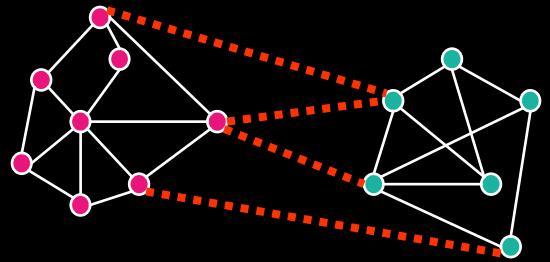
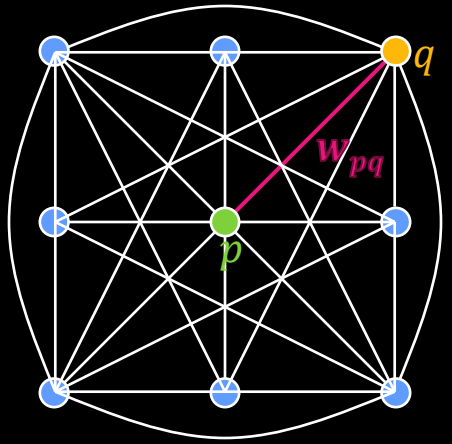
CS4495/6495
Introduction to Computer Vision

9A-L4 *Segmentation by graph partitioning*



Images as graphs



Fully-connected graph

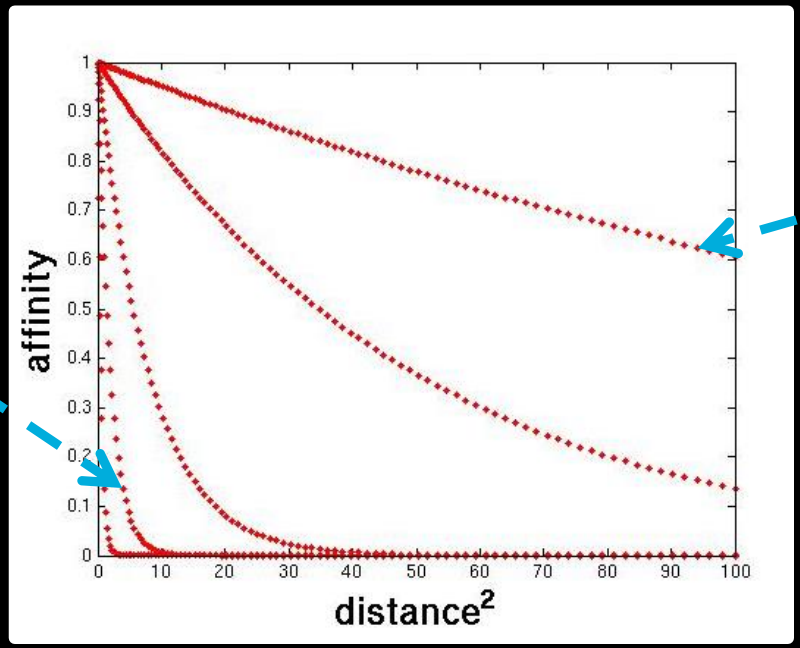
- 1 node (vertex) for every pixel
- A link between *every* pair of pixels, $\langle p, q \rangle$
- Affinity weight w_{pq} for each link (edge)
  - w_{pq} measures *similarity*: *Inversely proportional* to difference (in color and position…)

Steve Seitz

Measuring affinity

$$\text{aff}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2\sigma^2}\text{dist}(\mathbf{x}_i, \mathbf{x}_j)^2\right)$$

Small sigma: group only nearby points



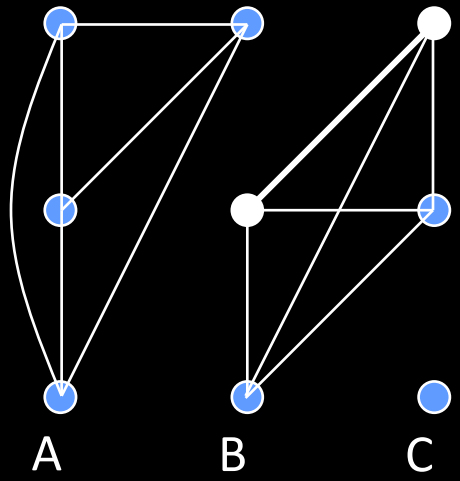
Large sigma: group distant points

Kristen Grauman

Segmentation by graph partitioning

Break Graph into *segments*

- Delete links that cross between segments
- Easiest to break links with low affinity

