Image Segmentation

→ Basic of Image.

→ Enhancement.

- Point to Point –
- Local → Convolⁿ
- Global → Histogram Eq.

→ Convolⁿ – spatial filtering

Smoothen
- box.
- WA
- GF.

→ med. f.

sharpening

FOD
- Ix, Iy
- Prewitt
- Sobel.

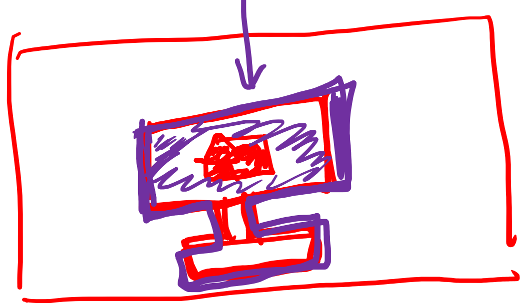
SOD → laplacian mn

sharp.

→ composite filter

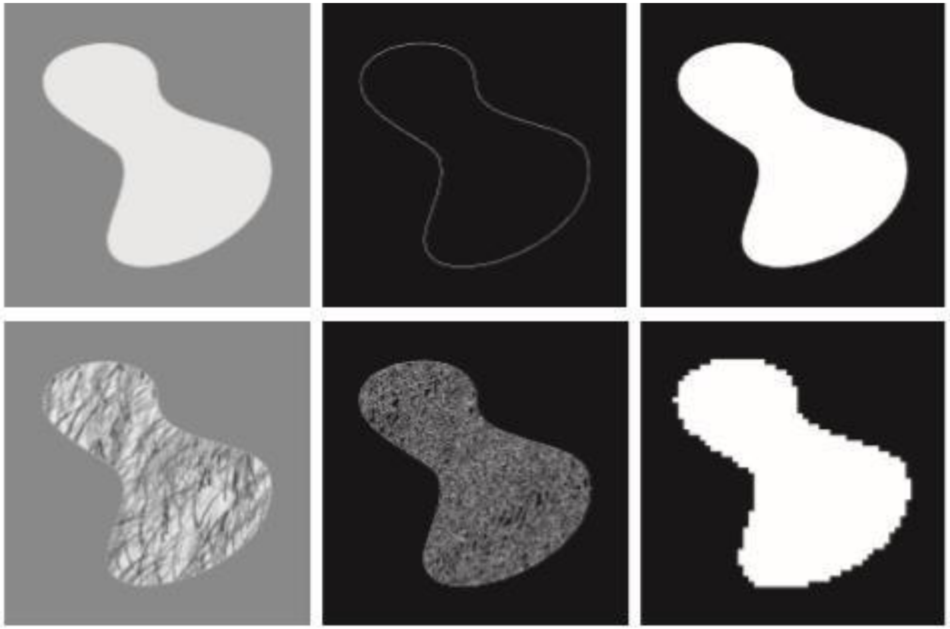
Edge.

LoG.



- connected comp
- edge detection
- thresholding.

Image Segmentation



a b c
d e f

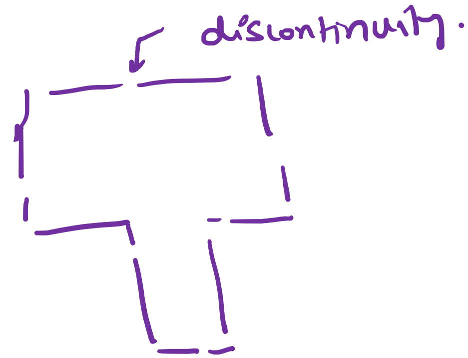
FIGURE 10.1
(a) Image of a constant intensity region.
(b) Boundary based on intensity discontinuities.
(c) Result of segmentation.
(d) Image of a texture region.
(e) Result of intensity discontinuity computations (note the large number of small edges).
(f) Result of segmentation based on region properties.

Segmentation

- Discontinuity based
  - Point, line ,edge (edge linking)
- Region based
  - Thresholding, Region Growing , Splitting and Merging

9 connected

similarity based.



Edge linking & Boundary Detection

Contents

- Edge Linking and Boundary Detection
  - Edges detected by various edge detection operators are not continuous because of following reasons
    - Non uniform illumination
    - Noise
    - Spurious edge points

Edge Linking and Boundary Detection

- Ideally, edge detection should yield sets of pixels lying only on edges.
- In practice, these pixels seldom characterize edges completely, there are breaks.
- Therefore, edge detection typically is followed by linking algorithms designed to assemble edge pixels into meaningful edges and/or region boundaries.

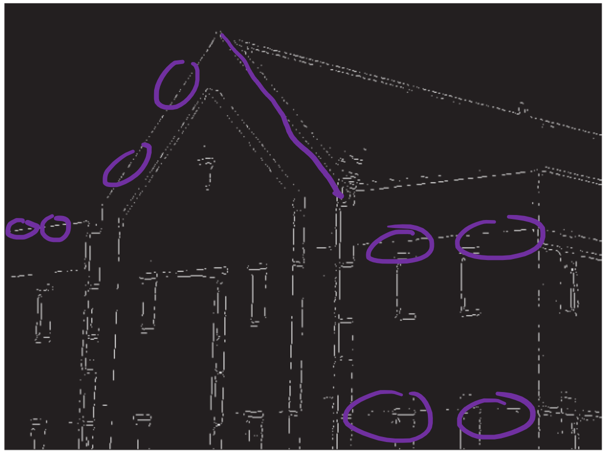
Edge Linking and Boundary Detection

- Basic approaches
  - Local Processing (requires knowledge about edge points in a local region, e.g. 3x3 neighborhood),
  - Global Processing (works with an entire edge image) via Hough Transform

Edge Detection

Edge is detected by finding discontinuities in intensity positions.

(a) Original Image



(b) Edge Map using LoG

(c) Edge Map using Canny

Edge Linking and Boundary Detection

- Ideally, edge detection should yield sets of pixels lying only on edges.
- Edges detected are not continuous due to:
  - Presence of noise
  - Spurious edge points
  - Non uniform illumination

- Edge detection -> Edge linking algorithms

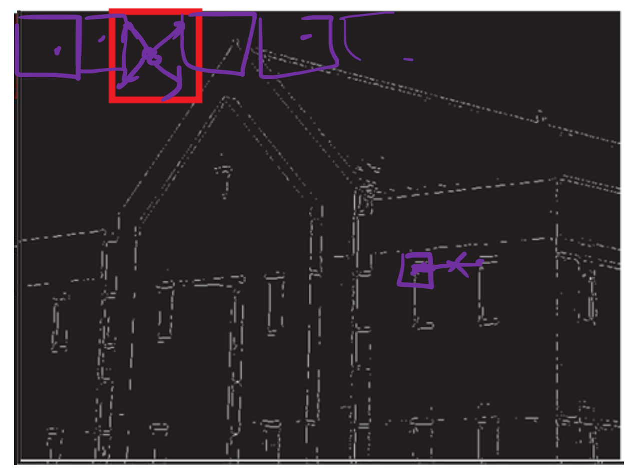
- Assemble edge pixels into meaningful edges and/or region boundaries.

Connect these edge points?

Local Processing

- **Input is a binary edge detected image**
- Analyze each pixel in small neighborhood of (x, y) and find neighbors (x', y') similar to (x, y)
- Similarity is measured by
  - (1) Strength of the response of gradient operators (∇) mag.
  - (2) Direction of the gradient (α)
- Points (x, y) and (x', y') are similar if
  - $\nabla f(x, y) - \nabla f(x', y') \leq T$ and
  - $\alpha(x, y) - \alpha(x', y') \leq A$

  where $(x', y') \in N_{(x,y)}$



Binary Edge Map Image

- Similar points are linked to form edges.
- The above strategy is expensive.
- A record has to be kept of all linked points by, for example, assigning a different label to every set of linked points.

3x3

Local Processing

- The above strategy is expensive. A record has to be kept of all linked points by, for example, assigning a different label to every set of linked points.

(1) **Compute $M(x,y)$ and $\alpha(x,y)$ of input image $f(x,y)$**

$M(x,y) = |\nabla f(x,y)|$: **Magnitude of gradient vector**

$\alpha(x,y)$: **Direction of gradient vector**

(2) **Form binary image**

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

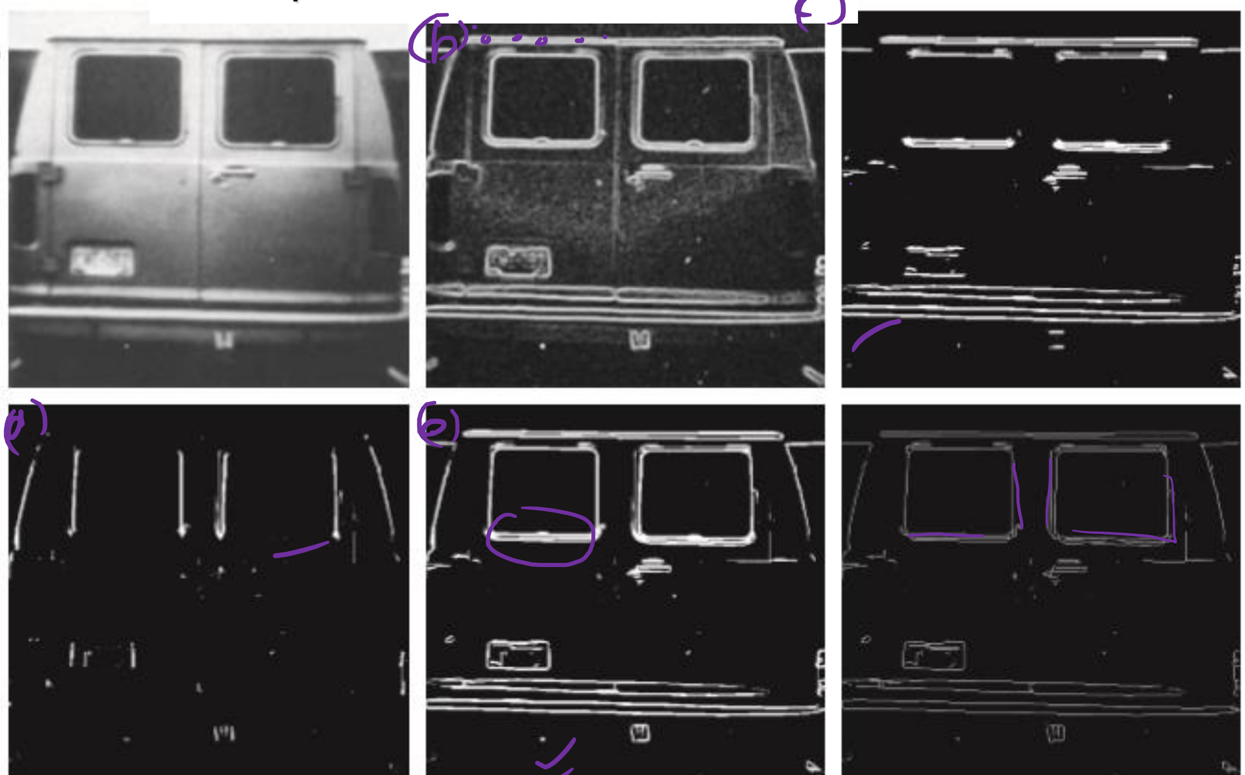
(3) **Scan rows of g and fill (set to 1) all gaps (sets of 0s) in each row that do not exceed a specified length K**

(4) **Rotate g by θ and apply step (3). Rotate result back by $-\theta$.**

To detect gaps in any other direction

Local Processing Example

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



a b c
d e f

FIGURE 10.27
(a) 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 (c) and (d).
(f) Final result, using morphological thinning. (Original image courtesy of Perceptics Corporation.)

- Figure 10.27(b) shows the gradient magnitude image, M(x,y) and Figs. 10.27(c) and (d) show the result obtained by letting T_M equal to 30% of the maximum gradient value, A = 90°, T_A = 45°, and filling all gaps of 25 or fewer pixels (approximately 5% of the image width).

binary

- A large range of allowable angle directions was required to detect the rounded corners of the license plate enclosure, as well as the rear windows of the vehicle.

Brute force is costly

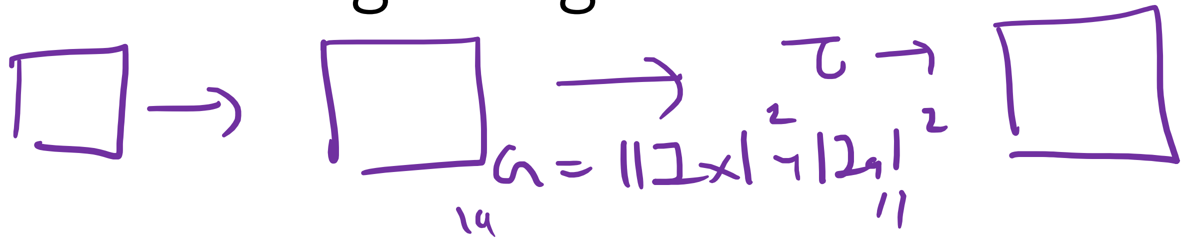
Global Techniq.
↳ Hough Transform.

- Often, we work in unstructured environments in which all we have is an edge map and no knowledge about where objects of interest might be.
- Here all pixels are candidates for linking, and thus have to be accepted or eliminated based on predefined *global* properties
- We attempt to link edge pixels that lie on specified curves

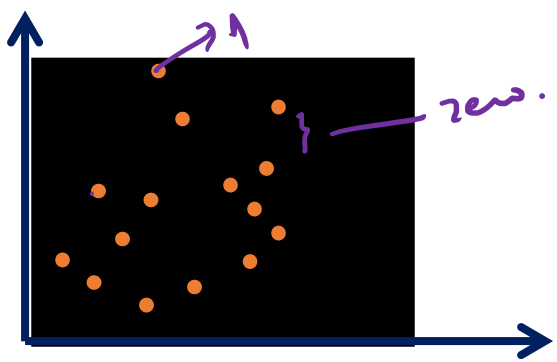
Global Processing - Hough Transform

- Richard Duda and Peter Hart in 1972
- Developed an approach based on whether sets of pixels lie on curves of a specified shape.
- Once detected, these curves form the edges or region boundaries of interest.

Line Detection using Hough Transform



- Input to the Hough transform is a binary thesholded edge image
- We have 'n' edge pixels that partially define boundary of some object.

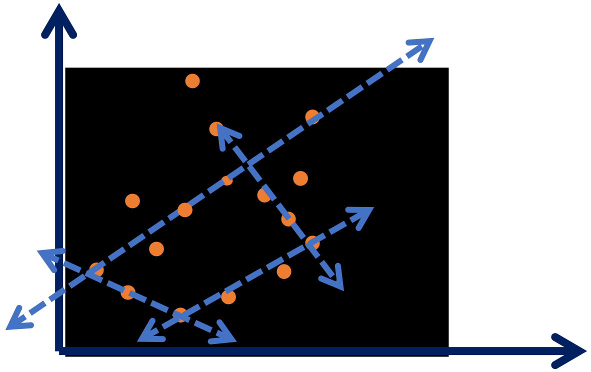


Binary Edge Map Image

- We wish to the find pixels that make up straight lines

Line Detection using Hough Transform

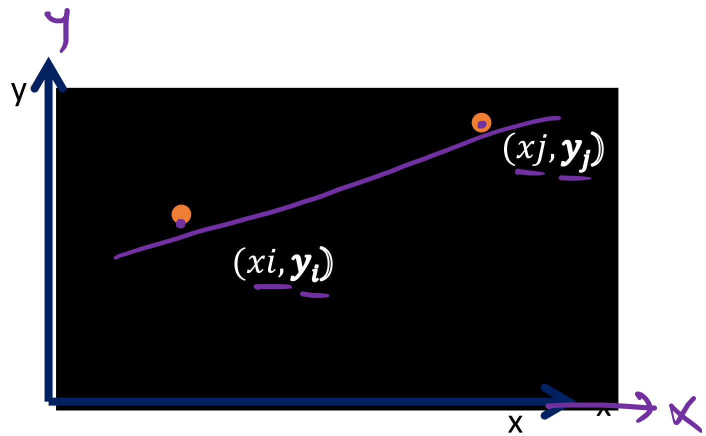
- Input to the Hough transform is a thesholded edge image
- We have 'n' edge pixels that partially define boundary of some object.



