Image Segmentation

CS 510

Lecture #14

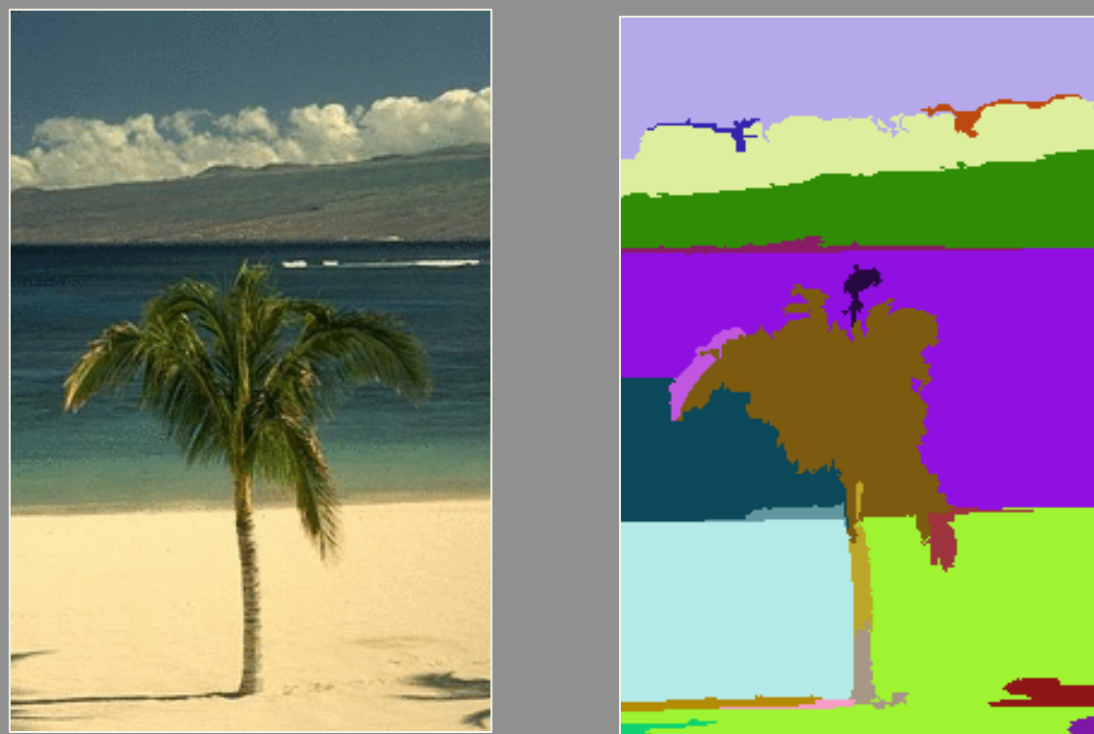
April 1, 2013

Colorado State University

Image Segmentation

- Goal:
  - Break the image into a set of non-overlapping tiles (a.k.a. regions)
  - Unachievable ideal #1: regions → objects
  - Unachievable ideal #2: regions → surfaces
- Available information
  - Pixel homogeneity (group similar pixels)
  - Edges as boundaries
  - (Top-down scenario): statistical surface models

