Lecture 4: Image Segmentation

Special Topics in Image Processing

jcchun@kgu.ac.kr

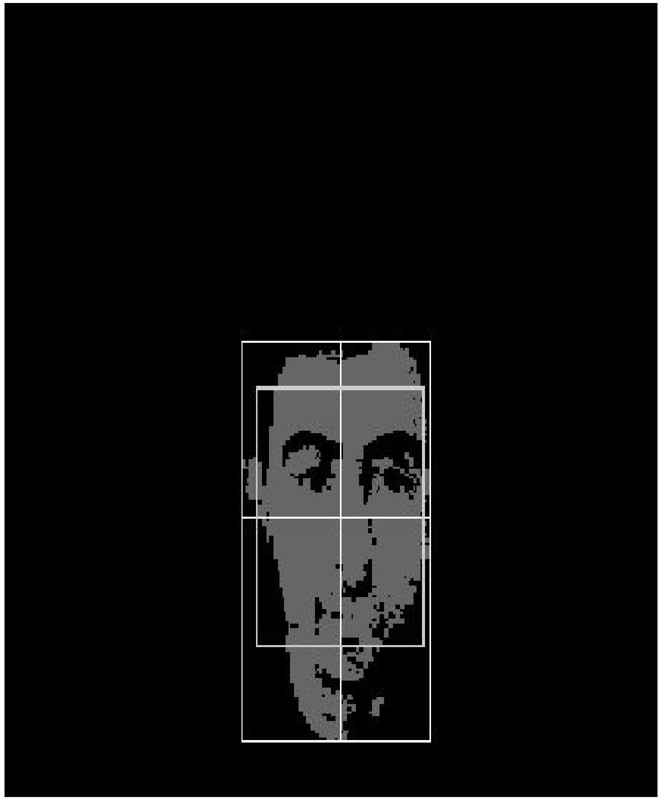
Image Segmentation

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

- into regions, which usually cover the image
- into linear structures, such as
  - line segments
  - curve segments
- into 2D shapes, such as
  - circles
  - ellipses
  - ribbons (long, symmetric regions)

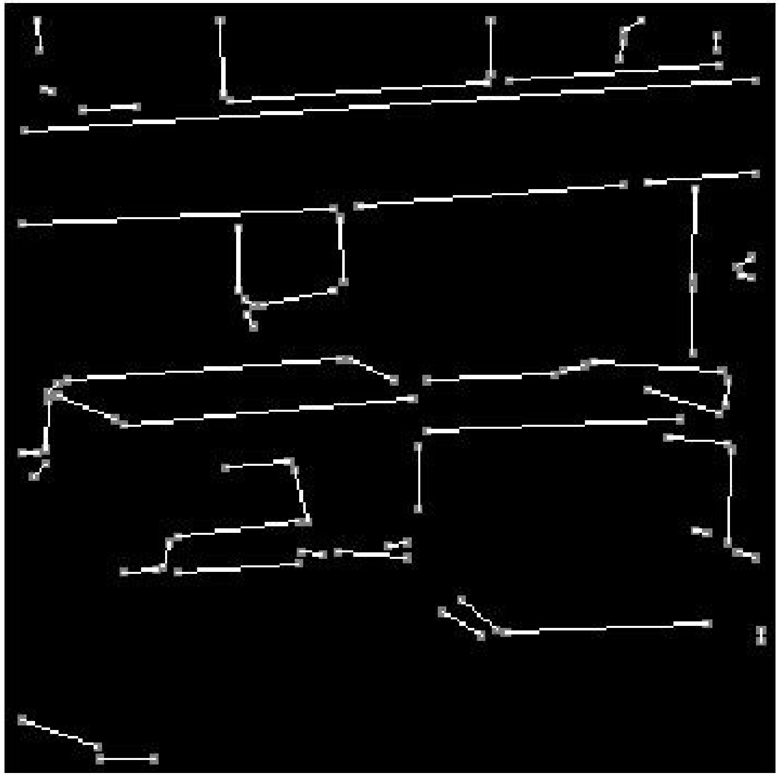
Segmenting an Image: Regions

1. Want "meaningful" regions of image
2. Objects often differ in appearance
3. We change representation from pixels to regions
4. Can there be quality segmentation without recognition?



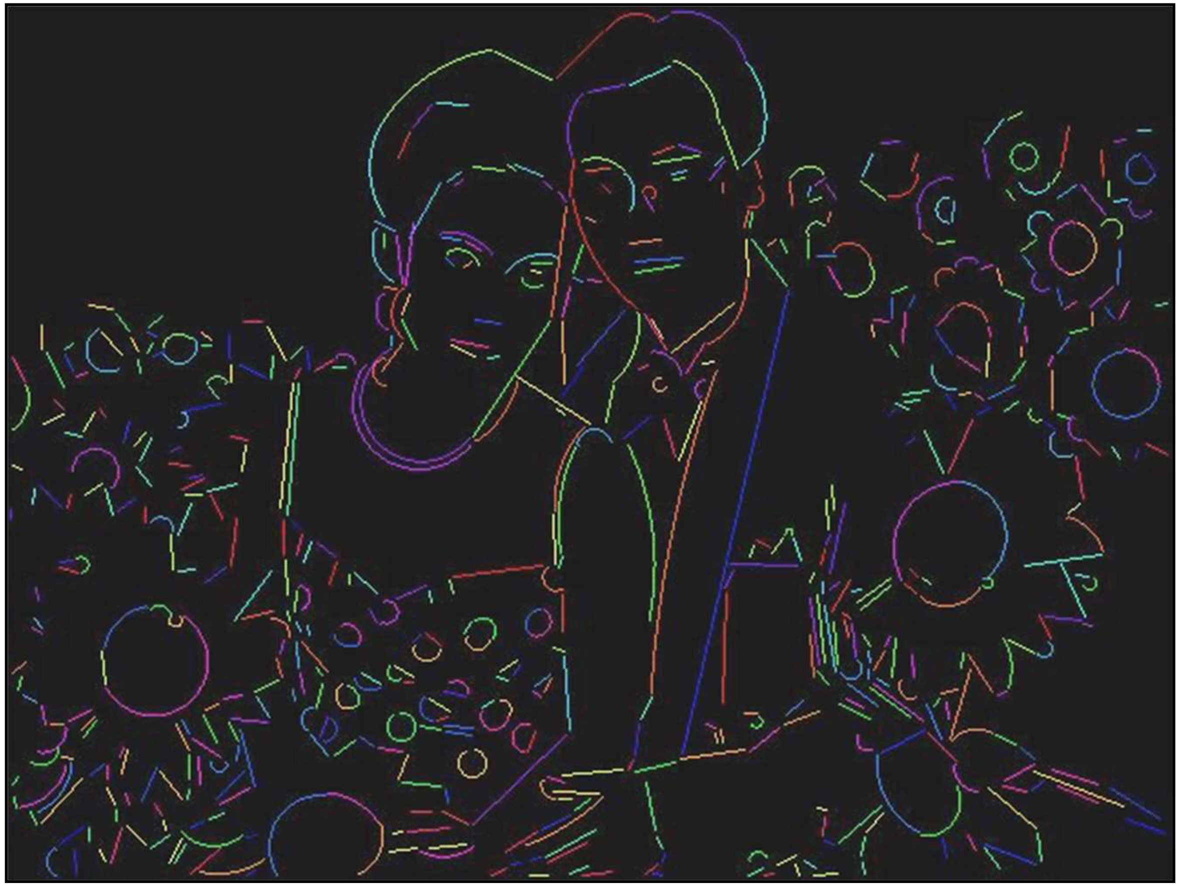
Region versus Border Segmentation

- Example: Two segmentations of blocks scene



Example 1: Regions

Example 2: Lines and Circular Arcs



Region Segmentation

Region Segmentation

- Find sets of pixels, $R_1, R_2, \ldots, R_n$ such that
  - $\bigcup_i R_i = I$
  - $\forall i \neq j, R_i \cap R_j = \varnothing$
  - All pixels in region *i* satisfy some constraint of similarity.

