

Image Segmentation

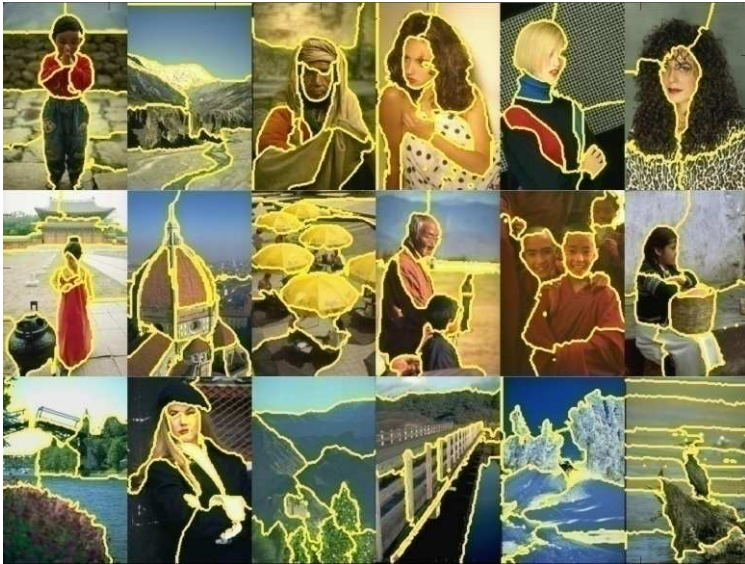
Selim Aksoy

Department of Computer Engineering

Bilkent University

saksoy@cs.bilkent.edu.tr

Examples of grouping in vision

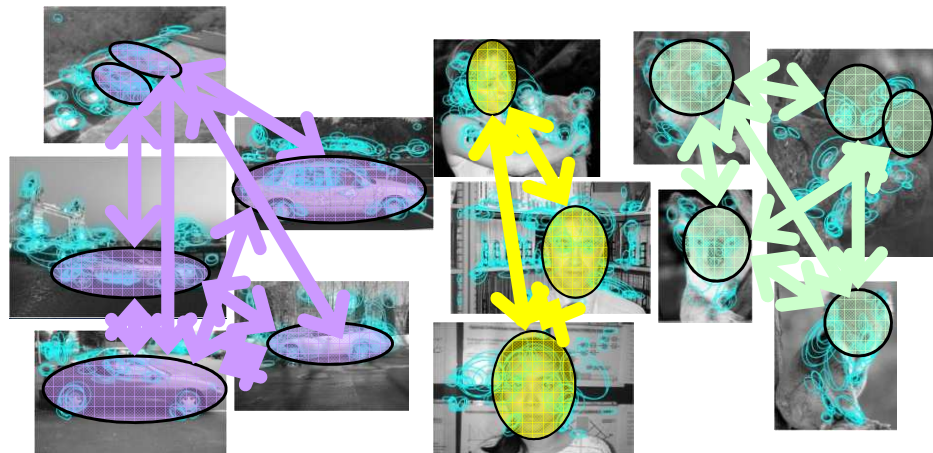


[Figure by J. Shi]

Determine image regions

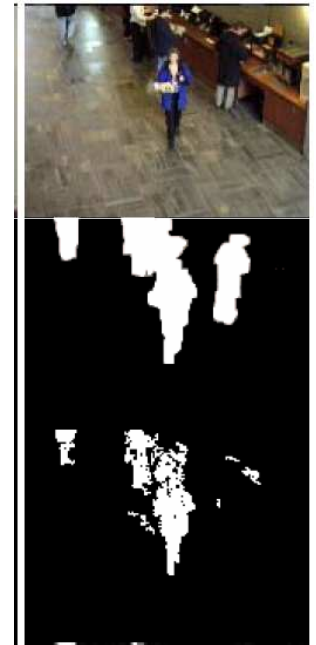


Group video frames into shots



[Figure by Grauman & Darrell]

Object-level grouping



[Figure by Wang & Suter]

Figure-ground separation

Image segmentation

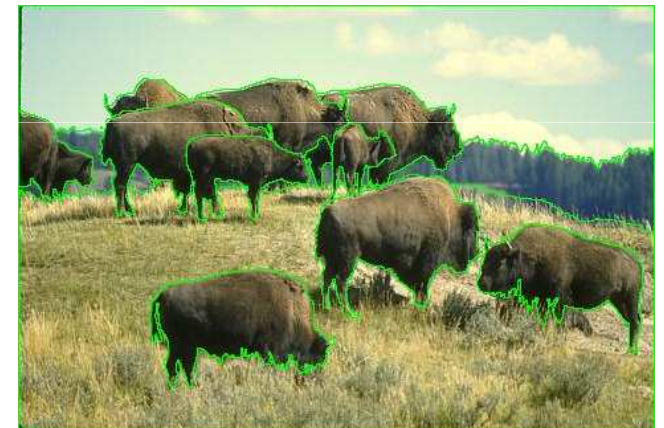
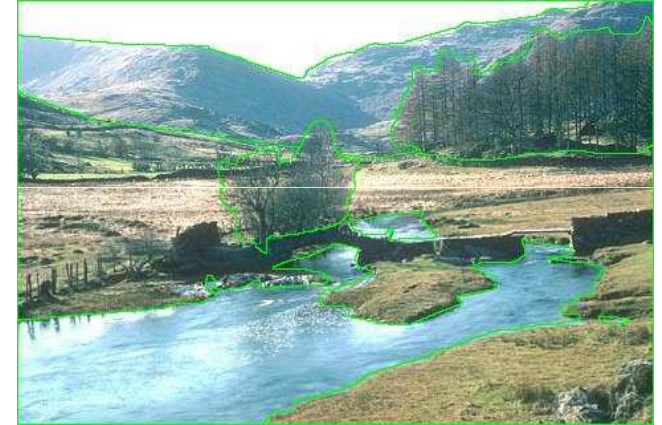
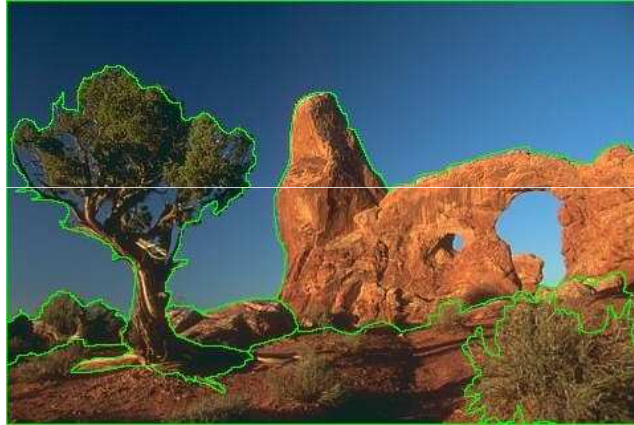
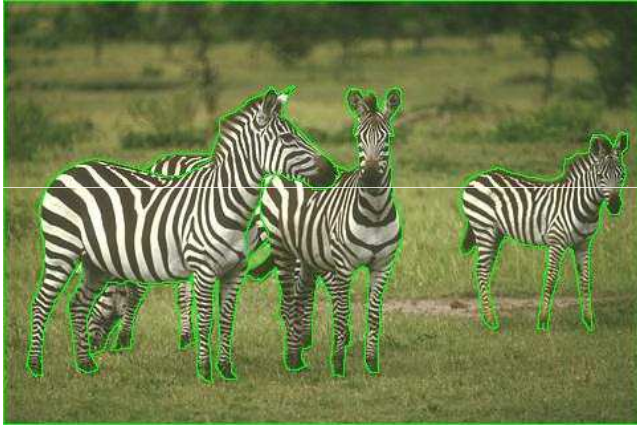
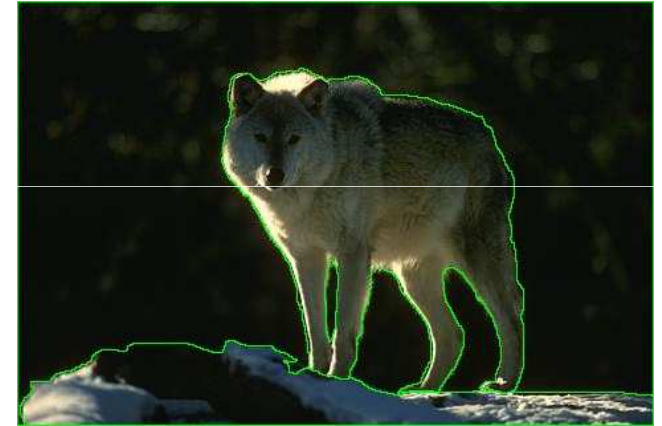
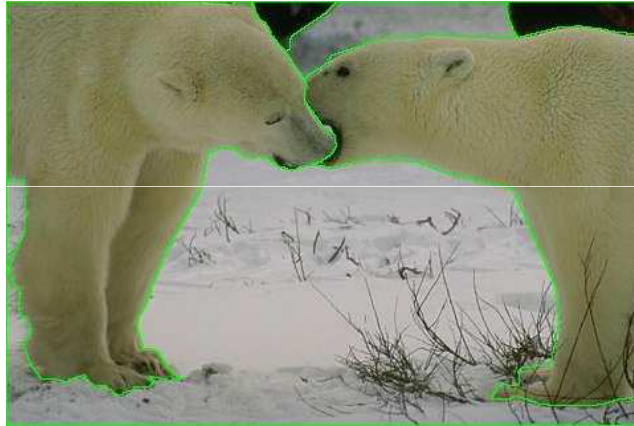


Image segmentation



From images to objects

- What defines an object?
 - Subjective problem, but has been well-studied.
 - Gestalt laws seek to formalize this.
 - “What is interesting and what is not” depends on the application.
 - Broad theory is absent at present.

Gestalt laws

- A series of factors affect whether elements should be grouped together.
 - Proximity: tokens that are nearby tend to be grouped.
 - Similarity: similar tokens tend to be grouped together.
 - Common fate: tokens that have coherent motion tend to be grouped together.
 - Common region: tokens that lie inside the same closed region tend to be grouped together.
 - Parallelism: parallel curves or tokens tend to be grouped together.
 - Closure: tokens or curves that tend to lead to closed curves tend to be grouped together.
 - Symmetry: curves that lead to symmetric groups are grouped together.
 - Continuity: tokens that lead to “continuous” curves tend to be grouped.
 - Familiar configuration: tokens that, when grouped, lead to a familiar object, tend to be grouped together.

