IF69D-S11 Proc. Digital de Imagens
APNP

Aula 07

Universidade Tecnológica Federal do Paraná - UTFPR - Curitiba
Departamento Acadêmico de Eletrônica - DAELN
Curso de Engenharia Eletrônica
Prof. Gustavo B. Borba

06.abr.2021

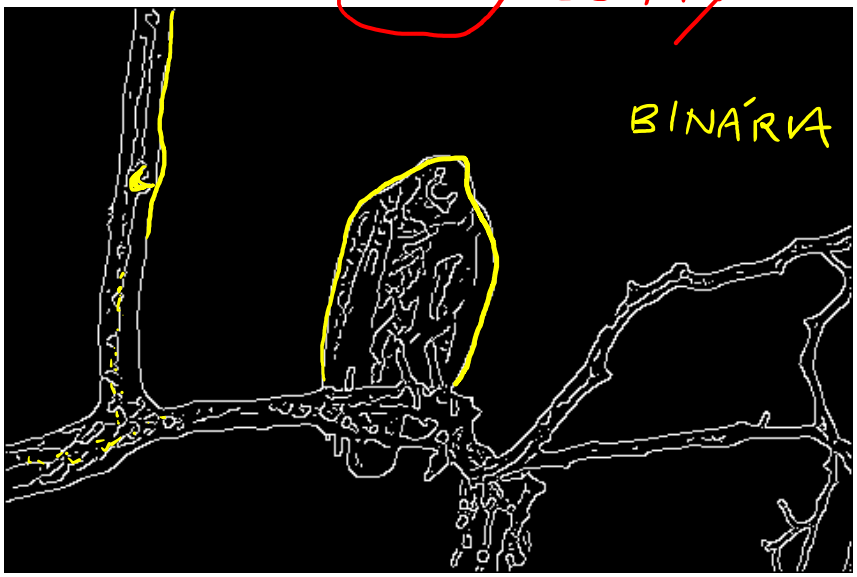
Detecção de bordas

Edge detection

- Bordas são descontinuidades na intensidade dos níveis de cinza.
- Deseja-se que um processo de detecção de bordas apresente na saída uma imagem binária, que indica as bordas da imagem de entrada.

IN

OUT Canny



BW ← White (branco)
black
correspondem às BORDAS da imagem de entrada

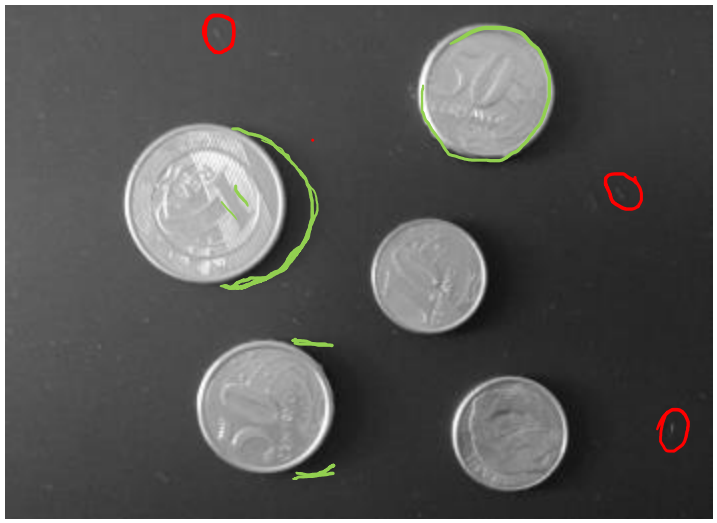
Octave:
FUNÇÃO edge

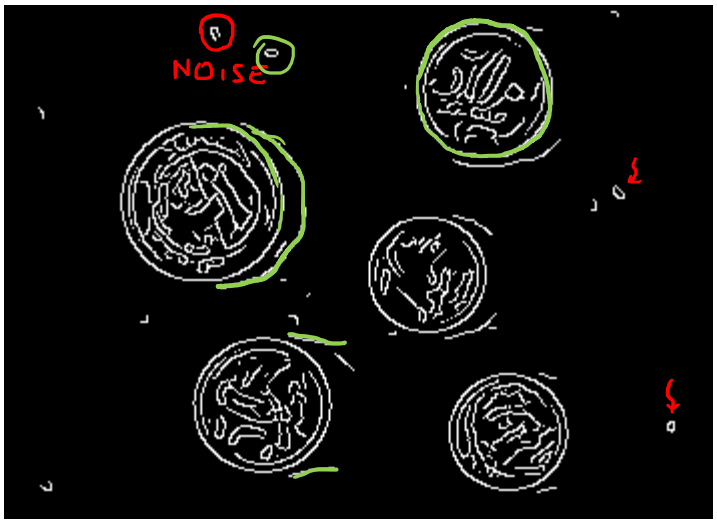
↓ Berkley

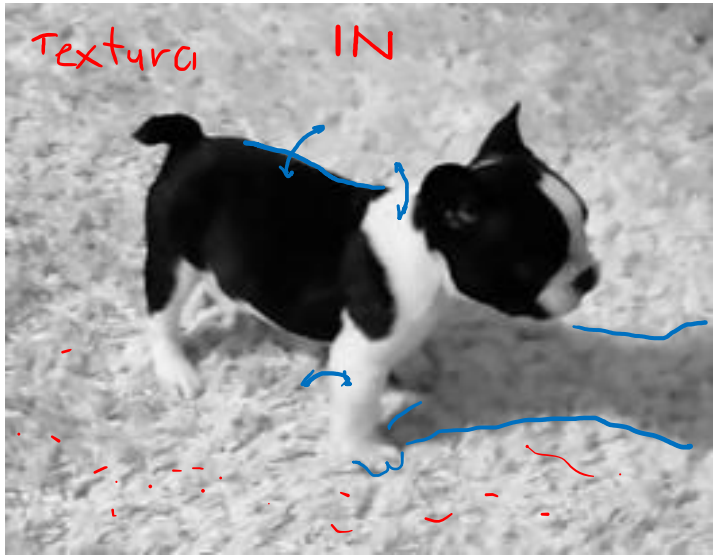
Imagem: https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/BSDS300/html/dataset/images/gray/42049.html

