CMPSCI 670: Computer Vision
Grouping continued …

University of Massachusetts, Amherst
October 15, 2014

Instructor: Subhransu Maji

Slides credit: Kristen Grauman and others

Last time ...

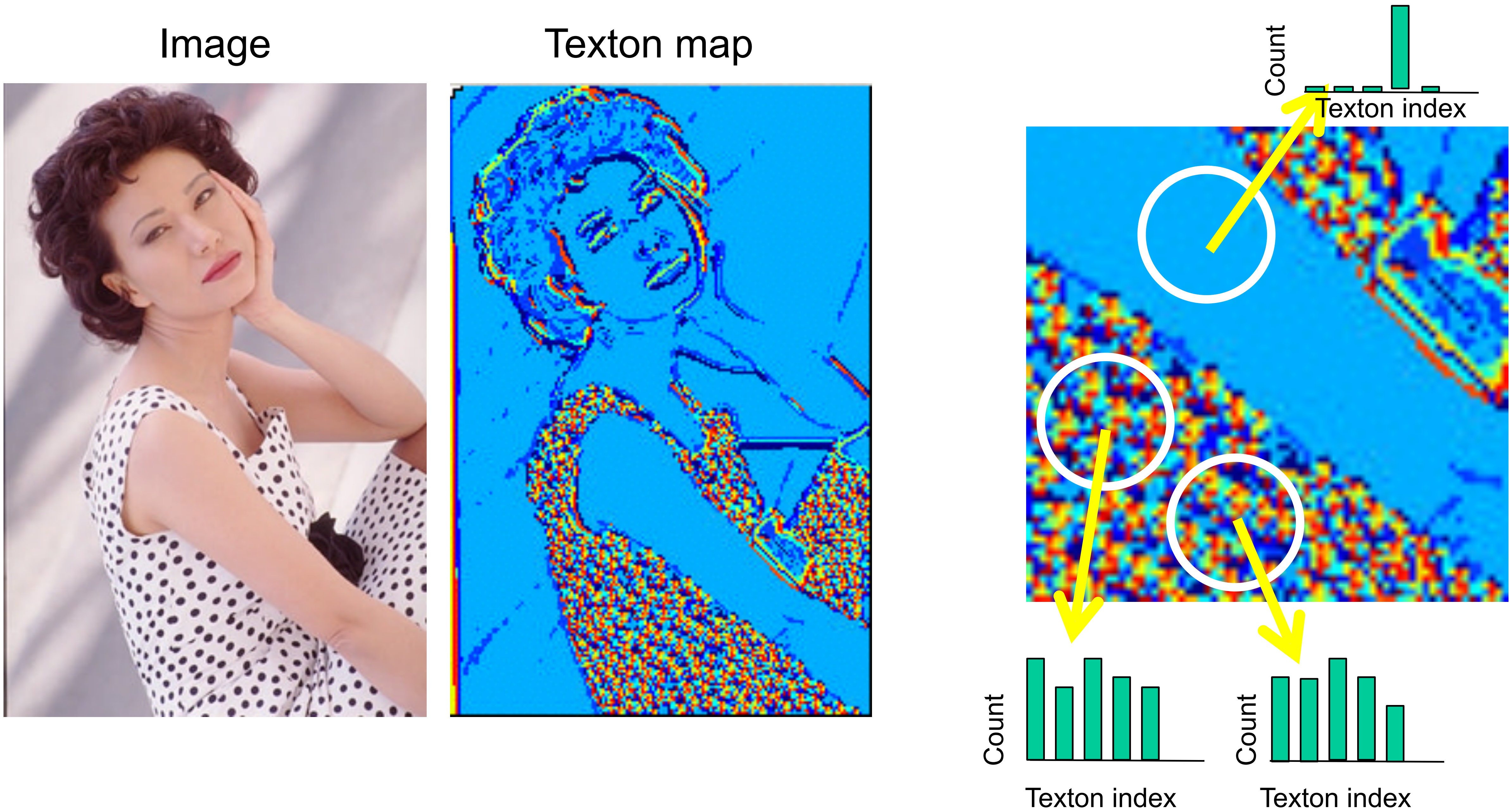
- What are grouping problems in vision?

- Inspiration from human perception
  - Gestalt properties

- Bottom-up segmentation via clustering
  - Algorithms:
    - Mode finding and mean shift: k-means, mean-shift
    - Graph-based: graph cuts, normalized cuts
  - Features: color, texture, …
    - Quantization for texture summaries

Segmentation with texture features

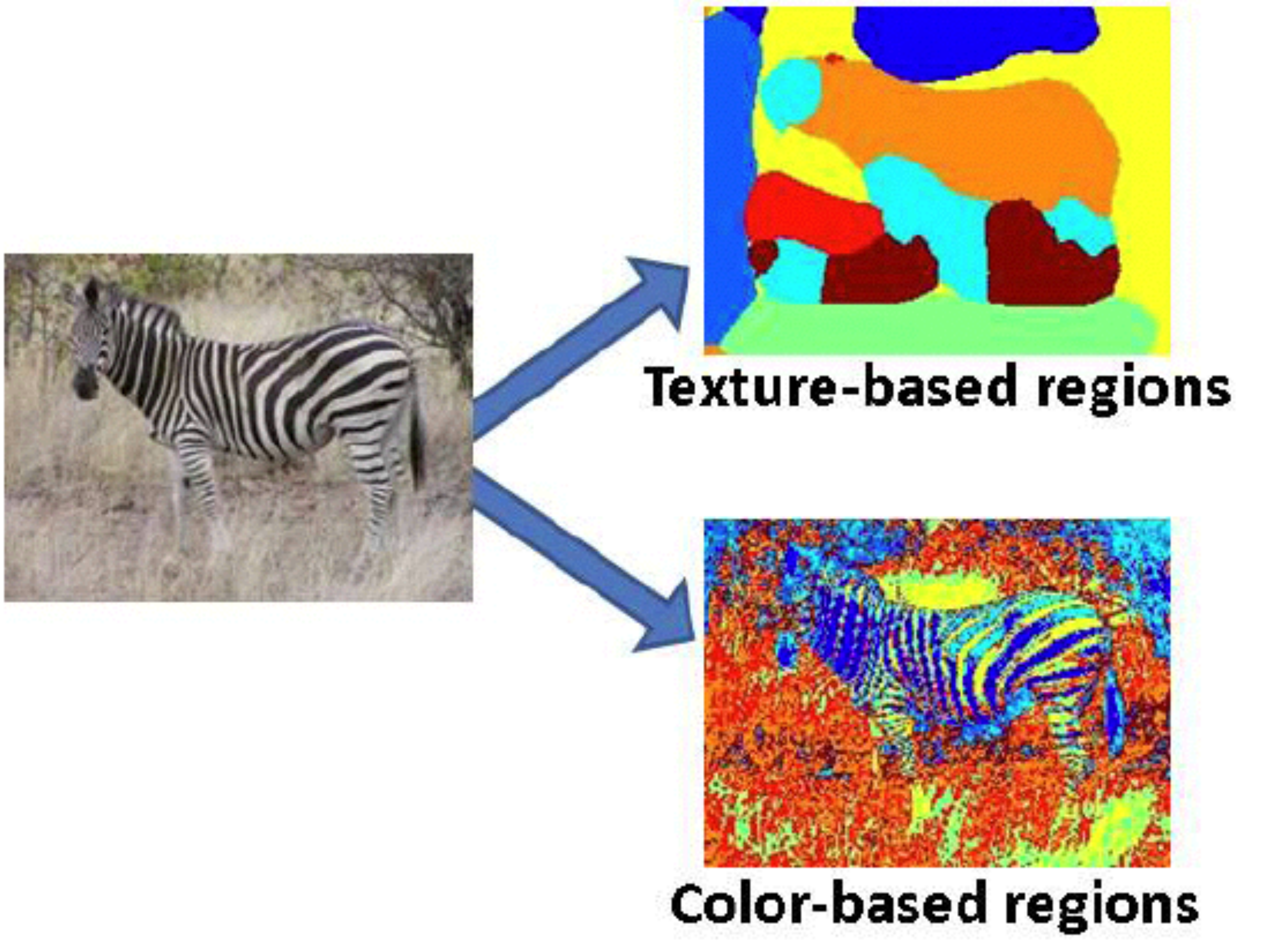
- Find “textons” by **clustering** vectors of filter bank outputs
- Describe texture in a window based on *texton histogram*



Malik, Belongie, Leung and Shi. IJCV 2001.

Adapted from Lana Lazebnik

Image segmentation example

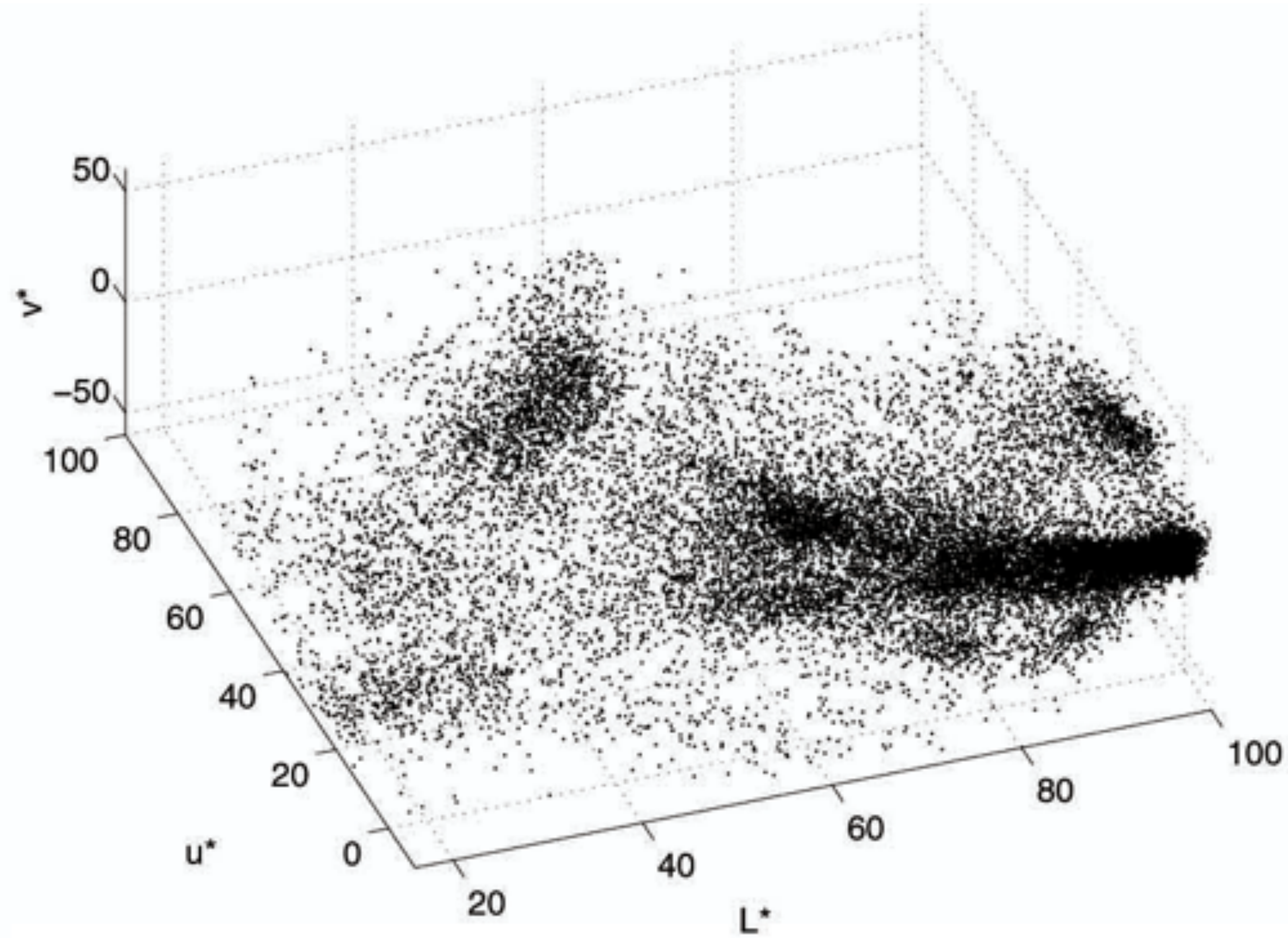


Mean shift algorithm

- The mean shift algorithm seeks *modes* or local maxima of density in the feature space

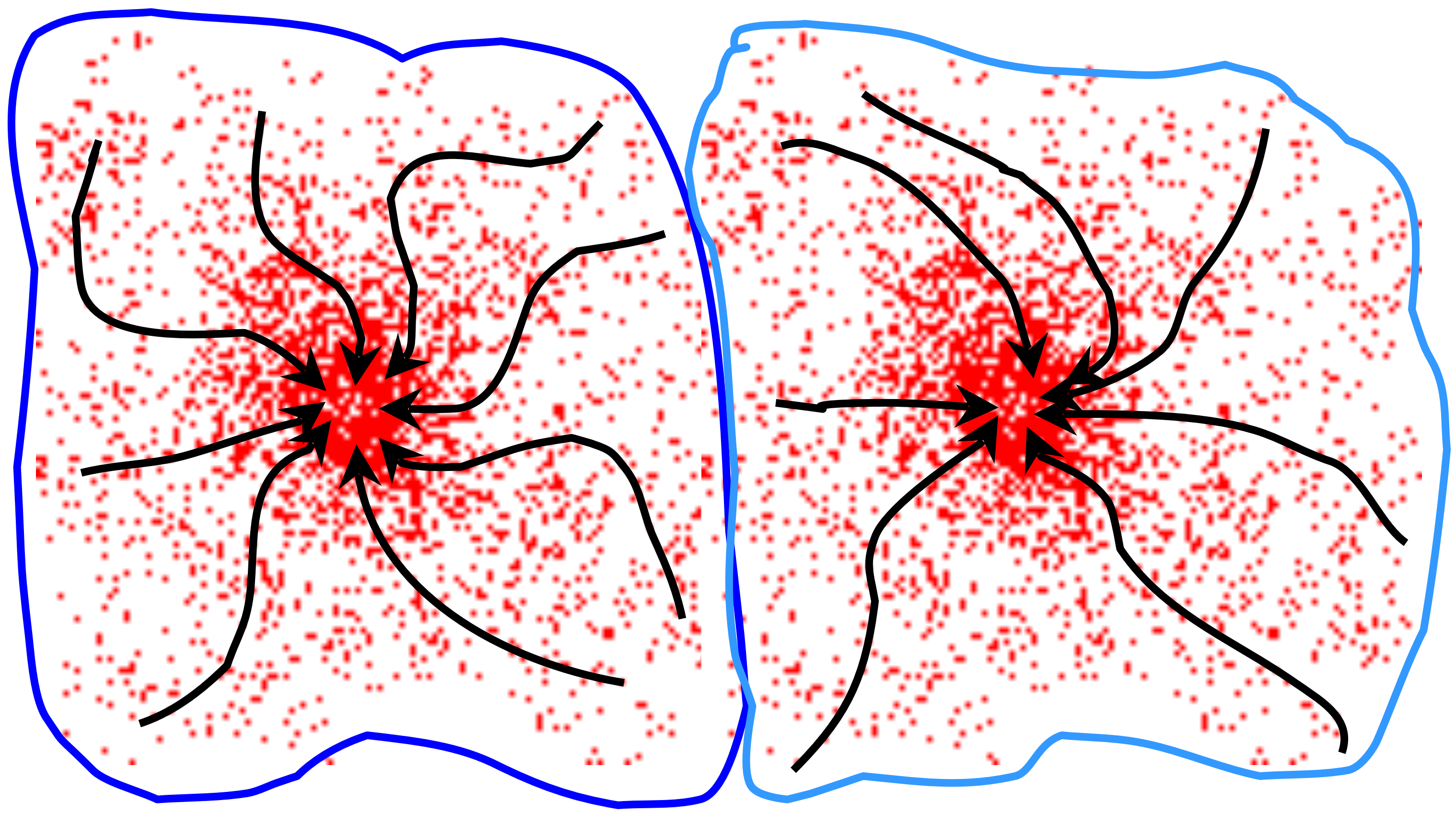
image

Feature space
(L*u*v* color values)