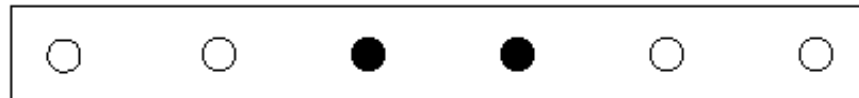
Gestalt laws



Not grouped



Proximity



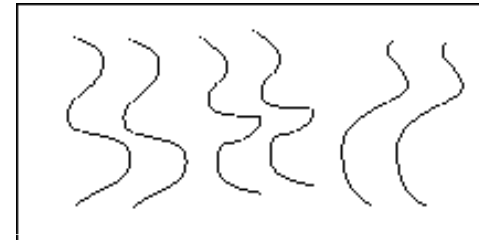
Similarity



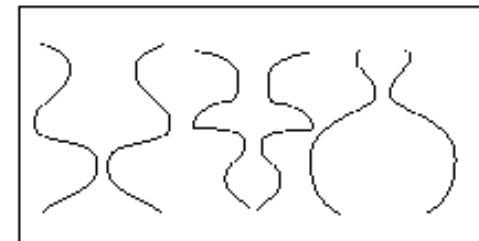
Similarity



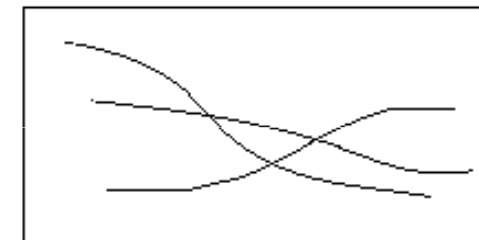
Common Fate



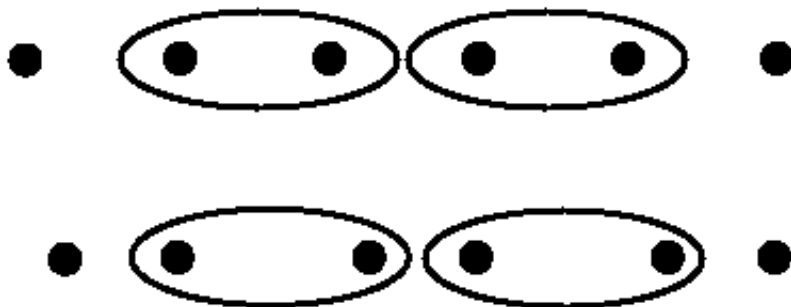
Parallelism



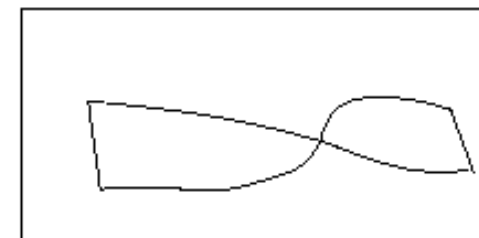
Symmetry



Continuity



Common Region

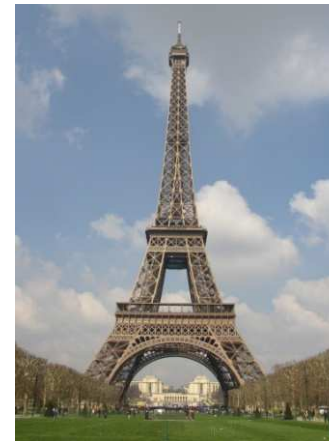


Closure

Gestalt laws



Similarity



Symmetry

Gestalt laws

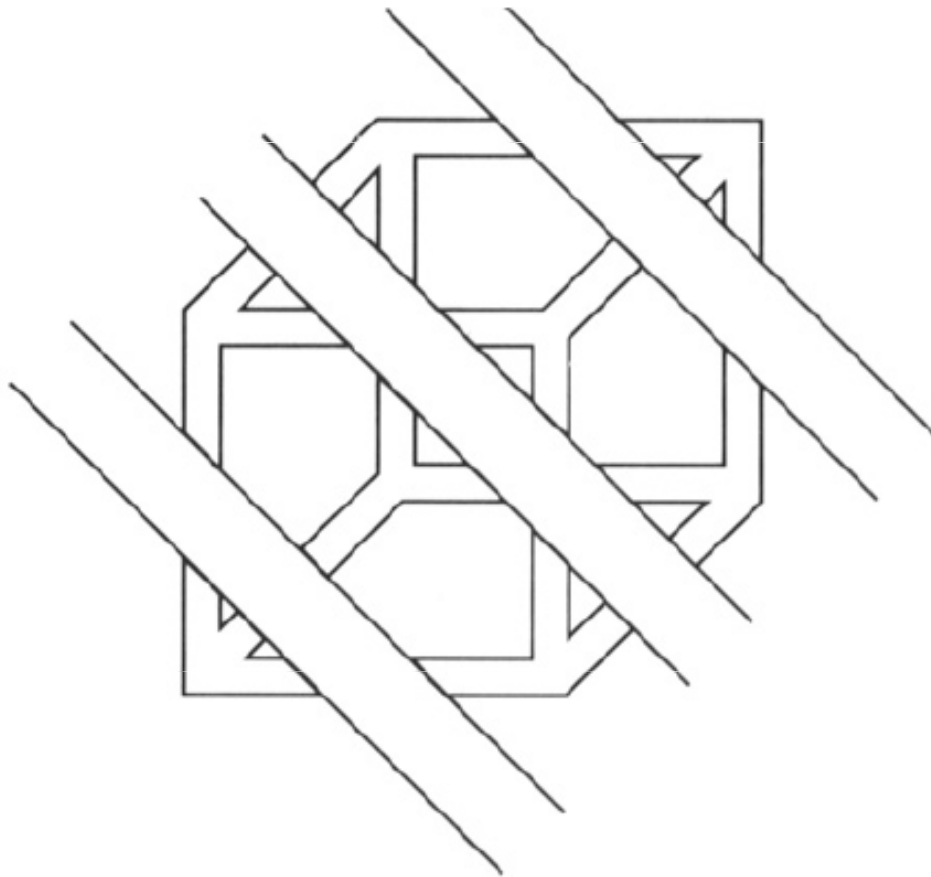


Proximity



Common fate

Gestalt laws



Continuity, explanation by occlusion

Image segmentation

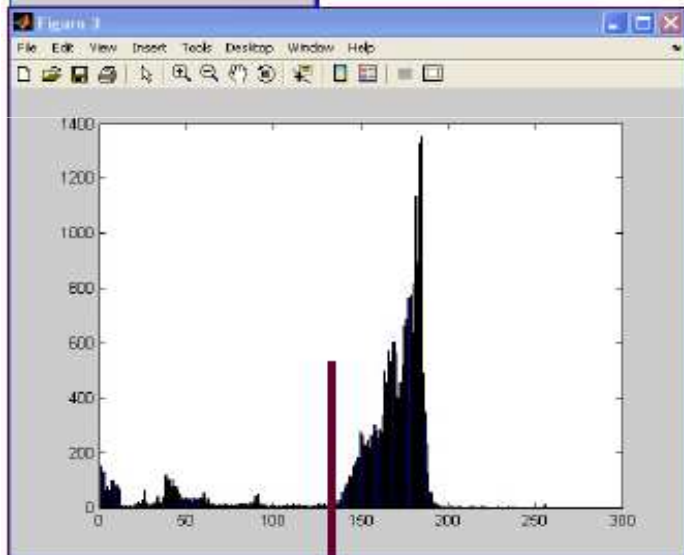
- Image segmentation is the operation of partitioning an image into a collection of connected sets of pixels.
- Segmentation criteria: a segmentation is a partition of an image I into a set of regions S satisfying:
 1. $\cup S_i = S$ Partition covers the whole image.
 2. $S_i \cap S_j = \emptyset, i \neq j$ No regions intersect.
 3. $\forall S_i, P(S_i) = \text{true}$ Homogeneity predicate is satisfied by each region.
 4. $P(S_i \cup S_j) = \text{false}, i \neq j, S_i \text{ adjacent } S_j$ Union of adjacent regions does not satisfy it.

Image segmentation

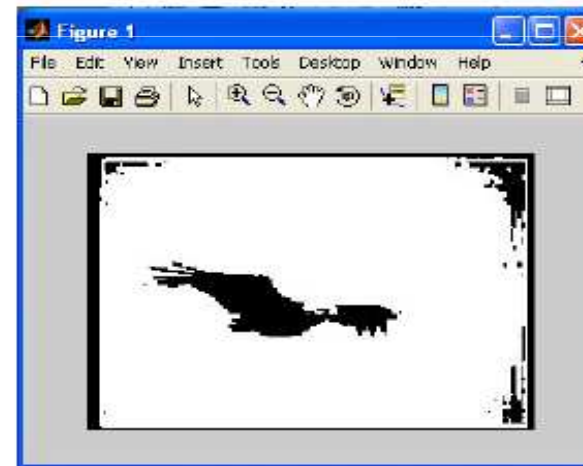
- So, all we have to do is to define and implement the similarity predicate.
 - But, what do we want to be similar in each region?
 - Is there any property that will cause the regions to be meaningful objects?
- Example approaches:
 - Histogram-based
 - Clustering-based
 - Region growing
 - Split-and-merge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

Histogram-based segmentation

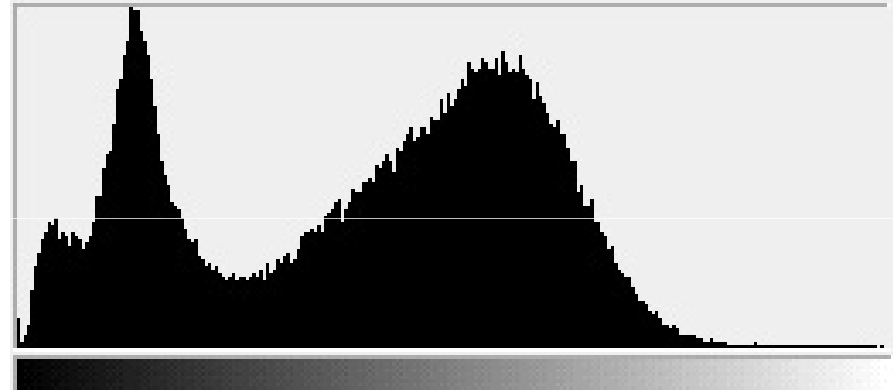
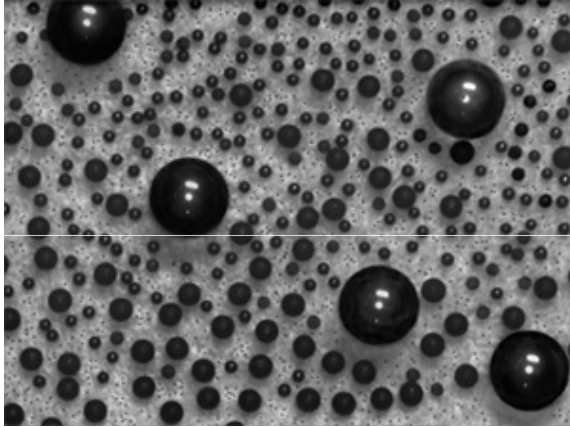
- How many “orange” pixels are in this image?
- This type of question can be answered by looking at the histogram.



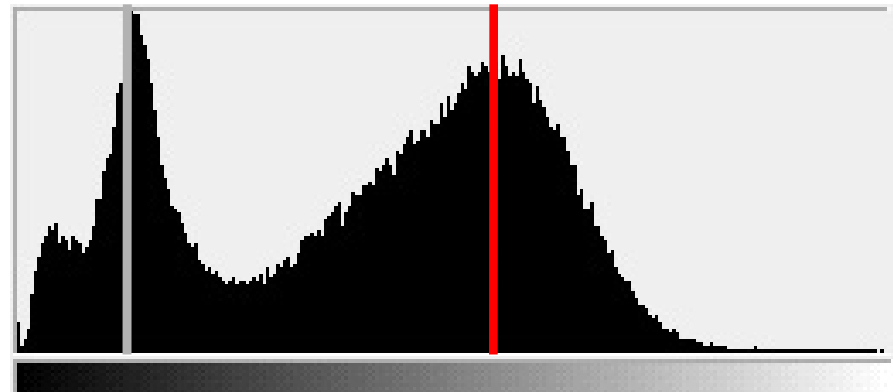
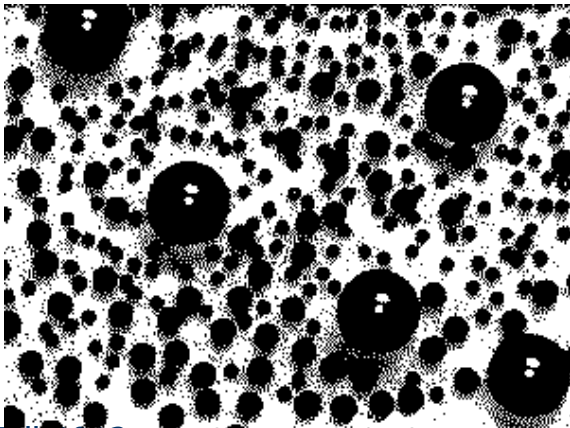
`imshow(B > 140)`



Histogram-based segmentation

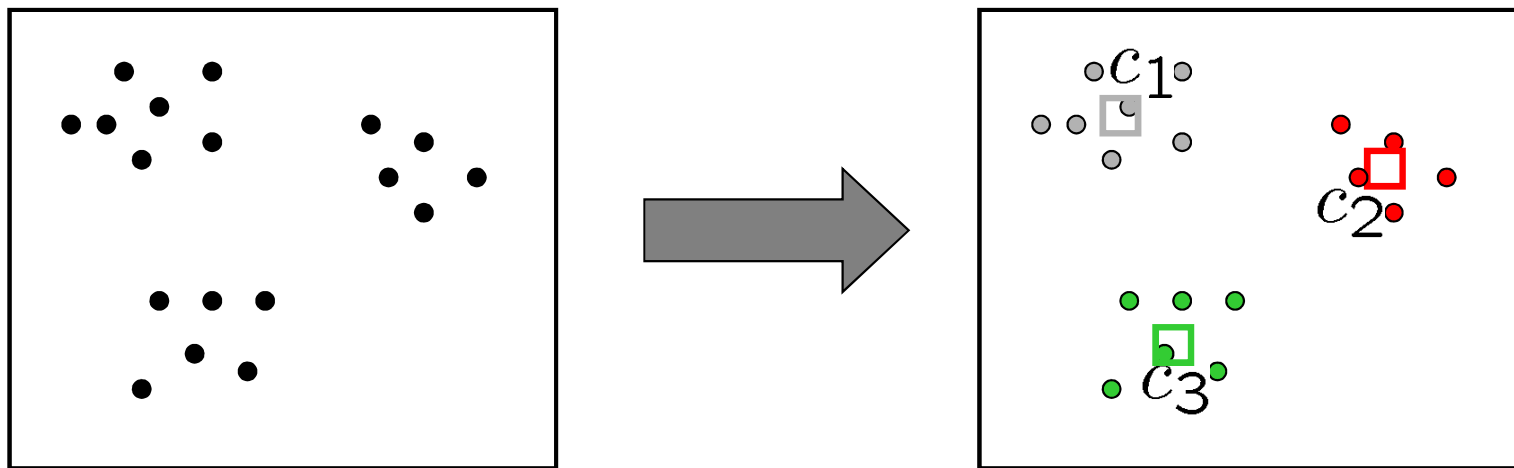


- How many modes are there?
- Solve this by reducing the number of colors to K and mapping each pixel to the closest color.
- Here's what it looks like if we use two colors.



Clustering-based segmentation

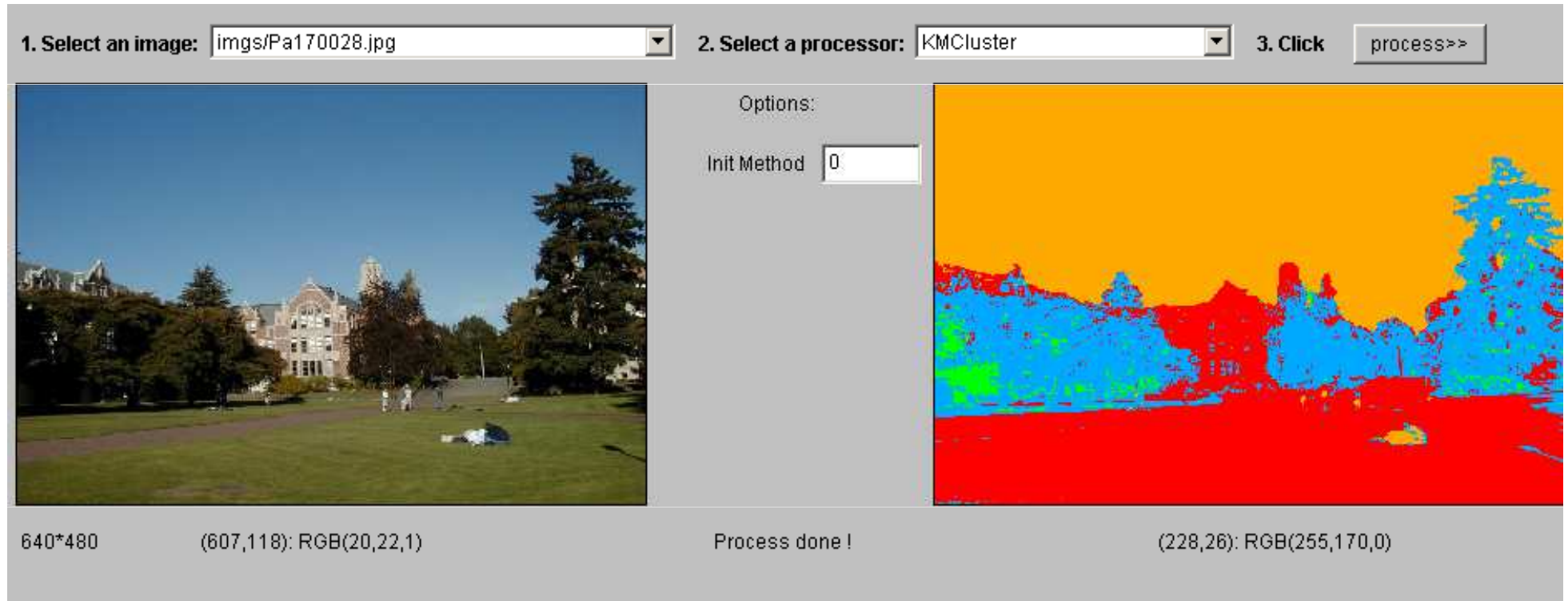
- How to choose the representative colors?
 - This is a clustering problem!



