Outline

- Image Segmentation
  - Introduction
  - Point/Line/Edge Detection
  - Boundary Detection

Image Segmentation: Introduction

- The goal of image segmentation is to partition the image into several parts depending on a similarity measure in a neighborhood
- Several applications include
  - Motion detection
  - Biometric recognition (e.g. face and eye segmentation)
  - Image/video compression
  - Fault detection in manufacturing
  - Medical imaging
- You can either use (i) discontinuities to separate regions, or (ii) use similarities to grow a region

Image Segmentation: Detection of Discontinuities

- Several masks can be used for this purpose depending on what needs to be detected
  - Point detection:

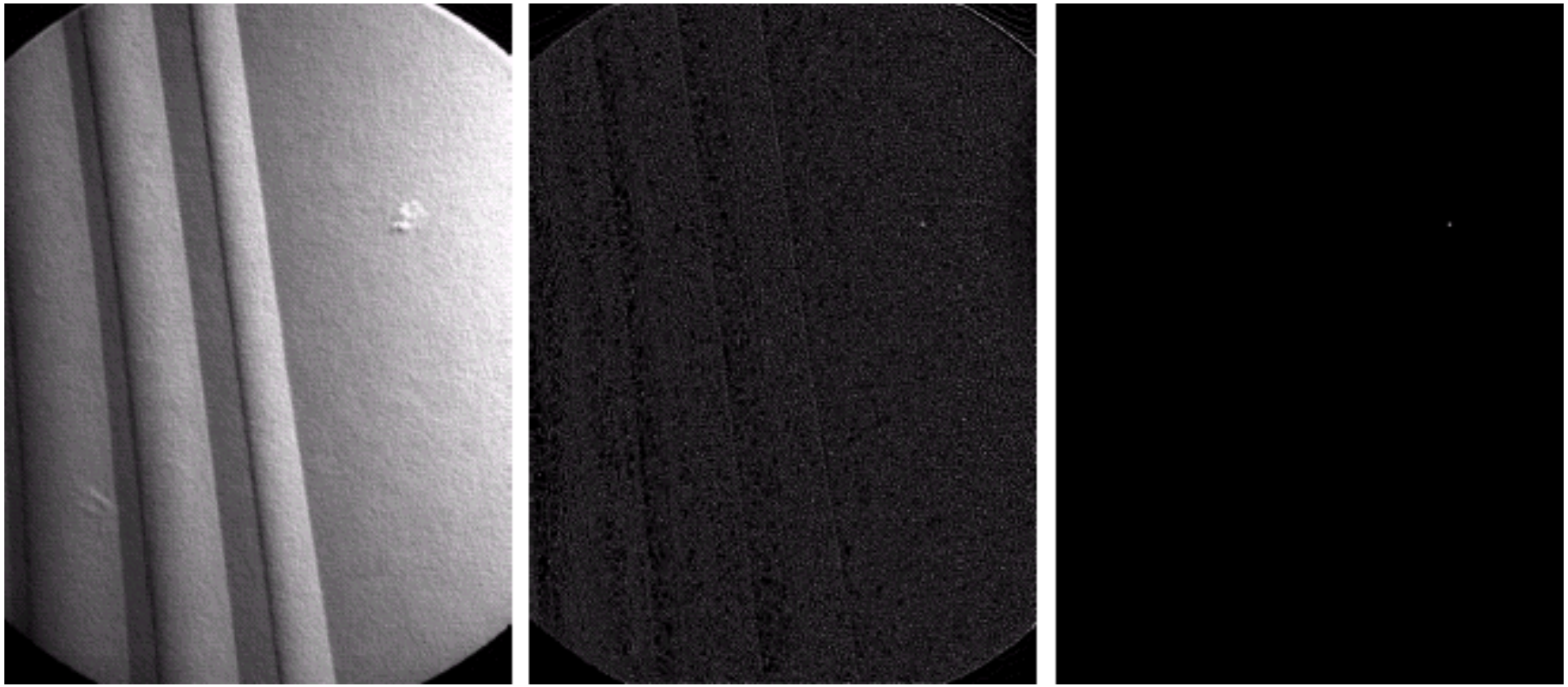
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

  - Line detection: from left to right horizontal, +45, vertical, -45

$$\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix} \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

