Edge Detection
(Phát hiện biên trong ảnh)

Nội dung

1. Giới thiêu
2. Các phương pháp phát hiện biên
   - 2.1. Phương pháp Gradient
   - 2.2. Canny Detector
   - 2.3. Laplace of Gaussian
3. Phát hiện đường thẳng
   - 3.1. Hough transform
   - 3.2. RANSAC

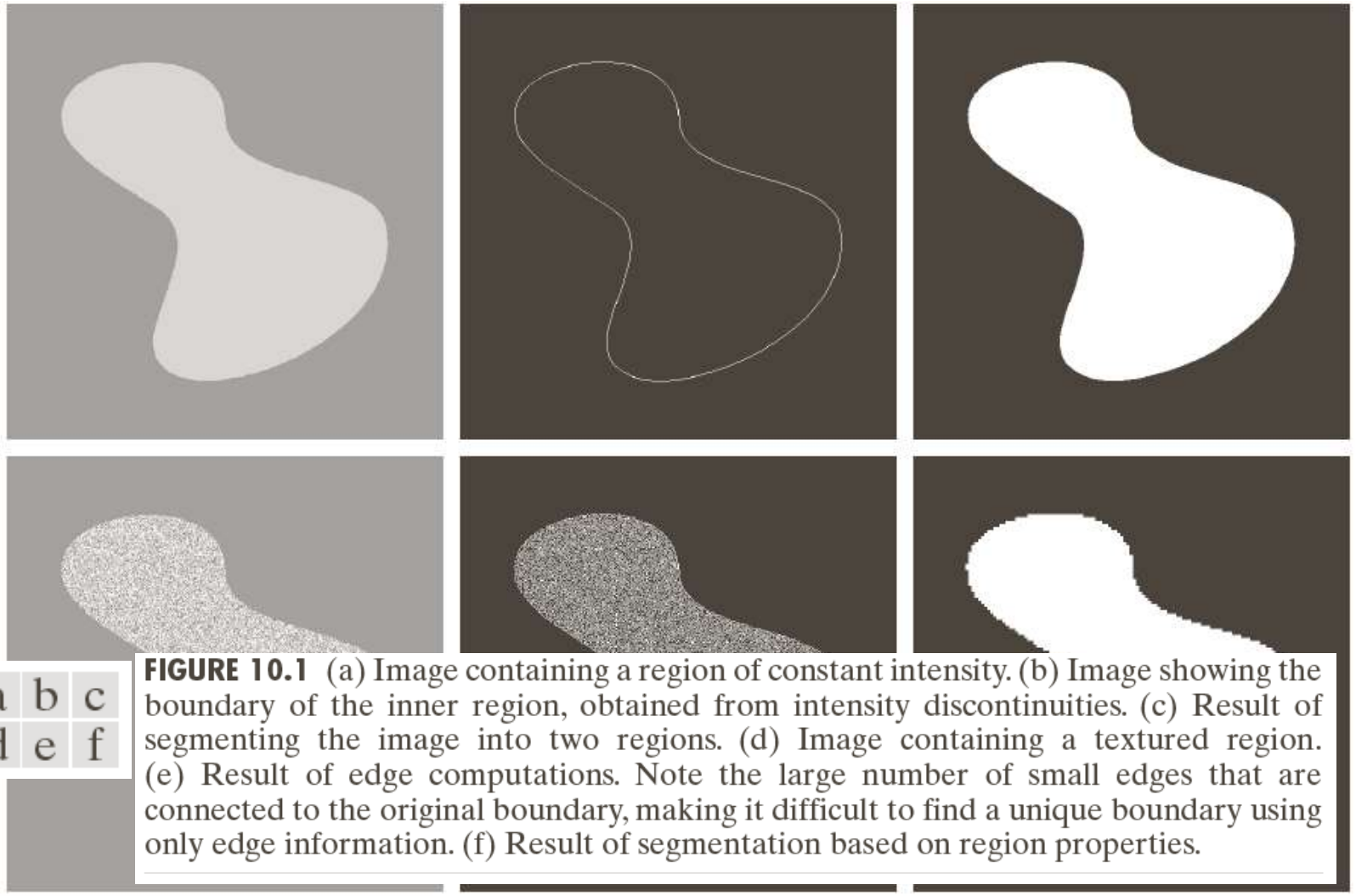
1. Giới thiệu

- Điểm biên: Một điểm ảnh được coi là điểm biên nếu có sự thay đổi nhanh hoặc đột ngột về mức xám (hoặc màu).
  - Ví dụ, trong ảnh nhị phân, điểm đen được gọi là điểm biên nếu lân cận của nó có ít nhất một điểm trắng.
- Đường biên còn gọi là đường bao (boundary): Là tập hợp các điểm biên liên tiếp.

1. Giới thiệu

- Ý nghĩa của đường biên
  - Đường biên là một loại đặc trưng cục bộ tiêu biểu trong phân tích, nhận dạng ảnh.
  - Người ta sử dụng biên làm phân cách các vùng xám (hoặc màu) cách biệt. Ngược lại, người ta cũng sử dụng các vùng ảnh để tìm phân cách.

Khó khăn của bài toán phát hiện biên

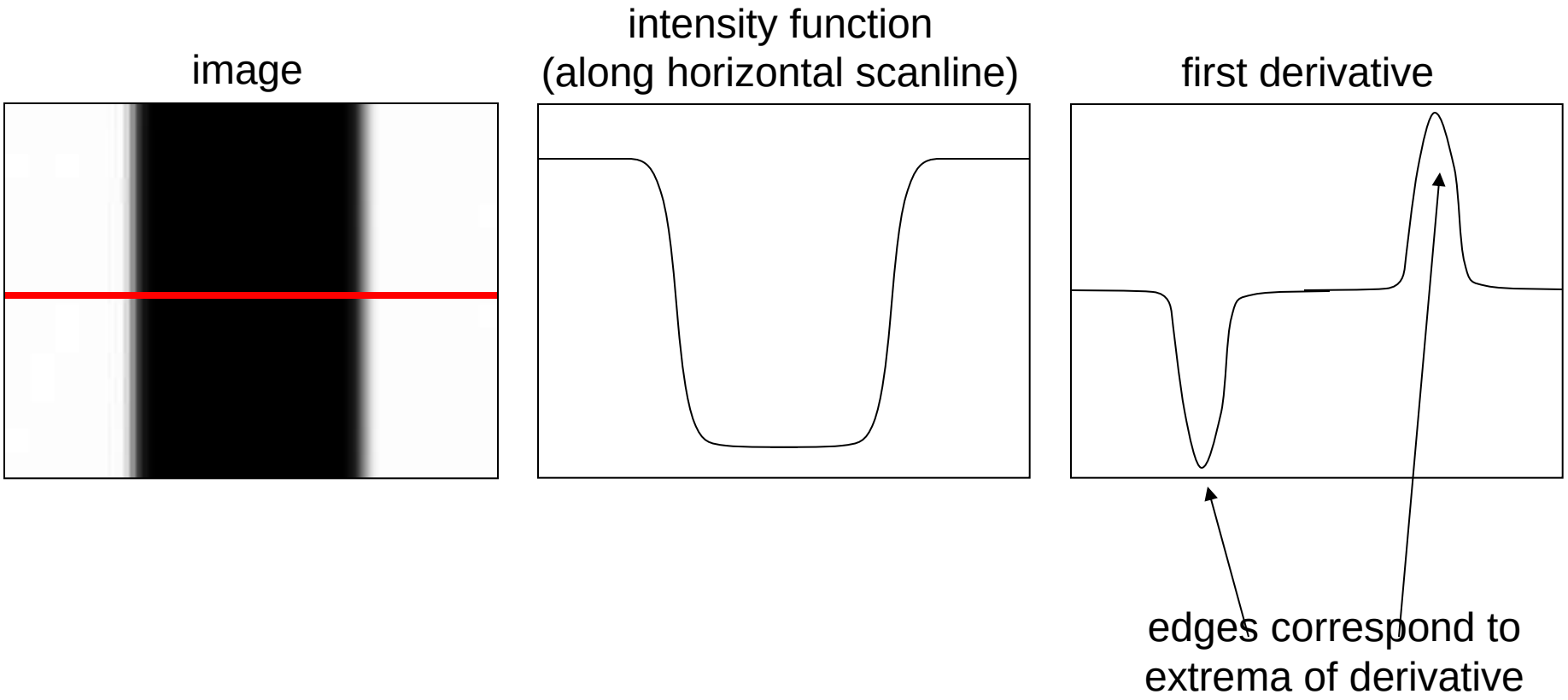


a b c
d e f

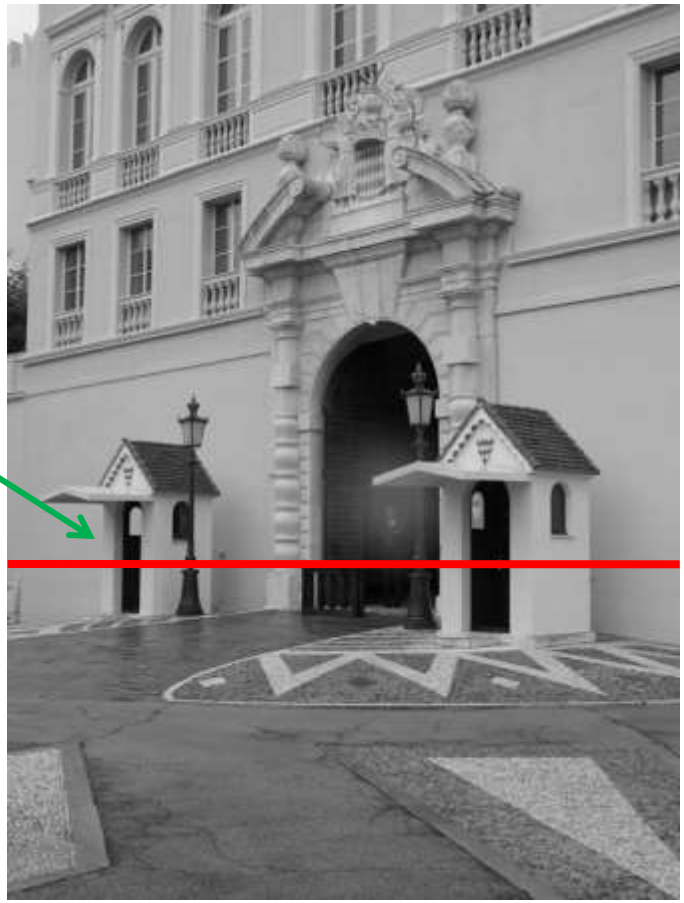
FIGURE 10.1 (a) Image containing a region of constant intensity. (b) Image showing the boundary of the inner region, obtained from intensity discontinuities. (c) Result of segmenting the image into two regions. (d) Image containing a textured region. (e) Result of edge computations. Note the large number of small edges that are connected to the original boundary, making it difficult to find a unique boundary using only edge information. (f) Result of segmentation based on region properties.

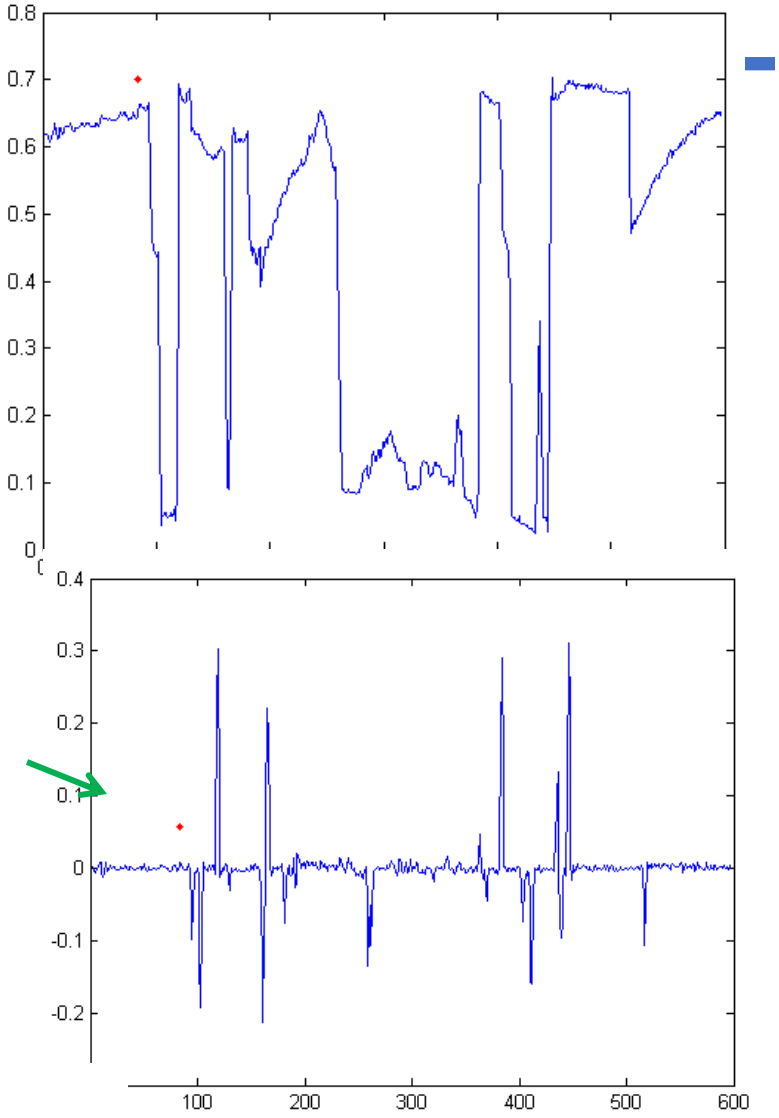
Source: D. Hoiem

- Characterizing edges: An edge is a place of rapid change in the image intensity function



