Computer Vision & Image Processing CSE 473 / 573

Instructor - Kevin R. Keane, PhD

TAs - Radhakrishna Dasari, Yuhao Du, Niyazi Sorkunlu

Lecture 24

October 25, 2017

Continued - Segmentation and clustering

Schedule

- Upcoming due dates
  - Project proposals 11/3/2017
  - HW3 11/6/2017
  - Project 12/15/2017
- Today
  - Segmentation, *continued*
- Readings
  - Today F&P chapter 9
  - Friday F&P chapter 10

Perceptual grouping

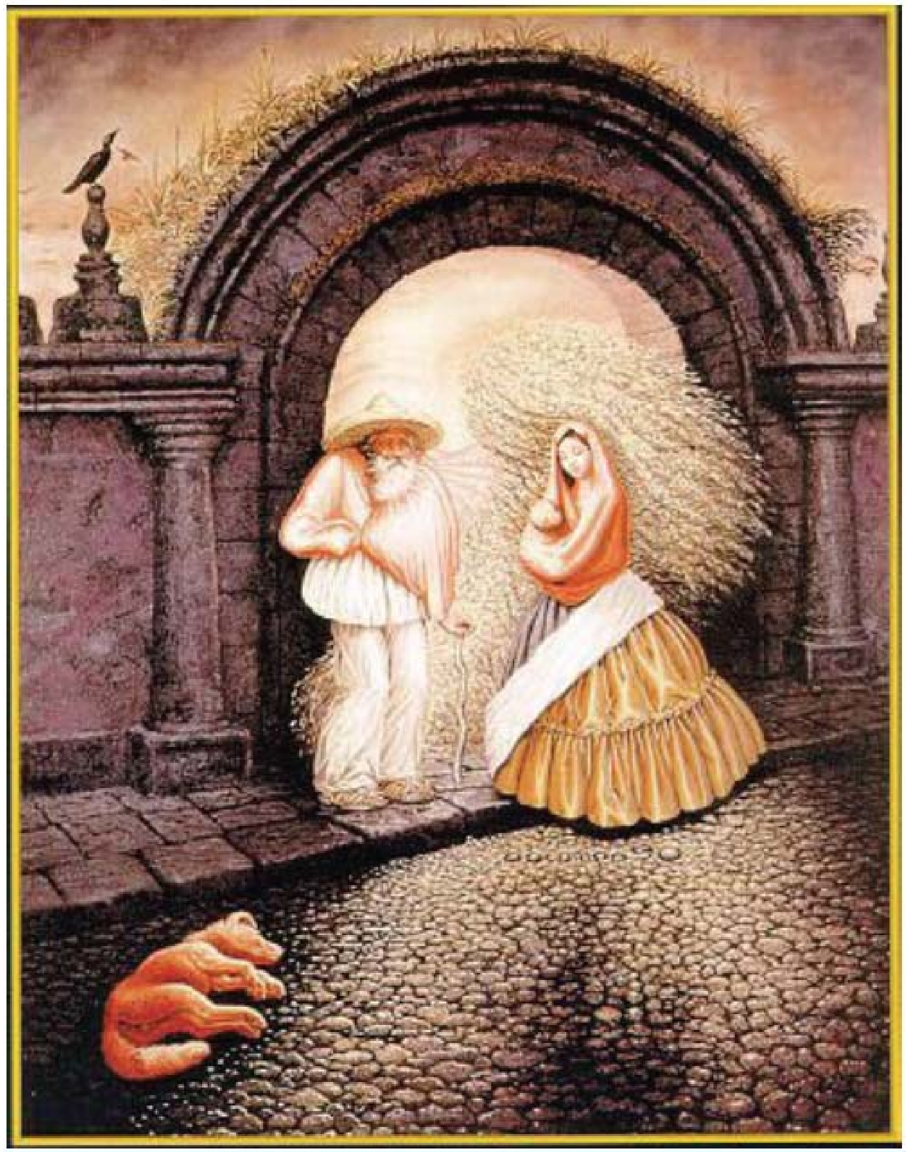
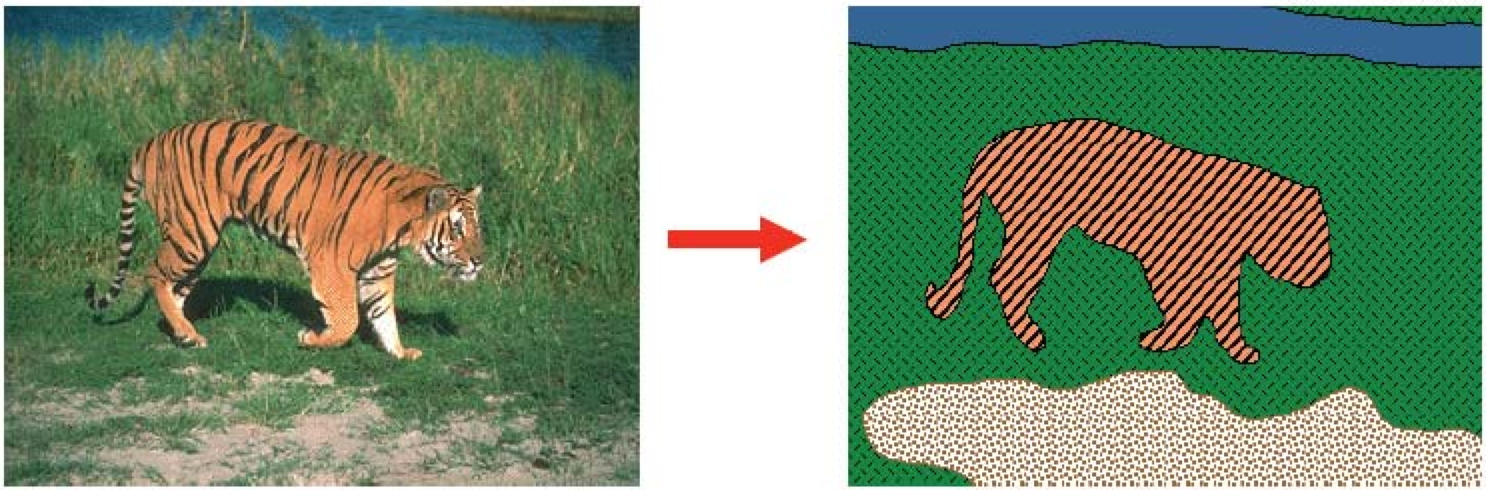


Image segmentation

The main goal is to identify groups of pixels/regions that "go together perceptually"



Examples of segmented images

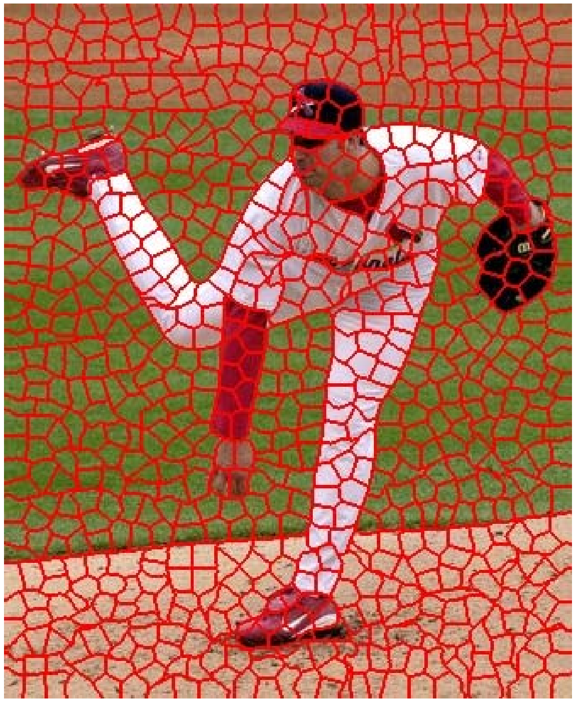
Why do segmentation?

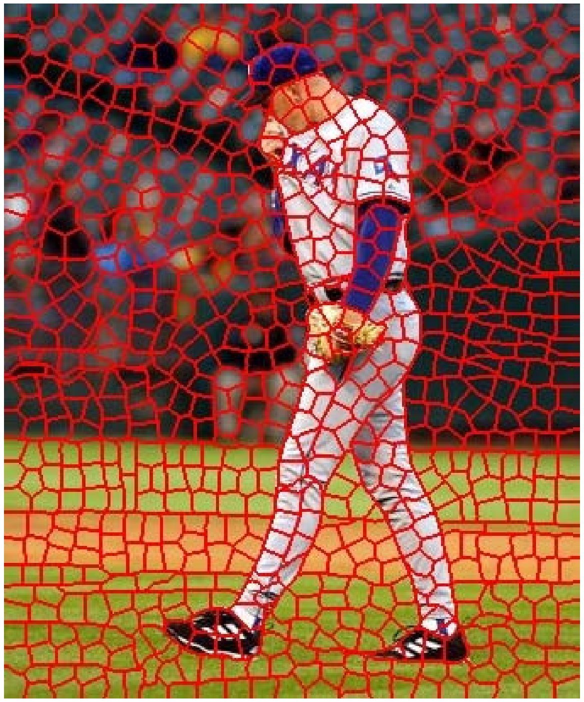
- To obtain primitives for other tasks
- For perceptual organization, recognition
- For graphics, image manipulation

Task 1: Primitives for other tasks

- Group together similar-looking pixels for efficiency of further processing
  - "Bottom-up" process
  - Unsupervised

