

## Computer Vision Week 7

Segmentation: Region based techniques:  
Chapter 10

## Segmentation

Given a set of image pixels  $I$  and a uniformity predicate  $P(\cdot)$ ; find a partition  $S$  of the image  $I$  into a set of  $n$  regions  $R_i$  such that:

1.  $\bigcup_i R_i = I \quad R_i \cap R_j = \emptyset$
2.  $P(R_i) = \text{True}$
3.  $P(R_i \cup R_j) = \text{False}$  for  $R_i$  adjacent to  $R_j$

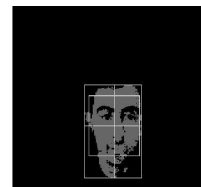
## Two Approaches for Segmentation

1. Detect discontinuities:  
Boundary based approach  
- edge detection followed  
by thinning, linking, etc.
2. Detect uniformity: Region  
growing



## Segmenting an Image: Regions

- Want “meaningful” regions of image
- Objects often differ in appearance
- We change representation from pixels to regions



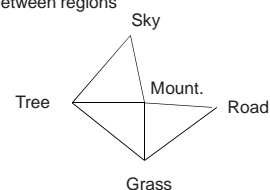
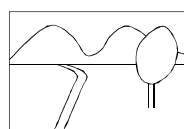
## Region Representation

1. “Labeled image”: Array representation (Labels for regions)
2. “overlay” or “mask”: binary image for each defined region
3. “boundary coding”: use perimeter set, chain code, or polygonal rep.
4. Hierarchical rep: pyramids or quadtrees
5. Symbolic representation (use features)

## Data Structures for Representation

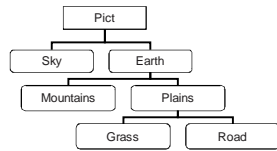
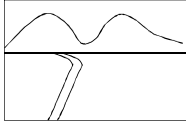
1. Region Adjacency Graph

Nodes  $\rightarrow$  Regions  
Arcs  $\rightarrow$  Boundary between regions



## Data Structures for Representation

- Picture Trees



## Region Based Techniques for Segmentation

1. Measurement space techniques: (Clustering)  
These techniques are not concerned with the position of pixels!
  - Thresholding
  - Color quantization
2. Region growing: These take into account the position of pixels
  - Connected components labeling

```

1111000001
1111000000
0000000000
0000001111
1111111111
  
```

## 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

- 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

## Region forming concepts also apply to boundary segmentation

- "boundary following uses adjacency as major control
- Hough transform uses spatial similarity as major control

## Thresholding, revisited

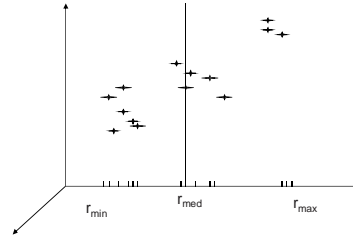
- Thresholding is clustering in 1D
- Histogram based
- Otsu threshold
- Other thresholding techniques: Kapur, Ohlander etc.

### Color quantization

- Color quantization is clustering in the color space (3D)
- Median cut: recursive thresholding in 1D
- BTS: makes use of PCA
- K-means
- Isodata clustering
- Shi's Graph partitioning technique

### Median-cut for color quantization

- Recursively threshold r,g and b axes at the median of the colors
- Decide which group to split by looking at  $c_{max}-c_{min}$  where c is r,g,b

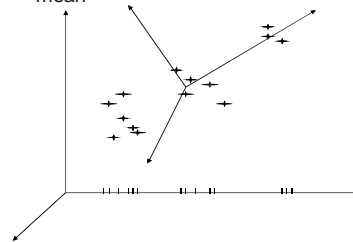


### More than two segments

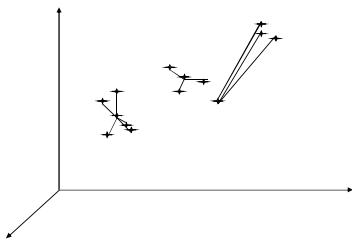
- Two options
  - Recursively split each side to get a tree, continuing till the eigenvalues are too small
  - Use the other eigenvectors

### BTS for color quantization

- Recursively apply PCA to the colors and split with planes perpendicular to the principal axis passing through the mean