Mean shift clustering

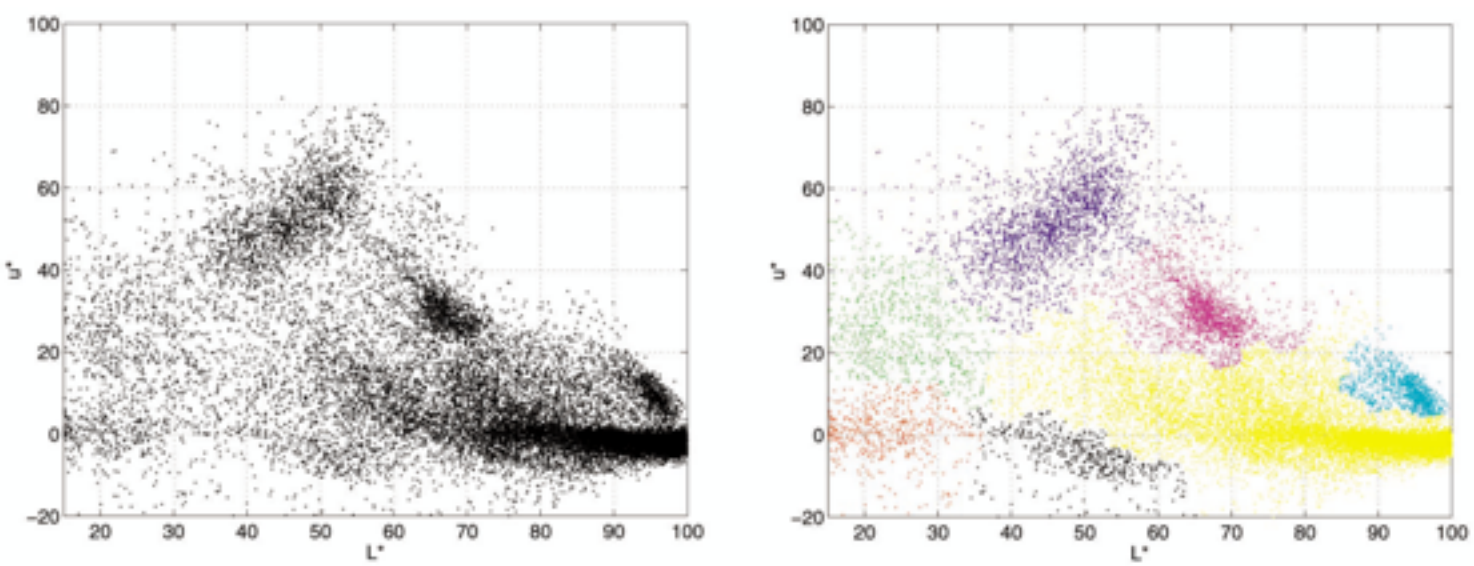
- Cluster: all data points in the attraction basin of a mode
- Attraction basin: the region for which all trajectories lead to the same mode



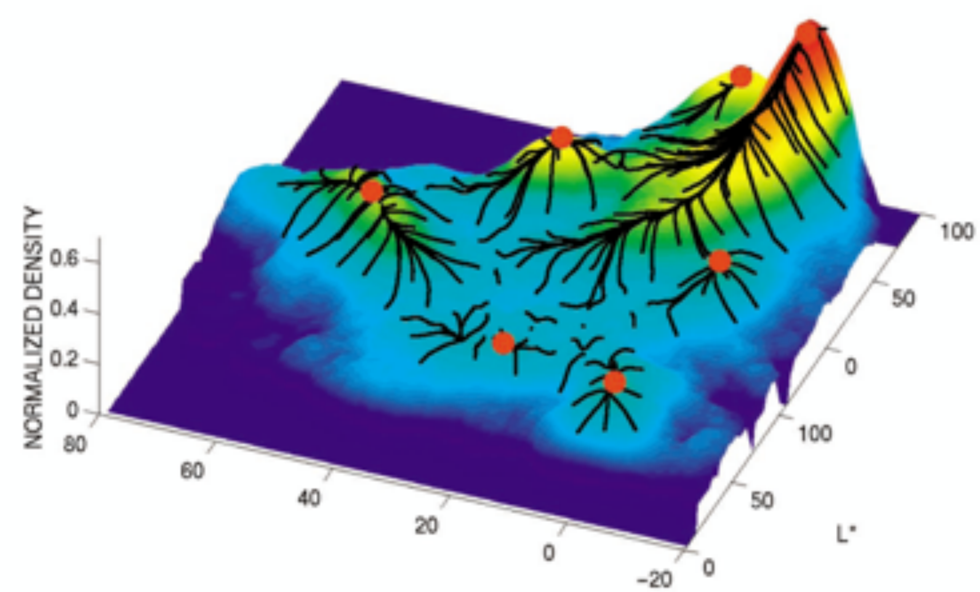
Slide by Y. Ukrainitz & B. Sarel

Mean shift clustering/segmentation

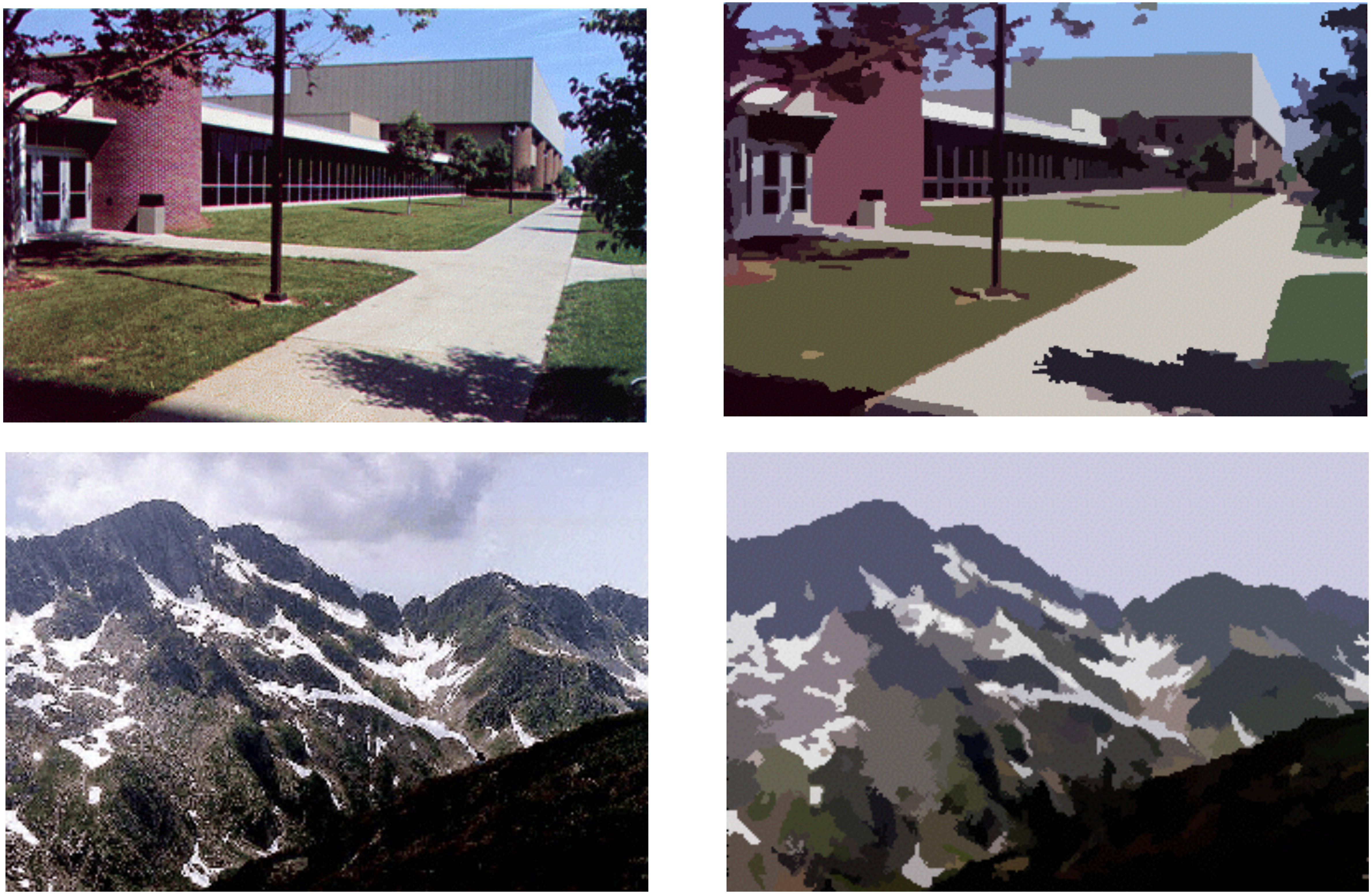
- Find features (color, gradients, texture, etc)
- Initialize windows at individual feature points
- Perform mean shift for each window until convergence
- Merge windows that end up near the same “peak” or mode



(a) (b)



Mean shift segmentation results



http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html

Mean shift clustering results




Outline

- What are grouping problems in vision?

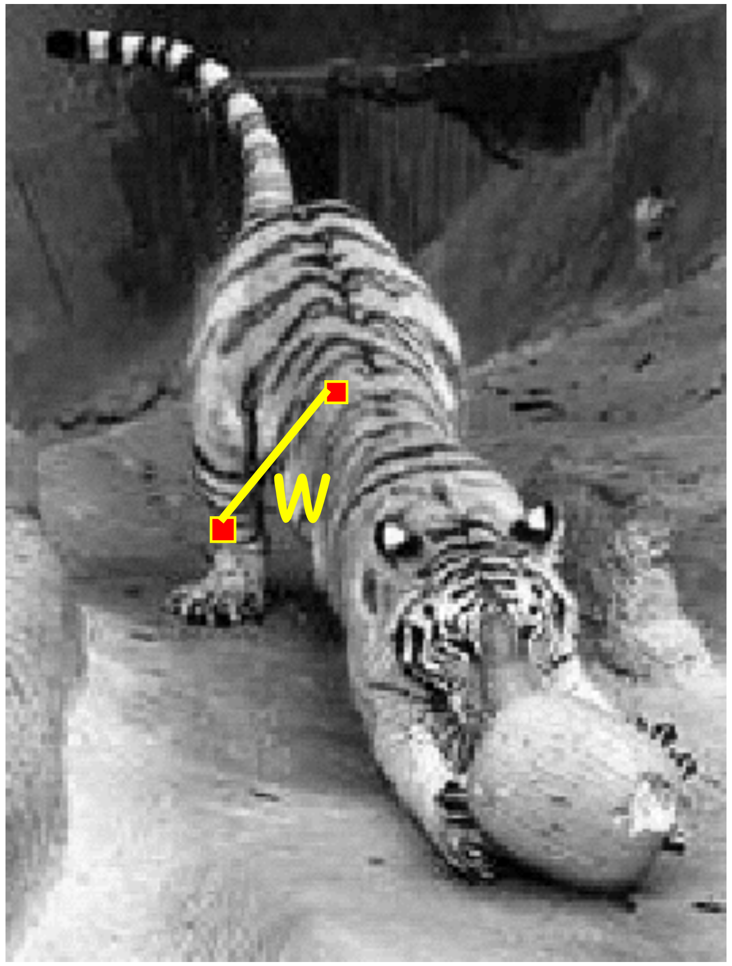
- Inspiration from human perception
  - Gestalt properties

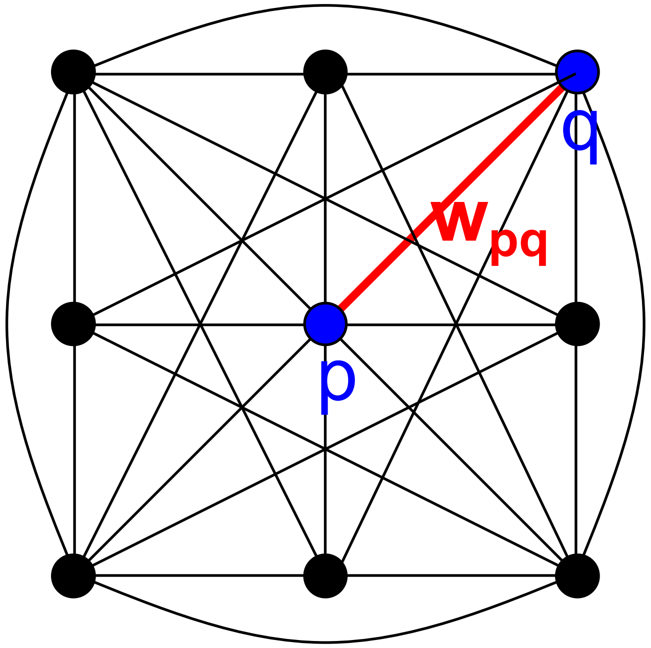
- Bottom-up segmentation via clustering
  - Algorithms:
    - Mode finding and mean shift: k-means, mean-shift
    - Graph-based: graph cuts, normalized cuts
  - Features: color, texture, …
    - Quantization for texture summaries

Images as graphs

Fully-connected graph

- node (vertex) for every pixel
- link between ***every*** pair of pixels, **p**,**q**
- affinity weight $\mathbf{w_{pq}}$ for each link (edge)
  - $\mathbf{w_{pq}}$ measures *similarity*
    - » similarity is *inversely proportional* to difference (in color and position…)





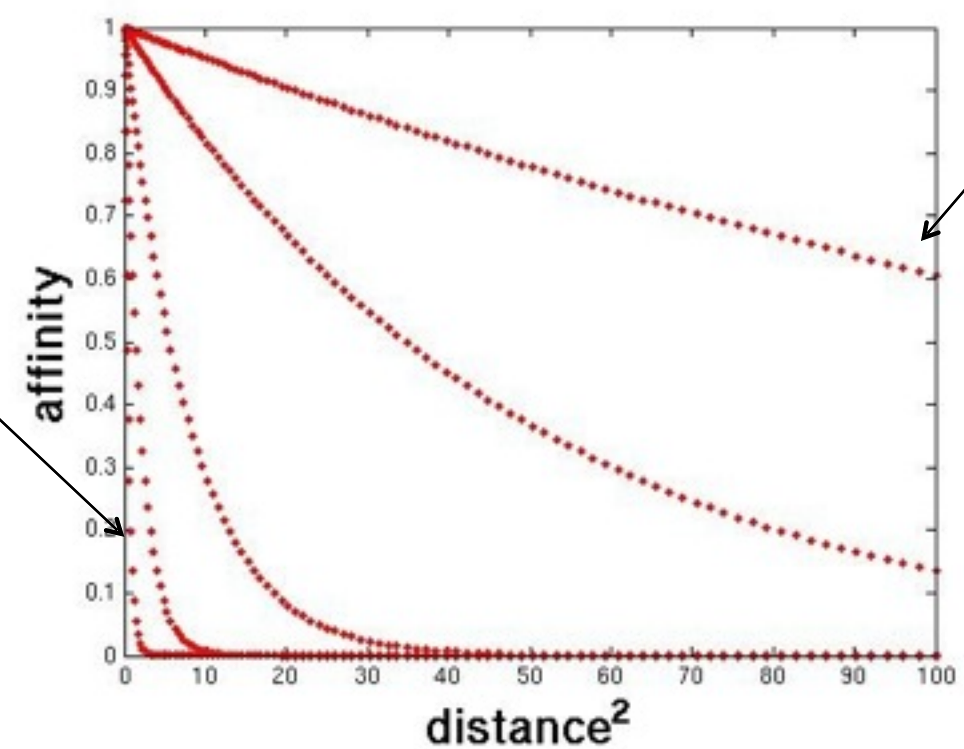
Source: Steve Seitz

Measuring affinity

- One possibility:

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

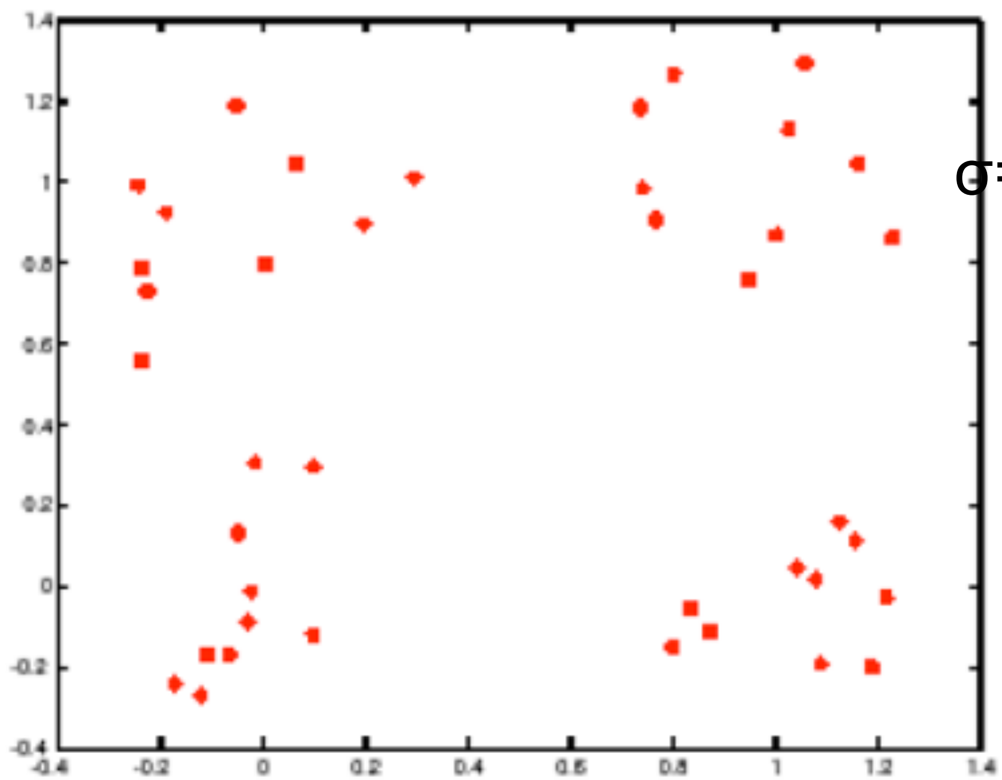
Small sigma: group only nearby points



Large sigma: group distant points

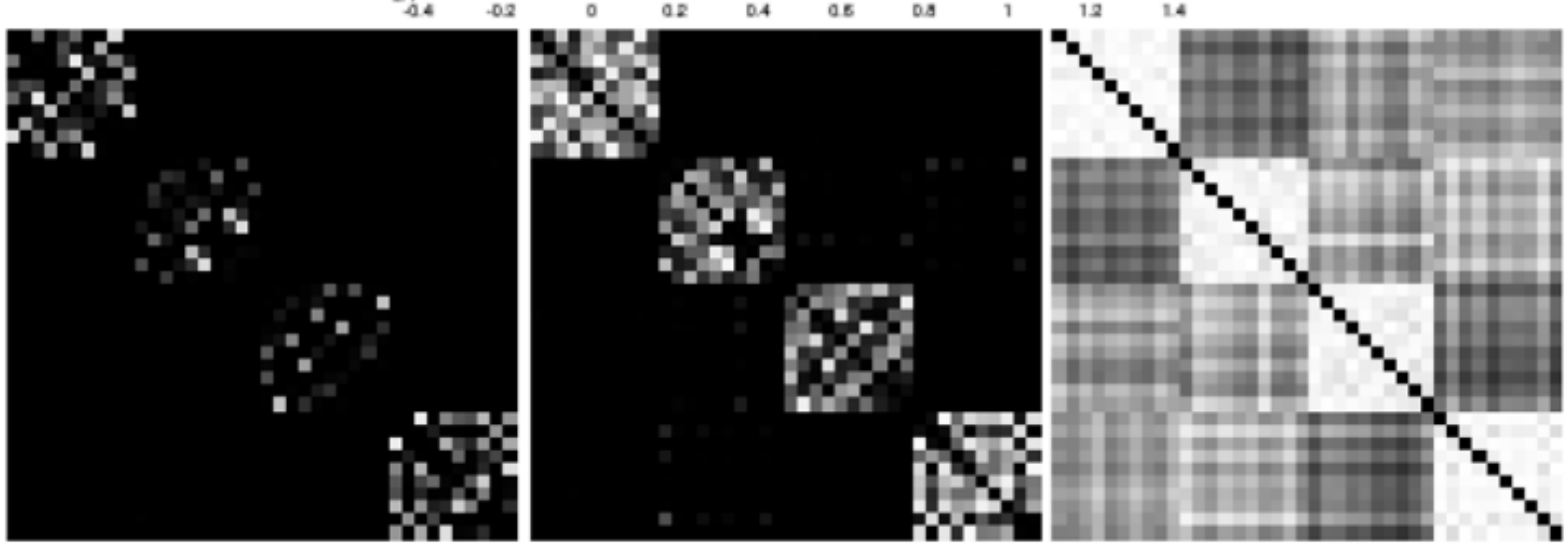
Measuring affinity

Data points



σ=.2

Affinity matrices

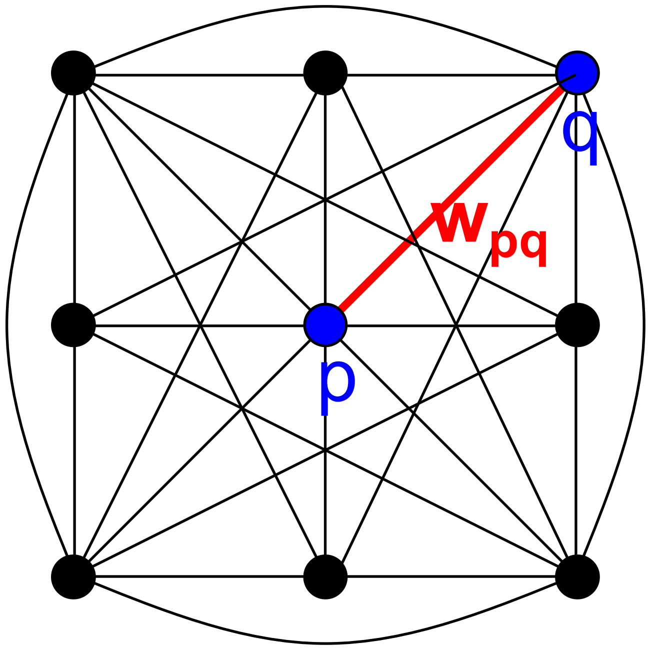


σ=.1 σ=.2 σ=1

Segmentation by Graph Cuts

Break graph into segments

- Want to delete links that cross **between** segments
- Easiest to break links that have low similarity (low weight)
  - similar pixels should be in the same segments
  - dissimilar pixels should be in different segments

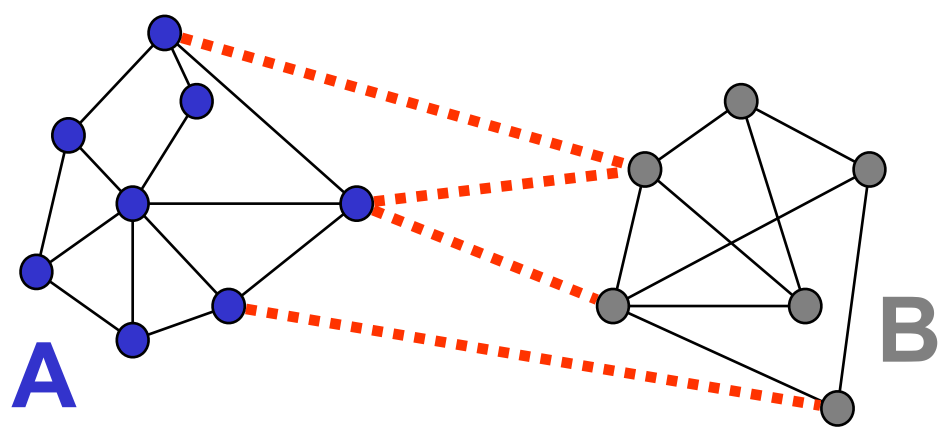


Source: Steve Seitz

Cuts in a graph: Min cut

Link Cut

- set of links whose removal makes a graph disconnected
- cost of a cut:



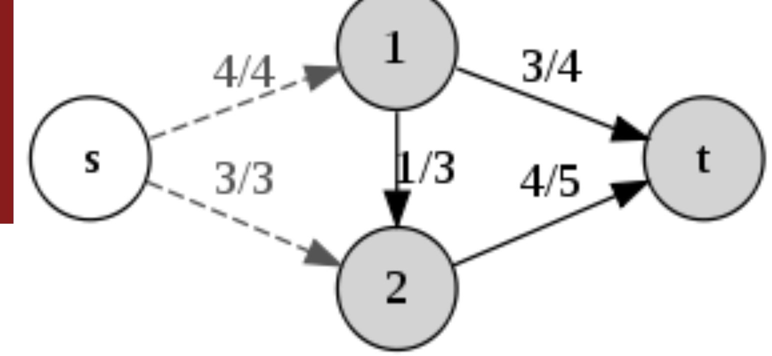
$$cut(A,B) = \sum_{p \in A, q \in B} w_{p,q}$$

Find minimum cut

- gives you a segmentation
- fast algorithms exist for doing this (max flow/min cut algorithms)

Source: Steve Seitz

Max-flow/min-cut



- Flow
  - A **flow** is a mapping $f: E \to \mathbf{R}^{+}$, denoted by f_{uv} or $f(u, v)$, subject to the following two constraints:

$\forall (u,v) \in E: \qquad f_{uv} \le c_{uv}$ capacity constraint

$\forall v \in V \setminus \{s,t\}: \qquad \sum_{\{u:(u,v)\in E\}} f_{uv} = \sum_{\{u:(v,u)\in E\}} f_{vu}.$ conservation of flow

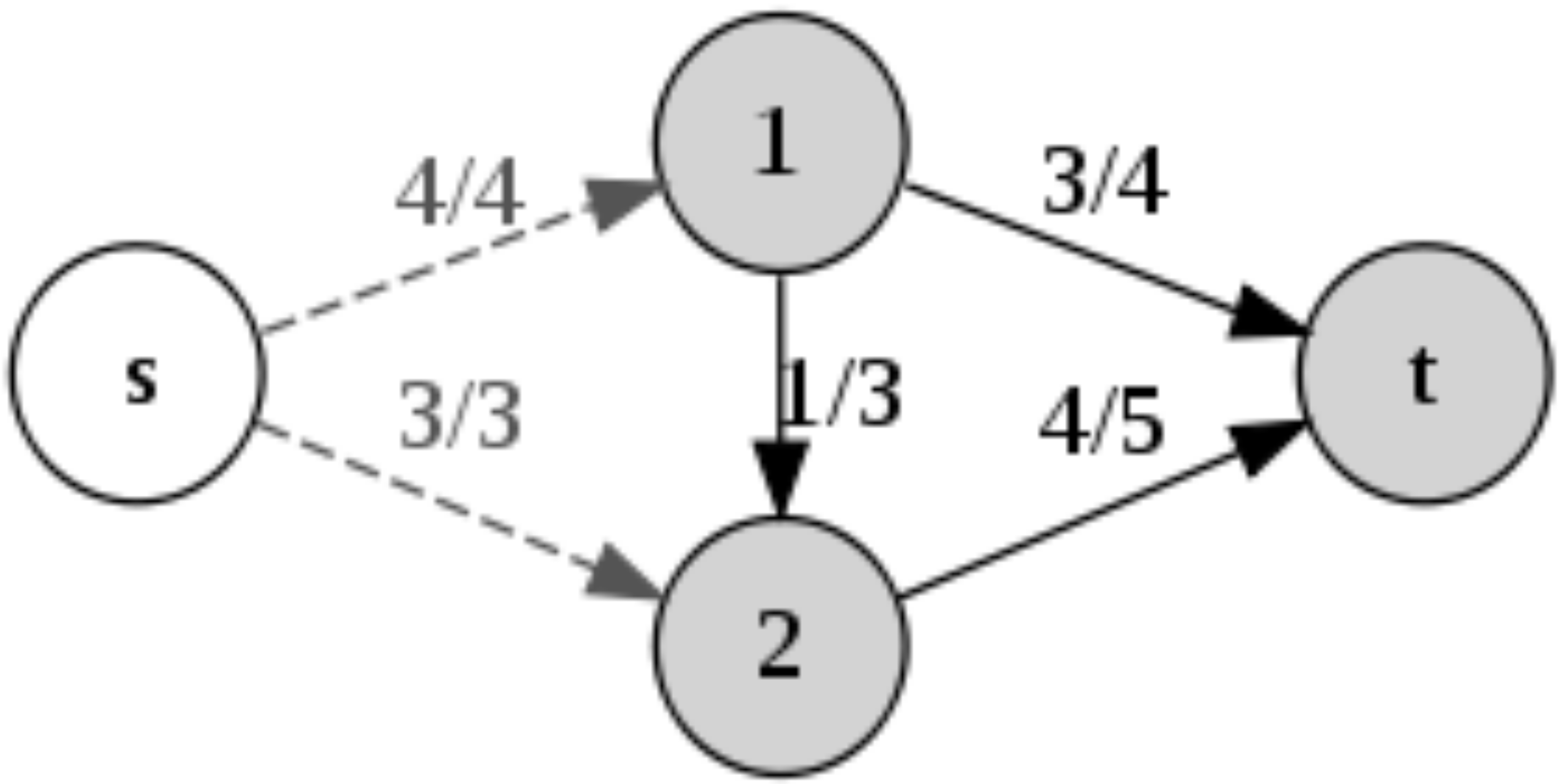
  - The **value** of a flow is defined as: $|f| = \sum_{v\in V} f_{sv},$

- Cut
  - *An* ***s-t cut*** *$C = (S, T)$ is a partition of V such that $s \in S$ and $t \in T$. The* ***cut-set*** *of C is the set* $\{(u,v) \in E \ : \ u \in S, v \in T\}.$
  - *The* ***capacity*** *of an s-t cut is defined by* $c(S,T) = \sum_{(u,v)\in S\times T} c_{uv}.$

Max-Flow Min-Cut Theorem. The maximum value of an s-t flow is equal to the minimum capacity over all s-t cuts.

Example

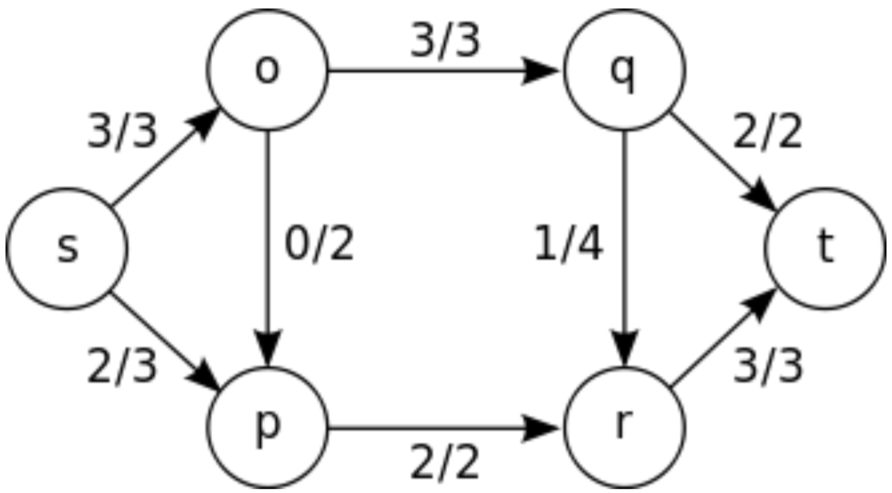
Flow = 7



- A network with flow value equal to the capacity of the s-t cut

Algorithms for max flow/min cut

- Polynomial time algorithms exist for computing max-flow
- Ford-Fulkerson algorithm
    - **basic idea:** iterate
        - find a path from source to sink
        - "delete" this path from the graph



- Better variant: Edmonds-Karp algorithm
- For images which have a more structure specialized variants of these algorithms exist
    - Boykov and Kolmogorov, **An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision, PAMI 2004**

Minimum cut

- Problem with minimum cut:

  Weight of cut proportional to number of edges in the cut; tends to produce small, isolated components.