Segmentation: Example



Source: http://people.cs.uchicago.edu/~pff/segment/

Background: Clustering

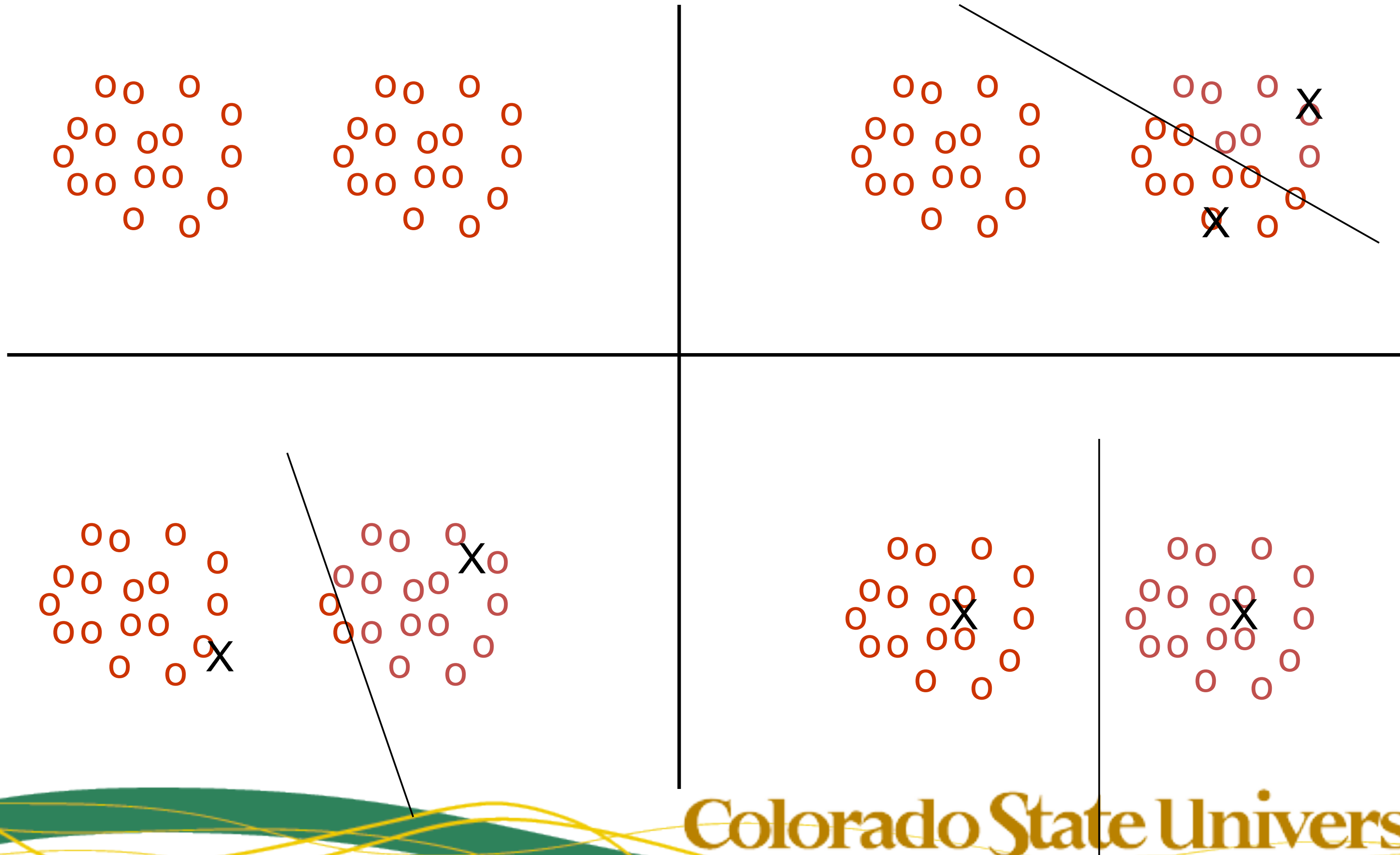
- Task: given a set of samples, collect them into groups of similar items
- Assumptions
  - K : the number of clusters
  - Every sample is a point in feature space
- Three main algorithms
  - K-Means
  - Expectation Maximization (EM)
  - Spectral clustering (not covered here)

Simple Clustering : K-Means

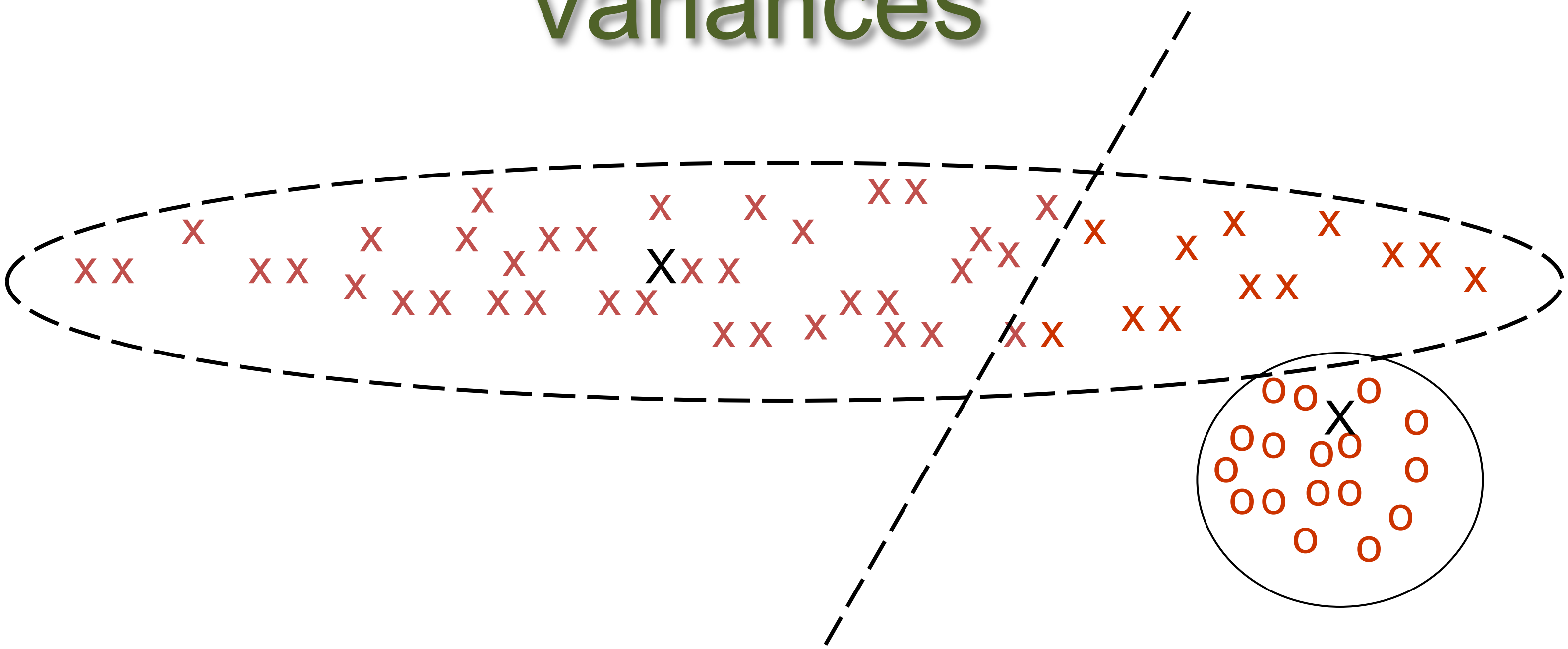
- Select K samples as random, make them cluster centers
  - There are useful variations on this step
- Iterate until no change:
  - Assign every sample to the nearest cluster center
  - Move every cluster center to the mean of the samples assigned to it