"superpixels"

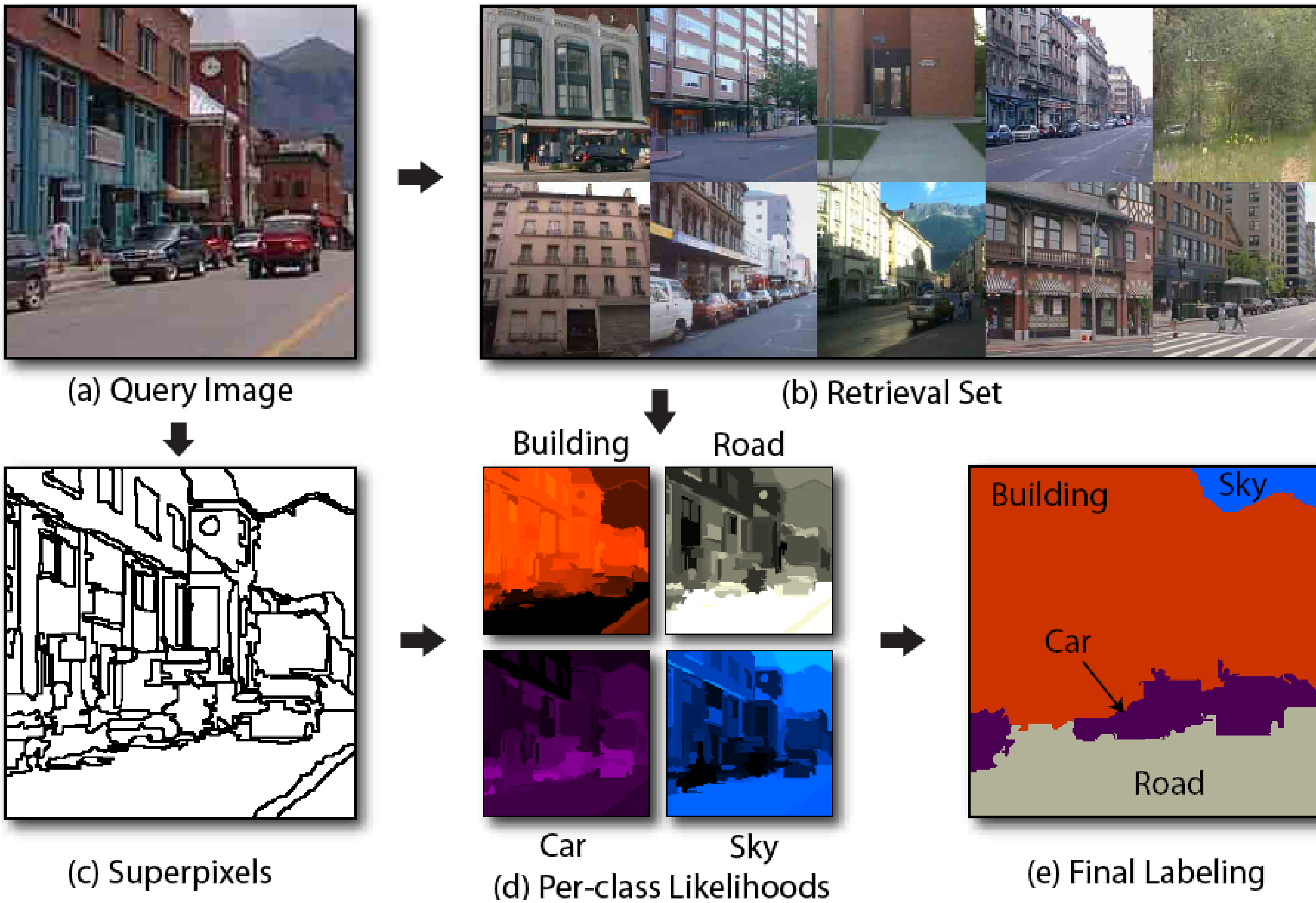




X. Ren and J. Malik. [Learning a classification model for segmentation.]() ICCV 2003.

Example of segments as primitives for recognition

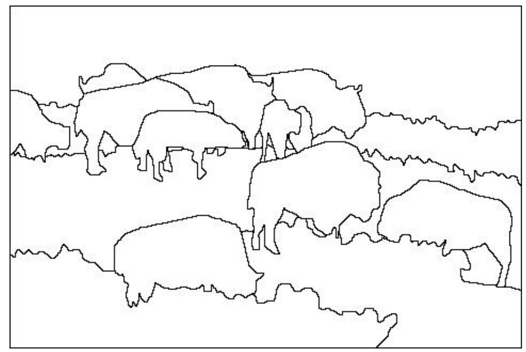
- |



J. Tighe and S. Lazebnik, ECCV 2010, IJCV 2013

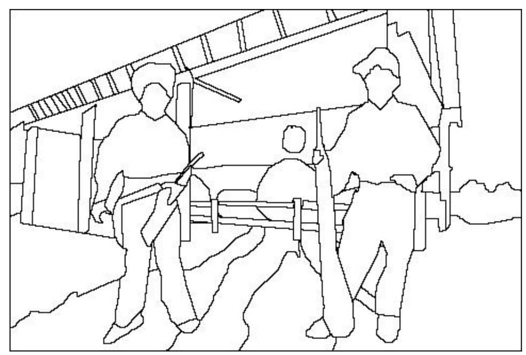
Task 2: Recognition

- Separate image into coherent "objects"
  - "Bottom-up" or "top-down" process?
  - Supervised or unsupervised?



image

human segmentation



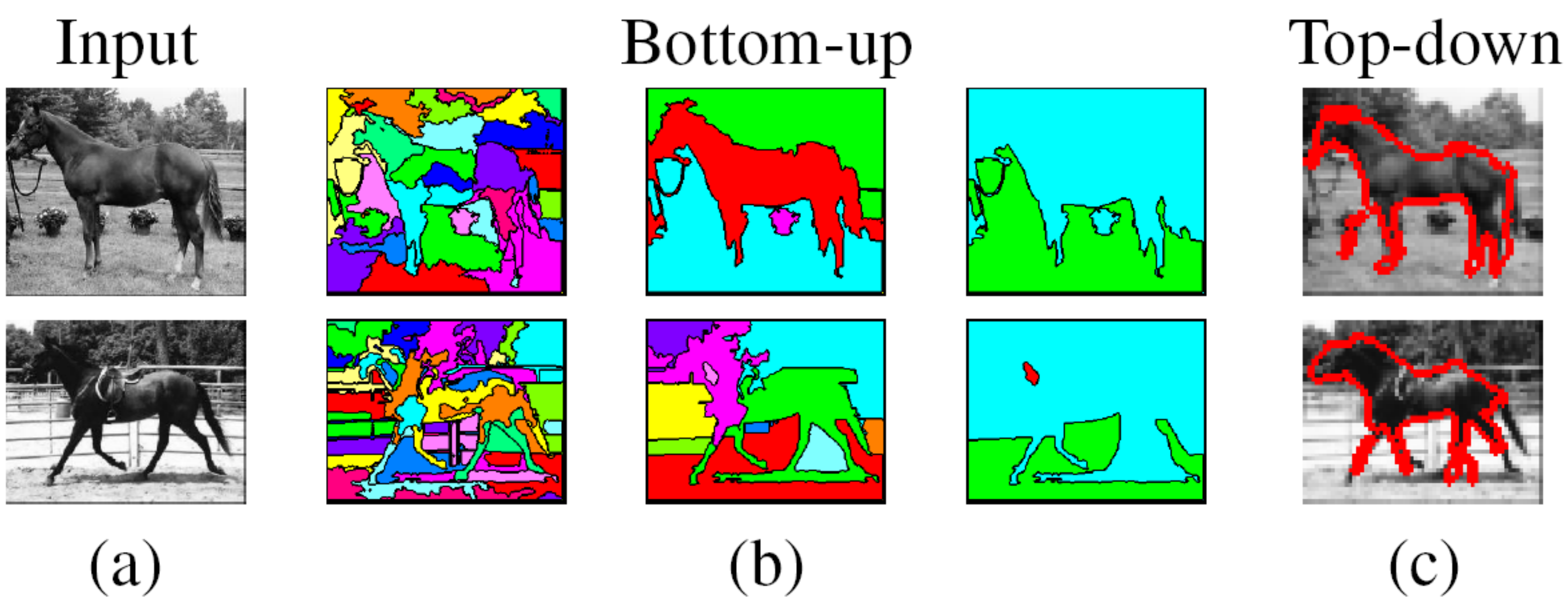
Berkeley segmentation database:
http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

Task 3: Image manipulation

- Interactive segmentation for graphics

High-level approaches to segmentation

- Bottom-up: group tokens with similar features
- Top-down: group tokens that likely belong to the same object



[Levin and Weiss 2006]

Approaches to segmentation

- Segmentation as clustering
- **Segmentation as graph partitioning**
- Segmentation as labeling

Segmentation as clustering

- **Clustering**: grouping together similar points and represent them with a single token
- Key Challenges:
  1. What makes two points/images/patches similar?
  2. How do we compute an overall grouping from pairwise similarities?

How to evaluate clusters?

