Segmentation & Texture Analysis

Sedat Ozer
Department of Computer Engineering
Bilkent University
sedat@cs.bilkent.edu.tr

Friday’s Class

- We will have a short class on this Friday too.
  (which will take about: less than an hour).

Final Projects

- Each group will prepare and submit:
  - **A Paper** written in CVPR format (unless stated otherwise). Need to be in LaTex format.
  - **A presentation** file (in PPT format) to be presented to the Instructor (about an hour long presentation will be given by all the group members. The missing group members will receive zero credit automatically).
  - **A Github page** for the project (include the code, a nice description, data or a link to data). If the links or files do not work on the Github page, students may not receive credit for the project. It is students' responsibility to make sure the github page contains clear definition.

Deadlines & Presentation Time

- Each group must submit all of their project materials (a conference paper, a ppt file and a link to the project's github page) **by 23:30, June 10, 2020.**
  - Include the Github link on the first page of your presentation and at the end of your conference paper.
  - You must submit all your documents by June 08 to able to present your ppt. You canNOT make any changes on your presentation, once you submit your files.

- Each group will make about **an hour long presentation** to the instructor. A time slot will be assigned to each group to make their final presentations. The time slots will be assigned as first-come-first-serve between **June 11 morning and June 15 evening (**based on the availability of the instructor). Groups should be prepared for questions during their presentations.

- **Time slots** are assigned as for one hour and fifteen minutes long starting at: **11:00, 12:15; 13:30; 14:45; 16:00; 17:15**.

Submission email

- In a single zip file include:
  - your paper files (as a PDF and all the Latex files in a folder)
  - Your PPT file that you will present (in .ppt or .pptx format)

and email it to: cs484cs555@gmail.com

with the subject: "CS484CS555_FinalProjectFiles_Group**<NAME>**" where you replace **<NAME>** with your group **name**.

Include all your group members name in your email too. You can use google drive if your file is too large.

Image Segmentation

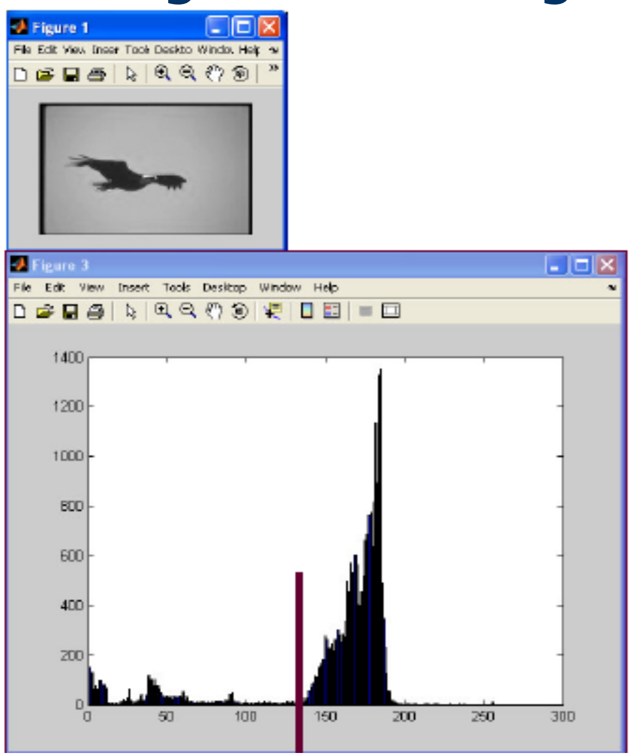
Goal of Segmentation is the operation of partitioning an image into sets of connected pixels (regions) … each of which has "reasonably" similar visual appearance.

Image segmentation

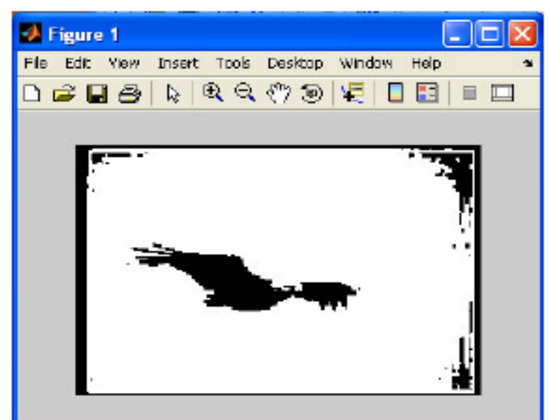
- So, all we have to do is to define and implement the similarity predicate.
  - But, what do we want to be similar in each region?
  - Is there any property that will cause the regions to be meaningful objects?
- Example approaches:
  - Histogram-based
  - Clustering-based
  - Region growing
  - Split-and-merge
  - Morphological
  - Graph-based
  - Superpixels

Histogram-based segmentation

- How many “orange” pixels are in this image?
- This type of question can be answered by looking at the histogram.



imshow(B > 140)



Clustering

An unsupervised learning technique

– Requires data, but no labels

– Detect patterns e.g. in

• Group emails or search results • Customer shopping patterns

• Regions of images

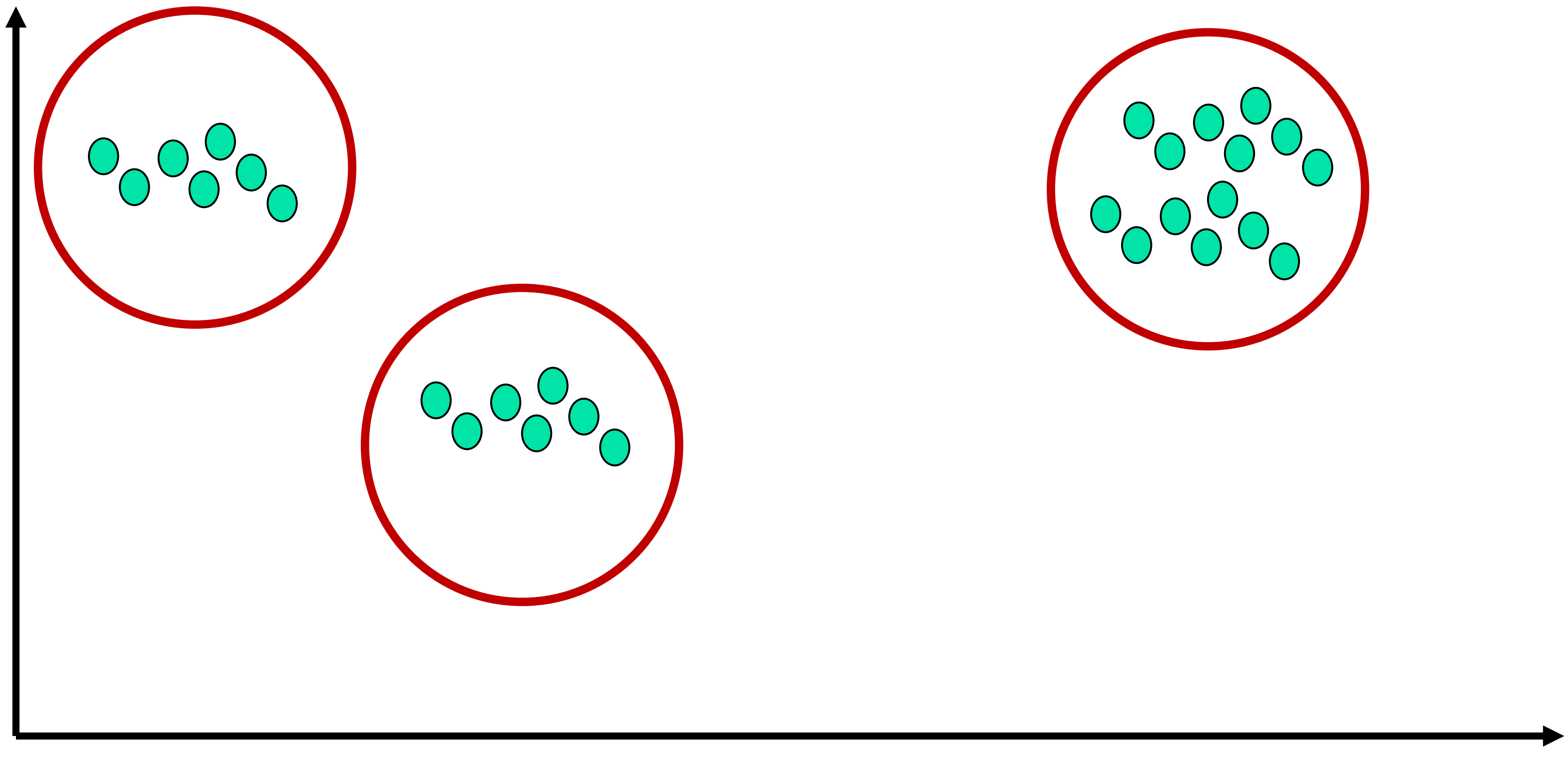
– Useful when don’t know what you’re looking for

