SAIR
Spatial AI & Robotics Lab

CSE 473/573-A

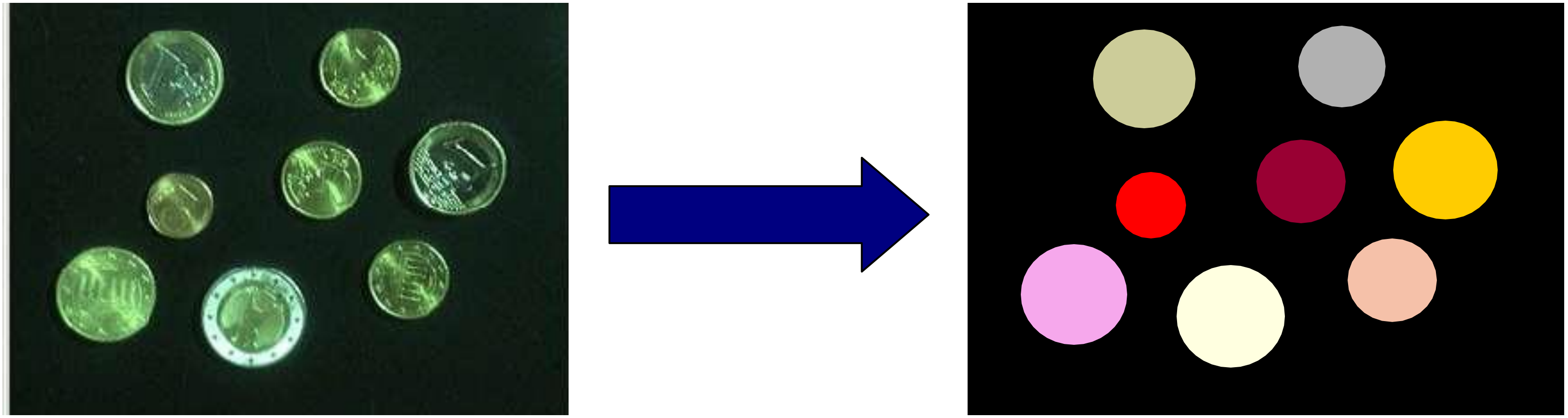
L16: SEGMENTATION

Chen Wang
Spatial AI & Robotics Lab
Department of Computer Science and Engineering

University at Buffalo The State University of New York

Image Segmentation

- **Partition** the pixels of an image into **groups** that strongly correlate with the objects in an image
- Typically, the first step in any automated computer vision application.



Why & What?

- Useful mid-level representation of an image
  - Can facilitate better downstream tasks
- Segmentation should be homogeneous with some characteristic (gray level, texture, color, motion)
- The desired type segmentation depends on the task
  - Broad theory is absent at present
- Variety of approaches, algorithms, and applications
  - Finding people, summarizing video, annotation figures, background subtraction, finding buildings/rivers in satellite images.

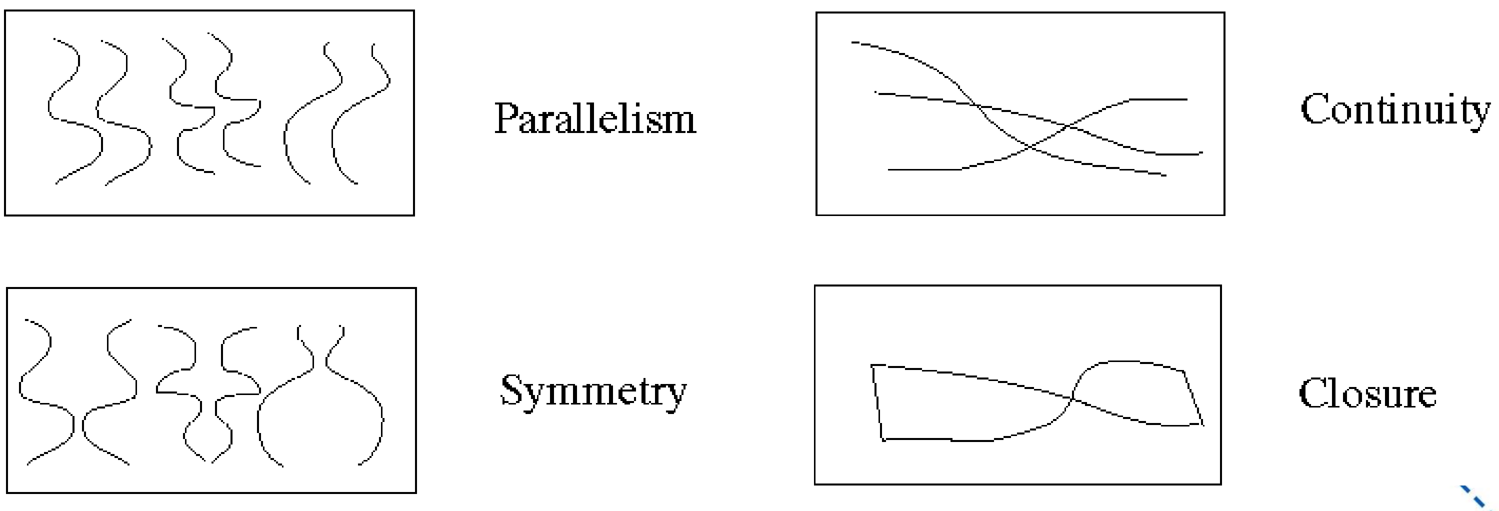
Thought Exercise….

- Based on what you already know about CVIP
  - How would you do segmentation?
- Did you come up with
  - Feature Detection?
  - Edge Detection?
  - Texture Grouping?
  - Hough Transform?
    - Let's assume types of objects are unknown.

Segmentation

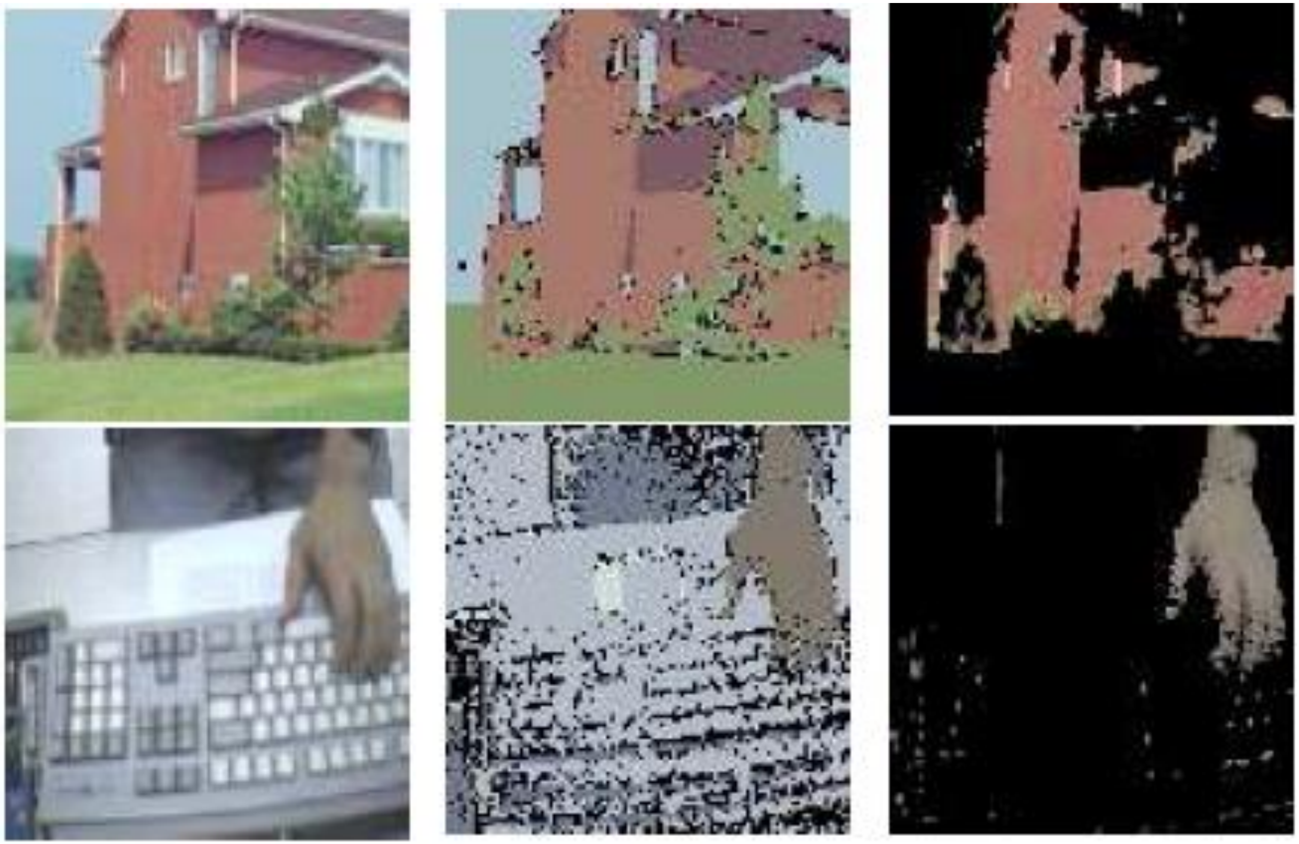
Segmentation algorithms generally are based on one of two basis properties of intensity values.

- **Discontinuity:** to partition an image based on abrupt changes in intensity (such as edges).
- **Similarity:** to partition an image into regions that are similar according to a set of predefined criteria.

