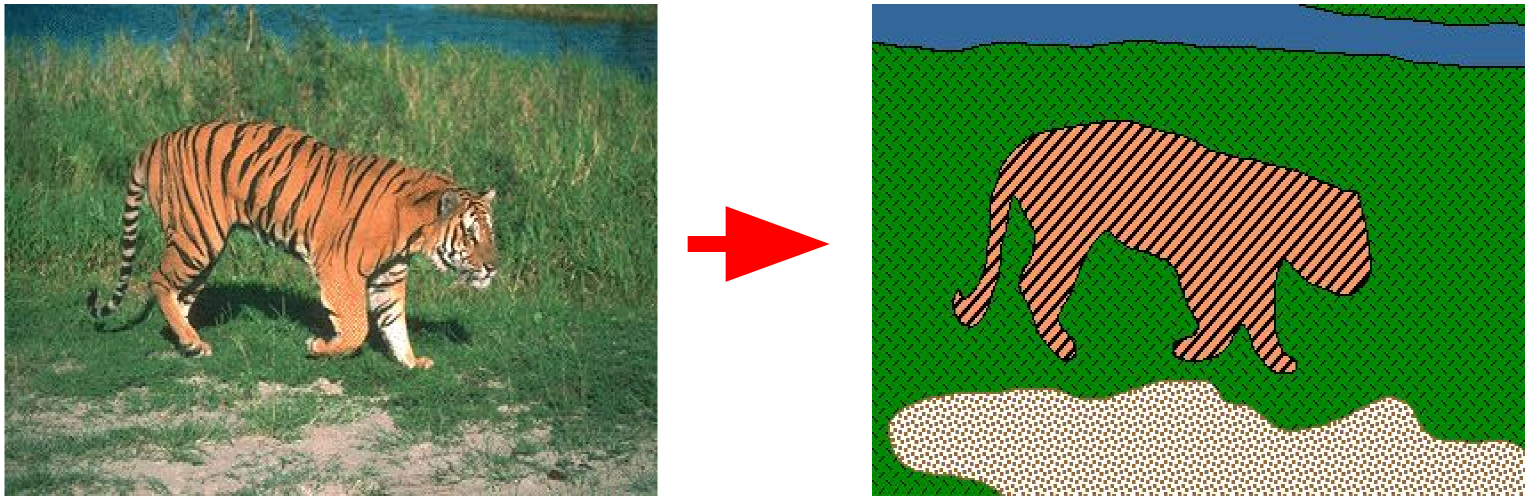
Lecture 13: k-means and mean-shift clustering

Juan Carlos Niebles
Stanford AI Lab

Professor Fei-Fei Li
Stanford Vision Lab

Recap: Image Segmentation

- Goal: identify groups of pixels that go together

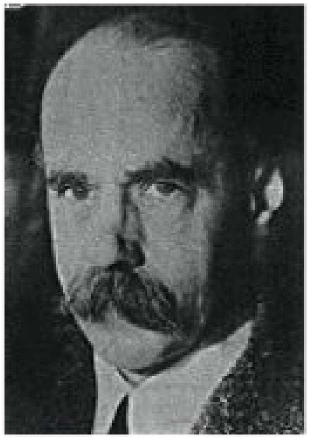


Recap: Gestalt Theory

- Gestalt: whole or group (German: "shape, form")
  - Whole is *other* than sum of its parts
  - Relationships among parts can yield new properties/features
- Psychologists identified series of factors that predispose set of elements to be grouped (by human visual system)

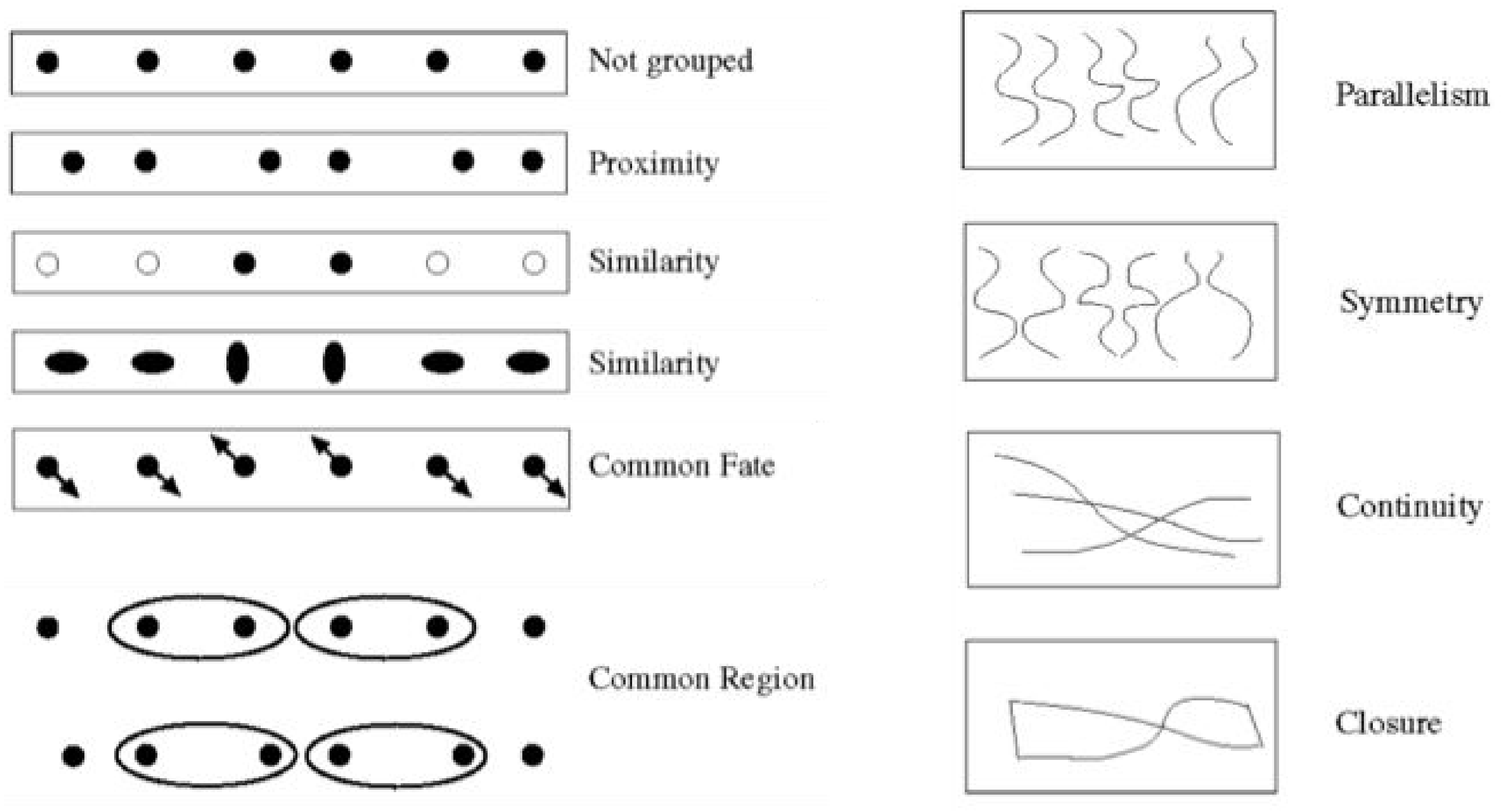
"I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have "327"? No. I have sky, house, and trees."

Max Wertheimer
(1880-1943)



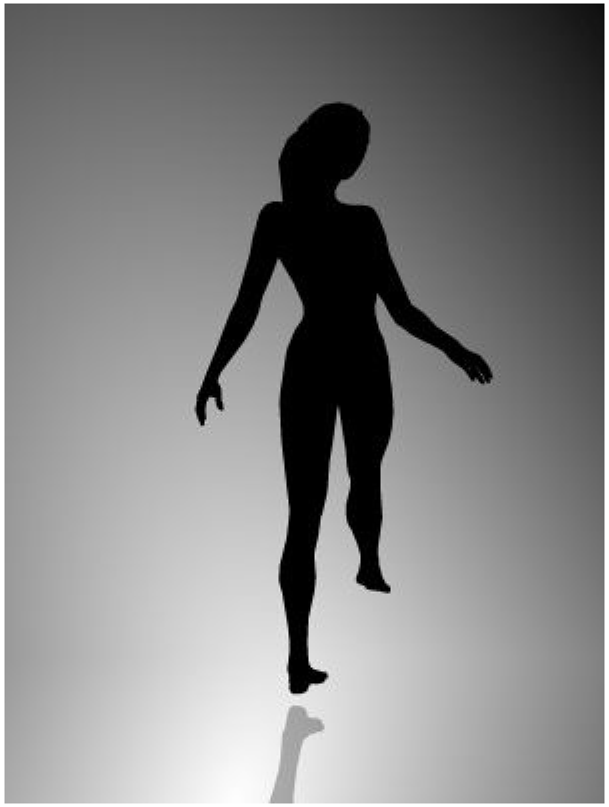
Untersuchungen zur Lehre von der Gestalt,
***Psychologische Forschung*, Vol. 4, pp. 301-350, 1923**

Recap: Gestalt Factors



- These factors make intuitive sense, but are very difficult to translate into algorithms.

Recap: Multistability



https://en.wikipedia.org/wiki/Spinning_Dancer

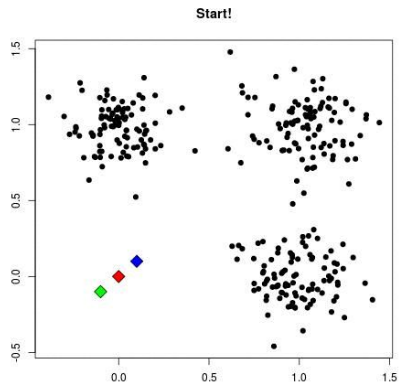
Recap: Agglomerative clustering

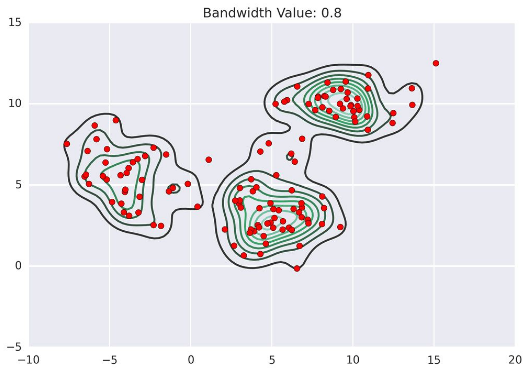
Simple algorithm

- Initialization:
    - Every point is its own cluster
- Repeat:
    - Find “most similar” pair of clusters
    - Merge into a parent cluster
- Until:
    - The desired number of clusters has been reached
    - There is only one cluster

What will we learn today?

- K-means clustering
- Mean-shift clustering

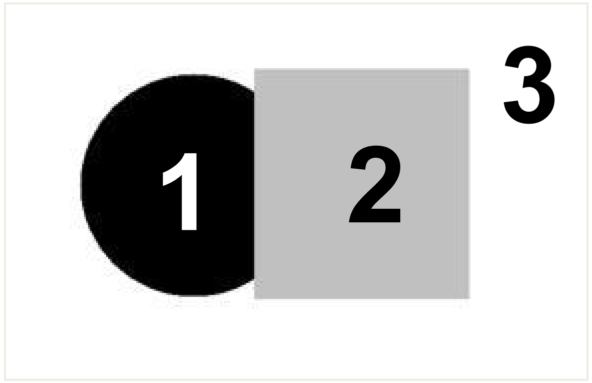




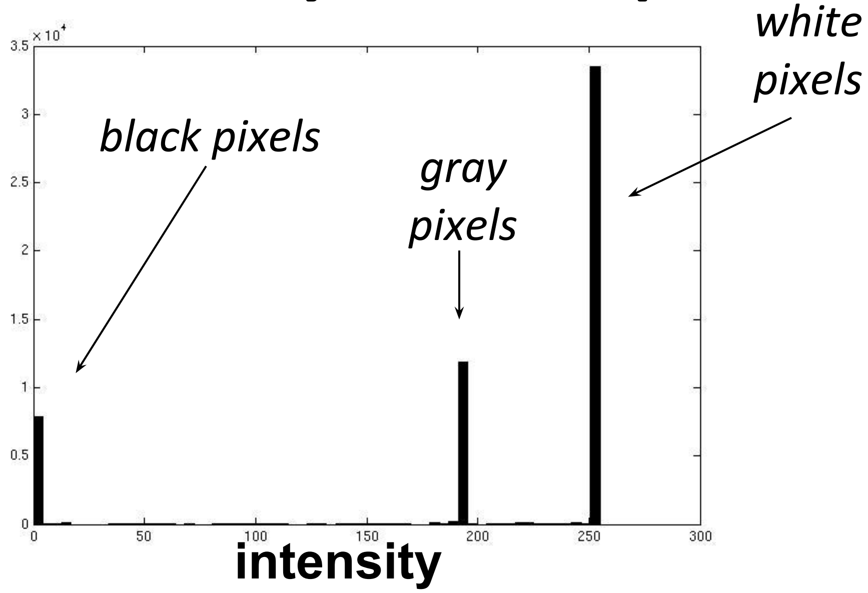
Reading material:
Forsyth & Ponce: Chapter 9.3
Comaniciu and Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002.

gifs: https://www.projectrhea.org

Image Segmentation: Toy Example

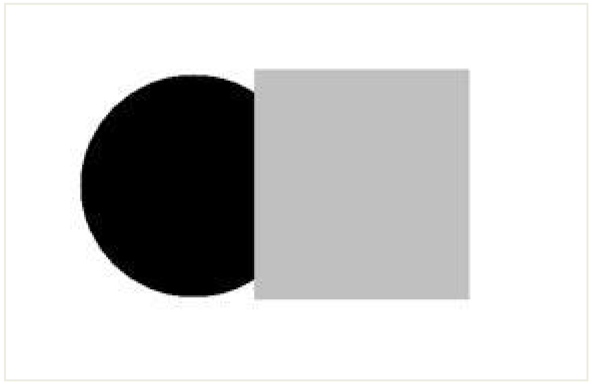


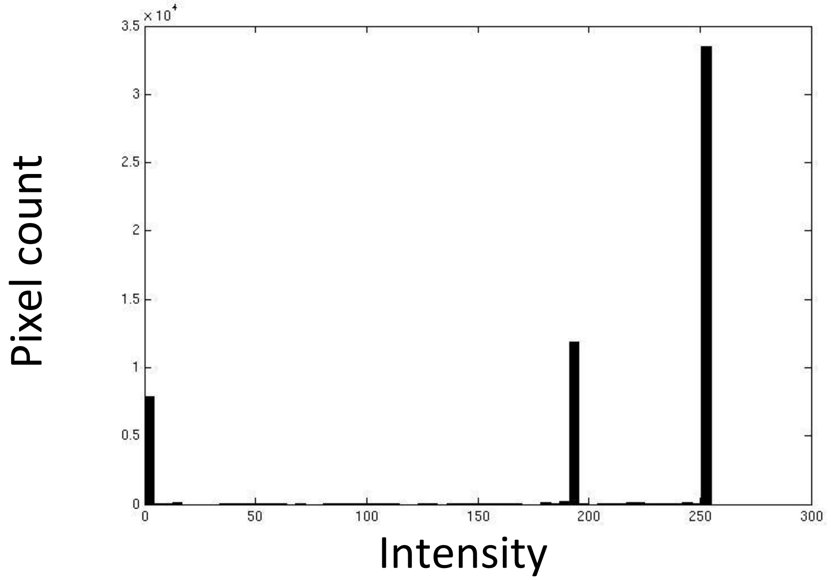
input image

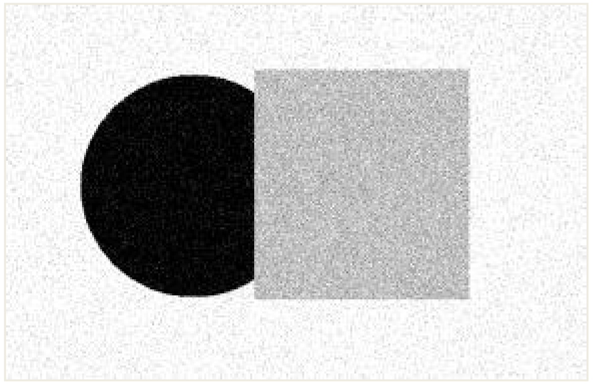


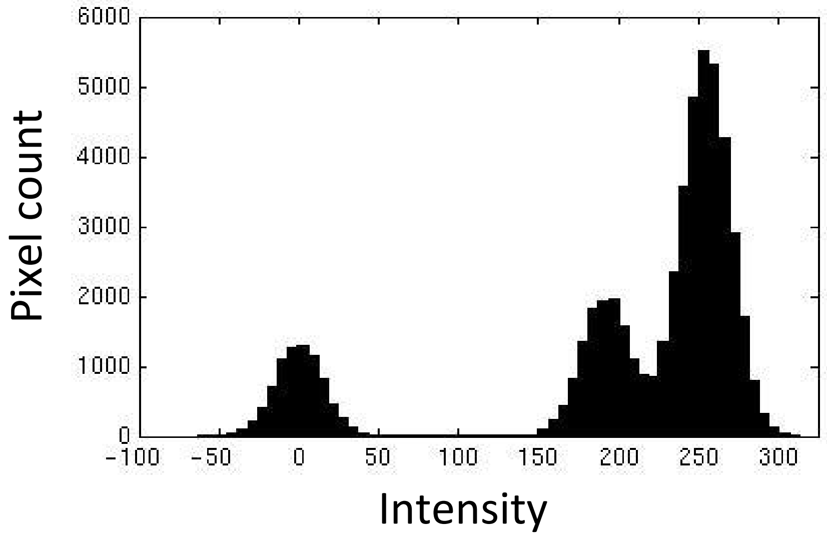
- These intensities define the
- We could label every pixel in the image according to which of these primary intensities it is.
  - i.e., segment the image based on the intensity feature.
- What if the image isn't quite so simple?