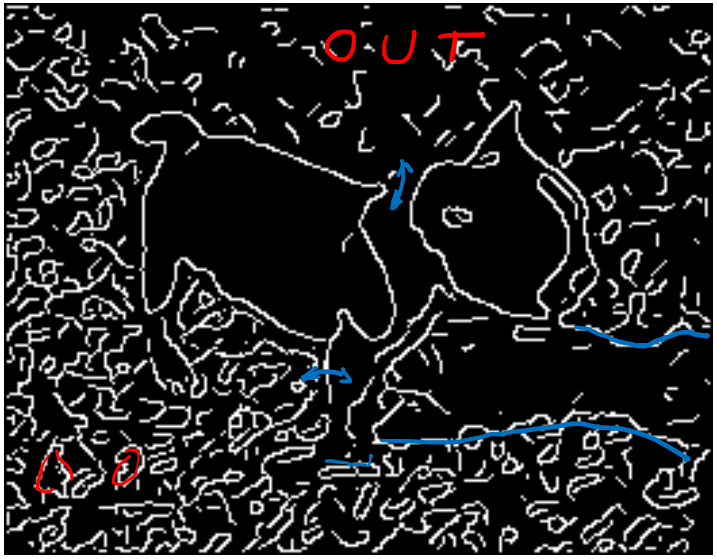
Detecção de bordas

Canny





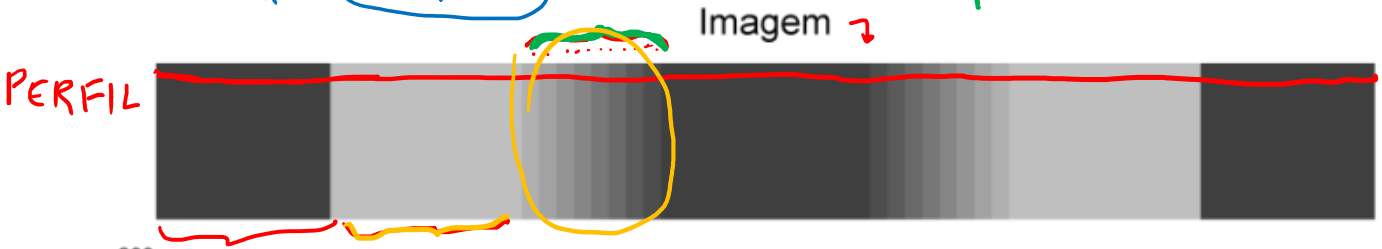


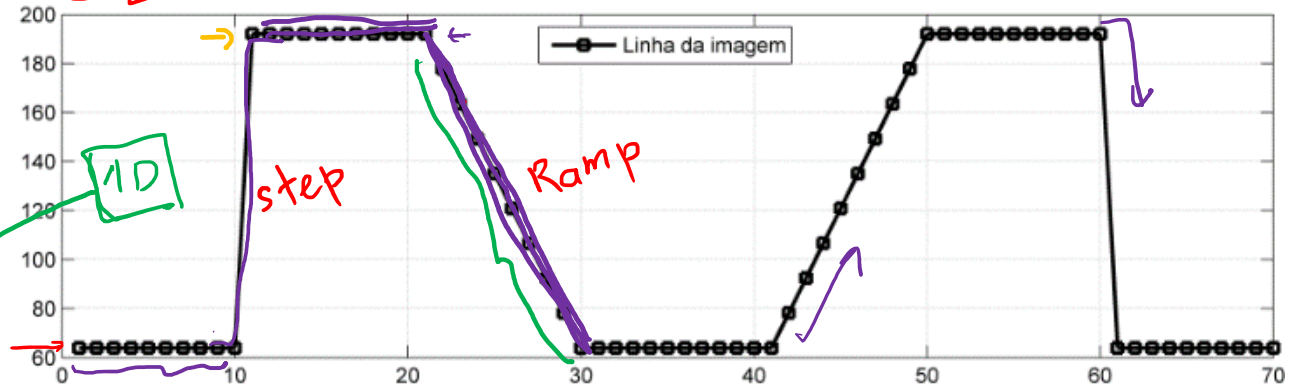


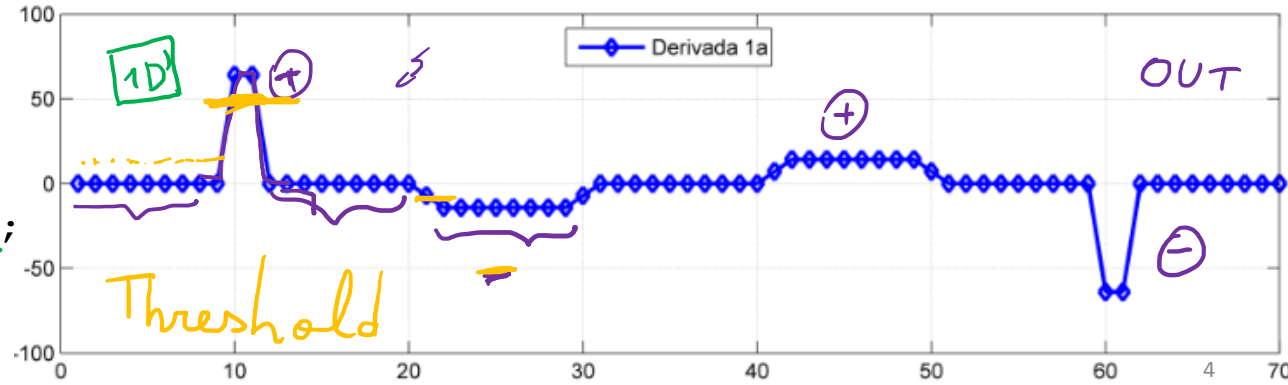
Detecção de bordas por gradiente (primeira derivada)

$$\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x-h)}{2h} = \frac{f(x+1) - f(x-1)}{2}$$

- A derivada de uma constante (não tem variação) é zero.
- A derivada de um degrau é um impulso.
- A derivada de uma reta (inclinação constante) é uma constante.

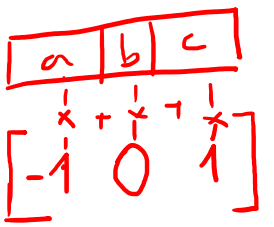




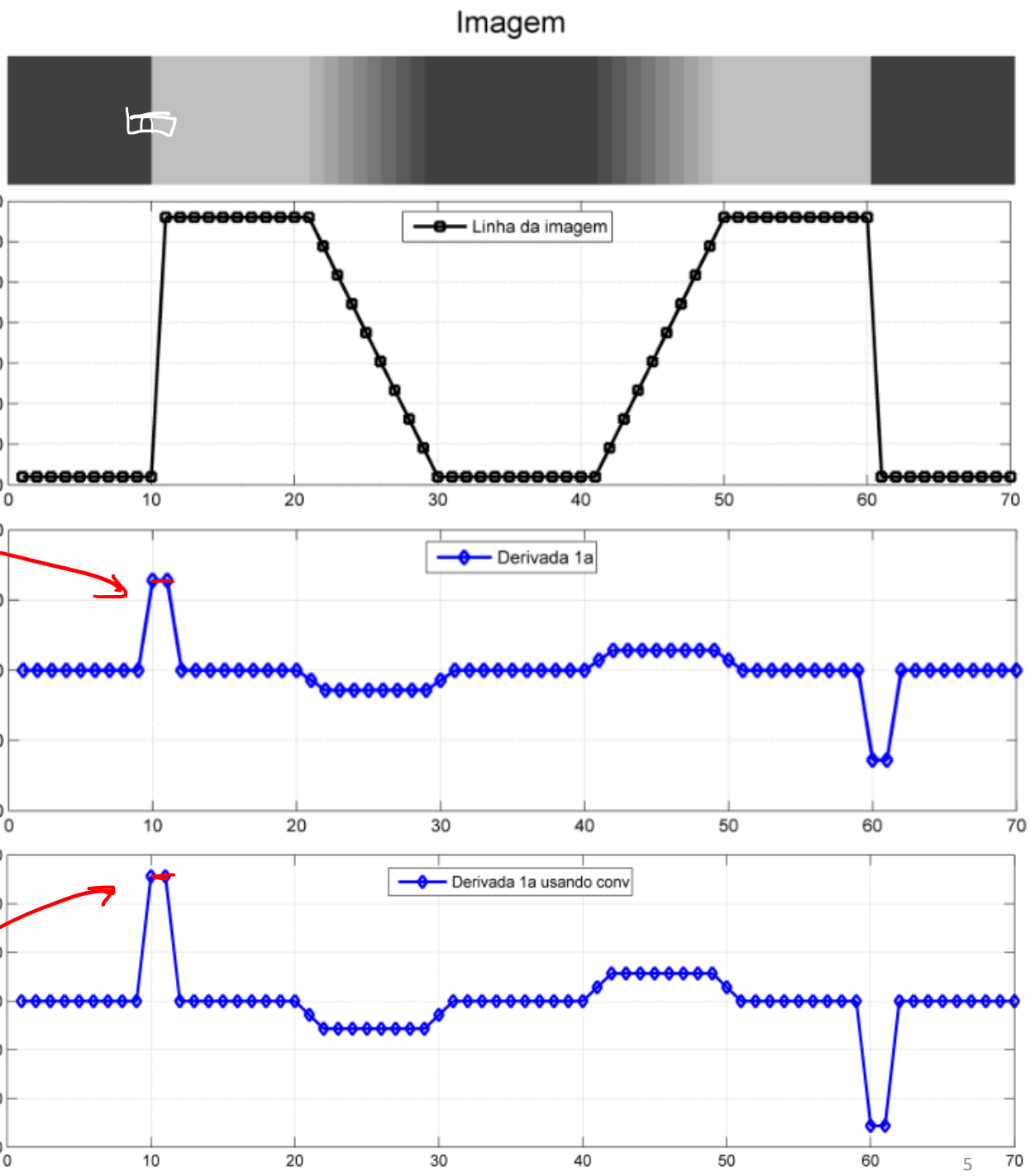



```
row = img(1,1:ncol);
% Derivada de
% primeira ordem
% segundo definição
for k=2:ncol-1
 d(k)=row(k+1)-row(k-1);
end
d=d/2; % Derivada 1a
```

