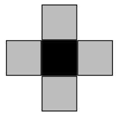
Binary Image Analysis

Jana Kosecka
http://cs.gmu.edu/~kosecka/cs482.html

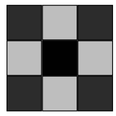


## Object extraction from binary images - connected components

- Definition: Given a pixel (i,j) its 4-neighbors are the points (i',j') such that |i-i'| + |j-j'| = 1
  - the 4-neighbors are (i±i, j) and (i,j±1)



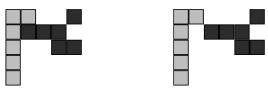
- Definition: Given a pixel (i,j) its 8-neighbors are the points (i',j') such that max(|i-i'|,|j-j'|) = 1
  - the 8- neighbors are (i, j±1), (i±1, j) and (i±1, j±1)





## Adjacency

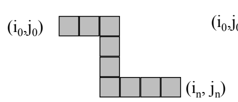
- Definition: Given two disjoint sets of pixels, A and B, A is 4-(8) adjacent to B is there is a pixel in A that is a 4-(8) neighbor of a pixel in B

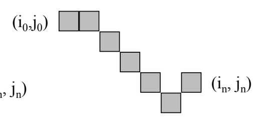




## Connected components

- Definition: A 4-(8)path from pixel $(i_0,j_0)$ to $(i_n,j_n)$ is a sequence of pixels $(i_0,j_0)$ $(i_1,j_1)$ $(i_2,j_2)$ , ... $(i_n,j_n)$ such that $(i_k, j_k)$ is a 4-(8) neighbor of $(i_{k+1}, j_{k+1})$, for k = 0, ..., n-1




Every 4-path is an 8-path!

## Connected components

- Definition: Given a binary image, B, the set of all 1's is called the foreground and is denoted by S
- Definition: Given a pixel p in S, p is 4-(8) connected to q in S if there is a path from p to q consisting only of points from S.
- The relation "is-connected-to" is an equivalence relation
  - Reflexive - p is connected to itself by a path of length 0
  - Symmetric - if p is connected to q, then q is connected to p by the reverse path
  - Transitive - if p is connected to q and q is connected to r, then p is connected to r by concatenation of the paths from p to q and q to r

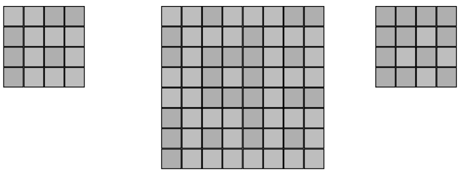


## Connected components

- Since the "is-connected-to" relation is an equivalence relation, it partitions the set S into a set of equivalence classes or components
  - these are called connected components
- Definition: *S* is the complement of S - it is the set of all pixels in B whose value is 0
  - *S* can also be partitioned into a set of connected components
  - Regard the image as being surrounded by a frame of 0's
  - The component(s) of *S* that are adjacent to this frame is called the background of B.
  - All other components of *S* are called holes



## Examples - Black = 1, Green = 0

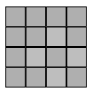


How many 4- (8) components of S?
What is the background?
Which are the 4- (8) holes?



## Background and foreground connectivity

- Use opposite connectivity for the foreground and the background
  - 8-foreground, 4-background: 4 single pixel objects and no holes
  - 4-background, 8-foreground: one 4 pixel object containing a 1 pixel hole



## Boundaries

- The ***boundary*** of S is the set of all pixels of S that have 4-neighbors in ***S.*** The boundary set is denoted as S'.
- The ***interior*** is the set of pixels of S that are not in its boundary: S-S'
- Definition: Region T ***surrounds*** region R (or R is ***inside*** T) if any 4-path from any point of R to the background intersects T
- Theorem: If R and T are two adjacent components, then either R surrounds T or T surrounds R.

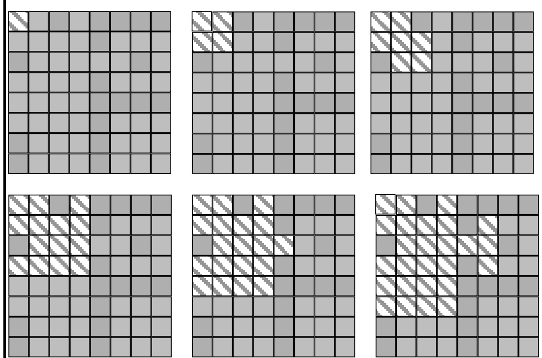


## Component labeling

- Given: Binary image B
- Produce: An image in which all of the pixels in each connected component are given a unique label.
- Solution 1: Recursive, depth first labeling
  - Scan the binary image from top to bottom, left to right until encountering a 1 (0).
  - Change that pixel to the next unused component label
  - Recursively visit all (8,4) neighbors of this pixel that are 1's (0's) and mark them with the new label



## Example





## Disadvantages of recursive algorithm

- Speed
  - requires number of iterations proportional to the largest **diameter** of any connected component in the image
- Topology
  - not clear how to determine which components of 0's are holes in which components of 1's