Slide credit: Kristen Grauman

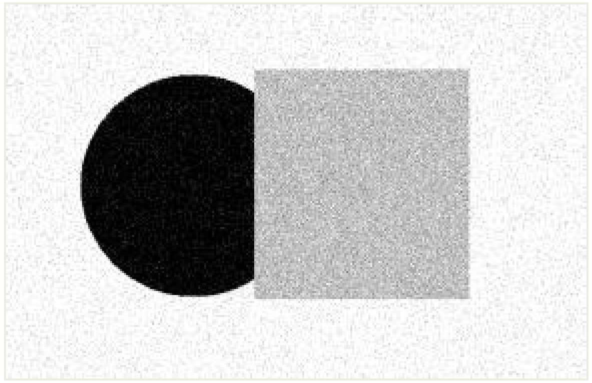

Input image

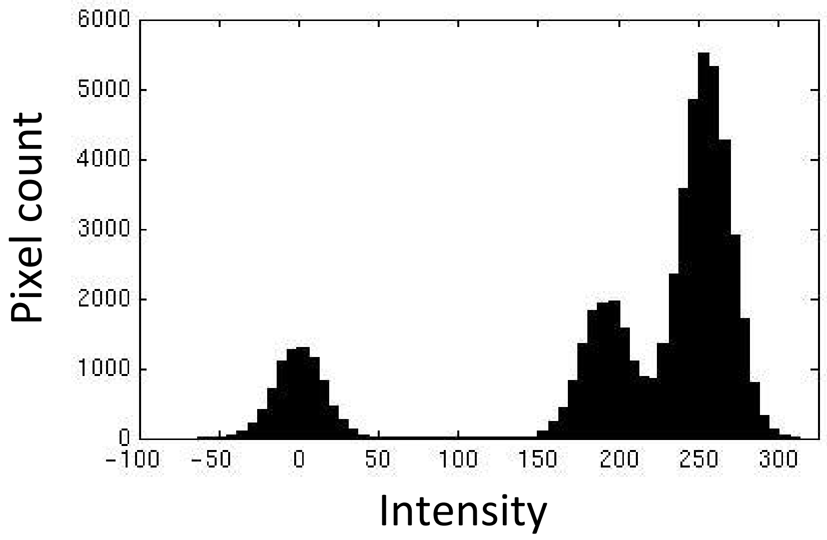



Input image



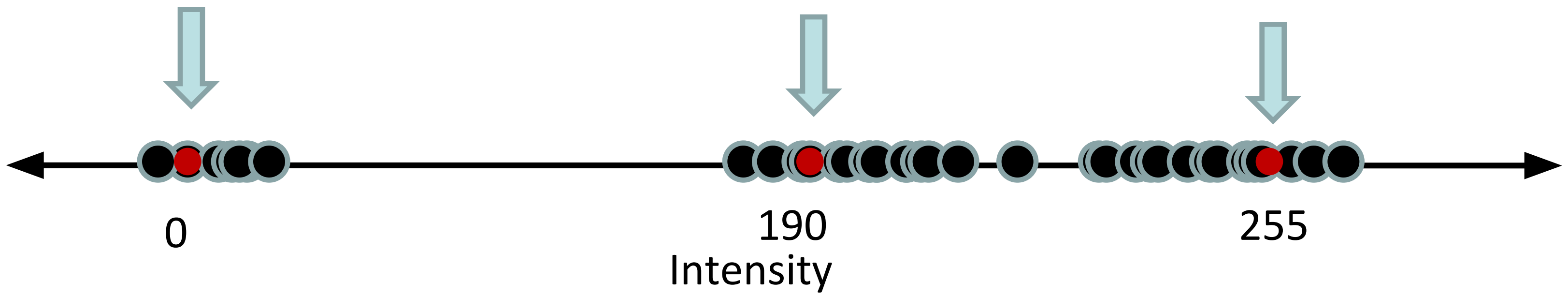
Slide credit: Kristen Grauman

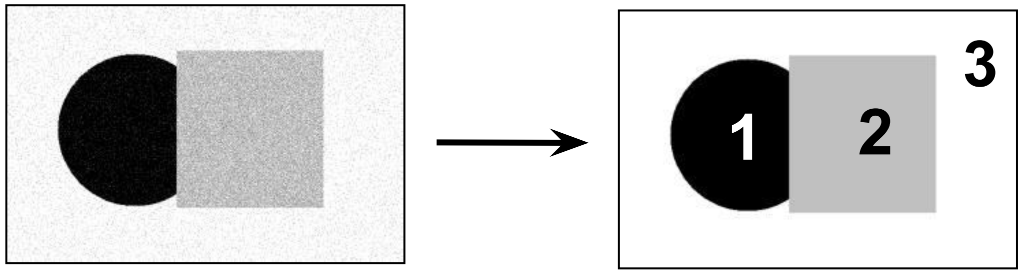

Input image



- Now how to determine the three main intensities that define our groups?
- We need to cluster.

Slide credit: Kristen Grauman





- Goal: choose three "centers" as the representative intensities, and label every pixel according to which of these centers it is nearest to.
- Best cluster centers are those that minimize Sum of Square Distance (SSD) between all points and their nearest cluster center c_i:

$$SSD = \sum_{i}^{k} \sum_{x \in c_i} (x - c_i)^2$$

Slide credit: Kristen Grauman

Objective function

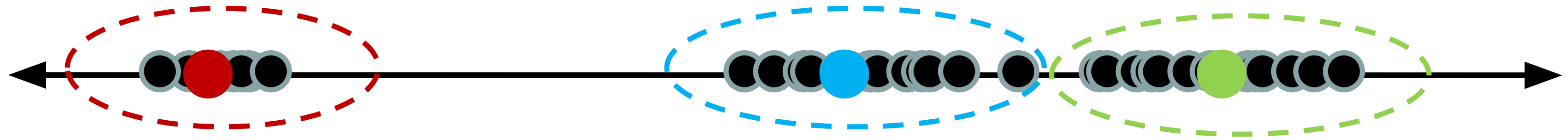
- Goal: minimize the distortion in data given clusters
  - Preserve information

$$c^*, \delta^* = \arg\min_{c,\delta} \frac{1}{N} \sum_{j}^{N} \sum_{i}^{k} \underset{\text{Whether } x_j \text{ is assigned to } c_i}{\delta_{ij}} (\overset{\text{Cluster center}}{c_i} - \overset{\text{Data}}{x_j})^2$$

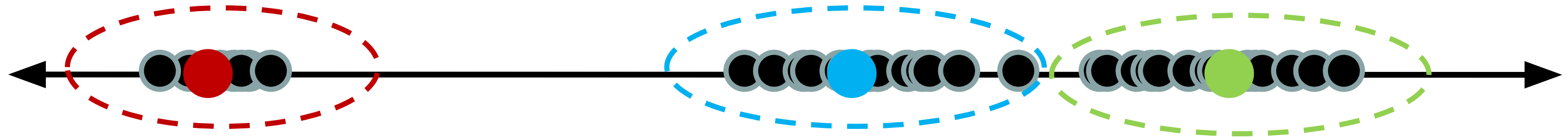
Slide: Derek Hoiem

Clustering

- With this objective, it is a "chicken and egg" problem:
  - If we knew the *cluster centers*, we could allocate points to groups by assigning each to its closest center.



  - If we knew the *group memberships*, we could get the centers by computing the mean per group.



Slide credit: Kristen Grauman

K-means Clustering

- Initialization:
  - choose *k* cluster centers
- Repeat:
  - **assignment step:**
    - For every point find its closest center
  - **update step:**
    - Update every center as the mean of its points
- Until:
  - The maximum number of iterations is reached, or
  - No changes during the assignment step, or
  - The average distortion per point drops very little

[Lloyd, 1957]

K-means Clustering

technical note

- **Input:** N examples $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ $(\mathbf{x}_n \in \mathbb{R}^D)$; the number of partitions K
- **Initialize:** K cluster centers $\mu_1, \ldots, \mu_K$. Several initialization options:
  - Randomly initialized anywhere in $\mathbb{R}^D$
  - Choose any K examples as the cluster centers
- **Iterate:**
  - Assign each of example $\mathbf{x}_n$ to its closest cluster center

$$\mathcal{C}_k = \{n : \quad k = \arg\min_k ||\mathbf{x}_n - \mu_k||^2\}$$

  ($\mathcal{C}_k$ is the set of examples closest to μ_k)
  - Recompute the new cluster centers μ_k (mean/centroid of the set $\mathcal{C}_k$)

$$\mu_k = \frac{1}{|\mathcal{C}_k|} \sum_{n \in \mathcal{C}_k} \mathbf{x}_n$$

  - Repeat while not converged

slide credit: P. Rai

K-means Clustering

technical note

The K-means objective function

- Let $\mu_1, \ldots, \mu_K$ be the K cluster centroids (means)
- Let $r_{nk} \in \{0, 1\}$ be indicator denoting whether point $\mathbf{x}_n$ belongs to cluster k
- K-means objective minimizes the total distortion (sum of distances of points from their cluster centers)

$$J(\mu, r) = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\mathbf{x}_n - \mu_k||^2$$

- Note: Exact optimization of the K-means objective is NP-hard
- The K-means algorithm is a heuristic that converges to a local optimum [1]

[1] L. Bottou and Y. Bengio. Convergence properties of the kmeans algorithm. NIPS, 1995.

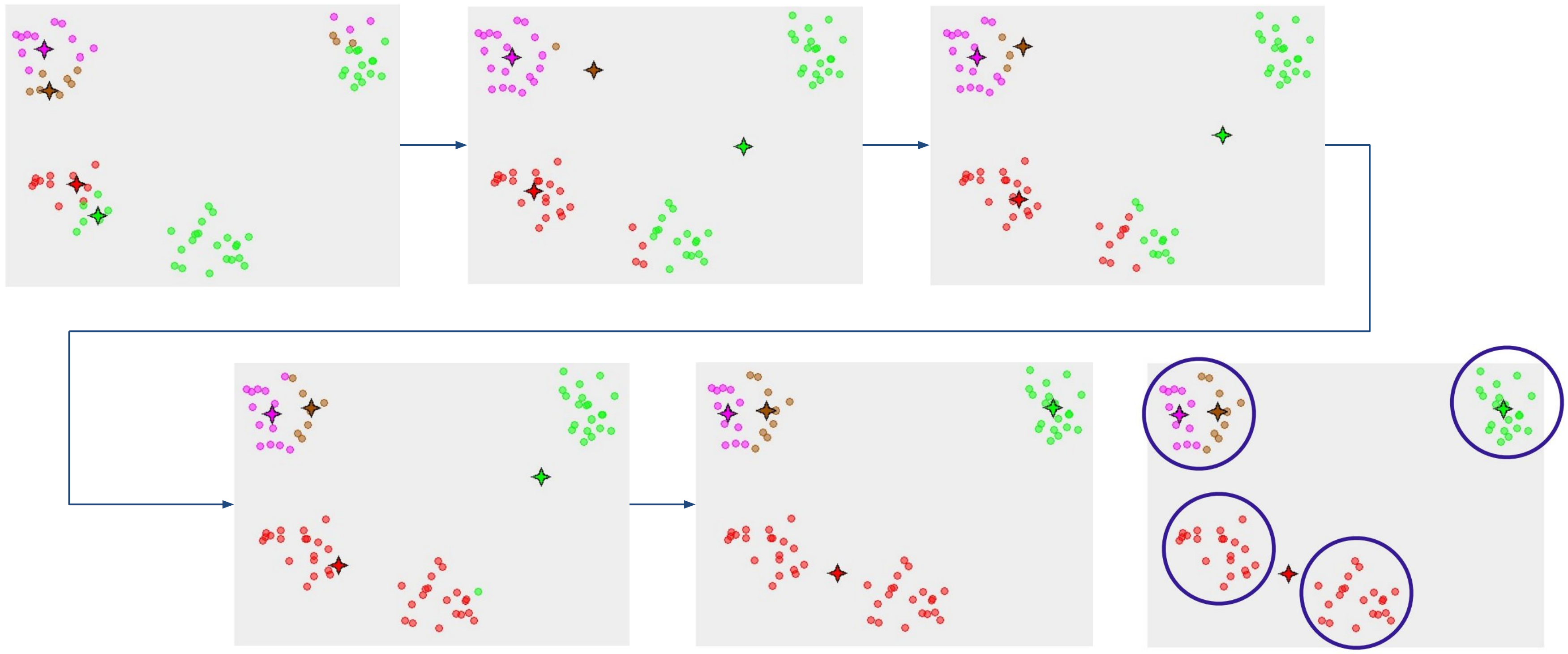
slide credit: P. Rai

K-means: Initialization

- k-means is *extremely sensitive* to initialization
- Bad initialization can lead to:
  - poor convergence speed
  - bad overall clustering
- How to initialize?
  - randomly from data
  - try to find K “spread-out” points (k-means++)
- Safeguarding measure:
  - try multiple initializations and choose the best

K-means: Initialization

- k-means is *extremely sensitive* to initialization
- Bad initialization can lead to:
  - poor convergence speed
  - bad overall clustering



K-means++

- Can we prevent arbitrarily bad local minima?

1. Randomly choose first center.
2. Pick new center with prob. proportional to $(x - c_i)^2$
   - (Contribution of *x* to total error)
3. Repeat until *K* centers.

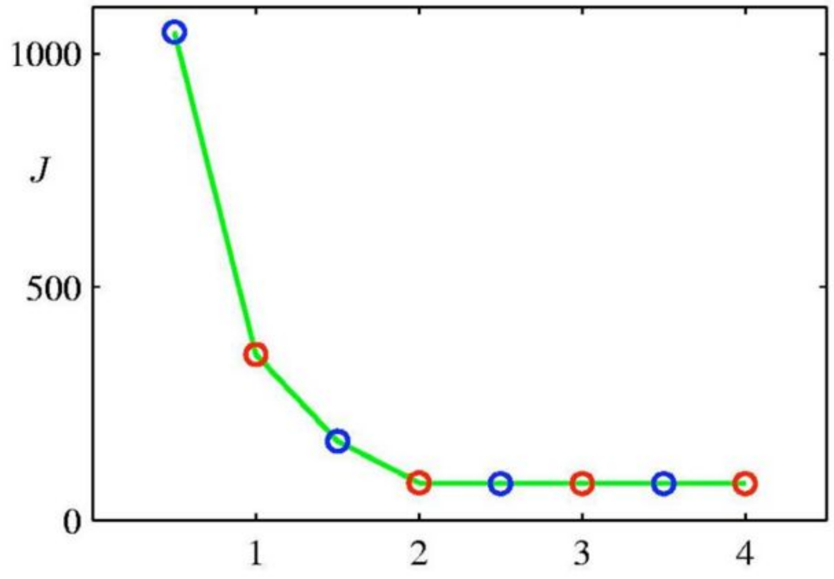
- Expected error O(logK) (optimal)

K-means++ animation

Arthur & Vassilvitskii 2007

K-means: choosing K

- One way to select K for the K-means algorithm is to try different values of K, plot the K-means objective versus K, and look at the "elbow-point" in the plot



- For the above plot, $K = 2$ is the elbow point

Picture courtesy: "Pattern Recognition and Machine Learning, Chris Bishop (2006)

slide credit: P. Rai

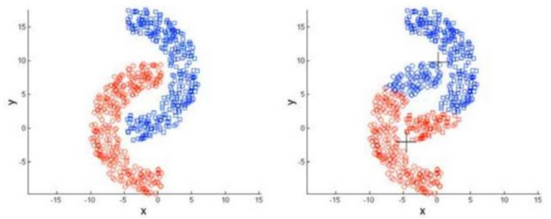
K-means: choosing K

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

Slide: Derek Hoiem

Distance Measure & Termination

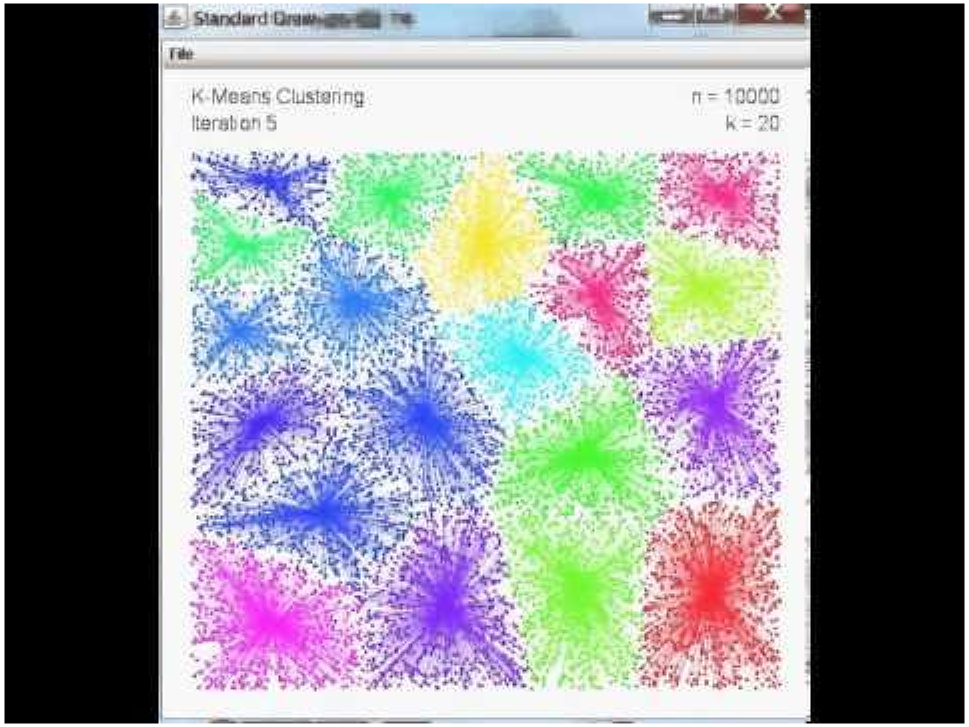
- Choice of “distance” measure:
    - Euclidean (most commonly used)
    - Cosine
    - non-linear! (Kernel k-means)