- K-means algorithm can be used for clustering.

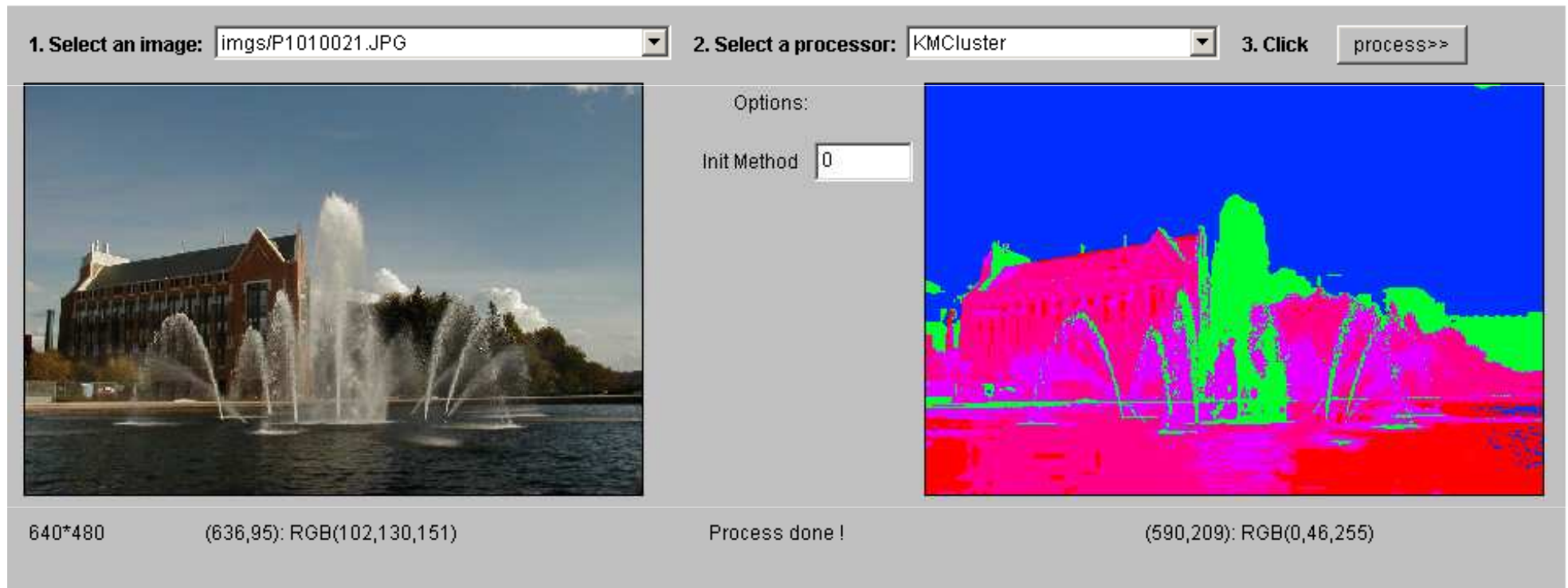
- http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html
- http://www.math.le.ac.uk/people/ag153/homepage/KmeansKmedoids/Kmeans_Kmedoids.html

Clustering-based segmentation



K-means clustering of color.

Clustering-based segmentation



K-means clustering of color.

Clustering-based segmentation

- Clustering can also be used with other features (e.g., texture) in addition to color.

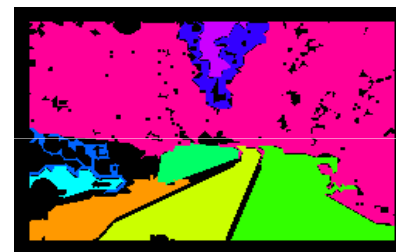
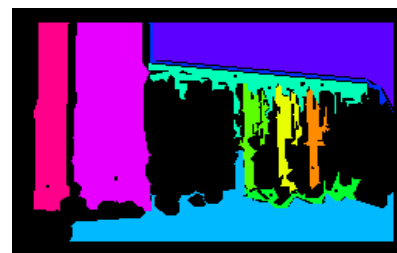
Original Images



Color Regions



Texture Regions



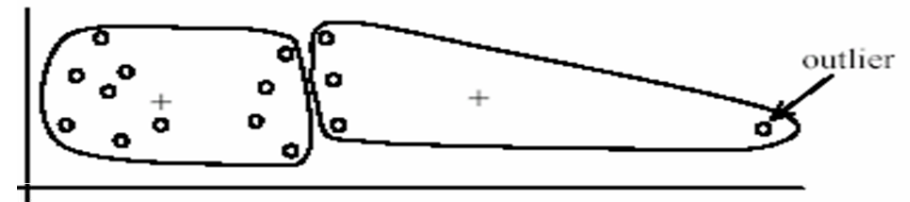
Clustering-based segmentation

■ Pros:

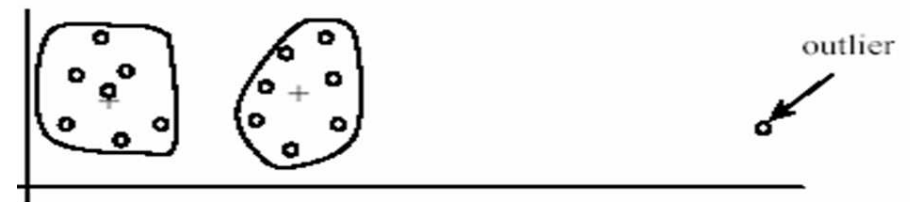
- Simple, fast to compute
- Converges to local minimum of within-cluster squared error

■ Cons:

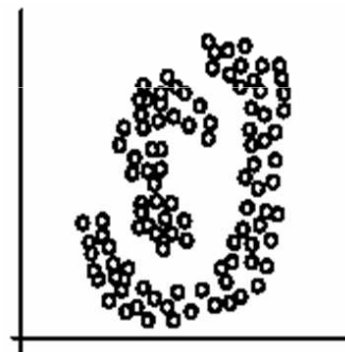
- Setting K?
- Sensitive to initial centers
- Sensitive to outliers
- Detects spherical clusters



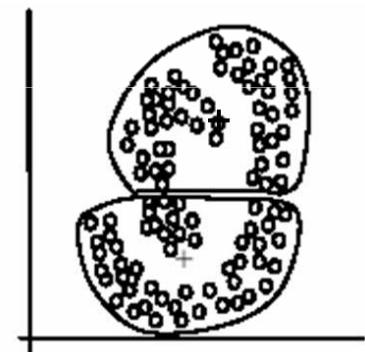
(A): Undesirable clusters



(B): Ideal clusters



(A): Two natural clusters



(B): k -means clusters

Clustering-based segmentation

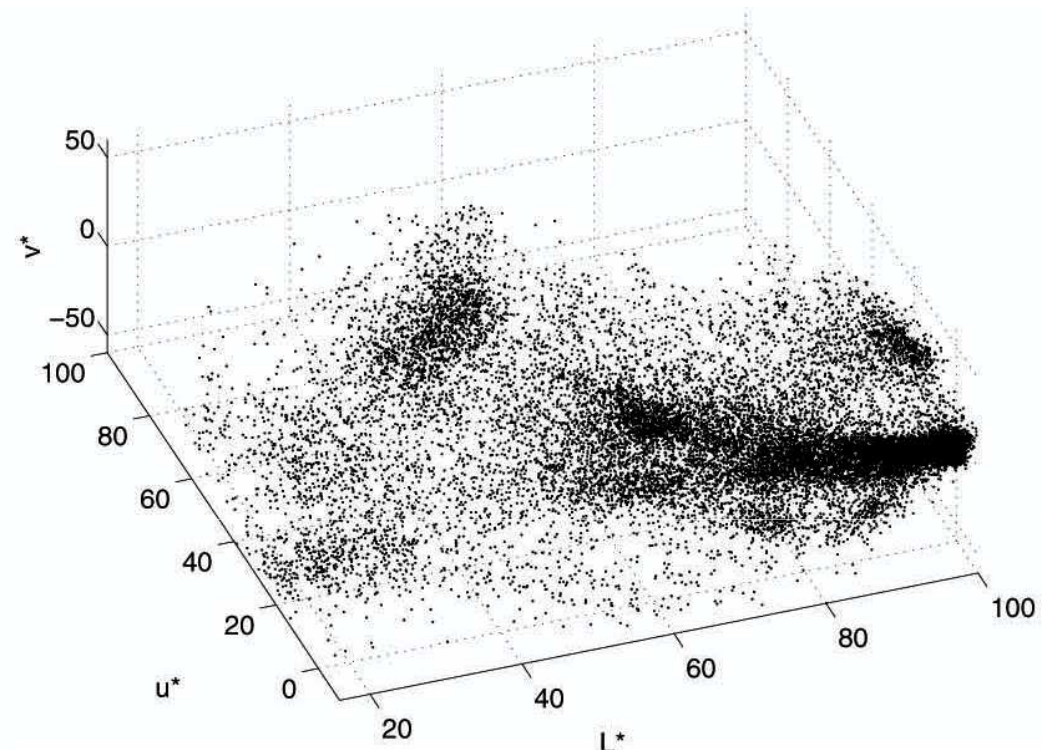
- K-means variants:
 - Different ways to initialize the means.
 - Different stopping criteria.
 - Dynamic methods for determining the right number of clusters (K) for a given image.
- Problem: histogram-based and clustering-based segmentation using color/texture/etc can produce messy/noisy regions. (Why?)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

