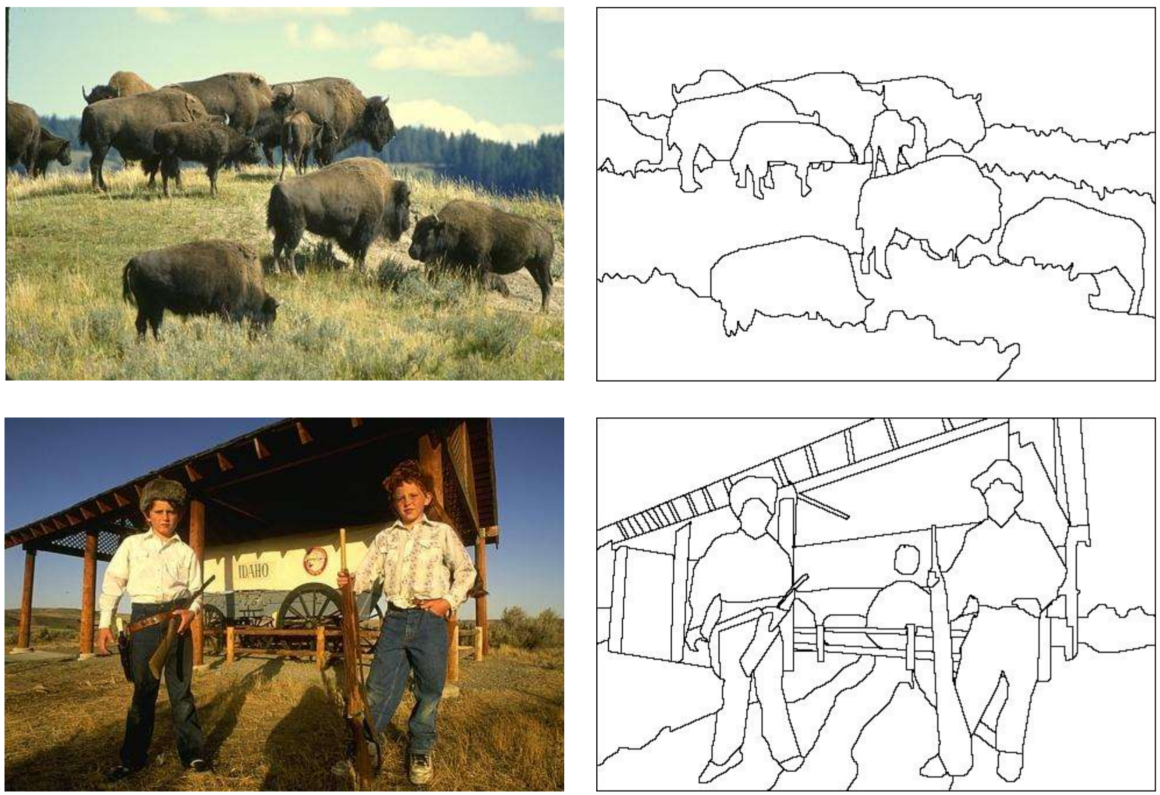
Image segmentation

Κώστας Παπουτσάκης, CVRL, Ινστιτούτο Πληροφορικής, ΙΤΕ.

ΔΠΜΣ Προηγμένα Συστήματα Παραγωγής, Αυτοματισμού και Ρομποτικής, ΕΛΜΕΠΑ, 04-12-2020

Definition of Segmentation

- Separate image into multiple, coherent segments (sets of pixels → "image objects"!)
- Reading material: Forsyth 9, Szeliski 5.

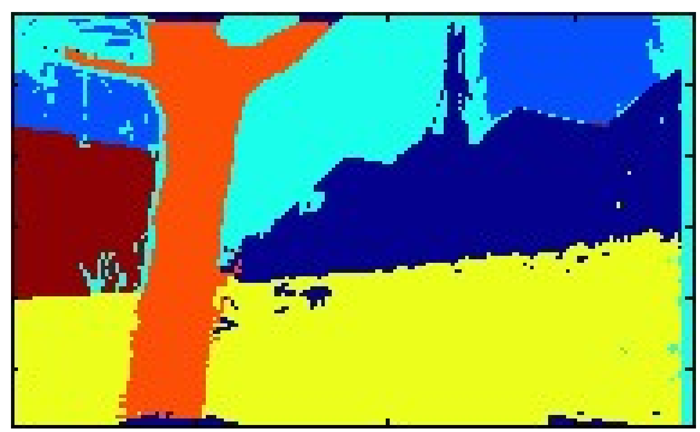


Region Segmentation

- **Given an image I, find sets of pixels, $R_1, \ldots, Rn$ such that:**

$$\bigcup_i R_i = I \qquad \forall i \neq j, R_i \cap R_j = \varnothing$$

  - The process of assigning a label to every pixel in an image,
  - such that pixels with the same label (region) share certain characteristics or similarity constraints.
    - color, intensity, or texture, depth, combination of cues (?)
  - Adjacent regions are significantly different with respect to the same characteristic(s).



The goals of segmentation

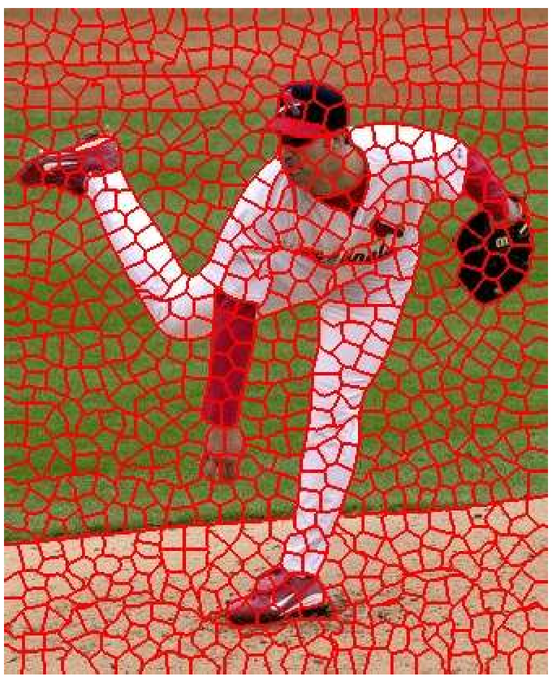
- Obtain primitives for other tasks
  - Simplify and/or change the representation of an image into something that is more meaningful and easier to analyze

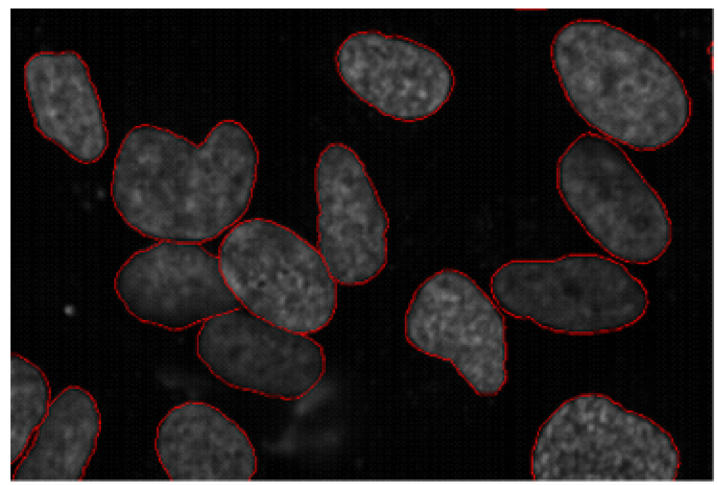
- Perceptual organization, recognition
  - Used to locate objects and boundaries (lines, curves, etc.).

- Graphics, image manipulation, 3D reconstruction.

Goal 1: Primitives for other tasks

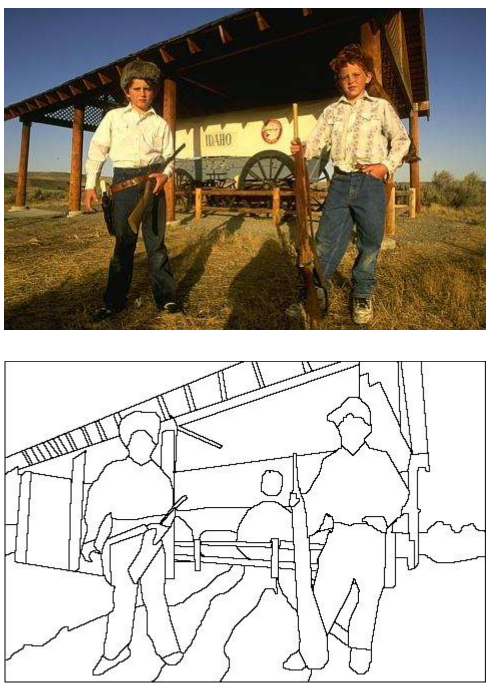
- Group together similar-looking pixels for efficiency of further processing
- Simplify and/or change the representation of an image into something that is more meaningful and easier to analyze
  - "Bottom-up" process
  - Unsupervised

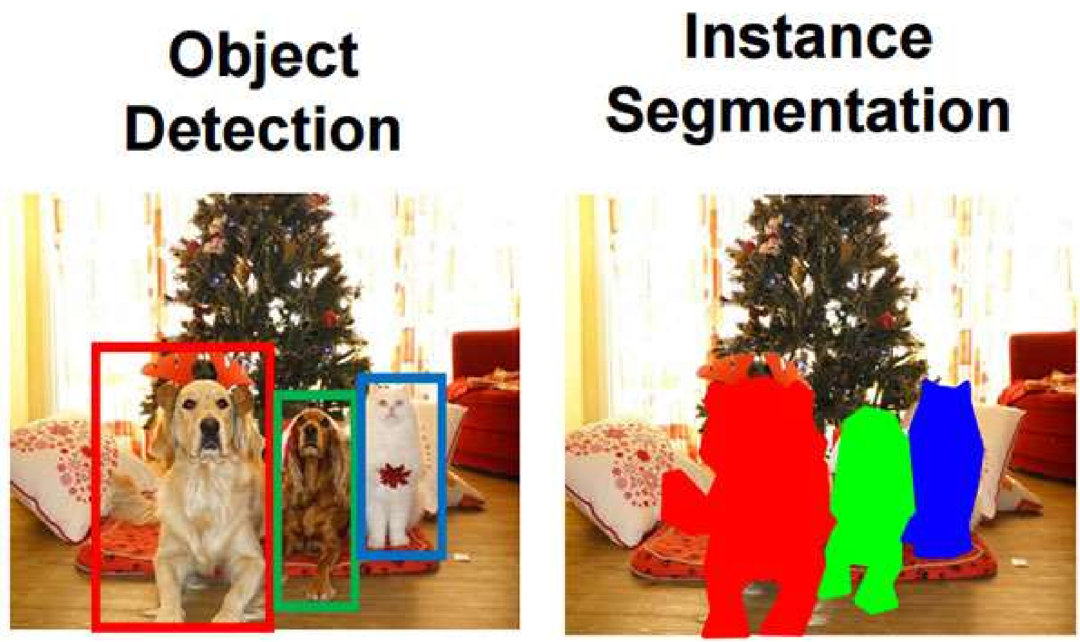




Goal 2: Recognition

- Used to locate objects and boundaries (lines, curves, etc.).
- Separate image into coherent "objects"
  - "Bottom-up" or "top-down" process?
  - Supervised or unsupervised?





Goal 3: Image manipulation

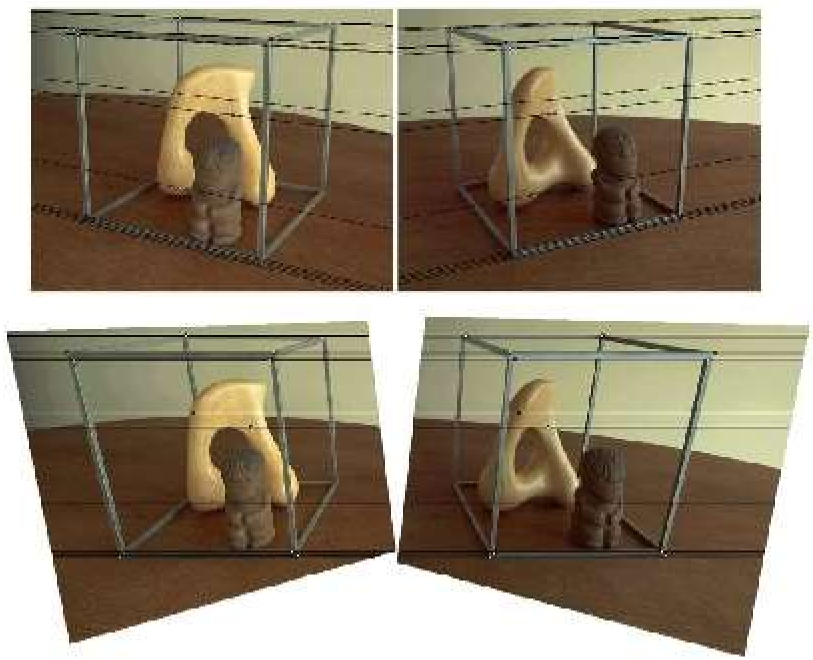
- Interactive segmentation for graphics





Goal 4: 3D reconstruction

- Correspondence search with similarity constraint
  - Object masks to speed up the process



