

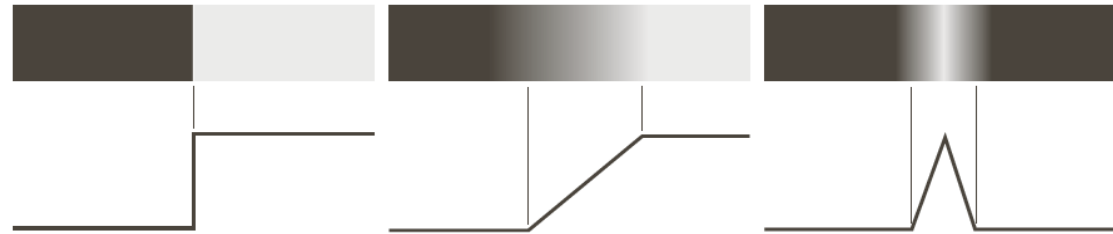


UNIVERSITY OF  
SOUTH CAROLINA

# CSCE 590 INTRODUCTION TO IMAGE PROCESSING

## Edge Detection

# Edge Models



a b c

**FIGURE 10.8** From left to right, models (ideal representations) of a step, a ramp, and a roof edge, and their corresponding intensity profiles.

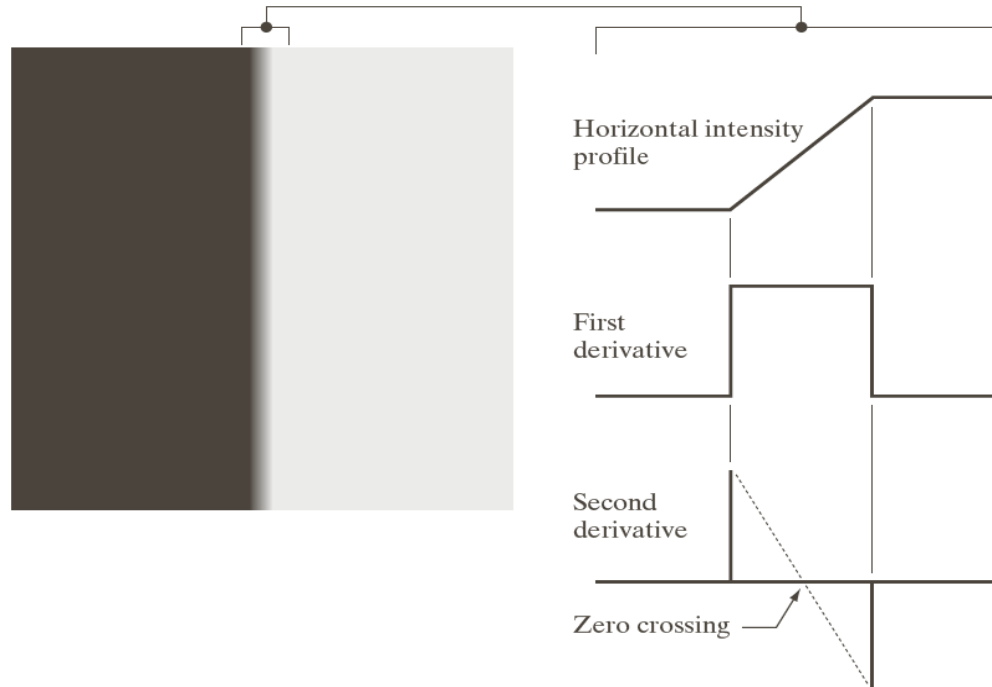


**FIGURE 10.9** A  $1508 \times 1970$  image showing (zoomed) actual ramp (bottom, left), step (top, right), and roof edge profiles. The profiles are from dark to light, in the areas indicated by the short line segments shown in the small circles. The ramp and “step” profiles span 9 pixels and 2 pixels, respectively. The base of the roof edge is 3 pixels. (Original image courtesy of Dr. David R. Pickens, Vanderbilt University.)



# Derivatives – Idea Cases

Observation:



a b

**FIGURE 10.10**

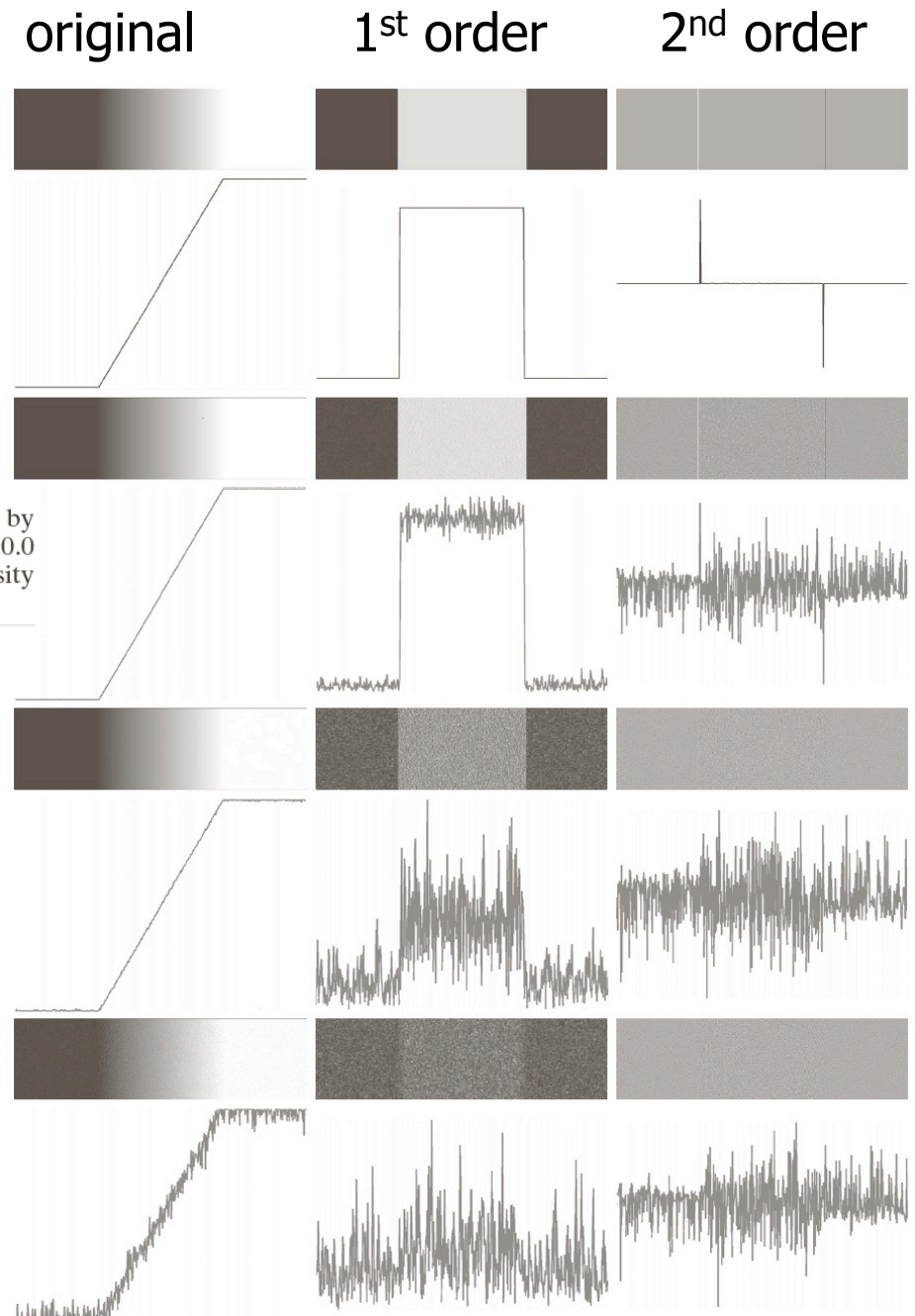
(a) Two regions of constant intensity separated by an ideal vertical ramp edge.  
(b) Detail near the edge, showing a horizontal intensity profile, together with its first and second derivatives.