General segmentation algorithms

- Collect pixels into regions
  - by adjacency in space (neighbors)
  - by similarity in color/texture
  - by adjacency and similarity of motion
- General algorithms can be tested quickly in applications.
- General algorithms should not be expected to work well in all cases.

Segmentation by boundaries

- Separate regions by differencing.
- Strong edges result between different regions.
- Very useful when boundaries have simple shape.
- Pursue boundary computation in separate presentation.

Clustering versus Region Growing

- Region growing: adjacency is major control, pixel similarity is secondary
  - start at some location[s] (seeds)
  - only add adjacent pixels that are similar (~ painting algorithm)
  - can grow multiple regions in parallel and in competition

Clustering versus Region Growing

- Clustering: Similarity of features is primary control, adjacency is secondary.

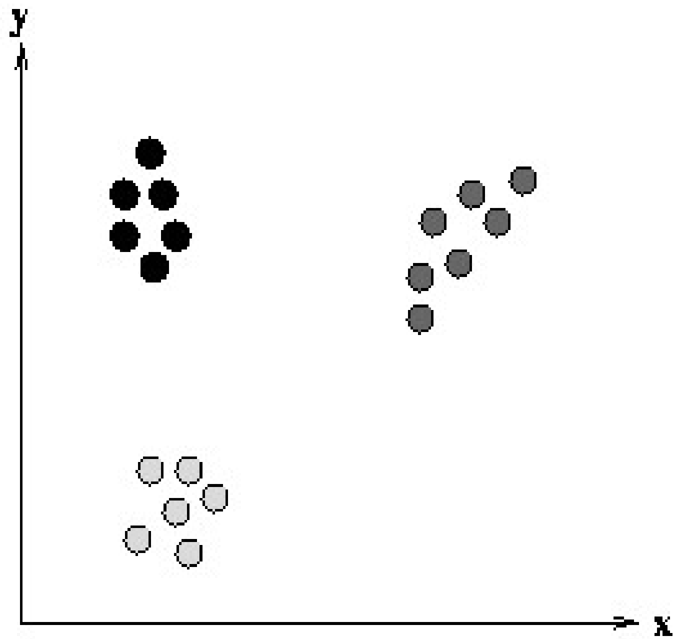
  - each pixel represented by n-D vector
  - clustering partitions all M x N pixels into K classes
  - connected components performed after clustering

Clustering versus Region Growing

- Region forming concepts also apply to boundary segmentation
  - “boundary following uses adjacency as major control
  - Hough transform uses spatial similarity as major control

Simple Example of Clusters in Feature Space

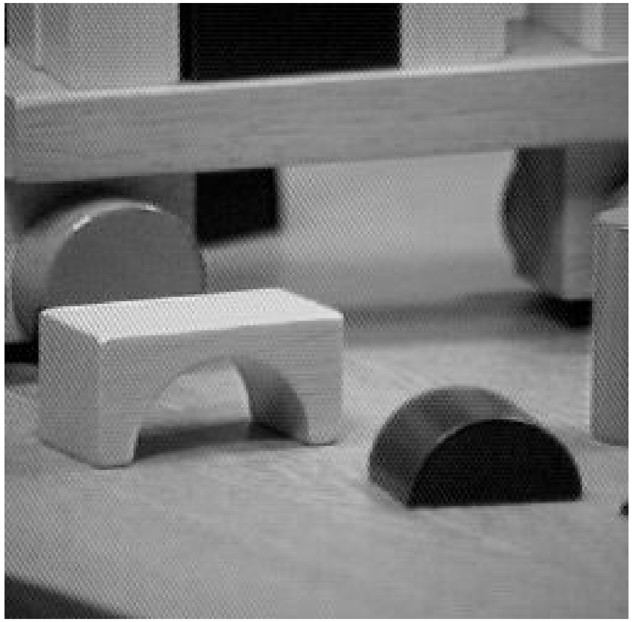
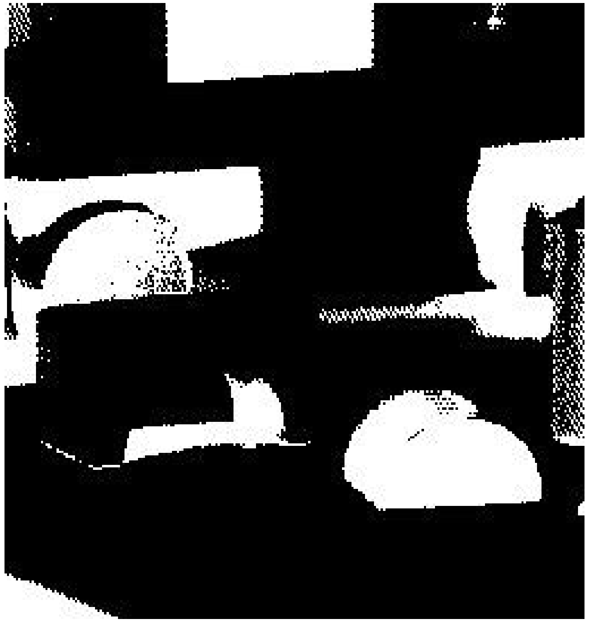
- Set of features that can be separated into 3 clusters.
- Each cluster contains pixels that are in some sense similar
- Feature space can have many dimensions.



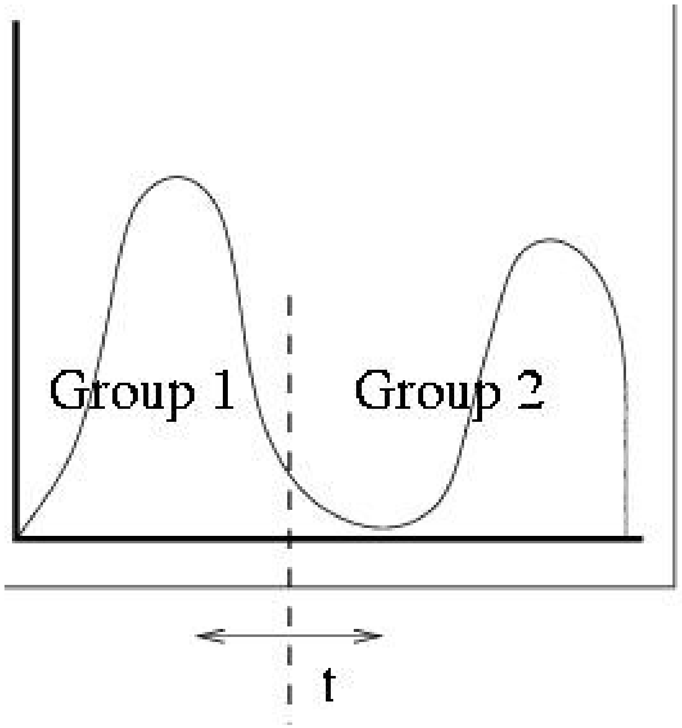
- Obtaining meaningful k-D vectors representing pixels
  - 3 color / intensity values at [r,c]
  - Can get histogram of neighborhood colors
  - Use LOG filter values on neighborhood
  - Use FFT on neighborhood
  - Project neighborhood onto Laws texture basis (masks)