K-Means Illustration

K = 2



K-Means : Problem with Unequal Variances



- Implicit assumption: Gaussian processes with equal variance
- Two Gaussians, but different variances
- Need to model each cluster, not just its center

Measuring Cluster Variance

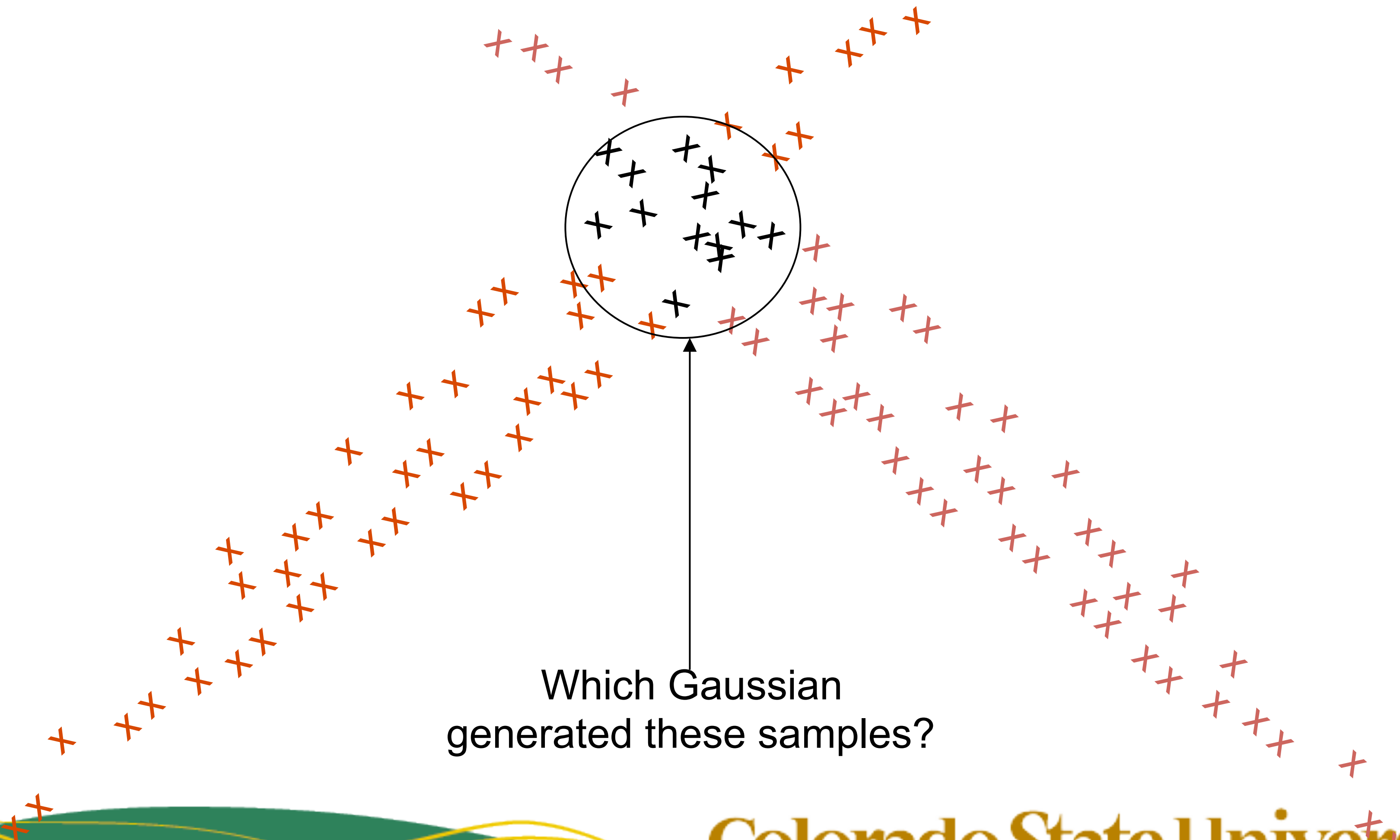
- Measure *covariance* Σ of PDFs:
  - Let X be the DxN set of mean-subtracted samples:

$$X = \begin{bmatrix} \vdots & \cdots & \vdots \\ s_1 & \cdots & s_n \\ \vdots & \cdots & \vdots \end{bmatrix}$$

