Artificial Intelligence II

Part 2: Lecture 3

Yalda Mohsenzadeh

Slides are adapted from Olga Vesker (UW), Steve Seitz (UW), David Jacobs (UMD), D. Lowe (UBC), Hong Man

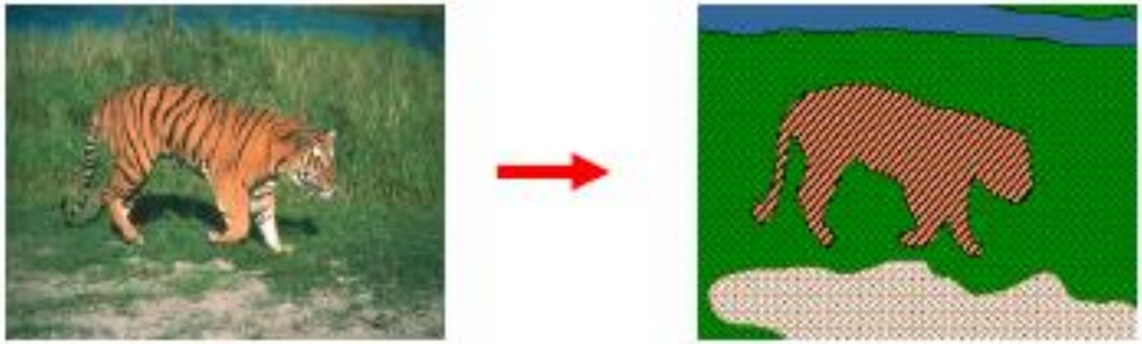
Computer Vision

Image Segmentation

Outline

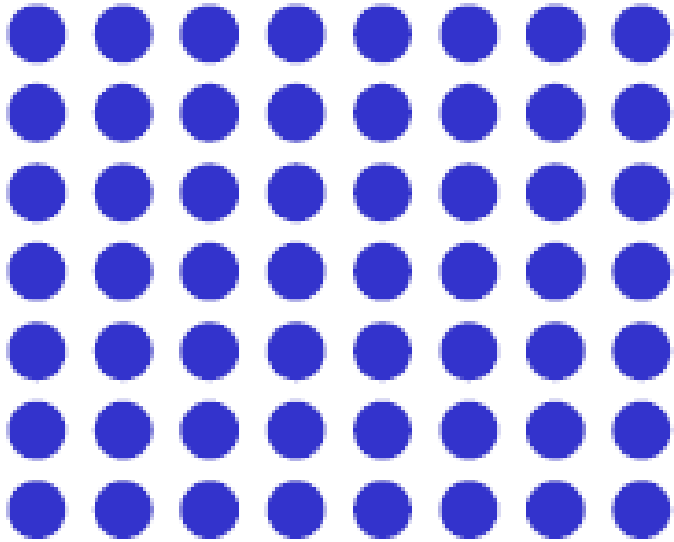
- Perceptual Grouping in humans
  - Gestalt perceptual grouping laws, describe grouping cues of humans
- Image segmentation (“Pixel Grouping”)
  - Clustering
    - Simple agglomerative algorithm
    - K-means
- Histogram based
  - Thresholding
  - Mode-finding
  - Mean shift

From Images to Objects



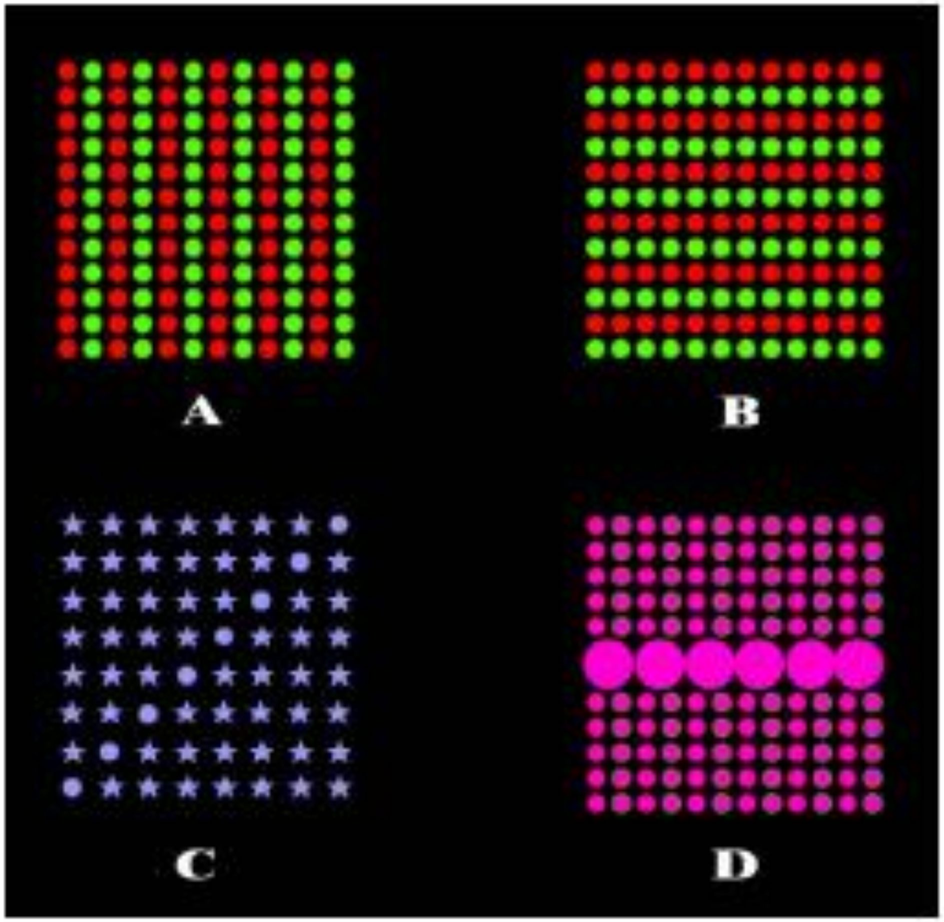
- Humans do not perceive the world as a collection of individual “pixels” but rather as a collection of objects and surfaces
- For many applications, it is useful to segment or group image pixels into blobs which are perceptually meaningful
  - Hopefully belong to the same “object” or surface
- How to do this without (necessarily) object recognition?
  - Subjective problem, but has been well-studied
  - Gestalt Laws seek to formalize this
    - Proximity, similarity, continuation, closure, common fate

Grouping



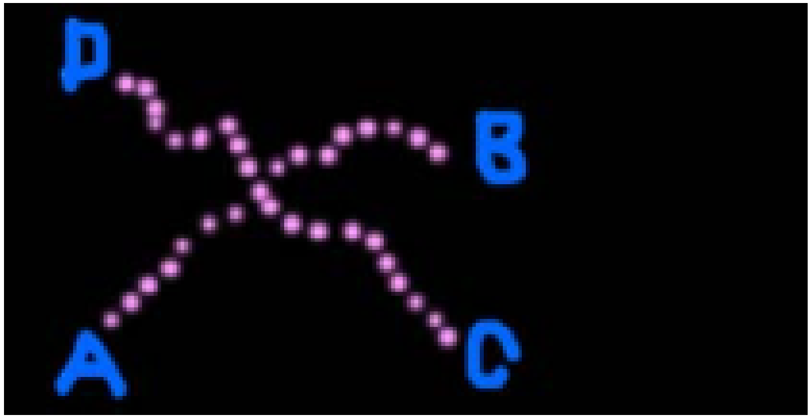
- Most human observers would report no particular grouping

Gestalt Principles of Grouping: Common Form (includes color and texture)



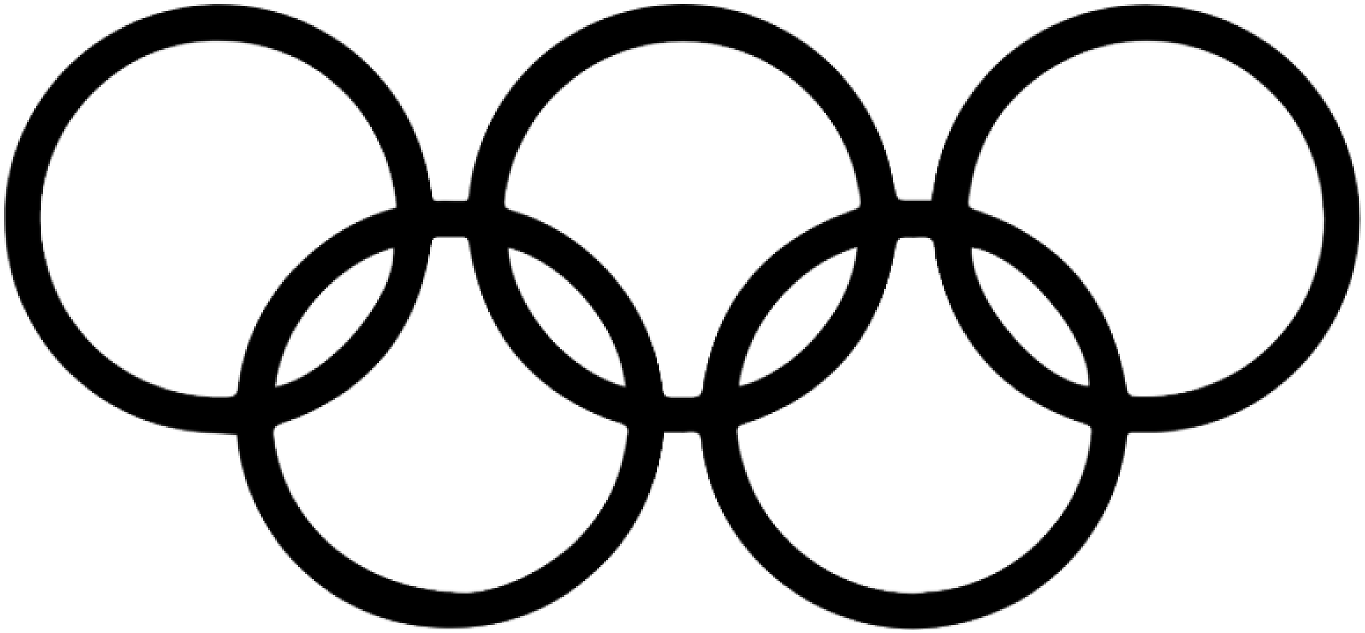
Gestalt Principles of Grouping: Proximity