Histogram-based methods

- Left: intensity range 31-100 made white
- Right: intensity range 101-179 made white


Otsu's automatic threshold method



1. assumes $K = 2$ regions: *objects* versus *background*

2. search over thresholds t

3. choose t that gives minimum within group variance

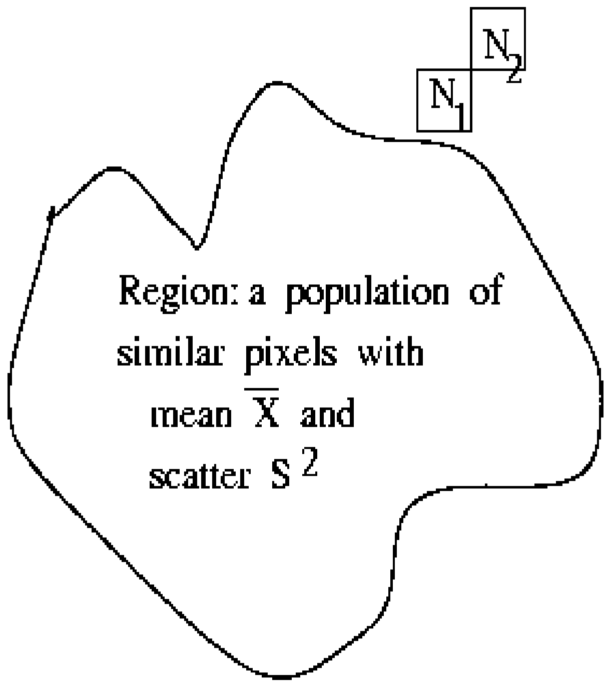
Optimize on: $\sigma_W^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t)$

where $q_1(t)$ is the number of pixels with property $< t$,

and $q_2(t)$ is the number of pixels with property $\geq t$,

Region-growing(painting)

- Region is a population with similar statistics
- Use statistical test to see if neighbor on border fits into the region population, if so, update region and stats
- If neighbor doesn't fit, start another region there.



- Deciding to grow and updating

$$\overline{X} = \frac{1}{N} \sum_{[r,c] \in R} I[r, c]$$

$$\overline{X}_{\text{new}} \leftarrow (N\overline{X}_{\text{old}} + y)/(N + 1)$$

$$S^2 = \sum_{[r,c] \in R} (I[r, c] - \overline{X})^2.$$

$$S^2_{\text{new}} \leftarrow S^2_{\text{old}} + (y - \overline{X}_{\text{new}})^2 + N(\overline{X}_{\text{new}} - \overline{X}_{\text{old}})^2$$

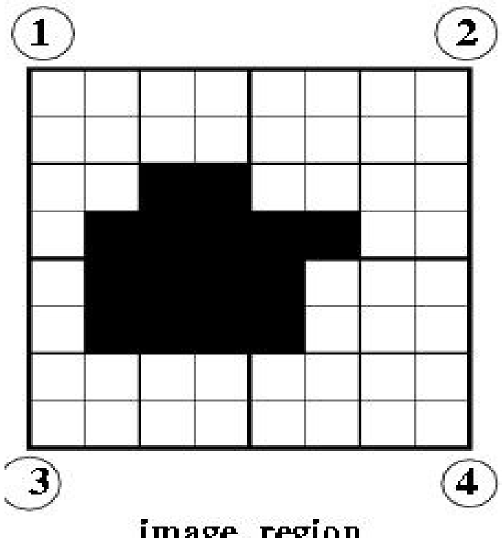
$$T = \left[\frac{(N-1)N}{(N+1)} (y - \overline{X})^2 / S^2 \right]^{\frac{1}{2}}$$

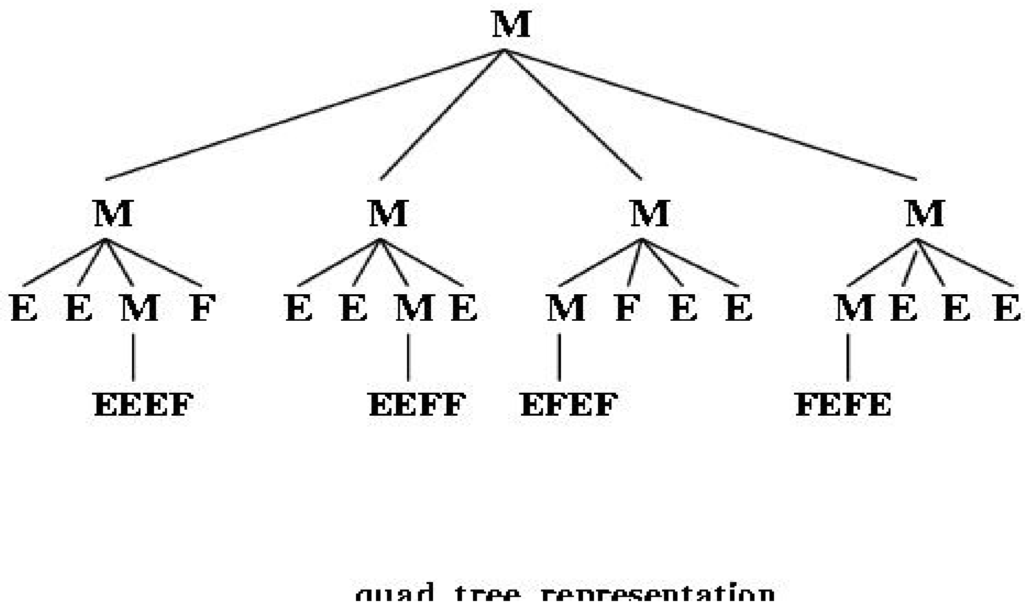
Representation of regions

- "overlay" or "mask": binary image for each defined region
- "labeled image": each integer label represents region or region type
- "boundary coding": use perimeter set, chain code, or polygonal rep.
- "quad tree": hierarchical spatial partition using white, black, gray nodes

Quad tree recursively partitions the 2D plane

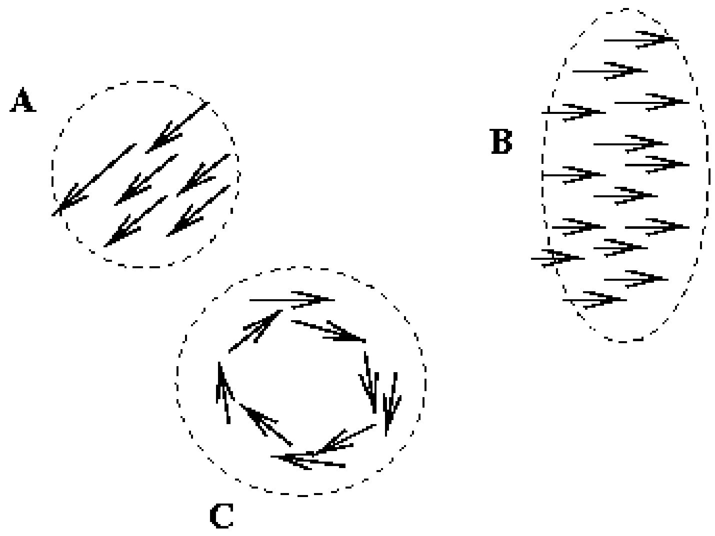
- E: empty, all background
- F: full, all foreground
- M: mixed, some background and some foreground





