Chapter-10
Image Segmentation

Elements of Image Processing

Preprocess

Image acquisition, restoration, and enhancement

↓

Intermediate process

Image segmentation and feature extraction

↓

High level process

Image interpretation and recognition

Importance of Image Segmentation

- Image segmentation is used to separate an image into constituent parts based on some image attributes
- Image segmentation is an important step in image analysis

Benefits of Image segmentation

1. Reduces huge amount of unnecessary data
2. Retains only desired parts of data for image analysis
3. Converts bitmap data into better structured data which is easier to interpret

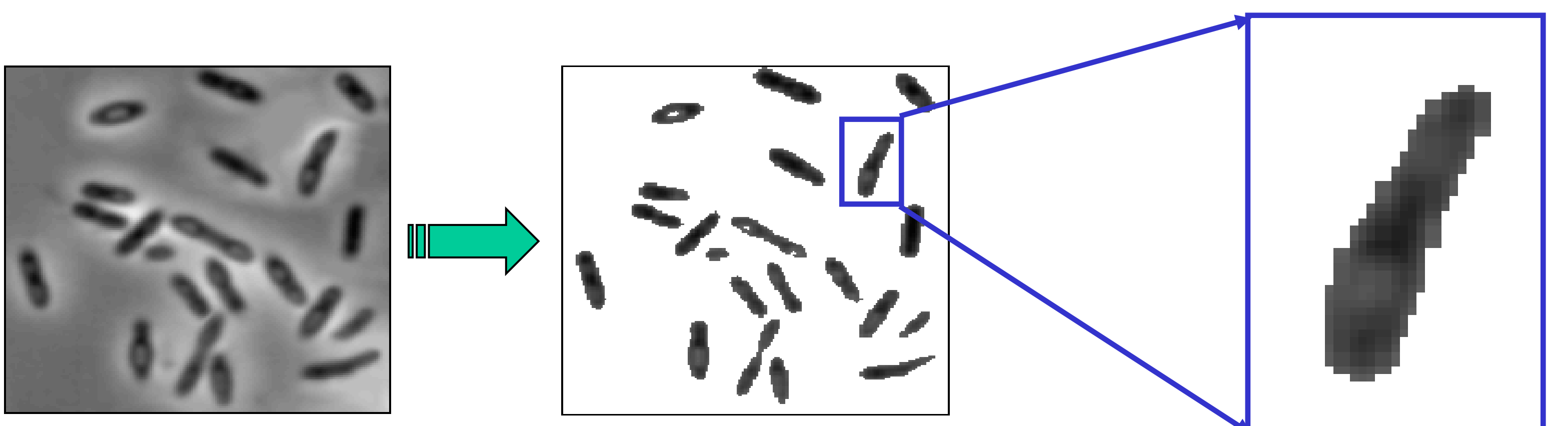
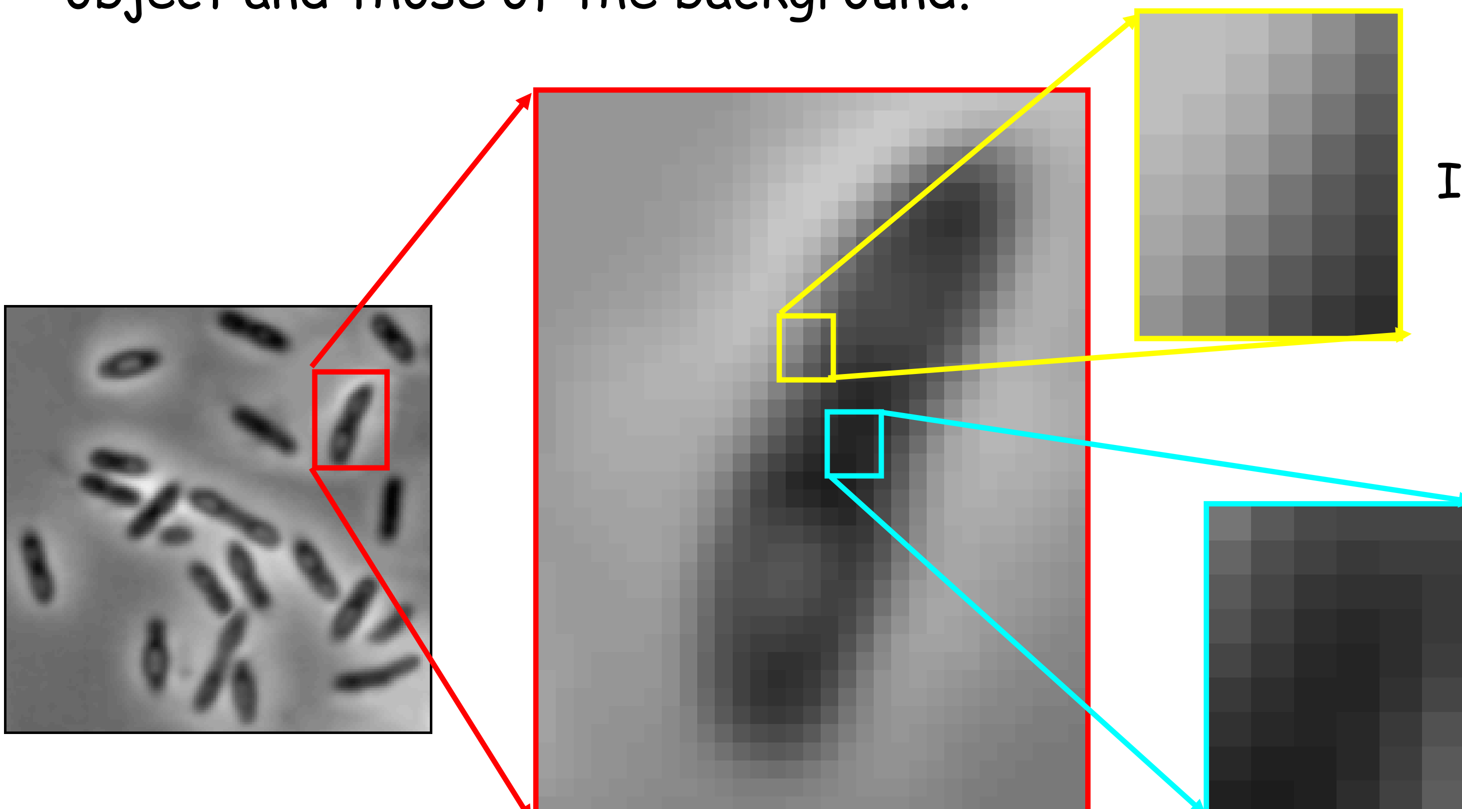


Image Attributes for Image Segmentation

1. Similarity properties of pixels inside the object are used to group pixels into the same set.
2. Discontinuity of pixel properties at the boundary between object and background is used to distinguish between pixels belonging to the object and those of the background.



Discontinuity: Intensity change at boundary

Similarity: Internal pixels share the same intensity

10.1 Fundamentals

- R: Entire spatial region occupied by an image.
- $Q(R_i)$: A logical predicate (or Rule)
- Image segmentation is a process that partitions R into n sub-regions, $R_1, R_2, \ldots, R_n$, such that

(a) $\bigcup_{i=1}^{n} R_i = R.$

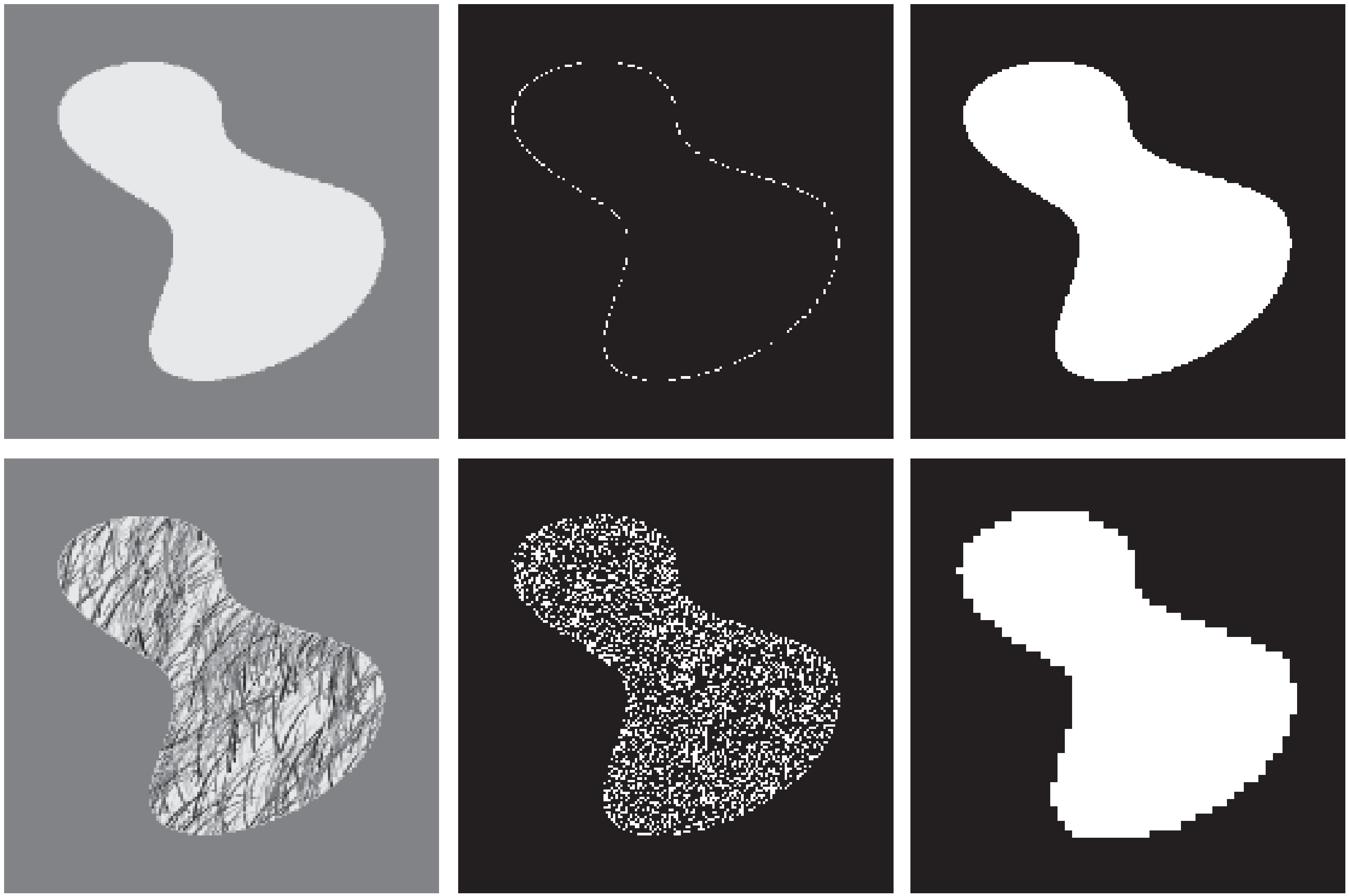
(b) R_i is a connected set. $i = 1, 2, \ldots, n.$

(c) $R_i \cap R_j = \varnothing \quad \text{for } \forall i, j, \quad i \neq j$

(d) $Q(R_i) = \text{TRUE for } i = 1, 2, \ldots, n.$

(e) $Q(R_i \cup R_j) = \text{FALSE for any adjacent regions } R_i \text{ and } R_j.$

Fundamentals



a b c
d e f

FIGURE 10.1
(a) Image of a constant intensity region.
(b) Boundary based on intensity discontinuities.
(c) Result of segmentation.
(d) Image of a texture region.
(e) Result of intensity discontinuity computations (note the large number of small edges).
(f) Result of segmentation based on region properties.

10.2 Background

- Derivatives play an important role in separating/segmenting different regions

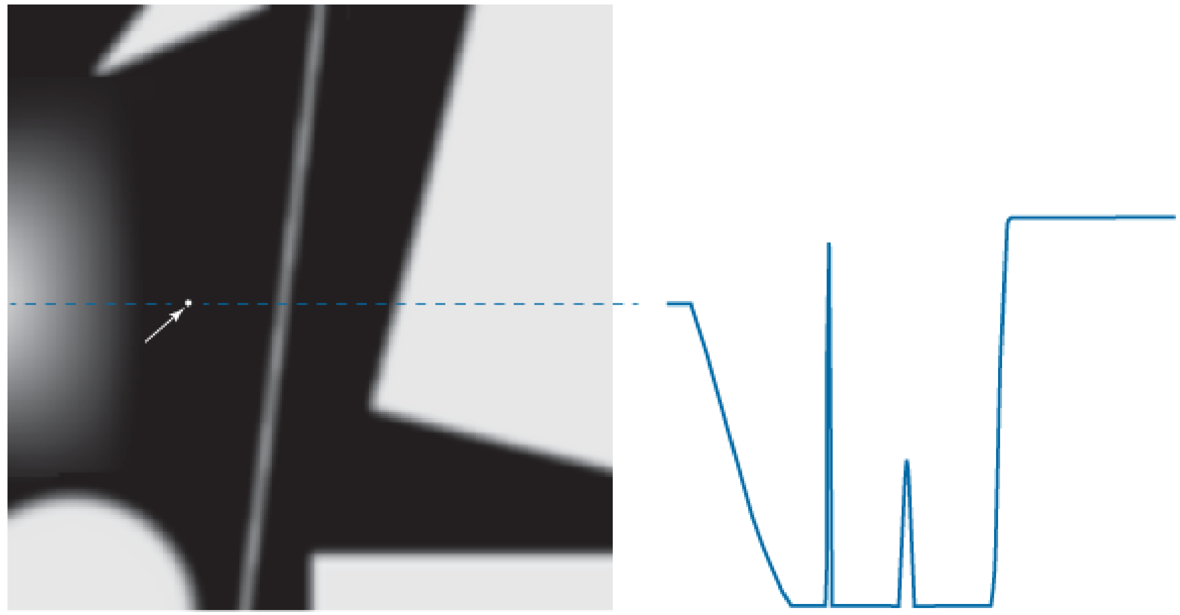
- First-order derivative

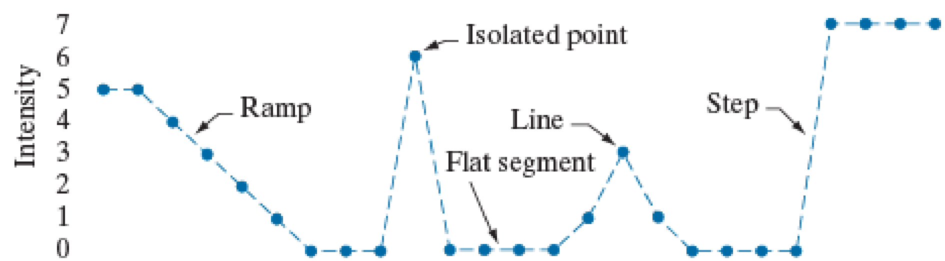
$$\frac{\partial f}{\partial x} = f'(x) = f(x+1) - f(x)$$

- Second-order derivative

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$

Background (contd.)





Intensity values: 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 . .

First derivative: −1 −1 −1 −1 −1 0 0 6 −6 0 0 0 1 2 −2 −1 0 0 0 7 0 0 0

Second derivative: −1 0 0 0 0 1 0 6 −12 6 0 0 1 1 −4 1 1 0 0 7 −7 0 0

Double line effect

a b
c

FIGURE 10.2
(a) Image.
(b) Horizontal intensity profile that includes the isolated point indicated by the arrow.
(c) Subsampled profile; the dashes were added for clarity. The numbers in the boxes are the intensity values of the dots shown in the profile. The derivatives were obtained using Eqs. (10-4) for the first derivative and Eq. (10-7) for the second.

- Effects of 1st and 2nd order derivatives

Characteristics of First and Second Order Derivatives

- First-order derivatives generally produce thicker edges in image
- Second-order derivatives have a stronger response to fine detail, such as thin lines, isolated points, and noise
- Second-order derivatives produce a double-edge response at ramp and step transition in intensity
- The sign of the second derivative can be used to determine whether a transition into an edge is from light to dark or dark to light

10.2.2 Point Detection

-1	-1	-1
-1	8	-1
-1	-1	-1

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

- We can use Laplacian masks for point detection
- Laplacian masks have the largest coefficient at the center of the mask while neighbor pixels have an opposite sign
- This mask will give the high response to the object that has the similar shape as the mask such as isolated points
- Notice that sum of all coefficients of the mask is equal to zero. This is due to the need that the response of the filter must be zero inside a constant intensity area

10.2.2 Detection of Isolated Points

- The Laplacian

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

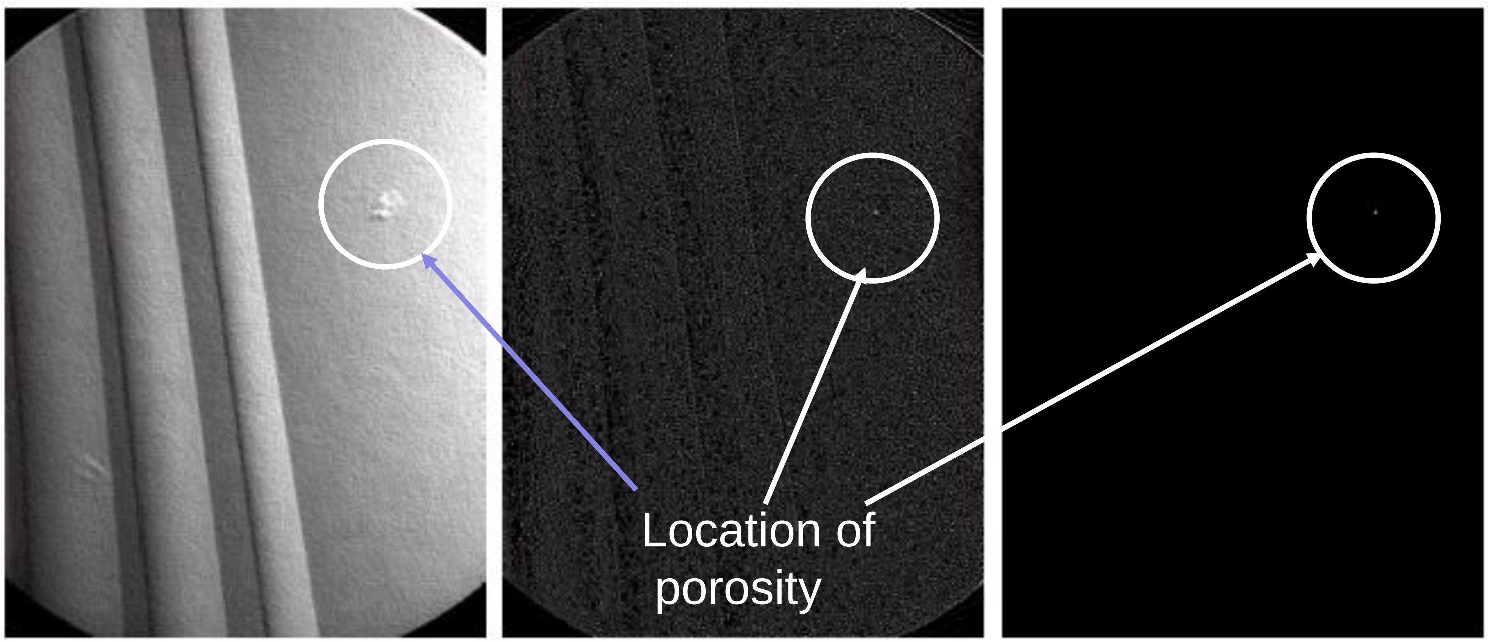
$$= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

$$g(x,y) = \begin{cases} 1 & \text{if } |R(x,y)| \geq T \\ 0 & \text{otherwise} \end{cases} \qquad R = \sum_{k=1}^{9} w_k z_k$$

Point Detection

Point detection can be done by applying the thresholding function

$$g(x,y)=\begin{cases}1 & |\nabla f(x,y)|\ge T\\ 0 & \text{otherwise}\end{cases}$$



X-ray image of the turbine blade with porosity

Laplacian image

After thresholding

1	1	1
1	−8	1
1	1	1

a
b c d

FIGURE 10.4 (a) Laplacian kernel used for point detection. (b) X-ray image of a turbine blade with a porosity manifested by a single black pixel. (c) Result of convolving the kernel with the image. (d) Result of using Eq. (10-15) was a single point (shown enlarged at the tip of the arrow). (Original image courtesy of X-TEK Systems, Ltd.)

10.2.3 Line Detection using Laplacian

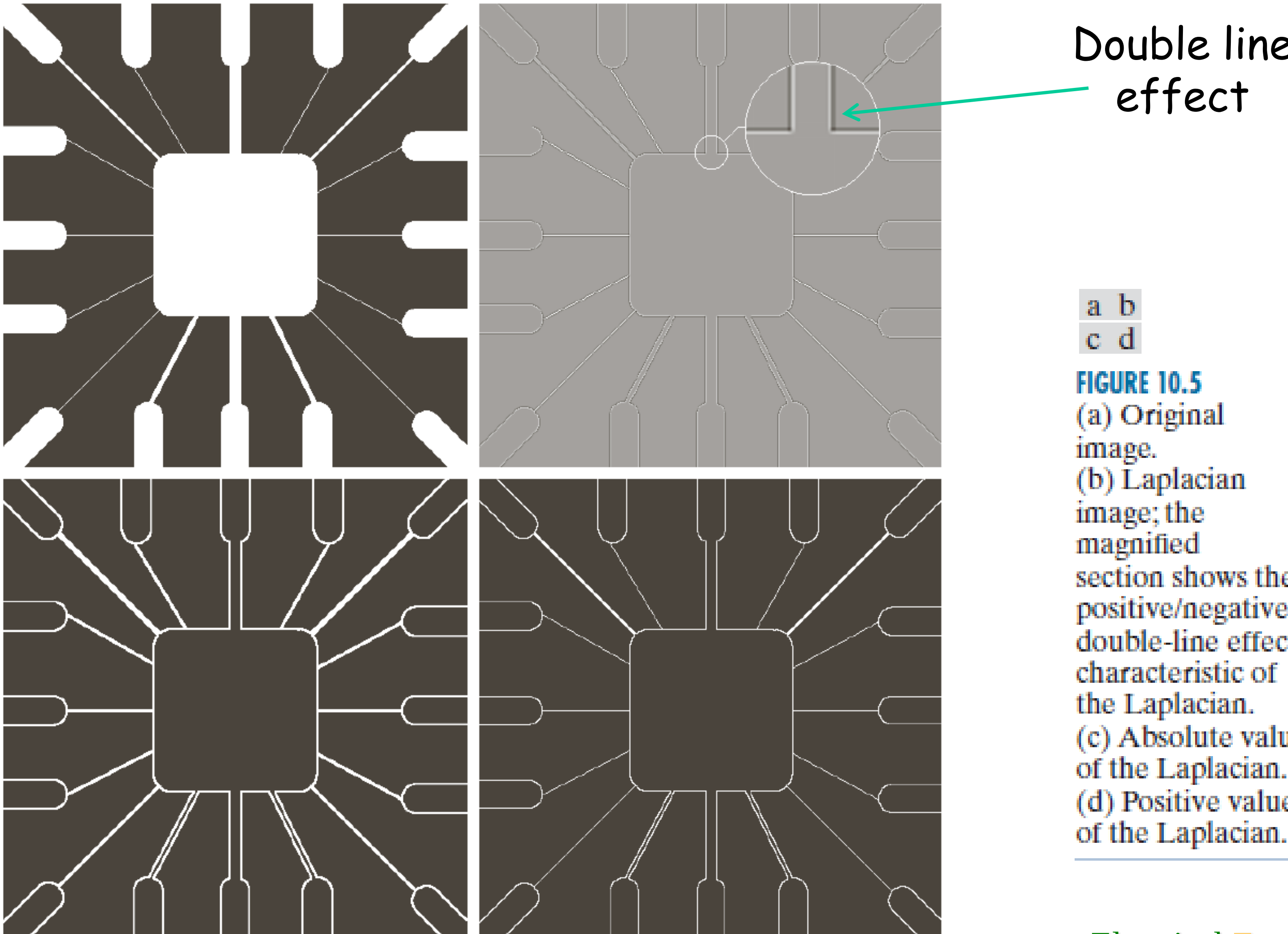


FIGURE 10.5 (a) Original image. (b) Laplacian image; the magnified section shows the positive/negative double-line effect characteristic of the Laplacian. (c) Absolute value of the Laplacian. (d) Positive values of the Laplacian.

Line Detection

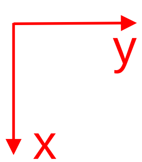
- Second derivatives
  - Result in a stronger response
  - Produce thinner lines than first derivatives

- Double-line effect of the second derivative must be handled properly

Detecting Line in Specified Directions

- Similar to point detection, line detection can be performed using the mask that has a shape similar to a part of a line
- A line in a digital image can be in several directions
- For simple line detection, 4 directions are mostly used:
    - Horizontal, +45 degree, vertical and -45 degree.

Line detection masks



Horizontal

−1	−1	−1
2	2	2
−1	−1	−1

+45°

2	−1	−1
−1	2	−1
−1	−1	2

Vertical

−1	2	−1
−1	2	−1
−1	2	−1

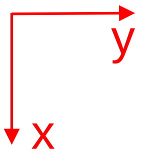
−45°

−1	−1	2
−1	2	−1
2	−1	−1

a b c d

FIGURE 10.6 Line detection kernels. Detection angles are with respect to the axis system in Fig. 2.19, with positive angles measured counterclockwise with respect to the (vertical) x-axis.

Detecting Line in Specified Directions



−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

2	−1	−1
−1	2	−1
−1	−1	2

+45°

−1	2	−1
−1	2	−1
−1	2	−1

Vertical

−1	−1	2
−1	2	−1
2	−1	−1

−45°

a b c d

FIGURE 10.6 Line detection kernels. Detection angles are with respect to the axis system in Fig. 2.19, with positive angles measured counterclockwise with respect to the (vertical) *x*-axis.

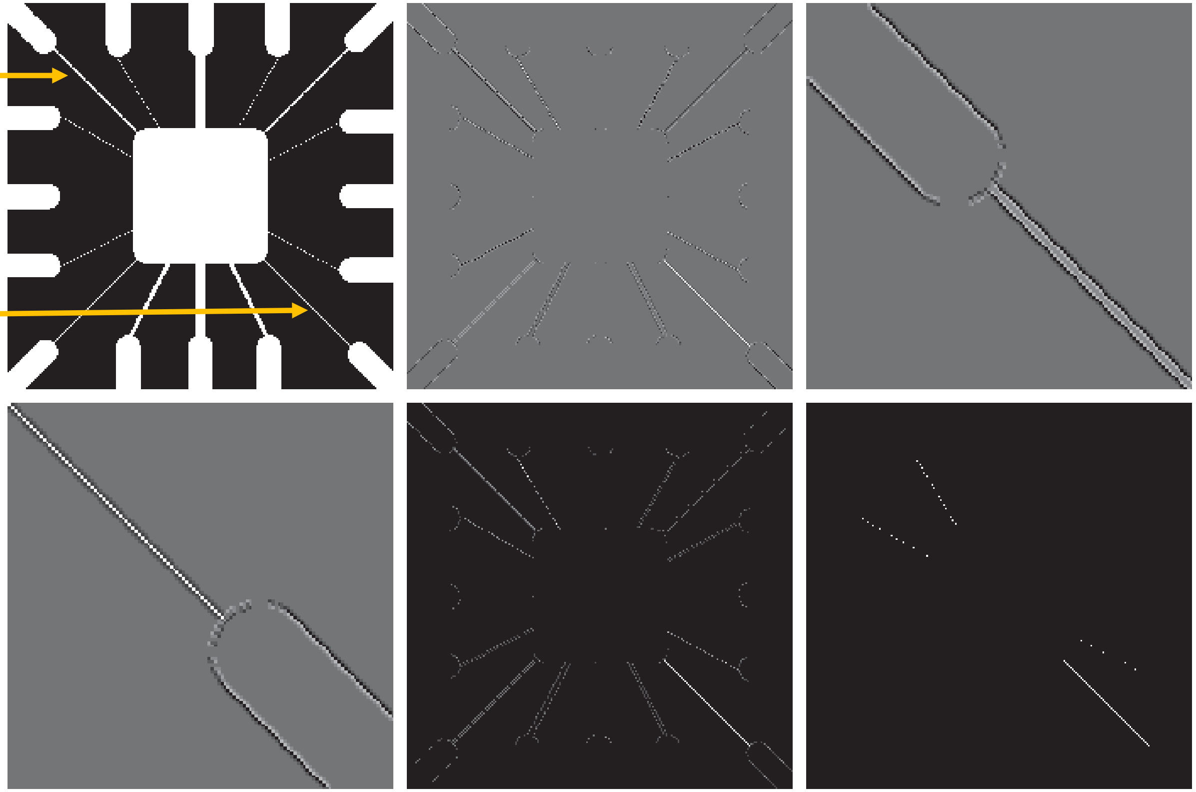
- Let R_1, R_2, R_3, and R_4 denote the responses of the masks in Fig. 10.6.
- At a given point in the image, if $|R_k|>|R_j|$, for all $j \neq k$, that point is said to be more likely associated with a line in the direction of mask k.