Conclusion

- The magnitude of first derivative -- the presence of an edge
- The sign of second derivative -- the intensity transition direction
  - double edge
  - zero crossing – the center of ramp



## But, In Practice ...



### 1. Image Smoothing

### 2. Detecting edge points

### 3. Edge localization



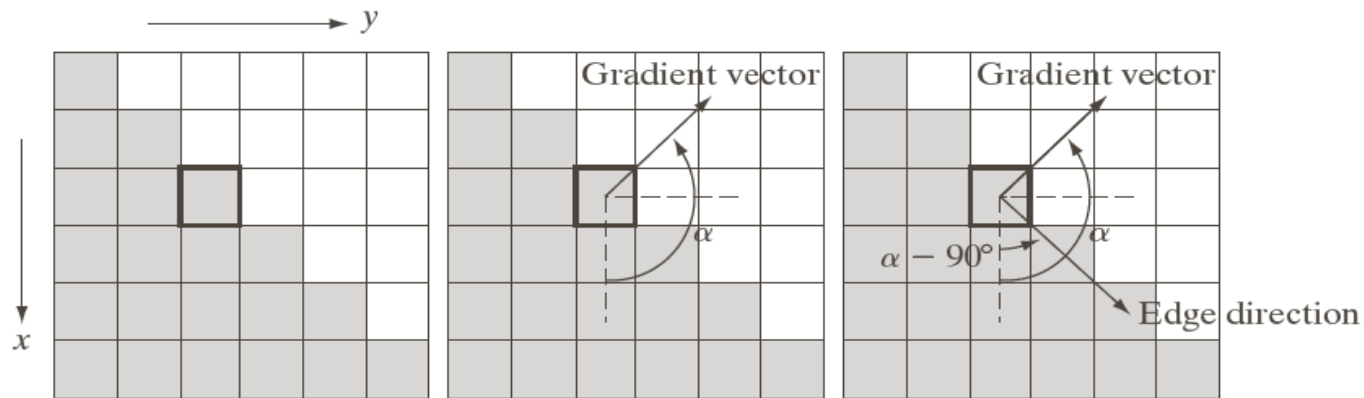
# Basic Edge Detection

First-order derivative:

Gradient  $\nabla f(x, y) = \text{grad}(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \partial f / \partial x \\ \partial f / \partial y \end{bmatrix}$

Edge strength  $M(x, y) = \sqrt{g_x^2 + g_y^2}$   
 $\approx |g_x| + |g_y|$

Edge direction  $\alpha(x, y) = \tan^{-1} \left[ \frac{g_y}{g_x} \right]$



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.



# Masks for Calculating the Gradient (2x2)

Gradient in vertical/horizontal direction

-1
1

-1	1
----	---

Gradient in diagonal direction

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

-1	0
0	1

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

0	-1
1	0



# Masks for Calculating the Gradient (3x3)

Gradient in vertical/horizontal

-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

Gradient in diagonal

0	1	1	-1	-1	0
-1	0	1	-1	0	1
-1	-1	0	0	1	1

Prewitt

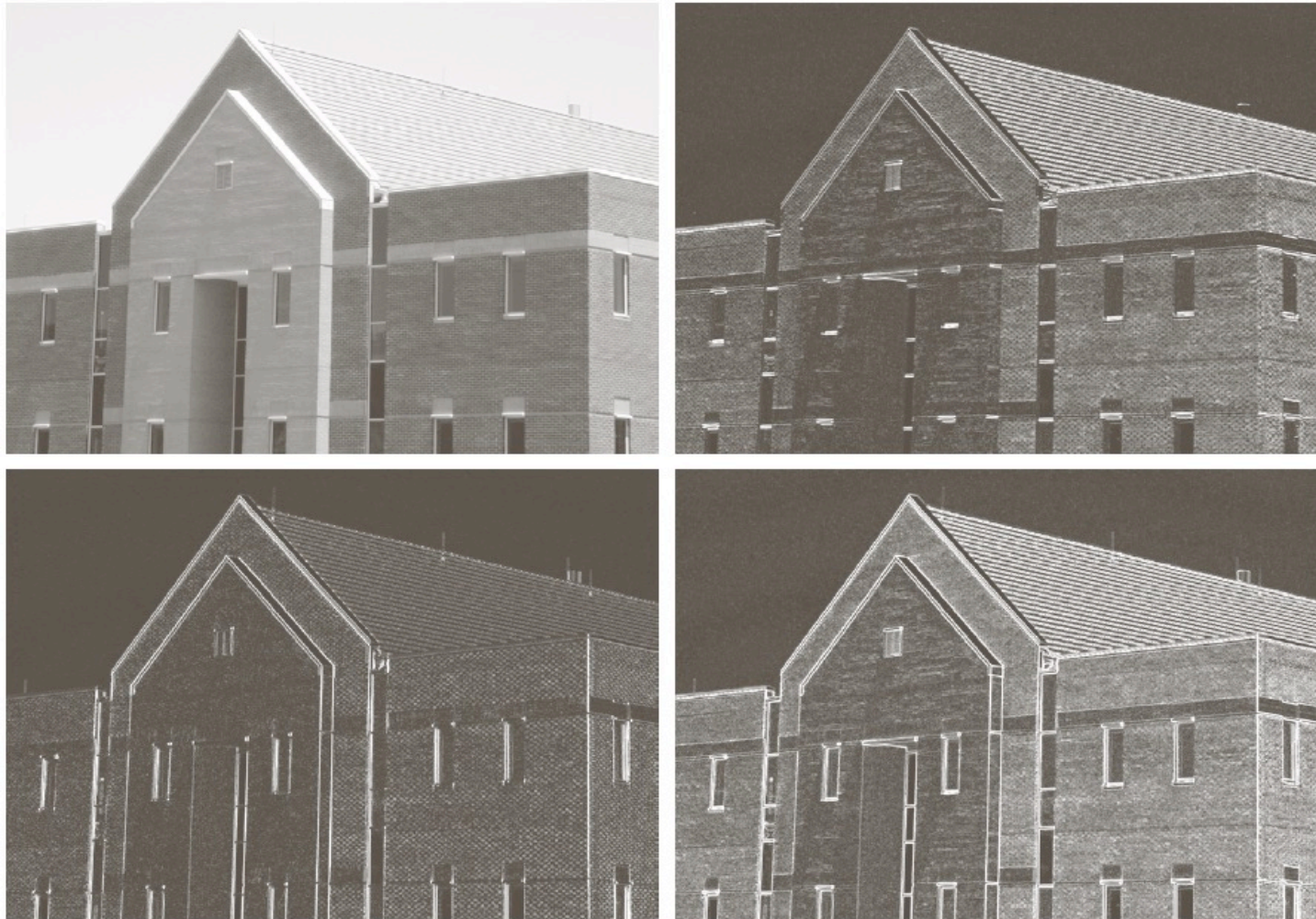
0	1	2	-2	-1	0
-1	0	1	-1	0	1
-2	-1	0	0	1	2

Sobel

Sobel operator performs edge detection and smoothing simultaneously.



# An Example



a b  
c d

**FIGURE 10.16**  
(a) Original image of size  $834 \times 1114$  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,  $|g_x| + |g_y|$ .





## An Example – Cont.



**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.

Angle information is employed in Canny edge detector and other feature representation, such as Histogram of Orientation (HOG).



## An Example – Cont.



a	b
c	d

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



## An Example – Cont.

45 degree



-45 degree



a b

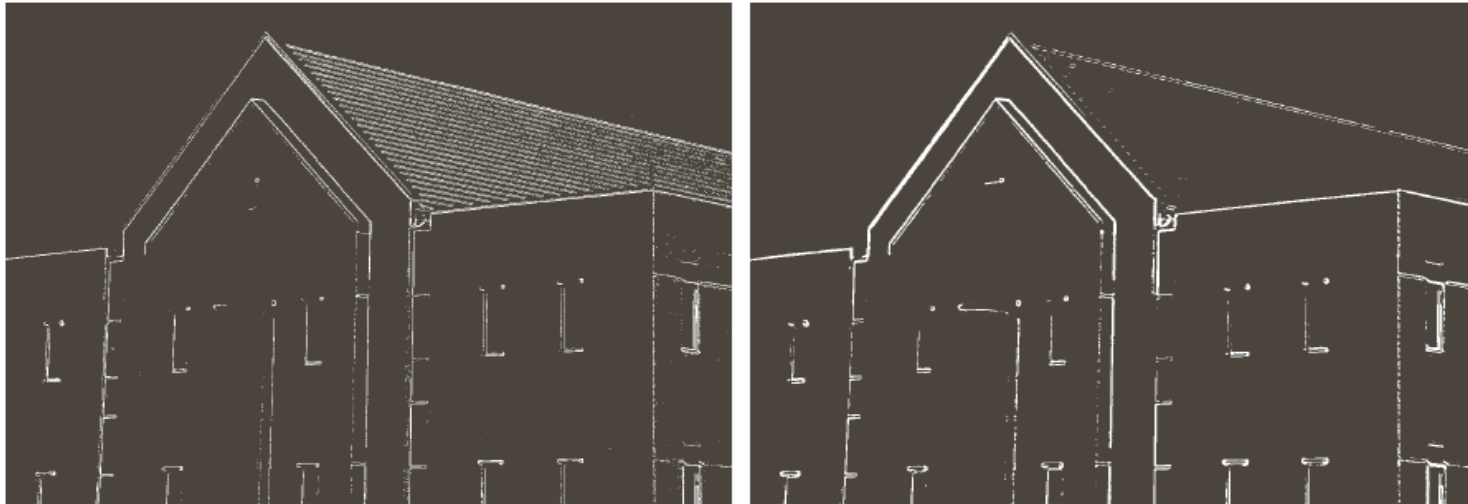
**FIGURE 10.19**  
Diagonal edge detection.  
(a) Result of using the mask in Fig. 10.15(c).  
(b) Result of using the mask in Fig. 10.15(d). The input image in both cases was Fig. 10.18(a).

Edge detection in diagonal directions.



# Combining the Gradient with Thresholding

Smoothed input



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.



