Lecture 9
Point, Line, Edge Detection, and Image Segmentation - Part 2

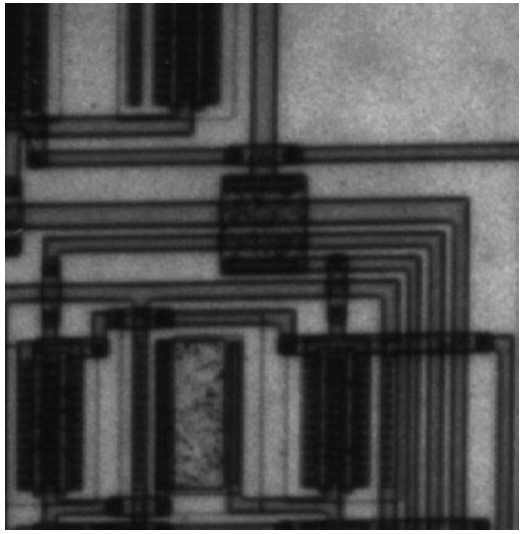
Instructor: Yiming Xiao

Email: yiming.xiao@concordia.ca

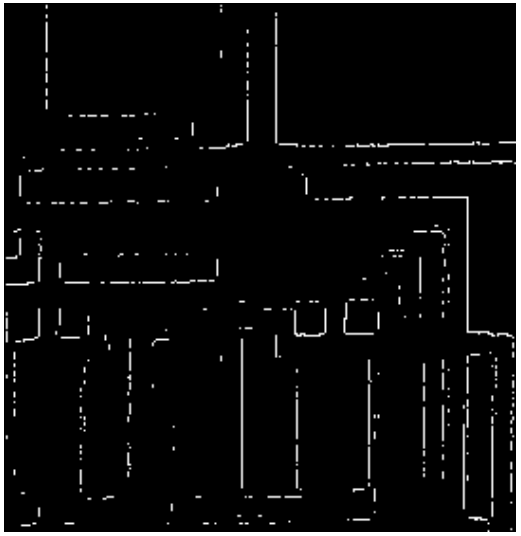
Department of Computer Science and Software Engineering
Concordia University

Slides modified from materials provided by Dr. Tien D Bui

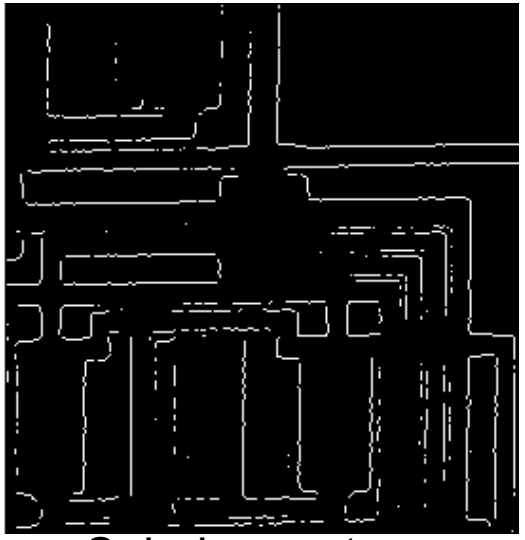
Edge detectors



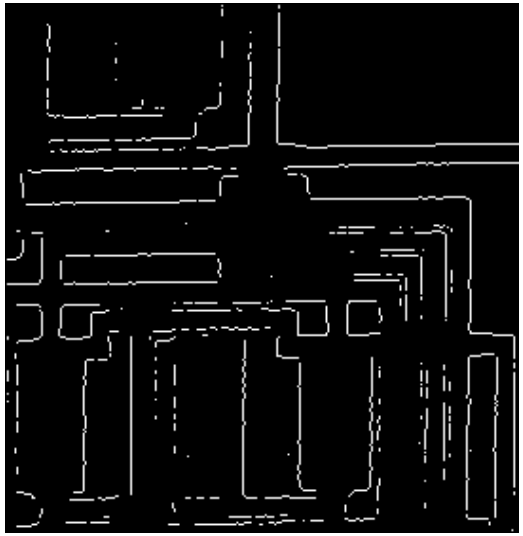
Original image



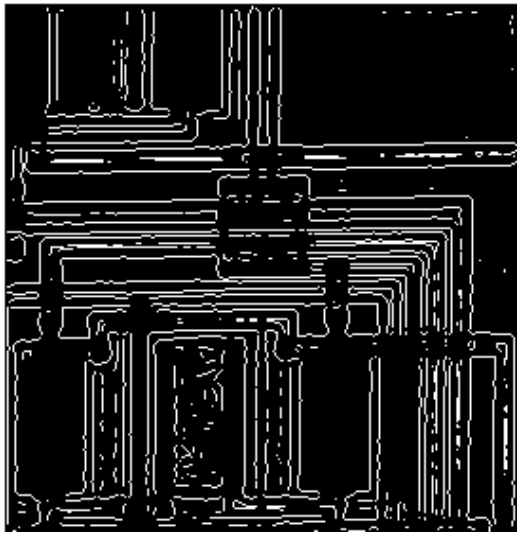
Roberts operator



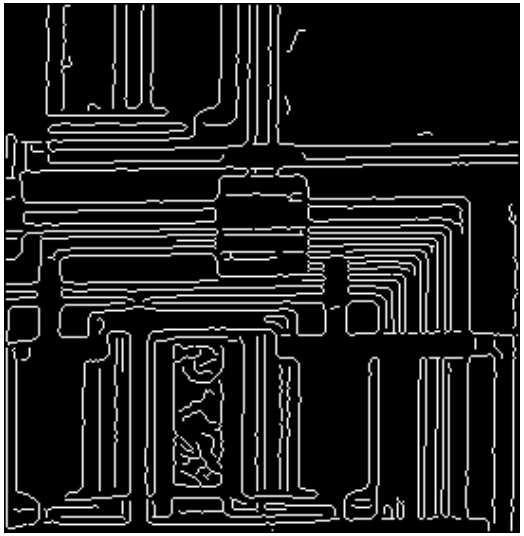
Sobel operator



Prewitt operator

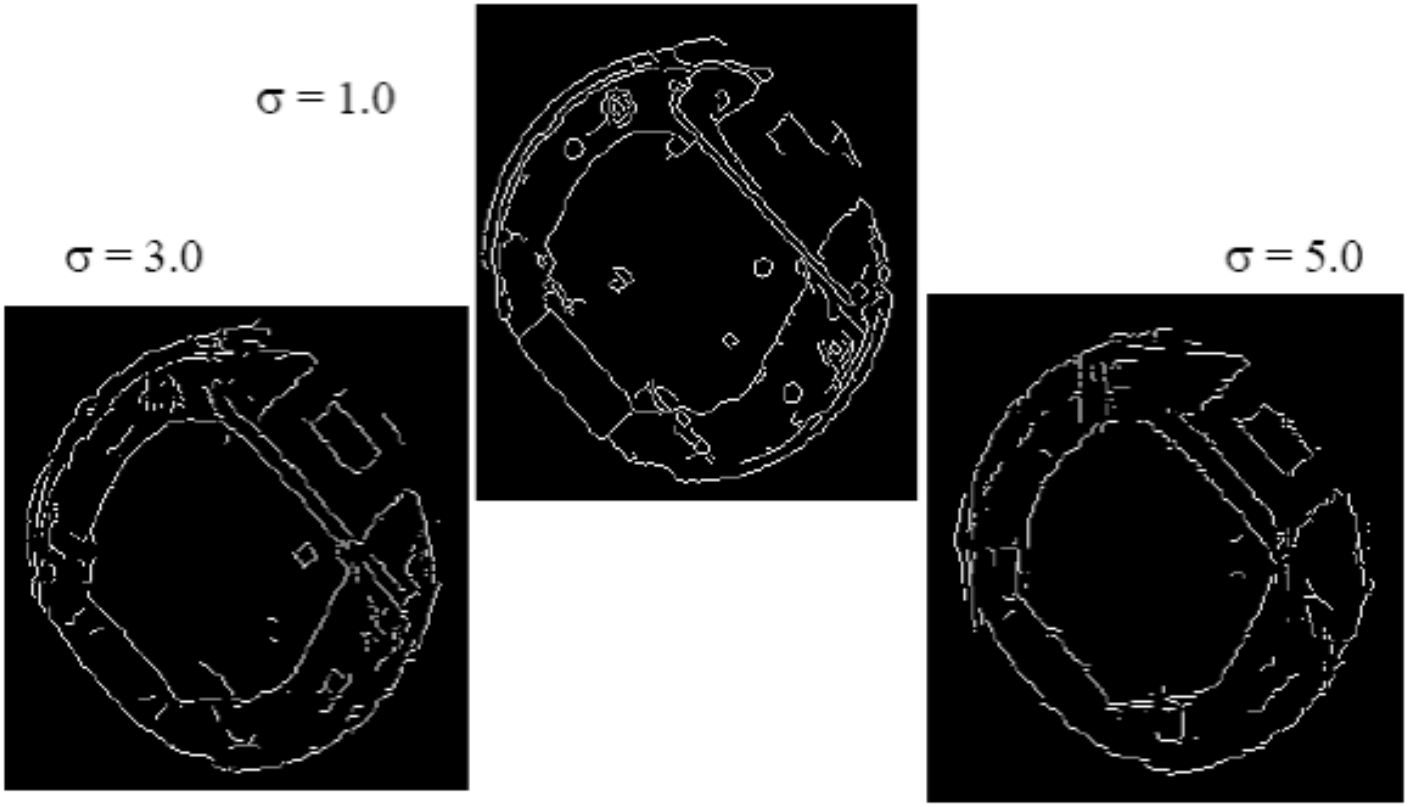


Laplacian operator



Canny

Scale Selection



Edge Linking and Boundary Detection

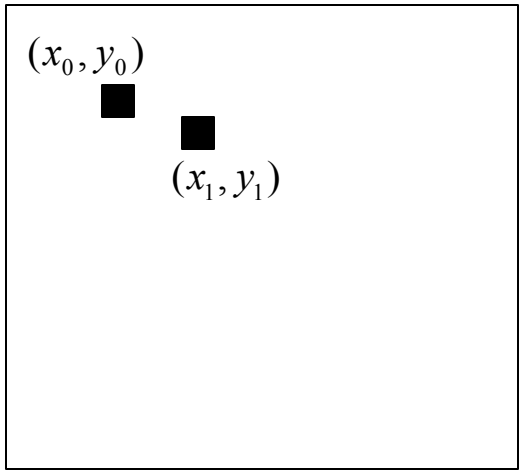
- Ideally edge detection should give sets of pixels lying only on the edges. In practice, due to noise or illumination, there are breaks in the edges.
- Hence edge detection often is followed by linking algorithms designed to assemble edge pixels into meaningful edges.
- Fundamental approaches to edge linking:
  1. Local processing: use local information (e.g., 3x3 region)
  2. Global processing: Hough transform