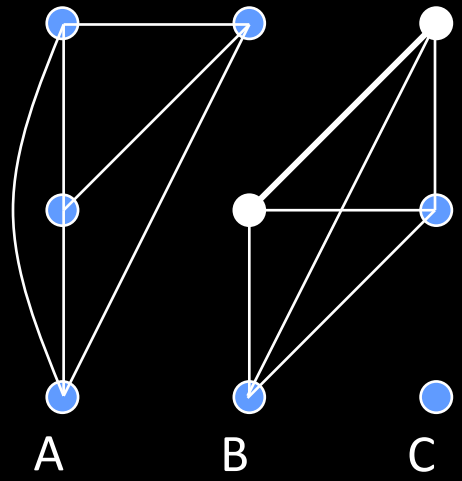


Steve Seitz

Segmentation by graph partitioning

Results

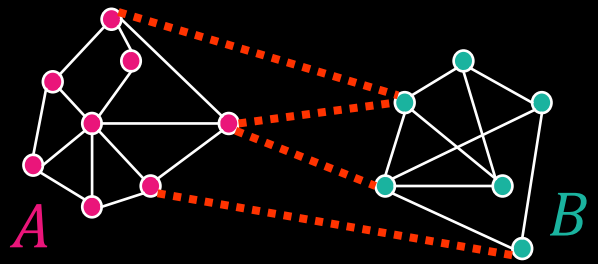
- Similar pixels should be in the same segments
- Dissimilar pixels should be in different segments



Steve Seitz

Graph cut

- Set of edges whose removal makes a graph disconnected
- Cost of a cut:
  Sum of weights of cut edges



$$cut(A, B) = \sum_{p \in A, q \in B} w_{pq}$$

A graph cut gives us a segmentation

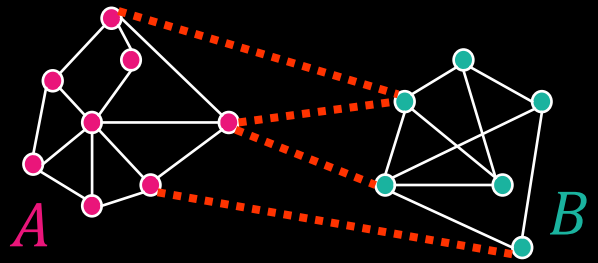
- What is a “good” graph cut and how do we find one?

Steve Seitz

A “good” graph cut

Find minimum cut

- Gives you a segmentation
- Fast min-cut algorithms exist



$$cut(A,B) = \sum_{p \in A, q \in B} w_{pq}$$

Steve Seitz

Minimum cut

Problem with min cut:

- Weight of cut proportional to number of edges in the cut
- Tends to produce small, isolated components

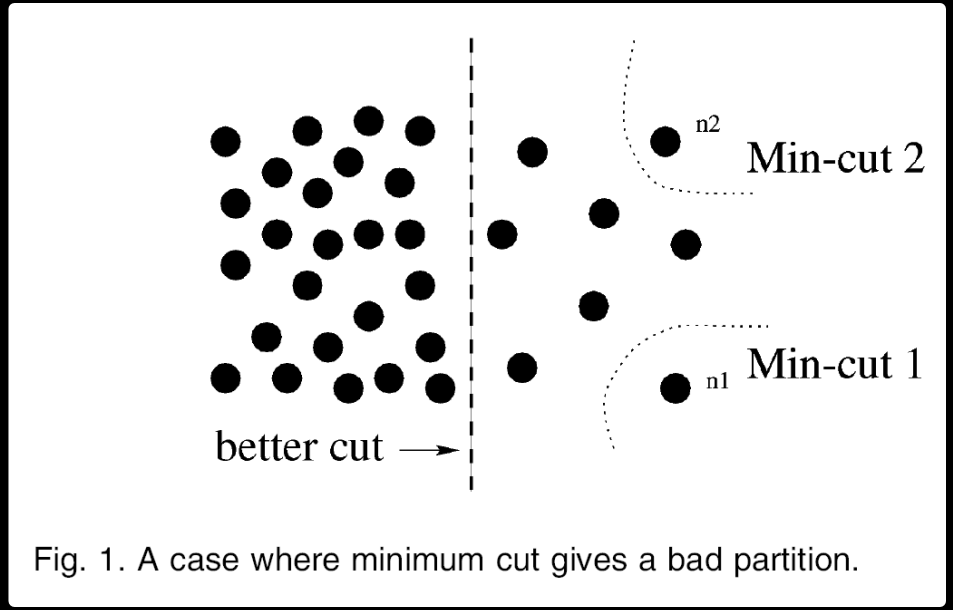


Fig. 1. A case where minimum cut gives a bad partition.