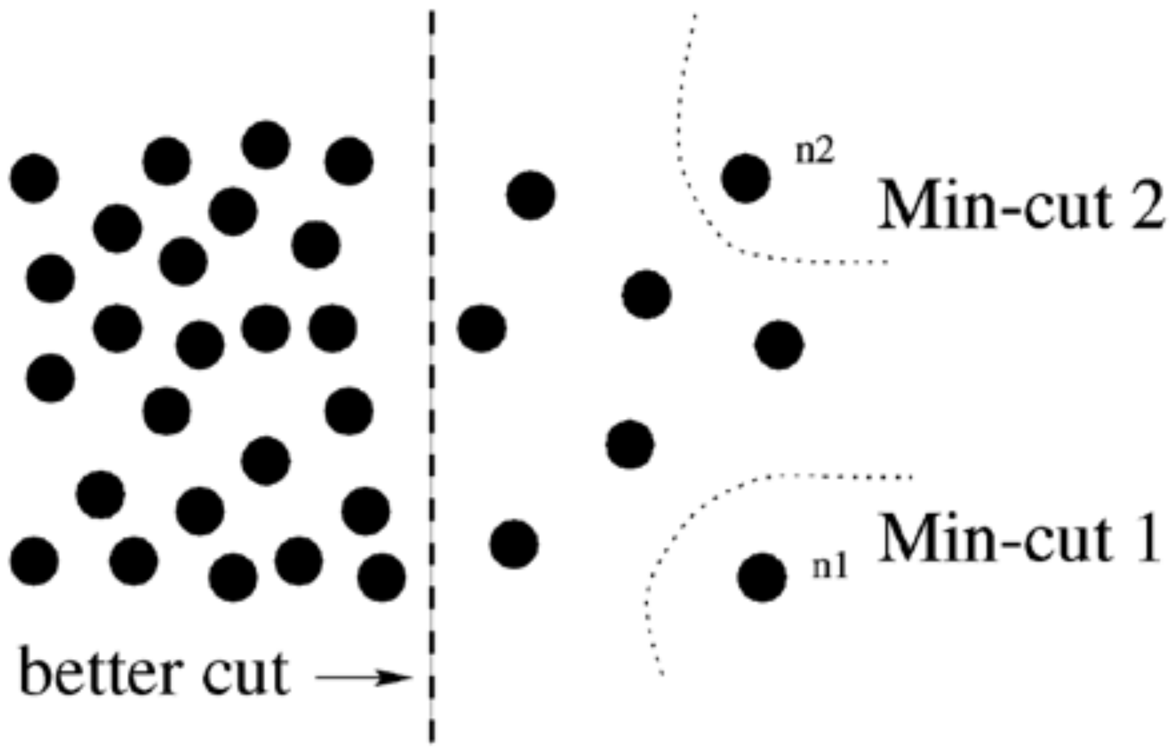
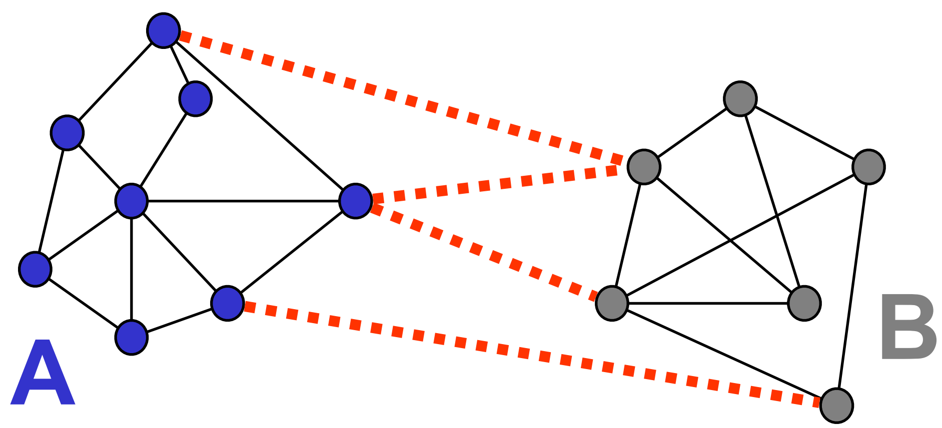


Fig. 1. A case where minimum cut gives a bad partition.

[Shi & Malik, 2000 PAMI]

Cuts in a graph: Normalized cut



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

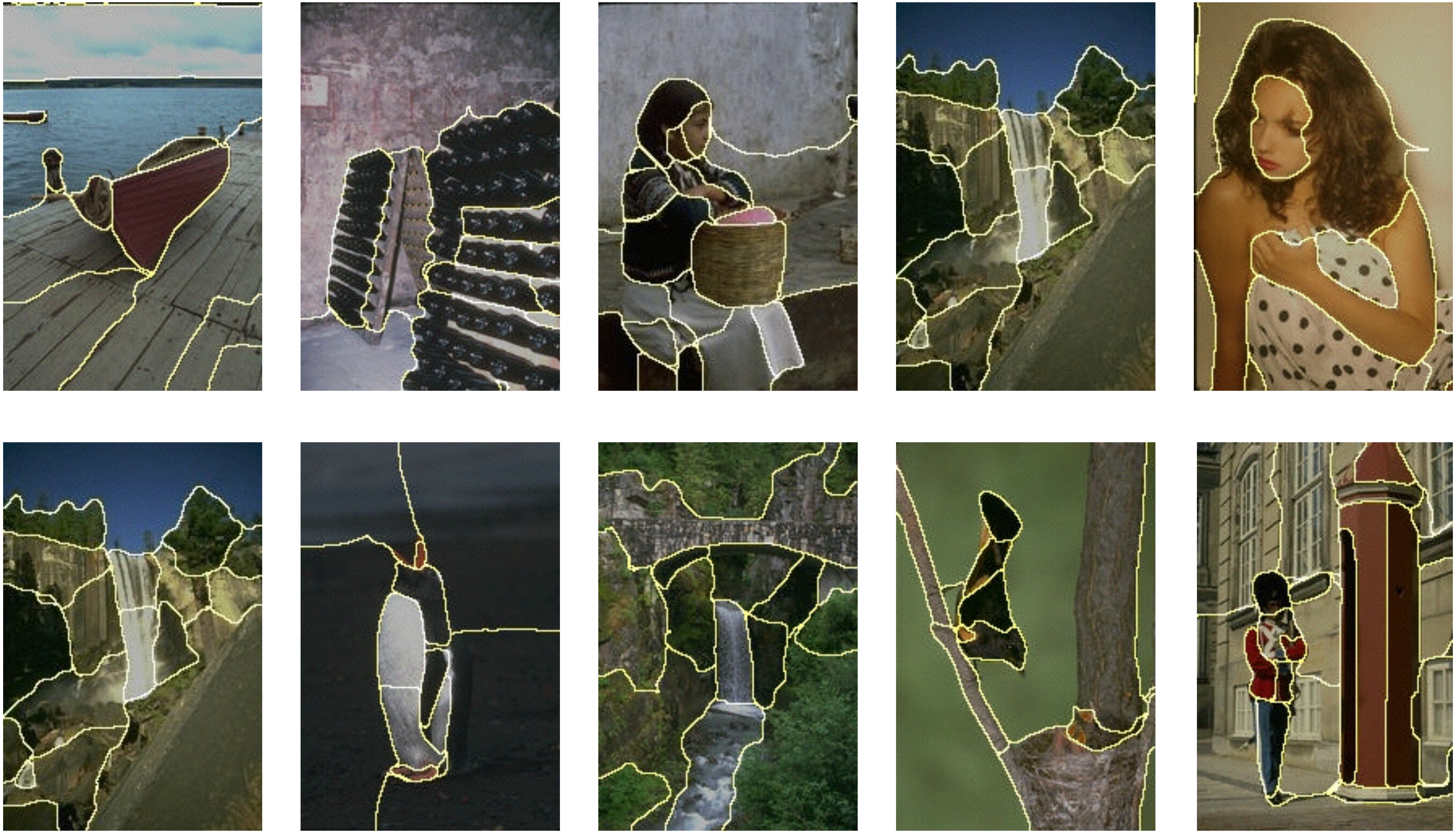
- Ncut value small when we get two clusters with many edges with high weights, and few edges of low weight between them
- Approximate solution for minimizing the n-cut value : generalized eigenvalue problem.

J. Shi and J. Malik, [Normalized Cuts and Image Segmentation](), CVPR, 1997

Source: Steve Seitz

Example results

Results: Berkeley Segmentation Engine



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

Normalized cuts: pros and cons

Pros:

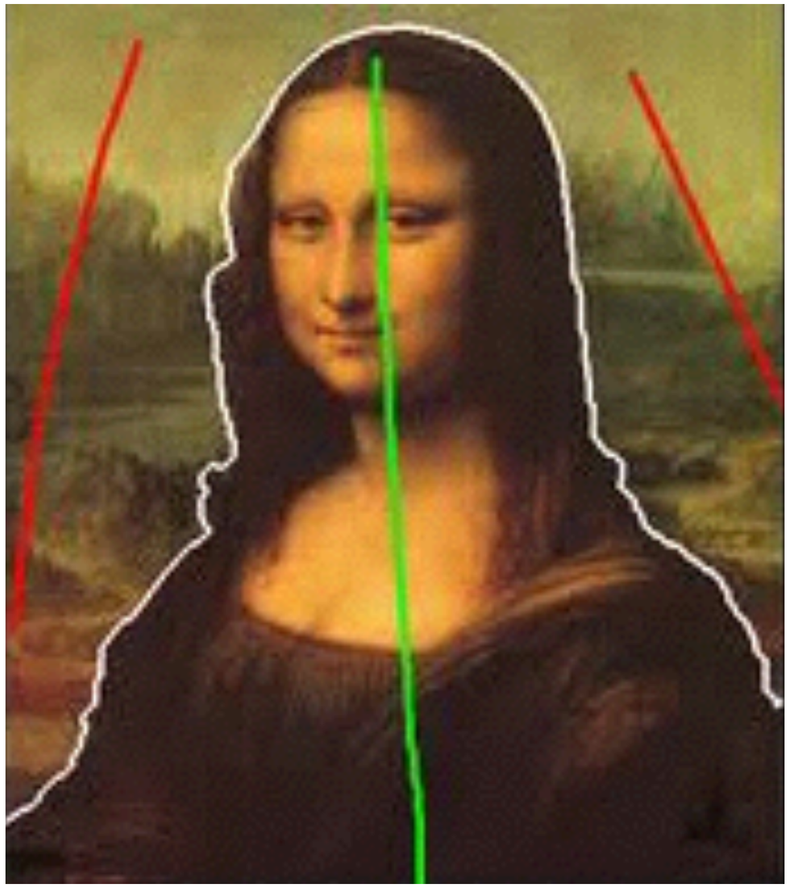
- Generic framework, flexible to choice of function that computes weights ("affinities") between nodes
- Does not require model of the data distribution

Cons:

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

Image segmentation with priors

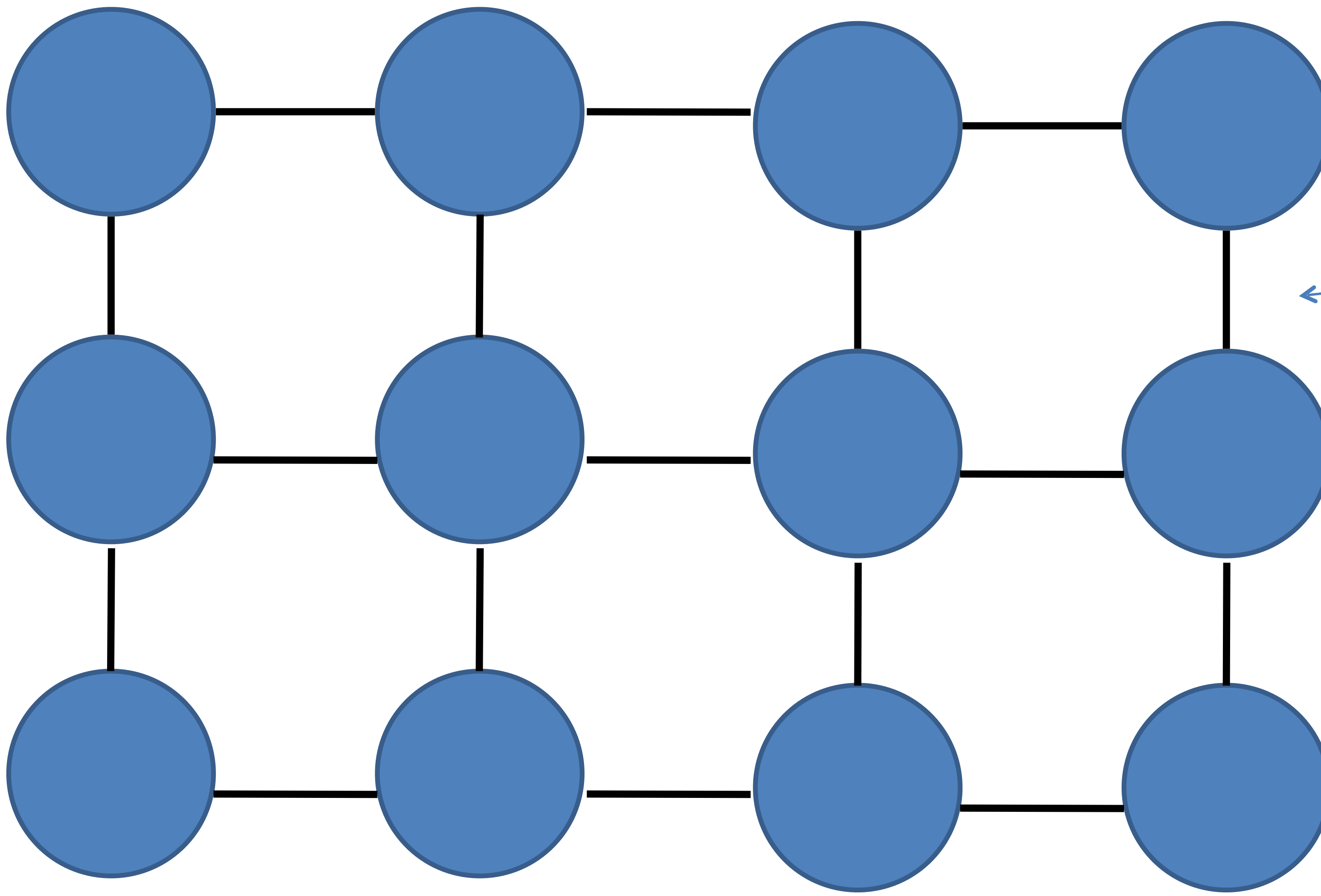
- Often we want to incorporate prior information
  - User input in interactive applications
  - Shape priors, e.g., we want a round object



Constrains the set of possible segmentations

Markov Random Fields

Node y_i: pixel label



Edge: constrained pairs

Cost to assign a label to each pixel

Cost to assign a pair of labels to connected pixels

$$Energy(\mathbf{y};\theta, data) = \sum_i \psi_1(y_i;\theta, data) + \sum_{i,j \in edges} \psi_2(y_i, y_j;\theta, data)$$

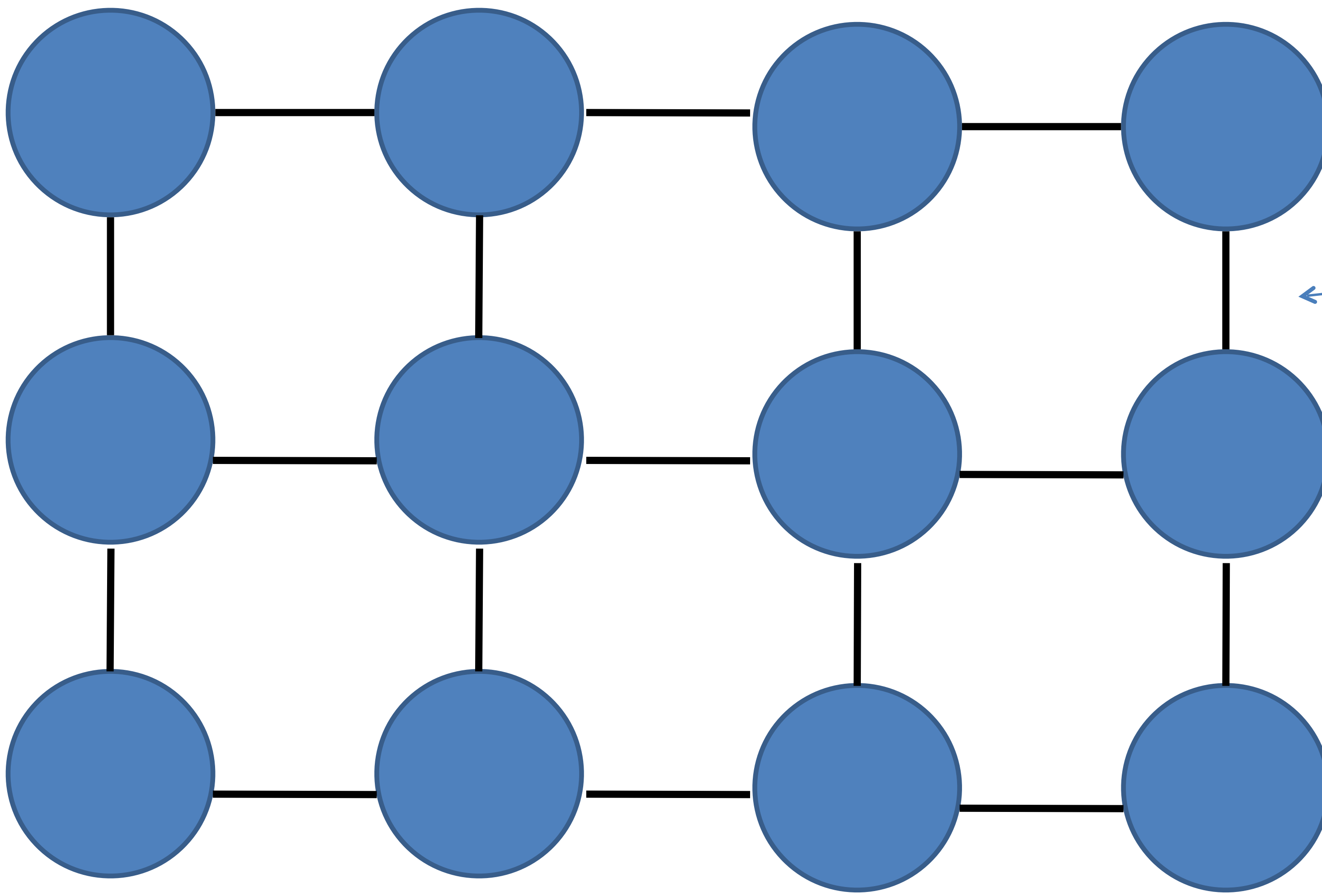
Markov Random Fields

- Example: "label smoothing" grid

Unary potential

0: -logP(y_i = 0 ; data)