  - Then Σ is the covariance matrix:

$$\Sigma = \frac{1}{N} X X^T$$

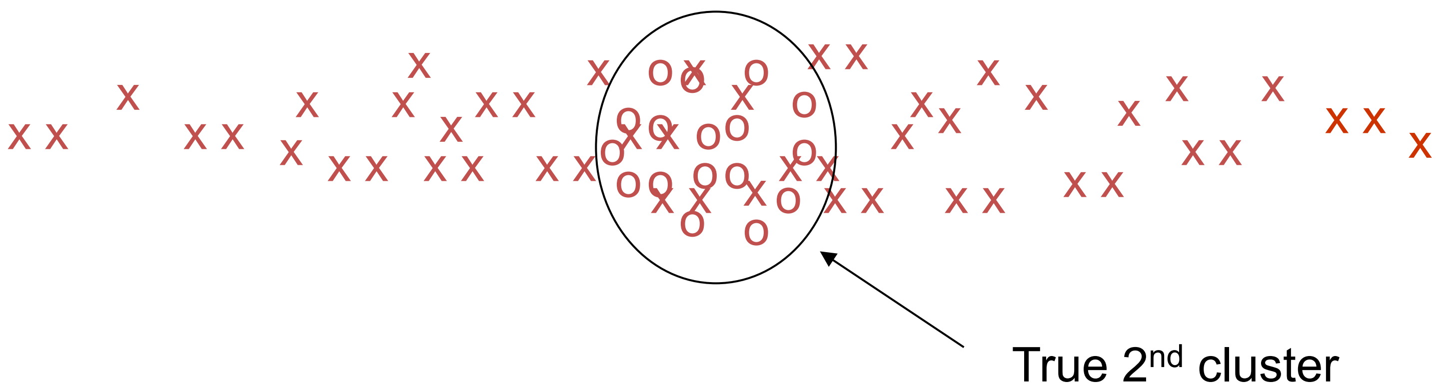
The Hard-Assignment Problem



Solution: soft assignments

- Which process generated the points in the middle?
  - Either could have
- For every sample/cluster pair, compute the likelihood that the sample was generated by the cluster
  - Note: the value is never zero
  - This is called "soft assignment"
  - Samples not uniquely assigned to clusters

Even Harder Overlapping Gaussians



Expectation Maximization (EM)

- Initialize clusters using random samples, uniform variance
- Iterate until minimal change
  - For every sample
    - Compute the likelihood that it could be generated by each cluster
    - Normalize likelihoods so that the sum is 1
      - The sample exists!
  - For every cluster
    - Compute mean and covariance matrix using probability-weighted samples
- If necessary, assign samples to most likely cluster

