Lecture 13

Segmentation

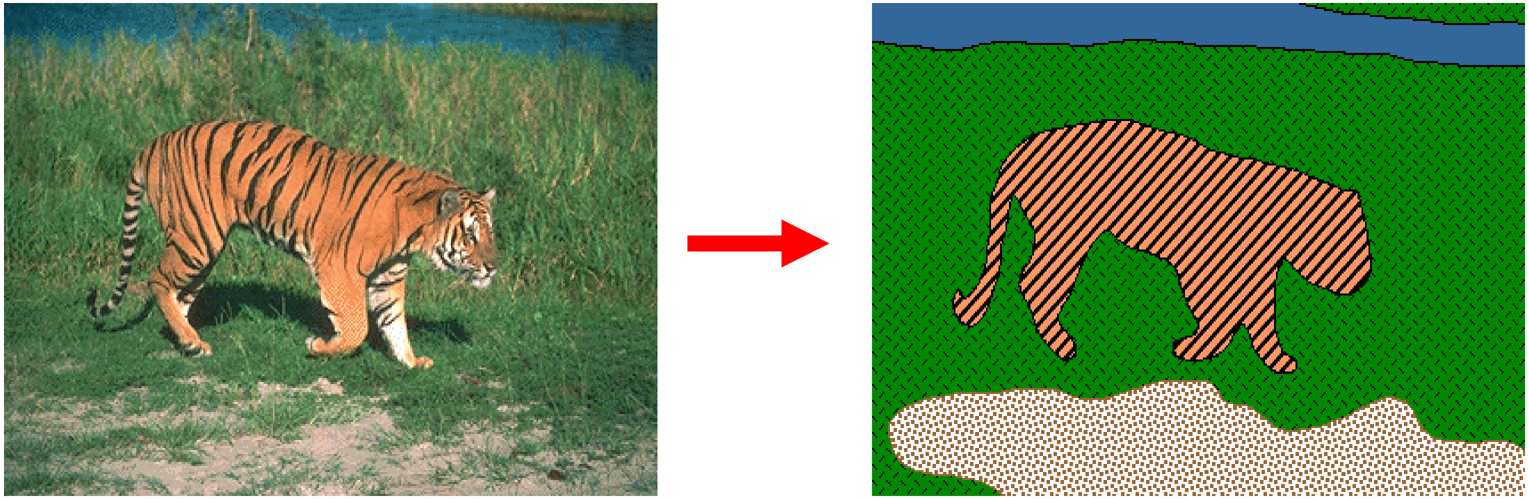
From Sandlot Science

Segmentation of today's lecture

- Histogram-based segmentation
- K-means clustering
  - EM algorithm
- Morphological operators
- Graph-cut based segmentation

- Last 15 minutes: Class photo session for Project 4

Image Segmentation



Goal: Partition an image into its constituent “objects”

Some slides adapted from Steve Seitz, Linda Shapiro

What is an object?

http://www.eventspecialistskc.com/

- Depends on the task
- If no task, must rely on general “bottom-up” image cues
- Gestalt Laws seek to formalize this
  - proximity, similarity, continuation, closure, common fate
  - see notes by Steve Joordens, U. Toronto

Image Segmentation

We will consider different methods

Already covered:

- Intelligent Scissors (contour-based, manual)

Today—automatic methods:

- K-means clustering (color-based)
- Normalized Cuts (region-based)

Recall: Image histograms

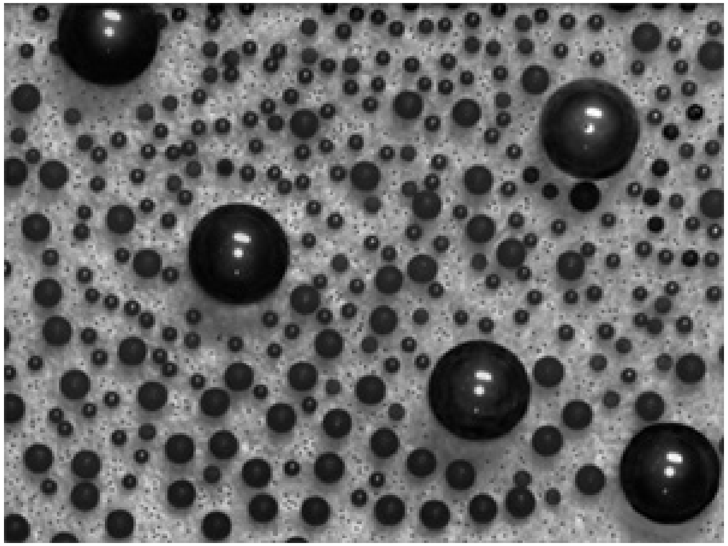
How many "orange" pixels are in this image?

- This type of question answered by looking at the *histogram*
- A histogram counts the number of occurrences of each color
  - Given an image $F[x, y] \rightarrow RGB$
  - The histogram is $H_F[c] = |\{(x, y) \mid F[x, y] = c\}|$

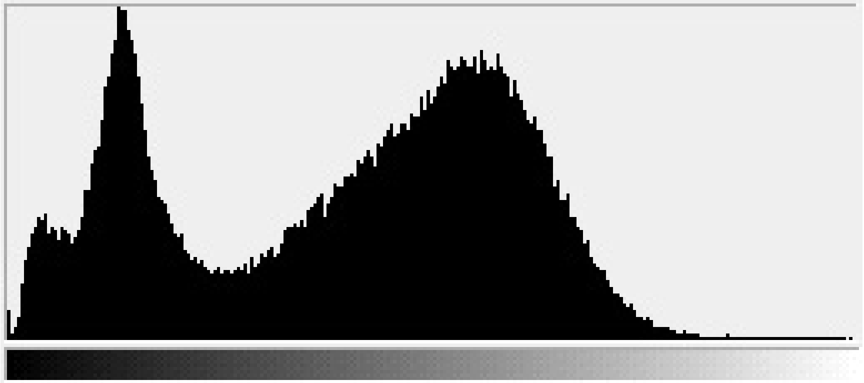
    i.e., for each color value c (x-axis), count # of pixels with that color (y-axis)

Histogram of grayscale intensities

Image



Histogram



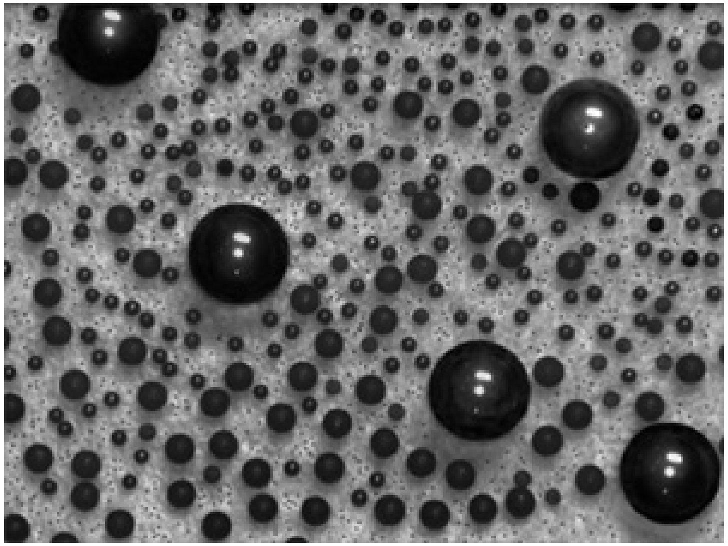
How Many Modes Are There?

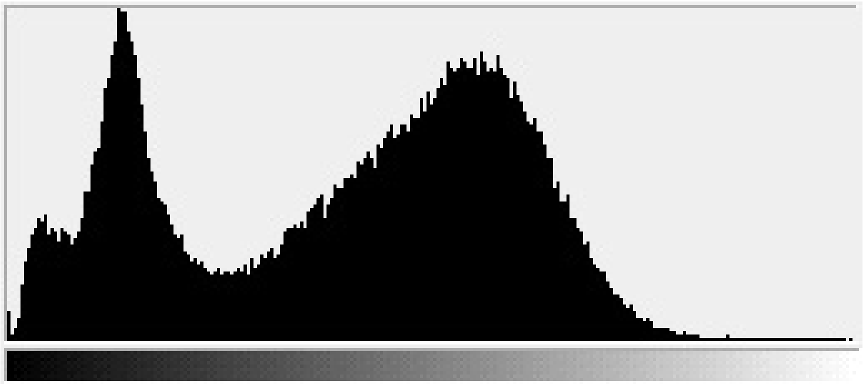
- Easy to see, hard to compute

Histogram-based segmentation

Idea:

- Break the image into K regions (segments) by reducing the number of colors to K and mapping each pixel to the closest color