Gestalt Principles of Grouping: Good Continuation

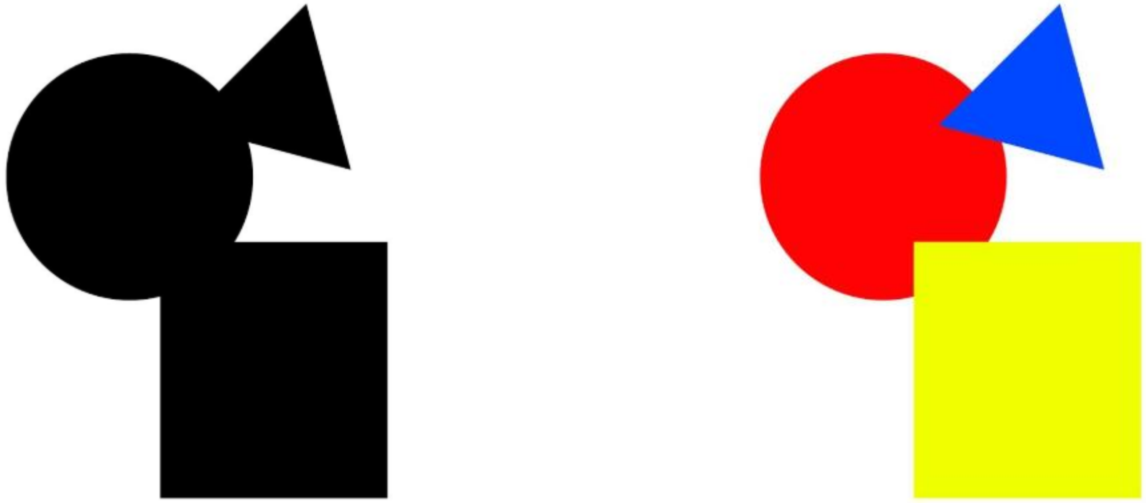


Gestalt Principles of Grouping: Figure/Ground

Gestalt Principles of Grouping: Symmetry

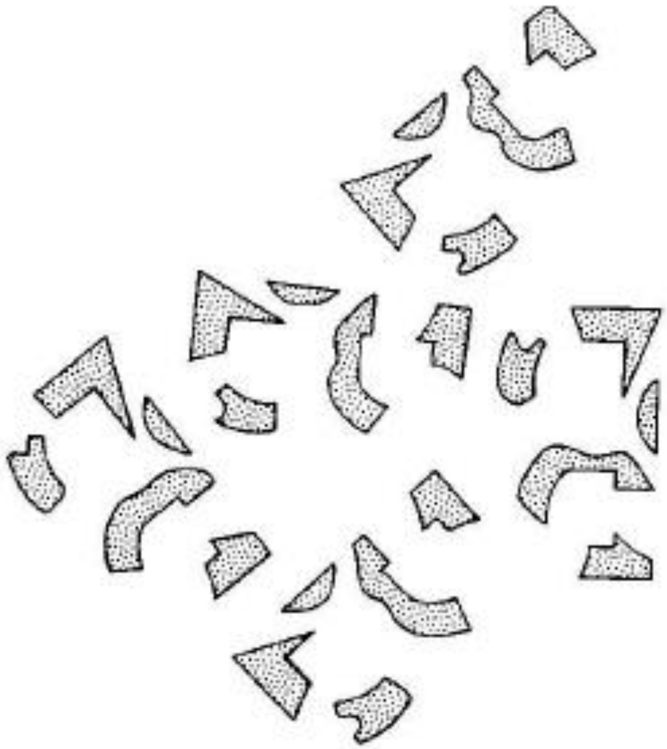


Gestalt Principles of Grouping: Symmetry

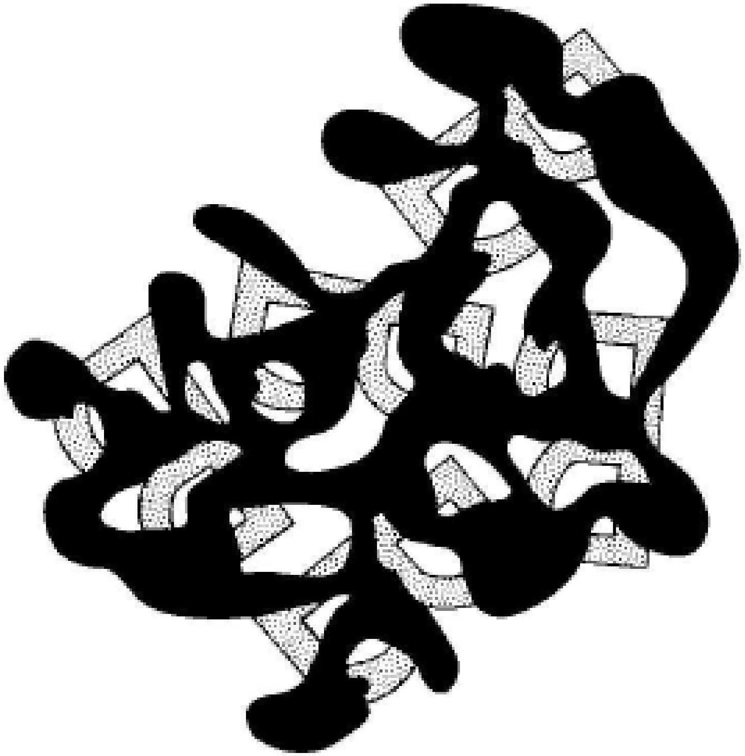


Gestalt Principles of Grouping: Closure

Gestalt Principles of Grouping: Closure



Gestalt Principles of Grouping: Closure



Higher level Knowledge

Take Home Message

- We perceive the world in terms of objects, not pixels
- What forms an object is determined by regularities and non-trivial inference

Human perceptual grouping

- Perceptual grouping has been significant inspiration to computer vision

- Why?
  - Perceptual grouping seems to rely partly on the nature of objects in the world
  - This is hard quantity, we hypothesize that human vision encodes the necessary knowledge

Computer Vision: Image Segmentation

- In vision, we usually refer to perceptual organization problem as image segmentation or clustering
- Image segmentation is the operation of partitioning an image into a collection of
  - Regions, which usually cover the whole image
  - Linear structures, such as
    - Line segments
    - Curve segments
  - Into 2D shapes, such as
    - Circles
    - Ellipses
    - Ribbons (long, symmetric, regions)
- Clustering is a more general term than image segmentation
  - Can cluster all sorts of data (usually represented as feature vectors), not just image pixels
    - Web pages, financial records, etc.
  - Clustering is a large area of machine learning (not supervised, that is labels of feature vectors are not known)

Example 1: Region Segmentation

Example 2: Lines and Circular Arcs Segmentation

Image Segmentation

- Image cues are used for grouping/segmentation:
- Pixel-based cues
  - Color
  - Motion
- Region-based cues
  - Texture
  - Region shape
- Contour-based cues
  - curvature

Image Segmentation Approaches

- Approaches can be roughly divided in two groups:

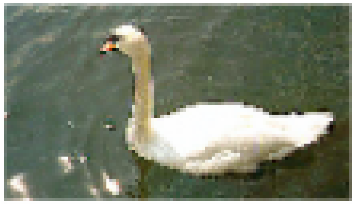
1. Parametric: We have a description of what we want, with parameters:
   - Examples: lines, circles, constant intensity regions, constant intensity regions + Gaussian noise
2. Non-parametric: have constraints the group should satisfy, or optimally criteria.
   - Examples: SNAKEs. Find the closed curve that is smoothest and that also best follows strong image gradients.

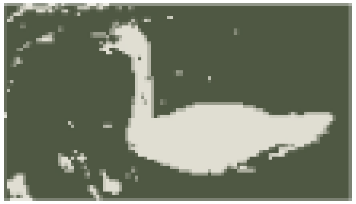
Clustering Algorithms

- Agglomerative
  - Start with each pixel in own cluster
  - Iteratively merge clusters together according to some predefined criterion
  - Stop when reached some stopping condition
- Divisive
  - Start with all pixels in one cluster
  - Iteratively choose and split a cluster into two according to some predefined criterion
  - Stop when reached some stopping condition
- There are clustering methods which are both agglomerative and divisive

Simplest Agglomerative Clustering based on Color/Intensity

- Initialize: Each pixel is a cluster (region)
- Loop
  - Find two adjacent regions with most similar color (or intensity)
  - Merge to form new region with:
    - All pixels of these regions
    - Average color (or intensity) of these regions
  - Several possibilities for stopping condition
    1. No regions similar (color or intensity differences between all neighboring regions is larger than some threshold, etc.)