Picture courtesy: Christof Monz (Queen Mary, Univ. of London)

- Termination:
    - The maximum number of iterations is reached
    - No changes during the assignment step (convergence)
    - The average distortion per point drops very little

K-means: Example



K-Means Clustering Example

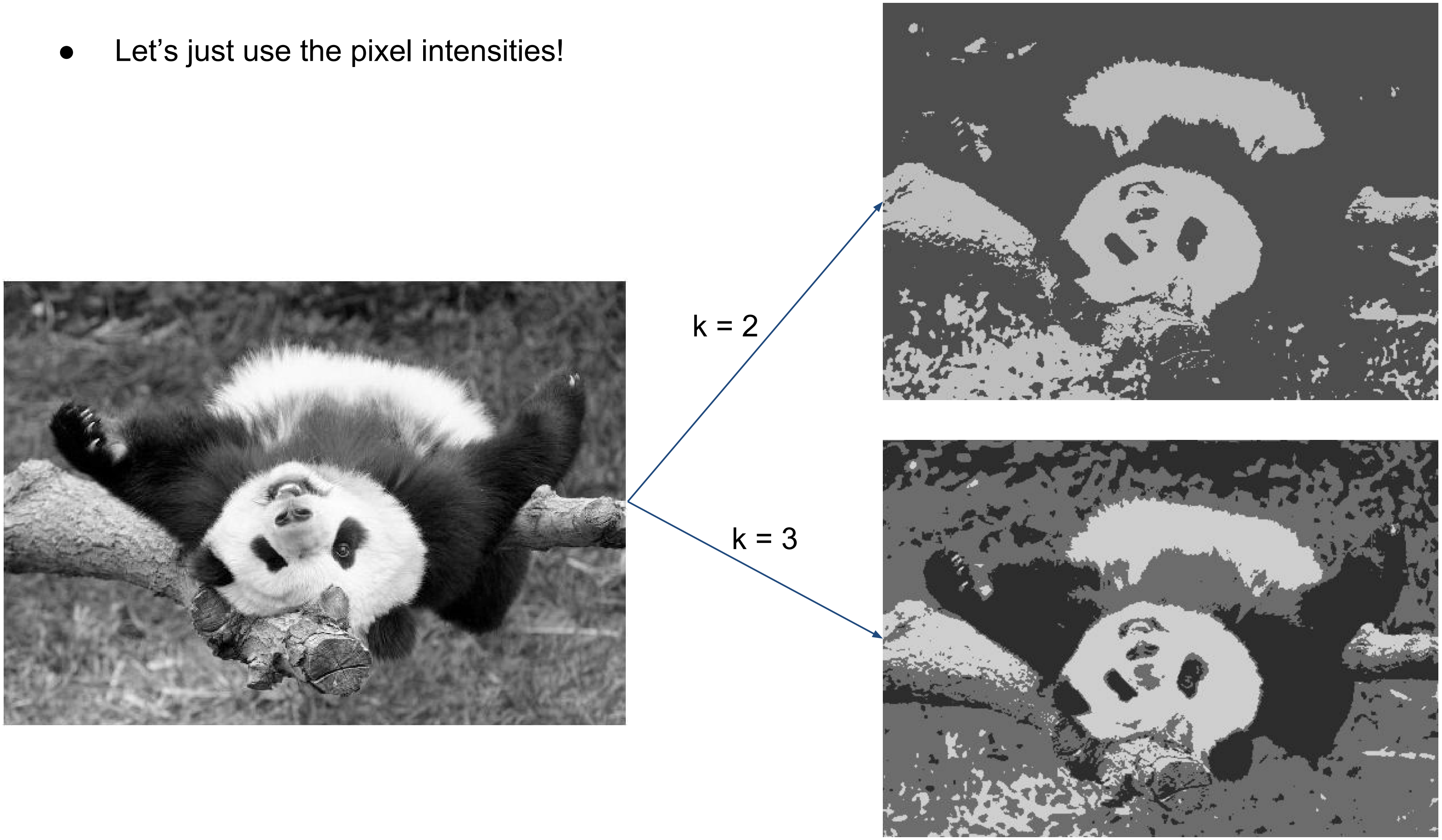
How to evaluate clusters?

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

Slide: Derek Hoiem

Segmentation as Clustering

- Let's just use the pixel intensities!

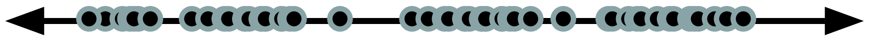


Slide credit: Kristen Grauman

Feature Space

- Depending on what we choose as the *feature space*, we can group pixels in different ways.

- Grouping pixels based on intensity similarity

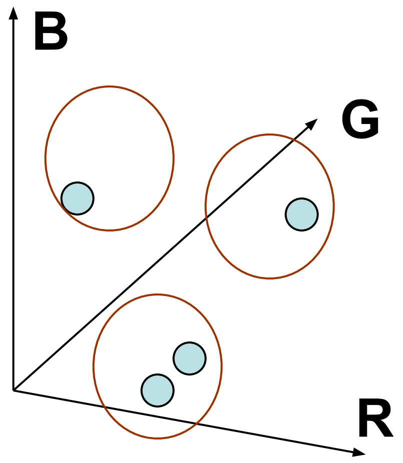


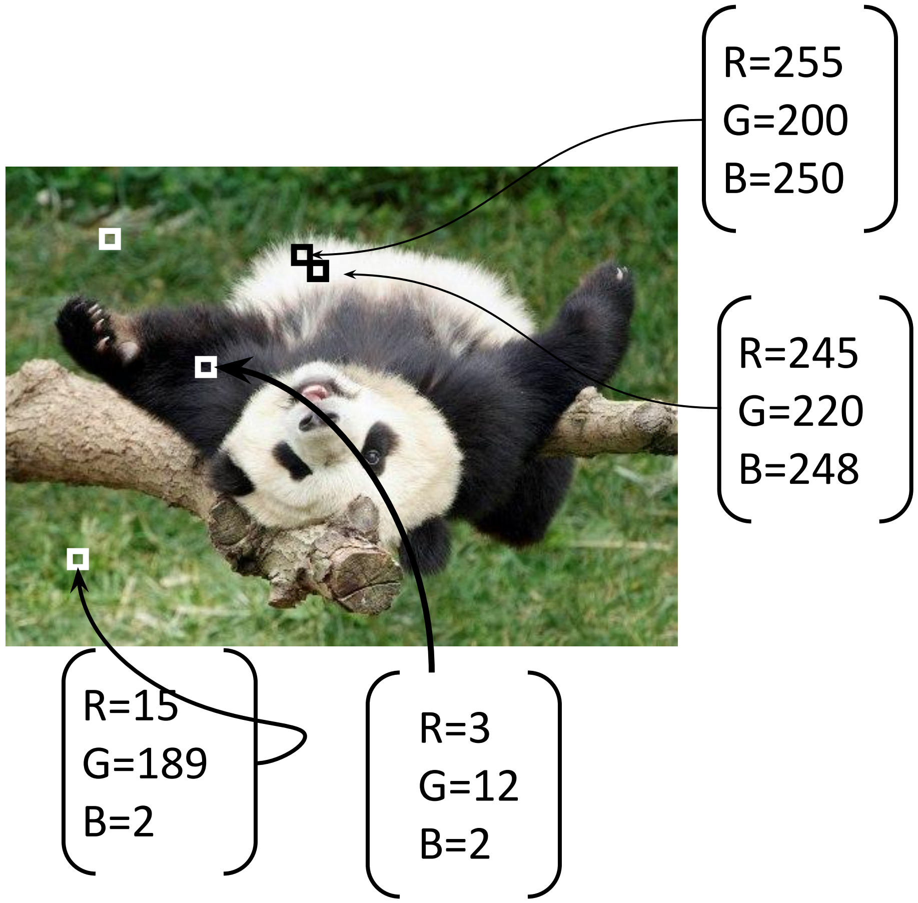
- Feature space: intensity value (1D)

Slide credit: Kristen Grauman

Feature Space

- Depending on what we choose as the *feature space*, we can group pixels in different ways.
- Grouping pixels based on color similarity





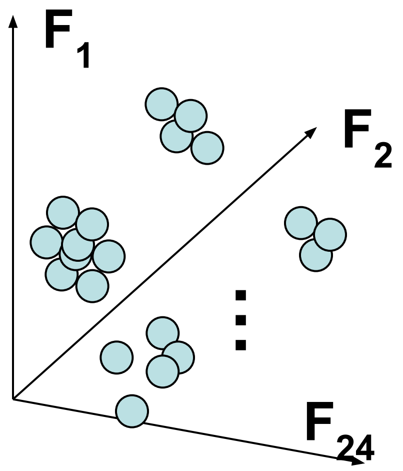
- Feature space: color value (3D)

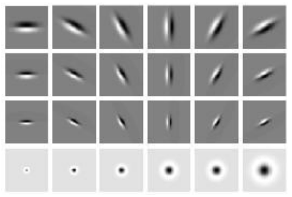
Slide credit: Kristen Grauman

Feature Space

- Depending on what we choose as the *feature space*, we can group pixels in different ways.

- Grouping pixels based on texture similarity





Filter bank of 24 filters

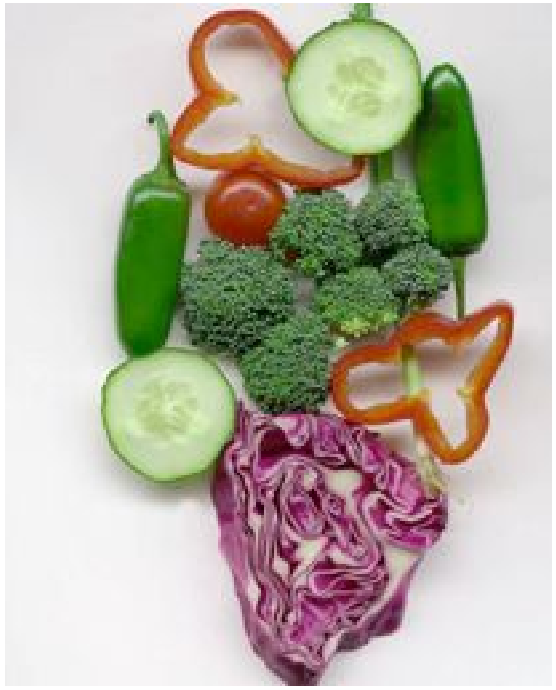
- Feature space: filter bank responses (e.g., 24D)

Slide credit: Kristen Grauman

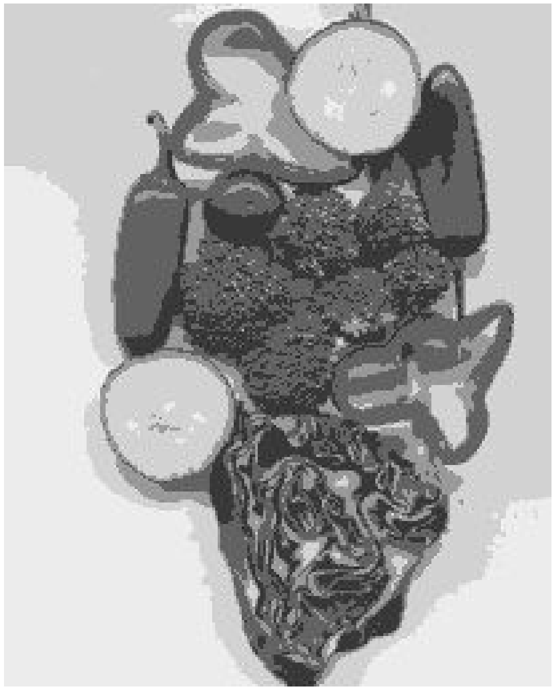
K-Means Clustering Results

- K-means 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

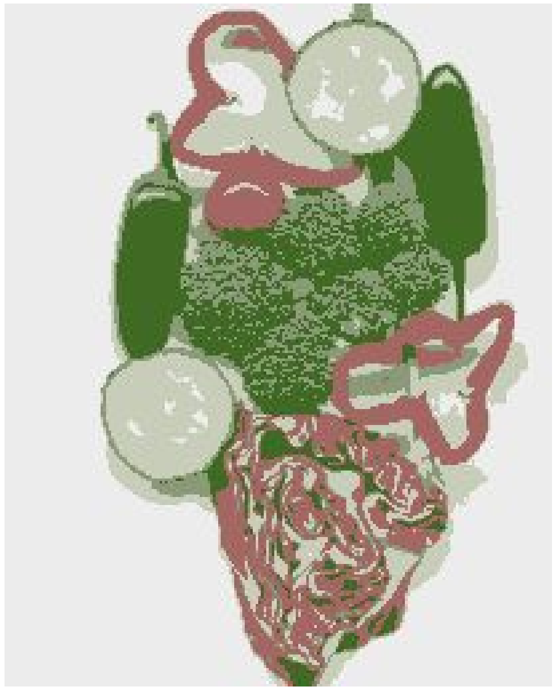
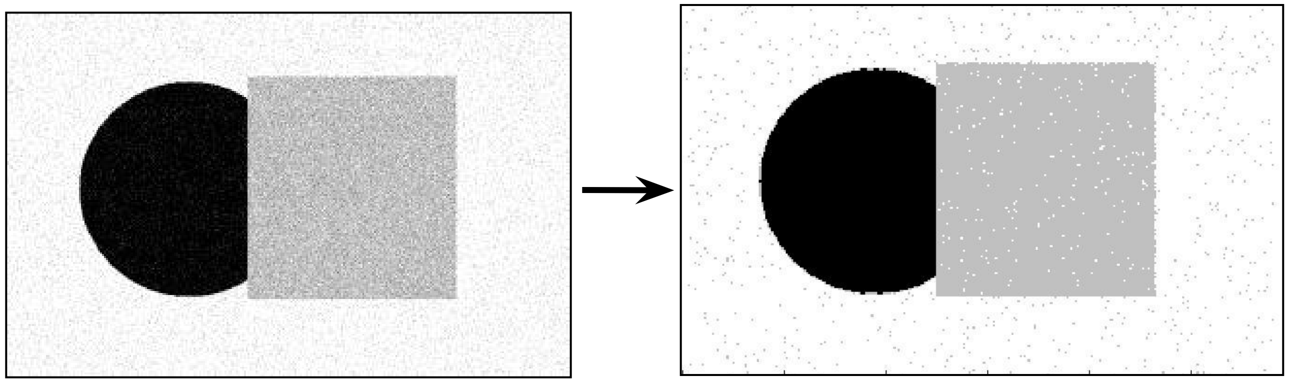


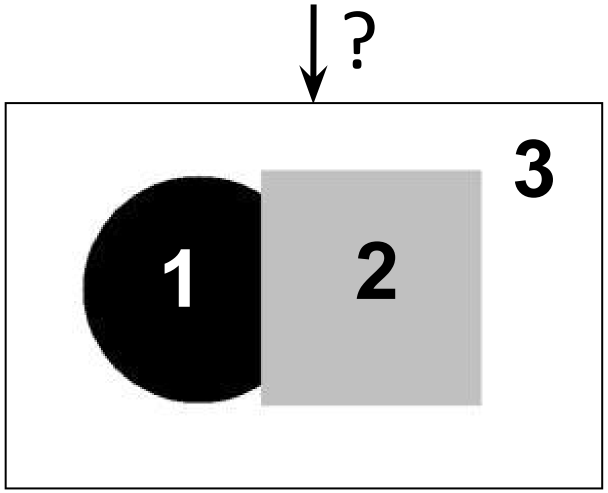
Image source: Forsyth & Ponce

Smoothing Out Cluster Assignments

- Assigning a cluster label per pixel may yield outliers:


Original

Labeled by cluster center's intensity

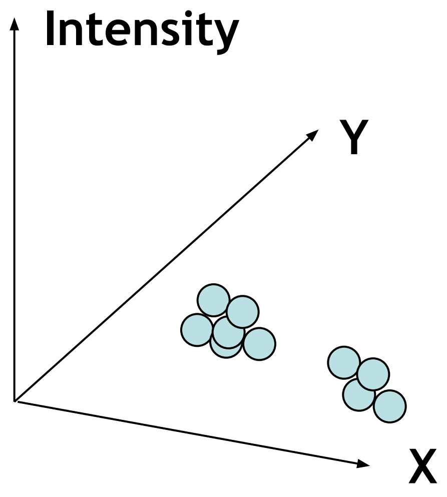


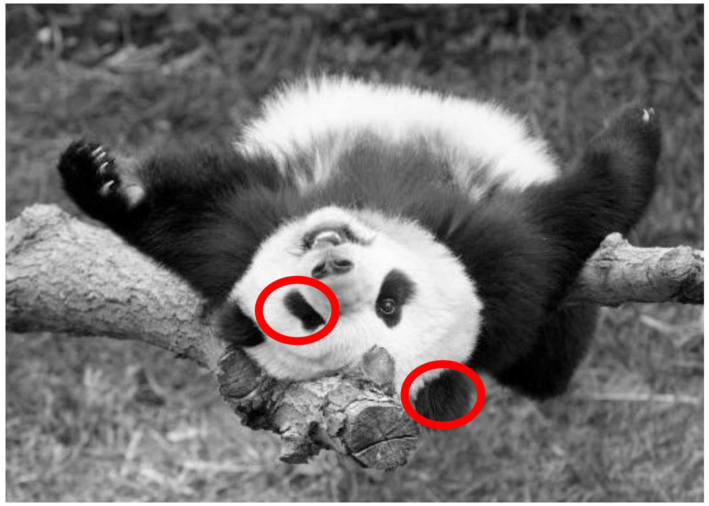
- How can we ensure they are spatially smooth?

