SAIR

Spatial AI & Robotics Lab

CSE 473/573-A

W8: MORPHOLOGY & SEGMENTATION

Chen Wang

Spatial AI & Robotics Lab

Department of Computer Science and Engineering

University at Buffalo The State University of New York

SAIR

Spatial AI & Robotics Lab

IMAGE PROCESSING

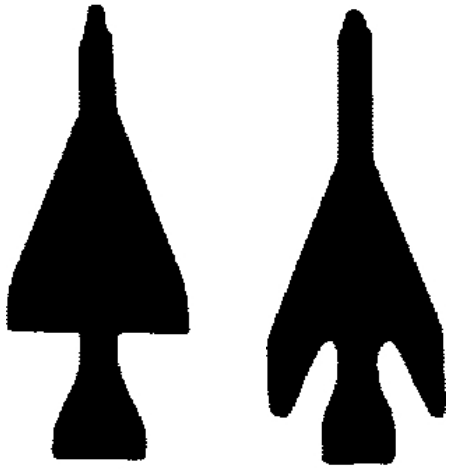
Morphology

Credit: Junsong Yuan

Morphology – Introduction

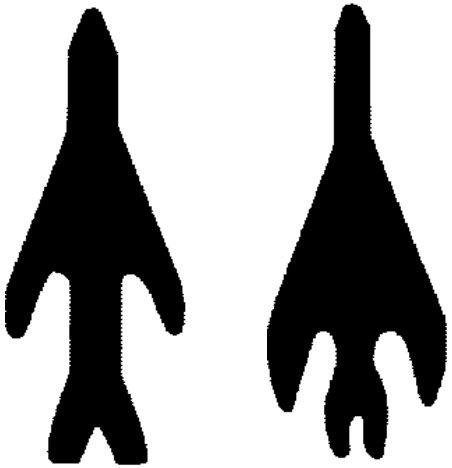
Looking at these images……

What is interesting, important or useful information we care about?

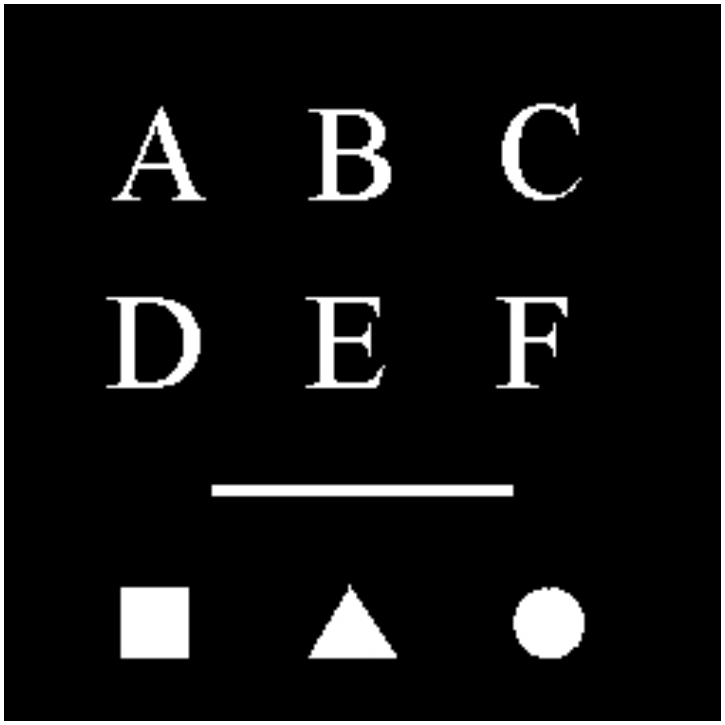


The pixel value of the image is not important as there are only two different values.

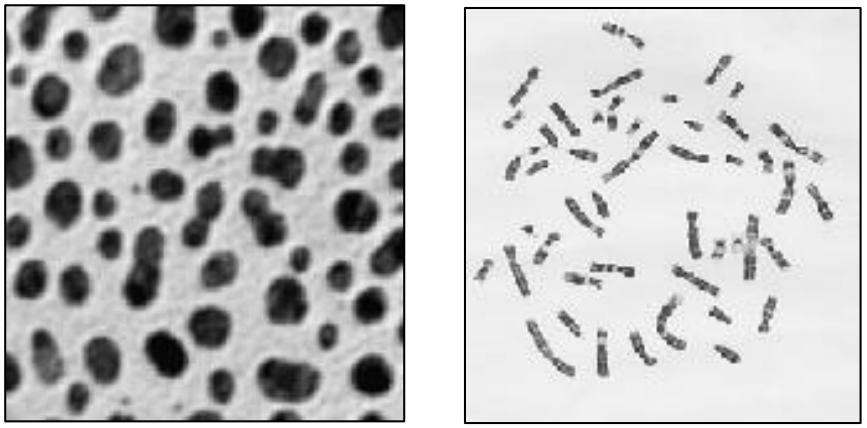
- Region shape and boundaries of object are important.



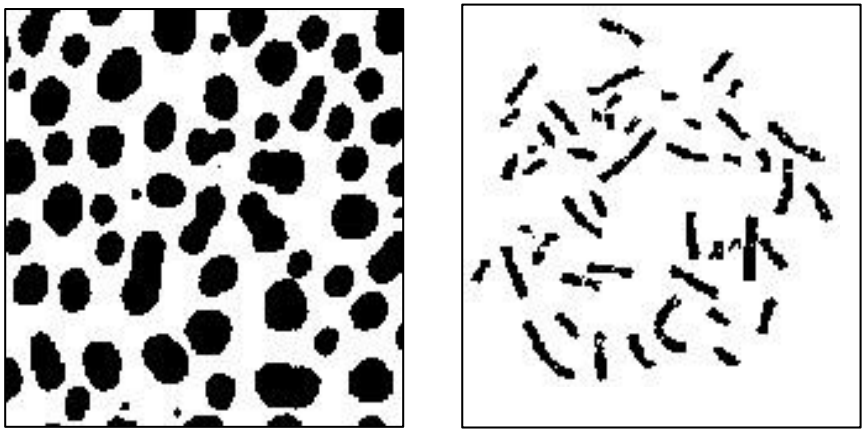
- Form and structure can be represented by object pixel set.



Morphology – Introduction



Grayscale Images



Binary Images

Image analysis needs to measure the characteristics of objects in the images.

Geometric measurements are important objects characteristics

- location, orientation, area, length of perimeter

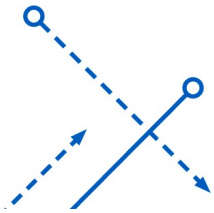
These geometric characteristics are often easier to be measured from binary images.

Morphology – Introduction

- Visual perception requires image processing to make explicit **shape information**.

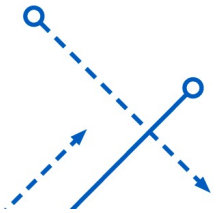
- **Goal:**
  - Distinguish meaningful shape information from irrelevant one.

- The vast majority of shape processing and analysis techniques are **based on designing a shape operator** which satisfies desirable properties.



Morphology –Introduction

- Morphology deals with form and structure
- Mathematical morphology is a tool for extracting image components useful in:
  - representation and description of region shape (e.g. boundaries)
  - pre- or post-processing (filtering, thinning, etc.)
- Morphological operations usually follow a segmentation task or an edge detection task.
  - Thus, often operate on binary images.
- Based on set theory and logic operations



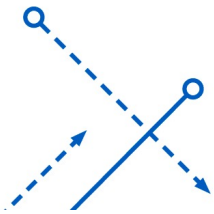
Morphology –Set Theory

- A two dimensional integer space is denoted by $\mathbf{Z}^2$.
- An element in this space has two components $a=(a_1, a_2)$.
- For image representation, $a=(a_1, a_2)$ are the x- and y-coordinates of a pixel.
- Let A be a set in $\mathbf{Z}^2$. If $a=(a_1, a_2)$ is an element of A, we denote

$$a \in A$$

- If not, then

$$a \notin A$$



Morphology –Set Theory

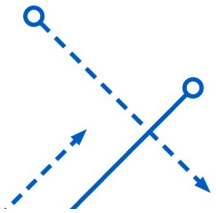
- $\varnothing$ denotes null (empty) set

- An example that specifies a set C:

$$C = \{ w \mid w = a+d,\ a \in A \},\ d = (8, 5).$$

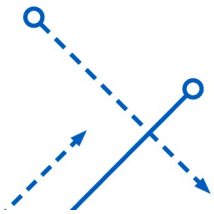
- If a set A is a subset of B, we denote:

$$A \subseteq B$$

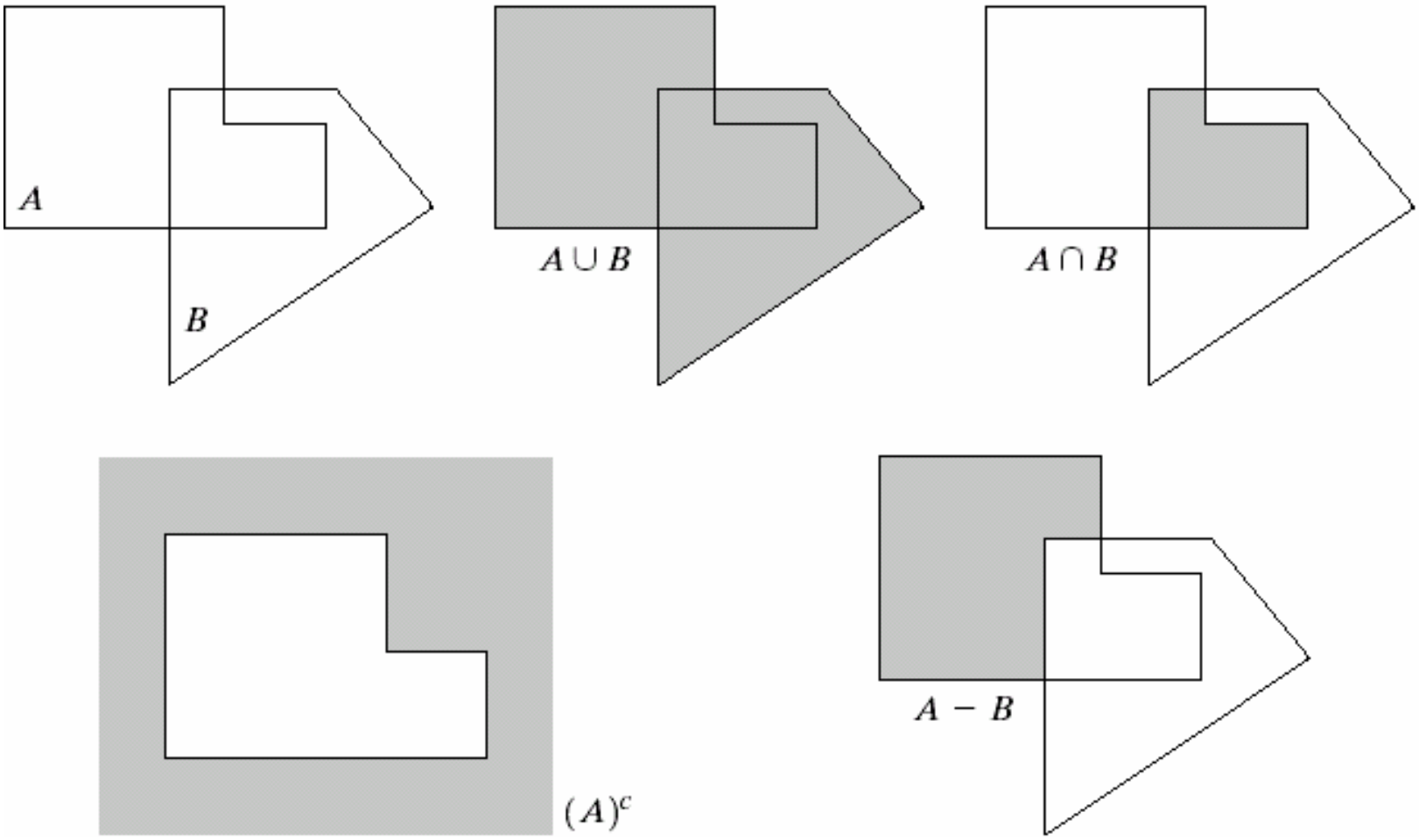


Morphology –Set Theory

- Union of A and B: $C = A \cup B$
- Intersection of A and B: $D = A \cap B$
- Disjoint sets: $A \cap B = \varnothing$
- Complement of A: $A^c = \{w \mid w \notin A\}$
- Difference of A and B: $A - B = \{w \mid w \in A,\ w \notin B\}$
  $= A \cap B^c$



Morphology –Set Theory

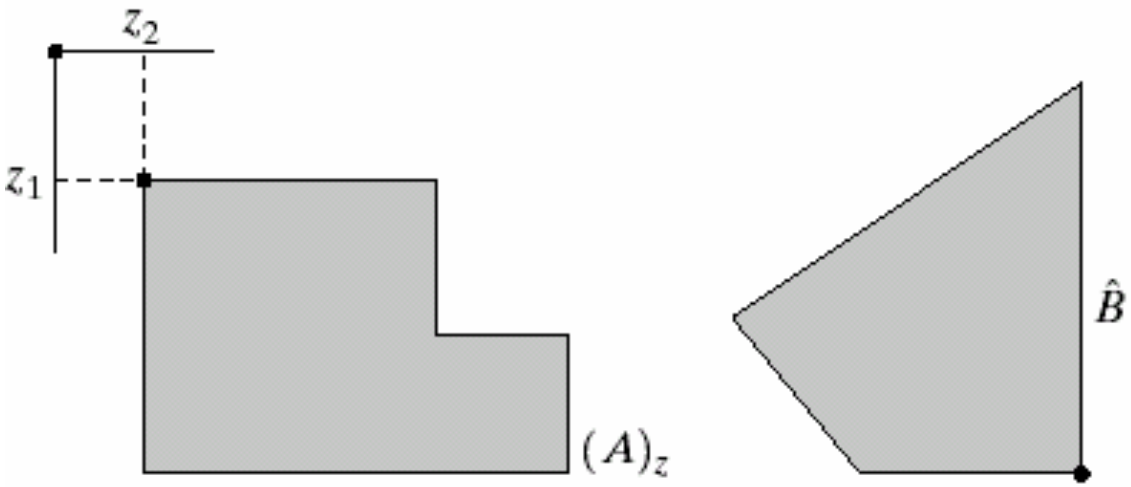


a b c
d e

(a) Two sets A and B. (b) The union of A and B. (c) The intersection of A and B. (d) The complement of A. (e) The difference between A and B.

Morphology –Set Theory

- Translation of A by $z=(z_1, z_2)$: $(A)_z = \{c \mid c = a+z, a \in A\}$



a b

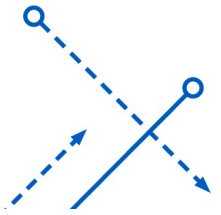
(a) Translation of A by z.
(b) Reflection of B.

- Reflection of B: $\hat{B} = \{w \mid w = -b, \ b \in B\}$

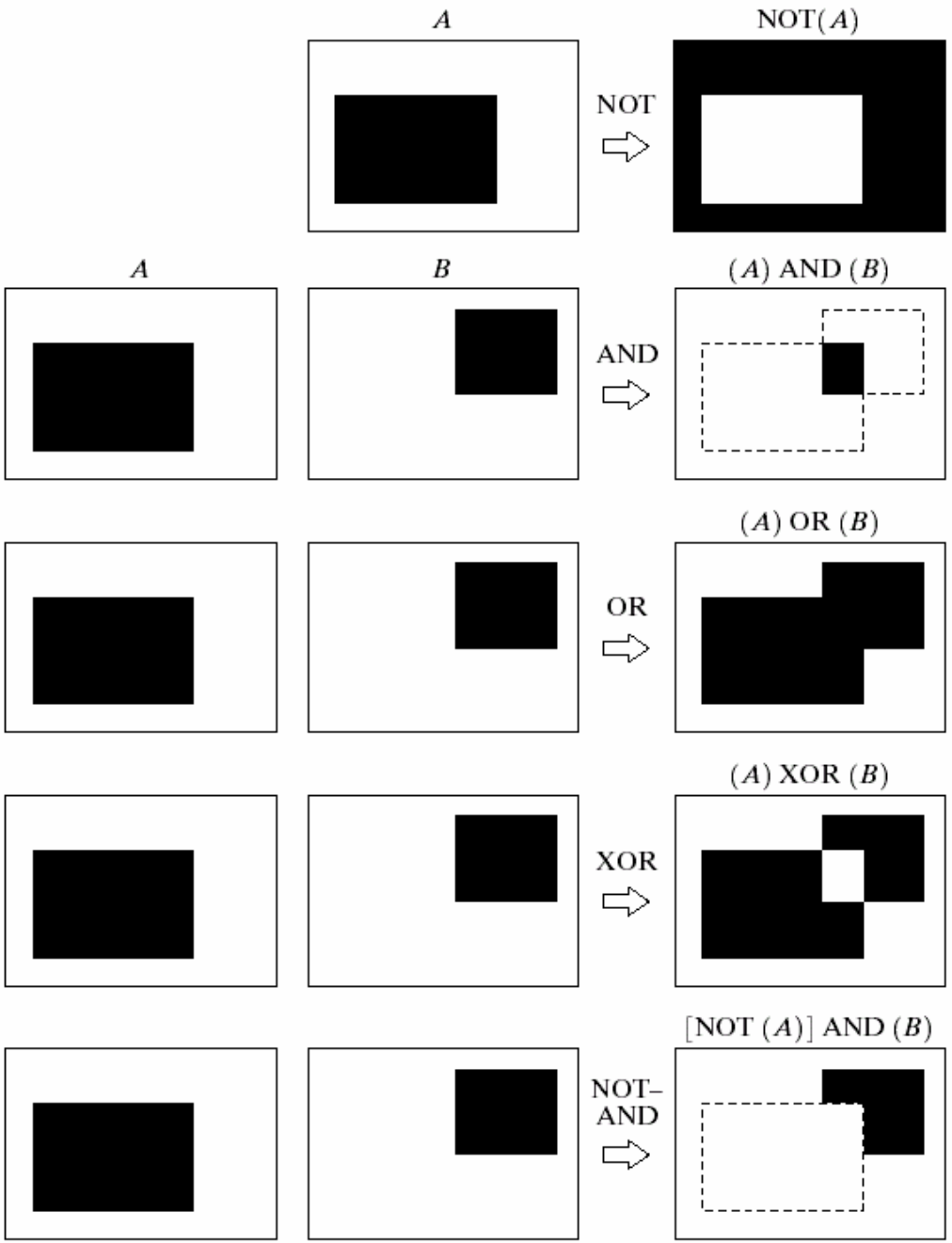
Morphology –Set Theory

- Three basic logical operations

p	q	p AND q (also $p \cdot q$)	p OR q (also $p + q$)	NOT (p) (also $\bar{p}$)
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0



Morphology



Some logic operations between binary images. Black represents binary 1s and white binary 0s in this example.

