Image segmentation:

- Segmentation refers to the process of partitioning a image into multiple regions.
- Two kinds of approaches to segmentation: discontinuity and similarity.
  - Similarity may be due to pixel intensity, color or texture.
  - Differences are sudden changes (discontinuities) in any of these, but especially sudden changes in intensity along a boundary line, which is called an edge.

- Let R represent the entire image region. We want to partition R into n sub regions, R1, R2, . . ., Rn such that:

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

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

(c) $R_i \cap R_j = \varnothing$ for all i and j, $i \neq j$.

(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 and R_j.

Fundamentals:

(a) segmentation must be complete

- all pixels must belong to a region

(b) pixels in a region must be connected

(c) Regions must be disjoint

(d) states that pixels in a region must all share the same property

- The logic predicate Q(Ri) over a region must return TRUE for each point in that region

(e) indicates that regions are different in the sense of the predicate Q

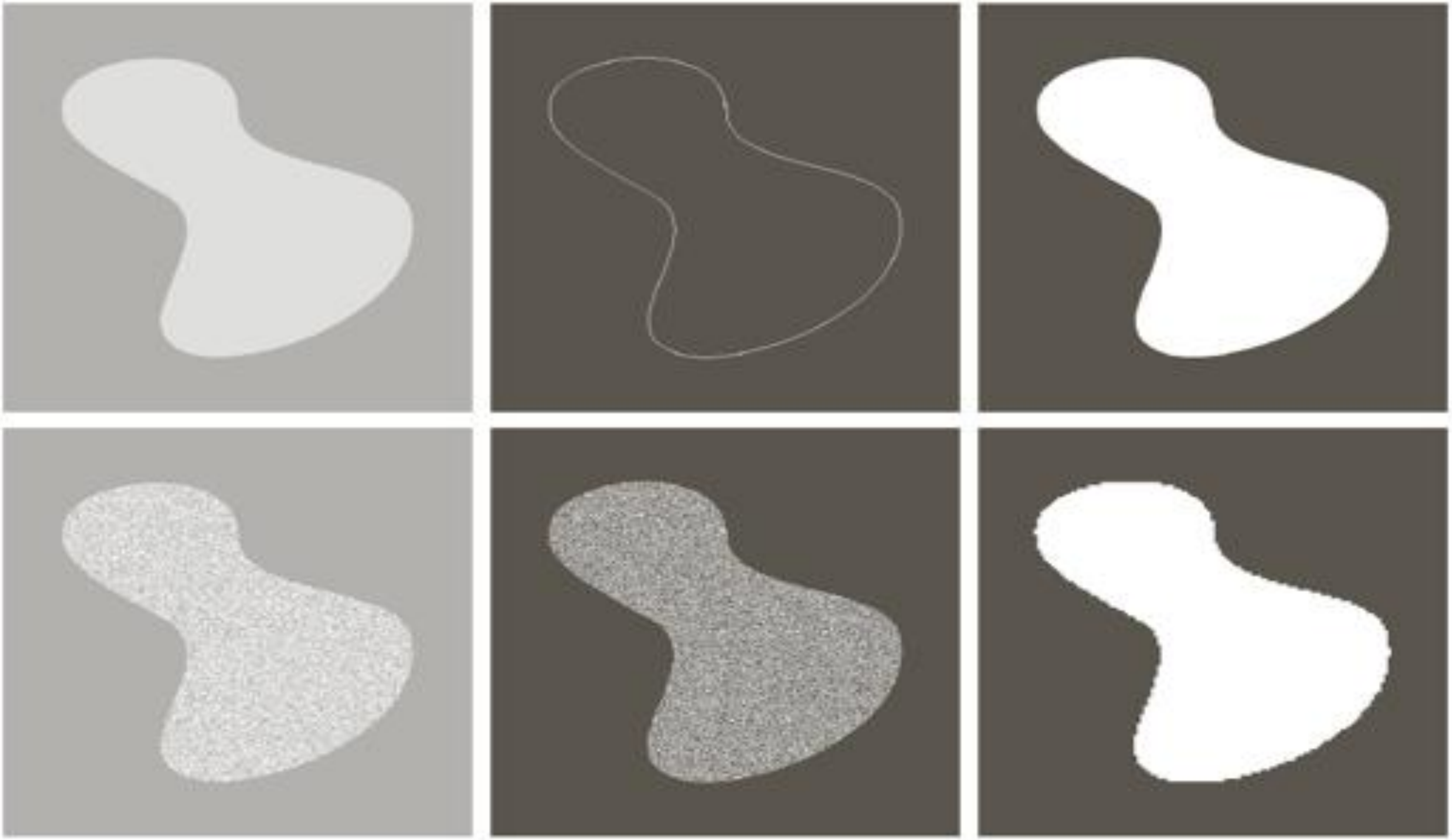


FIGURE 10.1 (a) Image containing a region of constant intensity. (b) Image showing the boundary of the inner region, obtained from intensity discontinuities. (c) Result of segmenting the image into two regions. (d) Image containing a textured region. (e) Result of edge computations. Note the large number 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.

Point, Edge and line pixel:

- The focus of this topic is on segmentation methods that are based on detecting sharp, local changes in intensity.

Three types of images features in which we are interested are:

- Edges: edge pixels are pixels at which the intensity of an image function changes abruptly, and edges are sets of connected edge pixels.

- Lines: line may be viewed as a edge segment in which the intensity of the background on either side of the line is either much higher or much lower than the intensity of the line pixels.

- Isolated point: It can be viewed as a line whose length and width are equal to one pixel.

Background:

- Local changes in intensity can be detected using derivatives.
- Derivatives of a digital function are defined in terms of differences.
- First order derivatives:
  - Must be zero in the areas of constant intensity.
  - Must be nonzero at the onset of an intensity step or ramp.
  - Must be nonzero along the ramp

Background:

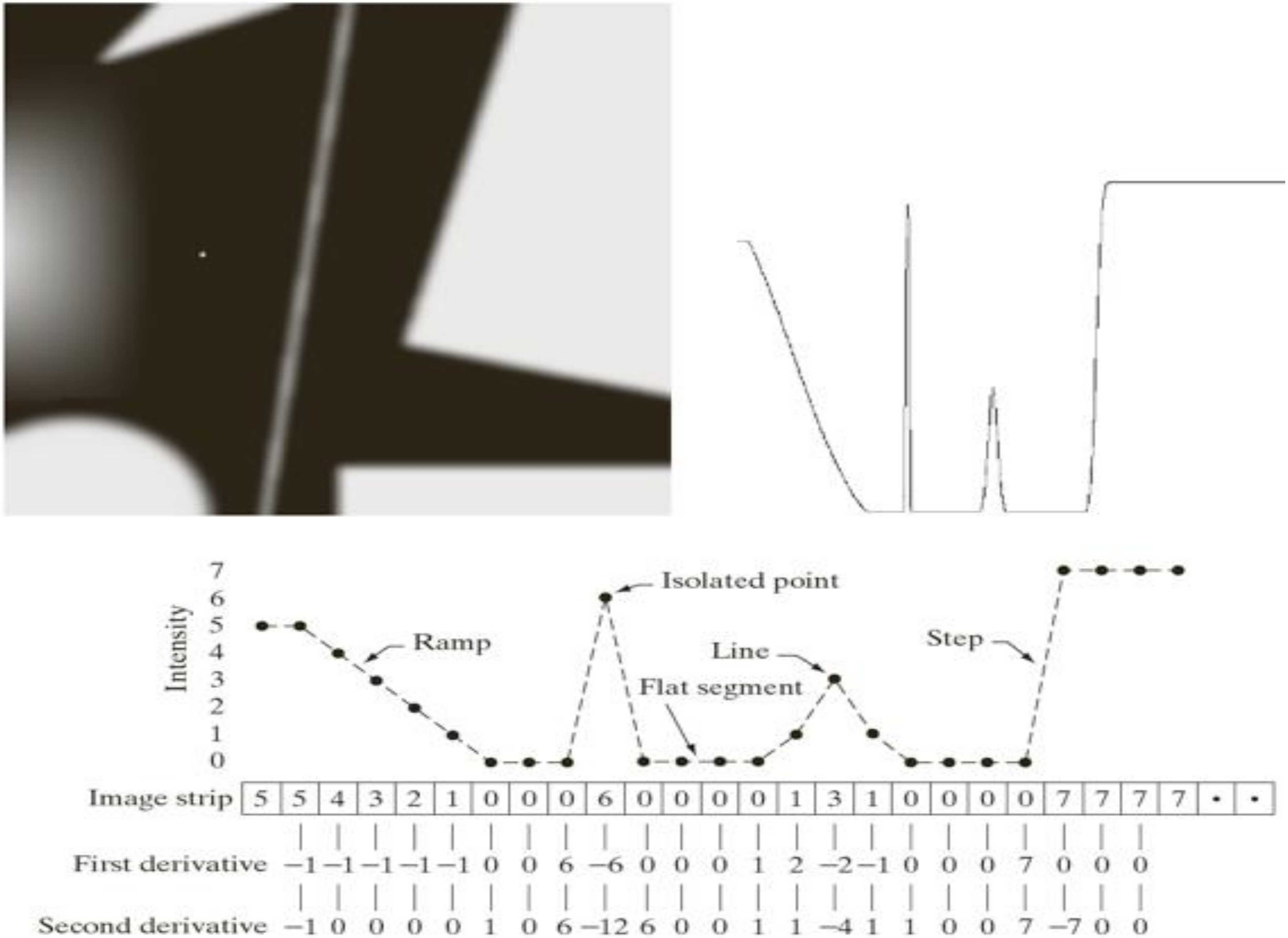
- Second derivative operator:

  – must be 0 on constant regions
  – must be nonzero on the onset of steps/ramps
  – must be 0 along ramps with constant slope

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

$$\begin{aligned}\frac{\partial^2 f}{\partial x^2} &= \frac{\partial f'(x)}{\partial x} = f'(x+1) - f'(x) \\ &= f(x+2) - f(x+1) - f(x+1) + f(x) \\ &= f(x+2) - 2f(x+1) + f(x)\end{aligned}$$

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


Intensity
7
6
5
4
3
2
1
0
Isolated point
Ramp
Line
Step
Flat segment
Image strip 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

We see that:

- First order derivative produce thicker edges in an image
- Second order derivative have high response to detail (points, noise...)
- Second order derivative produce double-edge response at ramps/steps
- The sign of the second derivative can be used to determine transitions dark➔light(positive) or light ➔ dark(negative)

Approach:

- To compute first and second order derivatives is to scan a small mask over the image:

$$R = w_1 z_1 + w_2 z_2 + \ldots + w_9 z_9 = \sum_{i=1}^{9} w_i z_i$$

FIGURE 10.1 A general 3×3 mask.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Detection of isolated points:

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

2-D Laplacian operation

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

x-direction

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

y-direction

$$\nabla^2 f(x, y) = [f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1)] - 4f(x, y)$$

0	1	0
1	-4	1
0	1	0

Fig: a

1	1	1
1	-8	1
1	1	1

Fig: b

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

Fig: c

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

Fig: d

Point Detection:

- The only differences that are considered of interest are those large enough (as Determined by T) to be considered isolated points.

$$|R| > T$$

$$g(x, y) = \begin{cases} 1 & \text{if } |R(x, y)| \geq T \\ 0 & \text{otherwise} \end{cases}$$

Mask:

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

Isolated point: response value 8

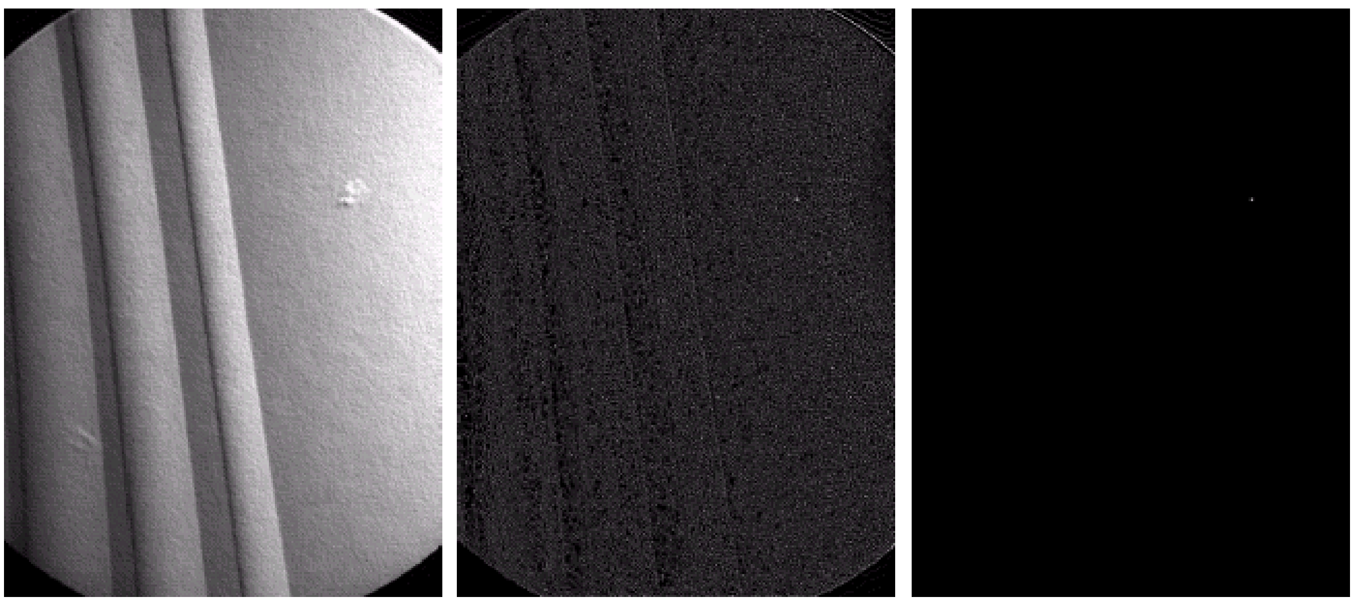
0	0	0
0	1	0
0	0	0

Point which is part of an line: response value 6

0	1	0
0	1	0
0	1	0

Detection of a point:

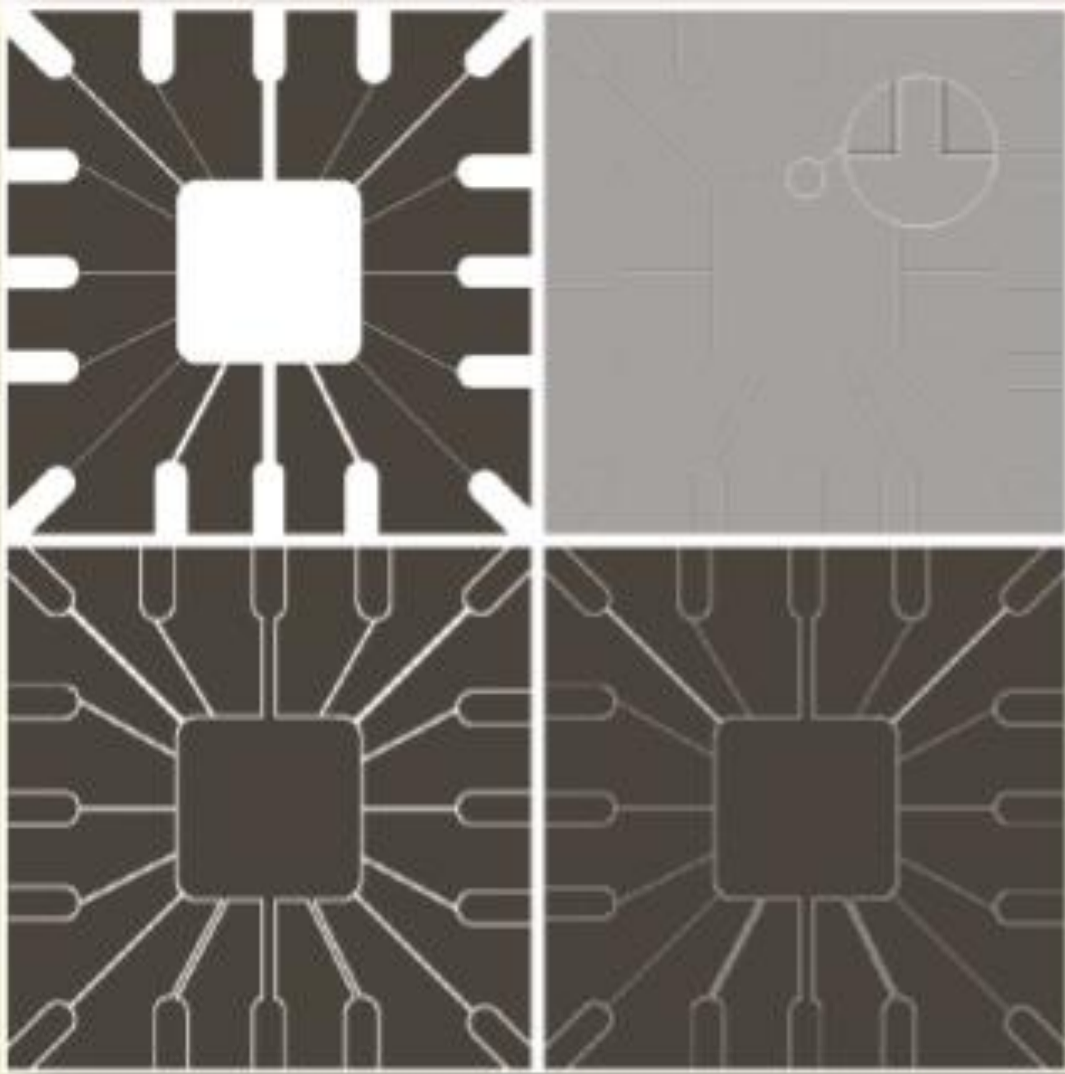
−1	−1	−1
−1	8	−1
−1	−1	−1



a
b c d

FIGURE 10.2 (a) Point detection mask. (b) X-ray image of a turbine blade with a porosity. (c) Result of point detection. (d) Result of using Eq. (10.1-2). (Original image courtesy of X-TEK Systems Ltd.)

Line detection:



a b
c d

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:

- Laplacian mask
- Double line effect
- “Zero valley”
- Isotropic

Line detection in specified direction:

−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

−1	−1	2
−1	2	−1
2	−1	−1

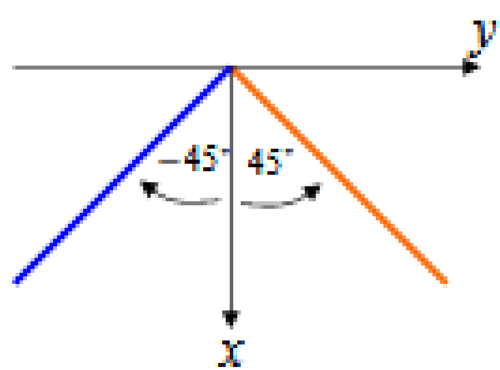
−45°

−1	2	−1
−1	2	−1
−1	2	−1

Vertical

2	−1	−1
−1	2	−1
−1	−1	2

+45°



Line detection:

- Apply every masks on the image
- let R1, R2, R3, R4 denotes the response of the horizontal, +45 degree, vertical and -45 degree masks, respectively.
- if, at a certain point in the image

$$|R_i| > |R_j|, \text{ for all } j \neq i,$$

that point is said to be more likely associated with a line in the direction of

mask i.

Line is assumed 1 pixel thick.

-1	-1	2
-1	2	-1
2	-1	-1

0	0	0	1
0	0	1	0
0	1	0	0
1	0	0	0

		-2	6
	-2	6	-2
-2	6	-2	
6	-2		

Presence of negative value in edge detection:

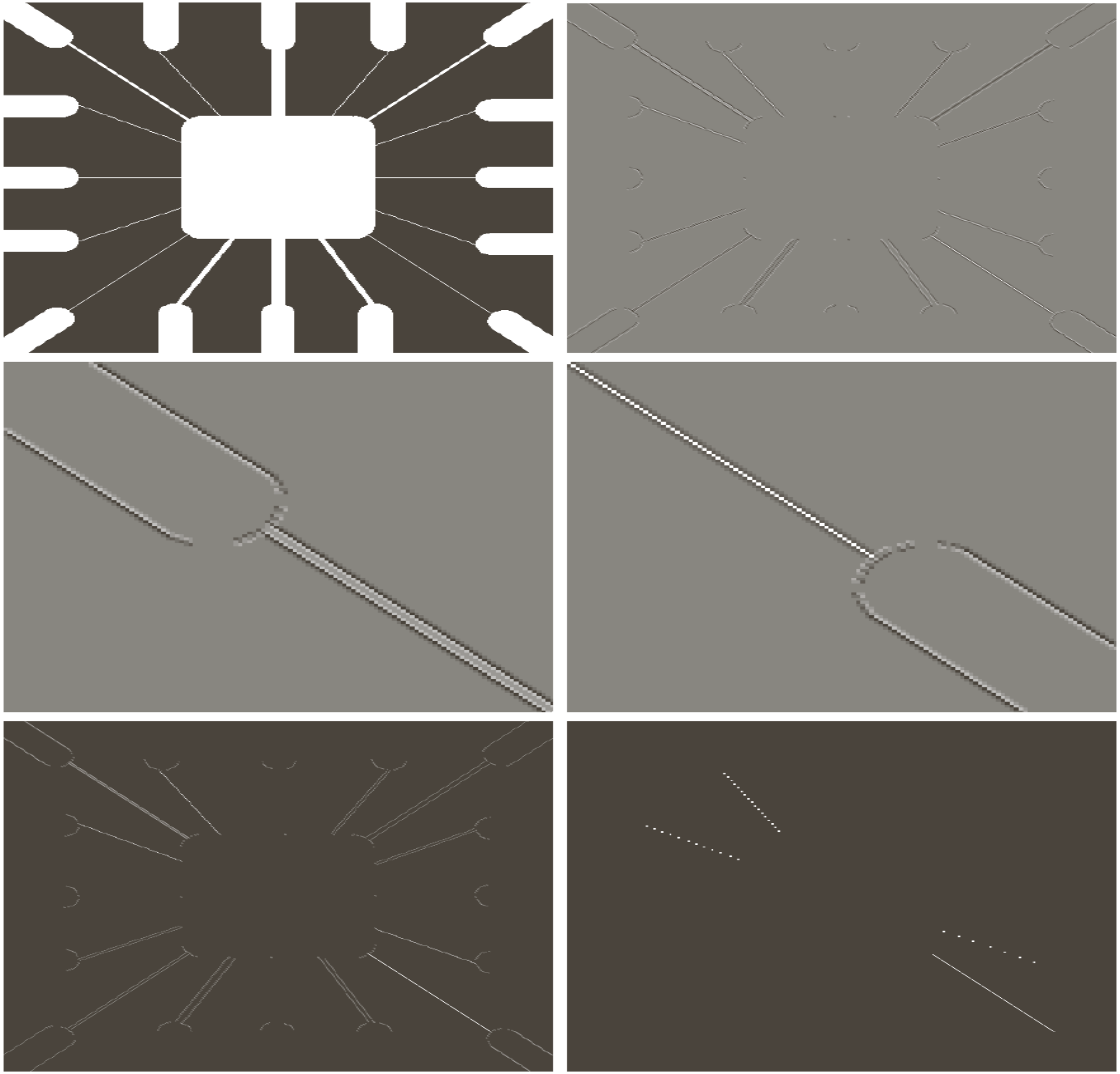
-1	-1	2
-1	2	-1
2	-1	-1

One pixel thick line➔ respeonse value 6

0	0	1
0	1	0
1	0	0

Three pixel thick line➔ response value 2

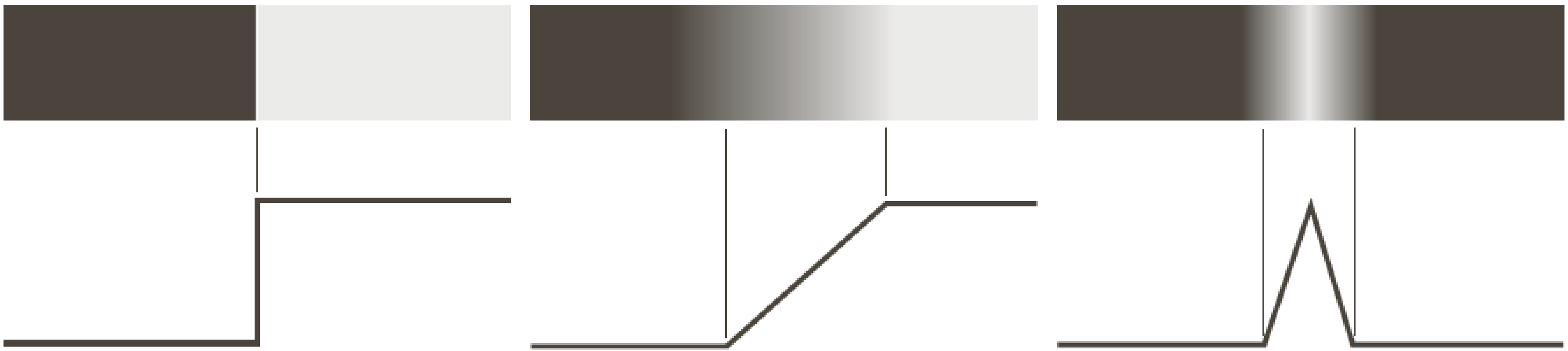
0	1	1
1	1	1
1	1	0



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 mask 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 $g \geq T$, where g is the image in (e). (The points in (f) were enlarged to make them easier to see.)

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.

Edge models:

- Step edge: Ideal one! Possible in computer synthesized images. Noisy images has gradual change in intensities.-ramp model of edge.
- Slope of ramp is inversely proportional to blurring.
- Ramp model line is no longer 1 pixel wide.
- Roof model: e.g in satellite imaging roads
- Image may contain all type of edges, but due to blurring and noise profile may not be the ideal one.

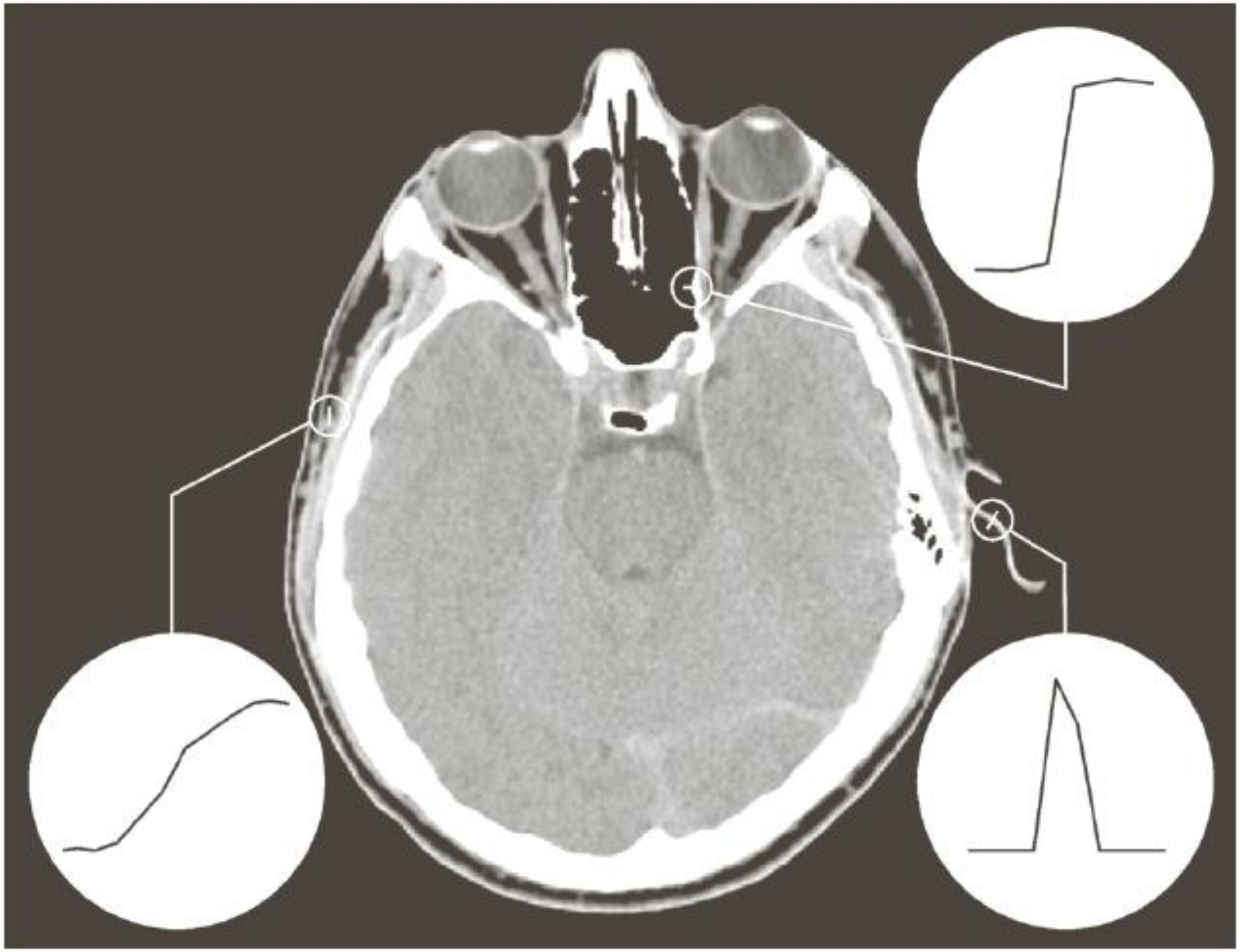
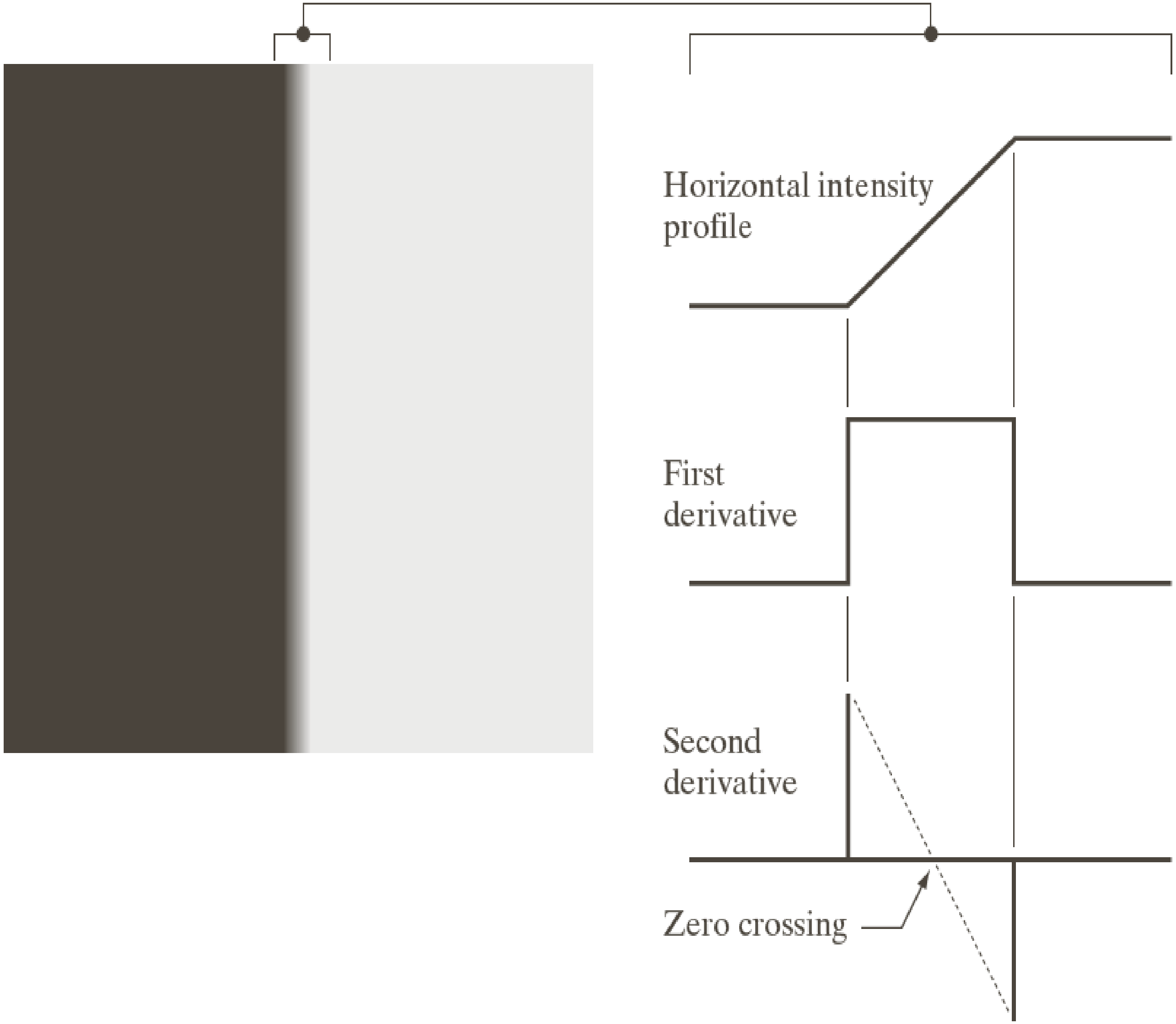


FIGURE 10.9 A 1508 × 1970 image showing (zoomed) actual ramp (bottom, left), step (top, right), and roof edge profiles. The profiles are from dark to light, in the areas indicated by the short line segments shown in the small circles. The ramp and "step" profiles span 9 pixels and 2 pixels, respectively.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



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.

- 1st derivative tells us where an edge is
- 2nd derivative can be used to determine whether an edge pixel lies on the dark or light side of an edge.
- Additional properties of second derivative is – produces two values for every edge in image.
- Zero crossings can be used for locating the canters of thick edges

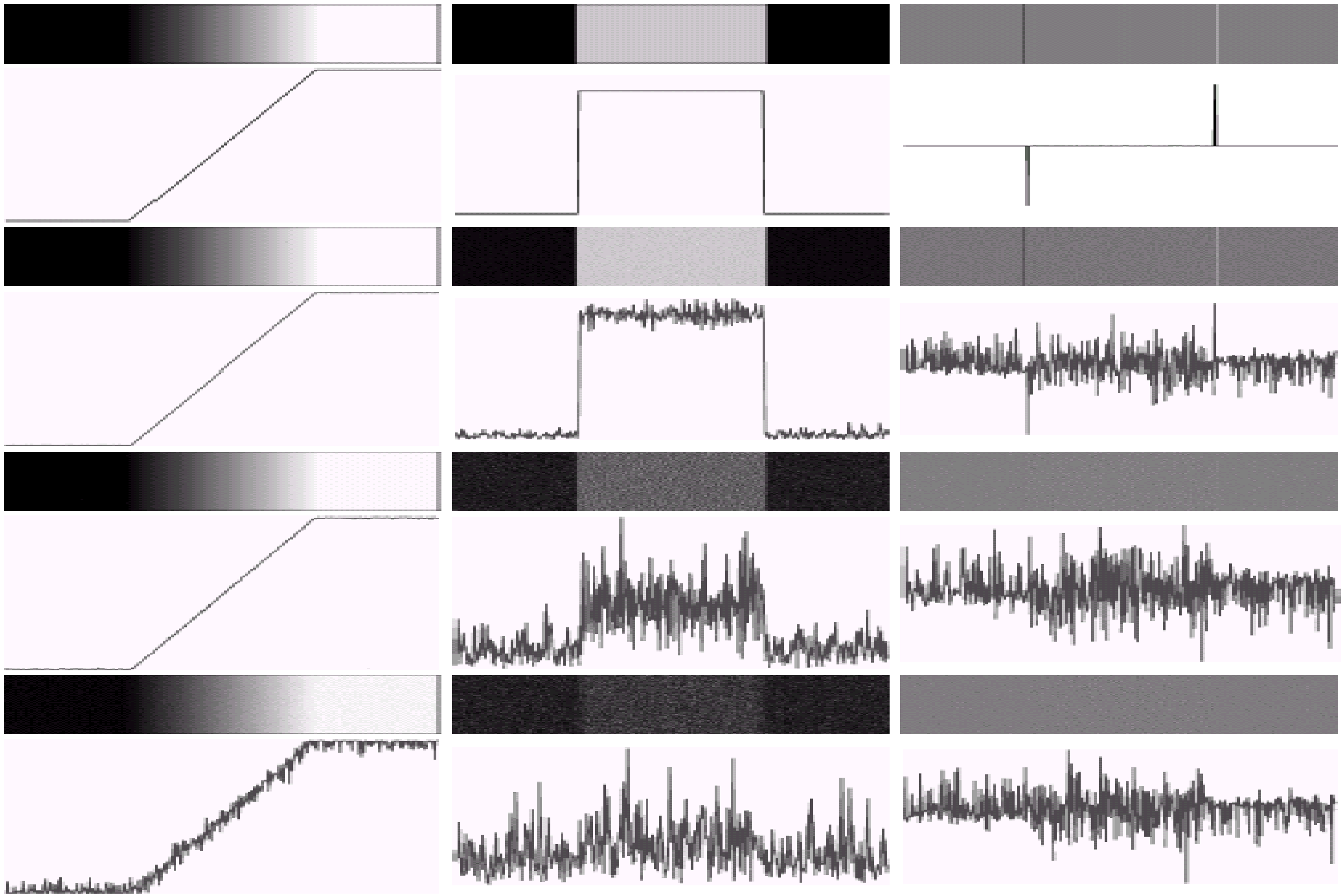


FIGURE 10.7 First column: images and gray-level profiles of a ramp edge corrupted by random Gaussian noise of mean 0 and $\sigma = 0.0, 0.1, 1.0$, and 10.0, respectively. Second column: first-derivative images and gray-level profiles. Third column: second-derivative images and gray-level profiles.

a
b
c
d

We conclude the edge models section by noting the there are three fundamental steps performed in edge detection:

1. *Image Smoothing for noise reduction*: the need of this step is to reduce the noise in an image.

