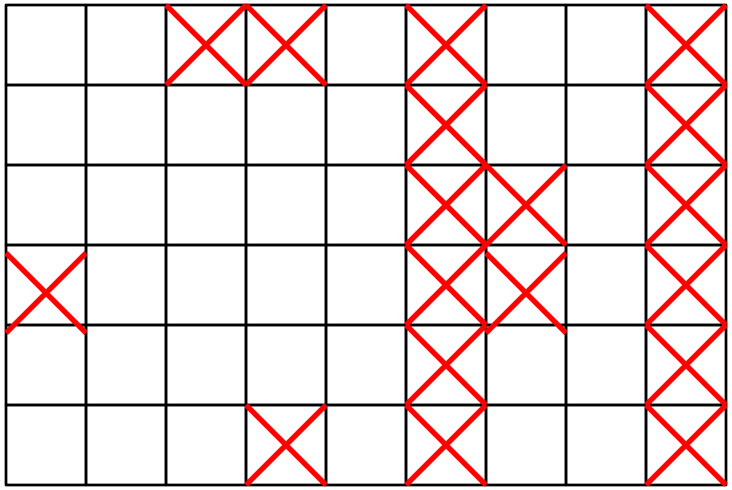
Seam Carving

- Project 1a due at midnight tonight.
- Project 1b goes out today.
  - We'll cover seam carving today.

Image resizing

Seam carving: idea

Cropping removes pixels from the image boundary.

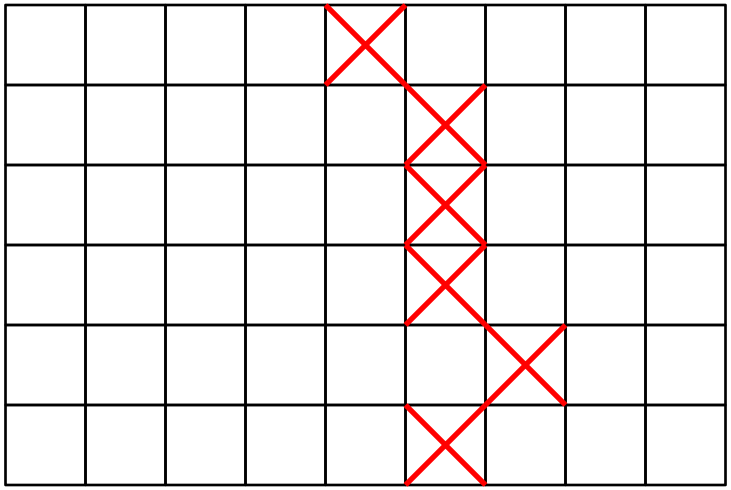


We want to remove only “unimportant” pixels.

How can we make sure the result is rectangular?

Seam carving: idea

Something in between: remove *seams*



Why not remove least important pixel from each row?

http://swieskowski.net/carve/

Pixel importance

- Many possible measures of importance
- One simple one is gradient magnitude

$$E[x,y] = \sqrt{I_x^2 + I_y^2}$$

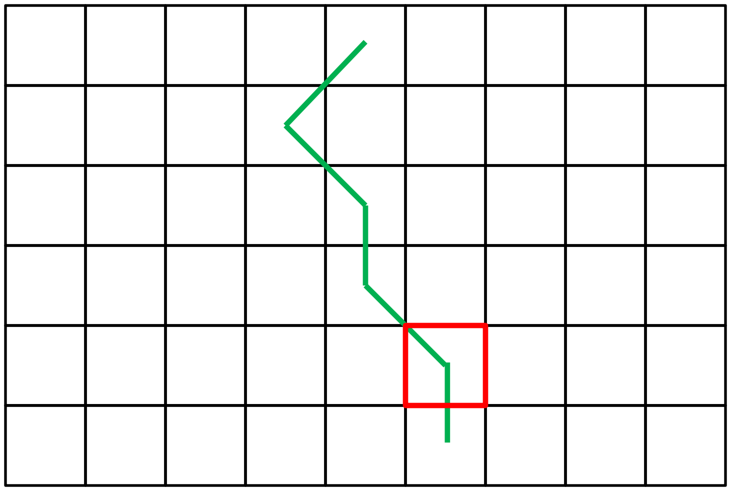
or $$E[x,y] = |I_x| + |I_y|$$

$$E(S) = \sum_{(x,y)\in S} E[x,y]$$

Computing the optimal seam

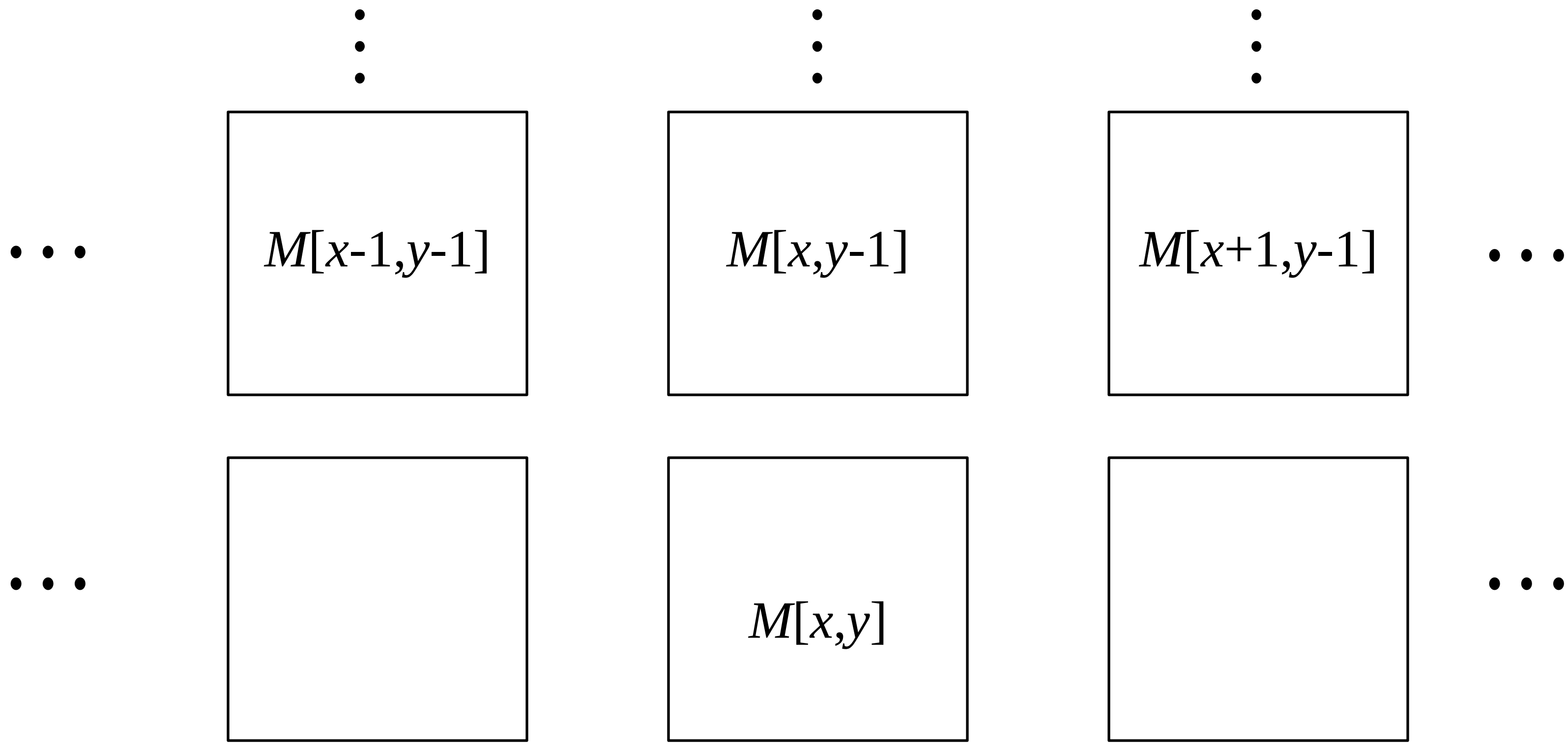
- How many vertical seams in an m-by-n image?