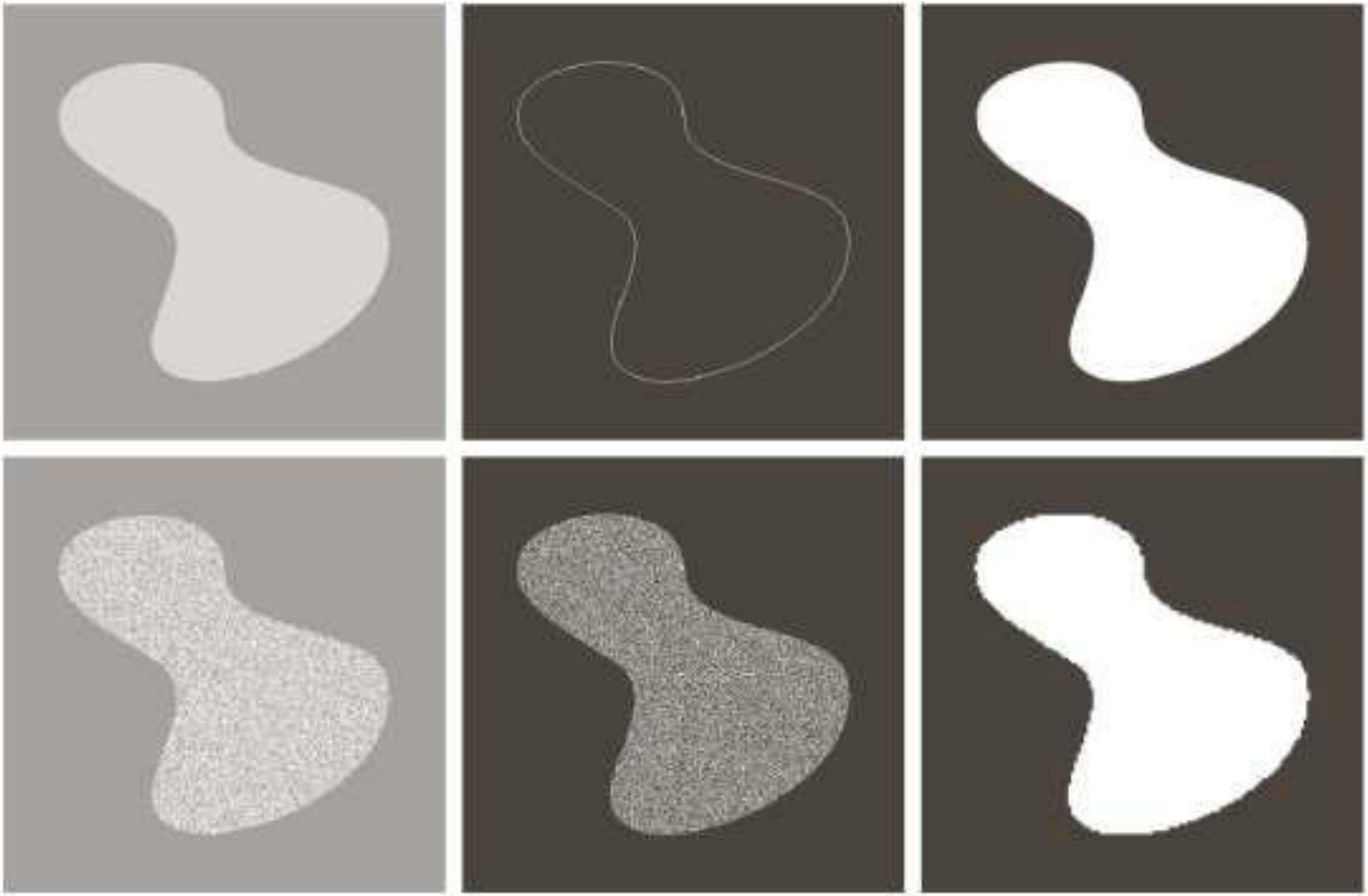


Segmentation needs Grouping/Clustering

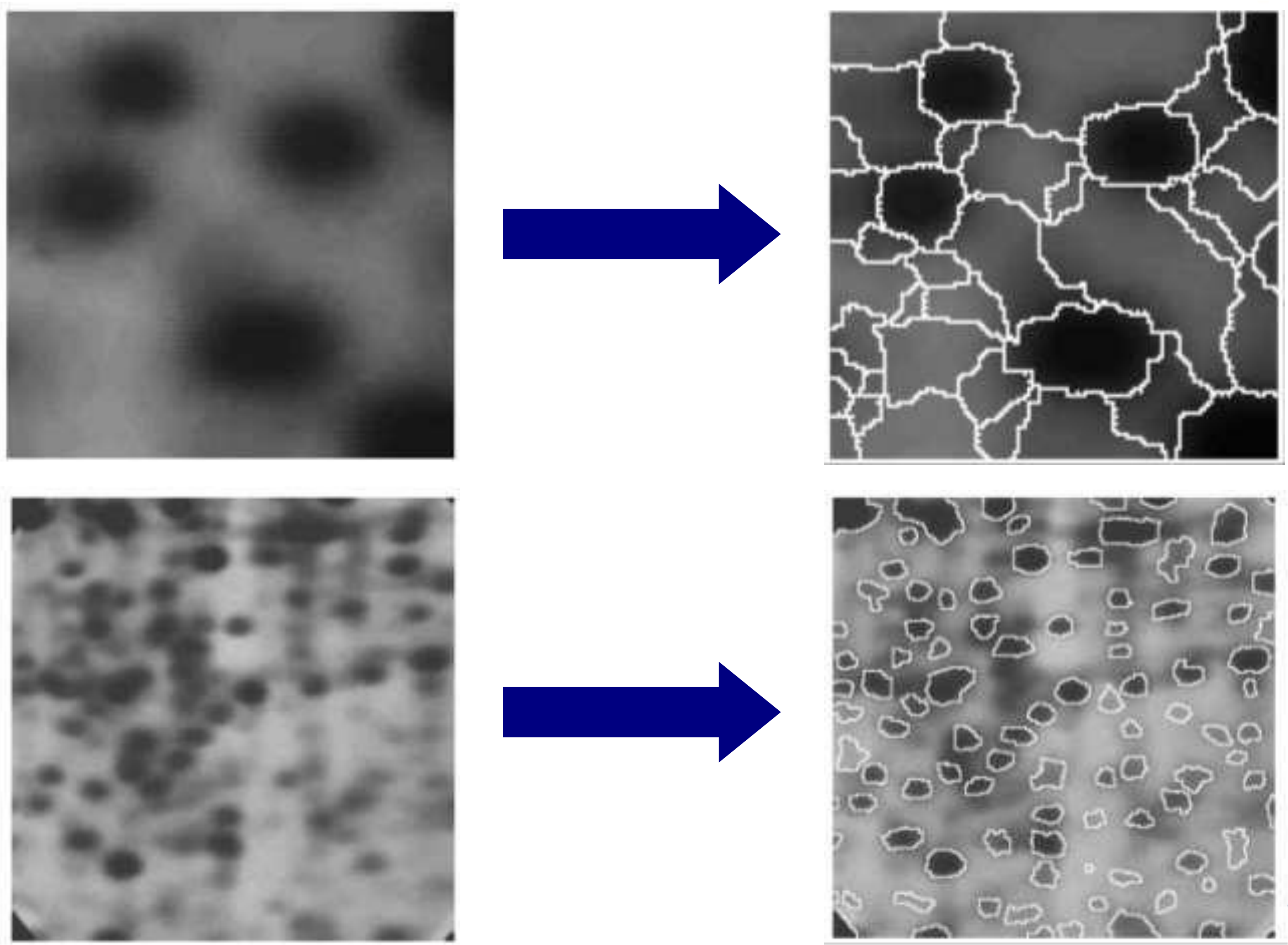
- Grouping (or clustering)
  - Collect pixels that "belong together"



Boundaries or Regions are equally good

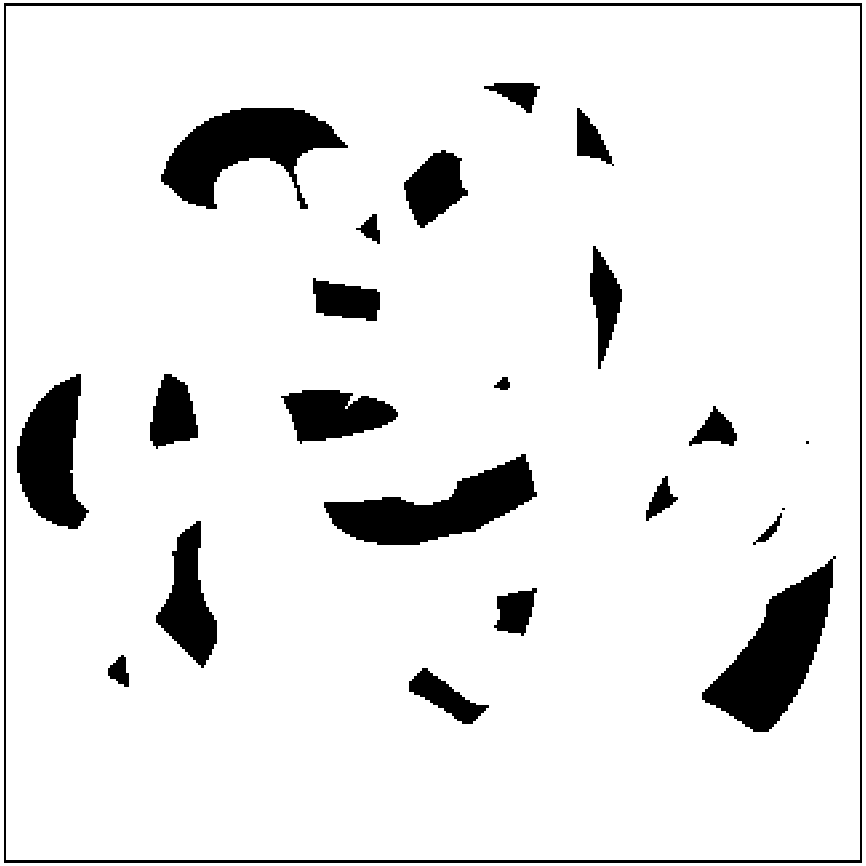


Boundaries or Regions are equally good



Occlusion cues are important

- What do you see?



Occlusion cues are important

- Aren't they?

Regions and Edges

- Regions are bounded by closed contours
  - We could “fill” closed contours to obtain regions
  - We could “trace” regions to obtain edges
- Unfortunately, these procedures rarely produce satisfactory results.

SAIR
Spatial AI & Robotics Lab

Regions and Edges

- **Edges** are found based on **DIFFERENCES** between values of adjacent pixels.
- **Regions** are found based on **SIMILARITIES** between values of adjacent pixels.
- We want group higher level units shared within regions of the image.

Machine Learning

	Supervised Learning	Unsupervised Learning
Discrete	classification or categorization	$\boxed{\text{clustering}}$
Continuous	regression	dimensionality reduction

Clustering example: image segmentation

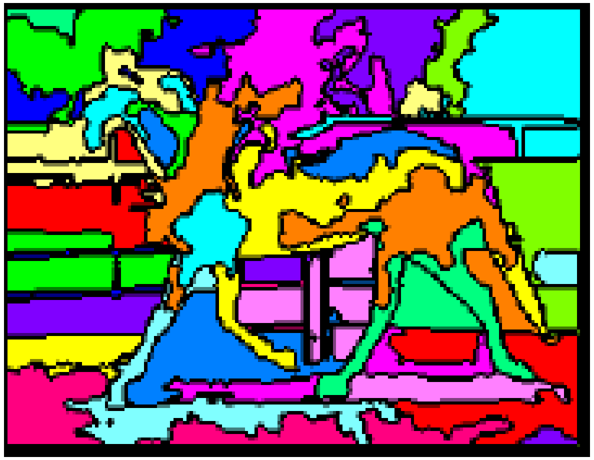
- Break up the image into **meaningful** or **perceptually similar** regions

