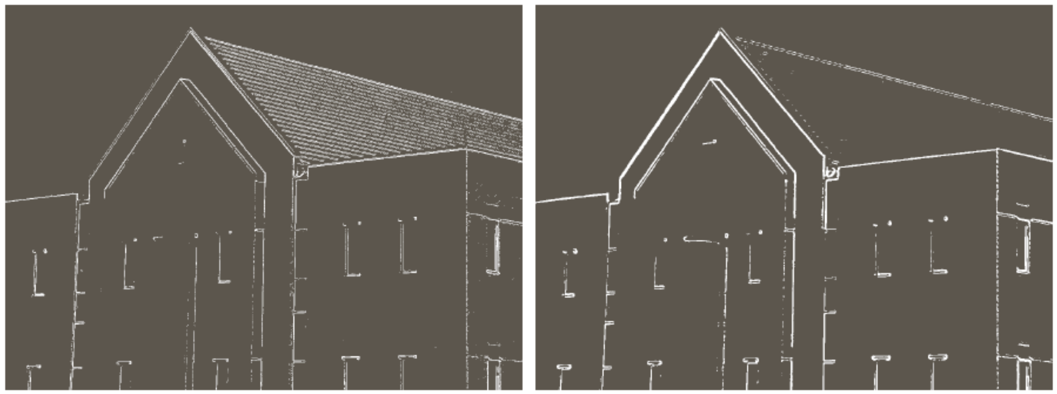
Lecture #20

- Edge operators: LoG, DoG, Canny
- Edge linking
- Polygonal line fitting, polygon boundaries
- Edge relaxation
- Hough transform

Image Segmentation



Thresholded gradient image w/o smoothing

Thresholded gradient image w/smoothing

Threshold = 1/3 max_value

a b

FIGURE 10.20 (a) Thresholded version of the image in Fig. 10.16(d), with the threshold selected as 33% of the highest value in the image; this threshold was just high enough to eliminate most of the brick edges in the gradient image. (b) Thresholded version of the image in Fig. 10.18(d), obtained using a threshold equal to 33% of the highest value in that image.

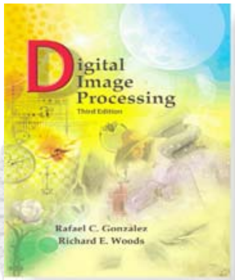
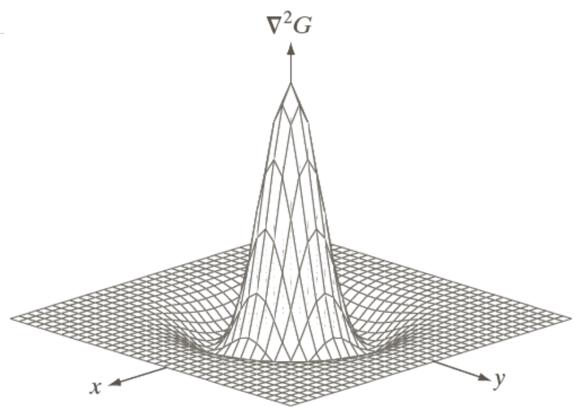


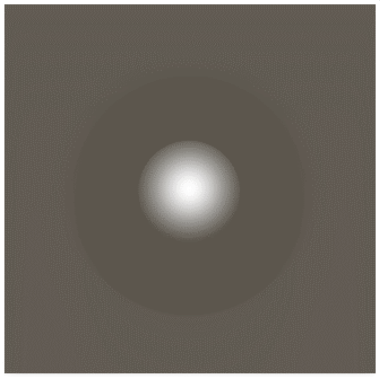
Image Segmentation

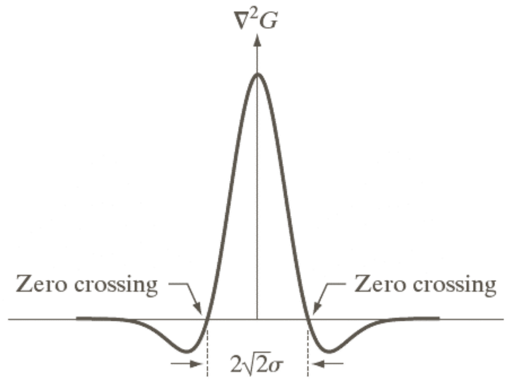
The LoG is sometimes called the Mexican hat operator

The Laplacian is NEVER used directly because of its strong noise sensitivity

Combining the Laplacian with a Gaussian gives the LoG







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

a b
c d

FIGURE 10.21 (a) 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.

$$\nabla^2 h(r) = -\left[\frac{r^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}}$$

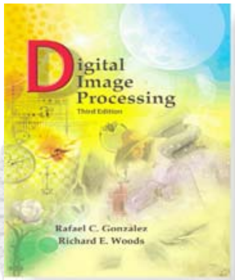


Image Segmentation

Marr-Hildereth algorithm:

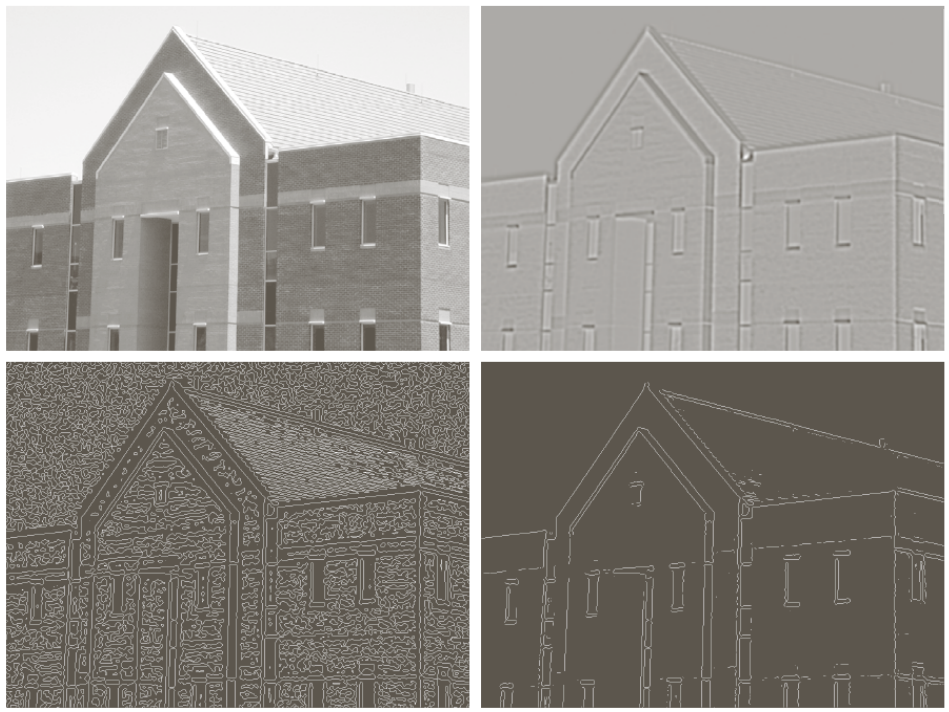
- Filter image with a nxn Gaussian low-pass filter $G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$
- Compute the Laplacian of the filtered image using an appropriate mask
- Find the zero crossings of this image

This operator is based upon a 2nd derivative operator and can be scaled using the parameter σ to fit a particular image or application, i.e., small operators for sharp detail and large operators for blurry edges

