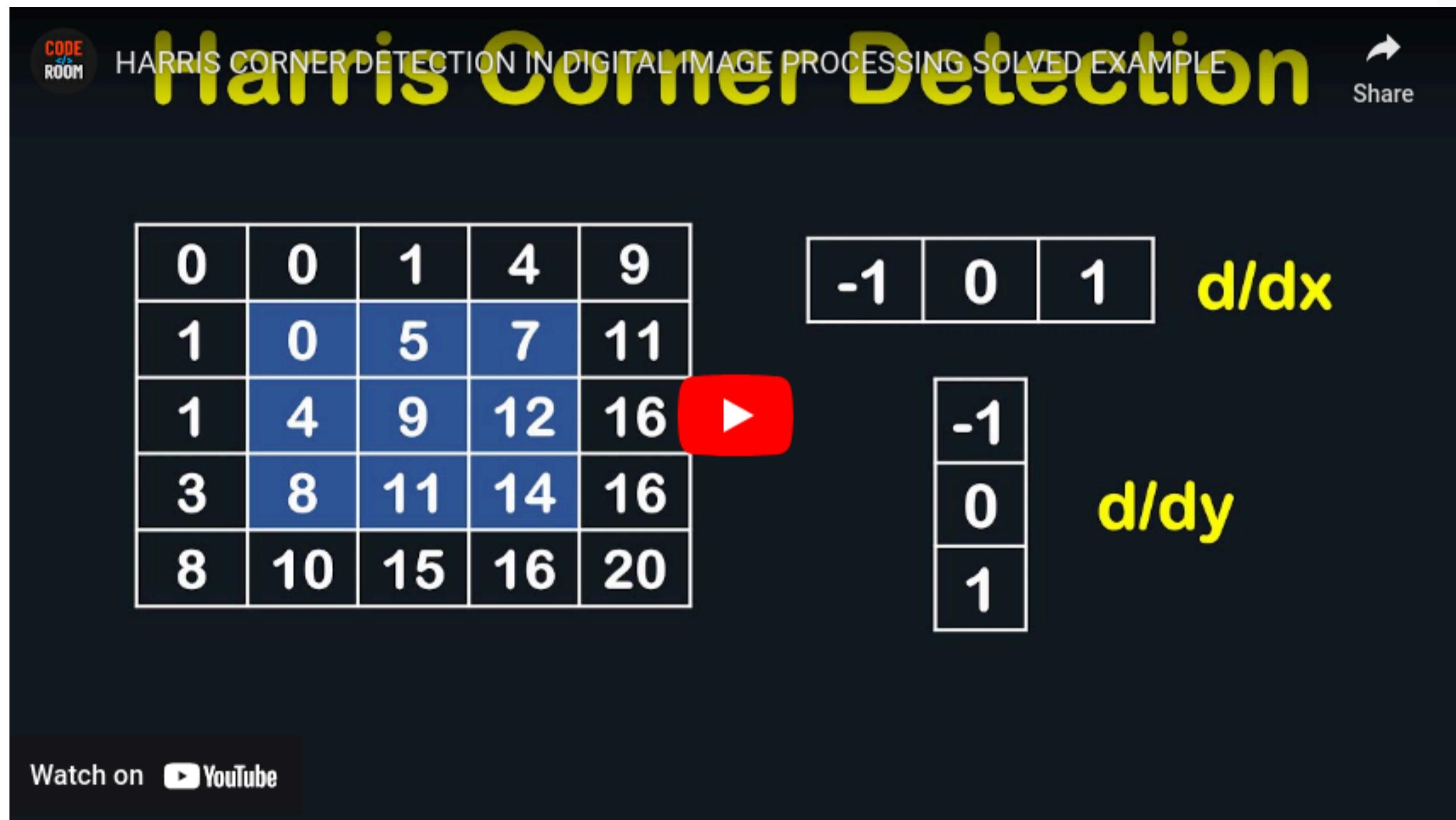
$$H = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}.$$

- (c) Compute the following response function at the two pixel positions (1, 1) and (1, 3):

$$R = \det(H) - k(\text{trace}(H))^2,$$

where $k = 0.05$, and the window size used is 3×3 .

Corner Detection - Harris Corner



Corner Detection - Harris Corner

Vantagens e Limitações do Detector de Harris

Vantagens:

- **Eficiência:** O método de Harris é relativamente rápido.
- **Invariância a Rotação:** O detector é invariante a rotações.

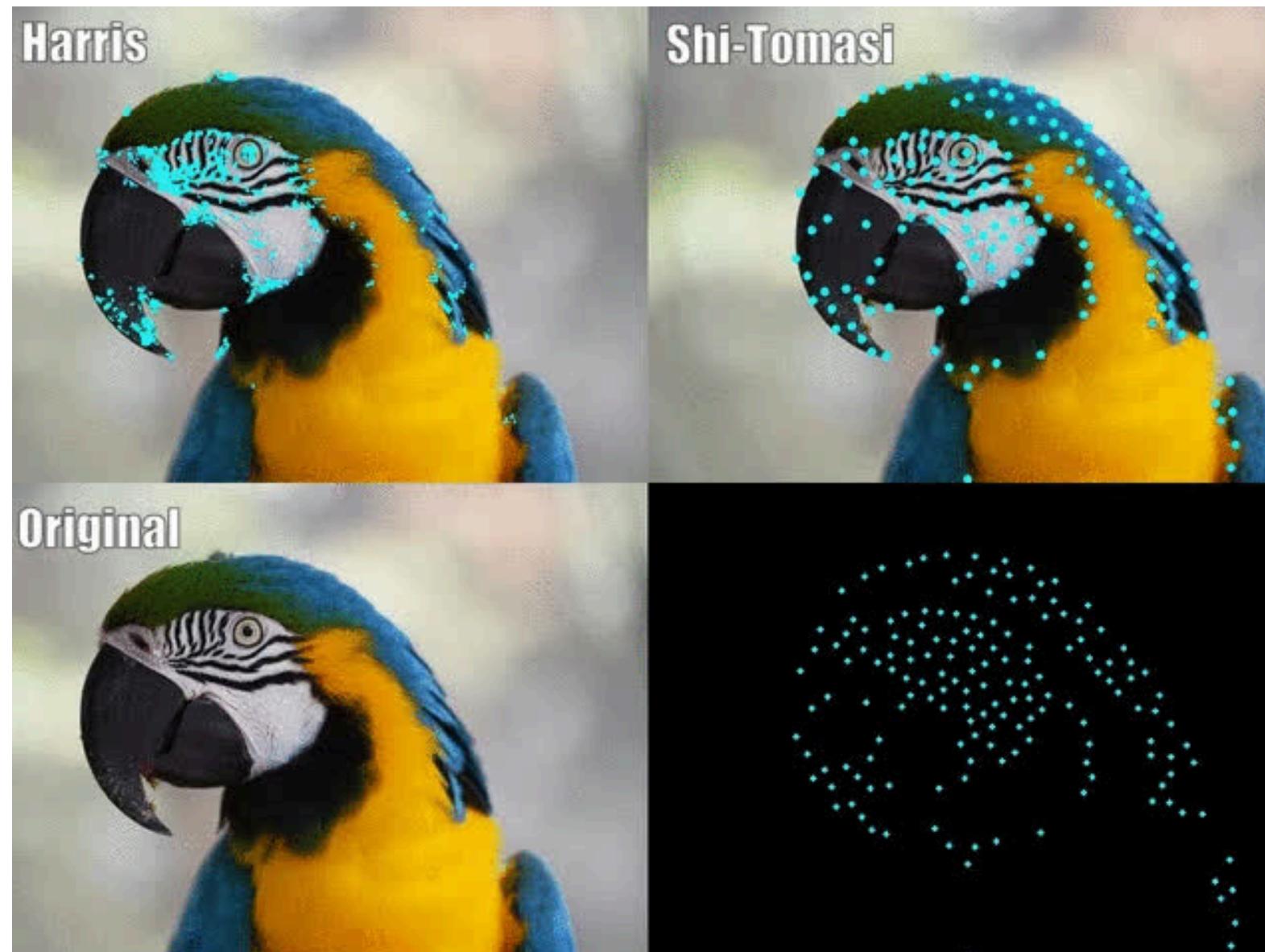
Limitações:

- **Não Invariante a Escala:** O algoritmo de Harris não é robusto a mudanças de escala, o que pode ser problemático ao lidar com objetos em diferentes tamanhos.
- **Sensibilidade a Ruído:** O método pode detectar cantos falsos em regiões ruidosas, especialmente sem o uso de uma função de suavização (filtro Gaussiano).

Corner Detection - Shi-Tomasi

Introduzido como uma **melhoria** ao **método de Harris**. Ele visa detectar pontos de interesse, principalmente para rastreamento, em aplicações como visão computacional e reconhecimento de padrões.

A **diferença** fundamental entre o método de Harris e o de Shi-Tomasi está no critério para determinar o que constitui um "canto". **Enquanto Harris utiliza uma função baseada no determinante e no traço da matriz de autocorrelação**, Shi-Tomasi propõe um critério mais intuitivo e robusto, focando diretamente nos **autovalores da matriz de autocorrelação**.



Corner Detection - Shi-Tomasi

Etapas:

1. Assim como no método de Harris, o método de Shi-Tomasi analisa a variação de intensidade em uma região ao redor de cada pixel, representada pela **matriz de autocorrelação M**



2. Mas ao contrário do método de Harris, que usa uma função para avaliar a resposta da matriz, o método de Shi-Tomasi considera diretamente os **autovalores de M**, denotados como λ_1 e λ_2

3. Critério de Detecção de Cantos em Shi-Tomasi:

Para um pixel ser considerado um "canto" no método de Shi-Tomasi, **o menor dos autovalores da matriz M deve ser maior que um limiar predefinido:**

$$\text{score} = \min(\lambda_1, \lambda_2)$$

Esse critério faz sentido intuitivo: um ponto com alta curvatura em duas direções terá autovalores grandes, indicando que a intensidade muda significativamente em ambas as direções.

No algoritmo:

- Se $\min(\lambda_1, \lambda_2) > \text{limiar}$, o ponto é classificado como um canto.
- Se $\min(\lambda_1, \lambda_2)$ é baixo ou próximo de zero, significa que a intensidade não varia significativamente em ambas as direções, então o ponto não é um canto.

Corner Detection - Shi-Tomasi

Vantagens e Limitações do Detector Shi-Tomasi

Vantagens

- **Maior Precisão:** Shi-Tomasi é mais estável do que o método de Harris, pois utiliza diretamente os autovalores, evitando erros de detecção em pontos de interesse sensíveis.
- **Robusto para Rastreamento:** Por ser mais confiável, o método é amplamente usado em aplicações que exigem rastreamento de pontos.

Limitações

- **Não Invariante a Escala:** Assim como Harris, o método de Shi-Tomasi não é robusto a mudanças de escala, limitando sua aplicação em imagens onde objetos variam de tamanho significativamente.
- **Sensível a Ruído:** Embora o filtro Gaussiano ajude a suavizar a imagem, o método ainda pode detectar cantos falsos em regiões ruidosas.

Feature Descriptors

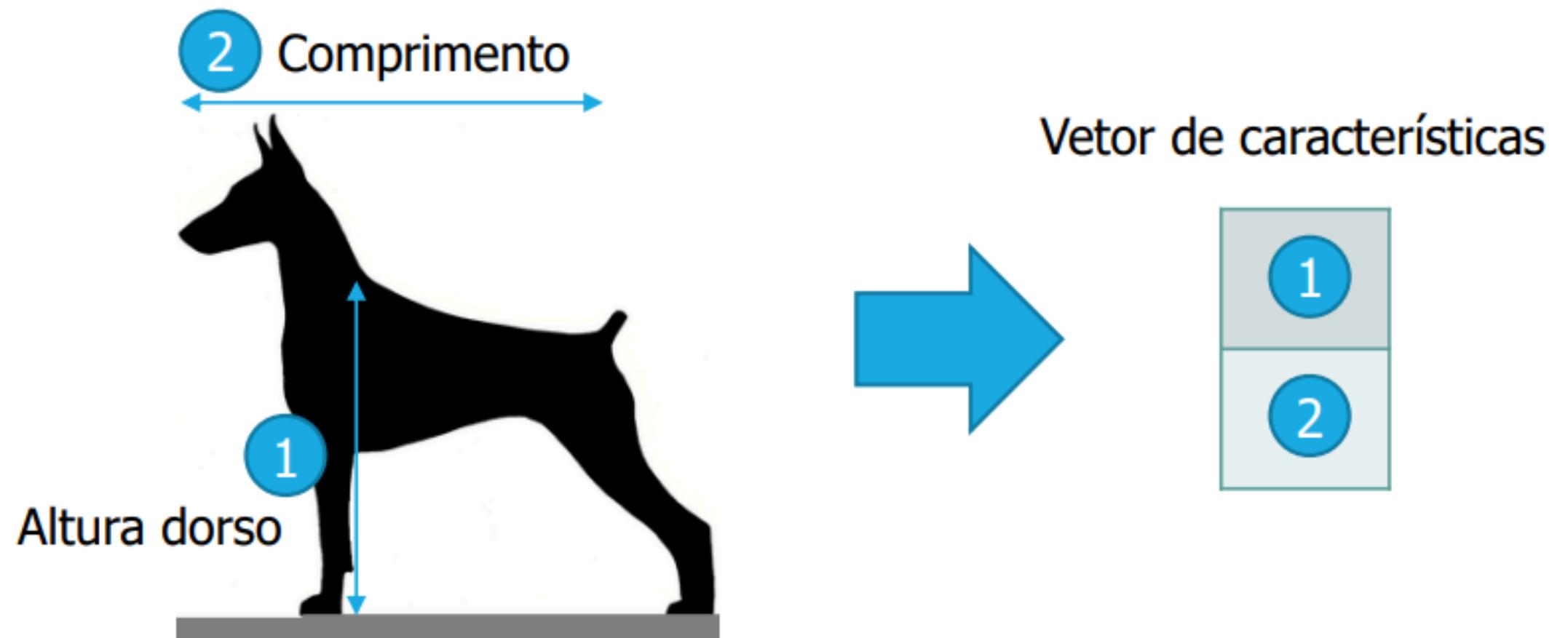
Como podemos descrever estas imagens, com cachorros, de modo a classifica-las quanto a raça?



Um descritor de imagem é justamente essa representação numérica que captura as características mais importantes de uma imagem, como bordas, texturas, cores e formas.