Slide credit: Kristen Grauman

Segmentation as Clustering

- Depending on what we choose as the *feature space*, we can group pixels in different ways.
- Grouping pixels based on *intensity+position* similarity





⇒ Way to encode both *similarity* and *proximity.*

Slide credit: Kristen Grauman

K-means clustering for superpixels

SLIC Superpixels:

- Feature space → intensity + position
  - L.a.b. color space
  - *limited region* (window 2*S)

$$[l_k, a_k, b_k, x_k, y_k]$$

- Distance metric:

$$d_c = \sqrt{(l_j - l_i)^2 + (a_j - a_i)^2 + (b_j - b_i)^2}$$
$$d_s = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$
$$D' = \sqrt{\left(\frac{d_c}{N_c}\right)^2 + \left(\frac{d_s}{N_s}\right)^2}.$$

- Initialization:
  - Spatial grid (grid step = S)
- Iterate over centers and not points

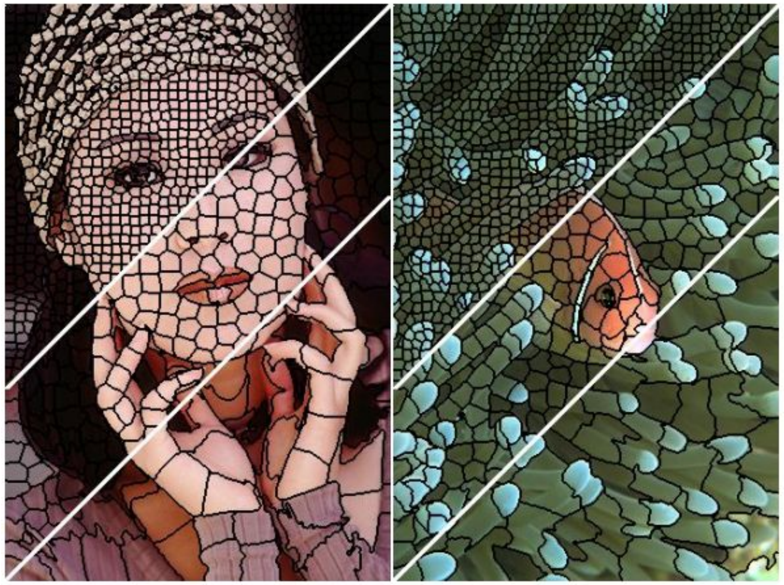
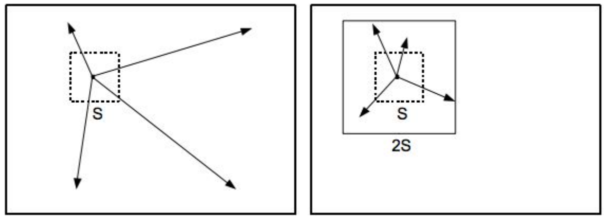


Fig. 1: Images segmented using SLIC into superpixels of size 64, 256, and 1024 pixels (approximately).

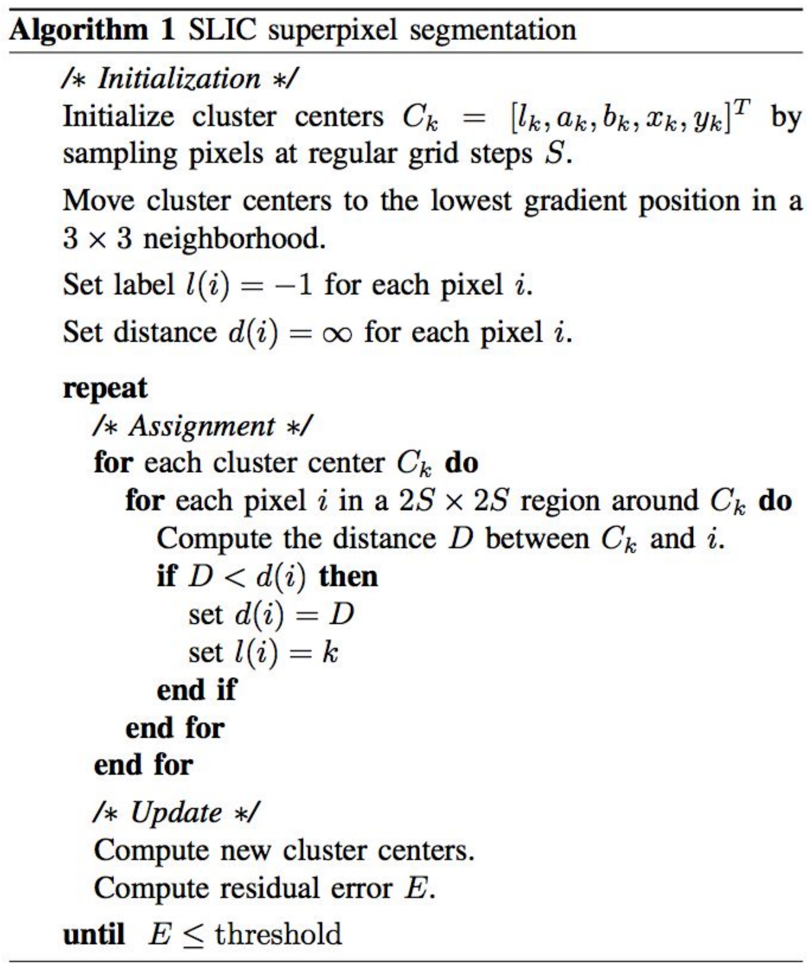


(a) standard k-means searches the entire image

(b) SLIC searches a limited region

Achanta *et al.*, SLIC Superpixels Compared to State-of-the-art Superpixel Methods, PAMI 2012.

K-means clustering for superpixels



Algorithm 1 SLIC superpixel segmentation

```
/* Initialization */
Initialize cluster centers $C_k = [l_k, a_k, b_k, x_k, y_k]^T$ by sampling pixels at regular grid steps $S$.
Move cluster centers to the lowest gradient position in a $3 \times 3$ neighborhood.
Set label $l(i) = -1$ for each pixel $i$.
Set distance $d(i) = \infty$ for each pixel $i$.
repeat
  /* Assignment */
  for each cluster center $C_k$ do
    for each pixel $i$ in a $2S \times 2S$ region around $C_k$ do
      Compute the distance $D$ between $C_k$ and $i$.
      if $D < d(i)$ then
        set $d(i) = D$
        set $l(i) = k$
      end if
    end for
  end for
  /* Update */
  Compute new cluster centers.
  Compute residual error $E$.
until $E \leq$ threshold
```

Achanta *et al.*, SLIC Superpixels Compared to State-of-the-art Superpixel Methods, PAMI 2012.

K-means clustering for superpixels

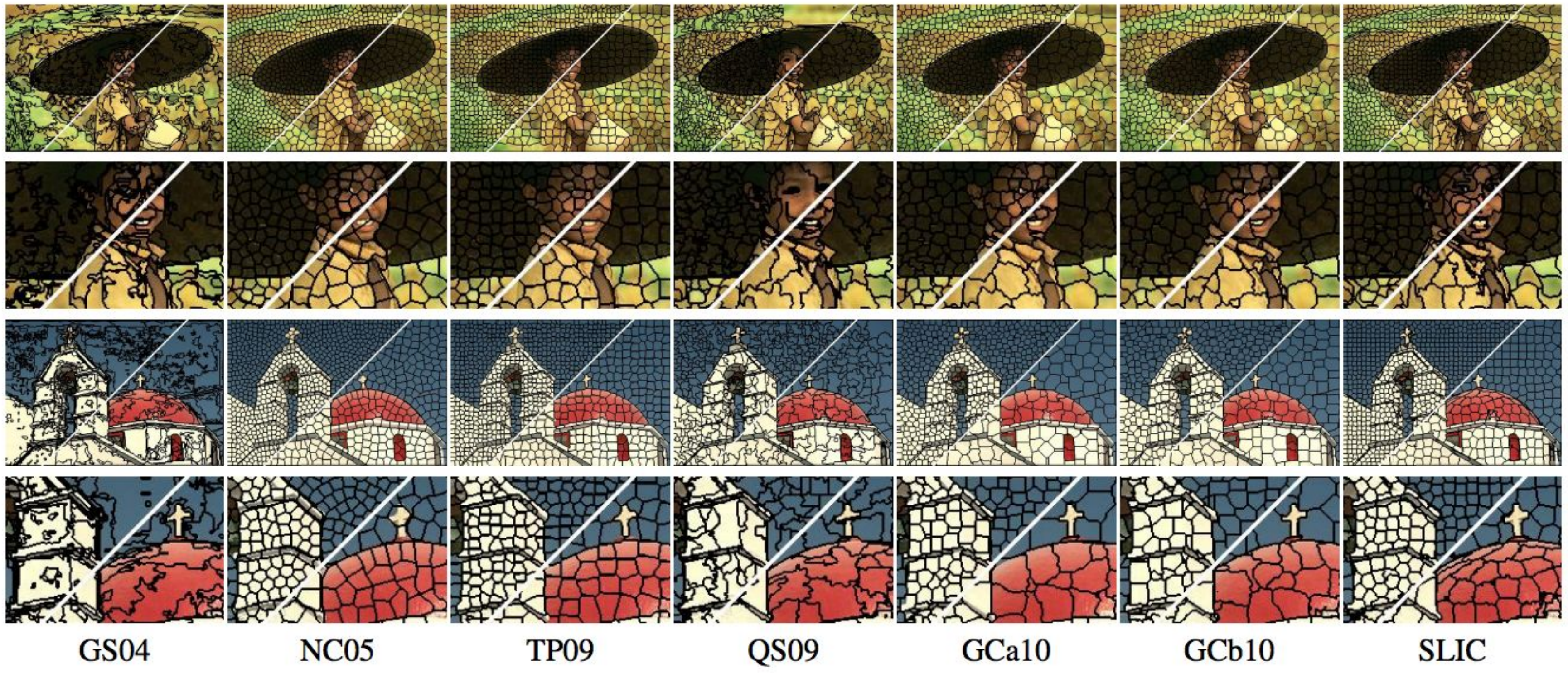
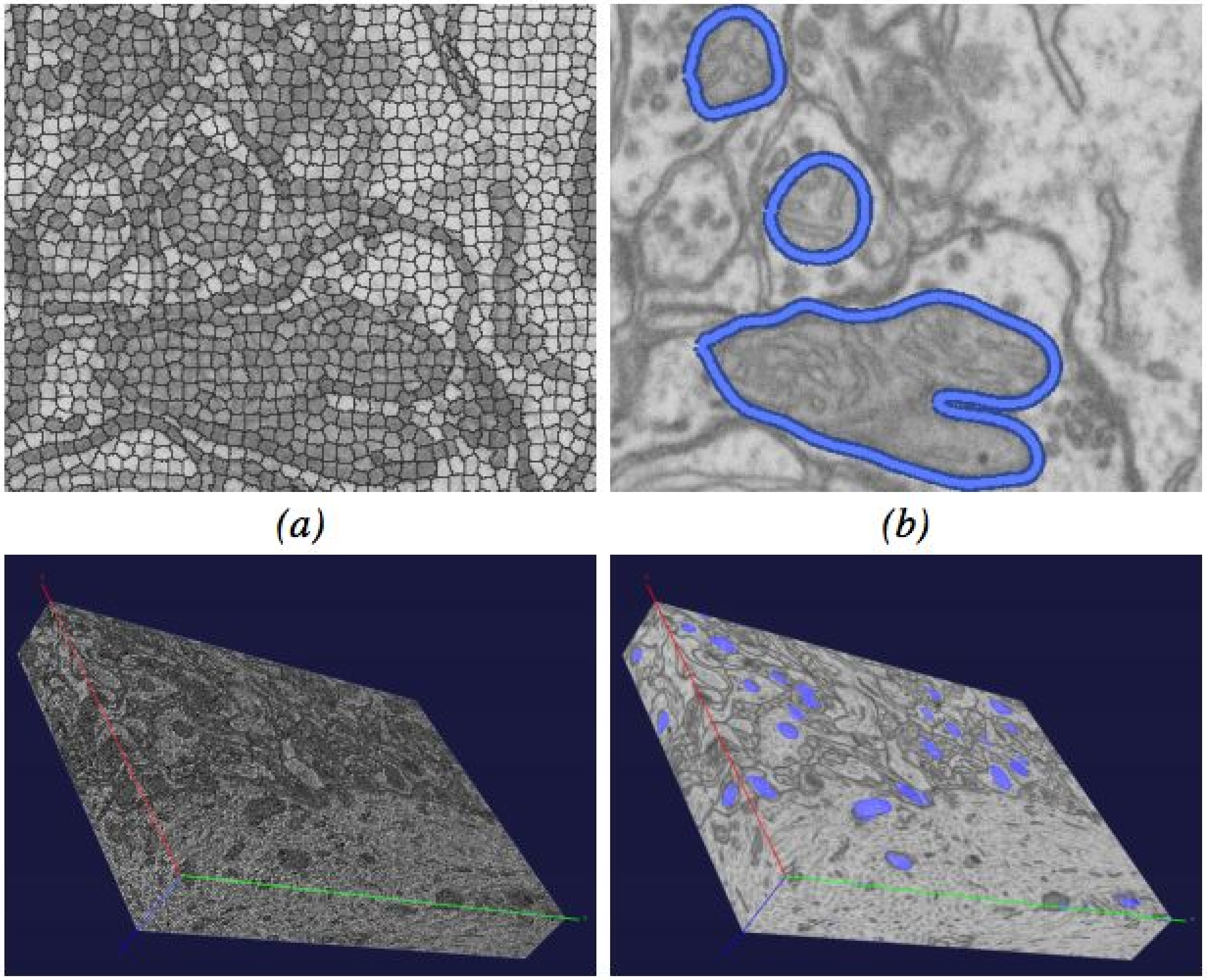


Fig. 7: Visual comparison of superpixels produced by various methods. The average superpixel size in the upper left of each image is 100 pixels, and 300 in the lower right. Alternating rows show each segmented image followed by a detail of the center of each image.

Achanta *et al.*, SLIC Superpixels Compared to State-of-the-art Superpixel Methods, PAMI 2012.

K-means clustering for superpixels



Achanta *et al.*, SLIC Superpixels Compared to State-of-the-art Superpixel Methods, PAMI 2012.

K-means Clustering: Limitations

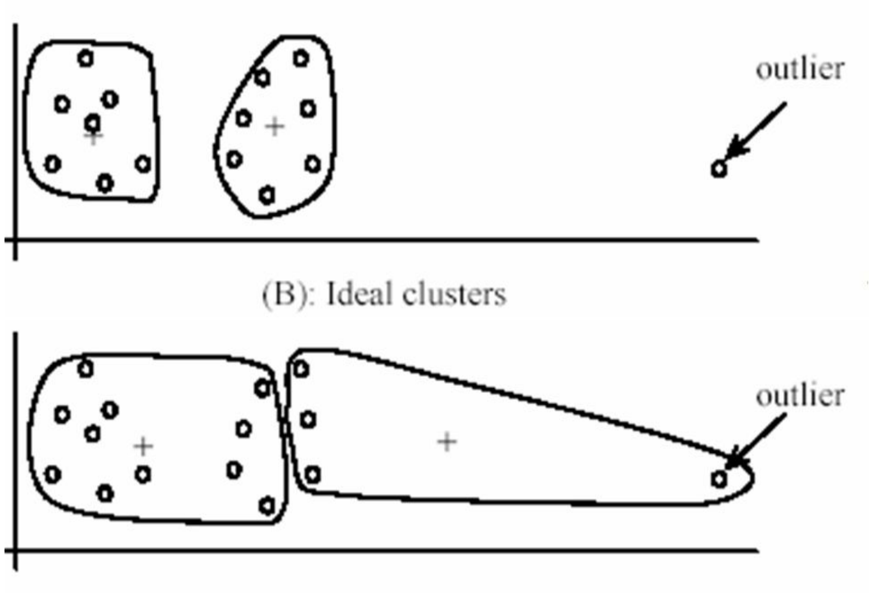
- Makes hard assignments of points to clusters
  - A point either completely belongs to a cluster or not belongs at all
  - No notion of a soft assignment (i.e., probability of being assigned to each cluster: say $K = 3$ and for some point $\mathbf{x}_n$, $p_1 = 0.7, p_2 = 0.2, p_3 = 0.1$)
  - Gaussian mixture models and Fuzzy K-means allow soft assignments
- Sensitive to outlier examples (such examples can affect the mean by a lot)
  - K-medians algorithm is a more robust alternative for data with outliers
  - Reason: Median is more robust than mean in presence of outliers
- Works well only for round shaped, and of roughly equal sizes/density clusters
- Does badly if the clusters have non-convex shapes
  - Spectral clustering or kernelized K-means can be an alternative [1]

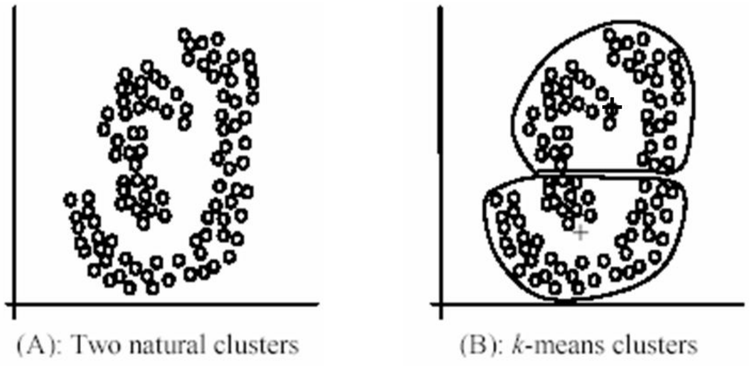
[1] Dhillon et al. Kernel k-means, Spectral Clustering and Normalized Cuts. KDD, 2004.

