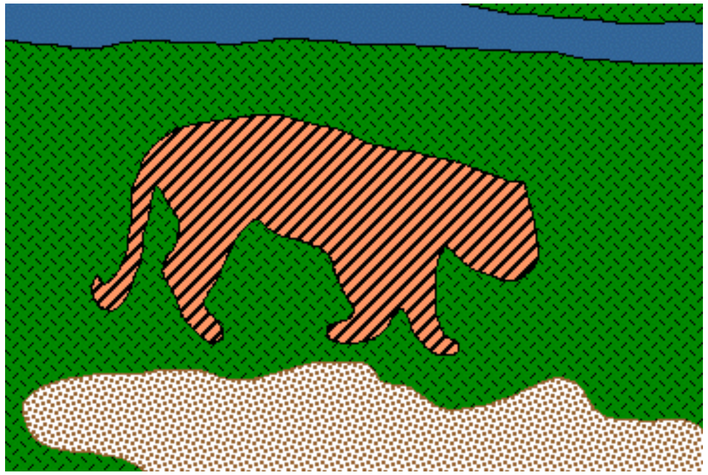
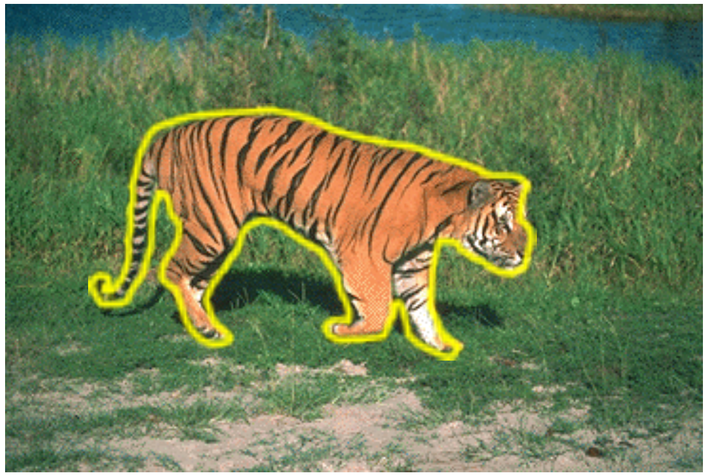
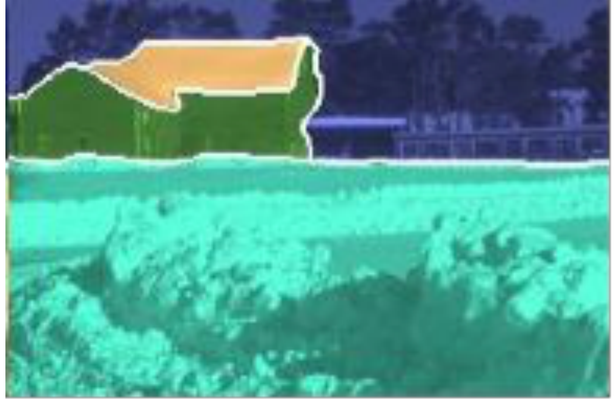
Segmentation

Gary Overett (Slides adapted from CMU 16-720 2014)







$k = 4$

$nc = .0017$

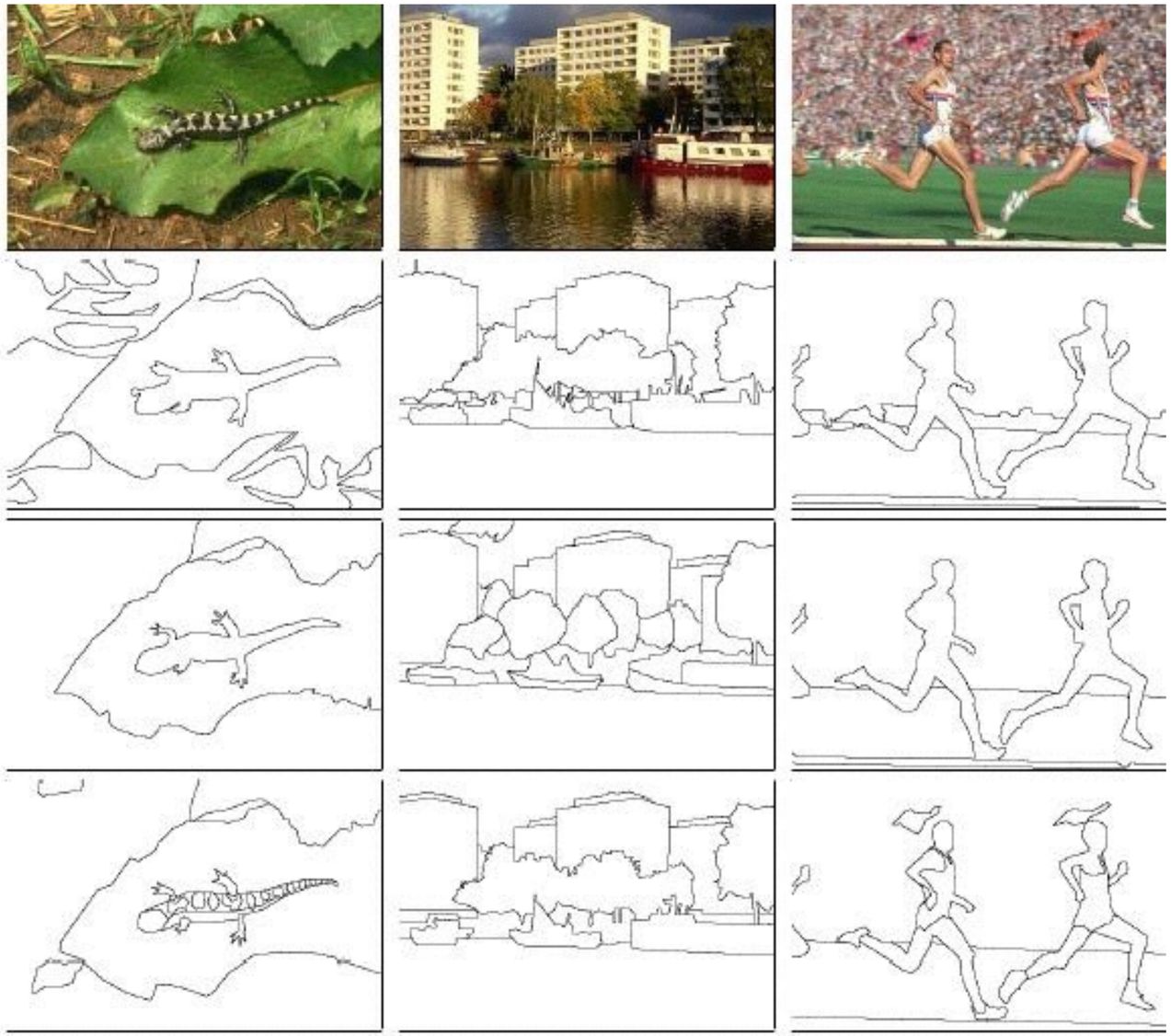
$k = 5$

$nc = .0060$

What is a “good” segmentation?

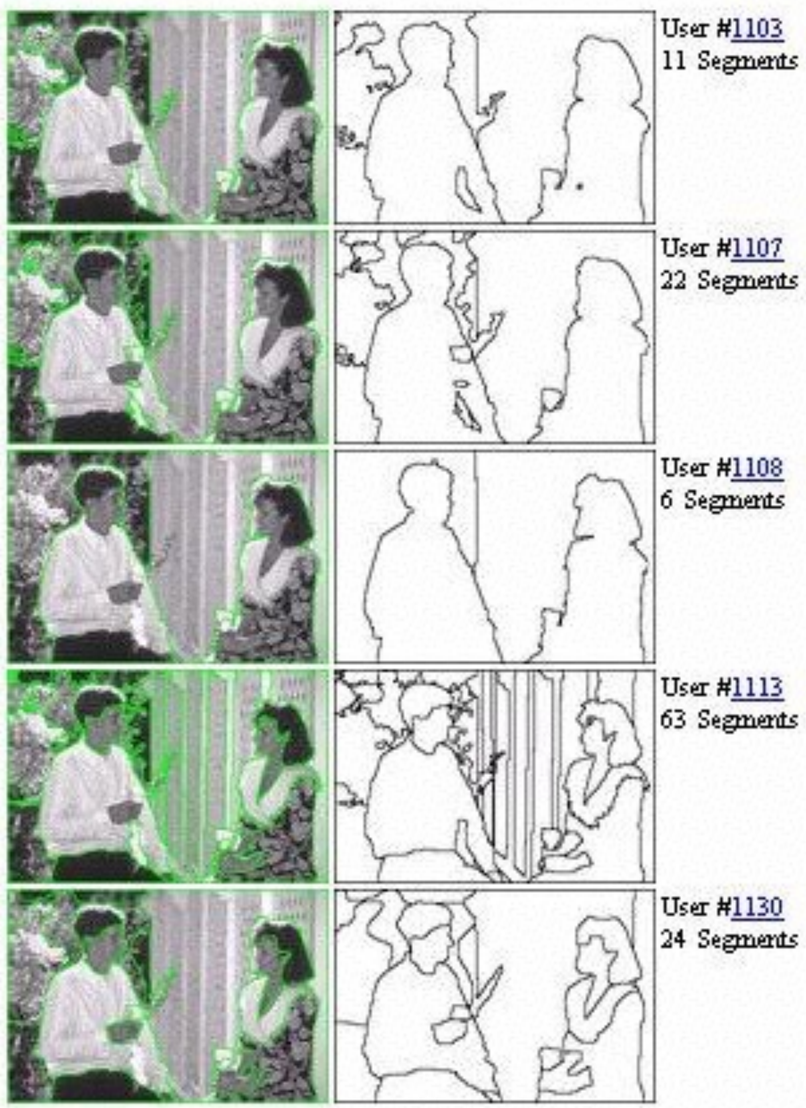
Key idea: Compare to human segmentation or to "ground truth"

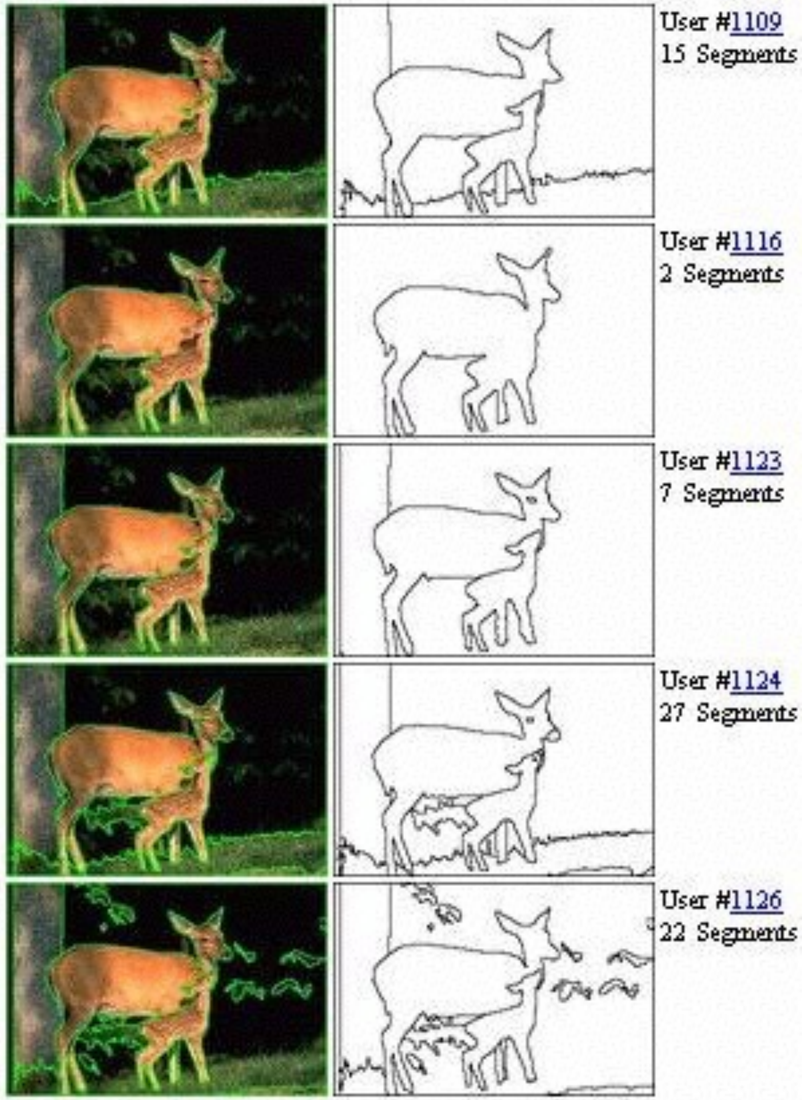
No objective definition of segmentation!



http://www.eecs.berkeley.edu/Research/Proje cts/CS/vision/grouping/resources.html

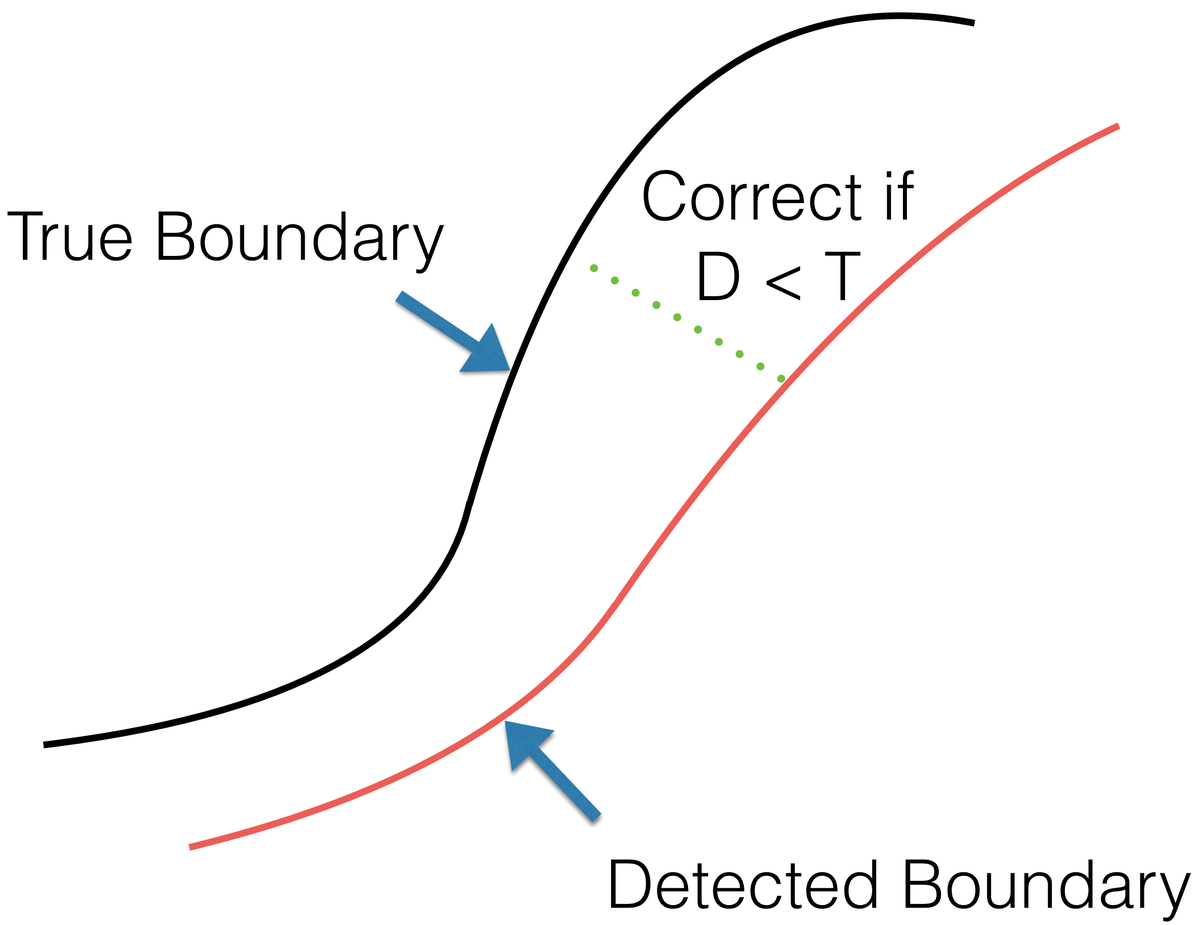
No objective definition of segmentation!

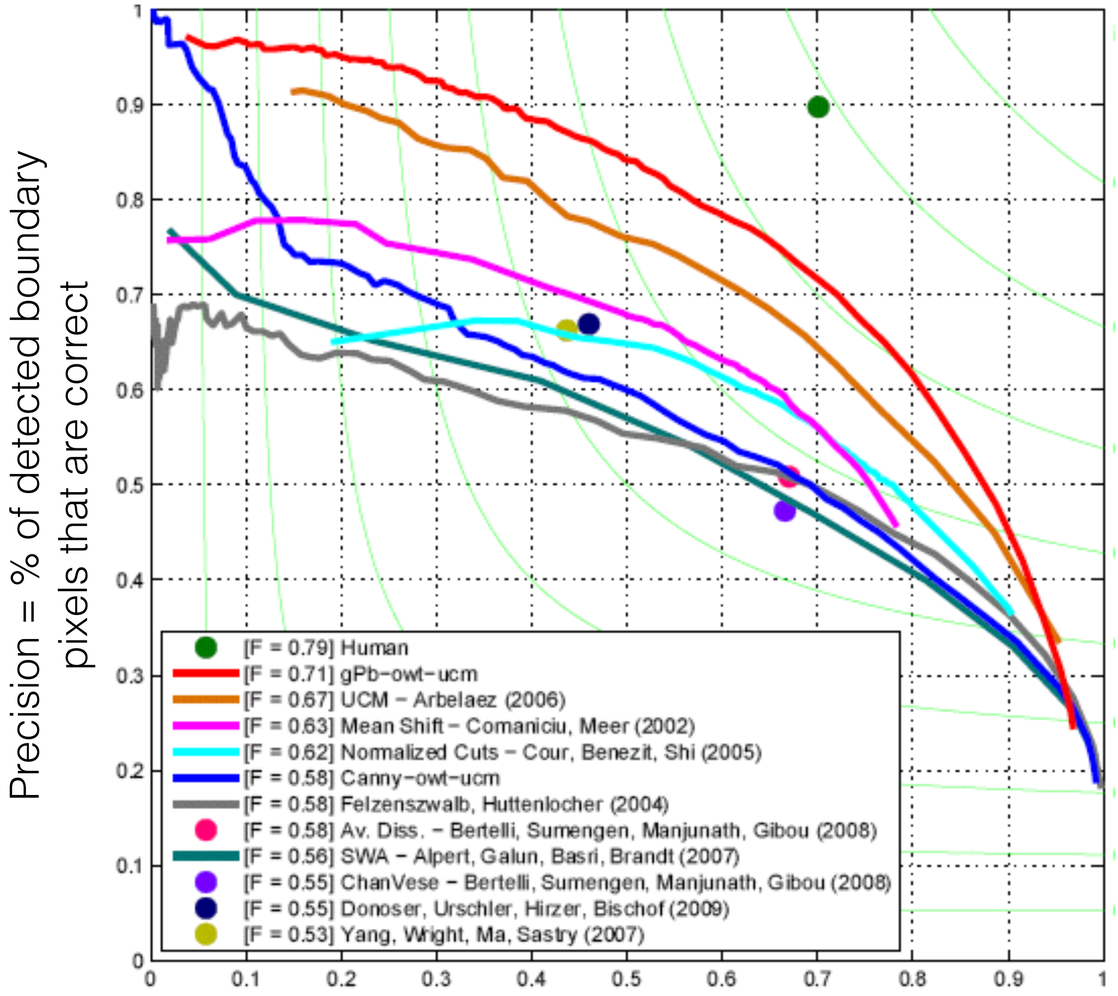




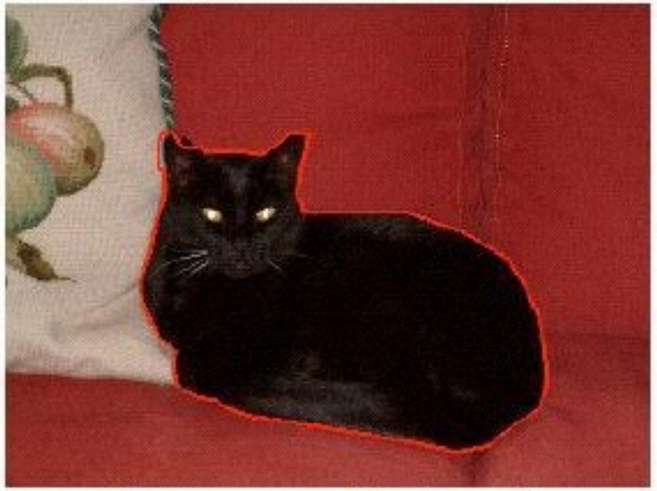
http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/BSDS300/html/dataset/images/color/317080.html