Image Segmentation

834x1134 pixel original image

After LoG operator with n=5, σ=4



a b
c d

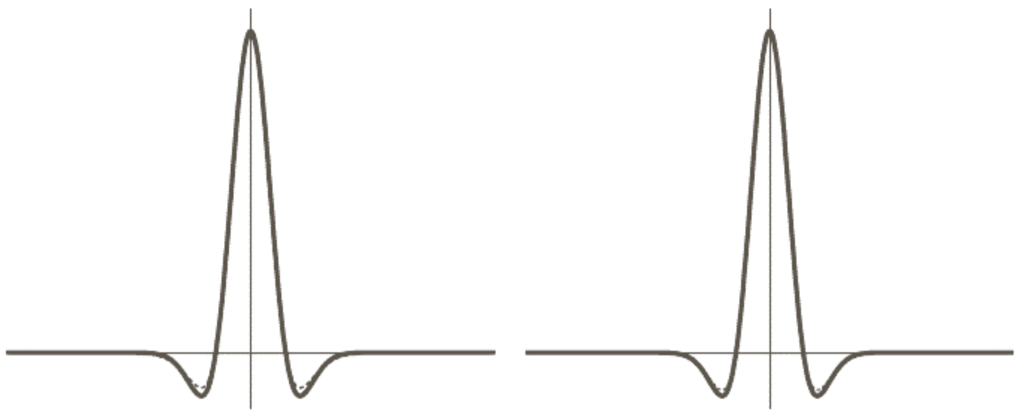
FIGURE 10.22 (a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) Results of Steps 1 and 2 of the Marr-Hildreth algorithm using $\sigma = 4$ and $n = 25$. (c) Zero crossings of (b) using a threshold of 0 (note the closed-loop edges). (d) Zero crossings found using a threshold equal to 4% of the maximum value of the image in (b). Note the thin edges.

Rules of thumb:
1. pick σ equal to the smallest dimension of interest in the image
2. Choose n such that n (smallest odd integer)≥6σ

Zero crossings with T=0 (spaghetti effect)

Zero crossings with T=0.4*Max_value

Image Segmentation



a b

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

The Laplacian of a Gaussian (LoG) can be approximated by a Difference of Gaussians (DoG) provided the ratio σ_1/σ_2 is picked appropriately. The ratio of 1.6:1 seems to work best in practice.

$$LoG(x,y) = -\left[\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right]e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad DoG(x,y) = \frac{1}{2\pi\sigma_1^2}e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2}e^{-\frac{x^2+y^2}{2\sigma_2^2}}$$

Canny Edge operator

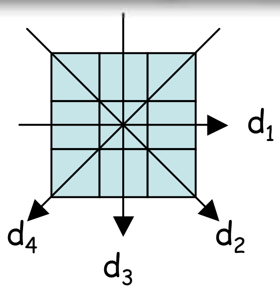
1. Smooth image with a Gaussian filter
2. Compute gradient magnitude M[x,y] and direction α[x,y]
3. Apply non-maximal suppression to the gradient magnitude
4. Use double thresholding (and subsequent connectivity analysis) to detect and link edges

Non-Maximal Suppression

- M[i,j] will have large values where the gradient is large. We need to find the local maxima in this array to locate the edges.
- Must <u>thin</u> so only points of greatest local change remain.

Non-Maximal Suppression

For a 3x3 region quantize α to four directions $\zeta[i,j]= \{d_1, d_2, d_3, d_4\}$



1. Pick the d_i which is closest to $\alpha[x,y]$
2. If $M[x,y]$ is less than one of its two neighbors along $\alpha[x,y]$ then $g_N[x,y]=0$ [suppress a non-maximum] else $g_N[x,y]=M[x,y]$

NOTE: Resulting contours may still be be multiple-pixels thick requiring use of a thinning algorithm

Denote the entire process $N[i,j]=Non\_maximal\_suppression\{M[I,j], \zeta[i,j]\}$

Non-Maximal Suppression

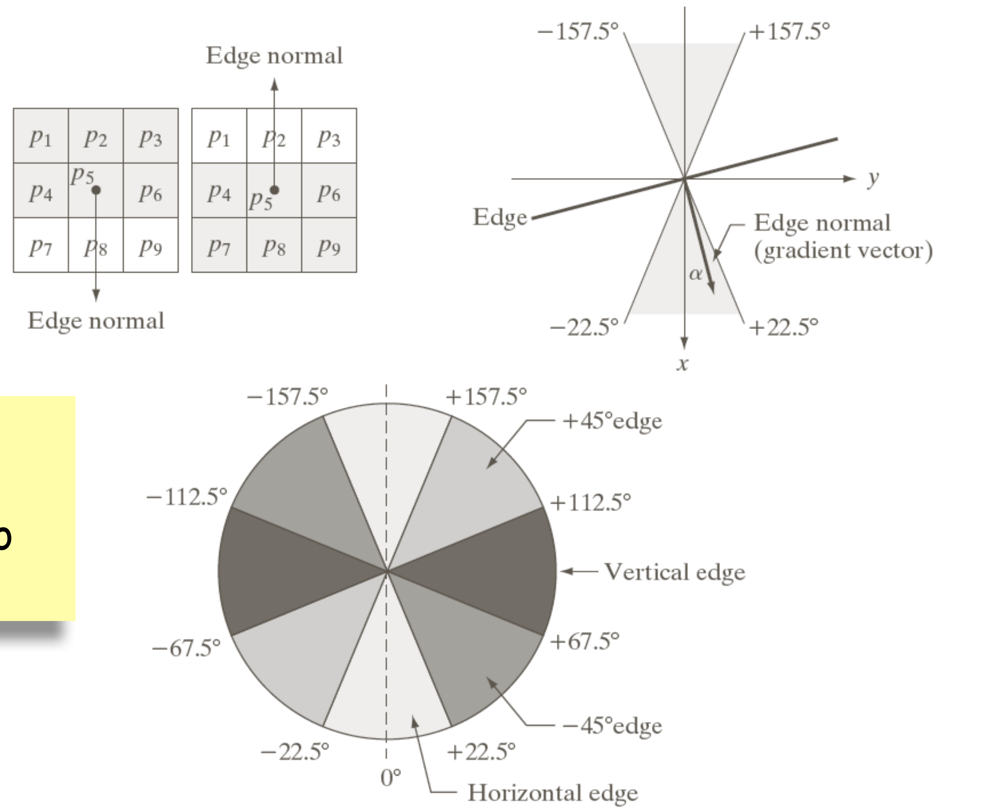
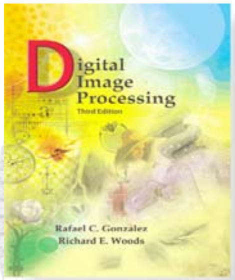


FIGURE 10.24 (a) Two possible orientations of a horizontal edge (in gray) in a 3 × 3 neighborhood. (b) Range of values (in gray) of α, the direction angle of the *edge normal*, for a horizontal edge. (c) The angle ranges of the edge normals for the four types of edge directions in a 3 × 3 neighborhood. Each edge direction has two ranges, shown in corresponding shades of gray.

Each direction $\{d_1, d_2, d_3, d_4\}$ actually corresponds to two edge directions



Double Thresholding

- After non-maximal suppression image contains many false edge fragments caused by noise and fine texture
- You can threshold N[i,j], but good results are difficult to achieve with a single threshold T.
- Use two thresholds T_1 and T_2. Initially link contours using threshold T_1. If a gap is encountered <u>drop</u> to threshold T_2 until you rejoin a T_1 contour.

