Lecture 6: Clustering and Segmentation – Part 2

Professor Fei-Fei Li

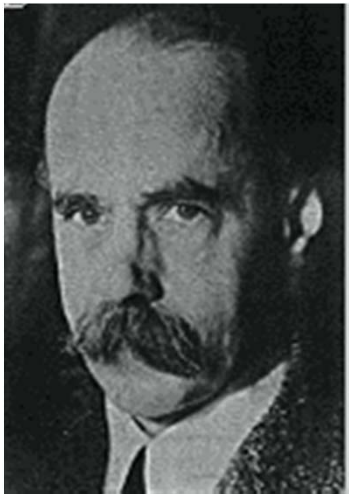
Stanford Vision Lab

Recap: Gestalt Theory

- Gestalt: whole or group
  - Whole is greater than sum of its parts
  - Relationships among parts can yield new properties/features
- Psychologists identified series of factors that predispose set of elements to be grouped (by human visual system)

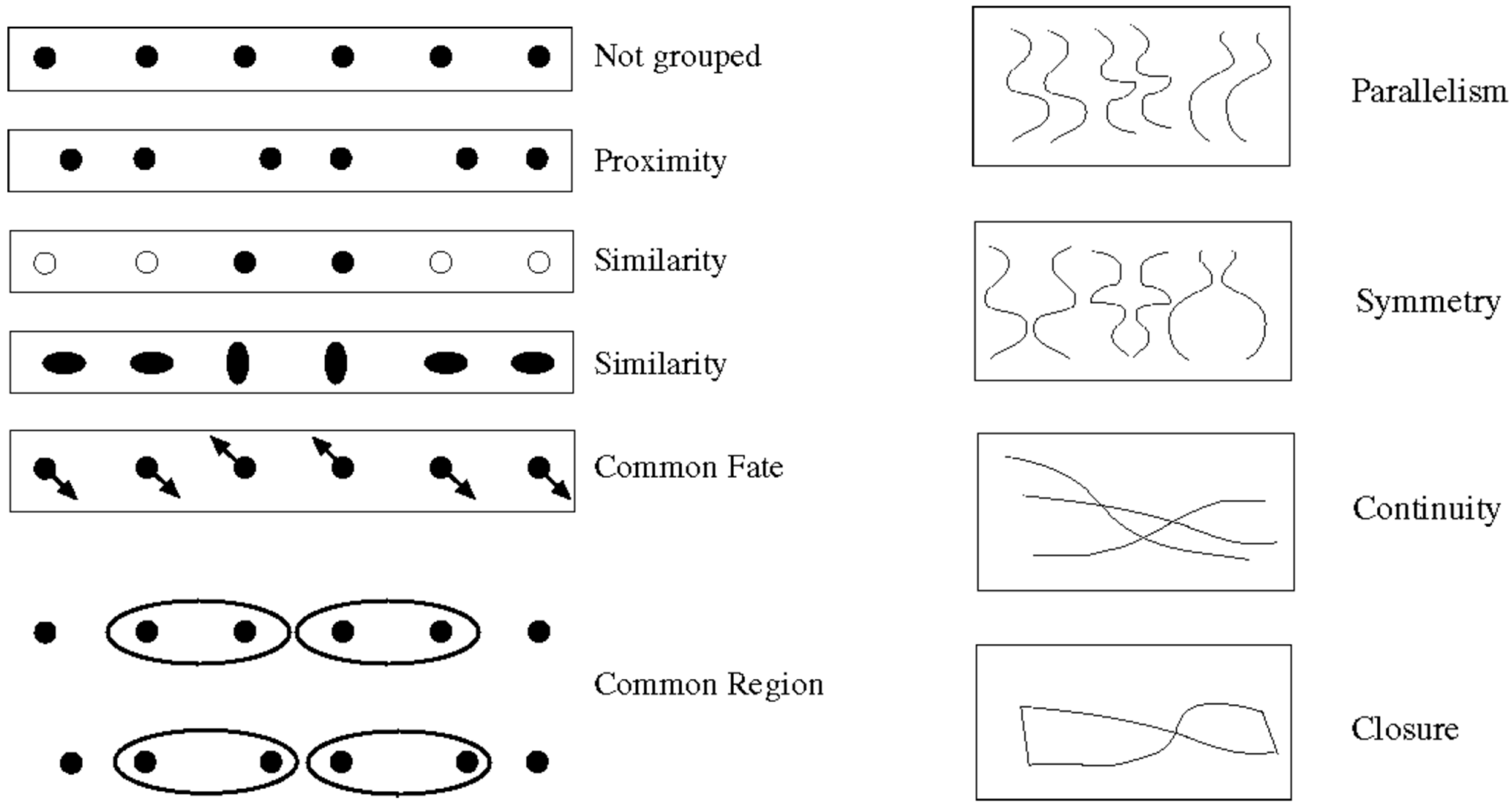
"I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have "327"? No. I have sky, house, and trees."

Max Wertheimer
(1880-1943)



Untersuchungen zur Lehre von der Gestalt,
Psychologische Forschung, **Vol. 4, pp. 301-350, 1923**
http://psy.ed.asu.edu/~classics/Wertheimer/Forms/forms.htm

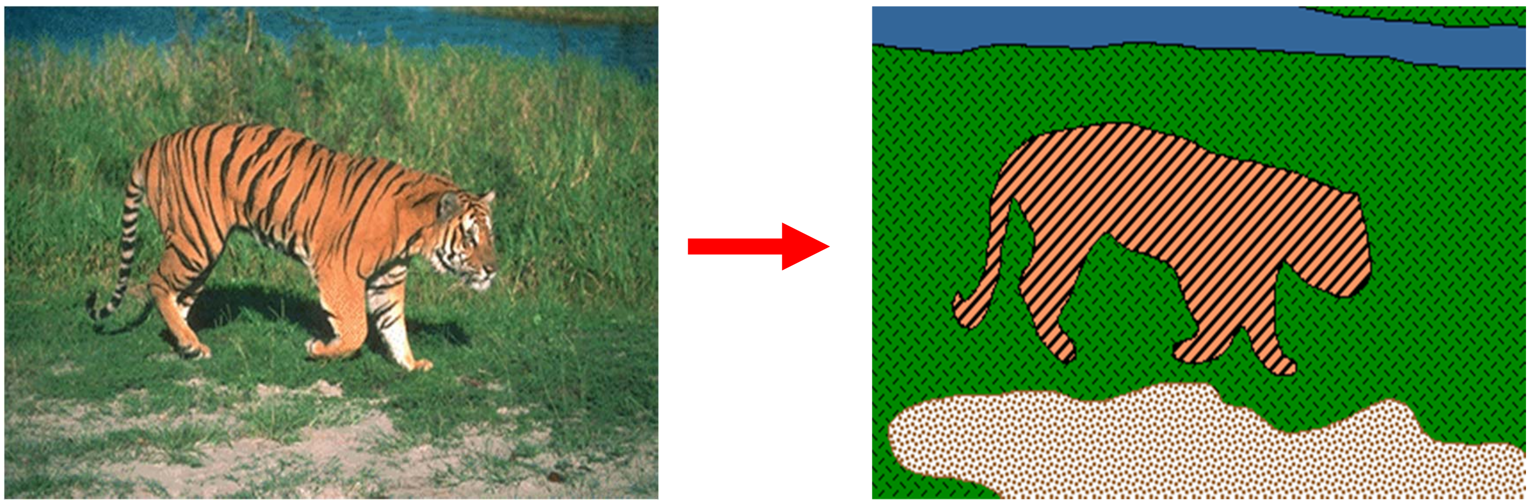
Recap: Gestalt Factors



- These factors make intuitive sense, but are very difficult to translate into algorithms.

Recap: Image Segmentation

- Goal: identify groups of pixels that go together



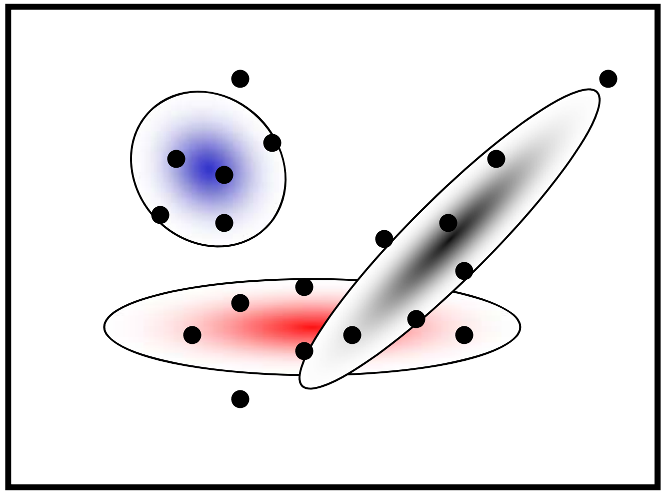
Recap: K-Means Clustering

- Basic idea: randomly initialize the *k* cluster centers, and iterate between the two steps we just saw.
  1. Randomly initialize the cluster centers, $c_1, ..., c_K$
  2. Given cluster centers, determine points in each cluster
     - For each point p, find the closest c_i. Put p into cluster i
  3. Given points in each cluster, solve for c_i
     - Set c_i to be the mean of points in cluster i
  4. If c_i have changed, repeat Step 2

- Properties
  - Will always converge to *some* solution
  - Can be a "local minimum"
    - Does not always find the global minimum of objective function:

$$\sum_{\text{clusters } i} \sum_{\text{points p in cluster } i} ||p - c_i||^2$$

Recap: Expectation Maximization (EM)



- Goal
  - Find blob parameters θ that maximize the likelihood function:

$$P(data|\theta) = \prod_x P(x|\theta)$$

- Approach:
  1. E-step: given current guess of blobs, compute ownership of each point
  2. M-step: given ownership probabilities, update blobs to maximize likelihood function
  3. Repeat until convergence

What we will learn today?

- Model free clustering
  - Mean-shift
- Graph theoretic segmentation
  - Normalized Cuts
  - Using texture features

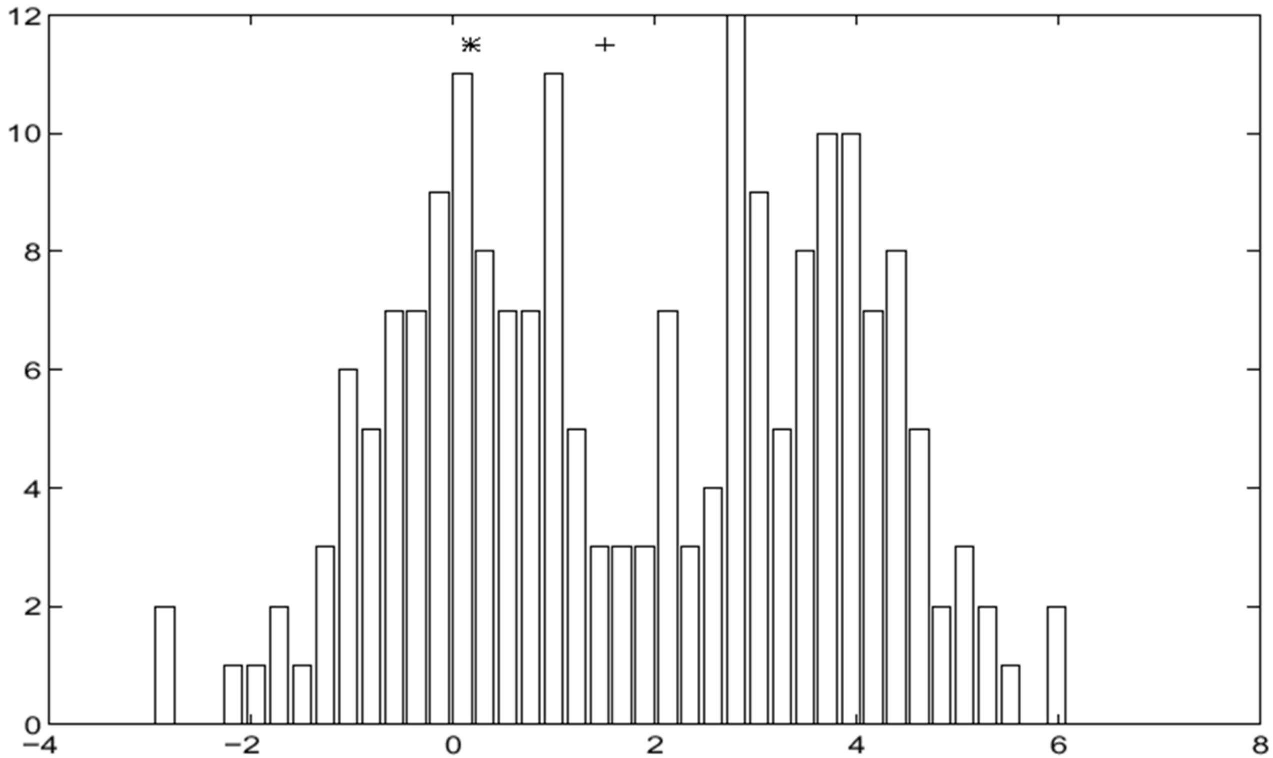
(Midterm materials)

- Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation (supp. materials)
  - s-t mincut algorithm (supp. materials)
  - Extension to non-binary case (supp. materials)
  - Applications

What we will learn today?

- "Model free" clustering
  - Mean-shift
- Graph theoretic segmentation
  - Normalized Cuts
  - Using texture features
- Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation (supp. materials)
  - s-t mincut algorithm (supp. materials)
  - Extension to non-binary case (supp. materials)
  - Applications

Finding Modes in a Histogram

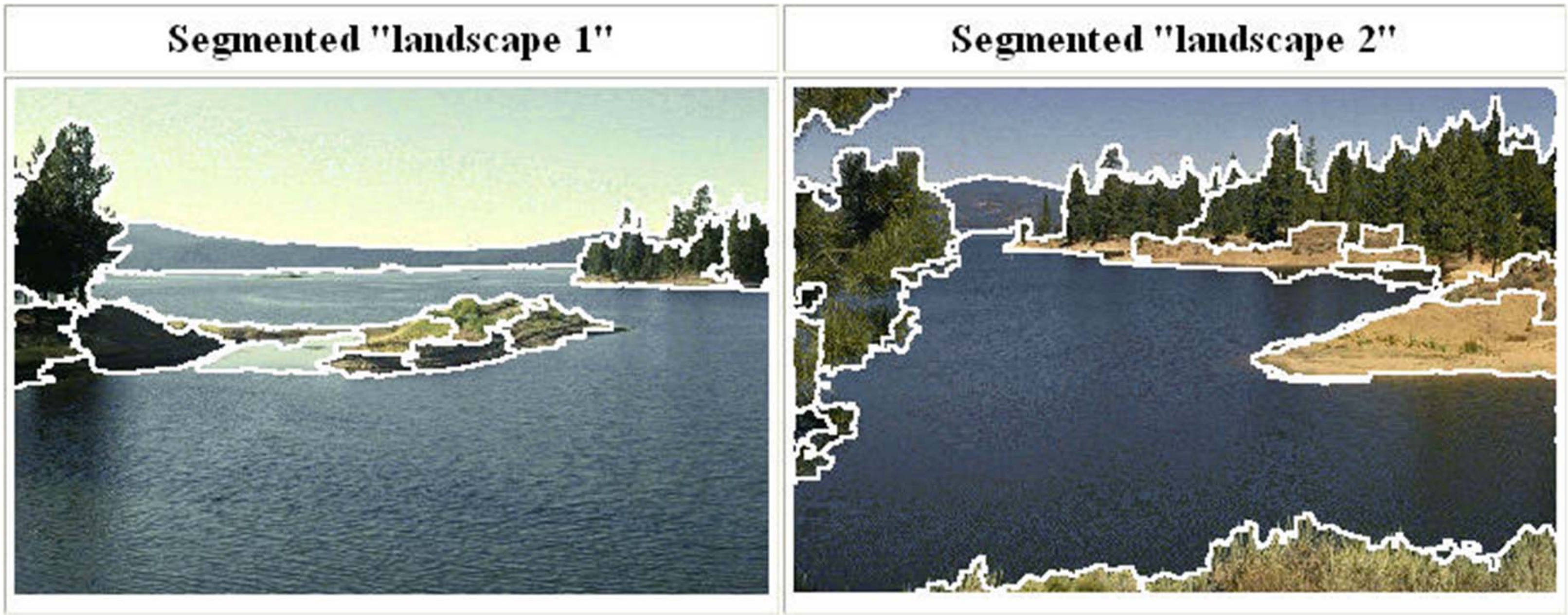


- How many modes are there?
  - *Mode* = local maximum of the density of a given distribution
  - Easy to see, hard to compute

Slide credit: Steve Seitz

Mean-Shift Segmentation

- An advanced and versatile technique for clustering-based segmentation

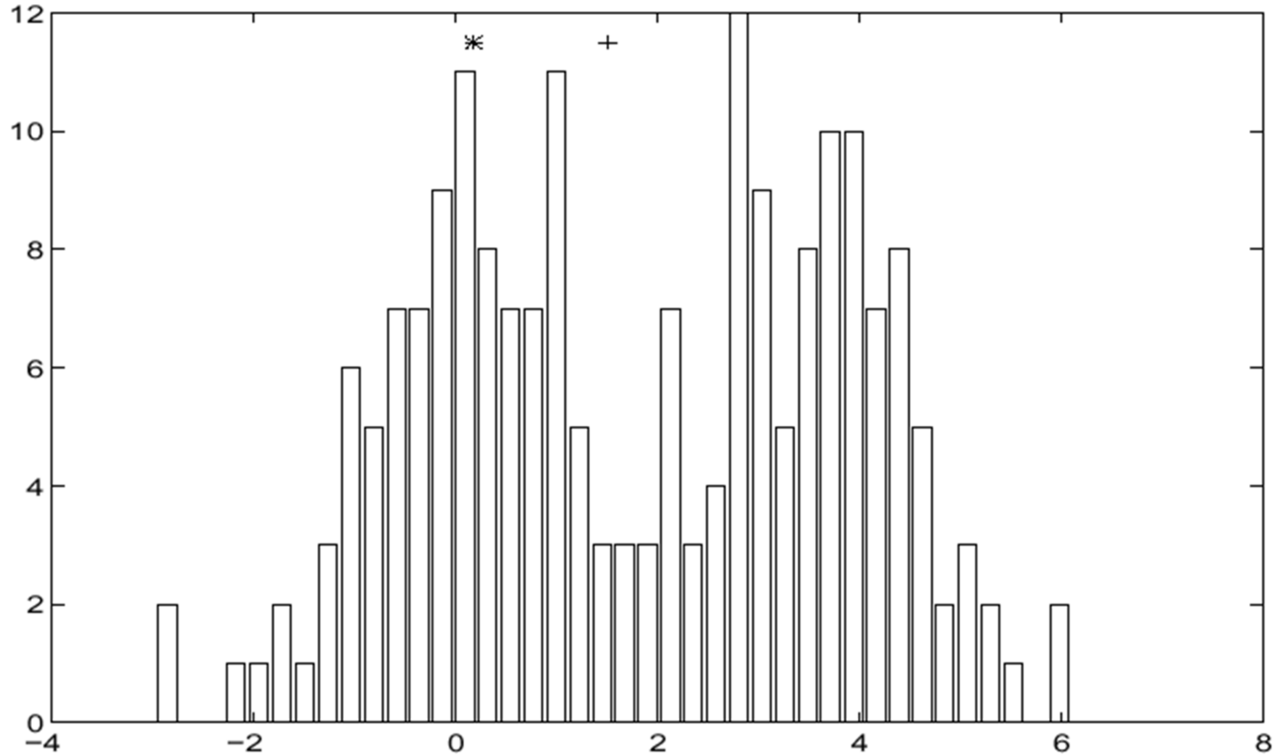


http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html

D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002.

Slide credit: Svetlana Lazebnik

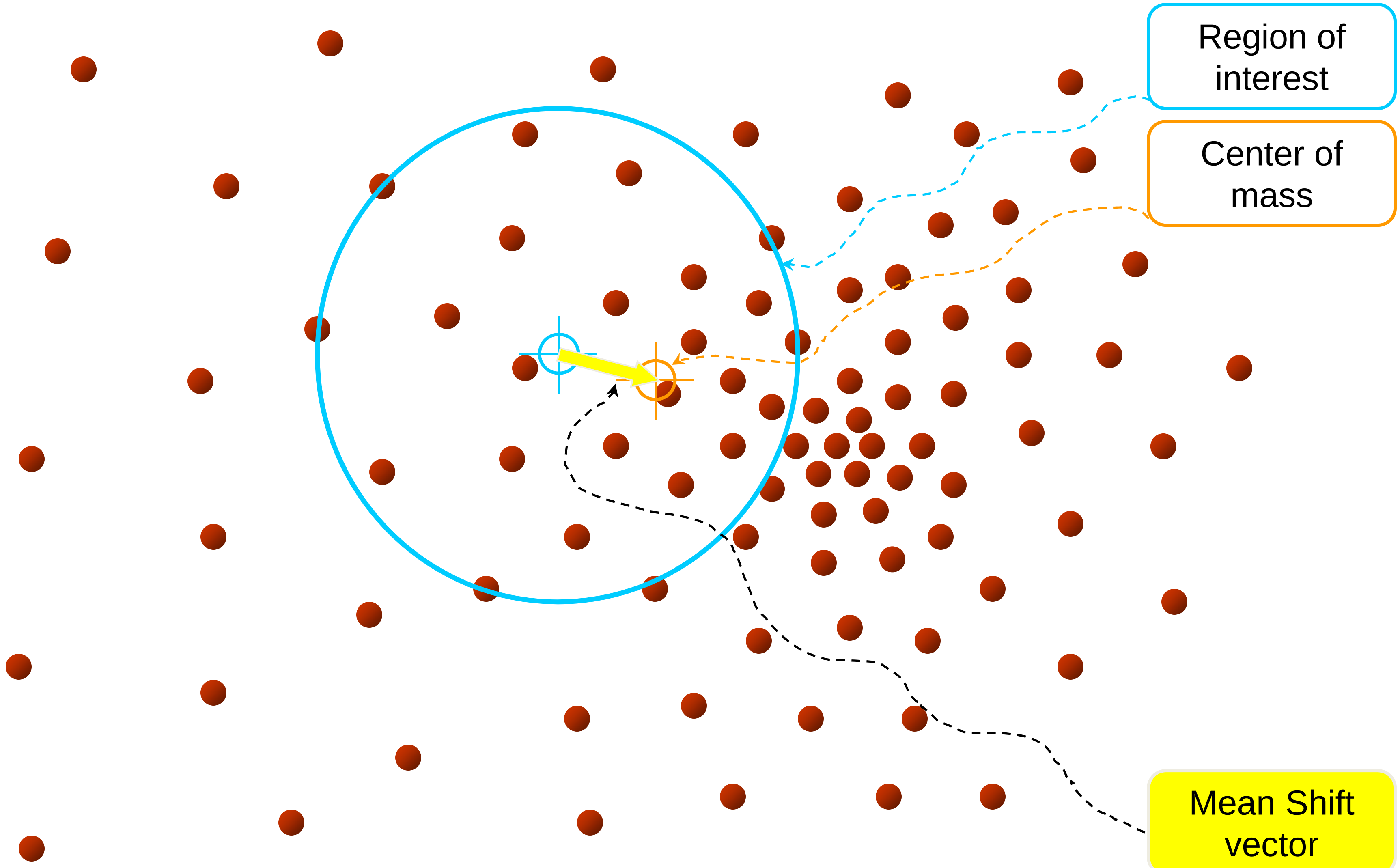
Mean-Shift Algorithm



- Iterative Mode Search
  1. Initialize random seed, and window W
  2. Calculate center of gravity (the "mean") of W: $\sum_{x \in W} x H(x)$
  3. Shift the search window to the mean
  4. Repeat Step 2 until convergence