1: -logP(y_i = 1 ; data)



Pairwise Potential

	0	1
0	0	K
1	K	0

$$Energy(\mathbf{y};\theta, data) = \sum_i \psi_1(y_i;\theta, data) + \sum_{i,j \in edges} \psi_2(y_i, y_j;\theta, data)$$

Image de-noising revisited

- Find X that minimizes the energy E(X)

Assigns X_v $(= 0 \text{ or } 1)$ to each pixel v

$$E(X) = \underbrace{\sum_{v \in V}}_{\text{All pixels}} \lambda \, | Y_v - X_v | + \underbrace{\sum_{(u,v) \in E}}_{\text{Neighboring}} \kappa \, | X_u - X_v |$$

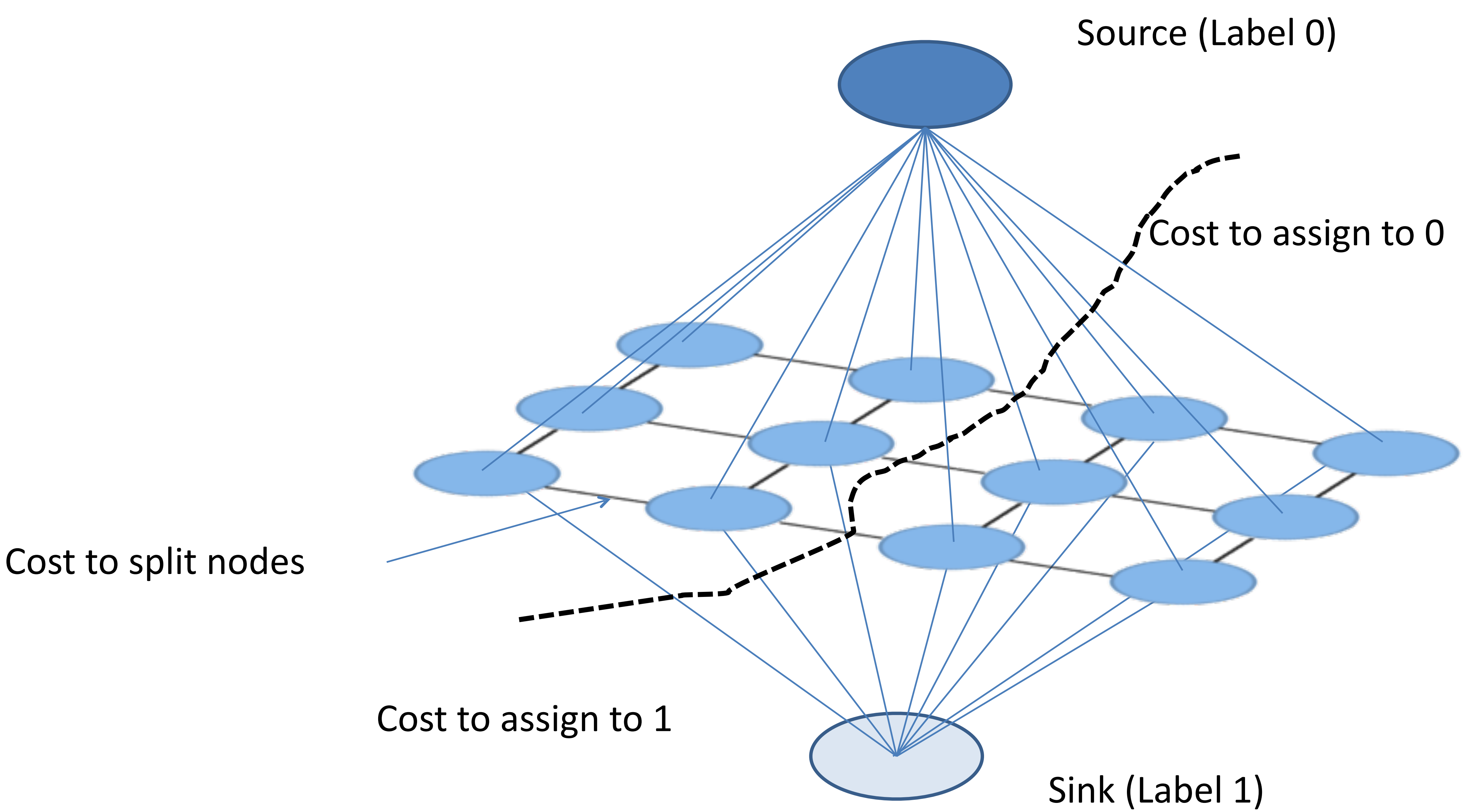
Y

Close to Y

X

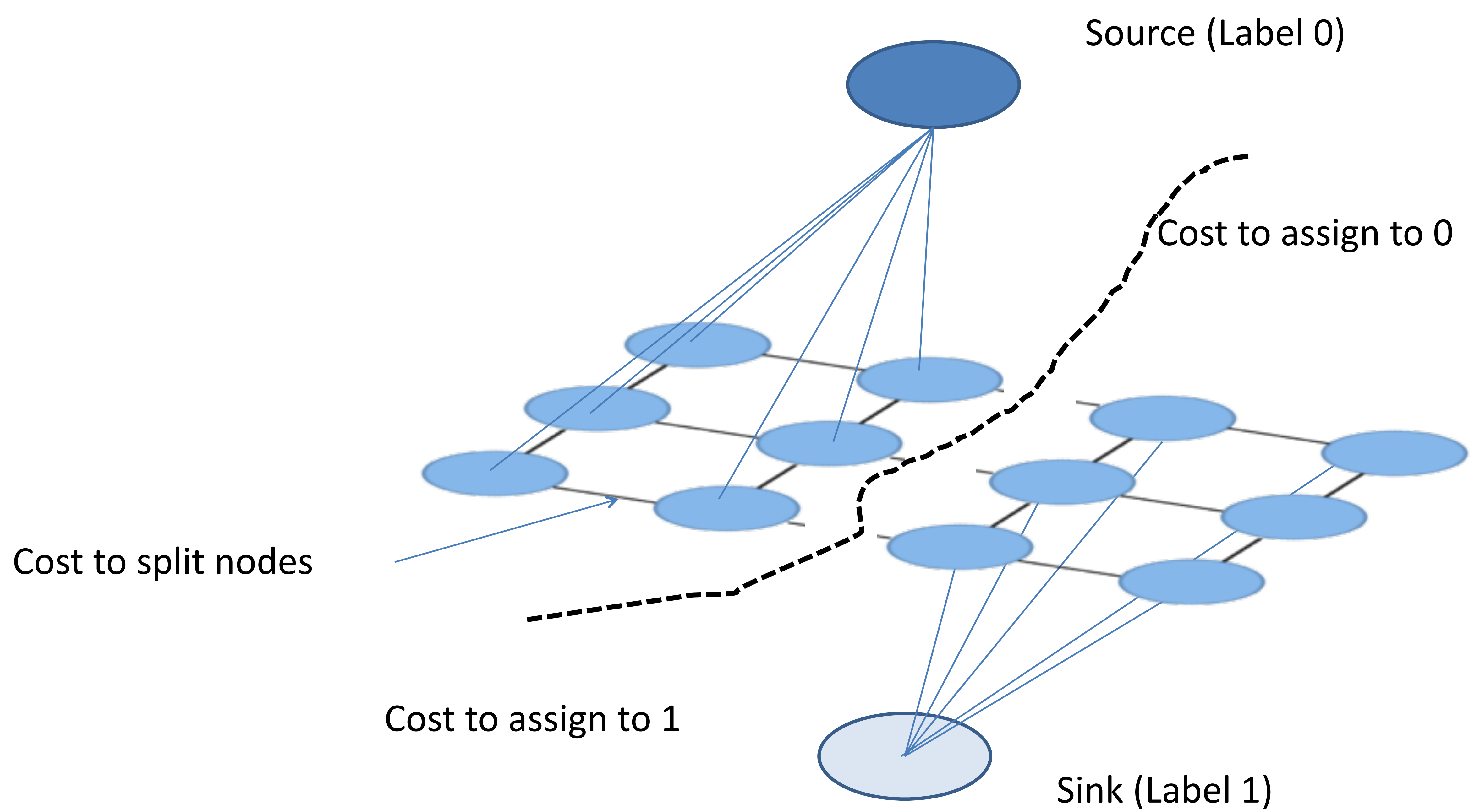
Doesn't change between pixels

Solving MRFs with graph cuts



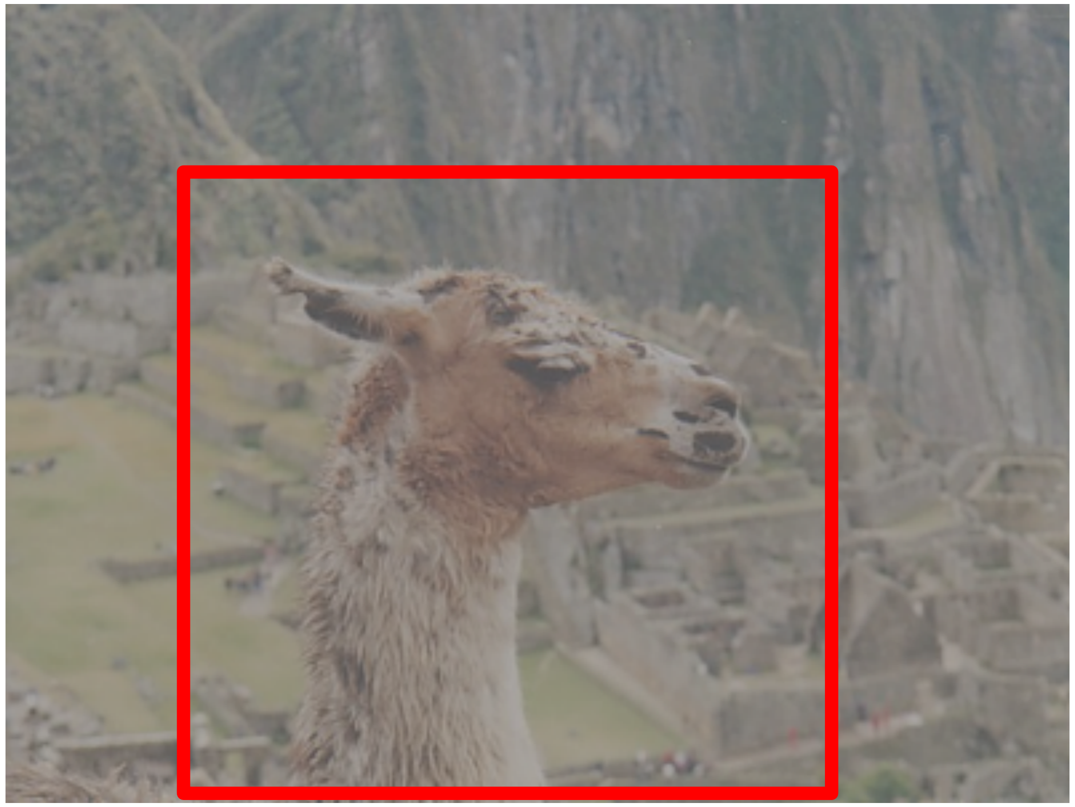
$$Energy(\mathbf{y};\theta, data) = \sum_i \psi_1(y_i;\theta, data) + \sum_{i,j \in edges} \psi_2(y_i, y_j;\theta, data)$$

Solving MRFs with graph cuts



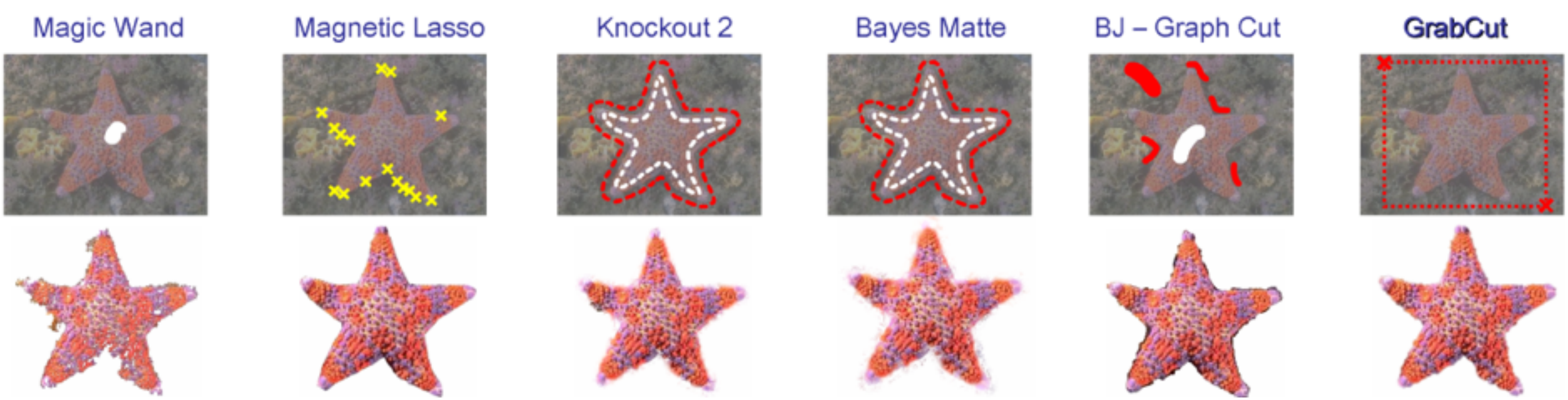
$$Energy(\mathbf{y};\theta, data) = \sum_i \psi_1(y_i;\theta, data) + \sum_{i,j \in edges} \psi_2(y_i, y_j;\theta, data)$$

Grabcut for interactive segmentation

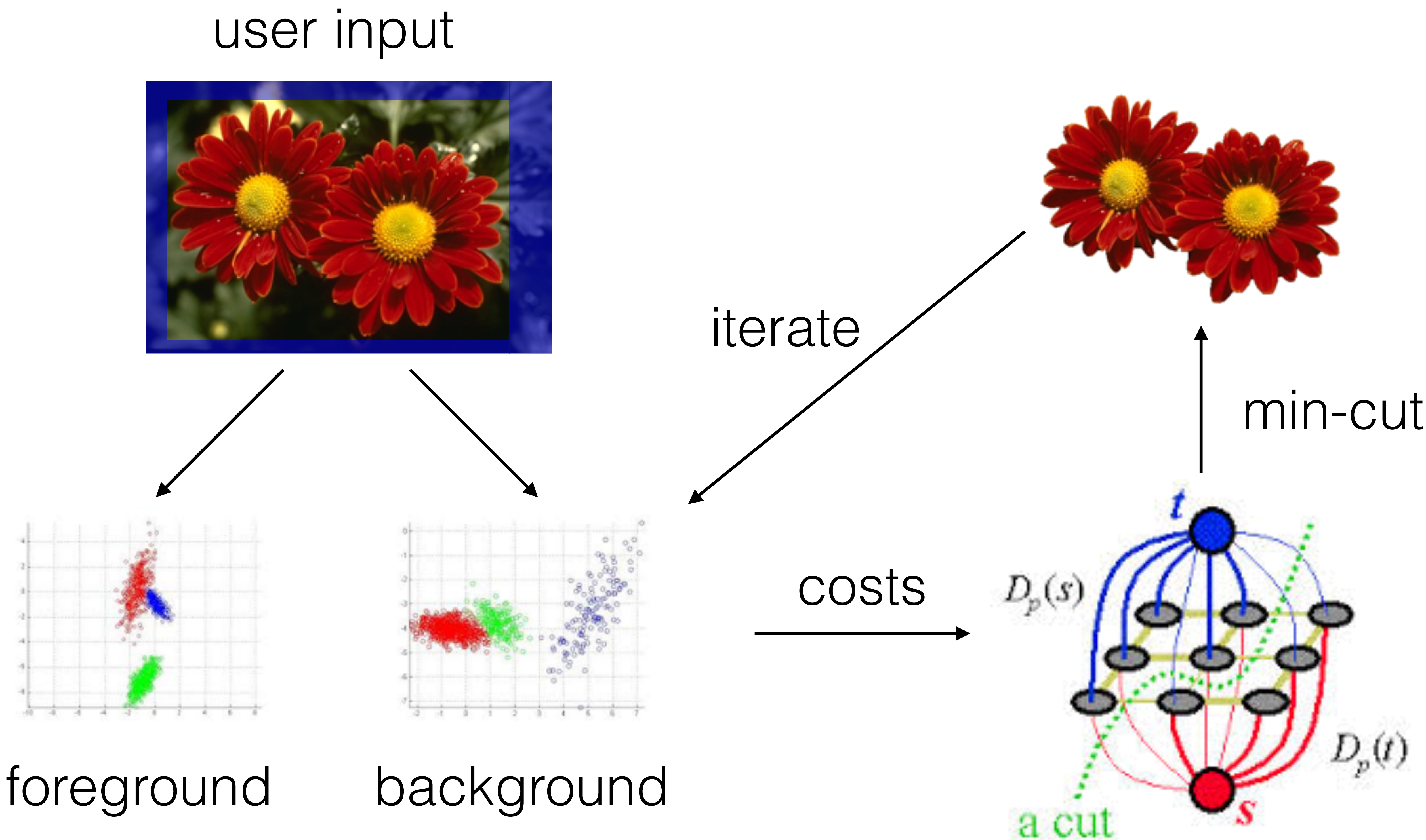


grabcut →

C. Rother, V. Kolmogorov, A. Blake. GrabCut: Interactive Foreground Extraction using Iterated Graph Cuts. *ACM Transactions on Graphics (SIGGRAPH'04), 2004*