Evaluation : Boundary Agreement





Evaluation : Intersection over Union



Ground Truth

Segment #1

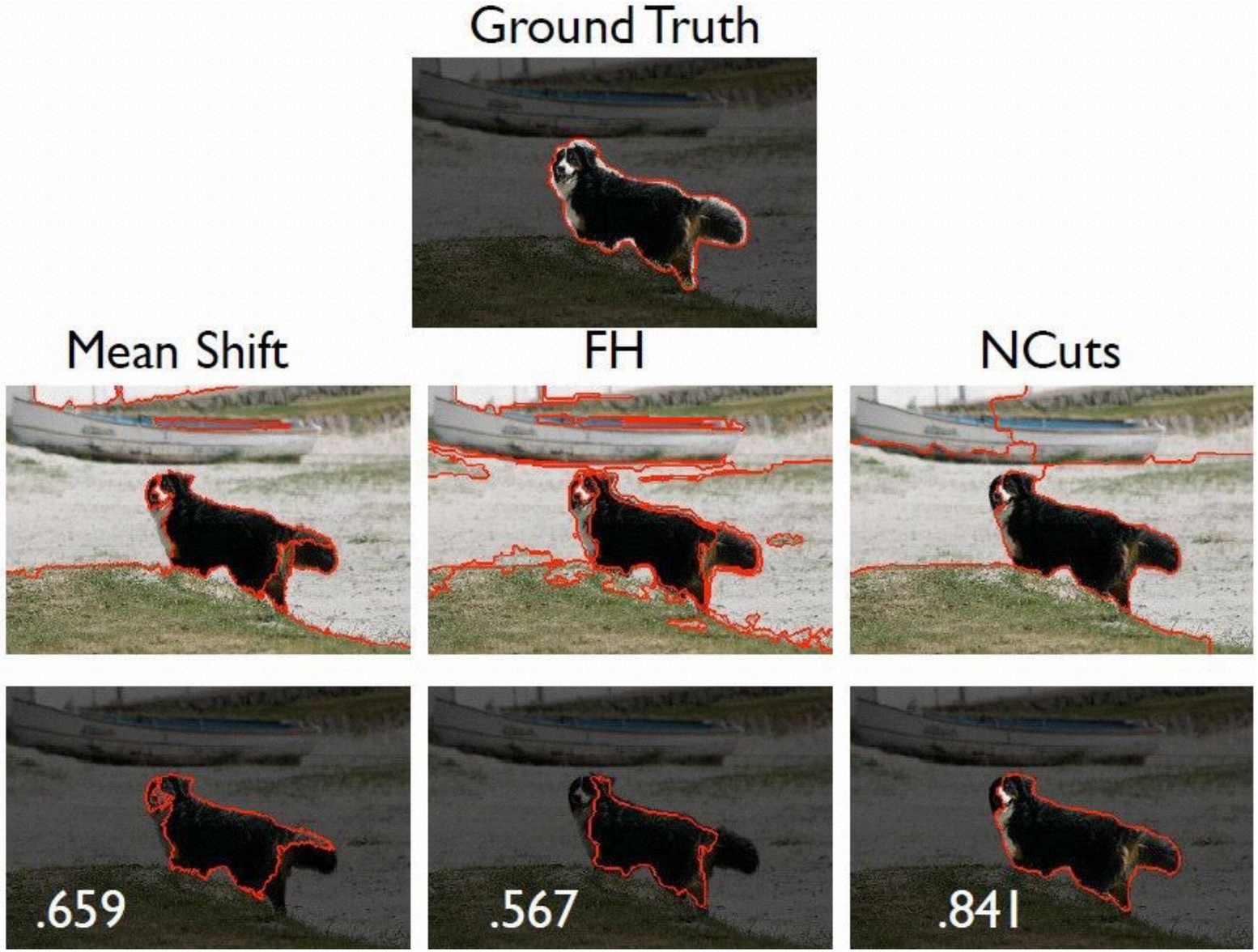
.825

Segment #2

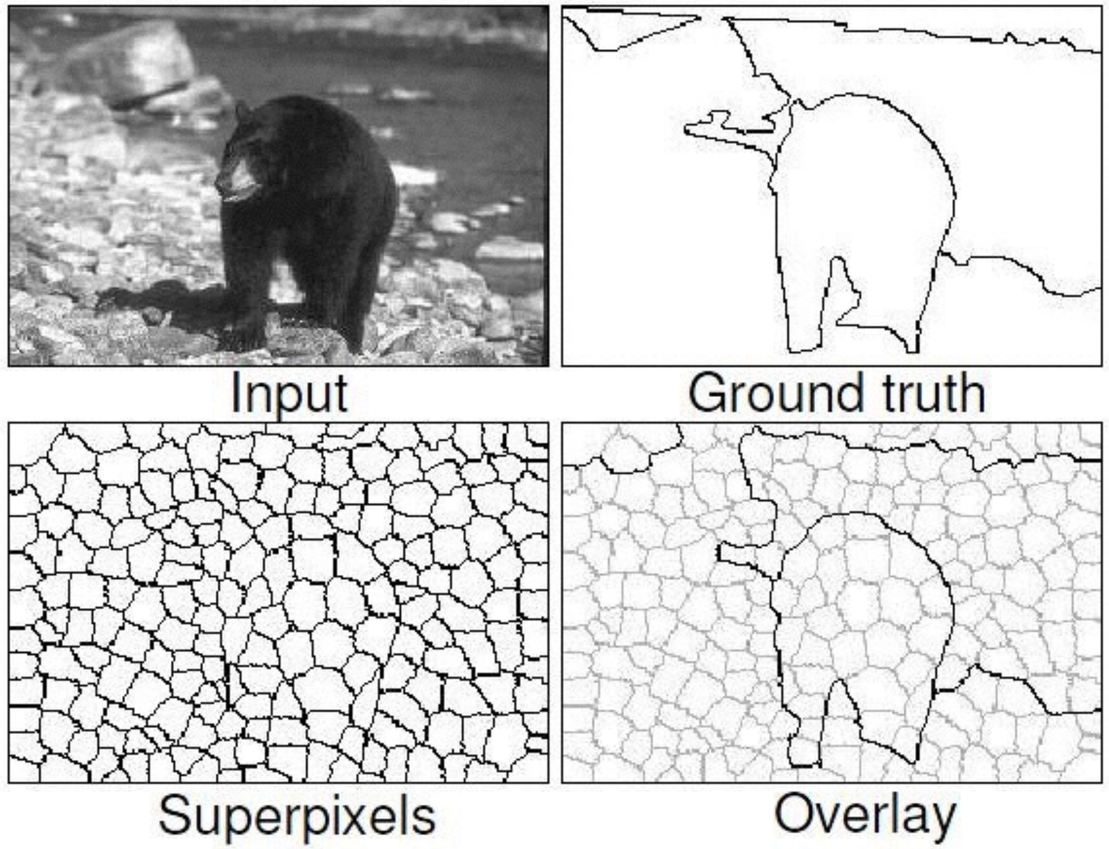
.892

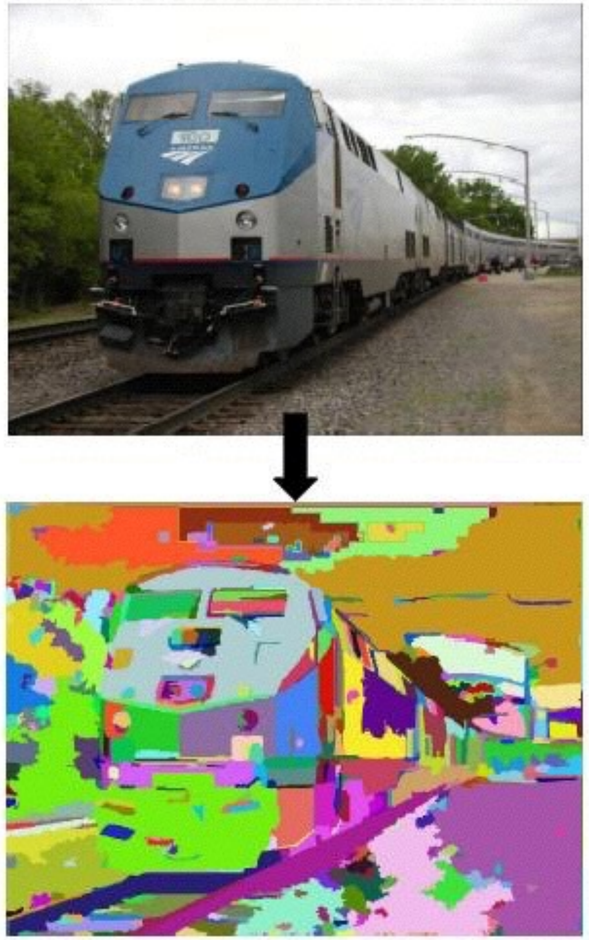
$$OS(S,G) = \frac{|S \cap G|}{|S \cup G|}$$

Evaluation : Intersection over Union



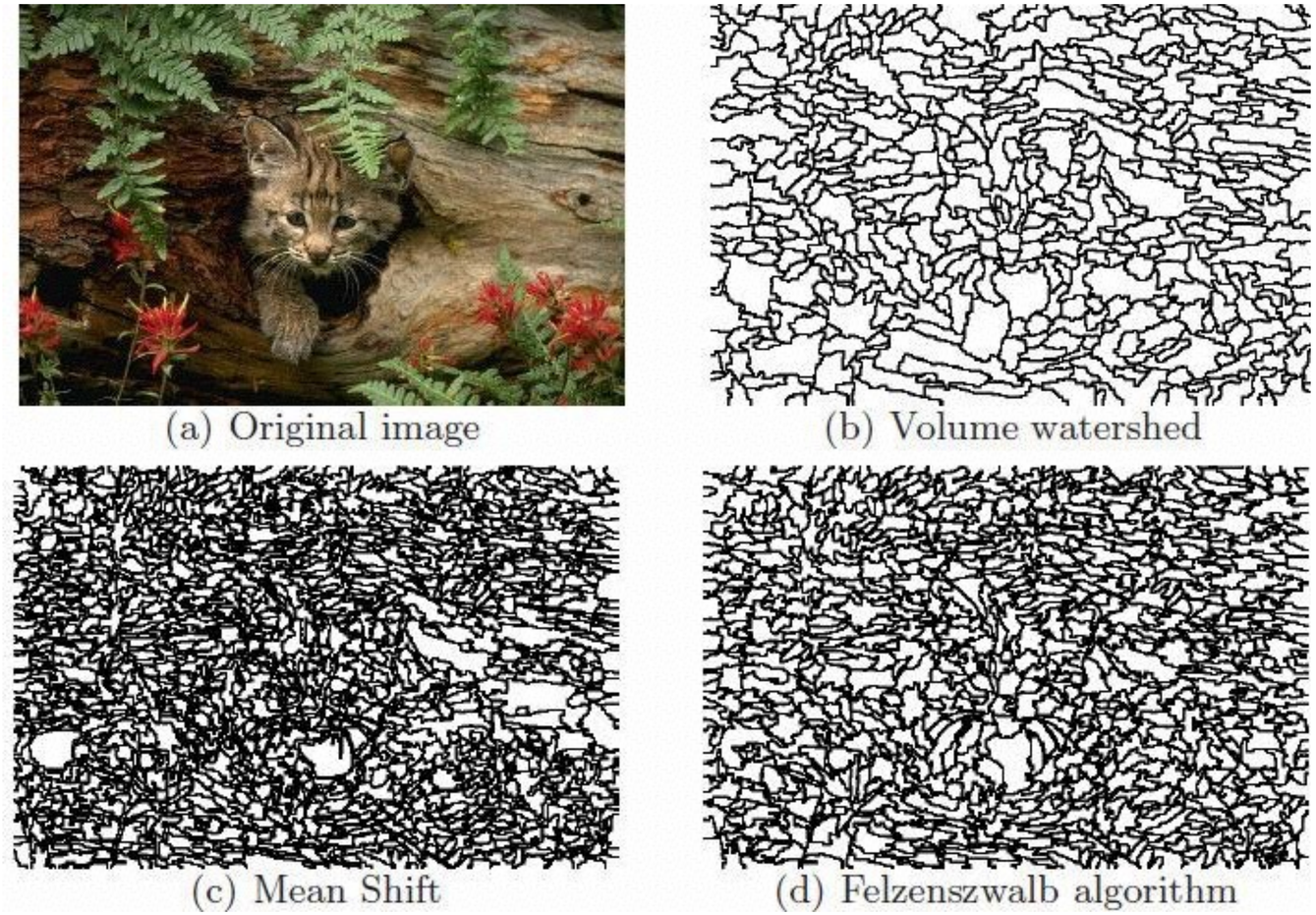
Key Idea : Superpixels





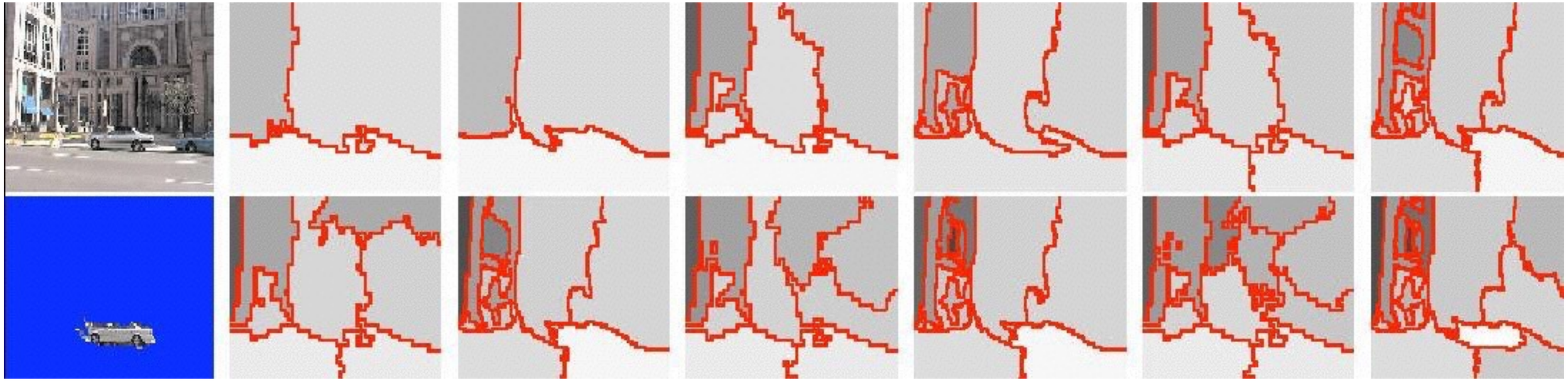
- Let's not even try to compute a "correct" segmentation
- Let's be content with an oversegmentation in which each region is very likely (formal guarantees are hard) to be uniform

Key Idea : Superpixels



- Example from: How Do Superpixels Affect Image Segmentation?
- Progress in Pattern Recognition, Image Analysis and Applications. Springer LNCS. Volume 5197/2008.

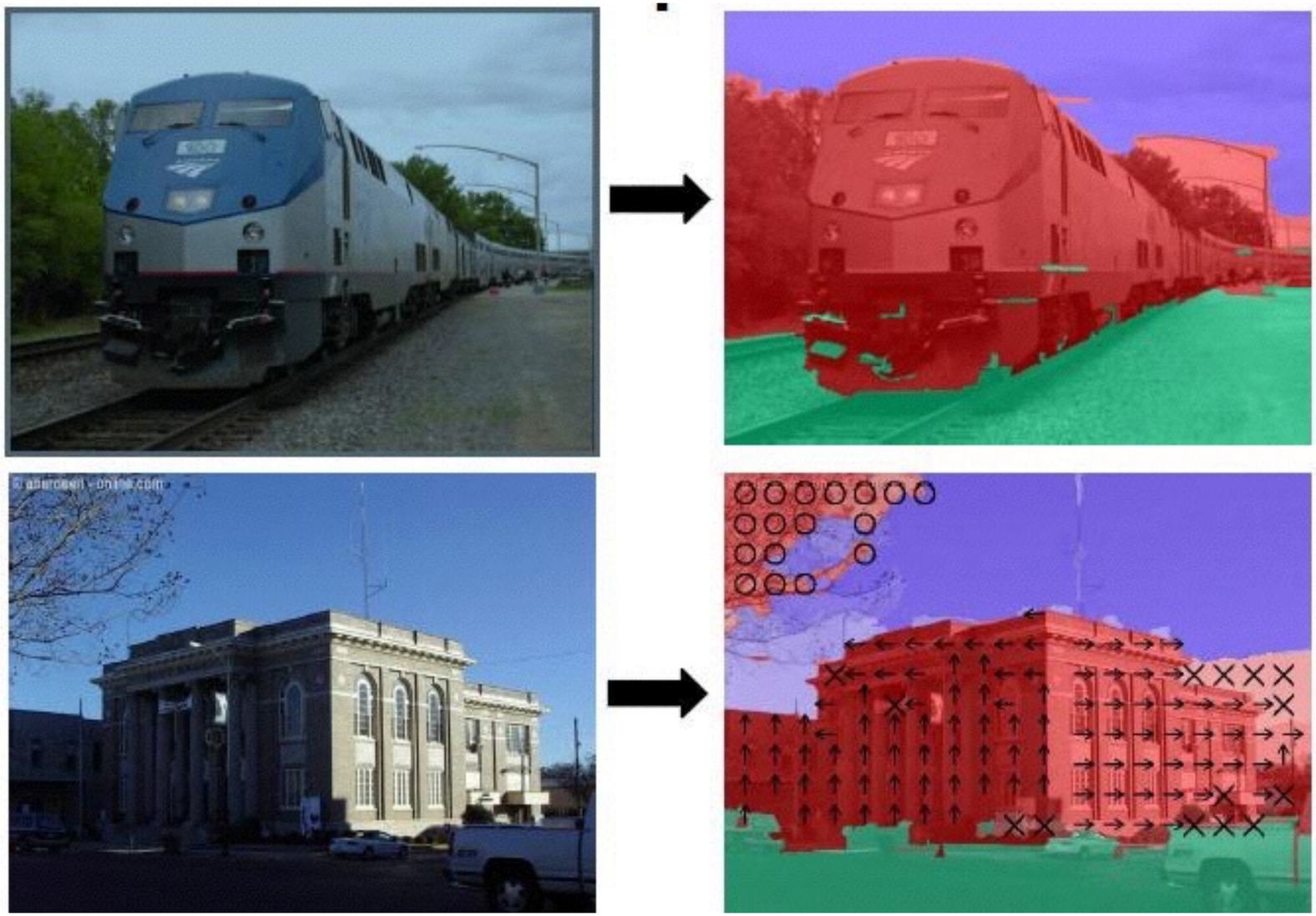
Key Idea : Multiple Segmentations



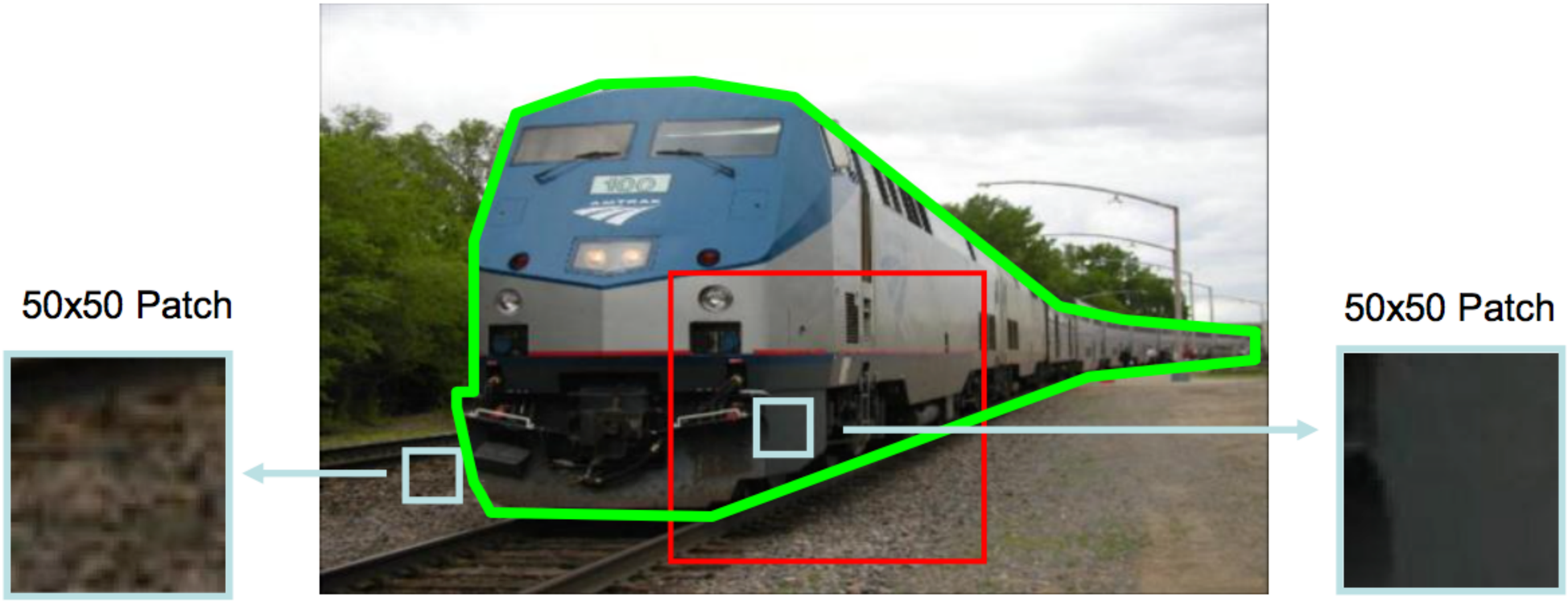
- Generate many segmentations of the same image
- Even though many regions are “wrong”, some consensus should emerge

Example: Improving Spatial Support for Objects via Multiple Segmentations Tomasz Malisiewicz and Alexei A. Efros. British Machine Vision Conference (BMVC), September, 2007.

Multiple Segmentations : Example

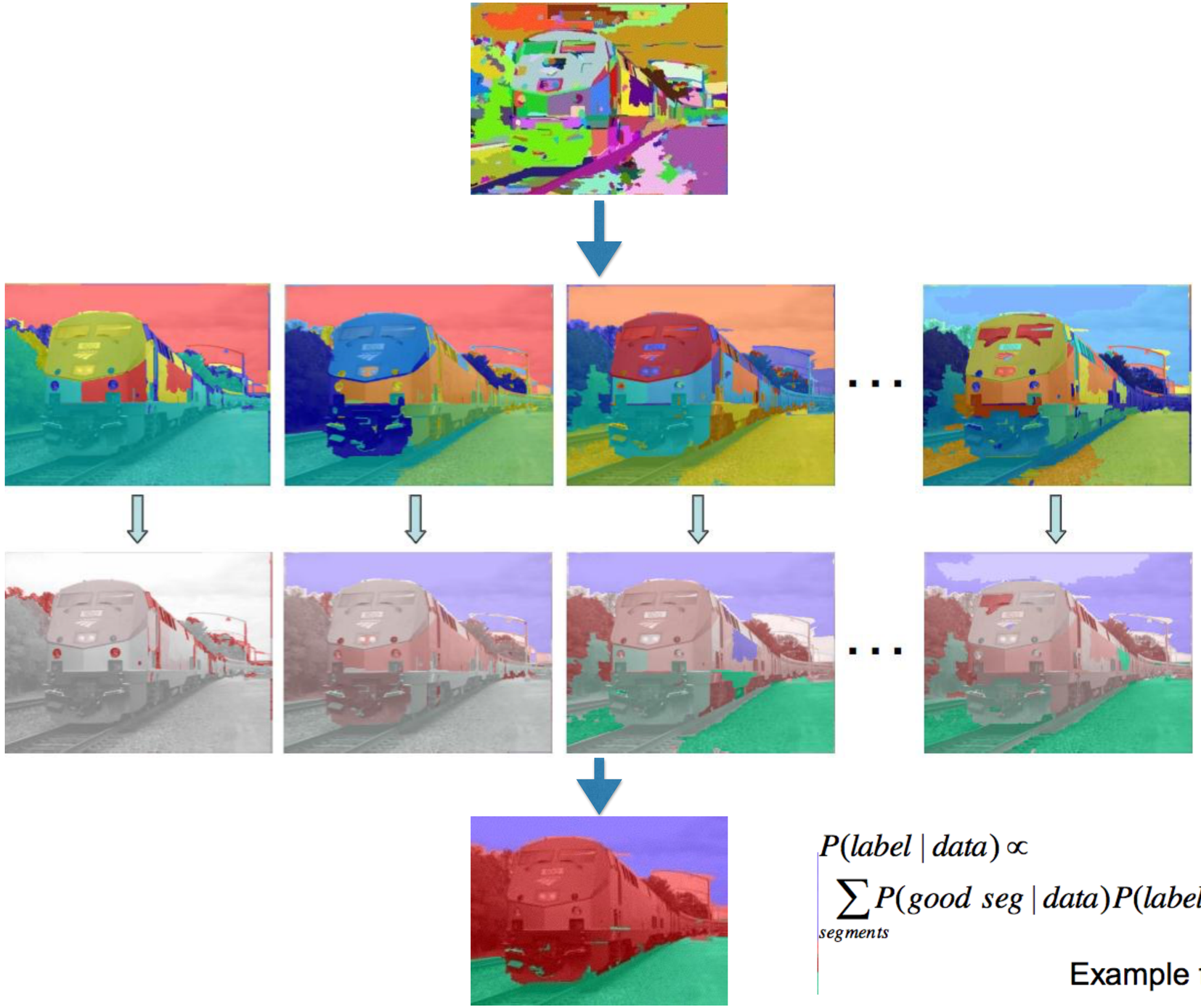


- Task: Regions -> Features -> Labels (horizontal, vertical, sky, etc.)