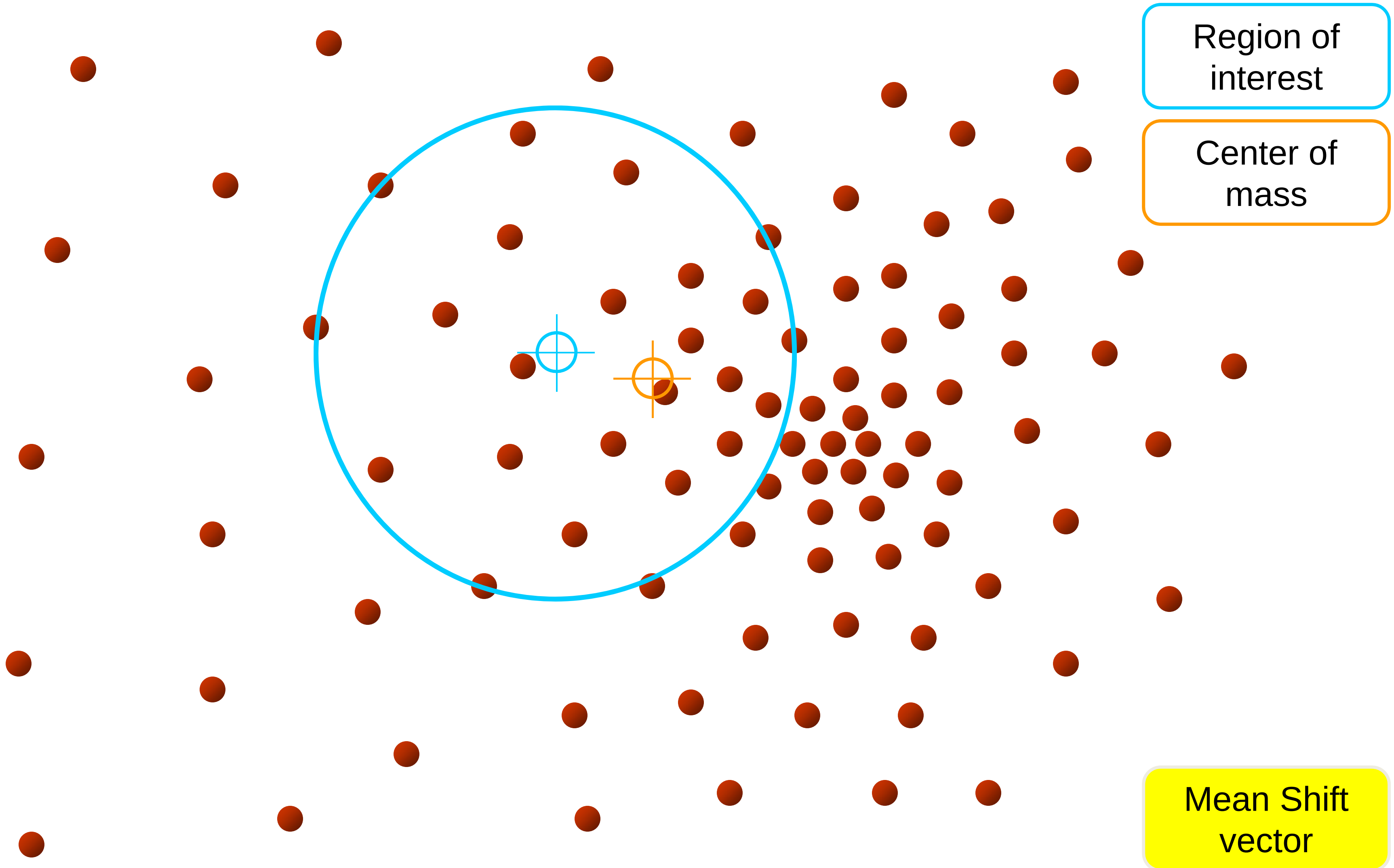
Slide credit: Steve Seitz

Mean-Shift



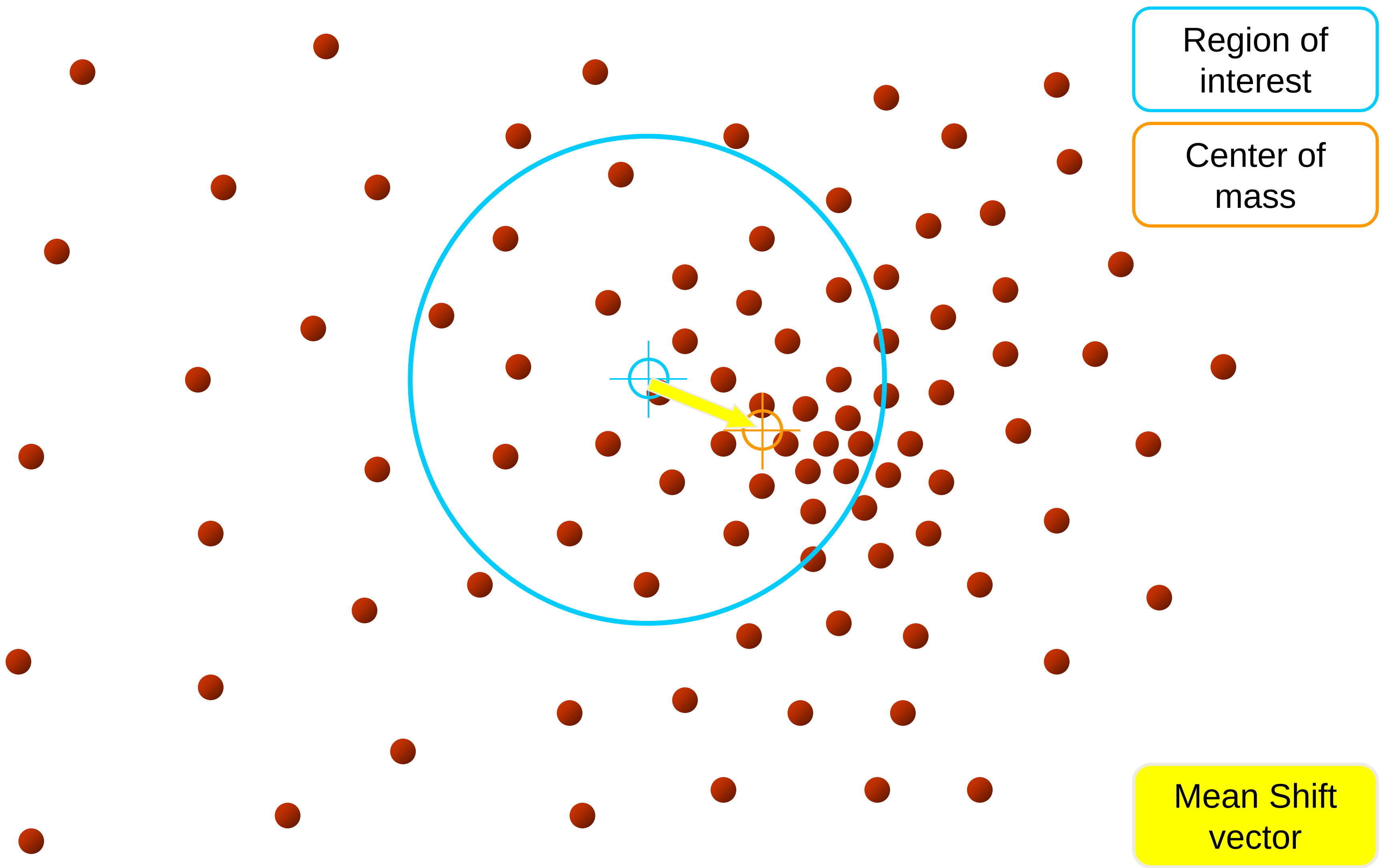
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



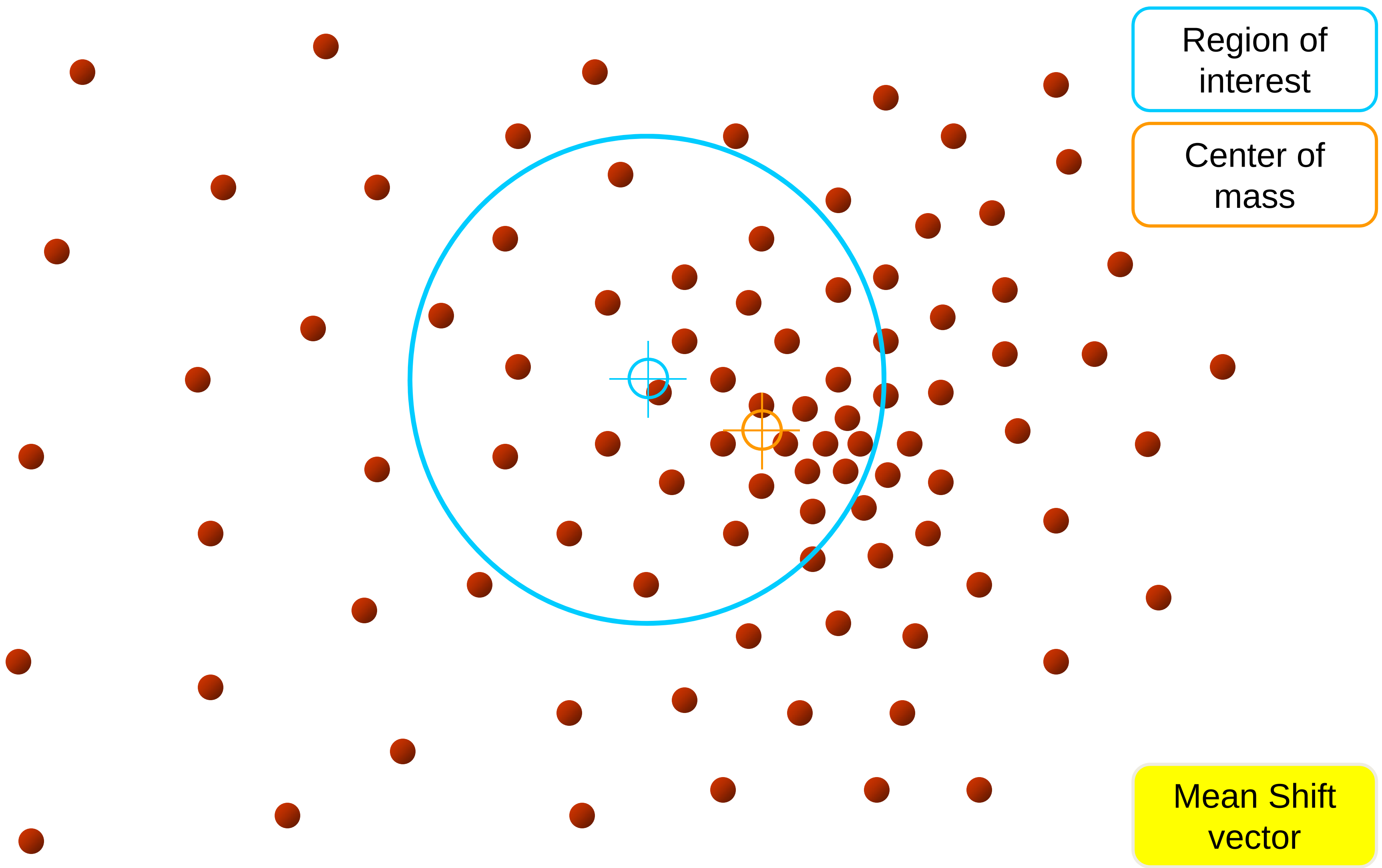
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



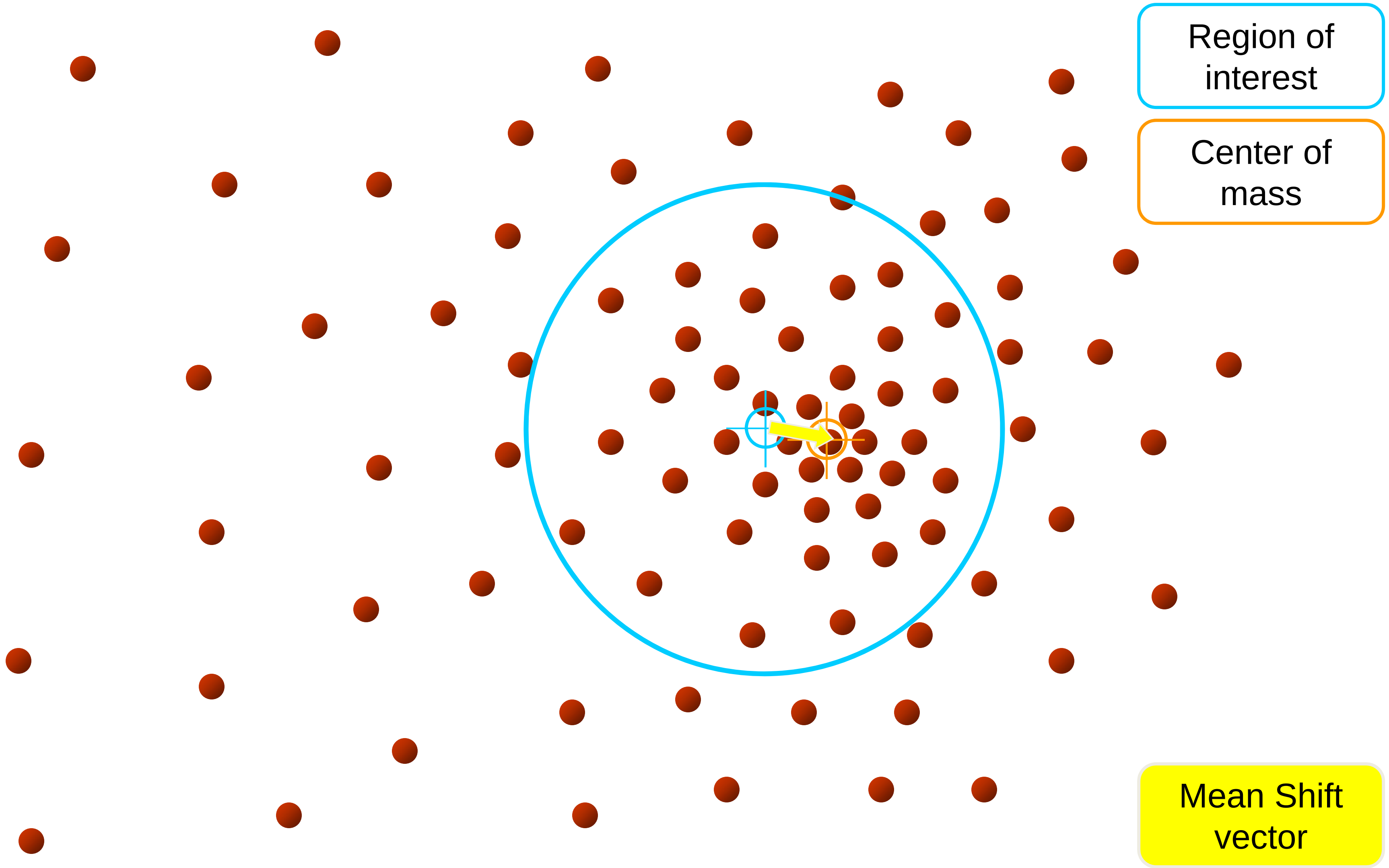
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



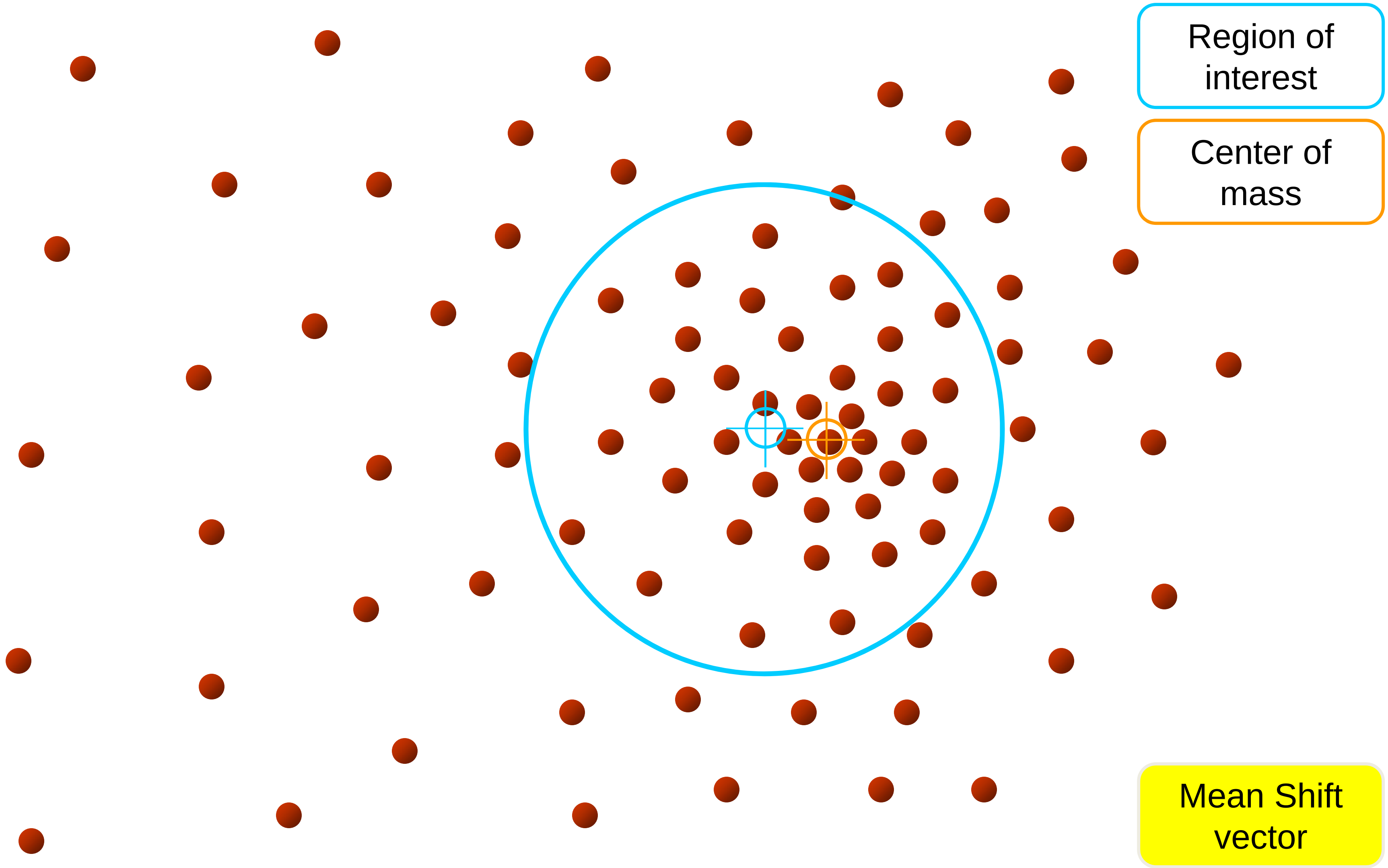
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



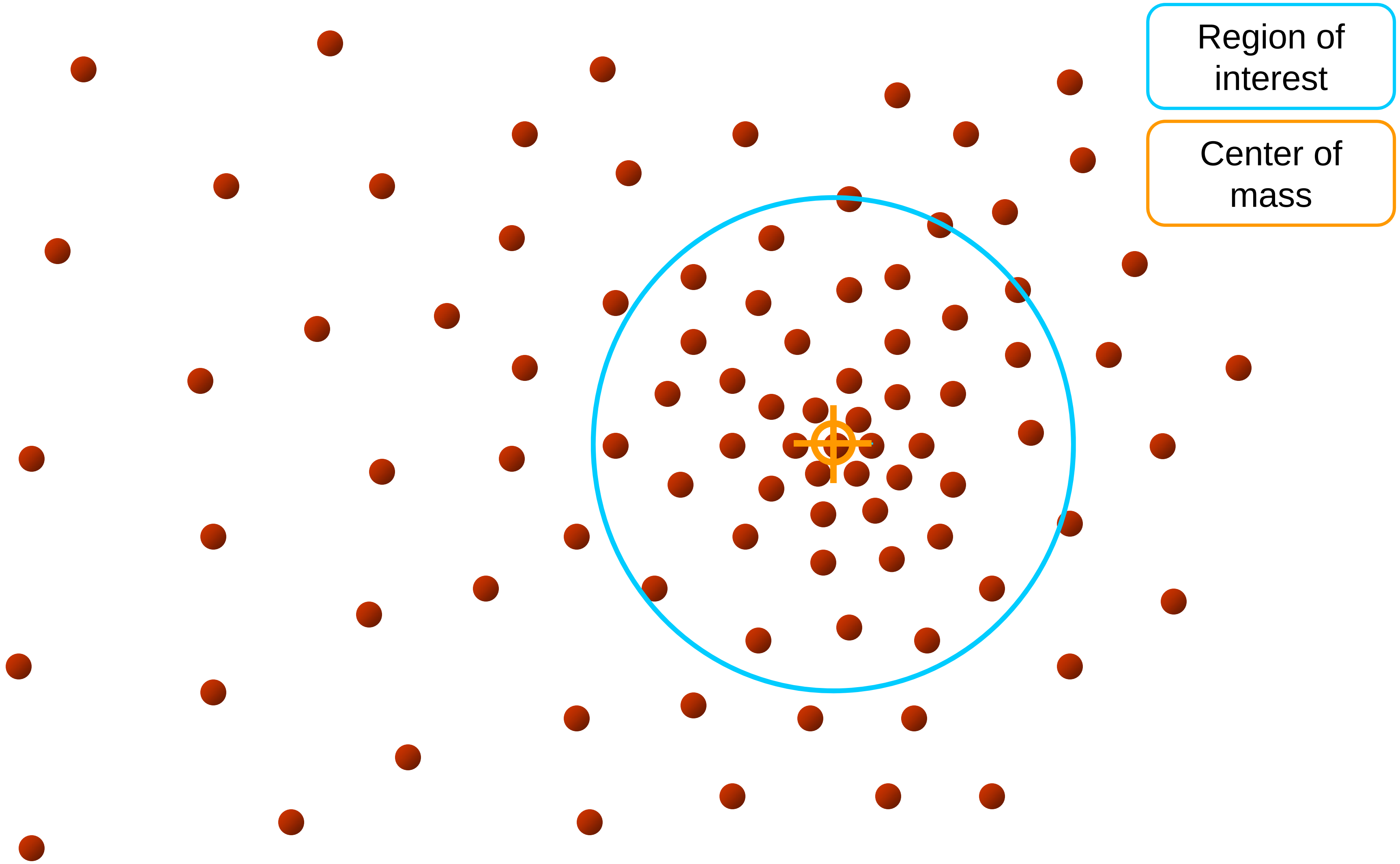
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



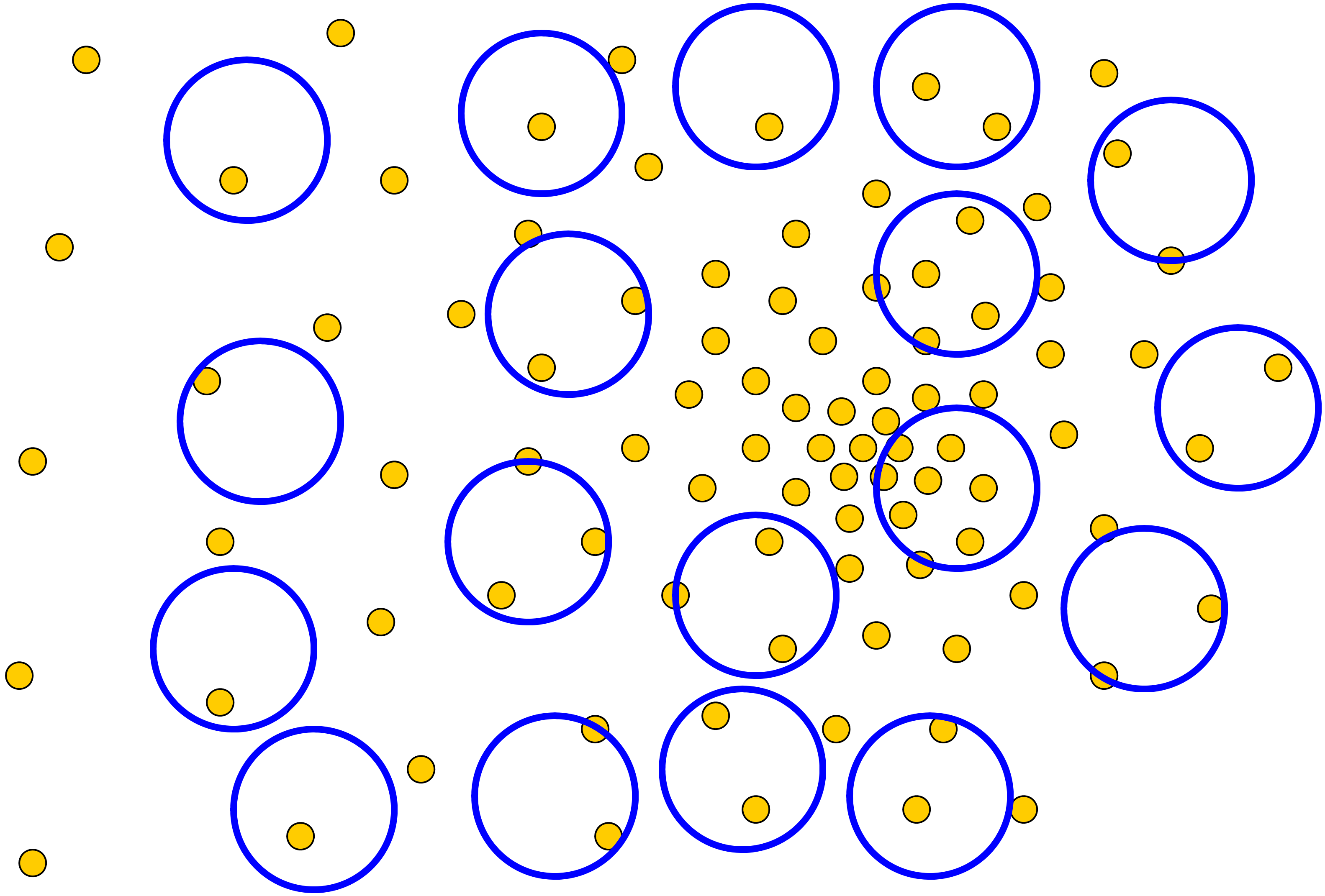
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



