Image Segmentation

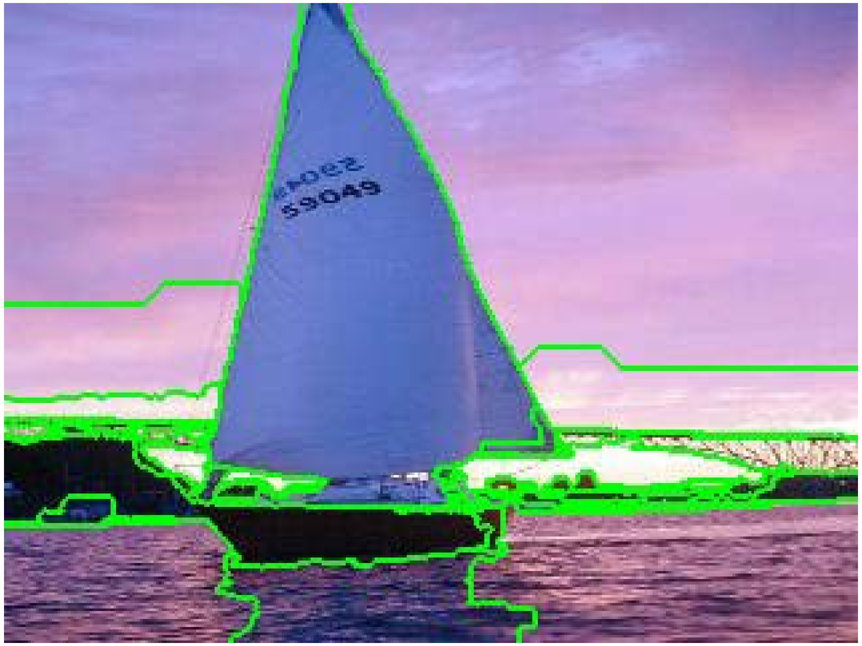
Image segmentation is the operation of partitioning an image into a collection of connected sets of pixels.

1. into regions, which usually cover the image

2. into linear structures, such as
    - line segments
    - curve segments

3. into 2D shapes, such as
    - circles
    - ellipses
    - ribbons (long, symmetric regions)

Example 1: Regions



Example 2:
Lines and Circular

Main Methods of Region Segmentation

1. ~~Region Growing~~
2. ~~Split and Merge~~
3. Clustering

Clustering

- There are K clusters $C_1, \ldots, C_K$ with means $m_1, \ldots, m_K$.

- The **least-squares error** is defined as

$$D = \sum_{k=1}^{K} \sum_{x_i \in C_k} \| x_i - m_k \|^2 .$$

- Out of all possible partitions into K clusters, choose the one that minimizes D.

Why don't we just do this?
If we could, would we get meaningful objects?

K-Means Clustering

Form K-means clusters from a set of n-dimensional vectors

1. Set ic (iteration count) to 1

2. Choose randomly a set of K means $m_1(1)$, ..., $m_K(1)$.

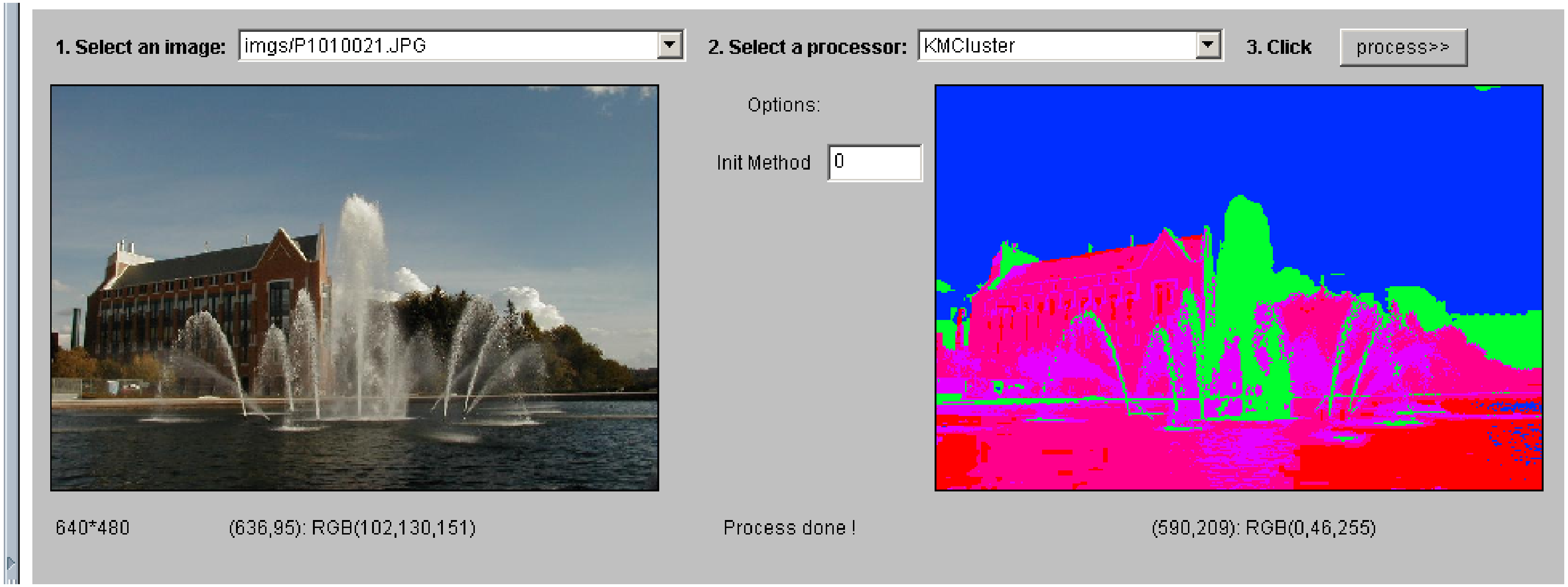
3. For each vector x_i compute $D(x_i , m_k(ic))$, k=1,...K and assign x_i to the cluster C_j with nearest mean.

