CSL7320: Digital Image Analysis

Segmentation

Image Segmentation

A process that partitions R into subregions $R_1, R_2, ..., R_n$

Microsoft multiclass segmentation data set

ide credit: Yan Tong

Image Segmentation - Applications

Object localization

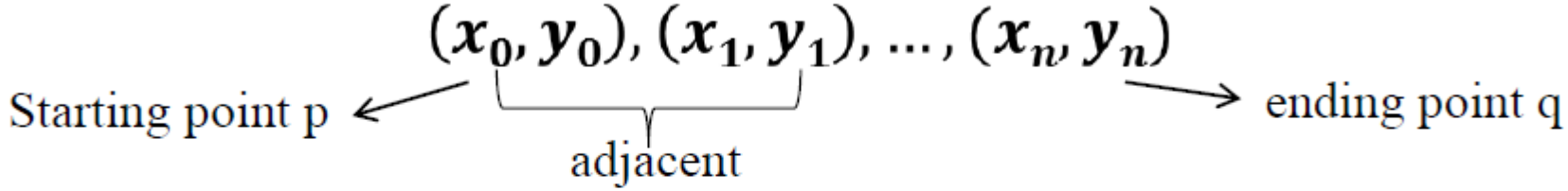
Object recognition

Specially important for medical imaging

Slide credit: Yan Tong

Brief Review of Connectivity

- Path from p to q: a sequence of distinct pixels with coordinates



- p and q are *connected*: if there is a path from p to q in S
- *Connected component*: all the pixels in S connected to p
- *Connected set*: S has only one connected component
- R is a region if R is a connected set
- R_i and R_j are adjacent if $R_i \cup R_j$ is a connected set

Slide credit: Yan Tong

Image Segmentation

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

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

(c) $R_i \cap R_j = \phi, \forall i \neq j$

(d) $Q(R_i) = TRUE$ ← Q is a criterion

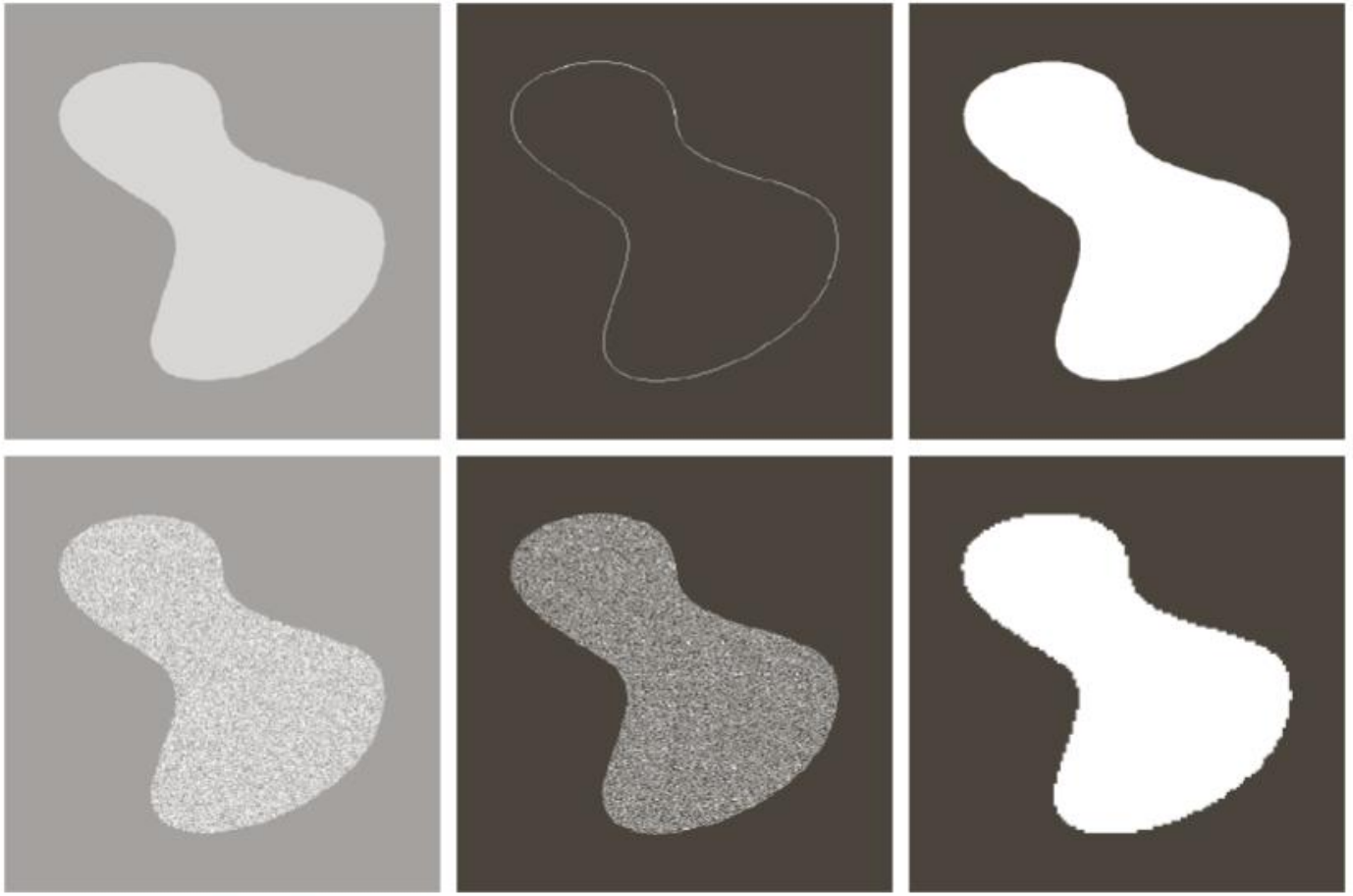
(e) $Q(R_i \cup R_j) = FALSE$ for adjacent regions R_i and R_j

Two categories based on intensity properties:

- Discontinuity – edge-based algorithms
- Similarity – region-based algorithms

Slide credit: Yan Tong

Image Segmentation



a b c
d e f

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

Slide credit: Yan Tong

Point, Line and Edge Detection

Segmentation based on detecting sharp, local changes in intensity

- **Edge pixels: pixels at which the intensity changes abruptly**
- **Edges: sets of connected edge pixels**
  - Edge detectors: methods to detect edges
- **Line (roof edges): edge segments in which the intensity on either side of the line is either higher or lower than the intensity of line pixels**
  - isolated point: a line with one-pixel length

Slide credit: Yan Tong

Background

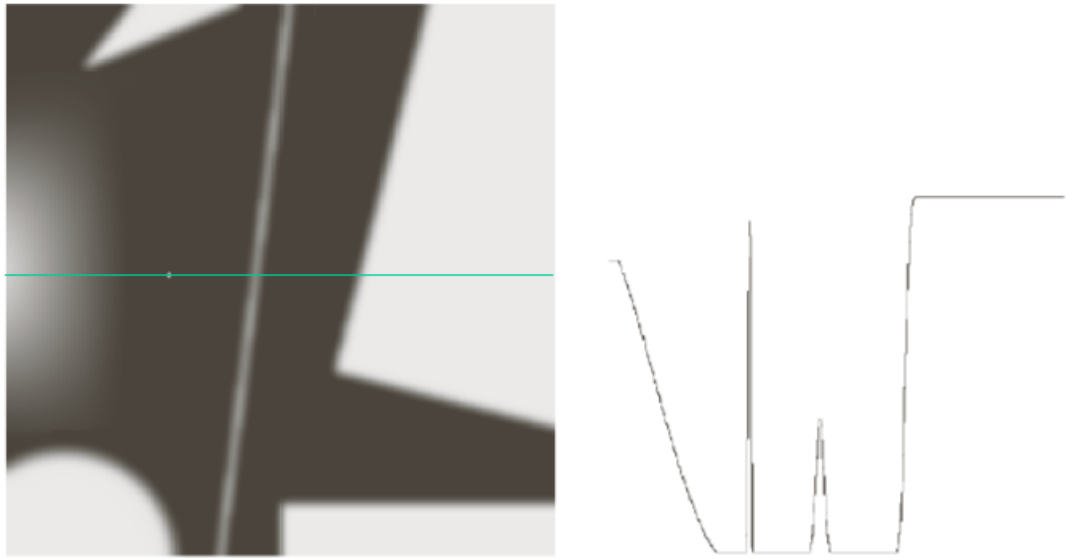
The first-order derivative

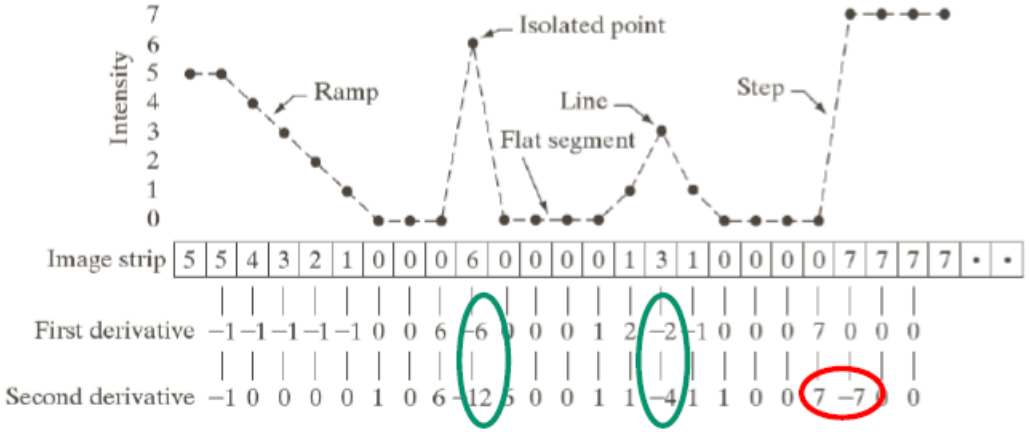
$$\frac{\partial f}{\partial x} = f'(x) = f(x+1) - f(x)$$

The second-order derivative

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

$$= f(x+1) - 2f(x) + f(x-1)$$





Slide credit: Yan Tong

Some Observations

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

Implementation by filtering

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

FIGURE 10.3
A general 3×3 spatial filter mask.

Slide credit: Yan Tong

Detection of Isolated Points

Laplacian

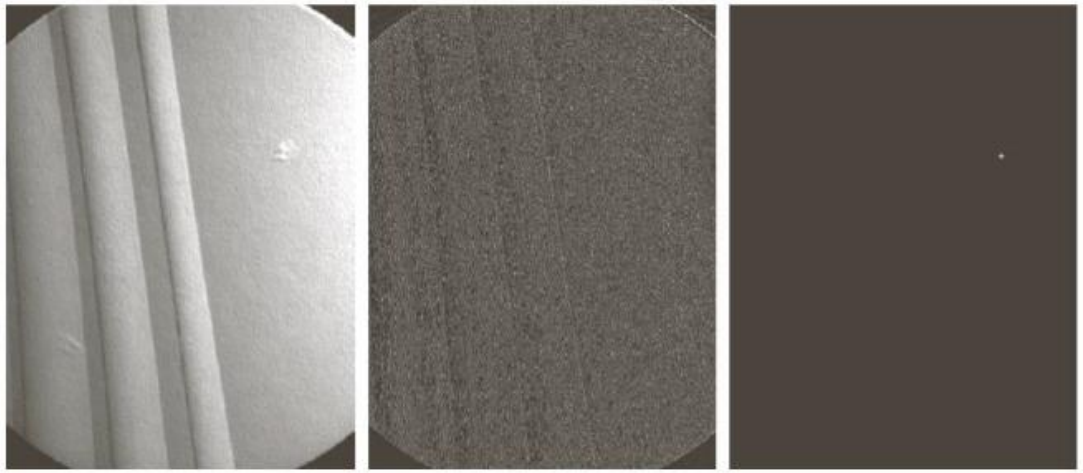
$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

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

*

$$R = \sum_{k=1}^{9} w_k z_k$$

1	1	1
1	−8	1
1	1	1

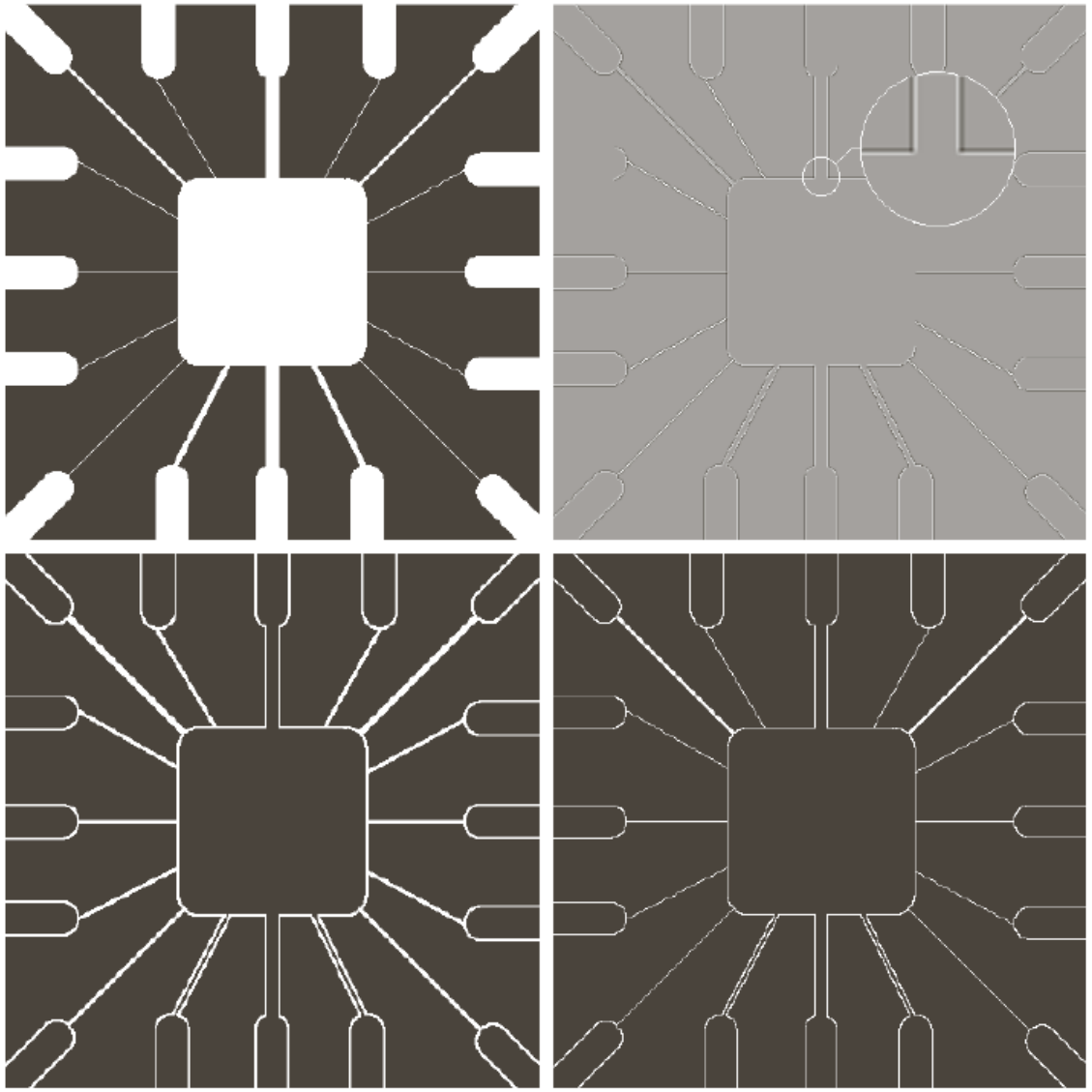


a
b c d

FIGURE 10.4 (a) Point detection (Laplacian) mask. (b) X-ray image of turbine blade with a porosity. The porosity contains a single black pixel. (c) Result of convolving the mask with the image. (d) Result of using Eq. (10.2-8) * showing a single point (the point was enlarged to make it easier to see). (Original image courtesy of X-TEK Systems, Ltd.)

Slide credit: Yan Tong

Line Detections by Laplacian



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.

