Robotics Science and Systems: Computer Vision

Image segmentation

Chris Williams, Oct 2014

Many slides in this lecture are due to Vittorio Ferrari; other authors are credited on the bottom right

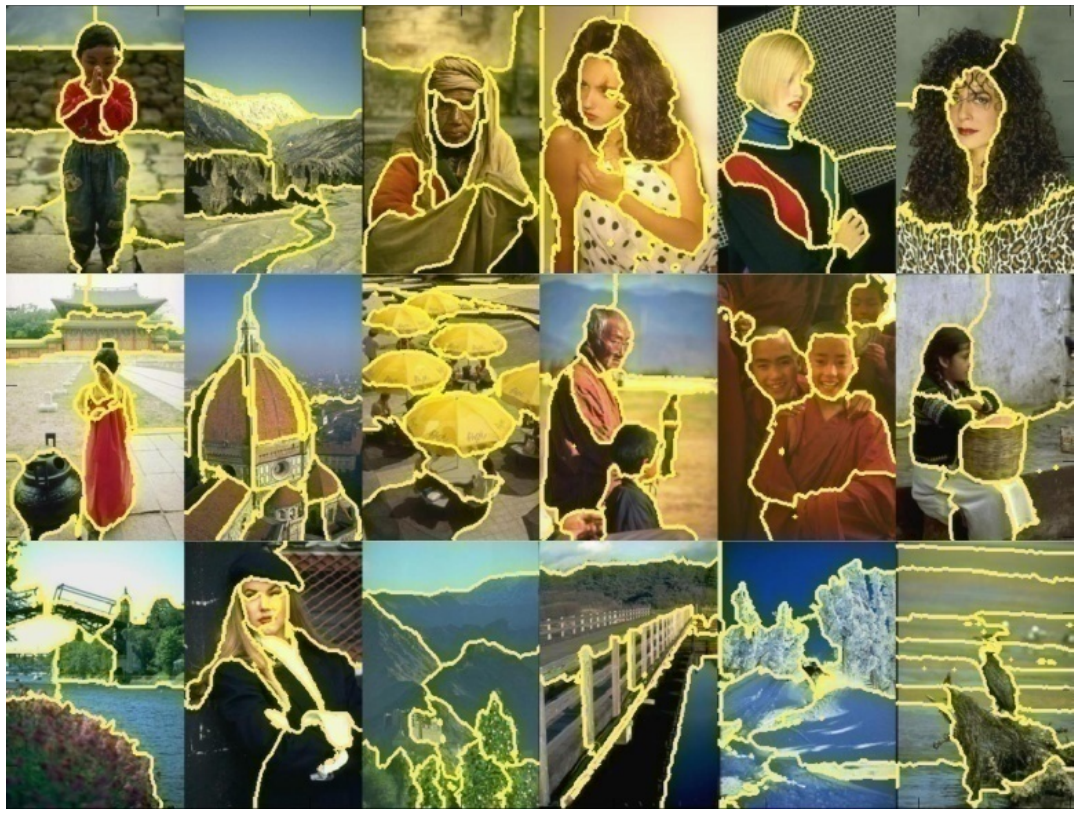
Topics of This Lecture

- Introduction
  - Gestalt principles
  - Image segmentation
- Segmentation as clustering
  - k-Means
  - Feature spaces
- Model-free clustering: Mean-Shift
- Interactive Segmentation with GraphCuts
- Reading: F+P chapter 9; Sz 5.3, 5.5

Examples of Grouping in Vision

What things should be grouped?

What cues indicate groups?



Determining image regions

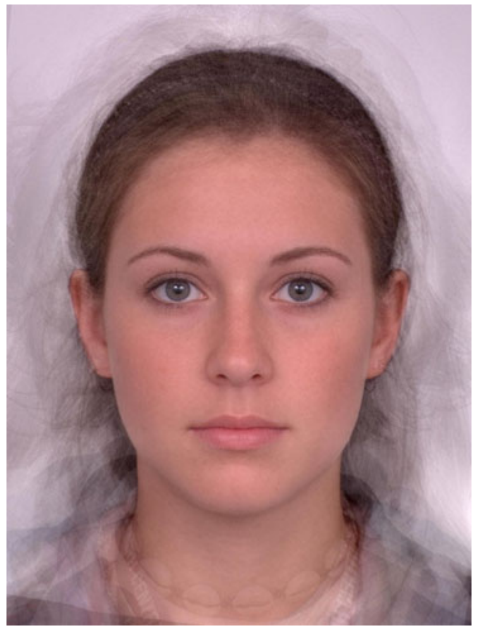
Slide modified from: Kristen Grauman

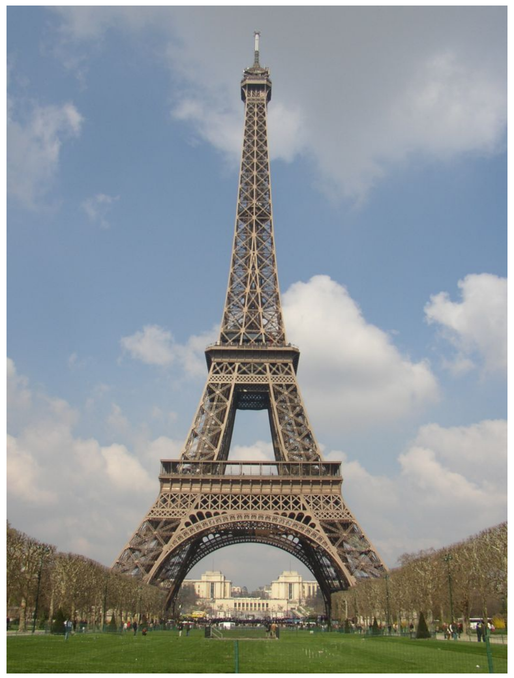
Similarity in appearance

Slide adapted from Kristen Grauman

http://chicagoist.com/attachments/chicagoist_alicia/GEESE.jpg, http://wwwdelivery.superstock.com/WI/223/1532/PreviewComp/SuperStock_1532R-0831.jpg

Symmetry





Slide credit: Kristen Grauman
http://seedmagazine.com/news/2006/10/beauty_is_in_the_processingtim.php

Common Fate

Image credit: Arthus-Bertrand (via F. Durand)

Slide credit: Kristen Grauman

Proximity

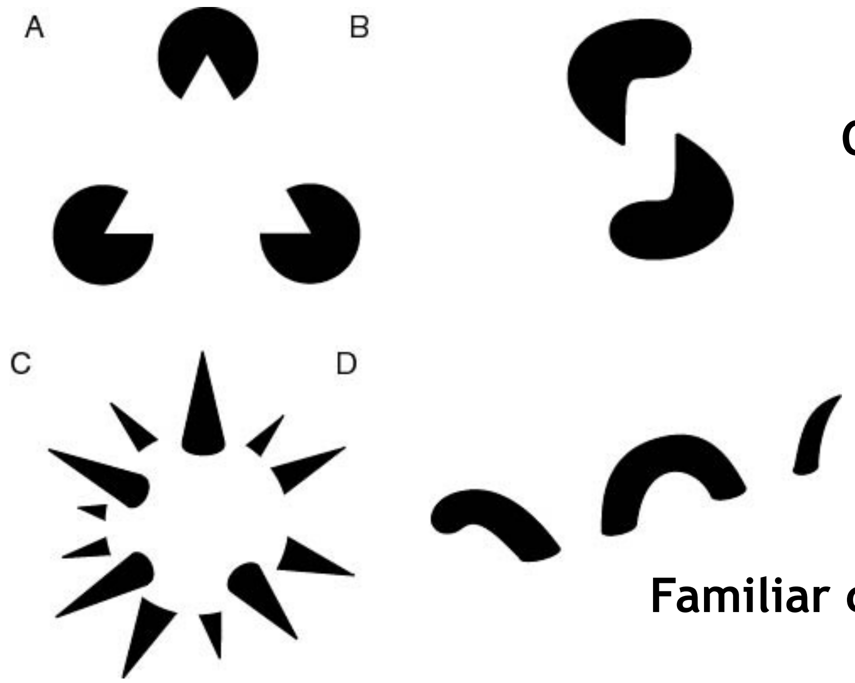
Slide credit: Kristen Grauman

http://www.capital.edu/Resources/Images/outside6_035.jpg

The Gestalt School

- Grouping is key to visual perception
- Elements in a collection can have properties that result from relationships
  - "The whole is other than than the sum of its parts"

Illusory/subjective contours



Occlusion

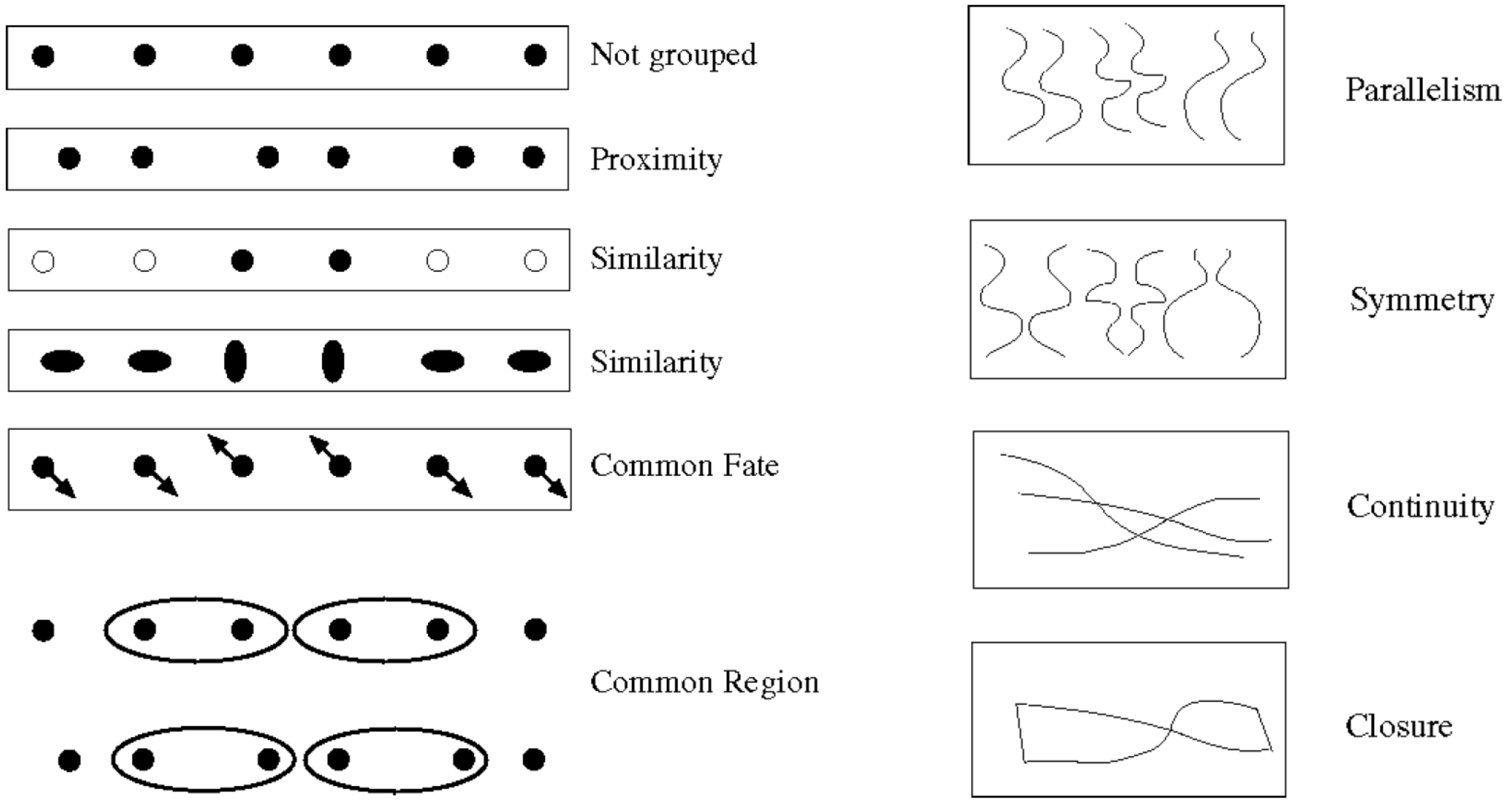
Familiar configuration

http://en.wikipedia.org/wiki/Gestalt_psychology

Slide credit: Svetlana Lazebnik

Image source: Steve Lehar

Gestalt Factors



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

Slide credit: B. Leibe

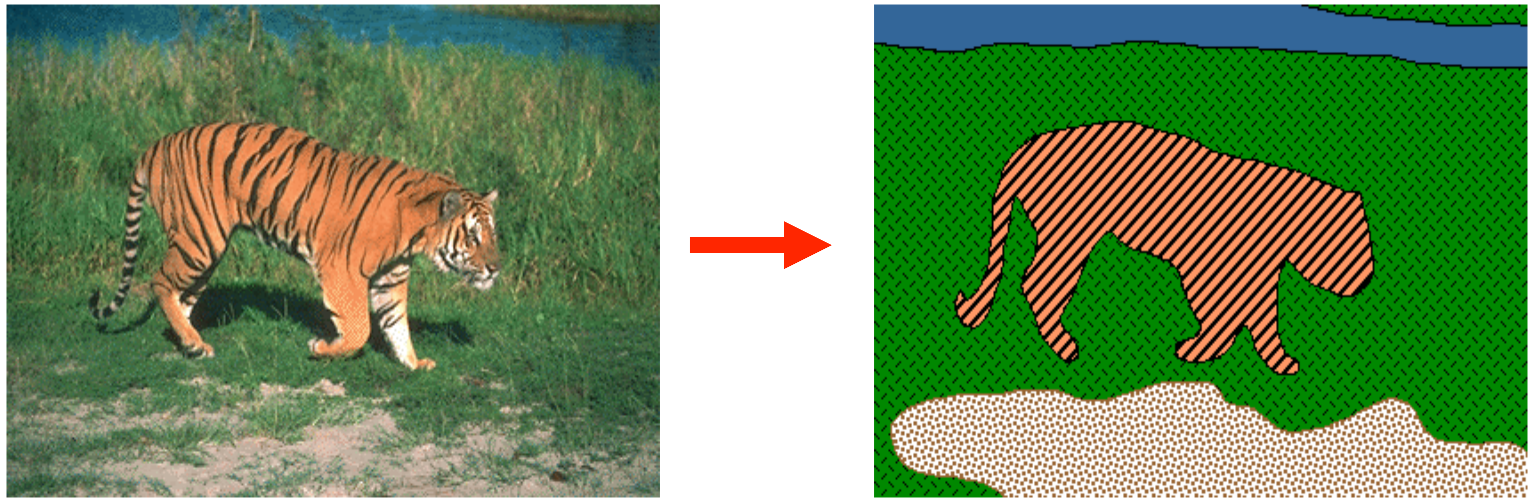
Image source: Forsyth & Ponce

The Ultimate Gestalt test

Slide adapted from B. Leibe

Image Segmentation

- **Goal: identify groups of pixels that go together**



Slide credit: Steve Seitz, Kristen Grauman

The Goals of Segmentation

- **Separate image into objects**

Image **Human segmentation**

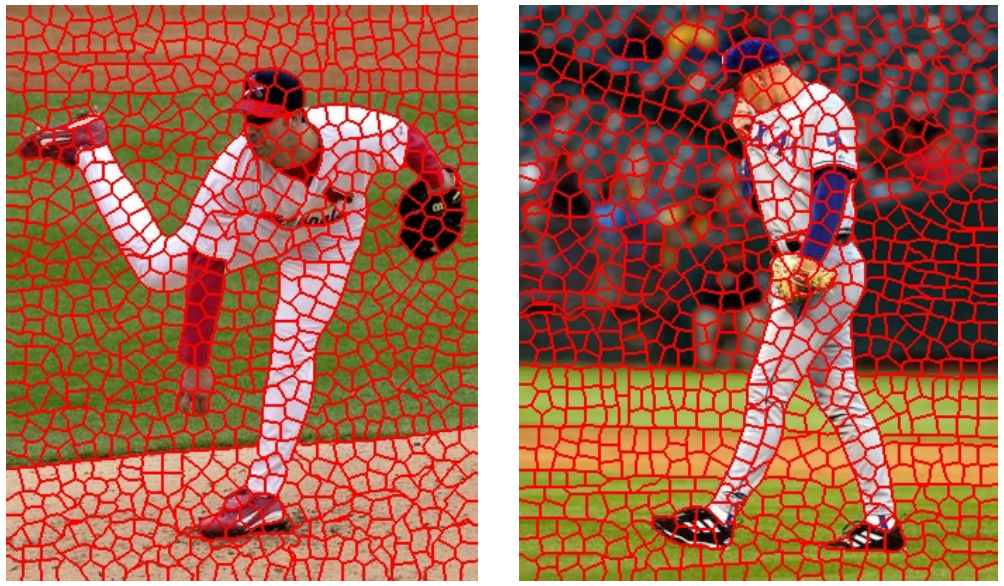
Slide credit: Svetlana Lazebnik

The Goals of Segmentation

- Separate image into objects

- Group together similar-looking pixels for efficiency of further processing

"superpixels"



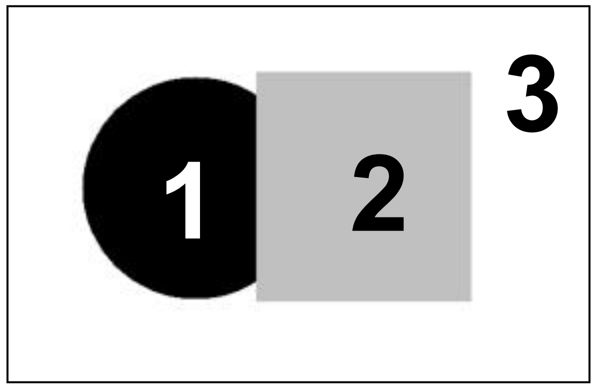
X. Ren and J. Malik. Learning a classification model for segmentation. ICCV 2003.