4. Increment ic by 1, update the means to get $m_1(ic)$,...,$m_K(ic)$.

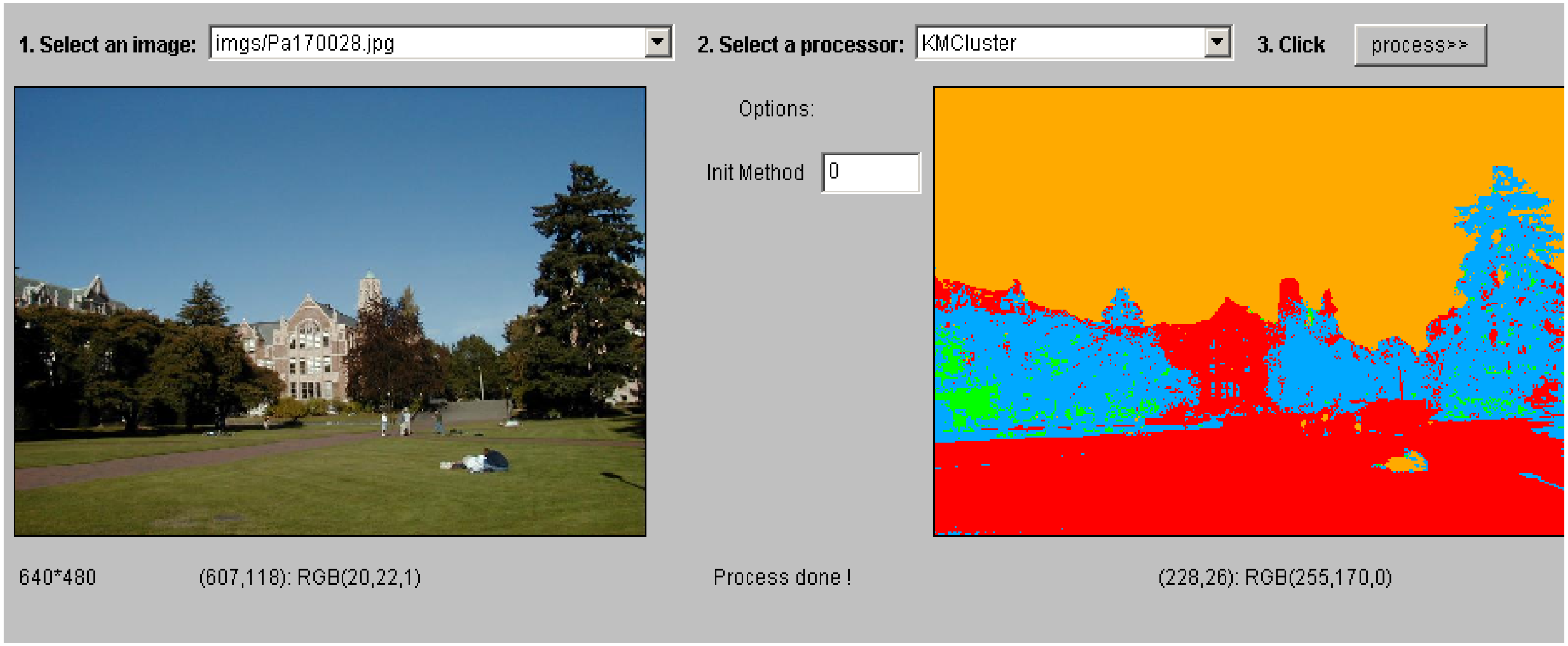
5. Repeat steps 3 and 4 until $C_k(ic) = C_k(ic+1)$ for all k.

K-Means Example 1

K-Means Example 2



K-Means Example 3



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

- The EM Algorithm: a probabilistic formulation

K-Means

- <u>Boot Step</u>:
  - Initialize *K* clusters: $C_1, \ldots, C_K$

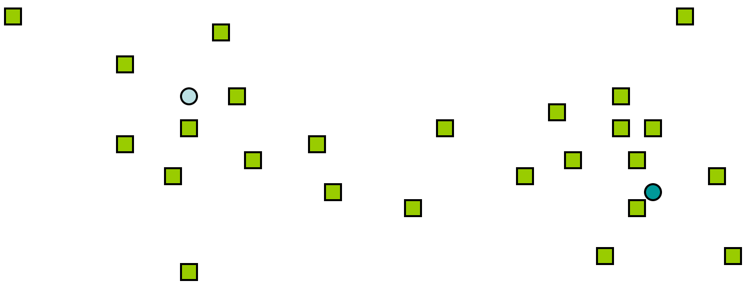
    Each cluster is represented by its mean m_j
- <u>Iteration Step</u>:
  - Estimate the cluster for each data point

