CS 534: Computer Vision
Segmentation II
Graph Cuts and Image Segmentation

Spring 2004

Ahmed Elgammal
Dept of Computer Science
Rutgers University

Outlines

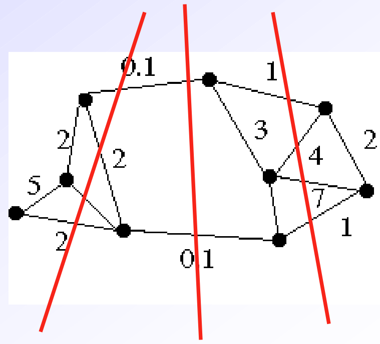
- What is Graph cuts
- Graph-based clustering
- Normalized cuts
- Image segmentation using Normalized cuts
- Other Cuts

Graph Cut

What is a Graph Cut:

- We have undirected, weighted graph $G=(V,E)$
- Remove a subset of edges to partition the graph into two disjoint sets of vertices A,B (two sub graphs):

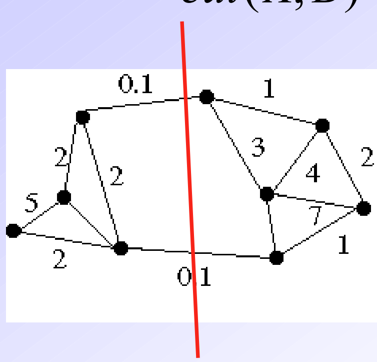
$A \cup B = V,\ A \cap B = \Phi$

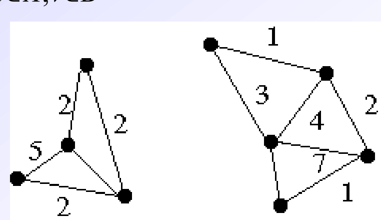


Graph Cut

- Each cut corresponds to some cost (cut): sum of the weights for the edges that have been removed.

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

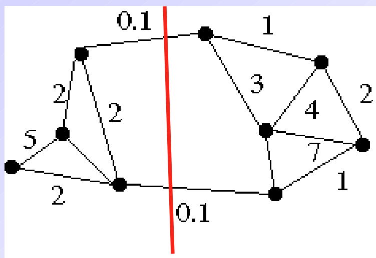


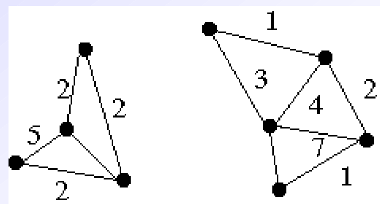


A B

Graph Cut

- In many applications it is desired to find the cut with minimum cost: *minimum cut*
- Well studied problem in graph theory, with many applications
- There exists efficient algorithms for finding minimum cuts





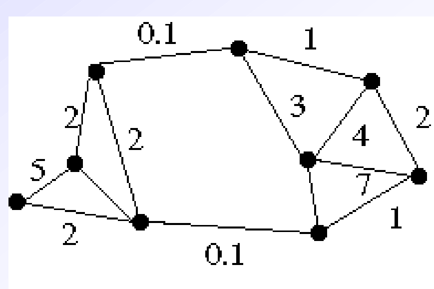
A B

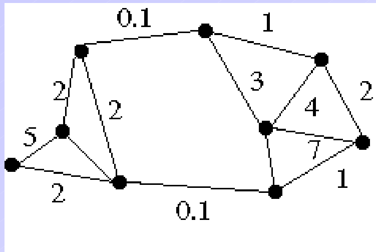
$$cut(A,B)=\sum_{u\in A,v\in B} w(u,v)$$

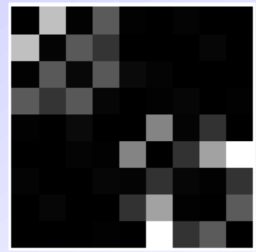
Graph theoretic clustering

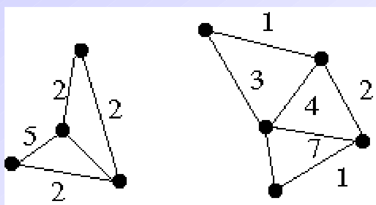
- Represent tokens using a weighted graph
  - Weights reflects similarity between tokens
  - *affinity* matrix
- Cut up this graph to get subgraphs such that:
  - Similarity within sets maximum.
  - Similarity between sets minimum.