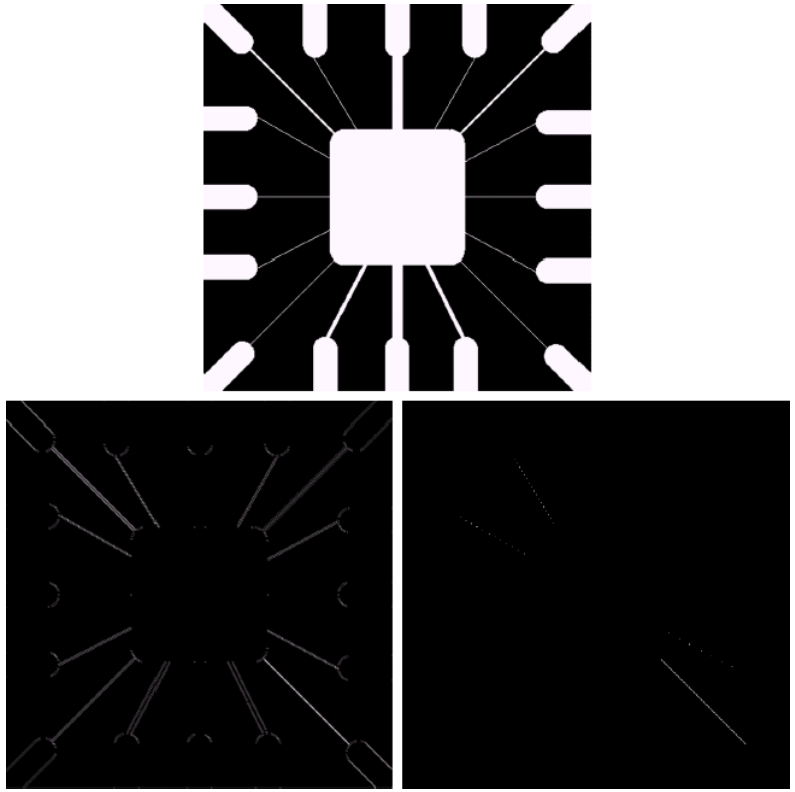
- We can see that smooth regions result in small values, whereas whenever there is a feature of interest we obtain high values
- After these masks are applied, thresholding is used to stress extract the features

Image Segmentation: Point Detection Example



Detection of a point

Image Segmentation: Line Detection Example

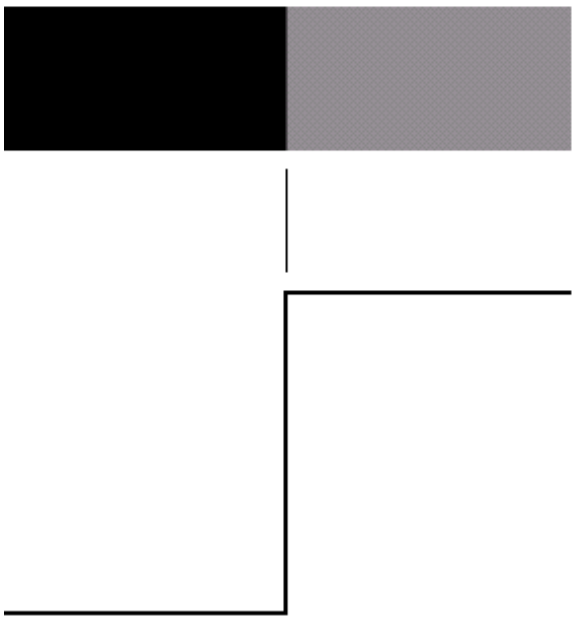


Detection of a line

Image Segmentation: Edge Detection

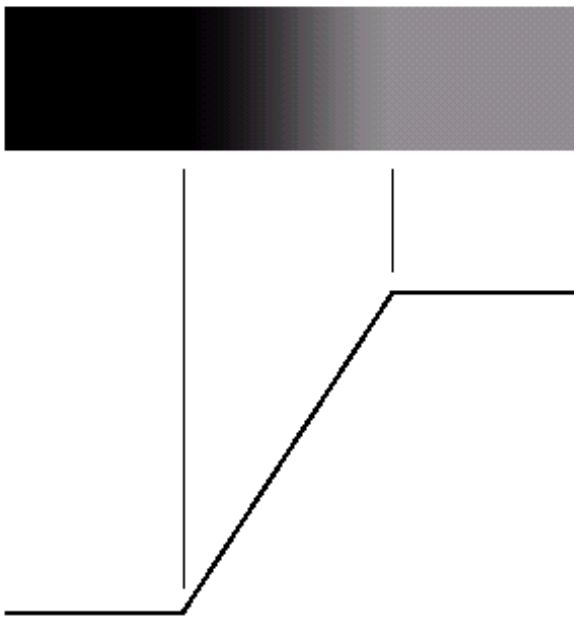
- Let us first have a look at what an edge looks like