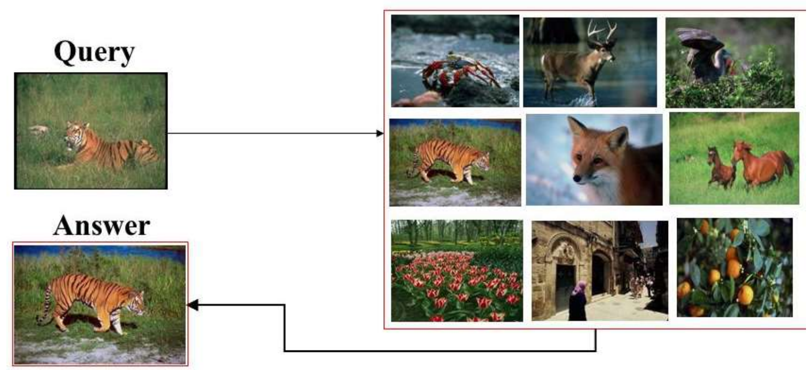
Applications

- object recognition

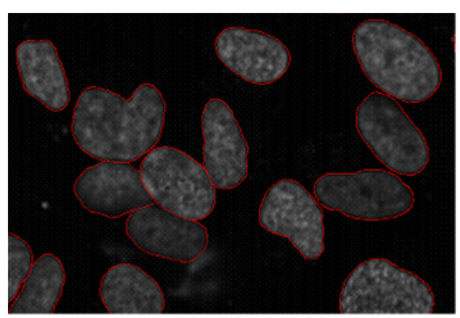
Image Localization

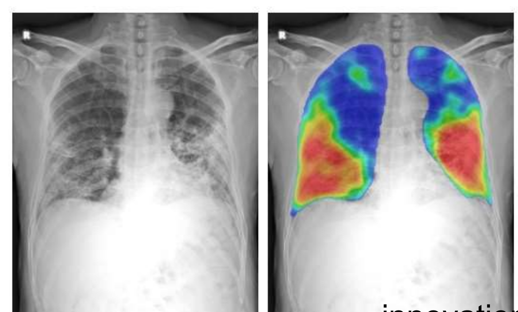
Object Detection

- image retrieval



- medical image analysis

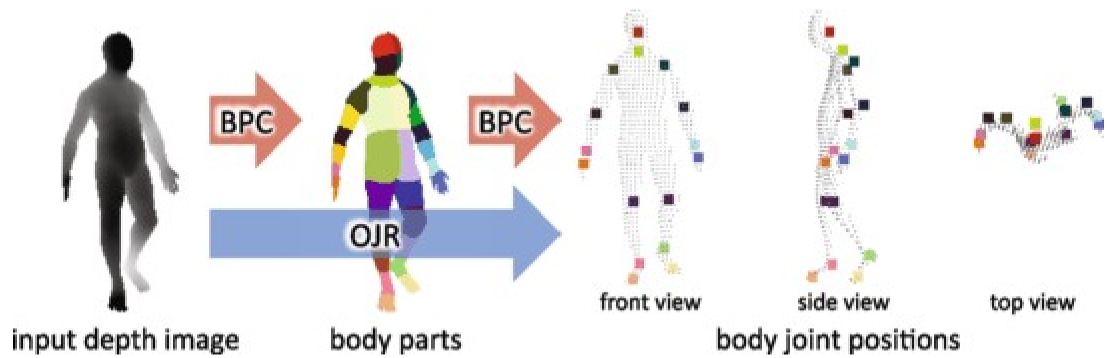




innovationorigins.com

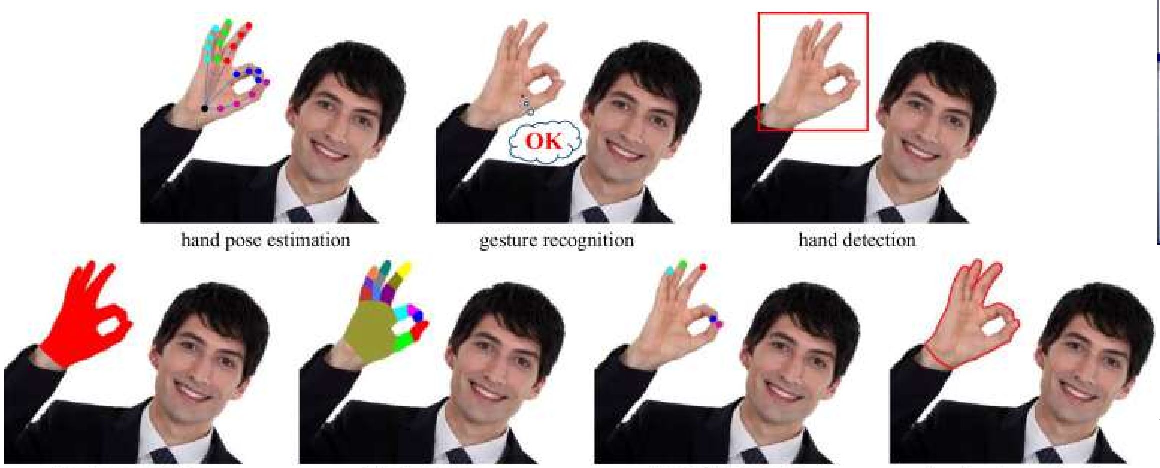
Applications – Human/Hand pose estimation

(a) (b) (c) (d)



Efficient Human Pose Estimationfrom Single Depth Images, Shotton et.al. PAMI2012

Chen, C. et al. "Segmentation of Human Body Parts Using Deformable Triangulation." IEEE Trans. SMCSH (2010)



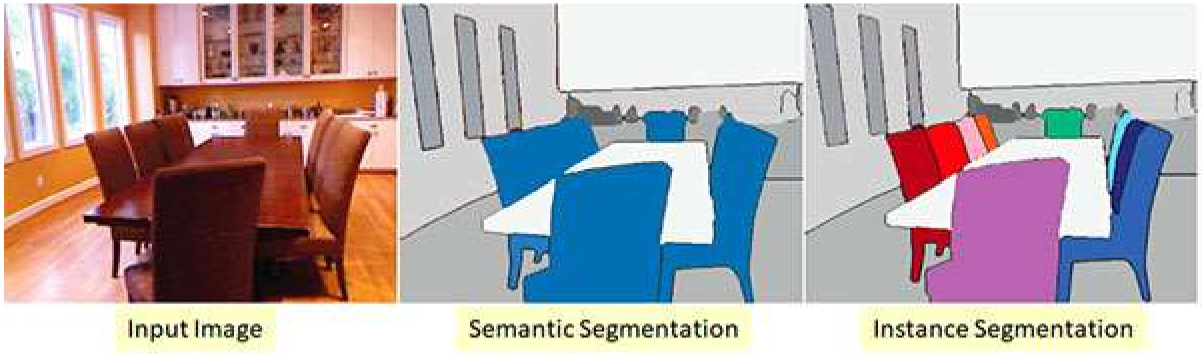
hand pose estimation, gesture recognition, hand detection, hand segmentation, hand parsing, fingertip detection, hand contour estimation

Li, Zhenyu Liu, Jianrong Tan,
A survey on 3D hand pose
estimation: Cameras, methods, and
datasets, Pattern Recognition 2019

Why image segmentation is important ?

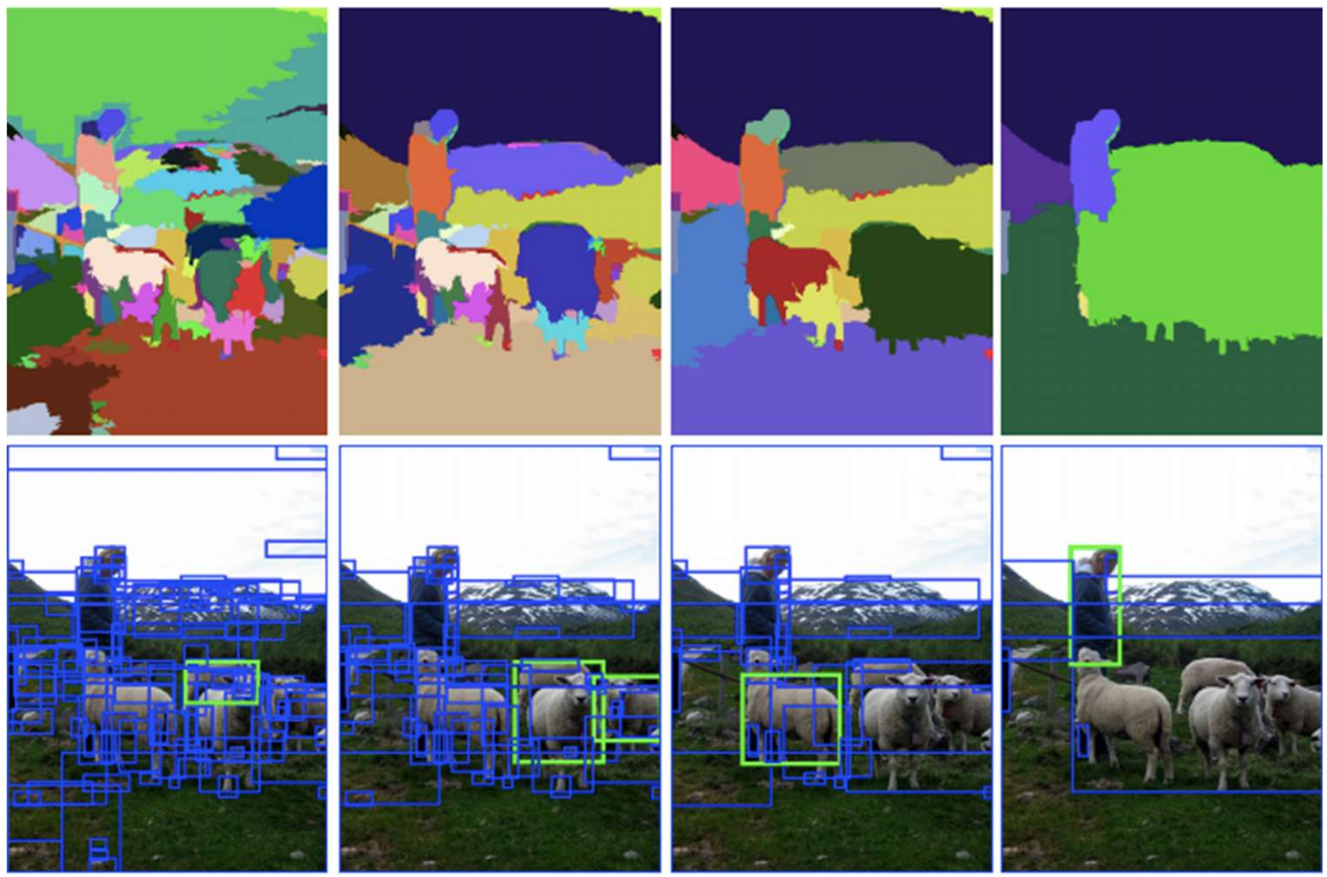
- Semantic segmentation

- Instance segmentation



Object detection

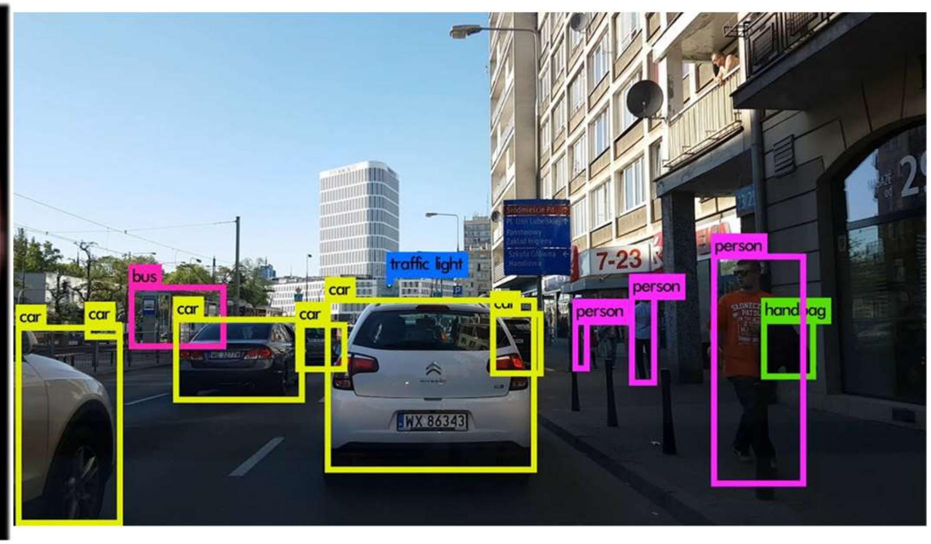
- Selective search: fewer proposals



- Segmentation vs Detection

Pixel-level labels
Category only

Bounding box labels
Category + instance

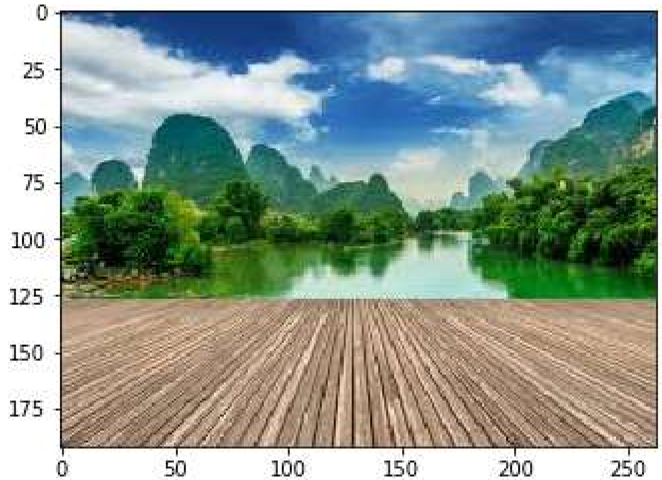


Approaches to segmentation

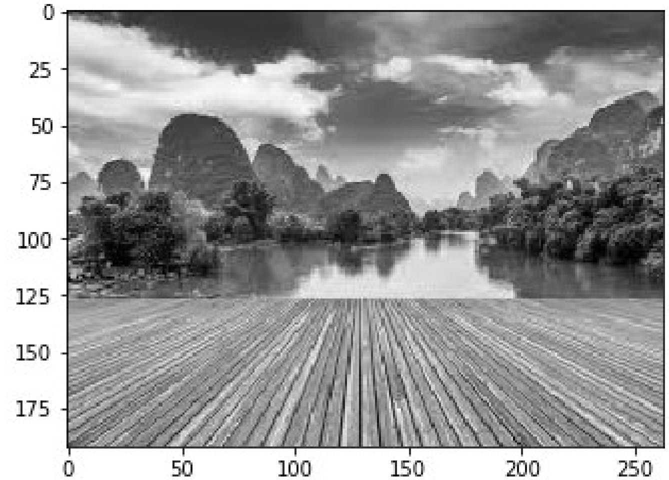
- Region-based segmentation
  - simple way to segment different objects could be to use their pixel values → use threshold values
- Edge Detection Segmentation
  - We can make use of the discontinuity between pixels to detect edges and hence define a boundary of the object for segmentation ?
- Segmentation as clustering.
  - dividing the population (data points) into a number of groups,
  - data points in the same groups → more similar
- Segmentation as graph partitioning.