Edge Linking

- Edge detection operators identify pixels individually as belonging to an edge or not
  - *Need to connect points thus identified to recover a proper edge.*
- Suppose two nearby points are identified as edge pixels.
- Compare gradients: points on the same edge have similar gradients magnitude and angles

Magnitude Match $\left|\nabla f(x_0, y_0)\right| - \left|\nabla f(x_1, y_1)\right| < T_m$

Angular Match $\left|\angle\nabla f(x_0, y_0) - \angle\nabla f(x_1, y_1)\right| < T_a$



Local neighbourhood processing, with a moving window

Edge Linking

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:

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

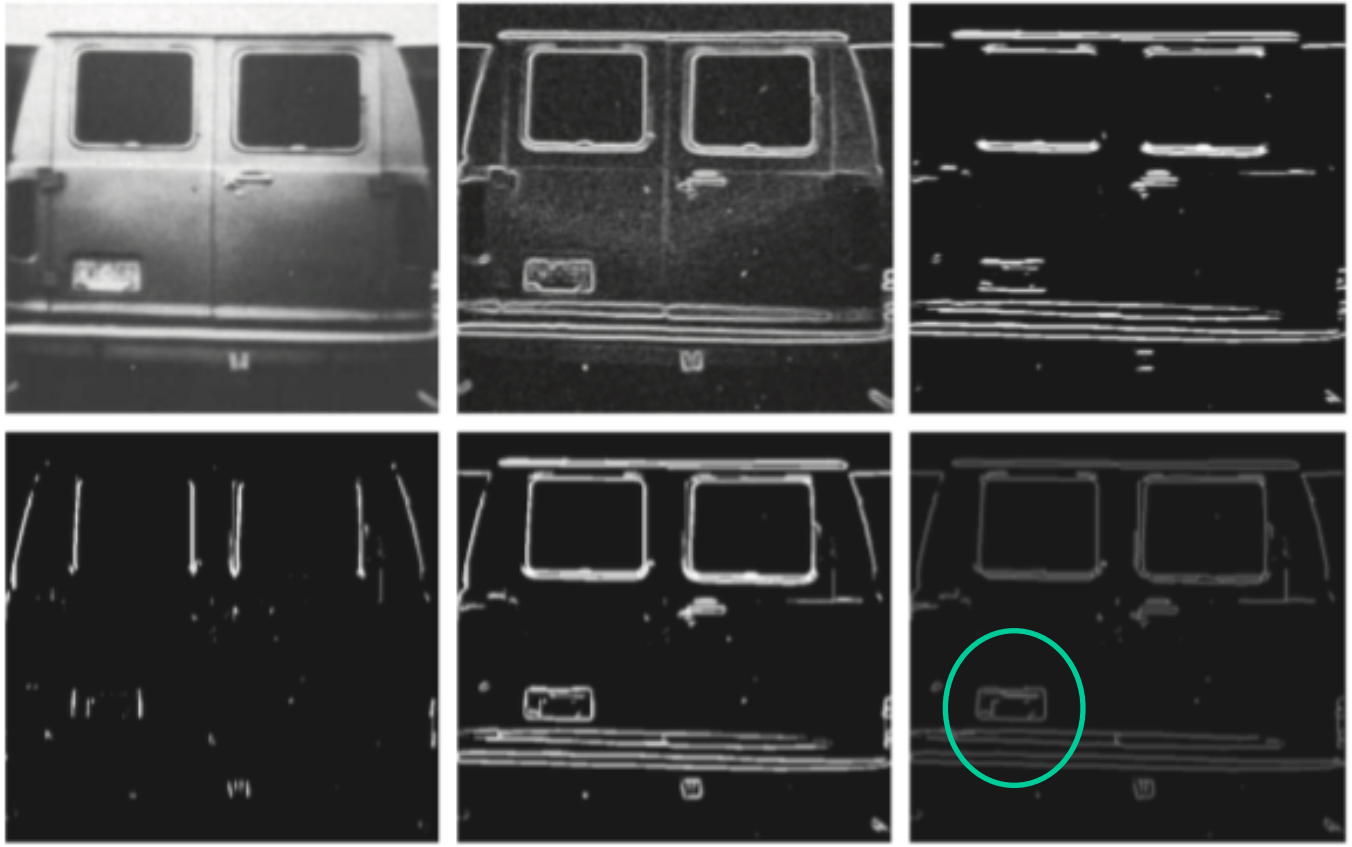
   where T_M is a threshold, A is a specified angle direction, and $\pm T_A$ defines a "band" of acceptable directions about A.
3. Scan the rows of g and fill (set to 1) all gaps (sets of 0's) in each row that do not exceed a specified length, L. Note that, by definition, a gap is bounded at both ends by 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

Example 10.10: Edge linking using local processing

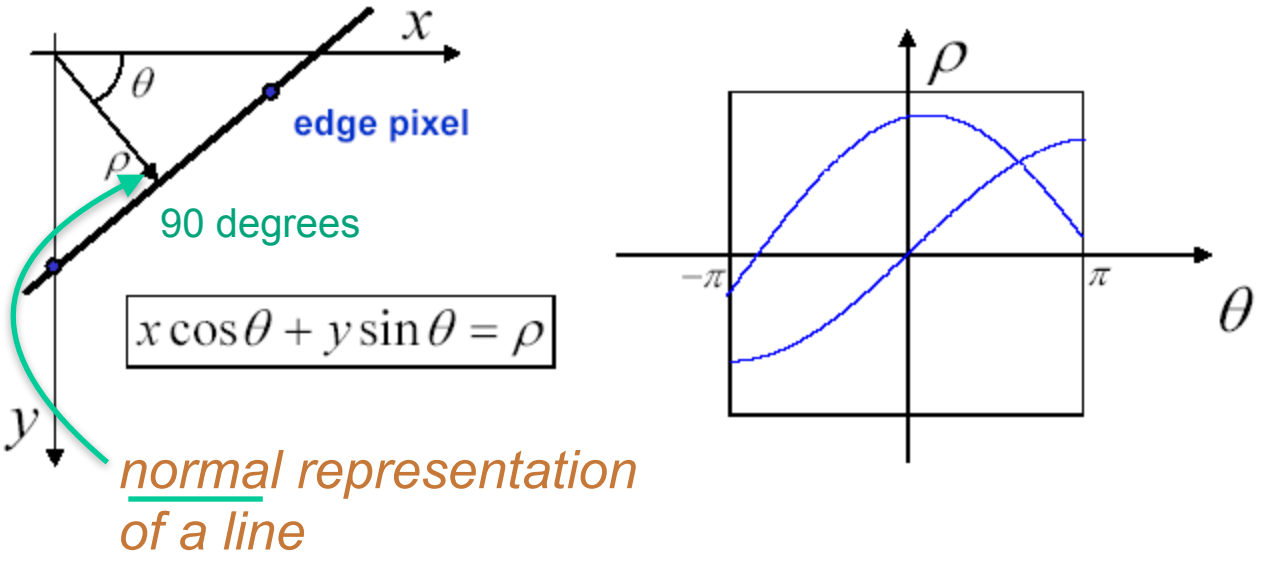
a b c
d e f

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.)

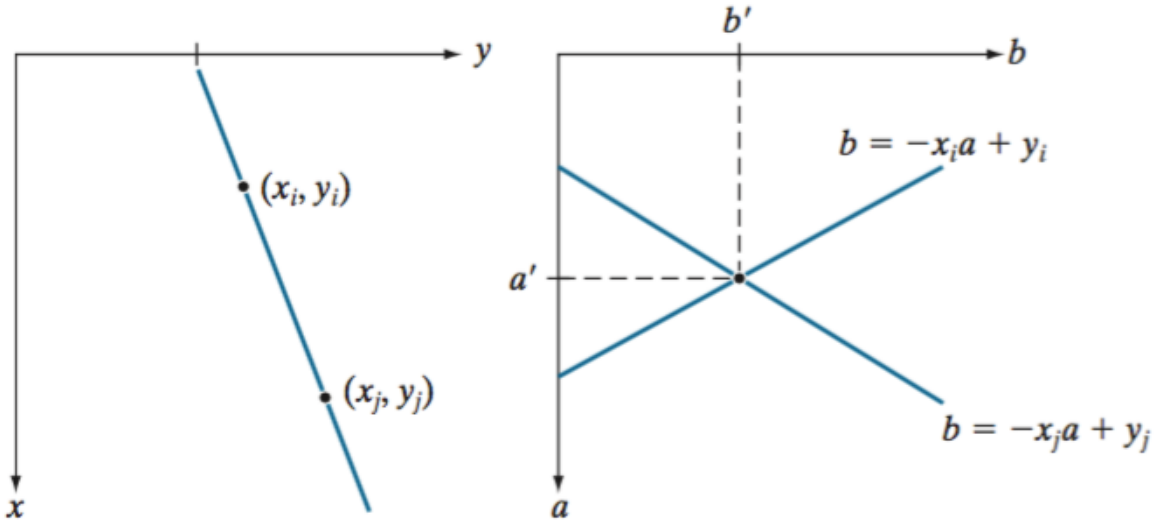


Fitting Point to a line

- Simplest case: Lines
  - Fit a straight line to a set of edge pixels.
  - Hough Transform (1962): Pattern Matching
  - Parameterization of a line in the plane:



Hough Transform

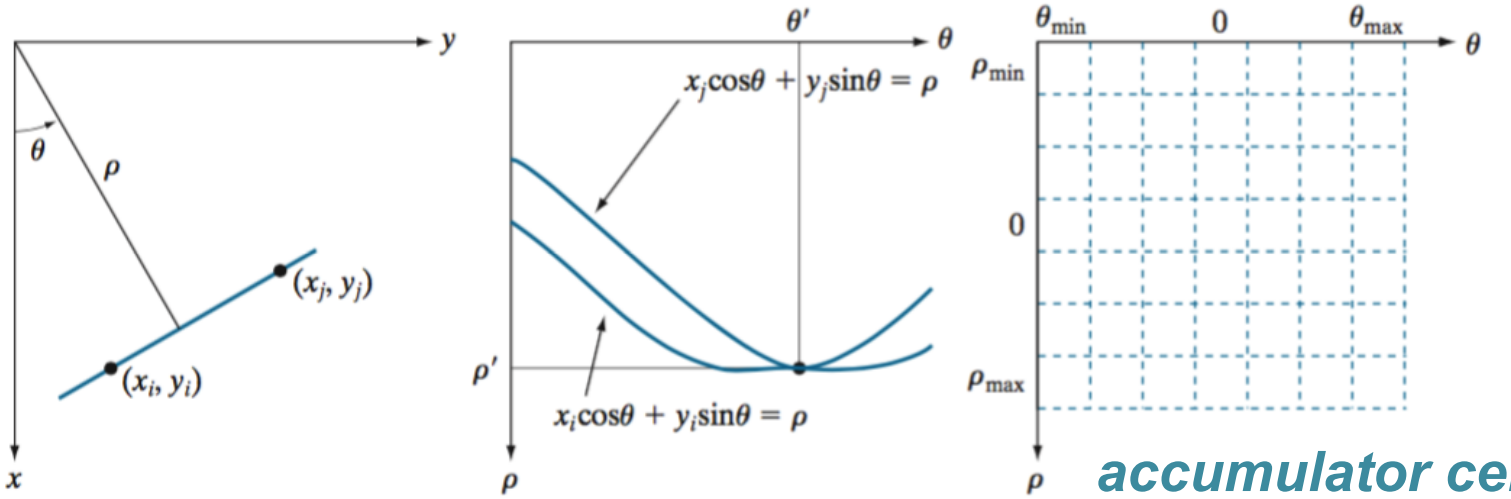


a b

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

- A line can be expressed as: $b = -ax + y$
- We can find a line using the ab-parameter space.
- The issue: the slope a can approach infinity when the line is near the vertical direction

Hough Transform



a b c

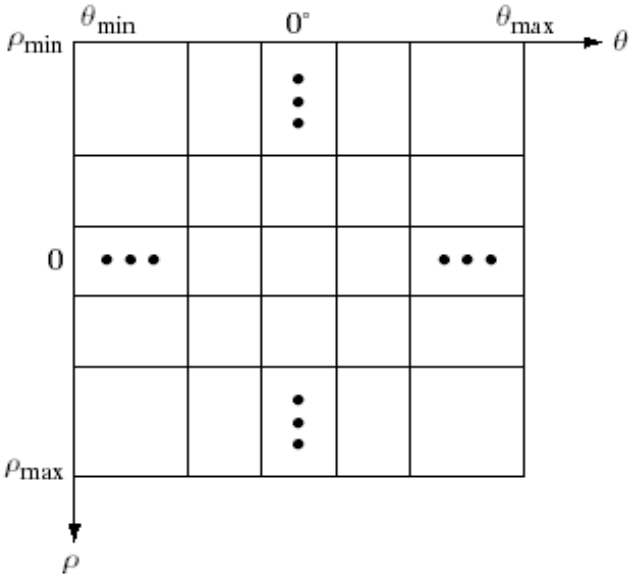
FIGURE 10.29 (a) (ρ,θ) parameterization of a line in the xy-plane. (b) Sinusoidal curves in the $\rho\theta$-plane; the point of intersection (ρ',θ') corresponds to the line passing through points (x_i,y_i) and (x_j,y_j) in the xy-plane. (c) Division of the $\rho\theta$-plane into accumulator cells.

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

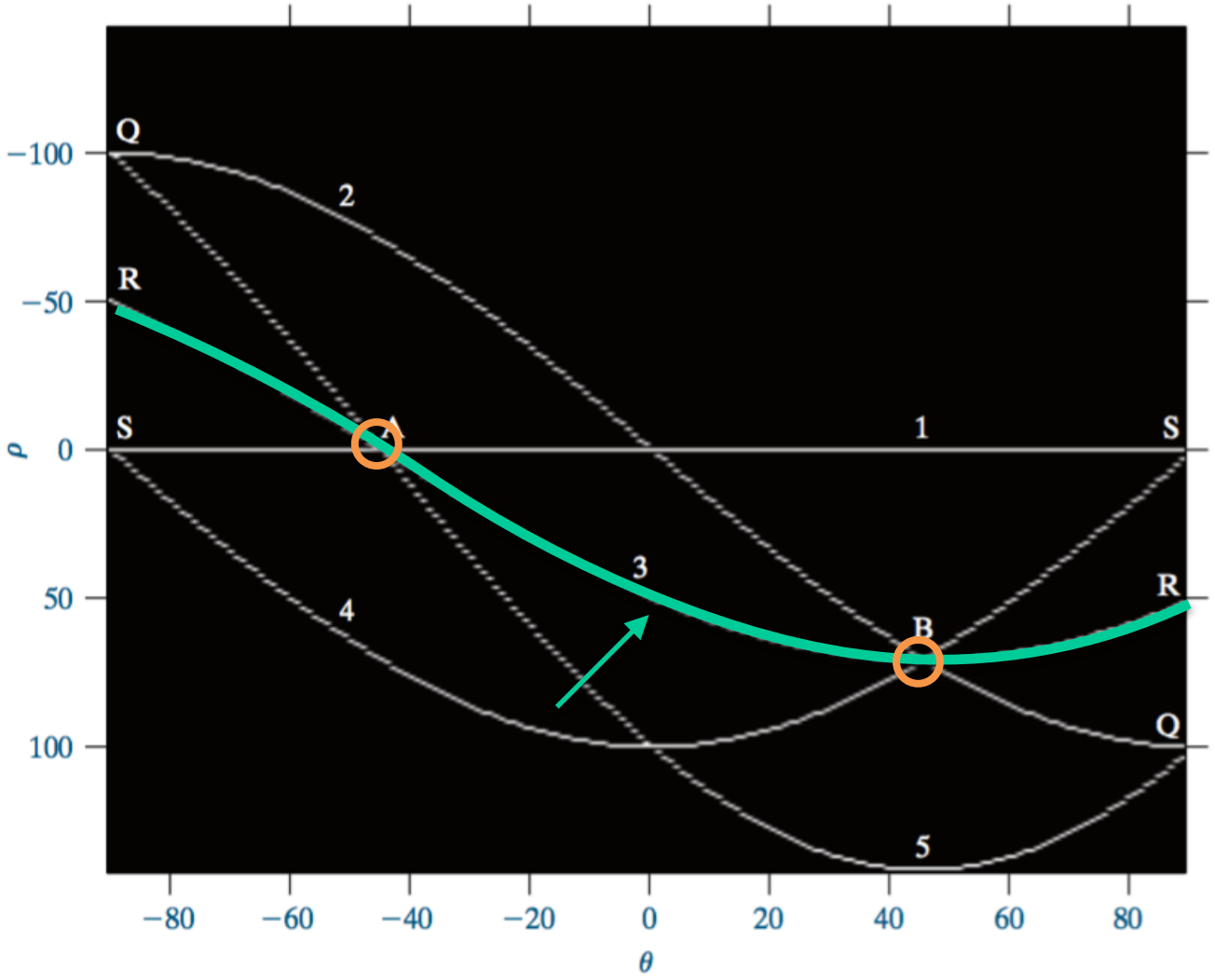
- The normal representation handles the infinitive slope issue
- The line is also found by the crossing of two curves

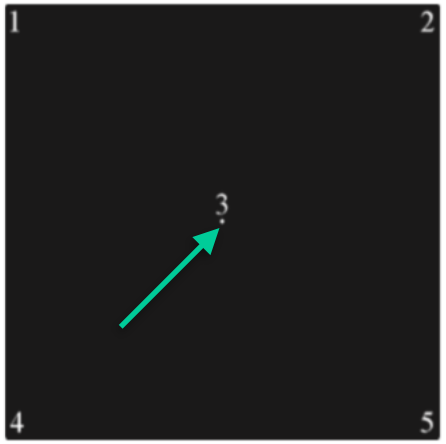
Hough Transform Algorithm

- Subdivide the parameter plane into bins
- Accumulate the total number of sinusoids that cross each bin
- Threshold the value of the bins in the parameter plane to declare the presence of lines
  - Note the effect of the length of lines on the accumulated values
- Note the similarity to Randon transform



