Lecture 11: Segmentation and Clustering

COS 429: Computer Vision

PRINCETON UNIVERSITY

Acknowledgments: T. Funkhouser, K. Grauman, S. Lazebnik, S. Seitz, X. Ren, S. Rusinkiewicz

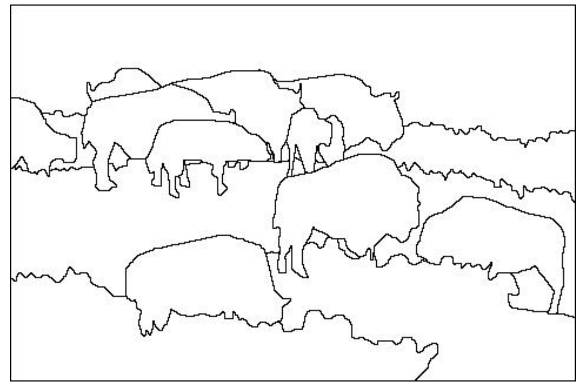
Segmentation and Clustering

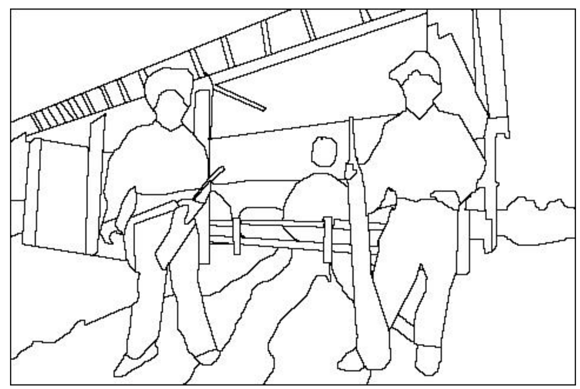
- Segmentation:
  Divide image
  into regions
  of similar contents

- Clustering:
  Aggregate pixels
  into regions
  of similar contents

Goal

- Separate image into coherent "regions"





Berkeley segmentation database:
http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

Lazebnik

Selective search for object recognition

Segmentation as Selective Search for Object Recognition.
Koen E. A. van de Sande, Jasper R. R. Uijlings, Theo Gevers, Arnold W. M. Smeulders
ICCV 2011

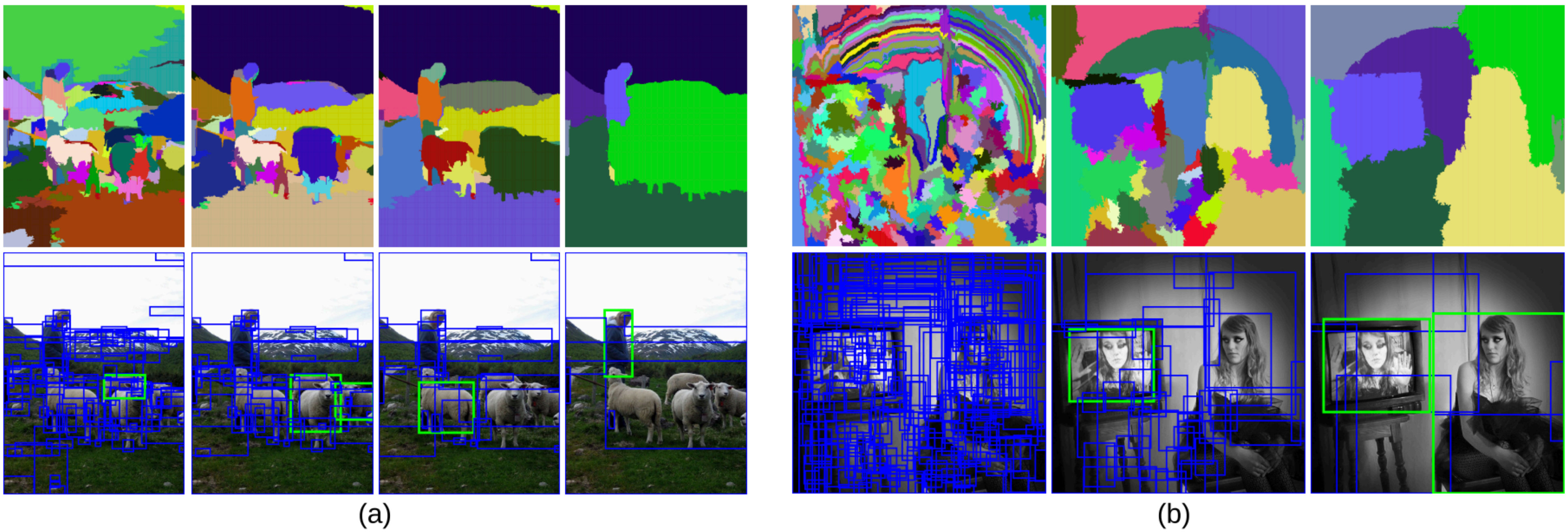
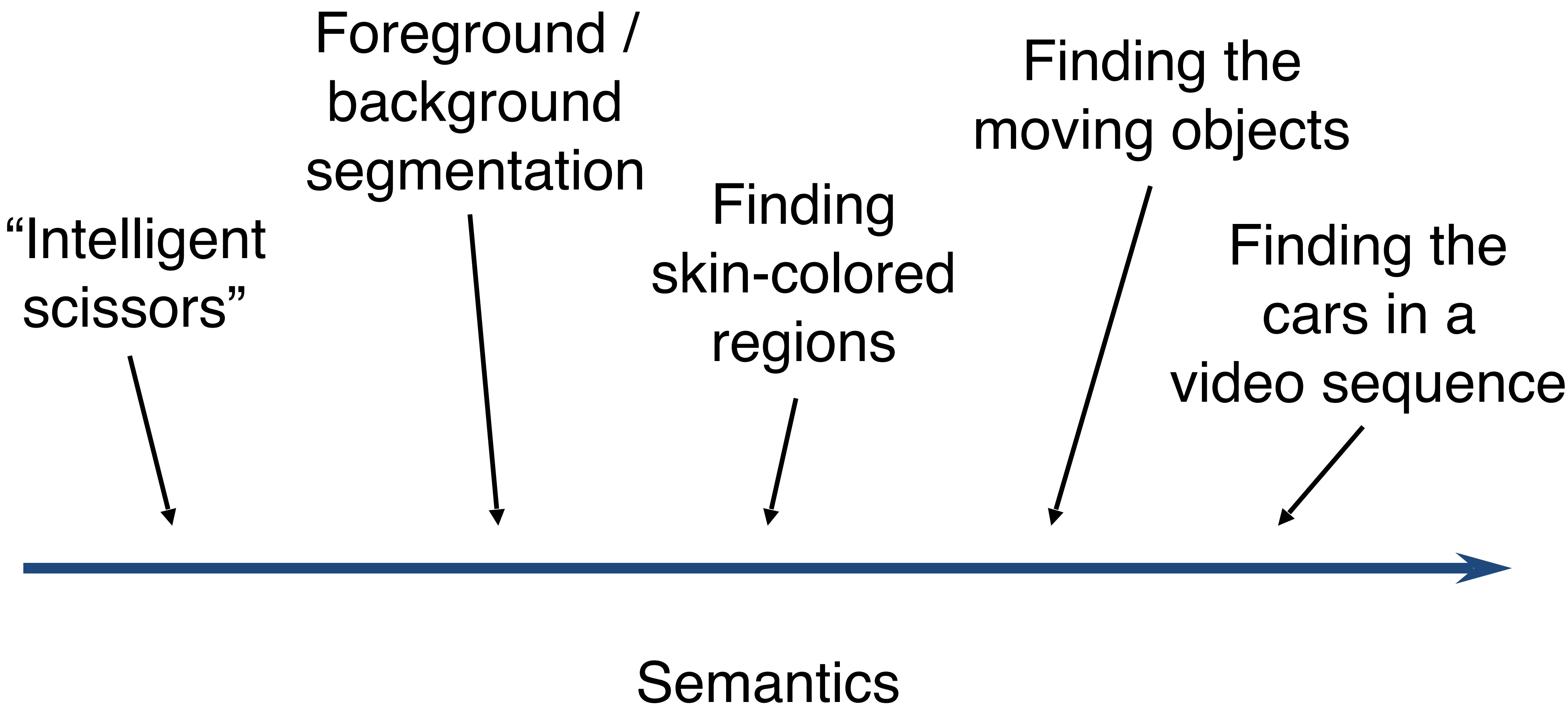


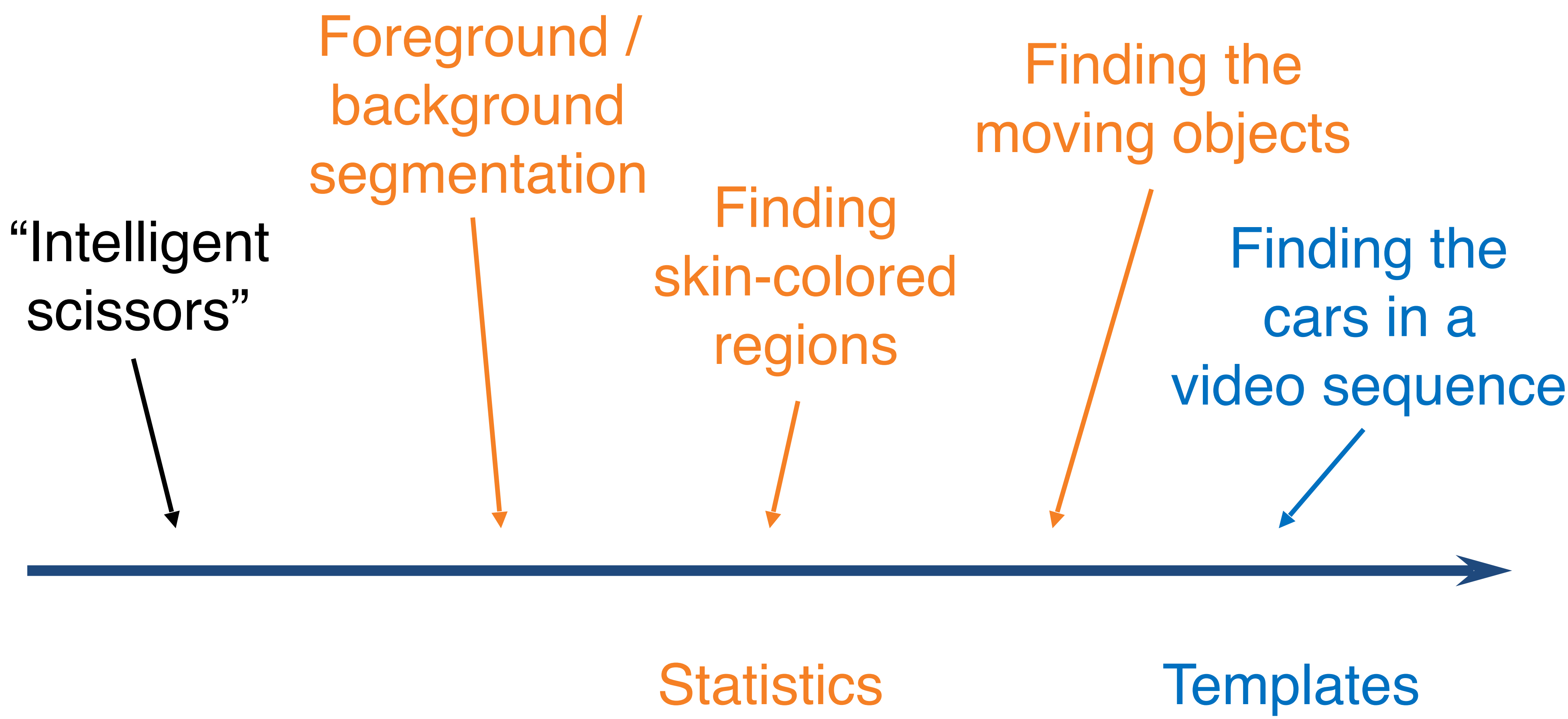
Figure 2: Two examples of our selective search showing the necessity of different scales. On the left we find many objects at different scales. On the right we necessarily find the objects at different scales as the girl is contained by the tv.

https://www.koen.me/research/selectivesearch/

Segmentation and Clustering Applications

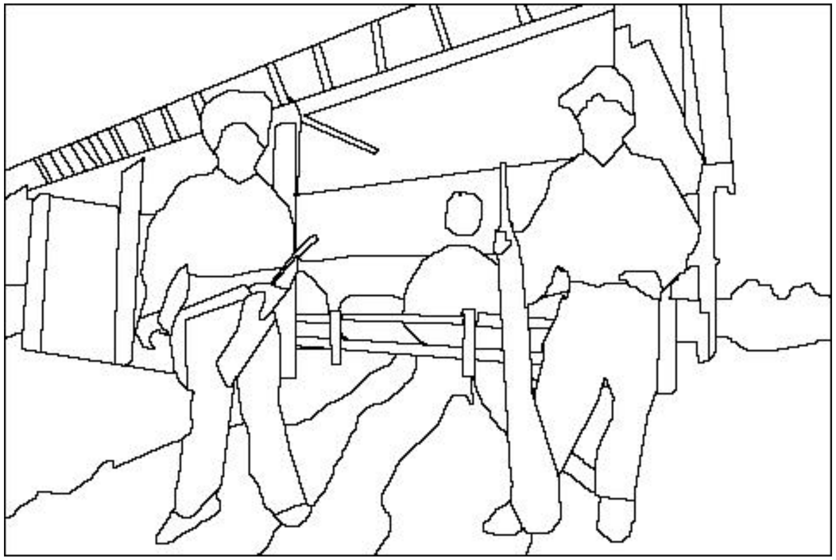


Segmentation and Clustering Applications



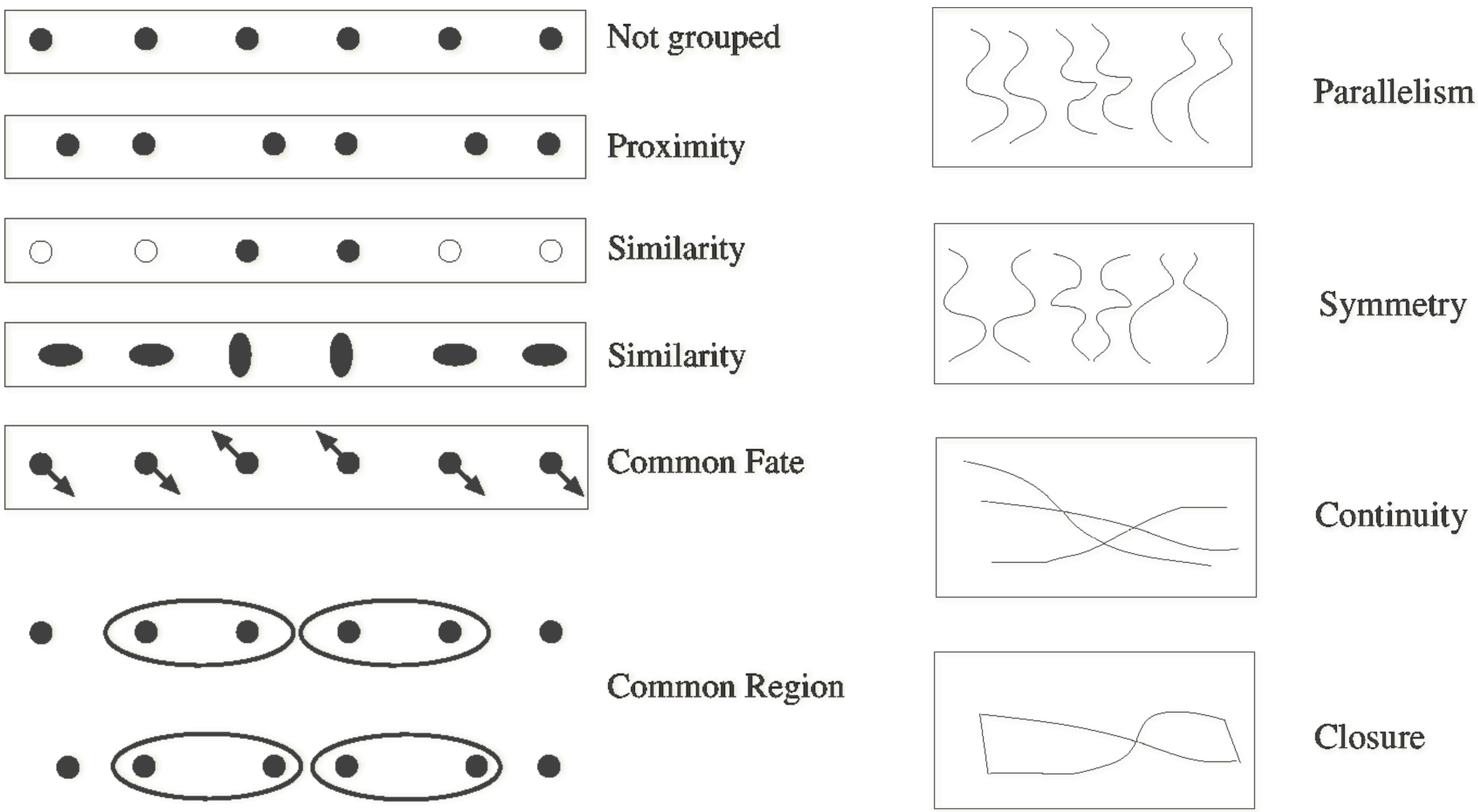
Questions

- What is coherent?
  - Similar color?
  - Similar texture?
  - Spatial proximity?