Shi and Malik, PAMI 2000

Normalized cut

Fix bias of min cut by normalizing for *size* of segments:

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

$assoc(A,V)$ = sum of weights of all edges that touch A

Approximate solution for minimizing the $Ncut$ value:
Generalized eigenvalue problem

J. Shi and J. Malik, CVPR 1997. Normalized Cuts and Image Segmentation.

Normalized cut

- Let W be the adjacency matrix of the graph
- Let D be the diagonal matrix with diagonal entries

$$D(i,i) = \sum_j W(i,j)$$

- Then the normalized cut cost can be written as

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

where y is an indicator vector with 1 in the i^{th} position if the i^{th} feature point belongs to A, negative constant otherwise

J. Shi and J. Malik, PAMI 2000. Normalized Cuts and Image Segmentation.

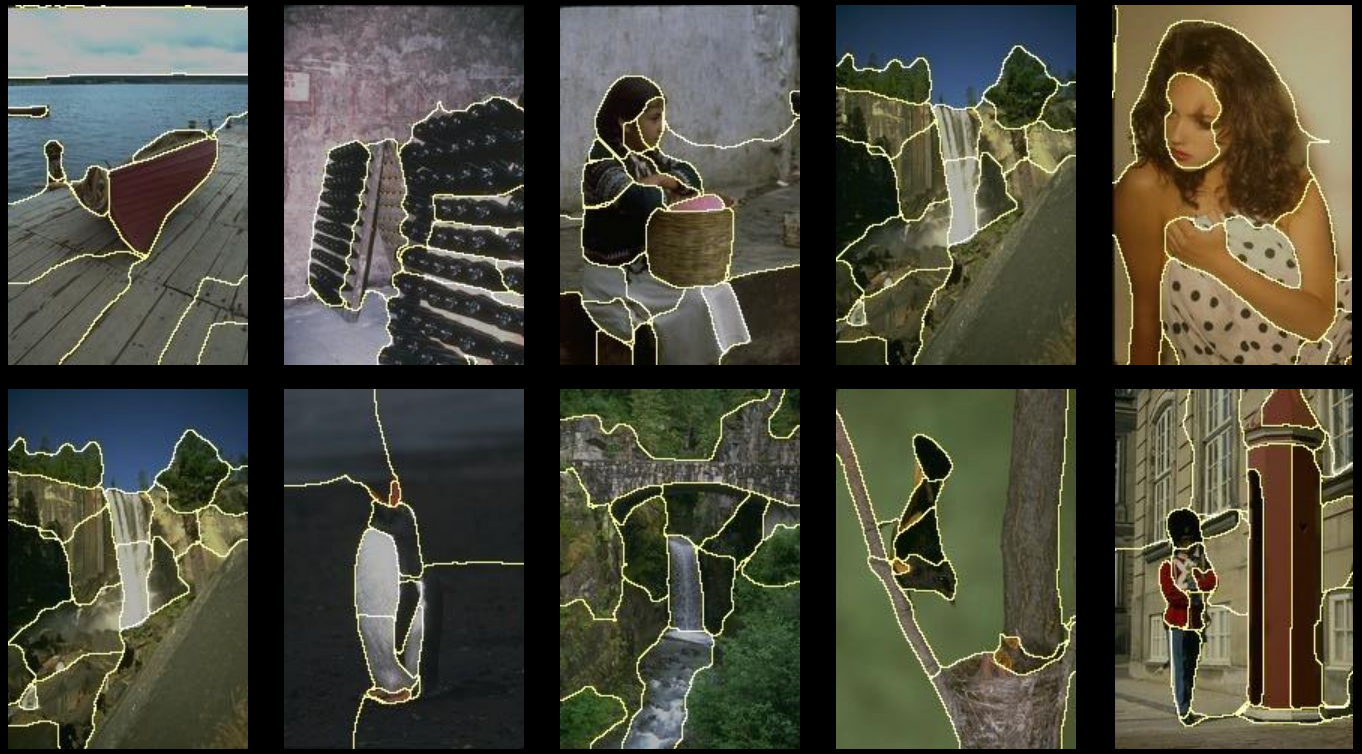
Normalized cut algorithm

1. Represent the image as a weighted graph $G = (V, E)$, compute weight of each edge and summarize in D and W

2. Solve $(D - W)y = \lambda Dy$ for the eigenvector with the second smallest eigenvalue

3. Use entries of the eigenvector to bipartition the graph

Results: Berkeley Segmentation Engine



http://www.cs.berkeley.edu/~fowlkes/BSE/

Normalized cuts: Pros and cons

Pros:

- Generic framework
  - Flexible to choice of function that computes weights (“affinities”) between nodes
- Does not require model of the data distribution

Normalized cuts: Pros and cons

Cons:

- Time complexity can be high
  - Dense, highly connected graphs
    → many affinity computations
  - Solving eigenvalue problem
- Preference for balanced partitions