Hough Transform

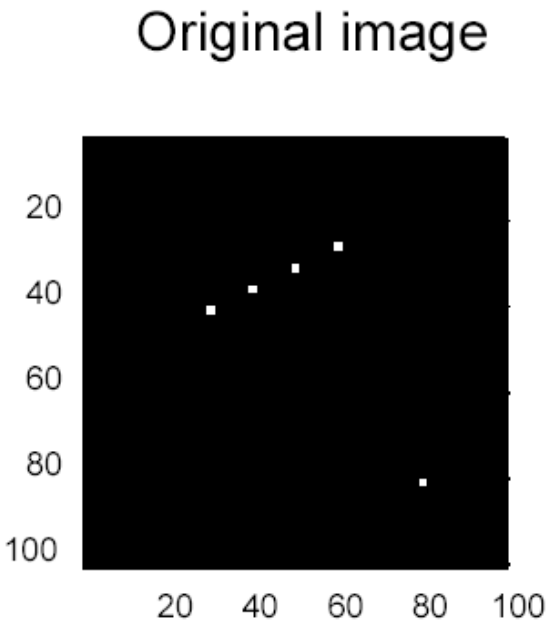


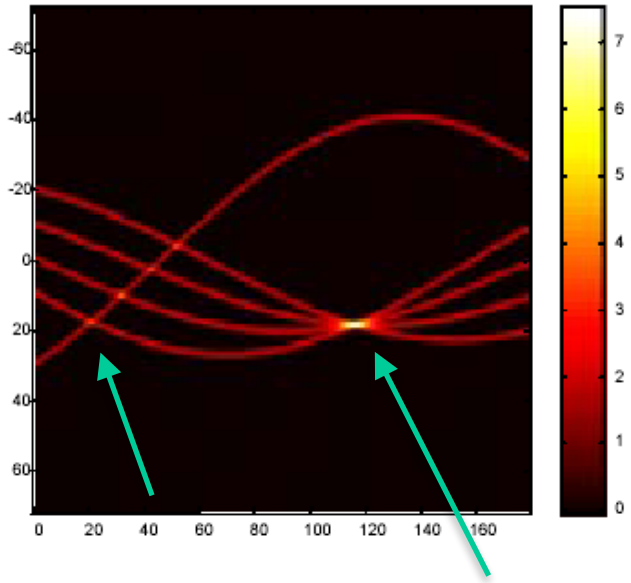


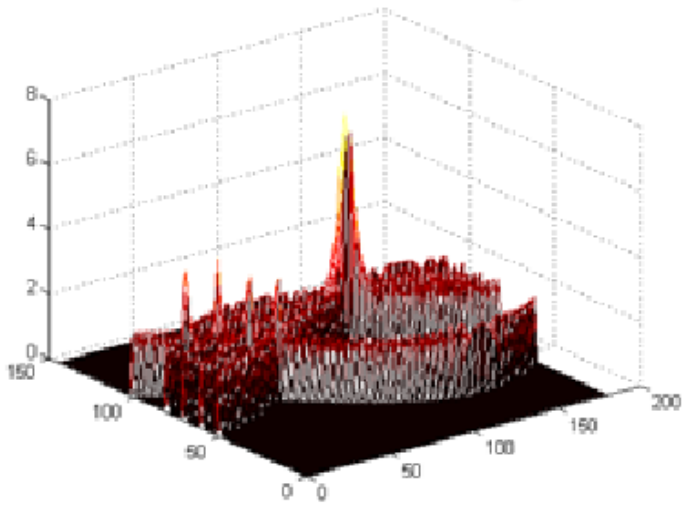
a
b

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

Hough Transform Examples

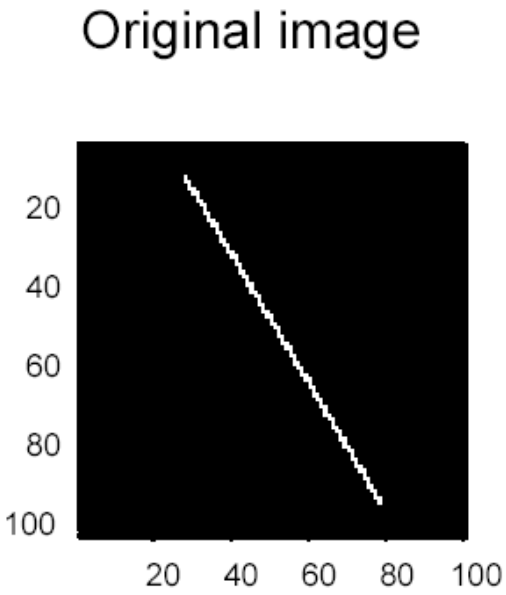


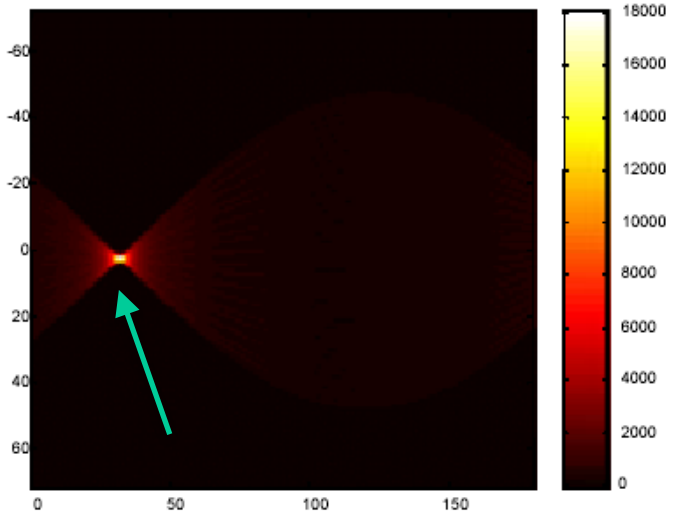


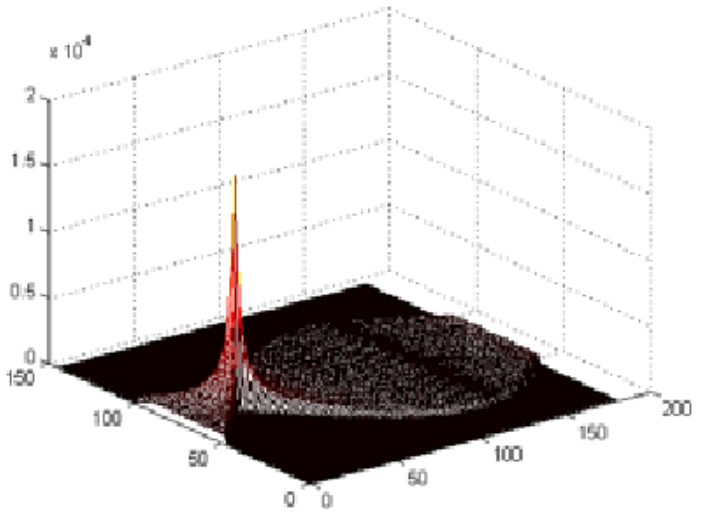


Courtesy: P. Salembier

Hough Transform Examples

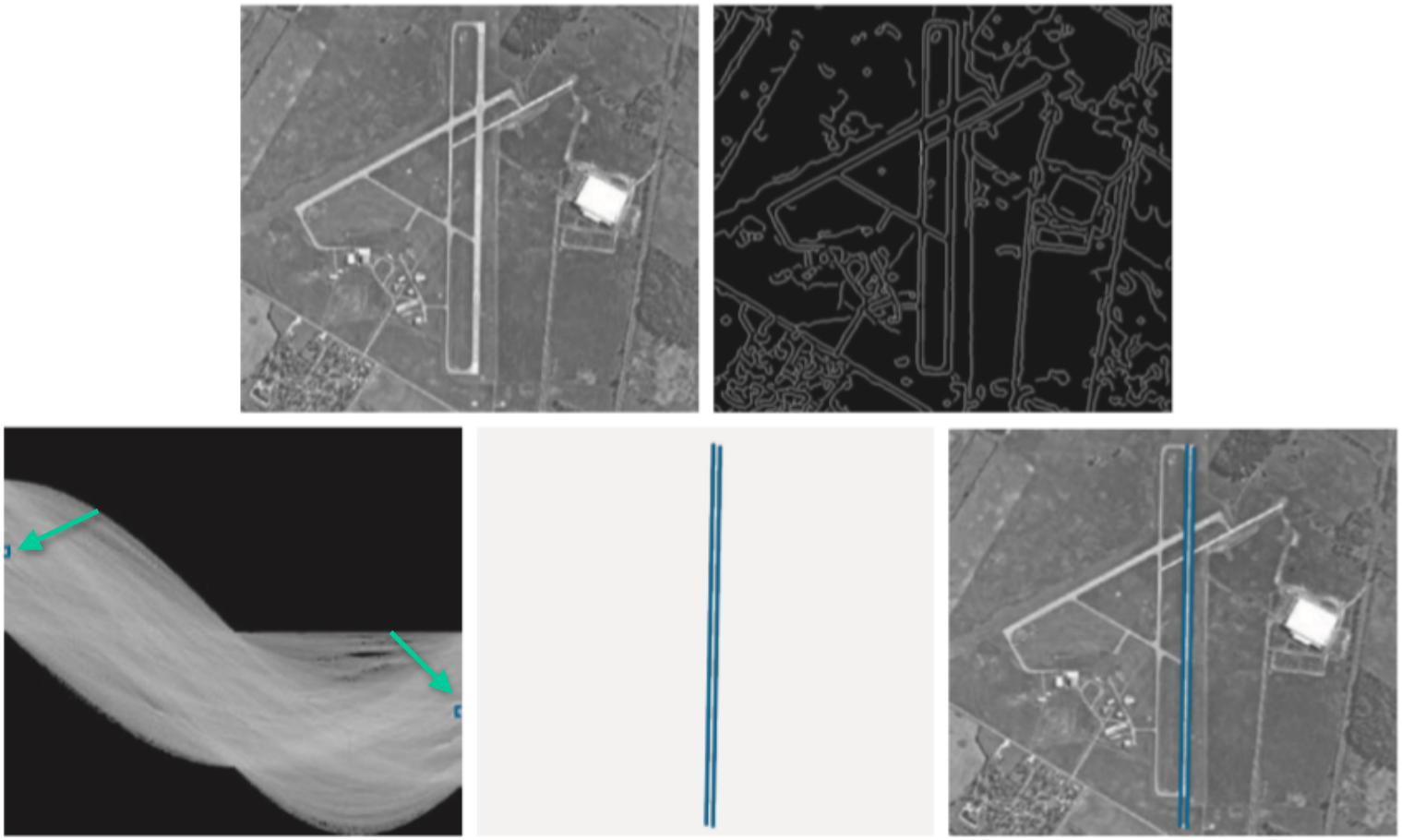






Courtesy: P. Salembier

Hough Transform



a b
c d e

FIGURE 10.31 (a) A 502×564 aerial image of an airport. (b) Edge map obtained using Canny's algorithm. (c) Hough parameter space (the boxes highlight the points associated with long vertical lines). (d) Lines in the image plane corresponding to the points highlighted by the boxes. (e) Lines superimposed on the original image.

Parameters for analytic curves

Form	Parameters	Equation
Line	ρ, θ	$x\cos\theta+y\sin\theta=\rho$
Circle	x_0, y_0, ρ	$(x-x_o)^2+(y-y_0)^2=\rho^2$
Parabola	x_0, y_0, ρ, θ	$(y-y_0)^2=4\rho(x-x_o)$
Ellipse	x_0, y_0, a, b, θ	$(x-x_o)^2/a^2+(y-y_0)^2/b^2=1$

Hough Transform

Line Detection using Hough Transform in MATLAB

https://www.youtube.com/watch?v=zy0932i2yog

How circle Hough Transform works

https://www.youtube.com/watch?v=Ltqt24SQQoI

Take a break!