- We can avoid trying all of them.
  - Suppose this is the optimal seam:



  - What can we say about this one?

Computing the optimal seam

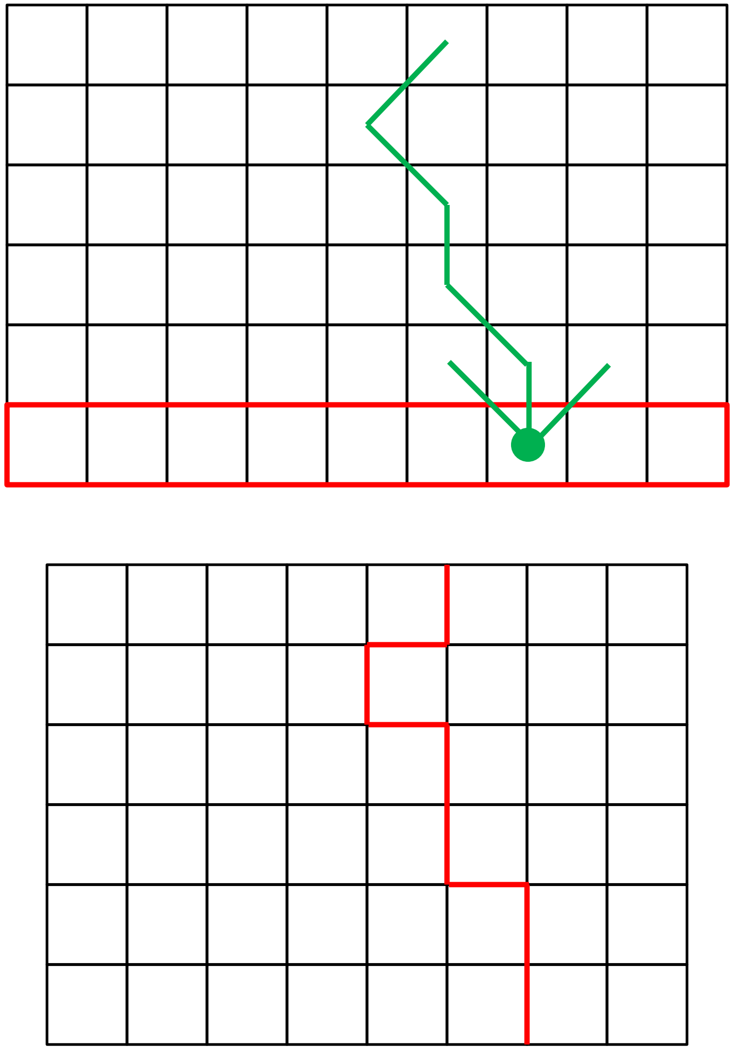


$\boldsymbol{M[x,y]}$ = cost of minimum-energy vertical seam through rows 1 to y

$$\boldsymbol{M[x,y]} = \min(M[x-1,y-1], M[x,y-1], M[x+1,y-1]) + E[x,y]$$

Computing the optimal seam

$M =$



1. Find pixel in bottom row with minimum seam cost.
2. Trace back optimal seam through image.
3. Remove seam pixels.

Seam carving algorithm

1. Compute energy at each pixel
2. While image larger than m-by-n:
   - Remove horizontal or vertical seam with minimum energy.

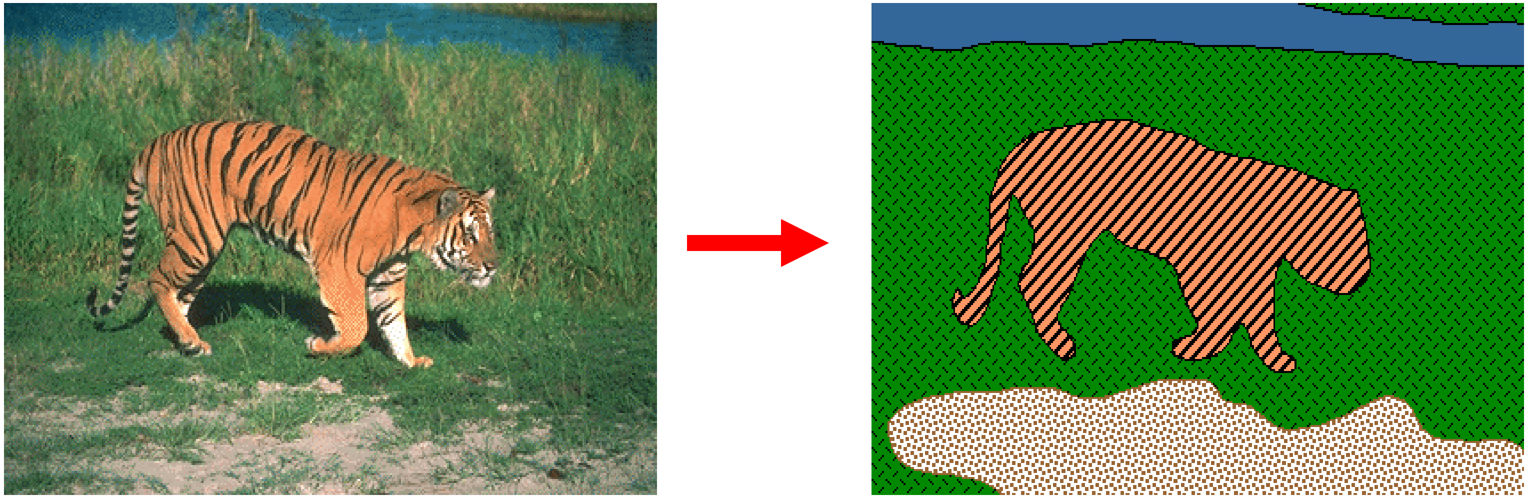
How do we choose between horizontal and vertical?