Slide credit: Svetlana Lazebnik

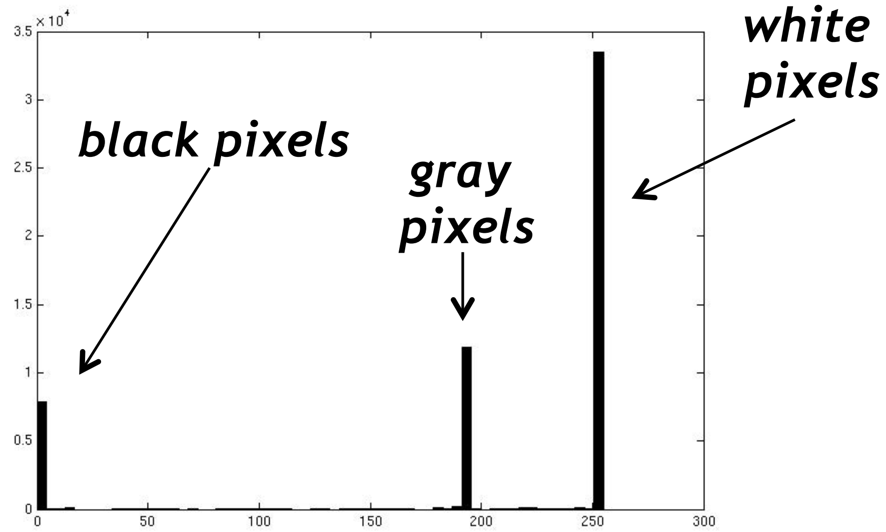
Topics of This Lecture

- Introduction
  - Gestalt principles
  - Image segmentation
- Segmentation as clustering
  - k-Means
  - Feature spaces
- Model-free clustering: Mean-Shift
- Interactive Segmentation with GraphCuts

Image Segmentation: Toy Example



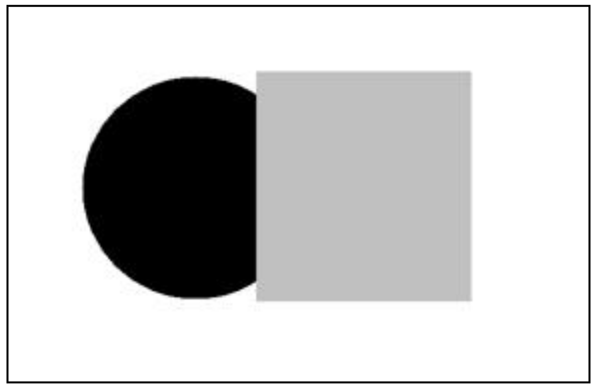
input image



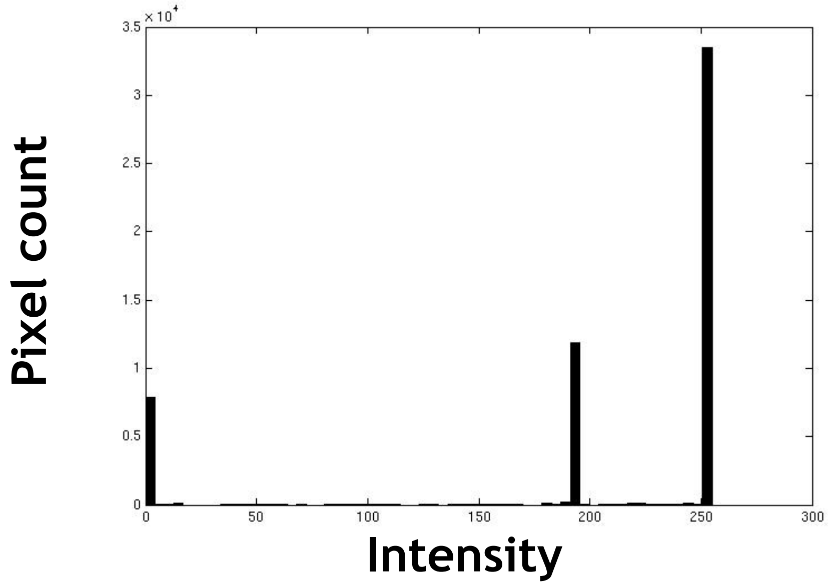
intensity

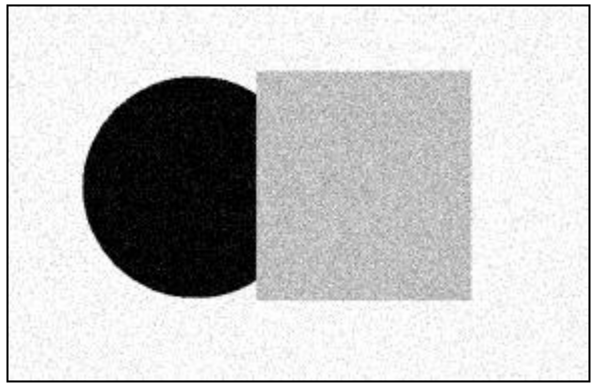
- These intensities define the three groups.
- We could label every pixel in the image according to which of these it is.
  - i.e. segment the image based on the intensity feature.
- What if the image isn't quite so simple?

Slide credit: Kristen Grauman

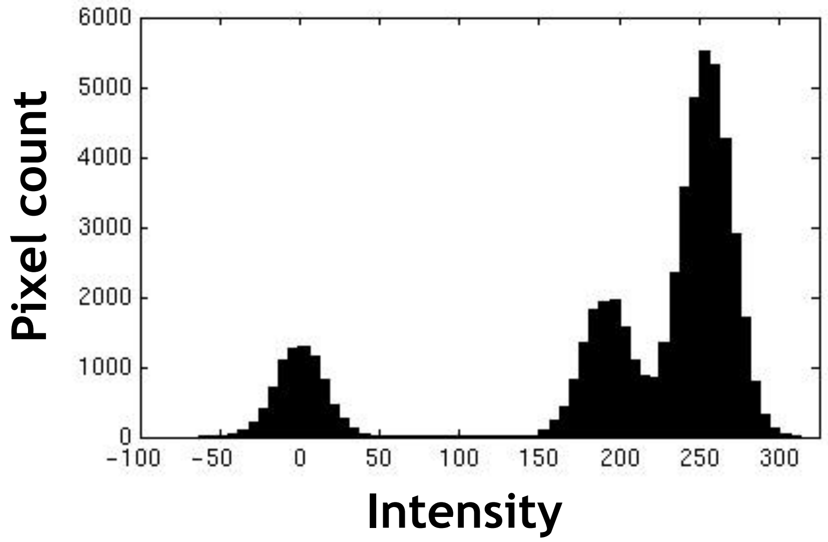


Input image

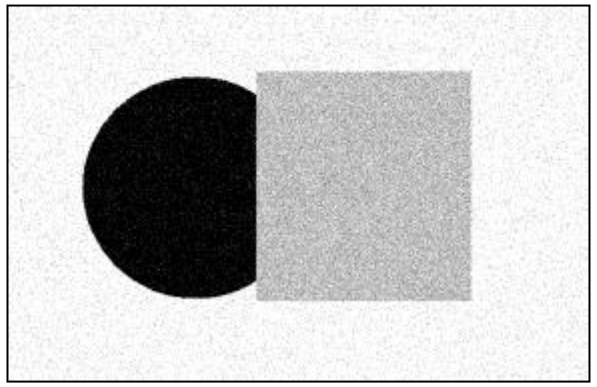




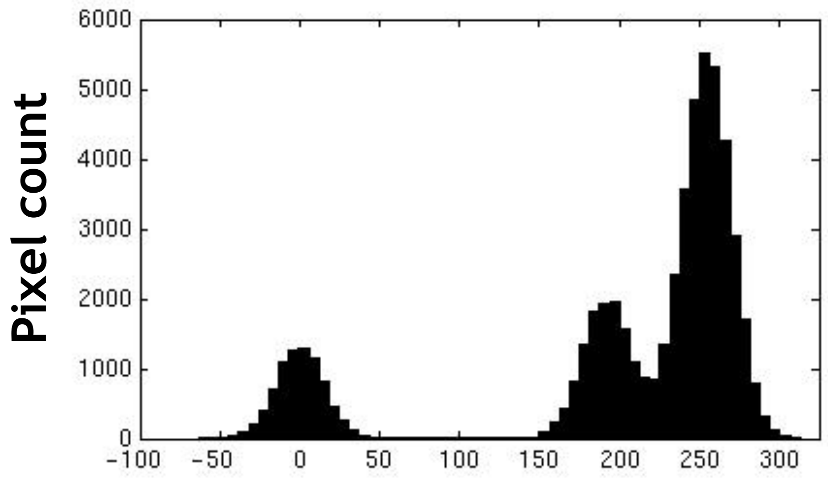
Input image



Slide credit: Kristen Grauman



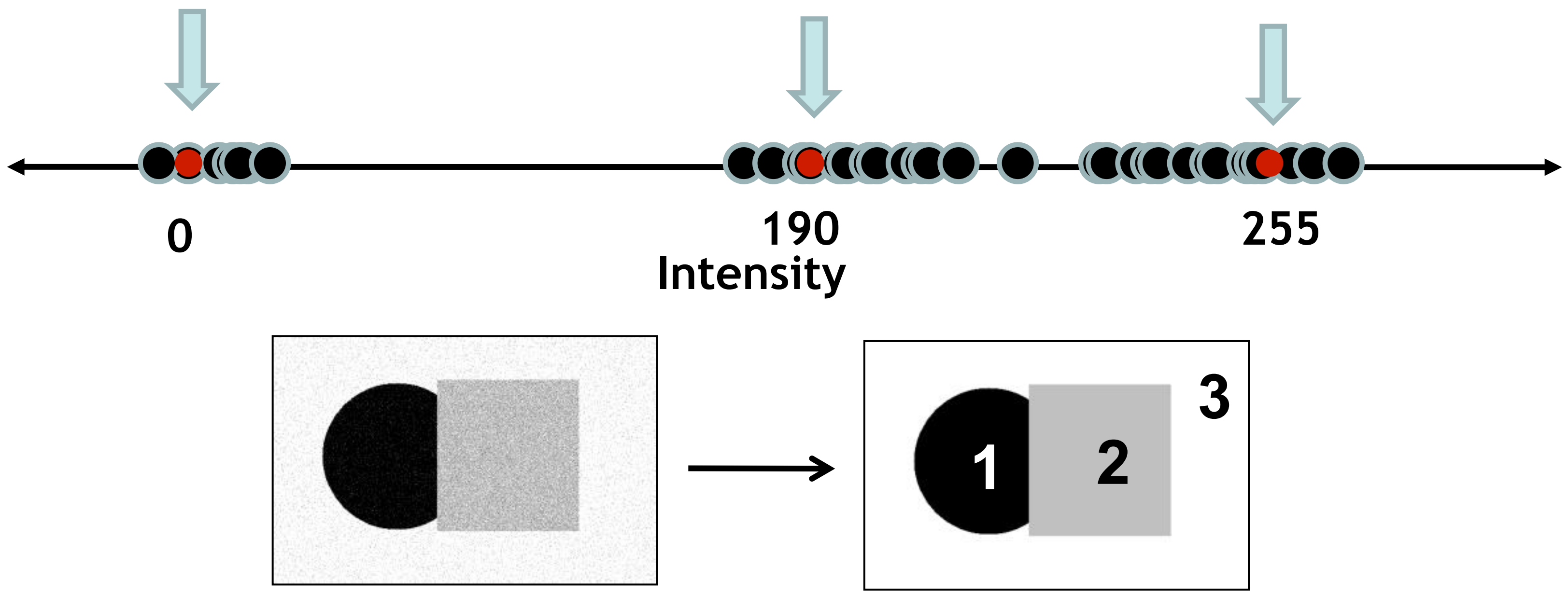
Input image



Intensity

- Now how to determine the three main intensities that define our groups?
- We need to cluster.

Slide credit: Kristen Grauman



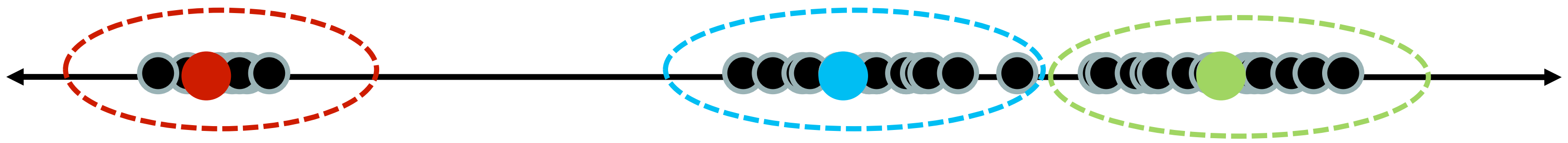
- Goal: choose three "centers" as the representative intensities, and label every pixel according to which of these centers it is nearest to.
- Best cluster centers are those that minimize SSD between all points and their nearest cluster center c_i:

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

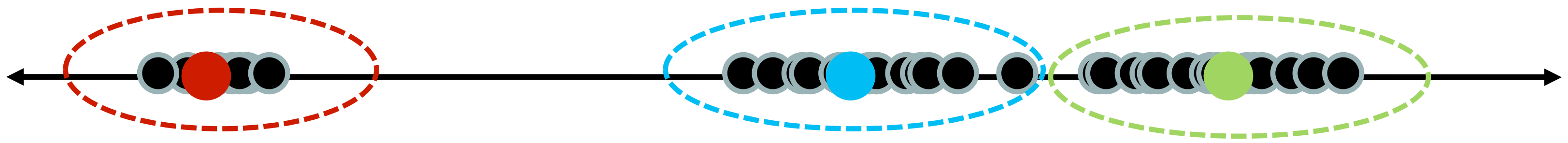
Slide credit: Kristen Grauman

Clustering

- With this objective, it is a "chicken and egg" problem:
  - If we knew the *cluster centers*, we could allocate points to groups by assigning each to its closest center.



  - If we knew the *group memberships*, we could get the centers by computing the mean per group.



Slide credit: Kristen Grauman

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, $c_1, \ldots, c_K$**
  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:**

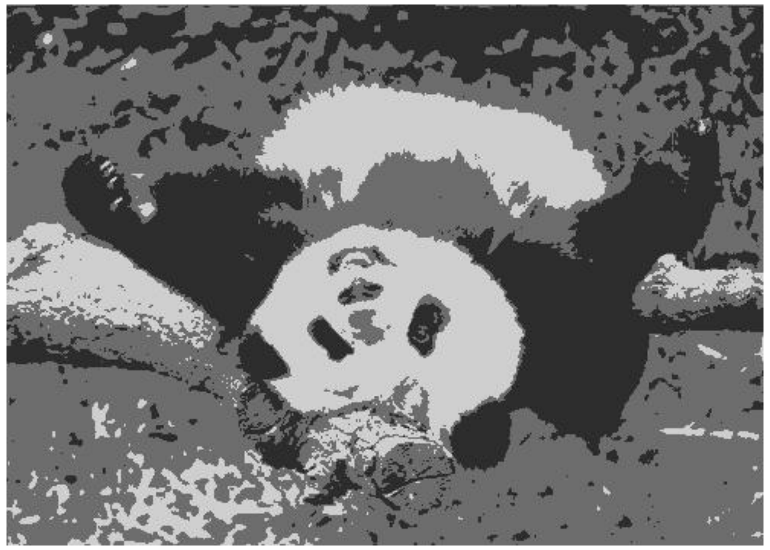
$$\sum_{\text{clusters } i} \sum_{\text{points p in cluster } i} ||p - c_i||^2$$

Slide credit: Steve Seitz

Segmentation as Clustering

K=2

K=3



```
img_as_col = double(im(:));
cluster_membs = kmeans(img_as_col, K);

labelim = zeros(size(im));
for i=1:k
   inds = find(cluster_membs==i);
   meanval = mean(img_as_column(inds));
   labelim(inds) = meanval;
end
```

Slide credit: Kristen Grauman

K-Means Clustering

- **Java demo:**

http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html

Feature Space

- Depending on what we choose as the *feature space*, we can group pixels in different ways.

- Grouping pixels based on intensity similarity

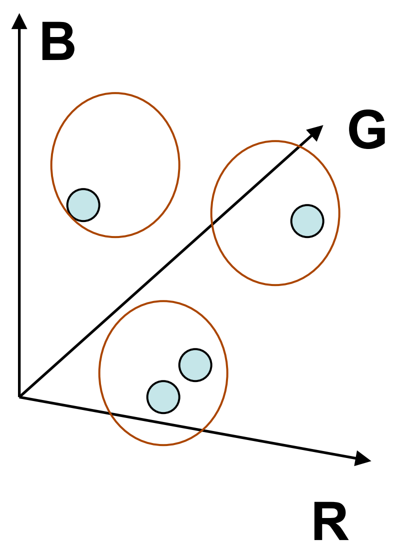
- Feature space: intensity value (1D)

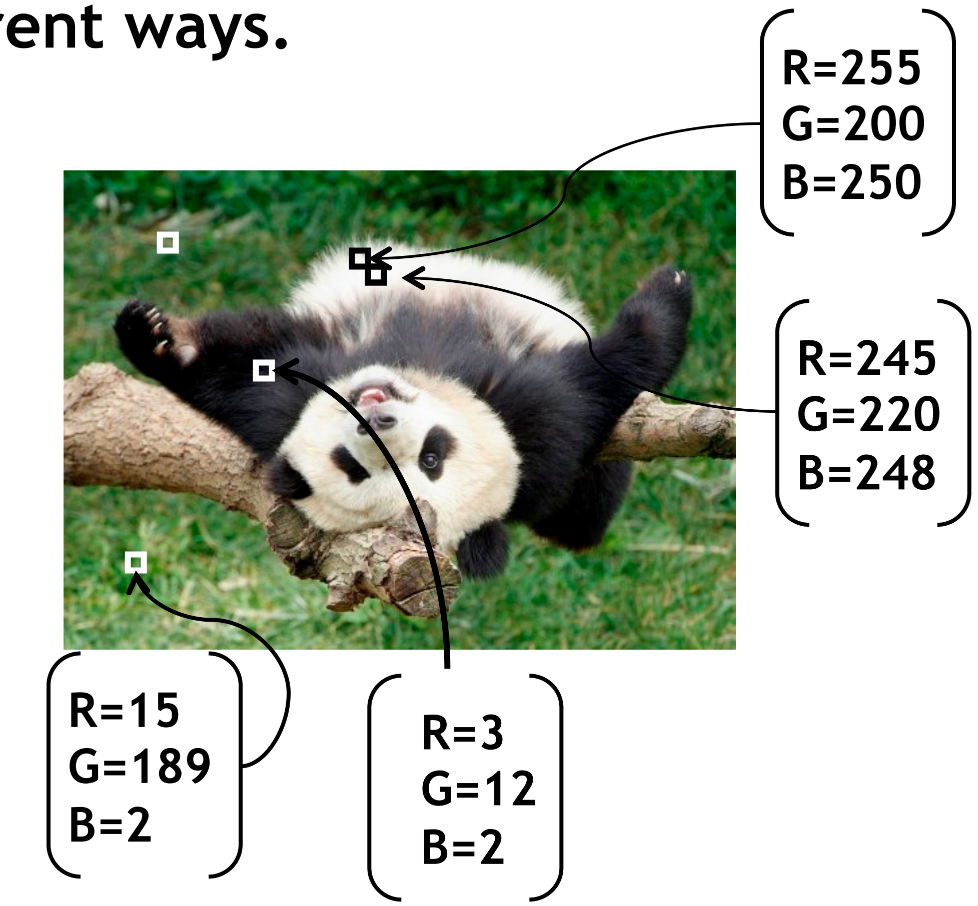
Slide credit: Kristen Grauman

Feature Space

- Depending on what we choose as the *feature space*, we can group pixels in different ways.

- Grouping pixels based on color similarity





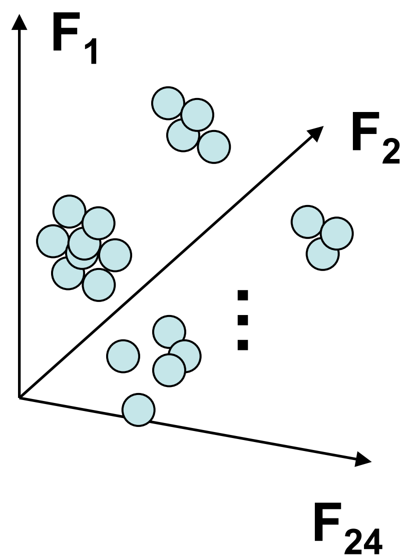
- Feature space: color value (3D)

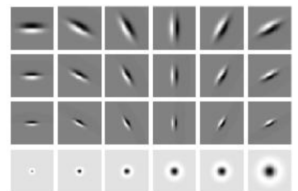
Slide credit: Kristen Grauman

Segmentation as Clustering

- Depending on what we choose as the *feature space*, we can group pixels in different ways.