Canny operator example

(http://homepages.inf.ed.ac.uk/rbf/HIPR2/canny.htm)

(a) Original image

(b) Canny, σ=1.0, T_1=255, T_2=1

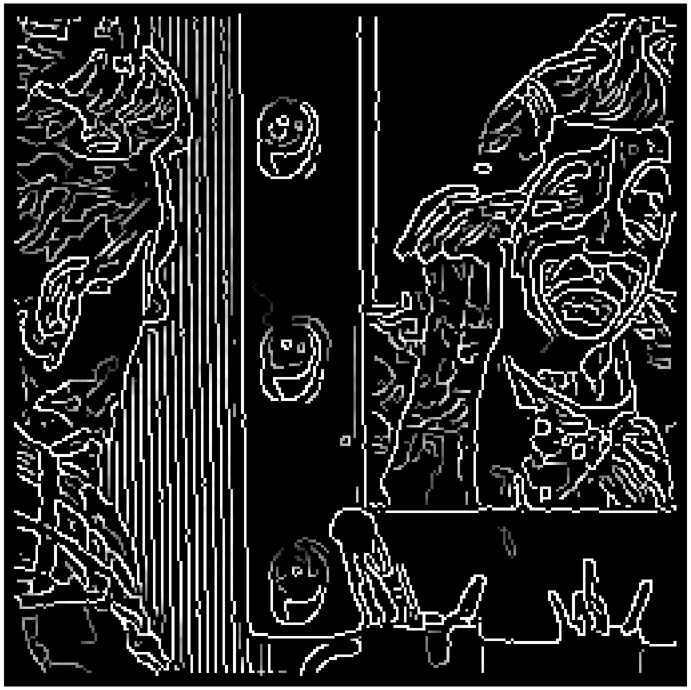
Canny Operator Example

(a) Original

(b) Canny, σ=1.0, T_1=255, T_2=220

Canny Operator Example

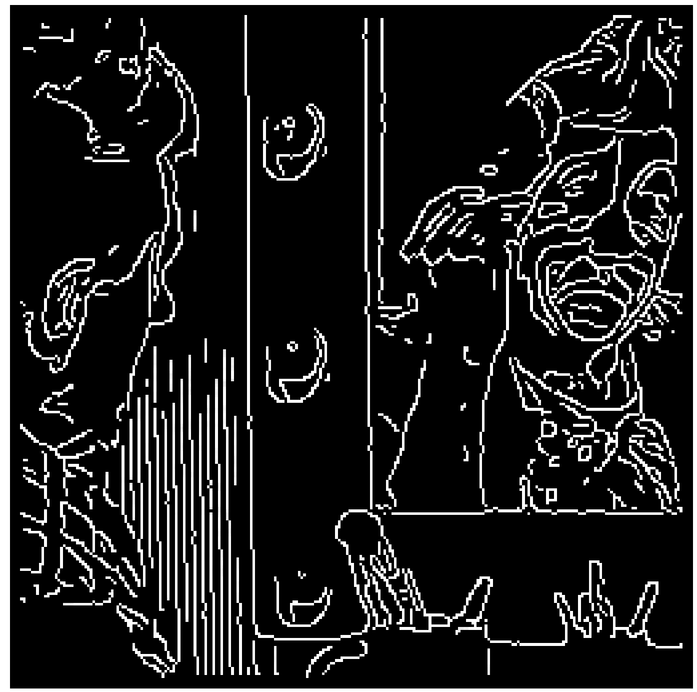
(a) Original



(b) Canny, σ=1.0, T_1=128, T_2=1

Canny Operator Example

(a) Original



(b) Canny, σ=2.0, T_1=128, T_2=1

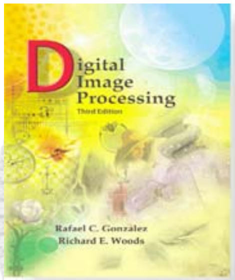


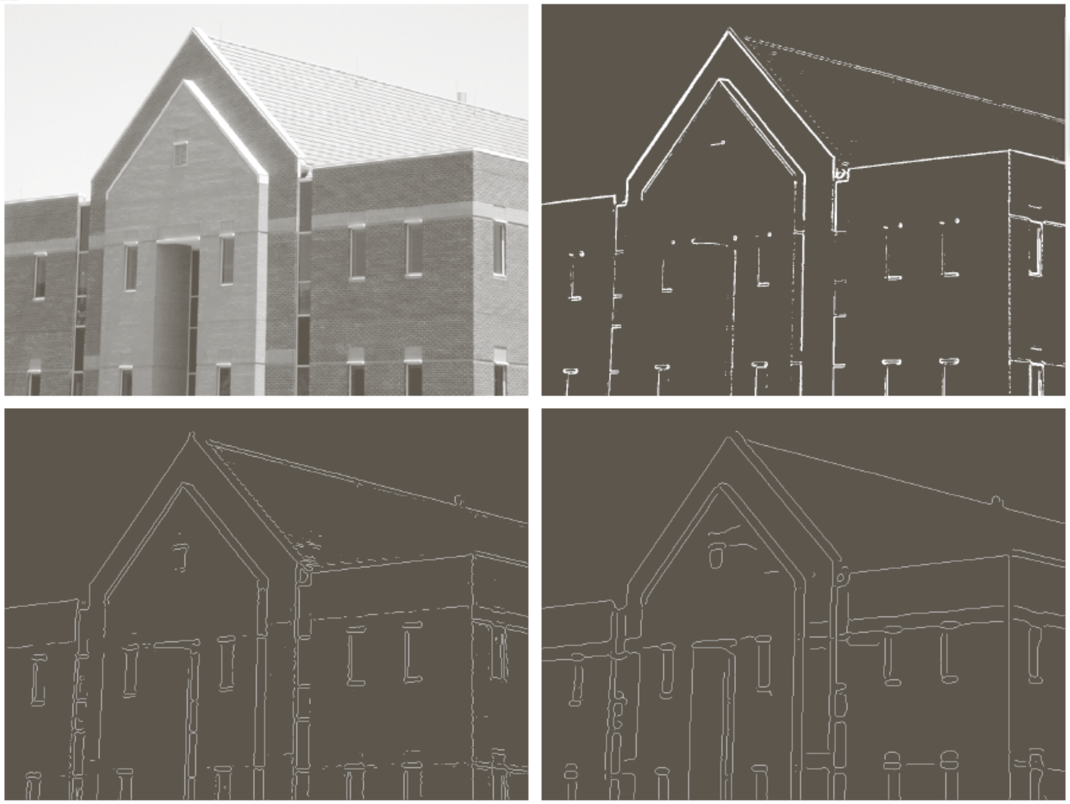
Image Segmentation

Original 834x1114 image

Thresholded gradient

Marr-Hildereth (LoG)

Canny T_L=0.04, T_H=0.1, σ =4, 25x25 mask



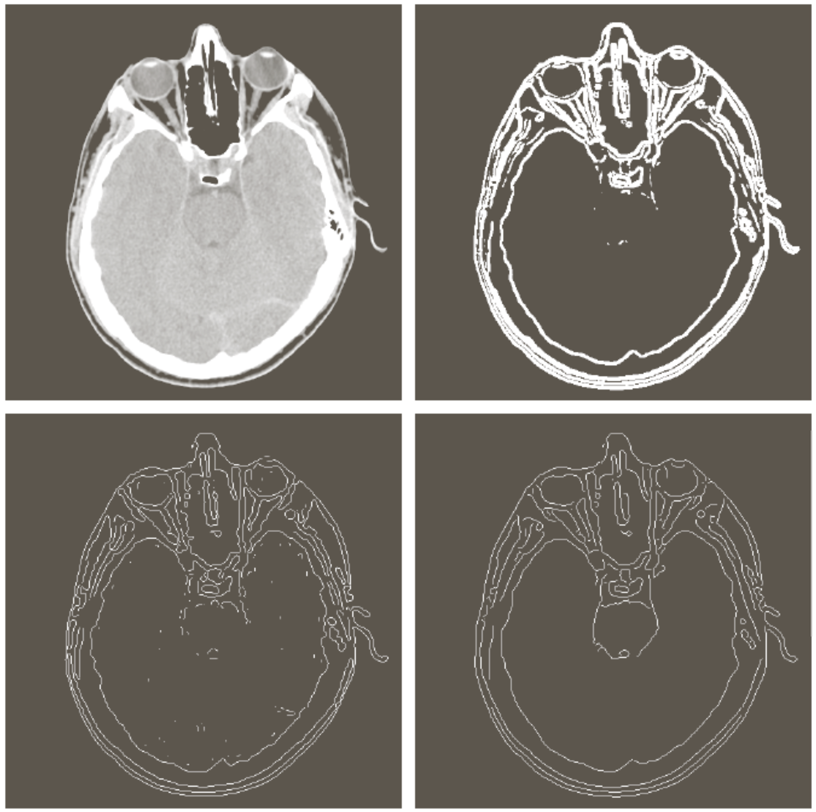
a b
c d

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.