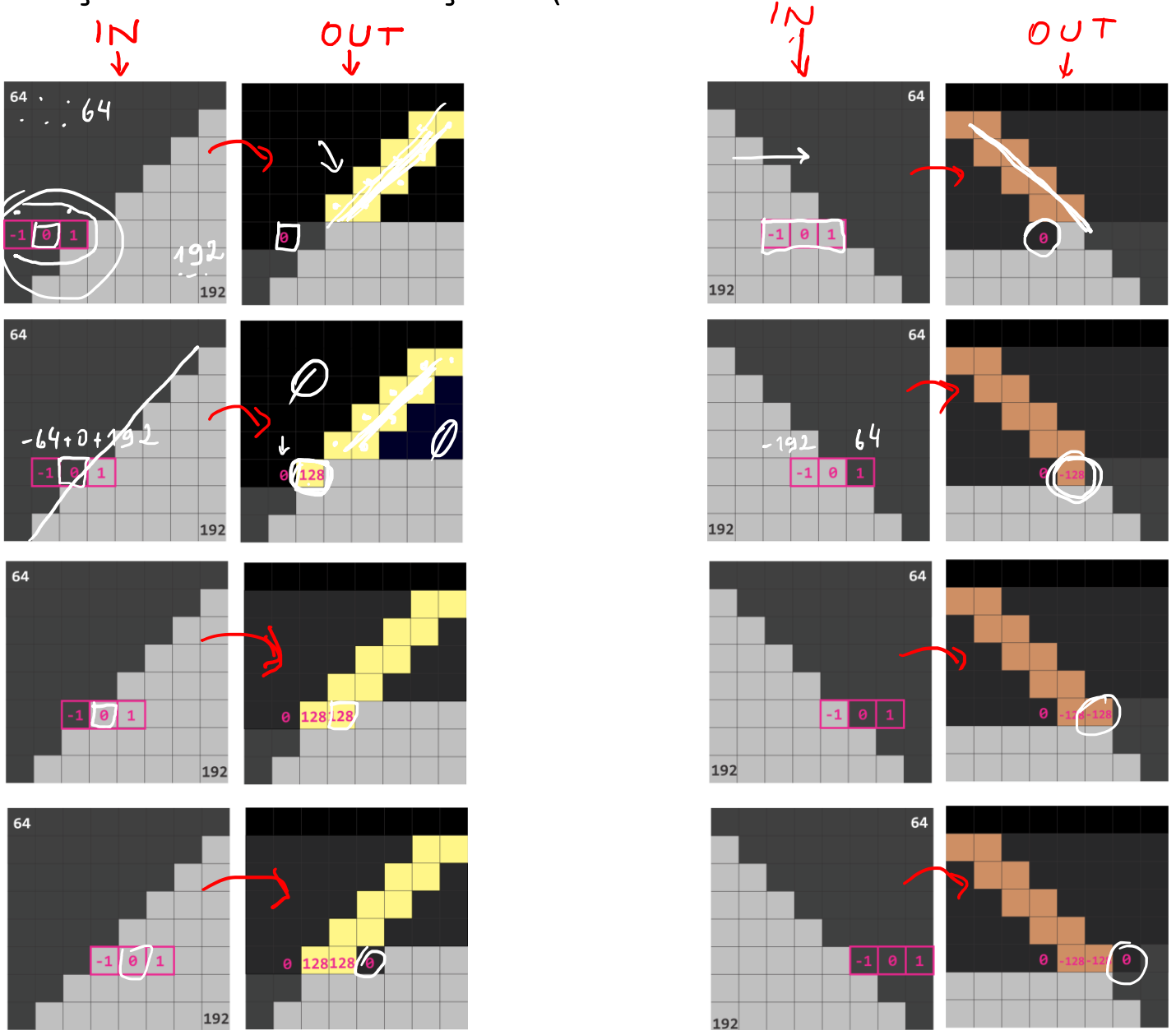
Detecção de bordas por gradiente (primeira derivada) usando a convolução



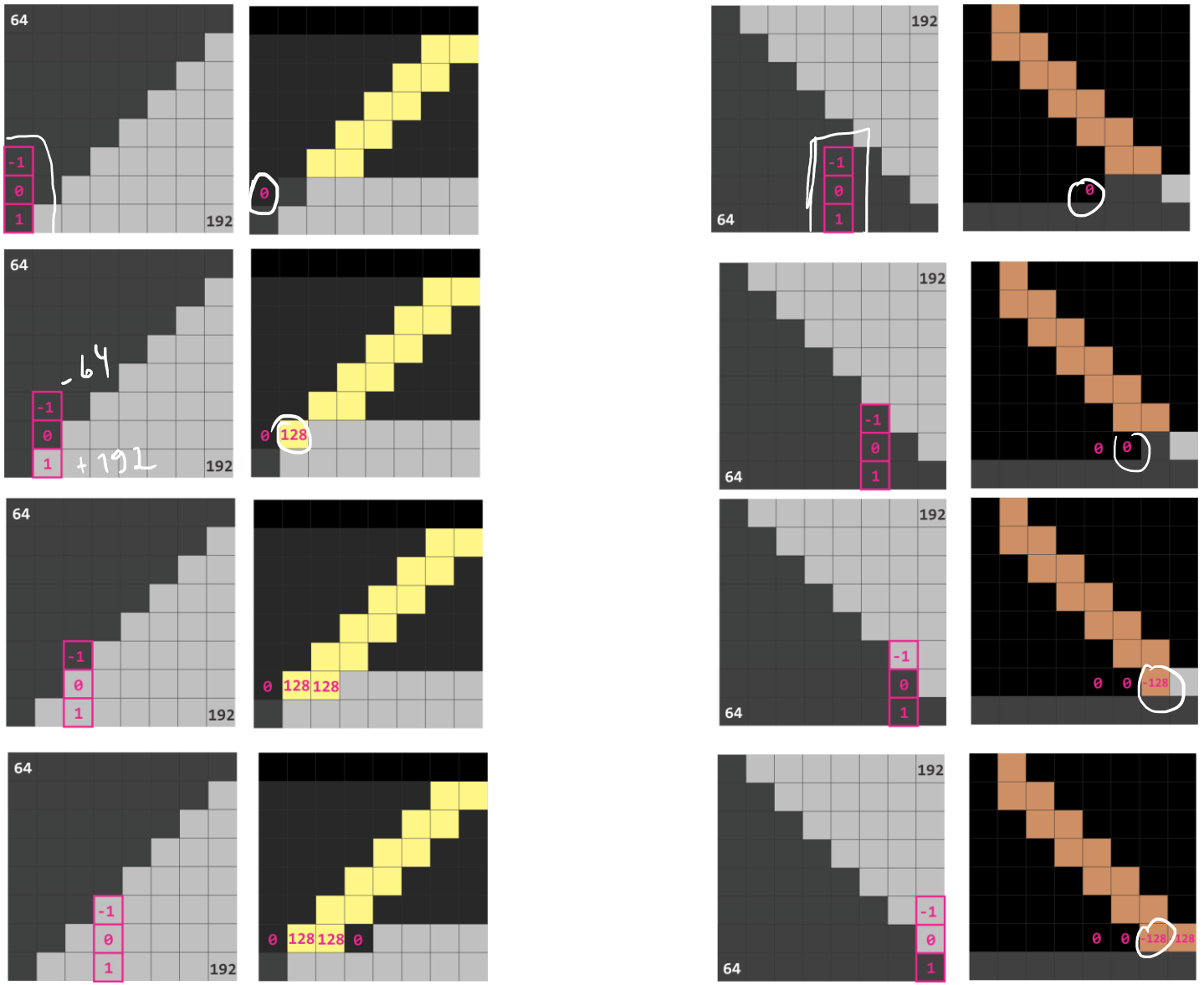

```
row = img(1,1:ncol);
% Derivada de
% primeira ordem
% segundo definição
for k=2:ncol-1
 d(k)=row(k+1)-row(k-1);
end
d=d/2; % Derivada 1a

% Máscara para derivada 1a
% por convolução para
% reproduzir f(x+1)-f(x-1).
% Fazer conv com [1 0 -1] e
% não com [-1 0 1] porque a
% convolução espelha a
% máscara em x e y.
h = [1 0 -1];
d = conv(row,h,'valid');
```