Clustering example: image segmentation

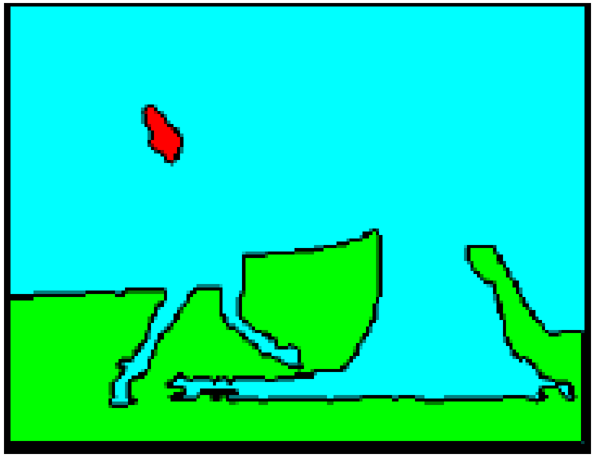
- Foreground/Background

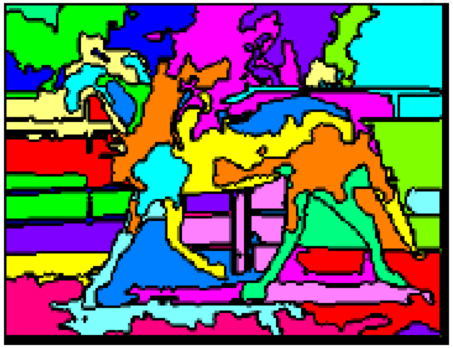
Types of segmentations

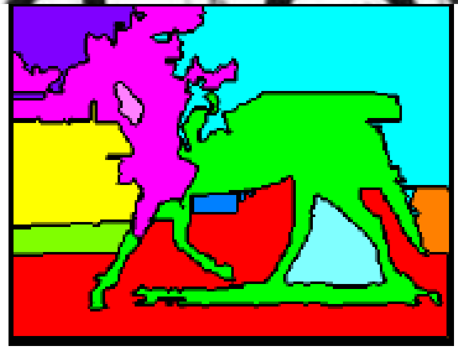
Oversegmentation

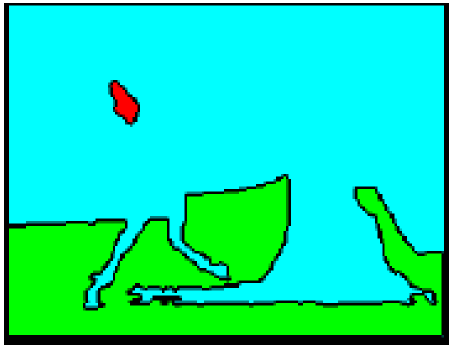


Undersegmentation





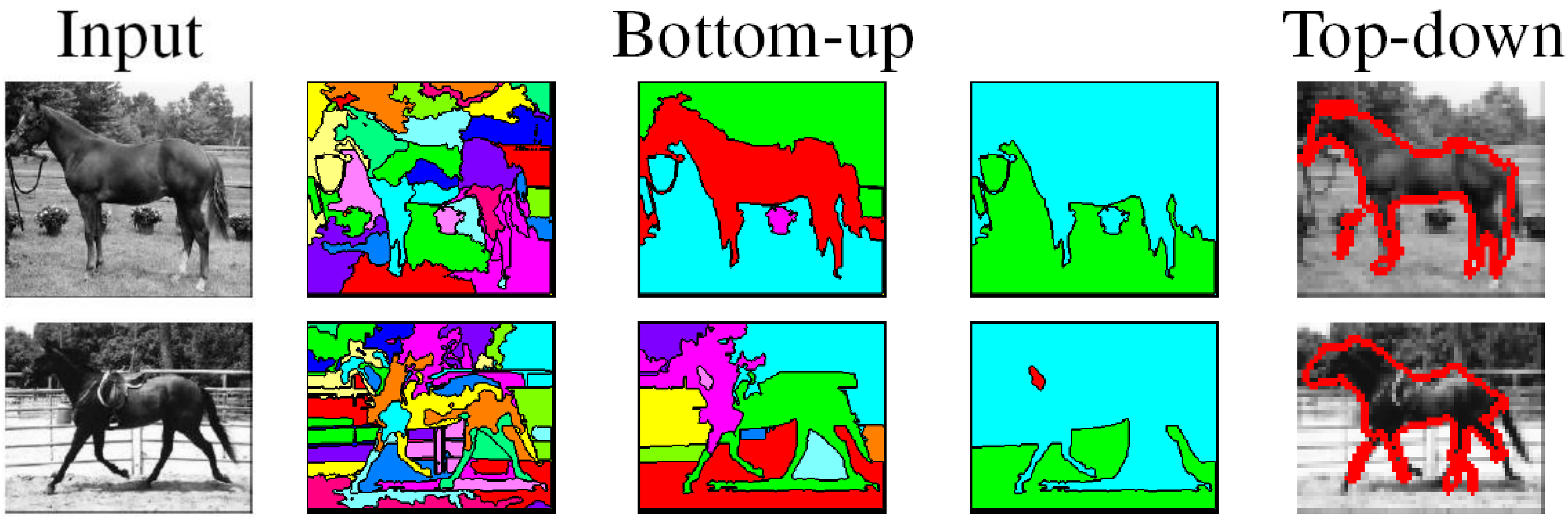




Multiple Segmentations

Major processes for segmentation

- Bottom-up
  - Group pixels with similar features
- Top-down
  - Group pixels that likely belong to the same object



[Levin and Weiss 2006]

Clustering

- Group together similar points and represent them with a single category.

- Key Challenges:
  - What makes two points/images/patches similar?
  - How do we compute an overall grouping from pairwise similarities?

How do we cluster?

- K-means
  - Iteratively re-assign points to nearest cluster center
- Single-link clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of PDF

Clustering for Summarization

- **A cluster**: a “center” (in feature space) and a list of data points it contains.
- Goal:
  - Minimize variance in data given clusters
  - Preserve information

Cluster Center

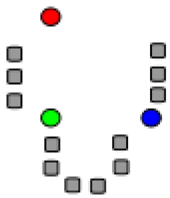
Data

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

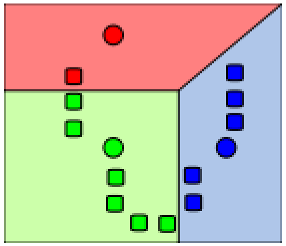
Whether x_j is assigned to c_i

K-means algorithm

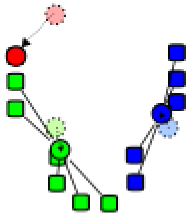
1. Randomly select K centers (means)



2. Assign each point to nearest center (mean)



3. Compute new center (mean) for each cluster

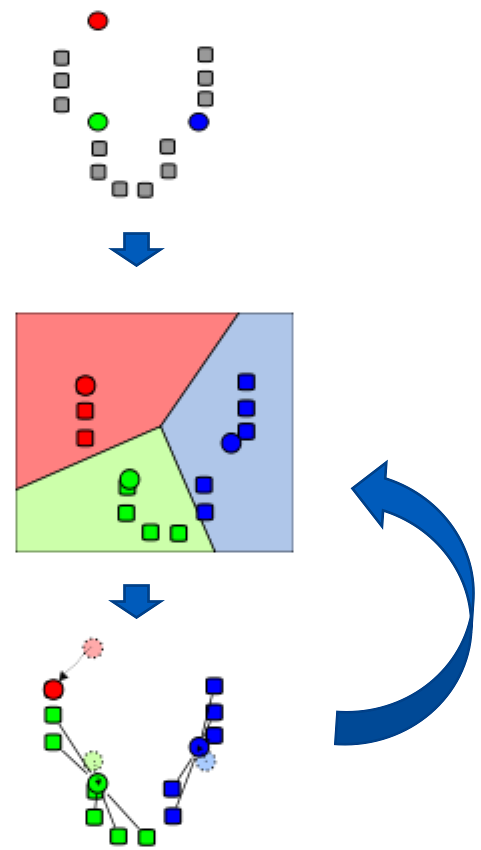


K-means algorithm

1. Randomly select K centers

2. Assign each point to nearest center

3. Compute new center (mean) for each cluster



Back to 2

Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means algorithm

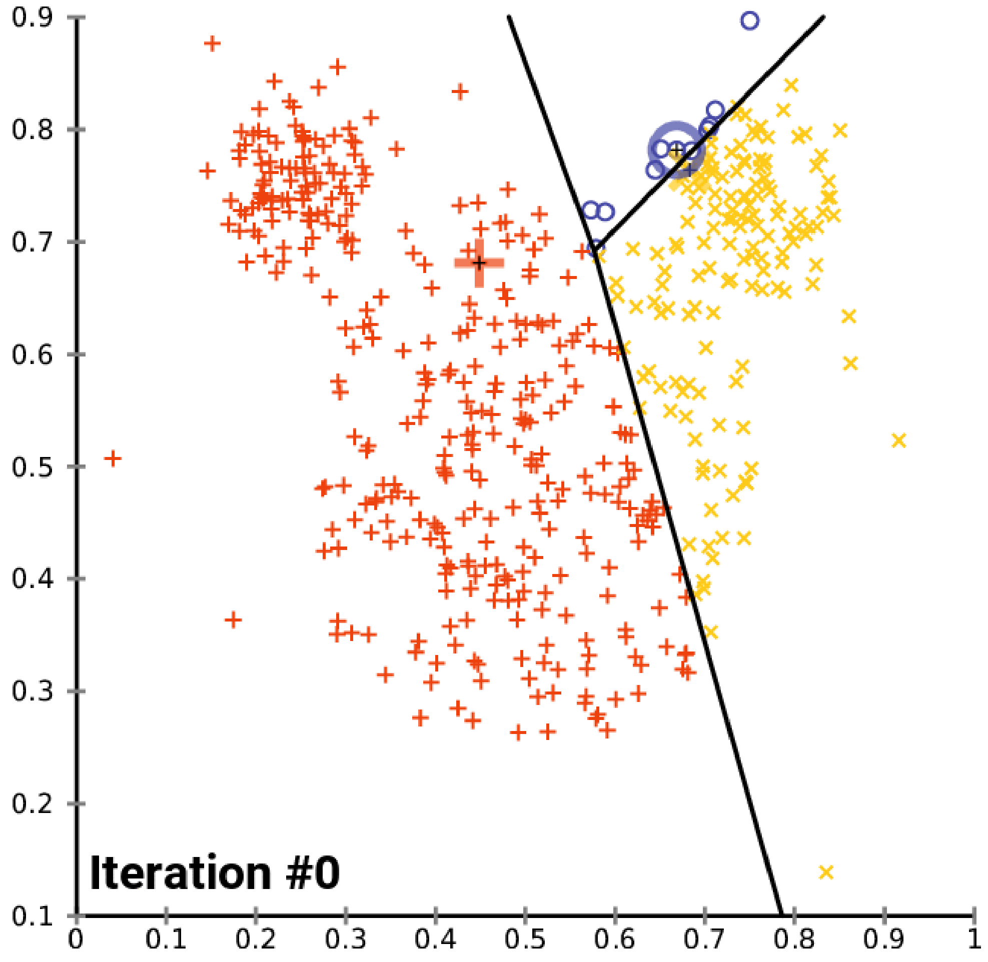


Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means algorithm

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

2. Assign each point to the closest center

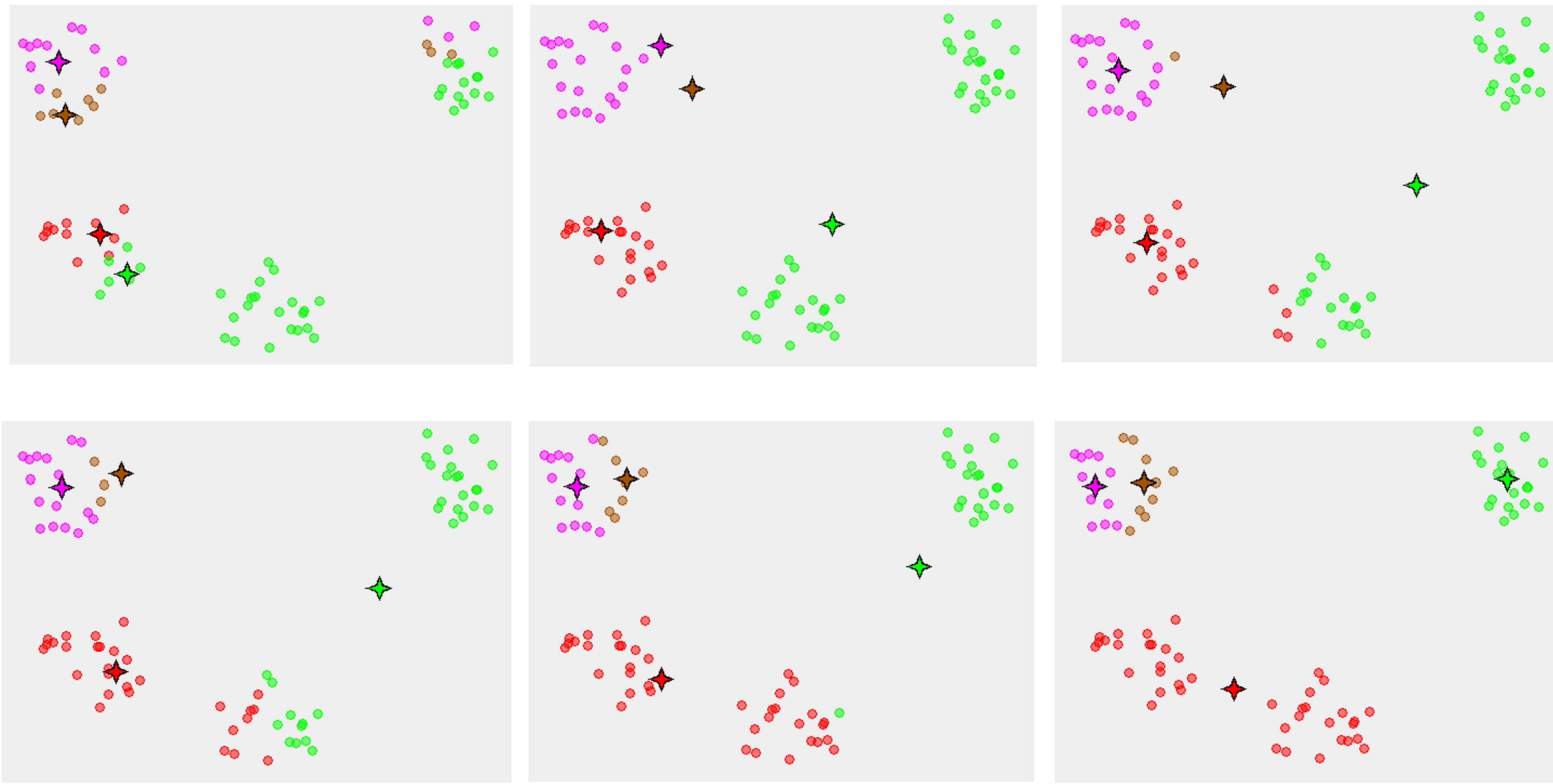
$$\boldsymbol{\delta}^t = \underset{\boldsymbol{\delta}}{\operatorname{argmin}} \ \tfrac{1}{N} \sum_j^N \sum_i^K \delta_{ij} \left(\mathbf{c}_i^{t-1} - \mathbf{x}_j\right)^2$$