Image Segmentation

Original
834x1114 image

Smoothed by
5x5 averaging
filter,
thresholded
gradient
T=0.15*max

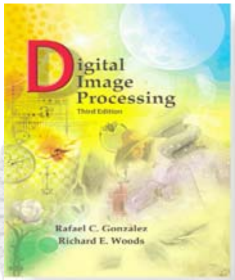


Marr-Hildereth
(LoG) T=0.002,
σ=3, 19x19 mask

Canny T_L=0.05,
T_H=0.15, σ=2,
13x13 mask

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.)



Edge Linking

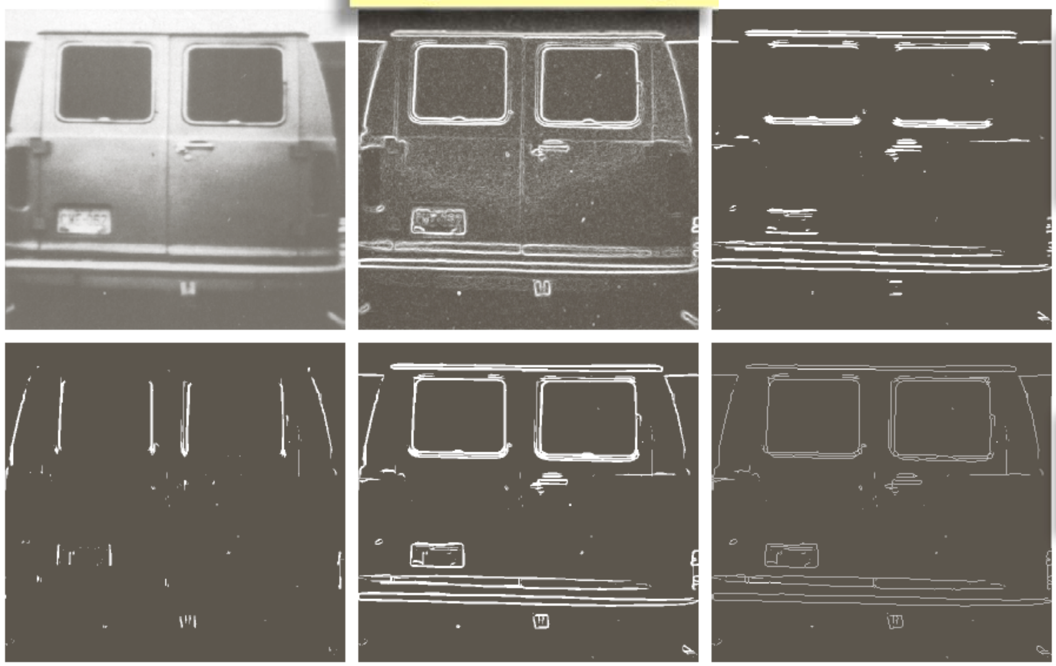
Gradient magnitude image

Original 534x566 image

Link edge pixels in horizontal direction

Link edge pixels in vertical direction

Thinned logical OR



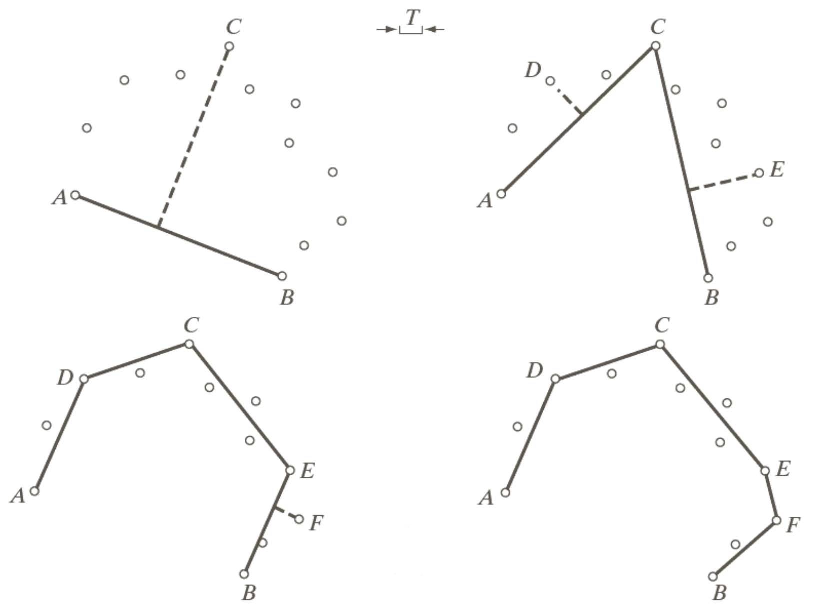
Logical OR of horizontally and vertically connected images

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.)

Line Fitting

This algorithm works for ordered sets of points on an open curve with known end points.



a b
c d

FIGURE 10.28 Illustration of the iterative polygonal fit algorithm.

1. Start with known end points A and B in a binary image.
2. Determine maximum perpendicular distant pixel C from AB.
3. If the distance from AB to C is greater than threshold T pick C as a new endpoint for new segments AC and CB.
4. Repeat until all perpendicular distances less than T.

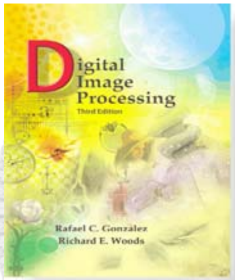
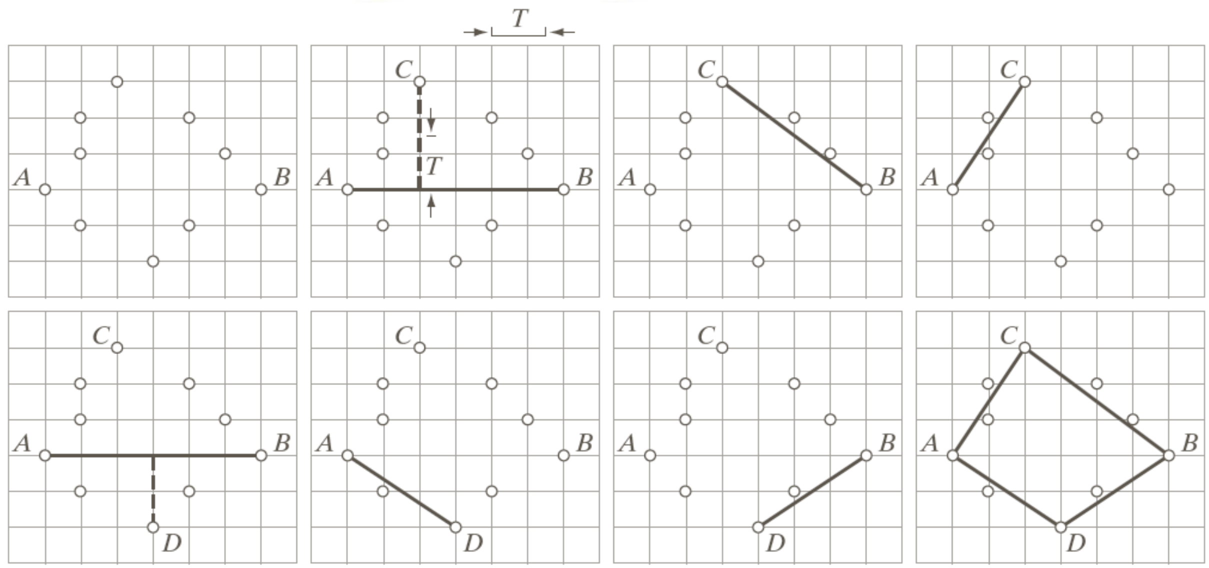


Image Segmentation



What if you don't know the end points?