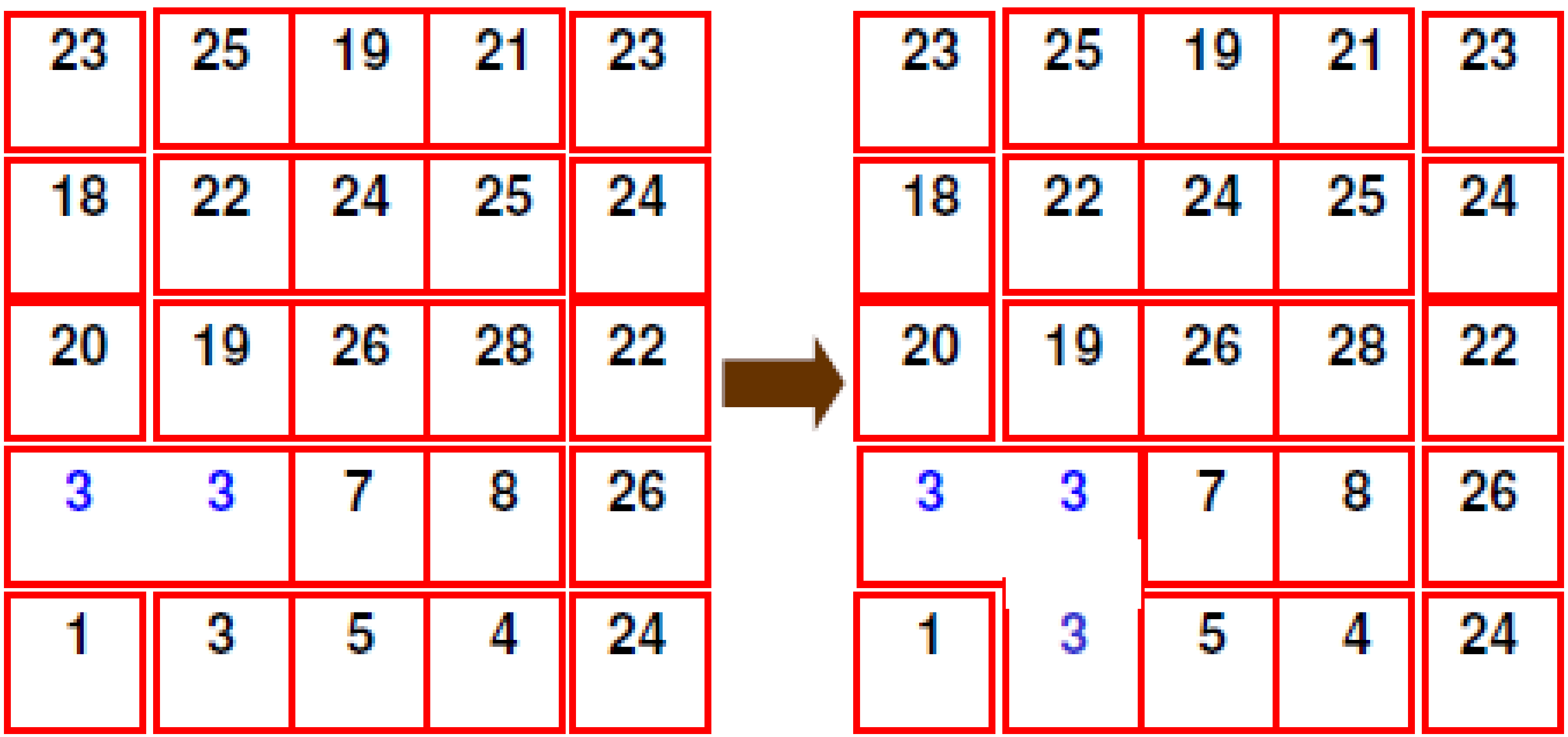


Example: Agglomerative Intensity Based Clustering

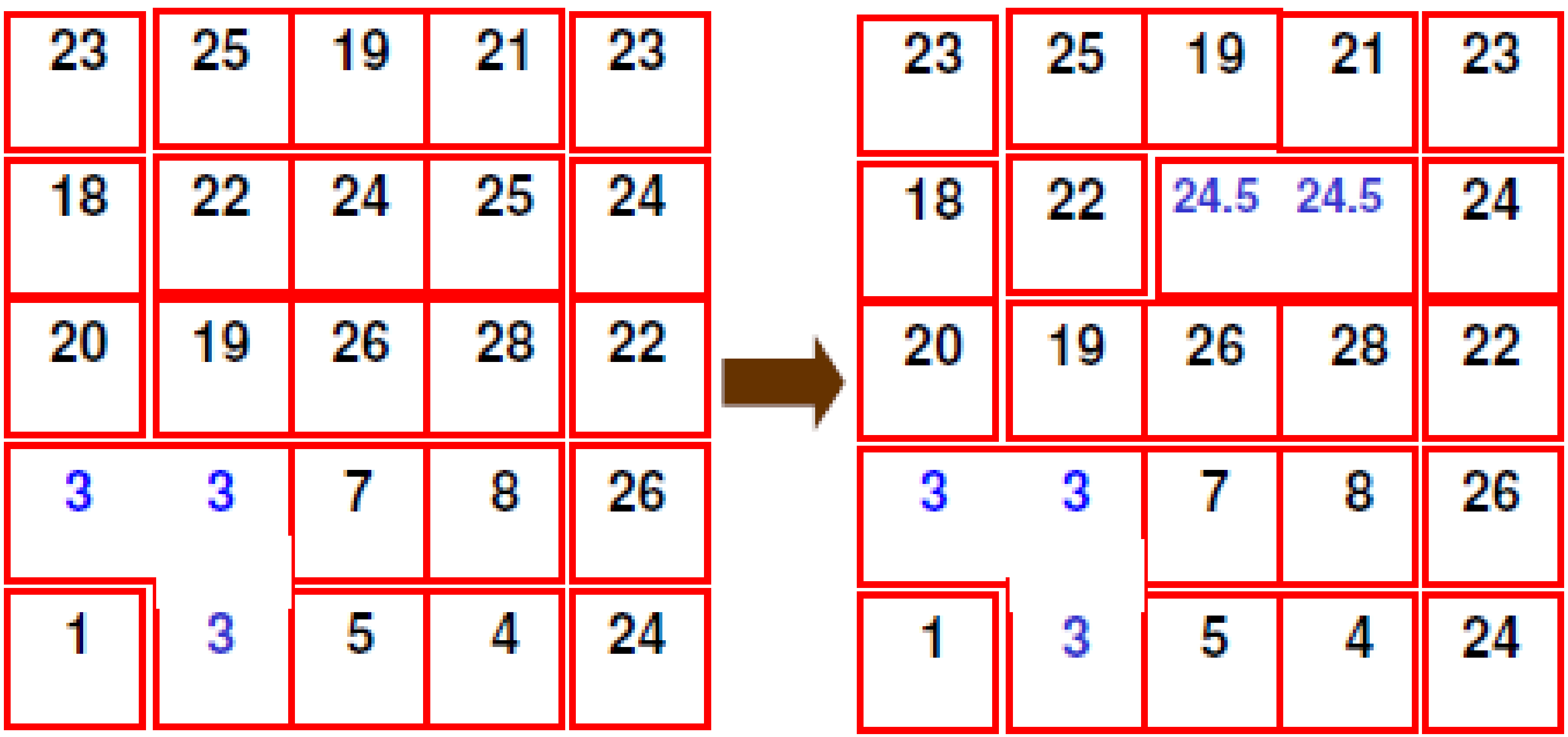
23	25	19	21	23
18	22	24	25	24
20	19	26	28	22
3	3	7	8	26
1	3	5	4	24