Region-based segmentation

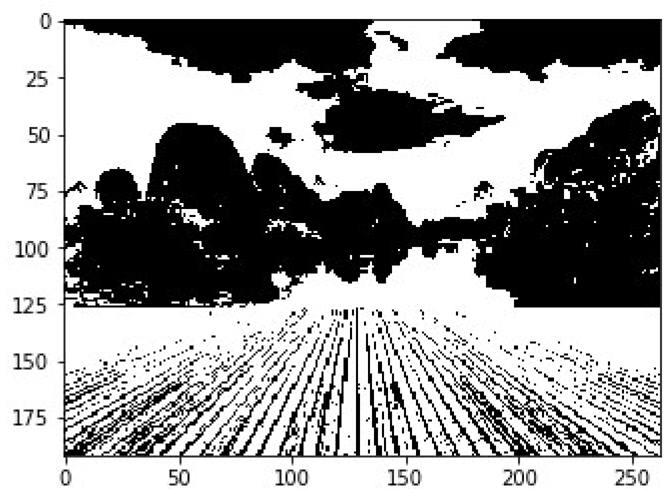
Input image



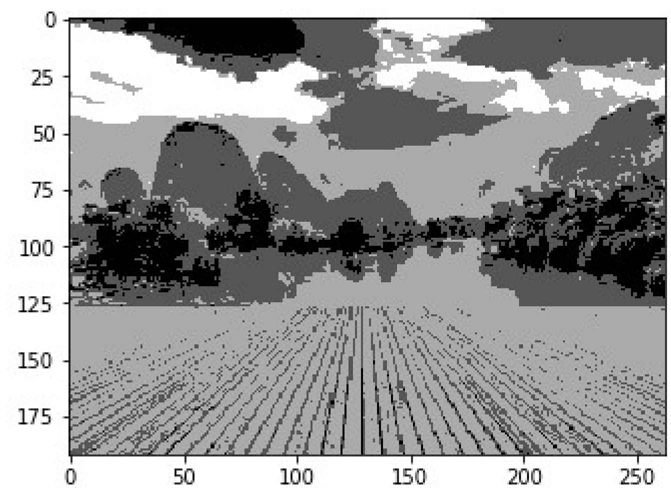
grayscale



Global thresholing



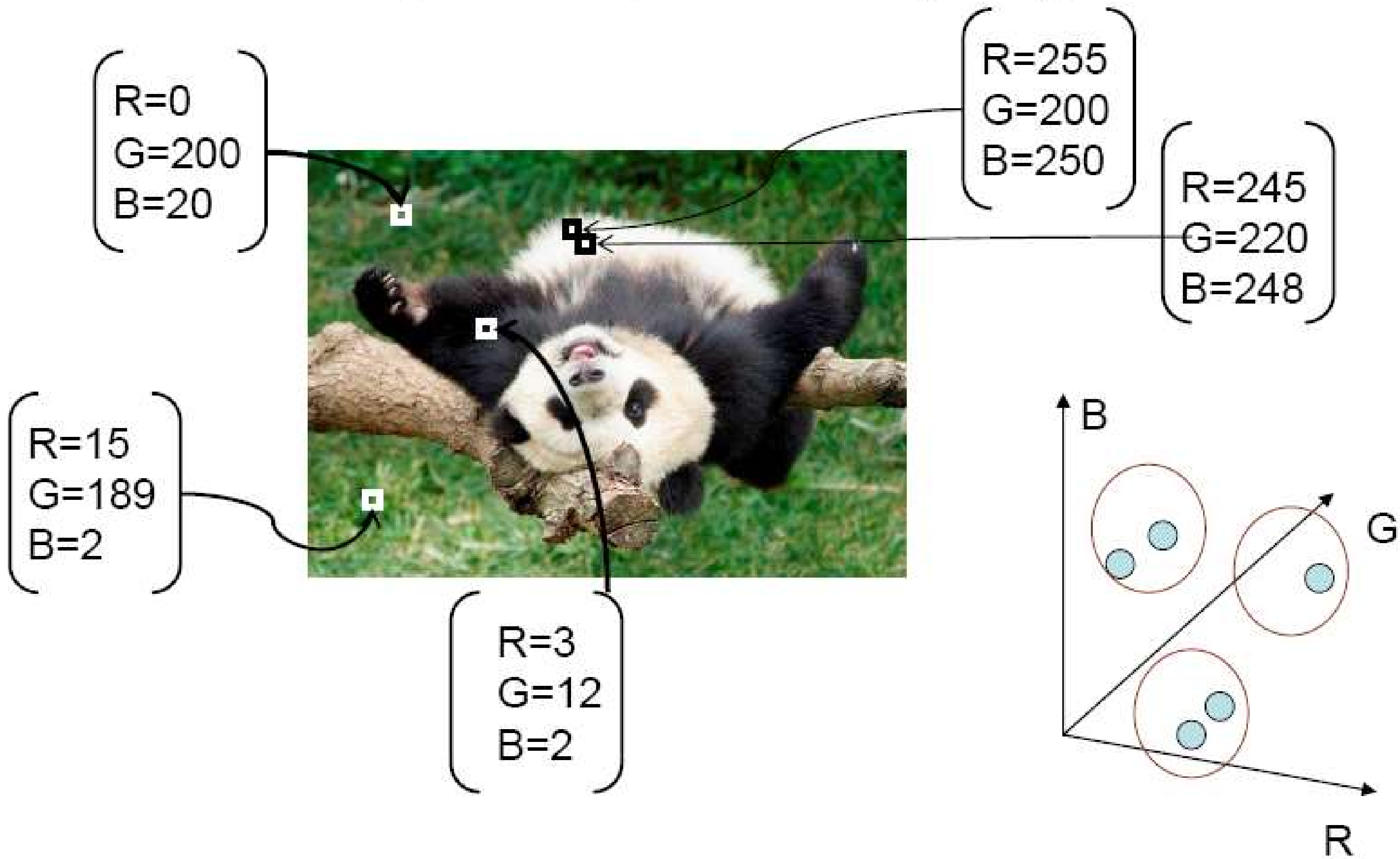
local thresholding



ce: K. Grauman

Segmentation as clustering

- Cluster similar pixels (features) together



Source: K. Grauman

Hierarchical Clustering Approach

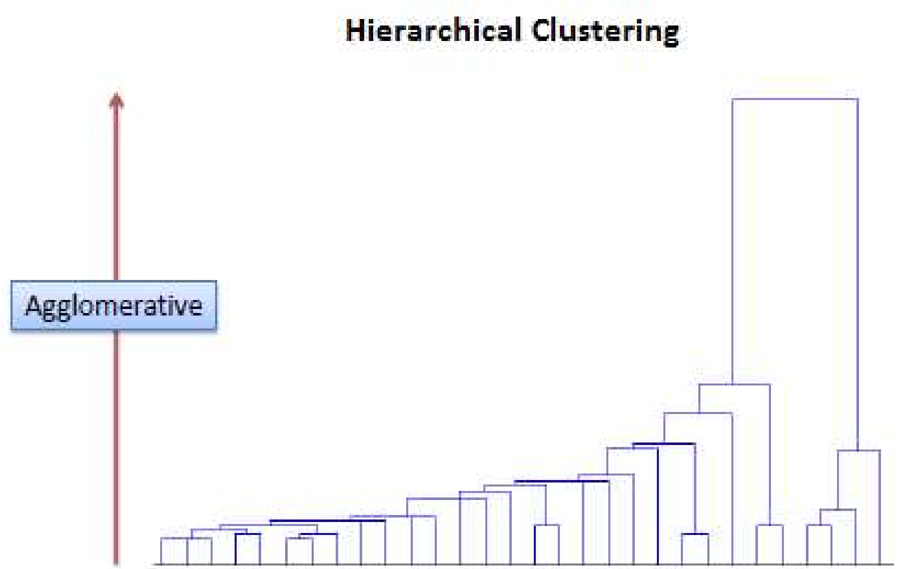
- A typical clustering analysis approach via partitioning data set sequentially
- Construct nested partitions layer by layer via grouping objects into a tree of clusters (without the need to know the number of clusters in advance)
- Use (generalised) distance matrix as clustering criteria

• Agglomerative vs. Divisive

- Two sequential clustering strategies for constructing a tree of clusters

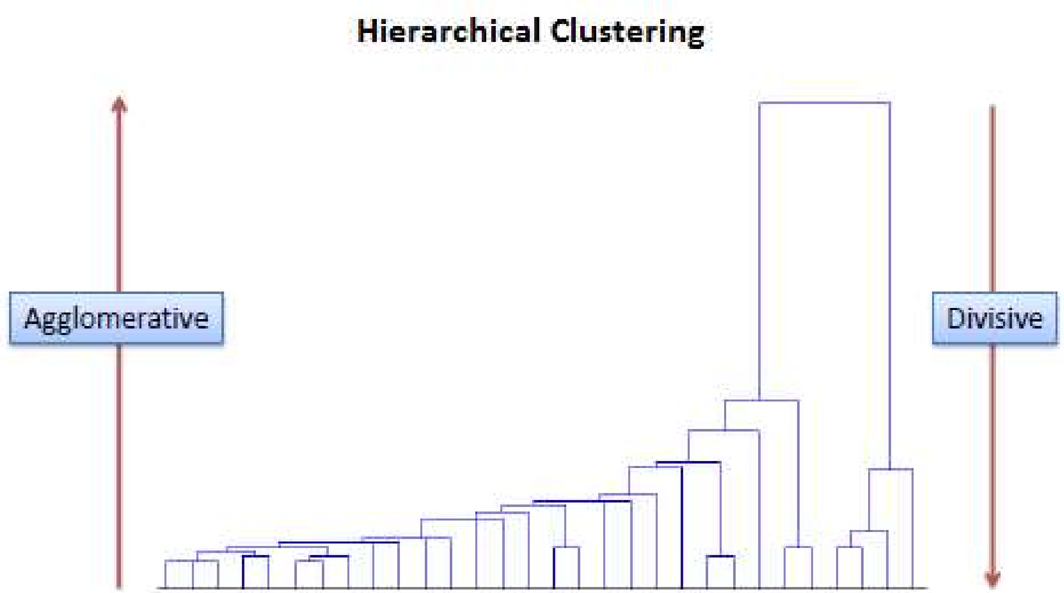
Agglomerative clustering

- Bottom-up strategy
  - Assume that each cluster is single pixel
    (i.e. every pixel is a cluster itself).
  - Merge Clusters i.e. attach closest to cluster it is closest to
    (if possible)
  - Repeat until no more clusters can be merged.



Divisive clustering

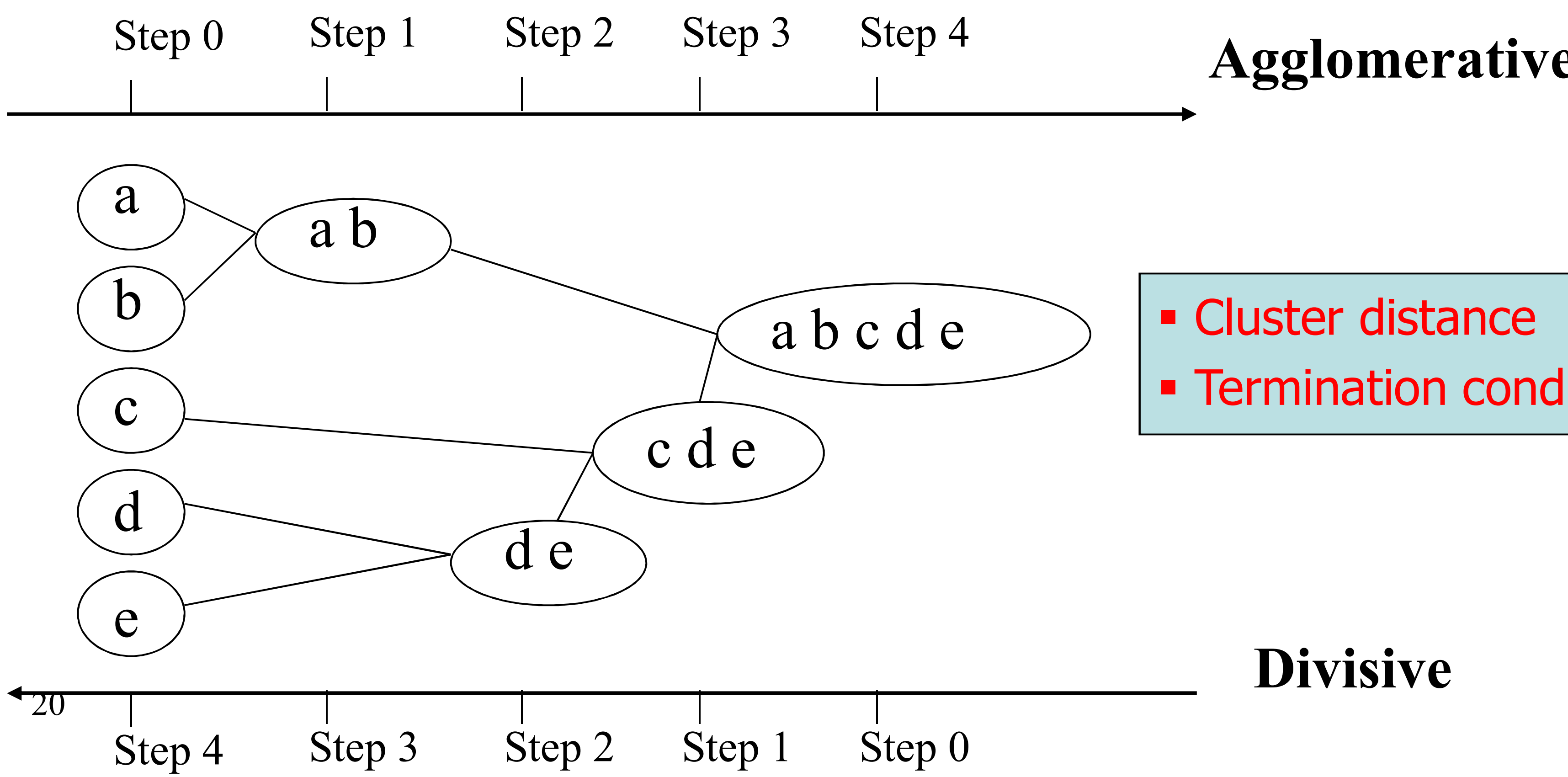
- Top-down strategy
  - Assume that whole image is a single cluster.
  - Split Clusters along best boundary (if exists)
  - Repeat previous step until no more clusters can be split.