1. Pick two points A,B on the curve (the most horizontally distant in this case)
2. Specify a threshold T and two stacks PENDING and SETTLED
3. If the curve is open put A into PENDING and B into both PENDING and SETTLED. If the curve is closed put A into PENDING and B into SETTLED
4. Compute the parameters of the line passing from the last vertex in SETTLED to the last vertex in PENDING
5. Compute the perpendicular distances for all edge points between these vertices and select the pixel V_{max} with the maximum distance D_{max}.
6. IF D_{max}>T place Vmax into PENDING as a new vertex and goto 4 ELSE remove the last vertex from PENDING and place it into SETTLED.
7. IF PENDING is not empty goto 4 ELSE exit.
8. The vertices in SETTLED are the verticles of the polygonal fit.

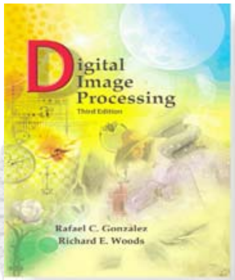
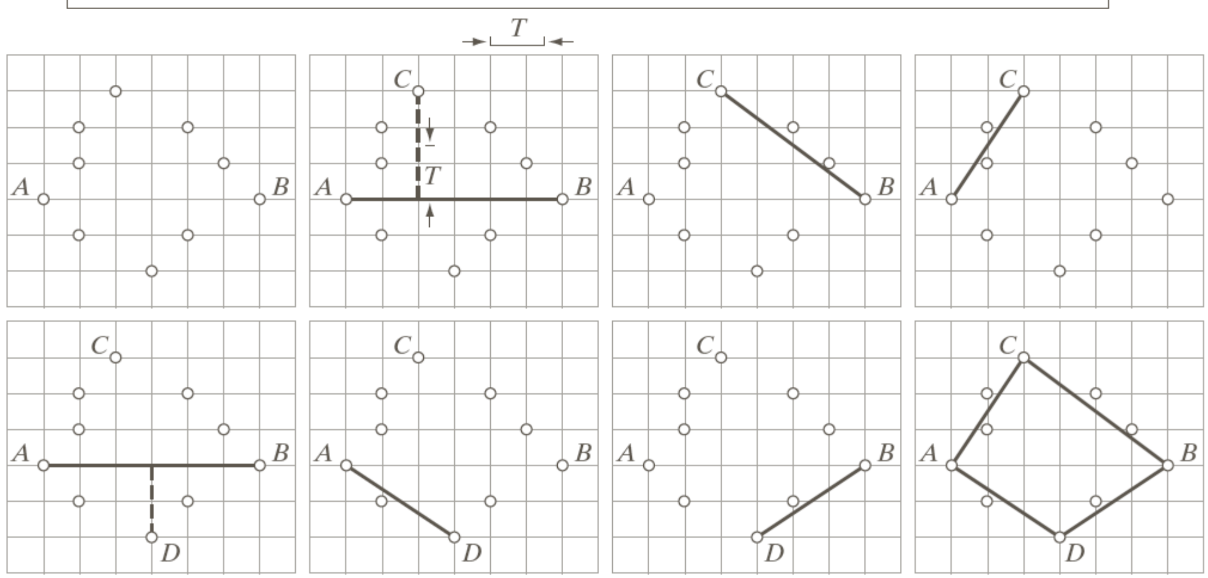


Image Segmentation

SETTLED	PENDING	Curve segment processed	Vertex generated
B	*B*, *A*	—	*A*, *B*
B	*B*, *A*	(*BA*)	*C*
B	*B*, *A*, *C*	(*BC*)	—
B, *C*	*B*, *A*	(*CA*)	—
B, *C*, *A*	*B*	(*AB*)	*D*
B, *C*, *A*	*B*, *D*	(*AD*)	—
B, *C*, *A*, *D*	*B*	(*DB*)	—
B, *C*, *A*, *D*, *B*	Empty	—	—

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



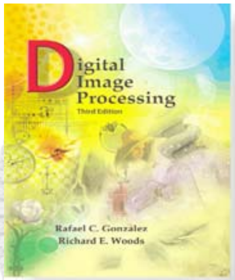


Image Segmentation

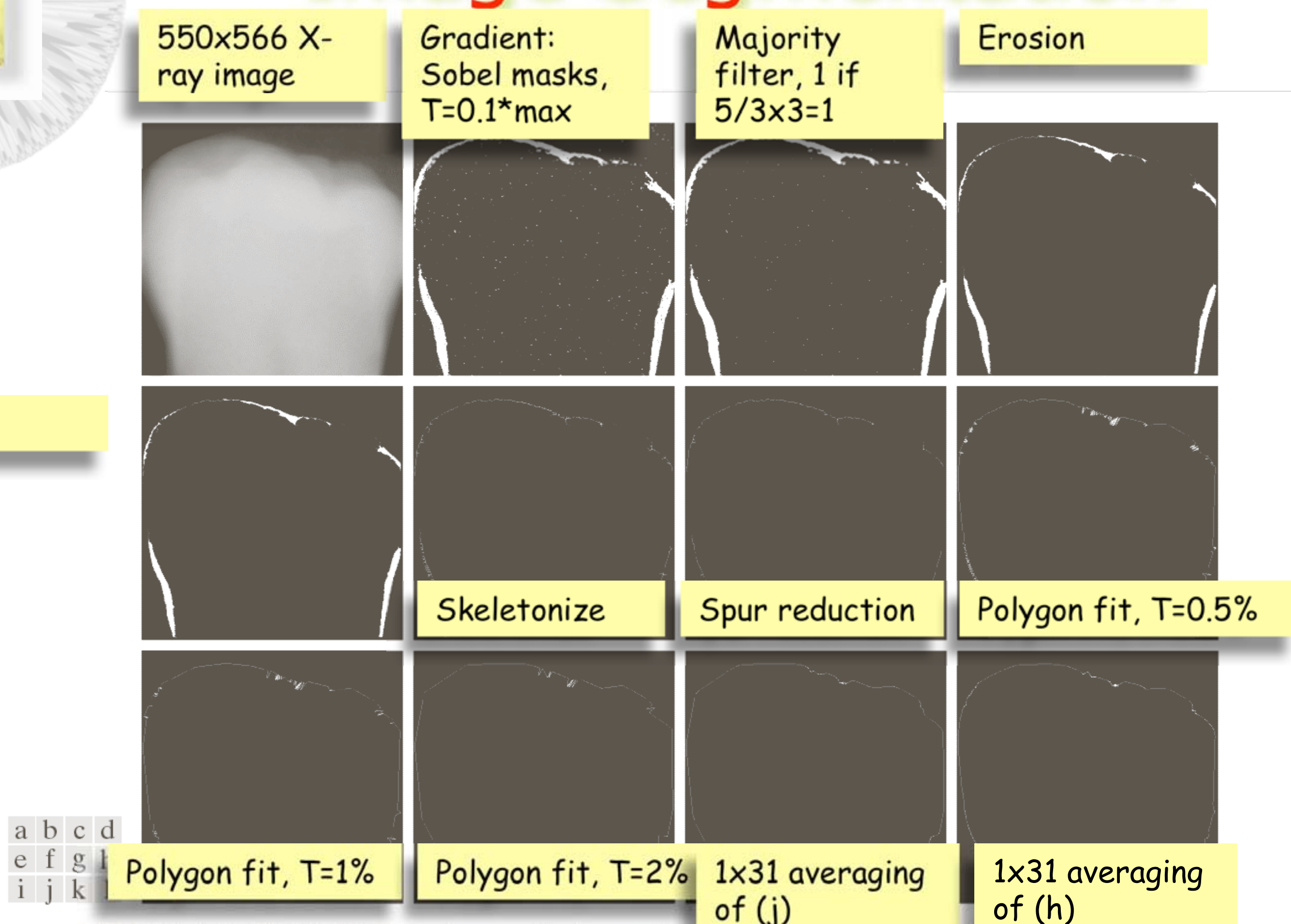
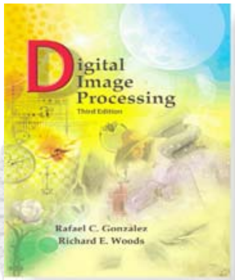