Segmentation via motion coherence

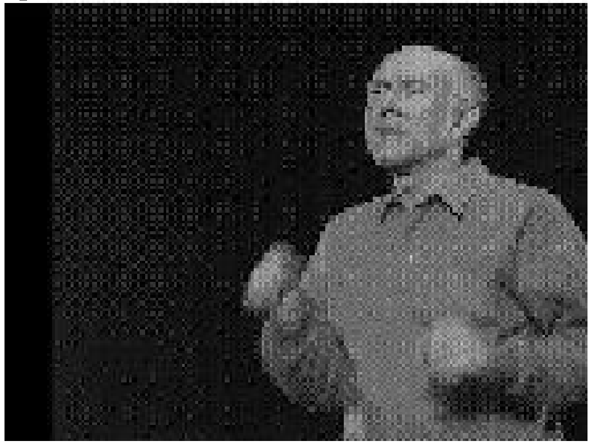
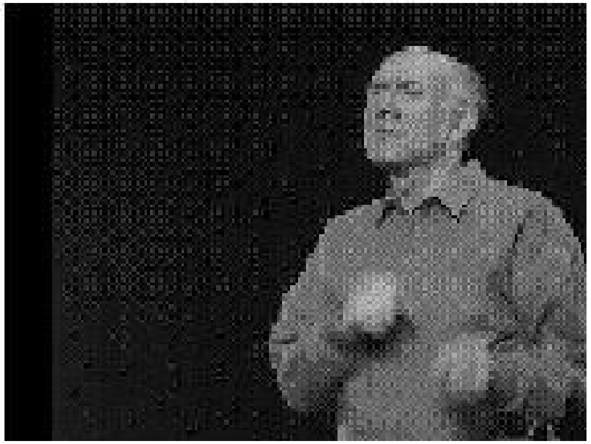
- Assume that neighboring pixels with the same motion are part of the same object
- Objects A, B translate, C rotates



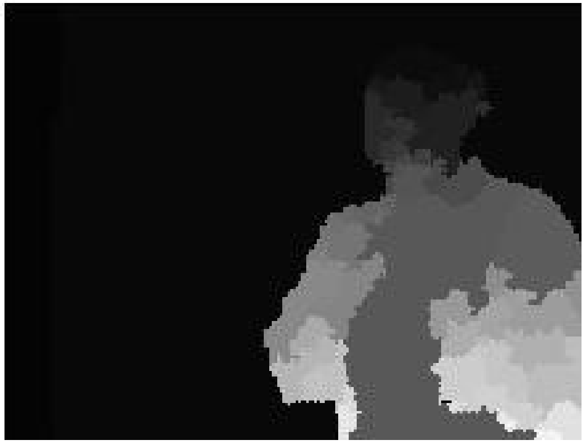
American Sign Language

- Find hands and face using color and motion
- Find skin-colored pixels; find connected regions of appropriate size (hand versus face)
- Analyze motion to decode message; decode motion of each hand, eventually use face expression also

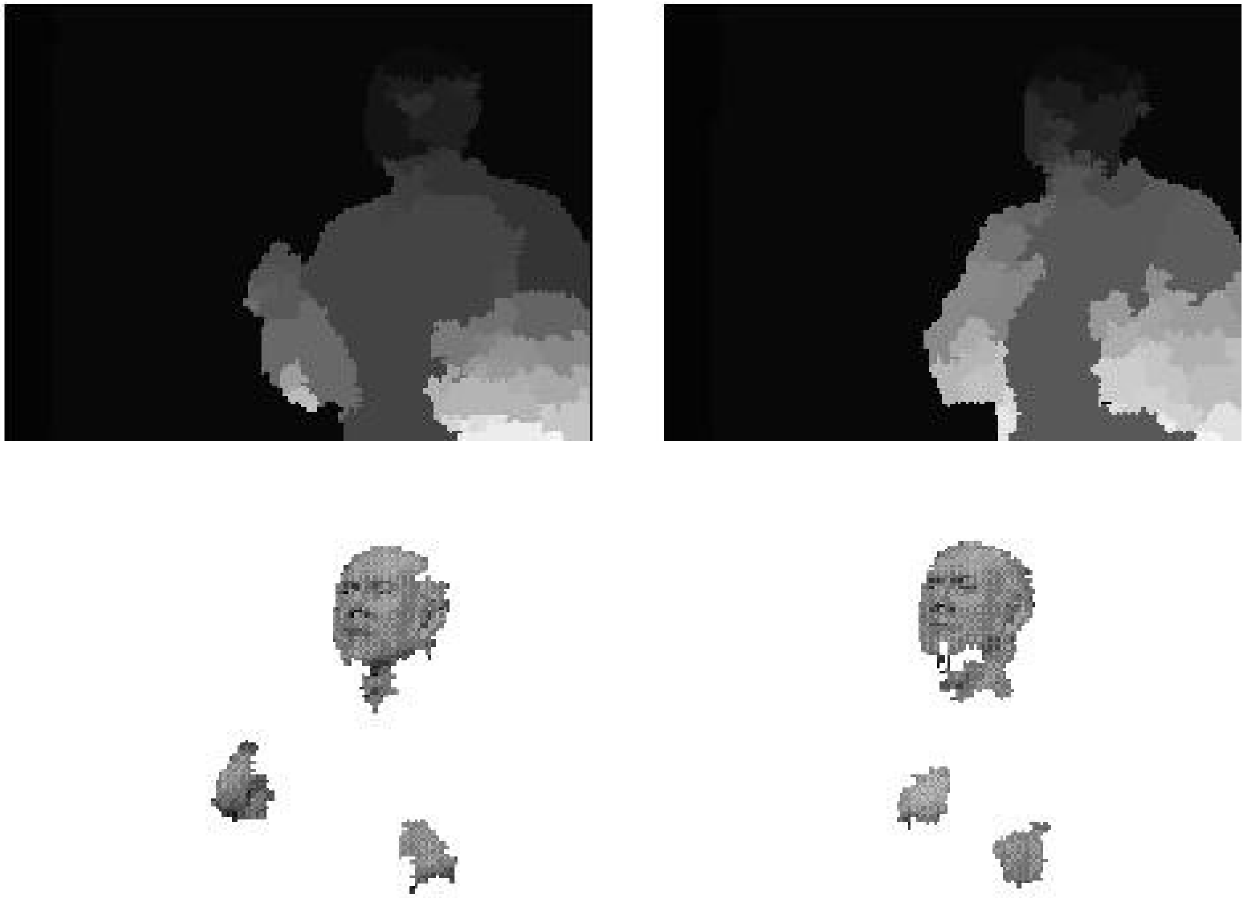
Compute motion of regions across two video frames(1)