slide credit: P. Rai

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
  - Unsupervised clustering
  - Rarely used for pixel segmentation

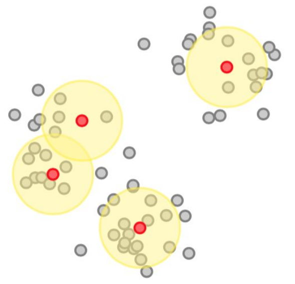


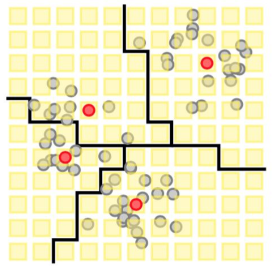


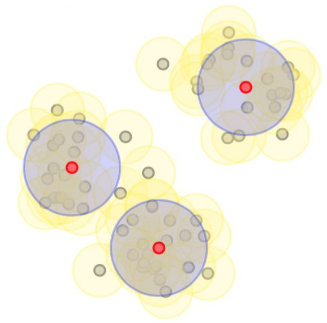
Scaling-up K-means clustering

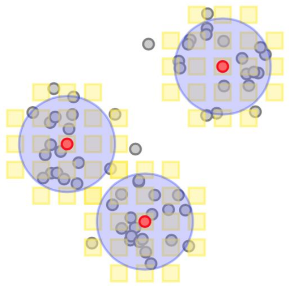
- Assignment step is the bottleneck

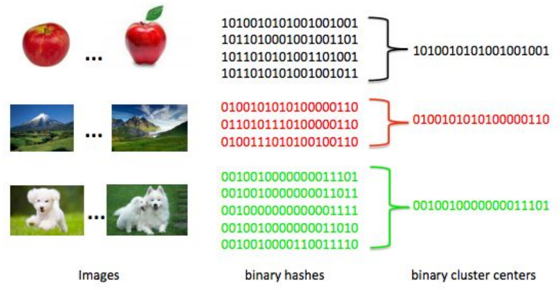
- Approximate assignments
  - [AK-means, CVPR 2007], [AGM, ECCV 2012]
- Mini-batch version
  - [mbK-means, WWW 2010]
- Search from every center
  - [Ranked retrieval, WSDM 2014]
- Binarize data and centroids
  - [BK-means, CVPR 2015]
- Quantize data
  - [DRVQ, ICCV 2013], [IQ-means, ICCV 2015]


Ranked retrieval


DRVQ


EGM


IQ-means



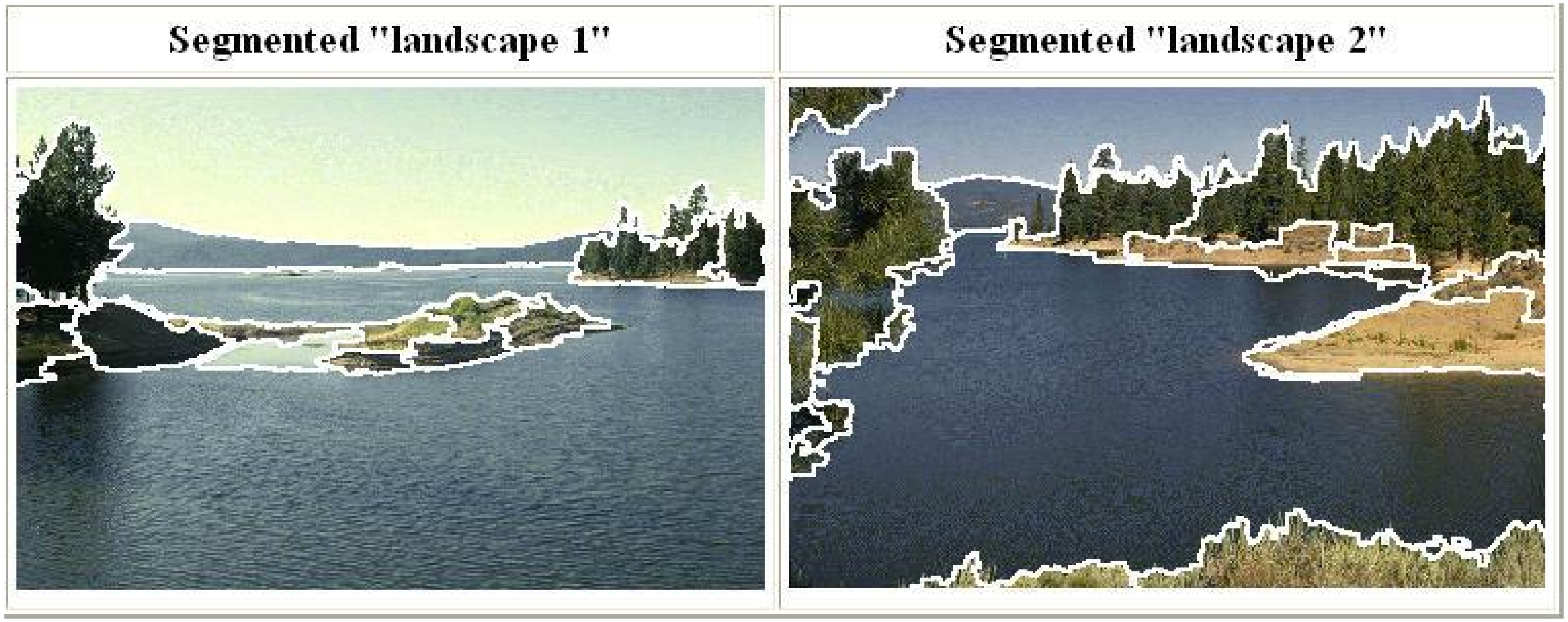
BK-means

What will we learn today?

- K-means clustering
- Mean-shift clustering

Mean-Shift Segmentation

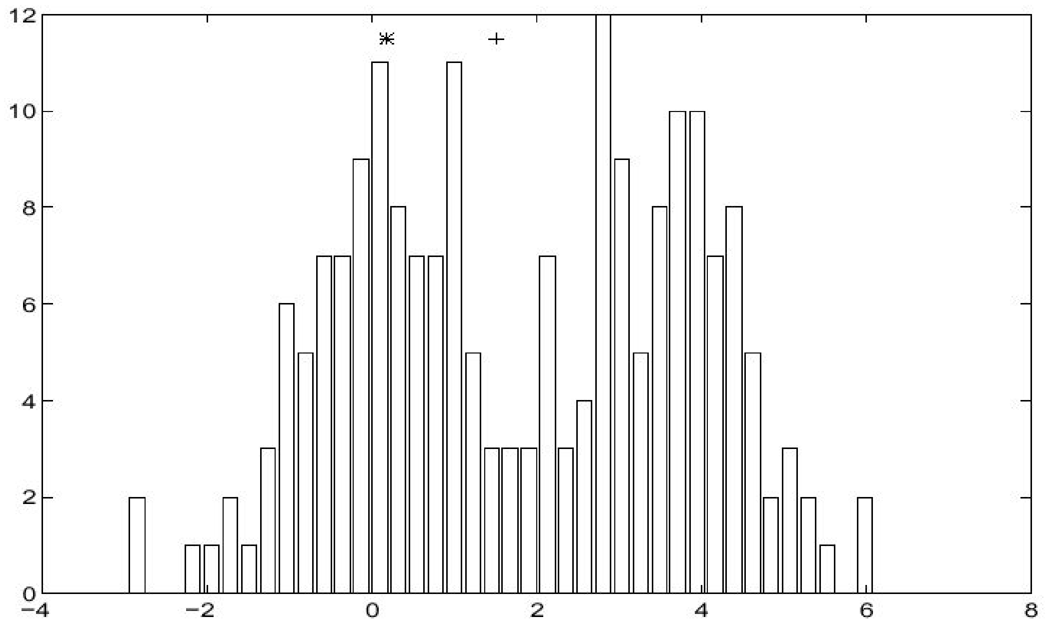
- An advanced and versatile technique for clustering-based segmentation



Slide credit: Svetlana Lazebnik

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

Mean-Shift Algorithm

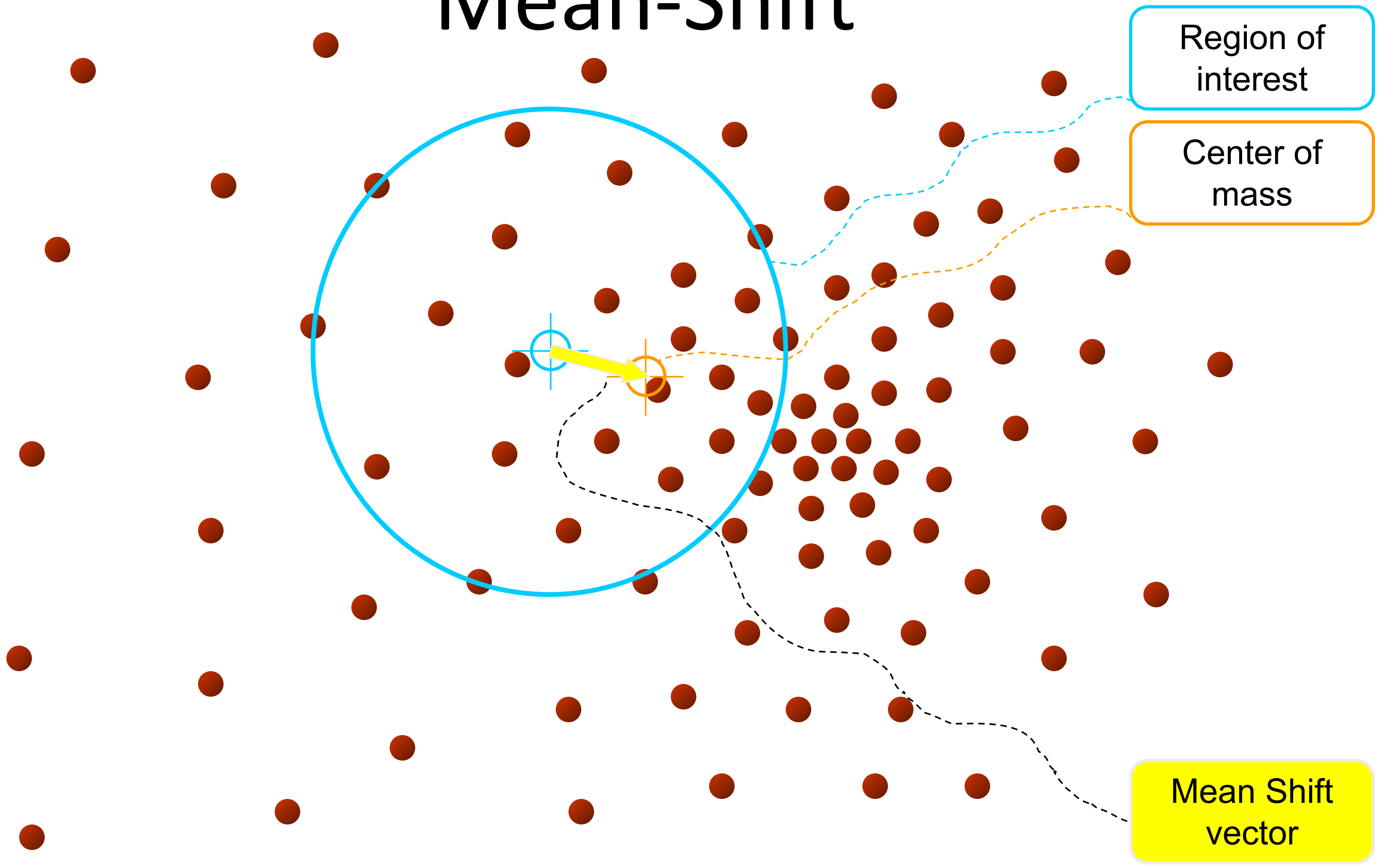


- Iterative Mode Search
  1. Initialize random seed, and window W
  2. Calculate center of gravity (the "mean") of W = $\sum_{x \in W} xH(x)$
  3. Shift the search window to the mean
  4. Repeat Step 2 until convergence

[Fukunaga & Hostetler, 1975]

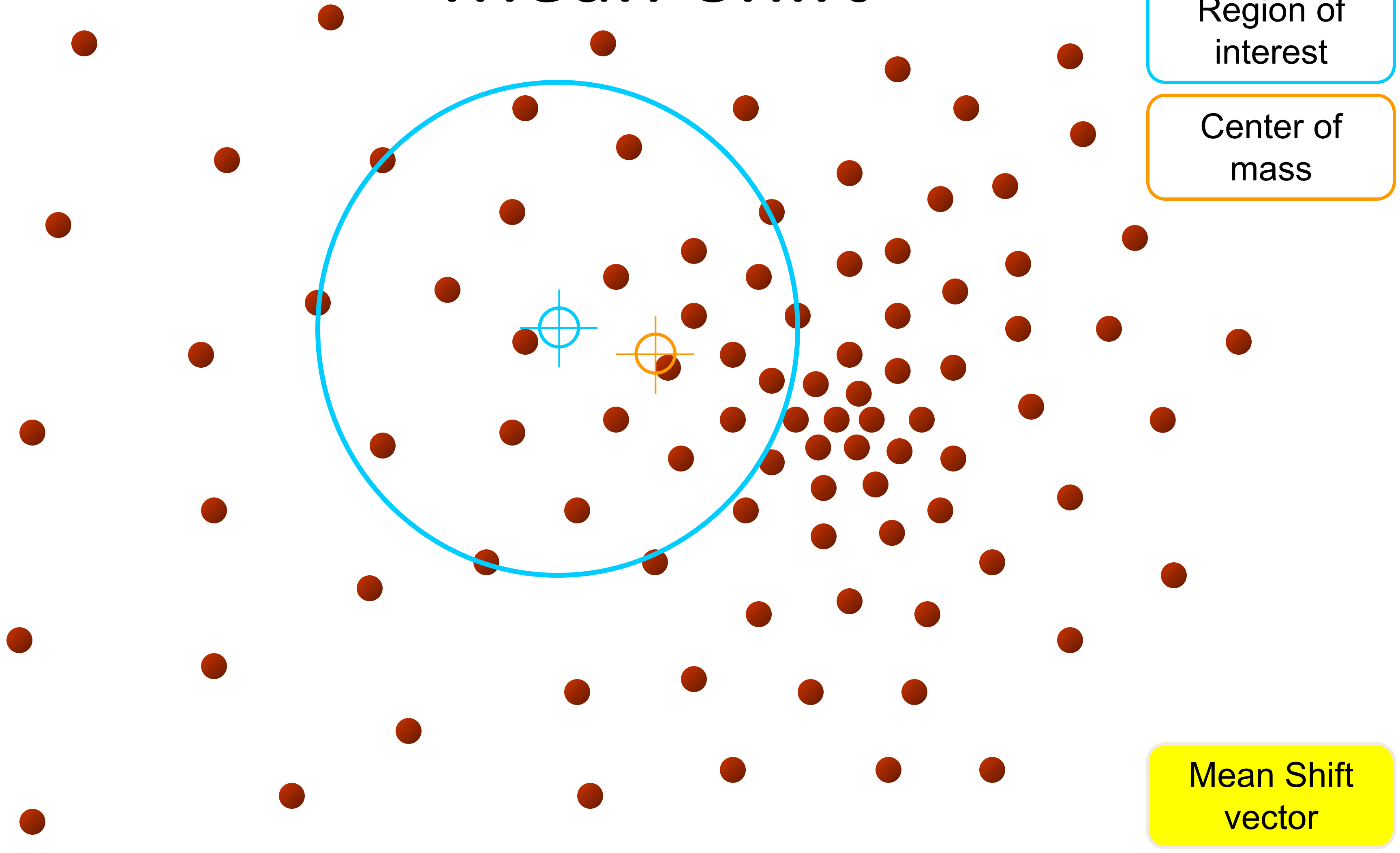
Slide credit: Steve Seitz

Mean-Shift



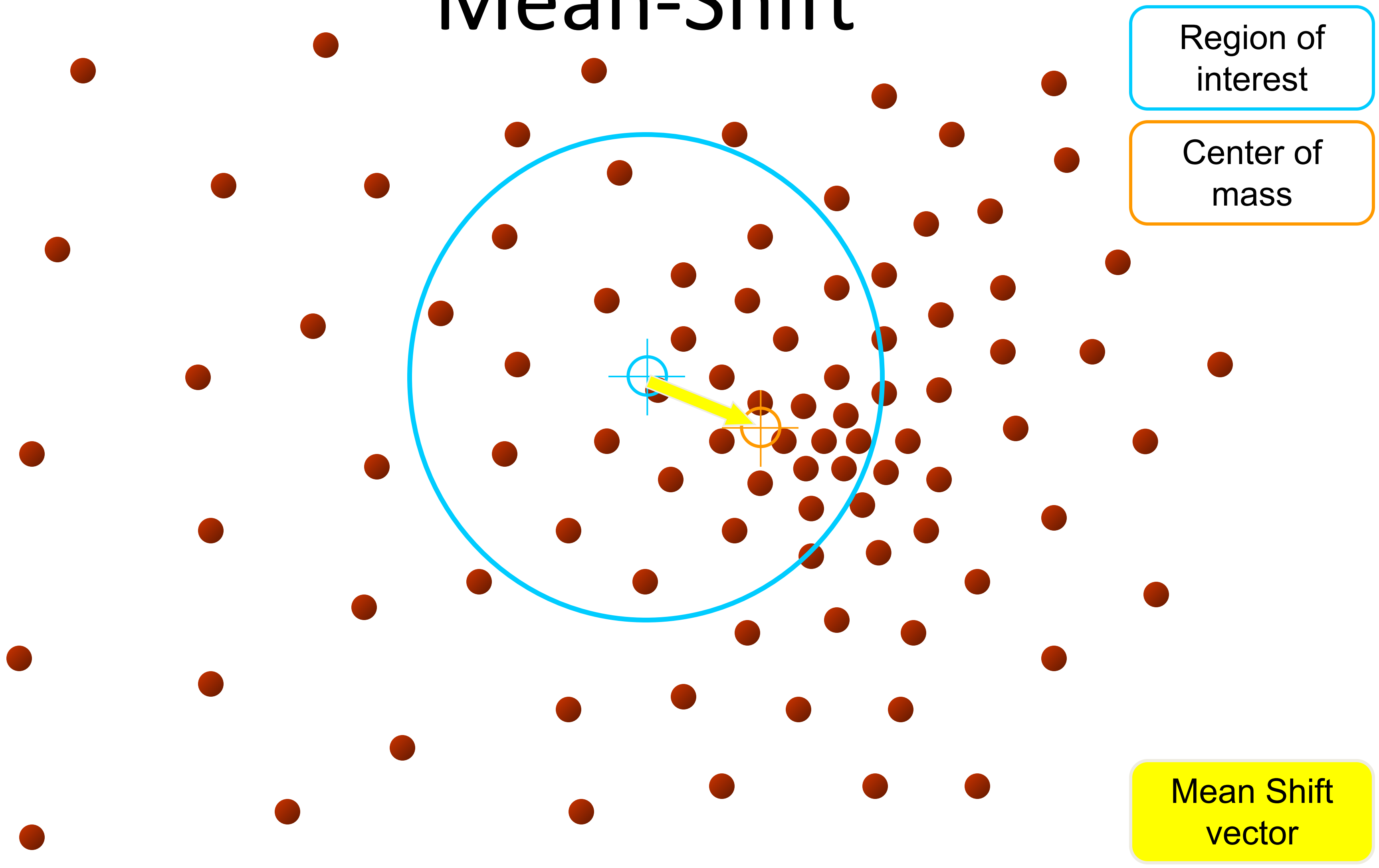
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