23	25	19	21	23
18	22	24	25	24
20	19	26	28	22
3	3	7	8	26
1	3	5	4	24

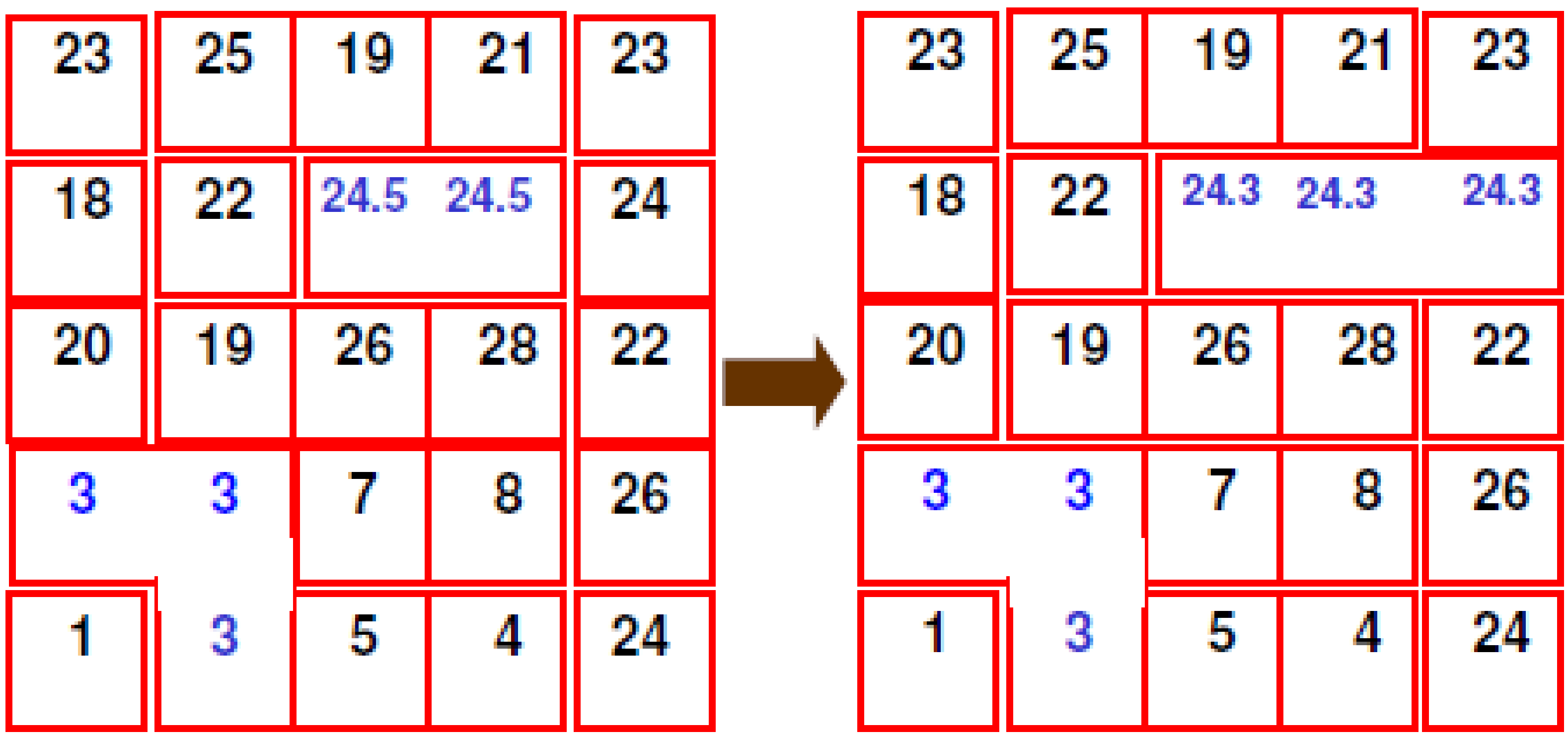
Example: Agglomerative Intensity Based Clustering



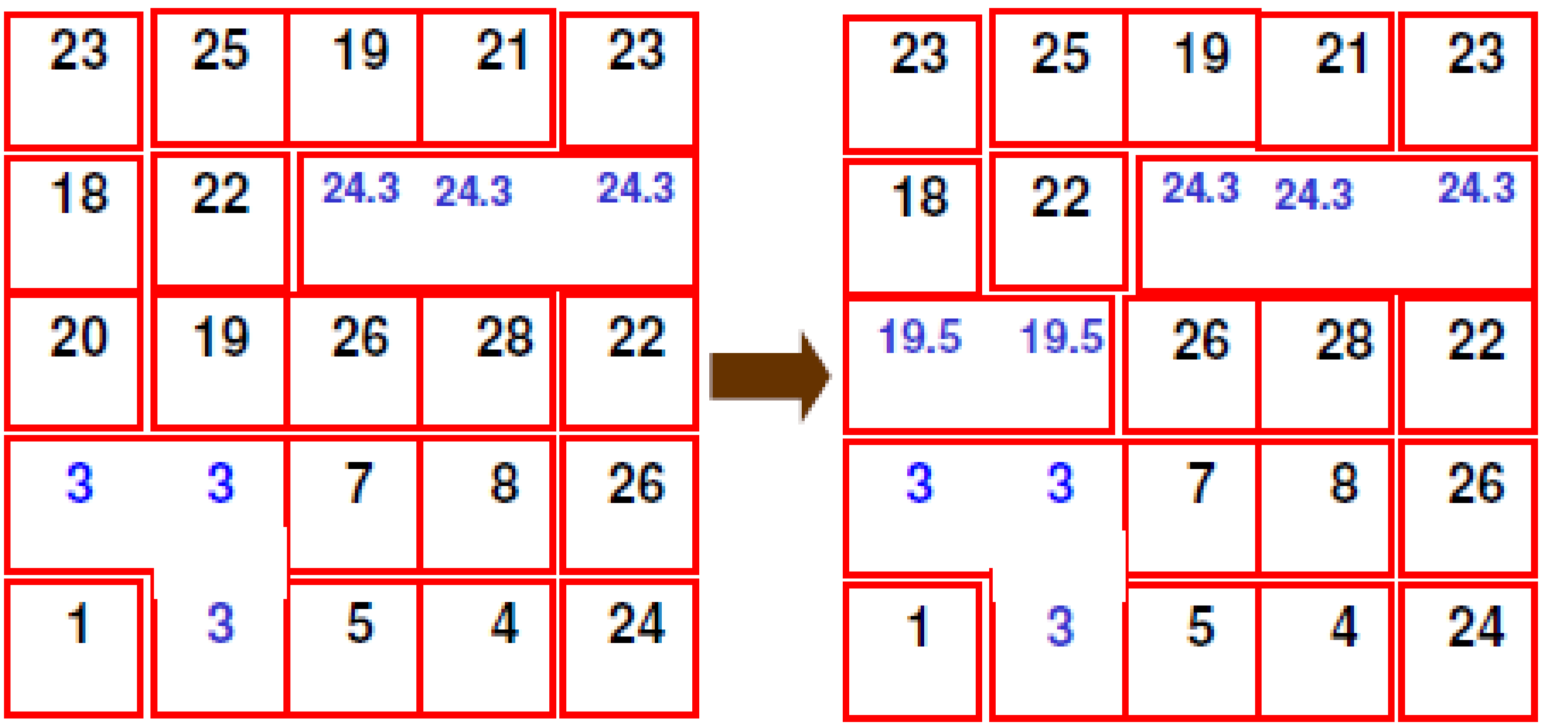
Example: Agglomerative Intensity Based Clustering



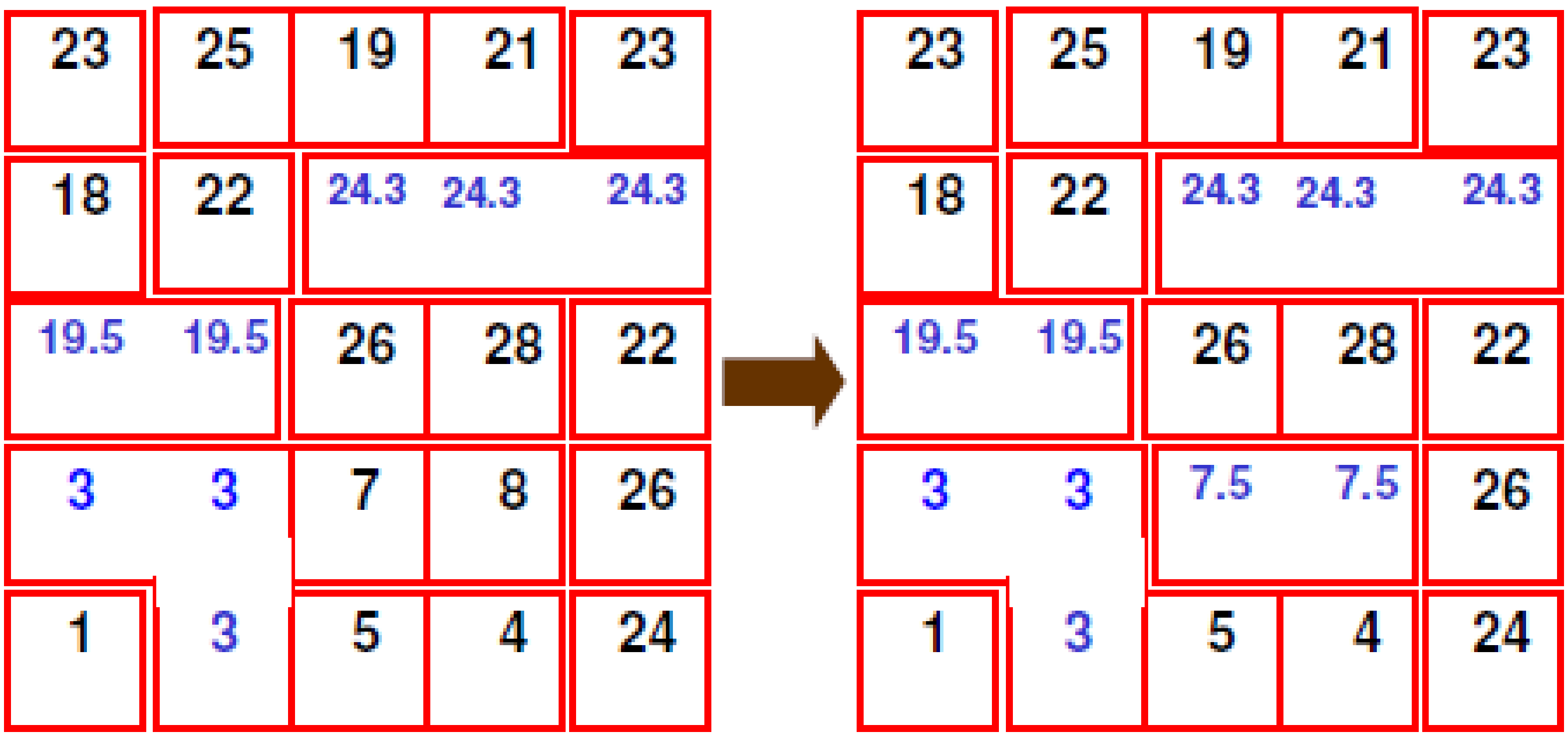
Example: Agglomerative Intensity Based Clustering