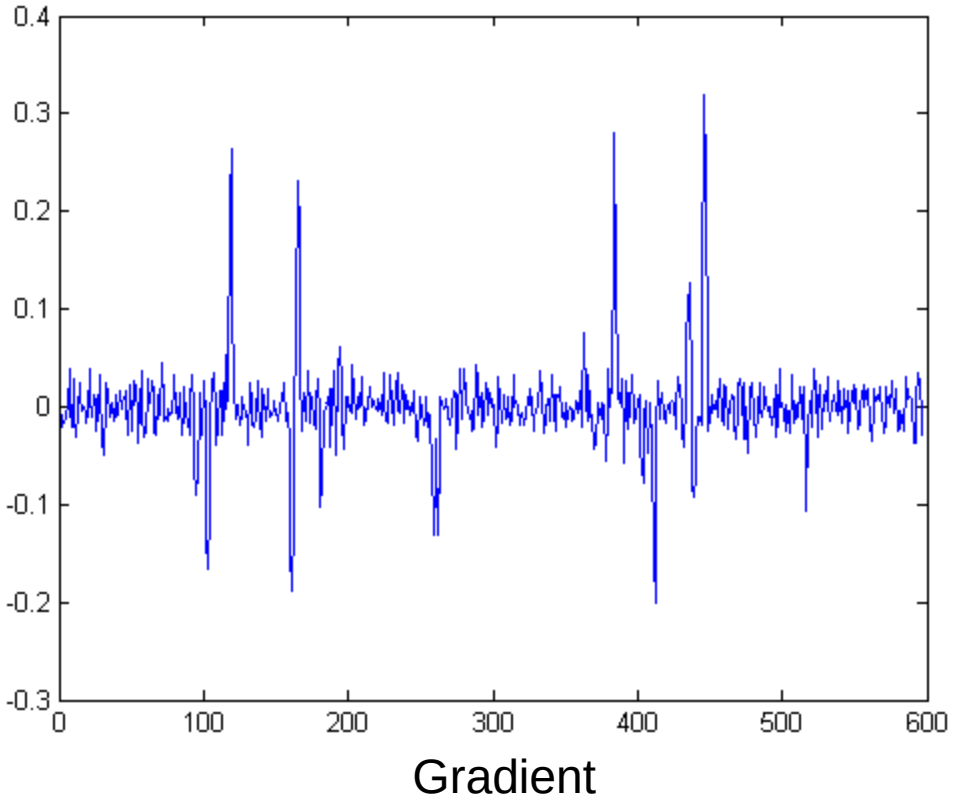
Intensity profile





Source: D. Hoiem

With a little Gaussian noise

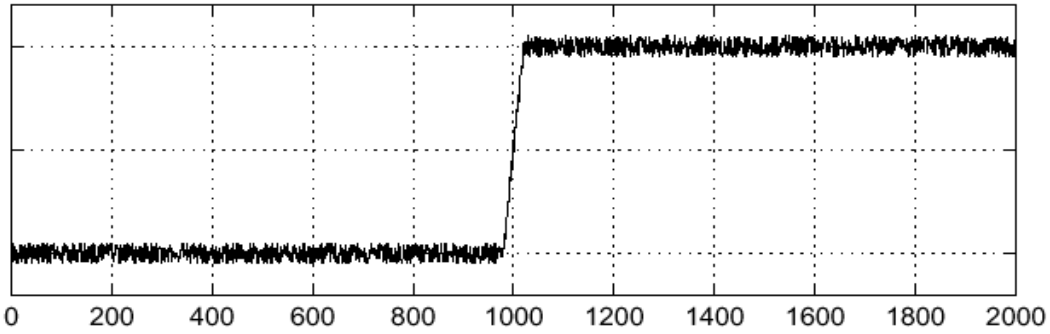


Source: D. Hoiem

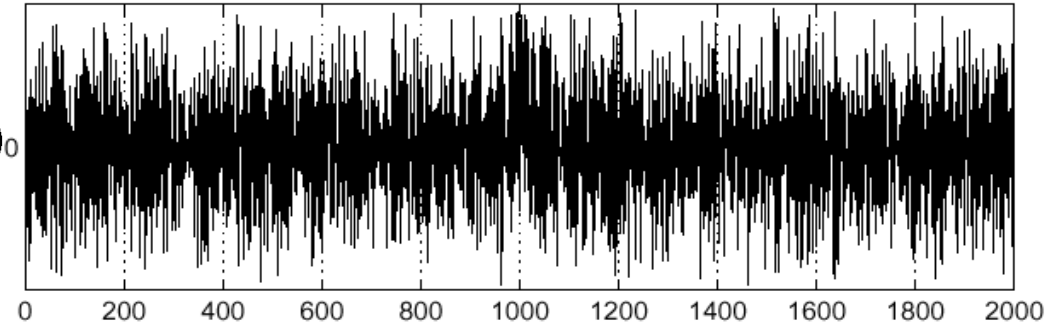
Effects of noise

- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal

$f(x)$



$\frac{d}{dx} f(x)$



Where is the edge?

Source: S. Seitz

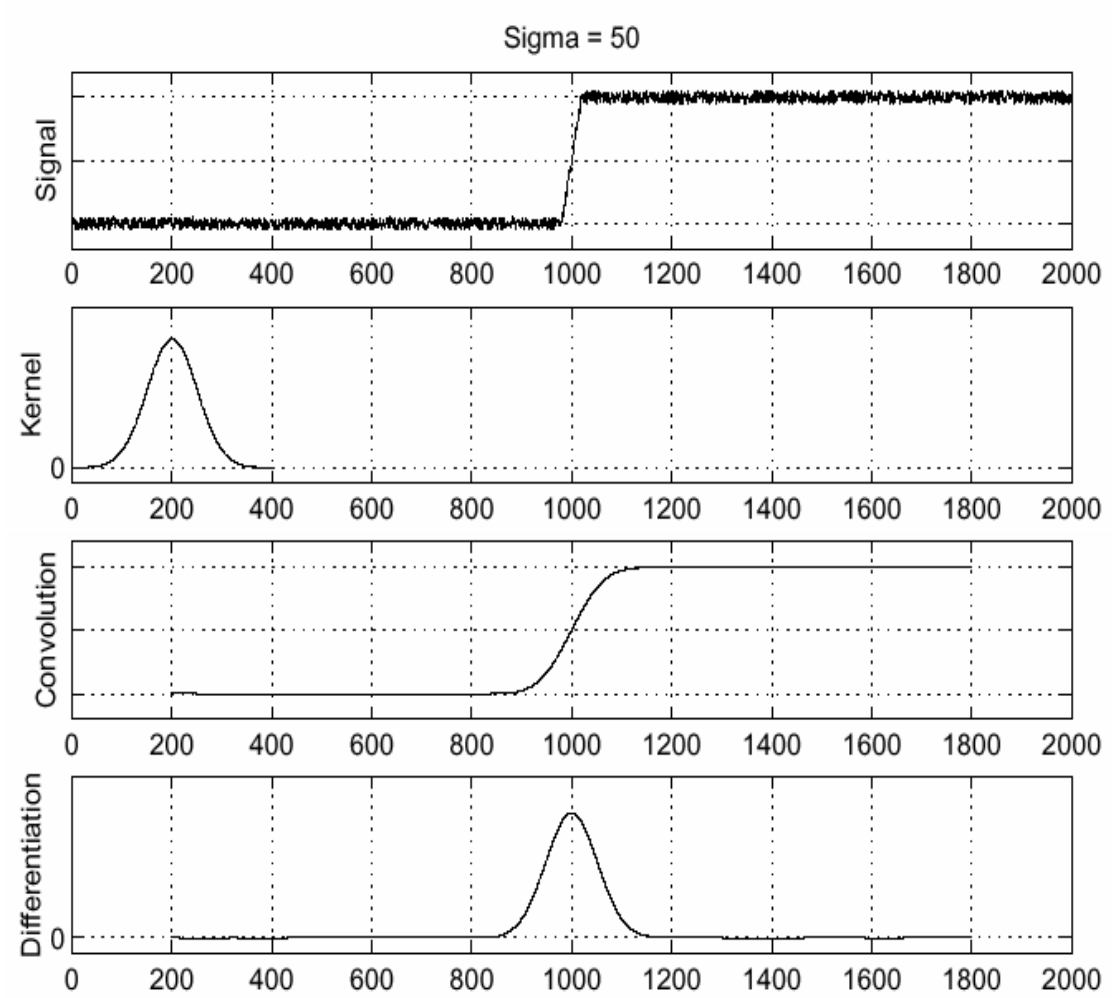
Effects of noise

- Difference filters respond strongly to noise
  - Image noise results in pixels that look very different from their neighbors
  - Generally, the larger the noise the stronger the response
- What can we do about it?

115	115	118	117
121	117	177	115
114	114	116	115
128	113	47	112

Source: D. Forsyth

Solution: smooth first



f

g

$f * g$

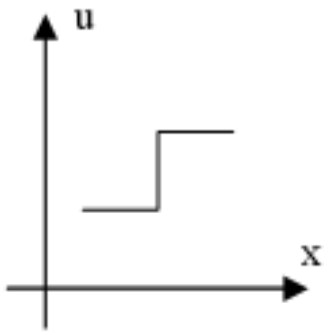
$\frac{d}{dx}(f * g)$

- To find edges, look for peaks in $\frac{d}{dx}(f * g)$

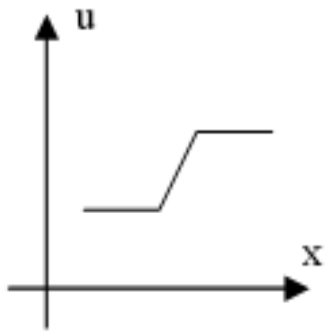
Source: S. Seitz

Mô hình biểu diễn đường biên

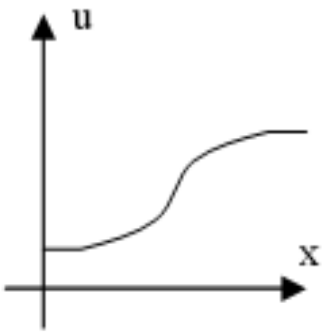
- Theo toán học, điểm ảnh có sự biến đổi mức xám u(x) một cách đột ngột.
- Hình minh họa:



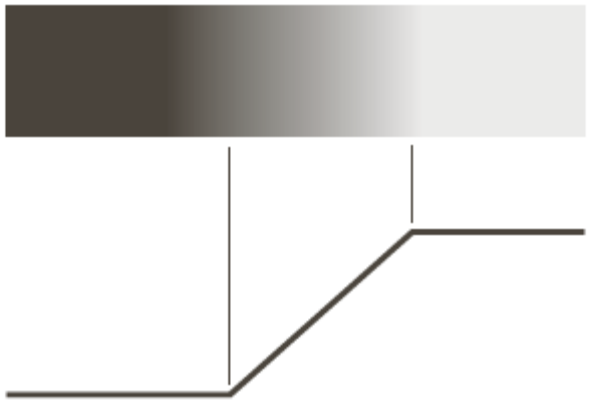
a, Đường biên lý tưởng



b, Đường biên bậc thang



c, Đường biên thực



2. Phương pháp phát hiện biên

- Phương pháp trực tiếp: dựa vào sự biến thiên độ sáng của điểm ảnh để làm nổi biên bằng kỹ thuật đạo hàm.
  - Đạo hàm bậc nhất của ảnh: phương pháp Gradient
  - Đạo hàm bậc hai của ảnh: phương pháp Laplace.

    Hai phương pháp này được gọi là phương pháp ***dò biên cục bộ***.
  - Nếu dùng nguyên lý quy hoạch động -> phương pháp *dò biên tổng thể*
- Phương pháp gián tiếp: Nếu ảnh đã được phân thành các vùng ảnh khác nhau thì đường phân cách giữa các vùng đó chính là biên (Bài toán *phân vùng* và *phát hiện biên* ảnh là 2 bài toán đối ngẫu).