Detecting Line in Specified Directions

More than 1-pixel thick

1-pixel thick

Desired:
→ Want to detect 1-pixel thick line only



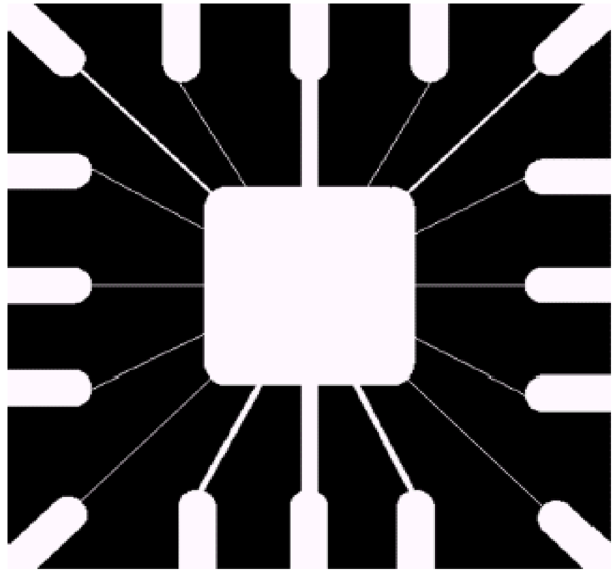
Mask: Tuned to detect 1-pixel thickness

a b c
d e f

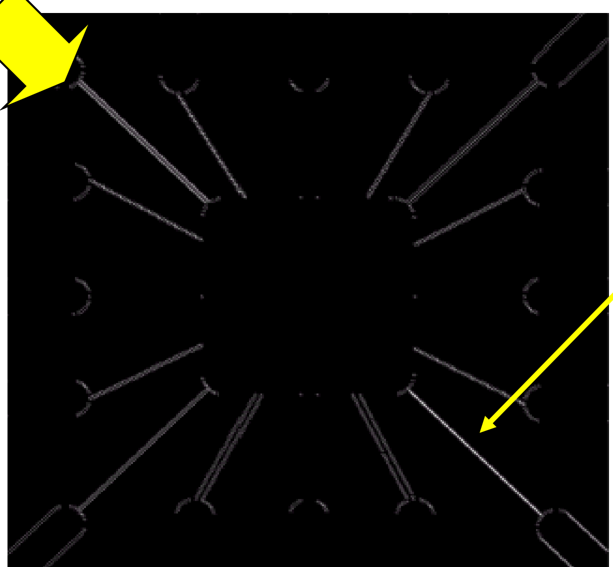
FIGURE 10.7 (a) Image of a wire-bond template. (b) Result of processing with the +45° line detector kernel in Fig. 10.6. (c) Zoomed view of the top left region of (b). (d) Zoomed view of the bottom right region of (b). (e) The image in (b) with all negative values set to zero. (f) All points (in white) whose values satisfied the condition $g > T$, where g is the image in (e) and $T = 254$ (the maximum pixel value in the image minus 1). (The points in (f) were enlarged to make them easier to see.)

Line Detection Example

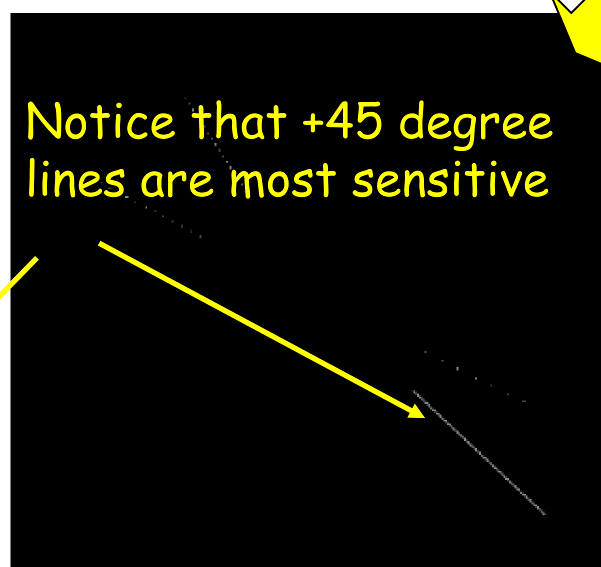
Binary wire bond mask image



Absolute value of result after processing with +45 line detector



Result after thresholding

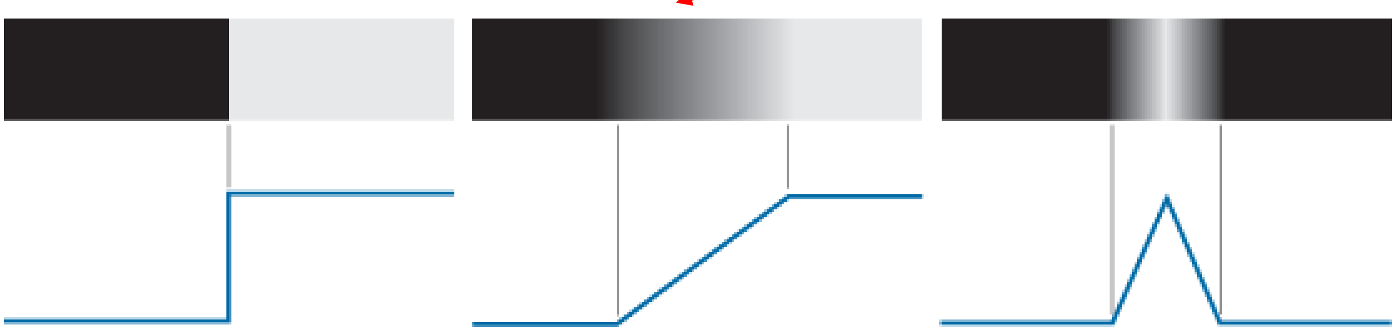


10.2.4 Edge Detection

- Generally, objects and background have different intensities.
- Edges are pixels where the brightness function changes abruptly
- Therefore, edges of the objects are the areas where abrupt intensity changes occur.
- Edge models

Blurred edge

Ideal step edge   Ideal ramp edge   Roof edge



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.

Edge Detection

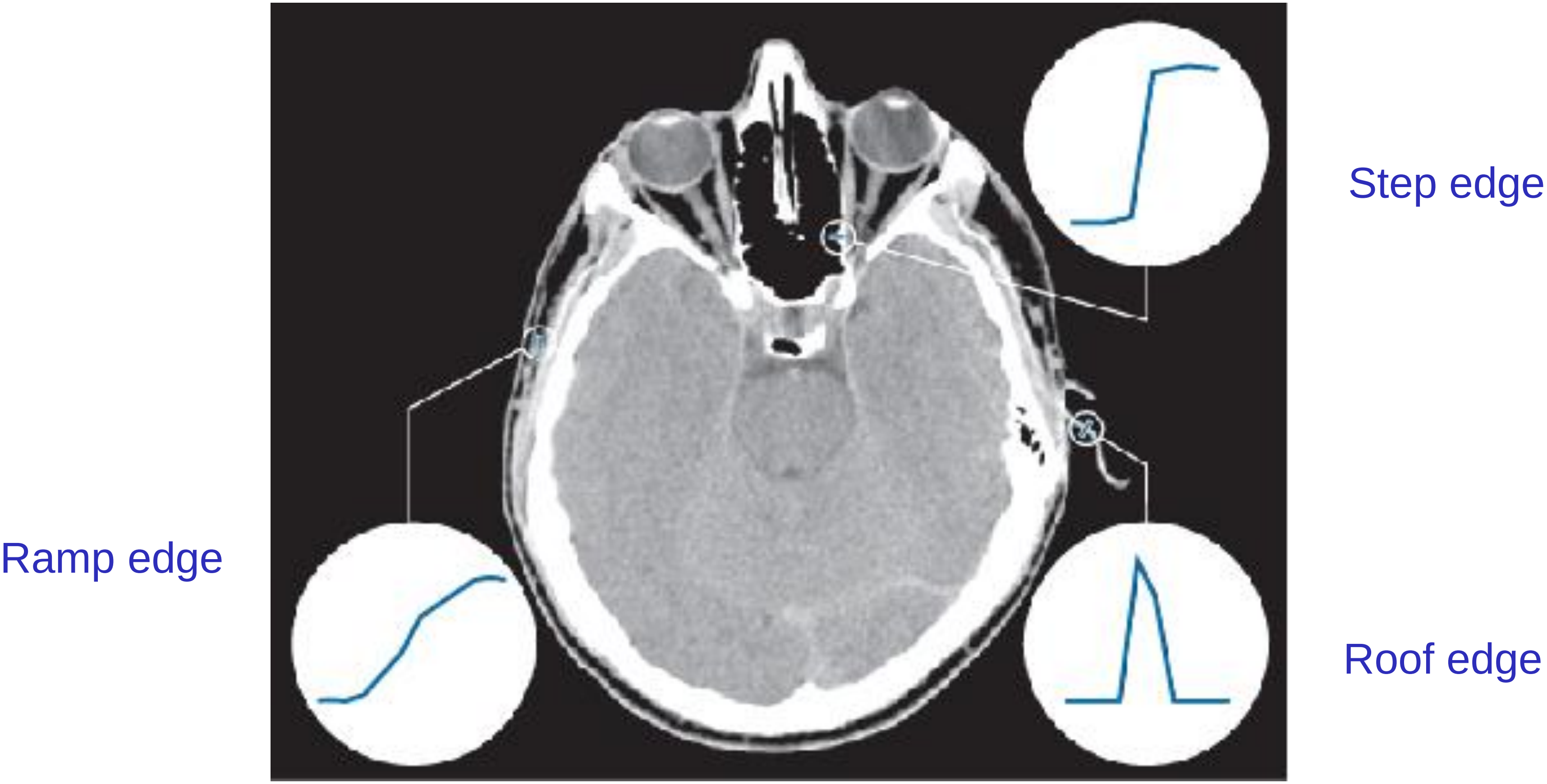


FIGURE 10.9 A 1508 × 1970 image showing (zoomed) actual ramp (bottom, left), step (top, right), and roof edge profiles. The profiles are from dark to light, in the areas enclosed by the small circles. The ramp and step profiles span 9 pixels and 2 pixels, respectively. The base of the roof edge is 3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

Ideal Ramp Edges and its Derivatives

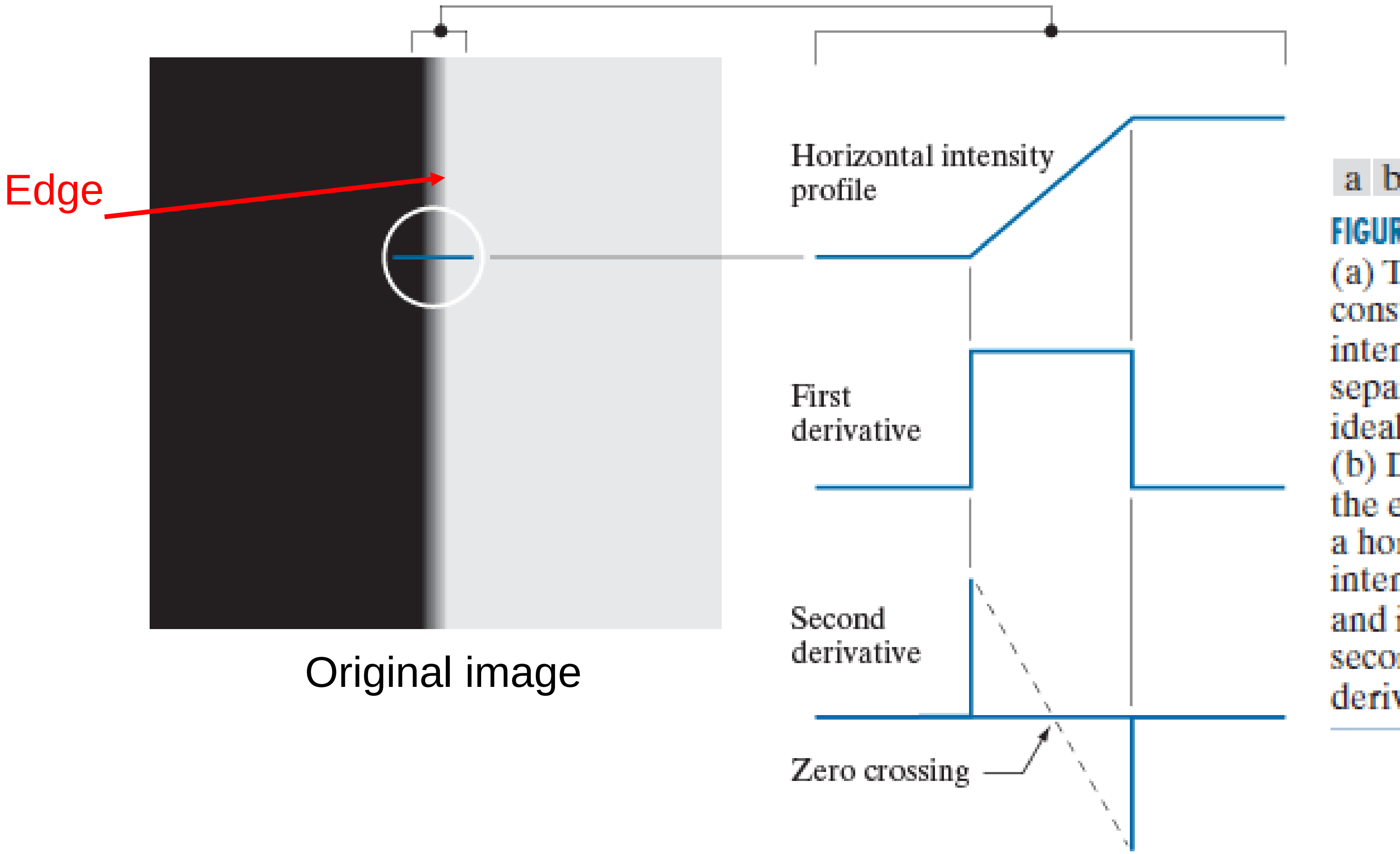


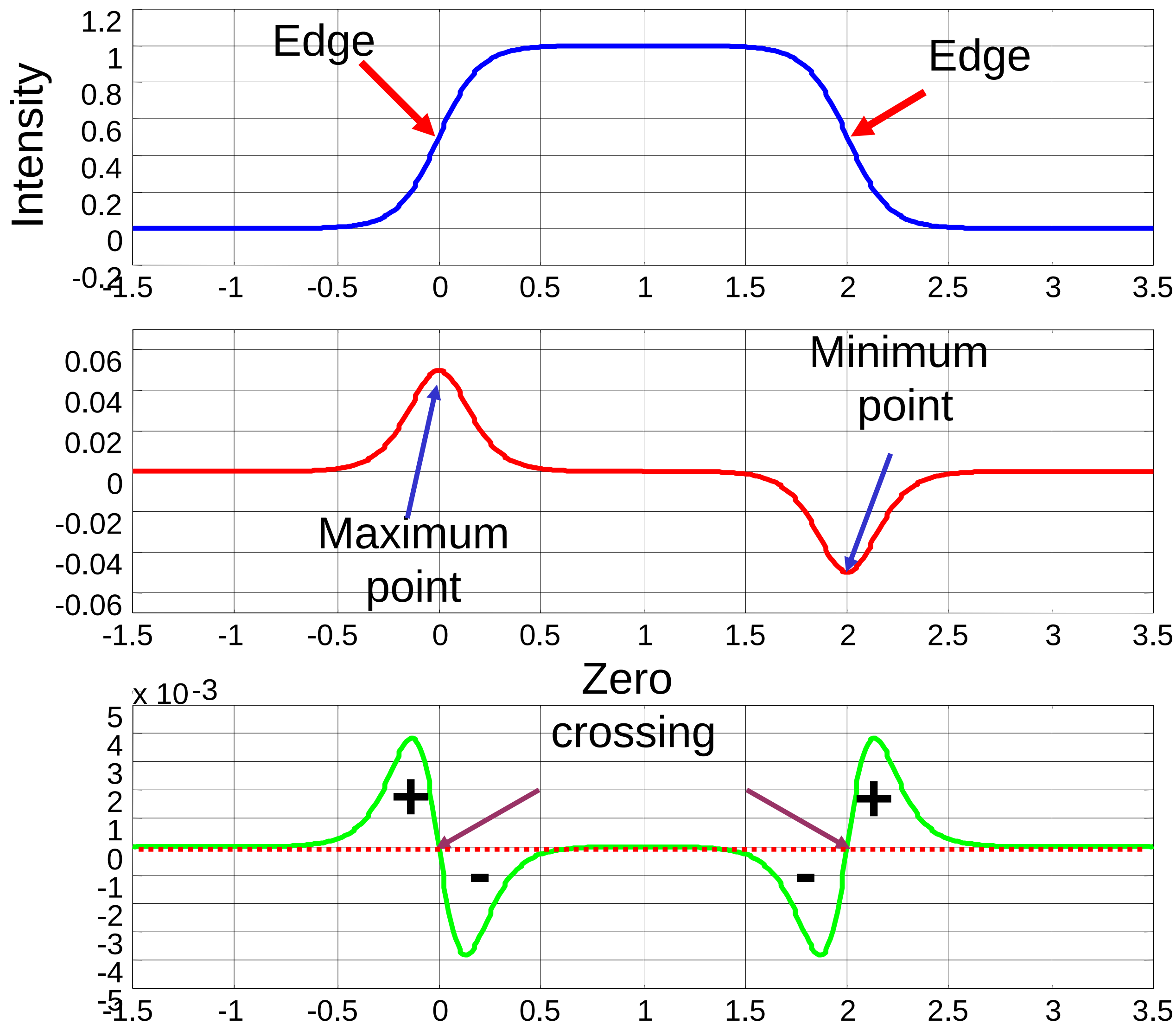
FIGURE 10.10 (a) Two regions of constant intensity separated by an ideal ramp edge. (b) Detail near the edge, showing a horizontal intensity profile, and its first and second derivatives.

Smoothed Step Edge and Its Derivatives

Gray level profile

The 1st derivative

The 2nd derivative



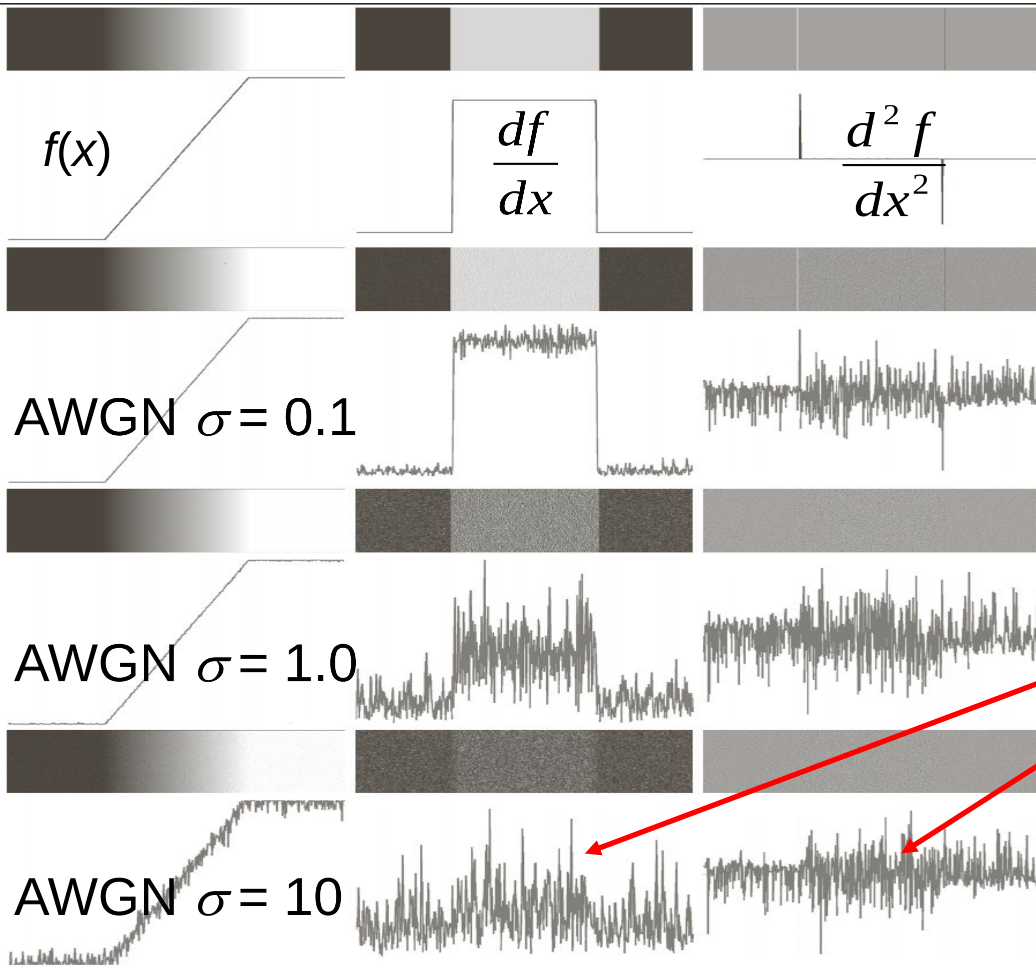
Derivative Based Edge Detection

- From the previous slide, we can conclude that:
  - Local maxima of the absolute value of the 1st derivative AND Zero crossing of the 2nd derivative occur at edges.

- Therefore, for detecting edges, we can apply zero crossing detection to the 2nd derivative image or thresholding the absolute value of the 1st derivative image.

- However, derivative operator is very sensitive to noise as shown in the next slide.

Noisy Edges and Derivatives



Derivative operator is a highpass filter and thus enhances noise.

Edge responses are buried by noise.

FIGURE 10.11 First column: 8-bit images with values in the range [0,255], and intensity profiles of a ramp edge corrupted by Gaussian noise of zero mean and standard deviations of 0.0, 0.1, 1.0, and 10.0 intensity levels, respectively. Second column: First-derivative images and intensity profiles. Third column: Second-derivative images and intensity profiles.

10.2 Basic Edge Detection by Using First-Order Derivative

$$\nabla f(x,y) \equiv \operatorname{grad}[f(x,y)] = \begin{bmatrix} g_x(x,y) \\ g_y(x,y) \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f(x,y)}{\partial x} \\ \dfrac{\partial f(x,y)}{\partial y} \end{bmatrix}$$

The magnitude of ∇f

$$M(x,y) = \|\nabla f(x,y)\| = \sqrt{g_x^2(x,y) + g_y^2(x,y)}$$

The direction of ∇f

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

Measured w.r.t. x-axis

The direction of the edge

$$\phi = \alpha(x,y) - 90^\circ$$

See Fig 10.12

Basic Edge Detection by Using First-Order Derivative

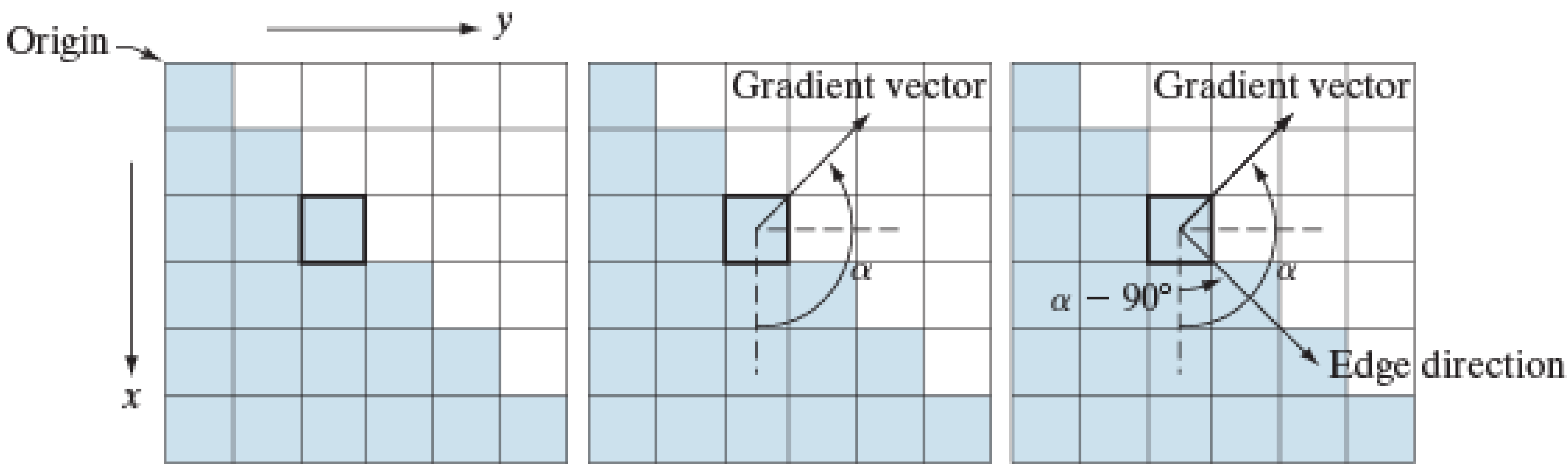
Edge normal: $\nabla f \equiv grad(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \end{bmatrix}$

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

In practice, sometimes the magnitude is approximated by

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

Basic Edge Detection by Using First-Order Derivative

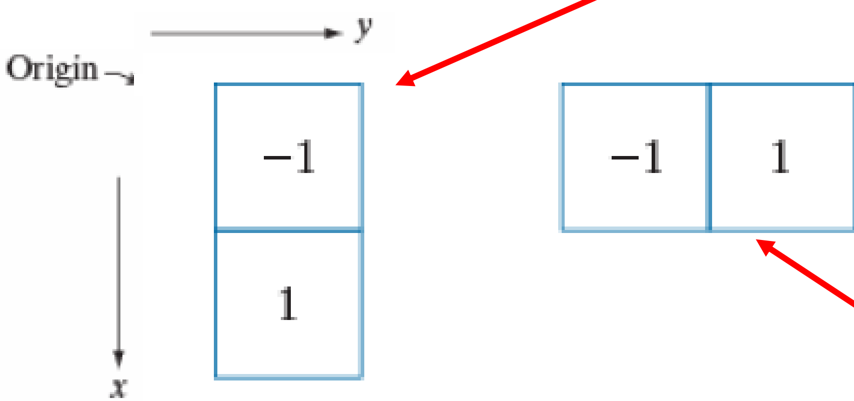


a b c

FIGURE 10.12 Using the gradient to determine edge strength and direction at a point. Note that the edge direction is perpendicular to the direction of the gradient vector at the point where the gradient is computed. Each square represents one pixel. (Recall from Fig. 2.19 that the origin of our coordinate system is at the top, left.)

1-D Kernels

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



a b

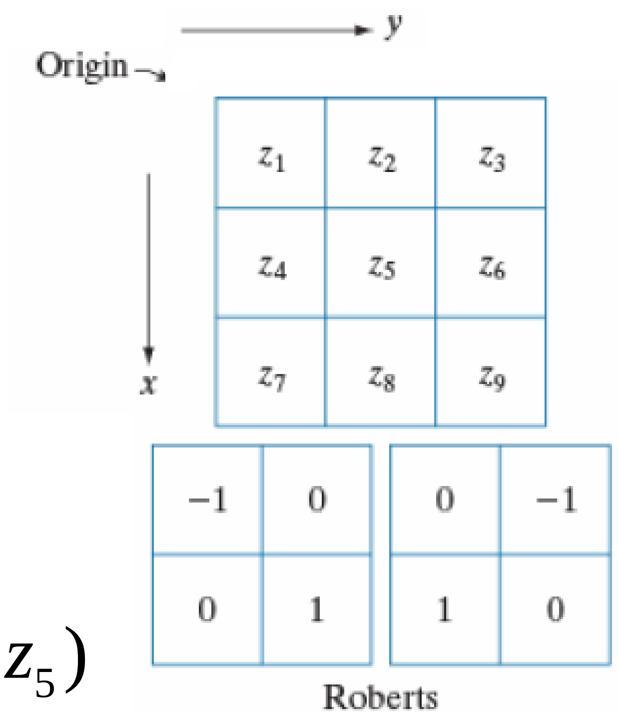
FIGURE 10.13
1-D kernels used to implement Eqs. (10-19) and (10-20).

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

2-D Kernels: Roberts Cross-Gradient Operator

- Kernels with Diagonal preference
- Simple but not very useful for Edge Detection

FIGURE 10.14
A 3×3 region of an image (the z's are intensity values), and various kernels



$$g_x = \frac{\partial f}{\partial x} = (z_9 - z_5)$$

$$g_y = \frac{\partial f}{\partial y} = (z_8 - z_6)$$

2-D Kernels: Prewitt Operator

- Approximate derivatives in x- and y-directions

FIGURE 10.14 A 3×3 region of an image (the z's are intensity values), and various kernels

Origin → y, ↓ x

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

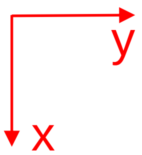
Prewitt

$$g_x = \frac{\partial f}{\partial x} = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

Masks for Estimating Partial Derivatives

In many cases, the mask for estimating partial derivative is anti-symmetric with respect to the orthogonal axis



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

g_x	−1	−1	−1	−1	0	1	g_y
	0	0	0	−1	0	1	
	1	1	1	−1	0	1	

Prewitt

g_x	−1	−2	−1	−1	0	1	g_y
	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 kernels used to compute the gradient at the point labeled z_5.

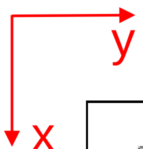
- For example, the Sobel mask for computing $\frac{\partial f}{\partial y}$ is anti-symmetric with respect to the y-axis.
- It has the positive sign on the right side and negative sign on the left side.
- **Sobel Kernels have better noise suppression (smoothing) characteristics**
- Notice that sum of all coefficients is equal to zero to ensure that the response of a constant intensity area is zero.

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

Recall: These Spatial filters were used in Chapter-3.6 for Sharpening of images

Masks for Detecting Diagonal Edges



0	1	1
−1	0	1
−1	−1	0

−1	−1	0
−1	0	1
0	1	1

Prewitt

0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

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 kernels used to compute the gradient at the point labeled z_5.