### K-means for color quantization



### Iterative K-means clustering

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

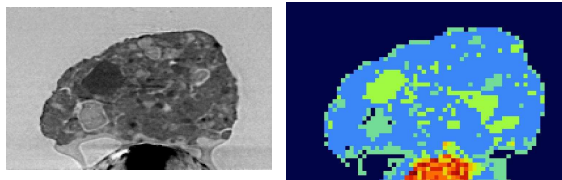
1. Set  $ic$  (iteration count) to 1.
2. Choose randomly a set of  $K$  means  $m_1(1), m_2(1), \dots, m_K(1)$ .
3. For each vector  $x_i$  compute  $D(x_i, m_k(ic))$  for each  $k = 1, \dots, K$  and assign  $x_i$  to the cluster  $C_j$  with the nearest mean.
4. Increment  $ic$  by 1 and update the means to get a new set  $m_1(ic), m_2(ic), \dots, m_K(ic)$ .
5. Repeat steps 3 and 4 until  $C_k(ic) = C_k(ic + 1)$  for all  $k$ .

Algorithm 10.1 K-Means Clustering.

### K-means results



### Slice through soil microvolume



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

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 using colour and position, 20 segments

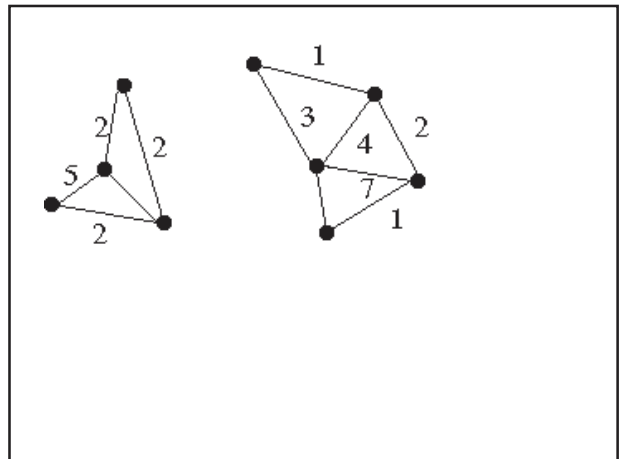
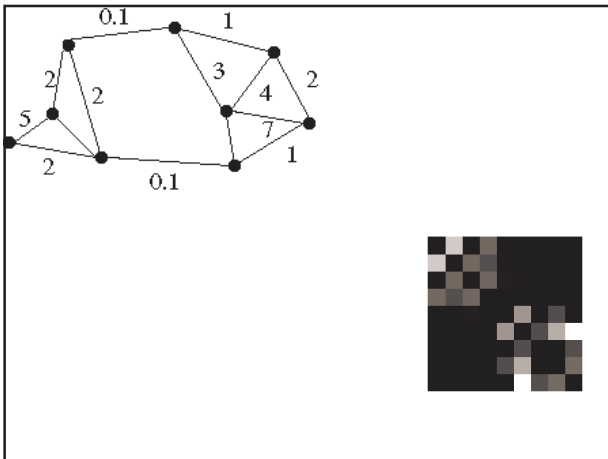


### Isodata clustering

- Form isodata clusters from a set of n-dimensional vectors:
- Assign  $x_i$  to the cluster  $i$  that minimizes
 
$$D_{\Sigma} = [x_i - m_i]^T \Sigma_i^{-1} [x_i - m_i]$$
  - Merge clusters  $i$  and  $j$  if  $|m_i - m_j| < \tau_v$
  - Split cluster  $k$  if the maximum eigenvalue of  $\Sigma_k$  is greater than  $\tau_v$
  - Stop when
 
$$|m_i(t) - m_i(t+1)| < \epsilon$$
 for every cluster  $i$  or when the maximum number of iterations has been reached.

### Graph theoretic clustering

- Represent tokens using a weighted graph.
  - affinity matrix
- Cut up this graph to get subgraphs with strong interior links



### Measuring Affinity

Intensity

$$aff(x, y) = \exp\left\{-\left(\frac{1}{2\sigma_i^2}\right)\left(\|I(x) - I(y)\|^2\right)\right\}$$

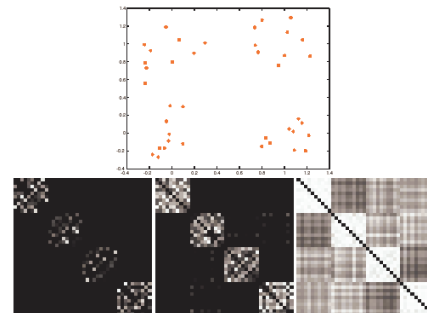
Distance

$$aff(x, y) = \exp\left\{-\left(\frac{1}{2\sigma_d^2}\right)\left(\|x - y\|^2\right)\right\}$$