FIGURE 10.30 (a) A 550 × 566 X-ray image of a human tooth. (b) Gradient image. (c) Result of majority filtering. (d) Result of morphological shrinking. (e) Result of morphological cleaning. (f) Skeleton. (g) Spur reduction. (h)–(j) Polygonal fit using thresholds of approximately 0.5%, 1%, and 2% of image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



Edge Relaxation

Improve edge operator estimate by re-adjusting edge estimate based upon local information. Let

$$C^0(e) = \frac{gradient\ \ magnitude\ \ at\ \ e}{\max\ gradient\ \ magnitude\ \ in\ \ image}$$

Initial confidence of edge is simply normalized gradient

k=1

While any $C^k(e) \neq (0 \text{ or } 1)$ do

Begin

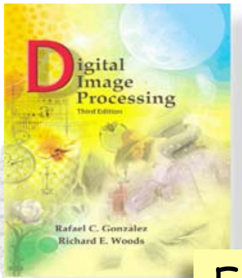
Edge_type=f(edge_neighbors) %classify edge type

$C^k(e)$=f(edge_type, $C^{k-1}(e)$) %adjust confidence

k=k+1

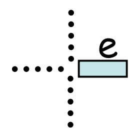
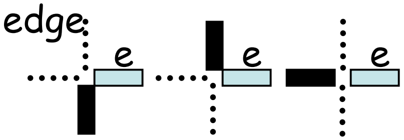
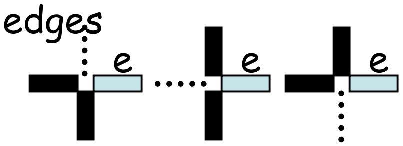
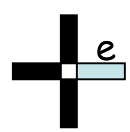
End

Algorithm forces all points in edge image to converge to 0 or 1



Edge Relaxation

Edge type is concatenation of left and right vertex types, i.e., edge(e)=(i,j) with (3,3) being the strongest edge

Left vertex type	Description (with figure, strong edges in black, weak edges dashed)	Confidence
0	All weak edges 	(m-a)(m-b)(m-c) All weak so decrease confidence
1	One strong edge 	a(m-b)(m-c) One strong edge so increase confidence
2	Two strong edges 	ab(m-c) Ambiguous case so keep confidence about the same
3	Three strong edges 	abc Ambiguous case so keep confidence about the same

Edge Relaxation Example

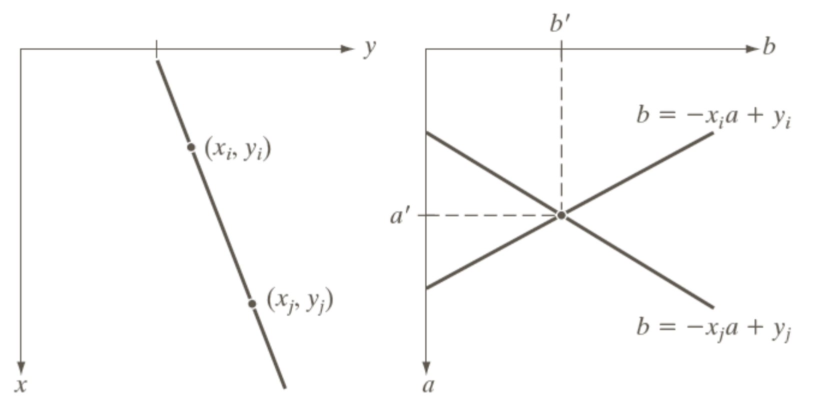
(Source: Ballard & Brown)

Gradient image (thresholded for display)

Fig. 3.22 Edge relaxation results. (a) Raw edge data. Edge strengths have been thresholded at 0.25 for display purposes only. (b) Results after five iterations of relaxation applied to (a). (c) Different version of (a). Edge strengths have been thresholded at 0.25 for display purposes only. (d) Results after five iterations of relaxation applied to (c).

Five iterations of relaxation

Hough Transform



a b

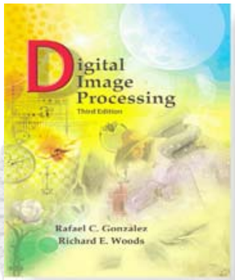
FIGURE 10.31
(a) *xy*-plane.
(b) Parameter space.

Points (x_i,y_i) and (x_j,y_j)

Define a line $y=ax+b$ in x-y space

Rearrange equation as $b=-ax+y$

For every point (x_i,y_i) there is a corresponding line describing all the possible slopes and y-intercepts. Their intersection is the original line in x-y space.



Hough Transform

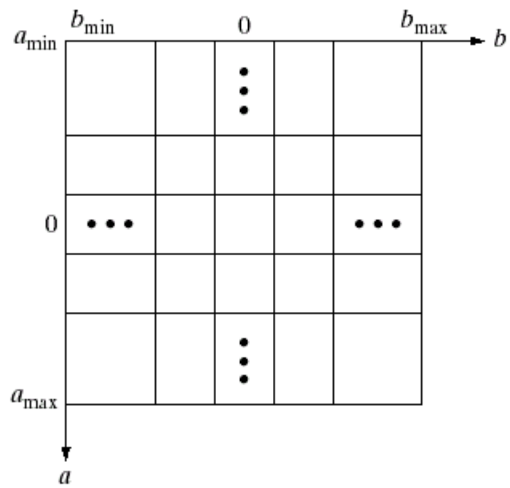
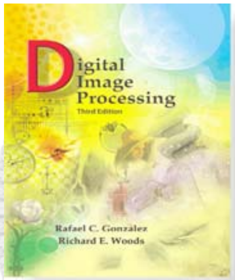


FIGURE 10.18 Subdivision of the parameter plane for use in the Hough transform.

Hough Transform

1. Quantize parameter space between appropriate maxima and minima for y-intercept b and slope a
2. Form an accumulator array $A[b,a]:=0$
3. For each **point** (x,y) in an edge-enhanced image such that $E(x,y)>T$, increment all points in $A[b,a]$ along the appropriate **line** in a-b space, i.e., $A[b,a]:=A[b,a]+1$ for $b=-ax+y$
4. **Local maxima** in $A[b,a]$ space correspond to collinear points (i.e., lines) in the image array. Values in $A[b,a]$ correspond to how many points exist on that line.