2. *Detection of edge points*: it is a local operation that extracts all points from an image, these points are potential candidates to become edge points.

3. *Edge Localization:* the objective of this point is to select from edge points only the points that are true members of the set of points comprising an edge.

Basic edge detection:

The image gradient and its properties:

- Gradient points in the direction of greatest rate of change.

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

The *magnitude* (*length*) of vector ∇f, denoted as $M(x, y)$, where

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

M is the value of rate of change in the direction of gradient vector

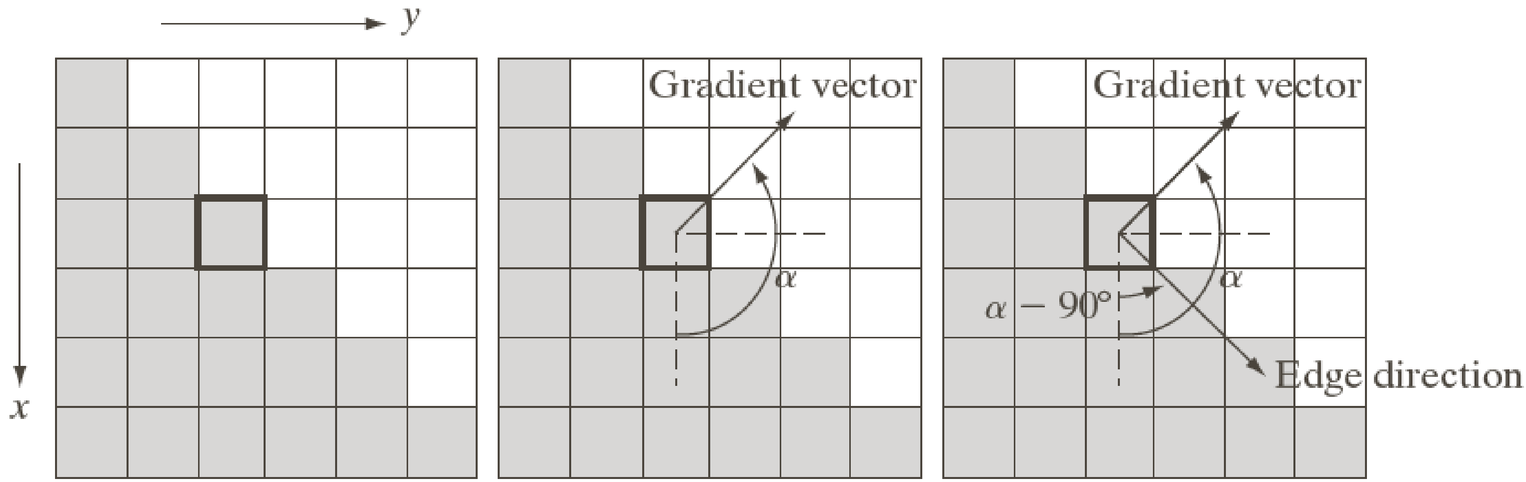
The *direction* of the gradient vector is given by the angle

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

The direction of an edge at an arbitrary point (x, y) is *orthogonal* to the direction, $\alpha(x, y)$, of the gradient vector at the point.

Detection of edges

Example



Each square is a pixel

Let the gray pixel be 0 and white pixel be 1.

Let us get the partial derivative along x and y direction.

Detection of edges

To get partial derivative in x direction

Subtract the pixels in the top row of the neighborhood from the pixels in the bottom row

0	1	1
0	0	1
0	0	0

We get

$$g_x = \frac{\partial f}{\partial x} = (0-0)+(0-1)+(0-1) = -2$$

Detection of edges

To get partial derivative in y direction

Subtract the pixels in the left column from the pixels in the right column.

0	1	1
0	0	1
0	0	0

We get

$$g_x = \frac{\partial f}{\partial y} = (1-0) + (1-0) + (0-0) = 2$$

Detection of edges

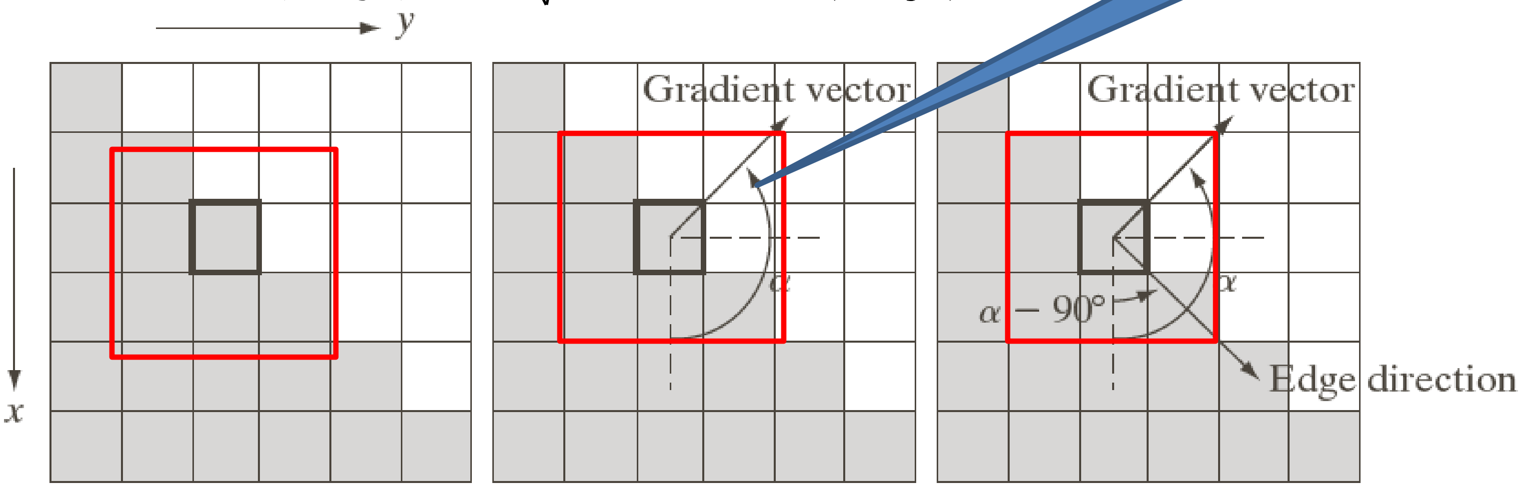
At the point of question, we get

$$\nabla f = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \end{bmatrix} = \begin{bmatrix} -2 \\ 2 \end{bmatrix}$$

Edge normal
If normalized to unit length then
Edge unit normal

$$M(x,y) = 2\sqrt{2} \quad and \quad \alpha(x,y) = -45^o$$

Which is same as
135^o measured
from the +ve axis

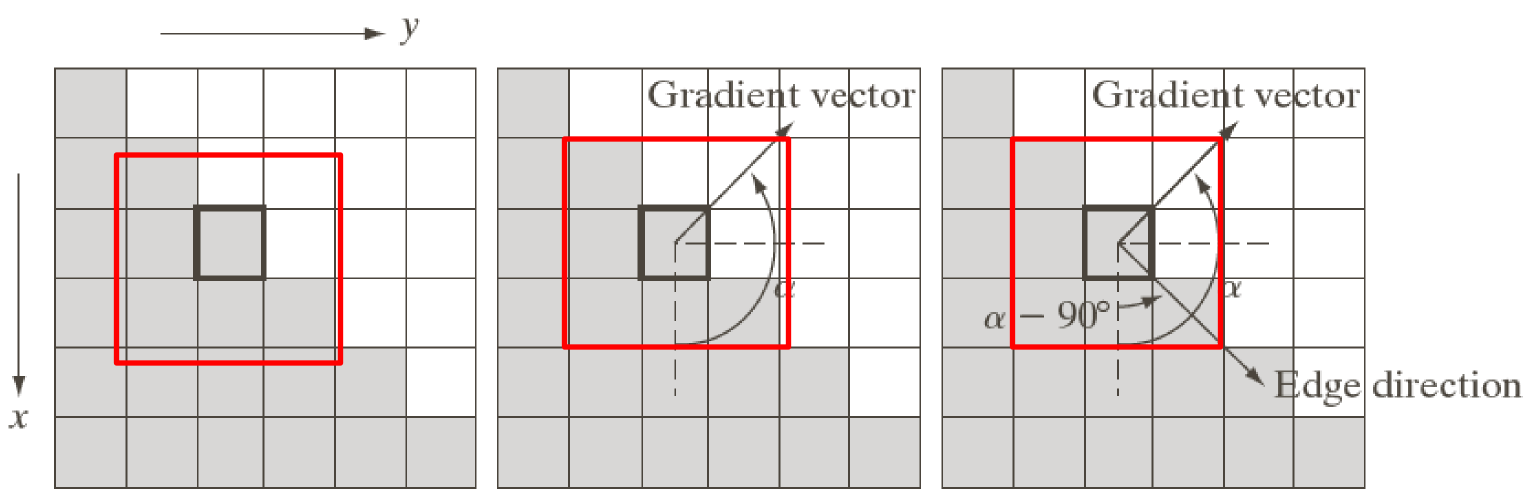


Detection of edges

Edge at a point is orthogonal to the gradient vector at that point.

So direction of the angle of the edge in this example is (α-90)=45

All edge points in the example shown below have same gradient so the entire segment is in the same direction.



Gradient operators:

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

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

−1
1

−1	1

Roberts cross gradient operators:

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

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

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

Prewitt operator:

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

Prewitt

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

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

Sobel operators:

−1	2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Sobel

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

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

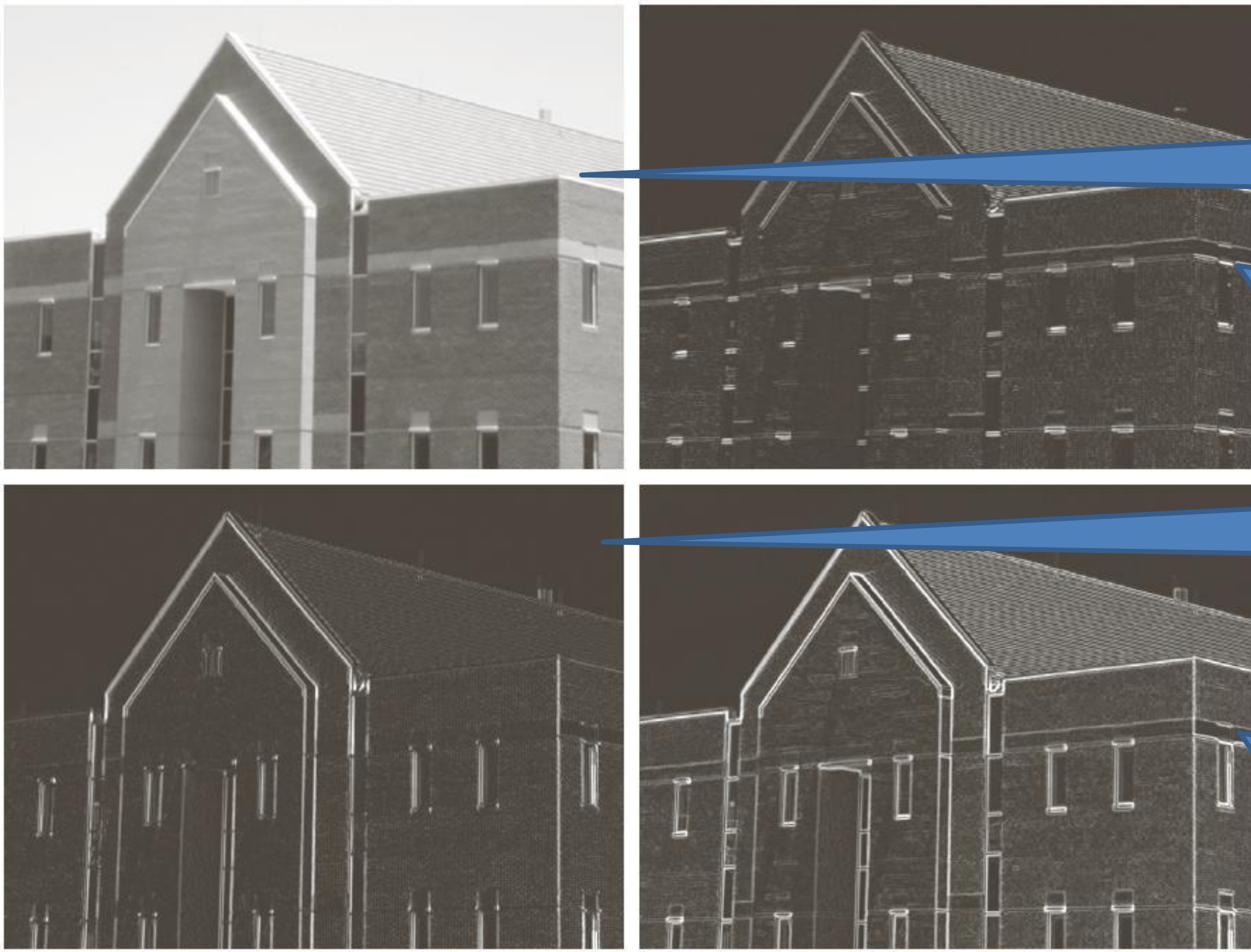
The Prewitt masks are simpler to implement than the Sobel masks, but, the slight computational difference between them typically is not an issue. The fact that the Sobel masks have better noise-suppression (smoothing)

- Using any of mask two partial derivatives gx and gy is obtained.
- Used to determine the magnitude and direction of the edge.

$$M(x, y) \approx |g_x| + |g_y|$$

- Terminology➔Edge map: when referring to an image whose principle features are edges such as gradient magnitude images.

Gradient operator (Example)



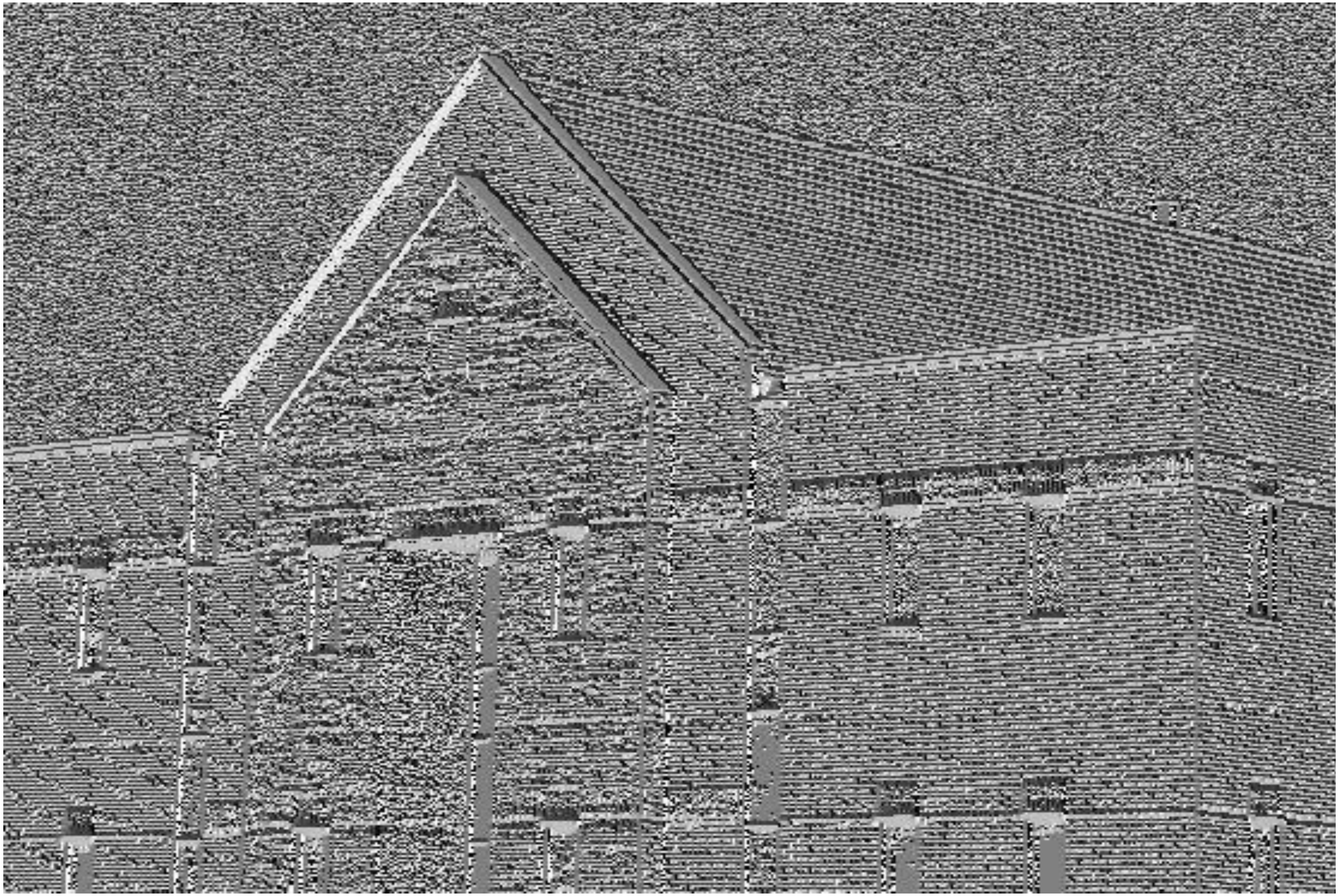
Original image (The intensity values scaled between [0 1]

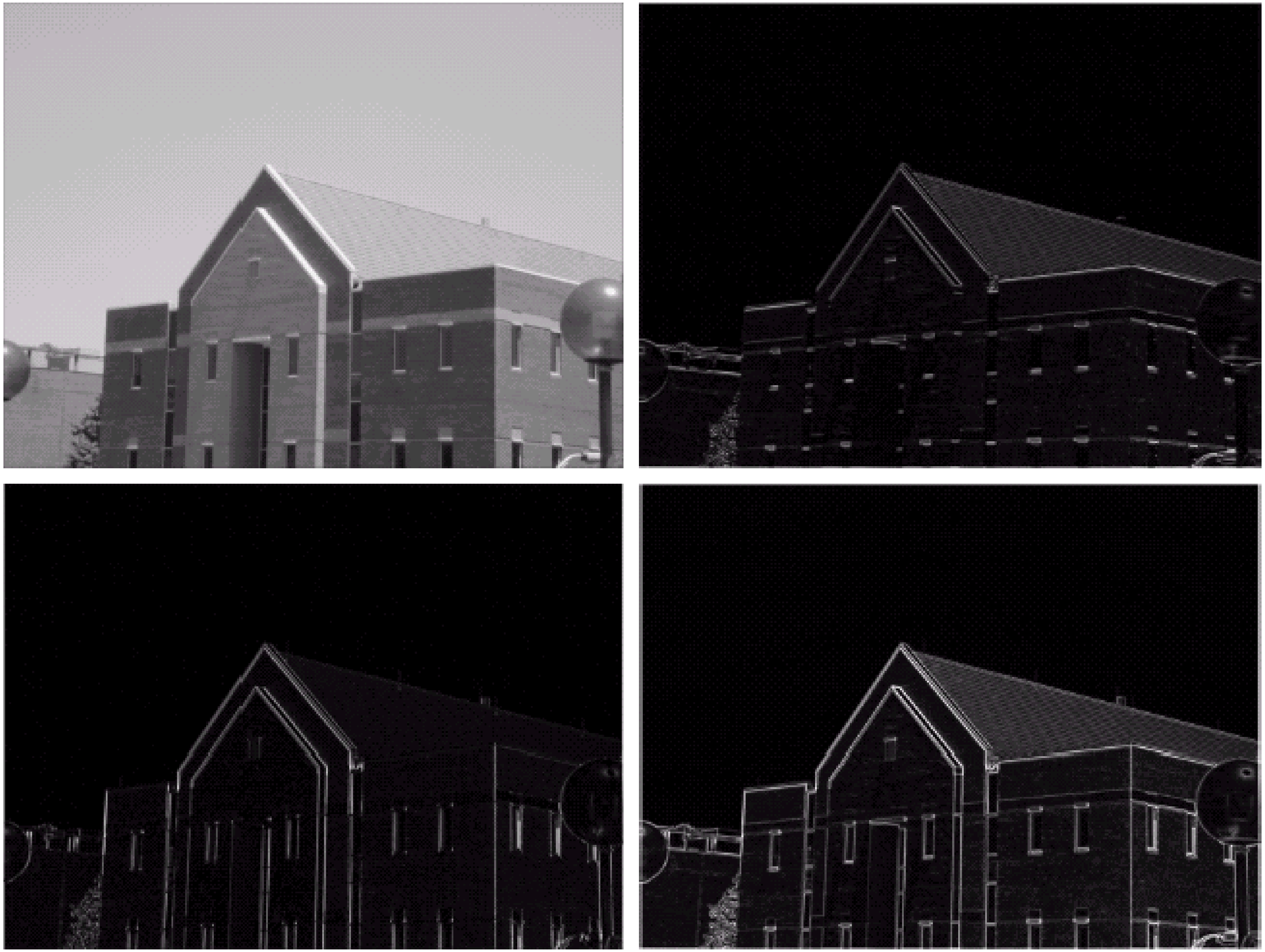
g_x component of the sobel mask

g_y component of the sobel mask

The gradient image $|g_x| + |g_y|$

Gradient angle image:

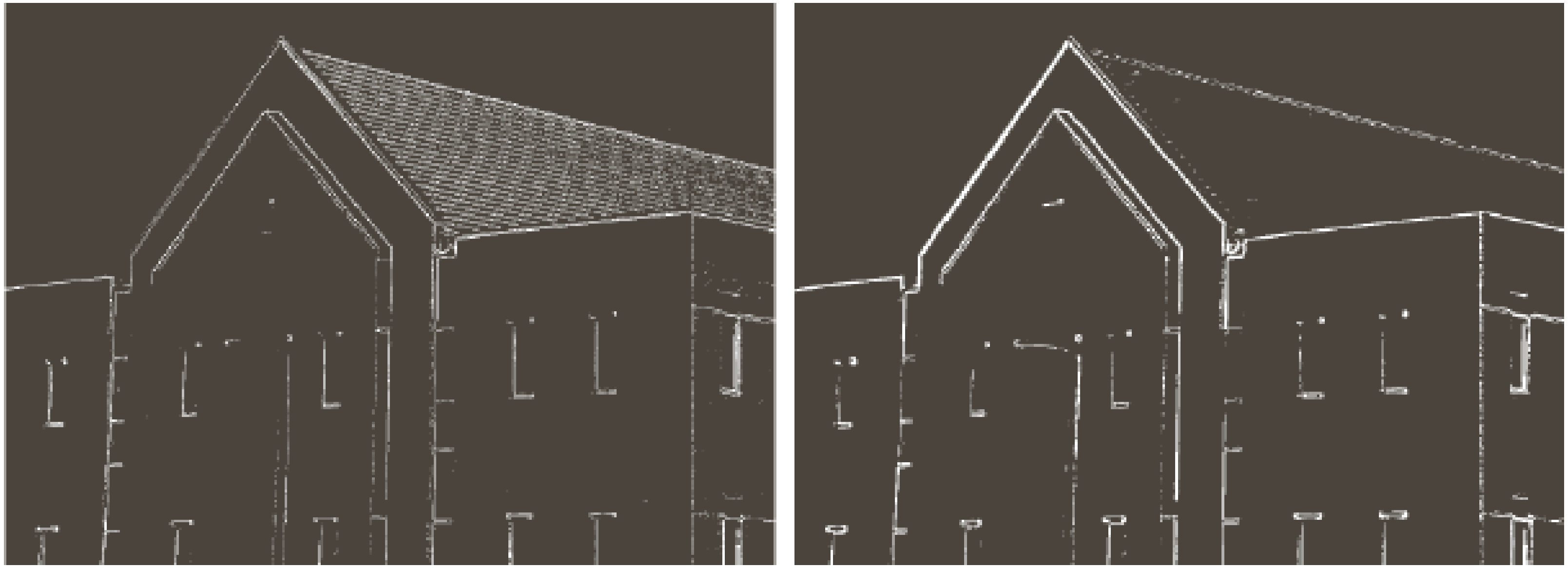




a	b
c	d

FIGURE 10.11 Same sequence as in Fig. 10.10, but with the original image smoothed with a 5×5 averaging filter.

Combined with Thresholding:



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.

Diagonal edge detection:

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

FIGURE 10.12 Diagonal edge detection. (a) Result of using the mask in Fig. 10.9(c). (b) Result of using the mask in Fig. 10.9(d). The input in both cases was Fig. 10.11(a).

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

45° and -45° lines

Advanced Techniques:

- More advanced techniques make attempt to improve the simple detection by taking into account factors such as noise, scaling etc.

Scale dependency:

- Large operator used to detect blurry edges and small operators to detect sharply focused fine details

Marr and Hildreth argued (1) that intensity changes are not independent of image scale and so their detection requires the use of operators of different sizes; and (2) that 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

Marr and Hildreth Edge Detector

- The derivative operators presented so far are not very useful because they are very sensitive to noise.
- To filter the noise before enhancement, Marr and Hildreth proposed a Gaussian Filter, combined with the Laplacian for edge detection. It is the Laplacian of Gaussian (LoG).

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

with standard deviation σ (sometimes σ is called the *space constant*)

Marr- Hildreth edge detector

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

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

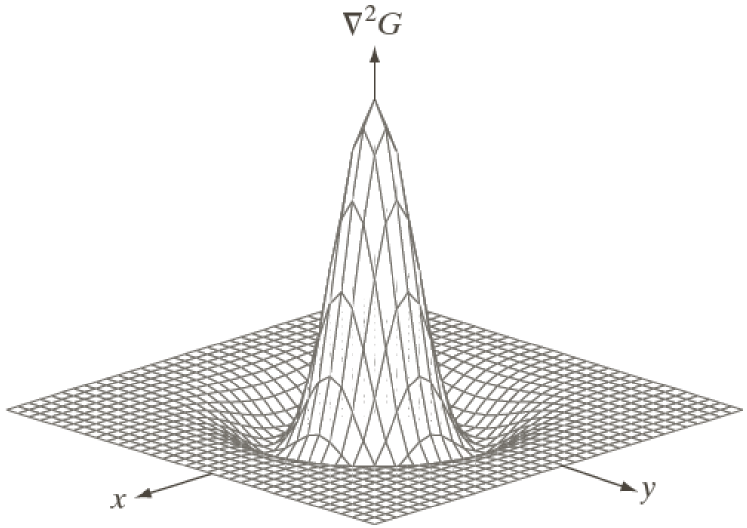
$$\nabla^2 G(x,y) = \frac{\partial}{\partial x}\left(\frac{-x}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \right) + \frac{\partial}{\partial y}\left(\frac{-y}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \right)$$

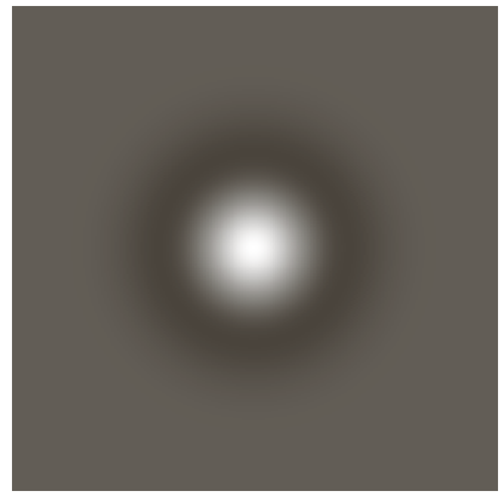
$$\nabla^2 G(x,y) = \left(\frac{x^2}{\sigma^4} - \frac{1}{\sigma^2} \right) e^{-\frac{x^2+y^2}{2\sigma^2}} + \left(\frac{y^2}{\sigma^4} - \frac{1}{\sigma^2} \right) e^{-\frac{x^2+y^2}{2\sigma^2}}$$

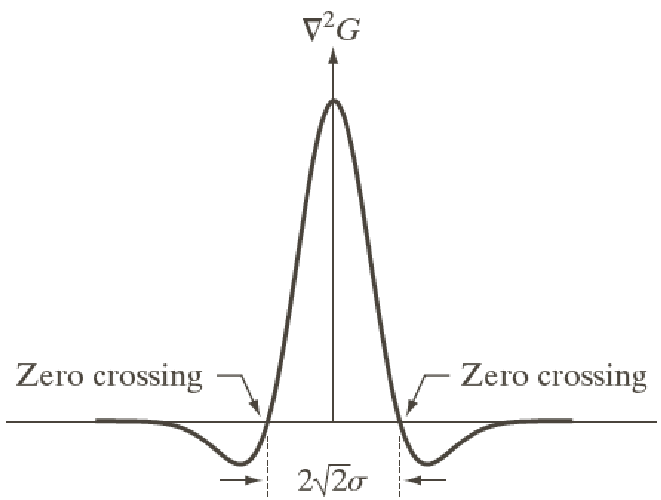
$$\nabla^2 G(x,y) = \left(\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4} \right) e^{-\frac{x^2+y^2}{2\sigma^2}}$$

This expression is called Laplacian of a Gaussion (LoG)

Marr- Hildreth edge detector







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

Maxican hat operator

Marr- Hildreth edge detector

The Marr-Hildreth algorithm consists of convolving the LoG filter with an input image, $f(x, y)$,

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

Because these are linear processes, Eq. can be written also as

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

The Marr-Hildreth edge-detection algorithm may be summarized as follows:

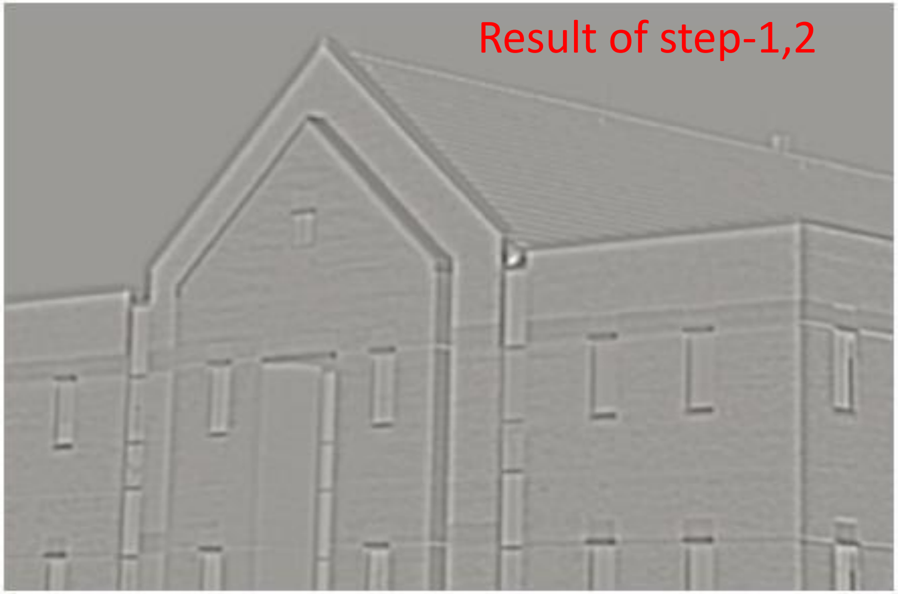
1. Filter the input image with an $n \times n$ Gaussian lowpass filter
2. Compute the Laplacian of the image resulting from Step 1 using, for example, the 3×3 mask
3. Find the zero crossings of the image from Step 2.

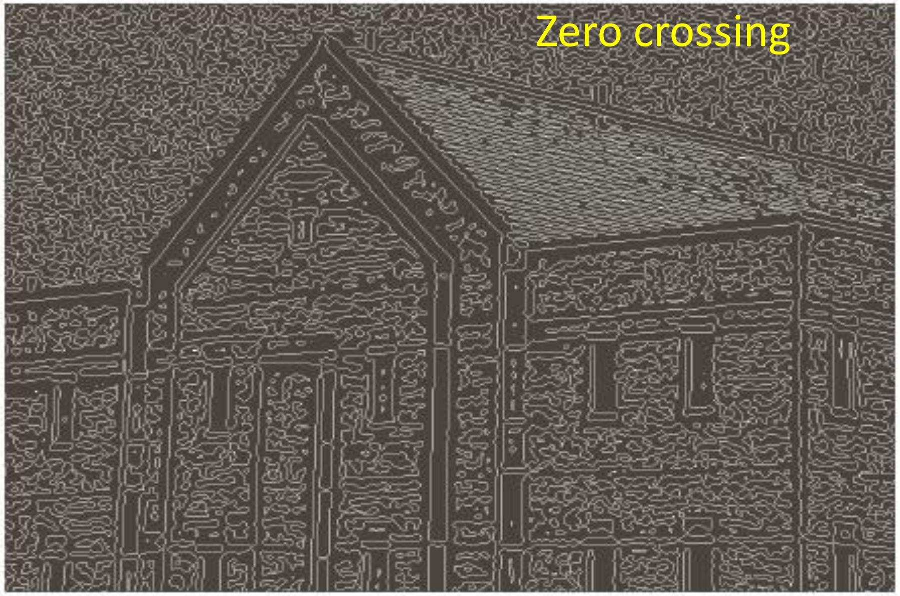
- What should be the size of the Gaussian filter mask nxn ?
- Most of the volume of Gaussian surface lies between $\pm 3\sigma$, so n is a smallest odd integer greater than or equal to 6σ Mask smaller than this will tend to truncate the LOG function.