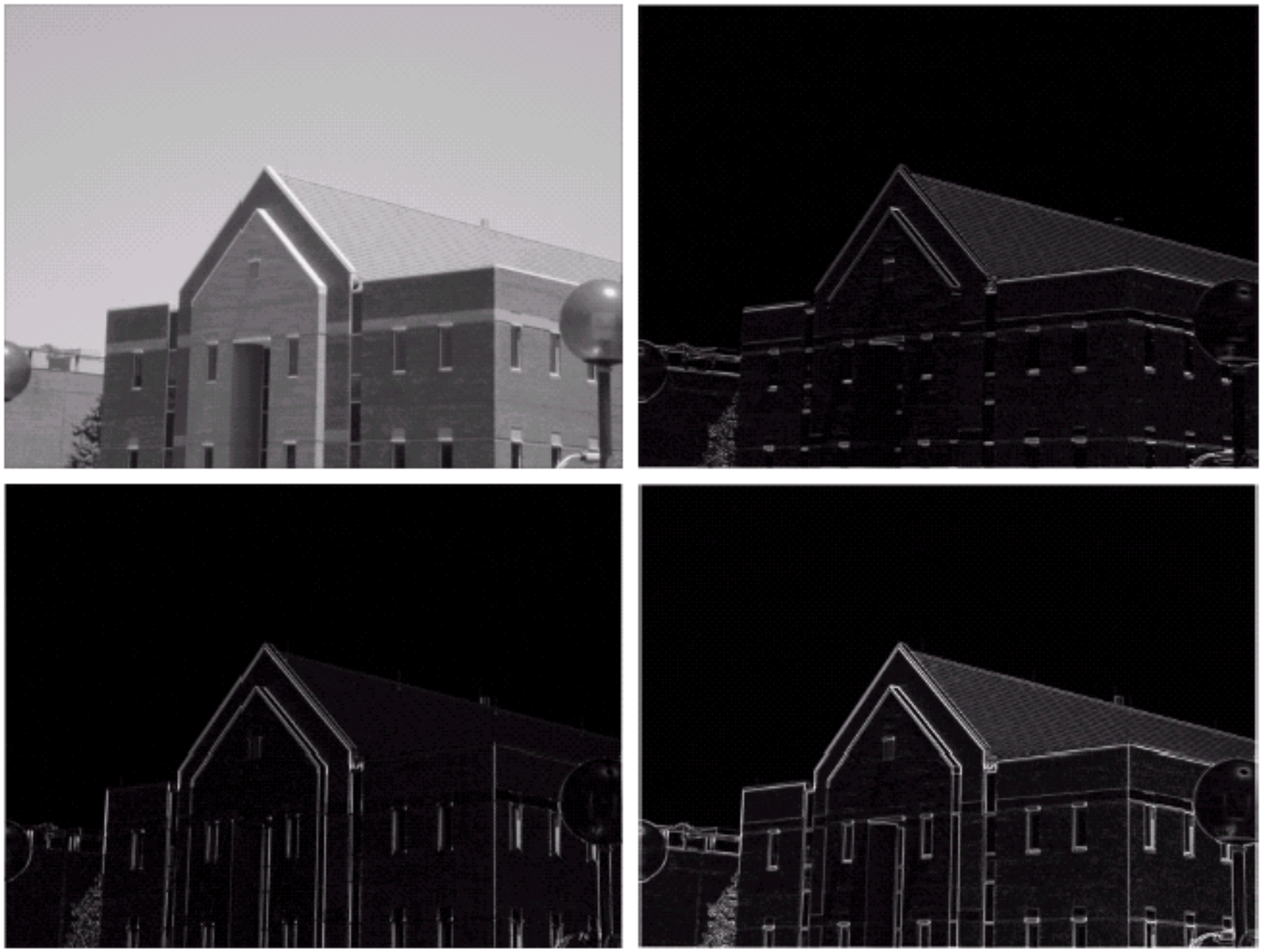
Bottom-up Segmentation

- Approach #1: Group similar pixels
- Example Algorithm: Comaniciu & Meer
  - Step 1: consider every pixel as a point in 5D
    - (r, g, b, x, y)
  - Step 2: cluster pixels into K categories
    - C&M use EM
  - Step 3: form connected components of pixels with the same labels
  - Step 4: Eliminate tiny regions
    - merge into most similar neighbors
- Many other algs have been proposed…

Bottom-up Segmentation

- Approach #2: Edge based
- Simplest method: zero-crossing
  - Compute the zero-crossing of the second derivative of the intensity surface
  - This will break the image up into “regions”
  - Merge adjacent regions that are “similar enough”

Example



a b
c d

FIGURE 10.11 Same sequence as in Fig. 10.10, but with the original image smoothed with a 5 × 5 averaging filter.

Source: www.spatial.maine.edu/~peggy/Teaching/SIE_434/Lecture17.ppt

Bottom-up Segmentation

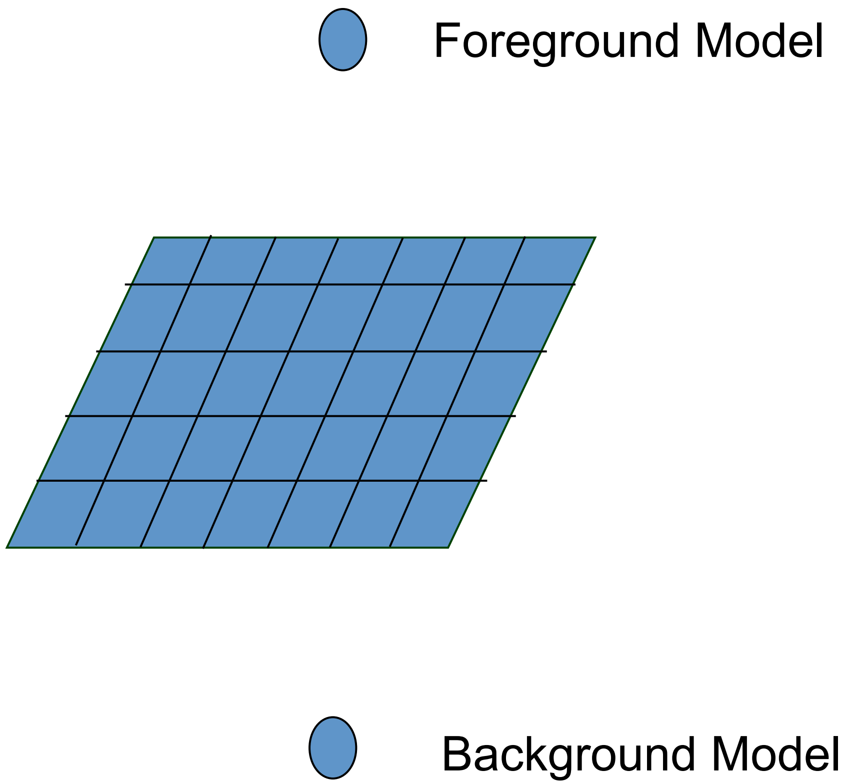
- You may note a certain lack of enthusiasm on my part
  - Segmentation is an important problem
  - Segmentation has been studied for 30 years
  - But, I don't think reliable bottom-up segmentation is possible (or necessary).

