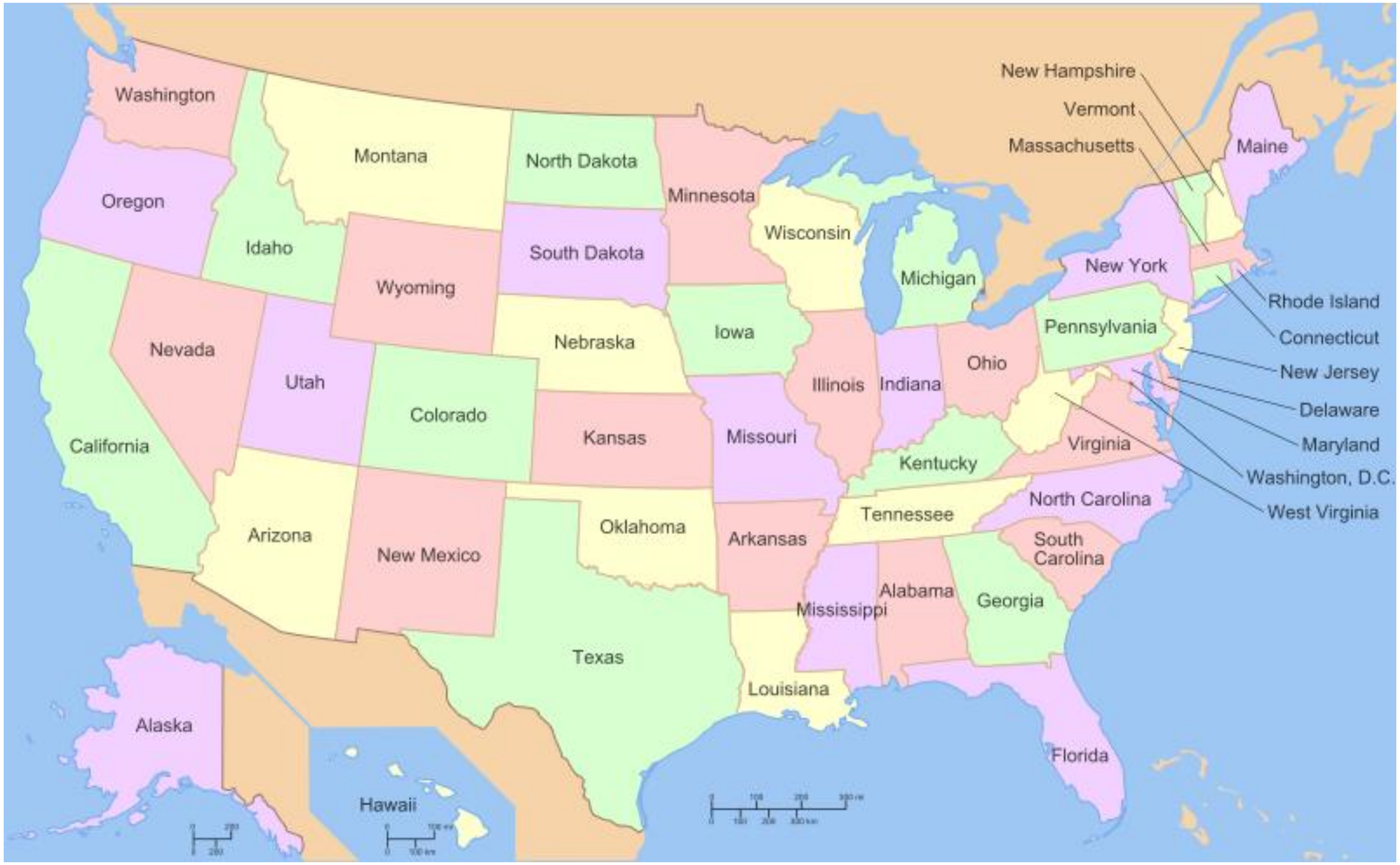
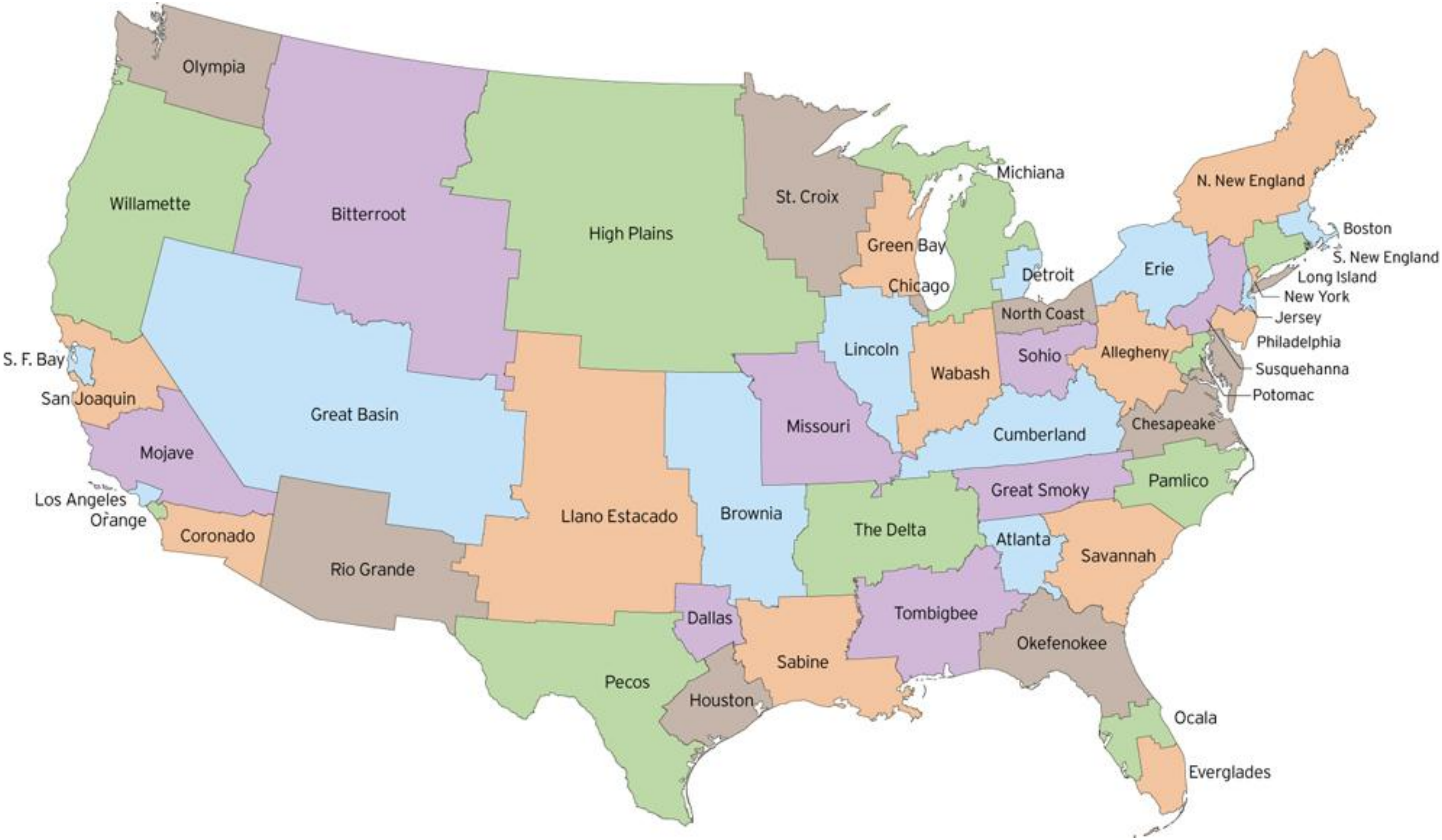


Washington
Montana
North Dakota
Minnesota
Oregon
Idaho
Wisconsin
South Dakota
Wyoming
Michigan
New York
New Hampshire
Vermont
Massachusetts
Maine
Rhode Island
Connecticut
New Jersey
Delaware
Maryland
Washington, D.C.
West Virginia
Pennsylvania
Nevada
Iowa
Nebraska
Ohio
Utah
Illinois
Indiana
Colorado
California
Kansas
Missouri
Kentucky
Virginia
North Carolina
Tennessee
Oklahoma
Arizona
New Mexico
Arkansas
South Carolina
Alabama
Mississippi
Georgia
Texas
Louisiana
Alaska
Florida
Hawaii



- http://fakeisthenewreal.org/reform/

Machine Learning: Clustering

Computer Vision

CS 143, Brown

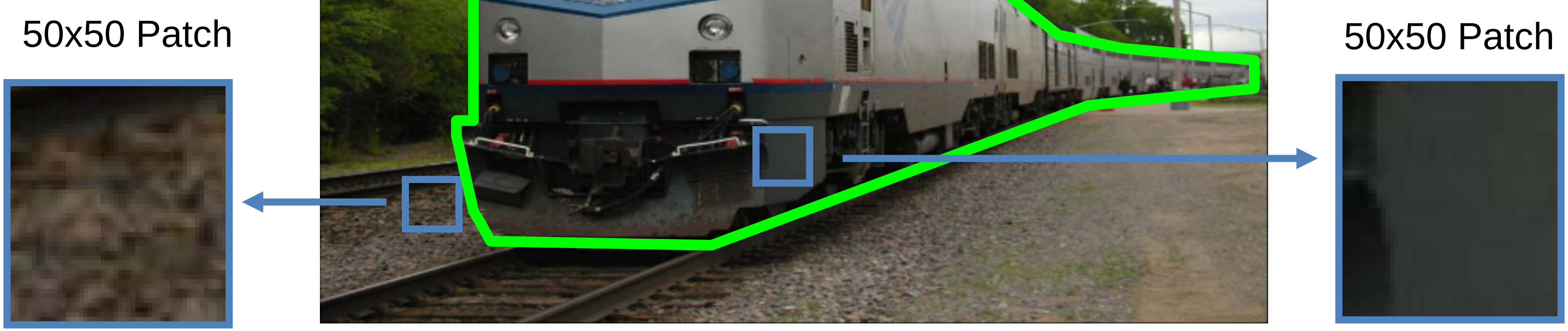
James Hays

Slides: Hoiem and others

Clustering example: image segmentation

Goal: Break up the image into meaningful or perceptually similar regions

Segmentation for feature support



Segmentation for efficiency

[Felzenszwalb and Huttenlocher 2004]

[Shi and Malik 2001]

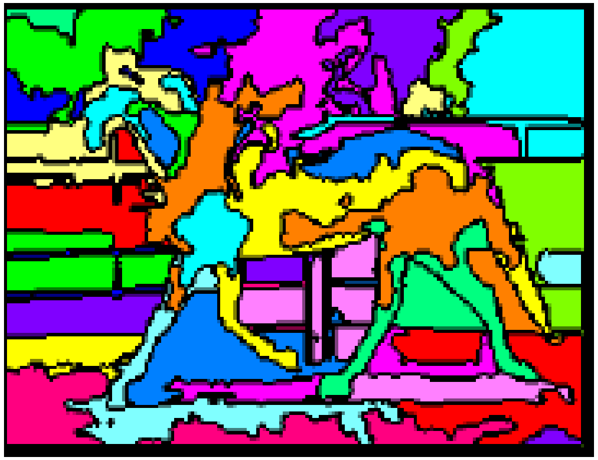
[Hoiem et al. 2005, Mori 2005]