– But: can get confusing or useless or not meaningful

Clustering

Basic idea: cluster (group) together similar instances

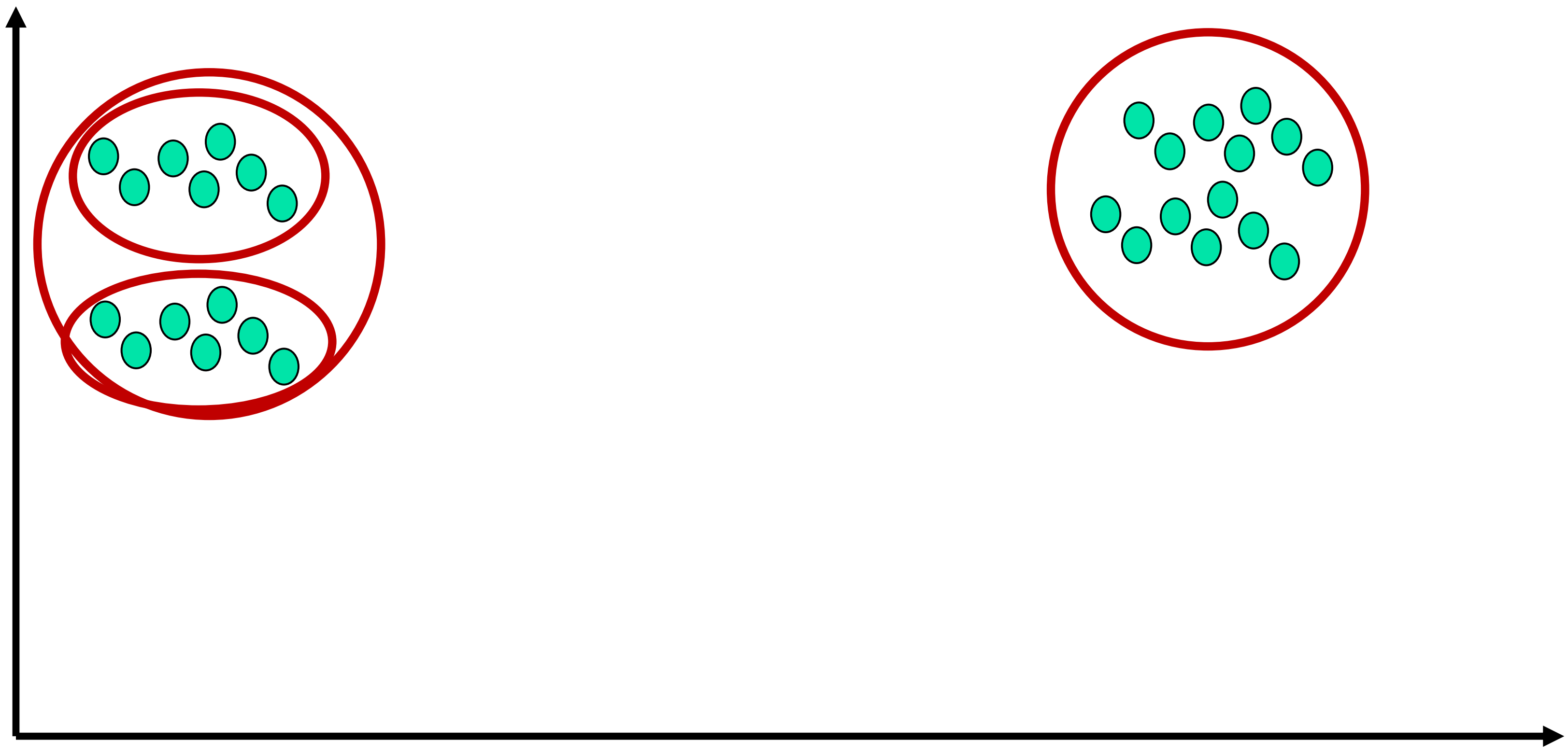
Example: 2D points



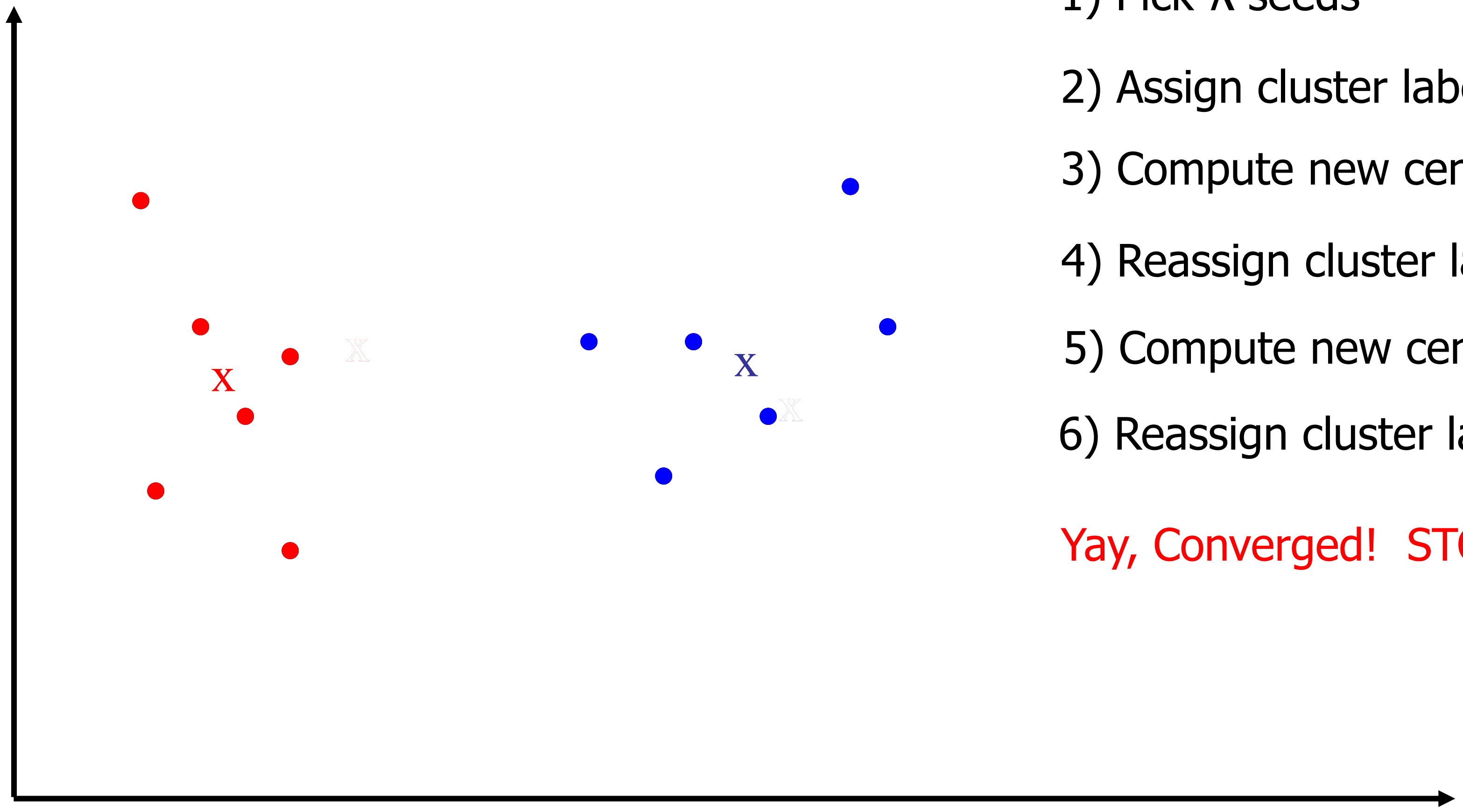
Clustering

Basic idea: cluster (group) together similar instances

Example: 2D points



Illustrative Example for *K*-Means (with *K*=2)



1) Pick *K* seeds

2) Assign cluster labels

3) Compute new centroids

4) Reassign cluster labels

5) Compute new centroids

6) Reassign cluster labels

Yay, Converged! STOP!!

Clustering Examples

Image segmentation

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

[Slide from James Hayes]

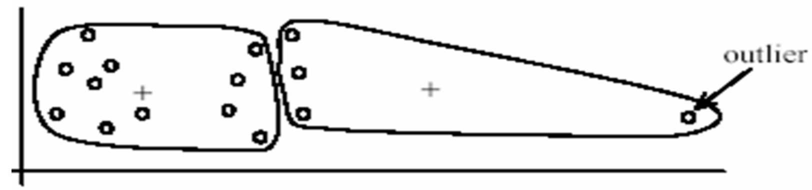
Example: K-Means for Segmentation

K=2

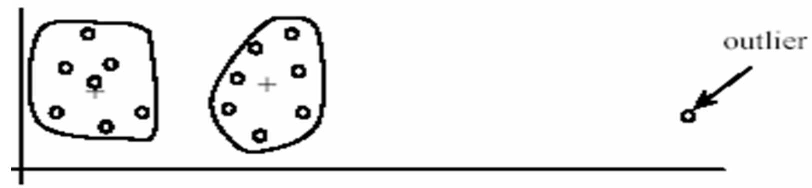
Original

Clustering-based segmentation

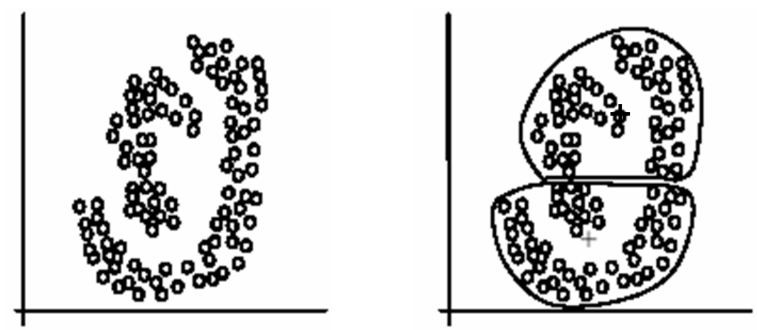
- Pros (of k-means):
  - Simple, fast to compute
  - Converges to local minimum of within-cluster squared error
- Cons (of k-means):
  - Setting K?
  - Sensitive to initial centers
  - Sensitive to outliers
  - Detects spherical clusters



(A): Undesirable clusters



(B): Ideal clusters



(A): Two natural clusters (B): *k*-means clusters

Adapted from Kristen Grauman

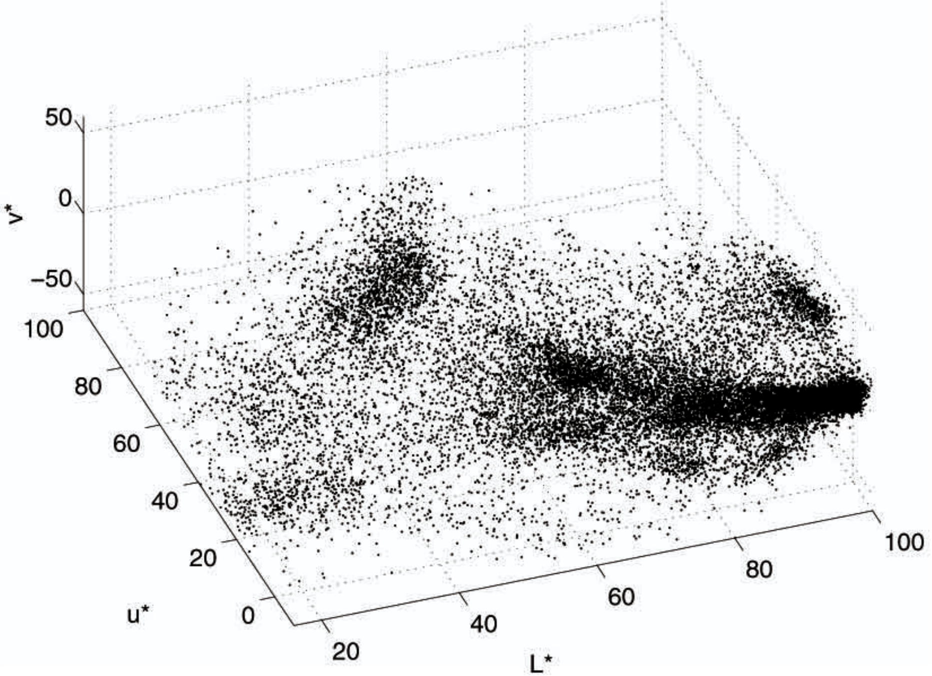
Clustering-based segmentation

- K-means variants:
  - Different ways to initialize the means.
  - Different stopping criteria.
  - Dynamic methods for determining the right number of clusters (K) for a given image.
- Problem: histogram-based and clustering-based segmentation using color/texture/etc can produce messy/noisy regions. (Why?)
- How can these be fixed?

Mean shift algorithm

- The mean shift algorithm seeks *modes* or local maxima of density in the feature space.