# Brief Review on Simple Edge Detectors


---

- First-order derivative
  - Roberts (2x2)
  - Prewitt (3x3)
  - Sobel (3x3, smooth + difference)
  - Thicker edge
  - One operator for one edge direction
- Second-order derivative
  - Laplacian (3x3)
  - Double edge
  - Zero-crossing
- Common issues:
  - Sensitive to image noise
  - Cannot deal with the scale change of the image



# Advanced Edge Detection Techniques

---

- Deal with image noise
- Exploit the properties of image
-  Work much better for real images
  
- Advanced edge detectors:
  - Laplacian of Gaussian (LoG)
  - Difference of Gaussian (DoG)
  - Canny




## Marr-Hildreth Detector (LoG)

### Observations:

- Intensity changes are dependent on the image scale
- A sudden intensity change (step) causes a peak/trough in the 1<sup>st</sup> order derivative and a zero-crossing in the 2<sup>nd</sup> order derivative
- The 2<sup>nd</sup> order derivative is especially sensitive to noise

➔ Smooth the image using a Gaussian filter first before applying the Laplacian

$$\text{Gaussian} \longrightarrow G(x, y) = e^{-\frac{x^2 + y^2}{2\sigma^2}}$$

$$g(x, y) = \nabla^2 [G(x, y) \otimes f(x, y)] = \nabla^2 G(x, y) \otimes f(x, y)$$


**Laplacian of a Gaussian (LoG)**



## Marr-Hildreth Detector (LoG)

**Laplacian of a Gaussian (LoG):**  $\nabla^2 G(x, y)$

Gaussian  $\longrightarrow G(x, y) = e^{-\frac{x^2 + y^2}{2\sigma^2}}$

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

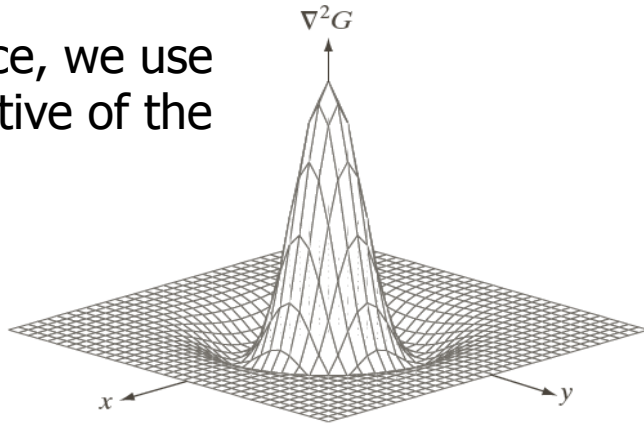
- Varying  $\sigma$  values for scale changes
- Rotation invariant in edge detection



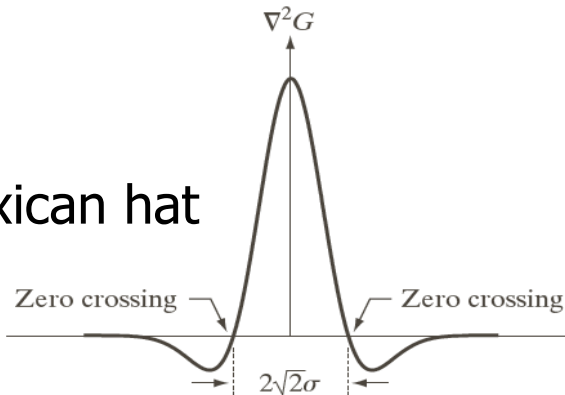


# LoG Cont'd

In practice, we use the negative of the mask



Mexican hat



a b  
c d

**FIGURE 10.21**

(a) Three-dimensional plot of the *negative* of the LoG. (b) Negative of the LoG displayed as an image. (c) Cross section of (a) showing zero crossings. (d)  $5 \times 5$  mask approximation to the shape in (a). The negative of this mask would be used in practice.

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

→ Sum to zero!

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



# LoG Filtering

---

$$\begin{aligned}g(x, y) &= \nabla^2 G(x, y) \otimes f(x, y) \\ &= \nabla^2 [G(x, y) \otimes f(x, y)]\end{aligned}$$

1. Filter the input image with an  $n \times n$  Gaussian filter.
  2. Compute the Laplacian of the intermediate image resulting from Step 1.
  3. Find the zero-crossings of the image from Step 2.
    - opposite signs of the neighbors
    - the difference should be significant
- Note:
    - Window size  $n \geq 6\sigma$  and  $n$  is an odd number

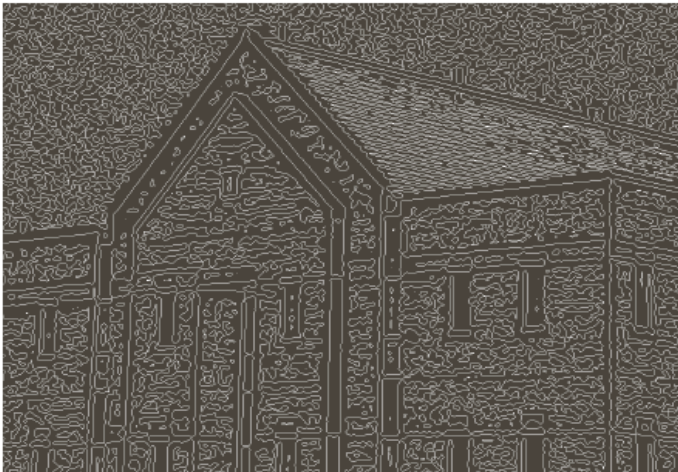


# An Example – Edges are 1 Pixel Thick

Original

LoG filtering

LoG filtering with  $T=0$



Zero-crossing with  $T=0$

Zero-crossing with  $T=4\% \max$

In practice, run LoG many times with different sigmas and keep the edges that are common for all sigmas



## Approximate LoG by DoG

LoG needs multiple filters for scale variations  
Difference of Gaussian (DoG)

$$DOG(x, y) = \frac{1}{2\pi\sigma_1^2} e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2} e^{-\frac{x^2+y^2}{2\sigma_2^2}}, \quad \sigma_1 > \sigma_2$$



a b

**FIGURE 10.23**  
(a) Negatives of the LoG (solid) and DoG (dotted) profiles using a standard deviation ratio of 1.75:1.  
(b) Profiles obtained using a ratio of 1.6:1.




# Canny Edge Detector

---

- The general goal:
  1. Low error rate: all edges should be found and there should be no false response
  2. Edge points should be well localized: the edges located must be as close as possible to the true edges
  3. Single edge point response: the detector should return only one point for each true edge point
- Canny edge detector: expressing these three criteria mathematically and then find optimal solutions




# Canny Edge Detector

- Gaussian smoothing + 1<sup>st</sup> order derivative
-  optimal step edge detector

$$f_s(x, y) = G(x, y) \otimes f(x, y)$$

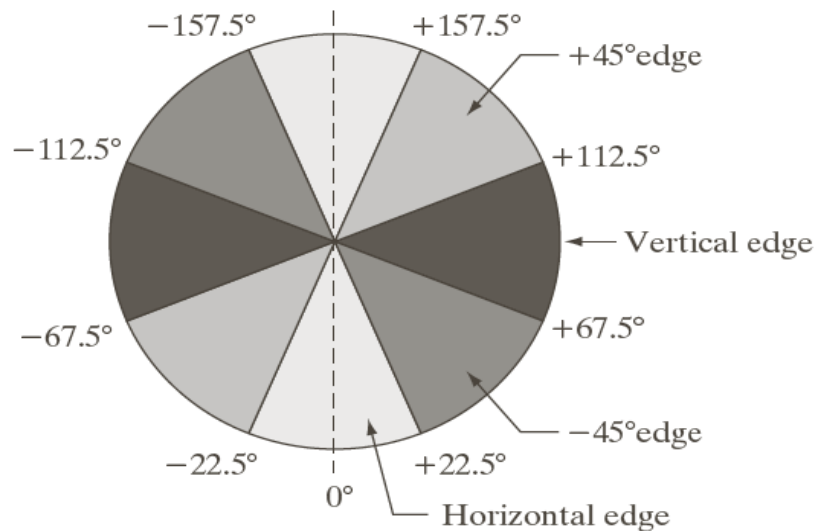
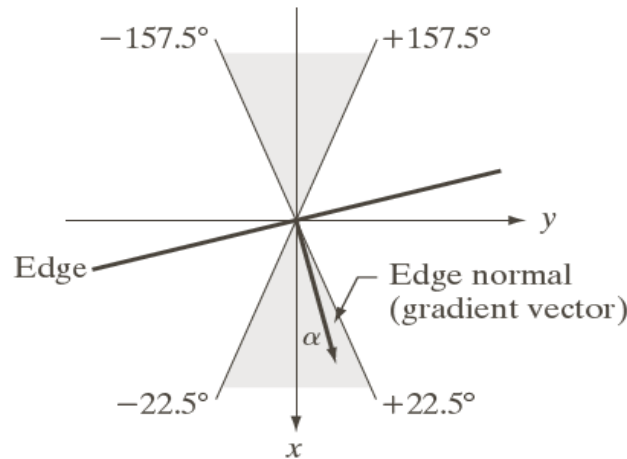
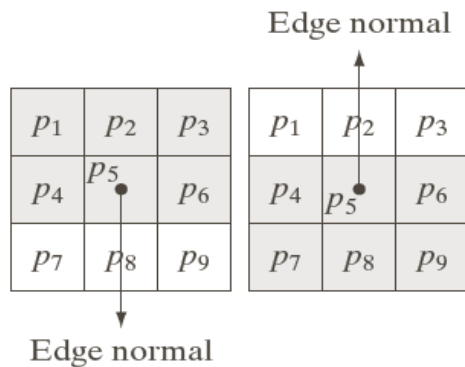
$$g_x = \partial f_s / \partial x \quad g_y = \partial f_s / \partial y$$

$$M(x, y) = \sqrt{g_x^2 + g_y^2} \quad \alpha(x, y) = \tan^{-1}(g_y / g_x)$$

**Thick edge**  **Single edge**  
Nonmaxima suppression



# Quantize the Edge Direction



a b  
c

**FIGURE 10.24** (a) Two possible orientations of a horizontal edge (in gray) in a  $3 \times 3$  neighborhood. (b) Range of values (in gray) of  $\alpha$ , the direction angle of the *edge normal*, for a horizontal edge. (c) The angle ranges of the edge normals for the four types of edge directions in a  $3 \times 3$  neighborhood. Each edge direction has two ranges, shown in corresponding shades of gray.



# Canny Detector -- Algorithm

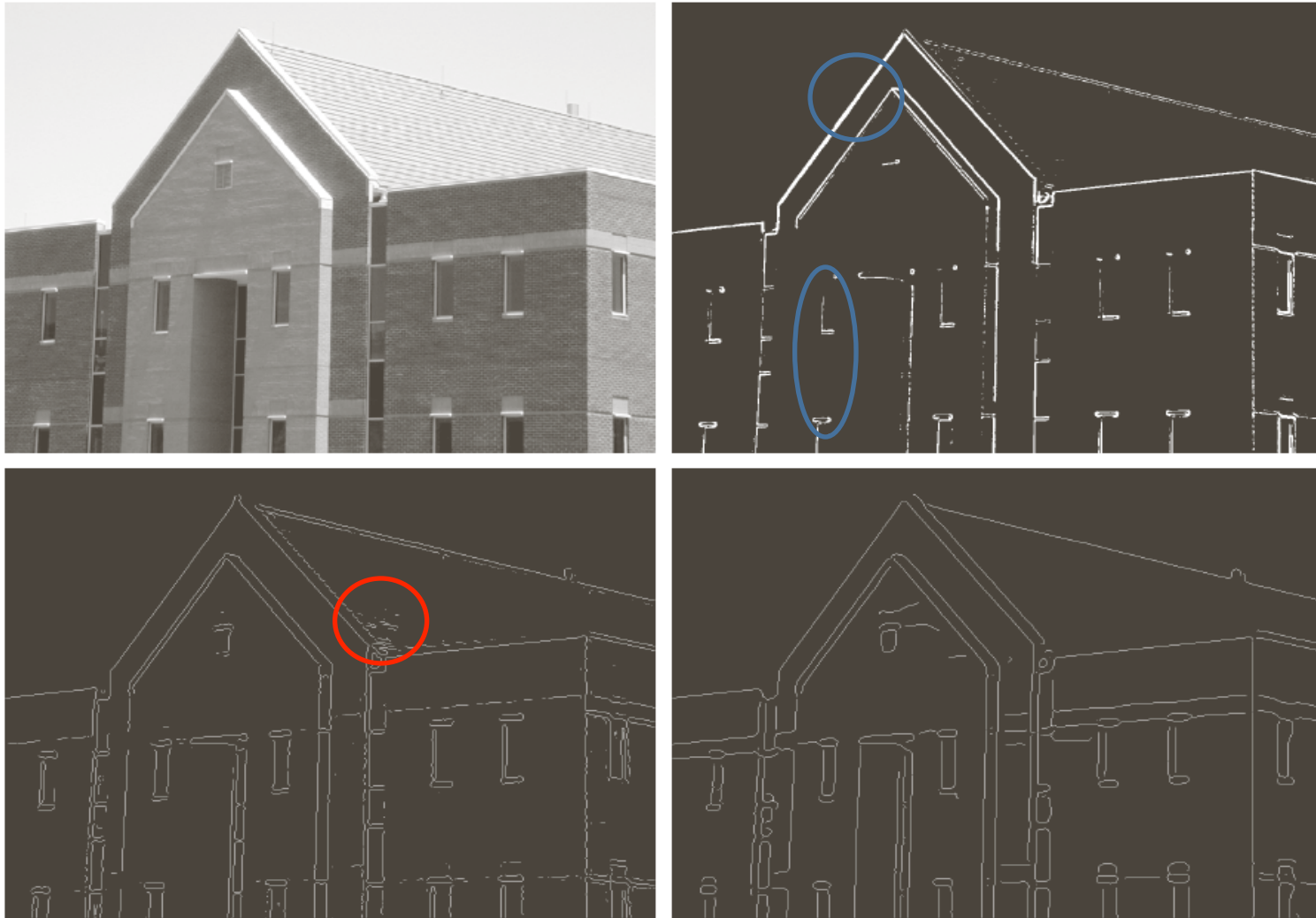
---

1. Smooth the input image with a Gaussian filter
2. Compute the gradient magnitude and angle images
3. Apply nonmaximum suppression on the gradient magnitude image
  1. At  $(x,y)$ , find the quantized edge normal  $d_k$
  2. If the value  $M(x,y)$  is less than at least one of its two neighbors along  $d_k$ , let  $g_N(x,y)=0$ ; otherwise  $g_N(x,y)=M(x,y)$
4. Reduce false edge: double thresholding and connectivity analysis to detect and link edges
  1. High-threshold  $\rightarrow$  strong edge pixels  $\rightarrow$  valid edge pixels
  2. Low-threshold  $\rightarrow$  weak edge pixels  $\rightarrow$  valid only when connected to strong edge pixels





# An Example

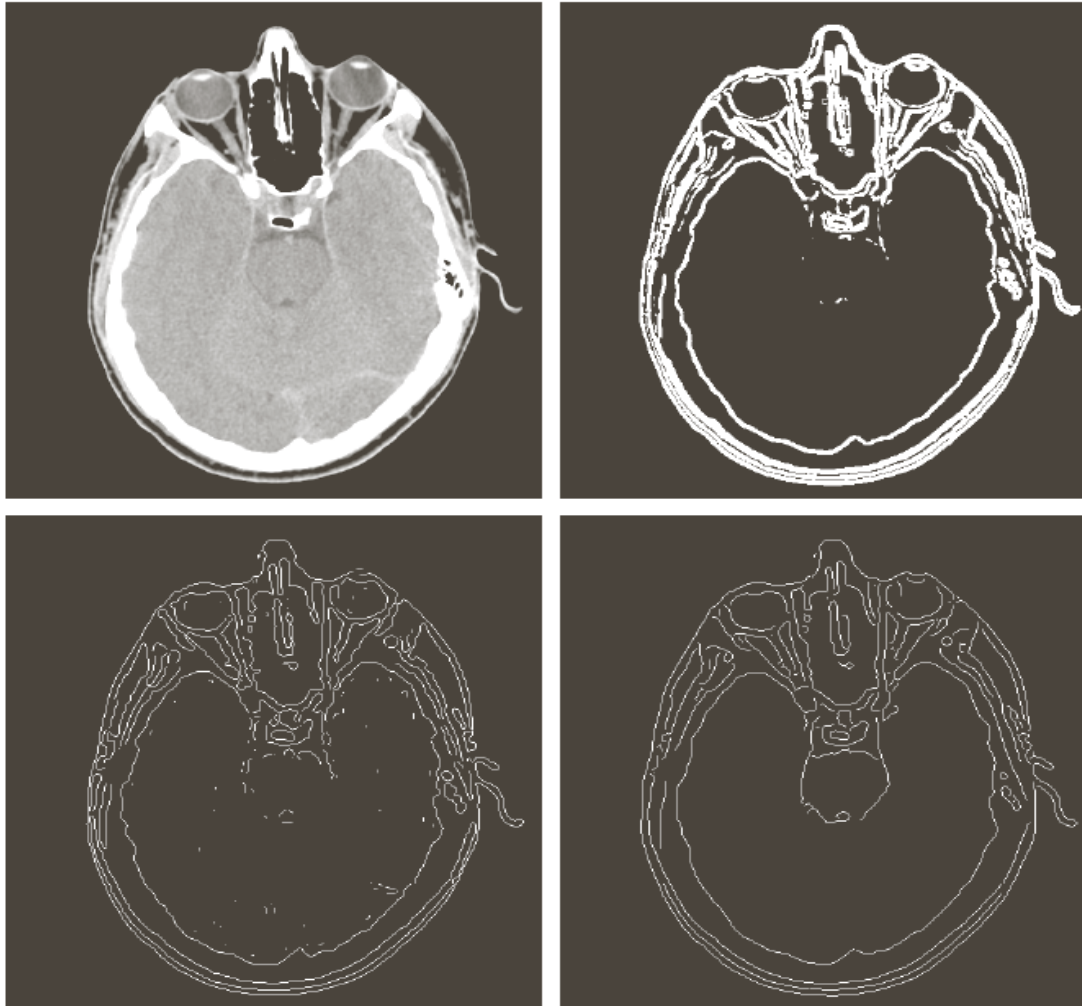


a	b
c	d

**FIGURE 10.25**  
(a) Original image of size  $834 \times 1114$  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.



## An Example



a	b
c	d

**FIGURE 10.26**

(a) Original head CT image of size  $512 \times 512$  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.)



# Edge Linking and Boundary Detection

---

- All edge detection algorithms can only detect fragments of boundaries, due to image noise, non-uniform illuminations, or other effects
- Edge linking: link edges into longer meaningful edges or a full region boundaries
- Three classic methods:
  - Local processing
  - Regional Processing (read textbook page 728-732)
  - Hough transform



# Edge Linking – Local Processing

---

- Link the edge points with similar properties:
  - Strength
  - Direction
- Two edge pixels are linked if

$$| M(s, t) - M(x, y) | \leq E$$

$$| \alpha(s, t) - \alpha(x, y) | \leq A$$



# A Local Processing Example

Objective: find rectangular region for license plate recognition.



**FIGURE 10.27** (a) A  $534 \times 566$  image of the rear of a vehicle. (b) Gradient magnitude image. (c) Horizontally connected edge pixels. (d) Vertically connected edge pixels. (e) The logical OR of the two preceding images. (f) Final result obtained using morphological thinning. (Original image courtesy of Perceptics Corporation.)



# Global Processing

---

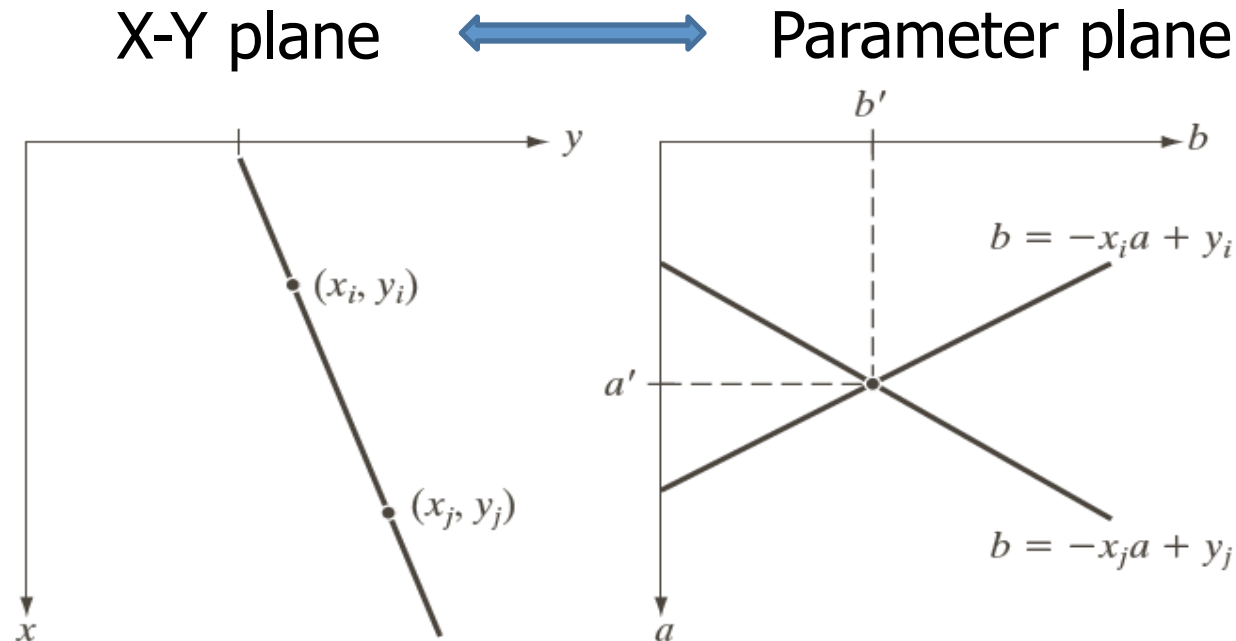
- A basic idea: Given  $n$  points
  1. Find  $n(n-1)/2$  lines between each pair of points
  2. Find all subset of points that are close to particular lines. This needs  $n(n-1)n/2$  comparisons
- This is computationally expensive!

 Hough transform



# Hough Transform

A line in x-y plane is a point in the parameter plane.  
A point in x-y plane is a line in the parameter plane.



a b

FIGURE 10.31  
(a)  $xy$ -plane.  
(b) Parameter space.

**Problem of slope-intercept form: the slope  $a$  approaches infinity for vertical lines**

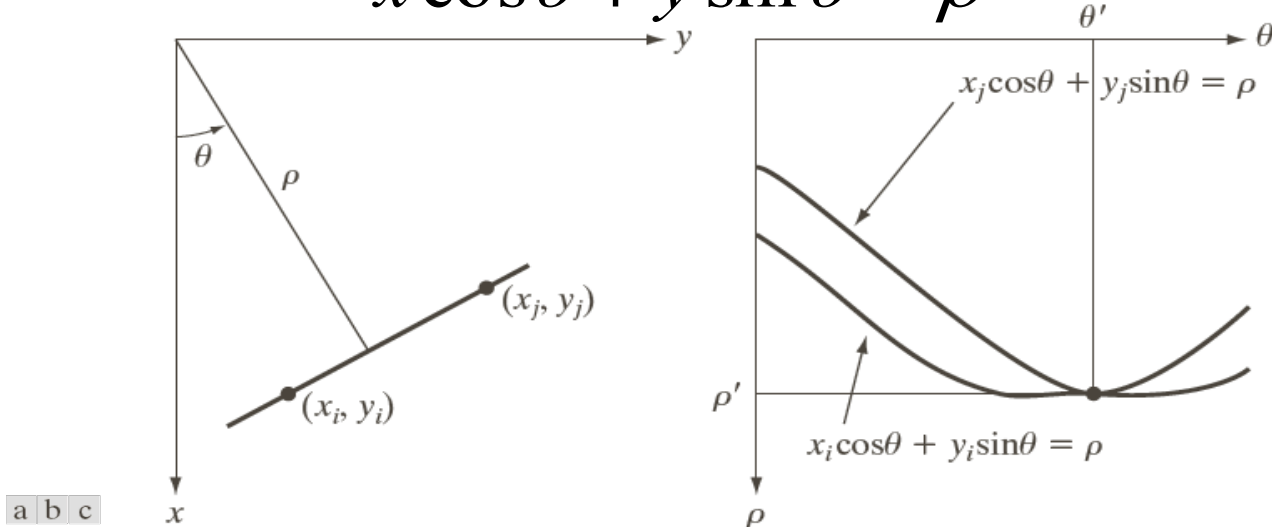


# Hough Transform

A line in x-y plane is a point in the parameter plane.

A point in x-y plane is a sinusoidal in the parameter plane (polar space).

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

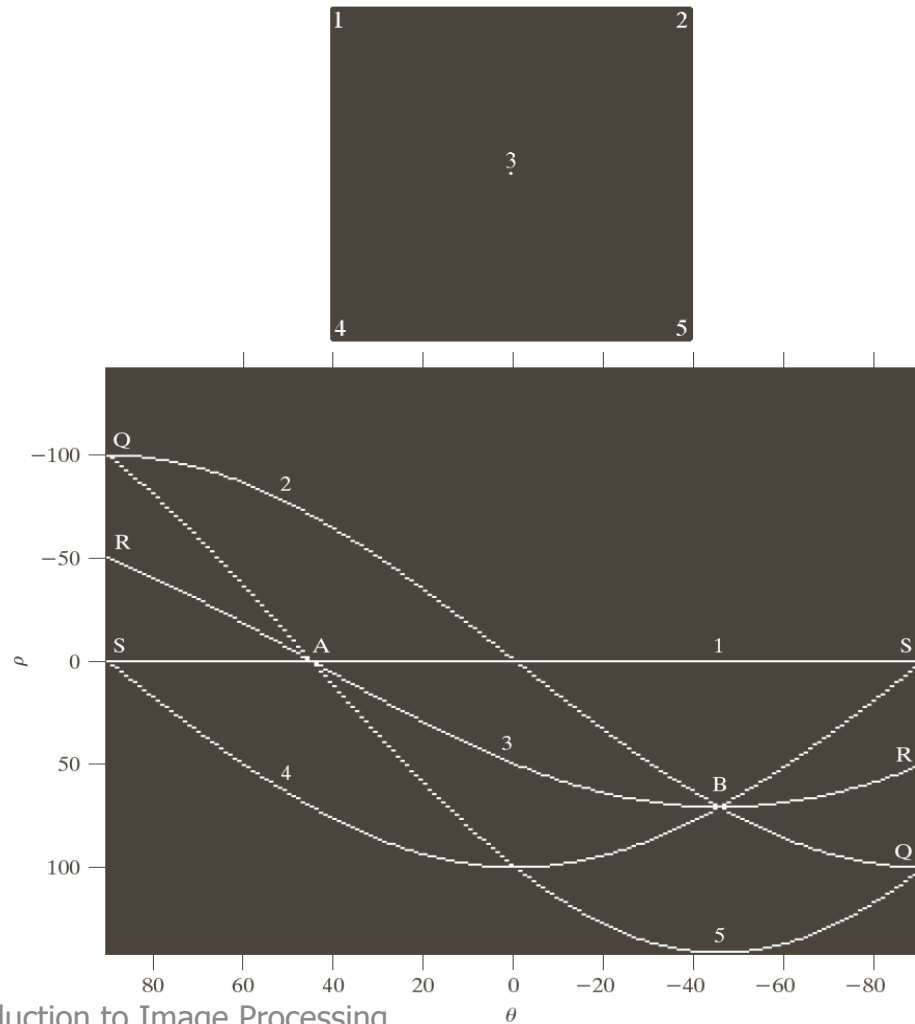


**FIGURE 10.32** (a)  $(\rho, \theta)$  parameterization of line in the  $xy$ -plane. (b) Sinusoidal curves in the  $\rho\theta$ -plane; the point of intersection  $(\rho', \theta')$  corresponds to the line passing through points  $(x_i, y_i)$  and  $(x_j, y_j)$  in the  $xy$ -plane. (c) Division of the  $\rho\theta$ -plane into accumulator cells.





# A Toy Example



a  
b

**FIGURE 10.33**

(a) Image of size  $101 \times 101$  pixels, containing five points.  
(b) Corresponding parameter space. (The points in (a) were enlarged to make them easier to see.)



# Hough Transform – Real Example

Find the runway



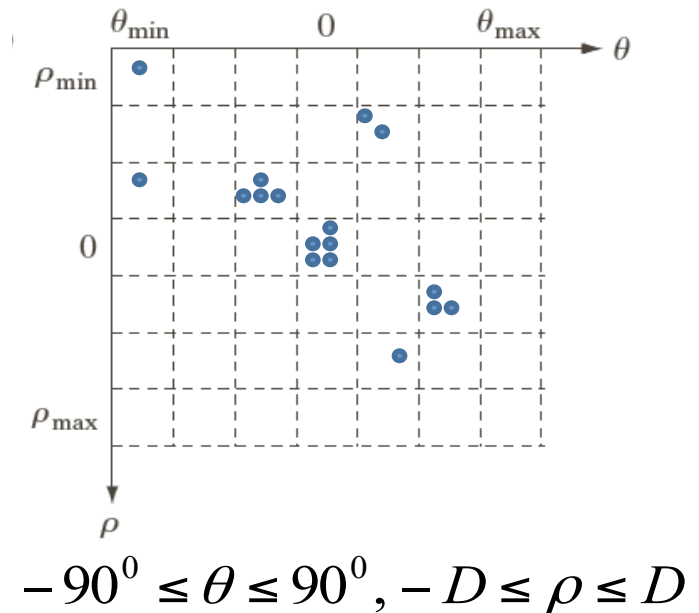
a	b
c	d e

**FIGURE 10.34** (a) A  $502 \times 564$  aerial image of an airport. (b) Edge image obtained using Canny's algorithm. (c) Hough parameter space (the boxes highlight the points associated with long vertical lines). (d) Lines in the image plane corresponding to the points highlighted by the boxes). (e) Lines superimposed on the original image.



# Hough Transform Algorithm

- Obtain a binary edge image using any edge detector
- Specify subdivisions in the  $\rho$ - $\theta$  plane
- For each edge point
  - For each  $\theta$  value in the accumulator cell
  - Update the accumulator cell with the corresponding  $\rho$
- Examine the counts of accumulator cell for high pixel concentrations
- For each chosen cell, link the pixels based on the continuity



# Note

---

- Thresholding is required to get edge pixels and realize edge-based segmentation
- Boundary detection (edge linking) is still a hot research topic in image processing and computer vision
- Incorporate domain knowledge:
  - Psychology rules on the boundary: smooth, convex, symmetry, closed, complete, etc
  - Template shape information: hand, stomach, lip, etc
  - Appearance information: region-based texture, intensity/color, etc.



# Questions?

---