The line detection is insensitive to the direction

Slide credit: Yan Tong

Line Detections 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

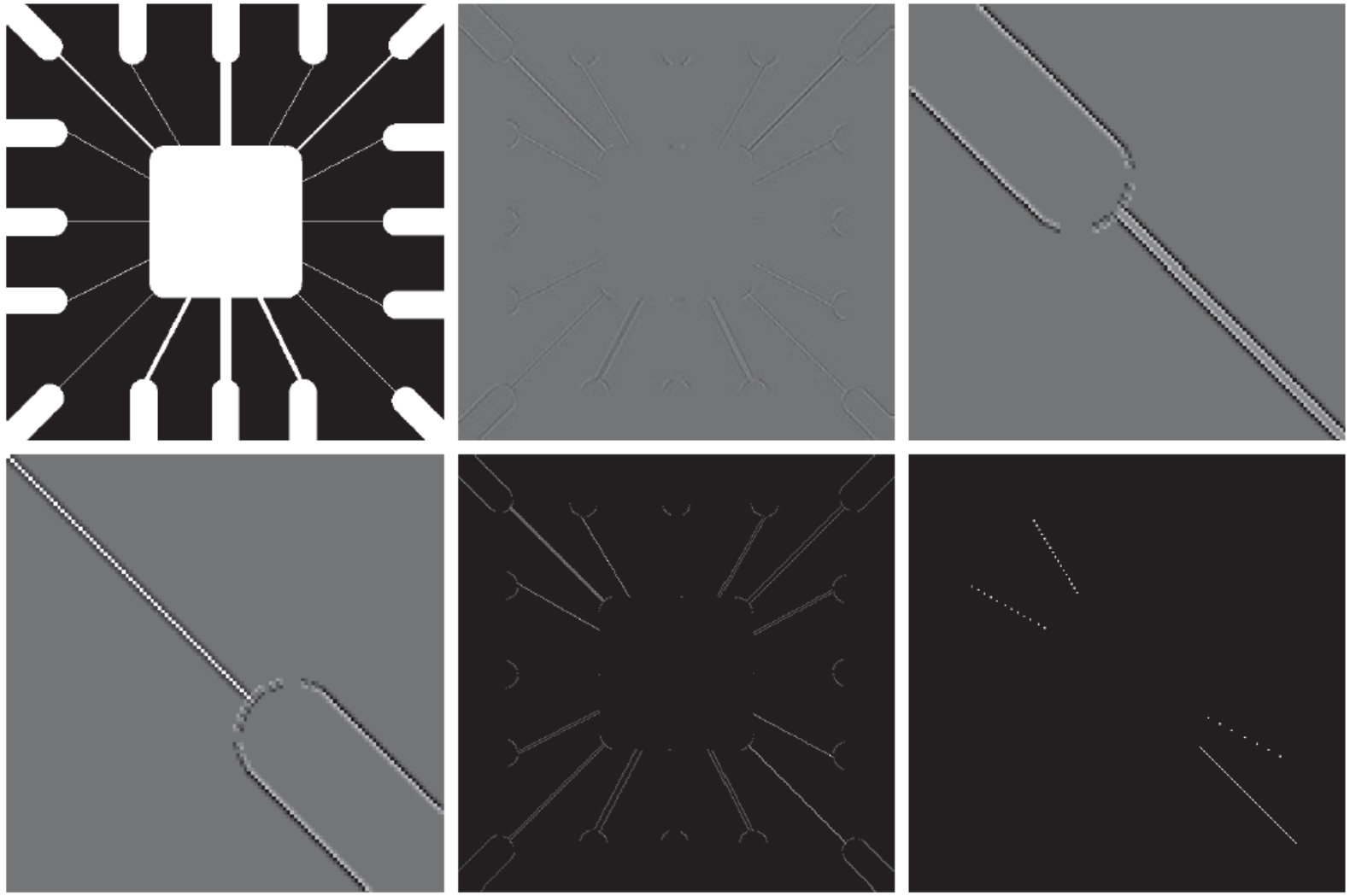
−1	−1	2
−1	2	−1
2	−1	−1

−45°

FIGURE 10.6 Line detection masks. Angles are with respect to the axis system in Fig. 2.18(b).

Slide credit: Yan Tong

An Example



a b c
d e f

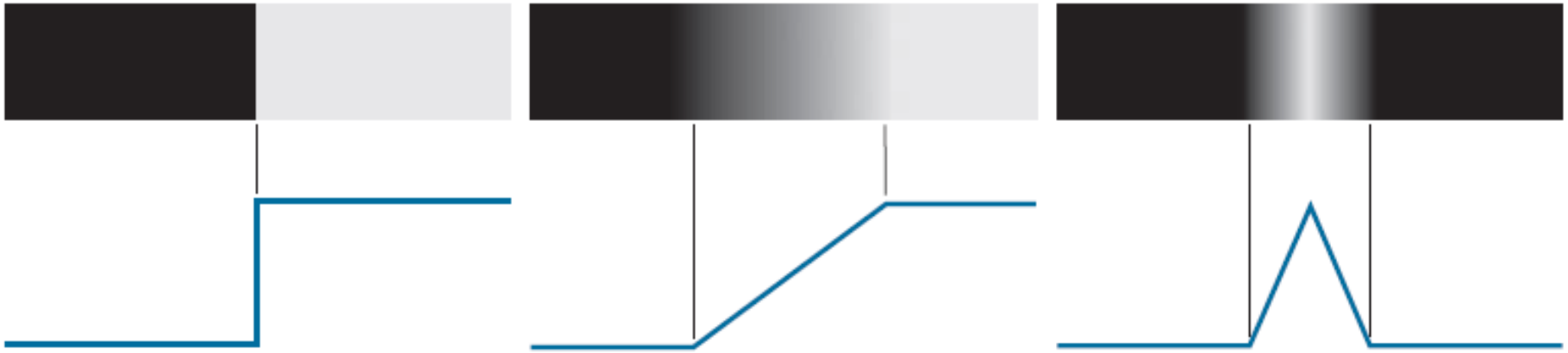
FIGURE 10.7 (a) Image of a wire-bond template. (b) Result of processing with the +45° line detector kernel in Fig. 10.6. (c) Zoomed view of the top left region of (b). (d) Zoomed view of the bottom right region of (b). (e) The image in (b) with all negative values set to zero. (f) All points (in white) 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.)

Slide credit: Yan Tong

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.



Slide credit: Yan Tong

Edge Models

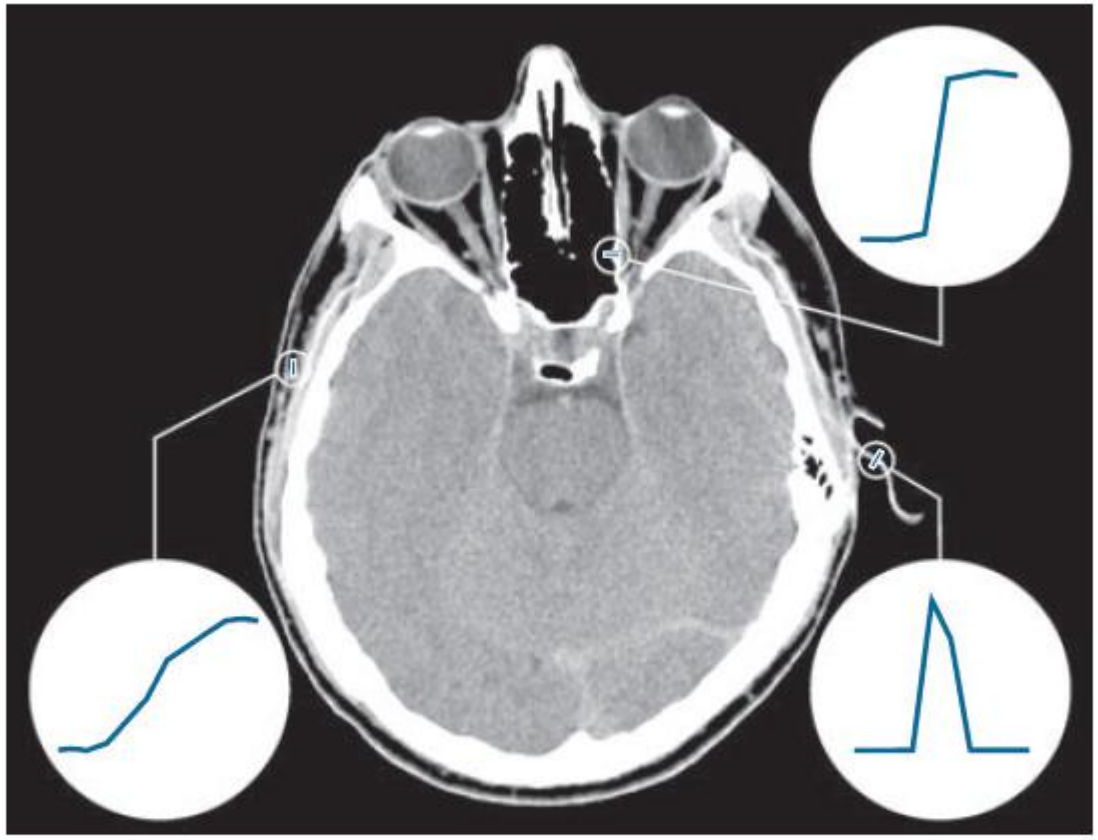
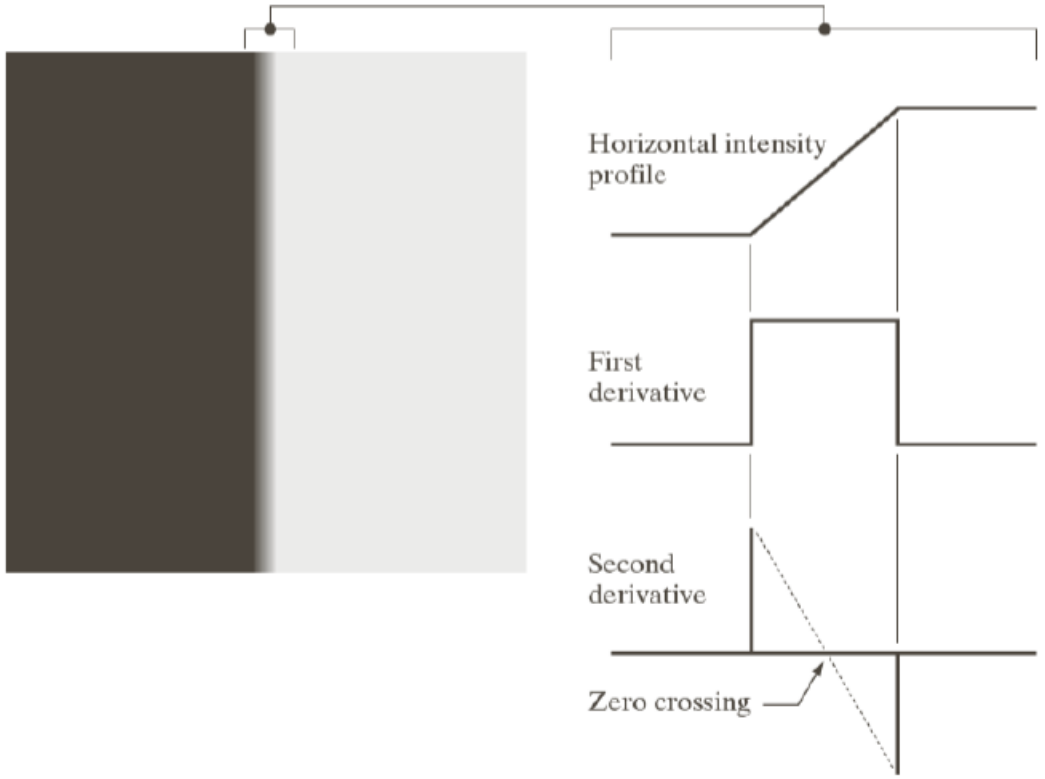


FIGURE 10.9 A 1508 × 1970 image showing (zoomed) actual ramp (bottom, left), step (top, right), and roof edge profiles. The profiles are from dark to light, in the areas enclosed by the small circles. The ramp and step profiles span 9 pixels and 2 pixels, respectively. The base of the roof edge is 3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

Slide credit: Yan Tong

Derivatives – Idea Cases

Observation:



a b

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

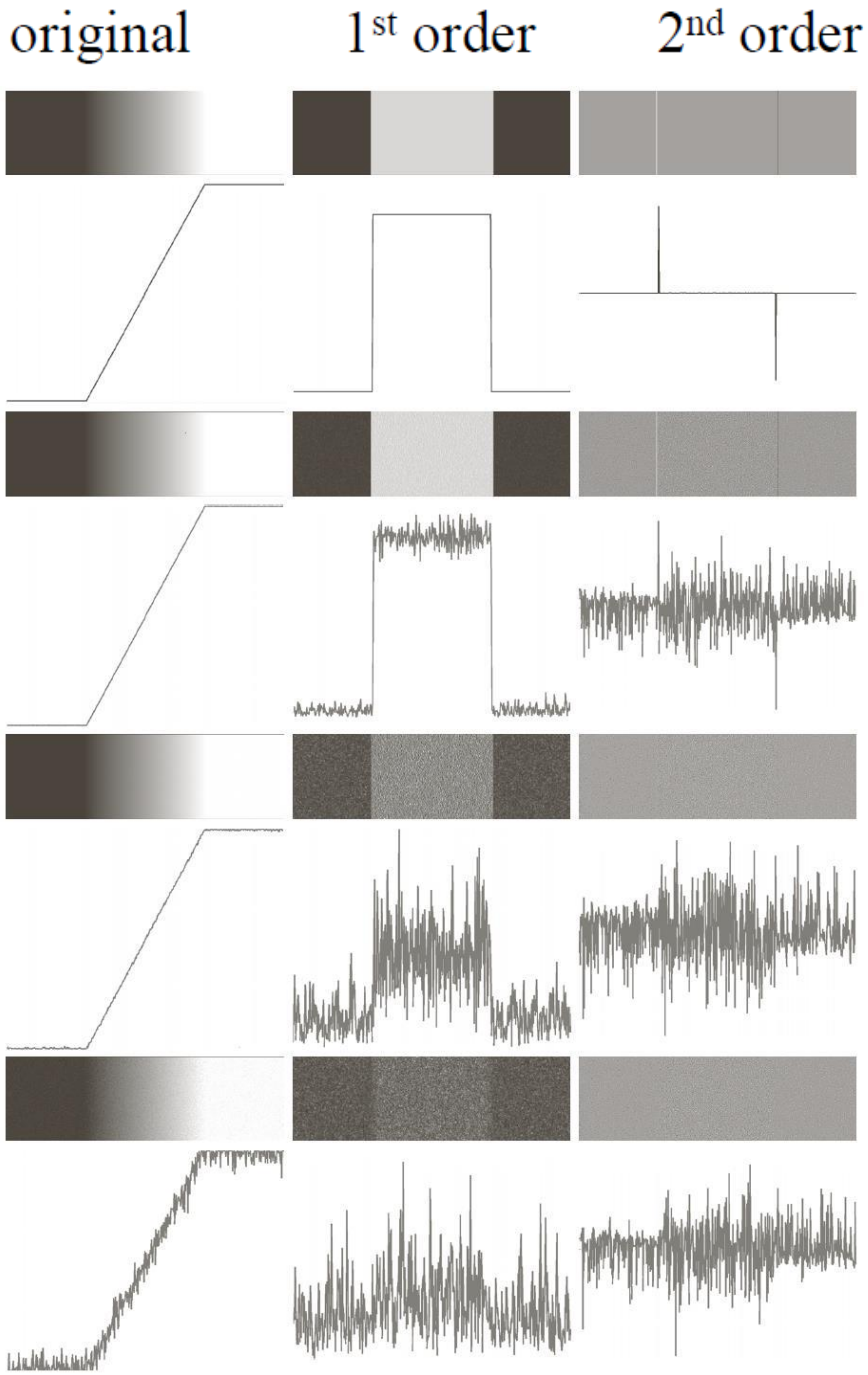
Conclusion

- The magnitude of first derivative -- the presence of an edge
- The sign of second derivative -- whether an edge pixel lies on the dark or light side of an edge
  - produces two values for every edge in an image
  - zero crossing – the center of ramp/thick edges

Slide credit: Yan Tong

But, In Practice …

FIGURE 10.11 First column: Images and intensity profiles of a ramp edge corrupted by random 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.



In summary, the three steps performed typically for edge detection are:

1. ***Image smoothing for noise reduction***. The need for this step is illustrated by the results in the second and third columns of Fig. 10.11.
2. ***Detection of edge points***. As mentioned earlier, this is a local operation that extracts from an image all points that are potential edge-point candidates.
3. ***Edge localization***. The objective of this step is to select from the candidate points only the points that are members of the set of points comprising an edge.

