TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

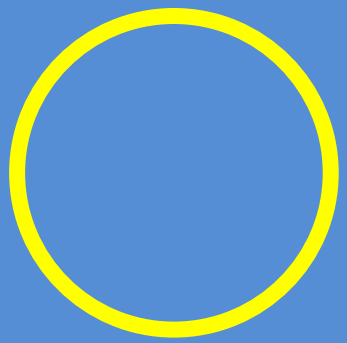
# CS231. Nhập môn Thị giác máy tính

## Đặc trưng Edge

Mai Tiến Dũng

# What is Edge ?

# What is Edge ?



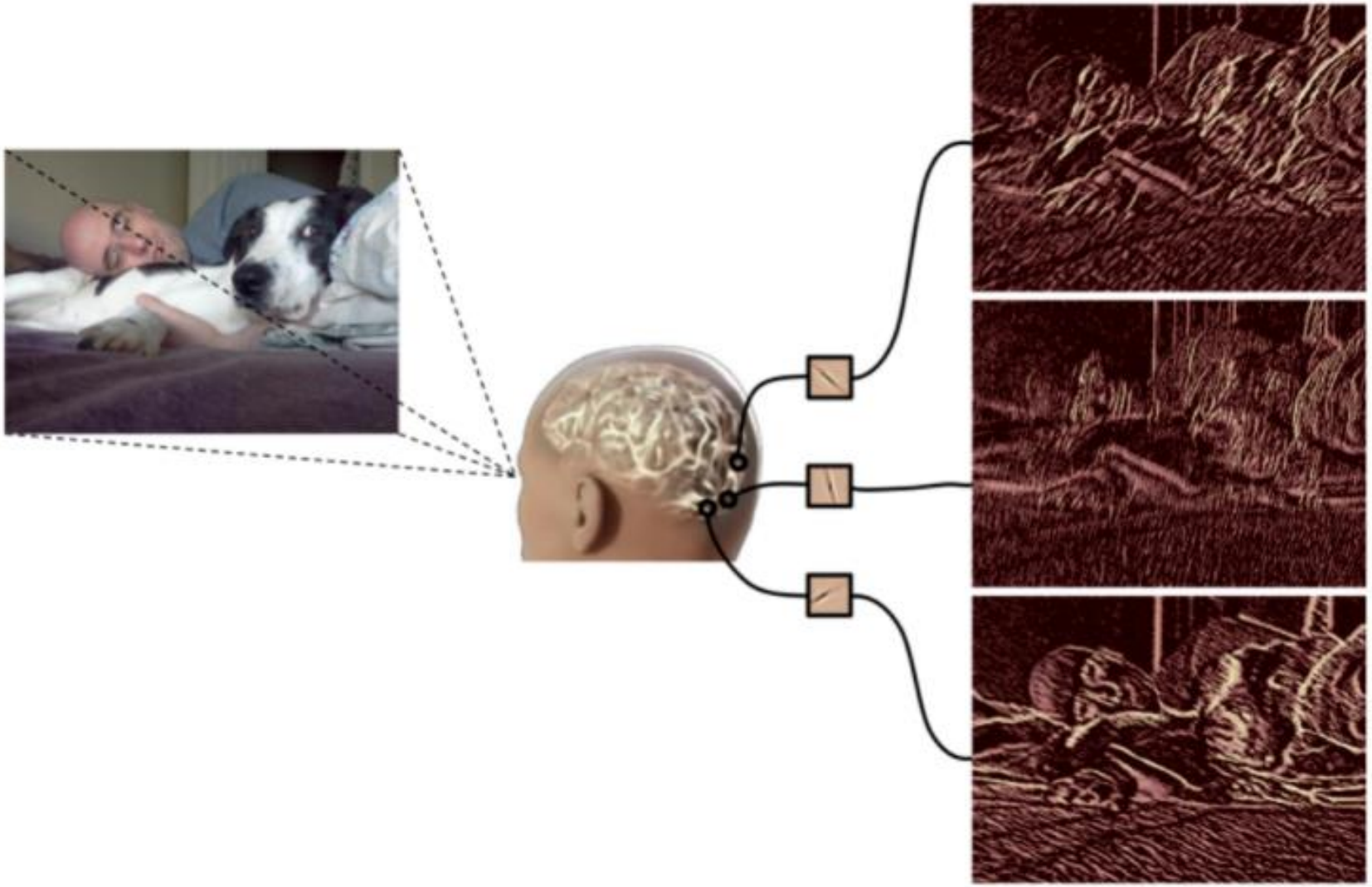


**Figure 2:** *Visual information is processed in an area of the brain where each neuron detects in the observed scene edges of a specific orientation and width. It is thought that what we (and other mammals) "see" is a processed interpolation of these edge detected images.*

https://kenndanielso.github.io/mlrefined/blog_posts/14_Convolutional_networks/14_2_Edge_histogram_based_features.html

# What is Edge ?

- Abrupt change in the intensity of pixels
- Discontinuity in image brightness or contrast
- Usually edges occur on the boundary of two regions.

# Goal of edge detection

- Edge detection is extensively used in image segmentation when we want to divide the image into areas corresponding to different objects.
- If we want to extract different objects from an image, we need Edge detection.
- Using edge detection, we can: recognition, image comparison, unaccepted object can be removed.

# Goal of edge detection

https://aaronhertzmann.com/2020/04/19/lines-as-edges.html

# Goal of edge detection

## Edge Detection Based Shape Identification

Vivek Kumar#[1], Sumit Pandey#, Amrindra Pal#, Sandeep Sharma#
*# Department of Electronics and Communication Engineering*
*Dehradun Institute of Technology*
*Dehradun: 248009, India*
*[1]Student Author - er_vivek@outlook..com*