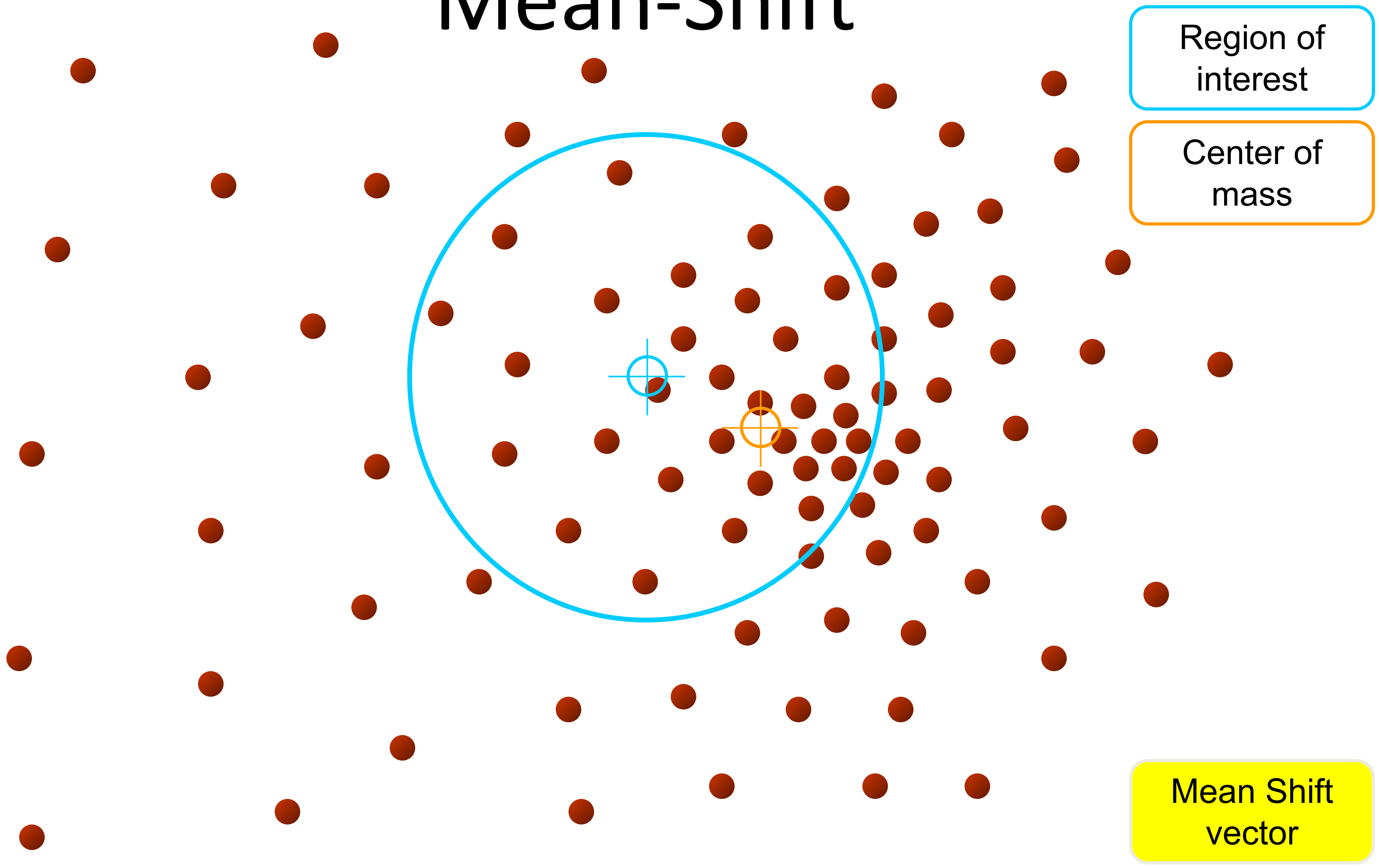
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



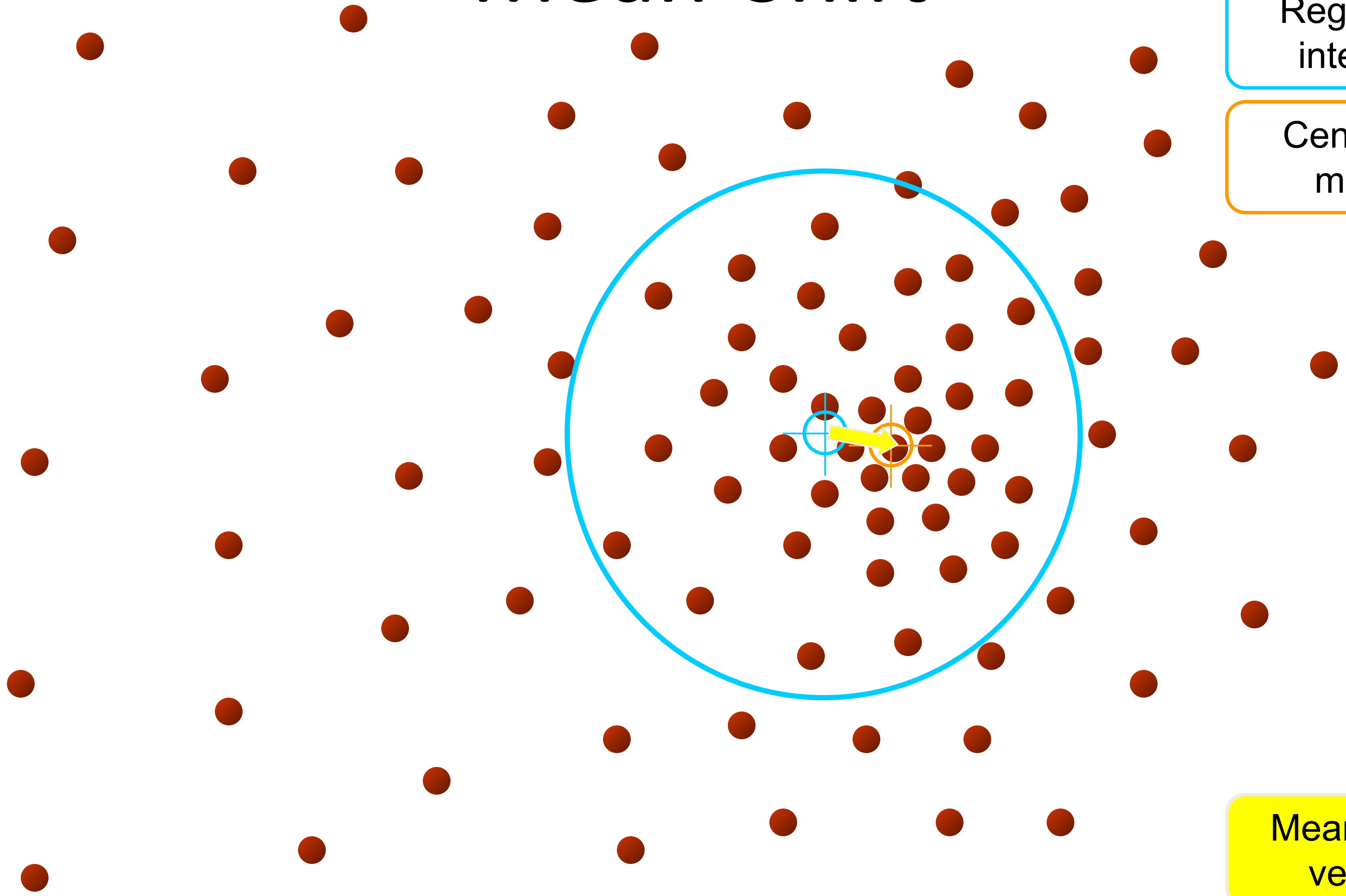
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