Slide: Derek Hoiem

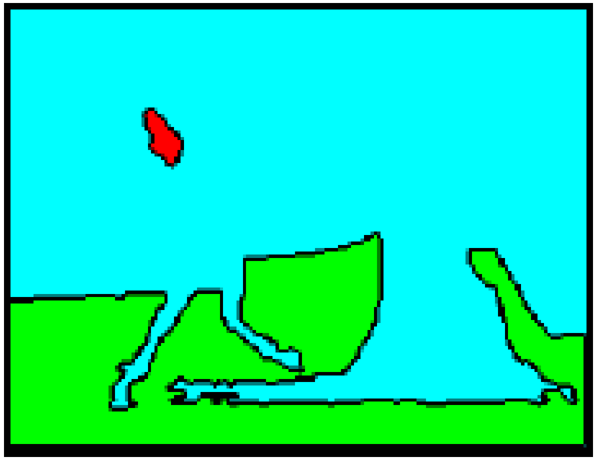
Segmentation as a result

Rother et al. 2004

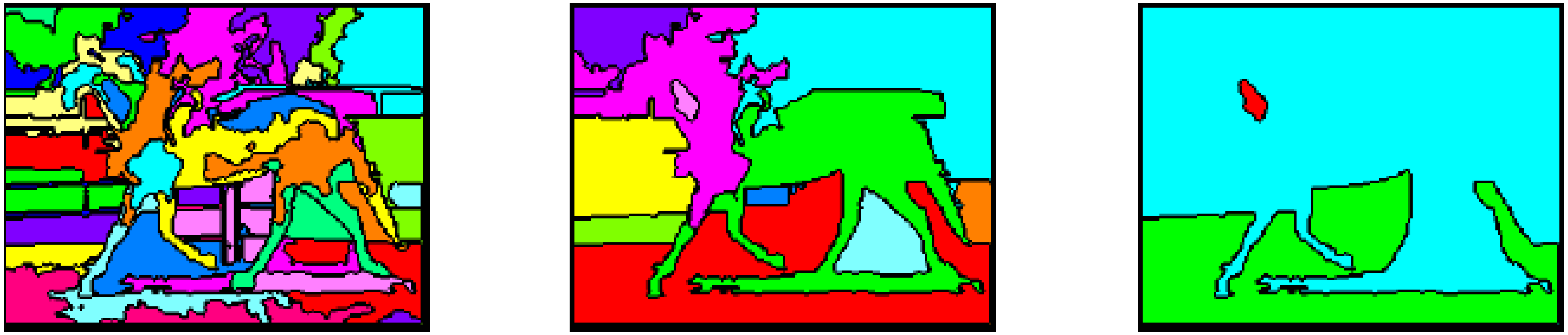
Types of segmentations



Oversegmentation



Undersegmentation



Multiple Segmentations

Clustering: group 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?

Slide: Derek Hoiem

Why do we cluster?

- **Summarizing data**
  - Look at large amounts of data
  - Patch-based compression or denoising
  - Represent a large continuous vector with the cluster number
- **Counting**
  - Histograms of texture, color, SIFT vectors
- **Segmentation**
  - Separate the image into different regions
- **Prediction**
  - Images in the same cluster may have the same labels

Slide: Derek Hoiem

How do we cluster?

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

Clustering for Summarization

Goal: cluster to minimize variance in data given clusters

– Preserve information

Cluster center

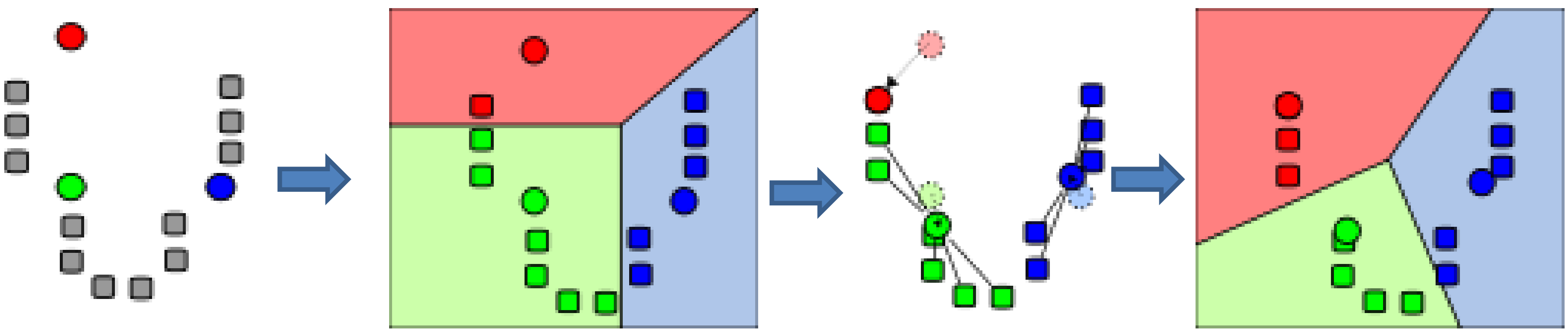
Data

$$\mathbf{c}^*, \boldsymbol{\delta}^* = \underset{\mathbf{c},\boldsymbol{\delta}}{\operatorname{argmin}} \frac{1}{N} \sum_{j}^{N} \sum_{i}^{K} \delta_{ij} \left(\mathbf{c}_i - \mathbf{x}_j\right)^2$$

Whether x_j is assigned to c_i

Slide: Derek Hoiem

K-means



0. Initialize Cluster Centers

1. Assign Points to Clusters

2. Re-compute Means

Repeat (1) and (2)

Illustration Source: wikipedia

K-means

1. Initialize cluster centers: $\mathbf{c}^0$; t=0

2. Assign each point to the closest center

$$\boldsymbol{\delta}^t = \underset{\boldsymbol{\delta}}{\operatorname{argmin}} \frac{1}{N} \sum_{j}^{N} \sum_{i}^{K} \delta_{ij} \left(\mathbf{c}_i^{t-1} - \mathbf{x}_j \right)^2$$

3. Update cluster centers as the mean of the points

$$\mathbf{c}^t = \underset{\mathbf{c}}{\operatorname{argmin}} \frac{1}{N} \sum_{j}^{N} \sum_{i}^{K} \delta_{ij}^t \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

4. Repeat 2-3 until no points are re-assigned (t=t+1)

Slide: Derek Hoiem

K-means: design choices

- Initialization
  – Randomly select K points as initial cluster center
  – Or greedily choose K points to minimize residual
- Distance measures
  – Traditionally Euclidean, could be others
- Optimization
  – Will converge to a *local minimum*
  – May want to perform multiple restarts

How to evaluate clusters?

- Generative
  - How well are points reconstructed from the clusters?
- Discriminative
  - How well do the clusters correspond to labels?
    - Purity
  - Note: unsupervised clustering does not aim to be discriminative

Slide: Derek Hoiem

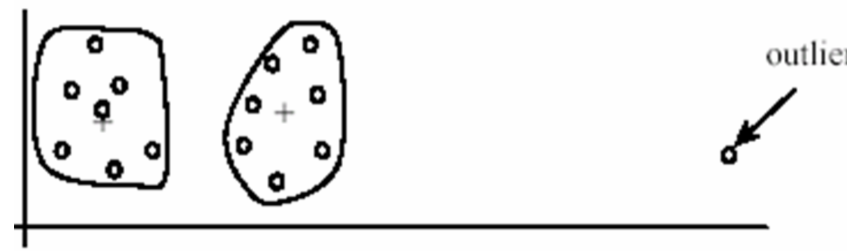
How to choose the number of clusters?

- Validation set
  - Try different numbers of clusters and look at performance
    - When building dictionaries (discussed later), more clusters typically work better

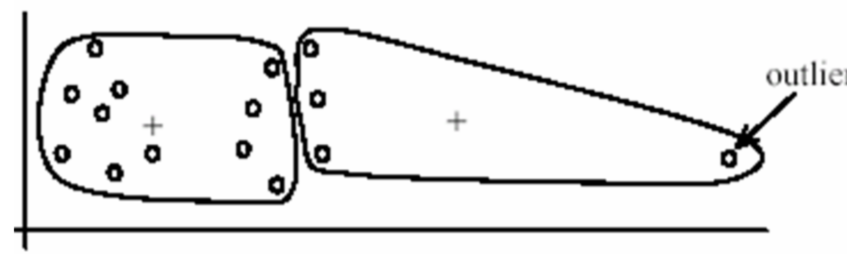
Slide: Derek Hoiem

K-Means pros and cons

- Pros
  - Finds cluster centers that minimize conditional variance (good representation of data)
  - Simple and fast*
  - Easy to implement
- Cons
  - Need to choose K
  - Sensitive to outliers
  - Prone to local minima
  - All clusters have the same parameters (e.g., distance measure is non-adaptive)
  - *Can be slow: each iteration is O(KNd) for N d-dimensional points
- Usage
  - Rarely used for pixel segmentation

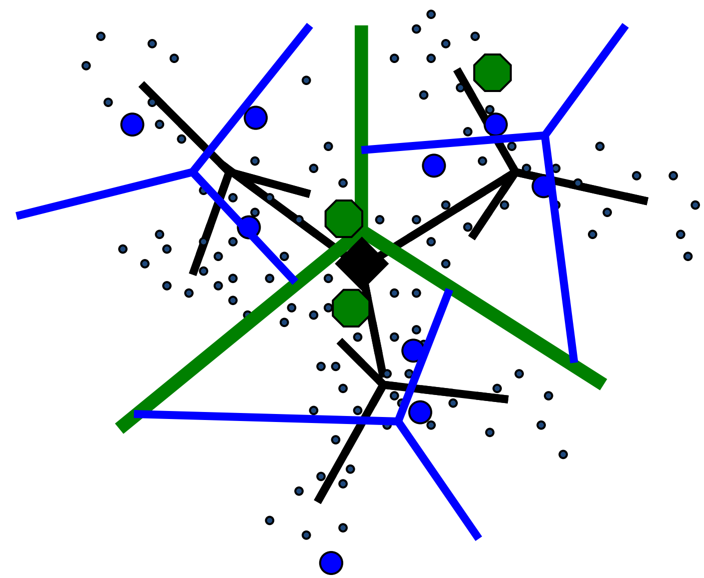


(B): Ideal clusters



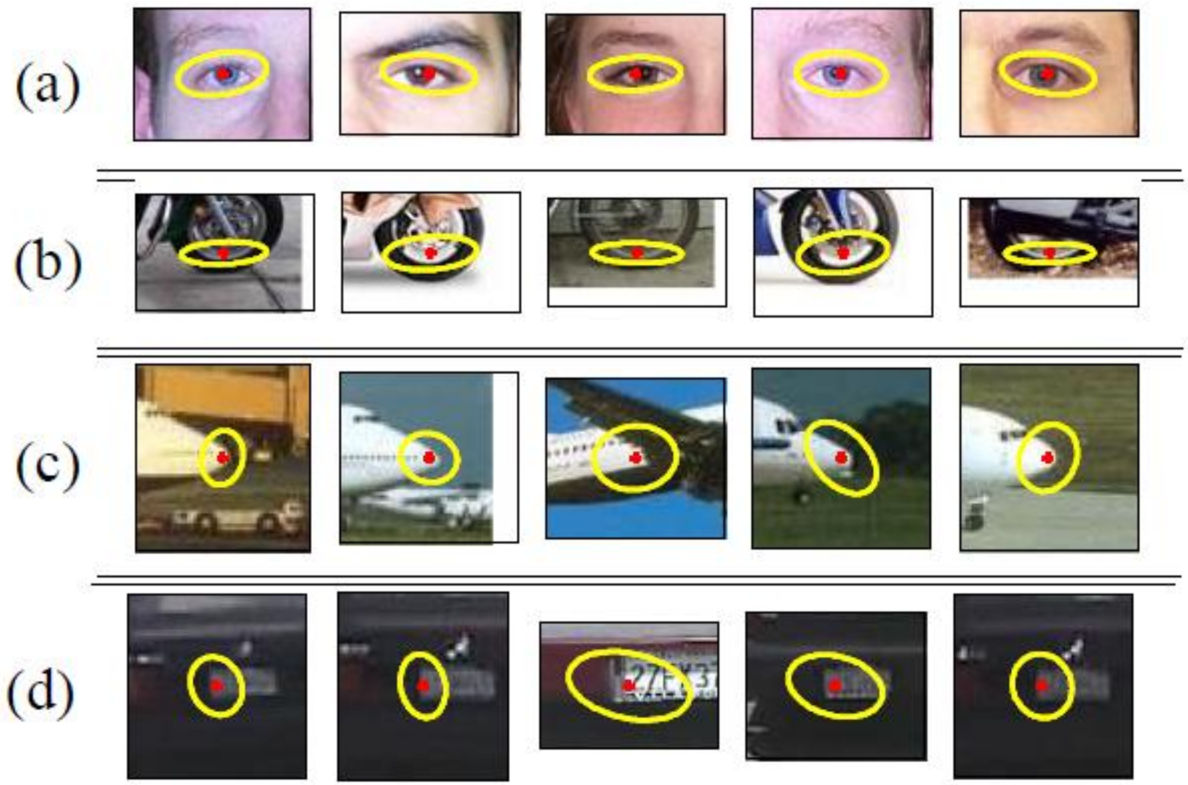
Building Visual Dictionaries

1. Sample patches from a database
   - E.g., 128 dimensional SIFT vectors
2. Cluster the patches
   - Cluster centers are the dictionary