- Grouping pixels based on texture similarity





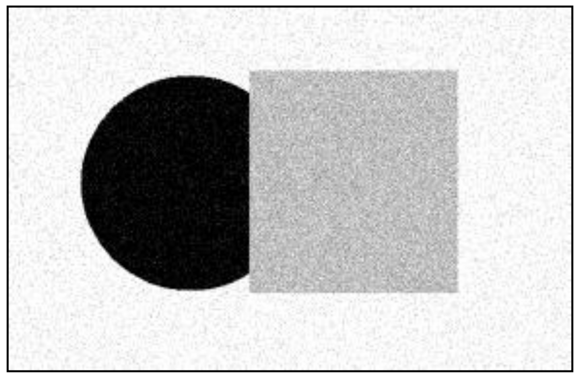
Filter bank of 24 filters

- Feature space: filter bank responses (e.g. 24D)

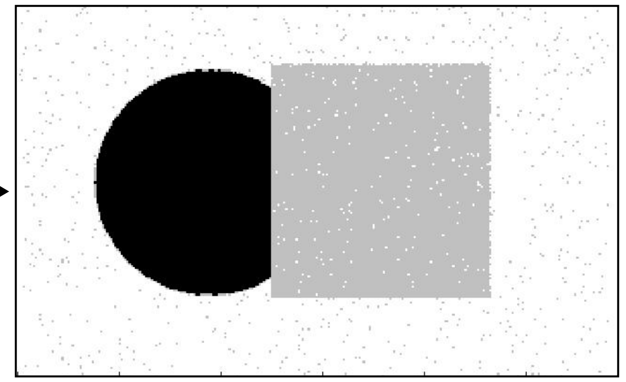
Slide credit: Kristen Grauman

Spatial coherence

- Assign a cluster label per pixel → possible discontinuities

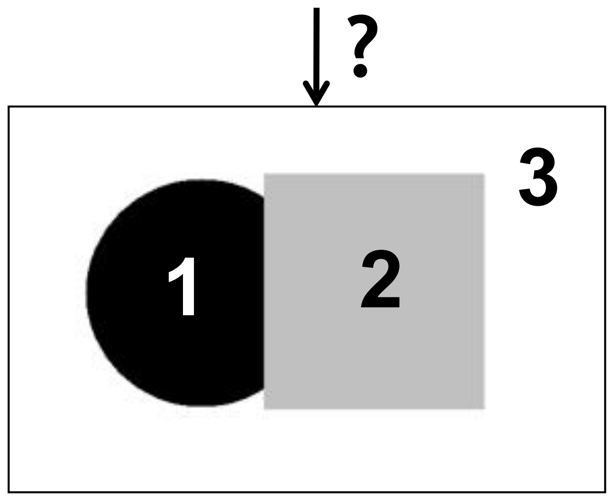


Original



Labeled by cluster center's intensity

- How can we ensure they are spatially smooth?

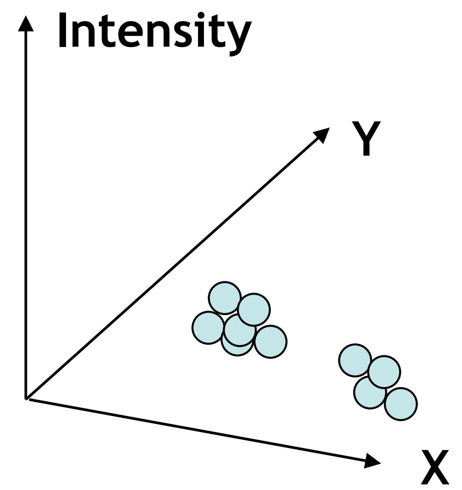


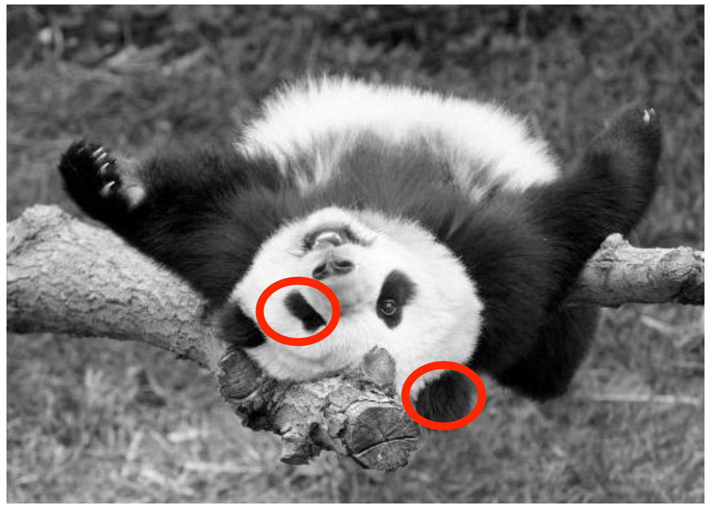
Slide adapted from Kristen Grauman

Spatial coherence

- Depending on what we choose as the *feature space*, we can group pixels in different ways.

- Grouping pixels based on *intensity+position* similarity





⇒ Way to encode both *similarity* and *proximity*.

Slide adapted from Kristen Grauman

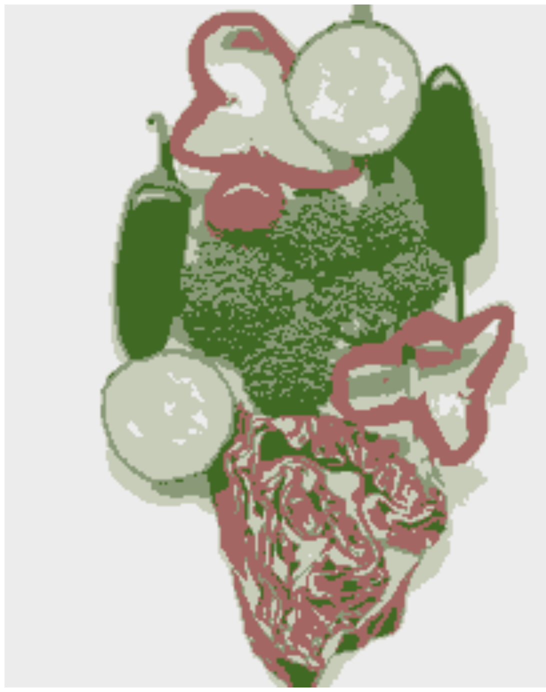
K-Means without spatial information

- K-means clustering based on intensity or color is essentially vector quantization of the image attributes
  - Clusters don't have to be spatially coherent

Image

Intensity-based clusters

Color-based clusters



Slide adapted from Svetlana Lazebnik

Image source: Forsyth & Ponce

K-Means with spatial information

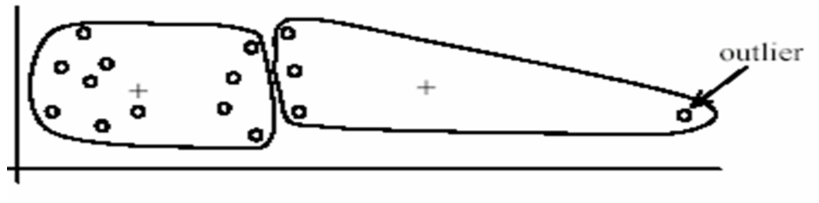
- K-means clustering based on intensity or color is essentially vector quantization of the image attributes
  - Clusters don't have to be spatially coherent
- Clustering based on (r,g,b,x,y) values enforces more spatial coherence

Slide adapted from Svetlana Lazebnik

Image source: Forsyth & Ponce

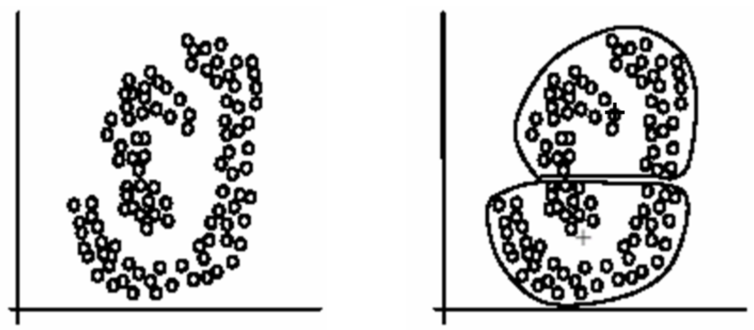
Summary K-Means

- **Pros**
  - Simple, fast to compute
  - Converges to local minimum of within-cluster squared error



(A): Undesirable clusters
(B): Ideal clusters

- **Cons/issues**
  - Setting k?
  - Sensitive to initial centers
  - Sensitive to outliers
  - Detects spherical clusters only
  - Assuming means can be computed


(A): Two natural clusters
(B): *k*-means clusters

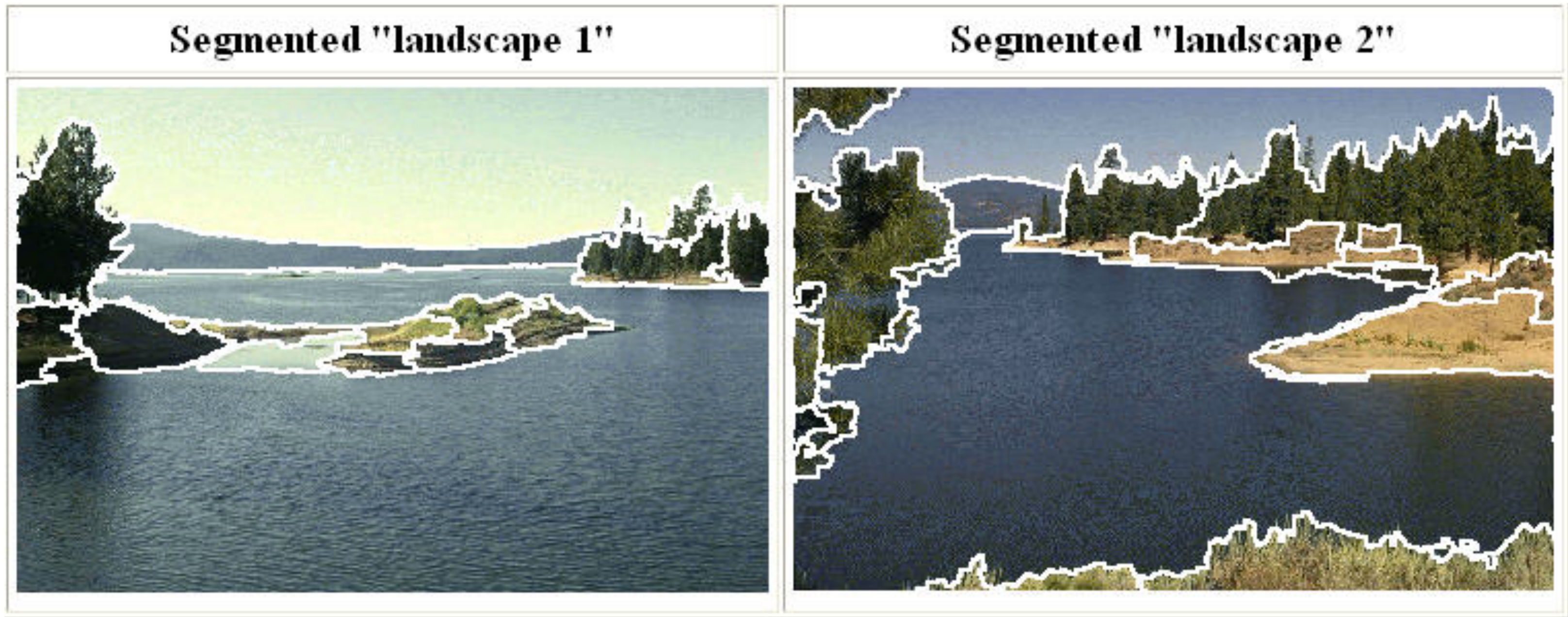
Slide credit: Kristen Grauman

Topics of This Lecture

- Introduction
  - Gestalt principles
  - Image segmentation
- Segmentation as clustering
  - k-Means
  - Feature spaces
- Model-free clustering: Mean-Shift
- Interactive Segmentation with GraphCuts

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

Finding Modes

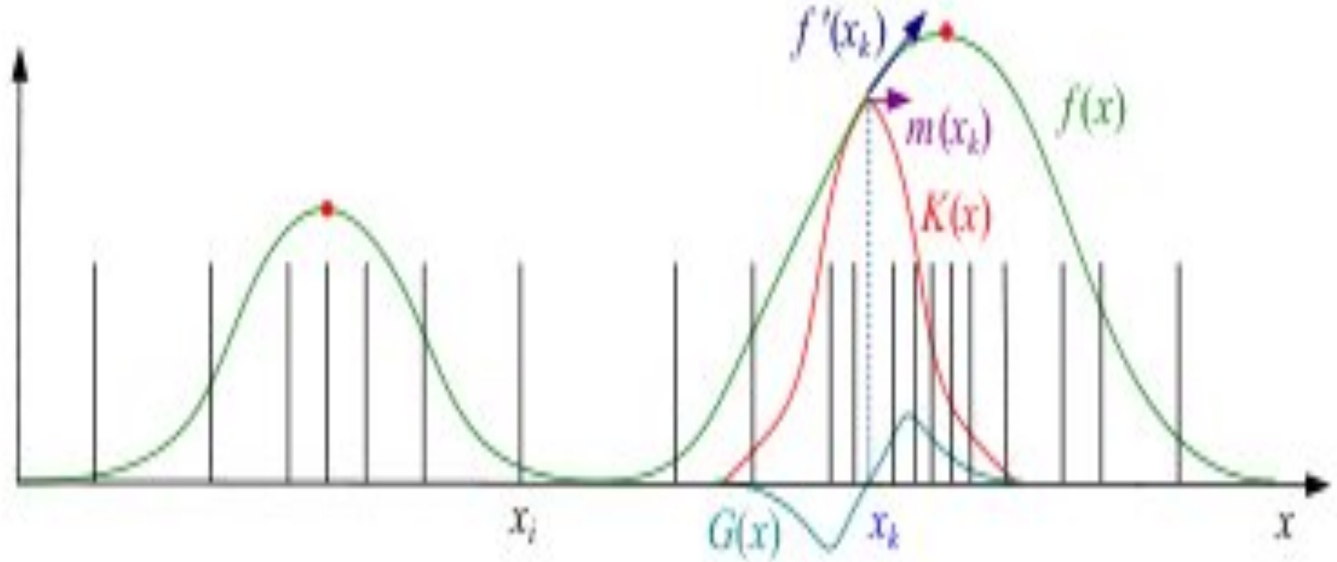


Figure credit: Szeliski (2011) Fig 5.17

$$f(x) = \sum_i K(x - x_i) \qquad K(x - x_i) = k\left(\frac{|x - x_i|^2}{h^2}\right)$$

Goal: find peaks (modes) of *f(x)*

Mean-Shift Algorithm

$$\nabla f(x) = \sum_i (x_i - x) G(x - x_i) = 0$$

$$y_{k+1} = \frac{\sum_i x_i G(y_k - x_i)}{\sum_i G(y_k - x_i)}$$

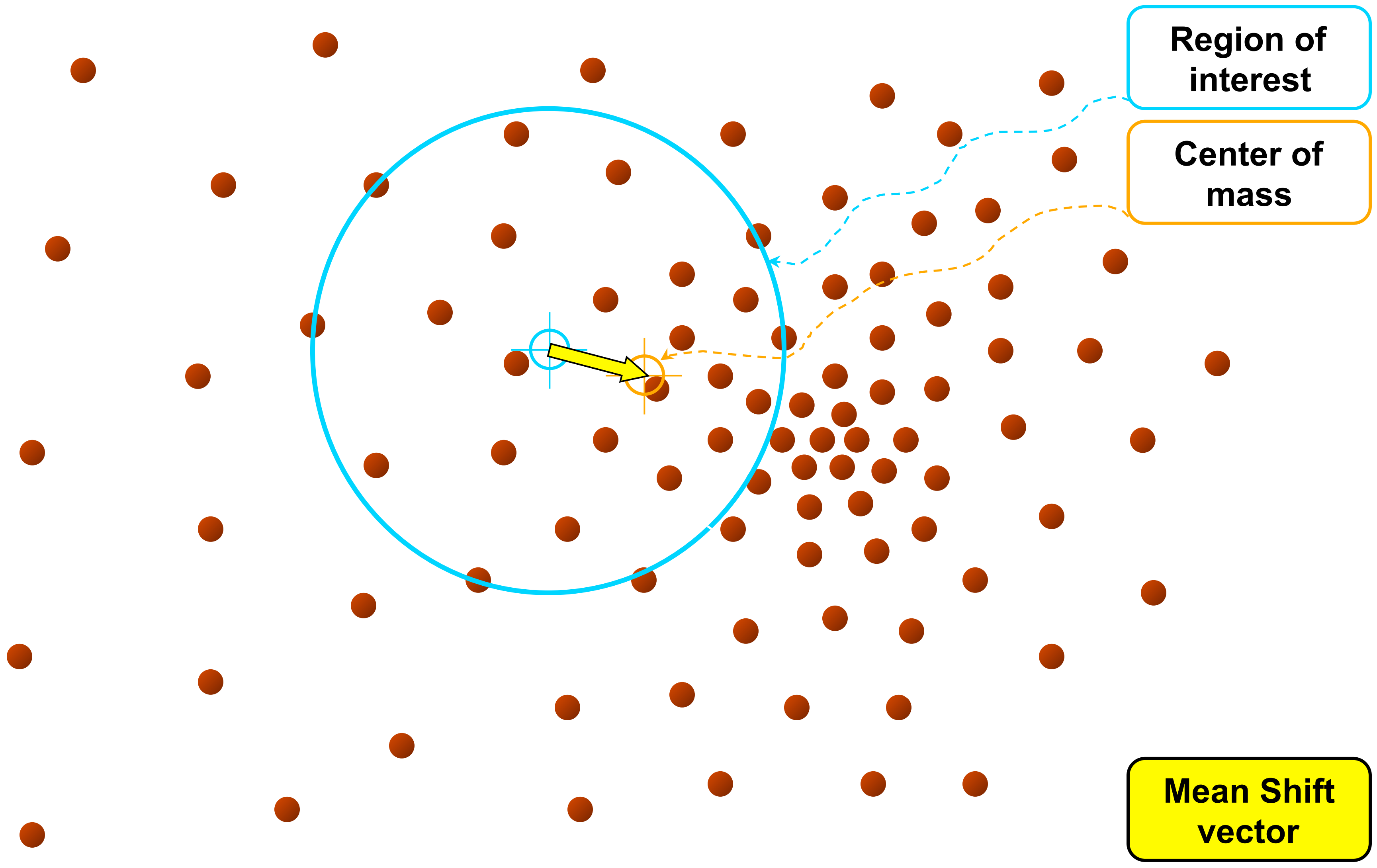
Note: G() is the derivative of K()

Iterative Mode Search

1. **Initialize random seed center y for k=0 (can be a data point)**
2. **Compute the weights** $G(y_k - x_i)$
3. **Calculate weighted mean** y_{k+1} **as above**
4. **Repeat steps 2+3 until convergence**