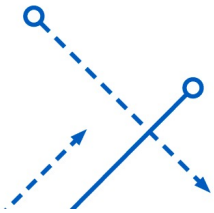
Morphology

a b c
d e f

(a) Binary image A. (b) Binary image B. (c) Complement ~A. (d) Union A | B. (e) Intersection A & B. (f) Set difference A & ~B.

Morphology – Operators

- Primary morphological operations are Dilation and Erosion
- More complicated morphological operators such as Opening and Closing can be designed by combining erosions and dilations
- Opening generally **smoothes the contour** of an image and **eliminates protrusions**
- Closing **smoothes sections of contours**, but it generally **fuses** breaks, holes and gaps



Morphology – Dilation

- Dilation of *A* by *B*, denoted by $A \oplus B$, is defined as:

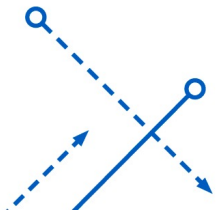
$$A \oplus B = \{ z \mid \left[\left(\hat{B} \right)_z \cap A \right] \neq \varnothing \}$$

- Interpretation:

  Obtaining the reflection of B about its origin and then shifting by z.

  Dilation of A by B is the set of all z displacements such that $\hat{B}$ and A overlap by at least one nonzero element.

- B is called the structuring element in Dilation.



Morphology – Dilation

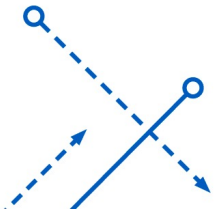
- Dilation of A by B can also be expressed as:

$$A \oplus B = \left\{ z \mid \left[\left(\hat{B} \right)_z \cap A \right] \subseteq A \right\}$$

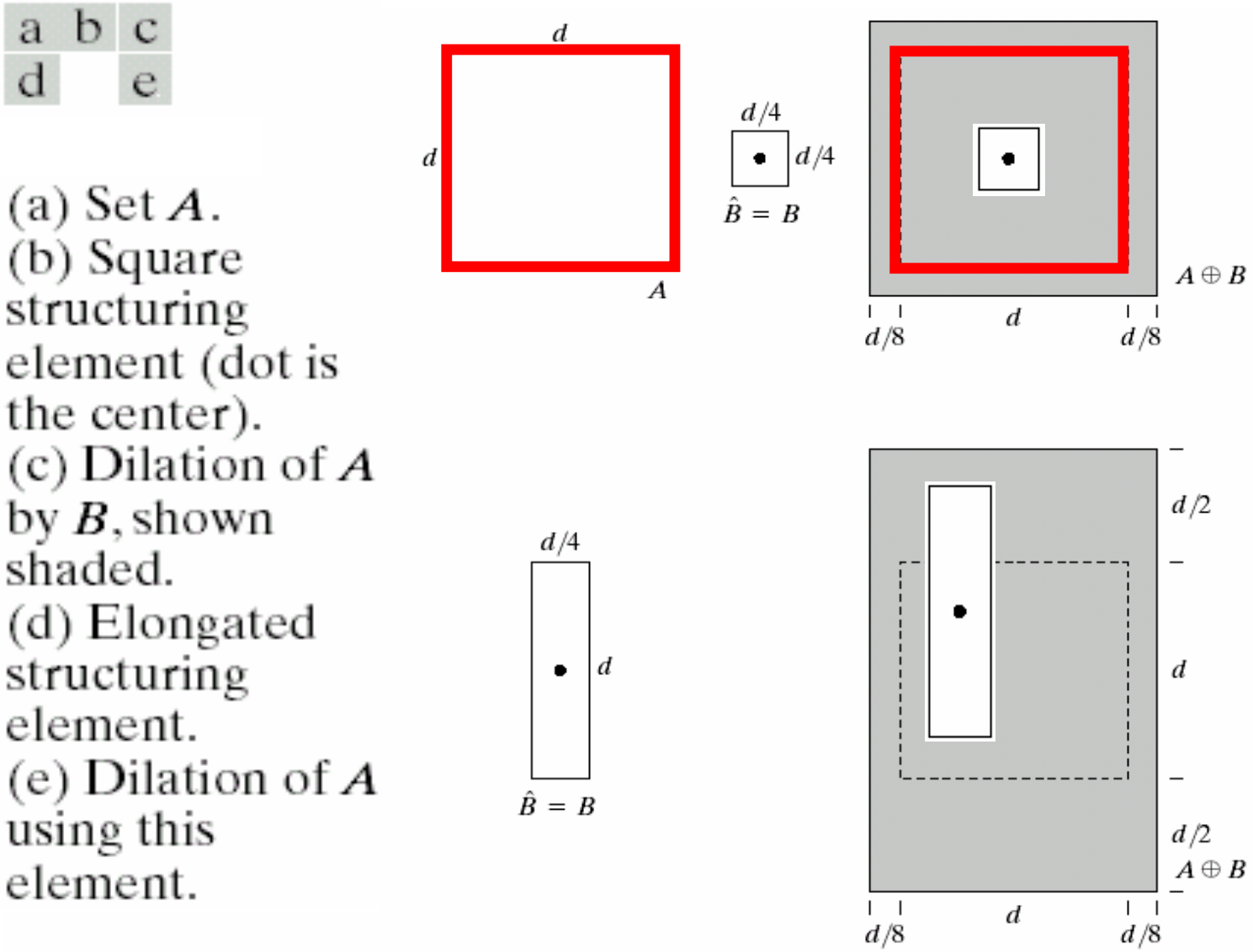
- Further Interpretation:

  Set B can be viewed as a convolution mask.

  The process of "flipping" B and then successively displace it so that it slides over set (image) A is analogous to the convolution.



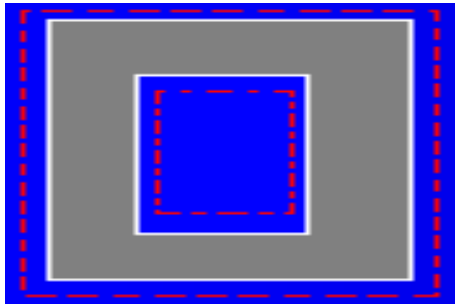
Morphology – Dilation



Morphology – Dilation

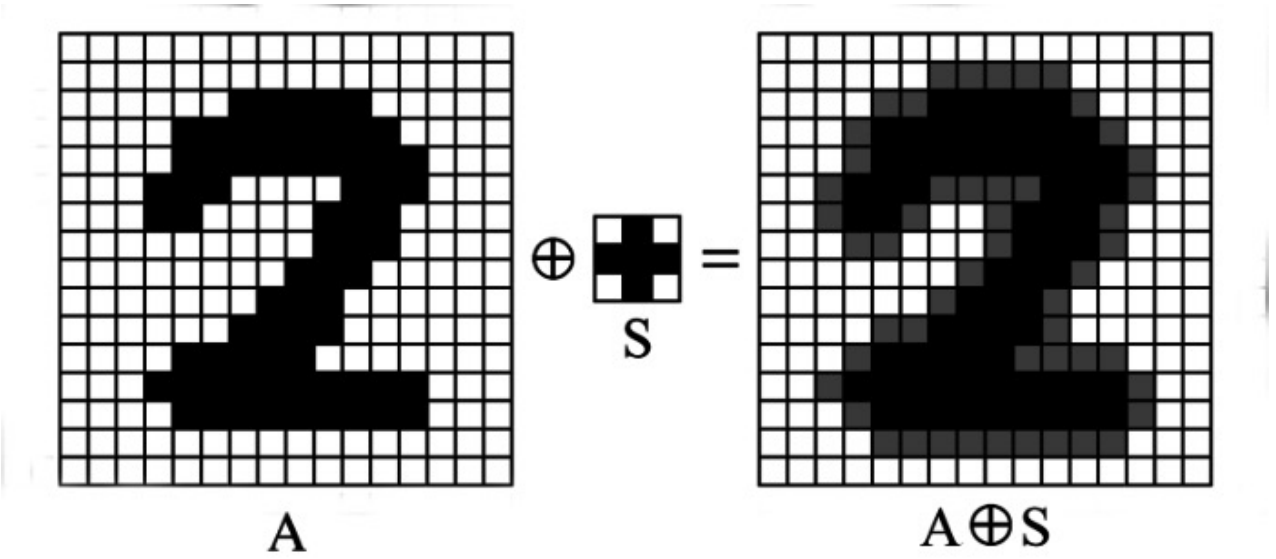
- The dilation morphological operation generates an output image g from an input image f using a structuring element h:

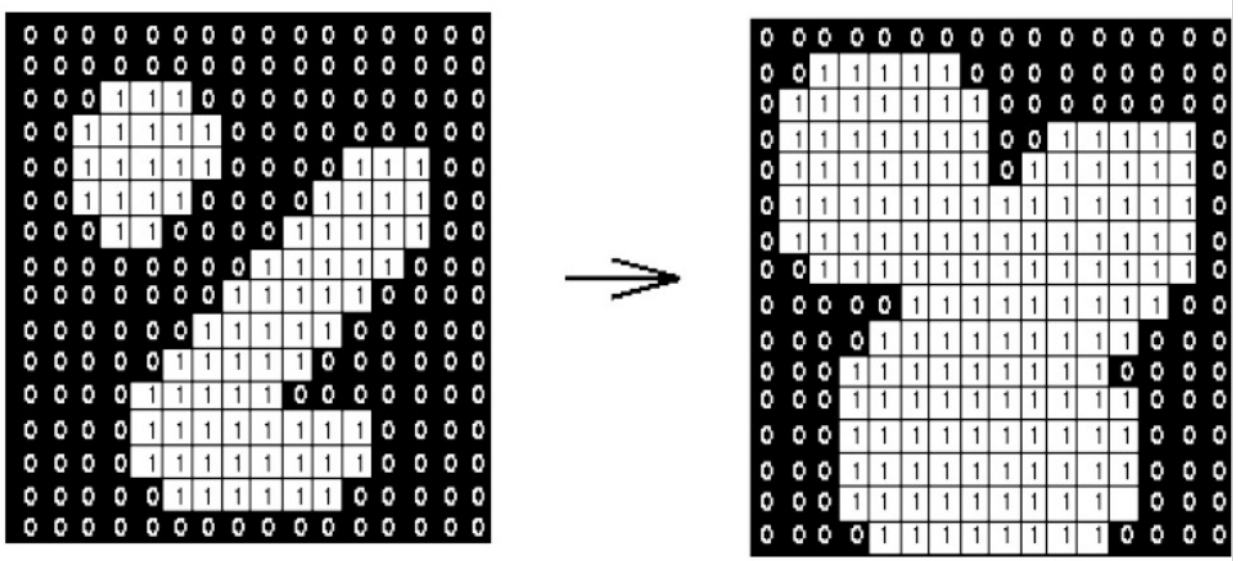
$$g(x, y) = \begin{cases} 1, & \text{if } h \text{ hints } f \\ 0, & \text{else} \end{cases}$$



- The effect of dilation with 3 x 3 mask is to add a single layer of pixels to the outer edge of an object and to decrease by a single layer of pixels to the holes in the object.
- A 5 x 5 mask will add two layers of pixels which is equivalent to applying a 3 x 3 mask twice.
- The main application of dilation is to remove small holes from the interior of an object.

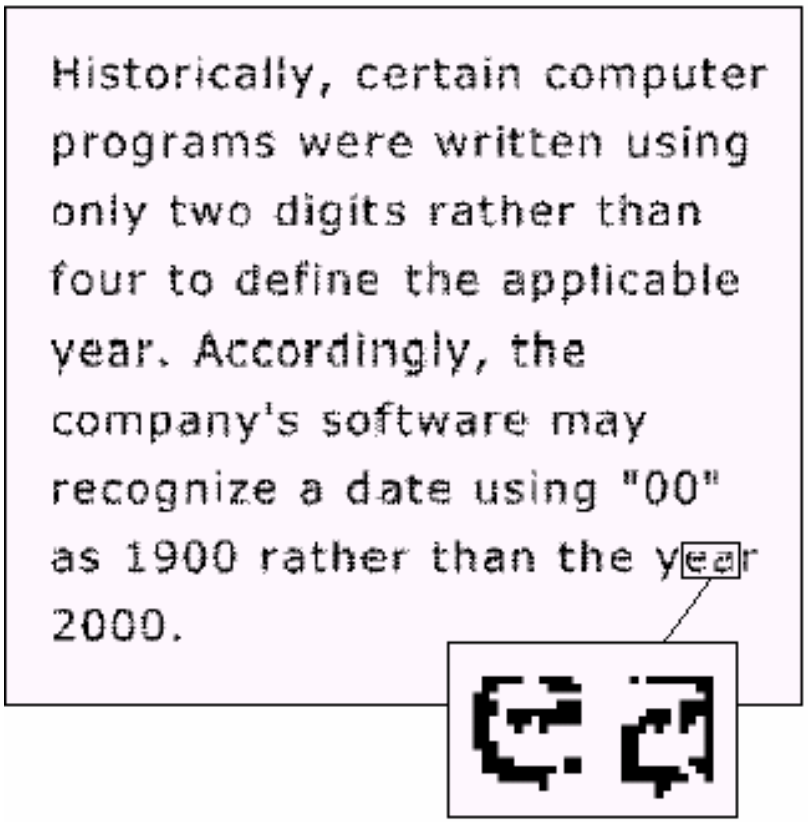
Dilation - Example





Effect of dilation using a 3×3 square structuring element

Dilation - Application



a c
b

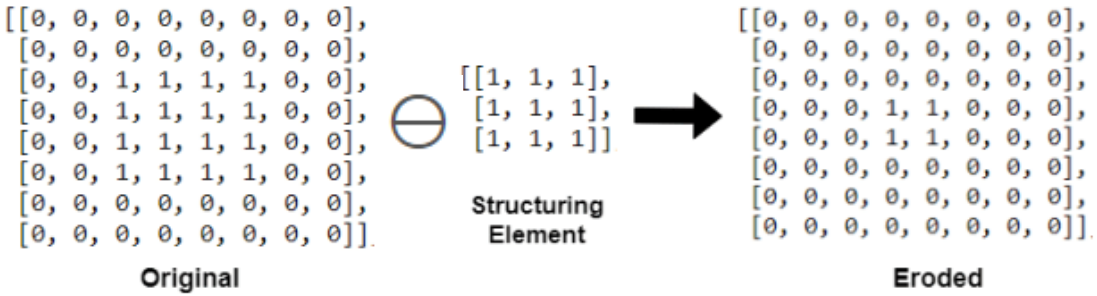
(a) Sample text of poor resolution with broken characters (magnified view).
(b) Structuring element.
(c) Dilation of (a) by (b). Broken segments were joined.