3. Assign a codeword (number) to each new patch, according to the nearest cluster

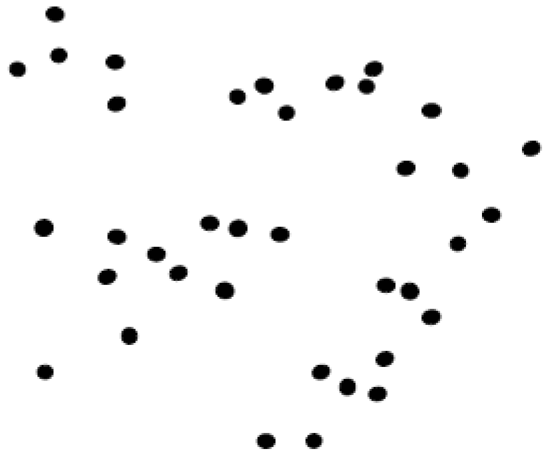
Examples of learned codewords



Most likely codewords for 4 learned "topics"
EM with multinomial (problem 3) to get topics

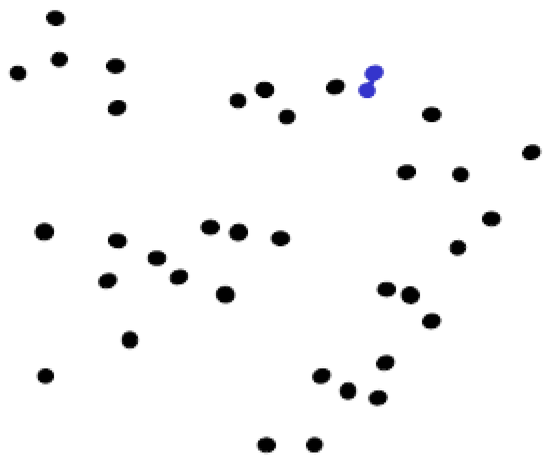
http://www.robots.ox.ac.uk/~vgg/publications/papers/sivic05b.pdf Sivic et al. ICCV 2005

Agglomerative clustering



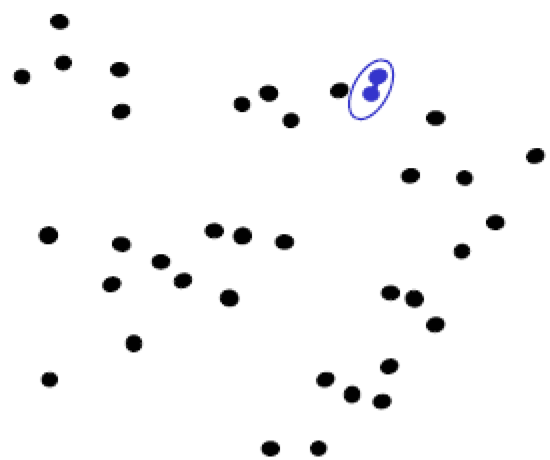
1. Say "Every point is its own cluster"

Agglomerative clustering

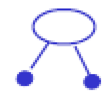


1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters

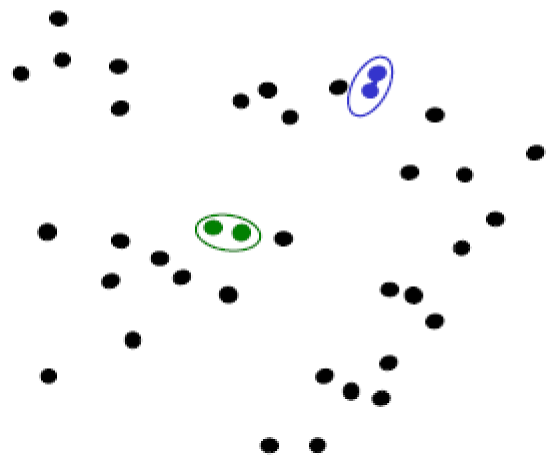
Agglomerative clustering



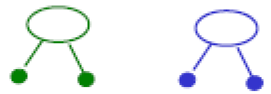
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster



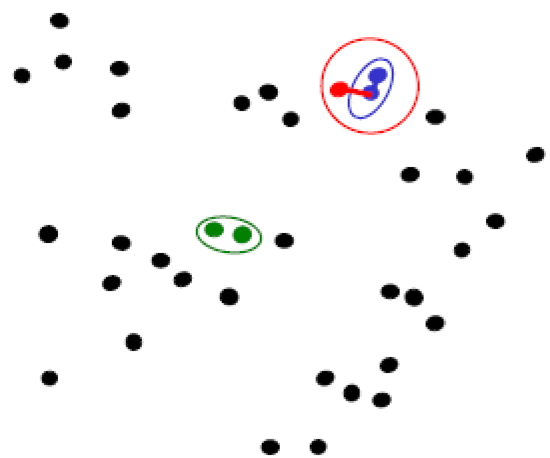
Agglomerative clustering



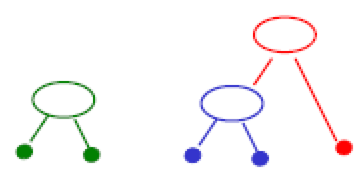
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat



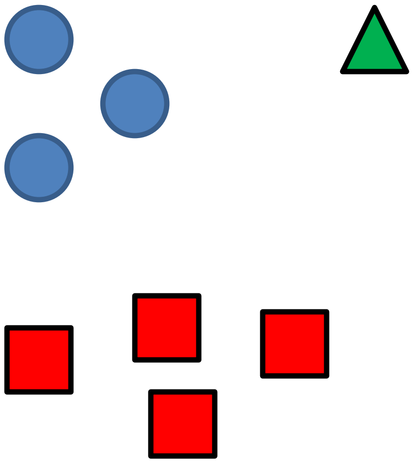
Agglomerative clustering



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat



Agglomerative clustering

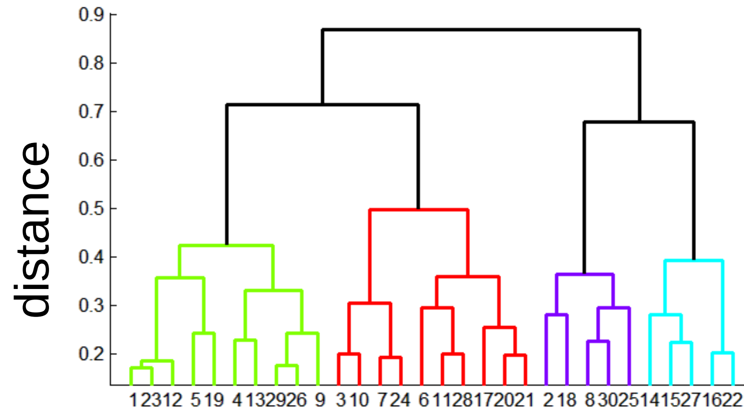


How to define cluster similarity?

- Average distance between points, maximum distance, minimum distance
- Distance between means or medoids

How many clusters?

- Clustering creates a dendrogram (a tree)
- Threshold based on max number of clusters or based on distance between merges



Conclusions: Agglomerative Clustering

Good

- Simple to implement, widespread application
- Clusters have adaptive shapes
- Provides a hierarchy of clusters

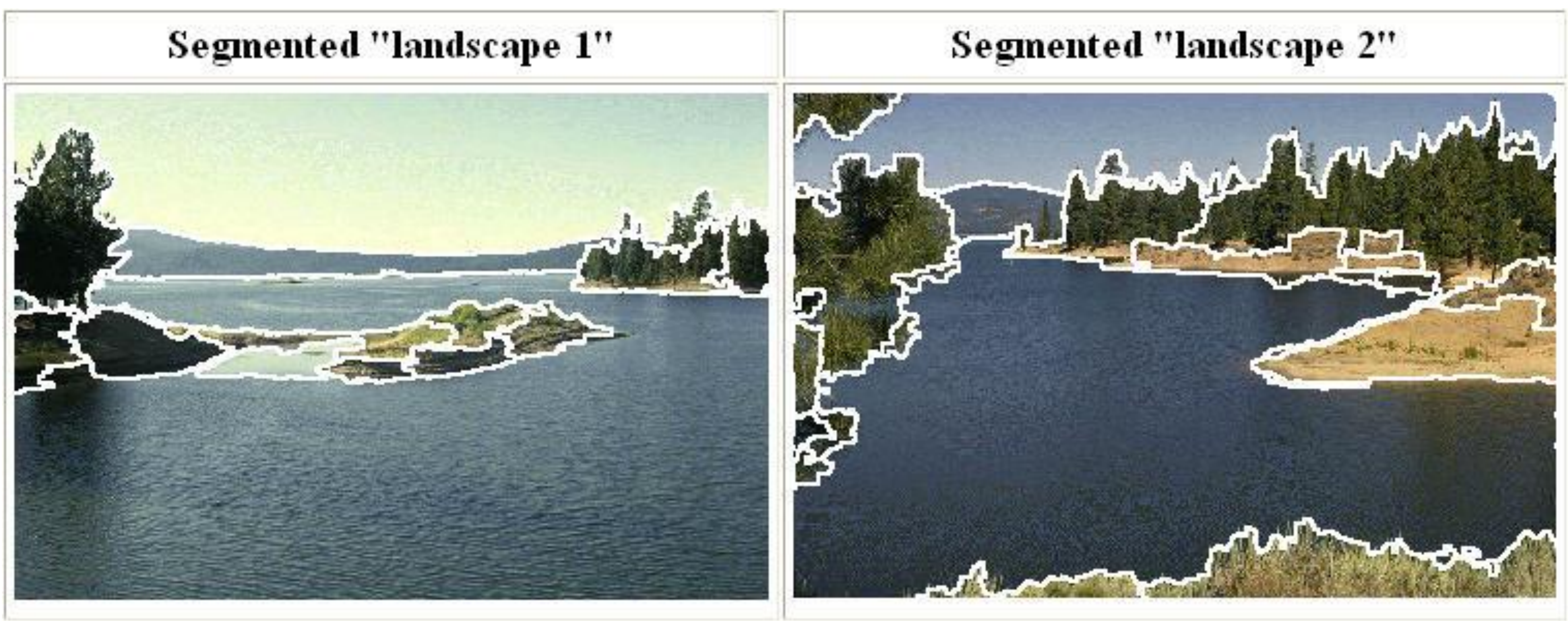
Bad

- May have imbalanced clusters
- Still have to choose number of clusters or threshold
- Need to use an “ultrametric” to get a meaningful hierarchy

Mean shift segmentation

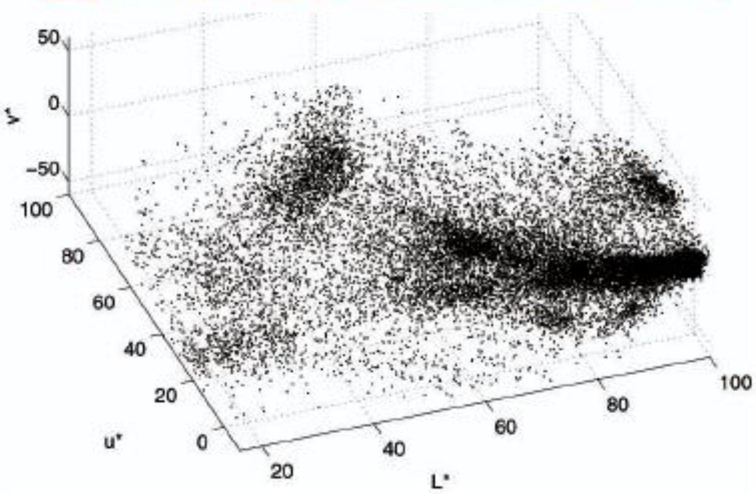
D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002.