$$x_i \Longrightarrow C(x_i)$$

  - Re-estimate the cluster parameters

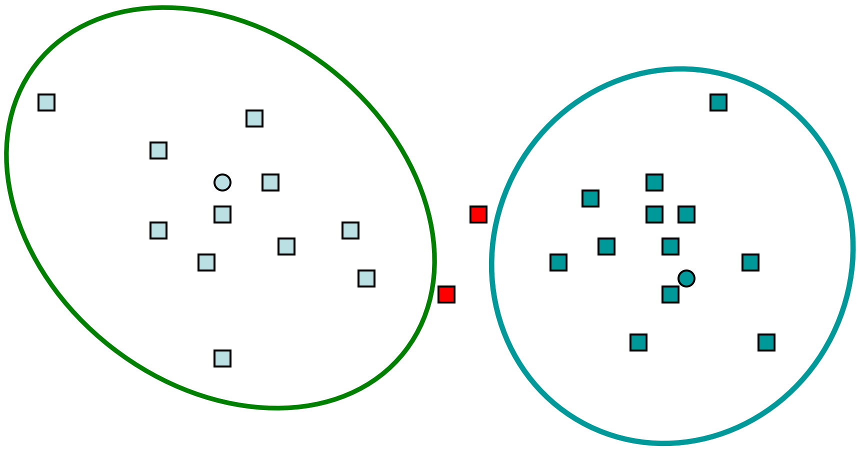
$$m_j = mean\{x_i \mid x_i \in C_j\}$$

K-Means Example



K-Means Example

Where do the red points belong?



K-Means → EM

- Boot Step:
  - Initialize K clusters: $C_1, \ldots, C_K$

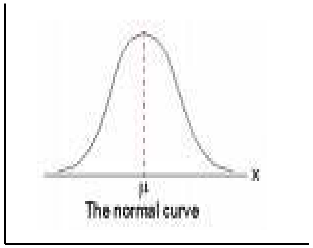
    (μ_j, Σ_j) and $P(C_j)$ for each cluster j.
- Iteration Step:
  - Estimate the cluster of each data point

    $p(C_j \mid x_i)$ ➡ Expectation
  - Re-estimate the cluster parameters

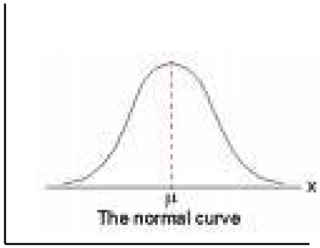
    $(\mu_j, \Sigma_j), p(C_j)$ For each cluster j ➡ Maximization

1-D EM with Gaussian Distributions

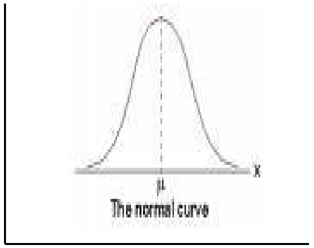
- Each cluster C_j is represented by a Gaussian distribution $N(\mu_j , \sigma_j)$.
- Initialization: For each cluster C_j initialize its mean μ_j , variance σ_j, and weight α_j.



$N(\mu_1 , \sigma_1)$
$\alpha_1 = P(C_1)$



$N(\mu_2 , \sigma_2)$
$\alpha_2 = P(C_2)$



$N(\mu_3 , \sigma_3)$
$\alpha_3 = P(C_3)$

Expectation

- For each point x_i and each cluster C_j compute $P(C_j \mid x_i)$.

- $P(C_j \mid x_i) = P(x_i \mid C_j)\, P(C_j) \,/\, P(x_i)$

- $P(x_i) = \sum_j P(x_i \mid C_j)\, P(C_j)$

- Where do we get $P(x_i \mid C_j)$ and $P(C_j)$?

1. Use the pdf for a normal distribution:

$$P(x_i \mid C_j) = \frac{1}{\sqrt{2\pi}\,\sigma_j} e^{-\frac{(x_i - \mu_j)^2}{2\sigma_j^2}}$$

2. Use $\alpha_j = P(C_j)$ from the current parameters of cluster C_j.

Maximization

- Having computed $P(C_j \mid x_i)$ for each point x_i and each cluster C_j, use them to compute new mean, variance, and weight for each cluster.

$$\mu_j = \frac{\sum_i p(C_j \mid x_i) \cdot x_i}{\sum_i p(C_j \mid x_i)}$$

$$\Sigma_j = \frac{\sum_i p(C_j \mid x_i) \cdot (x_i - \mu_j) \cdot (x_i - \mu_j)^T}{\sum_i p(C_j \mid x_i)}$$

$$p(C_j) = \frac{\sum_i p(C_j \mid x_i)}{N}$$

Multi-Dimensional Expectation Step for Color Image Segmentation

Input (Known)

$x_1=\{r_1, g_1, b_1\}$

$x_2=\{r_2, g_2, b_2\}$

…

$x_i=\{r_i, g_i, b_i\}$

…

\+

Input (Estimation)