Top-down Segmentation

- If you know what you are looking for…
  – Foreground vs background (not tiling)
- Statistical/spatial foreground object model
  – P(x,y) = prob that pixel (x,y) is foreground
  – 1 – P(x,y) = prob that pixel (x,y) is background
- Then you can divide the image into
  – Likely to be foreground region, and
  – Other

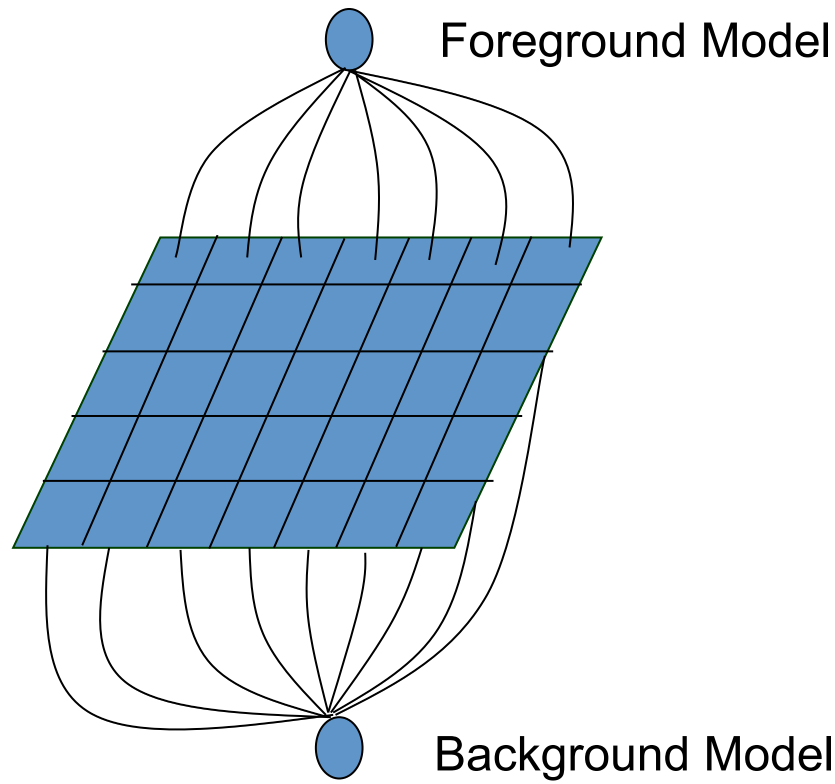
Graph Cut

Step 1: Create a graph in which foreground is a node, background is a node, and every pixel is a node



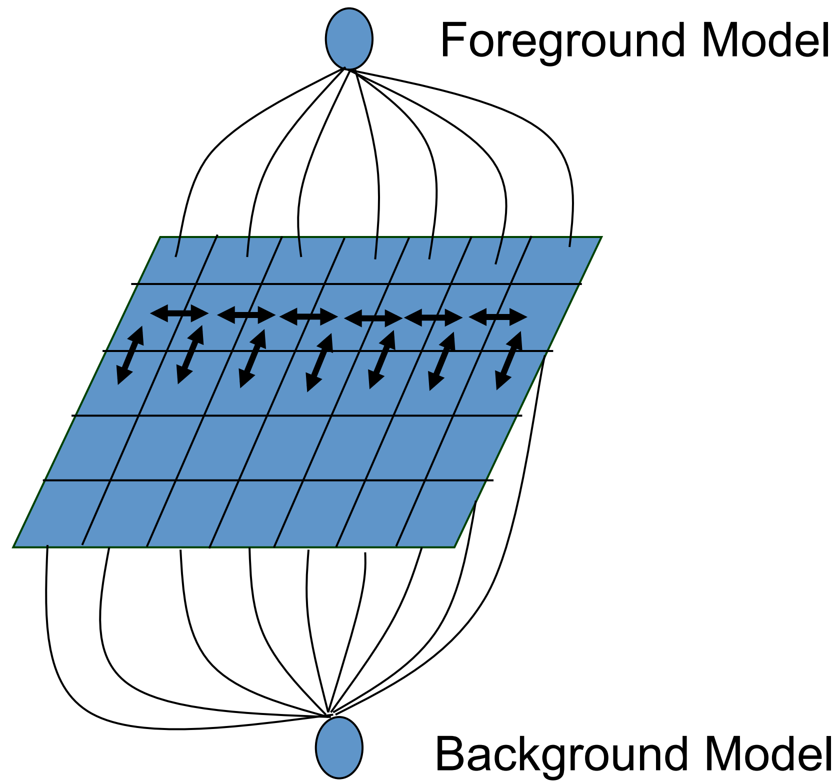
Graph Cut

Step 2: Connect every pixel (x,y) to the foreground node with an edge of strength p(x,y). Connect every pixel (x,y) to the background node with an edge of strength 1-p(x,y)