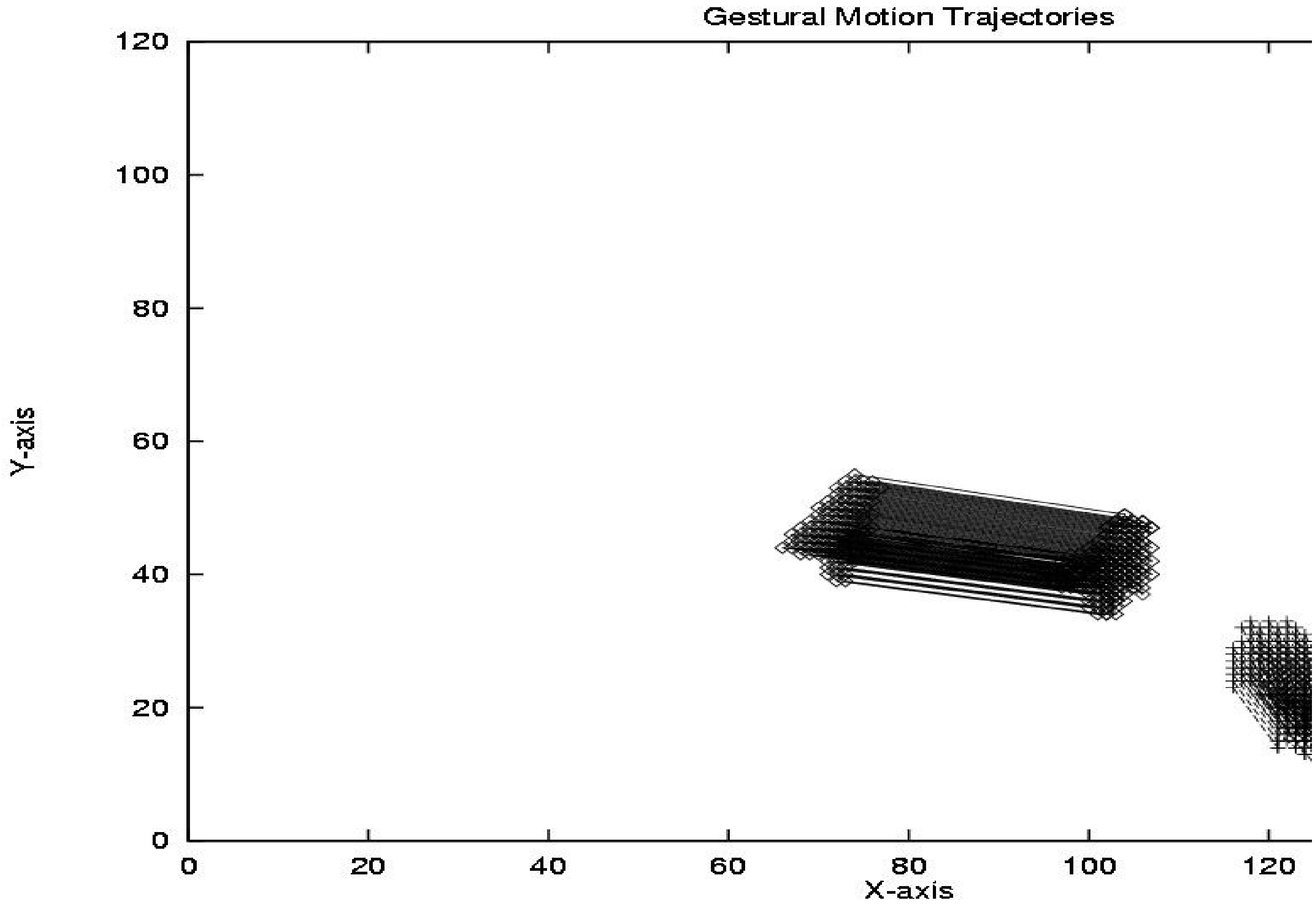




Find face and hand regions using skin color model and size(2)



Compute trajectory of hands(3)



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.

K-Means

- Choose a fixed number of clusters
- Choose cluster centers and point-cluster allocations to minimize error
- can't do this by search, because there are too many possible allocations.

$$\sum_{i \in \text{clusters}} \left\{ \sum_{j \in \text{elements of i'th cluster}} \left\| x_j - \mu_i \right\|^2 \right\}$$

- Algorithm
  - fix cluster centers; allocate points to closest cluster
  - fix allocation; compute best cluster centers
- x could be any set of features for which we can compute a distance (careful about scaling)

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 $m_1(1), \ldots, m_K(1)$.

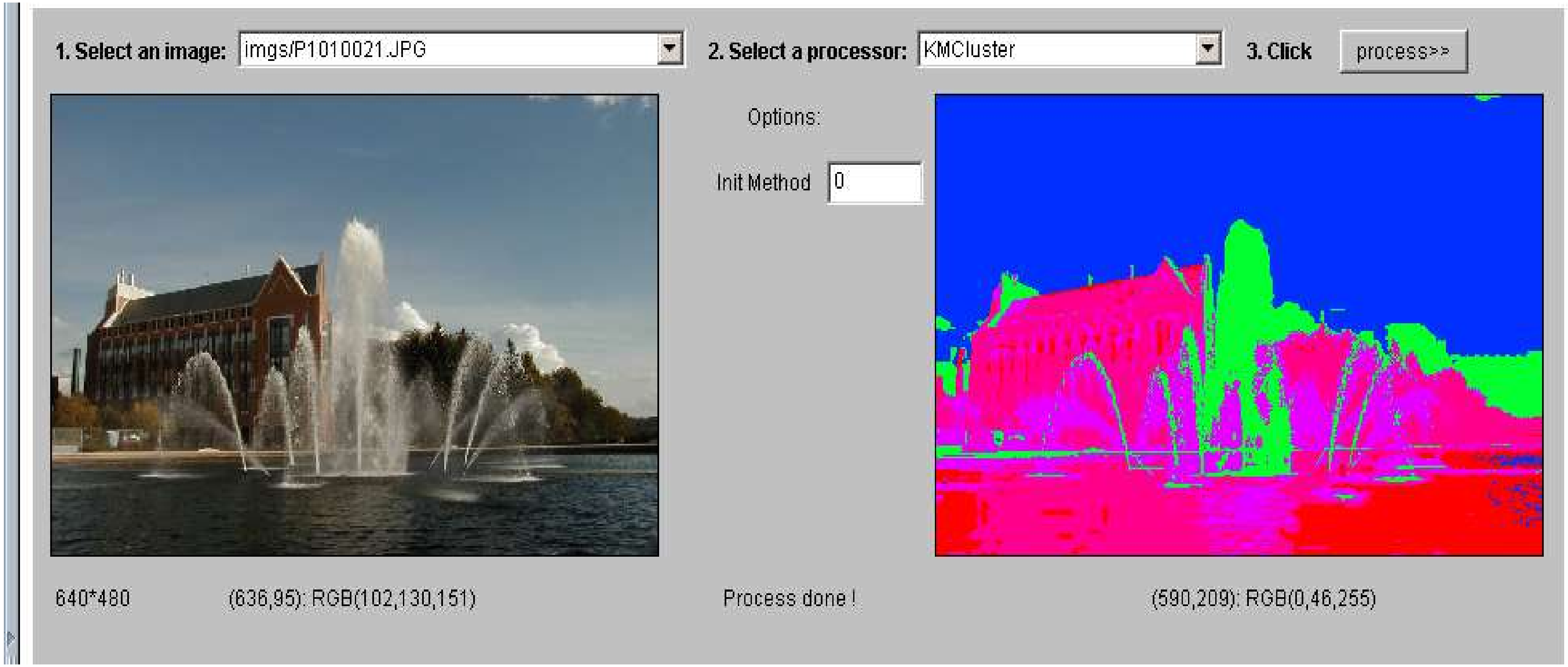
3. For each vector x_i compute $D(x_i, m_k(ic))$, $k=1,\ldots K$ and assign x_i to the cluster C_j with nearest mean.