- The mask for detecting +45° edges is anti-symmetric with respect to the + 45° lines (w.r.t. vertical x-axis)
- The mask for detecting - 45° edges is anti-symmetry with respect to the - - 45° lines. (w.r.t. vertical x-axis)

Kirsch Compass Kernels

- Approximates Edge magnitudes and directions
- Approach: (1) Convolve with all 8 compass directions, (2) Assign Edge magnitude to the strongest value at that point → Always positive

Rotate CCW by 45°

Rotate CCW by 45°

Origin → y, x

−3	−3	5
−3	0	5
−3	−3	5

N

−3	5	5
−3	0	5
−3	−3	−3

NW

5	5	5
−3	0	−3
−3	−3	−3

W

5	5	−3
5	0	−3
−3	−3	−3

SW

5	−3	−3
5	0	−3
5	−3	−3

S

−3	−3	−3
5	0	−3
5	5	−3

SE

−3	−3	−3
−3	0	−3
5	5	5

E

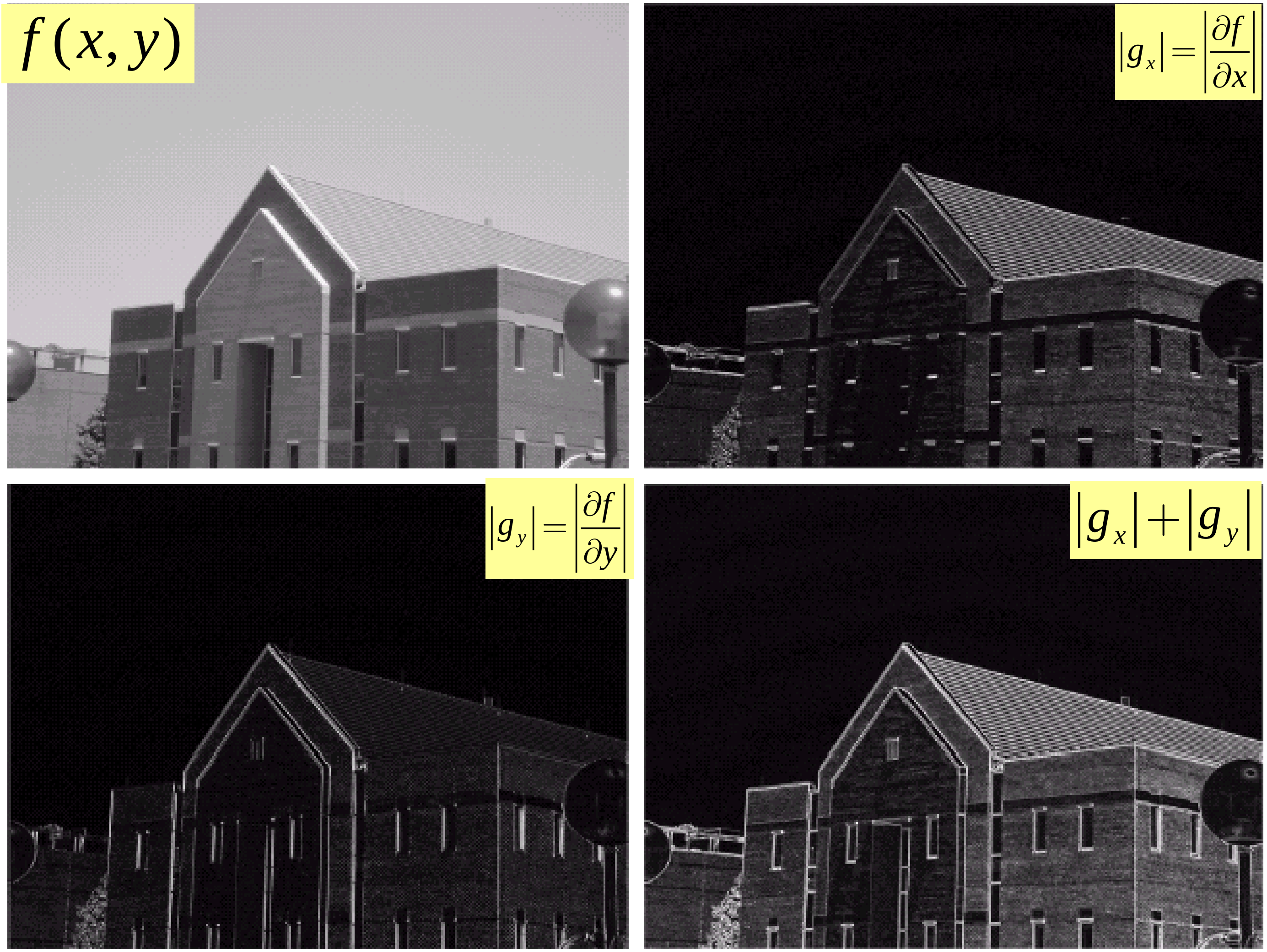
−3	−3	−3
−3	0	5
−3	5	5

NE

a	b	c	d
e	f	g	h

FIGURE 10.15 Kirsch compass kernels. The edge direction of strongest response of each kernel is labeled below it.

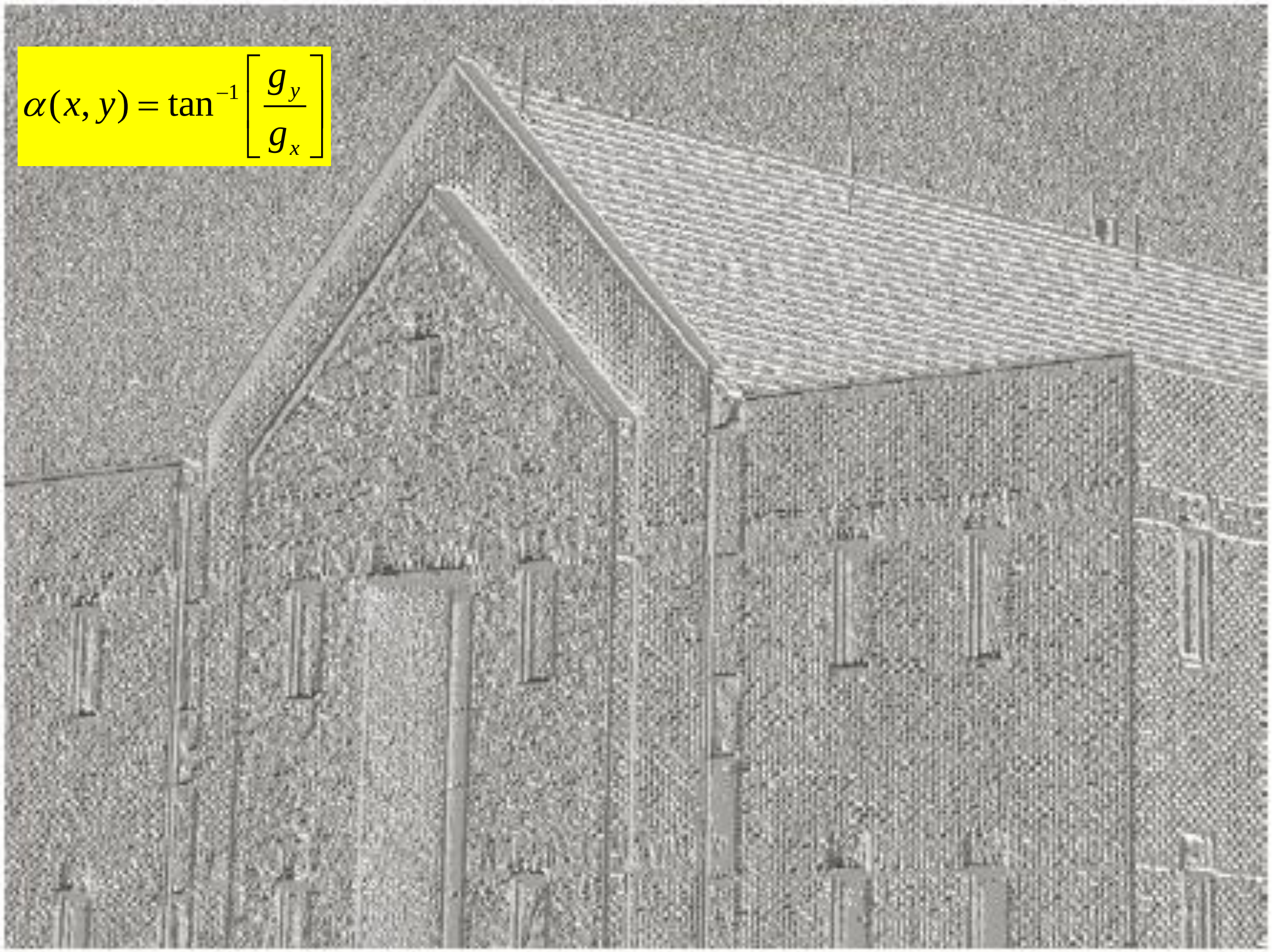
Examples of Image Gradient



a b
c d

FIGURE 10.16 (a) 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 kernel in Fig. 10.14(f) to filter the image. (c) $|g_y|$, obtained using the kernel in Fig. 10.14(g). (d) The gradient image, $|g_x| + |g_y|$.

Gradient Angle Image

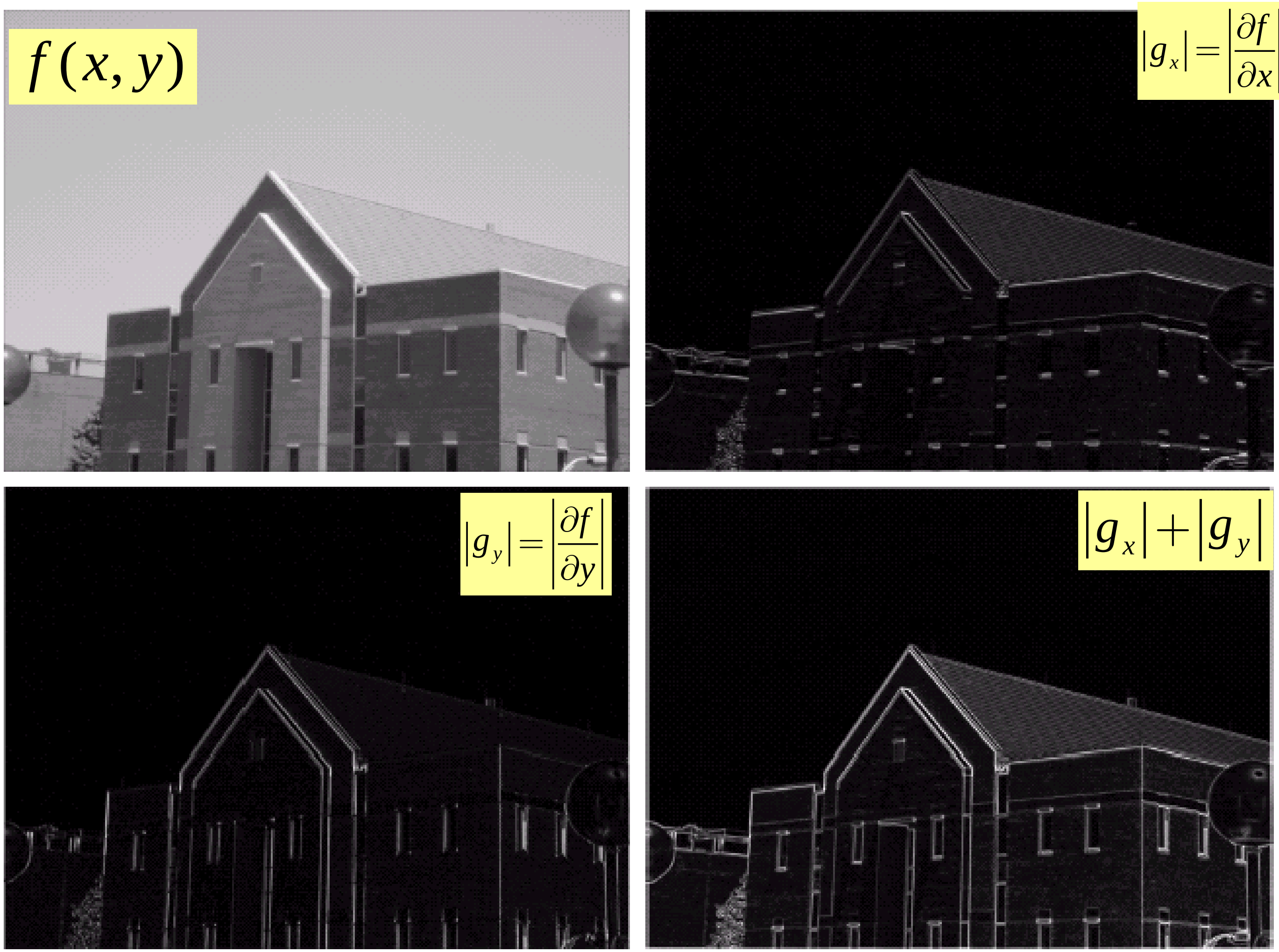


$$\alpha(x, y) = \tan^{-1}\left[\frac{g_y}{g_x}\right]$$

FIGURE 10.17 Gradient angle image computed using Eq. (10-18). 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.

Image Gradient after Smoothing

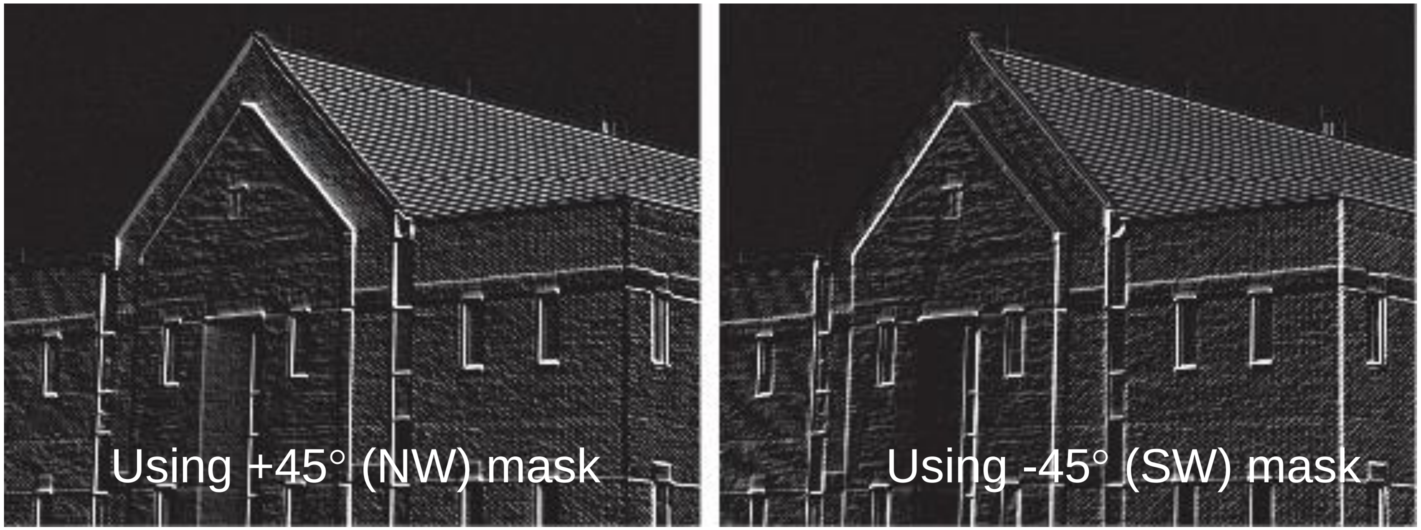
Note: The original image is smoothed by a 5x5 moving average mask first.

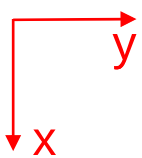


a b
c d

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

Example of Diagonal Edges





−3	5	5
−3	0	5
−3	−3	−3

NW

5	5	−3
5	0	−3
−3	−3	−3

SW

a b

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

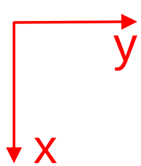
Correction: Should be 10.15 (b)

Note: The original image is smoothed by a 5x5 moving average mask first.

Example of Diagonal Edges

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).



0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

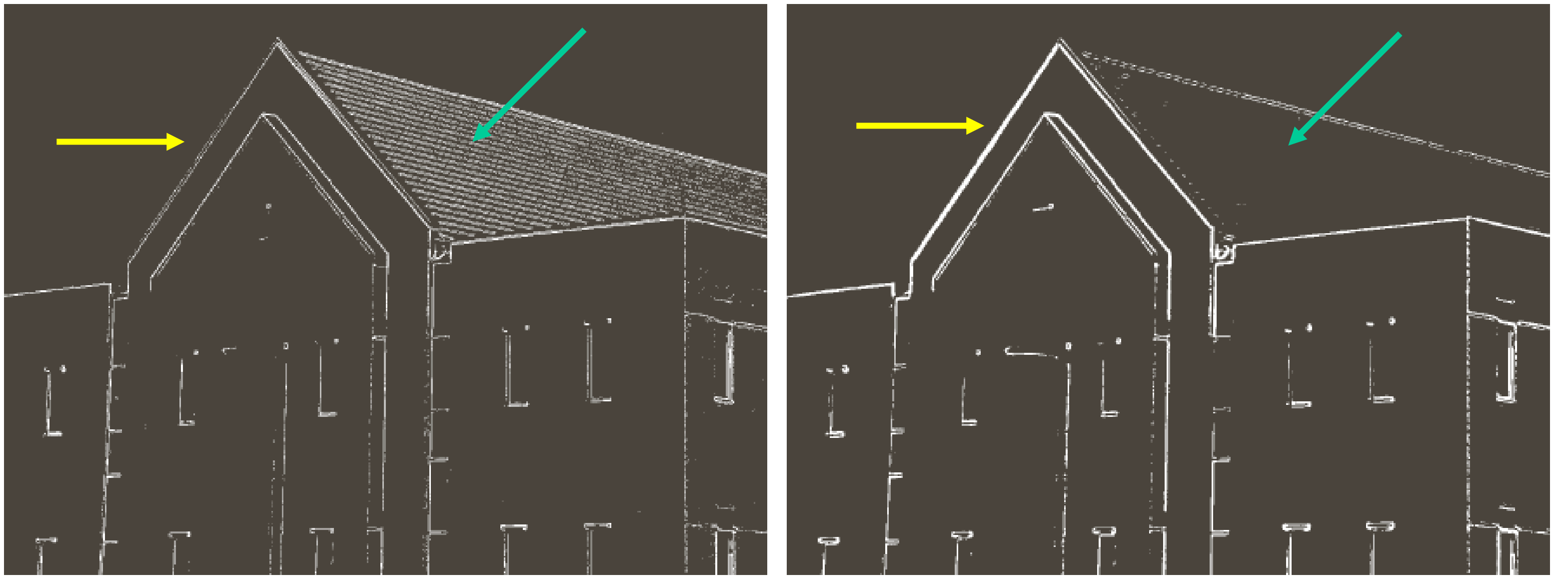
Sobel

Note: The original image is smoothed by a 5x5 moving average mask first.

From 3rd Ed

Combine Gradient with Thresholding

Thresholding Without smoothing **Thresholding after smoothing**



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.

From 3rd Ed

10.2.6 Advanced Techniques for Edge Detection

Motivation: Marr and Hildreth [1980] argued that

I. Intensity changes are not independent of image scale and so their detection requires the use of operators of different **sizes**; and

II. A sudden intensity change will give rise to a peak or trough in the first derivative or, equivalently, to a *zero-crossing in the second derivative*

- It is essentially a Zero-crossing based Edge detector
- For a large scale Laplacian mask, a Laplacian of Gaussian (LOG) mask is used

10.2.6 Advanced Techniques for Edge Detection

- The Marr-Hildreth edge detector

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}, \quad \sigma : \text{space constant.}$$

Laplacian of Gaussian (LoG)

$$\nabla^2 G(x,y) = \frac{\partial^2 G(x,y)}{\partial x^2} + \frac{\partial^2 G(x,y)}{\partial y^2}$$

$$= \frac{\partial}{\partial x}\left[\frac{-x}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}\right] + \frac{\partial}{\partial y}\left[\frac{-y}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}\right]$$

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

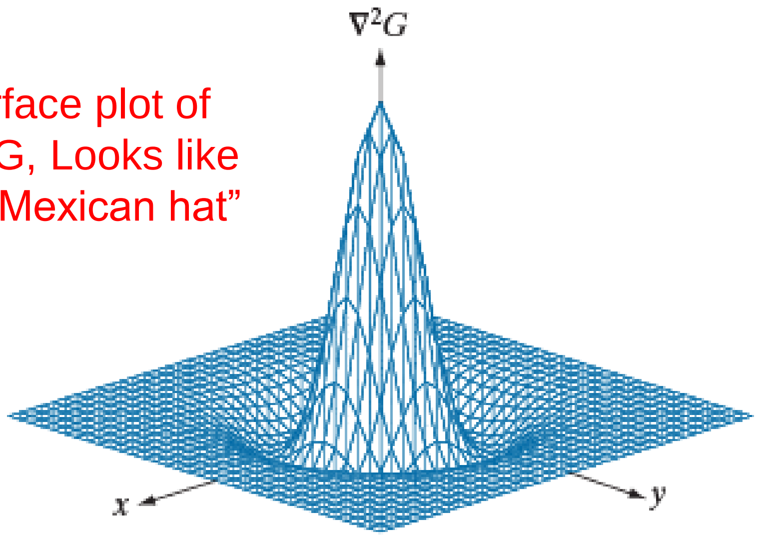
$$= \left[\frac{x^2 + y^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Note: Negative valued at x=0, y=0 and close to origin

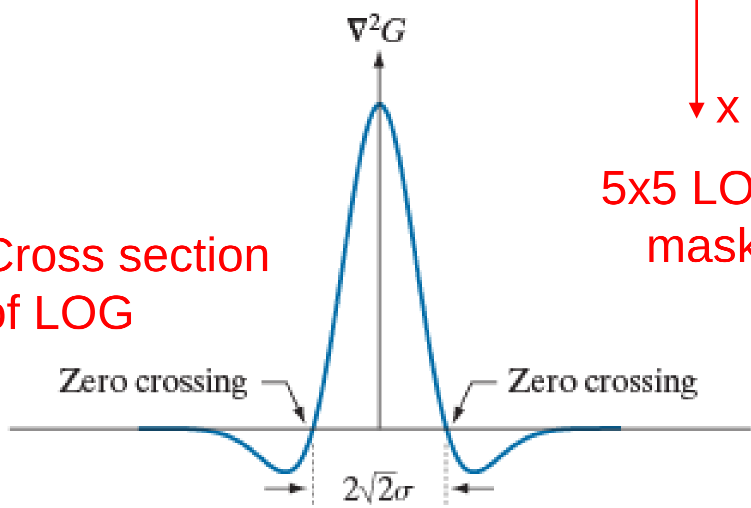
- For a large scale Laplacian mask, a Laplacian of Gaussian (LOG) mask is used

Laplacian of Gaussian (LoG)

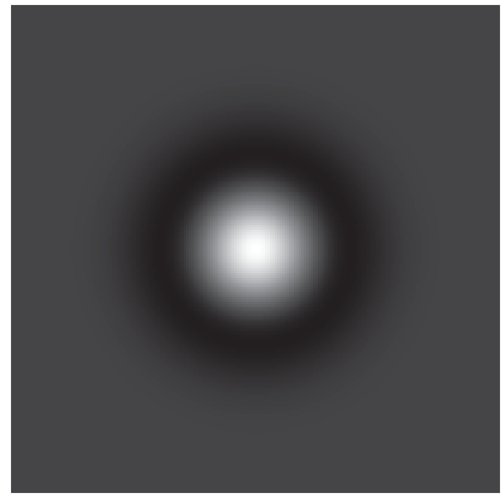
Surface plot of LOG, Looks like a “Mexican hat”



Cross section of LOG



LOG image



y

x

5x5 LOG mask

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

a b
c d

FIGURE 10.21
(a) 3-D 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 × 5 kernel approximation to the shape in (a). The negative of this kernel would be used in practice.

Marr-Hildreth Algorithm

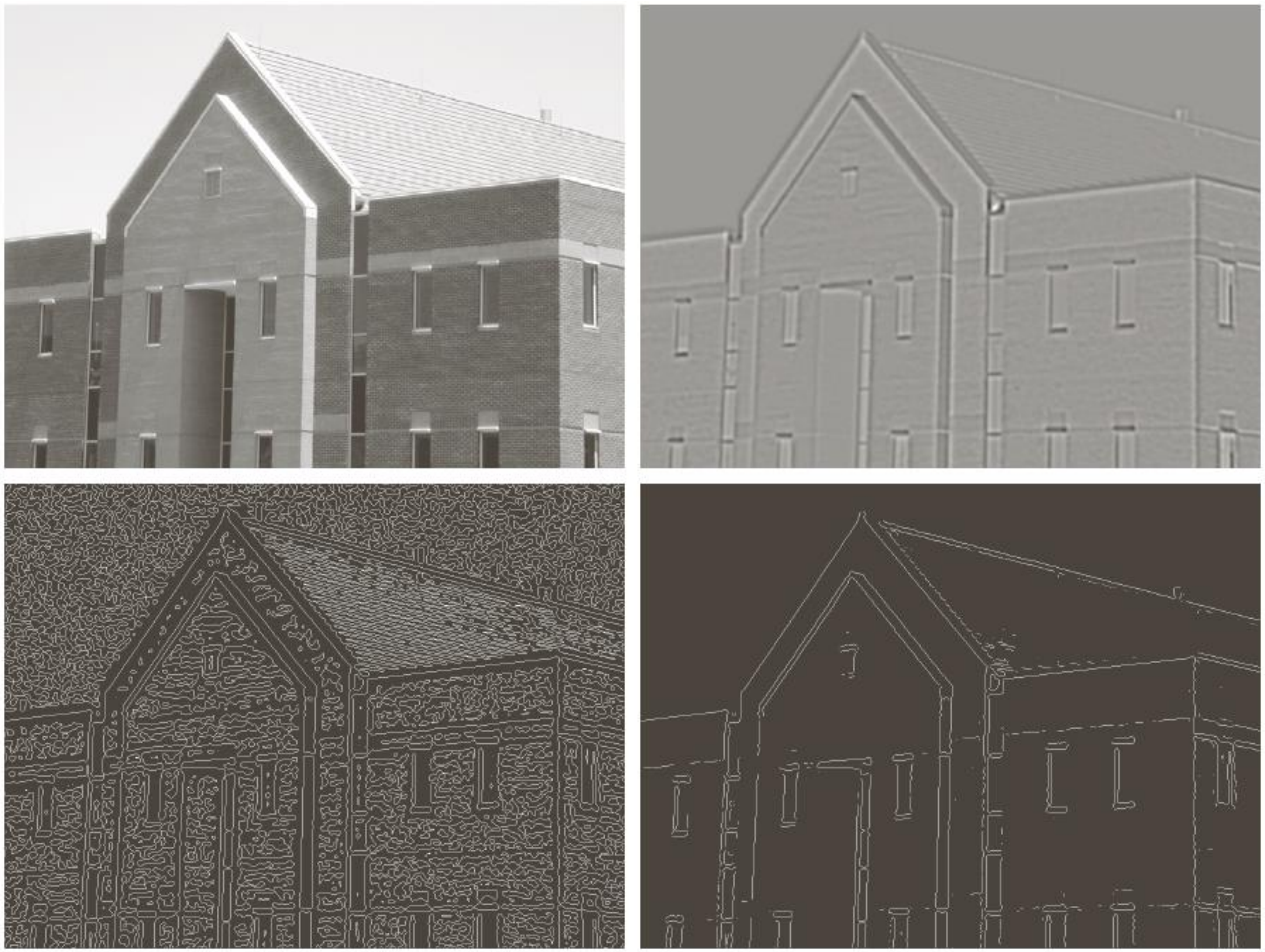
1. Filter the input image with an nxn Gaussian lowpass filter. n is the smallest odd integer greater than or equal to 6σ

2. Compute the Laplacian of the image resulting from Step-1

3. Find the zero-crossings of the image from Step-2

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

Marr-Hildreth Algorithm - Results



a b
c d

FIGURE 10.22
(a) Image of size 834×1114 pixels, with intensity values scaled to the range [0, 1].
(b) Result 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.