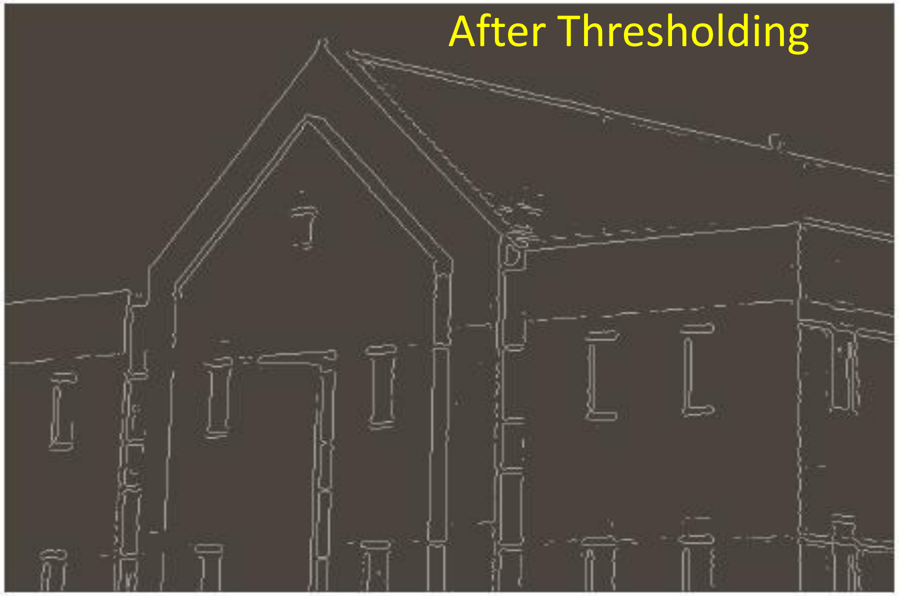
How do we find the zero crossings

- Consider 3x3 neighborhood across p, Zero crossing at p implies that signs of at least two of its opposing neighbour pixels must differ-4 cases to test-(up-down),(left-right),two diagonals.
- Absolute value of their numerical difference must also exceed the threshold

Marr- Hildreth edge detector (Example)







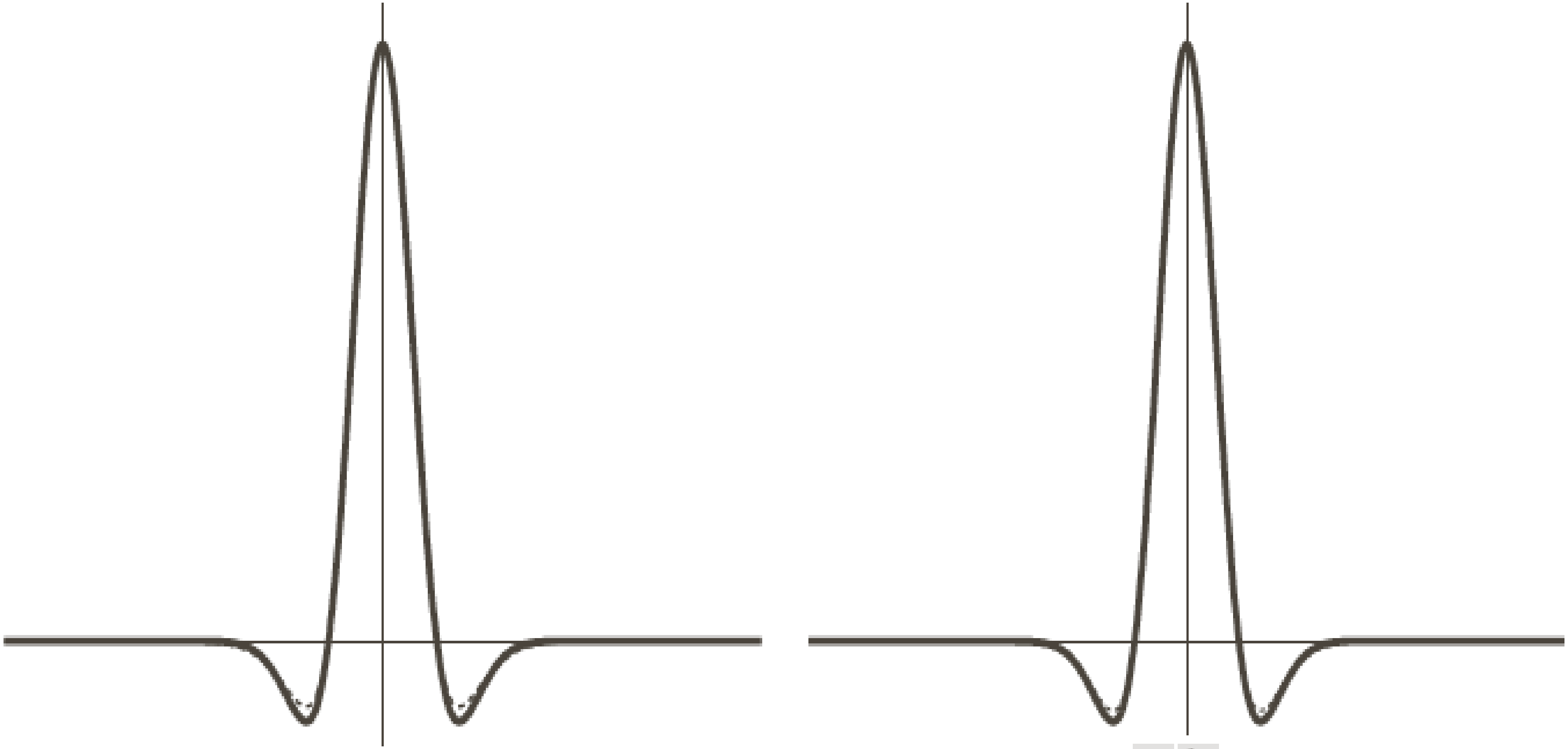
- To consider scale dependency-Filter the image for various values of sigma and keep the zero crossings that are common to all responses-complex procedure.
- Marr and Hildreth [1980] showed that LoG may be approximated by a difference of Gaussinas (DOG):

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

- Meaningful comparison between LoG and DoG may be obtained after selecting the value of σ for LoG so that LoG has the same zero crossings as DoG:

$$\sigma^2=\frac{\sigma_1^2\sigma_2^2}{\sigma_1^2-\sigma_2^2}\ln\left(\frac{\sigma_1^2}{\sigma_2^2}\right)$$

- By selecting this sigma LoG and DoG will have same zero crossings but to be compatible in amplitude scaling(normalization) is needed.



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.

The canny edge detector

- Objectives:

1. *Low error rate.* All edges should be found, and there should be no spurious responses. That is, 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. That is, the distance between a point marked as an edge by the detector and the center of the true edge should be minimum.
3. *Single edge point response.* The detector should return only one point for each true edge point. That is, the number of local maxima around the true edge should be minimum. This means that the detector should not identify multiple edge pixels where only a single edge point exists.

Canny edge detection algorithm:

-

1. Smooth the input image with a Gaussian filter.
2. Compute the gradient magnitude and angle images.
3. Apply nonmaxima suppression to the gradient magnitude image.
4. Use double thresholding and connectivity analysis to detect and link edges.

Step:1

- Canny has shown that the first derivative of the Gaussian closely approximates the operator that optimizes the product of *signal-to-noise ratio* and localization.

$$\frac{d}{dx} e^{-\frac{x^2}{2\sigma^2}} = \frac{-x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$

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

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

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

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

This operation is followed by computing the gradient magnitude and direction (angle), as discussed in Section 10.2.5:

Step:2

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

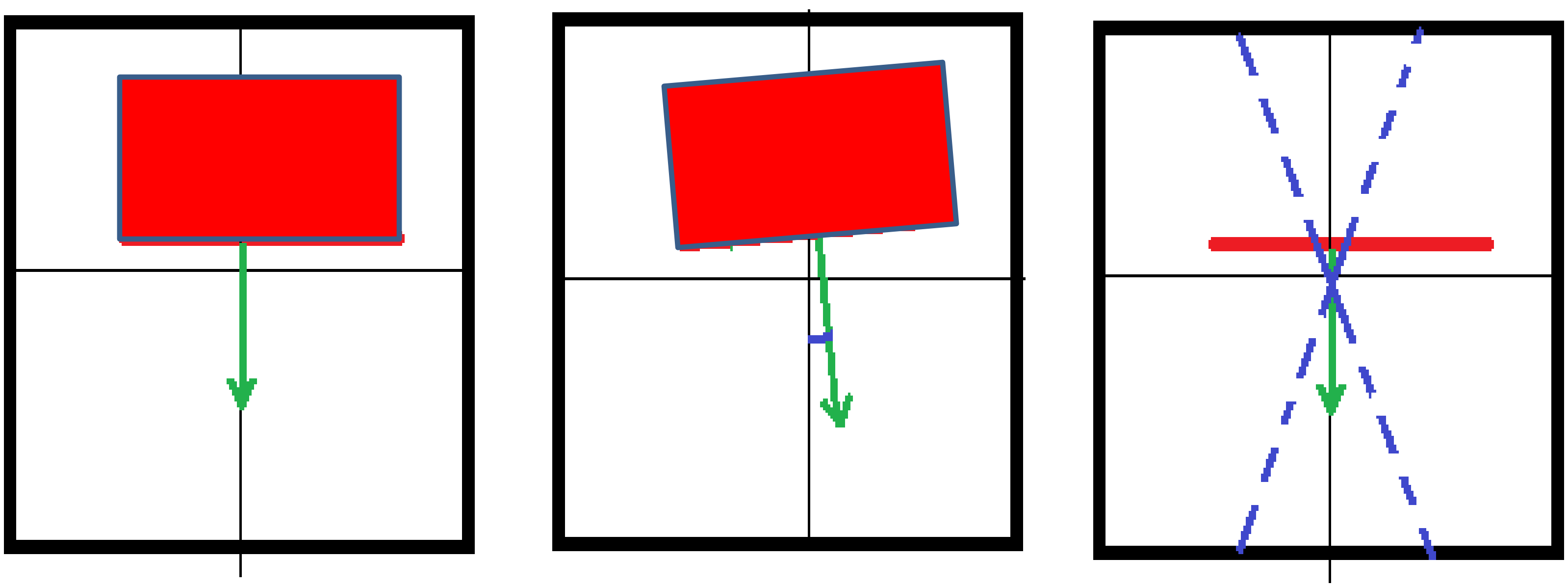
and

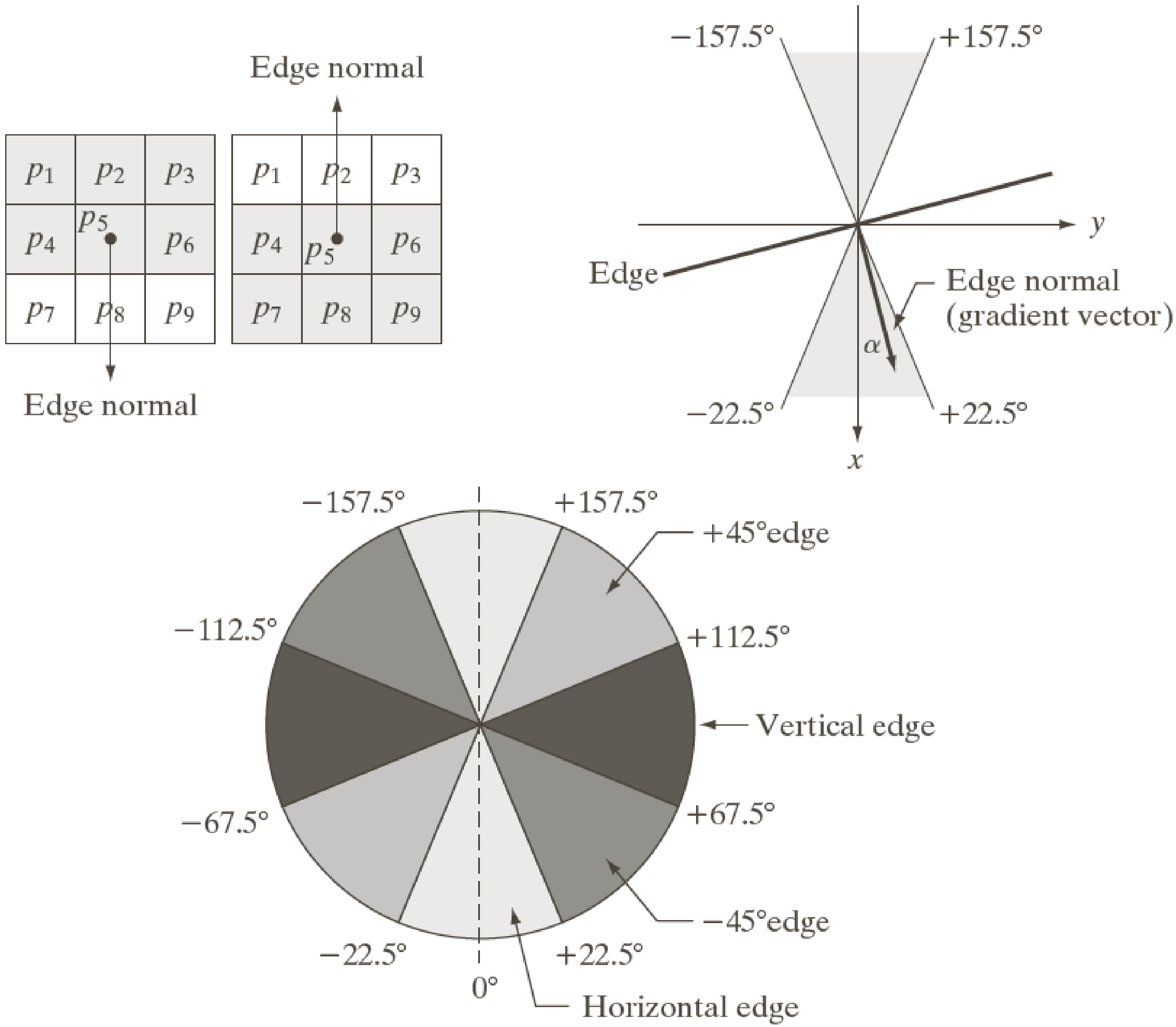
$$\alpha(x, y) = \tan^{-1}\left[\frac{g_y}{g_x}\right]$$

Step:3

Non maxima suppression:

- Gradient typically contains wide ridges around local maxima- non maxima suppression.
- Various ways to implement the same.





a b
c

FIGURE 10.24 (a) Two possible orientations of a horizontal edge (in gray) in a 3 × 3 neighborhood. (b) Range of values (in gray) 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 of gray.

Non maxima suppression:

Let d_1, d_2, d_3, and d_4 denote the four basic edge directions just discussed for a 3×3 region: horizontal, $-45°$, vertical, and $+45°$, respectively. We can formulate the following nonmaxima suppression scheme for a 3×3 region centered at *every* point (x, y) in $\alpha(x, y)$:

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 $g_N(x, y) = 0$ (suppression); otherwise, let $g_N(x, y) = M(x, y)$

Step:4

Thresholding for canny edge detection:

- Single threshold-
- Low value- false positive
- High value- false negative
- Canny’s algorithm implements hysteresis thresholding.- uses two thresholds.
- Canny suggested that ratio of the high to low thresholds should be 2:1 or 3:1

Thresholding for canny edge detection:

Two additional images are created.

$$g_{NH}(x, y) = g_N(x, y) \geq T_H$$

$$g_{NL}(x, y) = g_N(x, y) \geq T_L$$

$g_{NH}(x, y)$ will have fewer nonzero pixels than $g_{NL}(x, y)$ in general, but all the nonzero pixels in $g_{NH}(x, y)$ will be contained in $g_{NL}(x, y)$ because the latter image is formed with a lower threshold.

$$g_{NL}(x, y) = g_{NL}(x, y) - g_{NH}(x, y)$$

The nonzero pixels in $g_{NH}(x, y)$ and $g_{NL}(x, y)$ may be viewed as being "strong" and "weak" edge pixels, respectively.

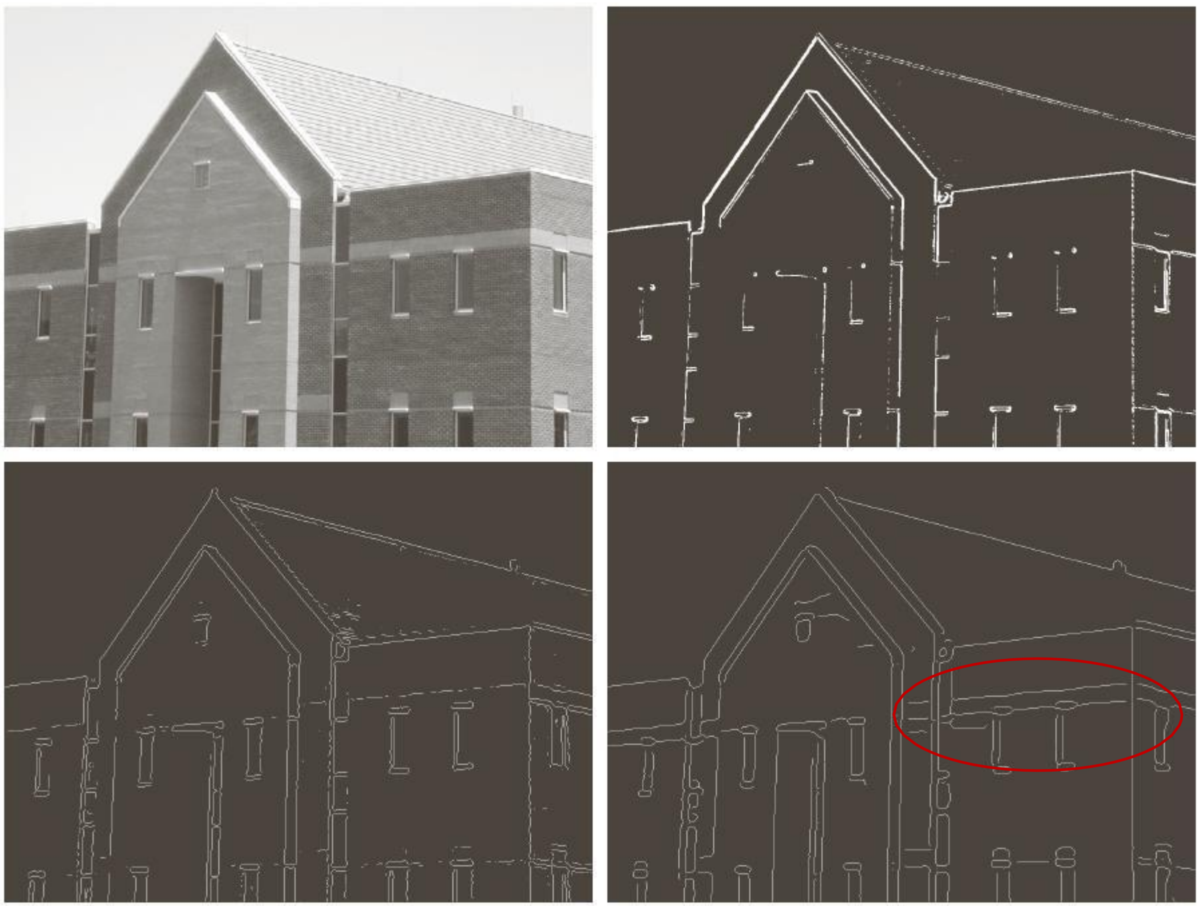
After the thresholding operations, all strong pixels in $g_{NH}(x, y)$ are assumed to be valid edge pixels and are so marked immediately. Depending on the value of T_H, the edges in $g_{NH}(x, y)$ typically have gaps. Longer edges are formed using the following procedure:

(a) Locate the next unvisited edge pixel, p, in $g_{NH}(x, y)$.
(b) Mark as valid edge pixels all the weak pixels in $g_{NL}(x, y)$ that are connected to p using, say, 8-connectivity.
(c) If all nonzero pixels in $g_{NH}(x, y)$ have been visited go to Step d. Else, return to Step a.
(d) Set to zero all pixels in $g_{NL}(x, y)$ that were not marked as valid edge pixels.

At the end of this procedure, the final image output by the Canny algorithm is formed by appending to $g_{NH}(x, y)$ all the nonzero pixels from $g_{NL}(x, y)$.

Canny vs LoG

Image Thresholded gradient

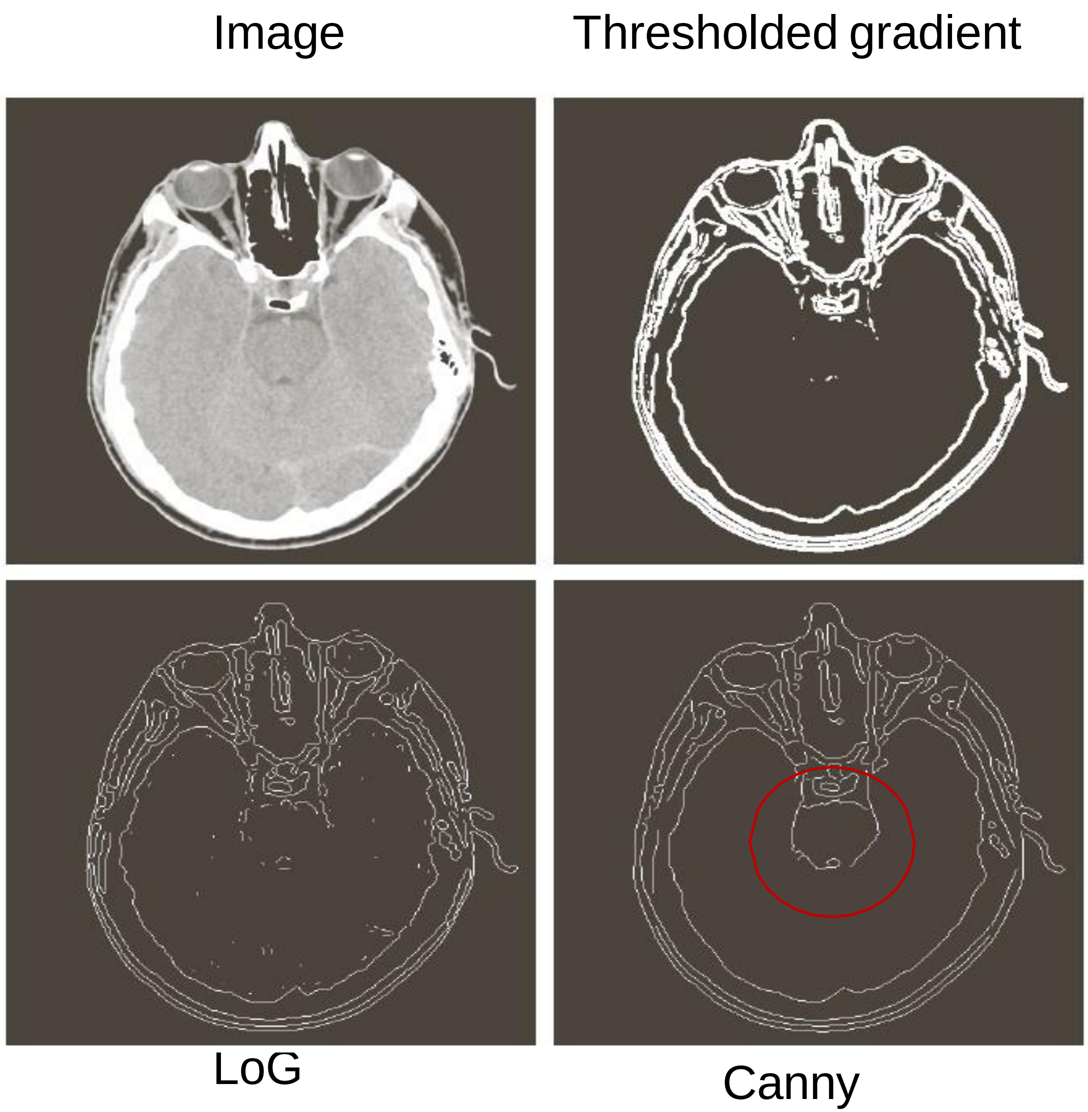


LoG Canny

Both edges of the concrete band lining the bricks were preserved.

Not a single edge due to roof tiles.

Quality of the lines with regard to continuity, thinness and straightness is superior



The posterior boundary of the brain was preserved.

Price paid for the improved performance is more complex implementation and more execution time

Edge linking:

- Even after hysteresis thresholding, the detected pixels do not completely characterize edges completely due to occlusions, non-uniform illumination and noise. Edge linking may be:

  - **Local**: requiring knowledge of edge points in a small neighbourhood.
  - **Regional**: requiring knowledge of edge points on the boundary of a region.
  - **Global**: the Hough transform, involving the entire edge image.

Edge Linking by Local Processing

A simple algorithm:

1. Compute the gradient magnitude and angle arrays $M(x,y)$ and $\alpha(x,y)$ of the input image $f(x,y)$.
2. Let S_{xy} denote the neighborhood of pixel (x,y).
3. A pixel (s,t) in S_{xy} in is linked to (x,y) if:

$$M(x,y) - M(s,t) \le E \text{ and } \qquad \alpha(x,y) - \alpha(s,t) \le A$$

Computationally expensive as all neighbours of every pixel should be examined.

A faster algorithm:

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:

$$g(x,y)=\begin{cases}1 & M(x,y)\geq T_M \text{ and } \alpha(x,y)\in[A-T_A, A+T_A]\\ 0 & \text{otherwise}\end{cases}$$

3. Scan the rows of $g(x,y)$ and fill (set to 1) all gaps (zeros) that do not exceed a specified length K.
4. To detect gaps in any other direction $A=\theta$, rotate $g(x,y)$ by θ and apply the horizontal scanning.

Edge Linking by Local Processing (cont.)

Image | Gradient magnitude | Horizontal linking

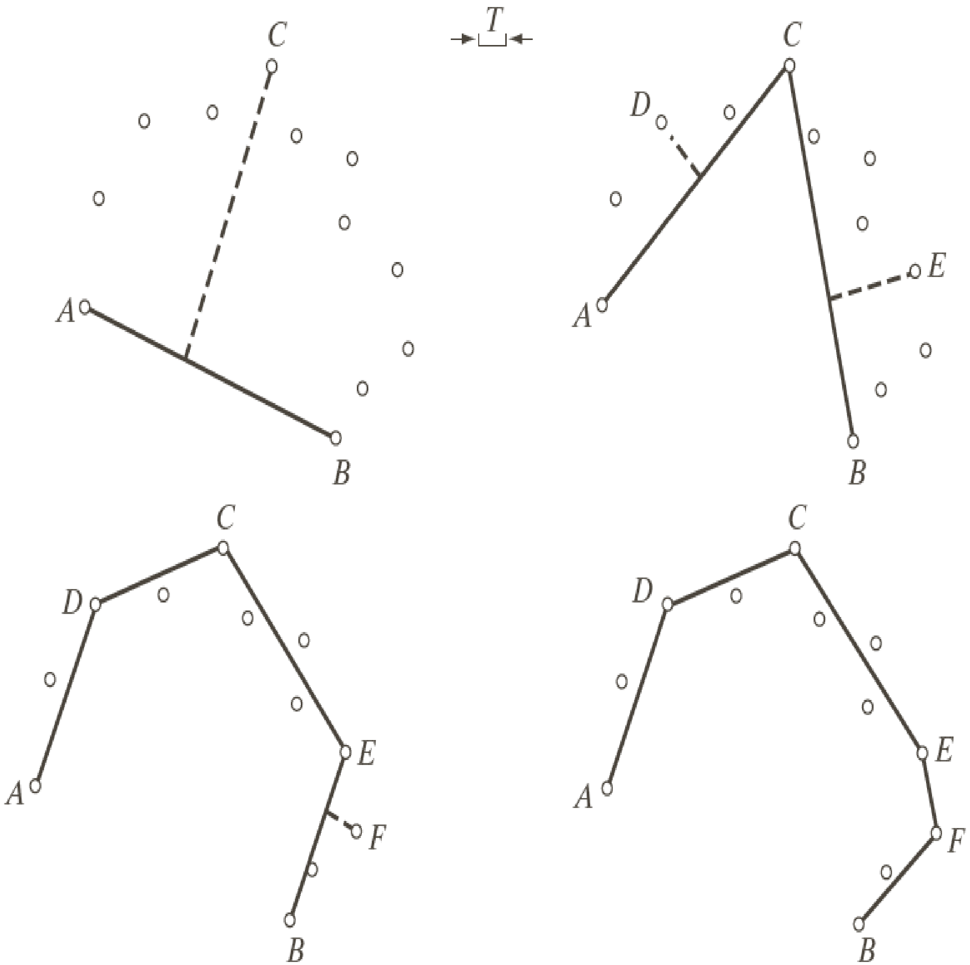
Vertical linking | Logical OR | Morphological thinning

We may detect the licence plate from the ratio width/length (2:1 in the USA)

Edge Linking by Regional processing:

- Often, the location of regions of interest is known and pixel membership to regions is available.
- Approximation of the region boundary by fitting a polygon. Polygons are attractive because:
  - They capture the essential shape.
  - They keep the representation simple.

Basic mechanism for polygon fitting.

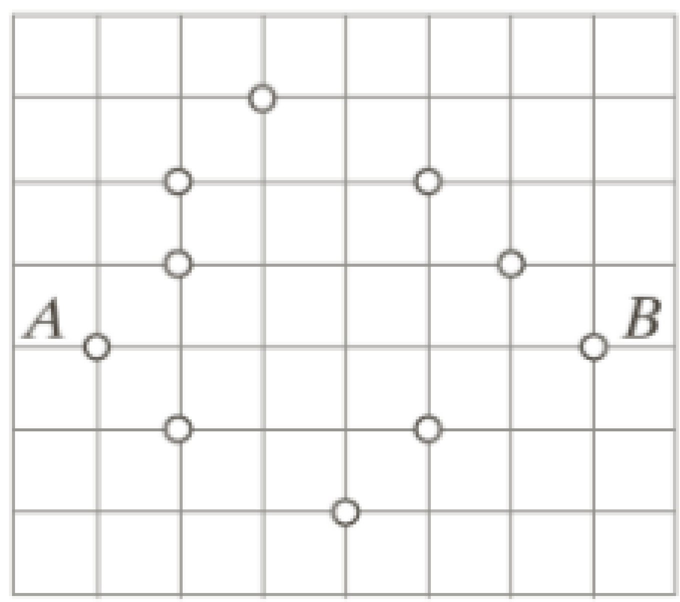


- Given the end points *A* and *B*, compute the straight line *AB*.
- Compute the perpendicular distance from all other points to this line.
- If this distance exceeds a threshold, the corresponding point *C* having the maximum distance from *AB* is declared a vertex.
- Compute lines *AC* and *CB* and continue.

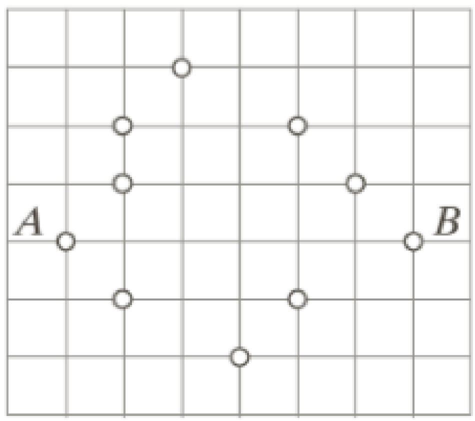
- Requirements
  - Two starting points must be specified (e.g. rightmost and leftmost points).
  - The points must be ordered (e.g. clockwise).
- Variations of the algorithm handle both open and closed curves.
- If this is not provided, it may be determined by distance criteria:
  - Uniform separation between points indicate a closed curve.
  - A relatively large distance between consecutive points with respect to the distances between other points indicate an open curve.

Algorithm:

1. Let P be the sequence of ordered, distinct, 1-valued points of a binary image. Specify two starting points, A and B.
2. Specify a threshold, T, and two empty stacks, OPEN and ClOSED.



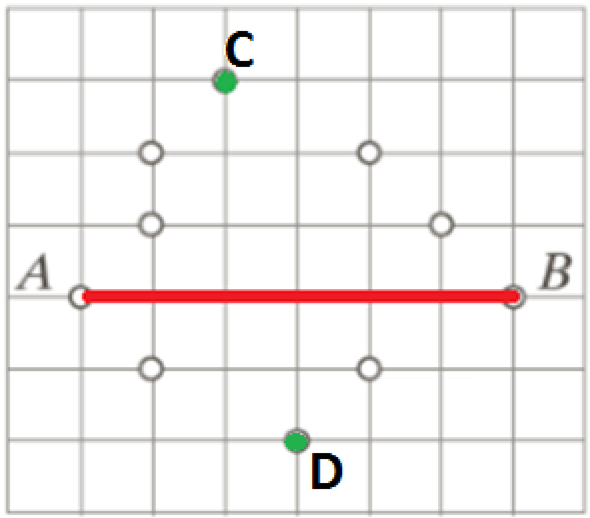
3. If the points in P correspond to a closed curve, put A into OPEN and put B into OPEN and CLOSED. If the points correspond to an open curve, put A into OPEN and B into CLOSED.



CLOSED	OPEN
B	B,A

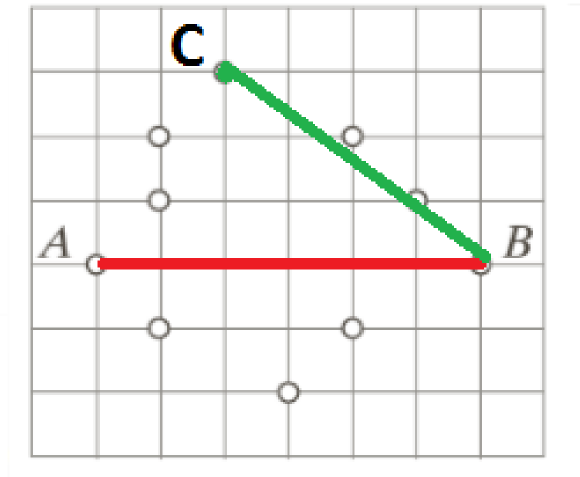
4. Compute the parameters of the line passing from the last vertex in CLOSED to the last vertex in OPEN (LINE: BA)

5. Compute the distances from the line in Step 4 to all the points in P whose sequence places them between the vertices from Step 4. Select the point, Vmax, with the maximum distance, Dmax



6. If Dmax> T, place Vmax at the end of the OPEN stack as a new vertex. Go to step 4.

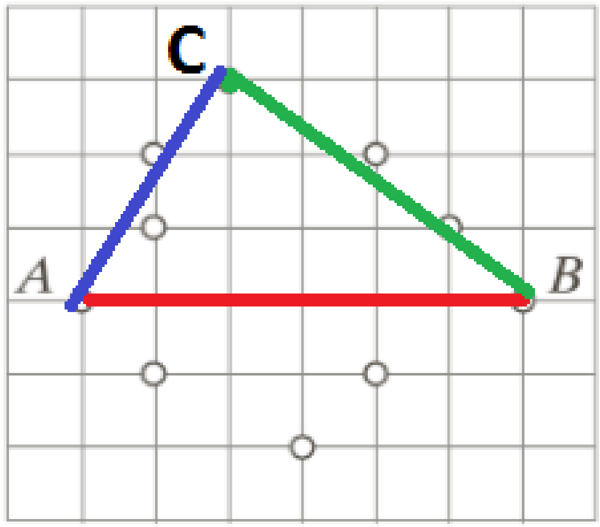
CLOSED	OPEN	CURVE SEGMENT SELECTED IN STEP :4
B	B,A,C	BC



- STEP: 5,6 Dmax does not exceed threshold.

7. Else, remove the last vertex from OPEN and insert it as the last vertex of CLOSED.
8. If OPEN is not empty, go to step 4.

CLOSED	OPEN	CURVE SEGMENT SELECTED IN STEP :4
B,C	B,A	CA



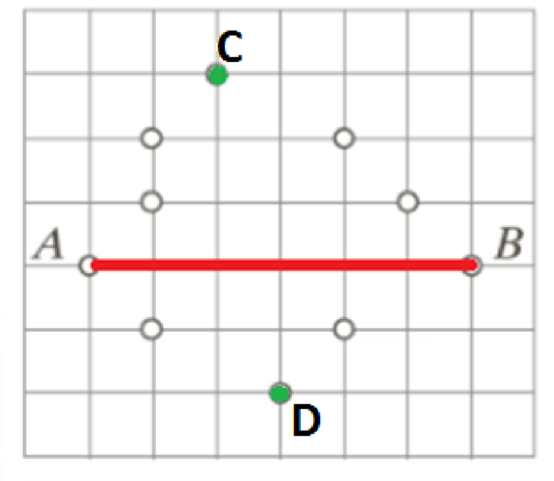
- STEP: 5,6 Dmax does not exceed threshold.

7. Else, remove the last vertex from OPEN and insert it as the last vertex of CLOSED.

8. If OPEN is not empty, go to step 4.

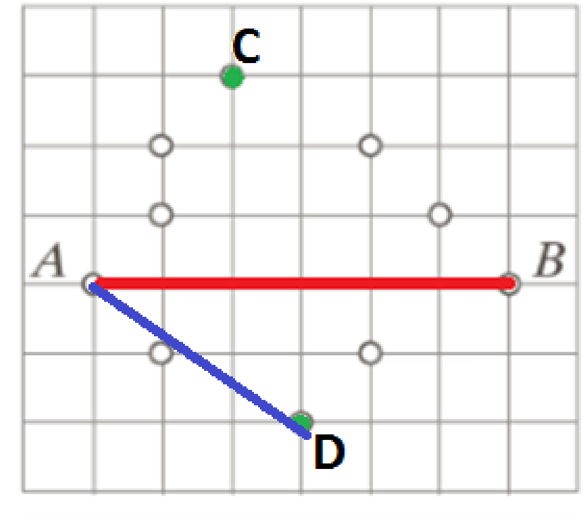
CLOSED	OPEN	CURVE SEGMENT SELECTED IN STEP :4
B,C,A	B	AB

5. Compute the distances from the line in Step 4 to all the points in P whose sequence places them between the vertices from Step 4. Select the point, Vmax, with the maximum distance, Dmax



6. If Dmax> T, place Vmax at the end of the OPEN stack as a new vertex. Go to step 4.

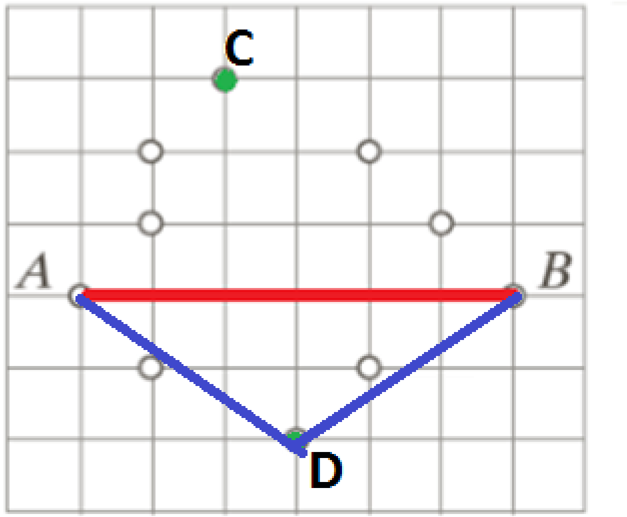
CLOSED	OPEN	CURVE SEGMENT SELECTED IN STEP :4
B,C,A	B,D	AD



- STEP: 5,6 Dmax does not exceed threshold.