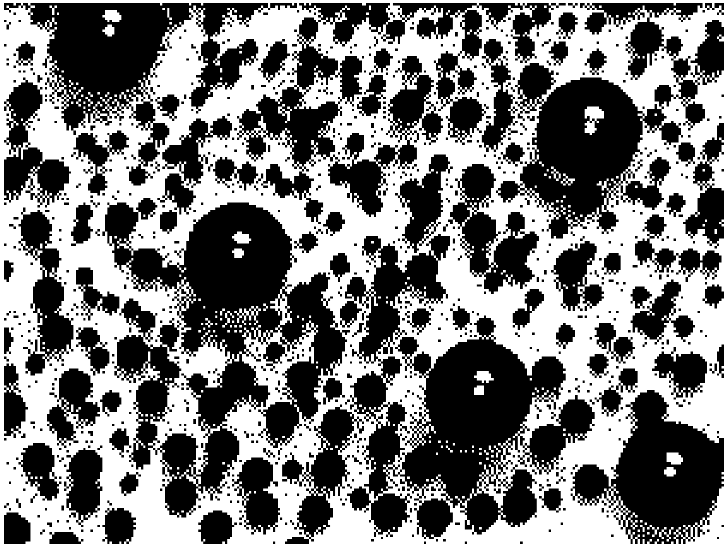


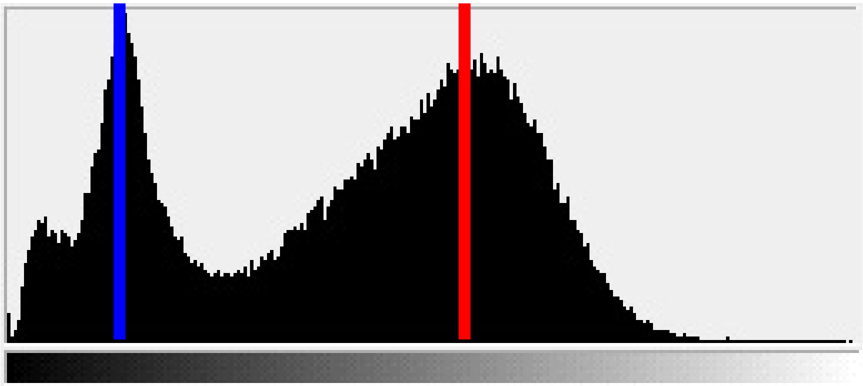
Histogram-based segmentation

Idea: Break the image into K regions (segments) by

- reducing the number of colors to K and
- assigning each pixel to the closest color

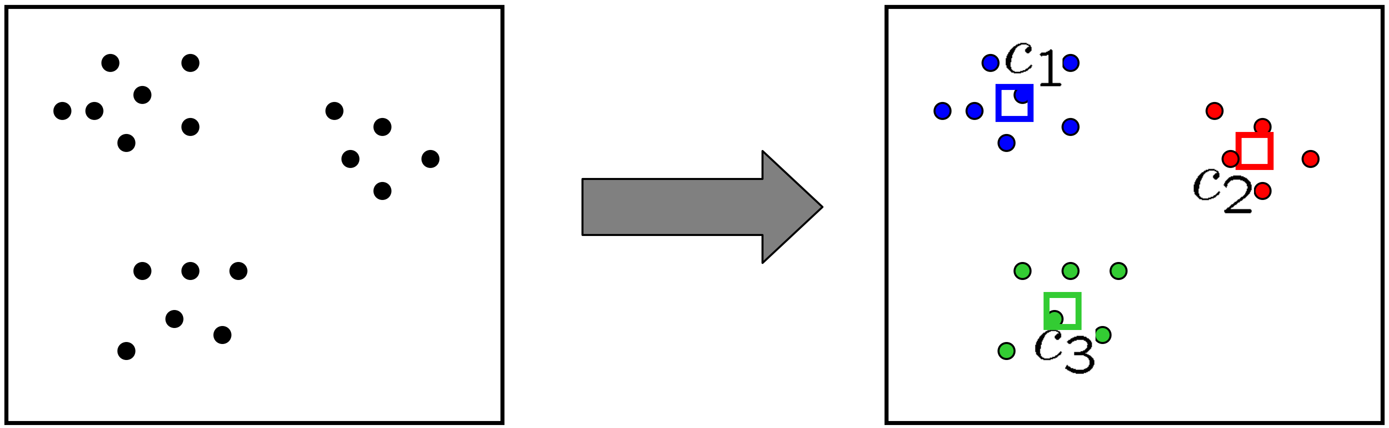
Here's what our image looks like if we use two colors (intensities)





Clustering

The idea in the previous slide can be formalized as a clustering problem



Objective

Minimize sum squared distance of each point to closest center

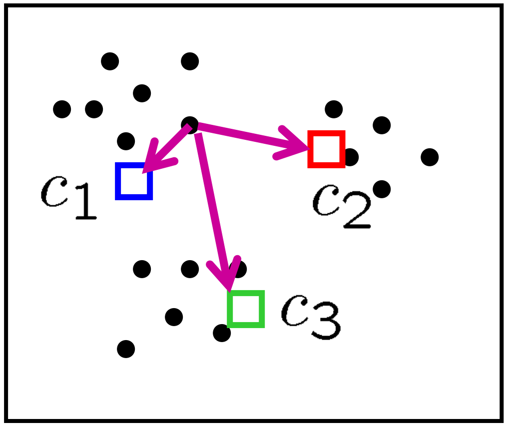
$$\sum_{\text{clusters } i} \; \sum_{\text{points p in cluster } i} \|p - c_i\|^2$$

Break it down into 2 subproblems

Suppose you are given the cluster centers c_i

Q: how do you assign points to a cluster?

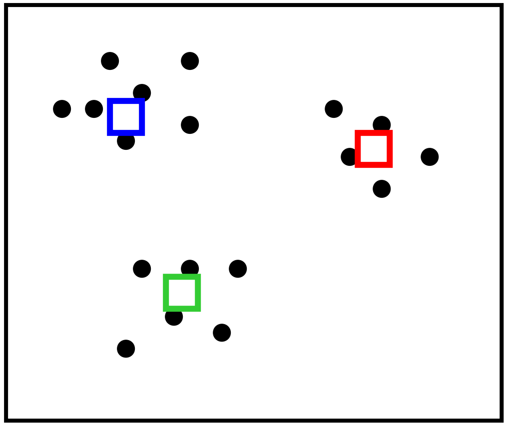
A: for each point p, choose closest c_i



Suppose you are given the points in each cluster

Q: how to re-compute each cluster's center?

A: choose c_i to be the mean of all the points in the cluster



K-means clustering

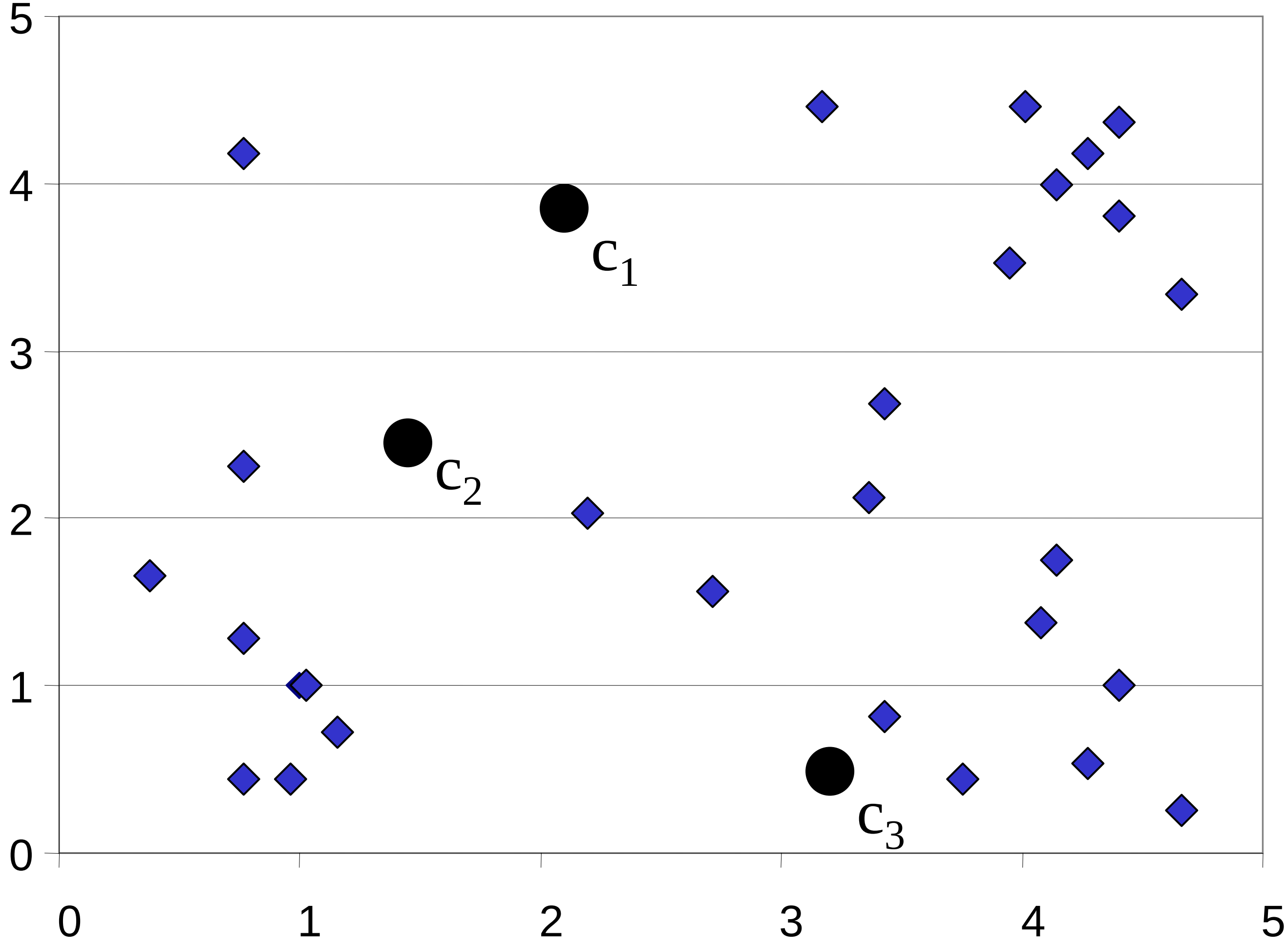
Algorithm

1. Randomly initialize the cluster centers, c_1, ..., c_K
2. Determine cluster membership
   - For each point p, find the *closest* c_i.
   - Put p into cluster i
3. Re-estimate cluster centers
   - Set c_i to be the mean of points in cluster i
4. If c_i have changed, repeat Step 2 else done.