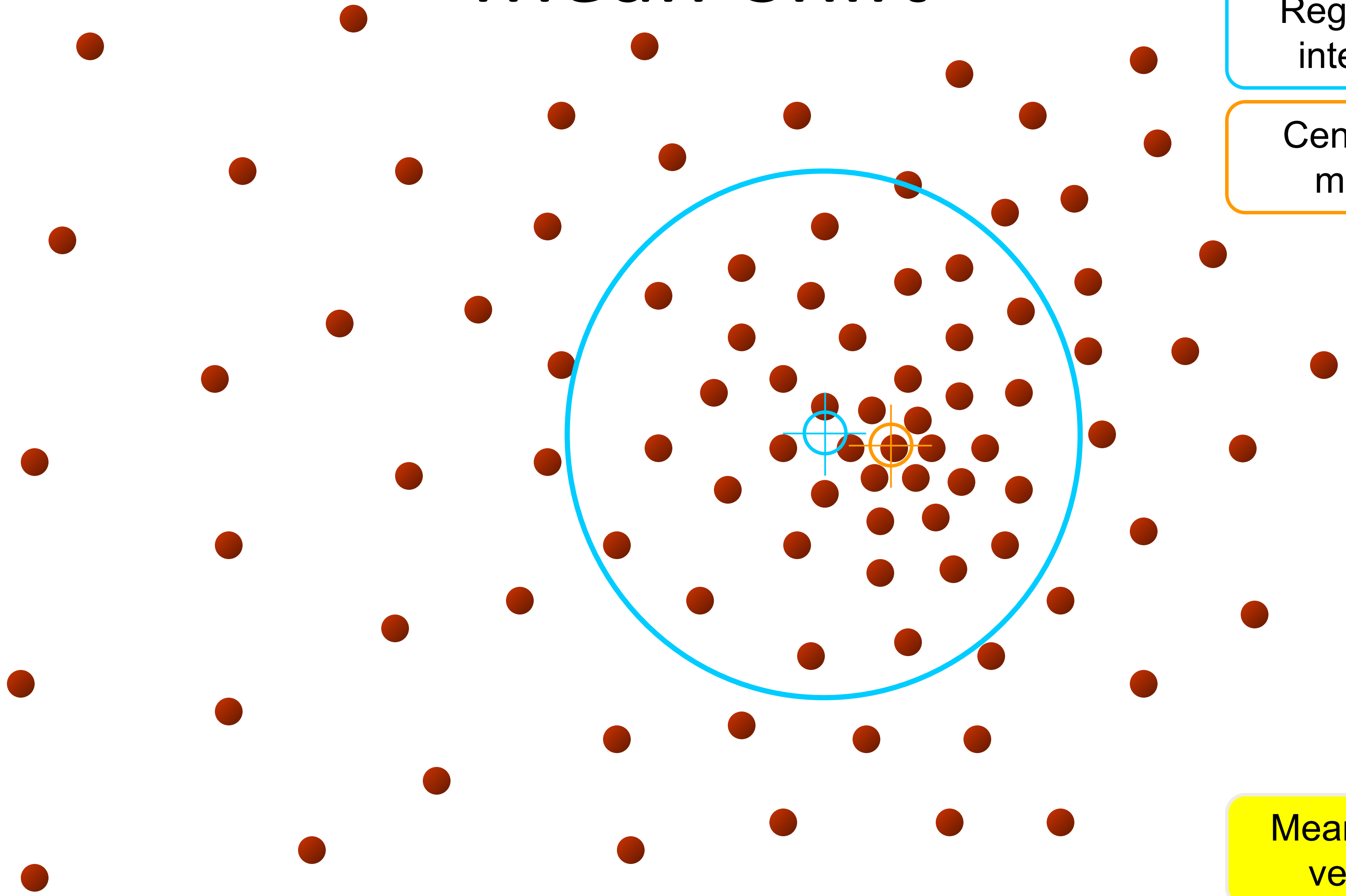
Region of interest

Center of mass

Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



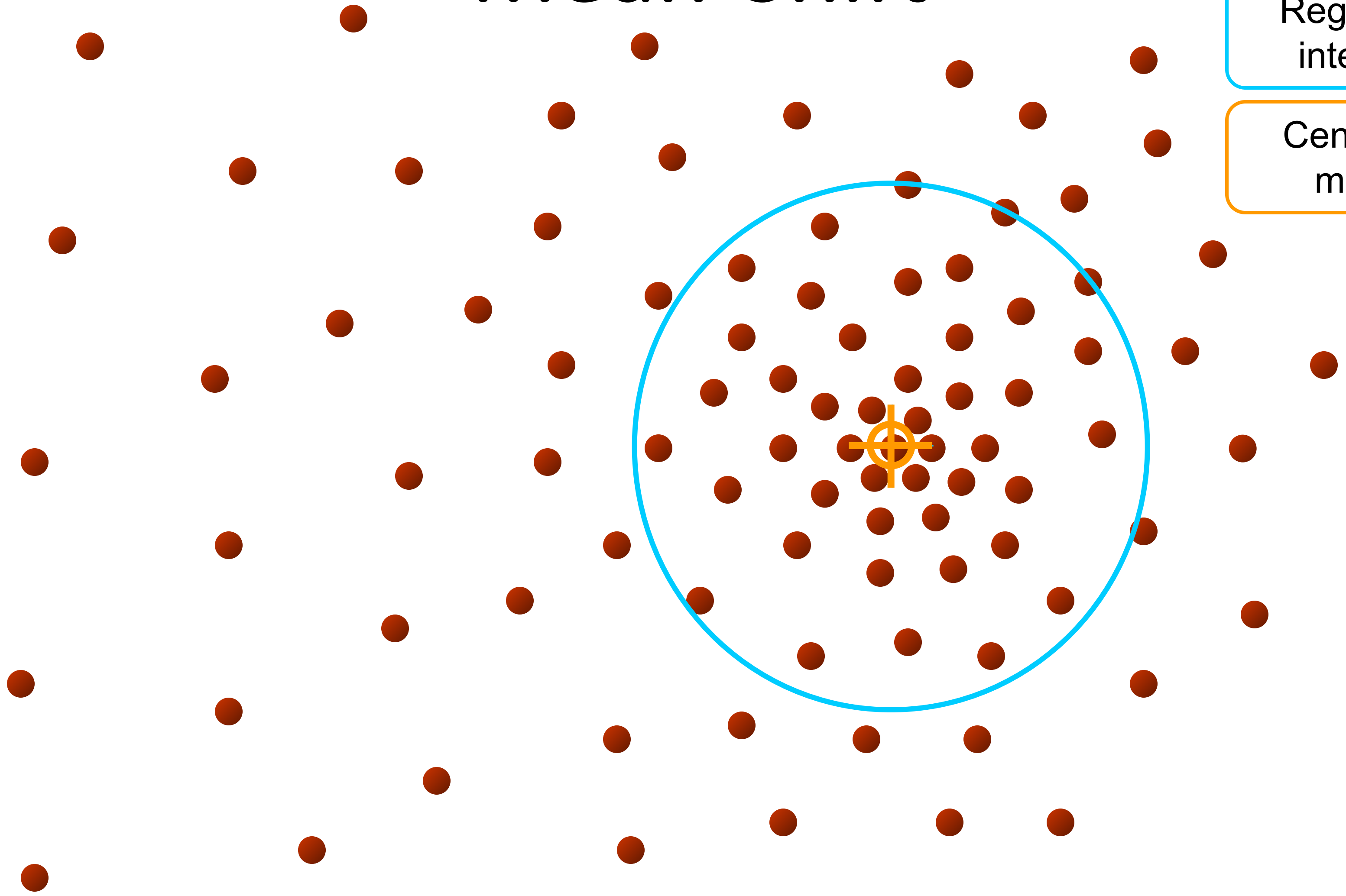
Region of interest

Center of mass

Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel

Mean-Shift

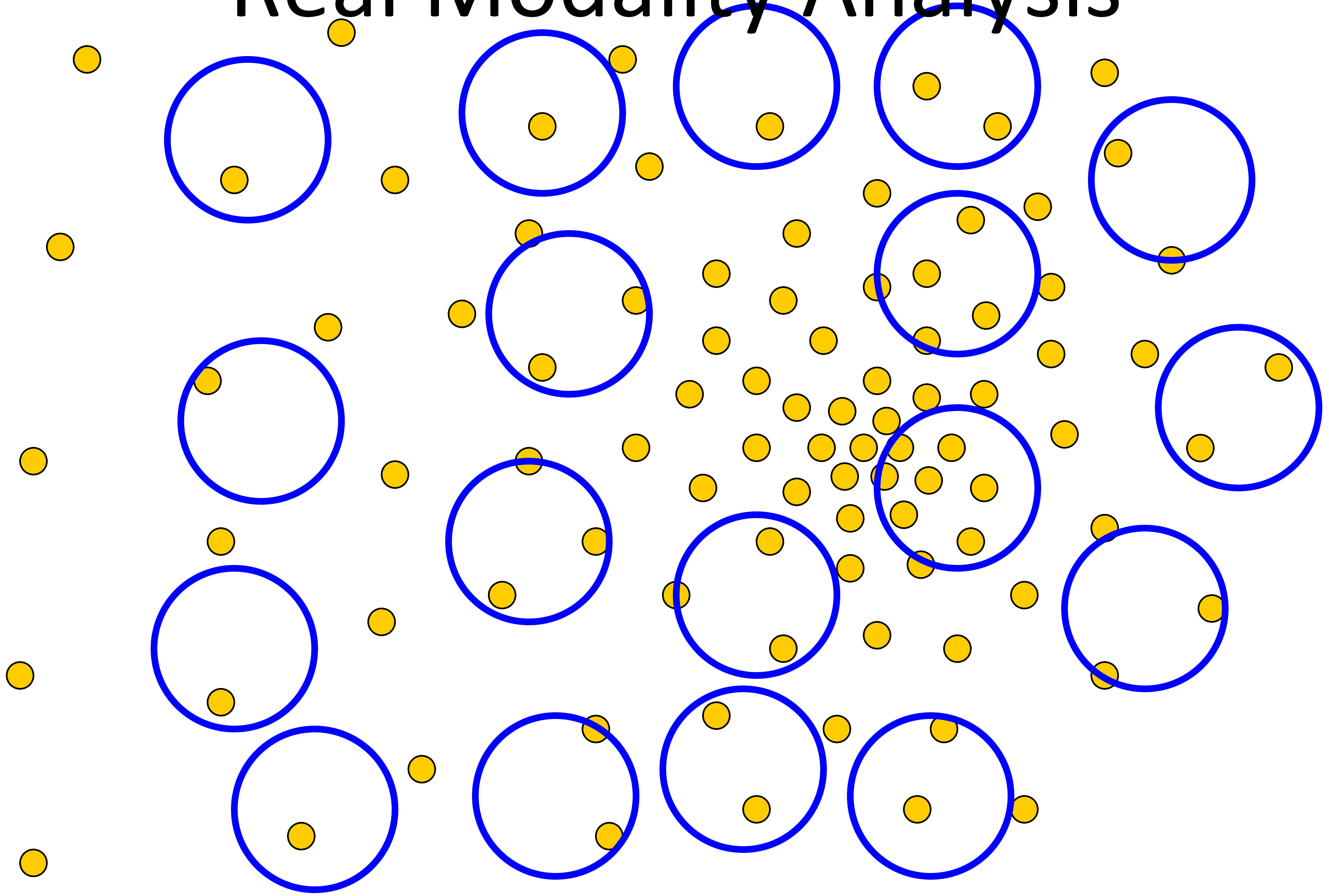


Region of interest

Center of mass

Slide by Y. Ukrainitz & B. Sarel

Real Modality Analysis

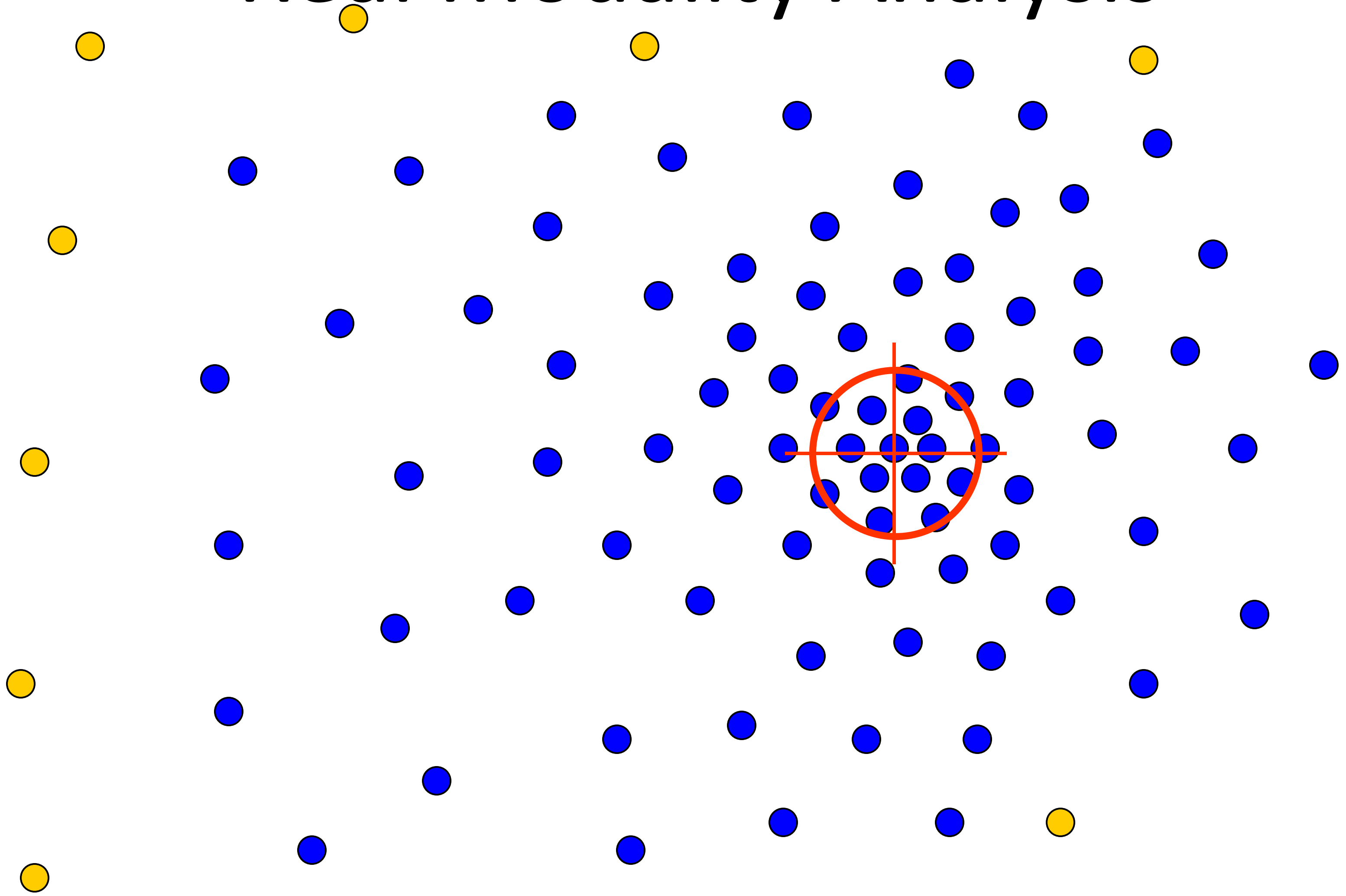


Tessellate the space with windows

Run the procedure in parallel

Slide by Y. Ukrainitz & B. Sarel

Real Modality Analysis

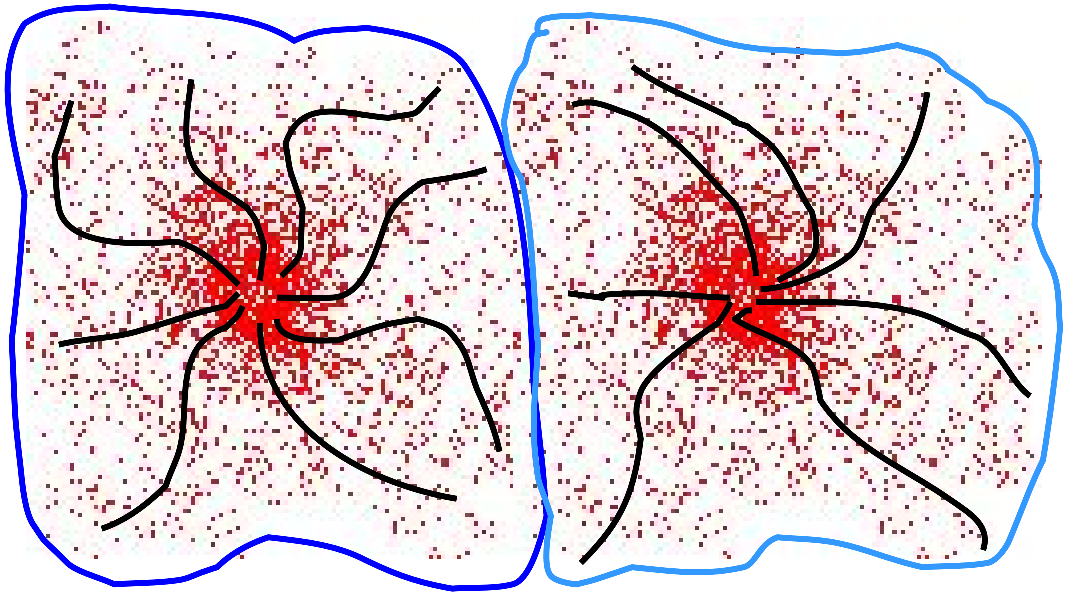


The blue data points were traversed by the windows towards the mode.

Slide by Y. Ukrainitz & B. Sarel

Mean-Shift Clustering

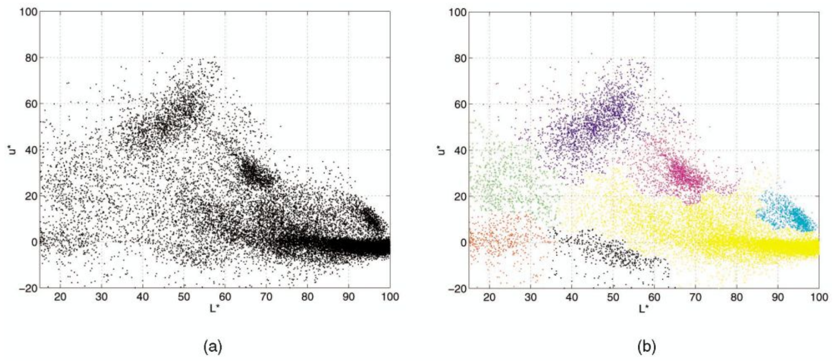
- Cluster: all data points in the attraction basin of a mode
- Attraction basin: the region for which all trajectories lead to the same mode

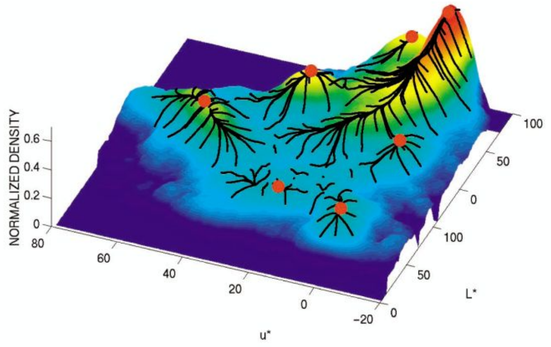


Slide by Y. Ukrainitz & B. Sarel

Mean-Shift Clustering/Segmentation

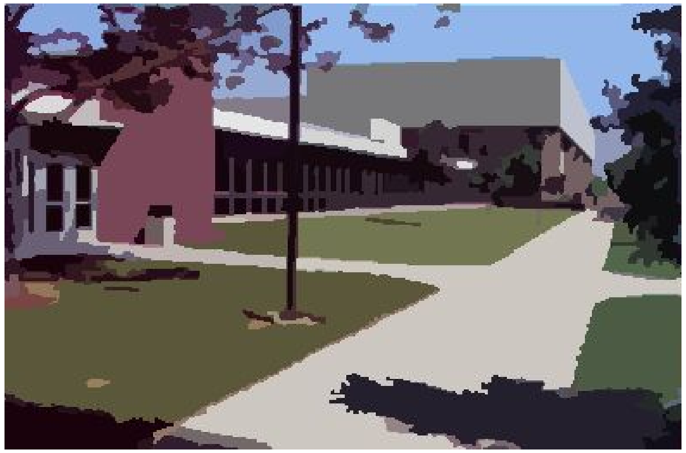
- Find features (color, gradients, texture, etc)
- Initialize windows at individual pixel locations
- Perform mean shift for each window until convergence
- Merge windows that end up near the same "peak" or mode