7. Else, remove the last vertex from OPEN and insert it as the last vertex of CLOSED.
8. If OPEN is not empty, go to step 4.

CLOSED	OPEN	CURVE SEGMENT SELECTED IN STEP :4
B,C,A,D	B	BD



- STEP: 5,6 Dmax does not exceed threshold.

7. Else, remove the last vertex from OPEN and insert it as the last vertex of CLOSED.
8. If OPEN is not empty, go to step 4.

CLOSED	OPEN	CURVE SEGMENT SELECTED IN STEP :4
B,C,A,D,B	-	-

9. Else, exit. The vertices in CLOSED are the vertices of the polygonal fit to the points in P.

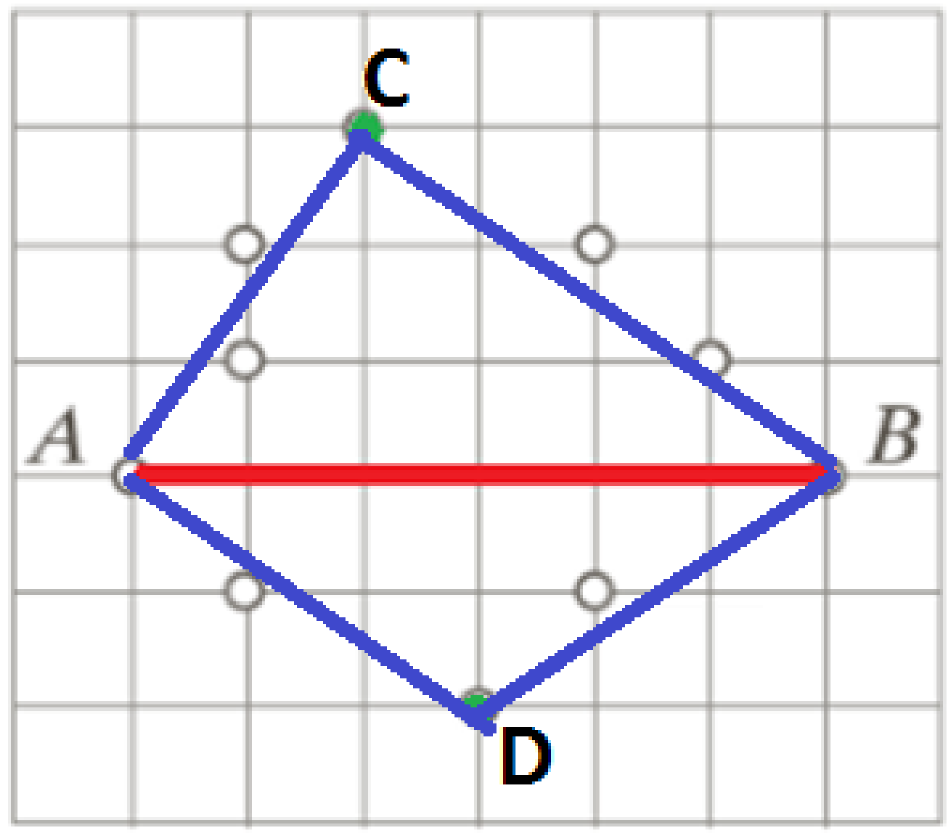

C
A
B
D

TABLE 10.1 Step-by-step details of the mechanics in Example 10.11.

CLOSED	OPEN	Curve segment processed	Vertex generated
B	B, A	–	A, B
B	B, A	(BA)	C
B	B, A, C	(BC)	–
B, C	B, A	(CA)	–
B, C, A	B	(AB)	D
B, C, A	B, D	(AD)	–
B, C, A, D	B	(DB)	–
B, C, A, D, B	Empty	–	–

- Regional processing for edge linking is used in combination with other methods in a chain of processing.

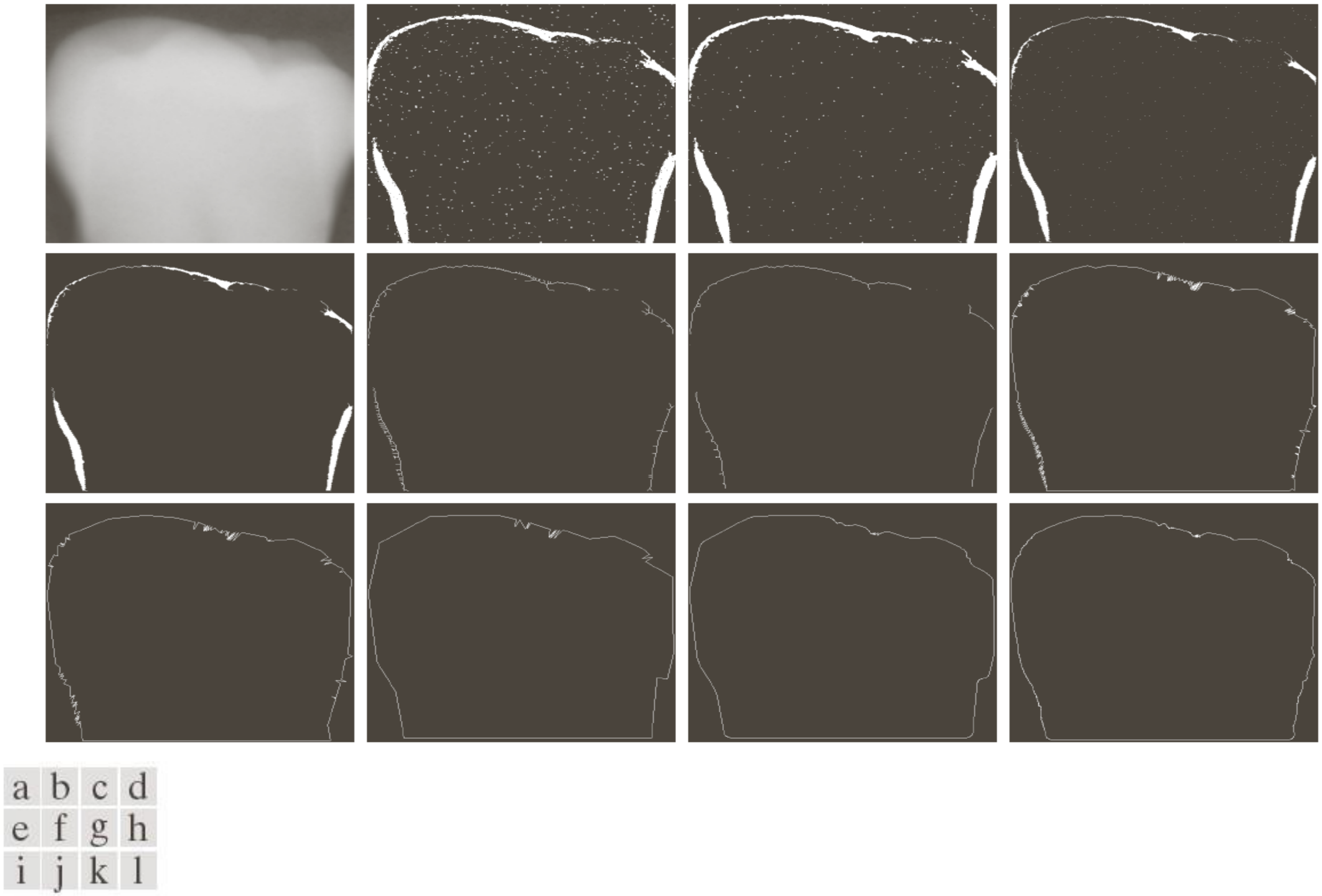


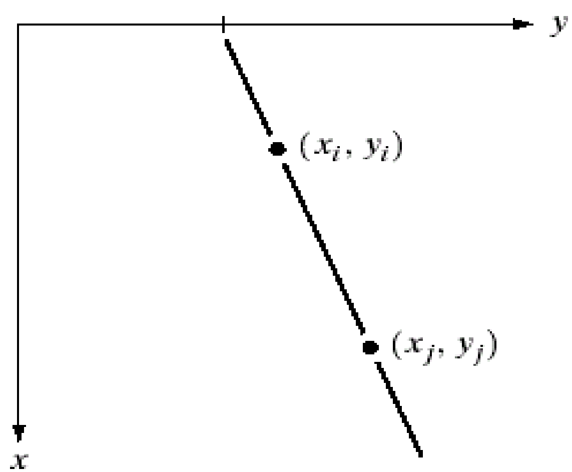
FIGURE 10.30 (a) A 550 × 566 X-ray image of a human tooth. (b) Gradient image. (c) Result of majority filtering. (d) Result of morphological shrinking. (e) Result of morphological cleaning. (f) Skeleton. (g) Spur reduction. (h)–(j) Polygonal fit using thresholds of approximately 0.5%, 1%, and 2% of image 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.

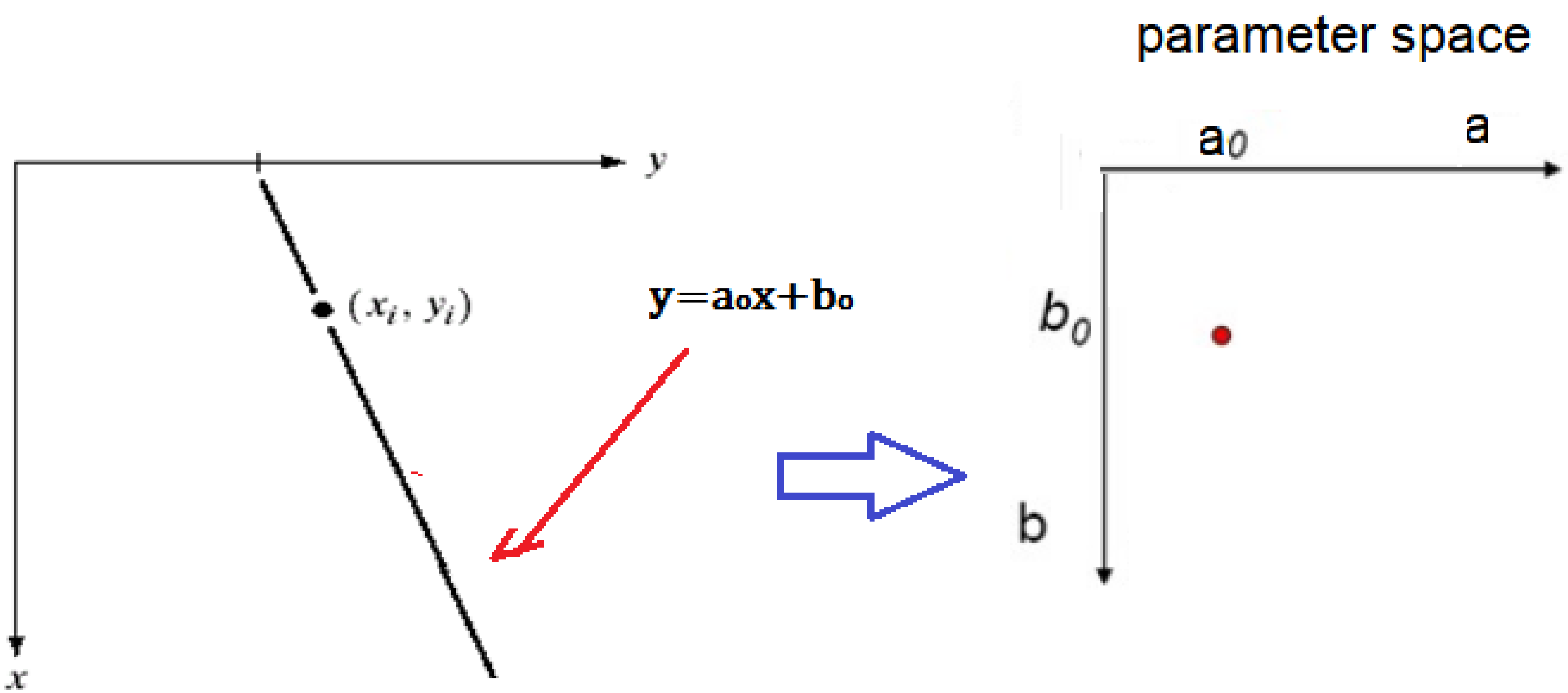
Advantages:

- Simple to implement and yields results that generally are acceptable.

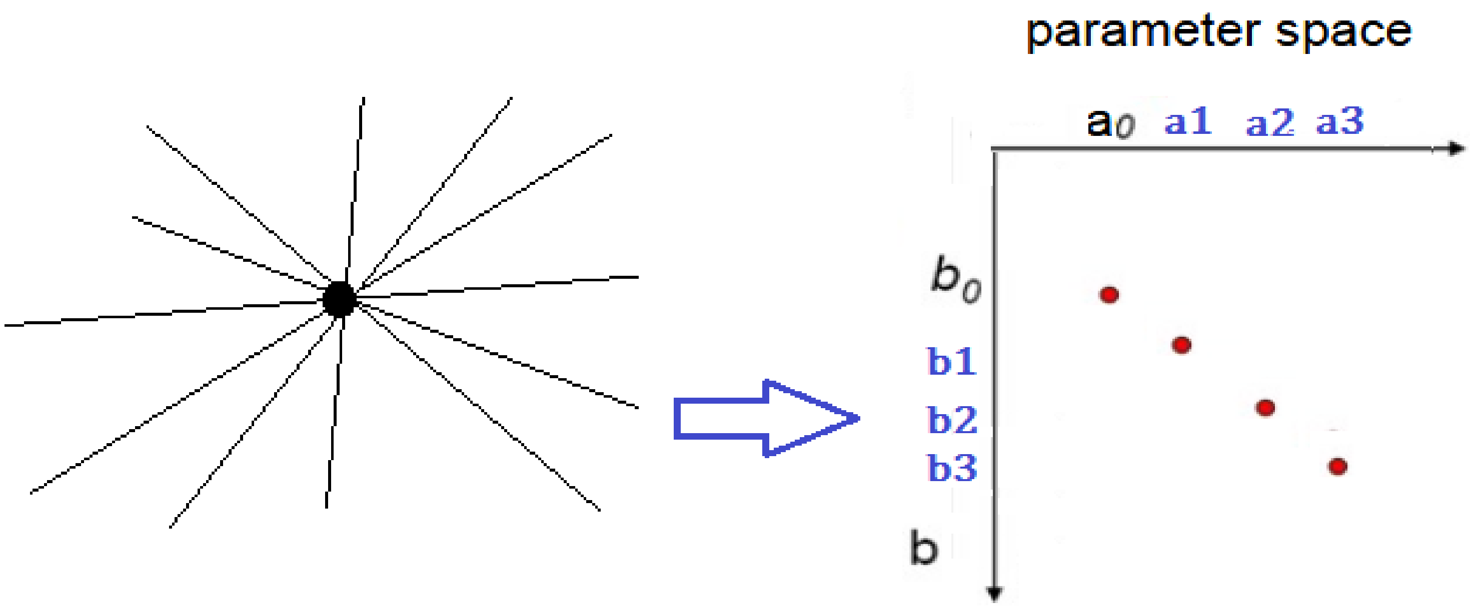
Hough transform:

- Each line is defined by two points (x_i, y_i) and (x_j, y_j).
- Equation of which can be given by : y = ax + b
- Points (xi,yi) and (xj,yj) (and all other points on line) have a line of parameter (a,b) which passes through them.



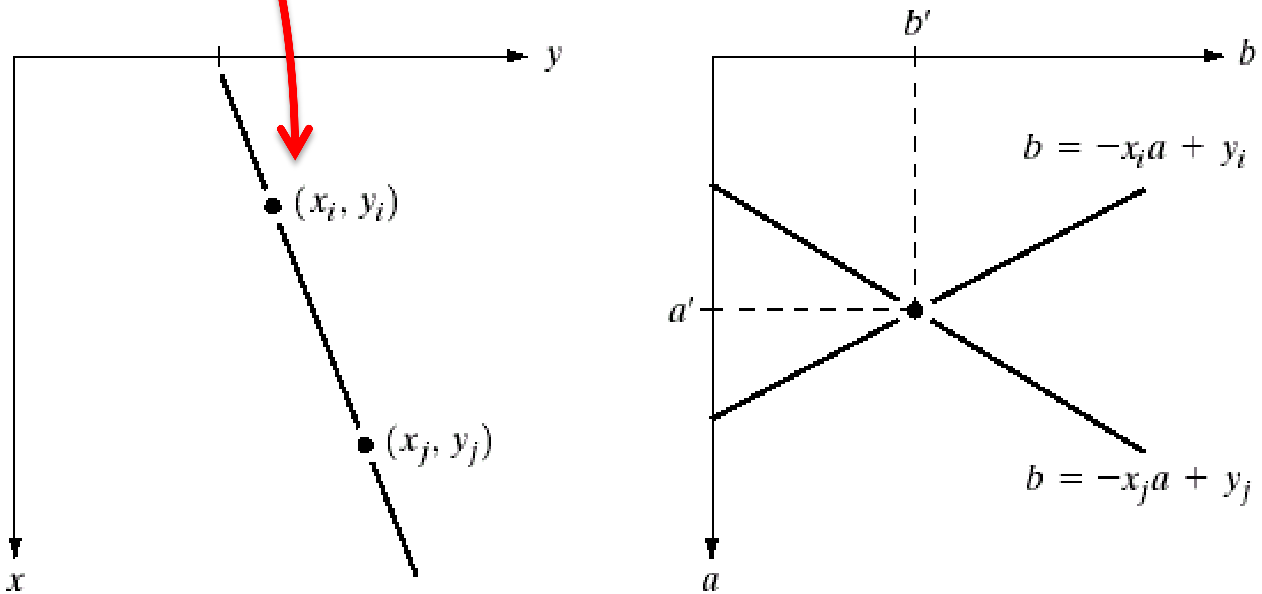

parameter space
y
x
(x_i, y_i)
$y=a_0x+b_0$
a_0
a
b_0
b

- Infinitely many line passes through point (x_i,y_i)
- Equations of which can be given by ➔ y=a1x+b1, y=a2x+b2 etc
- All will satisfy $y_i=ax_i+b$ for varying values of a and b.
- Writing equation as b= - x_i a+y_i ➔ its equation of a line.



For two points:

$y= a'x+b'$



a b

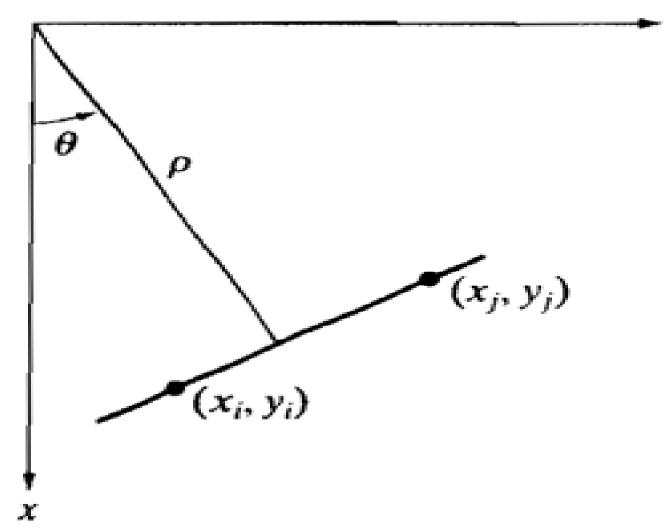
FIGURE 10.17 (a) xy-plane. (b) Parameter space.

All the points (x,y) on a line $y=a'x+b'$ have lines in parameter space that intersect at (a',b').

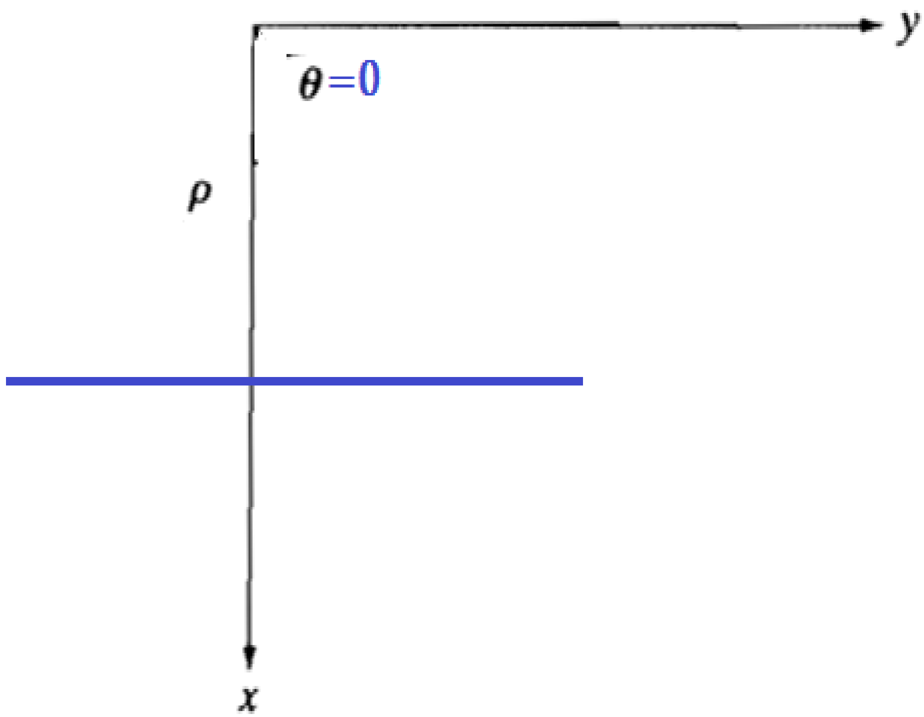
- In parameter space many lines intersect at one point- significance?
- In x-y plane those many points are on that line.
- The principle lines in x-y plane could be found by identifying points in parameter space where large numbers of parameter space lines intersects.
- Practical problem→ a is infinite for vertical line.
- Solution→ normal representation.

Normal representation of line:

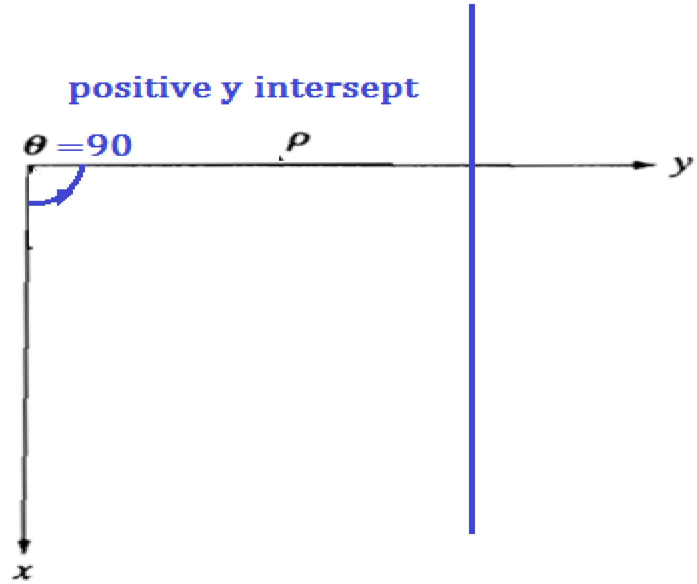
$$x \cos\theta + y \sin\theta = \rho$$

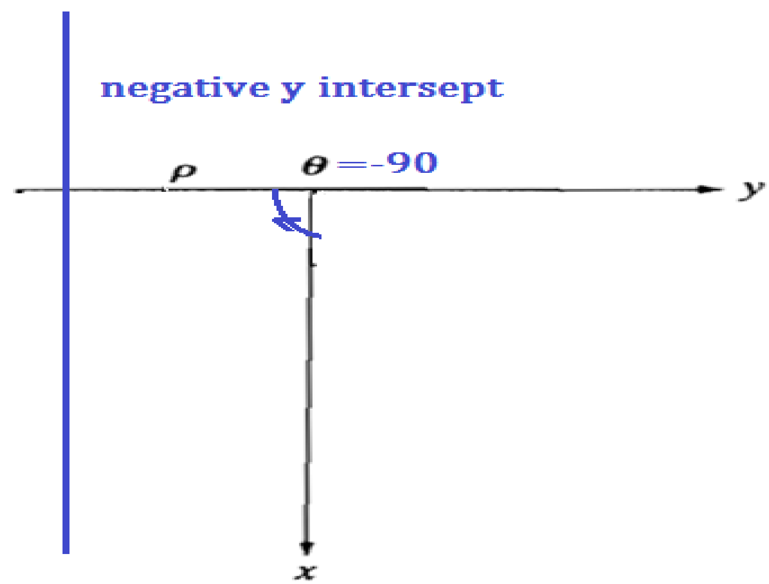


Horizontal line:

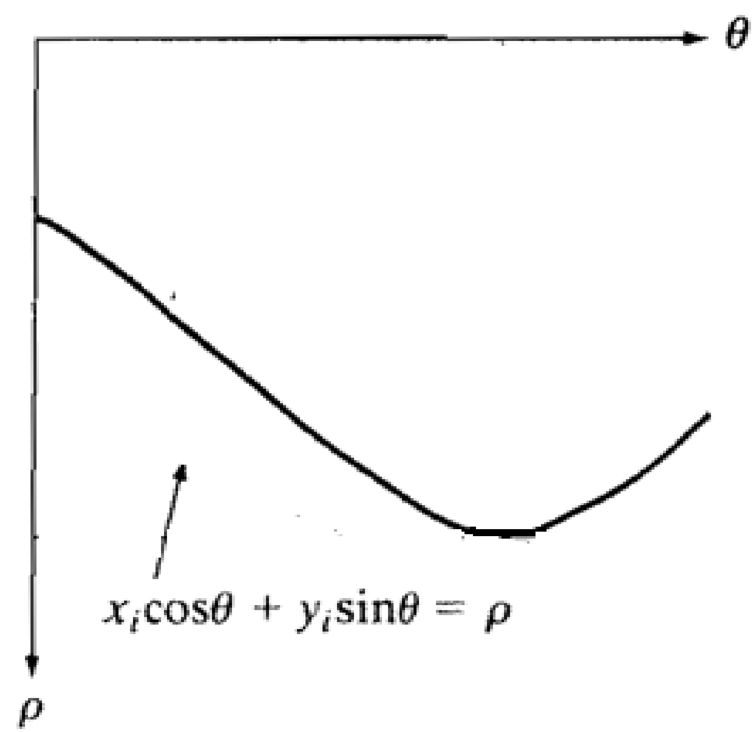


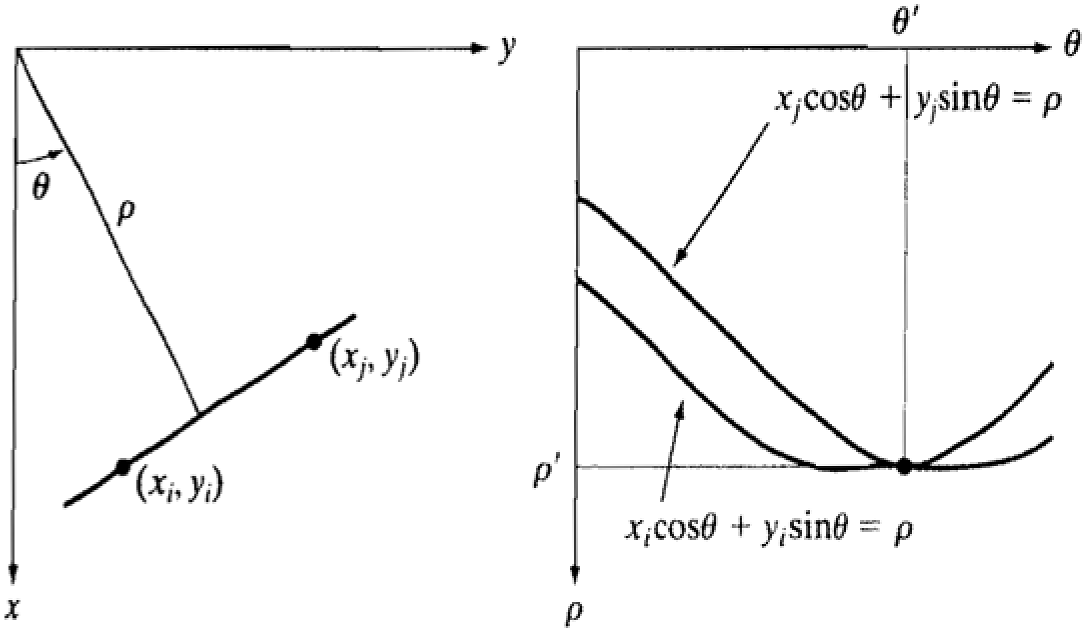
Vertical line:





- For a given point (xi,yi) number of line will pass, having varying values of (ρ , θ)

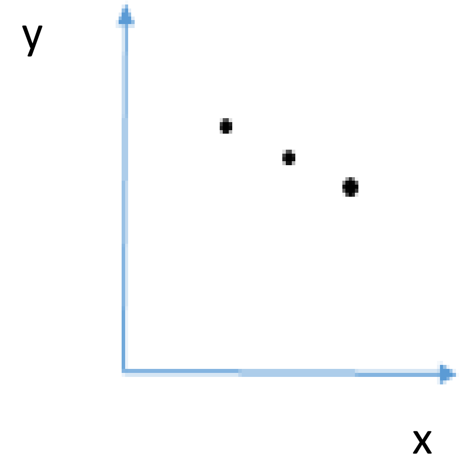




a b c

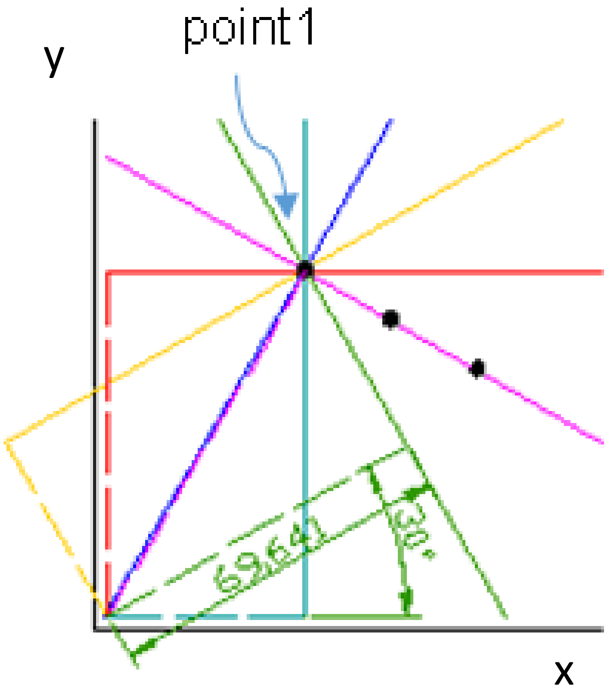
FIGURE 10.32 (a) (ρ, θ) parameterization of line in the xy-plane. (b) Sinusoidal curves in the $\rho\theta$-plane; the point of intersection (ρ', θ') corresponds to the line passing through points (x_i, y_i) and (x_j, y_j) in the xy-plane. (c) Division of the $\rho\theta$-plane into accumulator cells.

Question: Is there a straight line connecting the 3 dots?

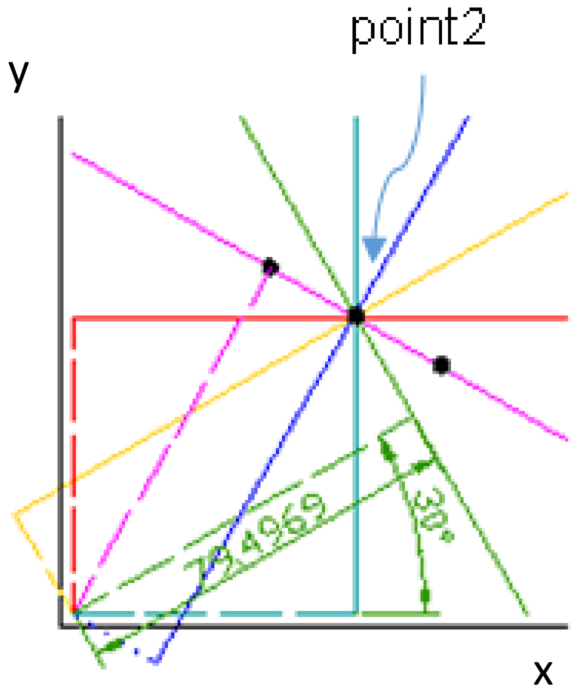


Procedure:

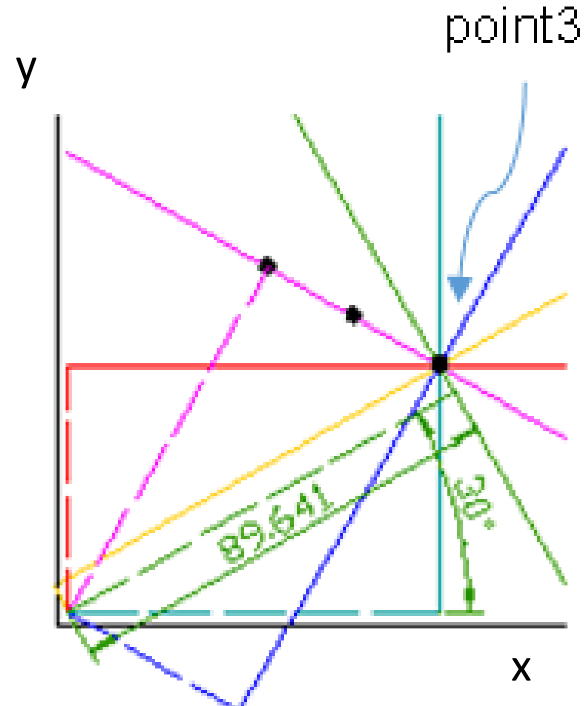
- For each data point, a number of lines are plotted going through it, all at different angles. These are shown here as solid lines.
- For each solid line a line is plotted which is perpendicular to it and which intersects the origin. These are shown as dashed lines.
- The length (i.e. perpendicular distance to the origin) and angle of each dashed line is measured. The results are shown in tables.
- This is repeated for each data point.
- A graph of the line lengths for each angle, known as a Hough space graph, is then created.



