XỬ LÝ ẢNH

(Image Processing)

Bài 7: PHÁT HIỆN BIÊN

(Edge detection)

Nội dung

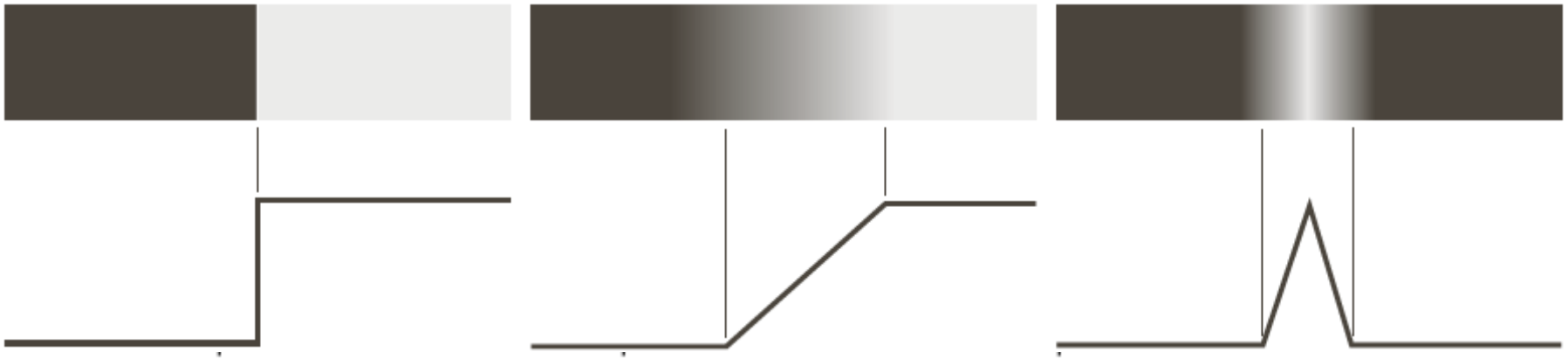
- Vai trò của biên và cách tiếp cận chung
- Phát hiện biên
  - Bộ lọc đạo hàm bậc 1
    - Bô lọc Robert Cross Gradient
    - Bô lọc Sobel
    - Bộ lọc Prewitt
  - Bộ lọc đạo hàm bậc 2
    - Bộ lọc Laplacian

Biên là gì (edge/contour) ?

- Nơi có sự thay đổi cường độ sáng trong ảnh
- Thường sảy ra ở ranh giới giữa các vùng khác nhau trong ảnh
- Biên có thể tạo ra bởi nhiều lý do:

Biên là gì?

- **Biên (edge)**: là nơi **cường độ sáng thay đổi đột ngột** trong ảnh (ví dụ ranh giới giữa đối tượng và nền).
- **Mục đích của lọc sắc nét:** làm nổi bậc các chi tiết trong ảnh, tức là làm nổi các biên ảnh.
- Do vậy, lọc sắc nét cũng là kỹ thuật làm nổi biên ảnh, xác định sự thay đổi mức xám các pixel trong ảnh.



a, Đường biên lý tưởng
Biên nhảy vọt (skip)

b, Đường biên bậc thang
Biên thoai thoải (ramp)

c, Đường biên thực
Biên mái nhà (roof)

Biên là gì?

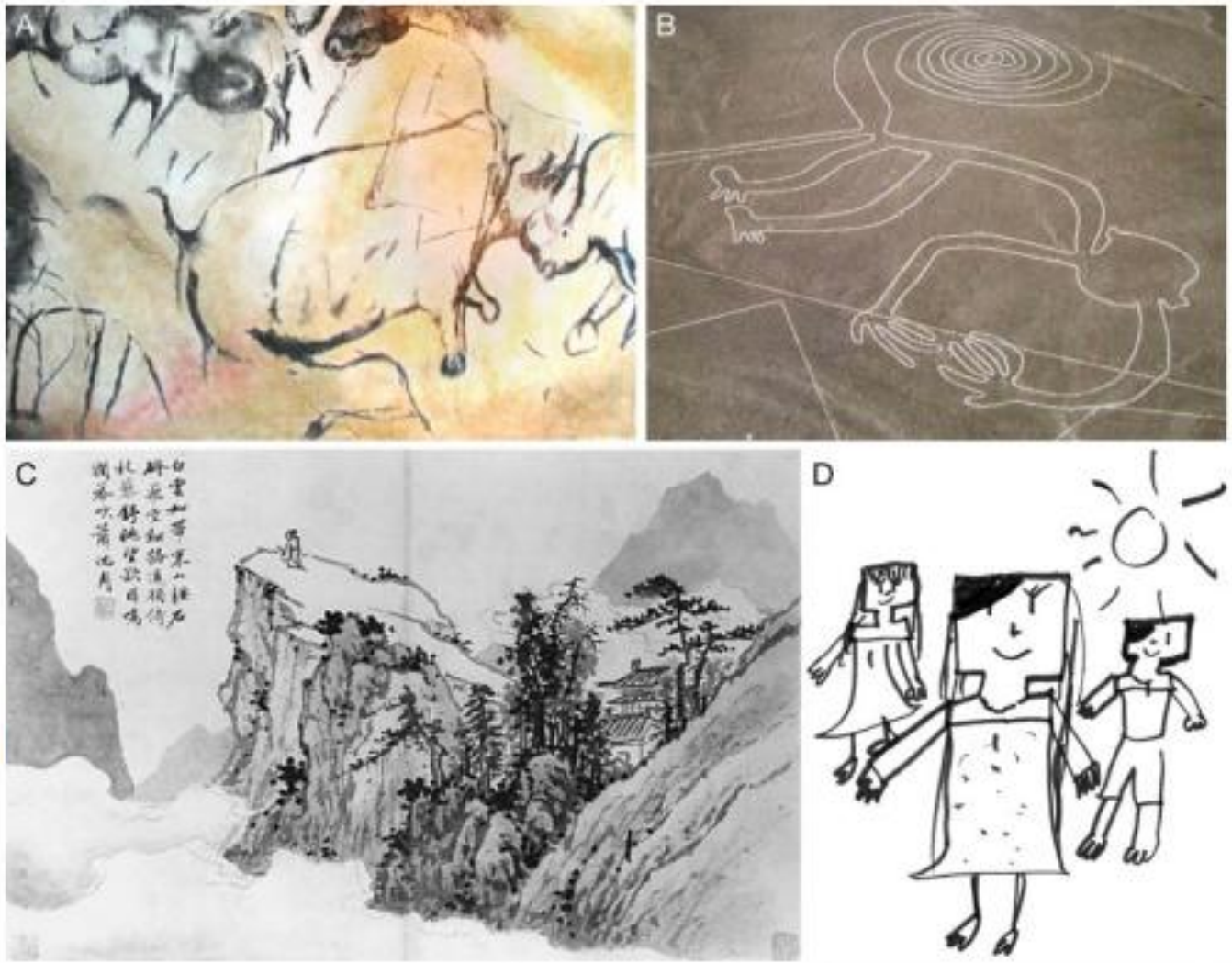
0	0	0	33
0	0	45	78
0	45	23	33
0	0	42	76
0	0	0	38

Vai trò của biên

- What do you see ?

Vai trò của biên

(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 I. Lleras (2010 A.D.).