Slide by Y. Ukrainitz & B. Sarel

Real Modality Analysis

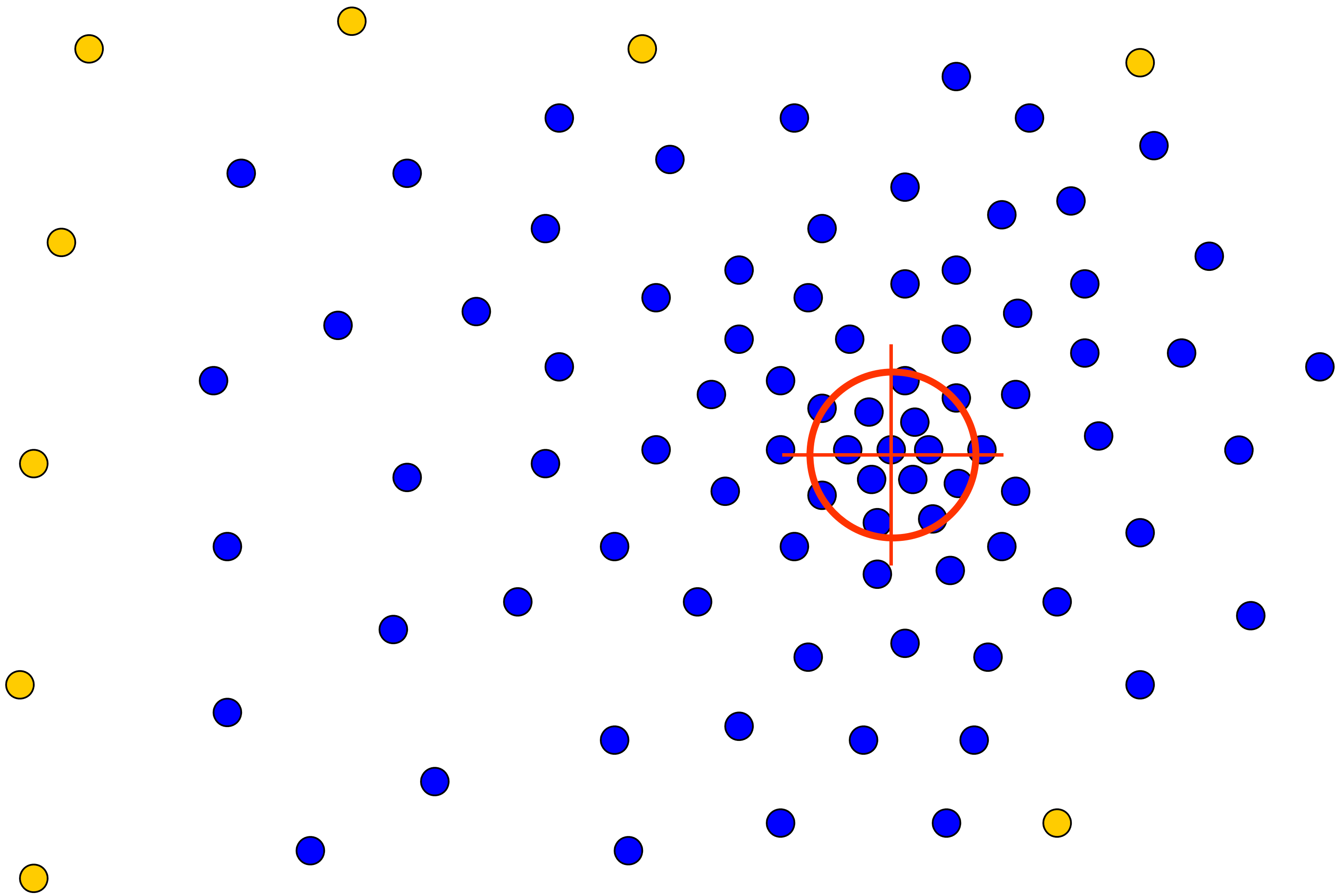


Tessellate the space with windows

Run the procedure in parallel

Slide by Y. Ukrainitz & B. Sarel

Real Modality Analysis

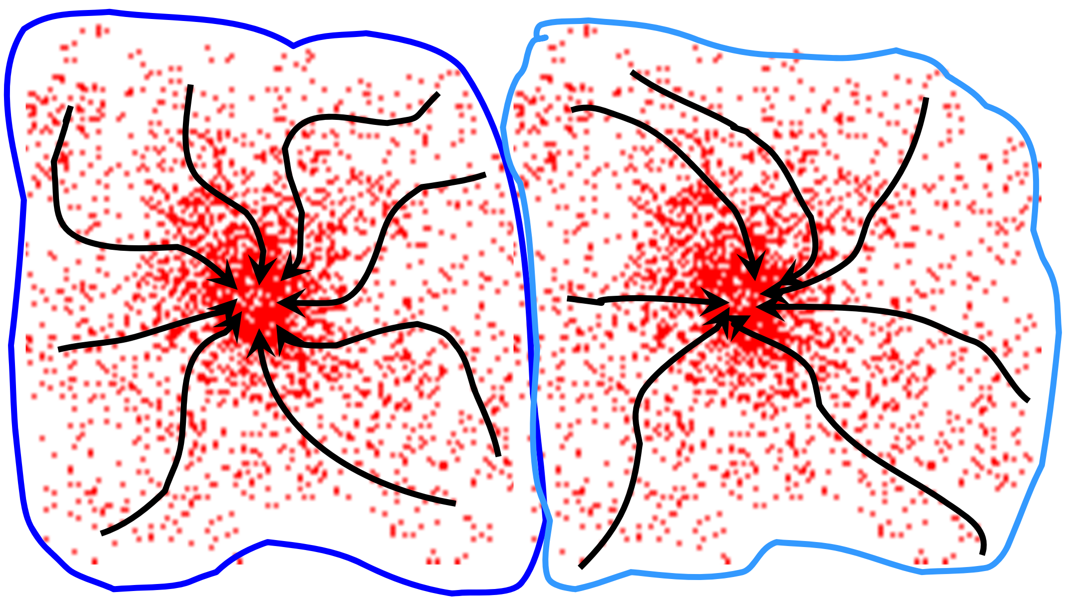


The blue data points were traversed by the windows towards the mode.

Slide by Y. Ukrainitz & B. Sarel

Mean-Shift Clustering

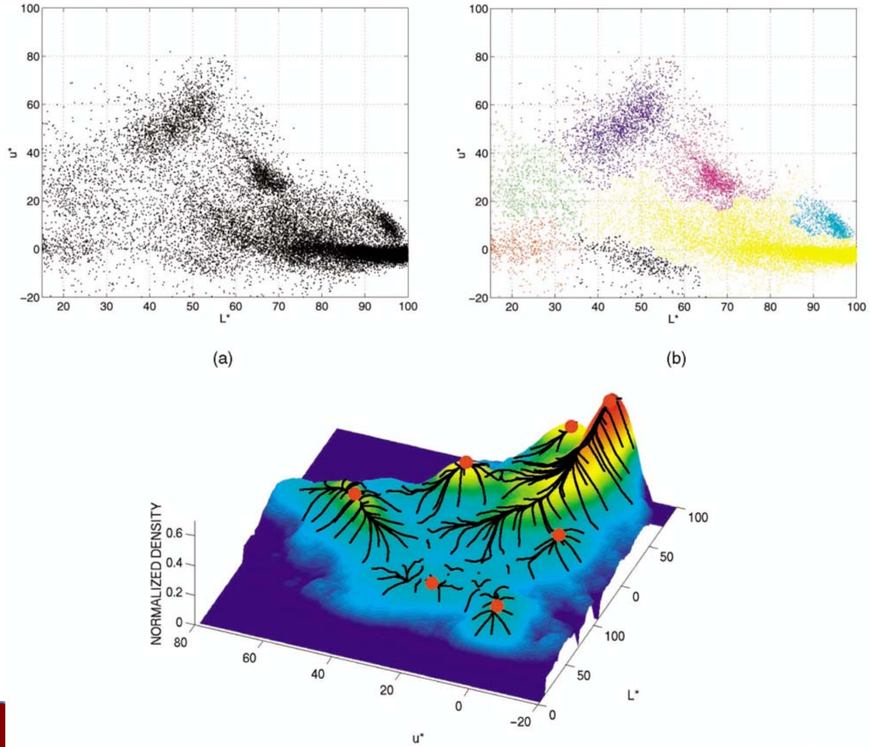
- Cluster: all data points in the attraction basin of a mode
- Attraction basin: the region for which all trajectories lead to the same mode



Slide by Y. Ukrainitz & B. Sarel

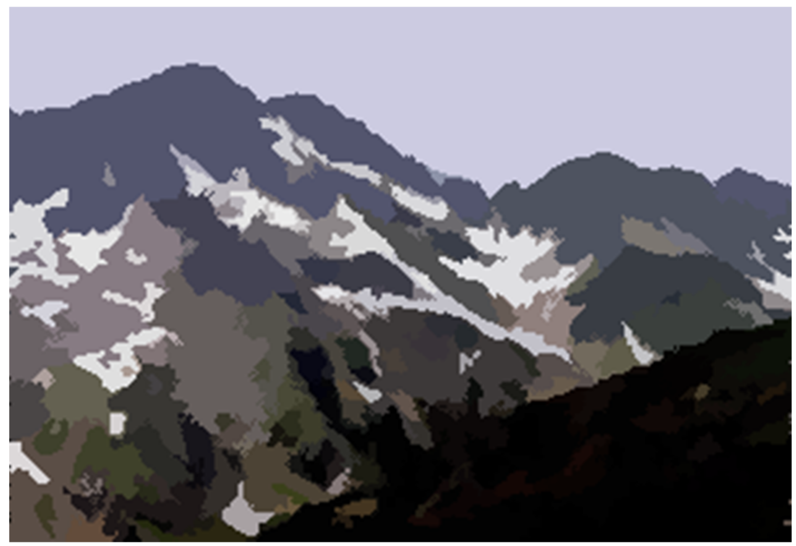
Mean-Shift Clustering/Segmentation

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



Slide credit: Svetlana Lazebnik

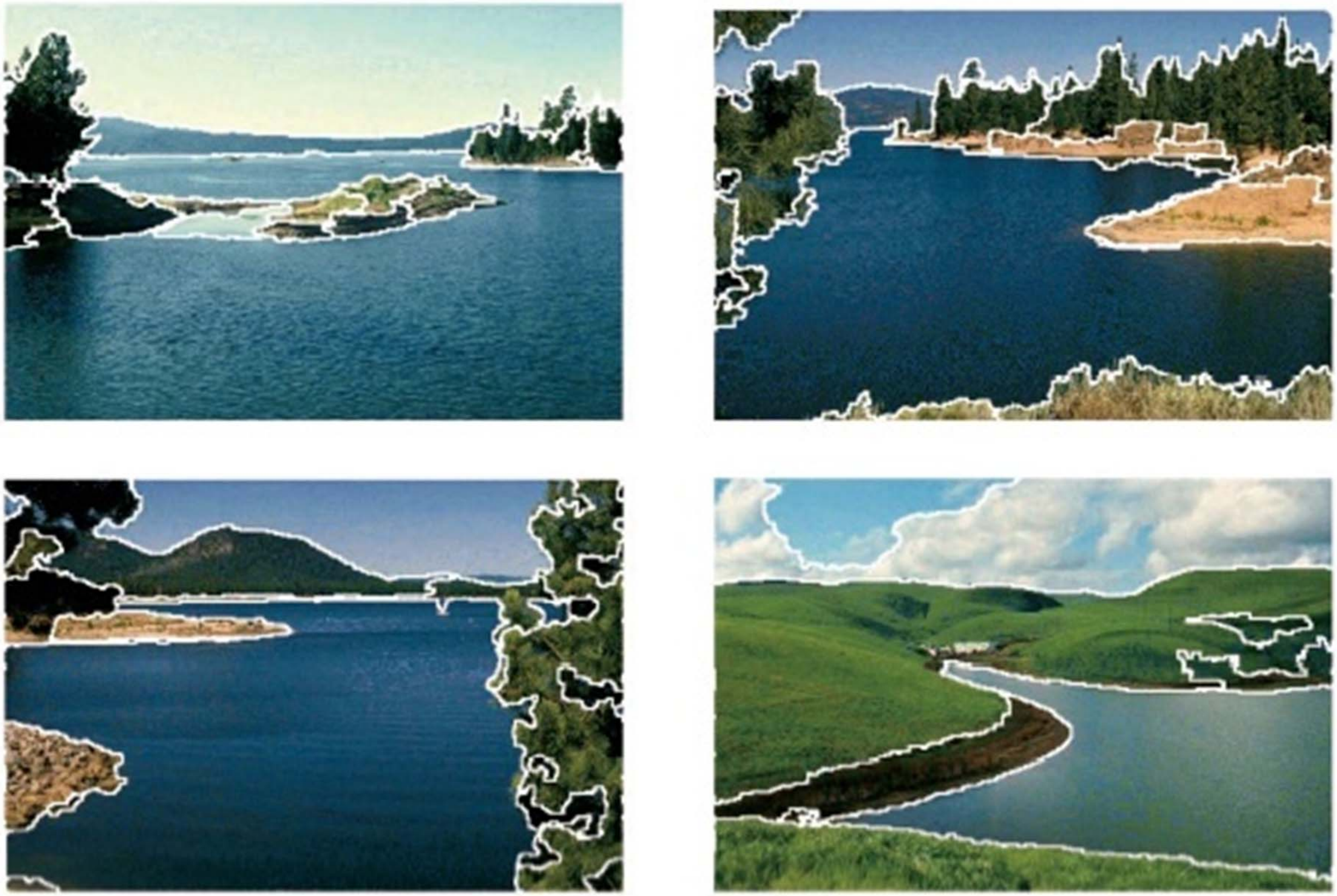
Mean-Shift Segmentation Results



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

Slide credit: Svetlana Lazebnik

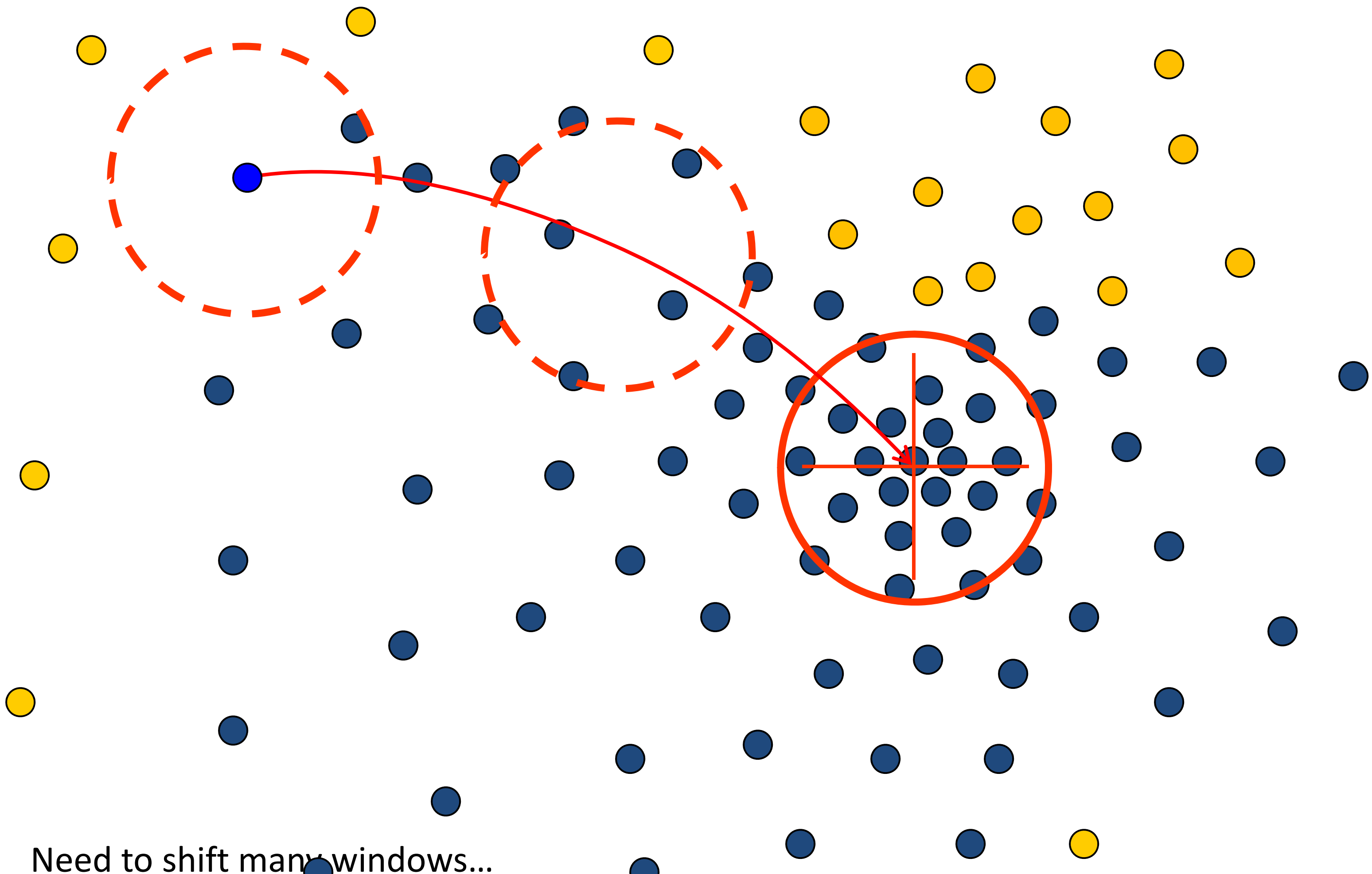
More Results



Slide credit: Svetlana Lazebnik

More Results

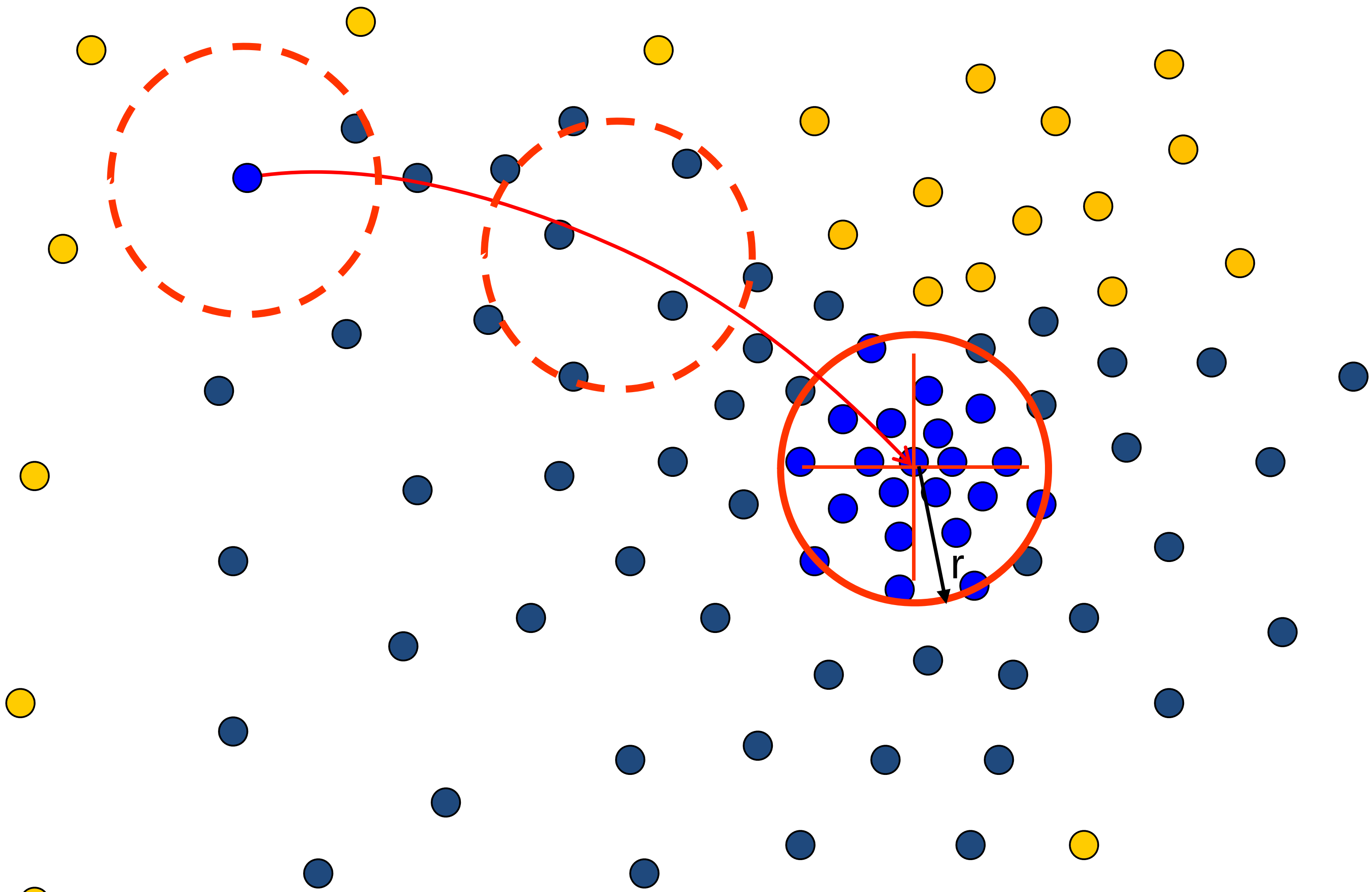
Problem: Computational Complexity



- Need to shift many windows…
- Many computations will be redundant.

