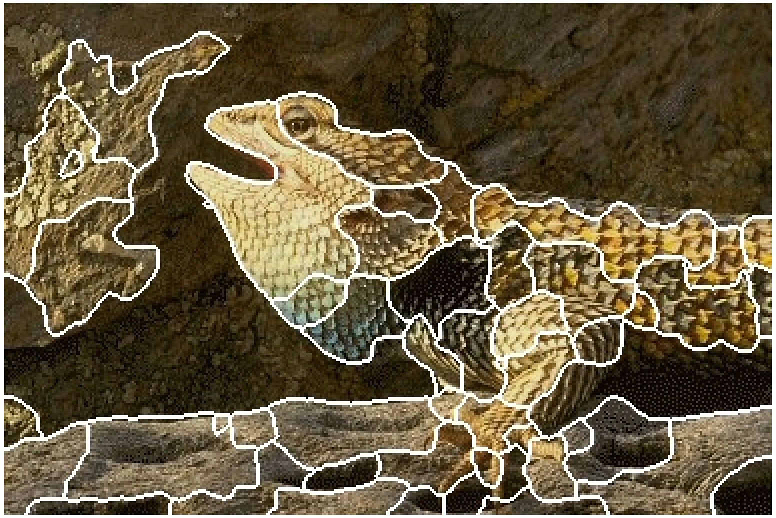
Segmentation, Edge Detection

Dr. Terence Sim

12 – July 2006

Segmentation



http://vision.ece.ucsb.edu/segmentation

Segmentation

– Let R be the entire image.

– Segmentation aims to partition *R* into *n* sub-regions, such that

(a) *R* consists of all *n* subregions, $\bigcup_{i=1}^{n} R_i = R$, so that segmentation is complete.

(b) R_i is a connected region, i=1,2, ... ,n. Pixels in a region are connected (no holes exist).

(c) Regions do not overlap $R_i \cap R_j = \phi$

(d) Pixels in the same subregion are similar.

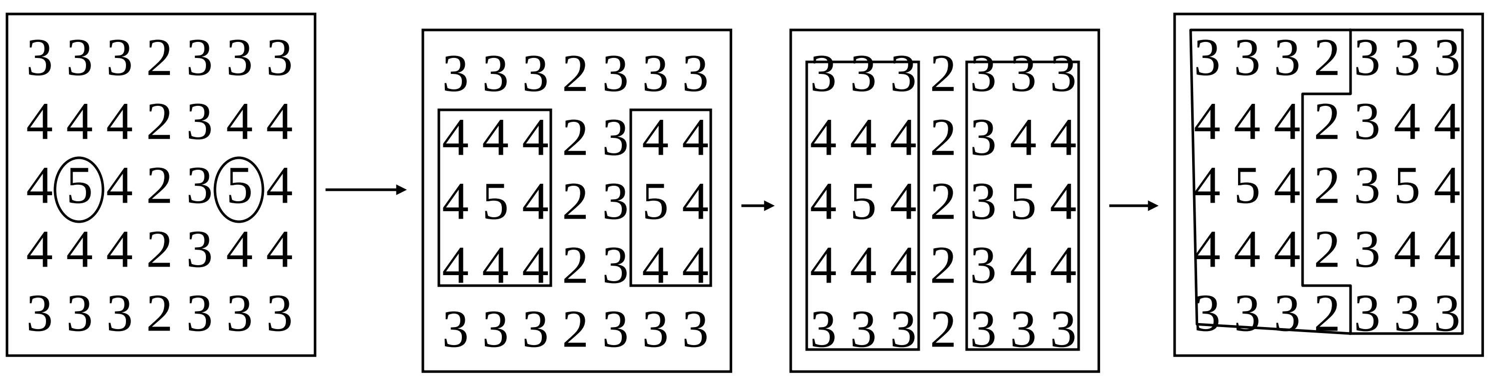
$$P(R_i) = \text{TRUE}$$

(e) Pixels in different subregions are dissimilar.

