Binary Image Processing

Introduction to Computer Vision
CSE 152
Lecture 8

Announcements

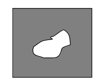
- Homework 1 is due Apr 24, 11:59 PM
- Homework 2 will be assigned this week
- Reading:
  - Chapter 3 Image processing

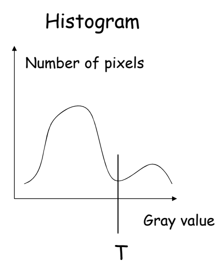
Binary System Summary

1. Acquire images and binarize (tresholding, color labels, etc.).
2. Possibly clean up image using morphological operators.
3. Determine regions (blobs) using connected component exploration
4. Compute position, area, and orientation of each blob using moments
5. Compute features that are rotation, scale, and translation invariant using Moments (e.g., Eigenvalues of normalized moments).

Histogram-based Segmentation

Ex: bright object on dark background:





- Select threshold
- Create binary image:

  I(x,y) < T → O(x,y) = 0

  I(x,y) ≥ T → O(x,y) = 1

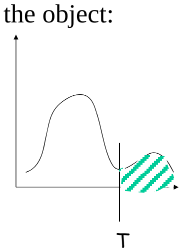
[From Octavia Camps]

How do we select a Threshold?

- Manually determine threshold experimentally.
  - Good when lighting is stable and high contrast.
- Automatic thresholding
  - P-tile method
  - Mode method
  - Peakiness detection

P-Tile Method

- If the *size* of the object is approx. known, pick T such that the area under the histogram corresponds to the size of the object:



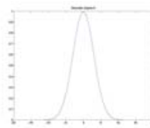
[From Octavia Camps]

Mode Method

- Model intensity in each region R_i as "constant" + $N(0,\sigma_i)$:

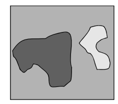
If $(x,y) \in R_i$ then, $I(x,y) = \mu_i + n_i(x,y)$

$$p(n_i) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{1}{2}\frac{n_i^2}{\sigma_i^2}}$$

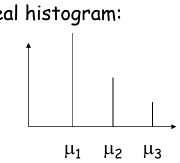


$$E(n_i) = 0 \qquad E(n_i^2) = \sigma_i^2$$

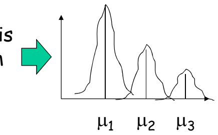
Example: Image with 3 regions



Ideal histogram:



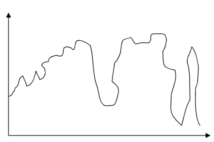
If above image is noisy, histogram looks like



- Approximate histogram as being comprised of multiple Gaussian modes.
  - How many modes?
  - Where are they centered, width
- Altenatively,the valleys are good places for thresholding to separate regions.

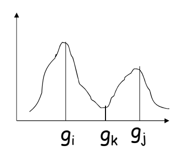
Finding the peaks and valleys

- It is a not trivial problem:



"Peakiness" Detection Algorithm

- Find the two **HIGHEST LOCAL MAXIMA** that **MINIMUM DISTANCE APART**: g_i and g_j
- Find **lowest point** between them: g_k
- Measure "peakiness":
  - $\min(H(g_i),H(g_j))/H(g_k)$
- Find (g_i,g_j,g_k) with highest peakiness



Regions

What is a region?

- "Maximal connected set of points in the image with same brightness value" (e.g., 1)
- Two points are *connected* if there exists a continuous path joining them.
- Regions can be *simply connected* (For every pair of points in the region, all smooth paths can be smoothly and continuously deformed into each other). Otherwise, region is *multiply connected* (holes)

Connected Regions

		1	1	1											
		1	1	1						1	1	1	1		
		1	1	1					1	1			1		
					1	1			1	1			1	1	
				1	1	1				1	1	1	1		
					1										

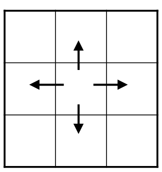
- What are the connected regions in this binary image?
- Which regions are contained within which region?

Connected Regions

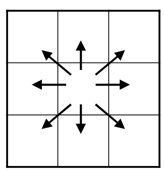
		1							1	1	1	1	1		
	1	1						1					1		
	1	1					1	1					1	1	
	1	1							1	1	1	1	1		

- What the connected regions in this binary image?
- Which regions are contained within which region?

Four & Eight Connectedness



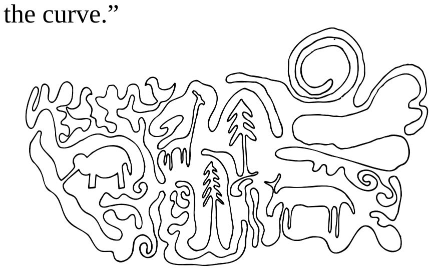
Four Connected



Eight Connected

Jordan Curve Theorem

- "Every closed curve in R^2 divides the plane into two region, the 'outside' and 'inside' of the curve."