## Solution 2 - row scanning up and down

- Start at the top row of the image
  - partition that row into runs of 0's and 1's
  - each run of 0's is part of the background, and is given the special background label
  - each run of 1's is given a unique component label
- For all subsequent rows
  - partition into runs
  - if a run of 1's (0's) has no run of 1's(0's) directly above it, then it is potentially a new component and is given a new label
  - if a run of 1's (0's) overlaps one or more runs on the previous row give it the minimum label of those runs
    - Let a be that minimal label and let $\{c_i\}$ be the labels of all other adjacent runs in previous row. Relabel all runs on previous row having labels in $\{c_i\}$ with a

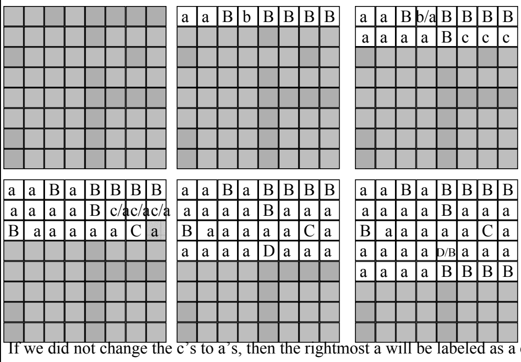


## Local relabeling

- What is the point of the last step?
  - We want the following invariant condition to hold after each row of the image is processed on the downward scan: The label assigned to the runs in the last row processed in any connected component is the **minimum** label of any run belonging to that component in the previous rows.
  - Note that this only applies to the connectivity of pixels in that part of B already processed. There may be subsequent merging of components in later rows



## Example




If we did not change the c's to a's, then the rightmost a will be labeled as a c and our invariant condition will fail.



## Upward scan

- A bottom to top scan will assign a unique label to each component
  - we can also compute simple properties of the components during this scan
- Start at the bottom row
  - create a table entry for each unique component label, plus one entry for the background if there are no background runs on the last row
  - Mark each component of 1's as being "inside" the background

## Properties

- Our goal is to recognize each connected component as one of a set of known objects
  - letters of the alphabet
  - good potatoes versus bad potatoes
- We need to associate measurements, or properties, with each connected component that we can compare against expected properties of different object types.



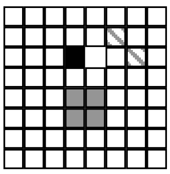
## Properties

- Area
- Perimeter
- Compactness: $P^2/A$
  - smallest for a circle: $4\pi^2r^2/\pi r^2 = 4\pi$
  - higher for elongated objects
- Properties of holes
  - number of holes
  - their sizes, compactness, etc.



How do we compute the perimeter of a connected component?

1. Count the number of pixels in the component adjacent to 0's
   - perimeter of black square would be 1
   - but perimeter of gray square, which has 4x the area, would be 4
   - but perimeter should go up as sqrt of area
2. Count the number of 0's adjacent to the component
   - works for the black and gray squares, but fails for the red dumbbell

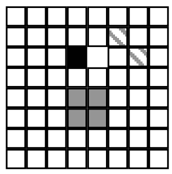


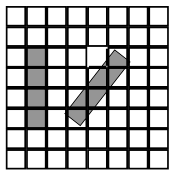


How do we compute the perimeter of a connected component?

3) Count the number of sides of pixels in the component adjacent to 0's
   - these are the **cracks** between the pixels
   - clockwise traversal of these cracks is called a crack code
   - perimeter of black is 4, gray is 8 and red is 8

- What effect does rotation have on the value of a perimeter of the digitization of a simple shape?
  - rotation can lead to large changes in the perimeter and the area!



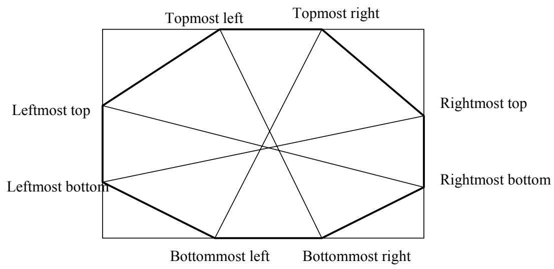


## Perimeter computation (cont.)

- We can give different weights to boundary pixels
  - 1 – vertical and horizontal pairs
  - $2^{1/2}$ – diagonal pairs
- The boundary can be approximated by a polygon line (or splines) and its length could be used
- It matters most for small (low resolution objects)



## Bounding Box and Extremal Points






## Other features

- Convex hull:
  - Create a monotone polygon from the boundary (leftmost and rightmost points in each row)
  - Connect the extremal points by removing all concavities (can be done by examining triples of boundary points)
- Minimal bounding box from the convex hull
- Deficits of convexity



## A better (and universal) set of features

- An “ideal” set of features should be independent of
  - the position of the connected component
  - the orientation of the connected component
  - the size of the connected component
    - ignoring the fact that as we “zoom in” on a shape we tend to see more detail
- These problems are solved by features called moments

## Central moments

- Let S be a connected component in a binary image
  - generally, S can be any subset of pixels, but for our application the subsets of interest are the connected components
- The (j,k)'th moment of S is defined to be