Example: Agglomerative Intensity Based Clustering



Example: Agglomerative Intensity Based Clustering



Example: Agglomerative Intensity Based Clustering

Example: Agglomerative Intensity Based Clustering

23	25	19	21	23
18	22	24	25	24
20	19	26	28	22
3	3	7	8	26
1	3	5	4	24

22.9	22.9	22.9	22.9	22.9
22.9	22.9	22.9	22.9	22.9
22.9	22.9	22.9	22.9	22.9
4.25	4.25	4.25	4.25	22.9
4.25	4.25	4.25	4.25	22.9

Agglomerative Clustering: Discussion

- Start with definition of good clusters
- Simple initialization
- Greedy: take steps that seem to most improve clustering
- This is a very general, reasonable strategy
- Can be applied to almost any problem
- But, not guaranteed to produce good quality answer

Clustering for Image Segmentation

- General clustering problem setting:
- Have samples (or points, or feature vectors) $x_1, \ldots, x_n$
  - For segmentation $x_1, \ldots, x_n$ correspond to n image pixels
  - Each x_1 can be
    - Intensity of pixel x_i (for gray image segmentation)
    - Color of pixel x_i (for color image segmentation)
  - For example

(2,44,55)	(22,4,5)	(32,5,6)
(4,4,25)	(6,14,6)	(7,8,91)

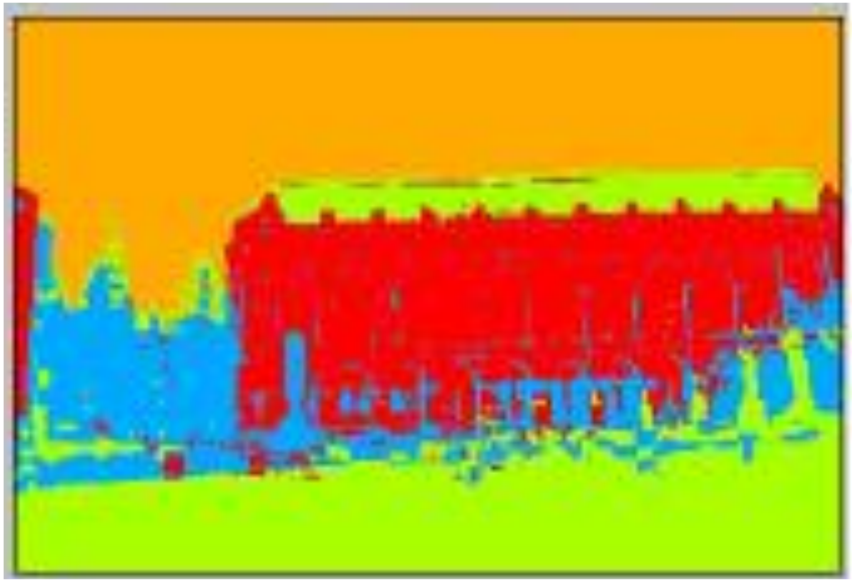
feature vectors for color based clustering

[2,44,55]
[22,4,5]
[32,5,6]
[4,4,25]
[6,14,6]
[7,8,91]

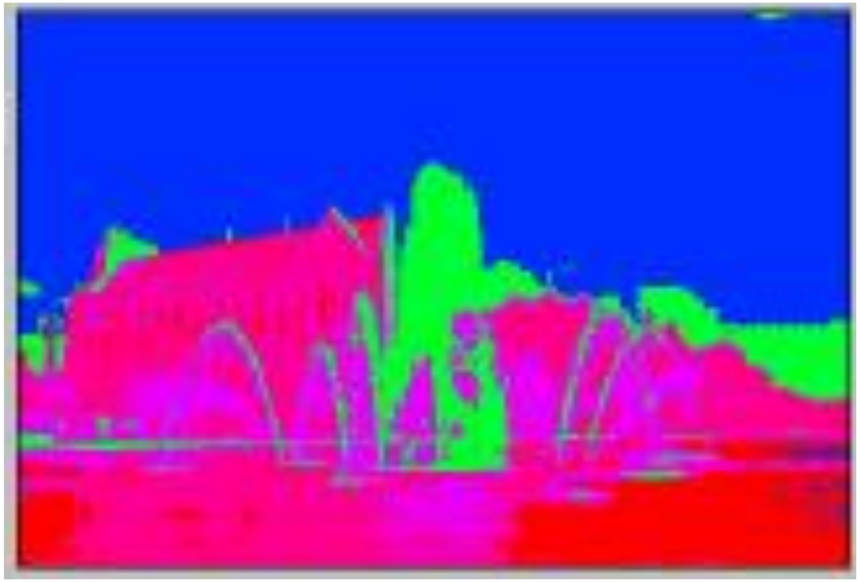
feature vectors for color and coordinates based clustering

[2,44,55,0,0]
[22,4,5,1,0]
[32,5,6,2,0]
[4,4,25,0,1]
[6,14,6,1,1]
[7,8,91,2,1]

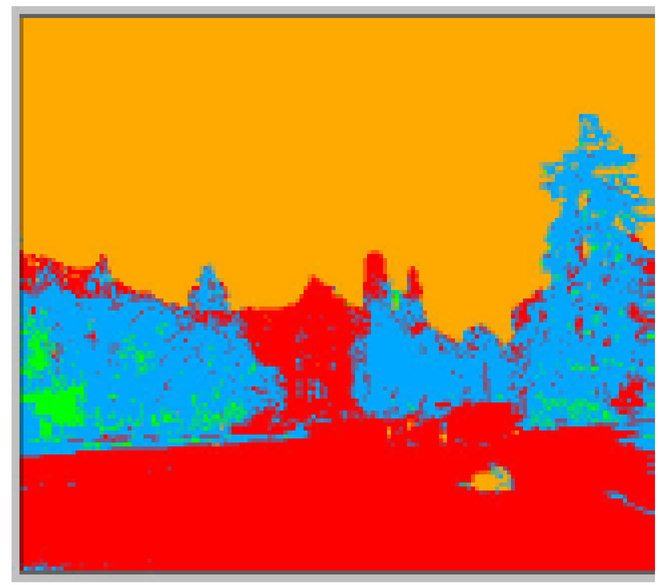
Example 1



Example 2



Example 3



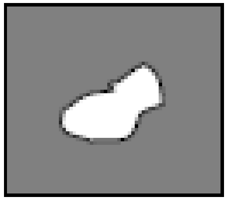
Histogram-Based Segmentation

- Segmentation by Histogram Processing
  - Given image with N colors, choose k
  - Each of K colors defines a region
    - not necessarily contiguous
  - Performed by computing color histogram, looking for modes

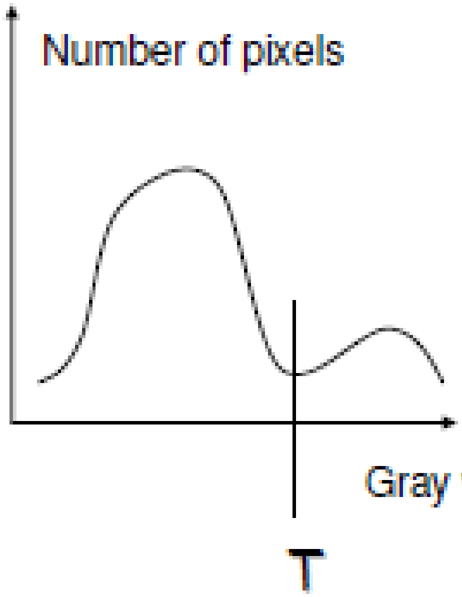
  - This is what happens when you downsample image color range, for instance in Photoshop

Histogram-Based Segmentation

Ex: bright object on dark background:



Histogram



Select threshold

Create binary image:

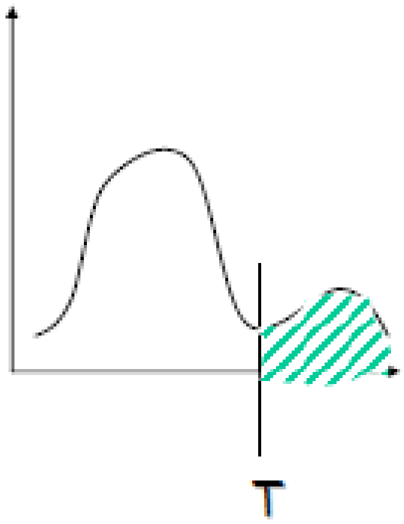
- $I(x,y) < T \Rightarrow O(x,y) = 0$
- $I(x,y) > T \Rightarrow O(x,y) = 1$

How do we select a Threshold?