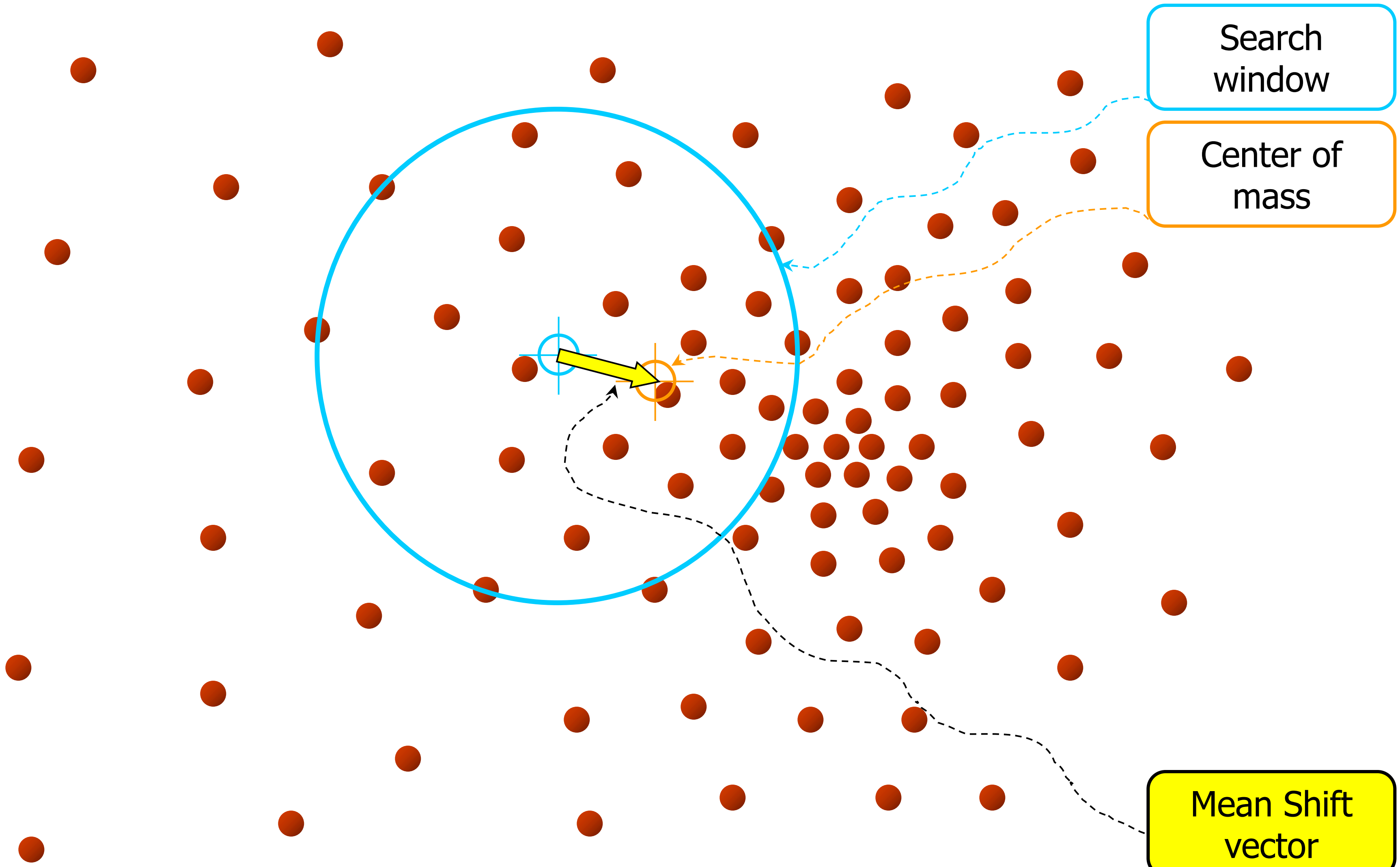
Image



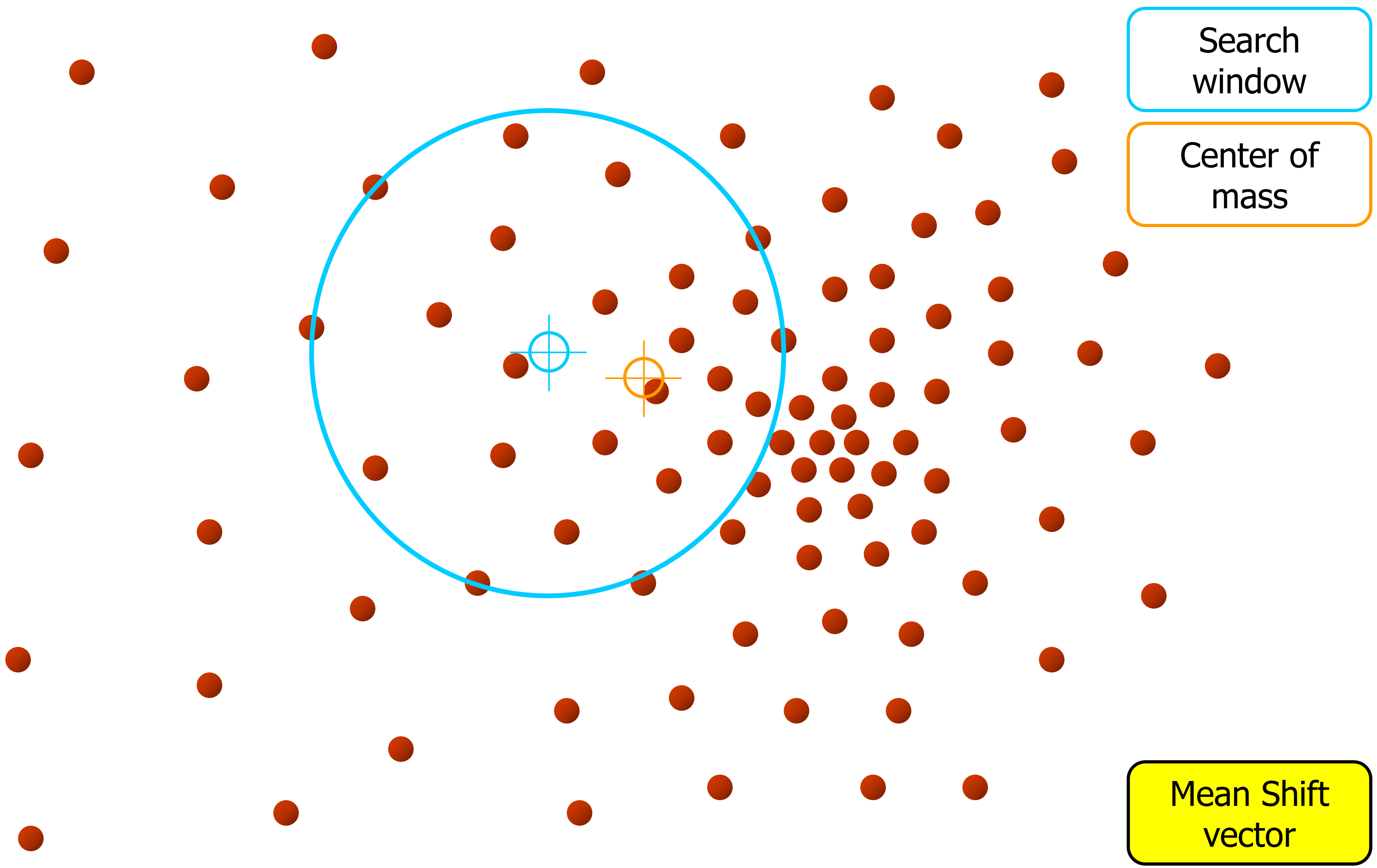
Feature space
(L*u*v* color values)