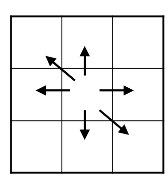
Problem of 4/8 Connectedness

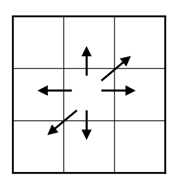
		1	1	1		
	1				1	
	1				1	
		1	1	1		

- 8 Connected:
  - 1's form a closed curve, but background only forms one region.
- 4 Connected
  - Background has two regions, but ones form four "open" curves (no closed curve)

To achieve consistency with respect to Jordan Curve Theorem

1. Treat background as 4-connected and foreground as 8-connected.
2. Use 6-connectedness





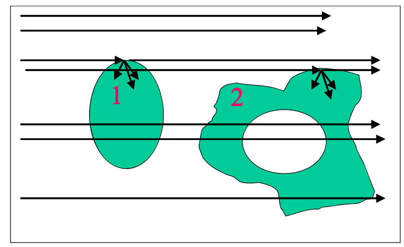
Recursive Labeling
Connected Component Exploration

```
Procedure Label (Pixel)
BEGIN
 Mark(Pixel) <- Marker;
 FOR neighbor in Neighbors(Pixel) DO
    IF Image (neighbor) = 1 AND Mark(neighbor)=NIL THEN
           Label(neighbor)
END

BEGIN Main
 Marker <- 0;
 FOR Pixel in Image DO
    IF Image(Pixel) = 1 AND Mark(Pixel)=NIL THEN
           BEGIN
              Marker <- Marker + 1;
              Label(Pixel);
           END;
END
```

Globals:
Marker: integer
Mark: Matrix same size as Image, initialized to NIL

Recursive Labeling
Connected Component Exploration



Some notes

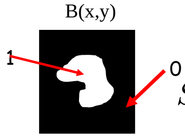
- Once labeled, you know how many regions (the value of Marker)
- From Mark matrix, you can identify all pixels that are part of each region (and compute area)
- How deep does stack go?
- Iterative algorithms
- Parallel algorithms

Properties extracted from binary image

- A tree showing containment of regions
- Properties of a region
  1. Genus – number of holes
  2. Centroid
  3. Area
  4. Perimeter
  5. Moments (e.g., measure of elongation)
  6. Number of “extrema” (indentations, bulges)
  7. Skeleton

Moments

B(x,y)



The region S is defined as:

$$S = \{(x,y) | B(x,y) = 1\}$$

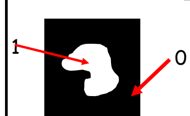
Given a pair of non-negative integers (j,k) the *discrete* (j,k)th *moment* of S is defined as:

$$M_{jk}(S) = \sum_{(x,y)\in S} x^j y^k$$

$$M_{jk} = \sum_{x=1}^{n} \sum_{y=1}^{m} B(x,y) x^j y^k$$

- Fast way to implement computation over n by m image or window
- One object

Moments: Area



$$S = \{(x,y) | f(x,y) = 1\}$$

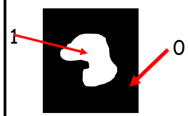
$$M_{jk}(S) = \sum_{(x,y)\in S} x^j y^k$$

Example:

$$M_{oo}(S) = \sum_{(x,y)\in S} x^0 y^0 = \sum_{(x,y)\in S} 1 = \#(S)$$

Area of S

Moments: Centroid



$$S = \{(x,y) | f(x,y) = 1\}$$

$$M_{jk}(S) = \sum_{(x,y)\in S} x^j y^k$$

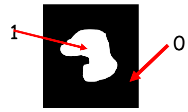
Example:

$$M_{10}(S) = \sum_{(x,y)\in S} x^1 y^0 = \sum_{(x,y)\in S} x \qquad M_{01}(S) = \sum_{(x,y)\in S} x^0 y^1 = \sum_{(x,y)\in S} y$$

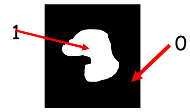
$$\frac{M_{10}(S)}{M_{oo}(S)} = \frac{\sum_{(x,y)\in S} x}{\#(S)} = \bar{x} \qquad \frac{M_{01}(S)}{M_{oo}(S)} = \frac{\sum_{(x,y)\in S} y}{\#(S)} = \bar{y}$$

Center of gravity (centroid, mean) of S

Shape recognition by Moments



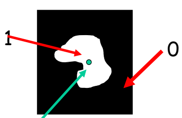
? =



Recognition could be done by comparing moments

However, moments M_{jk} are not invariant under:
- •Translation
- •Scaling
- •Rotation
- •Skewing