- What kinds of regions?
  - Nearly convex?
  - Smooth boundaries?
  - Nearly equal sizes?
  - What granularity?

Gestalt Grouping Cues



Segmentation and Clustering

- Defining regions
  - Should they be compact? Smooth boundary?
- Defining similarity
  - Color, texture, motion, …
- Defining similarity of regions
  - Minimum distance, mean, maximum

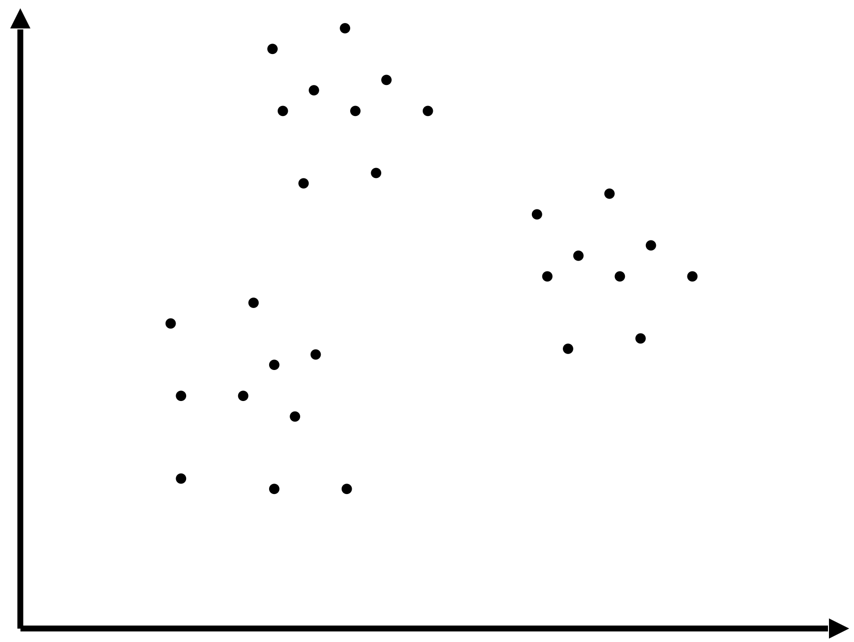
Let’s start simple

Clustering Based on Color

- Let's make a few concrete choices:
  - Arbitrary regions
  - Similarity based on color only
  - Similarity of regions = distance between mean colors

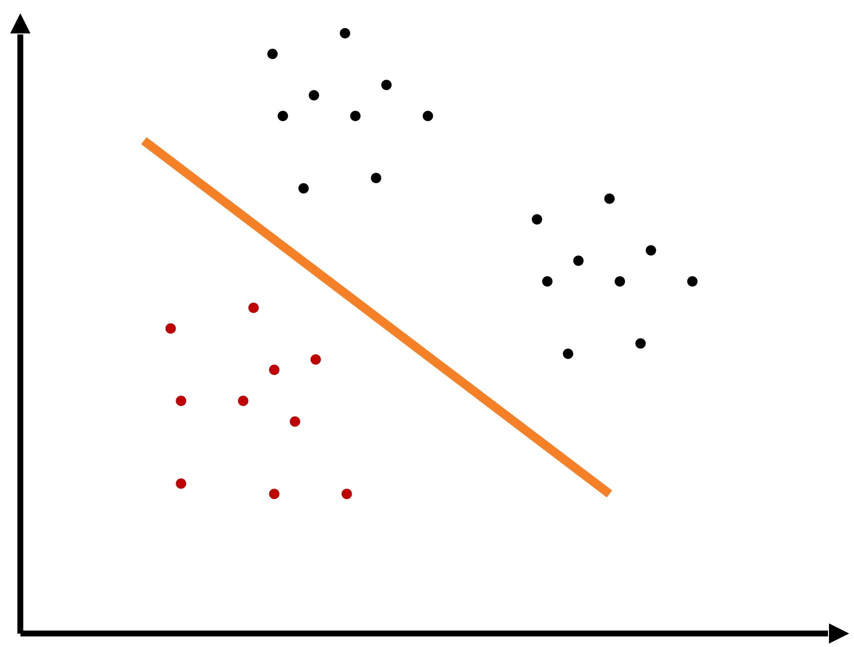
Divisive Clustering

- Start with whole image in one cluster
- Iterate:
  - Find cluster with largest intra-cluster variation
  - Split into two pieces that yield largest inter-cluster distance
- Stopping criteria?



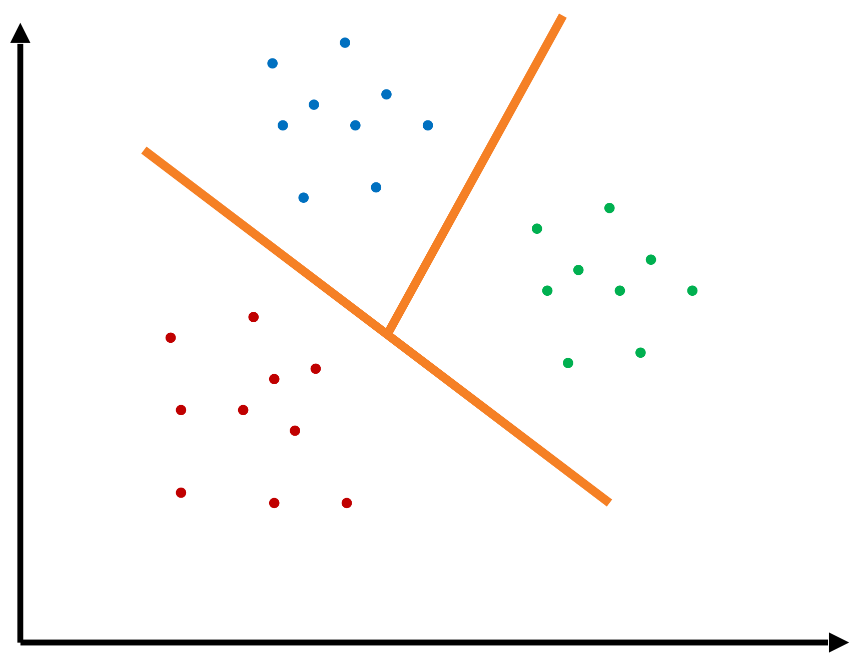
Divisive Clustering

- Start with whole image in one cluster
- Iterate:
  - Find cluster with largest intra-cluster variation
  - Split into two pieces that yield largest inter-cluster distance
- Stopping criteria?



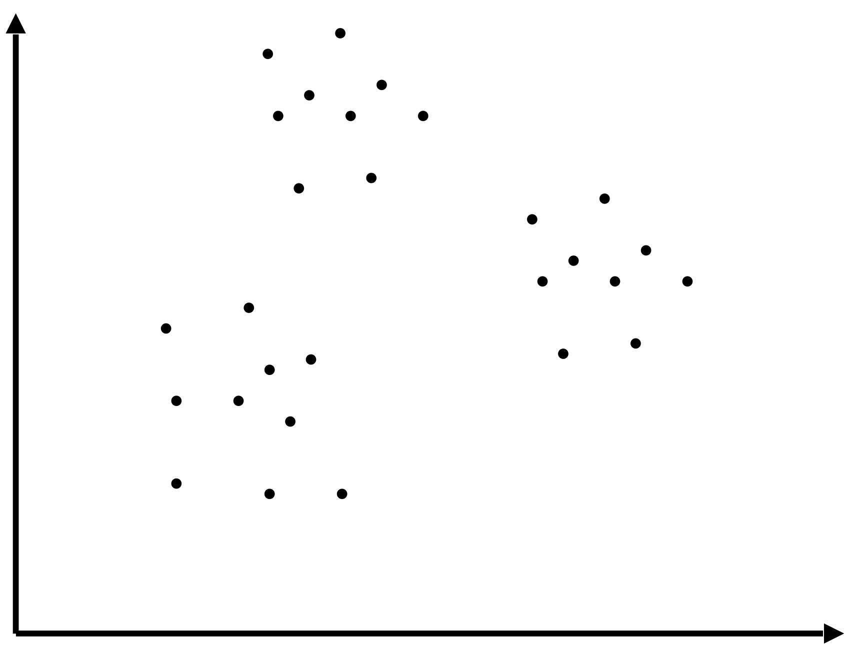
Divisive Clustering

- Start with whole image in one cluster
- Iterate:
  - Find cluster with largest intra-cluster variation
  - Split into two pieces that yield largest inter-cluster distance
- Stopping criteria?



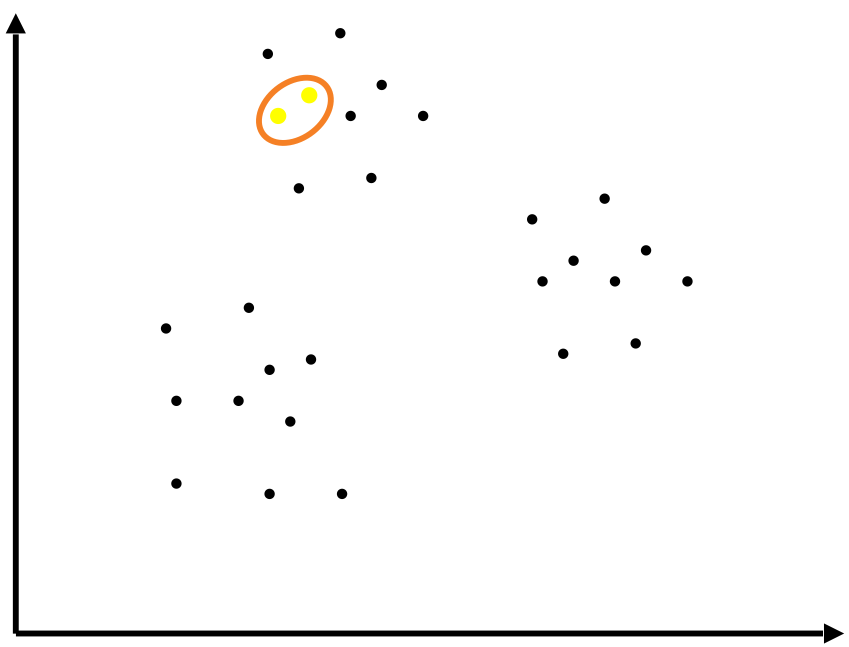
Hierarchical Clustering

- Start with each pixel in its own cluster
- Iterate:
  - Find pair of clusters with smallest inter-cluster distance
  - Merge
- Stopping criteria?



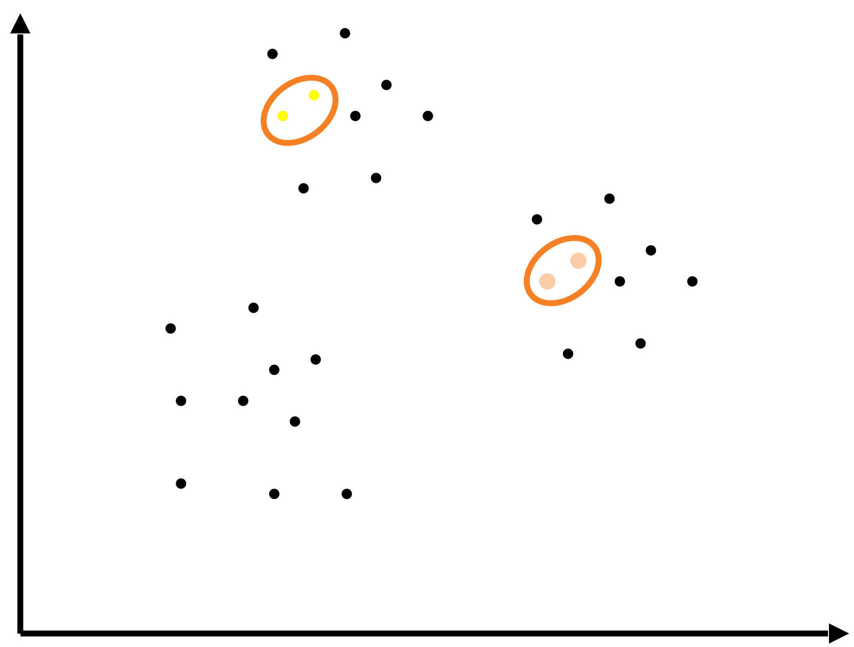
Hierarchical Clustering

- Start with each pixel in its own cluster
- Iterate:
  - Find pair of clusters with smallest inter-cluster distance
  - Merge
- Stopping criteria?



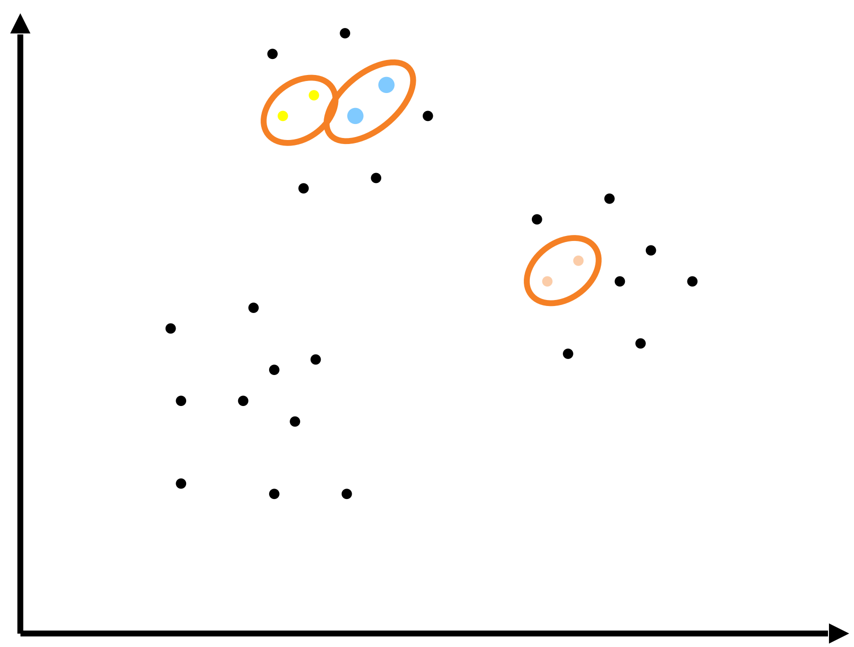
Hierarchical Clustering

- Start with each pixel in its own cluster
- Iterate:
  - Find pair of clusters with smallest inter-cluster distance
  - Merge
- Stopping criteria?



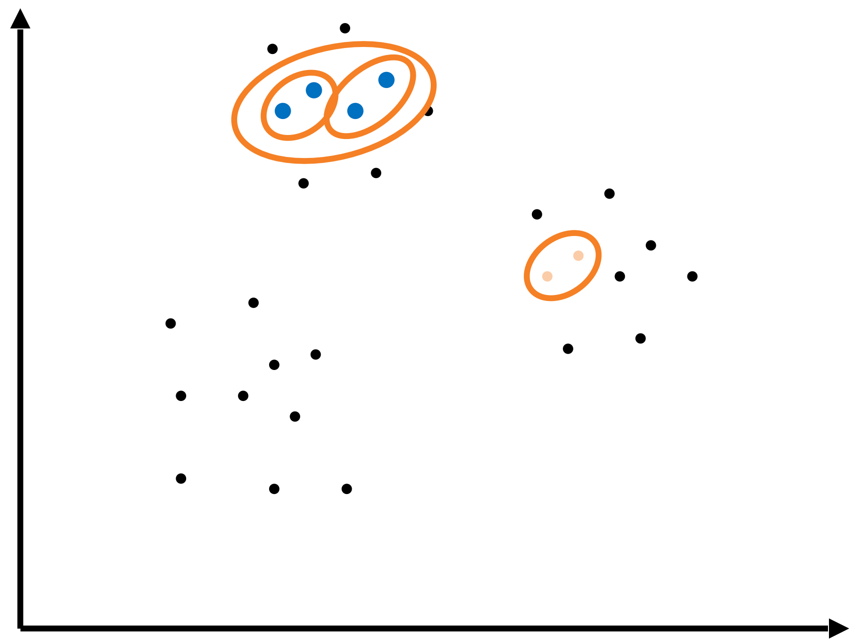
Hierarchical Clustering

- Start with each pixel in its own cluster
- Iterate:
  - Find pair of clusters with smallest inter-cluster distance
  - Merge
- Stopping criteria?



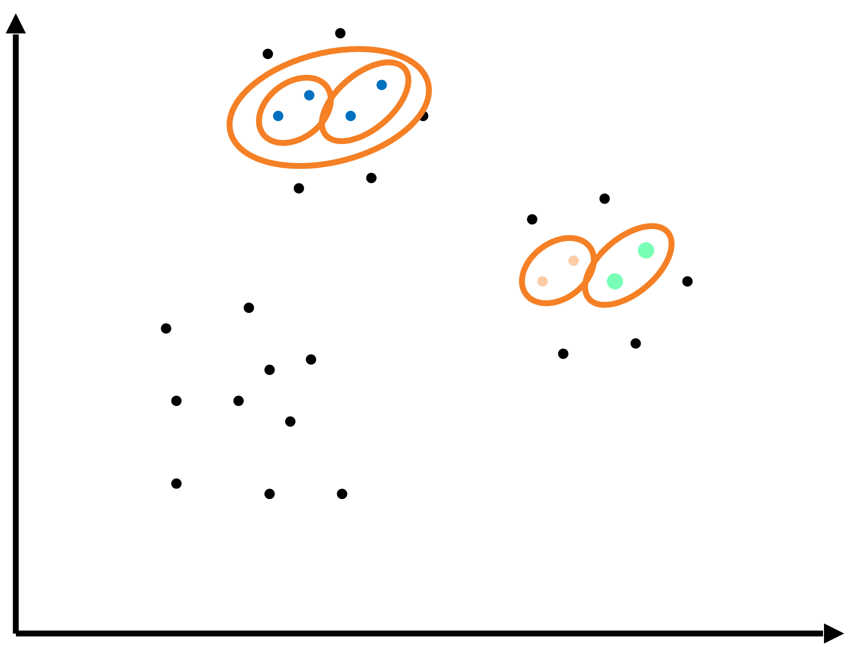
Hierarchical Clustering

- Start with each pixel in its own cluster
- Iterate:
  - Find pair of clusters with smallest inter-cluster distance
  - Merge
- Stopping criteria?