How might we enlarge an image?

Image Segmentation

From Sandlot Science

From images to objects



What Defines an Object?

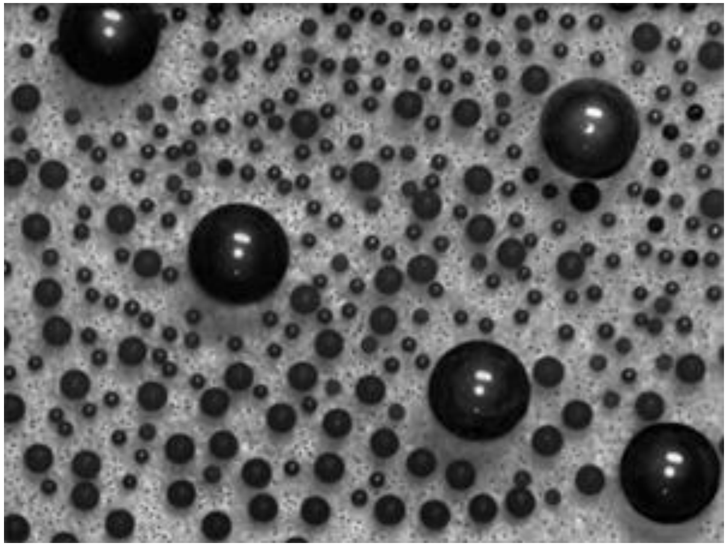
- Subjective problem, but has been well-studied
- Gestalt Laws seek to formalize this
  - proximity, similarity, continuation, closure, common fate
  - see notes by Steve Joordens, U. Toronto

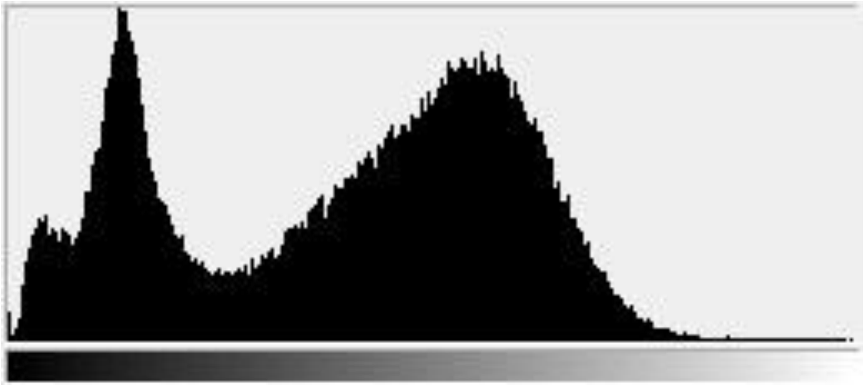
Image histograms

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 $F[x, y] \to RGB$
  - The histogram is $H_F[c] = |\{(x, y) \mid F[x, y] = c\}|$
    - » i.e., for each color value c (x-axis), plot # of pixels with that color (y-axis)

What do histograms look like?





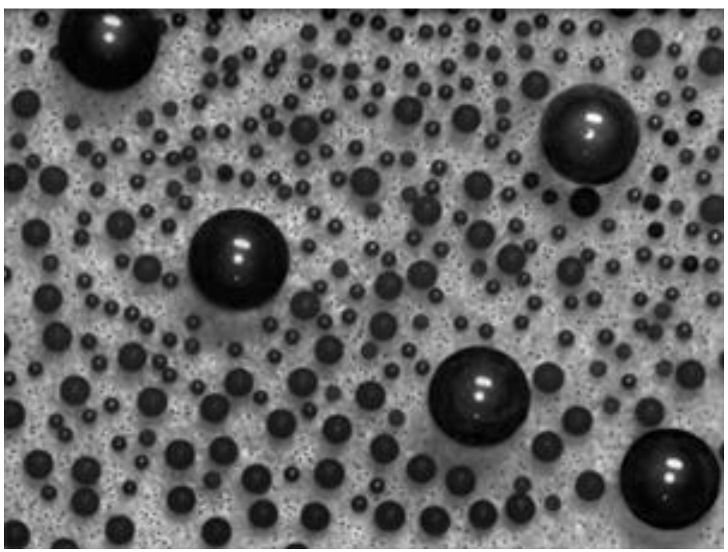
How Many Modes Are There?

- Easy to see, hard to compute

Histogram-based segmentation

Goal

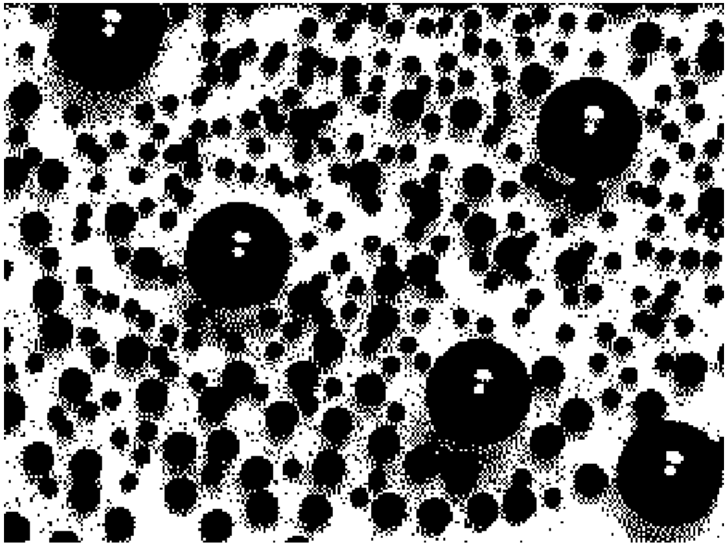
- Break the image into K regions (segments)
- Solve this by reducing the number of colors to K and mapping each pixel to the closest color