⇒ Minimum cut





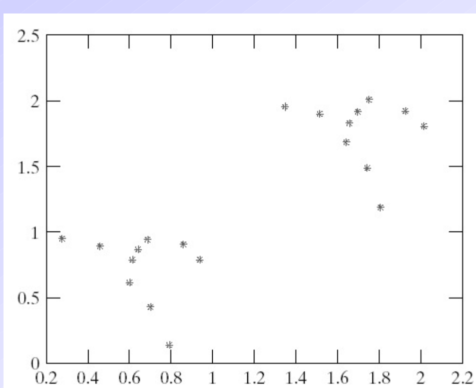


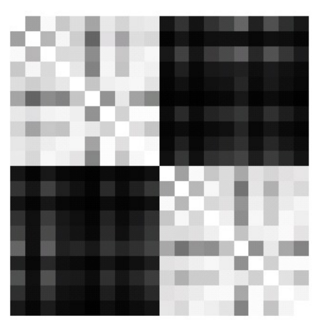


- Use exponential function for edge weights

$$w(x) = e^{-(d(x)/\sigma)^2}$$

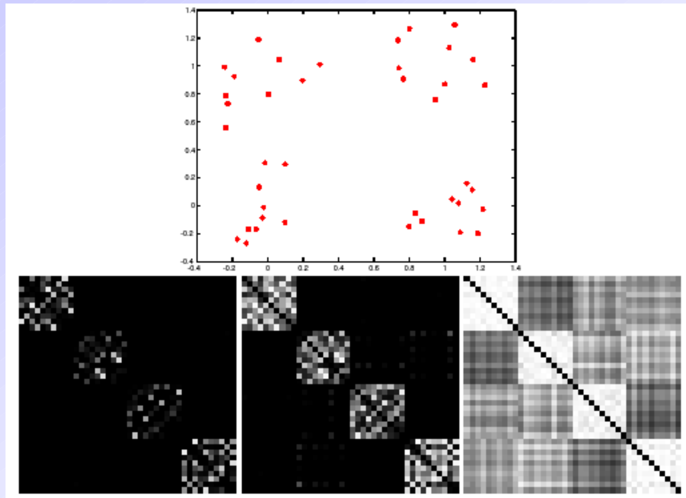
d(x) : feature distance





Scale affects affinity

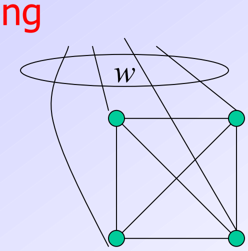
$$w(x) = e^{-(d(x)/\sigma)^2}$$



σ=0.1 σ=0.2 σ=1

Eigenvectors and clustering

- Simplest idea: we want a vector w 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



$$w_n^T A w_n$$

Sum of

Association of element i with cluster n ×

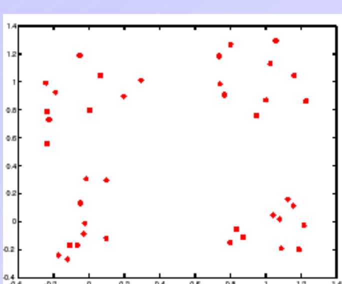
Affinity between i and j ×

Association of element j with cluster n

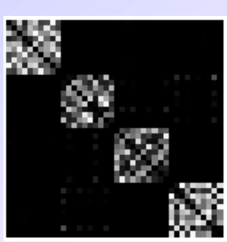
Eigenvectors and clustering

- We could maximize $$w_n^T A w_n$$
- But need the constraint $$w_n^T w_n = 1$$
- Using Lagrange multiplier λ
$$w_n^T A w_n + \lambda(w_n^T w_n - 1)$$
- Differentiation
$$A w_n = \lambda w_n$$
- This is an eigenvalue problem - choose the eigenvector of A with largest eigenvalue