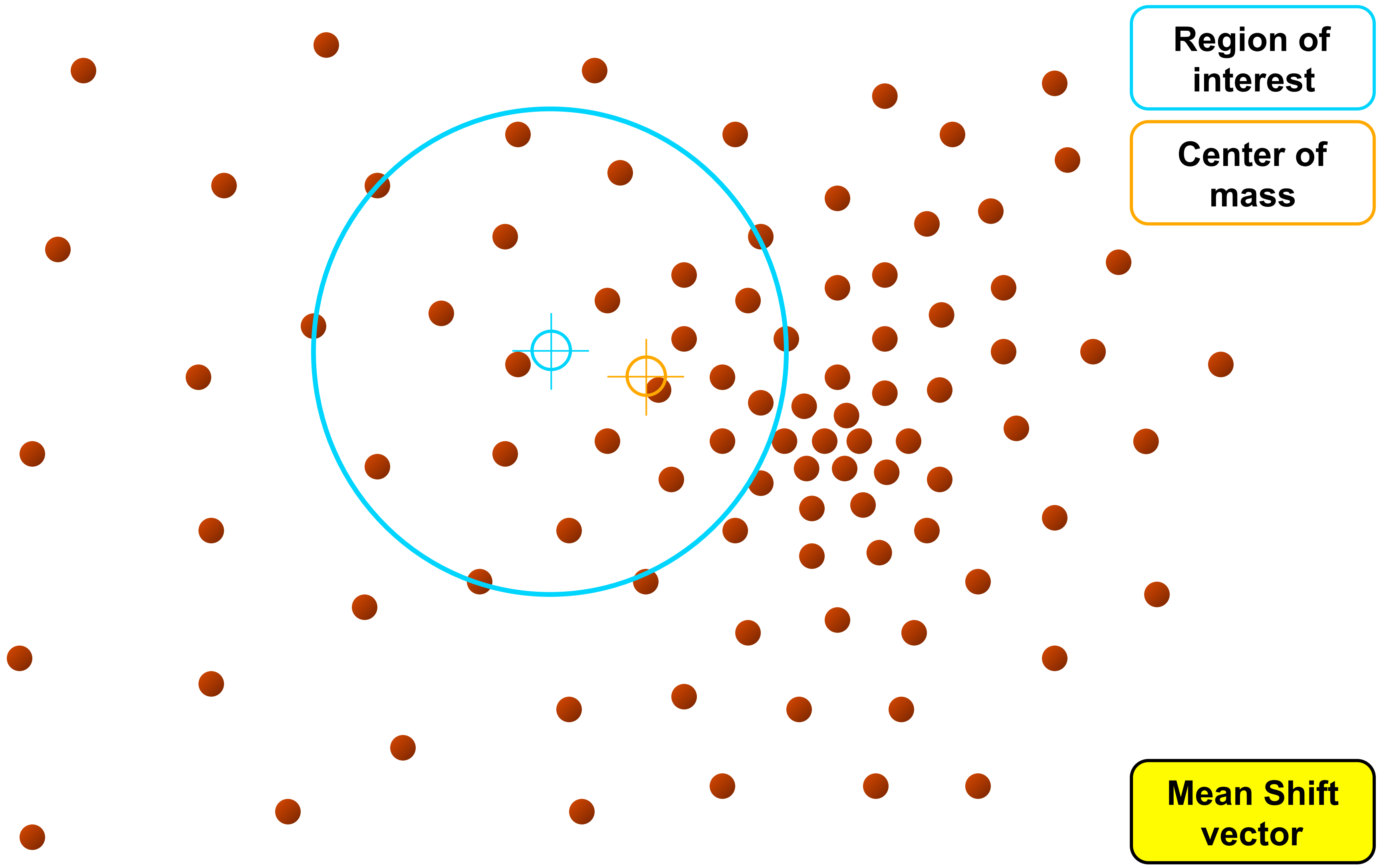
Slide adapted from Steve Seitz

Mean-Shift



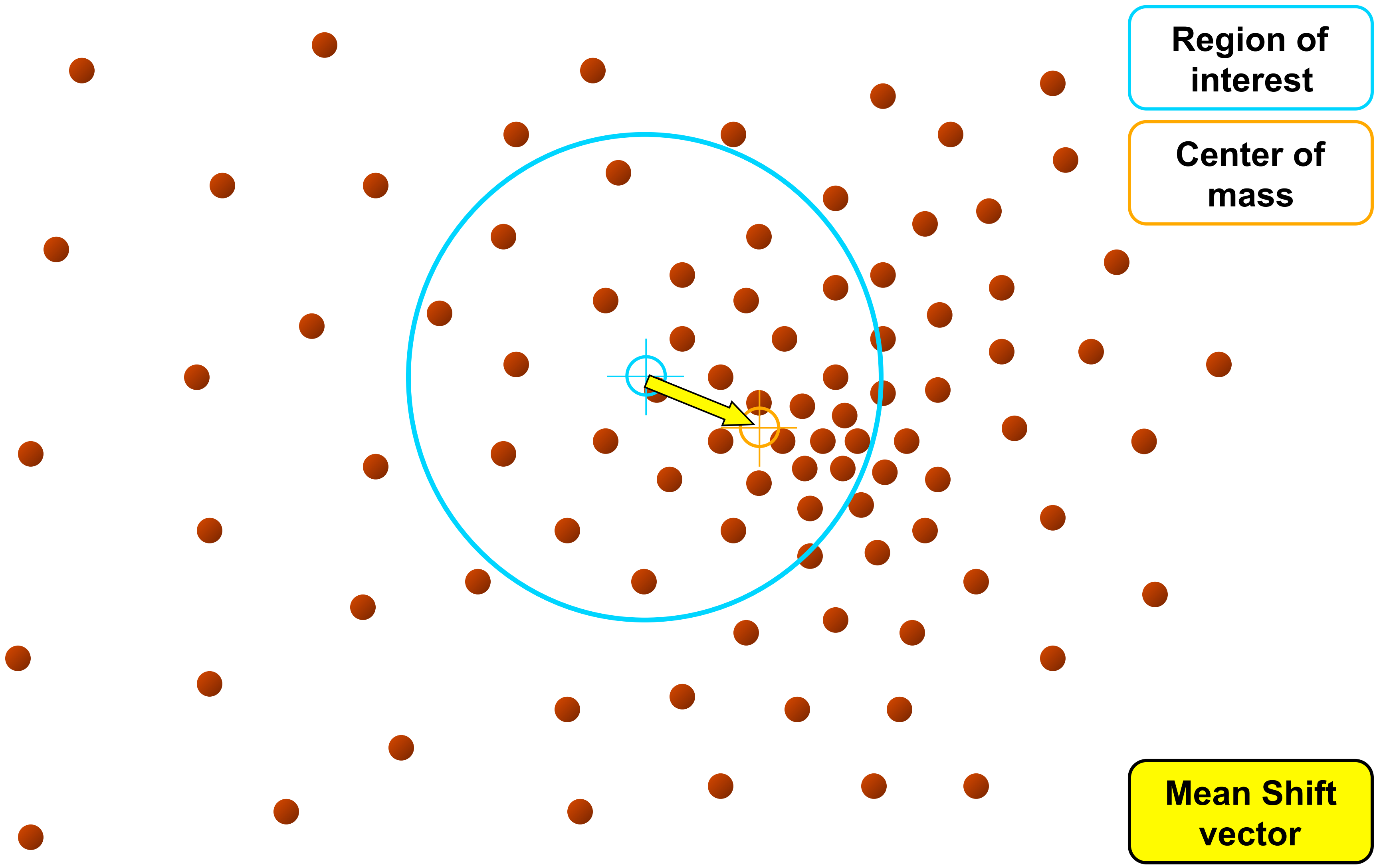
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



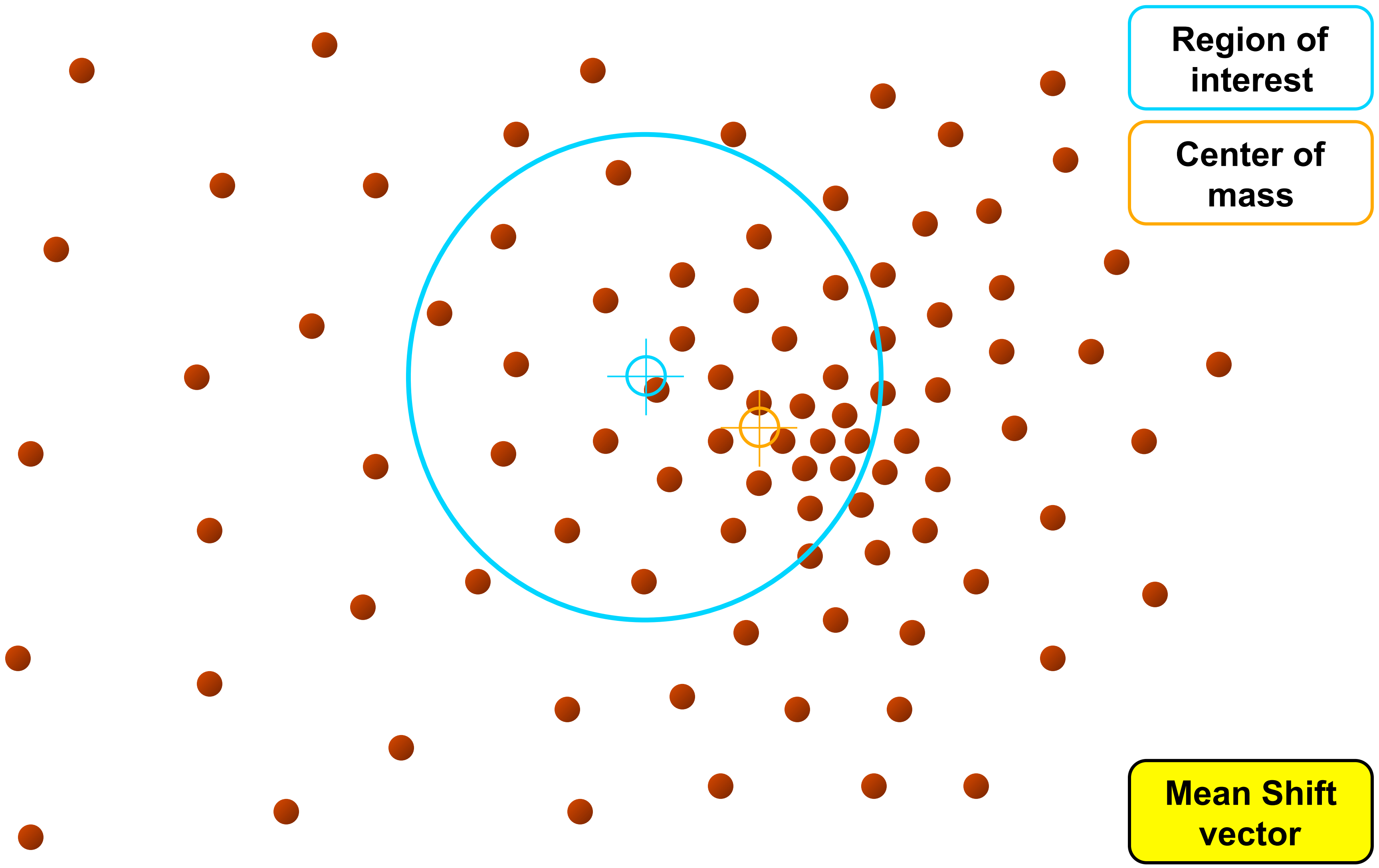
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



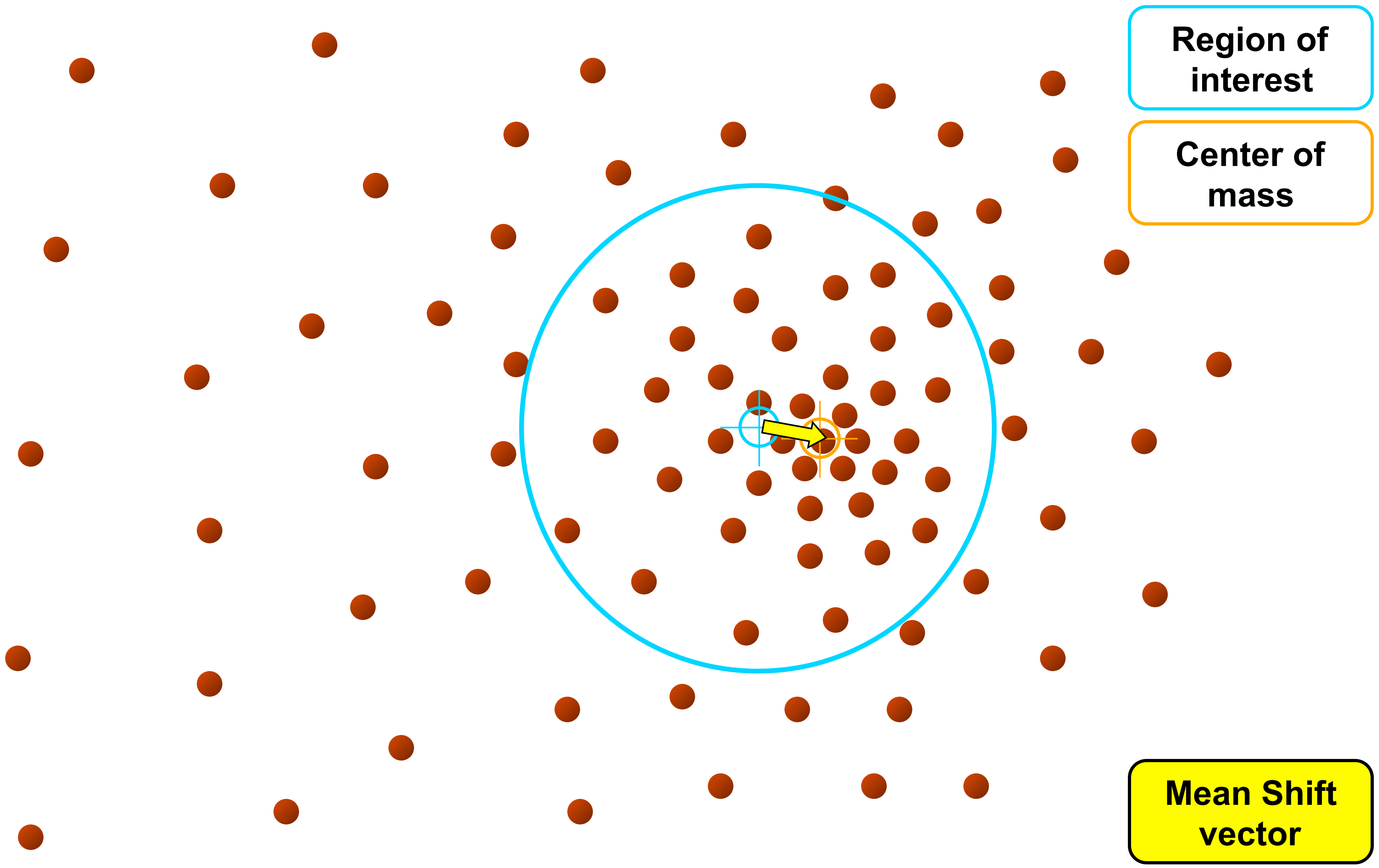
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



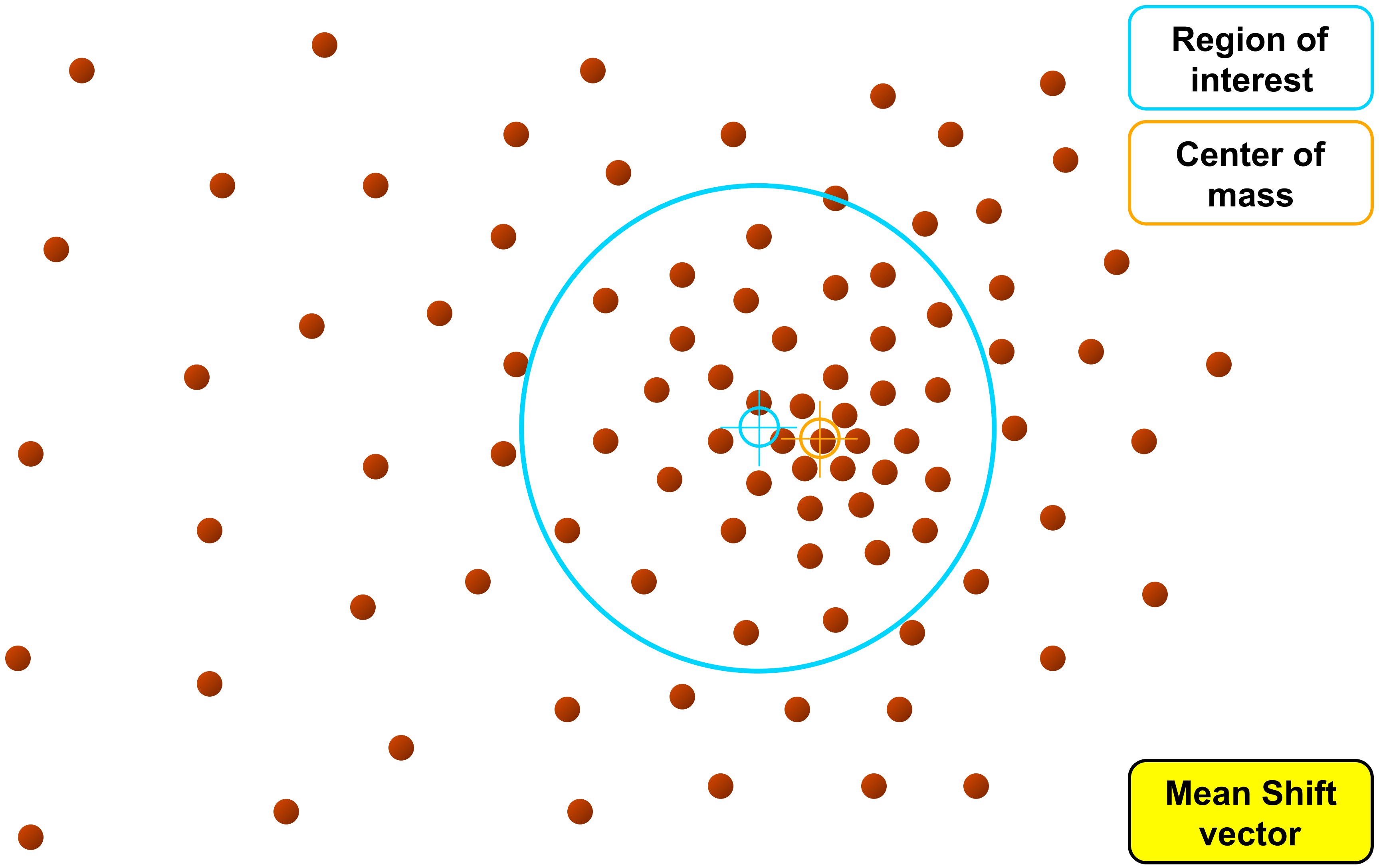
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