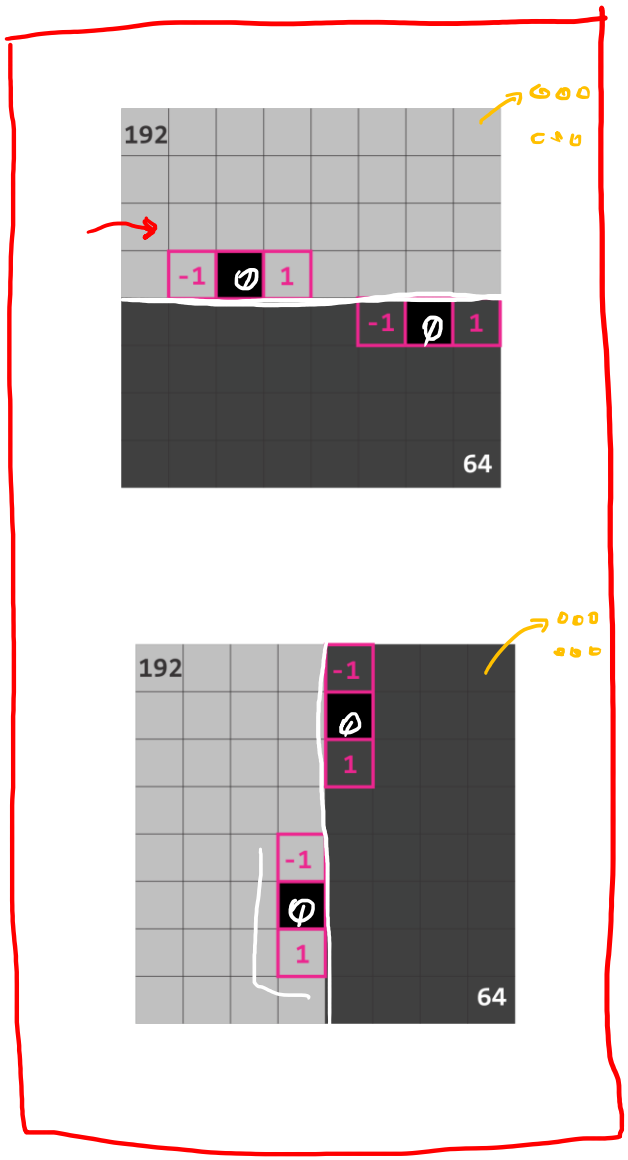
Detecção de bordas por gradiente (primeira derivada) com a convolução nas duas direções (máscaras vertical e horizontal)



Detecção de bordas por gradiente (primeira derivada) com a convolução nas duas direções (máscaras vertical e horizontal)



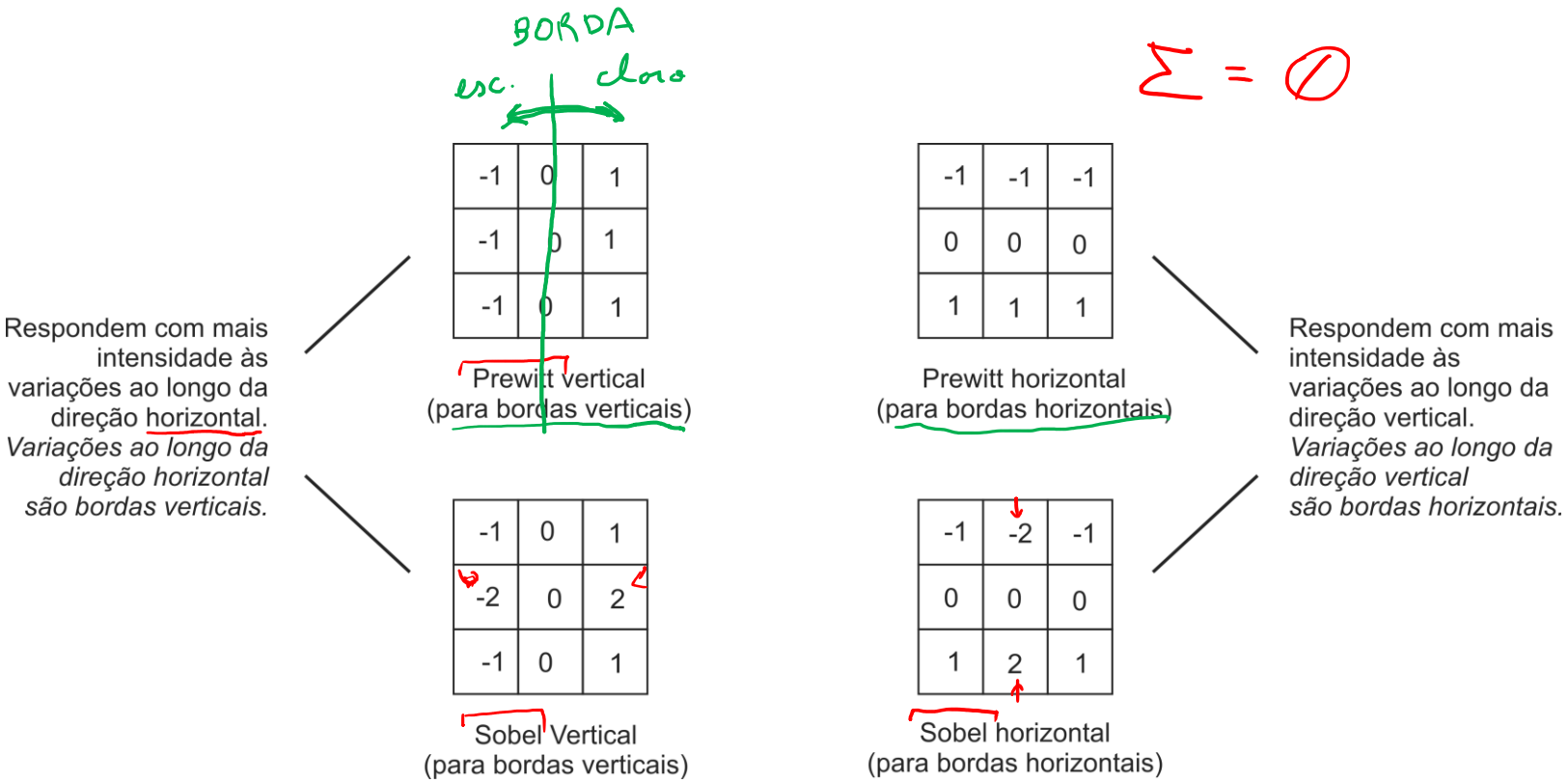
Detecção de bordas por gradiente (primeira derivada) com a convolução nas duas direções (máscaras vertical e horizontal)



Detecção de bordas por gradiente (primeira derivada)

Máscaras mas utilizadas:

de CONVOLUÇÃO

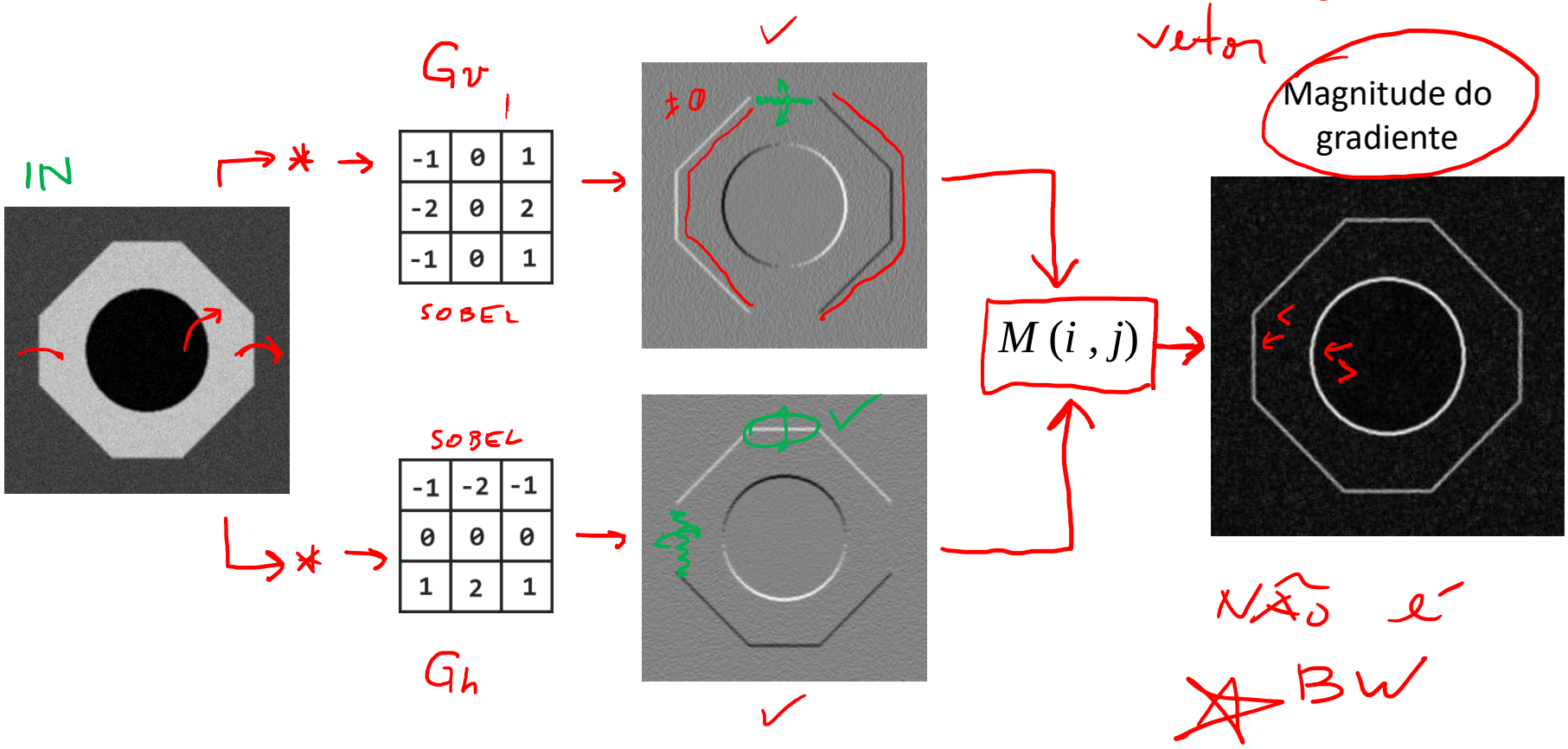


SOBEL

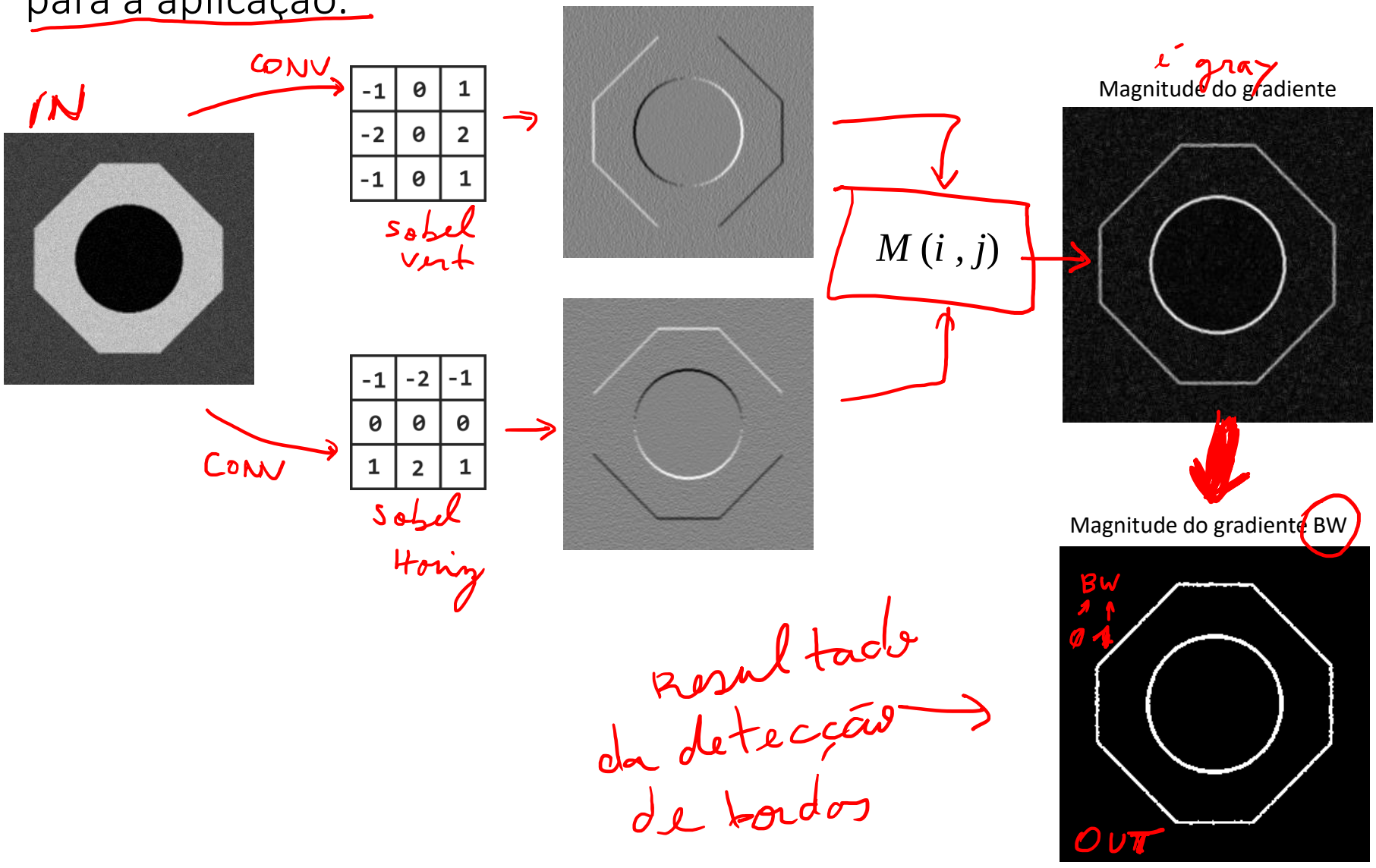
Detecção de bordas por gradiente (primeira derivada)
Combinando as saídas das máscaras nas duas direções:
Magnitude do gradiente:

$$M(i,j) = \sqrt{G_v(i,j)^2 + G_h(i,j)^2}$$

módulo de um vetor

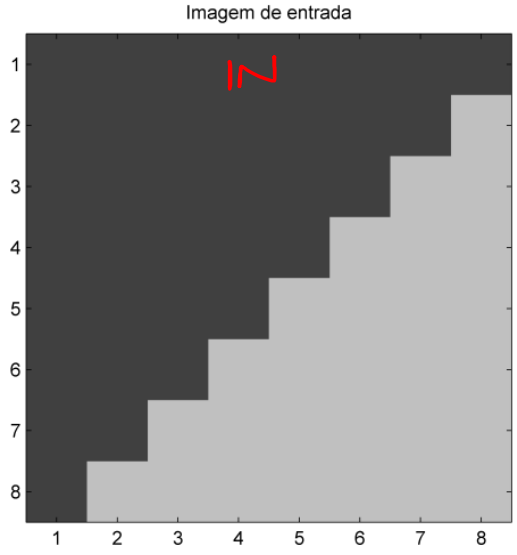


A saída esperada de um detector de bordas é uma imagem binária. Para binarização automática utiliza-se alguma técnica adequada para a aplicação.



Detecção de bordas por gradiente (primeira derivada)

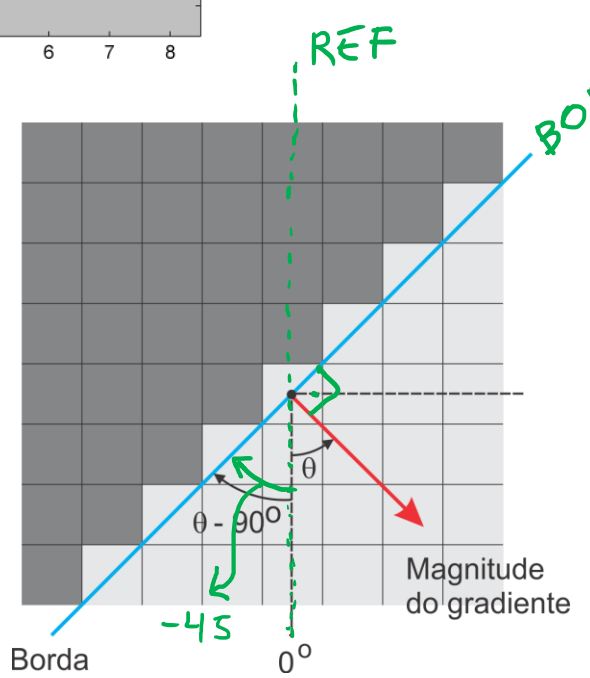
Orientação (ângulo) do vetor magnitude do gradiente e da borda:



$$\theta(i,j) = tan^{-1}\left(\frac{G_h(i,j)}{G_v(i,j)}\right)$$

arctg

NaN	NaN	NaN	NaN	NaN	NaN	45.00	71.57
NaN	NaN	NaN	NaN	NaN	45.00	45.00	63.43
NaN	NaN	NaN	NaN	45.00	45.00	45.00	45.00
NaN	NaN	NaN	45.00	45.00	45.00	45.00	NaN
NaN	NaN	45.00	45.00	45.00	45.00	NaN	NaN
NaN	45.00	45.00	45.00	45.00	NaN	NaN	NaN
45.00	45.00	45.00	45.00	NaN	NaN	NaN	NaN
18.43	26.57	45.00	NaN	NaN	NaN	NaN	NaN