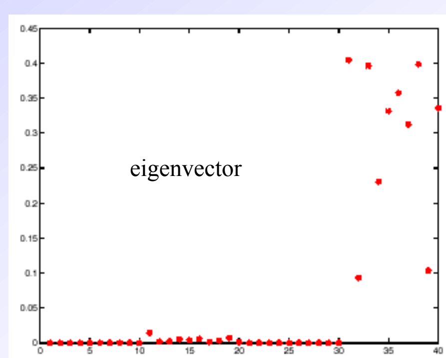
Example eigenvector



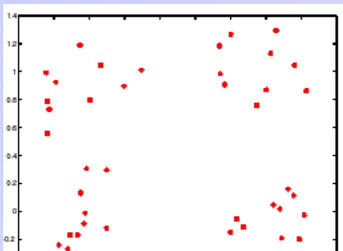
points



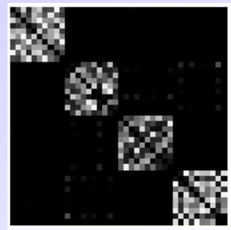
matrix



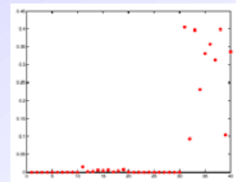
Example eigenvector



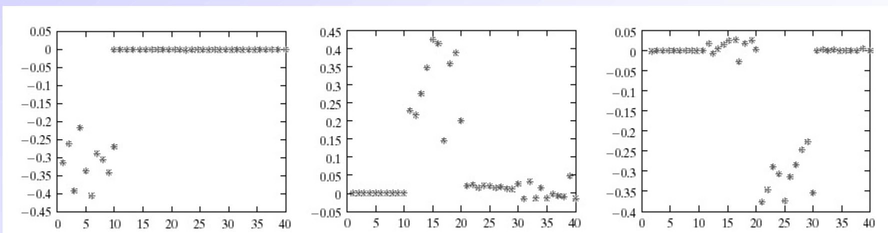
points



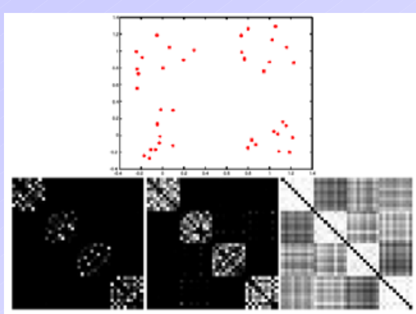
matrix

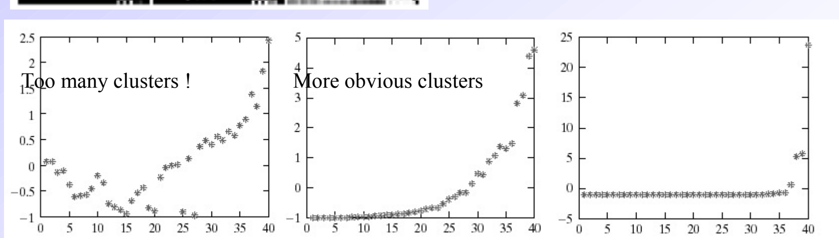


First eigenvectors



The three eigenvectors corresponding to the next three eigenvalues of the affinity matrix





eigenvalues for three different scales for the affinity matrix

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

Algorithm

- Construct an Affinity matrix A
- Computer the eigenvalues and eigenvectors of A
- Until there are sufficient clusters
  - Take the eigenvector corresponding to the largest unprocessed eigenvalue; zero all components for elements already clustered, and threshold the remaining components to determine which element belongs to this cluster, (you can choose a threshold by clustering the components, or use a fixed threshold.)
  - If all elements are accounted for, there are sufficient clusters

Graph Cuts and Image Segmentation

- Represents image as a graph
- A vertex for each pixel
- Edges between pixels
- Weights on edges reflect similarity (affinity) in:
  - Brightness
  - Color
  - Texture
  - Distance
  - …
- Connectivity:
  - Fully connected: edges between every pair of pixels
  - Partially connected: edges between neighboring pixels