- Generative
  - How well are points synthesized from the clusters?
- Discriminative
  - How well do the clusters correspond to labels?
    - Purity

Note: unsupervised clustering does not aim to be discriminative

Common similarity/distance measures

- P-norms $$\|\mathbf{x}\|_p := \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}$$
  - City Block (L1) $$\|\boldsymbol{x}\|_1 := \sum_{i=1}^{n} |x_i|$$
  - Euclidean (L2) $$\|\boldsymbol{x}\| := \sqrt{x_1^2 + \cdots + x_n^2}$$
  - L-infinity $$\|\mathbf{x}\|_\infty := \max\left(|x_1|, \ldots, |x_n|\right)$$

Here x_i is the distance between two points

- Mahalanobis
  - Scaled Euclidean $$d(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{N} \frac{(x_i - y_i)^2}{\sigma_i^2}}$$

- Cosine distance $$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$

Kernel density estimation (KDE)

- A non-parametric way to estimate the probability density function of a random variable.
- Inferences about a population are made based only on a finite data sample.

Kernel density estimation

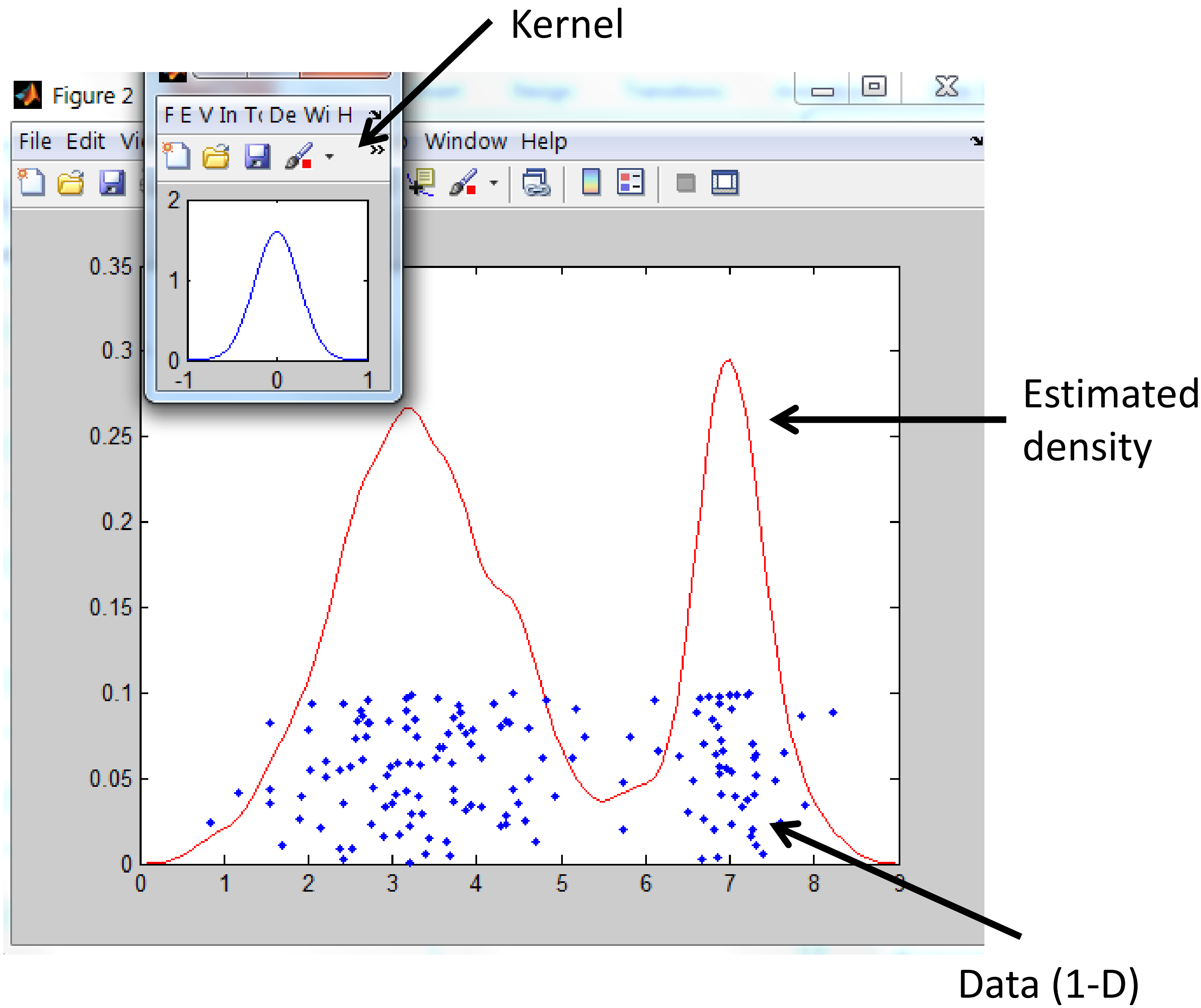
Kernel density estimation function

$$\widehat{f}_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

Gaussian kernel

$$K\left(\frac{x - x_i}{h}\right) = \frac{1}{\sqrt{2\pi}}\, e^{-\frac{(x-x_i)^2}{2h^2}}.$$

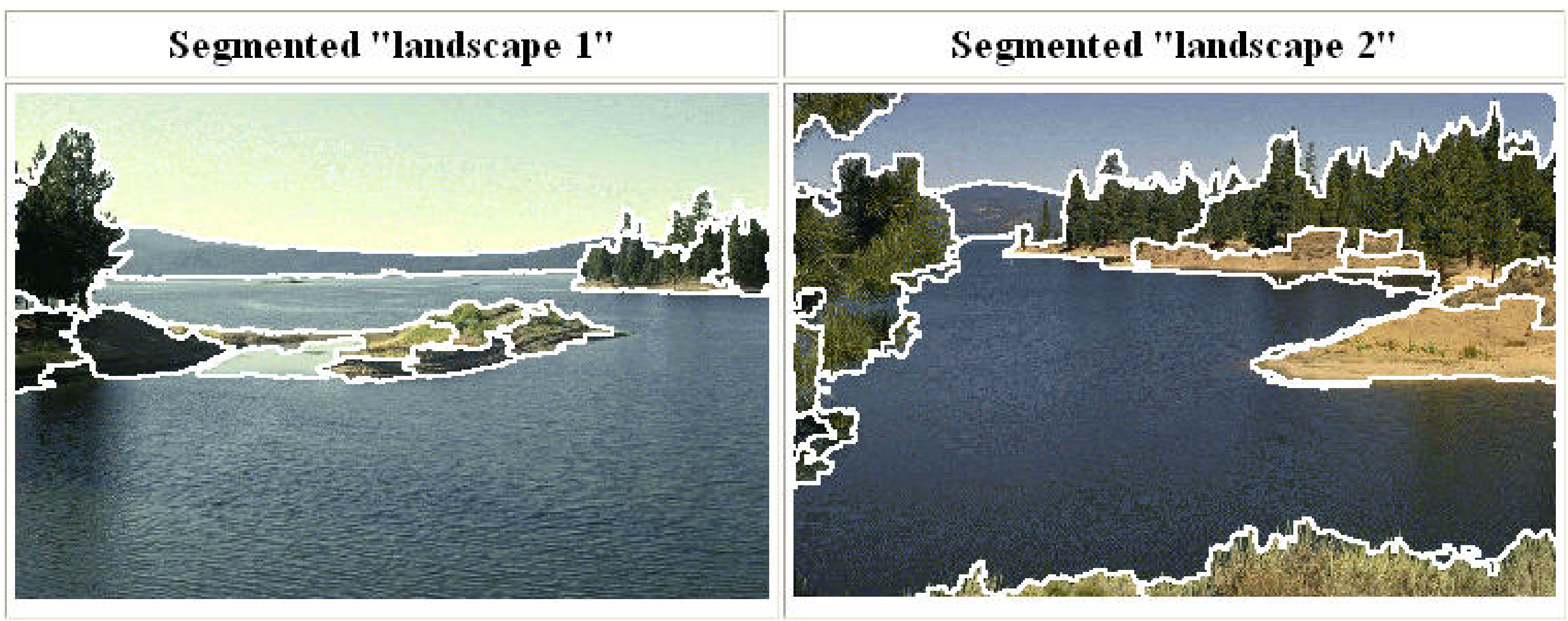
Kernel density estimation



Mean shift segmentation

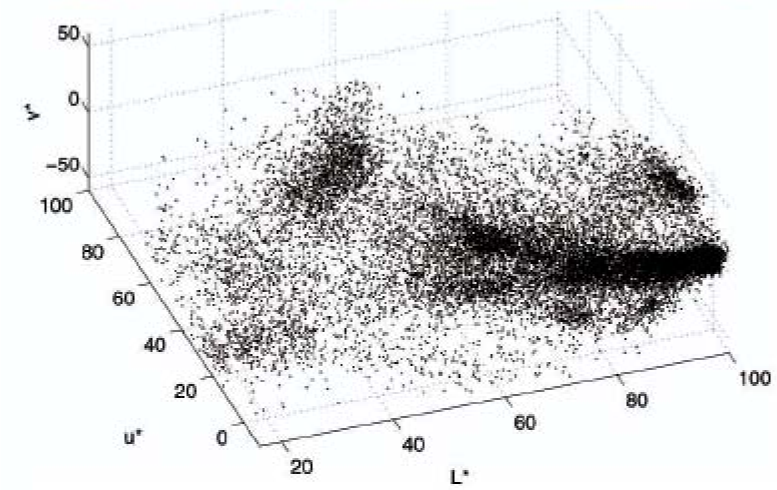
D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002.