Slide credit: Yan Tong

Basic Edge Detection

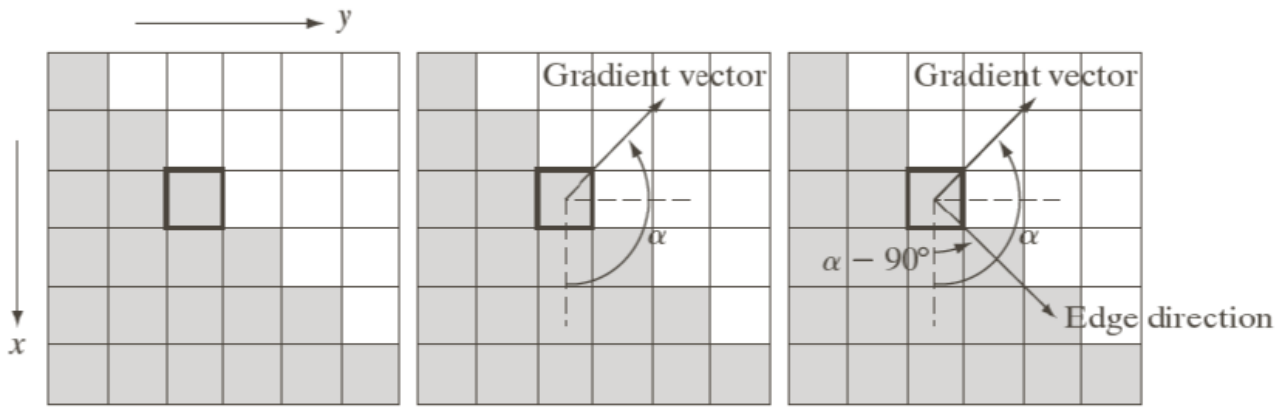
First-order derivative:

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

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

$$\approx |g_x| + |g_y|$$

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



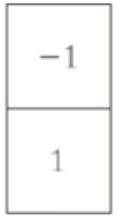
a b c

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

Slide credit: Yan Tong

Masks for Calculating the Gradient (2x2)

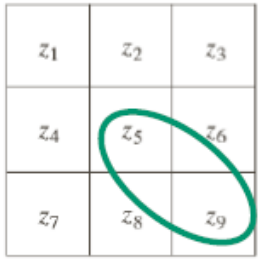
Gradient in vertical/horizontal direction

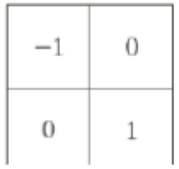


Along y => g_y

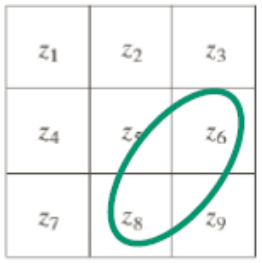
Along x => g_x

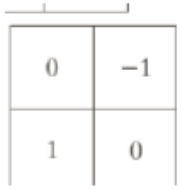
Gradient in diagonal direction





Along x => g_x





Along y => g_y

Slide credit: Yan Tong

Masks for Calculating the Gradient (3x3)

Gradient in vertical/horizontal

Along x => g_x

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

Along y => g_y

Prewitt

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Sobel

Gradient in diagonal

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

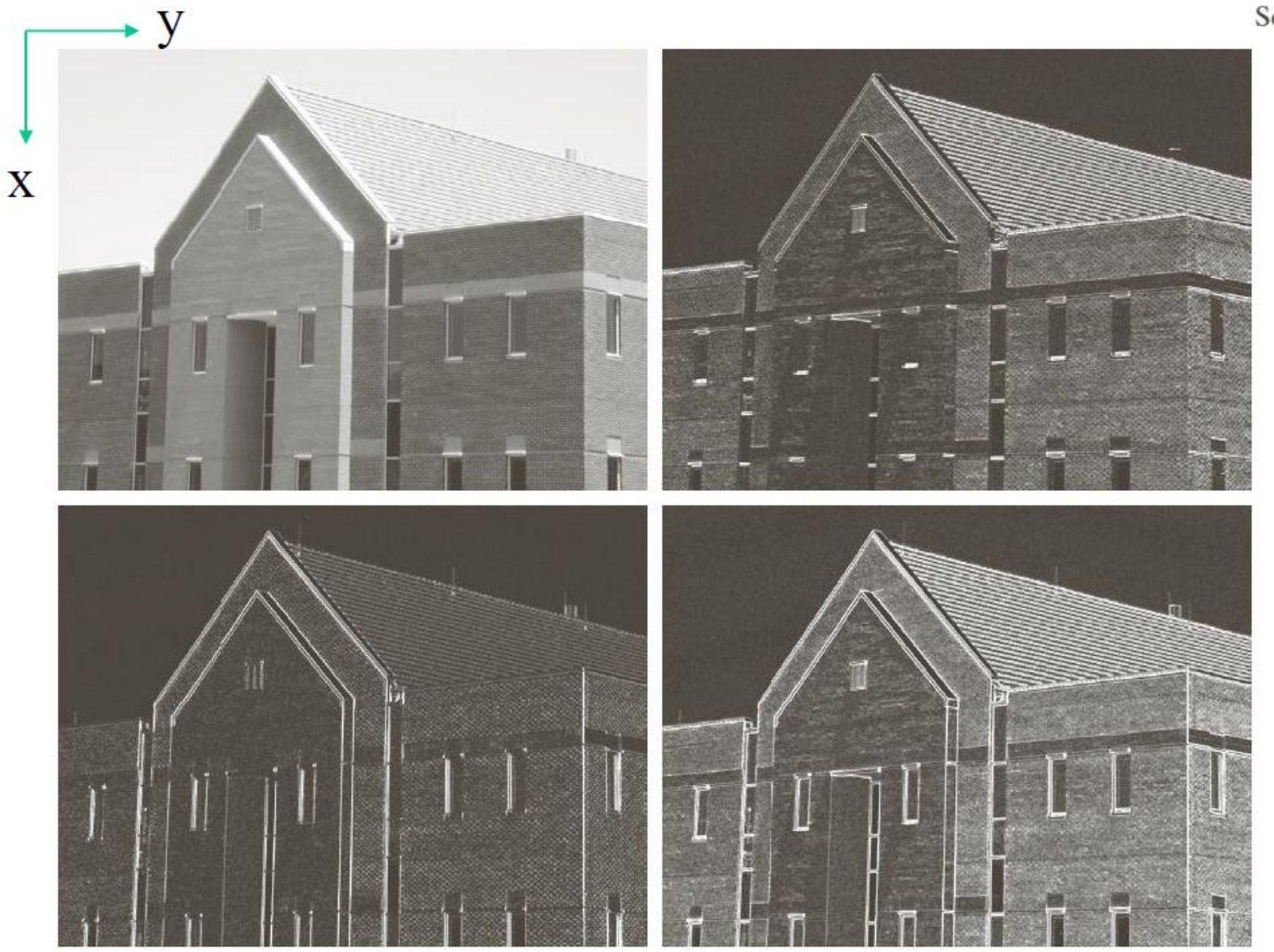
−2	−1	0
−1	0	1
0	1	2

Sobel

Sobel operator performs edge detection and smoothing simultaneously.

Slide credit: Yan Tong

An Example



Sobel

−1	−2	−1
0	0	0
1	2	1

Along x => g_x

−1	0	1
−2	0	2
−1	0	1

Along y => g_y

a b
c d

FIGURE 10.16
(a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) $|g_x|$, the component of the gradient in the x-direction, obtained using the Sobel mask in Fig. 10.14(f) to filter the image. (c) $|g_y|$, obtained using the mask in Fig. 10.14(g). (d) The gradient image, $|g_x| + |g_y|$.

Slide credit: Yan Tong

An Example

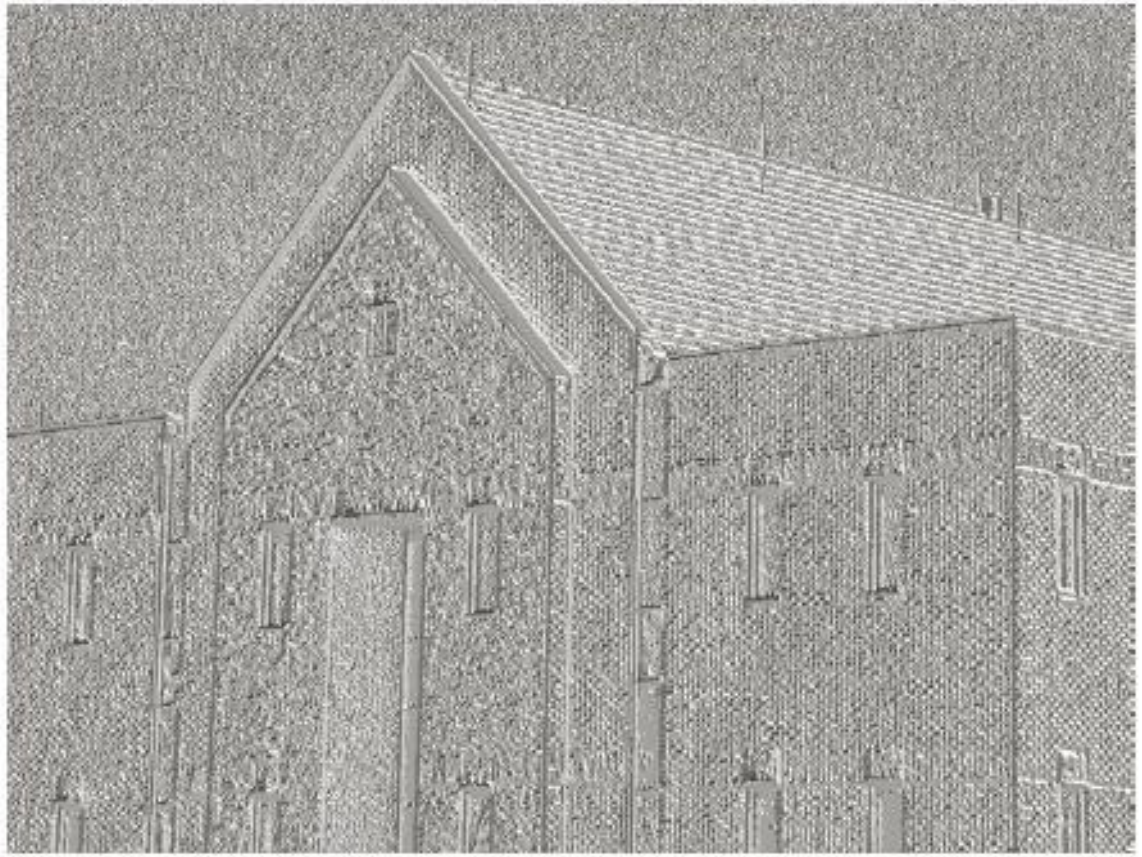


FIGURE 10.17 Gradient angle image computed using Eq. (10.2-11). Areas of constant intensity in this image indicate that the direction of the gradient vector is the same at all the pixel locations in those regions.

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

Slide credit: Yan Tong

An Example

a b
c d

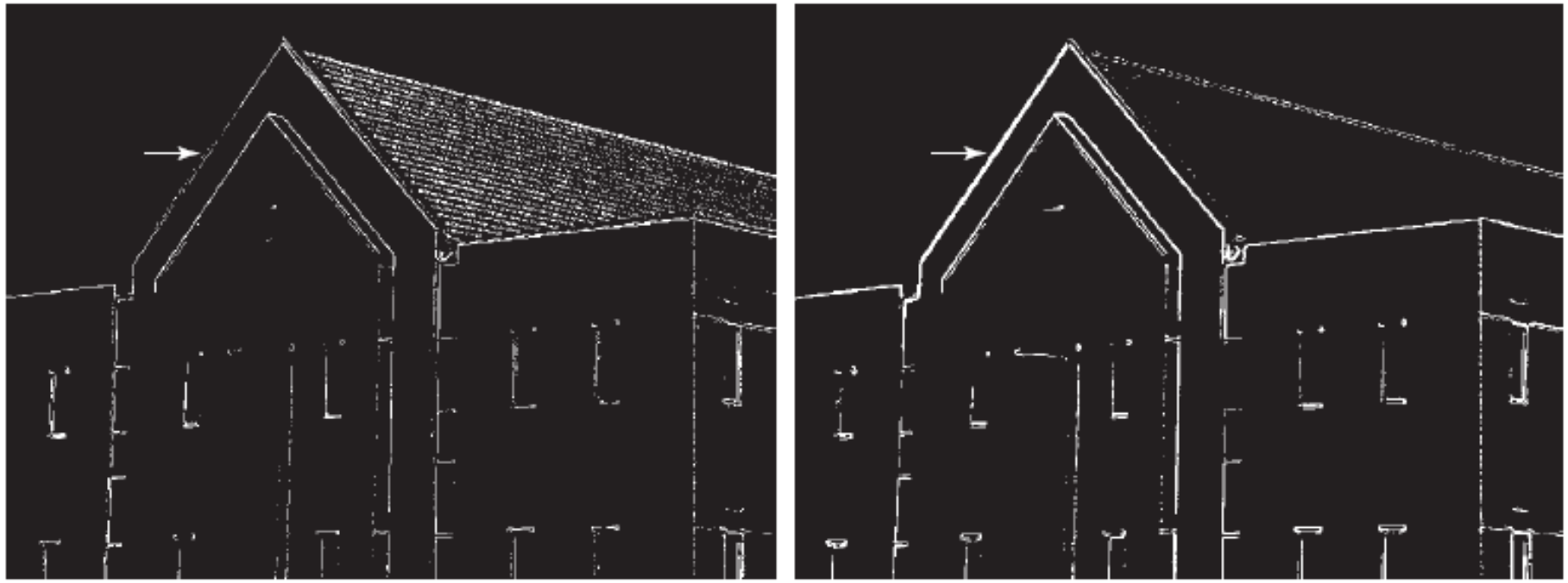
FIGURE 10.18
Same sequence as in Fig. 10.16, but with the original image smoothed using a 5×5 averaging filter prior to edge detection.

Slide credit: Yan Tong

Combining the Gradient with Thresholding: An Example

a b

FIGURE 10.20
(a) Result of thresholding Fig. 10.16(d), the gradient of the original image. (b) Result of thresholding Fig. 10.18(d), the gradient of the smoothed image.



Slide credit: Yan Tong

Edge Magnitude and Direction (Angle)

The *Kirsch compass kernels* in Fig. 10.15, are designed to detect edge magnitude *and* direction (angle) in all eight compass directions.

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.

0 degree

−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

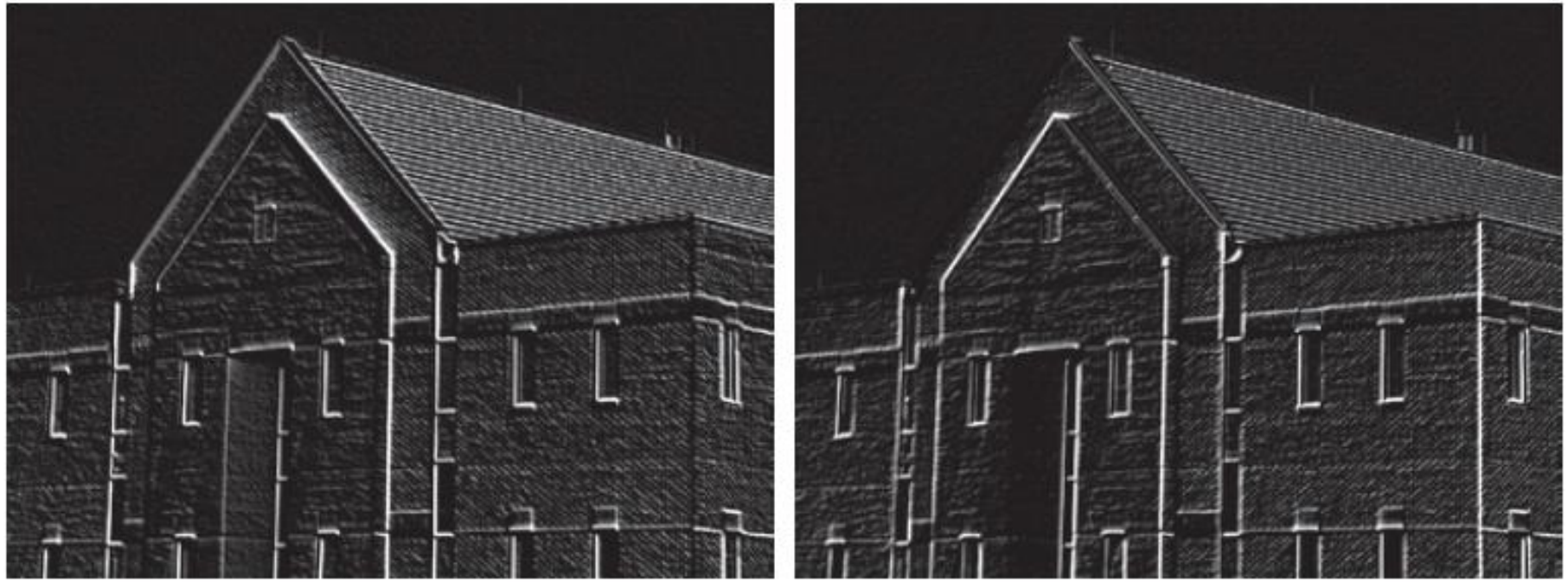
−3	−3	−3
−3	0	5
−3	5	5

NE

Edge Magnitude and Direction (Angle): Example

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).