Feature Descriptors

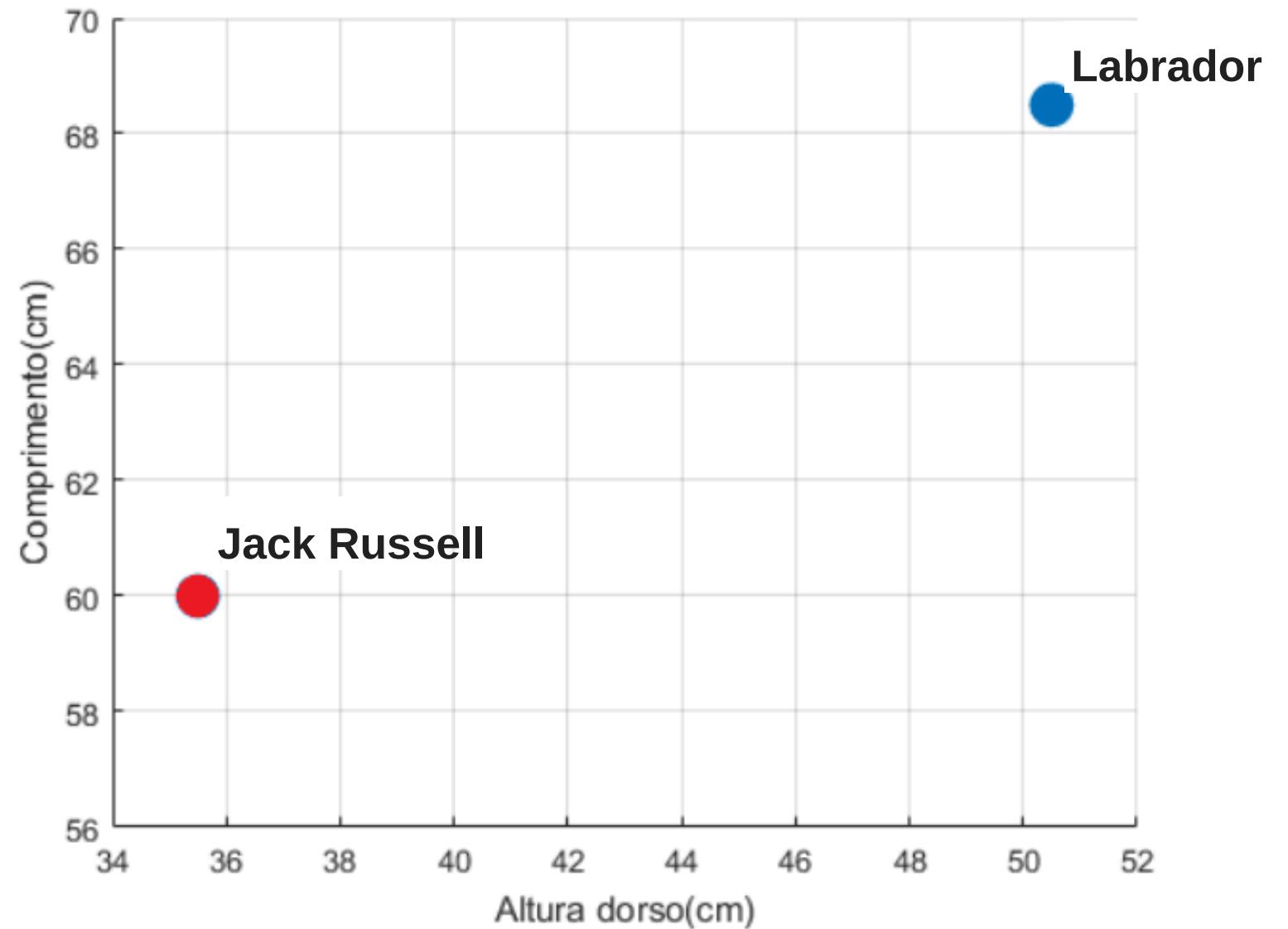
Exemplo:



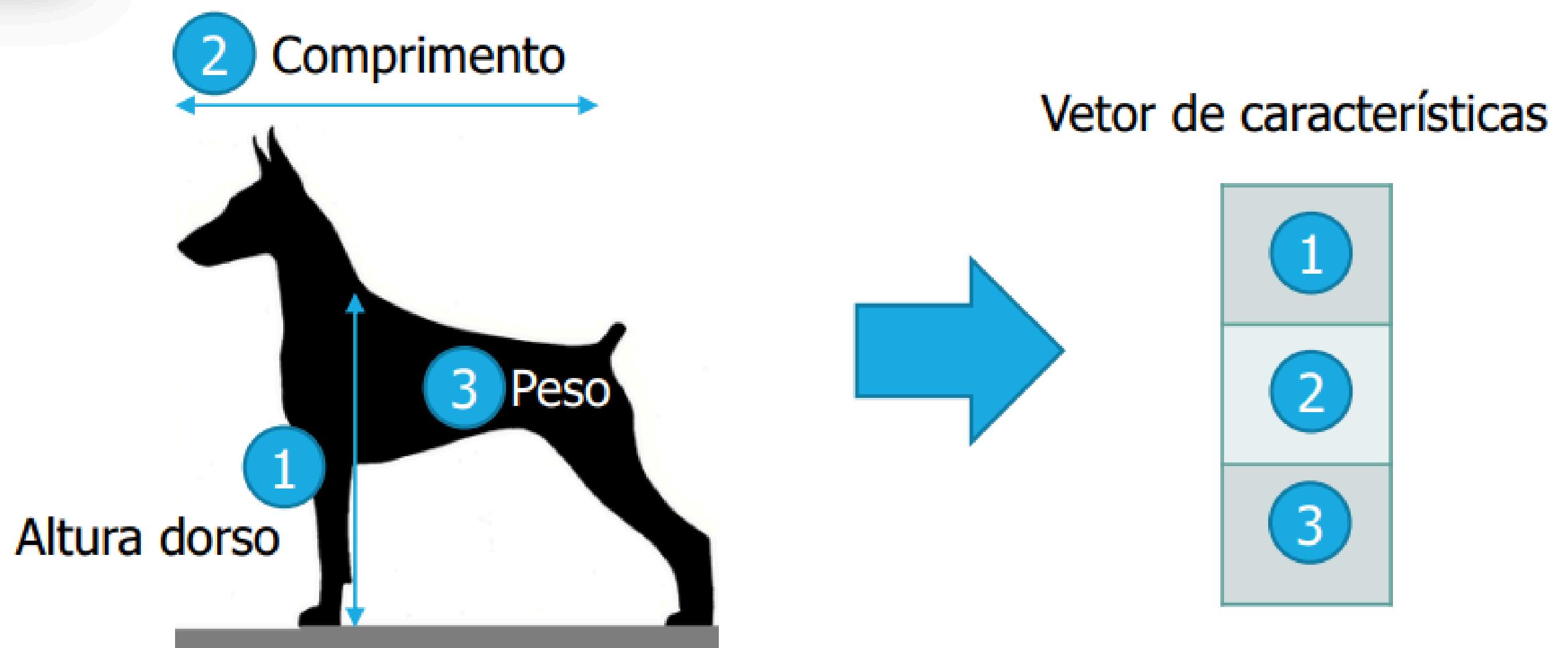
Feature Descriptors

Vetor com descritores de característica:

No espaço de atributos do vetor de característica:



Feature Descriptors



Feature Descriptors

Classificação dos Descritores:

Os descritores de imagem podem ser classificados de diversas formas, mas duas das classificações mais comuns são:

Globais: Consideram a imagem como um todo, capturando características como:

Histograma de cores;

Textura;

Forma.