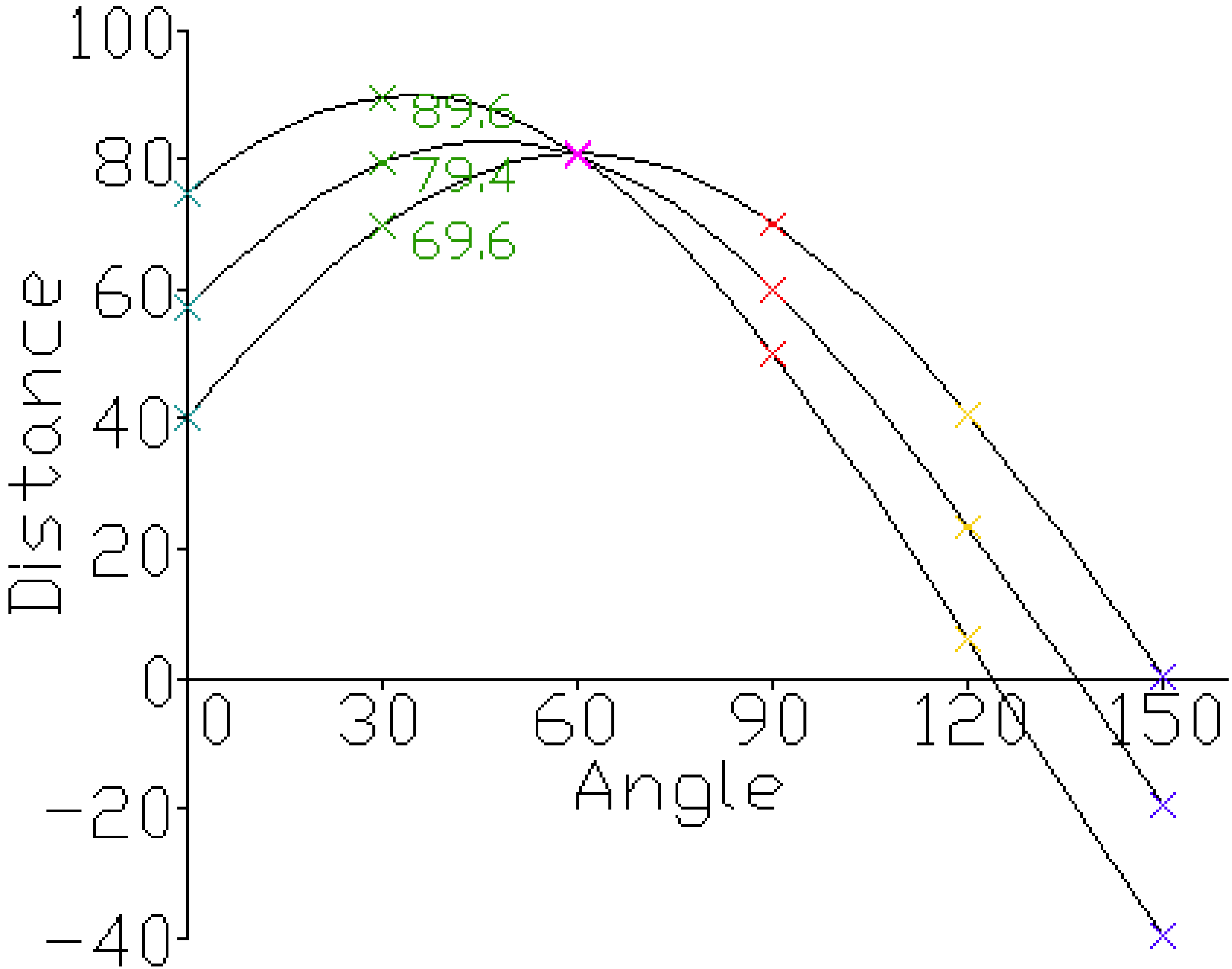
Angle	Dist.
0	40
30	69.6
60	81.2
90	70
120	40.6
150	0.4



Angle	Dist.
0	57.1
30	79.5
60	80.5
90	60
120	23.4
150	−19.5

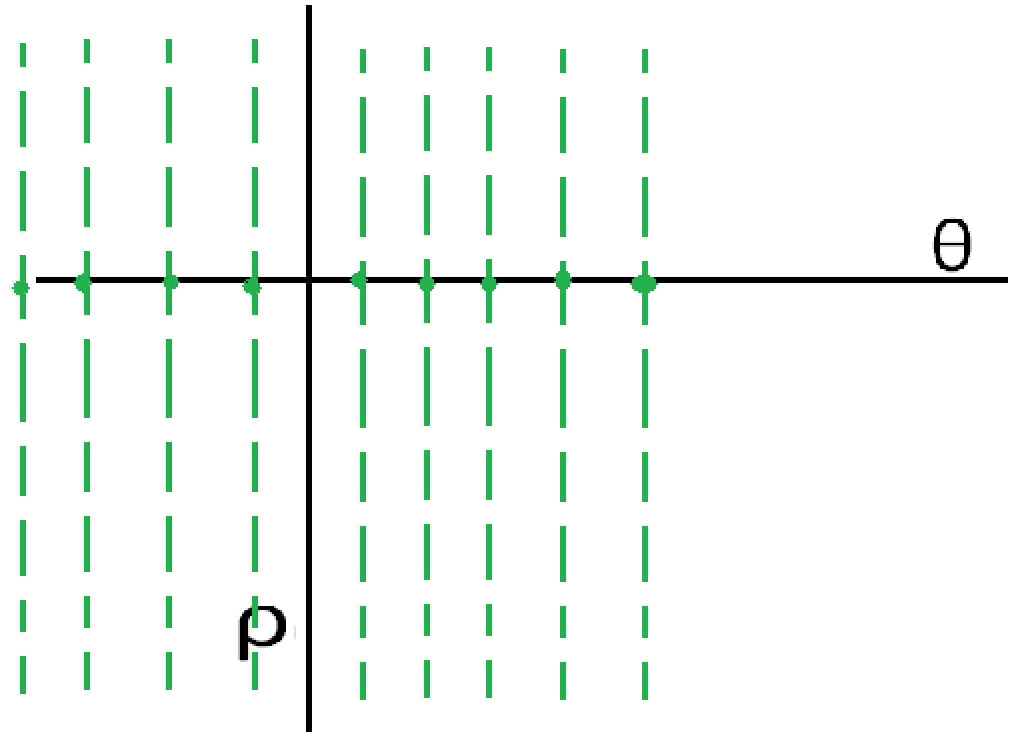


Angle	Dist.
0	74.6
30	89.6
60	80.6
90	50
120	6.0
150	−39.6

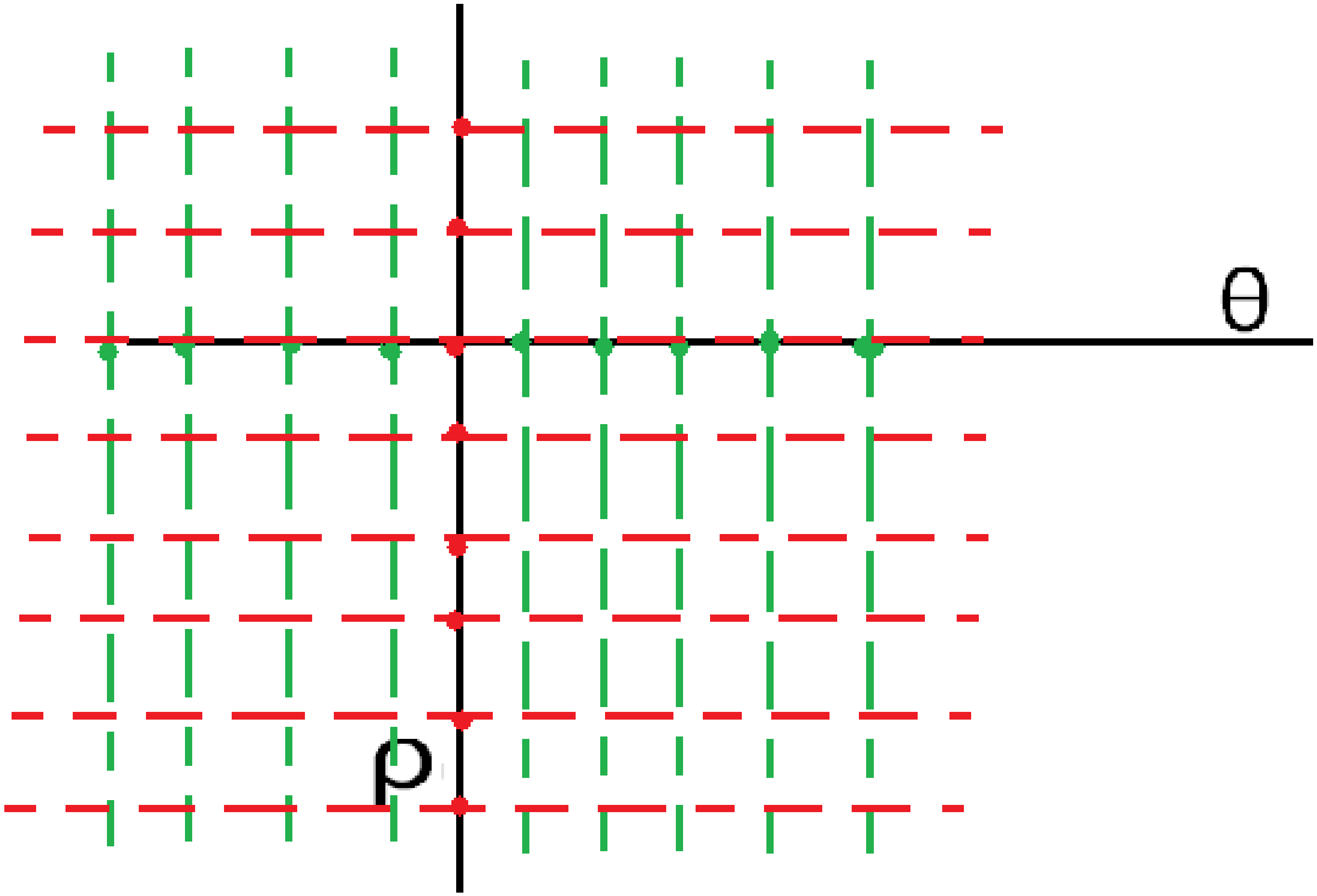


100
80
60
40
20
0
−20
−40
Distance
0
30
60
90
120
150
Angle
89.6
79.4
69.6

ρθ -plane to Accumulator cells:

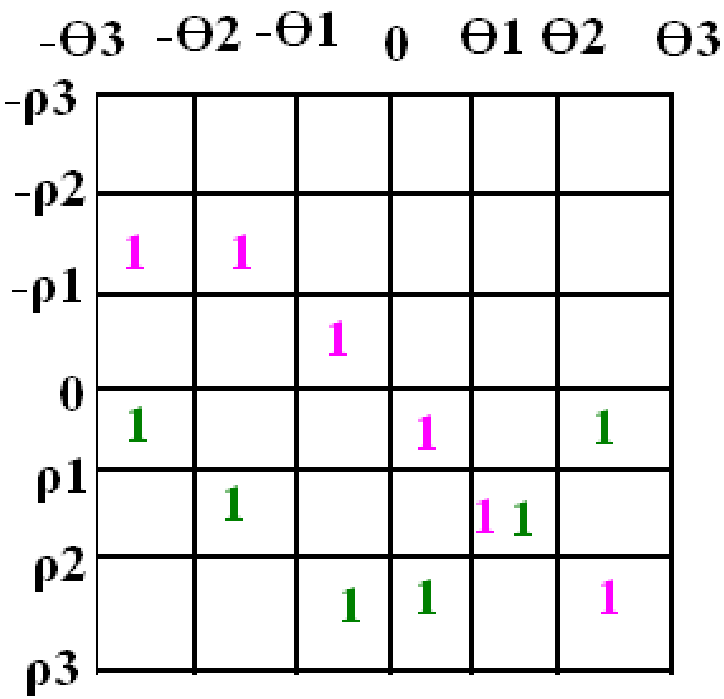


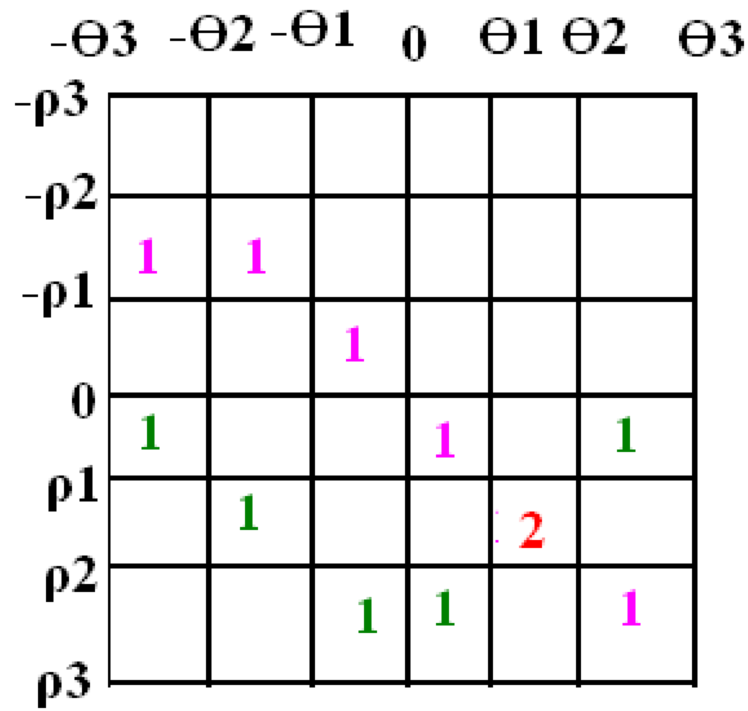
-90 degree <= θ<=90 degree

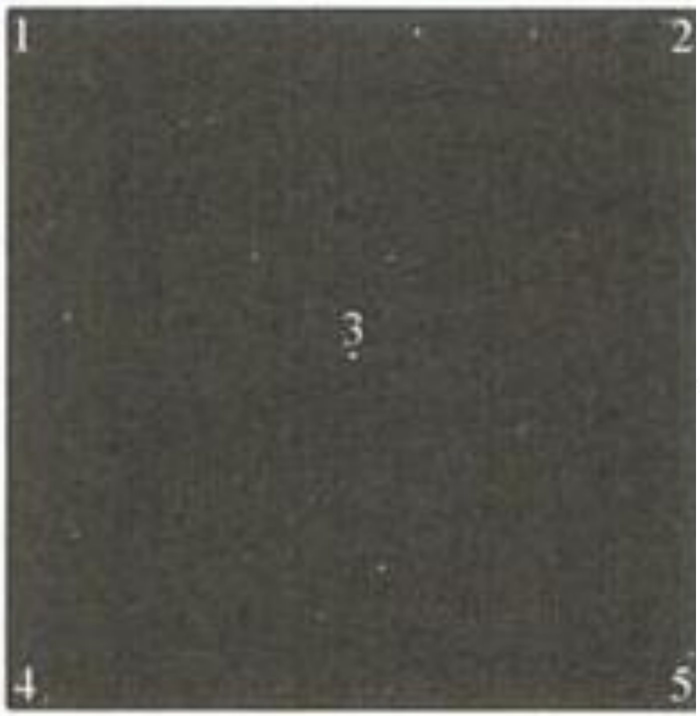


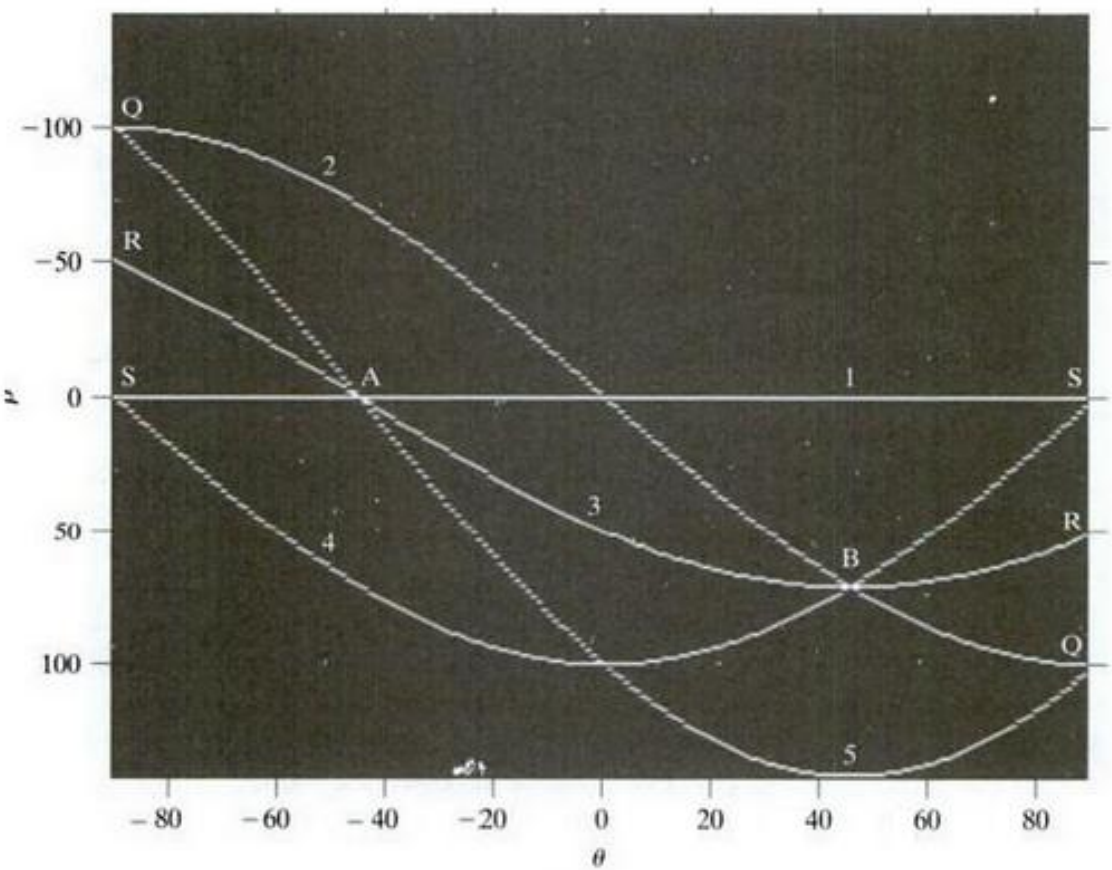
-D<=ρ <=D where D os the maximum distance between opposite corners in an image.

Accumulator cells:



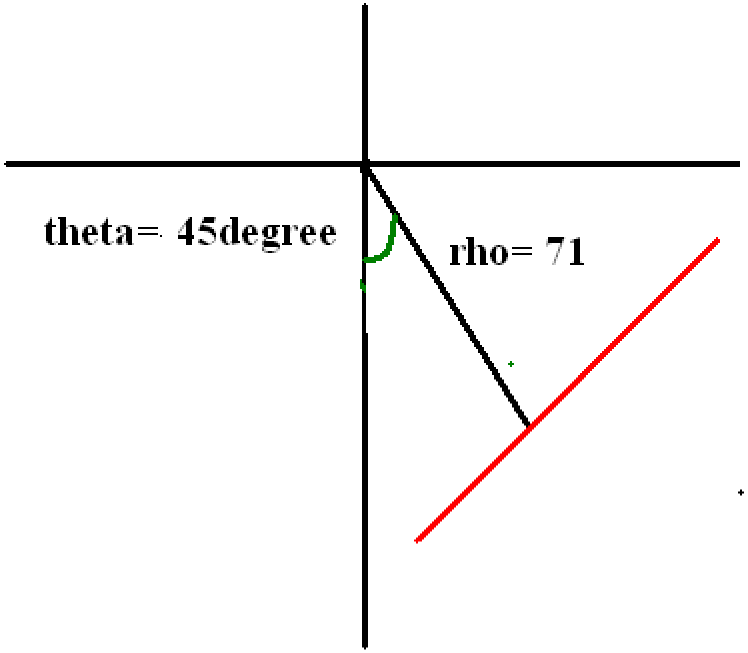



1
2
3
4
5


O
2
R
S
A
1
S
3
R
4
B
Q
5
−100
−50
0
50
100
ρ
− 80
− 60
− 40
− 20
0
20
40
60
80
θ

Consider point A:

Consider point B:



What are the parameters for line joining points 1 and 4?

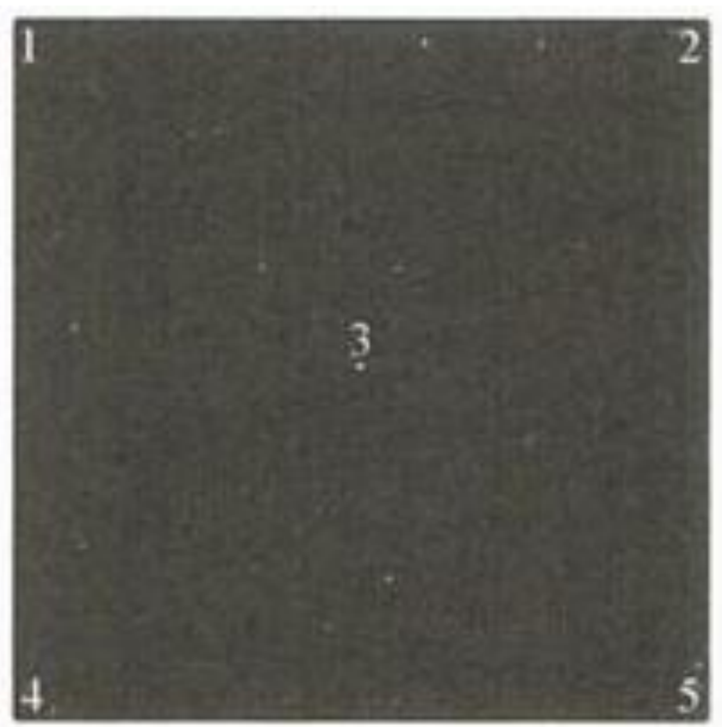
Passing through the origin so rho=0 and vertical line so angle=90

Point S

Observe periodicity

Line at angle -90 is also vertical.

What are the parameters for line joining points (1&2) and (2&5)?



1-2: Passing through the origin so rho=0 and vertical line so angle=0

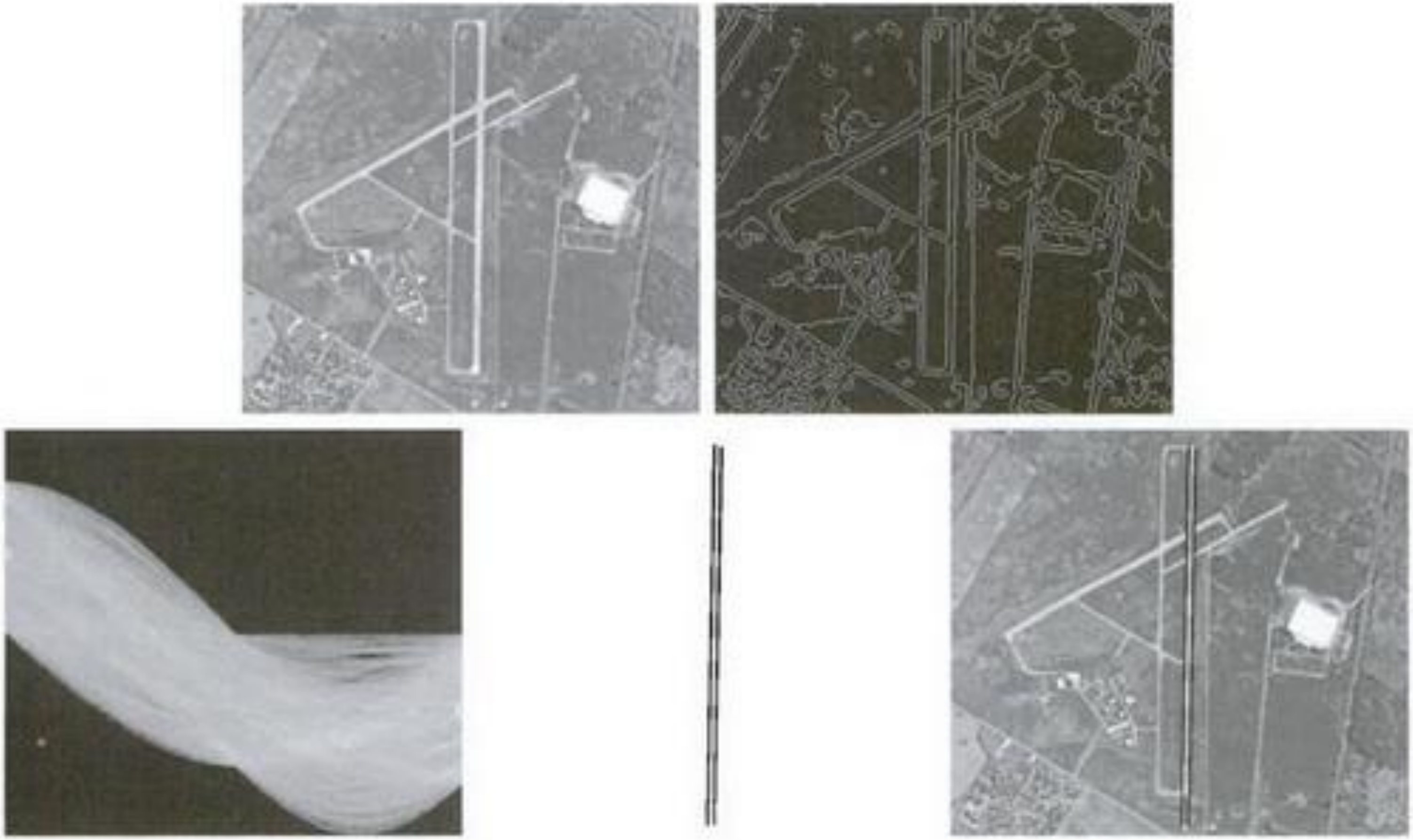
2-5: At 100 distance from origin and angle 90

Edge linking approach based on Hough transform:

1. Obtain a *binary* edge image using any of the techniques discussed earlier in this section.
2. Specify subdivisions in the $\rho\theta$-plane.
3. Examine the counts of the accumulator cells for high pixel concentrations.
4. Examine the relationship (principally for continuity) between pixels in a chosen cell.

- To check continuity- compute the distance between disconnected pixels corresponding to a given accumulator cell
- Bridge them if gap is less than a specified threshold.
- Global concept- applicable over entire image
- Pixels associated with specific accumulator cells are examined.

Extract two edges of airport runway



a b
c d e

FIGURE 10.34 (a) A 502 × 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 is applicable to any function of the form g(v,c)=0 where v is the vector of coordinates, c is the vector of coefficients
- For example for circle

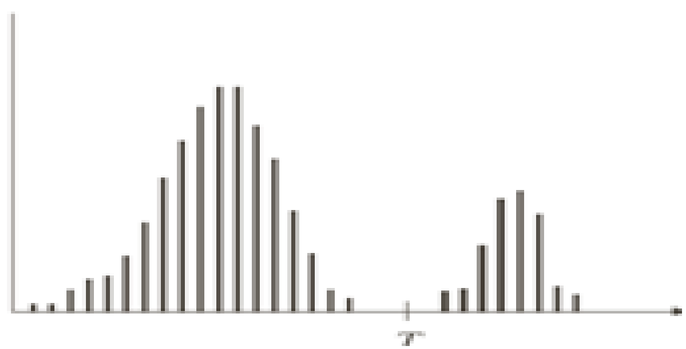
$$(x - c_1)^2 + (y - c_2)^2 = c_3^2$$

- Parameters will be(c1,c2,c3)-3D parameter space-cube like accumulator
- Complexity of Hough transform is depends on number of coefficients and coordinates in given functional representation

Thresholding

- Image partitioning into regions directly from their intensity values.
- Consider an image corresponds to dark background and light foreground

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



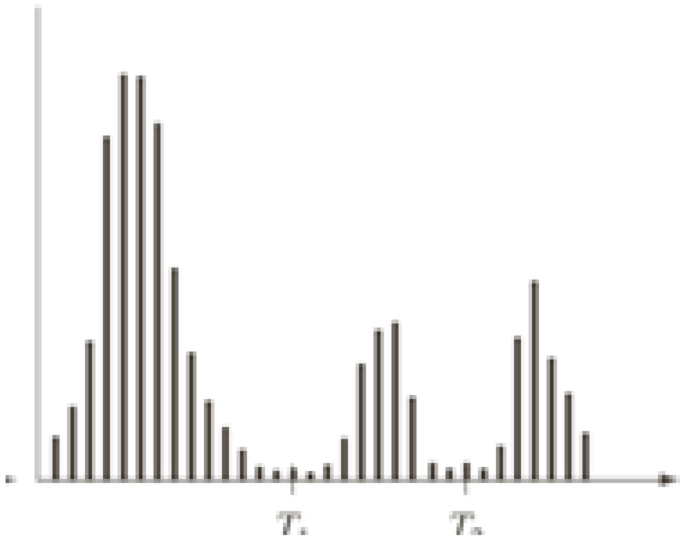
- Global thresholding- The value T is applicable over entire image.
- Variable thresholding - T changes over an image
- Local or regional thresholding -T depends on properties of neighborhood
- Dynamic or Adaptive thresholding- T depends on spatial coordinates themselves.

- The success of intensity thresholding is directly related to the width and depth of the valleys separating the histogram modes.
- Key factor affecting properties of the valleys are

1. Separation between peaks
2. The noise content in the image
3. The relative size of the background and objects
4. The uniformity of the illumination source
5. The uniformity of the reflectance properties of the image.

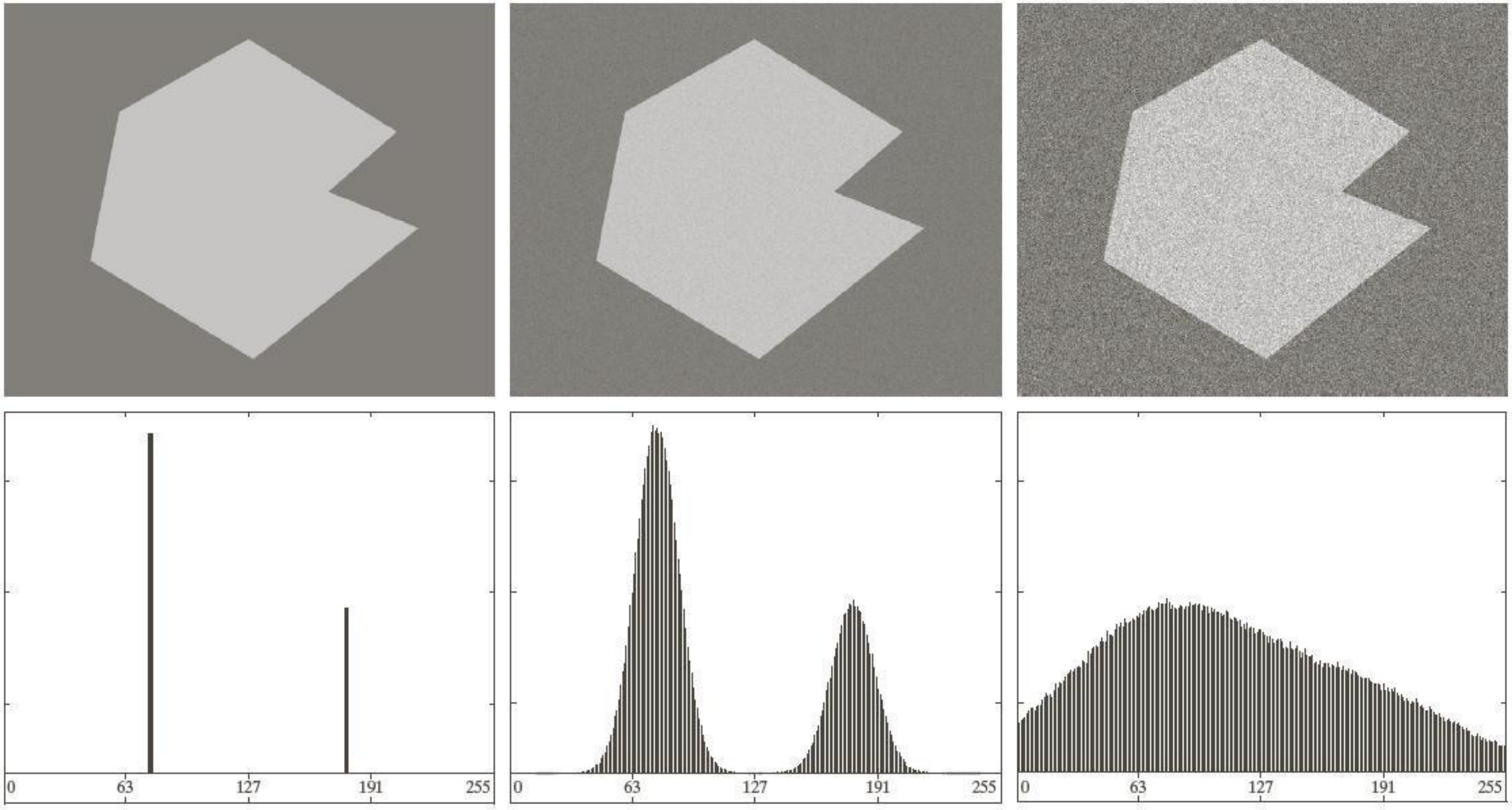
Two light objects on dark background

$$g(x,y)=\begin{cases}0 & \text{if } f(x,y)\le T_1\\ 1 & \text{if } T_1 < f(x,y)\le T_2\\ 2 & \text{if } f(x,y) > T_2\end{cases}$$



Noise in Thresholding

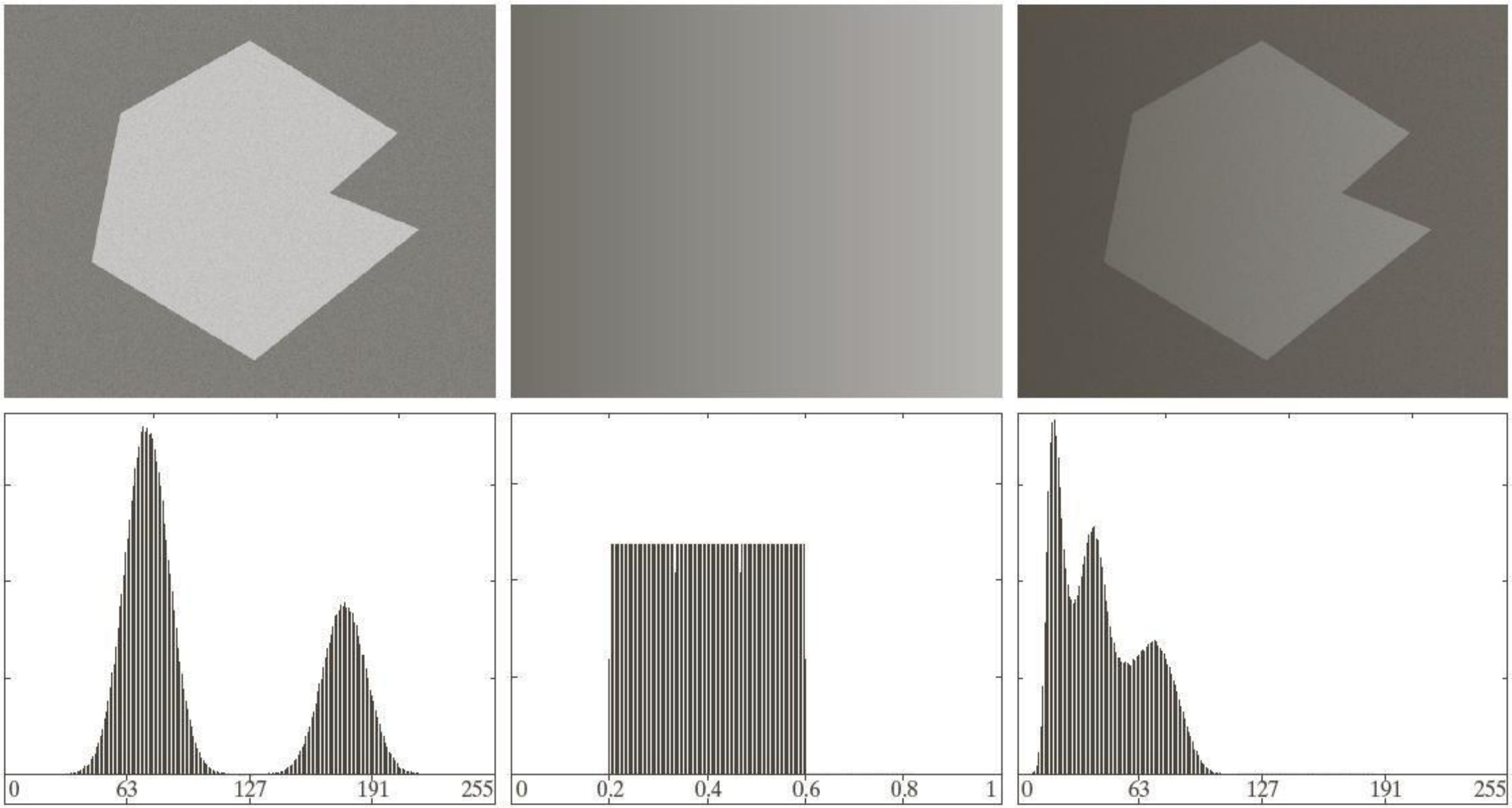
Difficulty in determining the threshold due to noise



Noiseless Gaussian (μ=0, σ=10) Gaussian (μ=0, σ=50)

Illumination in Thresholding

Difficulty in determining the threshold due to non-uniform illumination



(a) Noisy image

(b) Intensity ramp image

Multiplication

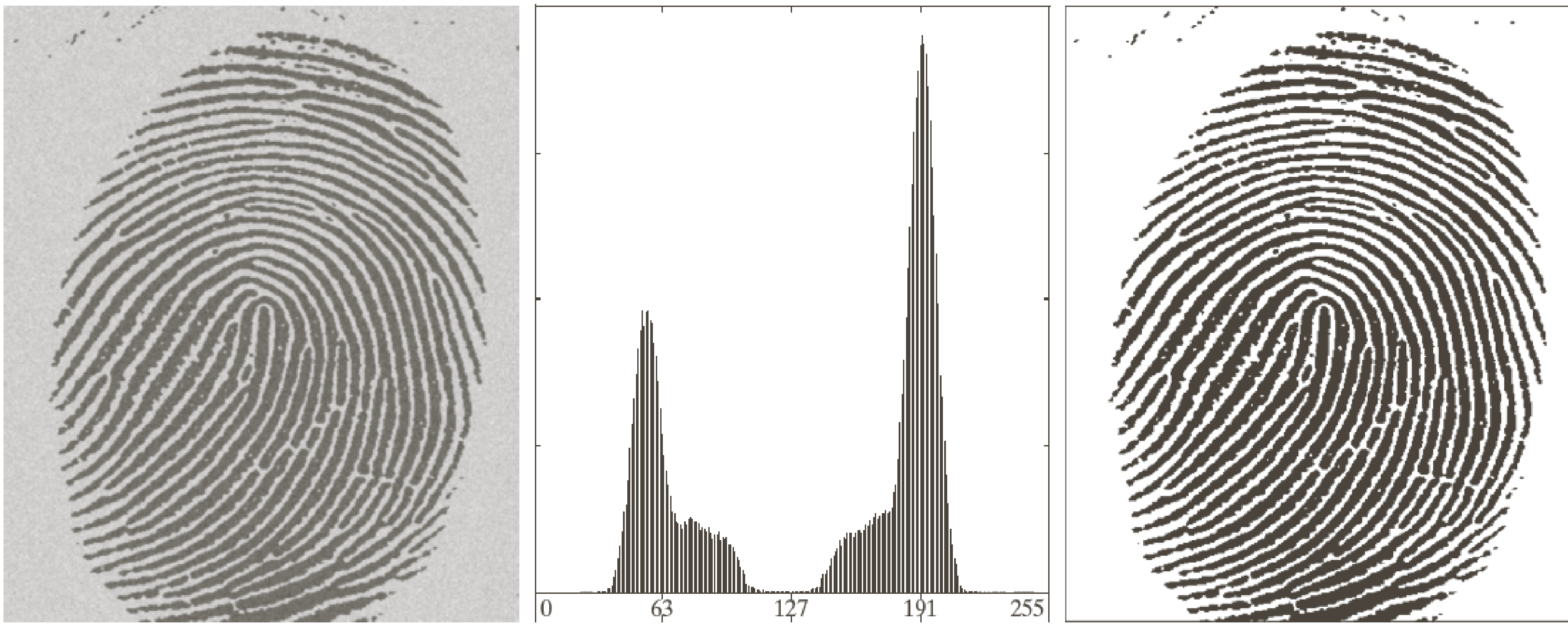
Basic Global Thresholding

- **Algorithm**
  - Select initial threshold estimate T.
  - Segment the image using T
    - Region G_1 (values > T) and region G_2 (values < T).
  - Compute the average intensities m_1 and m_2 of regions G_1 and G_2 respectively.
  - Set $\mathrm{T}=(m_1+m_2)/2$
  - Repeat until the change of T in successive iterations is less than ΔT.