Adapted from Kristen Grauman

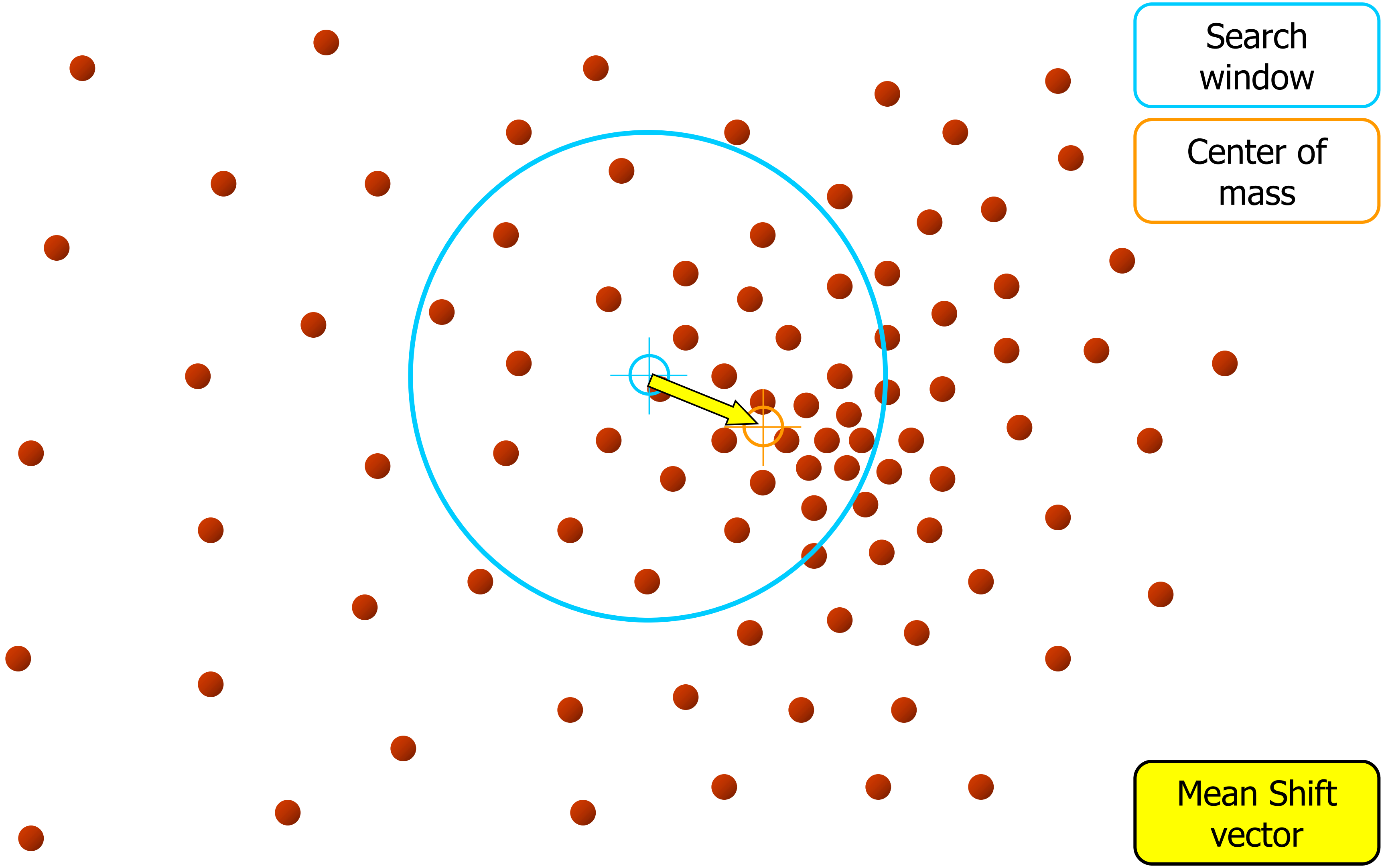
Mean shift algorithm



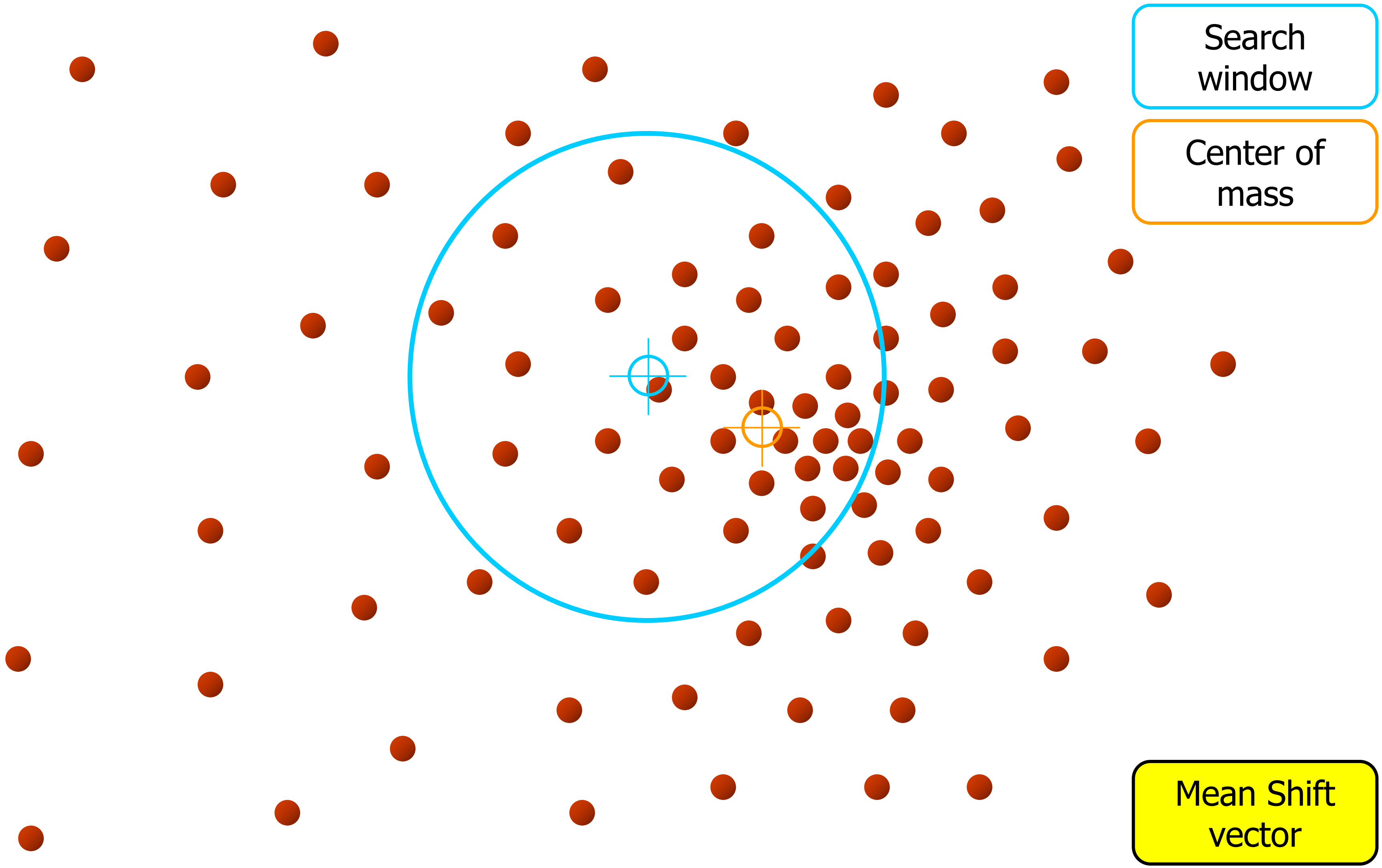
Mean shift algorithm



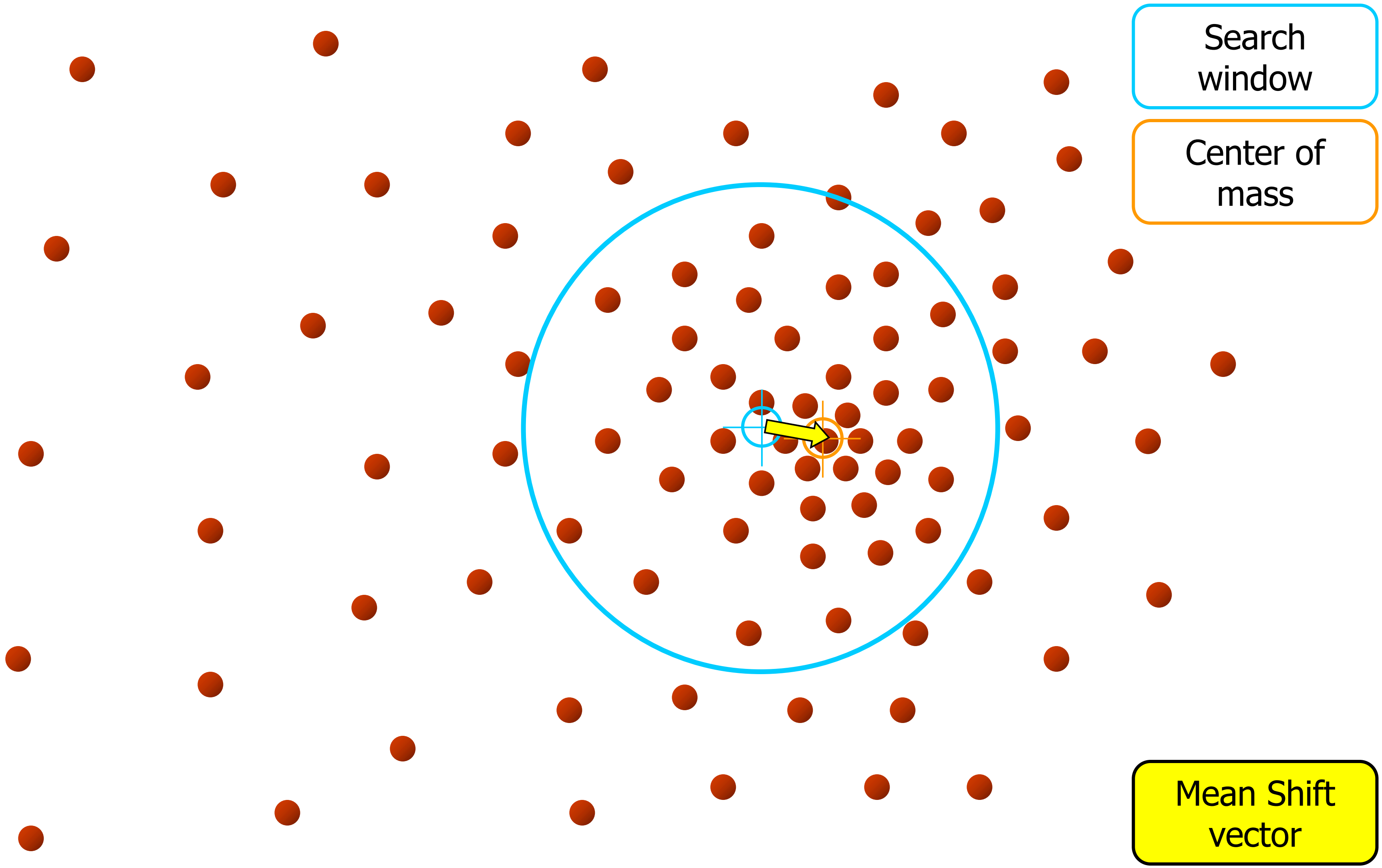
Mean shift algorithm



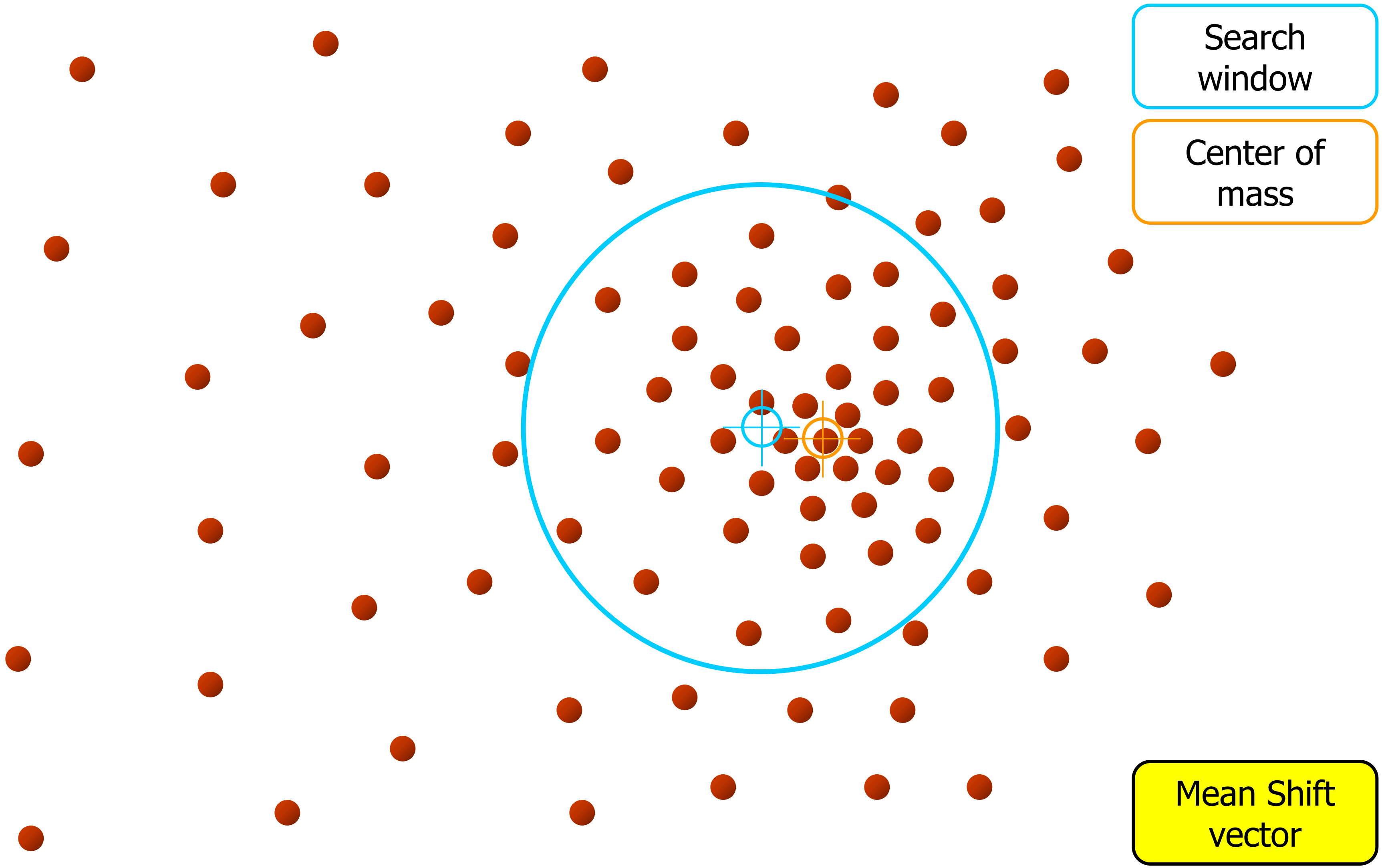
Mean shift algorithm



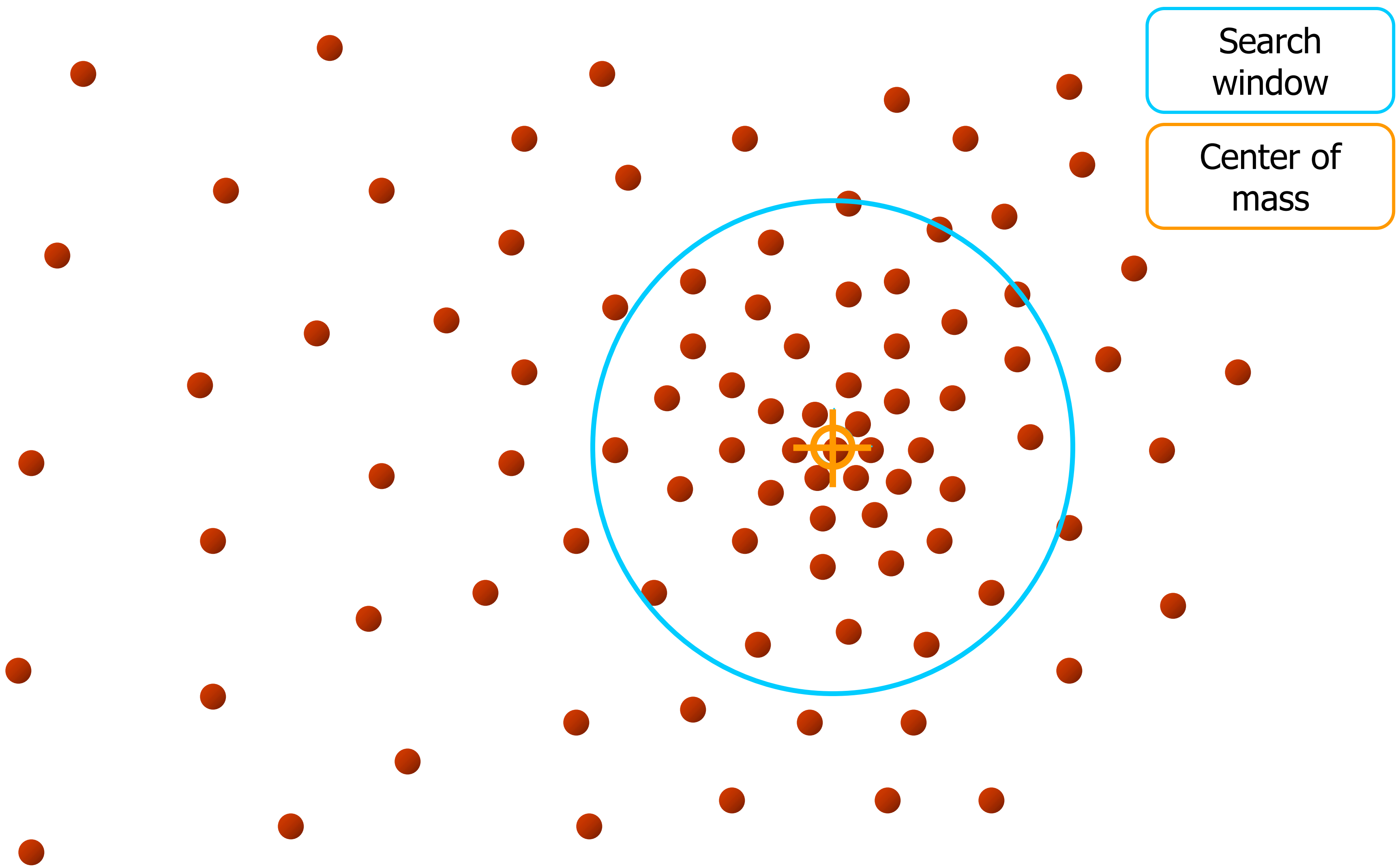
Mean shift algorithm



Mean shift algorithm

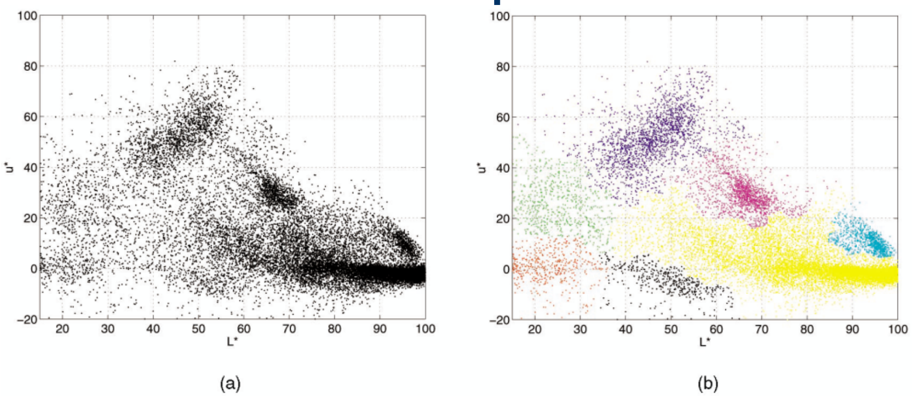


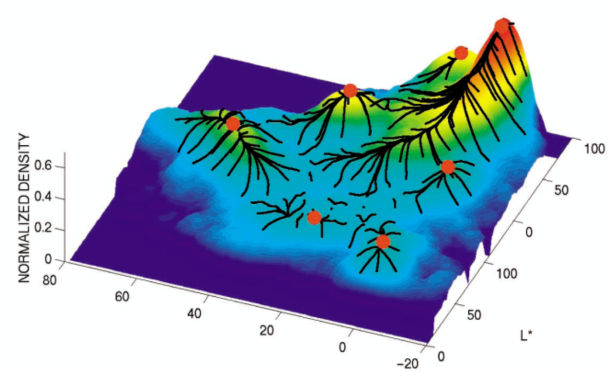
Mean shift algorithm



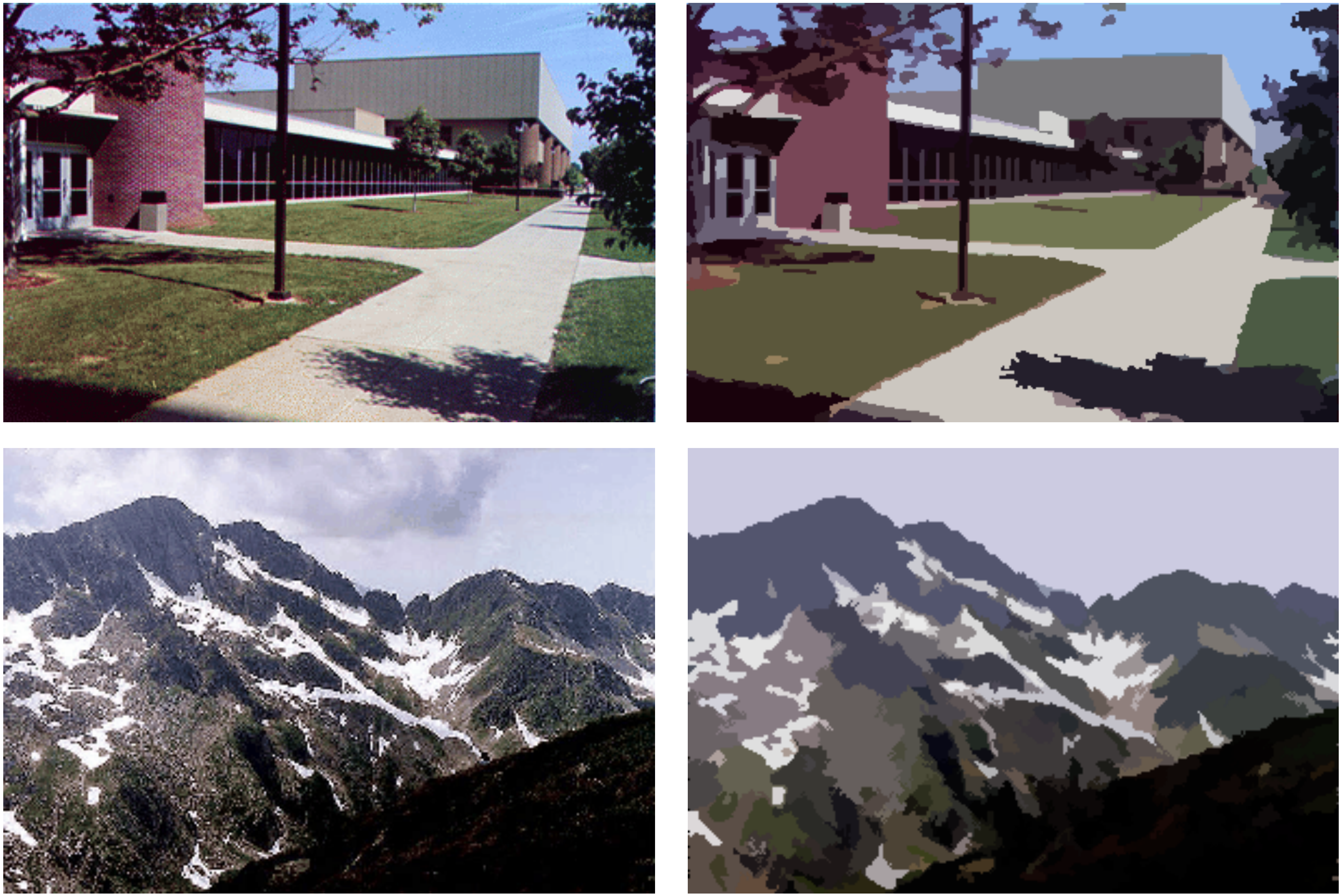
Mean shift clustering/segmentation

- Find features (color, gradients, texture, etc)
- 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 segmentation



Mean shift segmentation

- Pros:
  - Does not assume shape on clusters
  - One parameter choice (window size)
  - Generic technique
  - Find multiple modes
- Cons:
  - Selection of window size
  - Does not scale well with dimension of feature space

Adapted from Kristen Grauman

Region growing

- Region growing starts with one pixel (seed) of a potential region and tries to grow it by adding adjacent pixels till the pixels being compared are too dissimilar.
- We need to define a measure of similarity between a pixel and a set of pixels as well as a rule that makes a decision for growing.
- Usually a statistical test is used to decide which pixels can be added to a region.
  - Region is a population with similar statistics.
  - Use statistical test to see if neighbor on border fits into the region population.