Slide credit: Bastian Leibe

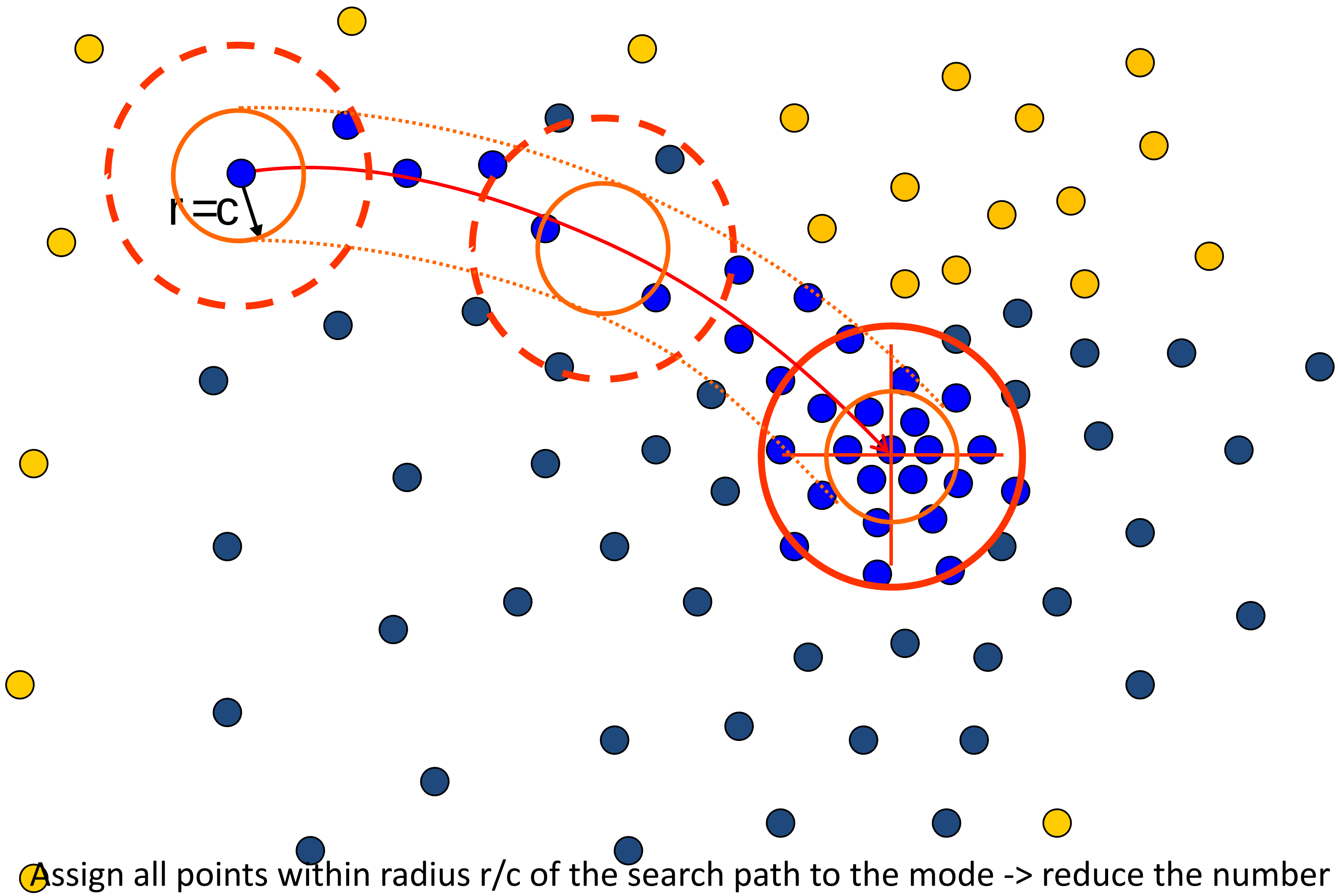
Speedups: Basin of Attraction



1. Assign all points within radius r of end point to the mode.

Slide credit: Bastian Leibe

Speedups



2. Assign all points within radius r/c of the search path to the mode -> reduce the number of data points to search.

Slide credit: Bastian Leibe

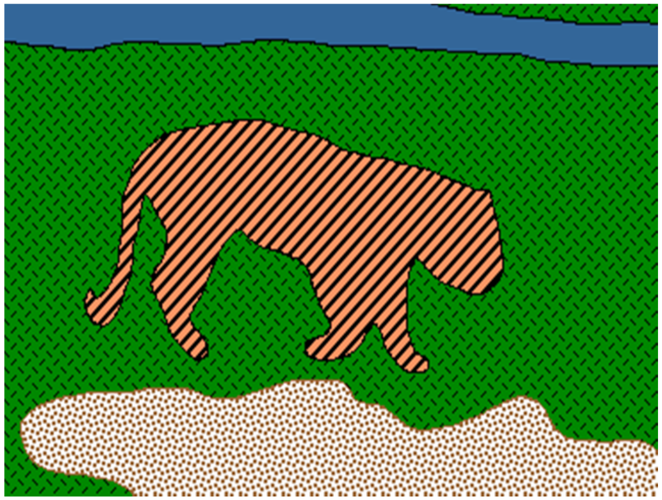
Summary Mean-Shift

- Pros
  - General, application-independent tool
  - Model-free, does not assume any prior shape (spherical, elliptical, etc.) on data clusters
  - Just a single parameter (window size h)
    - h has a physical meaning (unlike k-means)
  - Finds variable number of modes
  - Robust to outliers
- Cons
  - Output depends on window size
  - Window size (bandwidth) selection is not trivial
  - Computationally (relatively) expensive (~2s/image)
  - Does not scale well with dimension of feature space

Slide credit: Svetlana Lazebnik

Back to the Image Segmentation Problem…

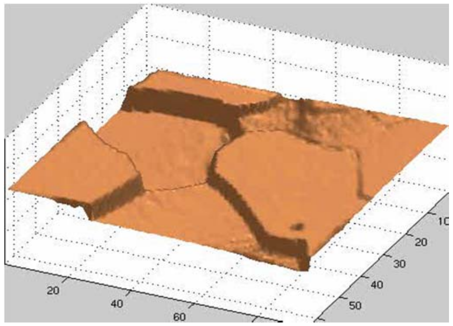
- Goal: identify groups of pixels that go together



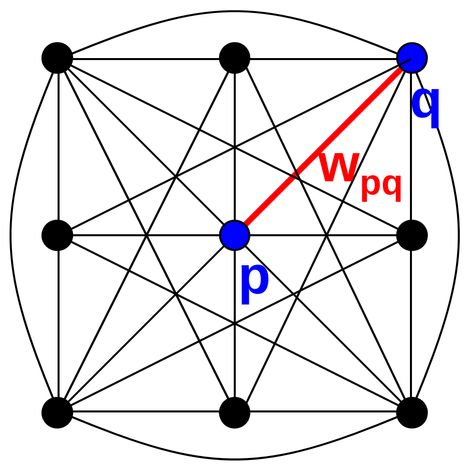
- Up to now, we have focused on ways to group pixels into image segments based on their appearance…
  - Segmentation as clustering.
- We also want to enforce region constraints.
  - Spatial consistency
  - Smooth borders

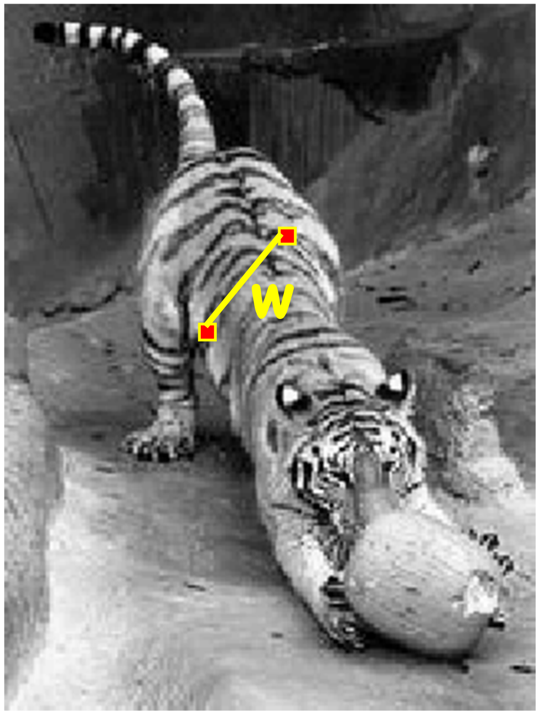
What we will learn today?

- Model free clustering
  - Mean-shift
- Graph theoretic segmentation
  - Normalized Cuts
  - Using texture features
- Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation (supp. materials)
  - s-t mincut algorithm (supp. materials)
  - Extension to non-binary case (supp. materials)
  - Applications



Images as Graphs

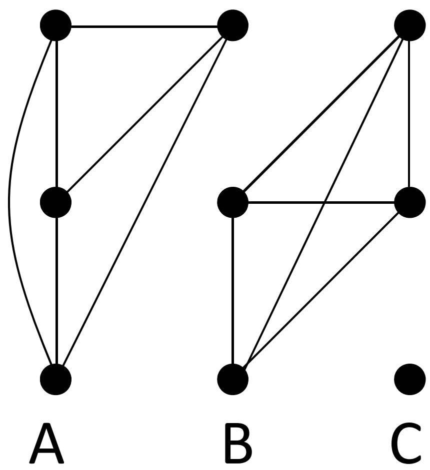




- *Fully-connected* graph
  - Node (vertex) for every pixel
  - Link between *every* pair of pixels, (p,q)
  - Affinity weight w_{pq} for each link (edge)
    - w_{pq} measures similarity
    - Similarity is *inversely proportional* to difference (in color and position…)

Slide credit: Steve Seitz

Segmentation by Graph Cuts



- Break Graph into Segments
  - 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

Slide credit: Steve Seitz

Measuring Affinity

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

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

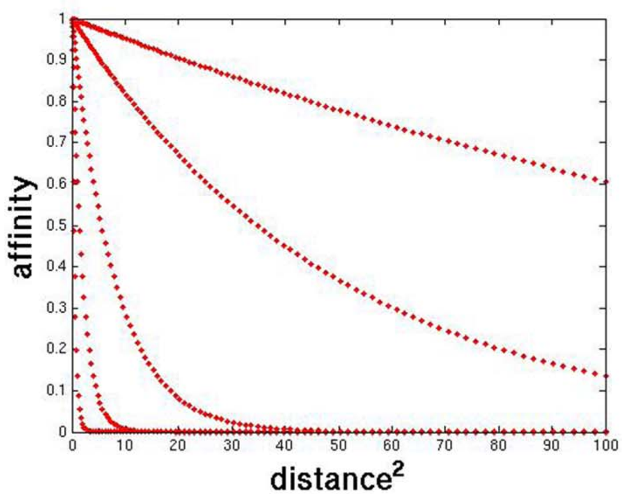
- Color $$aff(x,y) = \exp\left\{-\frac{1}{2\sigma_d^2}\underbrace{dist\left(c(x),c(y)\right)}_{\textbf{(some suitable color space distance)}}^2\right\}$$

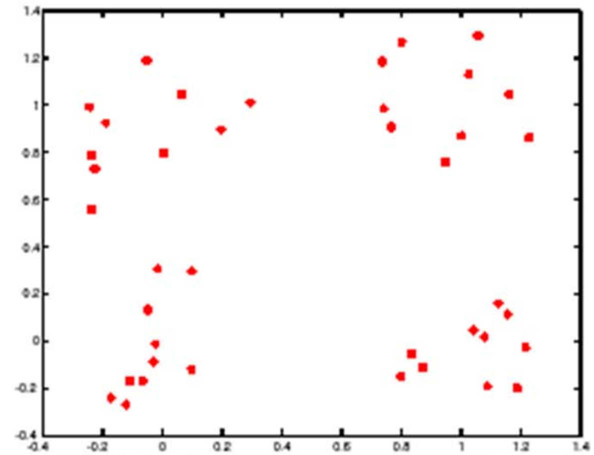
- Texture $$aff(x,y) = \exp\left\{-\frac{1}{2\sigma_d^2}\underbrace{\left\|f(x)-f(y)\right\|}_{\textbf{(vectors of filter outputs)}}^2\right\}$$

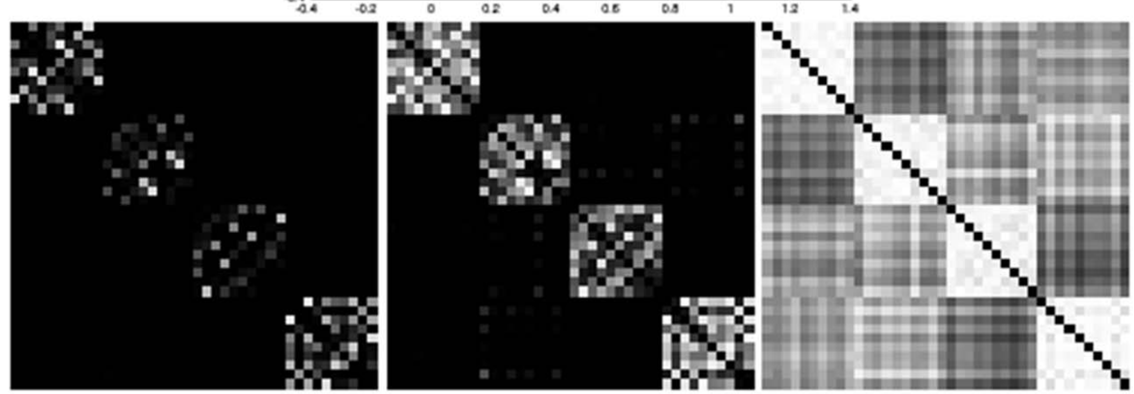
Source: Forsyth & Ponce

Scale Affects Affinity

- Small σ: group only nearby points
- Large σ: group far-away points







Small σ Medium σ Large σ

Slide credit: Svetlana Lazebnik

Graph Cut: using Eigenvalues

- Extract a single good cluster
  - Where elements have high affinity values with each other

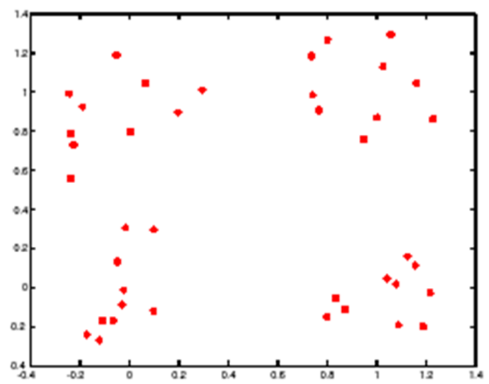
$$w_n^T A w_n$$

$$\{\text{association of element } i \text{ with cluster } n\} \times$$
$$\{\text{affinity between } i \text{ and } j\} \times$$
$$\{\text{association of element } j \text{ with cluster } n\}$$

$$w_n^T A w_n + \lambda \left(w_n^T w_n - 1 \right)$$

$$A w_n = \lambda w_n$$

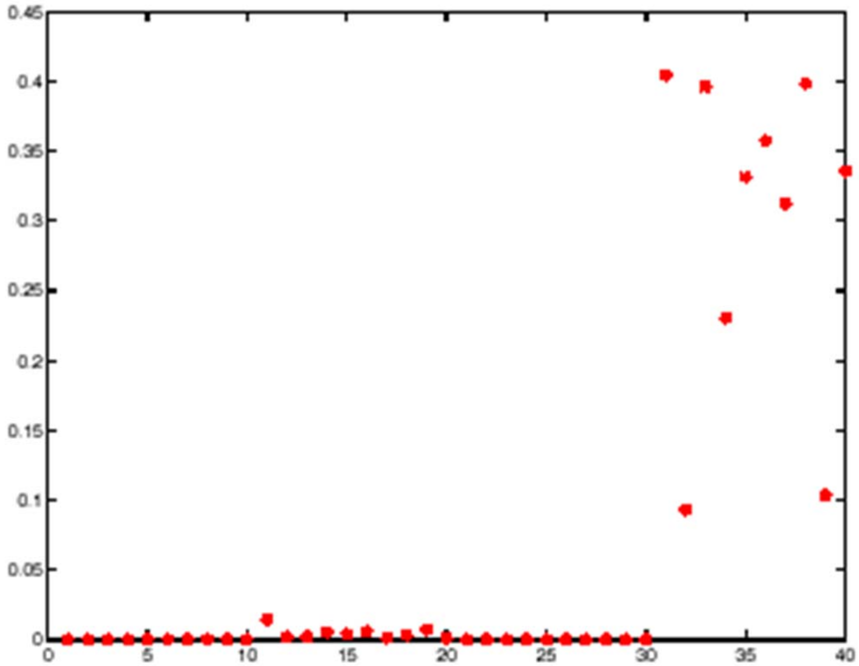
points



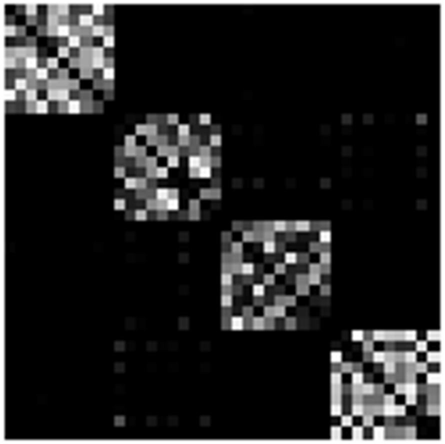
Graph Cut: using Eigenvalues

- Extract a single good cluster

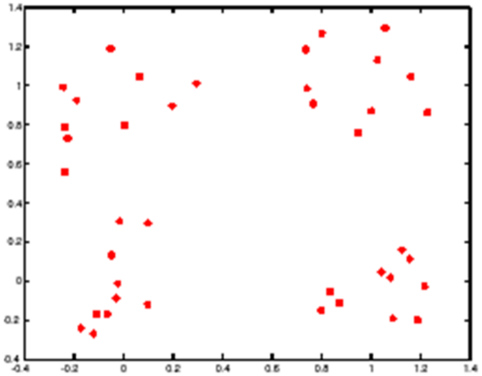
Eigenvector associated w/ the largest eigenvalue



matrix



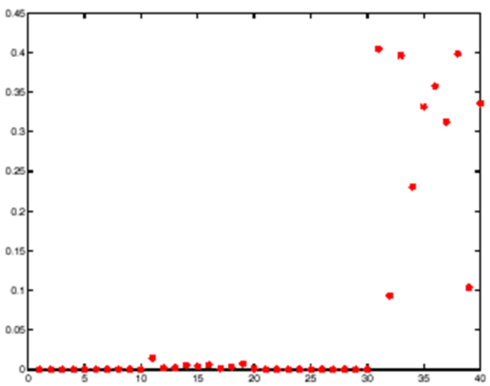
points



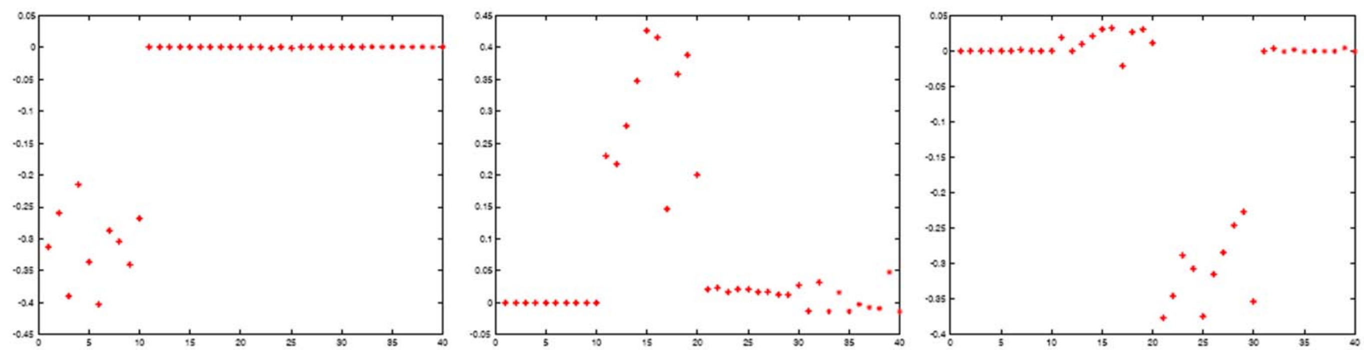
Graph Cut: using Eigenvalues

- Extract a single good cluster
- Extract weights for a set of clusters

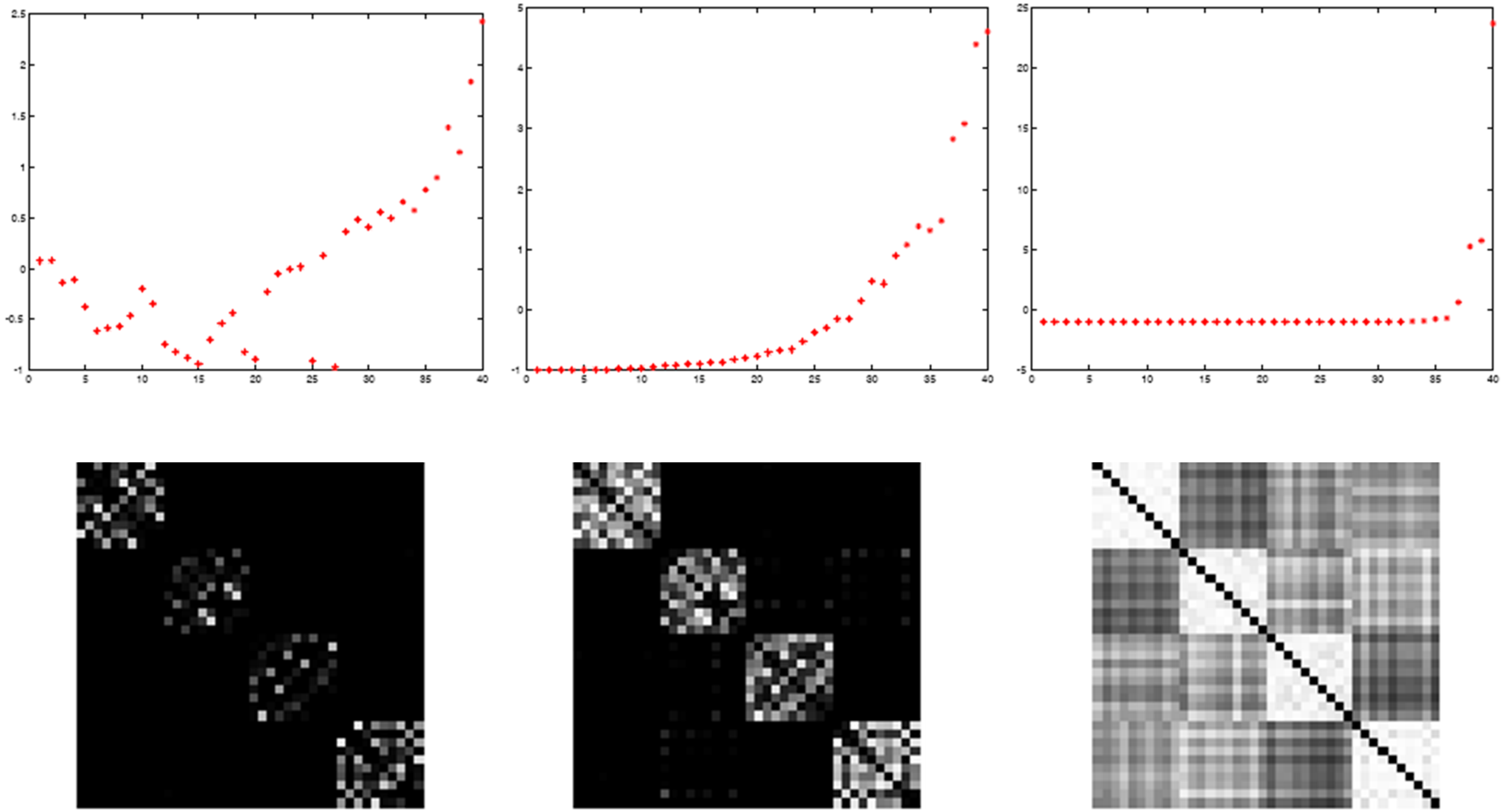
Eigenvector associated w/ the largest eigenvalue



Eigenvectors associated with other eigenvalues



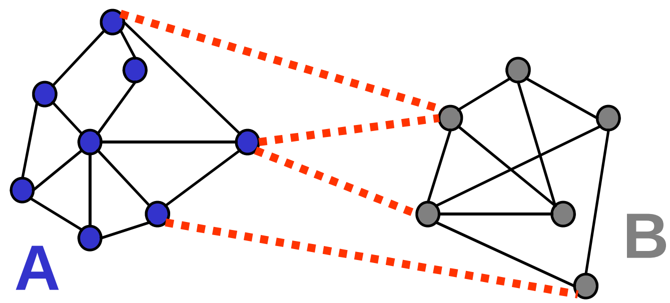
Graph Cut: using Eigenvalues (effect of the scaling factor)



Algorithm 14.6: Clustering by Graph Eigenvectors

Construct an affinity matrix
Compute the eigenvalues and eigenvectors of the affinity matrix
Until there are sufficient clusters
    Take the eigenvector corresponding to the
        largest unprocessed eigenvalue; zero all components corresponding
        to elements that have already been clustered, and threshold the
        remaining components to determine which element
        belongs to this cluster, choosing a threshold by
        clustering the components, or
        using a threshold fixed in advance.
    If all elements have been accounted for, there are
        sufficient clusters
end

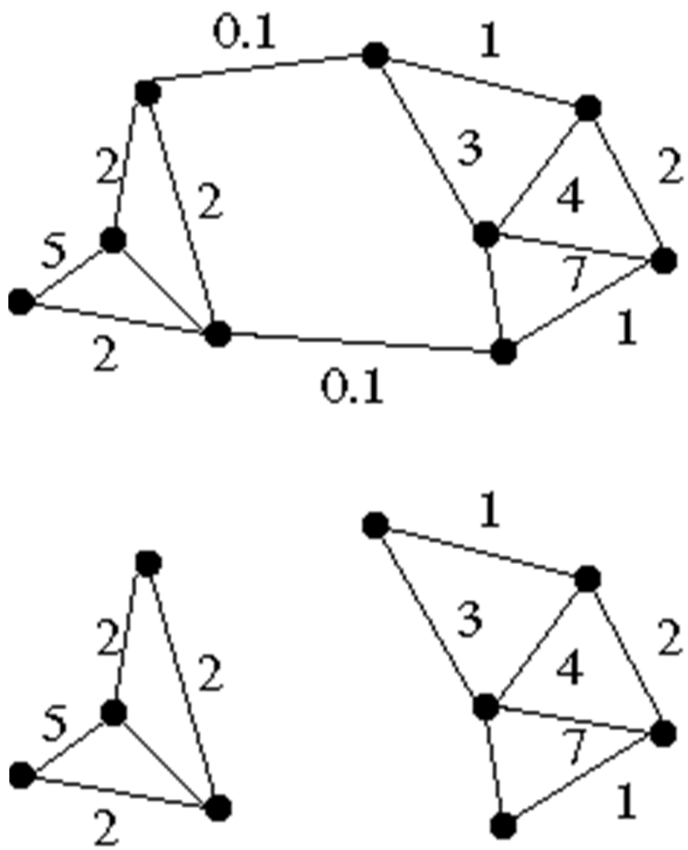
Graph Cut

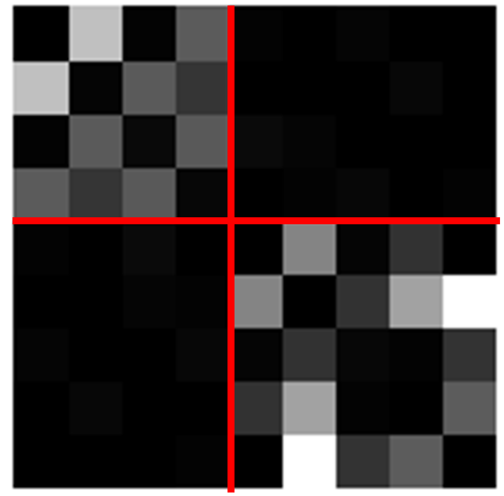


- Set of edges whose removal makes a graph disconnected
- Cost of a cut
  – Sum of weights of cut edges: $cut(A, B) = \sum_{p \in A, q \in B} w_{p,q}$
- A graph cut gives us a segmentation
  – What is a "good" graph cut and how do we find one?

Slide credit: Steve Seitz

Graph Cut





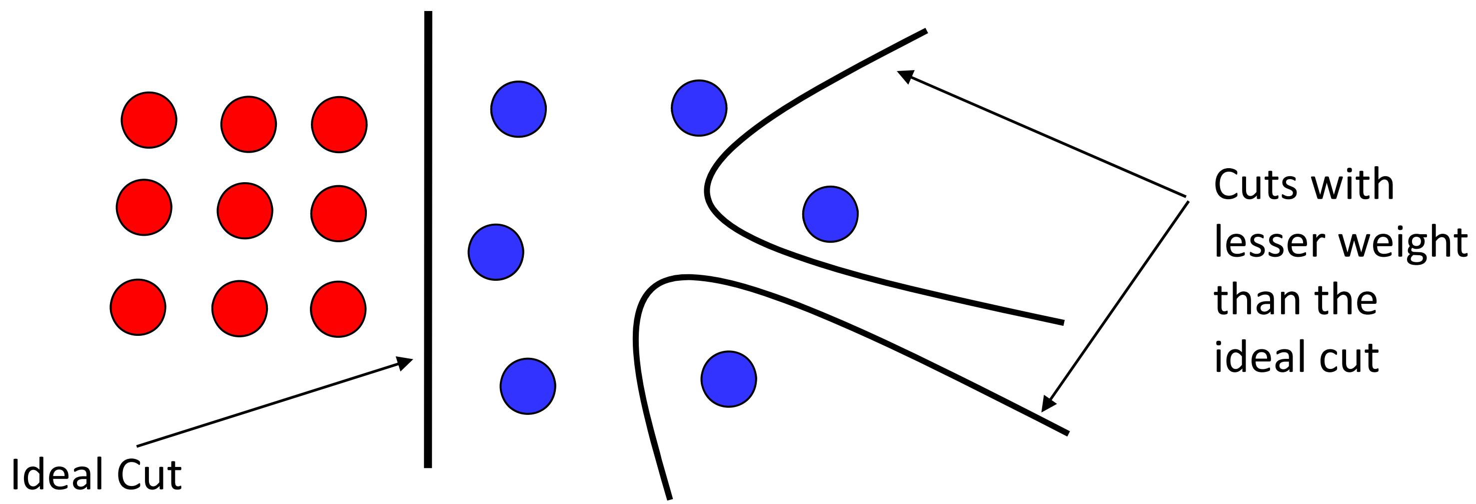
Here, the cut is nicely defined by the block-diagonal structure of the affinity matrix.

⇒ *How can this be generalized?*

Image Source: Forsyth & Ponce

Minimum Cut

- We can do segmentation by finding the *minimum cut* in a graph
  - a **minimum cut** of a graph is a cut whose cutset has the smallest number of elements (unweighted case) or smallest sum of weights possible.
  - Efficient algorithms exist for doing this
- Drawback:
  - Weight of cut proportional to number of edges in the cut
  - Minimum cut tends to cut off very small, isolated components



Slide credit: Khurram Hassan-Shafique

Normalized Cut (NCut)

- A minimum cut penalizes large segments
- This can be fixed by normalizing for size of segments
- The normalized cut cost is:

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

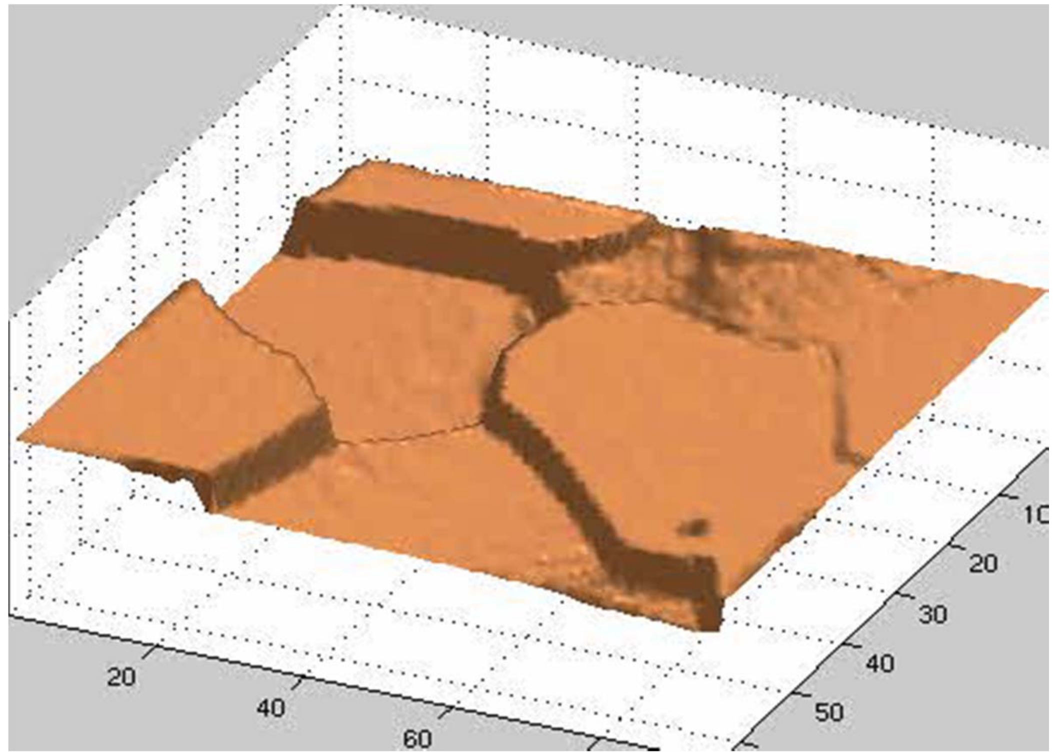
assoc(*A*,*V*) = sum of weights of all edges in V that touch A

$$= cut(A,B)\left[\frac{1}{\sum_{p\in A} w_{p,q}} + \frac{1}{\sum_{q\in B} w_{p,q}}\right]$$

- The exact solution is NP-hard but an approximation can be computed by solving a *generalized eigenvalue* problem.

J. Shi and J. Malik. Normalized cuts and image segmentation. PAMI 2000

Interpretation as a Dynamical System



- Treat the links as springs and shake the system
  - Elasticity proportional to cost
  - Vibration “modes” correspond to segments
    - Can compute these by solving a generalized eigenvector problem

Slide credit: Steve Seitz

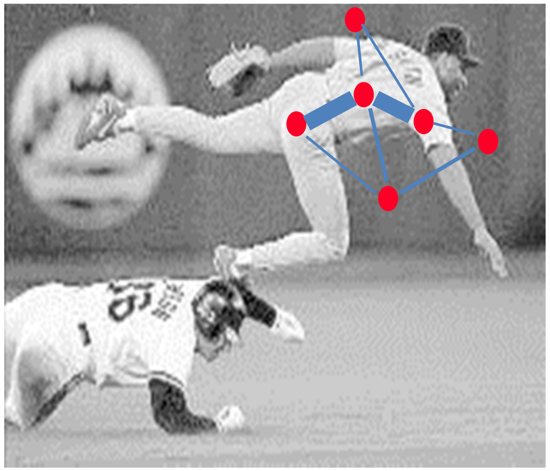
NCuts as a Generalized Eigenvector Problem

- Definitions

  W : **the affinity matrix,** $W(i,j) = w_{i,j};$

  D : **the diag. matrix,** $D(i,i) = \sum_j W(i,j);$

  x : **a vector in** $\{1,-1\}^N, x(i) = 1 \Leftrightarrow i \in A.$



- Rewriting Normalized Cut in matrix form:

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

$$= \frac{(1+x)^T(D-W)(1+x)}{k1^T D1} + \frac{(1-x)^T(D-W)(1-x)}{(1-k)1^T D1}; \quad k = \frac{\sum_{x_i>0} D(i,i)}{\sum_i D(i,i)}$$

$$= \ \ldots$$

Slide credit: Jitendra Malik

Some More Math…

We see again this is an unbiased measure, which reflects how tightly on average nodes within the group are connected to each other.

Another important property of this definition of association and disassociation of a partition is that they are naturally related:

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

Hence the two partition criteria that we seek in our grouping algorithm, minimizing the disassociation between the groups and maximizing the association within the group, are in fact identical, and can be satisfied simultaneously. In our algorithm, we will use this *normalized cut* as the partition criterion.

Having defined the graph partition criterion that we want to optimize, we will show how such an optimal partition can be computed efficiently.

2.1 Computing the optimal partition

Given a partition of nodes of a graph, V, into two sets A and B, let $\mathbf{x}$ be an $N = |\mathbf{V}|$ dimensional indicator vector, $x_i = 1$ if node i is in A, and -1 otherwise. Let $\mathbf{d}(i) = \sum_j w(i,j)$, be the total connection from node i to all other nodes. With the definitions $\mathbf{x}$ and $\mathbf{d}$ we can rewrite $Ncut(A,B)$ as:

$$\begin{aligned} Ncut(A,B) &= \frac{cut(A,B)}{asso(A,V)} + \frac{cut(B,A)}{asso(B,V)} \\ &= \frac{\sum_{(\mathbf{x}_i>0,\mathbf{x}_j<0)} -w_{ij}\mathbf{x}_i\mathbf{x}_j}{\sum_{\mathbf{x}_i>0} \mathbf{d}_i} \\ &\quad + \frac{\sum_{(\mathbf{x}_i<0,\mathbf{x}_j>0)} -w_{ij}\mathbf{x}_i\mathbf{x}_j}{\sum_{\mathbf{x}_i<0} \mathbf{d}_i} \end{aligned}$$

Let $\mathbf{D}$ be an $N \times N$ diagonal matrix with $\mathbf{d}$ on its diagonal, $\mathbf{W}$ be an $N \times N$ symmetrical matrix with $\mathbf{W}(i,j) = w_{ij}$, $\mathbf{k} = \frac{\sum_{\mathbf{x}_i>0} \mathbf{d}_i}{\sum_i \mathbf{d}_i}$, and $\mathbf{1}$ be an $N \times 1$ vector of all ones. Using the fact $\frac{\mathbf{1}+\mathbf{x}}{2}$ and $\frac{\mathbf{1}-\mathbf{x}}{2}$ are indicator vectors for $x_i > 0$ and $x_i < 0$ respectively, we can rewrite $4[Ncut(\mathbf{x})]$ as:

$$\begin{aligned} &= \frac{(\mathbf{1}+\mathbf{x})^T(\mathbf{D}-\mathbf{W})(\mathbf{1}+\mathbf{x})}{k\mathbf{1}^T\mathbf{D}\mathbf{1}} + \frac{(\mathbf{1}-\mathbf{x})^T(\mathbf{D}-\mathbf{W})(\mathbf{1}-\mathbf{x})}{(1-k)\mathbf{1}^T\mathbf{D}\mathbf{1}} \\ &= \frac{(\mathbf{x}^T(\mathbf{D}-\mathbf{W})\mathbf{x}+\mathbf{1}^T(\mathbf{D}-\mathbf{W})\mathbf{1})}{k(1-k)\mathbf{1}^T\mathbf{D}\mathbf{1}} + \frac{2(1-2k)\mathbf{1}^T(\mathbf{D}-\mathbf{W})\mathbf{x}}{k(1-k)\mathbf{1}^T\mathbf{D}\mathbf{1}} \end{aligned}$$

Let $\alpha(\mathbf{x}) = \mathbf{x}^T(\mathbf{D}-\mathbf{W})\mathbf{x}$, $\beta(\mathbf{x}) = \mathbf{1}^T(\mathbf{D}-\mathbf{W})\mathbf{x}$, $\gamma = \mathbf{1}^T(\mathbf{D}-\mathbf{W})\mathbf{1}$, and $M = \mathbf{1}^T\mathbf{D}\mathbf{1}$, we can then further expand the above equation as:

$$\begin{aligned} &= \frac{(\alpha(\mathbf{x})+\gamma) + 2(1-2\mathbf{k})\beta(\mathbf{x})}{\mathbf{k}(1-\mathbf{k})M} \\ &= \frac{(\alpha(\mathbf{x})+\gamma) + 2(1-2\mathbf{k})\beta(\mathbf{x})}{\mathbf{k}(1-\mathbf{k})M} - \frac{2(\alpha(\mathbf{x})+\gamma)}{M} \\ &\quad + \frac{2\alpha(\mathbf{x})}{M} + \frac{2\gamma}{M} \end{aligned}$$

dropping the last constant term, which in this case equals 0, we get

$$\begin{aligned} &= \frac{(1-2\mathbf{k}+2\mathbf{k}^2)(\alpha(\mathbf{x})+\gamma) + 2(1-2\mathbf{k})\beta(\mathbf{x})}{\mathbf{k}(1-\mathbf{k})M} + \frac{2\alpha(\mathbf{x})}{M} \\ &= \frac{\frac{(1-2k+2k^2)}{(1-k)^2}(\alpha(\mathbf{x})+\gamma) + \frac{2(1-2k)}{(1-k)^2}\beta(\mathbf{x})}{\frac{k}{1-k}M} + \frac{2\alpha(\mathbf{x})}{M} \end{aligned}$$

Letting $b = \frac{k}{1-k}$, and since $\gamma = 0$, it becomes,

$$\begin{aligned} &= \frac{(1+b^2)(\alpha(\mathbf{x})+\gamma) + 2(1-b^2)\beta(\mathbf{x})}{bM} + \frac{2b\alpha(\mathbf{x})}{bM} \\ &= \frac{(1+b^2)(\alpha(\mathbf{x})+\gamma)}{bM} + \frac{2(1-b^2)\beta(\mathbf{x})}{bM} + \frac{2b\alpha(\mathbf{x})}{bM} - \frac{2b\gamma}{bM} \\ &= \frac{(1+b^2)(\mathbf{x}^T(\mathbf{D}-\mathbf{W})\mathbf{x}+\mathbf{1}^T(\mathbf{D}-\mathbf{W})\mathbf{1})}{b\mathbf{1}^T\mathbf{D}\mathbf{1}} \\ &\quad + \frac{2(1-b^2)\mathbf{1}^T(\mathbf{D}-\mathbf{W})\mathbf{x}}{b\mathbf{1}^T\mathbf{D}\mathbf{1}} \\ &\quad + \frac{2b\mathbf{x}^T(\mathbf{D}-\mathbf{W})\mathbf{x}}{b\mathbf{1}^T\mathbf{D}\mathbf{1}} - \frac{2b\mathbf{1}^T(\mathbf{D}-\mathbf{W})\mathbf{1}}{b\mathbf{1}^T\mathbf{D}\mathbf{1}} \\ &= \frac{(\mathbf{1}+\mathbf{x})^T(\mathbf{D}-\mathbf{W})(\mathbf{1}+\mathbf{x})}{b\mathbf{1}^T\mathbf{D}\mathbf{1}} \\ &\quad + \frac{b^2(\mathbf{1}-\mathbf{x})^T(\mathbf{D}-\mathbf{W})(\mathbf{1}-\mathbf{x})}{b\mathbf{1}^T\mathbf{D}\mathbf{1}} \\ &\quad - \frac{2b(\mathbf{1}-\mathbf{x})^T(\mathbf{D}-\mathbf{W})(\mathbf{1}+\mathbf{x})}{b\mathbf{1}^T\mathbf{D}\mathbf{1}} \\ &= \frac{[(\mathbf{1}+\mathbf{x})-b(\mathbf{1}-\mathbf{x})]^T(\mathbf{D}-\mathbf{W})[(\mathbf{1}+\mathbf{x})-b(\mathbf{1}-\mathbf{x})]}{b\mathbf{1}^T\mathbf{D}\mathbf{1}} \end{aligned}$$

Setting $\mathbf{y} = (\mathbf{1}+\mathbf{x}) - b(\mathbf{1}-\mathbf{x})$, it is easy to see that

$$\mathbf{y}^T\mathbf{D}\mathbf{1} = \sum_{x_i>0} \mathbf{d}_i - b\sum_{x_i<0} \mathbf{d}_i = 0 \qquad (4)$$

since $b = \frac{k}{1-k} = \frac{\sum_{x_i>0} \mathbf{d}_i}{\sum_{x_i<0} \mathbf{d}_i}$, and

$$\begin{aligned} \mathbf{y}^T\mathbf{D}\mathbf{y} &= \sum_{x_i>0} \mathbf{d}_i + b^2\sum_{x_i<0} \mathbf{d}_i \\ &= b\sum_{x_i<0} \mathbf{d}_i + b^2\sum_{x_i<0} \mathbf{d}_i \\ &= b\left(\sum_{x_i<0} \mathbf{d}_i + b\sum_{x_i<0} \mathbf{d}_i\right) \\ &= b\mathbf{1}^T\mathbf{D}\mathbf{1}. \end{aligned}$$

Slide credit: Jitendra Malik

NCuts as a Generalized Eigenvalue Problem

- After simplification, we get

$$NCut(A,B) = \frac{y^T (D-W) y}{y^T D y}, \quad \text{with } y_i \in \{1, -b\},\ y^T D1 = 0.$$

This is hard, as y is discrete!

- This is a Rayleigh Quotient
  - Solution given by the "generalized" eigenvalue problem

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

  - Solved by converting to standard eigenvalue problem

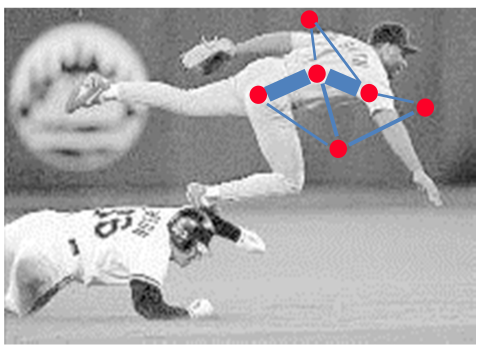
$$D^{-\frac{1}{2}}(D-W)D^{-\frac{1}{2}} z = \lambda z, \quad where\ z = D^{\frac{1}{2}} y$$

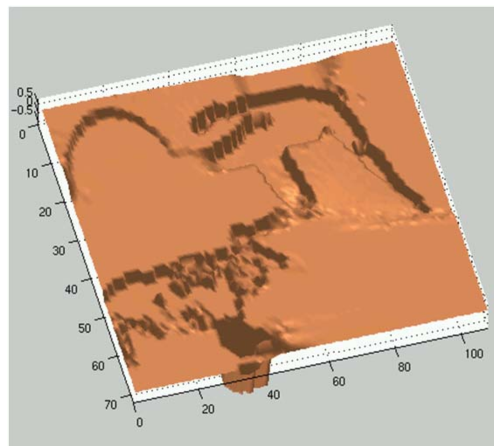
Relaxation: continuous y.

- Subtleties
  - Optimal solution is second smallest eigenvector
  - Gives continuous result—must convert into discrete values of y

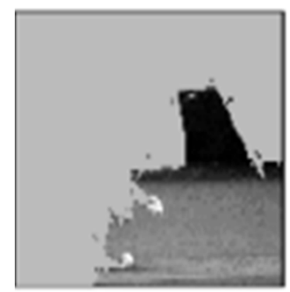
Slide credit: Alyosha Efros

NCuts Example





Smallest eigenvectors

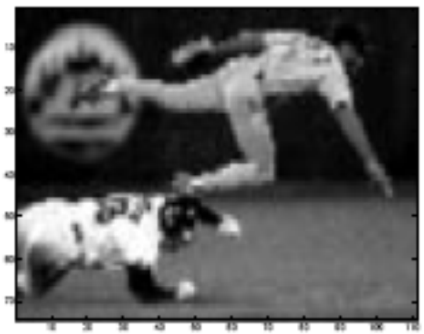


NCuts segments

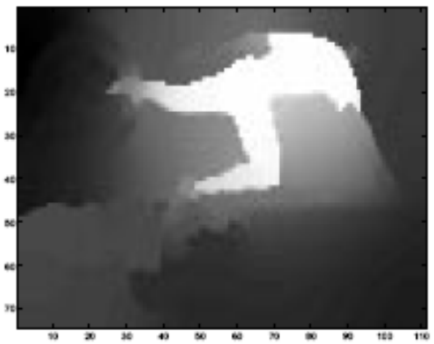
Image source: Shi & Malik

Discretization

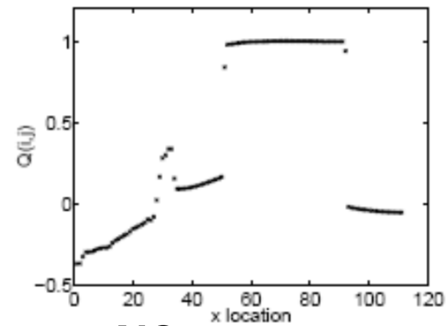
- Problem: eigenvectors take on continuous values
  - How to choose the splitting point to binarize the image?



Image



Eigenvector



NCut scores

- Possible procedures
  a) Pick a constant value (0, or 0.5).
  b) Pick the median value as splitting point.
  c) Look for the splitting point that has the minimum $NCut$ value:
     1. Choose n possible splitting points.
     2. Compute $NCut$ value.
     3. Pick minimum.

NCuts: Overall Procedure

1. Construct a weighted graph *G=(V,E)* from an image.
2. Connect each pair of pixels, and assign graph edge weights,

   $W(i,j) =$ Prob. that i and j belong to the same region.

3. Solve $(D-W)y = \lambda D y$ for the smallest few eigenvectors. This yields a continuous solution.
4. Threshold eigenvectors to get a discrete cut
   – This is where the approximation is made (we're not solving NP).
5. Recursively subdivide if NCut value is below a pre-specified value.