Global thresholding:

- The simple algorithm works well where there is a reasonably clear valley in histogram
- ΔT controls the speed of operation➔ no of iteration
- Larger the value ➔ less iterations
- Smaller value➔ more iterations
- Initial threshold should be greater than minimum value of image intensity and lesser than maximum value of image intensity
- Avg intensity can be a good choice for initial threshold

Basic Global Thresholding (cont.)



T=125

Otsu’s method:

- Basic idea is to obtain well separated classes.
- Threshold should be optimum
- It should maximize between the class variance.

Otsu’s Method :

- Intensity levels: {0,1,2,…,L-1}
- Size of the image: MxN
- n_i : the number of pixels with intensity i
- $MN=n_0+n_1+n_2+\ldots+n_{L-1}$
- The normalized histogram $p_i=n_i/MN$

$$\sum_{i=0}^{L-1} p_i = 1, \qquad p_i \geq 0$$

- A Threshold $T(k)=k$, $0<k<L-1$ is selected to threshold the input image into two classes C1 and C2.
- C1 ➔ all the pixels with intensity[0 ,k]
- C2 ➔ all the intensities [k+1,L-1]
- P1(k) and P2(k) are the probability that pixel is assigned to class C1 and C2 respectively or its probability of occurrence of class C1 and C2

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

the mean intensity value of the pixels assigned to class C_1 is

$$
\begin{aligned}
m_1(k) &= \sum_{i=0}^{k} iP(i/C_1) \\
&= \sum_{i=0}^{k} iP(C_1/i)P(i)/P(C_1) \\
&= \frac{1}{P_1(k)} \sum_{i=0}^{k} ip_i
\end{aligned}
$$

Bayes' formula:

$$P(A/B) = P(B/A)P(A)/P(B)$$

$P(C_1/i)$, the probability of C_1 given i, is 1 because we are dealing only with values of i from class C_1.

Similarly, the mean intensity value of the pixels assigned to class C_2 is

$$
\begin{aligned}
m_2(k) &= \sum_{i=k+1}^{L-1} iP(i/C_2) \\
&= \frac{1}{P_2(k)} \sum_{i=k+1}^{L-1} ip_i
\end{aligned}
$$

The cumulative mean (average intensity) up to level k is given by

$$m(k) = \sum_{i=0}^{k} ip_i$$

the average intensity of the entire image (i.e., the *global* mean) is given by

$$m_G = \sum_{i=0}^{L-1} ip_i$$

$$P_1 + P_2 = 1$$

we have omitted the ks temporarily in favor of notational clarity.

$$P_1 m_1 + P_2 m_2 = m_G$$

$$= P_1(k) \times \frac{1}{P_1(k)} \sum_{i=0}^{k} i p_i + P_2(k) \times \frac{1}{P_2(k)} \sum_{i=k+1}^{L-1} i p_i$$

$$= \sum_{i=0}^{L-1} i p_i$$

In order to evaluate the "goodness" of the threshold at level k we use the normalized, dimensionless metric

$$\eta = \frac{\sigma_B^2}{\sigma_G^2}$$

where σ_G^2 is the *global variance*

$$\sigma_G^2 = \sum_{i=0}^{L-1} (i - m_G)^2 p_i$$

and σ_B^2 is the *between-class variance*, defined as

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

This expression can be written also as

$$\sigma_B^2 = P_1 P_2 (m_1 - m_2)^2$$
$$= \frac{(m_G P_1 - m)^2}{P_1(1 - P_1)}$$

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

$$\sigma_B{}^2 = P_1(m_1{}^2 - 2m_1m_G + m_G{}^2) + P_2(m_2{}^2 - 2m_2m_G + m_G{}^2)$$

$$\sigma_B{}^2 = P_1m_1{}^2 - 2(P_1m_1 + P_2m_2)m_G + (P_1 + P_2)m_G{}^2 + P_2m_2{}^2$$

$$\sigma_B{}^2 = P_1m_1{}^2 - 2m_G{}^2 + m_G{}^2 + P_2m_2{}^2$$

$$\sigma_B{}^2 = P_1m_1{}^2 - m_G{}^2 + P_2m_2{}^2$$

$P_1m_1 + P_2m_2 = m_G$ $\qquad$ $P_1 + P_2 = 1$

$$\sigma_B{}^2 = P_1m_1{}^2 - (P_1{}^2m_1{}^2 + 2P_1P_2m_1m_2 + P_2{}^2m_2{}^2) + P_2m_2{}^2$$

$$\sigma_B{}^2 = P_1(1 - P_1)m_1{}^2 - 2P_1P_2m_1m_2 + P_2(1 - P_2)m_2{}^2$$

$$\sigma_B{}^2 = P_1P_2m_1{}^2 - 2P_1P_2m_1m_2 + P_1P_2m_2{}^2$$

$$\sigma_B{}^2 = P_1P_2(m_1 - m_2)^2$$

$$\sigma_B^2 = P_1P_2(m_1 - m_2)^2$$

$$= P_1P_2\left(\frac{1}{P_1}\sum_{i=0}^{k} ip_i - \frac{1}{P_2}\sum_{i=k+1}^{L-1} ip_i\right)^2$$

$$m = \sum_{i=0}^{k} ip_i \qquad m_G = \sum_{i=0}^{L-1} ip_i$$

$$= P_1P_2\left(\frac{1}{P_1}m - \frac{1}{P_2}(m_G - m)\right)^2$$

$$= \frac{P_1P_2}{P_1^2P_2^2}\left(P_2m - P_1(m_G - m)\right)^2$$

$$= \frac{1}{P_1P_2}\left((1 - P_1)m - P_1(m_G - m)\right)^2 = \frac{(m_GP_1 - m)^2}{P_1(1 - P_1)}$$

- σ_B^2 and η are measures of separability between classes.
- Larger σ_B^2 ➔ larger between the class variance ➔ more separability ➔ higher η
- Generally $\sigma_G^2 > 0$
- If $\sigma_G^2 = 0$ ➔ constant intensity in image➔ only one class possible➔ η =0➔ indicating no separation between the class.

Reintroducing k, we have the final results:

$$\eta(k) = \frac{\sigma_B^2(k)}{\sigma_G^2}$$

and

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

Ultimate goal is to find k for which $\sigma_B{}^2$ is maximum.

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

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

In other words, to find k^* we simply evaluate Eq. for all *integer* values of k select that value of k that yielded the maximum $\sigma_B^2(k)$.

If the maximum exists for more than one value of k, it is customary to average the various values of k for which $\sigma_B^2(k)$ is maximum.

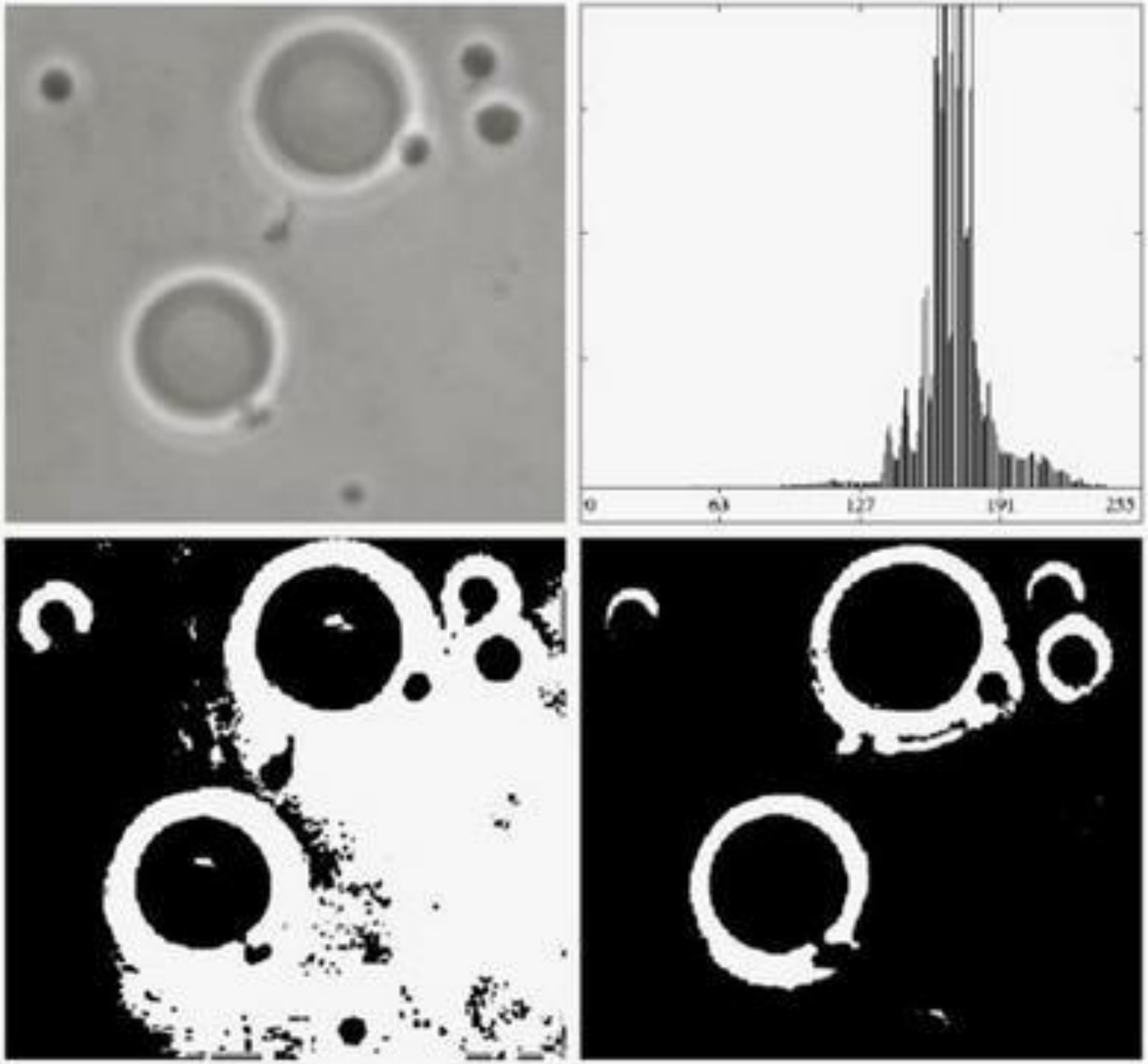
Once k^* has been obtained, the input image $f(x, y)$ is segmented as before:

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

for $x = 0, 1, 2, \ldots, M - 1$ and $y = 0, 1, 2, \ldots, N - 1$.

The normalized metric η, evaluated at the optimum threshold value, $\eta(k^*)$, can be used to obtain a quantitative estimate of the separability of classes, Otsu's algorithm may be summarized as follows:

1. Compute the normalized histogram of the input image. Denote the components of the histogram by p_i, $i = 0, 1, 2, \ldots, L - 1$.
2. Compute the cumulative sums, $P_1(k)$, for $k = 0, 1, 2, \ldots, L - 1$.
3. Compute the cumulative means, $m(k)$, for $k = 0, 1, 2, \ldots, L - 1$
4. Compute the global intensity mean, m_G
5. Compute the between-class variance, $\sigma_B^2(k)$, for $k = 0, 1, 2, \ldots, L - 1$
6. Obtain the Otsu threshold, k^*, as the value of k for which $\sigma_B^2(k)$ is maximum. If the maximum is not unique, obtain k^* by averaging the values of k corresponding to the various maxima detected.
7. Obtain the separability measure, η^*, at $k = k^*$.



a b
c d

FIGURE 10.39 (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 10.3.2. (d) Result obtained using Otsu's method. (Original image courtesy of Professor Daniel A. Hammer, the University of Pennsylvania.)

The threshold value computed by the basic algorithm was 169, while the threshold computed by Otsu's method was 181, which is closer to the lighter areas in the image defining the cells. The separability measure η was 0.467.

As a point of interest, applying Otsu's method to the fingerprint image in yielded a threshold of 125 and a separability measure of 0.944.

The separability measure is high due primarily to the relatively large separation between modes and the deep valley between them.

Multiple thresholds:

- In the case of K classes C1, C2,…CK between the class variance is defined by :

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

where

$$P_k = \sum_{i \in C_k} p_i$$

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

and m_G is the global mean. The K classes are separated by $K-1$ thresholds whose values, $k_1^*, k_2^*, \ldots, k_{K-1}^*$, are the values that maximize

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

For three classes consisting of three intensity intervals (which are separated by two thresholds) the between-class variance is given by:

$$\sigma_B^2 = P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2 + P_3(m_3 - m_G)^2$$

where

$$P_1 = \sum_{i=0}^{k_1} p_i$$

$$P_2 = \sum_{i=k_1+1}^{k_2} p_i$$

$$P_3 = \sum_{i=k_2+1}^{L-1} p_i$$

$$m_1 = \frac{1}{P_1}\sum_{i=0}^{k_1} i p_i$$

$$m_2 = \frac{1}{P_2}\sum_{i=k_1+1}^{k_2} i p_i$$

$$m_3 = \frac{1}{P_3}\sum_{i=k_2+1}^{L-1} i p_i$$

$$P_1 m_1 + P_2 m_2 + P_3 m_3 = m_G$$

and

$$P_1 + P_2 + P_3 = 1$$

$$\sigma_B^2(k_1^*, k_2^*) = \max_{0<k_1<k_2<L-1} \sigma_B^2(k_1, k_2)$$

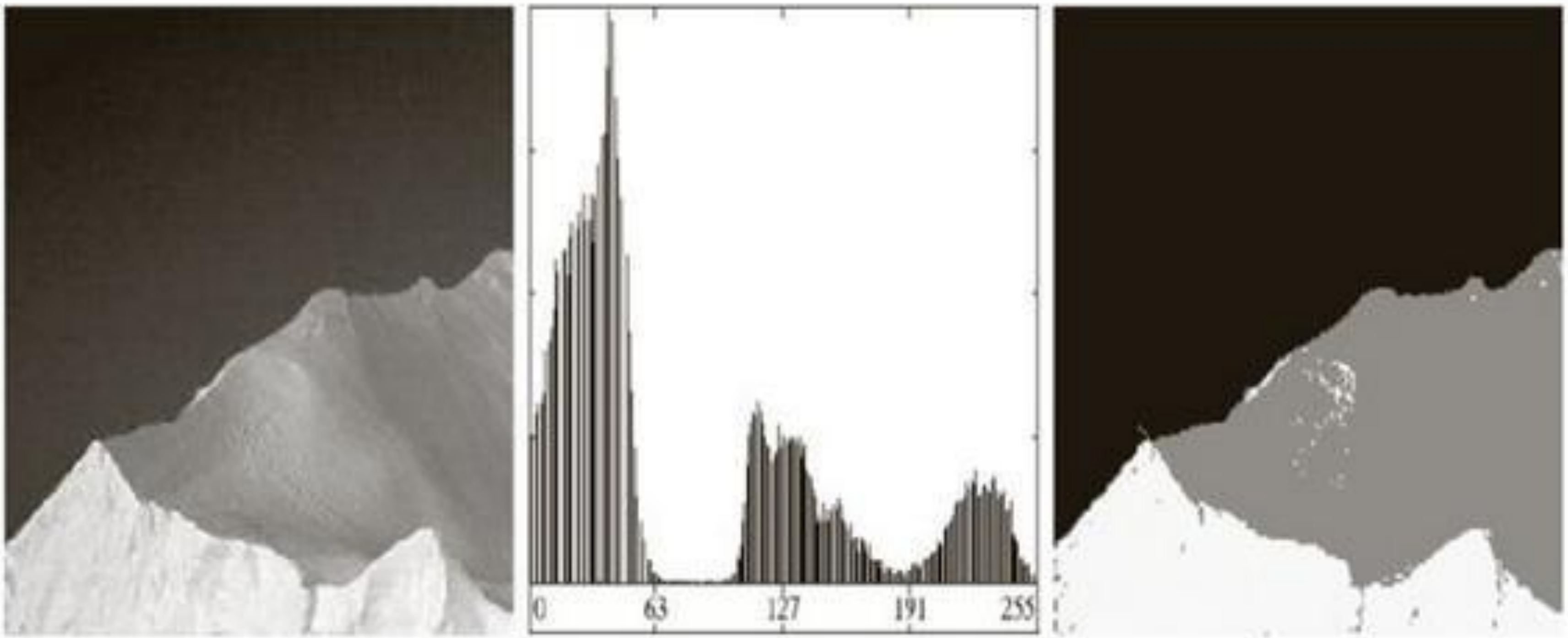
- Select the value of k1
- Select the value of k2 is incremented through all possible values which are greater than k1 but less than L-1
- Compute the between the class variance for each pair of k1 and k2
- Increment the value of k1 repeat the procedure.
- Select maximum value of between the class value and corresponding k1 and k2 are required thresholds.
- In case of several maxima, corresponding values of k1 and k2 are averaged to obtain final results.

$$g(x, y) = \begin{cases} a & \text{if } f(x, y) \leq k_1^* \\ b & \text{if } k_1^* < f(x, y) \leq k_2^* \\ c & \text{if } f(x, y) > k_2^* \end{cases}$$

where a, b, and c are any three valid intensity values.

$$\eta(k_1^*, k_2^*) = \frac{\sigma_B^2(k_1^*, k_2^*)}{\sigma_G^2}$$

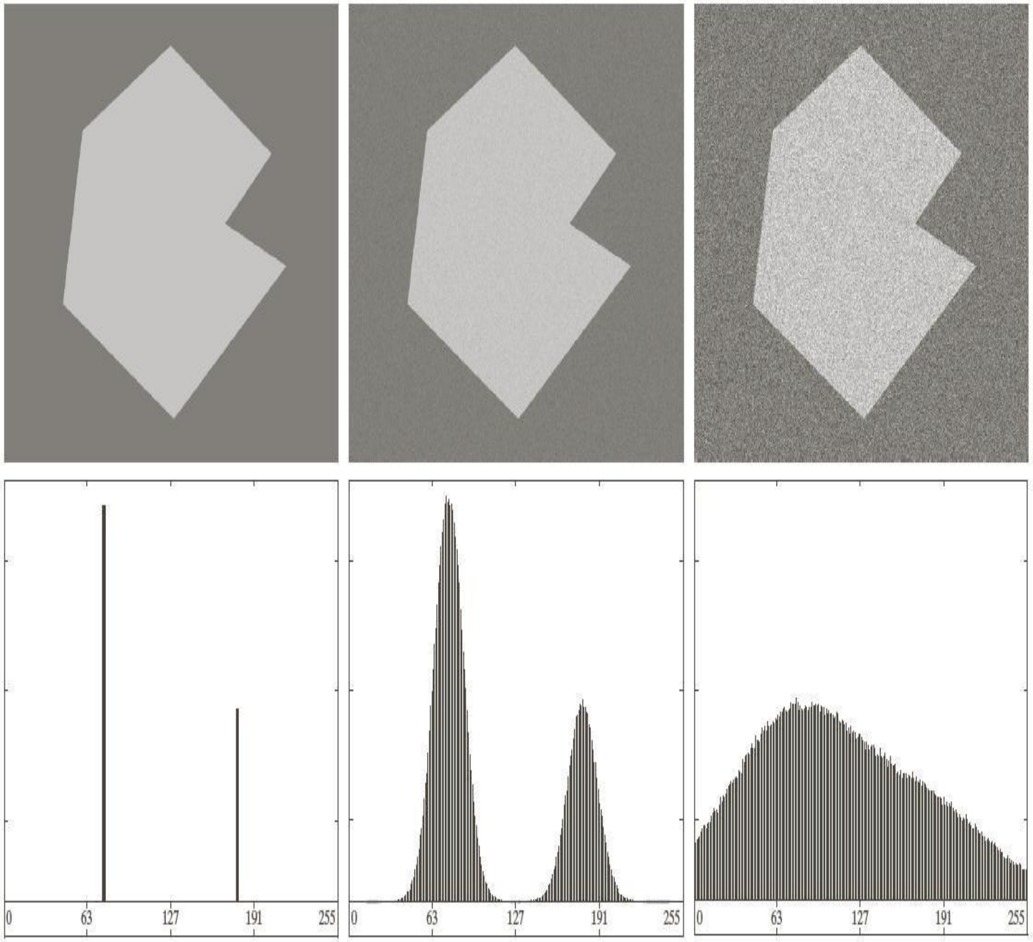
where σ_G^2 is the total image variance



a b c

FIGURE 10.45 (a) Image of iceberg. (b) Histogram. (c) Image segmented into three regions using dual Otsu thresholds. (Original image courtesy of NOAA.)

Smoothing to improve global thresholding:



- Noise creates unsolvable problem in thresholding.
- If noise can not be removed at source and thresholding is to be done then smoothing can enhance the performance.

Smoothing to improve global thresholding:

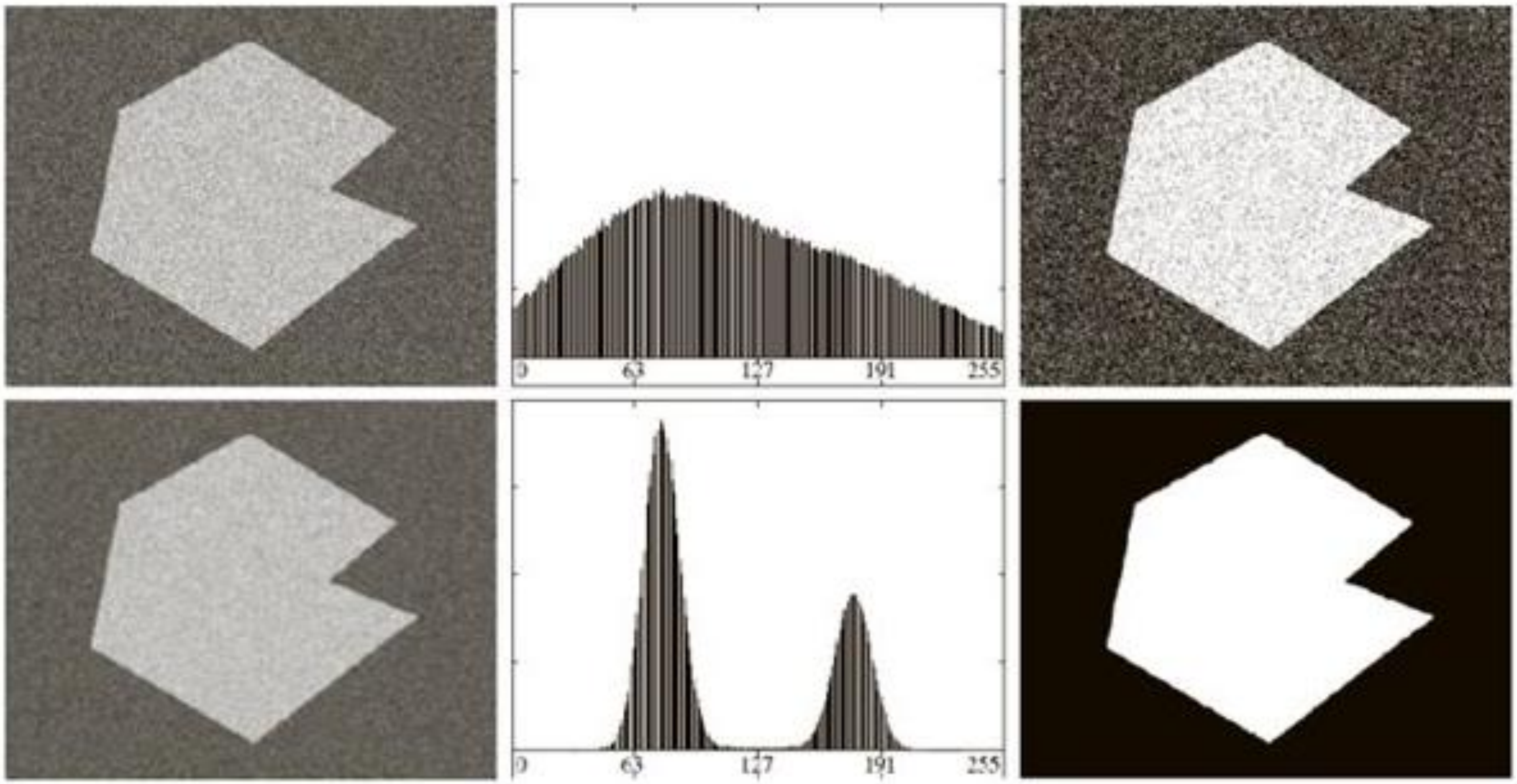
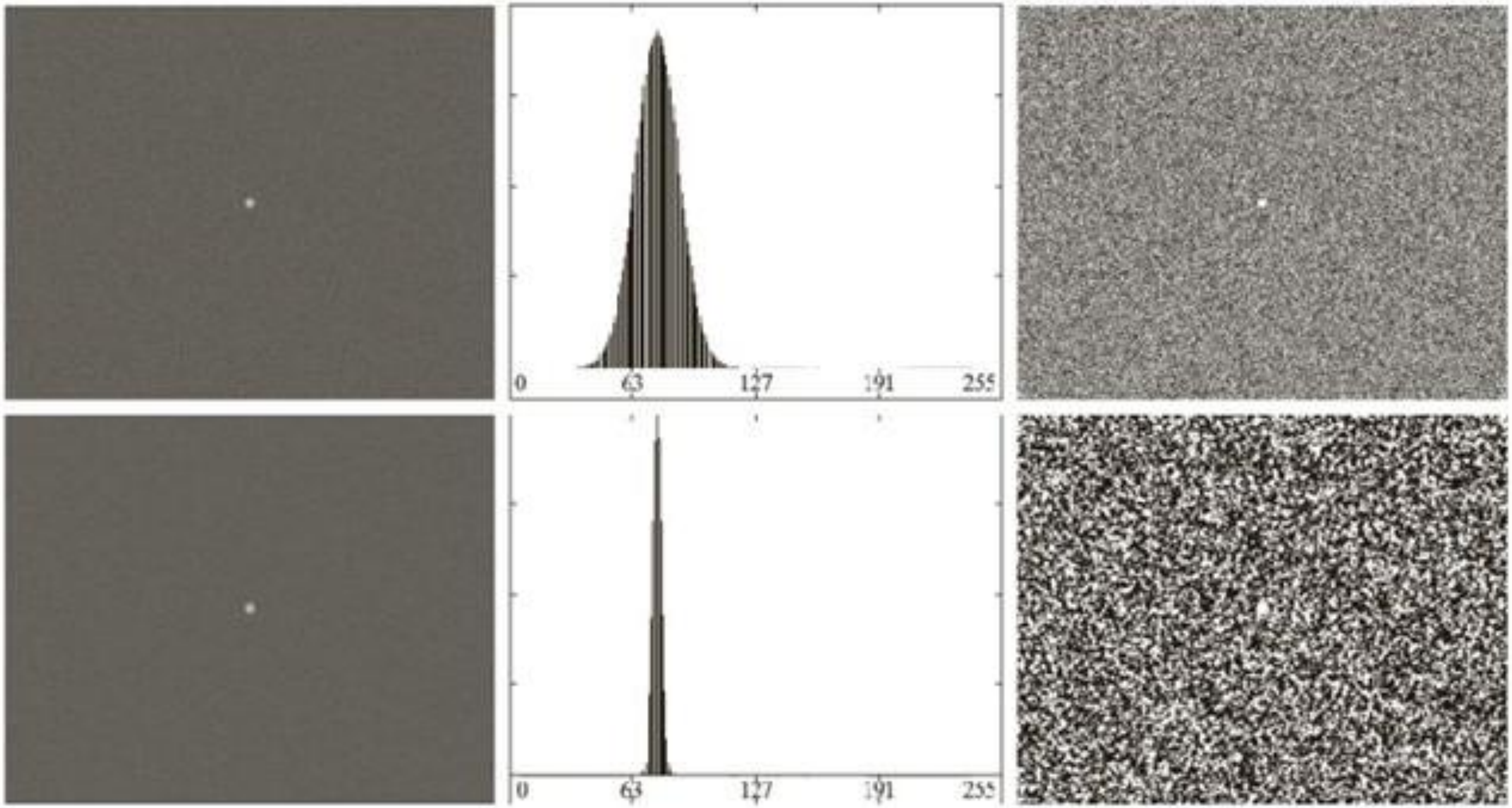


FIGURE 10.40 (a) Noisy image from Fig. 10.36 and (b) its histogram. (c) Result obtained using Otsu's method. (d) Noisy image smoothed using a 5 × 5 averaging mask and (e) its histogram. (f) Result of thresholding using Otsu's method.

Relative size:

- Reducing relative size of foreground w.r.t background➔ no clear valley in histogram.
- Smoothing reduces the spread as expected but distribution does not change➔ not effective.



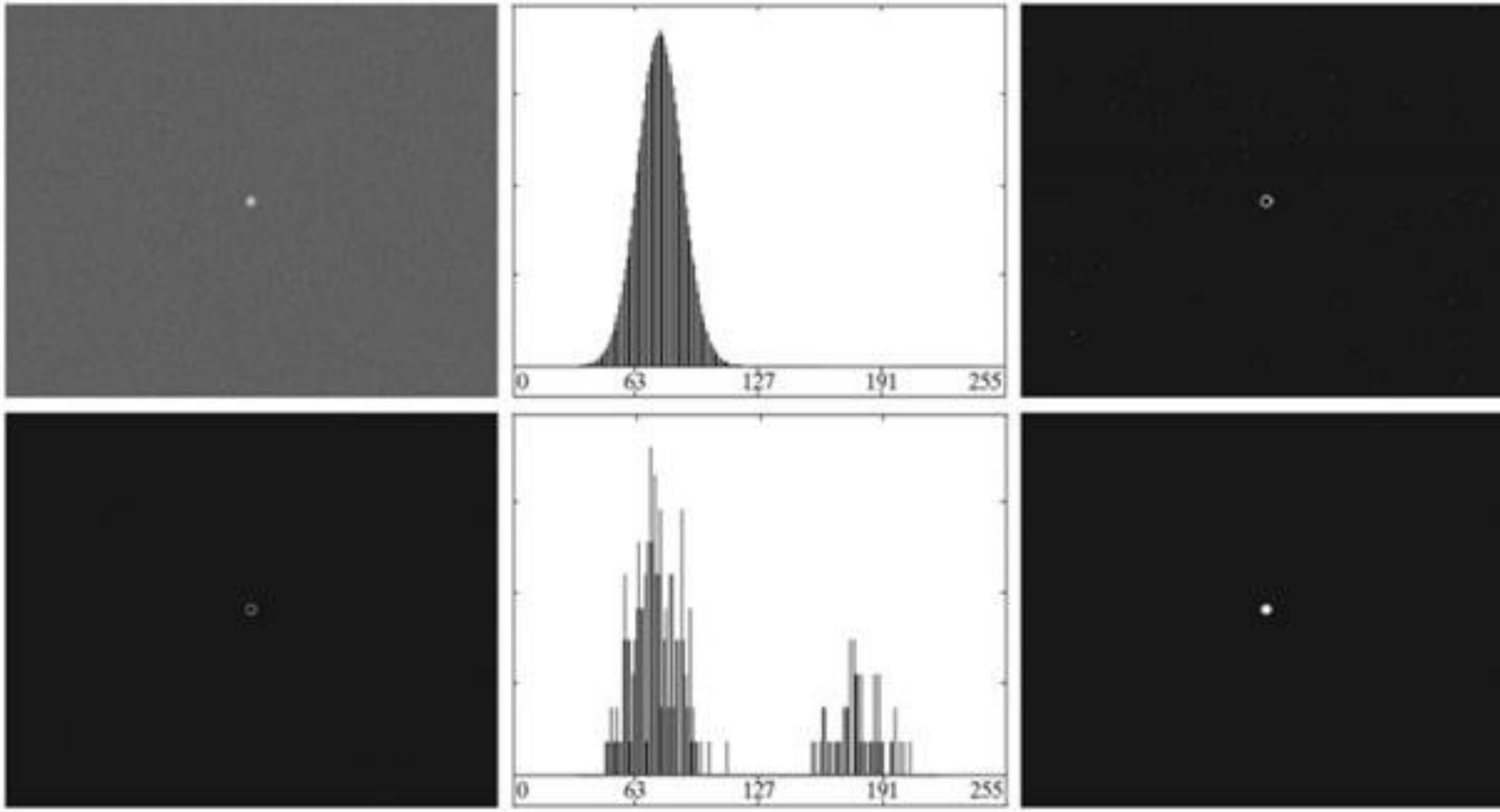
a b c
d e f

FIGURE 10.41 (a) Noisy image and (b) its histogram. (c) Result obtained using Otsu's method. (d) Noisy image smoothed using a 5×5 averaging mask and (e) its histogram. (f) Result of thresholding using Otsu's method. Thresholding failed in both cases.