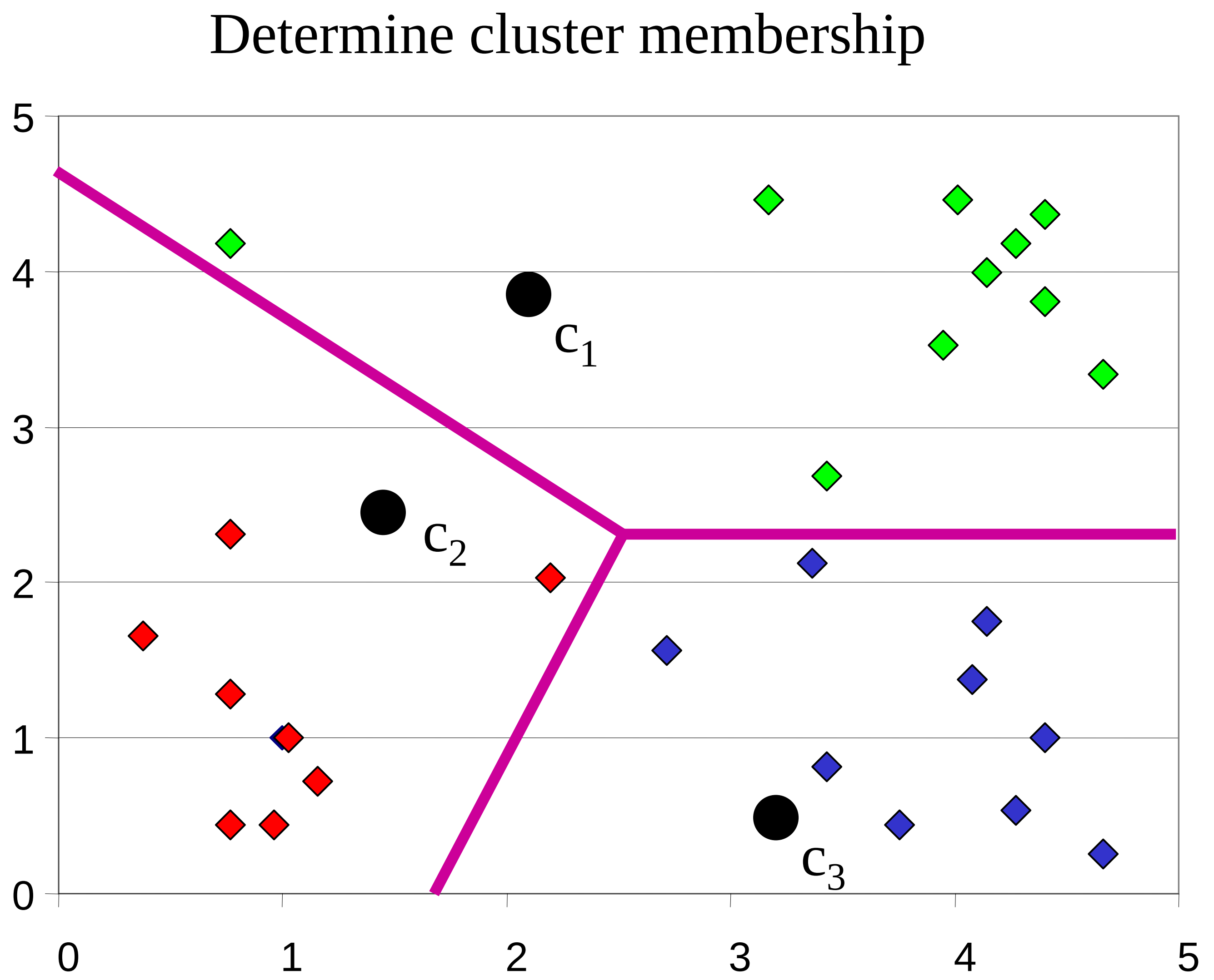
Java demo: http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html