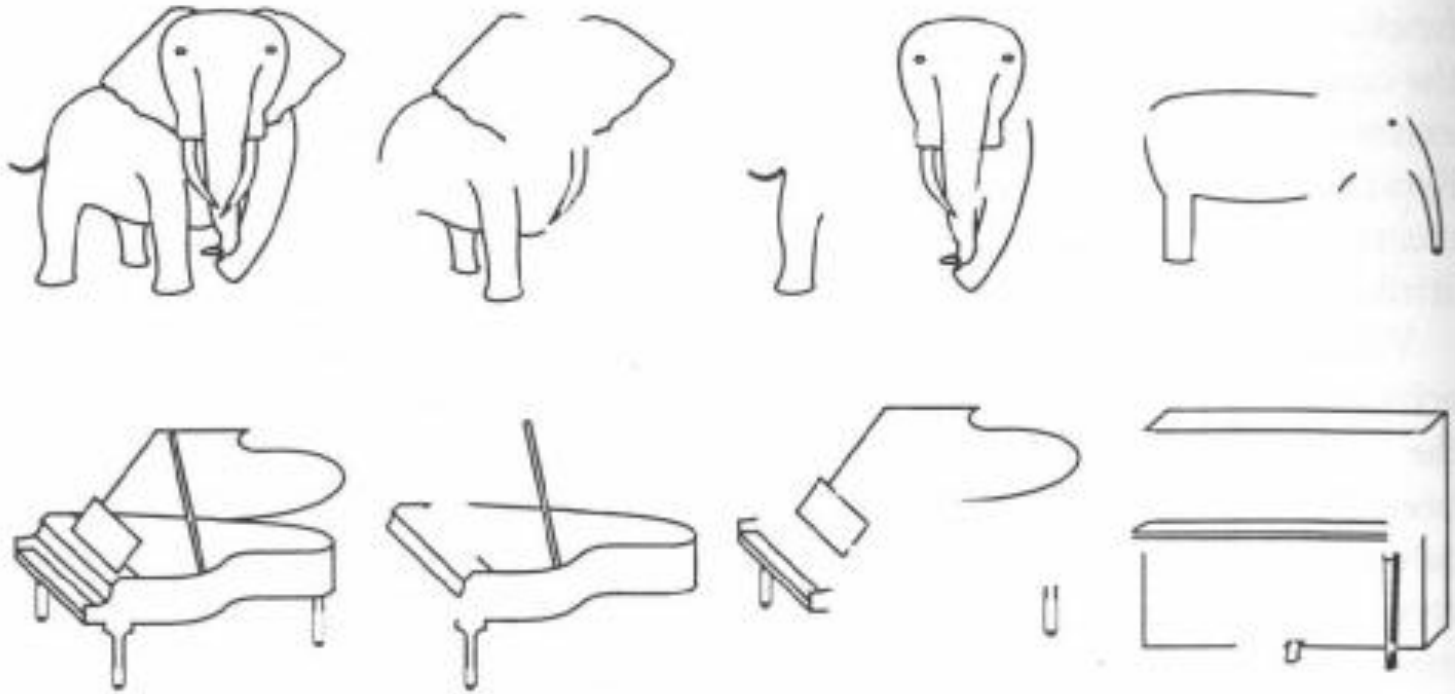


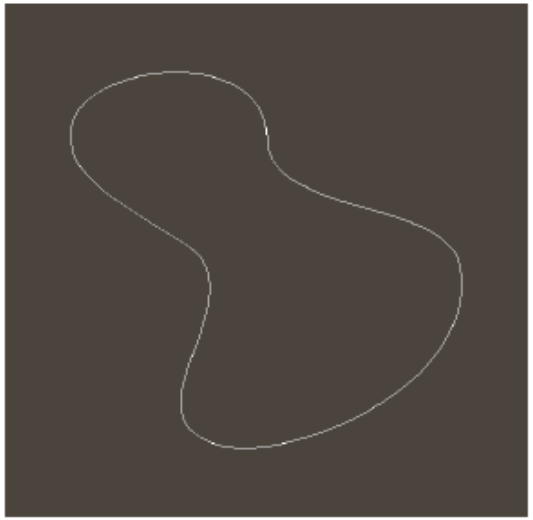
Figure 4.14
Complementary-part images. From an original intact image (left column), two complemen-

Can we recognize these objects?

Ý nghĩa của bài toán phát hiện biên:

- Phát hiện biên rất quan trọng trong nhận dạng đối tượng, phân đoạn ảnh, thị giác máy tính.
- Về mặt toán học, biên liên quan đến đạo hàm bậc nhất hoặc bậc hai của ảnh.

Ảnh gốc



Đường biên

Phân vùng

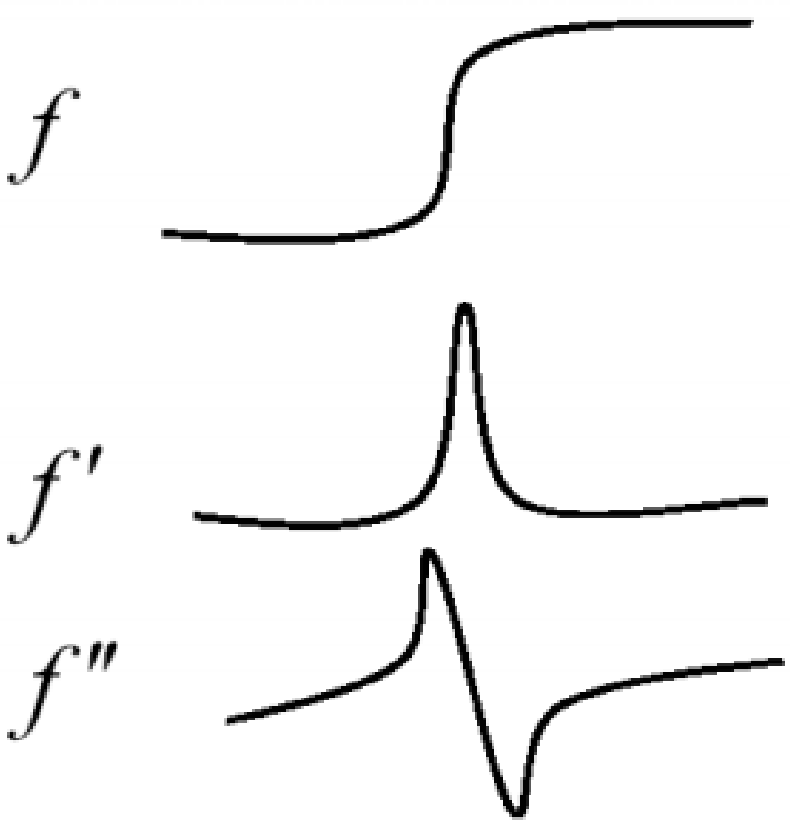
Ex: Edge Detection

Ex: Edge Detection

Ex: Edge Detection

Phát hiện biên ?

- Biên là vị trí có sự thay đổi nhanh về cường độ
  - Đạt cực trị trên đạo hàm bậc 1
  - Đi qua không (zero-crossing) ở đạo hàm bậc 2



Đạo hàm bậc 1

- Derivative in 1D:

$$\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x) = f_x$$

- Discrete derivative in 1D

$$\frac{df}{dx} = \frac{f(x) - f(x-1)}{1} = f'(x)$$

$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$

- **Ý nghĩa:** đạo hàm tại điểm x được xấp xỉ bằng giá trị hiện tại trừ đi giá trị phía trước.

Bộ lọc tính đạo hàm bậc 1

- Backward filter:

$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$ [0 1 -1]

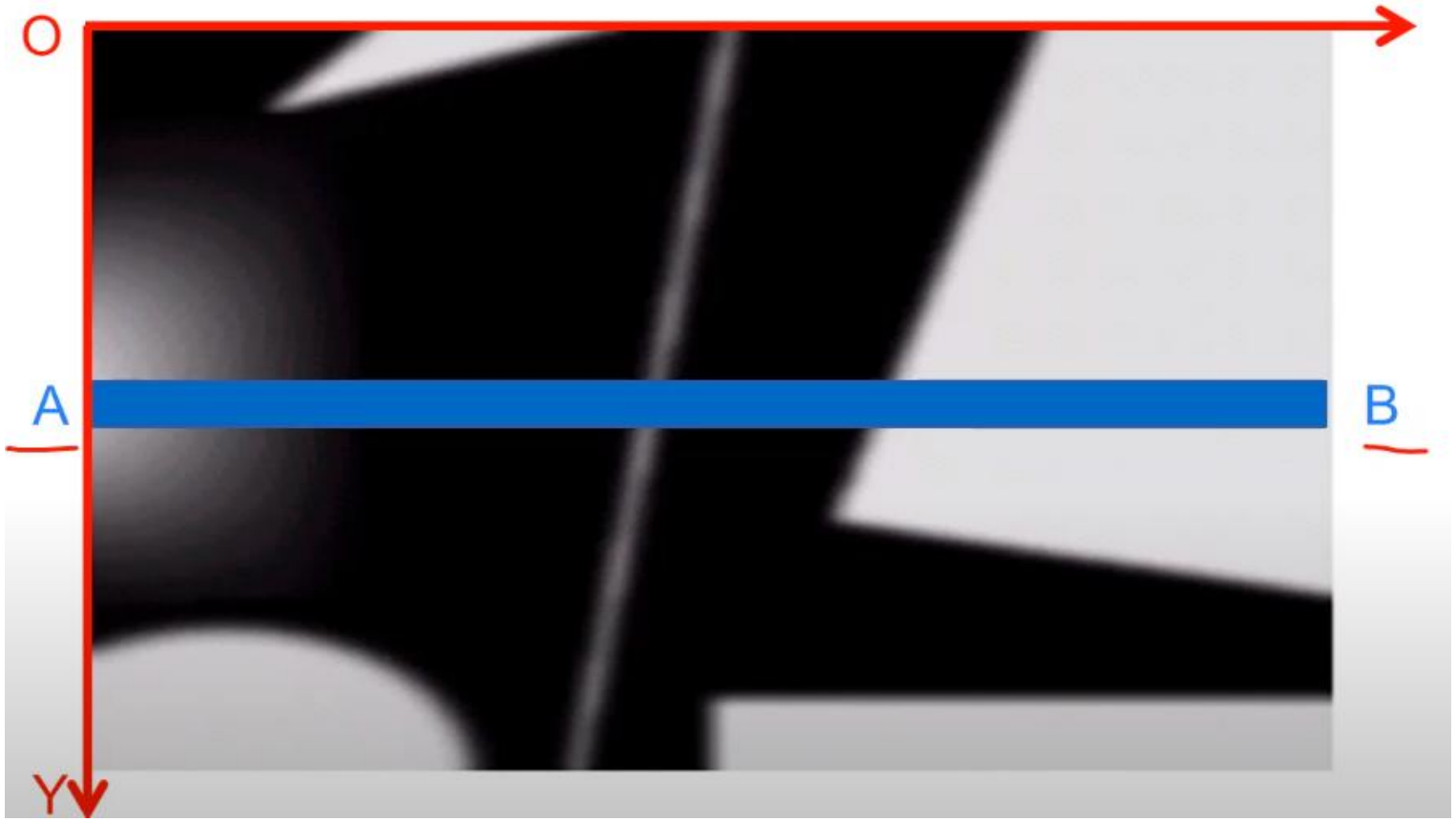
- Forward:

$$\frac{df}{dx} = f(x) - f(x+1) = f'(x)$$ [-1 1 0]