Locais: Focam em regiões específicas da imagem, como pontos de interesse, capturando detalhes mais finos. Conseguem capturar características locais invariantes a rotação e escala.

SIFT, SURF, ORB, LBP,

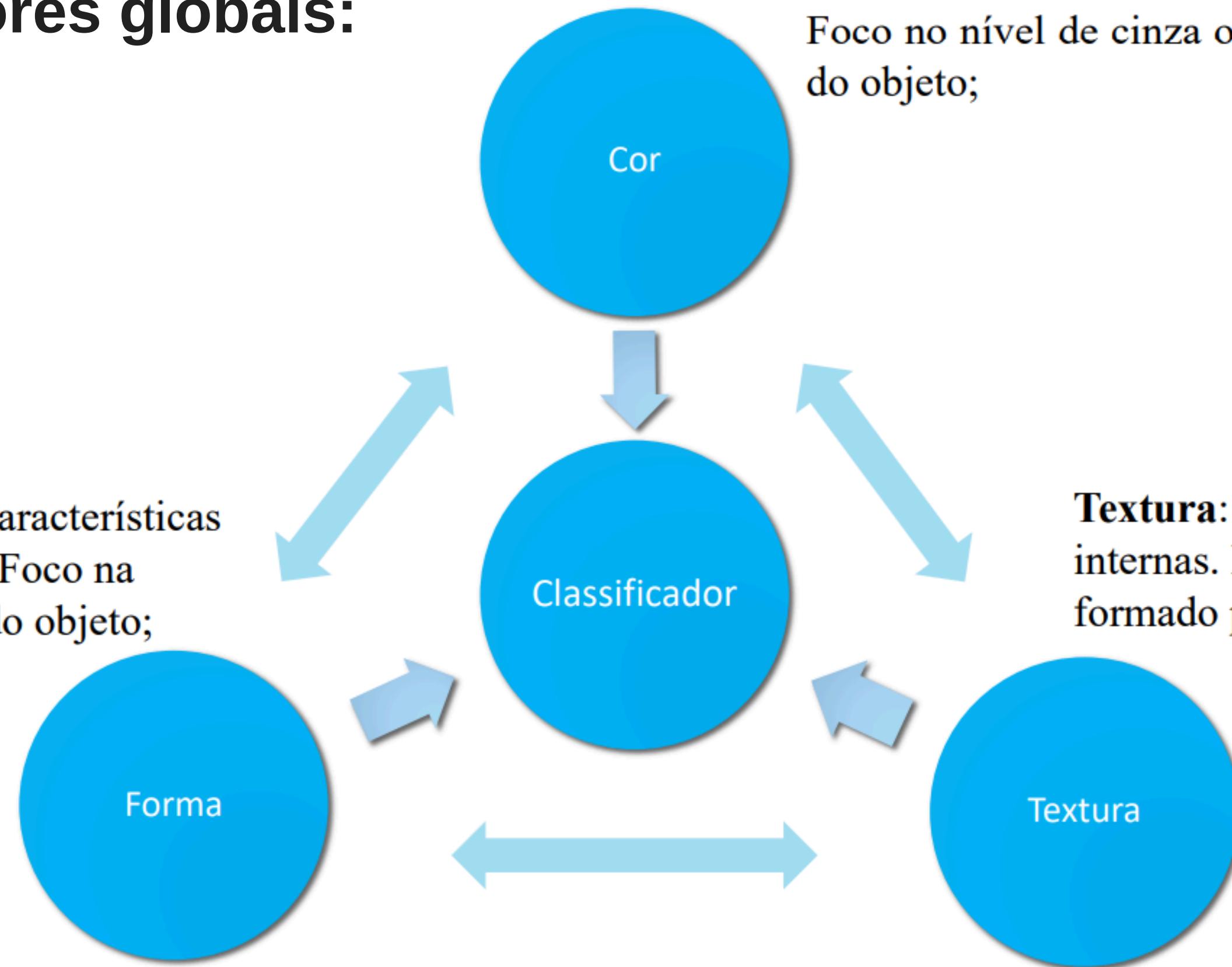
Feature Descriptors

Descritores globais:

Forma: Características externas. Foco na fronteira do objeto;

Cor: Características internas. Foco no nível de cinza ou cor do objeto;

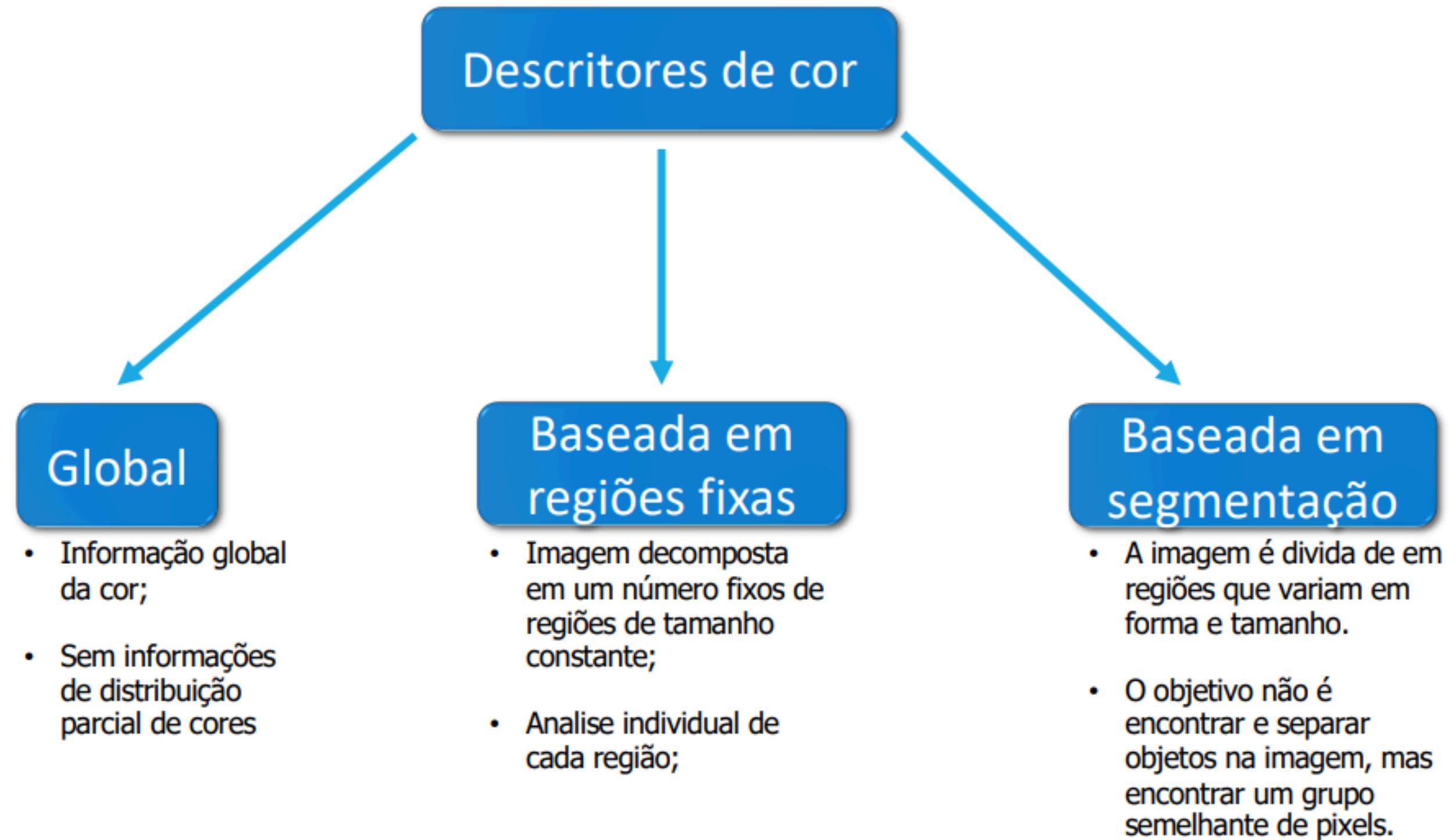
Textura: Características internas. Foco no padrão formado pela região.