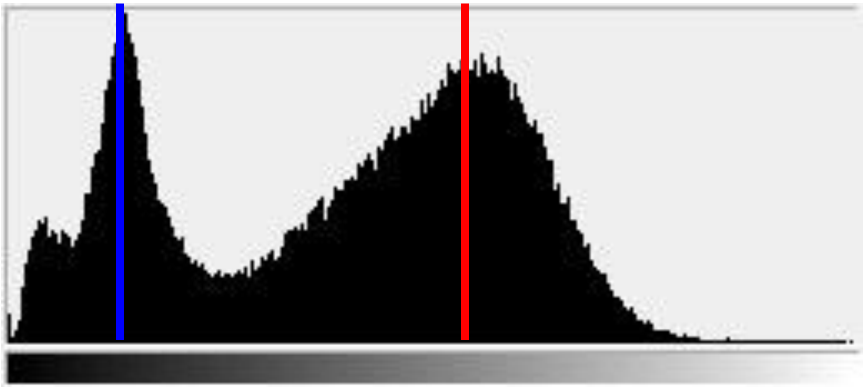


Histogram-based segmentation

Goal

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



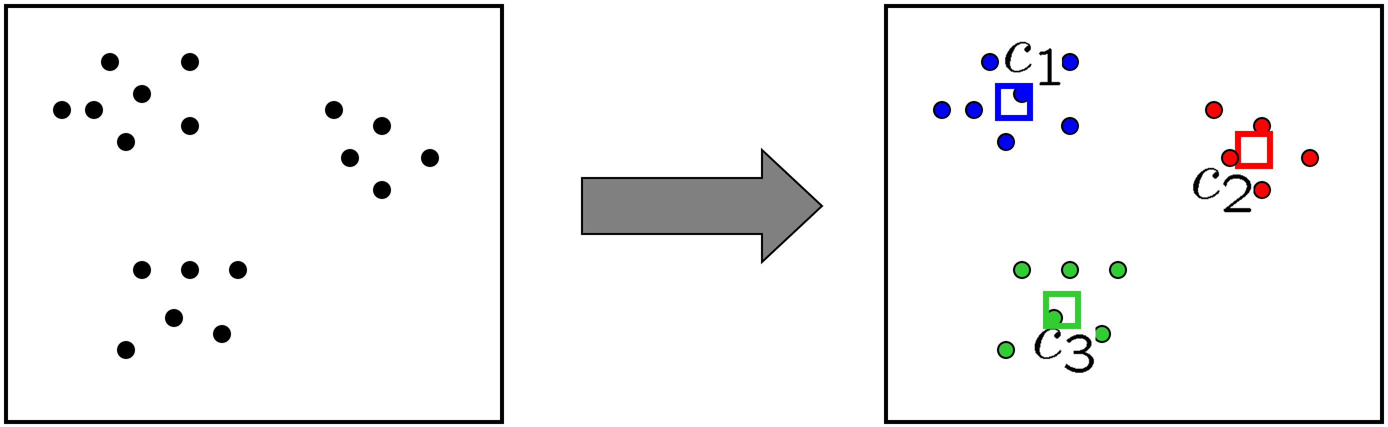


Here's what it looks like if we use two colors

Clustering

How to choose the representative colors?

- This is a clustering problem!



Objective

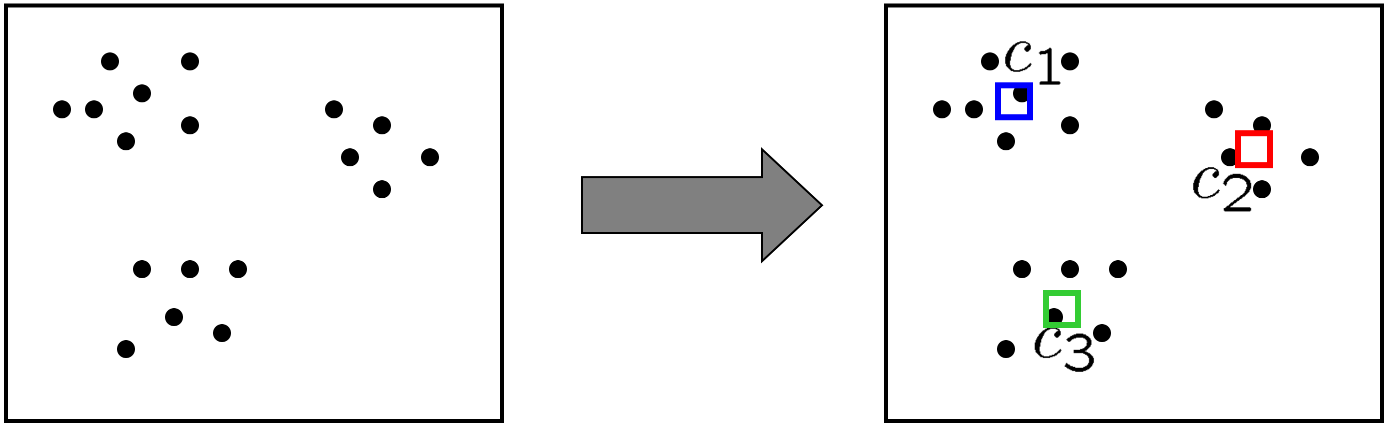
- Each point should be as close as possible to a cluster center
  - Minimize sum squared distance of each point to closest center

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

Break it down into subproblems

Suppose I tell you the cluster centers c_i

- Q: how to determine which points to associate with each c_i?
- A: for each point p, choose closest c_i



Suppose I tell you the points in each cluster

- Q: how to determine the cluster centers?
- A: choose c_i to be the mean of all points in the cluster

K-means clustering

K-means clustering algorithm

1. Randomly initialize the cluster centers, $c_1, ..., c_K$
2. Given cluster centers, determine points in each cluster
   - For each point p, find the closest c_i. Put p into cluster i
3. Given points in each cluster, solve for c_i
   - Set c_i to be the mean of points in cluster i
4. If c_i have changed, repeat Step 2

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

Properties

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

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

Probabilistic clustering

Basic questions

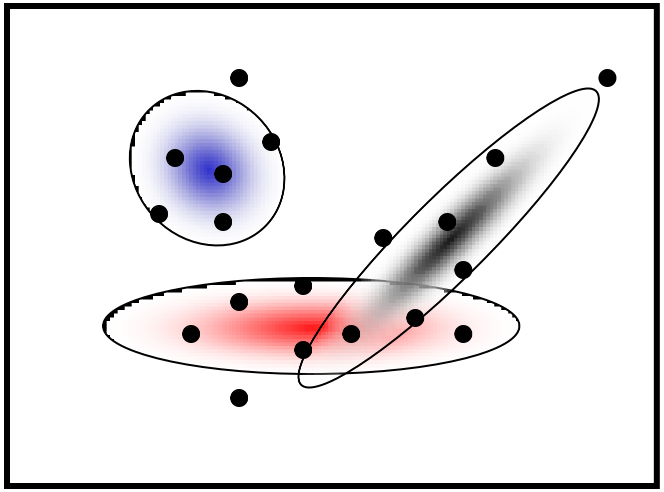
- what's the probability that a point $\mathbf{x}$ is in cluster m?
- what's the shape of each cluster?