Using edges to improve global thresholding:

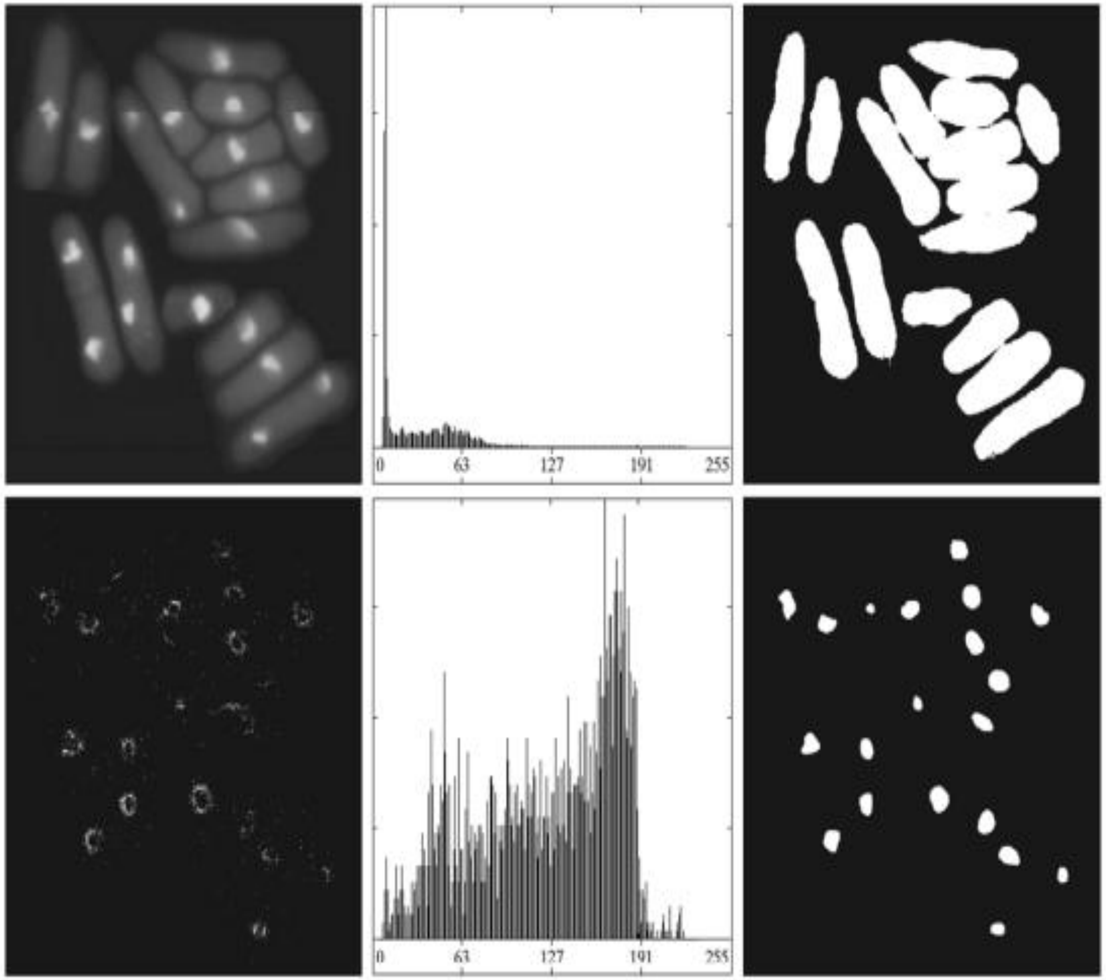
- Consider only those pixels which lies on or near to edges between foreground and background.
- Now histogram will be less dependent on sizes of foreground and background.
- Resulting histogram will have peaks of approximately same height
- The pixels that satisfy some measures based on gradient and laplacian has tendency to deepen the valleys in histogram.
- Laplacian is preferred as its computationally more attractive and isotropic.

1. Compute an edge image as either the magnitude of the gradient, or absolute value of the Laplacian, of $f(x, y)$ using any of the methods discussed in Section 10.2.
2. Specify a threshold value, T.
3. Threshold the image from Step 1 using the threshold from Step 2 to produce a binary image, $g_T(x, y)$. This image is used as a *mask image* in the following step to select pixels from $f(x, y)$ corresponding to "strong" edge pixels.
4. Compute a histogram using only the pixels in $f(x, y)$ that correspond to the locations of the 1-valued pixels in $g_T(x, y)$.
5. Use the histogram from Step 4 to segment $f(x, y)$ globally using, for example, Otsu's method.



a b c
d e f

FIGURE 10.42 (a) Noisy image from Fig. 10.41(a) and (b) its histogram. (c) 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.



99.5%, Th=115, η=0.762

a b c
d e f

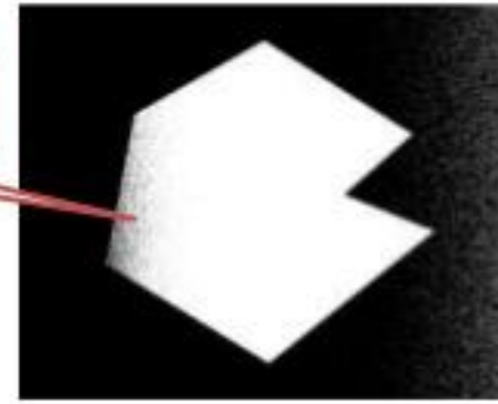
FIGURE 10.43 (a) Image of yeast cells. (b) Histogram of (a). (c) Segmentation of (a) with Otsu's method using the histogram in (b). (d) Thresholded absolute Laplacian. (e) Histogram of the nonzero pixels in the product of (a) and (d). (f) Original image thresholded using Otsu's method based on the histogram in (e). (Original image courtesy of Professor Susan L. Forsburg, University of Southern California.)

FIGURE 10.44 Image in Fig. 10.43(a) segmented using the same procedure as explained in Figs. 10.43(d)–(f), but using a lower value to threshold the absolute Laplacian image.

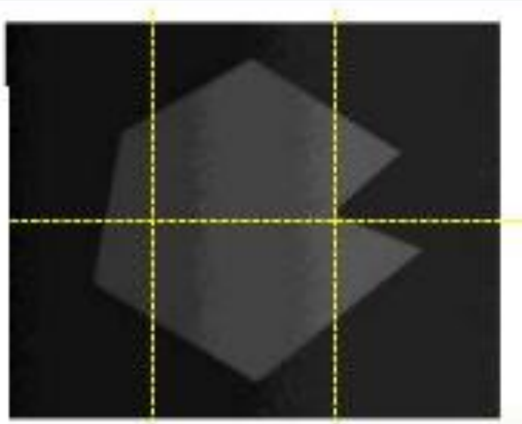
95%

Variable Thresholdig:

- Image partitioning: Subdivide the image in to non overlapping rectangles.
- This approach is used to compensate the non-uniformities in illumination/reflectance
- The rectangles are chosen small enough so that the illumination of each is approximately uniform.



Noisy shaded image	Histogram	Otsu Thresholding
Image sub divide into six sub images		Result of applying Otsu method to each sub image individually



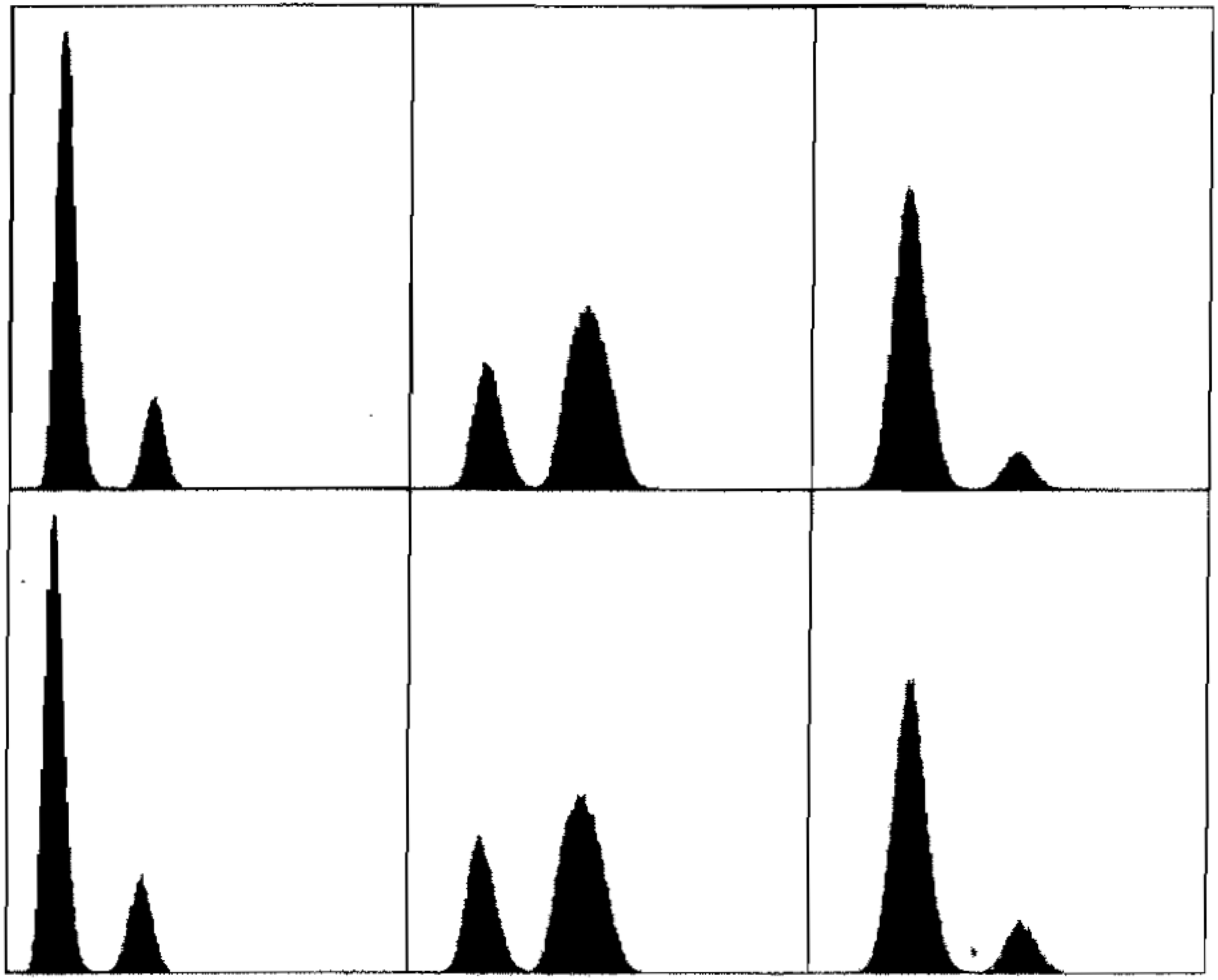


Image partitioning:

- Image subdivision generally works well when objects of interest and background occupy regions of reasonably comparable size
- When this is not the case method typically fails because of likelihood of subdivision containing only objects or only background pixels.

Variable thresholding

Based on local image properties:

- Use of standard deviation and mean of local neighbourhood on every point

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

following are common forms of variable, local thresholds:

$$T_{xy} = a\sigma_{xy} + bm_{xy}$$

where a and b are nonnegative constants, and

$$T_{xy} = a\sigma_{xy} + bm_G$$

where m_G is the global image mean.

The segmented image is computed as

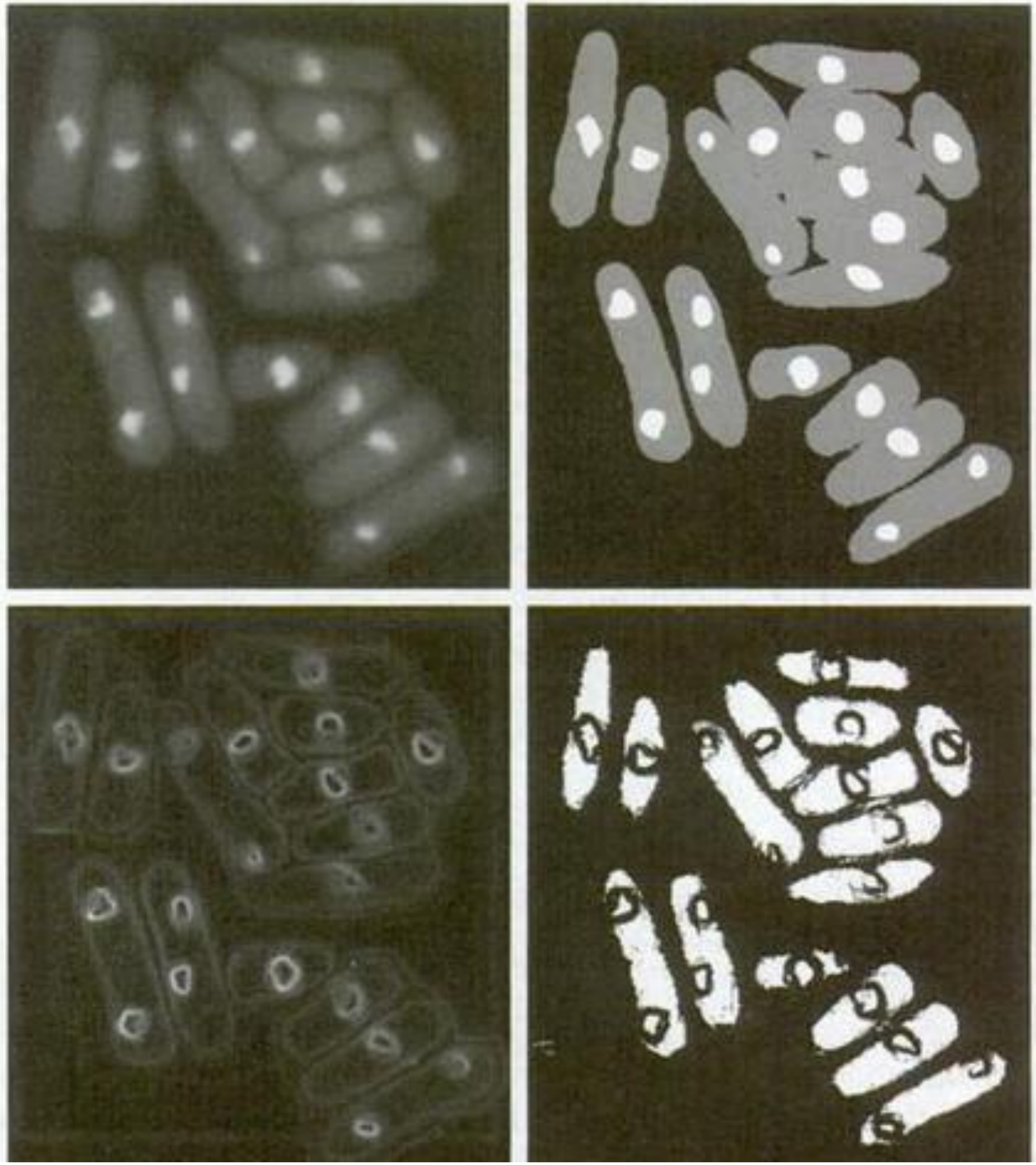
$$g(x, y) = \begin{cases} 1 & \text{if } f(x, y) > T_{xy} \\ 0 & \text{if } f(x, y) \le T_{xy} \end{cases}$$

where $f(x, y)$ is the input image. This equation is evaluated for all pixel locations in the image, and a different threshold is computed at each location (x, y) using the pixels in the neighborhood S_{xy}.

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

where Q is a *predicate* based on parameters computed using the pixels in neighborhood S_{xy}. For example, consider the following predicate, $Q(\sigma_{xy}, m_{xy})$, based on the local mean and standard deviation:

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



a b
c d

FIGURE 10.48
(a) Image from Fig. 10.43.
(b) Image segmented using the dual thresholding approach discussed in Section 10.3.6.
(c) Image of local standard deviations.
(d) Result obtained using local thresholding.

Moving average:

- A special case of local thresholding method.
- Method is based on computing a moving average along scan lines of an image.
- Scanning is carried out line by line in a zigzag pattern to reduce illumination bias.
- Useful in applications like document processing where speed is the fundamental requirement.

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 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 $m(1) = z_1/n$.

$T_{xy} = bm_{xy}$,where b is constant and is the moving average at point (*x*,*y*)

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

$$m[k] = \frac{z[k]}{3} + \frac{z[k-1]}{3} + \frac{z[k-2]}{3}$$

$$m[k] = \frac{z[k]}{n} + \frac{z[k-1]}{n} + \frac{z[k-2]}{n} + \cdots \frac{z[k-(n-3)]}{n} + \frac{z[k-(n-2)]}{n} + \frac{z[k-(n-1)]}{n}$$

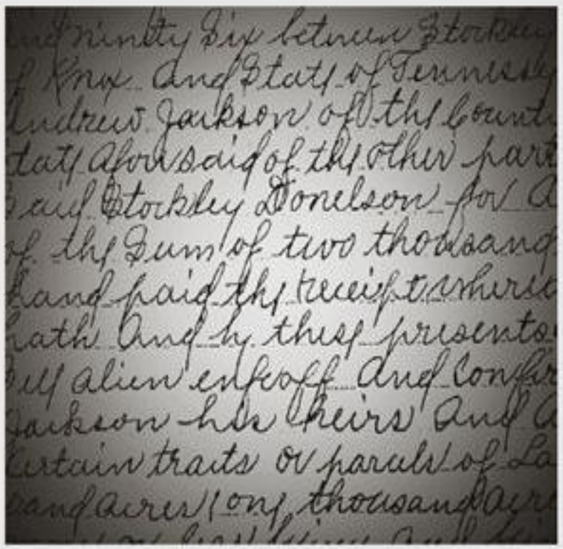
$$m[k+1] = \frac{z[k+1]}{n} + \frac{z[k+1-1]}{n} + \frac{z[k+1-2]}{n} + \cdots + \frac{z[k+1-(n-2)]}{n} + \frac{z[k+1-(n-1)]}{n}$$

$$m[k+1] = \frac{z[k+1]}{n} + m[k] - \frac{z[k-(n-1)]}{n}$$

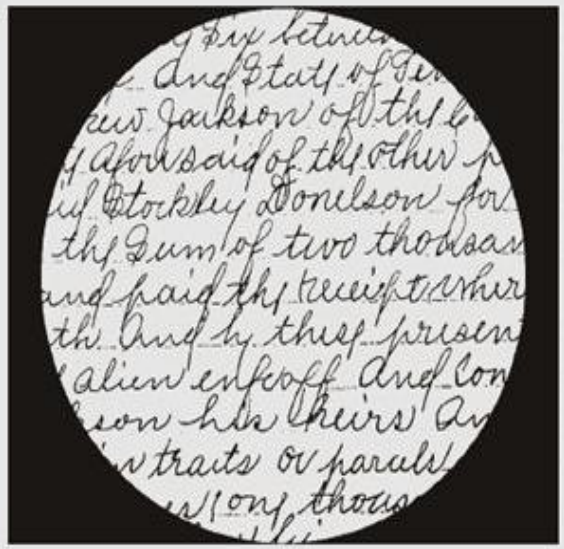
$$m[1] = \frac{z[1]}{n} + \frac{z[1-1]}{n} + \frac{z[1-2]}{n} + \cdots + \frac{z[1-(n-1)]}{n}$$

$$m[1] = \frac{z[1]}{n} + 0 + 0 + \cdots + 0$$

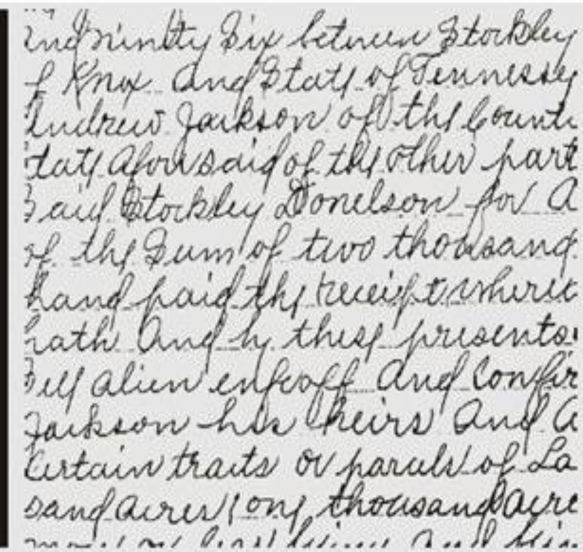
Thresholding (non-uniform background)



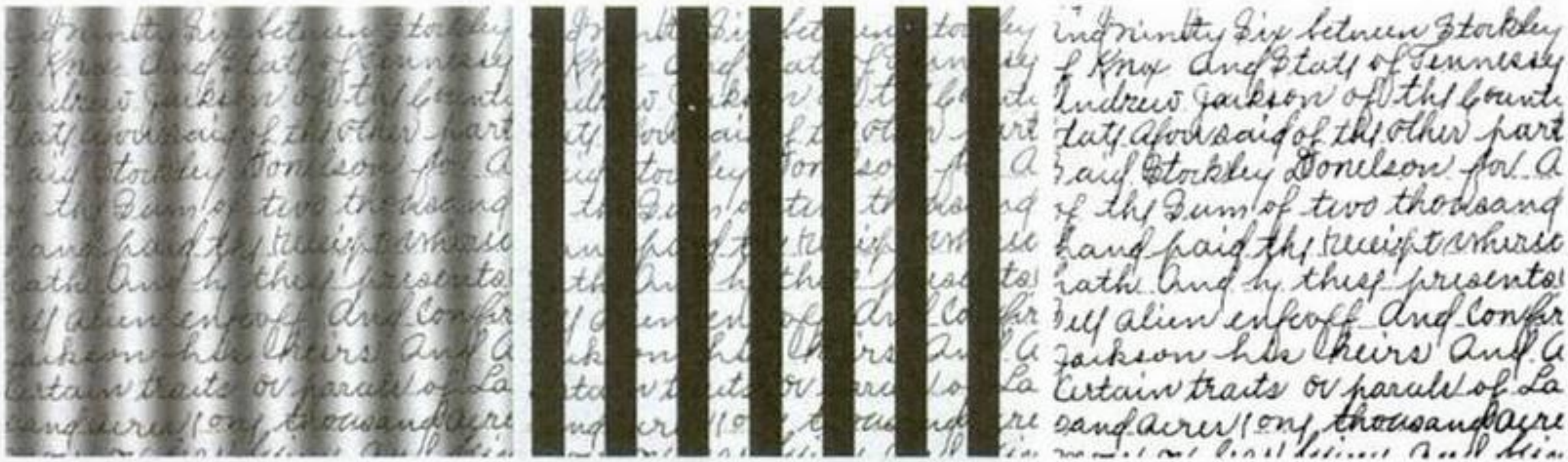
Input image



Global thresholding using Otsu's method



Local thresholding with moving average



a b c

FIGURE 10.50 (a) Text image corrupted by sinusoidal shading. (b) Result of global thresholding using Otsu's method. (c) Result of local thresholding using moving averages.

Region based segmentation:

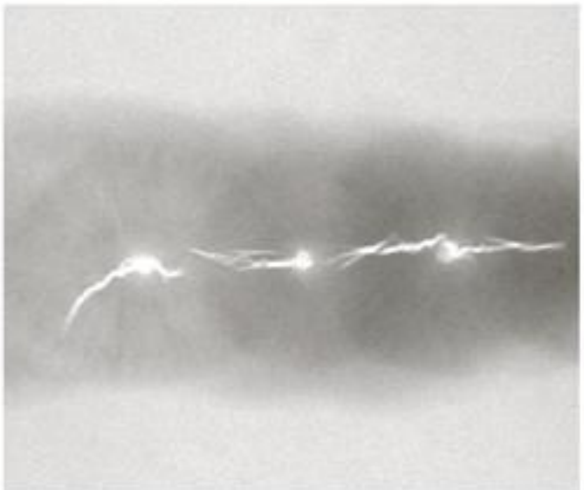
- Region growing: a procedure that groups pixels or subregions based on predefined criteria for growth.
- Procedure:
- Start with set of seeds.
- Grow the region by appending neighbouring pixels to seed based those have similarity with seed in predefined properties.

- Needs a stopping rule.

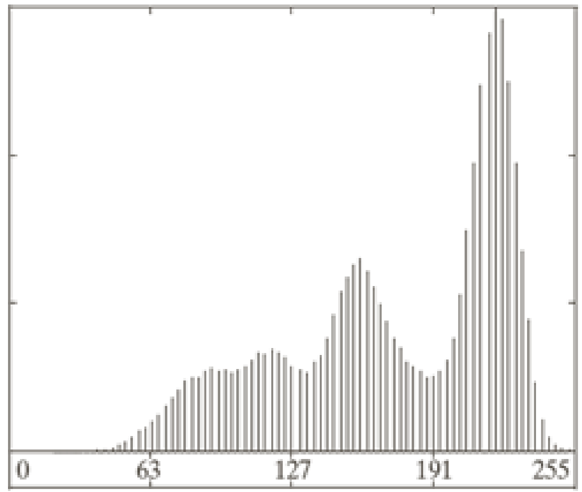
Let: $f(x, y)$ denote an input image array; $S(x, y)$ denote a *seed* array containing 1s at the locations of seed points and 0s elsewhere; and Q denote a predicate to be applied at each location (x, y). Arrays f and S are assumed to be of the same size. A basic region-growing algorithm based on 8-connectivity may be stated as follows.

1. Find all connected components in $S(x, y)$ and erode each connected component 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 the input image satisfies the given predicate, Q, at those coordinates; otherwise, let $f_Q(x, y) = 0$.
3. Let g be an image formed by appending to each seed point in S all the 1-valued points in f_Q that are 8-connected to that seed point.
4. Label each connected component in g with a different region label (e.g., $1, 2, 3, \ldots$). This is the segmented image obtained by region growing.

Region Growing



X-Ray image of weld with a crack we want to segment.



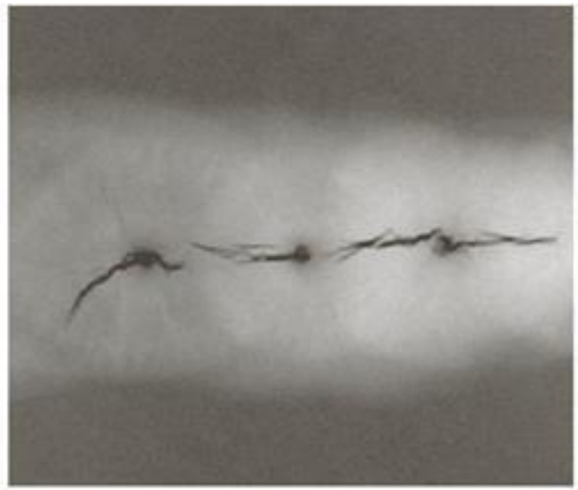
Histogram

Seed image (99.9% of max value in the initial image).
Crack pixels are missing.

The weld is very bright. The predicate used for region growing is to compare the absolute difference between a seed point and a pixel to a threshold. If the difference is below it we accept the pixel as crack

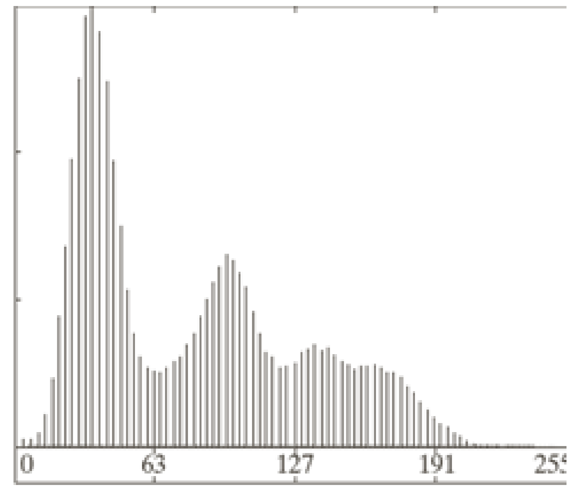
Region Growing

Seed image eroded to 1 pixel regions.

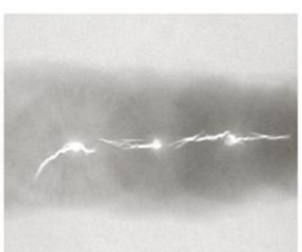


Difference between the original image and the initial seed image.

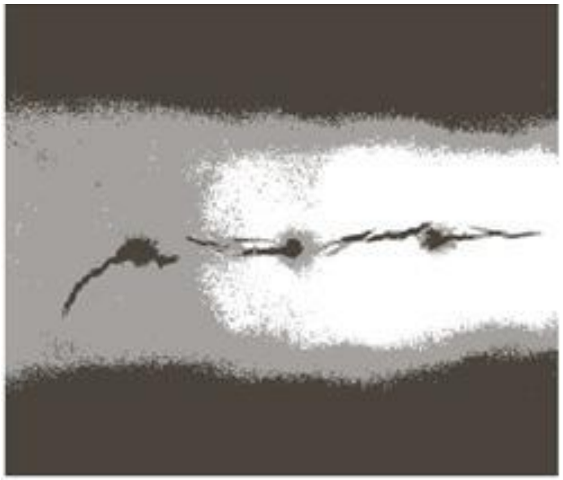
The pixels are ready to be compared to the threshold.



Histogram of the difference image. Two valleys at 68 and 126 provided by Otsu.



Region Growing



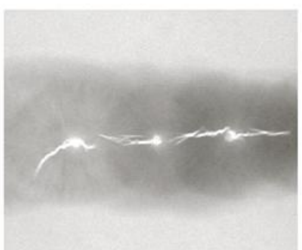
Otsu thresholding of the difference image to 3 regions (2 thresholds).

Thresholding of the difference image with the lowest of the dual thresholds.

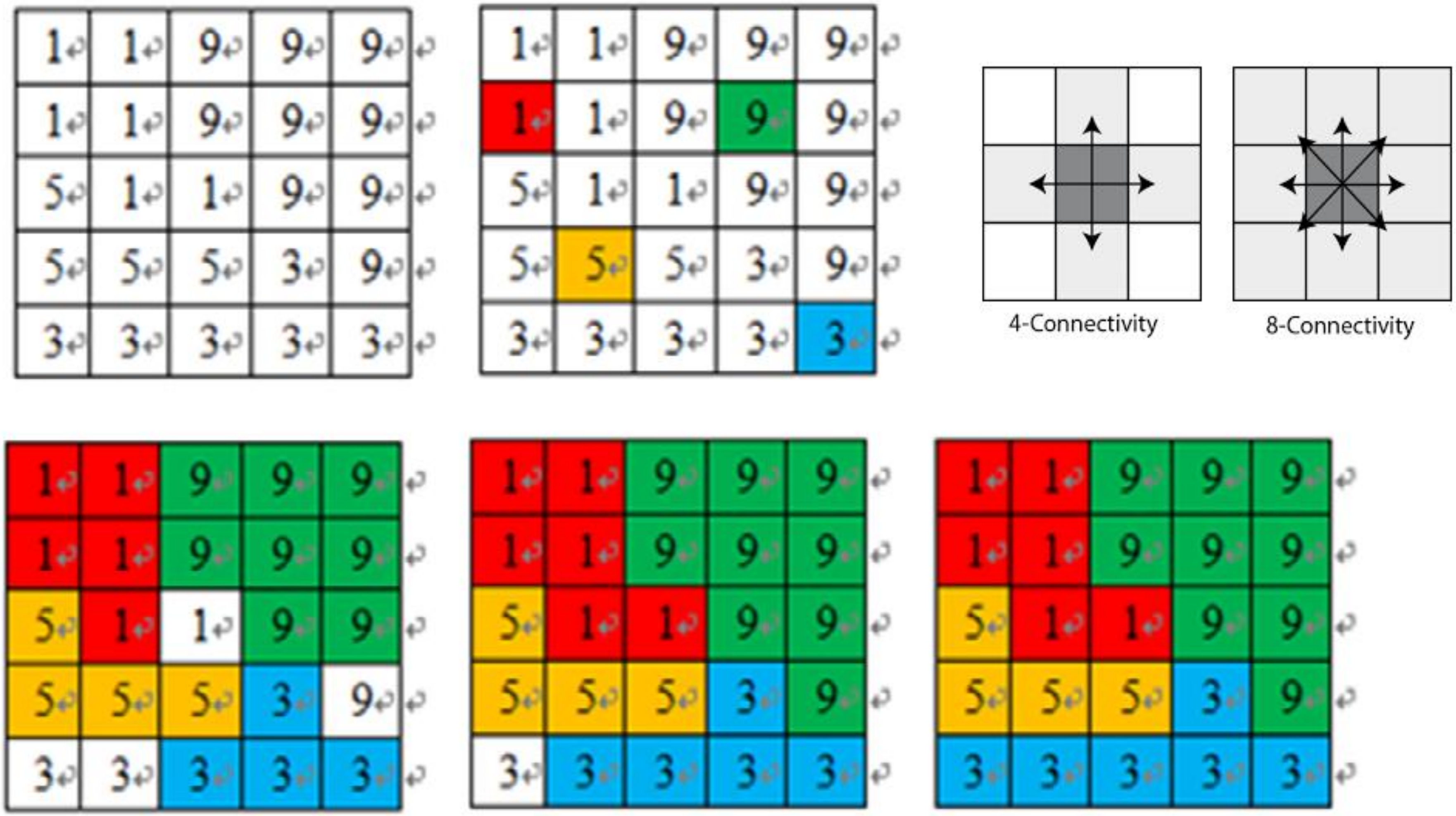
Notice that the background is also considered as crack.

Segmentation result by region growing.

The background is not considered as crack.
It is removed as it is not 8-connected to the seed pixels.



Q= difference between seed and pixel intensity should be zero.
8-connectivity



Q= difference between seed and pixel intensity should be zero.

4-connectivity

1	1	9	9	9
1	1	9	9	9
5	1	1	9	9
5	5	5	3	9
3	3	3	3	3

1	1	9	9	9
1	1	9	9	9
5	1	1	9	9
5	5	5	3	9
3	3	3	3	3

1	1	9	9	9
1	1	9	9	9
5	1	1	9	9
5	5	5	3	9
3	3	3	3	3

1	1	9	9	9
1	1	9	9	9
5	1	1	9	9
5	5	5	3	9
3	3	3	3	3

1	1	9	9	9
1	1	9	9	9
5	1	1	9	9
5	5	5	3	9
3	3	3	3	3

1	1	9	9	9
1	1	9	9	9
5	1	1	9	9
5	5	5	3	9
3	3	3	3	3

Region Splitting and Merging

- Based on quadtrees (*quadimages*).
- The root of the tree corresponds to the image.
- Split the image to sub-images that do not satisfy a predicate Q.
- If only splitting was used, the final partition would contain adjacent regions with identical properties.
- A merging step follows merging regions satisfying the predicate Q.

That is, two adjacent regions R_j and R_k are merged only if $Q(R_j \cup R_k) = \text{TRUE}$.

Region splitting and merging:

Let R represent the entire image region and select a predicate Q.

One approach for segmenting R is to subdivide it successively into smaller and smaller quadrant regions so that, for any region R_i, $Q(R_i) = \text{TRUE}$.

We start with the entire region. If $Q(R) = \text{FALSE}$, we divide the image into quadrants.

If Q is FALSE for any quadrant, we subdivide that quadrant into subquadrants, and so on.

This particular splitting technique has a convenient representation in the form of so-called *quadtrees*

the images corresponding to the nodes of a quadtree sometimes are called *quadregions* or *quadimages*

Region Splitting and Merging

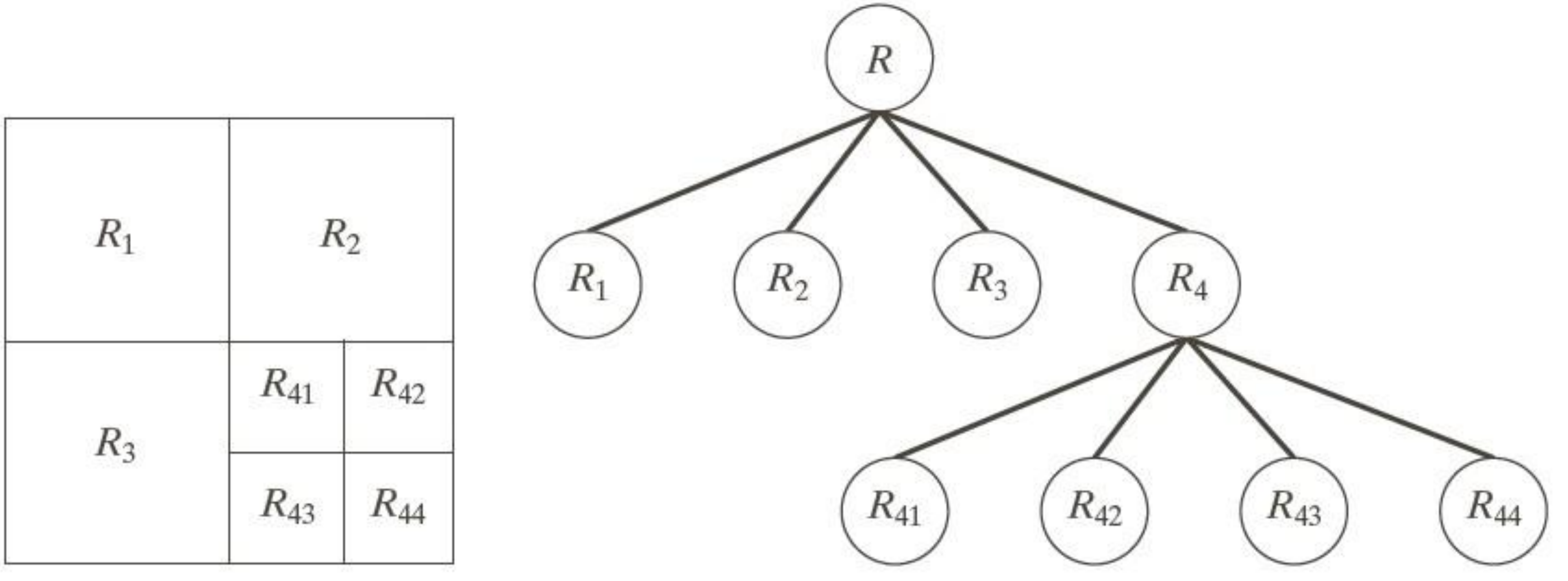
The preceding discussion can be summarized by the following procedure in which, at any step, we

1. Split into four disjoint quadrants any region R_i for which $Q(R_i) = \text{FALSE}$.
2. When no further splitting is possible, merge any adjacent regions R_j and R_k for which $Q(R_j \cup R_k) = \text{TRUE}$.
3. Stop when no further merging is possible.