Mean shift algorithm

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

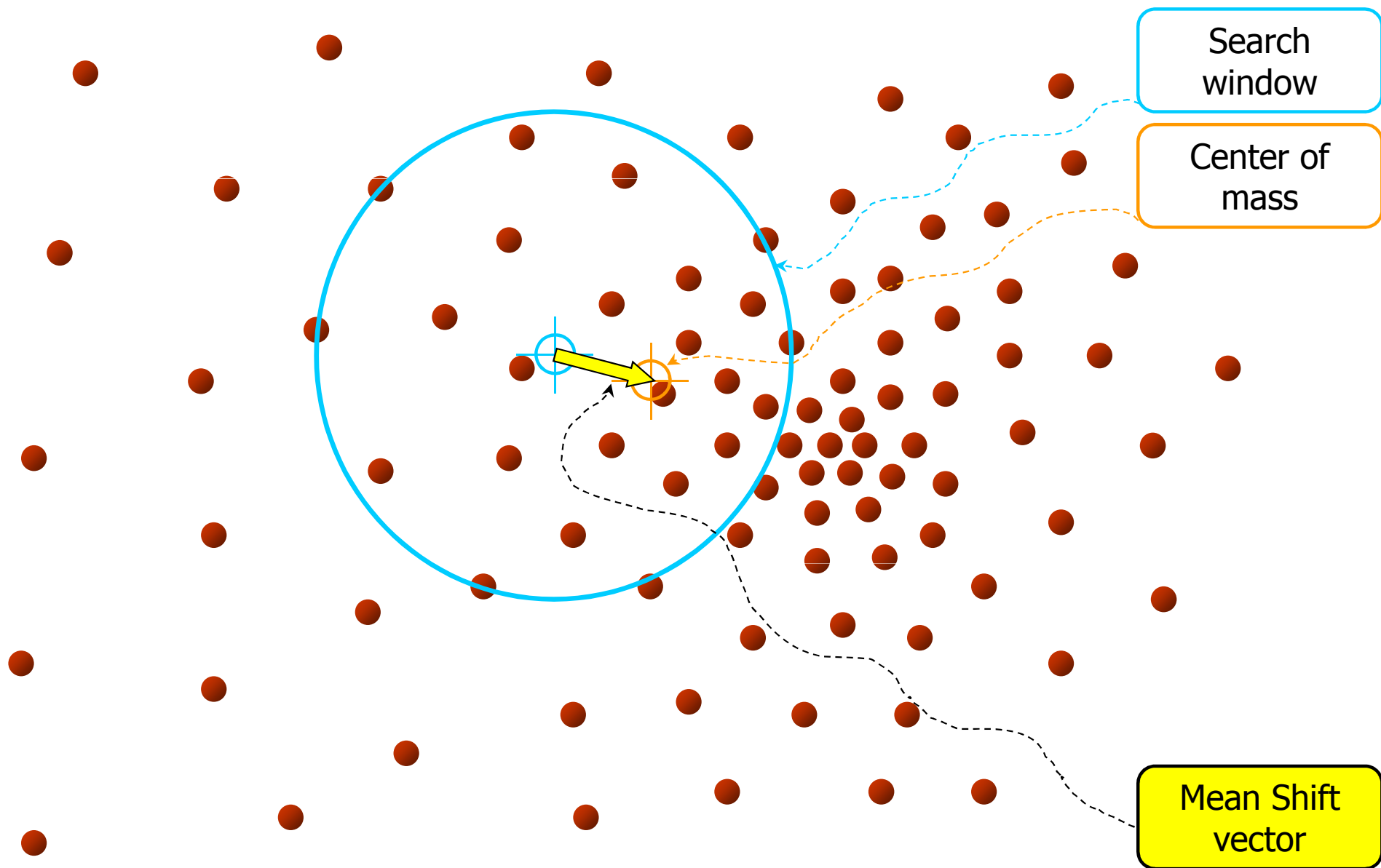


Image

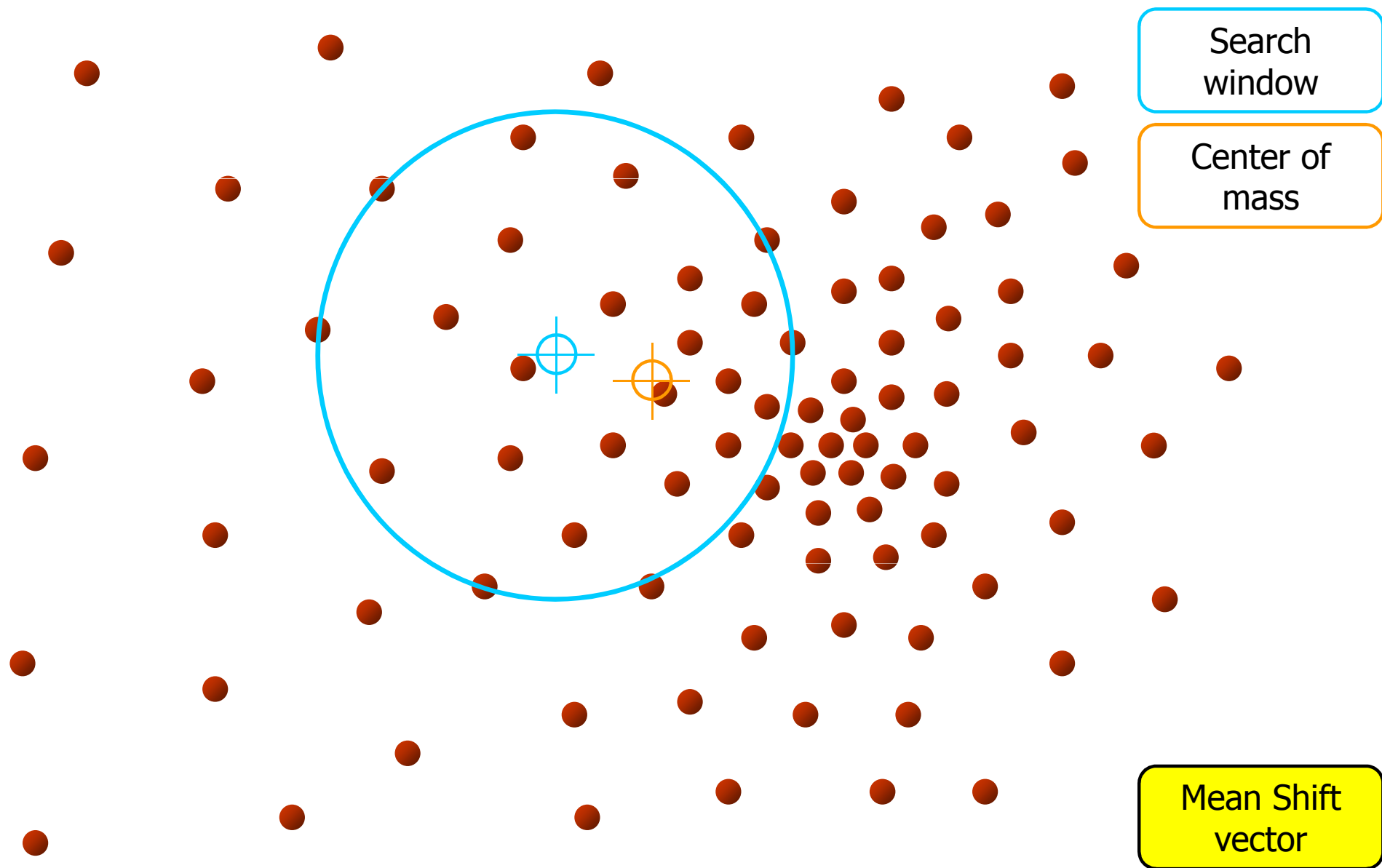


Feature space
($L^*u^*v^*$ color values)