2.1. Phát hiện biên dùng gradient

- Gradient là một vectơ có các thành phần biểu thị tốc độ thay đổi mức xám của điểm ảnh (theo hai hướng *x,y* đối với ảnh 2 chiều) tức là:

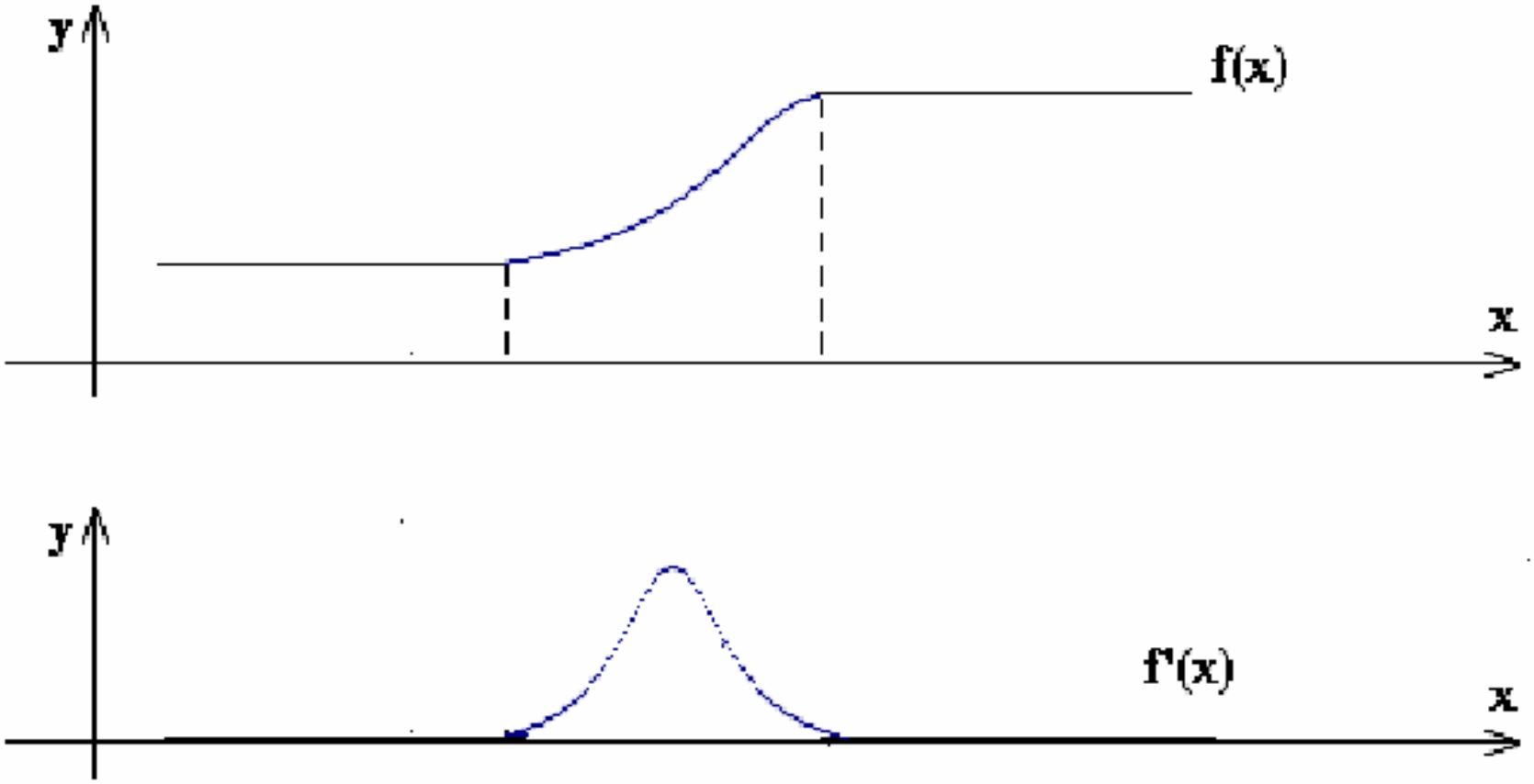
$$\Delta f = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix}$$

- Ta có:

$$\frac{\partial f(x,y)}{\partial x} = f'_x \approx \frac{f(x+dx,y) - f(x,y)}{dx}$$

$$\frac{\partial f(x,y)}{\partial y} = f'_y \approx \frac{f(x,y+dy) - f(x,y)}{dy}$$

  - Trong đó *dx, dy* là khoảng cách giữa 2 điểm kế cận theo hướng x,y tương ứng (thực tế chọn *dx*=*dy*=1)


y
f(x)
x
y
f'(x)
x

- Nếu tính toán Gradient trên ảnh, việc xử lý sẽ rất phức tạp.
- Để đơn giản, người ta tính toán Gradient thông qua dùng cặp mặt nạ trực giao H_1, H_2
- Nếu định nghĩa G_x, G_y tương ứng là Gradient theo hai hướng x,y, khi đó vector Gradient của một ảnh f(x,y) là:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

Ta có $$|\nabla f| = \sqrt{G_x^2 + G_y^2}$$

- Ví dụ biên độ G*(m,n)* tại điểm *(m,n)* được tính:

$$G(m,n) = \sqrt{G_x^2(m,n) + G_y^2(m,n)}$$

- Để giảm độ phức tạp tính toán, G(m,n) được tính gần đúng như sau:

$$G(m,n) = |G_x(m,n)| + |G_y(m,n)|$$

- Một số toán tử Gradien tiêu biểu:

Prewitt, Sobel, Robert, đẳng hướng (Isometric), 4-lân cận.

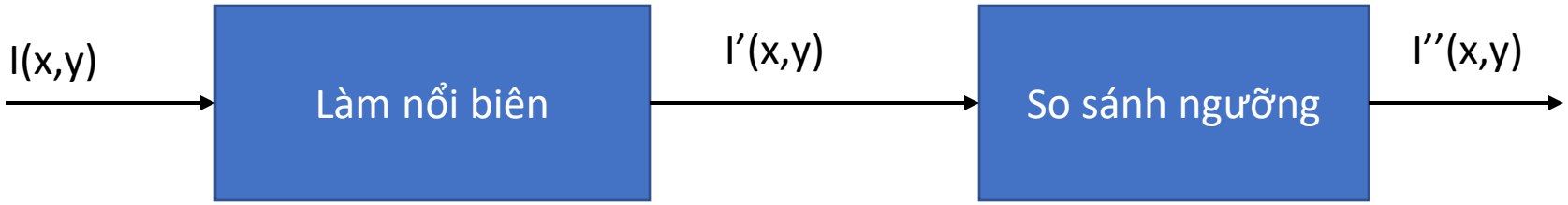
Edge normal: $\nabla f \equiv grad(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$

Edge unit normal: $\nabla f \,/\, \text{mag}(\nabla f)$

In practice,sometimes the magnitude is approximated by

$$\text{mag}(\nabla f) = \left|\frac{\partial f}{\partial x}\right| + \left|\frac{\partial f}{\partial y}\right| \text{ or } \text{mag}(\nabla f) = \max\left(\left|\frac{\partial f}{\partial x}\right|, \left|\frac{\partial f}{\partial y}\right|\right)$$

- Các công đoạn phát hiện biên theo kỹ thuật Gradient



- Thực tế: việc làm nổi biên là nhân chập ảnh I với một mặt nạ (ma trận lọc) sẽ được giới thiệu trong phần tiếp theo

2.1.1. Kỹ thuật Prewitt

- Kỹ thuật sử dụng 2 mặt nạ nhập chập 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 $I \otimes H_x$ và $I \otimes H_y$
- Bước 2: Tính $I \otimes H_x + I \otimes H_y$

- Ví dụ:

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

$$I \otimes H_x = \begin{pmatrix} 0 & 0 & -10 & -10 & * & * \\ 0 & 0 & -15 & -15 & * & * \\ 0 & 0 & -10 & -10 & * & * \\ 0 & 0 & -5 & -5 & * & * \\ * & * & * & * & * & * \\ * & * & * & * & * & * \end{pmatrix}$$

$$I \otimes H_y = \begin{pmatrix} 15 & 15 & 10 & 5 & * & * \\ 0 & 0 & 0 & 0 & * & * \\ -15 & -15 & -10 & -5 & * & * \\ -15 & -15 & -10 & -5 & * & * \\ * & * & * & * & * & * \\ * & * & * & * & * & * \end{pmatrix}$$

$$I \otimes H_x + I \otimes H_y = \begin{pmatrix} 15 & 15 & 0 & -5 & * & * \\ 0 & 0 & -15 & -15 & * & * \\ -15 & -15 & -20 & -15 & * & * \\ -15 & -15 & -15 & -10 & * & * \\ * & * & * & * & * & * \\ * & * & * & * & * & * \end{pmatrix}$$

- Ngoài ra để phát hiện biên theo đường chéo ta sử dụng 2 mặt nạ:

$$H_1 = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}; \quad H_2 = \begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

2.1.2. Kỹ thuật Sobel

- Tương tự như kỹ thuật Prewitt kỹ thuật Sobel sử dụng 2 mặt nạ nhân chập theo 2 hướng x, y là:

$$H_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

$$H_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

- Bước 1: Tính $I \otimes H_x$ và $I \otimes H_y$
- Bước 2: Tính $I \otimes H_x + I \otimes H_y$

2.1.3. Kỹ thuật Robert

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

Hướng ngang (x)

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

Hướng dọc (y)

Ưu: chỉ sử dụng 4 pixel, tính toán nhanh

Nhược: Mặt nạ nhỏ, nhạy với nhiễu

2.1.4. Kỹ thuật la bàn

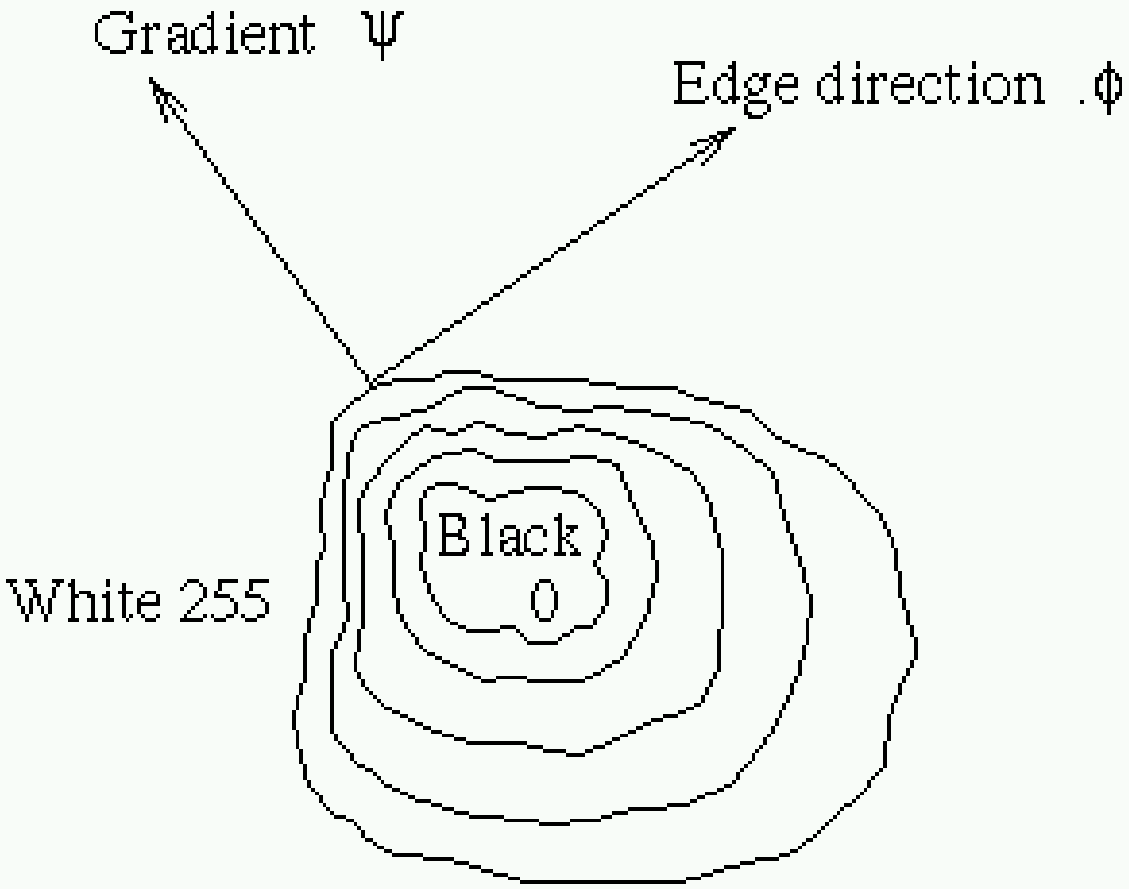
- Kỹ thuật sử dụng 8 mặt nạ nhân chập theo 8 hướng 0^0, 45^0, 90^0, 135^0, 180^0, 225^0, 270^0, 315^0
- Các bước tính toán thuật toán La bàn
  - Bước 1: Tính $I \otimes H_i$; i = 1,8
  - Bước 2: Tính $\sum_{i=1}^{8} I \otimes H_i$

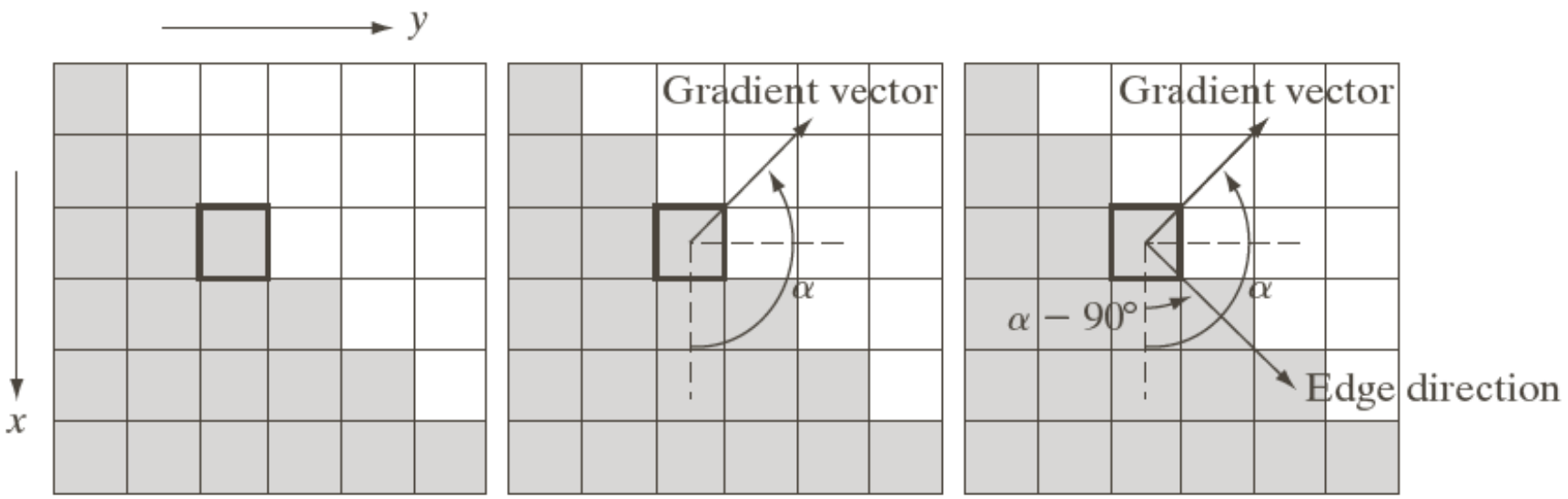
$$H_1 = \begin{pmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{pmatrix} \qquad H_2 = \begin{pmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{pmatrix}$$

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

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

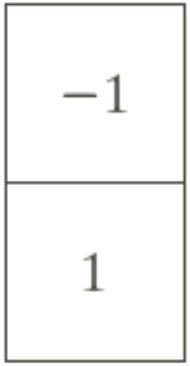
$$H_7 = \begin{pmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{pmatrix} \qquad H_8 = \begin{pmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{pmatrix}$$


Gradient Ψ
Edge direction .ϕ
White 255
Black
0



a b c

FIGURE 10.12 Using the gradient to determine edge strength and direction at a point. Note that the edge is perpendicular to the direction of the gradient vector at the point where the gradient is computed. Each square in the figure represents one pixel.



a b

FIGURE 10.13 One-dimensional masks used to implement Eqs. (10.2-12) and (10.2-13).

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0	0	−1
0	1	1	0

Roberts

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

Prewitt

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

Sobel

a
b c
d e
f g

FIGURE 10.14
A 3 × 3 region of an image (the z's are intensity values) and various masks used to compute the gradient at the point labeled z_5.

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.

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, $|g_x| + |g_y|$.

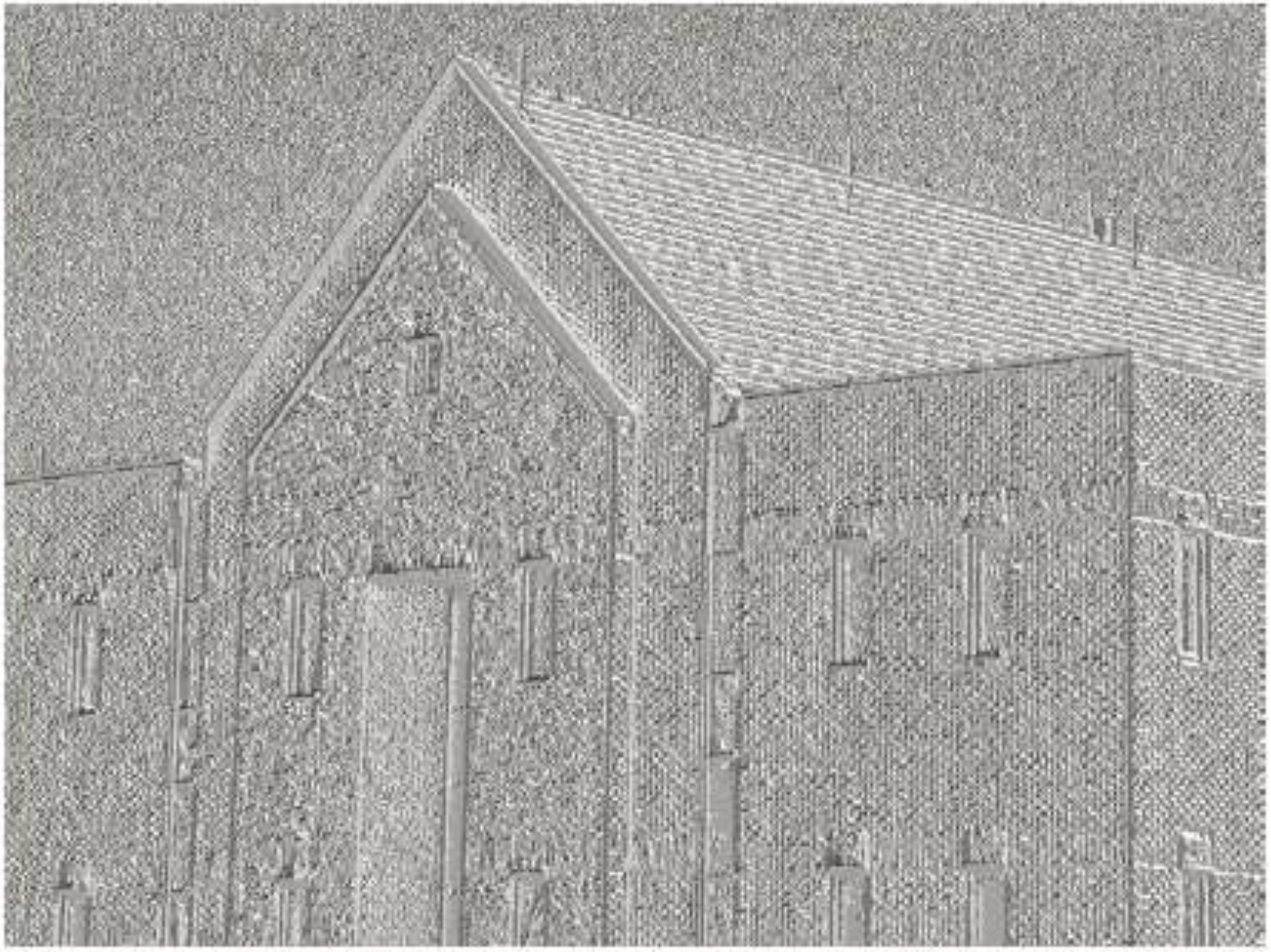
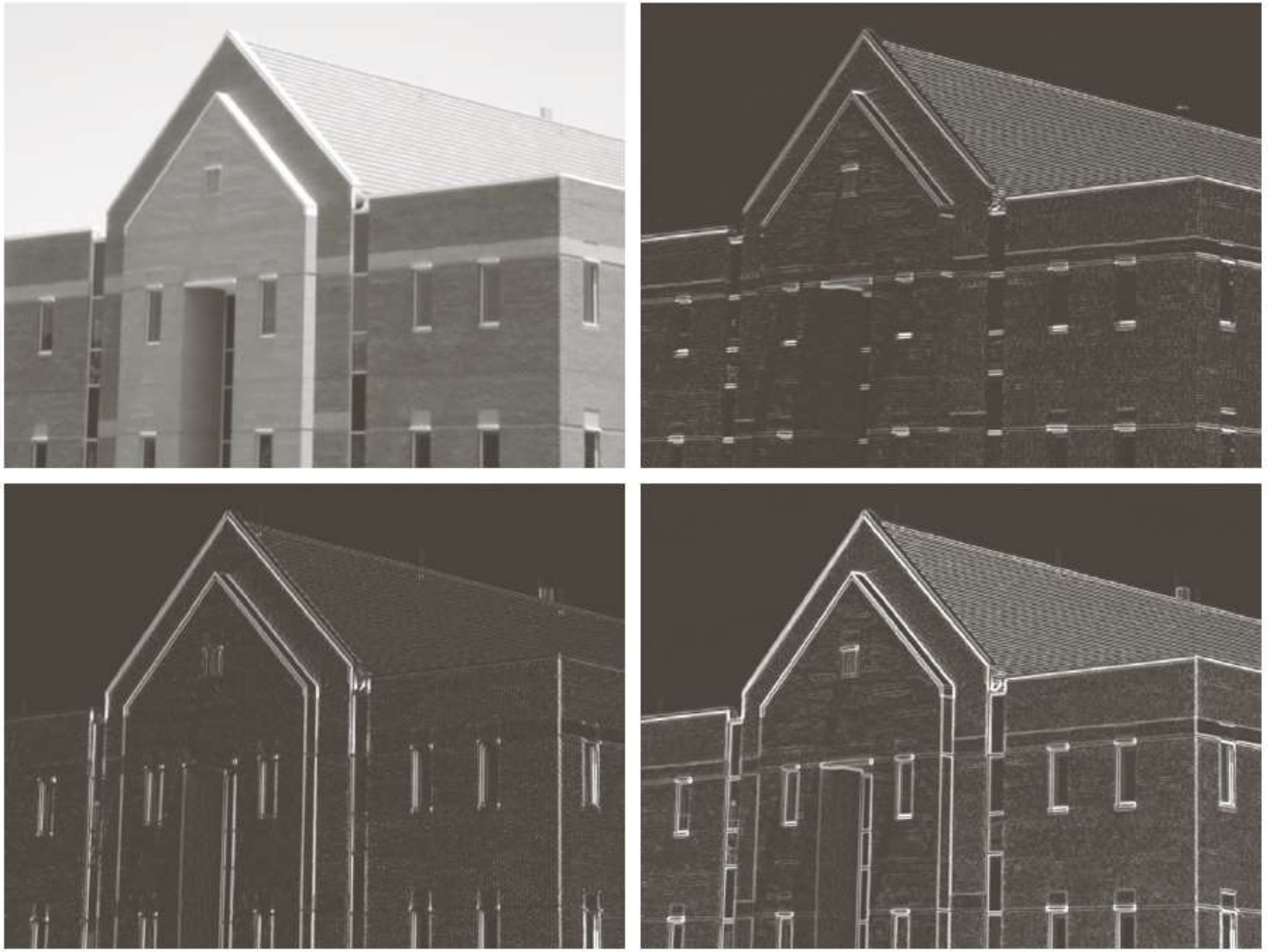
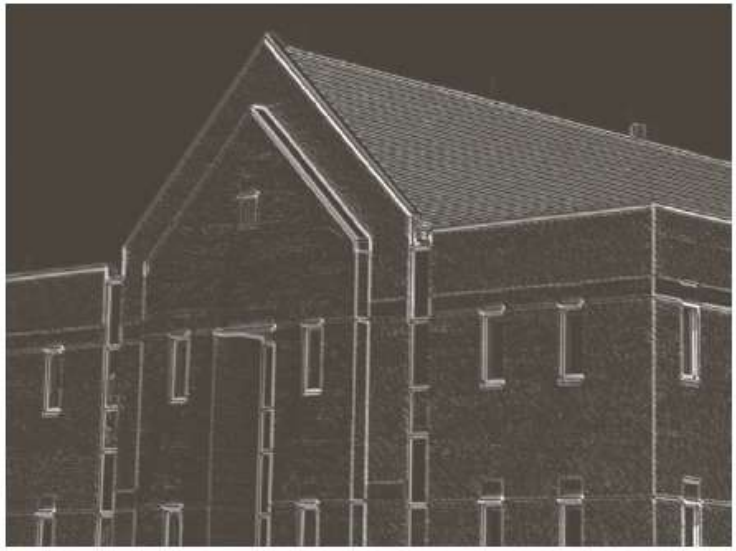


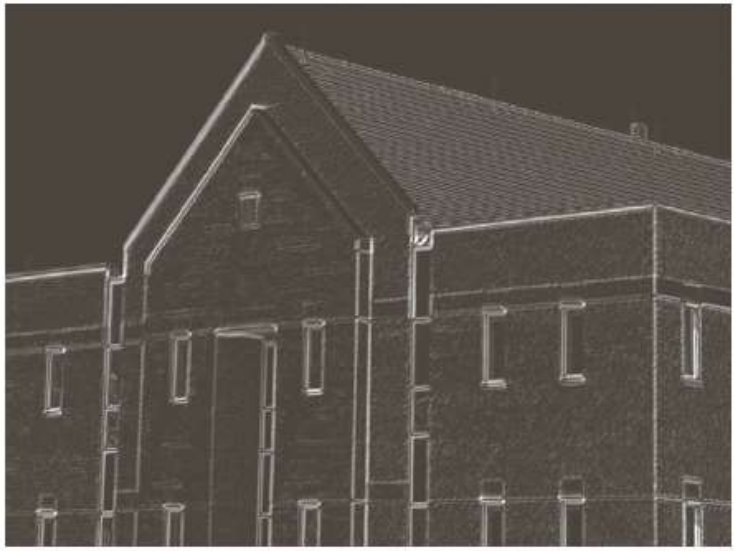
FIGURE 10.17 Gradient angle image computed using Eq. (10.2-11). Areas of constant intensity in this image indicate that the direction of the gradient vector is the same at all the pixel locations in those regions.