- Central:

$$\frac{df}{dx} = f(x+1) - f(x-1) = f'(x)$$ [1 0 -1]

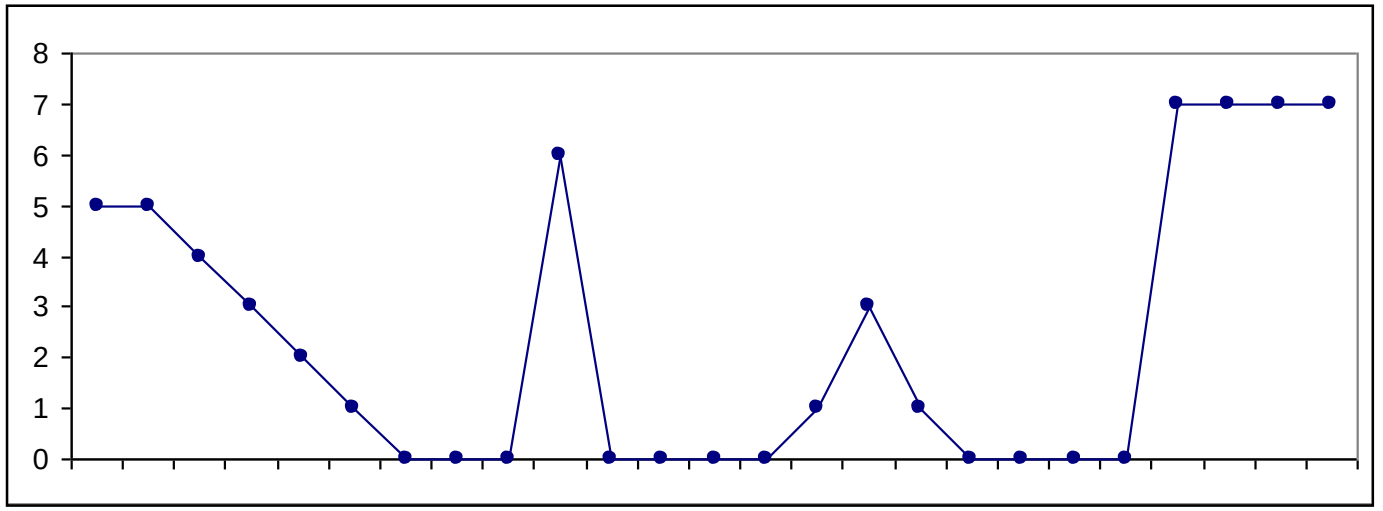
VÍ DỤ



Giá trị mức xám của các pixel trên dòng AB trích từ ảnh.

5	5	4	3	2	1	0	0	0	6	0	0	0	0	1	3	1	0	0	0	0	7	7	7	7

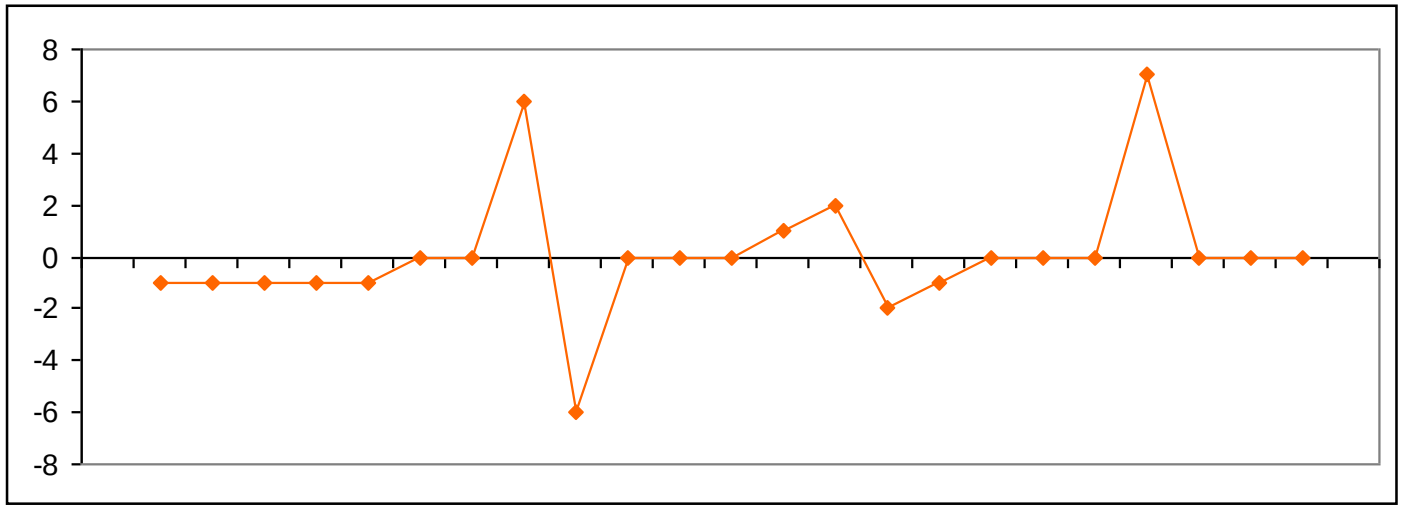
VÍ DỤ



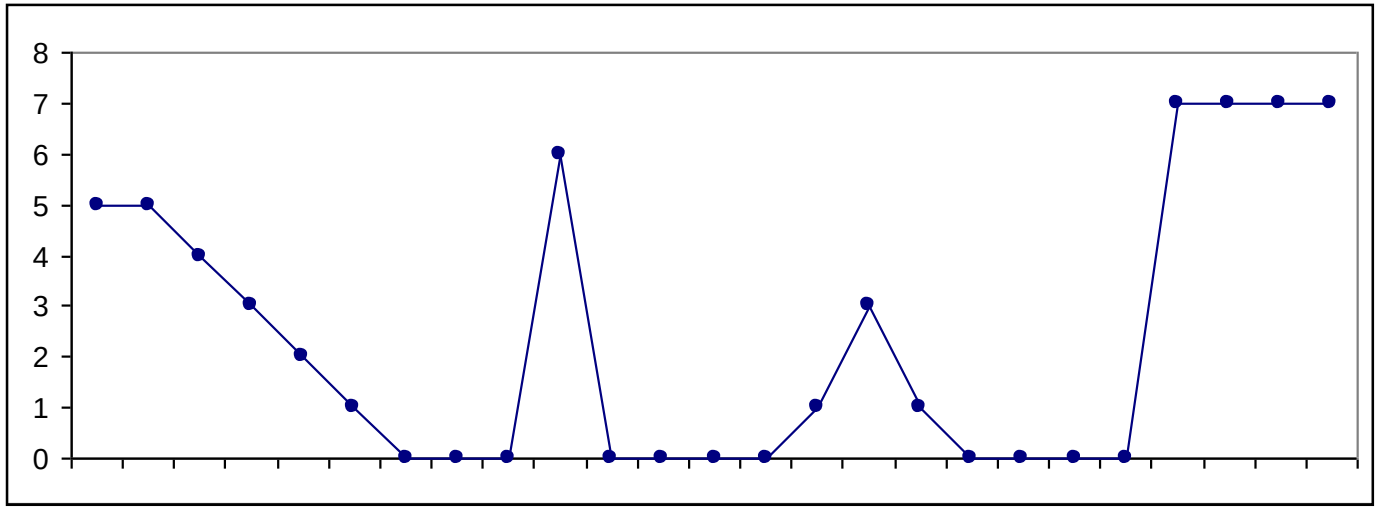
5	5	4	3	2	1	0	0	0	6	0	0	0	0	1	3	1	0	0	0	0	7	7	7	7

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

	0	-1	-1	-1	-1	0	0	6	-6	0	0	0	1	2	-2	-1	0	0	0	7	0	0	0	



5	5	4	3	2	1	0	0	0	6	0	0	0	0	1	3	1	0	0	0	0	7	7	7	7



$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

	0	-1	-1	-1	-1	0	0	6	-6	0	0	0	1	2	-2	-1	0	0	0	7	0	0	0	

- **Nhận xét:**
  - Bằng 0 trong vùng có mức xám không đổi.
  - Khác 0 tại điểm bắt đầu chuyển sang vùng thoai thoải và vùng biên nhảy vọt
  - Khác 0 trong vùng biên thoai thoải.