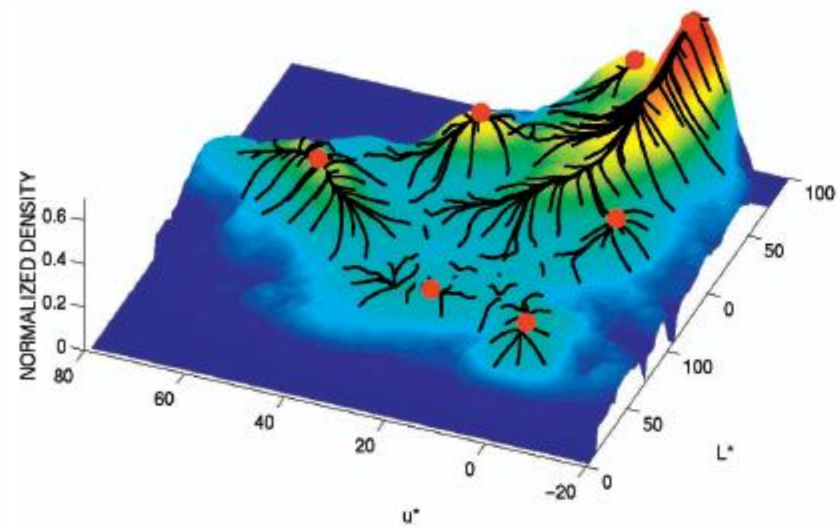
- Versatile technique for clustering-based segmentation

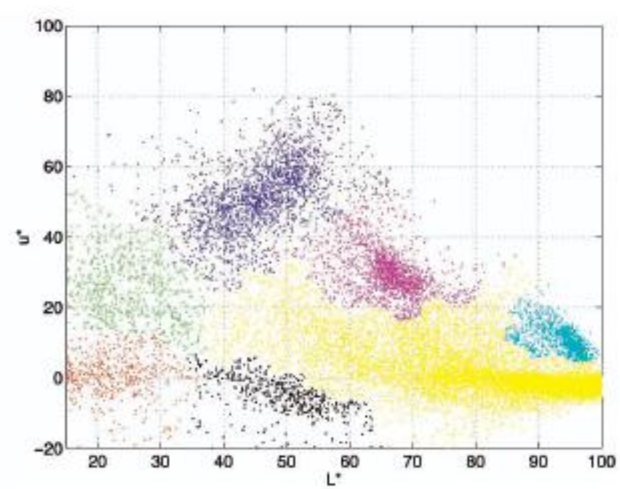


Mean shift algorithm

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







Kernel density estimation

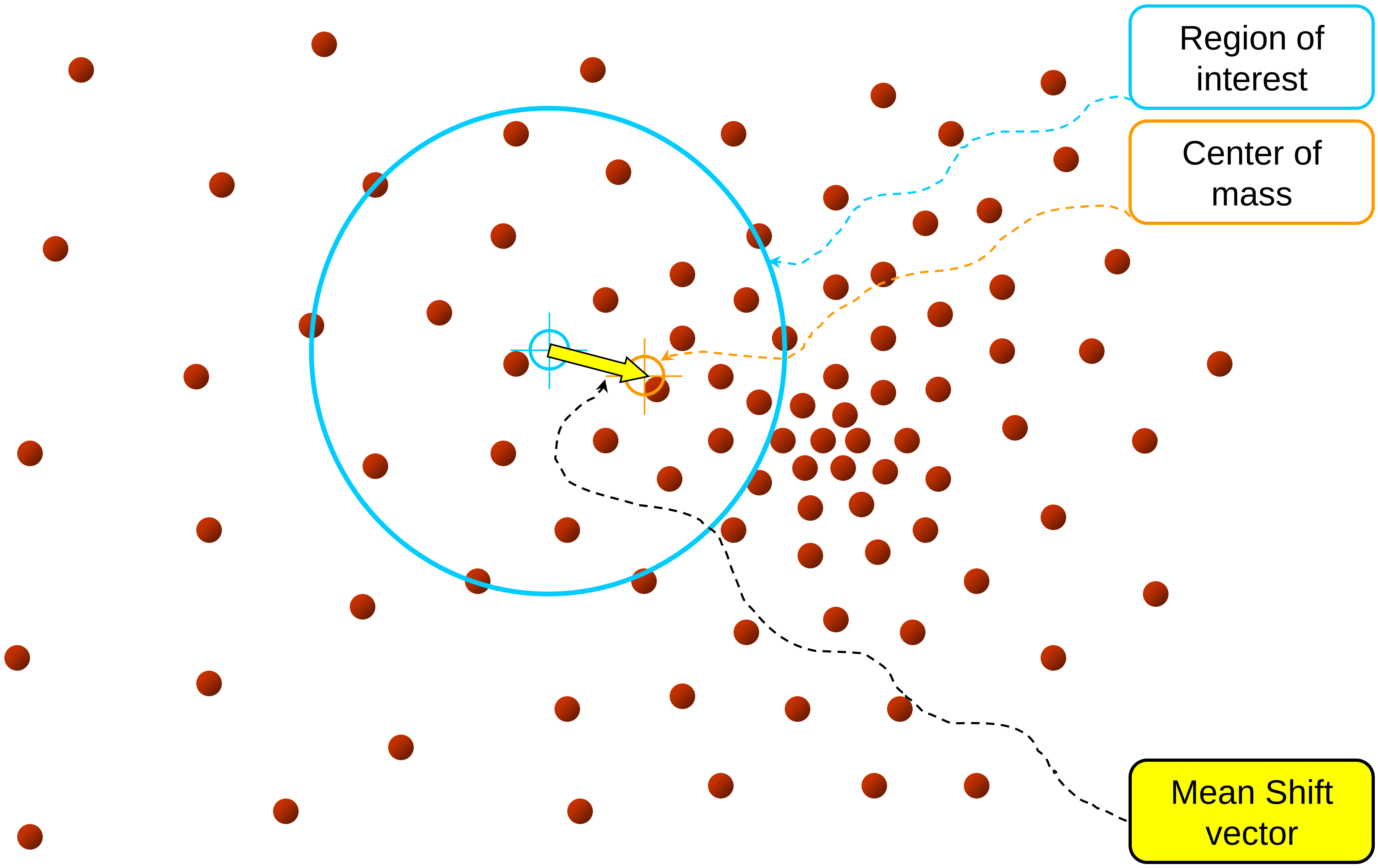
Kernel density estimation function

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

Gaussian kernel

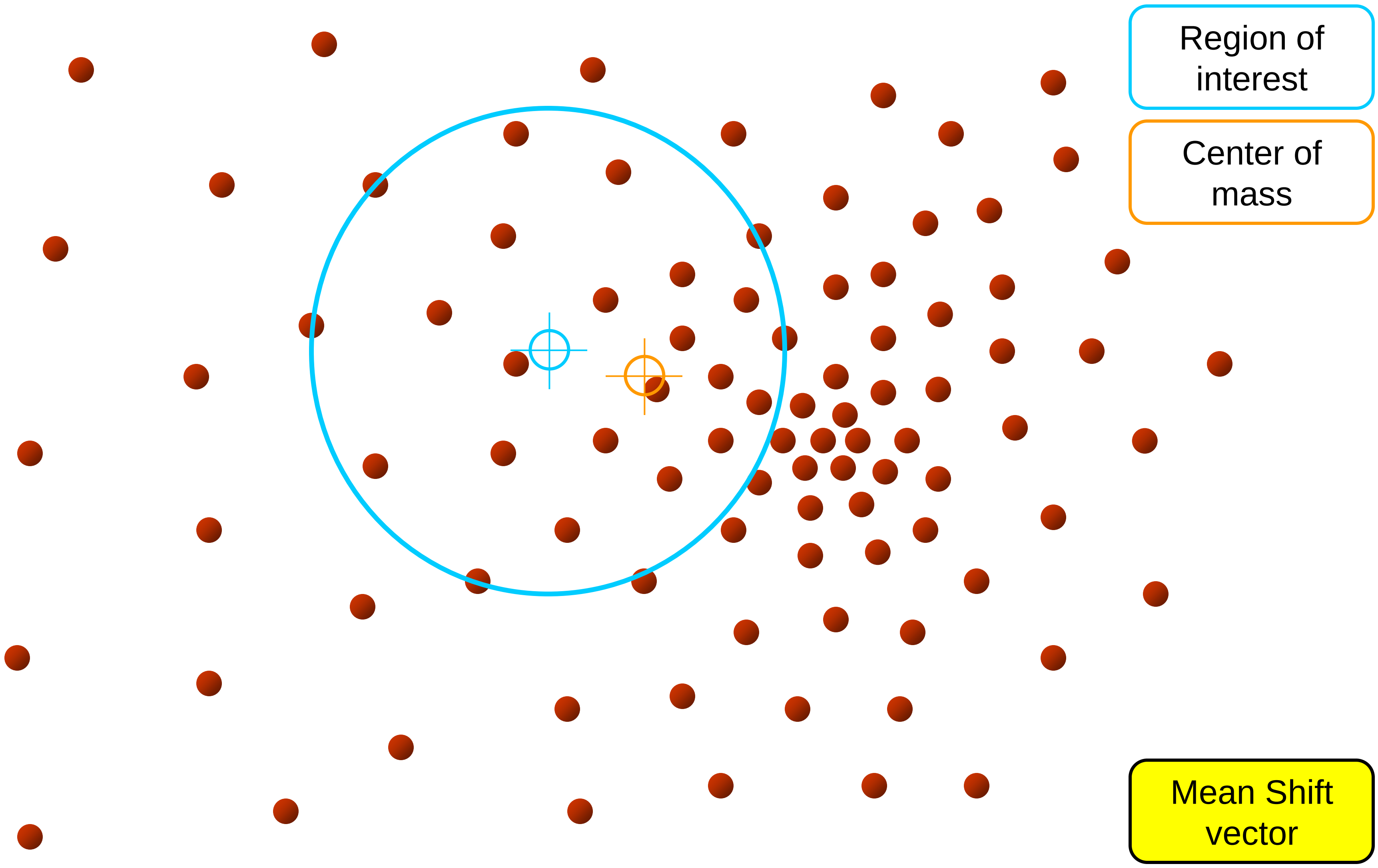
$$K\left(\frac{x - x_i}{h}\right) = \frac{1}{\sqrt{2\pi}}\, e^{-\frac{(x-x_i)^2}{2h^2}}.$$