$$P(R_i \cup R_j) = \text{FALSE}$$

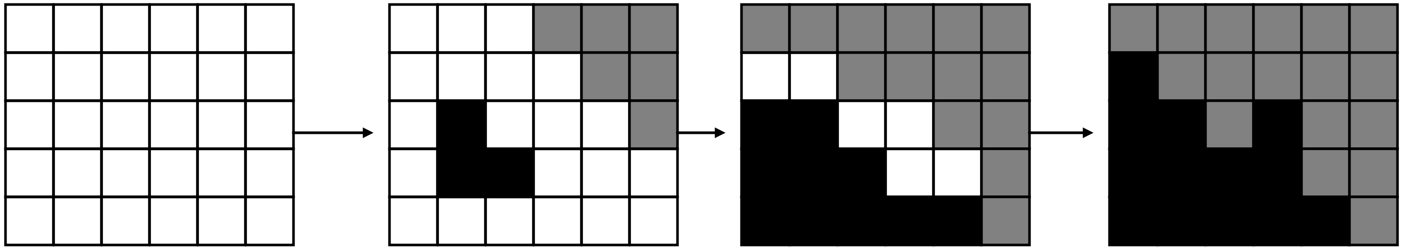
Region Growing

– Regions are grown from “seed” points.
– A “seed” point is selected for each potential region. Append to each point those neighbors that have similar properties (e.g. Gray level, color, texture, etc.)
– problems of this method
  - selection of seed
  - selection of similarity measure
  - termination condition



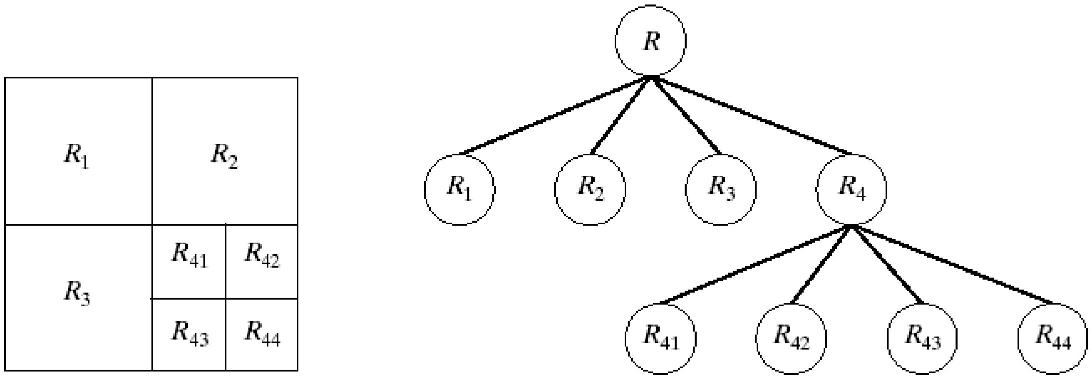
Region Splitting-Merging

- Image is systematically subdivided in to regions.
- Merge neighboring regions if they are similar.
- Split regions if pixels in them are dissimilar.
- Problem with this method is that the region boundary is jagged.



Quadtree Decomposition

- Start with entire image as single region.
- If pixels within region are similar then stop.
- Else divide region into 4 subregions, and recursively decompose each subregion.
- Matlab command: **`qtdecomp`**