- We wish to the find pixels that make up straight lines

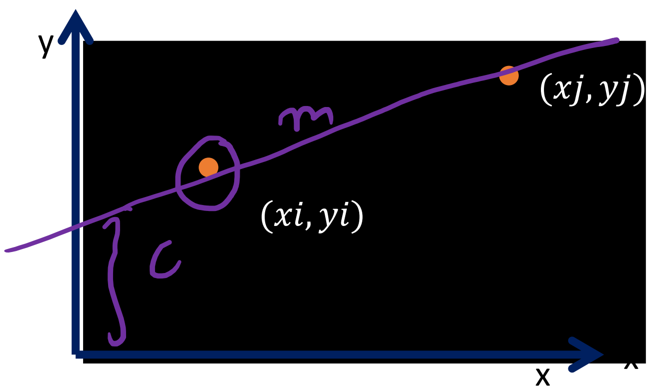
Line Detection using Hough Transform

- Transform coordinate space (x,y) to parameter space (m,c)



Line Detection using Hough Transform

- Transform coordinate space (x,y) to parameter space (m,c)



$$\left(y_i = \boldsymbol{m}x_i + \boldsymbol{c} \right) \Longleftrightarrow \boxed{c = -\boldsymbol{x_i}m + \boldsymbol{y_i}}$$

$y = mx + c$

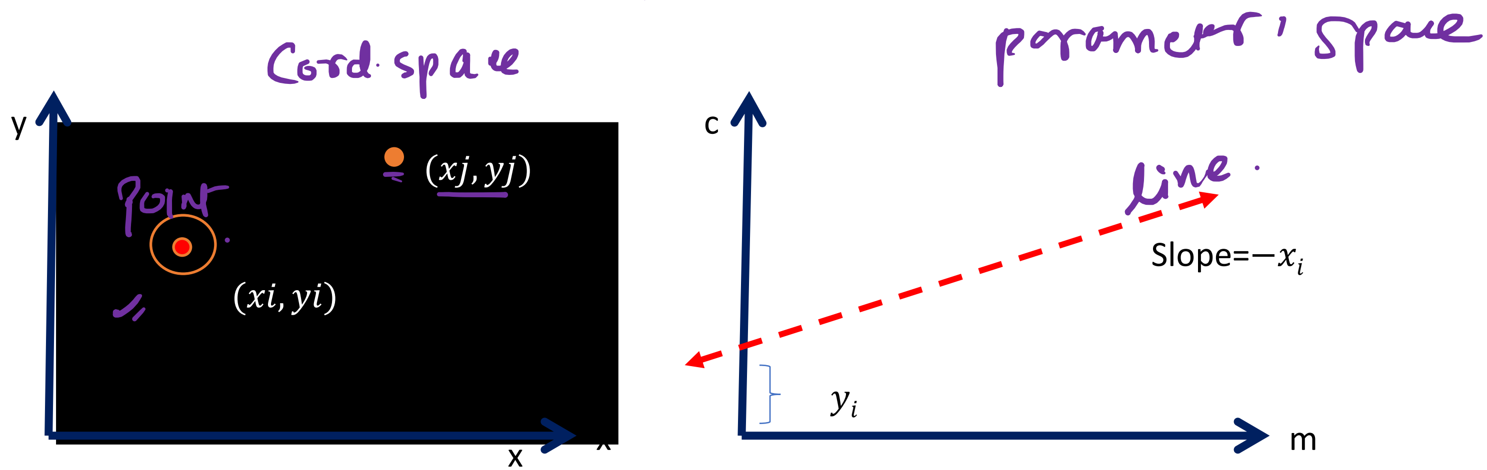
$-mx_i + y_i = c$

Slope **m**, intercept ***c***

Slope $\mathbf{x_i}$, intercept ($\mathbf{y_i}$)

Line Detection using Hough Transform

- Transform coordinate space (x,y) to parameter space (m,c)

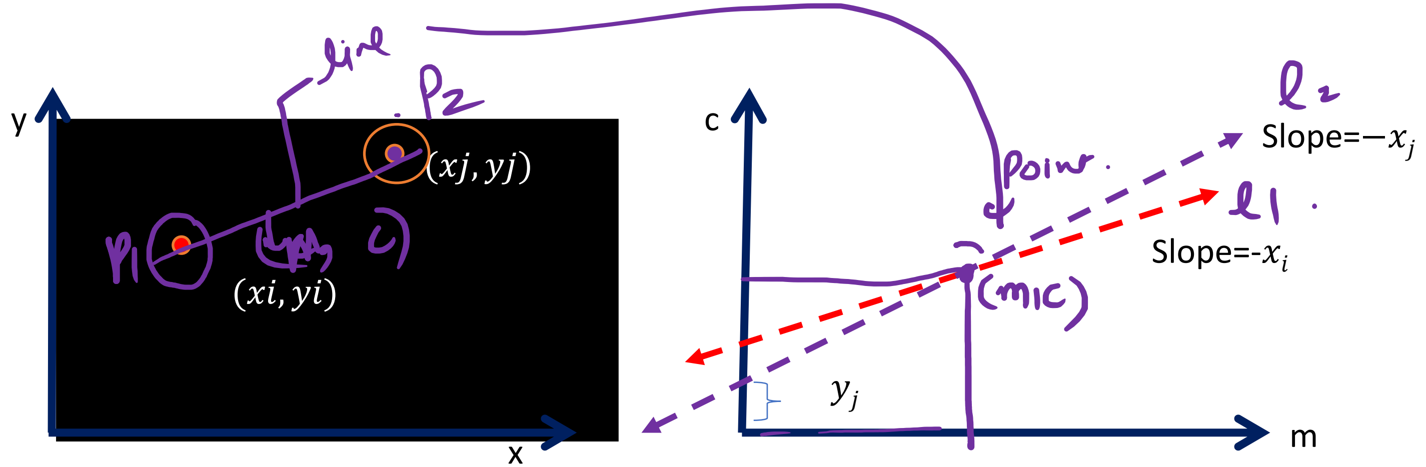


$$y_i = mx_i + c \quad \longleftrightarrow \quad c = -\boldsymbol{x_i} m + \boldsymbol{y_i}$$

- Observation 1: Point in (x, y) space becomes line in (m, c)space

Line Detection using Hough Transform

- Transform coordinate space (x,y) to parameter space (m,c)

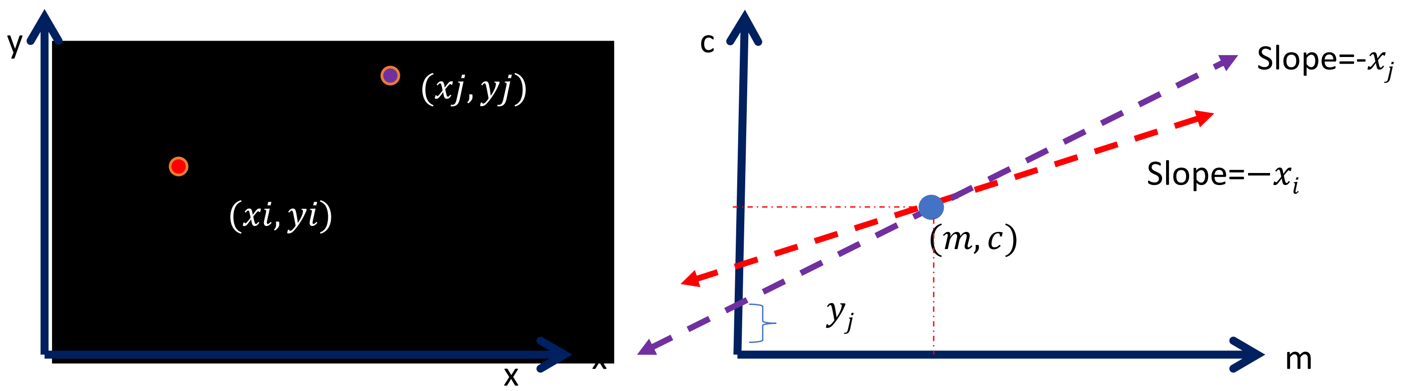


$$y_j = \boldsymbol{m}x_j + \boldsymbol{c}$$

$$c = -\boldsymbol{x_j}m + \boldsymbol{y_j}$$

Line Detection using Hough Transform

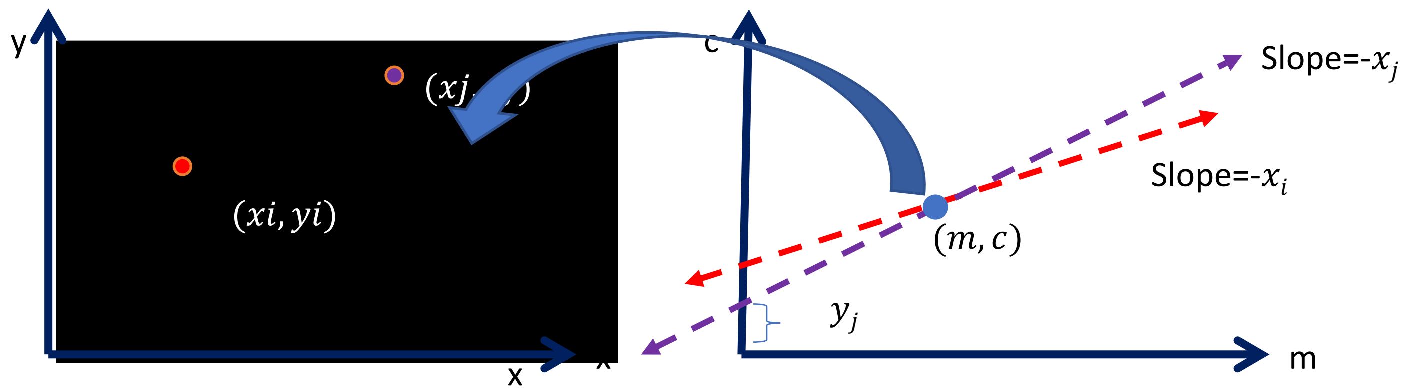
- Transform coordinate space (x,y) to parameter space (m,c)



$$c = -x_j m + yj$$
$$c = -x_i m + yi$$

Line Detection using Hough Transform

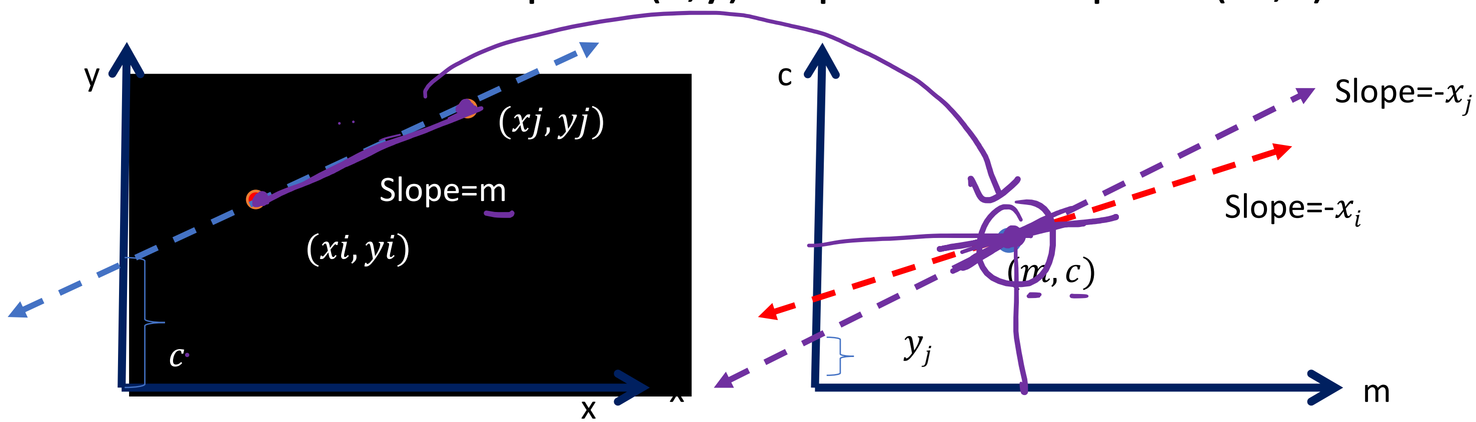
- Transform coordinate space (x,y) to parameter space (m,c)



$$c = -x_j m + yj$$
$$c = -x_i m + yi$$

Line Detection using Hough Transform

- Transform coordinate space (x,y) to parameter space (m,c)



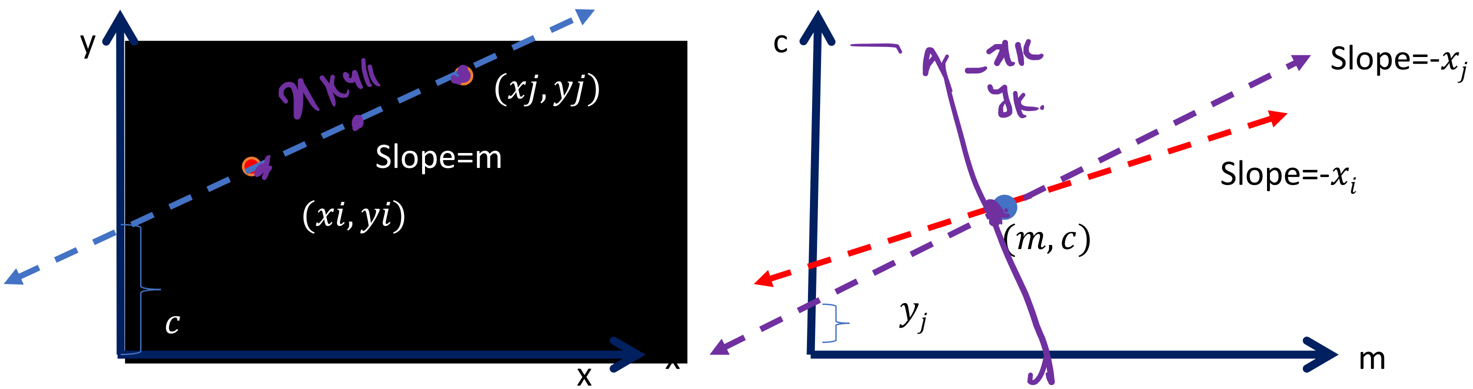
$$y_j = mx_j + c$$
$$y_i = mx_i + c$$

$$c = -x_j m + yj$$
$$c = -x_i m + yi$$

- Observation 2: Line in (x, y) space becomes point in (m, c) space

Line Detection using Hough Transform

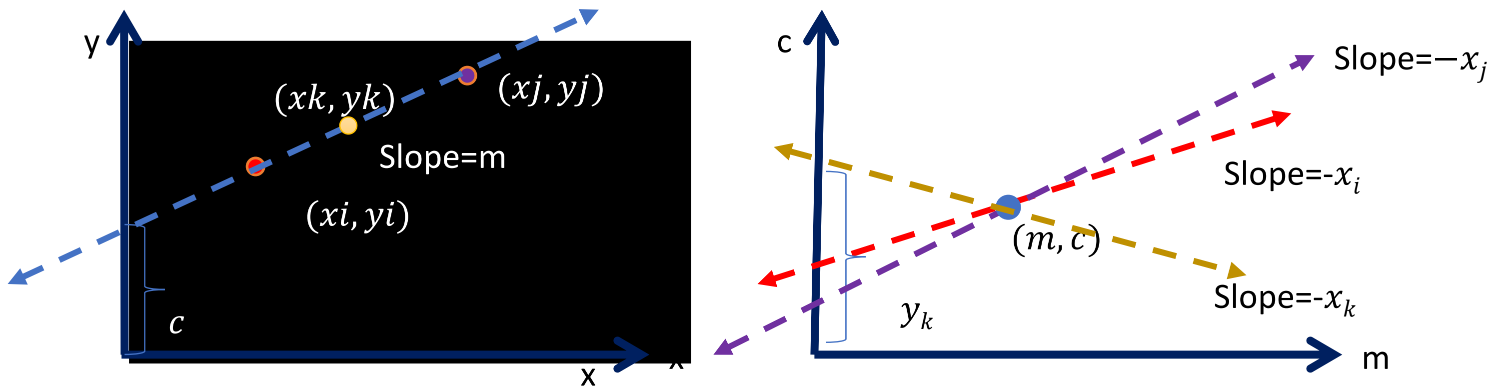
- Transform coordinate space (x,y) to parameter space (m,c)



- Observation 1: Point in (x,y) space ↔ line in (m,c) space
- Observation 2: Line in (x,y) space ↔ point in (m, c) space
- Observation 3: No. of points in same line in (x,y) space ↔ no. of intersecting lines in(m,c) space

Line Detection using Hough Transform

- Transform coordinate space (x,y) to parameter space (m,c)



- Observation 1: Point in (x,y) space ↔ line in (m,c) space
- Observation 2: Line in (x,y) space ↔ point in (m, c) space
- Observation 3: No. of points in same line in (x,y) space ↔ no. of intersecting lines in(m,c) space

How is it useful?