Summary on Simple Edge Detectors

-First-order derivative

- Roberts (2x2)
- Prewitt (3x3)
- Sobel (3x3, smooth + difference)
- Issues:
  - Thicker edge
  - One operator for one edge direction

-Second-order derivative

- Laplacian (3x3)
- Issues:
  - Double edge
  - Zero-crossing

-Common issues:

- Sensitive to image noise
- Cannot deal with the scale change of the image

Slide credit: Yan Tong

Advanced Edge Detection Techniques

- **Deal with image noise**
- **Exploit the properties of image**

Work much better for real images

Advanced edge detectors:

- Laplacian of Gaussian (LoG)
- Difference of Gaussian (DoG)
- Canny

Slide credit: Yan Tong

Marr-Hildreth Detector (LoG)

Observations:

- Intensity changes are dependent on the image scale
- A sudden intensity change (step) causes a peak/trough in the1st order derivative and a zero-crossing in the 2nd order derivative
- The 2nd order derivative is especially sensitive to noise

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

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

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

Laplacian of a Gaussian (LoG)

Slide credit: Yan Tong

Marr-Hildreth Detector (LoG)

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

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

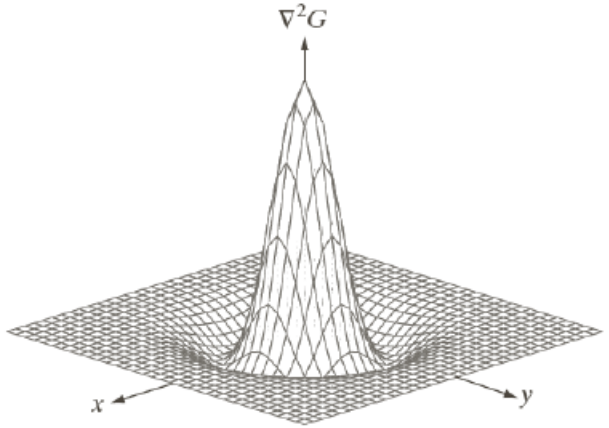
$$\nabla^2 G(x, y) = \left[\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$

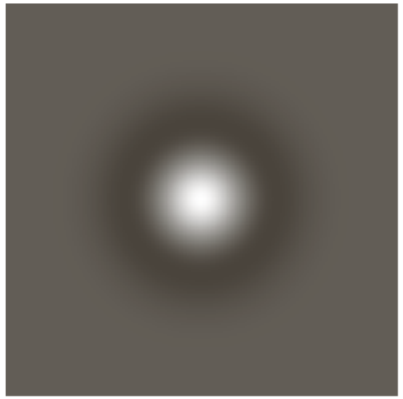
$$\begin{aligned}\nabla^2 G(x,y) &= \frac{\partial^2 G(x,y)}{\partial x^2} + \frac{\partial^2 G(x,y)}{\partial y^2} \\ &= \frac{\partial}{\partial x}\left(\frac{-x}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}\right) + \frac{\partial}{\partial y}\left(\frac{-y}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}\right)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

- Varying σ values for scale changes
- Rotation invariant in edge detection

Slide credit: Yan Tong

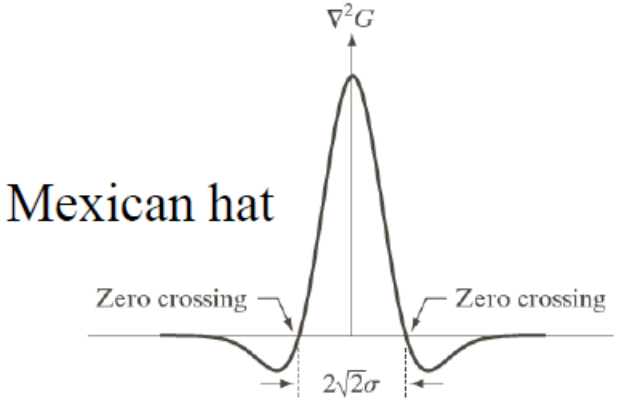
Marr-Hildreth Detector (LoG)





a b
c d

FIGURE 10.21 (a) Three-dimensional plot of the *negative* of the LoG. (b) Negative of the LoG displayed as an image. (c) Cross section of (a) showing zero crossings. (d) 5×5 mask approximation to the shape in (a). The negative of this mask would be used in practice.



0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

→ Sum to zero!

In practice, we use the negative of the mask

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

Slide credit: Yan Tong

LoG Filtering

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

1. **Filter the input image with an $n \times n$ Gaussian filter.**
2. **Compute the Laplacian of the intermediate image resulting from Step 1.**
3. **Find the zero-crossings of the image from Step 2.**
   - opposite signs of the neighbors
   - the difference should be significant

Note:

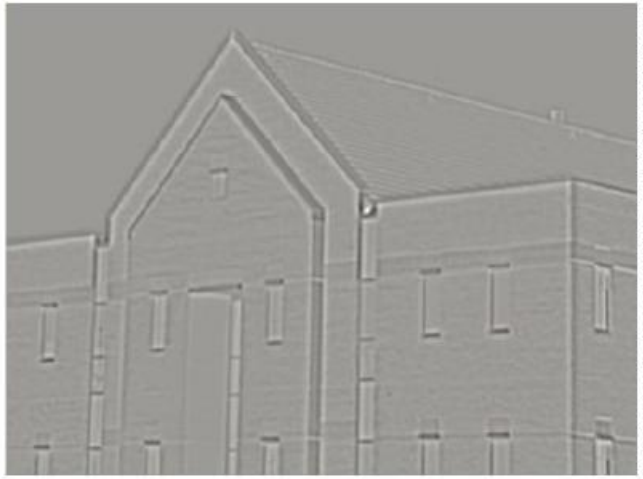
- Window size $n \geq 6\sigma$ and n is an odd number

Slide credit: Yan Tong

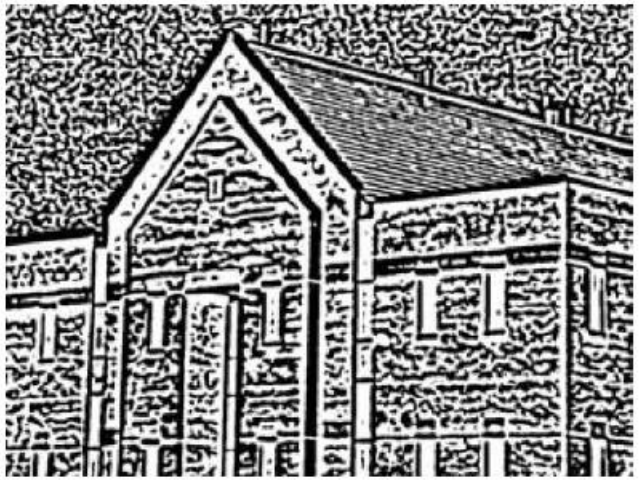
An Example – Edges are 1 Pixel Thick

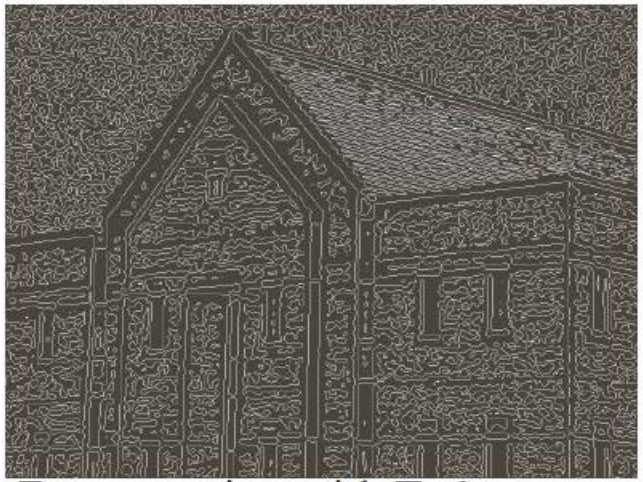
Original

LoG filtering



LoG filtering with T=0





Zero-crossing with T=0

Zero-crossing with T=4%max

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

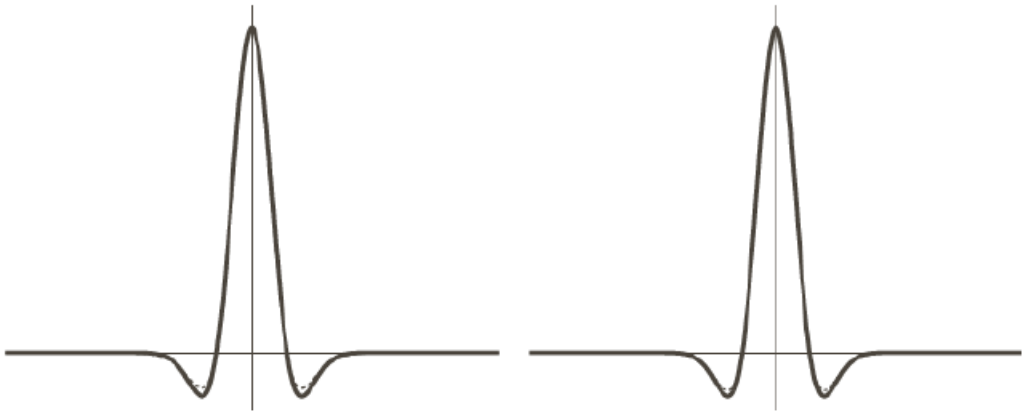
Slide credit: Yan Tong

Approximate LoG by DoG

LoG needs multiple filters for scale variations

⟹ Difference of Gaussian (DoG)

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



a b

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

Slide credit: Yan Tong

Canny Edge Detector

The general goal:

1. Low error rate: all edges should be found and there should be no false response

2. Edge points should be well localized: the edges located must be as close as possible to the true edges

3. Single edge point response: the detector should return only one point for each true edge point

Canny edge detector: expressing these three criteria mathematically and then find optimal solutions

Slide credit: Yan Tong

Canny Edge Detector

Gaussian smoothing + 1st order derivative

→ **optimal step edge detector**

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

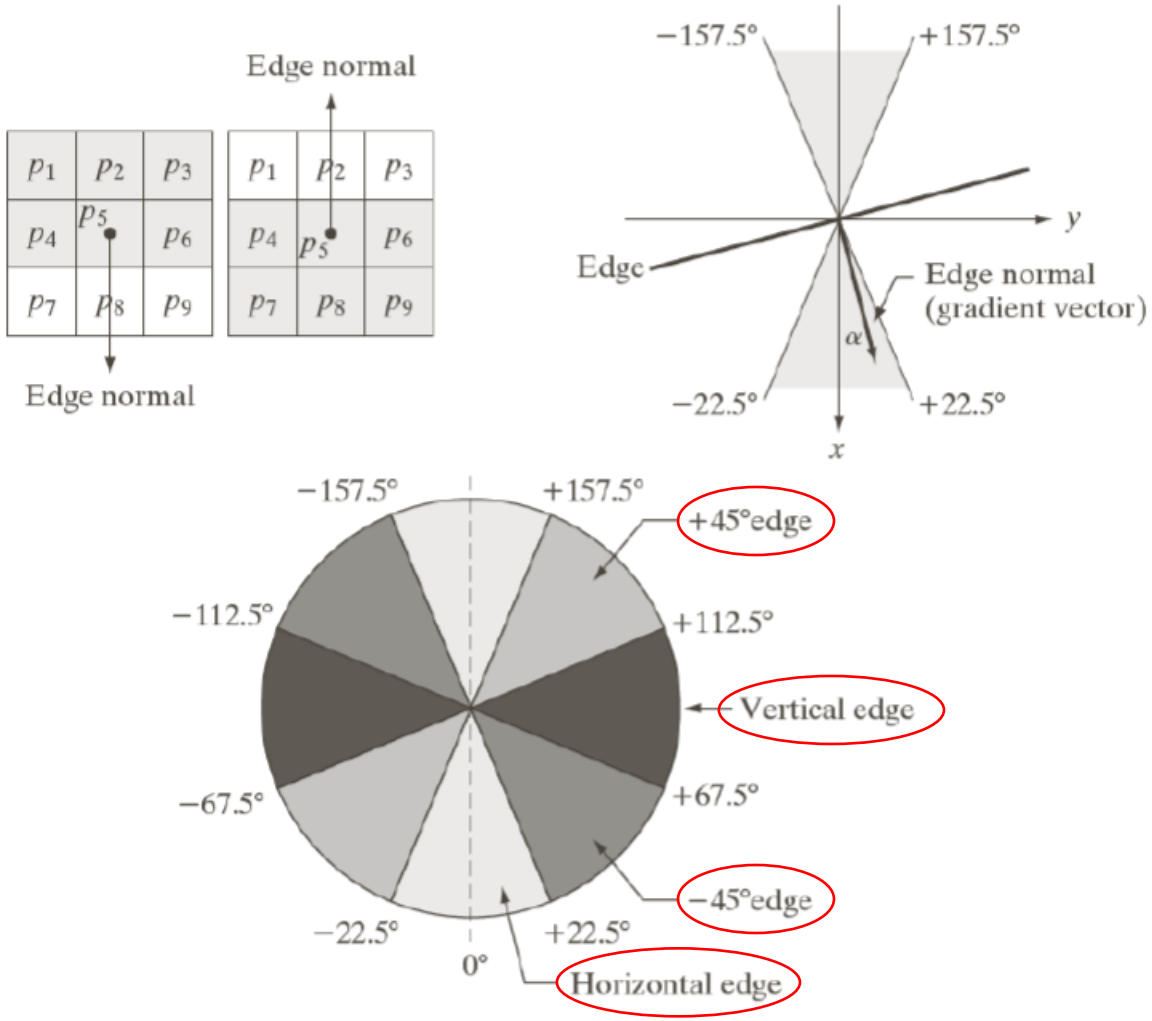
$$g_x = \partial f_s / \partial x \qquad g_y = \partial f_s / \partial y$$

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

Thick edge —Nonmaxima suppression→ **Single edge**

Slide credit: Yan Tong

Quantize the Edge Direction



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.

Slide credit: Yan Tong

Canny Detector -- Algorithm

1. **Smooth the input image with a Gaussian filter**
2. **Compute the gradient magnitude and angle images**
3. **Apply nonmaximum suppression on the gradient magnitude image**
   1. At (x, y), find the quantized edge normal d_k
   2. If the value $M(x, y)$ is less than at least one of its two neighbors along d_k, let $g_N(x, y) = 0$; otherwise $g_N(x, y) = M(x, y)$

Let d_1, d_2, d_3,and d_4 denote the four basic edge directions

+ve, -ve side

4. **Reduce false edge: double thresholding and connectivity analysis to detect and link edges**
   1. High-threshold → strong edge pixels →valid edge pixels
   2. Low-threshold → weak edge pixels → valid only when connected to strong edge pixels

If we set the threshold too low, there will still be some false edges (called *false positives*). If the threshold is set too high, then valid edge points will be eliminated (*false negatives*).

Slide credit: Yan Tong

Canny Detector -- Algorithm

Double Thresholding: Example

Uses two thresholds: a low threshold, *T_L* and a high threshold, *T_H*.

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

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

Two thresholded images

Modified

Initially, $g_{NH}(x,y)$ and $g_{NL}(x,y)$ are set to 0. After thresholding, $g_{NH}(x,y)$ will usually have fewer nonzero pixels than $g_{NL}(x,y)$, 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. We eliminate from $g_{NL}(x,y)$ all the nonzero pixels from $g_{NH}(x,y)$ by letting

$$g_{NL}(x,y) = g_{NL}(x,y) - g_{NH}(x,y) \qquad (10\text{-}41)$$

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.

Canny Detector -- Algorithm

Double Thresholding: Example

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 Detector -- Algorithm

Double Thresholding

We used two additional images, $g_{NH}(x, y)$ and $g_{NL}(x, y)$ to simplify the discussion. In practice, hysteresis thresholding can be implemented directly during nonmaxima suppression, and thresholding can be implemented directly on $g_N(x, y)$ by forming a list of strong pixels and the weak pixels connected to them.

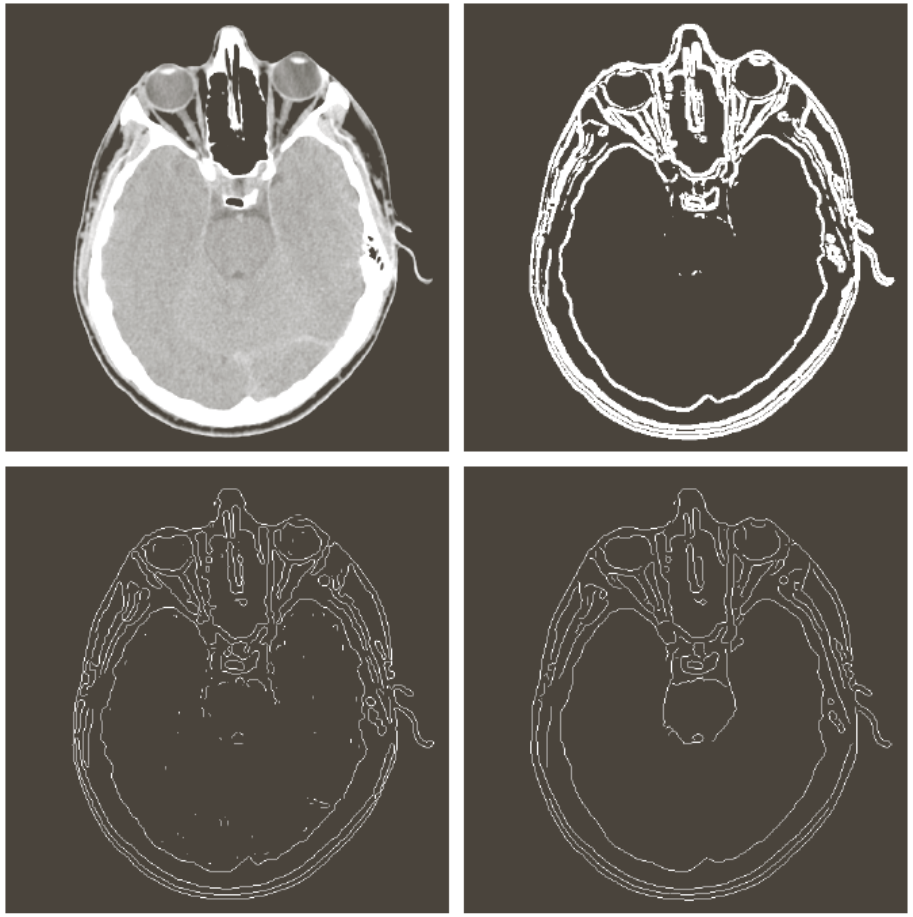
An Example

a b
c d

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

Slide credit: Yan Tong

An Example



a b
c d

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

Slide credit: Yan Tong

Edge Linking and Boundary Detection

All edge detection algorithms can only detect fragments of boundaries, due to image noise, non-uniform illuminations, or other effects

Edge linking: link edges into longer meaningful edges or a full region boundaries

Three classic methods:

- Local processing
- Regional Processing
- Hough transform **Global Processing**

Slide credit: Yan Tong

Edge Linking – Local Processing

Link the edge points with similar properties:

- Strength
- Direction

Two edge pixels are linked if

Let S_{xy} denote the set of coordinates of a neighborhood centered at point (x, y) in an image. An edge pixel with coordinates (s,t) in S_{xy} is similar in *magnitude* to the pixel at (x, y) if

$$| M(s,t) - M(x,y) | \leq E$$ where E is a positive threshold.