Divisive clustering

- Illustrative Example

  Agglomerative and divisive clustering on the data set
  {a, b, c, d ,e }



- Cluster distance
- Termination condition

Segmentation as clustering

- Clustering based on intensity or color is essentially vector quantization of the image attributes
  - Clusters don’t have to be spatially coherent

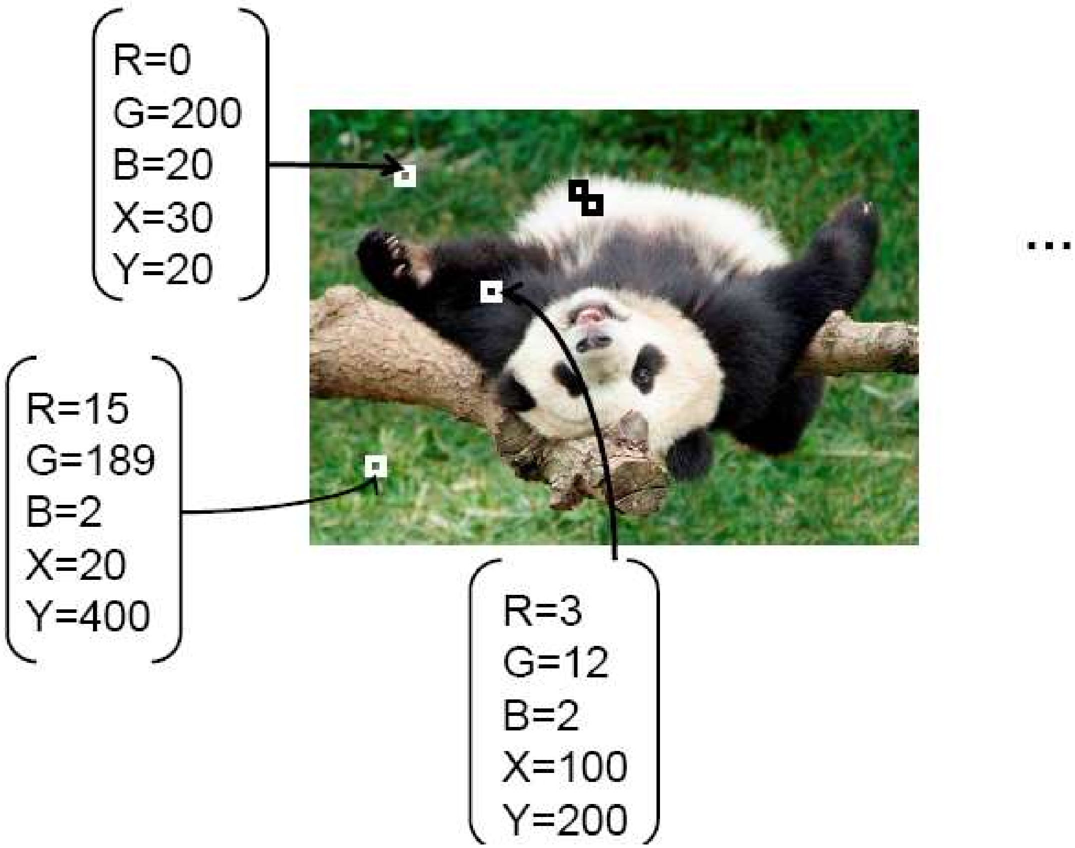
Image

Intensity-based clusters

Color-based clusters

Segmentation as clustering

- Use the position of pixels to perform clustering → spatial+color based segmentation
- Cluster similar pixels (features) together



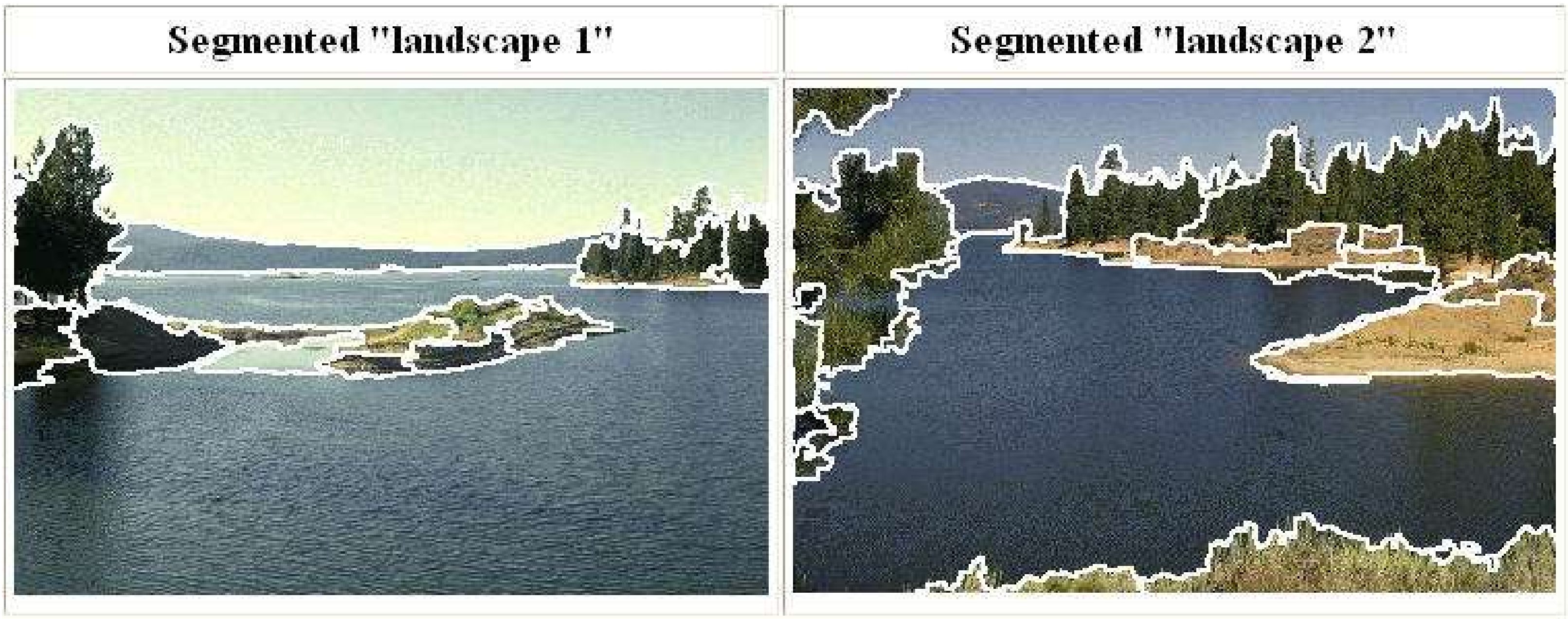
Source: K. Grauman

Segmentation as clustering

- With/without spatial information.

Mean shift segmentation

An advanced and versatile technique for clustering-based segmentation



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

Mean shift segmentation

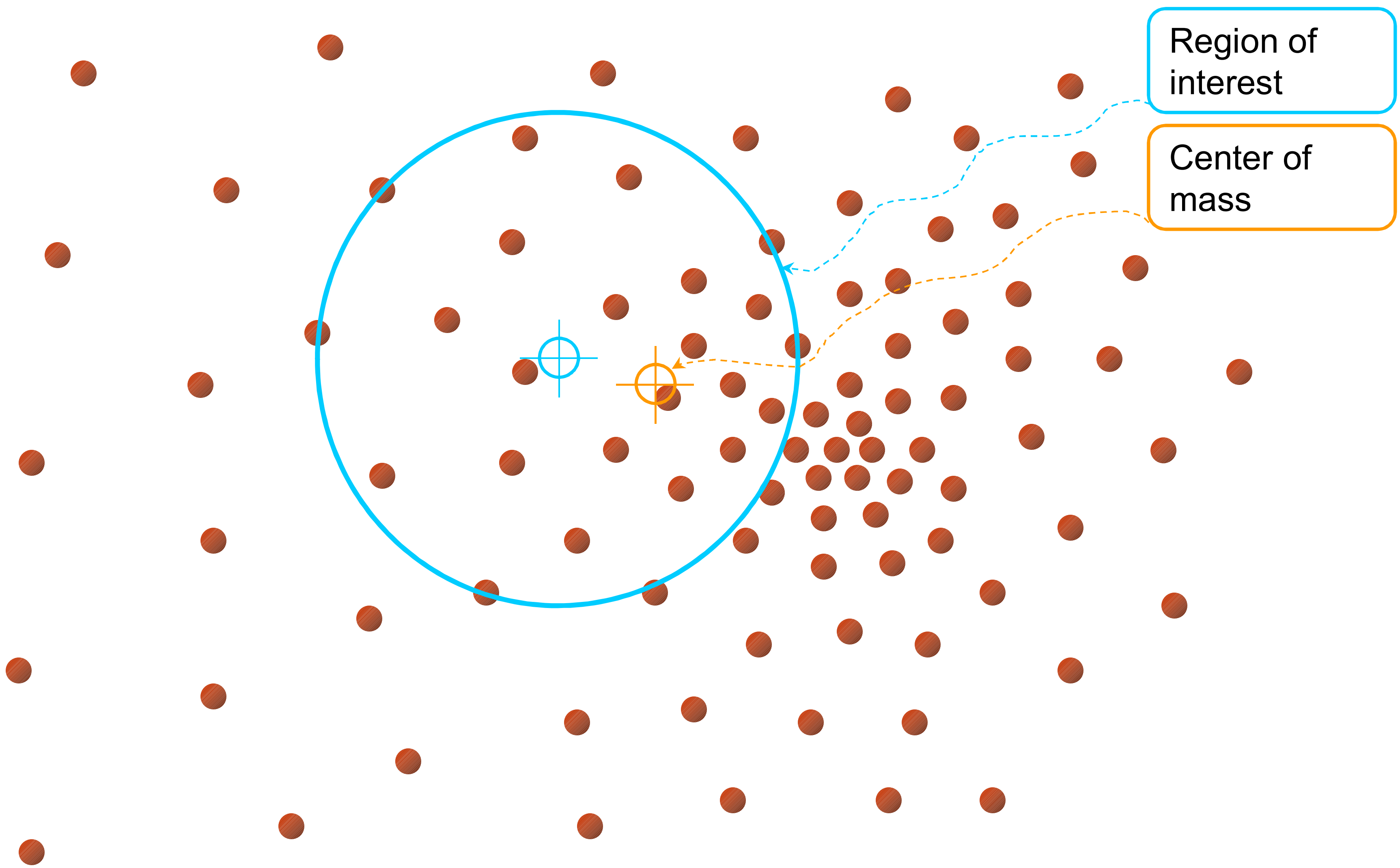
Mean Shift Algorithm

1. Choose a search window size.
2. Choose the initial location of the search window.
3. Compute the mean location (centroid of the data) in the search window.
4. Center the search window at the mean location computed in Step 3.
5. Repeat Steps 3 and 4 until convergence.