0	1	0
1	1	1
0	1	0

Morphology - Erosion

- Erosion of A by B, denoted $A \ominus B$, is defined as:

$$A \ominus B = \left\{ z \middle| (B)_z \subseteq A \right\}$$

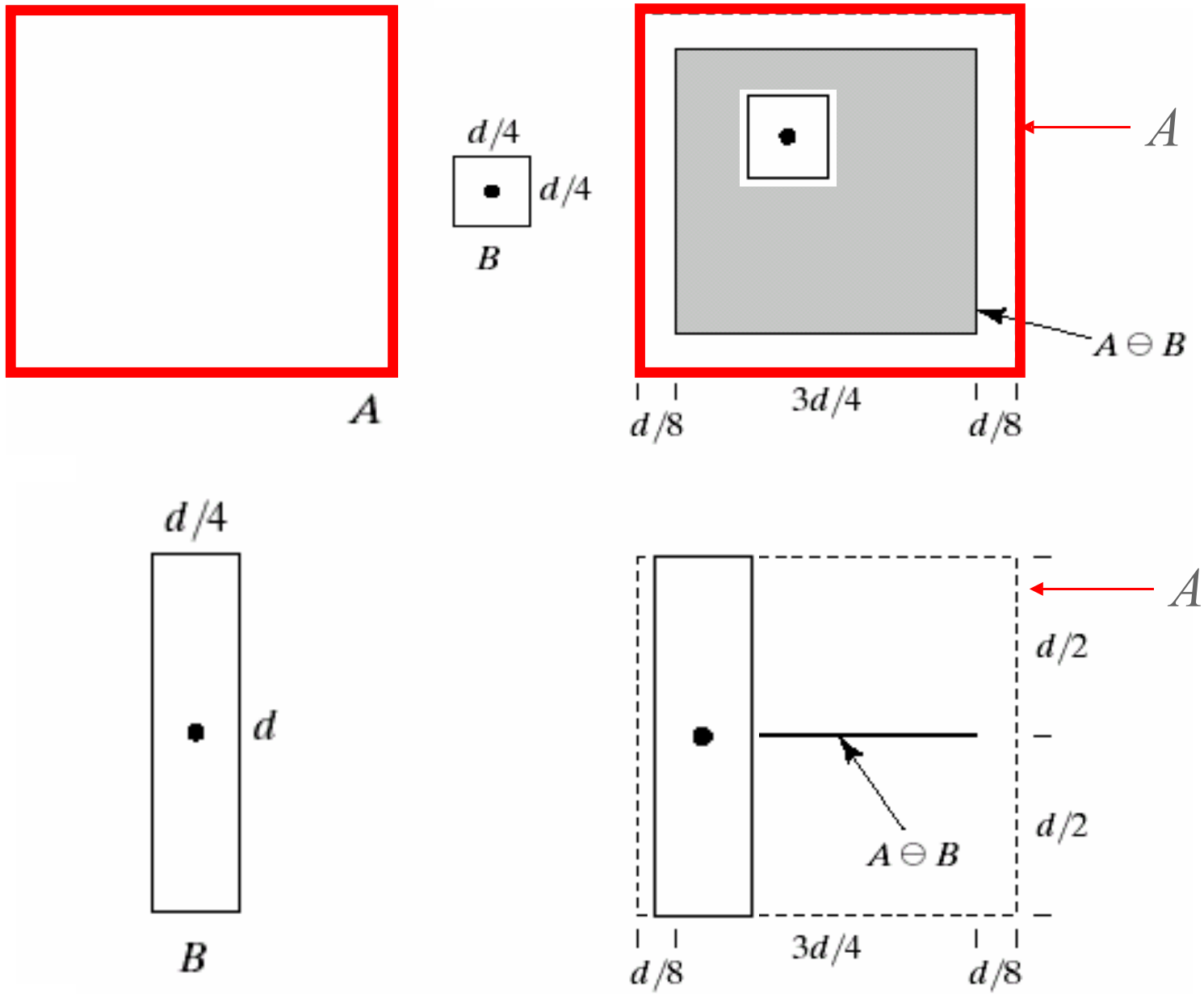


- Erosion of A by B is the set of all points z such that B, translated by z, is contained in A.

- Dilation and erosion are duals of each other with respect to set complementation and reflection. That is,

$$(A \ominus B)^c = A^c \oplus \hat{B}$$

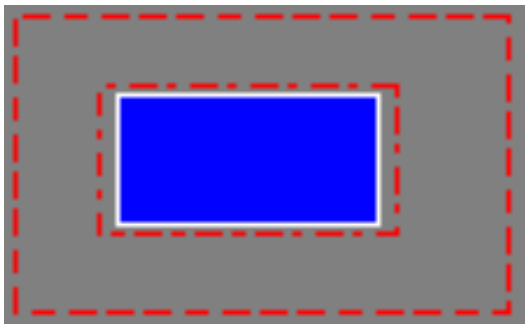
Erosion - Example



Erosion

- The erosion operation generates an output g from an input f using a structuring element h where :

$$g(x, y) = \begin{cases} 1, & \text{if } h \text{ completly falls in } f \\ 0, & \text{else} \end{cases}$$

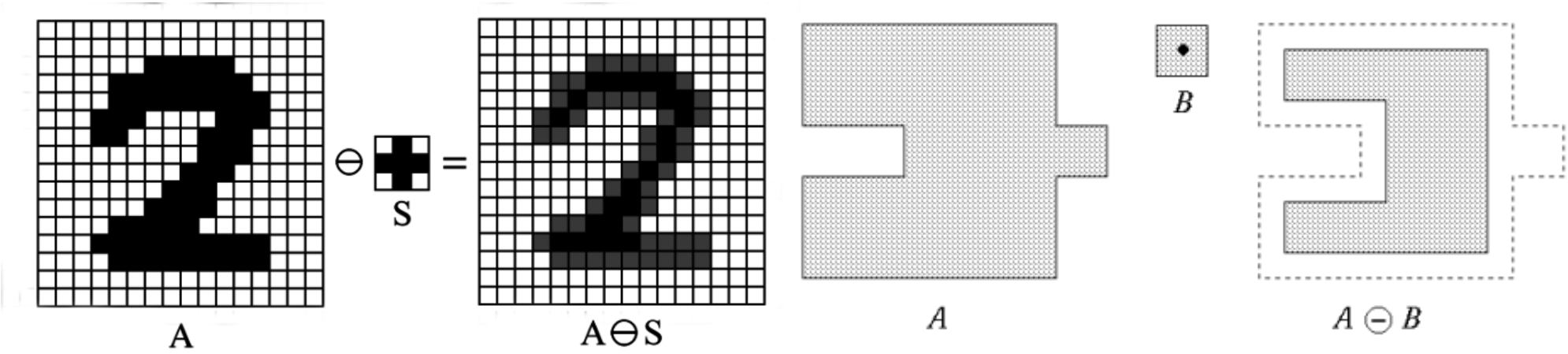


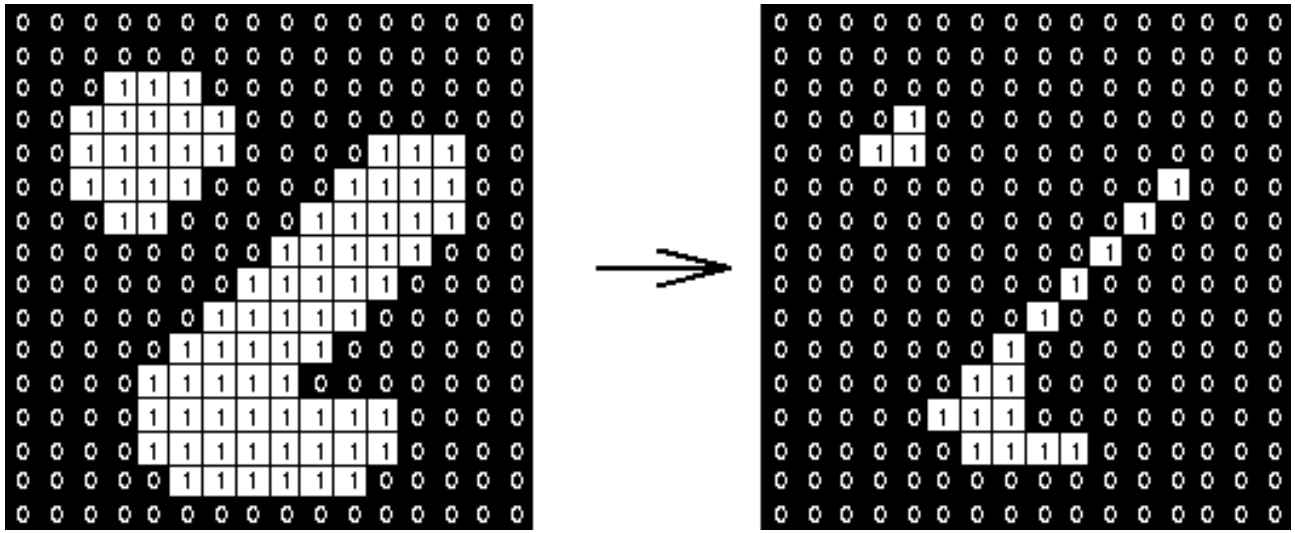
- The effect of an erosion with 3 x 3 mask is to strip a single layer of pixels from the outer edge of an object and to increase by a single layer of pixels to holes in the object.

- A 5 x 5 mask will strip off two layers of pixels which is equivalent to applying a 3 x 3 mask twice.

- The main application of erosion is to remove small noise artifacts from an image.

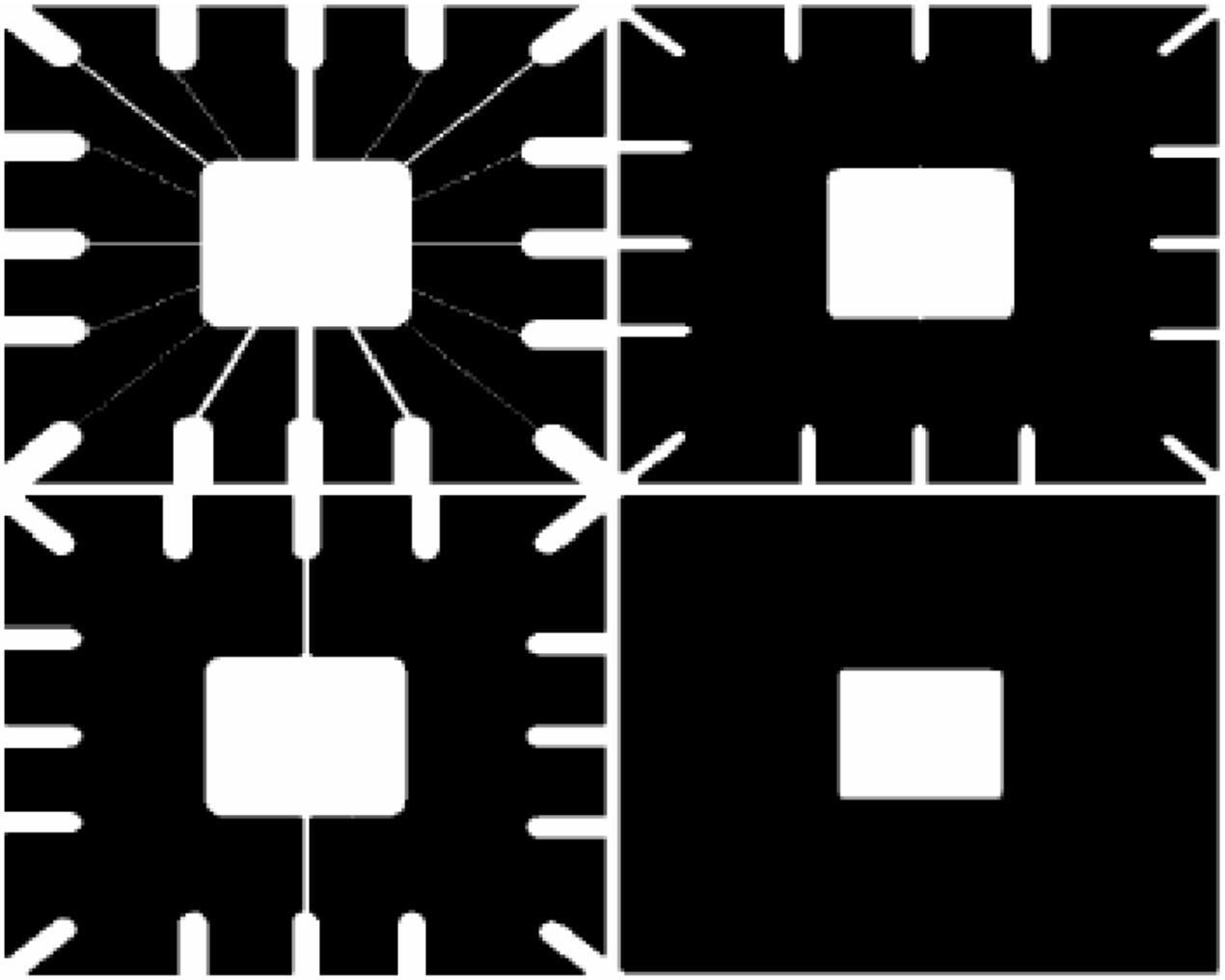
Erosion - Example





Effect of erosion using a 3×3 square structuring element

Morphology – Erosion



a b
c d

An illustration of erosion.
(a) Original image.
(b) Erosion with a disk of radius 10.
(c) Erosion with a disk of radius 5.
(d) Erosion with a disk of radius 20.

Erosion then Dilation

a b c

(a) Image of squares of size 1, 3, 5, 7, 9, and 15 pixels on the side. (b) Erosion of (a) with a square structuring element of 1's, 13 pixels on the side. (c) Dilation of (b) with the same structuring element.

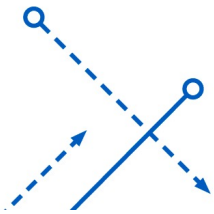
Morphology - Opening

- Compound operations – Opening
- A **compound** operation is when **two or more** morphological operations are performed in **succession**.
- A common example is opening which is an erosion followed by a dilation:

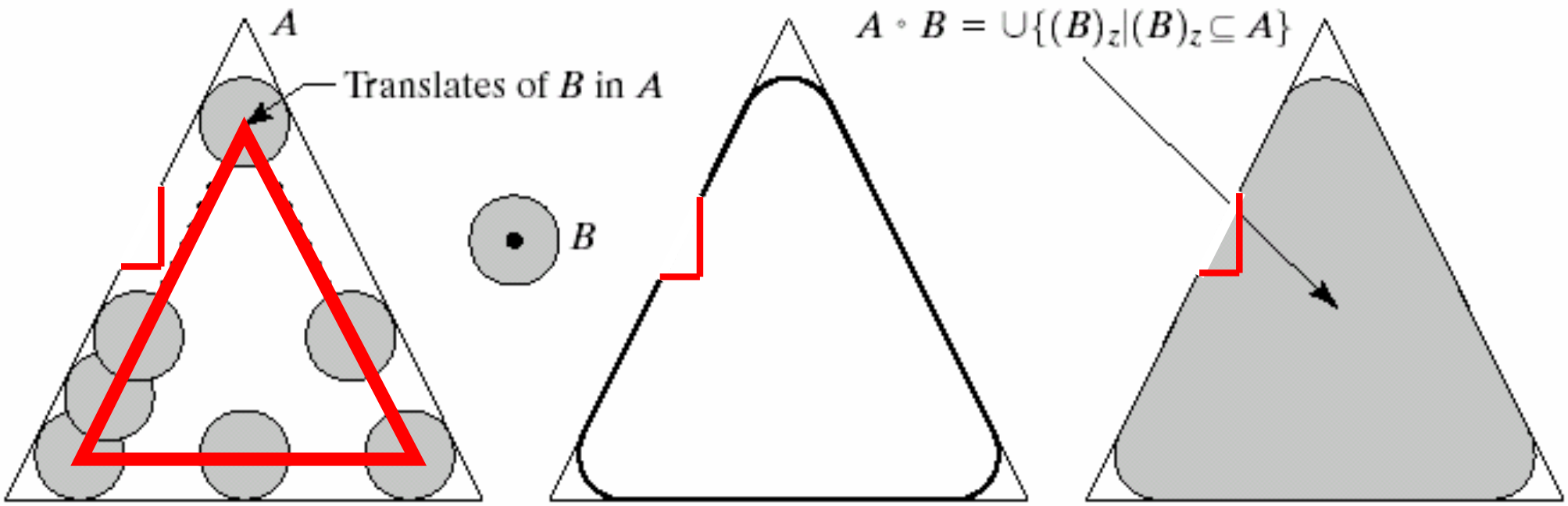
$$A \circ B = (A \ominus B) \oplus B$$

➢ The opening A by B is obtained by taking the union of all translates of B that fit into A. This can be expressed as a fitting processing such that:

$$A \circ B = \bigcup \{(B)_z \mid (B)_z \subseteq A\}$$



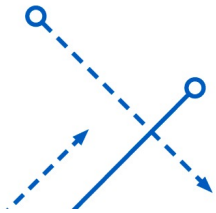
Morphology – Opening



a b c d

(a) Structuring element B "rolling" along the inner boundary of A (the dot indicates the origin of B). (c) The heavy line is the outer boundary of the opening. (d) Complete opening (shaded).

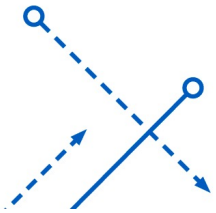
- Note that the outward pointing corners are rounded, where the inward pointing corners remain unchanged.



Morphology – Opening

$$A \circ B = (A \ominus B) \oplus B \qquad A \circ B = \bigcup \{(B)_z \mid (B)_z \subseteq A\}$$

- Opening is often performed to clear an image of noise whilst retaining the original object size.
- The opening operation tends to flatten the sharp peninsular projections on the object.
- Care must be taken that the operation does not distort the shape size of the object if this is significant.
- A useful way to see the effects is to look for differences between before and after opening by projecting these differences onto the original image.



Morphology – Closing

- Compound operations – Closing

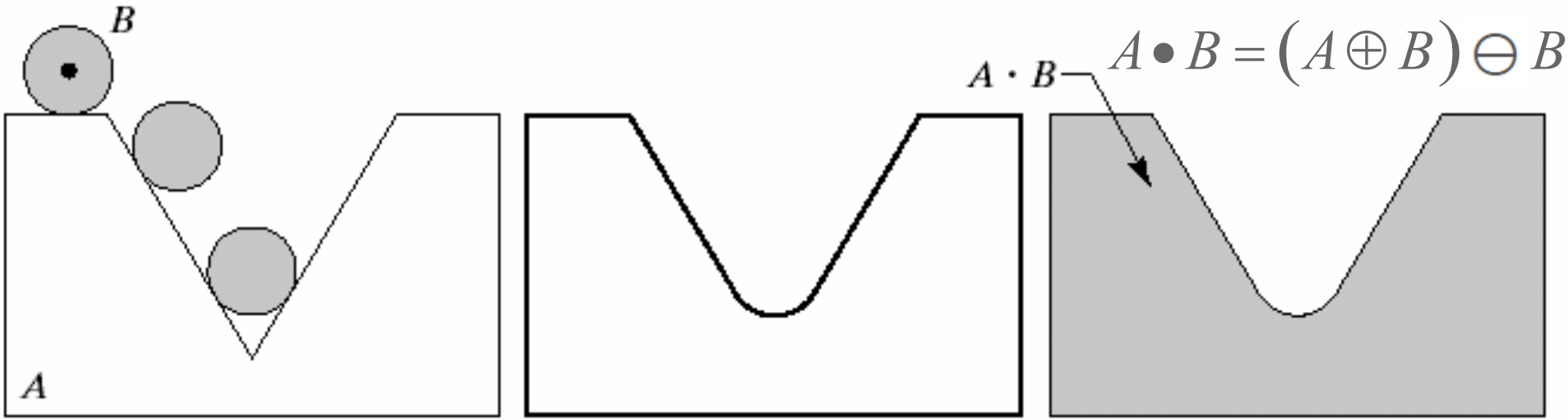
- Closing is the complementary operation of opening, defined as dilation followed by erosion.

$$A \bullet B = \left(A \oplus B\right) \ominus B$$

  - Opening and closing are duals of each other as:

$$\left(A \bullet B\right)^c = A^c \circ \hat{B}$$

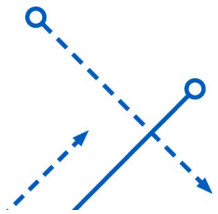
Morphology –Closing



a b c

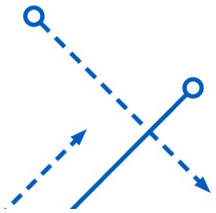
(a) Structuring element B "rolling" on the outer boundary of set A. (b) Heavy line is the outer boundary of the closing. (c) Complete closing (shaded).