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

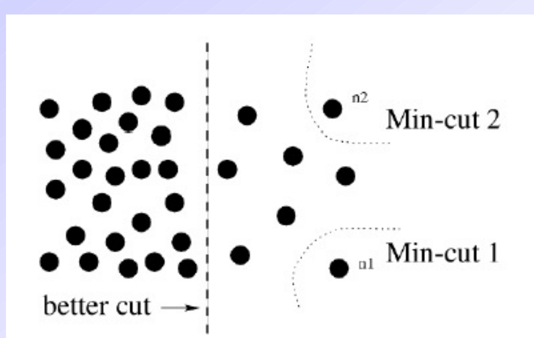
$$aff(x,y) = \exp\left\{-\left(\frac{1}{2\sigma_d^2}\right)\left(\|x-y\|^2\right)\right\}$$

color

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

Normalized Cuts

- Min cut is not always the best cut



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

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

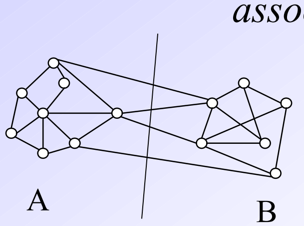
- i.e. construct A, B such that their within cluster similarity is high compared to their association with the rest of the graph

- Association between two sets of vertices: total connection between the two sets.

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

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

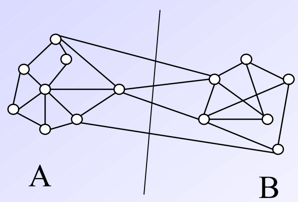
$$assoc(B,V)=\sum_{u\in B,t\in V} w(u,t)$$



- Normalize the cuts: compute the cut cost as a fraction of the total edge connections to all nodes in the graph

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

- Disassociation measure. The smaller the better.



$$cut(A,B) = assoc(A,B) = assoc(A,V) - assoc(A,A)$$

$$\begin{aligned} Ncut(A,B) &= \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)} \\ &= \frac{assoc(A,V) - assoc(A,A)}{assoc(A,V)} \\ &\quad + \frac{assoc(B,V) - assoc(B,B)}{assoc(B,V)} \\ &= 2 - \left(\frac{assoc(A,A)}{assoc(A,V)} + \frac{assoc(B,B)}{assoc(B,V)}\right) \\ &= 2 - Nassoc(A,B). \end{aligned}$$

$$Nassoc(A,B) = \frac{assoc(A,A)}{assoc(A,V)} + \frac{assoc(B,B)}{assoc(B,V)}$$

Total association (similarity) within groups, the bigger better

- By looking for a cut that minimizes *Ncut(A,B),*
  - Minimize the disassociation between the groups,
  - Maximize the association within group

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$
$$= 2 - Nassoc(A,B)$$

- Minimizing a normalized cut is NP-complete

Normalized cuts

- W: cost matrix: w(i,j)
- D: sum of the costs for every vertex
- Optimal Normalized cut can be found be solving for y that minimizes

$$D(i,i) = \sum_j W(i,j), \quad D(i,j) = 0 \quad i \neq j$$

$$\min_y \frac{y^T (D-W) y}{y^T D y} \qquad y \in \{1, -b\} \qquad y^T D1 = 0$$

- NP-complete problem,
- approximate real-valued solution by solving a generailized eigenvalue problem

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

- Real-valued solution is the second smallest eigenvector
- look for a quantization threshold that maximizes the criterion --- i.e all components of y above that threshold go to one, all below go to -b

Example - brightness

$$w_{ij} = e^{\frac{-\|\boldsymbol{F}_{(i)} - \boldsymbol{F}_{(j)}\|_2^2}{\sigma_I^2}} * \begin{cases} e^{\frac{-\|\boldsymbol{X}_{(i)} - \boldsymbol{X}_{(j)}\|_2^2}{\sigma_X^2}} & \text{if } \|\boldsymbol{X}(i) - \boldsymbol{X}(j)\|_2 < r \\ 0 & \text{otherwise.} \end{cases}$$