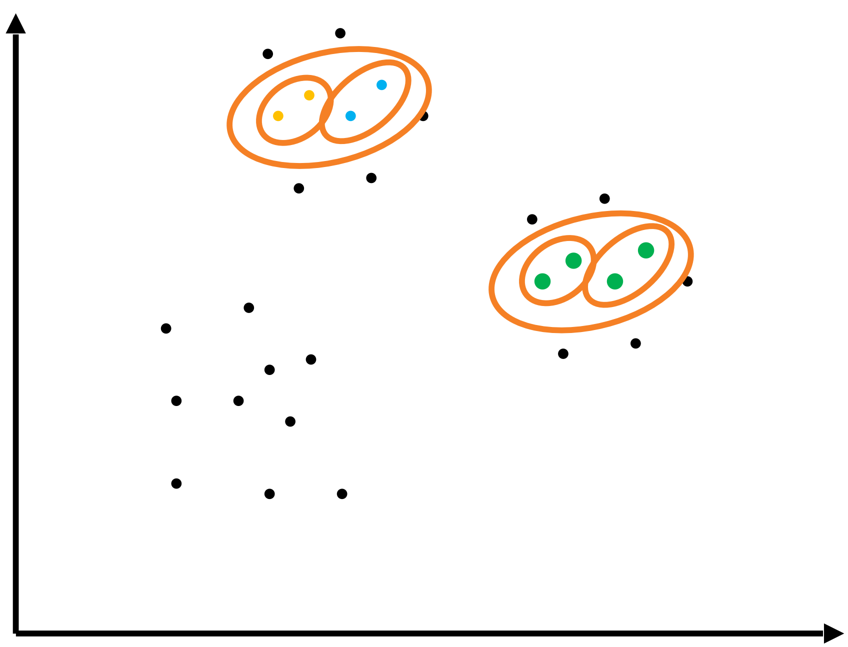
Hierarchical Clustering

- Start with each pixel in its own cluster
- Iterate:
  - Find pair of clusters with smallest inter-cluster distance
  - Merge
- Stopping criteria?



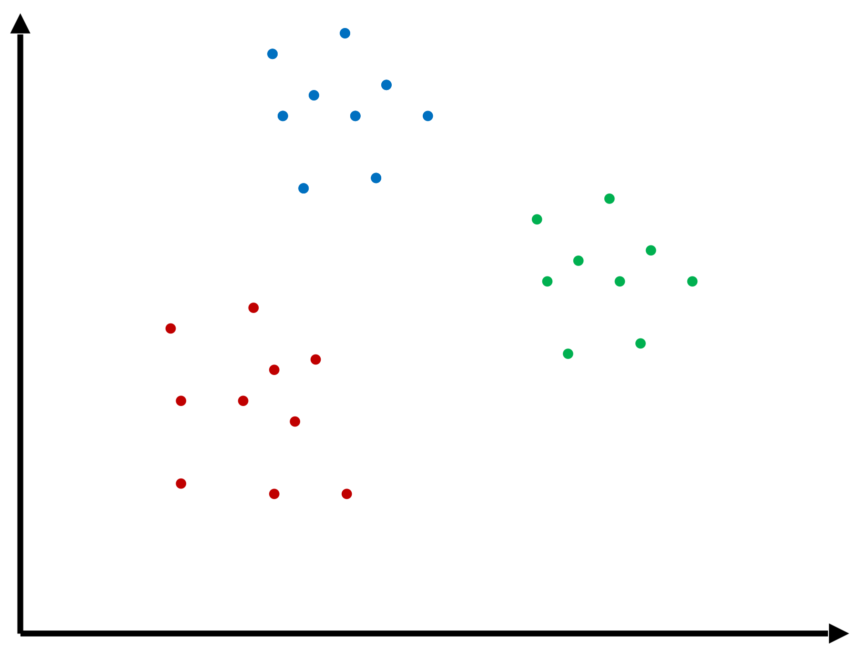
Hierarchical Clustering

- Start with each pixel in its own cluster
- Iterate:
  - Find pair of clusters with smallest inter-cluster distance
  - Merge
- Stopping criteria?



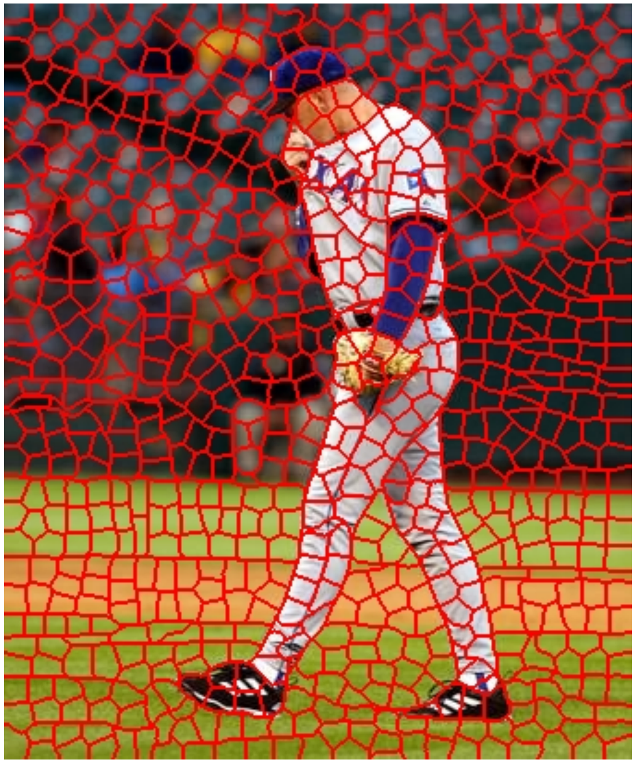
Hierarchical Clustering

- Start with each pixel in its own cluster
- Iterate:
  - Find pair of clusters with smallest inter-cluster distance
  - Merge
- Stopping criteria?



Hierarchical Clustering

- Conservative stopping criteria yields "superpixels", which can be used as starting point for more complex algorithms



Ren et al.

Problems with These Algorithms

- Greedy
  - Decisions made early in process dictate final result
- Making "good" early decisions is hard/expensive
  - Many possibilities at each iteration
  - Computing "good" merge or split is expensive
- Heuristics to speed things up:
  - For agglomerative clustering, approximate each cluster by average for distance computations
  - For divisive clustering, use summary (histogram) of a region to compute split

More advanced: iterative, better metrics, …

k-means Clustering

Instead of merging or splitting, start out with the clusters and move them around

1. Pick number of clusters k
2. Randomly scatter k “cluster centers” in color space
3. Repeat:
    a. Assign each data point to its closest cluster center
    b. Move each cluster center to the mean of the points assigned to it

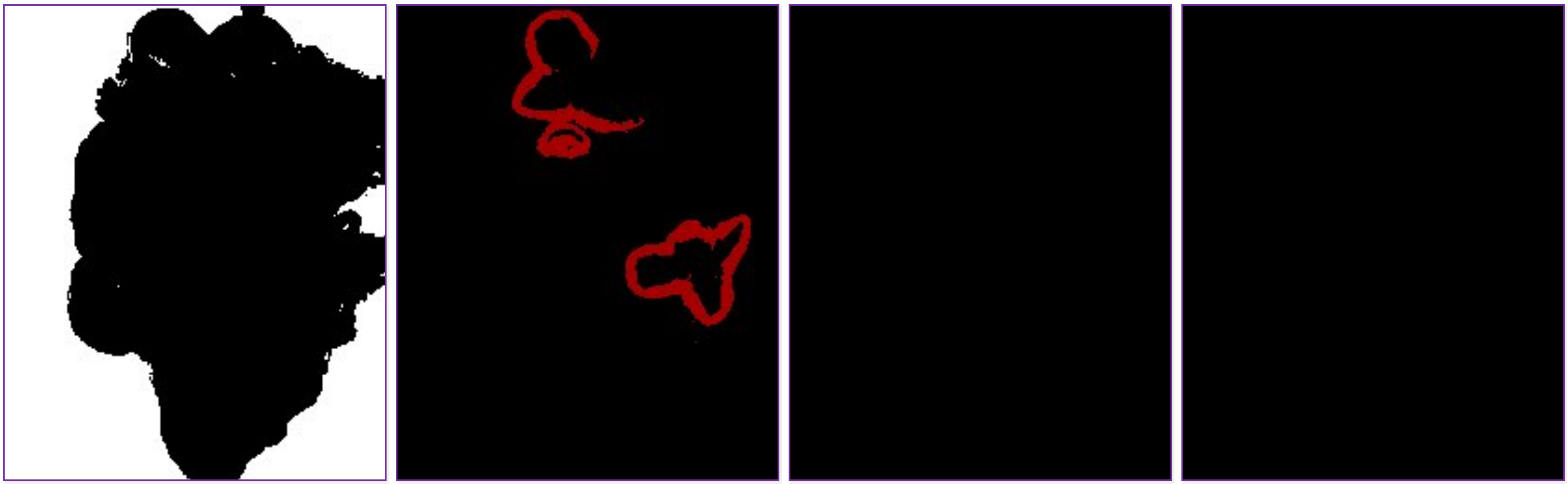
Results of Clustering

Original Image

k-means, *k*=5

k-means, *k*=11

Results of Clustering

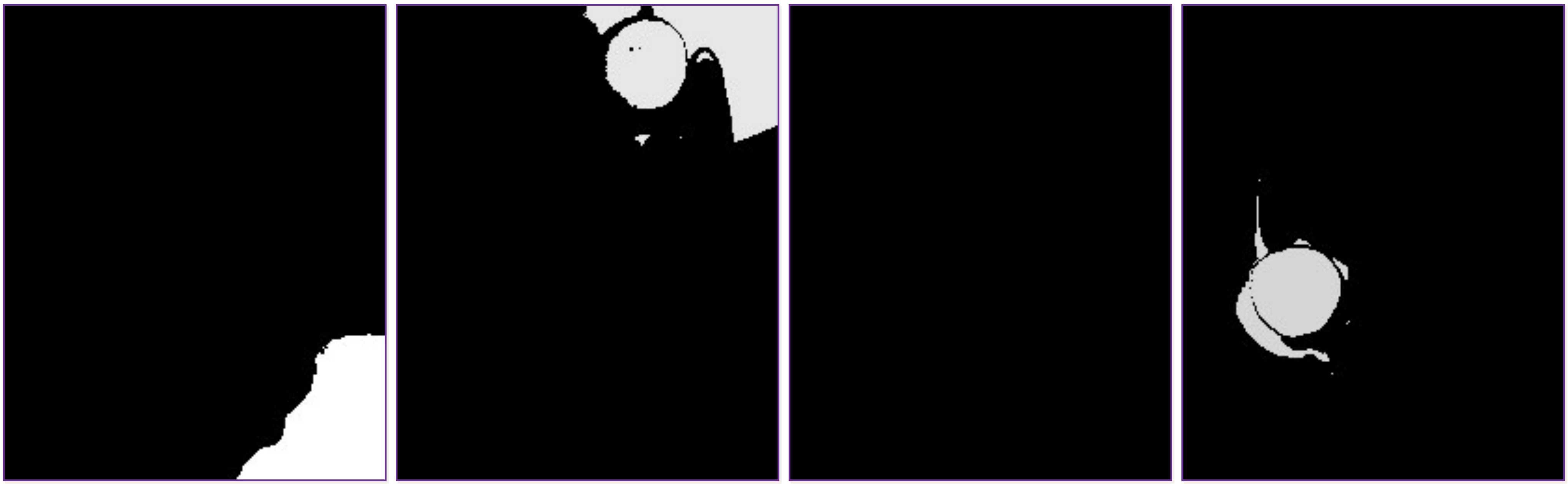


Sample clusters with *k*-means clustering
based on color

Other Distance Measures

- Suppose we want to have compact regions
- New feature space: 5D
  (2 spatial coordinates, 3 color components)
- Points close in this space are close both in color and in actual proximity

Results of Clustering



Sample clusters with *k*-means clustering
based on color and distance

Other Distance Measures

- Problem with simple Euclidean distance: what if coordinates range from 0-1000 but colors only range from 0-255?
  - Depending on how things are scaled, gives different weight to different kinds of data
- Weighted Euclidean distance: adjust weights to emphasize different dimensions

$$dist(\mathbf{x}, \mathbf{y})^2 = \sum_i c_i (x_i - y_i)^2$$

Mahalanobis Distance

- Automatically assign weights based on actual variation in the data

$$dist(\mathbf{x}, \mathbf{y})^2 = (\mathbf{x} - \mathbf{y})^T C^{-1} (\mathbf{x} - \mathbf{y})$$

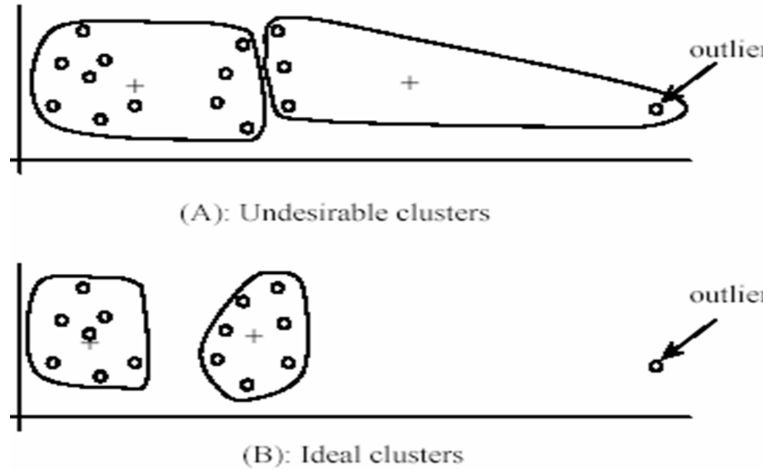
  where C is covariance matrix of all points

- Gives each dimension “equal” weight
- Also accounts for correlations between different dimensions

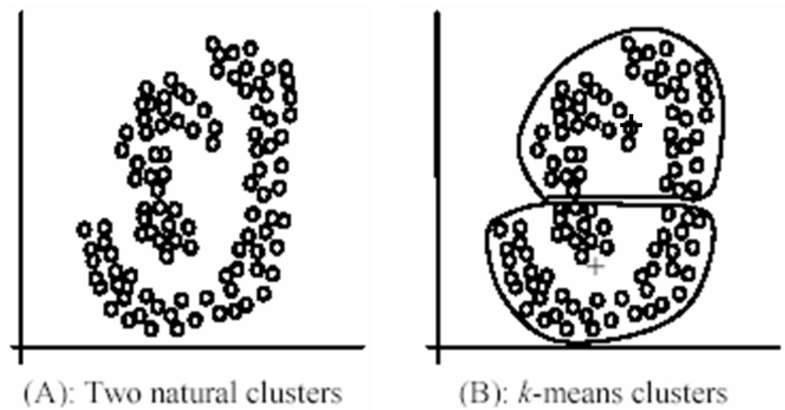
k-means Pros and Cons?

k-means Pros and Cons

- Pros
  - Very simple method
- Cons
  - Need to pick *k*
  - Converges to a local minimum
  - Sensitive to initialization
  - Sensitive to outliers
  - Only finds “spherical” clusters