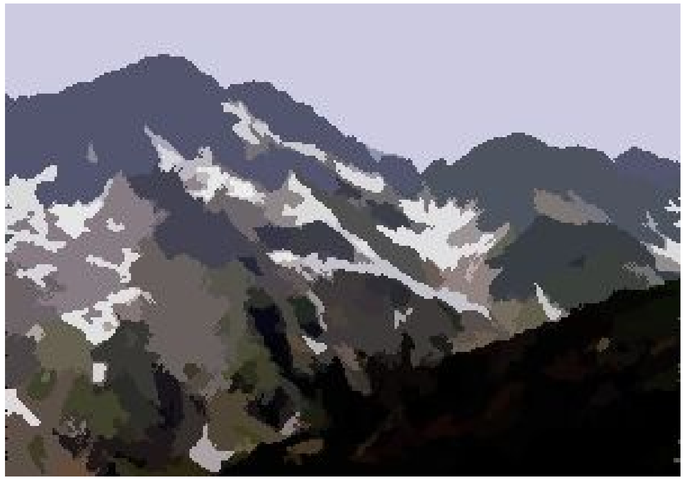




Slide credit: Svetlana Lazebnik

Mean-Shift Segmentation Results



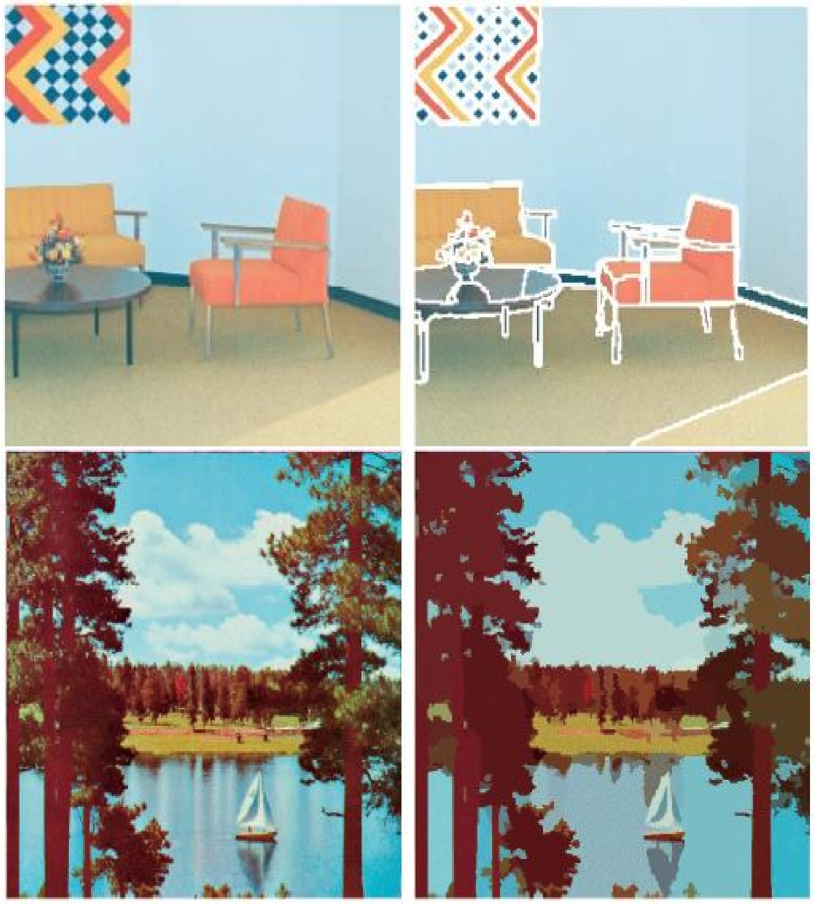


Slide credit: Svetlana Lazebnik

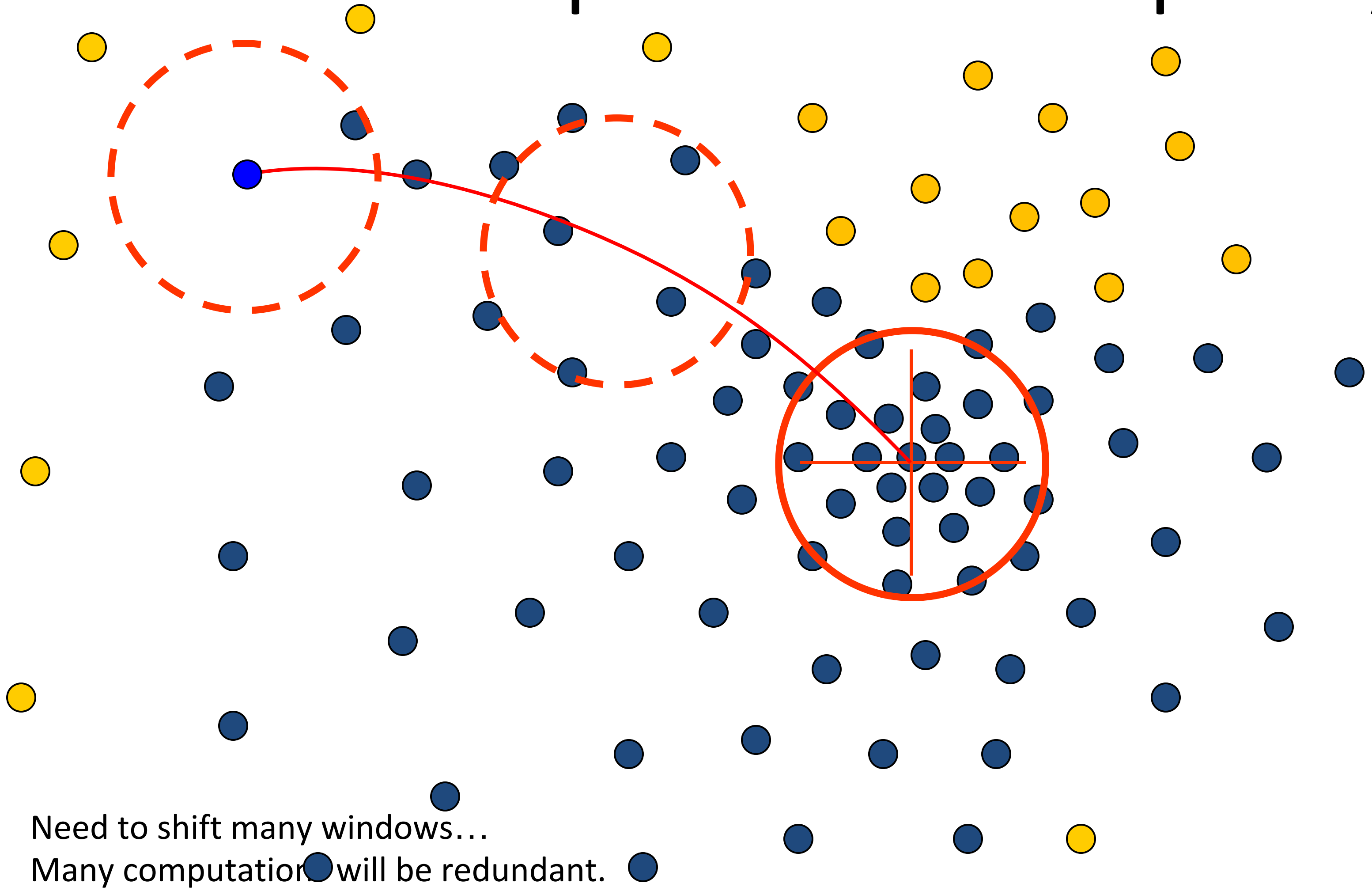
More Results

Slide credit: Svetlana Lazebnik

More Results



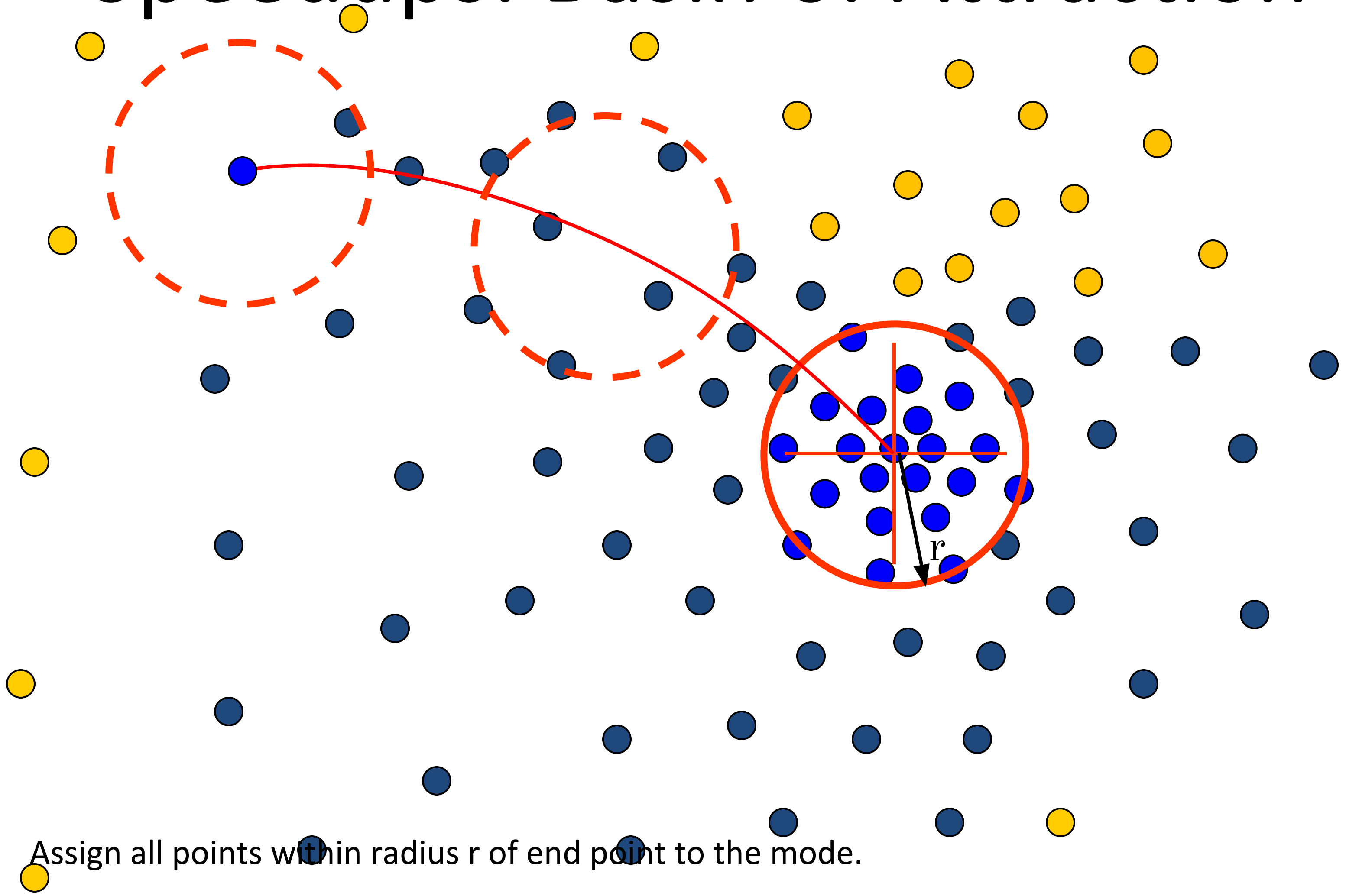
Problem: Computational Complexity



- Need to shift many windows…
- Many computations will be redundant.

Slide credit: Bastian Leibe

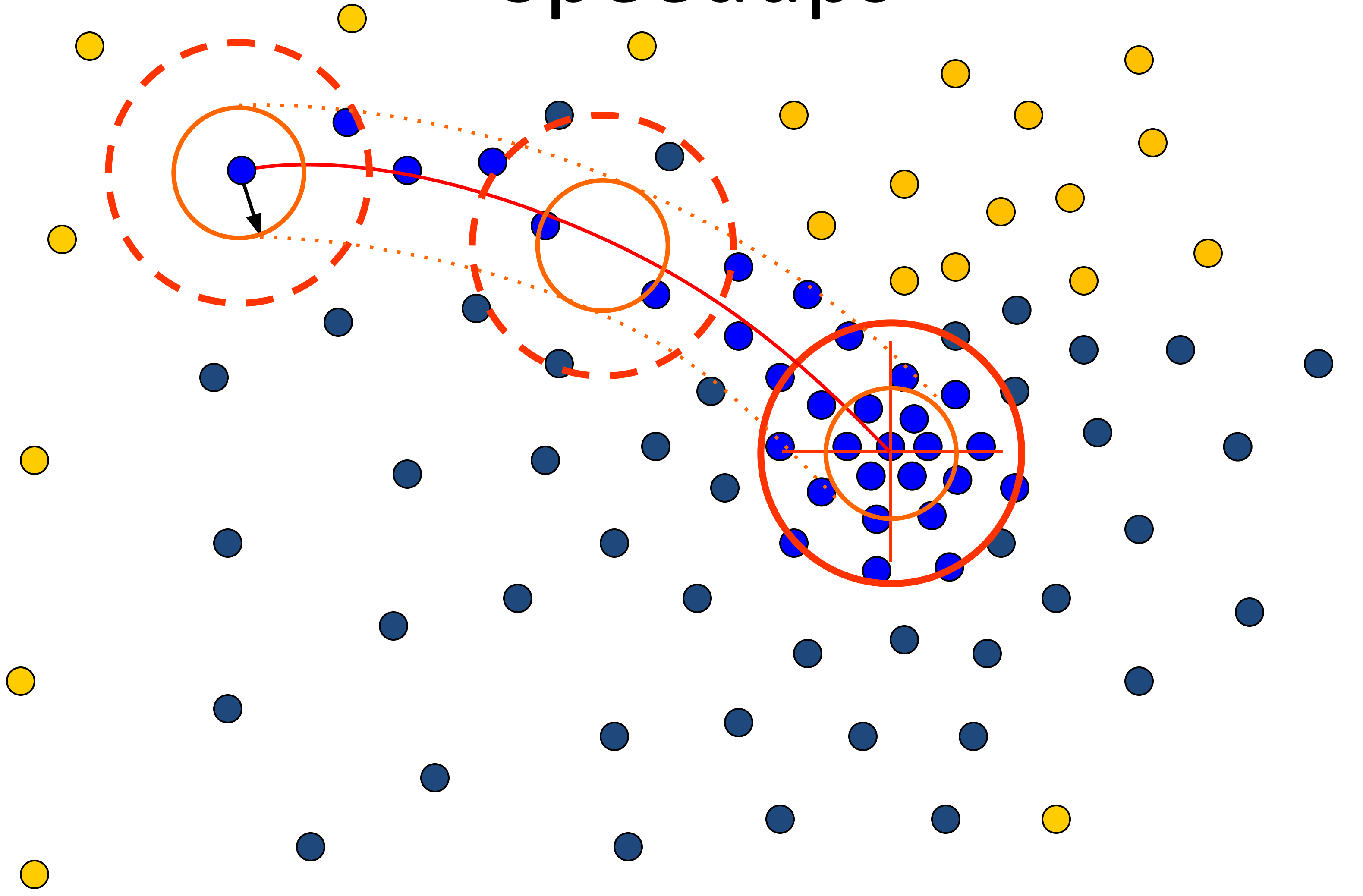
Speedups: Basin of Attraction



1. Assign all points within radius r of end point to the mode.

Slide credit: Bastian Leibe

Speedups



- Assign all points within radius r/c of the search path to the mode

Slide credit: Bastian Leibe

technical note

Mean-shift Algorithm

Comaniciu & Meer, 2002

Given n data points $\mathbf{x}_i \in \mathbb{R}^d$, the multivariate kernel density estimate using a radially symmetric kernel[1] (e.g., Epanechnikov and Gaussian kernels), $K(\mathbf{x})$, is given by,

$$\hat{f}_K = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right), \tag{1}$$

where h (termed the *bandwidth* parameter) defines the radius of kernel. The radially symmetric kernel is defined as,

$$K(\mathbf{x}) = c_k k(\|\mathbf{x}\|^2), \tag{2}$$

where c_k represents a normalization constant. Taking the gradient of the density estimator (1) and some further algebraic manipulation yields,

$$\nabla \hat{f}(\mathbf{x}) = \underbrace{\frac{2c_{k,d}}{nh^{d+2}} \left[\sum_{i=1}^{n} g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)\right]}_{\text{term 1}} \underbrace{\left[\frac{\sum_{i=1}^{n} \mathbf{x}_i g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)} - \mathbf{x}\right]}_{\text{term 2}}, \tag{3}$$

where $g(x) = -k'(x)$ denotes the derivative of the selected kernel profile. The first term is proportional to the density estimate at $\mathbf{x}$ (computed with the kernel $G = c_g g(\|\mathbf{x}\|^2)$). The second term, called the *mean shift* vector, $\mathbf{m}$, points toward the direction of maximum increase in density and is proportional to the density gradient estimate at point $\mathbf{x}$ obtained with kernel K. The mean shift procedure for a given point $\mathbf{x}_i$ is as follows: (see Fig. 1):

1. Compute the mean shift vector $\mathbf{m}(\mathbf{x}_i^t)$.
2. Translate density estimation window: $\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \mathbf{m}(\mathbf{x}_i^t)$.
3. Iterate steps 1. and 2. until convergence, i.e., $\nabla f(\mathbf{x}_i) = 0$.

technical note

Mean-shift Algorithm

Comaniciu & Meer, 2002

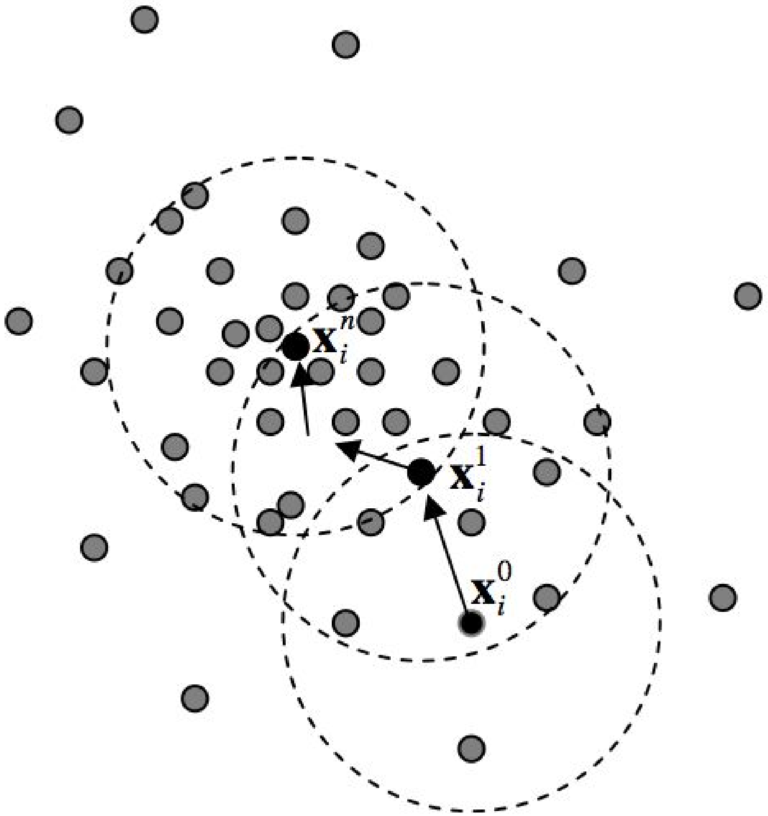


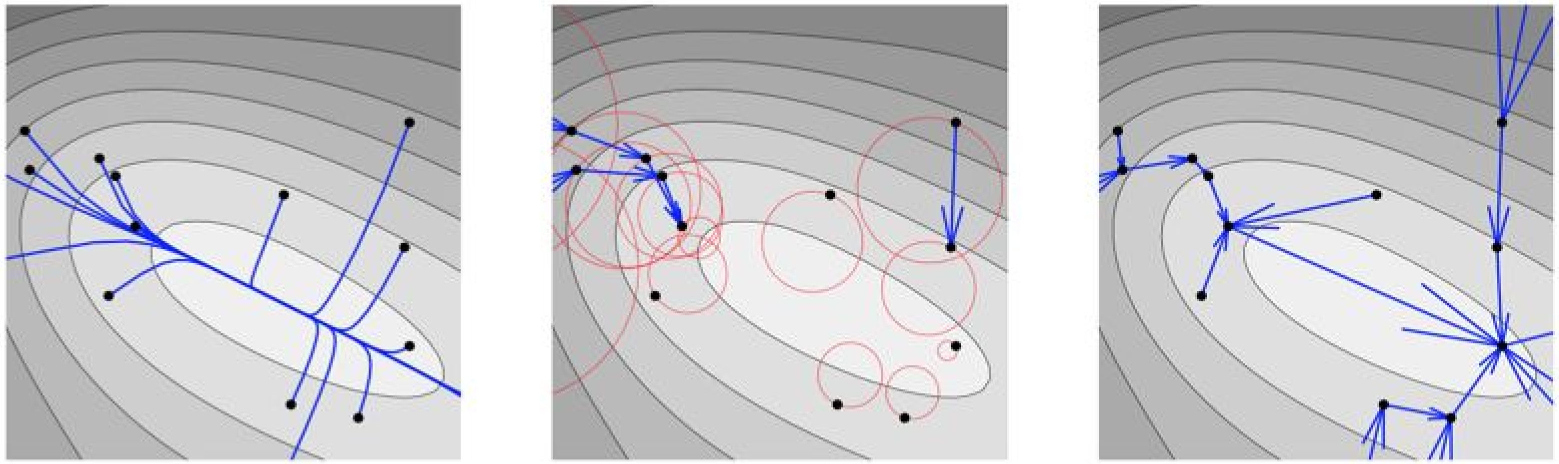
Figure 1: Mean shift procedure. Starting at data point $\mathbf{x}_i$, run the mean shift procedure to find the stationary points of the density function. Superscripts denote the mean shift iteration, the shaded and black dots denote the input data points and successive window centres, respectively, and the dotted circles denote the density estimation windows.

Summary Mean-Shift

- Pros
  - General, application-independent tool
  - Model-free, does not assume any prior shape (spherical, elliptical, etc.) on data clusters
  - Just a single parameter (window size h)
    - h has a physical meaning (unlike k-means)
  - Finds variable number of modes
  - Robust to outliers
- Cons
  - Output depends on window size
  - Window size (bandwidth) selection is not trivial
  - Computationally (relatively) expensive (~2s/image)
  - Does not scale well with dimension of feature space

Slide credit: Svetlana Lazebnik

Medoid-Shift & Quick-Shift



-
  - does not need the gradient or quadratic lower bound
  - only one step has to be computed for each point: simply moves each point to the nearest neighbor for which there is an increment of the density
  - there is no need for a stopping/merging heuristic
  - the data space X may be non-Euclidean

[Vedaldi and Soatto, 2008]

What have we learned today

- K-means clustering
- Mean-shift clustering

IPython Notebook for SLIC and Quickshift