It is customary to specify a minimum quadregion size beyond which no further splitting is carried out.

Region Splitting and Merging

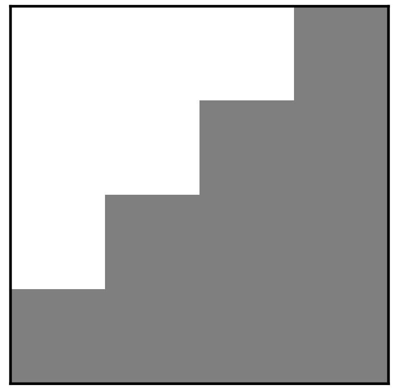
Quadregions resulted from splitting.



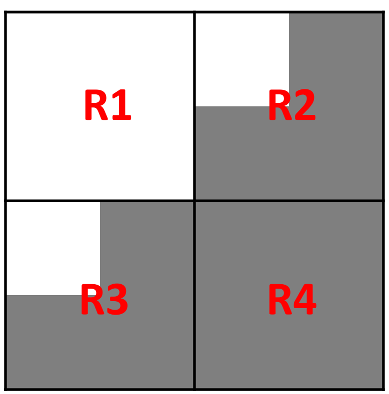
Merging examples:

- R_2 may be merged with R_{41}.
- R_{41} may be merged with R_{42}.

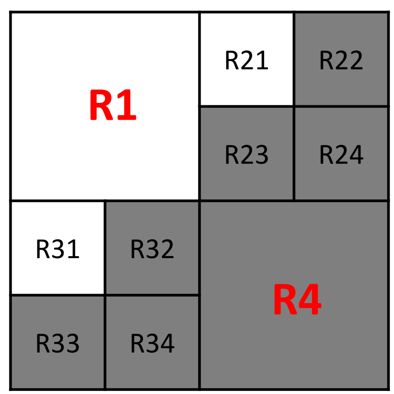
Example



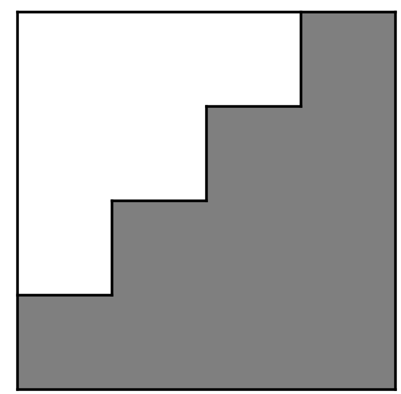
Input image



Split

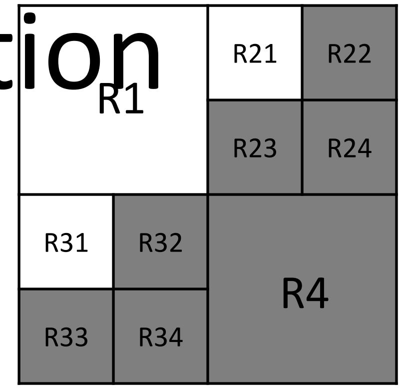


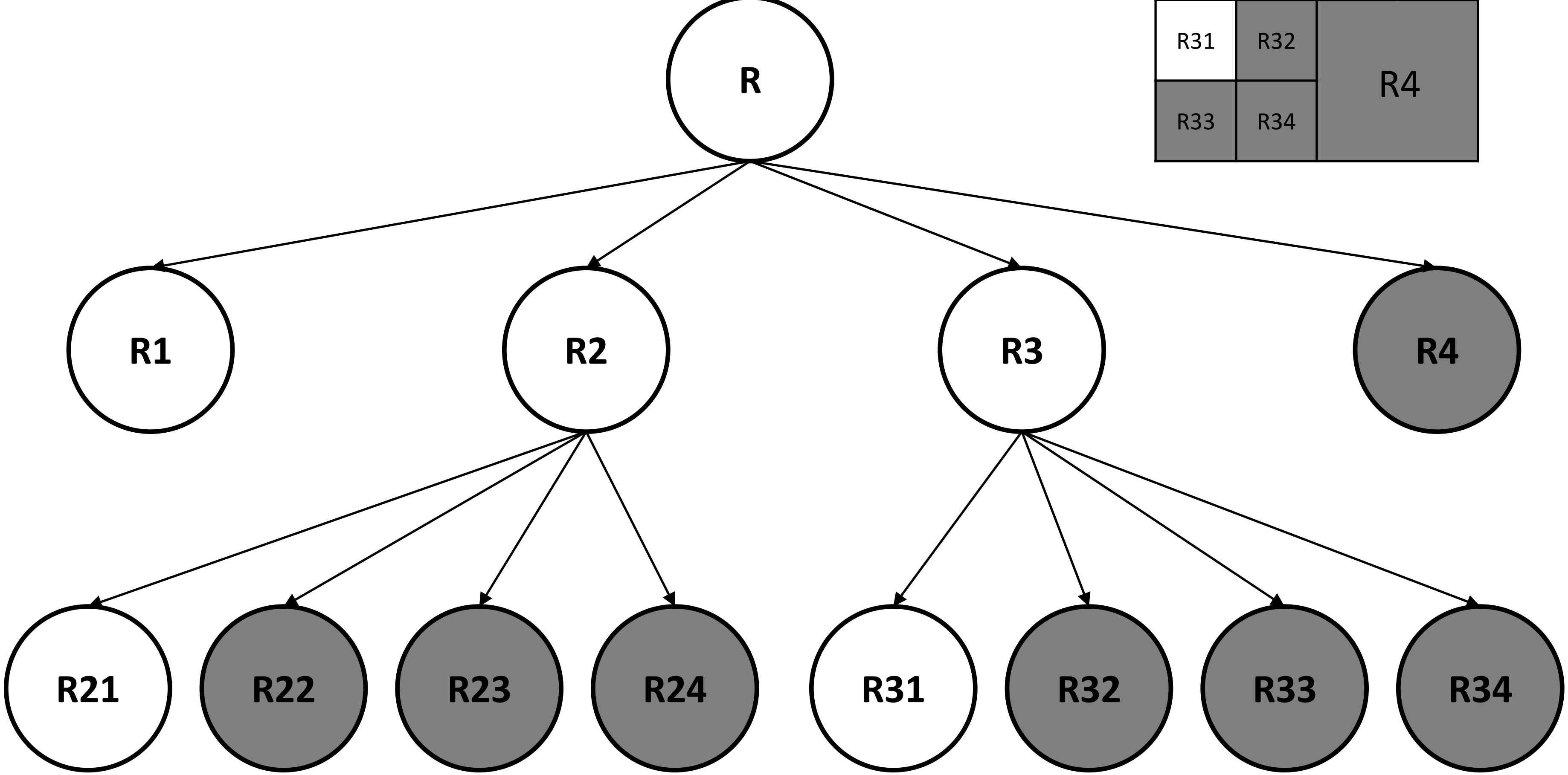
Split

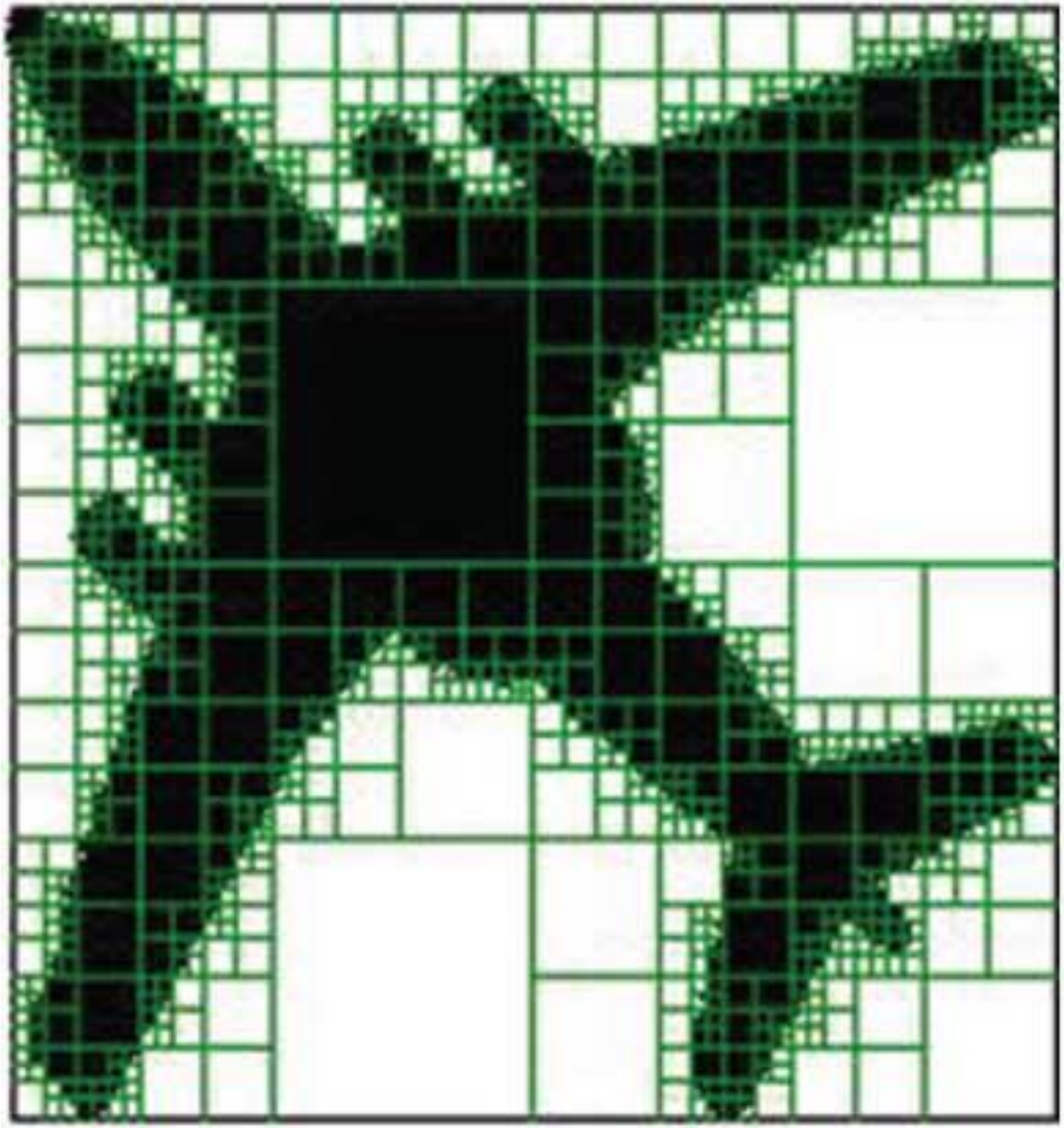


Merge

Quadtree representation

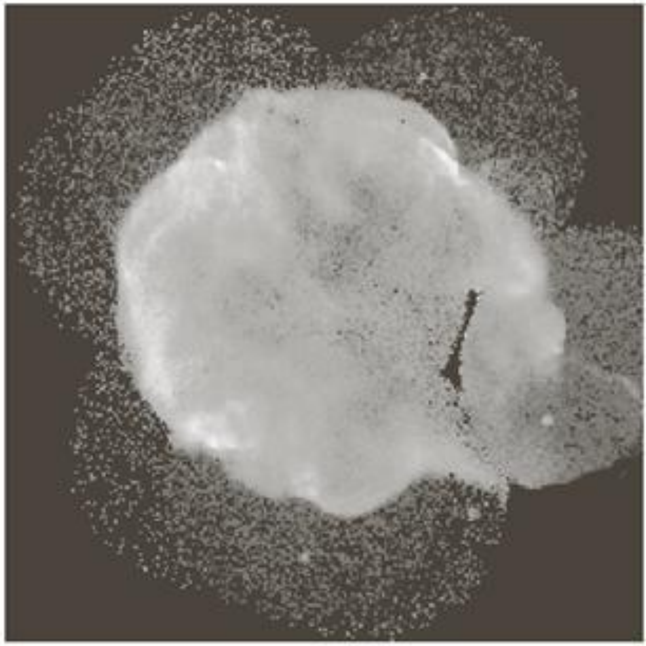






Region Splitting and Merging

- We want to segment the outer ring of less dense matter.
- Characteristics of the region of interest:
  - Standard deviation grater than the background (which is near zero) and the central region (which is smoother).
  - Mean value greater than the mean of background and less than the mean of the central region.

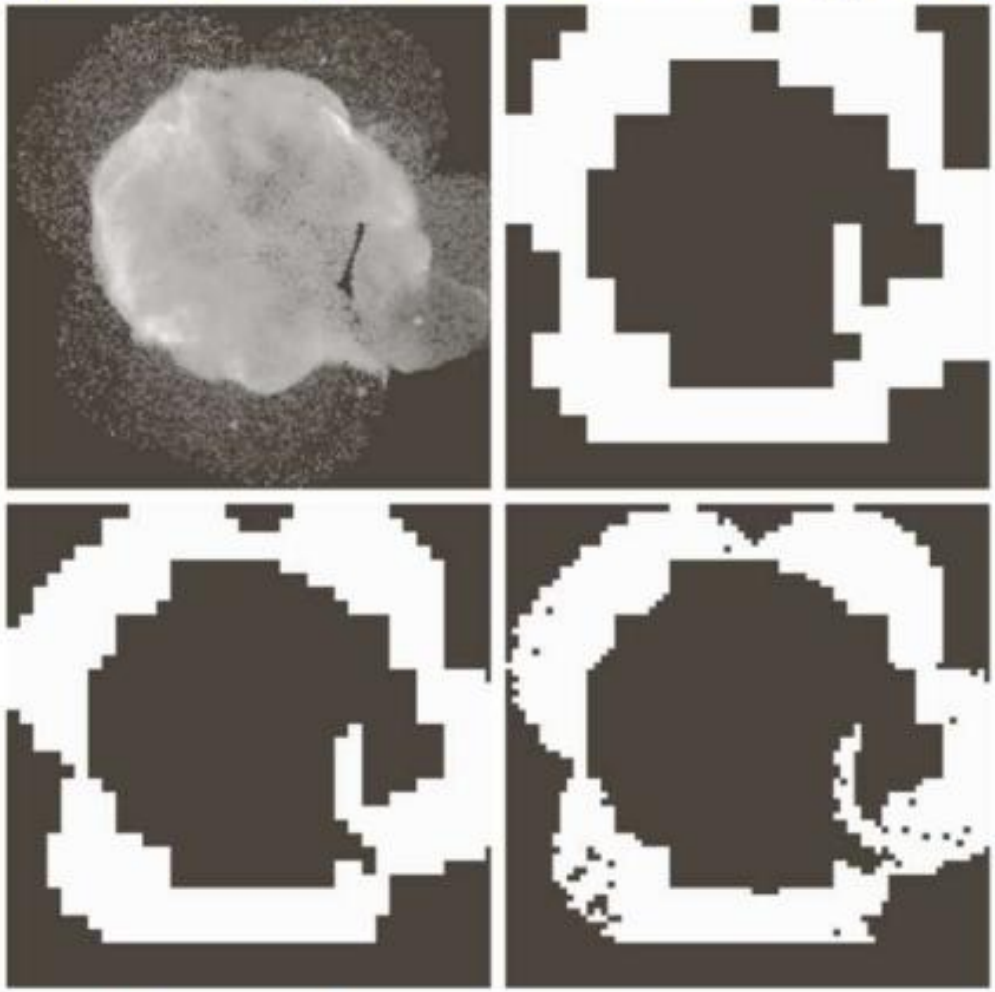


- Predicate:

$$Q = \begin{cases} \text{true} & \sigma > \alpha \ AND \ \ 0 < m < b \\ \text{false} & \text{otherwise} \end{cases}$$

Region splitting and merging

Original image



Smallest Quad-Region: 32x32

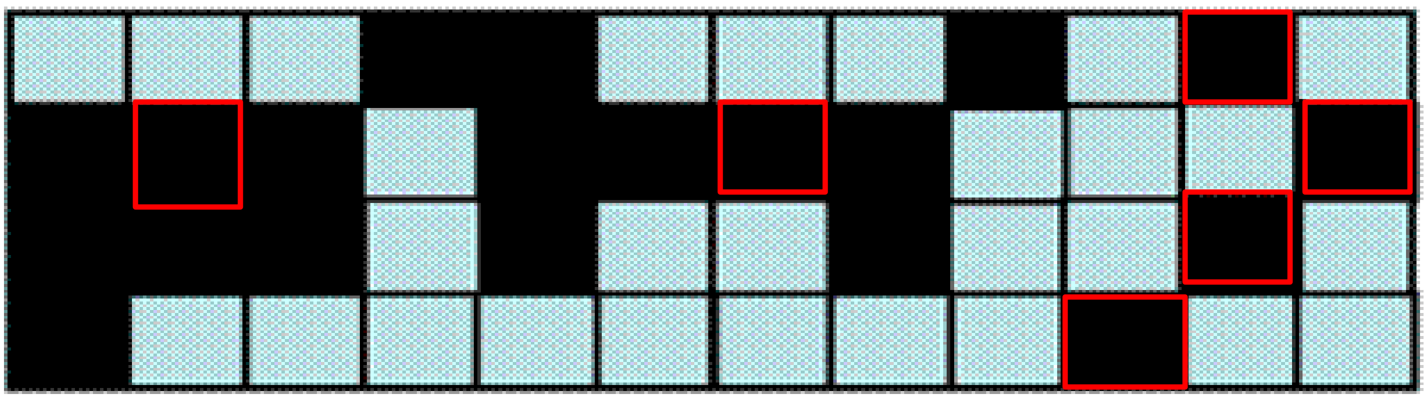
Smallest Quad-Region: 16x16

Smallest Quad-Region: 8x8

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

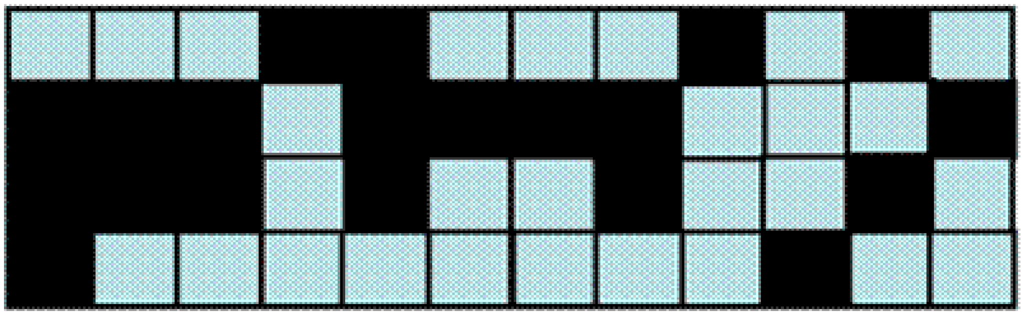
Example:

- For image of shown in figure below , segment foreground and background using region growing method. Seed pixels are shown with red boundary and condition for similarity is |seed-f(x,y)|<1
- Show your results for 1) 8 connectivity 2) 4- connectivity

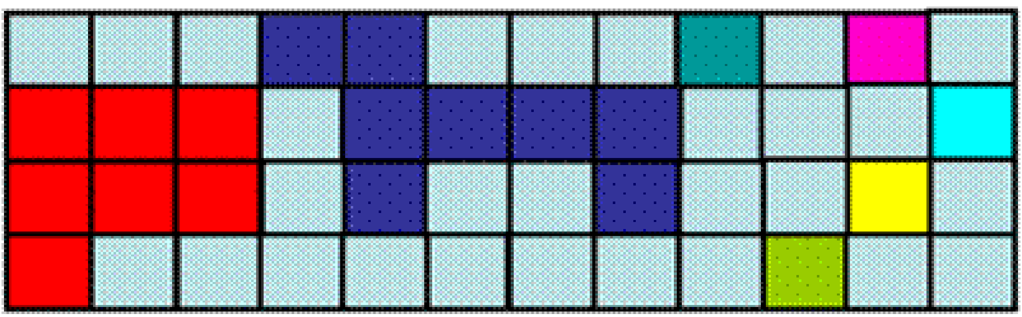


Binary image:
0 - objects;
1 -background

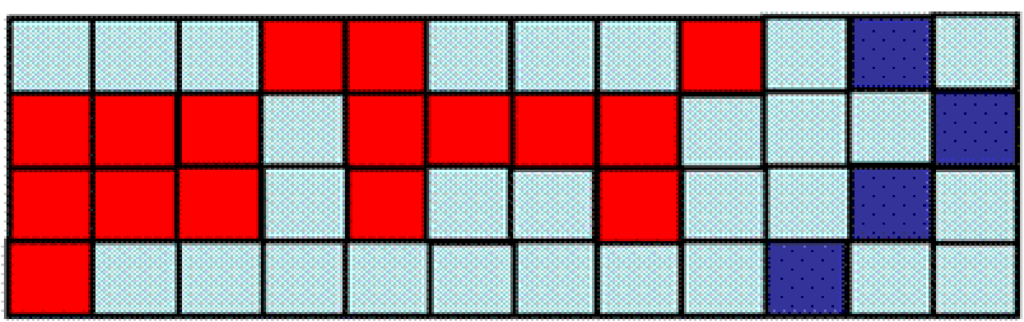
Solution:



Binary image:
0 - objects;
1 -background



4-connected
objects +
8-connected
background



8-connected
objects +
8-connected
background

Example :

4	5	9	9
3	3	1	5
4	2	2	5
10	3	6	9
9	9	4	6
9	5	8	7
6	6	3	8
4	5	5	7
10	11	9	7

- For image of size 9 x4 , segment foreground and background using region growing method if seed pixel is the pixel with intensity ‘1’ and condition for similarity is

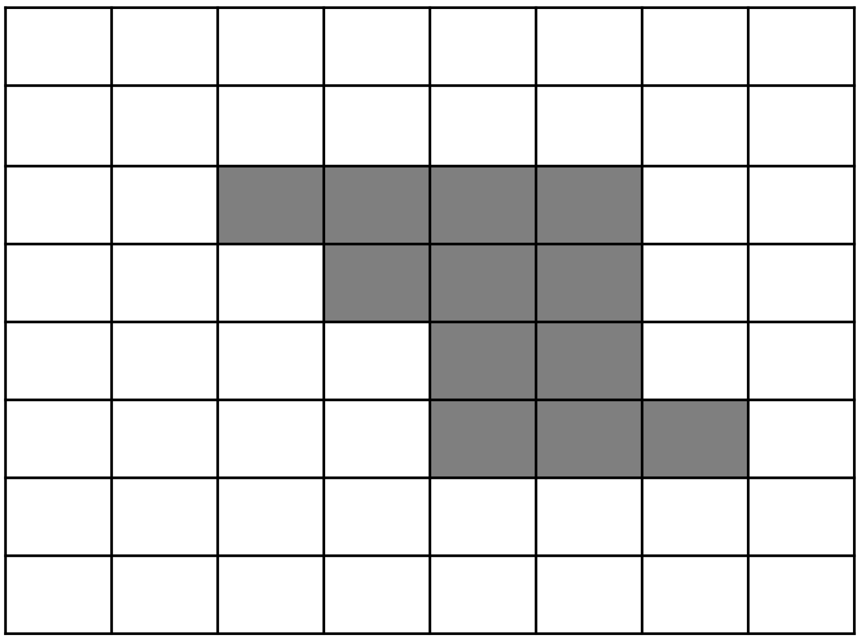
  $|seed-f(x,y)|<=1$

  Use 8-connectivity in the algorithm.

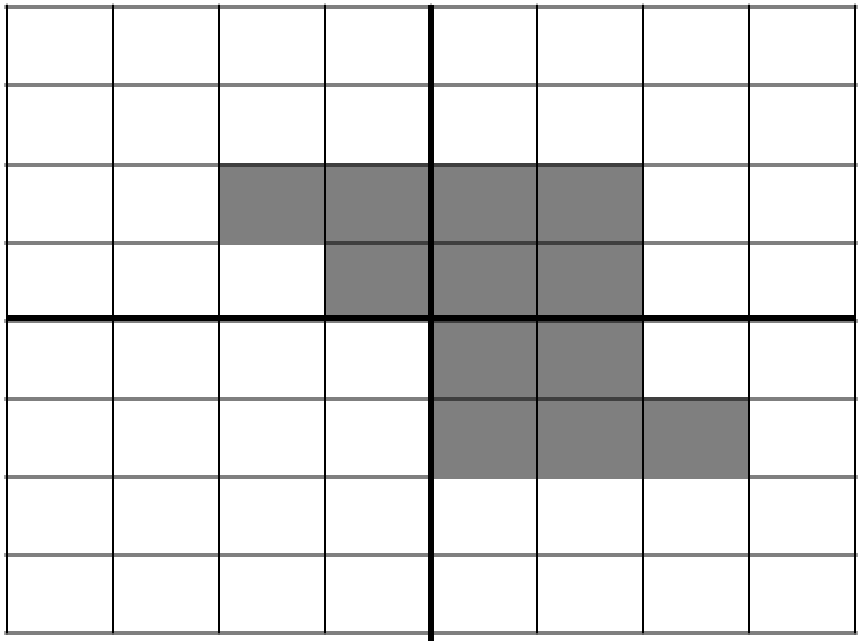
4	5	9	9
3	3	1	5
4	2	2	5
10	3	6	9
9	9	4	6
9	5	8	7
6	6	3	8
4	5	5	7
10	11	9	7

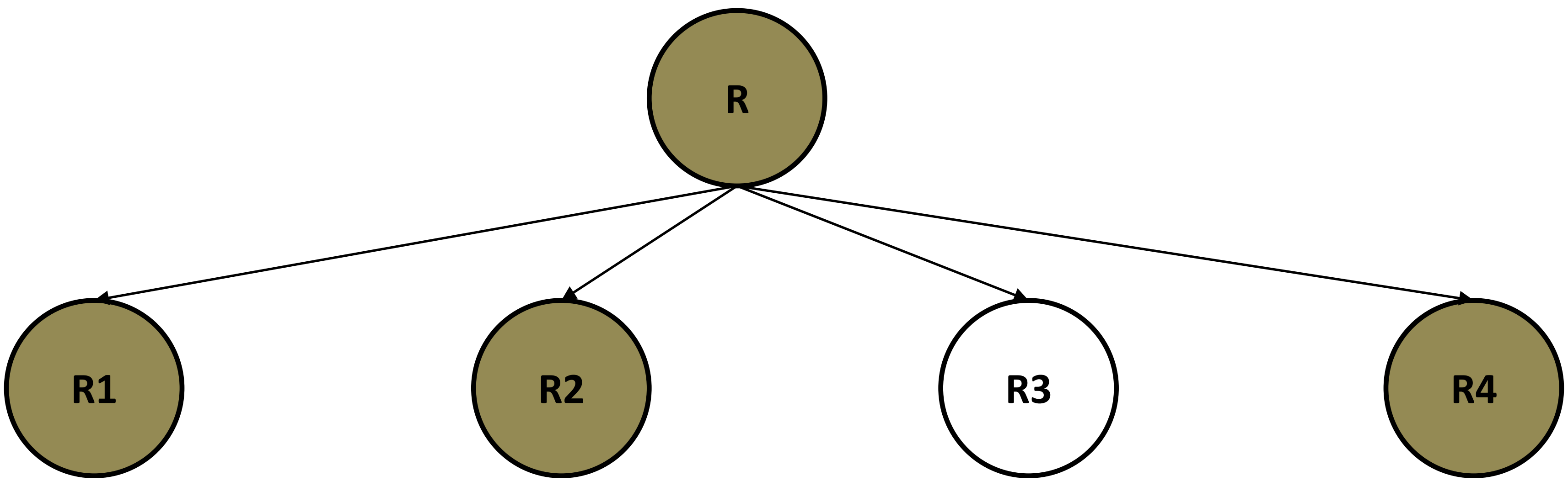
4	5	9	9
3	3	1	5
4	2	2	5
10	3	6	9
9	9	4	6
9	5	8	7
6	6	3	8
4	5	5	7
10	11	9	7

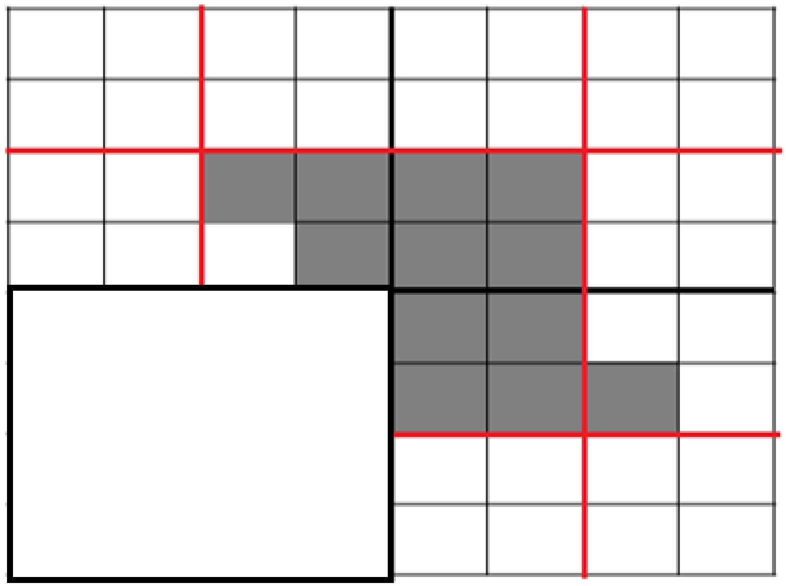
Example:

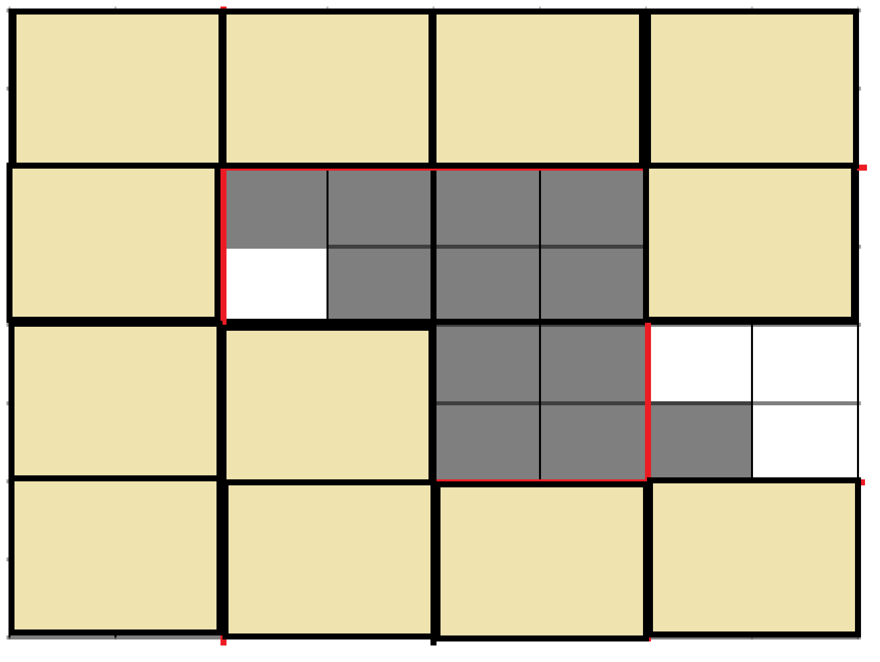


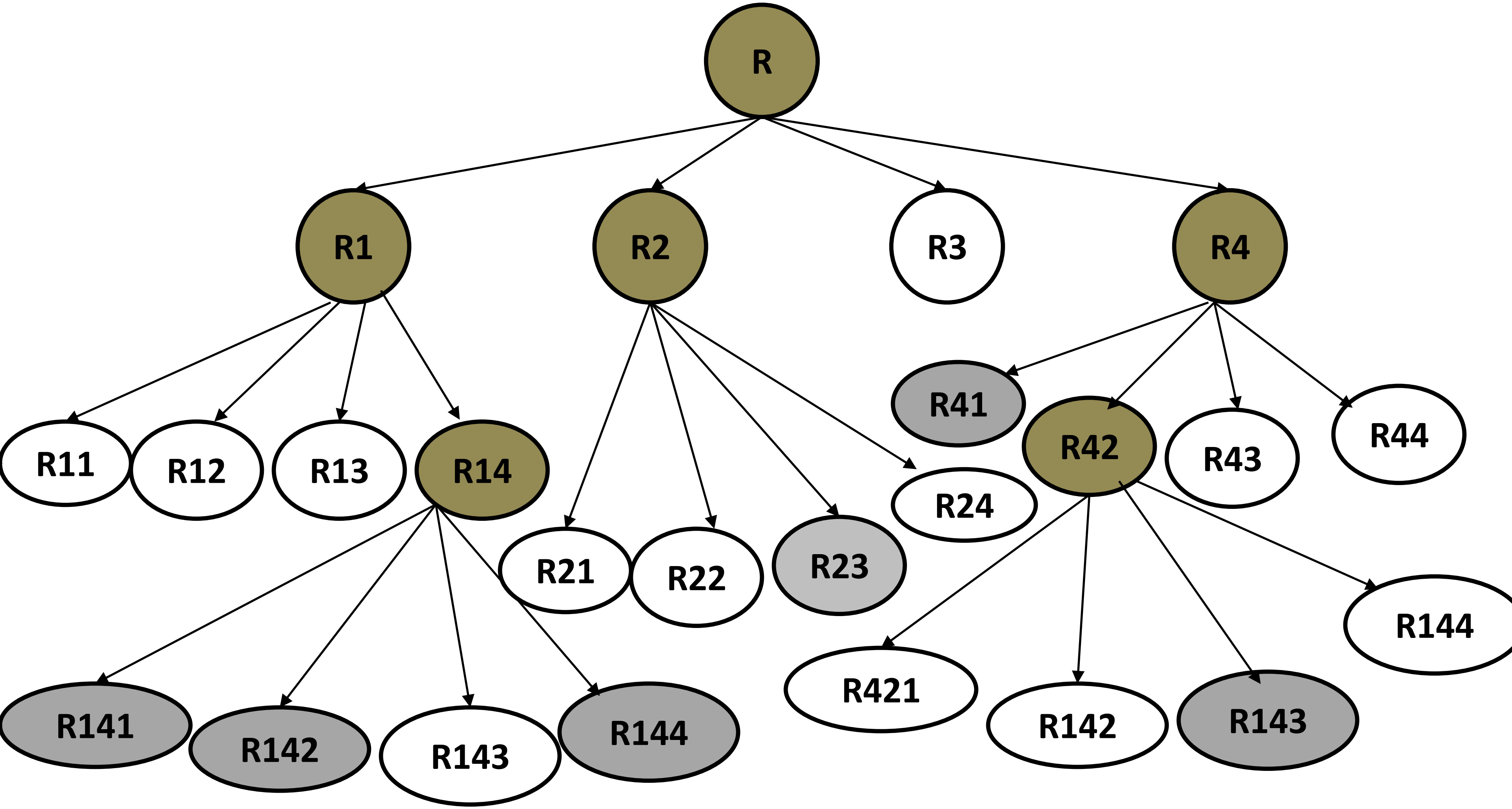
- Consider a 8 x 8 image. Segment the image using the splitt and merge procedure. Let Q(Ri)=TRUE if all pixels in Ri have the same intensity. Show quad tree corresponding to your procedure.




R
R1
R2
R3
R4






R
R1
R2
R3
R4
R11
R12
R13
R14
R21
R22
R23
R24
R41
R42
R43
R44
R141
R142
R143
R144
R421
R142
R143
R144

- Consider a following image. Segment the image using the Region growing procedure. Let Q(Ri)=TRUE if |seed-f(x,y)|<=1. Seed pixels are highlighted in image. Use 8 connectivity for algorithm

0	1	5	6	7
1	1	5	8	7
0	**1**	6	**7**	7
2	0	7	6	6
0	1	5	6	5

- Consider a following image. Segment the image using the Region growing procedure. Let Q(Ri)=TRUE if |seed-f(x,y)|<=7. Seed pixels are highlighted in image. Use 8 connectivity for algorithm

0	1	5	6	7
1	1	5	8	7
0	**1**	6	**7**	7
2	0	7	6	6
0	1	5	6	5