Sensitive to outliers



Spherical clusters

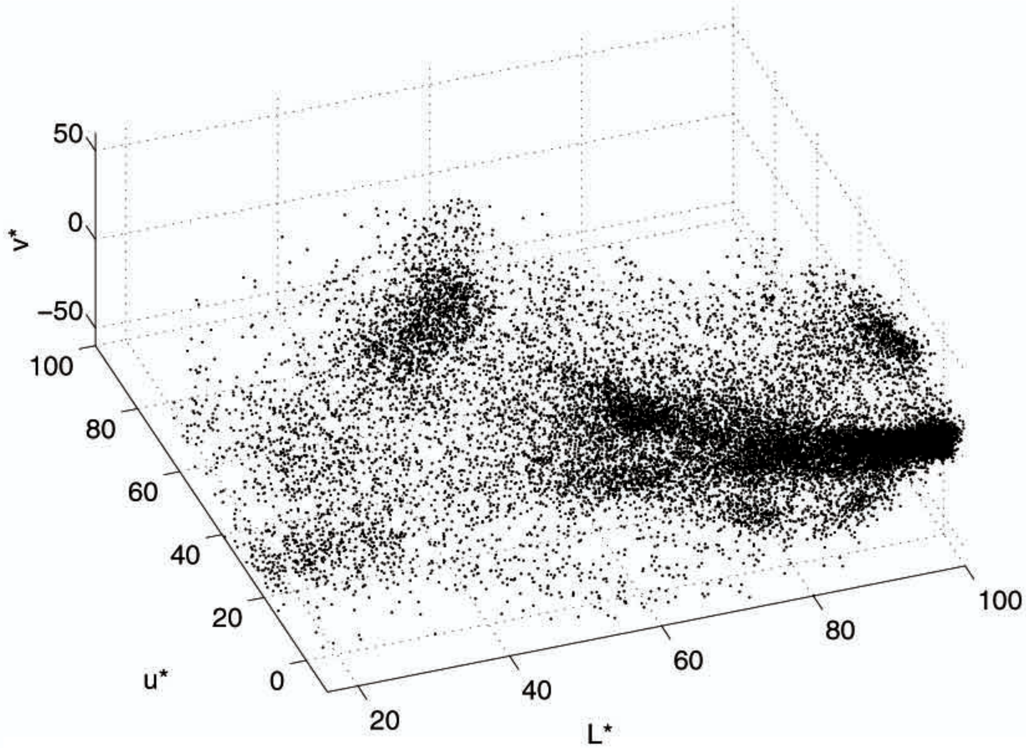
Grauman

Mean shift

Mean Shift Clustering

- Seek *modes* (peaks) of density in feature space

Image



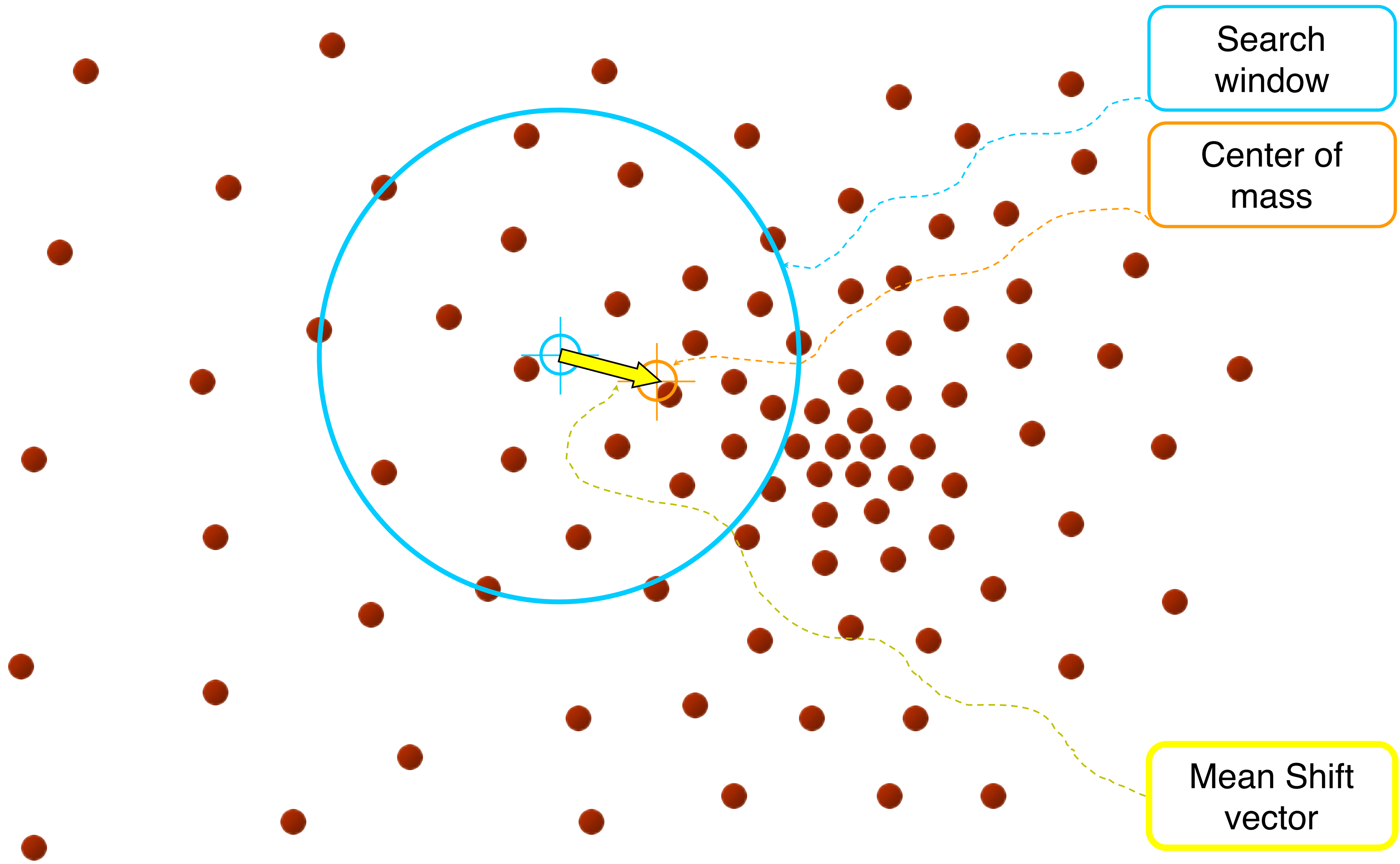
Feature space
(color values)

Ukrainitz & Sarel

Mean Shift Clustering

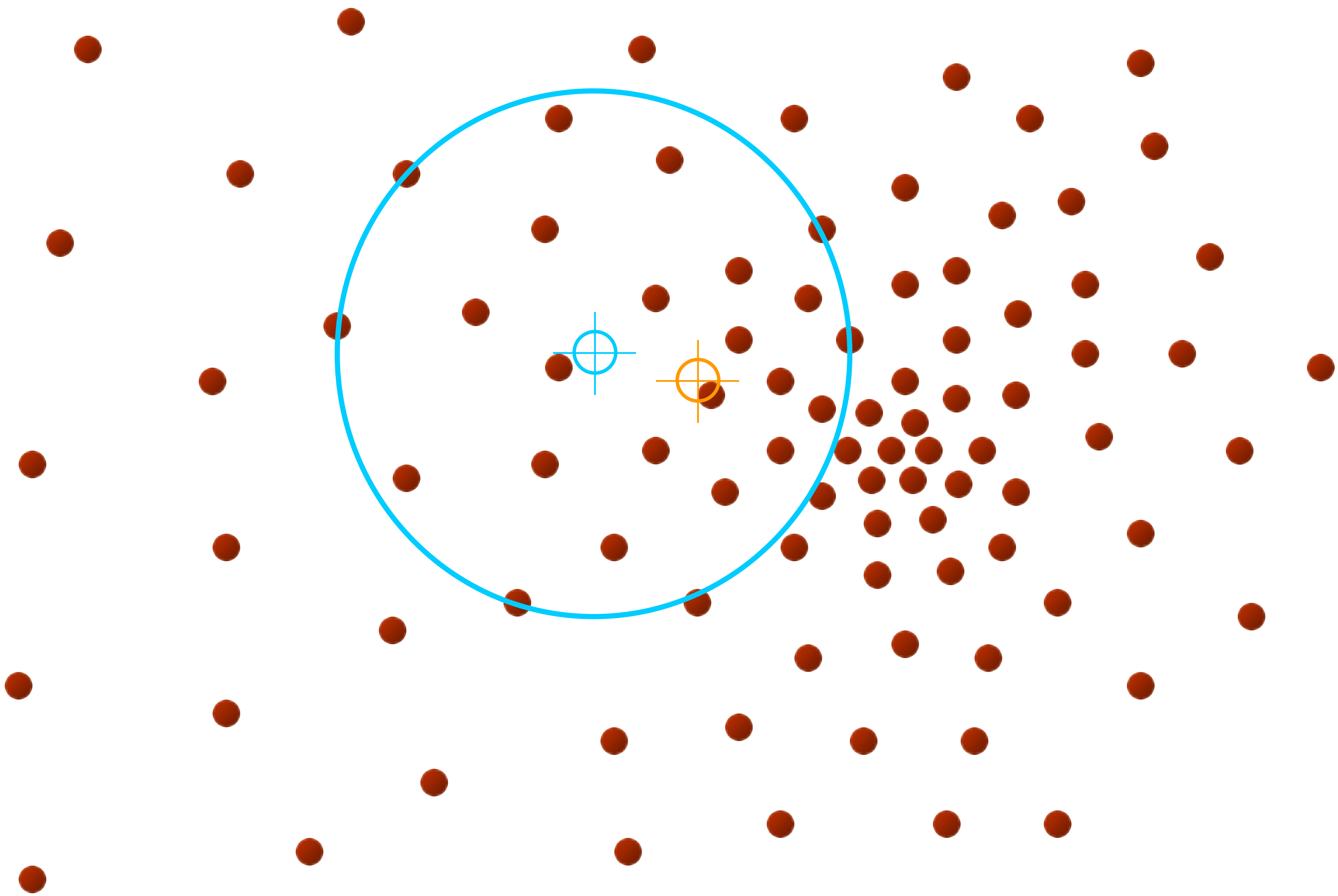
- Algorithm:
  - Initialize windows at individual feature points
  - Perform mean shift for each window until convergence
  - Merge windows that end up near the same “peak” or mode