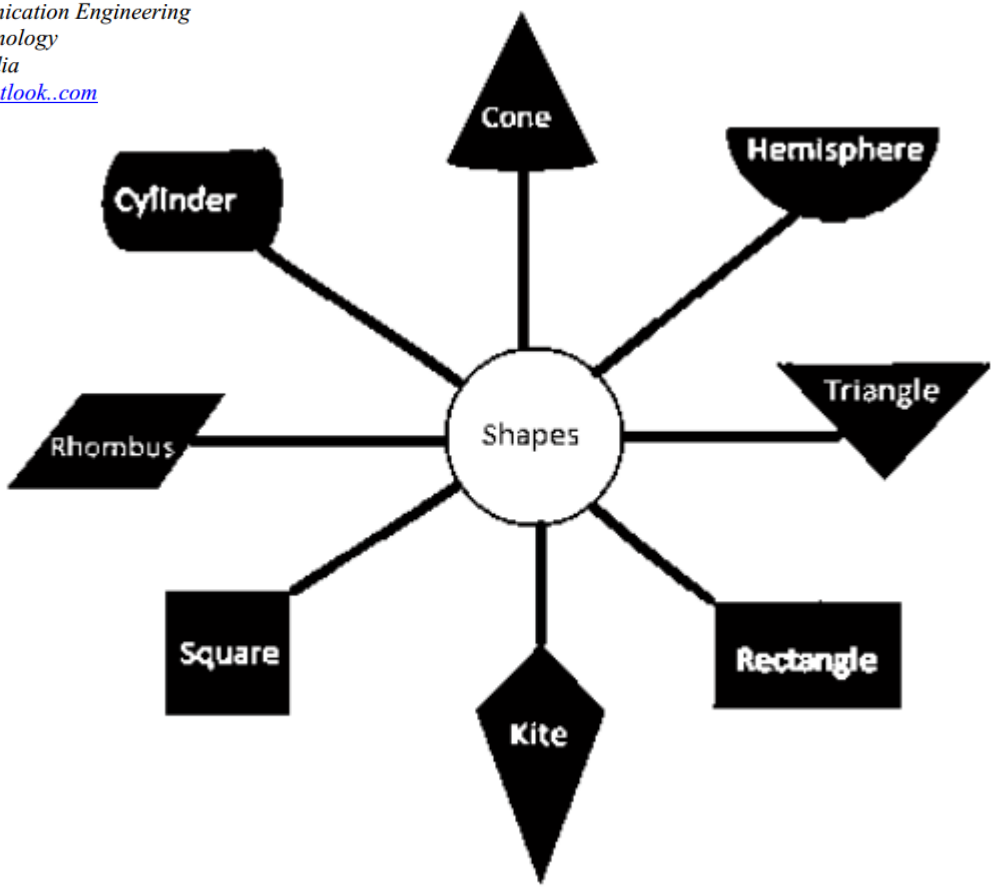
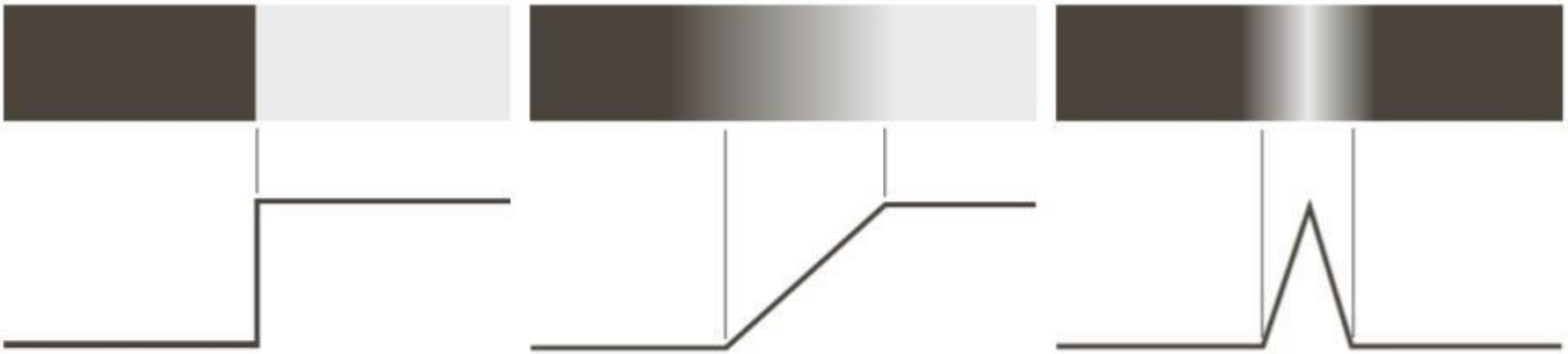



Fig. 2: Detectable 2-D Shapes

# Edge Models ?



Step edge

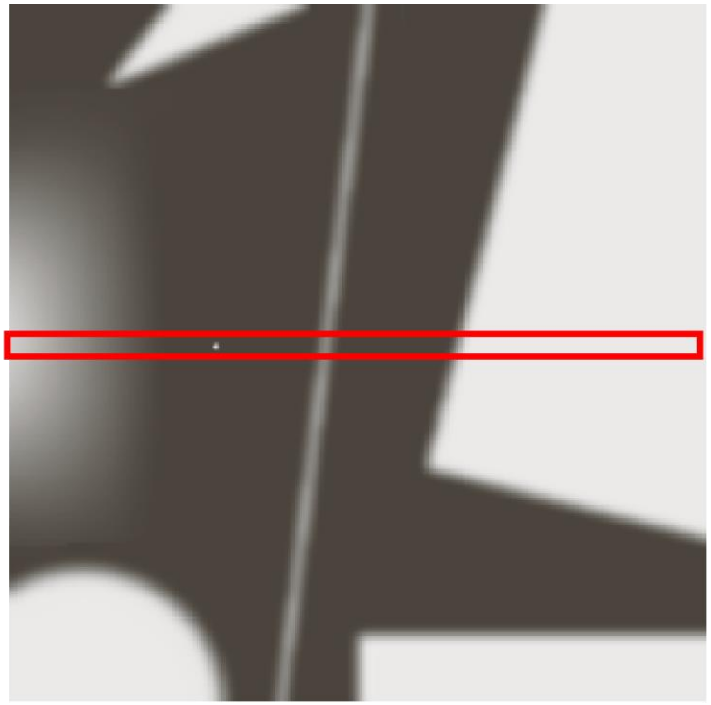
Ramp edge

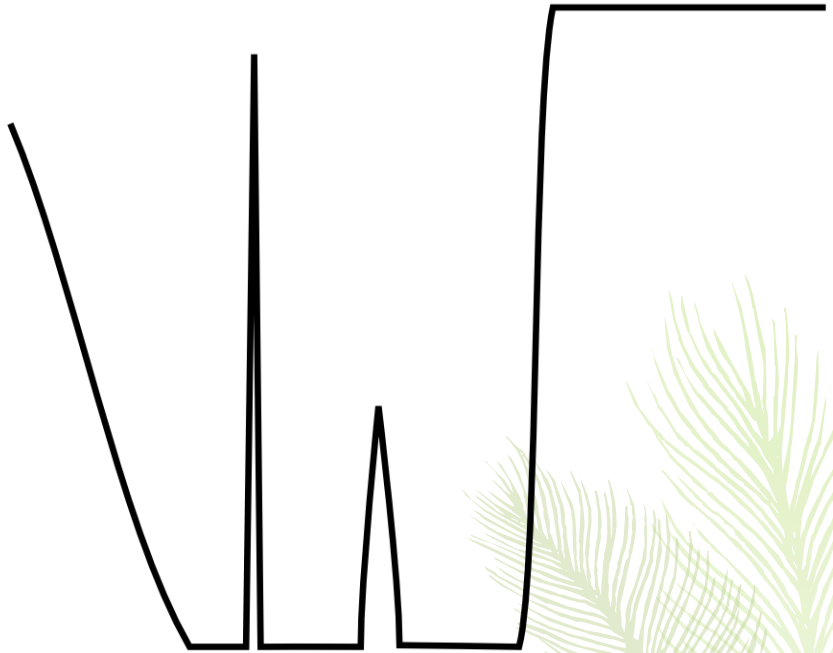
Roof edge

# Edge Models ?


Fig 7.11: The most common edge models (a) step or ideal edge, (b) ramp, (c) roof edge, and (d) the real edge

# Edge Models ?





https://miac.unibas.ch/SIP/07-Segmentation-media/figs/1st_vs_2nd_derivative.png

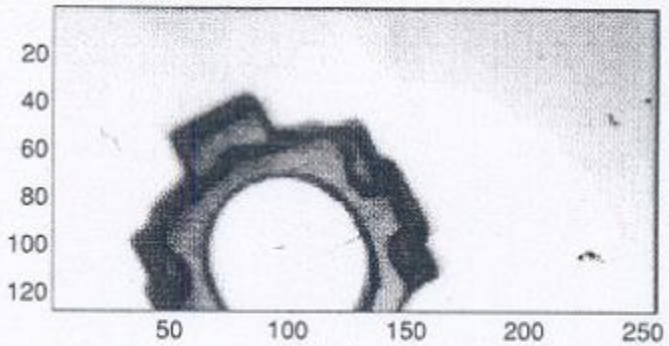
# Edge Detection

- Cạnh (đường viền) ảnh là một đặc trưng rất quan trọng do hệ thống thị giác của con người dựa trên cạnh để nhận thức được đối tượng.
- Việc dò tìm cạnh ảnh có thể dựa vào độ thay đổi của các giá trị pixel, hay còn gọi là gradient của ảnh.