K-means doesn't answer these questions

Basic idea

- instead of treating the data as a bunch of points, assume that they are all generated by sampling a continuous function
- This function is called a **generative model**
  - defined by a vector of parameters $\boldsymbol{\theta}$

Mixture of Gaussians



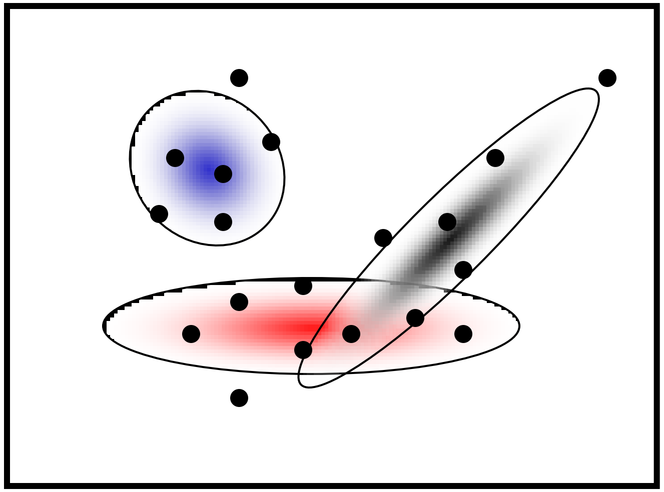
One generative model is a mixture of Gaussians (MOG)

- K Gaussian blobs with means $\boldsymbol{\mu}_\mathbf{b}$ covariance matrices $\mathbf{V}_\mathbf{b}$, dimension d
  - blob *b* defined by: $$P(x|\mu_b, V_b) = \frac{1}{\sqrt{(2\pi)^d |V_b|}} e^{-\frac{1}{2}(x-\mu_b)^T V_b^{-1}(x-\mu_b)}$$
- blob *b* is selected with probability α_b
- the likelihood of observing **x** is a weighted mixture of Gaussians

$$P(x|\theta) = \sum_{b=1}^{K} \alpha_b P(x|\theta_b)$$

- where $\theta = [\mu_1, \ldots, \mu_n, V_1, \ldots, V_n]$

Expectation maximization (EM)



Goal

- find blob parameters θ that maximize the likelihood function:

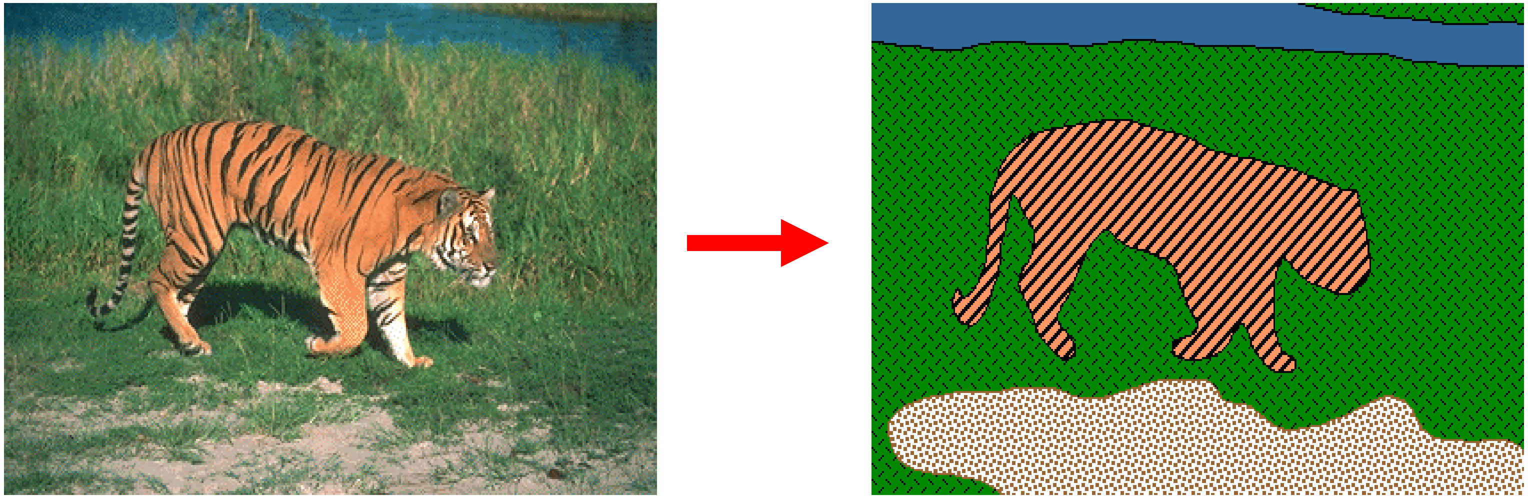
$$P(data|\theta) = \prod_x P(x|\theta)$$

Approach:

1. E step: given current guess of blobs, compute ownership of each point
2. M step: given ownership probabilities, update blobs to maximize likelihood function
3. repeat until convergence

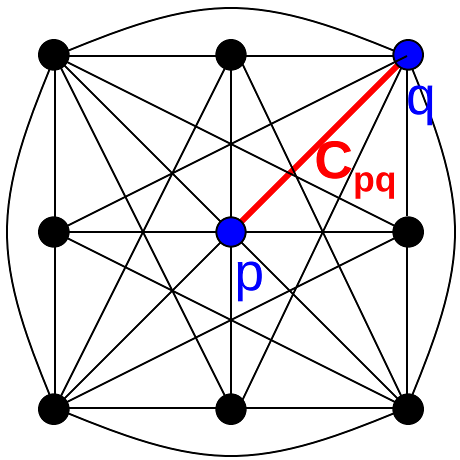
Grabcut [Rother et al., SIGGRAPH 2004]