Mean shift



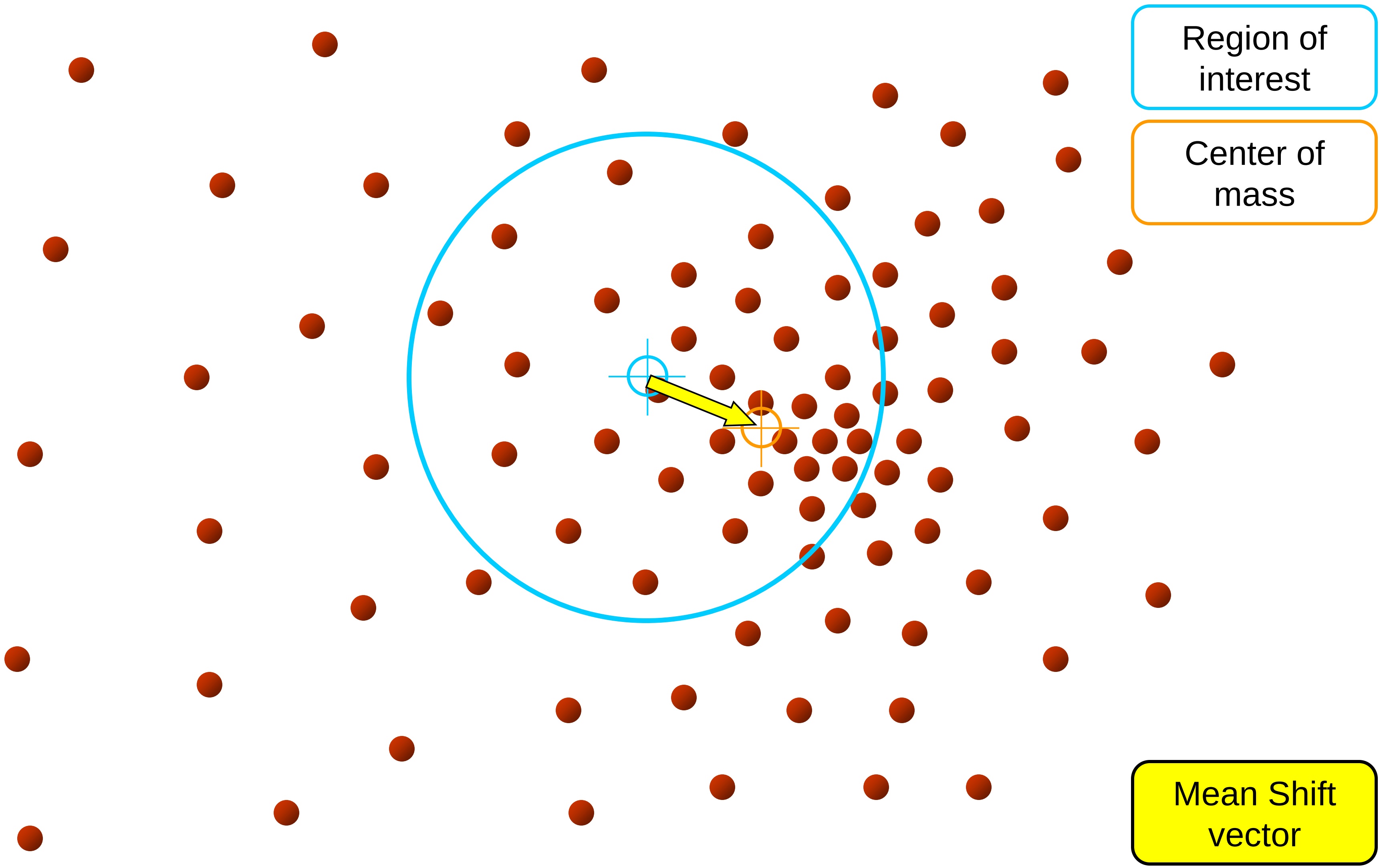
Slide by Y. Ukrainitz & B. Sarel

Mean shift



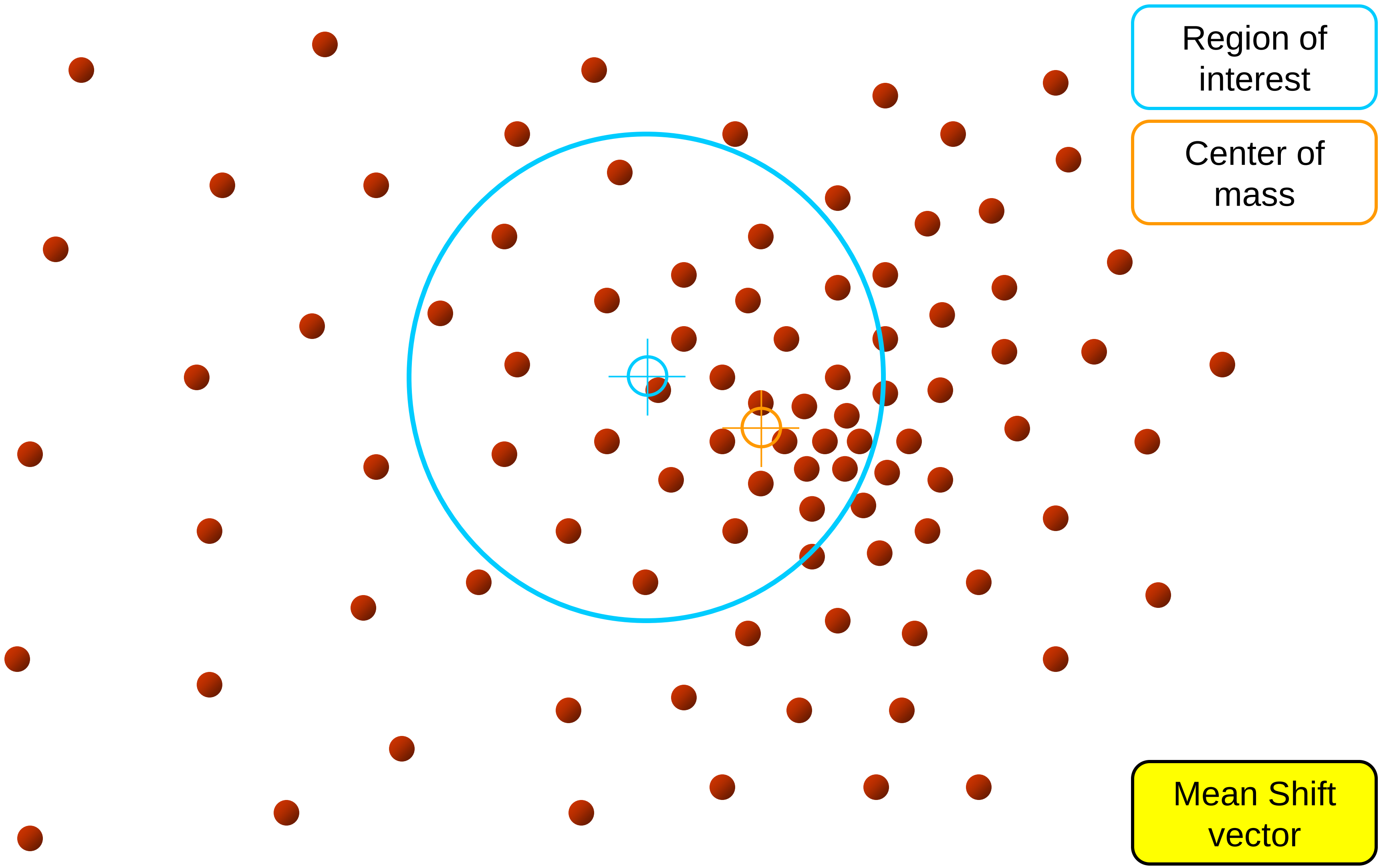
Slide by Y. Ukrainitz & B. Sarel

Mean shift



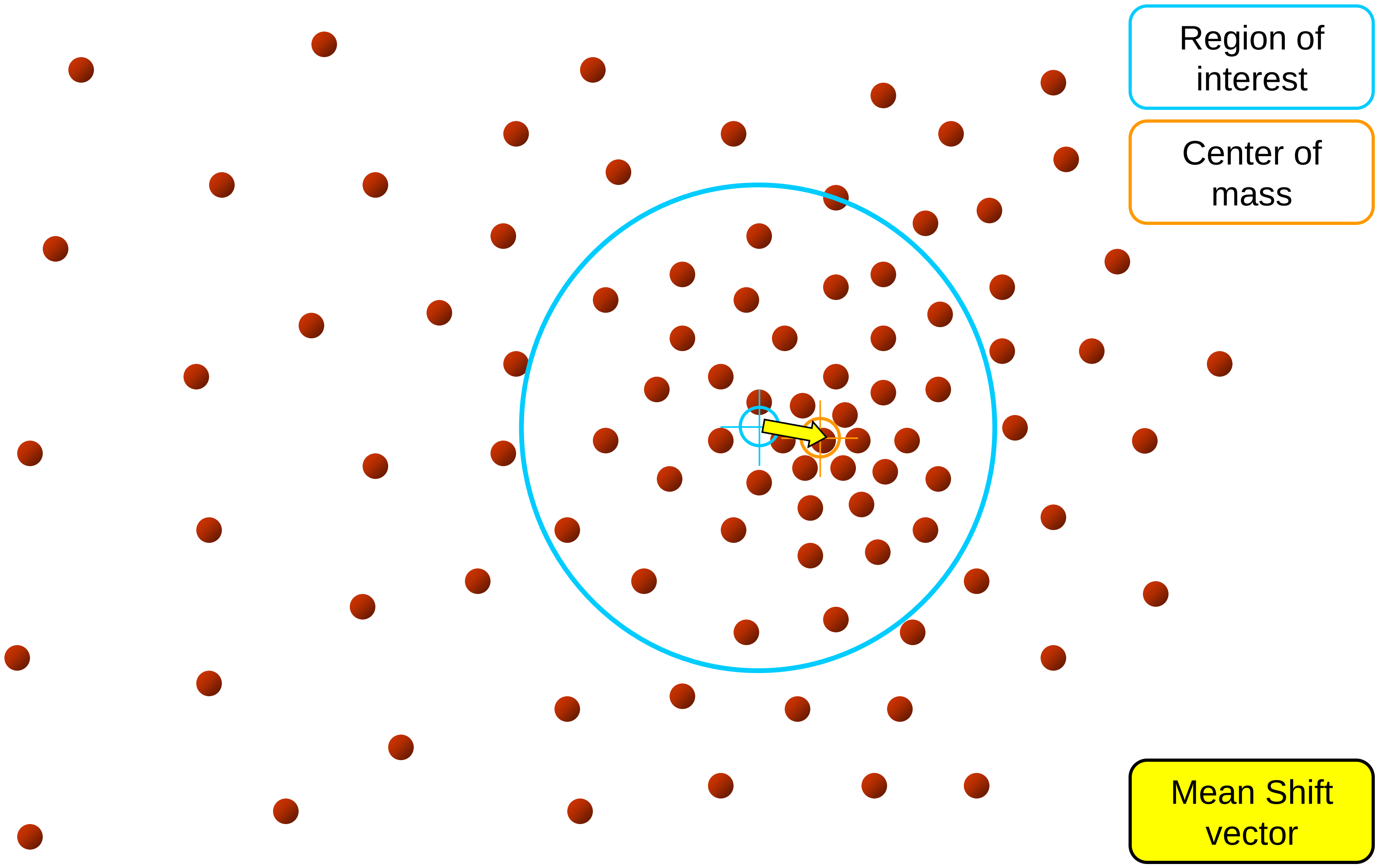
Slide by Y. Ukrainitz & B. Sarel

Mean shift



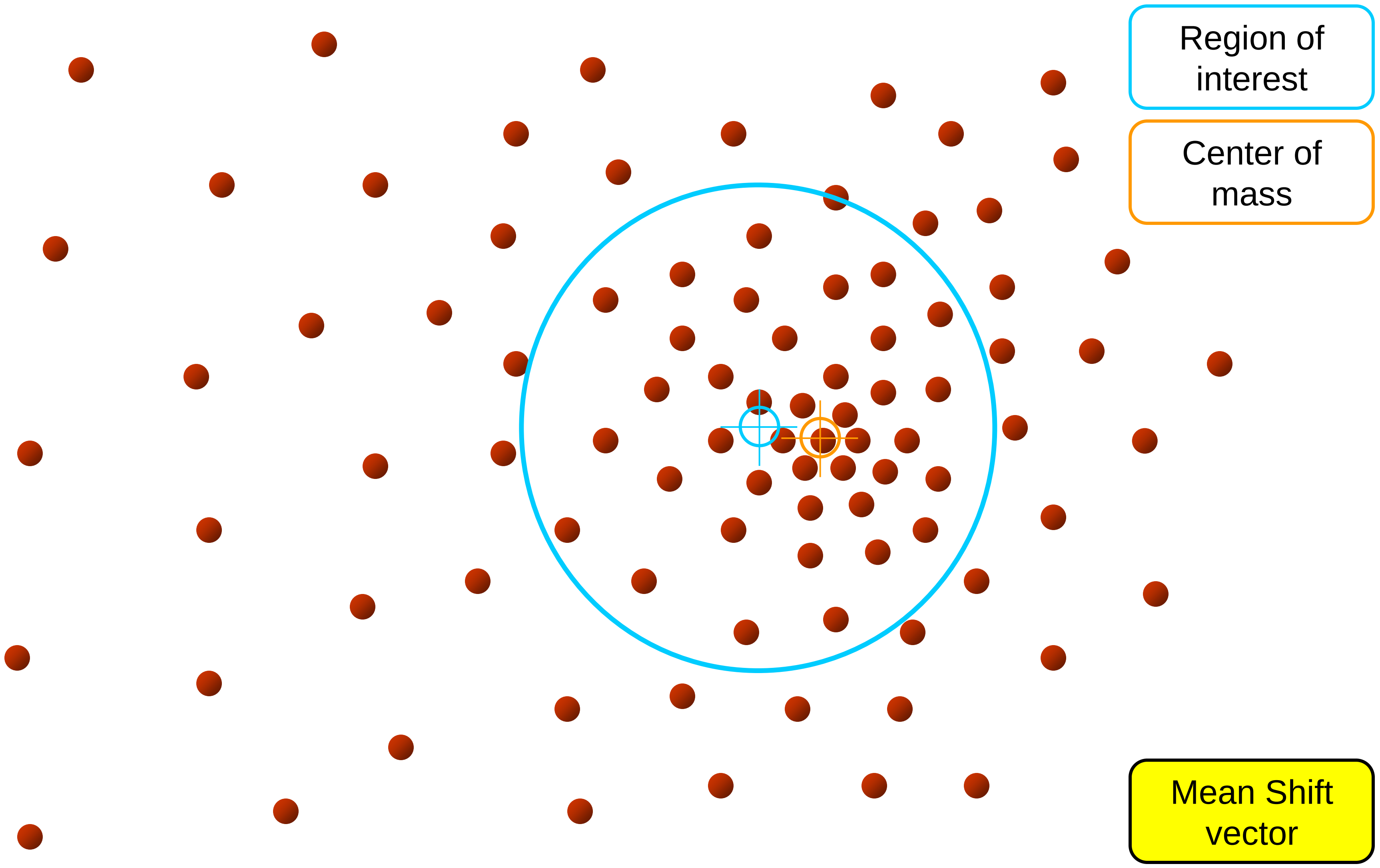