Region growing

- Example: Let R be the N pixel region obtained so far and p be a neighboring pixel with gray tone y.
- Define the mean X and scatter S^2 (sample variance) by

$$\overline{X} = \frac{1}{N} \sum_{(r,c) \in R} I(r,c) \qquad S^2 = \frac{1}{N} \sum_{(r,c) \in R} \left(I(r,c) - \overline{X} \right)^2$$

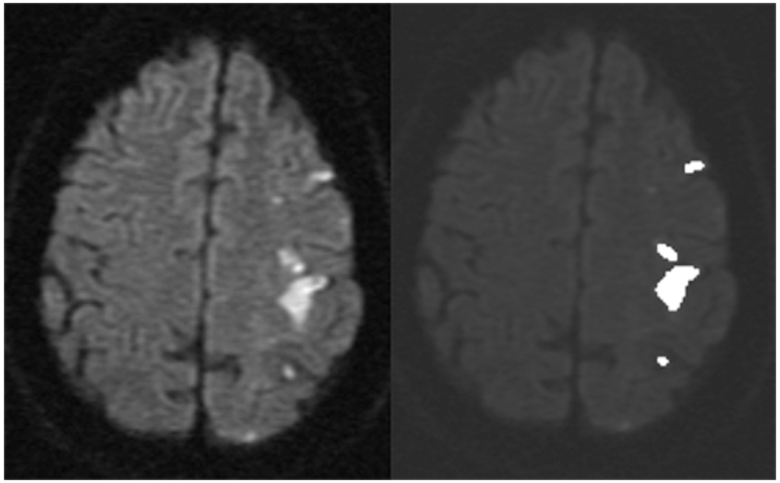
- The T statistic is defined by

$$T = \left(\frac{(N-1)N}{(N+1)} (p - \overline{X})^2 / S^2 \right)^{1/2}$$

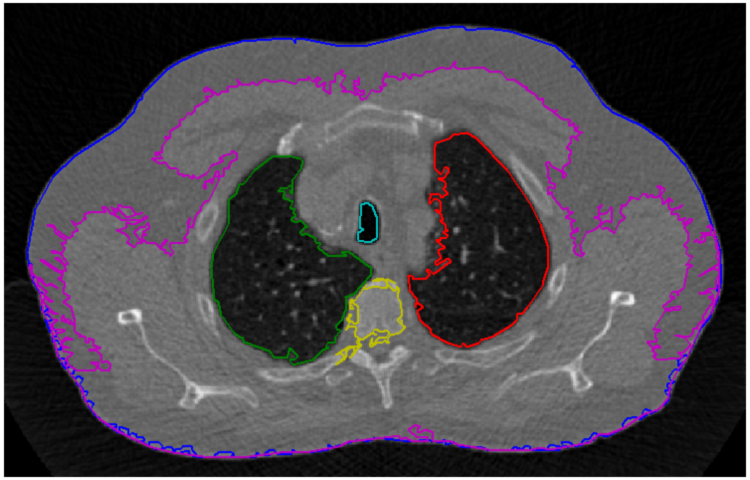
Region growing

- It has a T_{N-1} distribution if all the pixels in R and the test pixel p are independent and identically distributed Gaussians (i.i.d. assumption).
- For the T distribution, compute the probability $Pr(T \leq t)$ for a given degrees of freedom and a confidence level. Pick a suitable threshold t.
- If $T \leq t$ for desired confidence level, add p to region R and update the mean and scatter.
- If T is too high, the value p is not likely to have arisen from the population of pixels in R. Start a new region.

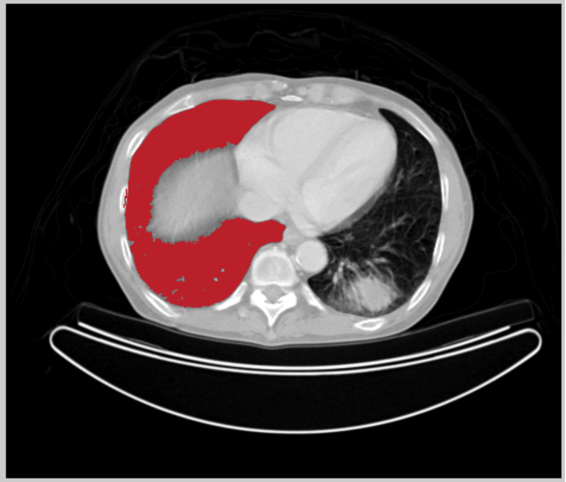
Region growing



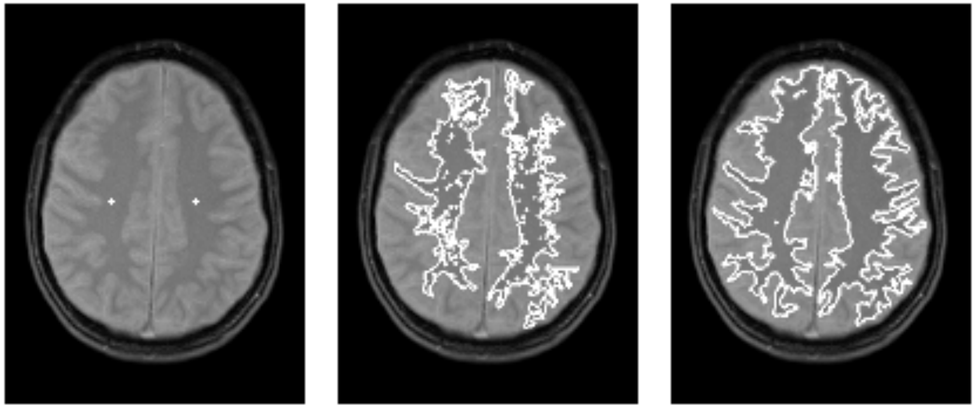
http://www.bigr.nl/website/static/research/regrow.html



http://www.mathworks.com/matlabcentral/fileexchange/32532-region-growing-2d3d-grayscale



http://www.mathworks.com/matlabcentral/fileexchange/19084-region-growing

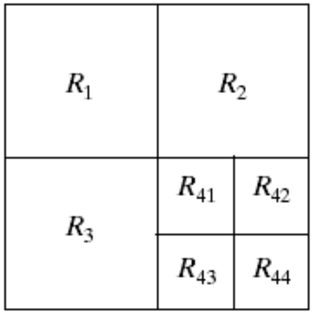


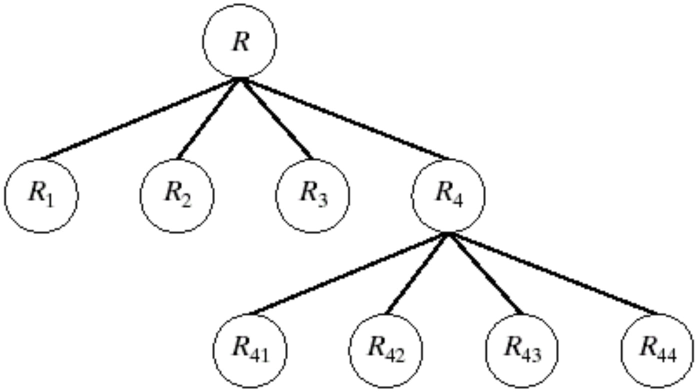
http://www.creatis.insa-lyon.fr/~grenier/?p=172

Split-and-merge

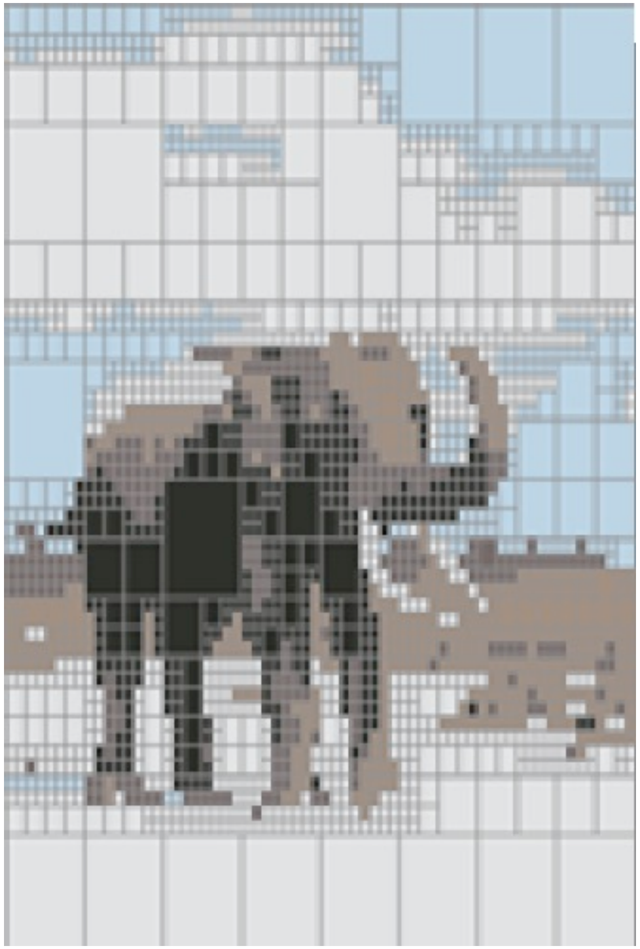
1. Start with the whole image.
2. If the variance is too high, break into quadrants.
3. Merge any adjacent regions that are similar enough.
4. Repeat steps 2 and 3, iteratively until no more splitting or merging occur.

→ Idea: good
Results: blocky



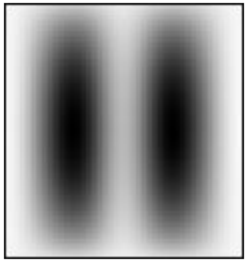


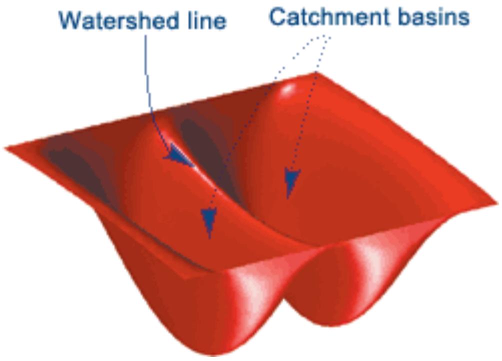
Split-and-merge



Watershed segmentation

- The image can be interpreted as a topographic surface, with both valleys and mountains.
- Three types of points can be considered:
  - Points belonging to a regional minimum.
  - Points at which a drop of water, if placed at the location of any of those points, would fall to a single minimum. → catchment basins
  - Points at which water would be equally likely to fall to more than one such minimum. → watershed lines





Watershed segmentation

- Assume that there is a hole in each minima and the surface is immersed into a lake.
- The water will enter through the holes at the minima and flood the surface.
- To avoid the water coming from two different minima to meet, a dam is build whenever there would be a merge of the water.
- Finally, the only thing visible of the surface would be the dams. These dam walls are called the watershed lines.

Watershed segmentation

a b
c d

FIGURE 10.44
(a) Original image.
(b) Topographic view. (c)–(d) Two stages of flooding.