1. A referência é o eixo vertical (0°)
2. O ângulo obtido para a magnitude do gradiente é θ = 45°
3. A borda é ortogonal à magnitude do gradiente
4. O ângulo da borda é dado por θ - 90° = 45° - 90° = -45°

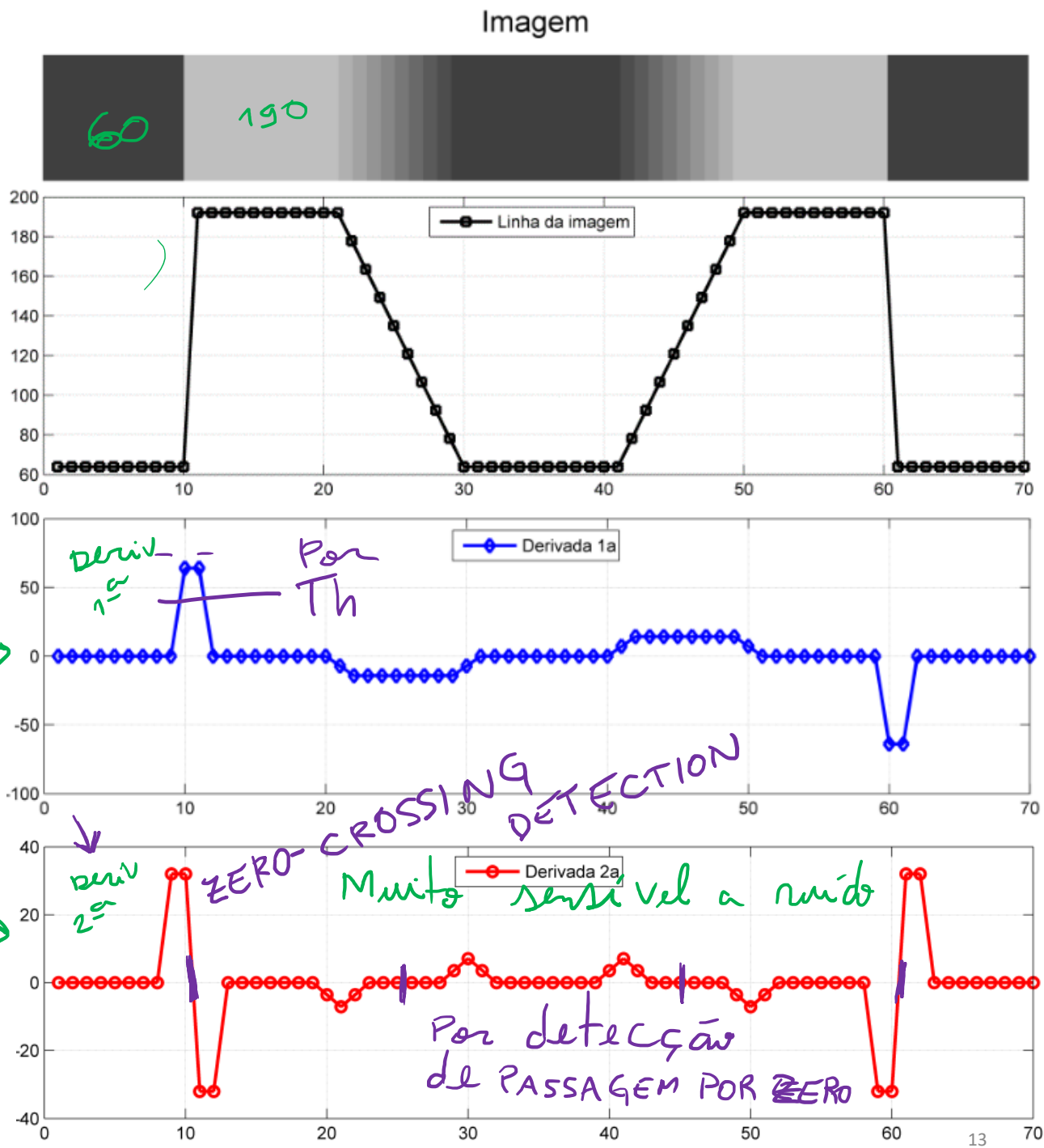
45 -45

Figura adaptada de Rafael C. Gonzalez, Richard E. Woods, Steven L. Eddins, Digital image processing, Pearson Prentice Hall, 3rd ed, 2008, Fig. 10.12.

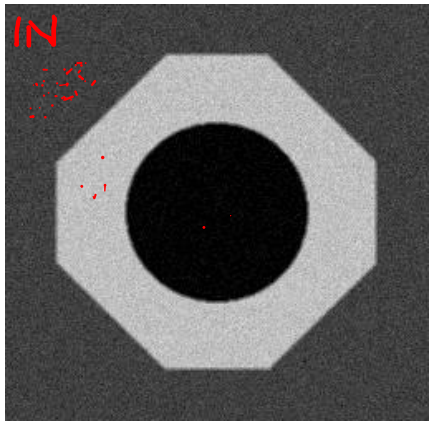
Detecção de bordas por derivada segunda

- A derivada de uma constante (não tem variação) é zero.
- A derivada de um degrau é um impulso.
- A derivada de uma reta (inclinação constante) é uma constante.

```
row = img(1,1:ncol);
% Derivada de
% primeira ordem
for k=2:ncol-1
 d(k)=row(k+1)-row(k-1);
end
d=d/2; % Derivada 1a

% Derivada de
% segunda ordem
for k = 2 : ncol-1
 dd(k)=d(k+1)-d(k-1);
end
dd=dd/2; % Derivada 2a
```



Detecção de bordas por derivada segunda. Operador que implementa derivada segunda de uma imagem: *Laplaciano*