K-means clustering example

Randomly initialize the cluster centers

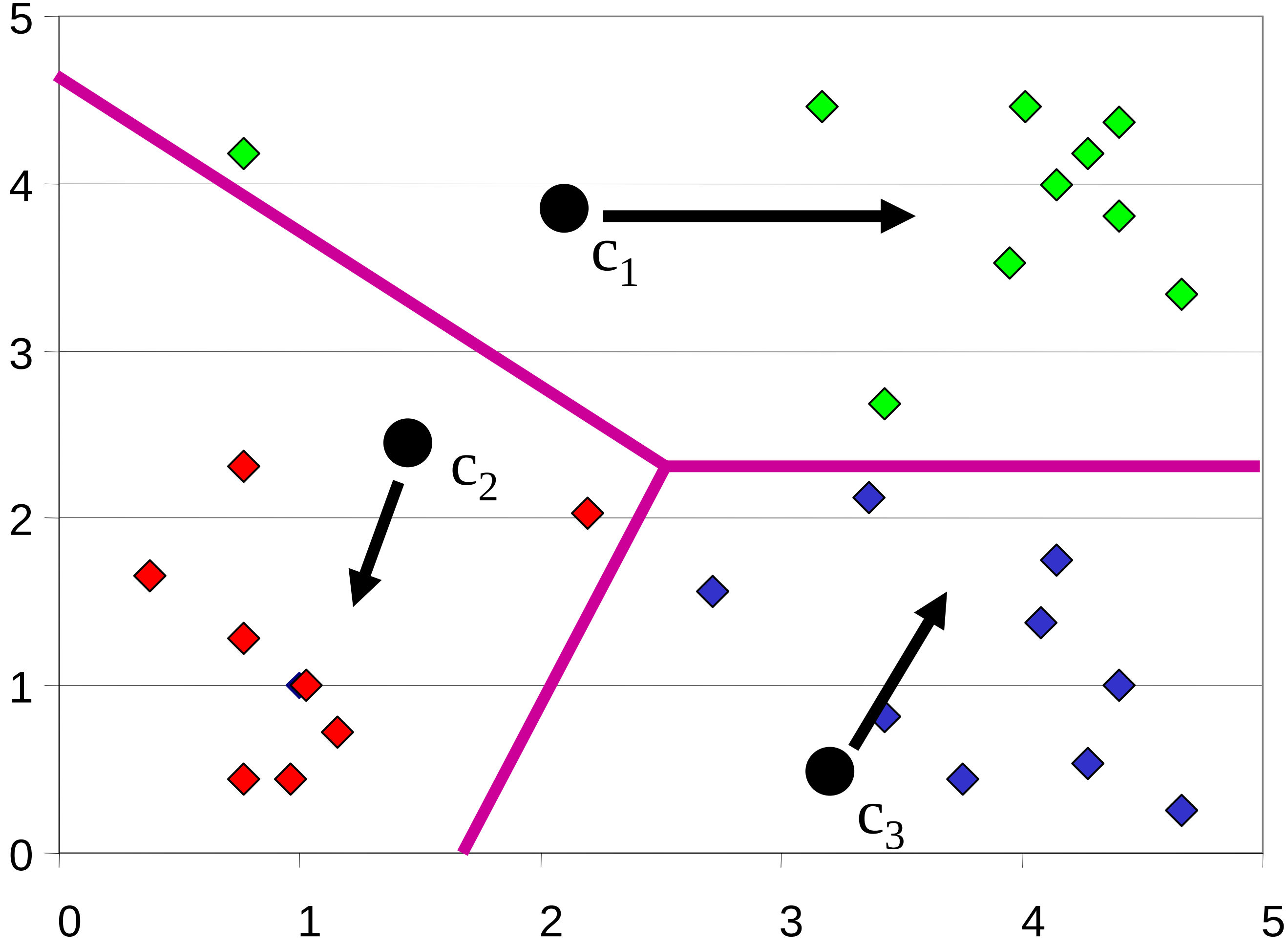


K-means clustering example

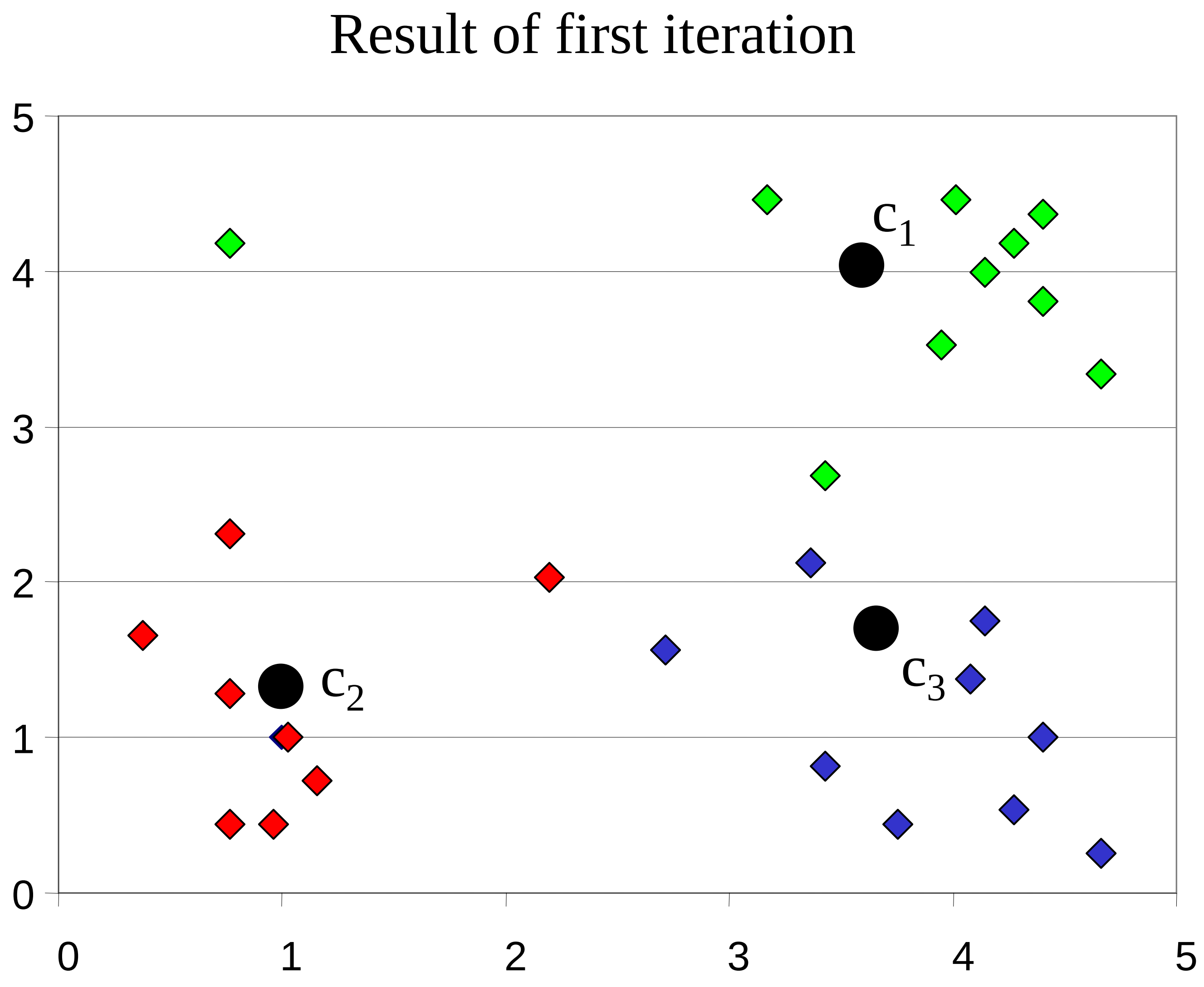


K-means clustering example

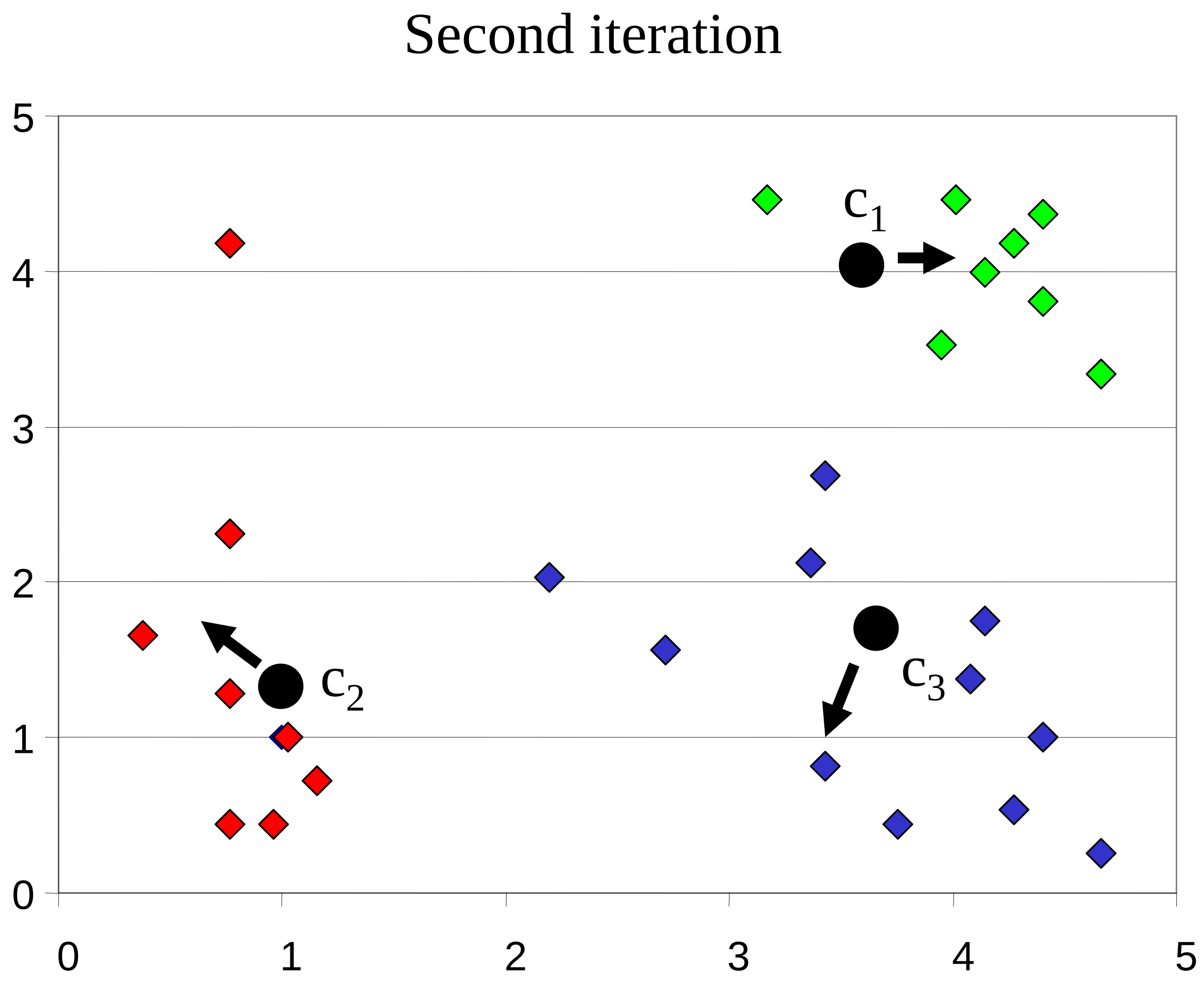
Re-estimate cluster centers



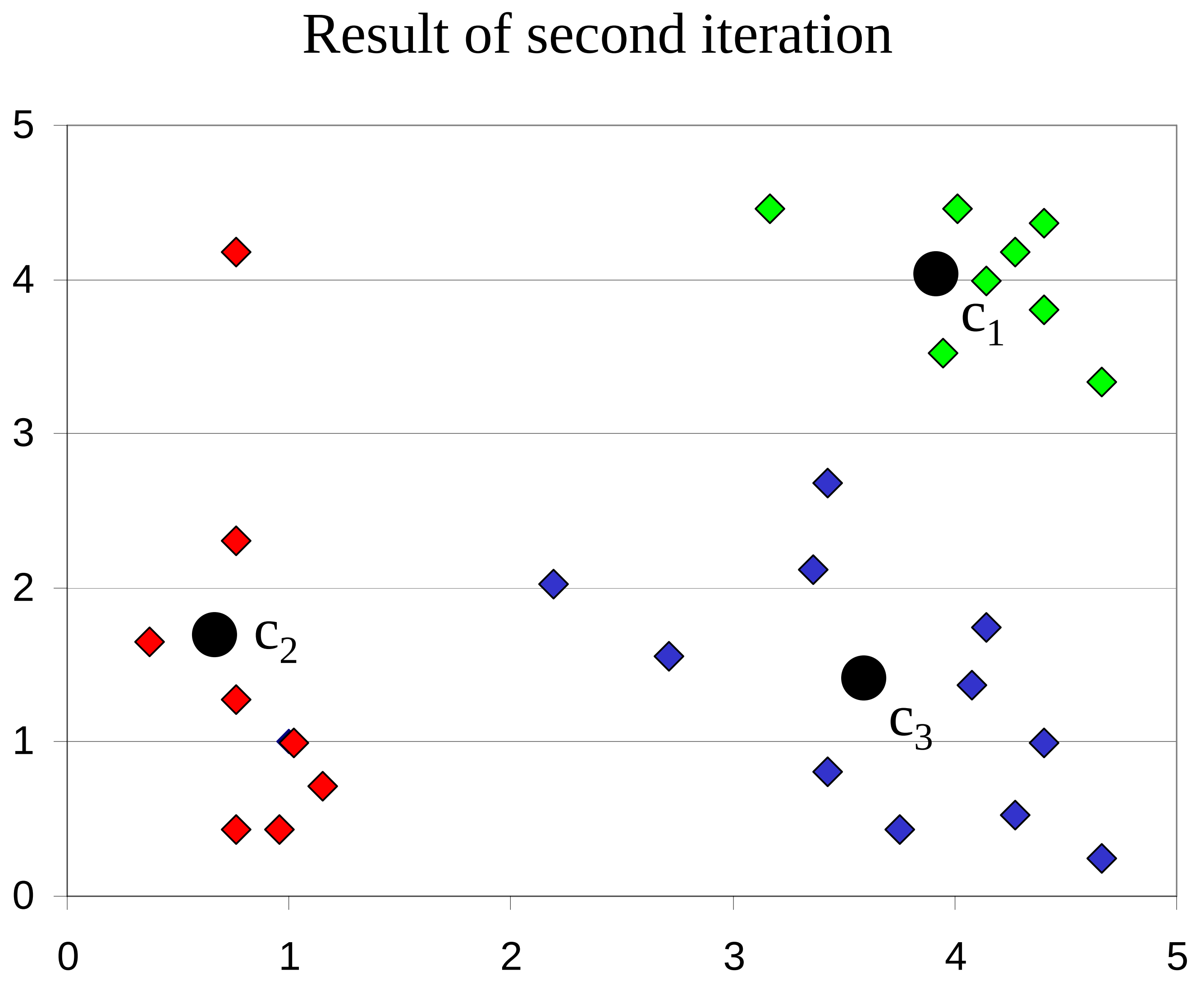
K-means clustering example



K-means clustering example



K-means clustering example



K-means clustering

Properties

- Will always converge to *some* solution
- Can be a "local minimum"
  - does not always find the global minimum of objective function:

$$\sum_{\text{clusters } i} \; \sum_{\text{points p in cluster } i} ||p - c_i||^2$$

Aside: K-means is related to the EM algorithm

- Can formalize K-means as *probability density estimation*
- Model data as a mixture of K Gaussians
- Estimate not only means but also covariances

- Expectation Maximization (EM) Algorithm overview:
  - Initialize K clusters: $C_1, \ldots, C_K$
    (μ_j, Σ_j) and $P(C_j)$ for each cluster j
  - Estimate which cluster each data point belongs to
    $p(C_j \mid x_i)$ ➡ Expectation step
  - Re-estimate cluster parameters
    $(\mu_j, \Sigma_j), p(C_j)$ ➡ Maximization step

Aside: EM algorithm

E step: Compute probability of membership in cluster based on output of previous M step ($p(x_i|C_j)$ = Gaussian(μ_j, Σ_j))

$$p(C_j \mid x_i) = \frac{p(x_i \mid C_j) \cdot p(C_j)}{p(x_i)} = \frac{p(x_i \mid C_j) \cdot p(C_j)}{\sum_j p(x_i \mid C_j) \cdot p(C_j)}$$

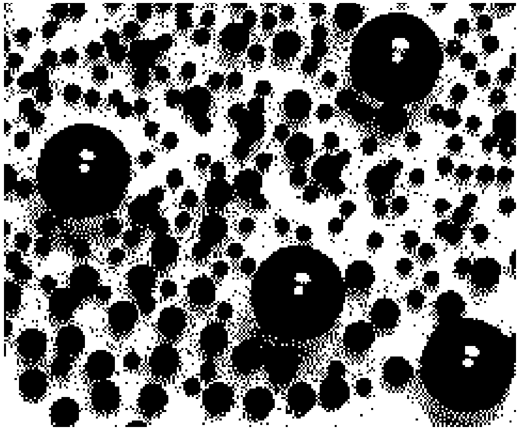
M step: Re-estimate cluster parameters based on output of E step

$$\mu_j = \frac{\sum_i p(C_j \mid x_i) \cdot x_i}{\sum_i p(C_j \mid x_i)} \qquad \Sigma_j = \frac{\sum_i p(C_j \mid x_i) \cdot (x_i - \mu_j) \cdot (x_i - \mu_j)^T}{\sum_i p(C_j \mid x_i)} \qquad p(C_j) = \frac{\sum_i p(C_j \mid x_i)}{N}$$