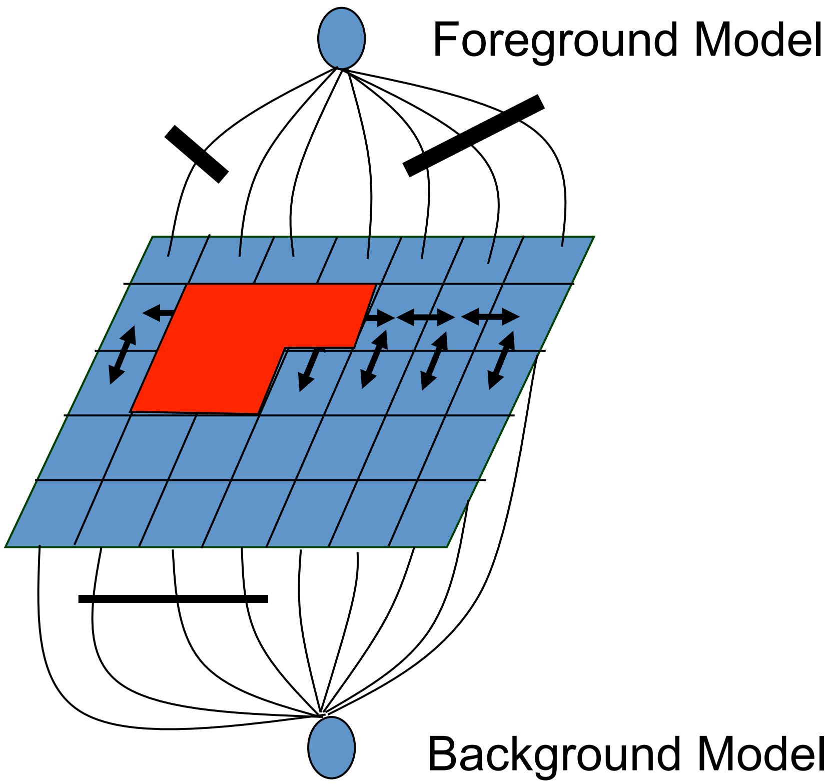
Graph Cut

Step 3: Connect every pixel to its four neighbors with an edge whose strength is inversely proportional to the image edge strength