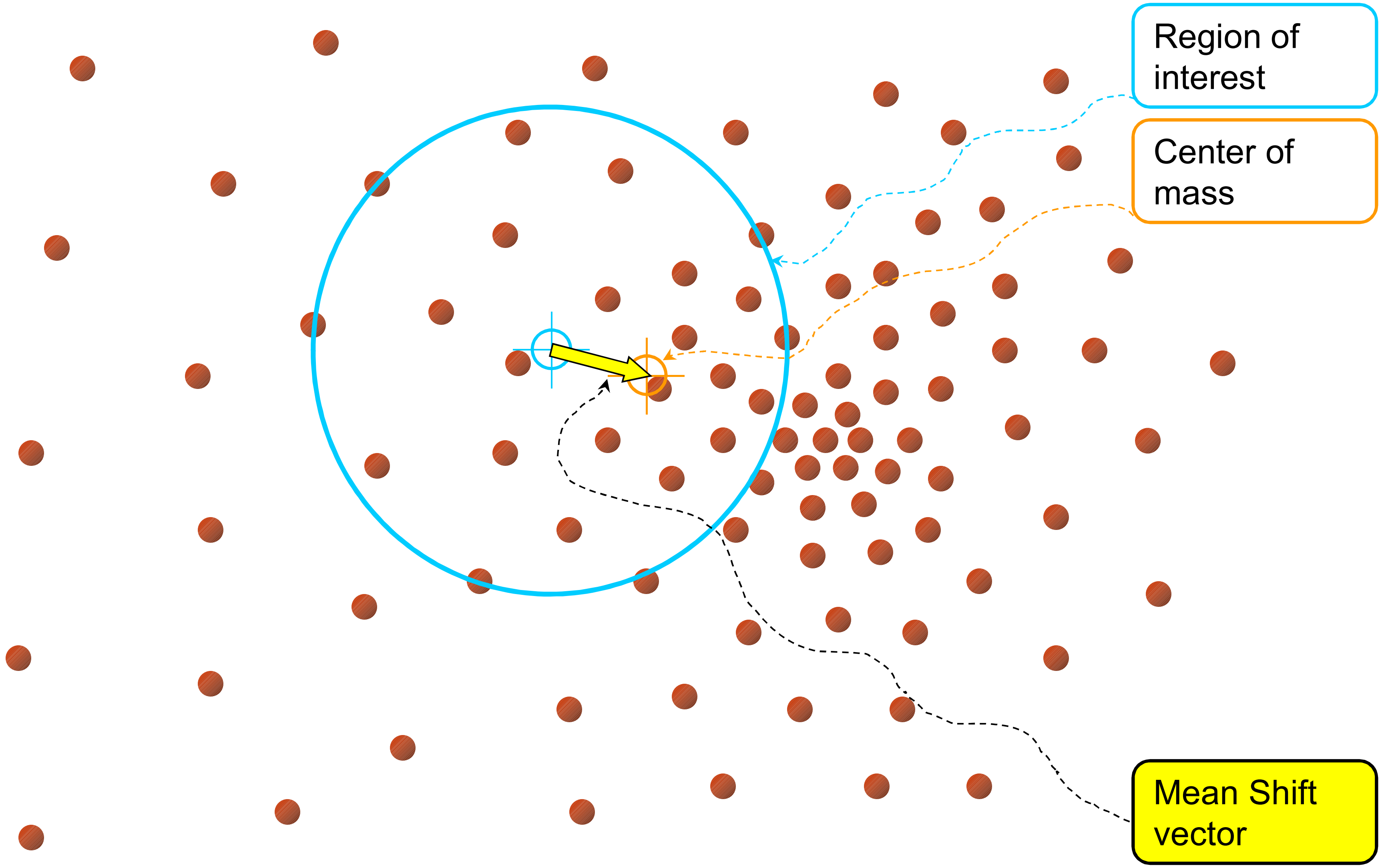
The mean shift algorithm seeks the "mode" or point of highest density of a data distribution:

Mean shift

The mean shift algorithm seeks the "mode" or point of highest density of a data distribution:

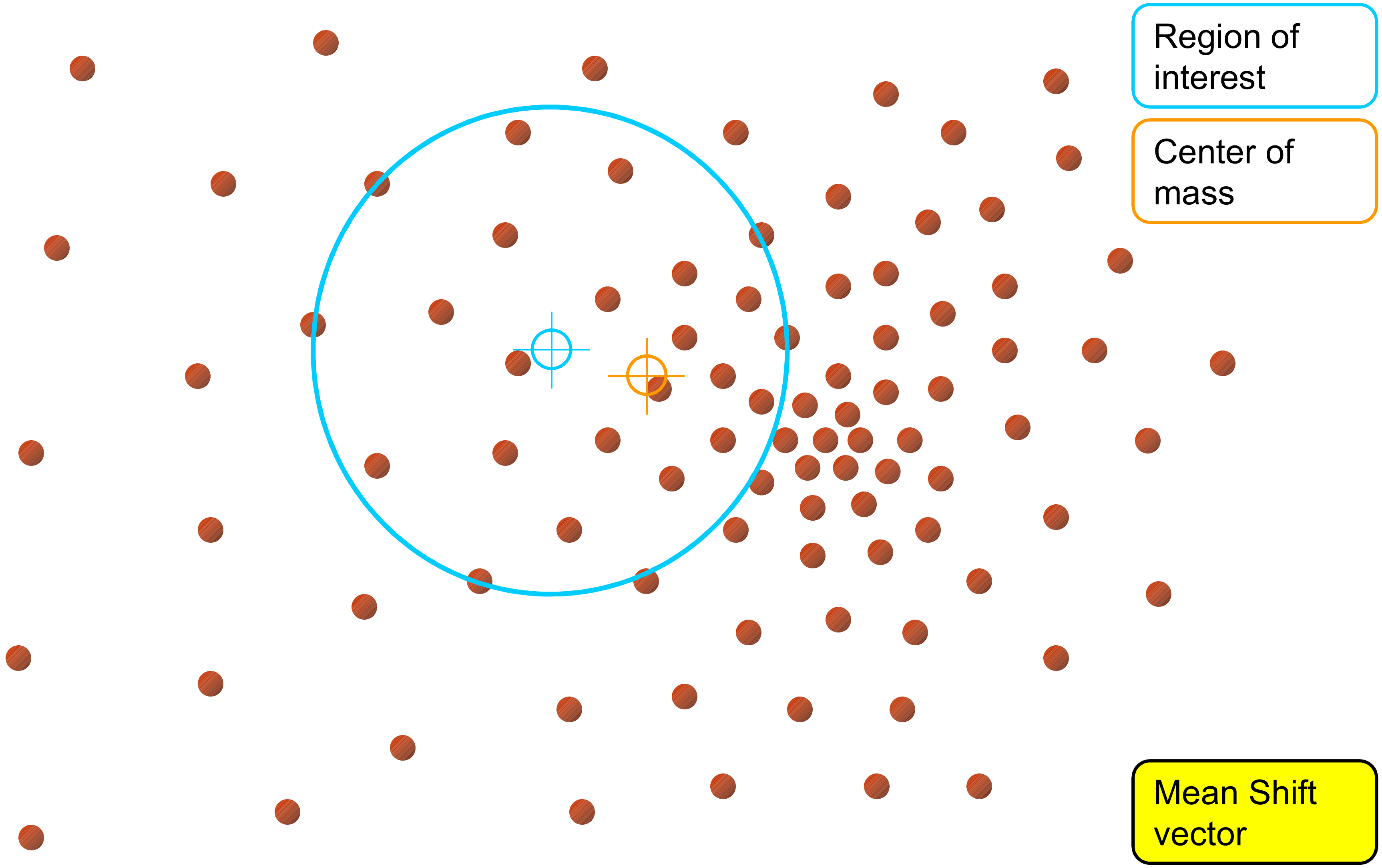


Mean shift



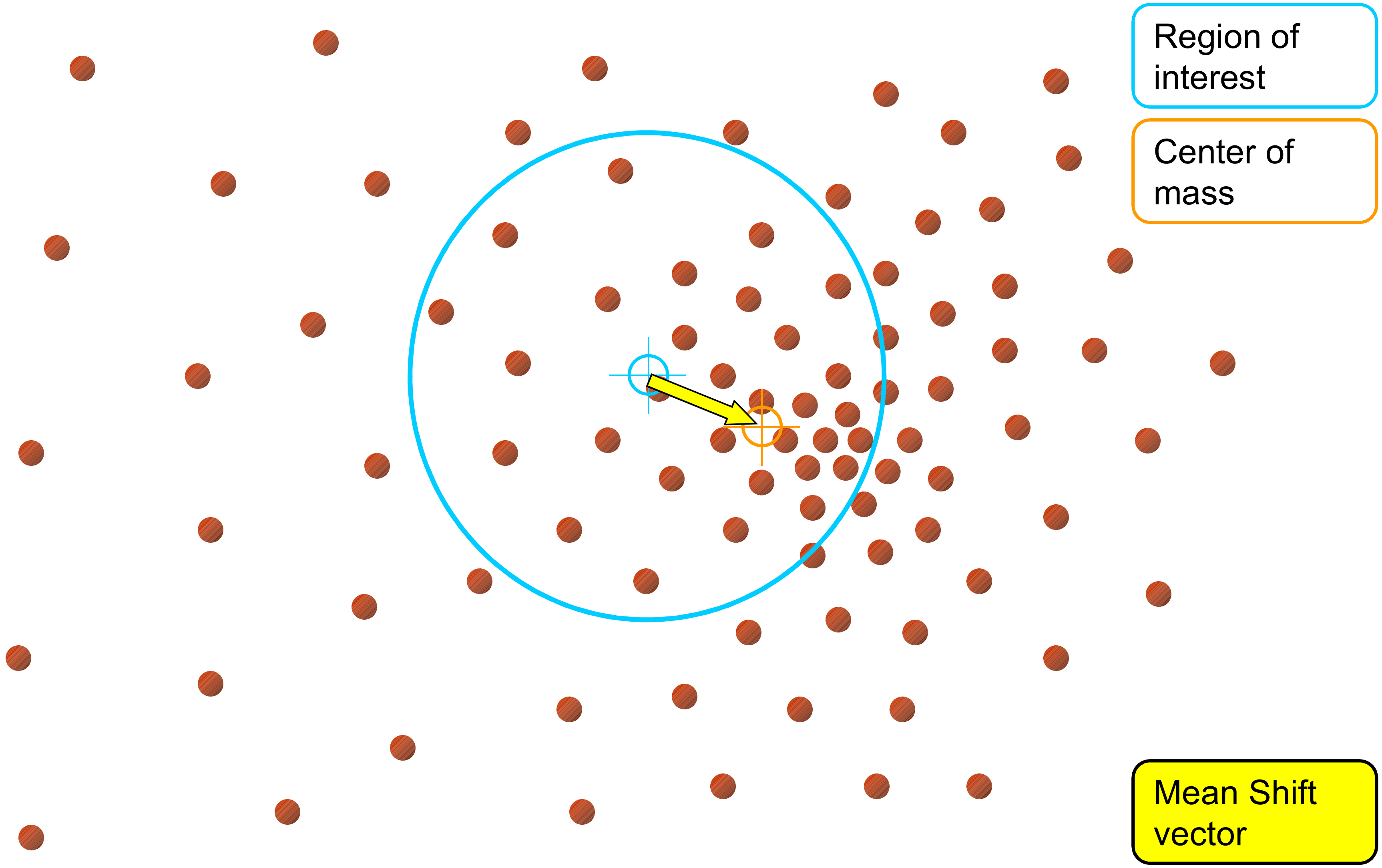
Slide by Y. Ukrainitz & B. Sarel

Mean shift



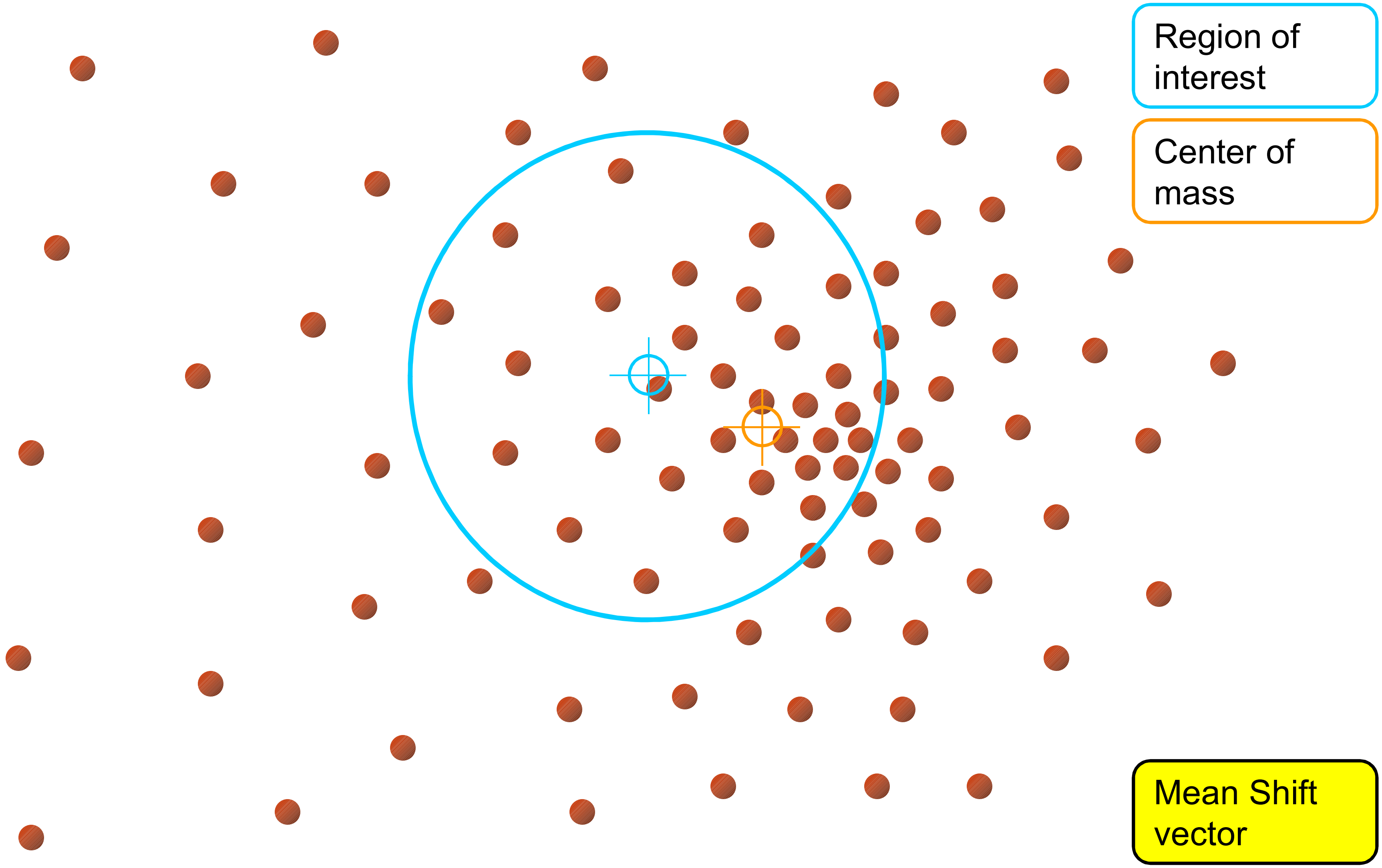
Slide by Y. Ukrainitz & B. Sarel

Mean shift



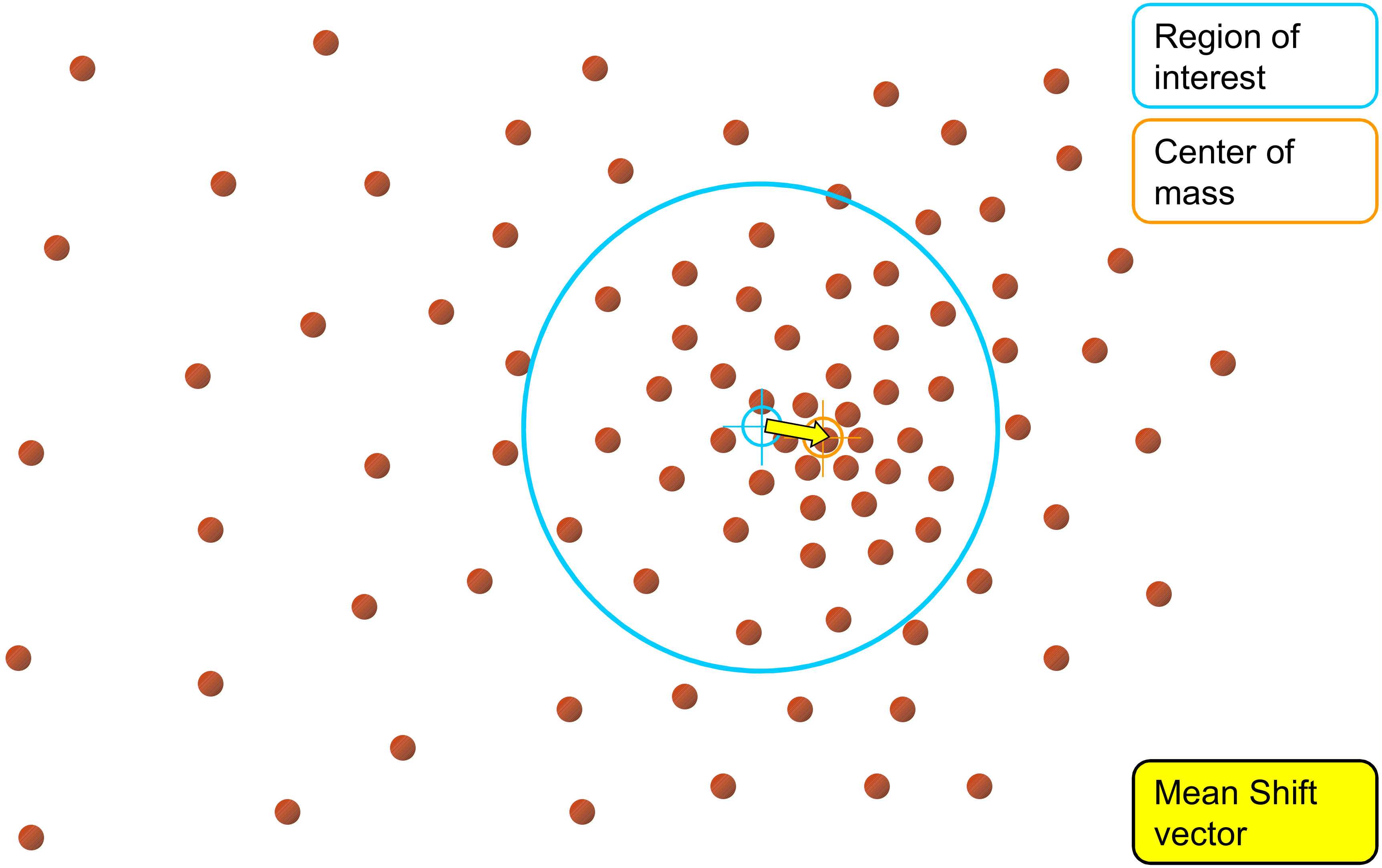
Slide by Y. Ukrainitz & B. Sarel

Mean shift



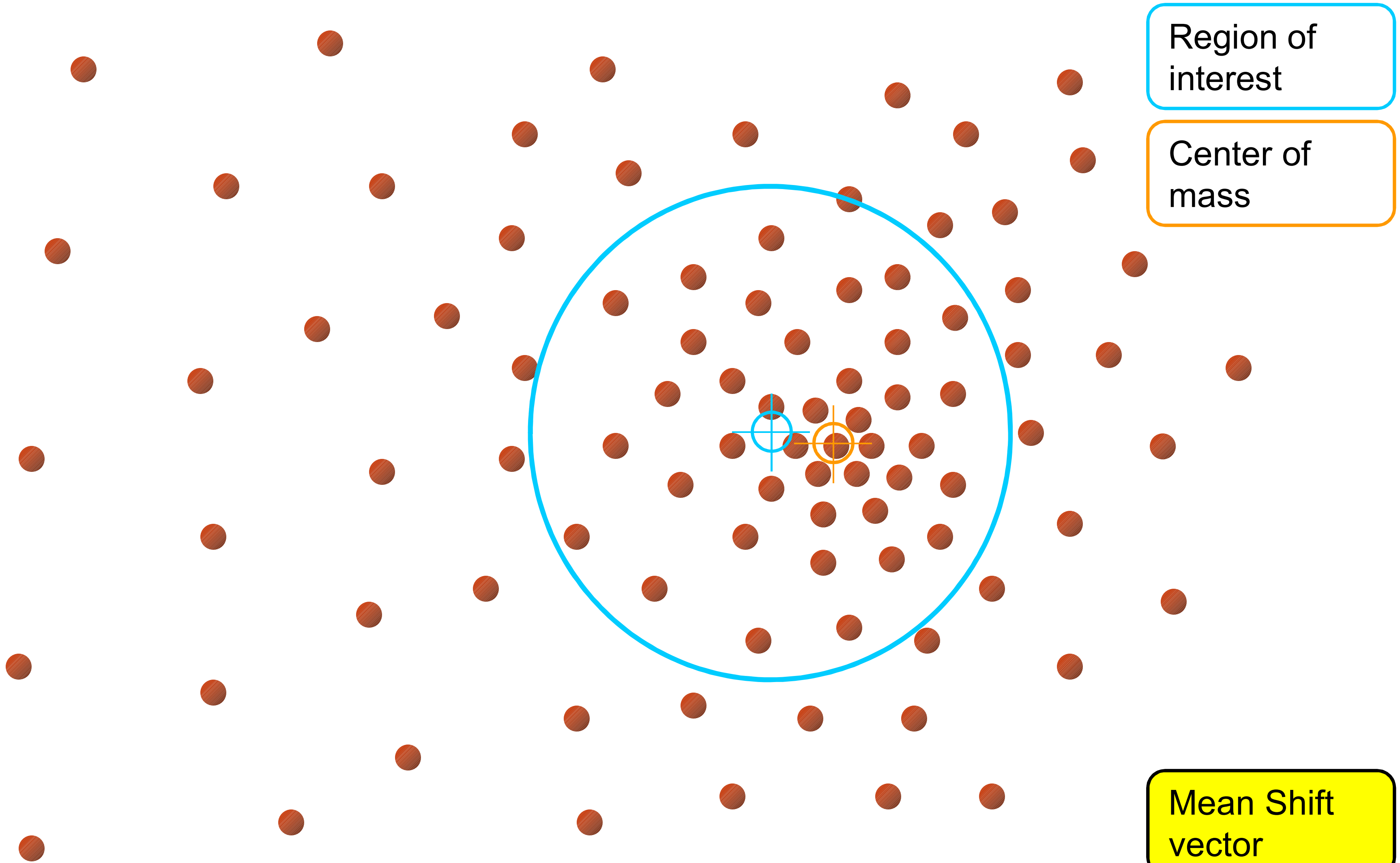
Slide by Y. Ukrainitz & B. Sarel

Mean shift



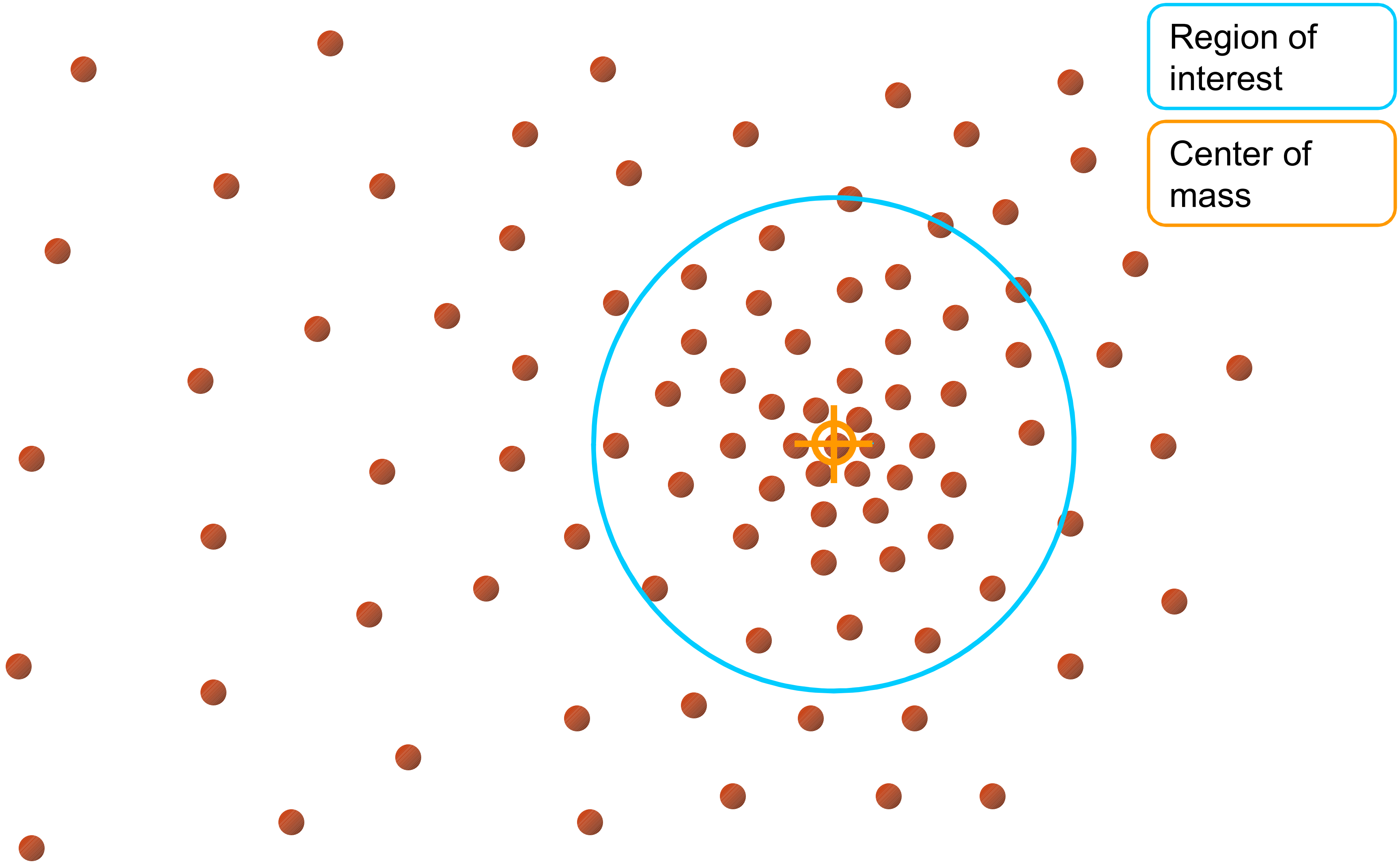
Slide by Y. Ukrainitz & B. Sarel

Mean shift



Slide by Y. Ukrainitz & B. Sarel

Mean shift

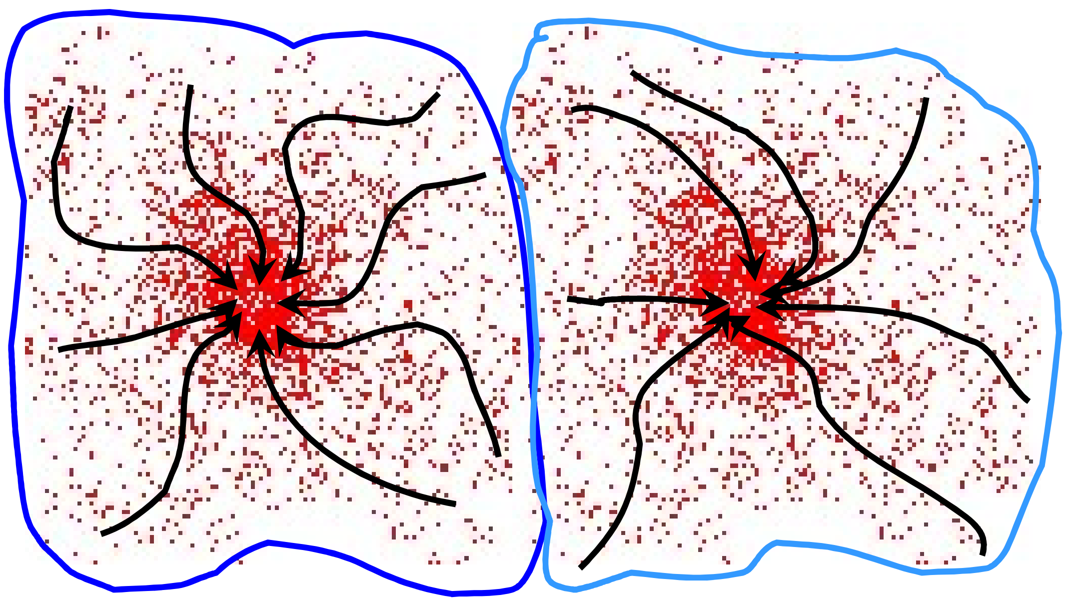


Slide by Y. Ukrainitz & B. Sarel

Mean shift clustering

Attraction basin: the region for which all trajectories lead to the same mode

Cluster: all data points in the attraction basin of a mode

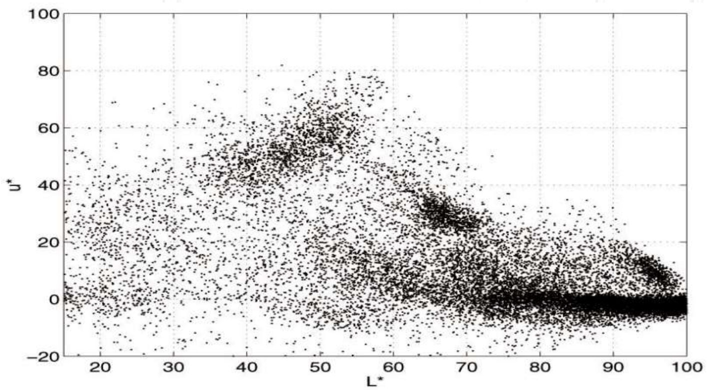


Slide by Y. Ukrainitz & B. Sarel

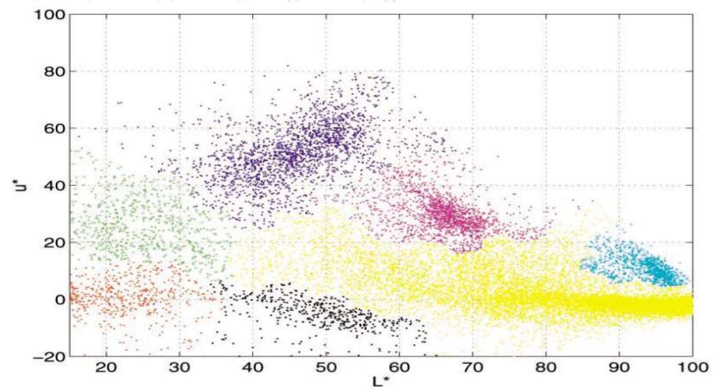
Mean Shift Segmentation

Mean Shift Setmentation Algorithm

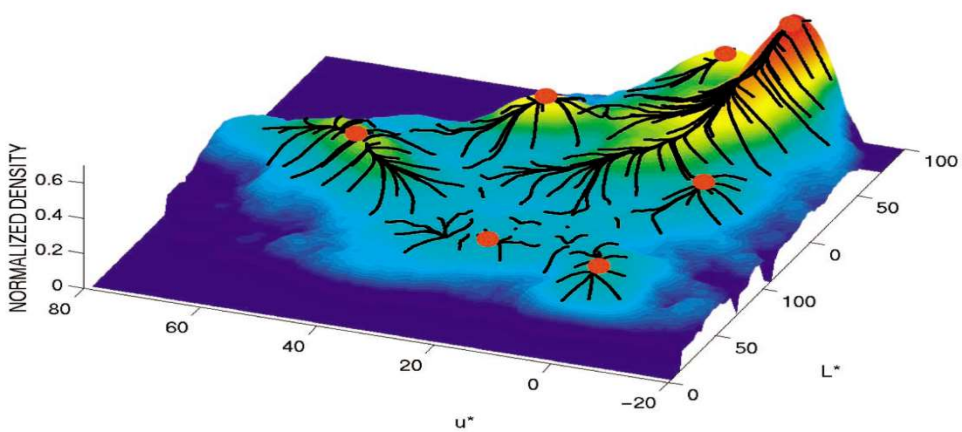
1. Convert the image into tokens (via color, gradients, texture measures etc).
2. Choose initial search window locations uniformly in the data.
3. Compute the mean shift window location for each initial position.
4. Merge windows that end up on the same "peak" or mode.
5. The data these merged windows traversed are clustered together.



(a)



(b)



(c)

Mean shift segmentation results




http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html