NCuts Matlab code available at
http://www.cis.upenn.edu/~jshi/software/

Slide credit: Jitendra Malik

Color Image Segmentation with NCuts

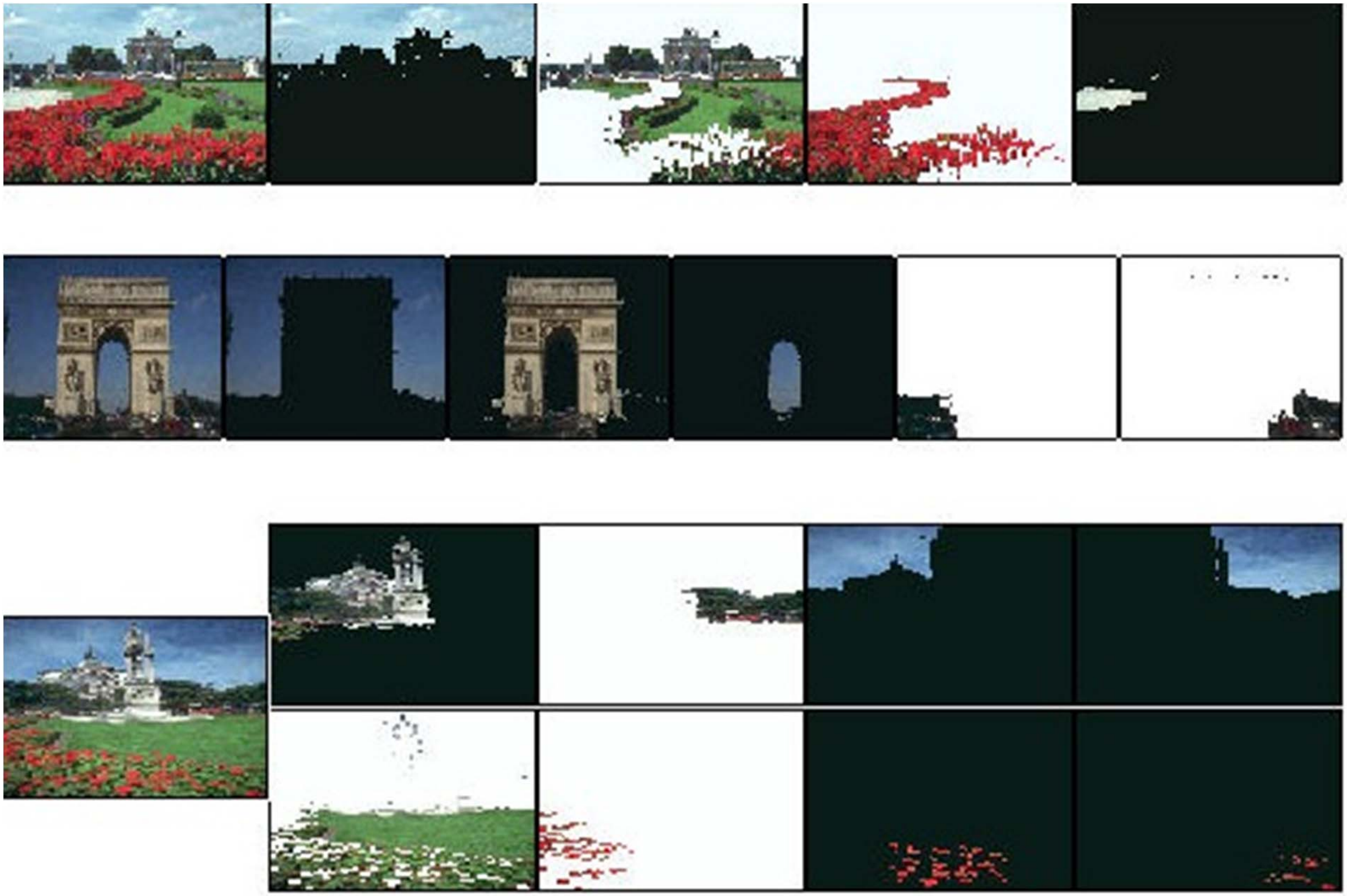
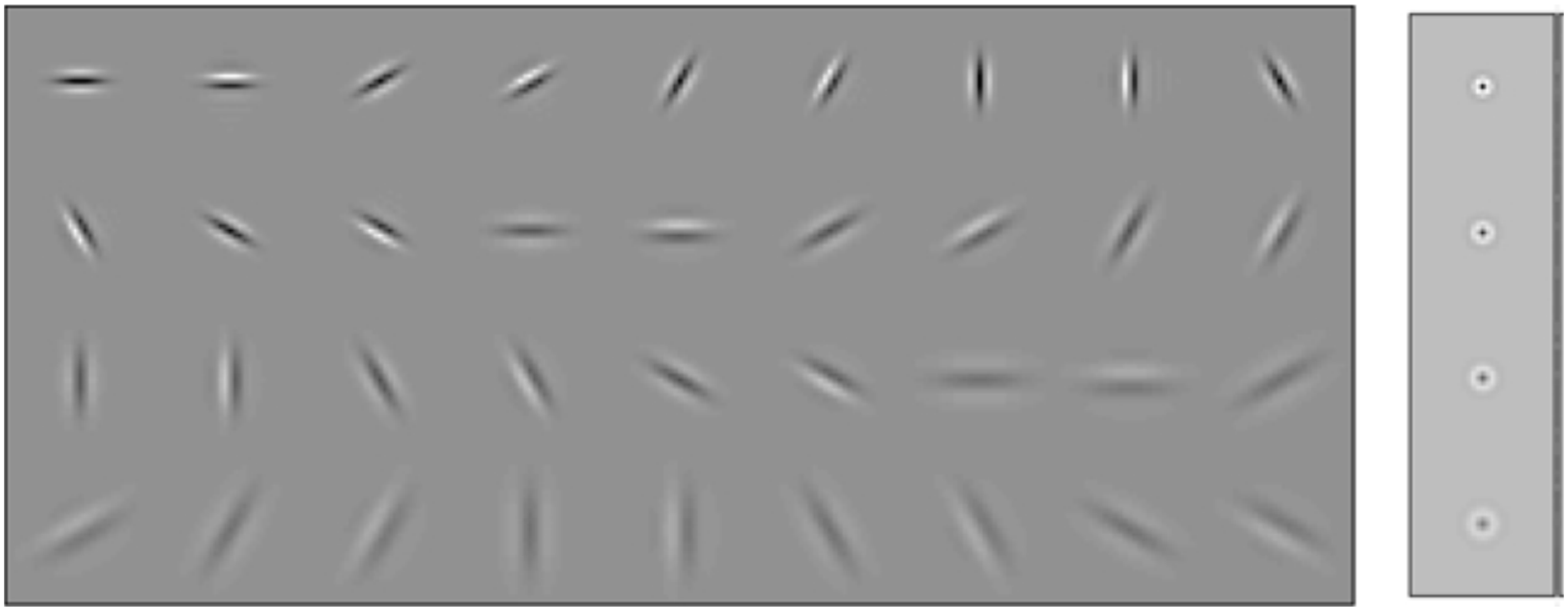


Image Source: Shi & Malik

Using Texture Features for Segmentation

- Texture descriptor is vector of filter bank outputs

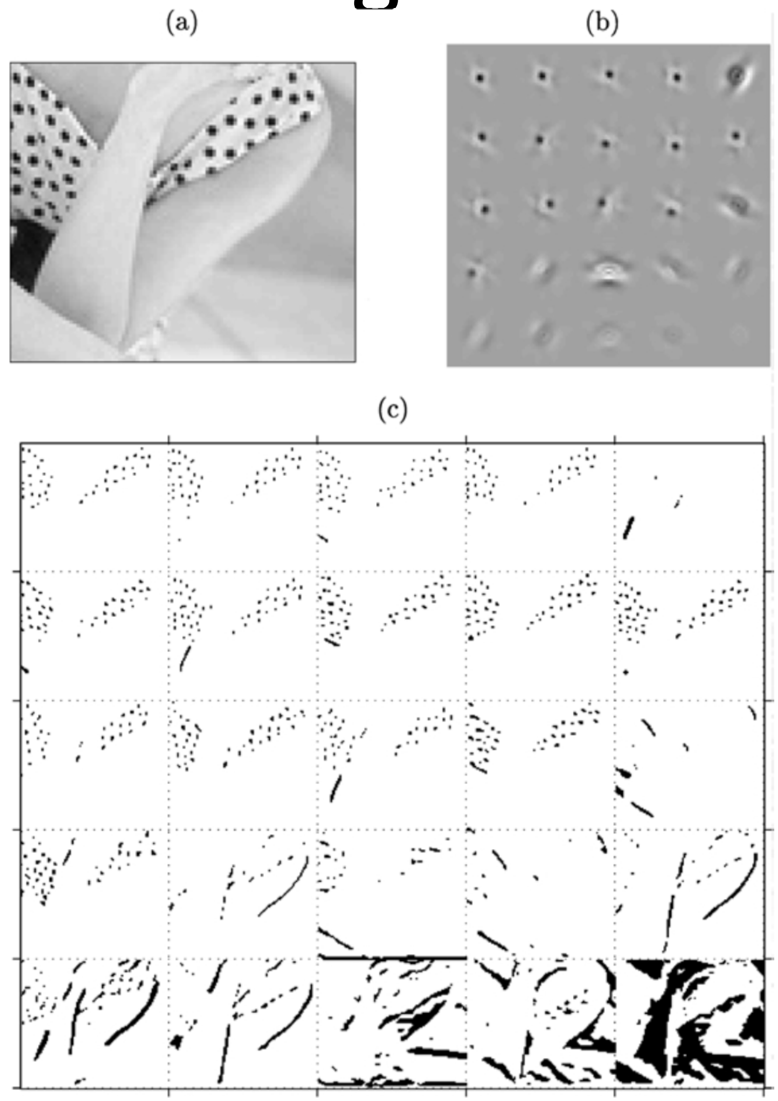


J. Malik, S. Belongie, T. Leung and J. Shi. "*Contour and Texture Analysis for Image Segmentation*". IJCV 43(1),7-27,2001.

Slide credit: Svetlana Lazebnik

Using Texture Features for Segmentation

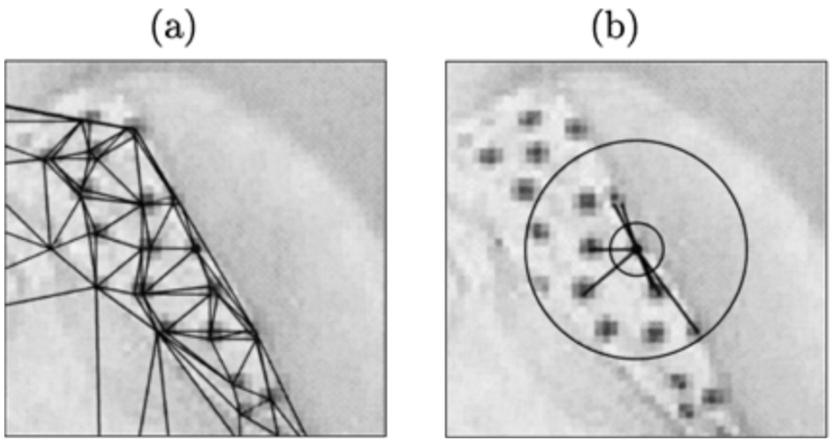
- Texture descriptor is vector of filter bank outputs.
- *Textons* are found by clustering.



Slide credit: Svetlana Lazebnik

Using Texture Features for Segmentation

- Texture descriptor is vector of filter bank outputs.
- *Textons* are found by clustering.
- Affinities are given by similarities of texton histograms over windows given by the "local scale" of the texture .



Slide credit: Svetlana Lazebnik

Results with Color & Texture

Summary: Normalized Cuts

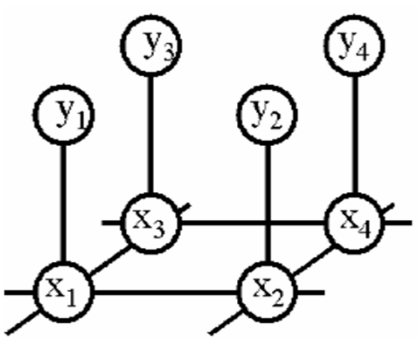
- Pros:
  - Generic framework, flexible to choice of function that computes weights ("affinities") between nodes
  - Does not require any model of the data distribution

- Cons:
  - Time and memory complexity can be high
    - Dense, highly connected graphs $\Rightarrow$ many affinity computations
    - Solving eigenvalue problem for each cut
  - Preference for balanced partitions
    - If a region is uniform, NCuts will find the modes of vibration of the image dimensions

Slide credit: Kristen Grauman

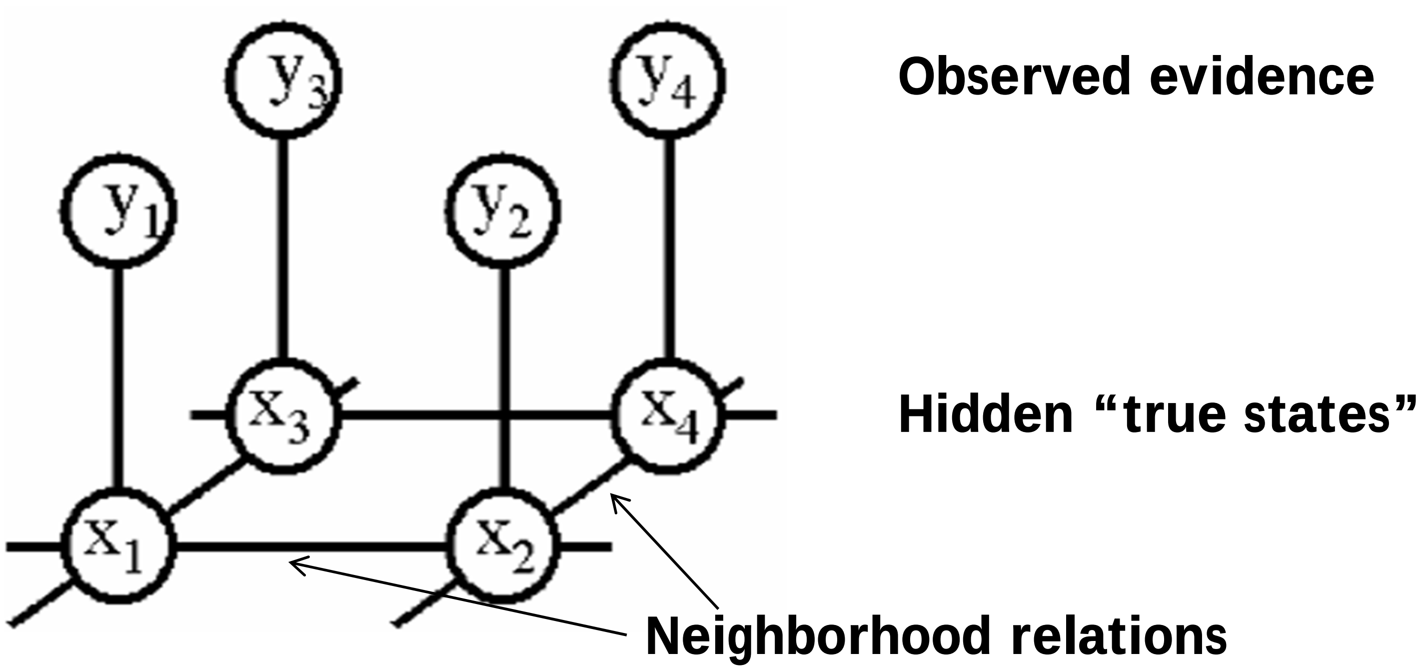
What we will learn today?

- Model free clustering
  - Mean-shift
- Graph theoretic segmentation
  - Normalized Cuts
  - Using texture features
- Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation (supp. materials)
  - s-t mincut algorithm (supp. materials)
  - Extension to non-binary case (supp. materials)
  - Applications



Markov Random Fields

- Allow rich probabilistic models for images
- But built in a local, modular way
  - Learn local effects, get global effects out



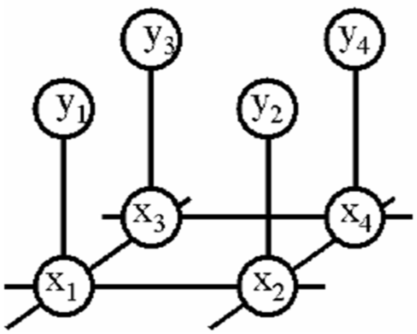
Slide credit: William Freeman

MRF Nodes as Pixels

Original image

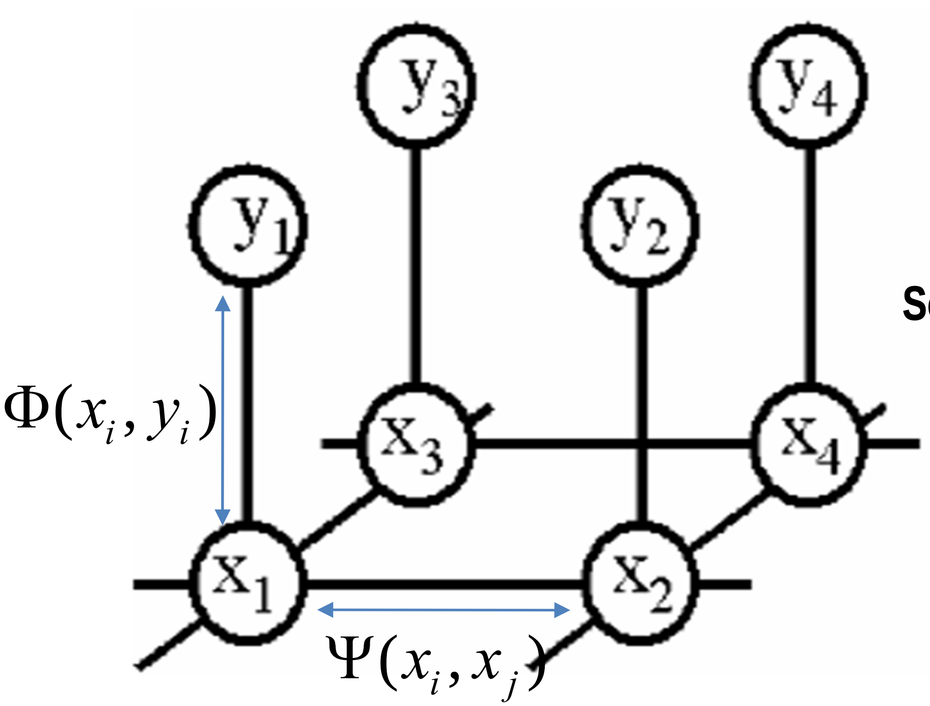
Degraded image

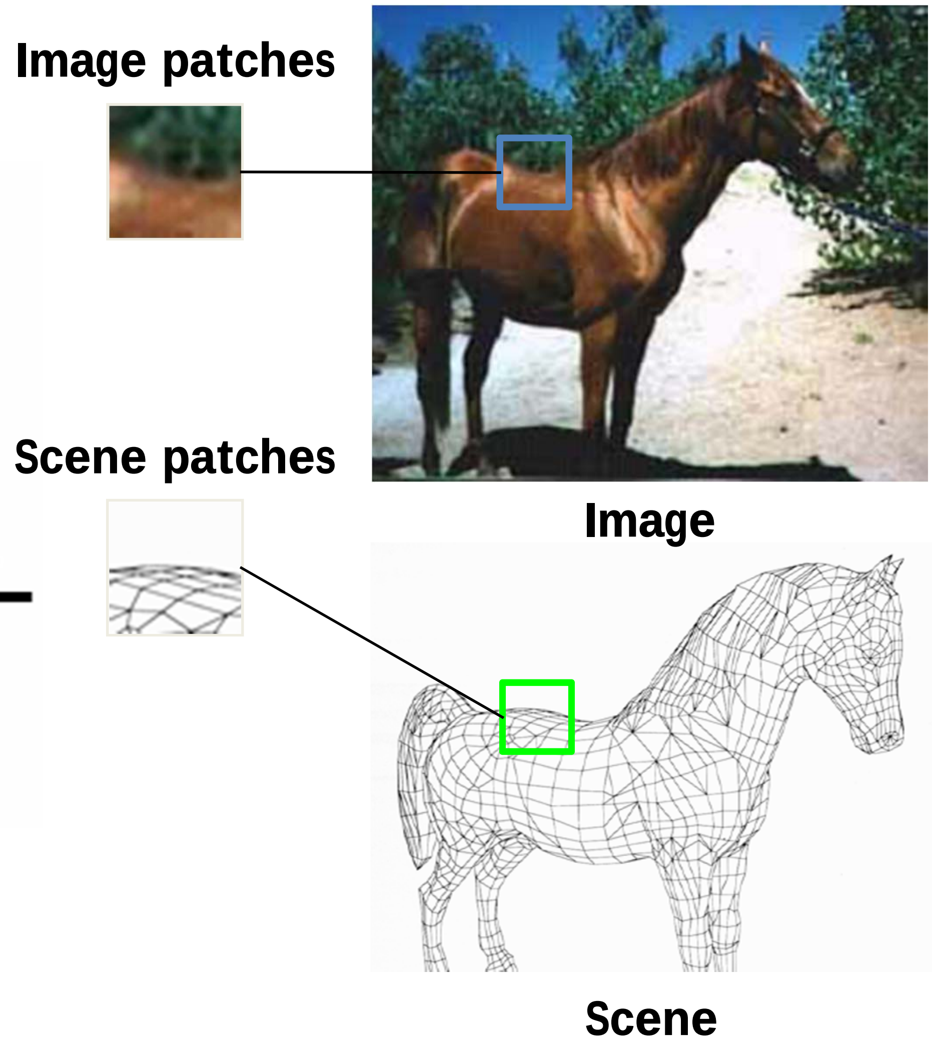
Reconstruction from MRF modeling pixel neighborhood statistics



Slide credit: Bastian Leibe

MRF Nodes as Patches





Slide credit: William Freeman

Network Joint Probability

$$P(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(x_i, x_j)$$

Scene

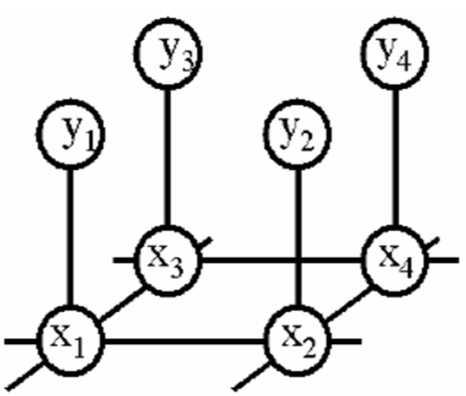
Image

Image-scene compatibility function

Local observations

Scene-scene compatibility function

Neighboring scene nodes



Slide credit: William Freeman

Energy Formulation

- Joint probability

$$P(x,y)=\prod_i \Phi(x_i,y_i)\prod_{i,j}\Psi(x_i,x_j)$$

- Taking the log p(.) turns this into an Energy optimization problem

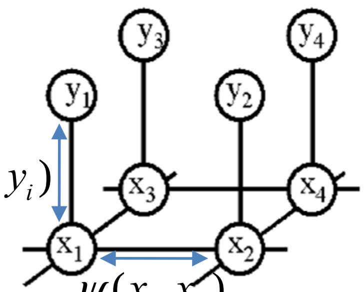
$$\log P(x,y)=\sum_i \log\ \Phi(x_i,y_i)+\sum_{i,j}\log\ \Psi(x_i,x_j)$$

$$E(x,y)=\sum_i \varphi(x_i,y_i)\qquad+\sum_{i,j}\psi(x_i,x_j)$$

- This is similar to free-energy problems in statistical mechanics (spin glass theory). We therefore draw the analogy and call E an *energy function*.
- φ and ψ are called *potentials*.

Slide credit: Bastian Leibe

Energy Formulation



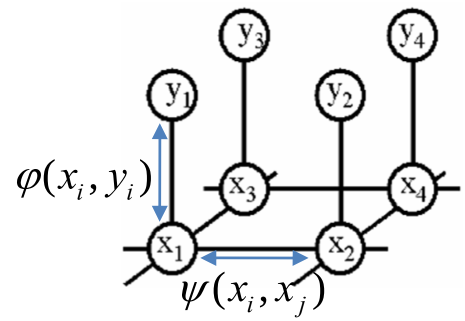
- Energy function

$$E(x,y) = \sum_i \underbrace{\varphi(x_i, y_i)}_{\text{Single-node potentials}} + \sum_{i,j} \underbrace{\psi(x_i, x_j)}_{\text{Pairwise potentials}}$$

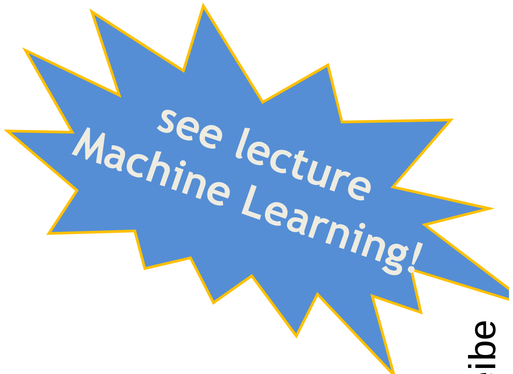
- Single-node potentials φ
  - Encode local information about the given pixel/patch
  - How likely is a pixel/patch to belong to a certain class (e.g. foreground/background)?
- Pairwise potentials ψ
  - Encode neighborhood information
  - How different is a pixel/patch's label from that of its neighbor? (e.g. based on intensity/color/texture difference, edges)

Slide credit: Bastian Leibe

Energy Minimization



- Goal:
  - Infer the optimal labeling of the MRF.
- Many inference algorithms are available, e.g.
  - Gibbs sampling, simulated annealing
  - Iterated conditional modes (ICM)
  - Variational methods
  - Belief propagation



- Recently, Graph Cuts have become a popular tool
  - Only suitable for a certain class of energy functions
  - But the solution can be obtained very fast for typical vision problems (~1MPixel/sec).

Slide credit: Bastian Leibe

What we will learn today?