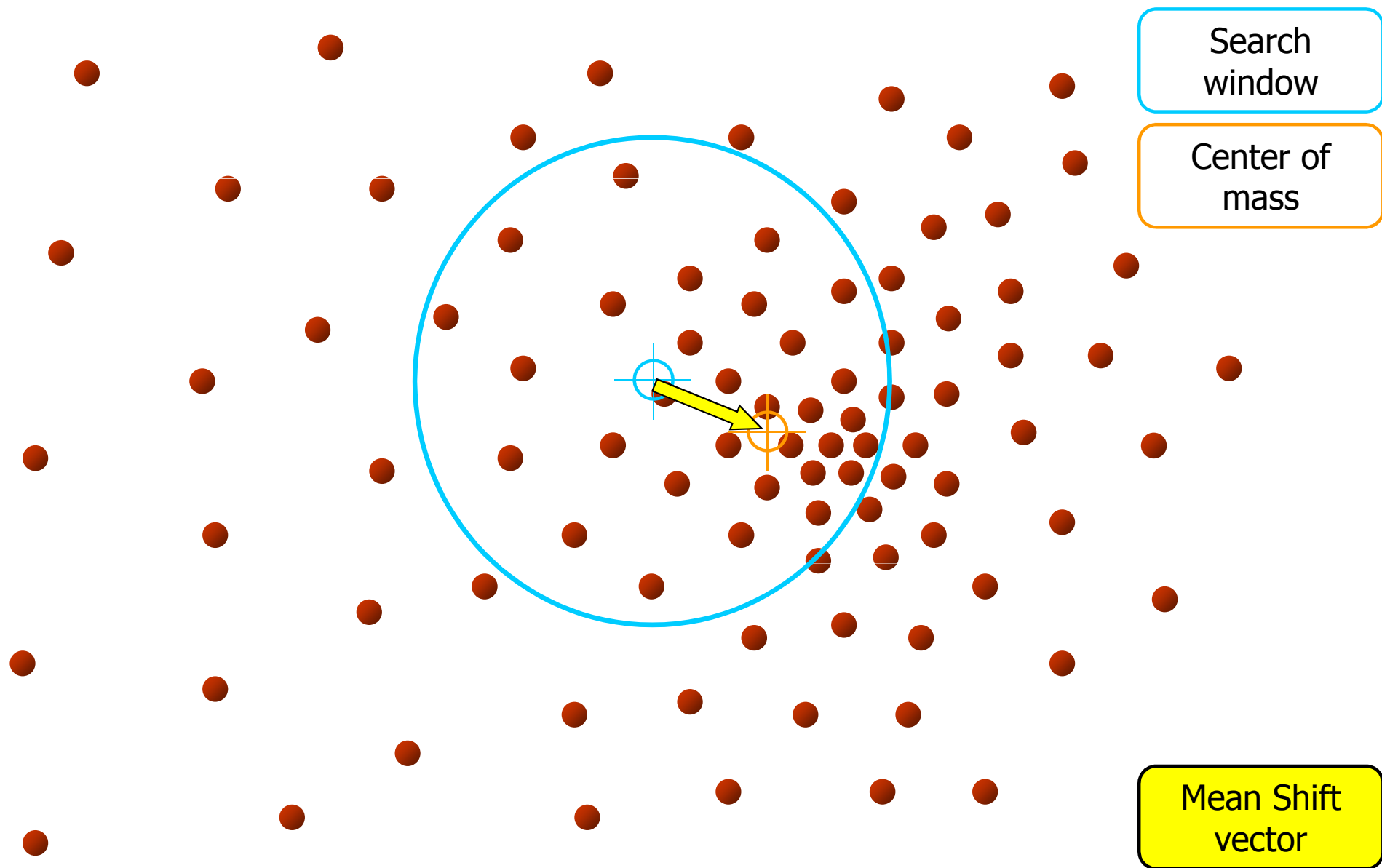
Mean shift algorithm



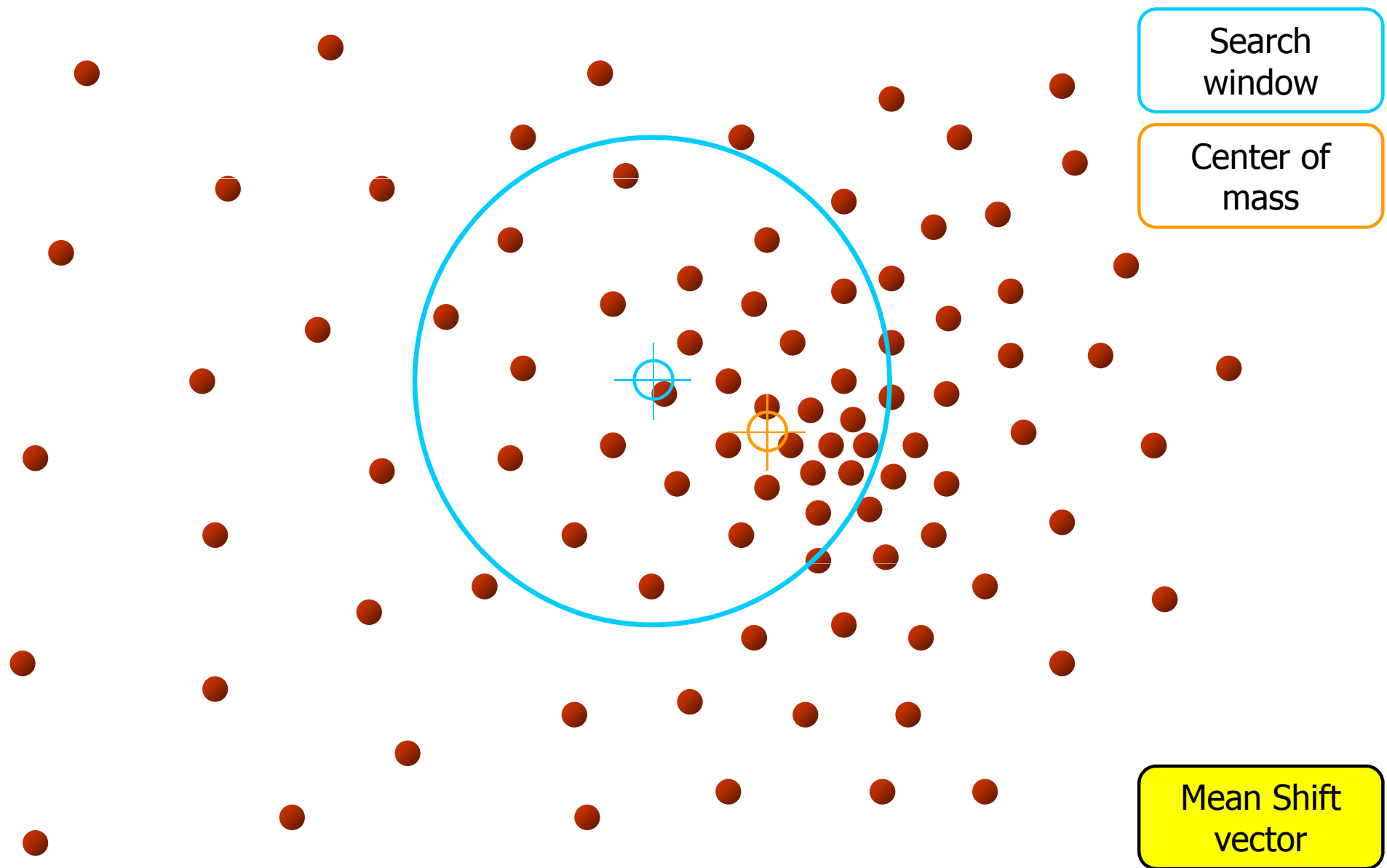
Mean shift algorithm



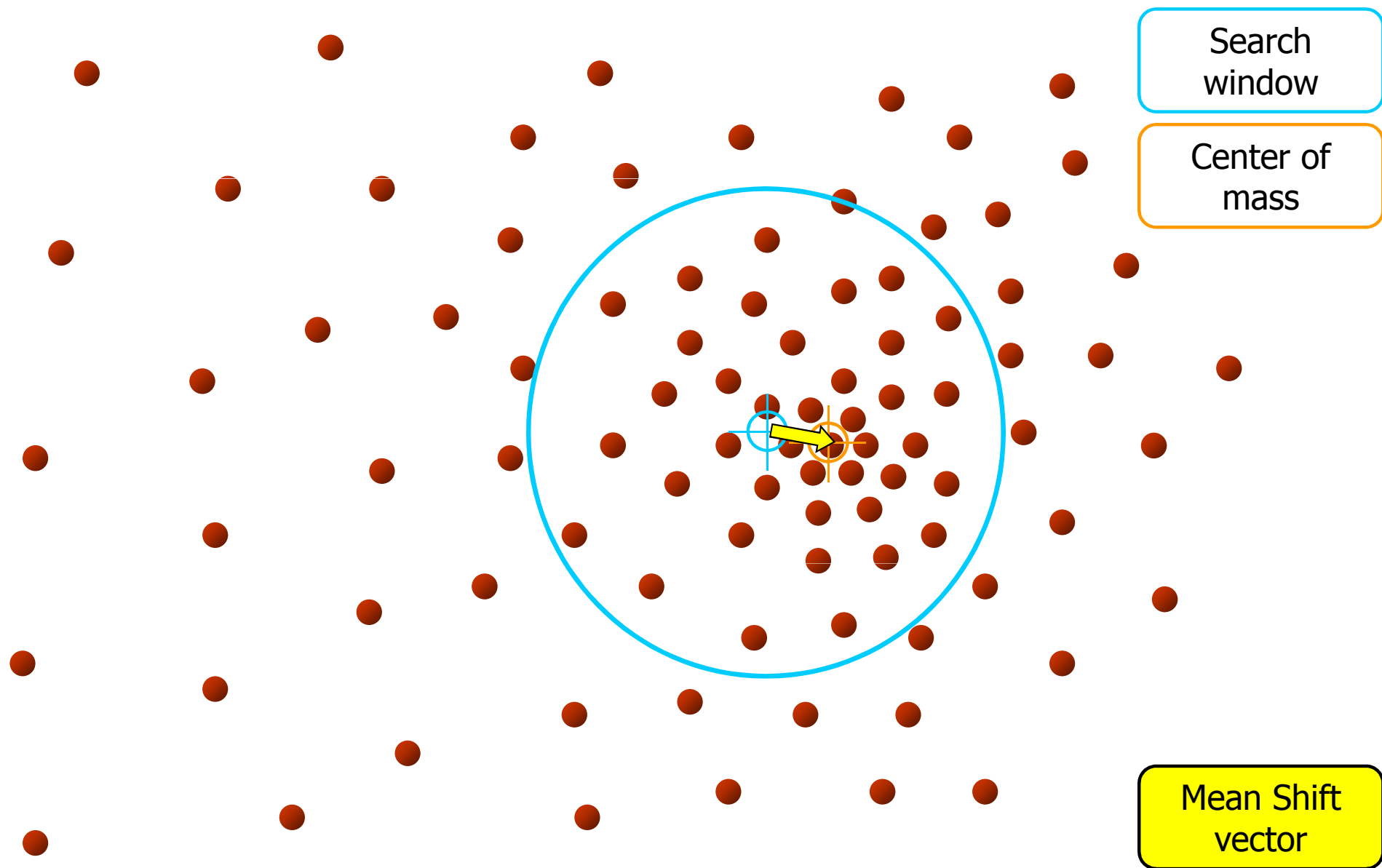
Mean shift algorithm



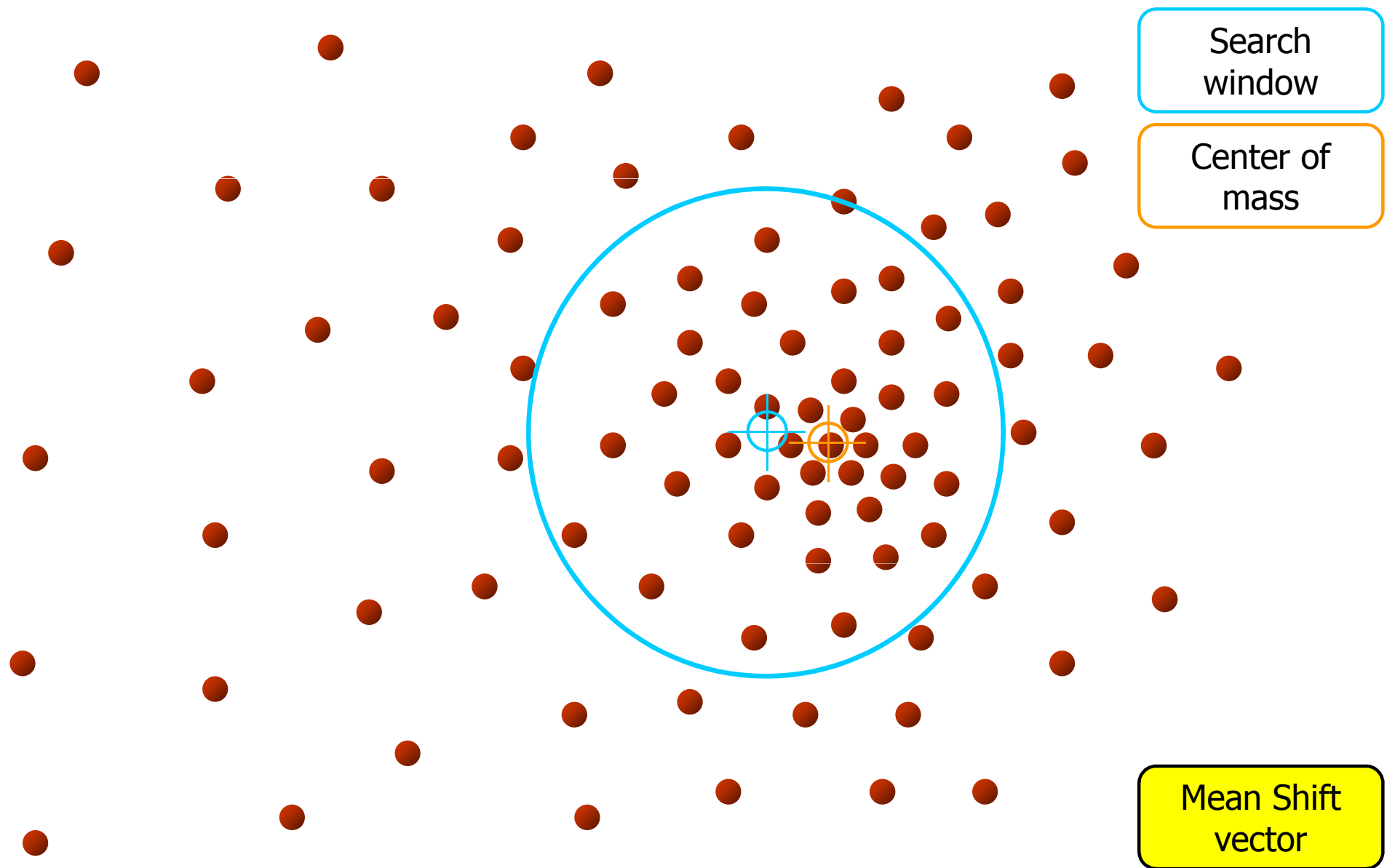
Mean shift algorithm



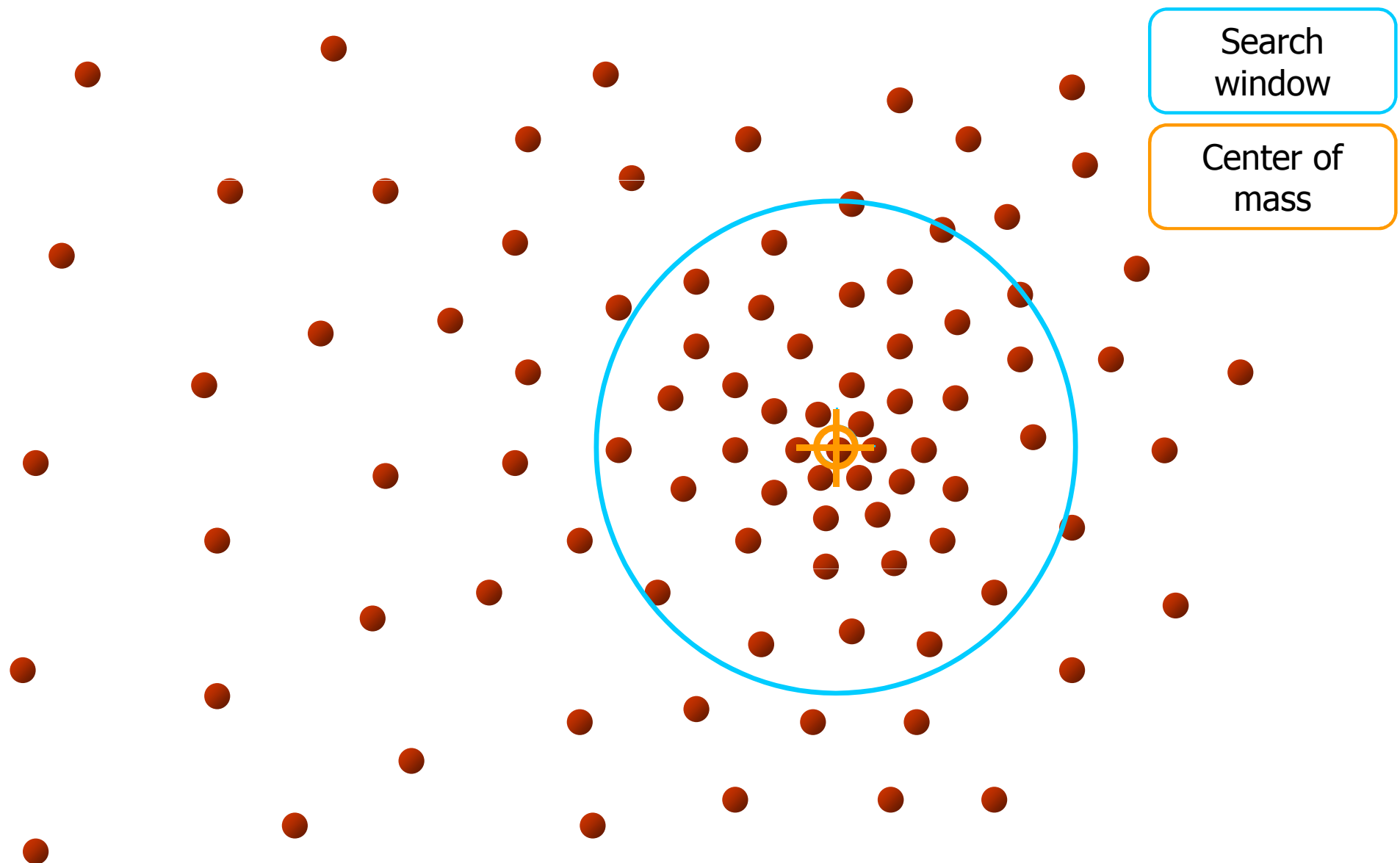
Mean shift algorithm



Mean shift algorithm

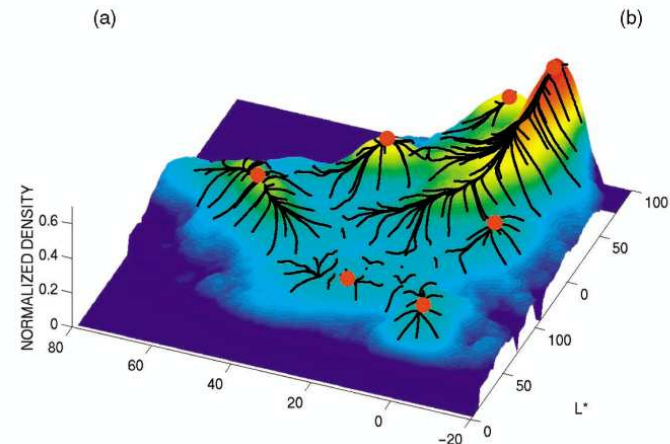
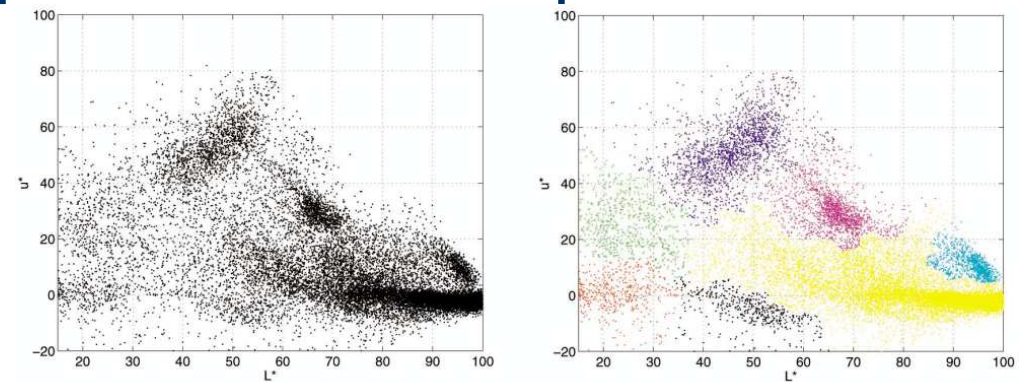


Mean shift algorithm

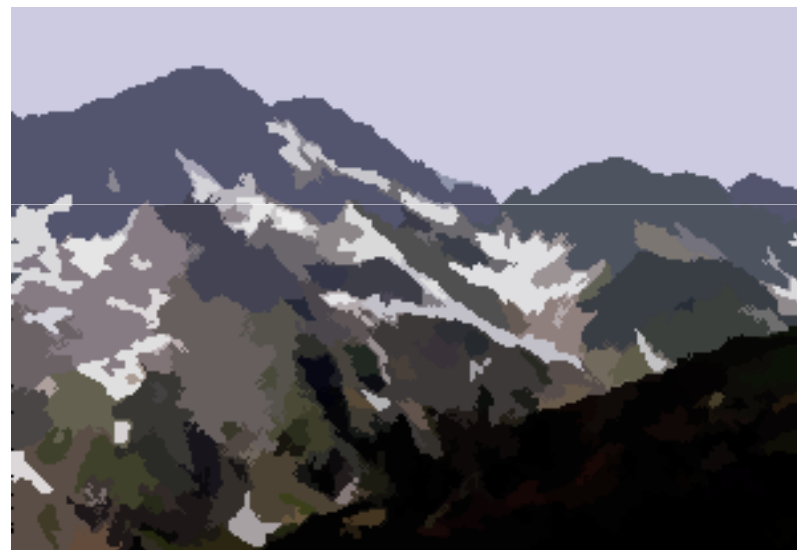


Mean shift clustering/segmentation

- 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



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

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

Region growing

- Region growing techniques start with one pixel of a potential region and try to grow it by adding adjacent pixels till the pixels being compared are too dissimilar.
- The first pixel selected can be just the first unlabeled pixel in the image or a set of seed pixels can be chosen (manually or automatically) from the image.
- 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 based on this measure.

Region 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.
- Let R be the N pixel region so far and p be a neighboring pixel with gray tone y .
- Define the mean \bar{X} and scatter S^2 (sample variance) by

$$\bar{X} = \frac{1}{N} \sum_{(r,c) \in R} I(r,c) \quad S^2 = \frac{1}{N} \sum_{(r,c) \in R} (I(r,c) - \bar{X})^2$$

Region growing

- The T statistic is defined by

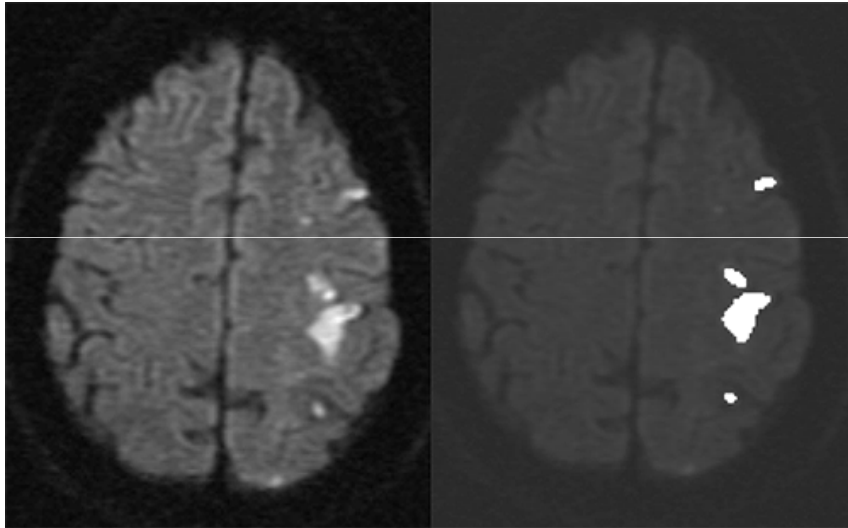
$$T = \left(\frac{(N-1)N}{(N+1)} (p - \bar{X})^2 / S^2 \right)^{1/2}$$

- 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).