Cleaning up after segmentation

Problem:

- Histogram-based segmentation can produce messy regions
  - segments do not have to be connected
  - may contain holes



How can these be fixed?

Use Morphological operators! (covered in Chap. 2)

Dilation operator: $G = H \oplus F$

Assume: binary image

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	0	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$F[x, y]$

1	1	1
1	1	1
1	1	1

$H[u, v]$

Dilation: Move mask H over image F, turning F's pixels to 1 if F and H both have 1s *anywhere* in the region of overlap

- G[x,y] = 1 if H[u,v] and F[x+u-1,y+v-1] are both 1 **somewhere**
  0 otherwise
- Written as $G = H \oplus F$

Dilation operator: $G = H \oplus F$

Assume: binary image

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$F[x, y]$

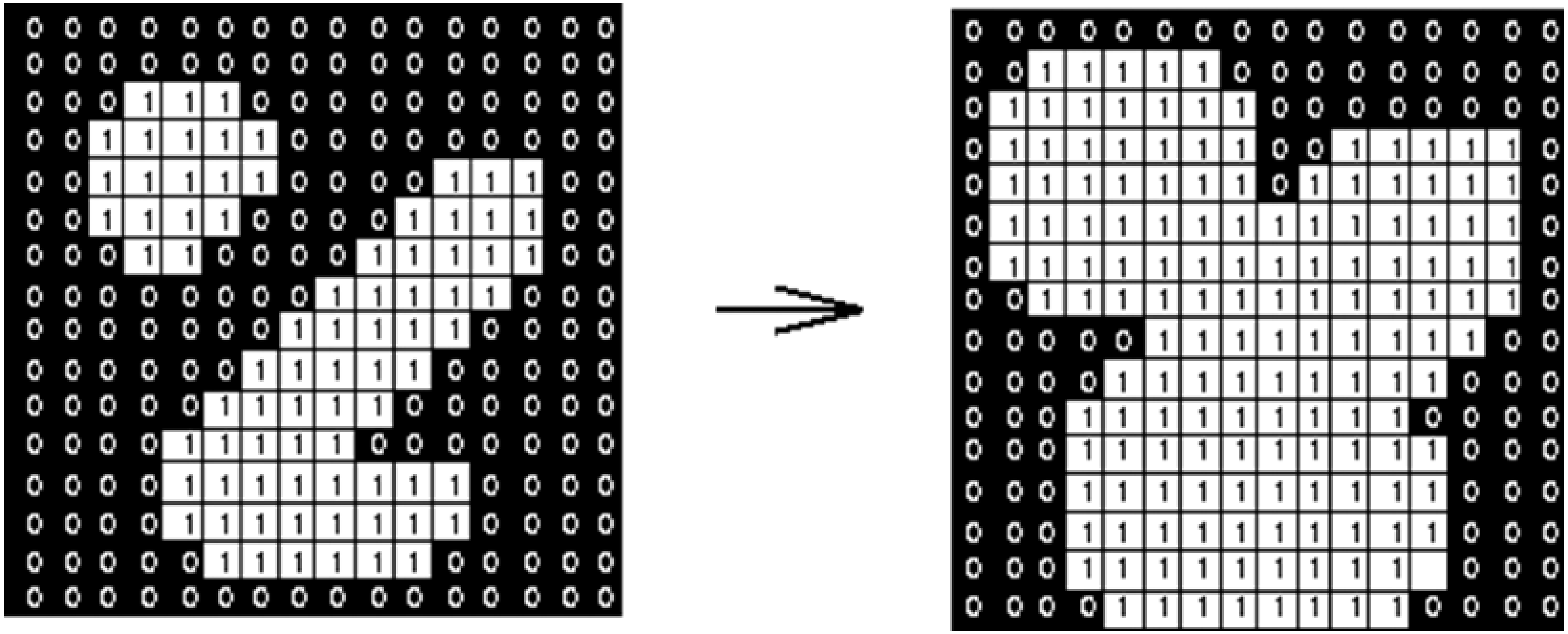
1	1	1
1	1	1
1	1	1

$H[u, v]$

Dilation: Move mask H over image F, turning F's pixels to 1 if F and H have 1s *anywhere* in the region of overlap

- G[x,y] = 1 if H[u,v] and F[x+u-1,y+v-1] are both 1 **somewhere**
  0 otherwise
- Written as $G = H \oplus F$

Dilation example



Demo: http://www.cs.bris.ac.uk/~majid/mengine/morph.html

Erosion operator: $G = H \ominus F$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	0	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$F[x, y]$

1	1	1
1	1	1
1	1	1

$H[u, v]$

Erosion: Move mask H over image F, turning F's pixels to 1 if F and H both have 1s *everywhere* in the region of overlap

- G[x,y] = 1 if F[x+u-1,y+v-1] is 1 **everywhere** that H[u,v] is 1
  0 otherwise
- Written $G = H \ominus F$