3. Update cluster centers as the mean of the points

$$\mathbf{c}^t = \underset{\mathbf{c}}{\operatorname{argmin}} \ \tfrac{1}{N} \sum_j^N \sum_i^K \delta_{ij}^t \left(\mathbf{c}_i - \mathbf{x}_j\right)^2$$

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

K-means may converge to local minimum



K-means: design choices

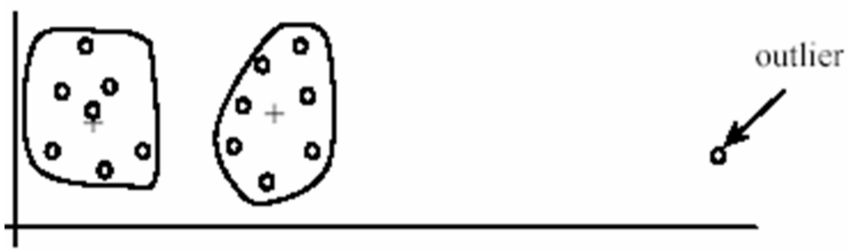
- Initialization
  - Randomly select K points as initial cluster center
  - Or greedily choose K points to minimize residual

- Distance measures
  - Traditionally Euclidean, could be others

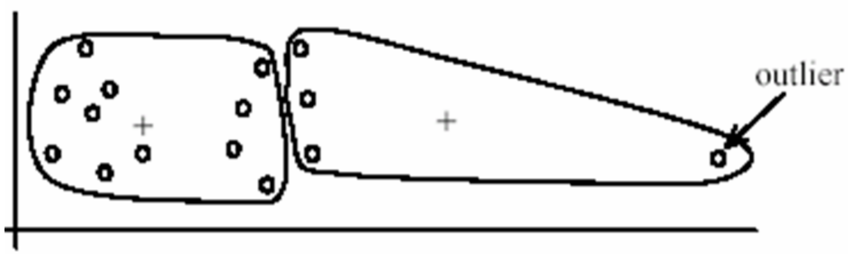
- Optimization
  - May converge to a *local minimum*
  - May want to perform **multiple restarts**

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



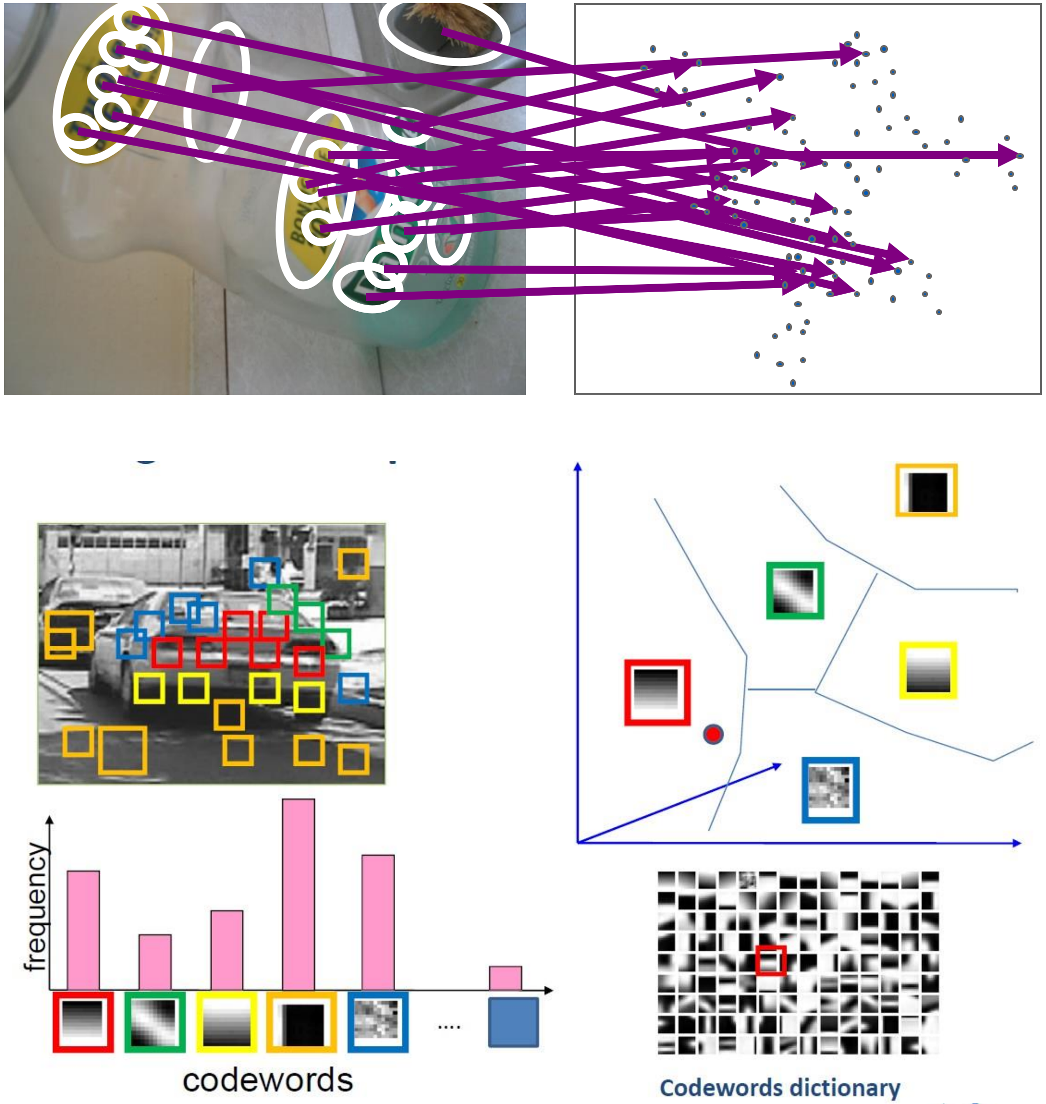
How to choose the number of clusters?

- On validation set

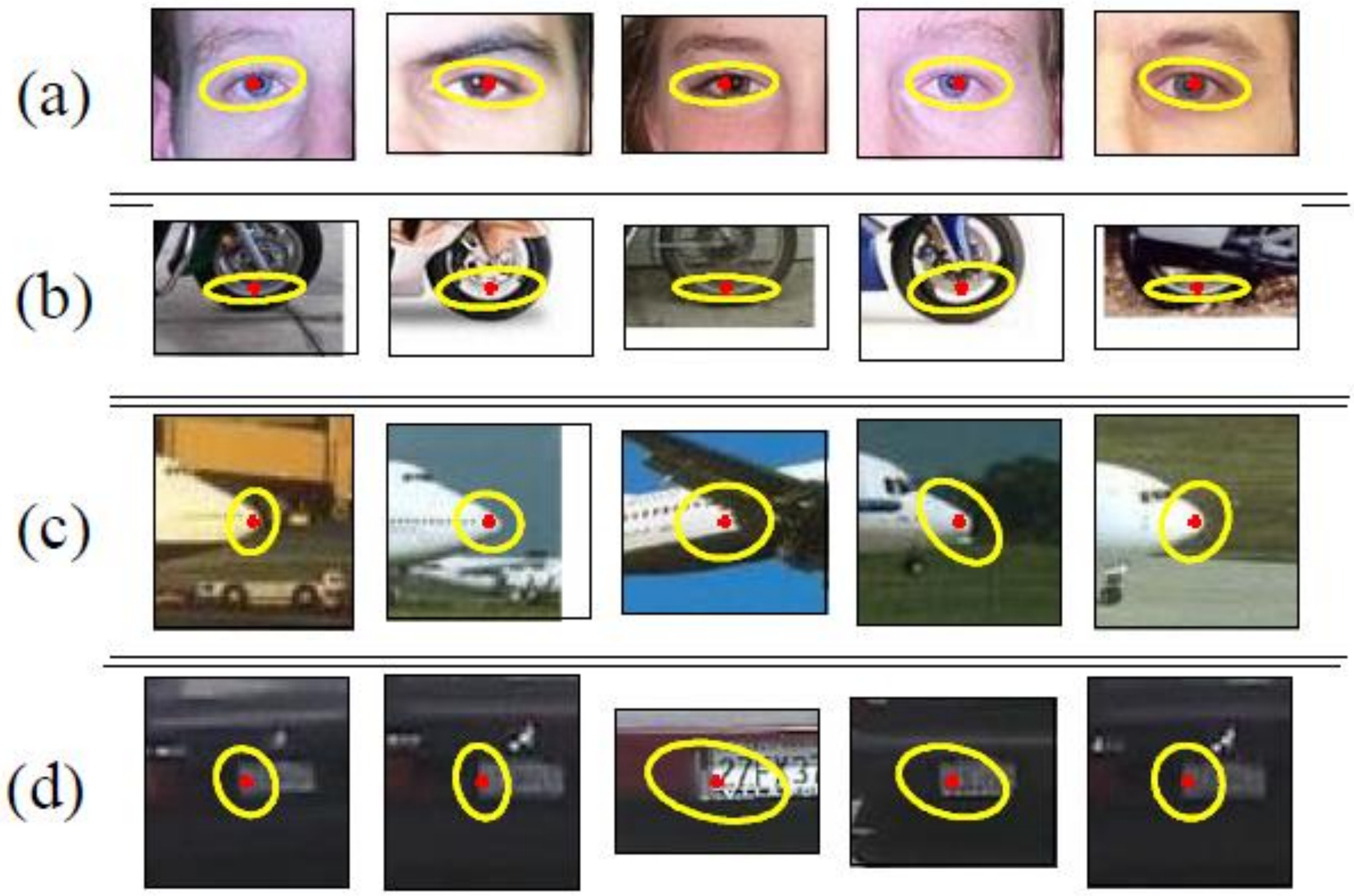
- Try different numbers of clusters and look at performance
  - When building dictionaries, more clusters typically work better

Building Visual Dictionaries

1. Sample patches from a database
   – E.g., 128 dim SIFT
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

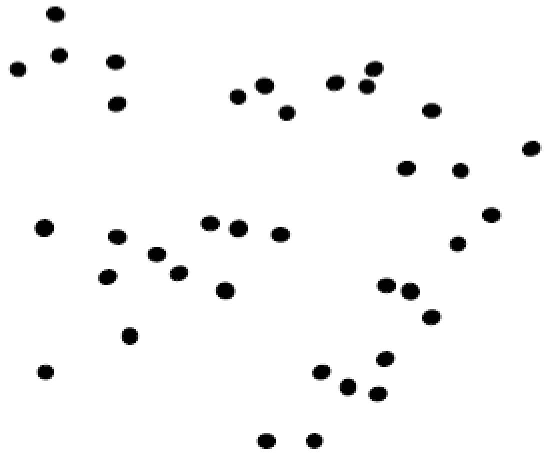


Sivic et al. ICCV 2005

How do we cluster?