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

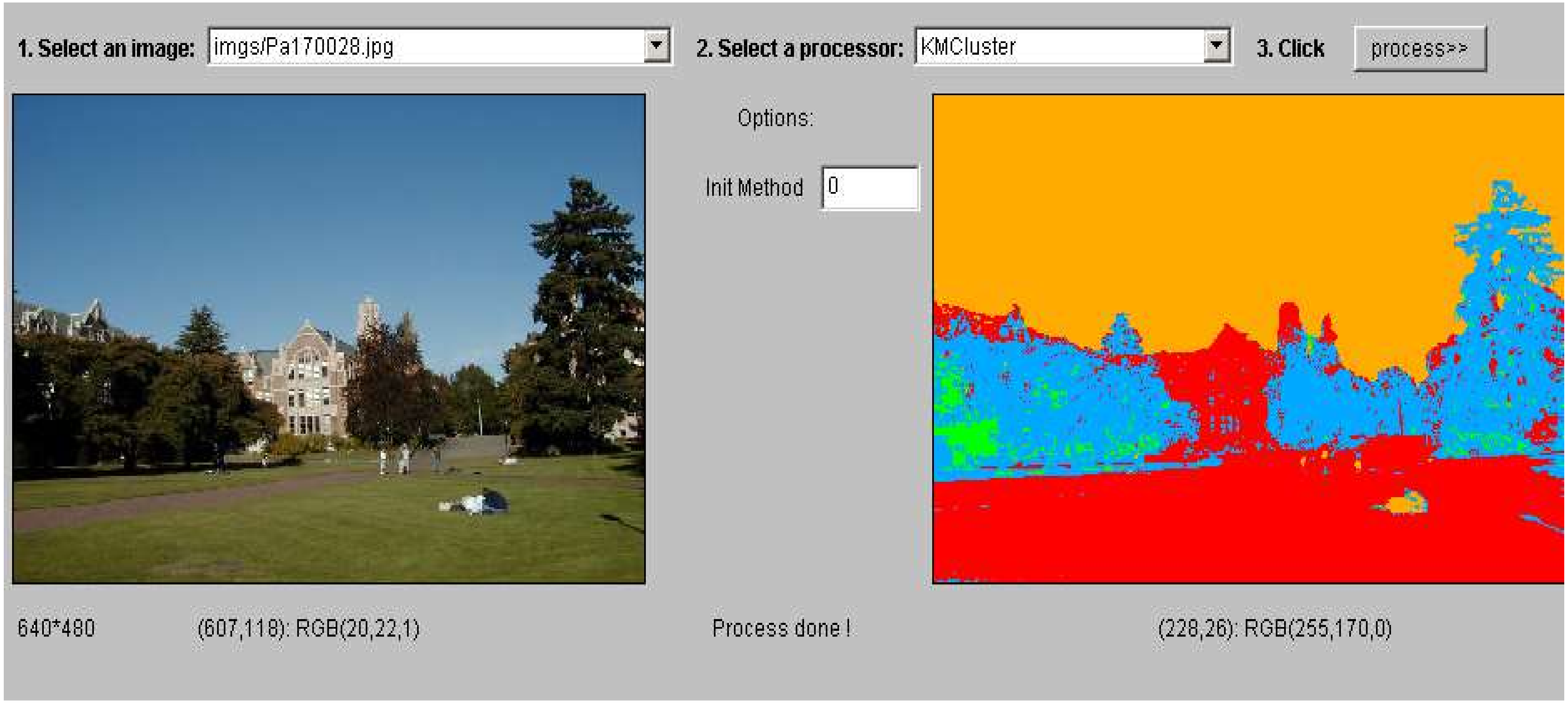
5. Repeat steps 3 and 4 until $C_k(ic) = C_k(ic+1)$ for all k.