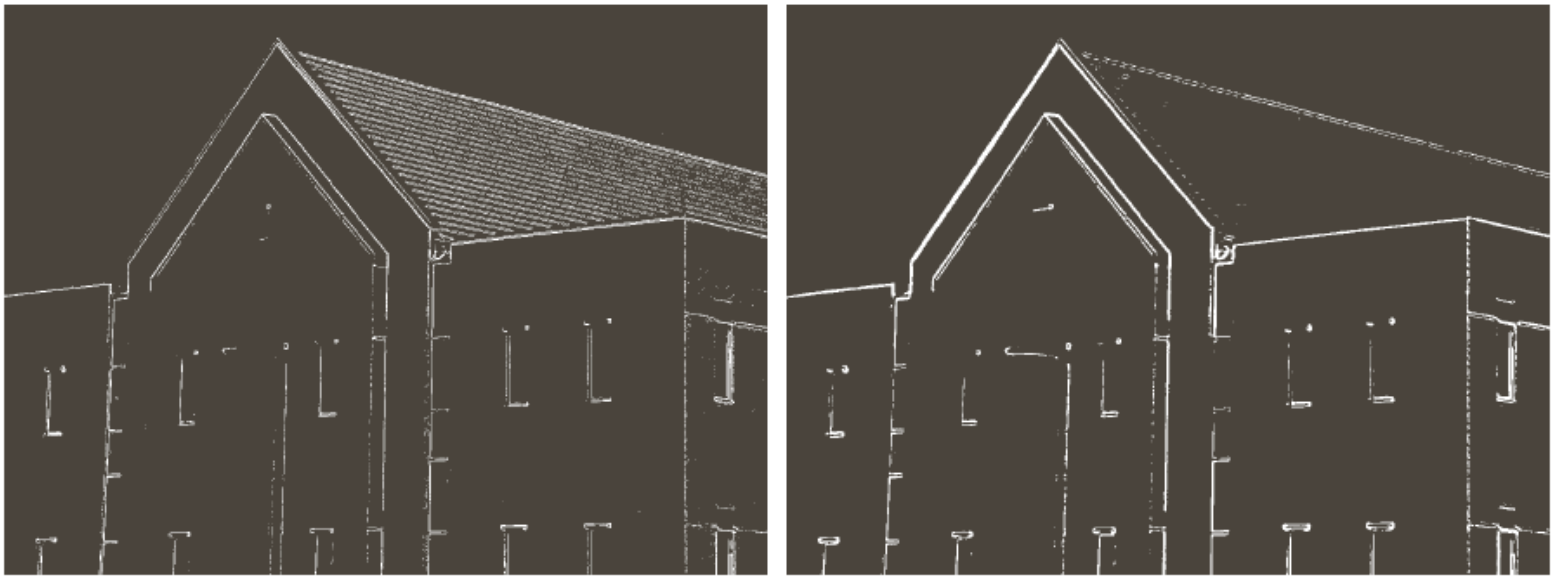


a b
c d

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.







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.

2.2. Kỹ thuật Laplace

- Các phương pháp đánh giá gradient ở trên làm việc khá tốt khi ảnh có cường độ sáng thay đổi rõ rệt.
- Nhược điểm: nhạy cảm với nhiễu và tạo các biên kép làm chất lượng biên thu được không cao.
- Khi mức xám thay đổi chậm, miền chuyển tiếp trải rộng, phương pháp hiệu quả hơn là sử dụng đạo hàm bậc hai Laplace
- Toán tử Laplace được xây dựng trên cơ sở đạo hàm bậc 2 của hàm biến đổi mức xám.

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

- $\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x}\left(\frac{\partial f}{dx}\right) \approx \frac{\partial}{\partial x}\big(f(x+1,y) - f(x,y)\big)$

$\approx \big[\big(f(x+1,y) - f(x,y)\big)\big]$

$\quad - \big[\big(f(x,y) - f(x-1,y)\big)\big]$

$\approx f(x+1,y) - 2f(x,y) + f(x-1,y)$

Tương tự:

- $\frac{\partial^2 f}{\partial y^2} \approx f(x,y+1) - 2f(x,y) + f(x,y-1)$
- $\Delta^2 f \approx f(x+1,y) + f(x-1,y) - 4f(x,y) + f(x,y+1) + f(x,y-1)$

- 3 kiểu mặt nạ thường dùng:

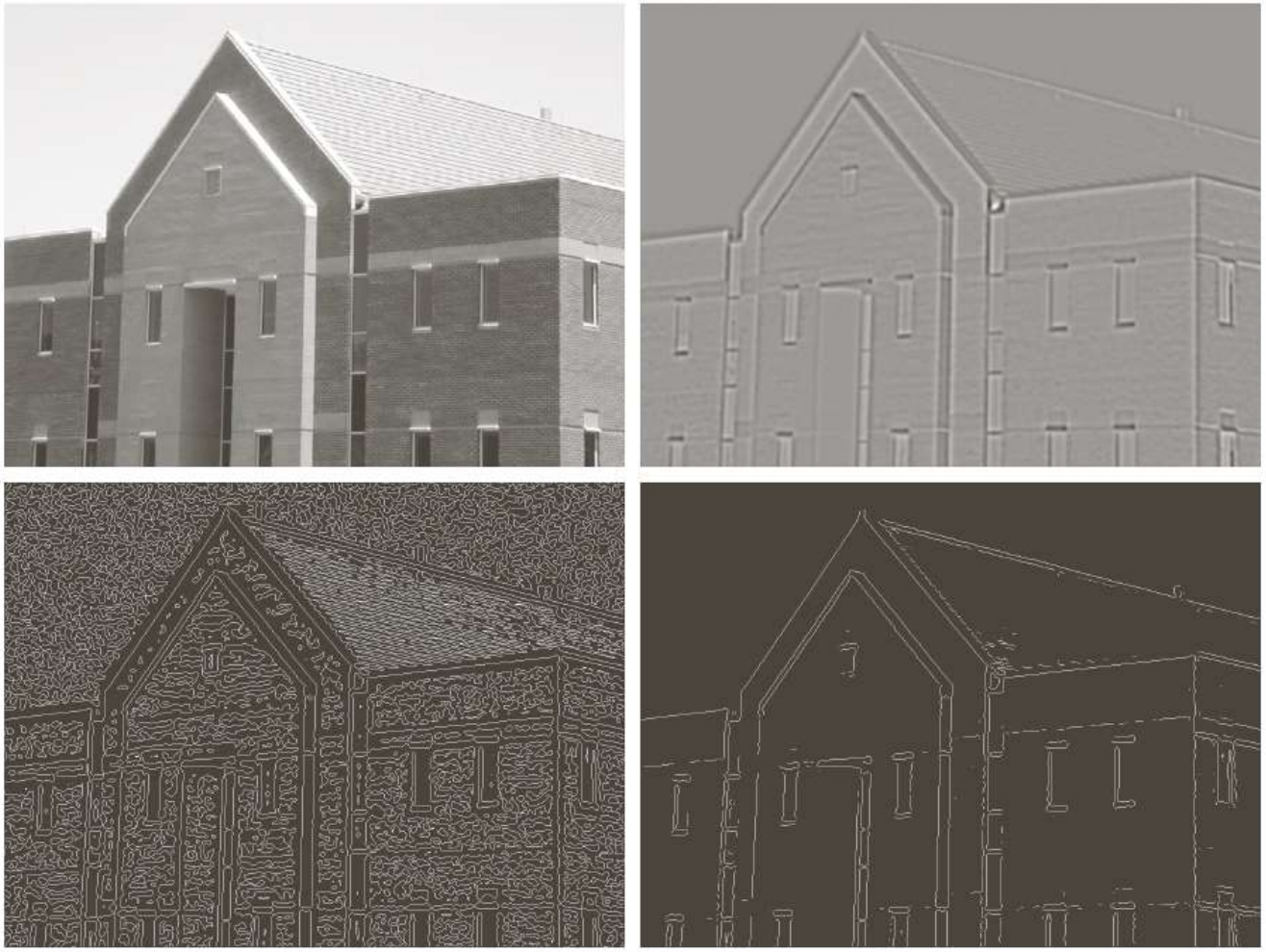
$$H_1 = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix} \qquad H_2 = \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix} \qquad H_3 = \begin{pmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{pmatrix}$$

- Lưu ý:
  - Mặt nạ Laplace cũng tương tự như các mặt nạ sử dụng phương pháp Gradient, nhưng nó đơn giản hơn ở chỗ chỉ dùng 1 mặt nạ thay vì 2.
  - Kỹ thuật theo toán tử Laplace tạo đường biên mảnh (độ rộng 1 pixel).
  - Kỹ thuật này rất nhạy với nhiễu, đường biên thu được thường kém ổn định.

Laplace of Gaussian (LoG)

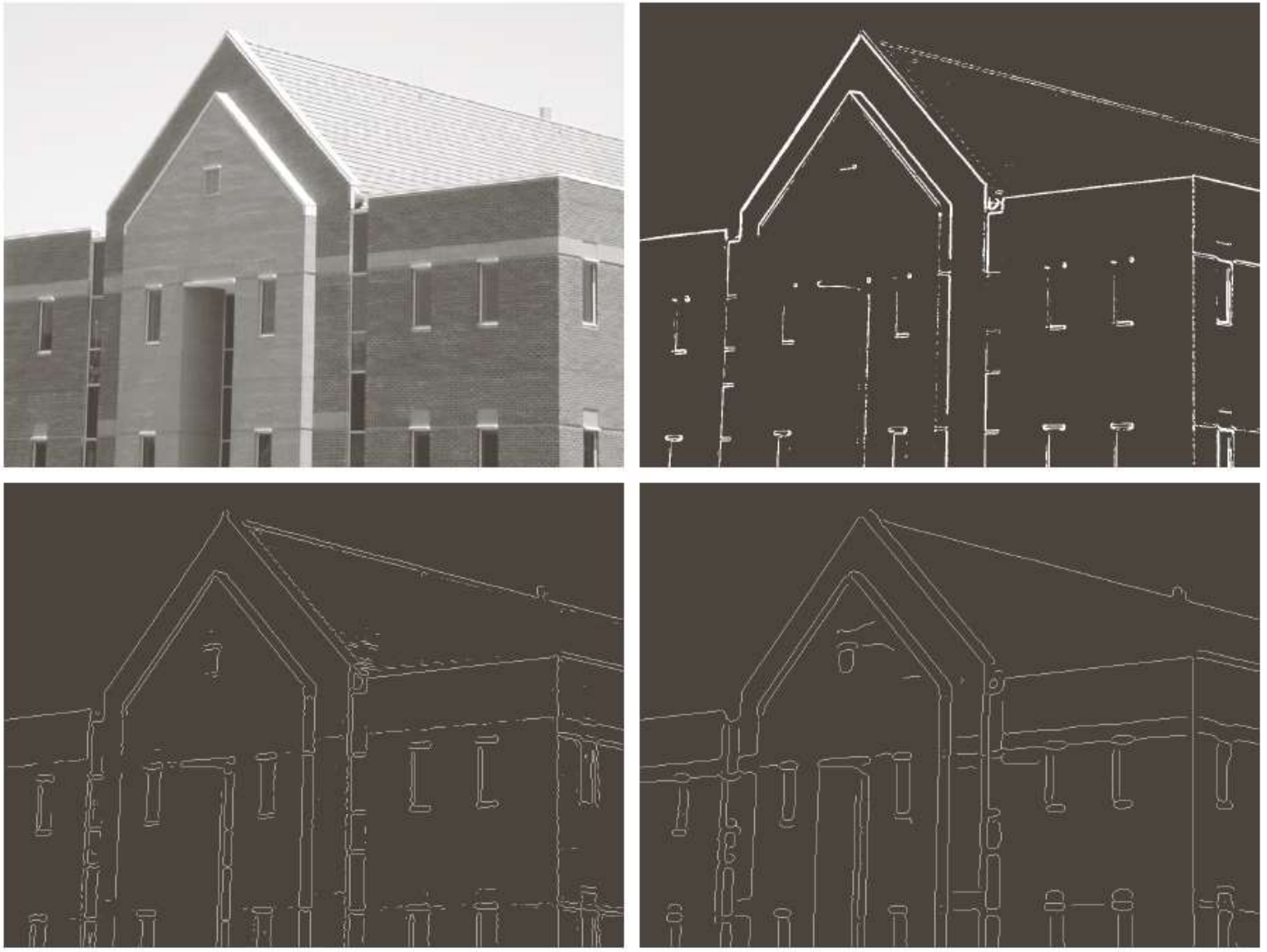
- Các mặt nạ sử dụng trong kỹ thuật Laplace đều là xấp xỉ của đạo hàm bậc 2 nên rất nhạy với nhiễu. Để khắc phục người ta thường làm trơn ảnh bởi bộ lọc Gaussian trước khi áp dụng toán tử Laplace. Quá trình này làm giảm các thành phần nhiễu có tần số cao trước khi lấy đạo hàm.
- LoG (`Laplacian of Gaussian') thường được tính nhu sau: nhân chập bộ lọc Gaussian với bộ lọc Laplace, sau đó nhân bộ lọc “lai” này với ảnh để thu được kết quả cuối cùng.

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$$



a b
c d

FIGURE 10.22
(a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) Results of Steps 1 and 2 of the Marr-Hildreth algorithm using $\sigma = 4$ and $n = 25$. (c) Zero crossings of (b) using a threshold of 0 (note the closed-loop edges). (d) Zero crossings found using a threshold equal to 4% of the maximum value of the image in (b). Note the thin edges.



a b
c d

FIGURE 10.25
(a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1].
(b) Thresholded gradient of smoothed image.
(c) Image obtained using the Marr-Hildreth algorithm.
(d) Image obtained using the Canny algorithm.
Note the significant improvement of the Canny image compared to the other two.

$T_L = 0.04; T_H = 0.10; \sigma = 4$ and a mask of size 25×25

a b
c d

FIGURE 10.26
(a) Original head CT image of size 512 × 512 pixels, with intensity values scaled to the range [0, 1]. (b) Thresholded gradient of smoothed image. (c) Image obtained using the Marr-Hildreth algorithm. (d) Image obtained using the Canny algorithm. (Original image courtesy of Dr. David R. Pickens, Vanderbilt University.)

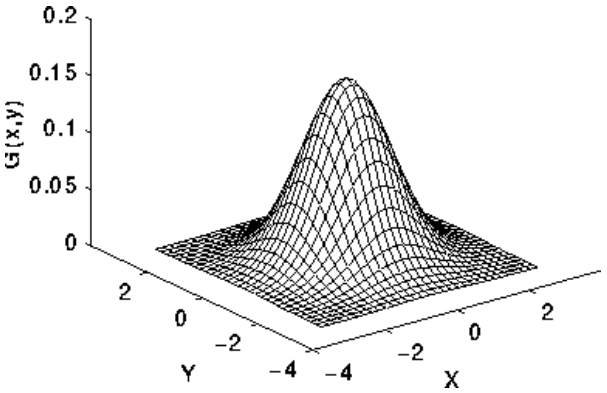
$T_L = 0.05; T_H = 0.15; \sigma = 2$ and a mask of size 13×13

Kỹ thuật Canny

- Đây là phương pháp tách đường biên dựa trên toán tử đạo hàm được dùng rất phổ biến
  - Phương pháp xấp xỉ đạo hàm bậc nhất của Gauss đạt hiệu quả cao cho việc khử nhiễu (khắc phục nhược điểm của phương pháp đạo hàm)
  - Là một bộ phát hiện biên được tối ưu hóa
  - Áp dụng cho ảnh đa mức xám

❖ Nhắc lại: bộ lọc Gaussian làm trơn ảnh: các hệ số là các giá trị Gaussian, có trọng số cao hơn cho pixel ở tâm, nhỏ dần cho các pixel ở lân cận

$$G_\sigma \equiv \frac{1}{2\pi\sigma^2}\exp\{-\frac{x^2+y^2}{2\sigma^2}\}$$



Kỹ thuật Canny

$$\nabla f = \nabla(G \otimes I) = f_x + f_y$$

I - ảnh

fx, fy – các đạo hàm riêng

$$\nabla f = \nabla(G \otimes I)_x + \nabla(G \otimes I)_y = (G_x \otimes I) + (G_y \otimes I)$$

$$G_x(x,y) = G_x(x) \otimes G(y) \text{ và } G_y(x,y) = G_y(y) \otimes G(x)$$

Từ đó ta có:

$$f_x(x,y) = G_x(x) \otimes G(y) \otimes I \text{ và } f_y(x,y) = G_y(y) \otimes G(x) \otimes I)$$

(Chỉ cần tính các đạo hàm riêng của Gaussian rồi nhân chập)

Canny edge detection algorithm

1. Filter image with derivative of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width
4. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high
   - Use the high threshold to start edge curves and the low threshold to continue them

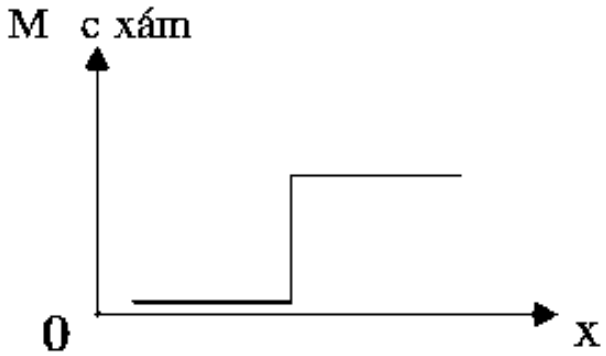
Nhân xét

- Các kỹ thuật dò biên đã nêu ở trên (Gradient, Laplace):
  - Rất nhạy cảm với nhiễu.
  - Thực tế chỉ làm nổi biên
  - Khó điều chỉnh ảnh biên
  - Khó sử dụng cho việc nhận dạng đối tượng

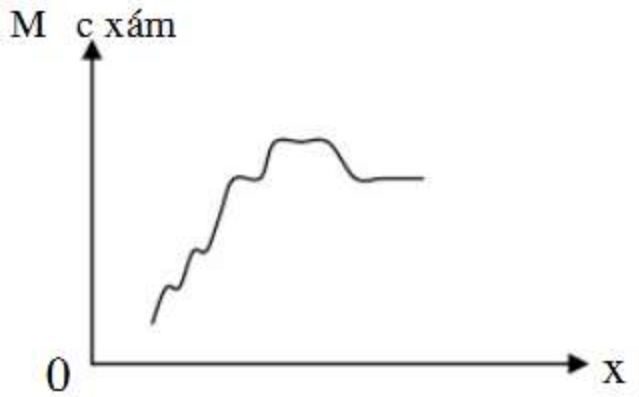
➔Các kỹ thuật này chưa đạt được sự hoàn thiện như mong muốn.

Lý do

- Một cách lý tưởng (ảnh đen trắng) đồ thị sự biến thiên mức xám của các điểm ảnh :



- Đối với các ảnh đa mức xám thì đồ thị có dạng



Liên kết các điểm biên và phát hiện đường biên

- Các kỹ thuật đã trình bày ở trên đều dùng để phát hiện điểm biên
- Cần có các giải thuật liên kết các điểm biên thành cạnh/biên hoặc đường bao (region boundaries)

- 3 hướng tiếp cận: Three approaches to edge linking

  Xử lý cục bộ (Local processing)

  Xử lý vùng (Regional processing)

  Xử lý toàn ảnh (Global processing)

3. Phát hiện đường thẳng

- Line detection using Hough transform
- RANSAC Algorithm

Hypothesize and test

1. Propose parameters
   - Try all possible
   - Each point votes for all consistent parameters
   - Repeatedly sample enough points to solve for parameters

2. Score the given parameters
   - Number of consistent points, possibly weighted by distance

3. Choose from among the set of parameters
   - Global or local maximum of scores

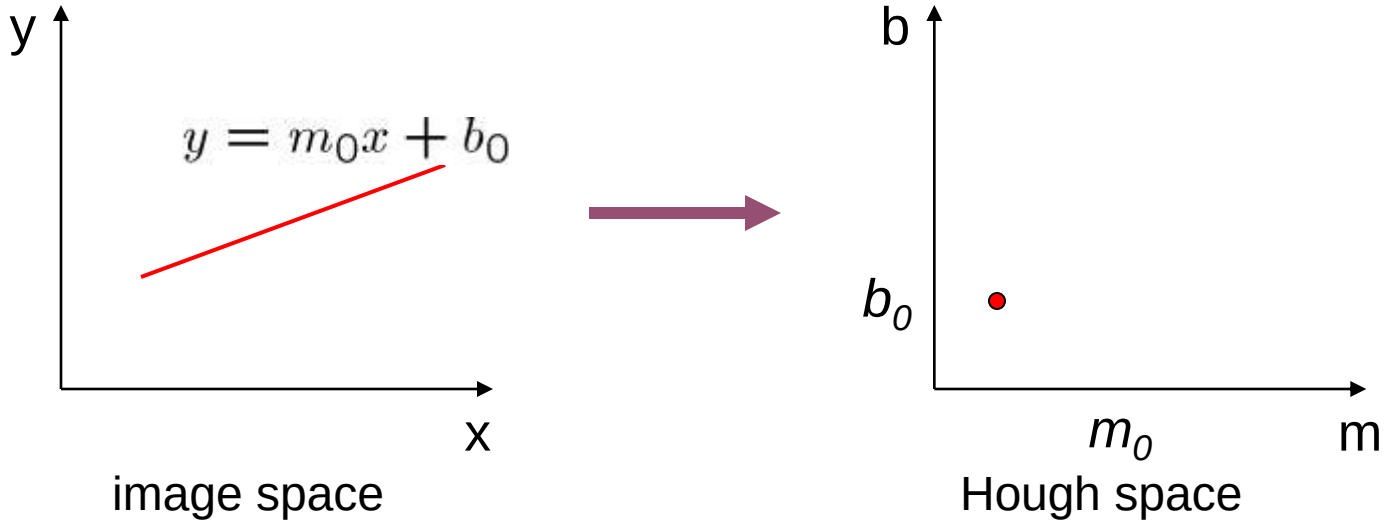
4. Possibly refine parameters using inliers

The following Slides are from Computer Vision, CS 143, Brown, by James Hays

Finding lines in an image

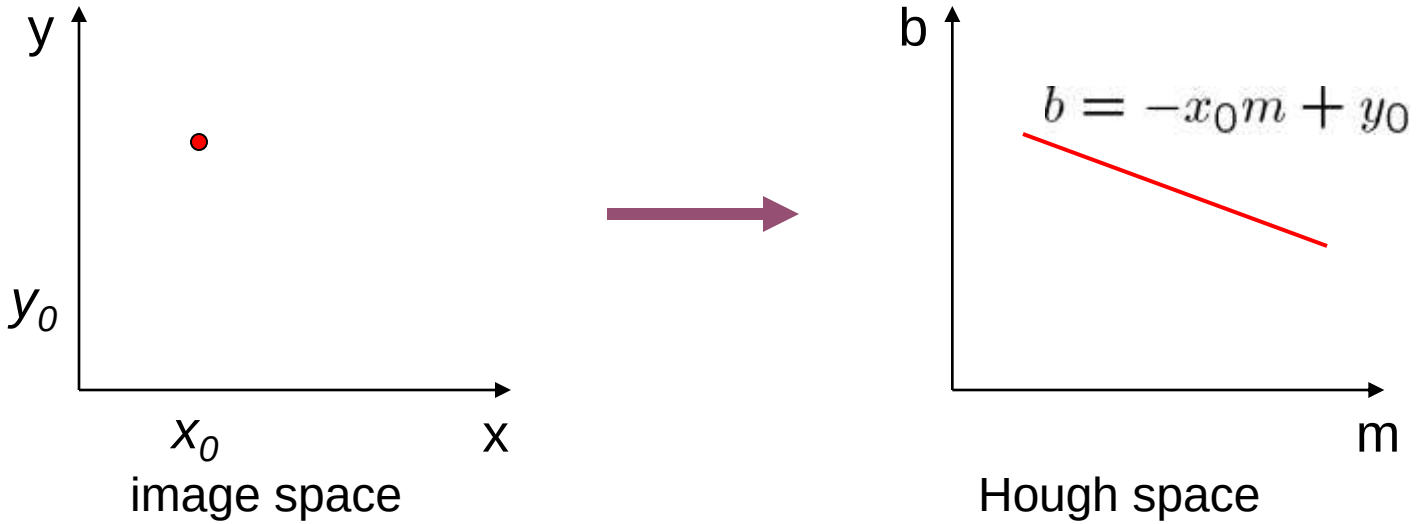
- Option 1:
  - Search for the line at every possible position/orientation
  - What is the cost of this operation?
- Option 2:
  - Use a voting scheme: Hough transform

Finding lines in an image



- Connection between image (x,y) and Hough (m,b) spaces
  - A line in the image corresponds to a point in Hough space
  - To go from image space to Hough space:
    - given a set of points (x,y), find all (m,b) such that y = mx + b

Finding lines in an image

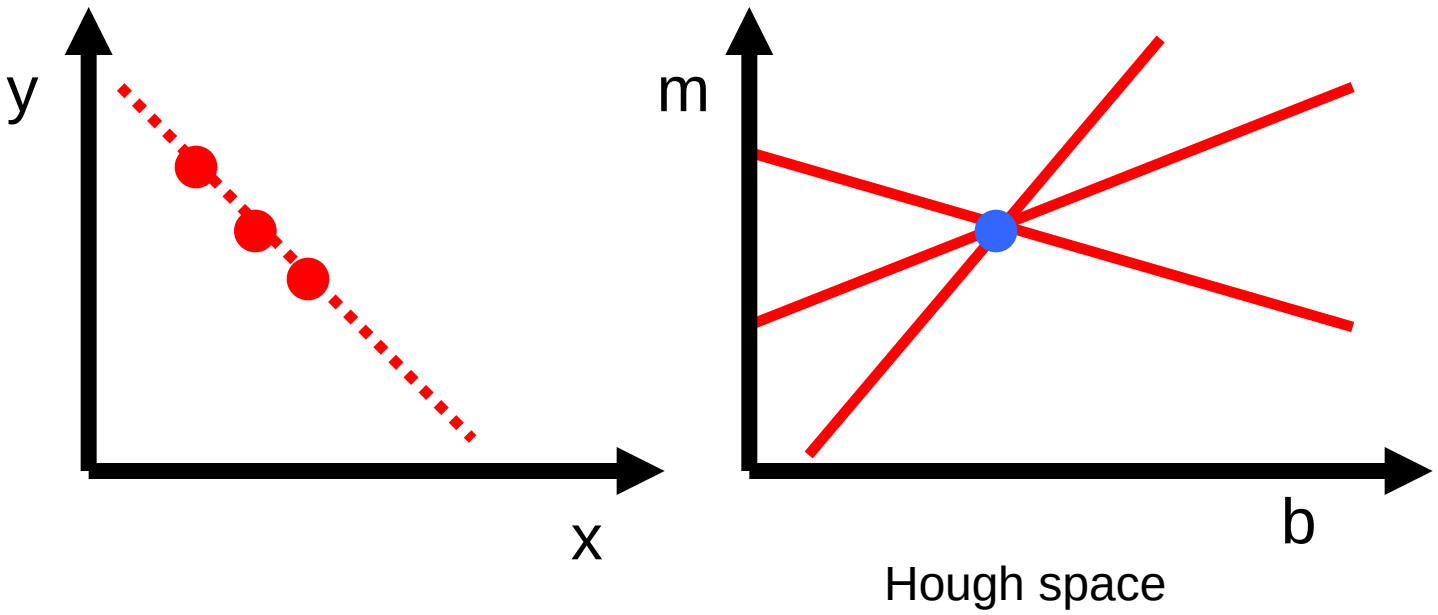


- Connection between image (x,y) and Hough (m,b) spaces
  - A line in the image corresponds to a point in Hough space
  - To go from image space to Hough space:
    - given a set of points (x,y), find all (m,b) such that y = mx + b
  - What does a point (x_0, y_0) in the image space map to?
    - A: the solutions of $b = -x_0m + y_0$
    - this is a line in Hough space