- Note that the inward pointing corners are rounded, where the outward pointing corners remain unchanged.

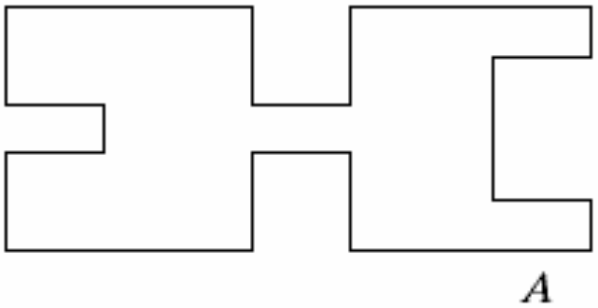


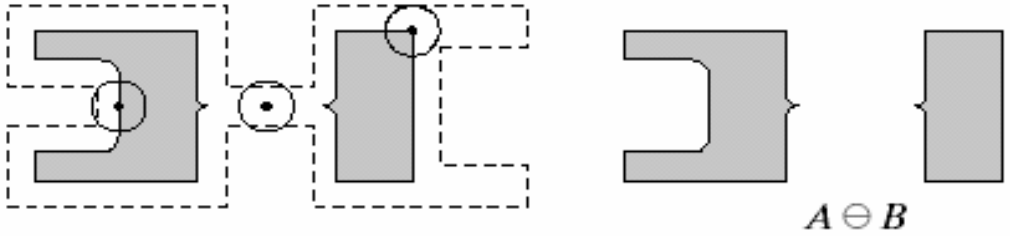
Morphology –Closing

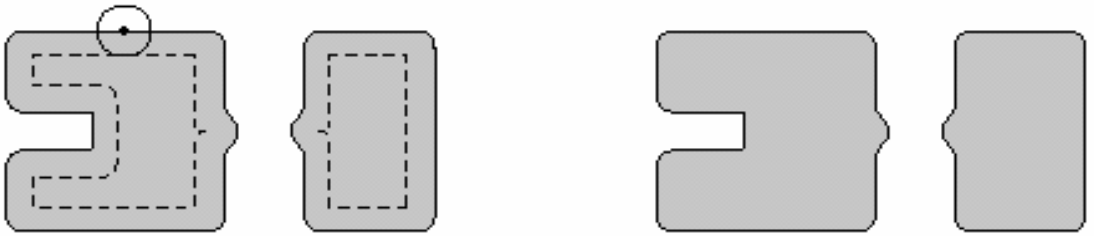
- The classic application of closing is to fill holes in a region whilst retaining the original object size.
- Dilation fills the holes and erosion restores the original region size.
- In addition to filling holes the closing operation tends to fill the ‘bays (凹)’ on the edge of a region.

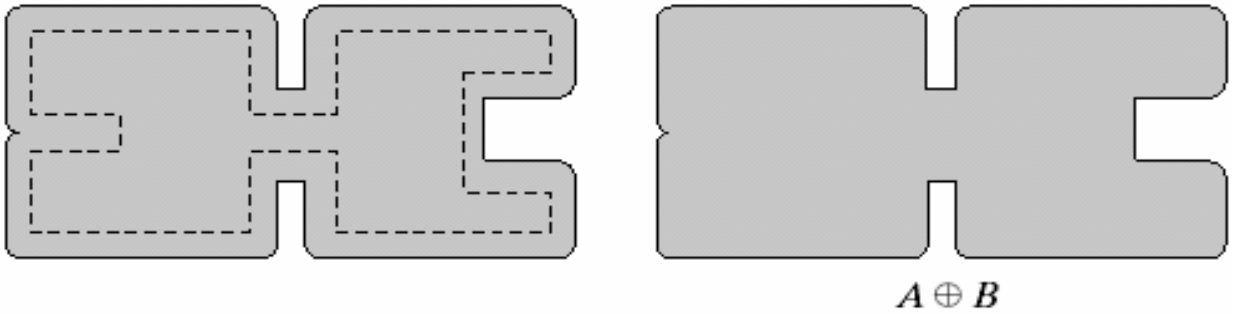


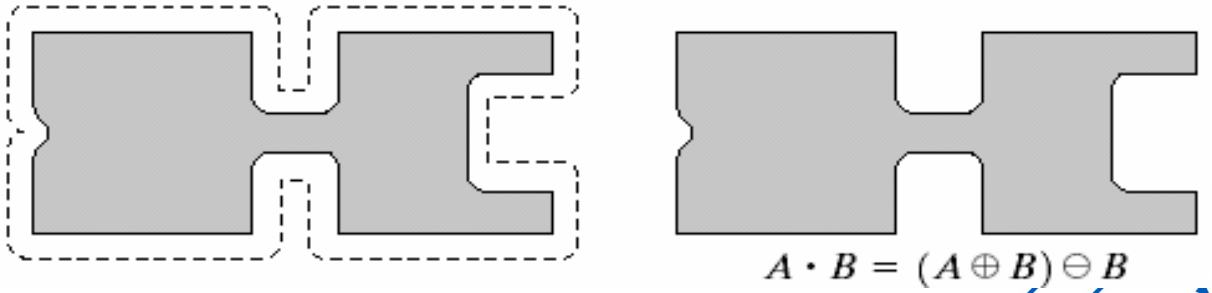
Erosion, Opening, Dilation, Closing





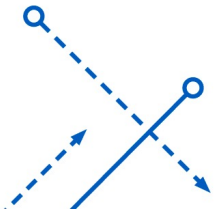






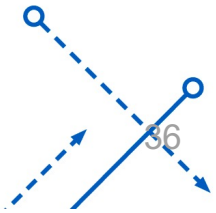
Morphology –Opening and Closing

- The opening operation satisfies the properties:
  - $A \circ B$ is a subset (subimage) of A
  - If C is a subset of D, then $C \circ B$ is a subset of $D \circ B$
  - $(A \circ B) \circ B = A \circ B$
- The closing operation satisfies the properties:
  - A is a subset (sub image) of $A \bullet B$
  - If C is a subset of D, then $C \bullet B$ is a subset of $D \bullet B$.
  - $(A \bullet B) \bullet B = A \bullet B$



Algorithms and Applications

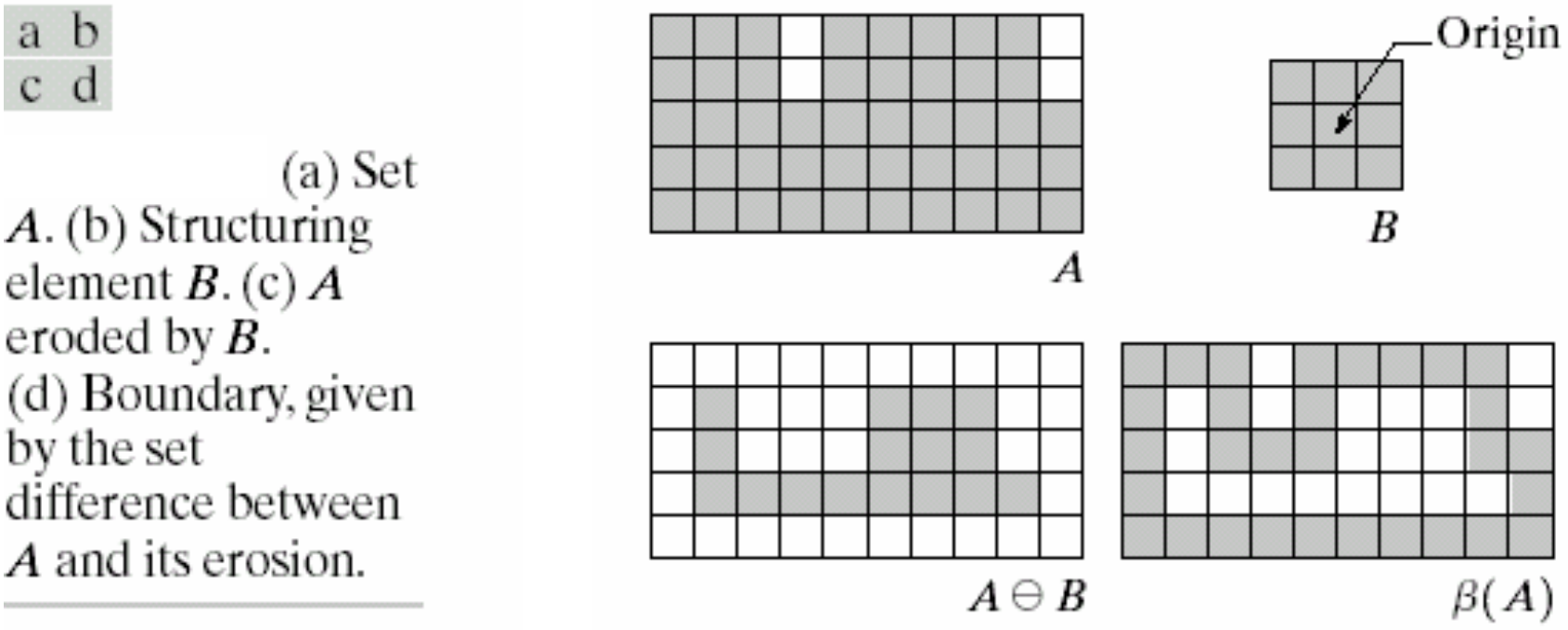
- Morphology can be used for many applications in image processing, pattern recognition, computer vision.
  - Boundary Extraction
  - Region Filling
  - Connected Components Extraction
  - Denoising



Boundary Extraction

- The boundary of a set A, denoted by $\beta(A)$:

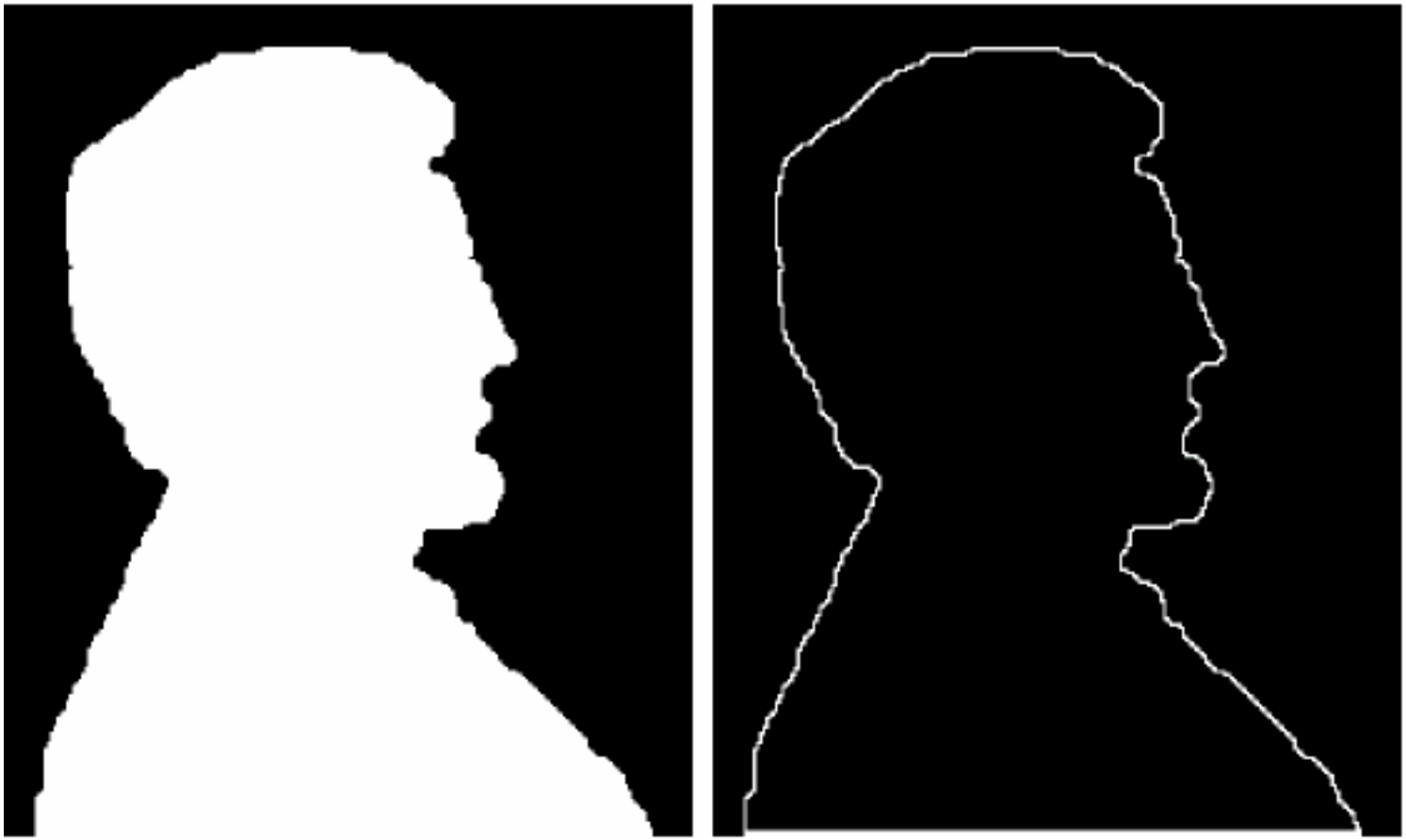
$$\beta(A) = A - (A \ominus B)$$



Boundary Extraction - Example

a b

(a) A simple binary image, with 1's represented in white. (b) Result of using Eq. (9.5-1) with the structuring element in Fig. 9.13(b).

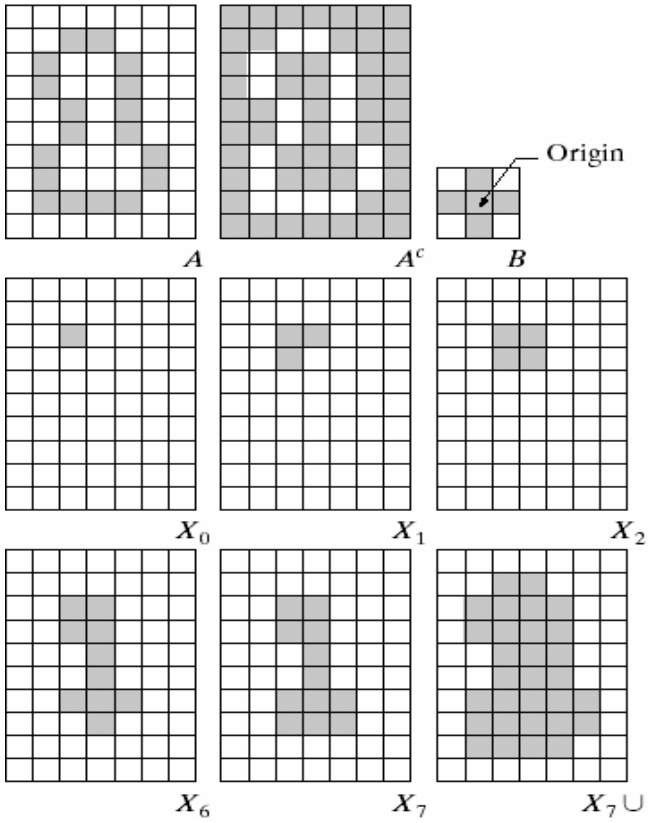


Region Filling

- Beginning with a point X_0 inside the boundary, the entire region inside the boundary is filled by the procedure:

$$A^F = X_k \cup A,$$

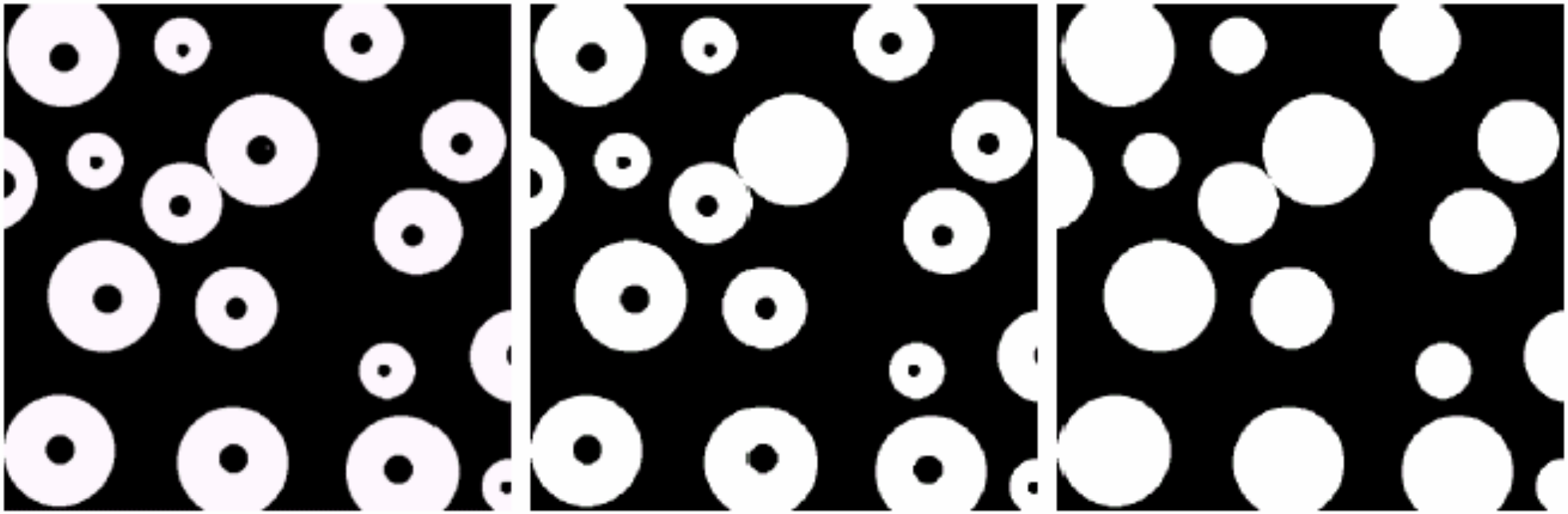
$$X_k = (X_{k-1} \oplus B) \cap A^c, \qquad k = 1, 2, 3 \ldots$$



a b c
d e f
g h i

Region filling. (a) Set A. (b) Complement of A. (c) Structuring element B. (d) Initial point inside the boundary. (e)–(h) Various steps of Eq. (9.5-2). (i) Final result [union of (a) and (h)].