Model of an ideal digital edge



Gray-level profile
of a horizontal line
through the image

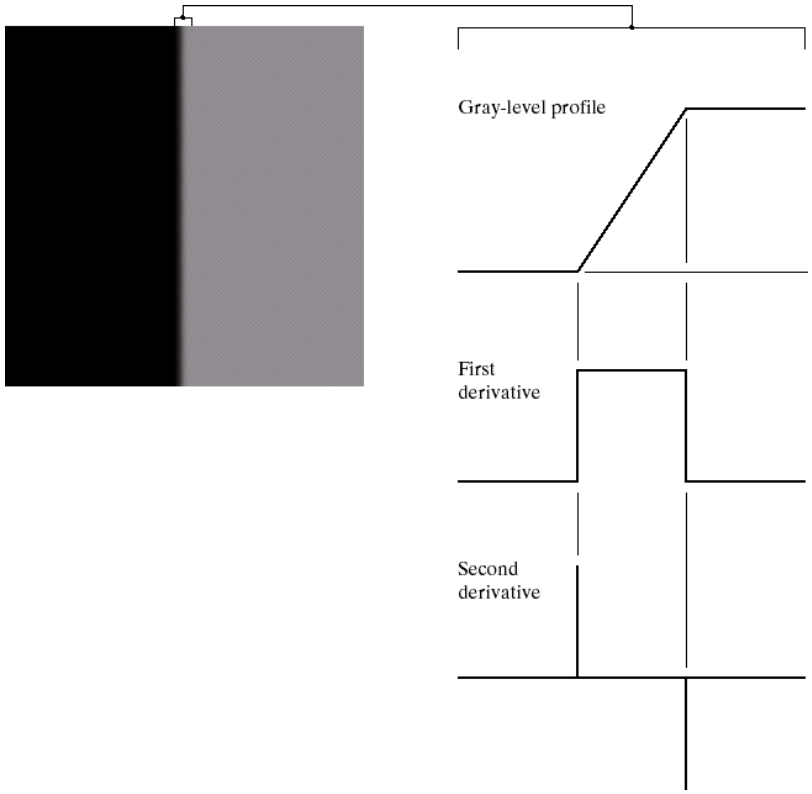
Model of a ramp digital edge



Gray-level profile
of a horizontal line
through the image

Image Segmentation: Edge Detection (Cont.)

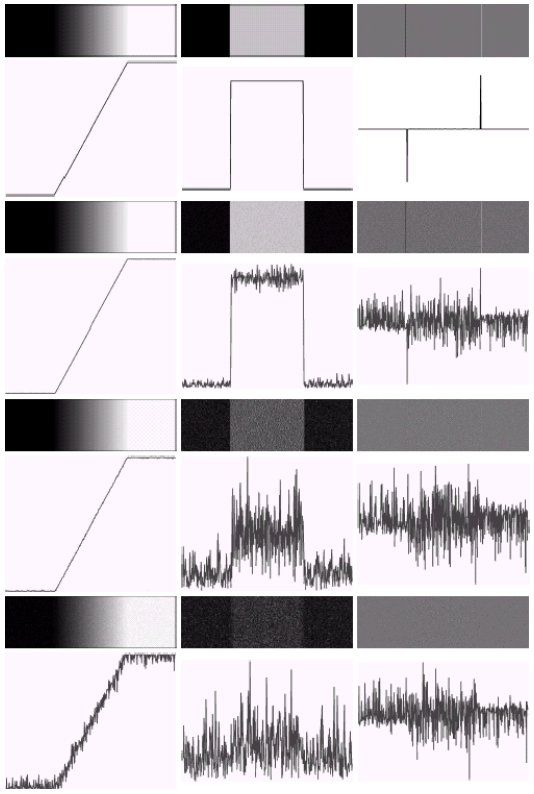
- We know from previous chapters that certain masks involving approximate derivatives can be used for edge detection



- A line connecting the peaks in the second derivative can be used to find the center of an edge

Image Segmentation: Edge Detection (Cont.)