Ví dụ:

- Backward filter: [0 1 -1]

$$f(x) = 10 \quad 15 \quad 10 \quad 10 \quad 25 \quad 20 \quad 20 \quad 20$$
$$f'(x) = 0 \quad 5 \quad -5 \quad 0 \quad 15 \quad -5 \quad 0 \quad 0$$

f(x) :	**0**	**0**	**0**	**0**	**0**	**50**	**50**	**50**	**50**	**50**
f'(x):	**0**	**0**	**0**	**0**	**0**	**50**	**0**	**0**	**0**	**0**

Đạo hàm rời rạc trên ảnh

- Ảnh xám: $I(x, y)$.
- Biên được xác định tại điểm có **đạo hàm lớn**:
  - Đạo hàm theo x:

$$G_x = \frac{\partial I}{\partial x}$$

  - Đạo hàm theo y:

$$G_y = \frac{\partial I}{\partial y}$$

- **Độ lớn gradient**:

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

- **Hướng biên**:

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

Gradient của ảnh

- Gradient theo hai hướng

$$G_x = \frac{\partial f}{\partial x} \qquad G_y = \frac{\partial f}{\partial y}$$

- Biên độ của gradient

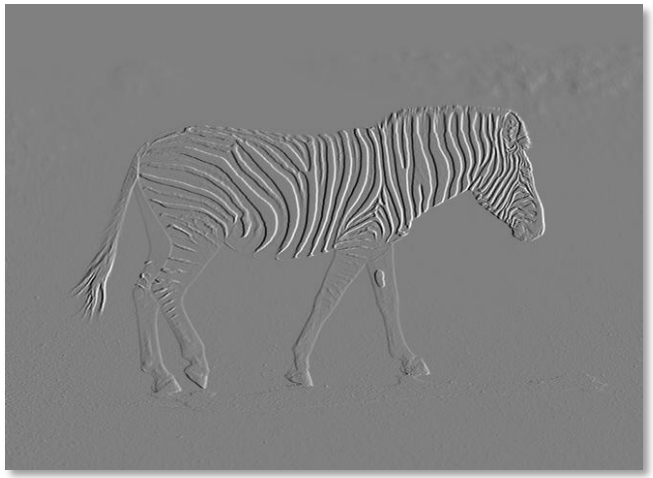
$$mag(\nabla f) = \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{1/2} \qquad mag(\nabla f) = \left[\left|\frac{\partial f}{\partial x}\right| + \left|\frac{\partial f}{\partial y}\right|\right]$$

- Tính gradient rời rac

$$G_x = f(x+1, y) - f(x, y) \qquad G_y = f(x, y+1) - f(x, y)$$

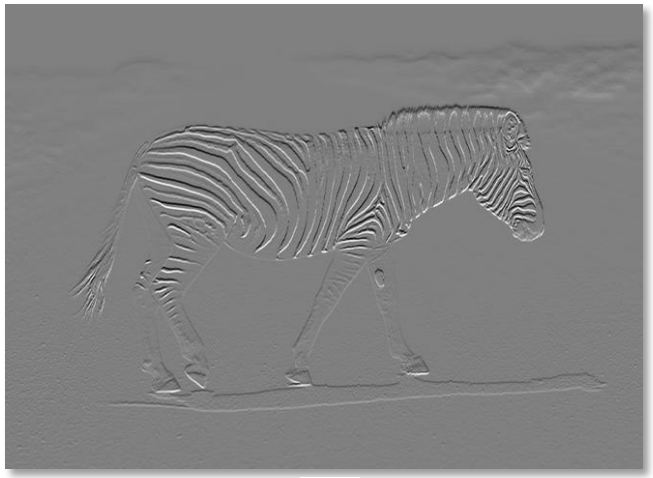
Image gradient

f



$\frac{\partial f}{\partial x}$

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



$\frac{\partial f}{\partial y}$

Original Image

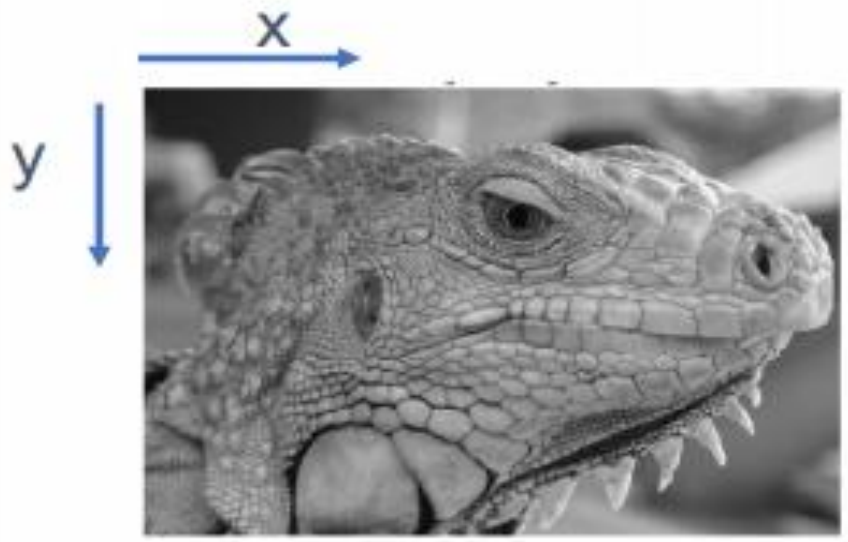
Gradient magnitude

x-direction

y-direction

- Which one is the gradient in the x-direction? How about y-direction?

Đạo hàm bậc 1 theo x và y



1
0
-1

1	0	-1

Đạo hàm bậc 1 theo x và y

(a) Kernel đạo hàm theo y

$$K_y = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

- Thực hiện phép trừ: **pixel phía trên – pixel phía dưới**.
- Nếu có sự thay đổi dọc theo trục **y** → gradient lớn → phát hiện biên ngang (horizontal edges).

Đạo hàm bậc 1 theo x và y

(b) Kernel đạo hàm theo x

$$K_x = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

- Thực hiện phép trừ: **pixel bên phải – pixel bên trái**.
- Nếu có sự thay đổi dọc theo trục **x** → gradient lớn → phát hiện biên dọc (vertical edges).

Với mỗi pixel, kernel trượt qua ảnh và tính toán giá trị gradient.Nếu ảnh có sự thay đổi cường độ mạnh tại vùng đó (ví dụ ranh giới mắt/thân con vật), gradient sẽ lớn.Nếu vùng phẳng (độ sáng ít thay đổi), gradient ≈ 0.

Đạo hàm bậc 1 theo x và y

- Sau khi tính được G_x và G_y, ta tính độ lớn gradient:

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

- Đây là ảnh biên hoàn chỉnh, kết hợp biên theo cả hai hướng.

```
# Kernel đạo hàm theo x và y
kx = np.array([[1, 0, -1]], dtype=np.float32)
ky = np.array([[1], [0], [-1]], dtype=np.float32)

# Tính gradient
gx = cv2.filter2D(img, cv2.CV_32F, kx)
gy = cv2.filter2D(img, cv2.CV_32F, ky)

# Biên độ gradient
mag = np.sqrt(gx**2 + gy**2)
```

Bộ lọc Robert (Roberts Cross Operator)

- **Robert Filter** (xấp xỉ đạo hàm bậc nhất của ảnh-1965)
- Tính **gradient theo hướng chéo (diagonal)** chứ không chỉ ngang/dọc.

f(x, y)	f(x+1, y)
f(x, y+1)	f(x+1, y+1)