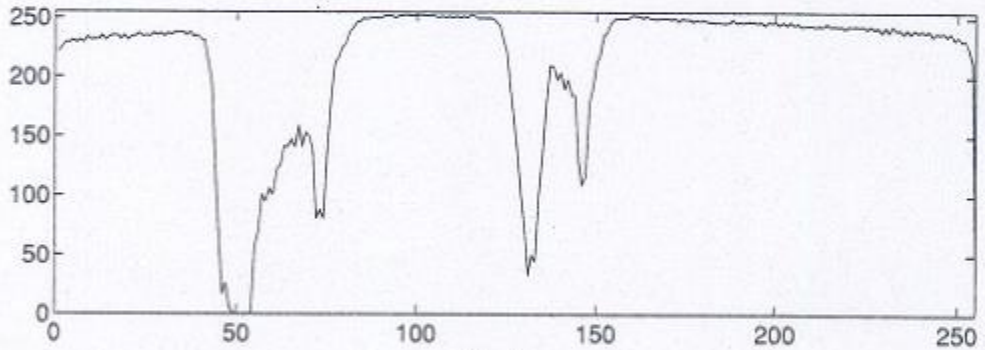
# Edge Detection

- Phương pháp dò cạnh có thể được chia làm 2 loại:
  - phụ thuộc hướng và
  - độc lập hướng.
- Phương pháp dò cạnh phụ thuộc hướng: sẽ dựa trên việc xác định gradient của ảnh theo các hướng x và y.

# Edge detection




(a)




(b)

Figure 5.2: (a) The top half of the connecting rod image. (b) The profile of an ideal step change in image intensity is plotted to show that the edges are not perfectly sharp and the image is corrupted by noise. The plot is a horizontal slice through the circular portion of the connecting rod corresponding to the bottom edge of the partial rod image shown above.

# Edge detection: Definitions

**Definition 5.1** *An edge point is a point in an image with coordinates* $[i, j]$ *at the location of a significant local intensity change in the image.*

**Definition 5.2** *An edge fragment corresponds to the* $i$ *and* $j$ *coordinates of an edge and the edge orientation* $\theta$*, which may be the gradient angle.*

**Definition 5.3** *An edge detector is an algorithm that produces a set of edges (edge points or edge fragments) from an image.*

**Definition 5.4** *A contour is a list of edges or the mathematical curve that models the list of edges.*

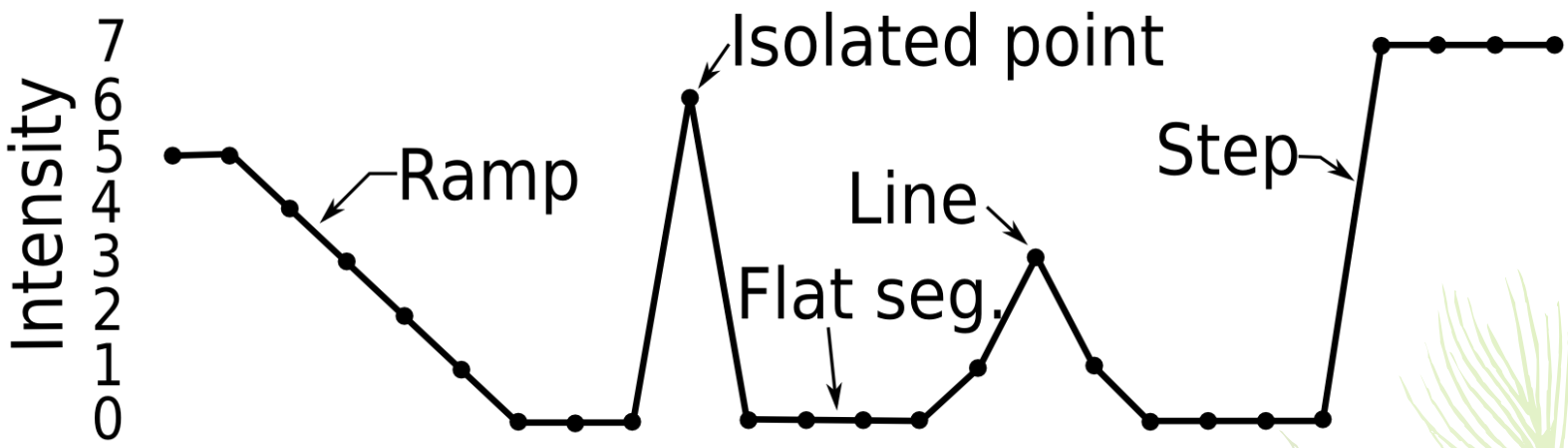
**Definition 5.5** *Edge linking is the process of forming an ordered list of edges from an unordered list. By convention, edges are ordered by traversal in a clockwise direction.*

**Definition 5.6** *Edge following is the process of searching the (filtered) image to determine contours.*

# Methods of edge detection

- First Order Derivative / Gradient Methods
  - Roberts Operator
  - Prewitt Operator
  - Sobel Operator
- Second Order Derivative
  - Laplacian
  - Laplacian of Gaussian
- Optimal Edge Detection
  - Canny edge detection

# Edge Models ?




| | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Image | 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 |
| 1st derivative | | -1 | -1 | -1 | -1 | -1 | 0 | 0 | 6 | -6 | 0 | 0 | 0 | 1 | 2 | -2 | -1 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | |
| 2nd derivative | | -1 | 0 | 0 | 0 | 0 | 1 | 0 | 6 | -12 | 6 | 0 | 0 | 1 | 1 | -4 | 1 | 1 | 0 | 0 | 7 | -7 | 0 | 0 | |

# Gradient Methods : Gradient

Gradient của hàm $f(\mathbf{v})$ với $\mathbf{v} = (v_1, v_2, \ldots, v_n)$ là một vector:

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial v_1} \\ \frac{\partial f}{\partial v_2} \\ \cdots \\ \frac{\partial f}{\partial v_n} \end{bmatrix}$$

Mỗi thành phần trong vector đó là một đạo hàm riêng phần (*partial derivative*) theo từng biến của hàm đó

# Gradient Methods : Gradient

The gradient is the two-dimensional equivalent of the first derivative and is defined as the *vector*

$$\mathbf{G}[f(x,y)] = \begin{bmatrix} G_x \\ \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \\ \frac{\partial f}{\partial y} \end{bmatrix}. \qquad (5.1)$$

$$\nabla f(x,y) = \begin{bmatrix} \dfrac{\partial f(x,y)}{\partial x} \\ \dfrac{\partial f(x,y)}{\partial y} \end{bmatrix}$$

# Gradient Methods : Gradient

There are two important properties associated with the gradient: (1) the vector $\mathbf{G}[f(x,y)]$ points in the direction of the maximum rate of increase of the function $f(x,y)$, and (2) the magnitude of the gradient, given by

$$G[f(x,y)] = \sqrt{G_x^2 + G_y^2}, \tag{5.2}$$

equals the maximum rate of increase of $f(x,y)$ per unit distance in the direction $G$. It is common practice, however, to approximate the gradient magnitude by absolute values:

$$G[f(x,y)] \approx |G_x| + |G_y| \tag{5.3}$$

or

$$G[f(x,y)] \approx \max(|G_x|, |G_y|). \tag{5.4}$$

# Gradient Methods : Gradient

From vector analysis, the *direction* of the gradient is defined as:

$$\alpha(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{5.5}$$

where the angle $\alpha$ is measured with respect to the $x$ axis.