- Although the derivative is a very useful tool in edge detection, it is very sensitive to noise (since it emphasizes abrupt changes including noise pixels)



- Then we should use smoothing before edge detection

Image Segmentation: Gradients

- The gradient of an image is

$$\nabla \boldsymbol{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial g}{\partial y} \end{bmatrix}$$

- The magnitude of this vector is related to the strength of the change, and its direction shows the direction of the change
- The amount of change given by the magnitude can be obtained with the euclidian distance or the sum of absolute differences in x and y directions

Image Segmentation: Gradients (Cont.)

- In discrete images, several ways exist to approximate these partial derivatives some of them are

$$G_{x\,\text{prewitt}} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad G_{y\,\text{prewitt}} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

$$G_{x\,\text{sobel}} = \begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad G_{y\,\text{sobel}} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Image Segmentation: Gradients

- We can similarly have operators for detecting diagonal discontinuties

$$G_{d1_{\text{prewitt}}} = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix} \quad G_{d2_{\text{prewitt}}} = \begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

$$G_{d1_{\text{sobel}}} = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \quad G_{d2_{\text{sobel}}} = \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

Image Segmentation: Gradient Image Examples

Top-left: original, top right $|G_x|$, center-left: $|G_y|$, center-right: $|G_x| + |G_y|$, bottom: diagonal edges

Image Segmentation: The Laplacian

- Remember the definition of the Laplacian

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

- We can approximate it using the following masks

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

- Usually Laplacian is not directly used because of its sensitivity to noise (second derivatives)

Image Segmentation: The Laplacian of Gaussian

- We can then combine it with smoothing to be able to use in practice, assume we use the following smoothing filter

$$h(r) = -e^{-\frac{r^2}{2\sigma^2}}$$

- Then we take the second derivative
- Since both operations are linear this is equivalent to taking the second derivative of h and then convolving with the image
- Then the total operation is given by convolving the image with

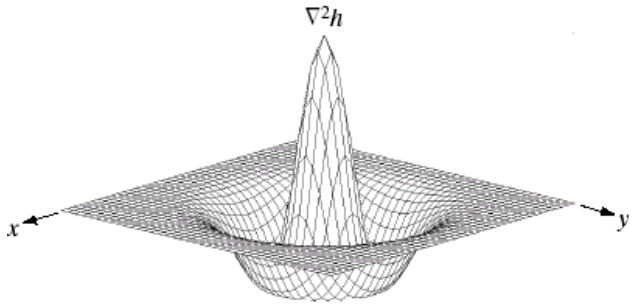
$$\nabla^2 h(r) = -\left[\frac{r^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}}$$

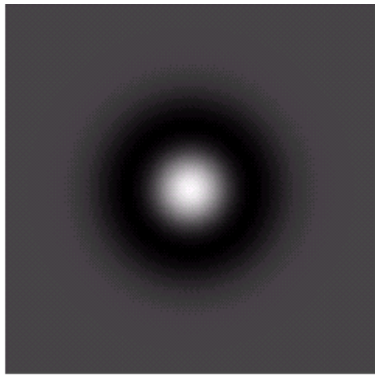
Image Segmentation: The Laplacian of Gaussian (Cont.)

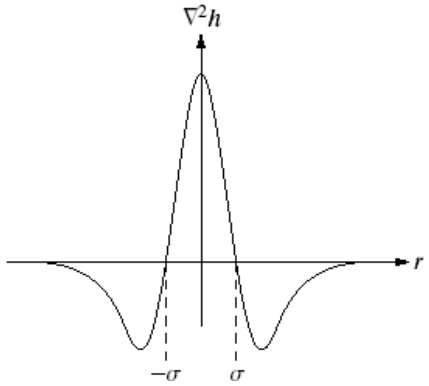
- This operation can be approximated by the following mask usually called LoG (Laplacian of Gaussian)