Cluster Parameters

(μ_1,Σ_1), $p(C_1)$ for C_1

(μ_2,Σ_2), $p(C_2)$ for C_2

…

(μ_k,Σ_k), $p(C_k)$ for C_k

⇒

Output

Classification Results

$p(C_1|x_1)$

$p(C_j|x_2)$

…

$p(C_j|x_i)$

…

$$p(C_j \mid x_i) = \frac{p(x_i \mid C_j)\cdot p(C_j)}{p(x_i)} = \frac{p(x_i \mid C_j)\cdot p(C_j)}{\sum_j p(x_i \mid C_j)\cdot p(C_j)}$$

Multi-dimensional Maximization Step for Color Image Segmentation

Input (Known)

$x_1=\{r_1, g_1, b_1\}$

$x_2=\{r_2, g_2, b_2\}$

…

$x_i=\{r_i, g_i, b_i\}$

…

+

Input (Estimation)

Classification Results

$p(C_1|x_1)$

$p(C_j|x_2)$

…

$p(C_j|x_i)$

…

Output

Cluster Parameters

(μ_1,Σ_1), $p(C_1)$ for C_1

(μ_2,Σ_2), $p(C_2)$ for C_2

…

(μ_k,Σ_k), $p(C_k)$ for C_k

$$\mu_j = \frac{\sum_i p(C_j | x_i)\cdot x_i}{\sum_i p(C_j | x_i)} \quad \Sigma_j = \frac{\sum_i p(C_j | x_i)\cdot (x_i - \mu_j)\cdot (x_i - \mu_j)^T}{\sum_i p(C_j | x_i)} \quad p(C_j) = \frac{\sum_i p(C_j | x_i)}{N}$$

Full EM Algorithm Multi-Dimensional

- <u>Boot Step</u>:
  - Initialize K clusters: $C_1, \ldots, C_K$

    (μ_j, Σ_j) and $P(C_j)$ for each cluster j.

- <u>Iteration Step</u>:
  - Expectation Step

$$p(C_j \mid x_i) = \frac{p(x_i \mid C_j) \cdot p(C_j)}{p(x_i)} = \frac{p(x_i \mid C_j) \cdot p(C_j)}{\sum_j p(x_i \mid C_j) \cdot p(C_j)}$$

  - Maximization Step

$$\mu_j = \frac{\sum_i p(C_j \mid x_i) \cdot x_i}{\sum_i p(C_j \mid x_i)} \qquad \Sigma_j = \frac{\sum_i p(C_j \mid x_i) \cdot (x_i - \mu_j) \cdot (x_i - \mu_j)^T}{\sum_i p(C_j \mid x_i)} \qquad p(C_j) = \frac{\sum_i p(C_j \mid x_i)}{N}$$

EM Demo

- Demo

  http://www.neurosci.aist.go.jp/~akaho/MixtureEM.html

- Example

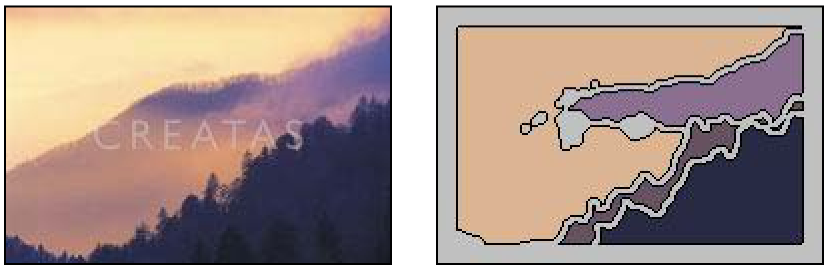
  http://www-2.cs.cmu.edu/~awm/tutorials/gmm13.pdf

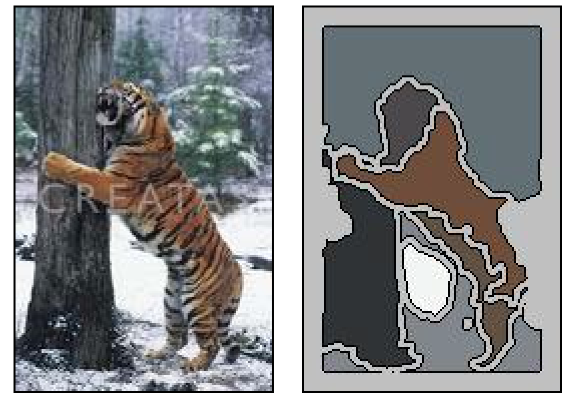
EM Applications

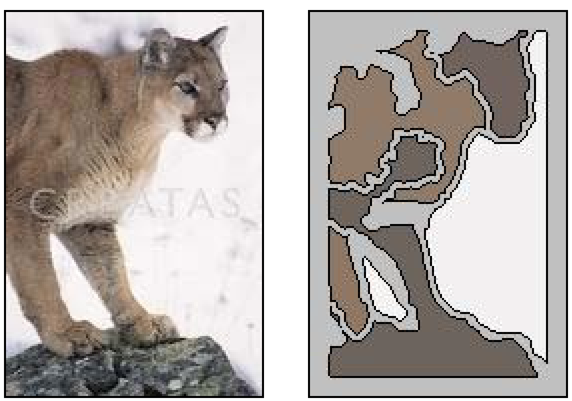
- Blobworld: Image segmentation using Expectation-Maximization and its application to image querying