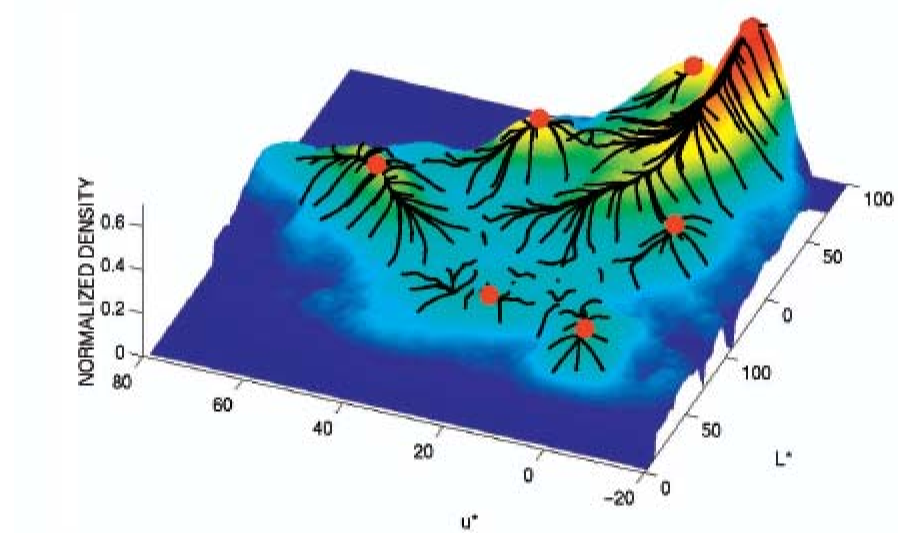
- Versatile technique for clustering-based segmentation

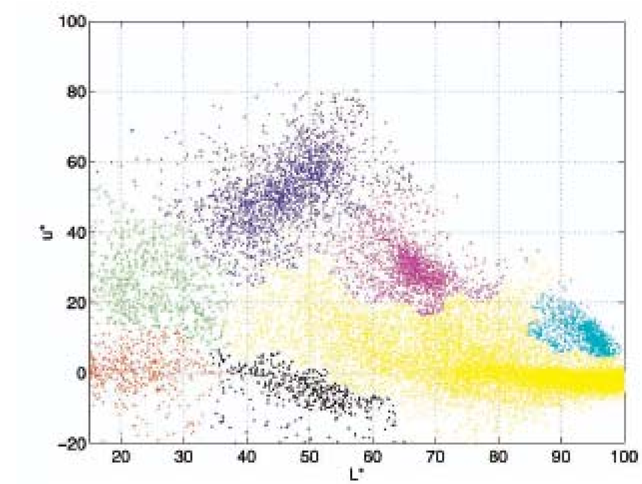


Mean shift algorithm

- Try to find *modes* of this non-parametric density







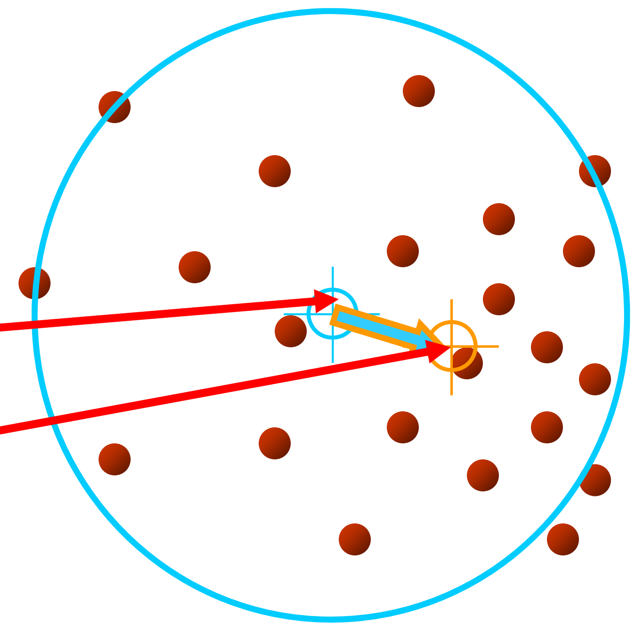
Computing the Mean Shift

<u>Simple Mean Shift procedure</u>:

• Compute mean shift vector

•Translate the Kernel window by **m(x)**

$$\mathbf{m}(\mathbf{x}) = \left[\frac{\sum_{i=1}^{n} \mathbf{x}_i g\left(\frac{\left\| \mathbf{x} - \mathbf{x}_i \right\|^2}{h} \right)}{\sum_{i=1}^{n} g\left(\frac{\left\| \mathbf{x} - \mathbf{x}_i \right\|^2}{h} \right)} - \boxed{\mathbf{x}} \right]$$



$g(\mathbf{x}) = -k'(\mathbf{x})$

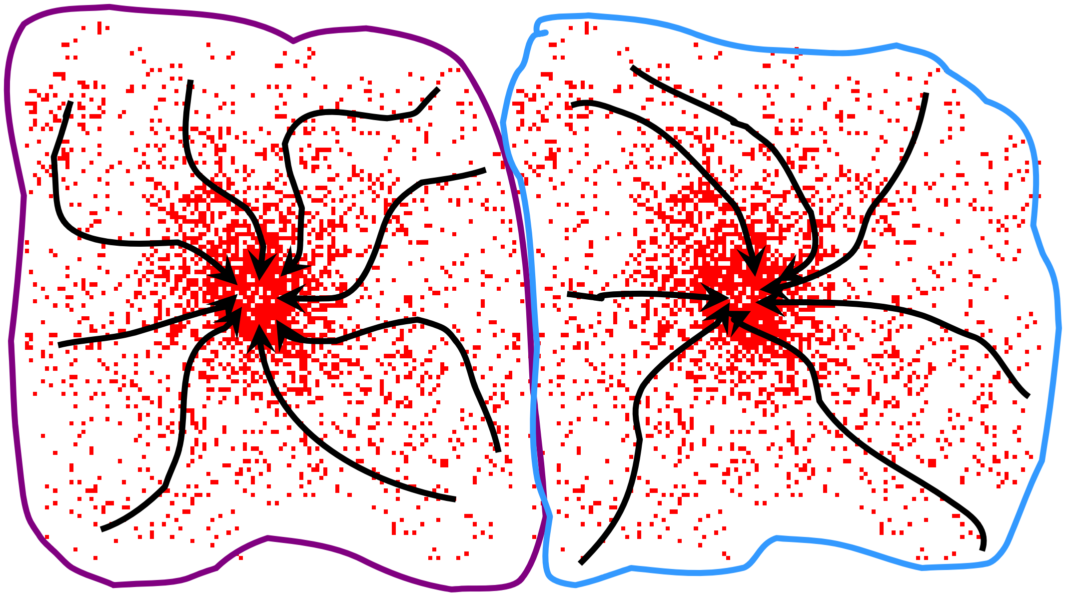
Slide by Y. Ukrainitz & B. Sarel

Mean shift clustering

- The mean shift algorithm seeks *modes* of the given set of points
  1. Choose kernel and bandwidth
  2. For each point:
     a) Center a window on that point
     b) Compute the mean of the data in the search window
     c) Center the search window at the new mean location
     d) Repeat (b,c) until convergence
  3. Assign points that lead to nearby modes to the same cluster