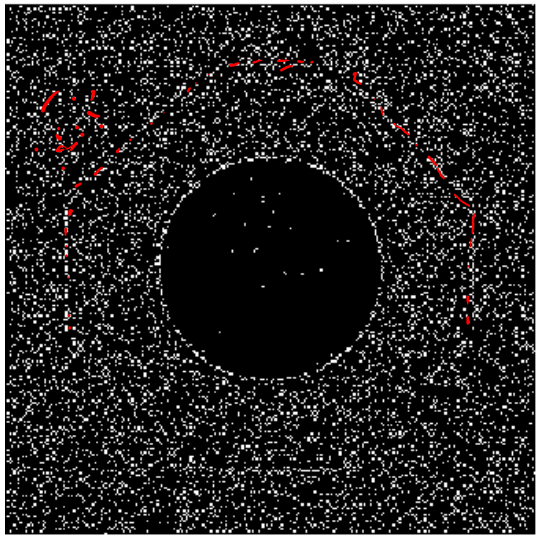
Derivada 2ª

0	1	0
1	-4	1
0	1	0

ZERO-CROSSING DETECTION

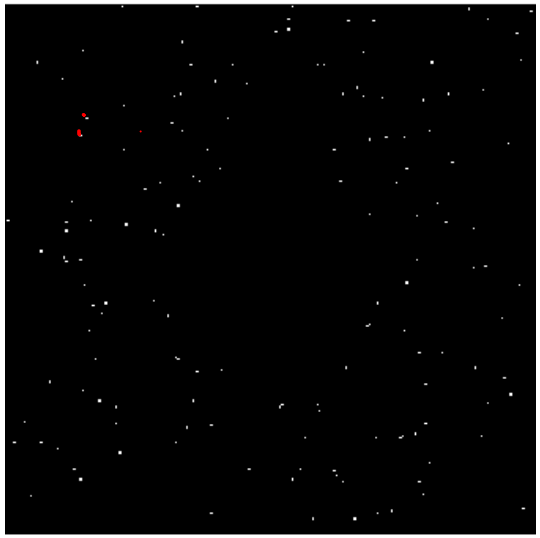
MUITO SENSÍVEL

Limiar baixo na detecção de passagem por zero



MÉDIO

Limiar médio na detecção de passagem por zero



POUCO SENSÍVEL A RUÍDO

Limiar alto na detecção de passagem por zero

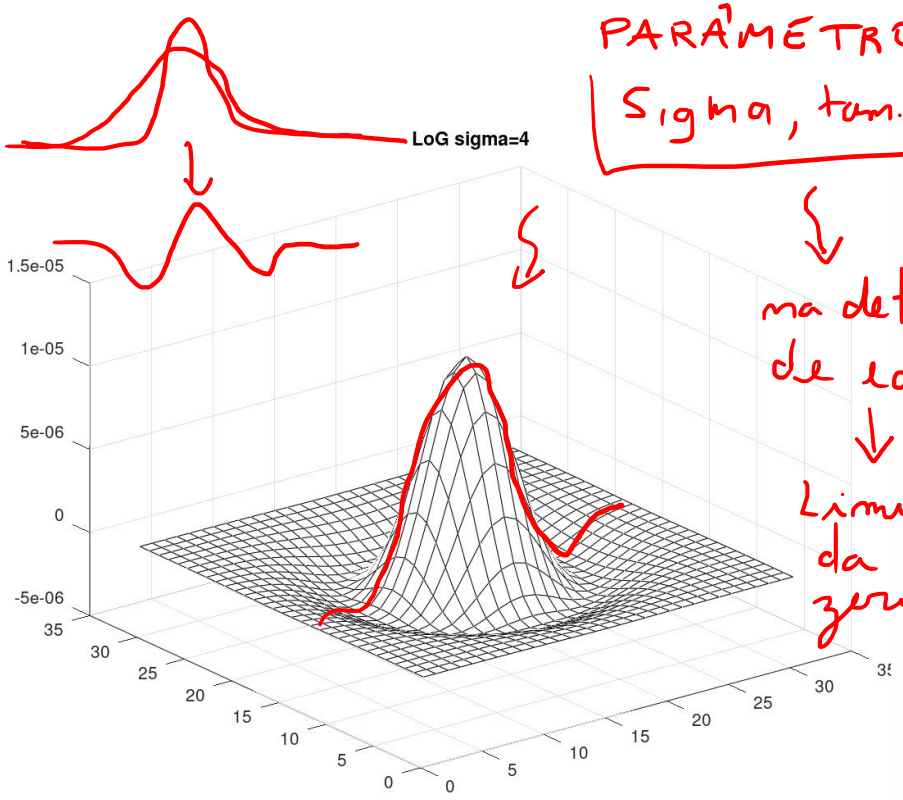
Detecção de bordas por derivada segunda. *Laplaciano* (derivada segunda 'pura') é muito sensível ao ruído. Utiliza-se então a função ***Laplaciano do Gaussino (LoG)***, que combina a derivada segunda com uma suavização. O **LoG** pode ser entendido como um Gaussiano (passa-baixas) associado a um Laplaciano: realiza uma suavização e a derivada segunda.

```
% Gera um LoG de sigma=4 em uma janela 31x31
% É 31x31 pois o objetivo é apenas visualizar
% uma LoG e não convoluir com uma imagem
logf = fspecial('log', [31 31], 4);
p = logf(16,:); %linha central
```

```
%Display
figure
mesh(-logf, 'EdgeColor', 'black')
title('LoG sigma=4')
figure
plot(1:31,-p,'LineWidth',2)
xlim([1 31])
grid
title('Perfil do LoG')
```

PARÂMETROS: Sigma, tam. da janela

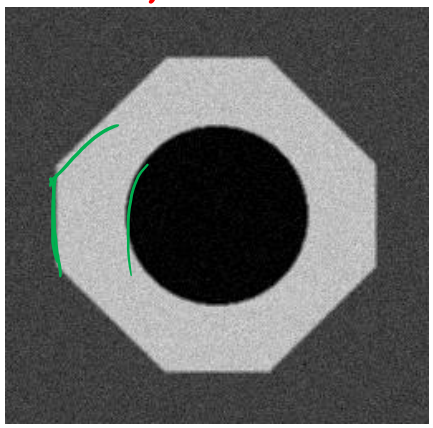
LoG → DoG

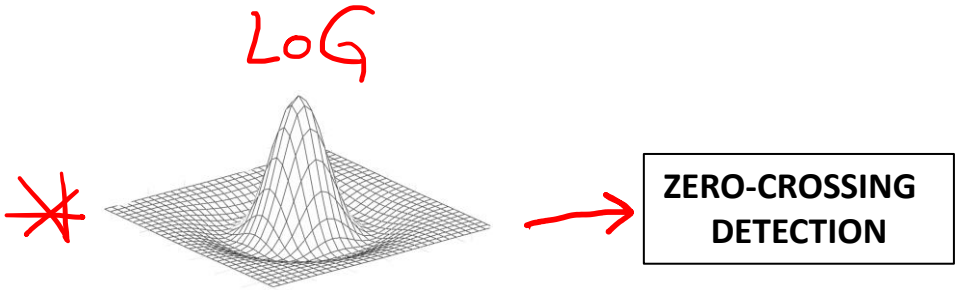


na detec. de edge → Limiar da zero-crossing

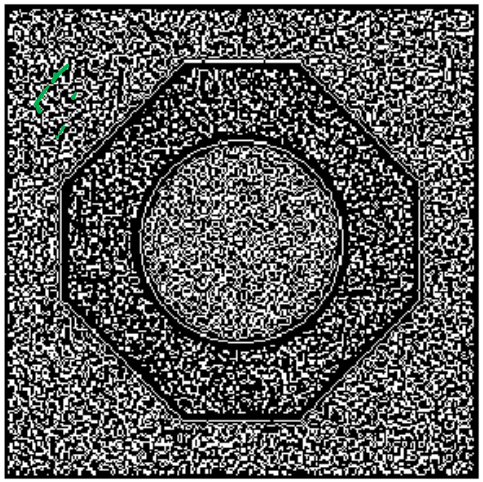
difference of Gaussians

Detecção de bordas por derivada segunda. Laplaciano do Gaussino (LoG)

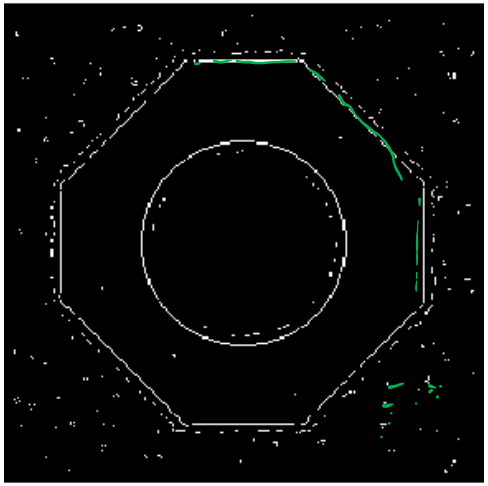




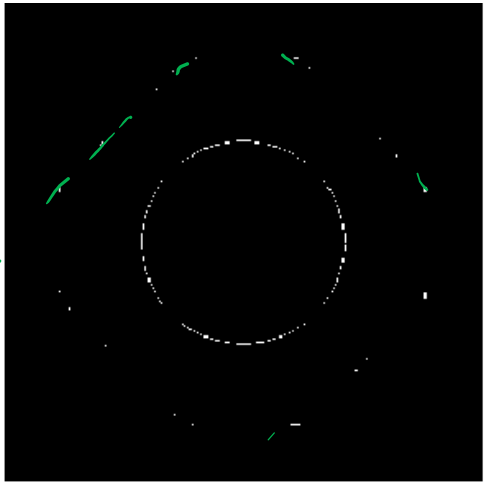
Limiar fixo para detecção de passagem por zero. **Sigma = 1**