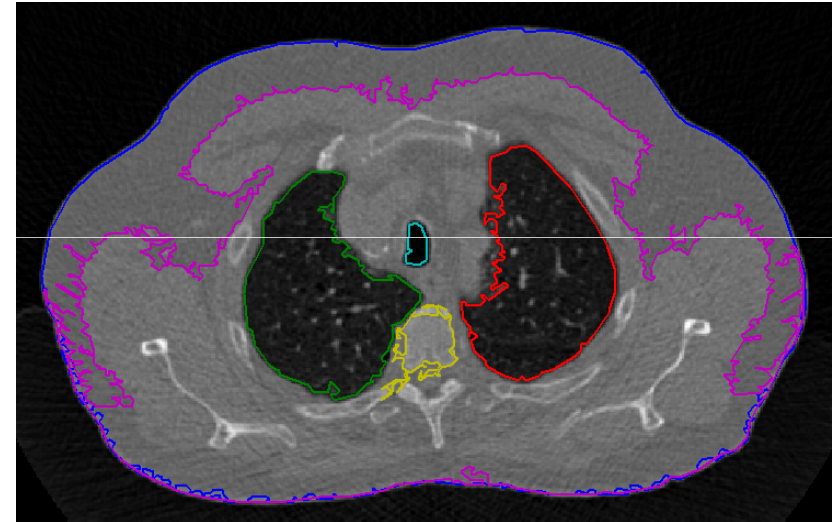
Region growing

- For the T distribution, statistical tables give us the probability $\Pr(T \leq t)$ for a given degrees of freedom and a confidence level. From this, pick a suitable threshold t .
- If the computed $T \leq t$ for desired confidence level, add p to region R and update the mean and scatter using p .
- 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.
- Many other statistical and distance-based methods have also been proposed for region growing.

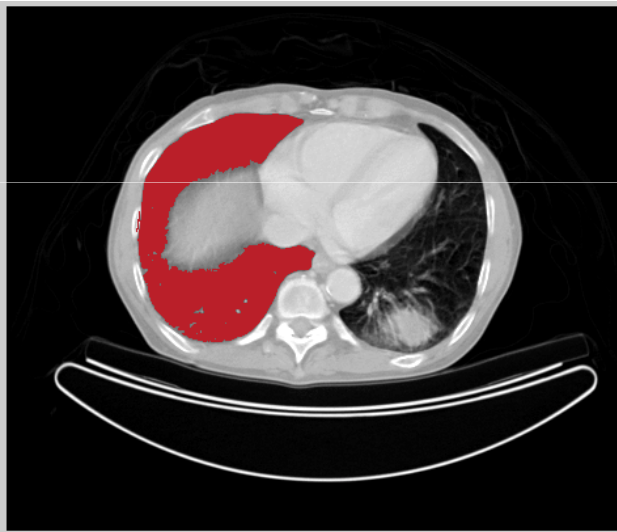
Region growing



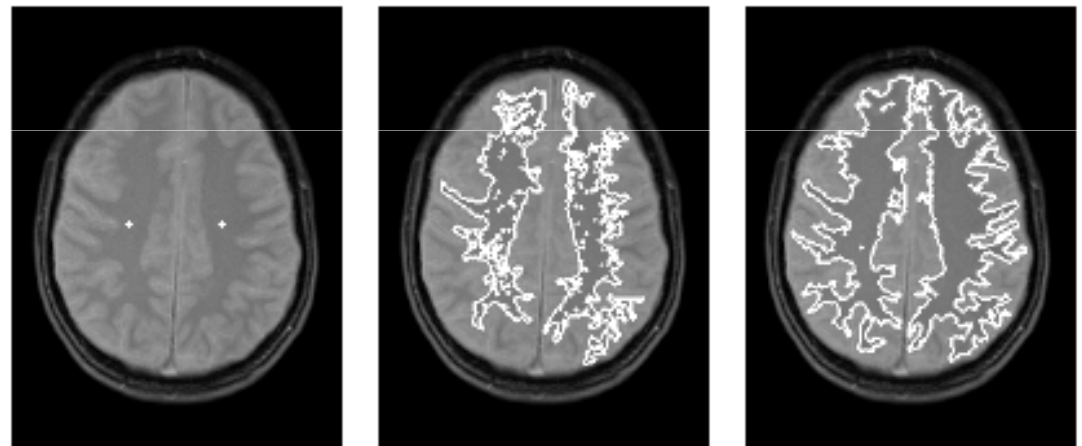
<http://www.bigr.nl/website/static/research/regrow.html>



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



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

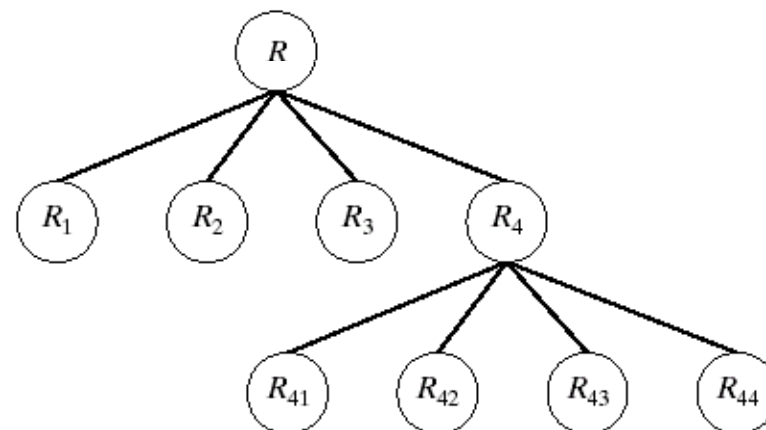
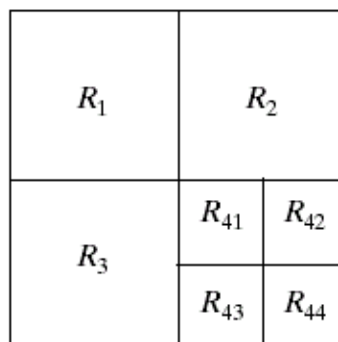


<http://www.creatis.insa-lyon.fr/~grenier/?p=172>

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



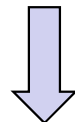
Split-and-merge

Split and merge example



VTU000130CT25CTW001
Image Processing

82



Split and merge example

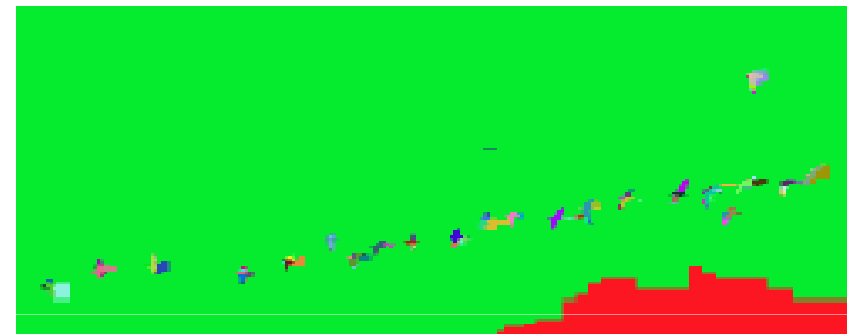


4087 regions

VTU000130CT25CTW001
Image Processing

84

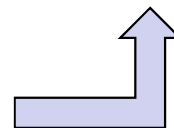
Split and merge example



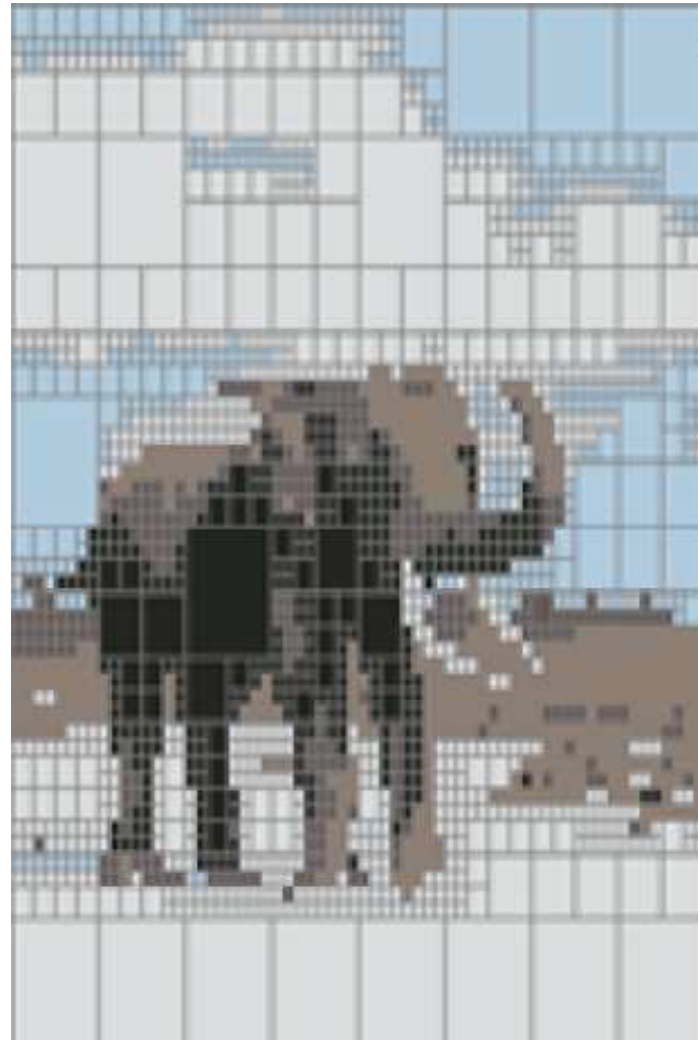
135 regions

VTU000130CT25CTW001
Image Processing

85

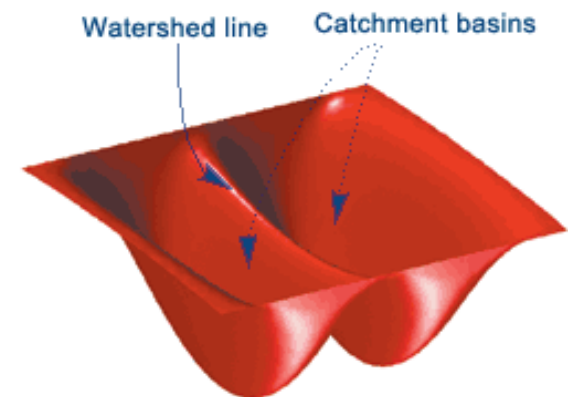
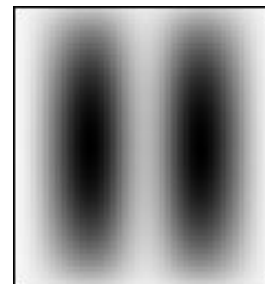


Split-and-merge



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

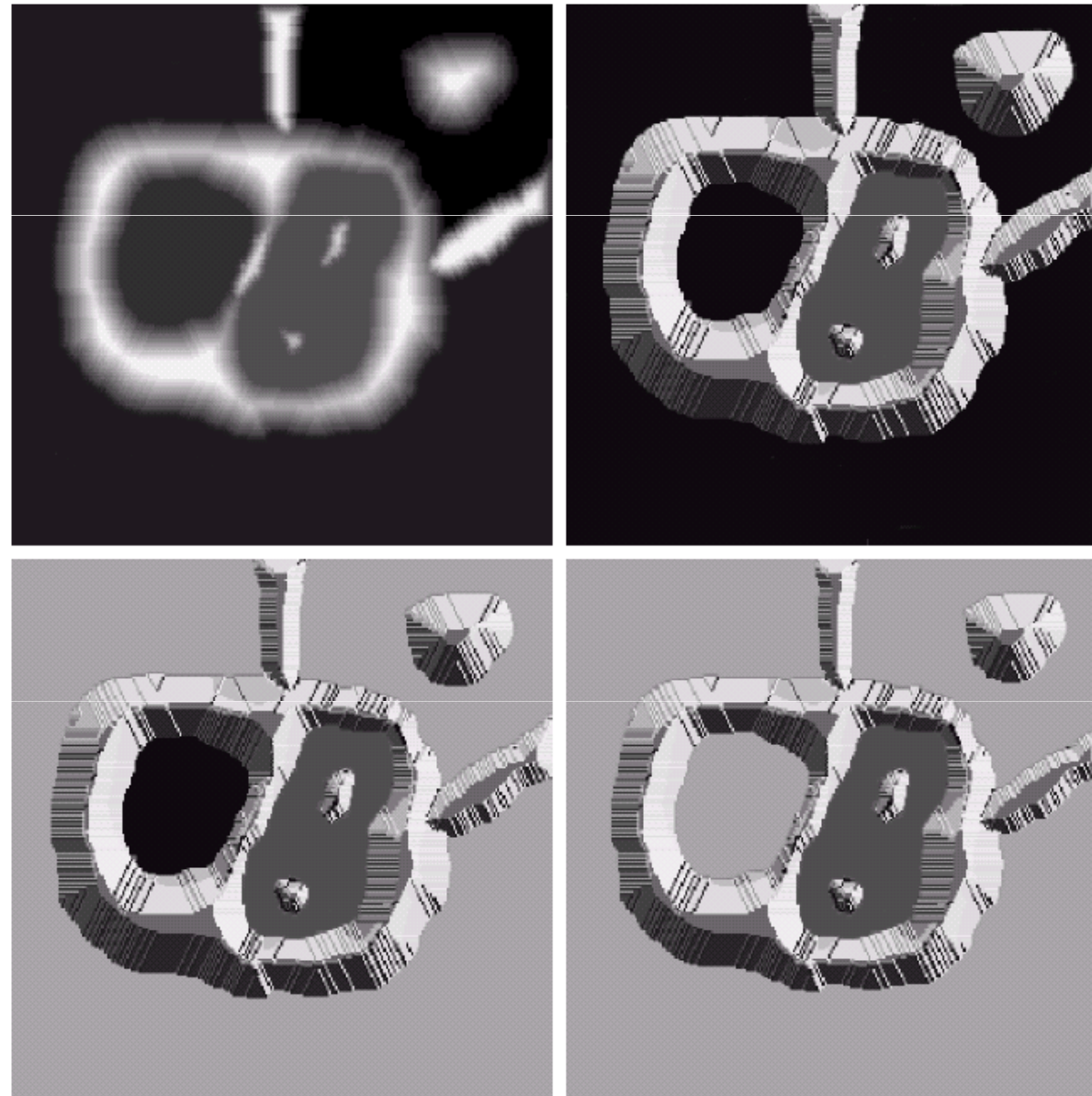
- Assume that there is a hole in each minima and the surface is immersed into a lake.
- The water will enter through the holes at the minima and flood the surface.
- To avoid the water coming from two different minima to meet, a dam is build whenever there would be a merge of the water.
- Finally, the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