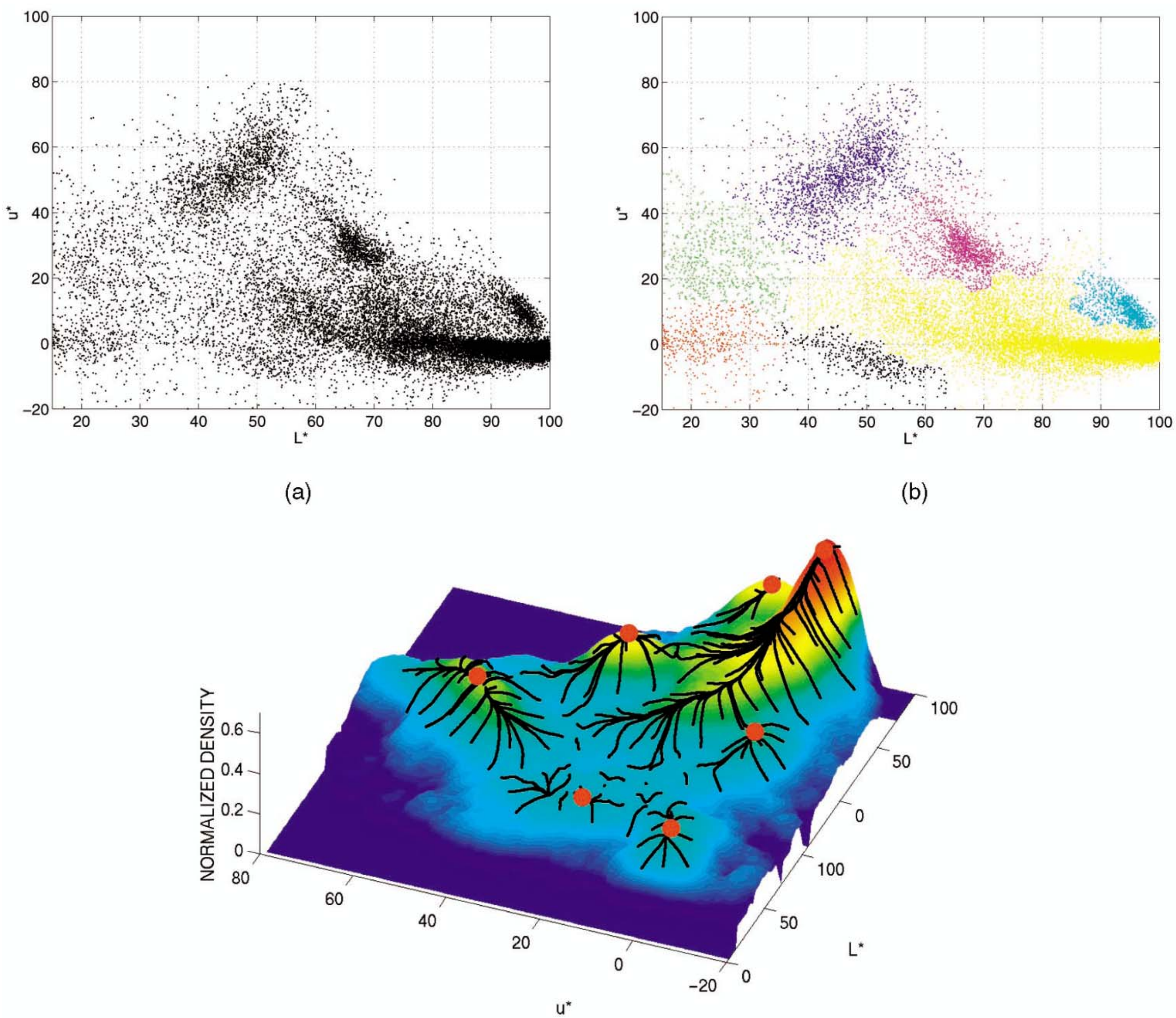
Attraction basin

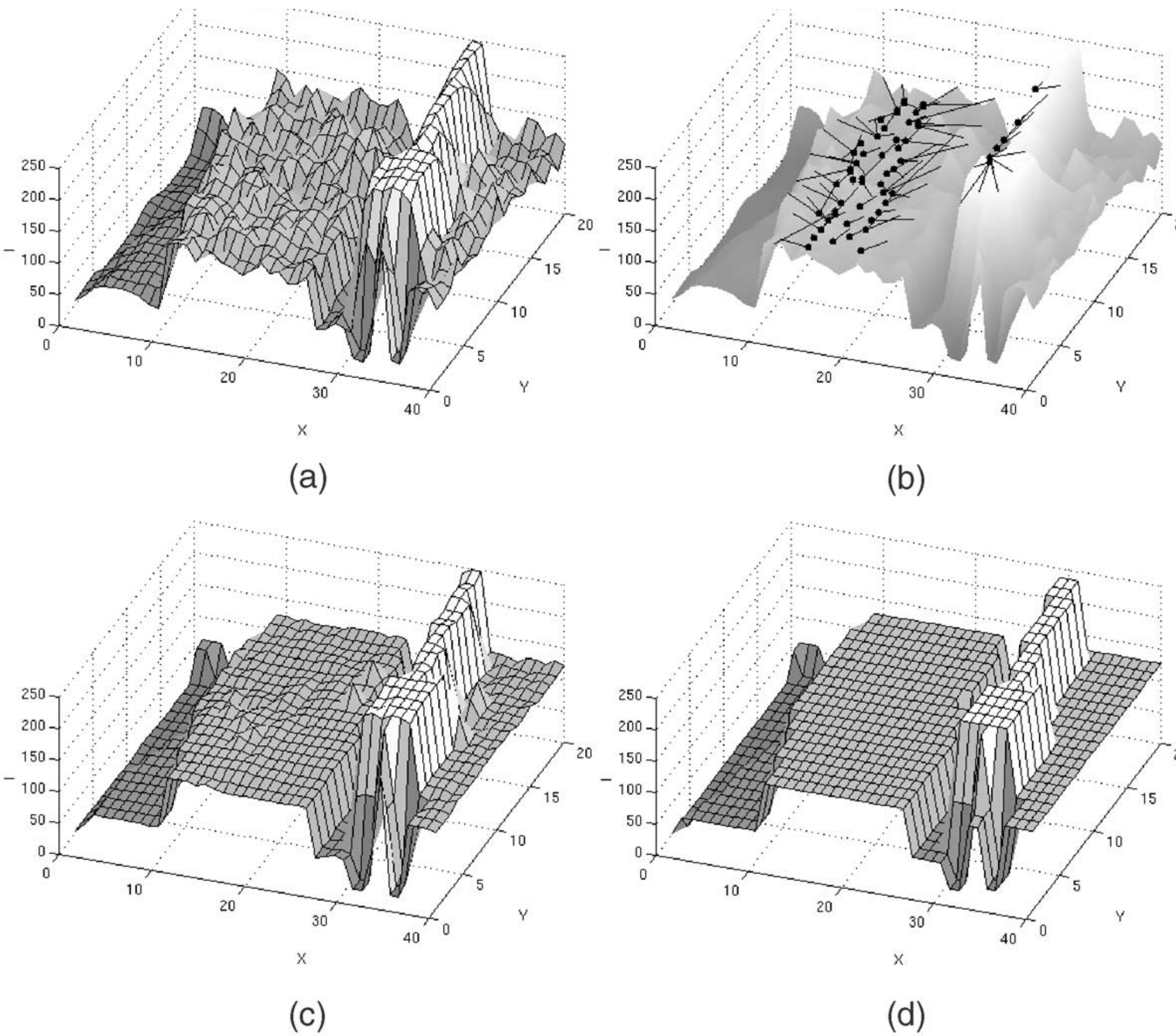
- Attraction basin: the region for which all trajectories lead to the same mode
- Cluster: all data points in the attraction basin of a mode



Slide by Y. Ukrainitz & B. Sarel

Attraction basin





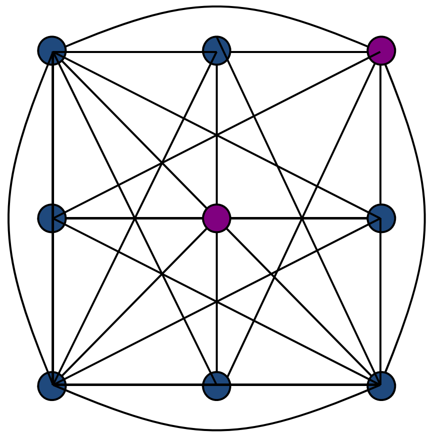
Mean shift filtering and segmentation for grayscale data; (a) input data (b) mean shift paths for the pixels on the plateaus (c) filtering result (d) segmentation result

http://www.caip.rutgers.edu/~comanici/Papers/MsRobustApproach.pdf

Today -

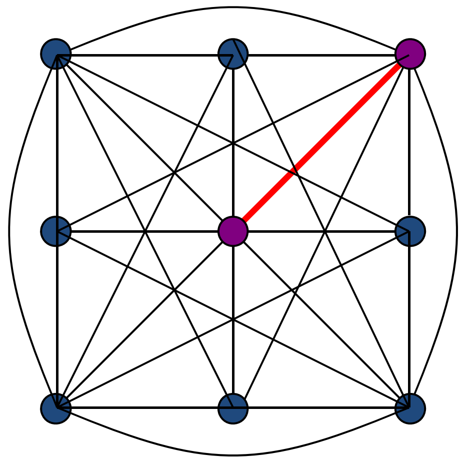
GRAPH BASED CLUSTERING AND SEGMENTATION

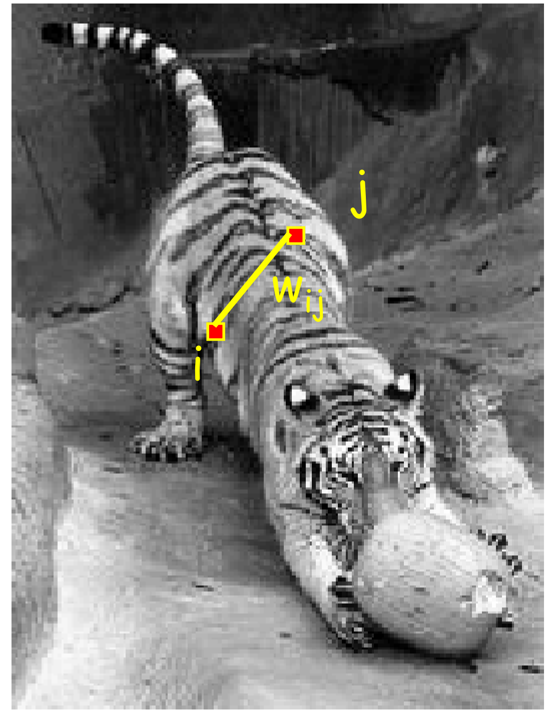
Segmentation as graph partitioning



- The set of points in an arbitrary feature space can be represented as a weighted undirected complete graph
  - $G = (V, E)$, where the nodes of the graph are the points in the feature space.
  - The weight w_{ij} of an edge $(i, j) \in E$ is a function of the similarity between the nodes and i and j.
- We can formulate image segmentation problem as a graph partitioning problem that asks for a partition of $V_1, \ldots\ldots V_k$, of the vertex set V; such that the vertices in Vi have high similarity and those in V_i, V_j have low similarity.