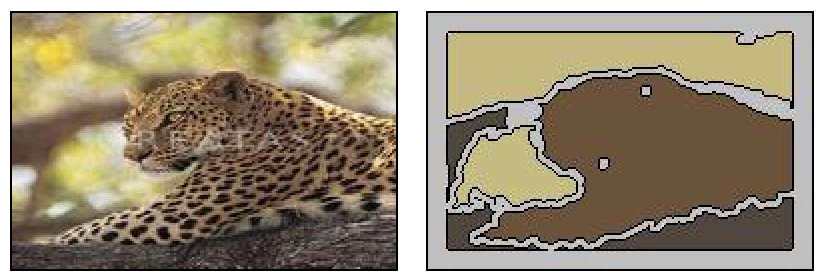
- Yi’s Generative/Discriminative Learning of object classes in color images

Blobworld: Sample Results



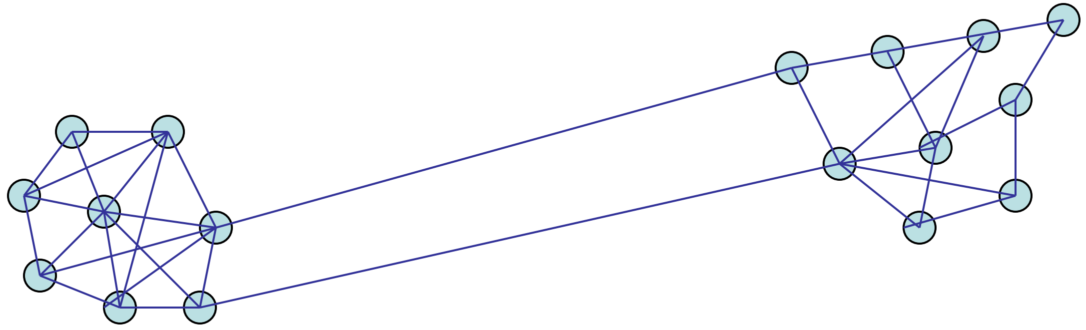






Jianbo Shi's Graph-Partitioning

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

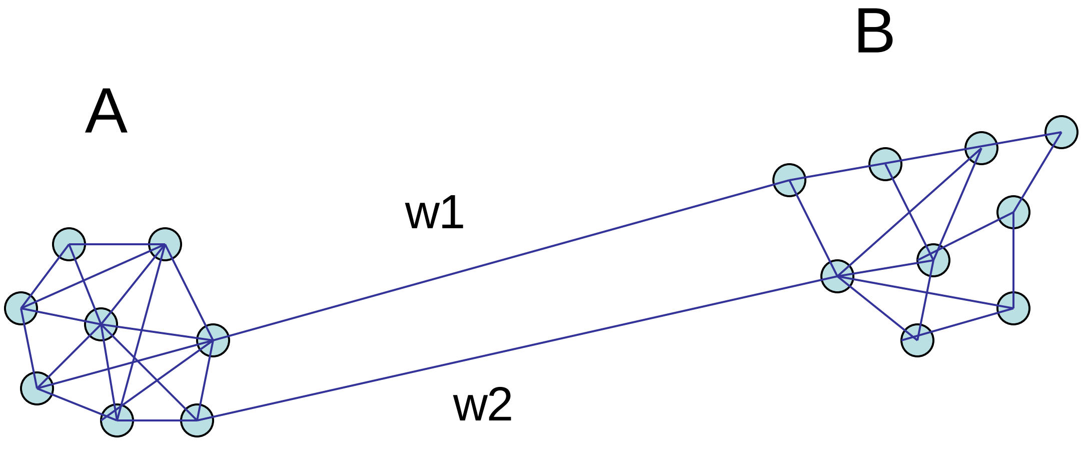


Minimal Cuts

- 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 $cut(A,B) = \sum_{u \varepsilon A,\, v \varepsilon B} w(u,v)$.
- One way to segment G is to find the minimal cut.

Cut(A,B)

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



Normalized Cut

Minimal cut favors cutting off small node groups, so Shi proposed the **normalized cut.**