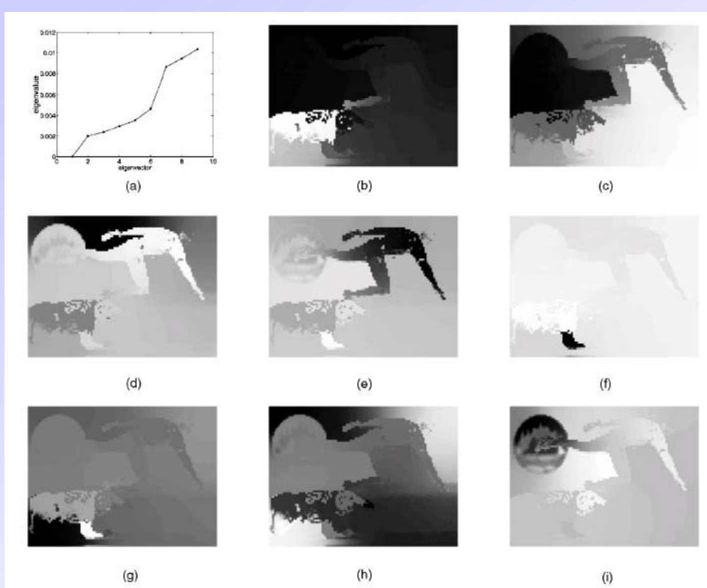


Figure from "Normalized cuts and image segmentation," Shi and Malik, copyright IEEE, 2000

Segmentation using Normalized cuts

Two algorithms:

- Recursive two-way Ncut
  - Use the second smallest eigenvector to obtain a partition to two segments.
  - Recursively apply the algorithm to each partition.
- Simultaneous K-way cut with multiple eigenvectors.
  - Use multiple (n) smallest eigenvectors as n dimensional class indicator for each pixel and apply simple clustering as k-means to obtain n clusters.

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

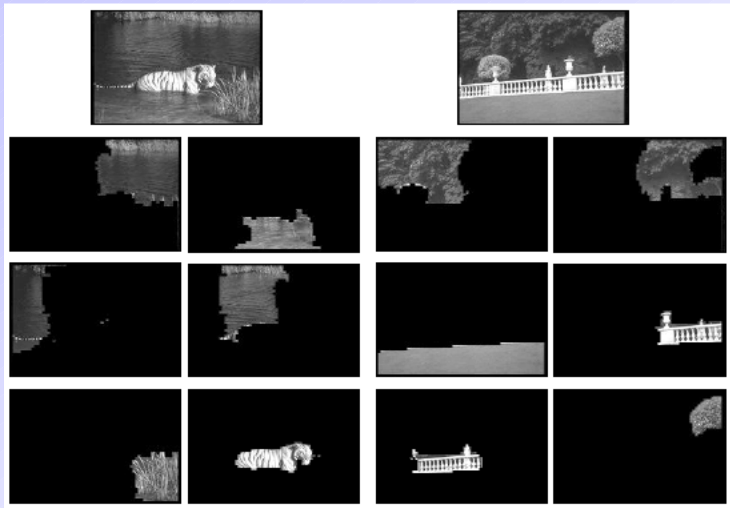
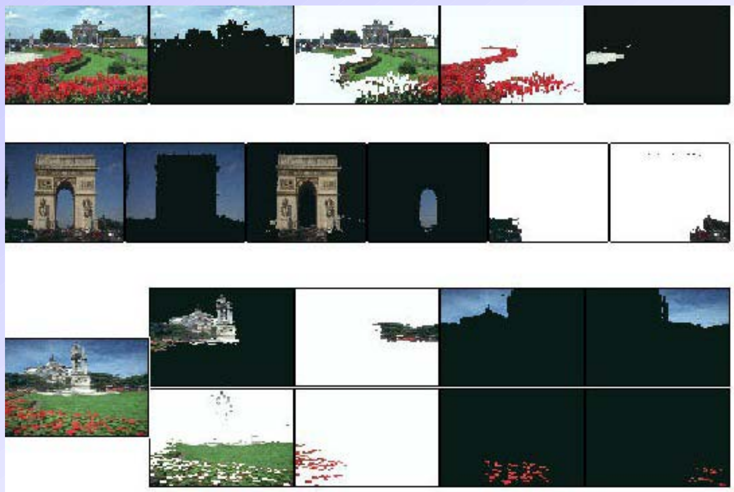


Figure from "Image and video segmentation: the normalised cut framework", by Shi and Malik, copyright IEEE, 1998

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



Sources

- Forsyth and Ponce, Computer Vision a Modern approach: chapter 14.
- Jianbo Shi and Jitendra Malik *"Normalized Cuts and Image Segmetnation"* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 22 No. 0, August 2000