Erosion operator: $G = H \ominus F$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	1	0	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$F[x,y]$

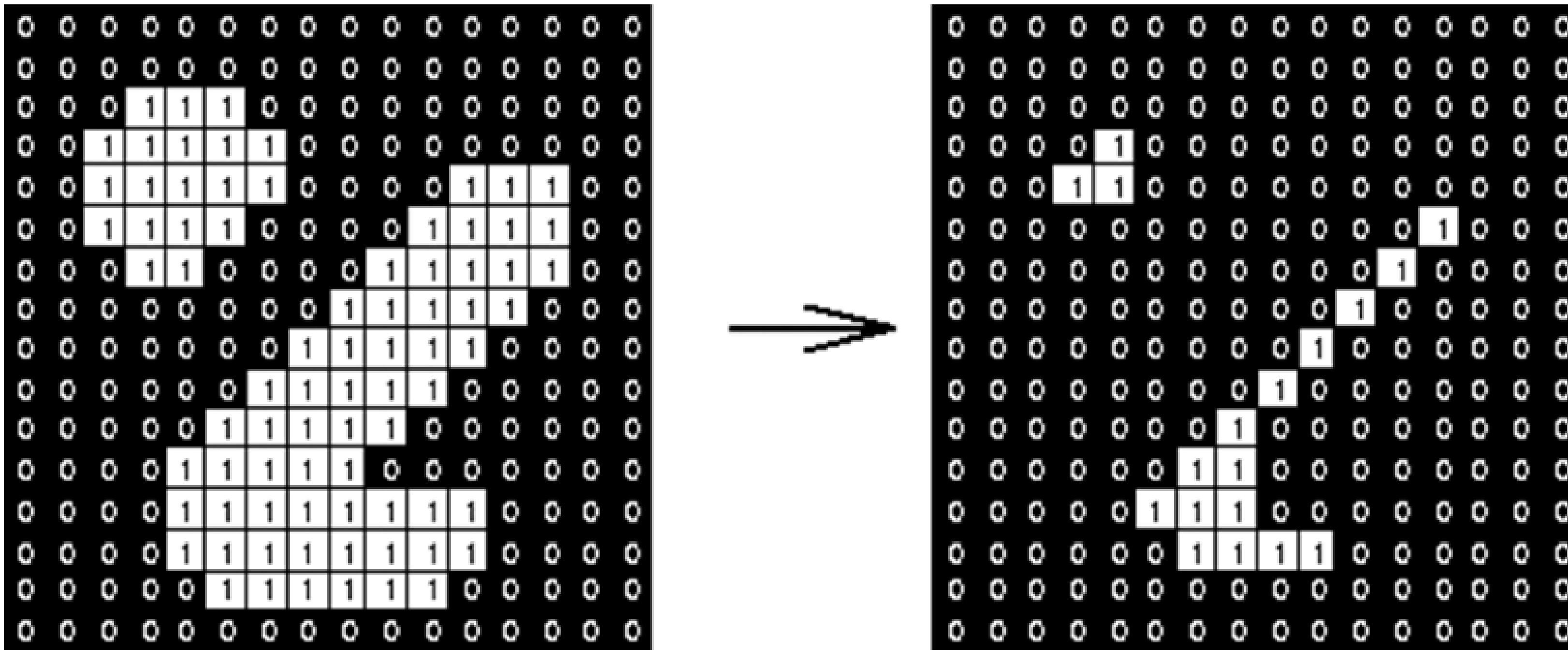
1	1	1
1	1	1
1	1	1

$H[u,v]$

Erosion: Move mask H over image F, turning F's pixels to 1 if F and H both have 1s *everywhere* in the region of overlap

- G[x,y] = 1 if F[x+u-1,y+v-1] is 1 **everywhere** that H[u,v] is 1
  0 otherwise
- Written $G = H \ominus F$

Erosion example



Demo: http://www.cs.bris.ac.uk/~majid/mengine/morph.html

Nested dilations and erosions

What does this operation do?

$$G = H \ominus (H \oplus F)$$

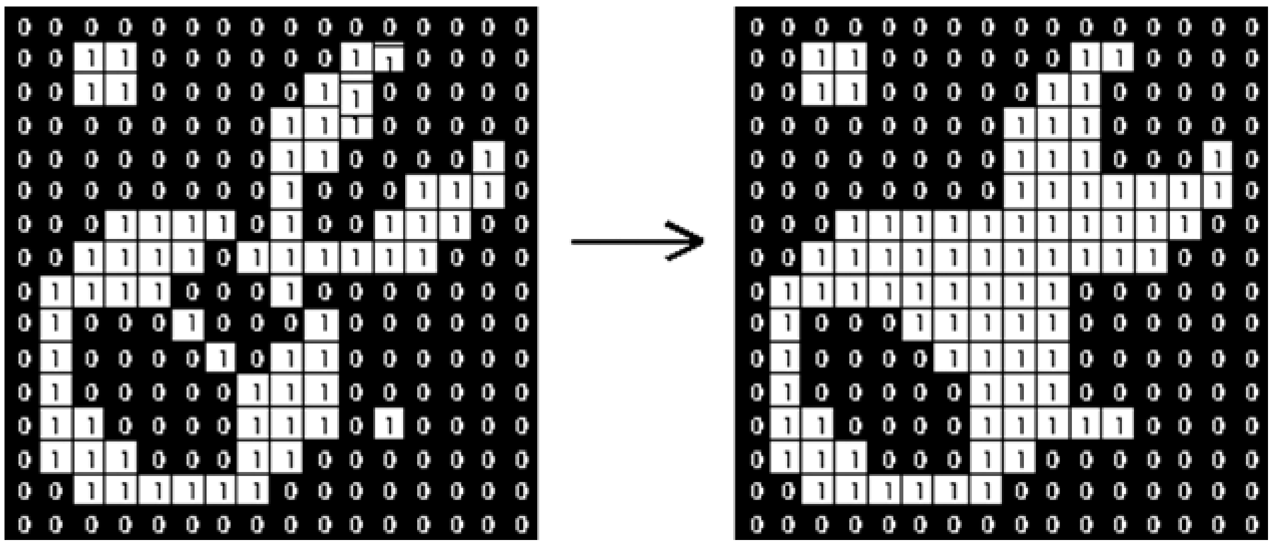
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0
0	0	0	1	1	1	1	0	1	0	0	1	1	1	0	0
0	0	1	1	1	1	0	1	1	1	1	1	1	0	0	0
0	1	1	1	1	0	0	0	1	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	1	0	1	1	0	0	0	0	0	0
0	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0
0	1	1	0	0	0	0	1	1	1	0	1	0	0	0	0
0	1	1	1	0	0	0	1	1	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- This is called a **closing** operation

Nested dilations and erosions

What does this operation do?

$$G = H \ominus (H \oplus F)$$



- This is called a **closing** operation

Is this the same thing as the following?

$$G = H \oplus (H \ominus F)$$

Nested dilations and erosions

What does this operation do?

$$G = H \oplus (H \ominus F)$$

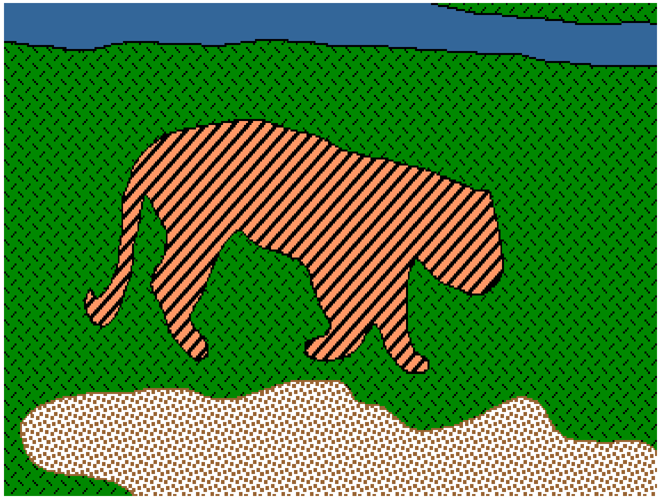
- This is called an **opening** operation
- http://www.dai.ed.ac.uk/HIPR2/open.htm

You can clean up binary images (e.g., results of segmentation) by applying combinations of dilations and erosions

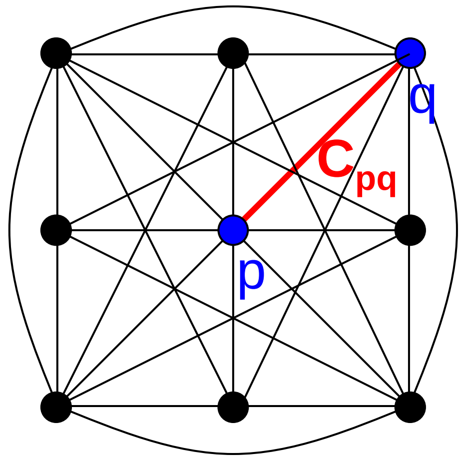
Dilations, erosions, opening, and closing operations are known as **morphological operations**

- see http://www.dai.ed.ac.uk/HIPR2/morops.htm

Graph-based segmentation?