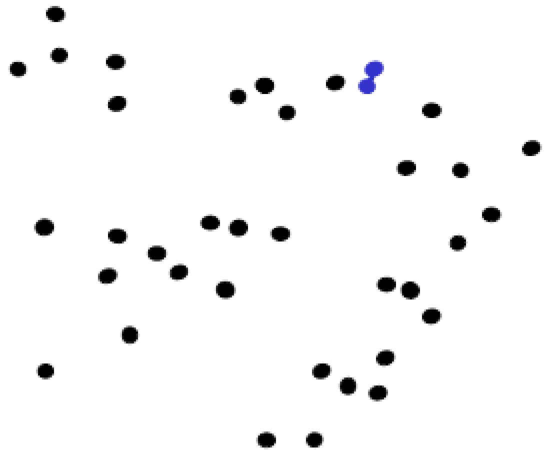
- K-means
  - Iteratively re-assign points to the nearest cluster center
- Single-link clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of PDF
- Spectral clustering
  - Split the nodes in a graph-based on assigned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

Single Link Clustering



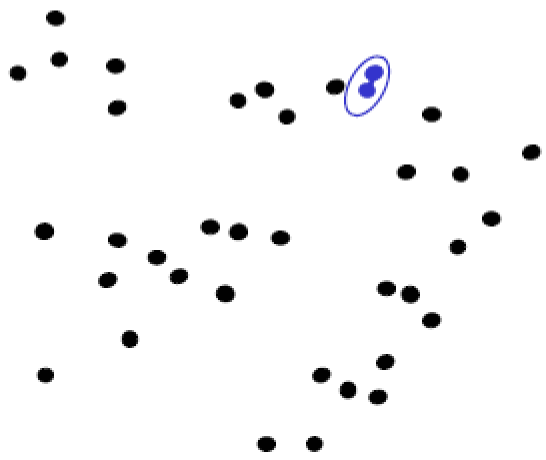
1. Say "Every point is its own cluster"

Single Link Clustering



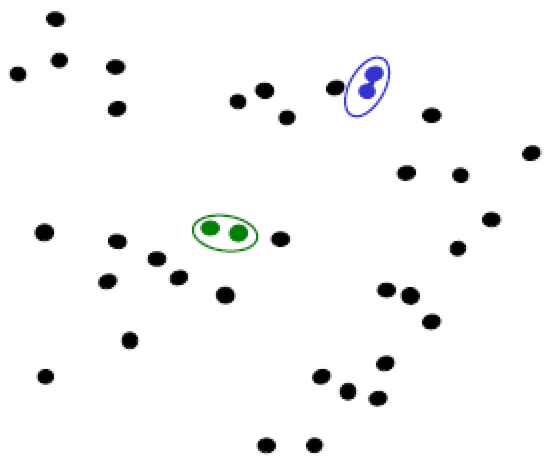
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters

Single Link Clustering



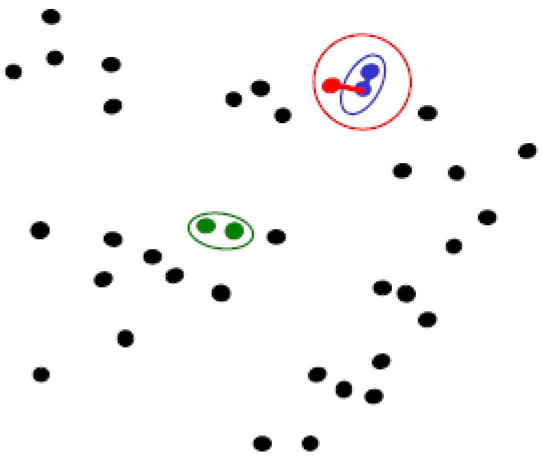
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster

Single Link Clustering

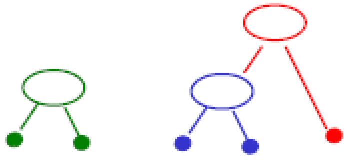


1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

Single Link Clustering

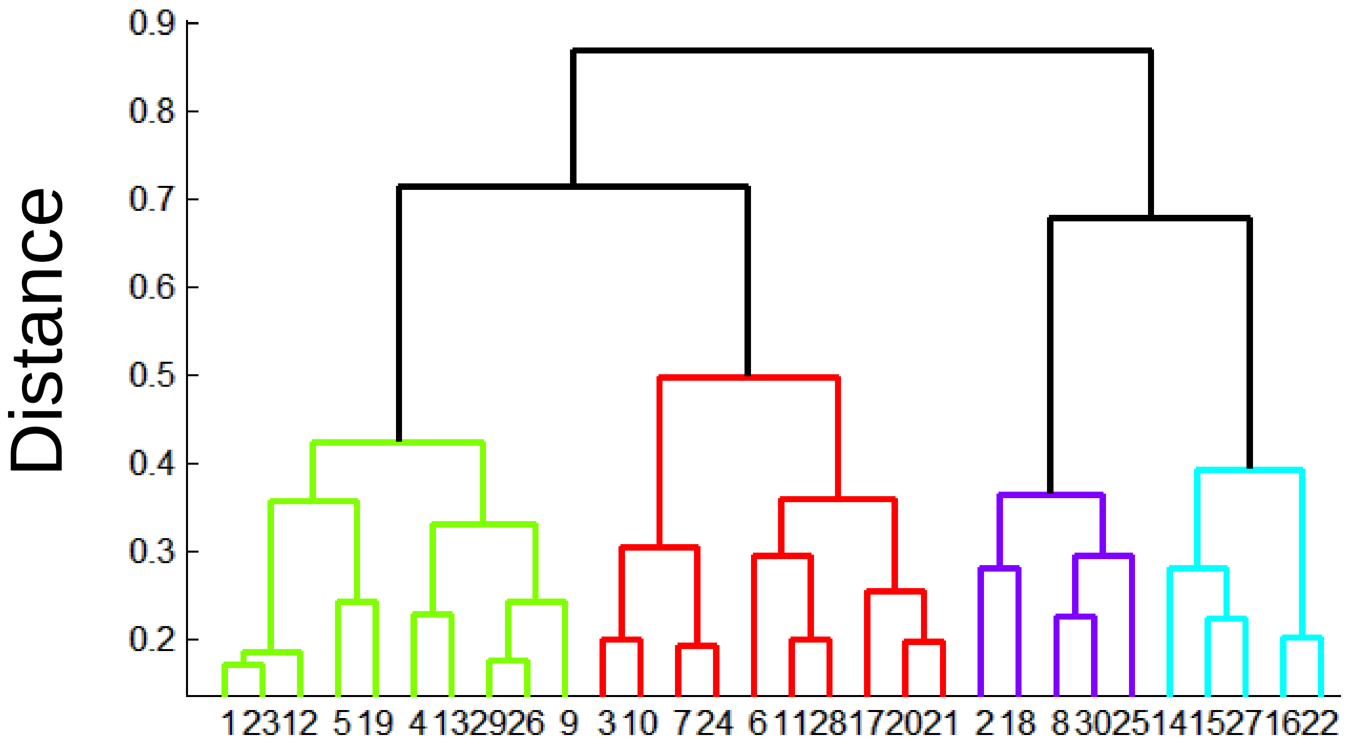


1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat



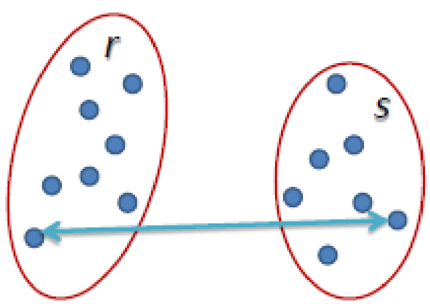
Single Link Clustering

- How many clusters?
  - Threshold based on max number of clusters or based on distance between merges

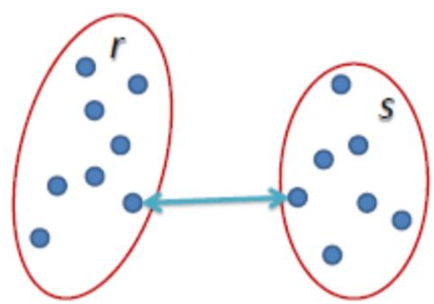


Single Link Clustering

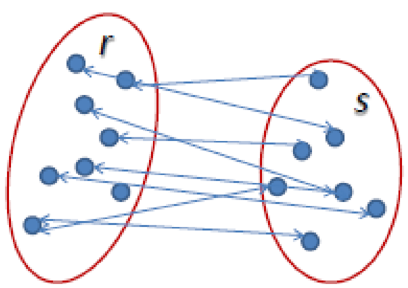
- How to define cluster similarity?
  - Average/maximum/minimum distance of points
  - Distance between means.



$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$



$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$



$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$

Summary: Single Link Clustering

Good

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

Bad

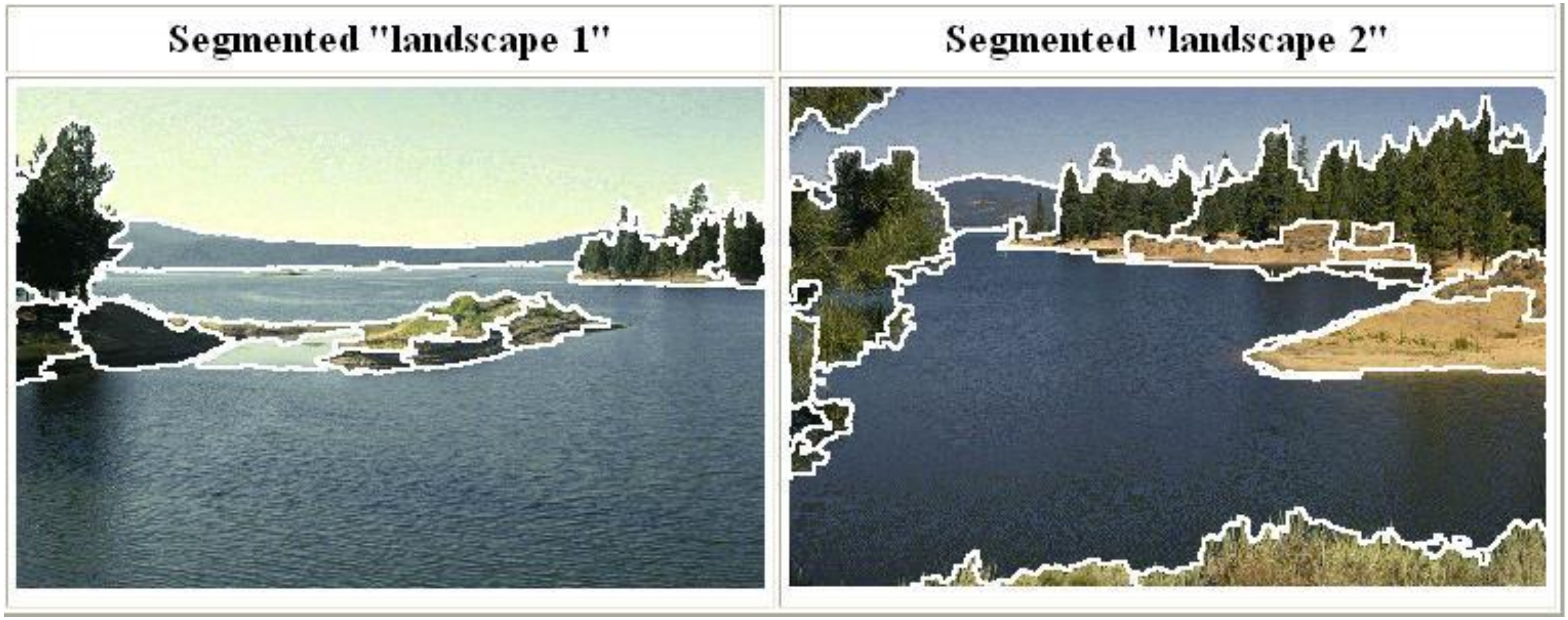
- May have **imbalanced** clusters
- Still have to choose number of clusters or threshold
- Need to use an “ultrametric” to get a meaningful hierarchy

How do we cluster?