Hough Transform

- Problem: a-b space is unbounded as for near-vertical lines a->$\pm\infty$
- Solution: convert to polar coordinates before transforming

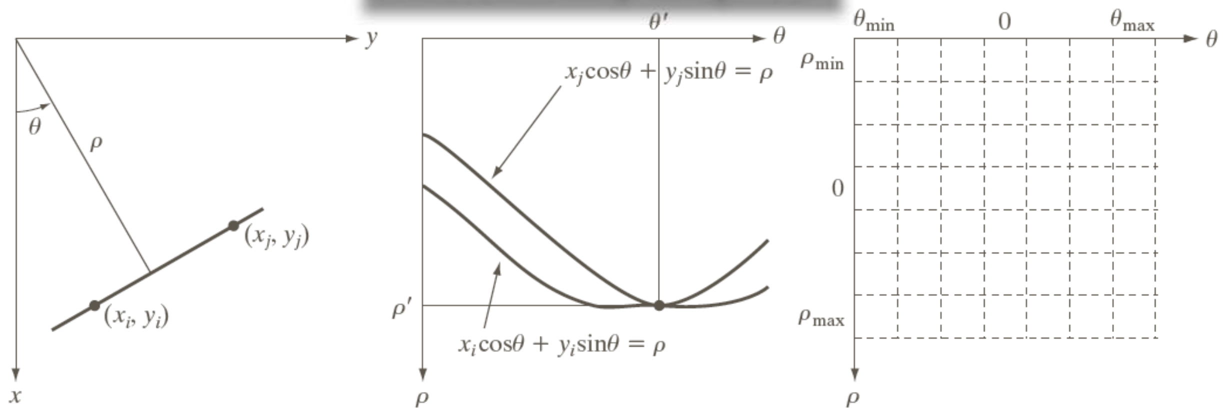
$$r = x\cos\theta + y\sin\theta$$

This will give sinusoids instead of straight lines

Hough Transform

The possible lines are sinusoids in ρ-θ space

Points (x_i,y_i) and (x_j,y_j) define a line in polar coordinates

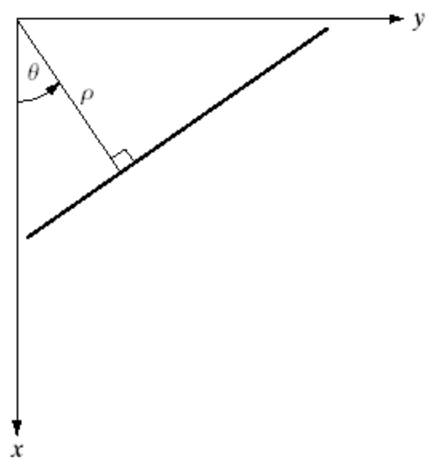


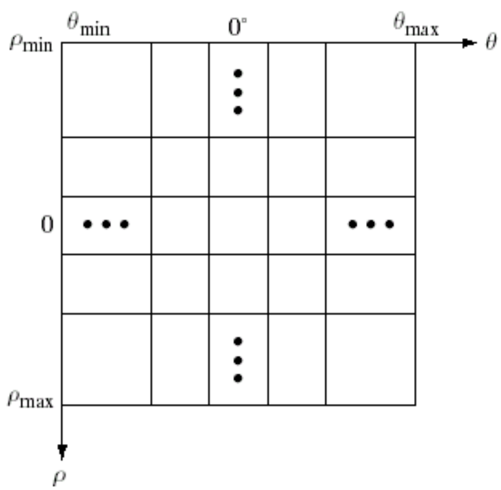
Map possible sinusoids into accumulator cells in discrete ρ-θ space

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.

Chapter 10
Image Segmentation

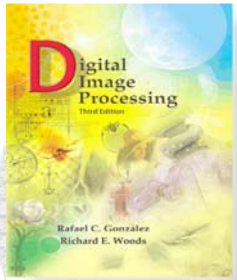




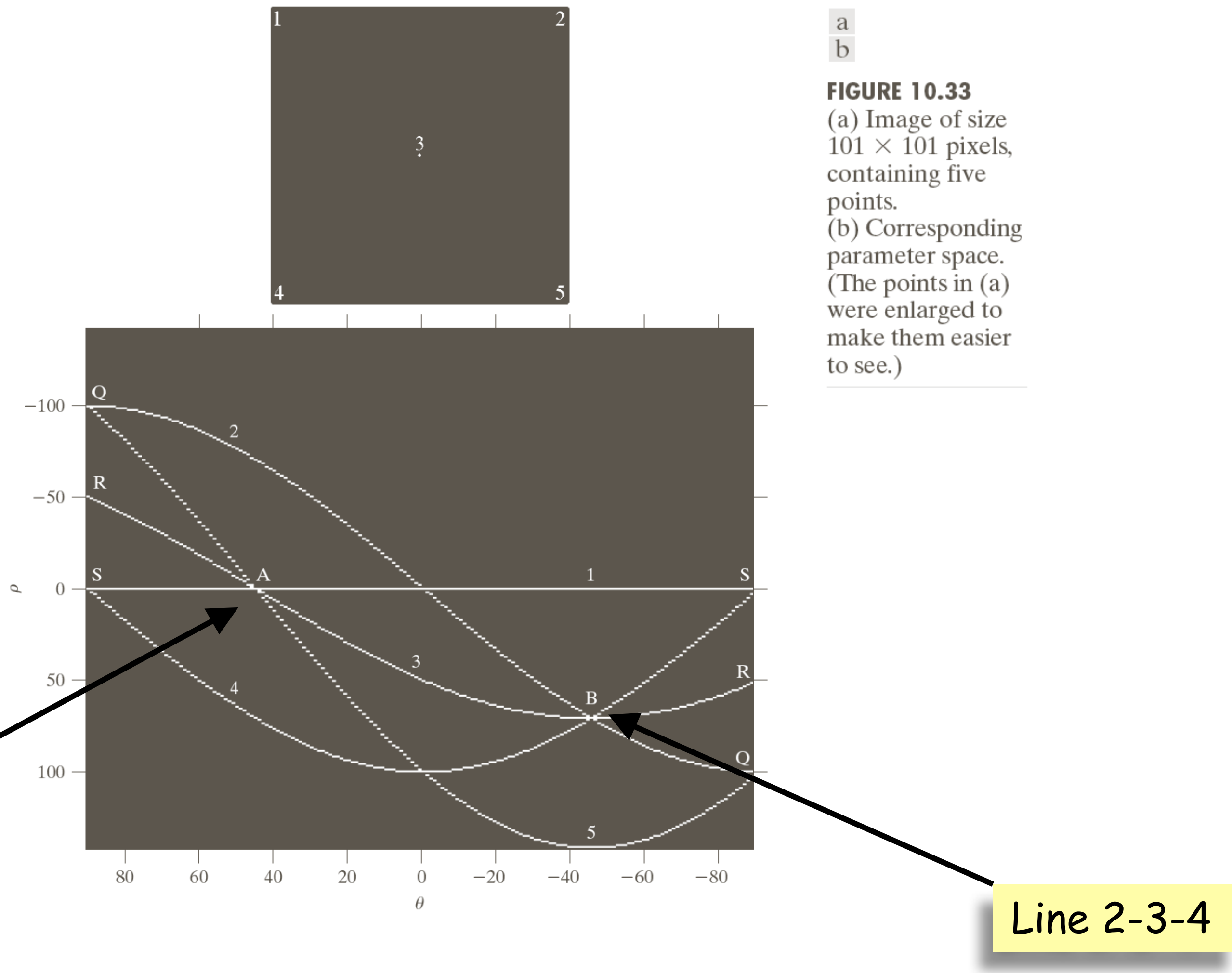
a b

FIGURE 10.19 (a) Normal representation of a line. (b) Subdivision of the $\rho\theta$-plane into cells.