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



## Central moments

- $M_{00}$ = the area of the connected component

$$M_{00}(S) = \sum_{(x,y)\in S} x^0 y^0 = \sum_{(x,y)\in S} 1 = |S|$$

- The center of gravity of S can be expressed as

$$\bar{x} = \frac{M_{10}(S)}{M_{00}(S)} = \frac{\sum x}{|S|}$$

$$\bar{y} = \frac{M_{01}(S)}{M_{00}(S)} = \frac{\sum y}{|S|}$$



## Central moments

- Using the center of gravity, we can define the central (j,k)'th moment of S as

$$\mu_{jk} = \sum (x-\bar{x})^j (y-\bar{y})^k$$

- If the component S is translated, this means that we have added some numbers (a,b) to the coordinates of each pixel in S
  - for example, if a = 0 and b = -1, then we have shifted the component up one pixel



## Central moments

- Central moments are not affected by translations of S. Let S'={(x', y'):x'=x+a, y'=y+b, (x,y) in S}
  - The center of gravity of S' is the c.o.g. of S shifted by (a,b)

$$\bar{x}(S') = \frac{\sum x'}{|S'|} = \frac{\sum (x+a)}{|S|} = \frac{\sum x}{|S|} + \frac{\sum a}{|S|} = \bar{x} + a$$

  - The central moments of S' are the same as those of S

$$\mu_{jk}(S') = \sum (x' - \bar{x}(S'))^j (y' - \bar{y}(S'))^k$$
$$= \sum (x + a - [\bar{x}(S) + a])^j (y + b - [\bar{y}(S) + b])^k$$
$$= \sum (x-\bar{x})^j (y-\bar{y})^k = \mu_{jk}(S)$$

## Central moments

- The standard deviations of the x and y coordinates of S can also be obtained from central moments:

$$\sigma_x = \sqrt{\frac{\mu_{20}}{|S|}}$$

$$\sigma_y = \sqrt{\frac{\mu_{02}}{|S|}}$$

- We can then create a set of normalized coordinates of S that we can use to generate moments unchanged by translation and scale changes

$$\tilde{x} = \frac{x - \bar{x}}{\sigma_x} \qquad \tilde{y} = \frac{y - \bar{y}}{\sigma_y}$$



## Normalized central moments

- The means of these new variables are 0, and their standard deviations are 1. If we define the normalized moments; $m_{jk}$ as follows

$$m_{jk} = \frac{\sum \tilde{x}^j \tilde{y}^k}{M_{00}}$$

- then these moments are not changed by any scaling or translation of S
- Let S* = {(x*,y*): x* = ax + b, y* = ay + c, (x,y) in S}
  - if b and c are 0, then we have scaled S by a
  - if a is 0, then we have translated S by (b,c)



## Normalized central moments

$$m_{jk}(S^*) = \frac{\sum (\frac{x^* - \overline{x(S^*)}}{\sigma_x(S^*)})^j (\frac{y^* - \overline{y(S^*)}}{\sigma_y(S^*)})^k}{|S|}$$

$$= \frac{\sum (\frac{a^j (x - \bar{x}(S))^j}{a^j \sigma_x{}^j(S)})(\frac{a^k (y - \bar{y}(S))^k}{a^k \sigma_y{}^k(S)})}{|S|}$$

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

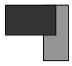
- Details of the proof are simple.



## Shortcomings of our machine vision system

- Object detection
  - thresholding will not extract intact objects in complex images
    - shading variations on object surfaces
    - texture
  - advanced segmentation methods
    - edge detection - locate boundaries between objects and background, between objects and objects
    - region analysis - find homogeneous regions; small combinations might correspond to objects.

## Shortcomings of our machine vision system



Occlusion
- What if one object is partially hidden by another?
  - properties of the partially obscured, or occluded, object will not match the properties of the class model
- Correlation - directly compare image of the "ideal" objects against real images
  - in correct overlap position, matching score will be high
- Represent objects as collection of local features such as corners of a rectangular shape
  - locate the local features in the image
  - find combinations of local features that are configured consistently with objects



## Shortcomings of our machine vision system

- Recognition of three dimensional objects
  - the shape of the image of a three dimensional object depends on the viewpoint from which it is seen
- Model a three dimensional object as a large collection of view-dependent models
- Model the three dimensional geometry of the object and mathematically relate it to its possible images
  - mathematical models of image geometry
  - mathematical models for recognizing three dimensional structures from two dimensional images



## Shortcomings of our machine vision system

- Articulated objects
  - pliers
  - derricks
- Deformable objects
  - faces
  - jello
- Amorphous objects
  - fire
  - water



## Agenda

- Advanced segmentation methods
  - edge detection
  - region recovery
- Occlusion in 2-D
  - correlation
  - clustering
- Articulations in 2-D
- Three dimensional object recognition
  - modeling 3-D shape
  - recognizing 3-D objects from 2-D images
  - recognizing 3-D objects from 3-D images
    - stereo
    - structured light range sensors