$m = \tan\theta.$

$(m) = \tan\theta.$ (range?)

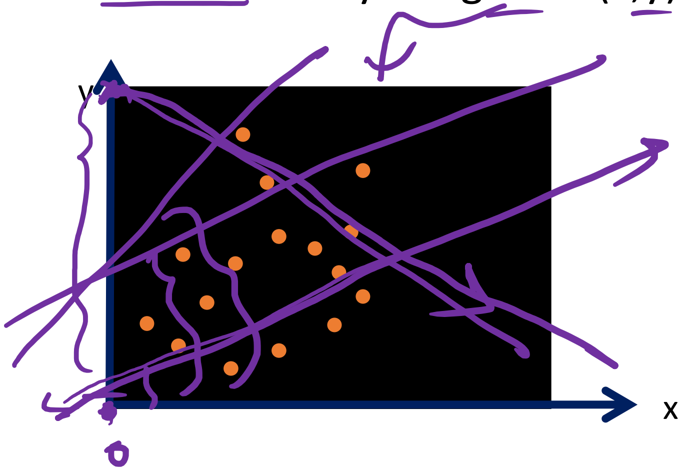
$m = \tan 0^\circ = 0 = m_{min}$

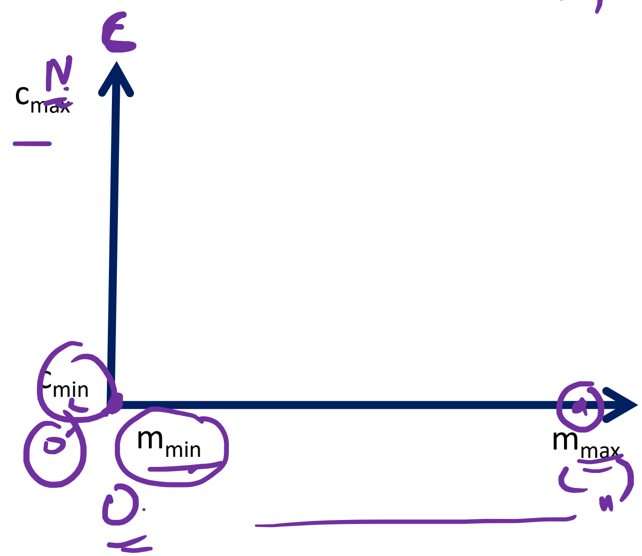
$m = \tan 90^\circ = \infty$

$(\ ) = \tan 85^\circ = \max M$

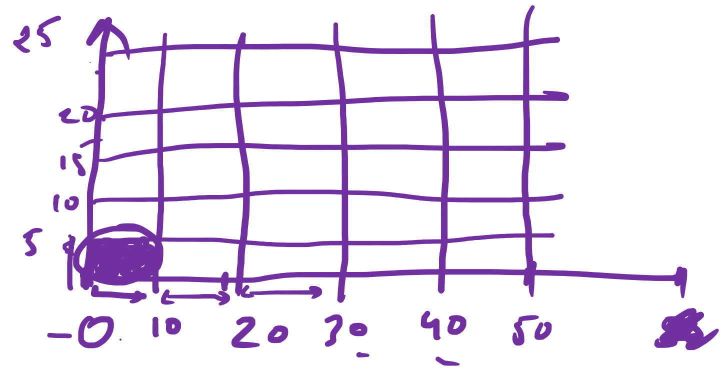
- Given a binary image I in (x,y) space

MIC.

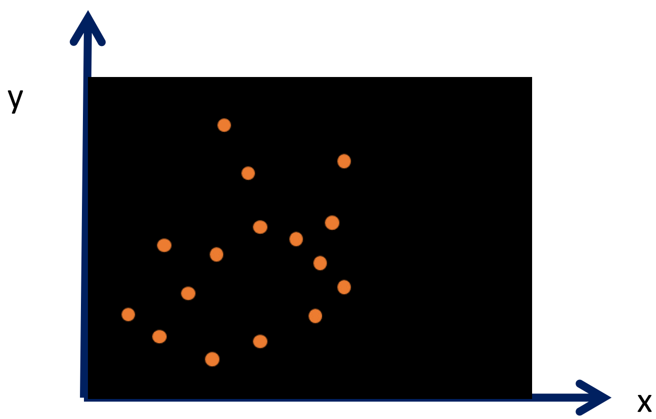


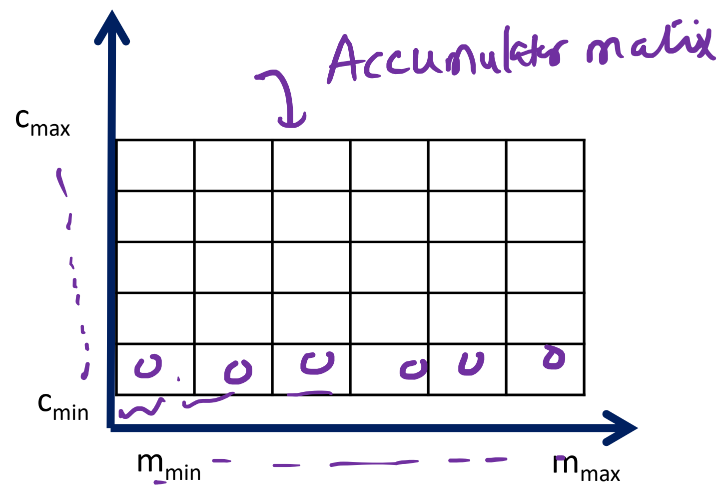


How is it useful?



- Given a binary image I in (x,y) space.





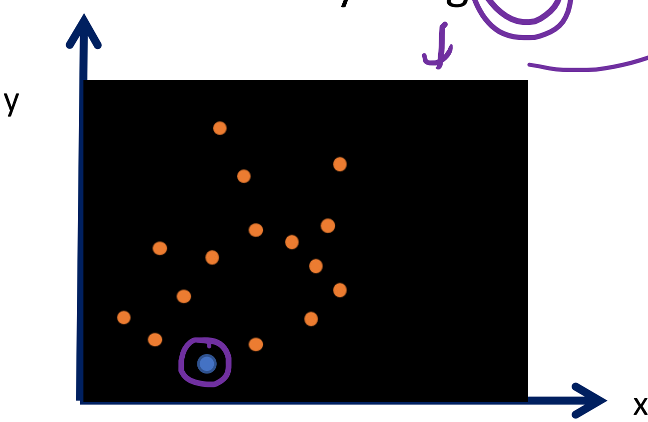
- Discretize and quantize (m,c) space to cells in range m_{min} - m_{max}, c_{min} - c_{max}

How is it useful?

(x_i, y_i) $c_k = -x_i(m_k) + y_i$

① $c_k = -x_i(m_{min}) + y_i$

- Given a binary image I



①

Accumulator matrix A

c_{max}	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
c_{min}	0	0	0	0	0	0	0
	m_{min}						m_{max}

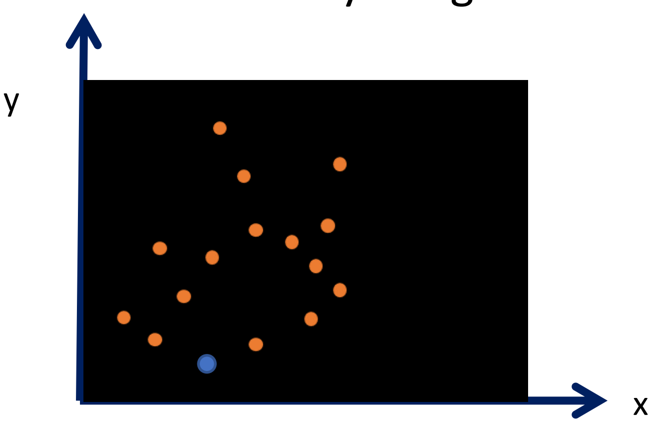
- Step 2: For each edge point (x_i, yi) ,

compute $c_k = -x_i m_k + (y_i)$, for all m_k, $m_{min} \le m_k \le m_{max}$

and increment $A(P_{m_k}, Q_{c_k}) = A(P_{m_k}, Q_{c_k}) + 1$

How is it useful?

- Given a binary image I



Accumulator matrix A

c_{max}	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
c_{min}	1	0	0	0	0	0	0
	m_{min}						m_{max}

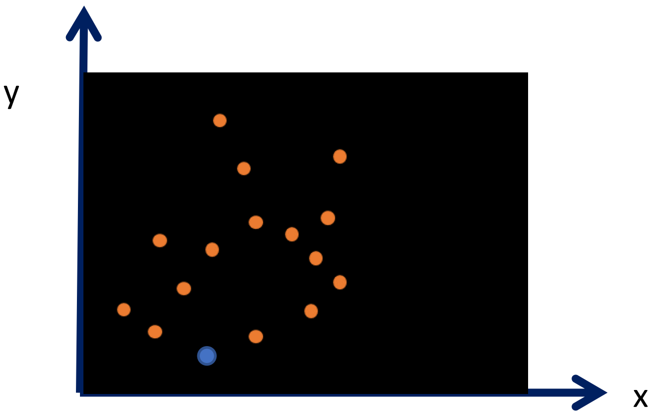
- Step 2: For each edge point (x_i, yi) ,

  compute $\boldsymbol{c_k} = -x_i \boldsymbol{m_k} + (y_i)$, for all m_k, $\boldsymbol{m_{min} \le m_k \le m_{max}}$

and increment $A(P_{m_k}, Q_{c_k}) = A(P_{m_k}, Q_{c_k}) + 1$

How is it useful?

- Given a binary image I



$Accumulator\ matrix\ A$

c_{max}	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	1	0	0	0	0	0
c_{min}	1	0	0	0	0	0	0
	m_{min}						m_{max}

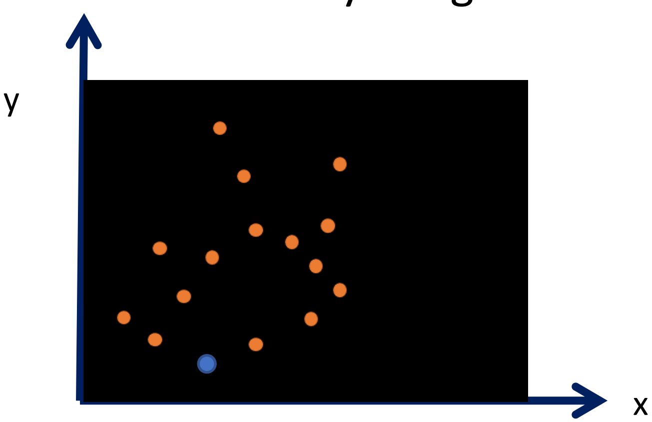
- Step 2: For each edge point (x_i, yi) ,

  compute $c_k = -x_i m_k + (y_i)$, for all m_k, $m_{min} \leq m_k \leq m_{max}$

and increment $A(P_{m_k}, Q_{c_k}) = A(P_{m_k}, Q_{c_k}) + 1$

How is it useful?

- Given a binary image I



Accumulator matrix A

c_{max}	0	0	0	0	0	0	0
	0	0	0	0	0	0	0
	0	0	0	1	0	0	0
	0	0	1	0	0	0	0
	0	1	0	0	0	0	0
c_{min}	1	0	0	0	0	0	0
	m_{min}						m_{max}

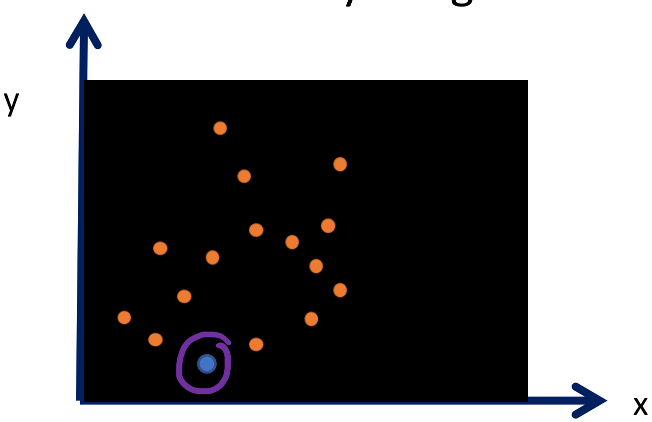
- Step 2: For each edge point (x_i, yi) ,

  compute $c_k = -x_i m_k + y_i$, for all m_k, $m_{min} \leq m_k \leq m_{max}$

and increment $A(P_{m_k}, Q_{c_k}) = A(P_{m_k}, Q_{c_k}) + 1$

How is it useful?