Grabcut algorithm



Gaussian mixture model (5-8 components)
(probabilistic version of k-means)

Moderately straightforward examples

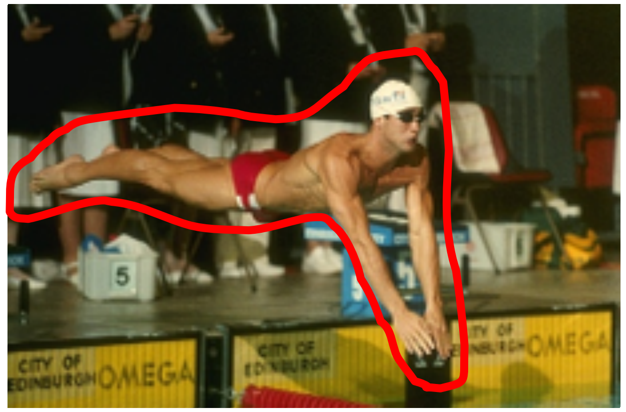
… GrabCut completes automatically

Difficult examples

Camouflage & Low Contrast

Fine structure

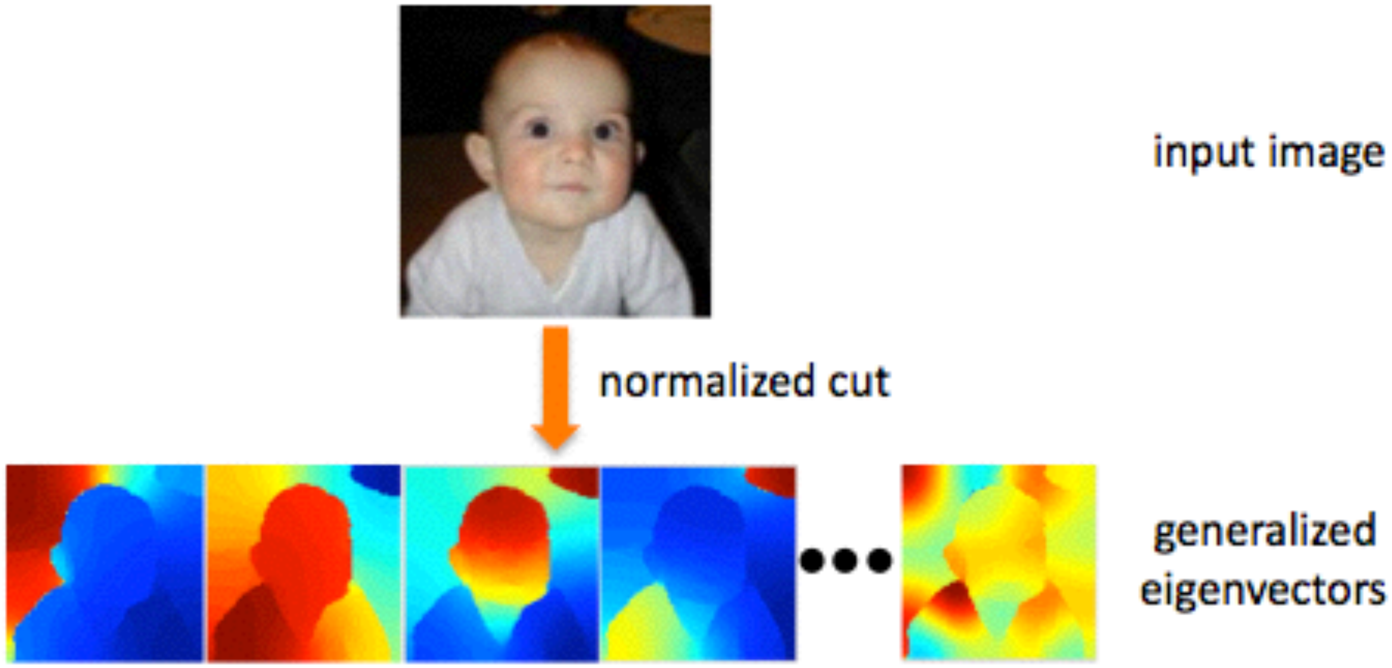
Harder Case



Grabcut algorithm

- Pros
  - Globally optimal solution using min-cut/max-flow algorithms
  - Fast algorithms exist for grid-graphs
- Cons
  - Need to recompute solution if the user input changes
  - Still slow for medium sized images (~10s for 480x640 image on my computer)

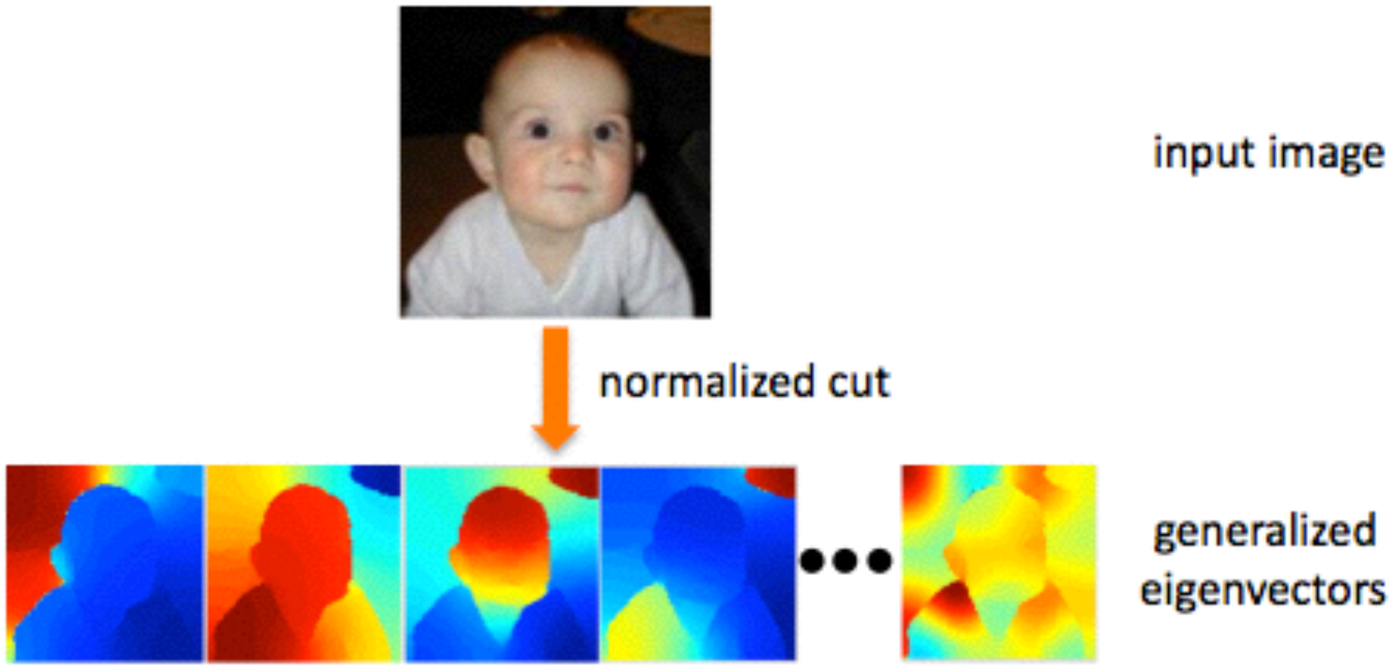
Biased normalized cuts



precompute (slow)

Subhransu Maji, Nisheeth Vishnoi and Jitendra Malik, Biased Normalized Cuts, CVPR 2011

Biased normalized cuts

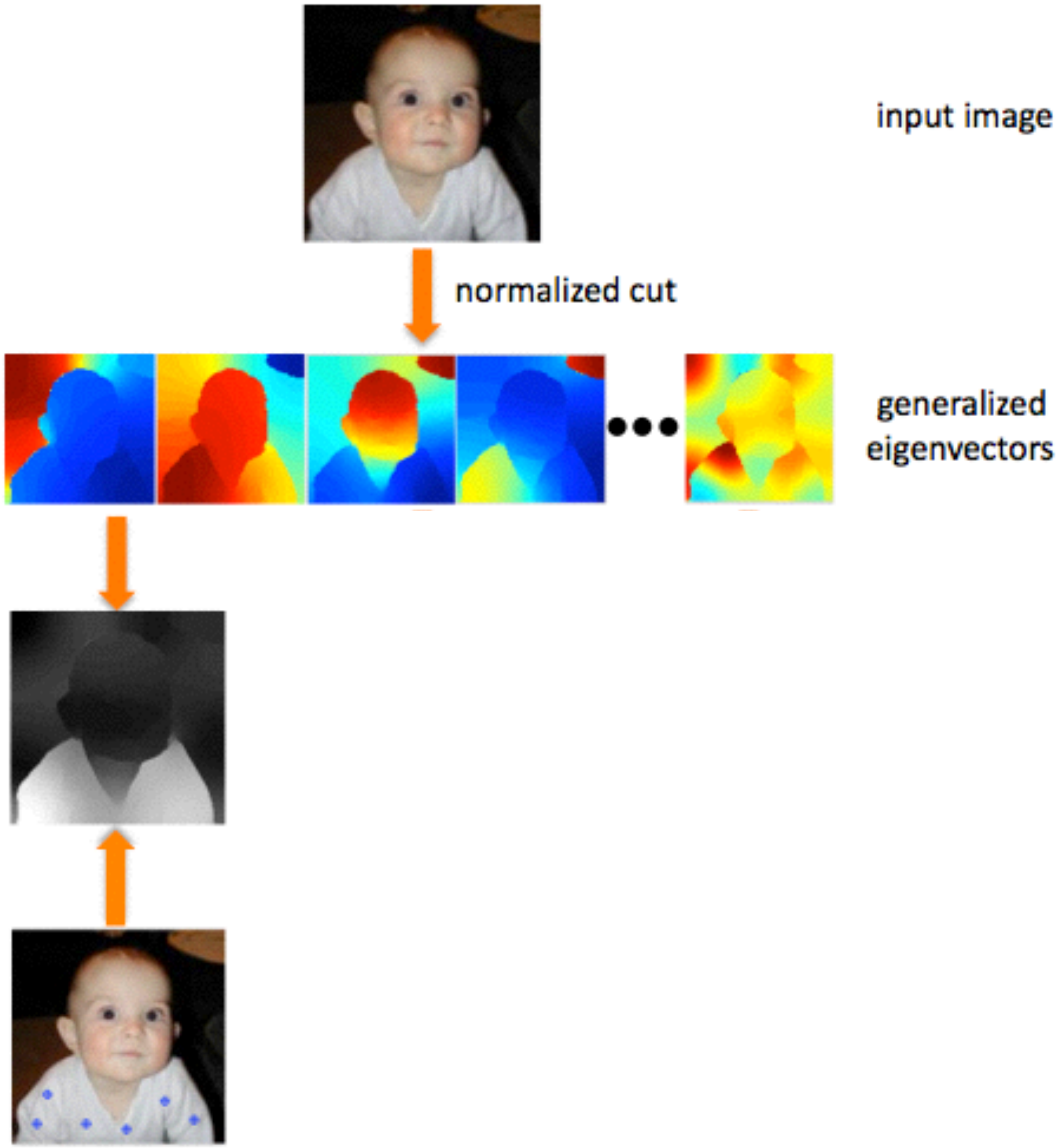


precompute (slow)

user input

Subhransu Maji, Nisheeth Vishnoi and Jitendra Malik, Biased Normalized Cuts, CVPR 2011

Biased normalized cuts



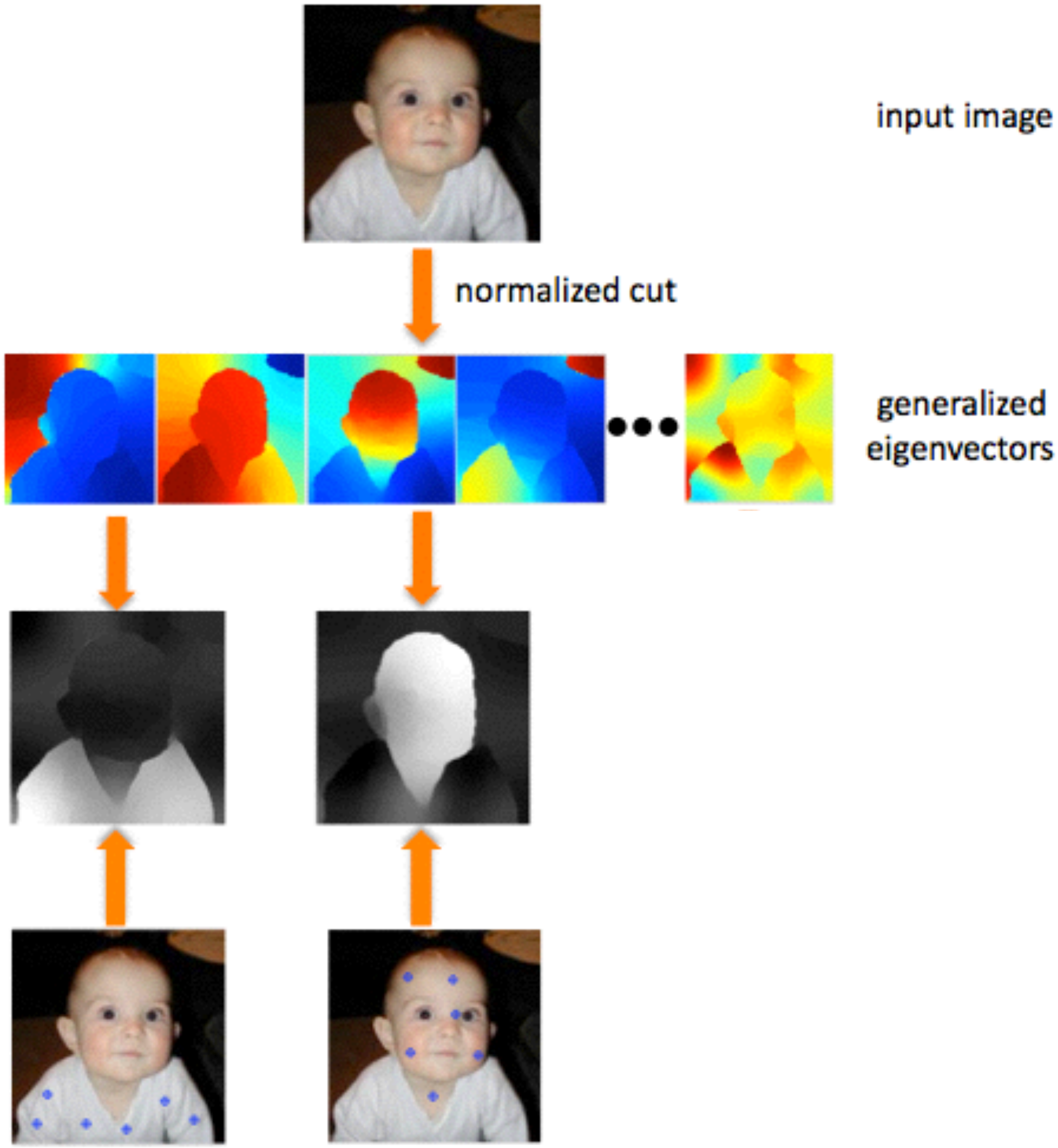
precompute (slow)

fast (10 ms per user input)

user input

Subhransu Maji, Nisheeth Vishnoi and Jitendra Malik, Biased Normalized Cuts, CVPR 2011