Edge Detection

Robert's cross gradient operator

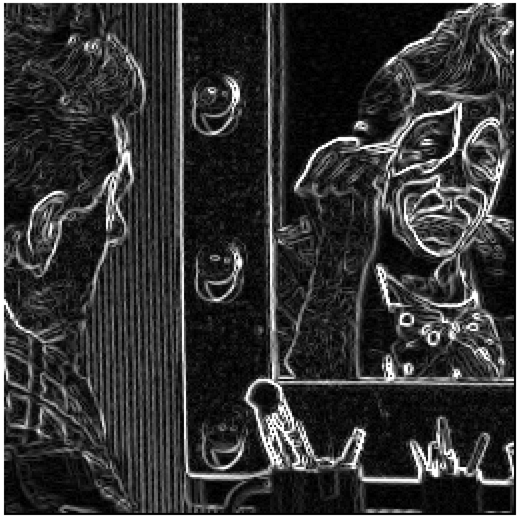
+1	0
0	-1

Gx

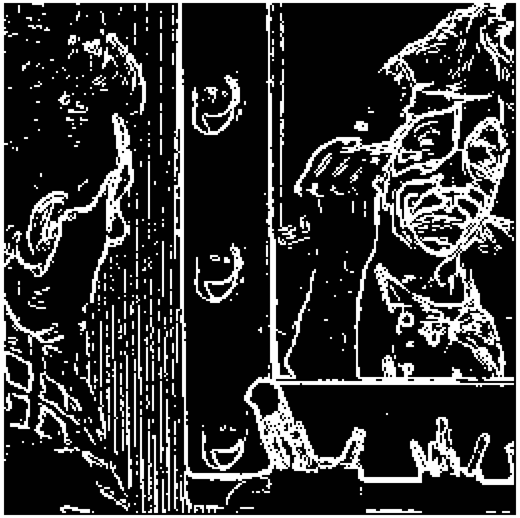
0	+1
-1	0

Gy

original



Convolved with Robert's operator

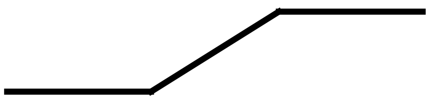


Thresholding on Robert's output

Edge Detection

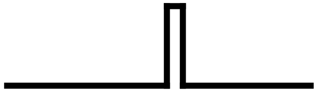
Edge Models

- Edges are places in the image with strong intensity contrast.
- Edges often occur at image locations representing object boundaries.
- Types of edges :
  - Ramp (1D profile)



  - Roof (1D profile)

  - Line (1D profile)



  - Step (1D profile)

The Step Edge Model

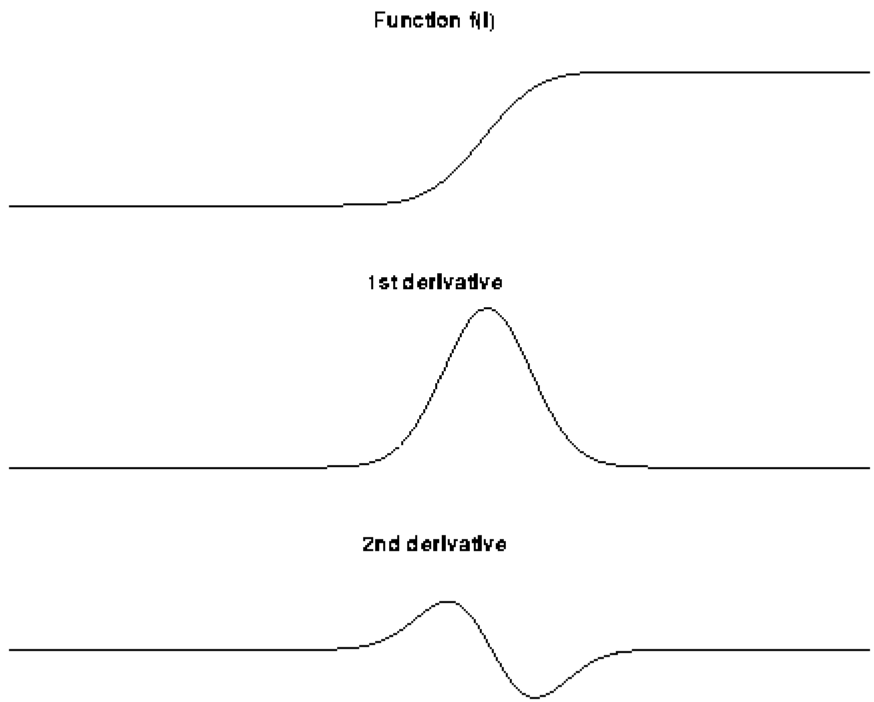
Step edge exist for artificially generated images.

Real images DO NOT have step edges because anti-aliasing filter used in the imaging system.