Mean shift pros and cons

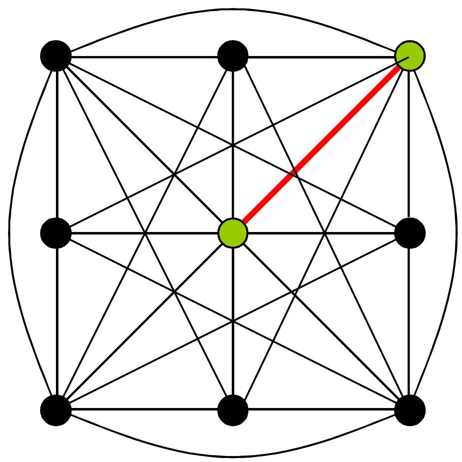
Pros

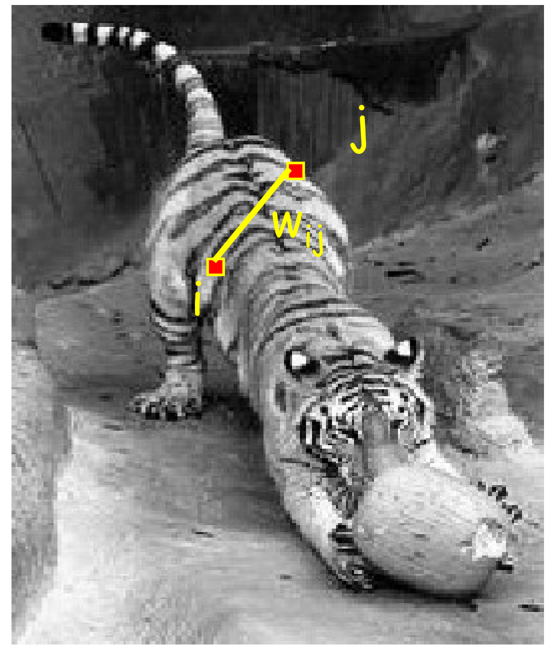
- Does not assume spherical clusters
- Just a single parameter (window size)
- Finds variable number of modes
- Robust to outliers

Cons

- Output depends on window size
- Computationally expensive
- Does not scale well with dimension of feature space

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

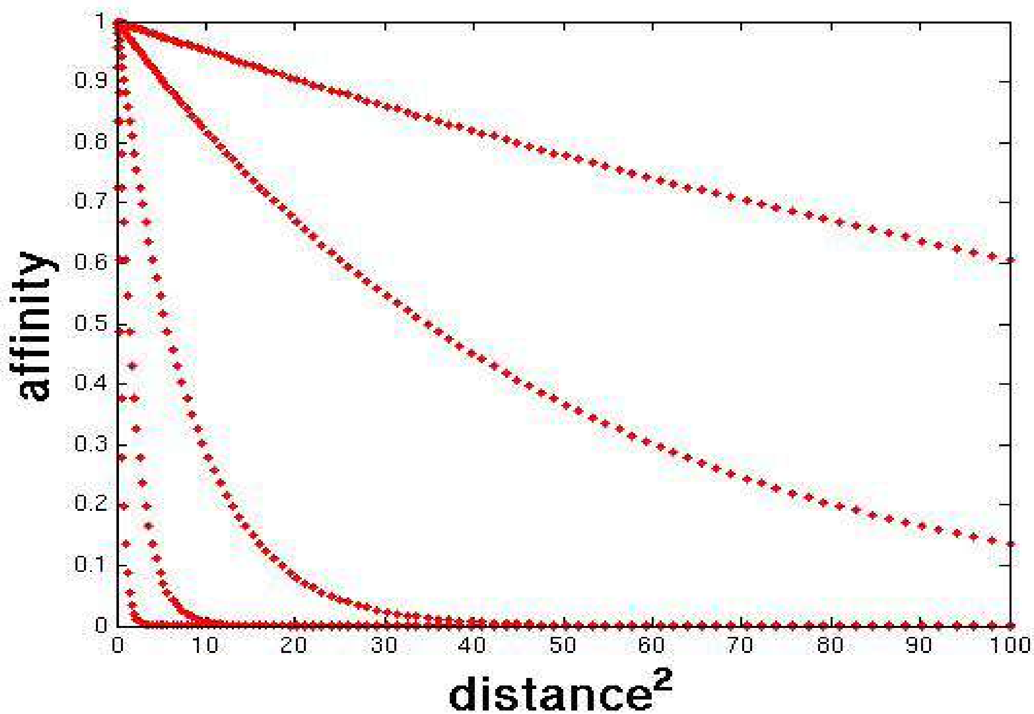
Source: S. Seitz

Measuring affinity

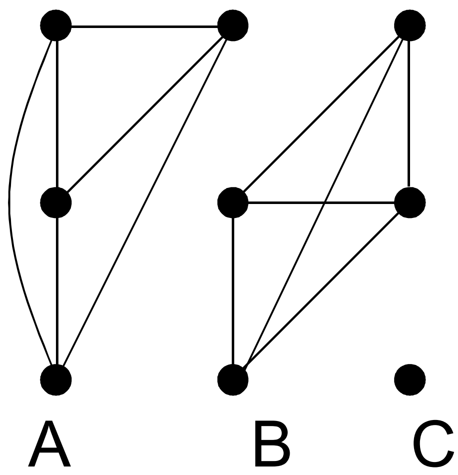
- Represent each pixel by a feature vector **x** and define an appropriate distance function

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

Role of σ



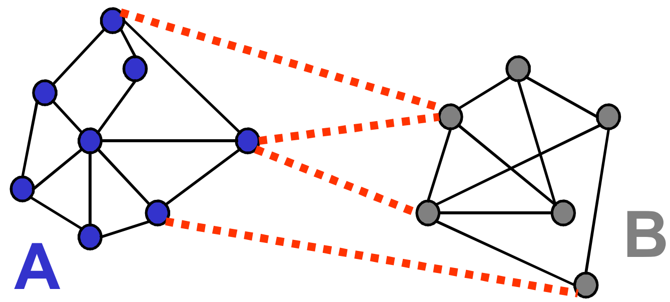
Segmentation as graph partitioning



- 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

Graph cut



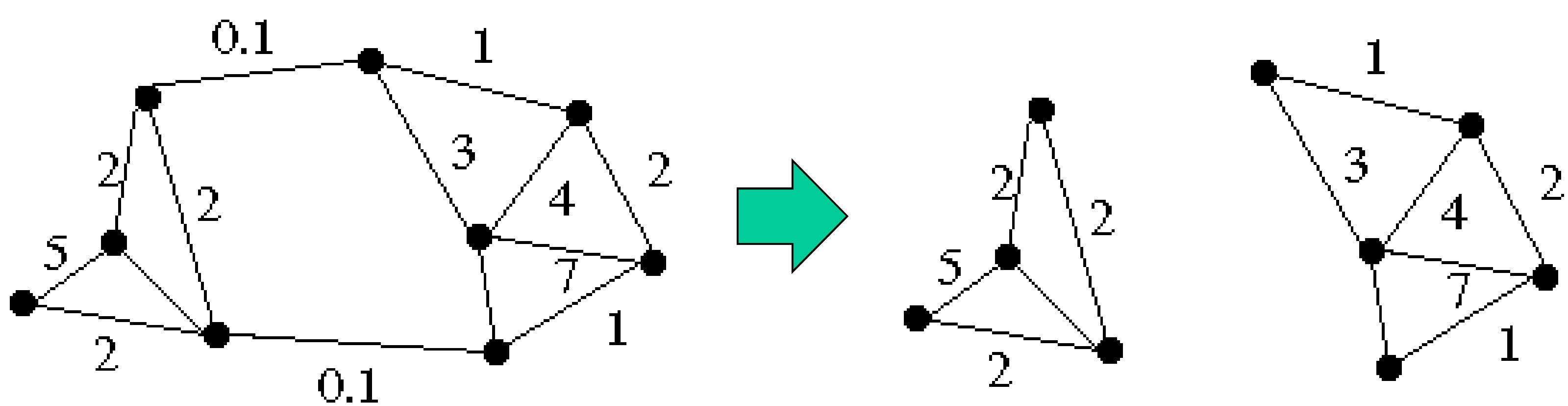
- Set of edges whose removal makes a graph disconnected
- Cost of a cut: sum of weights of cut edges
- A graph cut gives us a segmentation

Source: S. Seitz

Minimum cut

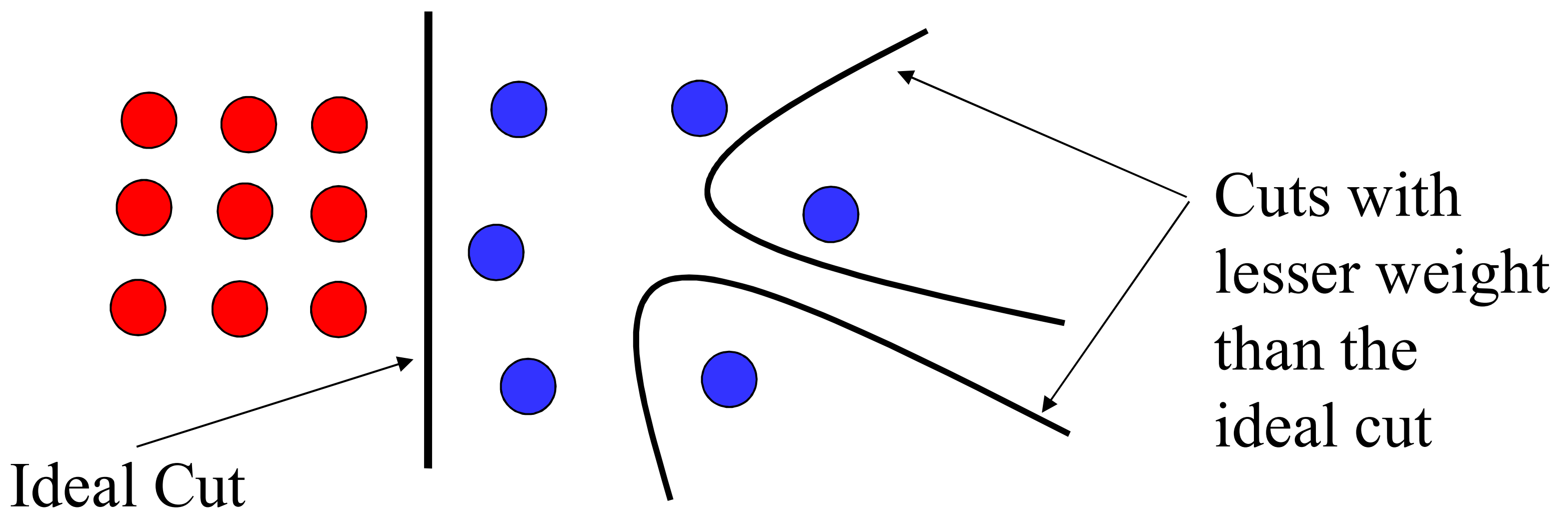
- We can do segmentation by finding the *minimum cut* in a graph

Minimum cut example



Normalized cut

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



* Slide from Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Normalized cut

- To encourage larger segments, normalize the cut by the total weight of edges incident to the segment.