Map possible sinusoids into accumulator cells in discrete ρ-θ space

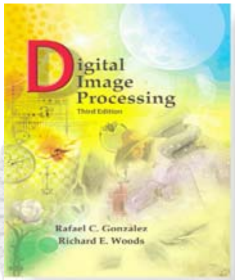


Hough Transform



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.)



Hough Transform

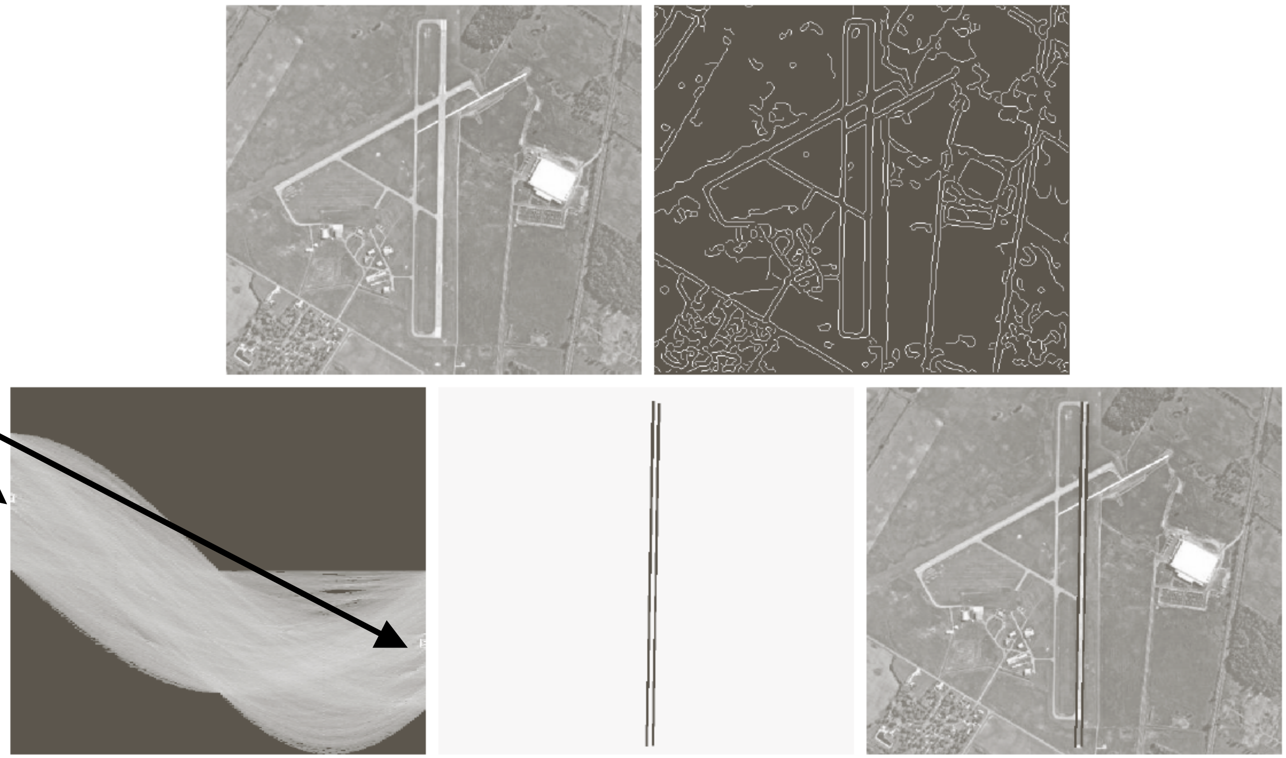
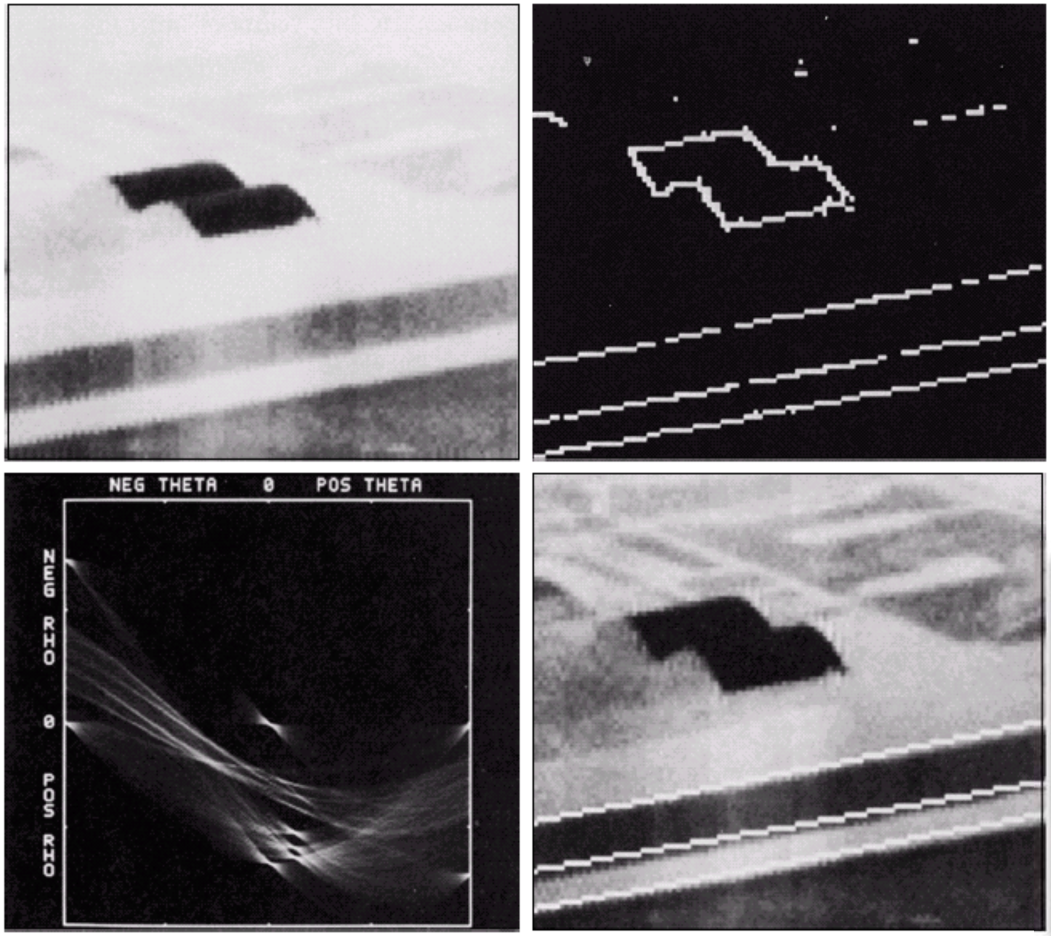


FIGURE 10.34 (a) A 502 × 564 aerial image of an airport. (b) Edge image obtained using Canny's algorithm. (c) Hough parameter space (the boxes highlight the points associated with long vertical lines). (d) Lines in the image plane corresponding to the points highlighted by the boxes). (e) Lines superimposed on the original image.

Hough Transform



a b
c d

FIGURE 10.21 (a) Infrared image. (b) Thresholded gradient image. (c) Hough transform. (d) Linked pixels. (Courtesy of Mr. D. R. Cate, Texas Instruments, Inc.)

Linked pixels from strongest points in accumulator. No gaps in linked image.