Real edge →

1st derivative →

2nd derivative →



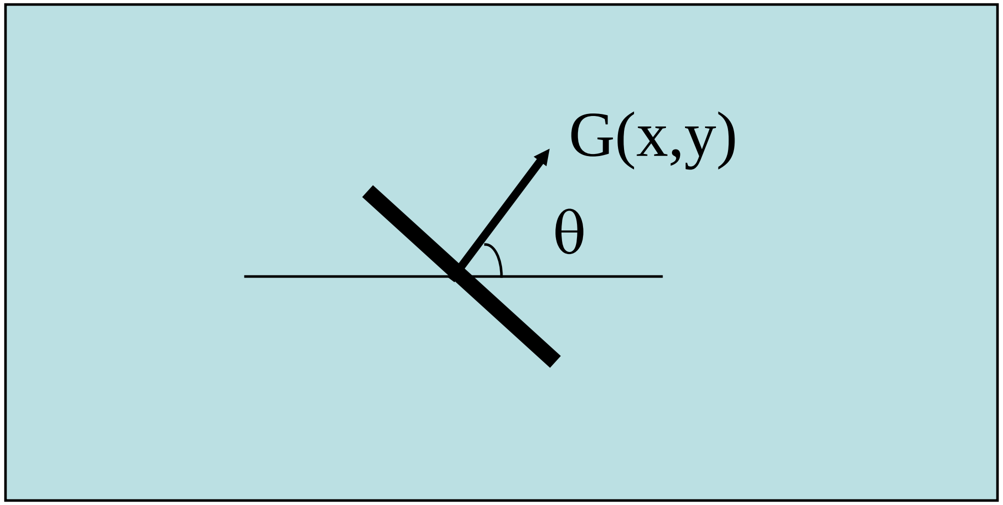
First derivative operator

Also known as Gradient operator.

For an edge in a 2D image,

$$G(x,y) = \frac{\partial F(x,y)}{\partial x}\cos(\theta) + \frac{\partial F(x,y)}{\partial y}\sin(\theta)$$

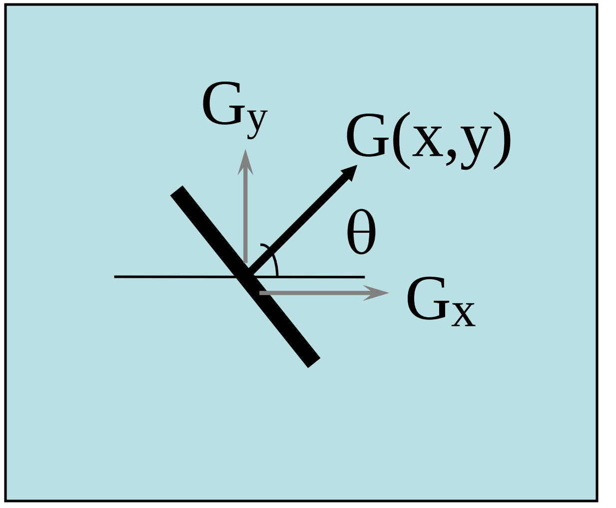
where G(x,y) is the gradient normal to the edge.



First derivative operator

Usually, we compute

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \partial f / \partial x \\ \partial f / \partial y \end{bmatrix}$$



where G_x is the horizontal gradient,

G_y is the vertical gradient.

$$|\nabla f| = \left[G_x^2 + G_y^2\right]^{1/2}$$

$$|\nabla f| \approx |G_x| + |G_y| \quad ; \theta(x, y) = \tan^{-1}\left(G_y / G_x \right)$$

Sobel's gradient operator

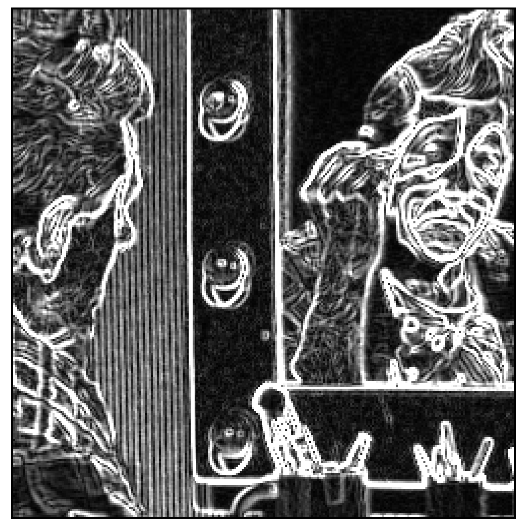
-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

original



Convolved with Sobel's operator

Second derivative operator

Laplacian operator:

The Laplacian L(x,y) of an image at I(x,y) is:

$$L(x,y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

Laplacian kernels:

0	1	0
1	−4	1
0	1	0

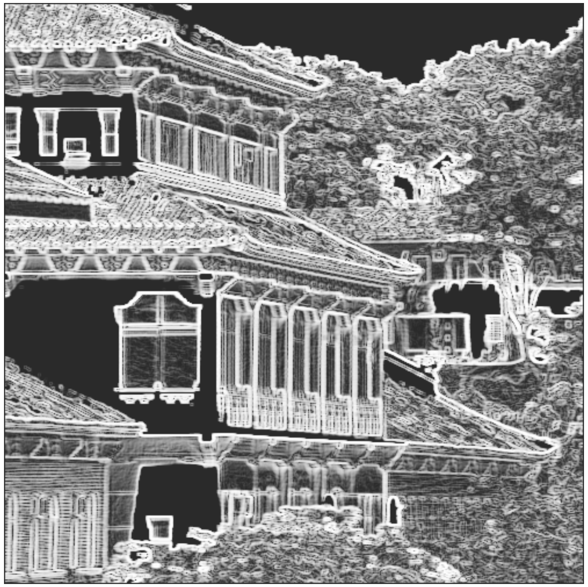
1	1	1
1	−8	1
1	1	1

−1	2	−1
2	−4	2
−1	2	−1

Laplacian operator usually produce closed contour

original

Sobel output



Laplacian output

Second derivative operator

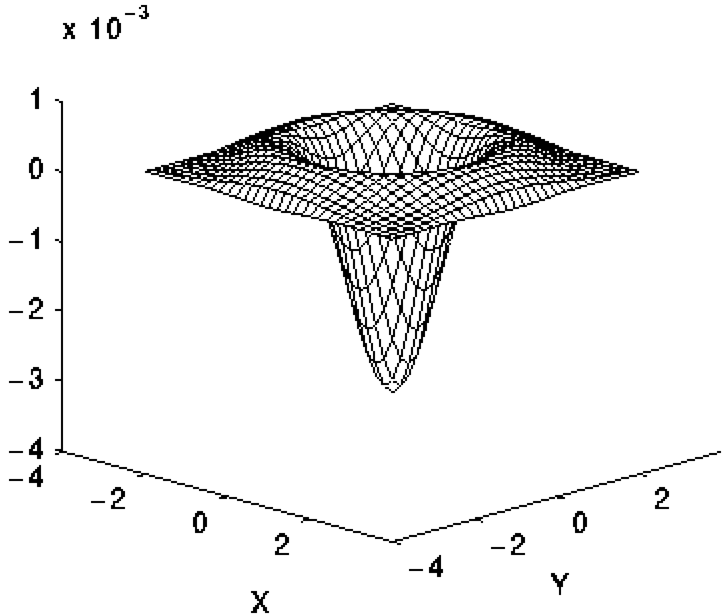
Laplacian of Gaussian (LoG) operator:

This is equivalent to Gaussian smoothing followed by Laplacian.

$$G(x,y) = \nabla^2[g(x,y) * I(x,y)]$$

$$= \nabla^2 g(x,y) * I(x,y)$$

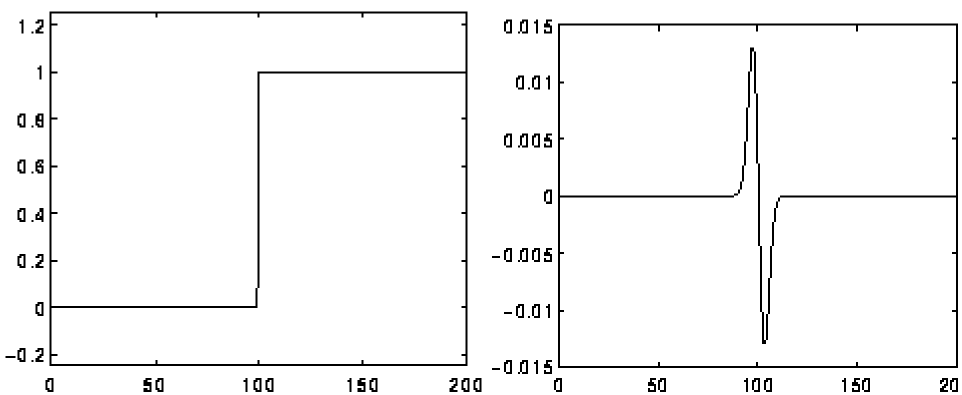
$$\nabla^2 g(x,y) = -\frac{1}{2\pi\sigma^4}\left(2 - \frac{x^2+y^2}{\sigma^2}\right)\exp\left[-\frac{x^2+y^2}{2\sigma^2}\right]$$