Color

$$aff(x, y) = \exp\left\{-\left(\frac{1}{2\sigma_c^2}\right)\left(\|c(x) - c(y)\|^2\right)\right\}$$

### Scale affects affinity



## Eigenvectors and cuts

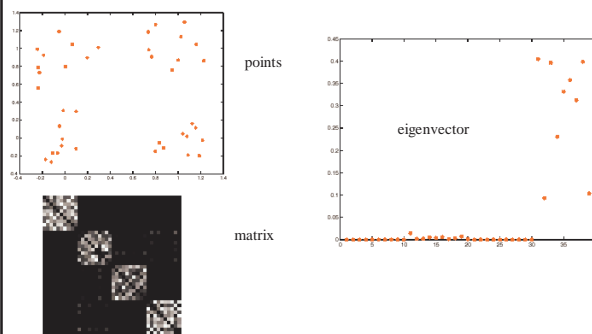
- Simplest idea: we want a vector  $a$ , giving the association between each element and a cluster
- We want elements within this cluster to, on the whole, have strong affinity with one another
- We could maximize
- But need the constraint

$$a^T A a$$

$$a^T a = 1$$

- This is an eigenvalue problem - choose the eigenvector of  $A$  with largest eigenvalue

## Example eigenvector



## More than two segments

- Two options
  - Recursively split each side to get a tree, continuing till the eigenvalues are too small
  - Use the other eigenvectors

## Normalized cuts

- Current criterion evaluates within cluster similarity, but not across cluster difference
- Instead, we'd like to maximize the within cluster similarity compared to the across cluster difference
- Write graph as  $V$ , one cluster as  $A$  and the other as  $B$
- Maximize
- i.e. construct  $A, B$  such that their within cluster similarity is high compared to their association with the rest of the graph

$$\left( \frac{assoc(A,A)}{assoc(A,V)} \right) + \left( \frac{assoc(B,B)}{assoc(B,V)} \right)$$

## Normalized cuts

- Write a vector  $y$  whose elements are 1 if item is in  $A$ , -1 if it's in  $B$
- This is hard to do, because  $y$ 's values are quantized
- Write the matrix of the graph as  $W$ , and the matrix which has the row sums of  $W$  on its diagonal as  $D$ ,  $\mathbf{1}$  is the vector with all ones.
- Criterion becomes
- and we have a constraint

$$\min_y \left( \frac{y^T (D - W) y}{y^T D y} \right)$$

$$y^T D \mathbf{1} = 0$$

## Normalized cuts

- Instead, solve the generalized eigenvalue problem
- which gives
- Now look for a quantization threshold that maximises the criterion --- i.e all components of  $y$  above that threshold go to one, all below go to -1

$$\max_y (y^T (D - W) y) \text{ subject to } (y^T D y = 1)$$

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

### Shi's Graph theoretic clustering

- Set up a weighted graph; nodes  $V$  are pixels, edges are  $E$ . The weight  $w(i,j)$  of an edge is the similarity between pixels. Weights are defined by:

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

where  $X$  is the spatial location and  $F$  is the feature

- The graph can be partitioned into two sets  $A$  and  $B$  by removing any edges between nodes in  $A$  and in  $B$ .
- The total weight of removed edges is called a cut.
- One way of segmenting the image is finding the minimum normalized cut.

### Shi's Graph theoretic clustering

- Set up a weighted graph; nodes  $V$  are pixels, edges are  $E$ . The weight  $w(i,j)$  of an edge is the similarity between pixels. Let

$$d(i) = \sum_j w(i, j)$$

represent the total connection from a node to all others.  $W$  is a  $N \times N$  matrix of weights and  $D$  is a diagonal matrix with  $d$  on its diagonal. Let  $x$  be a vector defined as:

$$x_i = \begin{cases} 1 & \text{if node } i \text{ is in } A \\ -1 & \text{otherwise} \end{cases}$$

and let  $y$  be the continuous approximation to  $x$  defined by

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

### Shi's Graph theoretic clustering

- Solve the system of equations:

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

for the eigenvectors  $y$  and eigenvalues  $\lambda$

- Use the eigenvector with the second smallest eigenvalue to bipartition the graph to find the splitting point such that  $N$  cut is minimized
- Decide if the current partition should be subdivided further by checking the stability of the cut and making sure that  $N$  cut is below a pre-specified threshold value.
- Recursively repartition the segmented parts if necessary

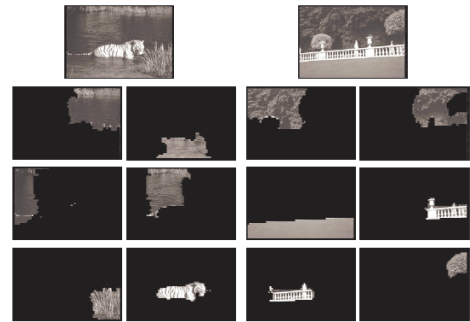


Figure from "Image and video segmentation: the normalised cut framework", by Shi and Malik, copyright IEEE, 1998



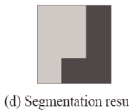
Figure from "Normalized cuts and image segmentation," Shi and Malik, copyright IEEE, 2000

### A different approach to graph cuts (Boykov & Jolly, Kolmogorov et al)

- Two kinds of vertices

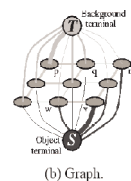


(a) Image with seeds.

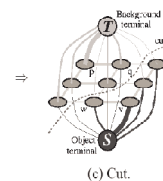


(d) Segmentation results.

- Two kinds of edges



(b) Graph.



(c) Cut.

- Cut - Segmentation

### Graph Cuts and Max-Flow/Min-Cut Algorithms

- A **flow network** is defined as a directed graph where an edge has a nonnegative capacity
- A **flow** in  $G$  is a real-valued (often integer) function that satisfies the following three properties:
  - Capacity Constraint:  
For all  $u, v \in V, f(u, v) \leq c(u, v)$
  - Skew Symmetry  
For all  $u, v \in V, f(u, v) = -f(v, u)$
  - Flow Conservation  
For all  $u \in (V \setminus \{s, t\}), \sum_{v \in V} f(u, v) = 0$

### How to Find the Minimum Cut?

- *Theorem*  
In graph  $G$ , the maximum source-to-sink flow possible is equal to the capacity of the minimum cut in  $G$ .

(L. R. Foulds, *Graph Theory Applications*, 1992 Springer-Verlag New York Inc., 247-248)

### Maximum Flow and Minimum Cut Problem

- Some Concepts
  - If  $f$  is a flow, then the net flow across the cut  $(S, T)$  is defined to be  $f(S, T)$ , which is the sum of all edge capacities from  $S$  to  $T$  subtracted by the sum of all edge capacities from  $T$  to  $S$
  - The capacity of the cut  $(S, T)$  is  $c(S, T)$ , which is the sum of the capacities of all edge from  $S$  to  $T$
  - A minimum cut is a cut whose capacity is the minimum over all cuts of  $G$ .

### Algorithms to Solve Max-Flow Problem

- Ford-Fulkerson Algorithm
- Push-Relabel Algorithm
- New Algorithm by Boykov, etc.

### Solving Image Segmentation Problem

- Data element set  $P$  representing the image pixels/voxels
- Neighborhood system as a set  $N$  representing all pairs  $\{p, q\}$  of neighboring elements in  $P$  (ordered or unordered)
- $A$  be a vector specifying the assignment of pixel  $p$  in  $P$ , each  $A_p$  can be either in the background or the object  
 $A = (A_1, A_2, \dots, A_p, \dots, A_{|P|})$
- $A$  defines a segmentation of  $P$

### Cost of Segmentation

$$E(A) = \lambda \cdot R(A) + B(A)$$

$$R(A) = \sum_{p \in P} R_p(A_p)$$

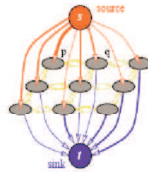
$$B(A) = \sum_{p \in P} \sum_{\{p, q\} \in N} B_{\{p, q\}} \cdot \delta(A_p, A_q)$$

$$\delta(A_p, A_q) = \begin{cases} 1 & A_p \neq A_q \\ 0 & A_p = A_q \end{cases}$$

$R(A)$  defines the penalties for assigning  $A_p$  to object or background, which are  $R_p(\text{"obj"})$  and  $R_p(\text{"bkg"})$   
 $B(A)$  describes the boundary properties of the segmentation,  $B_{\{p, q\}}$  is large when  $p$  and  $q$  are similar, it is close to 0 when  $p$  and  $q$  are very different