- Chicken and egg problem:
  - If we knew the regions, we could compute the features and label the right regions
  - But to know the right regions we need to know the labels!
- Solution:
  - Generate lots of segmentations
  - Combine the classifications to get consensus

Recovering Surface Layout from an Image. D. Hoiem, A.A. Efros, and M. Hebert. IJCV, Vol. 75, No. 1, October 2007.



$$P(label \mid data) \propto \sum_{segments} P(good\ seg \mid data) P(label \mid good\ seg, data$$

Example from D. Hoiem

Summary so far…

- Match ground truth (no objective definition)
- Superpixels = oversegmentation
- Using multiple segmentations

Main Approaches

- Clustering
- Mean shift
- Spectral techniques
- Graph-based techniques
- Segmentation as boundary detection

Main Approaches

- Clustering
- Mean shift
- Spectral techniques
- Graph-based techniques
- Segmentation as boundary detection

Segmentation as Clustering (Informally)

- Given a set of features $\{x_i\}$
- Given a "distance" between features $d(x,x')$
- Find a partition $\{C_j\}$ such that:
  - $d(x_i,x_j)$ is "small" if x_i and x_j belong to the same group
  - $d(x_i,x_j)$ is large if x_i and x_j belong to different groups

- Examples
  - Color segmentation: x_i is color of pixel i. Subsets are connected regions of "similar" color.
  - Texture segmentation: x_i is vector of responses to texture filters at pixel i. Subsets are regions of "similar" texture. $x_i = [F_1*I,..., F_n*I]$
  - Contour grouping: x_i is edge element. Subsets are sets of edges that form a smooth connected contour.

K-Means Clustering

If the number of clusters is know a-priori:

$$\Phi(\text{clusters}, \text{data}) = \sum_{j\,\text{cluster}} \sum_{i \in \text{cluster } j} \left\| \mathbf{x}_i - \mathbf{c}_j \right\|^2$$

1. Given $\{\mathbf{c}_j\}$, assign each $\mathbf{x}_i$ to the closest j
2. Compute the centers $\mathbf{c}_j = 1/N \, \Sigma \, \mathbf{x}_i$

Image

Clusters on intensity

Feature = intensity

Clusters on color

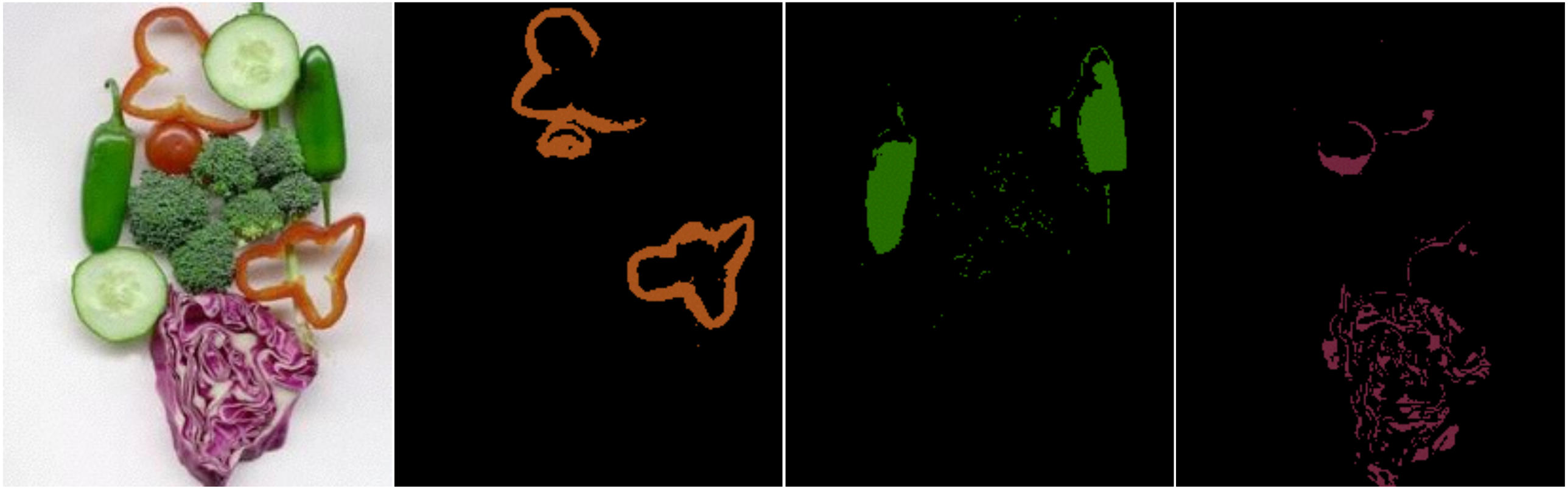
Feature = [r g b]

K-means clustering using intensity alone and color alone

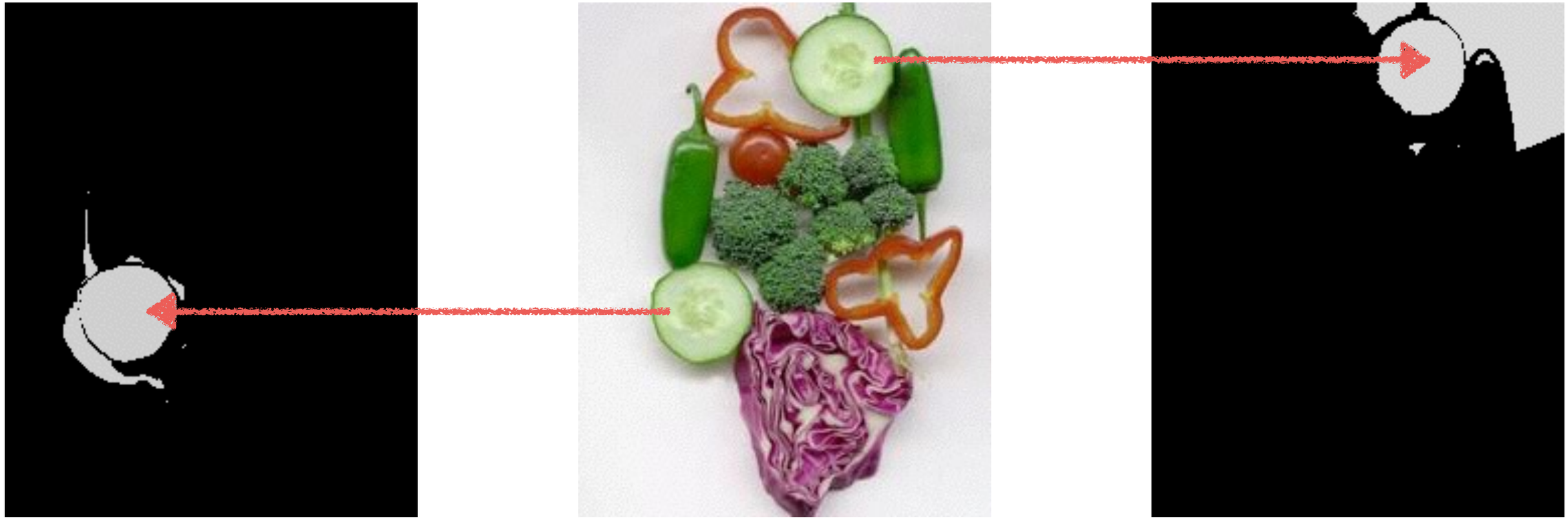
Image

Clusters on color

K-means using color alone, 11 segments



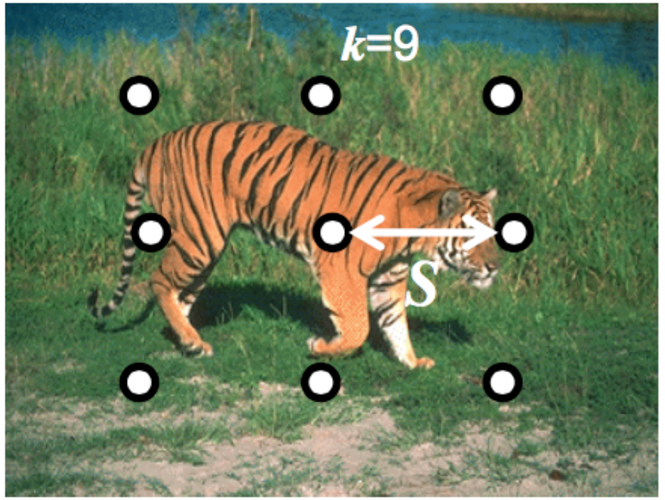
K-means using color alone, 11 segments. Feature = [r g b]



K-means using color and position, 20 segments Feature = [r g b x y]

State of the Art : SLIC Superpixels

- Find k superpixels such that
  – Respect boundaries
  – Are spatially roughly equally sized



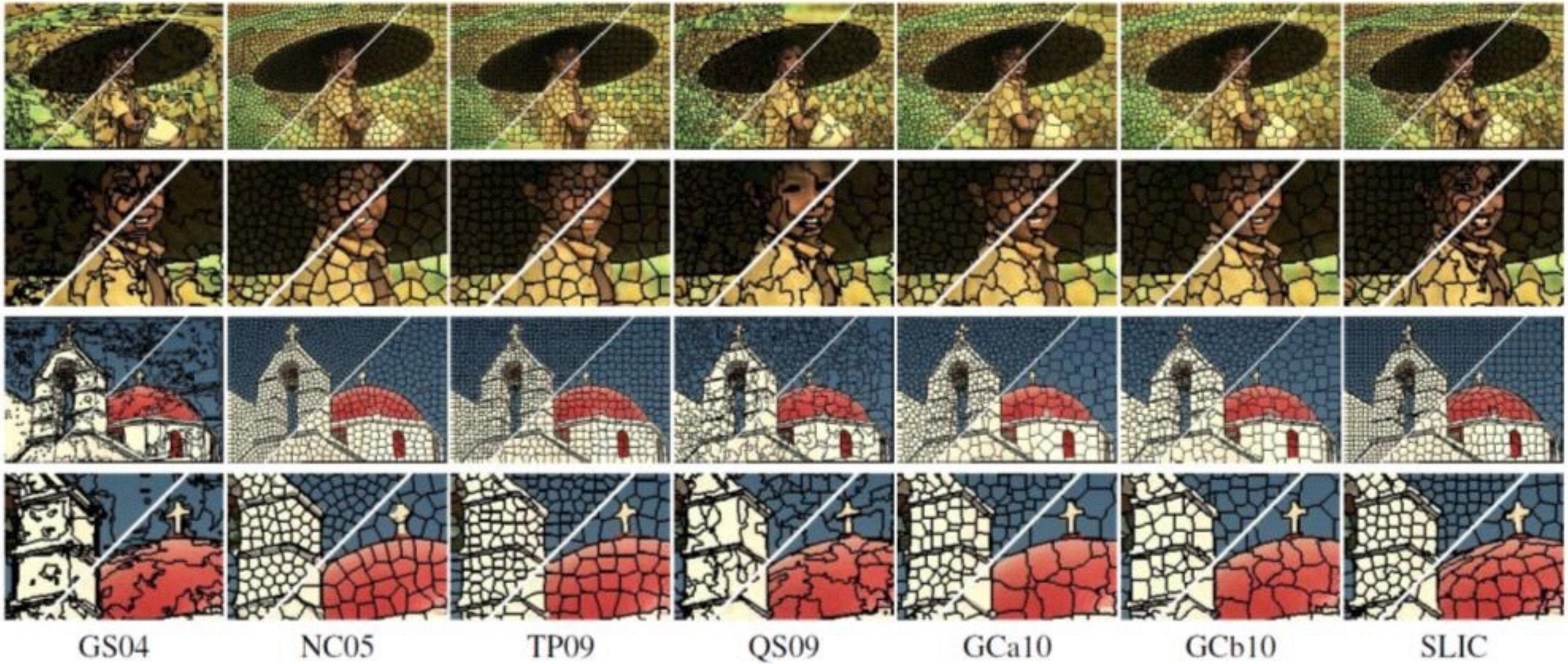
Select $\{\mathbf{c}_j\}$ on a regular grid of k centers

0. Initialize cluster centers $\{\mathbf{c}_j\}$

1. Given $\{\mathbf{c}_j\}$, assign each $\mathbf{x}_i$ to the closest j
2. Compute the centers $\mathbf{c}_j = 1/N \Sigma\, \mathbf{x}_i$

Search only in a neighborhood $2S$x$2S$ neighborhood around i

http://ivrg.epfl.ch/supplementary_material/RK_SLICSuperpixels/index.html

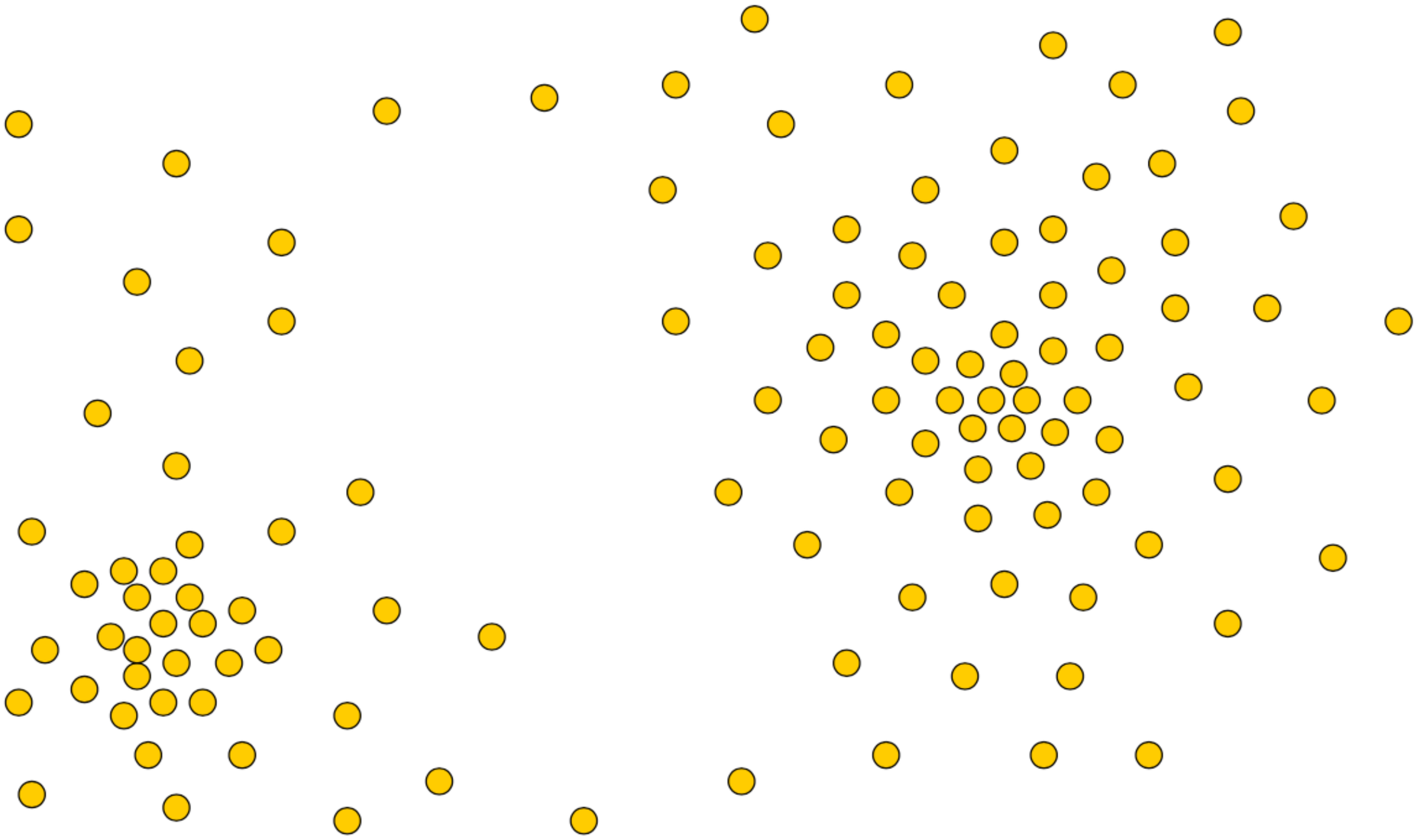


Main Approaches

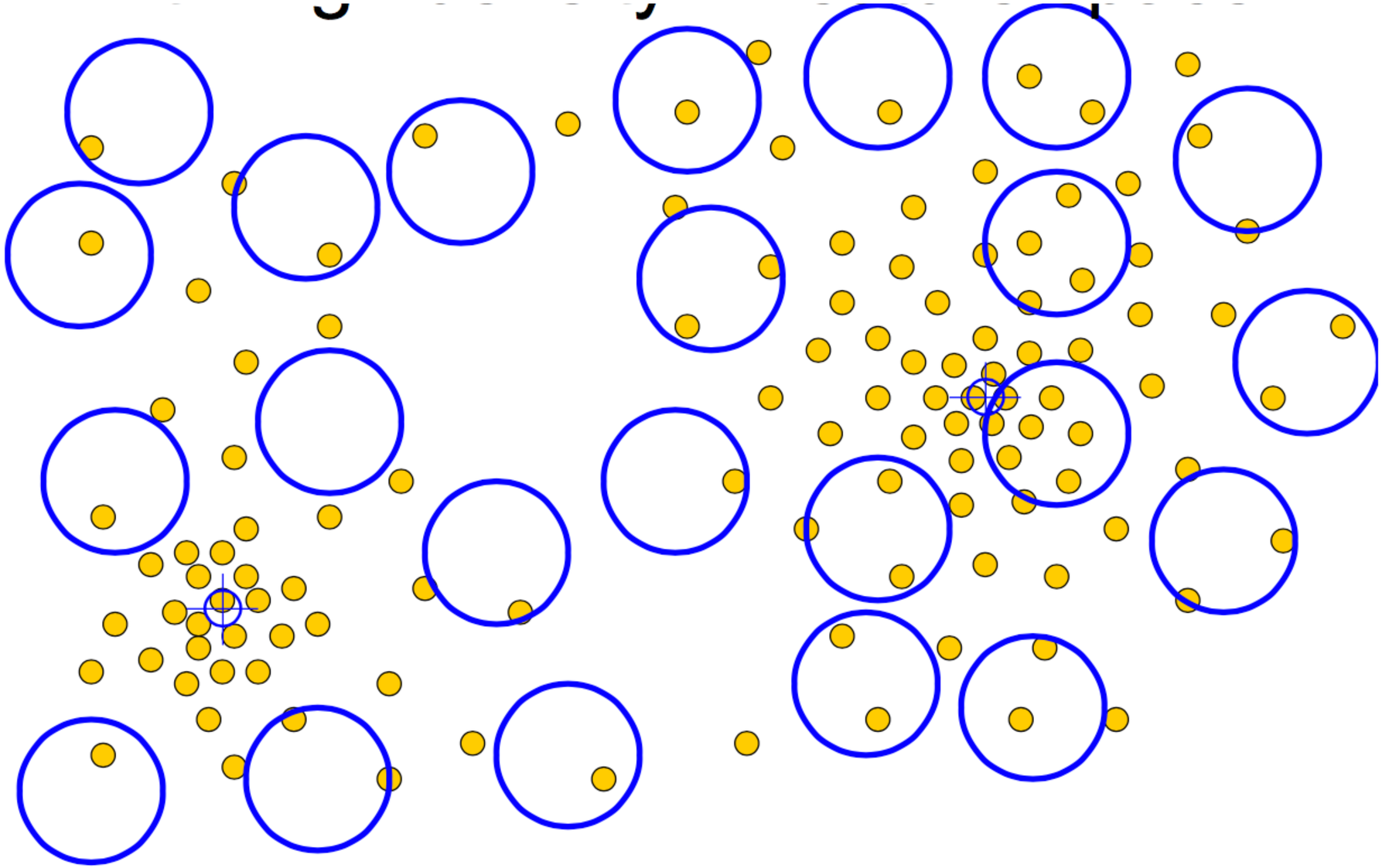
- Clustering
- Mean shift
- Spectral techniques
- Graph-based techniques
- Segmentation as boundary detection

Segmentation: The Mean-Shift Algorithm (hw3 warning)