Segmentation as graph partitioning





- Node for every pixel
- Edge between every pair of pixels (or every pair of "sufficiently close" pixels)
- Each edge is weighted by the *affinity* or similarity of the two nodes

Source: S. Seitz

Measuring affinity

Property	Affinity function	Notes
Distance	$\exp\left\{-\left((\boldsymbol{x}-\boldsymbol{y})^t(\boldsymbol{x}-\boldsymbol{y})/2\sigma_d^2\right)\right\}$	
Intensity	$\exp\left\{-\left((I(\boldsymbol{x})-I(\boldsymbol{y}))^t(I(\boldsymbol{x})-I(\boldsymbol{y}))/2\sigma_I^2\right)\right\}$	$I(\boldsymbol{x})$ is the intensity of the pixel at $\boldsymbol{x}$.
Color	$\exp\left\{-\left(\mathrm{dist}(\boldsymbol{c}(\boldsymbol{x}),\boldsymbol{c}(\boldsymbol{y}))^2/2\sigma_c^2\right)\right\}$	$\boldsymbol{c}(\boldsymbol{x})$ is the color of the pixel at $\boldsymbol{x}$.
Texture	$\exp\left\{-\left((\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{f}(\boldsymbol{y}))^t(\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{f}(\boldsymbol{y}))/2\sigma_I^2\right)\right\}$	$\boldsymbol{f}(\boldsymbol{x})$ is a vector of filter outputs describing the pixel at $\boldsymbol{x}$ computed as in Section 6.1.

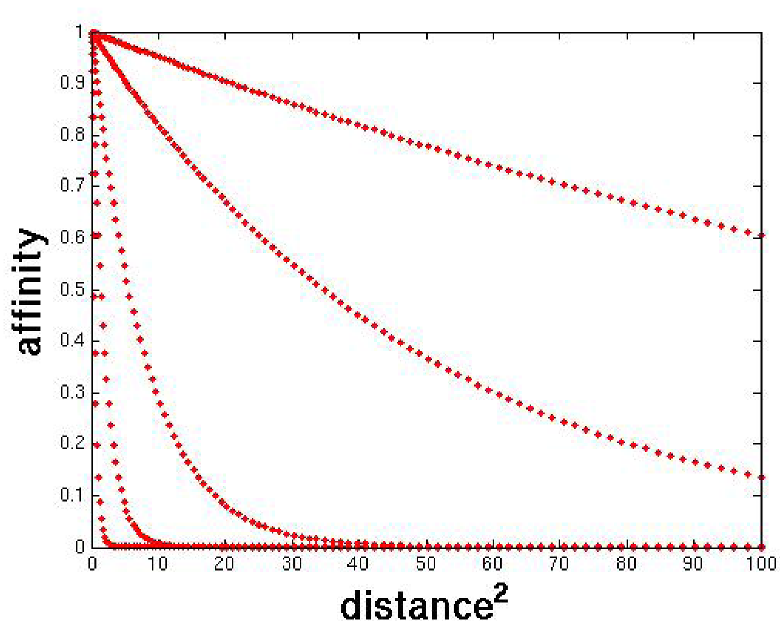
TABLE 9.1: Different affinity functions comparing pixels for a graph based segmenter. Notice that affinities can be combined. One attractive feature of the exponential form is that, say, location, intensity and texture affinities could be combined by multiplying them.

Measuring affinity

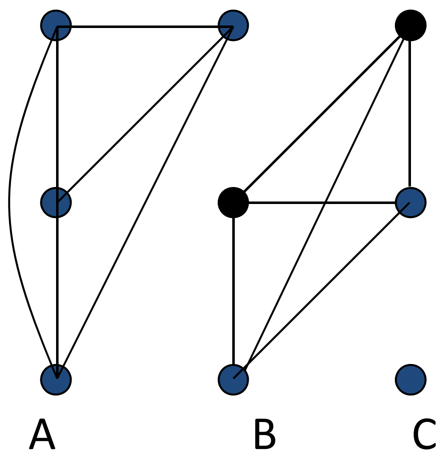
- Represent each pixel by a feature vector **x** and define an appropriate distance function

$$\text{affinity}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2\sigma^2} \text{dist}(\mathbf{x}_i, \mathbf{x}_j)^2 \right)$$

Role of σ



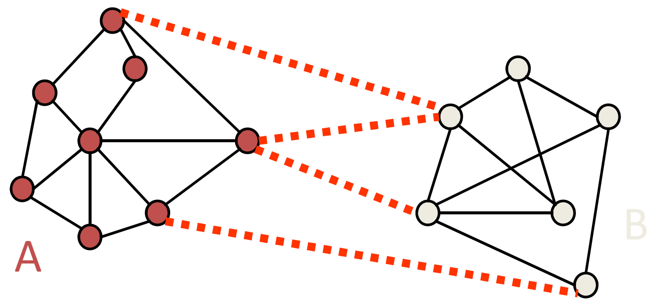
Segmentation as graph partitioning



- Break Graph into Segments
  - Delete links that cross between segments
  - Easiest to break links that have low affinity
    - similar pixels should be in the same segments
    - dissimilar pixels should be in different segments

Source: S. Seitz

General: Graph cut

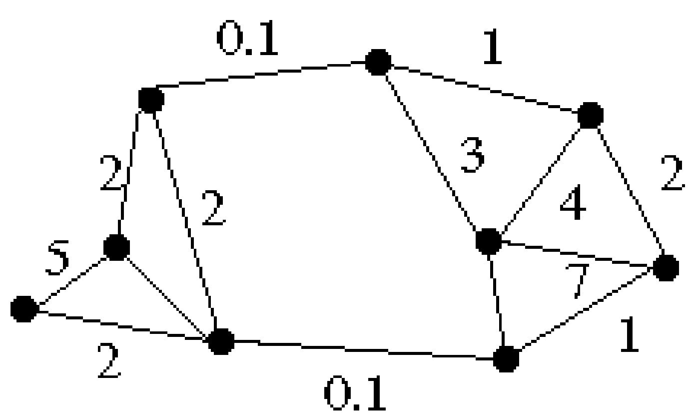


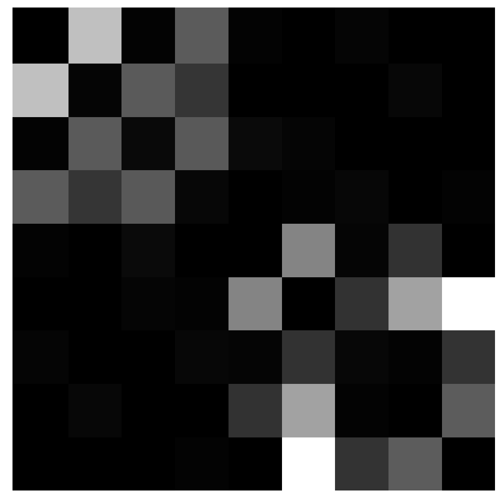
- Set of edges whose removal makes a graph disconnected
- Cost of a cut: sum of weights of cut edges
- A graph cut gives us a segmentation
  - What is a "good" graph cut and how do we find one?

Source: S. Seitz

Minimum cut

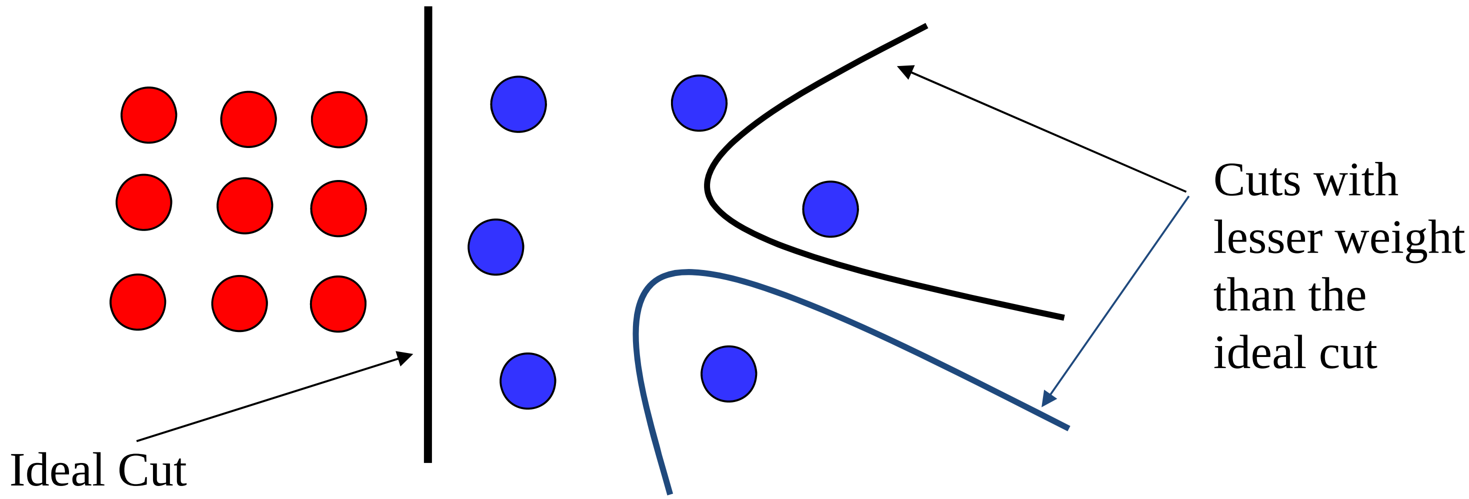
- We can do segmentation by finding the *minimum cut* in a graph
  - Efficient algorithms exist for doing this