  - Theo hướng chéo thứ nhất: $G_{cross1} = |f(x+1, y+1) - f(x, y)|$

-1	0
0	1

  - Theo hướng chéo thứ hai: $G_{cross2} = |f(x, y+1) - f(x+1, y)|$

0	-1
1	0

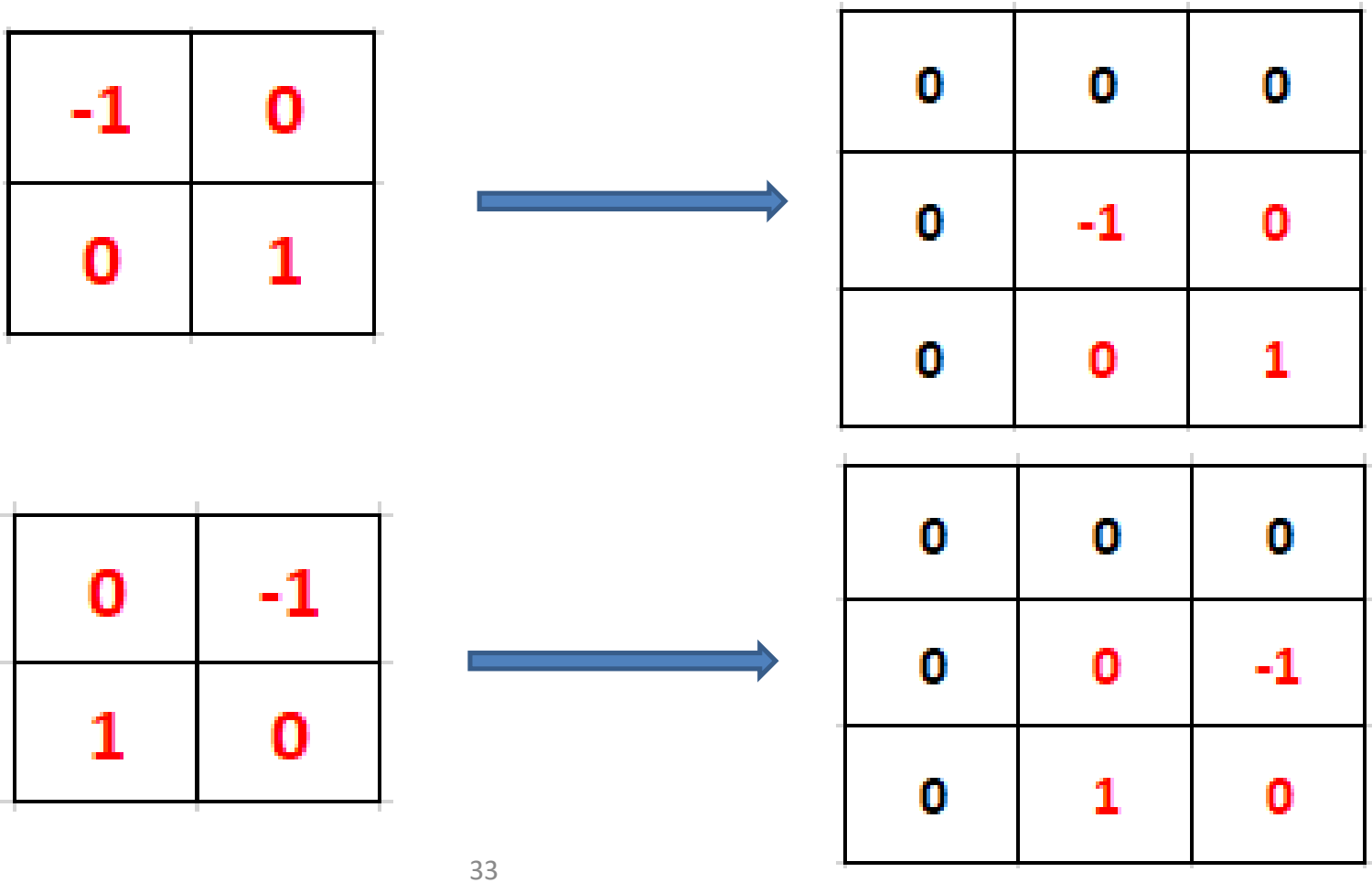
Bộ lọc Robert (Roberts Cross Operator)

- Ưu điểm:
  - Cực kỳ đơn giản, chỉ cần phép tính với 2×2 pixel.
  - Phát hiện biên chéo khá tốt.
  - Dễ triển khai (1965, thời kỳ máy tính còn hạn chế).

```
# Kernel Robert
kx = np.array([[-1, 0],
               [ 0, 1]], dtype=np.float32)

ky = np.array([[ 0, -1],
               [ 1,  0]], dtype=np.float32)

# Áp dụng filter2D
gx = cv2.filter2D(img, cv2.CV_32F, kx)
gy = cv2.filter2D(img, cv2.CV_32F, ky)

# Magnitude
mag = np.sqrt(gx**2 + gy**2)
mag = np.uint8((mag / mag.max()) * 255)
```

Bô lọc Robert

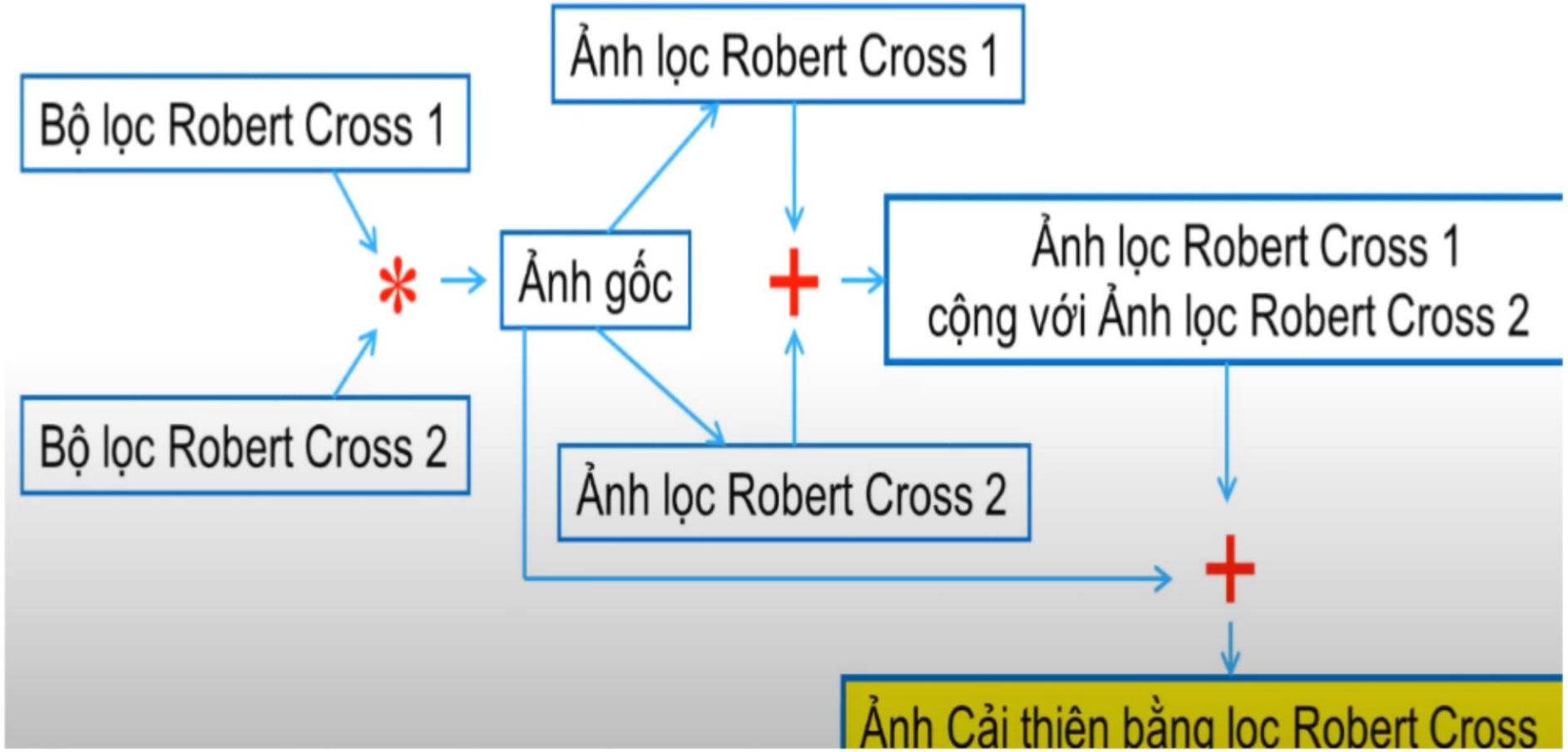
- Robert cross gradient kích thước 3 x 3