Mean Shift Clustering



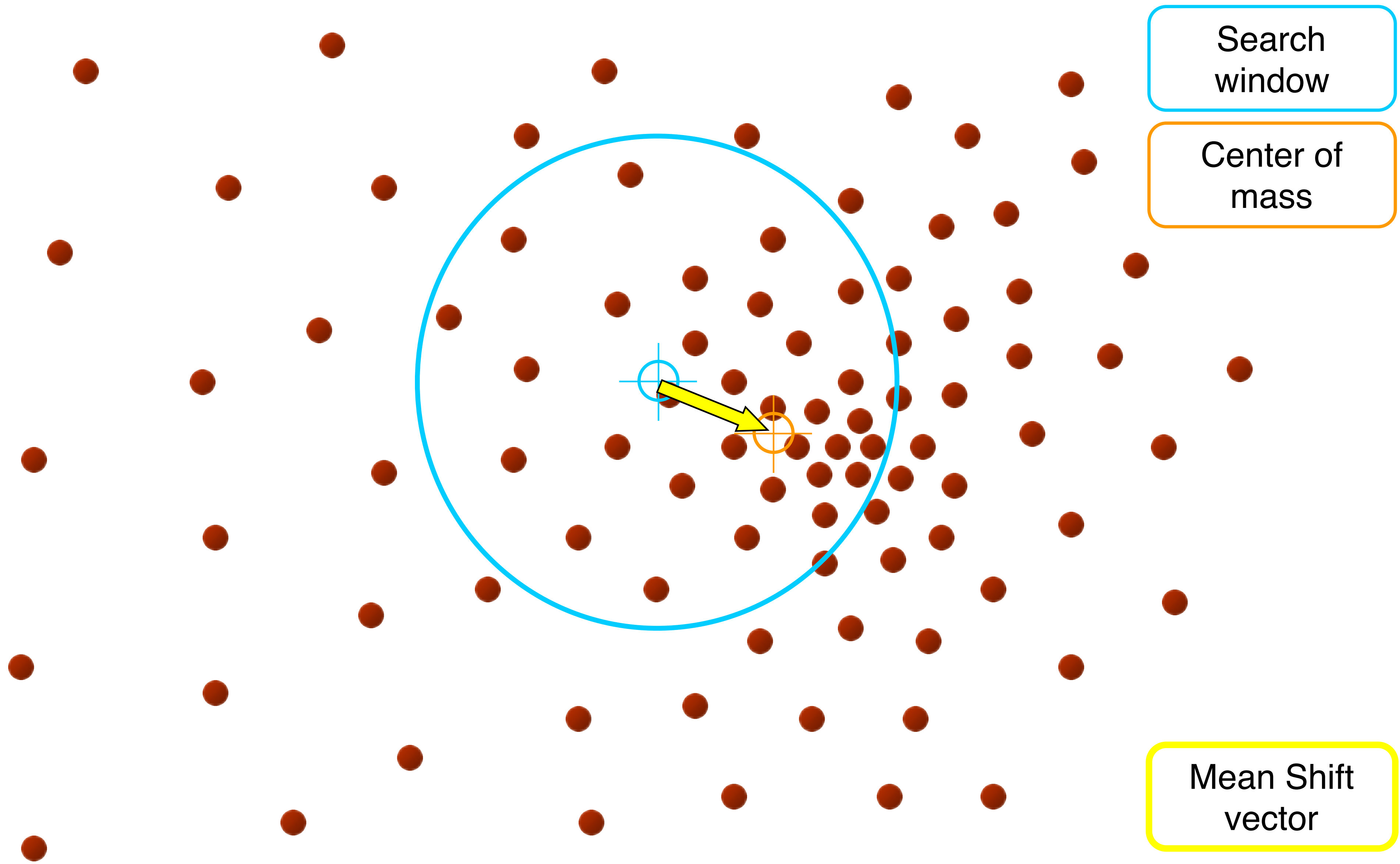
Ukrainitz & Sarel

Mean Shift Clustering



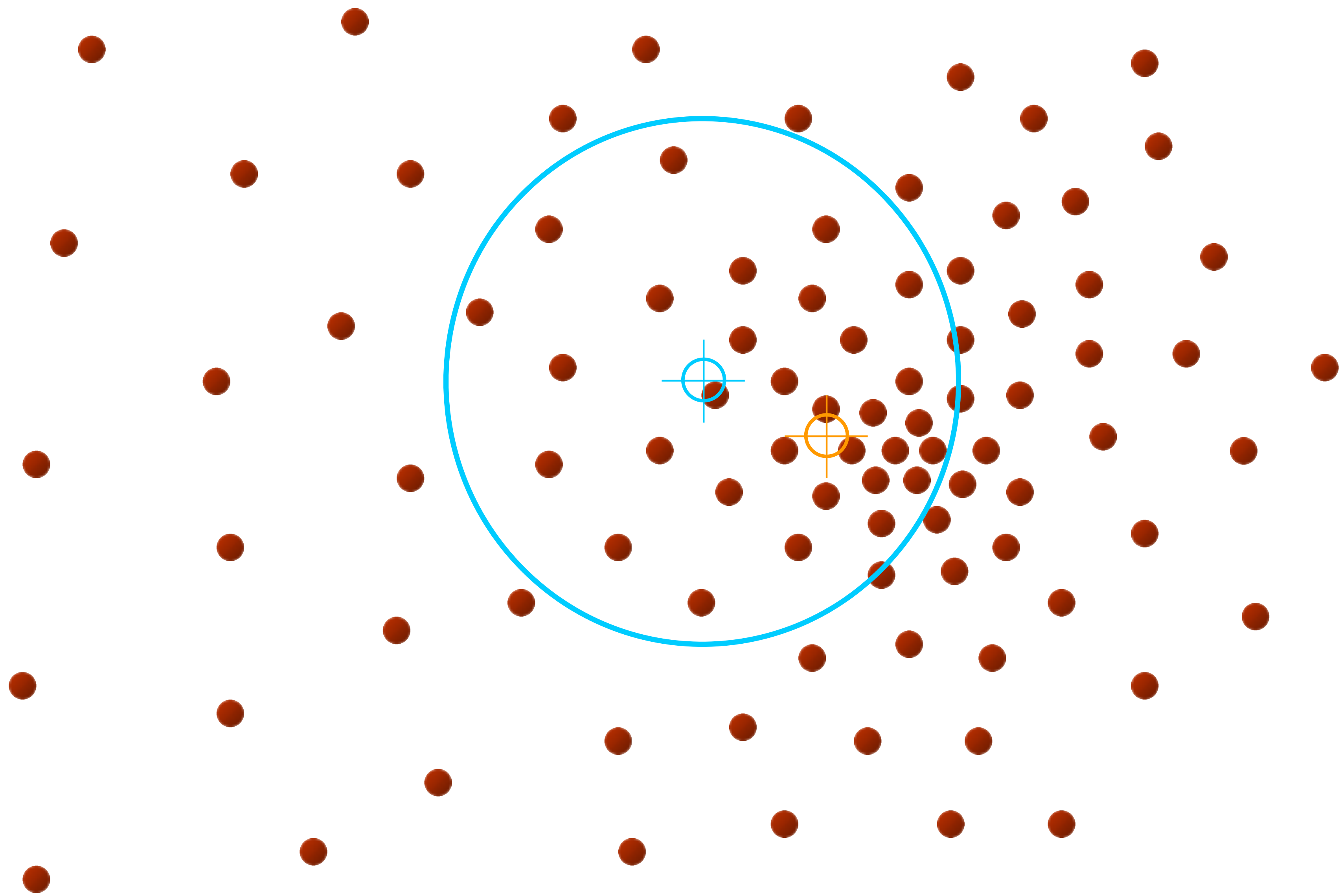
Ukrainitz & Sarel

Mean Shift Clustering



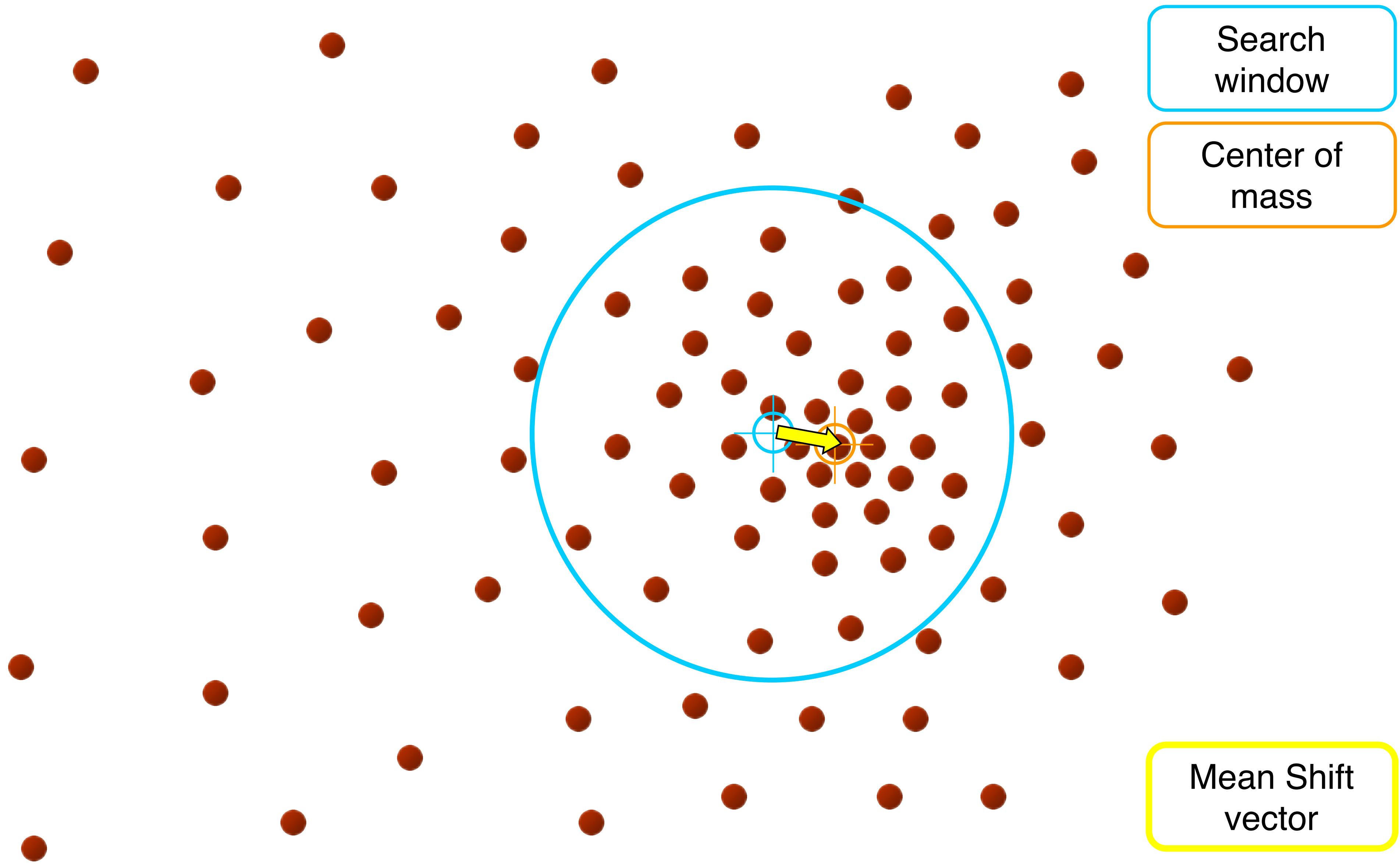
Ukrainitz & Sarel

Mean Shift Clustering



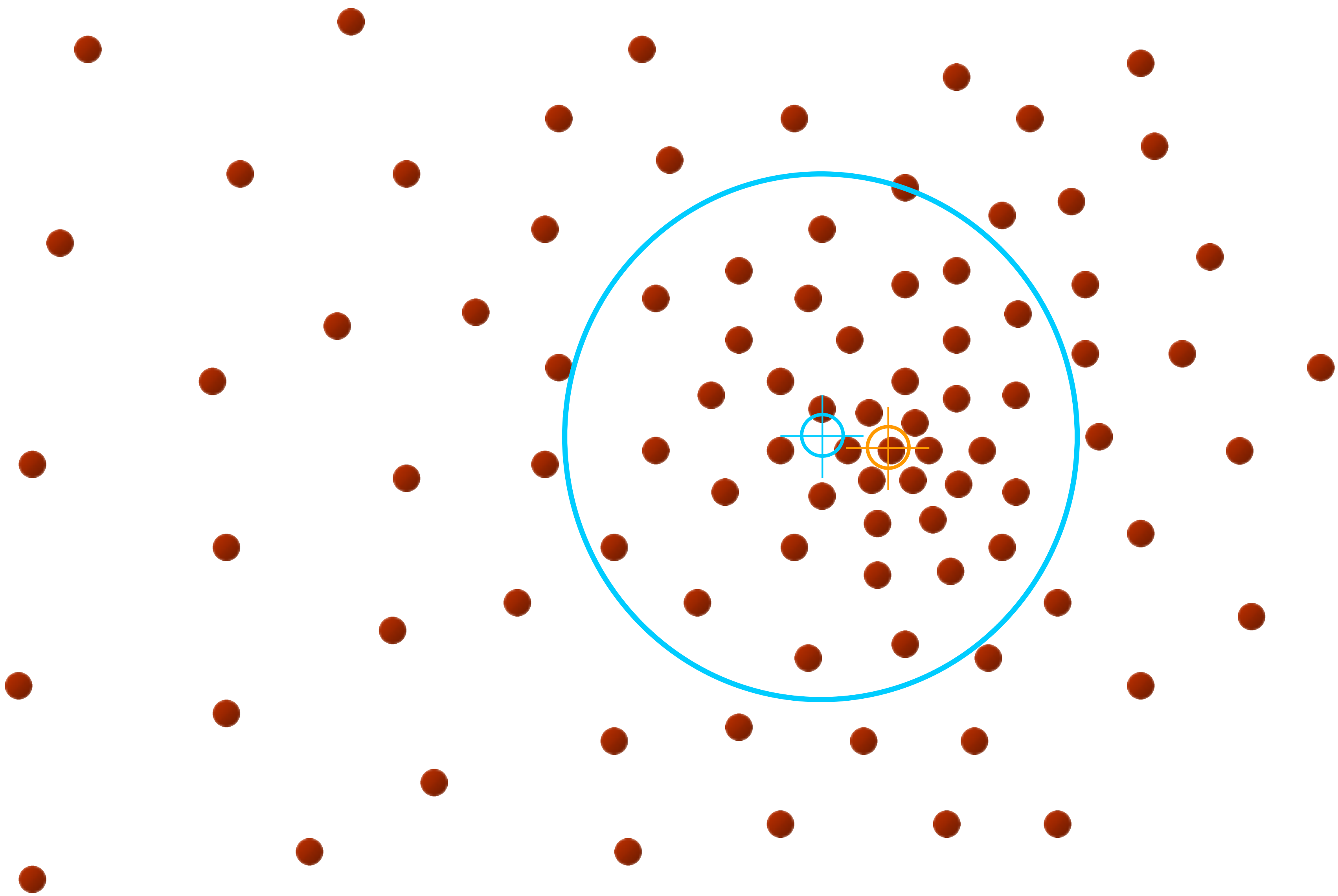
Ukrainitz & Sarel

Mean Shift Clustering



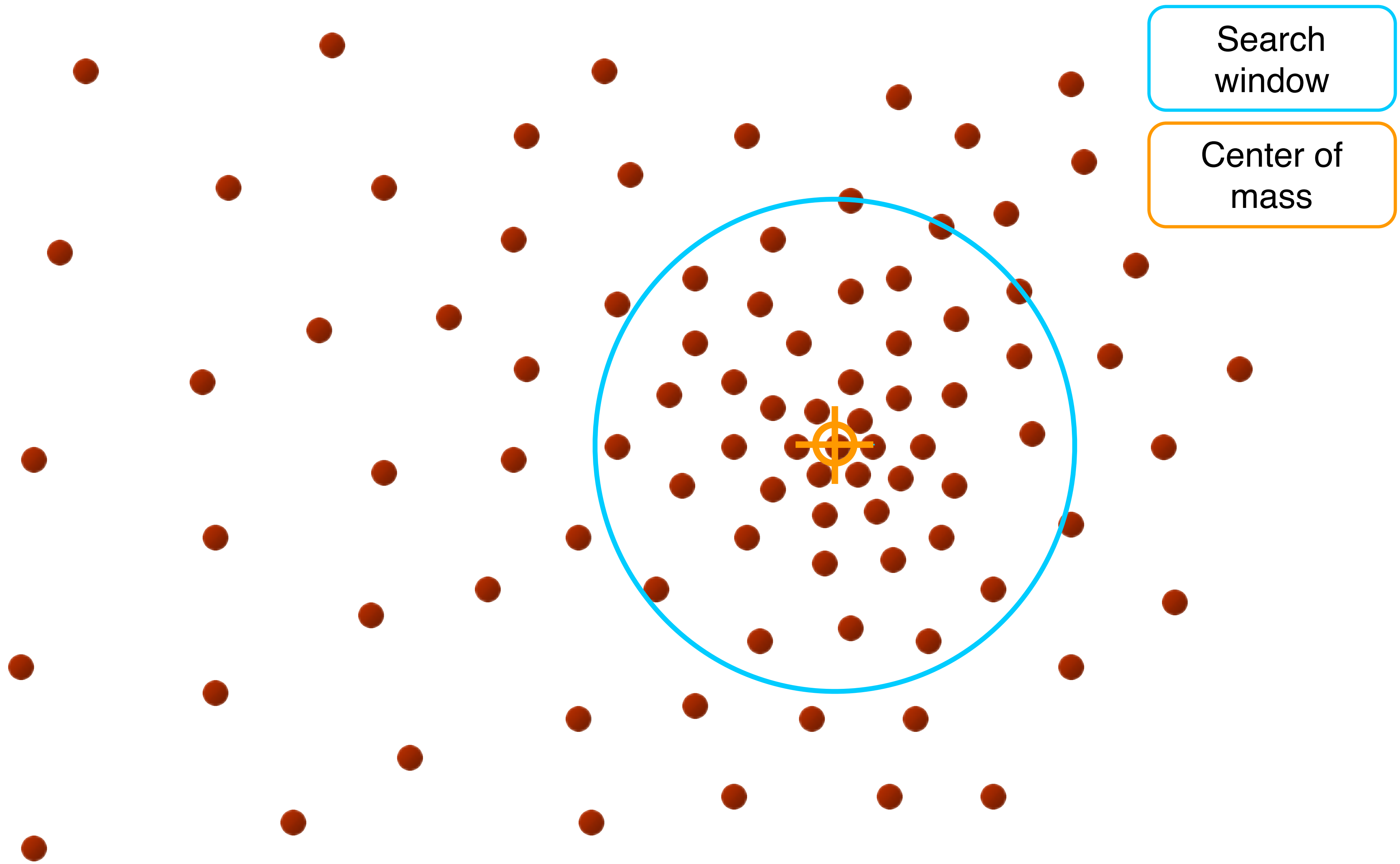
Ukrainitz & Sarel

Mean Shift Clustering



Ukrainitz & Sarel

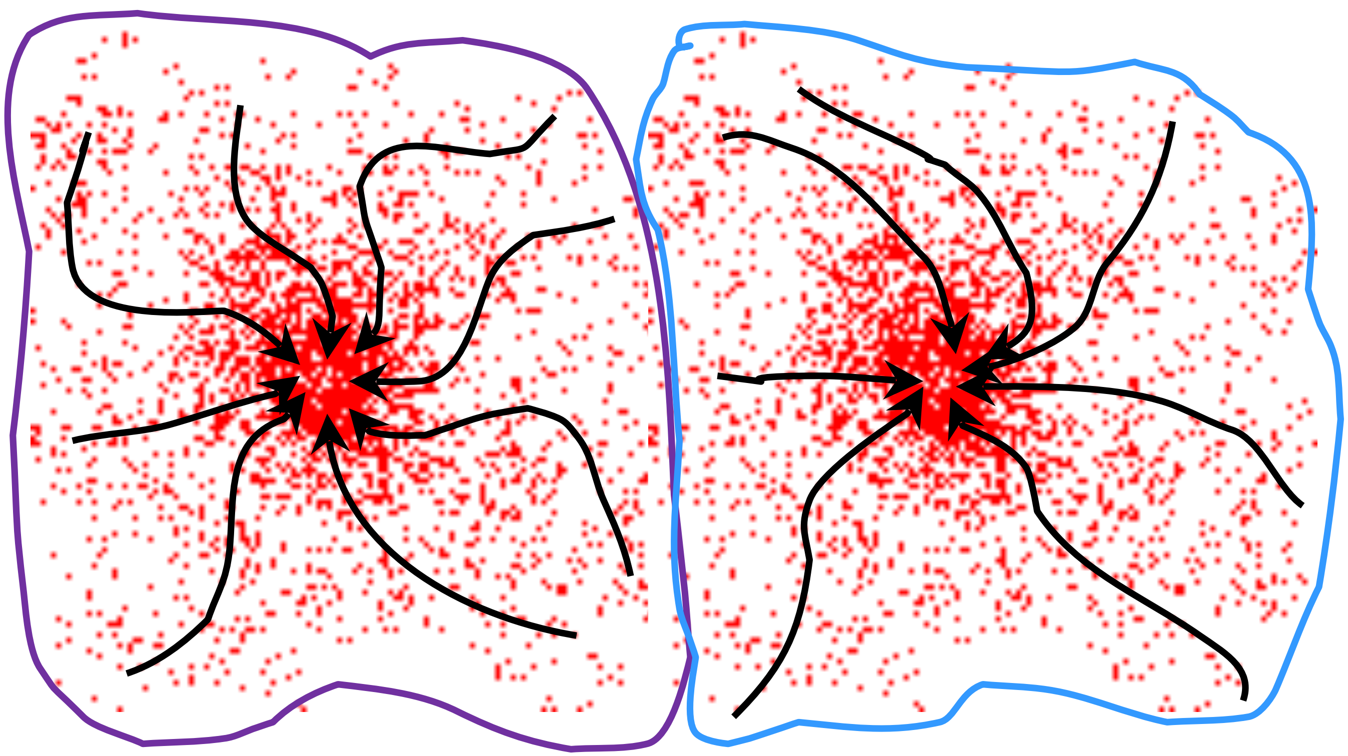
Mean Shift Clustering



Ukrainitz & Sarel

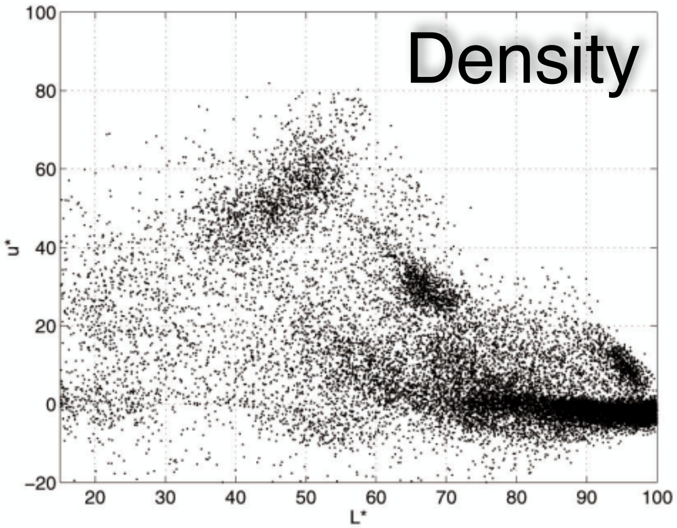
Mean Shift Clustering

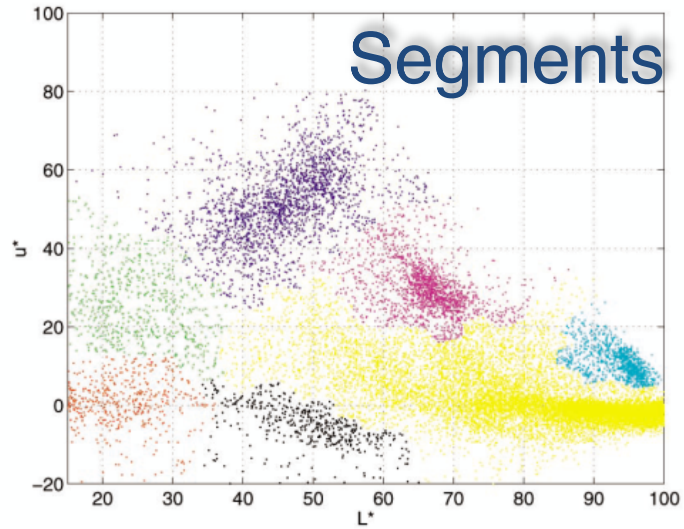
- Cluster data points in the attraction basin of a mode
  - Separate segment for each mode
  - Assign points to segments based on which mode is at the end of their mean shift trajectories