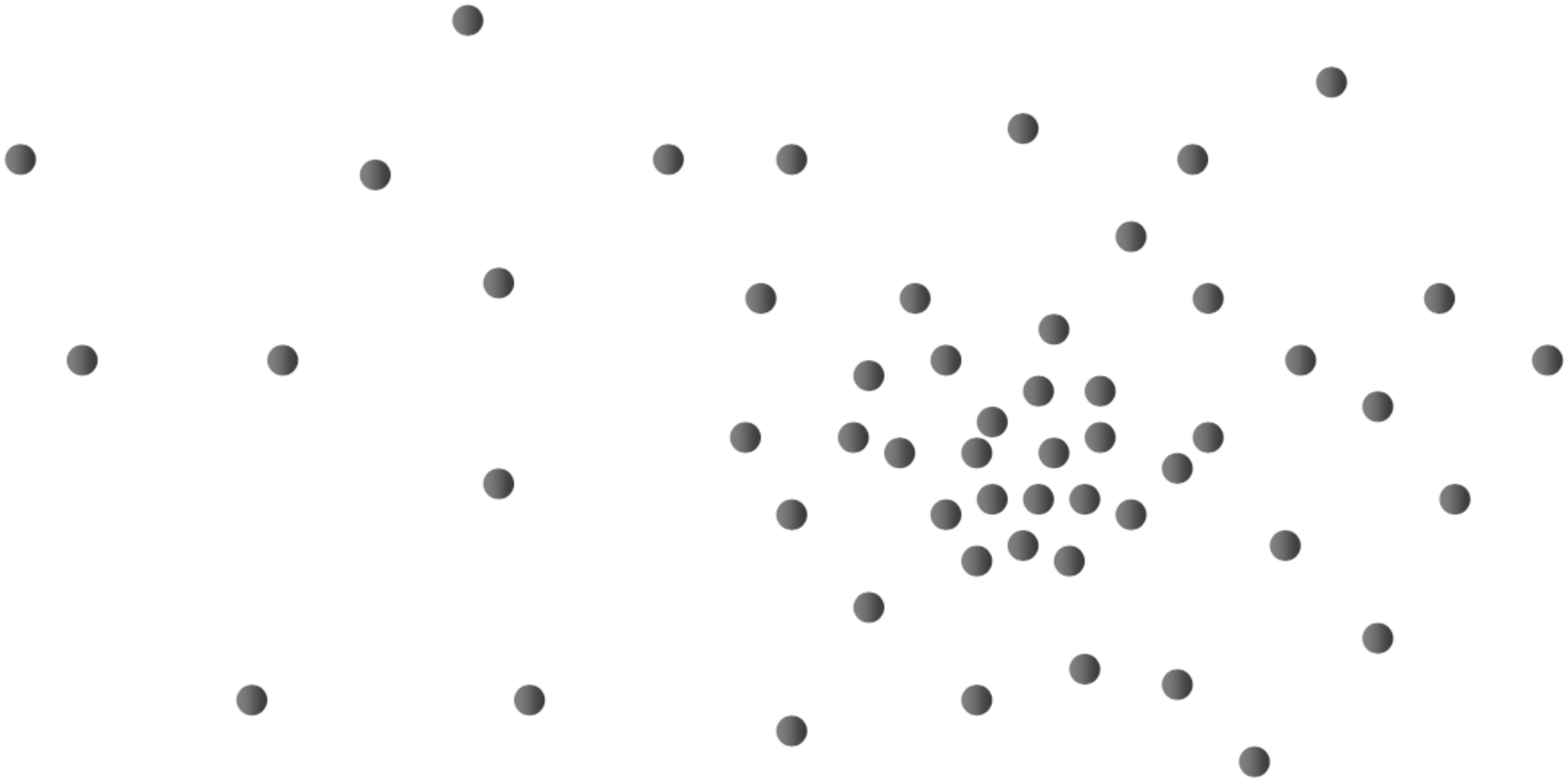
Segmentation as finding places of high density in feature space



Segmentation as finding places of high density in feature space

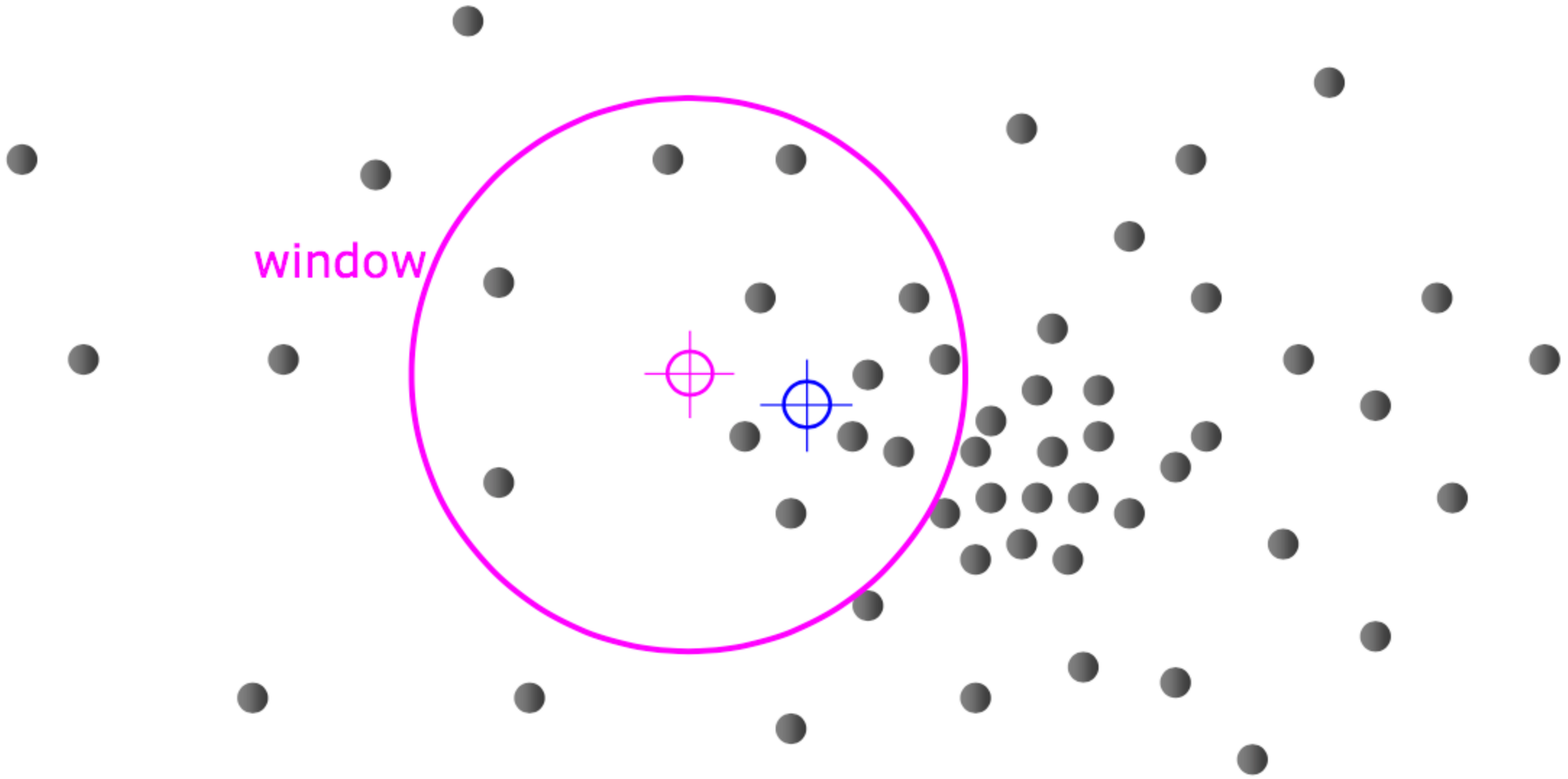


Segmentation as finding places of high density in feature space

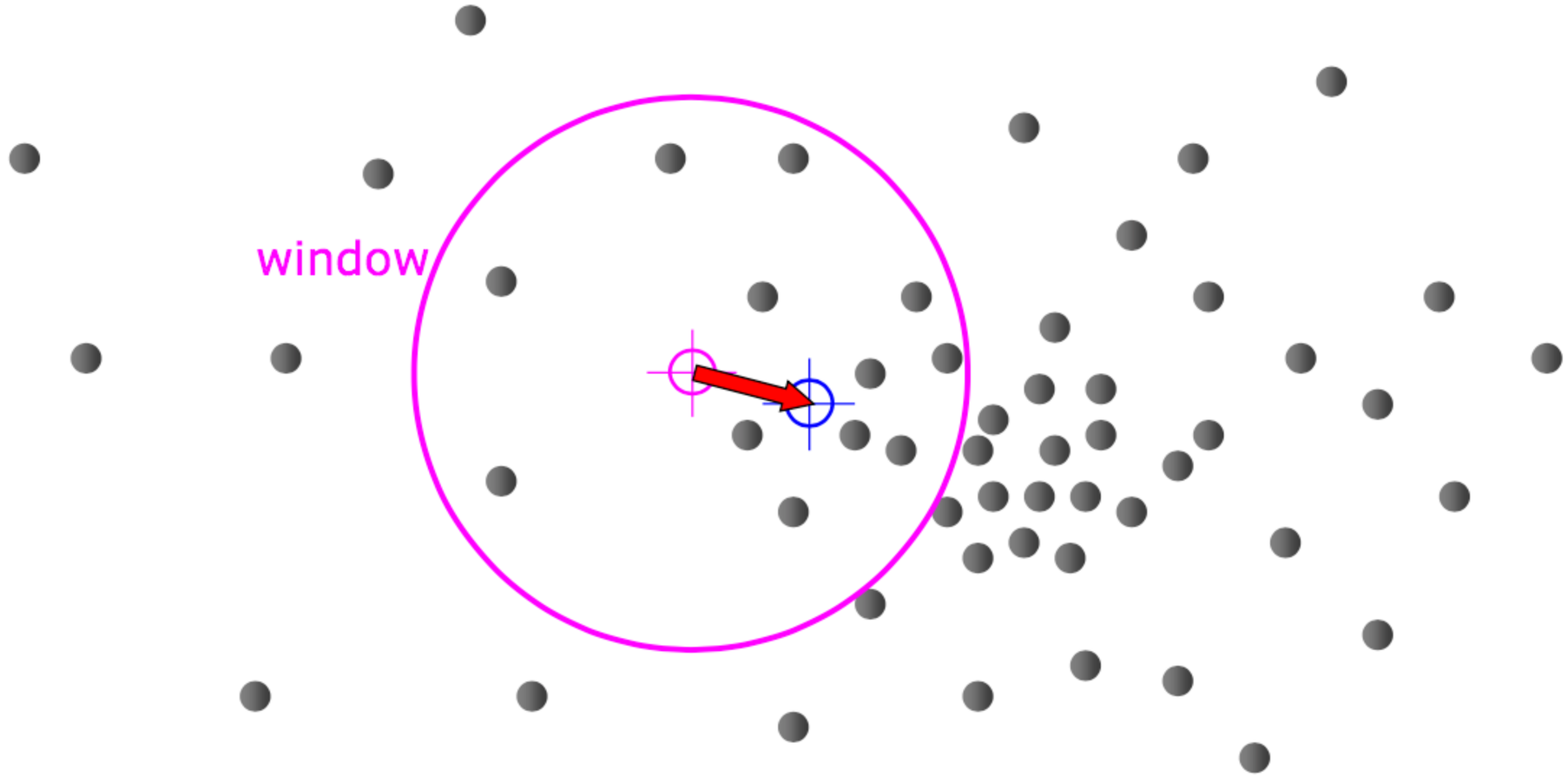


Example from: Ukrainitz&Sarel, Weizmann

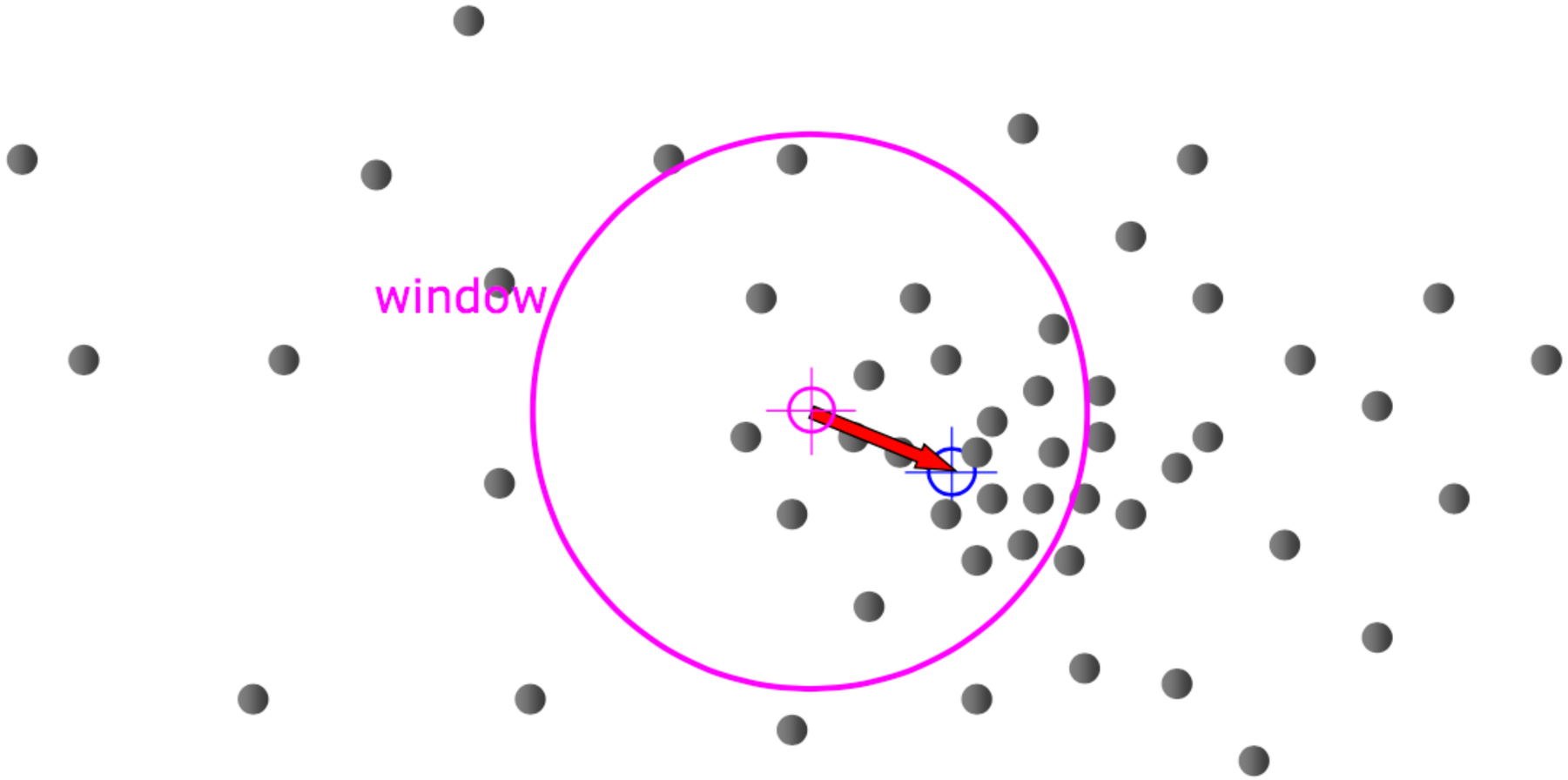
Segmentation as finding places of high density in feature space



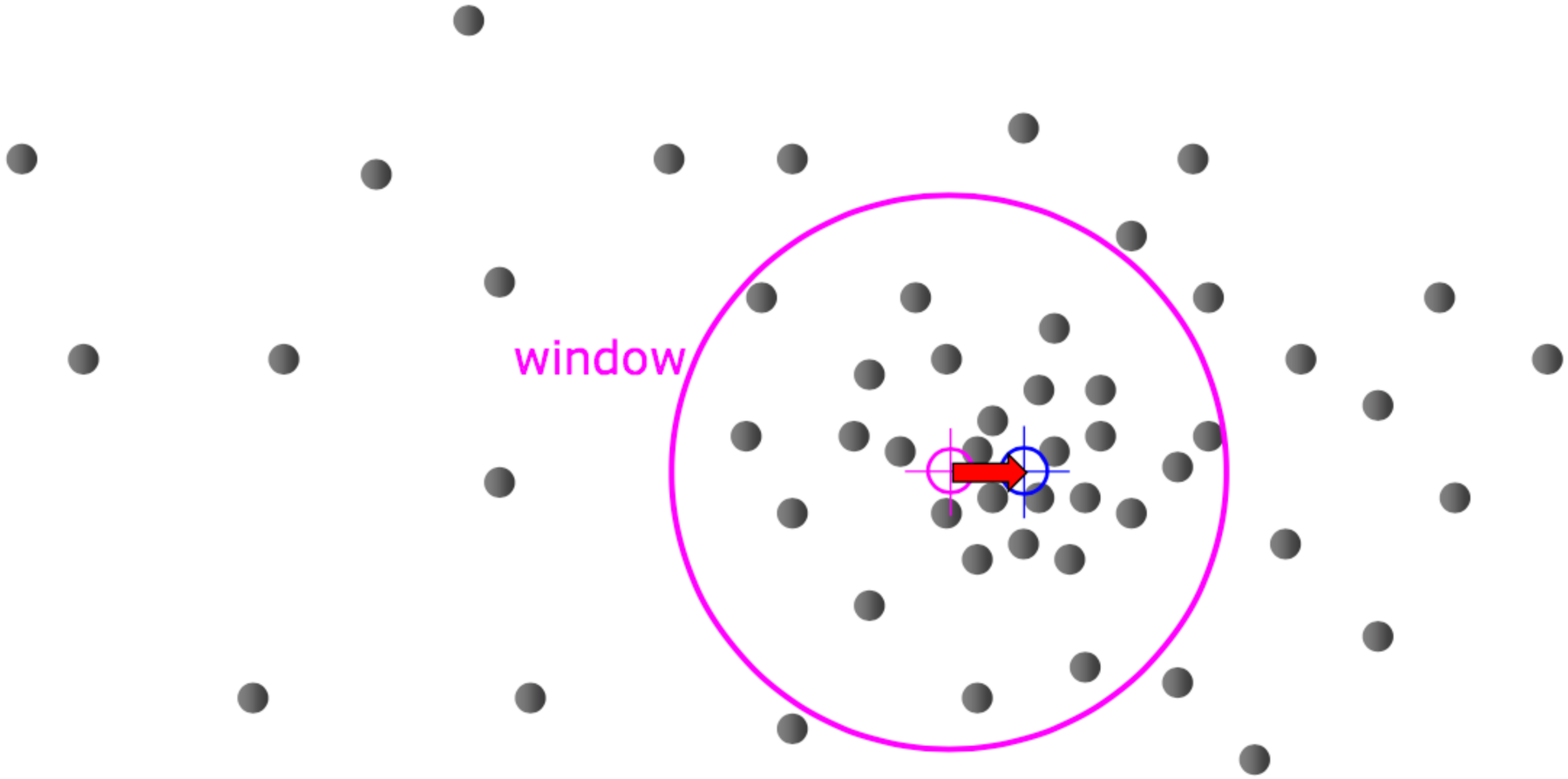
Segmentation as finding places of high density in feature space



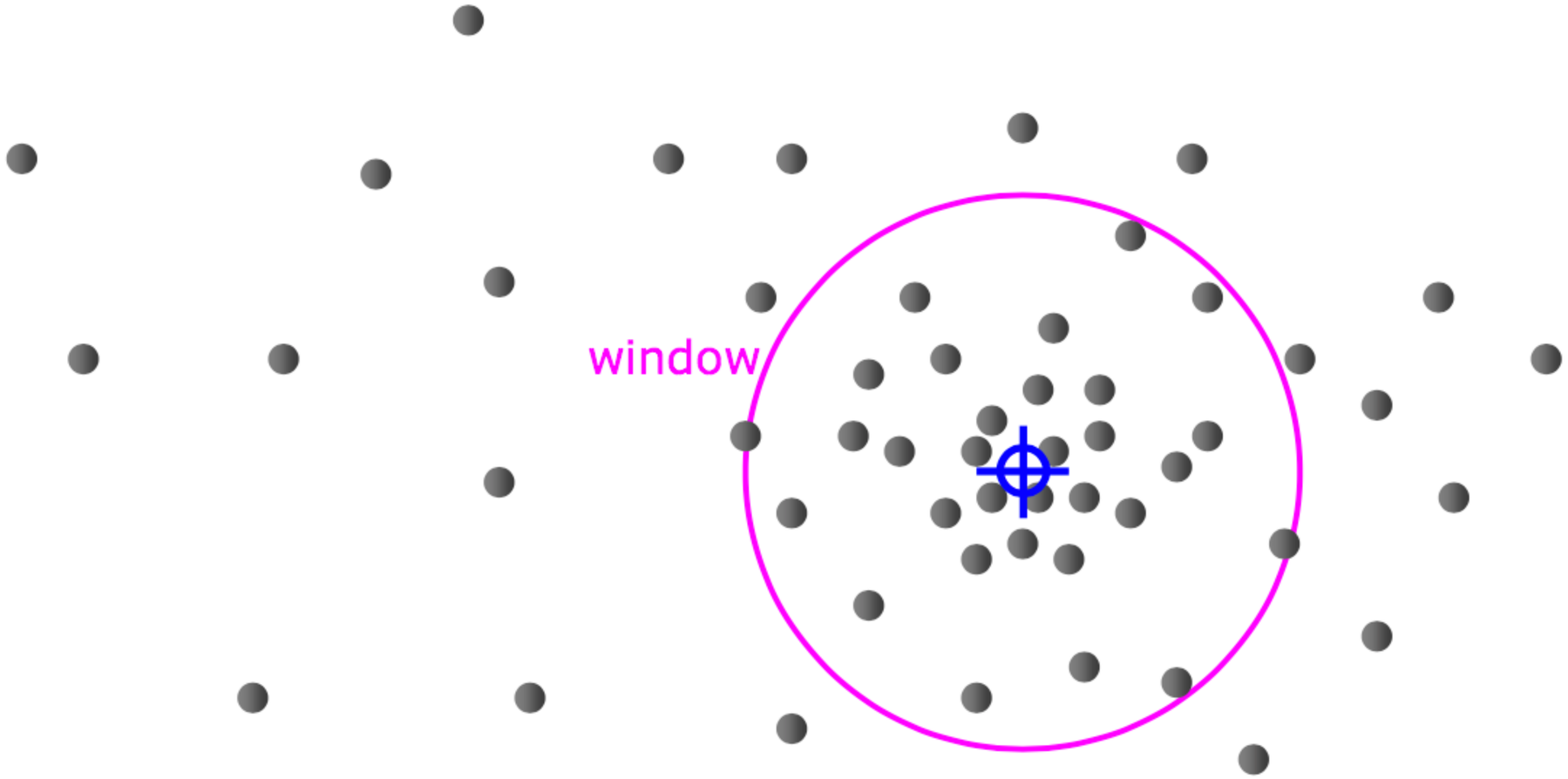
Segmentation as finding places of high density in feature space