Canny Edge Detector [1986]

Optimal **for step edges corrupted by white noise**

Canny's Objectives

1. **Low error rate**

   All edges should be found. No spurious edges. The edges detected must be as close as possible to the true edges

2. **Edge points should be well localized**

   The edges located must be as close as possible to the true edges, i.e., minimum distance between a detected edge-point to center of true edge

3. **Single edge point response**

   The number of local maxima around the true edge should be minimum – detector should not identify multiple edge pixels where single edge pixel exists

The Canny Edge Detector: Algorithm

Let $f(x, y)$ denote the input image and $G(x, y)$ denote the Gaussian function:

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

Form a smoothed image, $f_s(x, y)$ by convolving G and f:

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

Canny Edge Detector: Algorithm

Compute the gradient magnitude and direction (angle):

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

and

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

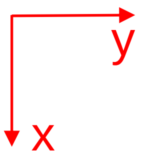
where $g_x = \partial f_s / \partial x$ and $g_y = \partial f_s / \partial y$

Slides 31 & 32 →

Note : Any of the filter mask pairs in Fig.10.14 can be used to obtain g_x and g_y

Masks for Estimating Partial Derivatives

In many cases, the mask for estimating partial derivative is anti-symmetric with respect to the orthogonal axis



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

g_x

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

g_y

Prewitt

g_x

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

g_y

Sobel

+45° edges (w.r.t. vertical x-axis)

- 45° edges (w.r.t. vertical x-axis)

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 e
f g

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

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

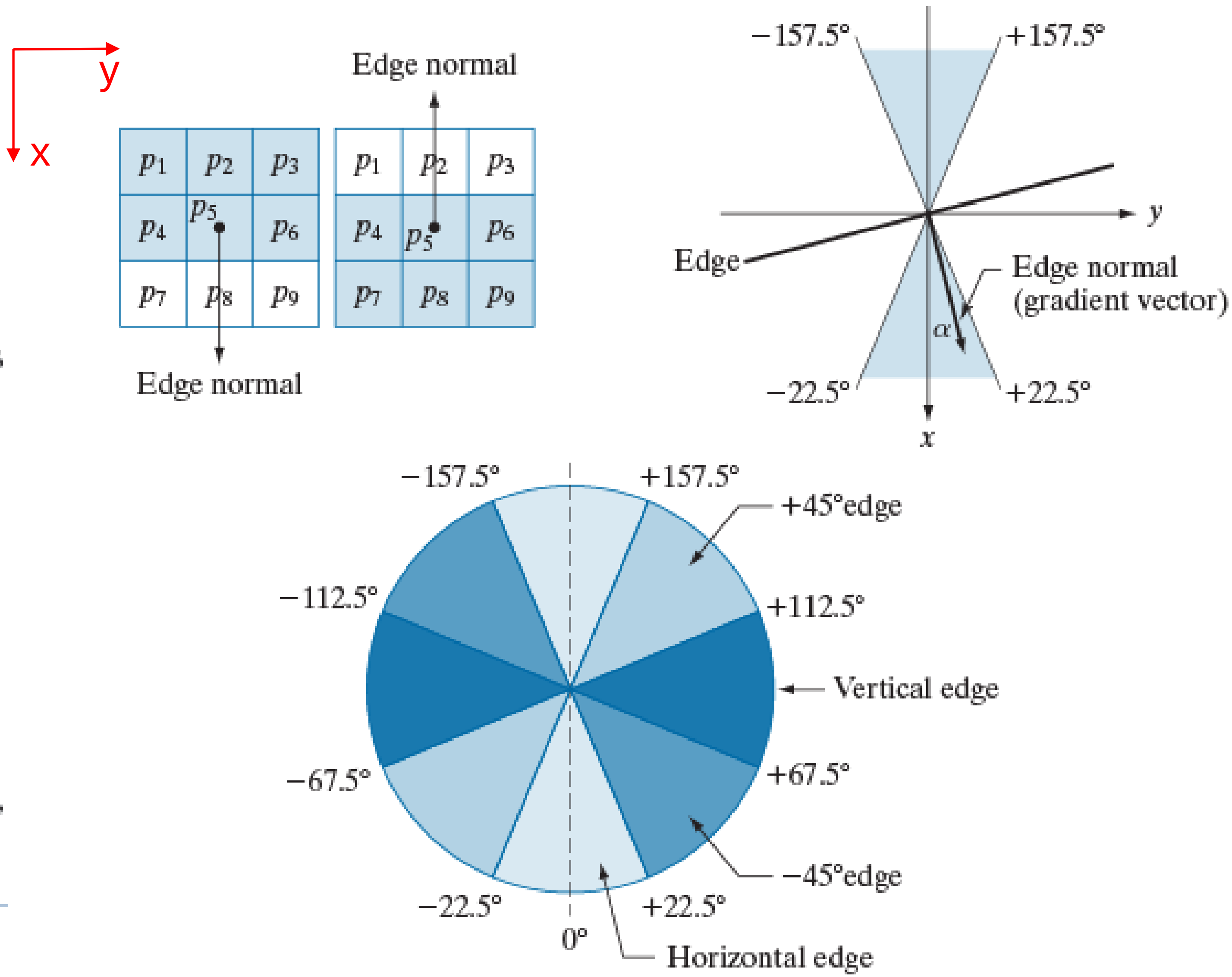
$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

WRIGHT STATE UNIVERSITY

Angle Ranges for Edge Normals

a b
c

FIGURE 10.24
(a) Two possible orientations of a horizontal edge (shaded) in a 3 × 3 neighborhood. (b) Range of values (shaded) of α, 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 × 3 neighborhood. Each edge direction has two ranges, shown in corresponding shades.



Canny Edge Detector: Algorithm

The gradient $M(x, y)$ typically contains wide ridge around local maxima. **Next step is to thin those ridges.**

Nonmaxima Suppression :

Let $d_1, d_2, d_3,$ and d_4 denote the four basic edge directions for a 3×3 region: horizontal, -45°, vertical, +45°, respectively.

1. Find the direction d_k that is closest to $\alpha(x, y)$.
2. If the value of $M(x, y)$ is less than at least one of its two neighbors *along* d_k, let $g_N(x, y) = 0$ (suppress: not a peak); otherwise, let $g_N(x, y) = M(x, y)$

where, $g_N(x, y)$: **Nonmaxima - Suppressed image**

Canny Edge Detector: Algorithm

The final operation is to **threshold** $g_N(x, y)$:

- **GOAL : Reduce false edge points.**

<u>Hysteresis thresholding</u> : Create two addional images

$g_{NH}(x, y) = g_N(x, y) \geq T_H$: High Threshold (strong edges)

$g_{NL}(x, y) = g_N(x, y) \geq T_L$: Low Threshold (weak edges)

- All strong pixels in $g_{NH}(x, y)$ are valid edges

And

- Use $g_{NL}(x, y) = g_{NL}(x, y) - g_{NH}(x, y)$ to first remove strong edges from the weak edges.

- Then link/connect with strong edges or discard

Canny Edge Detector: Algorithm

Depending on the value of T_H, the edges in $g_{NH}(x, y)$ typically have gaps. Longer edges are formed using the following procedure:

(a) Locate the next unvisited edge pixel, p, in $g_{NH}(x, y)$.

(b) Mark as valid edge pixel all the weak pixels in $g_{NL}(x, y)$ that are connected to p using 8-connectivity.

(c) If all nonzero pixel in $g_{NH}(x, y)$ have been visited go to step (d), else return to (a).

(d) Set to 0 all pixels in $g_{NL}(x, y)$ that were not marked as valid edge pixels.

Canny Edge Detection: Summary

STEPS

- Smooth the input image with a Gaussian filter
- Compute the gradient magnitude and angle images
- Apply non-maxima suppression to the gradient magnitude image
- Use double thresholding and connectivity analysis to detect and link edges

Comparison of Marr-Hildreth and Canny Algorithms

a b
c d

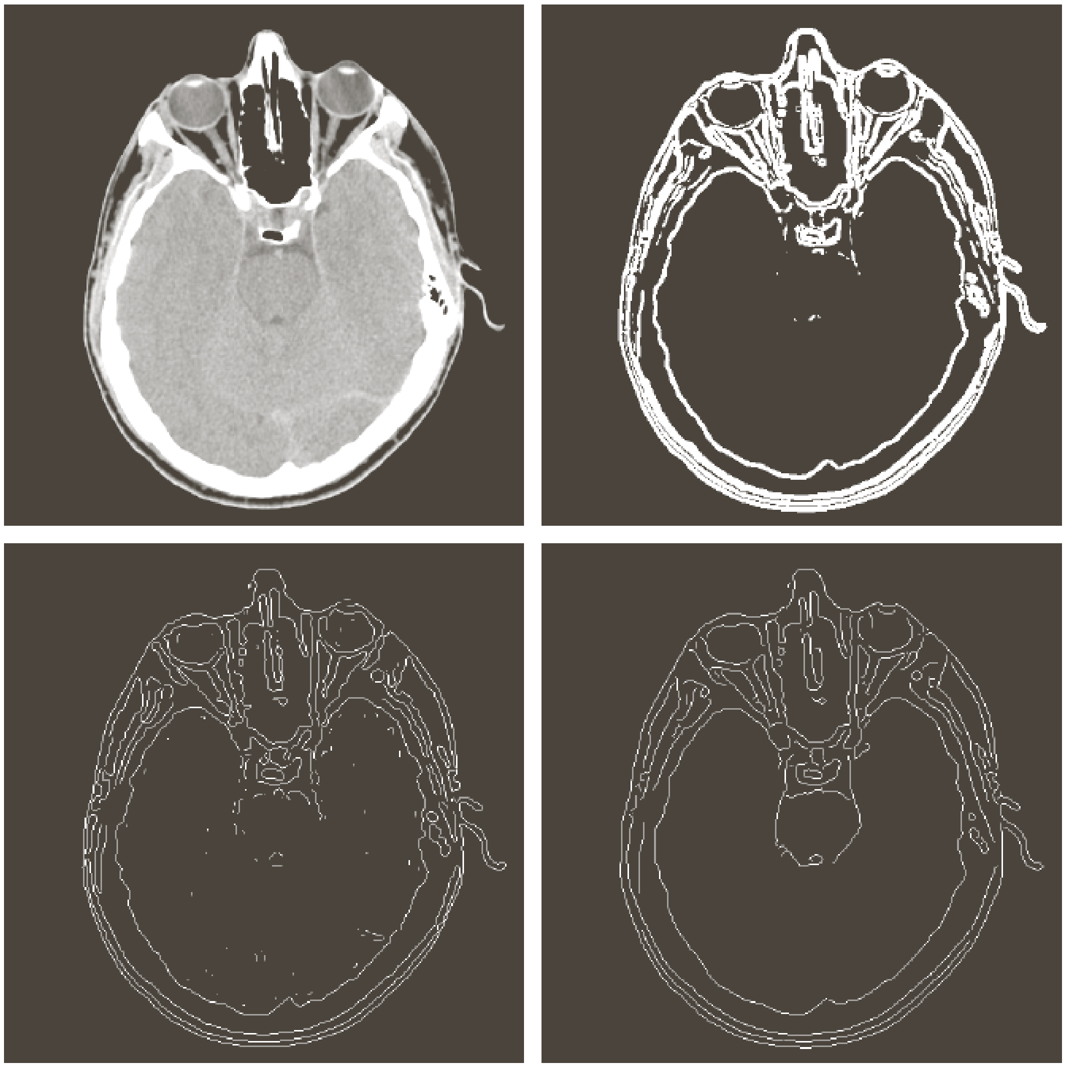
FIGURE 10.25 (a) Original image of size 834×1114 pixels, with intensity values scaled to the range [0, 1]. (b) Thresholded gradient of the smoothed image. (c) Image obtained using the Marr-Hildreth algorithm. (d) Image obtained using the Canny algorithm. Note the significant improvement of the Canny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

Slide-48

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

WRIGHT STATE UNIVERSITY

Comparison of Marr-Hildreth and Canny Algorithms



a b
c d

FIGURE 10.26 (a) Head CT image of size 512×512 pixels, with intensity values scaled to the range [0, 1]. (b) Thresholded gradient of the 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

Lenna Image

Results with Marr-Hildreth Edge Detector

Results of Canny Edge Detector

10.2.7 Edge Linking and Boundary Detection

- Edge detection typically is followed by linking algorithms designed to assemble edge pixels into meaningful edges and/or region boundaries

- Three approaches to edge linking
  - Local processing
  - Regional processing
  - Global processing

Local Processing

- Analyze the characteristics of pixels in a small neighborhood about every point (x, y) that has been declared as an edge point
- All points similar according to predefined criteria are linked, forming an edge of pixels.
- How to establish similarity ?
  (1) The strength (magnitude) and
  (2) The direction of the gradient vector.
- A pixel with coordinates (s, t) in S_{xy} is linked to the pixel at (x, y) if both magnitude and direction criteria are satisfied.

Local Processing

- Let S_{xy} denote the set of coordinates of a neighborhood centered at point (x, y) in an image.

- An edge pixel with coordinate (s,t) in S_{xy} is similar in *magnitude* to the pixel at (x, y) if

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

- An edge pixel with coordinate (s,t) in S_{xy} is similar in *angle* to the pixel at (x, y) if

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

Local Processing: Steps (1)

1. Compute the gradient magnitude and angle arrays, $M(x, y)$ and $\alpha(x, y)$ of the input image $f(x, y)$

2. Form a binary image, g, whose value at any pair of coordinates (x, y) is given by

$$g(x, y) = \begin{cases} 1 & \text{if } M(x, y) > T_M \text{ and } \alpha(x, y) = A \pm T_A \\ 0 & \text{otherwise} \end{cases}$$

T_M : Magnitude Threshold

A: Specified angle direction

T_A: "Band" of acceptable directions about A

Local Processing: Steps (2)

3. Scan the rows of g and fill (set to 1) all gaps (sets of 0s) in each row that do not exceed a specified length, K.

4. To detect gaps in any other direction, rotate g by this angle and apply the horizontal scanning procedure in step 3.

- Parameter Choices for Figure 10.27 (next slide)

T_M : 30% of maximum gradient

A: 90°

T_A: 45°

Finally, fill gaps of K=25 or fewer pixels (~5% of image width)

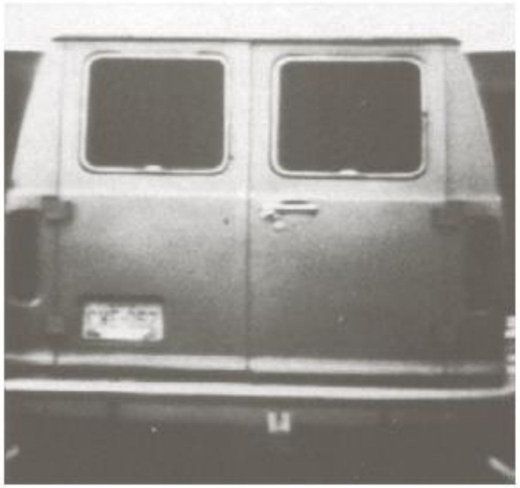
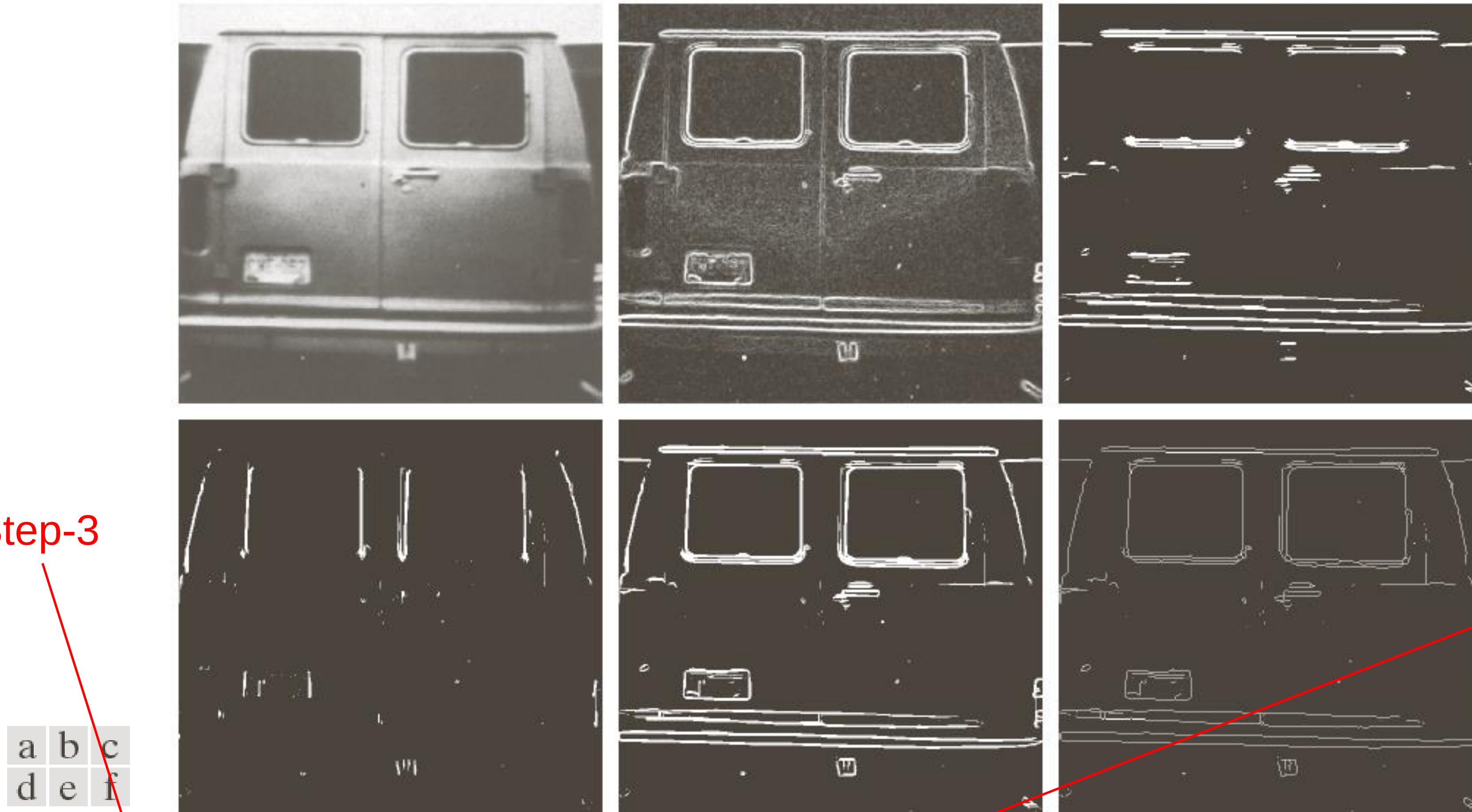


FIGURE 10.27 (a) A 534 × 566 image of the rear of a vehicle. (b) Gradient magnitude image. (c) Horizontally connected edge pixels. (d) Vertically connected edge pixels. (e) The logical OR of the two preceding images. (f) Final result obtained using morphological thinning. (Original image courtesy of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

Result with Local Processing



Step-3

Step-4

FIGURE 10.27 (a) A 534 × 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.)

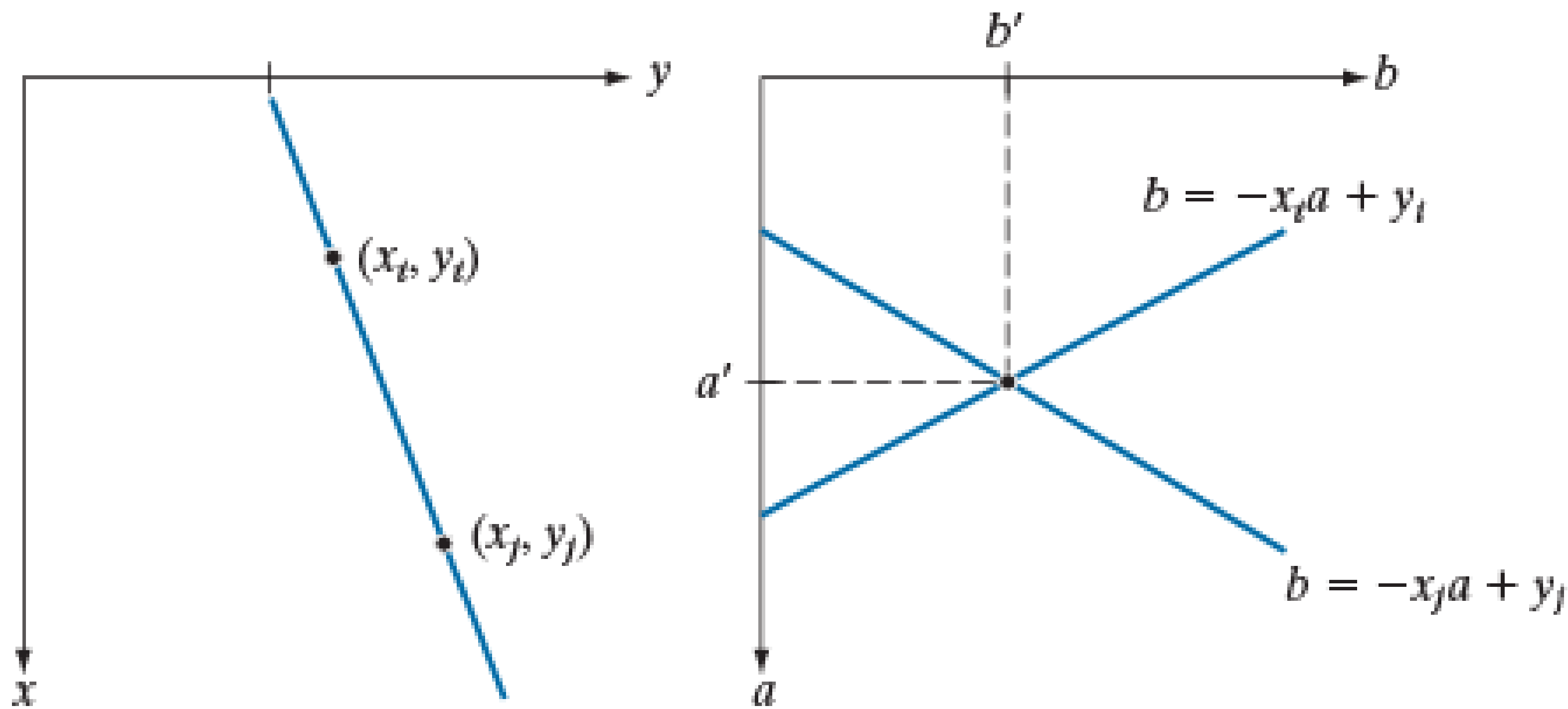
Chapter-9

Global Processing Using the Hough Transform

- **The Hough Transform:** A general technique for identifying the locations and orientations of certain types of features in a digital image.
- Developed by Paul Hough in 1962 and patented by IBM
- Consists of parameterizing a description of a feature at any given location in the original image's space.
- A mesh in the space defined by these parameter is then generated, and
- At each mesh point a value is accumulated, indicating how well an object generated by the parameters defined at that point fits the given image.
- Mesh points that accumulate relatively larger values then describe features that may be projected back onto the image, fitting to some degree the features actually present in the image.

http://planetmath.org/encyclopedia/HoughTransform.html

Parameter Space

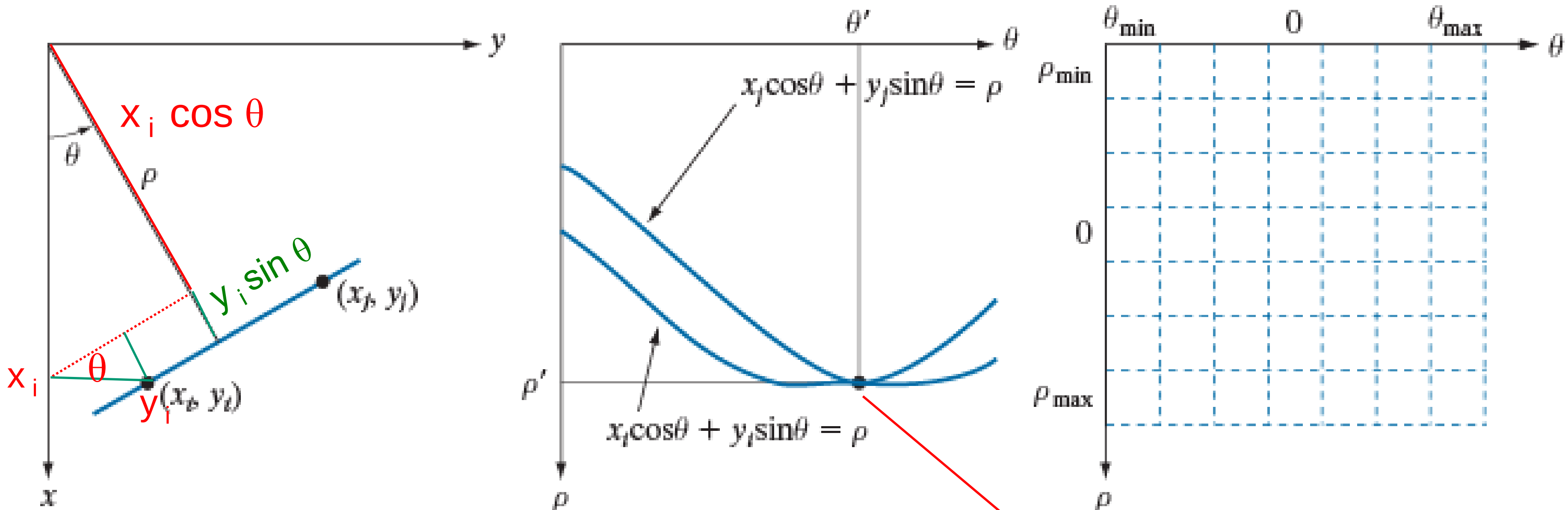


a b

FIGURE 10.28
(a) xy-plane.
(b) Parameter space.

Parameter Space

x-y plane ρ-θ plane



At $\theta = 0$, $x_i \cos\theta + y_i \sin\theta = x_i = \rho$
At $\theta = 90$, $x_i \cos\theta + y_i \sin\theta = y_i = \rho$

Both (x_i, y_i) and (x_j, y_j) corresponds to same ρ

a b c

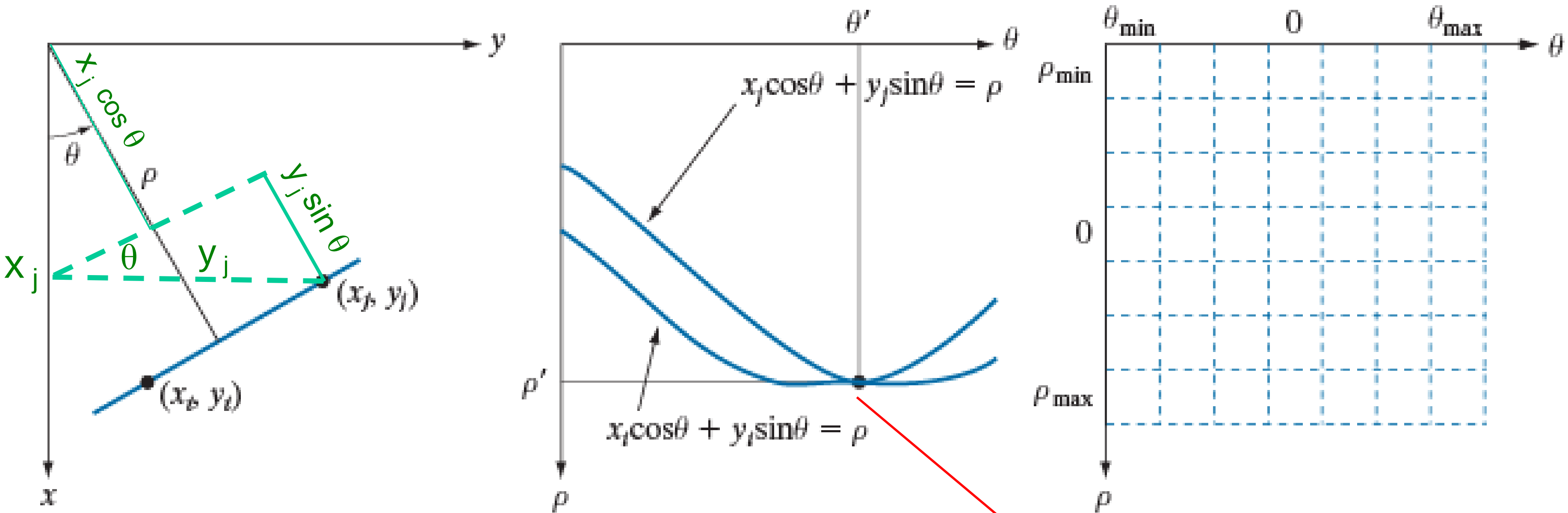
FIGURE 10.29 (a) (ρ, θ) parameterization of a line in the xy-plane. (b) Sinusoidal curves in the $\rho\theta$-plane; the point of intersection (ρ', θ') corresponds to the line passing through points (x_i, y_i) and (x_j, y_j) in the xy-plane. (c) Division of the $\rho\theta$-plane into accumulator cells.

- In (b), each sinusoidal curve represents the family of lines that pass through a particular point (x_k, y_k) in the xy-plane

Parameter Space

x-y plane

ρ-θ plane



At $\theta = 0$, $x_j \cos\theta + y_j \sin\theta = x_j = \rho$

At $\theta = 90$, $x_j \cos\theta + y_j \sin\theta = y_j = \rho$

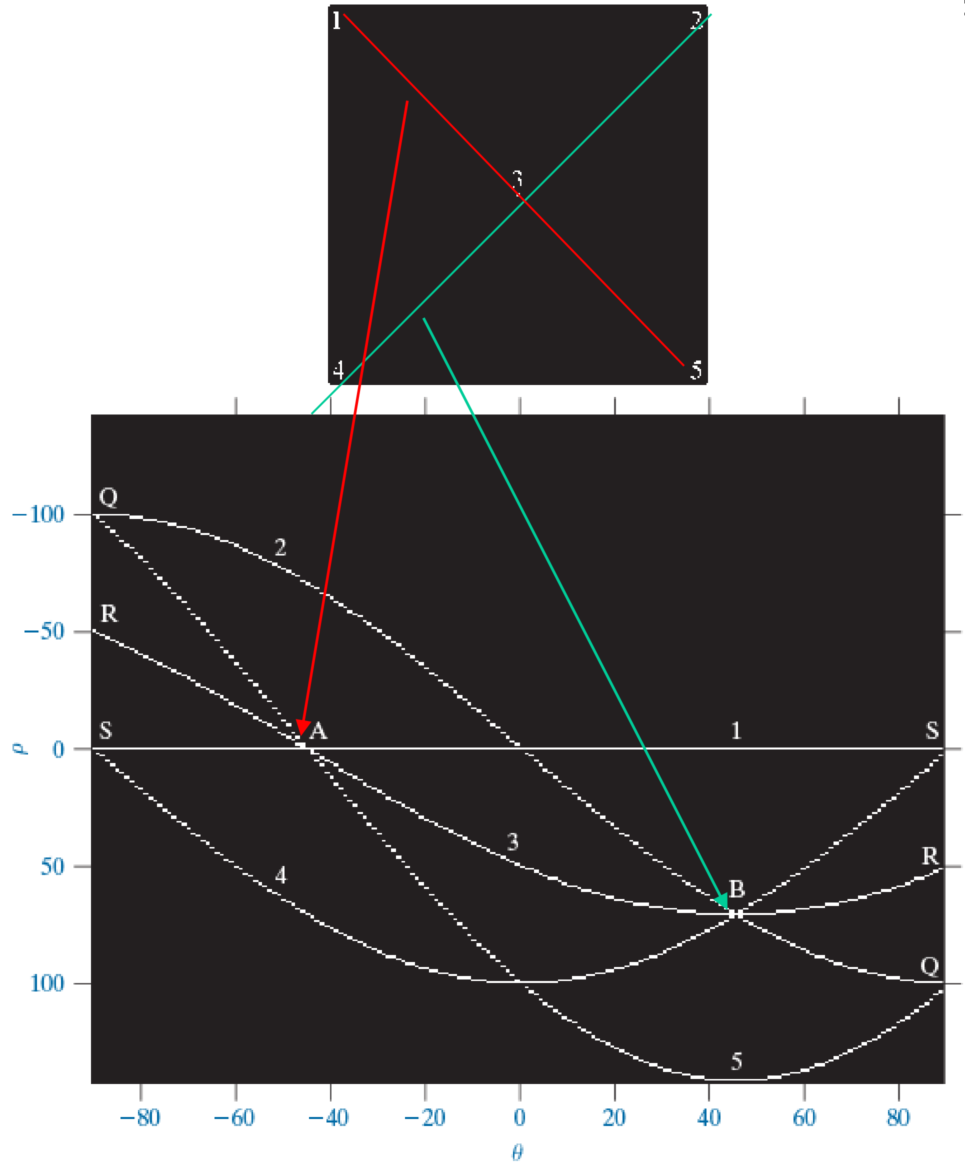
Both (x_i, y_i) and (x_j, y_j) corresponds to same ρ

a b c

FIGURE 10.29 (a) (ρ, θ) parameterization of a line in the *xy*-plane. (b) Sinusoidal curves in the $\rho\theta$-plane; the point of intersection (ρ', θ') 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.