- Automatic thresholding
  - P-tile method
  - Mode method
  - Peakiness detection
  - Mean-shift

P-Tile Method

- If the size and brightness range of the object is approximately known, pick T s.t. the area under the histogram corresponds to the size of the object:

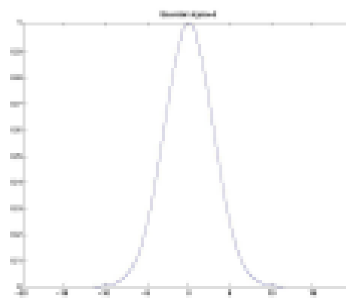


Mode Method

- Model each region as "constant" + noise
- Usually noise is modeled as $N(0, \sigma_j)$

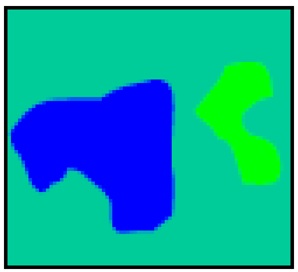
$$\text{If } (x,y) \in R_i \text{ then, } I(x,y) = \mu_i + n_i(x,y)$$

$$p(n_i) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{1}{2}\frac{n_i^2}{\sigma_i^2}}$$

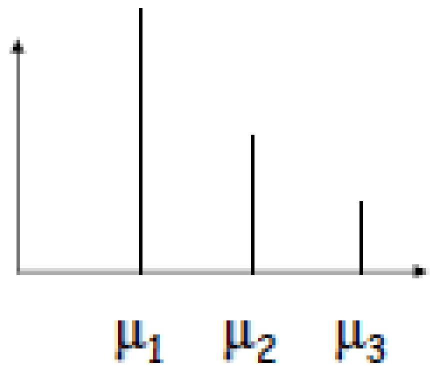


$$E(n_i) = 0 \qquad E(n_i^2) = \sigma_i^2$$

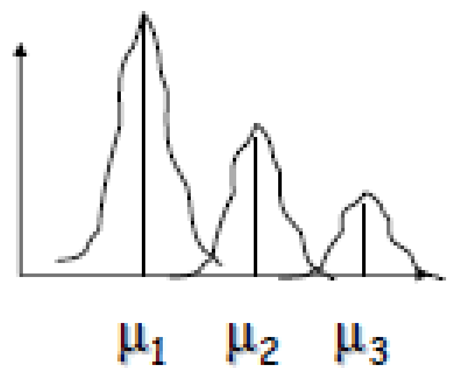
Example: Image with 3 regions



Ideal histogram:

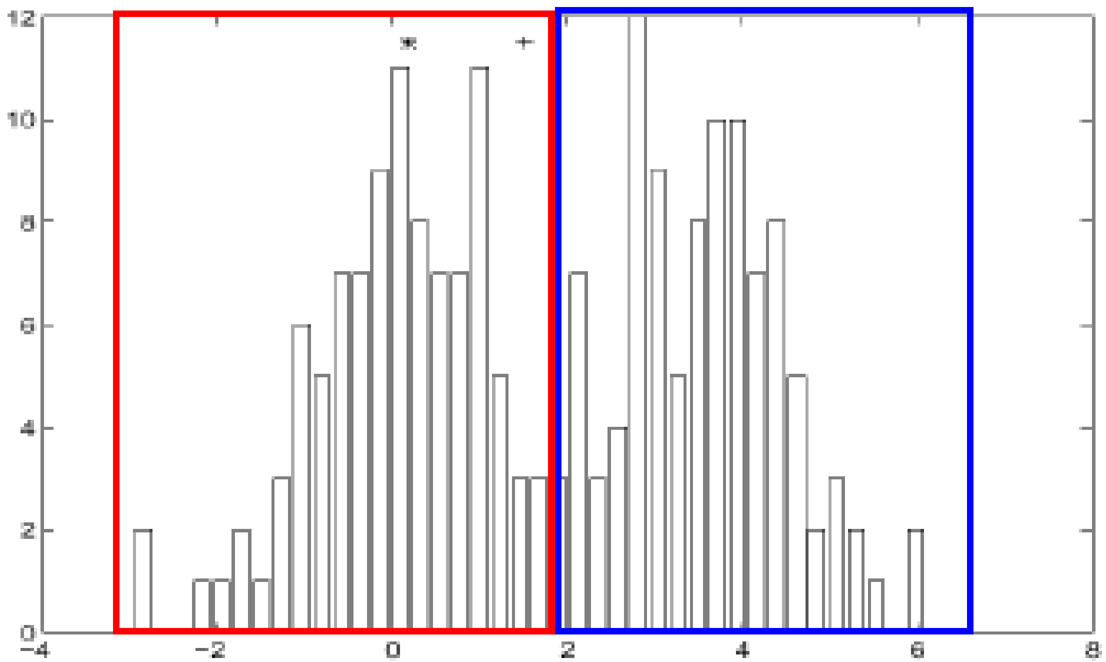


Add noise:



The valleys are good places for thresholding to separate regions.

Finding Modes in a Histogram

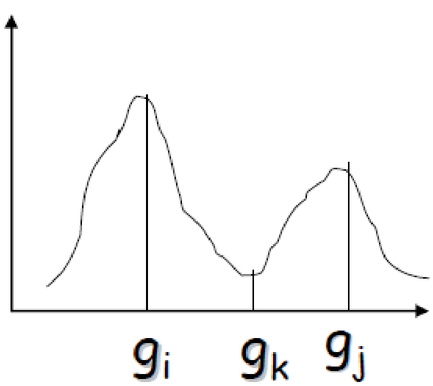


How Many Modes Are There?

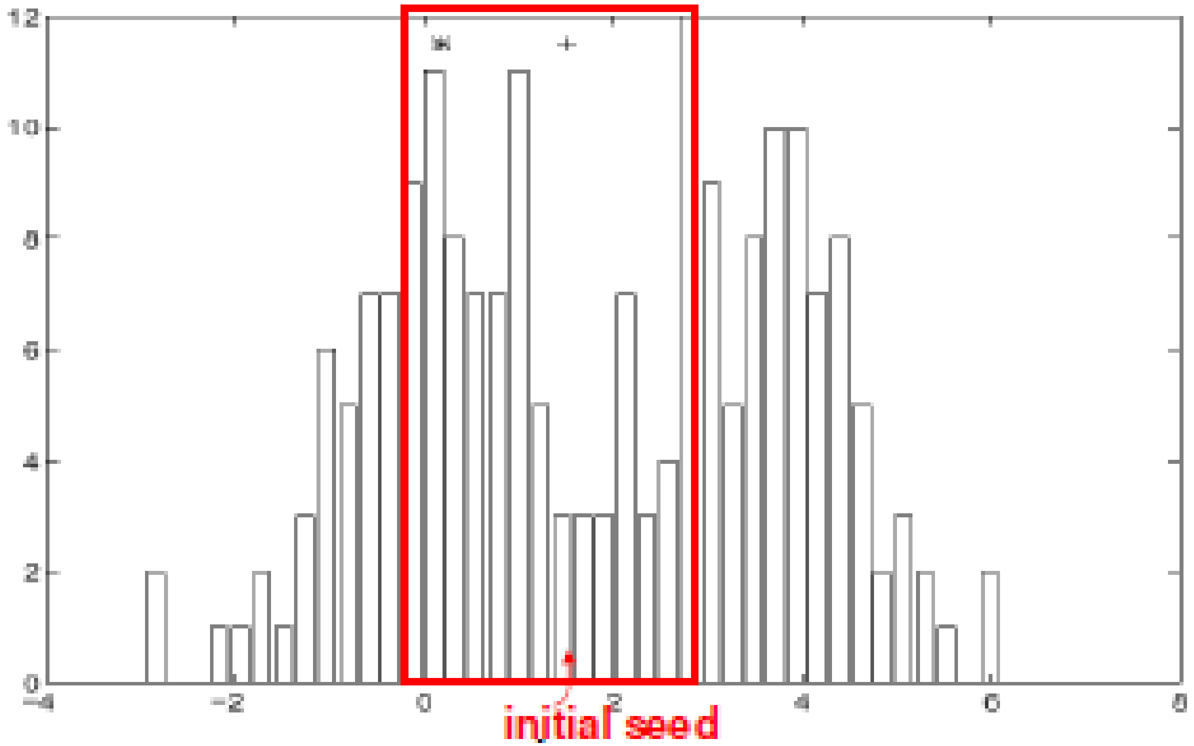
- Easy to see, hard to compute
- Not a trivial problem

Peakiness detection algorithm

- Find the two highest Local Maxima at a minimum distance apart: $g_i \ and \ g_j$
- Find lowest point between them: g_k
  - Measure "peakiness": $\min\left(H(g_i), H\left(g_j\right)\right) / H(g_k)$
- Find (g_i, g_j, g_k) with highest peakiness



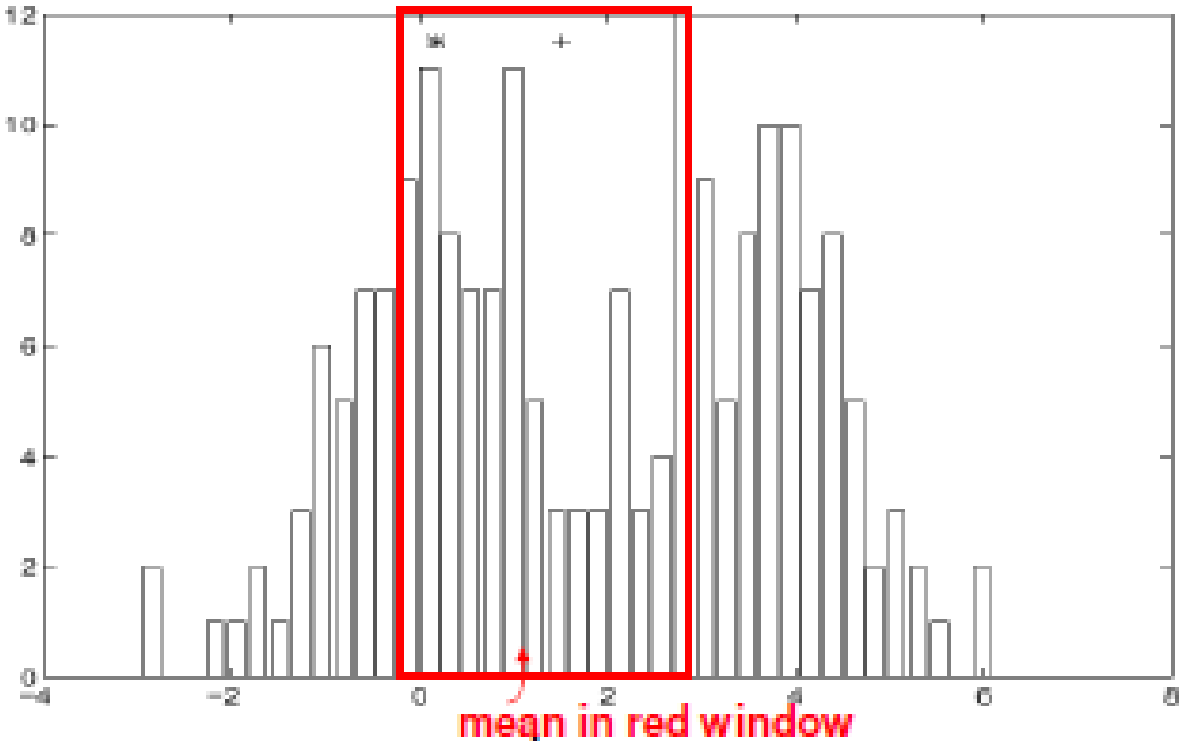
Mean Shift (Comaniciu & Meer)



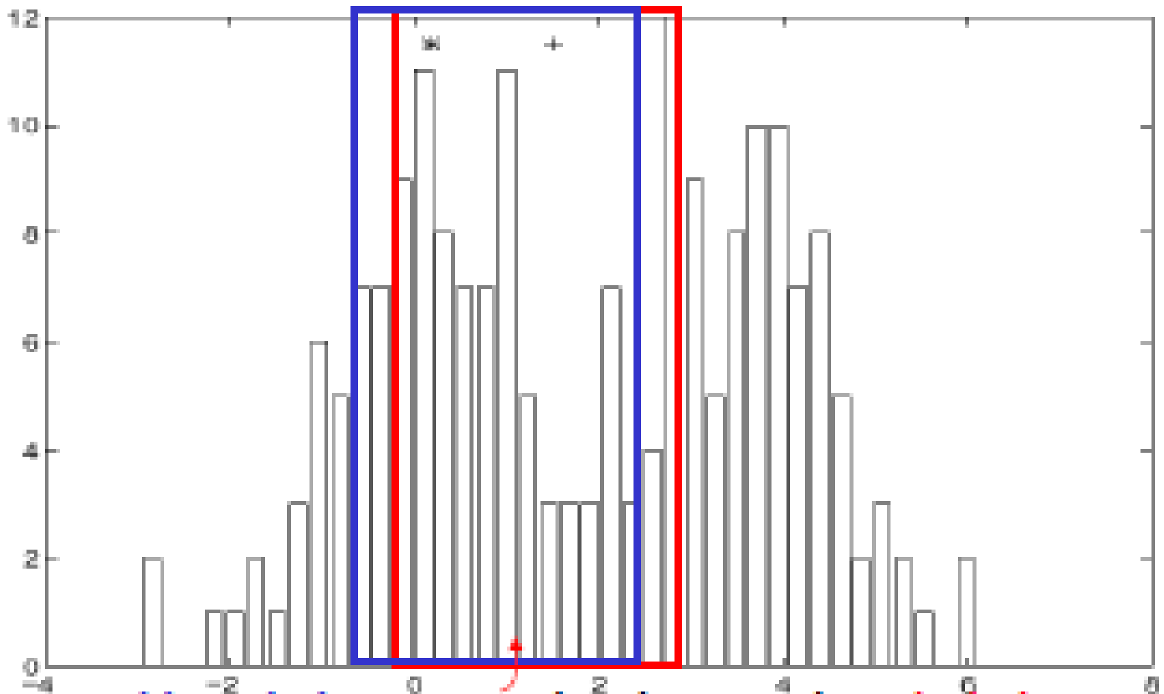
- Iterative Mode Search

1. Initialize random seed, and fixed window
2. Calculate center of gravity of the window (the “mean”)
3. Translate the search window to the mean
4. Repeat Step 2 until convergence

Mean Shift (Comaniciu & Meer)

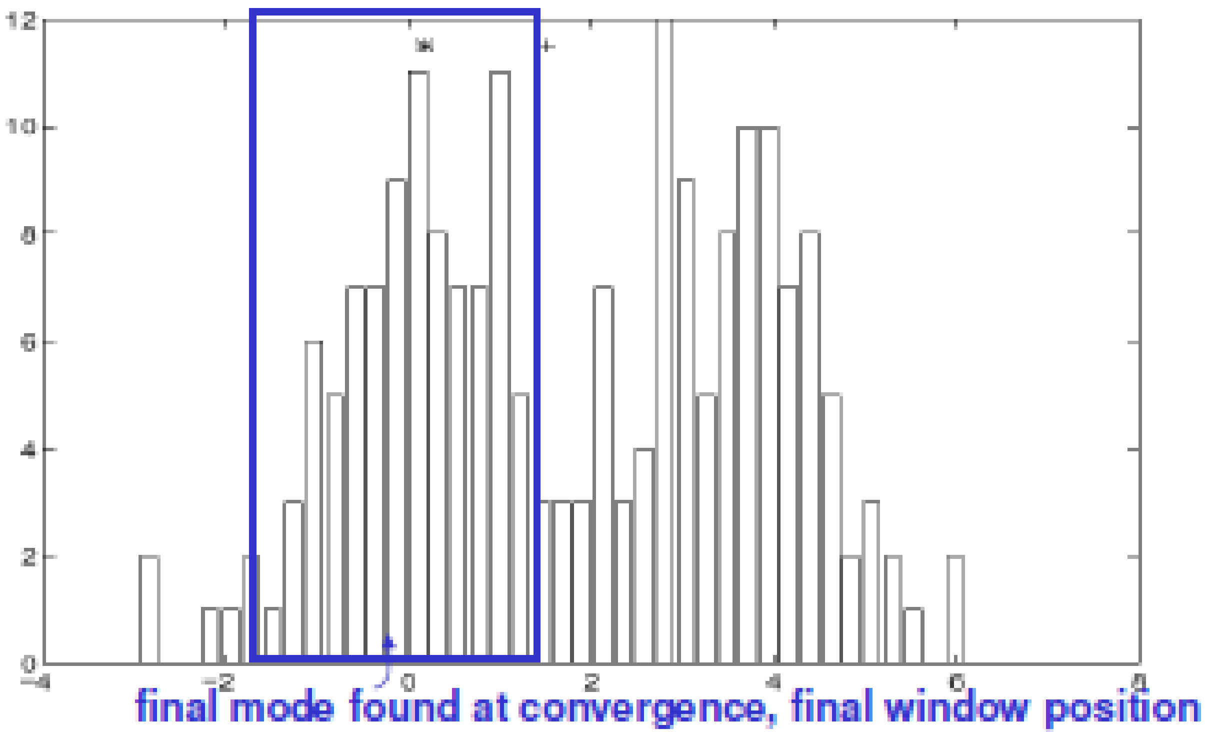


Mean Shift (Comaniciu & Meer)