Central Moments



$(\bar{x}, \bar{y})$

$$S = \{(x,y) | f(x,y) = 1\}$$

$$\bar{x} = \frac{M_{10}(S)}{M_{oo}(S)} \qquad \bar{y} = \frac{M_{01}(S)}{M_{oo}(S)}$$

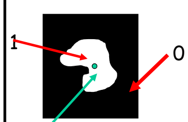
Given a pair of non-negative integers (j,k) the *central* (j,k)th *moment* of S is given by:

$$\mu_{jk}(S) = \sum_{(x,y)\in S} (x-\bar{x})^j (y-\bar{y})^k$$

Or the central moments can be computed from precomputed regular moments

$$\mu_{jk} = \sum_{m=1}^{i} \sum_{n=1}^{j} \binom{i}{m} \binom{j}{n} (-\bar{x})^{(i-m)} (-\bar{y})^{(j-n)} M_{mn}$$

Central Moments



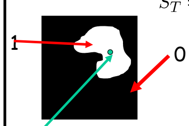
$(\bar{x}, \bar{y})$

$$S = \{(x,y) | f(x,y) = 1\}$$

$$\mu_{jk}(S) = \sum_{(x,y)\in S} (x-\bar{x})^j (y-\bar{y})^k$$

Translation by T = (a,b) :

$$S_T = \{(x^*, y^*) | x^* = x + a, y^* = y + b, (x,y) \in S\}$$



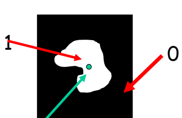
$(\bar{x}^*, \bar{y}^*)$

$$\bar{x}^* = \frac{M_{10}(S_T)}{M_{oo}(S_T)} = \bar{x} + a \qquad \bar{y}^* = \frac{M_{01}(S_T)}{M_{oo}(S_T)} = \bar{y} + b$$

$$\mu_{jk}(S_T) = \mu_{jk}(S)$$

Translation INVARIANT

Normalized Moments



$(\bar{x}, \bar{y})$

$$S = \{(x,y) | f(x,y) = 1\}$$

$$\mu_{jk}(S) = \sum_{(x,y)\in S} (x-\bar{x})^j (y-\bar{y})^k$$

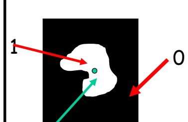
$$\sigma_x = \sqrt{\frac{\mu_{20}(S)}{M_{oo}(S)}} \qquad \sigma_y = \sqrt{\frac{\mu_{02}(S)}{M_{oo}(S)}}$$

Given a pair of non-negative integers (j,k) the *normalized* (j,k)th *moment* of S is given by:

$$m_{jk}(S) = \sum_{(x,y)\in S} \left(\frac{x-\bar{x}}{\sigma_x}\right)^j \left(\frac{y-\bar{y}}{\sigma_y}\right)^k$$

Normalized Moments

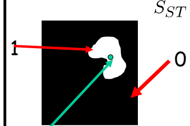
$$S = \{(x,y) | f(x,y) = 1\}$$



$(\bar{x}, \bar{y})$

Scaling by (a,c) and translating by T = (b,d) :

$$S_{ST} = \{(x^*, y^*) | x^* = ax + b, y^* = cy + d, (x,y) \in S\}$$

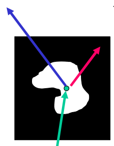


$(\bar{x}^*, \bar{y}^*)$

$$m_{jk}(S_{ST}) = m_{jk}(S)$$

Scaling and translation INVARIANT

Region orientation from Second Moment Matrix



$(\bar{x}, \bar{y})$

1. Compute second centralized moment matrix

$$\begin{bmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{bmatrix}$$

- Symmetric, positive definite matrix
- Positive Eigenvalues
- Orthogonal Eigenvectors

1. Compute Eigenvectors of Moment Matrix to obtain orientation
2. Eigenvalues are independent of orientation and translation

Binarization using Color

- Object's in robocup are distinguished by color.
- How do you binarize the image so that pixels where ball is located are labeled with 1, and other locations are 0?
- Let $C_b=(r\ g\ b)^T$ be the color of the ball.

Binarization using Color

- Let c(u,v) be the color of pixel (u,v)
- Simple method

$$b(u,v) = \begin{cases} 1 & \text{if} \left(\| \mathbf{c}(u,v) - \mathbf{c_b} \| \right)^2 \le \varepsilon \\ 0 & \text{otherwise} \end{cases}$$

- Better alternative (why?)
  - Convert **c**(u,v) to HSV space H(u,v), S(u,v) V(u,v)
  - Convert $\mathbf{c}_b$ to HSV
  - Check that HS distance is less than threshold ε and brightness (V) is greater than a treshold V>τ