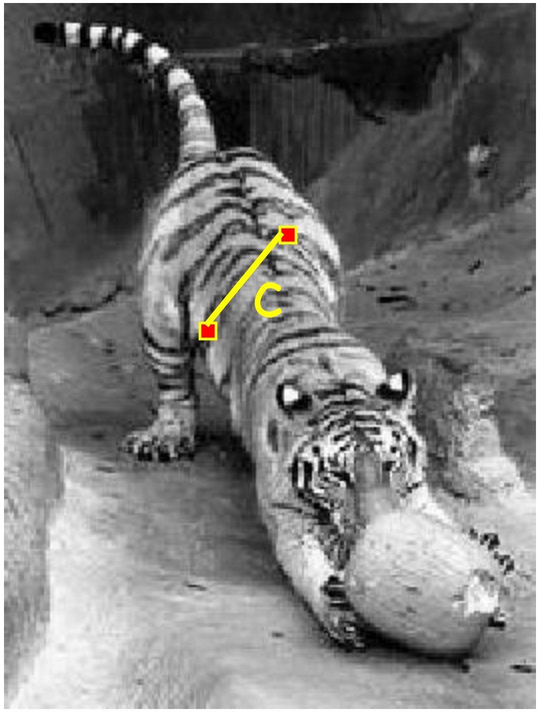
Graph-based segmentation?



What if we look at relationships between pixels?

Images as graphs

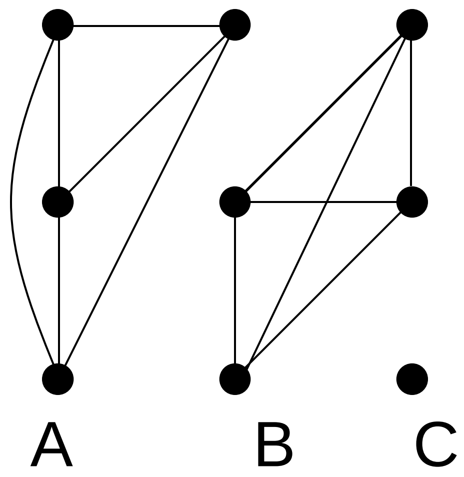




Fully-connected graph

- node for every pixel
- link between *every* pair of pixels, **p**,**q**
- cost $\mathbf{c_{pq}}$ for each link
  - $\mathbf{c_{pq}}$ measures *similarity*
    - » similarity is *inversely proportional* to difference in color and position

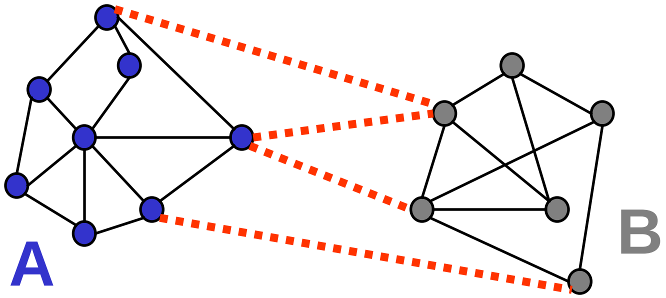
Segmentation by Graph Cuts



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

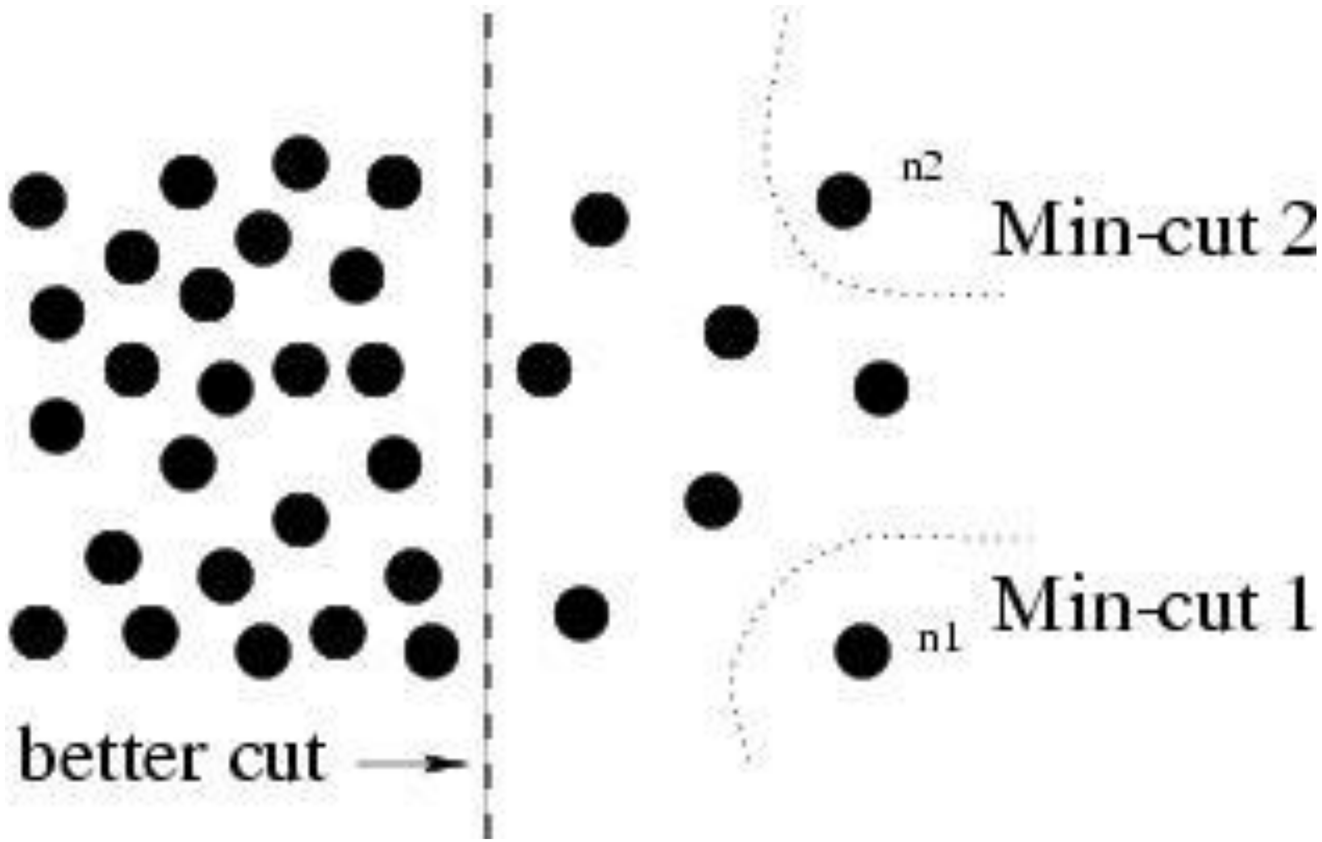
- set of links whose removal makes a graph disconnected
- cost of a cut:

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

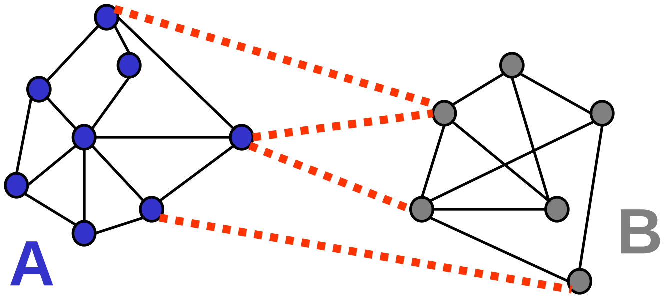
Find minimum cut

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

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



Cuts in a graph



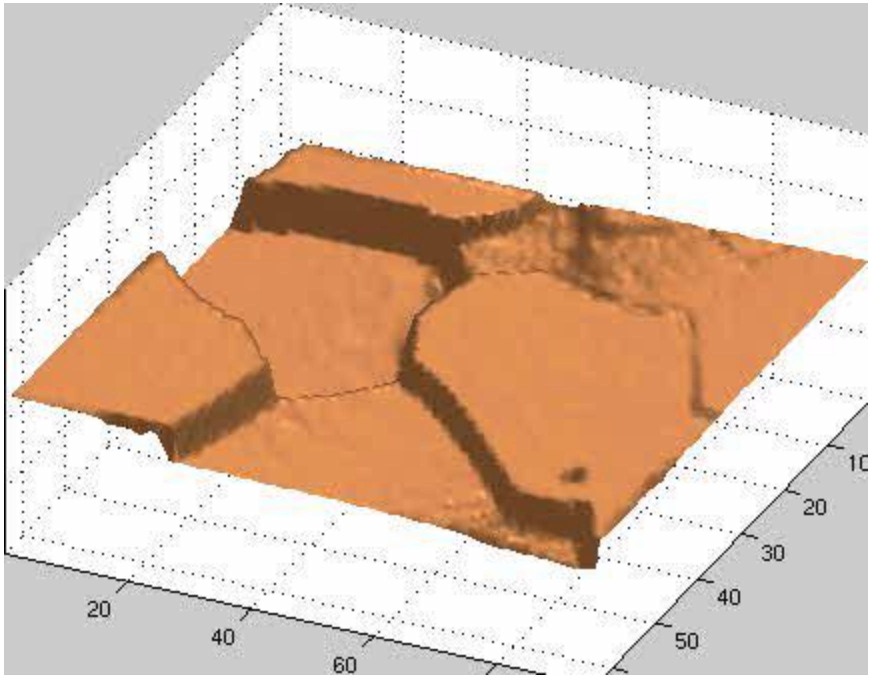
Normalized Cut

- a cut penalizes large segments
- 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