Region Filling - Example

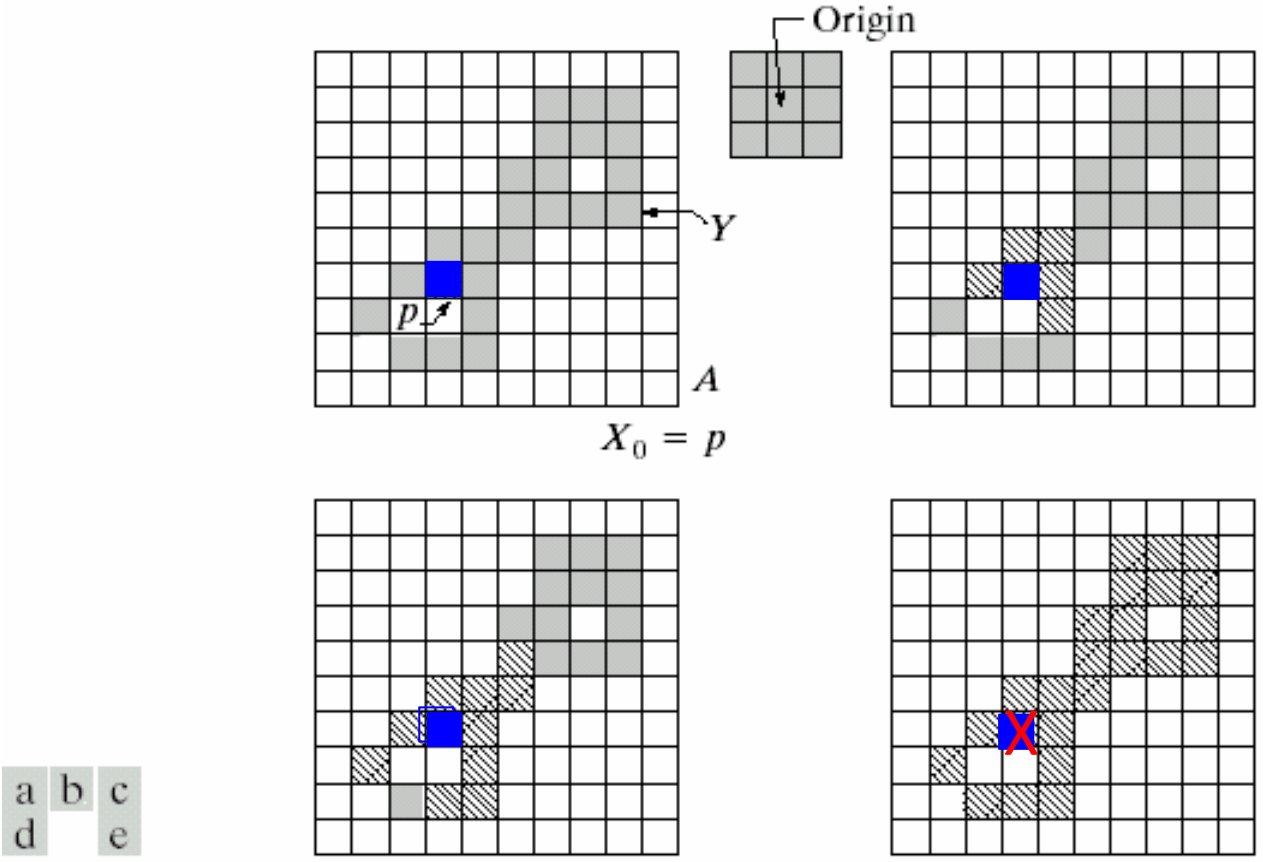


a b c

(a) Binary image (the white dot inside one of the regions is the starting point for the region-filling algorithm). (b) Result of filling that region (c) Result of filling all regions.

Extract connected components

- $X_k = (X_{k-1} \oplus B) \cap A, k = 1, 2, 3, \ldots$



(a) Set A showing initial point p (all shaded points are valued 1, but are shown different from p to indicate that they have not yet been found by the algorithm). (b) Structuring element. (c) Result of first iterative step. (d) Result of second step. (e) Final result.

Denoising

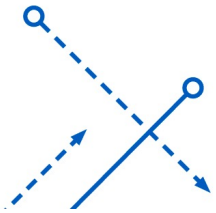
- Closing and Opening can be used to **eliminate noise**.

$$(A \circ B) \bullet B$$

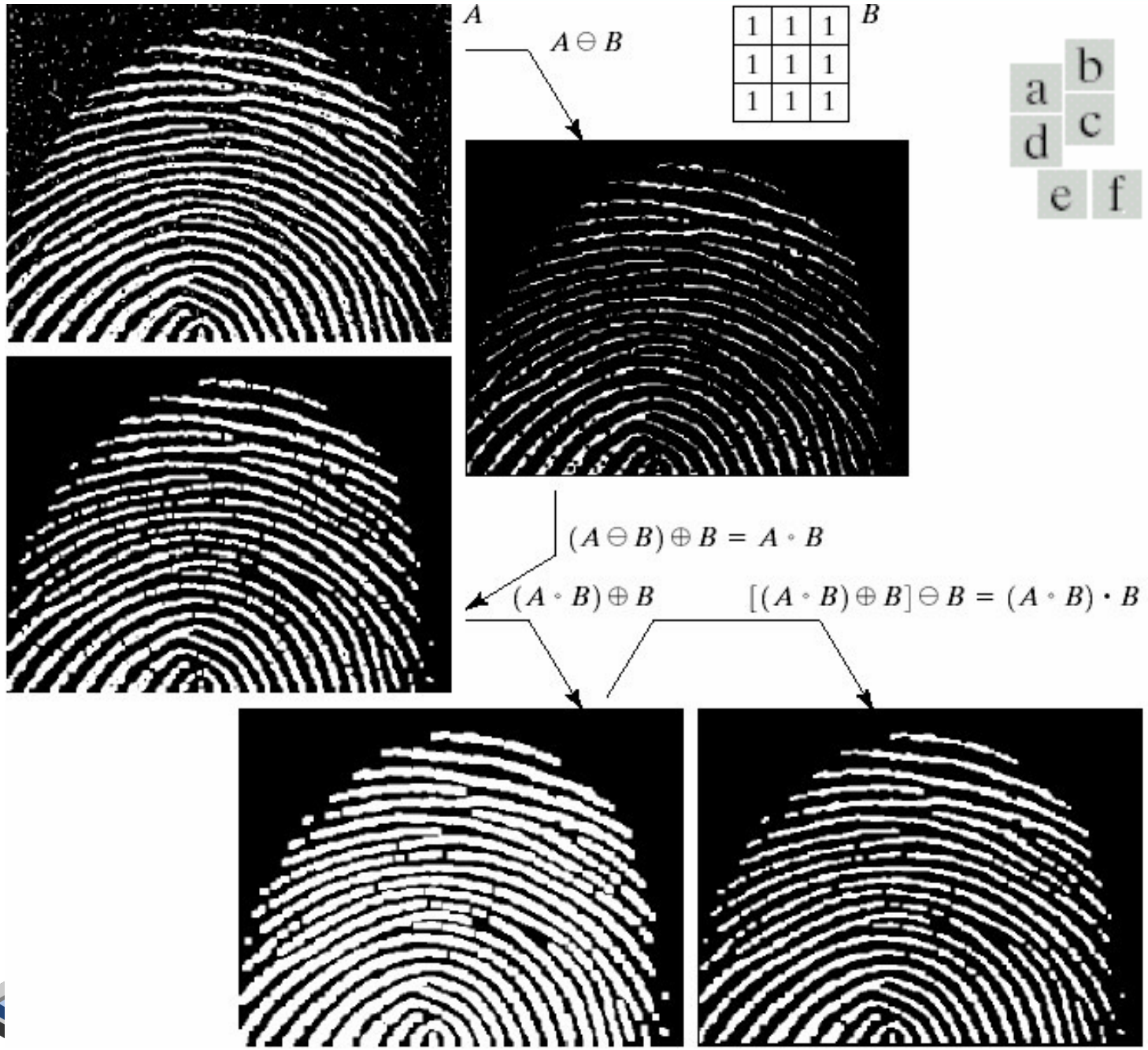
or

$$(A \bullet B) \circ B$$

- Noise **outside** the object are removed by **opening** with B
- Noise **inside** the object are removed by **closing** with B.



Algorithms and Applications



a b
d c
e f

(a) Noisy image.
(c) Eroded image.
(d) Opening of A.
(d) Dilation of the opening.
(e) Closing of the opening. (Original image for this example courtesy of the National Institute of Standards and Technology.)

Morphology –Summary

Operation	Equation	Comments (The Roman numerals refer to the structuring elements shown in Fig. 9.26).
Translation	$(A)_z = \{w \mid w = a + z, \quad \text{for } a \in A\}$	Translates the origin of A to point z.
Reflection	$\hat{B} = \{w \mid w = -b, \quad \text{for } b \in B\}$	Reflects all elements of B about the origin of this set.
Complement	$A^c = \{w \mid w \notin A\}$	Set of points not in A.
Difference	$A - B = \{w \mid w \in A, w \notin B\} = A \cap B^c$	Set of points that belong to A but not to B.
Dilation	$A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \varnothing\}$	"Expands" the boundary of A. (I)
Erosion	$A \ominus B = \{z \mid (B)_z \subseteq A\}$	"Contracts" the boundary of A. (I)

Morphology –Summary

Opening	$A \circ B = (A \ominus B) \oplus B$	Smoothes contours, breaks narrow isthmuses, and eliminates small islands and sharp peaks. (I)
Closing	$A \bullet B = (A \oplus B) \ominus B$	Smoothes contours, fuses narrow breaks and long thin gulfs, and eliminates small holes. (I)
Boundary extraction	$\beta(A) = A - (A \ominus B)$	Set of points on the boundary of set A. (I)
Region filling	$X_k = (X_{k-1} \oplus B) \cap A^c$; $X_0 = p$ and $k = 1, 2, 3, \ldots$	Fills a region in A, given a point p in the region. (II)
Connected components	$X_k = (X_{k-1} \oplus B) \cap A$; $X_0 = p$ and $k = 1, 2, 3, \ldots$	Finds a connected component Y in A, given a point p in Y. (I)

SAIR

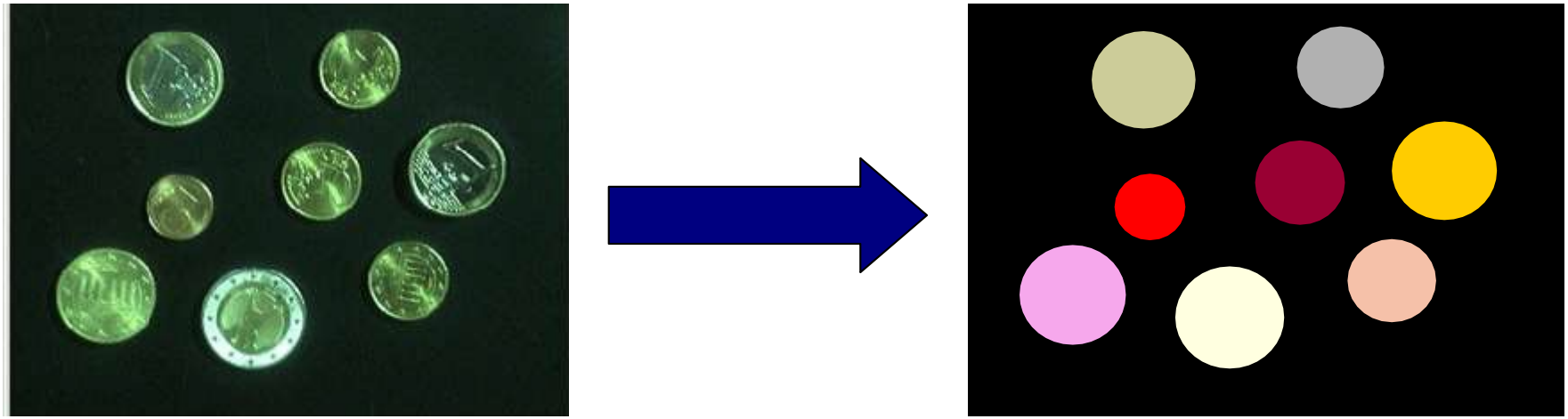
Spatial AI & Robotics Lab

COMPUTER VISION

Segmentation

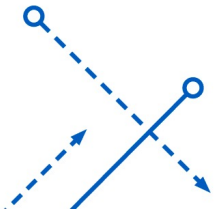
Image Segmentation

- **Partition** the pixels of an image into **groups** that strongly correlate with the objects in an image
- Typically, the first step in any automated computer vision application.



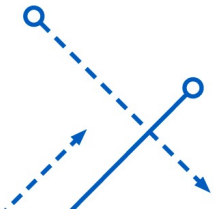
Why & What?

- Useful mid-level representation of an image
  - Can facilitate better downstream tasks
- Segmentation should be homogeneous with some characteristic (gray level, texture, color, motion)
- The desired type segmentation depends on the task
  - Broad theory is absent at present
- Variety of approaches, algorithms, and applications
  - Finding people, summarizing video, annotation figures, background subtraction, finding buildings/rivers in satellite images.



Thought Exercise….

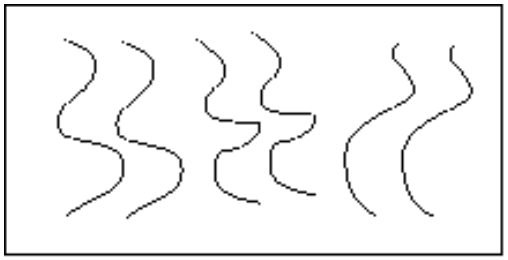
- Based on what you already know about CVIP
  - How would you do segmentation?
- Did you come up with
  - Hough Transform?
    - Let's assume types of objects are unknown.
  - Edge Detection?
  - Texture Grouping?
- What if the elements are characters?
  - Thresholding?



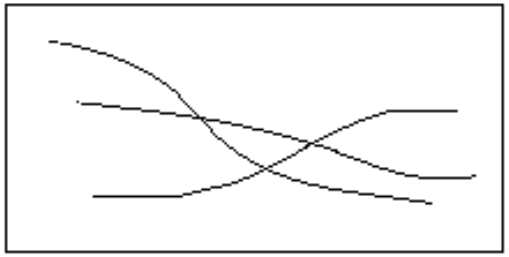
Segmentation

Segmentation algorithms generally are based on one of two basis properties of intensity values.

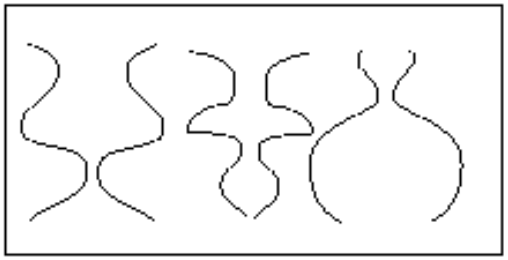
- **Discontinuity:** to partition an image based on abrupt changes in intensity (such as edges).
- **Similarity:** to partition an image into regions that are similar according to a set of predefined criteria.