Images as graphs



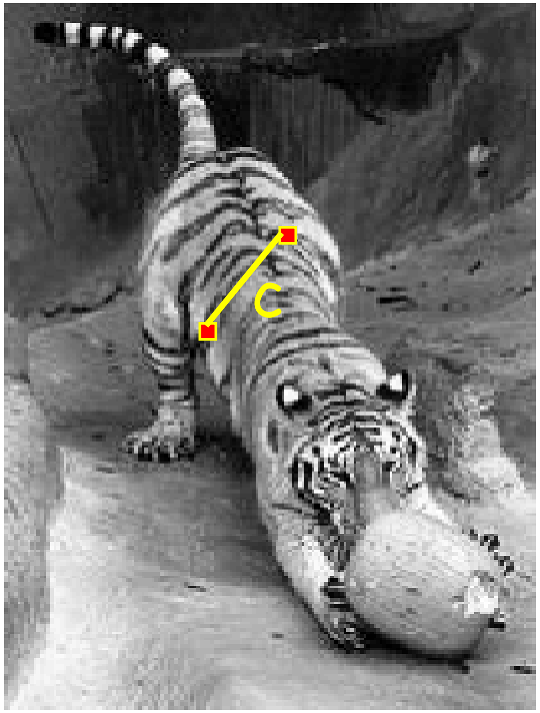
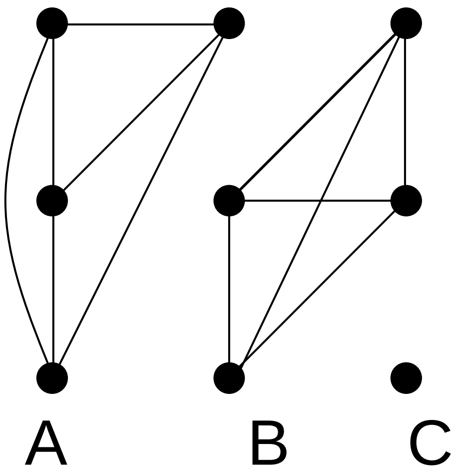


Image = *Fully-connected* graph

- node for every pixel
- link between *every* pair of pixels, **p**,**q**
- cost c_{pq} for each link
  - c_{pq} measures *similarity*
    - » similarity is *inversely proportional* to difference in color and distance
    - » this is different than the costs for intelligent scissors

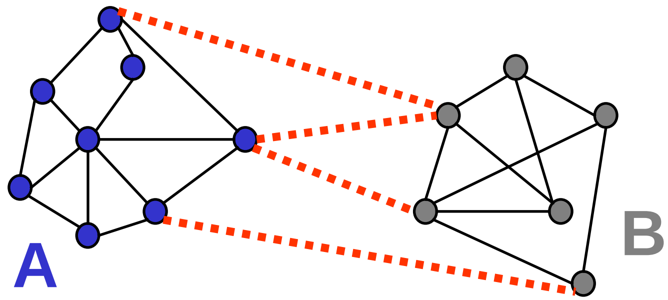
Segmentation by Graph Cuts



Goal: Break Graph into Segments

- Delete links that cross between segments
- Easiest to break links that have low cost (low similarity)
  - – similar pixels should be in the same segments
  - – dissimilar pixels should be in different segments

Cuts in a graph



Link Cut

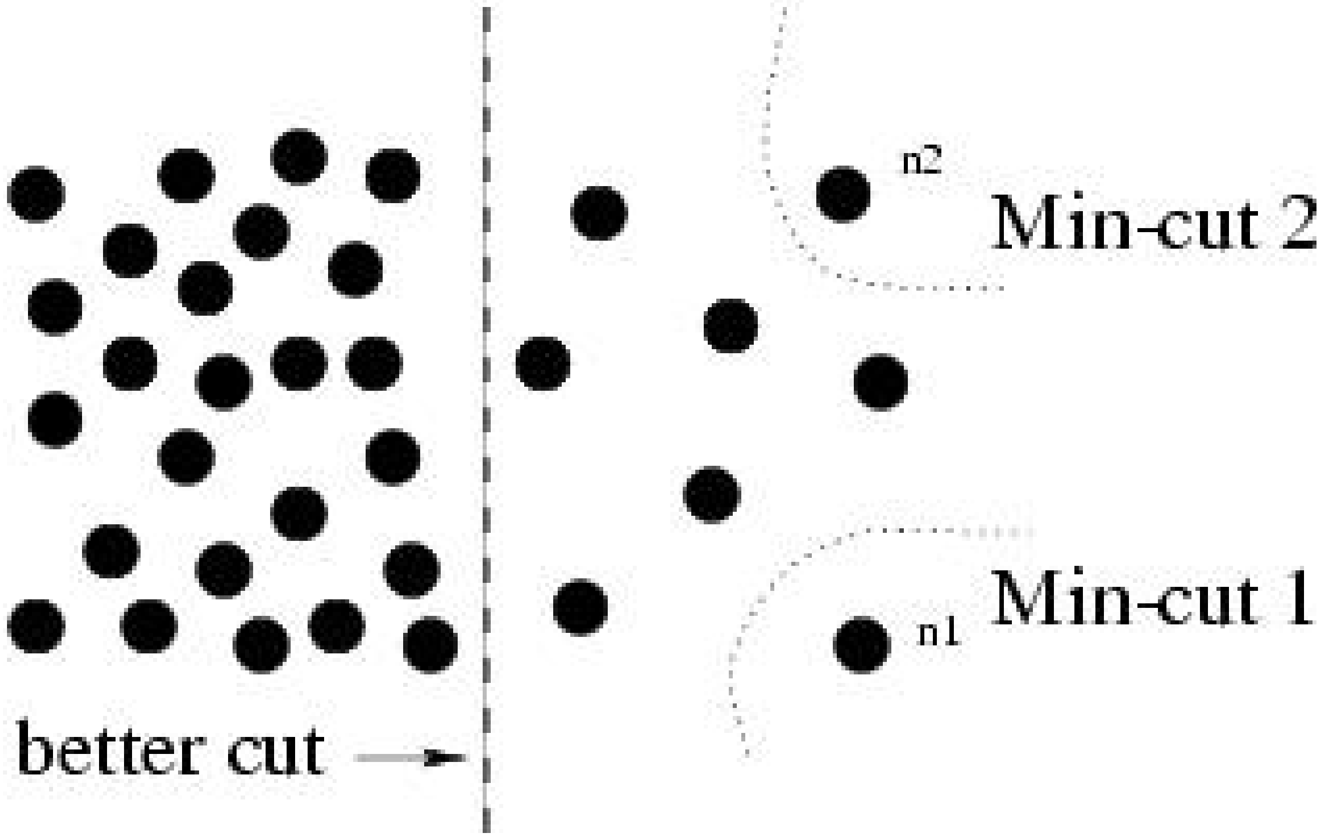
- set of links whose removal makes a graph disconnected
- cost of a cut:

$$cut(A,B) = \sum_{p \in A, q \in B} c_{p,q}$$

Find minimum cut

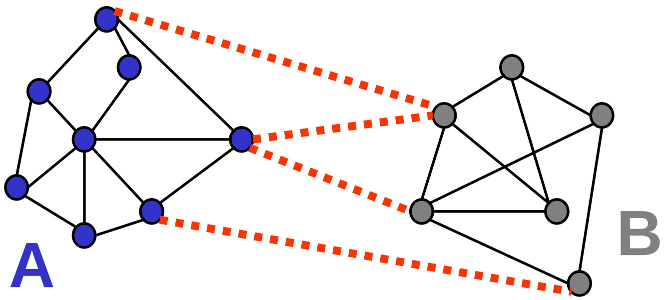
- gives you a segmentation
- fast algorithms exist for doing this

But min cut is not always the best cut...



Cut as defined penalizes large segments $cut(A,B) = \sum_{p \in A, q \in B} c_{p,q}$