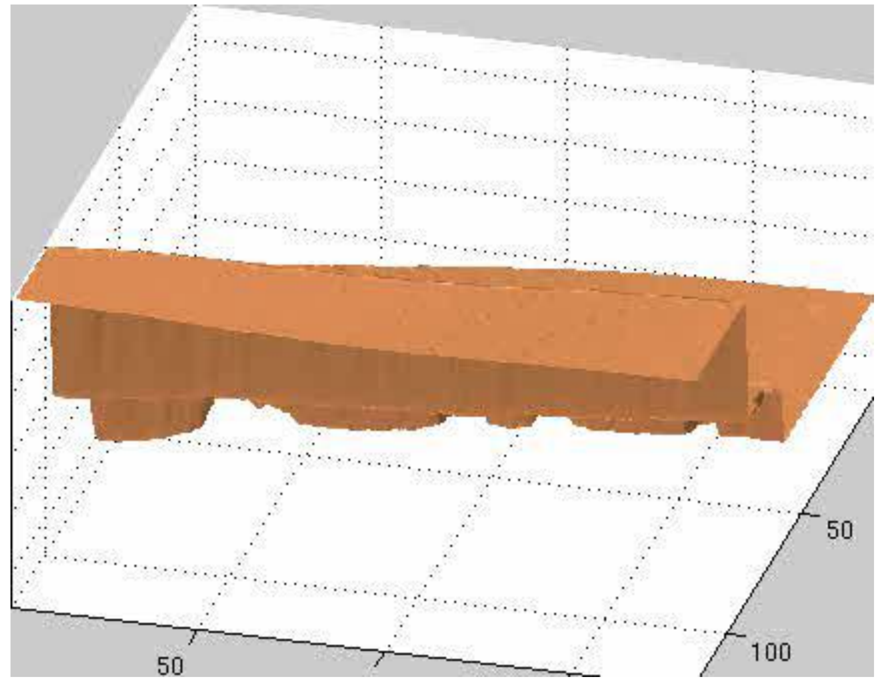
Interpretation as a Dynamical System



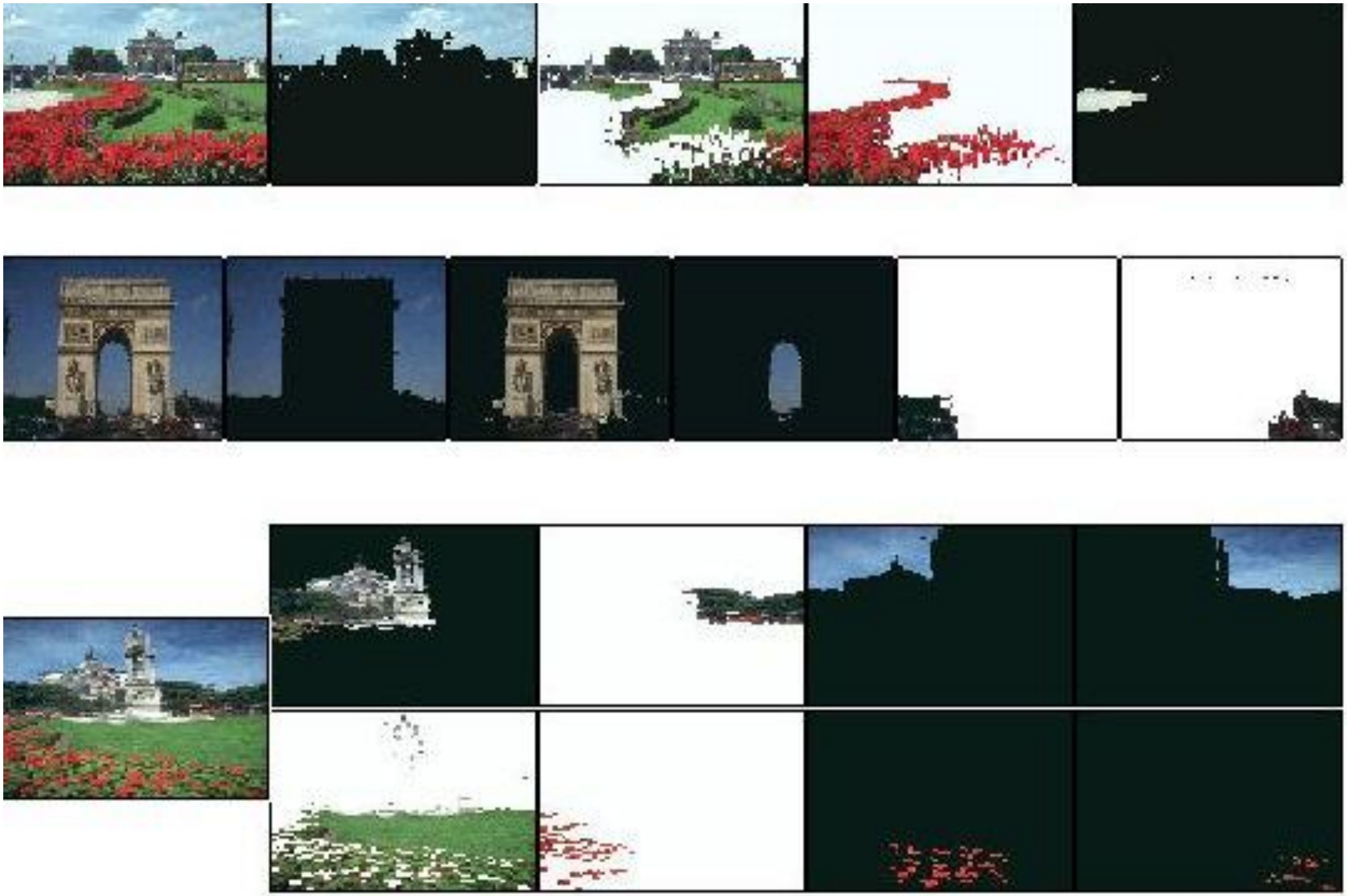
Treat the links as springs and shake the system

- 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, CVPR, 1997

Interpretation as a Dynamical System



Color Image Segmentation



Intelligent Scissors (demo)

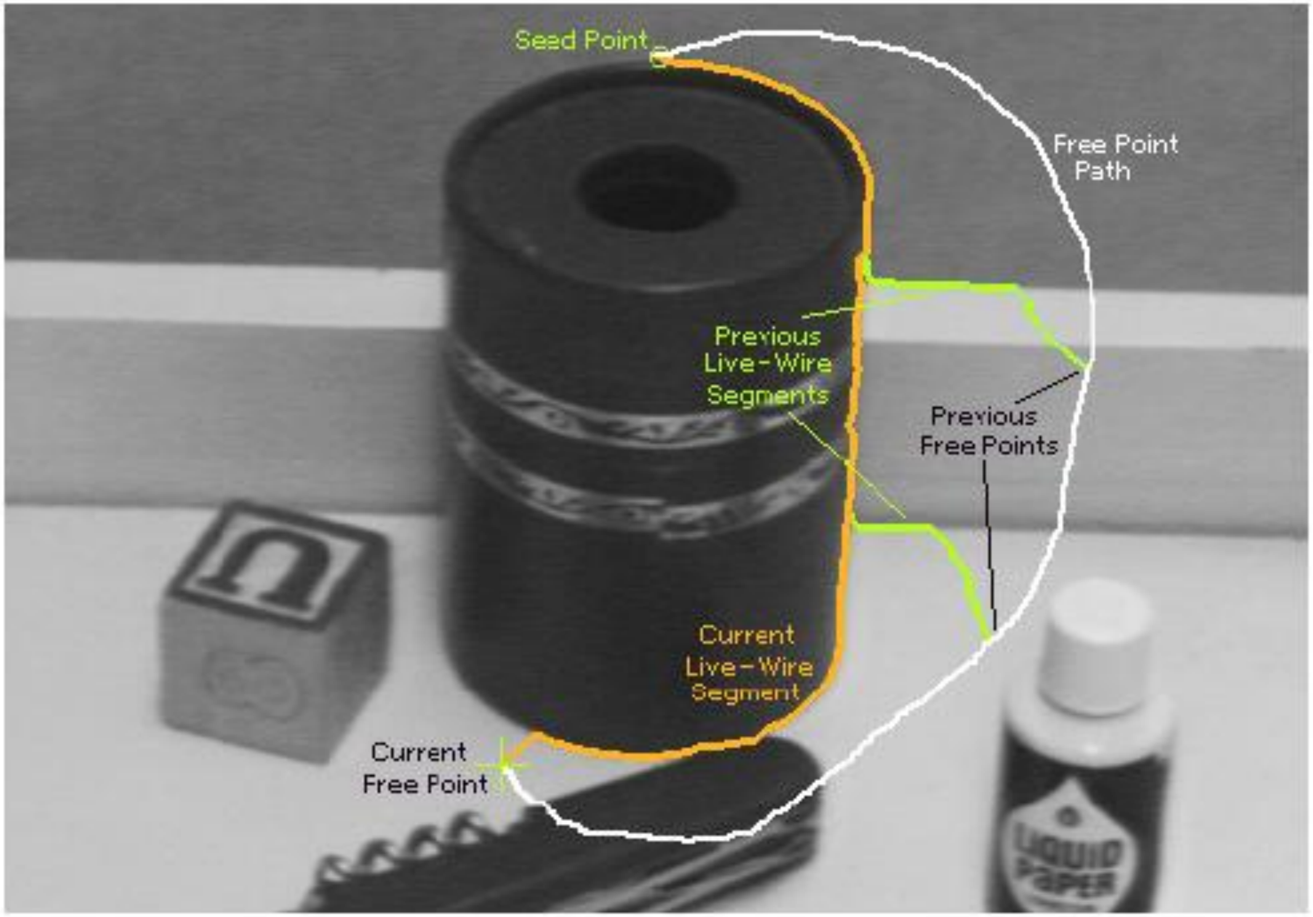


Figure 2: *Image demonstrating how the live-wire segment adapts and snaps to an object boundary as the free point moves (via cursor movement). The path of the free point is shown in white. Live-wire segments from previous free point positions (t_0, t_1, and t_2) are shown in green.*