Watershed segmentation

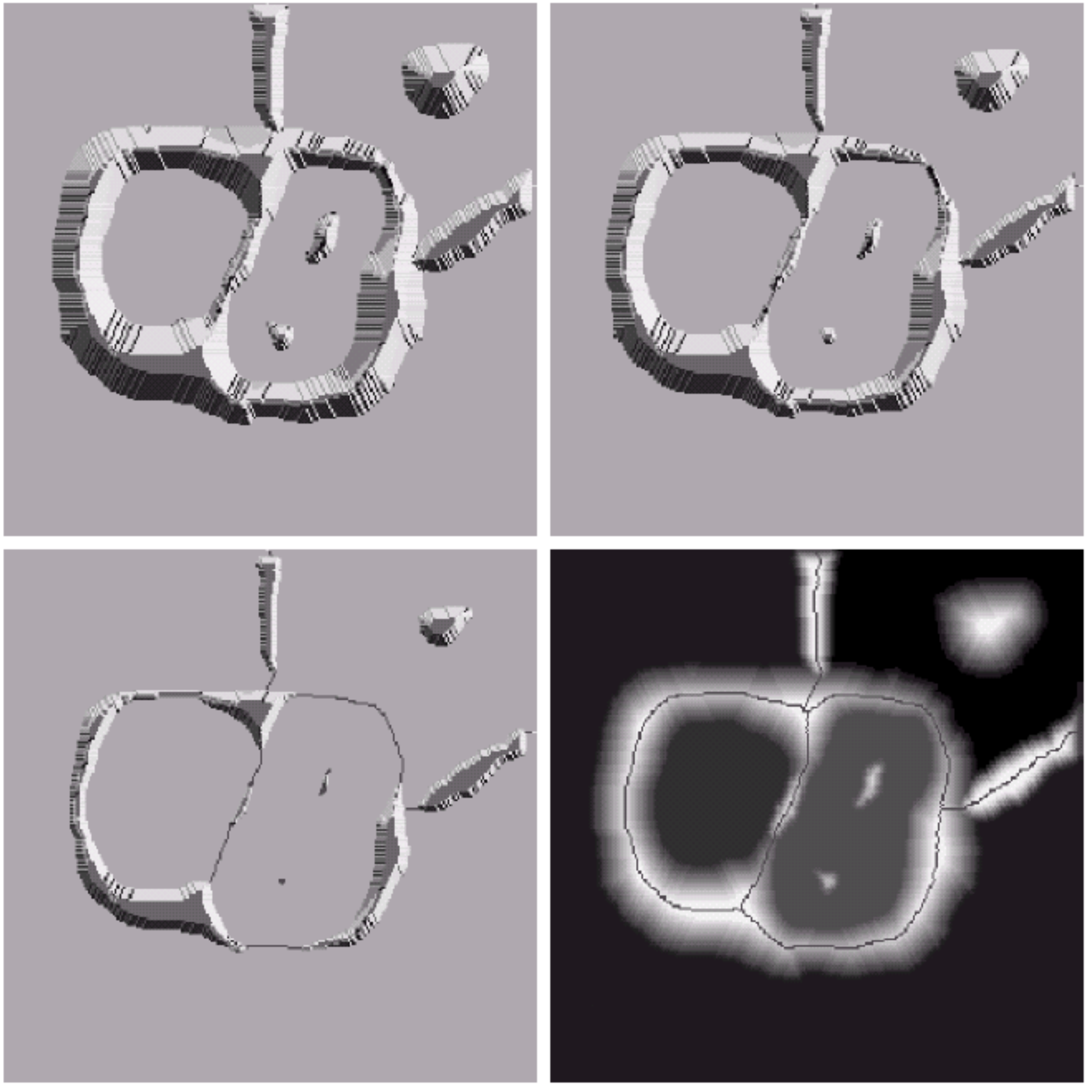
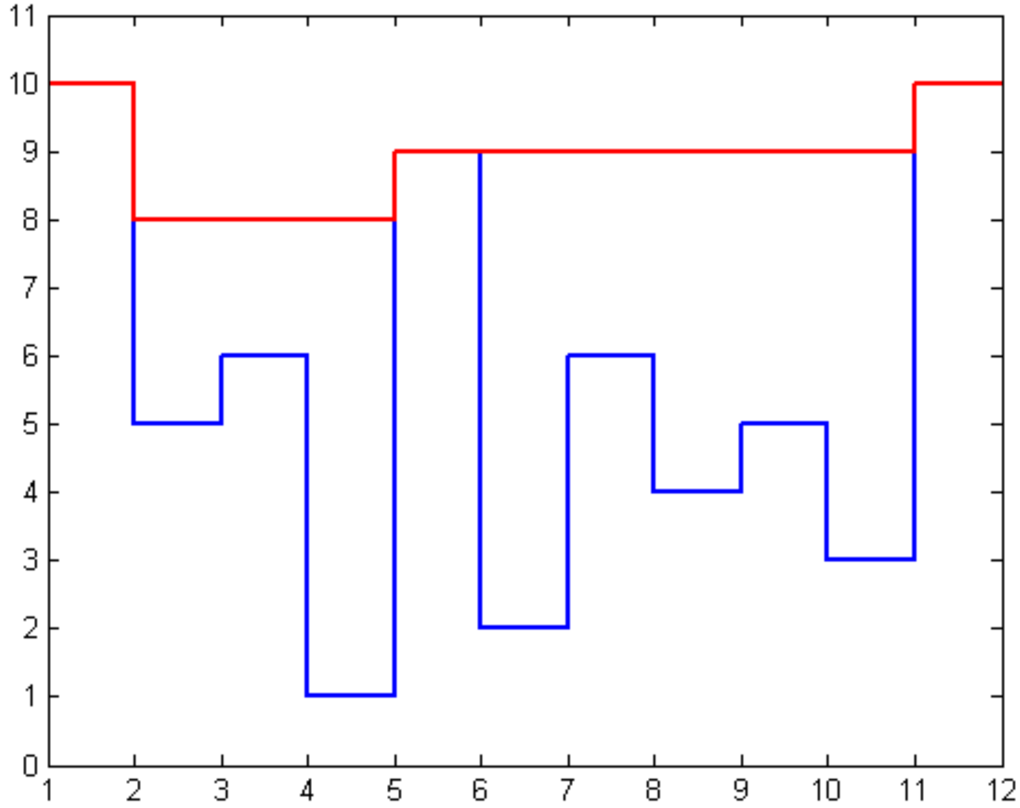


FIGURE 10.44 *(Continued)* (e) Result of further flooding. (f) Beginning of merging of water from two catchment basins (a short dam was built between them). (g) Longer dams. (h) Final watershed (segmentation) lines. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

Watershed segmentation

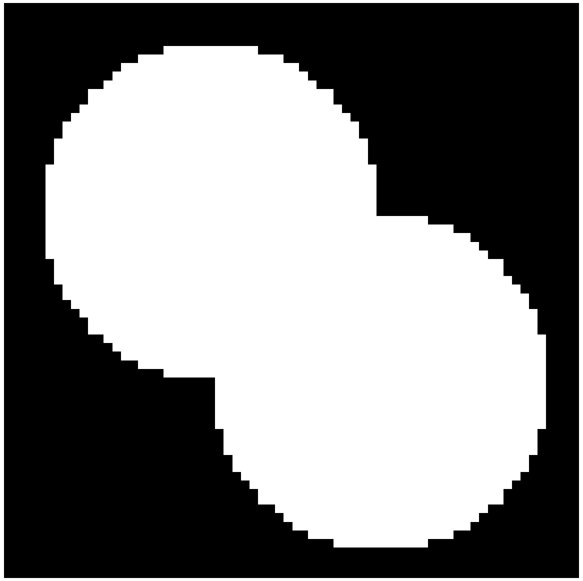
- A multi-scale segmentation can be obtained by iteratively smoothing the topographic surface.



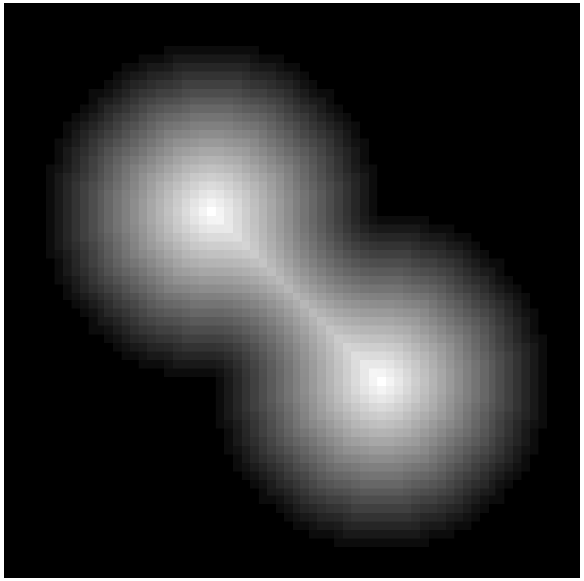
Watershed segmentation

- The key behind using the watershed transform for segmentation is this: change your image into another image whose catchment basins are the objects you want to identify.
- Examples:
  - Distance transform can be used with binary images where the catchment basins correspond to the foreground components of interest.
  - Gradient can be used with grayscale images where the catchment basins should theoretically correspond to the homogeneous grey level regions of the image.

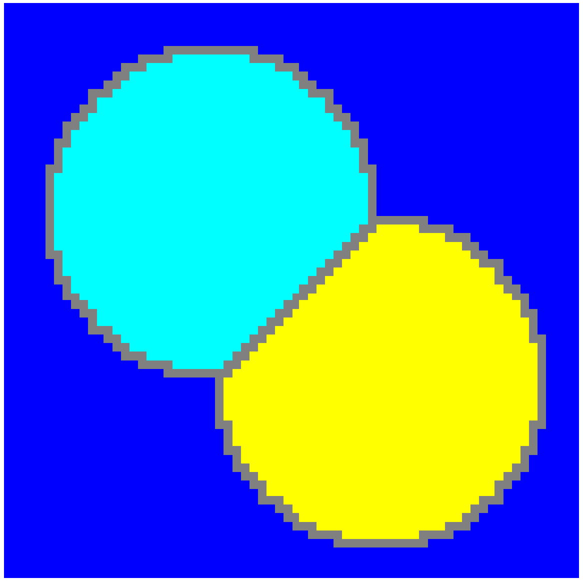
Watershed segmentation



Binary image.

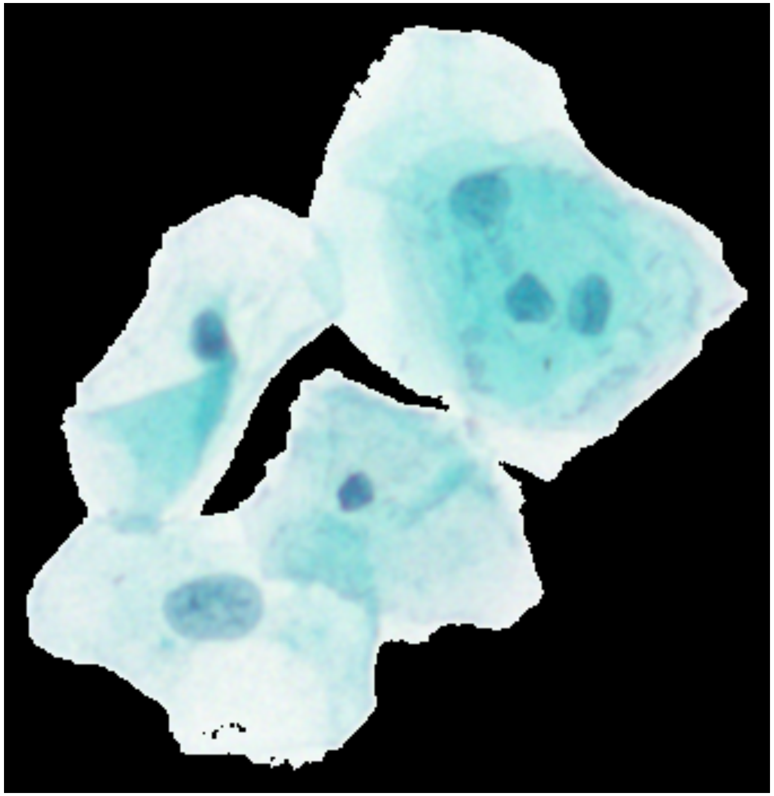


Distance transform of the complement of the binary image.

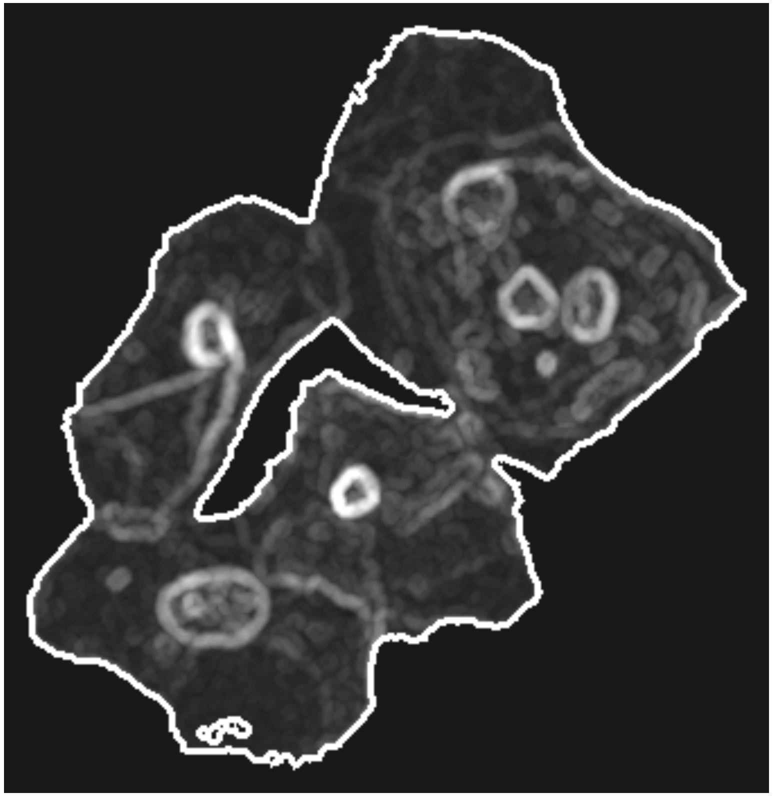


Watershed transform after complementing the distance transform, and forcing pixels that do not belong to the objects to be at –Inf.