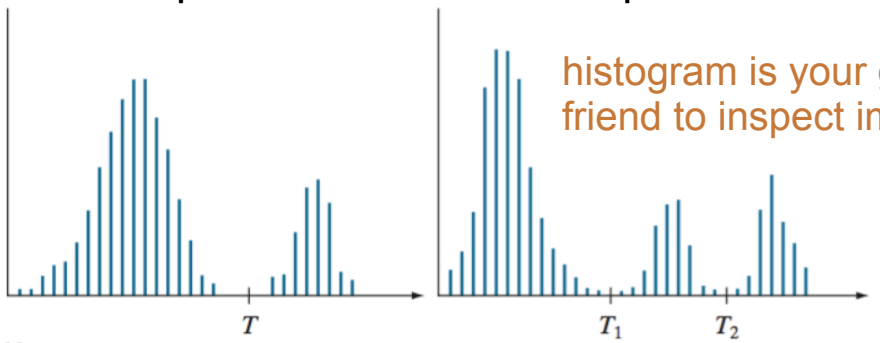
Segmentation by Thresholding

2 components

3 components

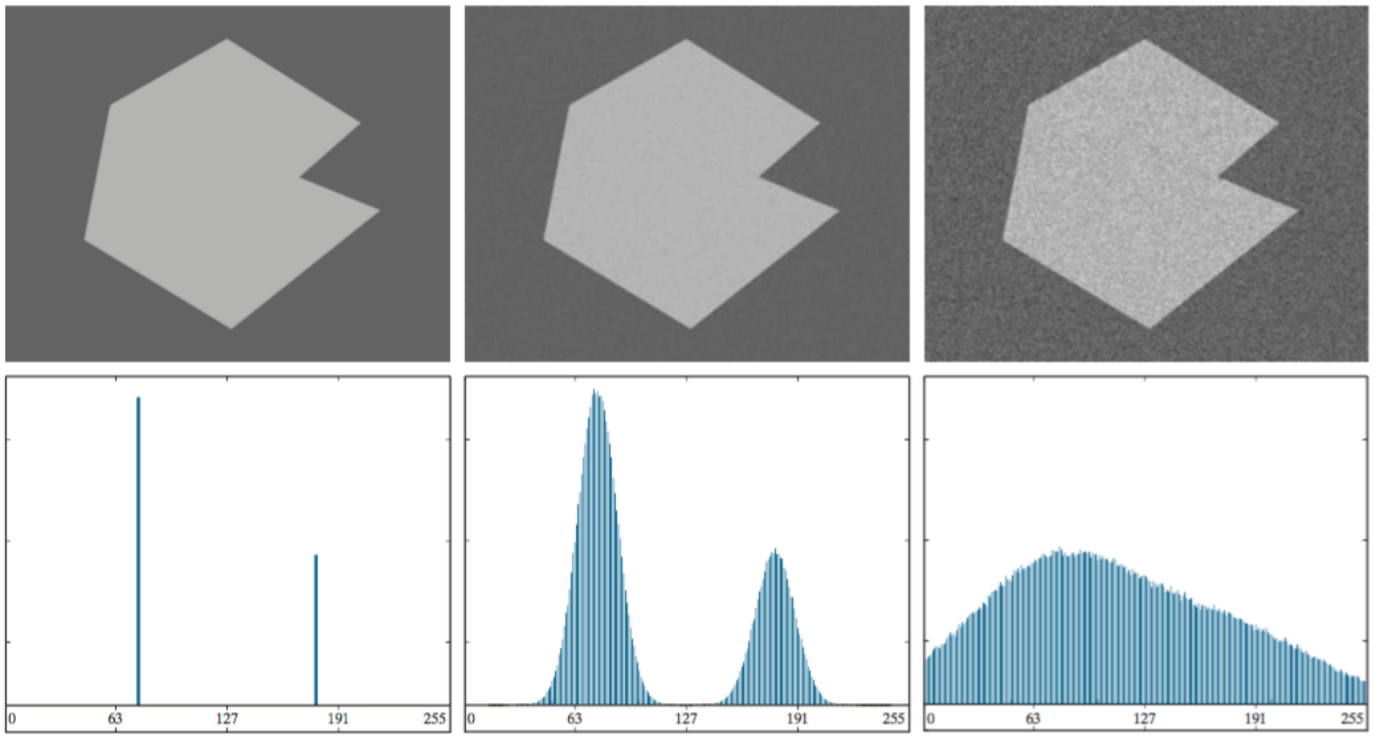
a b

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

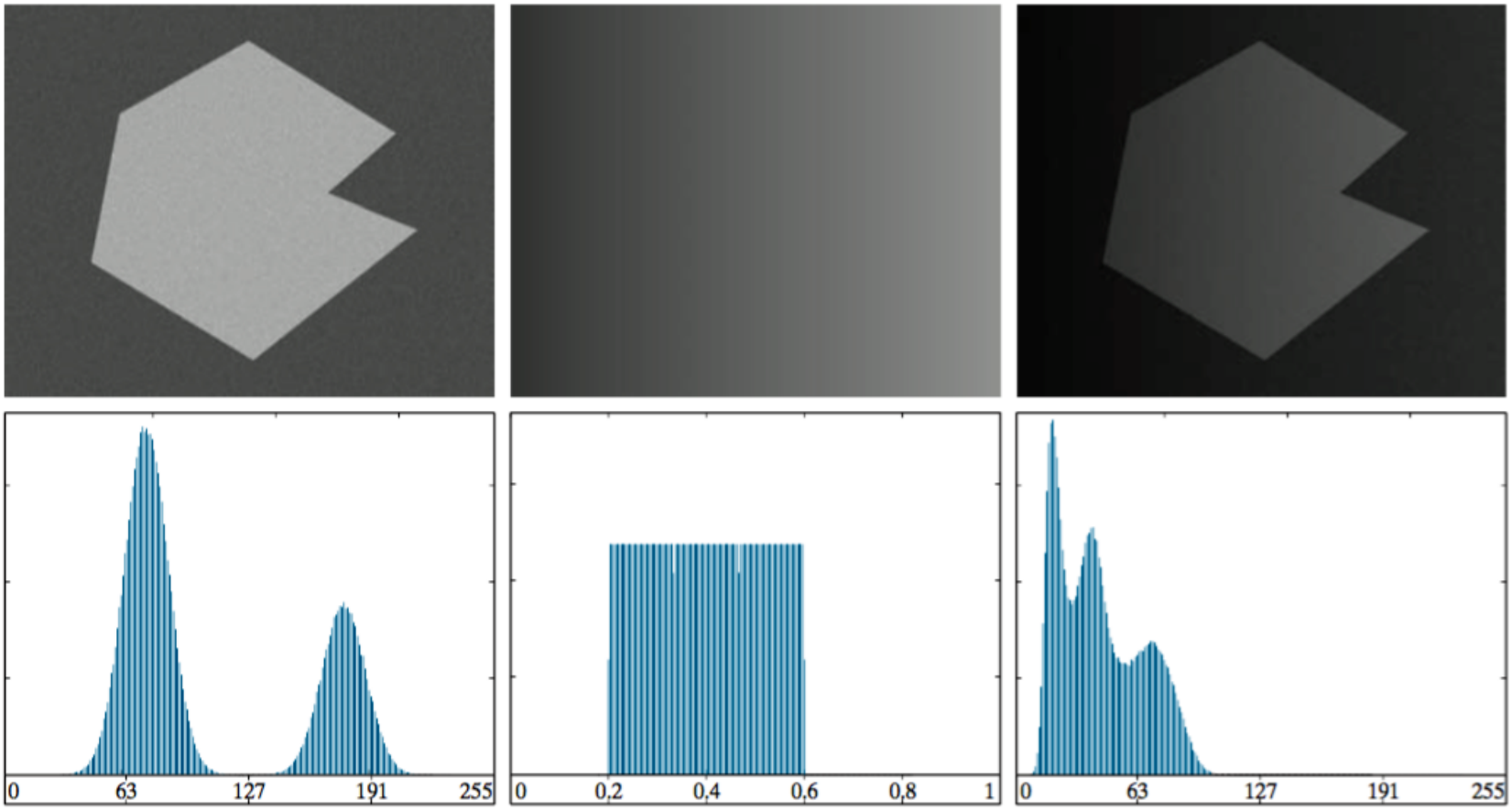


histogram is your great friend to inspect image properties

Noise effect on thresholding



Illumination effect on 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.

Basic global threshold

Eq 10-46

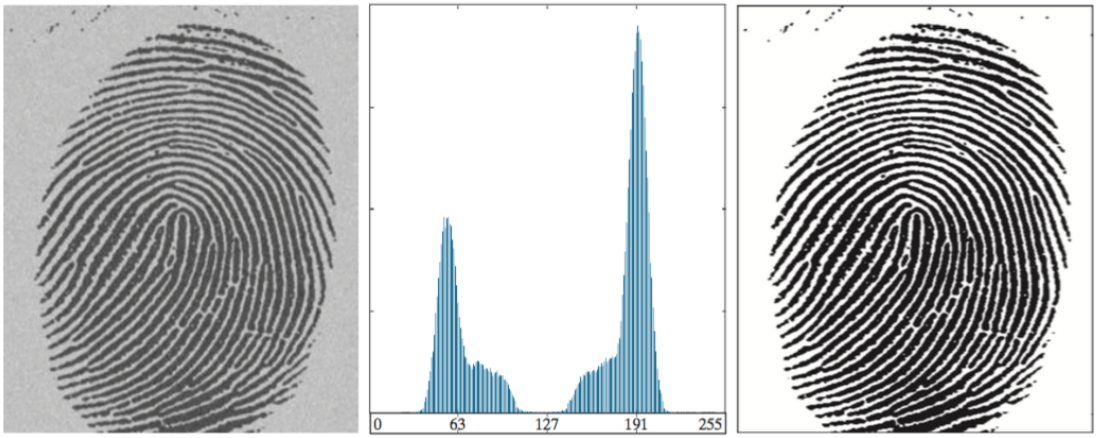
$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{if } f(x,y) \le T \end{cases}$$

1. Select an initial estimate for the global threshold, T.
2. Segment the image using T in Eq. (10-46). This will produce two groups of pixels: G_1, consisting of pixels with intensity values $> T$; and G_2, consisting of pixels with values $\le T$.
3. Compute the average (mean) intensity values m_1 and m_2 for the pixels in G_1 and G_2, respectively.
4. Compute a new threshold value midway between m_1 and m_2:

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

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

iterative solution

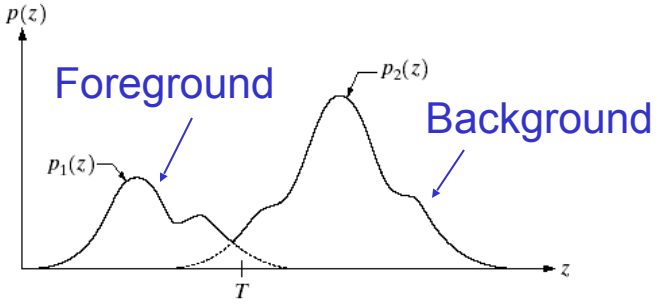


a b c

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

Some Statistical Analysis

- Histograms are almost never so nicely separated.



- Suppose each pixel belongs to foreground with probability ***a***, and to background with probability ***1-a***

- In classifying any pixel with threshold T, we can make two types of errors.

Background

$$e_1(T) = \int_{-\infty}^{T} p_2(z)dz$$

Error of misclassifying
a background pixel as foreground

Foreground

$$e_2(T) = \int_{T}^{\infty} p_1(z)dz$$

Error of misclassifying
a foreground pixel as background

Optimal Global Thresholding

- Pick threshold T to minimize the overall expected probability of error.

$$e(T) = (1-a)e_1(T) + ae_2(T)$$

$$\frac{\partial e(T)}{\partial T} = (1-a)\frac{\partial}{\partial T}\int_{-\infty}^{T} p_2(z)dz + a\frac{\partial}{\partial T}\int_{T}^{\infty} p_1(z)dz$$

$$= (1-a)p_2(T) - ap_1(T) = 0$$

$$\Rightarrow \quad \frac{p_1(T)}{p_2(T)} = \frac{1-a}{a}$$

Solve for threshold T

- Example: Two Gaussians with same variance and different means.

$$T = \frac{m_1 + m_2}{2} + \frac{\sigma^2}{m_1 - m_2}\ln\left(\frac{a}{1-a}\right)$$

Special case: a=1/2

$$T = \frac{m_1 + m_2}{2}$$

Otsu's Method

Let $\{0, 1, 2, \ldots, L-1\}$ be the L distinct intensity levels

n_i be the number of pixels with intensity i

The normalized histogram has components : $p_i = n_i / MN$

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

Let $T(k) = k, \quad 0 < k < L-1$ as threshold and use it to divide image into two classes $C_1 : [0, k]$ and $C_2 : [k+1, L-1]$.

With threshold the prob. that a pixel is assigned to C_1 is :

$$P_1(k) = \sum_{i=0}^{k} p_i$$

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

Otsu's Method

Mean intensity of the pixels assigned to class C_1

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

$P(C_1 \mid i) = 1;\ P(i) = p_i$; and $P(C_1) = P_1(k)$

Similarly: $m_2(k) = \frac{1}{P_2(k)} \sum_{i=k+1}^{L-1} ip_i$ for class C_2

The average intensity up to level k is:

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

The average intensity for the entire image (global mean) is:

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

Direct substitution: $P_1 m_1 + P_2 m_2 = m_G$ and $P_1 + P_2 = 1$

Otsu's Method

The goodness of the threshold at level k is by (dimensionless):

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

where σ_G^2 is the *global* variance: $\sigma_G^2 = \sum_{i=0}^{L-1} (i - m_G)^2 p_i$

and σ_B^2 is the *between-class* variance: $\sigma_B^2 = P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2$

$$\sigma_B^2 = P_1 P_2 (m_1 - m_2)^2 \quad (\text{Use: } P_1 m_1 + P_2 m_2 = m_G \text{ and } P_1 + P_2 = 1)$$

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

and $\sigma_B^2(k) = \dfrac{[m_G P_1(k) - m(k)]^2}{P_1(k)[1 - P_1(k)]}$ (expressed in P_1 only)

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

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

Otsu's Algorithm

1. Compute the normalized histogram of the input image:

   $p_i,\ i = 0, 1, 2, \ldots, L-1$

   $$p_i = n_i / MN$$

2. Compute the cumulative sums: $P_1(k), k = 0, 1, 2, \ldots, L-1$

   $$P_1(k) = \sum_{i=0}^{k} p_i$$

3. Compute the cumulative means: $m(k), k = 0, 1, 2, \ldots, L-1$

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

4. Compute the global intensity mean: m_G

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

5. Compute the between-class variance: $\sigma_B^2(k), k = 0, 1, 2, \ldots, L-1$

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

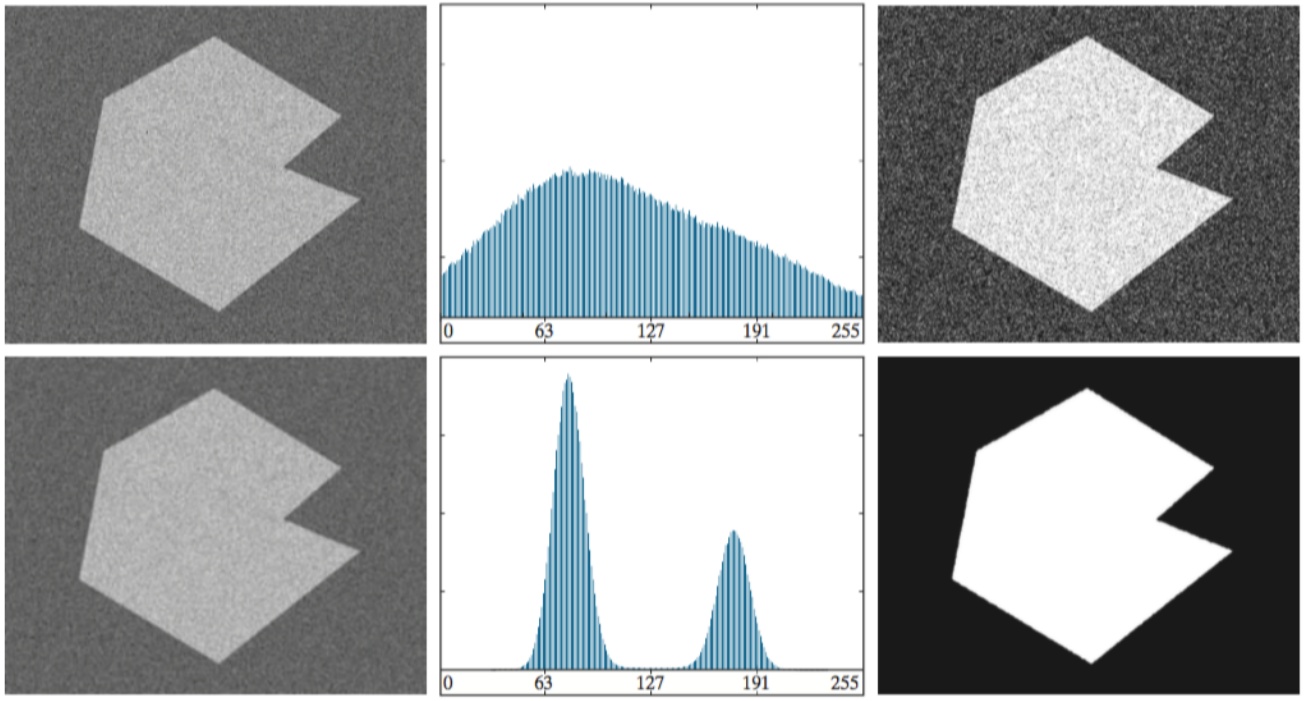
6. Obtain the Otsu's threshold: k^*

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

7. Obtain the separability measure: η^*

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

Otsu's Method



a b c
d e f

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

We have done a detailed study of Otsu's method for single threshold. It should be straightforward for students to read P747-753 in the textbook.

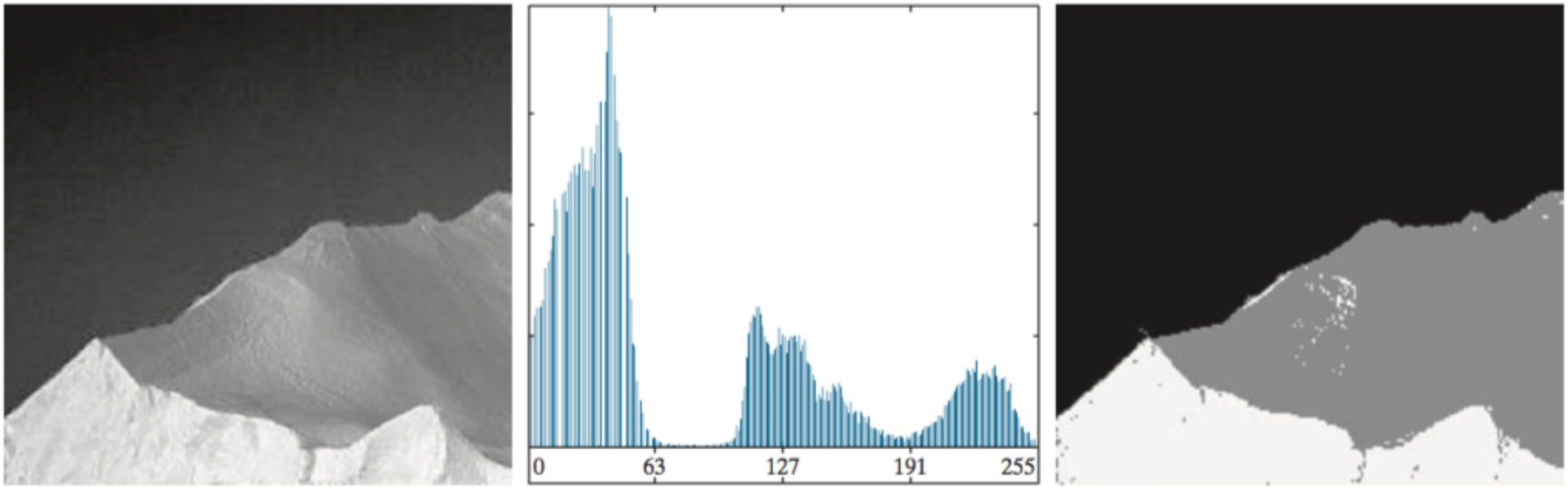
Otsu's method: Maximization of between-class variance

Otsu's Method

Otsu's method can be generalized to multiple thresholds.

$$\eta\left(k_1^*, k_2^*\right) = \frac{\sigma_B^2\left(k_1^*, k_2^*\right)}{\sigma_G^2} \qquad \sigma_B^2\left(k_1^*, k_2^*, \ldots, k_{K-1}^*\right) = \max_{0<k_1<k_2<\ldots k_K<L-1} \sigma_B^2\left(k_1, k_2, \ldots k_{K-1}\right)$$

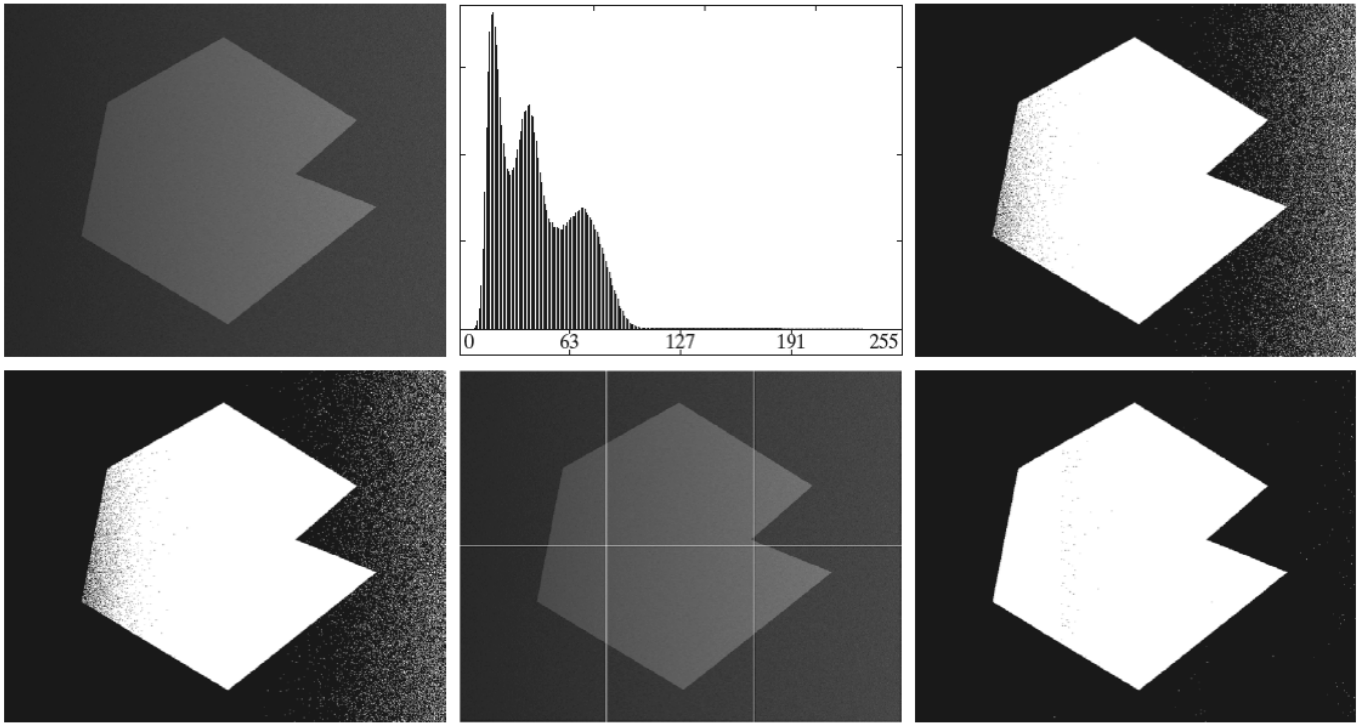
We can also perform multiple thresholds (P757-764)



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.)