K-Means Example 1

K-Means Example 2



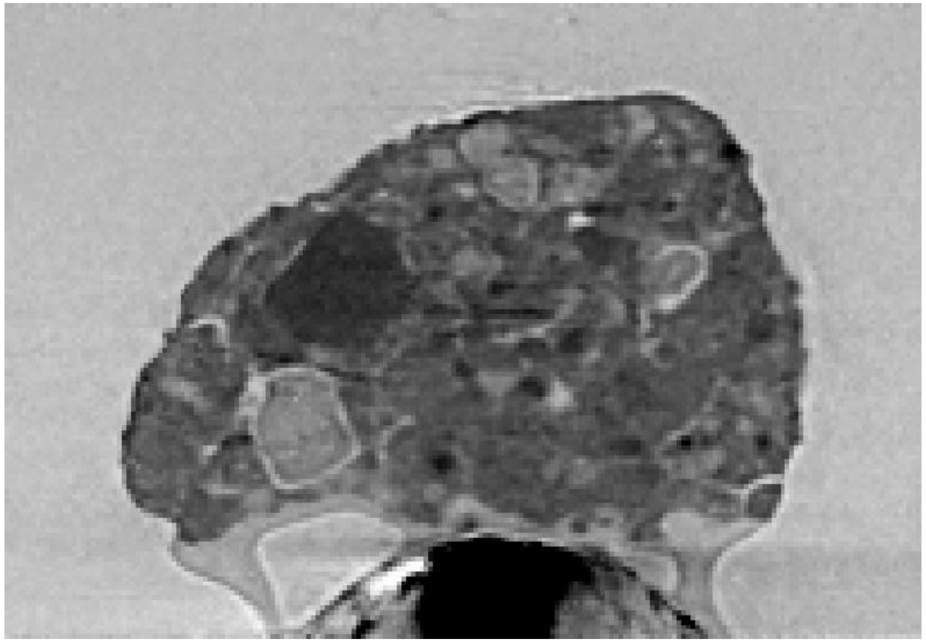
K-Means Example 3

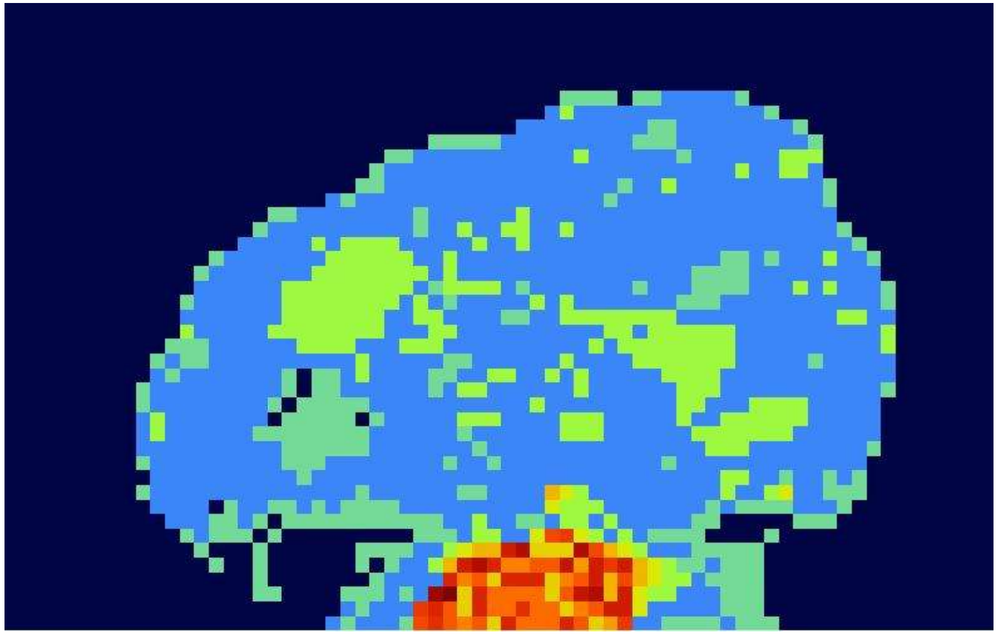


K-Means Example 4

K-Means Example 5

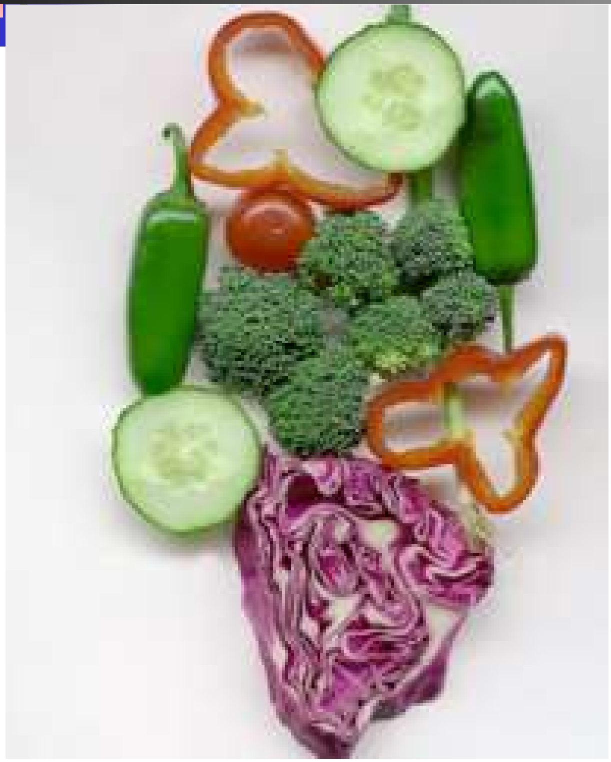
Slice through soil microvolume





- Several regions identified: intensity represents material density (similar to an X-ray image)

K-Means Example 6



Image

Clusters on intensity

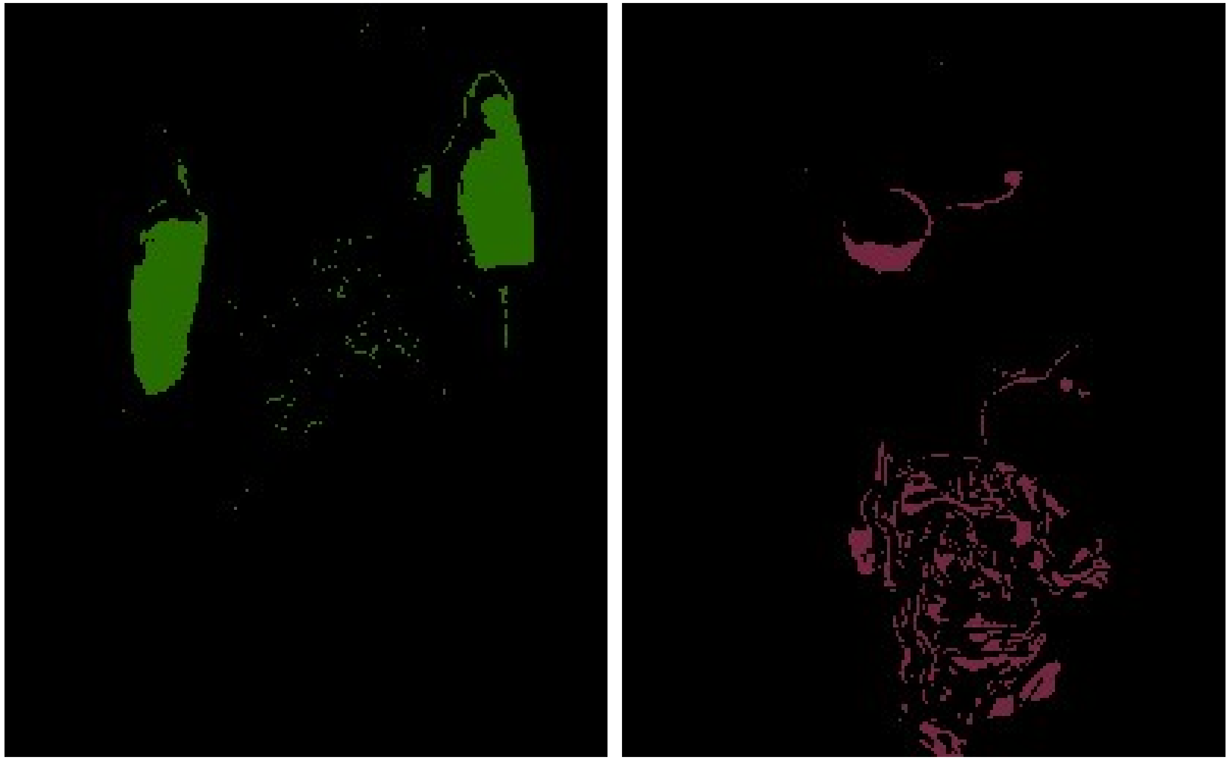
Clusters on color

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.

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

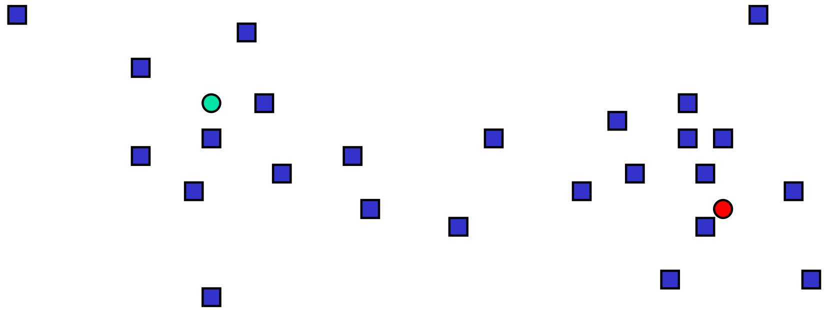
- Boot Step:
  - Initialize K clusters: $C_1, \ldots, C_K$

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

    $$x_i \Longrightarrow C(x_i)$$
  - Re-estimate the cluster parameters

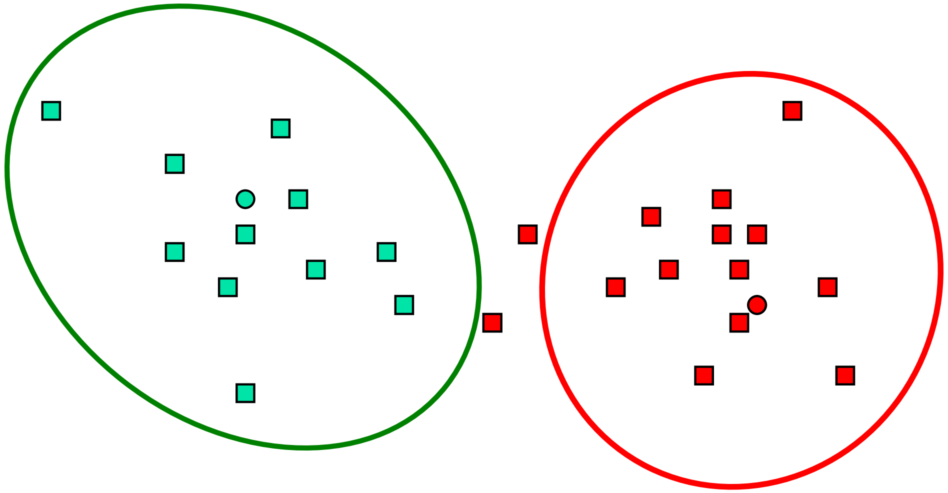
    $$\boxed{m_j = mean\{x_i \mid x_i \in C_j\}}$$