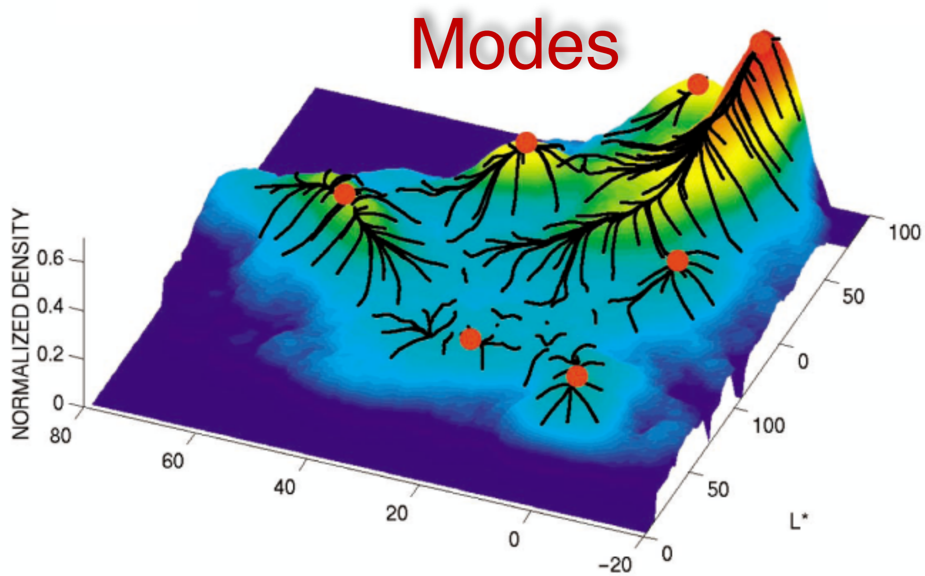


Ukrainitz & Sarel

Mean Shift Clustering

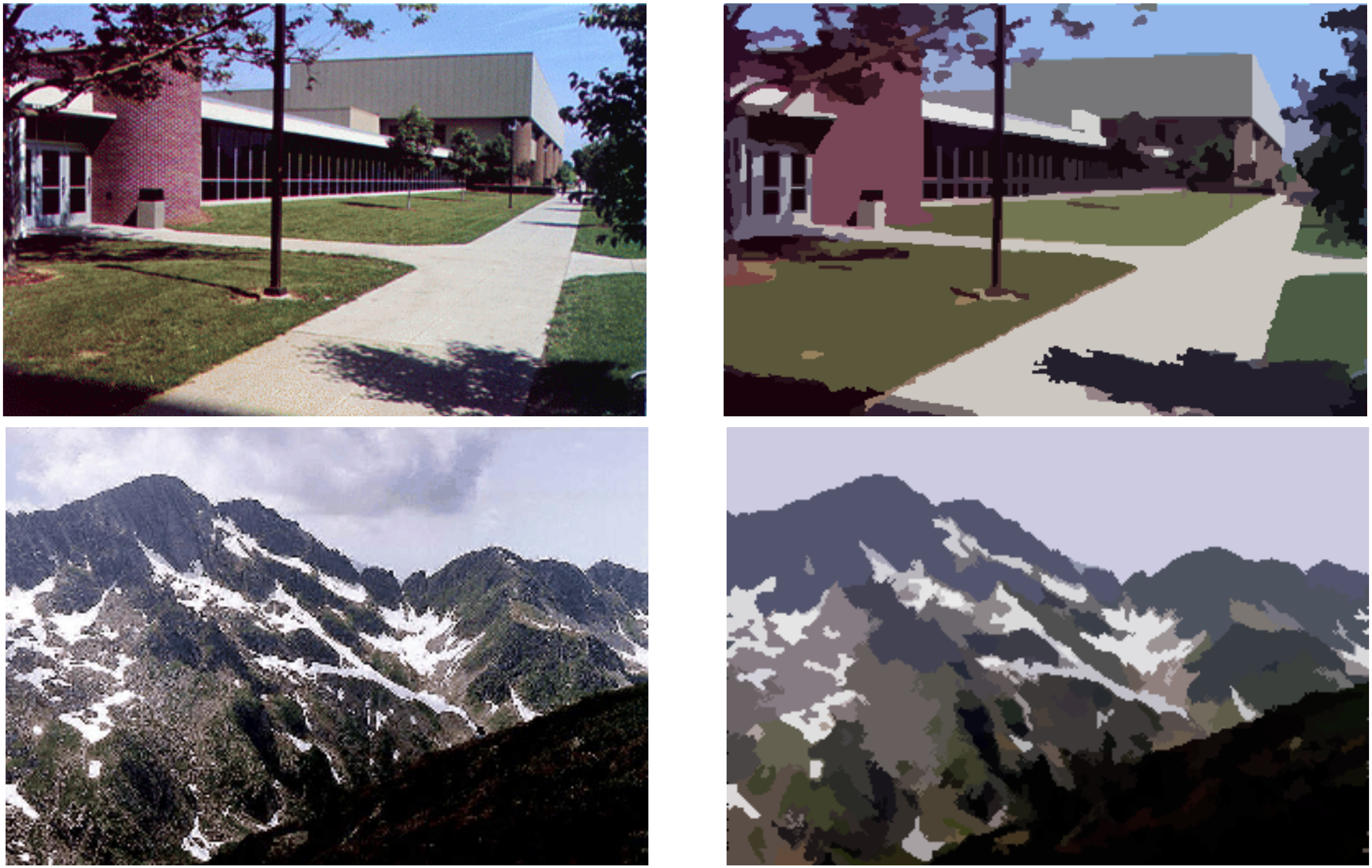






Ukrainitz & Sarel

Mean Shift Results



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

Mean Shift Results

Mean Shift Pros and Cons

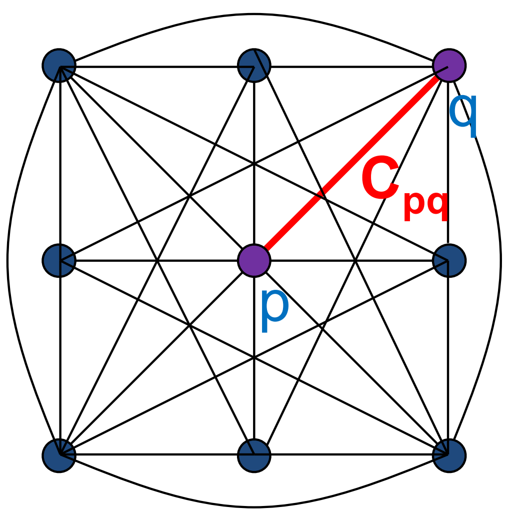
- Pros
  - Finds variable number of modes
  - Does not assume spherical clusters
  - Just a single parameter (window size)
  - Robust to outliers
- Cons
  - Output depends on window size
  - Computationally expensive
  - Does not scale well with dimension of feature space

Grauman

Graph-based algorithms

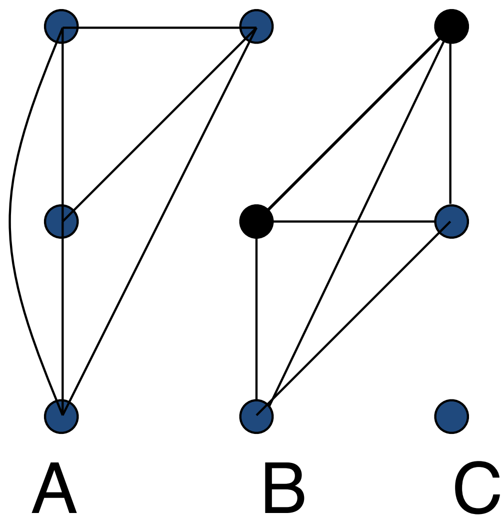
Segmentation Based on Graph Cuts

- Create weighted graph:
  - Nodes = pixels in image
  - Edge between each pair of nodes
  - Edge weight = similarity (intensity, color, texture, etc.)



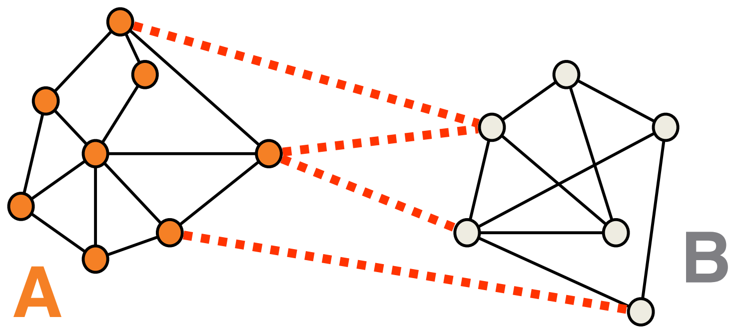
Seitz

Segmentation Based on Graph Cuts



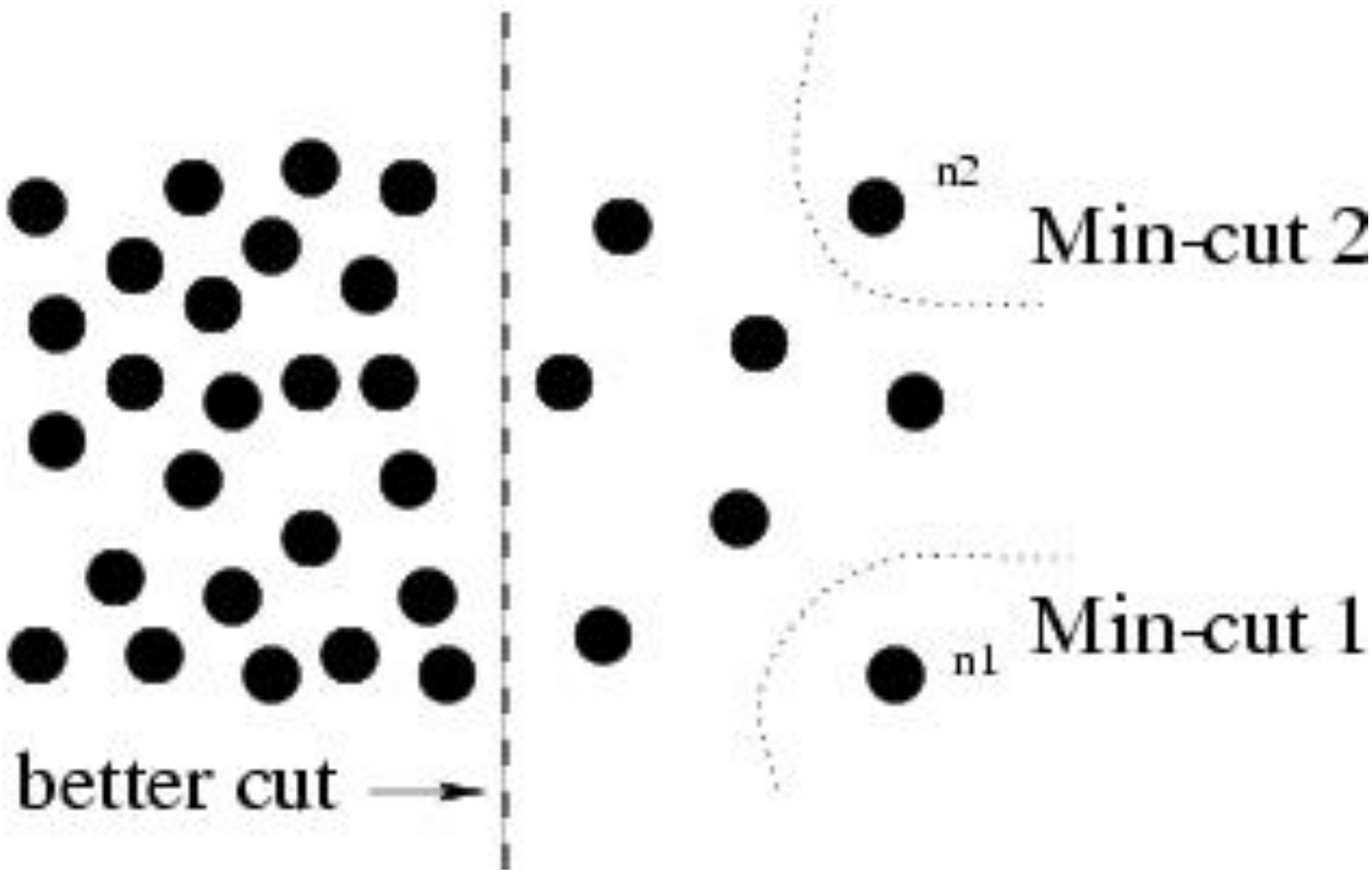
- Partition into disconnected segments
- Easiest to break links that have low cost (low similarity)
  - similar pixels should be in the same segments
  - dissimilar pixels should be in different segments

Cuts in a Graph



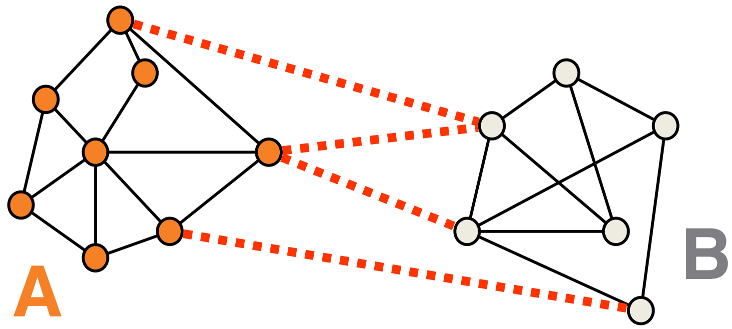
- Link Cut
  - set of links whose removal makes a graph disconnected
  - cost = sum of costs of all edges
- Min-cut
  - fast (polynomial-time) algorithm
  - gives segmentation

But Min Cut Is Not Always the Best Cut...



Seitz

Cuts in a Graph



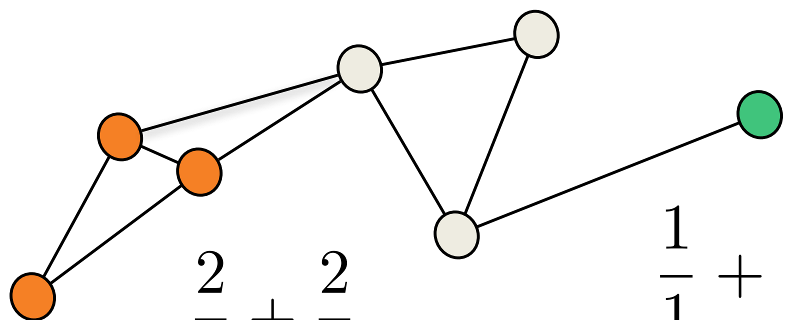
- Normalized Cut
  - removes penalty for large segments

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

  - volume(A) = sum of costs of all edges that touch A
  - no fast exact algorithms…

Cuts in a Graph

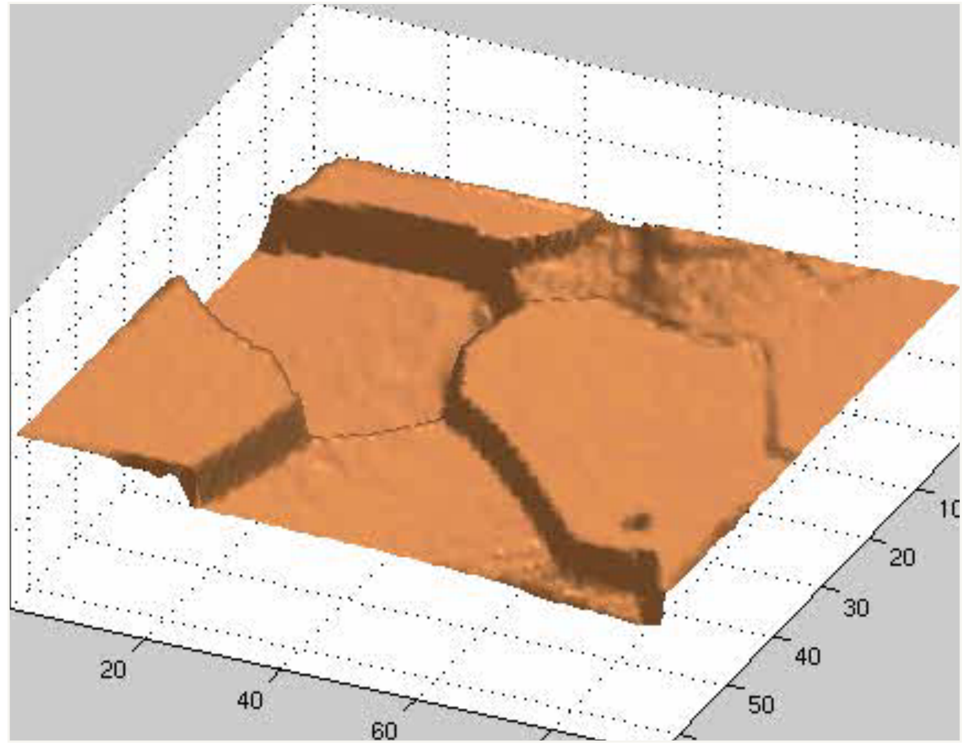
$$Ncut(A,B) = \frac{cut(A,B)}{volume(A)} + \frac{cut(A,B)}{volume(B)}$$



$$\frac{2}{5} + \frac{2}{6}$$

$$\frac{1}{1} + \frac{1}{9}$$

Interpretation as a Dynamical System



Treat the links as springs and shake the system

- elasticity proportional to cost
- vibration "modes" correspond to segments
  - can compute these by solving a generalized eigenvector problem
  - for more details, see
    - J. Shi and J. Malik, Normalized Cuts and Image Segmentation, CVPR, 1997

Seitz

Efficient graph-based segmentation

Efficient Graph-Based Image Segmentation
P. Felzenszwalb, D. Huttenlocher
International Journal of Computer Vision, Vol. 59, No. 2, September 2004

http://cs.brown.edu/~pff/segment/

Beautiful code
available

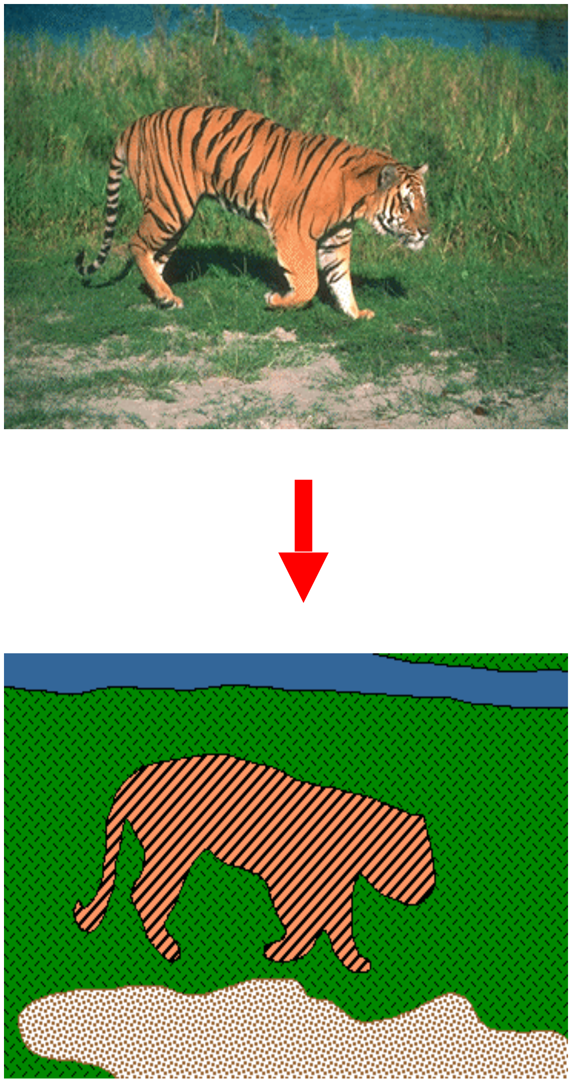
Example Results

Segmentation parameters: sigma = 0.5, K = 500, min = 50.