Watershed segmentation

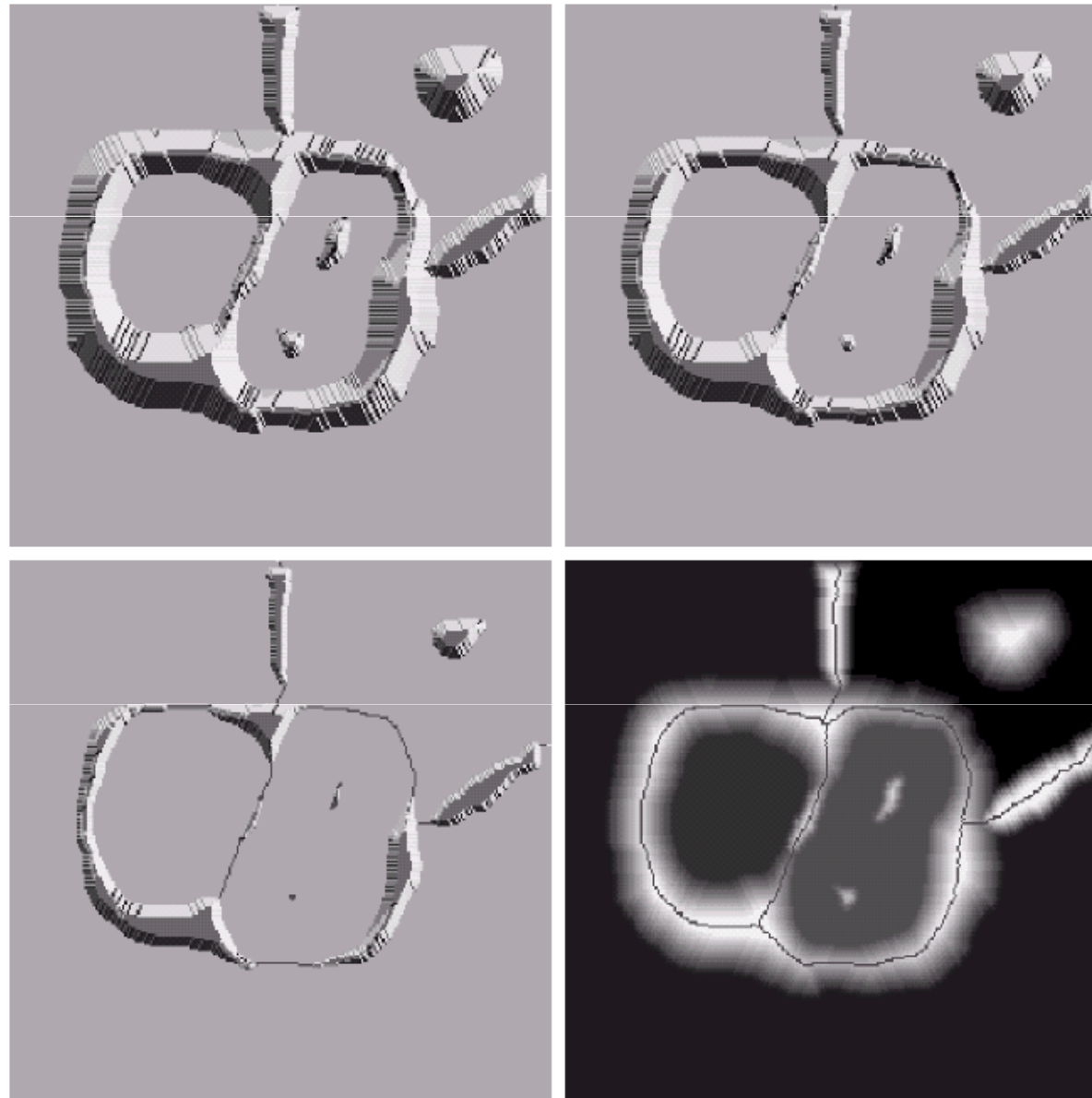
a b
c d

FIGURE 10.44

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



Watershed segmentation



e f
g h

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

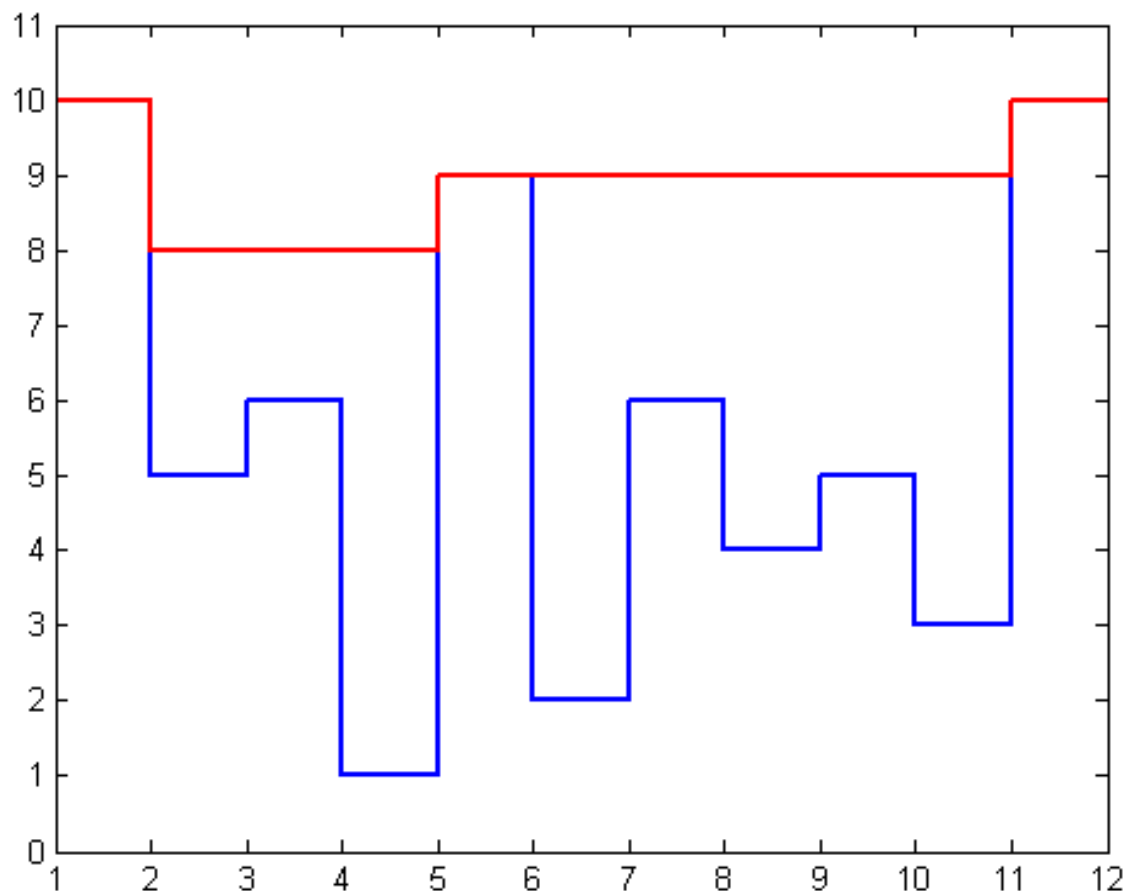
Dr. S. Beucher,

CMM/Ecole des

Mines de Paris.)

Watershed segmentation

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



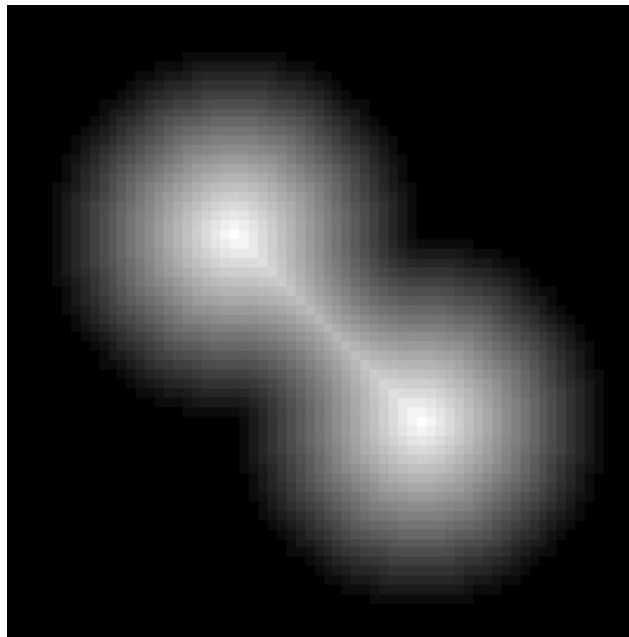
Watershed segmentation

- 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.

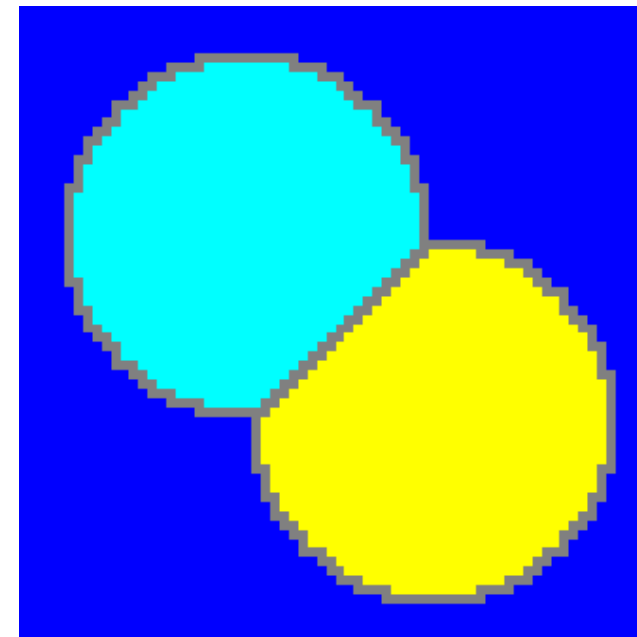
Watershed segmentation



Binary image.

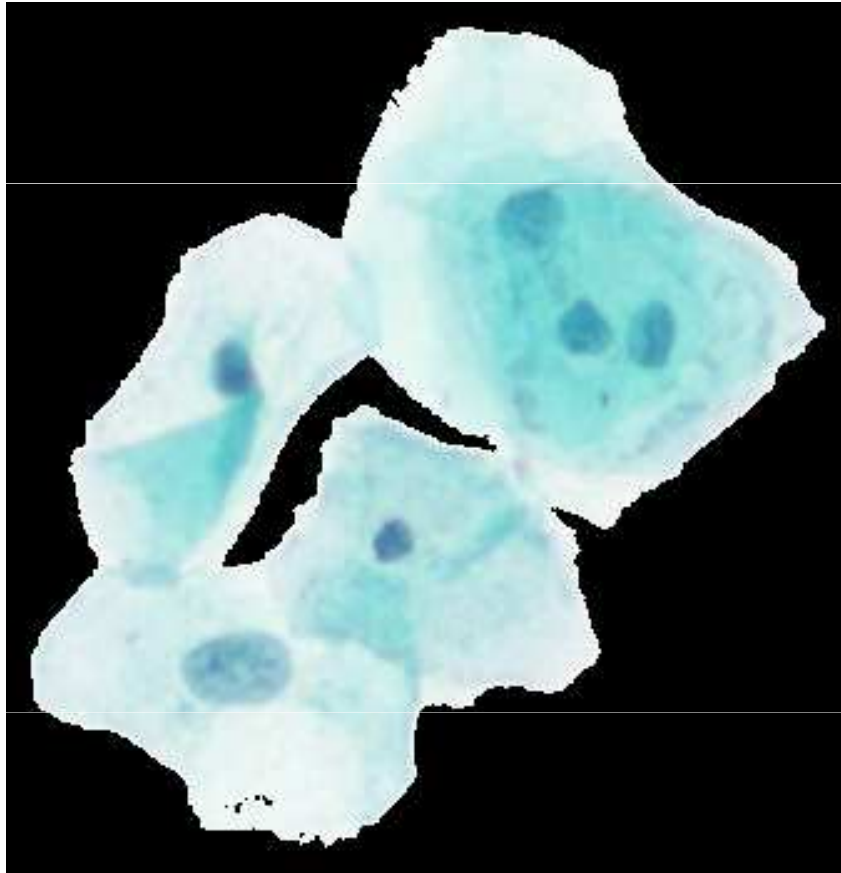


Distance transform of the complement of the binary image.

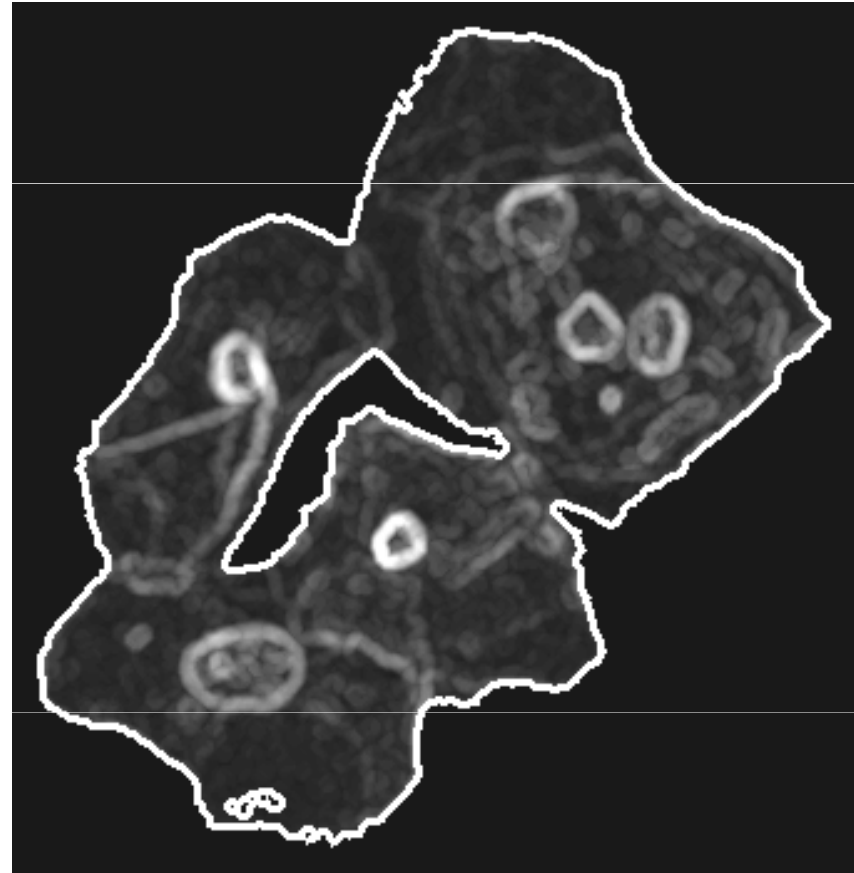


Watershed transform after complementing the distance transform, and forcing pixels that do not belong to the objects to be at $-\text{Inf}$.