- In (b), each sinusoidal curve represents the family of lines that pass throough a particular point (x_k, y_k) in the xy-plane

Figure 10.30



a
b

FIGURE 10.30
(a) Image of size 101 × 101 pixels, containing five white points (four in the corners and one in the center). (b) Corresponding parameter space.

Edge-linking Based on the Hough Transform

1. Obtain a binary edge image

2. Specify subdivisions in $\rho\theta - plane$

3. Examine the counts of the accumulator cells for high pixel concentrations

4. Examine the relationship between pixels in chosen cell

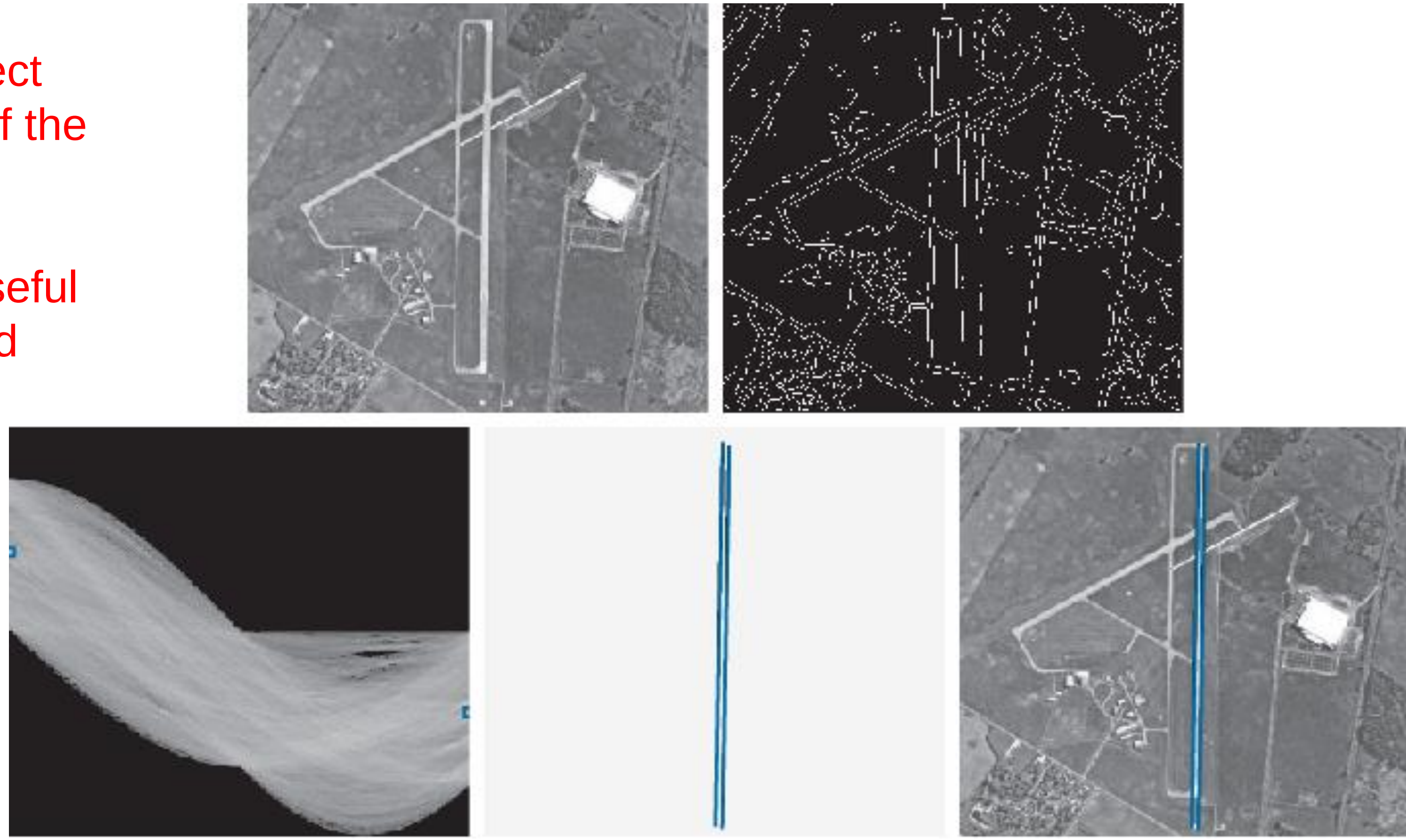
Goal: Extract the Two Edges of the Runway

Figure 10.31

Result of Hough Transform

Objective: Detect the two edges of the Runway

- Potentially useful for Automated Navigation



a b
c d e

FIGURE 10.31 (a) A 502 × 564 aerial image of an airport. (b) Edge map 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.

10.3 Thresholding

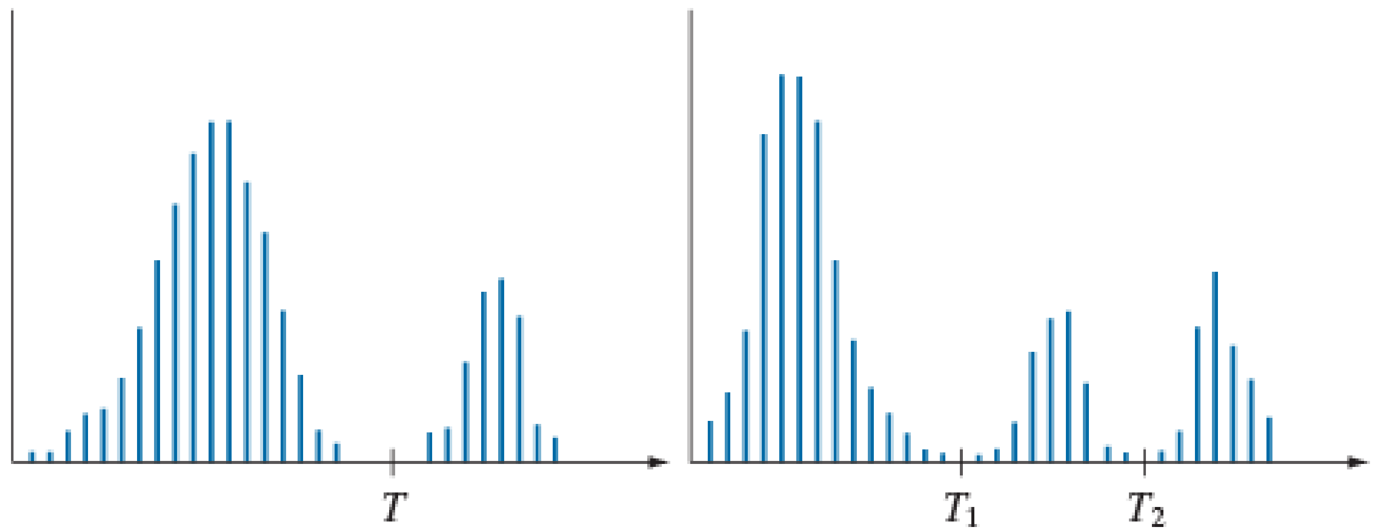
$$g(x,y)=\begin{cases}1 & \text{if } f(x,y)>T \quad \text{(object point)}\\ 0 & \text{if } f(x,y)\le T \quad \text{(background point)}\end{cases}$$

$$T:\text{Global thresholding}$$

Multiple thresholding

$$g(x,y)=\begin{cases}a & \text{if } f(x,y)>T_2\\ b & \text{if } T_1< f(x,y)\le T_2\\ c & \text{if } f(x,y)\le T_1\end{cases}$$

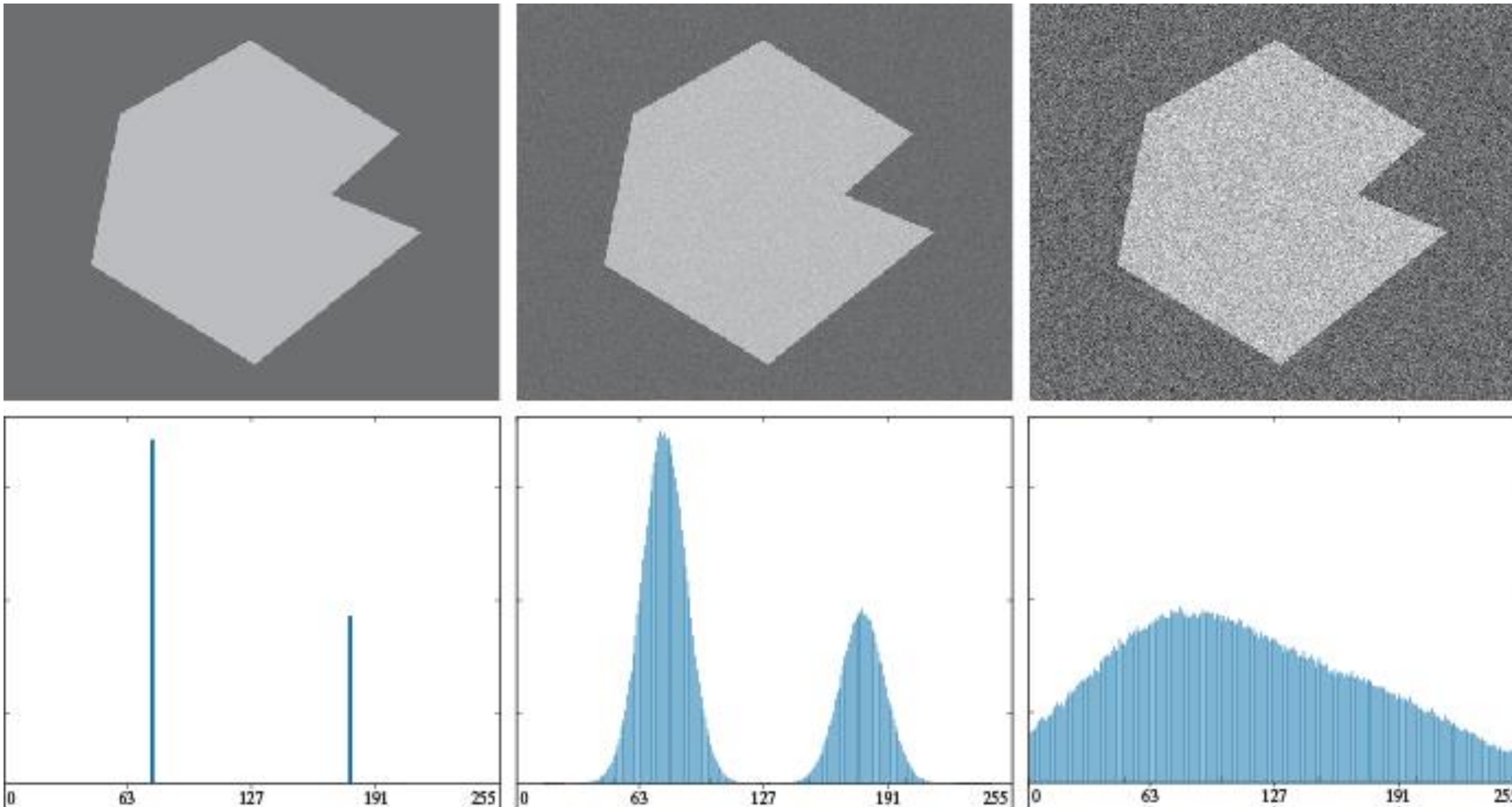
Partitioning Intensity Histograms



a b

FIGURE 10.32 Intensity histograms that can be partitioned (a) by a single threshold, and (b) by dual thresholds.

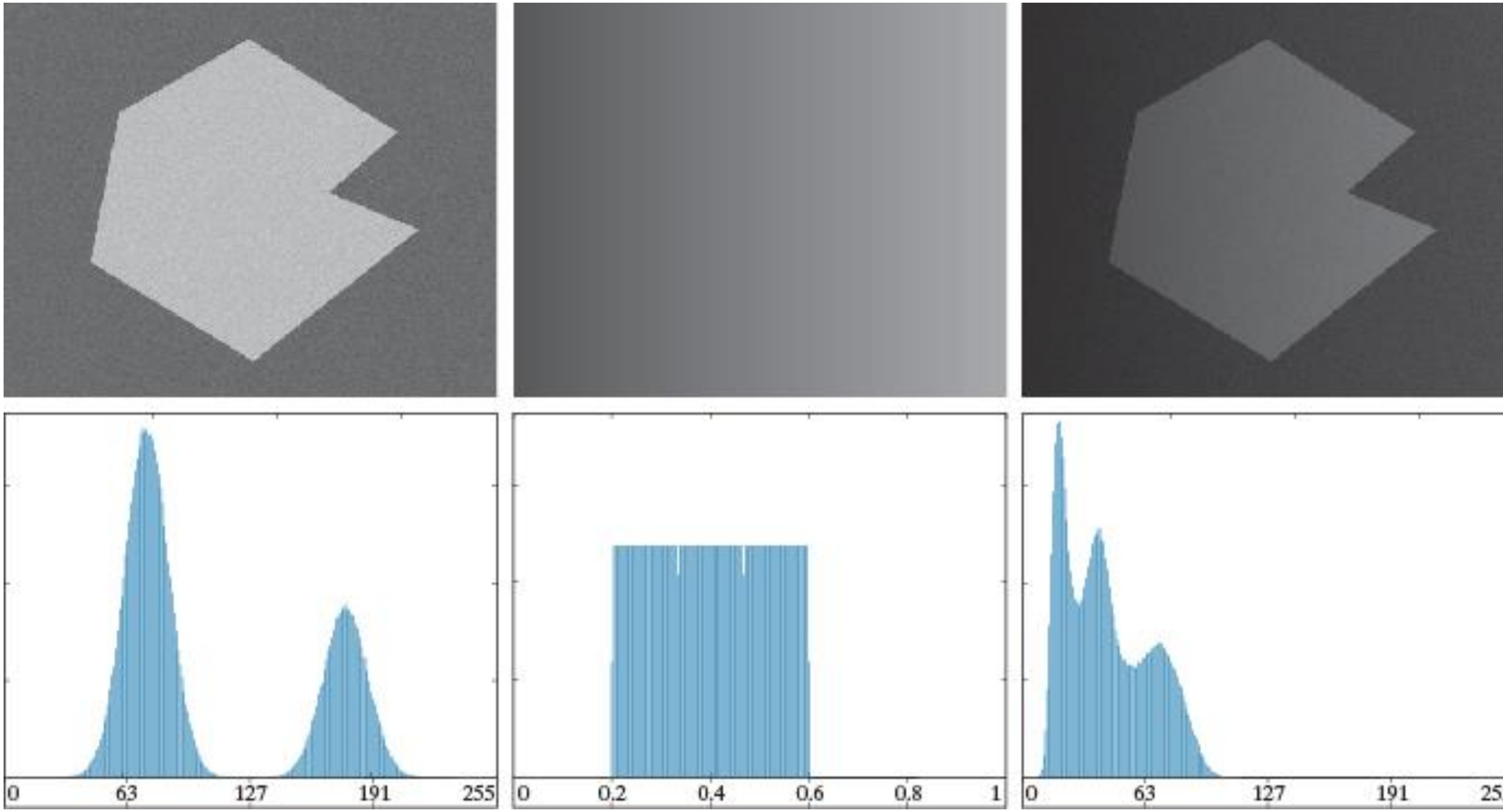
The Role of Noise in Image Thresholding



a b c
d e f

FIGURE 10.33 (a) Noiseless 8-bit image. (b) Image with additive Gaussian noise of mean 0 and standard deviation of 10 intensity levels. (c) Image with additive Gaussian noise of mean 0 and standard deviation of 50 intensity levels. (d) through (f) Corresponding histograms.

The Role of Illumination and Reflectance



a	b	c
d	e	f

FIGURE 10.34 (a) Noisy image. (b) Intensity ramp in the range [0.2, 0.6]. (c) Product of (a) and (b). (d) through (f) Corresponding histograms.

WRIGHT STATE UNIVERSITY

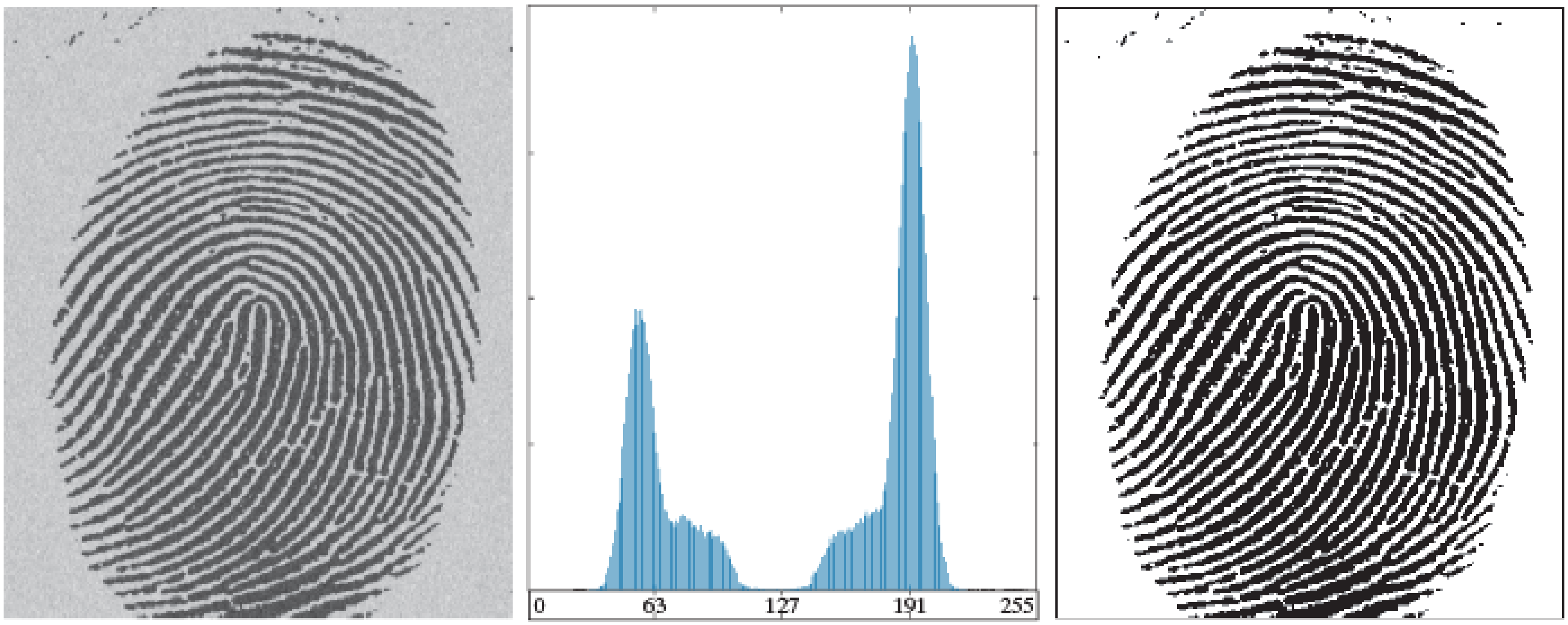
10.3.2 Basic Global Thresholding

1. Select an arbitrary estimate for the global threshold, T.
2. Segment the image using T. It will produce two groups of pixels: G_1 consisting of all pixels with intensity values > T and G_2 consisting of pixels with values $\leq T$.
3. Compute the average intensity values m_1 and m_2 for the pixels in G_1 and G_2, respectively.
4. Compute a new threshold value:

$$T = \frac{1}{2}(m_1 + m_2)$$

5. Repeat Steps 2 through 4 until the difference between values of T in successive iterations is smaller than a predefined parameter ΔT .

Result of Global Thresholding



a b c

FIGURE 10.35 (a) Noisy fingerprint. (b) Histogram. (c) Segmented result using a global threshold (thin image border added for clarity). (Original image courtesy of the National Institute of Standards and Technology.).

- Does not work well if the histogram does not have distinct peaks as shown above

10.3.3 Optimum Global Thresholding Using Otsu's Method ('79)

- Principle: Maximize the between-class variance

Let $\{0, 1, 2, ..., L\text{-}1\}$ denote the L distinct intensity levels in a digital image of size $M \times N$ pixels, and let n_i denote the number of pixels with intensity i.

Histogram: $$p_i = n_i / MN \quad \text{and} \quad \sum_{i=0}^{L-1} p_i = 1$$

k is a threshold value, $C_1 \rightarrow [0, k]$, $C_2 \rightarrow [k+1, L\text{-}1]$

$$P_1(k) = \sum_{i=0}^{k} p_i \quad \text{and} \quad P_2(k) = \sum_{i=k+1}^{L-1} p_i = 1 - P_1(k)$$

C_1 : Lower intensity levels

C_2 : Higher intensity levels

Optimum Global Thresholding Using Otsu's Method

- The mean intensity value of the pixels assigned to class C_1:

$$m_1(k) = \sum_{i=0}^{k} iP(i \mid C_1) = \frac{1}{P_1(k)} \sum_{i=0}^{k} ip_i$$

C_1 : Lower intensity levels

- The mean intensity value of the pixels assigned to class C_2:

$$m_2(k) = \sum_{i=k+1}^{L-1} iP(i \mid C_2) = \frac{1}{P_2(k)} \sum_{i=k+1}^{L-1} ip_i$$

C_2 : Higher intensity levels

$$P_1 m_1 + P_2 m_2 = m_G \quad \text{(Global mean value)}$$

Optimum Global Thresholding Using Otsu's Method

- **Performance Metric: Between-Class variance**

Between-class variance, σ_B^2 is defined as

$$\sigma_B^2 = P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2$$

$$= P_1 P_2 (m_1 - m_2)^2$$

$$= \frac{\left[m_G P_1 - m_1 P_1\right]^2}{P_1(1 - P_1)}$$

$$= \frac{\left[m_G P_1 - m\right]^2}{P_1(1 - P_1)}$$

Objective: Maximize between-class variance

Optimum Global Thresholding Using Otsu's Method

Objective: Maximize between-class variance

The optimum threshold is the value, k^* that maximizes $\sigma_B^2(k^*)$,

$$\sigma_B^2(k^*) = \max_{0 \le k \le L-1} \sigma_B^2(k)$$

Choose k for which $\sigma_B^2(k)$ *is maximum*

$$g(x,y) = \begin{cases} 1; & \text{if } f(x,y) > k^* \\ 0; & \text{if } f(x,y) \le k^* \end{cases}$$

Separability measure: $\eta = \dfrac{\sigma_B^2(k)}{\sigma_G^2}$

Otsu's Algorithm: Summary

1. Compute the normalized histogram of the input image. Denote the components of the histogram by p_i, $i = 0, 1, \ldots, L-1$.
2. Compute the cumulative sums, $P_1(k)$, for $k = 0, 1, \ldots, L-1$.
3. Compute the cumulative means, $m(k)$, for $k = 0, 1, \ldots, L-1$.
4. Compute the global intensity mean, m_G.
5. Compute the between-class variance, for $k = 0, 1, \ldots, L-1$.

$$\sigma_B^2(k) = \frac{\left[m_G P_1(k) - m\right]^2}{P_1(1 - P_1(k))}$$

Otsu's Algorithm: Summary

6. Obtain the Otsu's threshold, k^*.

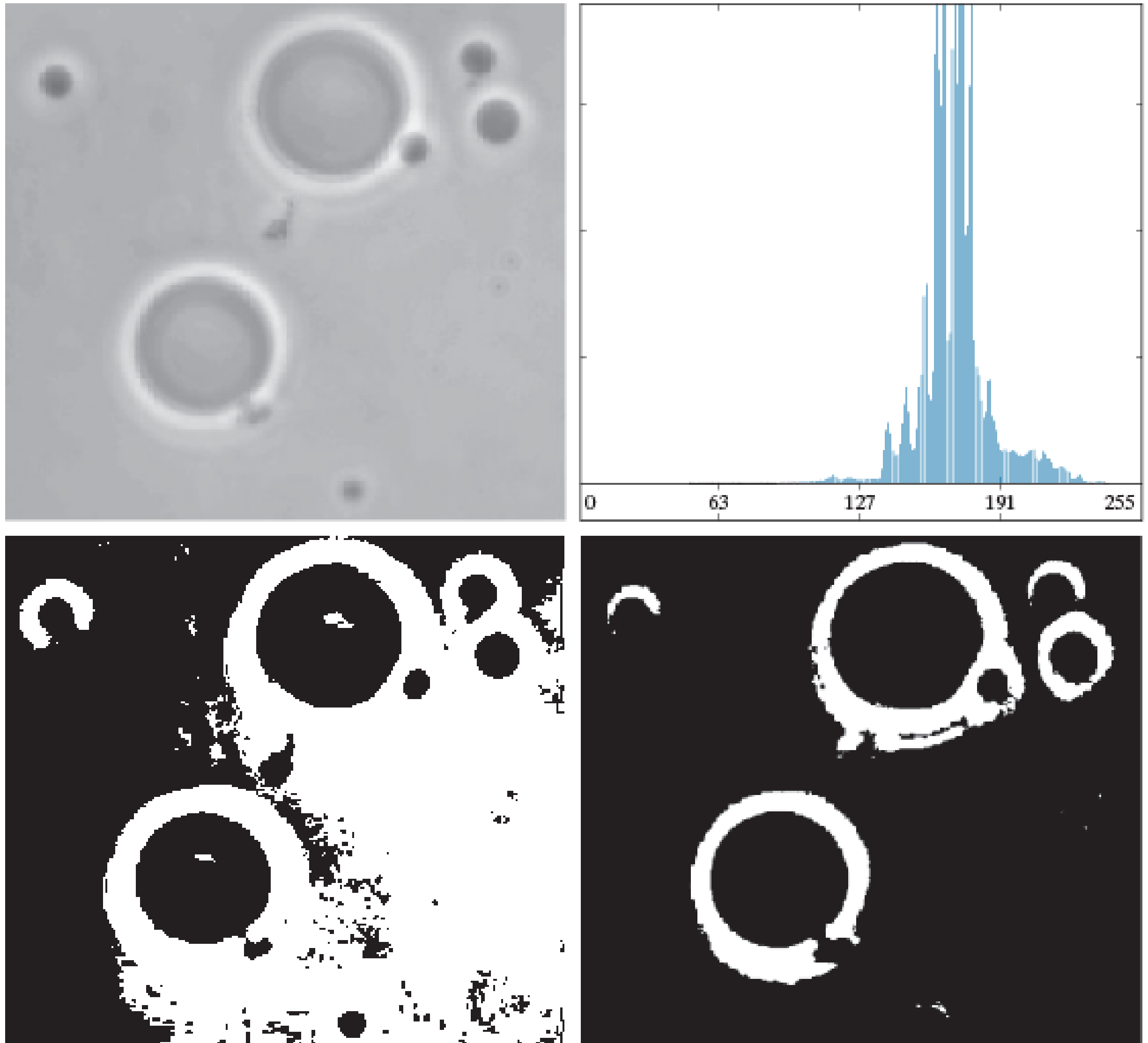
$$\sigma_B^2(k^*) = \max_{0 \le k \le L-1} \sigma_B^2(k)$$

7. Obtain the separability measure.

- Matlab in-built function

thresh = multithresh(A, N)

Results with Otsu's Method



a b
c d

FIGURE 10.36 (a) Original image. (b) Histogram (high peaks were clipped to highlight details in the lower values). (c) Segmentation result using the basic global algorithm from Section 10.3. (d) Result using Otsu's method. (Original image courtesy of Professor Daniel A. Hammer, the University of Pennsylvania.)