Segmentation parameters: sigma = 0.5, K = 1000, min = 100.

Boundary detection

Designing Grouping Features



Low-level cues

- Brightness similarity
- Color similarity
- Texture similarity

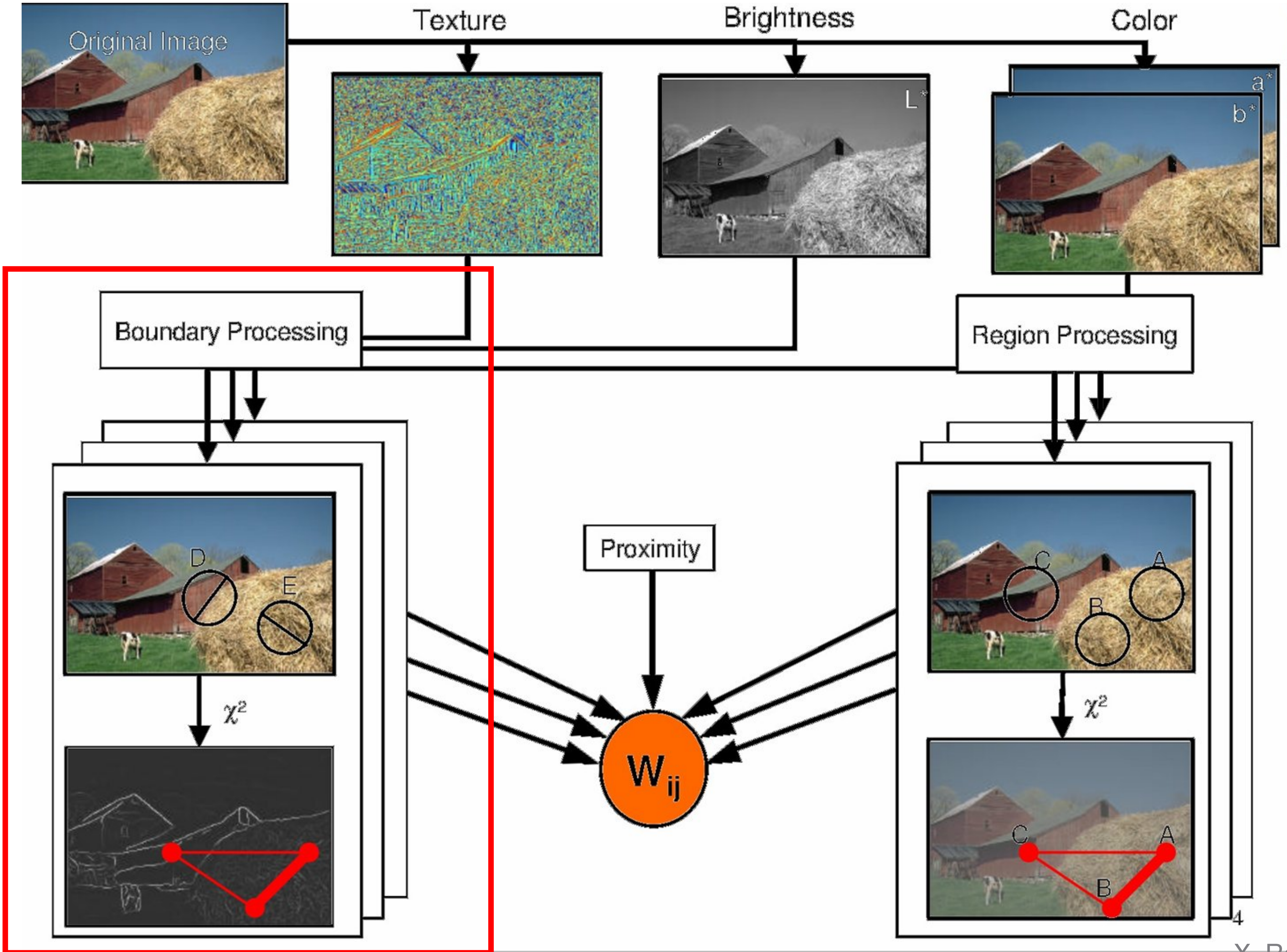
Mid-level cues

- Contour continuity
- Convexity
- Parallelism
- Symmetry

High-level cues

- Object knowledge
- Scene structure

X. Ren


Original Image
Texture
Brightness
Color
L*
a*
b*
Boundary Processing
Region Processing
D
E
χ^2
Proximity
W_{ij}
C
A
B
χ^2
C
A
B

X. Ren

Brightness and Color Contrast

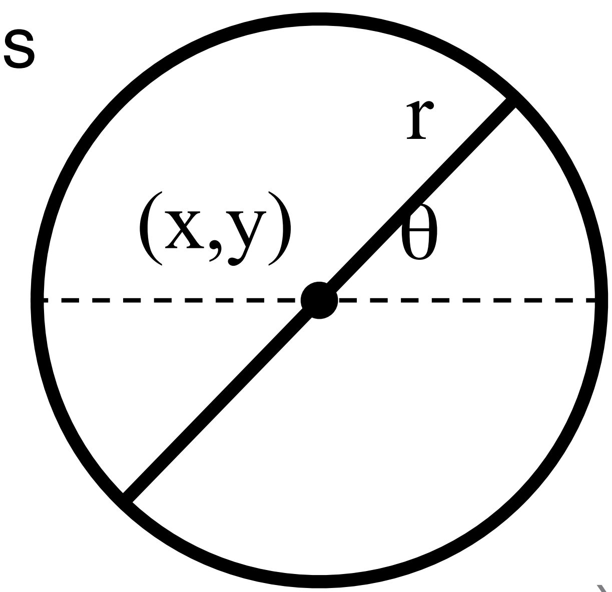
- Color (e.g., 1976 CIE L*a*b* colorspace)
- Brightness Gradient BG(x,y,r,θ)

  χ^2 difference in L* distribution

- Color Gradient CG(x,y,r,θ)

  χ^2 difference in a* and b* distributions

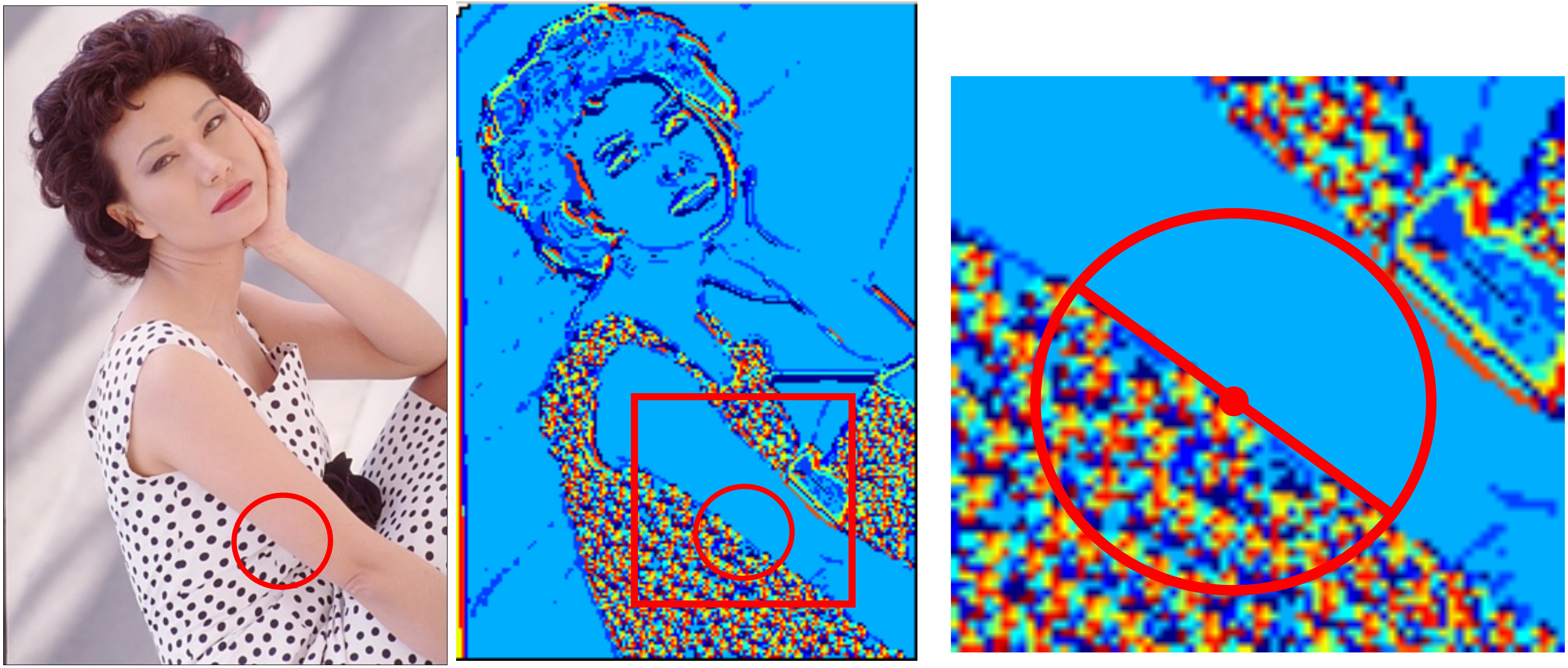
$$\chi^2(g,h) = \frac{1}{2}\sum_i \frac{(g_i - h_i)^2}{g_i + h_i}$$



X. Ren

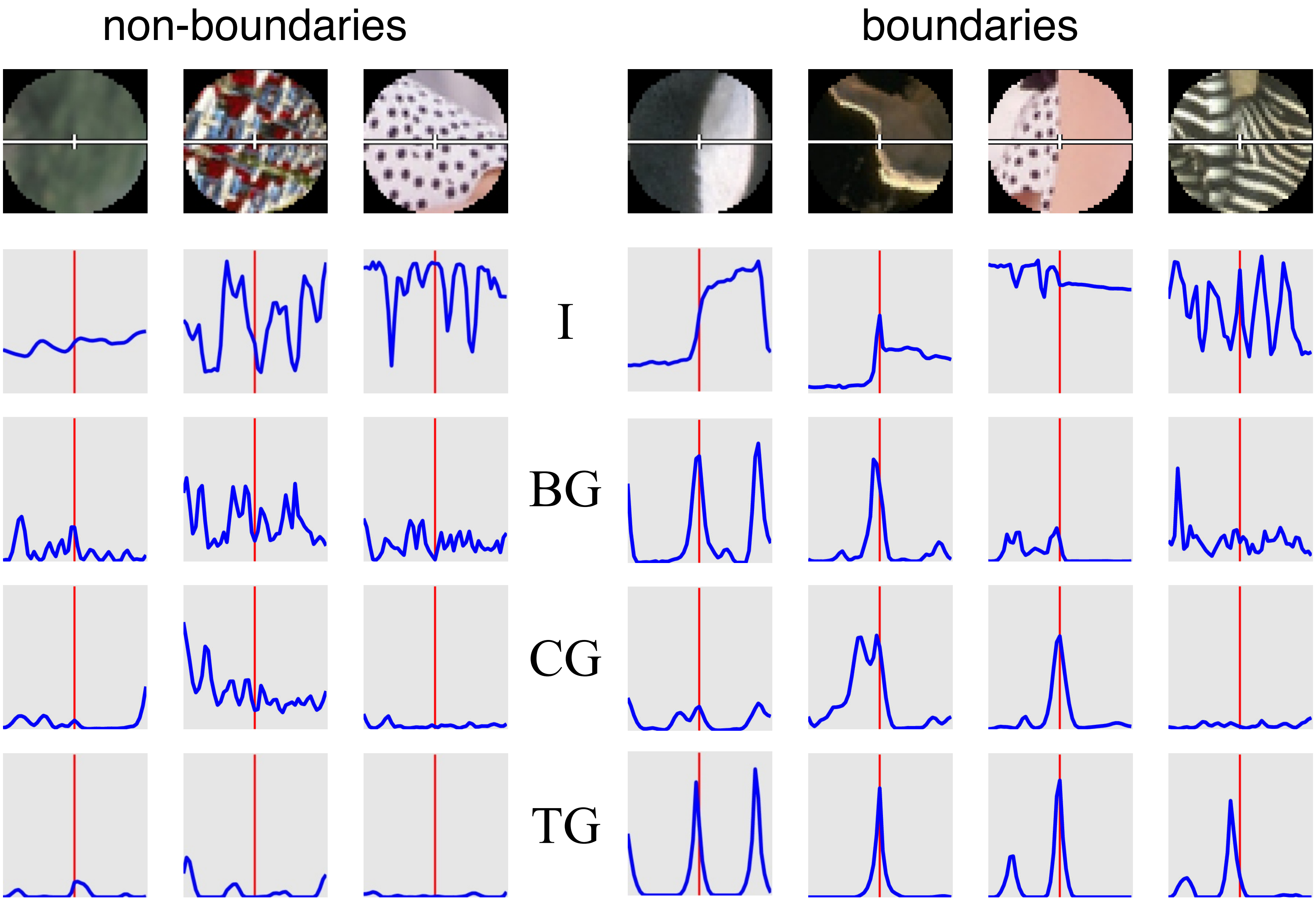
Texture Contrast

- Texture Gradient (next time)



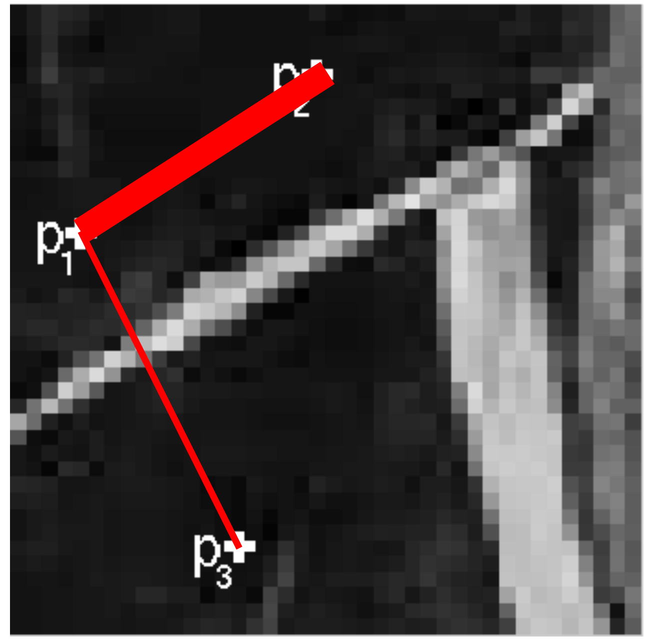
X. Ren

Boundary Classification



X. Ren

Affinity using Intervening Contour



W(p1,p2) >>W(p1,p3) as p1 and p2 are more likely to belong to the same region than are p1 and p3, which are separated by a strong boundary.