Biased normalized cuts



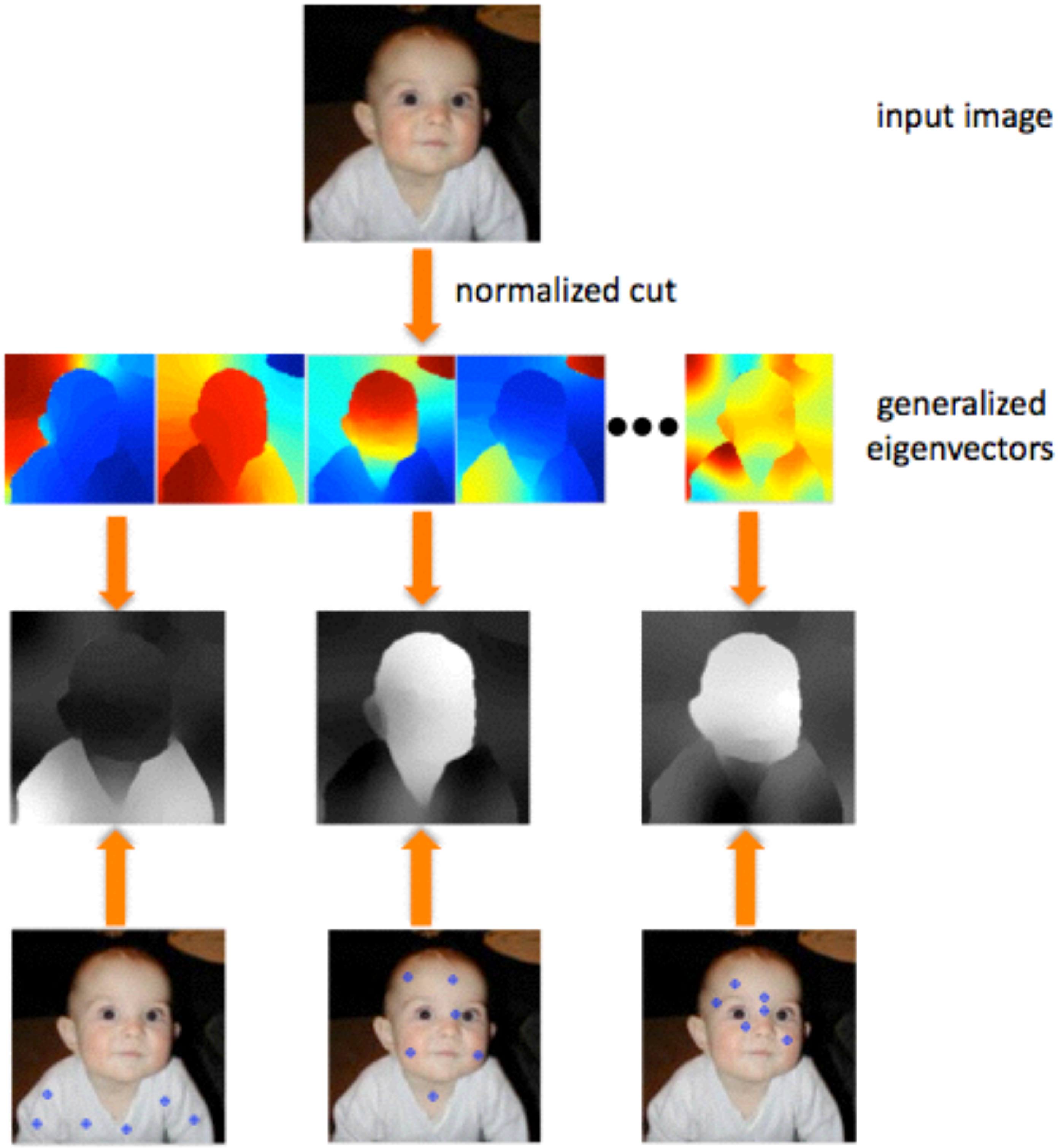
precompute (slow)

fast (10 ms per user input)

user input

Subhransu Maji, Nisheeth Vishnoi and Jitendra Malik, Biased Normalized Cuts, CVPR 2011

Biased normalized cuts



precompute (slow)

fast (10 ms per user input)

user input

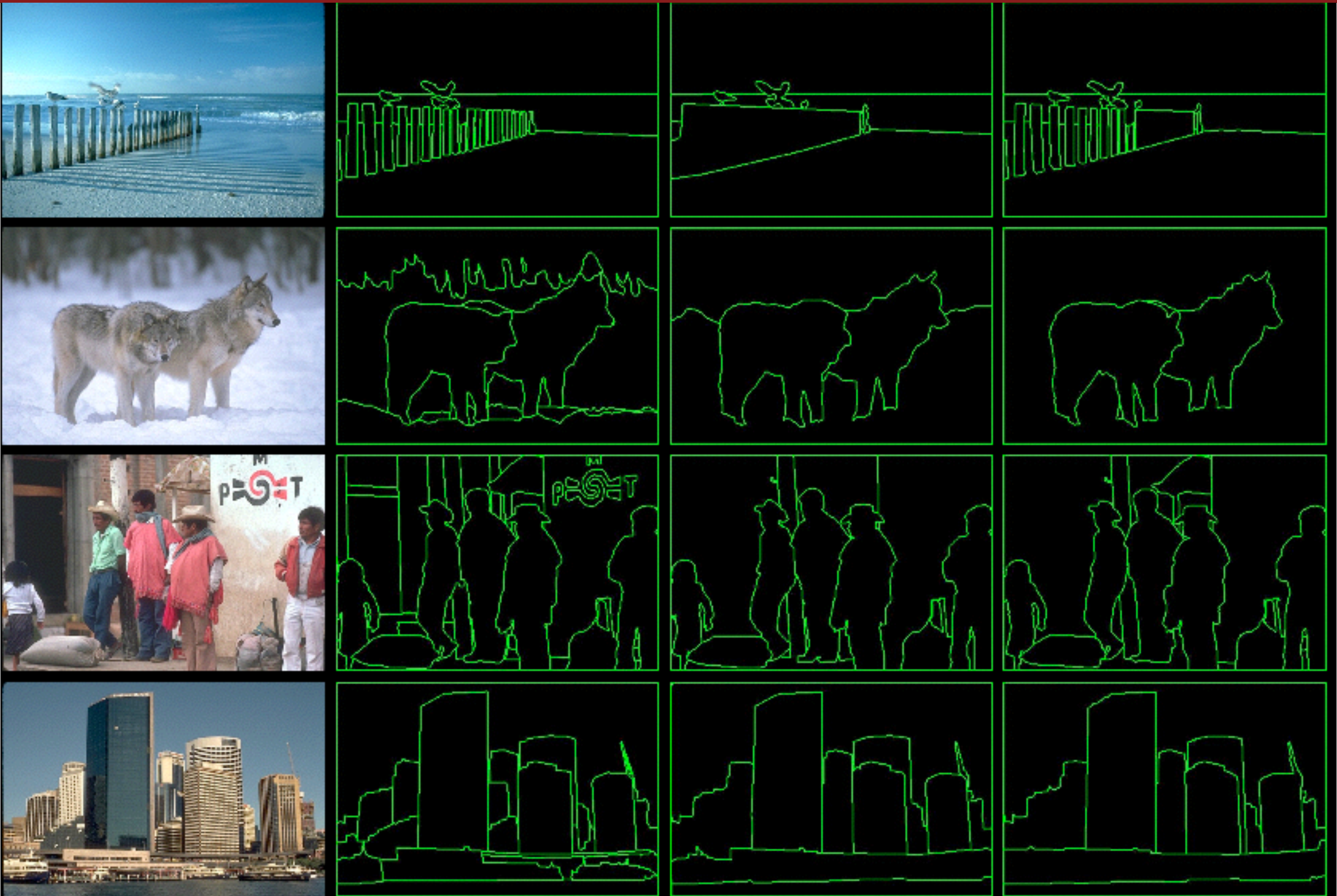
Subhransu Maji, Nisheeth Vishnoi and Jitendra Malik, Biased Normalized Cuts, CVPR 2011

Evaluating segmentations

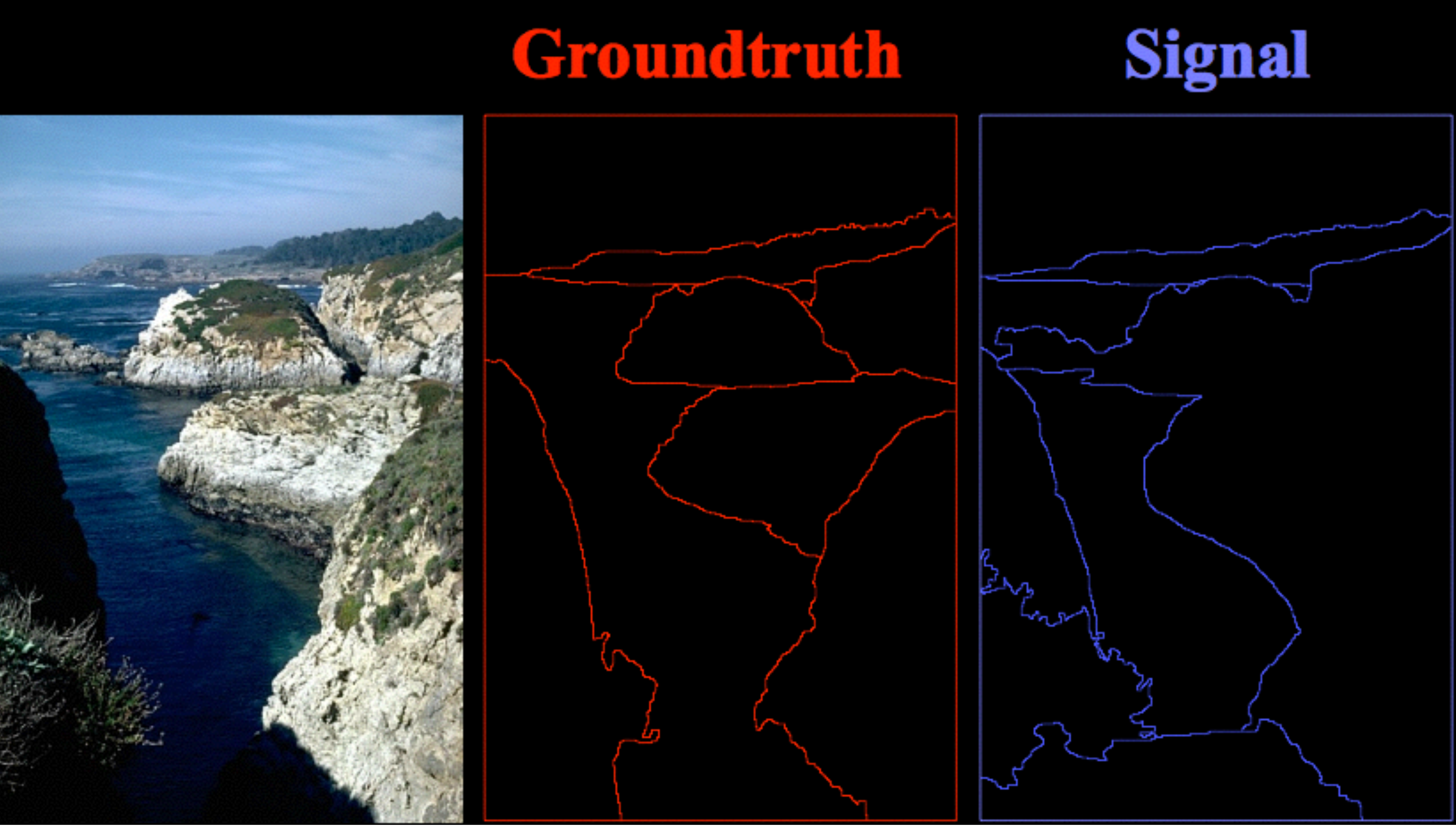
- **How do you know when a segmentation is good?**
- ~~The result should look good on these two images~~
- ~~Higher performance on the final goal we are interested in~~
  - grades, happiness, survival, …
- It did well on a standard segmentation benchmark

http://www.cis.upenn.edu/~jshi/GraphTutorial/Tutorial-ImageSegmentationGraph-cut2-Martin.pdf

Berkeley segmentation database



Measuring accuracy



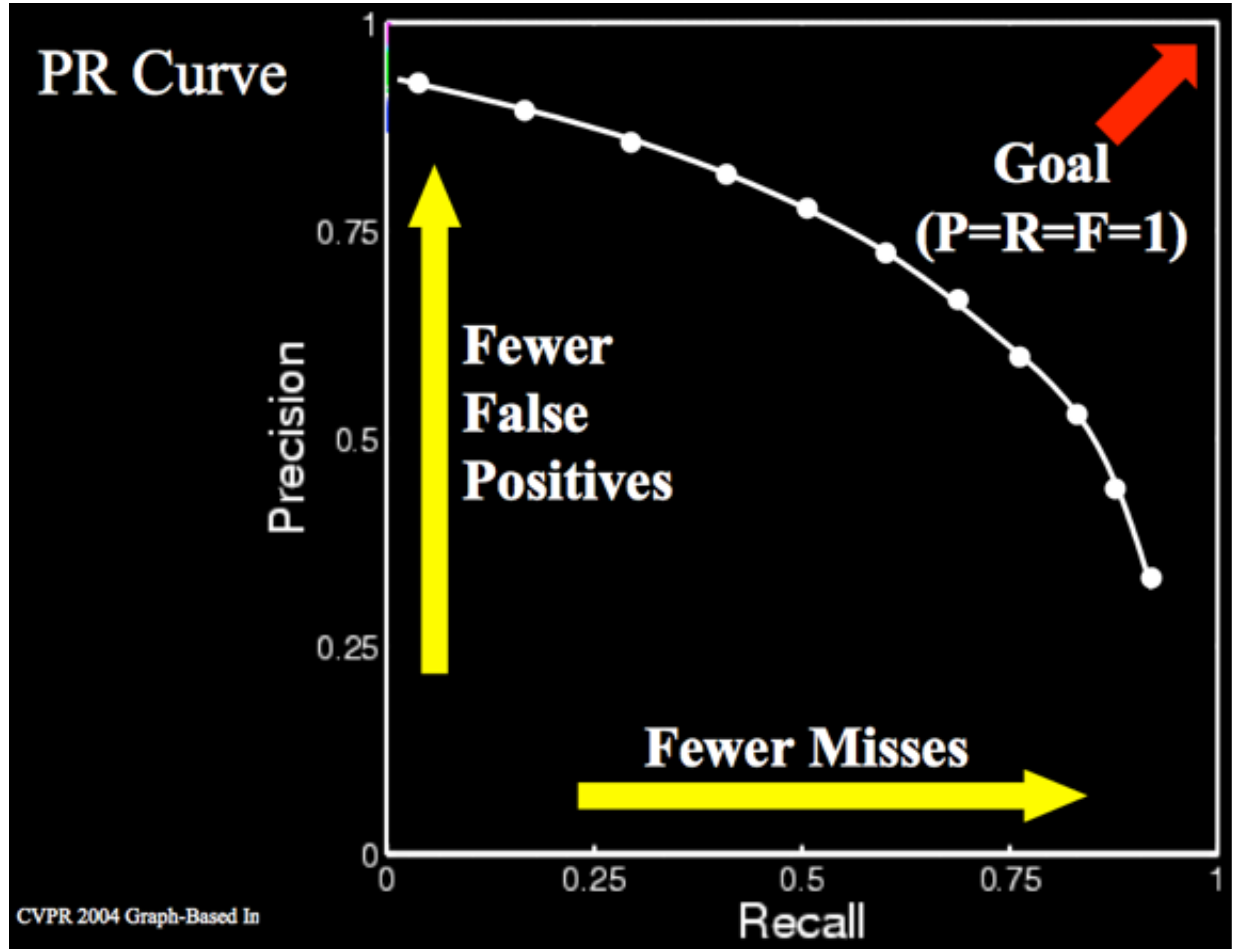
Signal at a pixel is correct if there is a unaccounted ground truth boundary pixel within a distance threshold