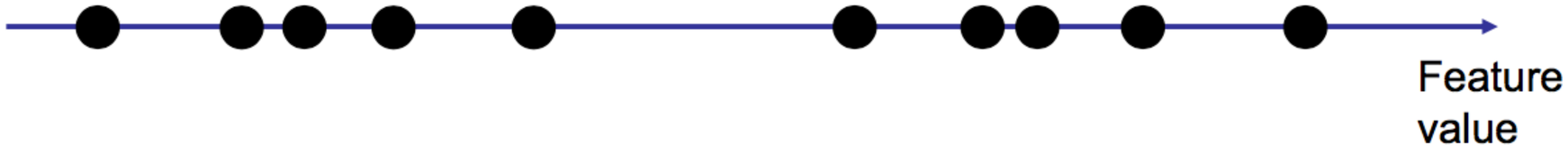
Segmentation as finding places of high density in feature space



Segmentation as finding places of high density in feature space

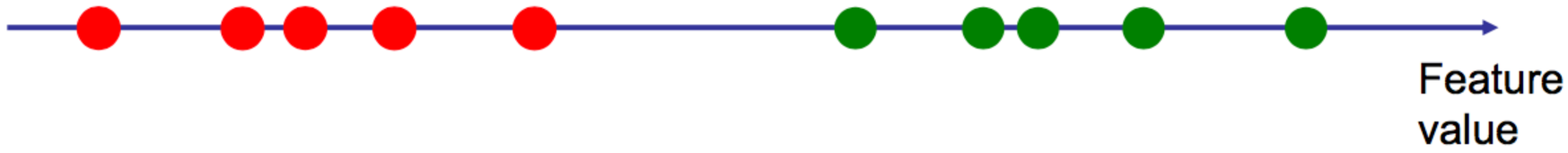


A 1-D Example



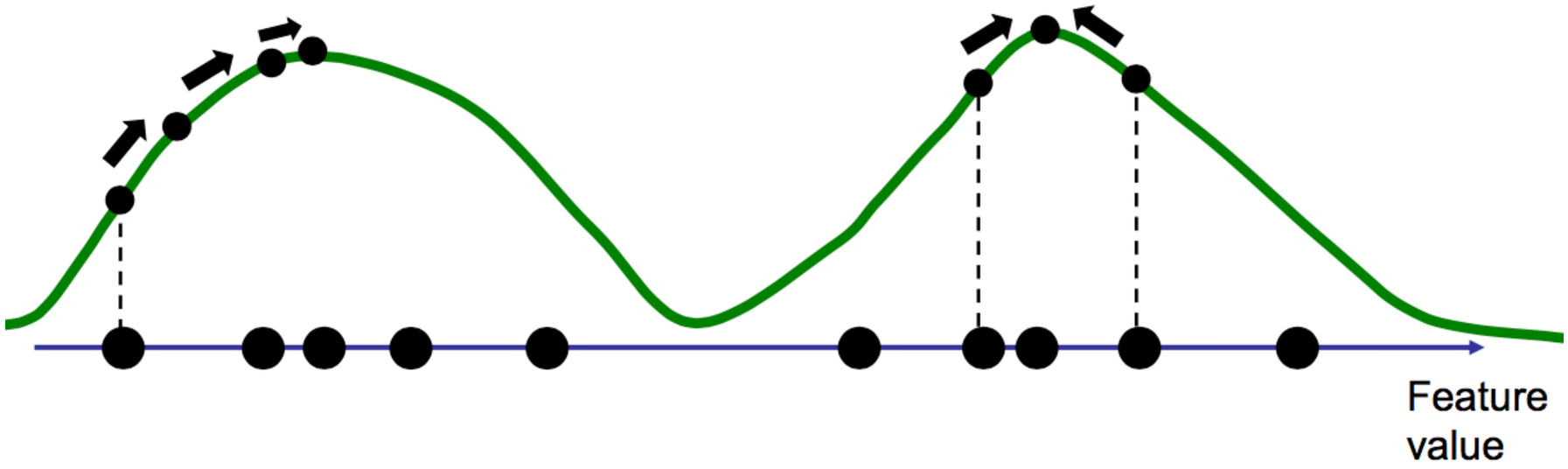
- Consider a set of points in a boring, one-dimensional feature space

A 1-D Example



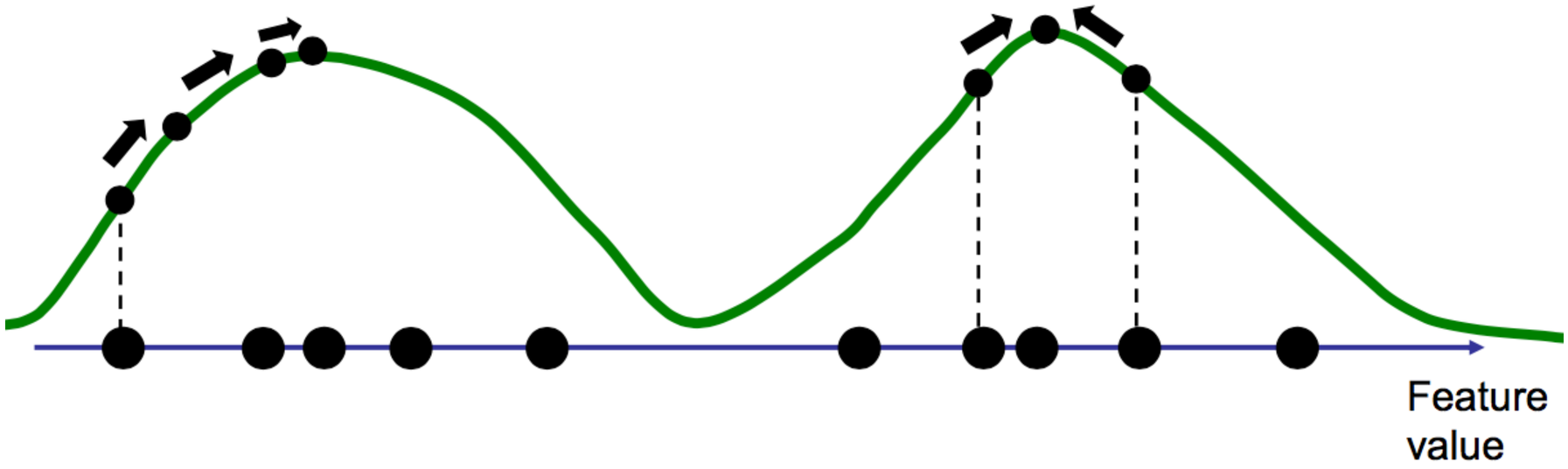
- Obviously, we would like to generate two groups, corresponding to the two parts of the feature space in which we have a *high density* of points
- How can we capture this notion of "high density" → *kernel density estimation*

A 1-D Example



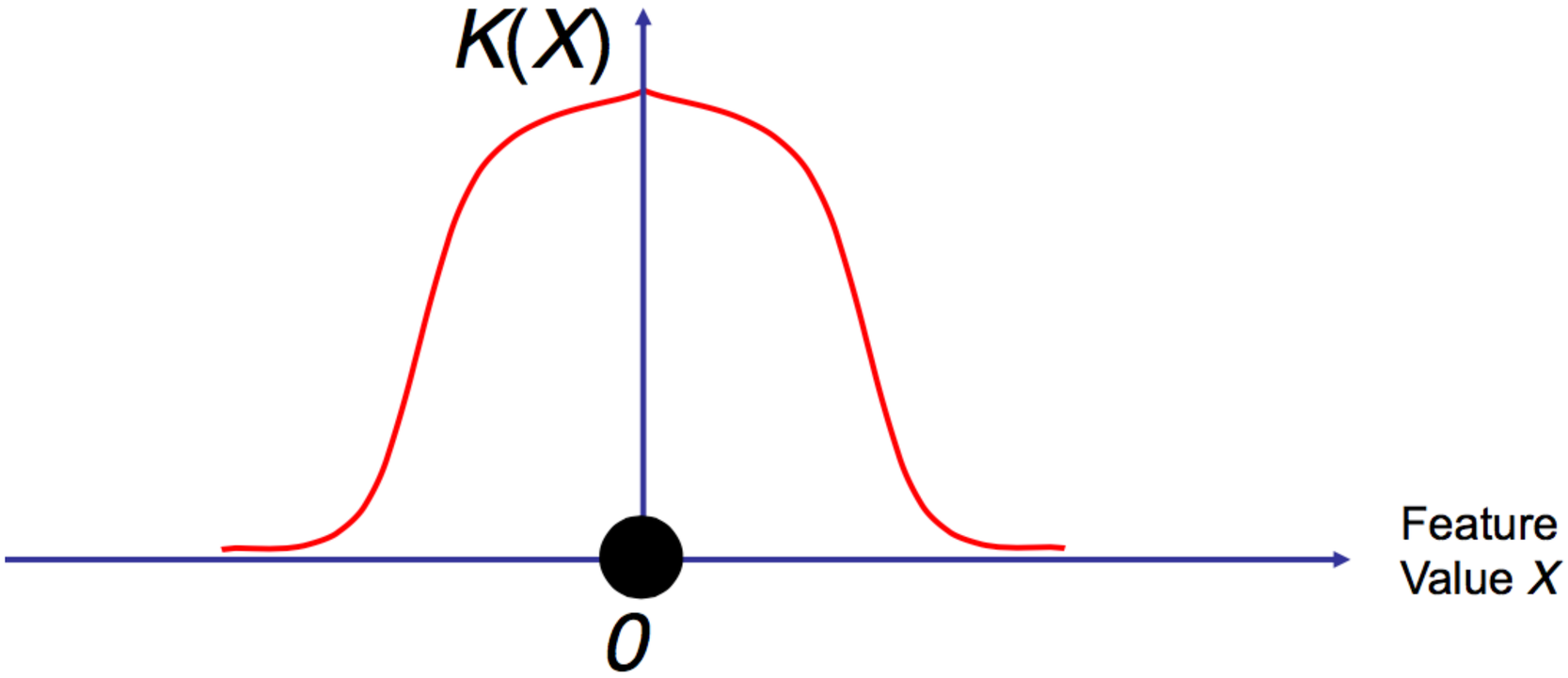
- If we had a continuous function instead of a bunch of data points, we could find the maxima by gradient ascent.
- How can we convert our set of points to a continuous function?

A 1-D Example



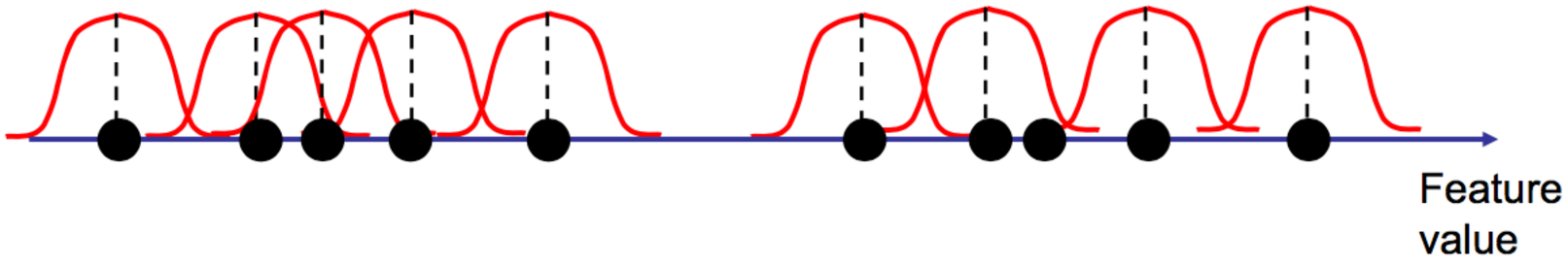
- If we had a continuous function instead of a bunch of data points, we could find the maxima by gradient ascent.
- How can we convert our set of points to a continuous function?

A 1-D Example



- Let us define a kernel function: $K(X)$, with the properties:
- K decays to zero far from 0
- K is maximum at 0
- K is symmetric

A 1-D Example

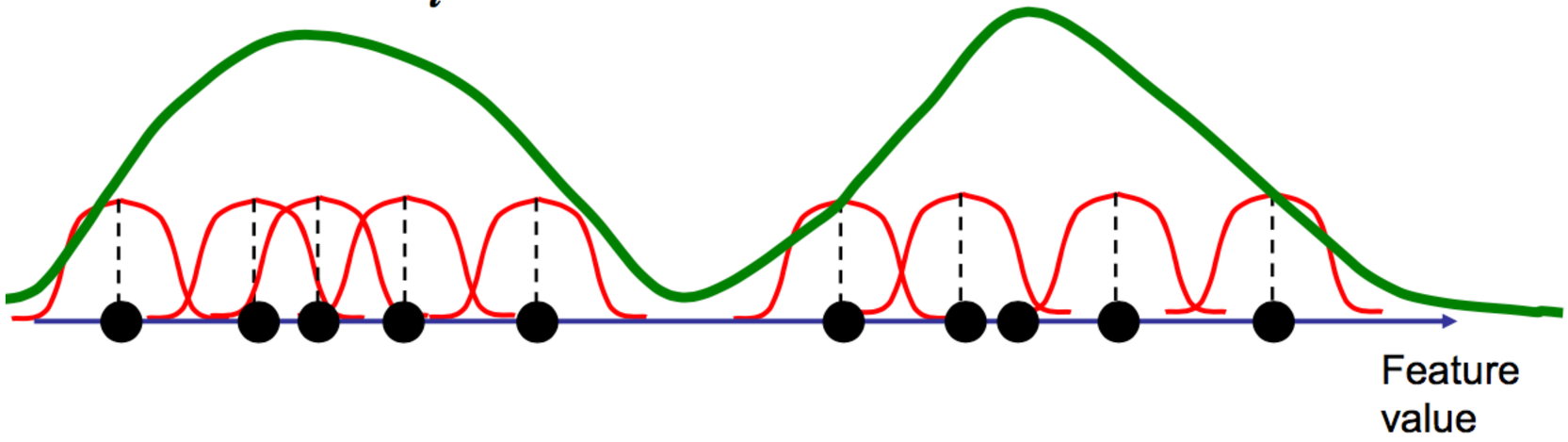


- We can define the kernel at each data point and sum up the result into a single function:

$$f(X) = \frac{1}{N} \sum_i K(X - X_i)$$

A 1-D Example

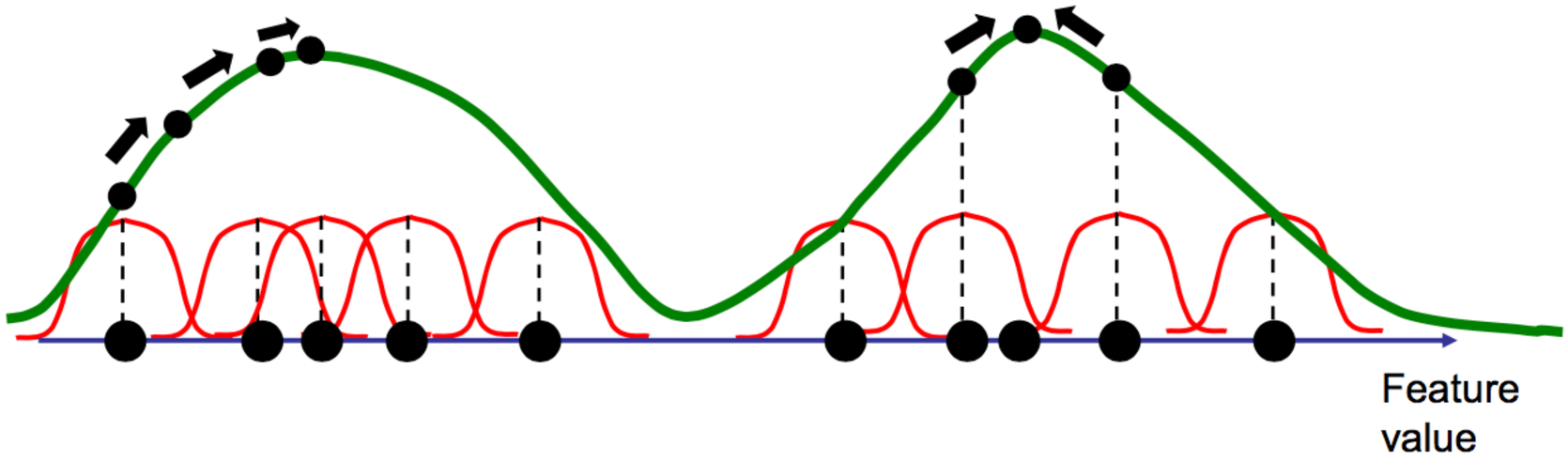
$$f(X) = \frac{1}{N}\sum_i K(X - X_i)$$



- *V* is a normalization term
- *f(X) approximates the probability that feature X is observed given the data points*
- The maxima of *f* (the modes of the pdf) correspond to the clusters in the data

A 1-D Example

$$f(X) = \frac{1}{N}\sum_i K(X - X_i)$$



- If we move each point in the direction of the gradient, we will converge to the closest mode
- How can we do this efficiently?

A 1-D Example

- For $i = 1,..,N$

  $$\boldsymbol{X} \leftarrow \boldsymbol{X}_i$$

  – Repeat

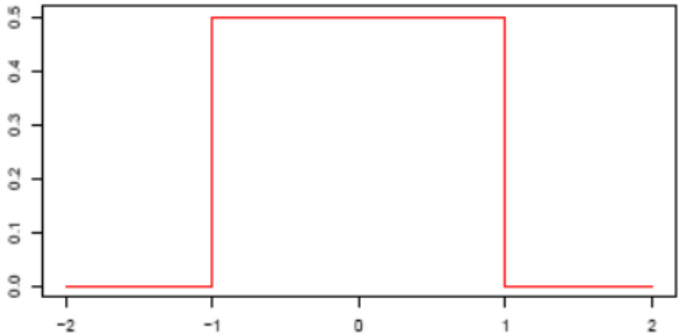
  $$\boldsymbol{X} \leftarrow \boldsymbol{X}_i + \nabla \boldsymbol{f}(\boldsymbol{X}) = \boldsymbol{X}_i + \frac{1}{\boldsymbol{N}} \sum_{\boldsymbol{i}} \nabla \boldsymbol{K}(\boldsymbol{X} - \boldsymbol{X}_i)$$

  – Until X does not change

Example kernels

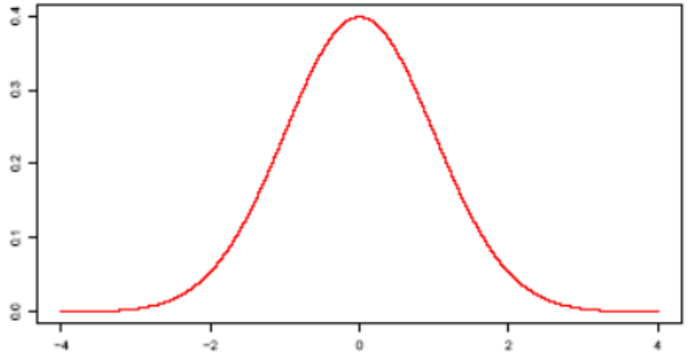
Uniform:

$$K_U(\mathbf{x}) = \begin{cases} c & \|\mathbf{x}\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$



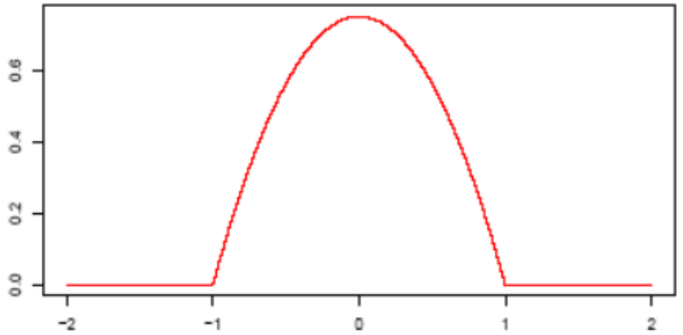
Gaussian:

$$K_N(\mathbf{x}) = c \cdot \exp\left(-\frac{1}{2} \|\mathbf{x}\|^2 \right)$$



Epanechnikov:

$$K_E(\mathbf{x}) = \begin{cases} c\left(1 - \|\mathbf{x}\|^2\right) & \|\mathbf{x}\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$



Bandwidth

- Kernel is defined as:

$$K(X) = ck\left(\left\|\frac{X}{h}\right\|^2\right)$$

- *h* is the bandwith of the kernel
- *k*(.) is:
  - For Gaussian: $k(t) = e^{-t/2}$
  - For Epanechnikov: $k(t) = \begin{cases} (1-t) \text{ if } |t| < 1 \\ 0 \end{cases}$

Bandwidth

- Kernel is defined as:

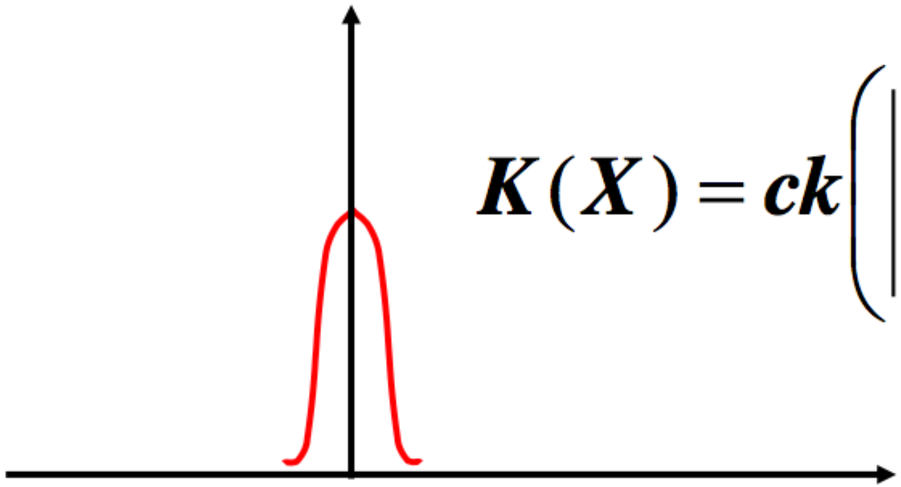
$$K(X) = ck\left(\left\|\frac{X}{h}\right\|^2\right)$$

- *h* is the bandwith of the ke
- *k*( ) is:

Bandwidth *h* controls the radius of influence of each data point.