Feature Descriptors

Descritores globais:

Descritores de cor são representações numéricas que capturam a distribuição das cores em uma imagem.



Feature Descriptors

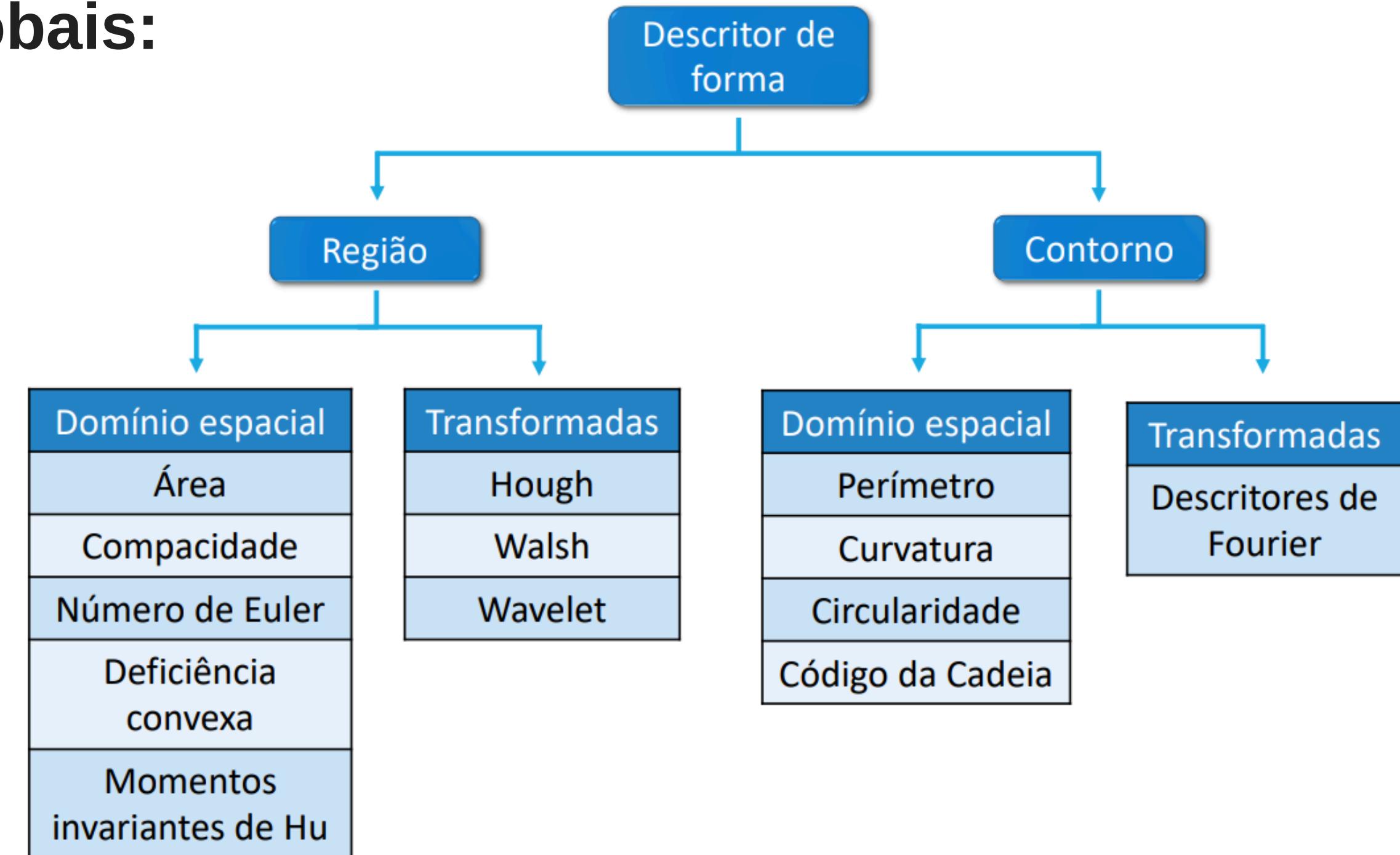
Descritores globais:

Identificação de pontos chaves

Descrição da forma usando varias técnicas

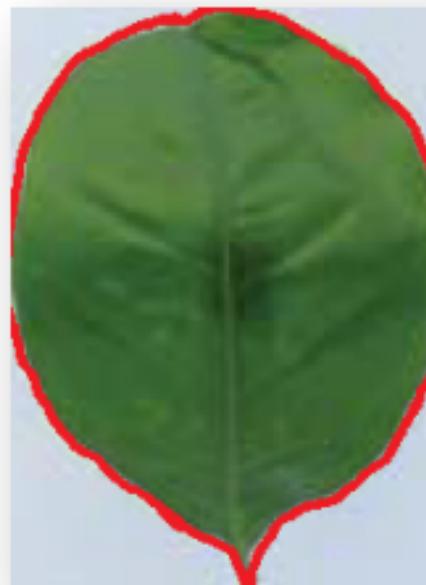
Robustez a Transformações:

Comparação de Descritores:



Feature Descriptors

Folha de café com
Deficiência de Boro



Folha de café com
deficiência de Cálcio



| Deficiência | Boro | Cálcio |
|---------------|--------|--------|
| Perímetro | 469,75 | 611,2 |
| Área | 15516 | 18412 |
| Compacidade | 14,22 | 20,29 |
| Circularidade | 83,15 | 58,84 |