Hough transform

P.V.C. Hough, *Machine Analysis of Bubble Chamber Pictures,* Proc. Int. Conf. High Energy Accelerators and Instrumentation, 1959

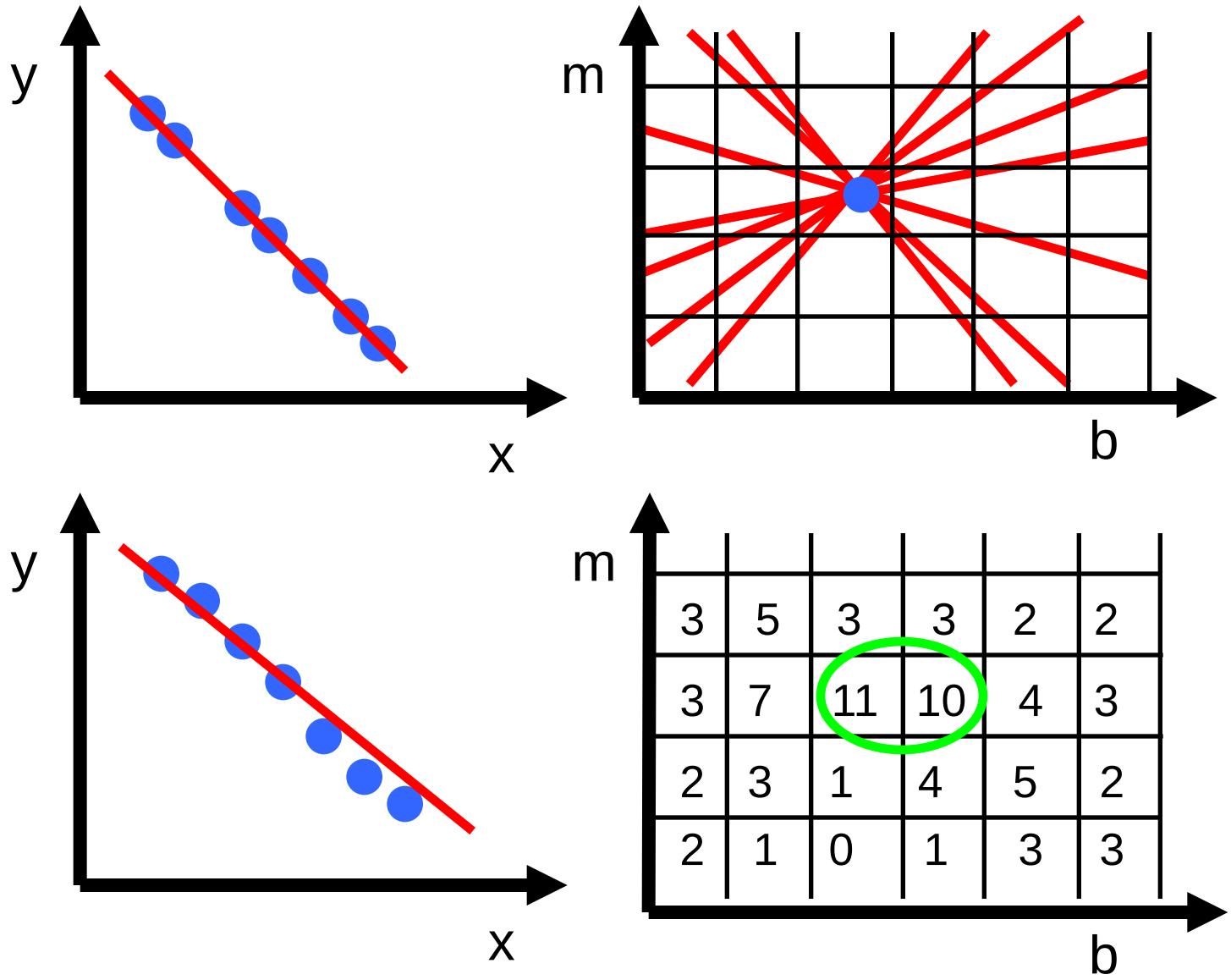
Given a set of points, find the curve or line that explains the data points best



$$y = m\,x + b$$

Slide from S. Savarese

Hough transform



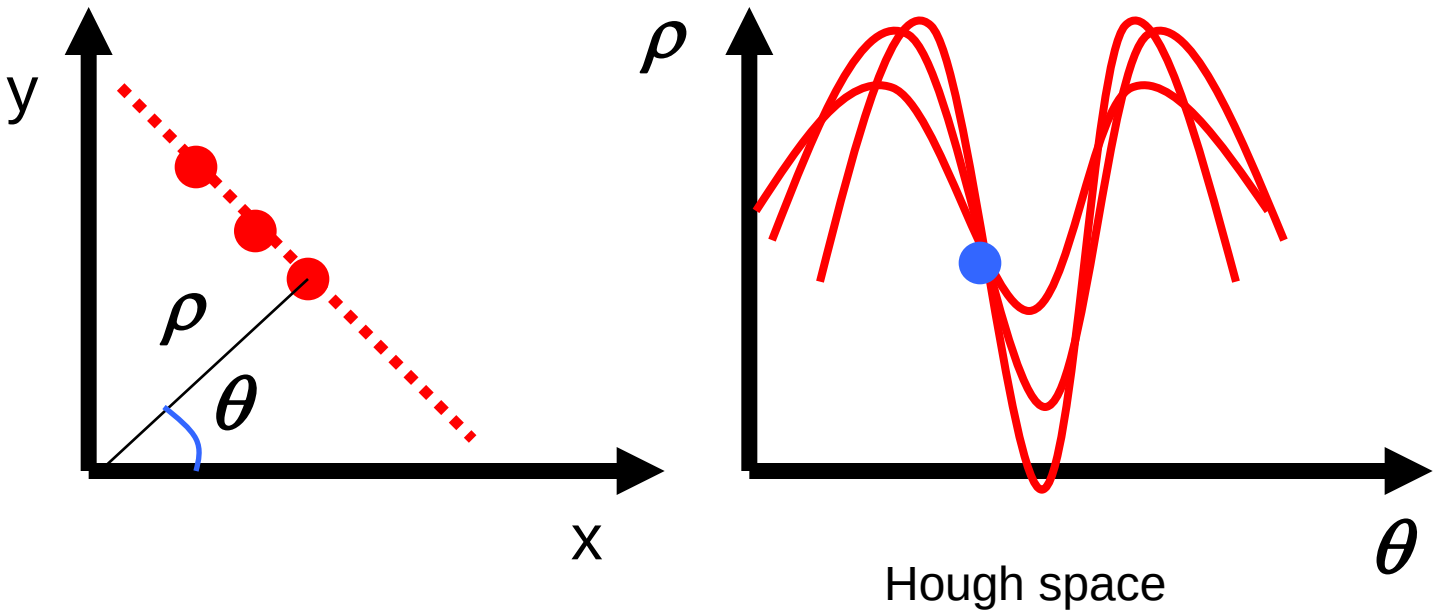
Slide from S. Savarese

Hough transform

P.V.C. Hough, *Machine Analysis of Bubble Chamber Pictures,* Proc. Int. Conf. High Energy Accelerators and Instrumentation, 1959

Issue : parameter space [m,b] is unbounded…

Use a polar representation for the parameter space

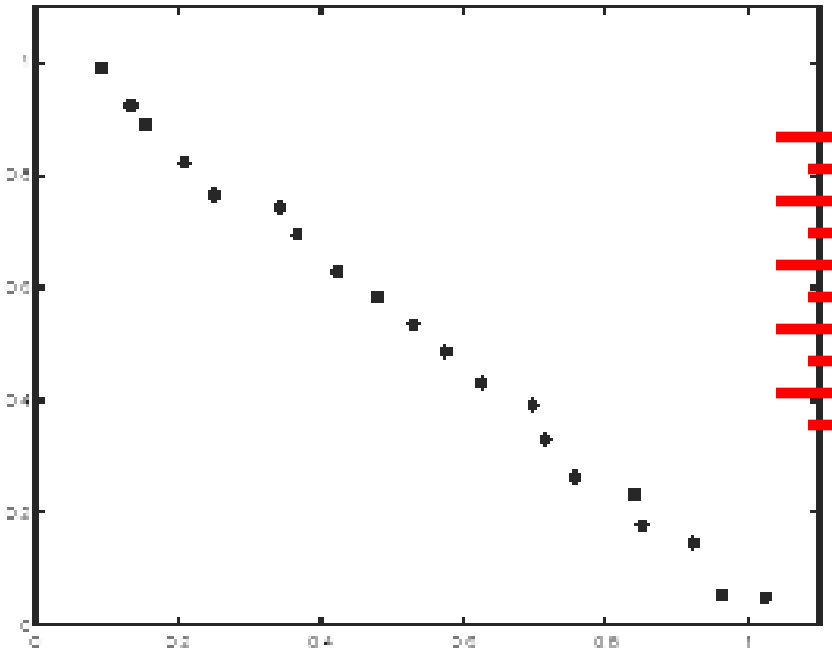


$$x\cos\theta + y\sin\theta = \rho$$

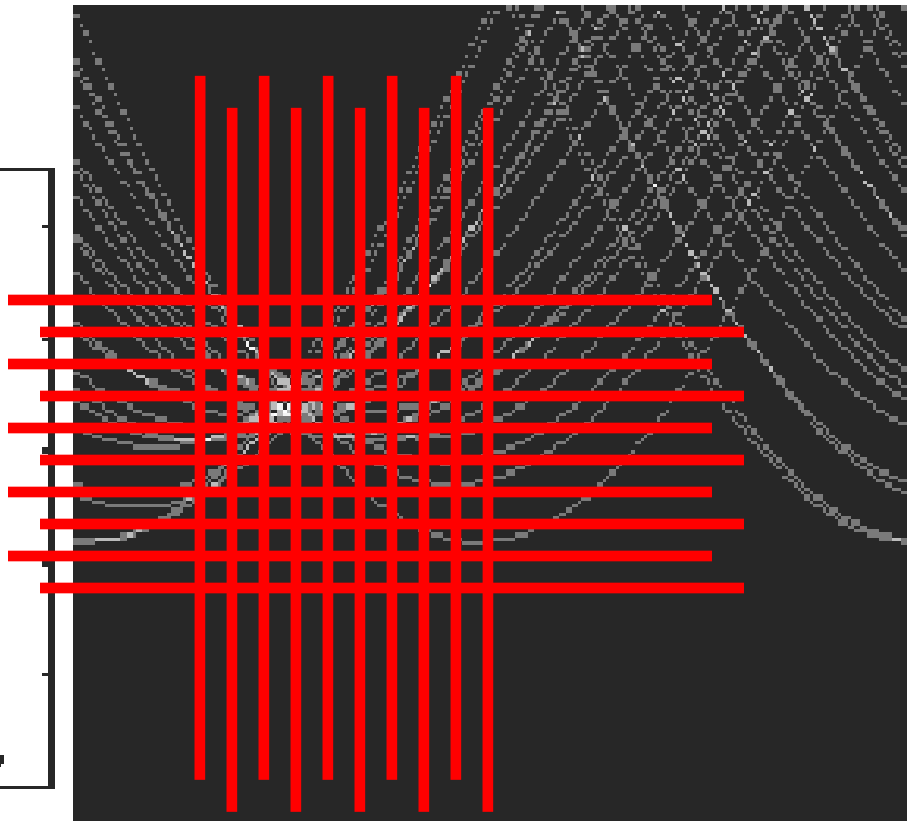
Slide from S. Savarese

Hough transform - experiments

Noisy data



features



votes

Issue: Grid size needs to be adjusted…

Slide from S. Savarese

Hough transform conclusions

Good

- Robust to outliers: each point votes separately
- Fairly efficient (often faster than trying all sets of parameters)
- Provides multiple good fits

Bad

- Some sensitivity to noise
- Bin size trades off between noise tolerance, precision, and speed/memory
  - Can be hard to find sweet spot
- Not suitable for more than a few parameters
  - grid size grows exponentially

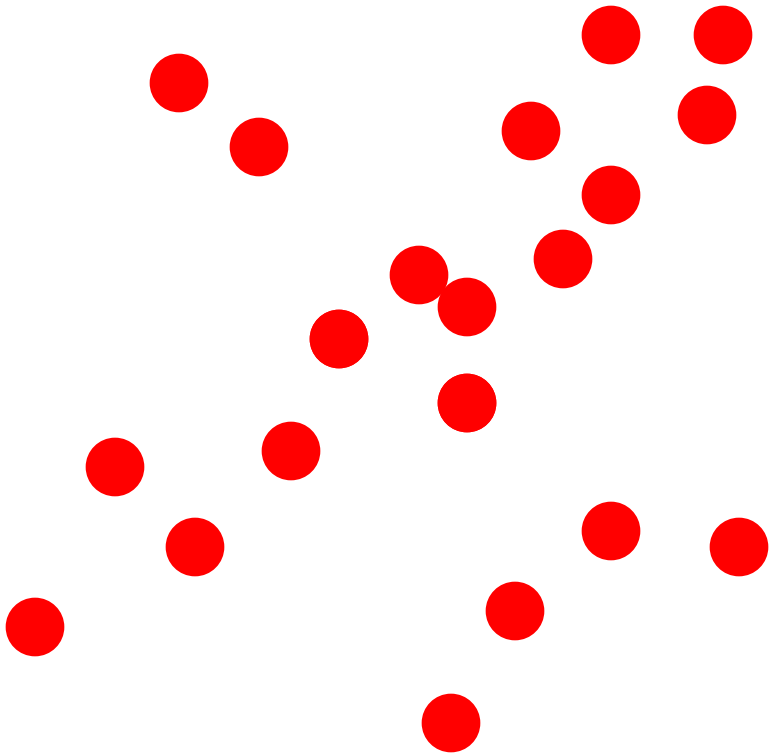
Common applications

- Line fitting (also circles, ellipses, etc.)
- Object instance recognition (parameters are affine transform)
- Object category recognition (parameters are position/scale)

RANSAC

(RANdom SAmple Consensus) :

Fischler & Bolles in ‘81.