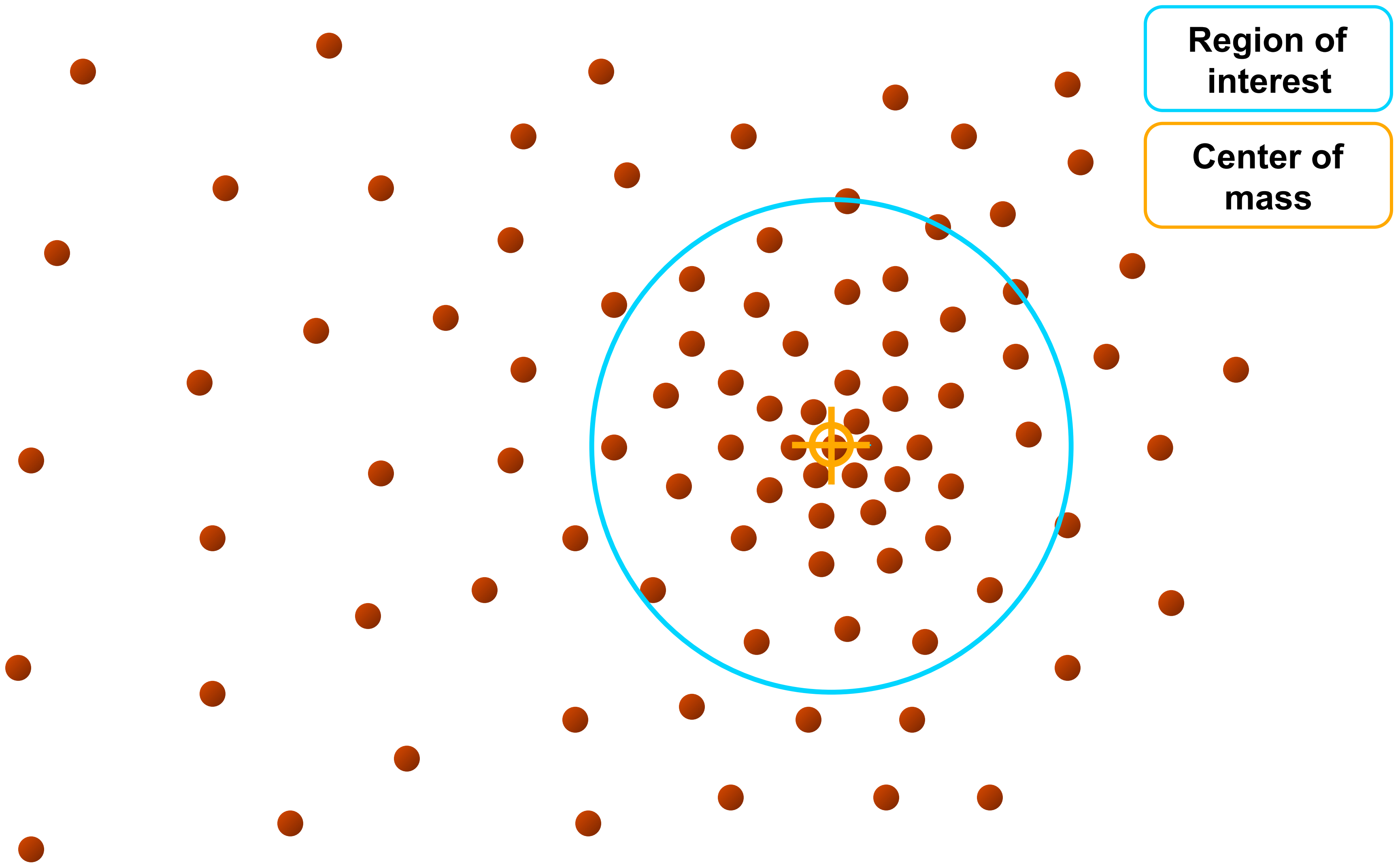
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



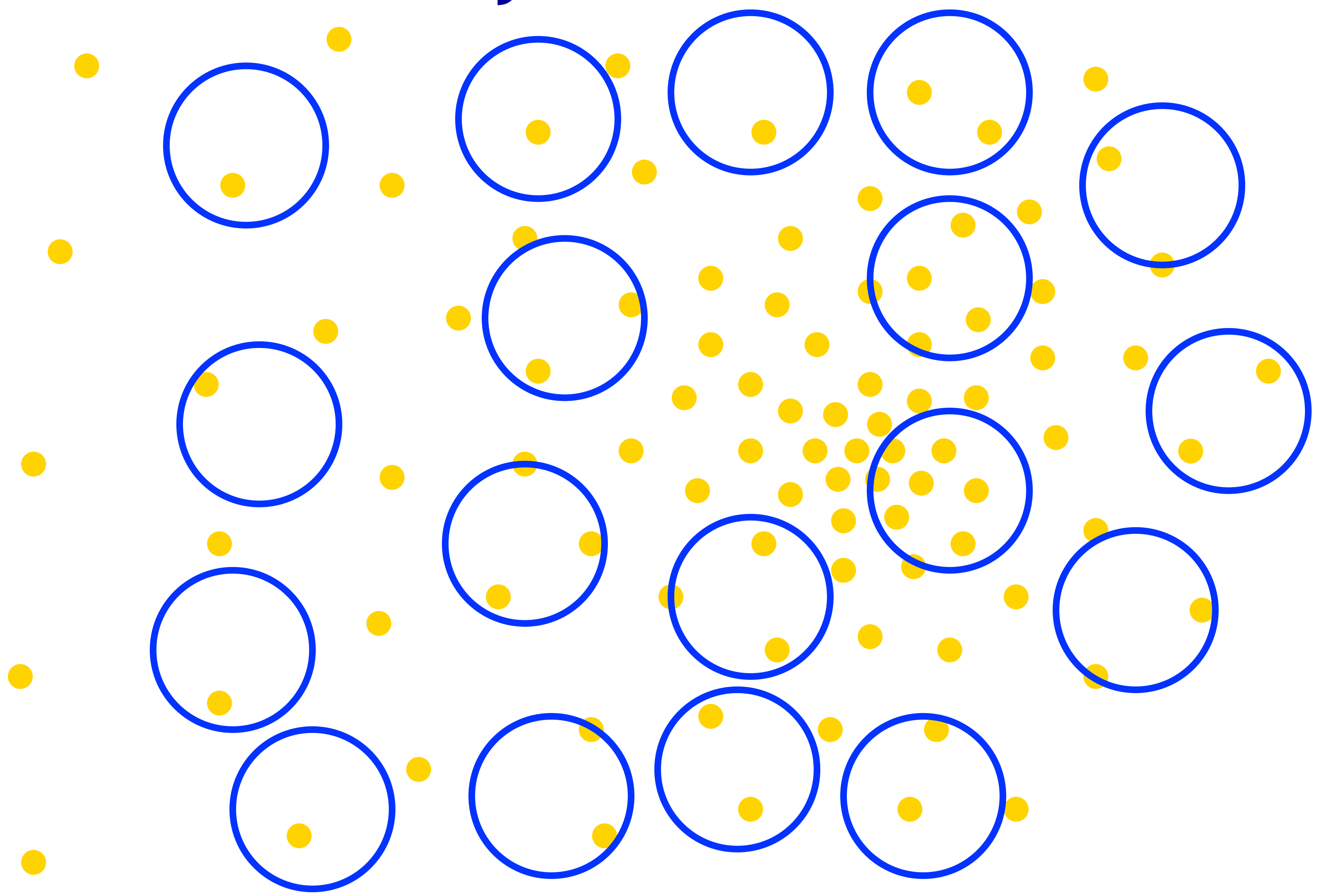
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



Slide by Y. Ukrainitz & B. Sarel

Real Modal Analysis

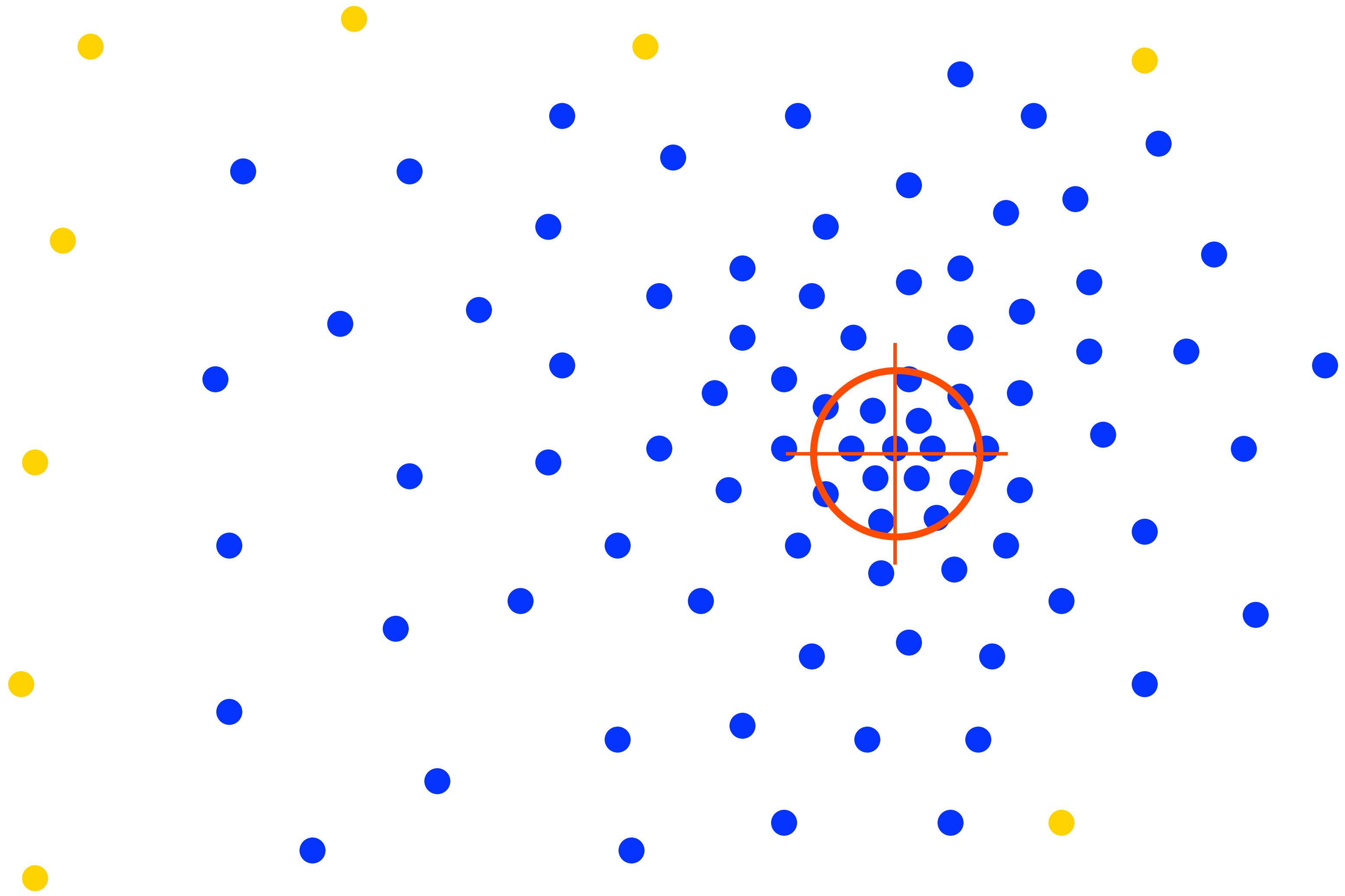


Tessellate the space with windows

Run the procedure in parallel

Slide by Y. Ukrainitz & B. Sarel

Real Modal Analysis

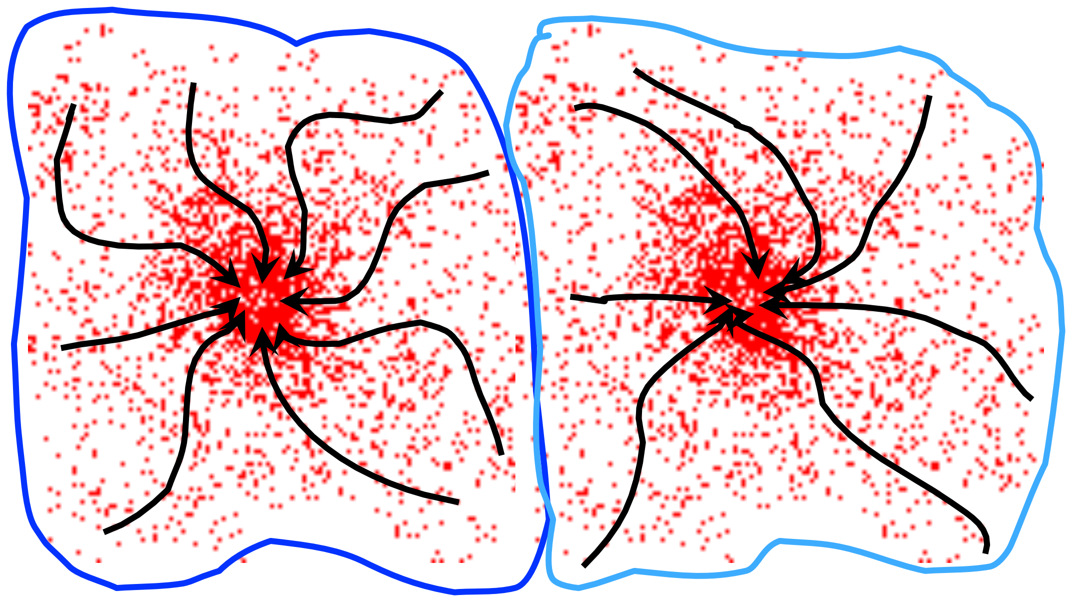


The blue data points were traversed by the windows towards the mode.

Slide by Y. Ukrainitz & B. Sarel

Mean-Shift Clustering

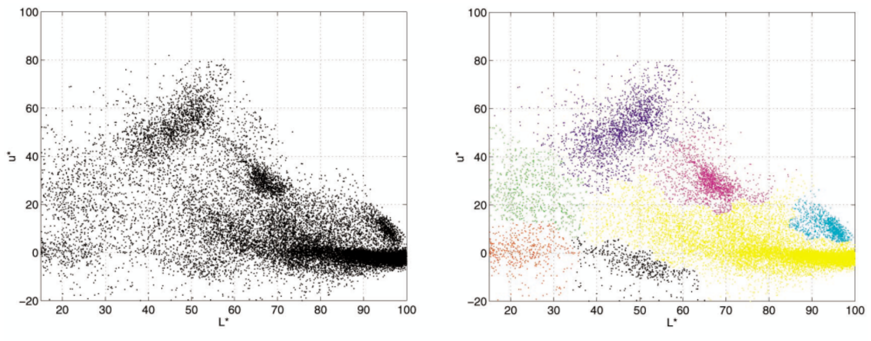
- Cluster: all data points in the attraction basin of a mode
- Attraction basin: the region for which all trajectories lead to the same mode



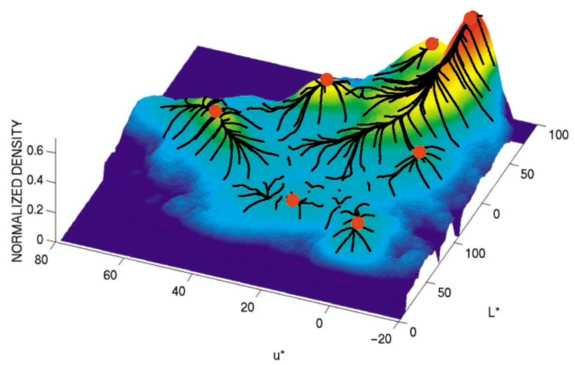
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift Clustering/Segmentation

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

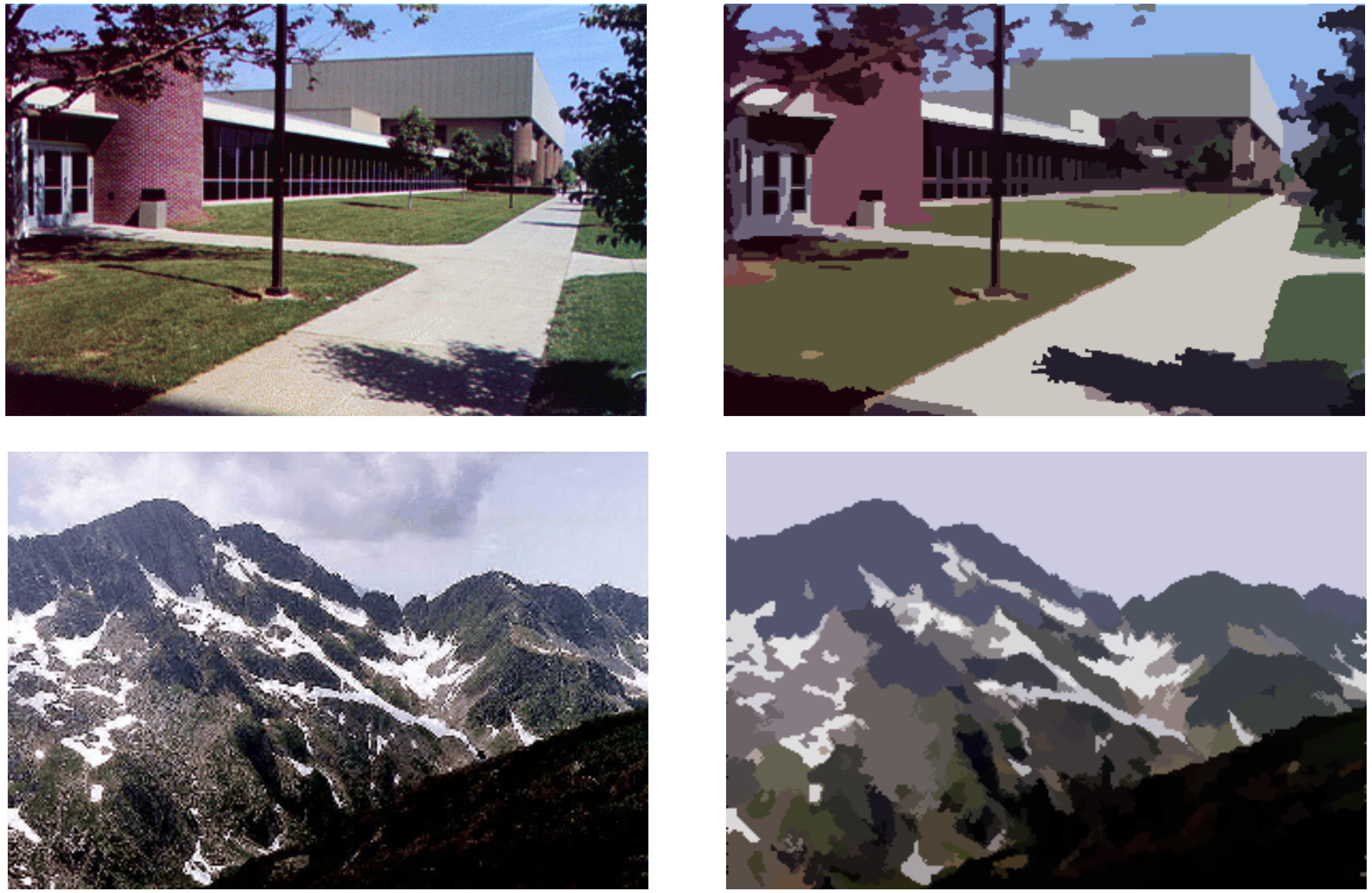


(a) (b)



Slide adapted from 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

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) == scale of clustering
  - Finds variable number of modes given the same h
  - Robust to outliers

- Cons
  - Output depends on window size h
  - Window size (bandwidth) selection is not trivial
  - Computationally rather expensive
  - Does not scale well with dimension of feature space