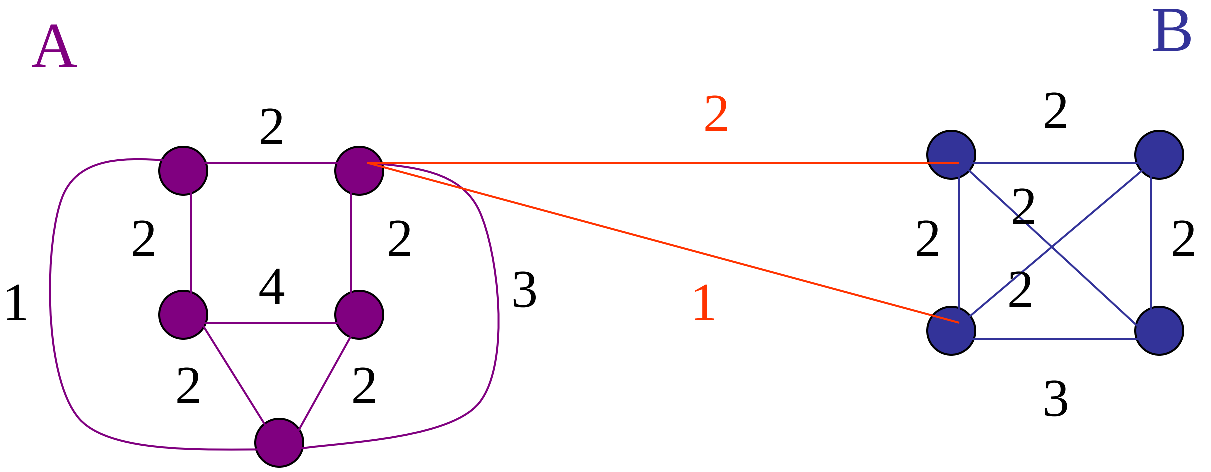
$$\mathrm{Ncut}(A,B) = \frac{\mathrm{cut}(A, B)}{\mathrm{asso}(A,V)} + \frac{\mathrm{cut}(A,B)}{\mathrm{asso}(B,V)}$$

normalized cut

$$\mathrm{asso}(A,V) = \sum_{u \in A,\, t \in V} w(u,t)$$

How much is A connected to the graph as a whole.

Example Normalized Cut



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

Shi 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: $W_{ij} = w_{ij}$

- 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 = λ 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

How Shi used the procedure

Shi defined the edge weights w(i,j) by

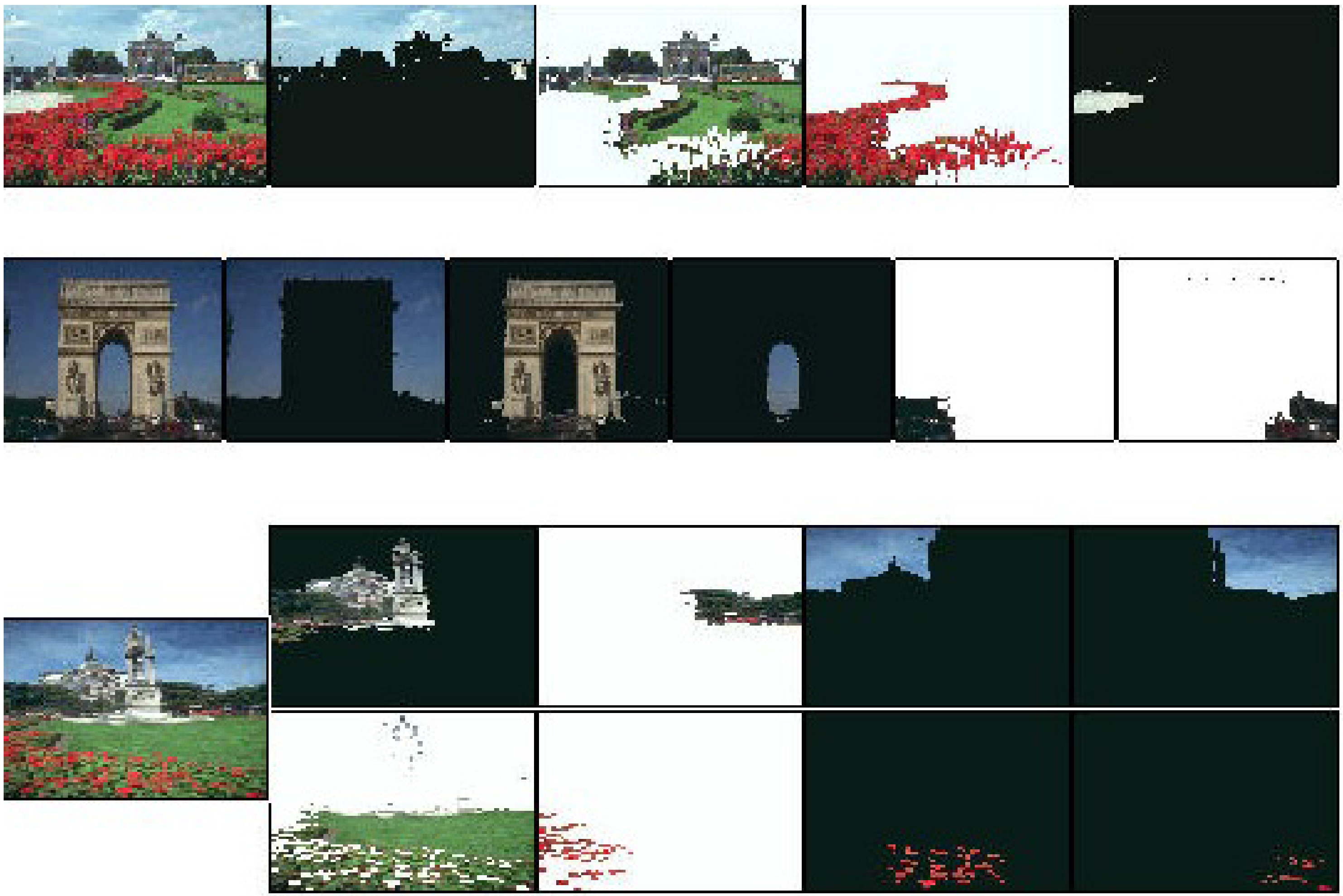
$$w(i,j) = e^{-\|F(i)-F(j)\|_2 / \sigma I} * \begin{cases} e^{-\|X(i)-X(j)\|_2 / \sigma X} & \text{if } \|X(i)-X(j)\|_2 < r \\ 0 & \text{otherwise} \end{cases}$$

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.