- The *normalized cut* cost is:

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

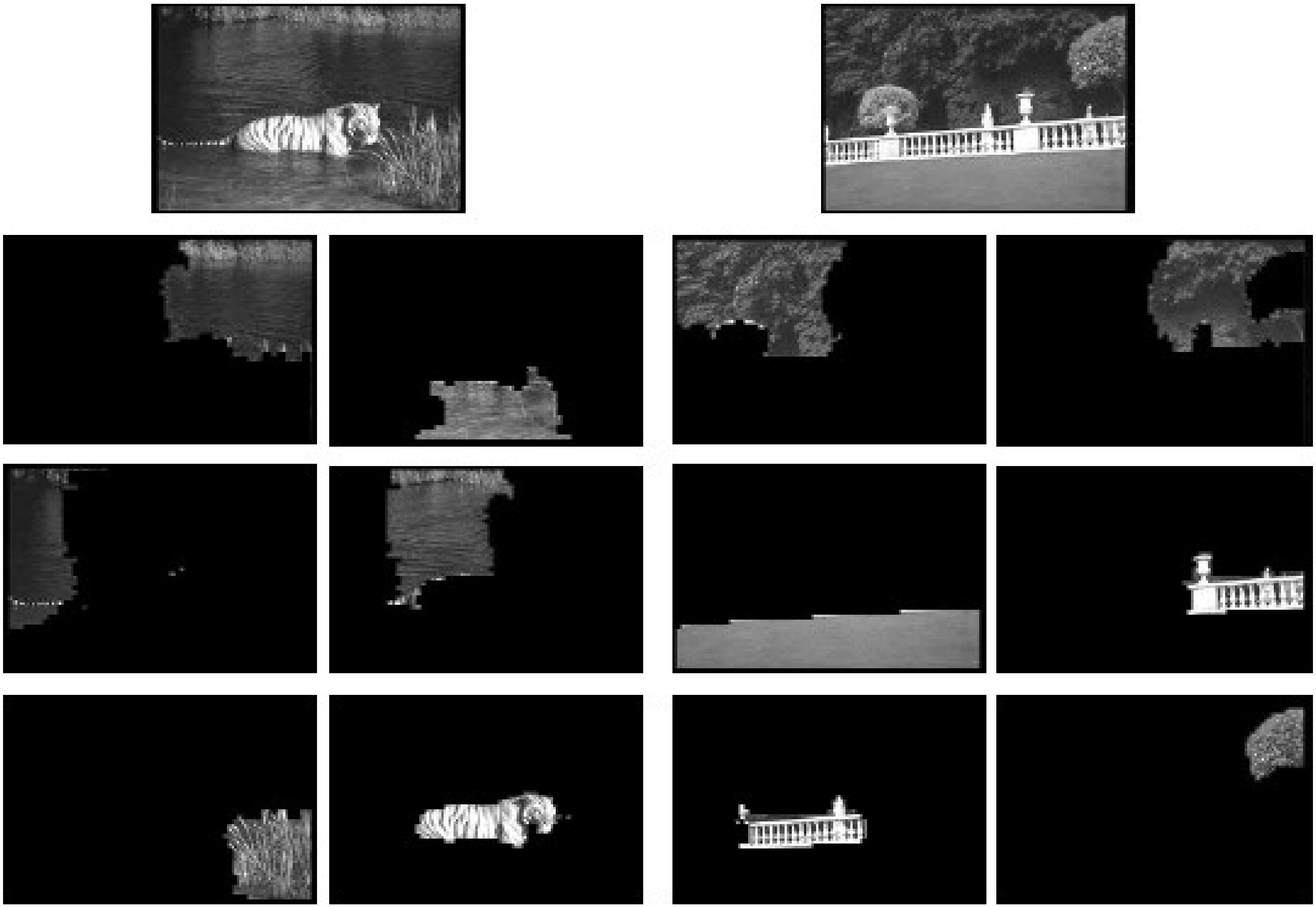
w(*A*, *B*) = sum of weights of all edges between *A* and *B*

Intuition: big segments will have a large w(A,V), thus decreasing ncut(A, 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

Example result



Challenge

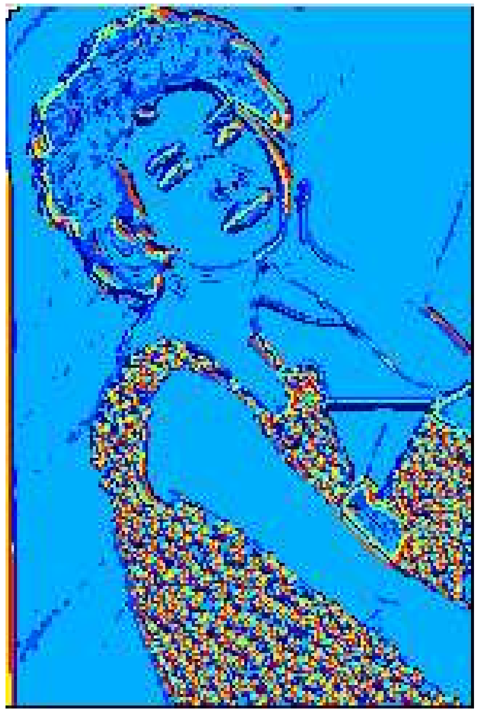
- How to define affinities for segmenting highly textured images?

Segmenting textured images

- Convolve image with a bank of filters
- Find *textons* by clustering vectors of filter bank outputs

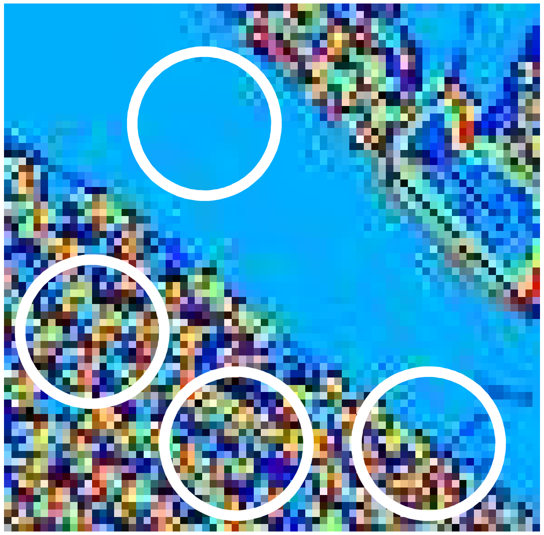
Image

Texton map



Segmenting textured images

- Convolve image with a bank of filters
- Find *textons* by clustering vectors of filter bank outputs
- Represent pixels by texton histograms computed over neighborhoods at some “local scale”
- Define affinities as similarities between local texton histograms



Results: Berkeley Segmentation Engine

http://www.cs.berkeley.edu/~fowlkes/BSE/

Berkeley Segmentation Engine

http://www.cs.berkeley.edu/~fowlkes/BSE/

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

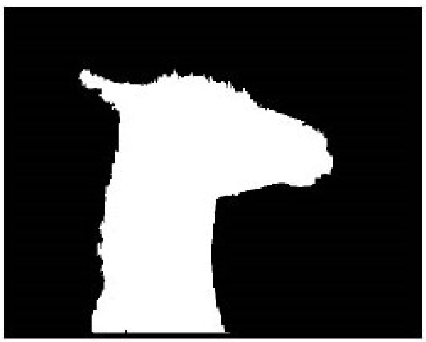
Energy minimization

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

smoothness term — Encourage spatially coherent segments



Source: N. Snavely

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

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

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

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

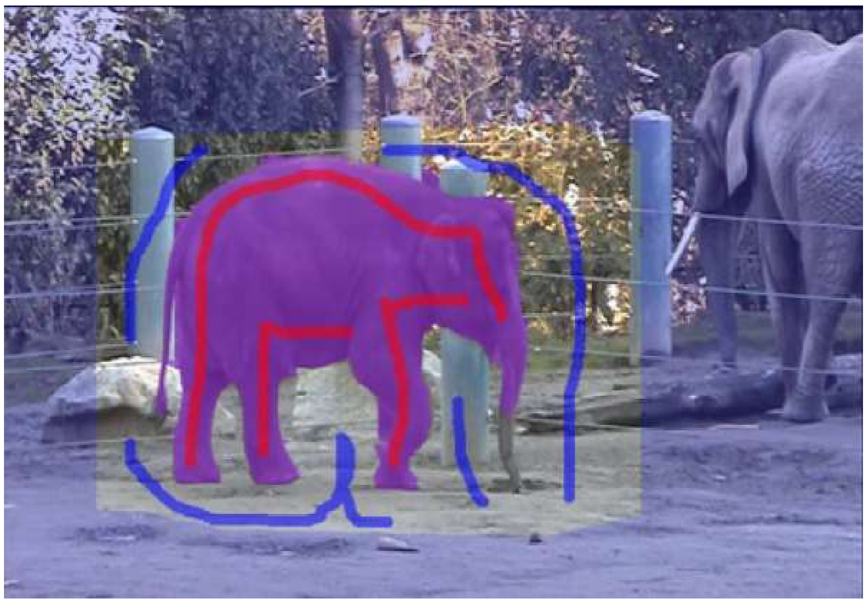
$C'(x, y, 0)$: "distance" from pixel to background

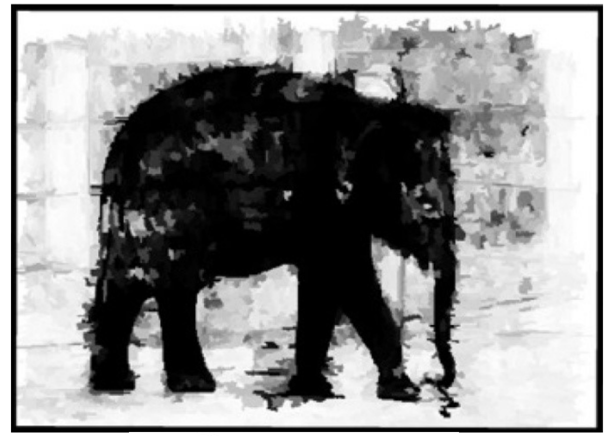
$C'(x, y, 1)$: "distance" from pixel to foreground

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

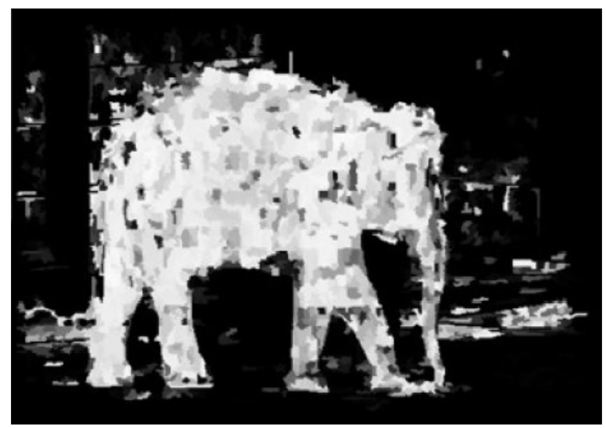
Source: N. Snavely

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





$C'(x, y, 0)$

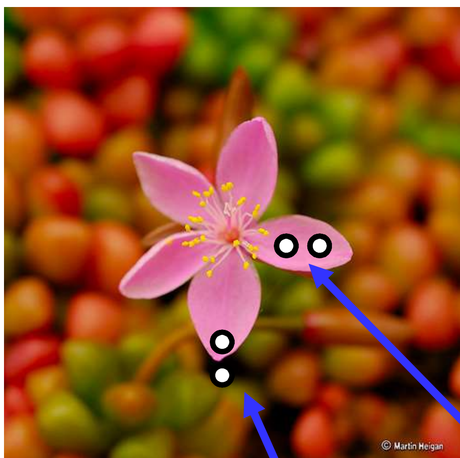


$C'(x, y, 1)$

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} = 0.1

w_{pq} = 10.0

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

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

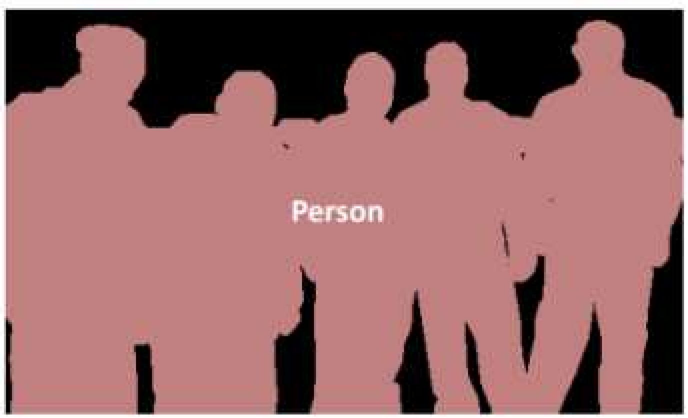
Y. Boykov and M.-P. Jolly, Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Images, ICCV 2001

Source: N. Snavely

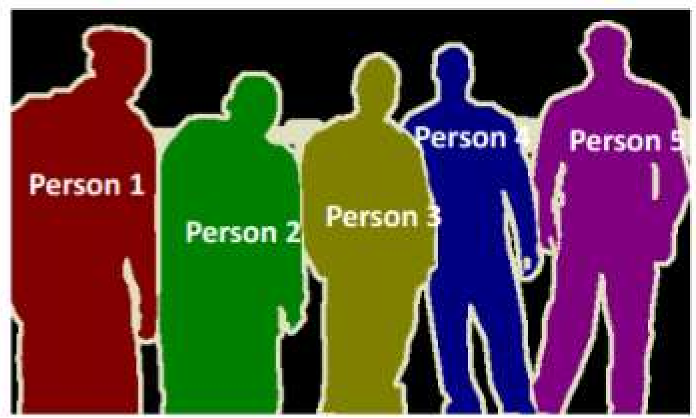
Instance Segmentation

Given an image produce instance-level segmentation

- Which class does each pixel belong to
- Also which instance



Semantic Segmentation



Instance Segmentation