Precision and Recall

		Truth	
		P	N
Signal	P	TP	FP
	N	FN	TN

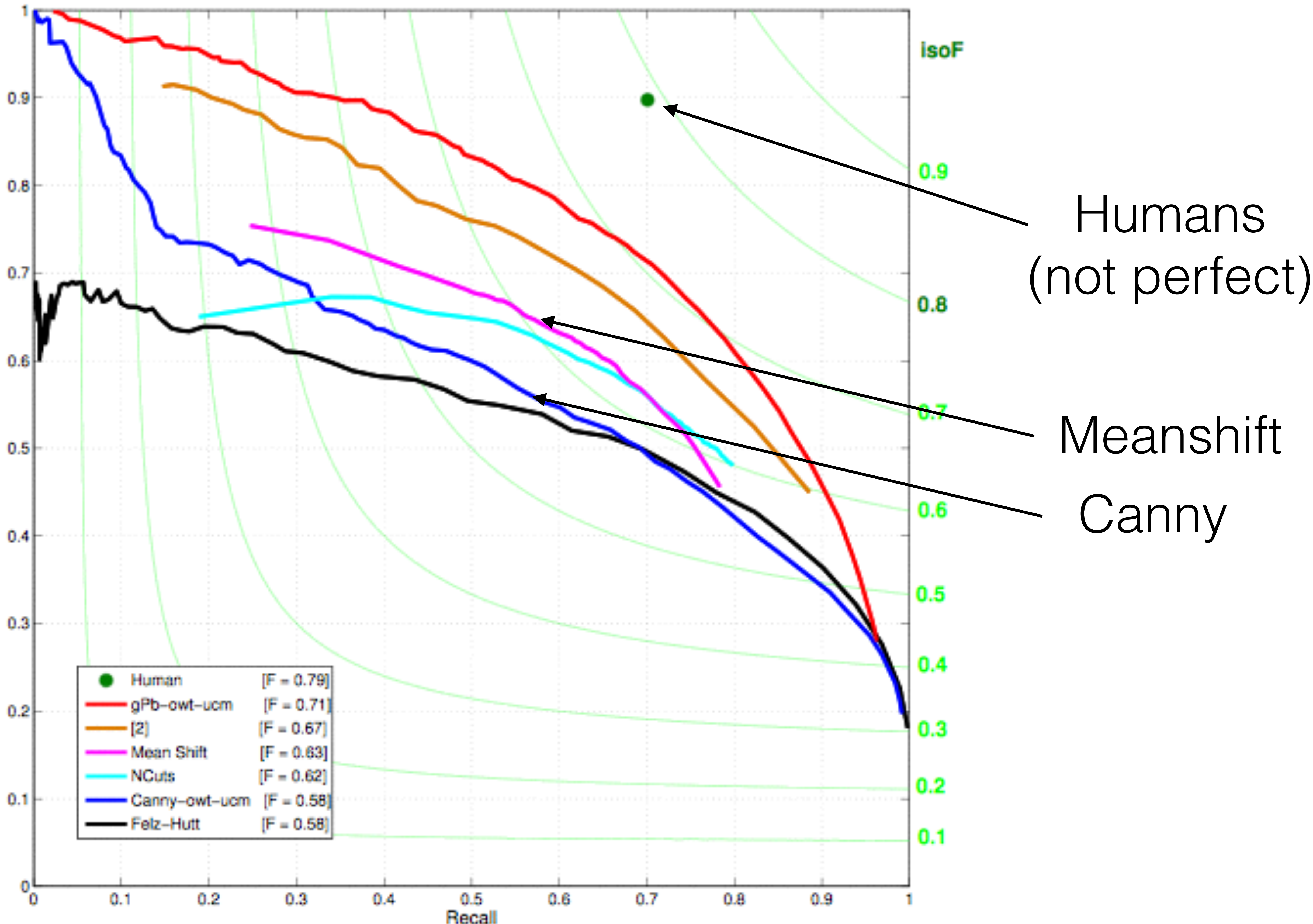
Precision = TP / (TP+FP)

Recall = TP / (TP+FN)



- Vary the threshold and plot precision vs. recall curve

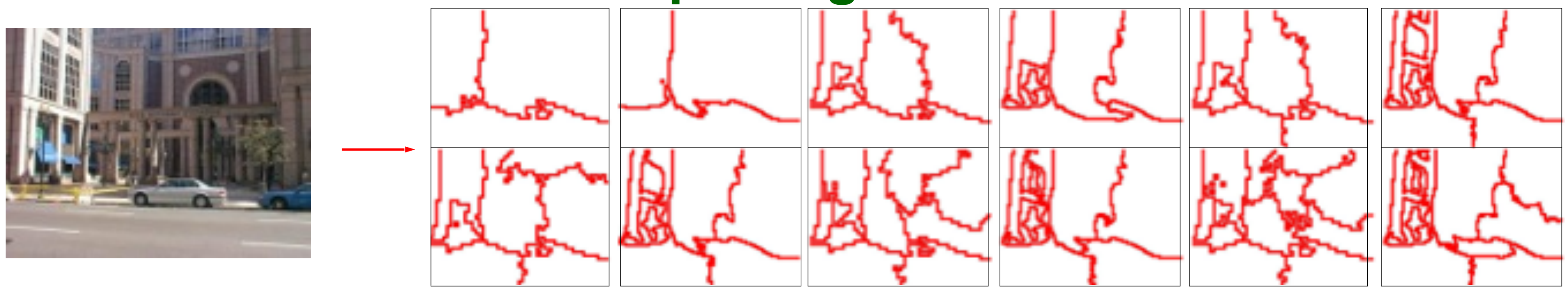
Current methods on BSDS



Arbelaez et al., CVPR 2009

Segments as primitives for recognition

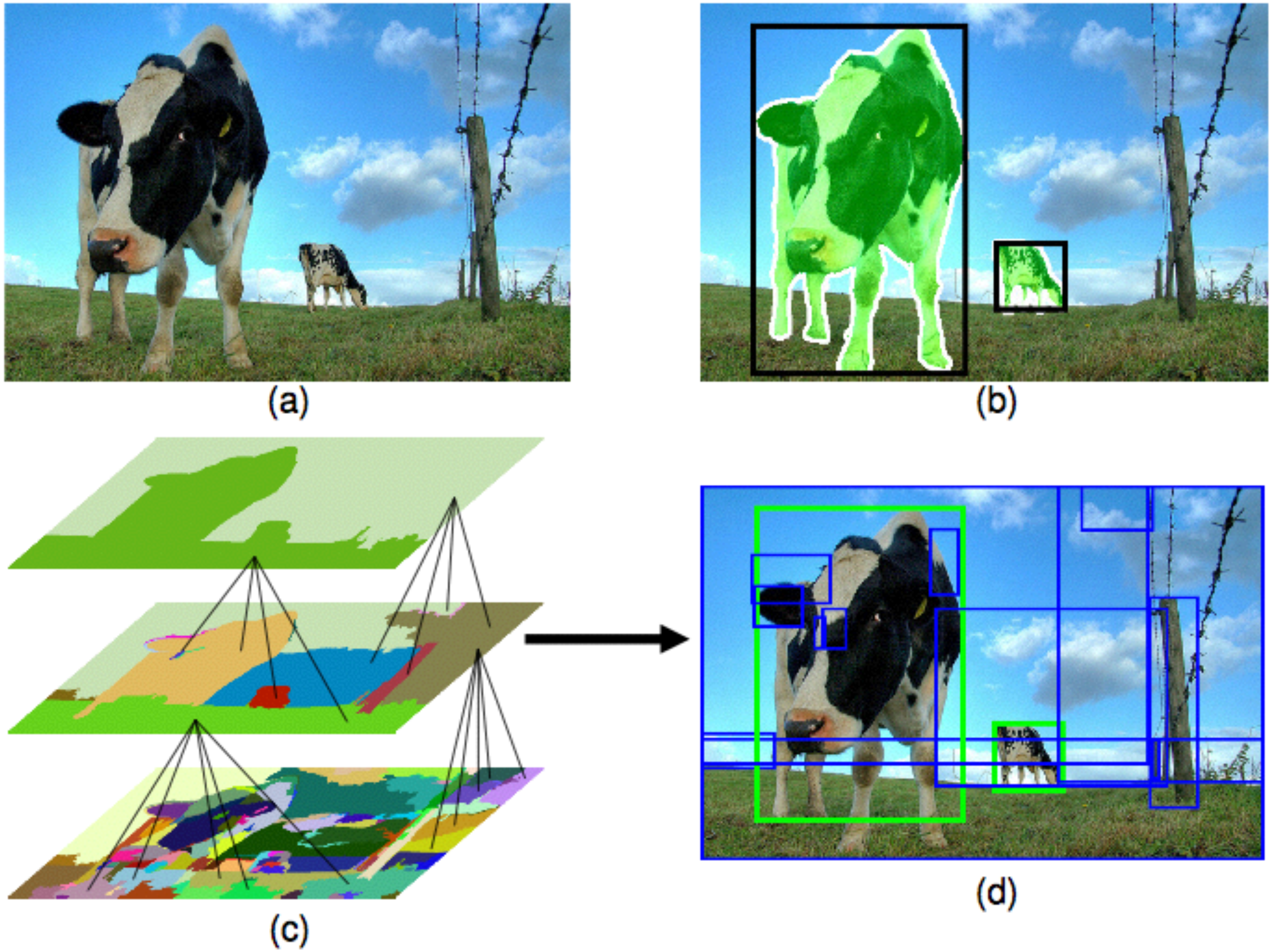
Multiple segmentations



B. Russell et al., **"Using Multiple Segmentations to Discover Objects and their Extent in Image Collections,"** CVPR 2006

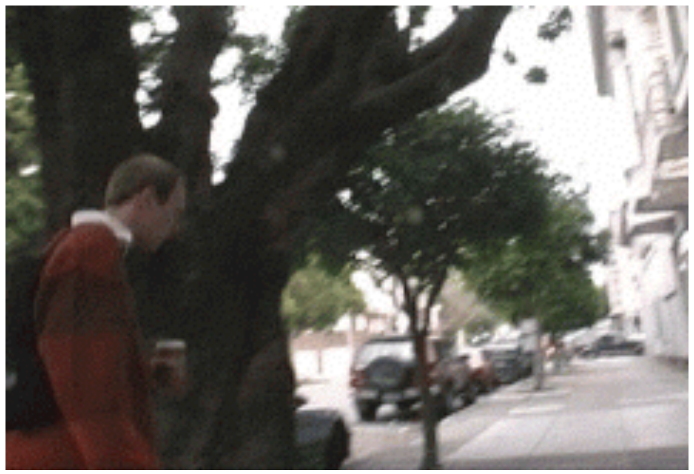
Regions proposals for detection

- Generate a set of regions for further classification



Van de Sande et al., ICCV 2011

Motion segmentation



Input sequence

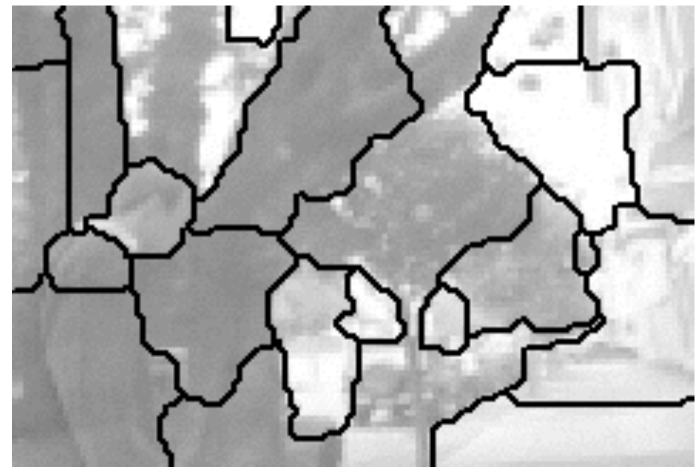
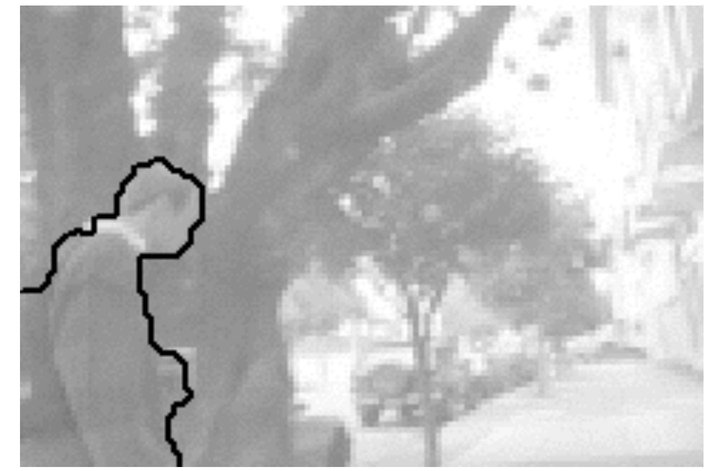
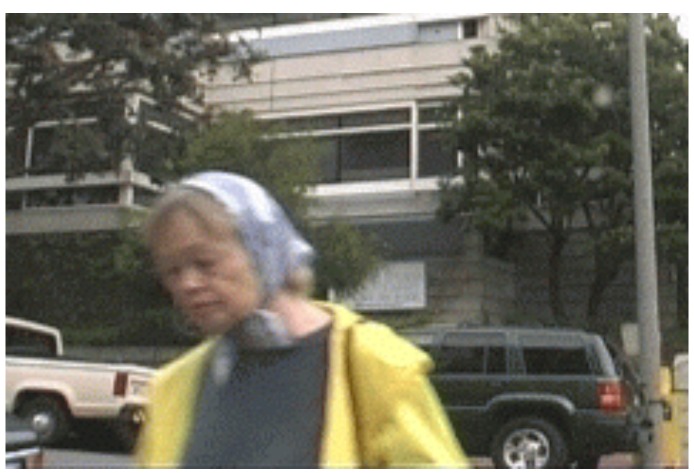


Image Segmentation



Motion Segmentation



Input sequence

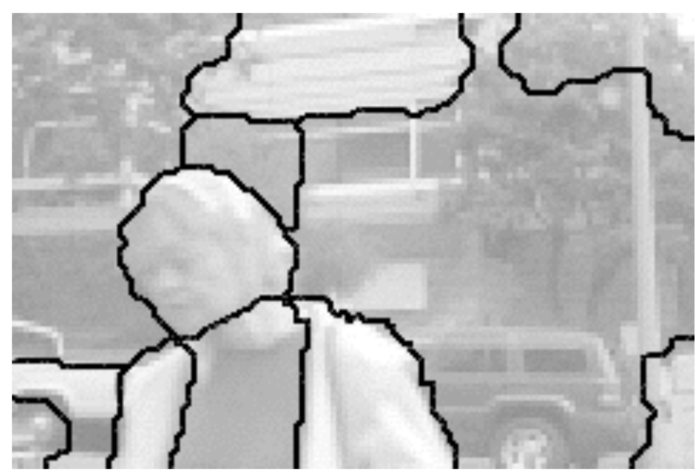


Image Segmentation

Motion Segmentation

A.Barbu, S.C. Zhu. Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities, *IEEE Trans. PAMI,* August 2005.

Summary

- Segmentation to find object boundaries or mid-level regions, tokens.
- Bottom-up segmentation via clustering
  - General choices -- features, affinity functions, and clustering algorithms
- Grouping methods also useful for quantization, can create new feature summaries
  - Texton histograms for texture within local region
- Example clustering methods
  - K-means
  - Mean shift
  - Graph cut, normalized cut
- Segmentation quality can be measured (BSDS)

Further thoughts and readings ..

- Gestalt psychology **http://en.wikipedia.org/wiki/Gestalt_psychology**
- Chapter 5, Richard Szeliski's book
- **Berkeley segmentation database and benchmark**
  - Also read about the Berkeley boundary detector
- **http://www.cis.upenn.edu/~jshi/GraphTutorial/**
- Image segmentation via. graph cuts
  - Boykov and Jolly, **Interactive graph cuts for optimal boundary & region segmentation of objects in ND images**, ICCV 2001
- Normalized cuts for image segmentation (Shi and Malik)
  - **http://www.cs.berkeley.edu/~malik/papers/SM-ncut.pdf**
- Biased normalized cuts
  - **http://people.cs.umass.edu/~smaji/projects/biasedNcuts/index.html**