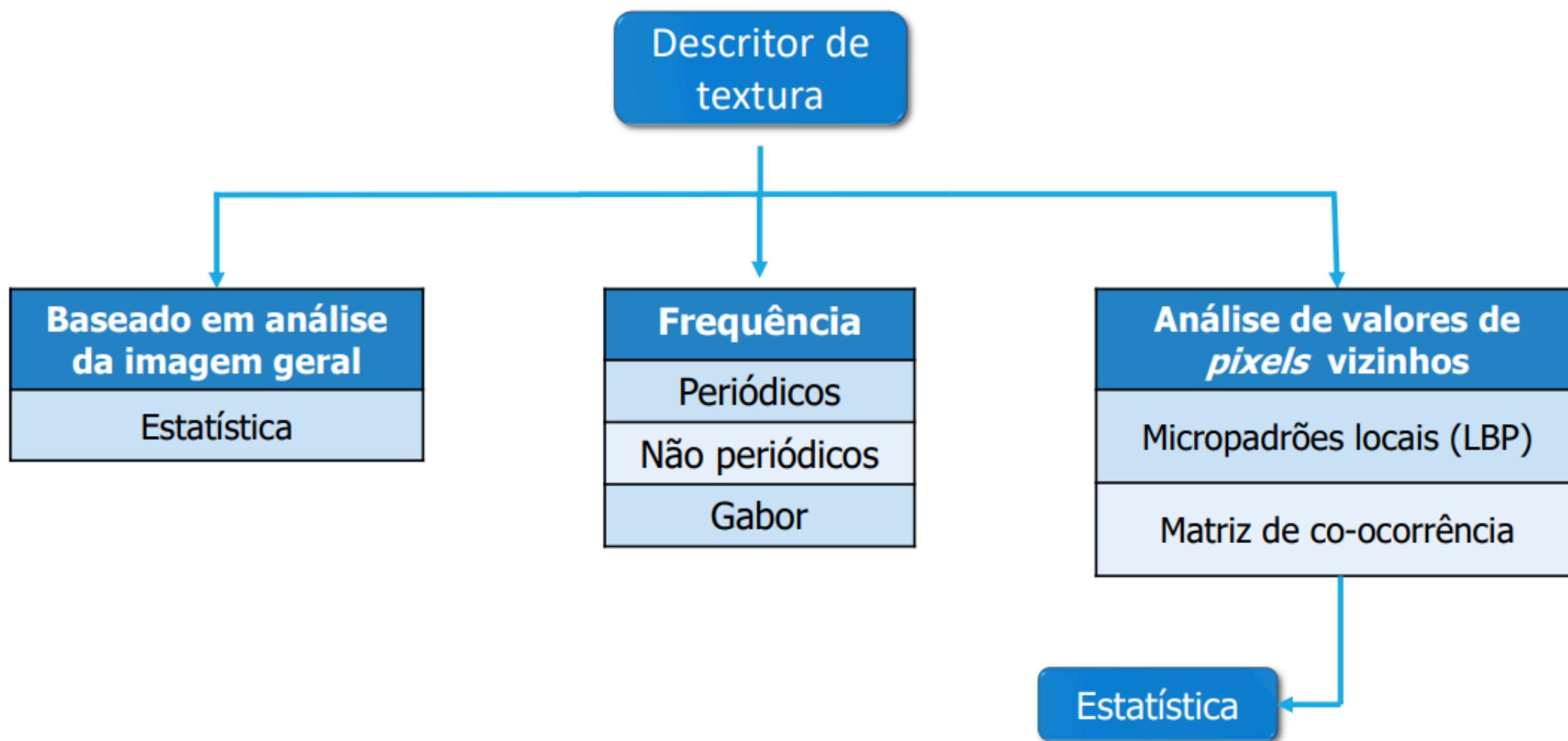
Feature Descriptors

Descritores globais:

A ideia básica é analisar a distribuição espacial dos níveis de cinza em uma região da imagem. Essa distribuição fornece informações sobre a regularidade, orientação e frequência dos padrões na textura.

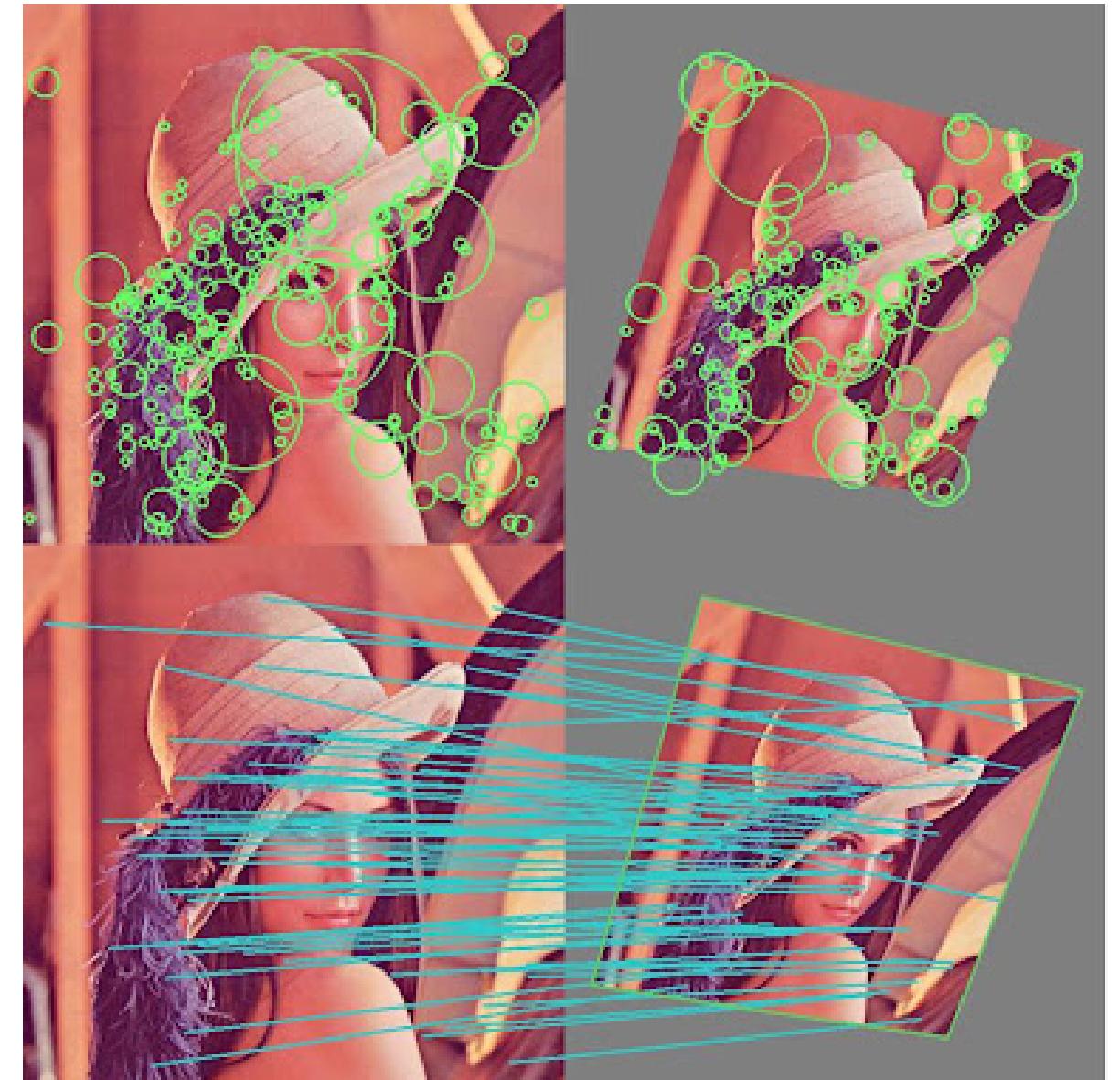


Padrão de textura de um tronco



SIFT (Scale Invariant Feature Transform)