2D LoG function

0	0	3	2	2	2	3	0	0
0	2	3	5	5	5	3	2	0
3	3	5	3	0	3	5	3	3
2	5	3	-12	-23	-12	3	5	2
2	5	0	-23	-40	-23	0	5	2
2	5	3	-12	-23	-12	3	5	2
3	3	5	3	0	3	5	3	3
0	2	3	5	5	5	3	2	0
0	0	3	2	2	2	3	0	0

2D LoG kernel



Response of the LoG to a step edge

Edge Detection

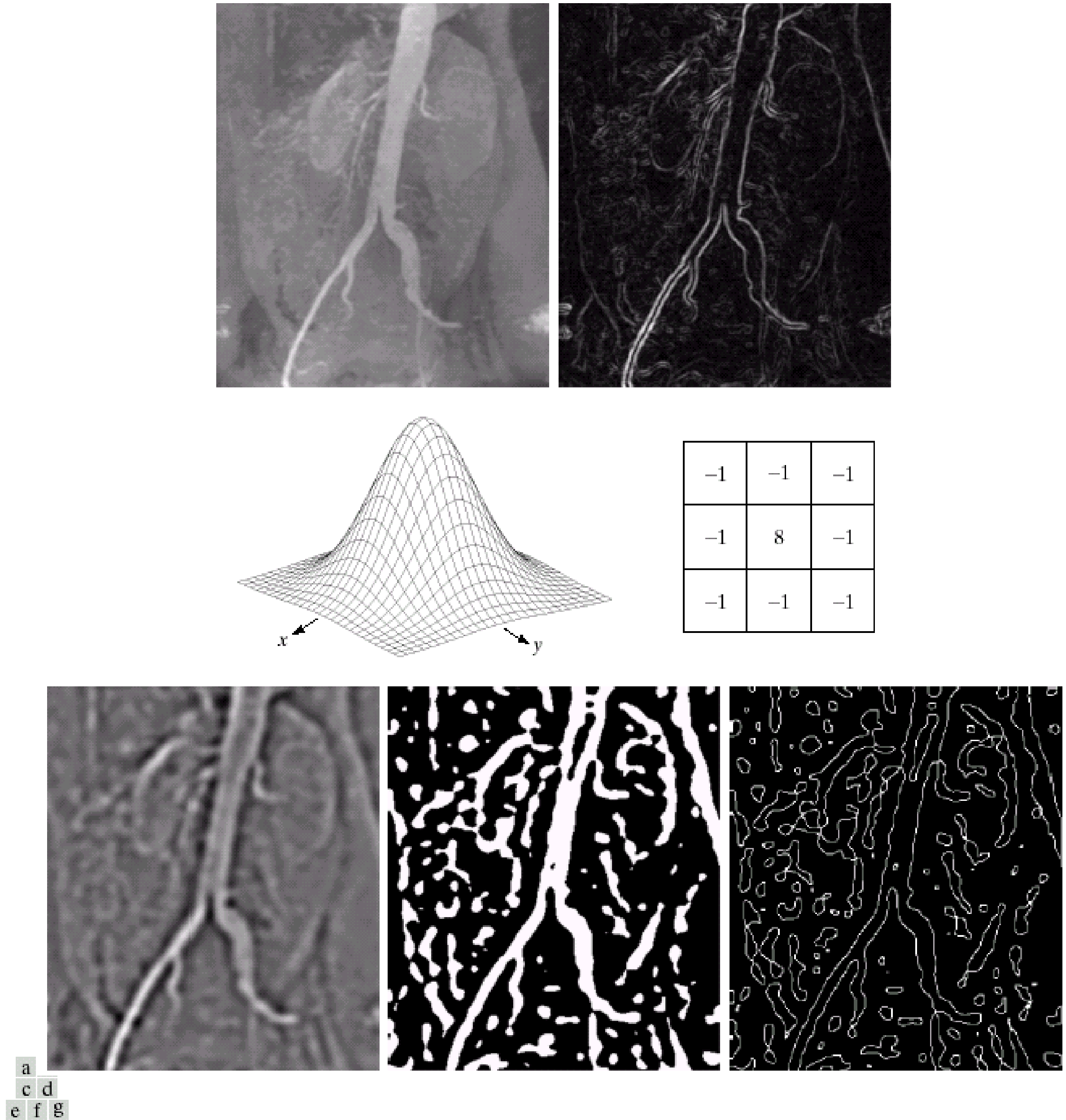


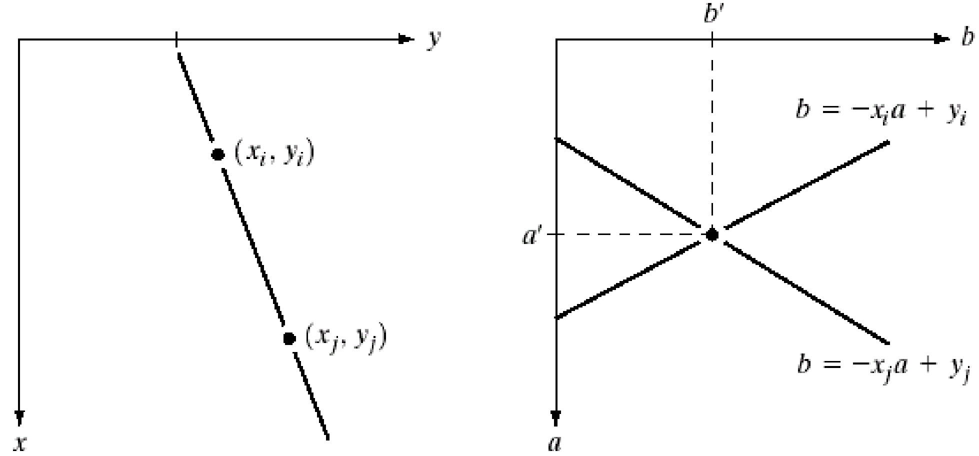
FIGURE 10.15 (a) Original image. (b) Sobel gradient (shown for comparison). (c) Spatial Gaussian smoothing function. (d) Laplacian mask. (e) LoG. (f) Thresholded