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

Image Segmentation: Plot of LoG

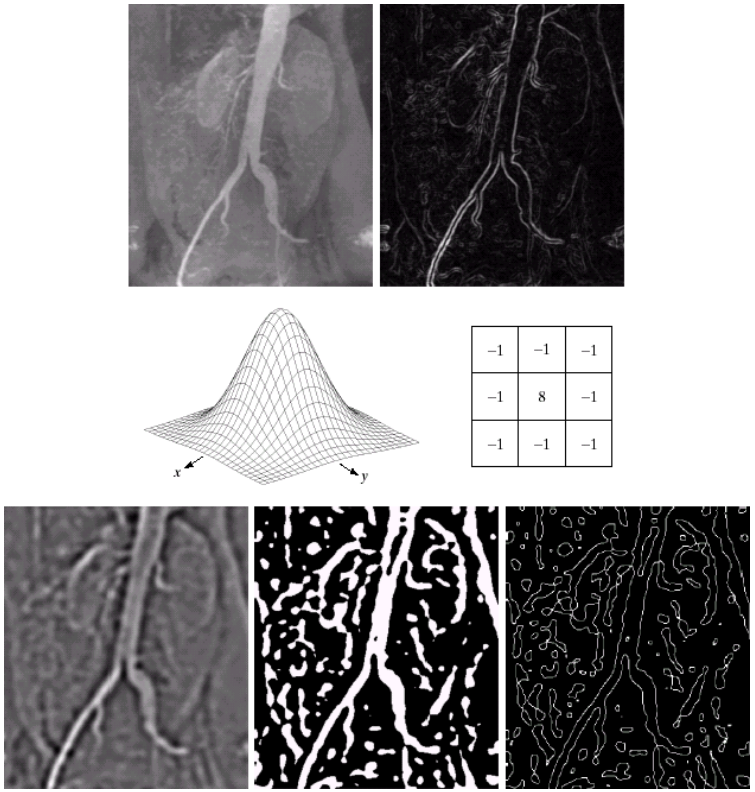






0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

Image Segmentation: LoG Image Example



Application of LoG

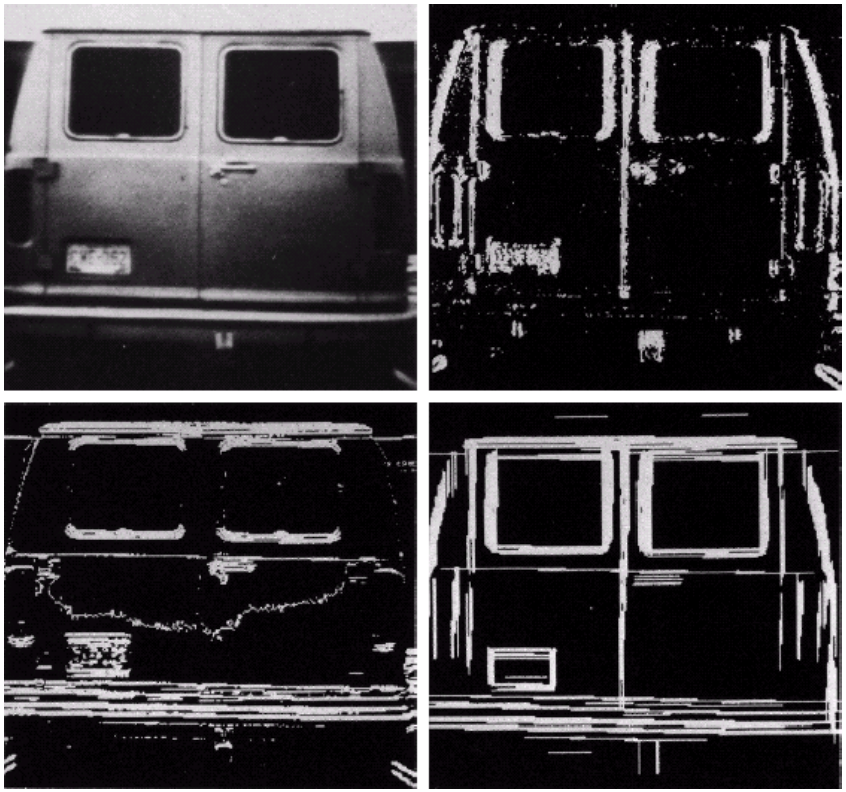
Image Segmentation: Boundary Detection

- To complete a segmentation algorithm edges are not sufficient, we need to conver these edges to boundaries that surround the segments of interest
- Detected edges are usually broken, due to noise and non-ideal objects etc., and need to be linked to form boundaries
- We can process the edges locally or globally for this purpose

Image Segmentation: Local Processing for Edge Linking

- We look at each and every point that are assigned as an edge pixel and its neighborhood
- We make the neighborhood pixel also an edge pixel if
  - the gradient magnitudes are similar: $|\nabla f(x, y) - \nabla f(x_0, y_0)| < T_1$
  - and the gradient directions are similar: $|\alpha(x, y) - \alpha(x_0, y_0)| < T_2$