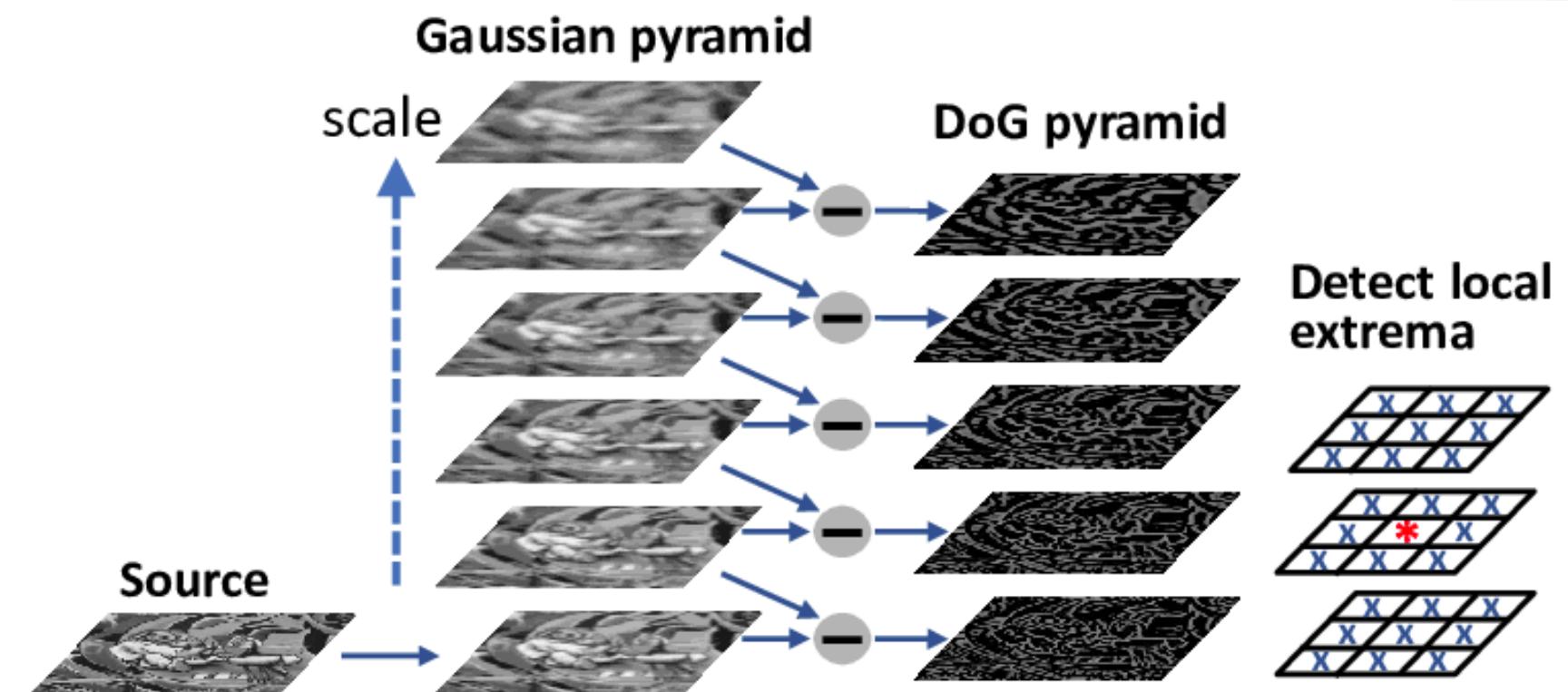
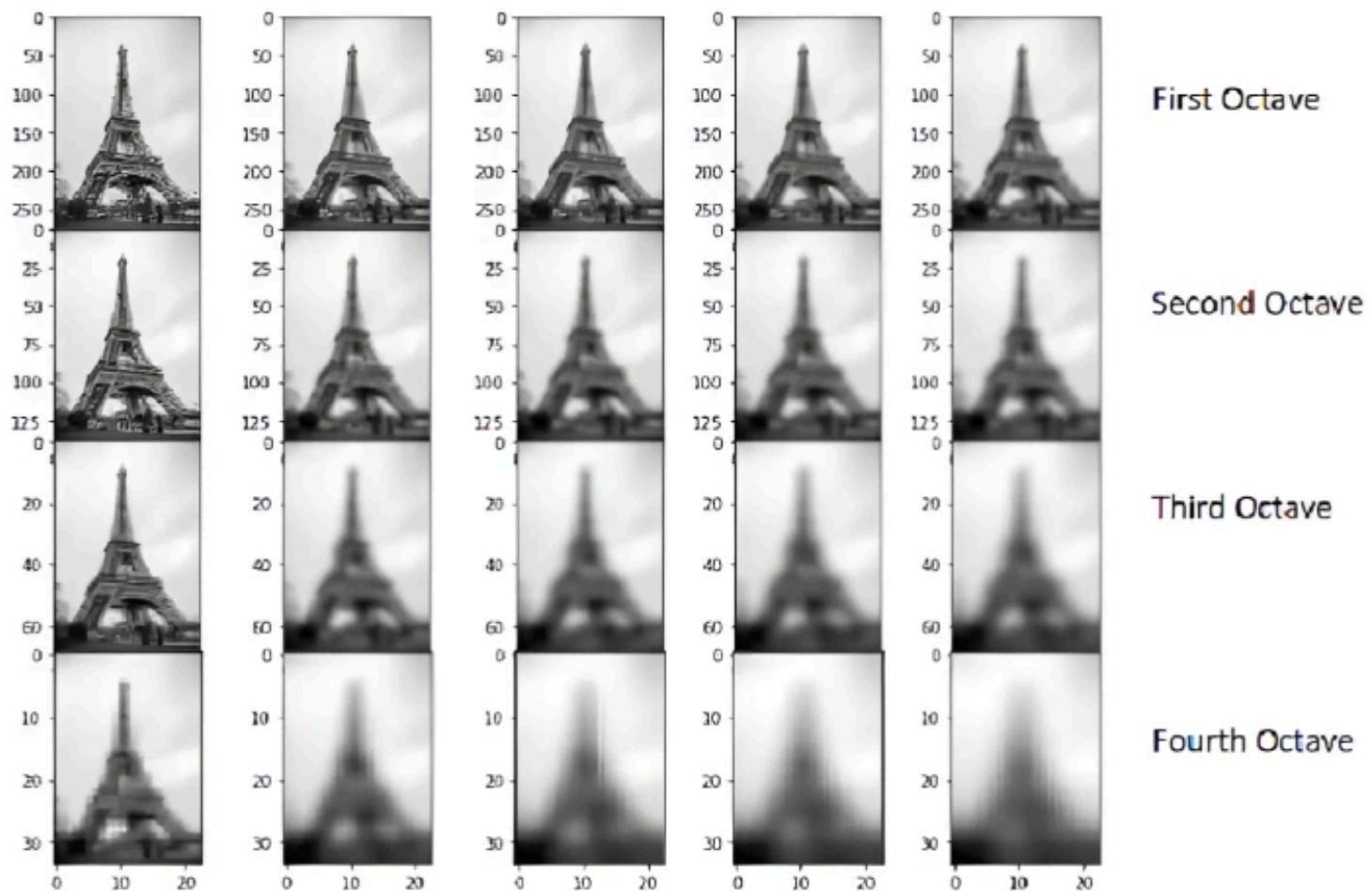
- Robusto à mudanças de escala e rotação
- Parcialmente robusto à mudanças de iluminação e ponto de vista
- Certa resistência à ruídos
- Usado em:
 - correspondência entre imagens
 - reconhecimento e rastreamento de objetos
 - criação de mosaicos e panoramas
 - reconstrução 3D
 - visão robótica



SIFT (Scale Invariant Feature Transform)

Como funciona o SIFT?