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Single-link 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

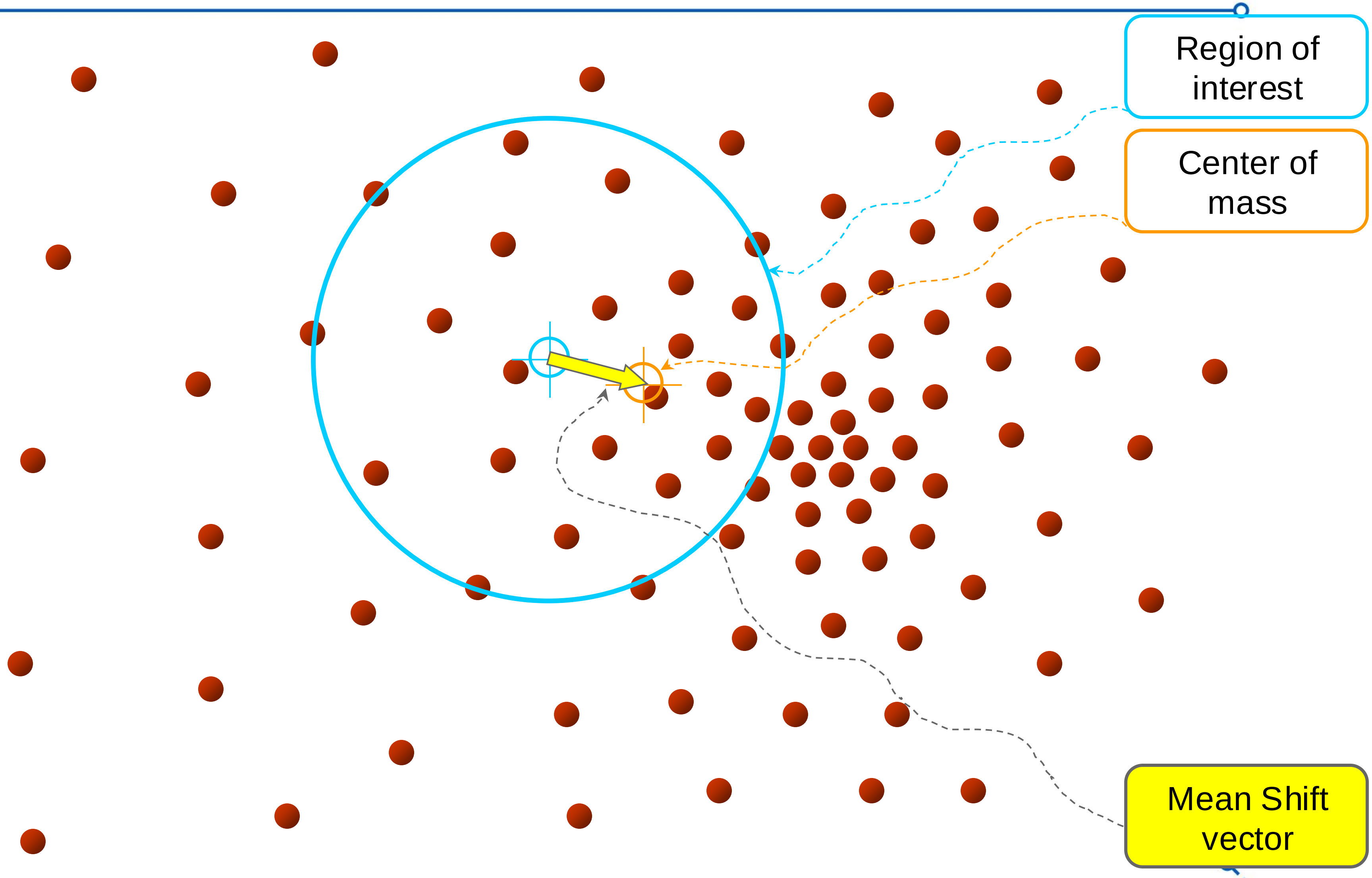
Mean shift algorithm

- Non-parametric feature-space analysis for locating the maxima of a density function.
- Versatile technique for clustering-based segmentation.

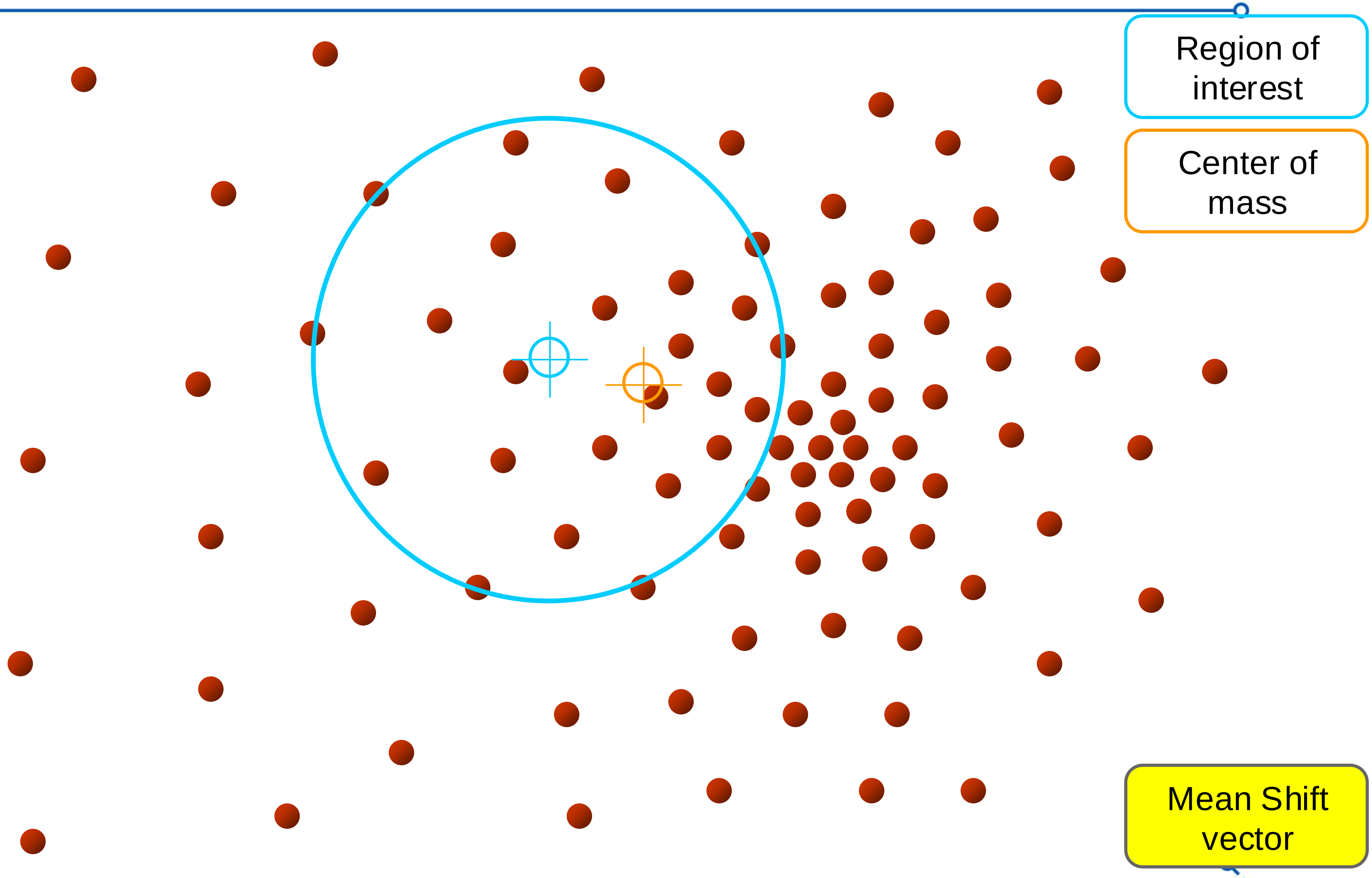


D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002.