Watershed segmentation

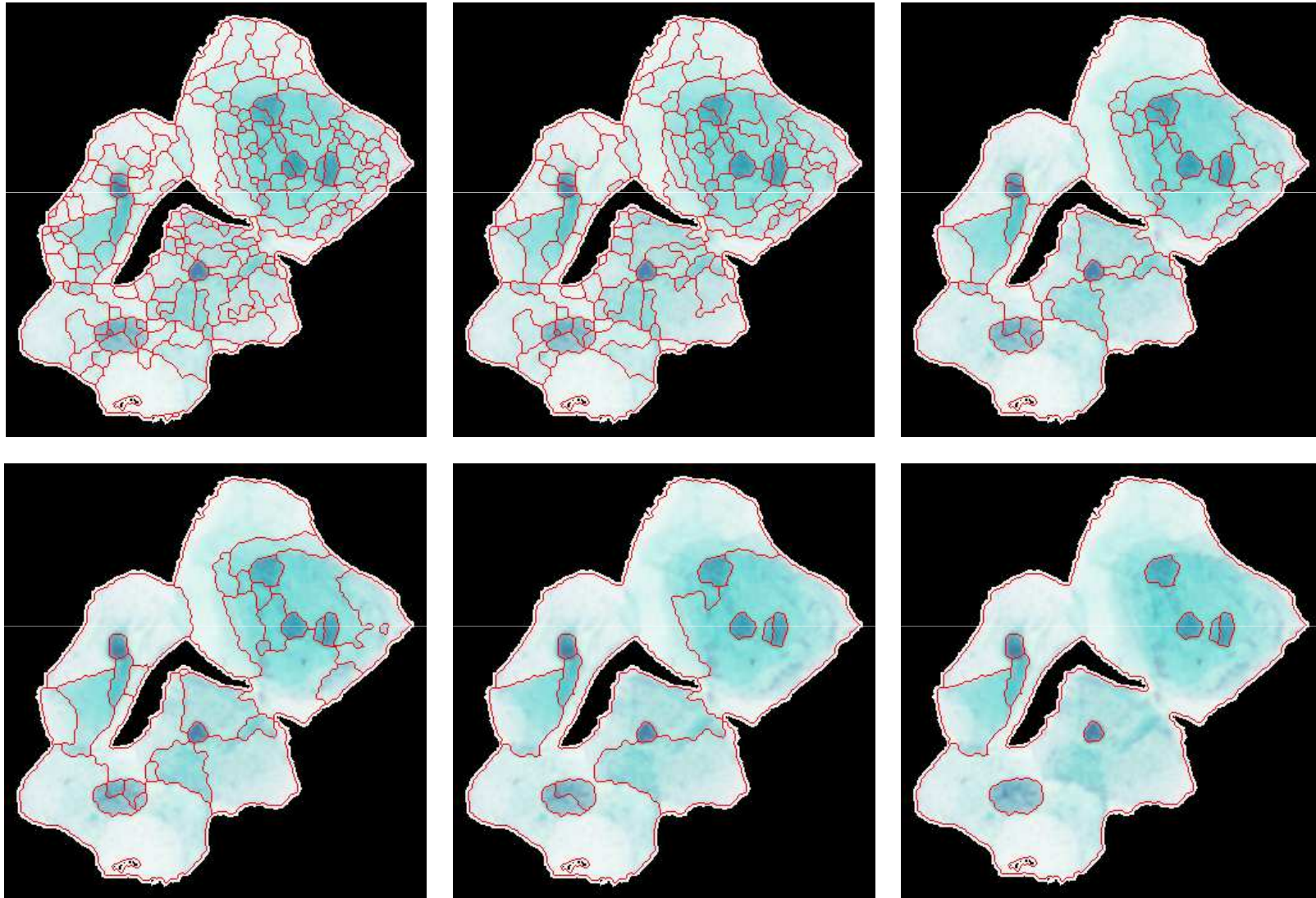


A cell image.



Gradient of the cell image.

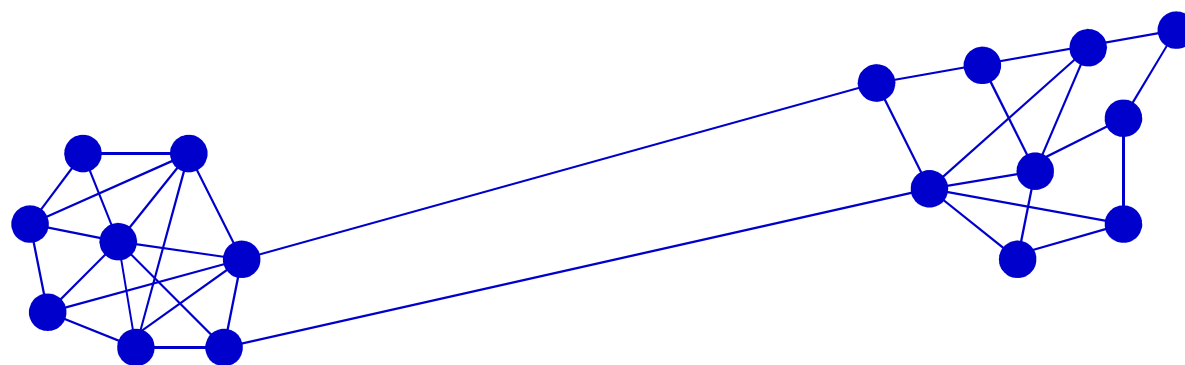
Watershed segmentation



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.

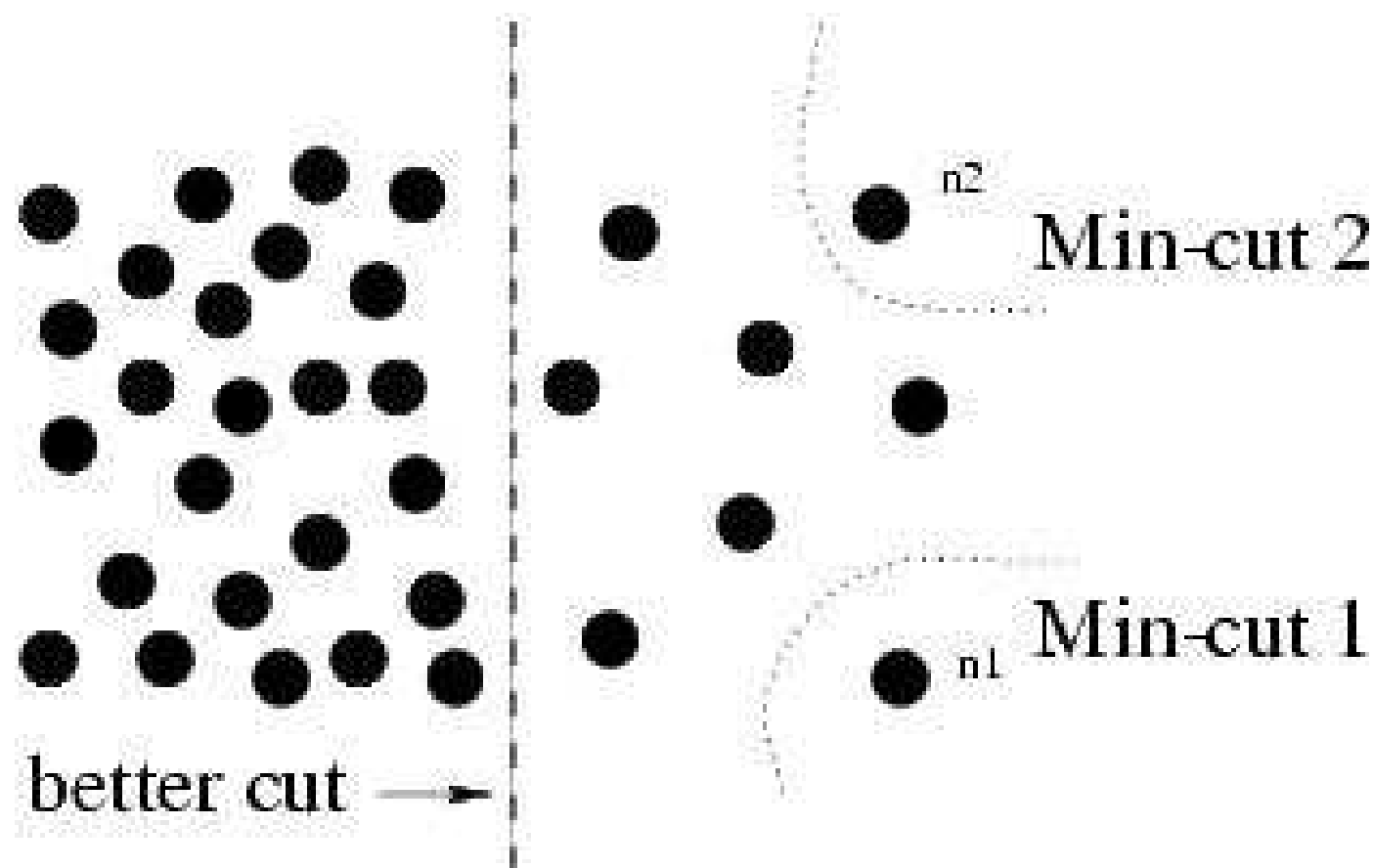


Graph-based segmentation

- 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 $\text{cut}(A, B) = \sum_{u \in A, v \in B} w(u, v)$.
- 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**.

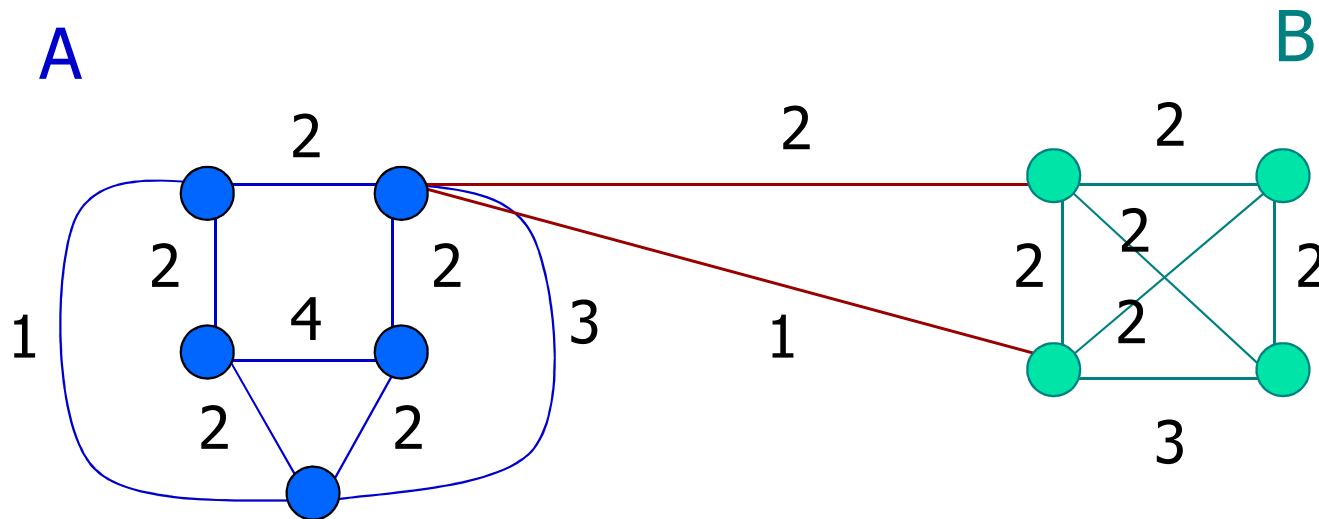
$$\text{Ncut}(A,B) = \frac{\text{cut}(A,B)}{\text{assoc}(A,V)} + \frac{\text{cut}(A,B)}{\text{assoc}(B,V)}$$

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



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

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.
 - $N \times N$ matrix D : D is a diagonal matrix with d on its diagonal.
 - $N \times N$ 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 ($y \rightarrow x \rightarrow A$).
- If further subdivision is merited, repeat recursively.

Graph-based segmentation

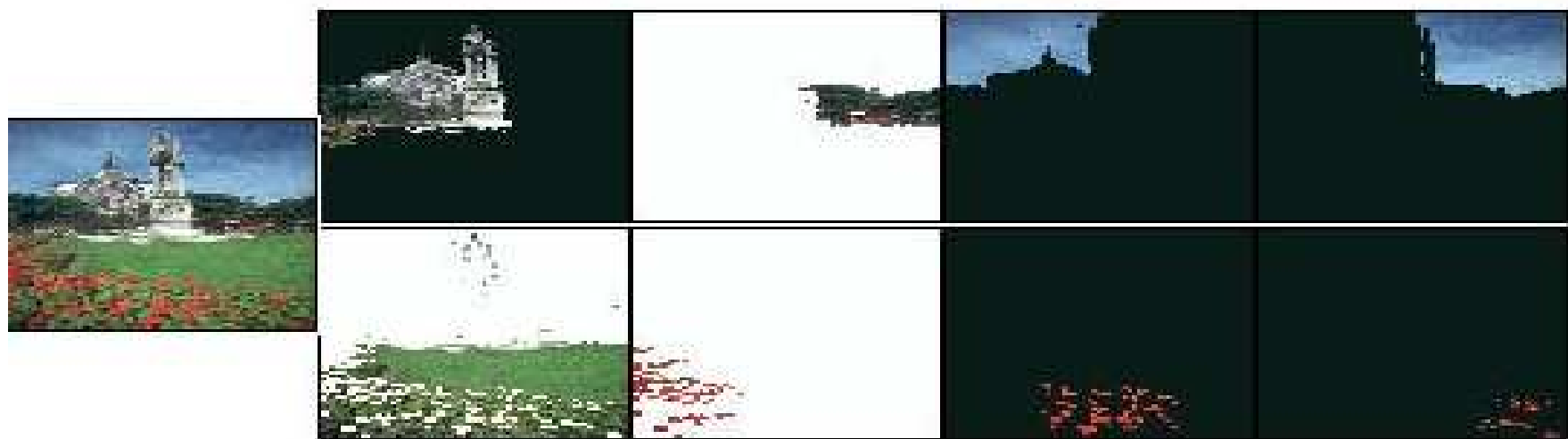
- Edge weights $w(i,j)$ can be defined by

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

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 → many affinity computations
- Solving eigenvalue problem