Slide by Y. Ukrainitz & B. Sarel

Mean shift



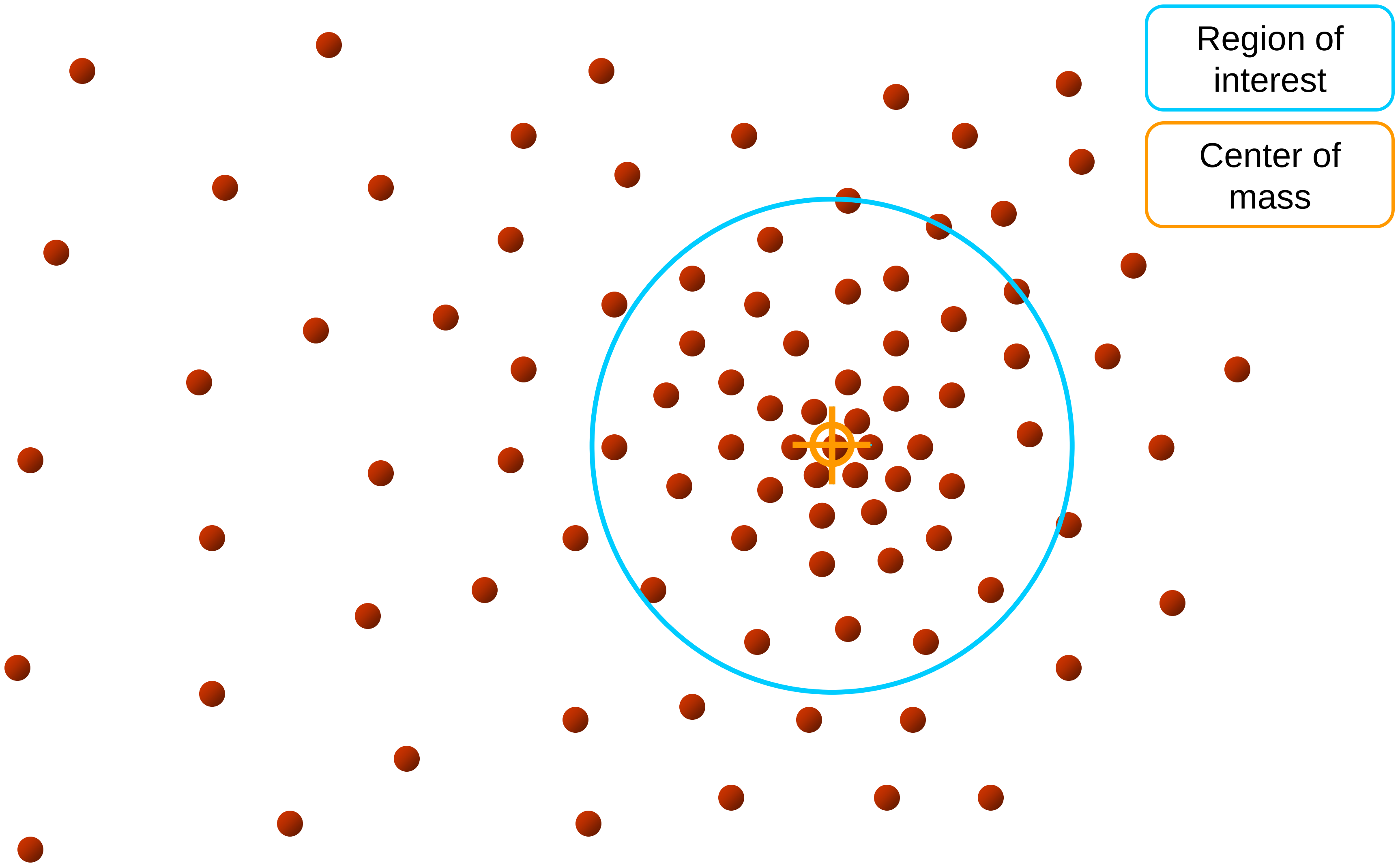
Slide by Y. Ukrainitz & B. Sarel

Mean shift



Slide by Y. Ukrainitz & B. Sarel

Mean shift



Slide by Y. Ukrainitz & B. Sarel

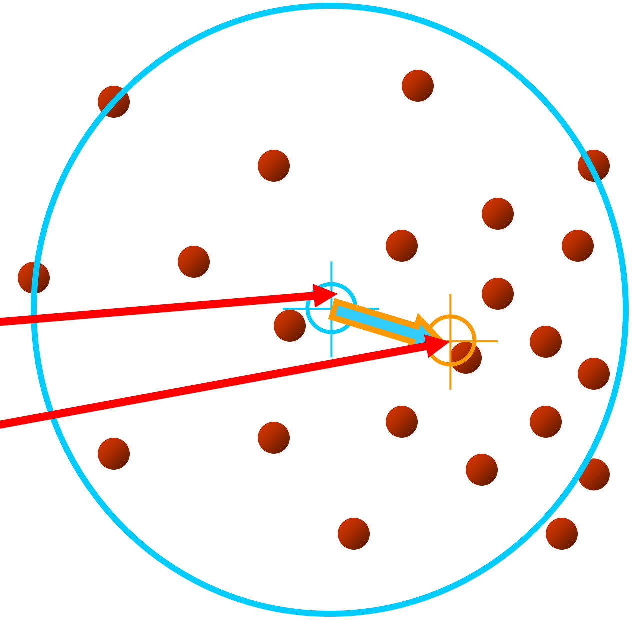
Computing the Mean Shift

Simple Mean Shift procedure:

• Compute mean shift vector

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

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

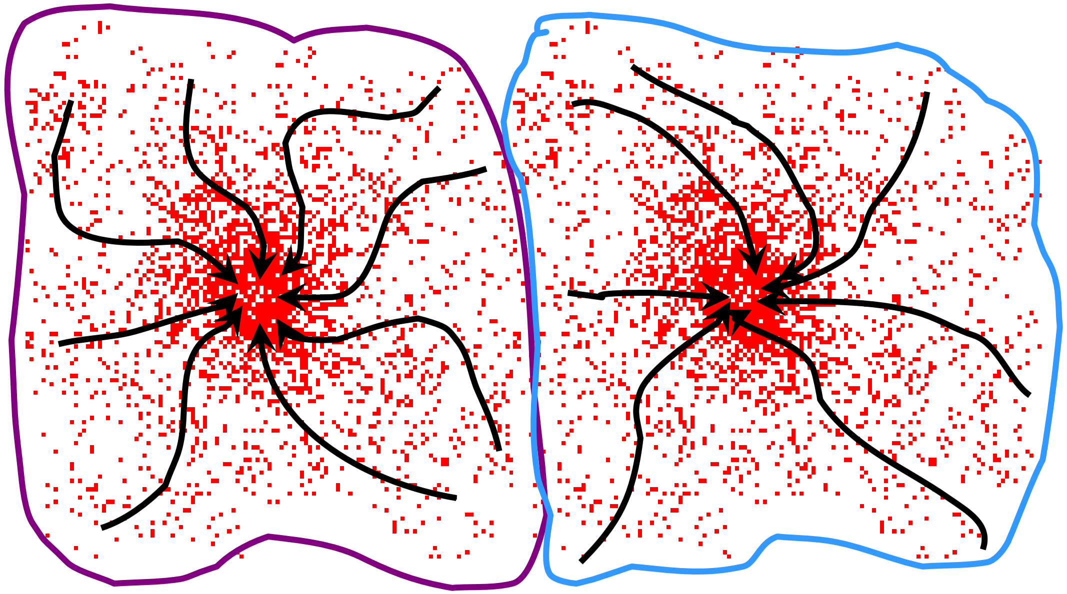


Slide by Y. Ukrainitz & B. Sarel

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

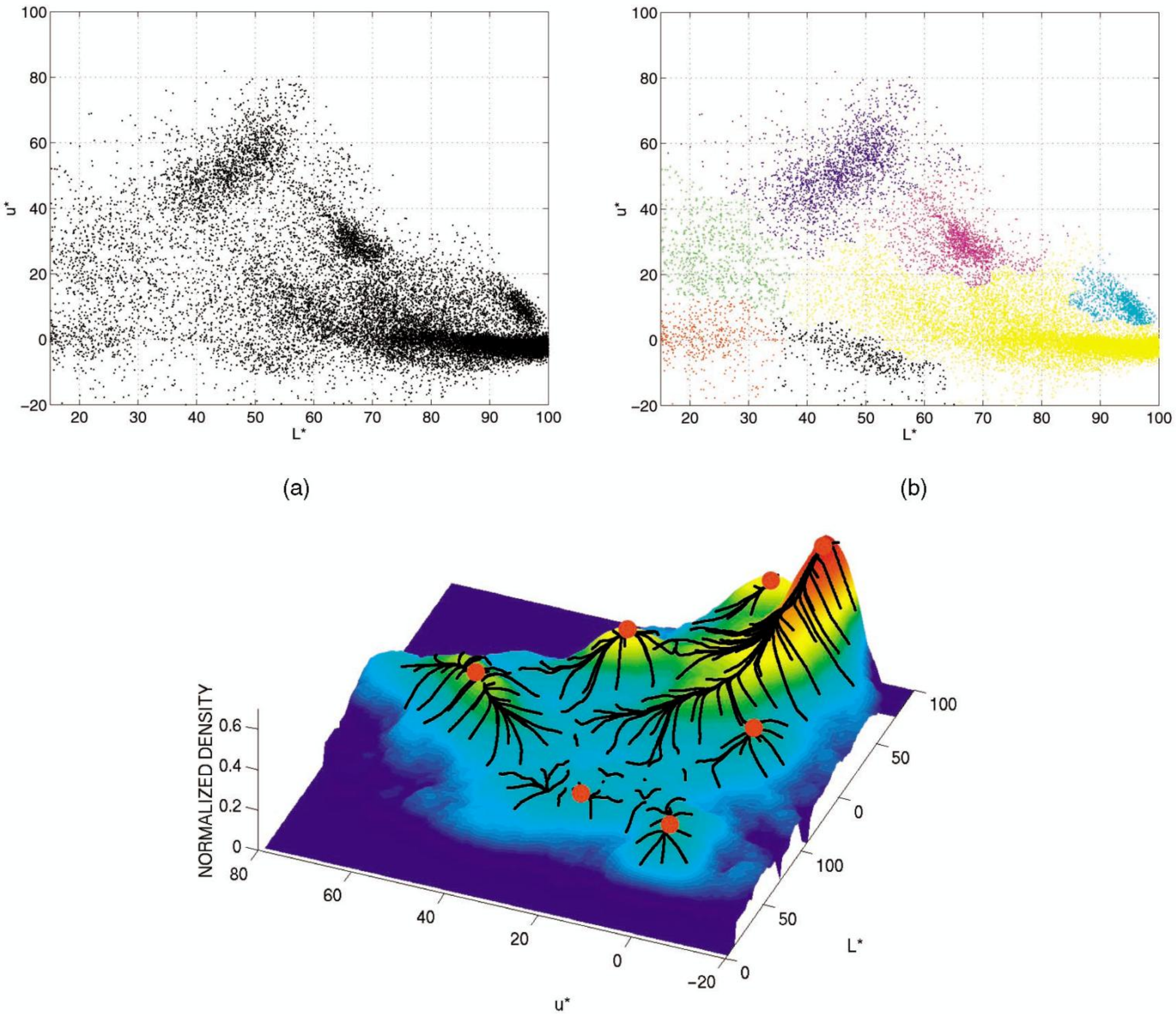
Attraction basin

- 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

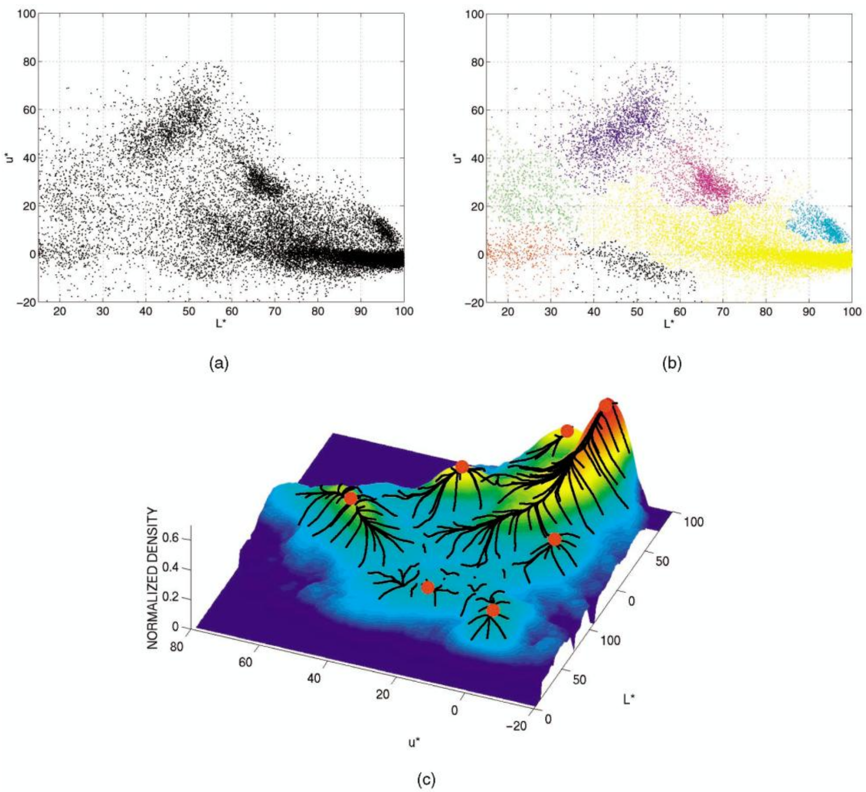


Mean shift clustering

- 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

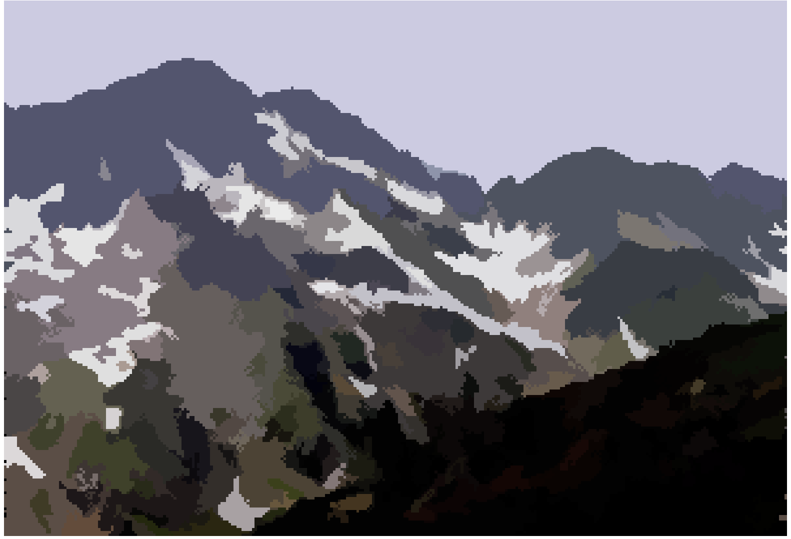
Segmentation by Mean Shift

- Compute features for each pixel (color, gradients, texture, etc)
- Set kernel size for features K_f and position K_s
- Initialize windows at individual pixel locations
- Perform mean shift for each window until convergence
- Merge windows that are within width of K_f and K_s

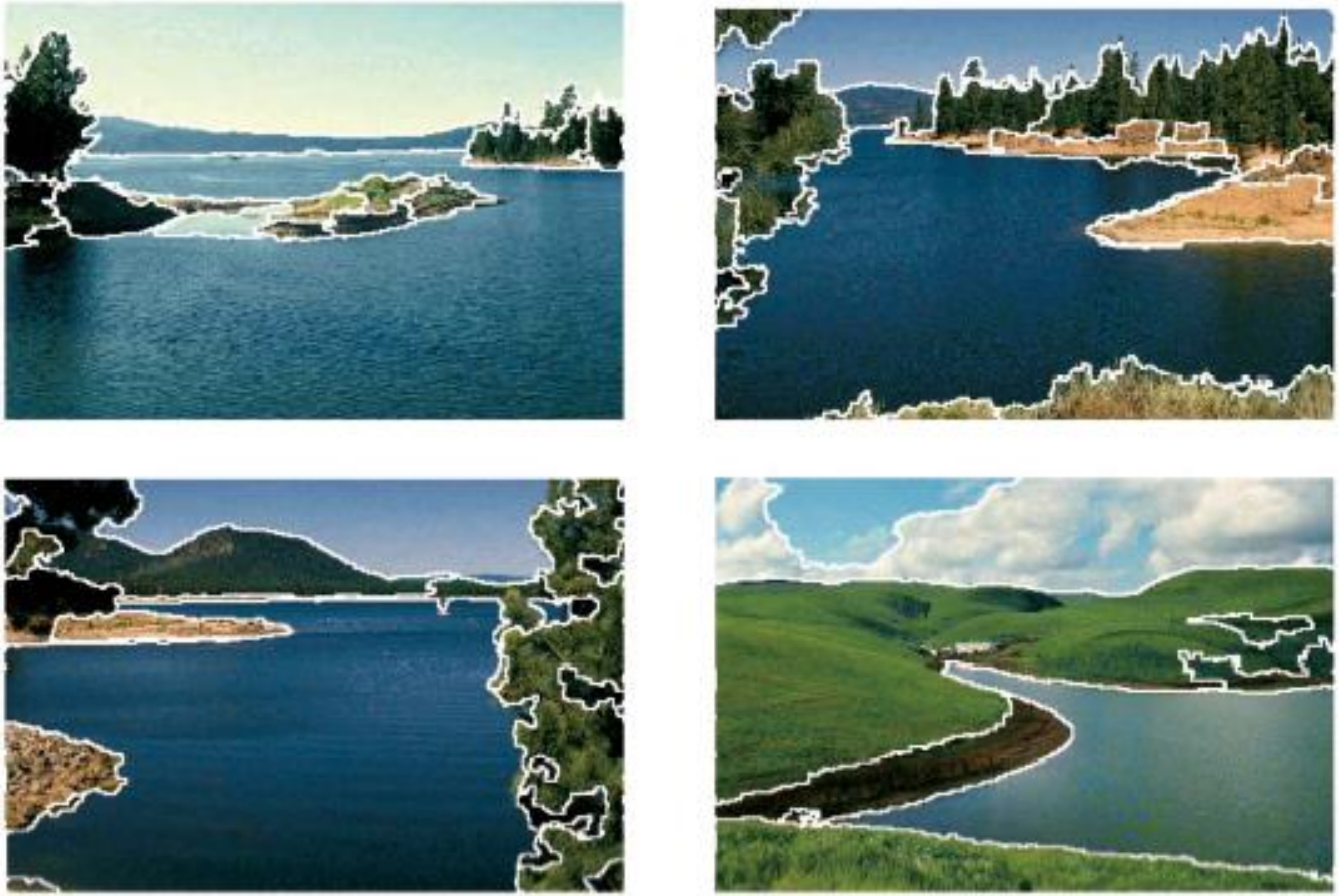


Mean shift segmentation results



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



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

Mean-shift: other issues

- Speedups
  - Binned estimation
  - Fast search of neighbors
  - Update each window in each iteration (faster convergence)

- Other tricks
  - Use kNN to determine window sizes adaptively

- Lots of theoretical support

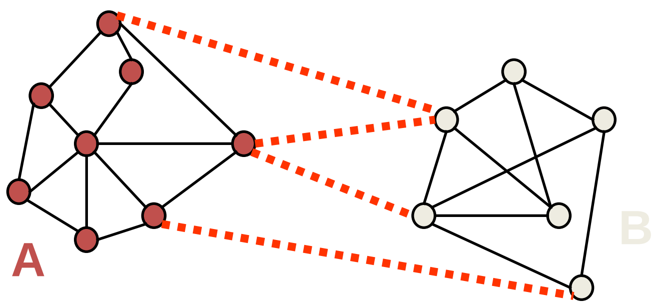
  D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002.