- Given a binary image I



Accumulator matrix A

c_{max}	0	0	0	0	0	1	1
	0	0	0	0	1	0	0
	0	0	0	1	0	0	0
	0	0	1	0	0	0	0
	0	1	0	0	0	0	0
c_{min}	1	0	0	0	0	0	0
	m_{min}						m_{max}

- Step 2: For each edge point (x_i, yi) ,

  compute $c_k = -x_i m_k + y_i$, for all m_k, $m_{min} \leq m_k \leq m_{max}$

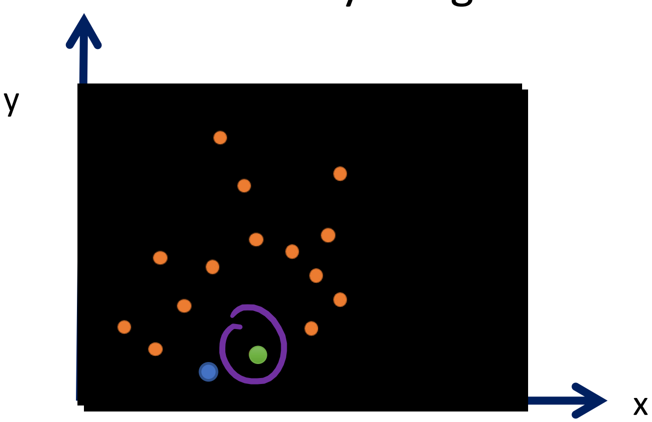
and increment $A(P_{m_k}, Q_{c_k}) = A(P_{m_k}, Q_{c_k}) + 1$

How is it useful?

$c_k = -x_i m_k + y_i$

$m_{min} \leq m_k \leq m_{max}$

- Given a binary image I



Accumulator matrix A

	m_{min}						m_{max}
c_{max}	0	0	0	0	0	1	1
	0	0	0	0	1	0	0
	0	0	0	1	0	0	0
	0	0	1	0	0	0	0
	0	1	0	0	0	0	0
c_{min}	1	0	0	0	0	0	0

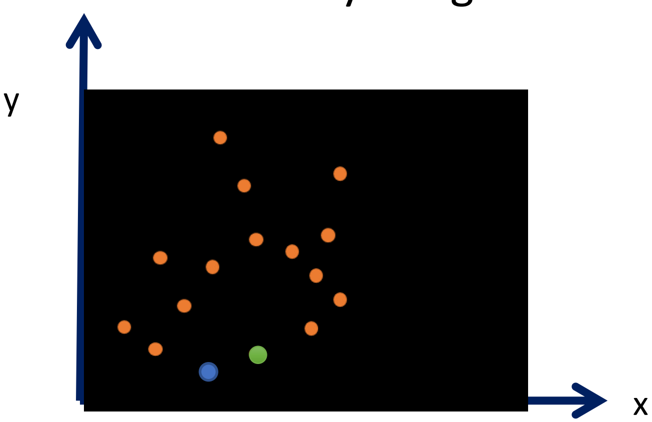
- Step 2: For each edge point (x_i, yi) ,

  compute $c_k = -x_i m_k + y_i$, for all m_k, $m_{min} \leq m_k \leq m_{max}$

and increment $A(P_{m_k}, Q_{c_k}) = A(P_{m_k}, Q_{c_k}) + 1$

How is it useful?

- Given a binary image I



Accumulator matrix A

c_{max}	1	0	0	0	0	1	1
	0	1	0	0	1	0	0
	0	0	0	1	0	0	0
	0	0	1	0	0	0	0
	0	1	0	0	0	0	0
c_{min}	1	0	0	0	0	0	0
	m_{min}						m_{max}

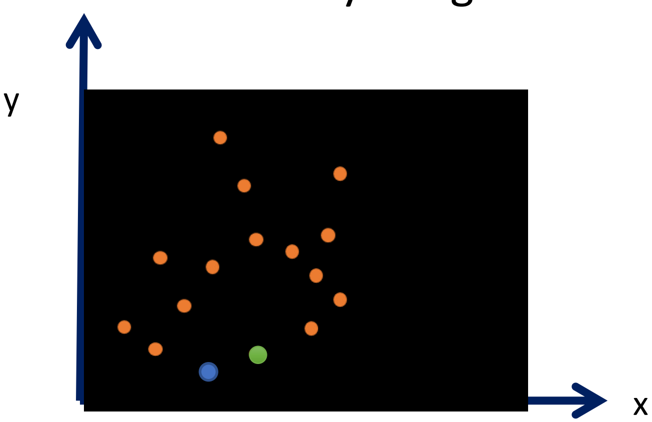
- Step 2: For each edge point (x_i, yi) ,

compute $c_k = -x_i m_k + y_i$, for all m_k, $m_{min} \leq m_k \leq m_{max}$

and increment $A(P_{m_k}, Q_{c_k}) = A(P_{m_k}, Q_{c_k}) + 1$

How is it useful?

- Given a binary image I



Accumulator matrix A

	m_{min}						m_{max}
c_{max}	1	0	0	0	0	1	1
	0	1	0	0	1	0	0
	0	0	1	2	0	0	0
	0	0	1	0	0	0	0
	0	1	0	0	0	0	0
c_{min}	1	0	0	0	0	0	0

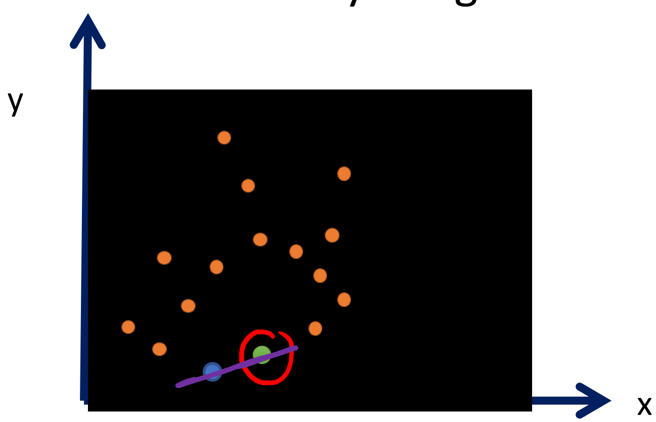
- Step 2: For each edge point (x_i, yi) ,

  compute $c_k = -x_i m_k + y_i$, for all m_k, $m_{min} \leq m_k \leq m_{max}$

and increment $A(P_{m_k}, Q_{c_k}) = A(P_{m_k}, Q_{c_k}) + 1$

How is it useful?

- Given a binary image I



Accumulator matrix A

c_{max}	1	0	0	0	0	1	1
	0	1	0	0	1	0	0
	0	0	1	2	0	0	0
	0	0	1	0	1	0	0
	0	1	0	0	0	1	0
c_{min}	1	0	0	0	0	0	1
	m_{min}						m_{max}

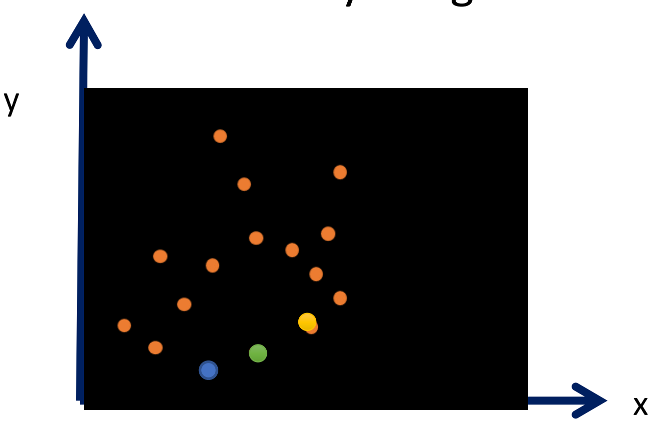
- Step 2: For each edge point (x_i, yi) ,

compute $c_k = -x_i m_k + y_i$, for all m_k, $m_{min} \leq m_k \leq m_{max}$

and increment $A(P_{m_k}, Q_{c_k}) = A(P_{m_k}, Q_{c_k}) + 1$

How is it useful?

- Given a binary image I



Accumulator matrix A

c_{max}	1	0	0	0	0	1	1
	0	1	0	0	1	0	0
	0	0	1	2	0	0	0
	0	0	2	0	1	0	0
	1	2	0	0	0	1	0
c_{min}	1	0	0	0	0	0	1
	m_{min}						m_{max}

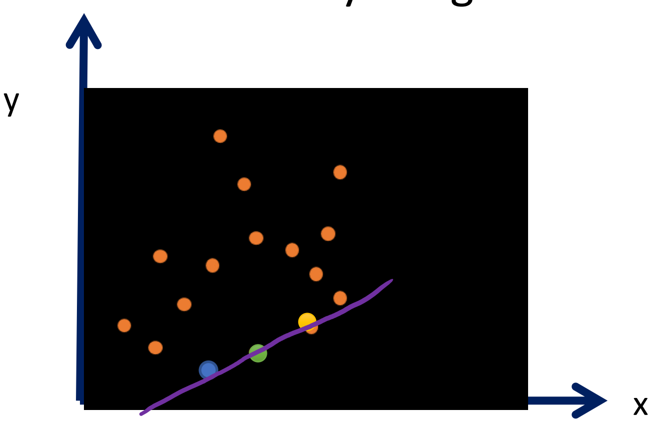
- Step 2: For each edge point (x_i, yi) ,

compute $c_k = -x_i m_k + y_i$, for all m_k, $m_{min} \leq m_k \leq m_{max}$

and increment $A(P_{m_k}, Q_{c_k}) = A(P_{m_k}, Q_{c_k}) + 1$

How is it useful?

- Given a binary image I



$Accumulator\ matrix\ A$

c_{max}	1	0	0	0	0	1	2
	0	1	0	0	1	1	0
	0	0	1	3	1	0	0
	0	0	2	0	1	0	0
	1	2	0	0	0	1	0
c_{min}	1	0	0	0	0	0	1
	m_{min}						m_{max}

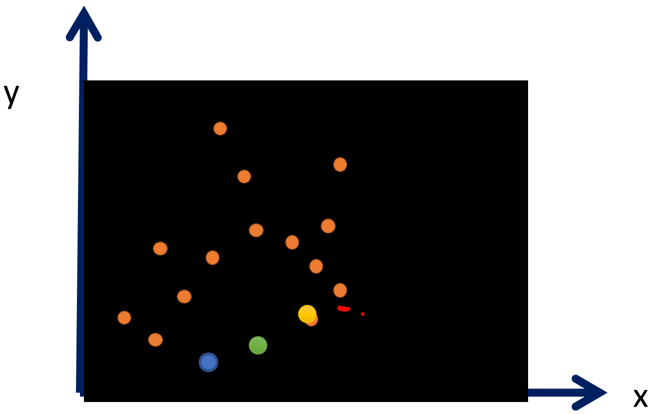
- Step 2: For each edge point (x_i, yi) ,

compute $c_k = -x_i m_k + y_i$, for all m_k, $m_{min} \leq m_k \leq m_{max}$

and increment $A(P_{m_k}, Q_{c_k}) = A(P_{m_k}, Q_{c_k}) + 1$

How is it useful?

- Given a binary image I



$Accumulator\ matrix\ A$

c_{max}	1	0	0	0	0	1	2
	0	1	0	0	1	1	1
	0	0	1	4	3	2	1
	0	1	4	1	1	0	0
	2	3	0	0	0	1	0
c_{min}	2	0	0	0	0	0	1
	m_{min}						m_{max}

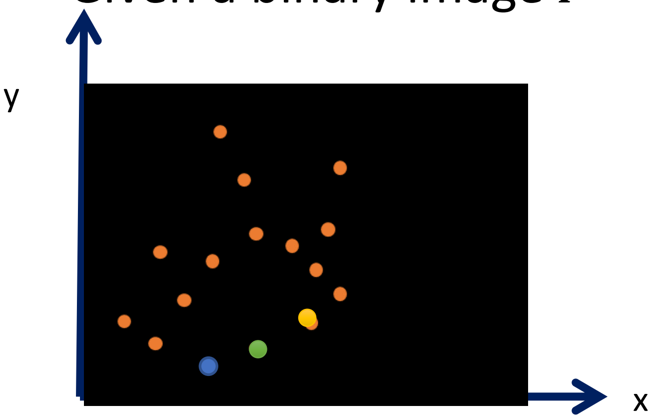
- Step 2: For each edge point (x_i, yi) ,

compute $c_k = -x_i m_k + y_i$, for all m_k, $m_{min} \leq m_k \leq m_{max}$

and increment $A(P_{mk}, Q_{ck}) = A(P_{mk}, Q_{ck}) + 1$

How is it useful?

- Given a binary image I



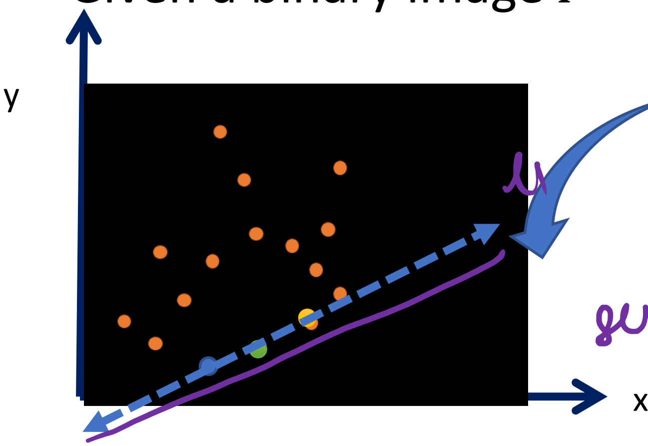
Accumulator matrix A

c_{max}	1	0	0	0	0	1	2
	0	1	0	0	1	1	1
	0	0	1	4	3	2	1
	0	1	4	1	1	0	0
	2	3	0	0	0	1	0
c_{min}	2	0	0	0	0	0	1
	m_{min}						m_{max}

- Step 3: Identify the cell having highest votes-> $A(P, Q)$

How is it useful?

- Given a binary image I



highest vote

Accumulator matrix A

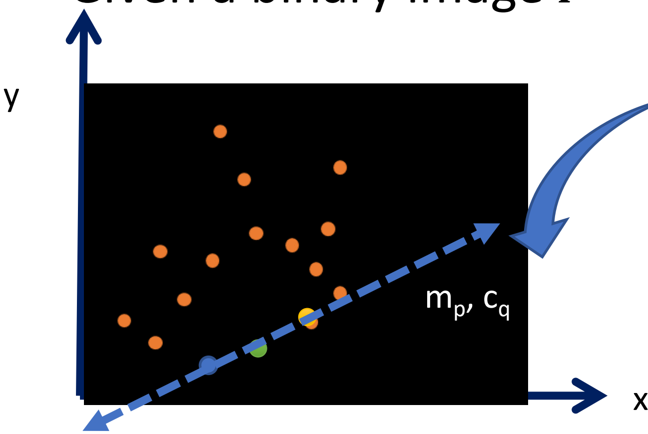
c_{max}	1	0	0	0	0	1	2
	0	1	0	0	1	1	1
	0	0	1	4	3	2	1
	0	1	4	1	1	0	0
	2	3	0	0	0	1	0
c_{min}	2	0	0	0	0	0	1
	m_{min}						m_{max}

second high

- Step 3: Identify the cell having highest votes-> $A(P, Q)$
  - $A(P, Q)$ corresponds to slope m_p and intercept c_q

How is it useful?

- Given a binary image I



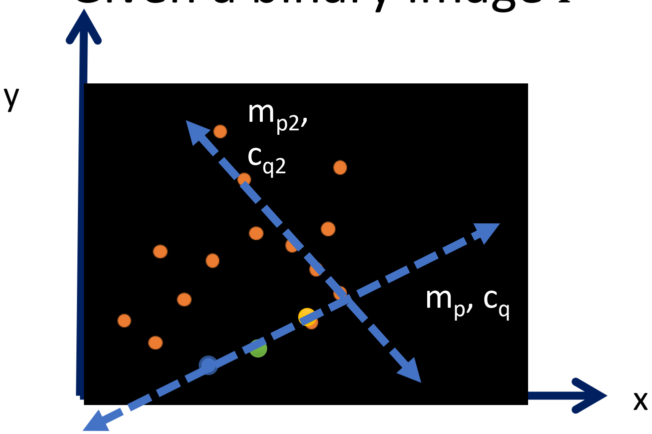
Accumulator matrix A

	m_{min}						m_{max}
c_{max}	1	0	0	0	0	1	2
	0	1	0	0	1	1	1
	0	0	1	4	3	2	1
	0	1	4	1	1	0	0
	2	3	0	0	0	1	0
c_{min}	2	0	0	0	0	0	1

- Step 3: Identify the cell having highest votes-> $A(P,Q)$
  - $A(P,Q)$ corresponds to slope m_p and intercept c_q

How is it useful?