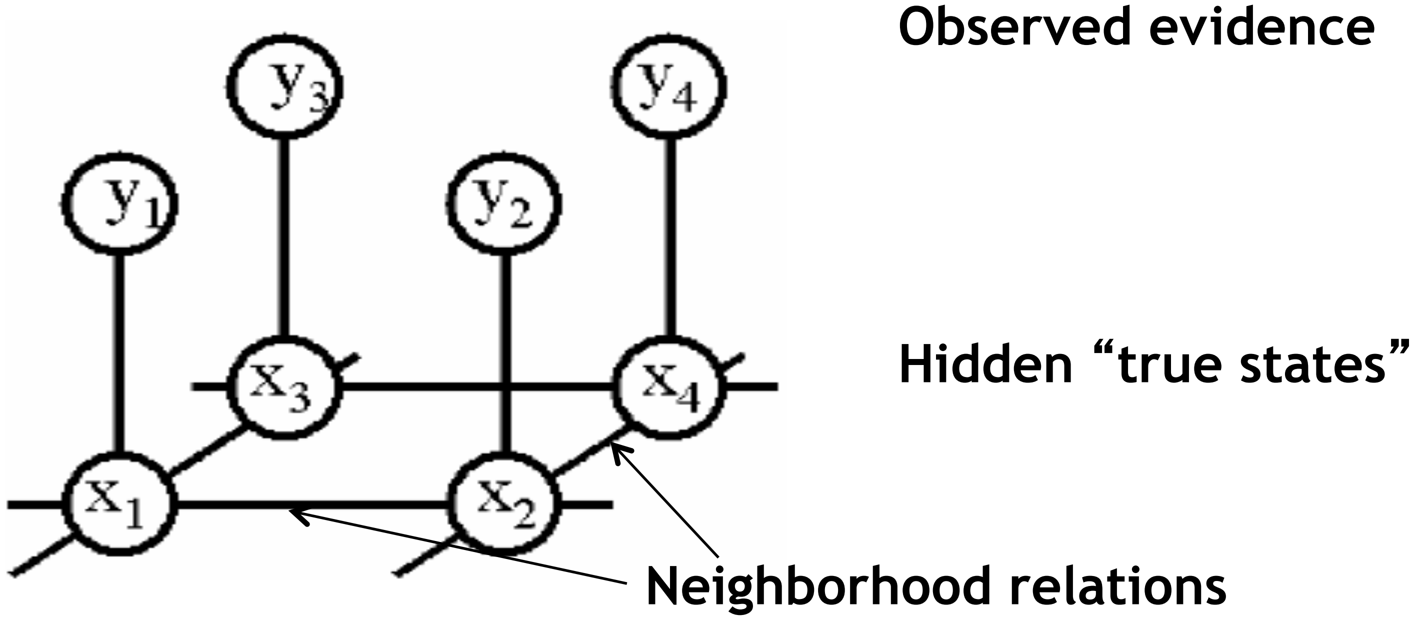
Slide adapted from Svetlana Lazebnik

Topics of This Lecture

- Introduction
  - Gestalt principles
  - Image segmentation
- Segmentation as clustering
  - k-Means
  - Feature spaces
- Model-free clustering: Mean-Shift
- Interactive Segmentation with GraphCuts

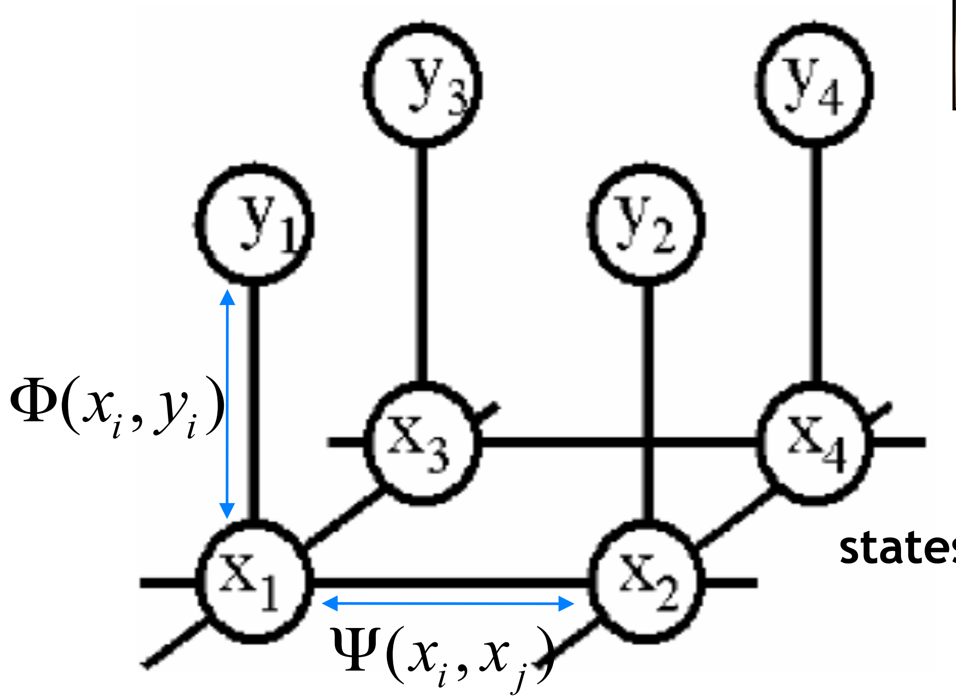
Markov Random Fields

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

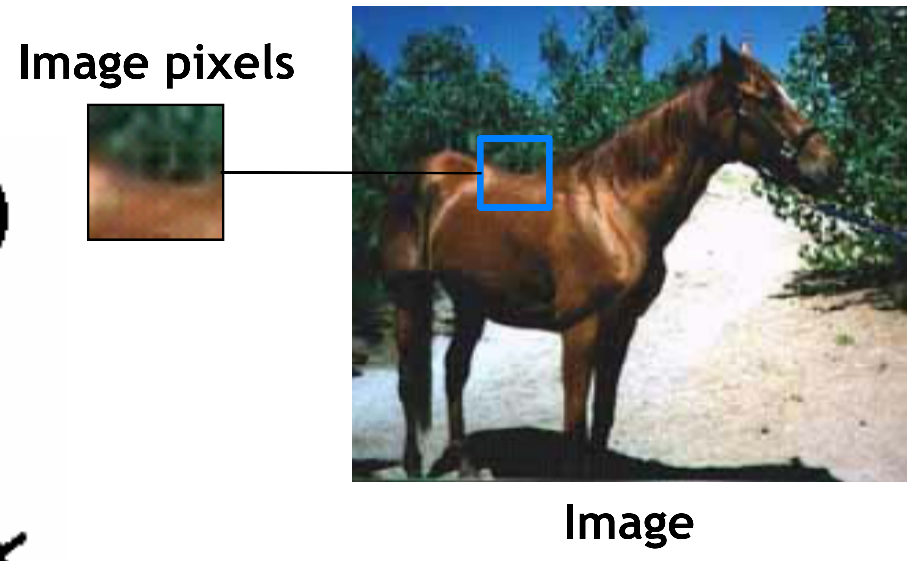


Slide credit: William Freeman

MRF Nodes as Pixels (or Patches)



states (e.g. foreground/background)



Slide adapted from William Freeman

Network Joint Probability

$$P(x,y) = \frac{1}{Z}\prod_i \Phi(x_i, y_i)\prod_{i,j}\Psi(x_i, x_j)$$

states

Image

Image-state compatibility function

Local observations

state-state compatibility function

Neighboring nodes

Slide adapted from William Freeman

Energy Formulation

- Joint probability

$$P(x,y) = \frac{1}{Z}\prod_i \Phi(x_i, y_i)\prod_{i,j}\Psi(x_i, x_j)$$

- Maximizing the joint probability is the same as minimizing the negative log

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

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

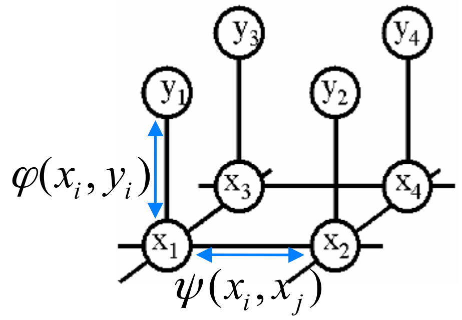
- This is similar to free-energy problems in statistical mechanics (spin glass theory). We therefore draw the analogy and call E an *energy function*.
- φ and ψ are called *potentials*.

Slide credit: B. Leibe

Energy Formulation

- Energy function

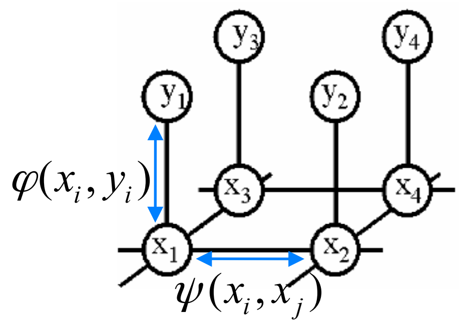
$$E(x,y) = \underbrace{\sum_i \varphi(x_i, y_i)}_{\text{Unary potentials}} + \underbrace{\sum_{i,j} \psi(x_i, x_j)}_{\text{Pairwise potentials}}$$



- Unary potentials φ
  - Encode local information about the given pixel/patch
  - How likely is a pixel/patch to be in a certain state ? (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. here independent of image data, but later based on intensity/color/texture difference)

Slide adapted from B. Leibe

Energy Minimization

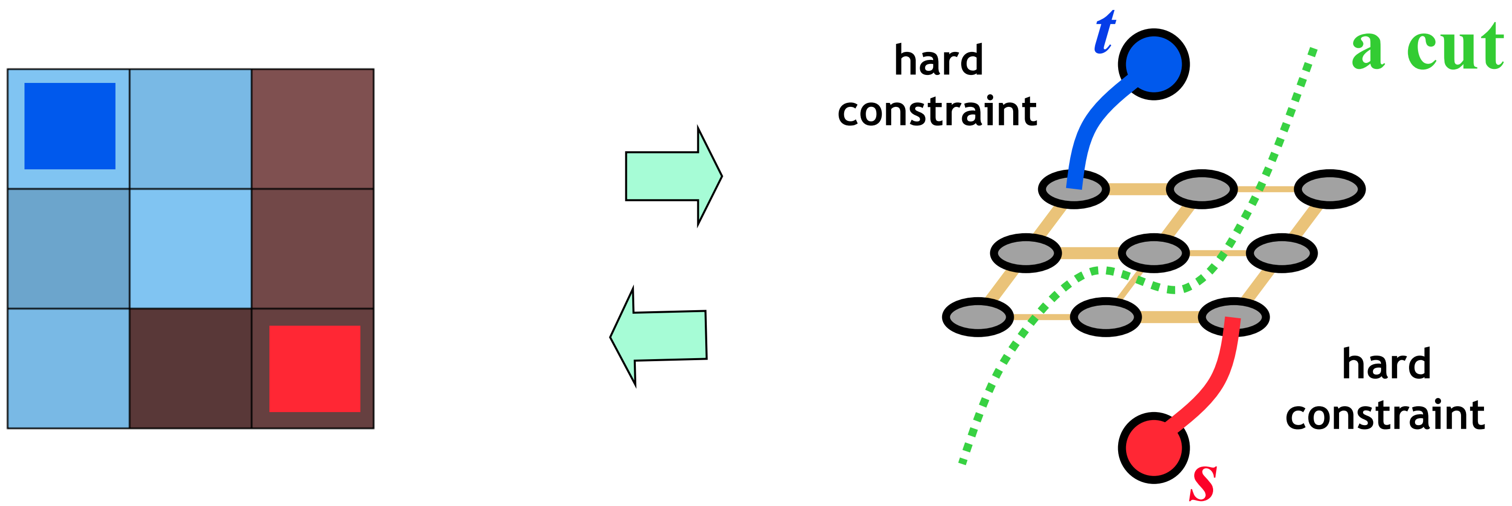


- Goal:
  - Infer the optimal labeling of the MRF.
- Many inference algorithms are available, e.g.
  - Gibbs sampling, simulated annealing
  - Iterated conditional modes (ICM)
  - Variational methods
  - Belief propagation
  - Graph cuts
- Recently, Graph Cuts have become a popular tool
  - Only suitable for a certain class of energy functions
  - But the solution can be obtained very fast for typical vision problems (~1MPixel/sec).

Slide credit: B. Leibe

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)

Slide adapted from Yuri Boykov

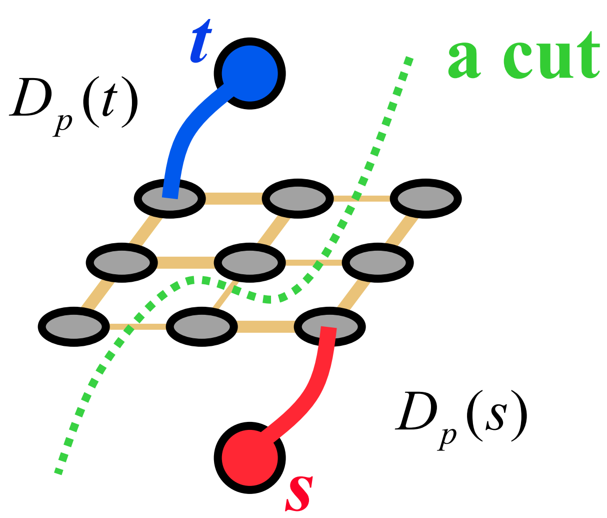
[Boykov & Jolly, ICCV' 01]

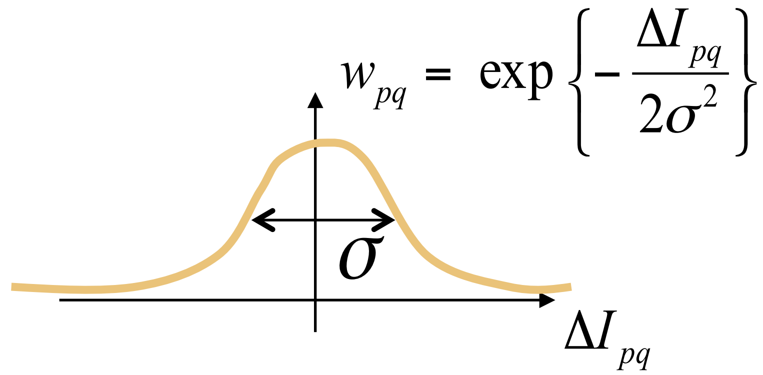
Simple Example of Energy

Regional term | Boundary term

$$E(L) = \sum_{p} D_p(L_p) + \sum_{pq \in N} w_{pq} \cdot \delta(L_p \neq L_q)$$

t-links | n-links



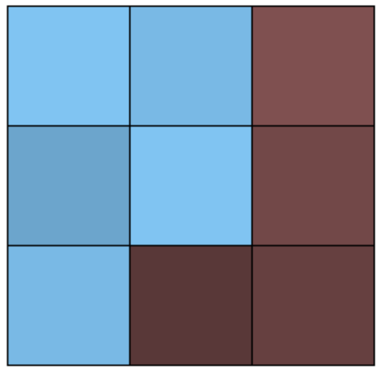


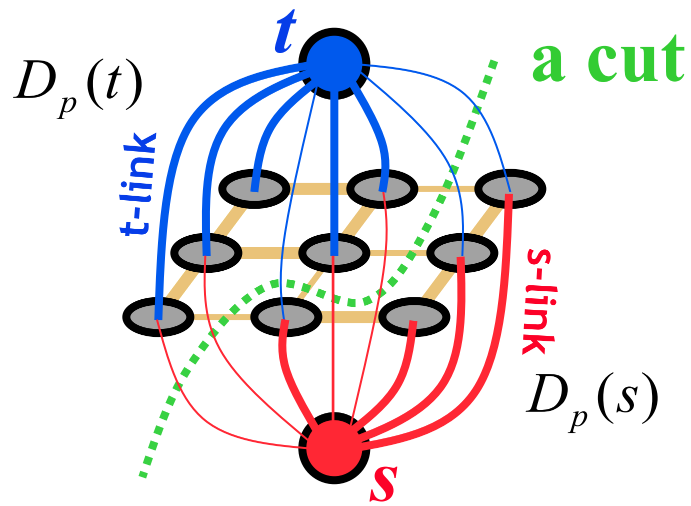
$$L_p \in \{s, t\}$$

(binary 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

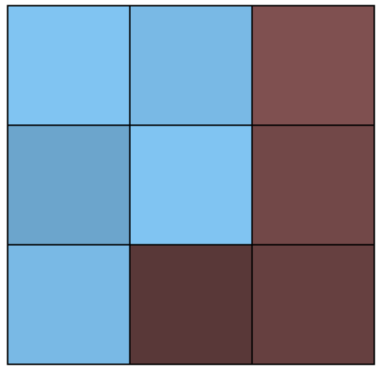
$$D_p(s) \propto \exp\left(-\| I_p - I^s \|^2 / 2\sigma^2\right)$$
$$D_p(t) \propto \exp\left(-\| I_p - I^t \|^2 / 2\sigma^2\right)$$

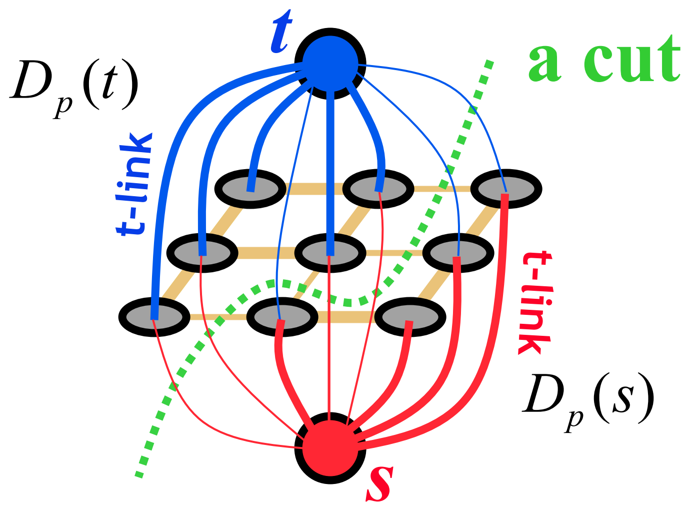
NOTE: hard constrains are not required, in general.