Otsu's Method



a b c
d e f

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

Variable thresholding based on moving averages

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

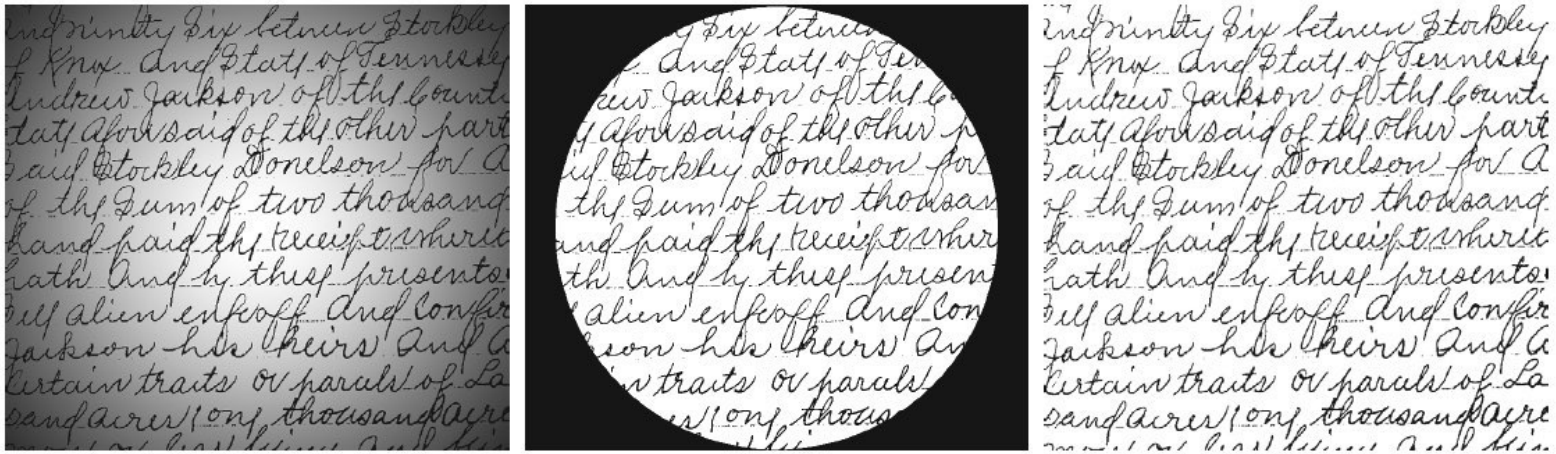
$$= m(k) + \frac{1}{n}\left(z_{k+1} - z_{k-n}\right) \qquad \text{for } k \geq n+1$$

Time saver

$$T_{xy} = cm_{xy}$$

c is a postive scalar

Line-by-line operation that is great for text processing



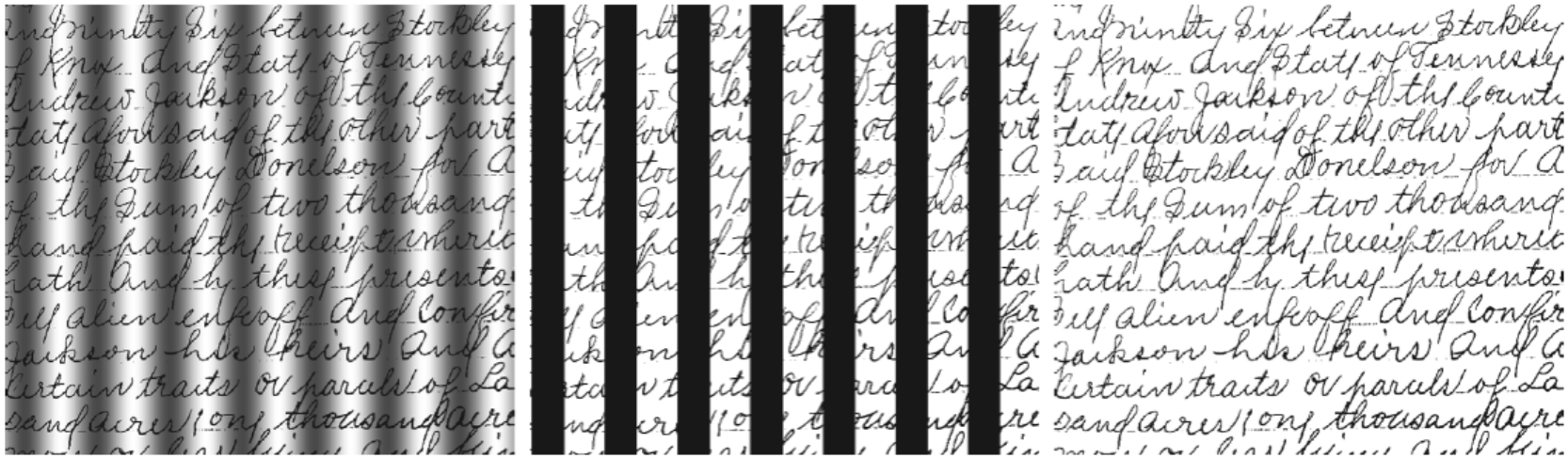
average width = 4 pixels, n=20, and c=0.5

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.

Variable thresholding based on moving averages

Otsu's method *moving averages*

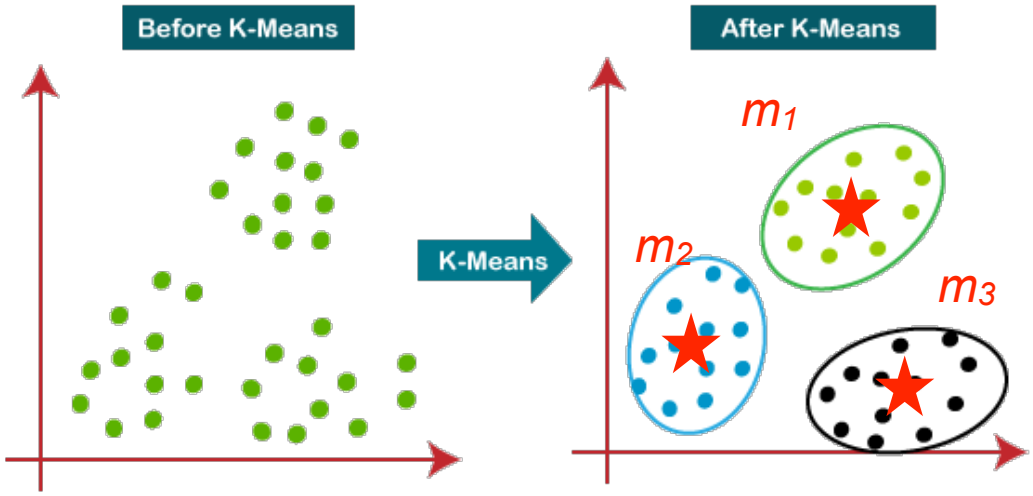


a b c

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.

Generally good for objects that are small and thin wrt. the image size

Region segmentation using k-means



- Unsupervised learning
- Group data into k clusters
- k is specified by the user
- Widely available in different software libraries

Steps

1. *Initialize the mean values (cluster centroids) of k clusters: m_1, m_2, m_3, etc.*
2. *Assign the data point to the nearest cluster centroids*
3. *Update the cluster centroids according to the assignments in Step 2*
4. *Repeat Step 2-3 till the centroids don't change beyond a threshold*

Region segmentation using k-means

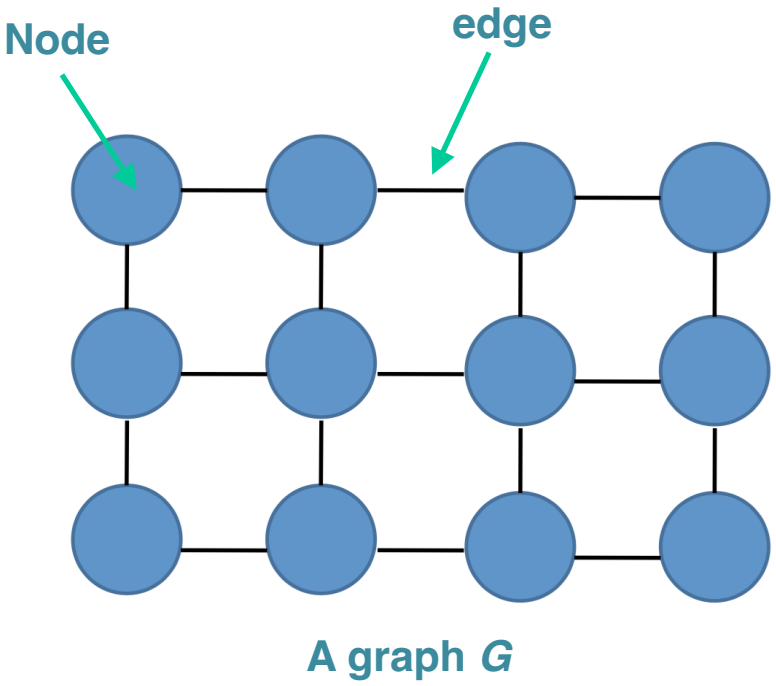
a b

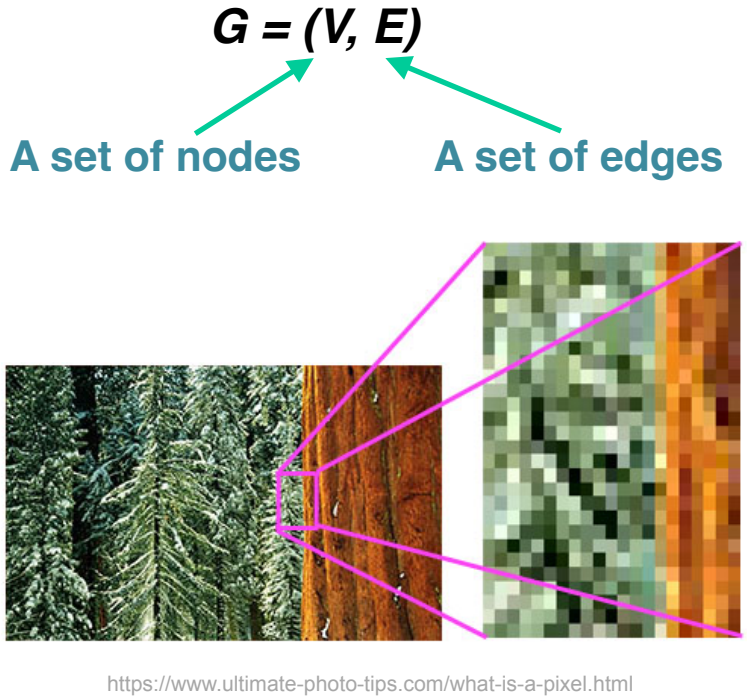
FIGURE 10.49
(a) Image of size 688 × 688 pixels. (b) Image segmented using the *k*-means algorithm with $k = 3$.