Similarly, $$| \alpha(s,t) - \alpha(x,y) | \leq A$$ where A is a positive angle threshold

Slide credit: Yan Tong

Edge Linking – Local Processing

Simplified Procedure

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, $g(x,y)$, whose value at any point (x,y) is given by:

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

   where T_M is a threshold, A is a specified angle direction, and $\pm T_A$ defines a "band" of acceptable directions about A.
3. Scan the rows of g and fill (set to 1) all gaps (sets of 0's) in each row that do not exceed a specified length, L. Note that, by definition, a gap is bounded at both ends by one or more 1's. The rows are processed individually, with no "memory" kept between them.
4. To detect gaps in any other direction, θ, rotate g by this angle and apply the horizontal scanning procedure in Step 3. Rotate the result back by $-\theta$.

Edge Linking – Local Processing Example

Objective: find rectangular region for license plate recognition.

a b c
d e f

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.)

Slide credit: Yan Tong

Edge Linking – Hough Transform

Global Processing

A basic idea: Given *n* points

1. **Find $n(n-1)/2$ lines between each pair of points** O(n^2)
2. **Find all subset of points that are close to particular lines. This needs $n(n-1)n/2$ comparisons** O(n^3)

This is computationally expensive!

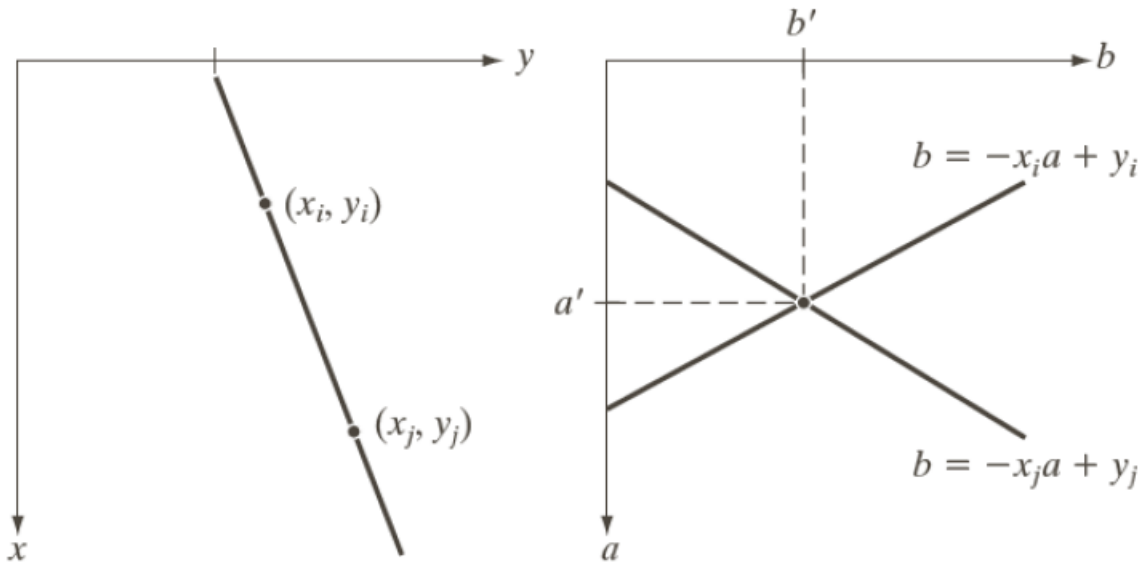
Hough transform

Slide credit: Yan Tong

Hough Transform

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

X-Y plane ⟺ Parameter plane



a b

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

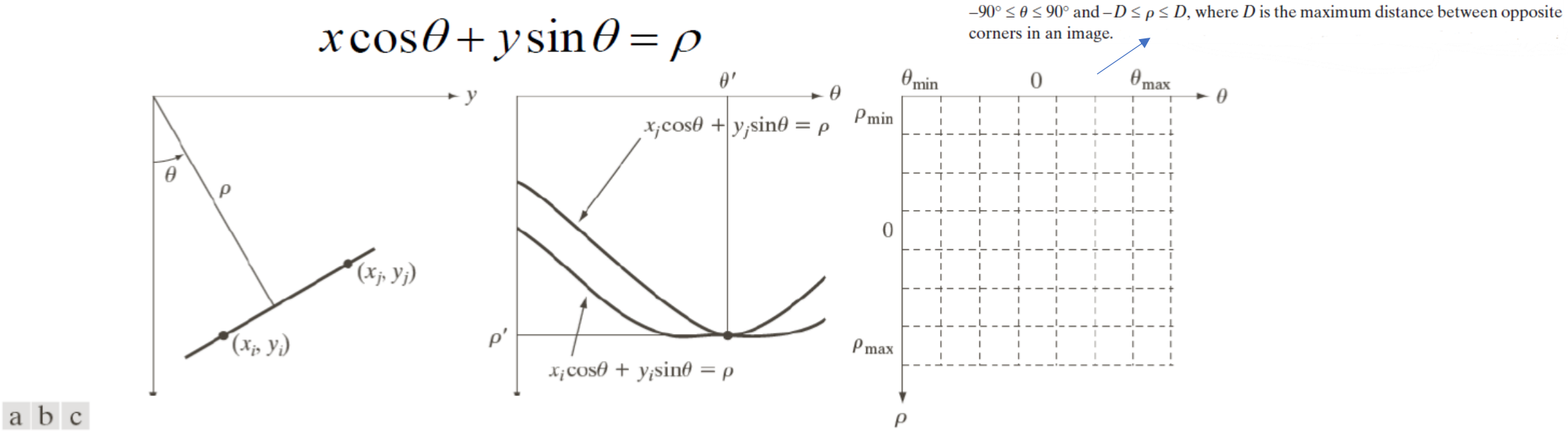
Problem of slope-intercept form: the slope a approaches infinity for vertical lines

Slide credit: Yan Tong

Hough Transform

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

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

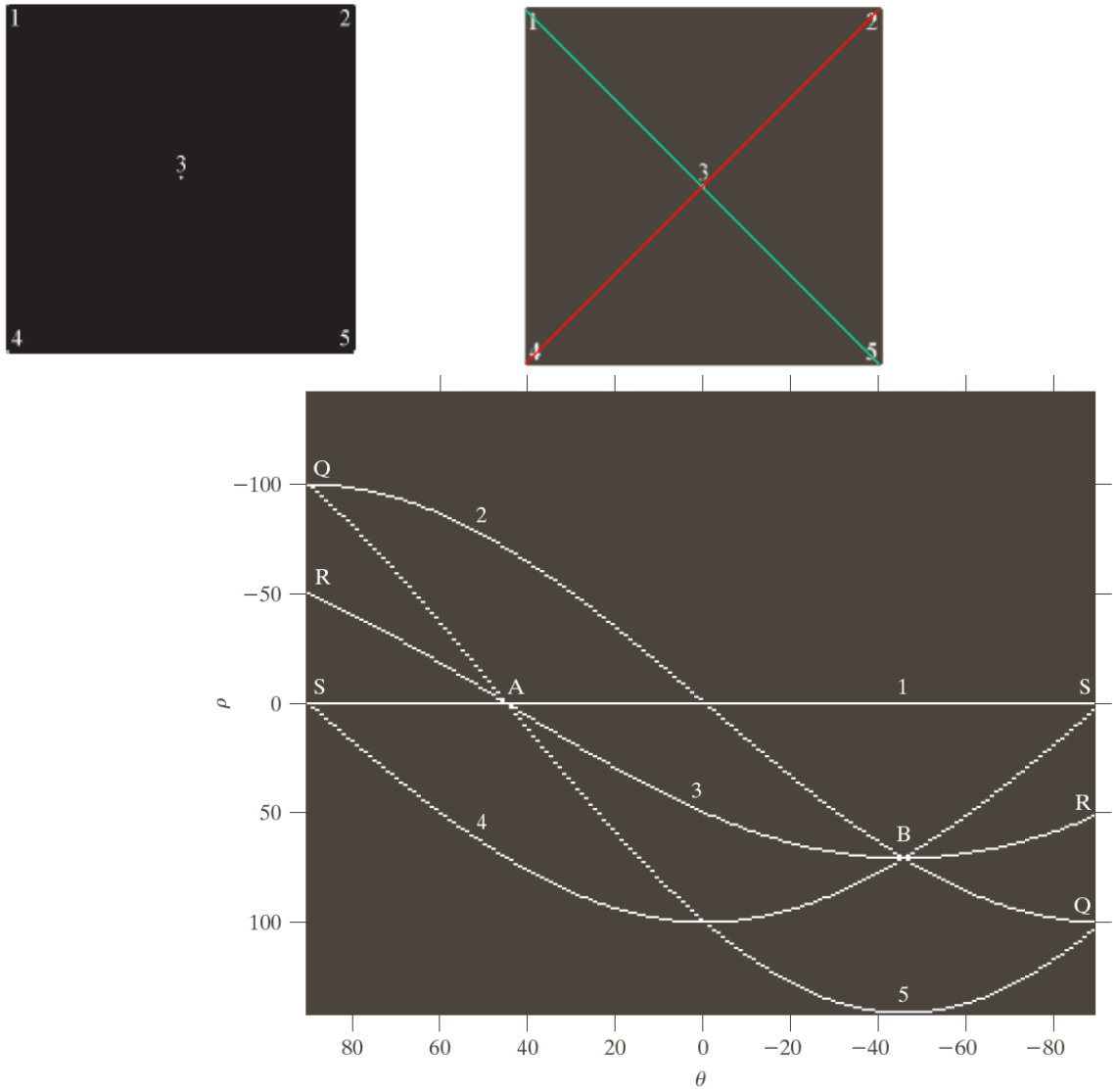


a b c

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.

Slide credit: Yan Tong

Hough Transform: Toy example



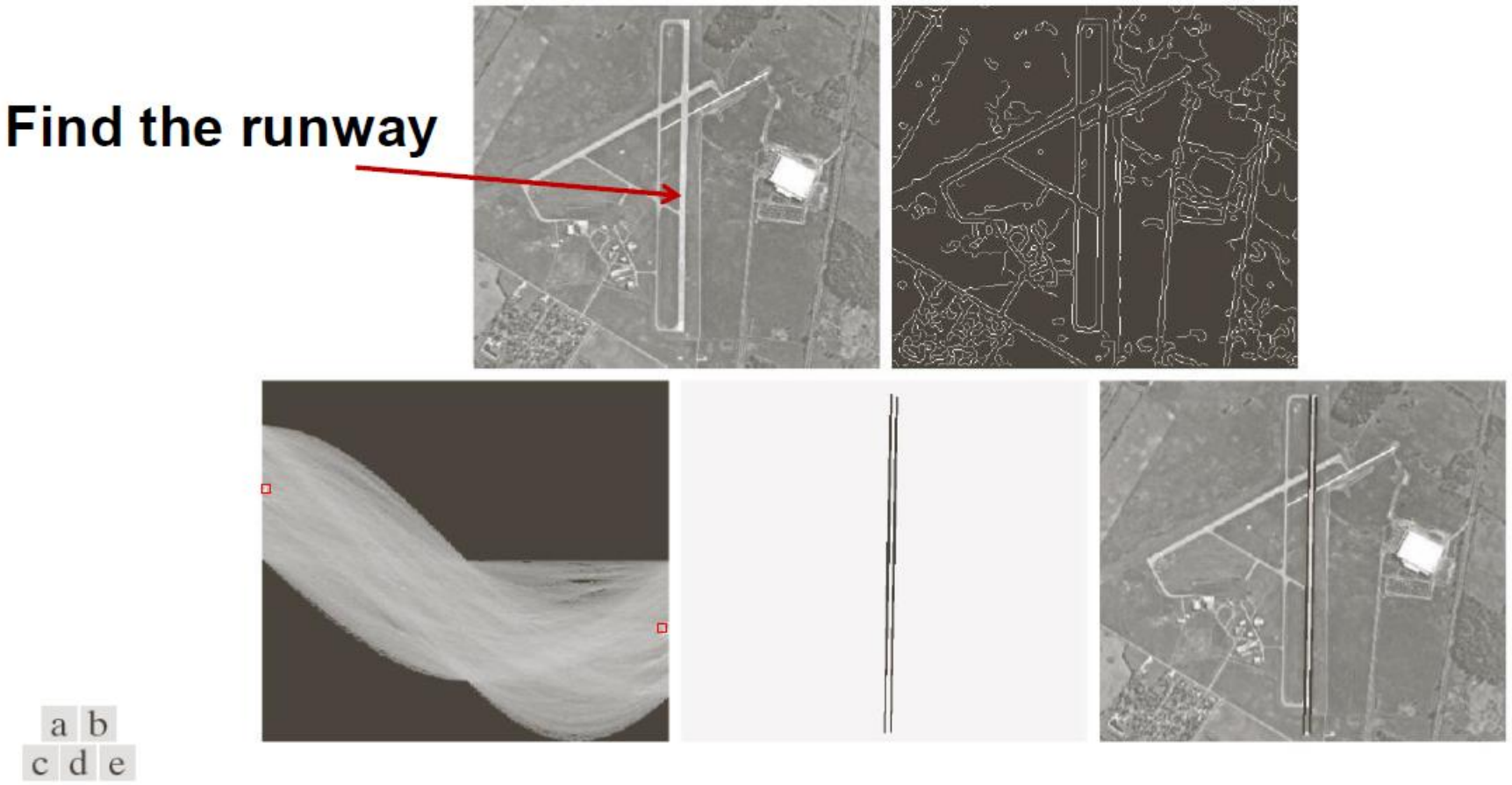
a
b

FIGURE 10.33
(a) Image of size 101 × 101 pixels, containing five points.
(b) Corresponding parameter space. (The points in (a) were enlarged to make them easier to see.)

A: green line
B: red line

Slide credit: Yan Tong

Hough Transform: Real example



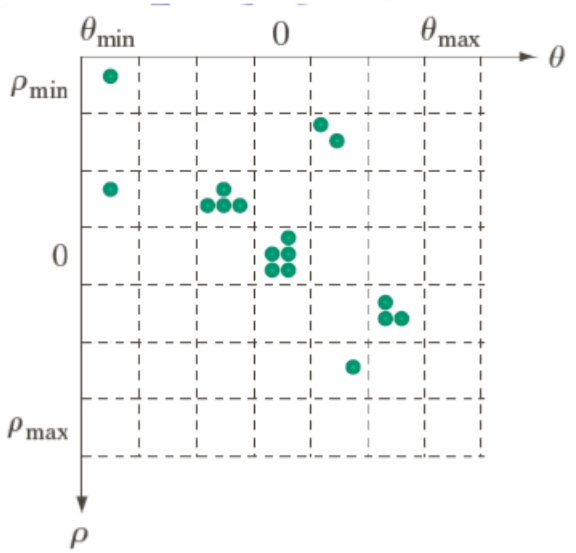
a b
c d e

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.

Slide credit: Yan Tong

Hough Transform: Real example

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



$$-90^0 \le \theta \le 90^0, -D \le \rho \le D$$

Slide credit: Yan Tong

Notes

Thresholding is required to get edge pixels and realize edge-based segmentation

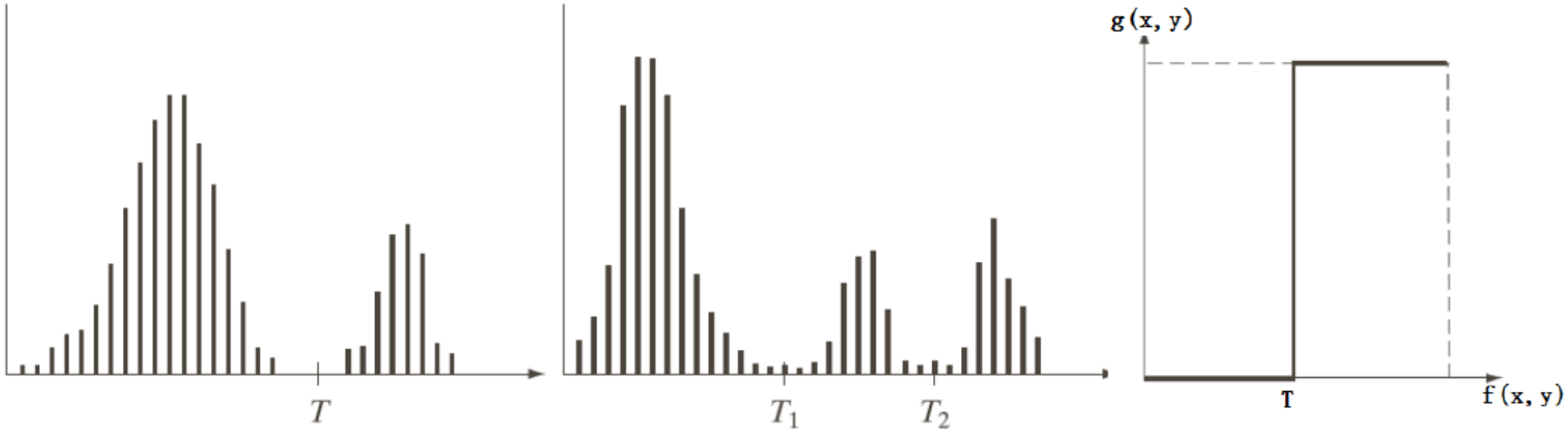
Boundary detection (edge linking) is still a hot research topic in image processing and computer vision

Incorporate domain knowledge:

- Psychology rules on the boundary: smooth, convex, symmetry, closed, complete, etc
- Template shape information: hand, stomach, lip, etc
- Appearance information: region-based texture, intensity/color, etc.