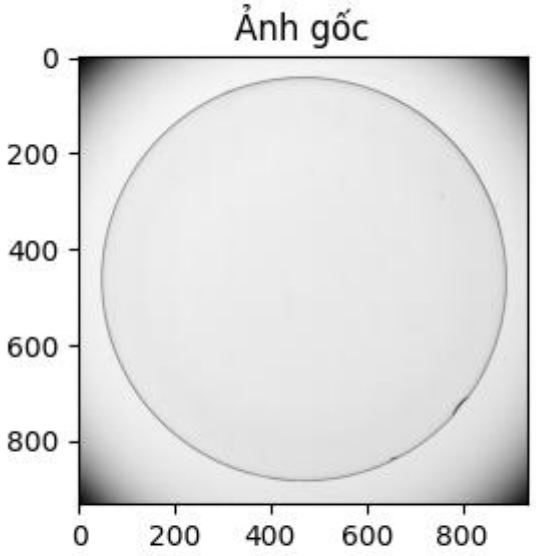
Bộ lọc Robert cross gradient kích thước 3 x 3

Bô lọc Robert

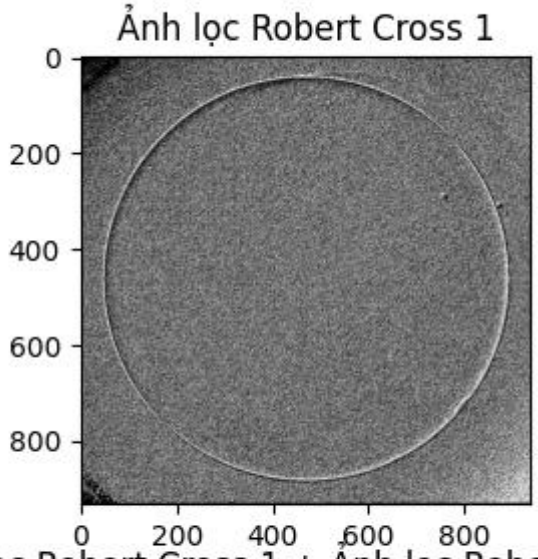
- Lọc sắc nét - Lọc đạo hàm bậc 1 - Bộ lọc Robert Cross Gradient
- Các bước thực hiện

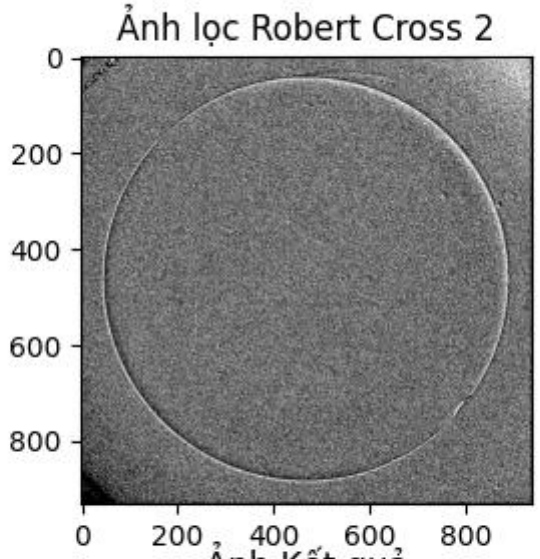


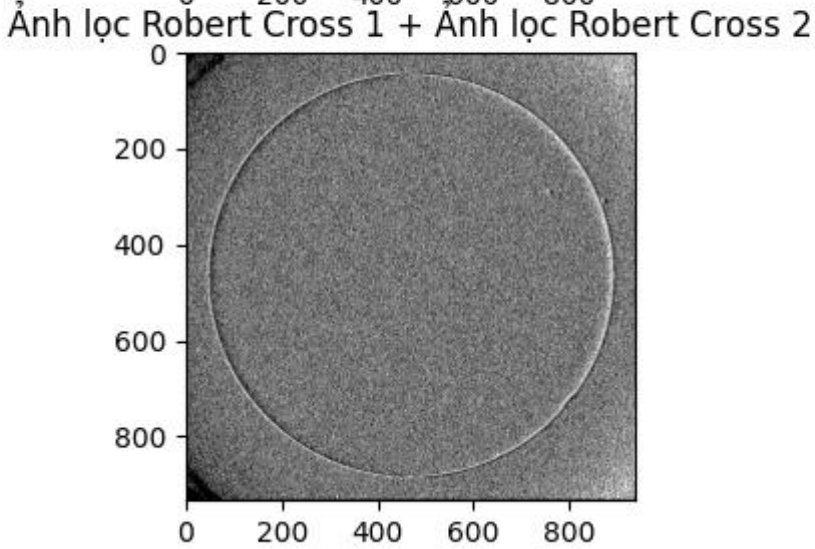
Bô lọc Robert – kết quả

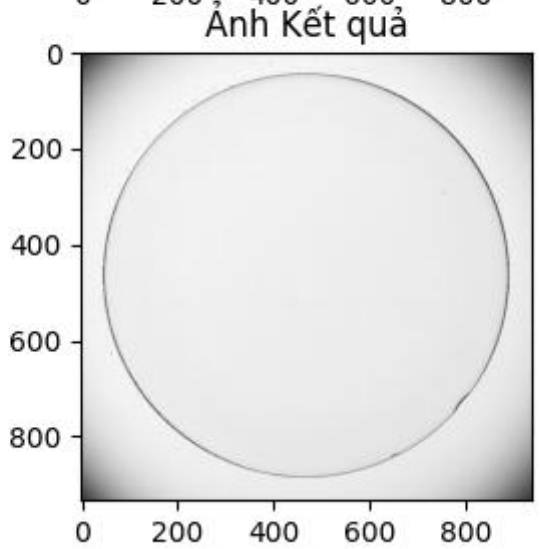


Kính áp tròng









Bộ lọc Sobel (Sobel Operator)

- Được đề xuất bởi Irwin Sobel & Gary Feldman (1970).
- Là một **toán tử phát hiện biên** dựa trên đạo hàm bậc nhất (gradient).
- Cải tiến từ **Prewitt** bằng cách thêm **trọng số 2 ở giữa** → giúp **làm mượt (smoothing)**, giảm nhiễu.
- Do đó, Sobel vừa tính đạo hàm vừa lọc nhiễu nhẹ.

Bộ lọc Sobel (Sobel Operator)

(a) Theo hướng X (Gradient theo X → phát hiện biên dọc)

$$K_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

- Dòng trên = hệ số âm → trừ các pixel phía trên.
- Dòng dưới = hệ số dương → cộng các pixel phía dưới.
- Kết quả: nhấn mạnh **sự thay đổi theo trục X** → phát hiện biên **dọc**.

(b) Theo hướng Y (Gradient theo Y → phát hiện biên ngang)

$$K_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

- Cột trái = hệ số âm → trừ pixel bên trái.
- Cột phải = hệ số dương → cộng pixel bên phải.
- Kết quả: nhấn mạnh **sự thay đổi theo trục Y** → phát hiện biên **ngang**.

Đặc tính của Sobel

- Tổng các phần tử trong mỗi kernel = 0 → lọc biên tốt (vì vùng đồng nhất sẽ cho giá trị 0).
- Trọng số "2" ở giữa → có tác dụng làm **mượt (smoothing)** theo chiều vuông góc → giảm nhiễu so với Prewitt.
- Kích thước nhỏ gọn (3×3) → tính toán nhanh.

Kết hợp 2 hướng

- Sau khi lọc bằng K_x và K_y, ta tính:

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

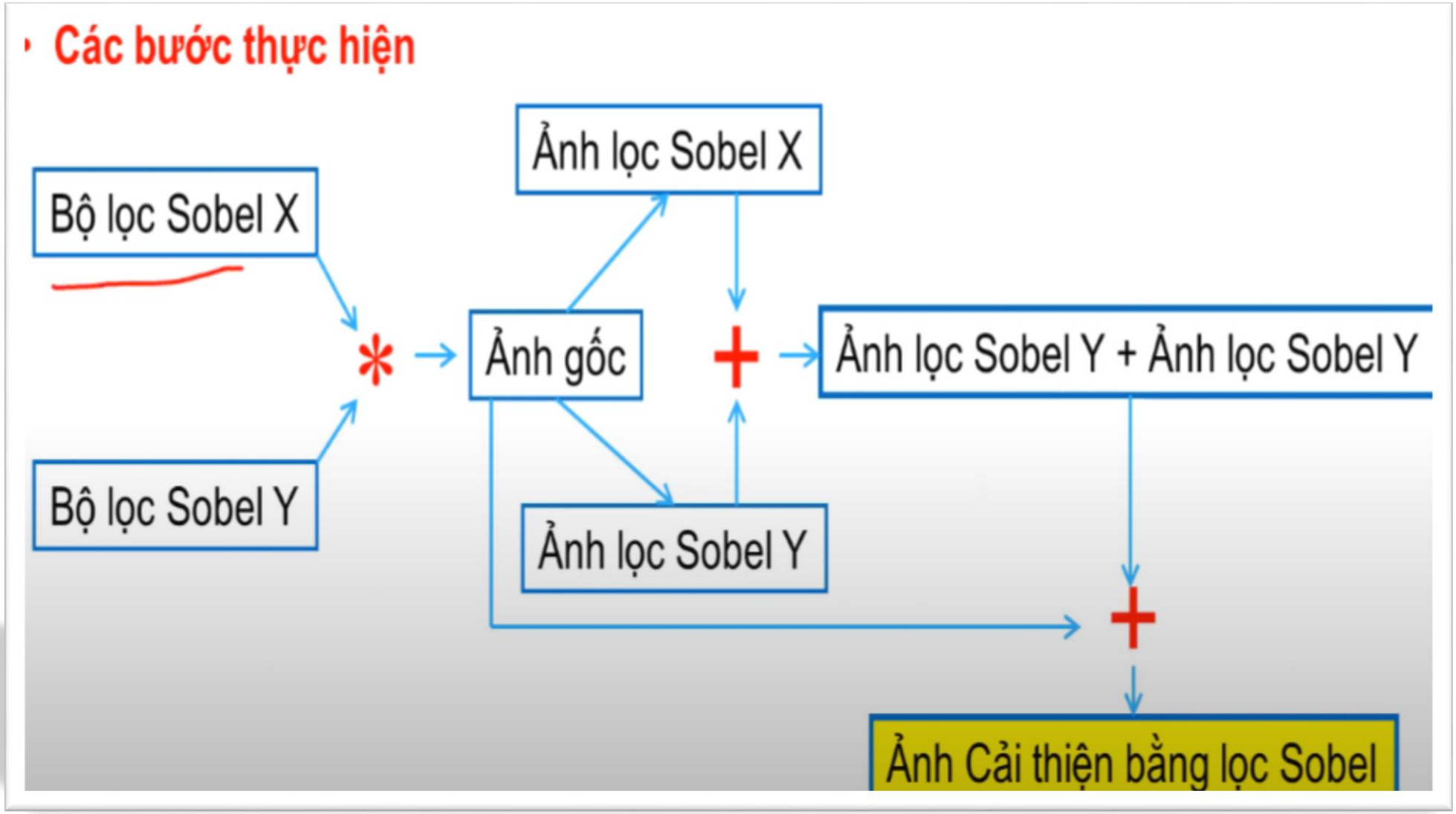
hoặc gần đúng:

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

- Đây chính là ảnh biên tổng hợp.

```python
# Sobel theo X và Y
sobelx = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=3)  # theo X
sobely = cv2.Sobel(img, cv2.CV_64F, 0, 1, ksize=3)  # theo Y

# Biên độ tổng hợp
sobel = cv2.magnitude(sobelx, sobely)
```

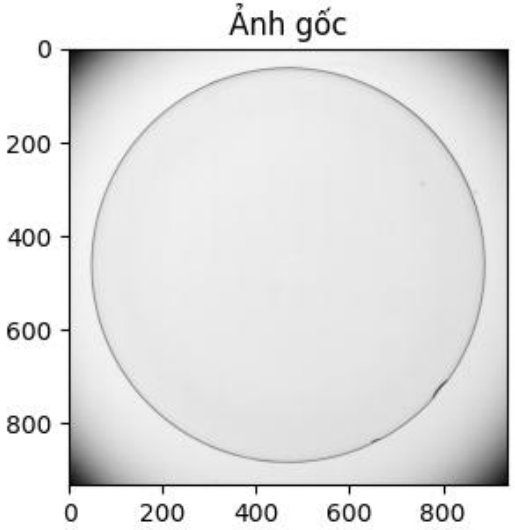
Bô lọc Sobel

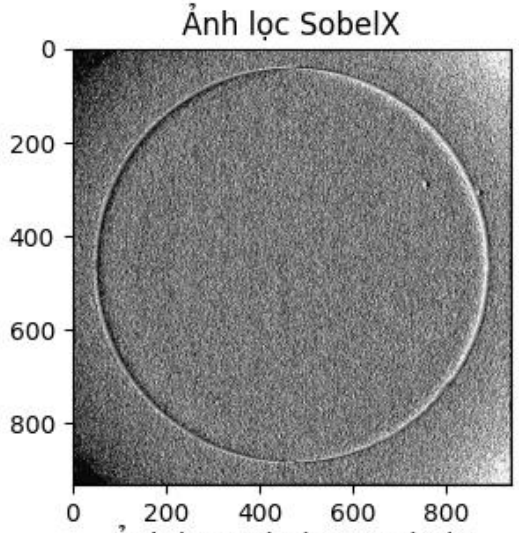

Các bước thực hiện
Bộ lọc Sobel X
Bộ lọc Sobel Y
Ảnh gốc
Ảnh lọc Sobel X
Ảnh lọc Sobel Y
Ảnh lọc Sobel Y + Ảnh lọc Sobel Y
Ảnh Cải thiện bằng lọc Sobel

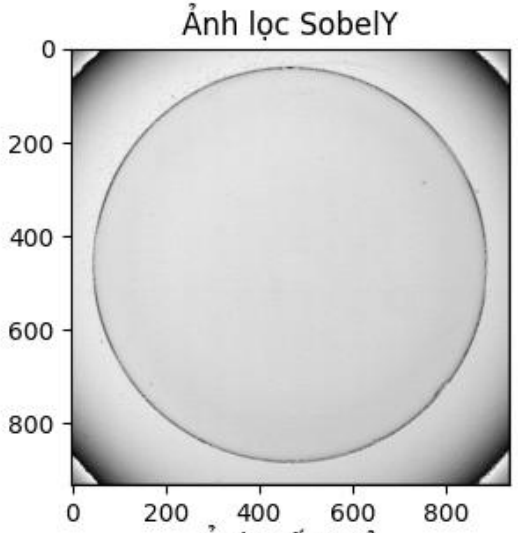
Bộ lọc Sobel

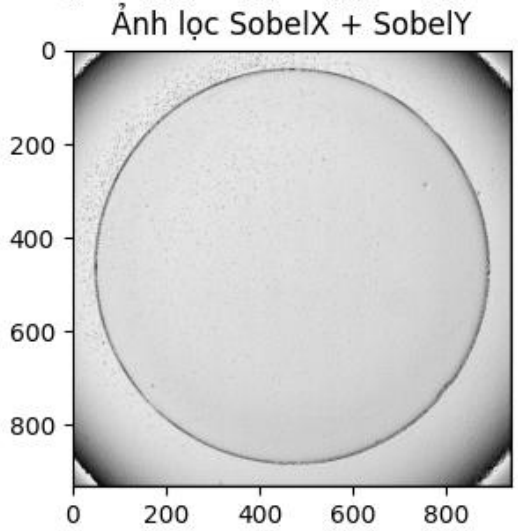
```
# 2. Sobel X và Y
sobel_x = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=3)
sobel_y = cv2.Sobel(img, cv2.CV_64F, 0, 1, ksize=3)

sobel_x = cv2.convertScaleAbs(sobel_x)
sobel_y = cv2.convertScaleAbs(sobel_y)

# 3. Kết hợp Sobel X và Y
sobel_combined = cv2.addWeighted(sobel_x, 0.5, sobel_y, 0.5, 0)

# 4. Cộng Sobel vào ảnh gốc để tăng cường chi tiết biên
enhanced_img = cv2.add(img, sobel_combined)
```

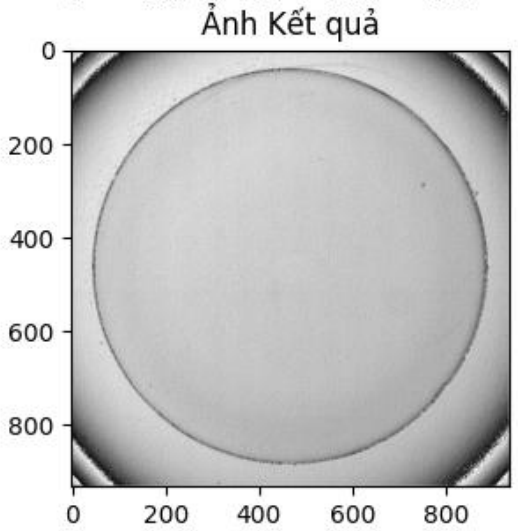
Bô lọc Sobel – kết quả











Prewitt Filter

- Giống như **Sobel**, **Prewitt** cũng dựa trên đạo hàm bậc nhất (gradient) để tìm biên.
- Điểm khác biệt chính: Prewitt dùng giá trị trung bình đơn giản (1, 0, -1) thay vì có trọng số (như Sobel dùng 2 ở giữa).
- Do đó, Prewitt đơn giản hơn nhưng kém "mượt" hơn Sobel trong việc giảm nhiễu.