X. Ren

Combining Cues

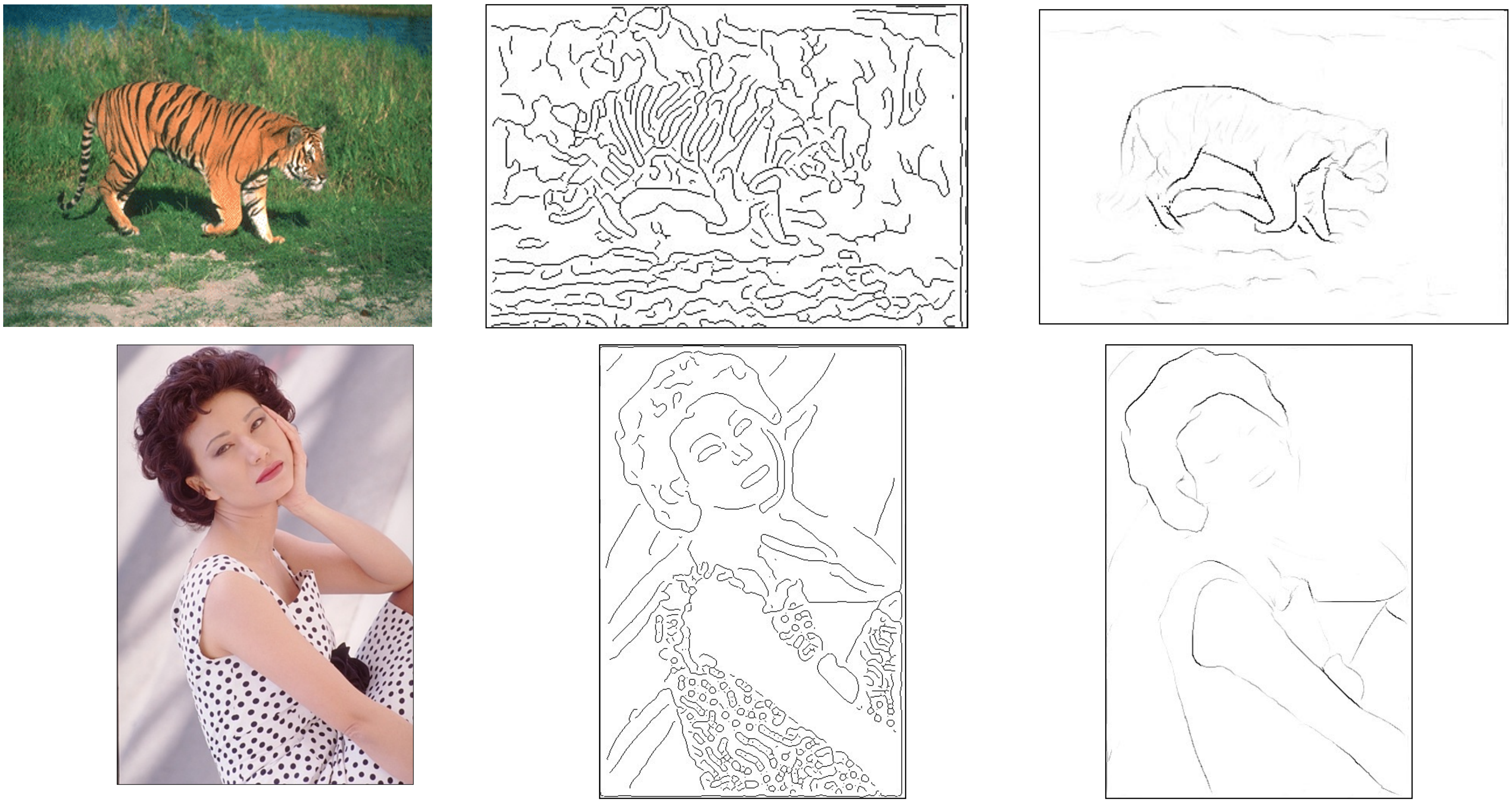
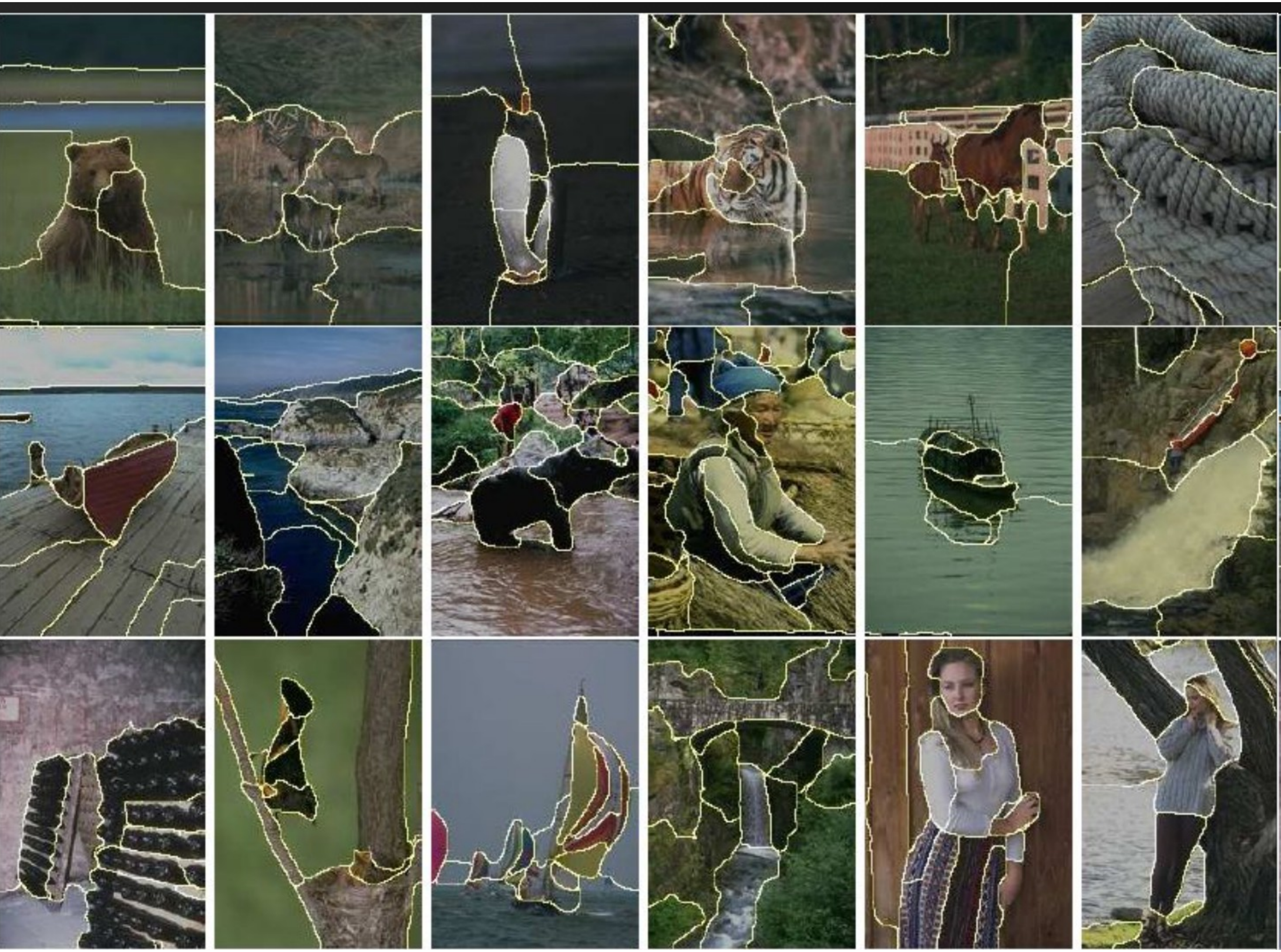


Image Canny Pb

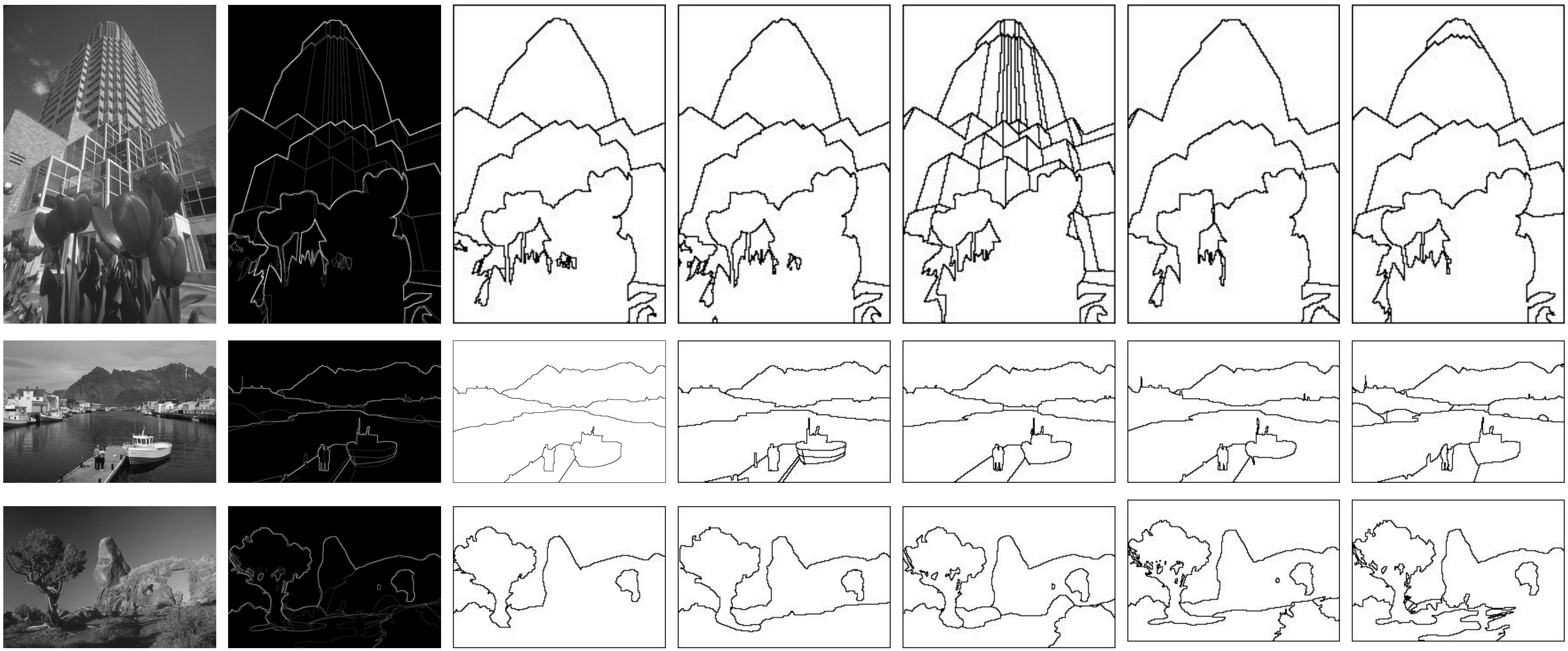
Martin, Fowlkes, Malik, *Learning to Detect Natural Image Boundaries Using Local Brightness, Color, and Texture Cues*, PAMI 2004



Evaluation

Option 1: human agreement

Berkeley segmentation dataset

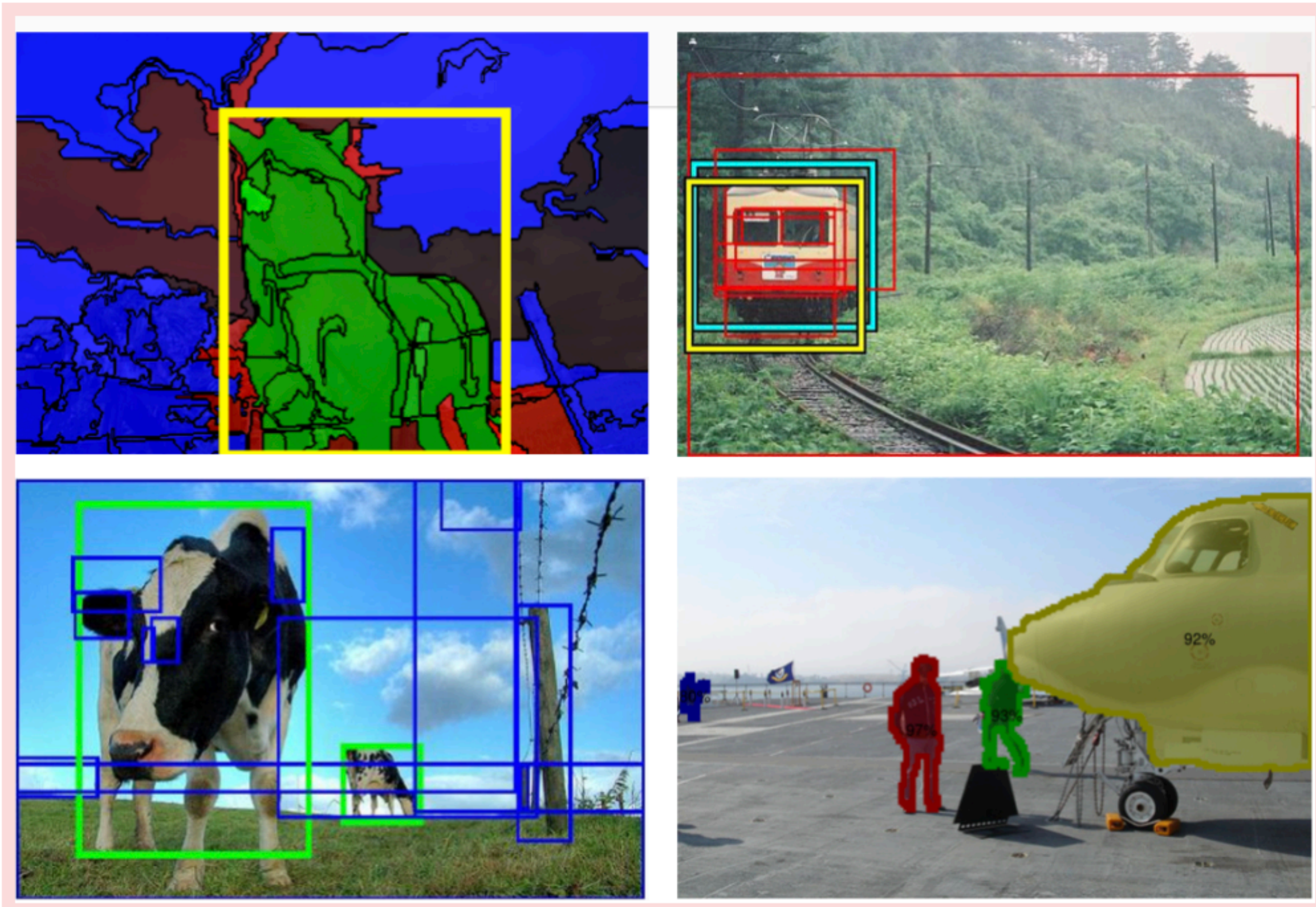


A Measure for Objective Evaluation of Image Segmentation Algorithms

R. Unnikrishnan C. Pantofaru M. Hebert

CVPR 2005

Option 2: usefulness for end goal



https://pdollar.wordpress.com/2013/12/22/generating-object-proposals/

However, worth noting:

Object-Proposal Evaluation Protocol is 'Gameable'
Neelima Chavali, Harsh Agrawal, Aroma Mahendru, Dhruv Batra
CVPR 2016

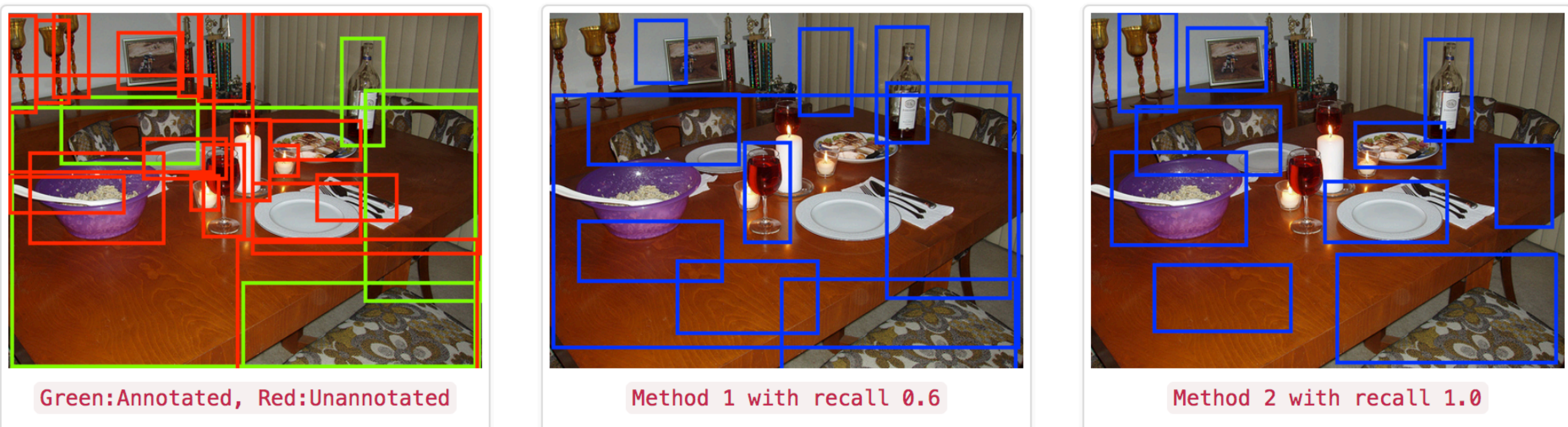


Figure 1: Method 1 visually seems to recall more categories such as plates, glasses that Method 2 missed. Despite that, the computed recall for Method 2 is higher because it recalled all instances of PASCAL categories that were present in the ground truth. Note that the number of proposals generated by both methods is equal in this figure.

https://filebox.ece.vt.edu/~aroma/web/object-proposals.html

Summary

- Segmentation:
  - Partitioning image into coherent regions
- Algorithms:
  - Divisive and hierarchical clustering
  - *k*-means clustering
  - Mean shift clustering
  - Graph cuts
- Applications
  - Image processing, object recognition, interactive image editing, etc.

Next week

http://viterbivoices.usc.edu/wp-content/uploads/2012/10/midterms-spring-08-300x300.jpeg