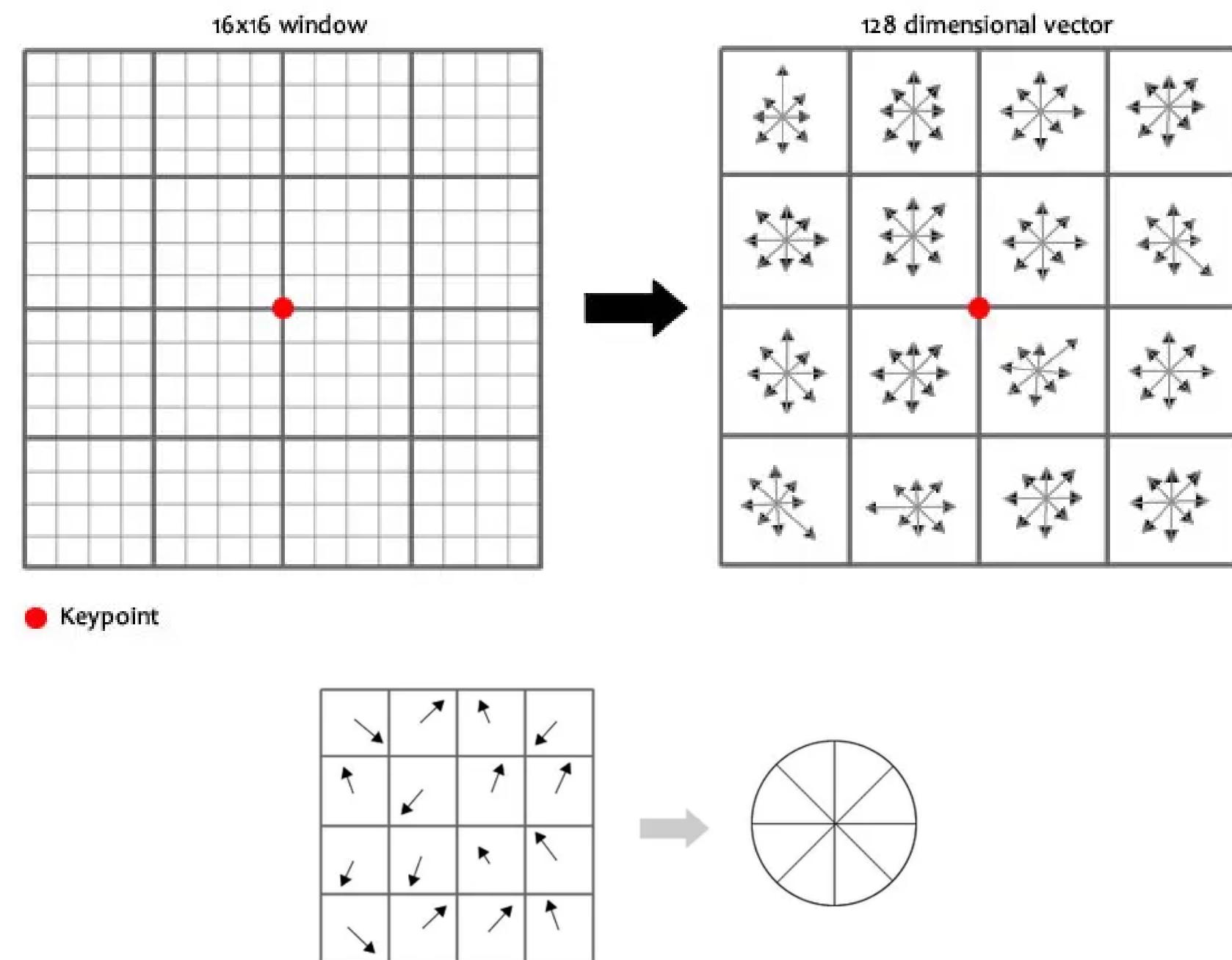
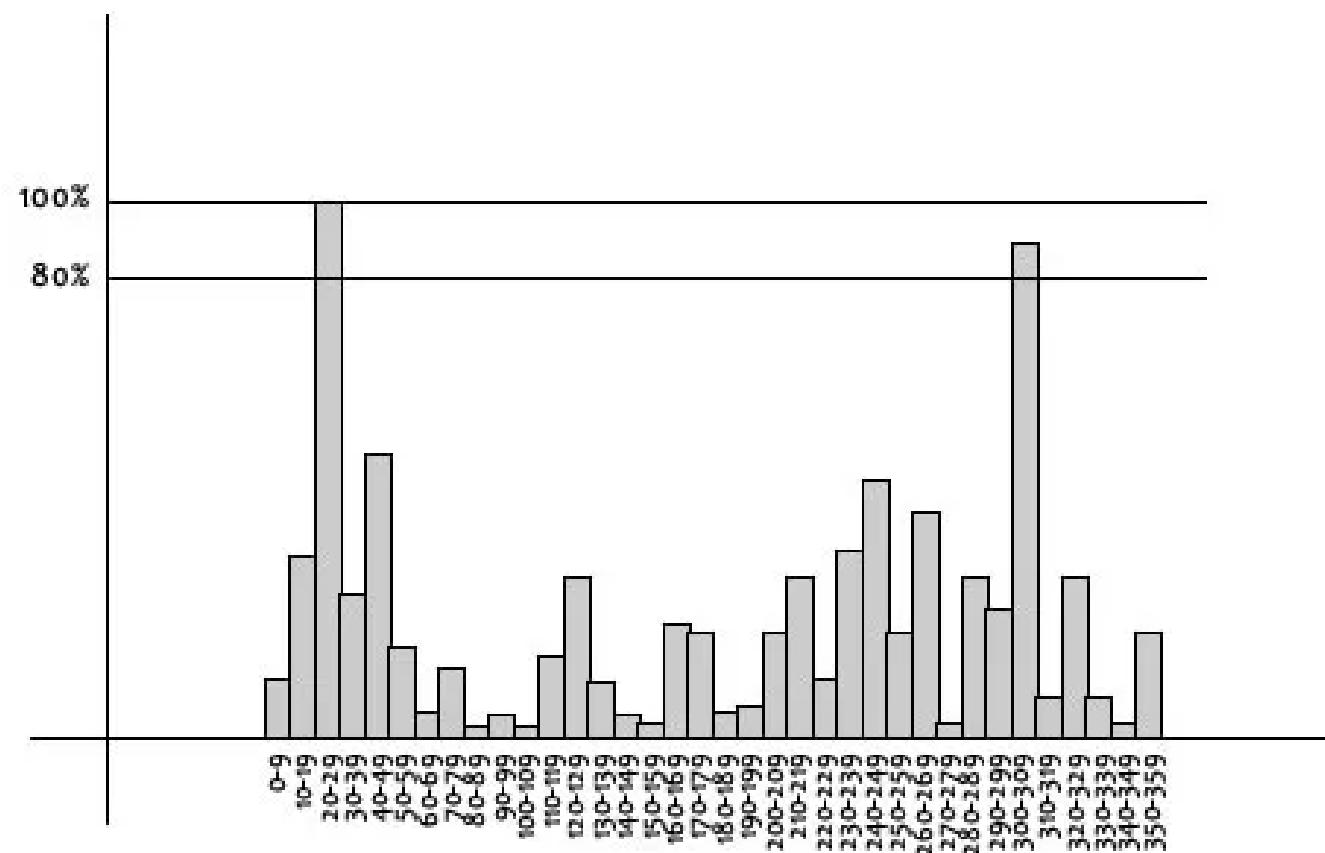
- Detecção de Extremos no Espaço de Escalas
 - Pirâmide Gaussiana
 - DoG - *Difference of Gaussian*



SIFT (Scale Invariant Feature Transform)

Como funciona o SIFT?

- Refinamento e Filtragem de Pontos de Interesse
- Atribuição de orientação
- Geração do Descritor



Obrigado!

Dúvidas ou sugestões:

davijesus@discente.ufg.br

rian.souza@discente.ufg.br

caiolucca@discente.ufg.br

INF
INSTITUTO DE
INFORMÁTICA