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

    $$p(C_j \mid x_i)$$

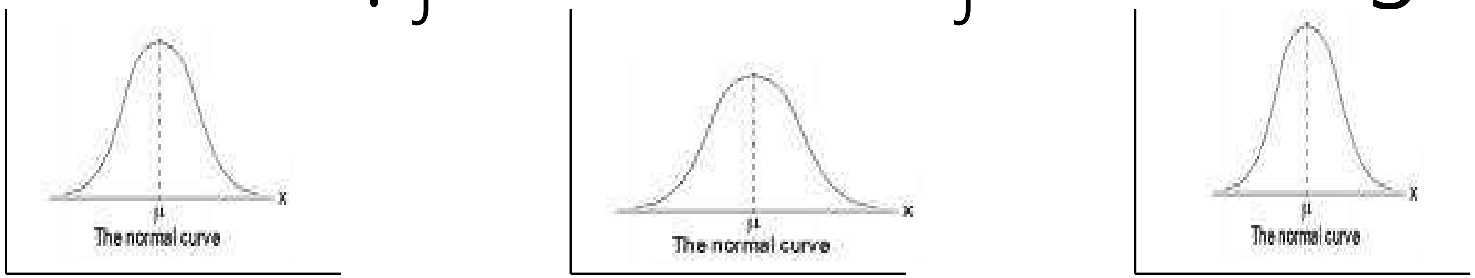
    ➡ Expectation
  - Re-estimate the cluster parameters

$$(\mu_j, \Sigma_j), p(C_j) \qquad \text{For each cluster } j$$

➡ 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 $P(C_j \mid x_i)$.
- $P(C_j \mid x_i) = P(x_i \mid C_j)\, P(C_j) \,/\, P(x_i)$
- $P(x_i) = \Sigma\, P(x_i \mid C_j)\, P(C_j)$
- Where do we get $P(x_i \mid C_j)$ and $P(C_j)$?

Expectation

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)		Input (Estimation)		Output
$x_1=\{r_1, g_1, b_1\}$ $x_2=\{r_2, g_2, b_2\}$ … $x_i=\{r_i, g_i, b_i\}$ …	+	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	⇒	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)		Input (Estimation)		Output
$x_1=\{r_1, g_1, b_1\}$ $x_2=\{r_2, g_2, b_2\}$ … $x_i=\{r_i, g_i, b_i\}$ …	+	Classification Results $p(C_1 \vert x_1)$ $p(C_j \vert x_2)$ … $p(C_j \vert x_i)$ …	⇒	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

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

Full EM Algorithm Multi-Dimensional

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

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

- Iteration Step:
  - 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 Applications

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

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

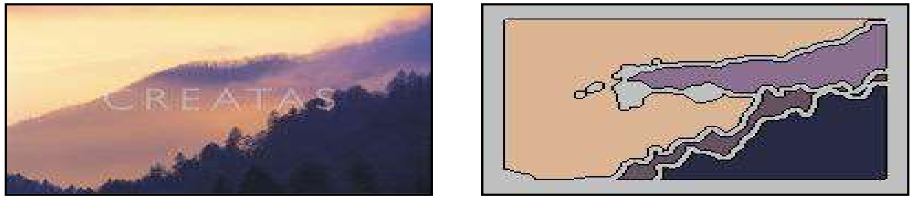
EM Applications

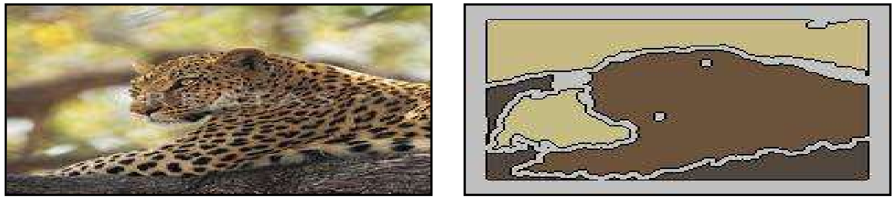
- Blobworld: Image segmentation using Expectation-Maximization and its application to image querying

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

Blobworld: Sample Results

- 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.





Agglomerative clustering

- Assume that each cluster is single pixel (i.e. every pixel is a cluster itself).
- Merge Clusters i.e. attach closest to cluster it is closest to (if possible)
- Repeat step 2 until no more clusters can be merged.

Divisive clustering

- Assume that whole image is a single cluster.
- Split Clusters along best boundary (if exists)
- Repeat step 2 until no more clusters can be split.

Inter-Cluster distance

- Single-link clustering: Minimum distance between an element of the first cluster and one of the second..
- Complete-link clustering: Maximum distance between an element of the first cluster and one of the second.
- Group-average clustering: Average of distances between elements in the clusters.

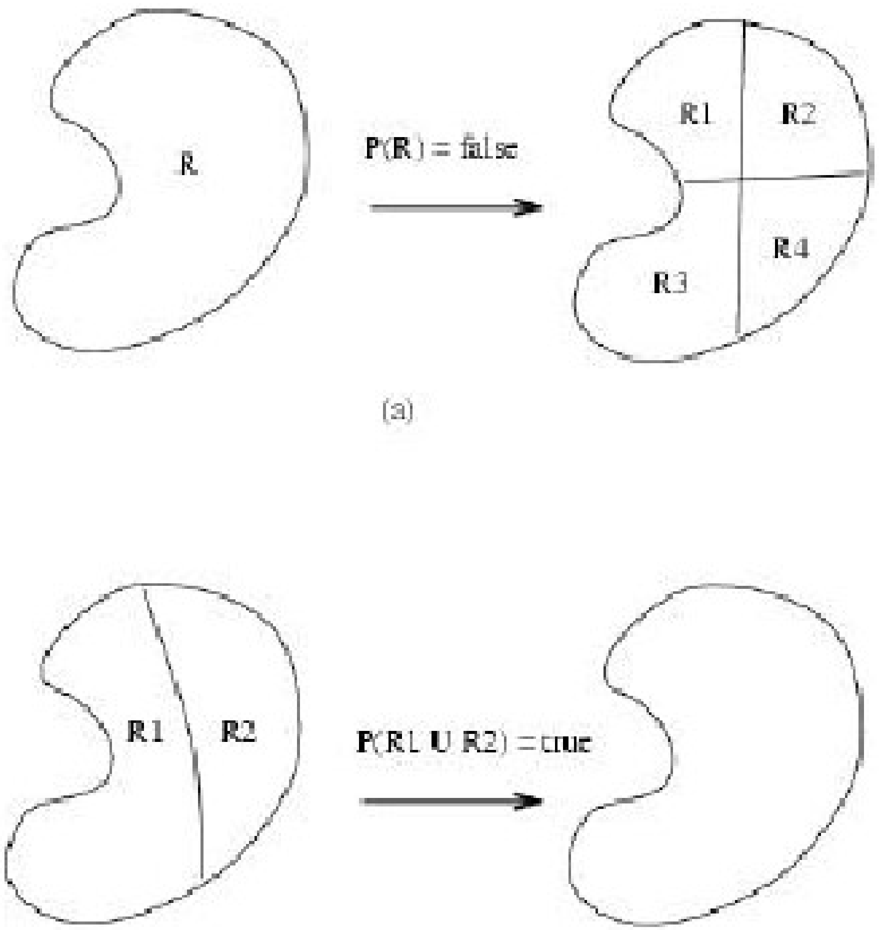
Segmentation by Split and Merge

- Start with an initial segmentation (for example by K-Means).
- Define a criteria P for goodness of region such that
  - P(*R*)=True, if *R* satisfies the criteria
  - P(*R*)=False, otherwise

- For each region *R*, split *R* in four regions (quadrants), if P(*R*) = False
- Merge any two adjacent regions *R* and *Q* if
- Repeat until no more clusters can be split or merged.

$$\mathrm{P}(Q \cup R) = \mathrm{True}$$

Segmentation by Split and Merge



Suggested Reading

- Chapter 14, David A. Forsyth and Jean Ponce, “Computer Vision: A Modern Approach”.

- Chapter 3, Mubarak Shah, “Fundamentals of Computer Vision”