Using Image Smoothing to Improve Global Thresholding

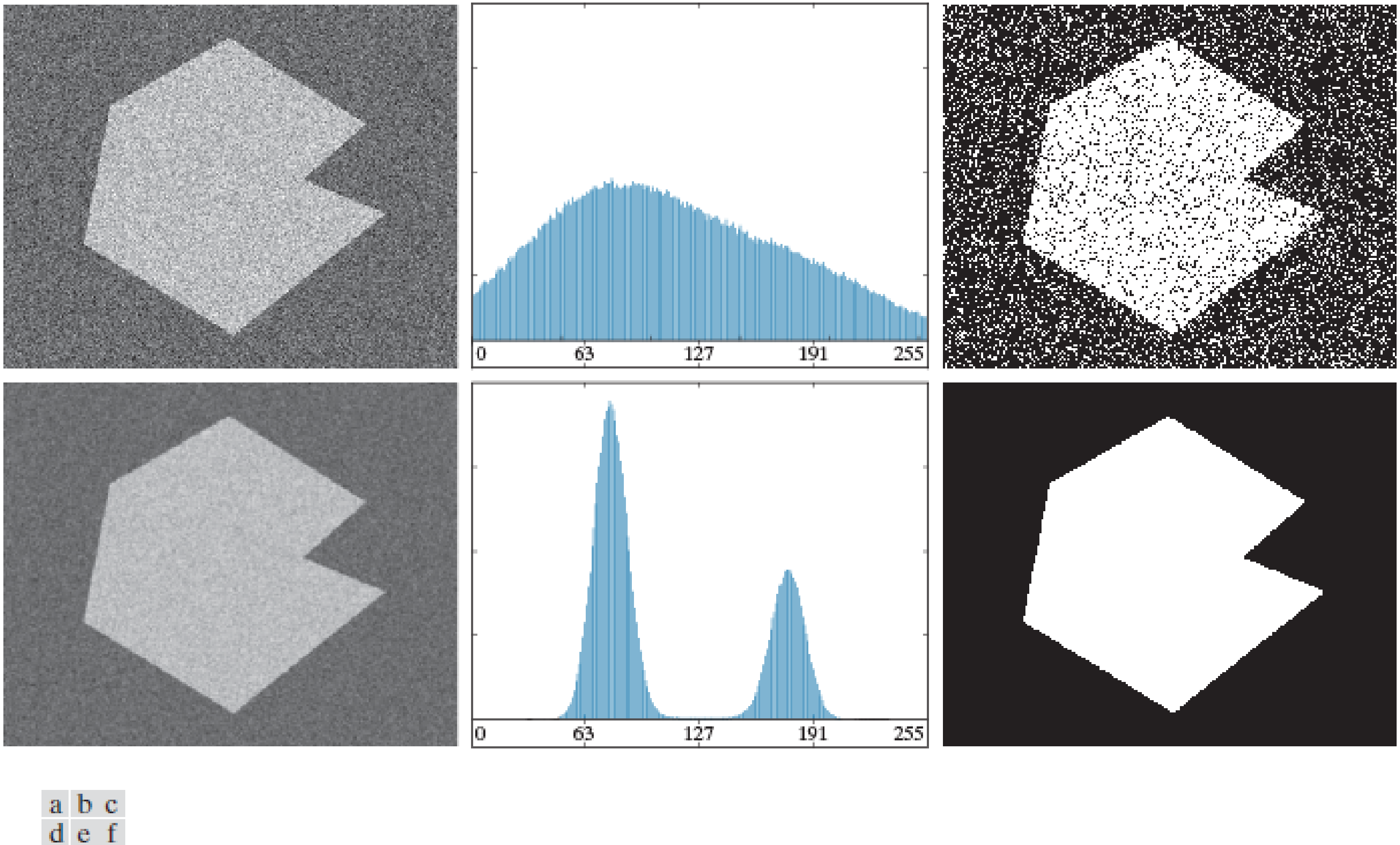
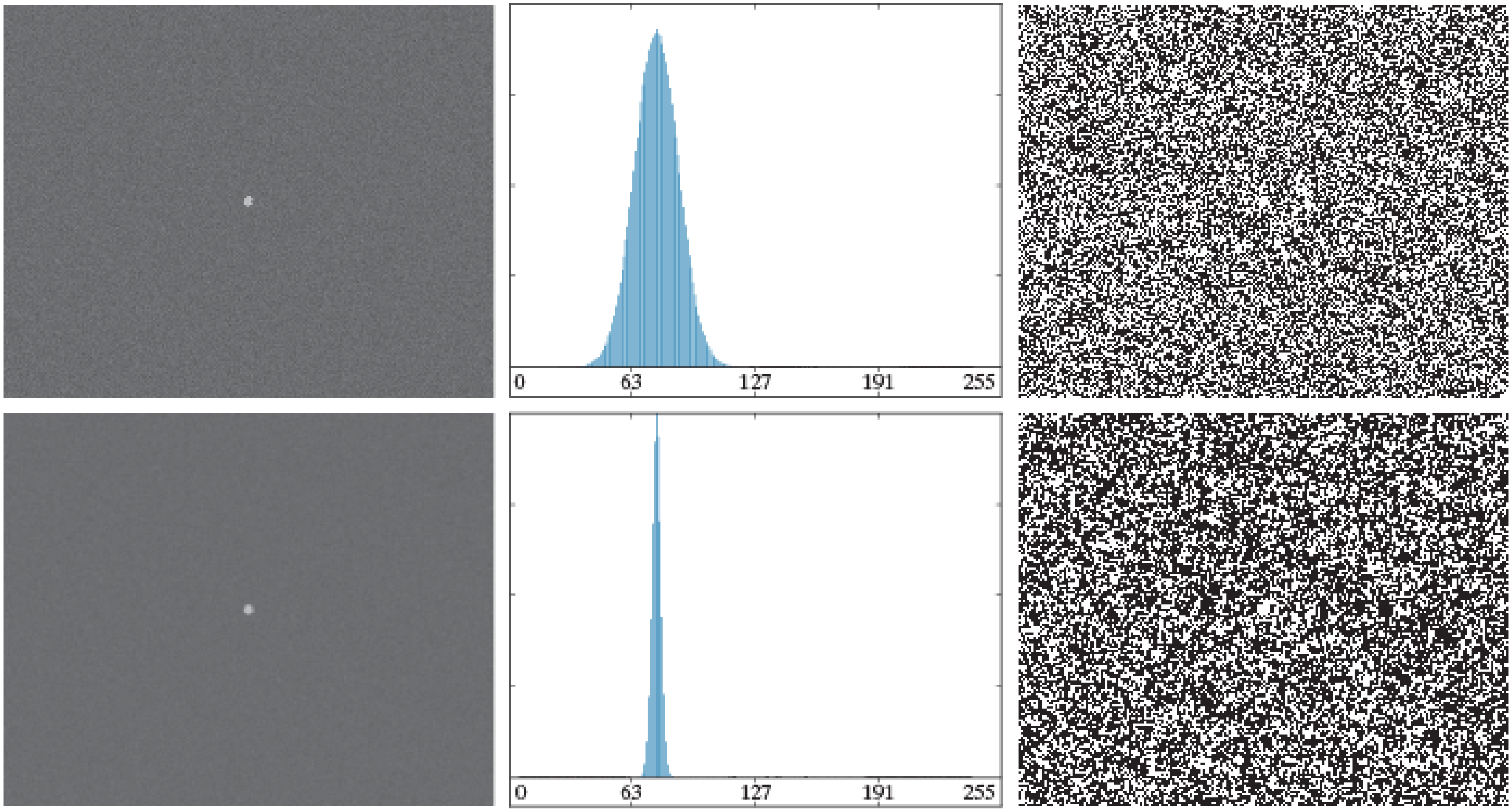


FIGURE 10.37 (a) Noisy image from Fig. 10.33(c) and (b) its histogram. (c) Result obtained using Otsu's method. (d) Noisy image smoothed using a 5×5 averaging kernel and (e) its histogram. (f) Result of thresholding using Otsu's method.

10.3.5 Using Edges to Improve Global Thresholding



Edges not used – Otsu's fail without and with Smoothing

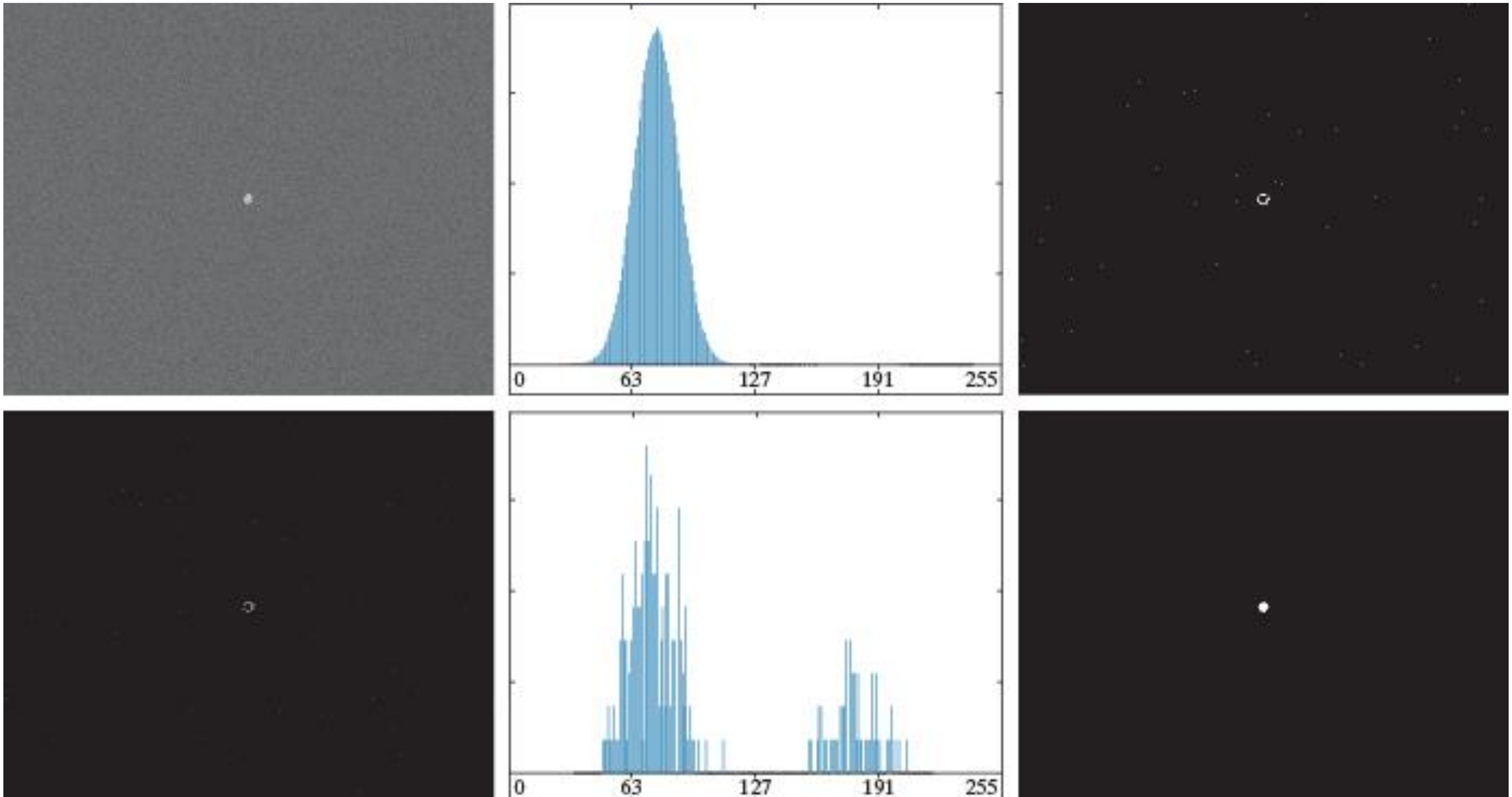
a b c
d e f

FIGURE 10.38 (a) Noisy image and (b) its histogram. (c) Result obtained using Otsu's method. (d) Noisy image smoothed using a 5×5 averaging kernel and (e) its histogram. (f) Result of thresholding using Otsu's method. Thresholding failed in both cases to extract the object of interest. (See Fig. 10.39 for a better solution.)

Solution: Use Edges to Improve Global Thresholding

1. Compute an edge image as either the magnitude of the gradient, or absolute value of Laplacian of $f(x, y)$
2. Specify a threshold value T
3. Threshold the image in Step-1 using threshold in Step-2 and produce a binary image $g_T(x, y)$, which is used as a mask image; and select pixels from $f(x, y)$ corresponding to “strong” edge pixels
4. Compute a histogram using only the chosen pixels in $f(x, y)$ that corresponds to 1-valued pixels in $g_T(x, y)$
5. Use the histogram from Step 4 to segment $f(x, y)$ globally, using Otsu’s method

Results with Otsu's Method + Edges



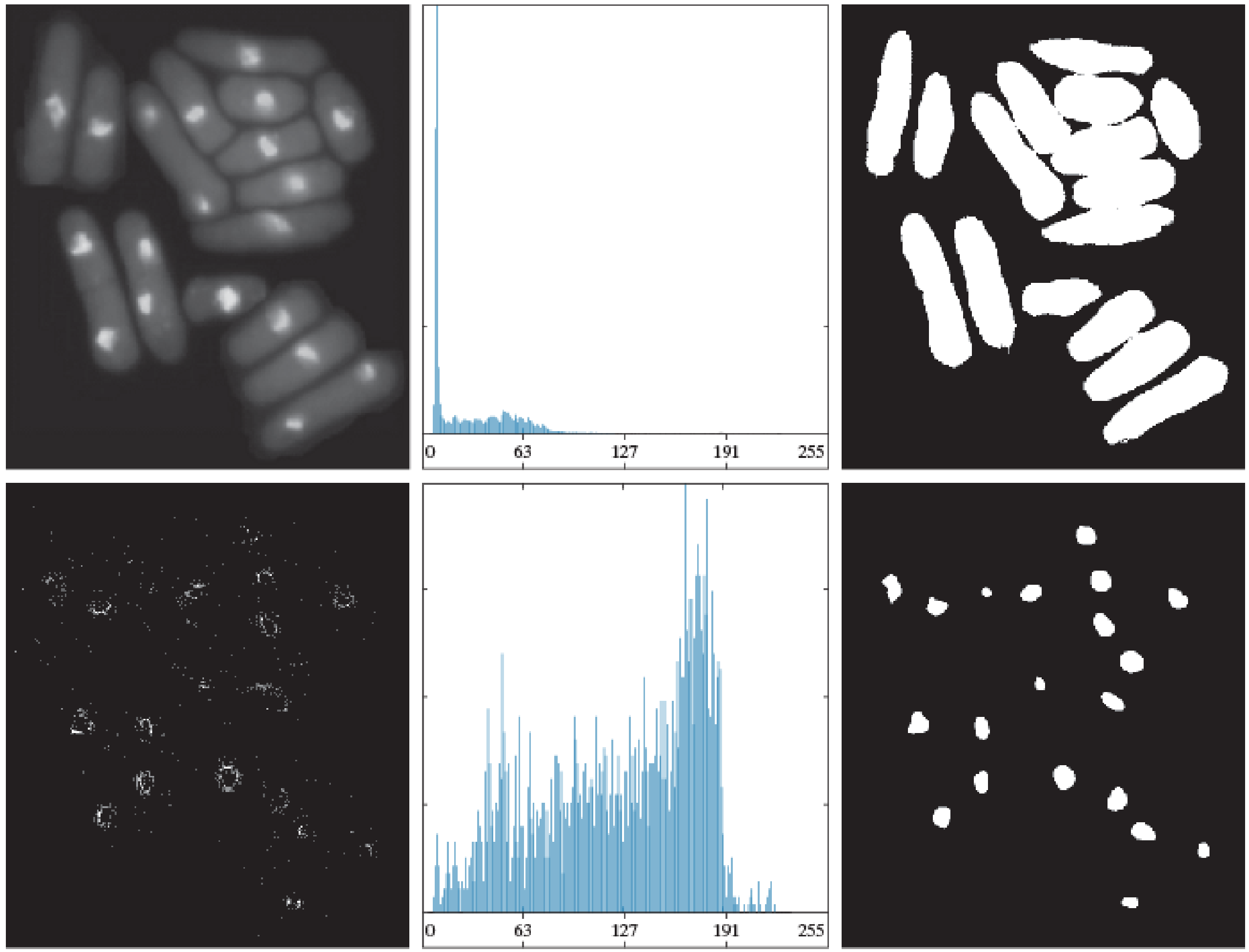
a b c
d e f

FIGURE 10.39 (a) Noisy image from Fig. 10.38(a) and (b) its histogram. (c) Mask image formed as the gradient magnitude image thresholded at the 99.7 percentile. (d) Image formed as the product of (a) and (c). (e) Histogram of the nonzero pixels in the image in (d). (f) Result of segmenting image (a) with the Otsu threshold based on the histogram in (e). The threshold was 134, which is approximately midway between the peaks in this histogram.

Results with Otsu's Method + Edges

Objective:

- Segment the Bright Spots



a b c
d e f

FIGURE 10.40 (a) Image of yeast cells. (b) Histogram of (a). (c) Segmentation of (a) with Otsu's method using the histogram in (b). (d) Mask image formed by thresholding the absolute Laplacian image. (e) Histogram of the nonzero pixels in the product of (a) and (d). (f) Original image thresholded using Otsu's method based on the histogram in (e). (Original image courtesy of Professor Susan L. Forsburg, University of Southern California.)

Results with Otsu's Method + Edges

Objective:

- Segment the Bright Spots

FIGURE 10.41 Image in Fig. 10.40(a) segmented using the same procedure as explained in Figs. 10.40(d) through (f), but using a lower value to threshold the absolute Laplacian image.

10.3.6 Multiple Thresholds

In the case of K classes, $C_1,\ C_2,\ \ldots,\ C_K$, the between-class variance is

$$\sigma_B^2 = \sum_{k=1}^{K} P_k \left(m_k - m_G\right)^2$$

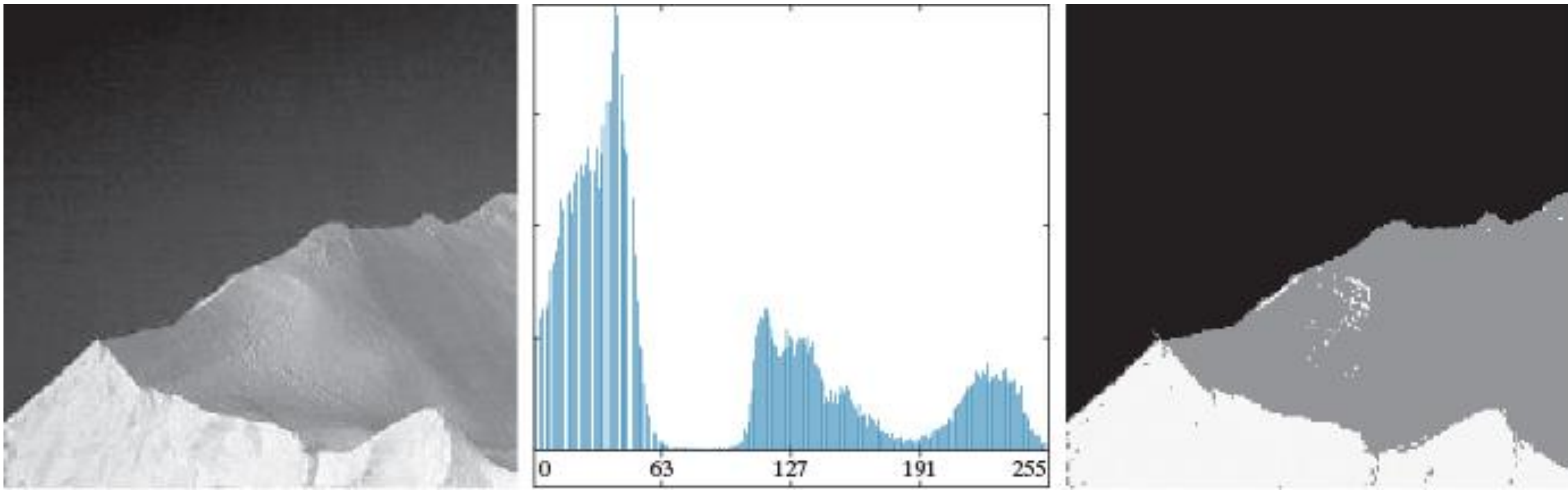
where $\quad P_k = \sum_{i \in C_k} p_i \quad$ and $\quad m_k = \frac{1}{P_k} \sum_{i \in C_k} i p_i$

The optimum threshold values, $k_1^*,\ k_2^*,\ \ldots, k_{K-1}^*$ that maximize

$$\sigma_B^2(k_1^*,\ k_2^*,\ \ldots, k_{K-1}^*) = \max_{0 \le k \le L-1} \sigma_B^2(k_1,\ k_2,\ \ldots, k_{K-1})$$

- Usually limited to three classes ($K = 3$) and two thresholds only
- For $K > 3$, other descriptors (e.g., color) is used

Result using Multiple (=2) Otsu Thresholds



a b c

FIGURE 10.42 (a) Image of an iceberg. (b) Histogram. (c) Image segmented into three regions using dual Otsu thresholds. (Original image courtesy of NOAA.)

- Results with three classes (K=3) and two thresholds
- $k_1^* = 80$ and $k_2^* = 177$

10.3.7 Variable Thresholding: Image Partitioning

Potential Problems:

- Noise and Nonuniform illumination may limit global thresholding performance
- Smoothing or using Edges may be impractical or ineffective

Solution: Use Variable Thresholding

- Subdivide an image into non-overlapping rectangles
- The rectangles are chosen small enough so that the illumination of each is approximately uniform.

Results using Otsu's + Partitioning

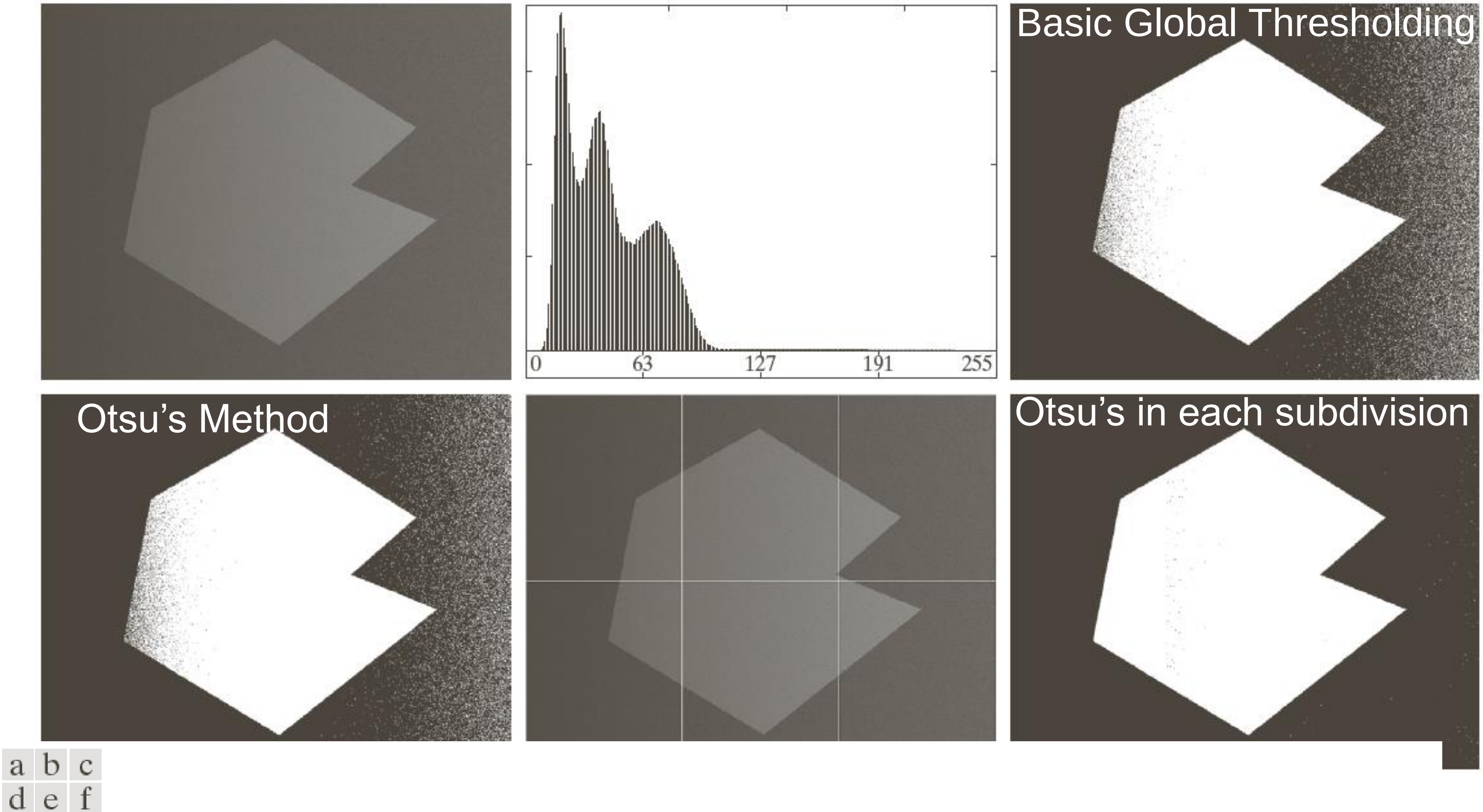


FIGURE 10.46 (a) Noisy, shaded image and (b) its histogram. (c) Segmentation of (a) using the iterative global algorithm from Section 10.3.2. (d) Result obtained using Otsu's method. (e) Image subdivided into six subimages. (f) Result of applying Otsu's method to each subimage individually.

3rd Edition

Histogram of Subimages

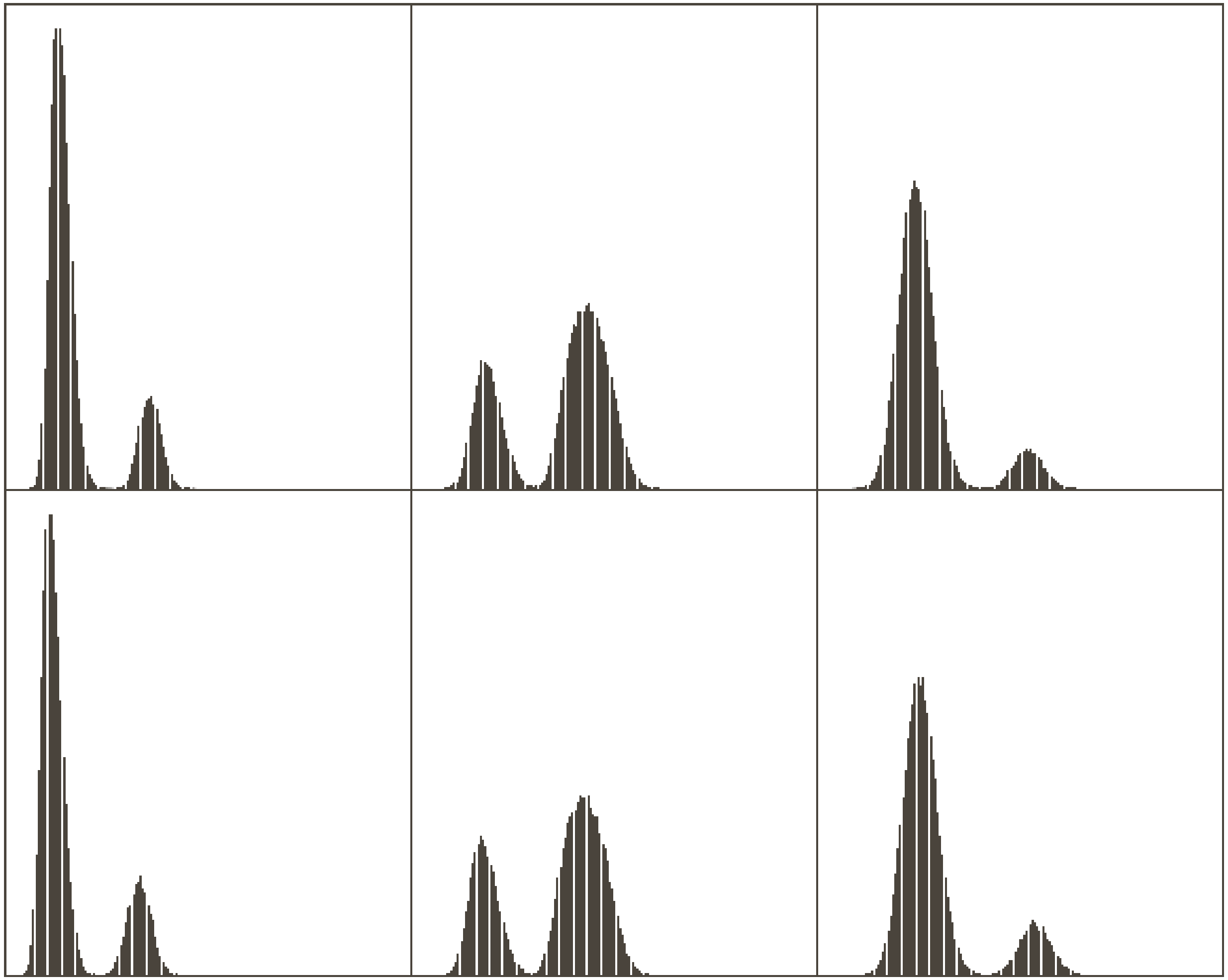


FIGURE 10.47 Histograms of the six subimages in Fig. 10.46(e).

- Each subimage has bimodal histograms with deep valleys
- Conducive to effective thresholding

10.3.7 Variable Thresholding Based on Local Image Properties

• General Approach: Compute threshold at every point in the image based on one or more properties in the neighborhood

Let σ_{xy} and m_{xy} denote the standard deviation and mean value of the set of pixels contained in a neighborhood S_{xy}, centered at coordinates (x, y) in an image.