Slide credit: Yan Tong

Intensity Thresholding



Object/background segmentation:

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

- A constant T - global thresholding
- A variable T - local/regional thresholding; adaptive thresholding
- Multiple T - multiple thresholding

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

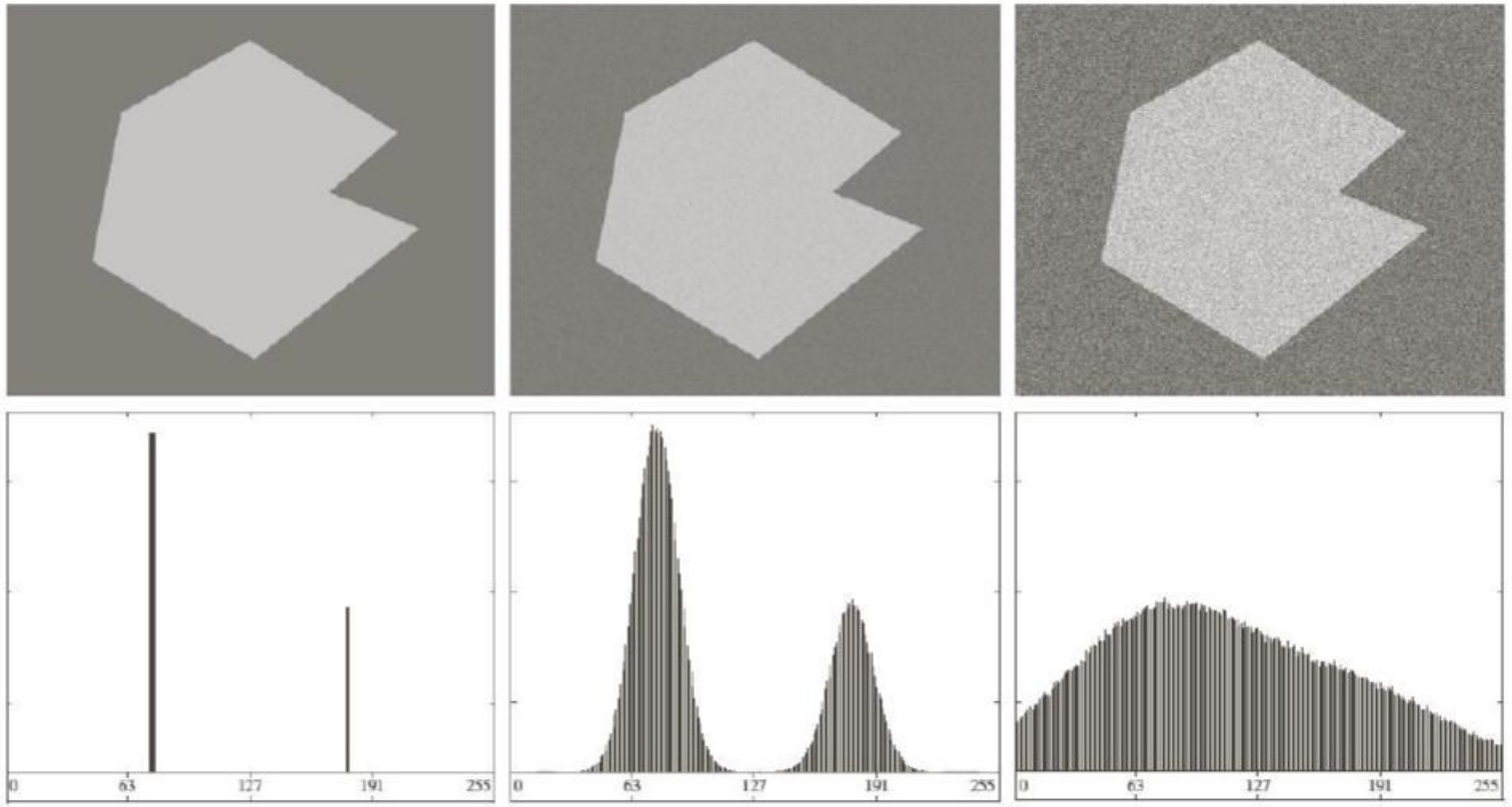
Slide credit: Yan Tong

Key Factors Affect Thresholding

- Separation between peaks
- Noise level
- Relative sizes of objects and background
- Uniformity of the illumination source
- Uniformity of the reflectance of the image

Slide credit: Yan Tong

The Role of Noise in Image Thresholding

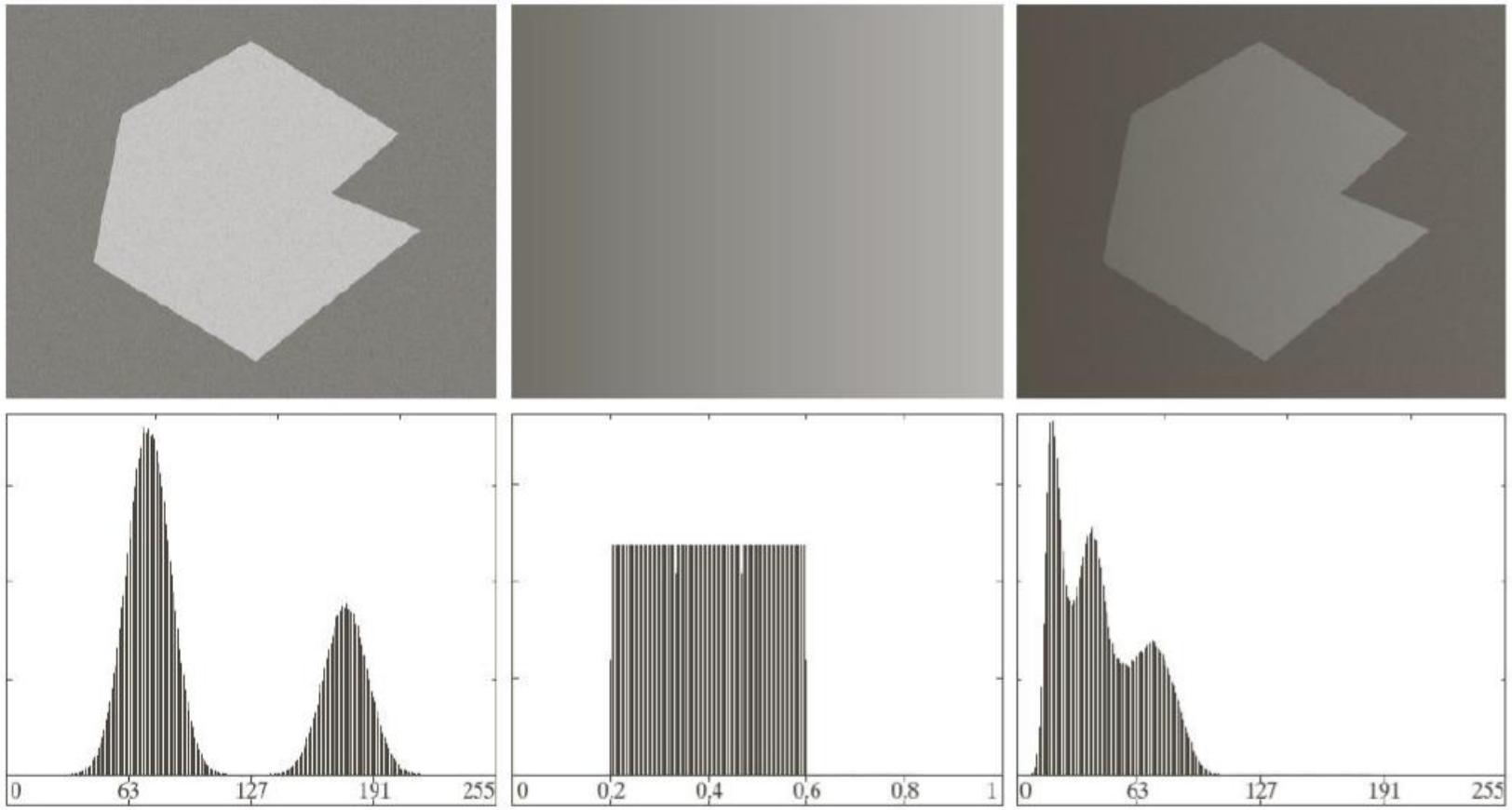


a b c
d e f

FIGURE 10.36 (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)–(f) Corresponding histograms.

Slide credit: Yan Tong

The Role of Illumination in Thresholding



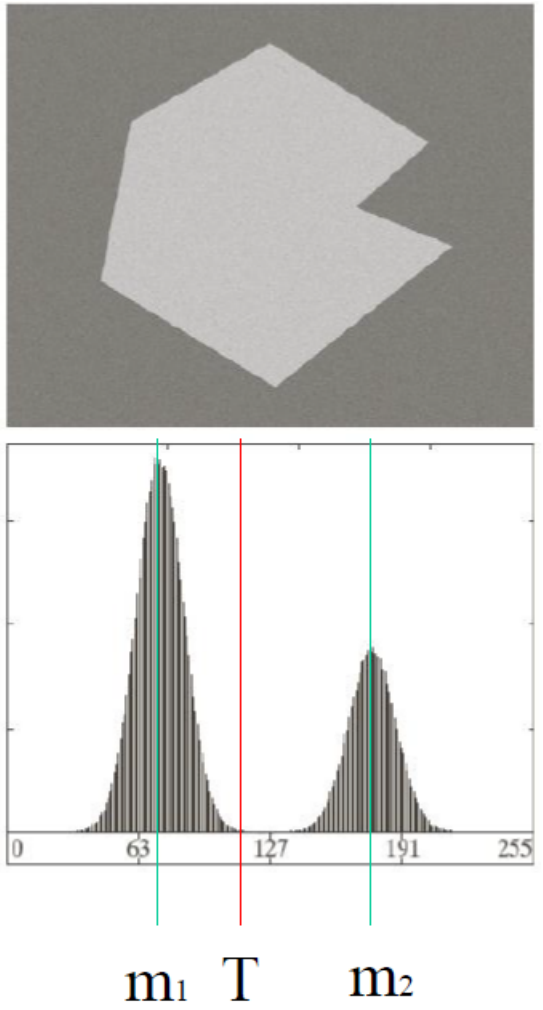
a b c
d e f

FIGURE 10.37 (a) Noisy image. (b) Intensity ramp in the range [0.2, 0.6]. (c) Product of (a) and (b). (d)–(f) Corresponding histograms.

Slide credit: Yan Tong

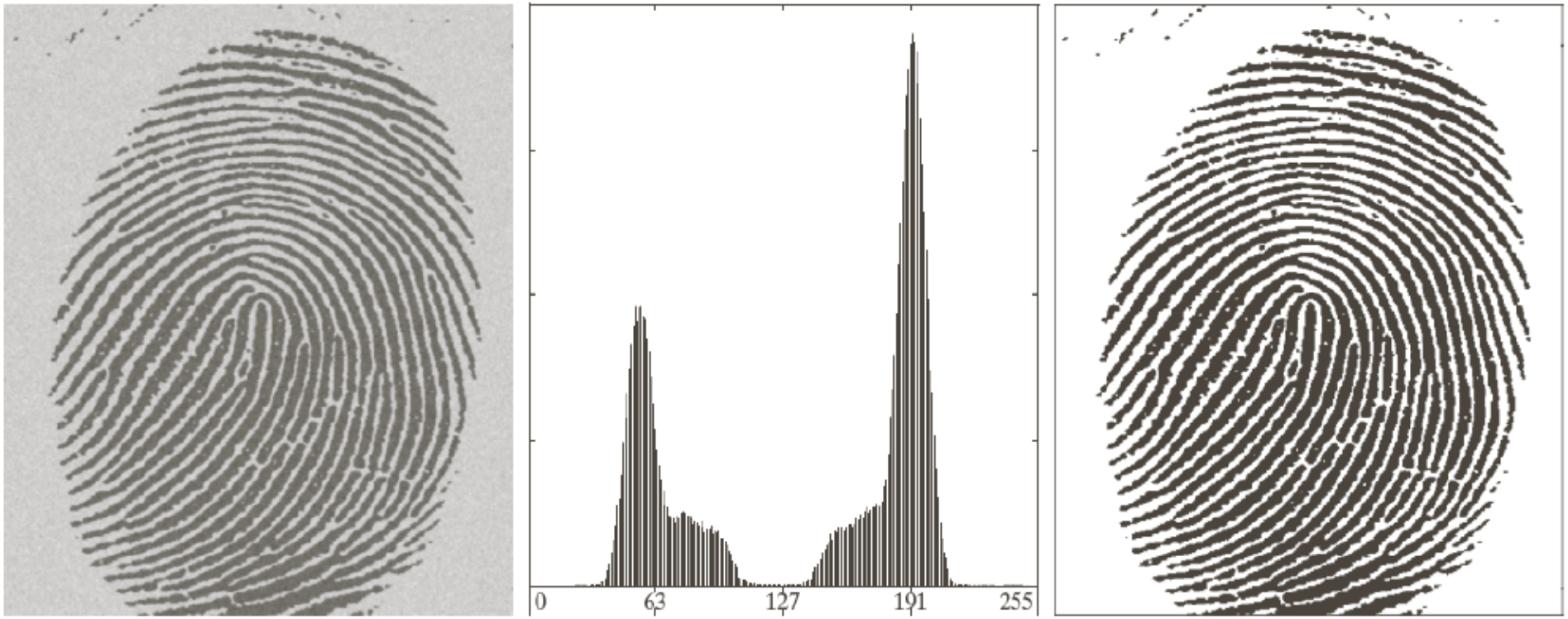
How to Pick the Threshold

1. **Select an initial estimate for the global threshold, *T*.**
2. **Segment the image using *T* by producing two groups of pixels**
3. **Compute the mean of these two groups of pixels, say m_1 and m_2.**
4. **Update the threshold T=(m_1 +m_2)/2**
5. **Repeat Steps 2 through 4 until convergence**



Slide credit: Yan Tong

An Example



a b c

FIGURE 10.38 (a) Noisy fingerprint. (b) Histogram. (c) Segmented result using a global threshold (the border was added for clarity). (Original courtesy of the National Institute of Standards and Technology.)