Note that the magnitude of the gradient is actually independent of the direction of the edge. Such operators are called *isotropic operators*.

# Gradient Methods : Gradient

## Numerical Approximation

For digital images, the derivatives in Equation 5.1 are approximated by differences. The simplest gradient approximation is

$$G_x \cong f[i, j+1] - f[i, j] \tag{5.6}$$

$$G_y \cong f[i, j] - f[i+1, j]. \tag{5.7}$$

Remember that $j$ corresponds to the $x$ direction and $i$ to the negative $y$ direction. These can be implemented with simple convolution masks as shown below:

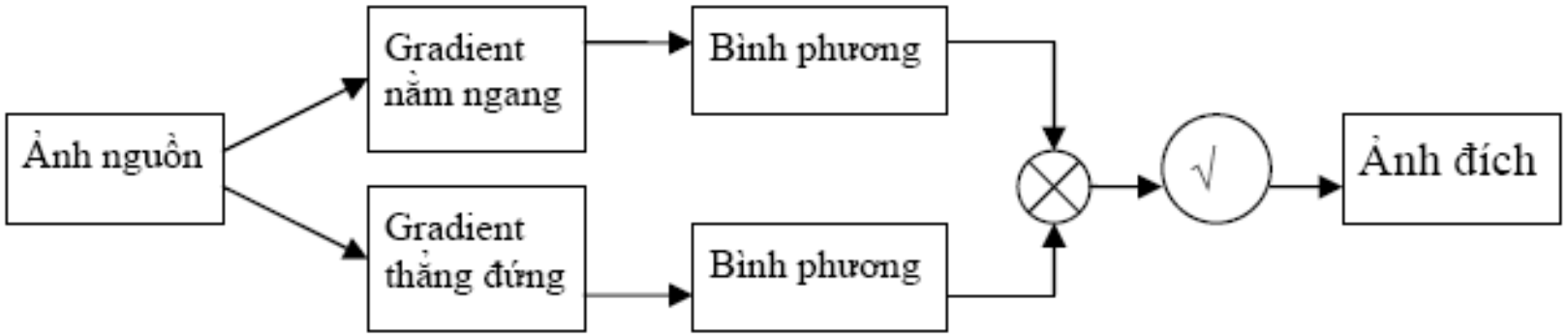
$$G_x = \boxed{-1 \mid 1} \qquad G_y = \begin{array}{|c|}\hline 1 \\ \hline -1 \\ \hline \end{array} \tag{5.8}$$

# Ví dụ

| | | | | | |
|---|---|---|---|---|---|
| **0** | **0** | **0** | **255** | **255** | **255** |
| 0 | 0 | 0 | 255 | 255 | 255 |
| 0 | 0 | 0 | 255 | 255 | 255 |
| 0 | 0 | 0 | 255 | 255 | 255 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |

# Gradient Methods

- **Sơ đồ thực hiện**



# Gradient Methods: Roberts

- Xét vùng ảnh như sau:

| | | |
|---|---|---|
| $z_1$ | $z_2$ | $z_3$ |
| $z_4$ | $z_5$ | $z_6$ |
| $z_7$ | $z_8$ | $z_9$ |

- Khi đó, ta có thể xấp xỉ:

$$|\nabla f(x,y)| \approx \left[(z_5 - z_8)^2 + (z_5 - z_6)^2\right]^{1/2}$$

- Khi cài đặt, ta có thể dùng các nhân tương ứng:

$$h_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \text{ and } h_2 = \begin{bmatrix} 1 & -1 \end{bmatrix}$$

- Hay ta có thể viết:

$$|\nabla f(x,y)| \approx \left[(f * h_1)^2 + (f * h_2)^2\right]^{1/2}$$

# Gradient Methods: Roberts

- Ngoài ra, ta có thể xấp xỉ:

$$|\nabla f(x,y)| \approx \left[(z_5 - z_9)^2 + (z_6 - z_8)^2\right]^{1/2}$$

- Khi cài đặt, ta có thể dùng các nhân tương ứng:

$$h_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \text{ and } h_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

- Hay ta có thể viết:

$$|\nabla f(x,y)| \approx \left[(f * h_1)^2 + (f * h_2)^2\right]^{1/2}$$

# Gradient Methods: Roberts

- Các nhân tích chập trên còn được gọi là toán tử **Roberts cross-gradient operators**

$$h_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \text{ and } h_2 = \begin{bmatrix} 1 & -1 \end{bmatrix}$$

$$h_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \text{ and } h_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

# Gradient Methods: Prewitt

## Prewitt operation

- Một xấp xỉ tốt hơn giá trị của gradient có thể được tính như sau:

$$|\nabla f(x,y)| \approx \left[((z_7+z_8+z_9)-(z_1+z_2+z_3))^2+((z_3+z_6+z_9)-(z_1+z_4+z_7))^2\right]^{1/2}$$

- Tương ứng với nhân tích chập sau:

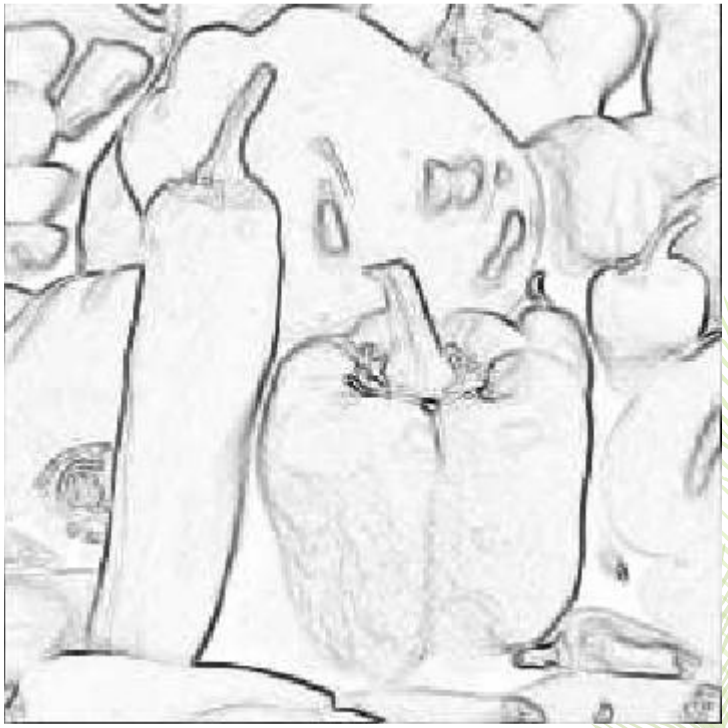
$$h_1=\begin{bmatrix}-1 & -1 & -1\\ 0 & 0 & 0\\ 1 & 1 & 1\end{bmatrix} \text{ and } h_2=\begin{bmatrix}-1 & 0 & 1\\ -1 & 0 & 1\\ -1 & 0 & 1\end{bmatrix}$$

| | | | | | |
|---|---|---|---|---|---|
| **0** | **0** | **0** | **255** | **255** | **255** |
| 0 | 0 | 0 | 255 | 255 | 255 |
| 0 | 0 | 0 | 255 | 255 | 255 |
| 0 | 0 | 0 | 255 | 255 | 255 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |

# Gradient Methods: Prewitt

## Prewitt operation

Ảnh gốc



Prewitt

# Gradient Methods: Sobel

## Sobel Operator:

- Sử dụng 2 nhân tích chập sau:

H=

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

Gy

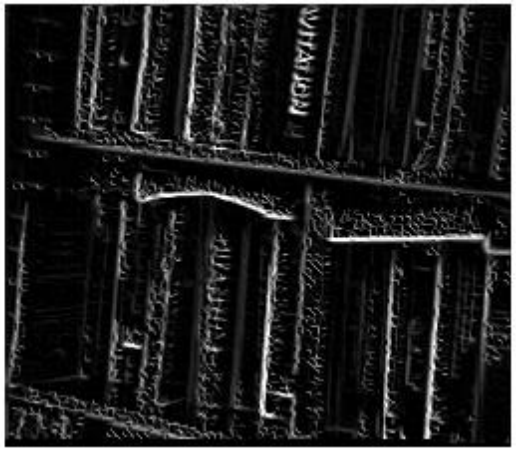
V=

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

Gx

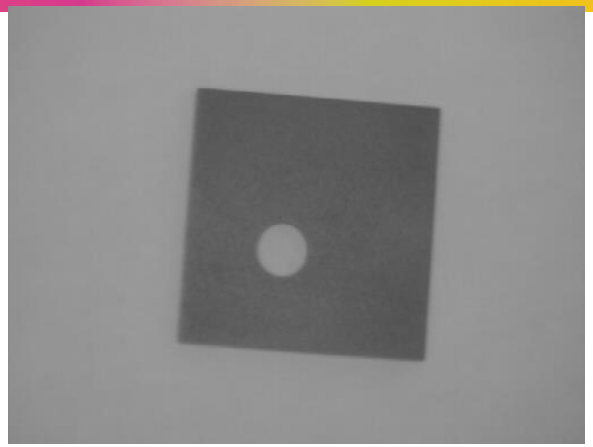
# Gradient Methods: Sobel

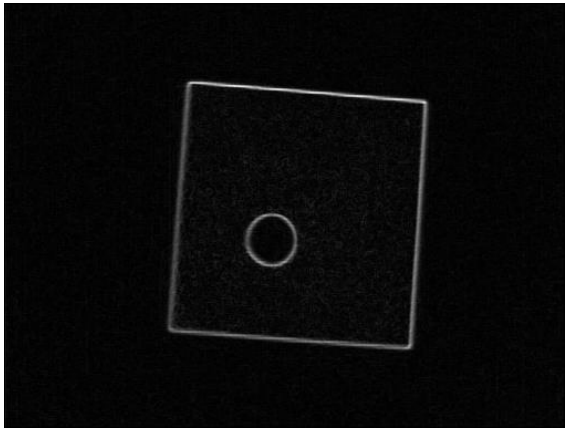
- Ví dụ:



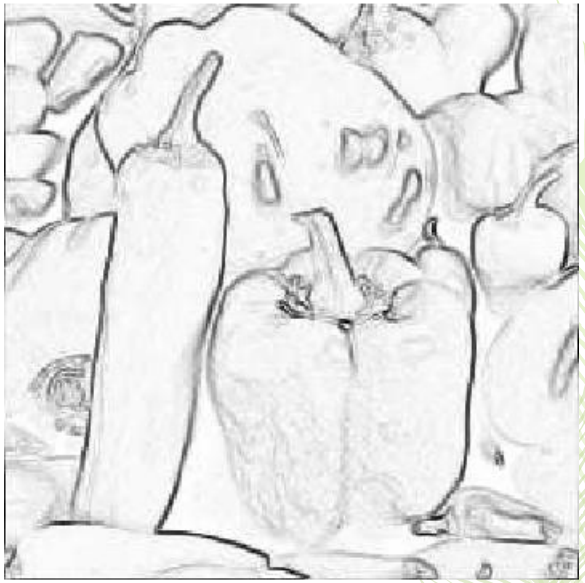
- → Hạn chế của phương pháp này: nhạy cảm đối với nhiễu (nếu nhiễu lớn thì nó sẽ lấy nhiễu làm cạnh).

# Gradient Methods: Sobel



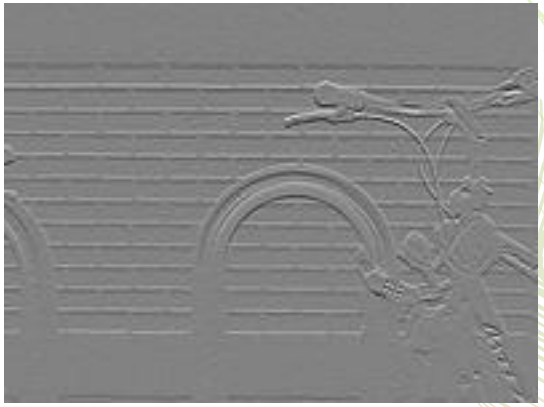


Ảnh gốc

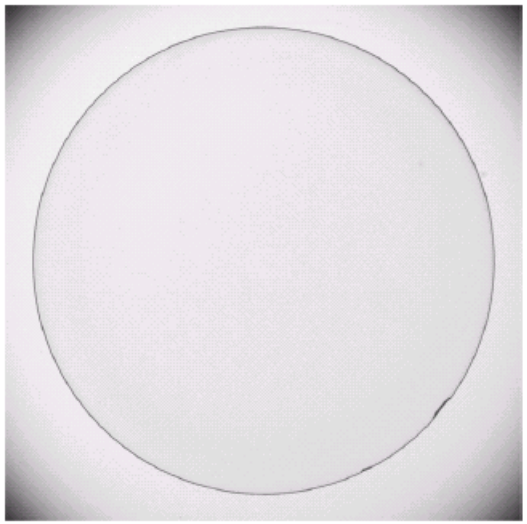


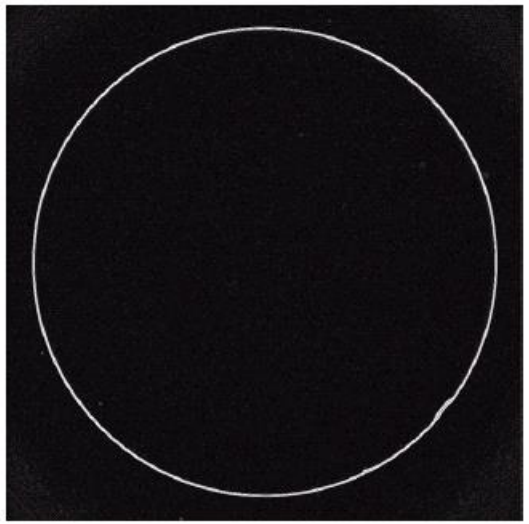
Sobel

# Gradient Methods: Sobel

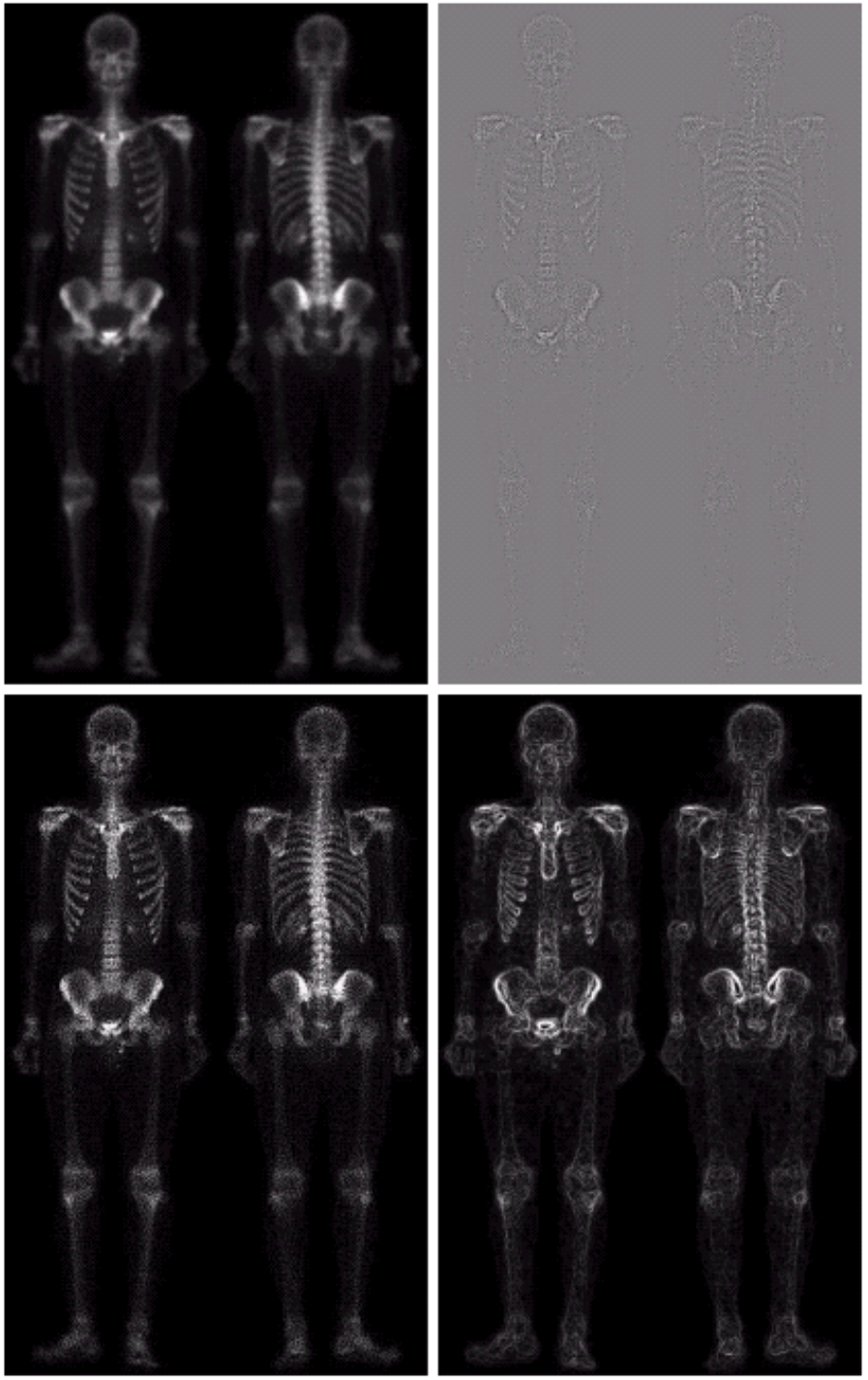


# Gradient Methods: Sobel





An image of a contact lens which is enhanced in order to make defects (at four and five o'clock in the image) more obvious



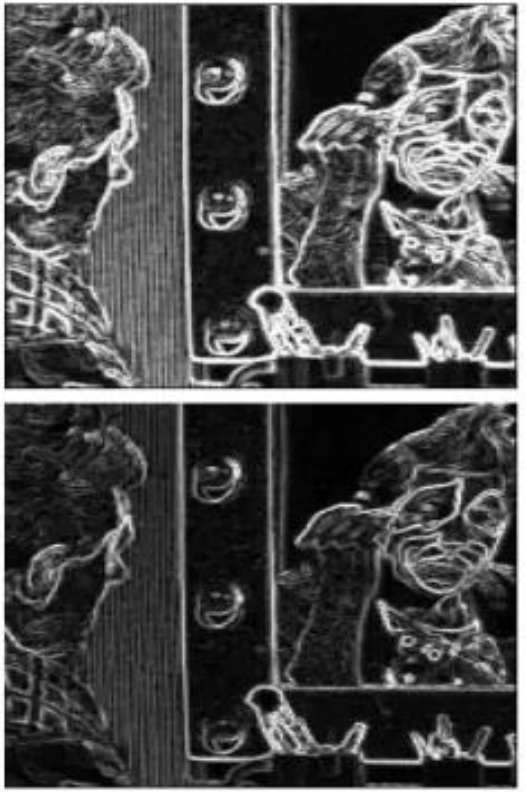
a b
c d

**FIGURE 3.46**
(a) Image of whole body bone scan.
(b) Laplacian of (a). (c) Sharpened image obtained by adding (a) and (b). (d) Sobel of (a).

# Edge Detection

Compare the output of the Sobel Operator with Roberts

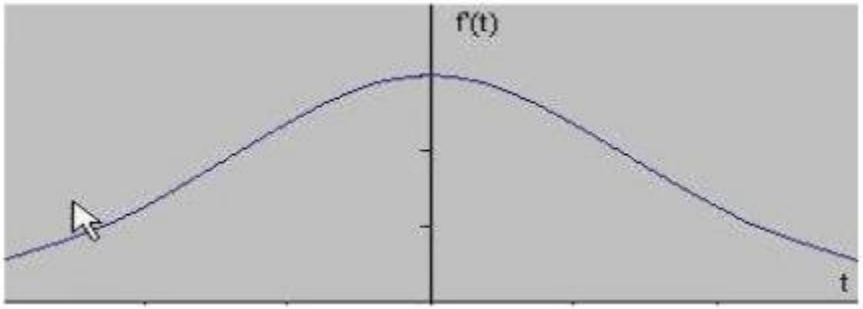
- The spurious edges are still present but they are relatively less intense
- Roberts operator has missed a few edges
- Sobel operator detects thicker edges