Slide credit: Yuri Boykov

[Boykov & Jolly, ICCV' 01]

Adding Regional Properties





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

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

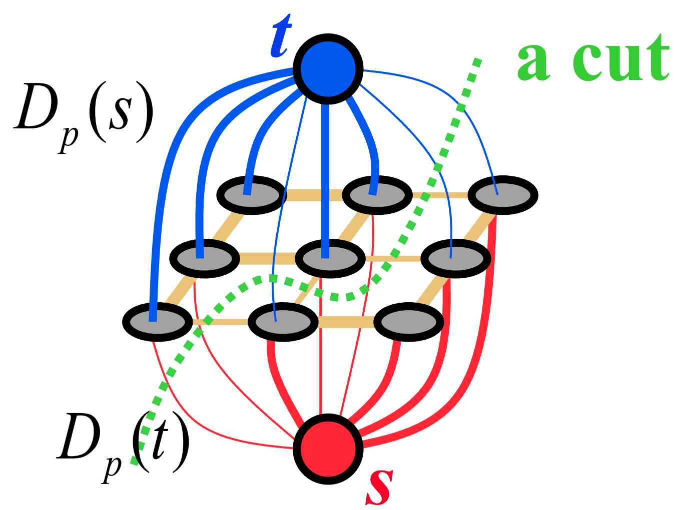
EM-style optimization

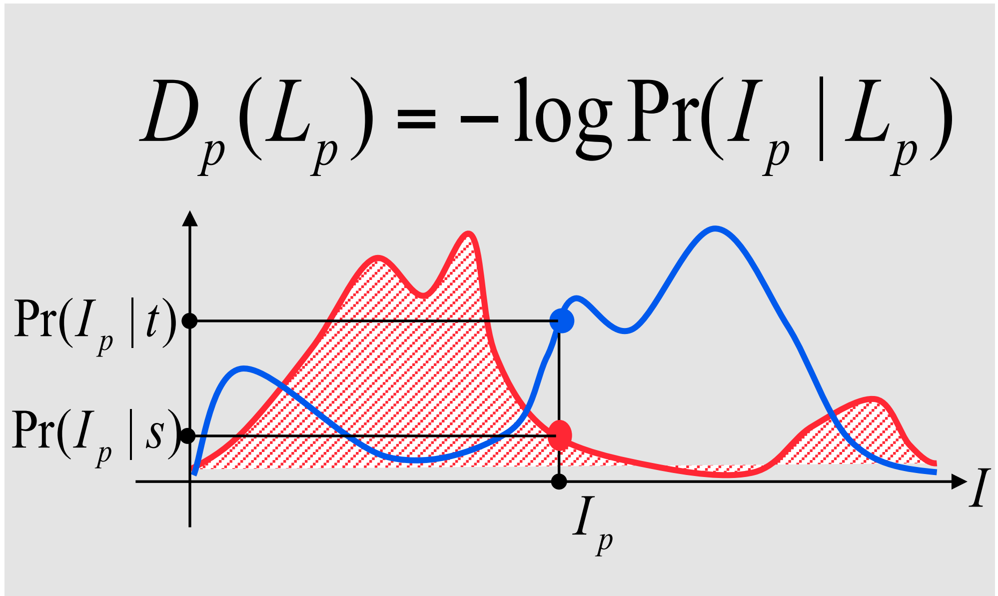
Slide credit: Yuri Boykov

[Boykov & Jolly, ICCV' 01]

Adding Regional Properties

- More generally, regional bias can be based on any appearance model of object and background





given object and background intensity histograms

Slide credit: Yuri Boykov

[Boykov & Jolly, ICCV' 01]

How to Set the Potentials? Some Examples

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

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

- **Edge potentials**
  - **e.g. a "contrast sensitive Potts model"**

$$\psi(x_i, x_j, g_{ij}(y); \theta_\phi) = \gamma g_{ij}(y) \delta(x_i \neq x_j)$$

where

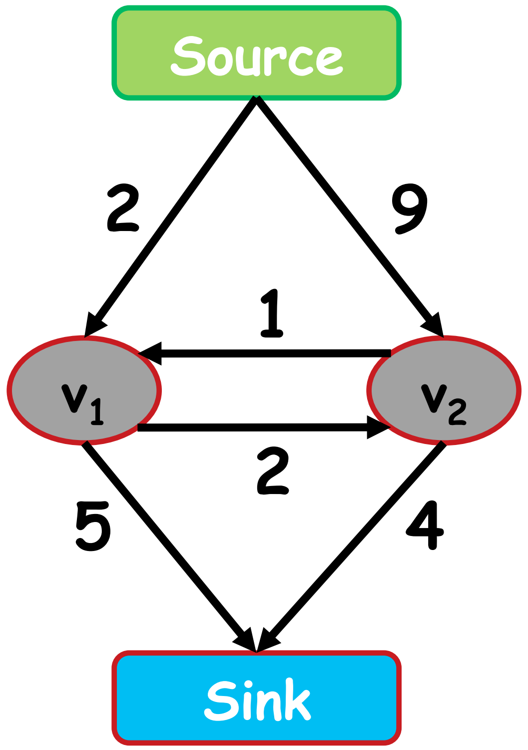
$$g_{ij}(y) = e^{-\beta \|y_i - y_j\|^2} \qquad \beta = 2 \cdot avg\left(\|y_i - y_j\|^2\right)$$

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

Slide credit: B. Leibe

[Shotton & Winn, ECCV'06]

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



Graph (V, E, C)

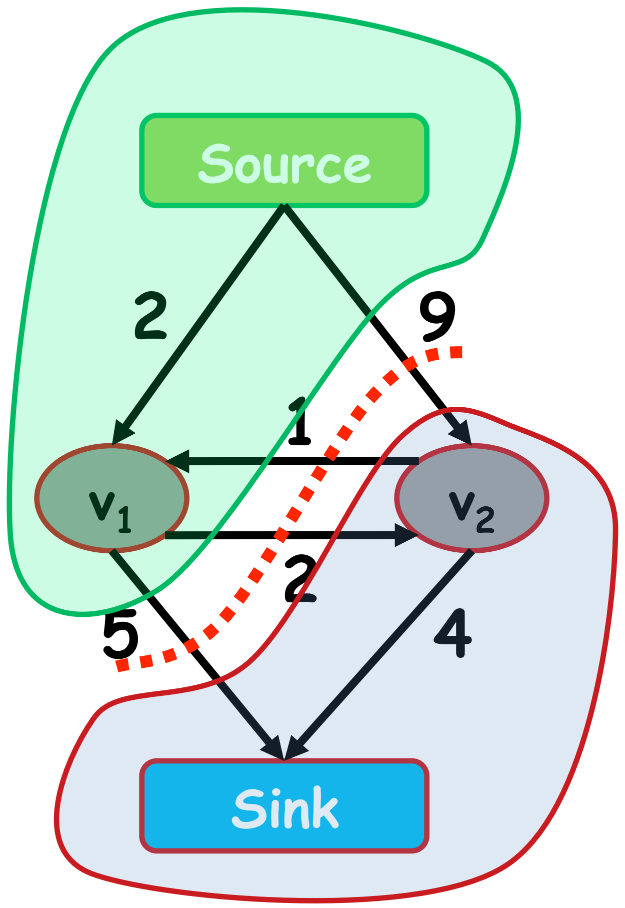
Vertices $V = \{v_1, v_2 \dots v_n\}$

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

Slide credit: Pushmeet Kohli

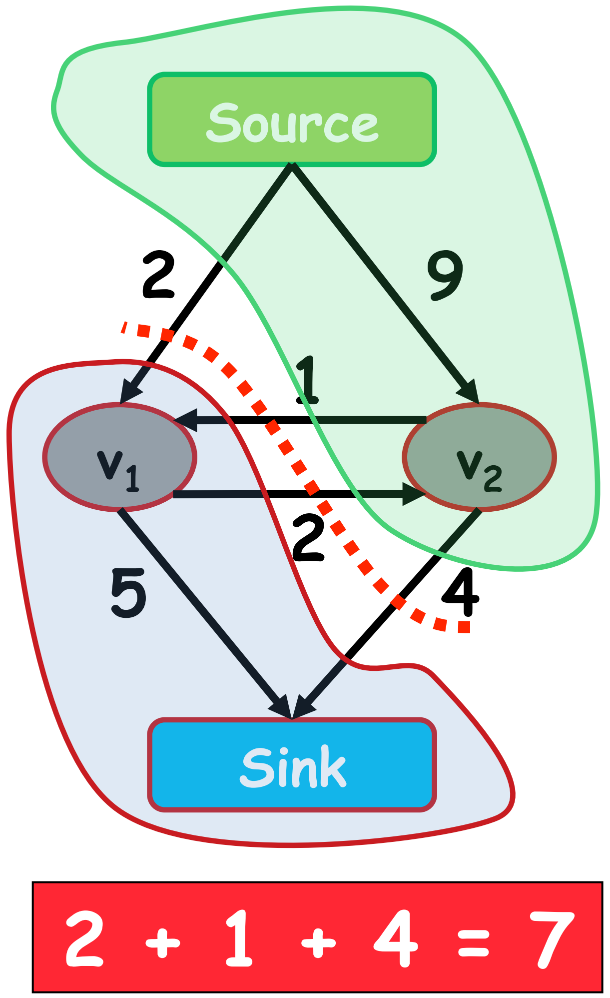
What is an st-cut?

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

The s-t-Mincut Problem



Slide credit: Pushmeet Kohli

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

History of Maxflow Algorithms

Augmenting Path and **Push-Relabel**

year	discoverer(s)	bound
1951	Dantzig	$O(n^2 mU)$
1955	Ford & Fulkerson	$O(m^2 U)$
1970	Dinitz	$O(n^2 m)$
1972	Edmonds & Karp	$O(m^2 \log U)$
1973	Dinitz	$O(nm \log U)$
1974	Karzanov	$O(n^3)$
1977	Cherkassky	$O(n^2 m^{1/2})$
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

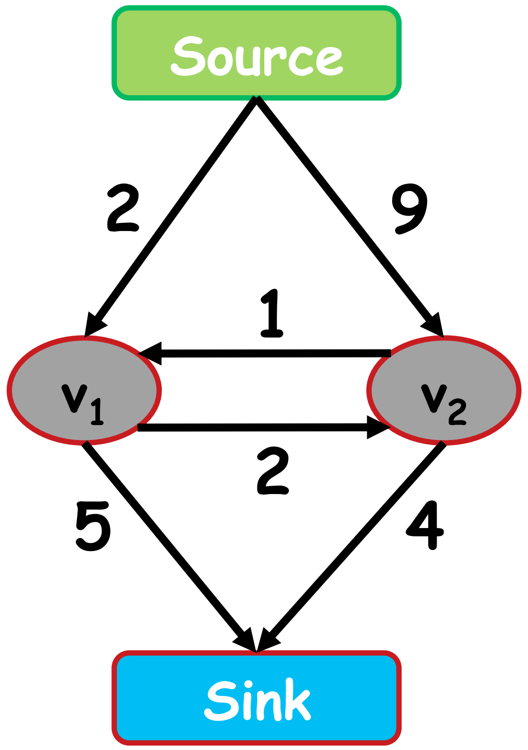
m: #edges

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

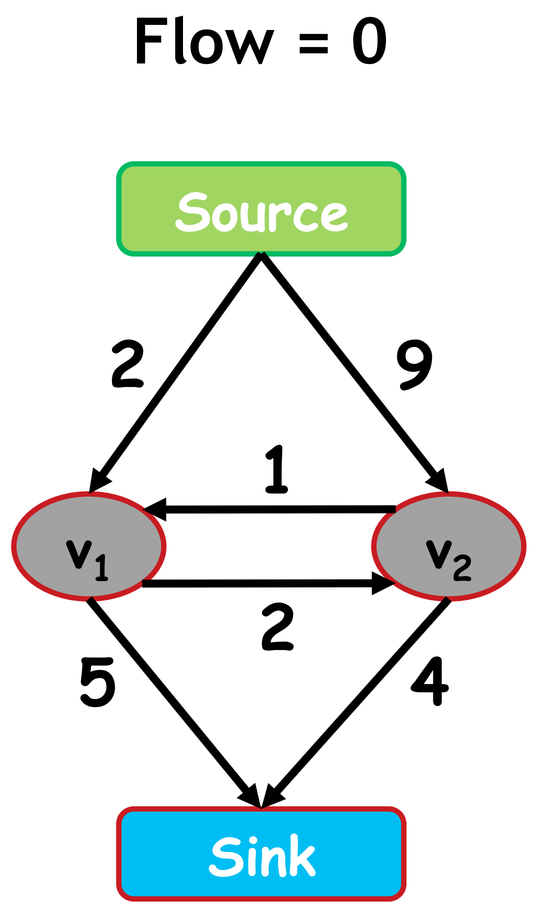
Nodes: Flow in = Flow out

Min-cut/Max-flow Theorem

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

Slide credit: Pushmeet Kohli

Maxflow Algorithms



Augmenting Path Based Algorithms

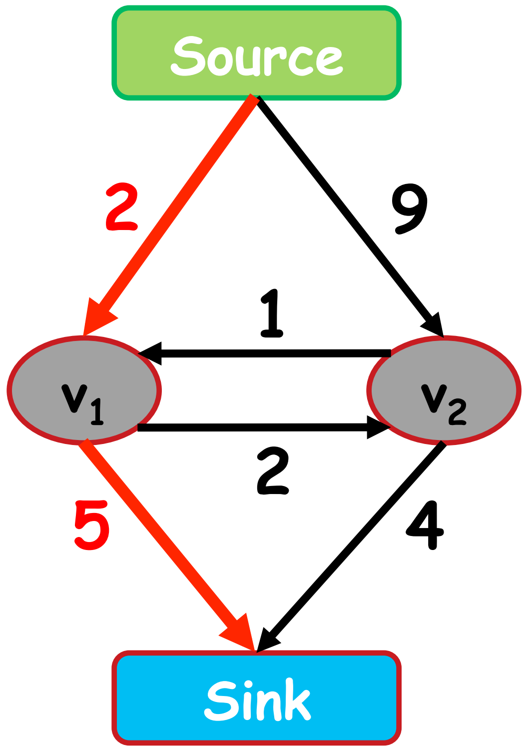
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 0



Augmenting Path Based Algorithms

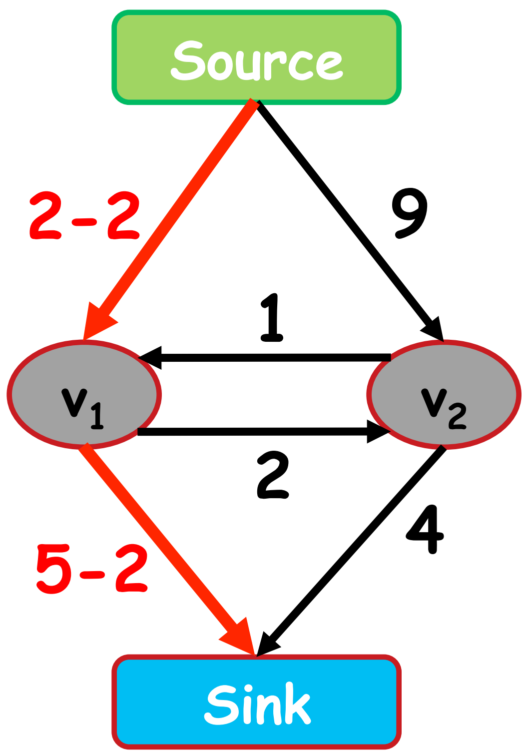
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 0 + 2



Augmenting Path Based Algorithms

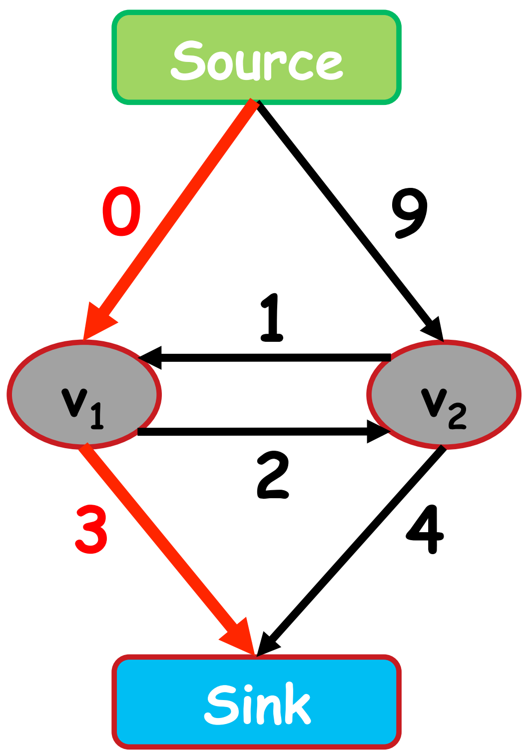
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 2



Augmenting Path Based Algorithms

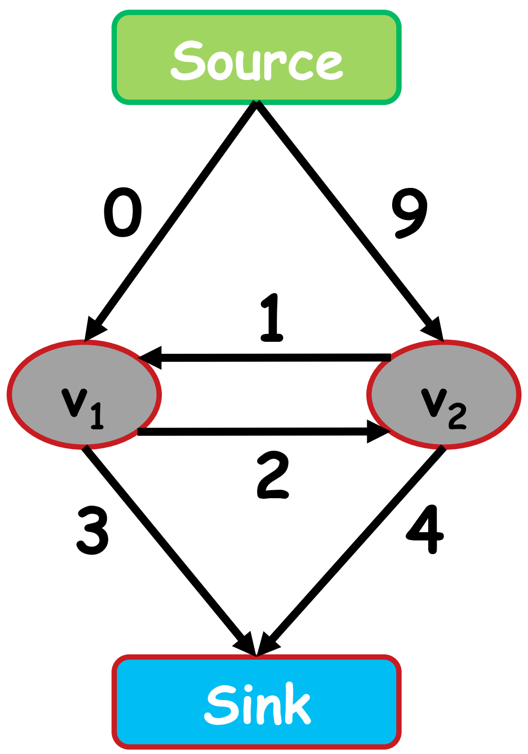
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 2



Augmenting Path Based Algorithms

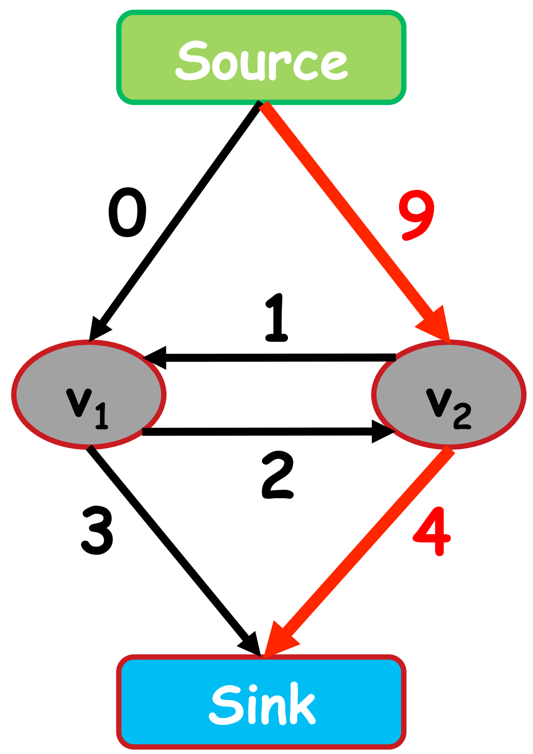
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 2



Augmenting Path Based Algorithms

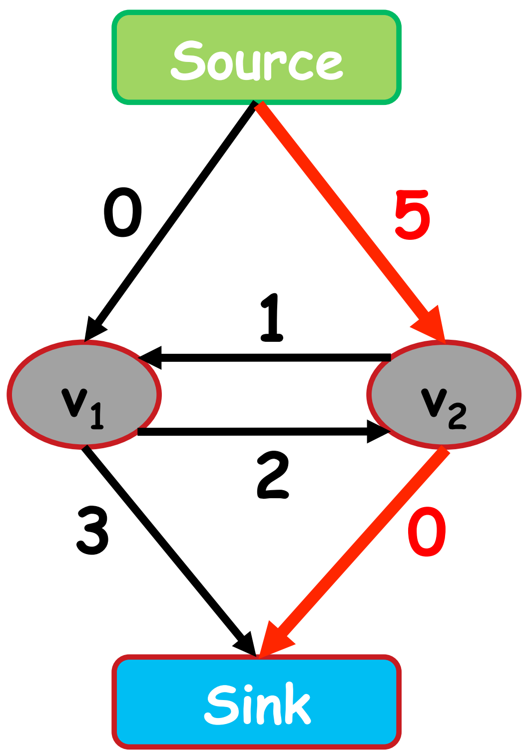
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 2 + 4



Augmenting Path Based Algorithms

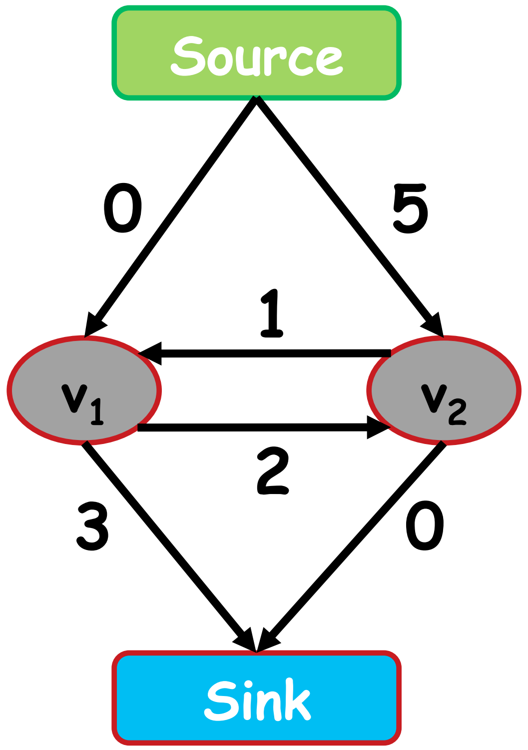
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 6