- Too small: Overfits the data points
- Too large: Smoothes out the details of the data

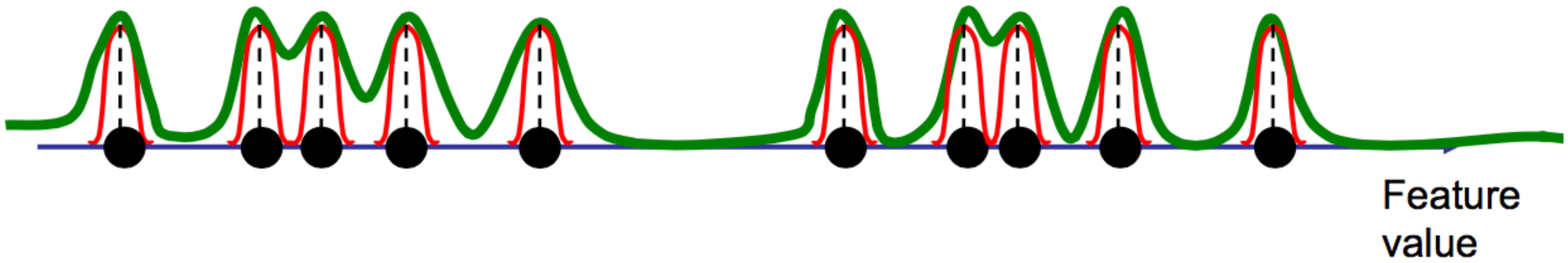
$$k(t) = \begin{cases} \\ 0 \end{cases}$$

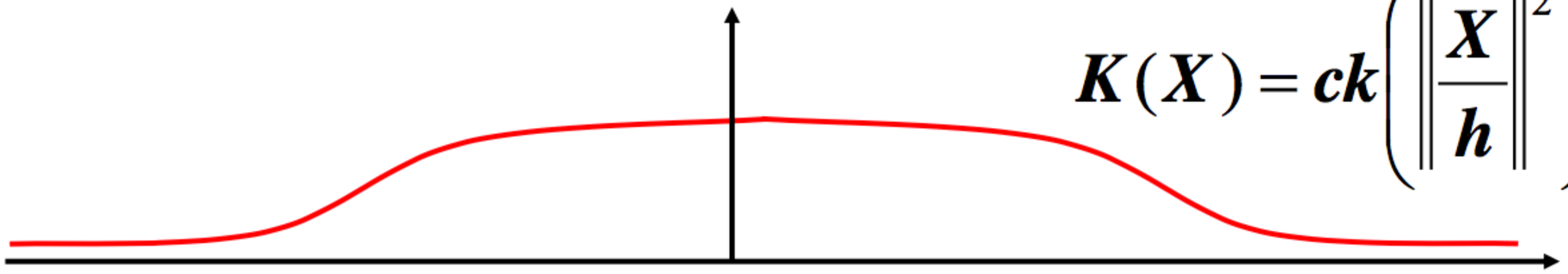


$$K(X) = ck\left(\left\|\frac{X}{h}\right\|^2\right)$$

h too small: The pdf overfits the noise in the data → Too many modes

$$f(X) = \sum_i ck\left(\left\|\frac{X - X_i}{h}\right\|^2\right)$$



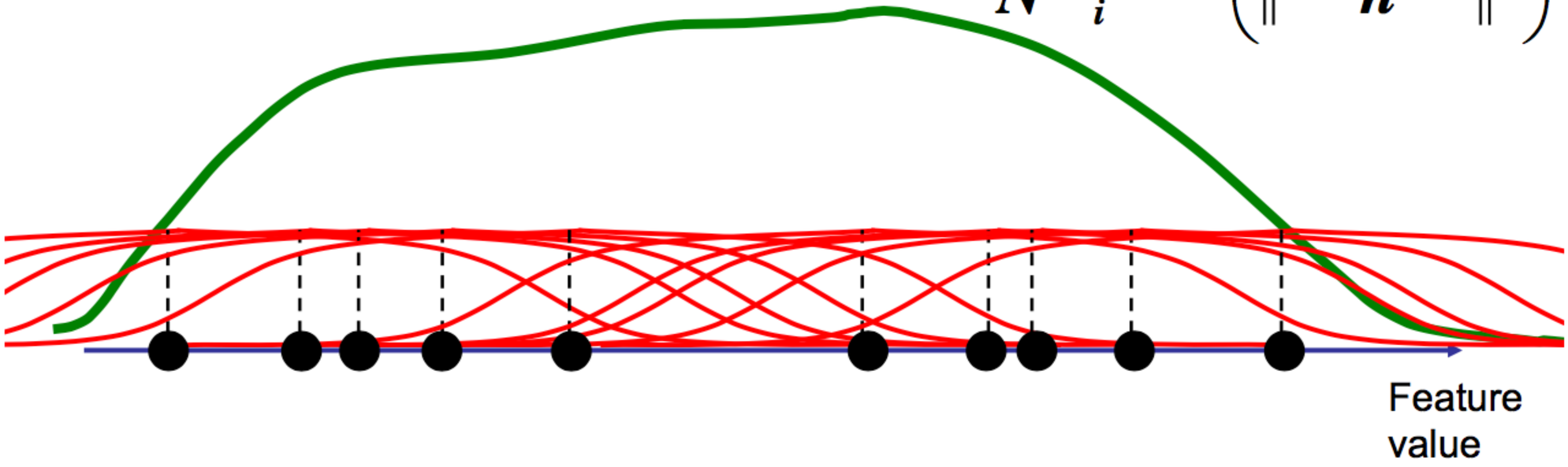


$$K(X) = ck\left(\left\|\frac{X}{h}\right\|^2\right)$$

h too large: The details of the initial data are smoothed out

→ Too few modes

$$f(X) = \frac{1}{N}\sum_i ck\left(\left\|\frac{X - X_i}{h}\right\|^2\right)$$



Choice of Kernel

- The kernel must satisfy a few technical conditions (*aka Parzen windows*).
- Integrates to 1 so that *f*(.) is a pdf: $\int_{R^d} K(\mathbf{x})d\mathbf{x} = 1$
- Symmetric
- Decays quickly (exponentially) as |*x*| increases: $\lim_{\|\mathbf{x}\|\to\infty} \|\mathbf{x}\|^d K(\mathbf{x}) = 0$
- The extent of the kernel is the same along all the dimensions: $\int_{R^d} \mathbf{x}\mathbf{x}^T K(\mathbf{x})d\mathbf{x} = c\mathbf{I}$

Computing the Gradient

- Now we have a representation of the pdf from which, in principle, we can find the modes by following the gradient.
- How can we do this efficiently?
- Notations:

$$g(t) = -k'(t)$$

- Gradient of each individual entry in the sum defining $f(.)$:

$$\nabla K(X - X_i) = \nabla\left(ck\left(\frac{\|X - X_i\|^2}{h^2}\right)\right) = \frac{2c}{h^2}(X_i - X)g\left(\frac{\|X - X_i\|^2}{h^2}\right)$$

Computing the Gradient

- Gradient of the entire pdf:

$$\nabla f(X) = \frac{1}{N}\sum_i \nabla K(X - X_i) = \frac{2c}{Nh^2}\sum_i (X_i - X) g\left(\frac{\|X - X_i\|^2}{h^2}\right)$$

$$\Downarrow$$

$$\nabla f(X) = \left(\frac{2c}{Nh^2}\sum_i g\left(\frac{\|X - X_i\|^2}{h^2}\right)\right)\left(\frac{\sum_i X_i g\left(\frac{\|X - X_i\|^2}{h^2}\right)}{\sum_i g\left(\frac{\|X - X_i\|^2}{h^2}\right)} - X\right)$$

$$\nabla f(X) = \left(\frac{2c}{Nh^2} \sum_i g\left(\frac{\|X - X_i\|^2}{h^2} \right) \right) \underbrace{\left(\frac{\sum_i X_i g\left(\frac{\|X - X_i\|^2}{h^2} \right)}{\sum_i g\left(\frac{\|X - X_i\|^2}{h^2} \right)} - X \right)}$$

Mean shift vector, $M(X)$ = Difference between X and the mean of the data points weighted by $g(.)$ (points further from X count less)

- Key result: The mean shift vector points in the same direction as the gradient
- Solution: Iteratively move in the direction of the mean shift vector

The Mean Shift Algorithm

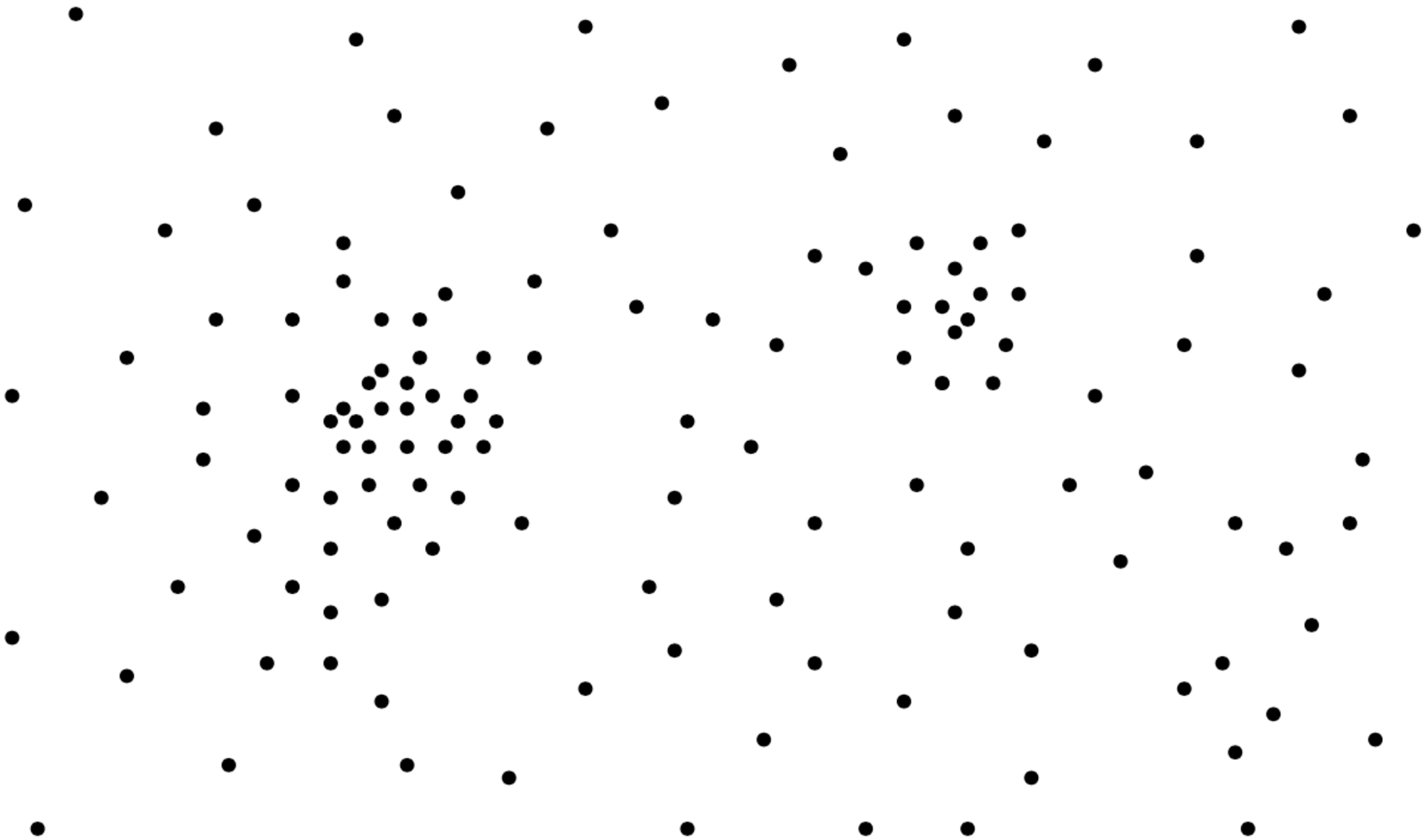
- Initialize: Set X to the value of the point to classify
- Repeat:
  - Move X by the corresponding mean shift vector:

$$X \leftarrow X + M(X) = \frac{\sum_i X_i g\left(\frac{\|X - X_i\|^2}{h^2}\right)}{\sum_i g\left(\frac{\|X - X_i\|^2}{h^2}\right)}$$

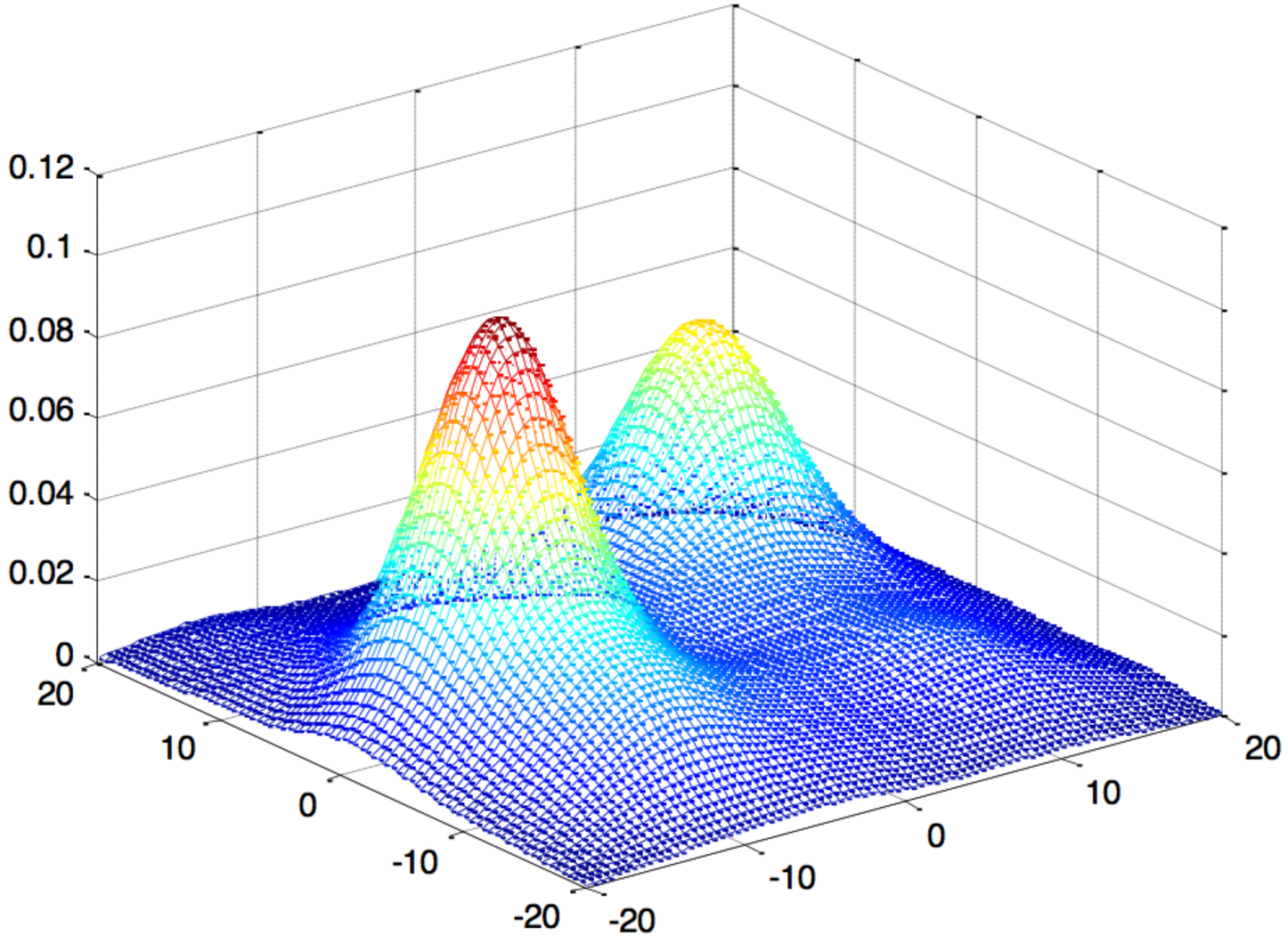
- Until X converges
- Note: Convergence is guaranteed.

2-D Example

$$f(X) = \frac{c}{N}\sum_{i=1}^{N} k\left(\left\|\frac{X - X_i}{h}\right\|^2\right)$$

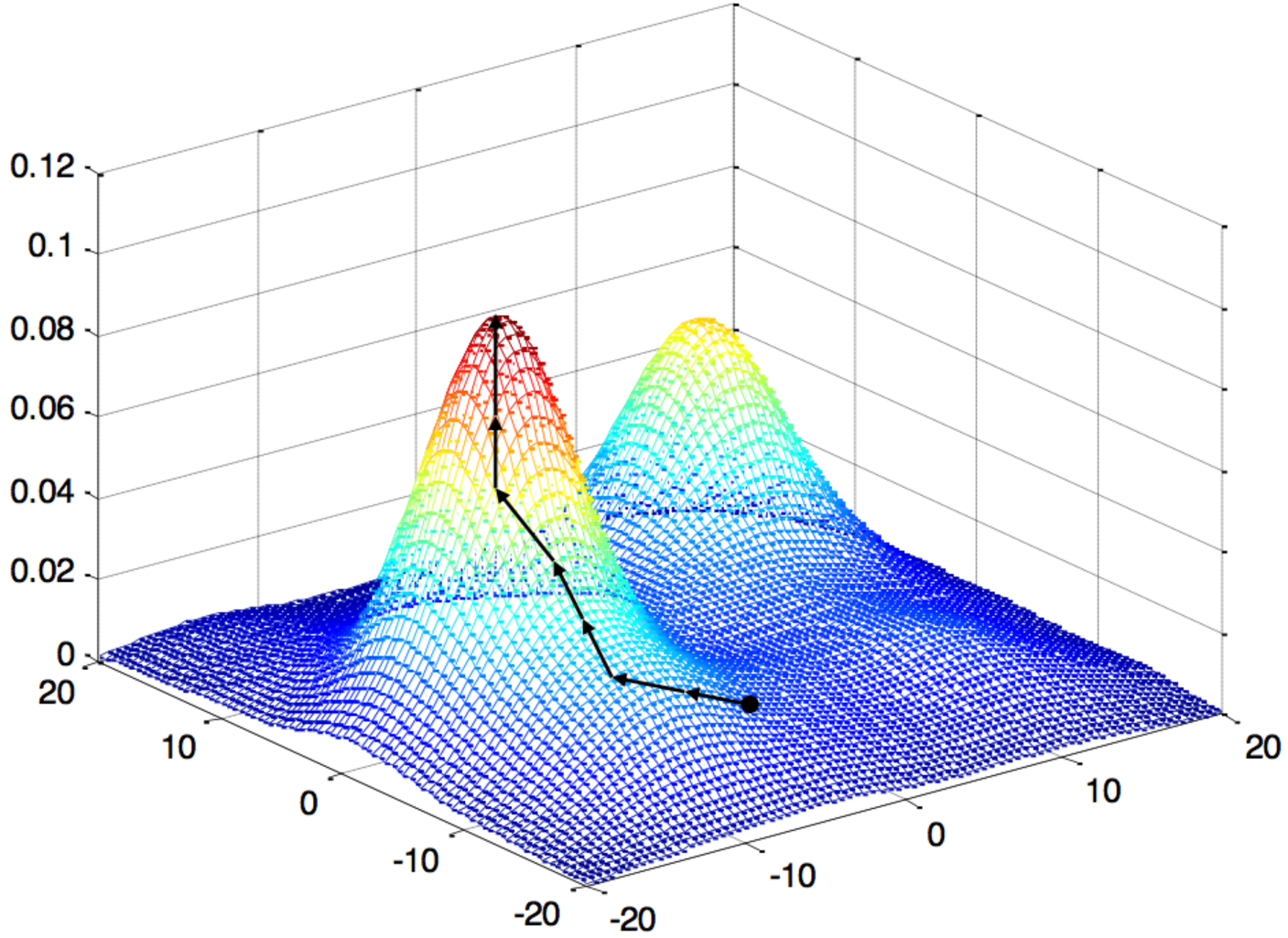


Estimated PDF: $$f(X) = \frac{c}{N}\sum_{i=1}^{N} k\left(\left\|\frac{X - X_i}{h}\right\|^2\right)$$



The trajectory of locations for finding modes

$$f(X) = \frac{c}{N}\sum_{i=1}^{N} k\left(\left\|\frac{X - X_i}{h}\right\|^2\right)$$



The Reality

- This is all much simpler than it looks!!
- For Epanechnikov:
  - $k(t) = (1-t)$ if $|t| < 1$, 0 otherwise
  - $g(t) = 1$ if $|t| < 1$, 0 otherwise
- So, the "mean" part of $M(X)$ is:

$$\boxed{\frac{\sum_i X_i g\left(\frac{\|X - X_i\|^2}{h^2}\right)}{\sum_i g\left(\frac{\|X - X_i\|^2}{h^2}\right)} = \frac{\sum_{\|X - X_i\| < h} X_i}{\sum_{\|X - X_i\| < h} 1} = \frac{\sum_{\|X - X_i\| < h} X_i}{N_h(X)}}$$

The Reality

This is *simply* the *average* of the data points within a radius *h* of *X*!!!

$$\frac{\sum_i X_i g\left(\frac{\|X - X_i\|^2}{h^2}\right)}{\sum_i g\left(\frac{\|X - X_i\|^2}{h^2}\right)} = \boxed{\frac{\sum_{\|X - X_i\| < h} X_i}{N_h(X)}}$$

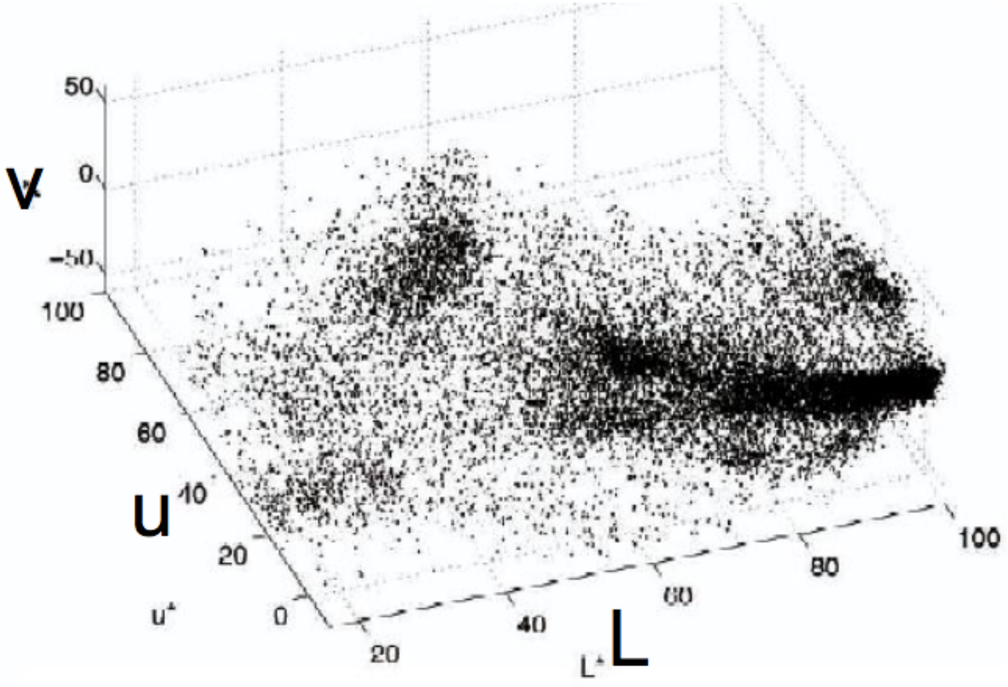
Number of data points within a radius *h* of *X*