LoG. (g) Zero crossings. (Original image courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)

Edge Detection

- In Matlab, the **`edge`** command performs edge detection.
- Edge pixels are colored 1, non-edge pixels are colored 0.
- Also has Canny, Prewitt edge detectors
- Type **`help edge`** for more info

Hough Transform

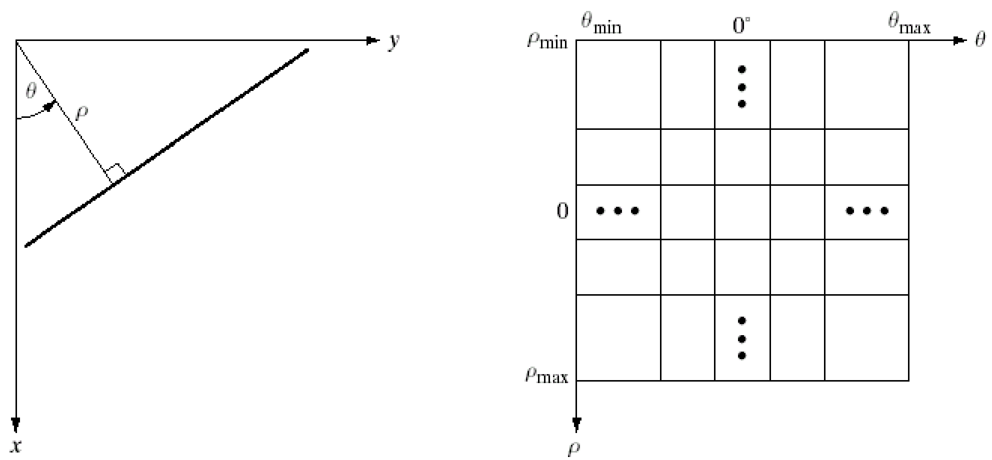
- For detecting lines, arcs



- Key idea: each point “votes” for a line in parameter space.
- The parameters with the most votes wins.

