Lecture-9

Region Segmentation

Program-I (due Feb 22)

- Implement Canny edge detector in C (either on PC, or on Unix).
  - Implement all steps of Canny
    - Compute gradient using first derivative of Gaussian masks
    - Perform non-maxima suppression using gradient direction
    - Hysteresis threshold non-maxima suppressed gradient magnitude
  - You should ask user to input image name (in pgm format), sigma for Gaussian, high and low thresholds
  - Display edge image
- Write a short report (1-2 pages):
  - Method used
  - Problems/difficulties
  - Analysis of results.
- Demonstrate your program in the vision lab (time to be fixed later) on un-seen images.

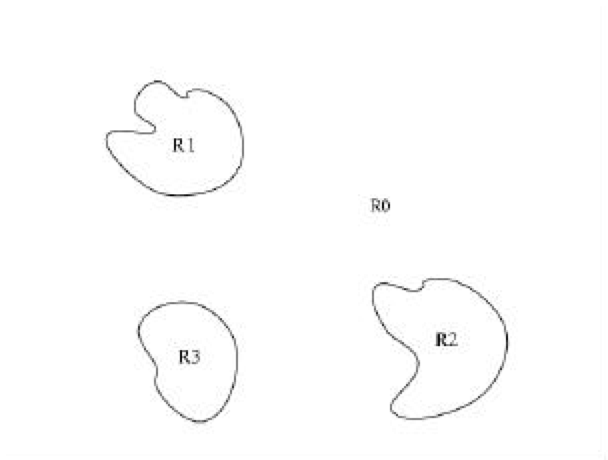
Program-I (due Feb 22)

- Implement Haralick's edge detector in C (either on PC, or on Unix).
  - Implement all steps of Canny
    - Fit a bi-cubic polynomial to a small neighborhood of a pixel (compute *k*s).
    - Compute second and third directional derivatives
    - Detect edges if the second directional derivative is zero, and third is negative
  - You should ask user to input image name (in pgm format) and rho
  - Display edge image
- Write a short report (1-2 pages):
  - Method used
  - Problems/difficulties
  - Analysis of results.
- Demonstrate your program in the vision lab (time to be fixed later) on un-seen images.

Homework-2 (Due Feb 27)

- Problems 7, 8, 9 and 10 from Chapter 2.

Segmentation



Segmentation

- Partition $f(x,y)$ into sub-images: $R_1, R_2, \ldots, R_n$ such that the following constraints are satisfied:
  - $\bigcup_{i=1}^{n} R_i = f(x,y)$
  - $R_i \cap R_j = \boldsymbol{f}, i \neq j$
  - Each sub-mage satisfies a predicate or set of predicates
    - All pixels in any sub-image musts have the same gray levels.
    - All pixels in any sub-image must not differ more than some threshold
    - All pixels in any sub-image may not differ more than some threshold from the mean of the gray of the region
    - The standard deviation of gray levels in any sub-image must be small.

Simple Segmentation

$$B(x,y) = \begin{cases} 1 & \text{if } f(x,y) < T \\ 0 & \text{Otherwise} \end{cases}$$

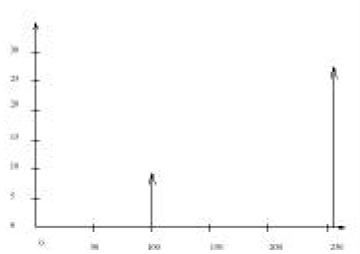
$$B(x,y) = \begin{cases} 1 & \text{if } T_1 < f(x,y) < T_2 \\ 0 & \text{Otherwise} \end{cases}$$

$$B(x,y) = \begin{cases} 1 & \text{if } f(x,y) \in Z \\ 0 & \text{Otherwise} \end{cases}$$

Histogram

Histogram graphs the number of pixels in an image with a Particular gray level as a function of the image of gray levels.

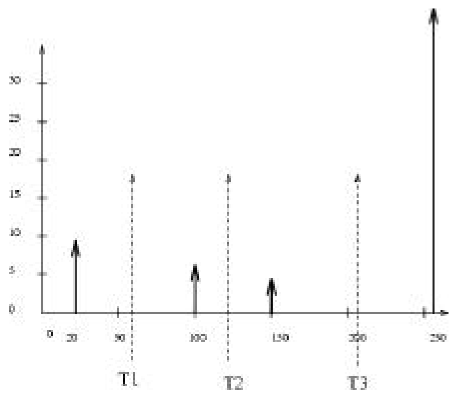
255	255	255	255	255	255
255	255	255	255	255	255
255	255	100	100	100	255
255	255	100	100	100	255
255	255	100	100	100	255
255	255	255	255	255	255



```
For (I=0, I<m, I++)
 For (J=0, J<m, J++)
   histogram[f(I,J)]++;
```

Example

255	255	255	255	255	255	255	20
255	255	255	100	100	255	20	20
255	255	255	100	100	255	20	20
255	255	255	100	100	255	20	20
255	255	255	255	255	255	20	20
255	255	255	255	255	255	255	255
150	150	255	255	255	255	255	255
150	150	255	255	255	255	255	255



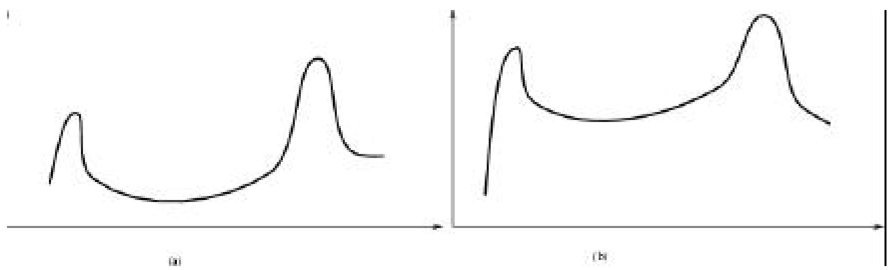
Segmentation Using Histogram

$$B_1(x,y) = \begin{cases} 1 & \text{if } 0 < f(x,y) < T_1 \\ 0 & \text{Otherwise} \end{cases}$$

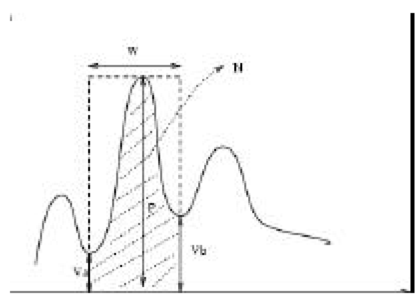
$$B_2(x,y) = \begin{cases} 1 & \text{if } T_1 < f(x,y) < T_2 \\ 0 & \text{Otherwise} \end{cases}$$

$$B_3(x,y) = \begin{cases} 1 & \text{if } T_2 < f(x,y) < T_3 \\ 0 & \text{Otherwise} \end{cases}$$

Realistic Histogram



Peakiness Test



$$Peakiness = \left(1 - \frac{(V_a + V_b)}{2P}\right)\left(1 - \frac{N}{(W.P)}\right)$$

Connected Component

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 & a & 0 \\ b & b & 0 & a & a \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c & c & 0 \\ 0 & d & 0 & c & 0 \end{bmatrix}$$

4

Connectedness

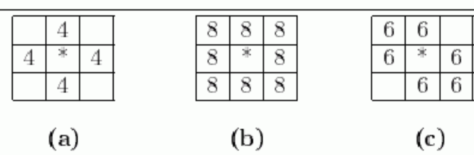


Figure 3.6: Pixel connectedness. (a) 4-connected. (b) 8-connected. (c) 6-connected.

Connected Component

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 & a & 0 \\ b & b & 0 & a & a \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c & c & 0 \\ 0 & c & 0 & c & 0 \end{bmatrix}$$

8

Recursive Connected Component Algorithm

1. Scan the binary image left to right, top to bottom.
2. If there is an unlabeled pixel with a value of '1' assign a new label to it.
3. Recursively check the neighbors of the pixel in step 2 and assign the same label if they are unlabeled with a value of '1'.
4. Stop when all the pixels of value '1' have been labeled.

Figure 3.7: Recursive Connected Component Algorithm.

Sequential

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 & a & 0 \\ b & b & 0 & a & a \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c & c & 0 \\ 0 & d & c & c & 0 \end{bmatrix} \quad d=c$$

Sequential Connected Component Algorithm

1. Scan the binary image left to right, top to bottom.
2. If an unlabeled pixel has a value of '1', assign a new label to it according to the following rules:

$$\begin{matrix} & 0 \\ 0 & 1 \end{matrix} \rightarrow \begin{matrix} & 0 \\ 0 & L \end{matrix} \qquad \begin{matrix} & 0 \\ L & 1 \end{matrix} \rightarrow \begin{matrix} & 0 \\ L & L \end{matrix}$$

$$\begin{matrix} & L \\ 0 & 1 \end{matrix} \rightarrow \begin{matrix} & L \\ 0 & L \end{matrix} \qquad \begin{matrix} & L \\ M & 1 \end{matrix} \rightarrow \begin{matrix} & L \\ M & L \end{matrix} \quad (\text{Set } L = M).$$

3. Determine equivalence classes of labels.
4. In the second pass, assign the same label to all elements in an equivalence class.

Figure 3.8: Sequential Connected Component Algorithm.

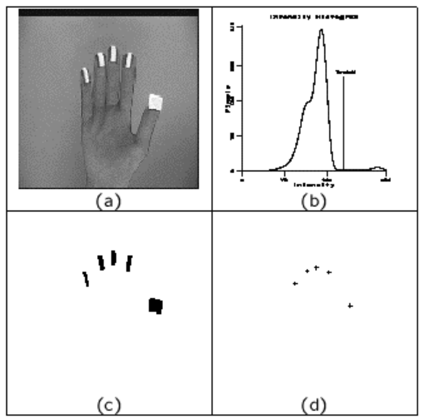
Recursive

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 & a & 0 \\ b & b & 0 & a & a \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c & c & 0 \\ 0 & c & c & c & 0 \end{bmatrix}$$

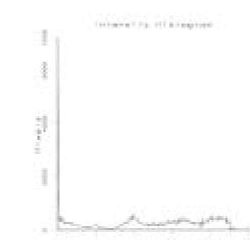
Steps in Segmentation Using Histogram

1. Compute the histogram of a given image.
2. Smooth the histogram by averaging peaks and valleys in the histogram.
3. Detect good peaks by applying thresholds at the valleys.
4. Segment the image into several binary images using thresholds at the valleys.
5. Apply connected component algorithm to each binary image find connected regions.

Example: Detecting Fingertips

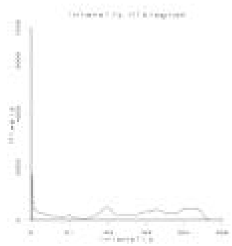


Example-II

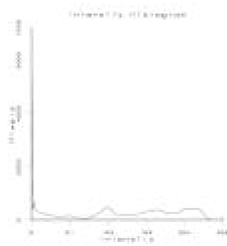


93 peaks

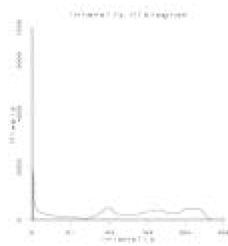
Smoothed Histograms



Smoothed histogram
(averaging using mask
Of size 5)
54 peaks (once)
After peakiness 18



Smoothed histogram
21 peaks (twice)
After peakiness 7



Smoothed histogram
11 peaks (three times)
After peakiness 4

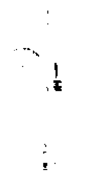
Regions

(0,40)

(40, 116)

Regions

(116,243)



(243,255)