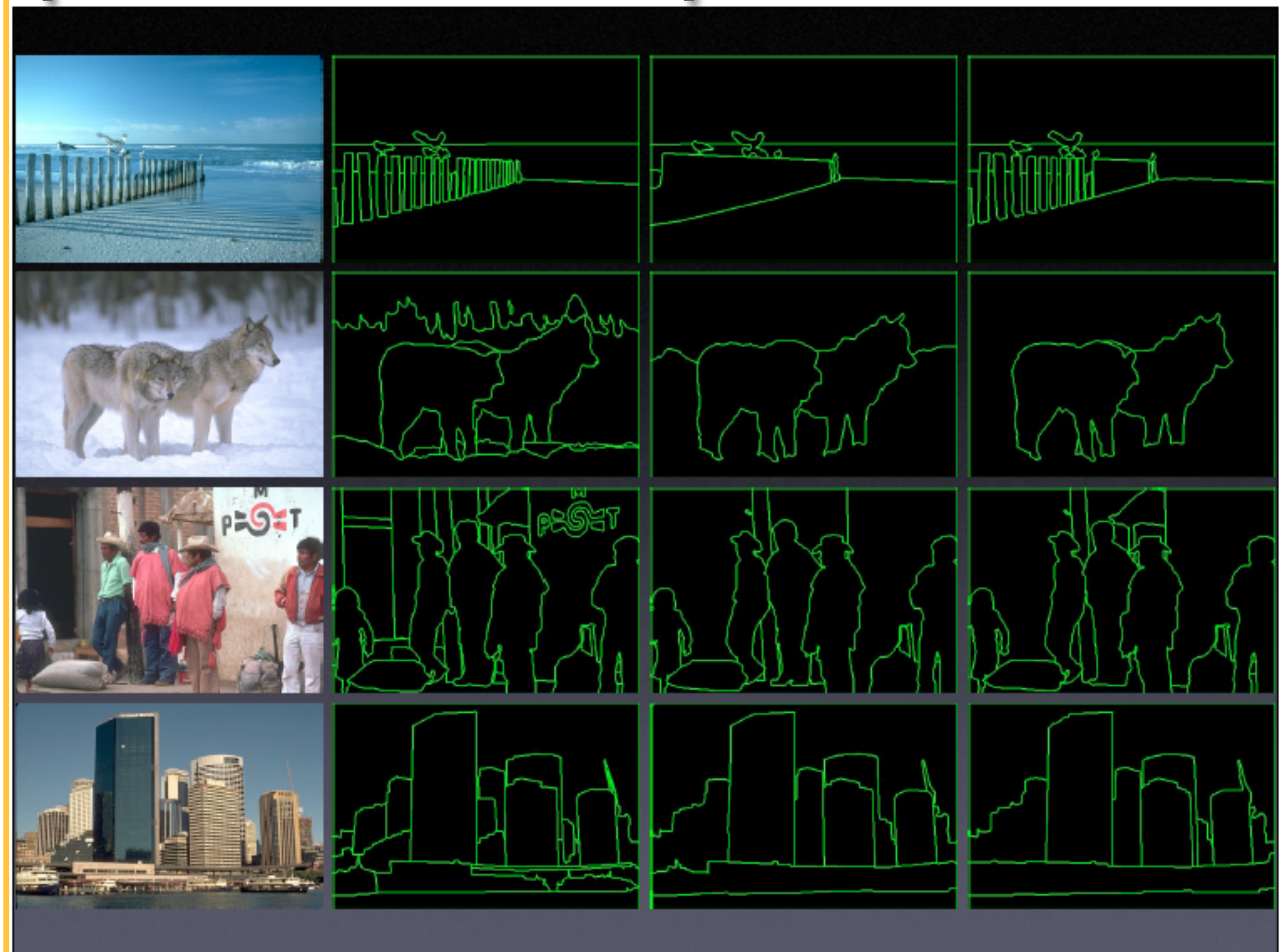


Graph Cut

Step 4: Find the max-flow partition of this graph. Pixels that remain connected to Foreground are foreground, the rest are background.



Graph Cut Examples

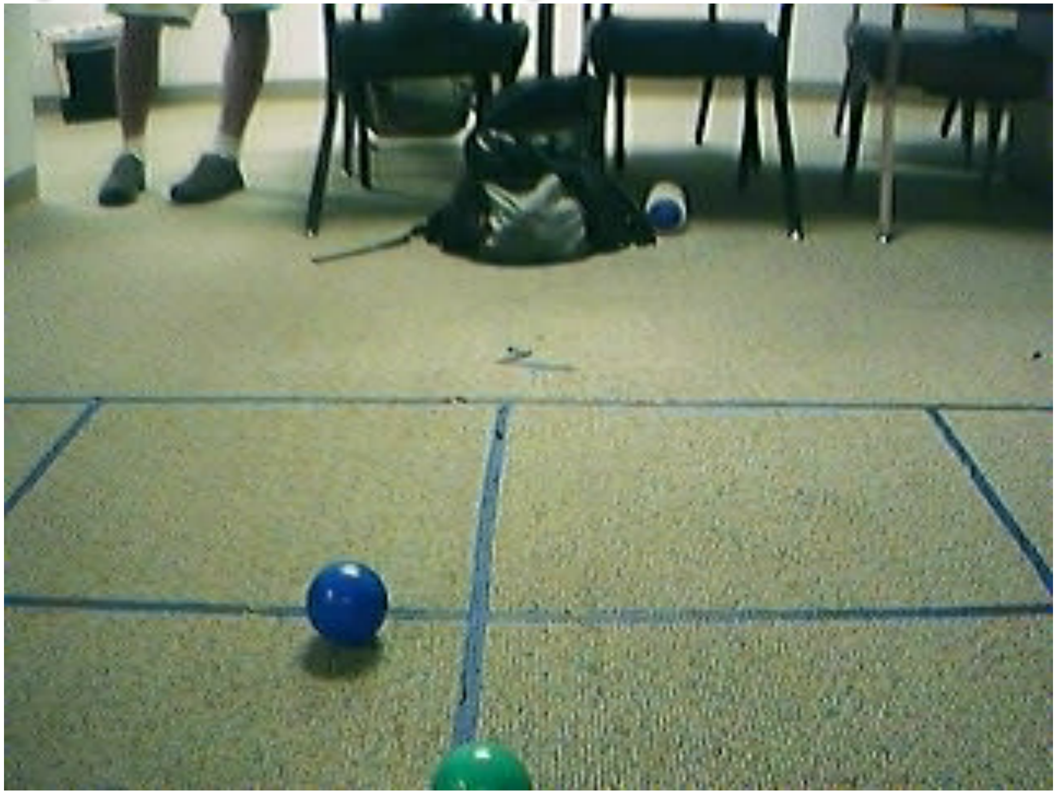


Source: http://www.cis.upenn.edu/~jshi/GraphTutorial/Tutorial-ImageSegmentationGraph-cut1-Shi.pdf

The Big Question

- We have edges, lines, regions & corners
  - And we'll pick up more features soon…
- How do we put them together?
  - Ah, research … no one really knows
  - Rigid, hierarchical schemes don't seem to work
  - Evidence combination works better

A Simple Example



Find the blue ball in this picture
(simple, right?)

Example (cont.)

- Approaches:
  - Hough for a circle
  - Find a connected circle of edges
  - Find bottom-up a blue region (roughly circular)
  - Learn ball properties, find top-down region

- Evidence combination : all of the above!