Examples of Shi Clustering

See Shi's Web Page
http://www.cis.upenn.edu/~jshi/



Problems with Graph Cuts

- Need to know when to stop
- Very Slooooow

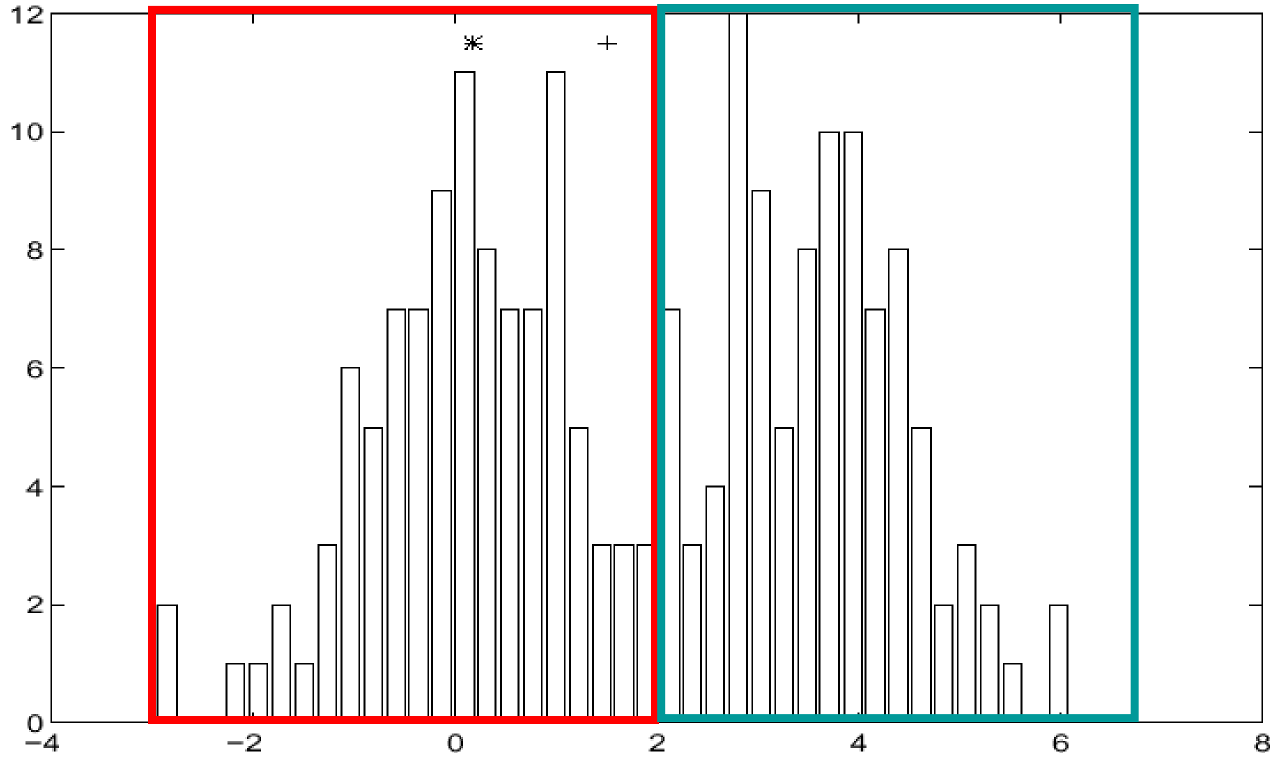
Problems with EM

- Local minima
- Need to know number of segments
- Need to choose generative model