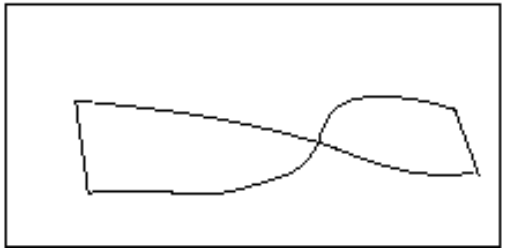
Parallelism



Continuity



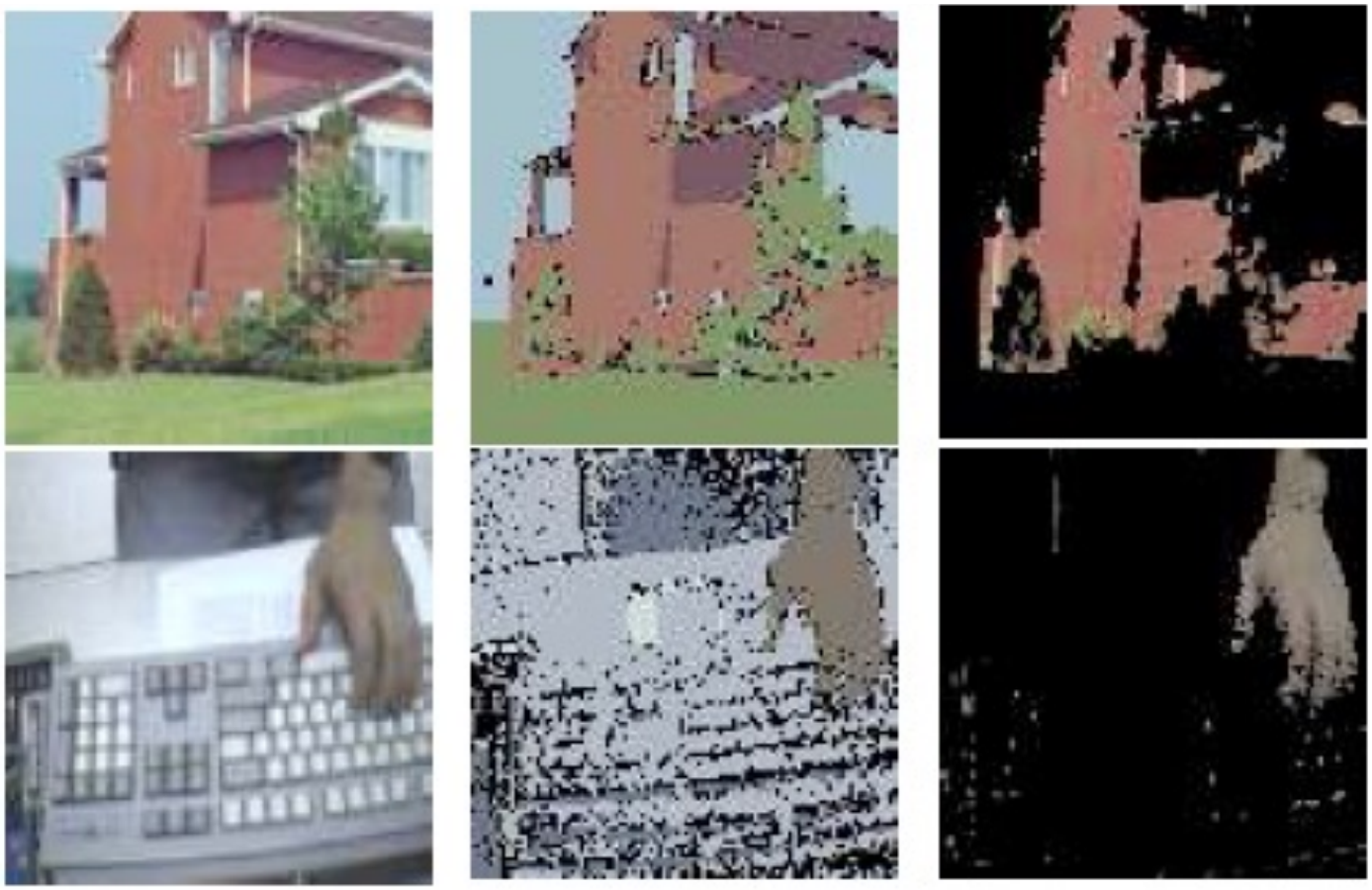
Symmetry



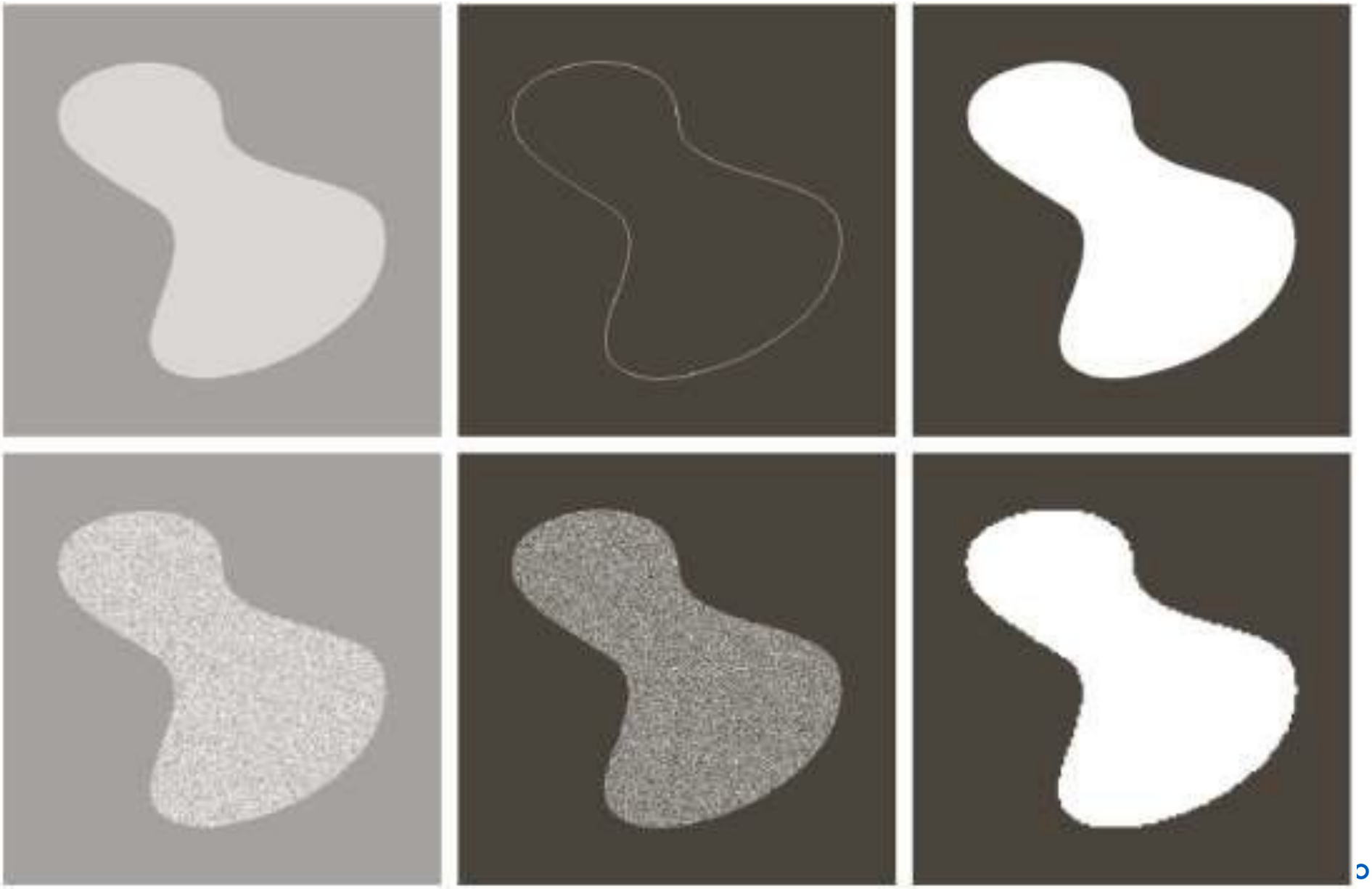
Closure

Segmentation needs Grouping/Clustering

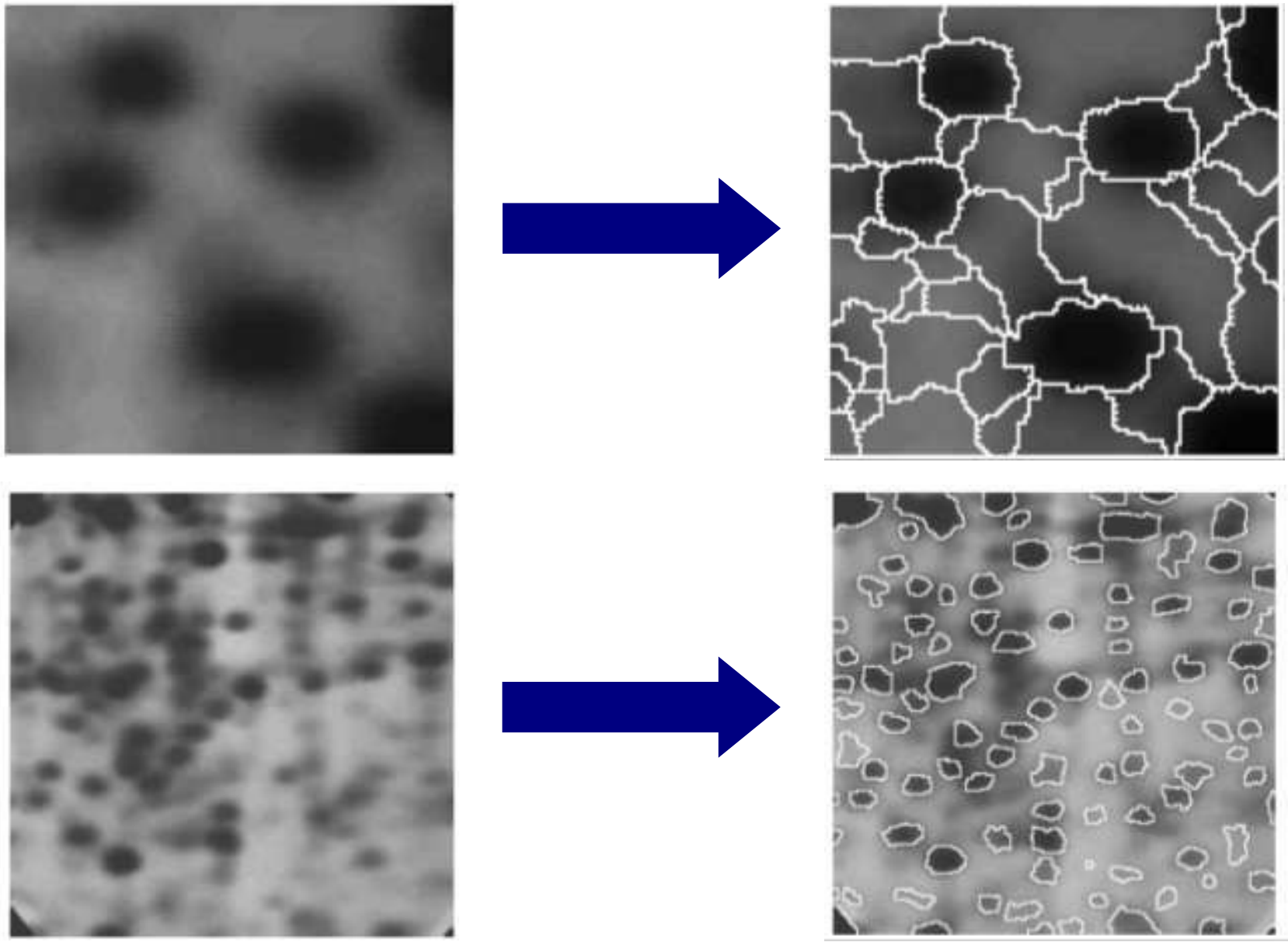
- Grouping (or clustering)
  - Collect pixels that “belong together”



Boundaries or Regions are equally good

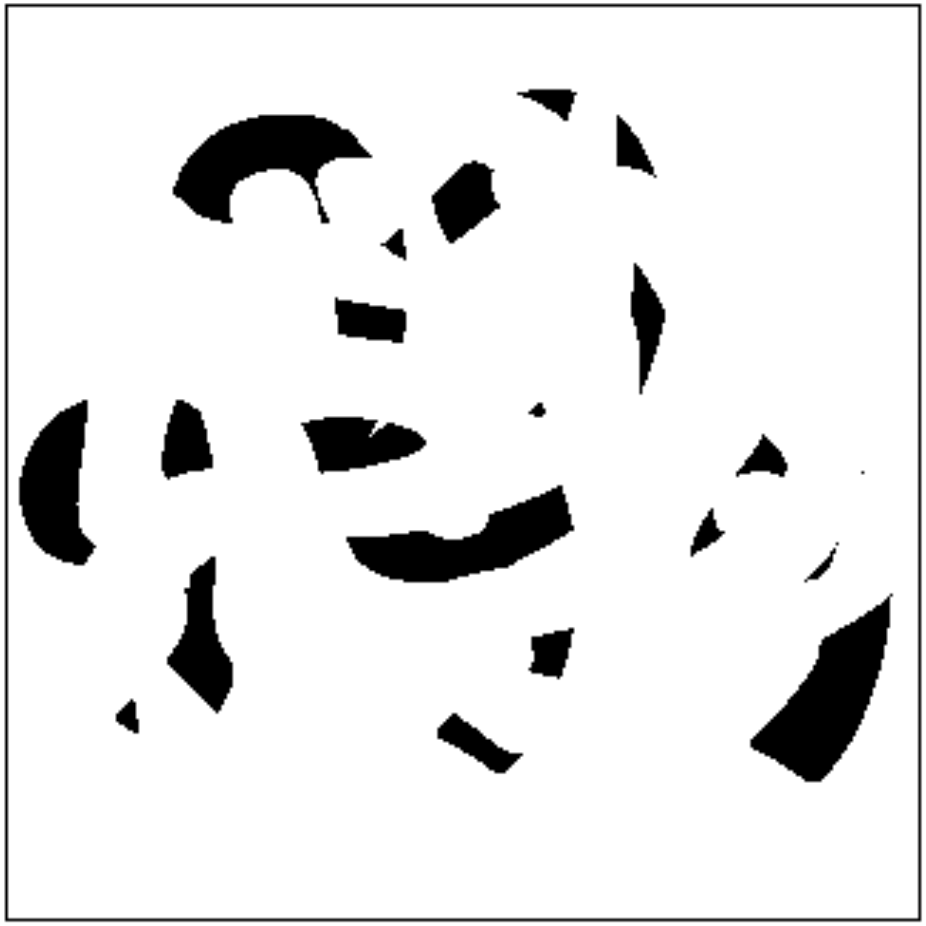


Boundaries or Regions are equally good



Occlusion cues are important

- What do you see?



Occlusion cues are important

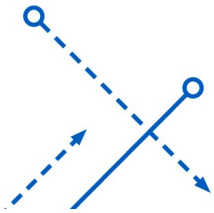
- Aren't they?

Regions and Edges

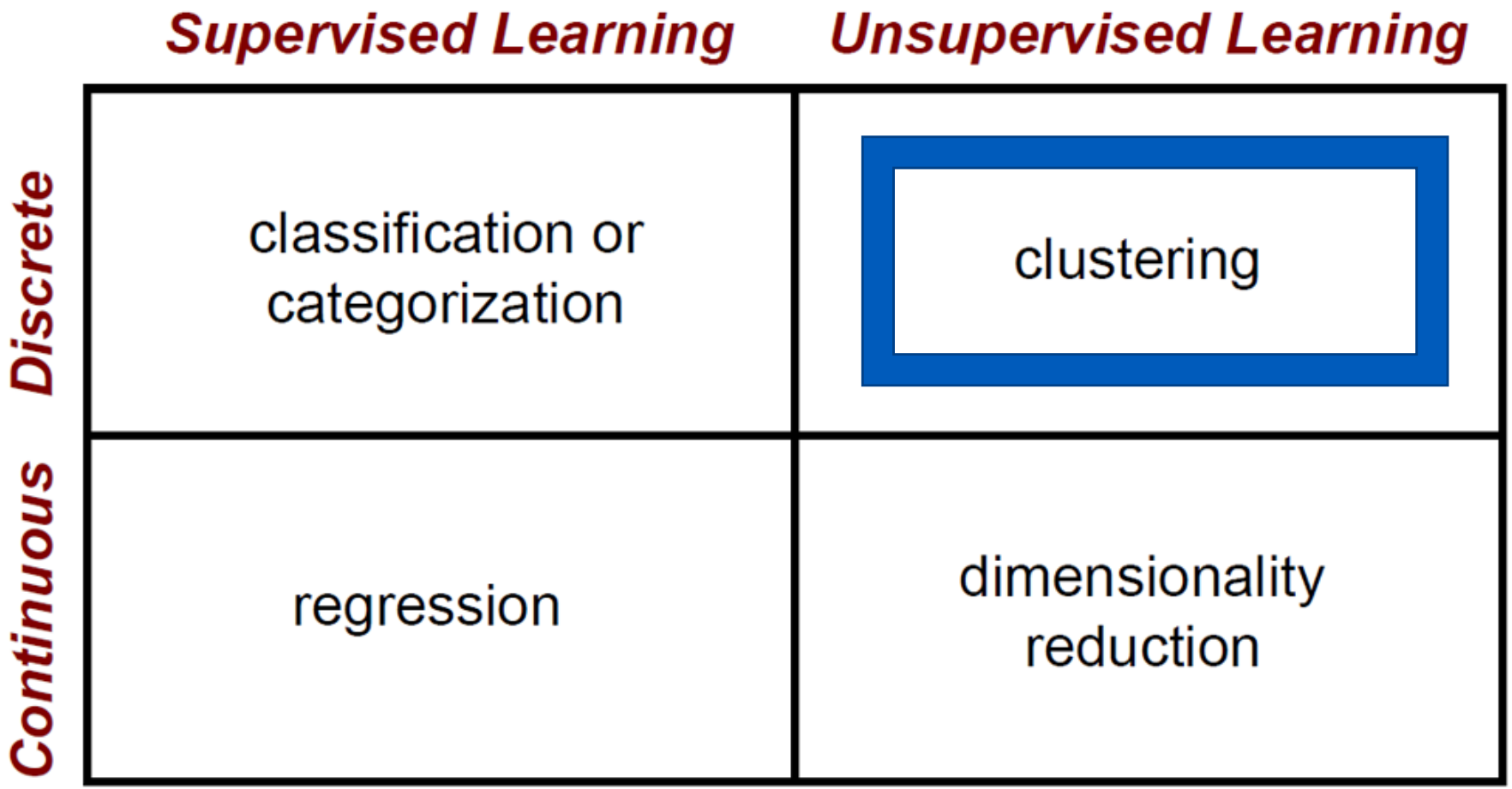
- Regions are bounded by closed contours
  - We could “fill” closed contours to obtain regions
  - We could “trace” regions to obtain edges
- Unfortunately, these procedures rarely produce satisfactory results.