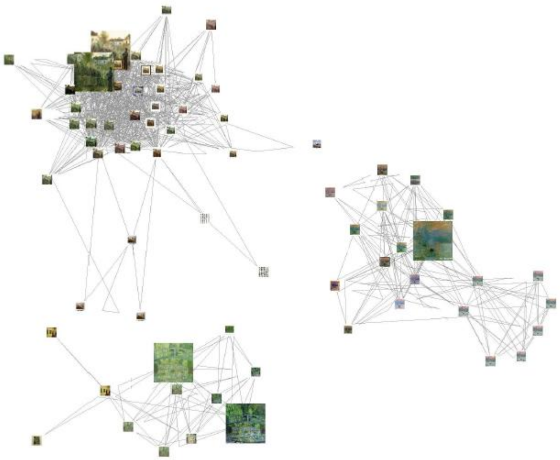
Mean shift pros and cons

- Pros
  - Good general-practice segmentation
  - Flexible in number and shape of regions
  - Robust to outliers
- Cons
  - Have to choose kernel size in advance
  - Not suitable for high-dimensional features
- When to use it
  - Oversegmentatoin
  - Multiple segmentations
  - Tracking, clustering, filtering applications

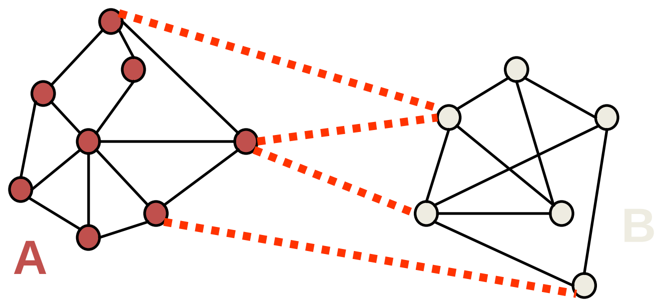
Spectral clustering

Group points based on links in a graph





Cuts in a graph



Normalized Cut

- a cut penalizes large segments
- fix by normalizing for size of segments

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

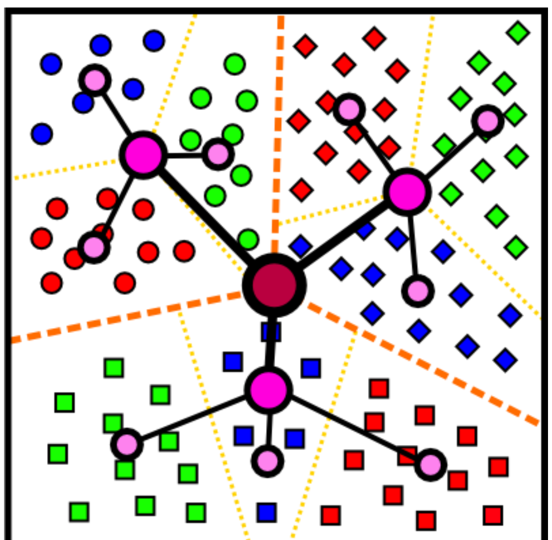
- volume(A) = sum of costs of all edges that touch A

Source: Seitz

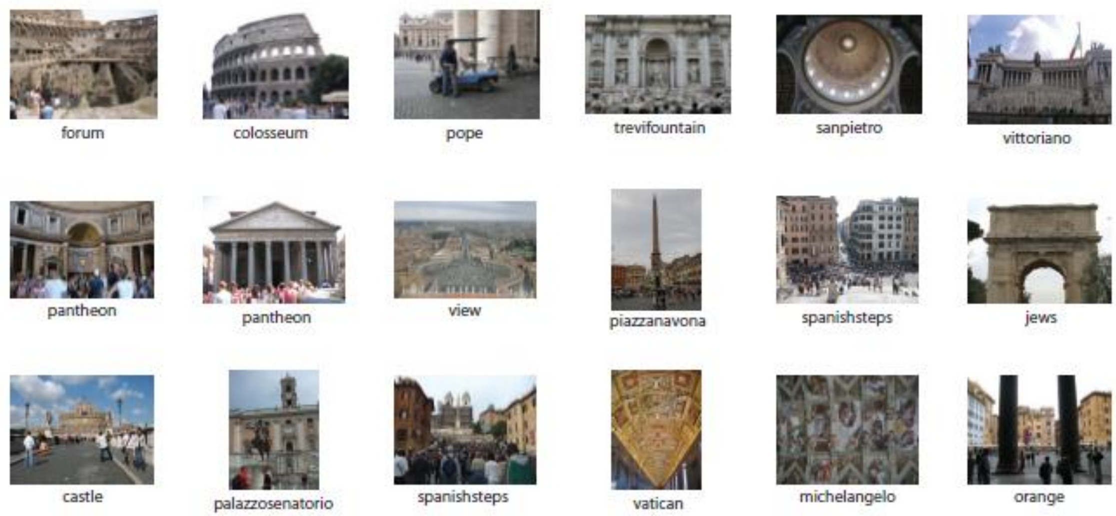
Normalized cuts for segmentation

Which algorithm to use?

- Quantization/Summarization: K-means
  - Aims to preserve variance of original data
  - Can easily assign new point to a cluster



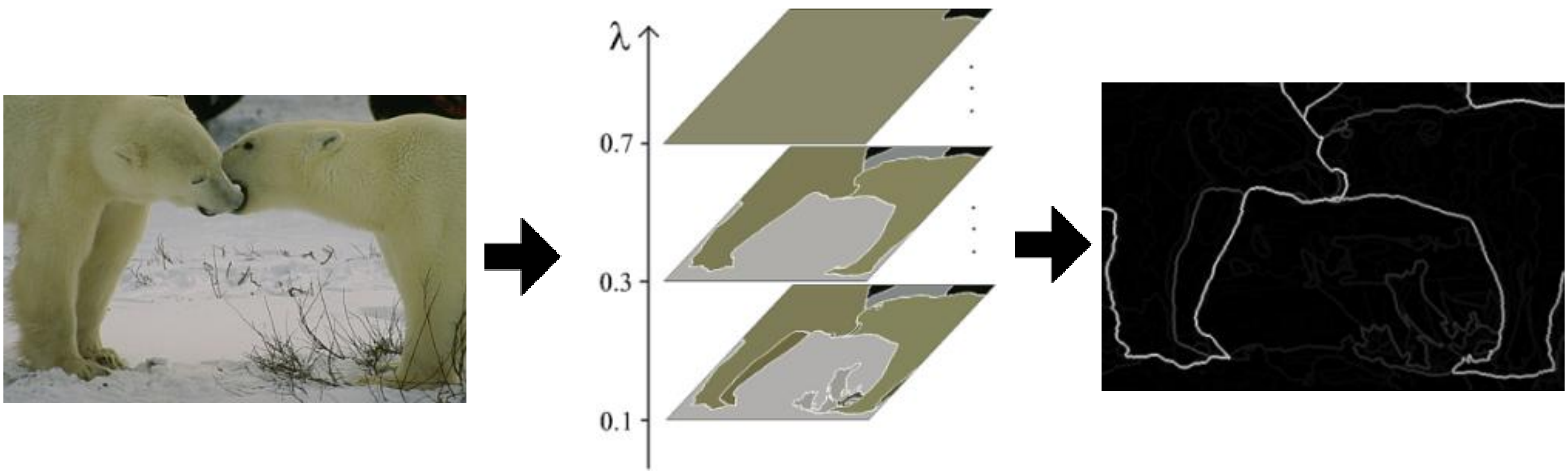
Quantization for computing histograms



Summary of 20,000 photos of Rome using "greedy k-means"
http://grail.cs.washington.edu/projects/canonview/

Which algorithm to use?

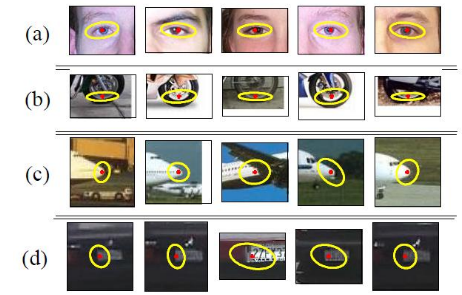
- Image segmentation: agglomerative clustering
  - More flexible with distance measures (e.g., can be based on boundary prediction)
  - Adapts better to specific data
  - Hierarchy can be useful



http://www.cs.berkeley.edu/~arbelaez/UCM.html

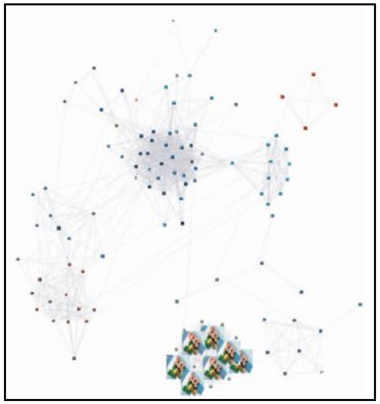
Things to remember

- K-means useful for summarization, building dictionaries of patches, general clustering



- Agglomerative clustering useful for segmentation, general clustering

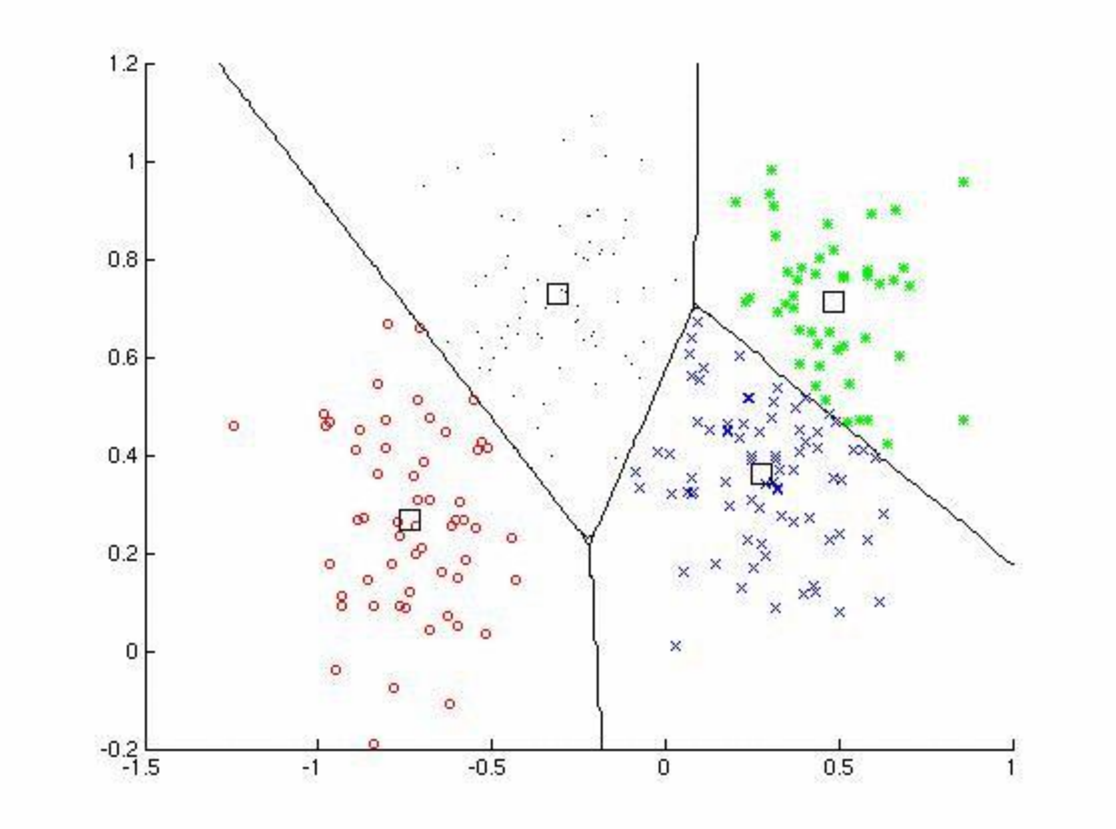
- Spectral clustering useful for determining relevance, summarization, segmentation



Clustering

Key algorithm

- K-means



Next Lecture:

- Machine Learning: Classification