- Possible local thresholds,

$$T_{xy} = a\sigma_{xy} + bm_{xy} \quad (a, b: \text{ Non-negative constants})$$

OR, if the background is nearly constant, use

$$T_{xy} = a\sigma_{xy} + bm_G \quad (m_G: \text{ Global Image Mean})$$

$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T_{xy} \\ 0 & \text{if } f(x,y) \le T_{xy} \end{cases}$$

• Computed for each location (x, y)

Variable Thresholding Based on Local Image Properties

• More effective Local thresholding achieved using predicates based on parameters computed in the neighborhood of (x, y)

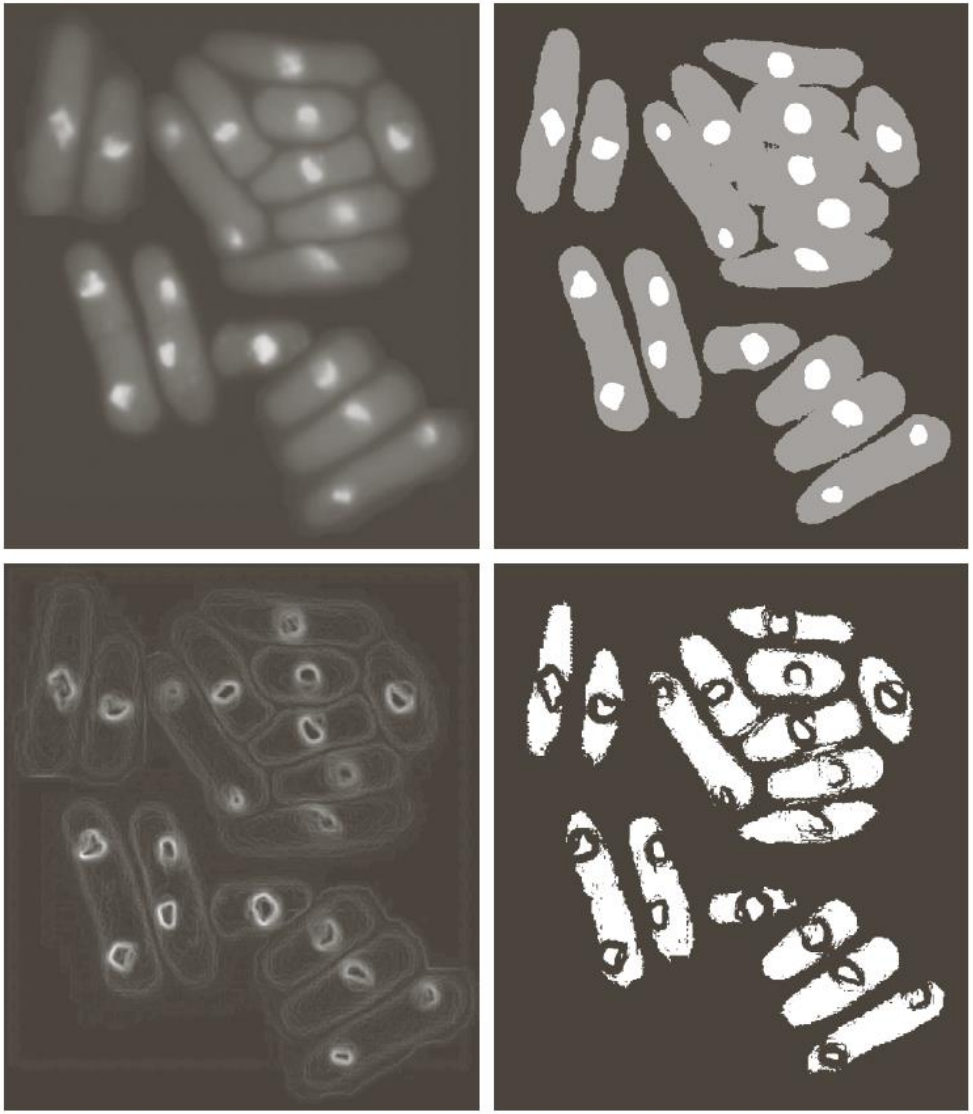
A modified thresholding

$$g(x,y)=\begin{cases}1 & \text{if } Q(\text{local parameters}) \text{ is true}\\ 0 & \text{otherwise}\end{cases}$$

e.g.,

$$Q(\sigma_{xy}, m_{xy})=\begin{cases}\text{true} & \text{if } f(x,y)>a\sigma_{xy} \text{ AND } f(x,y)>bm_{xy}\\ \text{false} & \text{otherwise}\end{cases}$$

Variable Thresholding Based on Local Image Properties



a b
c d

FIGURE 10.43 (a) Image from Fig. 10.40. (b) Image segmented using the dual thresholding approach given by Eq. (10-76). (c) Image of local standard deviations. (d) Result obtained using local thresholding.

- Right side not segmented properly

- Neighborhood size: 3x3

$$a = 30$$
$$b = 1.5$$
$$m_{xy} = m_G$$

Variable Thresholding Using Moving Averages

- Thresholding based on moving averages (along scan-lines) works well when objects are small with respect to the image size

- Quite useful in document processing

- The scanning (moving) typically is carried out line-by-line in zigzag pattern to reduce illumination bias

Variable Thresholding Using Moving Averages

Let z_{k+1} denote the intensity of the point encountered in the scanning sequence at step $k+1$. The **moving average (mean intensity)** at this new point is given *recursively* by

$$m(k+1) = \frac{1}{n} \sum_{i=k+2-n}^{k+1} z_i = m(k) + \frac{1}{n}(z_{k+1} - z_{k-n})$$

where n denotes the number of points used in computing the average and $m(1) = z_1 / n$, the border of the image were padded with n-1 zeros.

Variable Thresholding Using Moving Averages

$$g(x,y)=\begin{cases}1 & \text{if } f(x,y) > T_{xy} \\ 0 & \text{if } f(x,y) \le T_{xy}\end{cases}$$

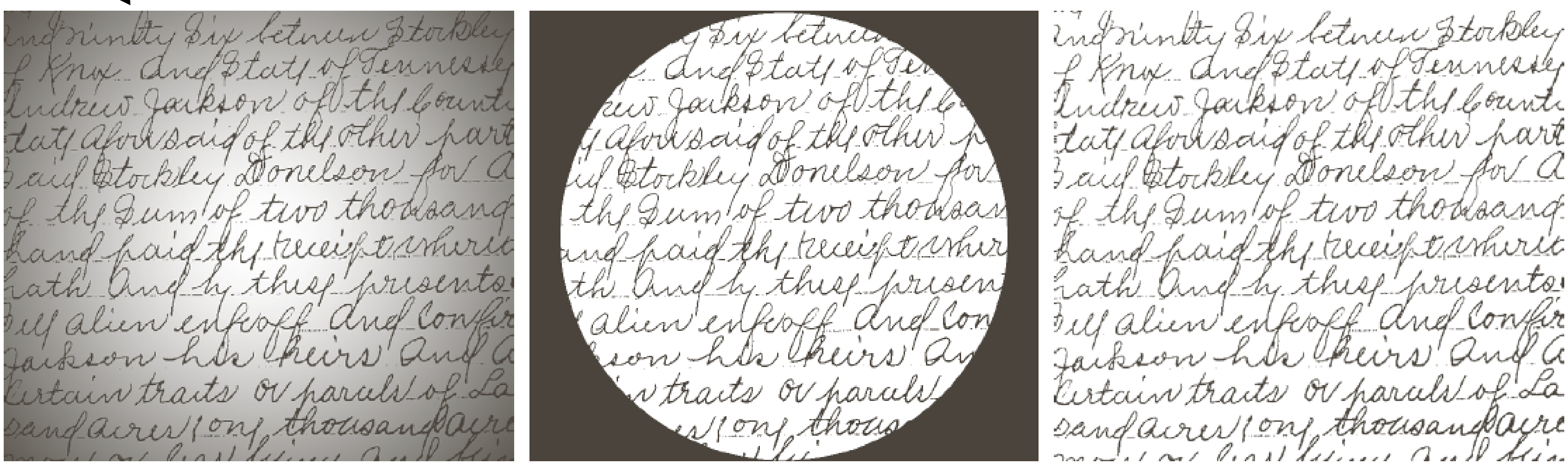
$$T_{xy} = bm_{xy}$$

b : Constant (user selected)

Result using Otsu's + Local Thresholding using Moving Averages

- Could be effect of using photographic flash

n = 20
b=0.5

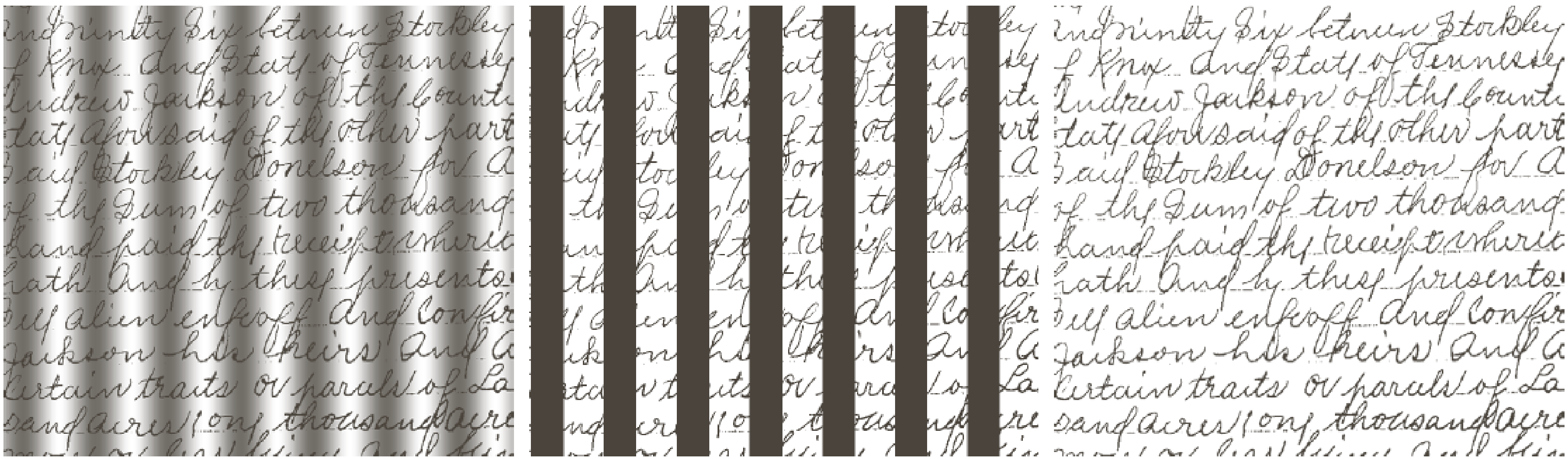


a b c

FIGURE 10.44 (a) Text image corrupted by spot shading. (b) Result of global thresholding using Otsu's method. (c) Result of local thresholding using moving averages.

Result using Otsu's + Local Thresholding using Moving Averages

• Sinusoidal illumination variation due to improper grounding



a b c

FIGURE 10.45 (a) Text image corrupted by sinusoidal shading. (b) Result of global thresholding using Otsu's method. (c) Result of local thresholding using moving averages..

10.4 Region-Based Segmentation

- Region Growing

1. Region growing is a procedure that groups pixels or sub-regions into larger regions.

2. The simplest of these approaches is ***pixel aggregation***, which starts with a set of "seed" points (user-selected) and from these grows regions by appending to each seed points those **neighboring pixels** that have **similar properties** (such as gray level, texture, color, shape).

3. Region growing based techniques are better than the edge-based techniques in noisy images where edges are difficult to detect.

Region-Based Segmentation

Example: Region Growing based on 4 or 8-connectivity

$f(x, y)$: input image array

$S(x, y)$: seed array containing 1s (seeds) and 0s

$Q(x, y)$: predicate

	$(x\text{-}1,y)$	
$(x,y\text{-}1)$	(x,y)	$(x,y\text{+}1)$
	$(x\text{+}1,y)$	

$(x\text{-}1,y\text{-}1)$	$(x\text{-}1,y)$	$(x\text{-}1,y\text{+}1)$
$(x,y\text{-}1)$	(x,y)	$(x,y\text{+}1)$
$(x\text{+}1,y\text{-}1)$	$(x\text{+}1,y)$	$(x\text{+}1,y\text{+}1)$

Recall: 4-Connectivity

8-connectivity

(Section 2.5)

Region Growing based on 8-connectivity

1. Find all connected components in $S(x,y)$ and erode each connected components to one pixel; label all such pixels found as 1. All other pixels in S are labeled 0.
2. Form an image f_Q such that, at a pair of coordinates (x,y), let $f_Q(x,y)=1$ if predicate-Q is satisfied, otherwise $f_Q(x,y)=0$.
3. Let g be an image formed by appending to each seed point in S all the 1-value points in f_Q that are 8-connected to that seed point.
4. Label each connencted component in g with a different region label. This is the segmented image obtained by region growing.

Region Growing based on x-Connectivity

- Predicate used

$$Q = \begin{cases} \text{TRUE} & \text{if the absolute difference of the intensities} \\ & \quad \text{between the seed and the pixel at } (x, y) \text{ is } \leq T \\ \text{FALSE} & \text{otherwise} \end{cases}$$

Region Growing Example

Suppose that we have the image given below.

(a) Use the region growing idea to segment the object. The seed for the object is the center of the image. Region is grown in horizontal and vertical directions, and when the difference between two pixel values is less than or equal to 5.

Table 1: Show the result of Part (a) on this figure.

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	<u>60</u>	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

Region Growing Example: 4-Connectivity

- Seed: Center pixel of the image
- Region grown in horizontal and vertical directions
- Condition: Difference between two pixel values ≤ 5

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	60	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

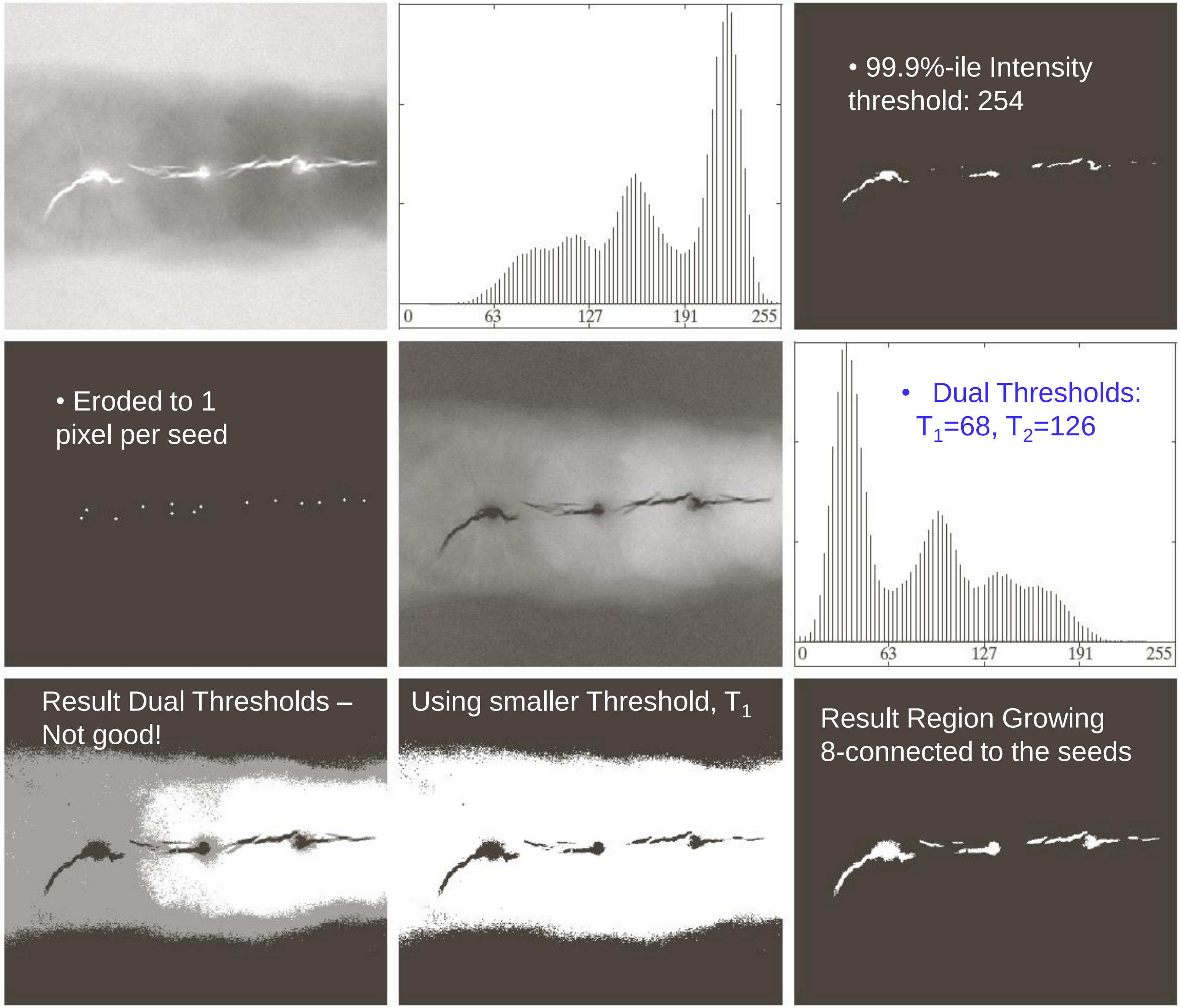
4-connectivity

Region Growing Example: 8-Connectivity

- Seed: Center pixel of the image
- Region grown in horizontal, vertical and diagonal directions
- Condition: Difference between two pixel values ≤ 5

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	60	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

8-connectivity



a b c
d e f
g h i

Figure 10.46 (a) X-ray image of a defective weld. (b) Histogram. (c) Initial seed image. (d) Final seed image (the points were enlarged for clarity). (e) Absolute value of the difference between the seed value (255) and (a). (f) Histogram of (e). (g) Difference image thresholded using dual thresholds. (h) Difference image thresholded with the smallest of the dual thresholds. (i) Segmentation result obtained by region growing. (Original image courtesy of X-TEK Systems, Ltd.)

10.5 Region Segmentation using Clustering (New in 4th Edition)

- Region Segmentation using Clustering
  - Objective: Partition a set Q of observations into a specified number k, of clusters.
  - In k -means Clustering, each observation is assigned to the cluster with the closest mean
  - Each mean is called the prototype of its cluster
  - k -means Clustering algorithm is iterative that successively refines the means until convergence

Region Segmentation using k-Means Clustering

- $[\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_Q]$: Vectors of observations (pixels, samples)
- Each vector $\mathbf{z} = [z_1, z_2, \cdots, z_n]^T$: 1-D intensity (gray) or 3-D RGB or multispectral band values at each pixel
- Objective of k-means clustering: Partition Q observations into $k\,(\leq Q)$ disjoint cluster sets:

  $C = \{C_1, C_2, \cdots, C_k\}$ such that $\underset{C}{\arg\min}\left(\sum_{i=1}^{k}\sum_{z \in C_i}\|\mathbf{z}-\mathbf{m}_i\|^2\right)$

  $\mathbf{m}_i$: Mean-vector or centroid of the samples in C_i
- Unfortunately, finding minimum is NP-hard problem with no practical solution. Approximation is used.

Implementation Steps for k-Means Clustering

- Given: $\left[\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_Q\right]$ and a specified value k

1. Specify an initial set of $\mathbf{m}_i(1); i = 1, 2, \ldots, k.$
2. Assign each sample to cluster with closest mean:

$$\mathbf{z}_q \to C_i, \text{ if } \left\|\mathbf{z}_q - \mathbf{m}_i\right\|^2 < \left\|\mathbf{z}_q - \mathbf{m}_j\right\|^2 \quad j = 1, 2, \ldots k\,(j \neq i);\; q = 1, 2, \ldots, Q$$

3. Update the Cluster Means: $\mathbf{m}_i = \frac{1}{|C_i|} \sum_{\mathbf{z} \in C_i} \mathbf{z} \quad i = 1, 2, \ldots, k$
4. Compute the residual error, E → The sum of Euclidean Norms of the differences between the mean vectors in the current and previous steps. If $E < T$ (a specified threshold) then STOP, else go back to Step-2.

Image segmentation with k -Means Clustering

a b

FIGURE 10.49 (a) Image of size 688 × 688 pixels. (b) Image segmented using the k-means algorithm with $k = 3$.

- Entire segmentation was done using clustering of Intensity only
- In general, the power of k-means clustering to discriminate between regions increases as the number of components in the z-vector increases

10.5 Segmentation using Morphological Watersheds

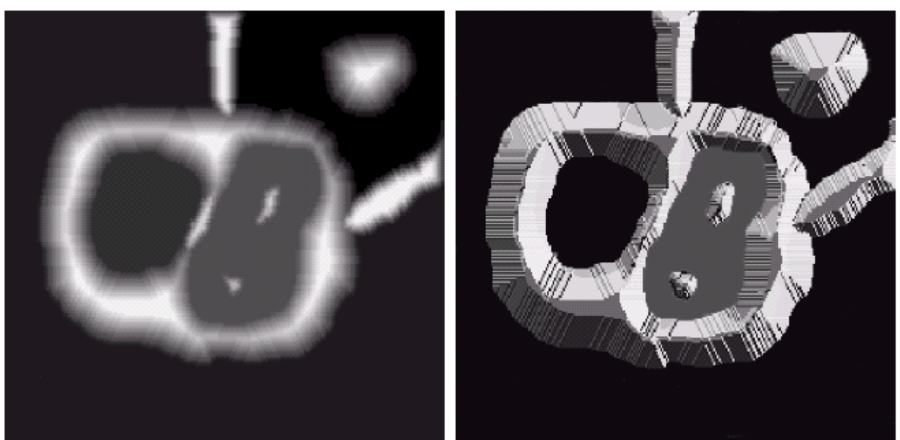
- Motivation for Morphological Watershed:
  - Embodies many of the segmentation concepts used
    (a) Edge Detection
    (b) Thresholding and
    (c) Region Growing
  - Often produces more stable segmentation results, including connected segmentation boundaries
  - Background: Visualize an image in 3-D: Two spatial coordinates vs. intensity in the 3rd dimension

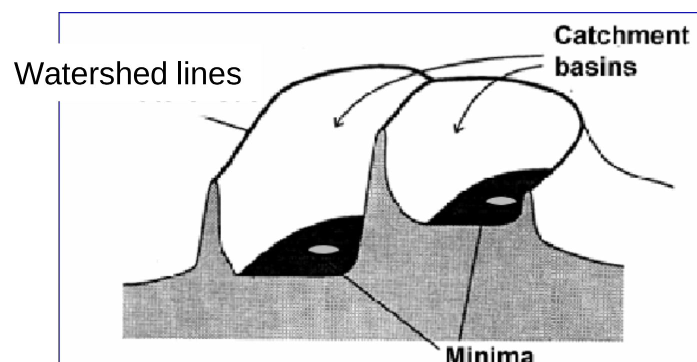
10.7 Segmentation using Morphological Watersheds

- Three types of points in 3D topographic interpretation:
  - Points belonging to a regional minimum
  - Points at which a drop of water would certainly fall to a single minimum.

    → *Catchment basin* or *watershed* of that minimum
  - Points at which a drop of water would be equally likely to fall to more than one minimum.

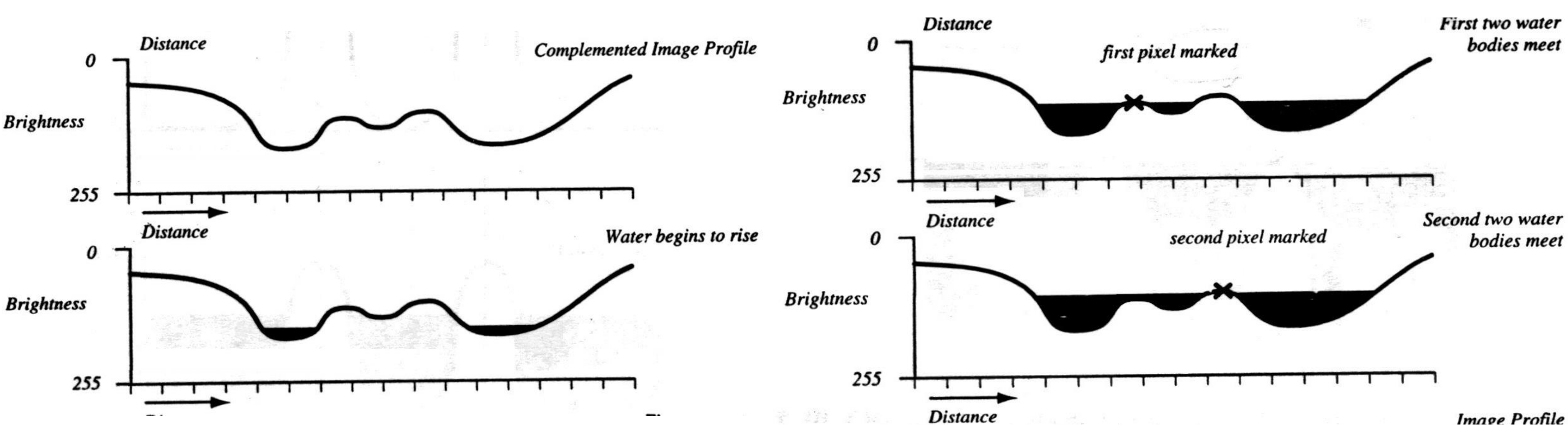
    → The *divide lines* or *watershed lines*



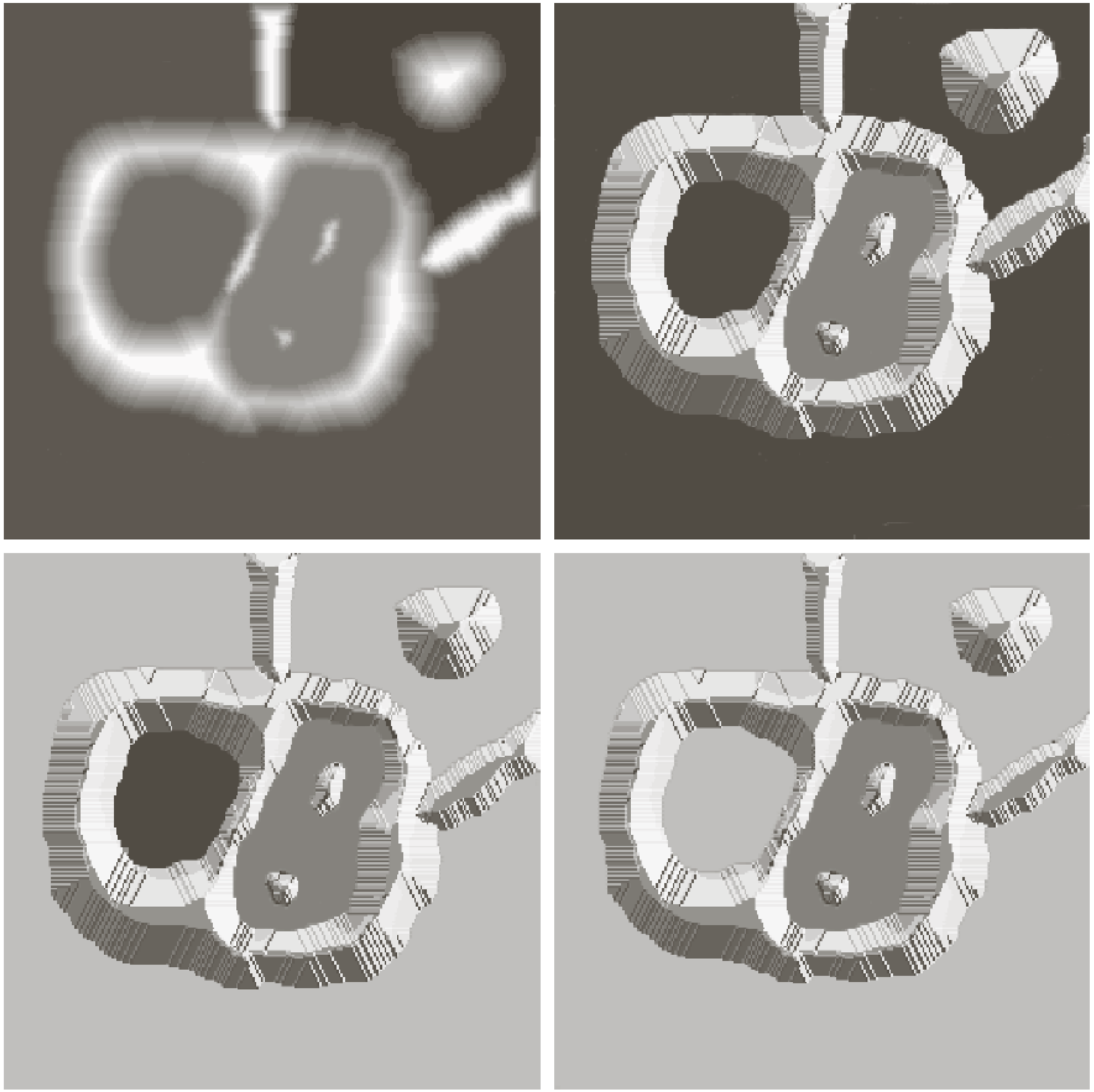


Watershed Segmentation: Example

- The objective is to find watershed lines.
- The idea is simple:
  - Suppose that a hole is punched in each regional minimum and that the entire topography is flooded from below by letting water rise through the holes at a uniform rate.
  - When rising water in distinct catchment basins is about to merge, a dam is built to prevent merging. These dam boundaries correspond to the watershed lines.