Prewitt Filter

- Sử dụng 2 mặt nạ và xấp xỉ đạo hàm theo 2 hướng x và y là:

$$H_x = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix}$$

$$H_y = \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

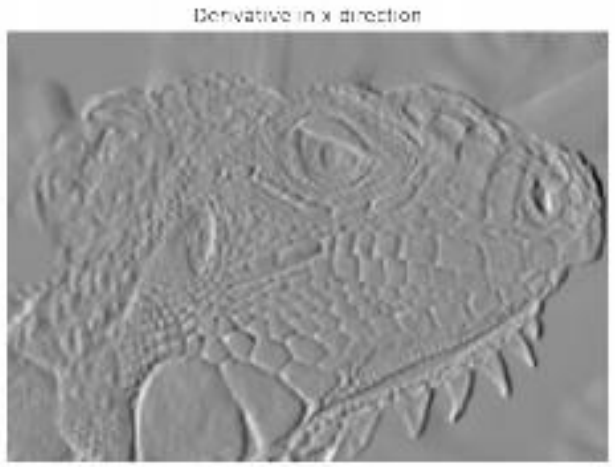
- Bước 1: Tính $G_x = I * H_x$ và $G_y = I * H_y$
- Bước 2: Tính $G_x + G_y$

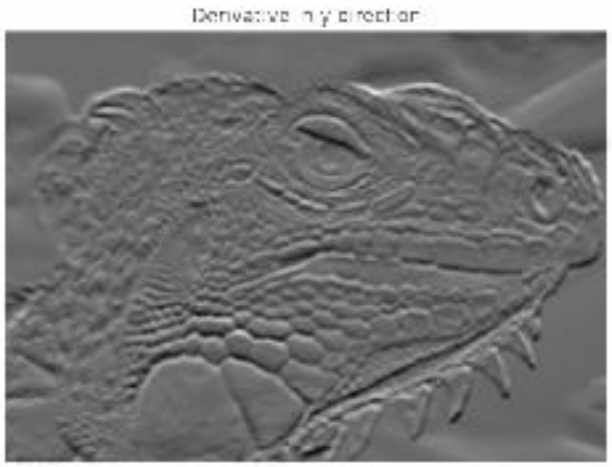
Prewitt Filter

```
# Kernel Prewitt X và Y
prewitt_x = np.array([[-1, 0, 1],
                      [-1, 0, 1],
                      [-1, 0, 1]])

prewitt_y = np.array([[-1, -1, -1],
                      [ 0,  0,  0],
                      [ 1,  1,  1]])

# Áp dụng filter2D
gx = cv2.filter2D(img, -1, prewitt_x)
gy = cv2.filter2D(img, -1, prewitt_y)

# Kết hợp hai hướng
g = cv2.addWeighted(gx, 0.5, gy, 0.5, 0)
```

3x3 image gradient filters

$$\frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$





Chia cho 3 để **chuẩn hóa** giá trị, tránh ảnh quá sáng.

```
# Kernel Prewitt
prewitt_x = np.array([[-1, 0, 1],
                      [-1, 0, 1],
                      [-1, 0, 1]]) / 3

prewitt_y = np.array([[ 1,  1,  1],
                      [ 0,  0,  0],
                      [-1, -1, -1]]) / 3

# Lọc ảnh
gx = cv2.filter2D(img, -1, prewitt_x)
gy = cv2.filter2D(img, -1, prewitt_y)
```

```
g = np.sqrt(gx**2 + gy**2)
g = np.uint8(g / g.max() * 255)   # Chuẩn hóa về [0,255]
```

Bài tập

Image I (9 x 8)

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	100	100	100	100	100	100
0	0	0	100	100	100	100	100	100
0	0	0	100	100	100	100	100	100
0	0	0	100	100	100	100	100	100
0	0	0	100	100	100	100	100	100
0	0	0	100	100	100	100	100	100

M_x: Derivative filter wrt x

1	0	-1

1
0
-1

M_y: Derivative filter wrt y

Bài tập

$$I = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 5 & 5 & 5 & 0 & 0 \\ 5 & 5 & 5 & 5 & 0 & 0 \\ 5 & 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Phát hiện cạnh đơn giản sử dụng đạo hàm bậc 1

- Nhân chập ảnh với 2 mặt nạ để xấp xỉ đạo hàm bậc 1 theo x và y

- Tính độ lớn của gradient

- Lấy ngưỡng: cạnh là điểm có độ lớn gradient > T

Phát hiện cạnh đơn giản sử dụng đạo hàm bậc 1

Original image

Gradient magnitud using Sobel operator

Threshold T = 25

Threshold T = 60

Không làm trơn trước khi tính đạo hàm

a b
c d

FIGURE 10.16
(a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) $|g_x|$, the component of the gradient in the x-direction, obtained using the Sobel mask in Fig. 10.14(f) to filter the image. (c) $|g_y|$, obtained using the mask in Fig. 10.14(g). (d) The gradient image, $|g_x| + |g_y|$.

Làm trơn trước khi tính đạo hàm

a b
c d

FIGURE 10.18
Same sequence as in Fig. 10.16, but with the original image smoothed using a 5×5 averaging filter prior to edge detection.

Không làm trơn
trước tính đạo hàm

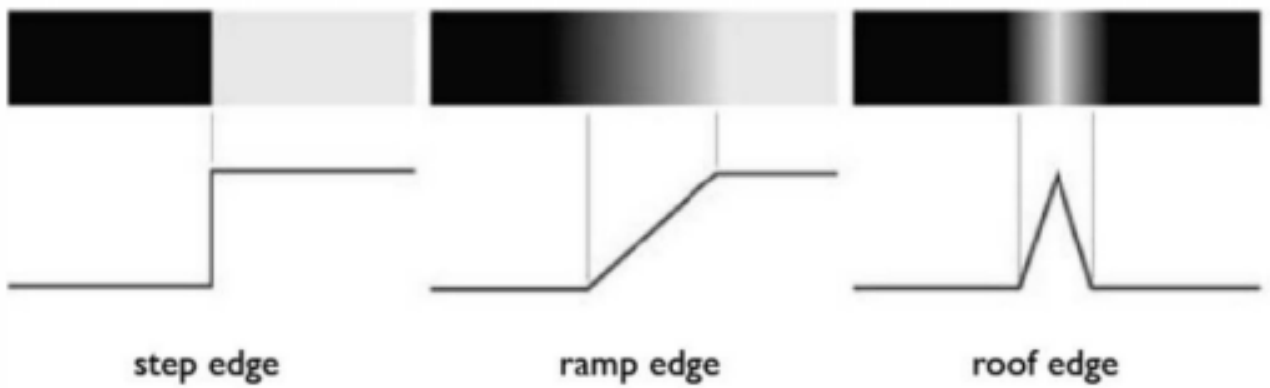
Làm trơn
trước tính đạo hàm

a b

FIGURE 10.20 (a) Thresholded version of the image in Fig. 10.16(d), with the threshold selected as 33% of the highest value in the image; this threshold was just high enough to eliminate most of the brick edges in the gradient image. (b) Thresholded version of the image in Fig. 10.18(d), obtained using a threshold equal to 33% of the highest value in that image.

Vấn đề

- Vị trí không chính xác (biên dày)
- Giá trị ngưỡng ưu ái cạnh theo 1 vài hướng hơn là các hướng khác
  - Có thể thiếu các đường biên chéo hơn là biên ngang hoặc dọc ➔ bỏ sót biên



Một số mặt nạ khác

- Để tránh ưu ái cạnh theo 1 vài hướng ➔ sử dụng kỹ thuật la bàn:
  - Nhân chập ảnh với 8 mặt nạ theo 8 hướng (0, 45, 90, ..)
  - Cộng kết quả nhận chập lại

$$H_1 = \begin{pmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{pmatrix} \qquad H_2 = \begin{pmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{pmatrix}$$

$$H_3 = \begin{pmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{pmatrix} \qquad H_4 = \begin{pmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{pmatrix}$$

$$H_5 = \begin{pmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{pmatrix} \qquad H_6 = \begin{pmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{pmatrix}$$

$$H_7 = \begin{pmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{pmatrix} \qquad H_8 = \begin{pmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{pmatrix}$$

Một số mặt nạ khác

- Mặt nạ Prewitt, Sobel cho phát hiện biên chéo

0	1	1	−1	−1	0
−1	0	1	−1	0	1
−1	−1	0	0	1	1

Prewitt

0	1	2	−2	−1	0
−1	0	1	−1	0	1
−2	−1	0	0	1	2

Sobel

a b
c d

FIGURE 10.15
Prewitt and Sobel masks for detecting diagonal edges.

Một số mặt nạ khác

- Mặt nạ Sobel cho phát hiện biên chéo

a b

FIGURE 10.19 Diagonal edge detection. (a) Result of using the mask in Fig. 10.15(c). (b) Result of using the mask in Fig. 10.15(d). The input image in both cases was Fig. 10.18(a).

PHÁT HIỆN BIÊN VỚI ĐẠO HÀM BẬC 2

Next week