Augmenting Path Based Algorithms

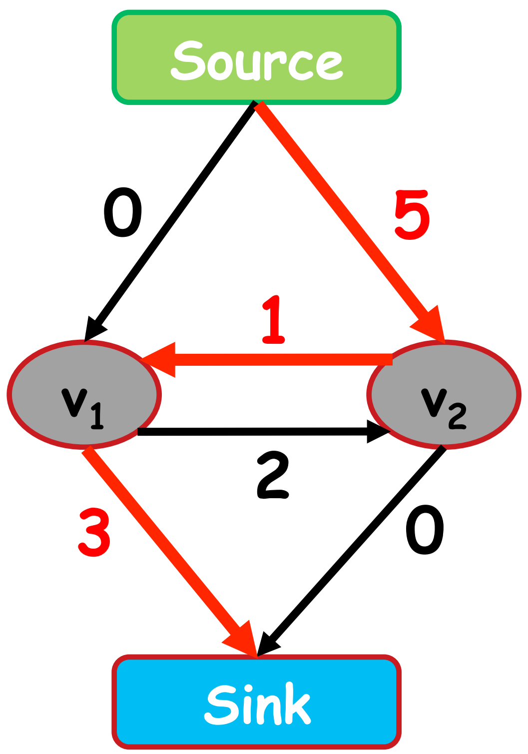
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 6



Augmenting Path Based Algorithms

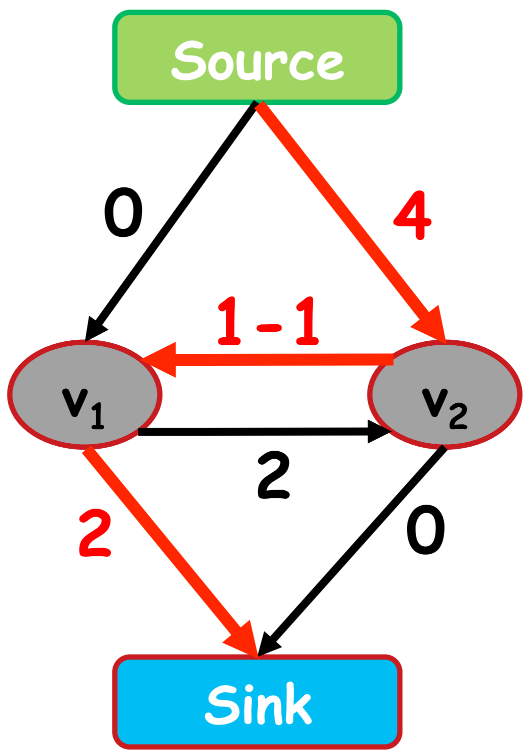
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 6 + 1



Augmenting Path Based Algorithms

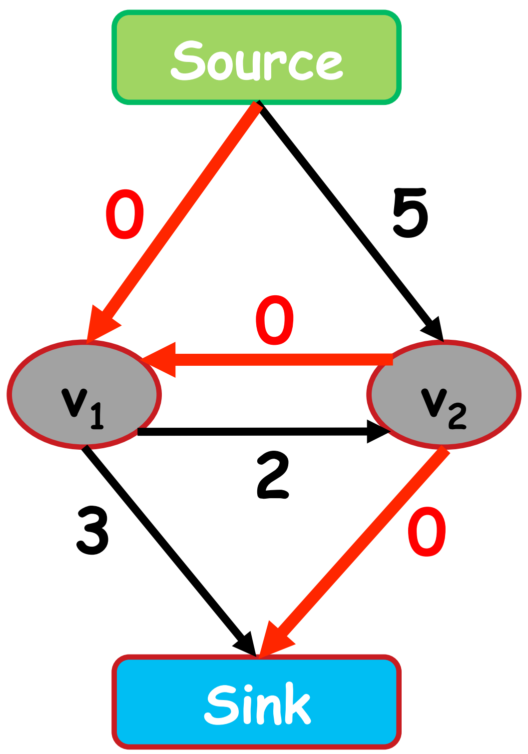
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 7



Augmenting Path Based Algorithms

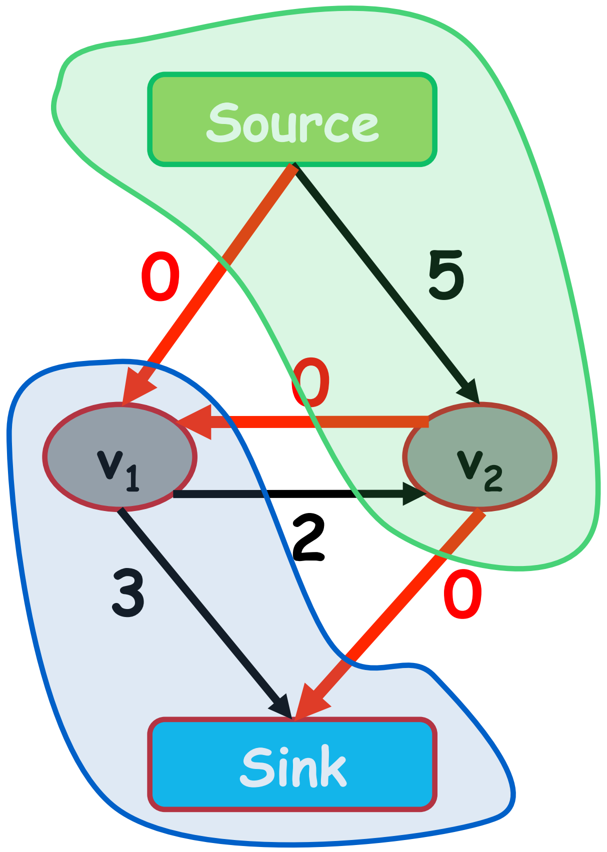
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

Maxflow Algorithms

Flow = 7



Augmenting Path Based Algorithms

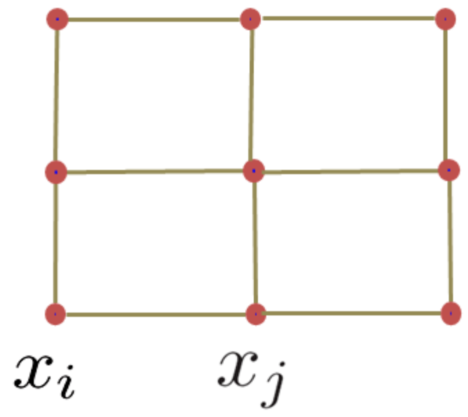
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

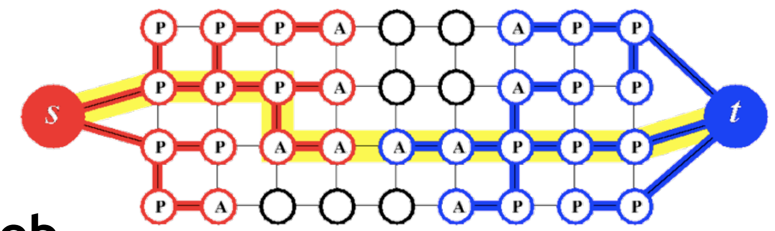
Slide credit: Pushmeet Kohli

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

$$E(L) = \sum_{p} E_p(L_p) + \sum_{pq \in N} E(L_p, L_q)$$

t-links | n-links

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

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

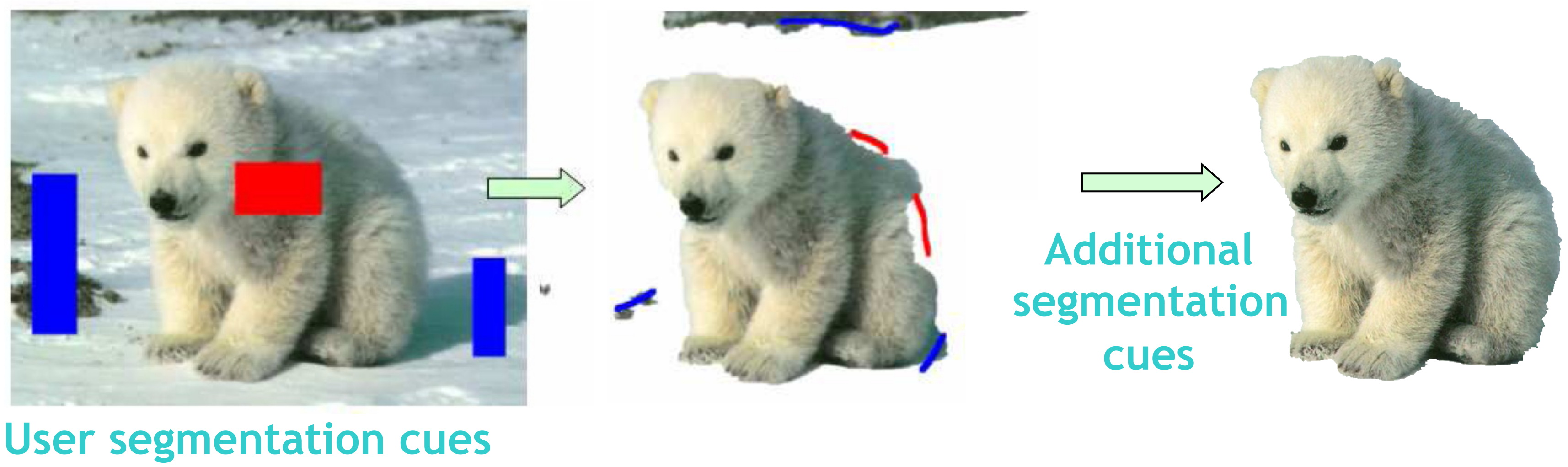
E(L) can be minimized by *s-t* graph cuts	$\Leftrightarrow$	$E(s,s) + E(t,t) \leq E(s,t) + E(t,s)$
		Submodularity ("convexity")

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

Slide credit: B. Leibe

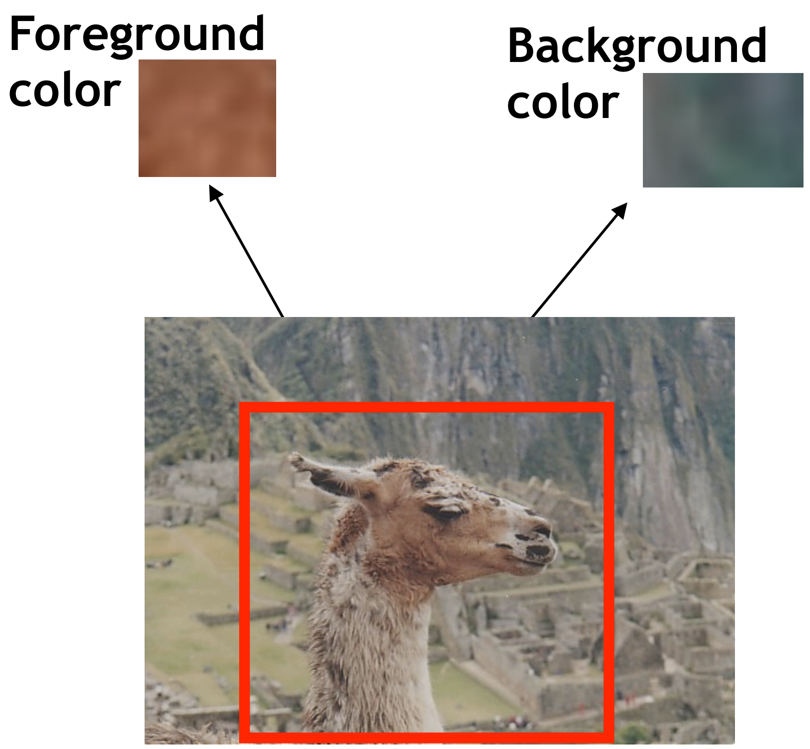
GraphCut Applications: "GrabCut"

- Interactive Image Segmentation [Boykov & Jolly, ICCV' 01]
  - Rough region cues sufficient
  - Segmentation boundary can be extracted from edges
- Procedure
  - User marks foreground and background regions with a brush → get initial segmentation → correct by additional brush strokes



Slide adapted from Matthieu Bray

GrabCut: Data Model



Global optimum of
the unary energy

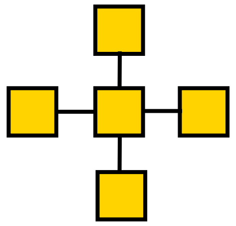
- **Obtained from interactive user input**
  - **User marks foreground and background regions with a brush**
  - **Alternatively, user can specify a bounding box**

Slide adapted from Carsten Rother

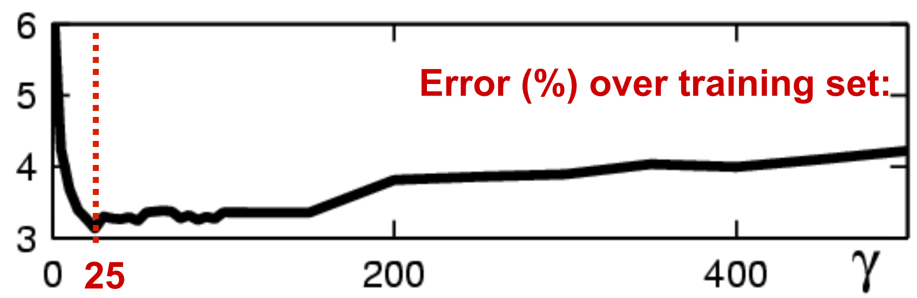
GrabCut: Coherence Model

- **An object is a coherent set of pixels:**

$$\psi(x,y) = \gamma \sum_{(m,n)\in C} \delta\left[x_n \neq x_m\right] e^{-\beta \|y_m - y_n\|^2}$$



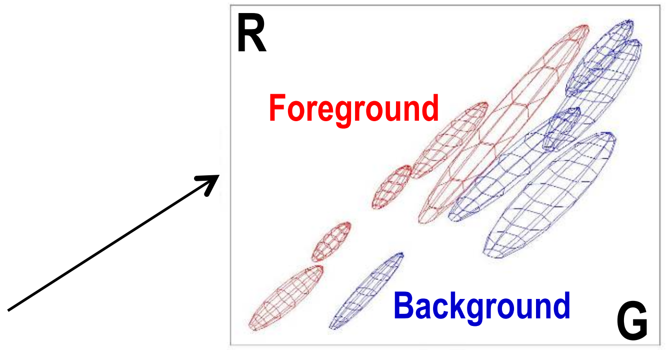
How to choose γ ?



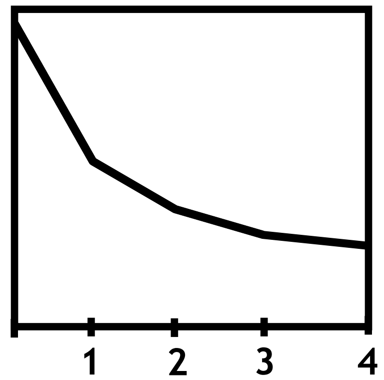
Slide credit: Carsten Rother

Iterated Graph Cuts

Result



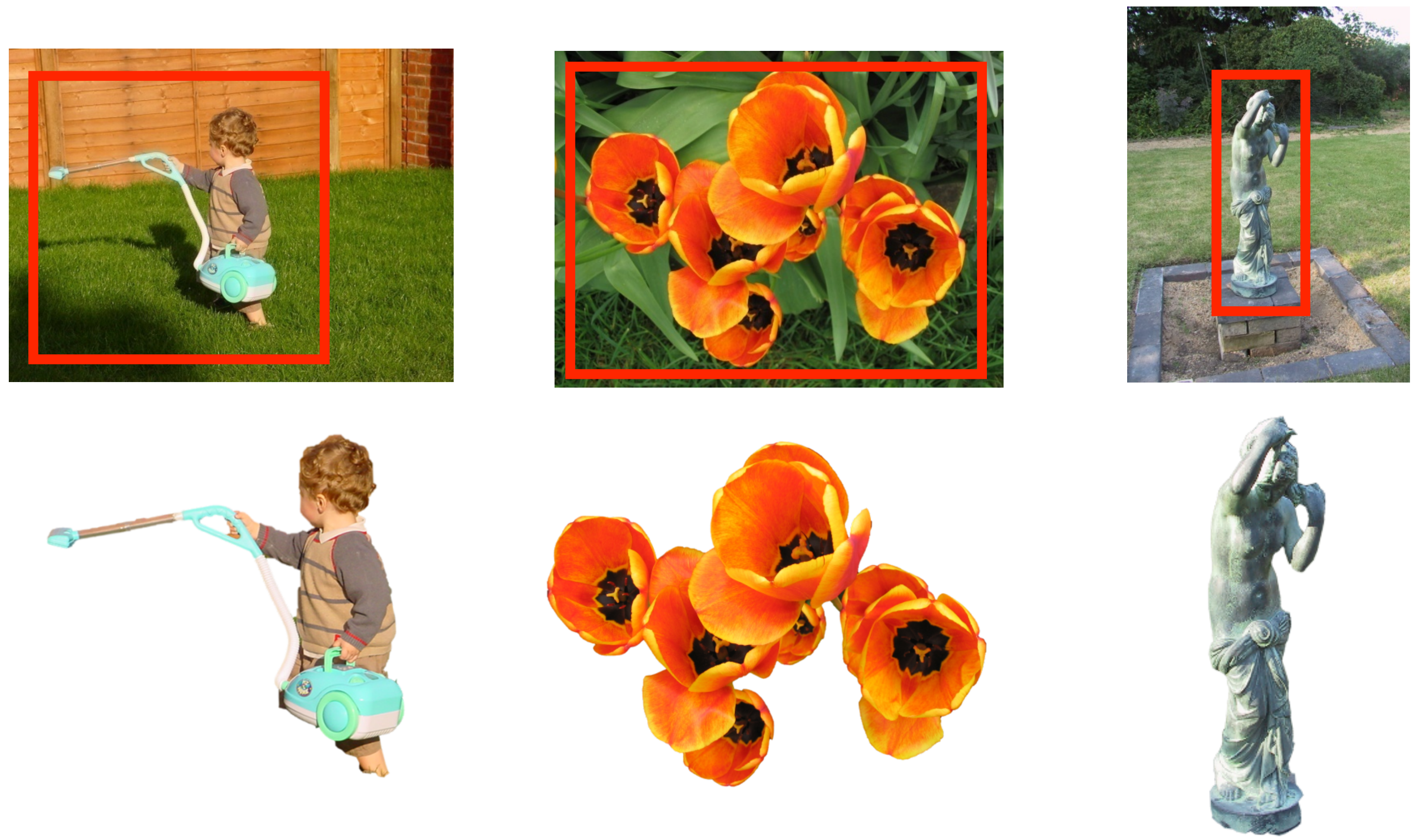
Color model
(Mixture of Gaussians)



Energy after
each iteration

Slide credit: Carsten Rother

GrabCut: Example Results



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

Slide credit: B. Leibe

Summary

Introduction
- Gestalt principles
- Image segmentation

Segmentation as clustering
- k-Means
- Feature spaces

Model-free clustering: Mean-Shift

Interactive Segmentation with GraphCuts

Reading: F+P chapter 9; Sz 5.3, 5.5