Slide credit: Yan Tong

Optimal global thresholding using Otsu's method

- **Otsu's method (1979) maximizes between-class variance**
- **Based entirely on computations performed on histogram (1-D) of image**
- **Normalized histogram:** $p_i = \frac{n_i}{MN},\ i = 0, \ldots, L-1$, **with** $\sum_{i=0}^{L-1} p_i = 1,\ p_i \geq 0$
- **Select threshold** $T(k)$ **to segment image** $\longrightarrow$ **Class** C_1 **(values** $[0, k]$**)**
  Class C_2 **(values** $[k+1, L-1]$**)**

$\Rightarrow$ **Prob of pixel assigned to** C_1 **(ie of** C_1 **occuring):** $P_1(k) = \sum_{i=0}^{k} p_i$

$\Rightarrow$ **Prob of pixel assigned to** C_2 **(ie of** C_2 **occuring):** $P_2(k) = \sum_{i=k+1}^{L-1} p_i = 1 - P_1(k)$

Slide credit: Milton Maritz

Optimal global thresholding using Otsu's method

$\Rightarrow$ **Mean value of pixels assigned to C_1:**

$$
\begin{aligned}
m_1(k) &= \sum_{i=0}^{k} i\, P(i/C_1) \\
&= \sum_{i=0}^{k} i\ \overbrace{P(C_1/i)}^{=1}\ \underbrace{P(i)}_{=p_i}\ /\ \overbrace{P(C_1)}^{=P_1(k)} \qquad \textbf{(Bayes' formula)} \\
&= \frac{1}{P_1(k)} \sum_{i=0}^{k} i\, p_i
\end{aligned}
$$

$\Rightarrow$ **Mean value of pixels assigned to C_2:**

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

Slide credit: Milton Maritz

Optimal global thresholding using Otsu's method

$\Rightarrow$ **Mean intensity up to level** k**:** $m(k) = \sum_{i=0}^{k} i\, p_i$

- **Global mean:** $m_G = \sum_{i=0}^{L-1} i\, p_i$

  $\Rightarrow$ $\boxed{P_1 m_1 + P_2 m_2 = m_G}$ **and** $\boxed{P_1 + P_2 = 1}$ **(**k**s temporarily omitted)**

- **"Goodness" of threshold at level** k **evaluated by dimensionless metric:**

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

$$\sigma_G^2 = \sum_{i=0}^{L-1} (i - m_G)^2\, p_i \quad \textbf{(Global variance)}$$

$$\sigma_B^2 = P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2 \quad \textbf{(Between-class variance)}$$

Also: $\sigma_B^2 = P_1 P_2 (m_1 - m_2)^2 = \dfrac{(m_G P_1 - m)^2}{P_1(1 - P_1)}$ $\leftarrow$ **most efficient**

Slide credit: Milton Maritz

Optimal global thresholding using Otsu's method

Reintroduce $k \rightsquigarrow$ final results:

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

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

Optimum threshold is k^* that maximizes $\sigma_B^2(k)$:

$$\sigma_B^2(k^*) = \max_{k \in [0, L-1]} \sigma_B^2(k)$$

Segmentation is as follows:

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

The metric $\eta(k^*)$ can be used to obtain a quantative estimate of the separability of the classes and has values in the range:

$$\eta(k^*) \in [0, 1]$$

Slide credit: Milton Maritz

Optimal global thresholding using Otsu's method

Summary of Otsu's algorithm

(1) Compute normalized histogram of the image, $p_i = \frac{n_i}{MN},\ i = 0, \ldots, L-1$

(2) Compute cumulative sums, $P_1(k) = \sum_{i=0}^{k} p_i,\ k = 0, \ldots, L-1$

(3) Compute cumulative means, $m(k) = \sum_{i=0}^{k} i\, p_i,\ k = 0, \ldots, L-1$

(4) Compute global intensity mean, $m_G = \sum_{i=0}^{L-1} i\, p_i$

(5) Compute between-class variance, $\sigma_B^2(k) = \frac{[m_G P_1(k) - m(k)]^2}{P_1(k)[1 - P_1(k)]},\ k = 0, ., L-1$

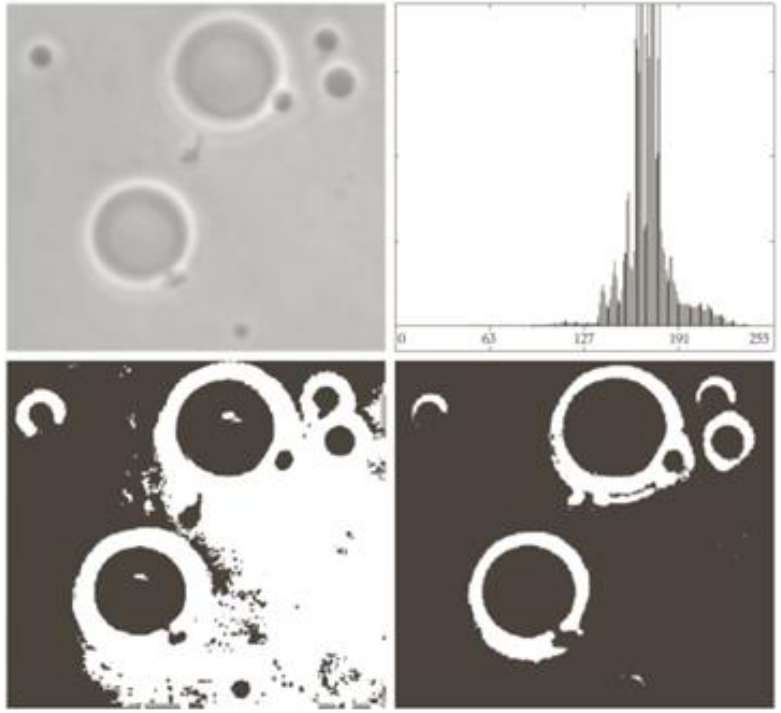
(6) Obtain the Otsu threshold, k^*, that is the value of k for which $\sigma_B^2(k^*)$ is a maximum – if this maximum is not unique, obtain k^* by avaraging the values of k that correspond to the various maxima detected

(7) Obtain the separability measure $\eta(k^*) = \frac{\sigma_B^2(k^*)}{\sigma^2}$

Slide credit: Milton Maritz

Optimal global thresholding using Otsu's method

Optimal global thresholding using Otsu's method



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.)

For the above image... Threshold found by basic algorithm: $T = 161$;

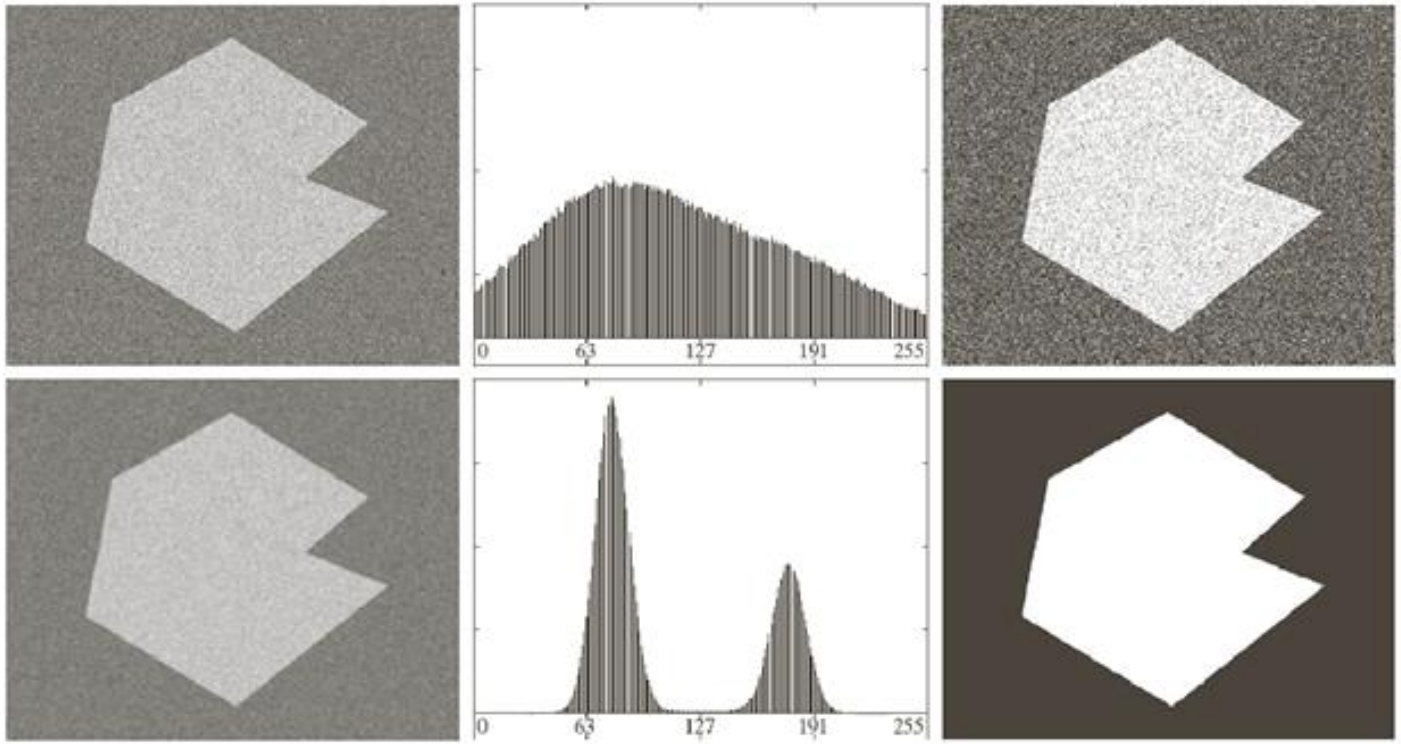
Threshold found by Otsu's algorithm: $T = 181$ **(Sep measure:** $\eta = 0.467$**)**

For fingerprint image... basic and Otsu's algorithm: $T = 125$ $(\eta = 0.944)$

Slide credit: Milton Maritz

Optimal global thresholding using Otsu's method

Using image smoothing to improve global thresholding



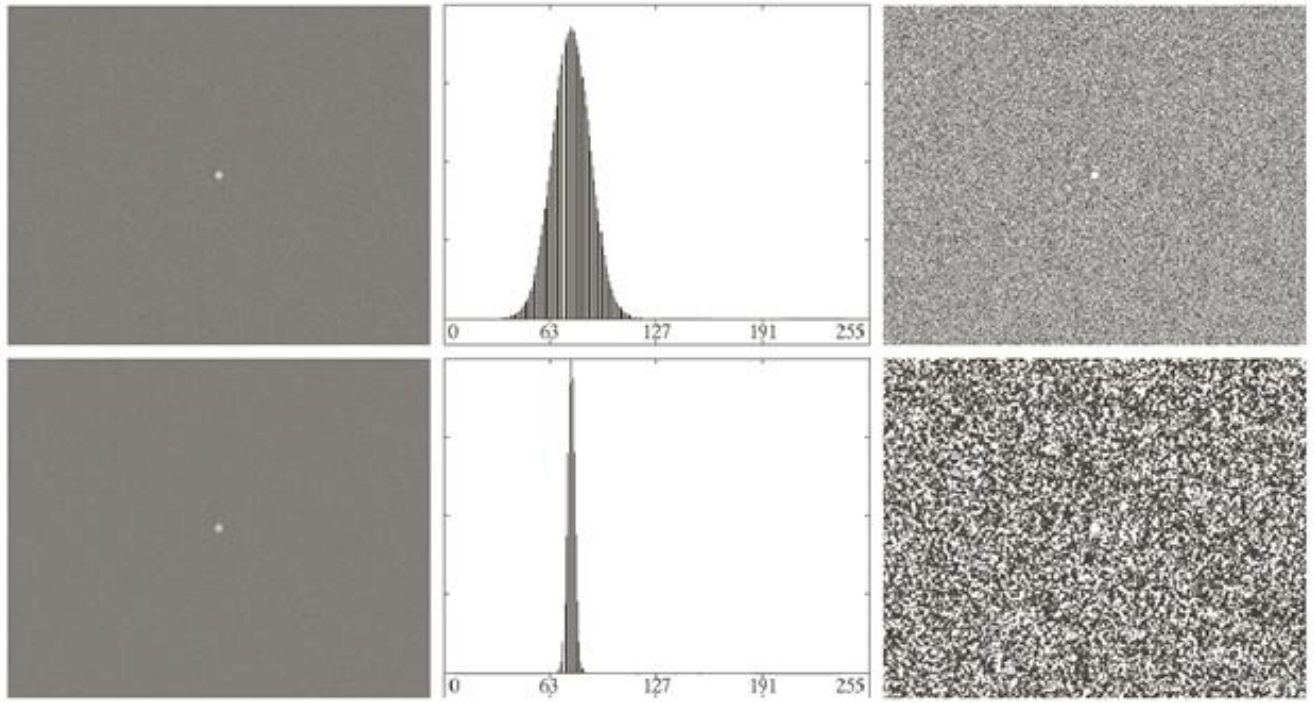
a b c
d e f

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.

Slide credit: Milton Maritz

Optimal global thresholding using Otsu's method

Small object ⇒ thresholding fails even after smoothing



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.

Slide credit: Milton Maritz

Multiple Thresholds

Otsu's method extended to K classes, $C_1, C_2, \ldots, C_K$

Between-class variance:

$$\sigma_B^2 = \sum_{k=1}^{K} P_k(m_k - m_G)^2, \quad \textbf{where} \quad P_k = \sum_{i \in C_k} p_i \quad \textbf{and} \quad m_k = \sum_{i \in C_k} i p_i$$

The K classes are separated by $K-1$ thresholds whose values $k_1^*, k_2^*, \ldots k_{K-1}^*$ maximize

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

Otsu's method extended to three classes, C_1, C_2, C_3

Between-class variance:

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

$$P_1 = \sum_{i=0}^{k_1} p_i, \quad P_2 = \sum_{i=k_1+1}^{k_2} p_i, \quad P_3 = \sum_{i=k_2+1}^{L-1} p_i$$

Slide credit: Milton Maritz

Multiple Thresholds

$$m_1 = \frac{1}{P_1}\sum_{i=0}^{k_1} ip_i, \quad m_2 = \frac{1}{P_2}\sum_{i=k_1+1}^{k_2} ip_i, \quad m_3 = \frac{1}{P_3}\sum_{i=k_2+1}^{L-1} ip_i$$

The following relationships also hold:

$$P_1 m_1 + P_2 m_2 + P_3 m_3 = m_G, \qquad P_1 + P_2 + P_3 = 1$$

The three classes are separated by two thresholds whose values k_1^* and k_2^* maximize

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

Algorithm

(1) Let $k_1 = 1$

(2) Increment k_2 through all its values greater than k_1 and less than $L - 1$

(3) Increment k_1 to its next value and increment k_2 through all its values greater than k_1 and less than $L - 1$

(4) Repeat until $k_1 = L - 3$

This results in a 2-D array $\sigma_B^2(k_1, k_2)$, after which k_1^* and k_2^* that correspond to the maximum value in the array, are selected

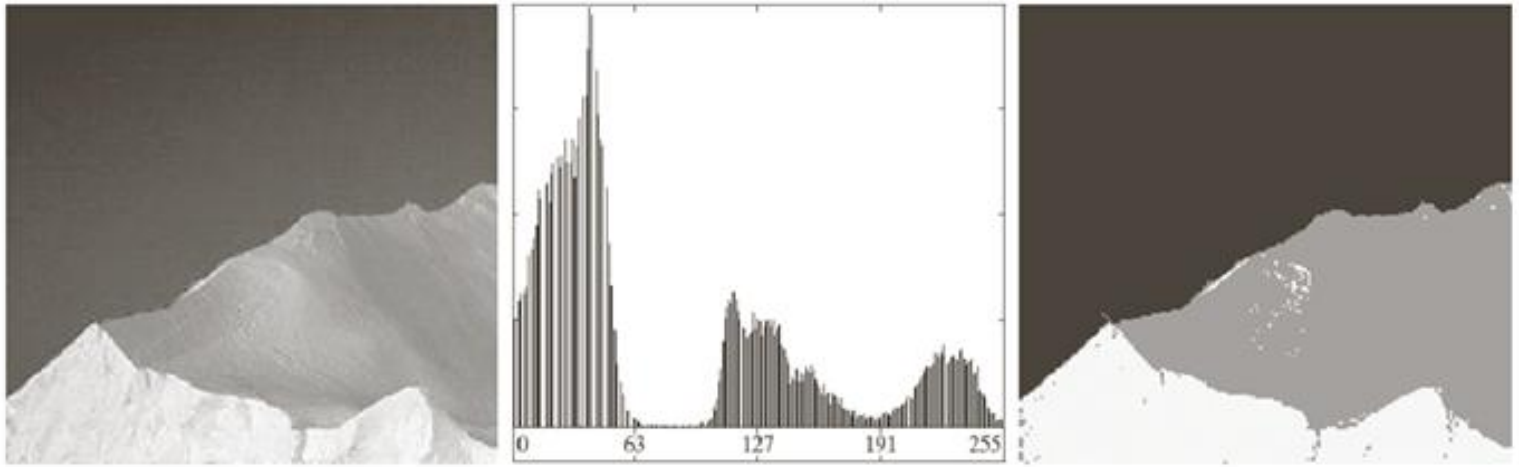
Slide credit: Milton Maritz

Multiple Thresholds

Segmentation is as follows: $g(x,y) = \begin{cases} a, & \text{if } f(x,y) \le k_1^* \\ b, & \text{if } k_1^* < f(x,y) \le k_2^* \\ c, & \text{if } f(x,y) > k_2^* \end{cases}$

Separability measure: $\eta(k_1^*, k_2^*) = \dfrac{\sigma_B^2(k_1^*, k_2^*)}{\sigma_G^2}$

Example 10.19: Multiple global thresholding



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.)