- Given a binary image I



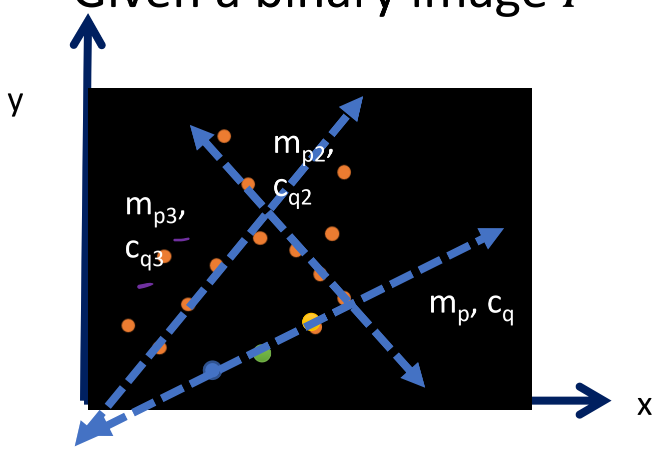
Accumulator matrix A

c_{max}	1	0	0	0	0	1	2
	0	1	0	0	1	1	1
	0	0	1	4	3	2	1
	0	1	4	1	1	0	0
	2	3	0	0	0	1	0
c_{min}	2	0	0	0	0	0	1
	m_{min}						m_{max}

- Step 3: Identify the cell having next highest votes-> $A(P, Q)$
  - $A(P, Q)$ corresponds to slope m_{p2} and intercept c_{q2}

How is it useful?

- Given a binary image I



$Accumulator\ matrix\ A$

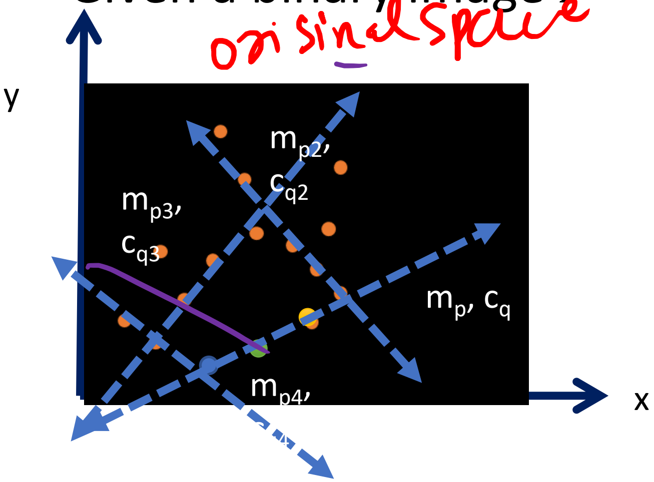
c_{max}	1	0	0	0	0	1	2
	0	1	0	0	1	1	1
	0	0	1	4	3	2	1
	0	1	4	1	1	0	0
	2	3	0	0	0	1	0
c_{min}	2	0	0	0	0	0	1
	m_{min}						m_{max}

- Step 3: Identify the cell having next highest votes-> $A(P,Q)$
  - $A(P,Q)$ corresponds to slope m_{p3} and intercept c_{q3}

How is it useful?

- Given a binary image I

original space



parameter space

Accumulator matrix A

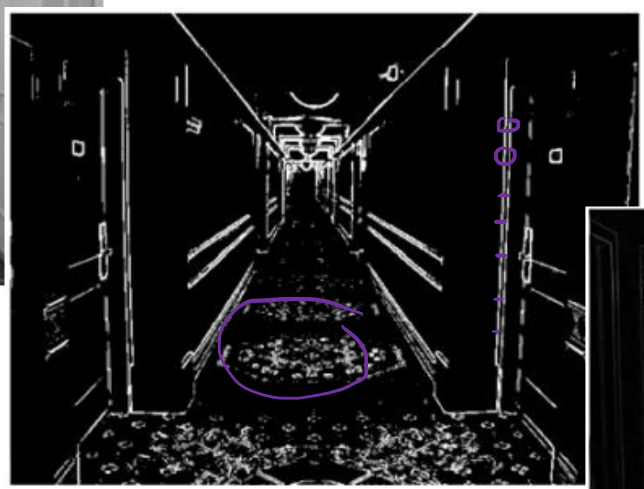
m1c1

c_{max}	1	0	0	0	0	1	2
	0	1	0	0	1	1	1
	0	0	1	4	3	2	1
	0	1	4	1	1	0	0
	2	3	0	0	0	1	0
c_{min}	2	0	0	0	0	0	1
	m_{min}	m2c2					m_{max}

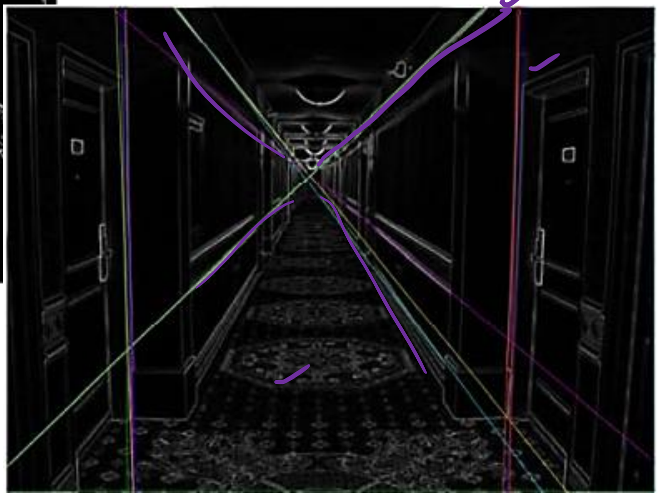
- Step 3: Identify the cell having next highest votes-> $A(P, Q)$
  - $A(P, Q)$ corresponds to slope m_{p4} and intercept c_{q4}

Results

Original image



Edge map Sobel



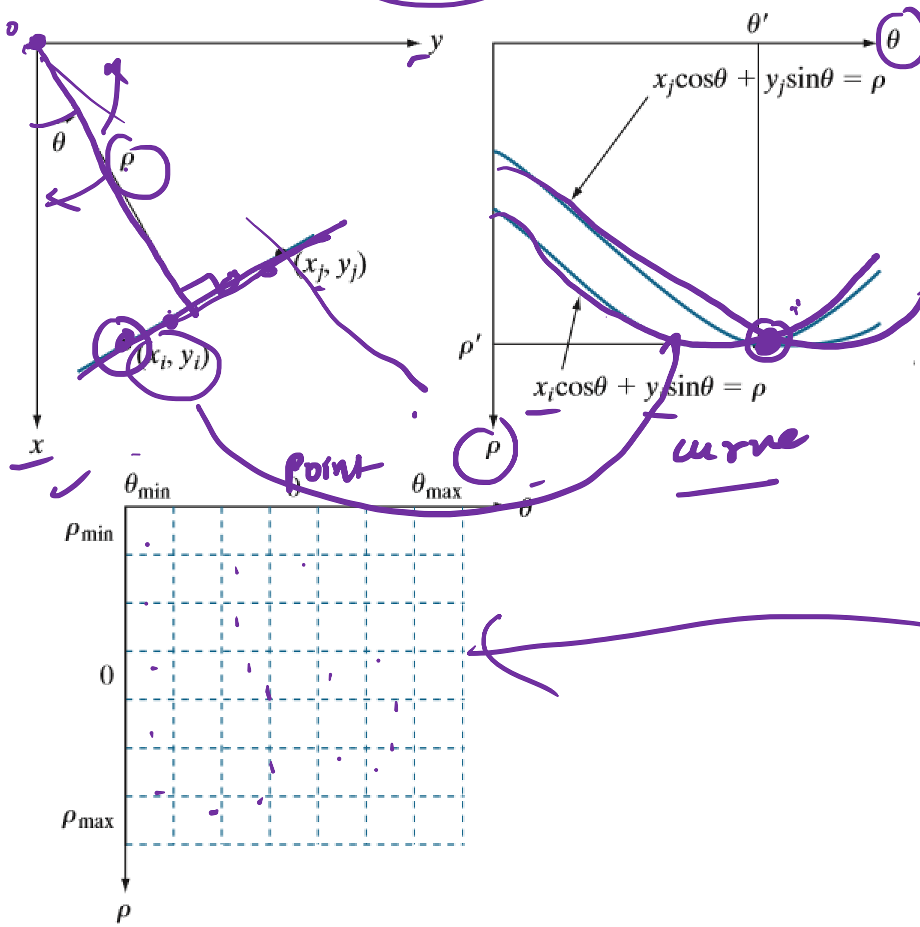
10 prominent lines by Hough

Question?

- There is one limitation here: $\tan 90^\circ = \infty$.:
- Cannot detect vertical lines ?

Point in x-y space → curve in (θ-ρ) $\rho = x\cos\theta + y\sin\theta$

Normal form of line



- Vertical lines can not be dealt with *mc*-plane
- Normal form of line equation is used:
- $\rho = x\cos\theta + y\sin\theta$
  - θ range is $\mp 90°$
  - ρ range is $\mp\sqrt{M^2 + N^2}$
- A point in xy-space gives a sinusoidal in $\theta\rho$-space

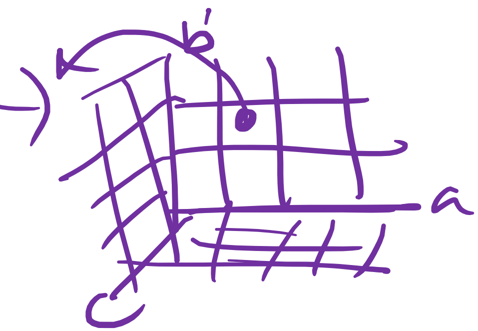
Advantages

$x^2+y^2=r^2$

$(x-a)^2+(y-b)^2=r^2$

$\boxed{ax^2+by^2=c^2}$

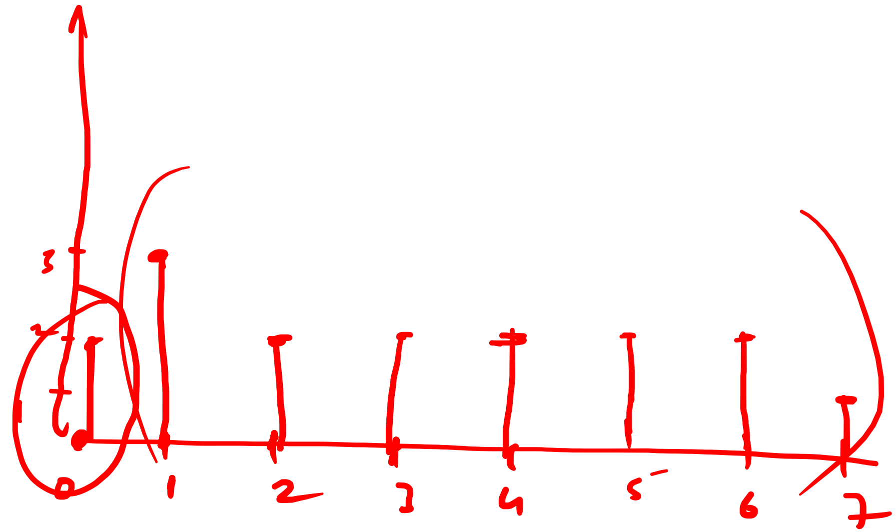
- Conceptually simple.
- Easy implementation.
- Handles missing and occluded data very gracefully.
- Can be adapted to many types of forms, not just lines.



Thresholding

0	3	1	2
1	2	0	3
1	7	5	4
5	4	6	6

0 - 7



T → Intensity histogram.

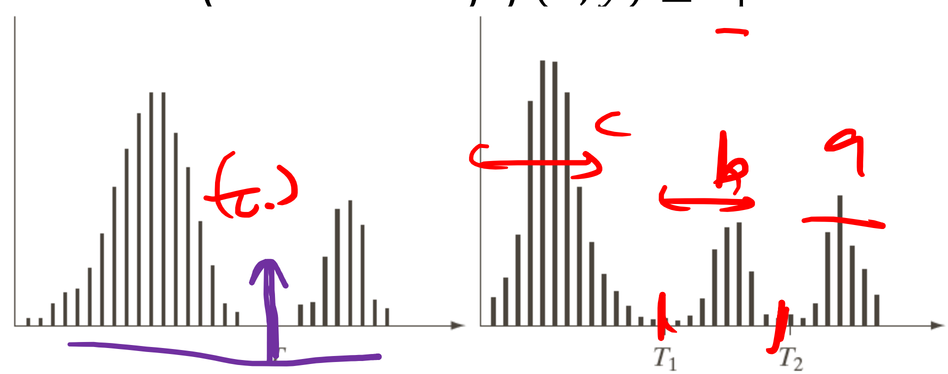
Basic of Intensity Thresholding

- Intensity histogram of an image $f(x,y)$ containing light objects on dark background is shown below. To extract object from background select a threshold T.
- Any image point is called object if $f(x,y) > T$ otherwise background.
- Thresholded image $g(x,y)$ is defined as

$$g(x,y) = \begin{cases} a & \leftarrow \; if\ f(x,y) > T \\ b & \leftarrow \; if\ f(x,y) \le T \end{cases}$$

- Pixels labeled a corresponds to object and labeled b as background.
- Multilevel thresholding is done to classify pixels that belong to background, first object and second object, and thesholded image is defined as

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



a b

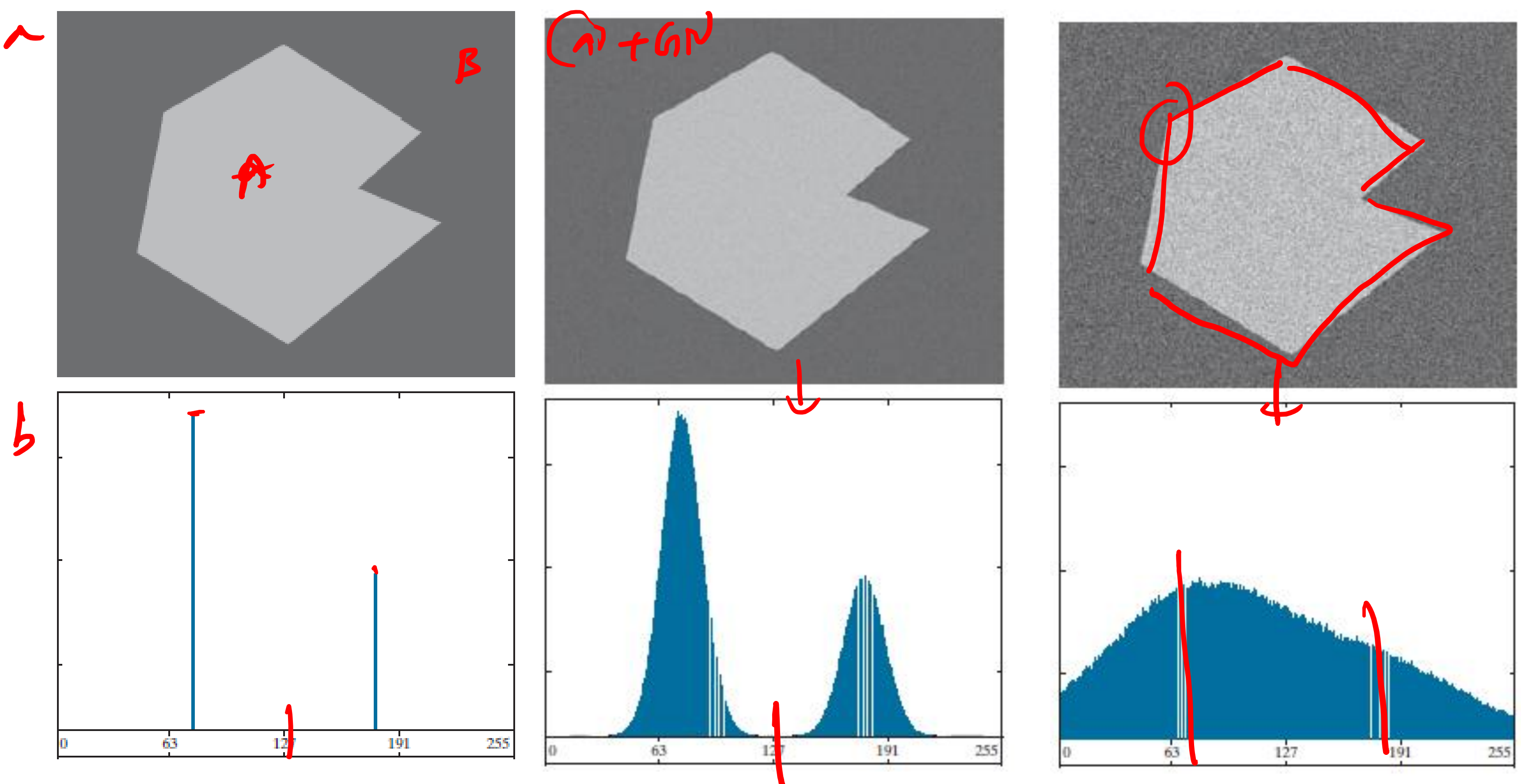
FIGURE 10.35 Intensity histograms that can be partitioned (a) by a single threshold, and (b) by dual thresholds.

Thresholding approaches

- **Global thresholding**
  - When T is constant over entire image.
- **Variable thresholding**
  - When value of T at any point (x, y) depends upon the pixels in neighborhood of (x, y)referred as *local* or *regional* thresholding.
  - If T depends on the spatial coordinates (x, y) themselves, then variable thresholding is often referred as *dynamic* or *adaptive* thresholding.

Thresholding determined by intensity separation using histograms. Good threshold value T can be easily selected if the intensity distribution have tall, narrow, and well separated peaks.

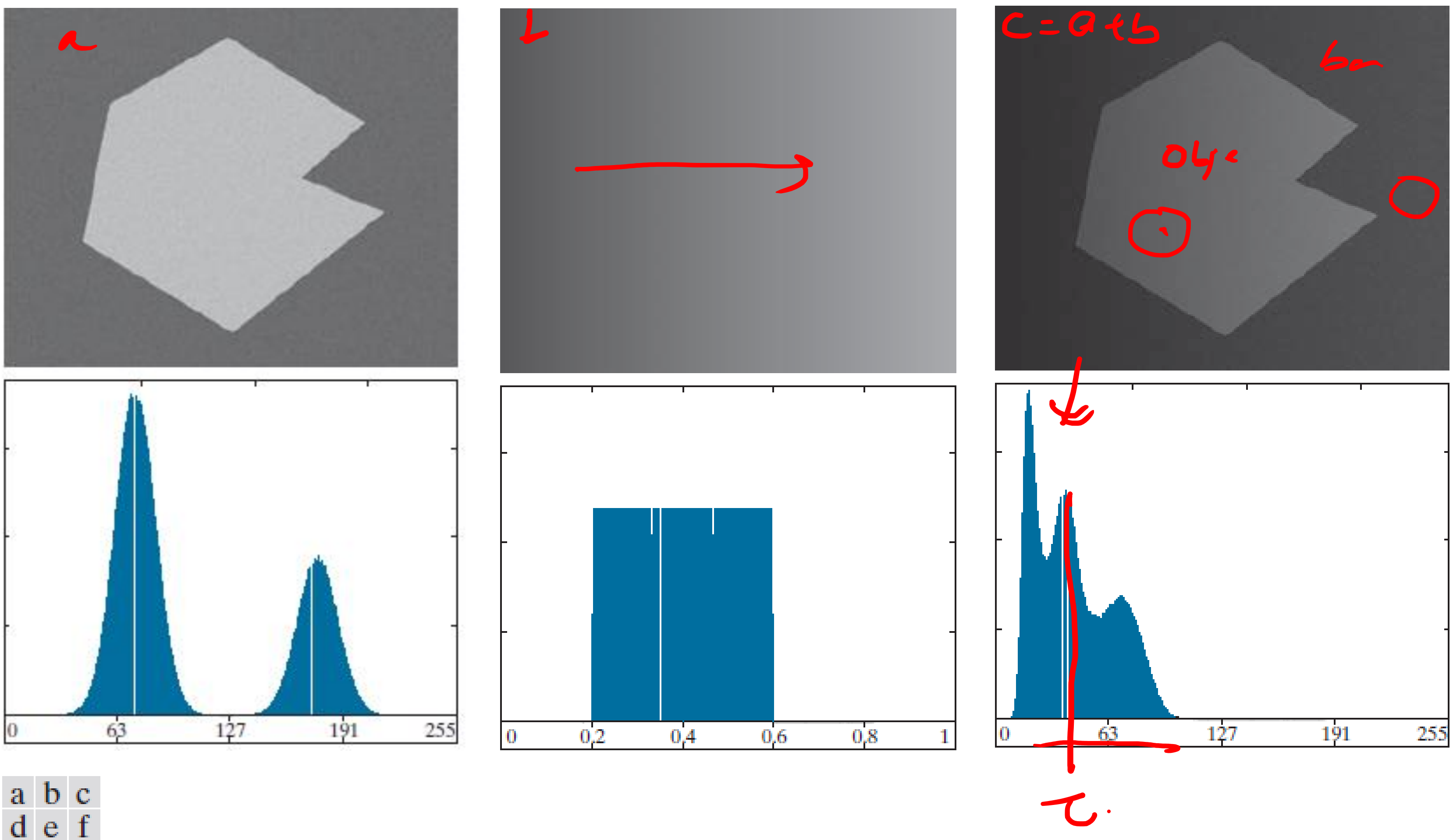
The Role of Noise in Image Thresholding



a b c
d e f

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

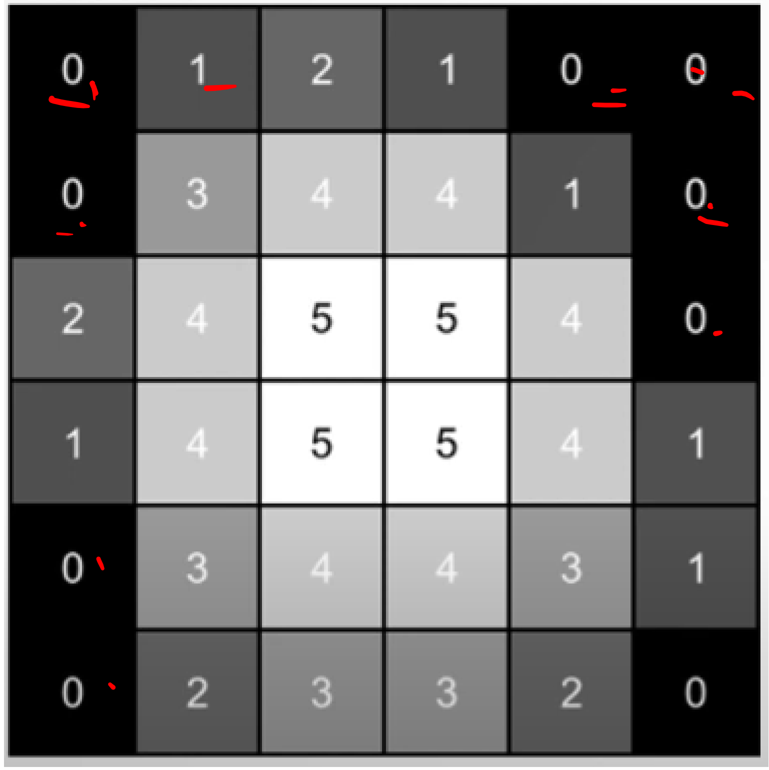
The Role of Illumination and Reflectance in Image Thresholding

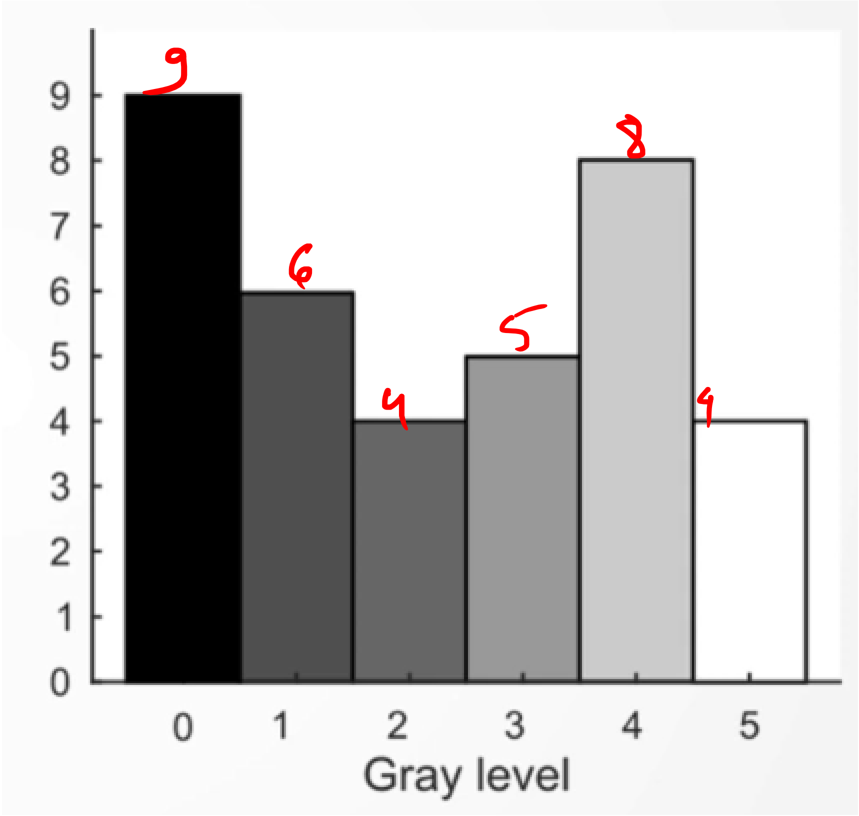


a	b	c
d	e	f

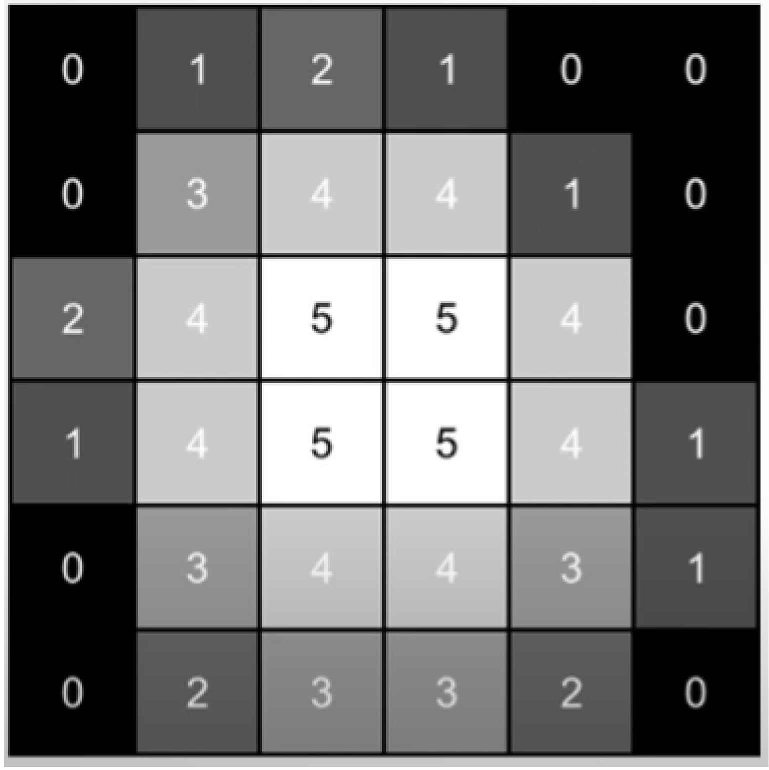
FIGURE 10.34 (a) Noisy image. (b) Intensity ramp in the range [0.2, 0.6]. (c) Product of (a) and (b). (d) through (f) Corresponding histograms.

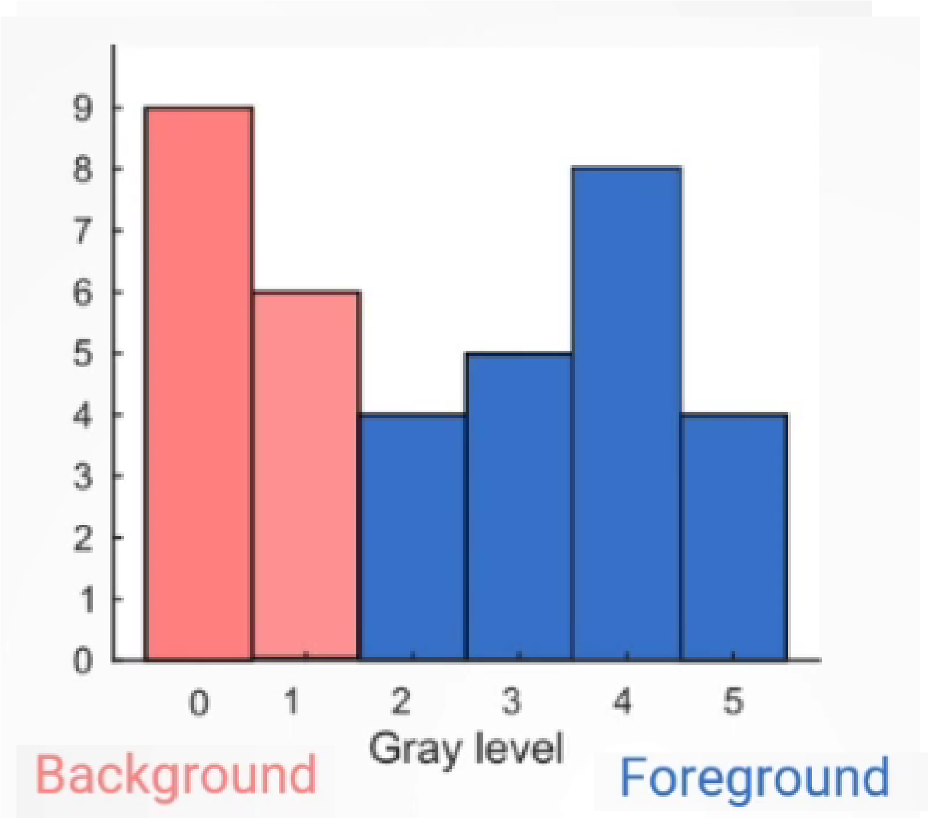
Otsu’s Method





Otsu’s Method





Select an optimum threshold ‘t’ in the sense that it maximizes the *between-class variance*

it is based entirely on computations performed on the histogram of an image

Global Thresholding- OTSU'S method (1979)

- Let components of an image histogram be denoted by

$$\boldsymbol{p_q} = \frac{\boldsymbol{n_q}}{\boldsymbol{n}}\ ,\quad \boldsymbol{q} = 0,1,2,\ldots,L-1$$

  Where $\boldsymbol{n}$ is the number of pixels in image, $\boldsymbol{n_q}$ is of pixels having intensity level $\boldsymbol{q}$ and $\boldsymbol{L}$ is the total number of intensity levels in image.