Image Segmentation: Local Professing for Edge Linking - Image Example



Application of edge linking

Image Segmentation: Global Processing for Edge Linking

- Here, we look at the pixels globally, and determine if they belong to a prespecified shape (which is or is part of a boundary of a segment)
- Hough Transform can be used for this purpose
- Hought Transform is simply an algorithm to find the points that lie on a specified curve

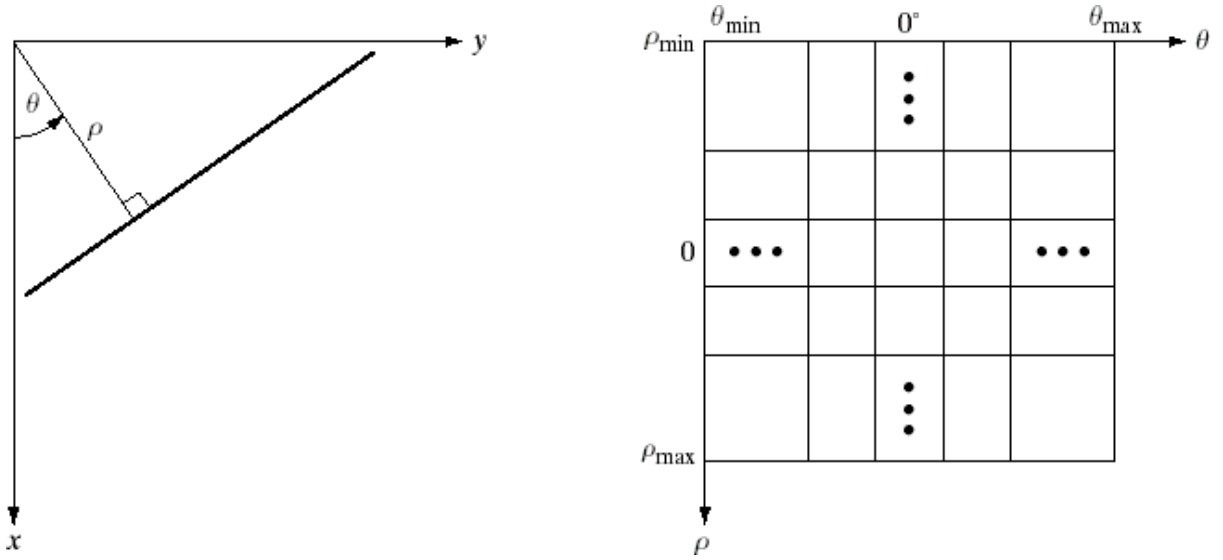
Image Segmentation: Hough Transform for Edge Linking

- Assume that we are trying to find points on a line $y_i = ax_ib$, our goal is to find the line (parametrized by a and b) so the unknown quantities are actually a and b
- Then, $b = -x_ia + y_i$
- We divide the all possible values of a and b into groups $a = [a_1, a_2, \ldots, a_K]$ and $b = [b_1, b_2, \ldots, b_K]$
- For each of the a value and for each point (x_i, y_i) we solve for b round it to the closest b value, and count the points that belongs to each of the a, b pair
- Obviously each of the (a, b) pairs represent a line, and the higher the K the more accurate and more computationally expensive the algorithm is

Image Segmentation: Hough Transform for Edge Linking

- The cartesian form of the line has the problem that, vertical lines have values of infinity
- To avoid that numerical problem we can use the normal representation of a line

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



- The Hough Transform can be used for other curves (other than line), e.g. a circle: $(x - c_1)^2 + (y - c_2)^2 = c_3^2$