Slide credit: Milton Maritz

Variable Thresholding

Examples: **(1)** $T_{xy} = a\,\sigma_{xy} + b\,m_{xy}$ **(2)** $T_{xy} = a\,\sigma_{xy} + b\,m_G$

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

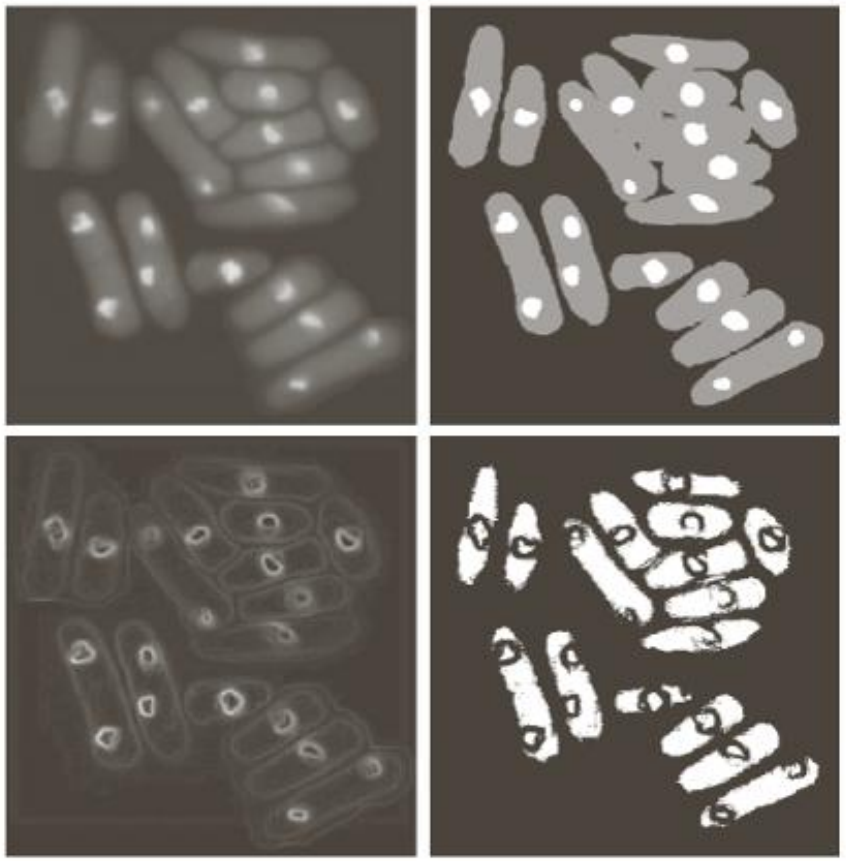
$$g(x,y) = \begin{cases} 1, & \textbf{if } Q\textbf{(local parameters) is true} \\ 0, & \textbf{if } Q\textbf{(local parameters) is false} \end{cases}$$

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

Slide credit: Milton Maritz

Variable Thresholding

Variable thresholding based on local image properties



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.

Slide credit: Milton Maritz

Variable Thresholding

Variable thresholding

- **Moving averages**

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

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

a b c

FIGURE 10.49 (a) Text image corrupted by spot shading. (b) Result of global thresholding using Otsu's method. (c) Result of local thresholding using moving averages.

(10.3.8 Multi-variable thresholding: See 6.7.2)

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.

Slide credit: Milton Maritz

Region-Based Segmentation

- **Region growing**
- **Region splitting and merging**

Slide credit: Yan Tong

Region Growing Algorithm

-A procedure that groups pixels or subregions into larger regions based on predefined criteria for growth

-Start with a set of "seed" points and grow regions by appending neighboring pixels that satisfy the given criteria

- Connectivity
- Stopping rules
  - Local criteria: intensity values, textures, color
  - Prior knowledge: size and shape of the object

Slide credit: Yan Tong

Region Growing Algorithm

Algorithm (based on 8-connectivity):

- $f(x, y) \equiv$ **input image array;**
- $S(x, y) \equiv$ **seed array (1s at locations of seed points and 0s elsewhere);**
- $Q \equiv$ **predicate to be applied at each location (x, y)**

(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 labelled 0.

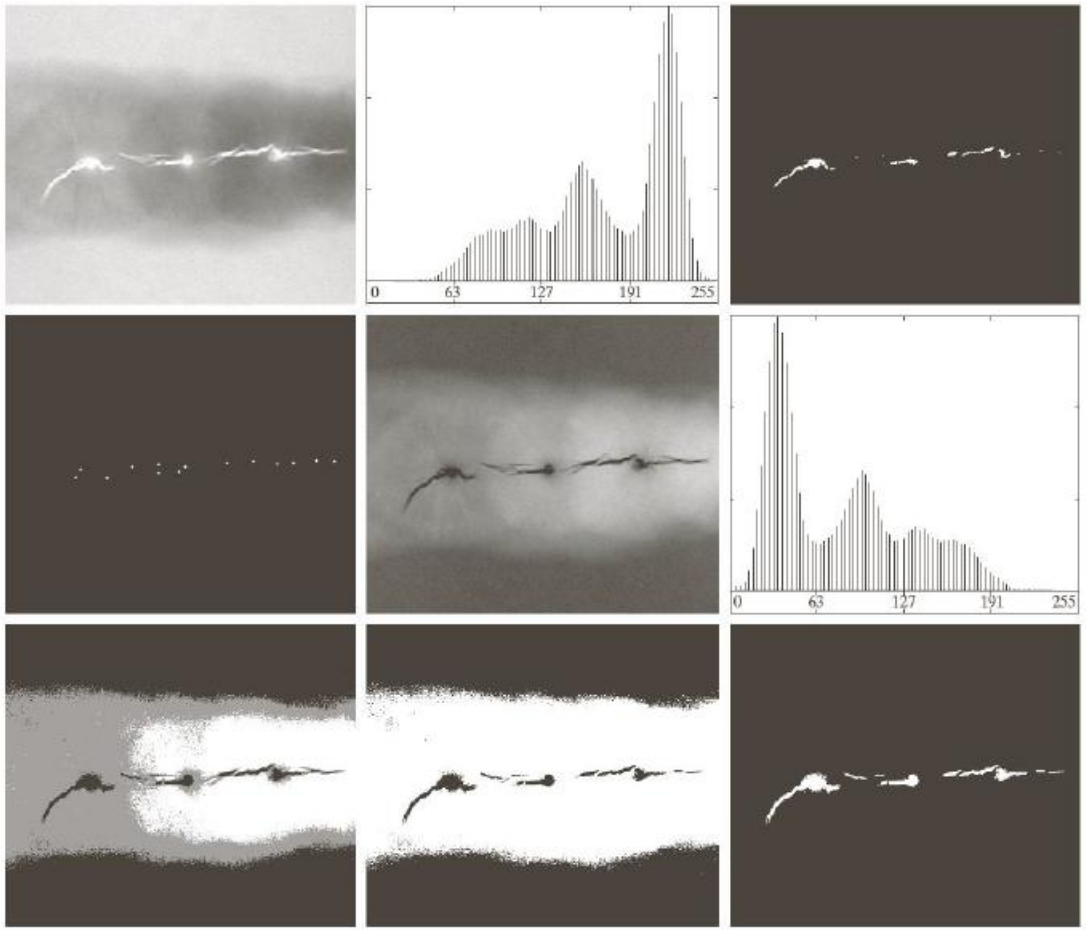
(2) Form and image f_Q such that, at a pair of coordinates (x, y), let $f_Q(x, y) = 1$ if the input image satisfies the given predicate, Q, at those coordinates; otherwise let $f_Q(x, y) = 0$.

(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 constitutes the segmented image obtained by region growing.

Slide credit: Milton Maritz

An Example



a b c
d e f
g h i

FIGURE 10.51 (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 (a) and (c). (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.)

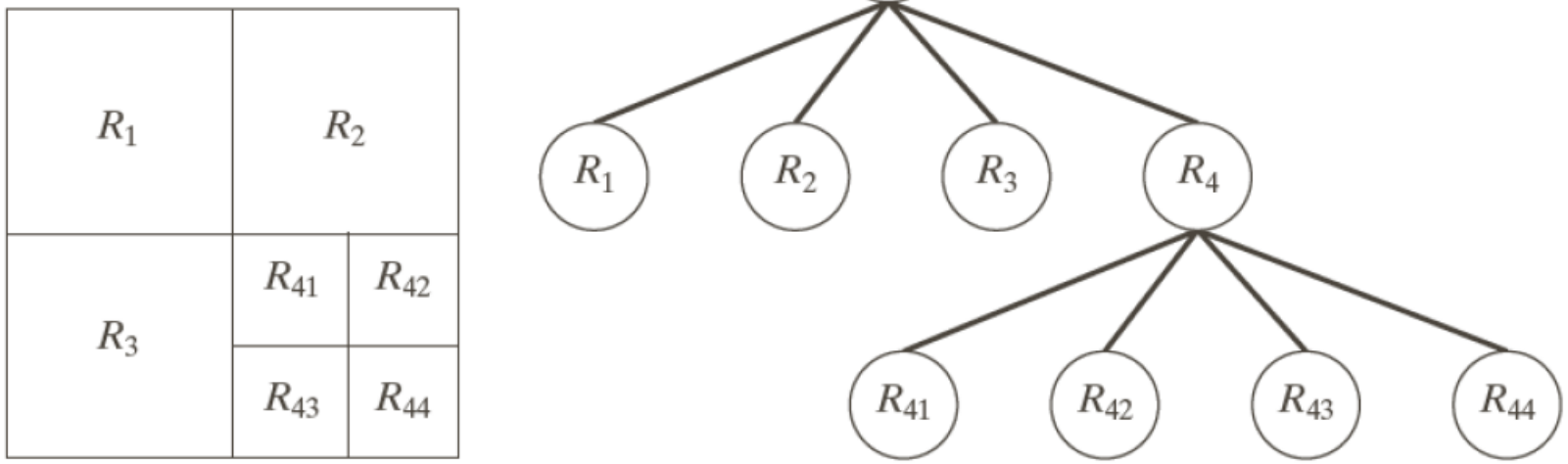
$$Q = \begin{cases} True & if\ |I_{seed} - I(x,y)| \leq T \\ False & otherwise \end{cases}$$

Slide credit: Yan Tong

Region-Splitting and Merging Algorithm

Step1: Keep splitting the region while $Q(R_i) = FALSE$ and $R_i > \min Size$

Step 2: Merge the subregions while $Q(R_i \cup R_j) = TRUE$



a b

FIGURE 10.52 (a) Partitioned image. (b) Corresponding quadtree. R represents the entire image region.

Slide credit: Yan Tong

Region-Splitting and Merging Algorithm

Subdivide an image initially into a set of arbitrary, disjoint regions and then merge and/or split the regions in an attempt to satisfy the necessary conditions

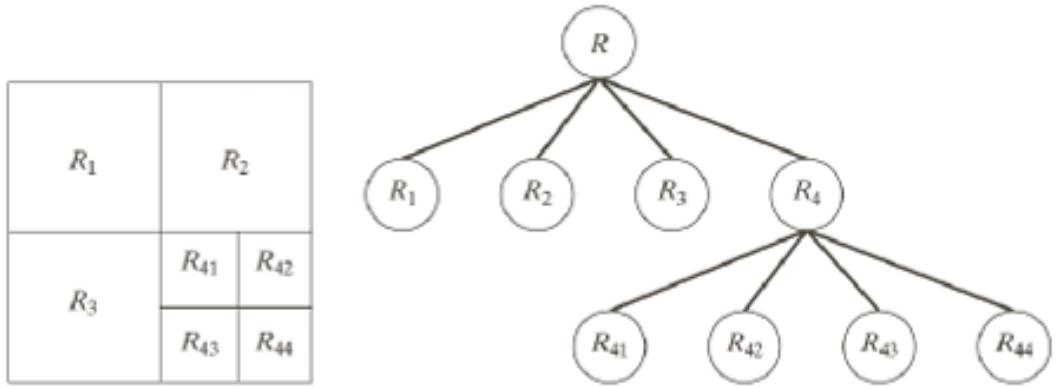
Let R represent entire image region and select a predicate Q

(1) Split into four disjoint quadrants any region R_i for which $Q(R_i) =$ FALSE

(2) Merge any adjacent regions R_j and R_k for which $Q(R_j \cup R_k) =$ TRUE

(3) Stop when no further merging or splitting is possible

Several variations of this theme are possible

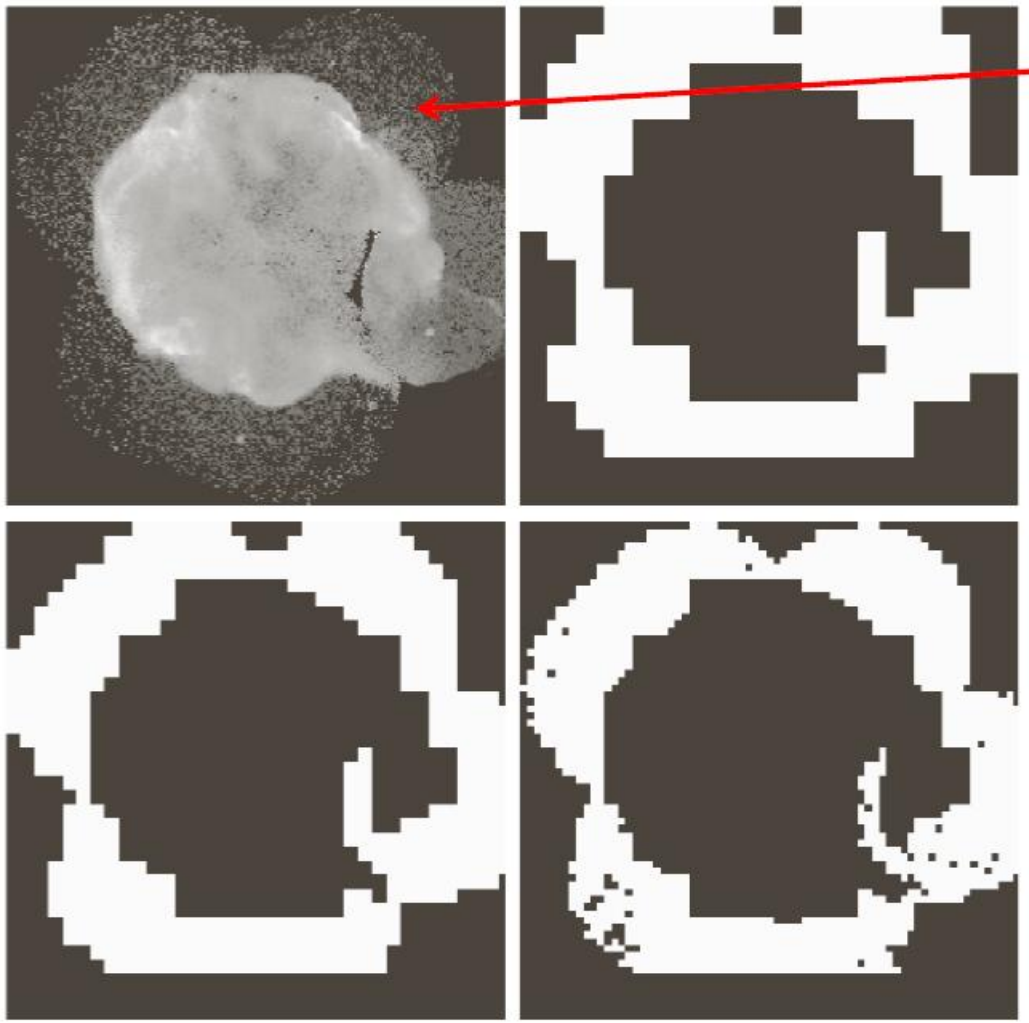


a b

FIGURE 10.52 (a) Partitioned image. (b) Corresponding quadtree. R represents the entire image region.

Slide credit: Milton Maritz

An Example



a b
c d

FIGURE 10.53 (a) Image of the Cygnus Loop supernova, taken in the X-ray band by NASA's Hubble Telescope. (b)–(d) Results of limiting the smallest allowed quadregion to sizes of 32×32, 16×16, and 8×8 pixels, respectively. (Original image courtesy of NASA.)

Extract the outer ring

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

where σ_R and m_R are the standard deviation and mean of the region being processed, and a and b are nonnegative constants.

Slide credit: Yan Tong

Advanced Approaches for Image Segmentation

Image segmentation is still a research problem that is being investigated by many researchers

General-purpose image segmentation is far from well solved

- Image segmentation by K-means clustering
- Image segmentation with Graphic Models (MRF, CRF, etc.)

Semantic Segmentation with Deep Learning

http://blog.qure.ai/notes/semantic-segmentation-deep-learning-review

Slide credit: Yan Tong

Image Segmentation: K-Means Clustering

1. ***Initialize the algorithm:*** Specify an initial set of means, $\mathbf{m}_i(1)$, $i = 1, 2, \ldots, k$.
2. ***Assign samples to clusters:*** Assign each sample to the cluster set whose mean is the closest (ties are resolved arbitrarily, but samples are assigned to only *one* cluster):

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

3. ***Update the cluster centers (means):***

   $$\mathbf{m}_i = \frac{1}{|C_i|} \sum_{\mathbf{z} \in C_i} \mathbf{z} \qquad i = 1, 2, \ldots, k$$

   where $|C_i|$ is the number of samples in cluster set C_i.
4. ***Test for completion:*** Compute the Euclidean norms of the differences between the mean vectors in the current and previous steps. Compute the residual error, E, as the sum of the k norms. Stop if $E \leq T$, where T a specified, nonnegative threshold. Else, go back to Step 2.

$$\arg\min_{C} \left(\sum_{i=1}^{k} \sum_{\mathbf{z} \in C_i} \|\mathbf{z} - \mathbf{m}_i\|^2 \right)$$

Image Segmentation: K-Means Clustering

a b

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

Additional Notes and Readings

Image segmentation is a fundamental problem in image processing and computer vision

There are huge number of algorithms available for different applications

General-purpose image segmentation is far from well solved

It is still a research problem that is being investigated by many researchers

Deep-learning-based Tools and resources:

Image Segmentation with Deep Learning (Guide) - viso.ai Click here

Segment Anything | Meta AI (segment-anything.com) Click here

Paper tables with annotated results for Segment Everything Everywhere All at Once | Papers With Code Click here

Slide credit: Yan Tong