Mean-Shift Clustering

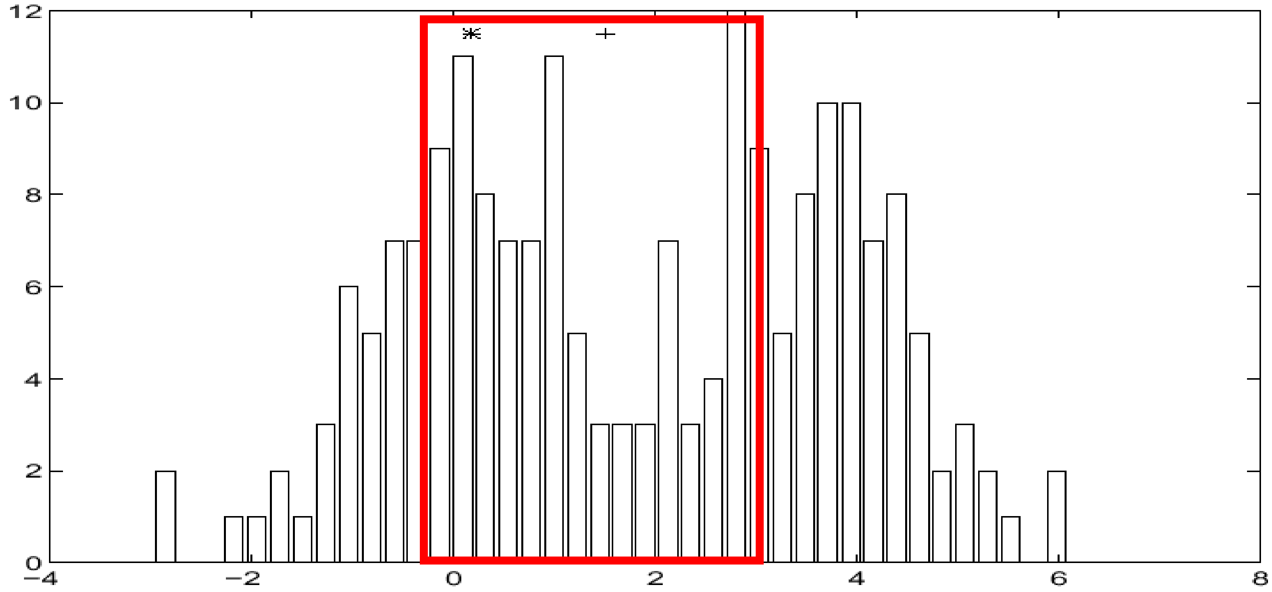
- Simple, like K-means
- But you don't have to select K
- Statistical method
- Guaranteed to converge to a fixed number of clusters.

Finding Modes in a Histogram



- How Many Modes Are There?
  - Easy to see, hard to compute

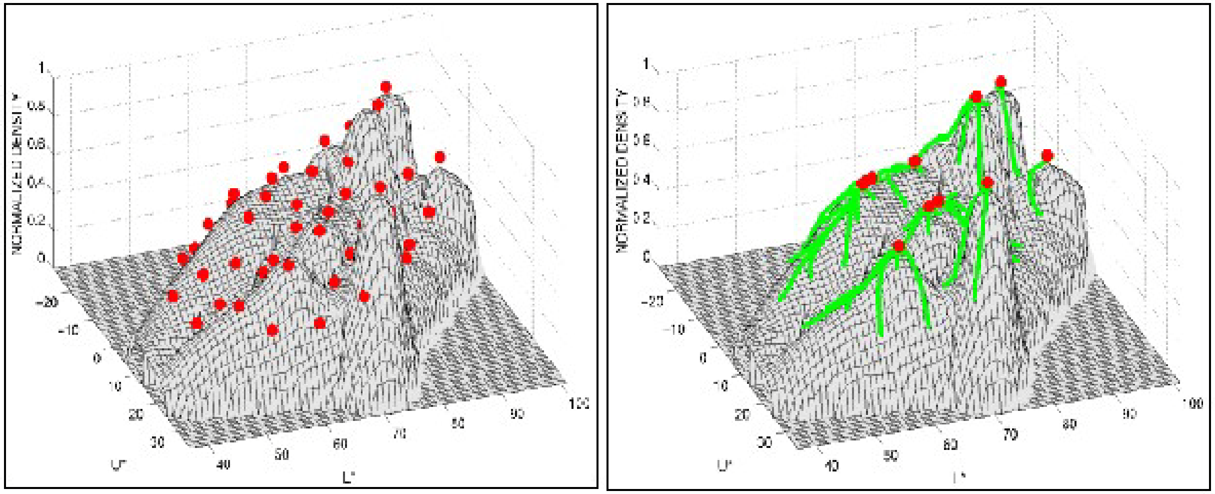
Mean Shift [Comaniciu & Meer]



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

Mean Shift Approach

- Initialize a window around each point
- See where it shifts—this determines which segment it's in
- Multiple points will shift to the same segment



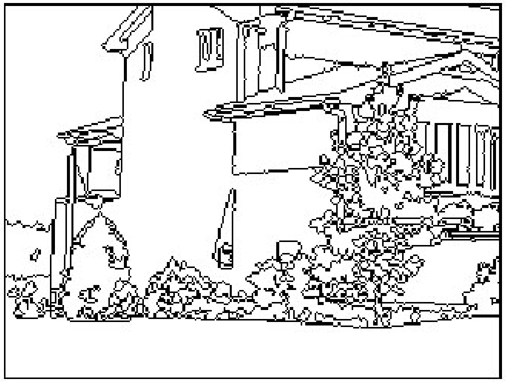
Mean shift trajectories

Segmentation Algorithm

- First run the mean shift procedure for each data point x and store its convergence point z.
- Link together all the z's that are closer than .5 from each other to form clusters
- Assign each point to its cluster
- Eliminate small regions

Mean-shift for image segmentation

- Useful to take into account spatial information
  - instead of (R, G, B), run in (R, G, B, x, y) space

References

– Shi and Malik, "Normalized Cuts and Image Segmentation," Proc. CVPR 1997.

– Carson, Belongie, Greenspan and Malik, "Blobworld: Image Segmentation Using Expectation-Maximization and its Application to Image Querying," IEEE PAMI, Vol 24, No. 8, Aug. 2002.

– Comaniciu and Meer, "Mean shift analysis and applications," Proc. *ICCV* 1999.