Image Segmentation: Illustration of Hough Transform

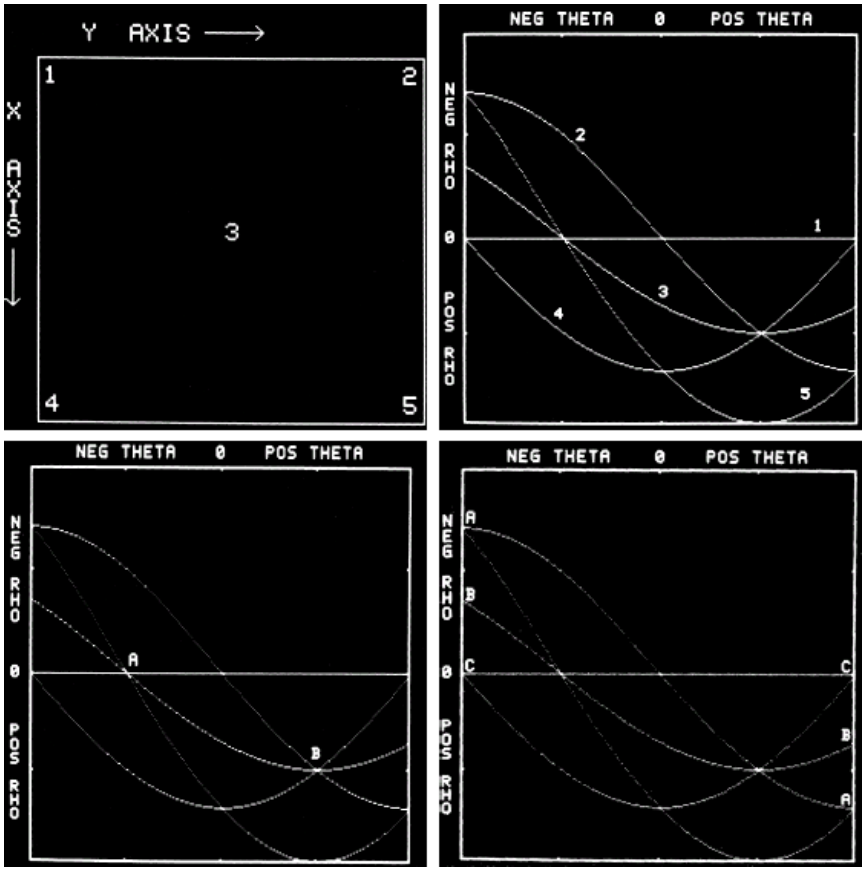
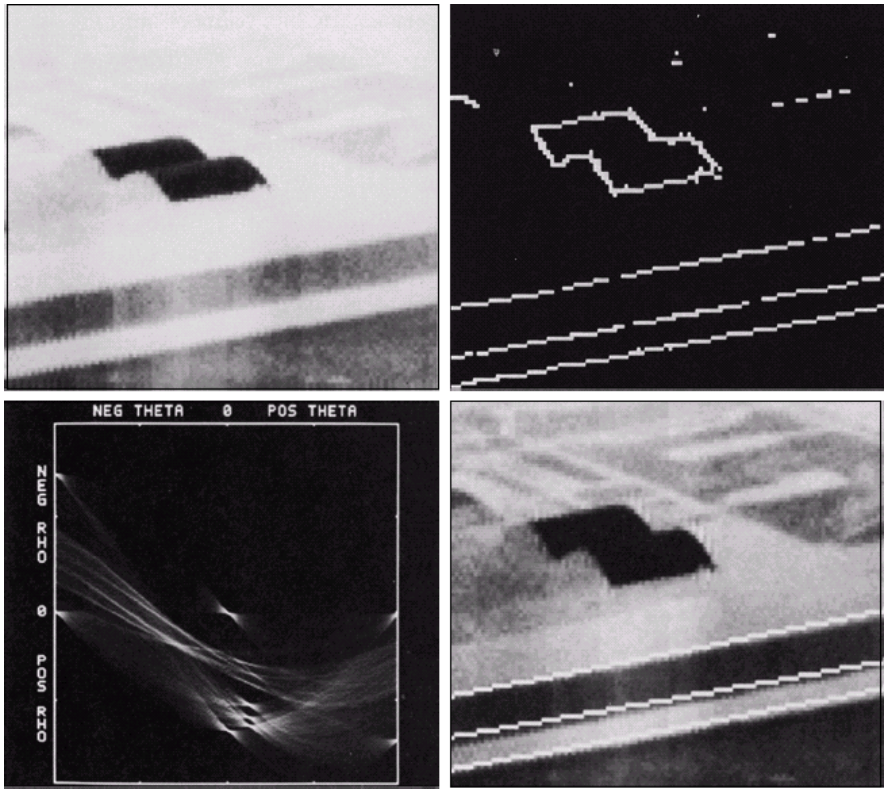


Illustration of the Hough Transform

Image Segmentationl: Hough Transform - Image Example



Application of the Hough Transform