- Choose a threshold $\boldsymbol{k}$ and divide pixels in two sets $\boldsymbol{C_1}$ and $\boldsymbol{C_2}$, such that $\boldsymbol{C_1}$ is set of pixels with levels $[\mathbf{0},\mathbf{1},\mathbf{2},\ldots,\boldsymbol{k}]$ and $\boldsymbol{C_2}$ is set of pixels with levels $[\boldsymbol{k}+\mathbf{1},\ldots,\boldsymbol{L}-\mathbf{1}]$.
- Otsu's method optimizes the above separation of set and chooses a value of $\boldsymbol{k}$ such that it maximizes the between-class variance ${\boldsymbol{\sigma_B}}^{\mathbf{2}}(\boldsymbol{k})$, defined as

$${\boldsymbol{\sigma_B}}^{\mathbf{2}}(\boldsymbol{k}) = \boldsymbol{P_1}(\boldsymbol{k})[\boldsymbol{m_1}(\boldsymbol{k}) - \boldsymbol{m_G}]^2 + \boldsymbol{P_2}(\boldsymbol{k})[\boldsymbol{m_2}(\boldsymbol{k}) - \boldsymbol{m_G}]^2$$

  Here $\mathbf{P_1}(\mathbf{k})$ is the probability of occurring of set $\mathbf{C_1}$ and $\mathbf{P_2}(\mathbf{k})$ is the probability of occurring of set 2 is given by cumulative sum:

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

  For example if we set $k = 0$, the probability of set C_1 having any pixels assigned is zero.

  Also, $\boldsymbol{P_2}(\boldsymbol{k}) = \mathbf{1} - \boldsymbol{P_1}(\boldsymbol{k})$.

Global Thresholding- OTSU'S method

- Terms $\boldsymbol{m_1(k)}$ and $\boldsymbol{m_2(k)}$ are mean intensities of set $\boldsymbol{C_1}$ and $\boldsymbol{C_2}$ respectively denoted as

$$\boldsymbol{m_1(k) = \sum_{i=0}^{k} iP(i/C_1)}$$

$$\boldsymbol{= \sum_{i=0}^{k} iP(C_1/i).P(i)\,/P(C_1)}$$

$$\boldsymbol{= \frac{1}{P_1(k)} \sum_{i=0}^{k} i\,p_i}$$

as $\boldsymbol{P(C_1/i)}$ is probability of $\boldsymbol{C_1}$ given $\boldsymbol{i}$, which is 1 as we are dealing with values $\boldsymbol{i}$ of from $\boldsymbol{0-k}$ lying in set $\boldsymbol{C_1}$. $\boldsymbol{P(i)}$ is the probability of $\boldsymbol{i^{th}}$ value, which is equal to $\boldsymbol{i^{th}}$ component of histogram $\boldsymbol{p_i}$. Finally $\boldsymbol{P(C_1)}$ is the probability of class $\boldsymbol{C_1}$ which is equal to $\mathbf{P_1(k)}$. Similarly, $\boldsymbol{m_2(k) = \frac{1}{P_2(k)} \sum_{i=k}^{L-1} i\,p_i}$

- and $\boldsymbol{m_G}$ is global mean (of entire image), $\boldsymbol{m_G} = \sum_{i=0}^{L-1} i\,p_i$.
- Let the term $\boldsymbol{m(k)}$ be the cumulative mean upto level $\boldsymbol{k}$ given by

$$\boldsymbol{m(k) = \sum_{i=0}^{k} i\,p_i}$$

Global Thresholding- OTSU'S method

- Let us temporarily omit $\boldsymbol{k}$ for notational clarity
- Validity of the following two equations can be verified by directly substitution of the preceding results:

$$\boldsymbol{P_1 m_1 + P_2 m_2 = m_G} \text{ and}$$
$$\boldsymbol{P_1 + P_2 = 1}$$

- To evaluate the "goodness" of threshold at level $\boldsymbol{k}$ following index is computed

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

where $\boldsymbol{\sigma_G}^2$ is the global variance of all pixels and

$$\boldsymbol{\sigma_B}^2 = \boldsymbol{P_1[m_1 - m_G]^2 + P_2[m_2 - m_G]^2}$$

This expression can also be written as

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

- The expression is computationally efficient as only two parameters, $\boldsymbol{m}$ and $\boldsymbol{P_1}$ have to be computed for all values of $\boldsymbol{k}$ ($\boldsymbol{m_G}$ is computed only once).
- Farther the two means $\boldsymbol{m_1}$ and $\boldsymbol{m_2}$, larger would be $\boldsymbol{\sigma_B}^2$ indicating better separability.

Global Thresholding- OTSU'S method

- The objective is to determine $\boldsymbol{k}$ that maximizes the between class variance $\boldsymbol{\sigma_B}^{\mathbf{2}}$.
- Reintroducing $\boldsymbol{k}$ we have final results as:

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

and

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

- Then the optimum threshold is the value $\mathbf{k}^*$, that maximizes $\boldsymbol{\sigma_B{}^2(k)}$:

$$\boldsymbol{\sigma_B}^{\mathbf{2}}(\mathbf{k}^*) = \text{maximize}\, \boldsymbol{\sigma_B{}^2(k)} \text{ for } \mathbf{0} \leq \boldsymbol{k} \leq \boldsymbol{L} - \mathbf{1}$$

- Once $\mathbf{k}^*$ is computed the input image $f(x, y)$ is segmented as before

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

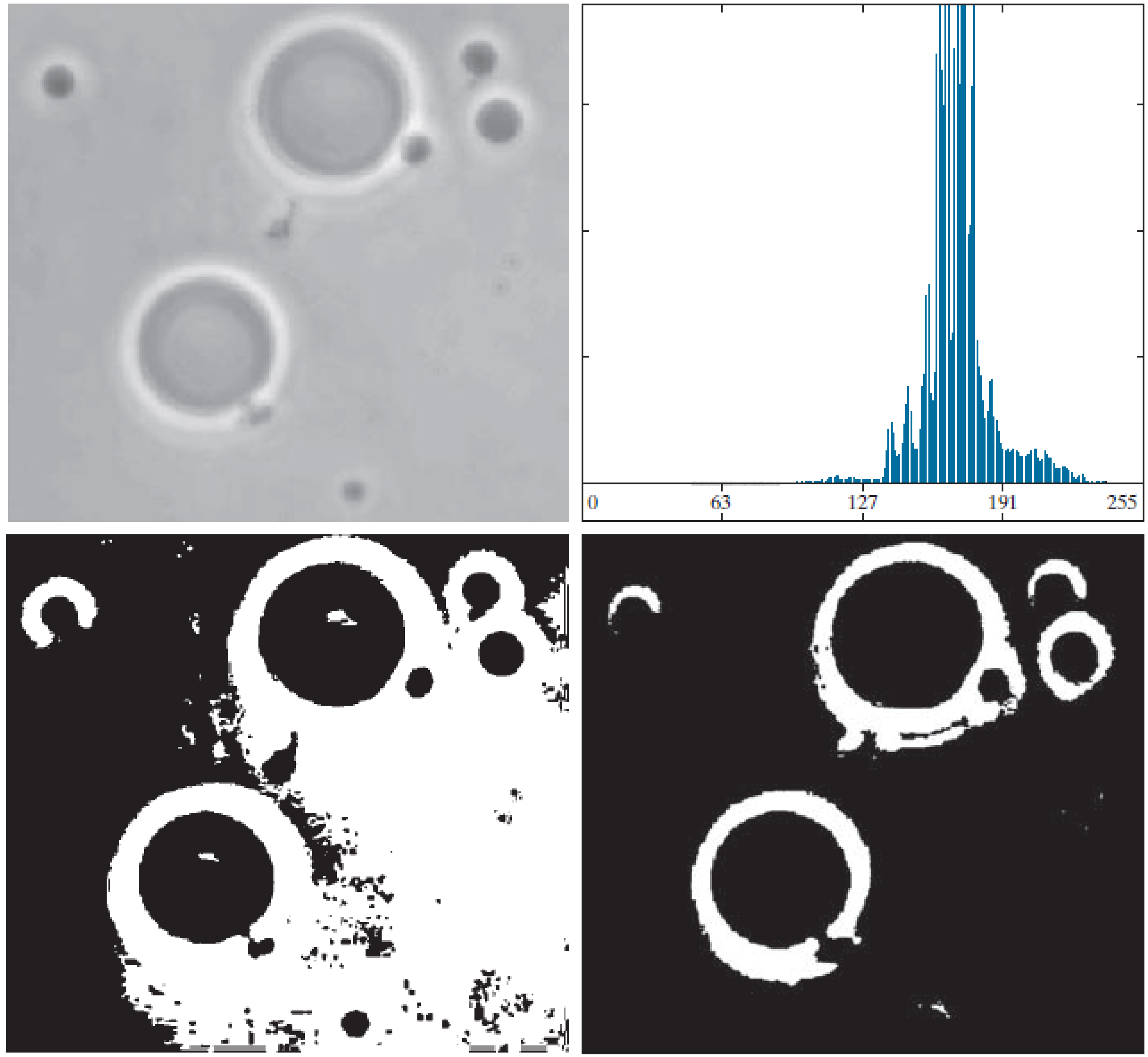
Global Thresholding- **OTSU'S method**

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 $k = 0, 1, 2, \ldots, L-1$, using Eq. (10-49).
3. Compute the cumulative means, $m(k)$, for $k = 0, 1, 2, \ldots, L-1$, using Eq. (10-53).
4. Compute the global mean, m_G, using Eq. (10-54).
5. Compute the between-class variance term, $\sigma_B^2(k)$, for $k = 0, 1, 2, \ldots, L-1$, using Eq. (10-62).
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. Compute the global variance, σ_G^2, using Eq. (10-58), and then obtain the separability measure, η^*, by evaluating Eq. (10-61) with $k = k^*$.

Read Gonzales pg 751

Example - OTSU'S method



a b
c d

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

Edge detection to improve Global Thresholding

- Edge detection prior to inputting subjecting an to global thresholding also improves the output results.
- Indication whether a pixel is of or o edge can be obtained by computing its gradient or absolute value of Laplacian.

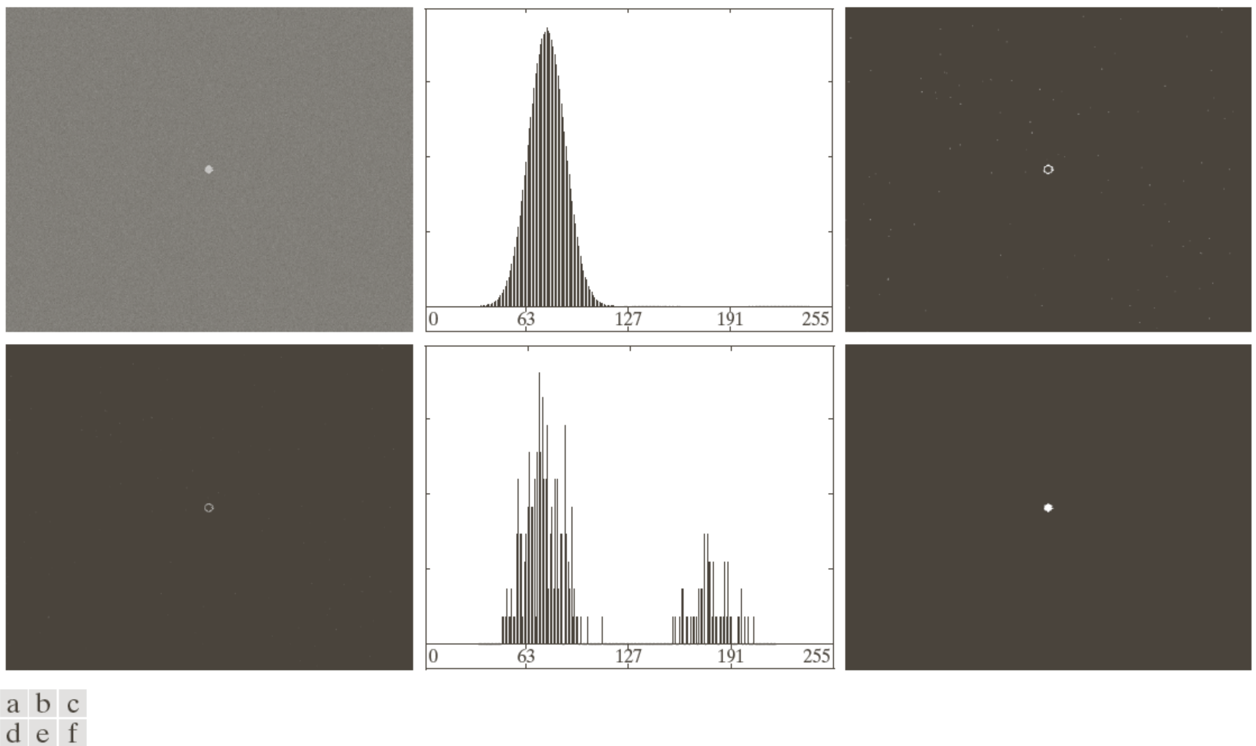
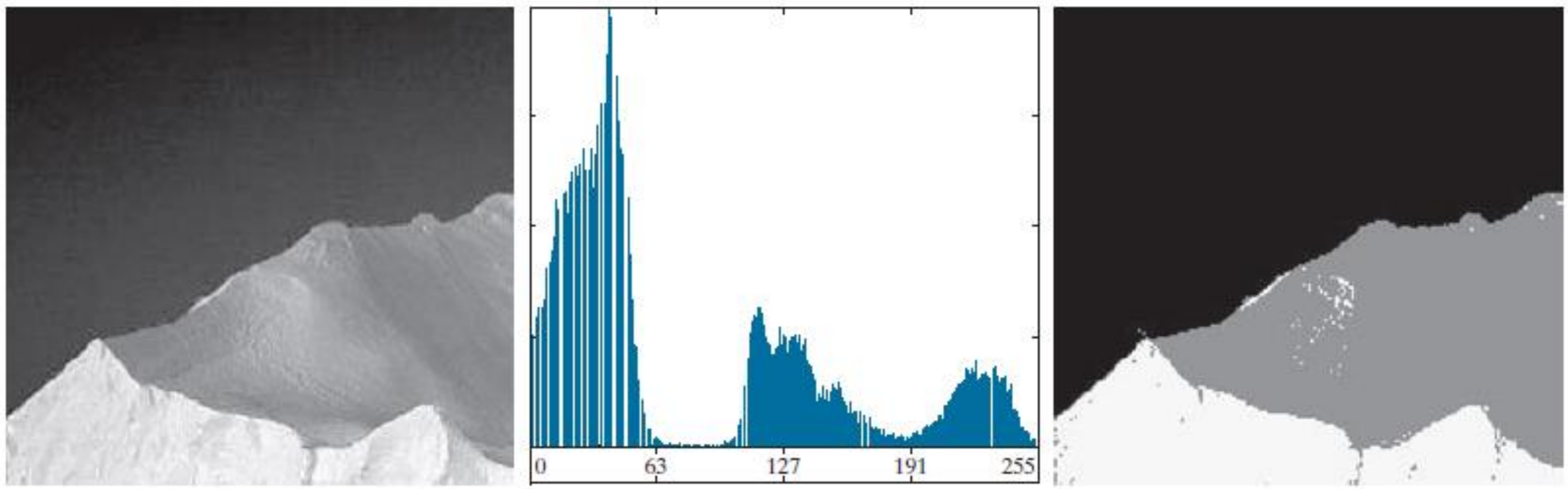


FIGURE 10.42 (a) Noisy image from Fig. 10.41(a) and (b) its histogram. (c) Gradient magnitude image thresholded at the 99.7 percentile. (d) Image formed as the product of (a) and (c). (e) Histogram of the nonzero pixels in the image in (d). (f) Result of segmenting image (a) with the Otsu threshold based on the histogram in (e). The threshold was 134, which is approximately midway between the peaks in this histogram.