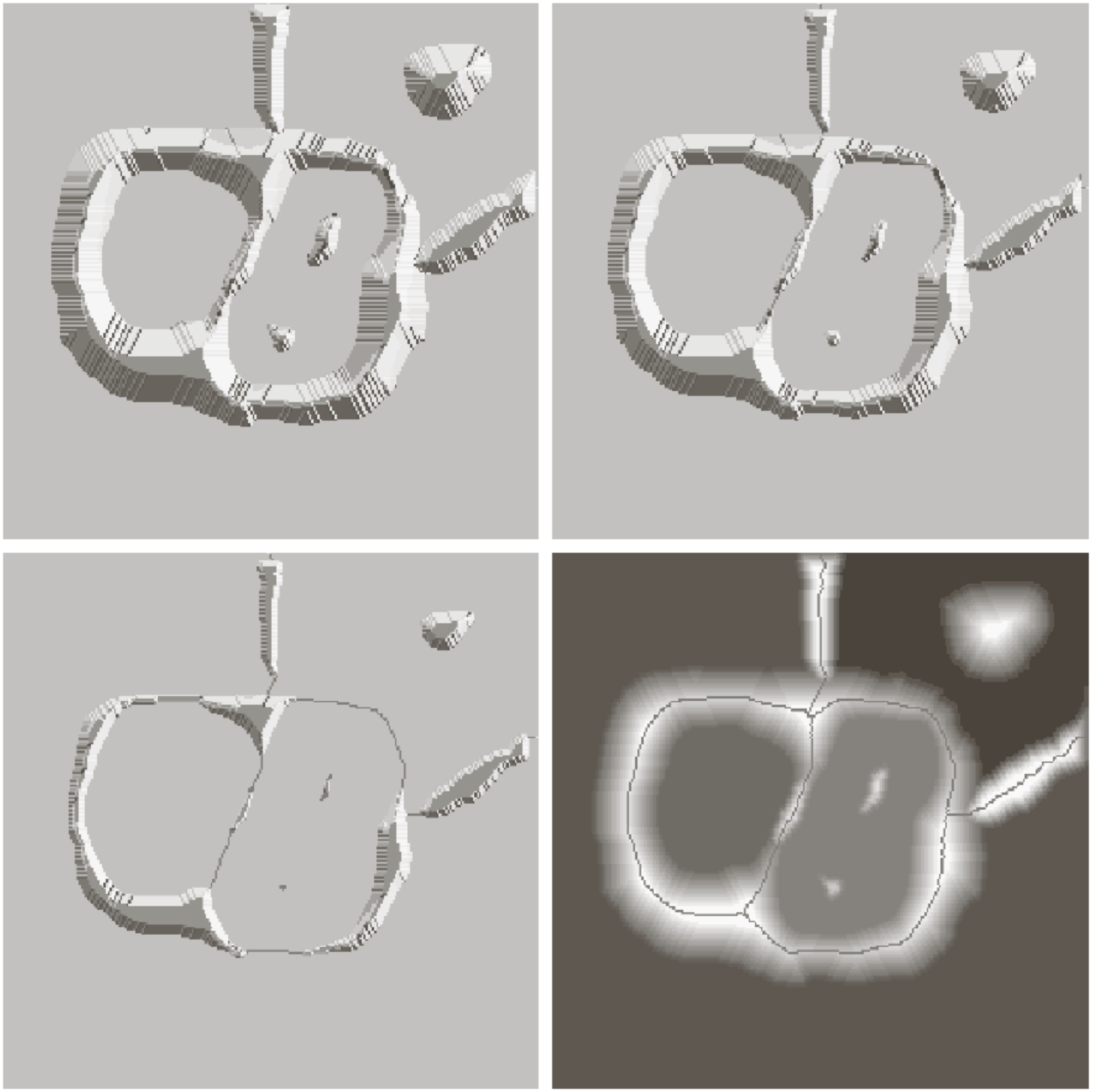
Watershed Segmentation: Example



a c
b d

FIGURE 10.60 (a) Original image. (b) Topographic view. Only the background is *black*. The basin on the left is slightly lighter than black. (c) and (d) Two stages of flooding. All constant dark values of gray are intensities in the original image. Only constant *light gray* represents "water." (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.) (*Continued on next page.*)

Watershed Segmentation: Example



e f
g h

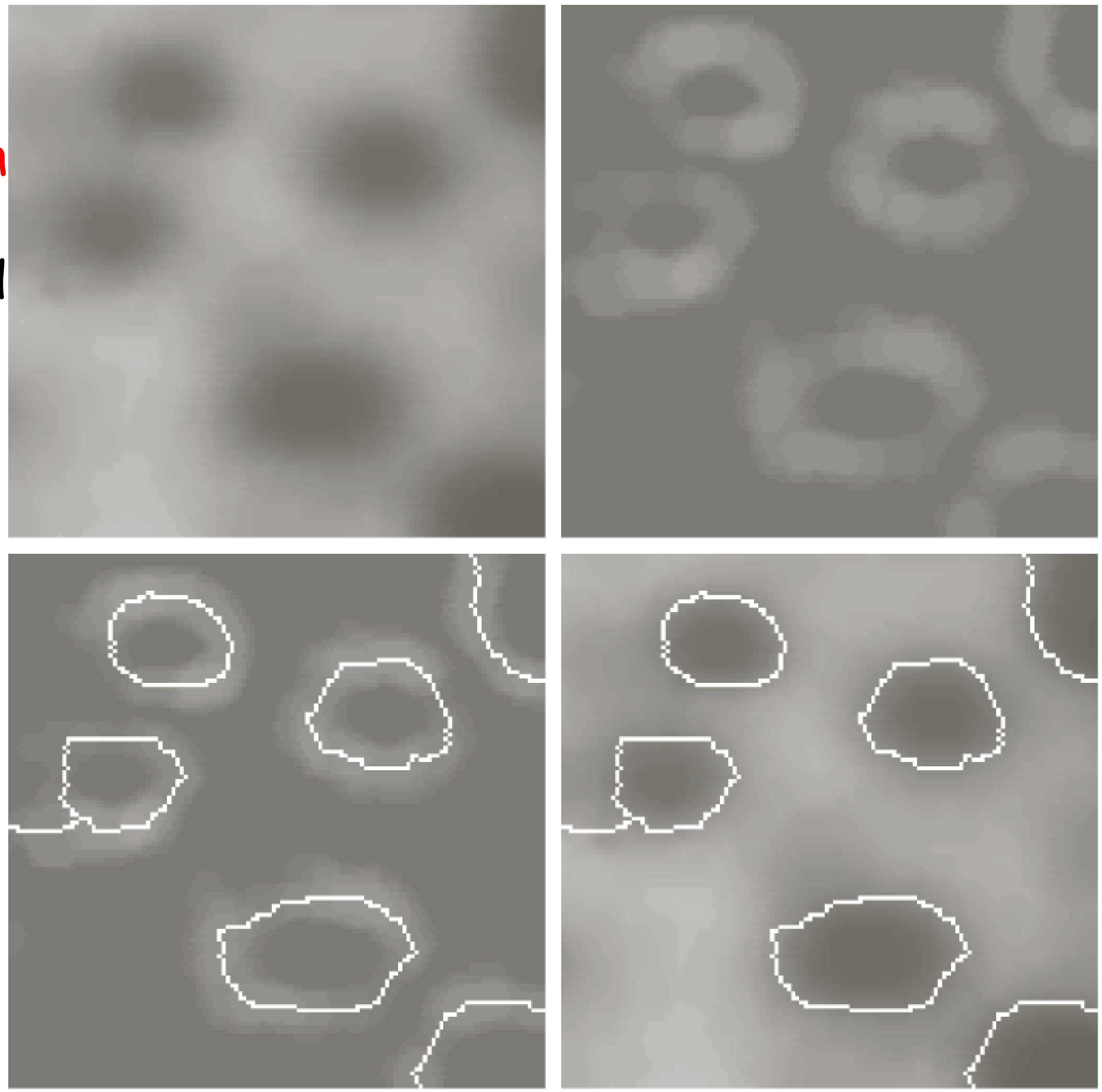
FIGURE 10.60 *(Continued)* (e) Result of further flooding. (f) Beginning of merging of water from two catchment basins (a short dam was built between them). (g) Longer dams. (h) Final watershed (segmentation) lines superimposed on the original image. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

Watershed Segmentation Algorithm

- Start with all pixels with the lowest possible value.
  - These form the basis for initial watersheds
- For each intensity level k:
  - For each group of pixels of intensity k
    1. If adjacent to exactly one existing region, add these pixels to that region (catchment)
    2. Else if adjacent to more than one existing regions, mark as boundary
    3. Else start a new region

Watershed Segmentation: Examples

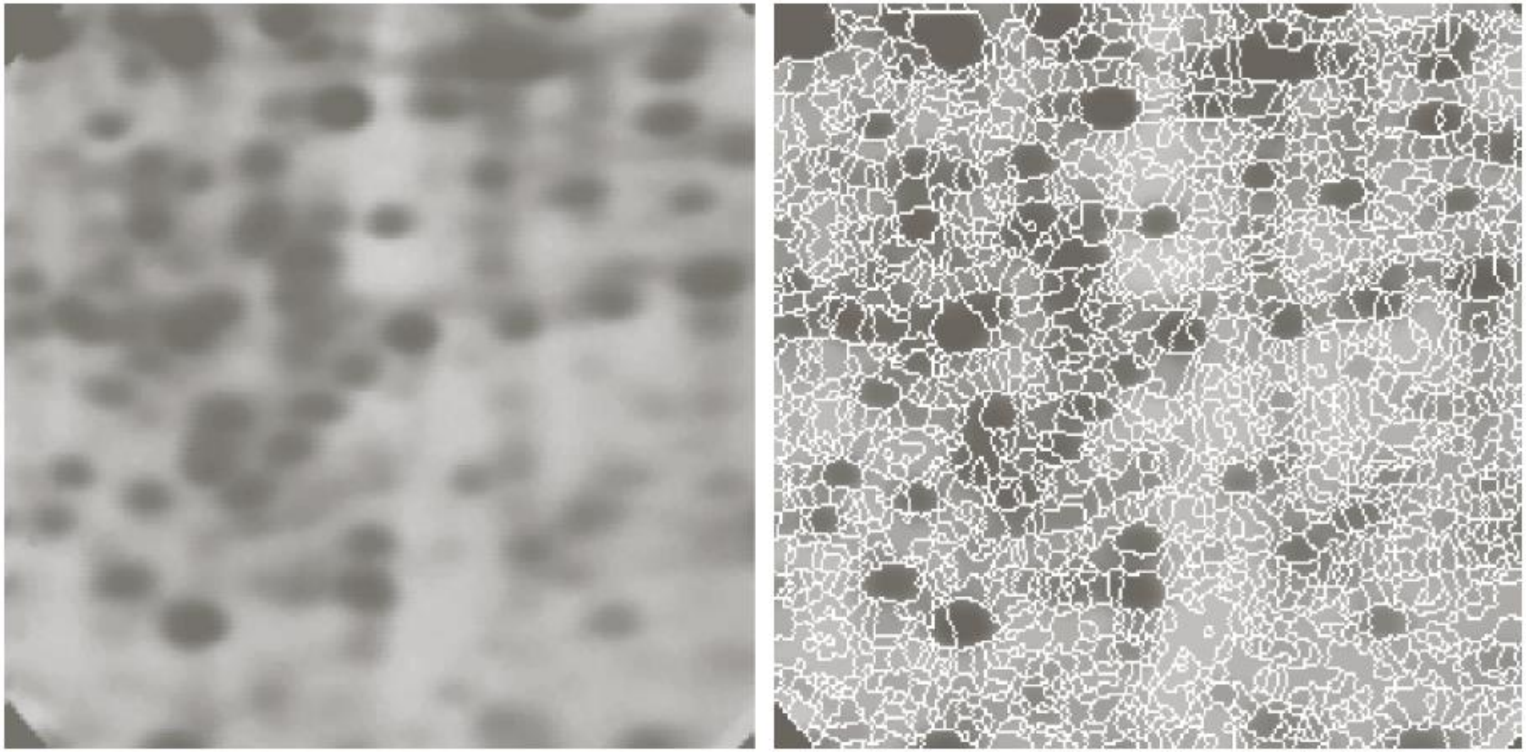
Watershed algorithm is often used on the gradient image instead of the original image.



a b
c d

FIGURE 10.62 (a) Image of blobs. (b) Image gradient. (c) Watershed lines, superimposed on the gradient image. (d) Watershed lines superimposed on the original image. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

Watershed Segmentation: Over-segmentation problem



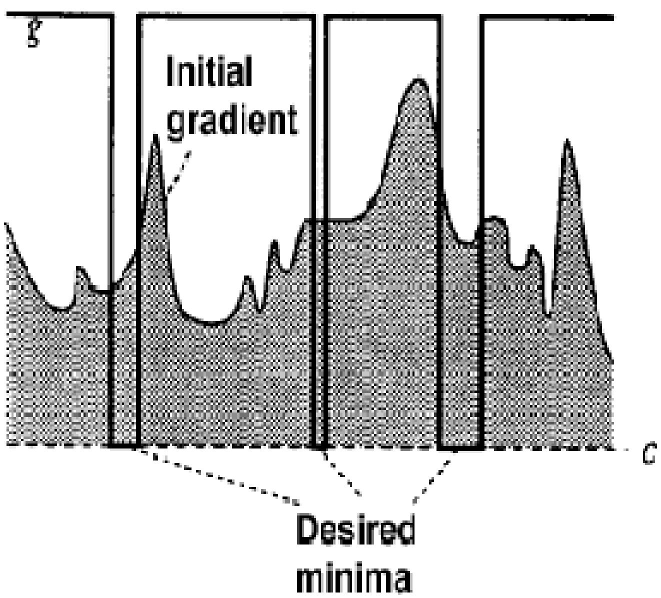
a b

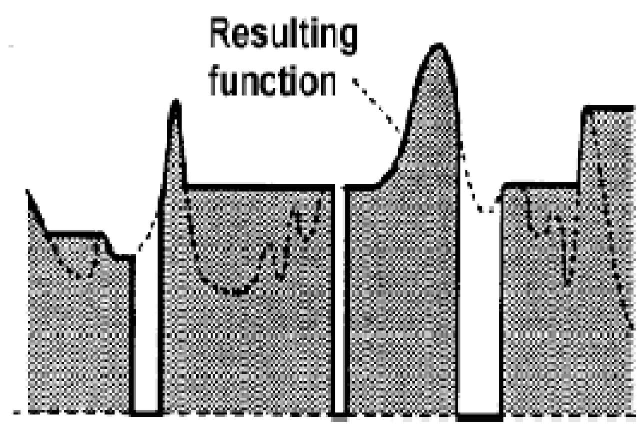
FIGURE 10.63
(a) Electrophoresis image.
(b) Result of applying the watershed segmentation algorithm to the gradient image. Over-segmentation is evident. (Courtesy of Dr. S. Beucher, CMM/ Ecole des Mines de Paris.)

Due to noise and other local irregularities of the gradient, over-segmentation might occur.

Watershed Segmentation: Use of Markers

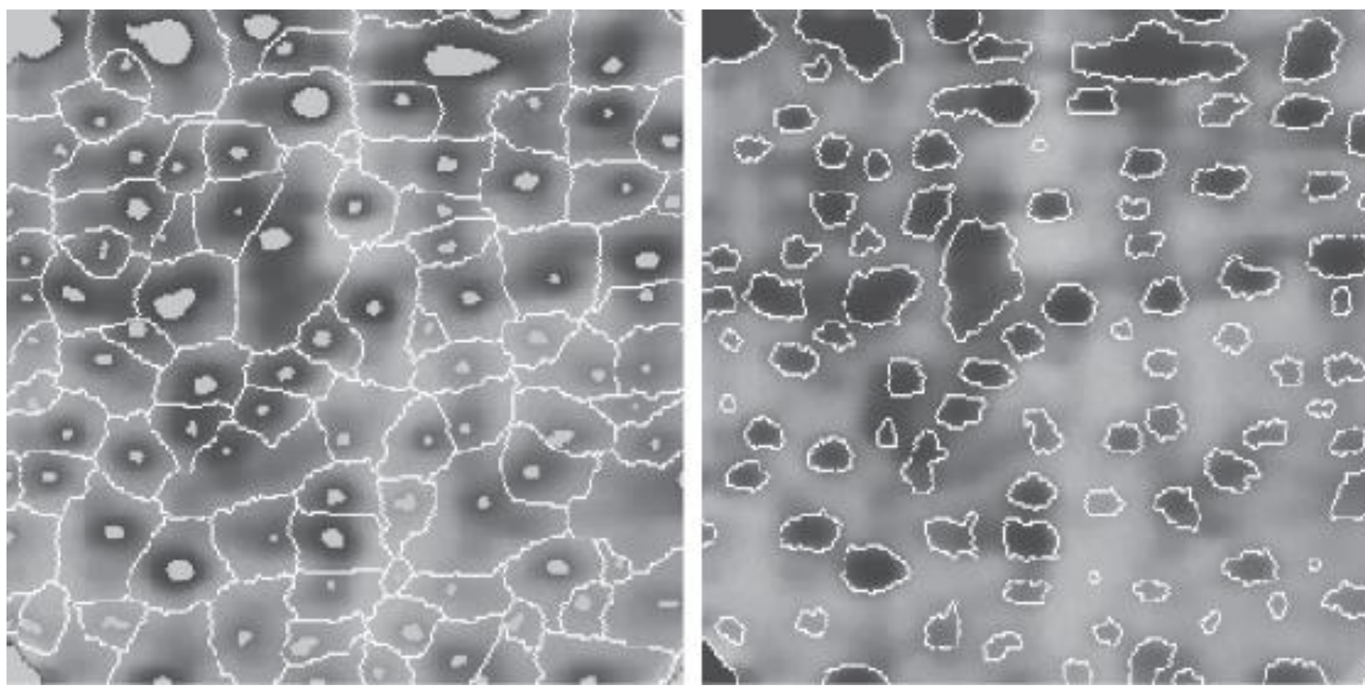
- A solution is to limit the number of regional minima.
- Some of the minimas may be irrelevant detail
- Perform image-smoothing to remove small spatial details
- Use markers (internal and background) to specify only the allowed regional minima.





Watershed Segmentation: Examples

- A solution is to limit the number of regional minima.
- Use markers to specify only the allowed regional minima. (For example, gray-level values might be used as a marker.)



a b

FIGURE 10.64
(a) Image showing internal markers (light gray regions) and external markers (watershed lines).
(b) Result of segmentation. Note the improvement over Fig. 10.63(b). (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

10.6 Use of Motion in Segmentation

Objective: Remove the principal moving object

Use ADI (Accumulative difference Image): Compares the reference image with every subsequent image

a b c

FIGURE 10.66 Building a static reference image. (a) and (b) Two frames in a sequence. (c) Eastbound automobile subtracted from (a), and the background restored from the corresponding area in (b). (Jain and Jain.)

Acknowlegements

The slides are primarily based on the figures and images in the Digital Image Processing textbook by Gonzalez and Woods:

- http://www.imageprocessingplace.com/DIP-3E/dip3e_book_images_downloads.htm

In addition, slides have been adopted and modified from the following sources:

- http://gear.kku.ac.th/~nawapak/178353.html
- https://cs.nmt.edu/~ip/index.html

Regional Processing

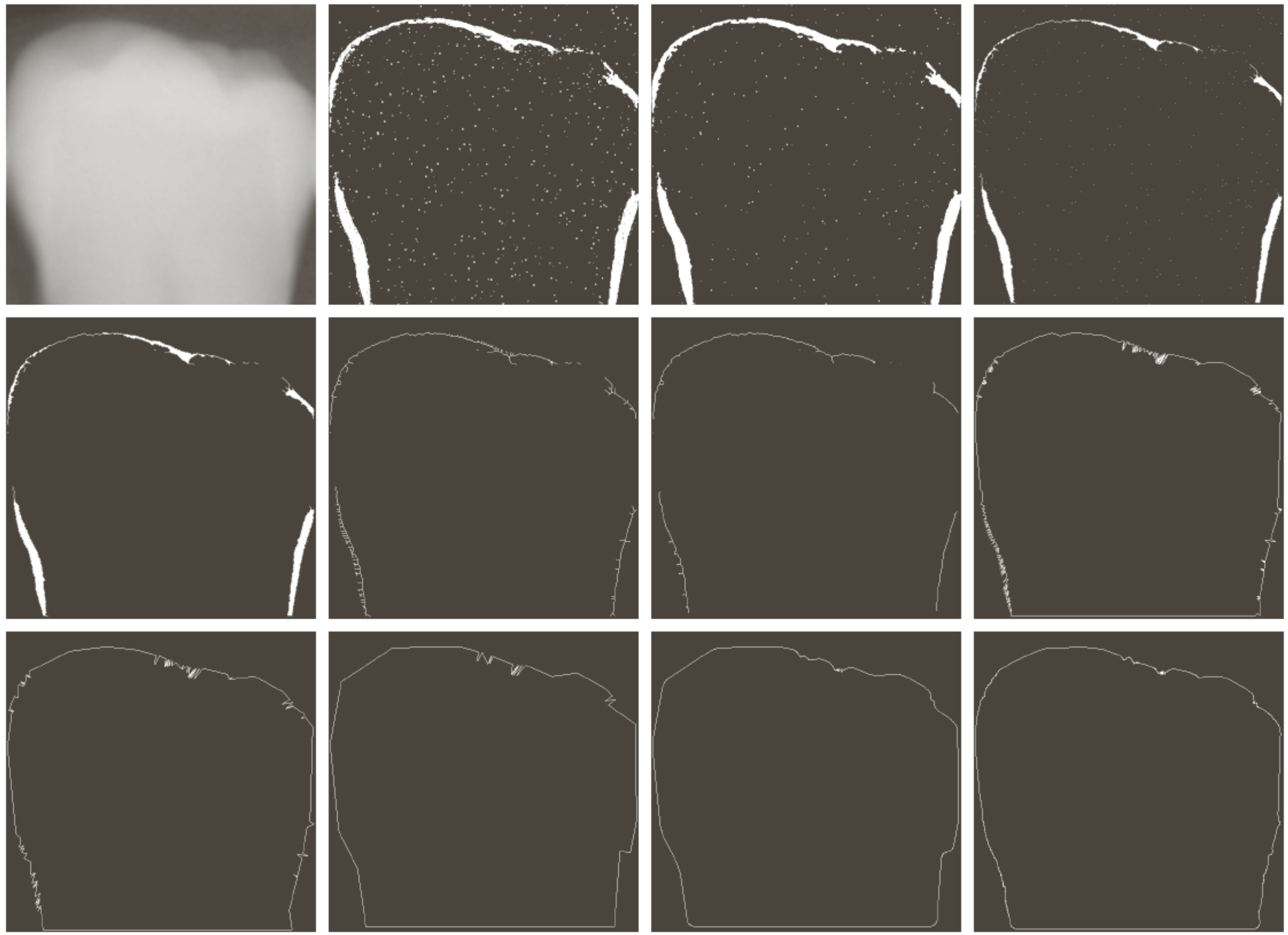
- The location of regions of interest in an image are known or can be determined

- Polygonal approximations can capture the essential shape features of a region while keeping the representation of the boundary relatively simple

- Open or closed curve

  Open curve: A large distance between two consecutive points in the ordered sequence relative to the distance between other points

Result with Regional Processing



a	b	c	d
e	f	g	h
i	j	k	l

FIGURE 10.30 (a) A 550 × 566 X-ray image of a human tooth. (b) Gradient image. (c) Result of majority filtering. (d) Result of morphological shrinking. (e) Result of morphological cleaning. (f) Skeleton. (g) Spur reduction. (h)–(j) Polygonal fit using thresholds of approximately 0.5%, 1%, and 2% of image width ($T = 3$, 6, and 12). (k) Boundary in (j) smoothed with a 1-D averaging filter of size 1 × 31 (approximately 5% of image width). (l) Boundary in (h) smoothed with the same filter.

10.4.2 Region Splitting and Merging

R : entire image R_i: sub-image Q: predicate

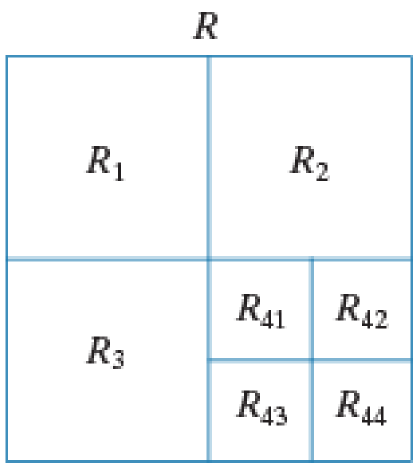
1. For any region R_i, If $Q(R_i)$ = FALSE, we divide the image R_i into quadrants.
2. When no further splitting is possible, merge any adjacent regions R_j and R_k for which $Q(R_j \cup R_k)$ = TRUE.
3. Stop when no further merging is possible.

$$Q = \begin{cases} \text{TRUE} & \text{if } \sigma > a \text{ AND } 0 < m < b \\ \text{FALSE} & \text{otherwise} \end{cases}$$

Region Splitting and Merging

a b

FIGURE 10.47
(a) Partitioned image.
(b) Corresponding quadtree.
R represents the entire image region.



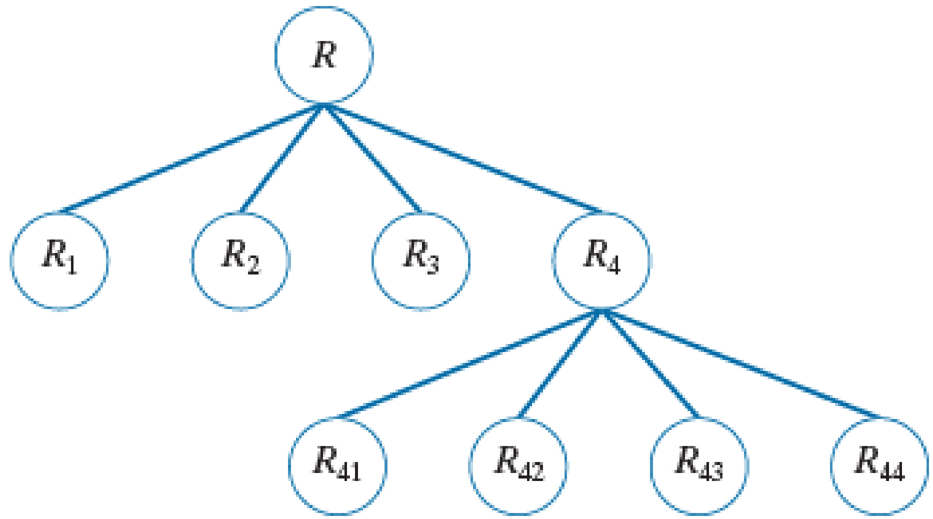


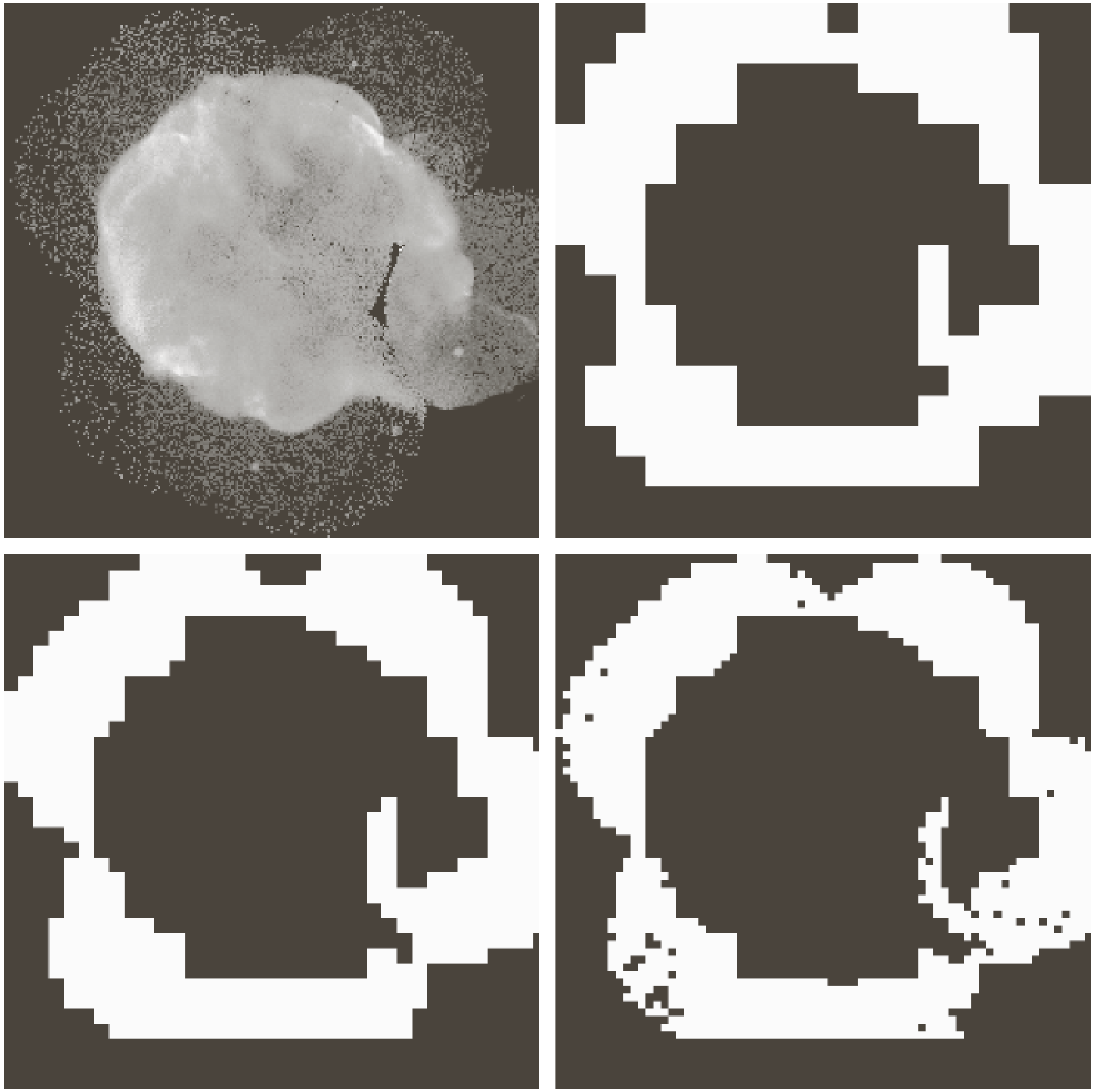
Figure 10.53: Region Splitting and Merging

- Predicate used to segment the region of interest

$$Q = \begin{cases} \text{TRUE} & \text{if } \sigma > a \text{ and } 0 < m < b \\ \text{FALSE} & \text{otherwise} \end{cases}$$

- m : Mean
- σ : Standard Deviation
- a and b: Constants

Region Splitting and Merging



a = 125
b =10

a b
c d

FIGURE 10.48 (a) Image of the Cygnus Loop supernova, taken in the X-ray band by NASA's Hubble Telescope. (b) through (d) Results of limiting the smallest allowed quadregion to be of sizes of 32×32, 16×16, and 8×8 pixels, respectively. (Original image courtesy of NASA.)