Normalized cut

- Drawback: minimum cut tends to cut off very small, isolated components



* Slide from Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Normalized cut

- Drawback: minimum cut tends to cut off very small, isolated components
- This can be fixed by normalizing the cut by the weight of all the edges incident to the segment
- The *normalized cut* cost is:

$$\mathrm{ncut}(A,B) = \frac{w(A,B)}{w(A,V)} + \frac{w(A,B)}{w(B,V)}$$

$w(A, B)$ = sum of weights of all edges between A and B

- Finding the globally optimal cut is NP-complete, but a relaxed version can be solved using a generalized eigenvalue problem

J. Shi and J. Malik. Normalized cuts and image segmentation. PAMI 2000

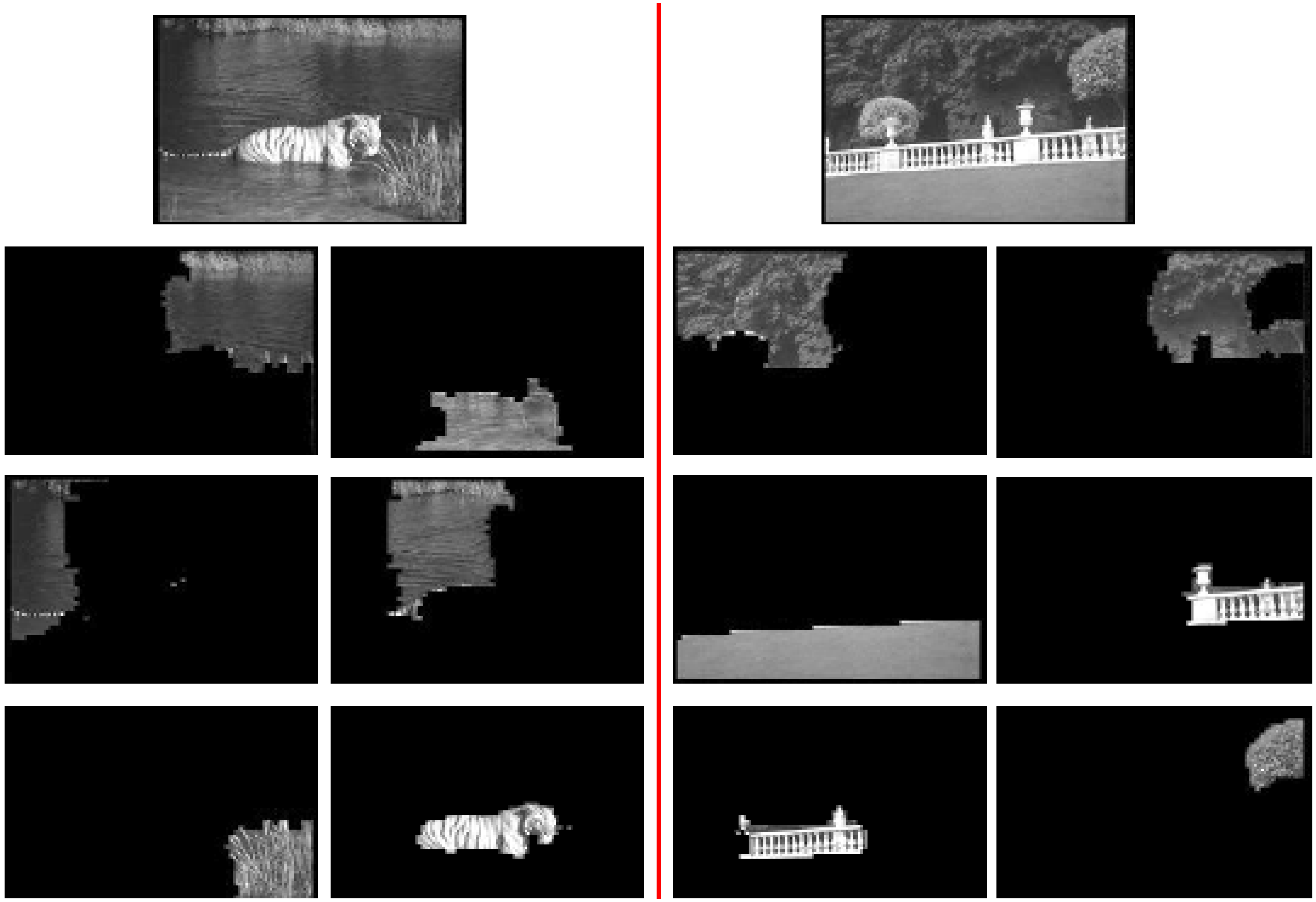


FIGURE 9.24: The images on top are segmented using the normalized cuts framework, described in the text, into the components shown. The affinity measures used involved intensity and texture, as in Table 9.1. The image of the swimming tiger yields one segment that is essentially tiger, one that is grass, and four components corresponding to the lake. Similarly, the railing shows as three reasonably coherent segments. Note the improvement over k-means segmentation obtained by having a texture measure. *This figure was originally published as Figure 2 of "Image and video segmentation: the normalized cut framework," by J. Shi, S. Belongie, T. Leung, and J. Malik, Proc. IEEE Int. Conf. Image Processing, 1998 © IEEE, 1998.*

Normalized cuts: Pro and con

- Pro
  - Generic framework, can be used with many different features and affinity formulations
- Con
  - High storage requirement and time complexity: involves solving a generalized eigenvalue problem of size $n \times n$, where n is the number of pixels

Segmentation as labeling

- Suppose we want to segment an image into foreground and background
  - Binary labeling problem

Credit: N. Snavely

Segmentation as labeling

- Suppose we want to segment an image into foreground and background
  - Binary labeling problem

User sketches out a few strokes on foreground and background…

How do we label the rest of the pixels?

Source: N. Snavely

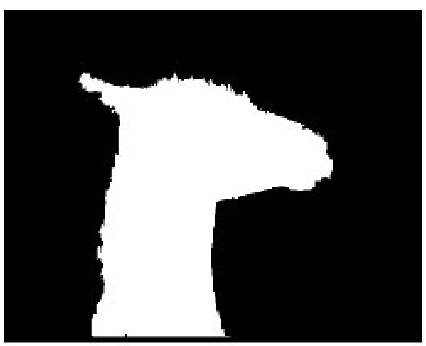
Binary segmentation as energy minimization

- Define a labeling *L* as an assignment of each pixel with a 0-1 label (background or foreground)
- Find the labeling *L* that minimizes

$$E(L) = E_d(L) + \lambda E_s(L)$$

data term

smoothness term



How similar is each labeled pixel to the foreground or background?

Encourage spatially coherent segments

Source: N. Snavely

$$E(L) = E_d(L) + \lambda E_s(L)$$

$\tilde{L}(x,y)$

$$E_d(L) = \sum_{(x,y)} C(x,y,L(x,y))$$

$$C(x,y,L(x,y)) = \begin{cases} \infty & \text{if } L(x,y) \neq \tilde{L}(x,y) \\ C'(x,y,L(x,y)) & \text{otherwise} \end{cases}$$

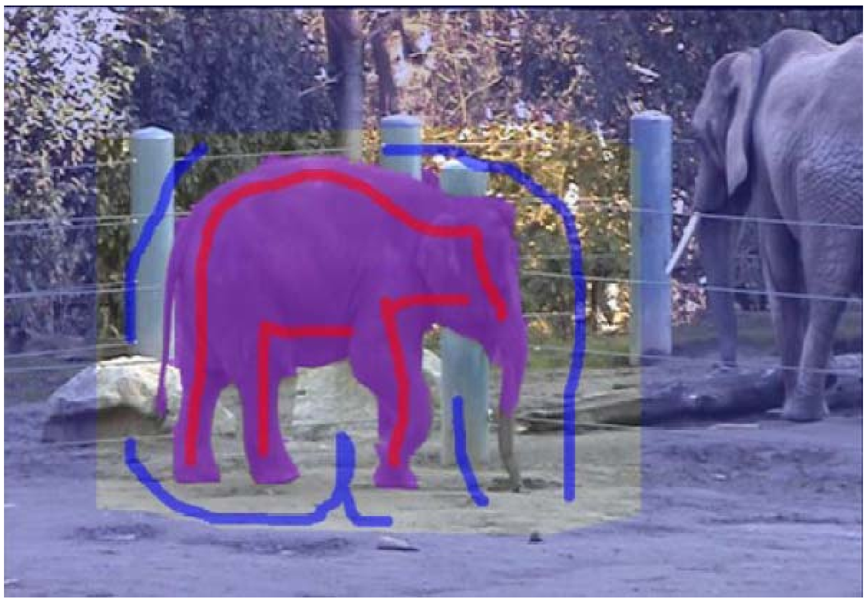
$C'(x,y,0)$: “distance” from pixel to background

$C'(x,y,1)$: “distance” from pixel to foreground

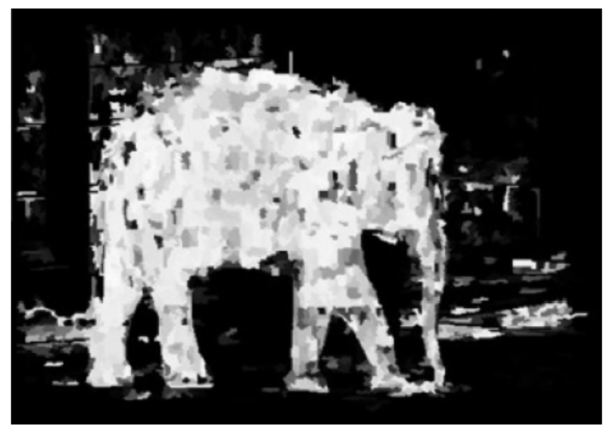
} computed by creating a *color model* from user-labeled pixels