### Graph Construction

Edge	Weight	Condition
n-link $\{p, q\}$	$B_{\{p,q\}}$	$\{p, q\} \in N$
t-link $\{p, s\}$	$\lambda \cdot R_p$ ("bkg")	$p \in P, p \in OUB$
	$K$	$p \in O$
	$0$	$p \in B$
t-link $\{p, t\}$	$\lambda \cdot R_p$ ("obj")	$p \in P, p \in OUB$
	$0$	$p \in O$
	$K$	$p \in B$



### n-link Construction

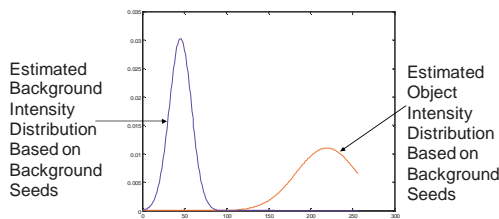
$$B_{\{p,q\}} \propto \exp\left(-\frac{(I_p - I_q)^2}{2\sigma^2}\right) \cdot \frac{1}{dist(p,q)}$$

- $I_p$  and  $I_q$  are the intensities of pixel  $p$  and  $q$
- $\sigma$  sets the penalty of discontinuities between pixels of similar intensities, when  $|I_p - I_q| < \sigma$  the penalty is large, when  $|I_p - I_q| > \sigma$  the penalty is small
- $dist(p, q)$  penalizes the distance between  $p$  and  $q$ , and generally when they are in the neighborhood system this term is one

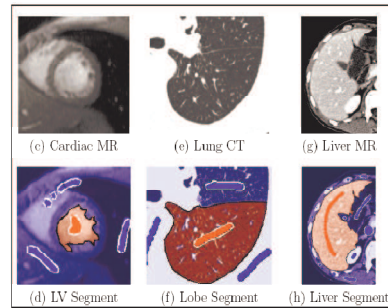
### t-link Construction

$$R_p(\text{"obj"}) = -\ln \Pr(I_p|O)$$

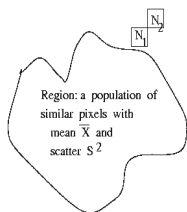
$$R_p(\text{"bkg"}) = -\ln \Pr(I_p|B) \quad K = 1 + \max_{p \in P} \left( \sum_{q: \{p,q\} \in N} B_{\{p,q\}} \right)$$



### Experimental Data



### 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.

### Single Linkage Region Growing

- Link two neighbor pixels if a similarity measure is satisfied
- Example:  $|pixel1 - pixel2| < T$
- Connected components works this way
- Disadvantage: "leaks" are easy

### Hybrid Linkage Region growing

- Similarity measure incorporates neighborhood information
- Let  $f(p)$  be a property vector computed on a  $K \times K$  neighborhood
- Similarity measure:  $d(f(p_1), f(p_2)) < T$

### Centroid Linkage Region Growing

- New pixel's value is compared against the centroid of a region.
- If close, pixel is added to region and centroid updated

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

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

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

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

$$S_{new}^2 \leftarrow S_{old}^2 + (y - \bar{X}_{new})^2 + N(\bar{X}_{new} - \bar{X}_{old})^2$$

### Advantages and Disadvantages

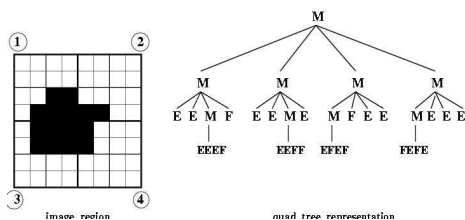
- Single linkage places boundaries accurately BUT leaks may occur
- Centroid linkage can place boundaries in weak gradient areas

### Variant hybrid region growing

- Use boundary information also:
  - Apply edge detection
  - Dilate the edges to obtain connected edge region
  - Do centroid linkage for edge pixels
  - Do single linkage for edge pixels

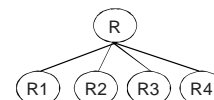
### Quad tree recursively partitions the 2D plane

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



### Split and Merge

- Can be quadtree or hierarchy-based
- If  $P(R)=FALSE$ , split
- If two adjacent regions are similar, merge



## Watershed

- Start with gradient minima in the image, grow regions simultaneously starting at the minimum level
- In 1D:

