Computer Vision – Lecture 6

Segmentation

09.11.2016

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

Course Outline

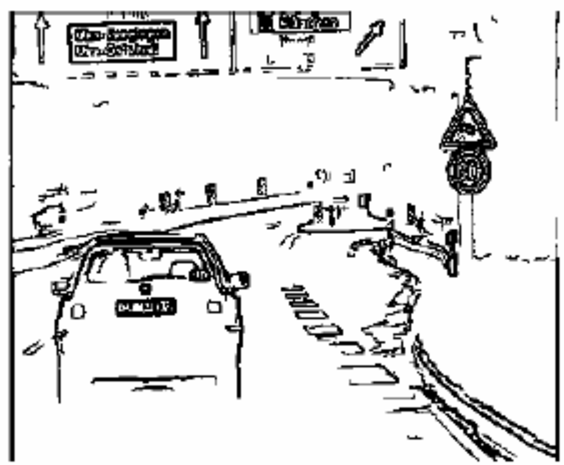
- Image Processing Basics
  - Structure Extraction
- Segmentation
  - Segmentation as Clustering
  - Graph-theoretic Segmentation
- Recognition
  - Global Representations
  - Subspace representations
- Local Features & Matching
- Object Categorization
- 3D Reconstruction

Recap: Chamfer Matching

- Chamfer Distance
  - Average distance to nearest feature

$$D_{chamfer}(T, I) \equiv \frac{1}{|T|} \sum_{t \in T} d_I(t)$$

  - This can be computed efficiently by correlating the edge template with the distance-transformed image



Edge image

Distance transform image

[D. Gavrila, DAGM'99]

Recap: Hough Transform

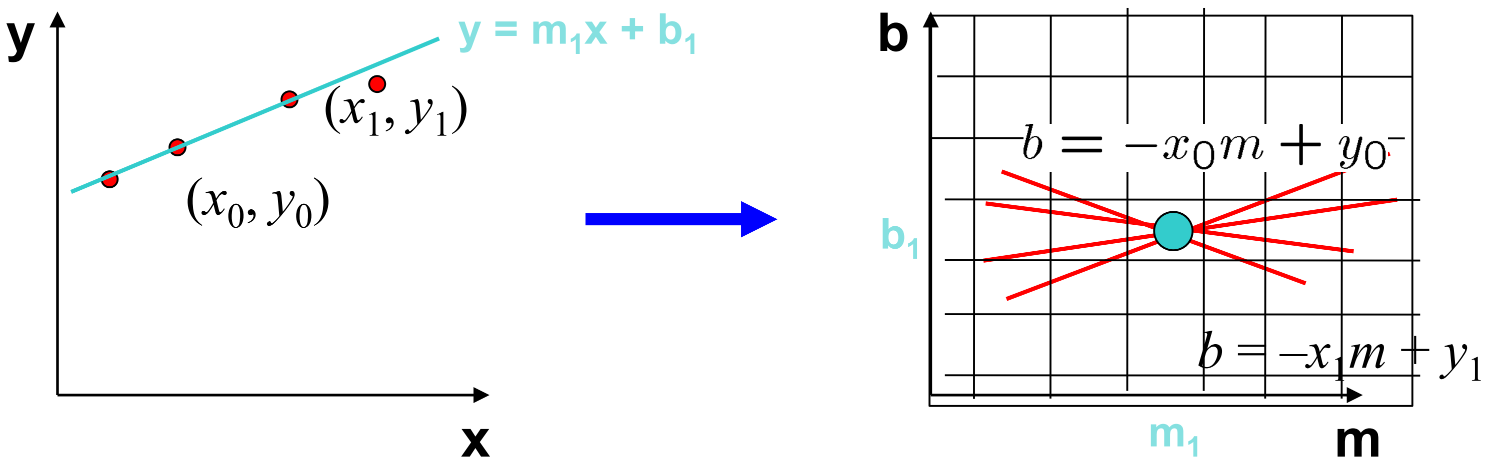


Image space

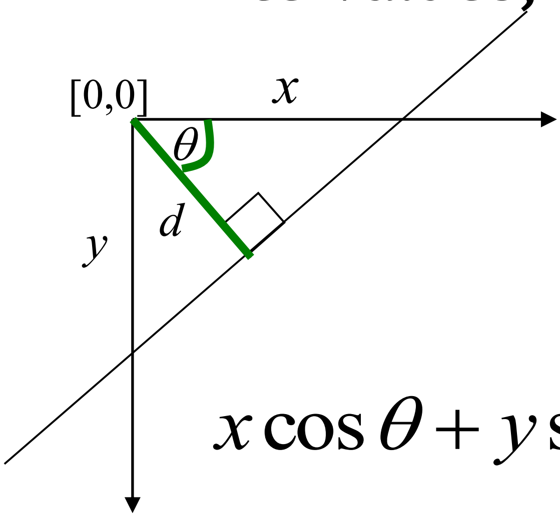
Hough (parameter) space

- How can we use this to find the most likely parameters (m,b) for the most prominent line in the image space?
  - Let each edge point in image space *vote* for a set of possible parameters in Hough space
  - Accumulate votes in discrete set of bins; parameters with the most votes indicate line in image space.

Slide credit: Steve Seitz

Recap: Hough Transf. Polar Parametrization

- Usual (m,b) parameter space problematic: can take on infinite values, undefined for vertical lines.

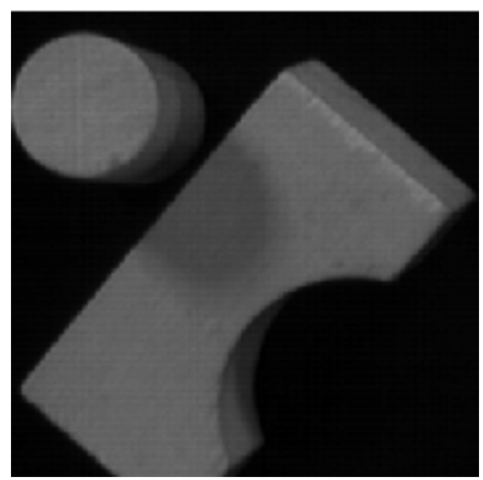


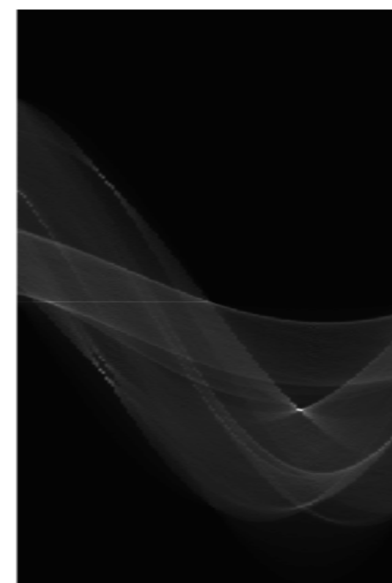
$$x\cos\theta + y\sin\theta = d$$

d : perpendicular distance from line to origin

θ : angle the perpendicular makes with the x-axis

- Point in image space $\Rightarrow$ sinusoid segment in Hough space





Slide credit: Steve Seitz

Recap: Hough Transform for Circles

- **Circle: center (a,b) and radius r**

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

- **For an unknown radius r, unknown gradient direction**

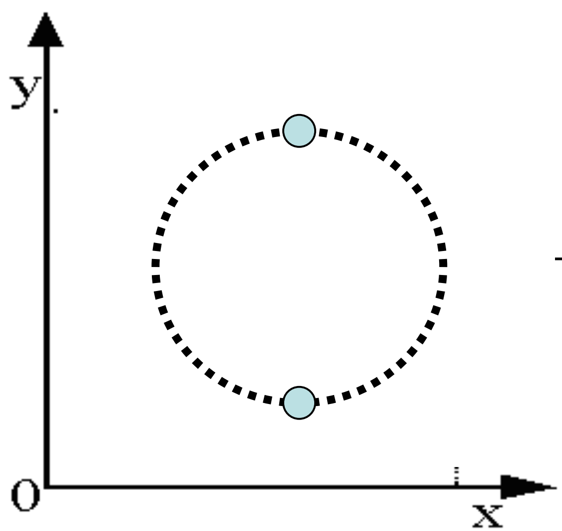
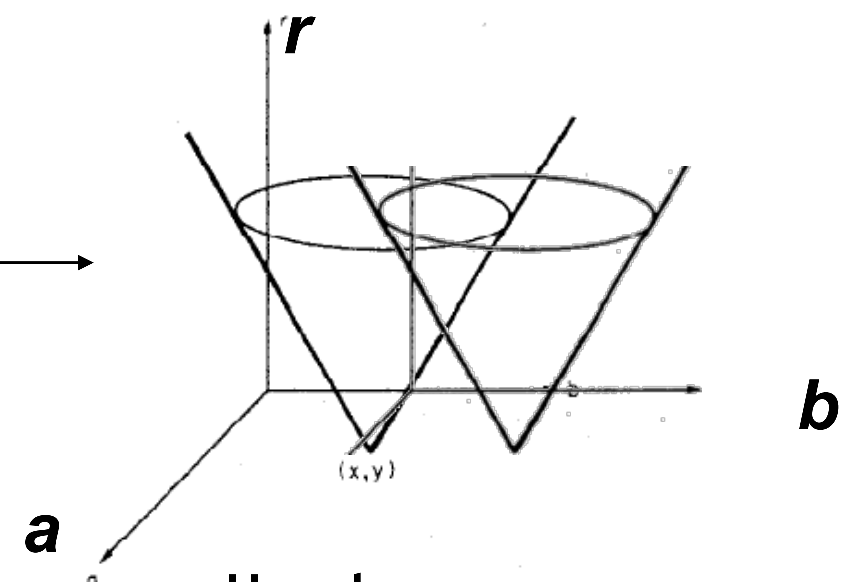


Image space



Hough space

Slide credit: Kristen Grauman

Generalized Hough Transform

- **What if we want to detect arbitrary shapes defined by boundary points and a reference point?**

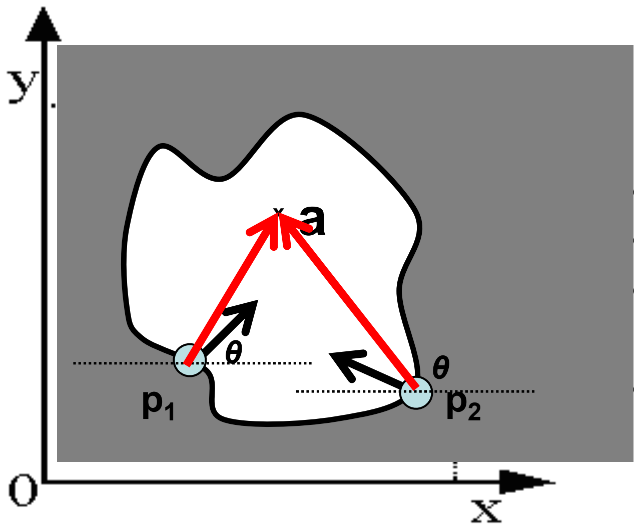


Image space

At each boundary point, compute displacement vector: $r = a - p_i$.

For a given model shape: store these vectors in a table indexed by gradient orientation θ.

[Dana H. Ballard, Generalizing the Hough Transform to Detect Arbitrary Shapes, 1980]

Generalized Hough Transform

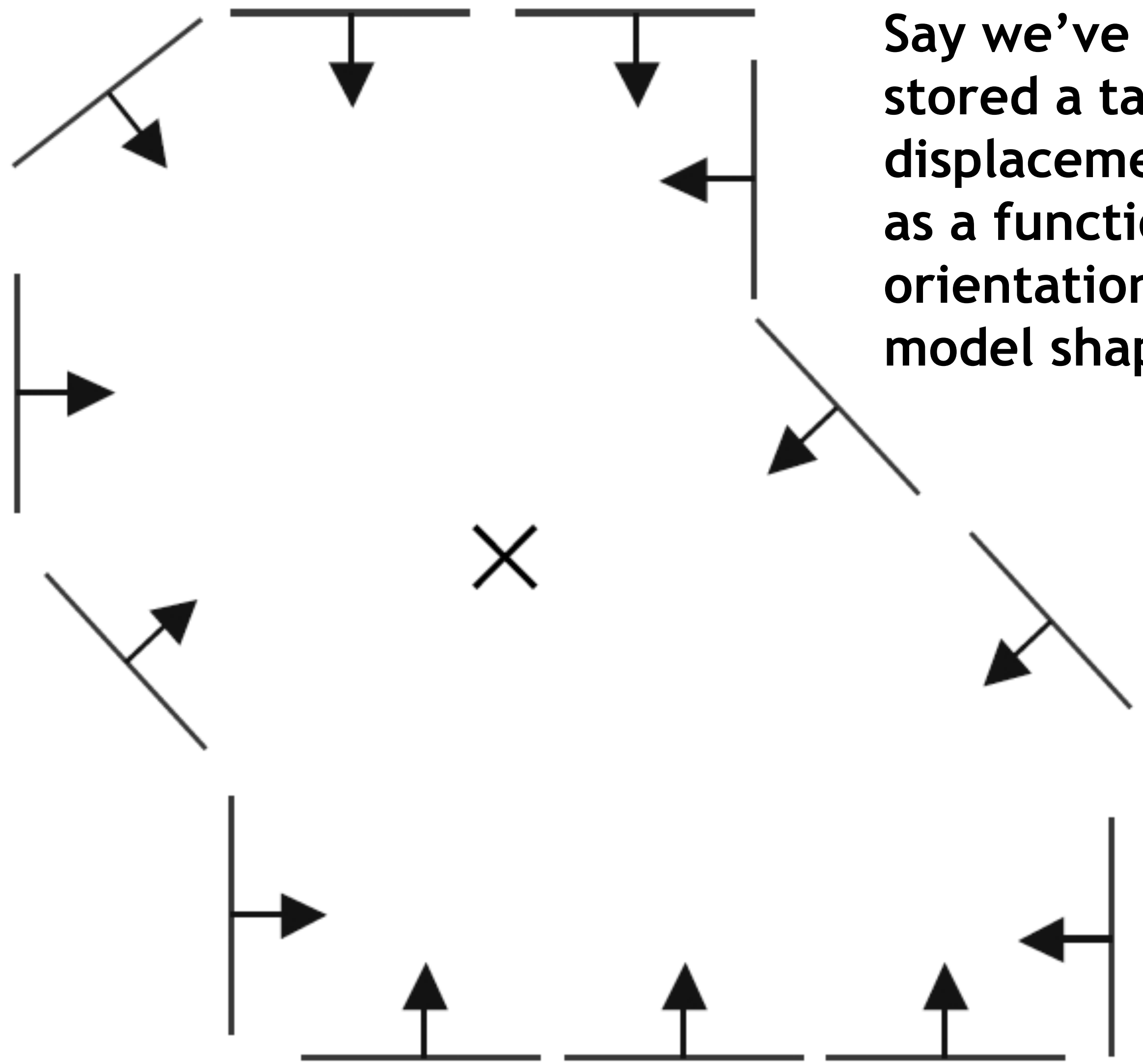
To *detect* the model shape in a new image:

- For each edge point
  - Index into table with its gradient orientation θ
  - Use retrieved r vectors to vote for position of reference point
- Peak in this Hough space is reference point with most supporting edges

Assuming translation is the only transformation here, i.e., orientation and scale are fixed.

Example: Generalized Hough Transform

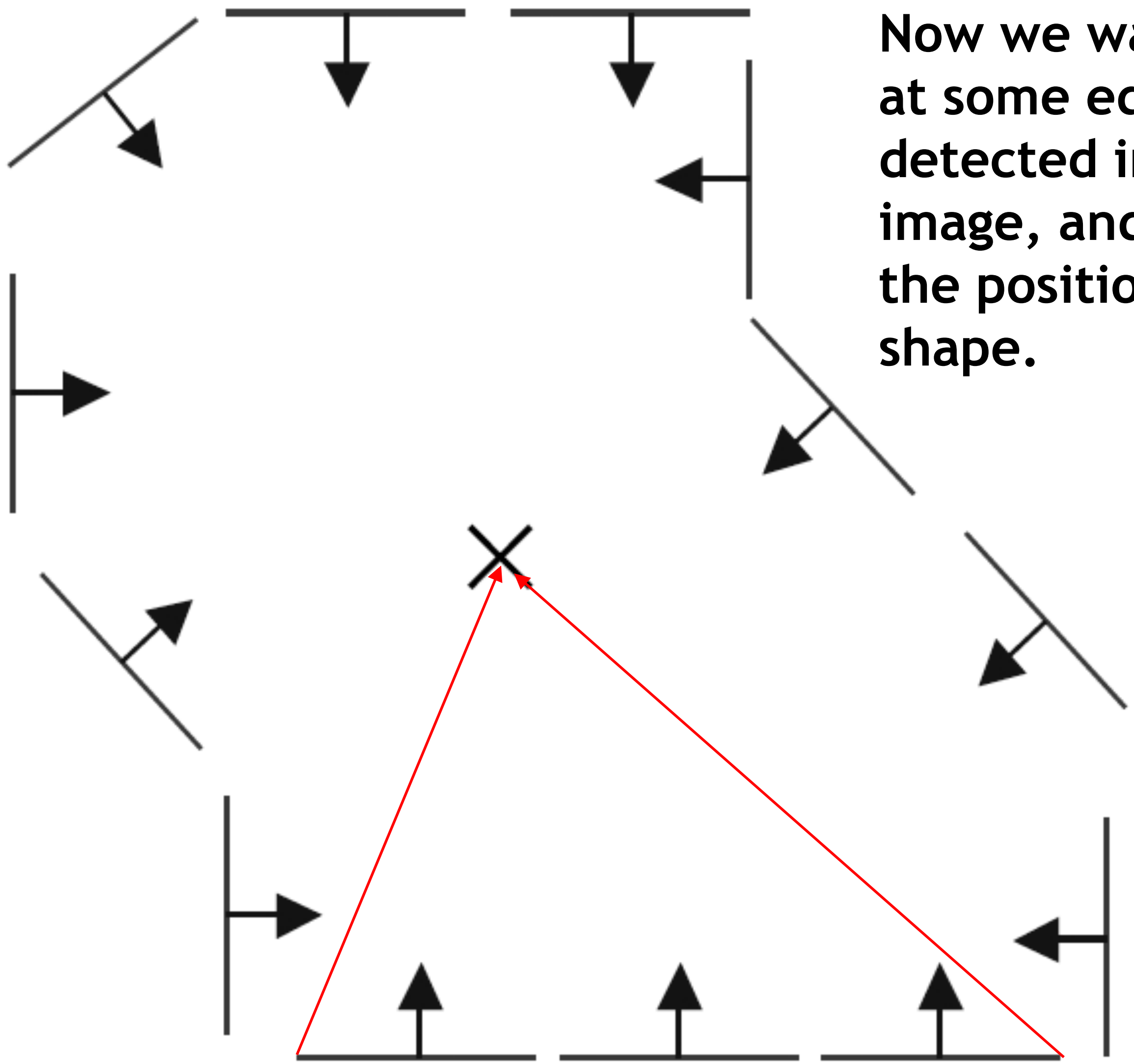
Say we've already stored a table of displacement vectors as a function of edge orientation for this model shape.



Model shape

Slide credit: Svetlana Lazebnik

Example: Generalized Hough Transform

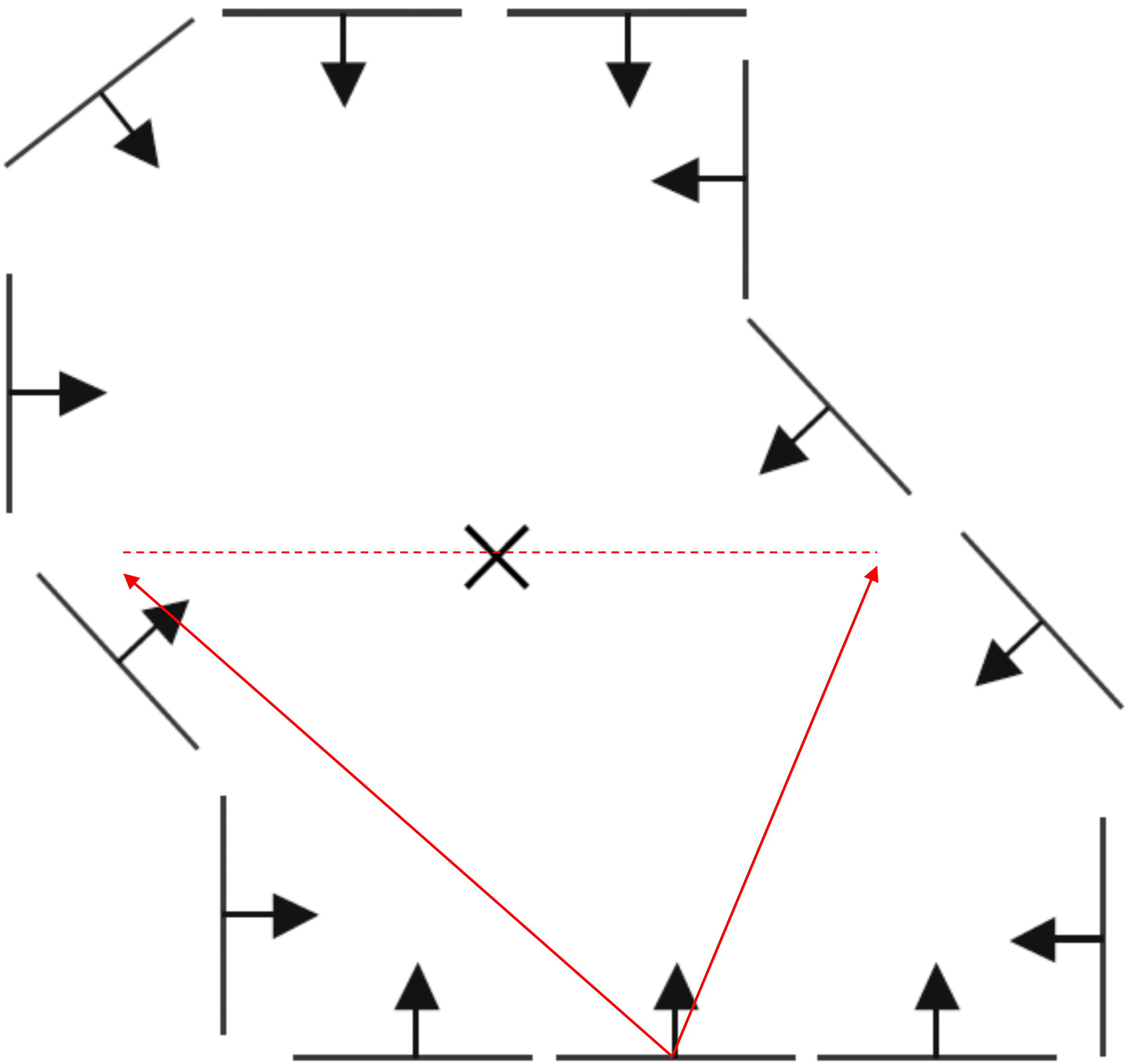


Now we want to look at some edge points detected in a *new* image, and vote on the position of that shape.

Displacement vectors for model points

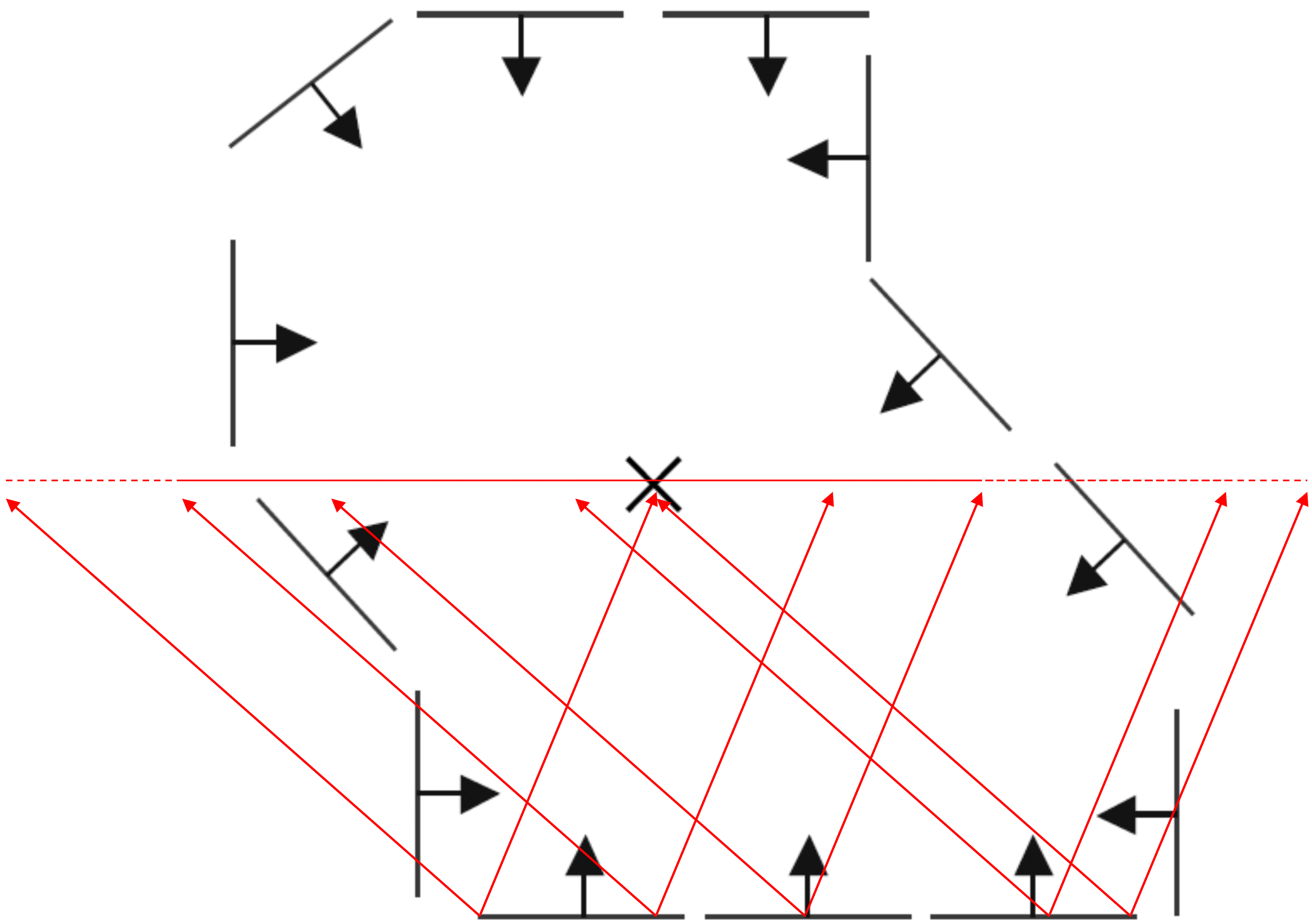
Slide credit: Svetlana Lazebnik

Example: Generalized Hough Transform



Range of voting locations for test point

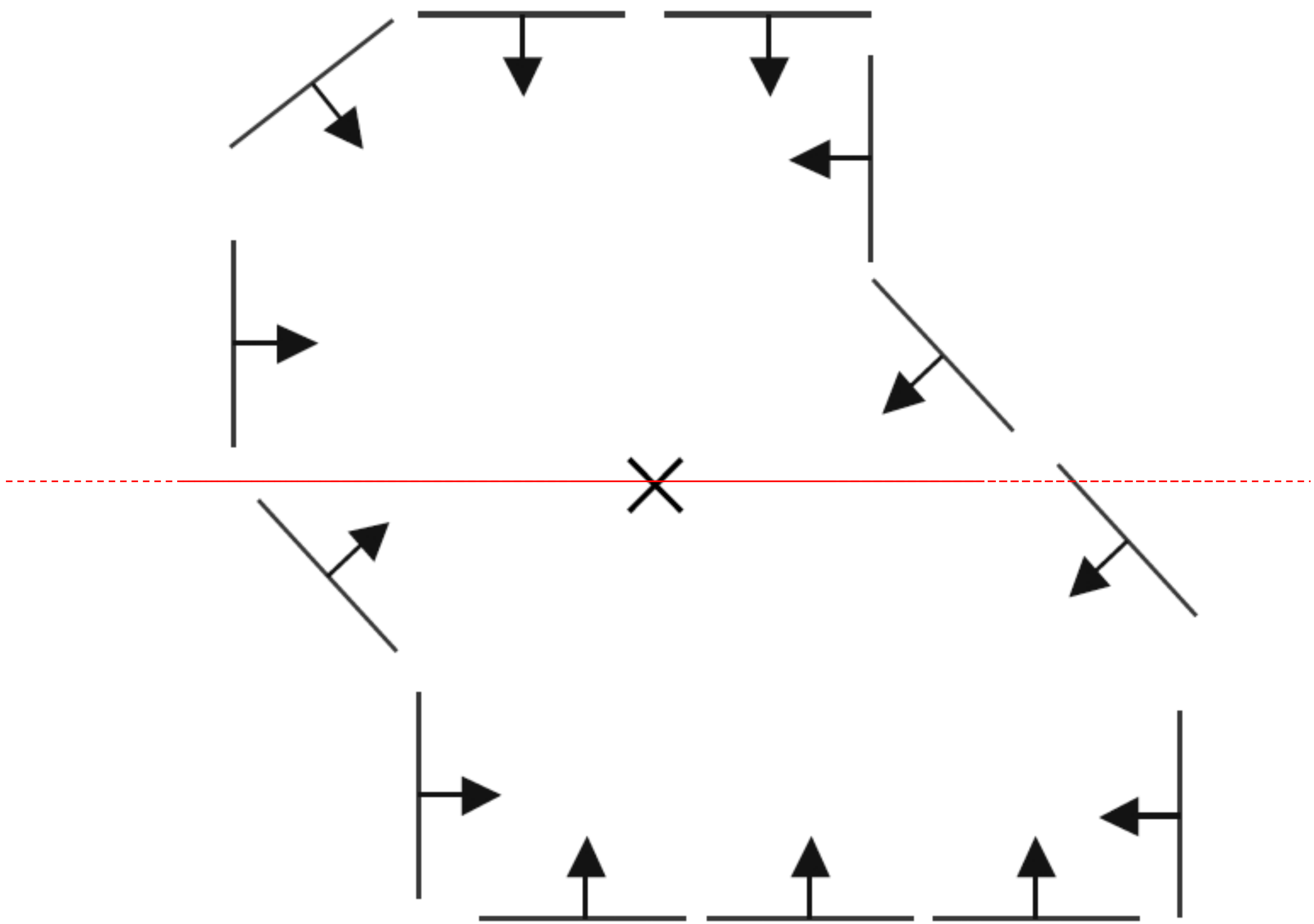
Example: Generalized Hough Transform



Range of voting locations for test point

Slide credit: Svetlana Lazebnik

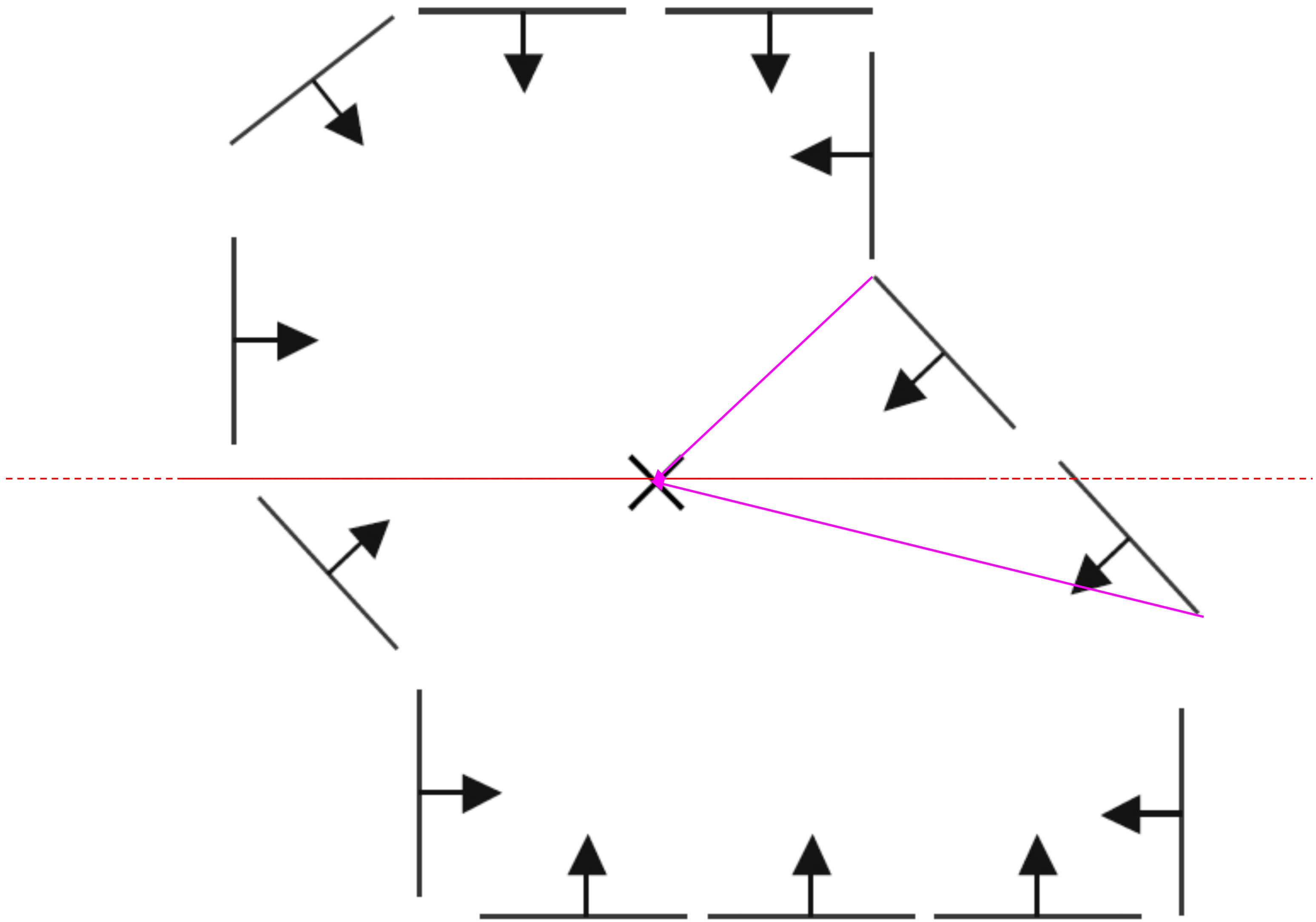
Example: Generalized Hough Transform



Votes for points with $\theta = \uparrow$

Slide credit: Svetlana Lazebnik

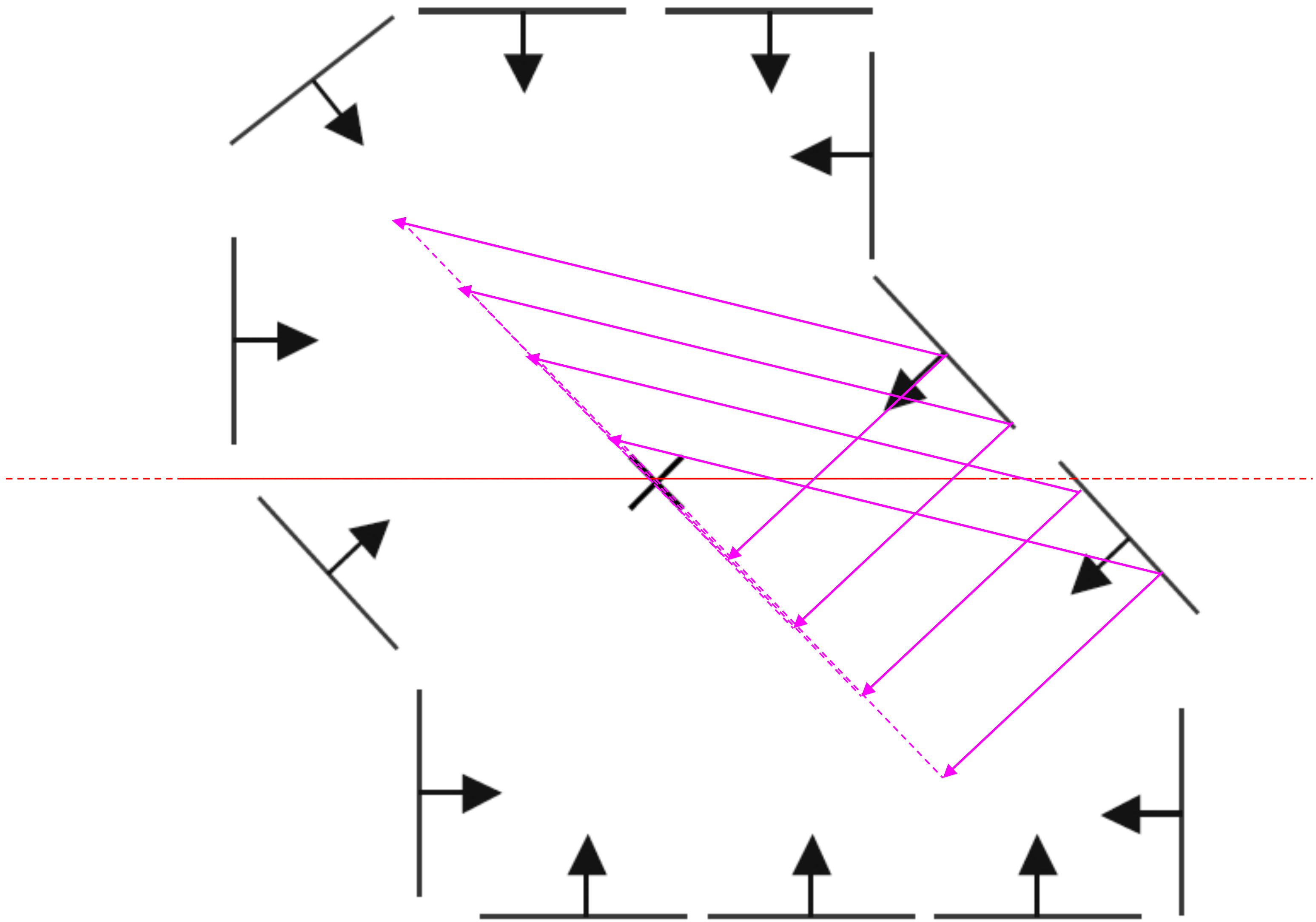
Example: Generalized Hough Transform



Displacement vectors for model points

Slide credit: Svetlana Lazebnik

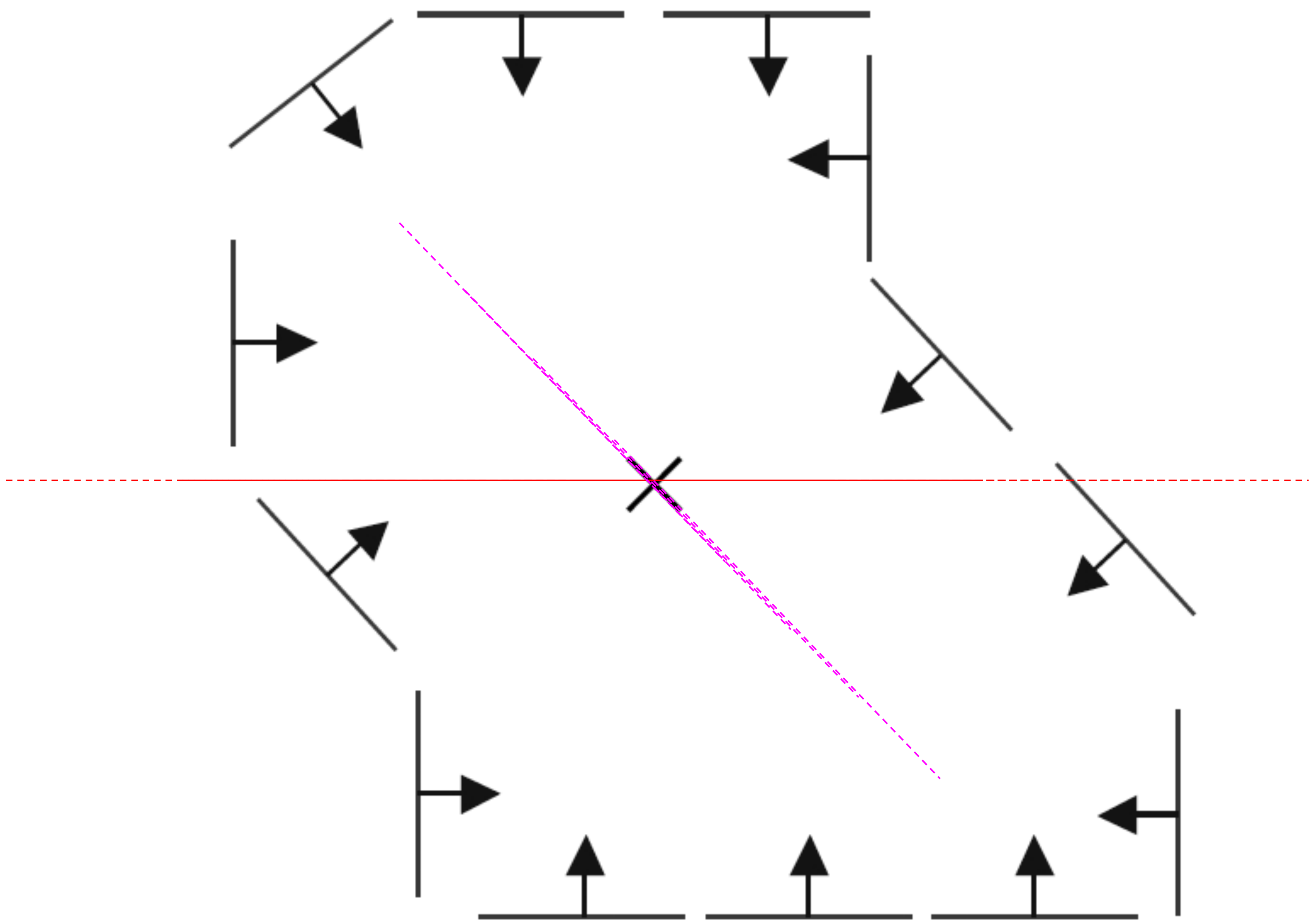
Example: Generalized Hough Transform



Range of voting locations for test point

Slide credit: Svetlana Lazebnik

Example: Generalized Hough Transform

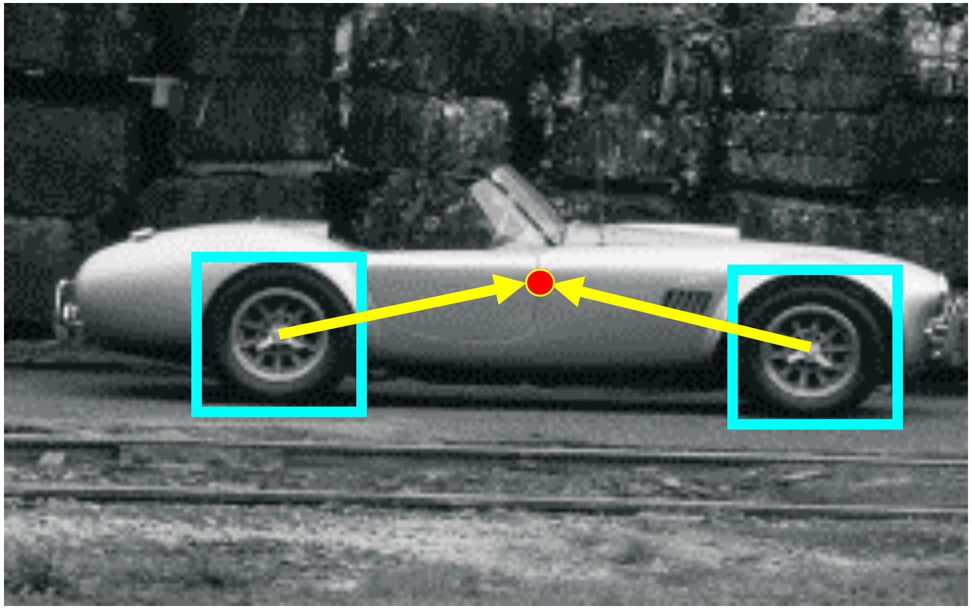


Votes for points with θ =

Slide credit: Svetlana Lazebnik

Application in Recognition

- Instead of indexing displacements by gradient orientation, index by "visual codeword".



Training image

Visual codeword with displacement vectors

B. Leibe, A. Leonardis, and B. Schiele, Robust Object Detection with Interleaved Categorization and Segmentation, International Journal of Computer Vision, Vol. 77(1-3), 2008.

Application in Recognition

- Instead of indexing displacements by gradient orientation, index by "visual codeword".

Test image

- We'll hear more about this in later lectures...

Topics of This Lecture

- Segmentation and grouping
  - Gestalt principles
  - Image Segmentation
- Segmentation as clustering
  - k-Means
  - Feature spaces
- Probabilistic clustering
  - Mixture of Gaussians, EM
- Model-free clustering
  - Mean-Shift clustering

Examples of Grouping in Vision

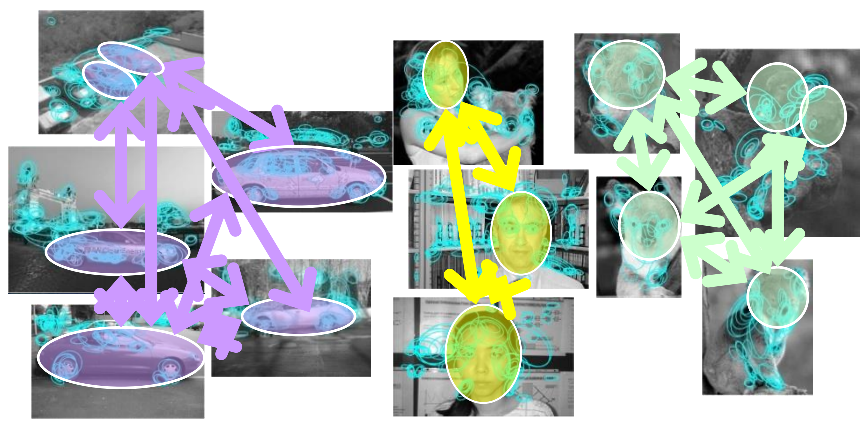
Determining image regions

Grouping video frames into shots

Figure-ground

What things should be grouped?

What cues indicate groups?



Object-level grouping

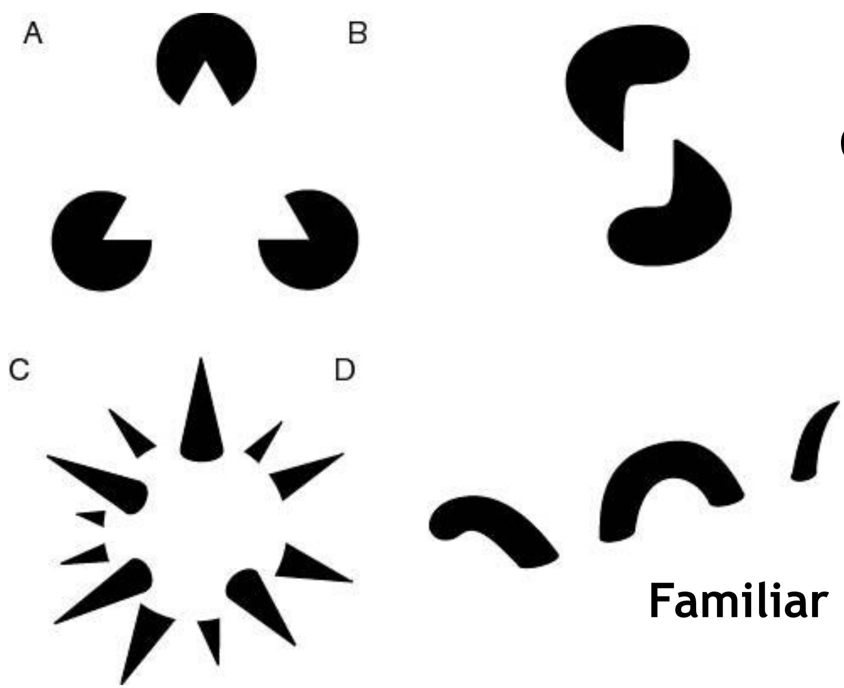
The Gestalt School

Grouped objects should have some inherent relationships b/w them, implicit or explicit

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

Illusory/subjective contours

Occlusion



Familiar configuration

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

Slide credit: Svetlana Lazebnik

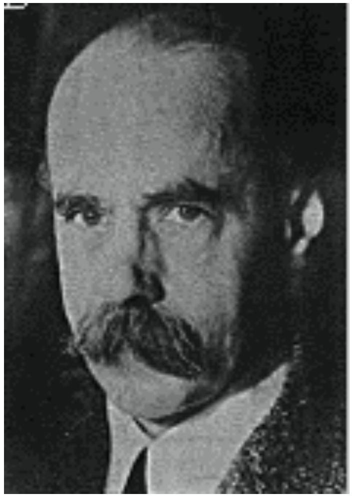
Image source: Steve Lehar

Gestalt 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)

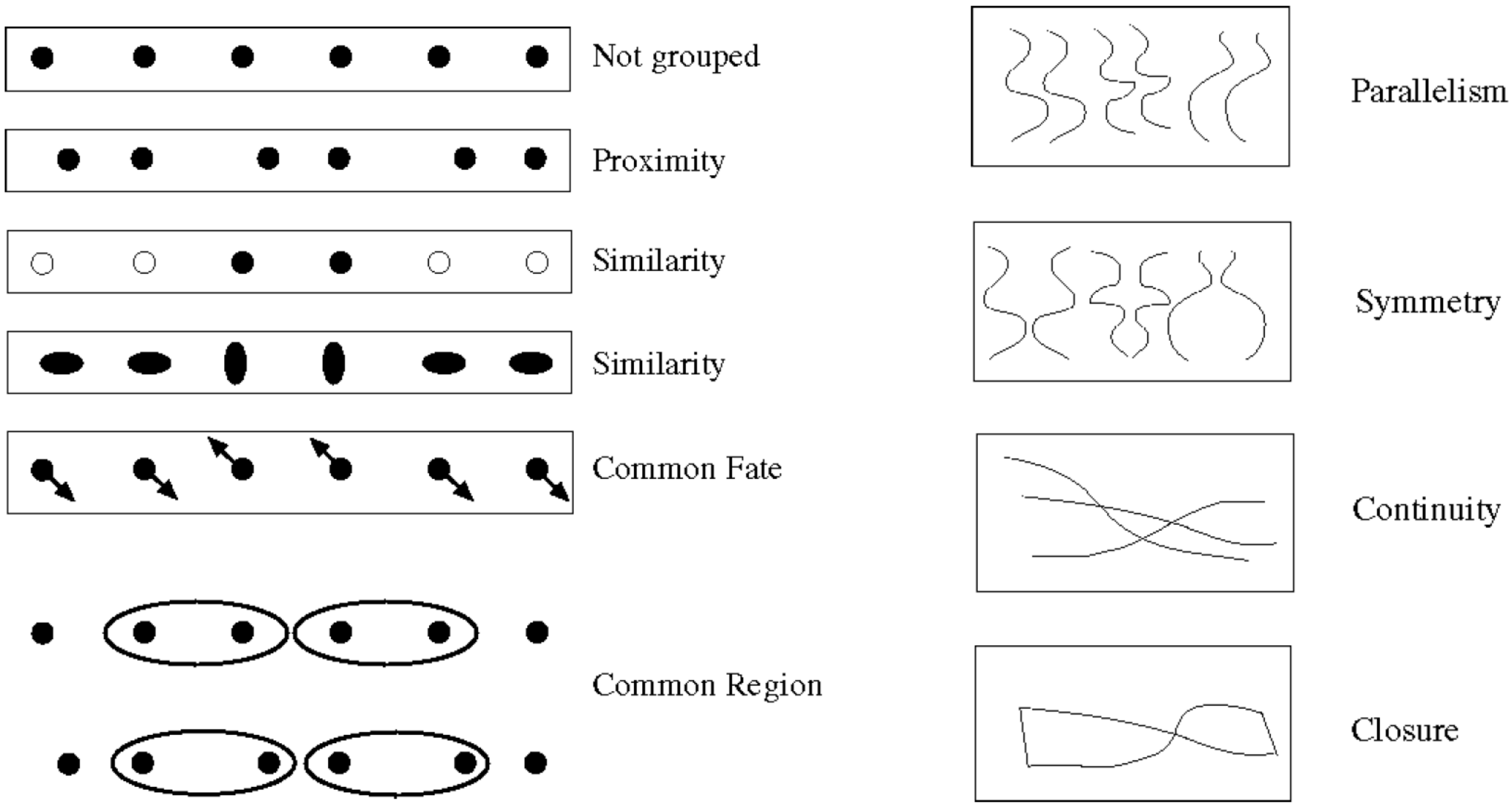
"I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have "327"? No. I have sky, house, and trees."

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

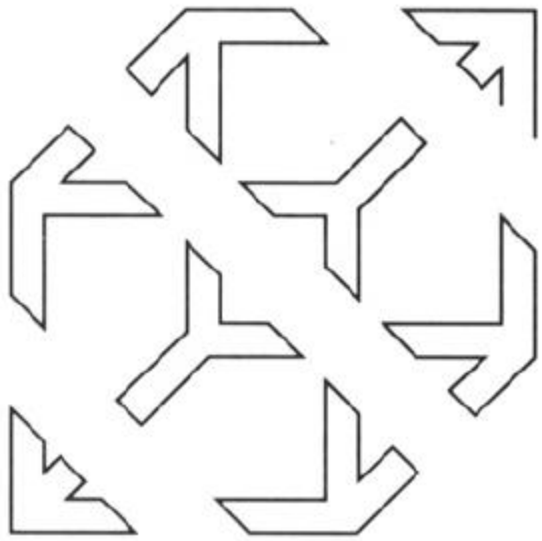
Gestalt Factors



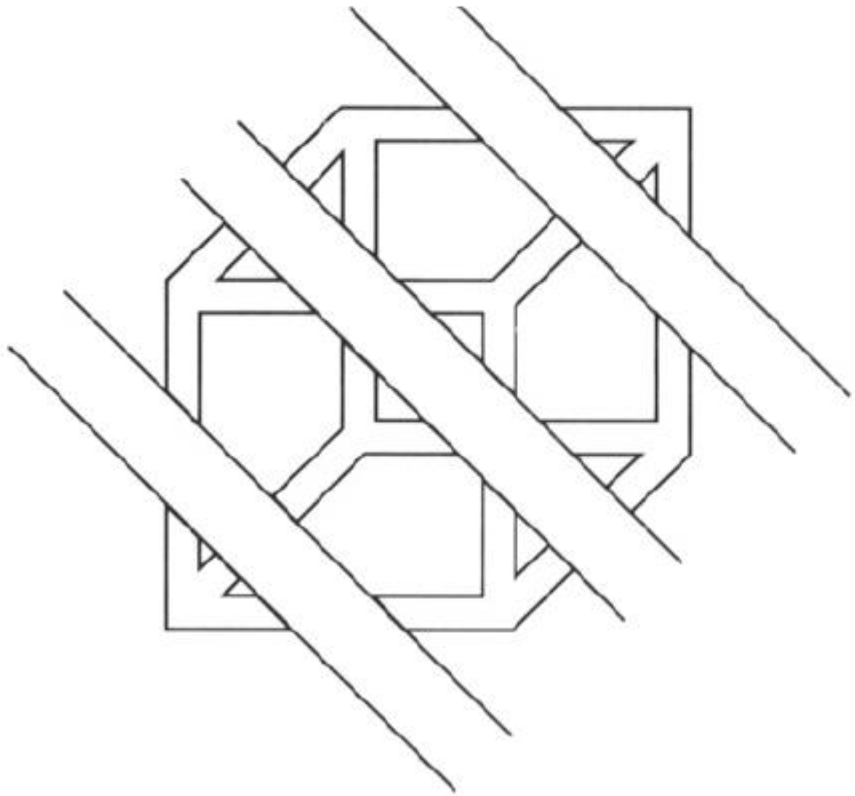
- **These factors make intuitive sense, but are very difficult to translate into algorithms.**

Image source: Forsyth & Ponce

Continuity through Occlusion Cues



Continuity through Occlusion Cues



Continuity, explanation by occlusion

Continuity through Occlusion Cues

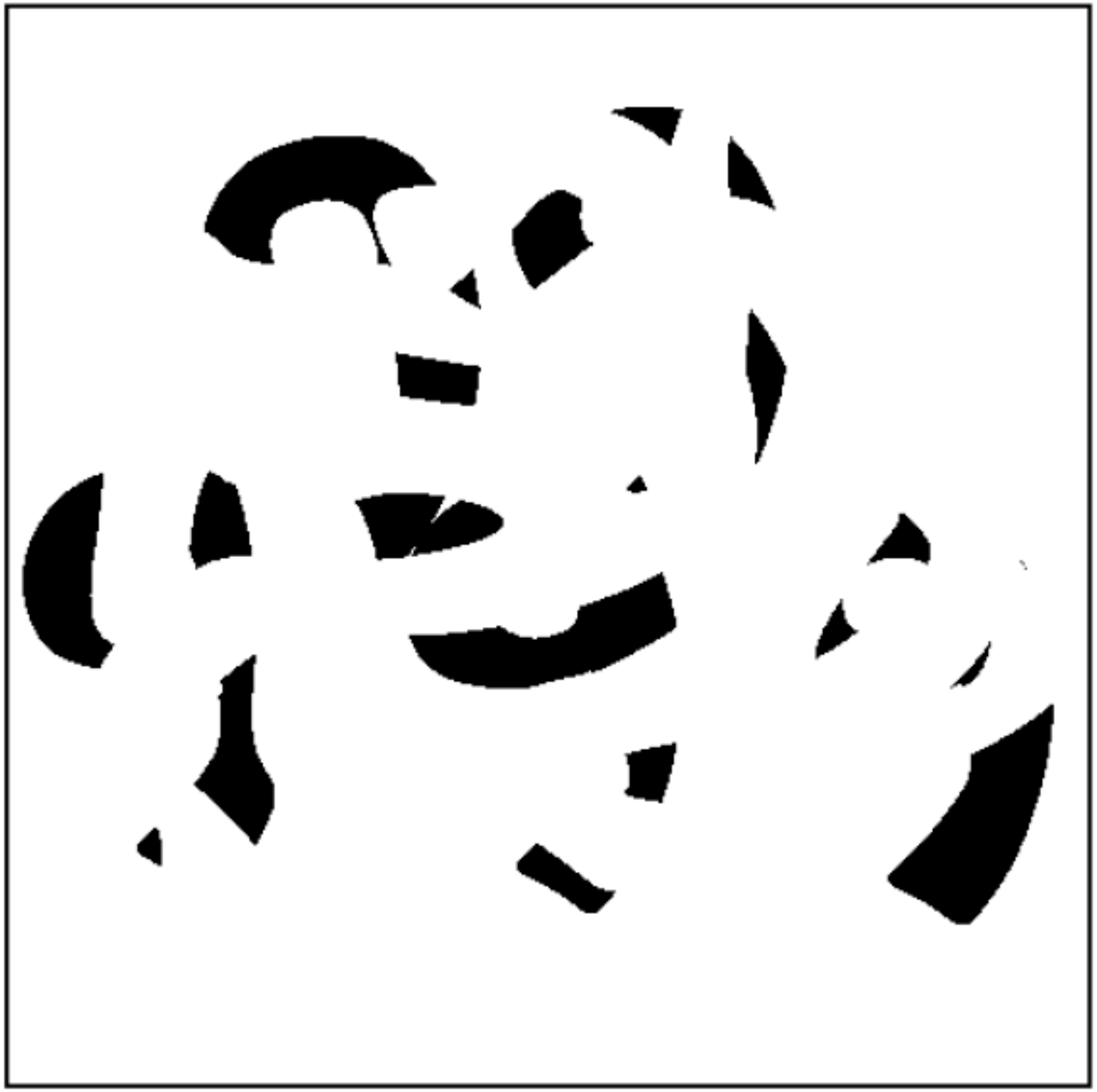


Image source: Forsyth & Ponce

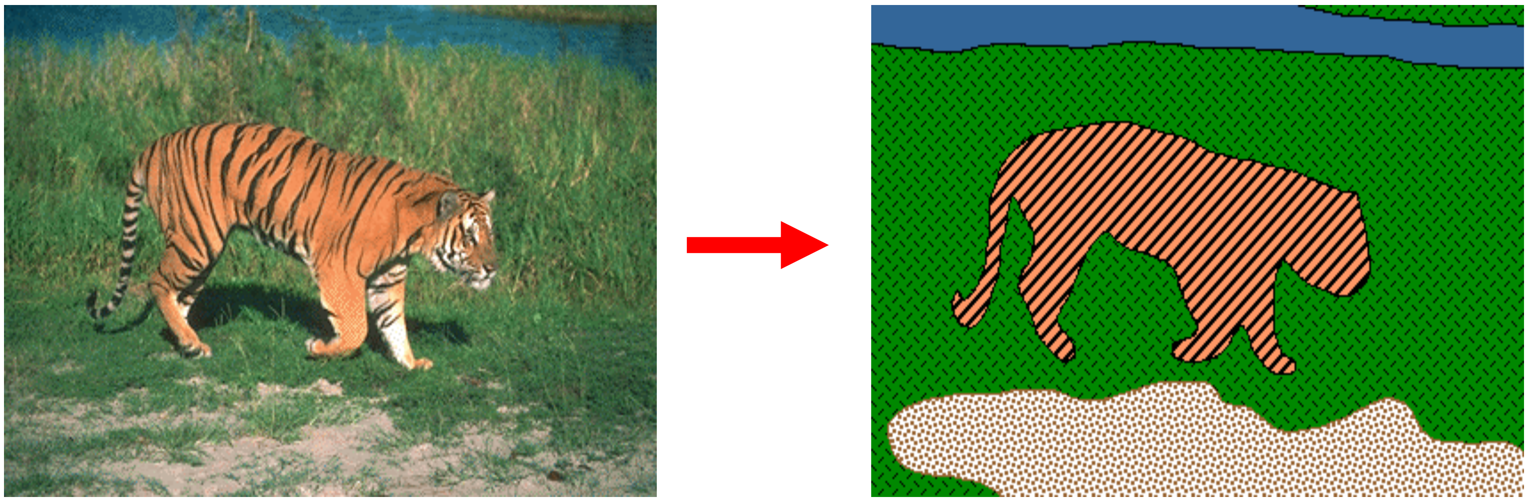
Continuity through Occlusion Cues

Image source: Forsyth & Ponce

The Ultimate Gestalt?

Image Segmentation

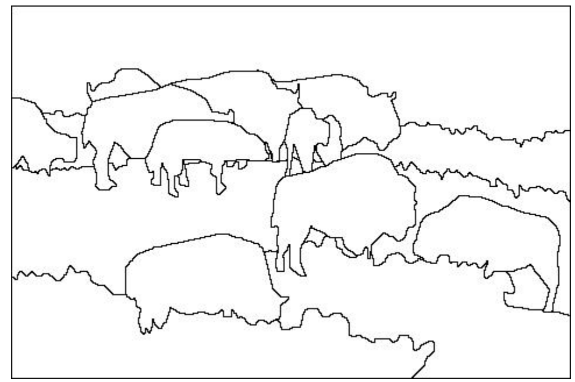
- Goal: identify groups of pixels that go together

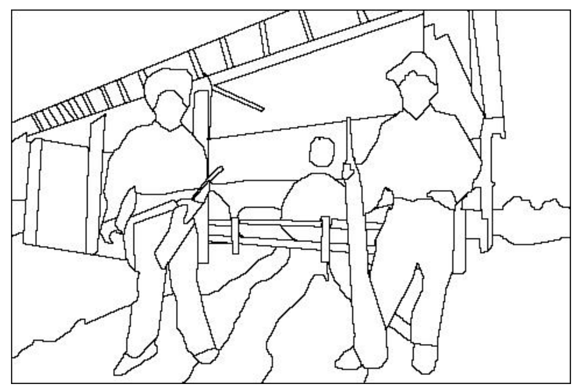


The Goals of Segmentation

- Separate image into coherent “objects”

Image | **Human segmentation**

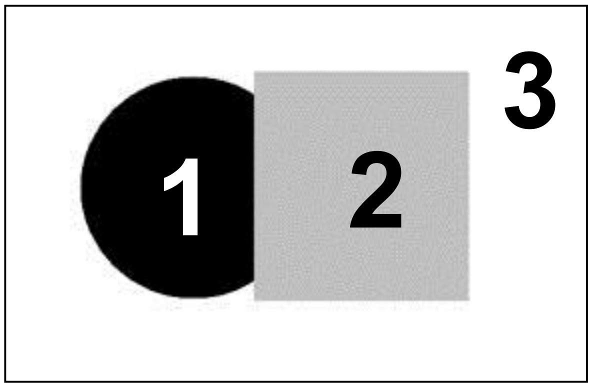



Slide credit: Svetlana Lazebnik

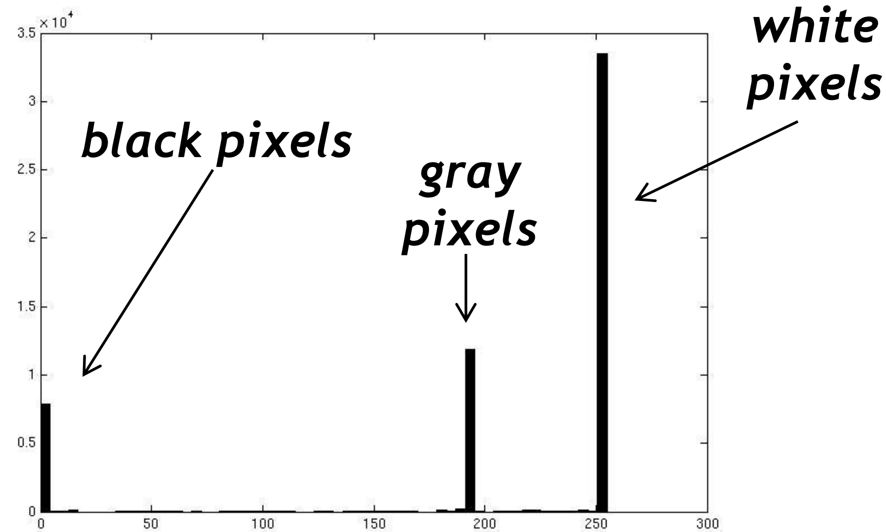
Topics of This Lecture

- Segmentation and grouping
  - Gestalt principles
  - Image Segmentation
- **Segmentation as clustering**
  - **k-Means**
  - **Feature spaces**
- Probabilistic clustering
  - Mixture of Gaussians, EM
- Model-free clustering
  - Mean-Shift clustering

Image Segmentation: Toy Example

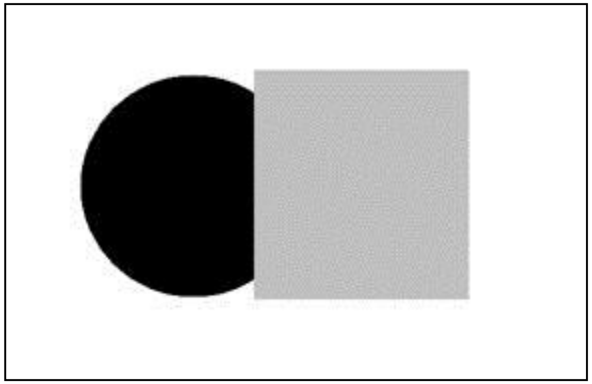


input image

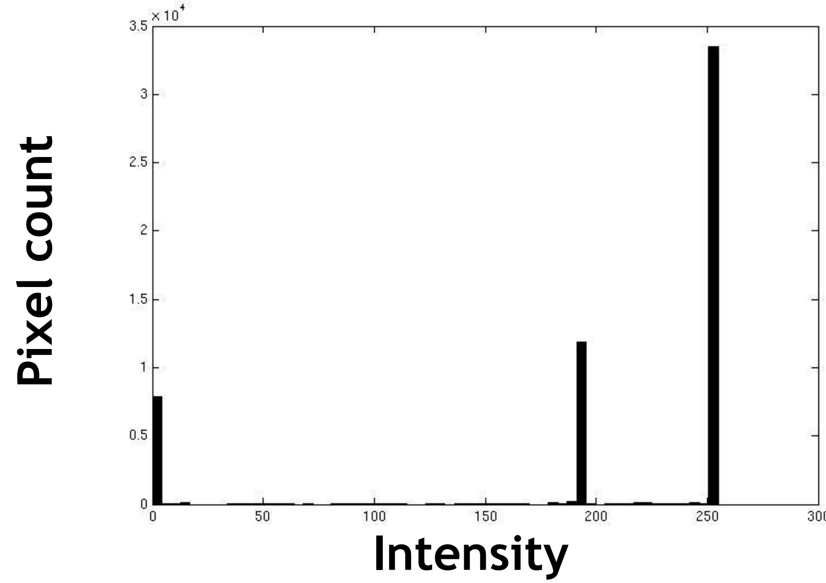


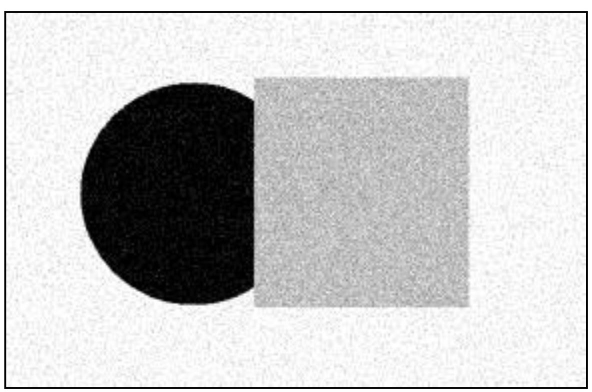
intensity

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

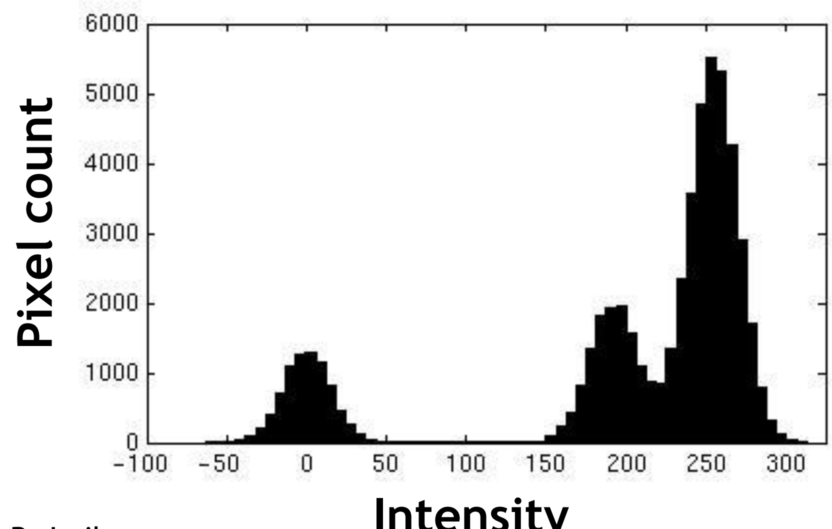


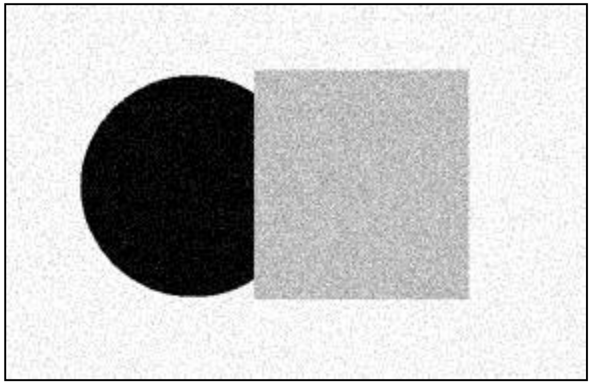
Input image



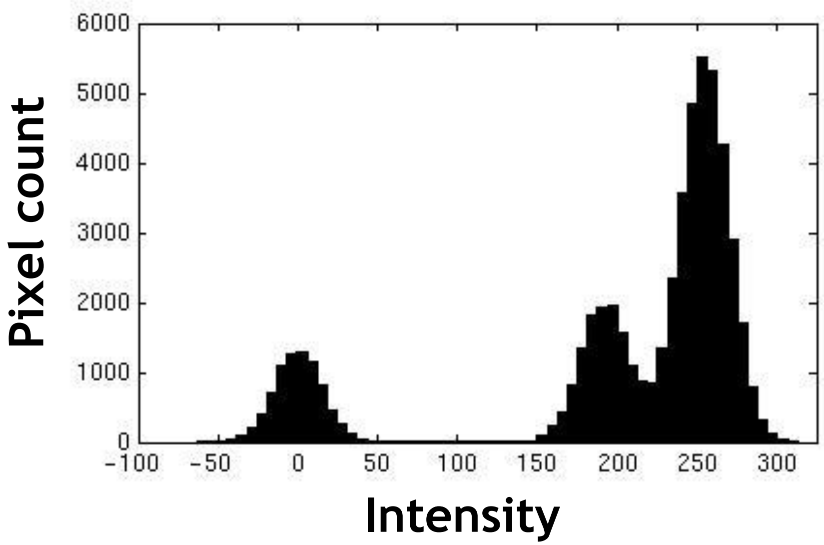


Input image

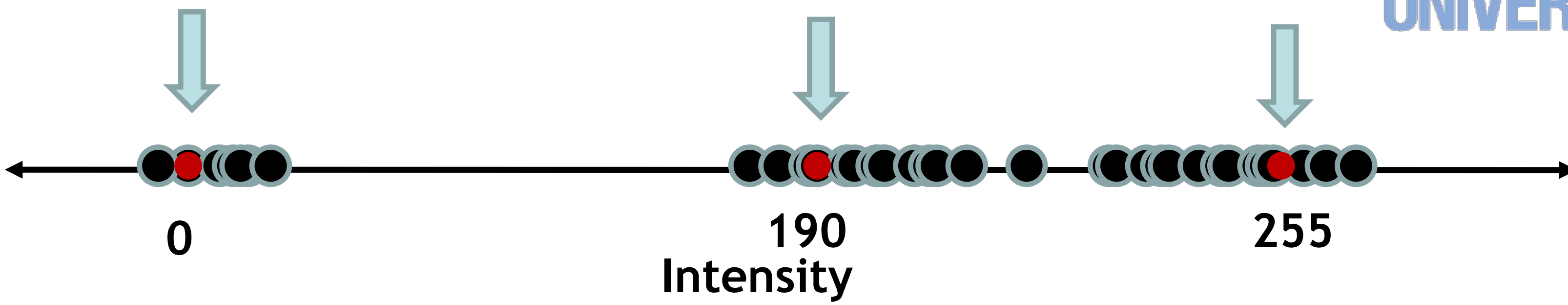


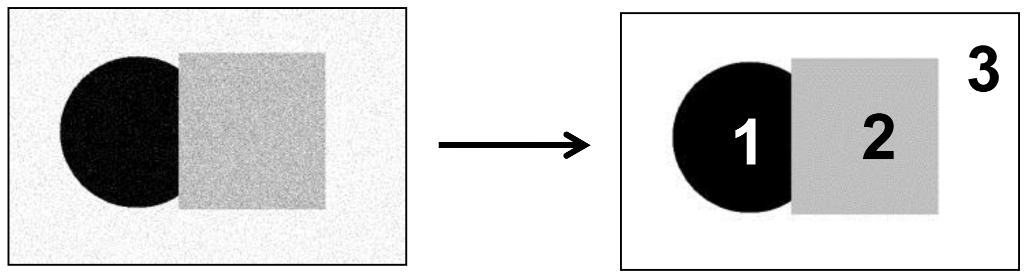


Input image



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



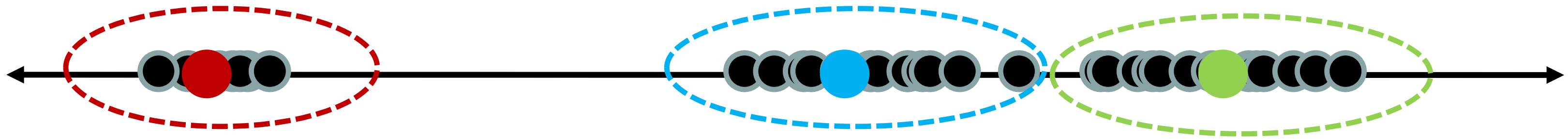


- Goal: choose three "centers" as the representative intensities, and label every pixel according to which of these centers it is nearest to.
- Best cluster centers are those that minimize SSD between all points and their nearest cluster center c_i:

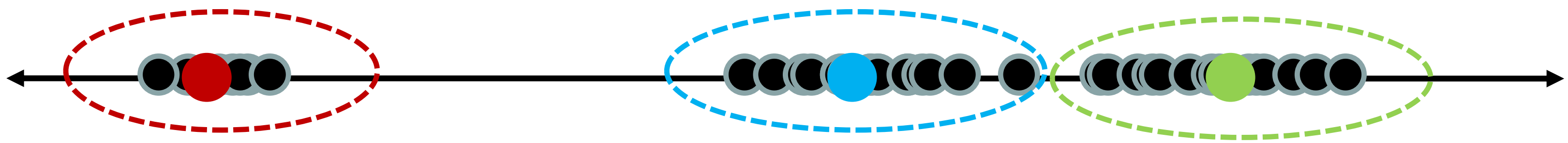
$$\sum_{\text{clusters } i} \sum_{\text{points p in cluster } i} ||p - c_i||^2$$

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.



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

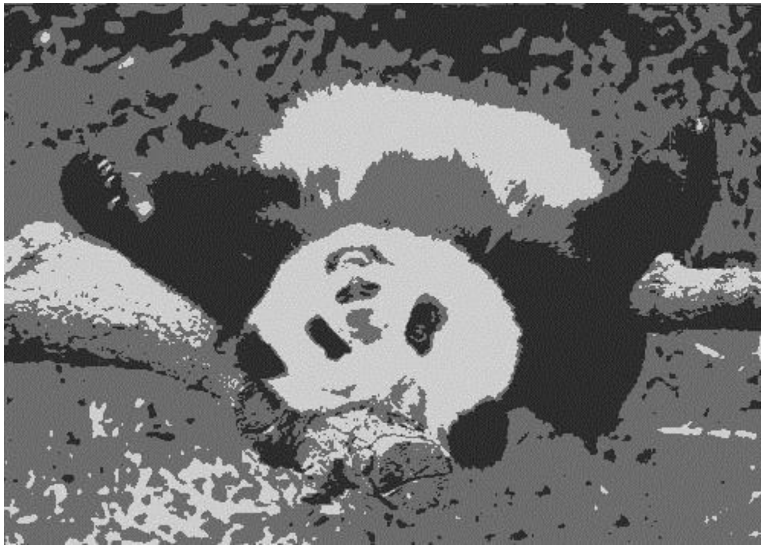
- Properties
  - Will always converge to *some* solution
  - Can be a "local minimum"
    - Does not always find the global minimum of objective function:

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

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++

- Can we prevent arbitrarily bad local minima?

1. Randomly choose first center.
2. Pick new center with prob. proportional to $||p - c_i||^2$
   - (Contribution of *p* to total error)
3. Repeat until *k* centers.

- Expected error = *O*(log *k*) * optimal

[Arthur & Vassilvitskii 2007](#)

Feature Space

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

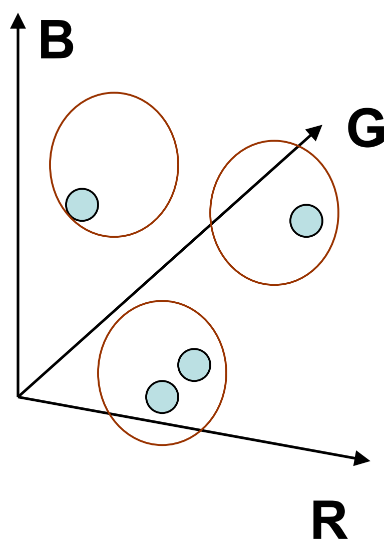
- Grouping pixels based on intensity similarity

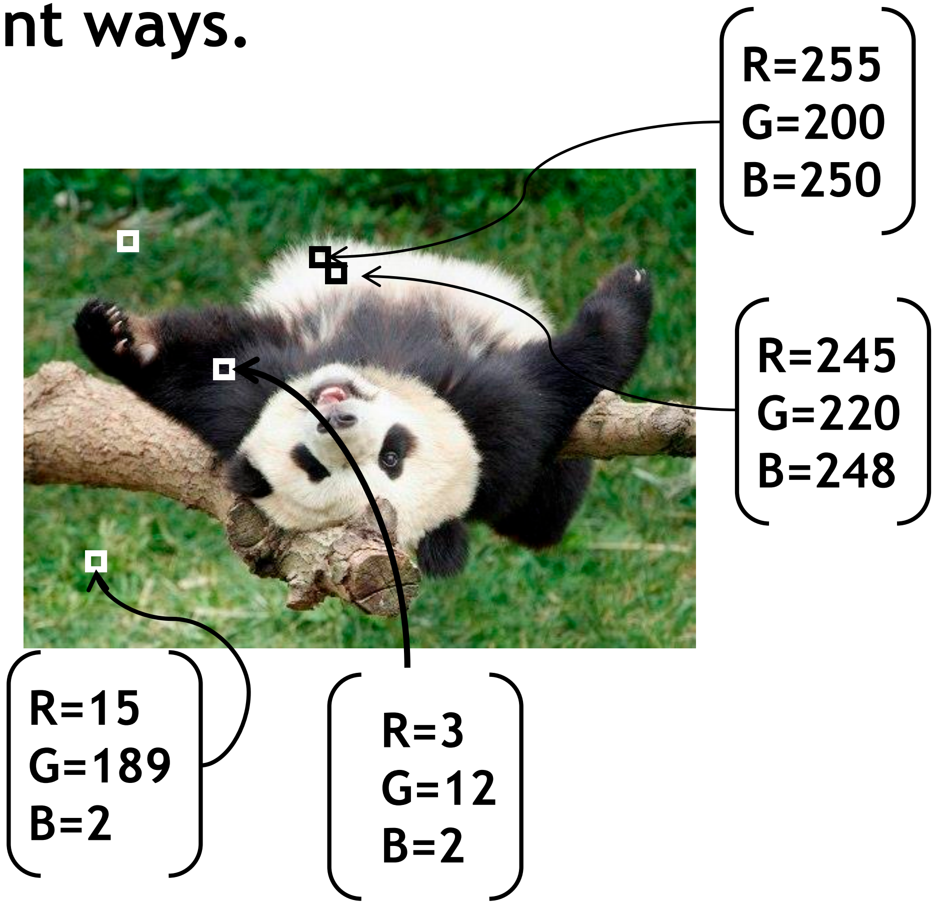
- Feature space: intensity value (1D)

Feature Space

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

- Grouping pixels based on color similarity



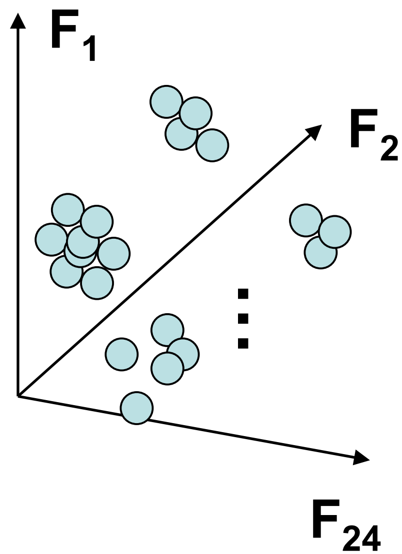


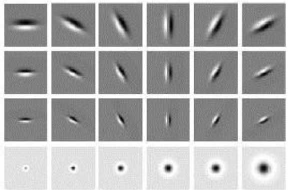
- Feature space: color value (3D)

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



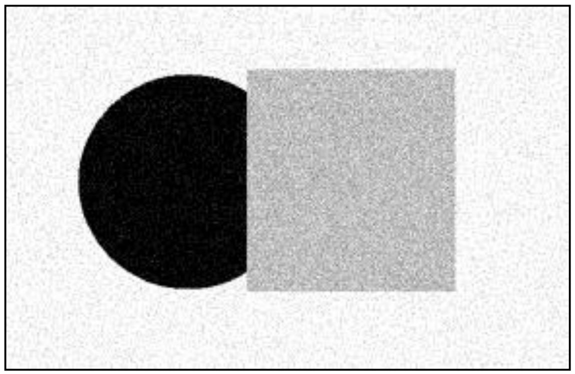


Filter bank of 24 filters

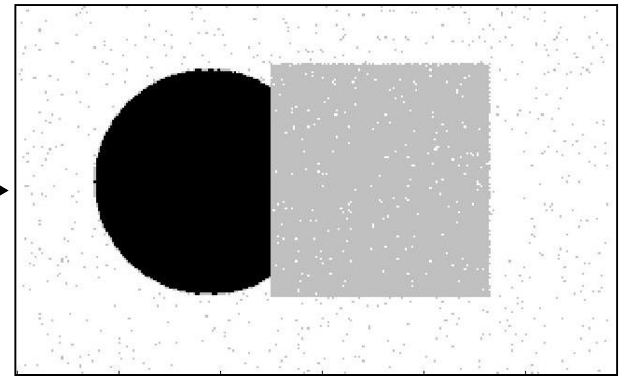
- Feature space: filter bank responses (e.g., 24D)

Smoothing Out Cluster Assignments

- Assigning a cluster label per pixel may yield outliers:

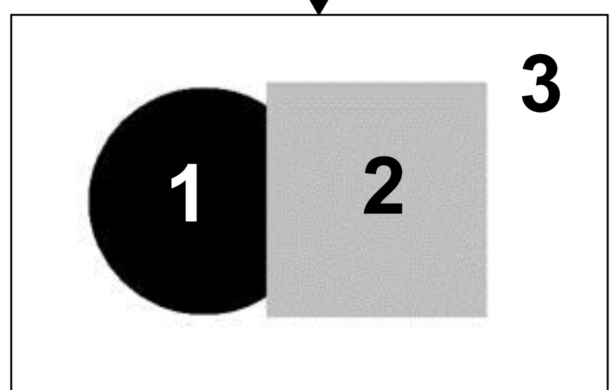


Original



Labeled by cluster center's intensity

?

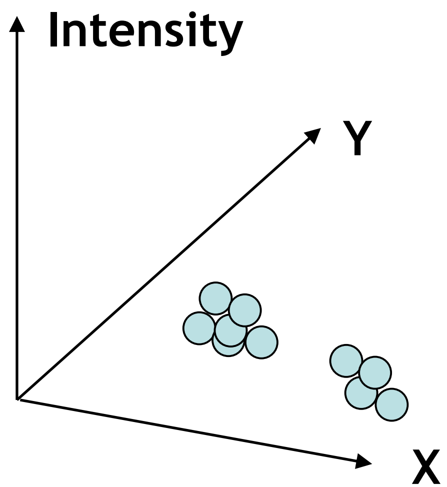


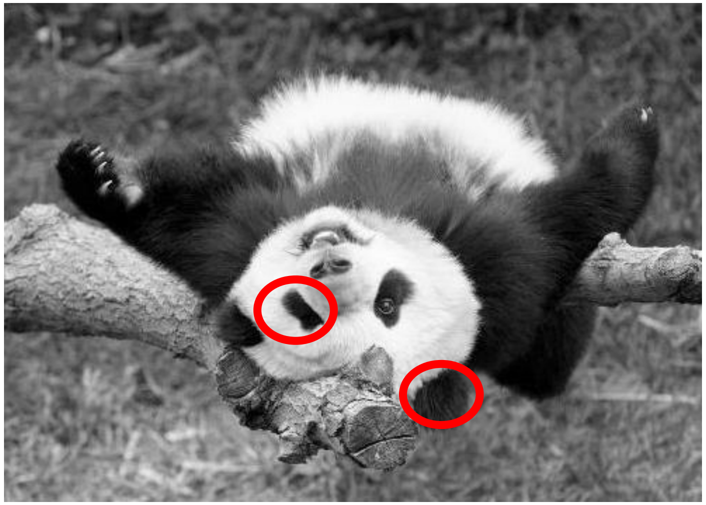
- How can we ensure they are spatially smooth?

Segmentation as Clustering

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

- Grouping pixels based on *intensity+position* similarity



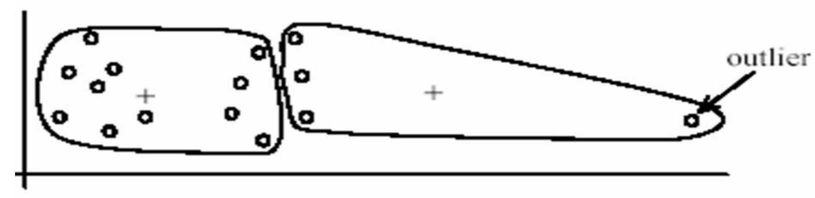


⇒ Simple way to encode both *similarity* and *proximity*.

Summary K-Means

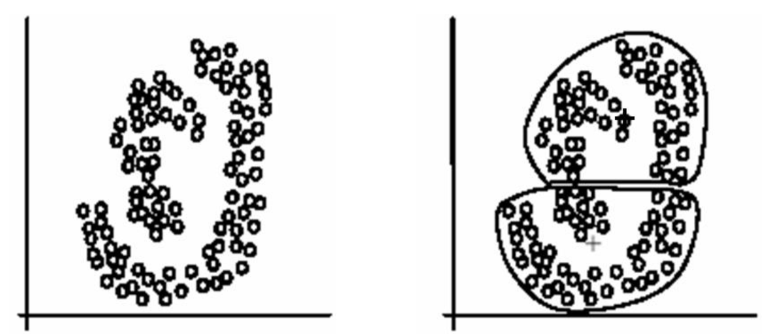
- **Pros**
  - Simple, fast to compute
  - Converges to local minimum of within-cluster squared error

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



(A): Undesirable clusters

(B): Ideal clusters



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

Topics of This Lecture

- Segmentation and grouping
  - Gestalt principles
  - Image Segmentation
- Segmentation as clustering
  - k-Means
  - Feature spaces
- **Probabilistic clustering**
  - **Mixture of Gaussians, EM**
- Model-free clustering
  - Mean-Shift clustering

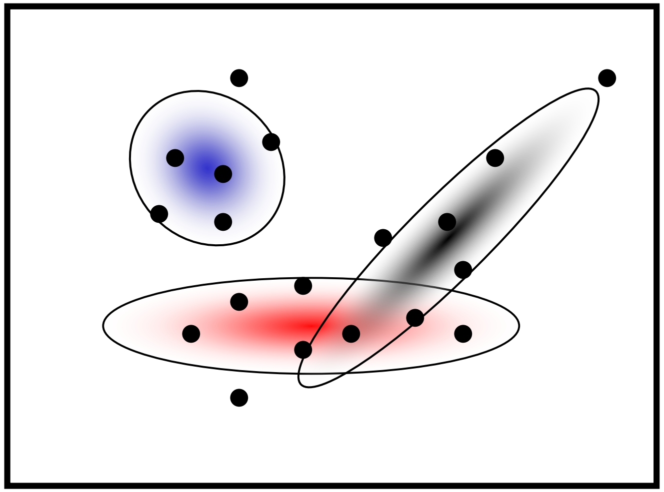
Probabilistic Clustering

- Basic questions
  - What's the probability that a point x is in cluster m?
  - What's the shape of each cluster?
- K-means doesn't answer these questions.

- Basic idea
  - Instead of treating the data as a bunch of points, assume that they are all generated by sampling a continuous function.
  - This function is called a generative model.
  - Defined by a vector of parameters θ

MoG tries to model the clustering problem in a generative manner. It tries to generate the continuous sampling function that might have generated the data points in the feature space

Mixture of Gaussians



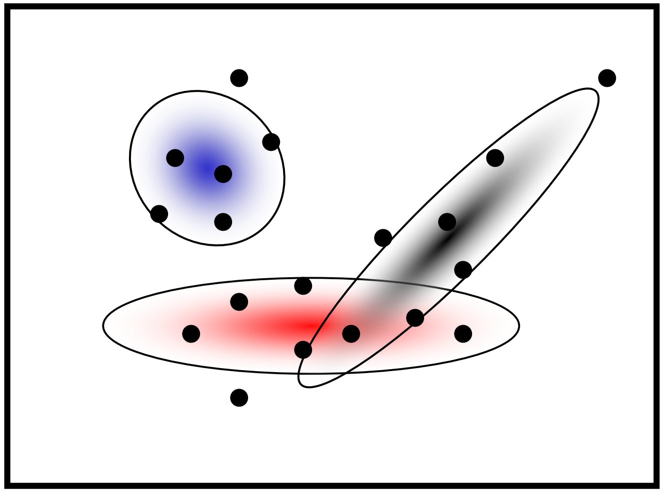
- One generative model is a mixture of Gaussians (MoG)
  - K Gaussian blobs with means μ_j, cov. matrices Σ_j, dim. D

$$p(\mathbf{x}|\theta_j) = \frac{1}{(2\pi)^{D/2}|\mathbf{\Sigma}_j|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_j)^{\mathrm{T}}\mathbf{\Sigma}_j^{-1}(\mathbf{x}-\boldsymbol{\mu}_j)\right\}$$

  - Blob j is selected with probability π_j
  - The likelihood of observing $\mathbf{x}$ is a weighted mixture of Gaussians

$$p(\mathbf{x}|\theta) = \sum_{j=1}^{K} \pi_j p(\mathbf{x}|\theta_j) \qquad \theta = (\pi_1, \boldsymbol{\mu}_1, \mathbf{\Sigma}_1, \ldots, \pi_M, \boldsymbol{\mu}_M, \mathbf{\Sigma}_M)$$

Expectation Maximization (EM)



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

$$p(data|\theta) = \prod_{n=1}^{N} p(\mathbf{x}_n|\theta)$$

- **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

EM Algorithm

see lecture Machine Learning!

- **Expectation-Maximization (EM) Algorithm**
  - **E-Step**: softly assign samples to mixture components

$$\gamma_j(\mathbf{x}_n) \leftarrow \frac{\pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \qquad \forall j = 1, \ldots, K, \quad n = 1, \ldots, N$$

  - **M-Step**: re-estimate the parameters (separately for each mixture component) based on the soft assignments

$$\hat{N}_j \leftarrow \sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)$$ = soft number of samples labeled j

$$\hat{\pi}_j^{\text{new}} \leftarrow \frac{\hat{N}_j}{N}$$

$$\hat{\boldsymbol{\mu}}_j^{\text{new}} \leftarrow \frac{1}{\hat{N}_j} \sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)\mathbf{x}_n$$

$$\hat{\boldsymbol{\Sigma}}_j^{\text{new}} \leftarrow \frac{1}{\hat{N}_j} \sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_j^{\text{new}})(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_j^{\text{new}})^{\text{T}}$$

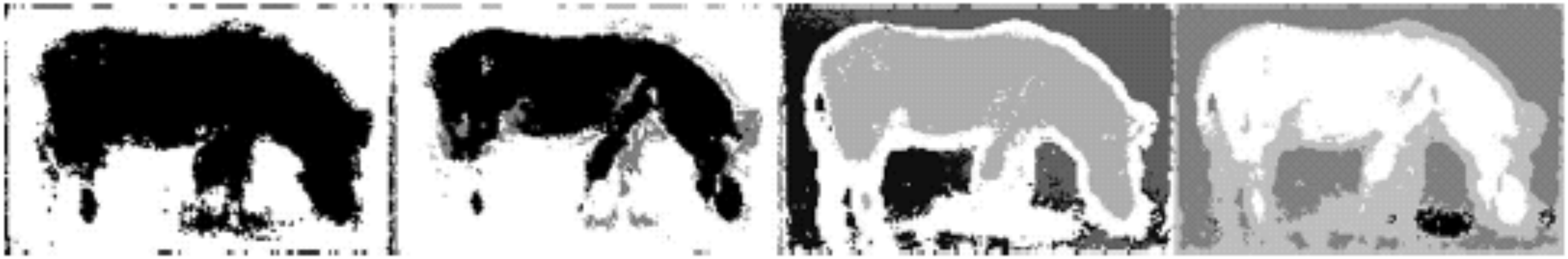
Applications of EM

- Turns out this is useful for all sorts of problems
  - Any clustering problem
  - Any model estimation problem
  - Missing data problems
  - Finding outliers
  - Segmentation problems
    - Segmentation based on color
    - Segmentation based on motion
    - Foreground/background separation
  - ...

Segmentation with EM

Original image

EM segmentation results



k=2 k=3 k=4 k=5

Image source: Serge Belongie

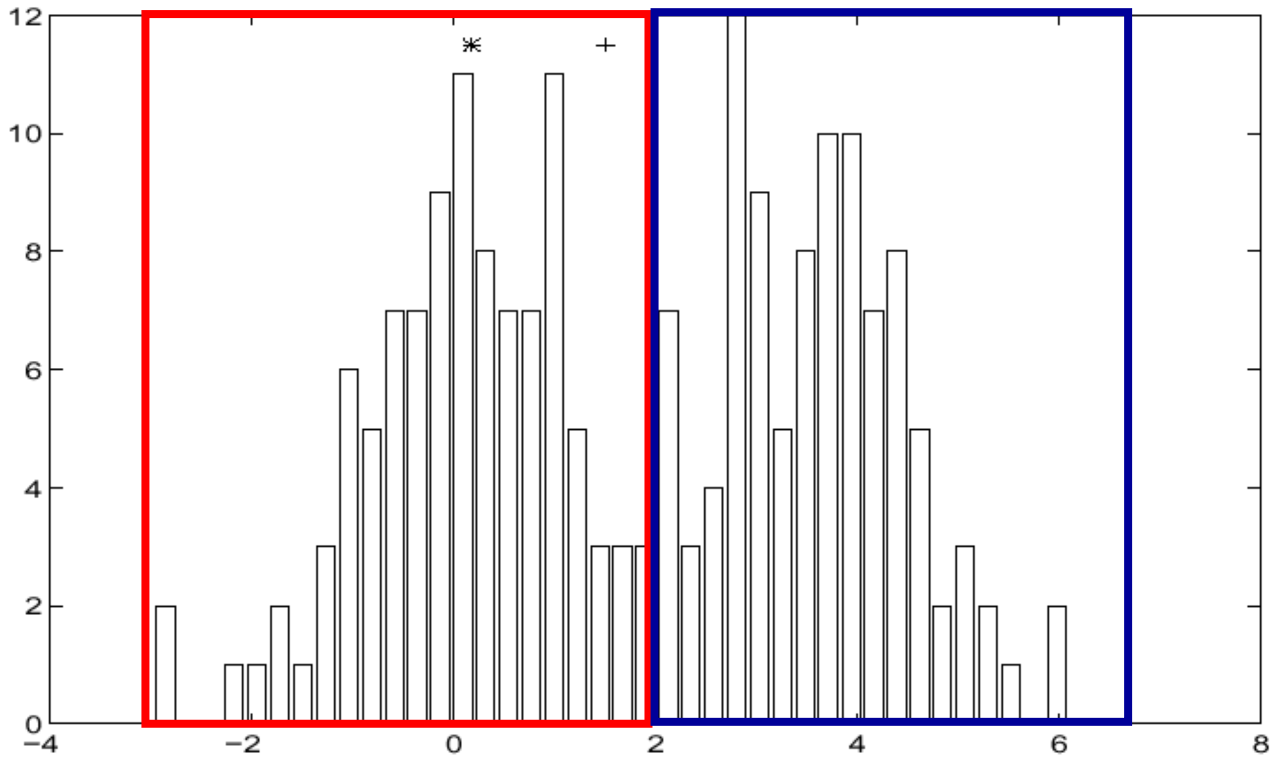
Summary: Mixtures of Gaussians, EM

- Pros
  - Probabilistic interpretation
  - Soft assignments between data points and clusters
  - Generative model, can predict novel data points
  - Relatively compact storage
- Cons
  - Local minima
    - k-means is NP-hard even with k=2
  - Initialization
    - Often a good idea to start with some k-means iterations.
  - Need to know number of components
    - Solutions: model selection (AIC, BIC), Dirichlet process mixture
  - Need to choose generative model
  - Numerical problems are often a nuisance

Topics of This Lecture

- Segmentation and grouping
  - Gestalt principles
  - Image segmentation
- Segmentation as clustering
  - k-Means
  - Feature spaces
- Probabilistic clustering
  - Mixture of Gaussians, EM
- **Model-free clustering**
  - **Mean-Shift clustering**

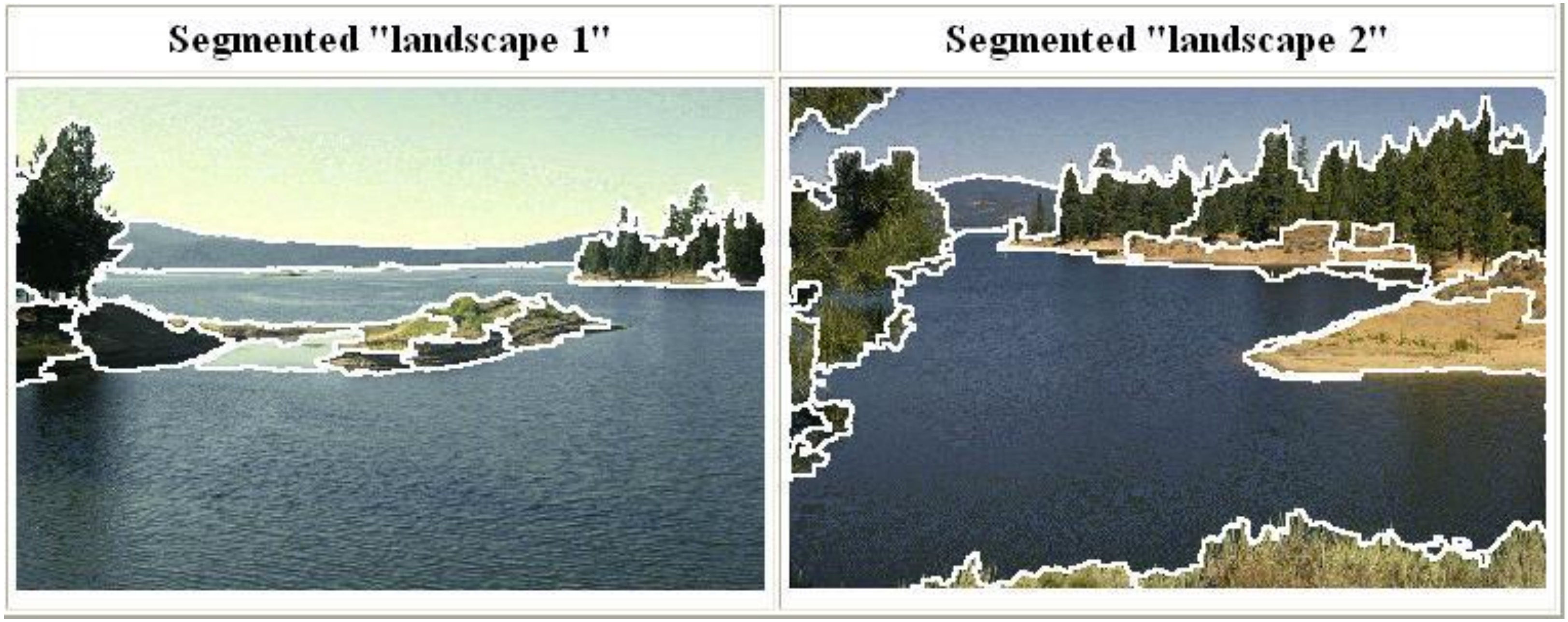
Finding Modes in a Histogram



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

Mean-Shift Segmentation

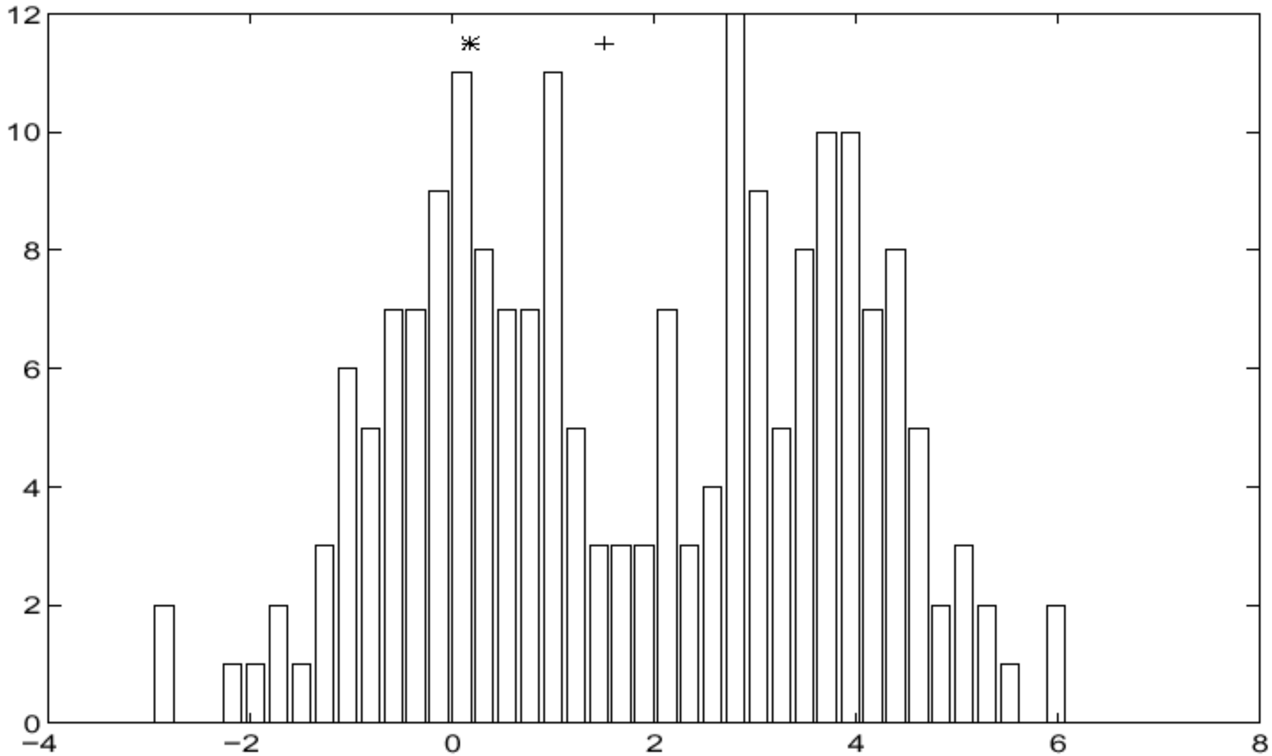
- An advanced and versatile technique for clustering-based segmentation



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

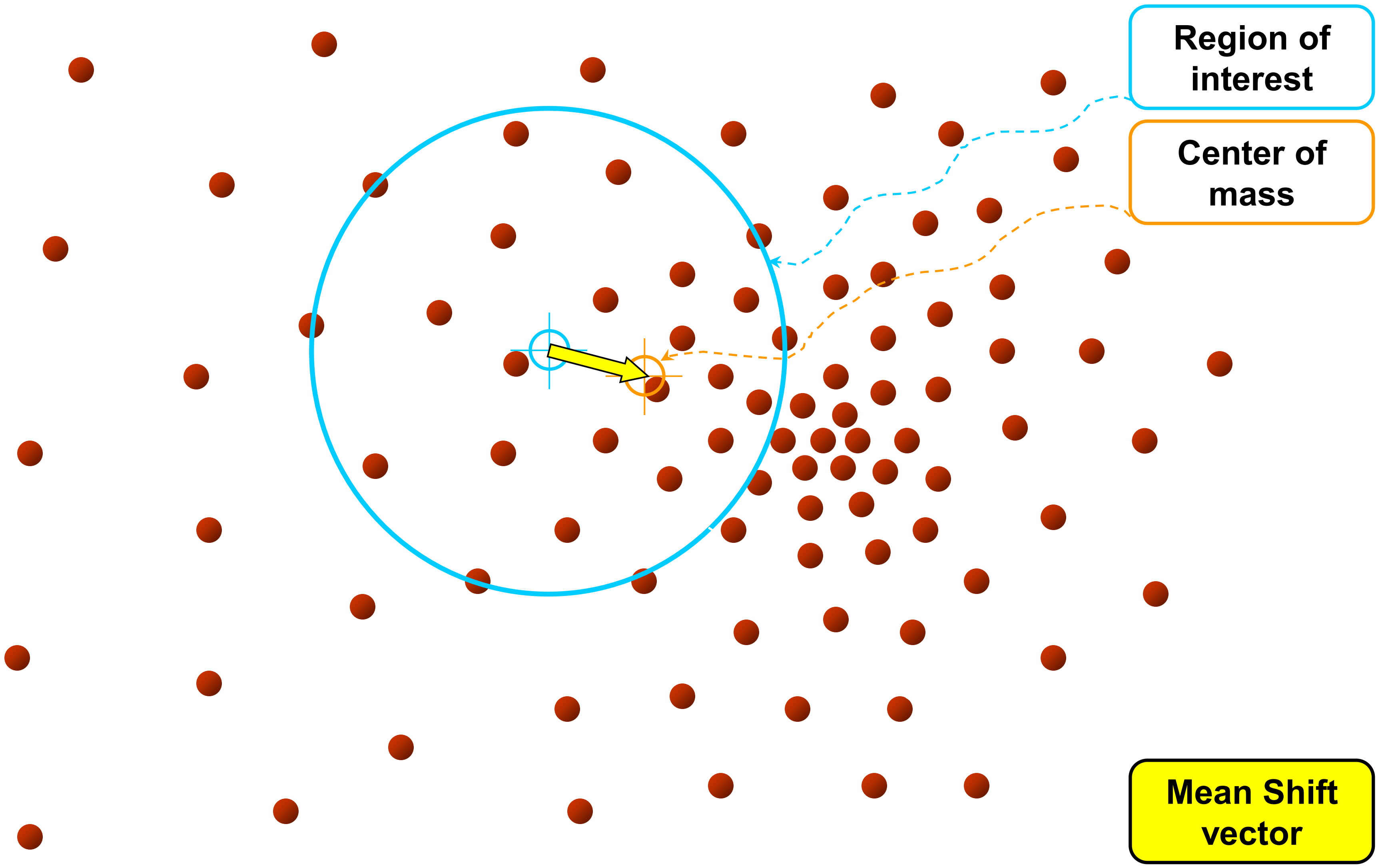
D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002.

Mean-Shift Algorithm



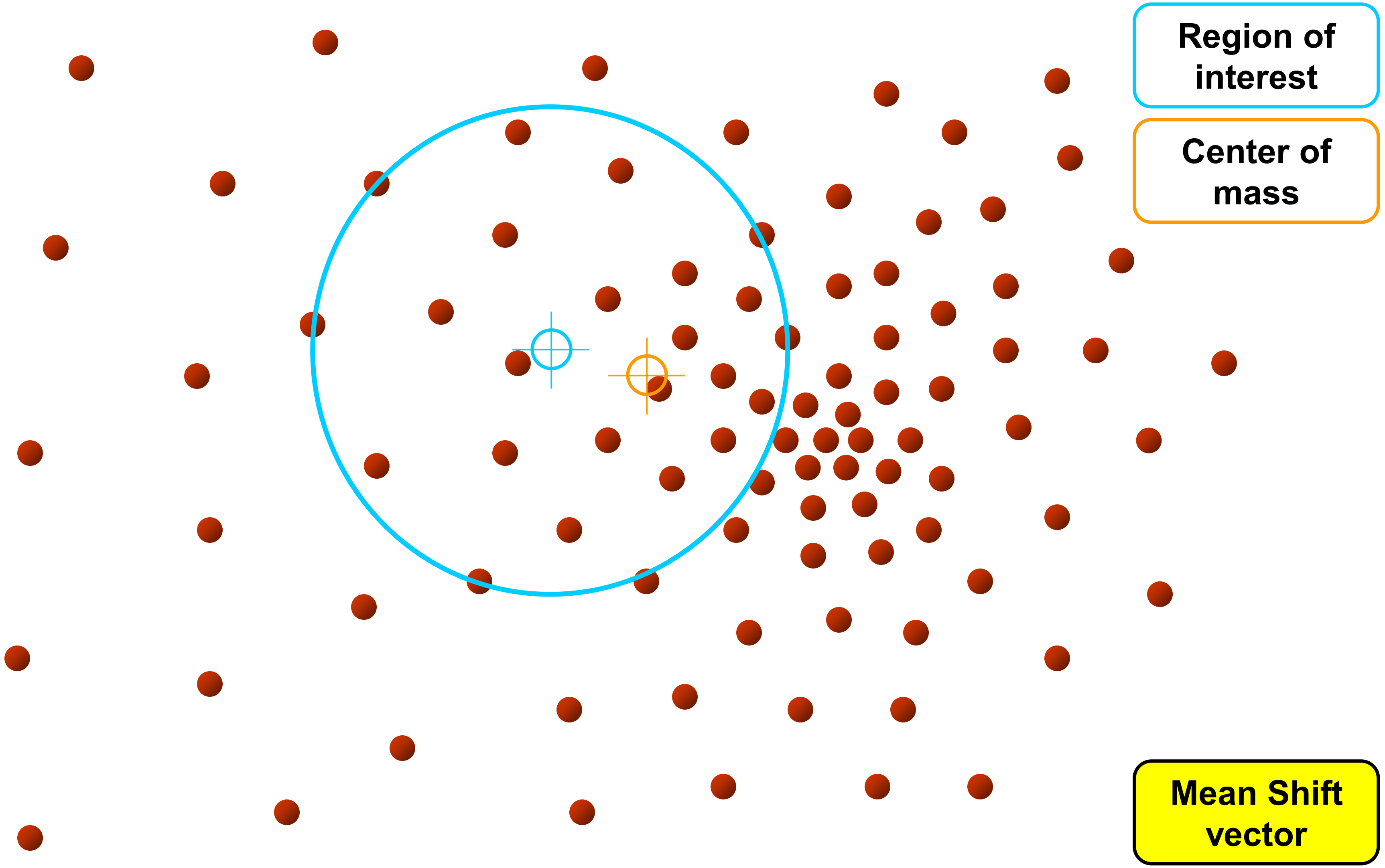
- **Iterative Mode Search**
  1. Initialize random seed, and window W
  2. Calculate center of gravity (the "mean") of W: $\sum_{x \in W} xH(x)$
  3. Shift the search window to the mean
  4. Repeat Step 2 until convergence

Mean-Shift



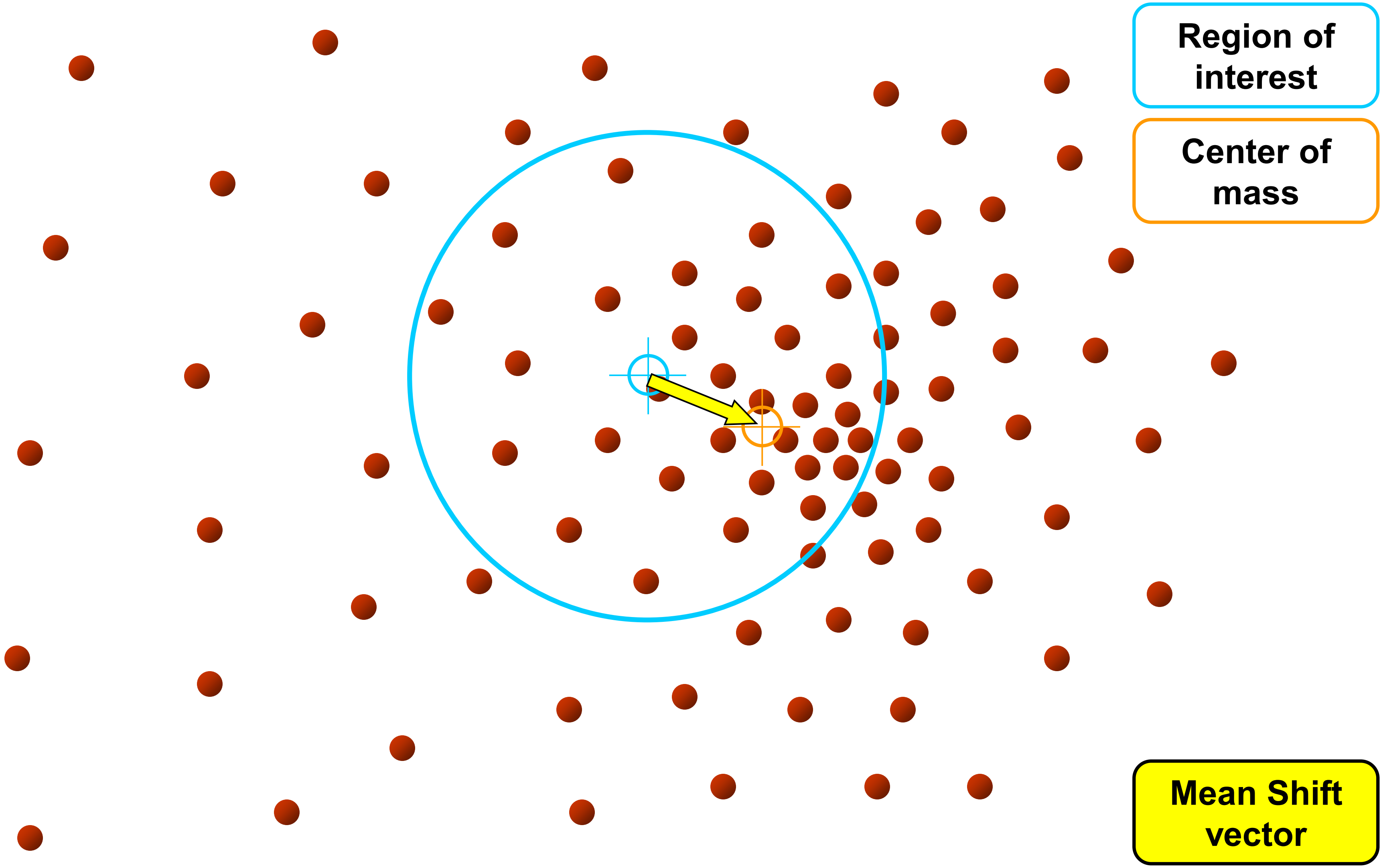
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



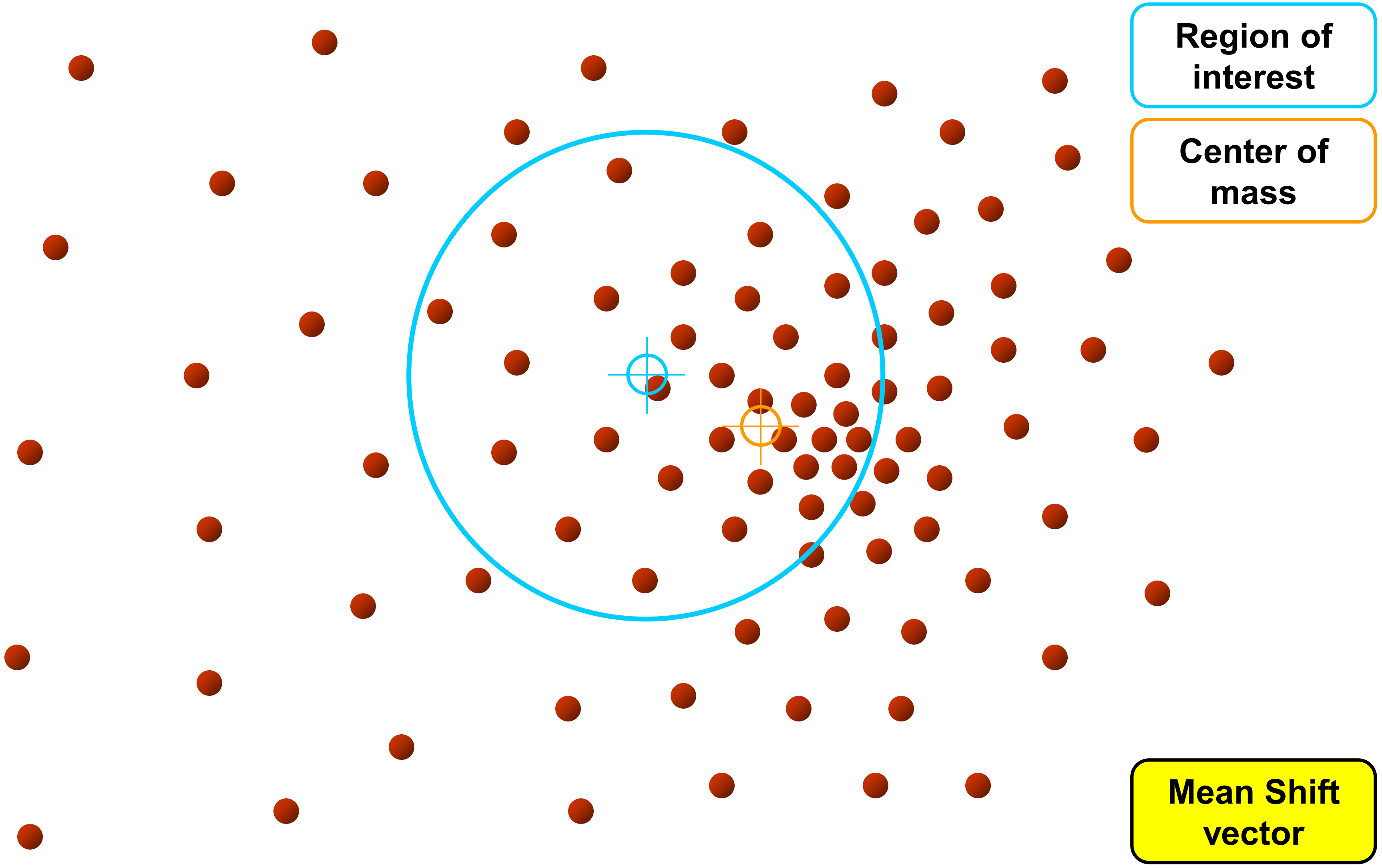
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



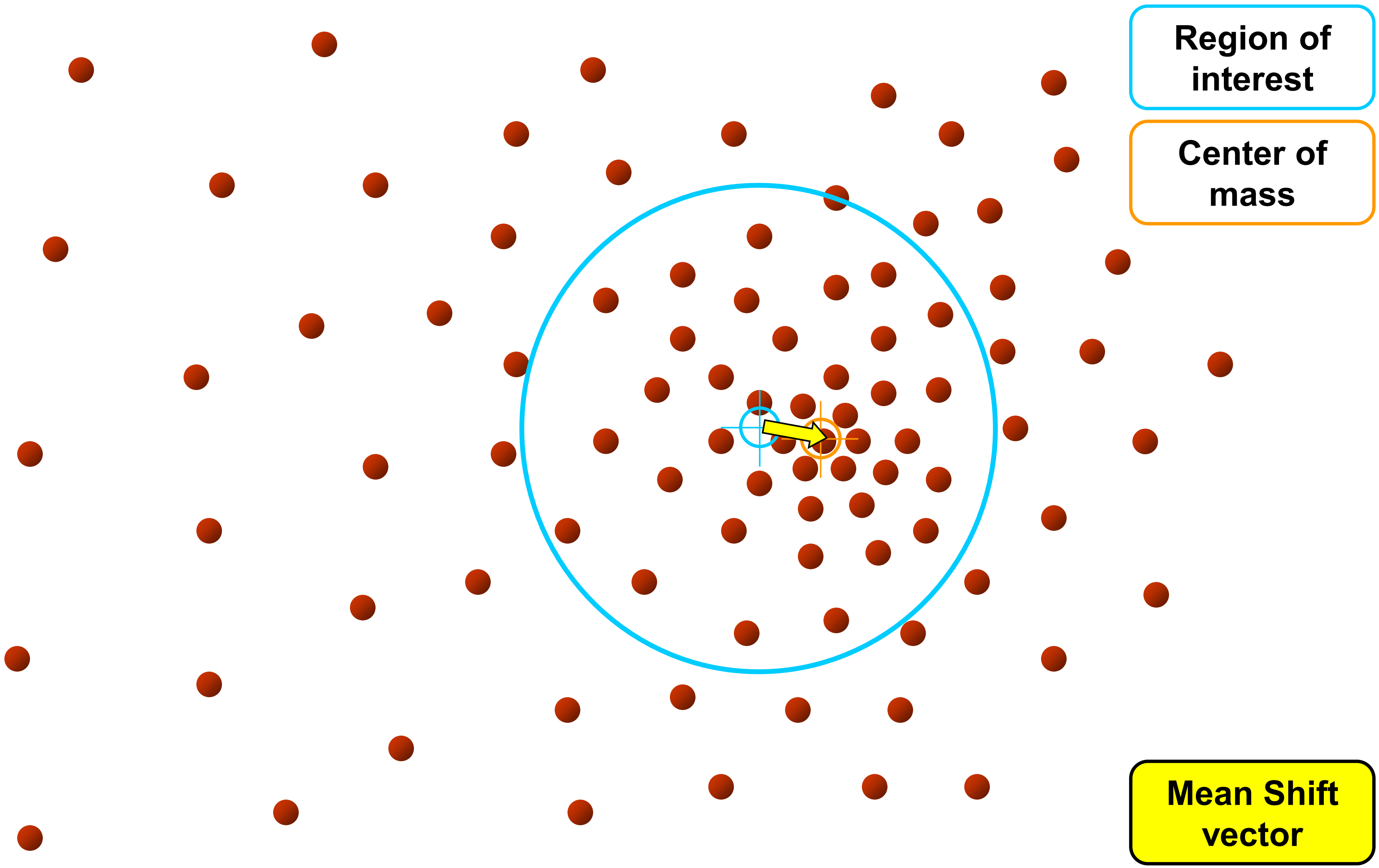
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift

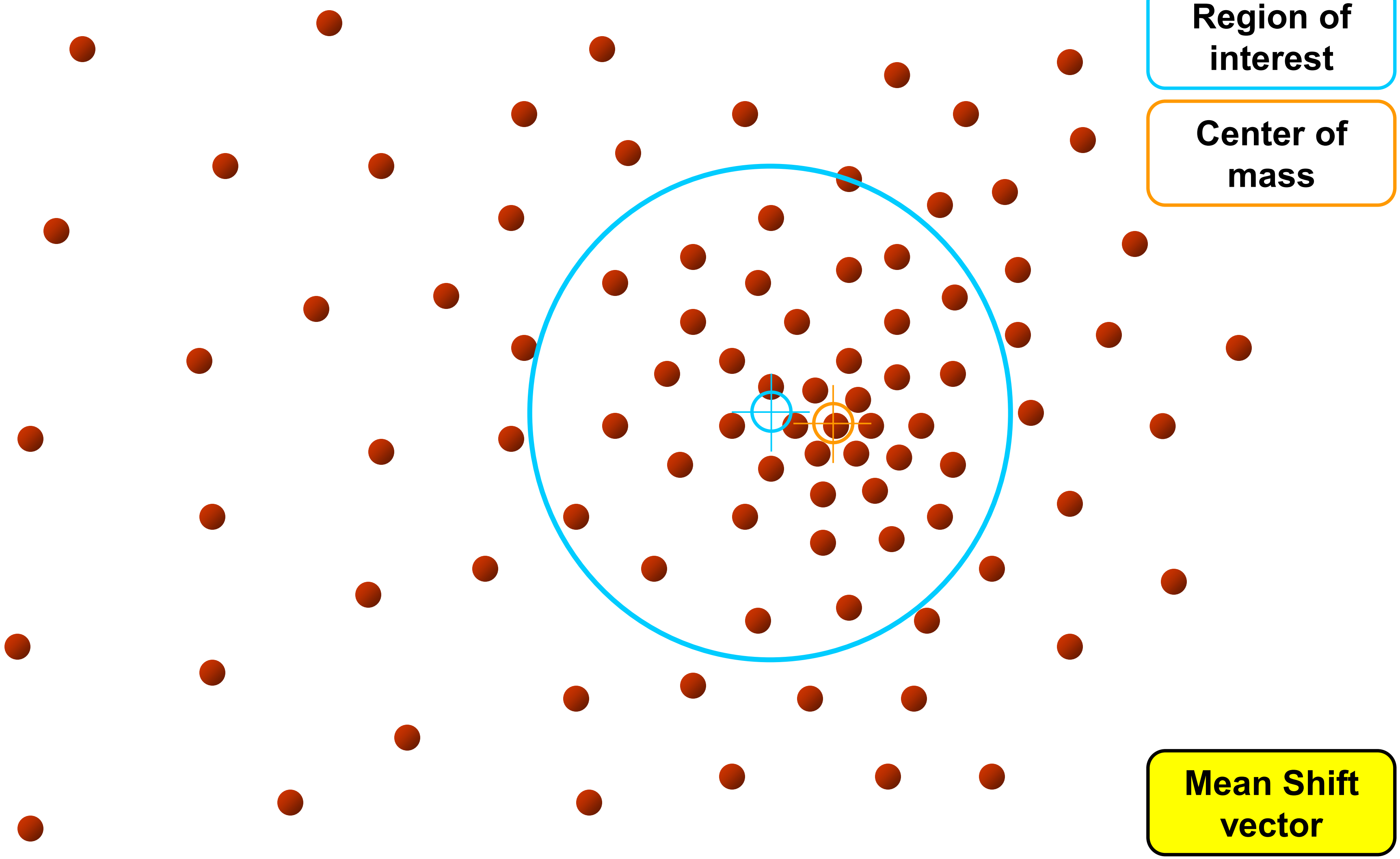


Slide by Y. Ukrainitz & B. Sarel

Mean-Shift

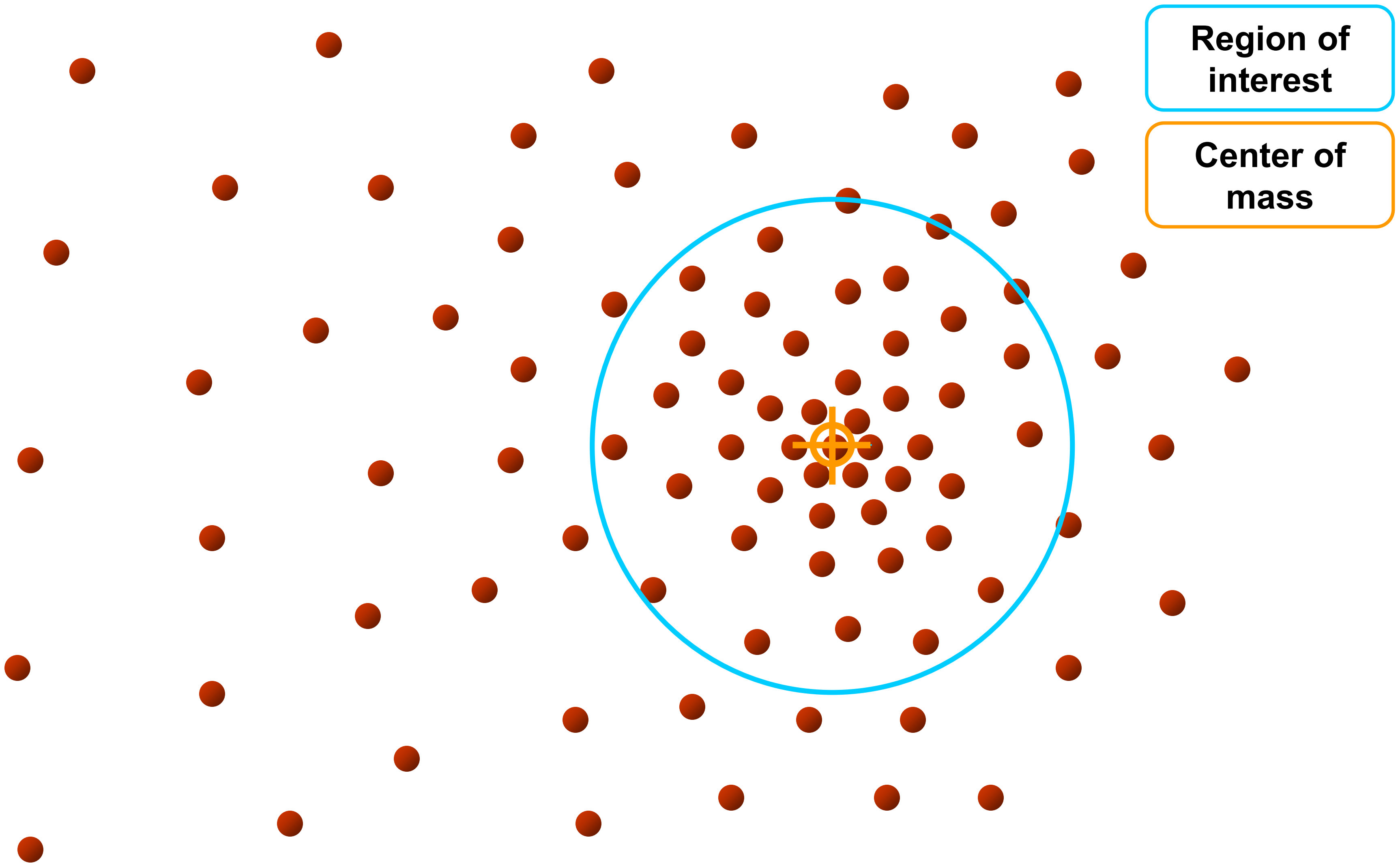


Mean-Shift



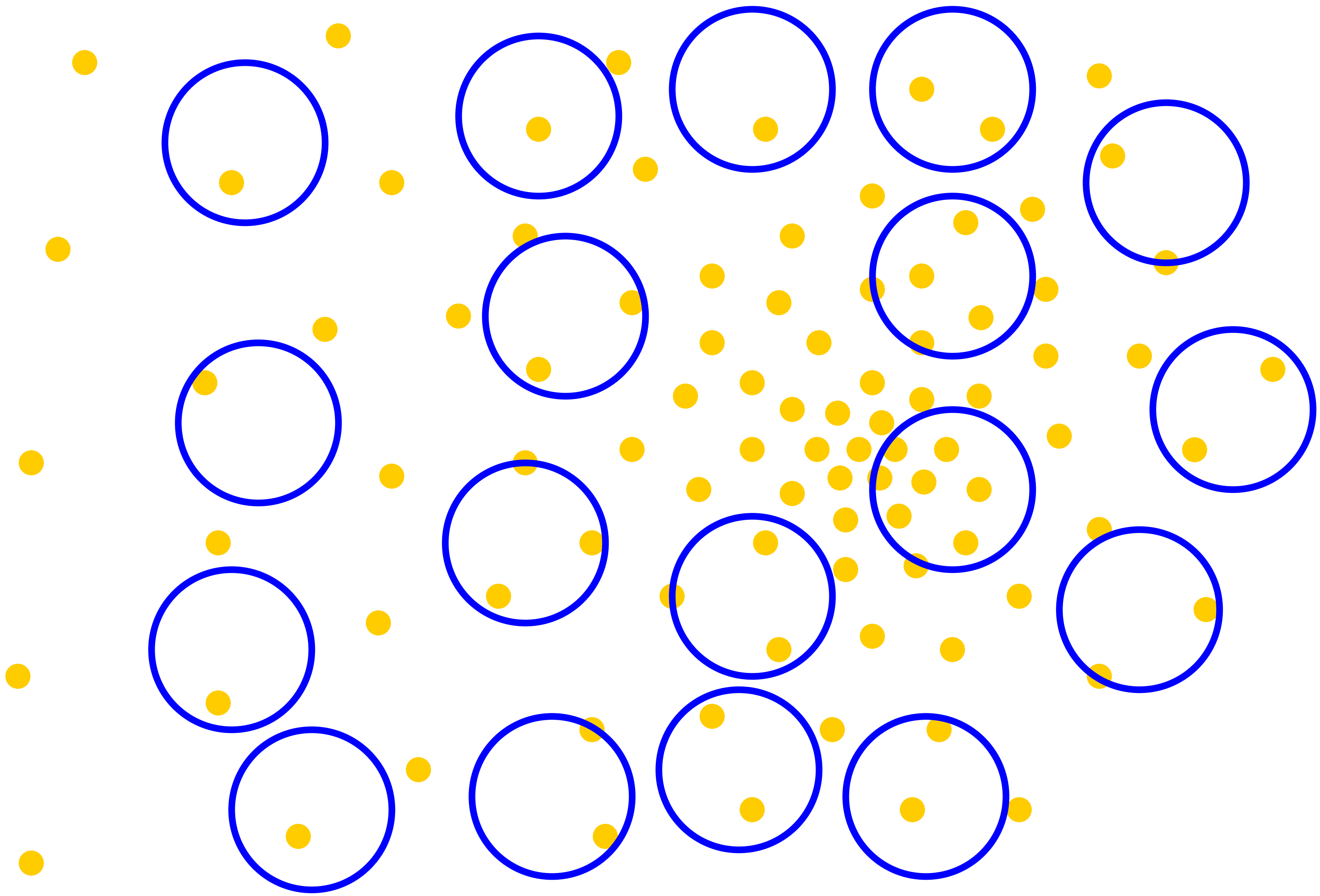
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift



Slide by Y. Ukrainitz & B. Sarel

Real Modality Analysis

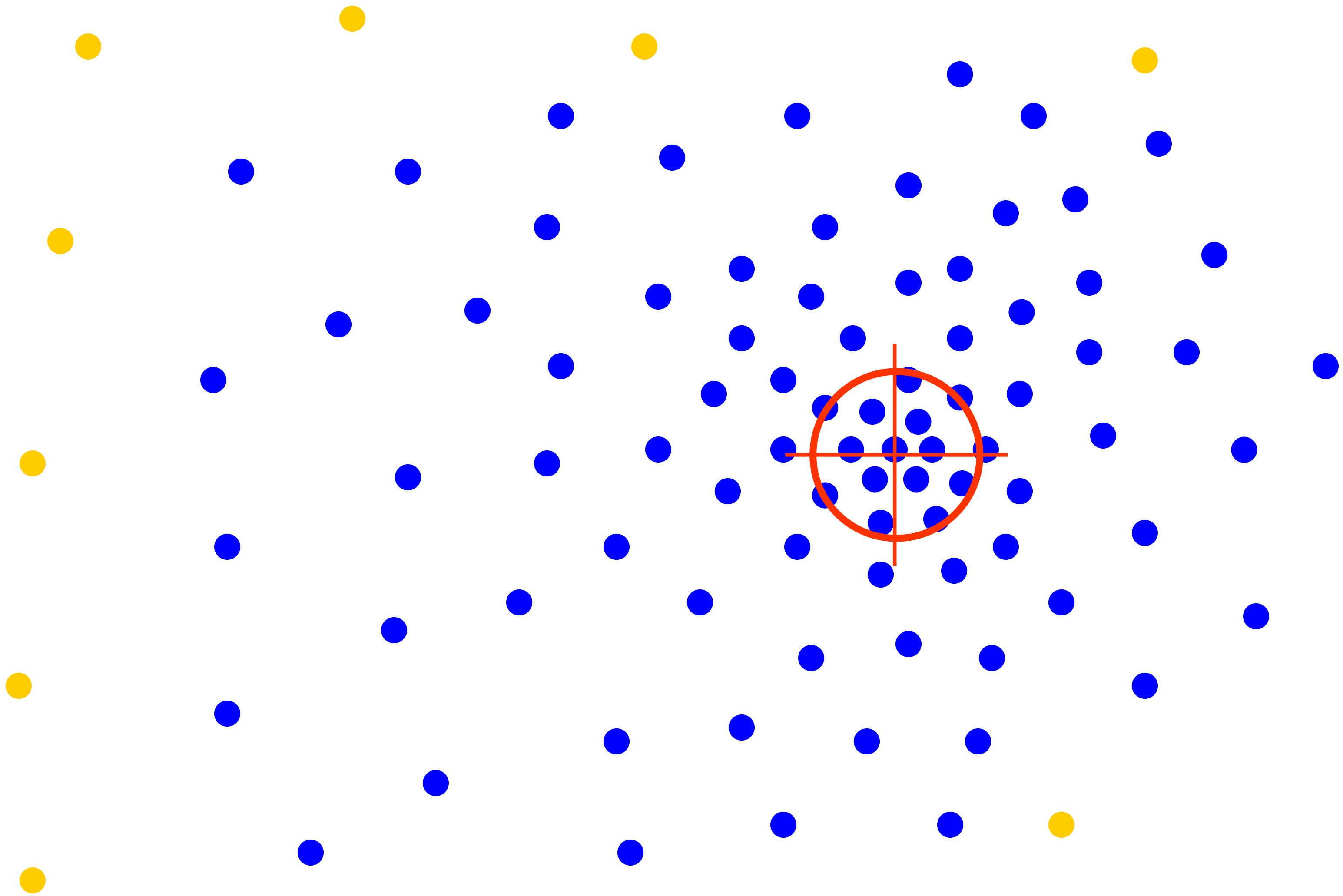


Tessellate the space with windows

Run the procedure in parallel

Slide by Y. Ukrainitz & B. Sarel

Real Modality Analysis

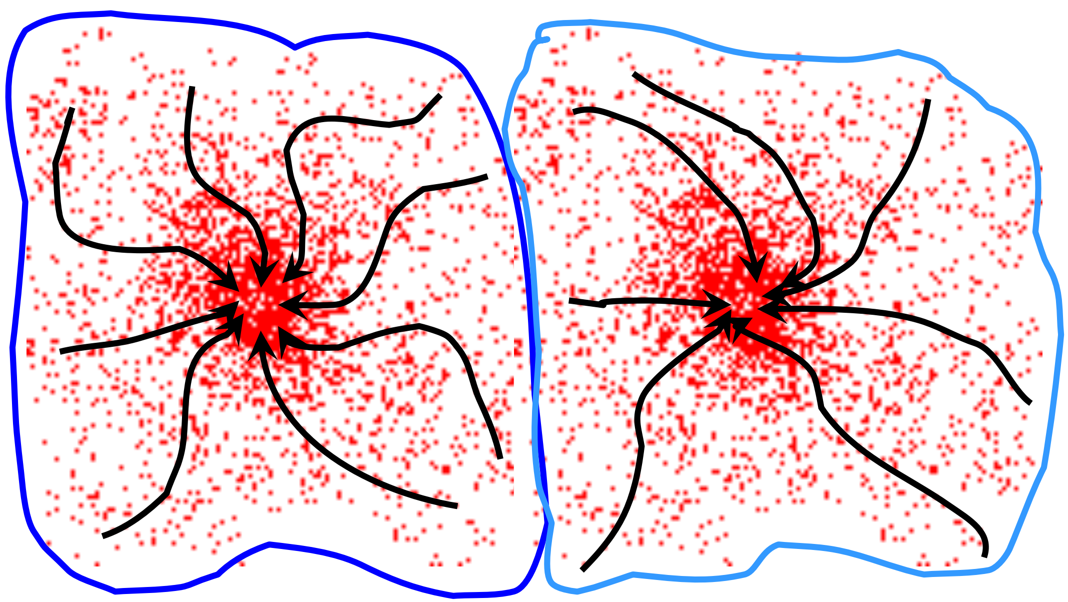


The blue data points were traversed by the windows towards the mode.

Slide by Y. Ukrainitz & B. Sarel

Mean-Shift Clustering

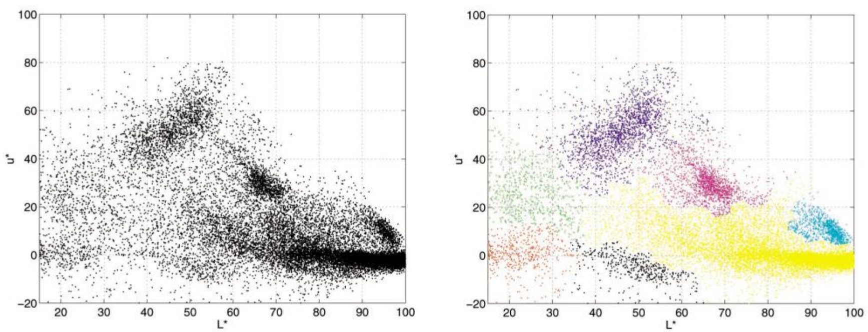
- Cluster: all data points in the attraction basin of a mode
- Attraction basin: the region for which all trajectories lead to the same mode