- Graph theoretic segmentation
  - Normalized Cuts
  - Using texture features
  - Extension: Multi-level segmentation
- Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Extension to non-binary case
  - Applications

GrabCut: live demo

Reported results

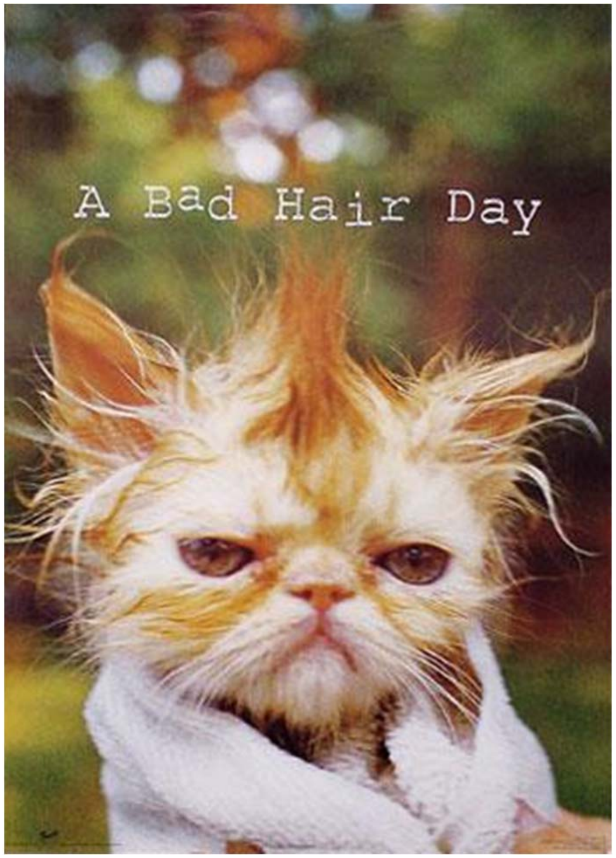
- *Included in MS Office 2010 (let's try it)*

GrabCut: live demo

Reported results

- *Included in MS Office 2010 (let's try it)*

GrabCut: live demo



- *Included in MS Office 2010 (let's try it)*

GraphCut Image Synthesis Results

Source: Vivek Kwatra

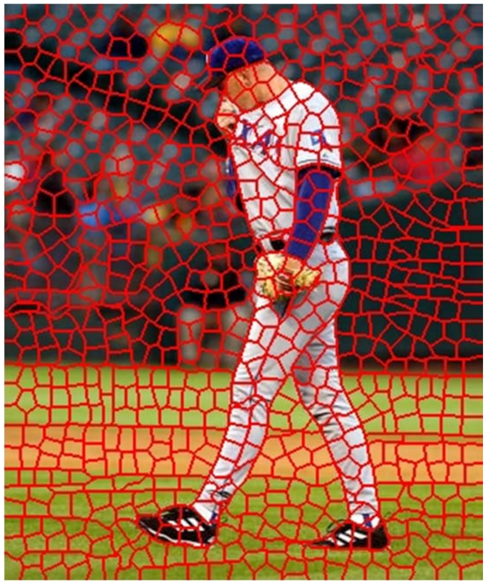
Application: Texture Synthesis in the Media

- Currently, still done manually…

Slide credit: Kristen Grauman

Improving Efficiency of Segmentation

- Problem: Images contain many pixels
  - Even with efficient graph cuts, an MRF formulation has too many nodes for interactive results.
- Efficiency trick: Superpixels
  - Group together similar-looking pixels for efficiency of further processing.
  - Cheap, local oversegmentation
  - Important to ensure that superpixels do not cross boundaries
- Several different approaches possible
  - Superpixel code available here
  - http://www.cs.sfu.ca/~mori/research/superpixels/



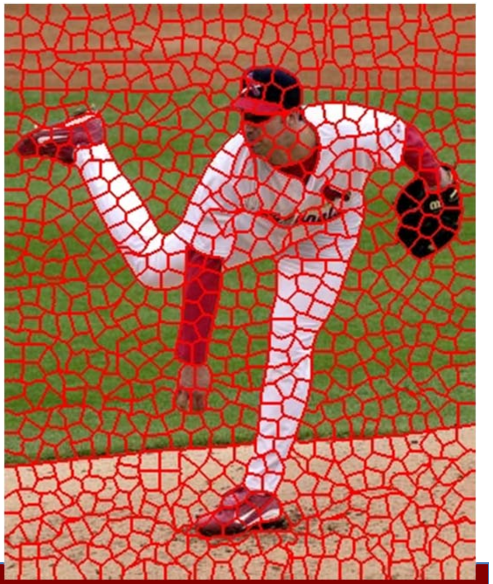
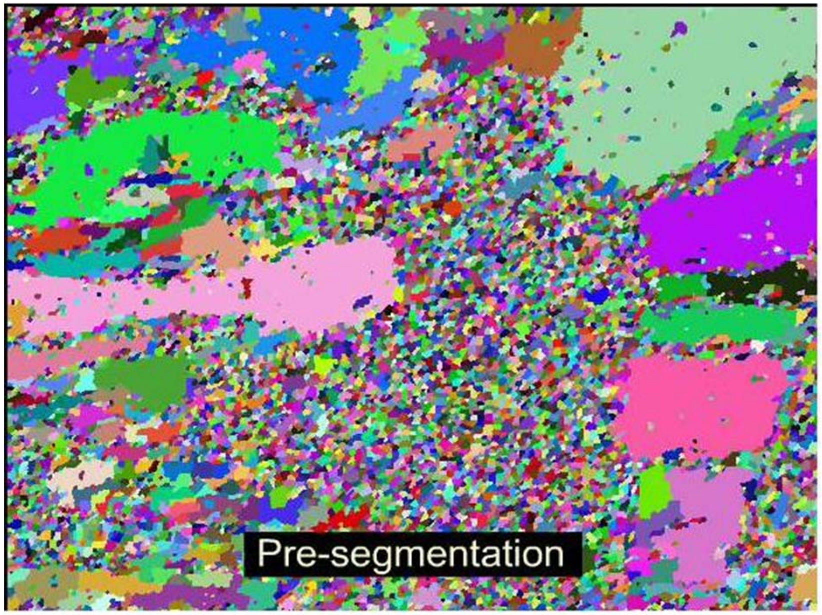
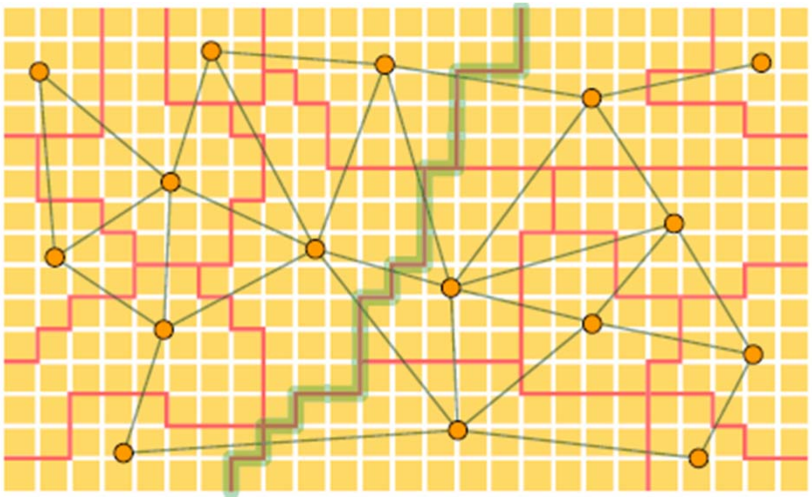


Image source: Greg Mori

Superpixels for Pre-Segmentation





Graph structure

Image	Dimension	Nodes Ratio	Edges Ratio	Lag with Pre-segmentation	Lag without Pre-segmentation
Boy	$(408, 600)$	10.7	16.8	$0.12s$	$0.57s$
Ballet	$(440, 800)$	11.4	18.3	$0.21s$	$1.39s$
Twins	$(1024, 768)$	20.7	32.5	$0.25s$	$1.82s$
Girl	$(768, 1147)$	23.8	37.6	$0.22s$	$2.49s$
Grandpa	$(1147, 768)$	19.3	30.5	$0.22s$	$3.56s$

Speedup

Summary: Graph Cuts Segmentation

- Pros
  - Powerful technique, based on probabilistic model (MRF).
  - Applicable for a wide range of problems.
  - Very efficient algorithms available for vision problems.
  - Becoming a de-facto standard for many segmentation tasks.
- Cons/Issues
  - Graph cuts can only solve a limited class of models
    - Submodular energy functions
    - Can capture only part of the expressiveness of MRFs
  - Only approximate algorithms available for multi-label case

What we have learned today

- Model free clustering
  - Mean-shift
- Graph theoretic segmentation
  - Normalized Cuts
  - Using texture features

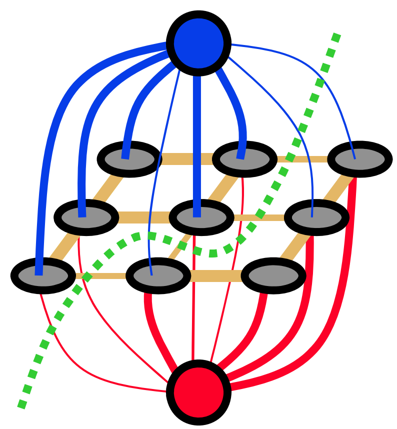
(Midterm materials)

- Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation (supp. materials)
  - s-t mincut algorithm (supp. materials)
  - Extension to non-binary case (supp. materials)
  - Applications

Supplementary materials

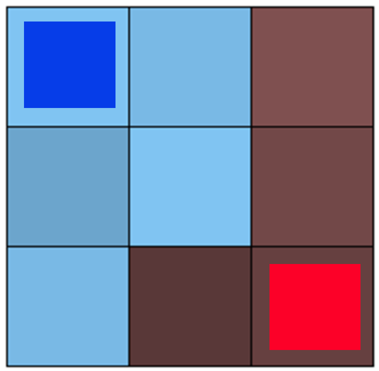
Supplementary materials

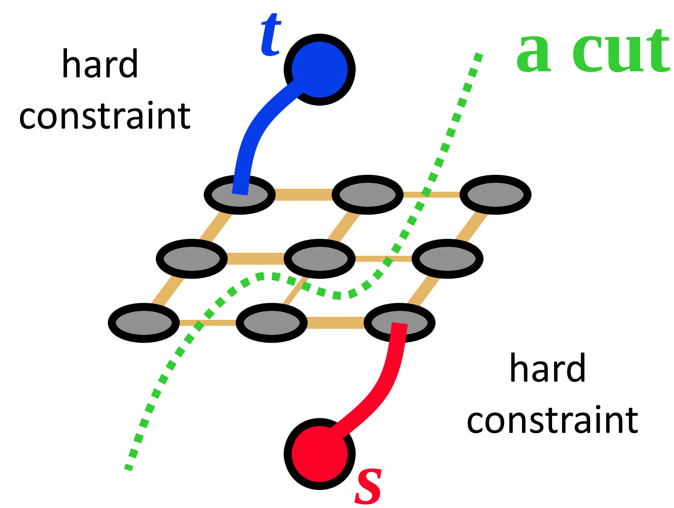
- Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation (supp. materials)
  - s-t mincut algorithm (supp. materials)
  - Extension to non-binary case (supp. materials)
  - Applications



Graph Cuts for Optimal Boundary Detection

- Idea: convert MRF into source-sink graph

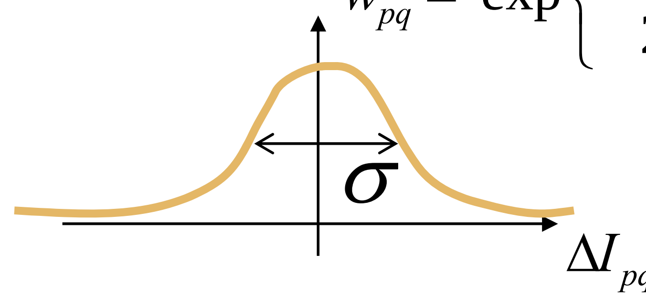




Minimum cost cut can be computed in polynomial time

(max-flow/min-cut algorithms)

$$w_{pq} = \exp\left\{-\frac{\Delta I_{pq}}{2\sigma^2}\right\}$$

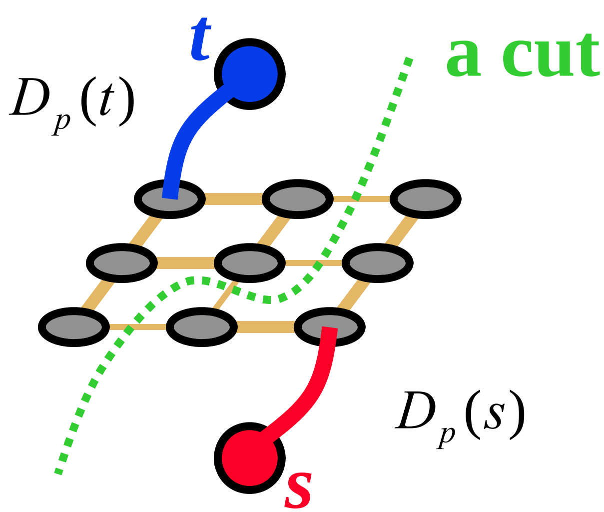


Slide credit: Yuri Boykov

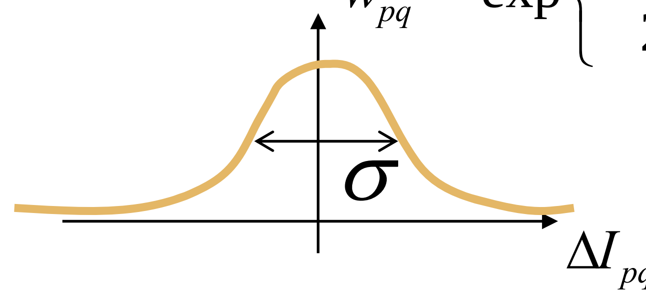
Simple Example of Energy

$$E(L) = \underbrace{\sum_p D_p(L_p)}_{\text{t-links}} + \underbrace{\sum_{pq \in N} w_{pq} \cdot \delta(L_p \neq L_q)}_{\text{n-links}}$$

Regional term (t-links) **Boundary term** (n-links)



$$w_{pq} = \exp\left\{-\frac{\Delta I_{pq}}{2\sigma^2}\right\}$$

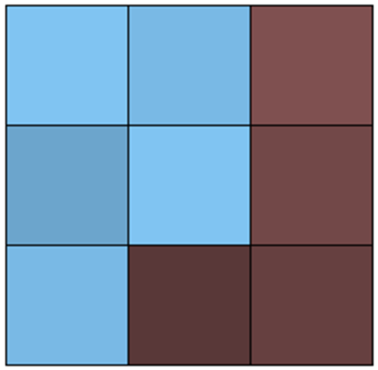


$$L_p \in \{s, t\}$$

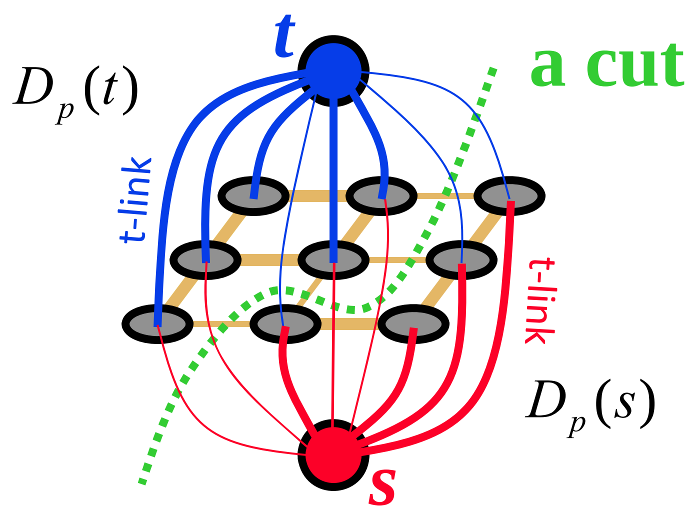
(binary object segmentation)

Slide credit: Yuri Boykov

Adding Regional Properties



Regional bias example



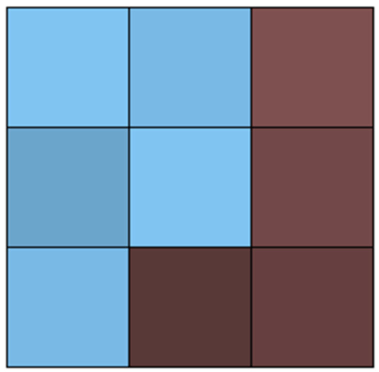
Suppose I^s and I^t are given "expected" intensities of object and background

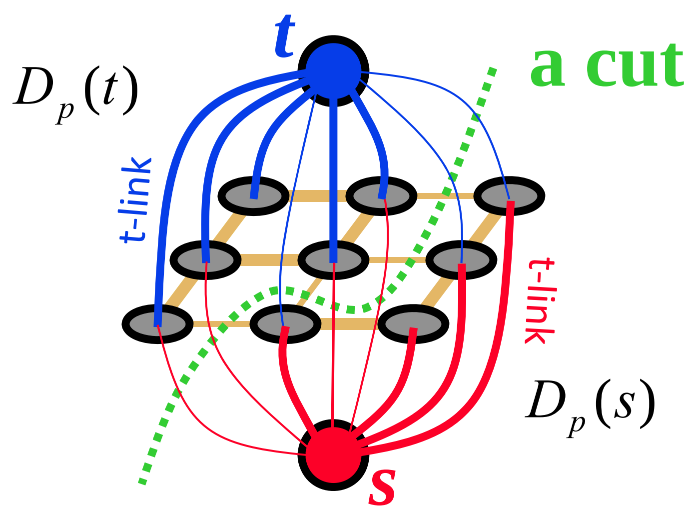
$$D_p(s) \propto \exp\left(-\| I_p - I^s \|^2 / 2\sigma^2\right)$$
$$D_p(t) \propto \exp\left(-\| I_p - I^t \|^2 / 2\sigma^2\right)$$

NOTE: hard constrains are not required, in general.

Slide credit: Yuri Boykov

Adding Regional Properties





"expected" intensities of
object and background
I^s and I^t
can be re-estimated

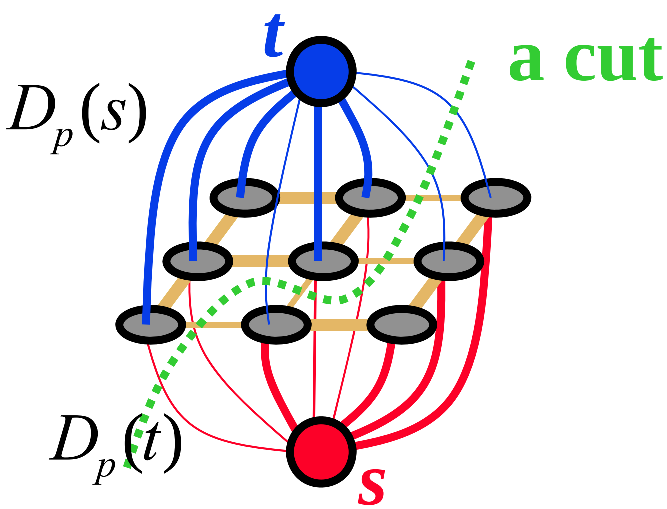
$$D_p(s) \propto \exp\left(-\| I_p - I^s \|^2 / 2\sigma^2\right)$$
$$D_p(t) \propto \exp\left(-\| I_p - I^t \|^2 / 2\sigma^2\right)$$

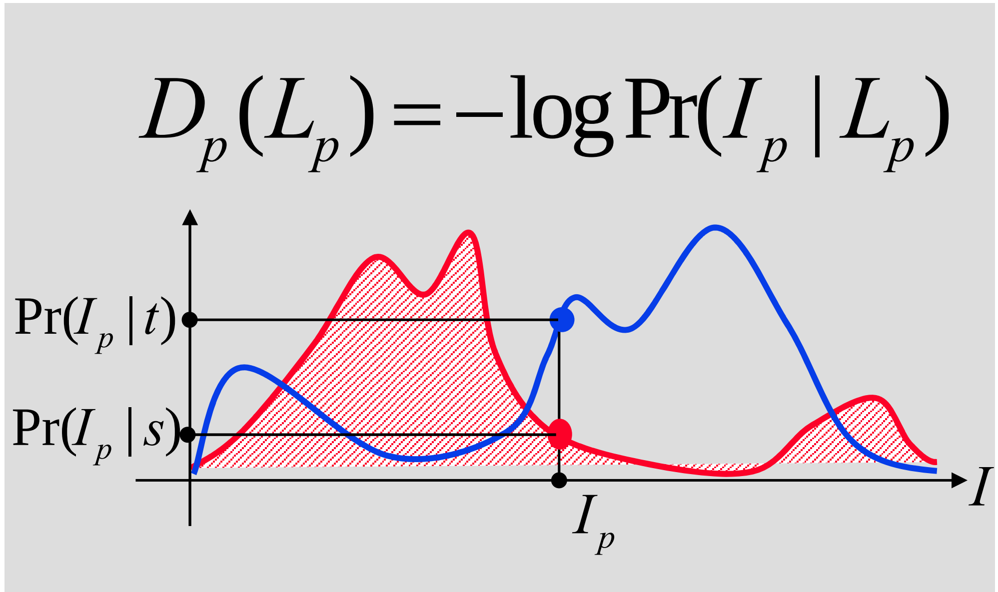
EM-style optimization

Slide credit: Yuri Boykov

Adding Regional Properties

- More generally, regional bias can be based on any intensity models of object and background





given object and background intensity histograms

Slide credit: Yuri Boykov

How to Set the Potentials? Some Examples

- Color potentials
  - e.g. modeled with a Mixture of Gaussians

$$\pi(x_i, y_i; \theta_\pi) = \log \sum_k \theta_\pi(x_i, k) P(k \mid x_i) N(y_i; \bar{y}_k, \Sigma_k)$$

- Edge potentials
  - e.g. a “contrast sensitive Potts model”

$$\phi(x_i, x_j, g_{ij}(y); \theta_\phi) = -\theta_\phi^T g_{ij}(y) \delta(x_i \neq x_j)$$

where

$$g_{ij}(y) = e^{-\beta \left\| y_i - y_j \right\|^2} \qquad \beta = 2 \cdot avg\left(\left\| y_i - y_j \right\|^2 \right)$$

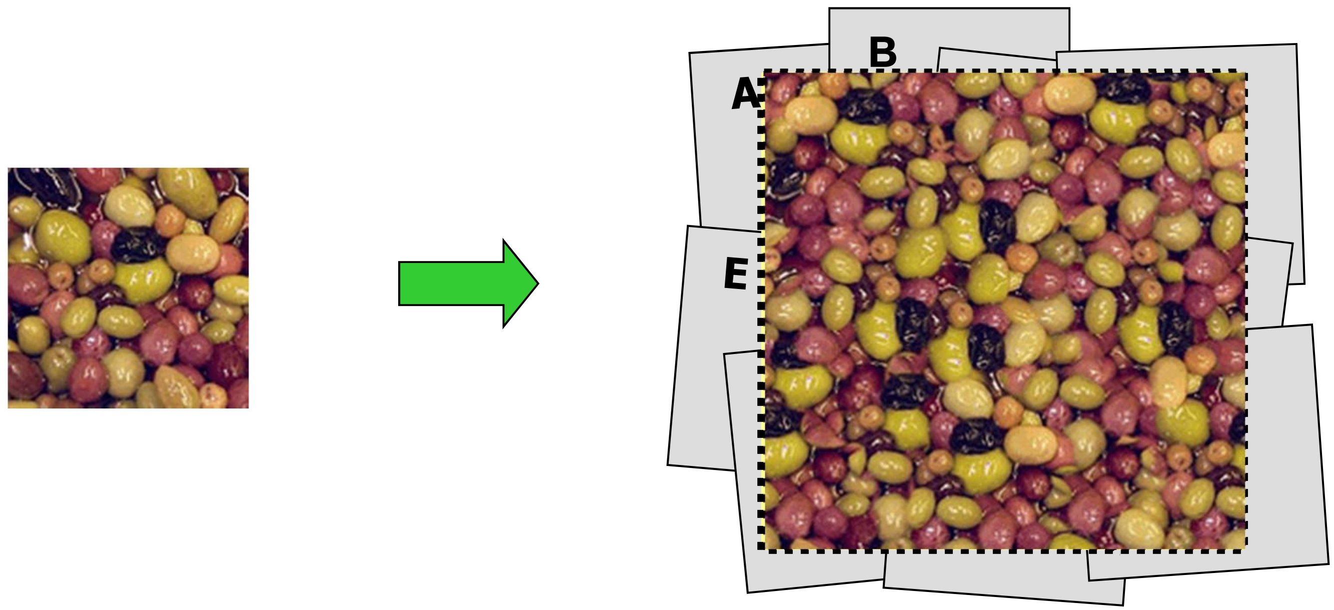
- Parameters θ_π, θ_ϕ need to be learned, too!