Normalized cuts



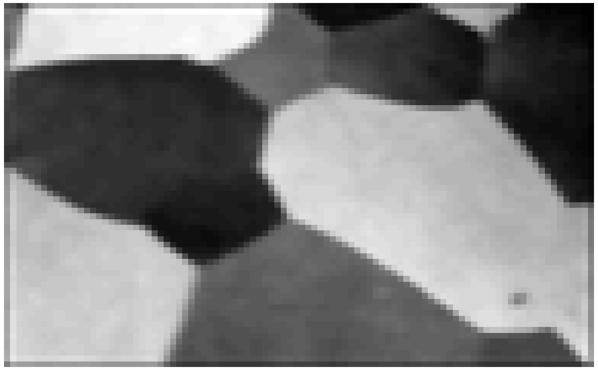
Normalized Cut

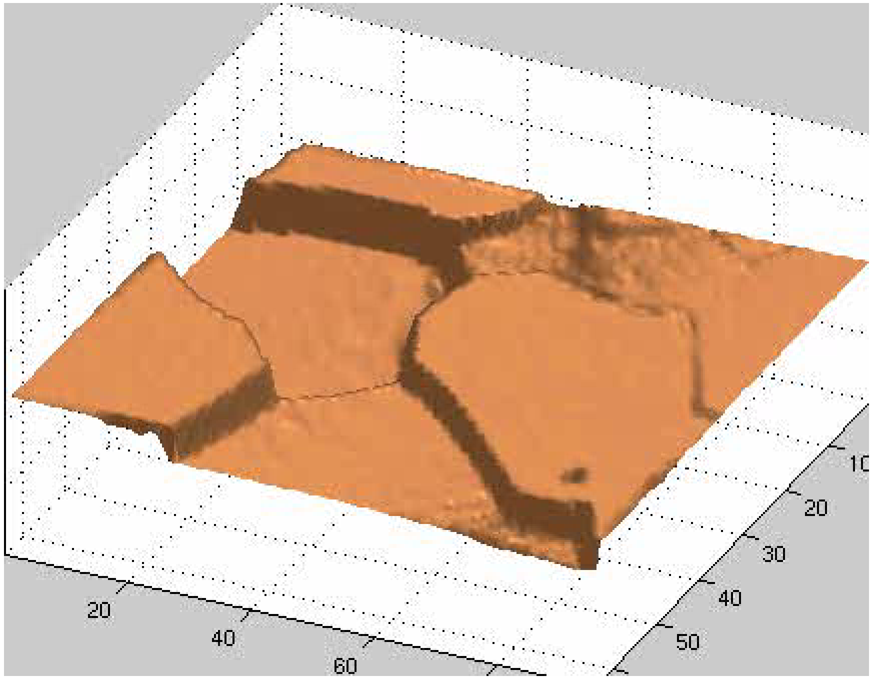
- A regular cut penalizes large segments $cut(A, B) = \sum_{p \in A, q \in B} c_{p,q}$
- fix by normalizing for size of segments

$$Ncut(A, B) = \frac{cut(A, B)}{volume(A)} + \frac{cut(A, B)}{volume(B)}$$

- volume(A) = sum of costs of all edges that touch A

Interpretation as a Dynamical System

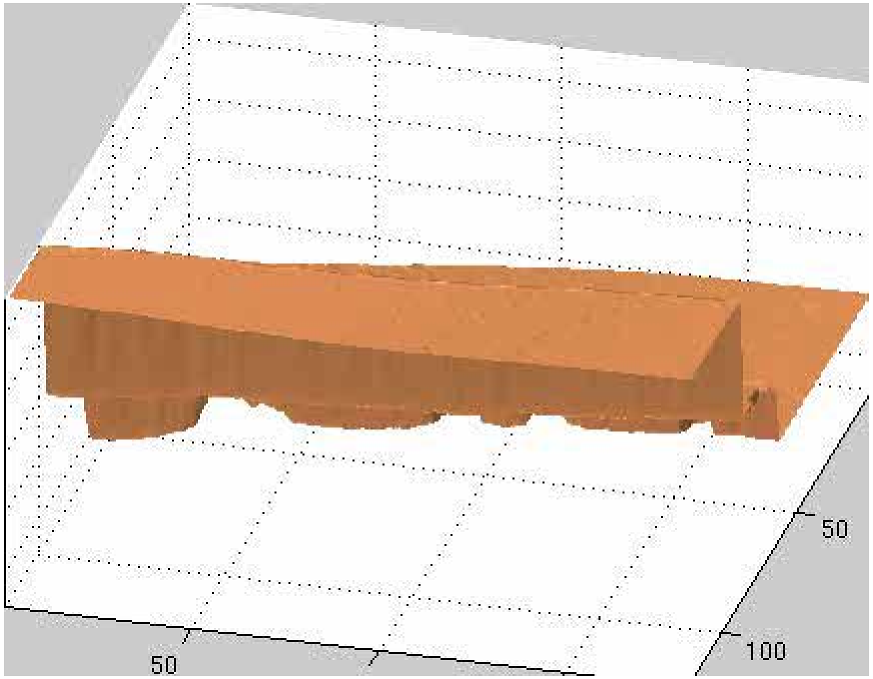




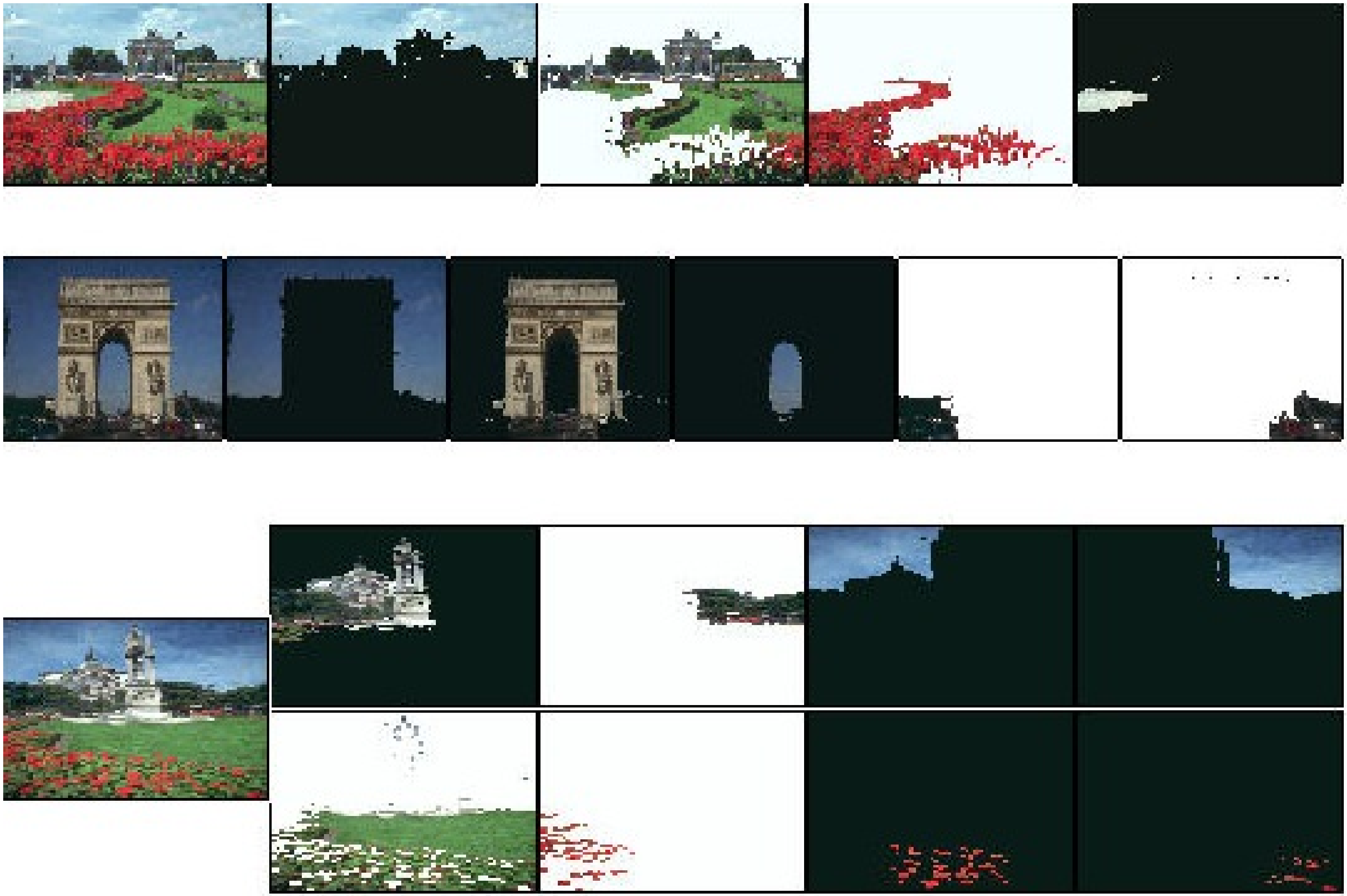
Treat the links as springs and shake the system

- elasticity proportional to cost
- vibration "modes" correspond to segments
  - can compute these by solving an eigenvector problem
  - for more details, see
    - » J. Shi and J. Malik, Normalized Cuts and Image Segmentation

Interpretation as a Dynamical System



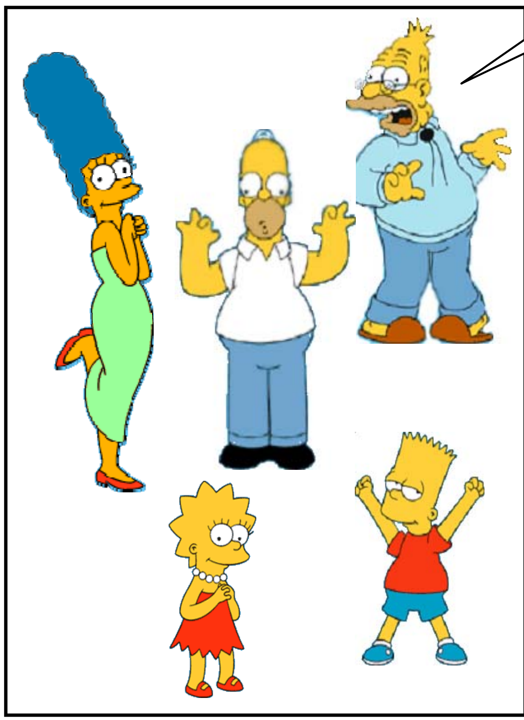
Color Image Segmentation using Normalized Cuts



Next Time: Guest lecture by Prof. Linda Shapiro

Things to do:

- Work on Project 3
- Read Chap. 8



Cluster 1

Cluster 2