Regions and Edges

- **Edges** are found based on **DIFFERENCES** between values of adjacent pixels.
- **Regions** are found based on **SIMILARITIES** between values of adjacent pixels.
- Goal associate some higher level – more meaningful units with the regions of the image.



Machine Learning



Clustering example: image segmentation

- Break up the image into **meaningful** or **perceptually similar** regions

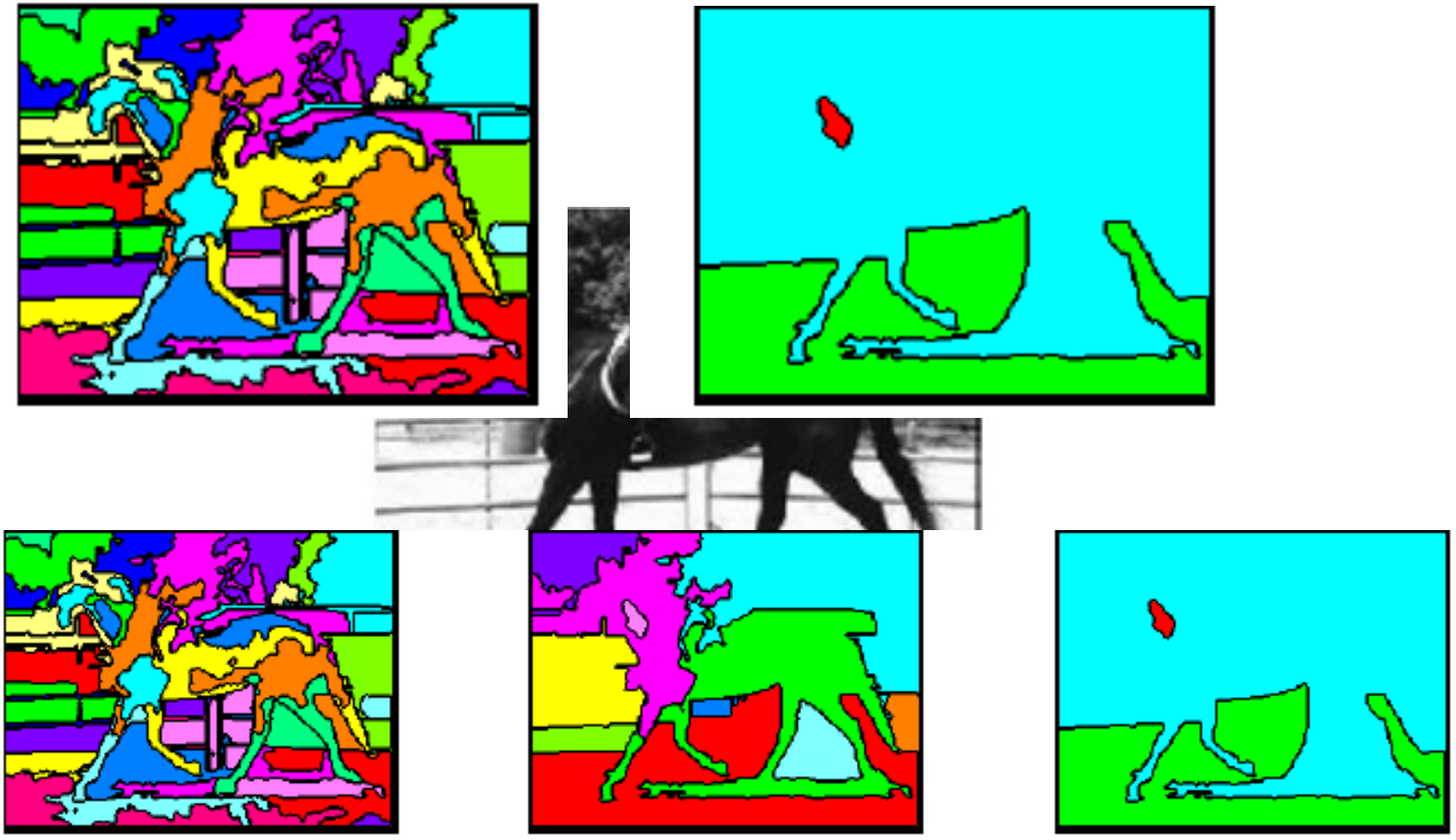
Clustering example: image segmentation

- Foreground/Background

Types of segmentations

Oversegmentation

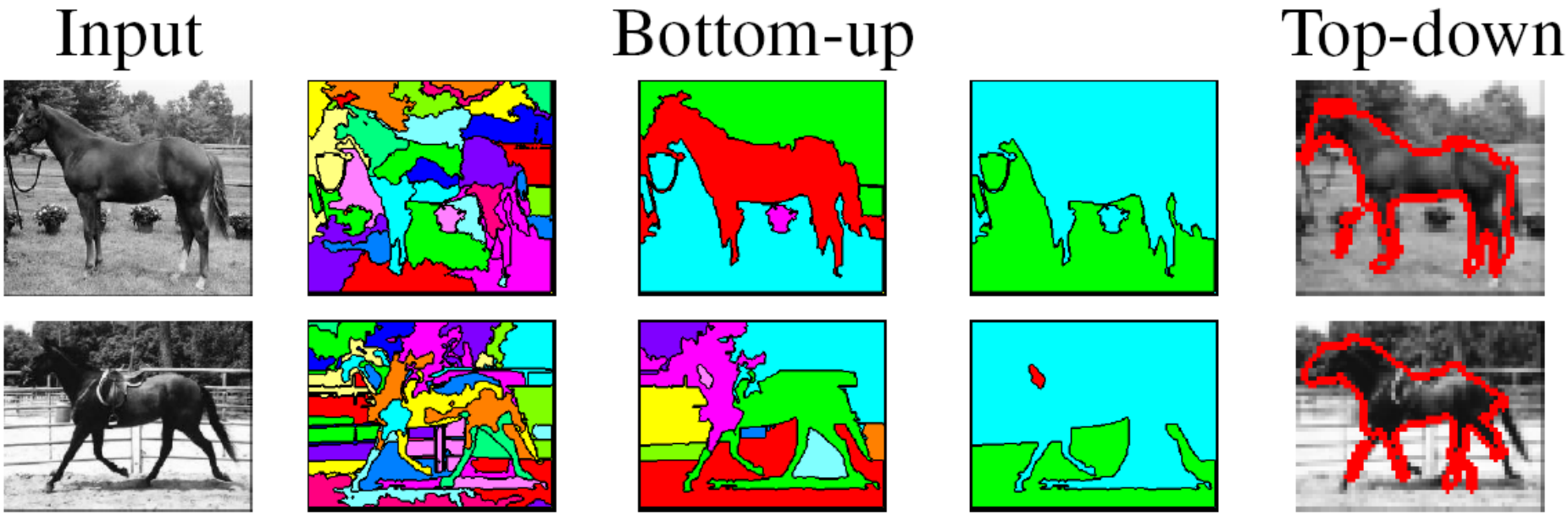
Undersegmentation



Multiple Segmentations

Major processes for segmentation

- Bottom-up
  - Group pixels with similar features
- Top-down
  - Group pixels that likely belong to the same object

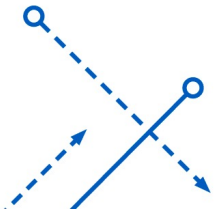


[Levin and Weiss 2006]

Clustering

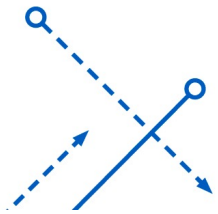
- Group together similar points and represent them with a single category.

- Key Challenges:
  - What makes two points/images/patches similar?
  - How do we compute an overall grouping from pairwise similarities?



How do we cluster?

- K-means
  - Iteratively re-assign points to nearest cluster center
- Single-link clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of PDF
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights



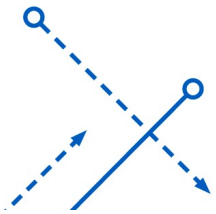
Clustering for Summarization

- Represent a cluster by its "center" (in feature space) and the list of data points it contains.
- Goal:
  - Minimize variance in data given clusters
  - Preserve information

Cluster Center Data

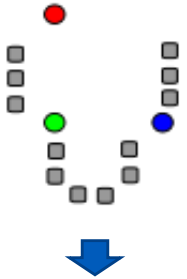
$$\mathbf{c}^*, \boldsymbol{\delta}^* = \underset{\mathbf{c},\boldsymbol{\delta}}{\operatorname{argmin}} \frac{1}{N} \sum_{j}^{N} \sum_{i}^{K} \delta_{ij} \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

Whether x_j is assigned to c_i

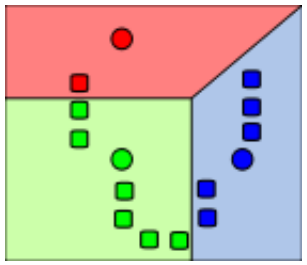


K-means algorithm

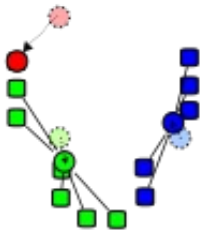
1. Randomly select K centers (means)



2. Assign each point to nearest center (mean)

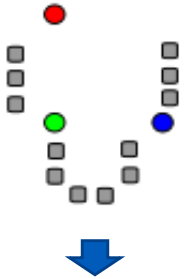


3. Compute new center (mean) for each cluster

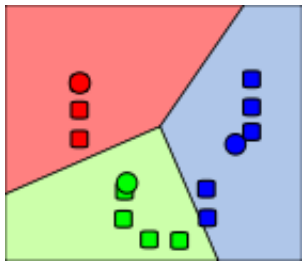


K-means algorithm

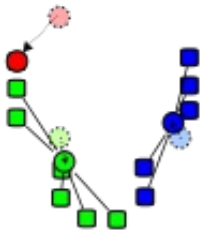
1. Randomly select K centers



2. Assign each point to nearest center



3. Compute new center (mean) for each cluster



Back to 2

K-means algorithm

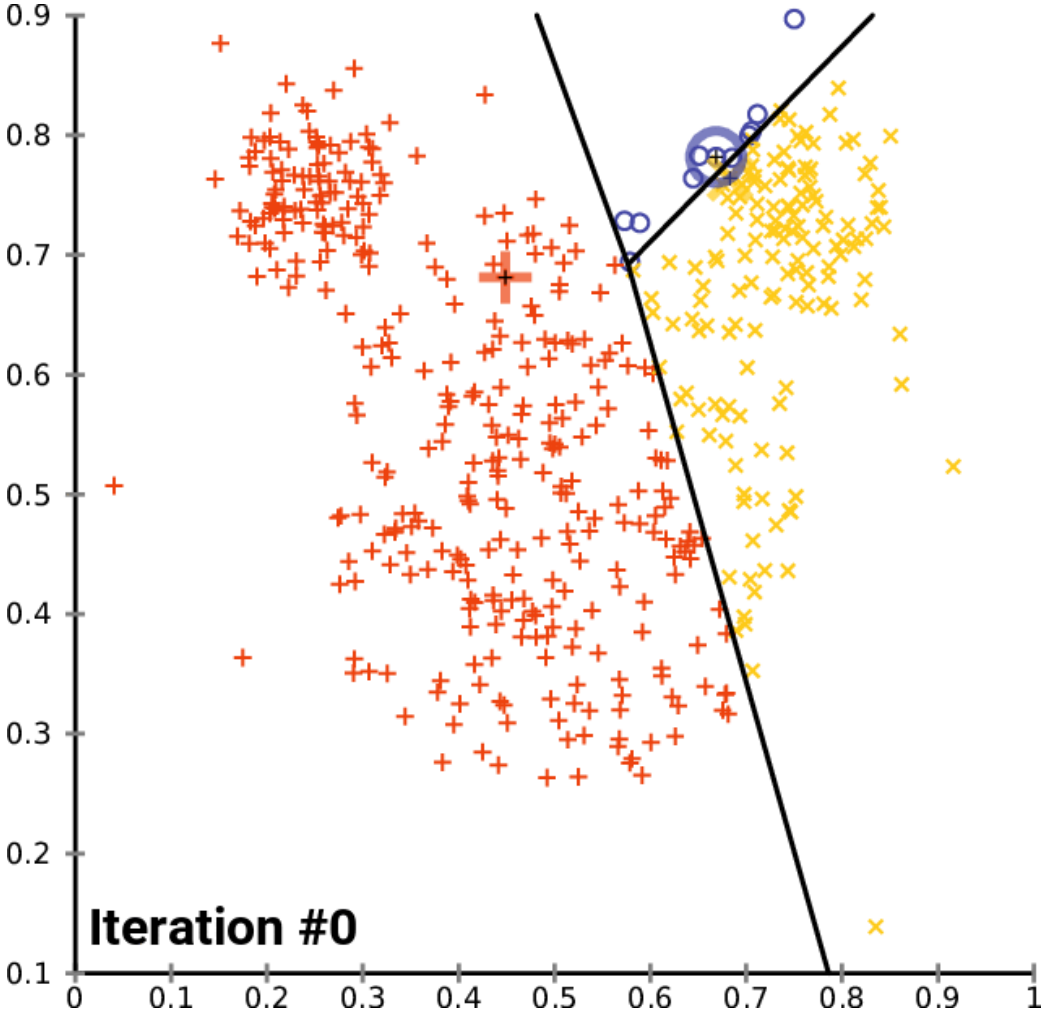


Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means algorithm

1. Initialize cluster centers: $\mathbf{c}^0$; t=0

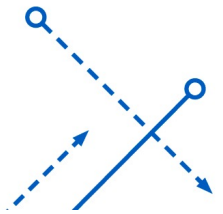
2. Assign each point to the closest center

$$\boldsymbol{\delta}^t = \underset{\boldsymbol{\delta}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij} \left(\mathbf{c}_i^{t-1} - \mathbf{x}_j \right)^2$$

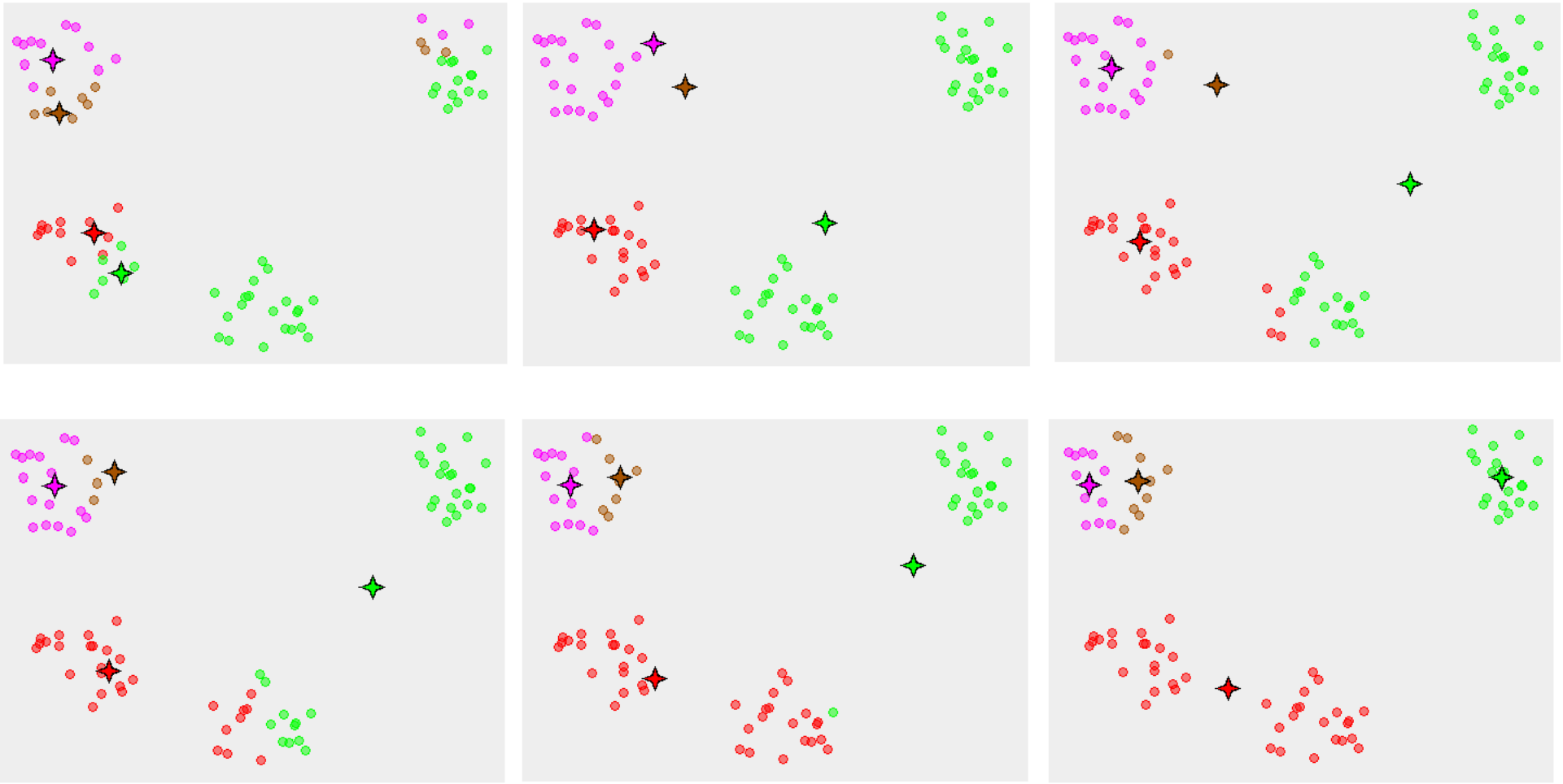
3. Update cluster centers as the mean of the points

$$\mathbf{c}^t = \underset{\mathbf{c}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij}^t \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