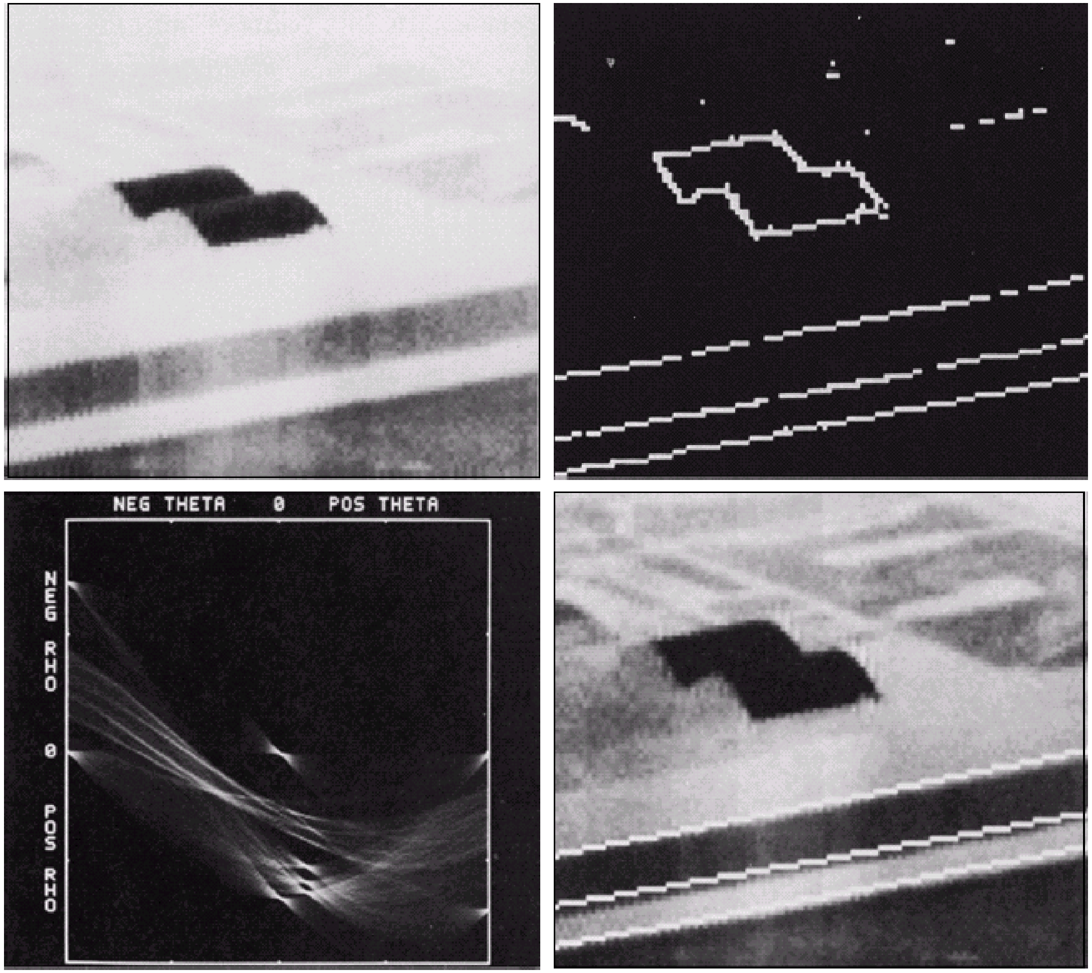
Hough Transform

- To handle vertical lines (infinite slope), ρ-θ parameters are used instead.



- Parameter values are quantized into bins. Each line point is a vote for some bins but not others.

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.)

Summary

- The Holy Grail in image understanding is object recognition.
- Segmentation is an attempt at this.
  – Albeit a poor one.
- For many tasks, segmentation is enough.
- Region segmentation is still an active area of research.
- Edge, line detection are well understood.
- Possible to detect corners, arcs, also.