Source: N. Snavely

$$E(L) = E_d(L) + \lambda E_s(L)$$



$C'(x, y, 0)$



$C'(x, y, 1)$

Source: N. Snavely

$$E(L) = E_d(L) + \lambda E_s(L)$$

- Neighboring pixels should generally have the same labels
  - Unless the pixels have very different intensities

w_{pq} = 10.0

w_{pq} = 0.1

$$E_s(L) = \sum_{\text{neighbors } (p,q)} w_{pq} |L(p) - L(q)|$$

w_{pq} : similarity in intensity of *p* and *q*

Source: N. Snavely

Binary segmentation as energy minimization

$$E(L) = E_d(L) + \lambda E_s(L)$$

- For this problem, we can efficiently find the global minimum using the max flow / min cut algorithm

Y. Boykov and M.-P. Jolly, Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Images, ICCV 2001

Source: N. Snavely

Efficient graph-based segmentation

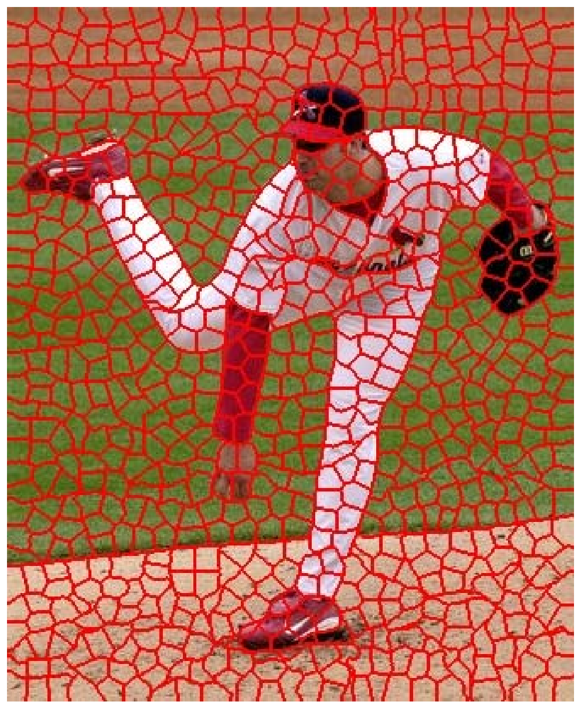


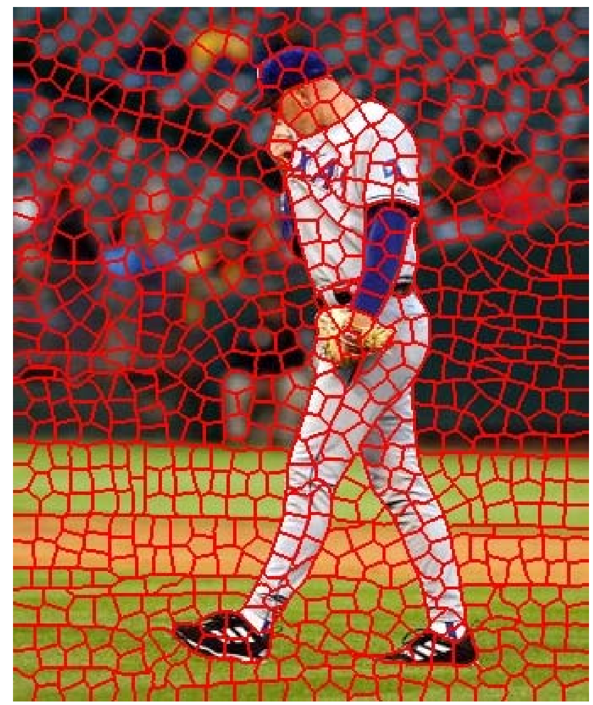

P. Felzenszwalb and D. Huttenlocher, Efficient Graph-Based Image Segmentation, IJCV 2004

Superpixels

- Group together similar-looking pixels for efficiency of further processing
  - “Bottom-up” process
  - Unsupervised

“superpixels”





X. Ren and J. Malik. [Learning a classification model for segmentation.]() ICCV 2003.

Felzenszwalb and Huttenlocher: Graph-Based Segmentation

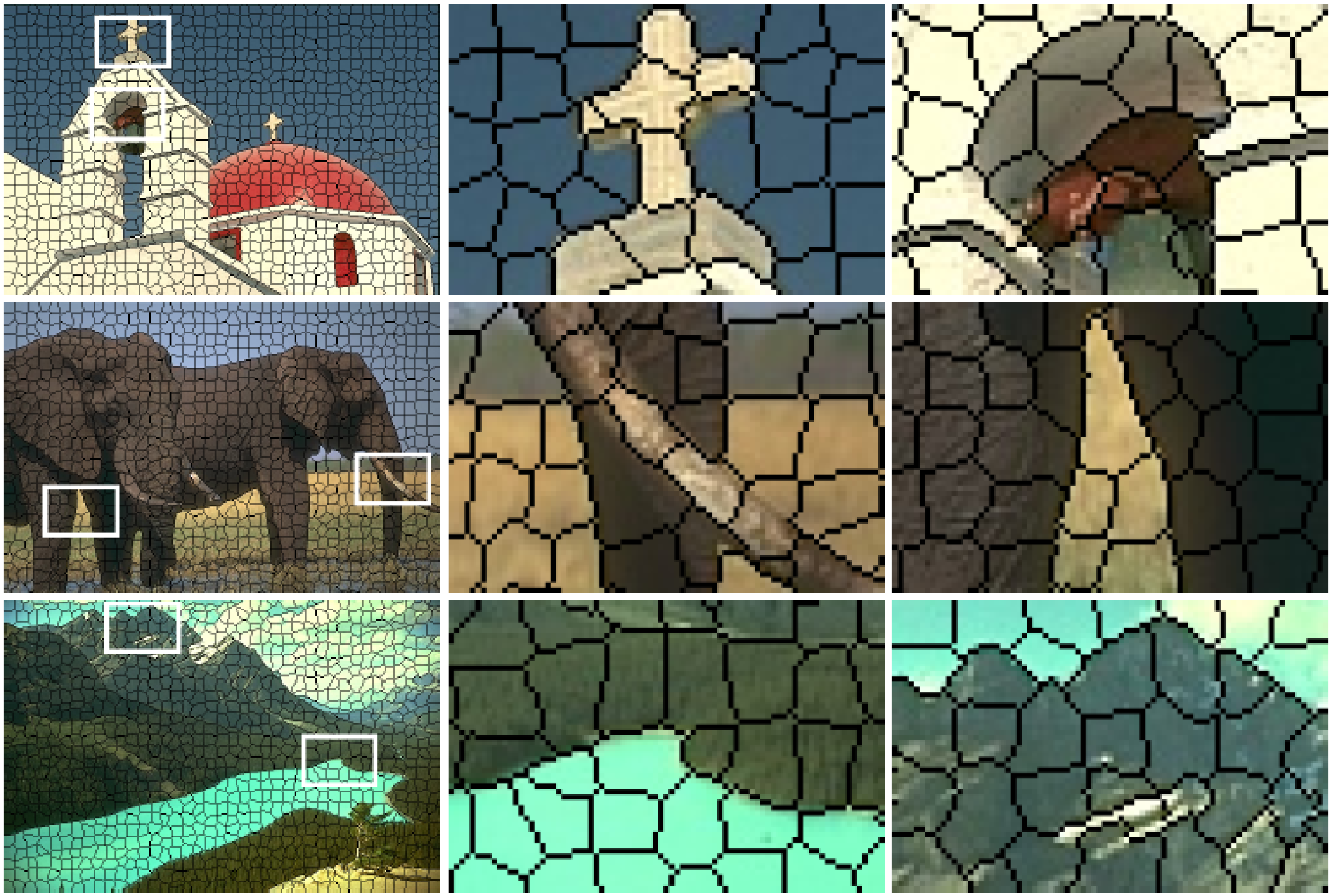
http://www.cs.brown.edu/~pff/segment/

+ Good for thin regions
+ Fast, runs in time nearly linear in the number of edges
+ Easy to control coarseness of segmentations
+ Can include both large and small regions
- Often creates regions with strange shapes
- Sometimes makes very large errors

Turbo Pixels: Levinstein et al. 2009

http://www.cs.toronto.edu/~kyros/pubs/09.pami.turbopixels.pdf

Tries to preserve boundaries and produces more regular regions



Applications of segmentation

- Shot boundary detection
  - summarize videos by
    - finding shot boundaries
    - obtaining “most representative” frame
- Background subtraction
  - find “interesting bits” of image by subtracting known background
    - e.g. sports videos
    - e.g. find person in an office
    - e.g. find cars on a road
- Interactive segmentation
  - user marks some foreground/background pixels
  - system cuts object out of image
    - useful for image editing, etc.

Final thoughts

- Segmentation is a difficult topic with a huge variety of implementations.
  - It is typically hard to assess the performance of a segmenter at a level more useful than that of showing some examples
- There really is not much theory available to predict what should be clustered and how.
- Everyone should know about some clustering techniques like k-means, mean shift, and at least one graph-based clustering algorithm
  - these ideas are just so useful for so many applications
  - segmentation is just one application of clustering.

Slide Credits

- Svetlana Lazebnik – UIUC
- Derek Hoiem – UIUC
- David Forsyth - UIUC

Questions