Watershed segmentation

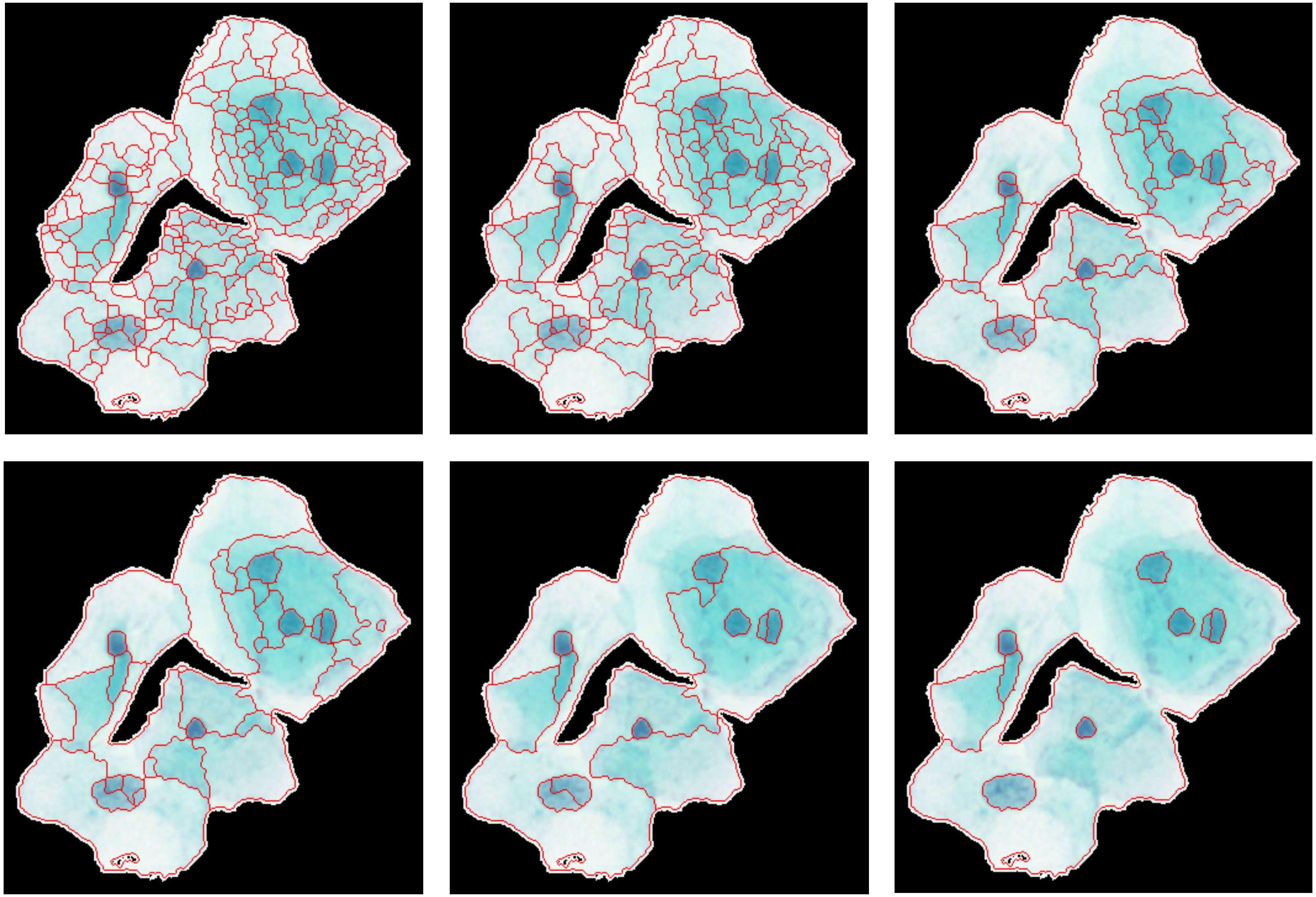


A cell image.



Gradient of the cell image.

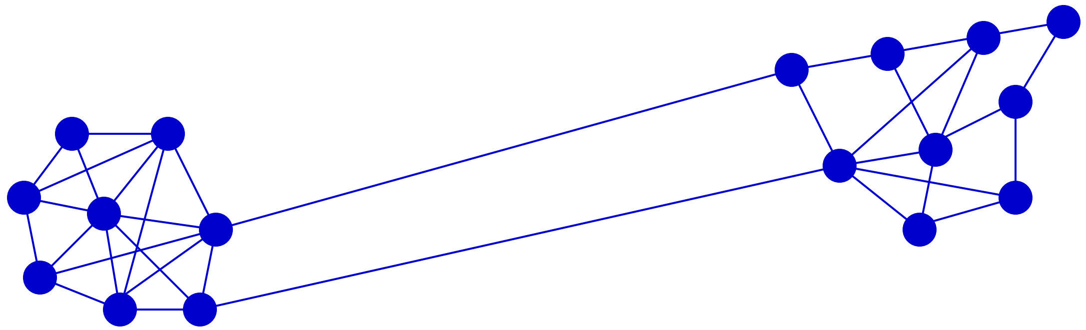
Watershed segmentation



Multi-scale watershed segmentation of the cell image.

Graph-based segmentation

- An image is represented by a graph whose nodes are pixels or small groups of pixels.
- The goal is to partition the nodes into disjoint sets so that the similarity within each set is high and across different sets is low.

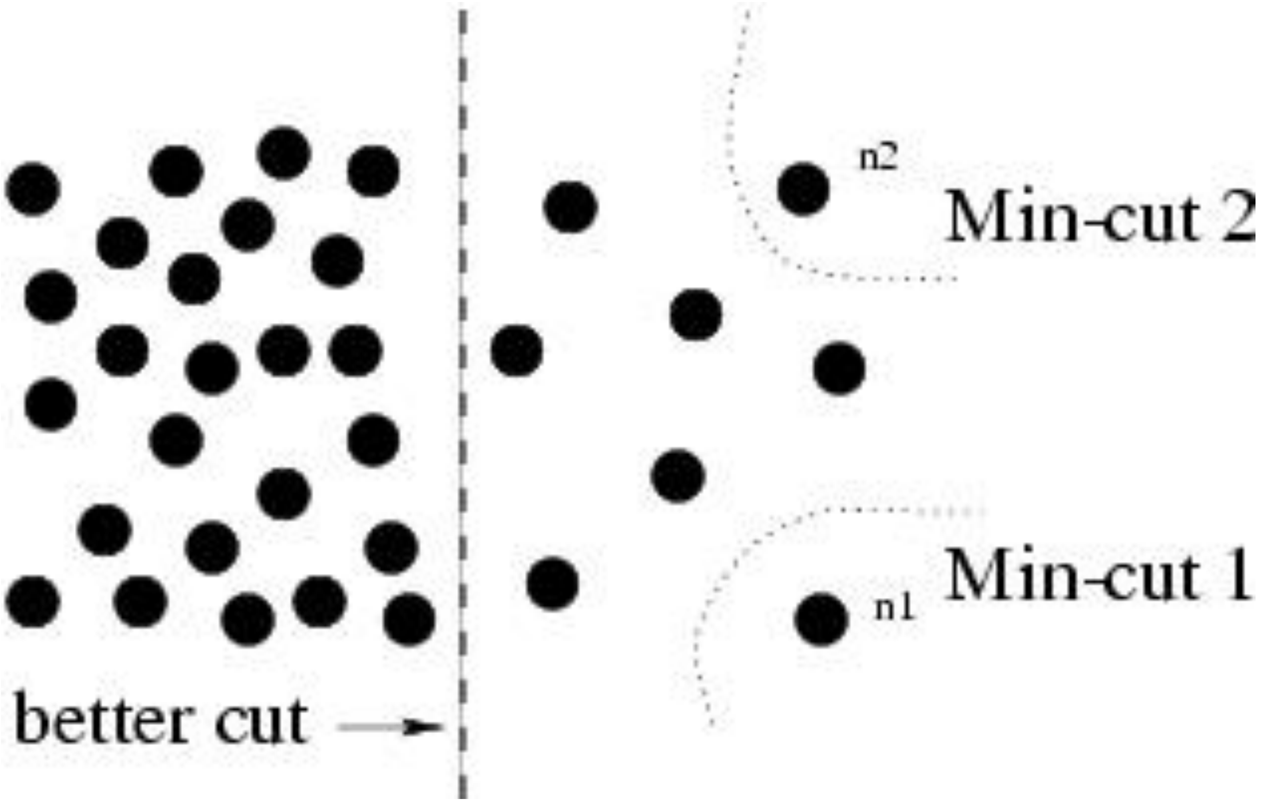


Graph-based segmentation

- Let $G = (V,E)$ be a graph. Each edge (u,v) has a weight $w(u,v)$ that represents the similarity between u and v.
- Graph G can be broken into 2 disjoint graphs with node sets A and B by removing edges that connect these sets.
- Let $cut(A,B) = \sum_{u \in A,\, v \in B} w(u,v)$.
- One way to segment G is to find the minimum cut.

Graph-based segmentation

- Problem with minimum cut: weight of cut proportional to number of edges in the cut; tends to produce small, isolated components.



Graph-based segmentation

- Shi and Malik proposed the normalized cut.

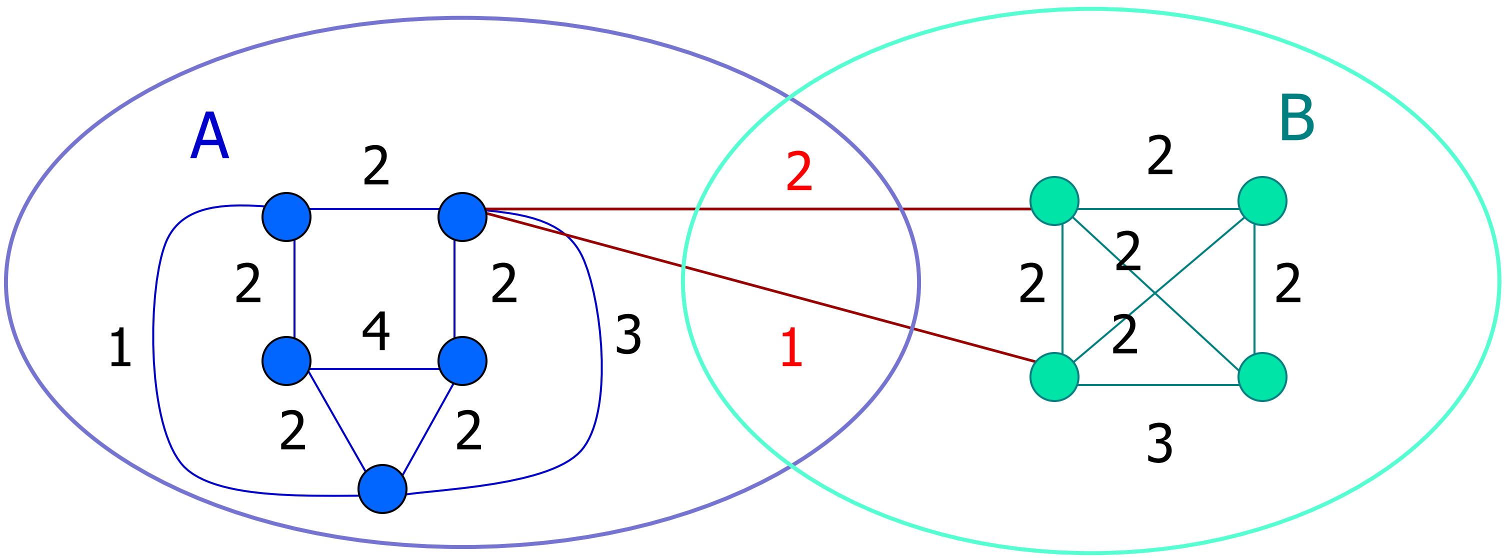
$$\text{Ncut}(A,B) = \frac{\text{cut}(A,B)}{\text{assoc}(A,V)} + \frac{\text{cut}(A,B)}{\text{assoc}(B,V)}$$

Normalized cut

$$\text{assoc}(A,V) = \sum_{u \in A,\ t \in V} w(u,t)$$

How much is A connected to the graph as a whole

Graph-based segmentation



$$\mathrm{Ncut}(A,B) = \frac{3}{21} + \frac{3}{16}$$

Graph-based segmentation

- Shi and Malik turned graph cuts into an eigenvector/eigenvalue problem.
- Set up a weighted graph $G=(V,E)$.
  - V is the set of (N) pixels.
  - E is a set of weighted edges (weight w_{ij} gives the similarity between nodes i and j).
  - Length N vector d: d_i is the sum of the weights from node i to all other nodes.
  - N x N matrix D: D is a diagonal matrix with d on its diagonal.
  - N x N symmetric matrix W: $W_{ij} = w_{ij}$.

Graph-based segmentation

- Let x be a characteristic vector of a set A of nodes.
  - x_i = 1 if node i is in a set A
  - x_i = -1 otherwise
- Let y be a continuous approximation to x

$$y = (1 + x) - \frac{\sum_{x_i > 0} d_i}{\sum_{x_i < 0} d_i}(1 - x).$$

- Solve the system of equations

  $(D - W)\, y = \lambda\, D\, y$

  for the eigenvectors y and eigenvalues λ.
- Use the eigenvector y with second smallest eigenvalue to bipartition the graph (y → x → A).
- If further subdivision is merited, repeat recursively.