[Shotton & Winn, ECCV’06]

Other Applications: Texture Synthesis

Graph-cut textures

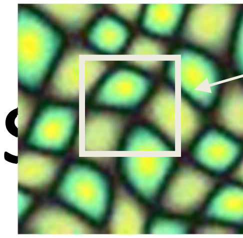
(Kwatra, Schodl, Essa, Bobick 2003)



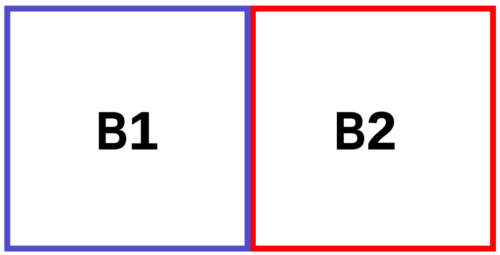
Similar to "image-quilting" (Efros & Freeman, 2001)

Slide credit: Yuri Boykov

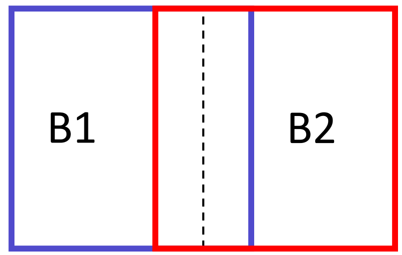
Bas ea



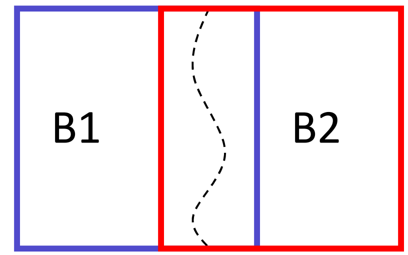
Input texture



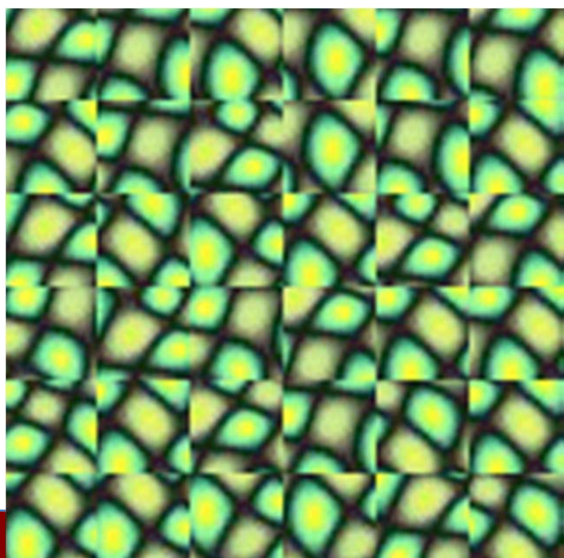
Random placement
of blocks

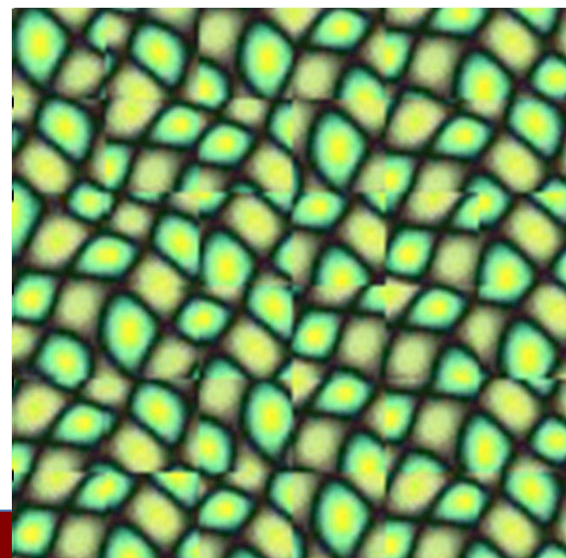


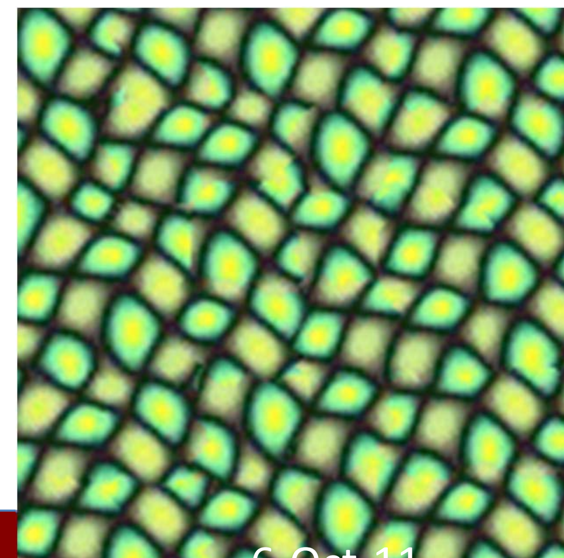
Neighboring blocks
constrained by overlap



Minimal error
boundary cut



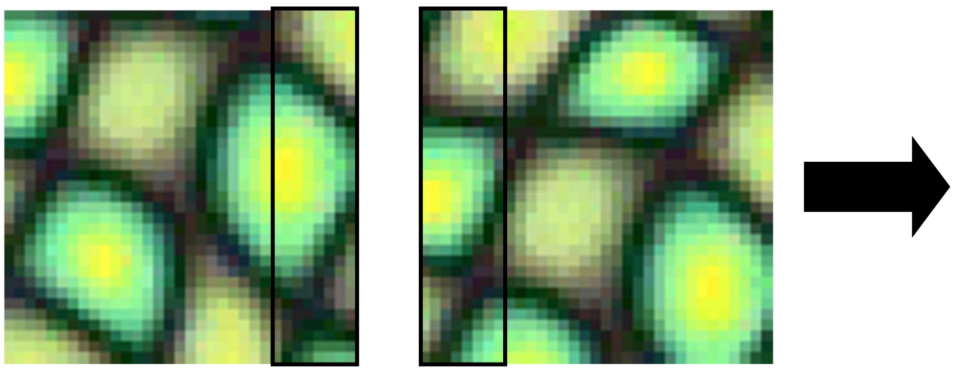




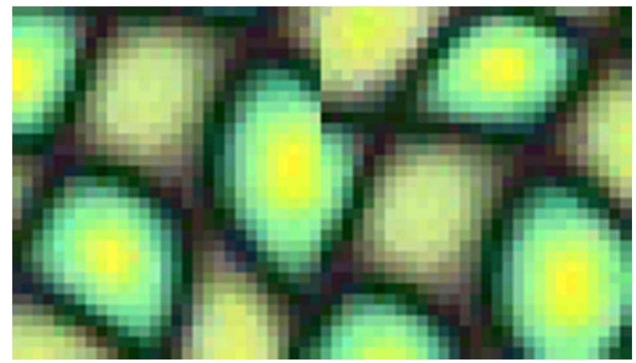
Slide from Alyosha Efros

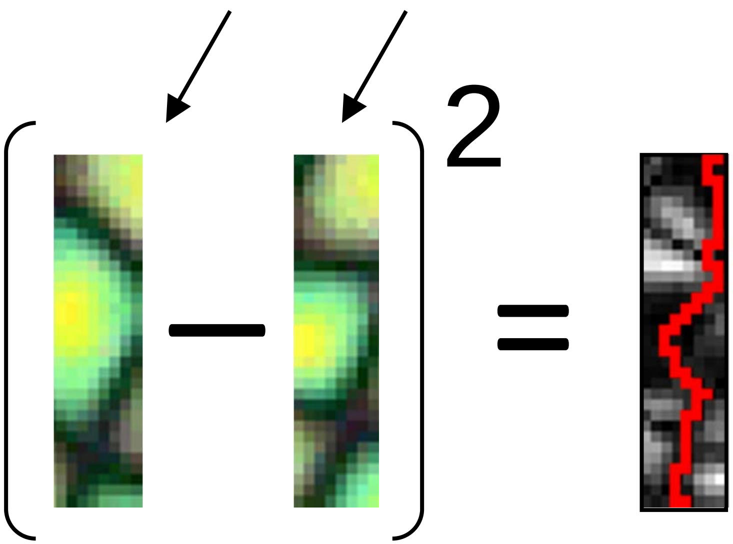
Minimal Error Boundary

Overlapping blocks

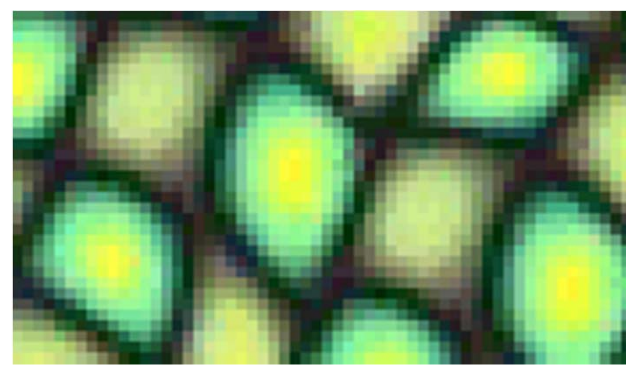


Vertical boundary





Overlap error



min. error boundary

Slide from Alyosha Efros

GraphCut Texture Synthesis Results

Original fragments

Results (with perspective correction)

Source: Vivek Kwatra

Application: Texture Synthesis in the Media

- Currently, still done manually…

Slide credit: Kristen Grauman

How Do we Know?

http://www.dailykos.com/story/2004/10/27/22442/878

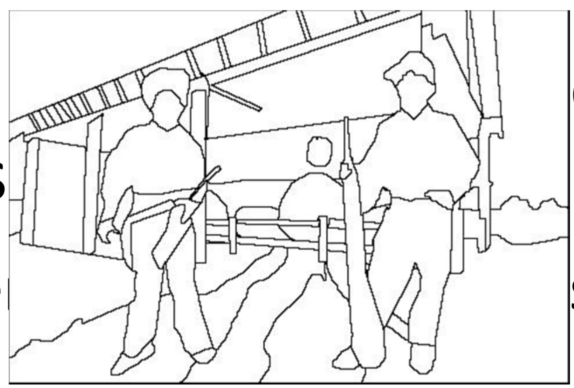
Another Example

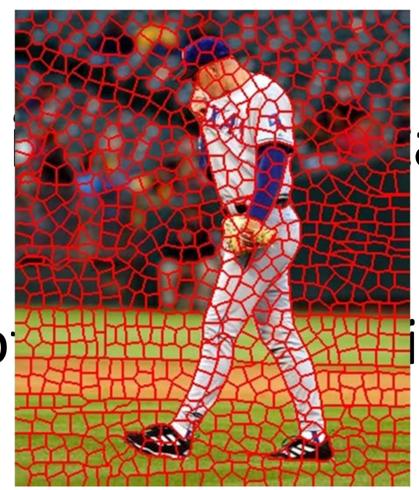
http://thelede.blogs.nytimes.com/2008/07/10/in-an-iranian-image-a-missile-too-many/

Slide credit: Kristen Grauman

Segmentation: Caveats

- We’ve looked at *bottom-up* ways to segment an image into regions, yet finding meaningful segments is intertwined with the recognition problem.
- Often want to avoid making hard decisions too soon
- Difficu en ation succes
  - Ofte st o ipeline.

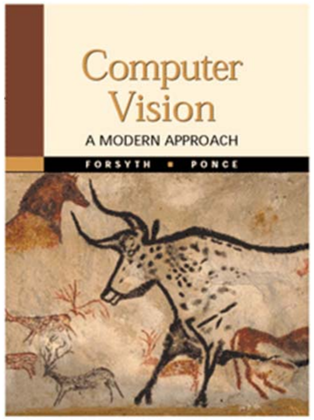




Slide credit: Kristen Grauman

References and Further Reading

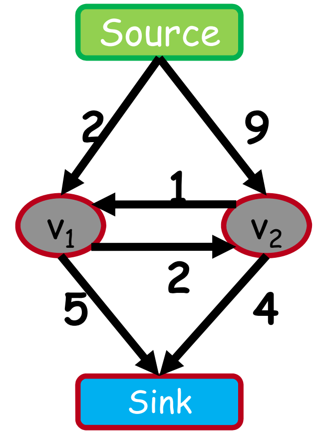
- Background information on Normalized Cuts can be found in Chapter 14 of
  – D. Forsyth, J. Ponce, *Computer Vision – A Modern Approach*. Prentice Hall, 2003

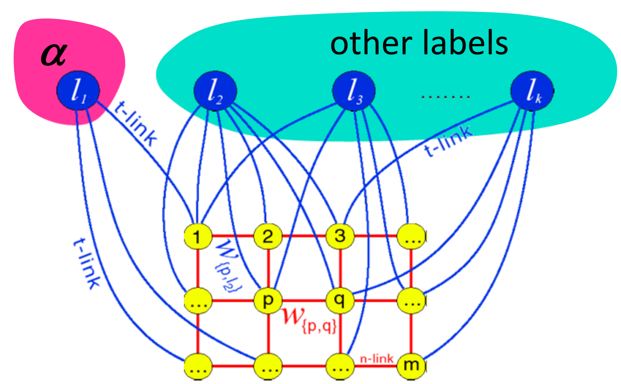


- Try the NCuts Matlab code at
  – http://www.cis.upenn.edu/~jshi/software/
- Try the GraphCut implementation at
  http://www.adastral.ucl.ac.uk/~vladkolm/software.html

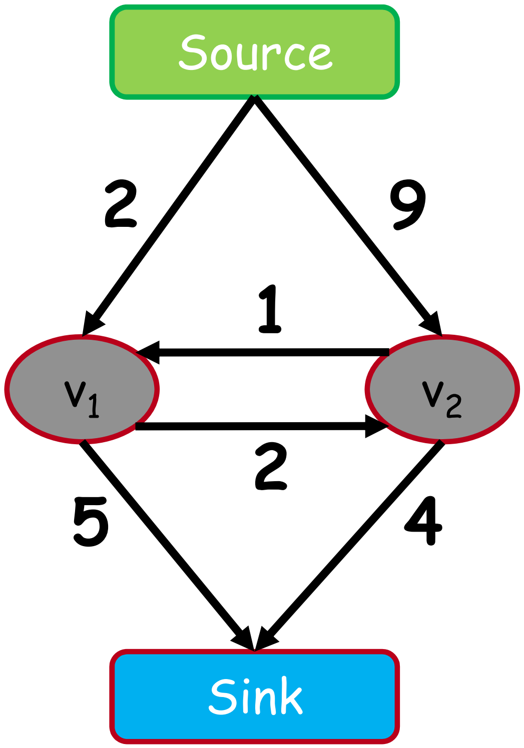
Supplementary materials

- Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Extension to non-binary case
  - Applications





How Does it Work? The s-t-Mincut Problem



Graph (V, E, C)

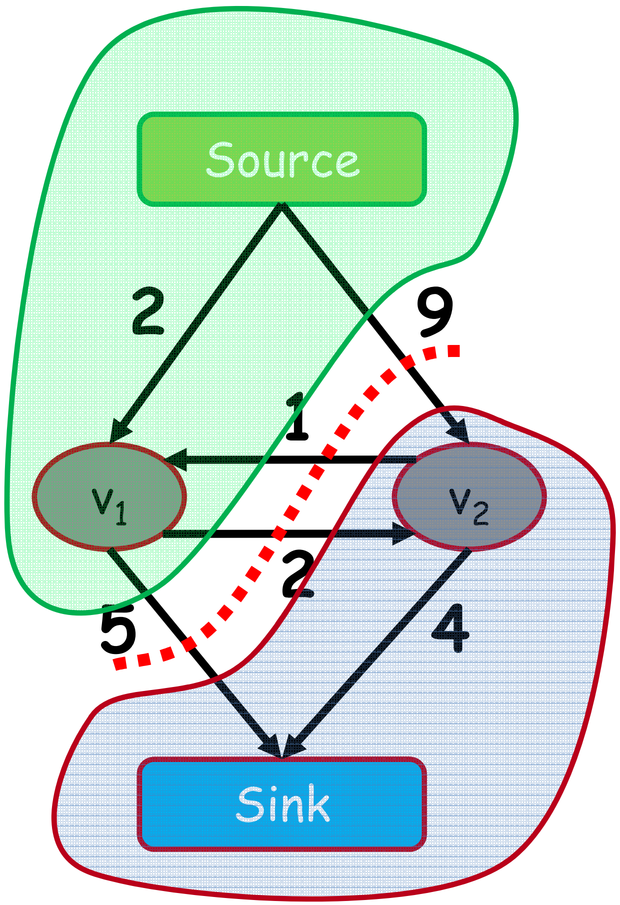
Vertices $V = \{v_1, v_2 \dots v_n\}$

Edges $E = \{(v_1, v_2) \dots\}$

Costs $C = \{c_{(1, 2)} \dots\}$

Slide credit: Pushmeet Kohli

The s-t-Mincut Problem



5 + 2 + 9 = 16

What is an st-cut?

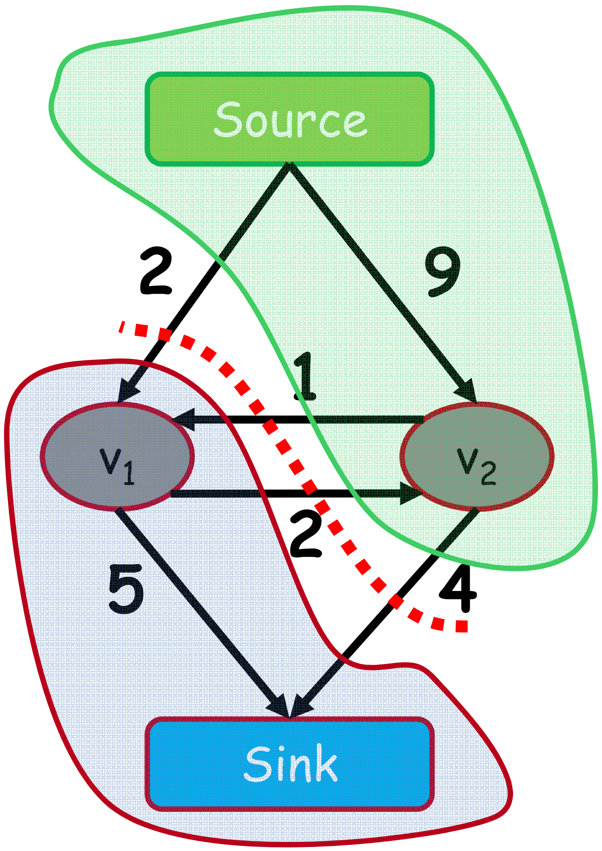
An st-cut (S,T) divides the nodes between source and sink.

What is the cost of a st-cut?

Sum of cost of all edges going from S to T

Slide credit: Pushmeet Kohli

The s-t-Mincut Problem



2 + 1 + 4 = 7

What is an st-cut?

An st-cut (S,T) divides the nodes between source and sink.

What is the cost of a st-cut?

Sum of cost of all edges going from S to T

What is the st-mincut?

st-cut with the minimum cost

Slide credit: Pushmeet Kohli

History of Maxflow Algorithms

Augmenting Path and Push-Relabel

year	discoverer(s)	bound
1951	Dantzig	$O(n^2mU)$
1955	Ford & Fulkerson	$O(m^2U)$
1970	Dinitz	$O(n^2m)$
1972	Edmonds & Karp	$O(m^2 \log U)$
1973	Dinitz	$O(nm \log U)$
1974	Karzanov	$O(n^3)$
1977	Cherkassky	$O(n^2m^{1/2})$
1980	Galil & Naamad	$O(nm \log^2 n)$
1983	Sleator & Tarjan	$O(nm \log n)$
1986	Goldberg & Tarjan	$O(nm \log(n^2/m))$
1987	Ahuja & Orlin	$O(nm + n^2 \log U)$
1987	Ahuja et al.	$O(nm \log(n\sqrt{\log U}/m))$
1989	Cheriyan & Hagerup	$E(nm + n^2 \log^2 n)$
1990	Cheriyan et al.	$O(n^3/\log n)$
1990	Alon	$O(nm + n^{8/3} \log n)$
1992	King et al.	$O(nm + n^{2+\epsilon})$
1993	Phillips & Westbrook	$O(nm(\log_{m/n} n + \log^{2+\epsilon} n))$
1994	King et al.	$O(nm \log_{m/(n \log n)} n)$
1997	Goldberg & Rao	$O(m^{3/2} \log(n^2/m) \log U)$ $O(n^{2/3}m \log(n^2/m) \log U)$

n: **#nodes**

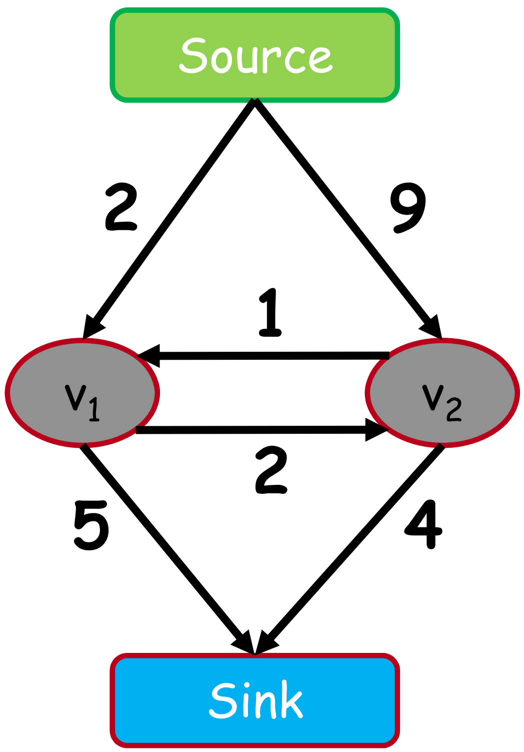
m: **#edges**

U: **maximum edge weight**

Algorithms assume non-negative edge weights

Slide credit: Andrew Goldberg

How to Compute the s-t-Mincut?



Solve the dual maximum flow problem

Compute the maximum flow between Source and Sink

Constraints

Edges: Flow < Capacity

Nodes: Flow in = Flow out

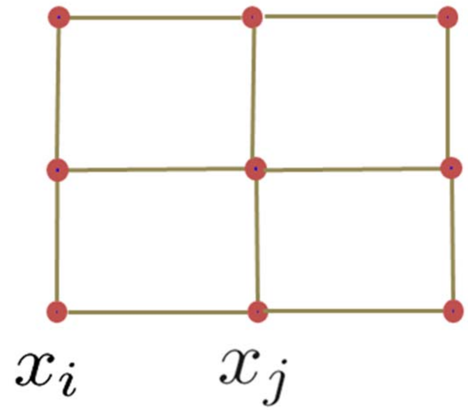
Min-cut/Max-flow Theorem

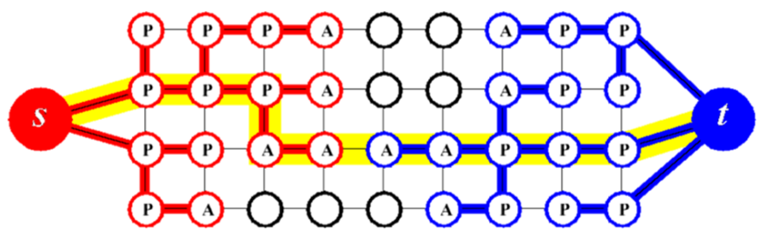
In every network, the maximum flow equals the cost of the st-mincut

Slide credit: Pushmeet Kohli

Maxflow in Computer Vision

- Specialized algorithms for vision problems
  - Grid graphs
  - Low connectivity (m ~ O(n))
- Dual search tree augmenting path algorithm
  [Boykov and Kolmogorov PAMI 2004]
  - Finds approximate shortest augmenting paths efficiently
  - High worst-case time complexity
  - Empirically outperforms other algorithms on vision problems
  - Efficient code available on the web
    http://www.adastral.ucl.ac.uk/~vladkolm/software.html





Slide credit: Pushmeet Kohli

When Can s-t Graph Cuts Be Applied?

Regional term **Boundary term**

$$E(L) = \underbrace{\sum_{p} E_p(L_p)}_{\text{t-links}} + \underbrace{\sum_{pq \in N} E(L_p, L_q)}_{\text{n-links}} \qquad L_p \in \{s, t\}$$

- s-t graph cuts can only globally minimize binary energies that are submodular. **[Boros & Hummer, 2002, Kolmogorov & Zabih, 2004]**

E(L) can be minimized by *s-t* graph cuts $\Leftrightarrow$ $\boxed{E(s,s) + E(t,t) \leq E(s,t) + E(t,s)}$

Submodularity ("convexity")

- Non-submodular cases can still be addressed with some optimality guarantees.
  - Current research topic

Slide credit: Bastian Leibe

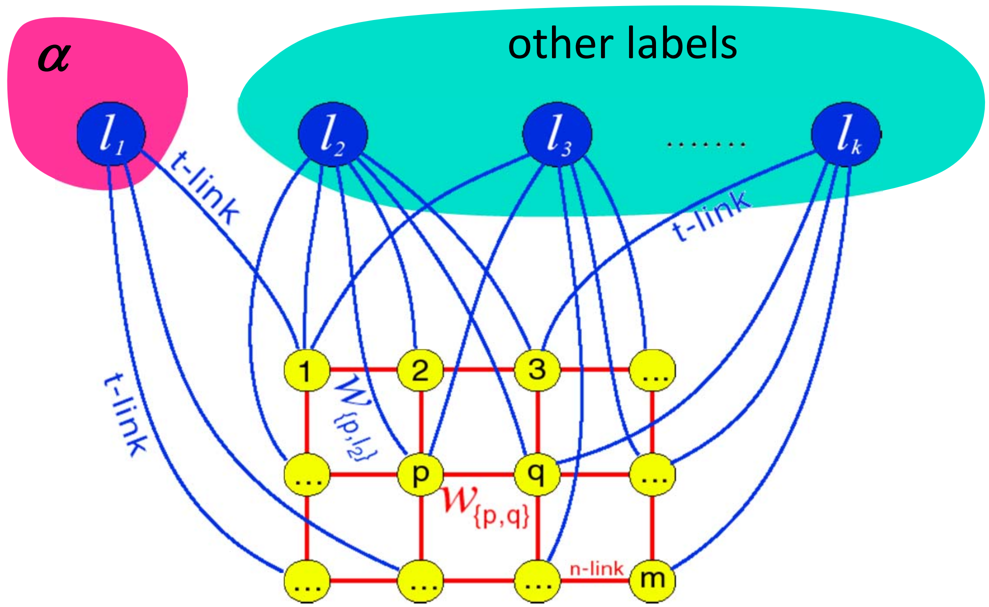
Dealing with Non-Binary Cases

- For image segmentation, the limitation to binary energies is a nuisance.

  $\Rightarrow$ Binary segmentation only
- We would like to solve also multi-label problems.
  – NP-hard problem with 3 or more labels
- There exist some approximation algorithms which extend graph cuts to the multi-label case
  – α-Expansion
  – $\alpha\beta$-Swap
- They are no longer guaranteed to return the globally optimal result.
  – But α-Expansion has a guaranteed approximation quality (2-approx) and converges in a few iterations.

α-Expansion Move

- Basic idea:
  - Break multi-way cut computation into a sequence of binary s-t cuts.



Slide credit: Yuri Boykov

α-Expansion Algorithm

1. Start with any initial solution
2. For each label "α" in any (e.g. random) order
    1. Compute optimal α-expansion move (s-t graph cuts)
    2. Decline the move if there is no energy decrease

- Stop when no expansion move would decrease energy

Slide credit: Yuri Boykov

α-Expansion Moves

- In each a-expansion a given label "α" grabs space from other labels

initial solution

-expansion

-expansion

-expansion

-expansion

-expansion

-expansion

-expansion

For each move we choose the expansion that gives the largest decrease in the energy: binary optimization problem

Slide credit: Yuri Boykov