Example : Color Segmentation

- Feature space: (L,u,v,x,y) → Intensity + (u,v) color channels + Position in image (x,y)
- Apply meanshift in the 5-dimensional space
- For each pixel (x_i,y_i) of intensity L_i and color (u_i,v_i), find the corresponding mode c_k
- All of the pixel (x_i,y_i) corresponding to the same mode c_k are grouped into a single region

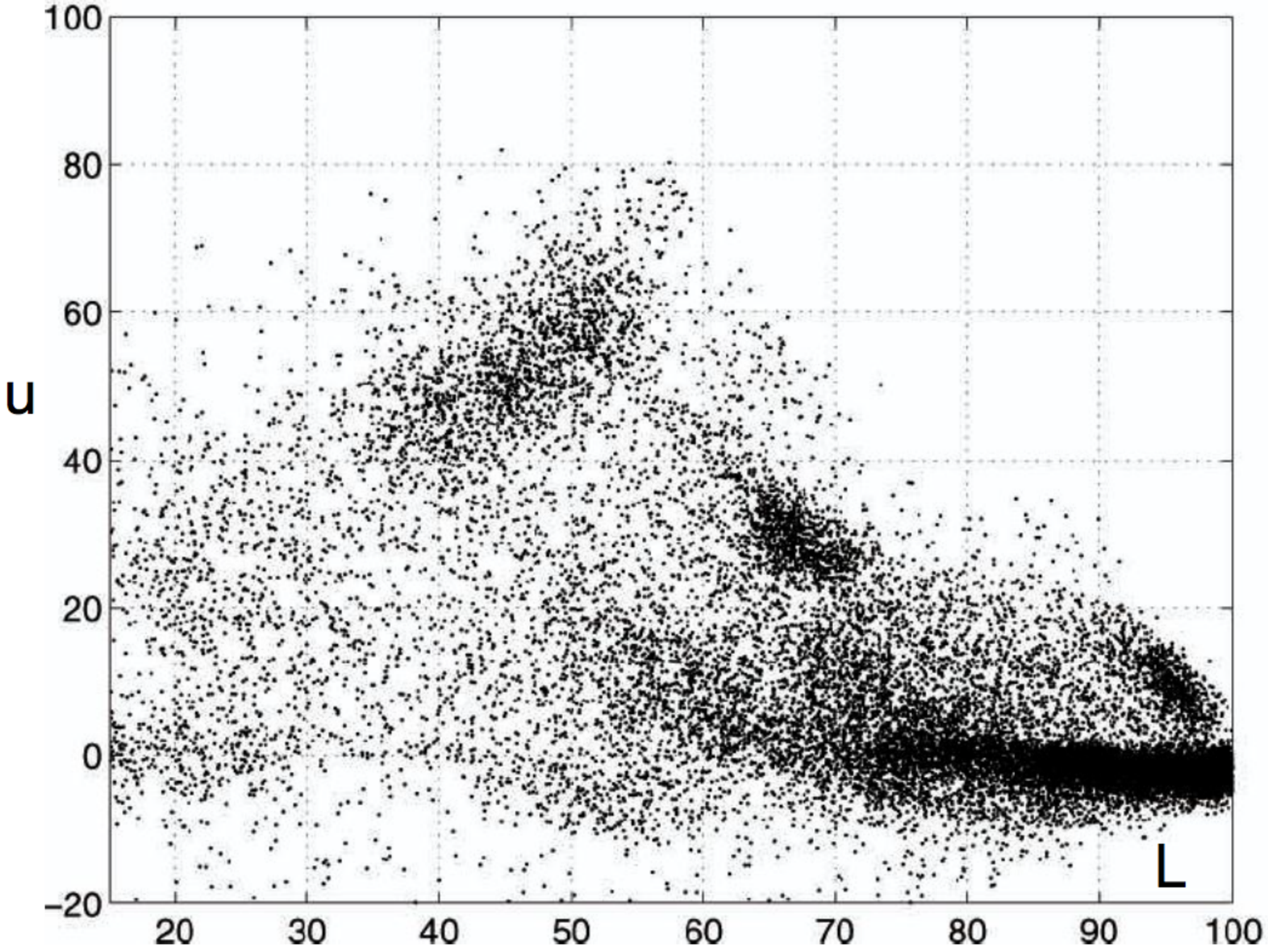
Example : Color Segmentation

Input Image



Luv Space ()

Example from D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach Toward Feature Space Analysis".



110,400 data points.

$$K_{h_{pos}h_{col}}(X) = ck\left(\frac{\|X_{pos}\|^2}{h_{pos}^{\ 2}}\right)k\left(\frac{\|X_{col}\|^2}{h_{col}^{\ 2}}\right)$$

Kernel on position (*x*,*y*)

Kernel on color (*L*,*u*,*v*)

- *Good news:* We don't need to know the number of regions (modes, clusters).
- *Bad news:* We need to choose the bandwidths h_{pos} and h_{col}

The Mean Shift Process

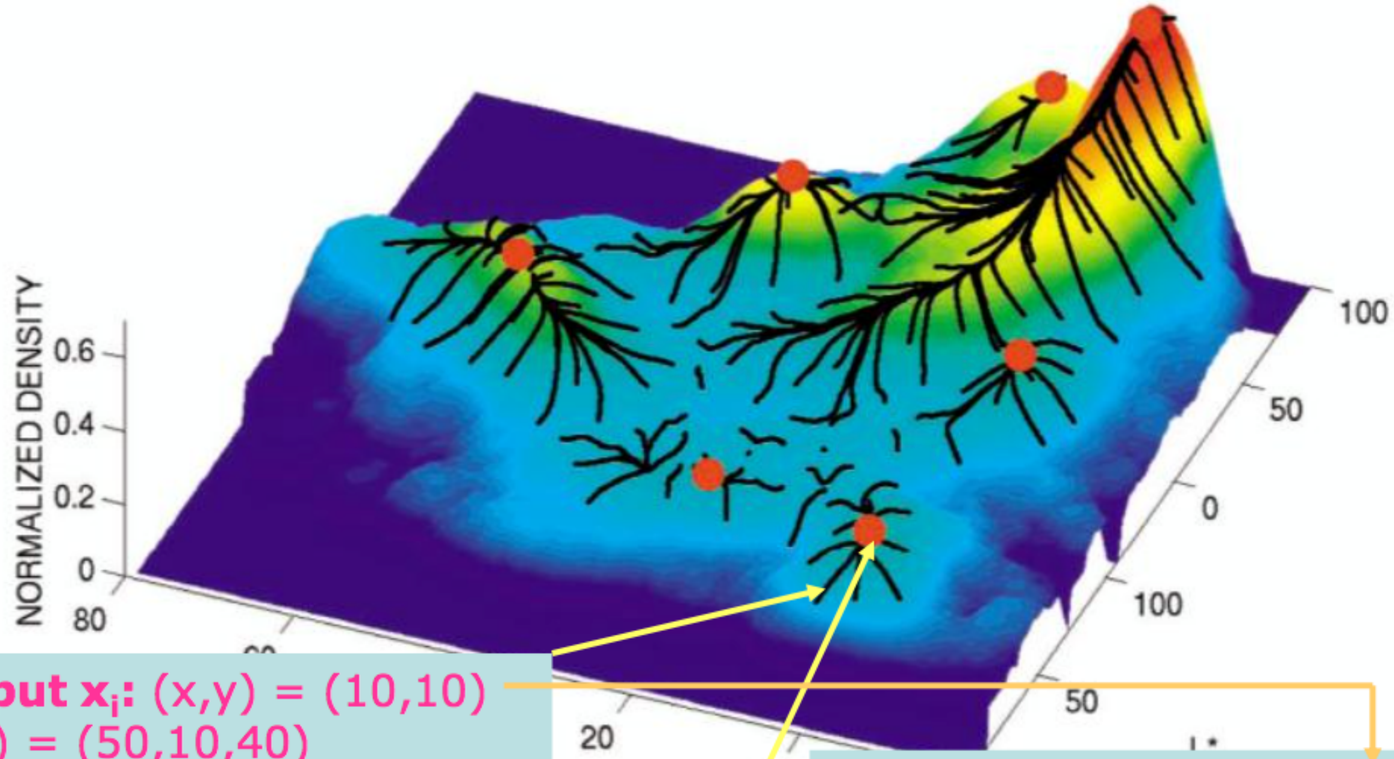
Notes:

- If we do not apply the last step, we get "smoothing"
→ Replacing each color by the closest mode

- The "color" part of the feature can be replaced by other things like texture (bank of filter outputs) or other values (multispectral). The only change is to increase the dimension *p* of the feature space

- The fundamental operation to compute the kernels is to find the neighbors within some radius (defined by *h*). This can be very expensive in high dimension with lots of points → Need smart "nearest-neighbor" data structures.

Example Color



1) Input x_i: (x,y) = (10,10)
(L,u,v) = (50,10,40)

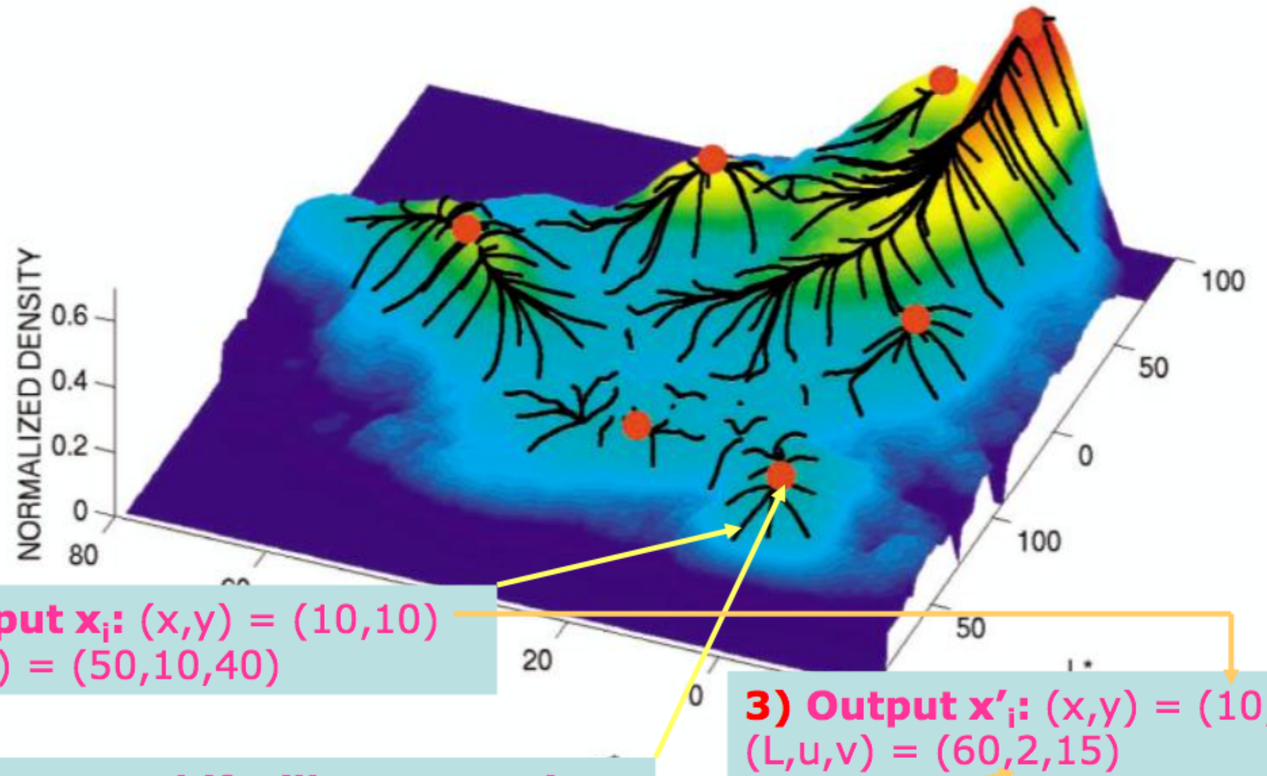
2) Apply mean shift till converged

c_i: (x,y) = (15,20) (L,u,v) = (60,2,15)

3) Output x'_i: (x,y) = (10,10)
(L,u,v) = (60,2,15)

D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach Toward Feature Space Analysis".

Example Color



1) Input x_i: (x,y) = (10,10)
(L,u,v) = (50,10,40)

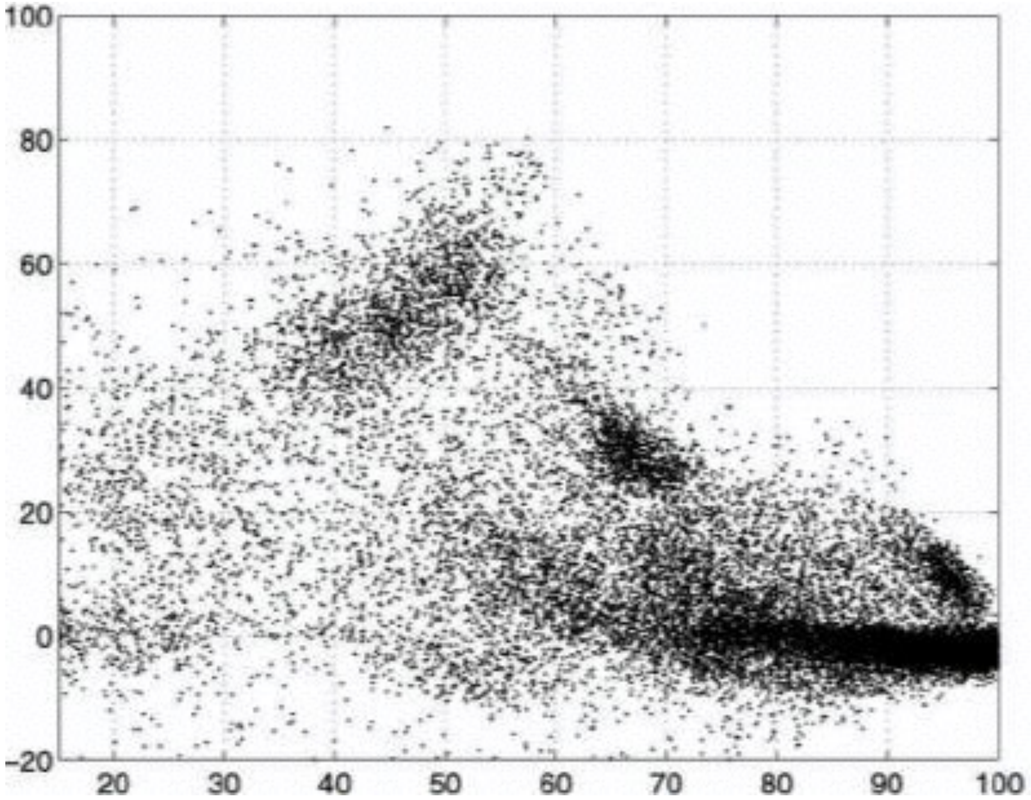
2) Apply mean shift till converged

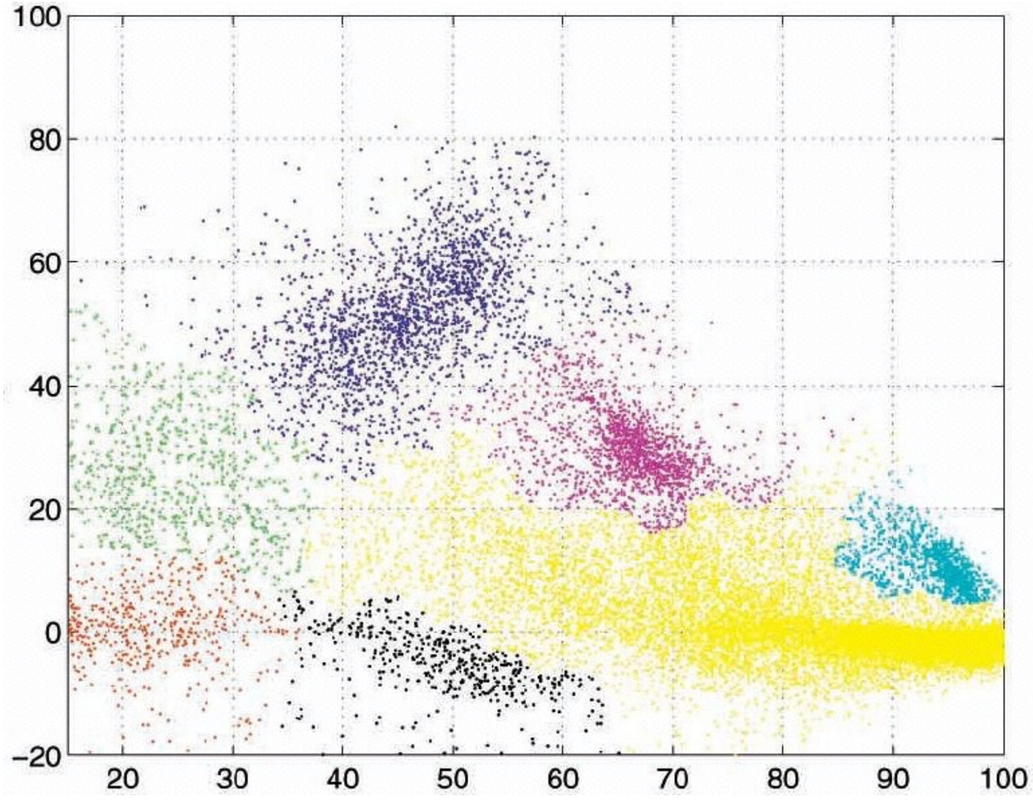
c_i: (x,y) = (15,20) (L,u,v) = (60,2,15)

3) Output x'_i: (x,y) = (10,10)
(L,u,v) = (60,2,15)

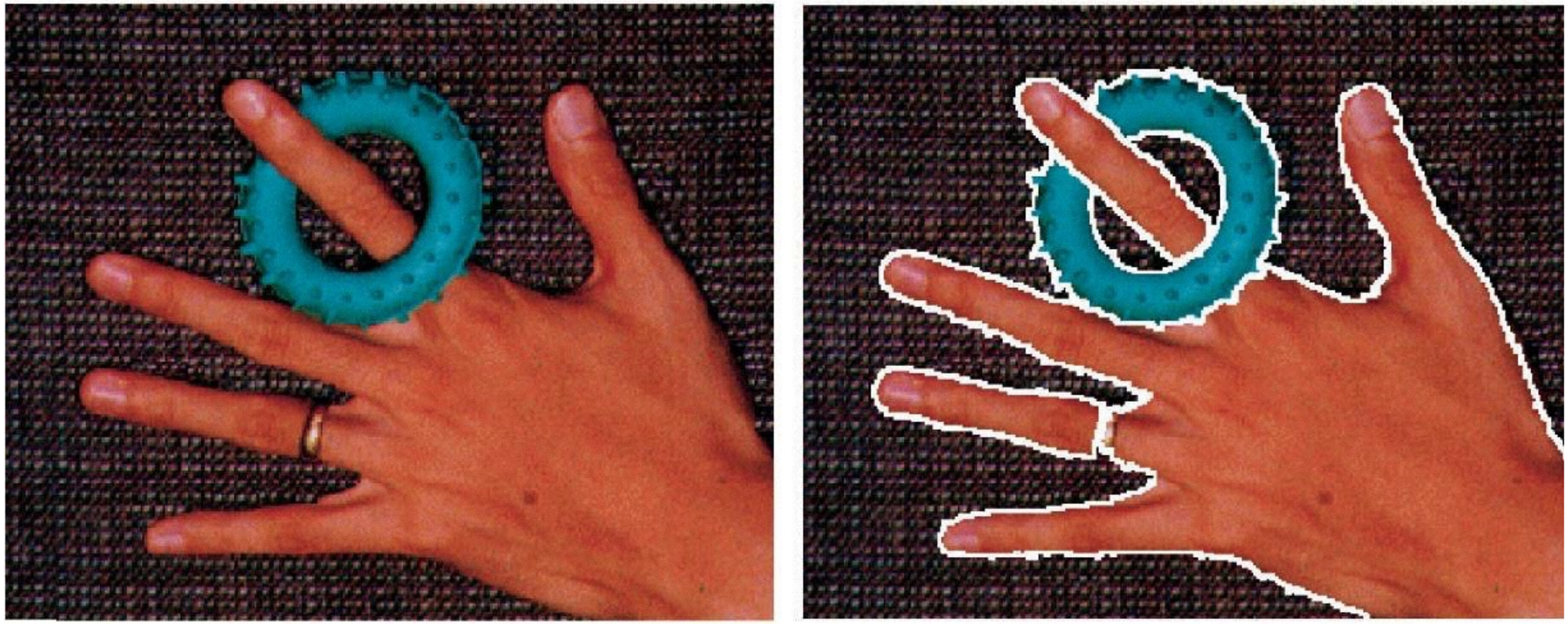
Note: In practice, all points may not converge to the same mode -→ Need an additional (easy) clustering step to group the converged locations to the location