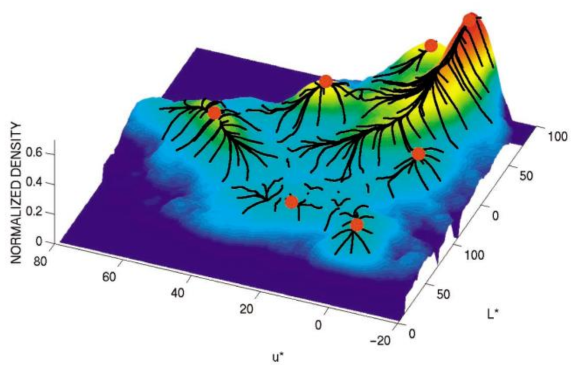
Slide by Y. Ukrainitz & B. Sarel

Mean-Shift Clustering/Segmentation

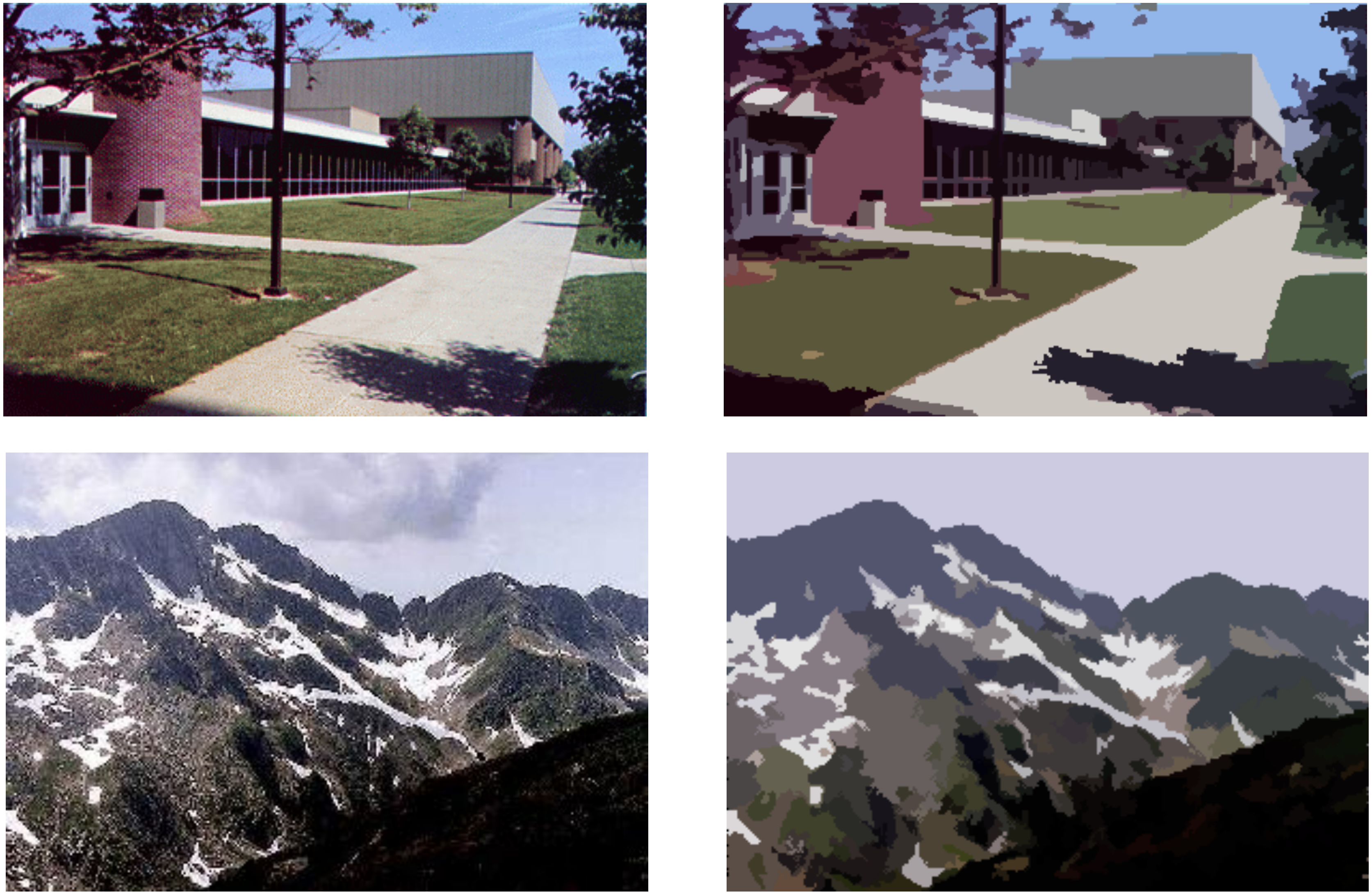
- Find features (color, gradients, texture, etc)
- Initialize windows at individual pixel locations
- Perform mean shift for each window until convergence
- Merge windows that end up near the same “peak” or mode



(a) (b)

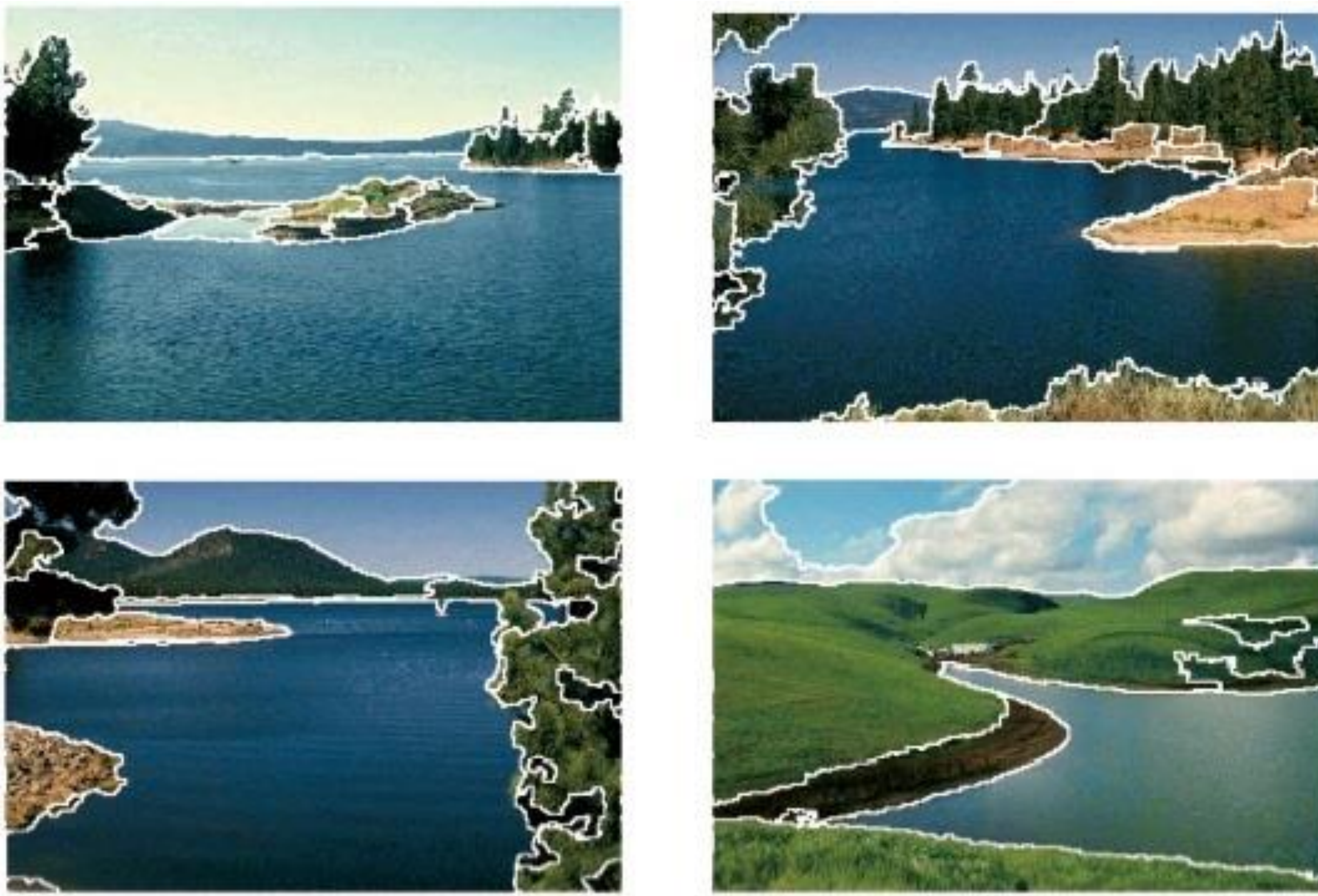


Mean-Shift Segmentation Results



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

More Results

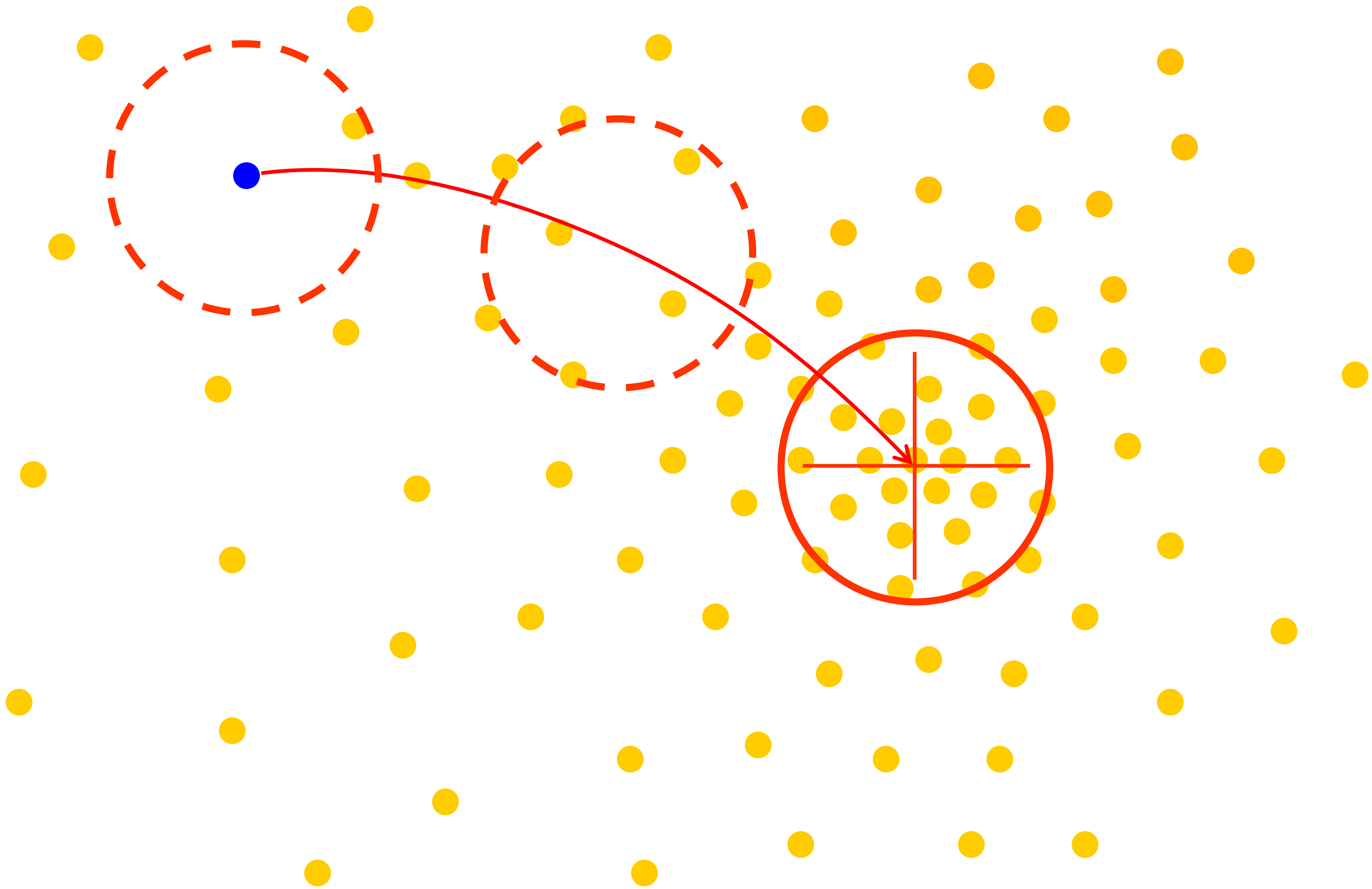


Slide credit: Svetlana Lazebnik

More Results

Slide credit: Svetlana Lazebnik

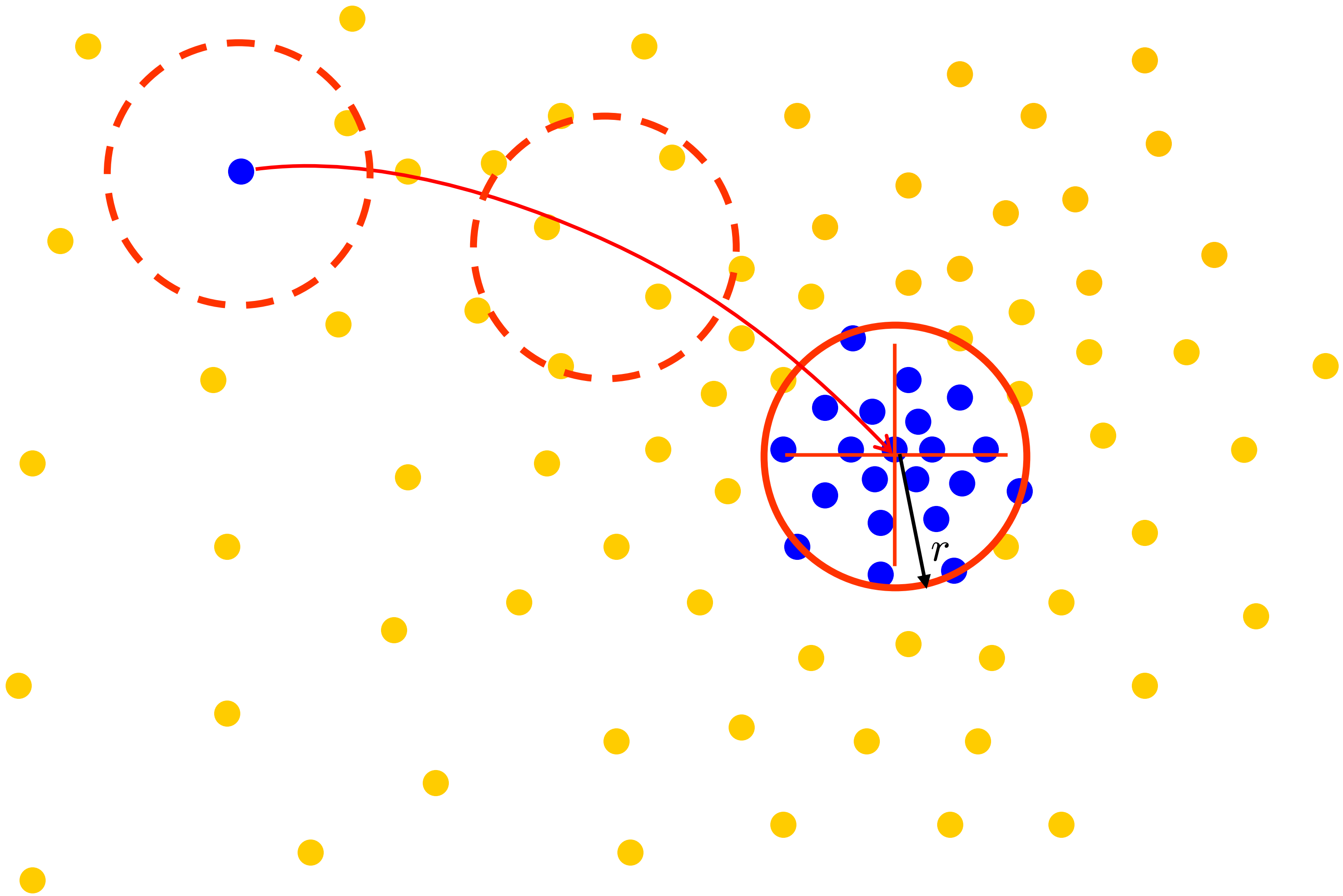
Problem: Computational Complexity



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

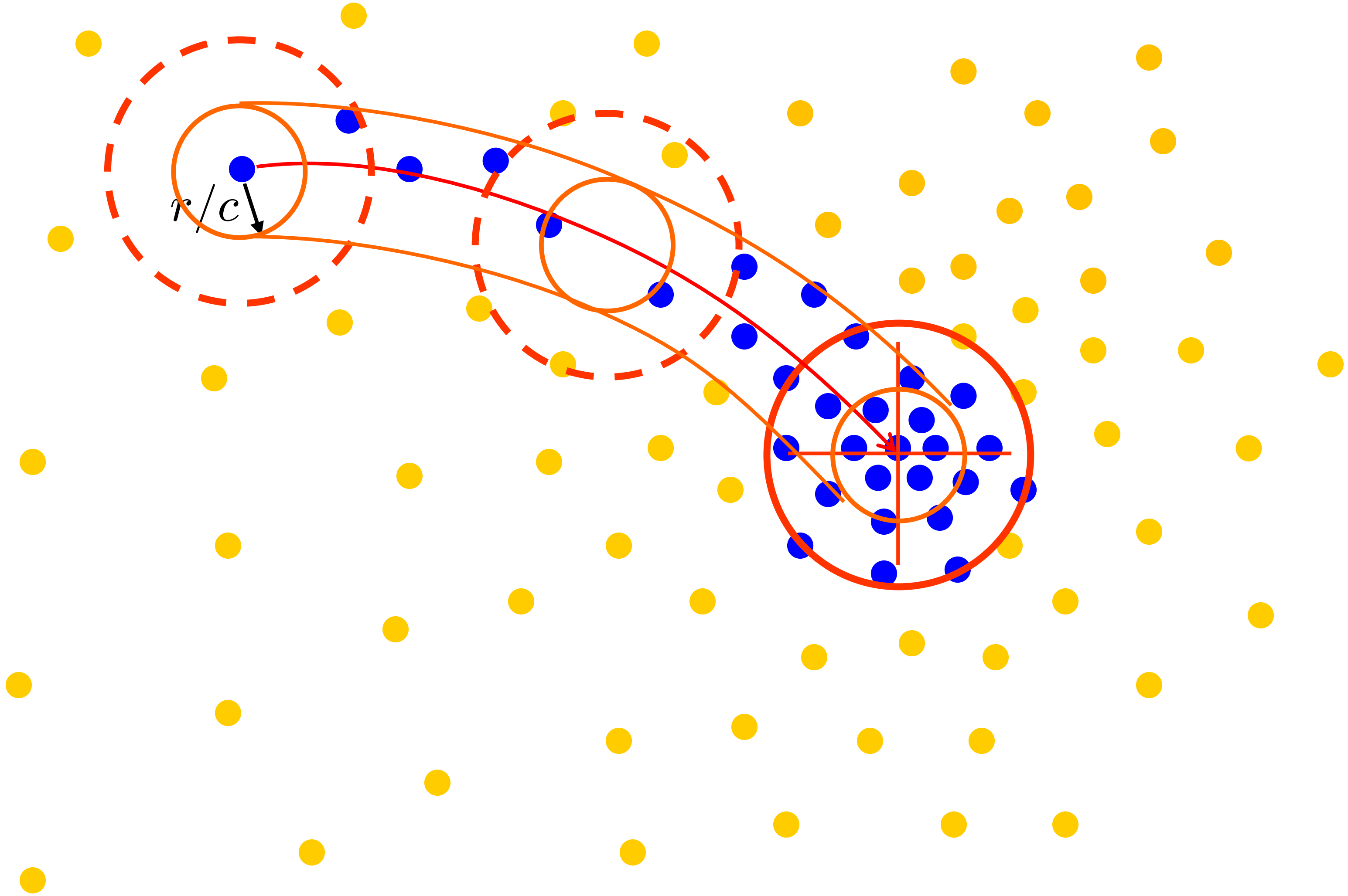
RWTH AACHEN UNIVERSITY

Speedups: Basin of Attraction



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

Speedups



2. Assign all points within radius r/c of the search path to the mode.

Summary Mean-Shift

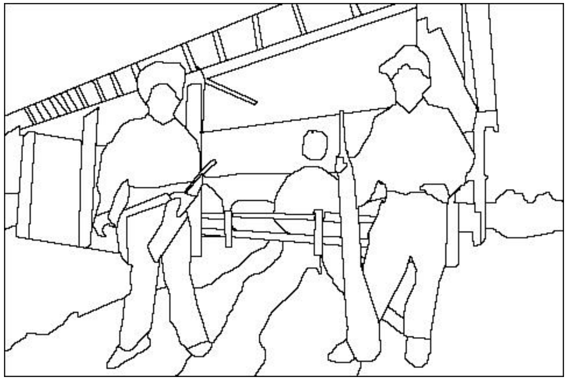
- **Pros**
  - General, application-independent tool
  - Model-free, does not assume any prior shape (spherical, elliptical, etc.) on data clusters
  - Just a single parameter (window size h)
    - h has a physical meaning (unlike k-means)
  - Finds variable number of modes
  - Robust to outliers

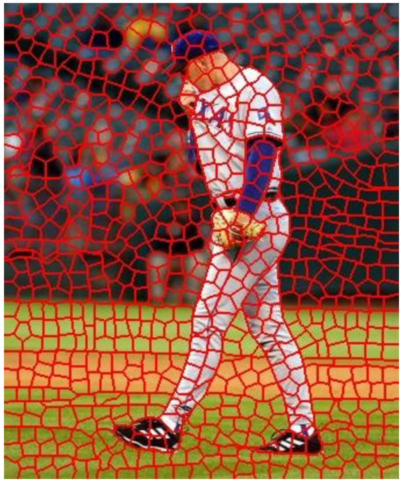
- **Cons**
  - Output depends on window size
  - Window size (bandwidth) selection is not trivial
  - Computationally (relatively) expensive (~2s/image)
  - Does not scale well with dimension of feature space

Slide credit: Svetlana Lazebnik

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
- Difficult to evaluate; when is a segmentation successful?





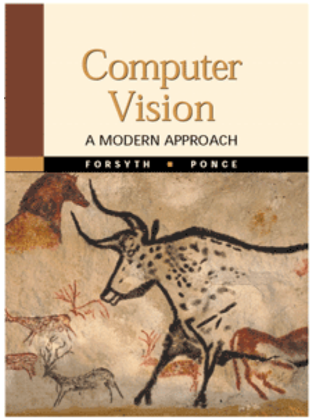
Slide credit: Kristen Grauman

Generic Clustering

- We have focused on ways to group pixels into image segments based on their appearance
  - Find groups; "quantize" feature space
- In general, we can use clustering techniques to find groups of similar "tokens", provided we know how to compare the tokens.
  - *E.g.*, segment an image into the types of motions present
  - *E.g.*, segment a video into the types of scenes (shots) present

References and Further Reading

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



- More on the EM algorithm can be found in Chapter 16.1.2.