- Image intensities are also data points available for k-means classification
- Easy to implement and parameter setting is straightforward
- Generally works well with clusters that have approximately Gaussian distribution

Region segmentation using graph cuts

- Model an image as a graph ***G*** made with nodes
- Each pixel can be viewed a node, and is connected to its neighbours in an undirected graph through the edges
- Graph cuts intends to split the nodes into different sets based on the pixel values or image features (gradient etc.) at the pixel location



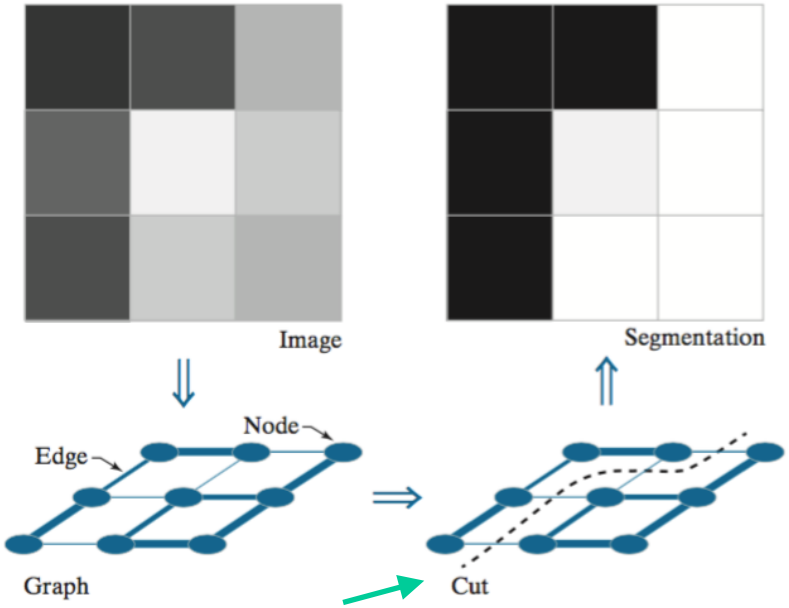


https://www.ultimate-photo-tips.com/what-is-a-pixel.html

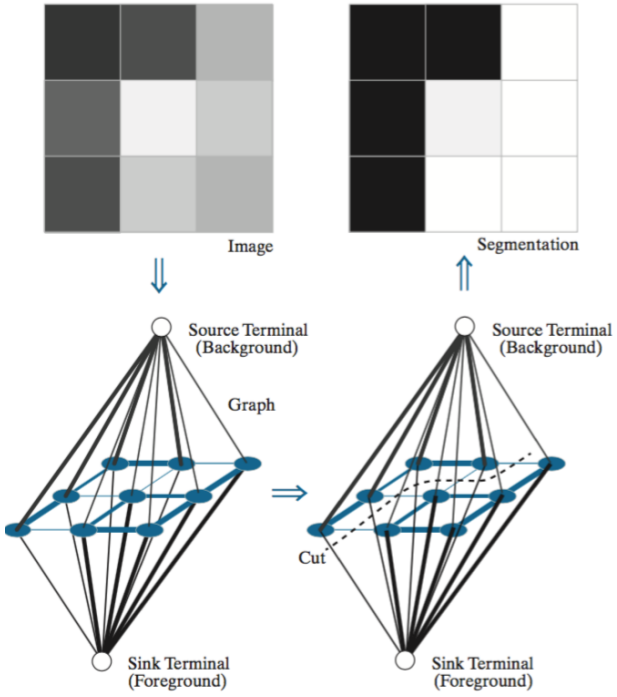
Region segmentation using graph cuts

- A source and a sink is defined to connect through the nodes
- *Minimum graph cut*: the smallest total weight of edges that, if removed, would disconnect the sink from the source

$$cut(A,B) = \sum_{u \in A, v \in B} w(u,v)$$,where $$w(i,j) = 1/(|I(n_i) - I(n_j)| + c)$$



Minimum cut

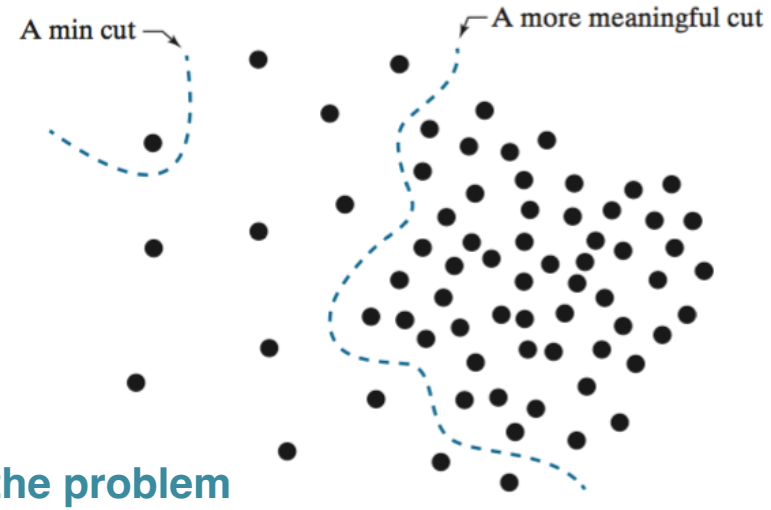


A flow network

There are lots of solutions…

Region segmentation using graph cuts

Finding more meaningful cut



Normalized cut (Ncut)

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$

Shi and Malik (2000)

Closed-form solution that approximate the problem

$$Ncut(A,B) = \frac{cut(A,B)}{cut(A,V)} + \frac{cut(A,B)}{cut(B,V)}$$

$$= \frac{\sum_{x_i>0,\, x_j<0} -w(i,j)x_i x_j}{\sum_{x_i>0} d_i} + \frac{\sum_{x_i<0,\, x_j>0} -w(i,j)x_i x_j}{\sum_{x_i<0} d_i}$$

$$d_i = \sum_j w(i,j)$$

$$(\mathbf{D} - \mathbf{W})\mathbf{y} = \lambda \mathbf{D}\mathbf{y}$$

Use eigenvector for the second smallest eigenvalue to determine the split that minimizes Ncut(A,B), using a threshold value

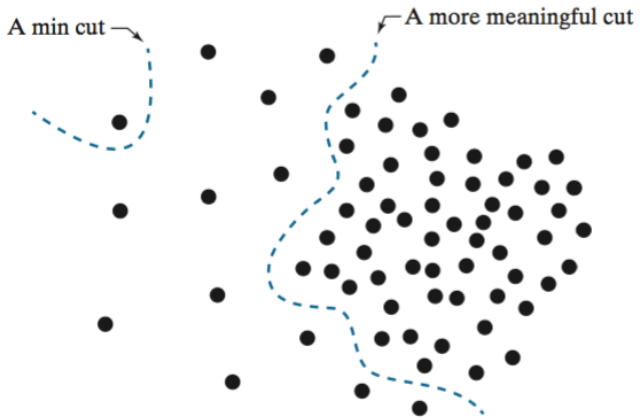
Region segmentation using graph cuts

Finding more meaningful cut

1. Given a set of features, specify a weighted graph, $G = (V, E)$ in which V contains the points in the feature space, and E contains the edges of the graph. Compute the edge weights and use them to construct matrices $\mathbf{W}$ and $\mathbf{D}$. Let K denote the desired number of partitions of the graph.
2. Solve the eigenvalue system $(\mathbf{D} - \mathbf{W})\mathbf{y} = \lambda\mathbf{D}\mathbf{y}$ to find the eigenvector with the second smallest eigenvalue.
3. Use the eigenvector from Step 2 to bipartition the graph by finding the splitting point such that $Ncut(A, B)$ is minimized.
4. If the number of cuts has not reached K, decide if the current partition should be subdivided by checking the stability of the cut.
5. Recursively repartition the segmented parts if necessary.

Normalized cut (Ncut)

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$



Region segmentation using graph cuts

a b c

FIGURE 10.56 (a) Image of size 600 × 600 pixels. (b) Image smoothed with a 25 × 25 box kernel. (c) Graph cut segmentation obtained by specifying two regions.

Interactive segmentation with graph cut

https://sandipanweb.wordpress.com/2018/02/11/interactive-image-segmentation-with-graph-cut/

In summary

- Edge linking: fixing broken edges
- Hough transform: line representation & algorithms
- Image threshold: basic global, optimal global & Otsu's method
- Image segmentation: k-means & graph cuts

Reading materials

- Chapter 10
- Page 737-753, 763-764, 770-772, 777-785

Questions

1. What is the motivation for using "normal representation of points/lines"?
2. What are the drawbacks of k-means algorithms for image segmentation? and what are the ways to mitigate the potential drawbacks?
3. What are the scenarios for using Otsu's method and moving average? How we can relate this to removing periodic noise (e.g., notch filter)?
4. What are the differences between the simple local edge linking in Lecture 9 and the edge linking algorithm in Canny edge detection?
5. Why do we use normalized cut to decide minimum cuts in graph-cut-based segmentation? (please refer to the textbook regarding meaningful cuts)
6. What is the meaning of an edge in a graph in the set up of a graph-cut segmentation?
7. What is the key metric we are optimizing for Otsu's method?