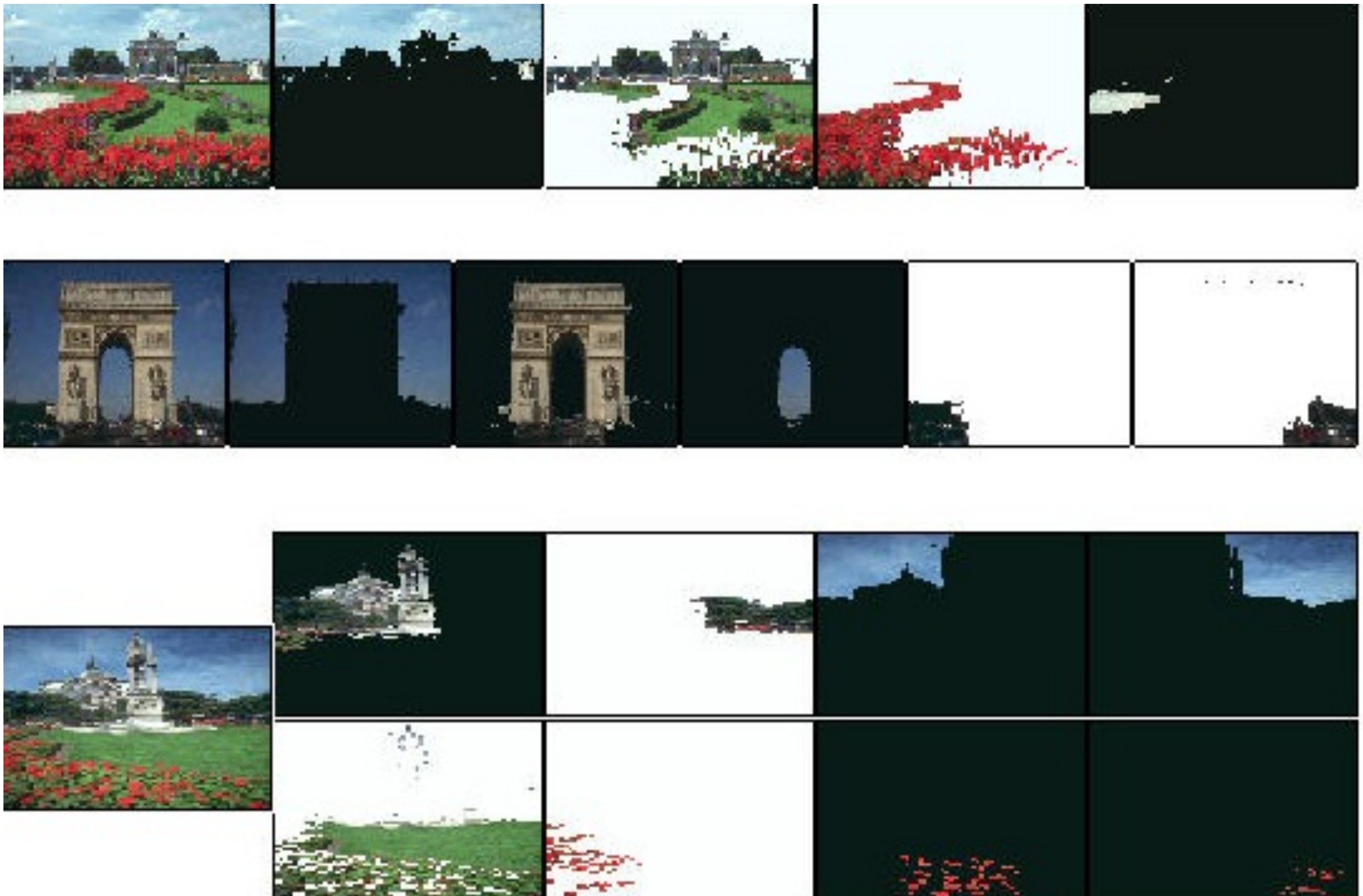
Graph-based segmentation

- Edge weights w(i,j) can be defined by

$$w(i,j) = e^{-||F(i)-F(j)||^2 / \sigma_I^2} * \begin{cases} e^{-||X(i)-X(j)||^2 / \sigma_X^2} & \text{if } ||X(i)-X(j)||^2 < r \\ 0 & \text{otherwise} \end{cases}$$

  where
  - X(i) is the spatial location of node I
  - F(i) is the feature vector for node I
    which can be intensity, color, texture, motion…
- The formula is set up so that w(i,j) is 0 for nodes that are too far apart.

Graph-based segmentation



Graph-based segmentation

- Pros:
  - Generic framework, flexible to choice of function that computes weights (“affinities”) between nodes
  - Does not require model of the data distribution

- Cons:
  - Time complexity can be high
  - Dense, highly connected graphs $\rightarrow$ many affinity computations
  - Solving eigenvalue problem

Adapted from Kristen Grauman

Superpixels

- Goal is to divide the image into a large number of regions, such that each region lies within object boundaries.
- Desirable properties:
  - Good adherence to object boundaries
  - Regular shape and similar size
  - Fast to compute and simple to use
- Popular methods:
  - Graph-based superpixel methods
    - Ncut
  - Gradient-based superpixel methods
    - Waterpixel
    - SLIC

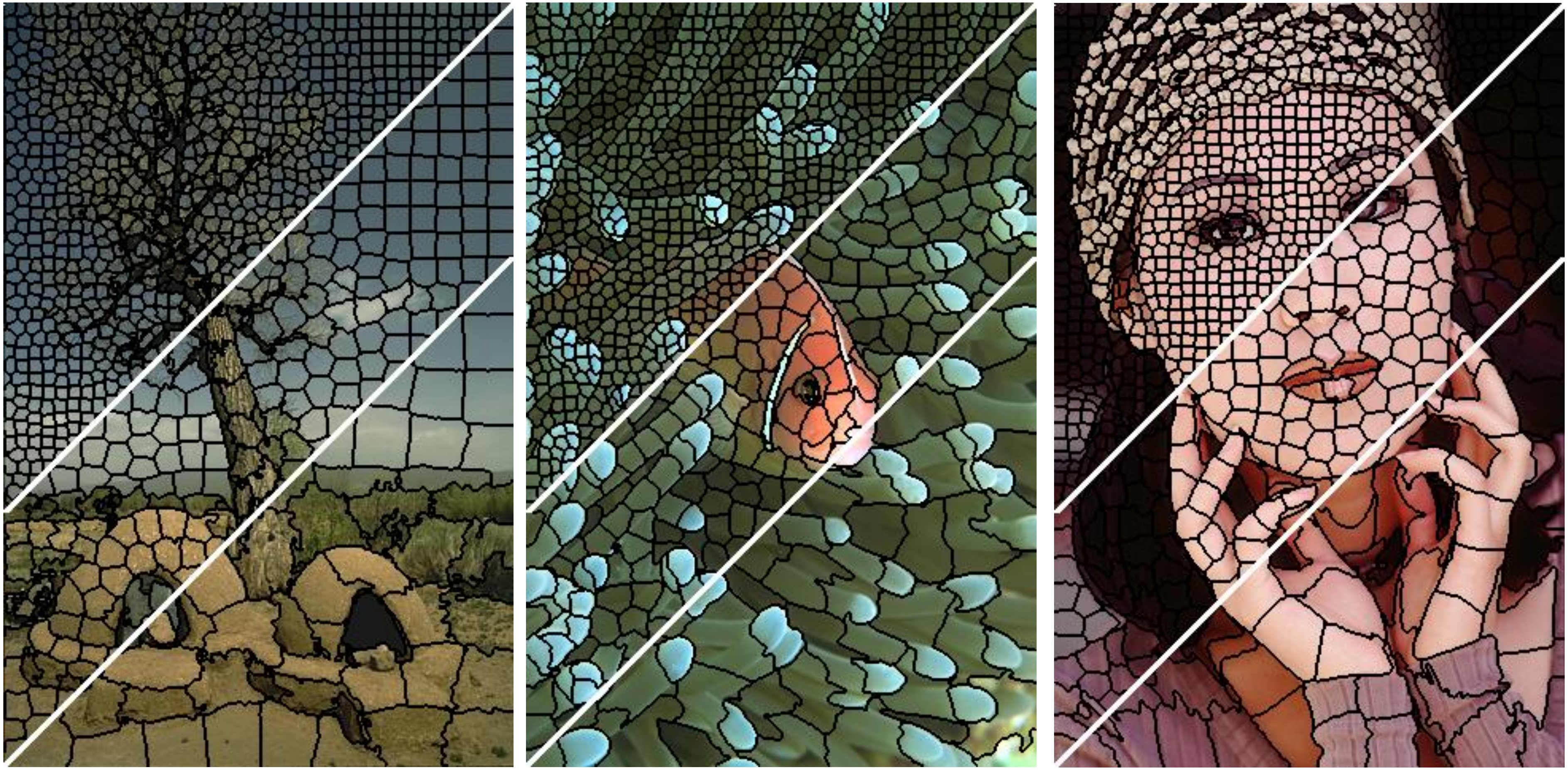
Adapted from Derek Hoiem and Hsuan-Yi Ko

Superpixels

- Start from rough initial clusters and iteratively refine them until some convergence criterion is met.
- Simple linear iterative clustering (SLIC):
  1. Initialize cluster centers on pixel grid in steps S.
     - Features: Lab color, x-y position
  2. Move centers to position in 3x3 window with smallest gradient.
  3. Compare each pixel to cluster center within 2S pixel distance and assign to nearest.
  4. Recompute cluster centers as mean color/position of pixels belonging to each cluster.
  5. Stop when residual error is small.

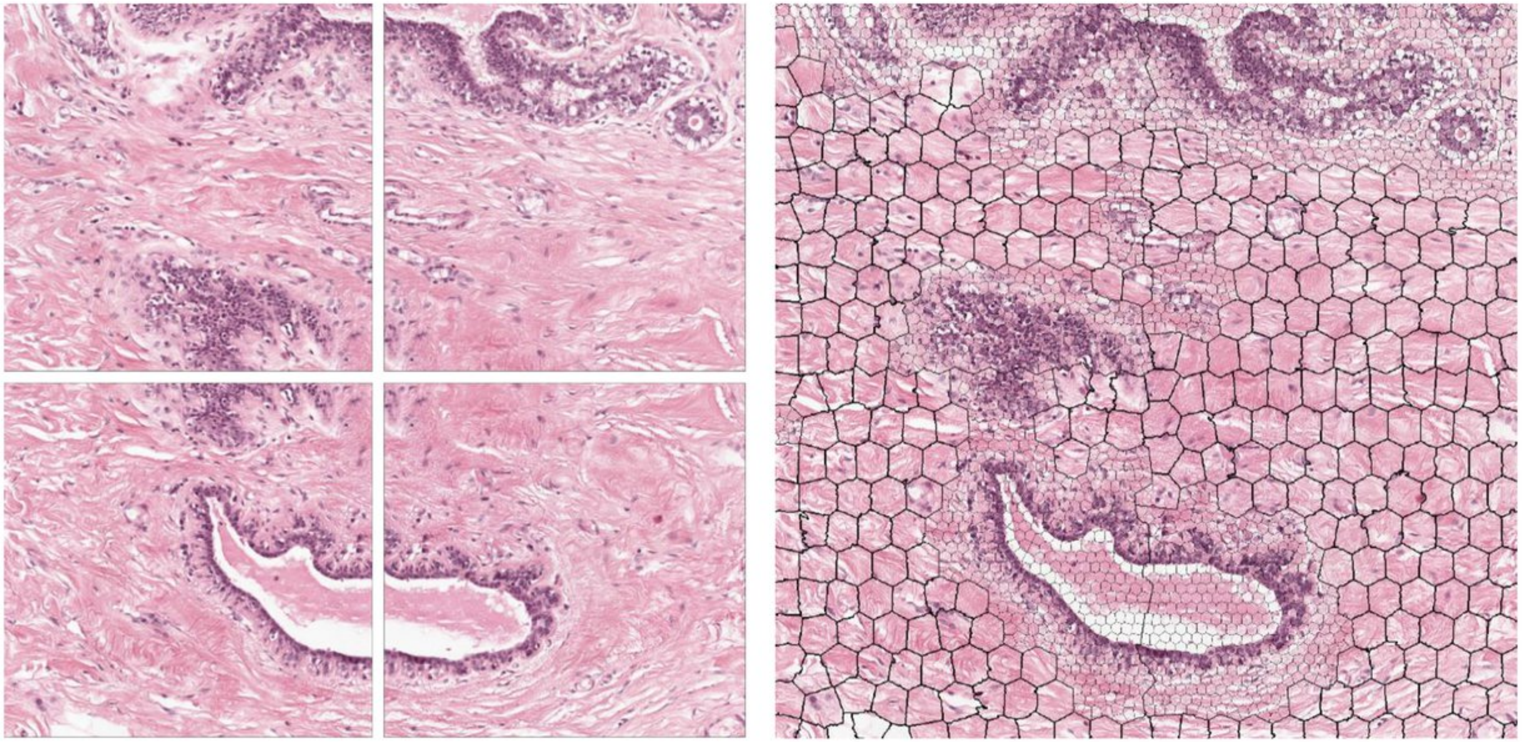
Adapted from Derek Hoiem

Superpixels



http://ivrl.epfl.ch/research/superpixels

Superpixels



Multi-scale superpixel segmentation of a breast biopsy image.

B. E. Bejnordi, G. Litjens, M. Hermsen, N. Karssemeijer, and J. A. van der Laak, "A multi-scale superpixel classification approach to the detection of regions of interest in whole slide histopathology images," SPIE Medical Imaging Symposium, 2015.