Multilevel Otsu

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



a b c

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

VARIABLE THRESHOLDING

- Compute a threshold at every point, (*x*, *y*), in the image based on one or more specified properties in a neighborhood of (*x*, *y*).
  - Let m_{xy} - *mean* of set of pixel values in neighborhood, S_{xy},
  - and σ_{xy} standard deviation of set of pixel values in neighborhood, S_{xy},
  - centered at coordinates (x, y) in an image
- The following are common forms of variable thresholds based on the local image properties:

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

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

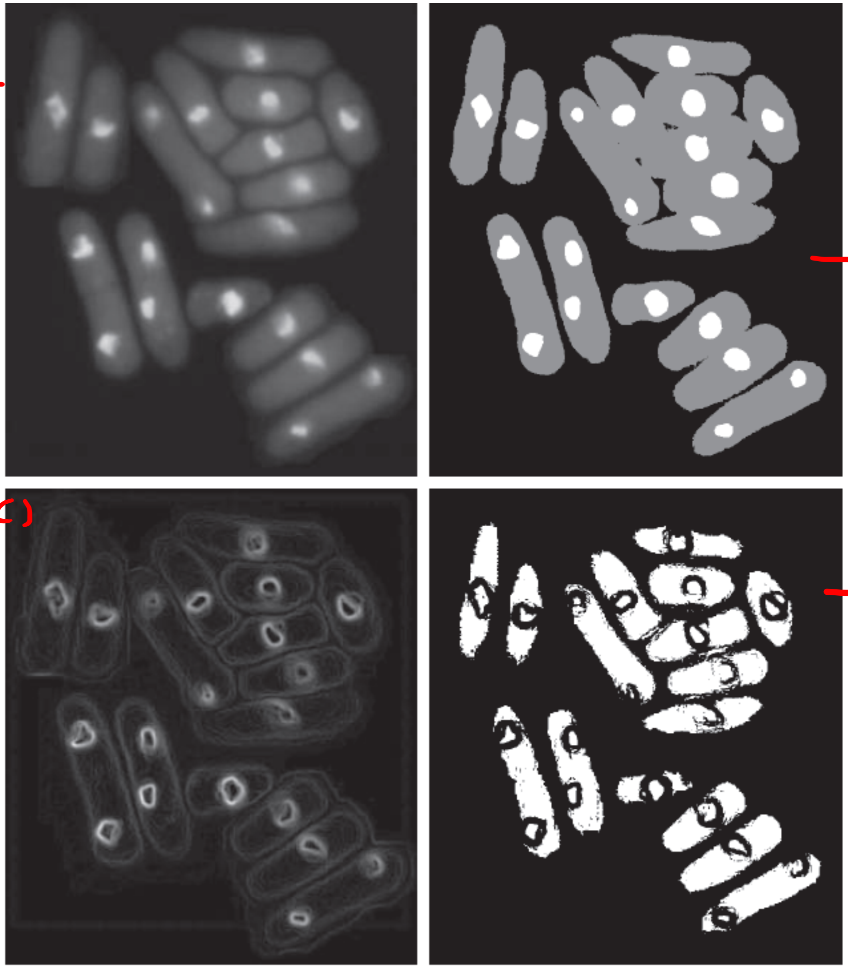
where *a* and *b* are nonnegative constants, and m_G is the global mean

- The segmented image is computed

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

a b
c d

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



(c)

→ Global

→ locally.

Q is a *predicate*

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

- local standard deviation σ_{xy} for all (x, y) input image using a neighborhood of size 3 × 3
- The values a = 30 and b = 1 5

Variable Thresholding Based on Moving Averages

- Raster Scan process
- 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

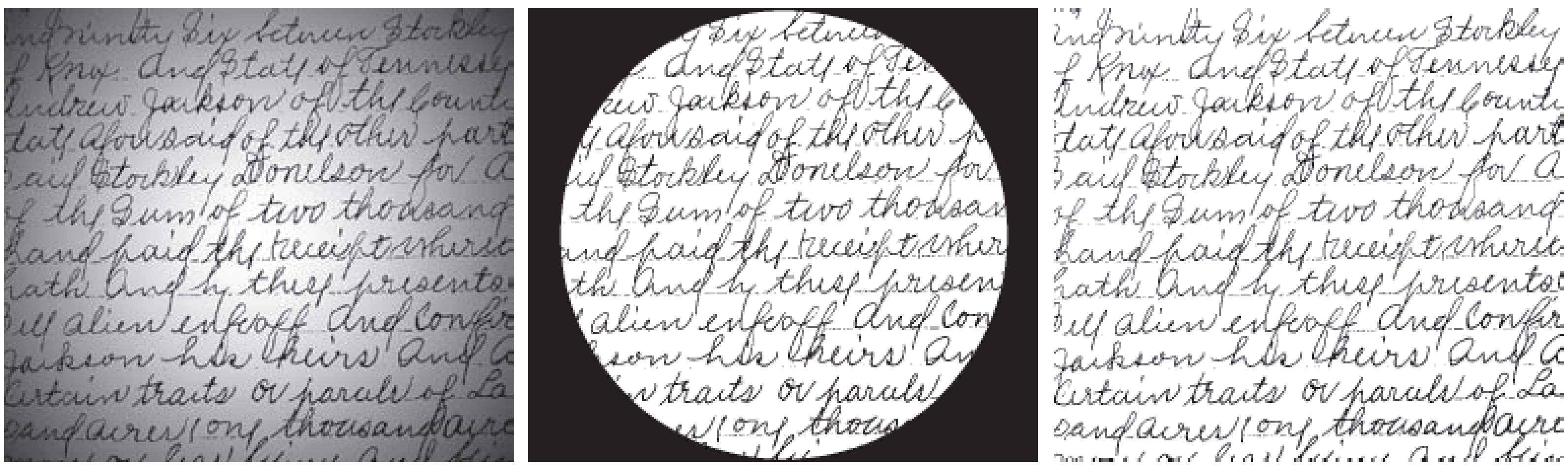
$$m(k+1) = \frac{1}{n}\sum_{i=k+2-n}^{k+1} z_i \qquad \text{for } k \geq n-1$$

$$= m(k) + \frac{1}{n}\left(z_{k+1} - z_{k-n}\right) \qquad \text{for } k \geq n+1$$

- where n is the number of points used in computing the average, and $m(1)= z1$

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

Example



a b c

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

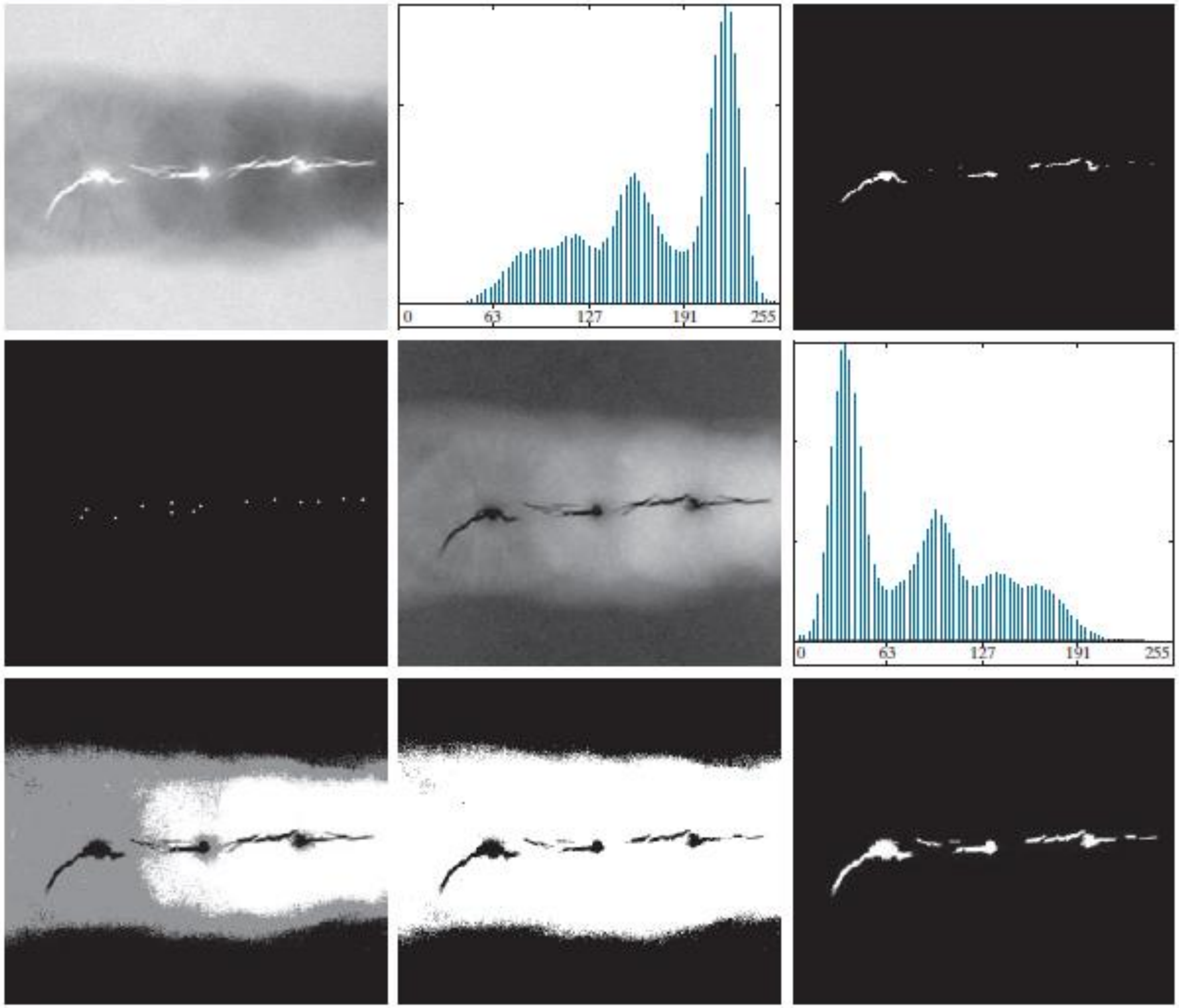
REGION GROWING

- *Region growing* is a procedure that groups pixels or subregions into larger regions based on predefined criteria for growth.
- The basic approach is to start with a set of "seed" points,
- Grow regions by appending to each seed those neighboring pixels that have predefined properties similar to the seed (such as ranges of intensity or color).

Segmentation by region growing

1. Find all connected components in $S(x, y)$ and reduce 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 each point (x, y), $f_Q(x, y) = 1$ if the input image satisfies a given predicate, Q, at those coordinates, and $f_Q(x, y) = 0$ otherwise.
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.,integers or letters). This is the segmented image obtained by region growing.

The following example illustrates the mechanics of this algorithm.



a b c
d e f
g h i

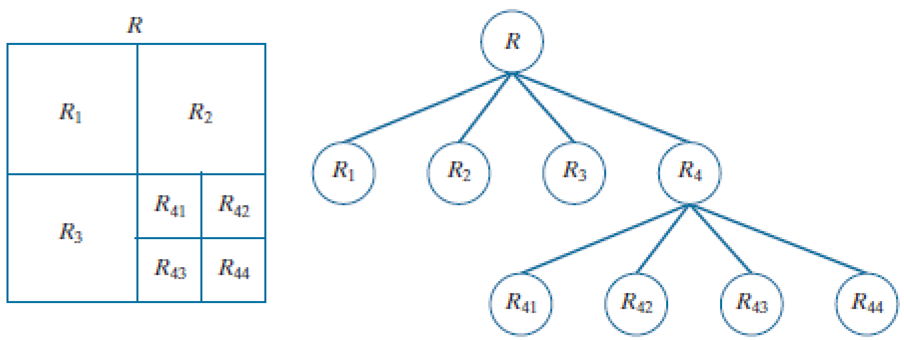
Figure 10.46 (a) X-ray image of a defective weld. (b) Histogram. (c) Initial seed image. (d) Final seed image (the points were enlarged for clarity). (e) Absolute value of the difference between the seed value (255) and (a). (f) Histogram of (e). (g) Difference image thresholded using dual thresholds.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

Read Gonzales pg 767

REGION SPLITTING AND MERGING

- An alternative is to subdivide an image initially into a set of disjoint regions and then merge and/or split the regions in an attempt to satisfy the conditions of segmentation

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 $Q(R_j \cup R_k) = \text{TRUE}$.

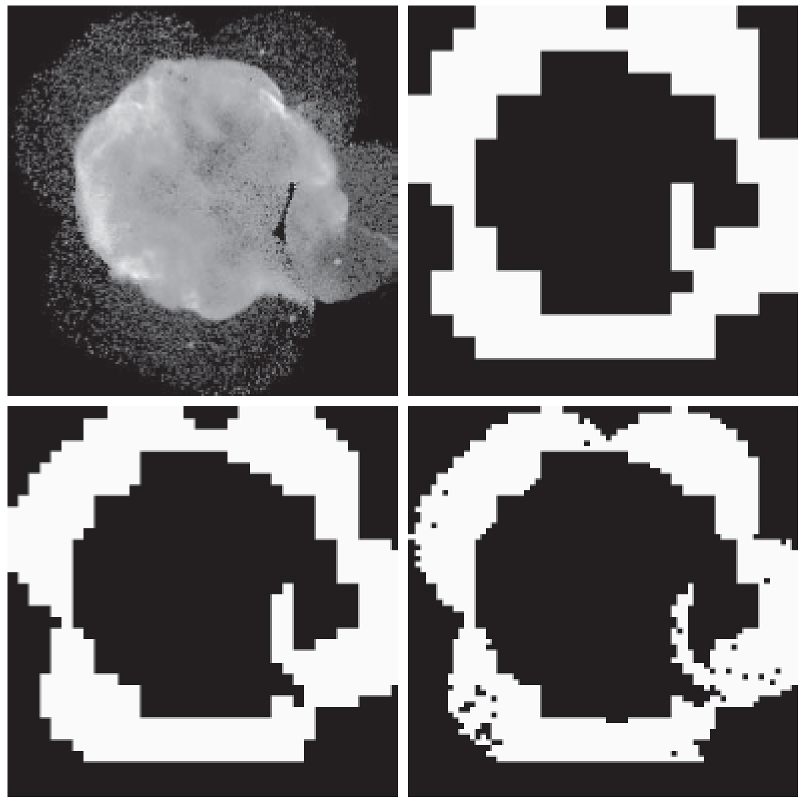


Read Gonzales pg 768

REGION SPLITTING AND MERGING

a b
c d

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



Read Gonzales pg 768-70