Limiar fixo para detecção de passagem por zero. **Sigma = 2**

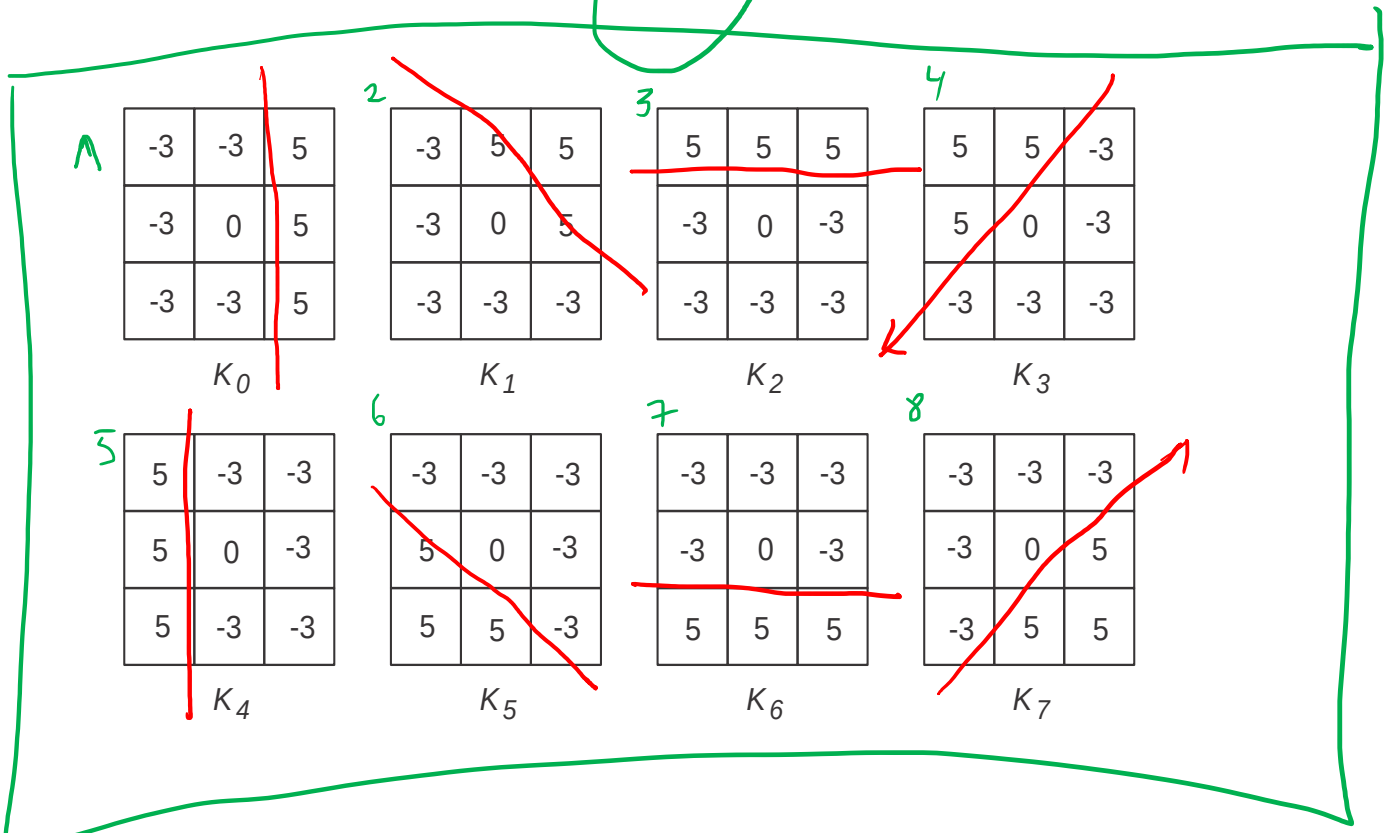


Limiar fixo para detecção de passagem por zero. **Sigma = 3**



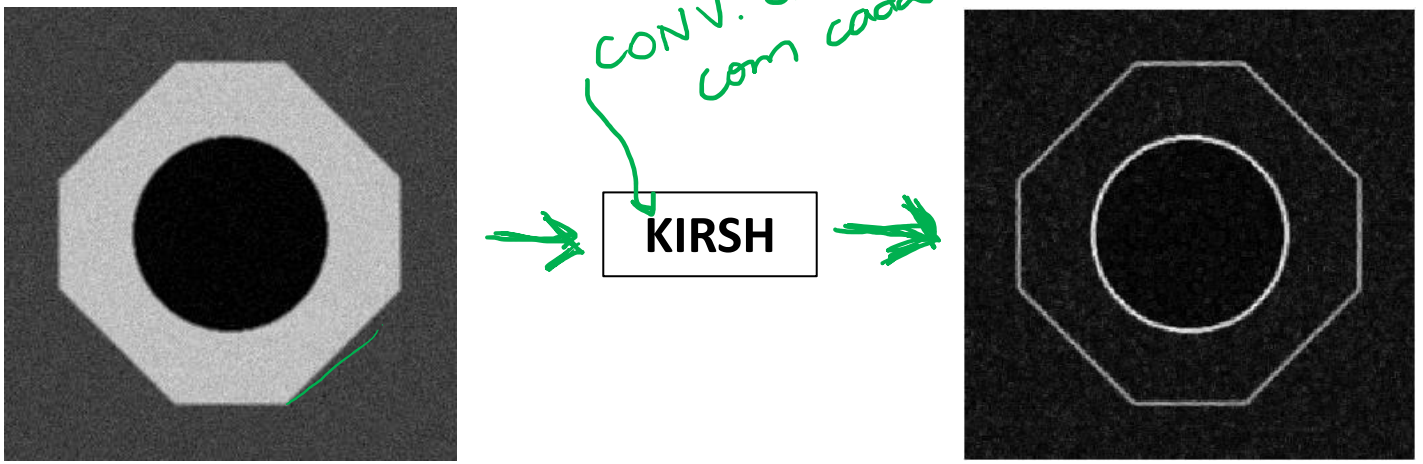
Detecção de bordas por compass operator.

Máscaras compass do método de ***Kirsh:***



A saída do detector de bordas de Kirsh é o máximo das saídas das oito máscaras. A orientação é a da própria máscara que forneceu o máximo (múltiplos de 45º).

Detecção de bordas por compass operator.
Máscaras compass do método de ***Kirsh:***

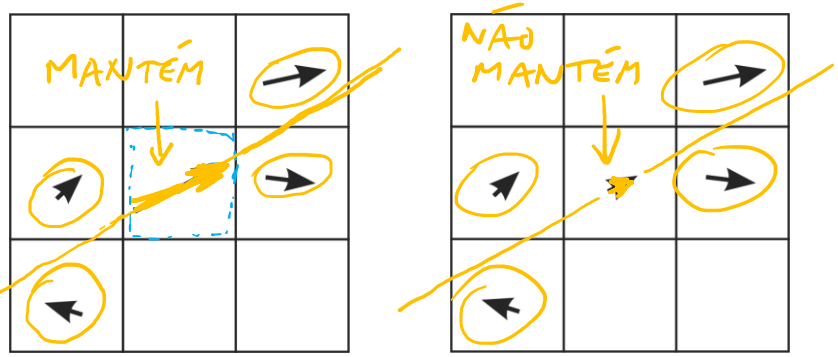


Na prática, o método de Kirsh fornece resultados bastante parecidos aos dos filtros de gradiente. Por isso, para aplicações em geral, o método de Sobel é mais indicado que os baseados em compass masks.

Detecção de bordas pelo método de **Canny**. É o mais utilizado.
É básicamente um método para limiarizar a magnitude do gradiente.
Há muitas variações da implementação. Uma possibilidade:

Low-Pass

1) LPF usando o Gaussiano.
2) Calcula magnitude do gradiente usando máscaras Sobel.
3) Nonmaximal supression: um pixel só é mantido (considerado borda e analisado no próximo passo) se tiver magnitude do gradiente maior que seus vizinhos determindos pela **direção** do seu gradiente.

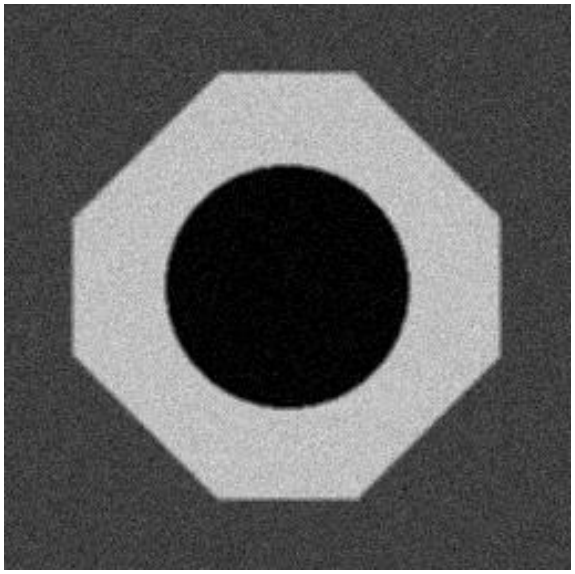


"Faz edge detec. aê"
↓
"Bedê, vou passar um Canny"

4) Histeresys threshoding: limiares T_L e T_H. Se > T_H → strong edge
Se entre T_L e T_H → weak edge
5) Edge linking: incorpora pixels de weak edge dentro de uma vizinhança 8 de um strong pixel.

Detecção de bordas pelo método de **Canny**. É o mais utilizado.
É básicamente um método para limiarizar a magnitude do gradiente.
Há muitas variações da implementação

IN



OUT