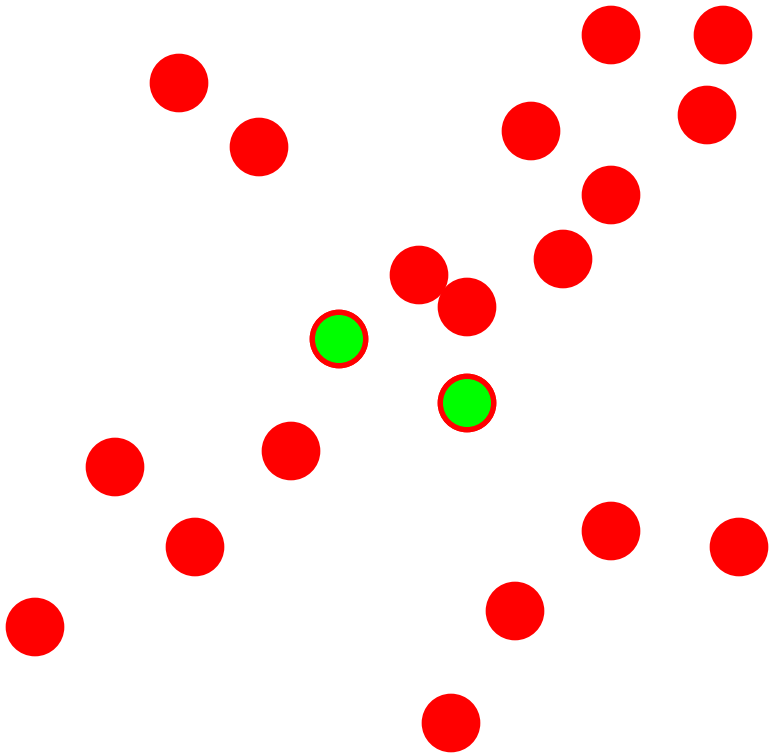
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

RANSAC

Line fitting example



Algorithm:

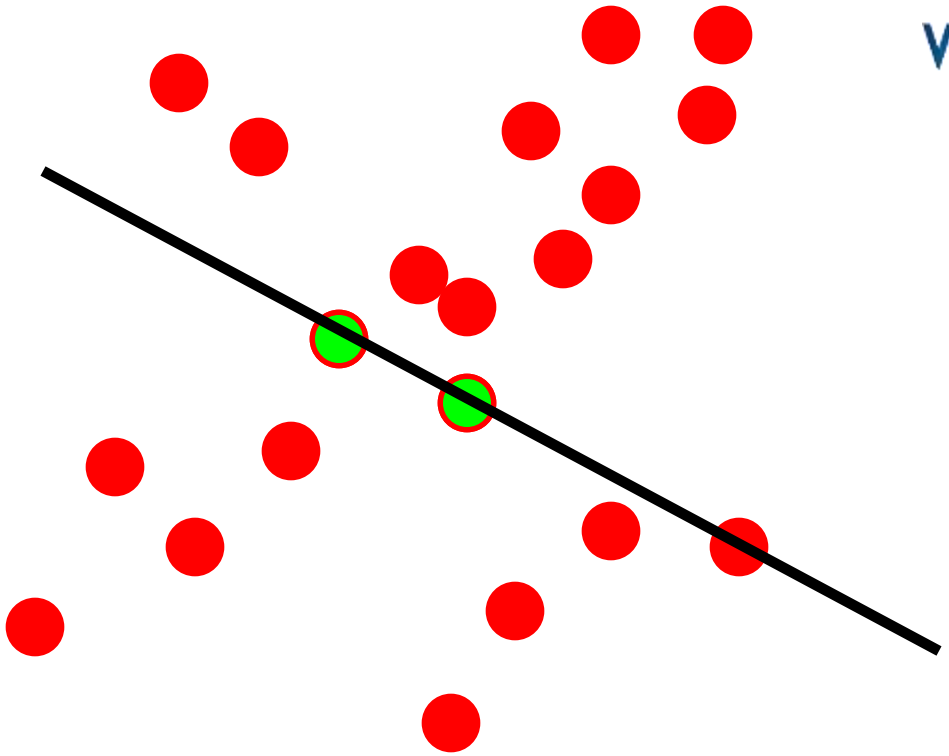
1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Illustration by Savarese

RANSAC

Line fitting example



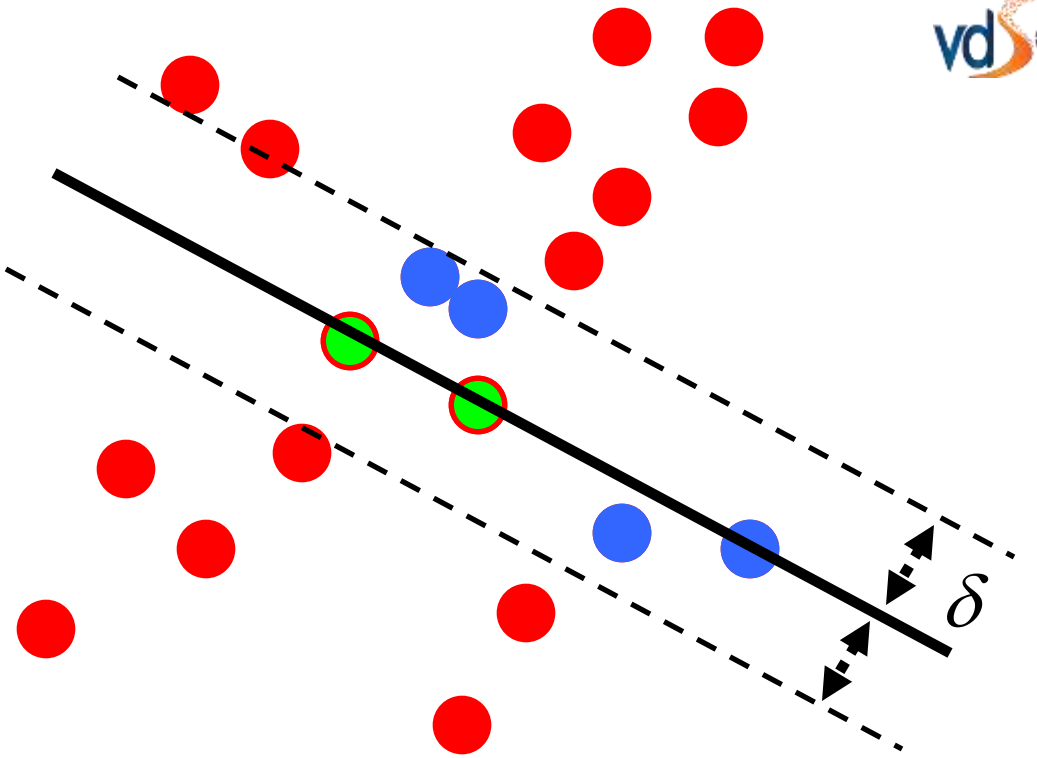
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

RANSAC

Line fitting example



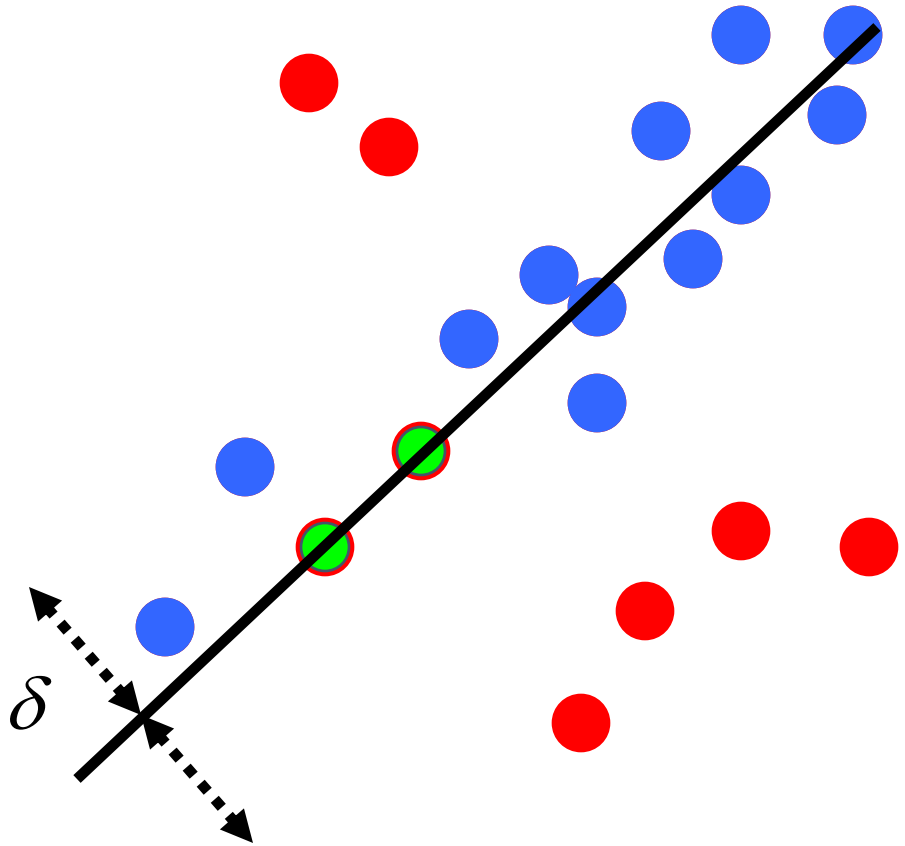
$$N_I = 6$$

Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

RANSAC



$$N_I = 14$$

Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Choosing the parameters

- Initial number of points *s*
  - Typically minimum number needed to fit the model
- Distance threshold *t*
  - Choose *t* so probability for inlier is *p* (e.g. 0.95)
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$
- Number of samples *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)

$$\left(1-\left(1-e\right)^s\right)^N = 1-p$$

$$N = \log\left(1-p\right) / \log\left(1-\left(1-e\right)^s\right)$$

	proportion of outliers *e*						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

Source: M. Pollefeys

RANSAC conclusions

Good

- Robust to outliers
- Applicable for larger number of parameters than Hough transform
- Parameters are easier to choose than Hough transform

Bad

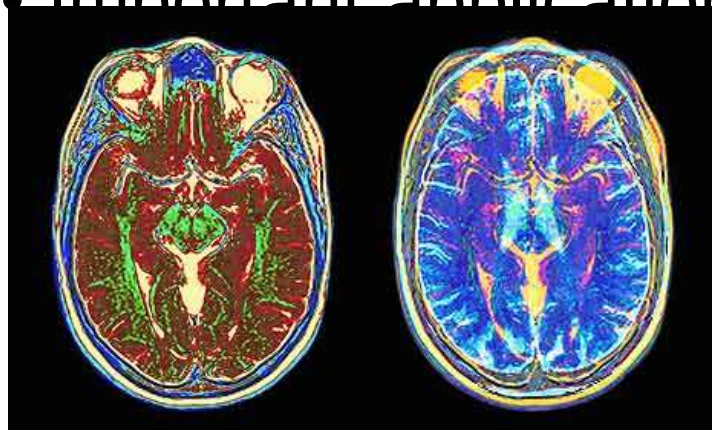
- Computational time grows quickly with fraction of outliers and number of parameters
- Not good for getting multiple fits

Common applications

- Computing a homography (e.g., image stitching)
- Estimating fundamental matrix (relating two views)

What if you want to align but have no prior matched pairs?

- Hough transform and RANSAC not applicable
- Important applications



Medical imaging: match brain scans or contours

Robotics: match point clouds

Slide from Derek Hoiem

THỊ GIÁC MÁY TÍNH

AI Academy Vietnam

CẢM ƠN!