Example Color

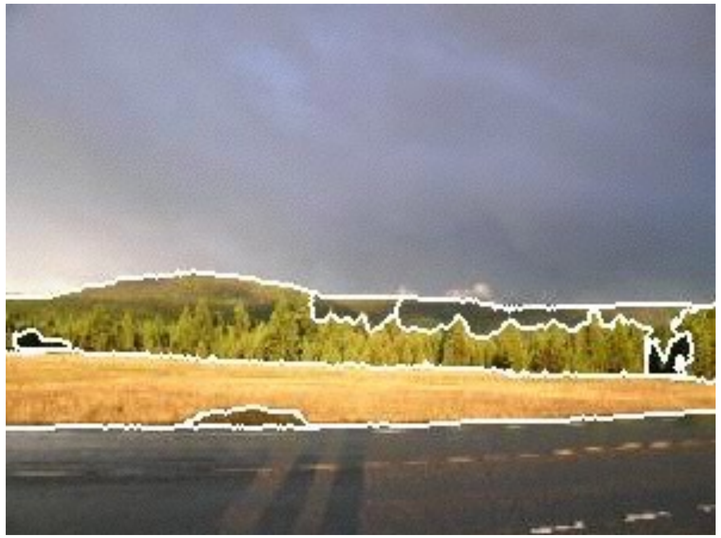


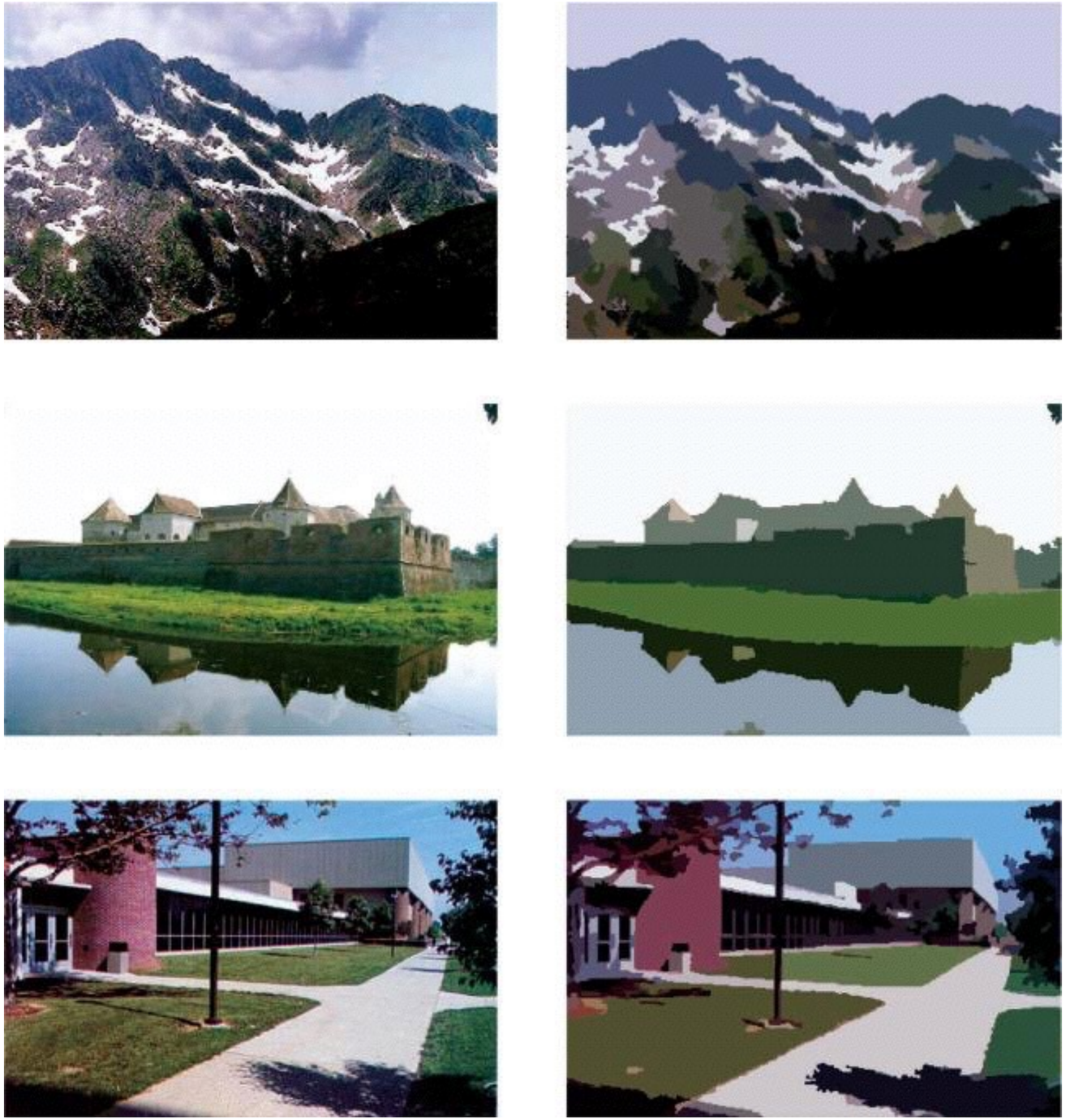


Experimental Results



Experimental Results





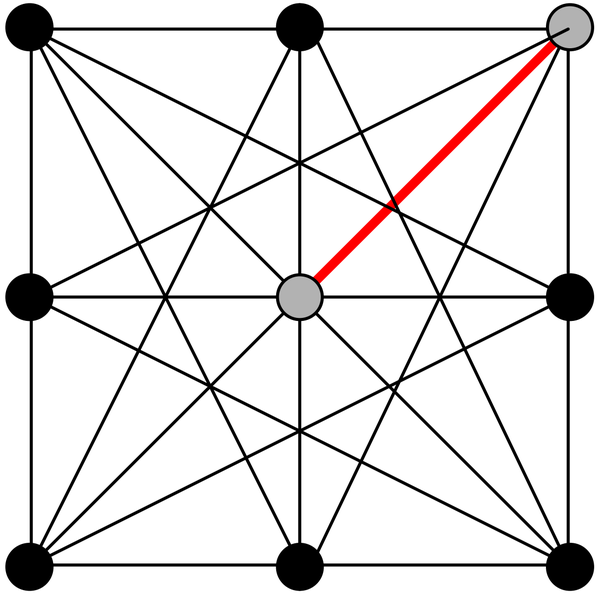
Main Approaches

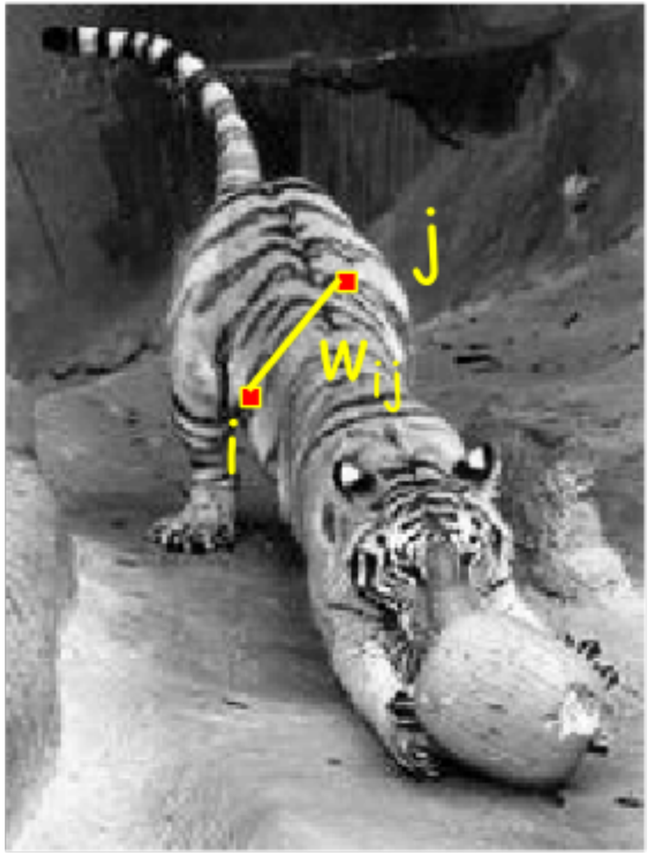
- Clustering
- Mean shift
- Spectral techniques (skipped)
- Graph-based techniques
- Segmentation as boundary detection

Main Approaches

- Clustering
- Mean shift
- Spectral techniques
- Graph-based techniques (summary)
- Segmentation as boundary detection

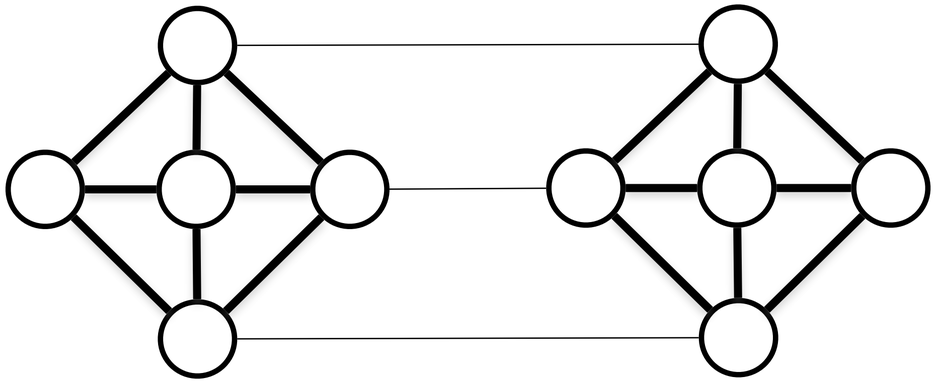
Images as Graphs





- Node for every pixel
- Edges between “neighbors”
- Edge weights/capacities are some measure of similarity

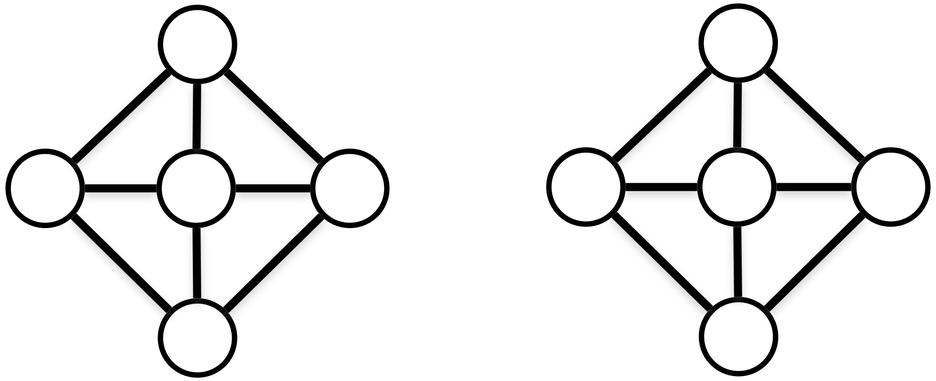
Connected Components



- Define similarity w_{ij} between nodes i,j
  - If similar, $w_{ij} \rightarrow 0$

Connected Components

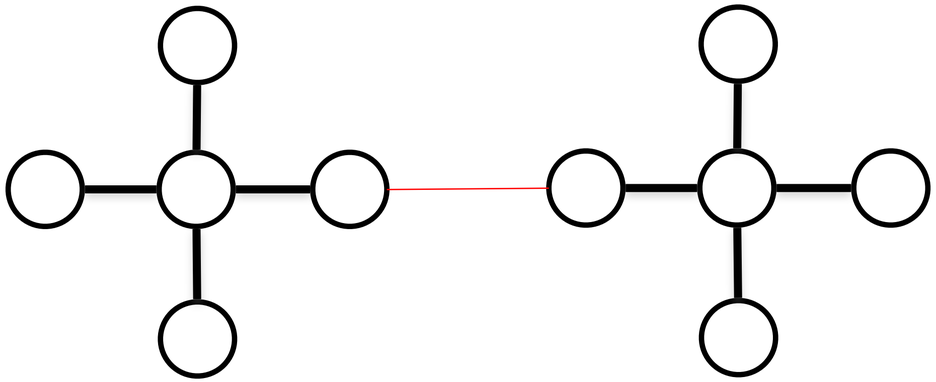
- Define similarity w_{ij} between nodes i,j
  - If similar, $w_{ij} \rightarrow 0$



- Remove edges with $w_{ij} \geq \tau$

Connected Components

- Define similarity w_{ij} between nodes i,j
  - If similar, $w_{ij} \rightarrow 0$

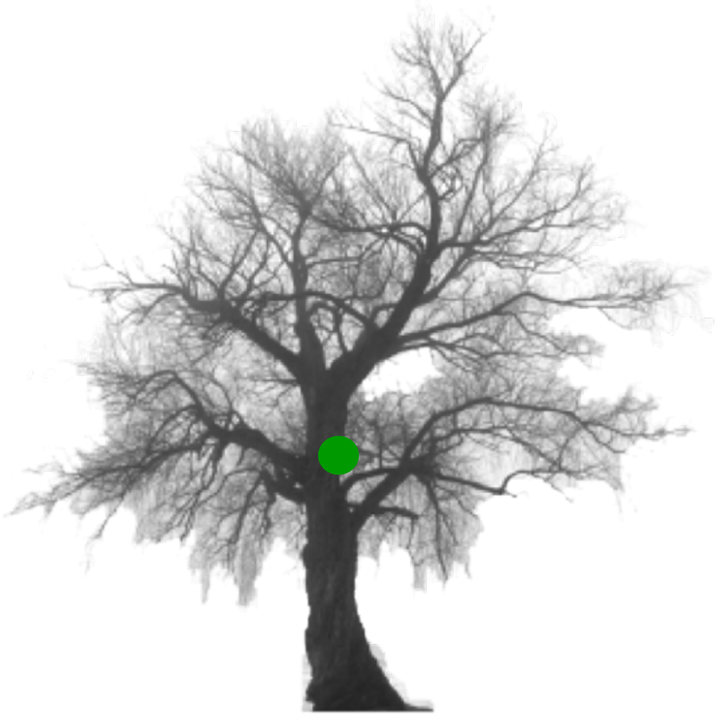


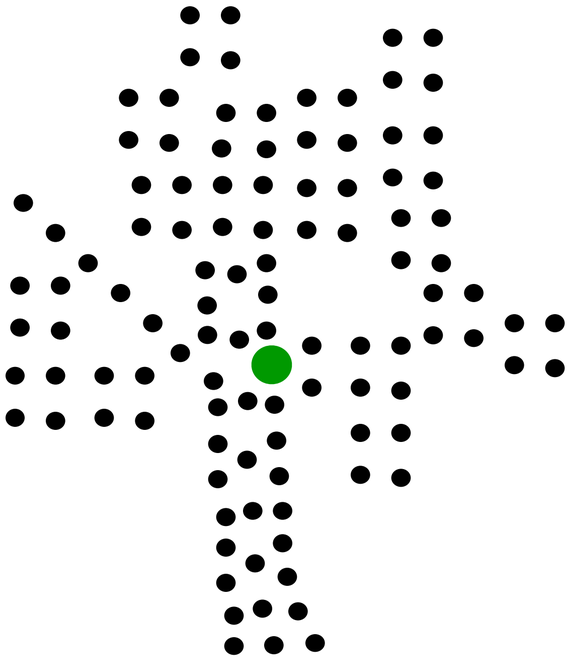
- Remove edges with $w_{ij} \geq \tau$
- Easily done by pruning the minimum spanning tree

Minimum Spanning Forests

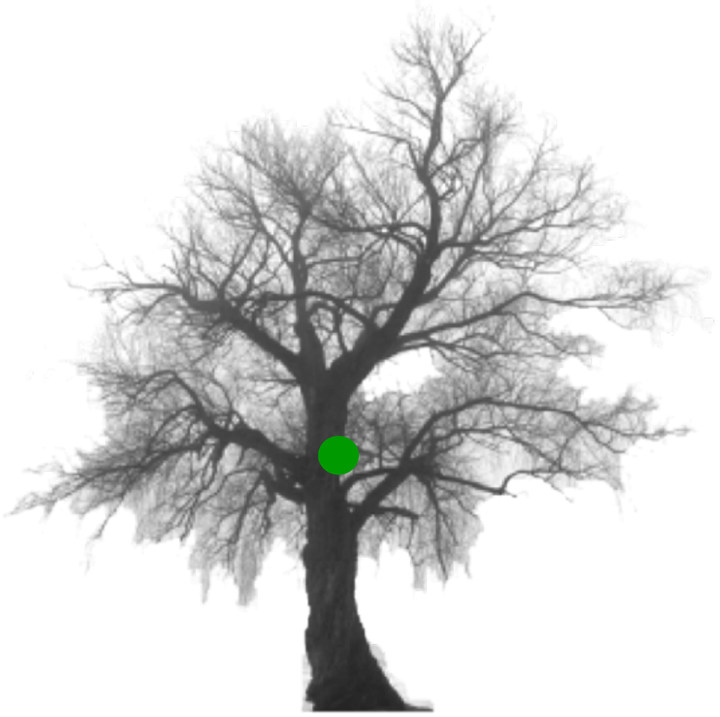
- Initialize each node as a component
- Iterate over w_{ij} in increasing order
  - If $w_{ij} \geq \tau$, STOP
  - If component(i) != component(j)
    - Merge(component(i), component(j))
- Problem: τ very sensitive
  - 1 incorrect merge can lead to creating a huge segment

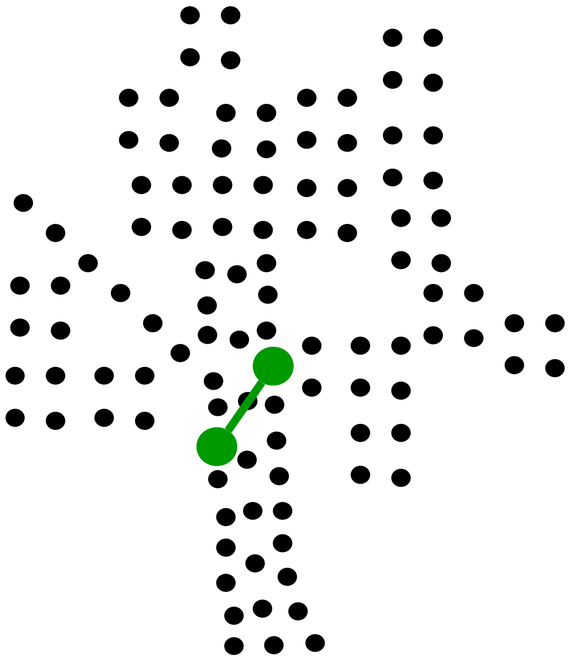
Graphical Approach : Higher Order Potentials



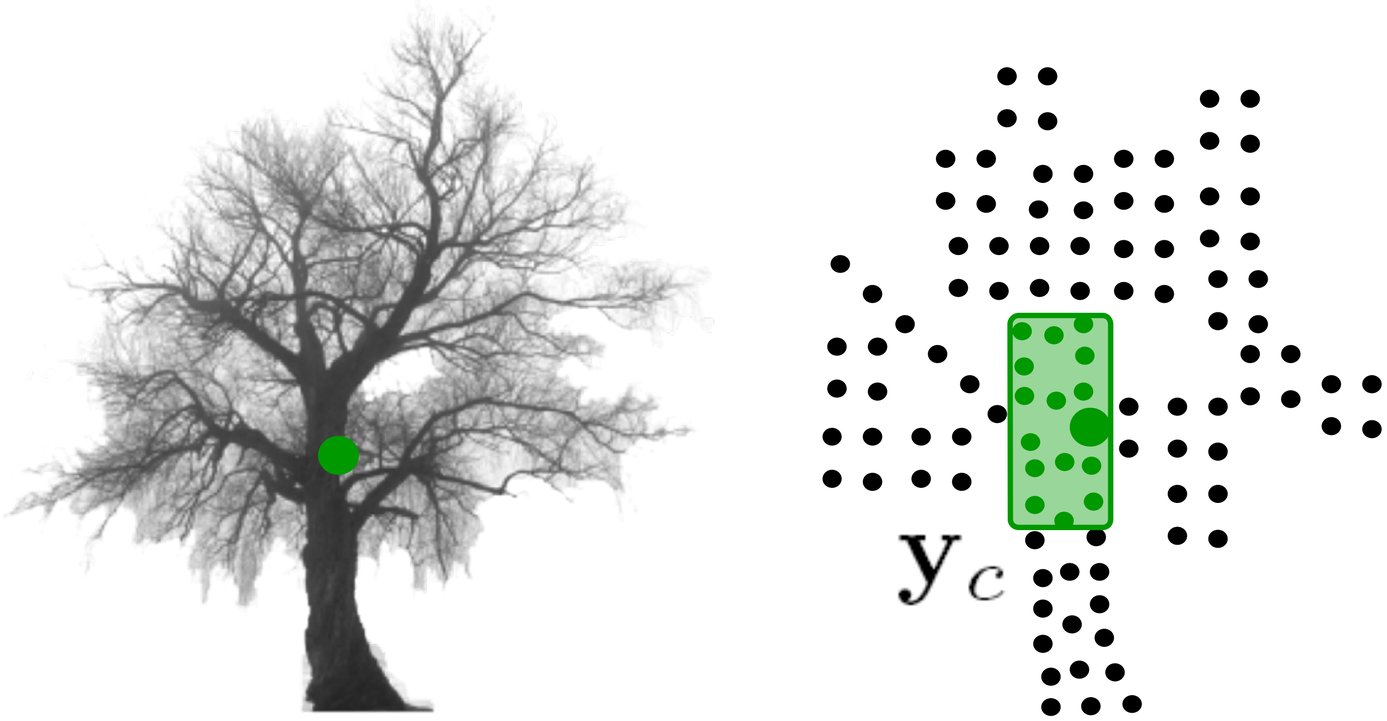


$$\mathbf{y}^* = \arg\max_{\mathbf{y}} \sum_{i=1}^{N} \boxed{\phi_i(y_i, \mathbf{x})} + \sum_{(ij)\in E} \phi_{ij}(y_i, y_j, \mathbf{x}) + \sum_{c\in S} \phi_c(\mathbf{y}_c, \mathbf{x})$$





$$\mathbf{y}^* = \arg\max_{\mathbf{y}} \sum_{i=1}^{N} \phi_i(y_i, \mathbf{x}) + \sum_{(ij)\in E} \boxed{\phi_{ij}(y_i, y_j, \mathbf{x})} + \sum_{c\in S} \phi_c(\mathbf{y}_c, \mathbf{x})$$



$$\mathbf{y}^* = \arg\max_{\mathbf{y}} \sum_{i=1}^{N} \phi_i(y_i, \mathbf{x}) + \sum_{(ij)\in E} \phi_{ij}(y_i, y_j, \mathbf{x}) + \sum_{c\in S} \phi_c(\mathbf{y}_c, \mathbf{x})$$

- Inference: Kohli et al., “Robust Higher Order Potentials for Enforcing Label Consistency”, IJCV 2009
- Learning: Munoz et al., “Contextual Classification with Functional Max-Margin Markov Networks”, CVPR 2009

Main Approaches

- Clustering
- Mean shift
- Spectral techniques
- Graph-based techniques
- Segmentation as boundary detection (skipped)