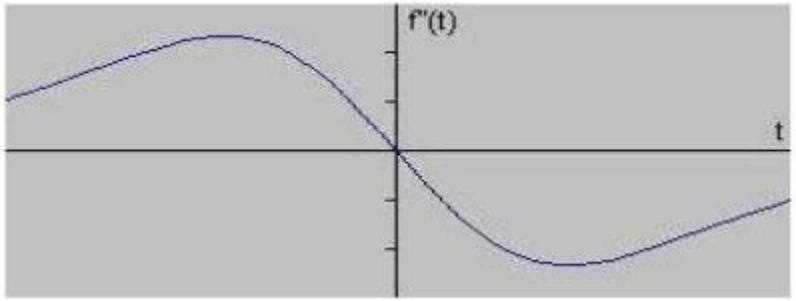


**Outputs of Sobel (top) and Roberts operator**

# Second Order Derivative Methods

Zero crossing of the second derivative of a function indicates the presence of a maxima



# Second Order Derivative Methods: Laplacian

- Defined as $$L(x,y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

- Mask

| | | |
|---|---|---|
| 0 | 1 | 0 |
| 1 | -4 | 1 |
| 0 | 1 | 0 |

- Very susceptible to noise, filtering required, use Laplacian of Gaussian

# Second Order Derivative Methods: Laplacian of Gaussian

- Smooth the image using Gaussian filter
- Enhance the edges using Laplacian operator
- Zero crossings denote the edge location
- Use linear interpolation to determine the sub-pixel location of the edge

# Canny Edge Detection

- It is a multi-stage algorithm used to detect/identify a wide range of edges.
- The following are the various stages of the Canny edge detection algorithm:

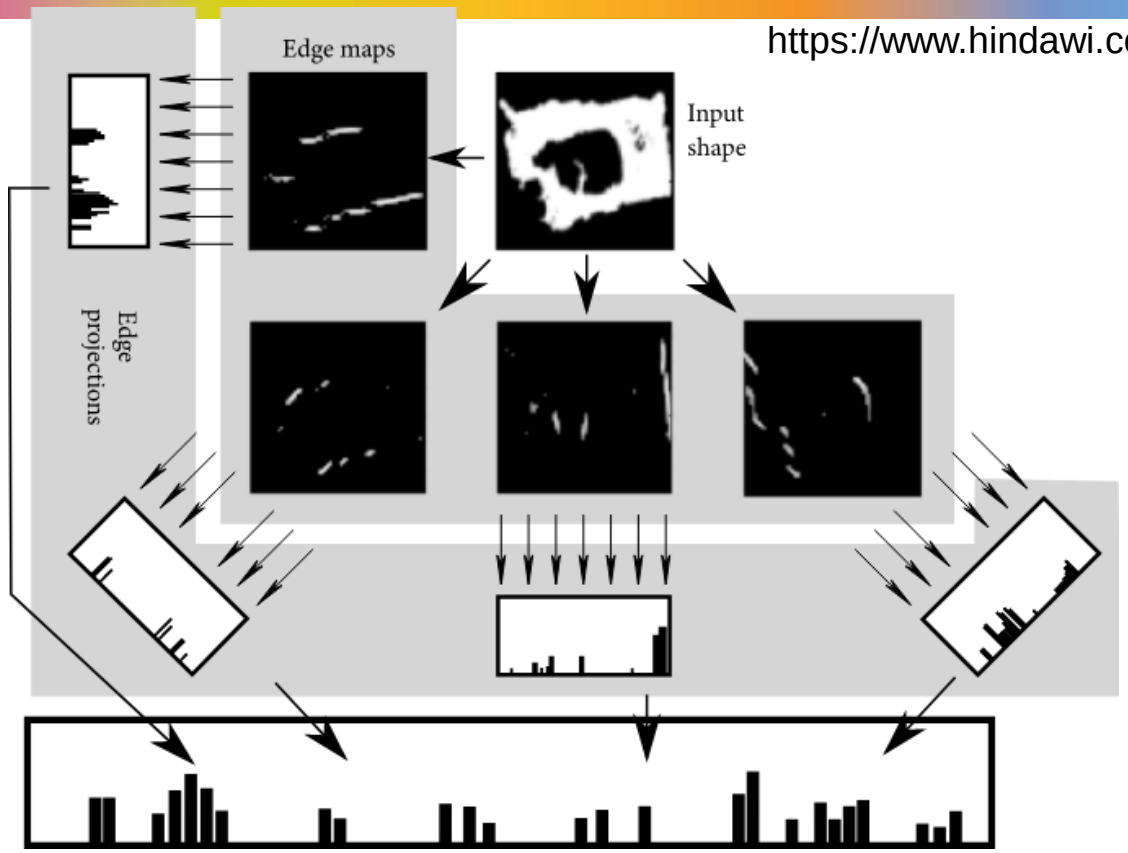
# Canny Edge Detection

- Convert the image to grayscale
- Reduce noise – as the edge detection that using derivatives is sensitive to noise, we reduce it.
- Calculate the gradient – helps identify the edge intensity and direction.
- Non-maximum suppression – to thin the edges of the image.
- Double threshold – to identify the strong, weak and irrelevant pixels in the images.
- Hysteresis edge tracking – helps convert the weak pixels into strong ones only if they have a strong pixel around them.

```
from scipy import ndimage

def sobel_filters(img):
    Kx = np.array([[-1, 0, 1], [-2, 0, 2], [-1, 0, 1]], np.float32)
    Ky = np.array([[1, 2, 1], [0, 0, 0], [-1, -2, -1]], np.float32)

    Ix = ndimage.filters.convolve(img, Kx)
    Iy = ndimage.filters.convolve(img, Ky)

    G = np.hypot(Ix, Iy)
    G = G / G.max() * 255
    theta = np.arctan2(Iy, Ix)

    return (G, theta)
```

```
kernels = np.array([ [-1,0,1],
                     [-2,0,2],
                     [-1,0,1]
                     ]

img_rst = cv2.filter2D(img_src,-1,kernels)
```

https://www.hindawi.com/journals/mpe/2018/4763050/




Construction of the EPPSED feature vector. Edges are detected in four directions; then thresholding and maxima selection are applied; finally projections are concatenated and normalized.