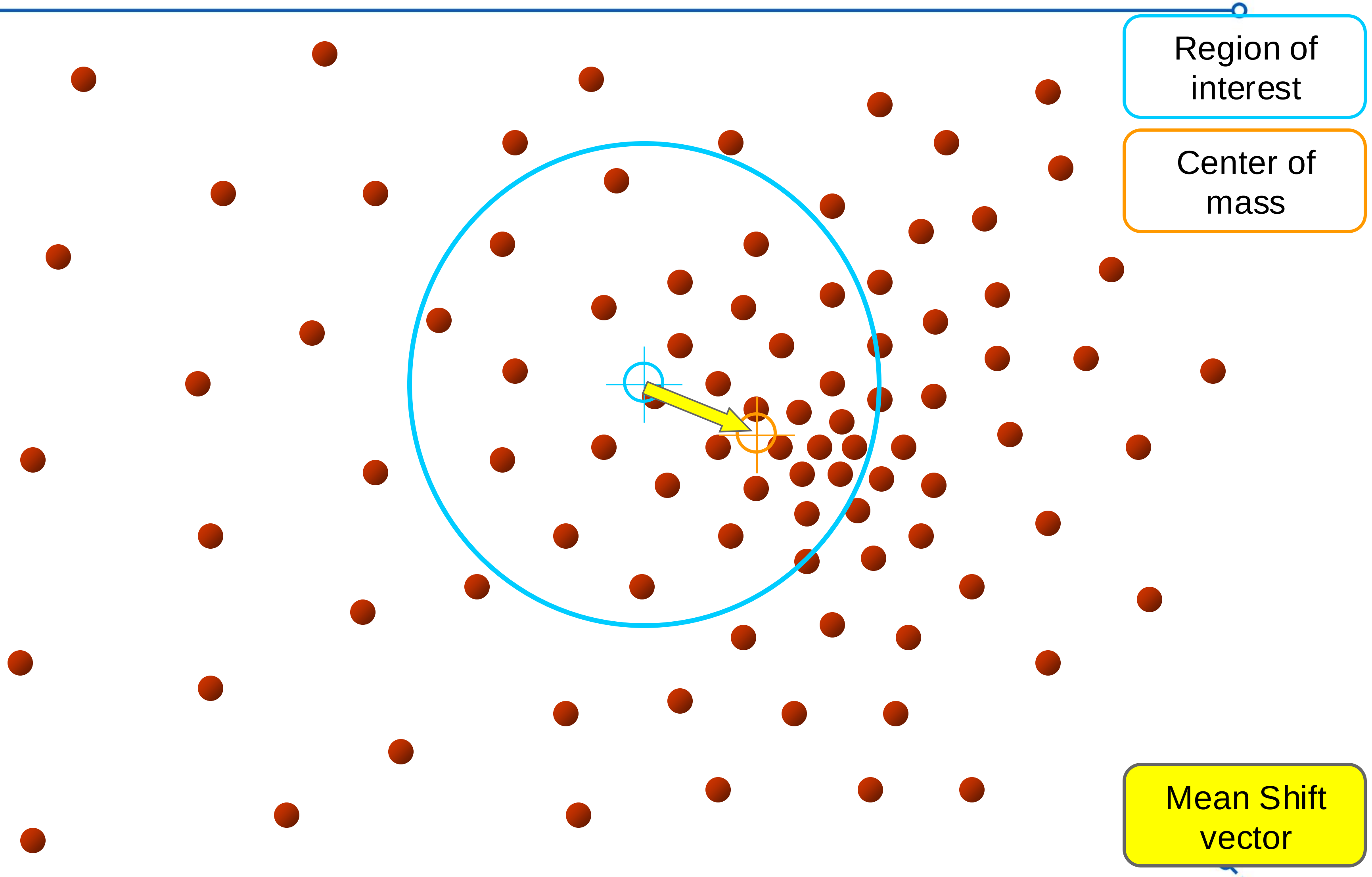
Mean shift



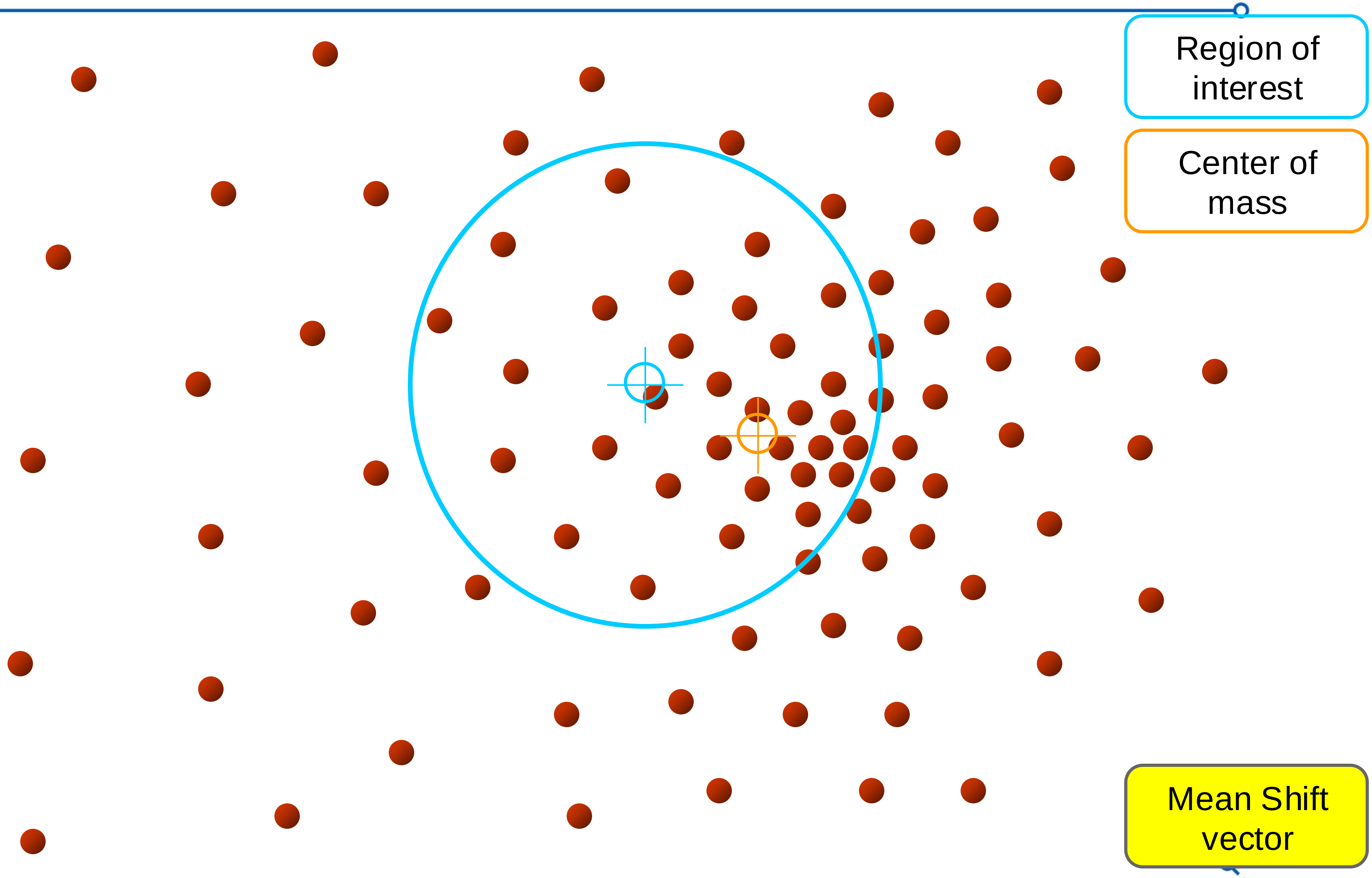
Mean shift



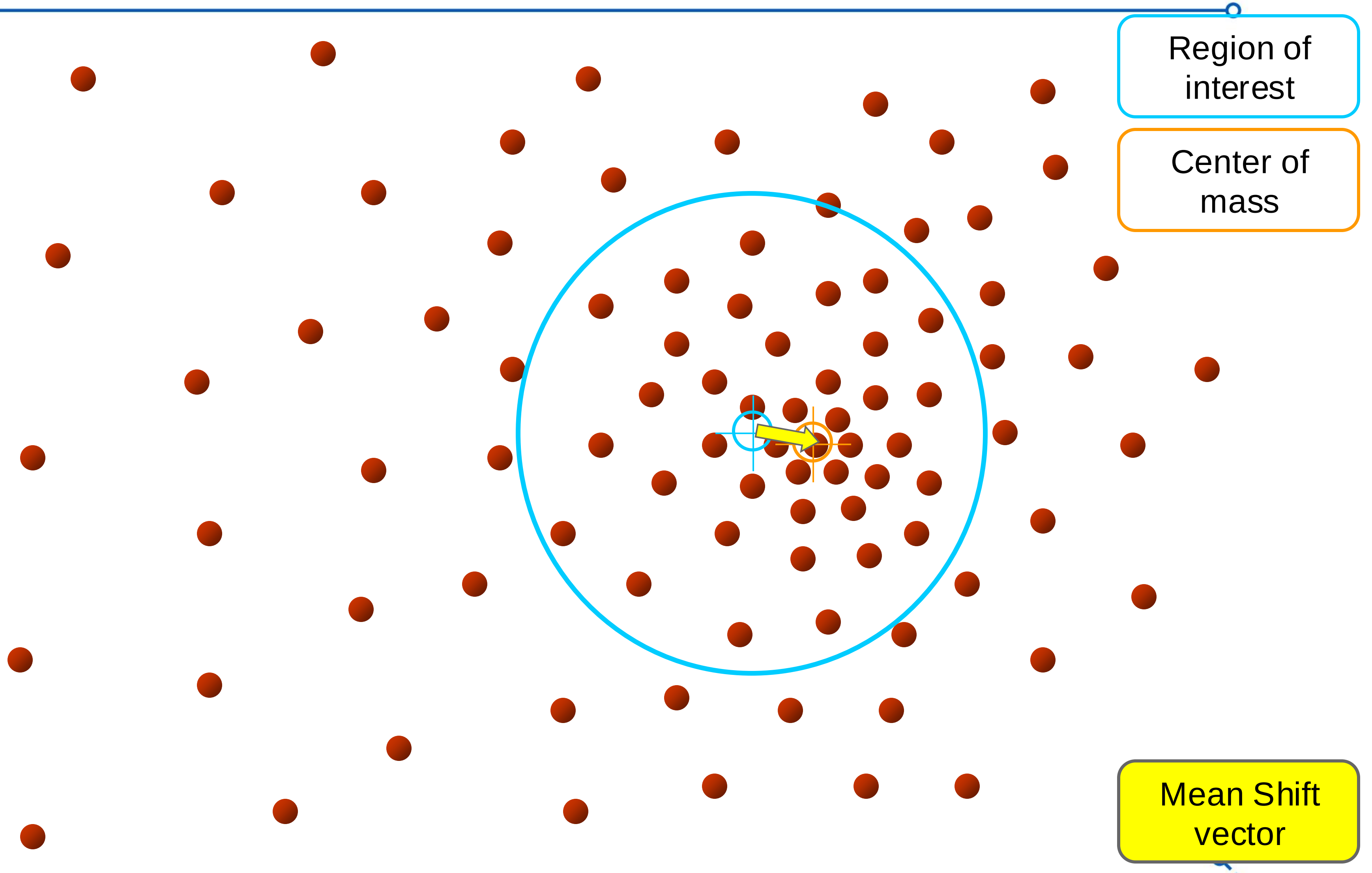
Mean shift



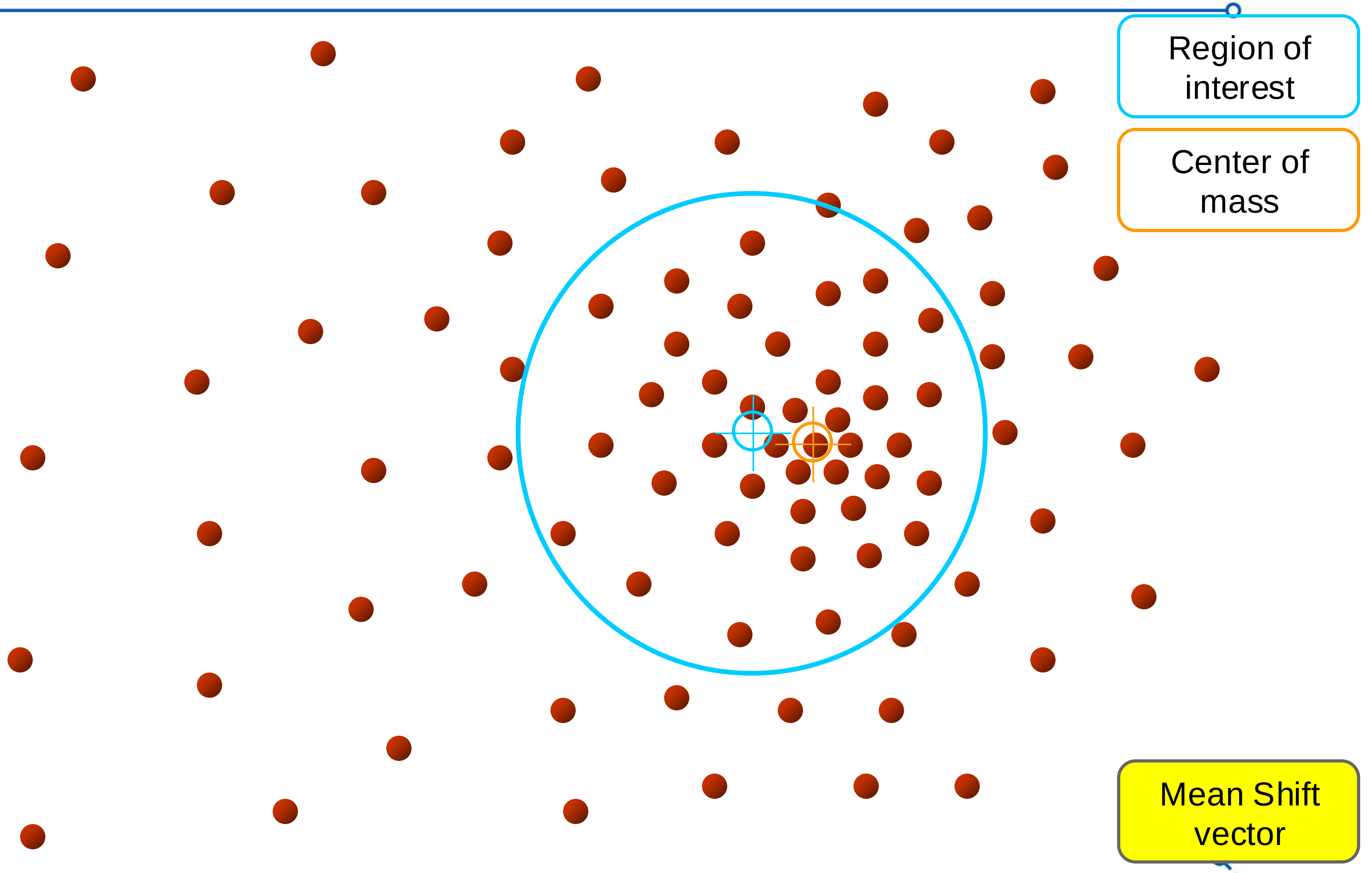
Mean shift



Mean shift

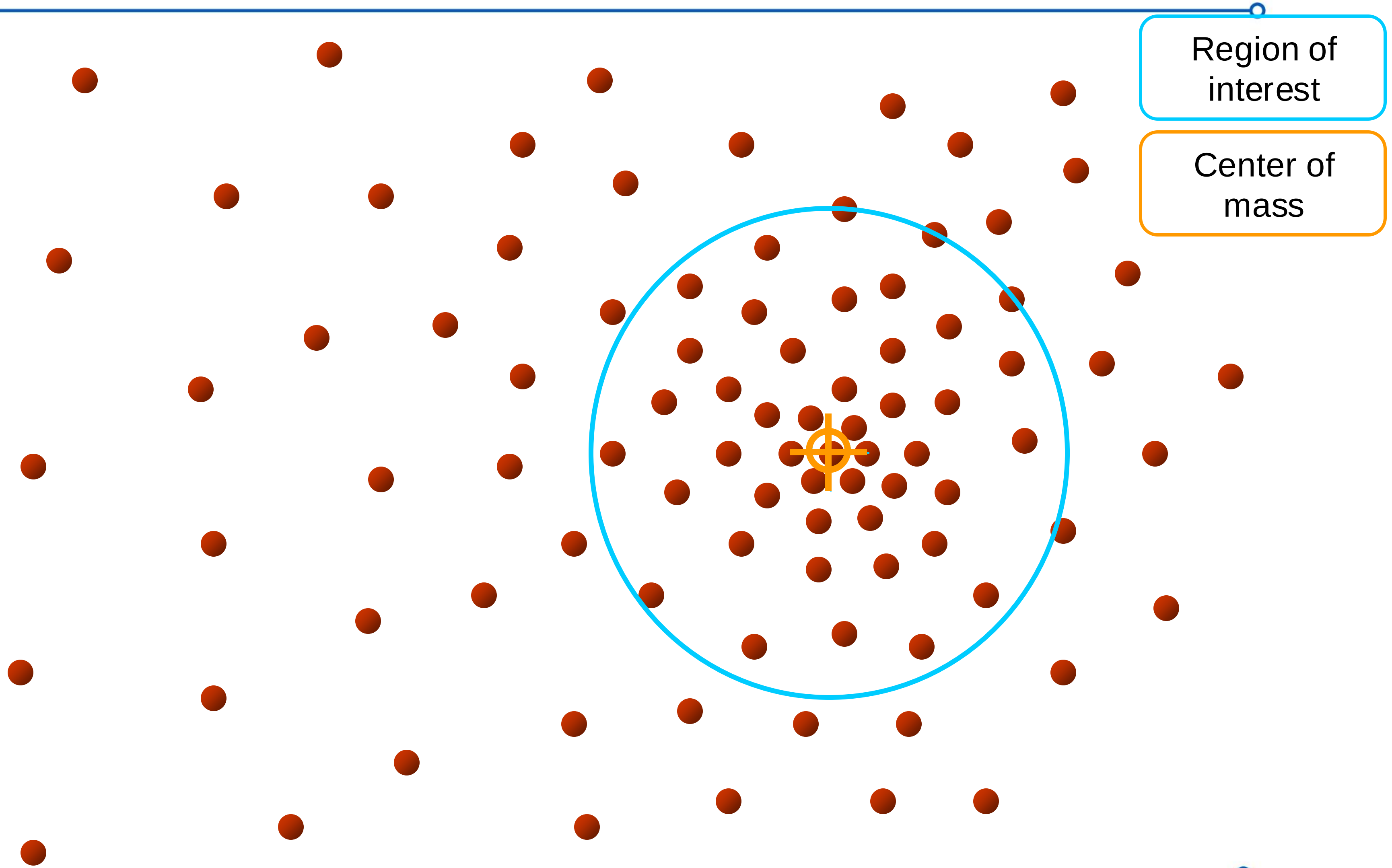


Mean shift



Slide by Y. Ukrainitz & B. Sarel

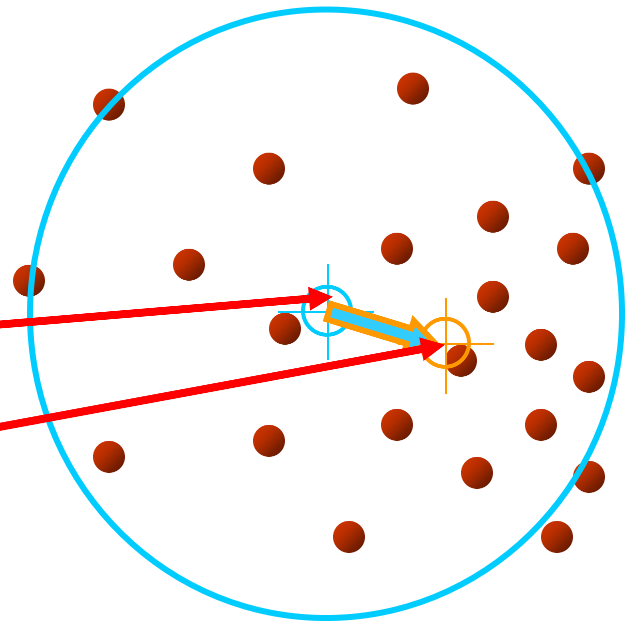
Mean shift



Computing the Mean Shift

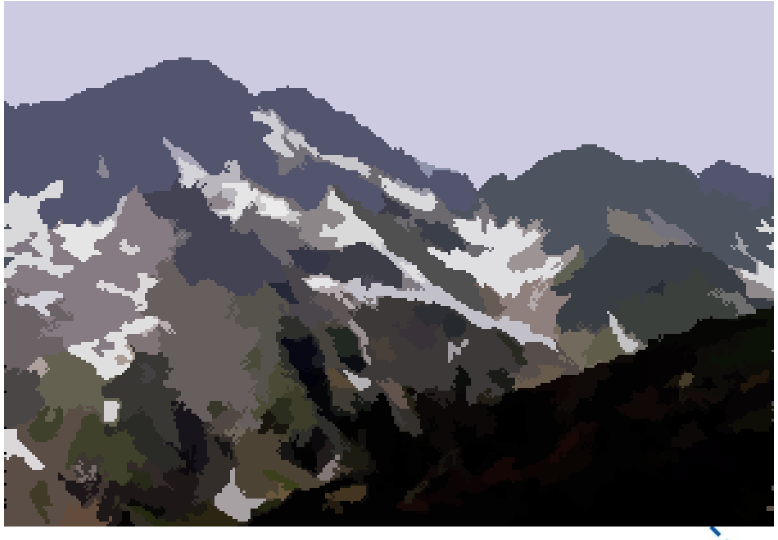
- Compute mean shift vector.
- Translate mean by m(x), weighted by kernel function.

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



$$g(x_i - x) = e^{-c\|x_i - x\|^2}$$

Mean shift segmentation results



Comaniciu and Meer 2002

Mean shift segmentation results

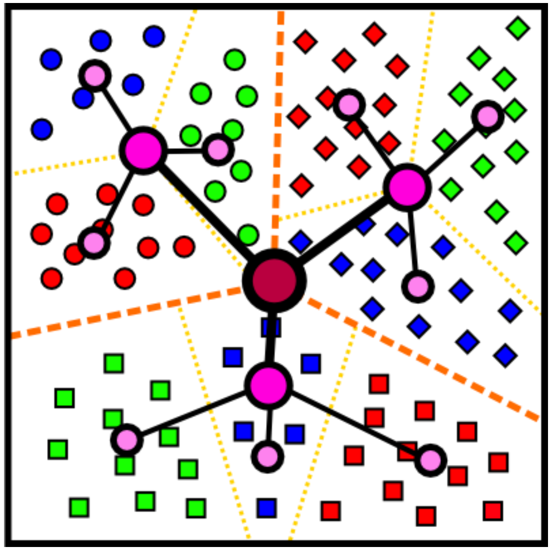
Comaniciu and Meer 2002

Mean shift pros and cons

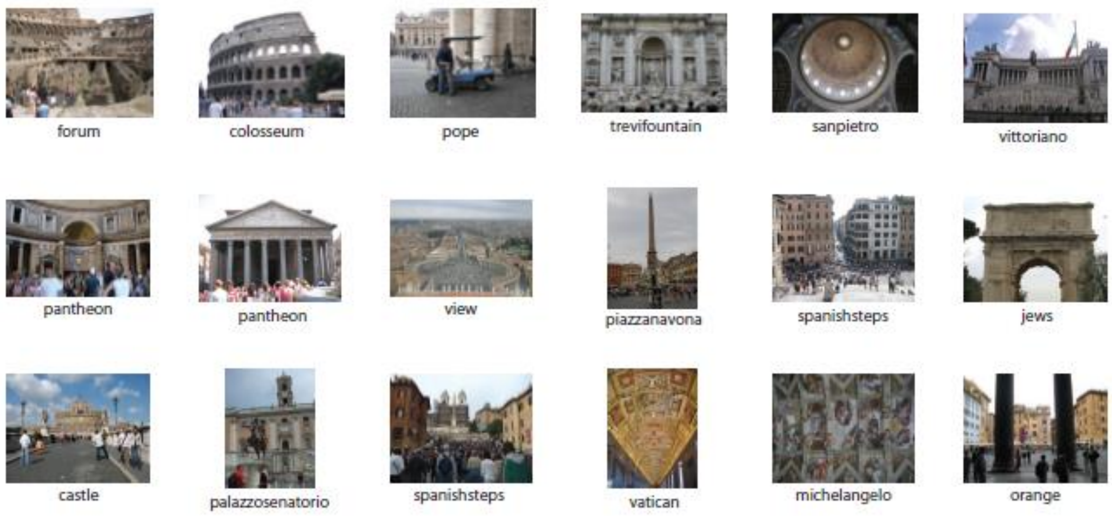
- Pros
  - Good general-practice segmentation
  - Flexible in number and shape of regions
  - Robust to outliers