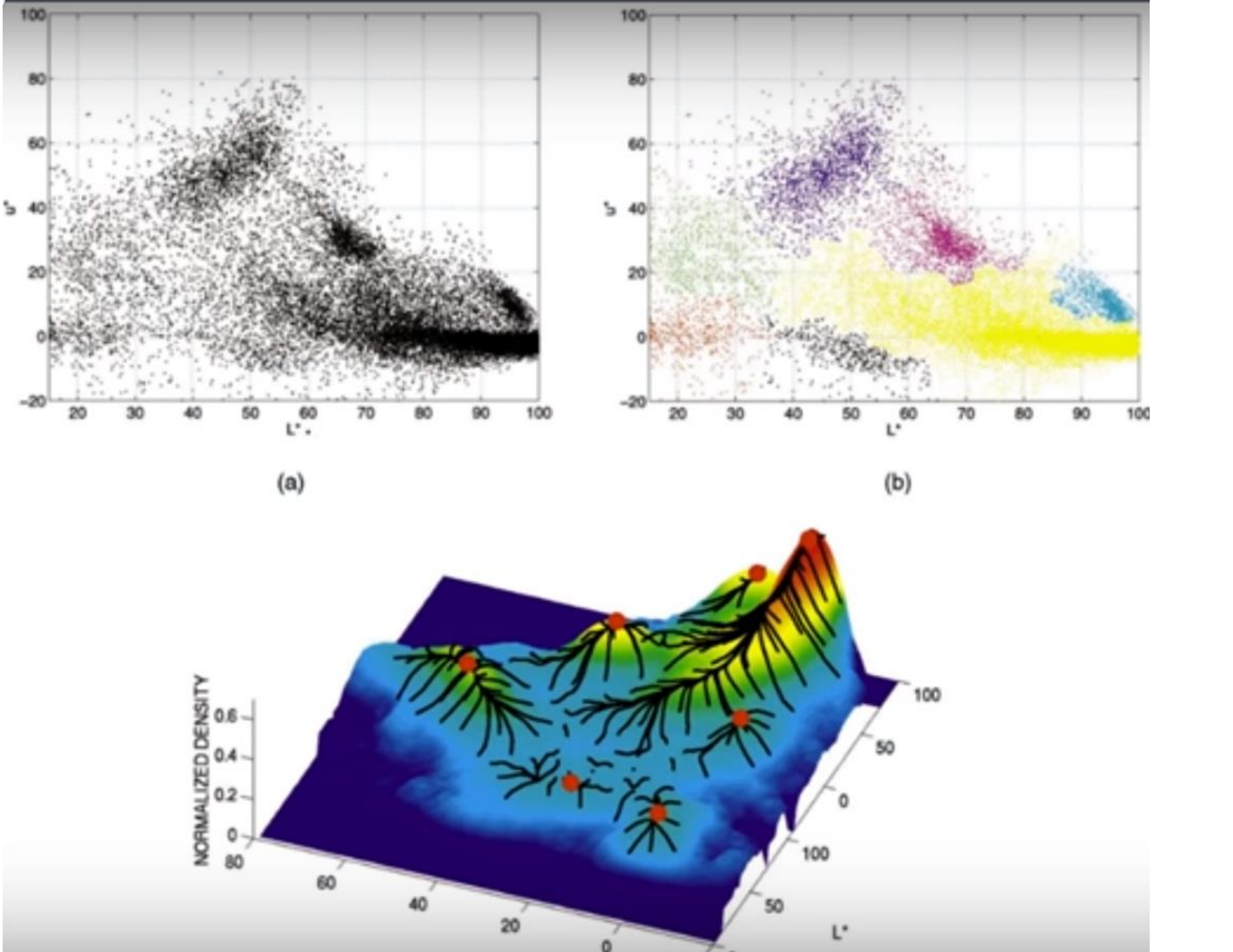
blue window centered at the mean in red window

Mean Shift (Comaniciu & Meer)



Algorithm Mean Shift to find the histogram Peak

1. Choose a Window size (for example 5)
2. Choose the initial location of the search window
3. Compute the mean location in the search window
4. Center the window at the location computed in 3
5. Repeat steps 3 and 4 until convergence



100
80
60
40
20
0
-20
20 30 40 50 60 70 80 90 100
L*
(a)
(b)
NORMALIZED DENSITY
0.6
0.4
0.2
0
80
60
40
20
0
100
50
0
100
50
L*

Algorithm: Mean Shift for Image Segmentation

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

Algorithm: Mean Shift for Image Segmentation

- Find image histogram, choose window size
- Choose initial location of search window:
  - Randomly select a number M of image pixels
  - Find the average value in a 3x3 window for each of these pixels
  - Set the center of the window to the value with largest histogram count
- Apply mean shift to find the window peak
- Remove pixels in the window from the image and the histogram
  - Say peak was at intensity 44 and window size is 5
  - Pixels with intensities between [39,49] become one group
  - Remove these pixels from further consideration
- Repeat steps 2 to 4 until no pixels are left

Algorithm Mean Shift

- Previous slides assumed features are gray pixel values
  - Feature vectors are one dimensional
- Can do the same thing for color images
  - Feature vectors are 3 dimensional
- Can also include the (x,y) pixel coordinates
  - Feature vectors are 5 dimensional
- In all these cases, taking a window around feature vector ***y*** corresponds to taking all feature vectors ***x*** s.t.

$$\|y - x\|^2 \le r$$

- New window center is shifted from ***y*** to

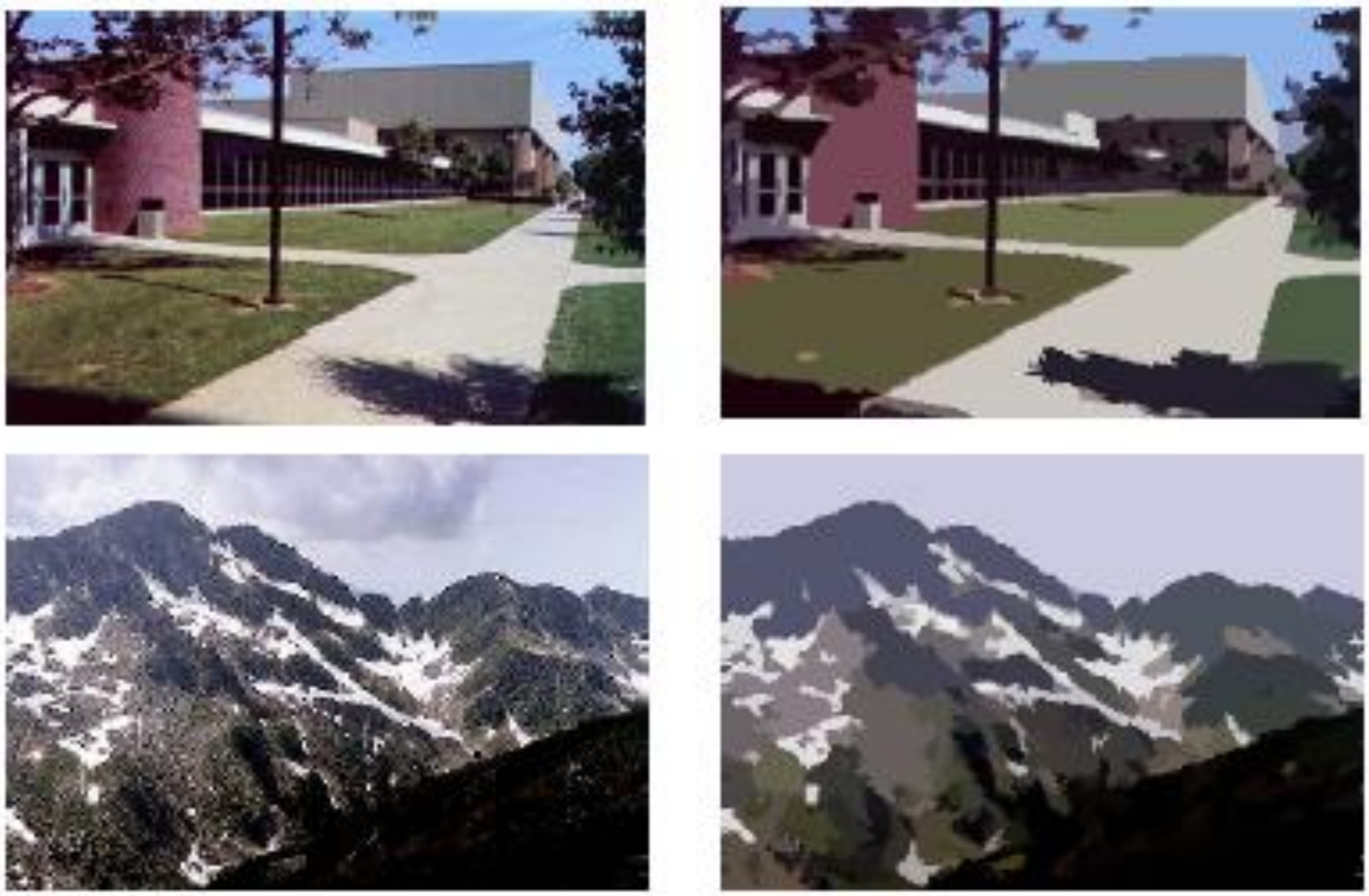
$$\frac{1}{n}\sum_{x \in S} x$$

- Where S is the set of all feature vectors x s.t. $\|y - x\|^2 \le r$, and n is the size of S

Mean Shift Segmentation: Example

The Mean Shift segmentation is a local homogenization technique that is very useful for damping shading or tonality differences in localized objects.

Mean Shift Segmentation: Example



Mean Shift Segmentation: Example

Mean shift: Strengths and Weaknesses

Strengths

- Does not assume any prior shape (e.g. elliptical) on the data structure
- Can handle arbitrary feature spaces
- Only one parameter to choose
  - h the window size

Weaknesses

- The window size is not trivial
- Inappropriate window size can cause modes to be merged (giving too few segments) or generate additional shallow modes (giving too many segments)
- There are adaptive window size extensions