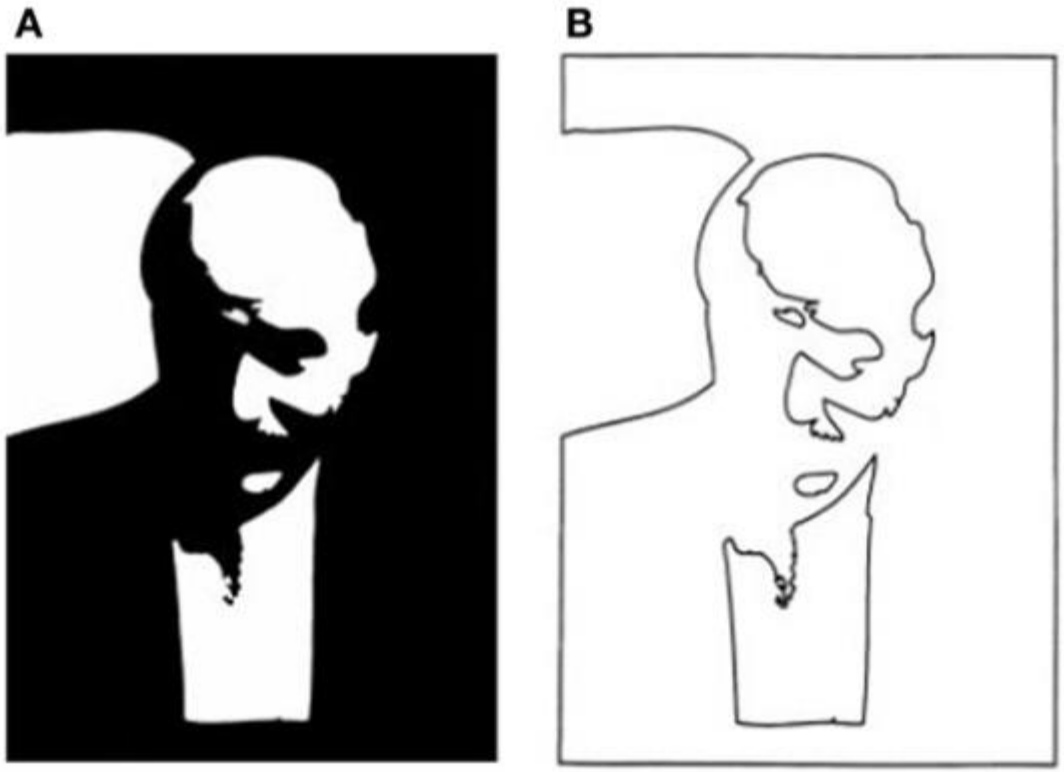



**FIGURE 4 | (A)** Two-tone image of Kennedy (1997) where the dark areas could be dark pigment or dark shadow. The face can be recovered only if the shadows are correctly identified. **(B)** A line version of the same contours is unrecognizable. The polarity information required for finding shadows is lost. The shadow boundaries can no longer be discounted and are taken as object contours that form meaningless shapes unrelated to the original.

https://aaronhertzmann.com/2020/04/19/lines-as-edges.html

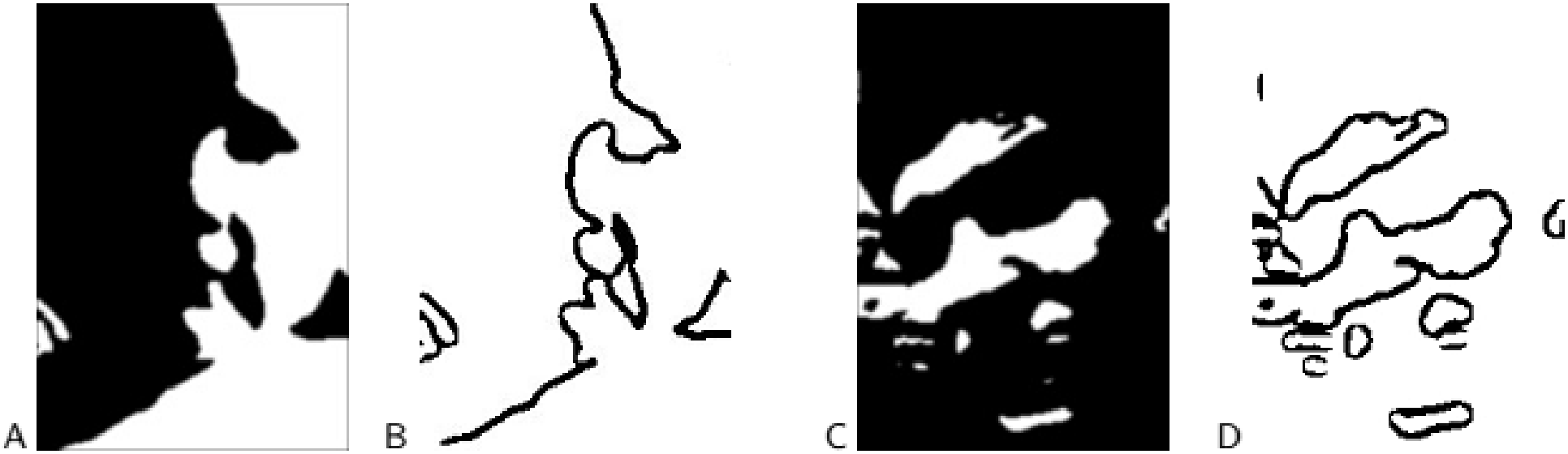



Figure 2: Lines-As-Edges treats lines and edges as perceptually-equivalent. Hence, according to Lines-As-Edges, the image in Panel A should give the same shape percept as Panel B, as should C and D. (Images A and C from Schwiedrzik et al. (2018), CC-BY.)

# Yêu cầu 1

- Đọc ảnh và hiển thị ảnh kết quả sau khi dò tìm cạnh: Gx, Gy và G
  - 2 ảnh: Bicycle.jpg và ( bansoxe.jpg hoặc houseg.bmp)

# Yêu cầu 2

- Viết hàm xác định đặc trưng ảnh dựa trên thông tin cạnh: theo trục x và trục y
- Tính khoảng cách euclidean giữa 2 ảnh: sử dụng thư viện của scipy
- Tính sự tương đồng theo cosin giữa 2 ảnh.

---

2 cặp ảnh:

< image_0814.jpg, image_0816.jpg>

< image_0814.jpg, image_1248.jpg>

# Yêu cầu 2: Gợi ý

```
def sobel_filters(img):
    Sx=np.array([[-1,0,1],[-2,0,2],[-1,0,1]],np.float32)
    Sy=np.array([[1,2,1],[0,0,0],[-1,-2,-1]],np.float32)

    Ix = cv.filter2D(img, -1, Sx)
    Iy = cv.filter2D(img, -1, Sy)

    G=np.hypot(Ix,Iy)
    G=G/G.max()*255
    theta=np.arctan2(Iy,Ix)

    return Ix,Iy,G,theta
```

# Yêu cầu 2: Gợi ý

```
def calFeartureVector(img_src):
  img_dst = img_src.copy()
  img_dst = cv.resize(img_dst, (256, 256))
  Ix, Iy, G, theta  = sobel_filters(img_dst)

  fearture = []
  for i in range(G.shape[0]):
    tmp = 0
    for j in range(G.shape[1]):
      tmp = tmp + G[i,j]
    fearture.append(tmp)

  for j in range(G.shape[1]):
    tmp = 0
    for i in range(G.shape[0]):
      tmp = tmp + G[i,j]
    fearture.append(tmp)
  return fearture
```

# Yêu cầu 2: Gợi ý

- Hoặc:

```
def calFeatureVector(img):
    img_ = cv.resize(img, (256, 256))
    _, _, G, _ = sobel_filters(img_)
    row_sum = np.sum(G, axis=1, dtype=np.float64)
    col_sum = np.sum(G, axis=0, dtype=np.float64)
    return (np.hstack((row_sum.T, col_sum)))
```

# Yêu cầu 2: Gợi ý

```
def show2image(img_src, img_rst, title1, title2):
  plt.figure(figsize=(20, 20))
  # show img src
  plt.subplot(1,2,1)
  plt.title(title1)
  img_src = cv2.cvtColor(img_src, cv2.COLOR_BGR2RGB)
  plt.imshow(img_src, interpolation='bicubic')

  # show img result
  plt.subplot(1,2,2)
  plt.title(title2)
  img_rst = cv2.cvtColor(img_rst, cv2.COLOR_BGR2RGB)
  plt.imshow(img_rst, interpolation='bicubic')
```

# Tài liệu tham khảo


- https://miac.unibas.ch/SIP/
- https://www.mygreatlearning.com/blog/introduction-to-edge-detection/
- https://www.slideshare.net/bhavanakhivsara/edge-detection-73118840