- Cons
  - Have to choose kernel size in advance
    - Use kNN to determine window sizes adaptively
  - Not suitable for high-dimensional features
- When to use it
  - Oversegmentation, Multiple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

Which algorithm to use?

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



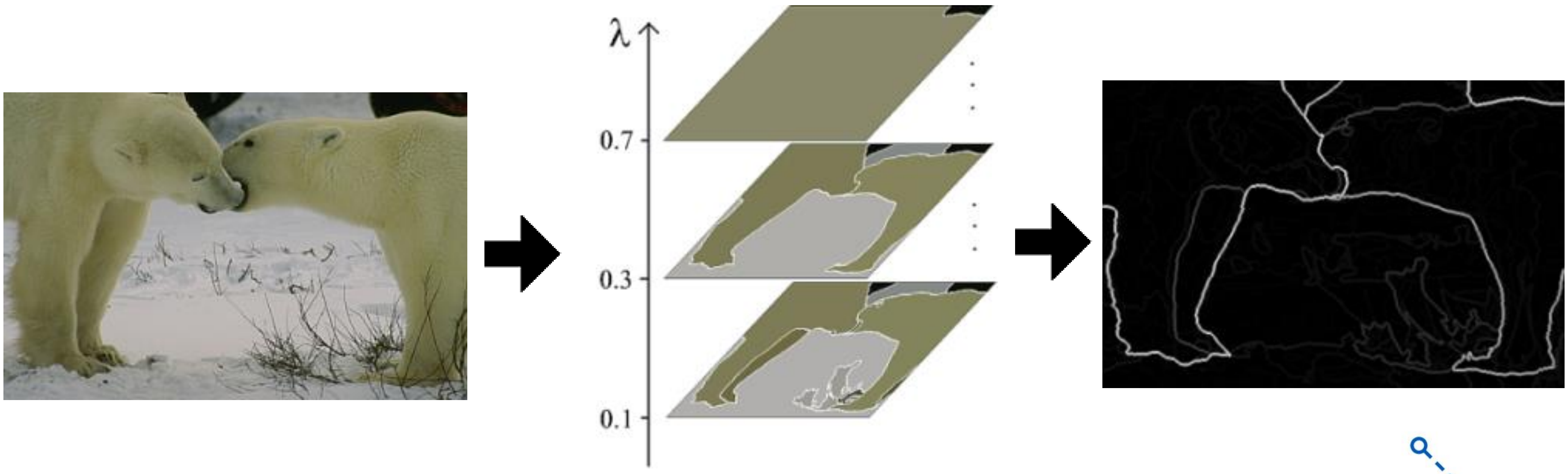
Quantization for computing histograms



[Summary of 20,000 photos of Rome using "greedy k-means"]

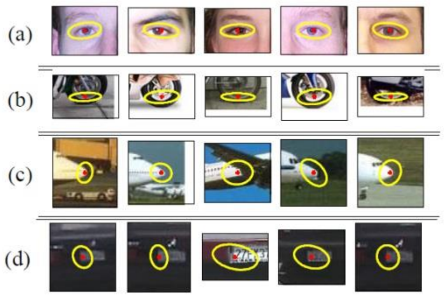
Which algorithm to use?

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



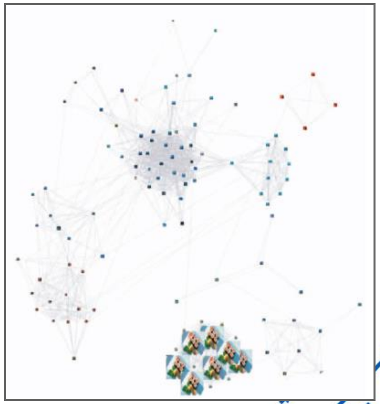
http://www.cs.berkeley.edu/~arbelaez/UCM.html

Things to remember



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

- Single link clustering useful for segmentation, general clustering



- Mean-shift is useful for determining relevance, summarization, segmentation

Application: Motion Segmentation