4. Repeat 2-3 until no points are re-assigned (t=t+1)

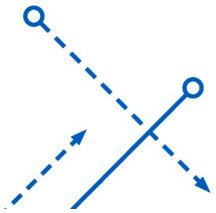


K-means may converge to local minimum



K-means: design choices

- Initialization
  - Randomly select K points as initial cluster center
  - Or greedily choose K points to minimize residual
- Distance measures
  - Traditionally Euclidean, could be others
- Optimization
  - May converge to a *local minimum*
  - May want to perform multiple restarts

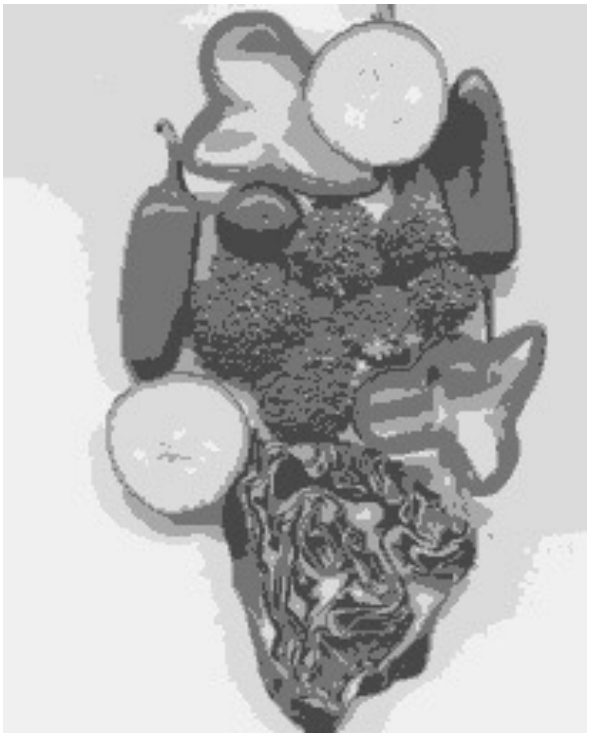


K-means clustering w/ intensity or color

Image

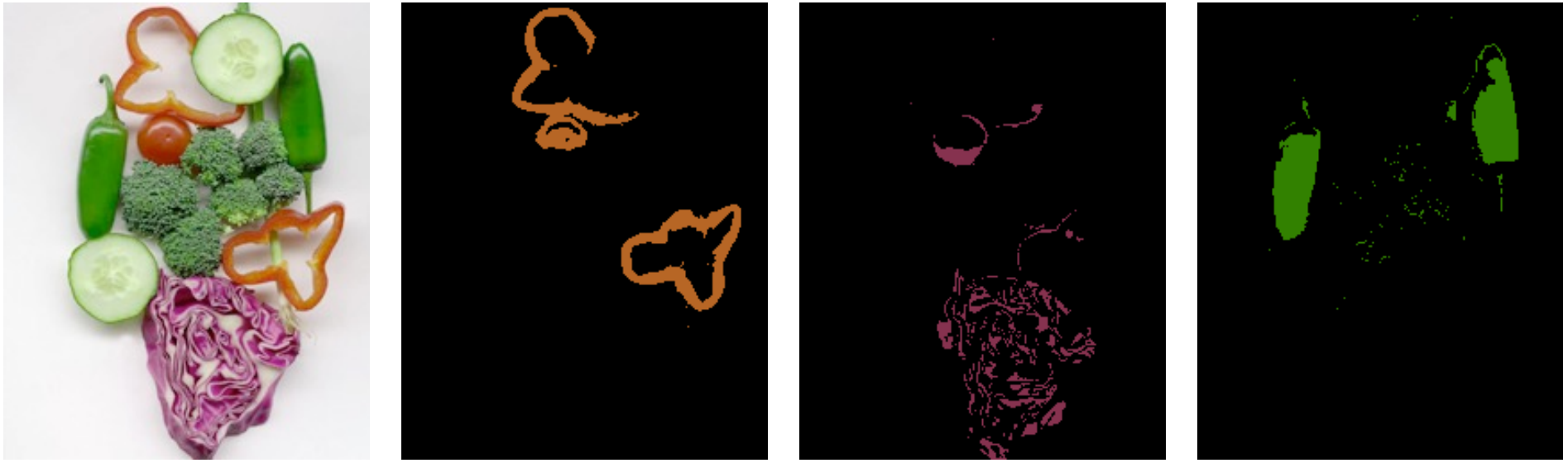
Clusters on intensity

Clusters on color

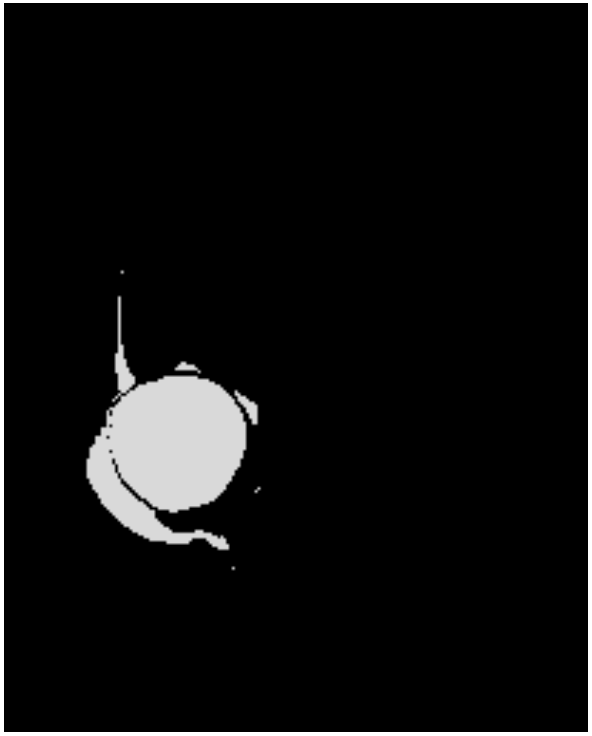


Based Only on Color

- 11 different regions



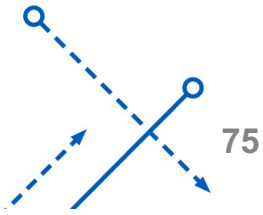
w/ Color and Position



How to choose the number of clusters?

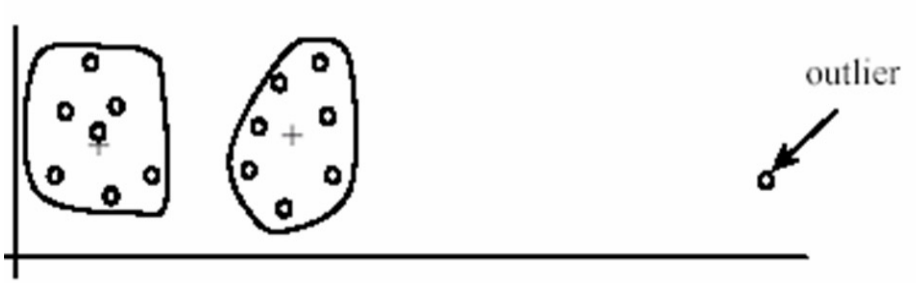
- On validation set

- Try different numbers of clusters and look at performance
  - When building dictionaries (discussed later), more clusters typically work better

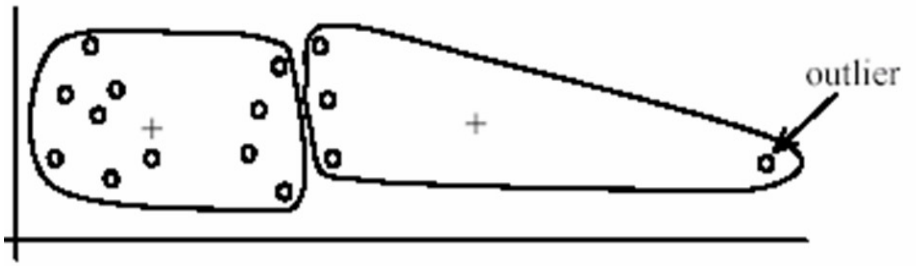


K-Means pros and cons

- Pros
  - Finds cluster centers that minimize conditional variance (good representation of data)
  - Simple and fast*
  - Easy to implement
- Cons
  - Need to choose K
  - Sensitive to outliers
  - Prone to local minima
  - All clusters have the same parameters (e.g., distance measure is non-adaptive)
  - *Can be slow: each iteration is O(KNd) for N d-dimensional points
- Usage
  - Rarely used for pixel segmentation



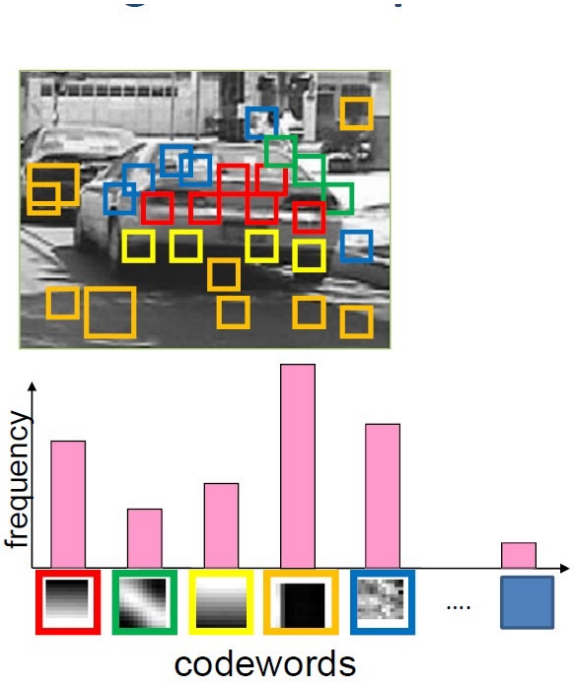
(B): Ideal clusters

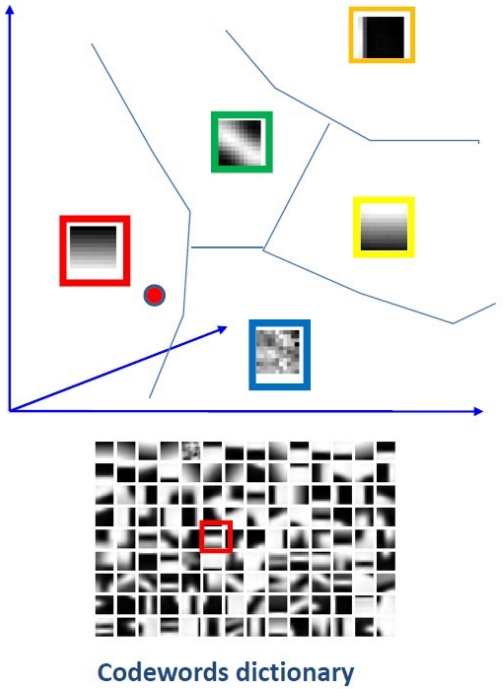


76

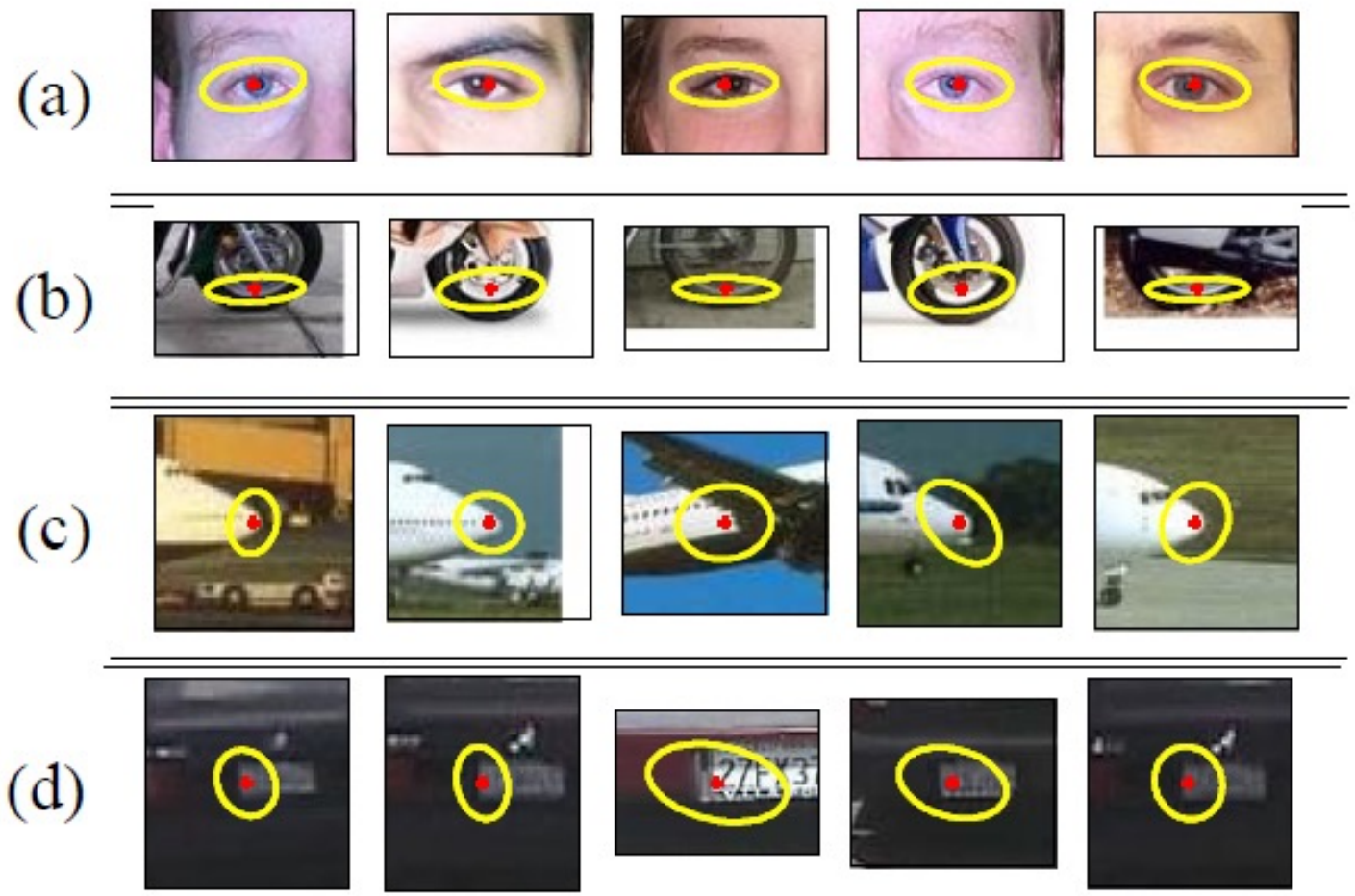
Building Visual Dictionaries

1. Sample patches from a database
   - E.g., 128 dim SIFT
2. Cluster the patches
   - Cluster centers are the dictionary
3. Assign a codeword (number) to each new patch, according to the nearest cluster





Examples of learned codewords

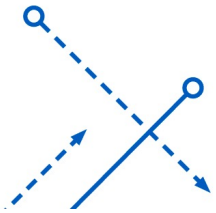


Most likely codewords for 4 learned “topics”
EM with multinomial (problem 3) to get topics

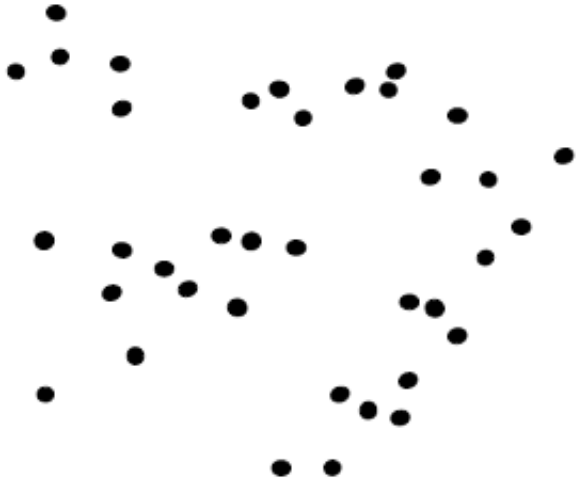
Sivic et al. ICCV 2005

How do we cluster?

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Single-link clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of PDF
- Spectral clustering
  - Split the nodes in a graph-based on assigned links with similarity weights

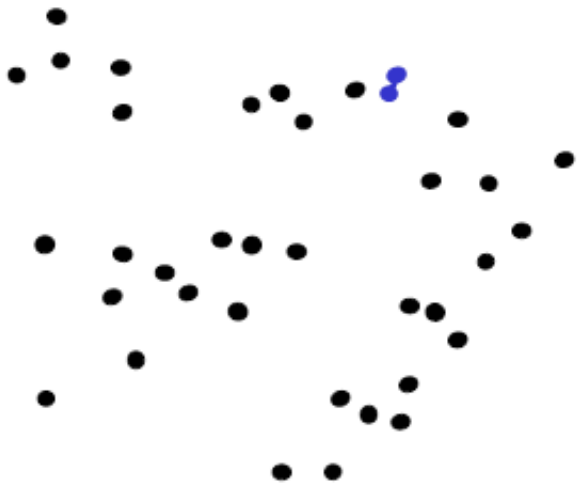


Single Link Clustering

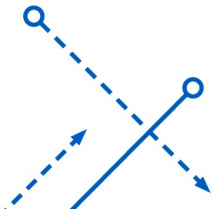


1. Say "Every point is its own cluster"

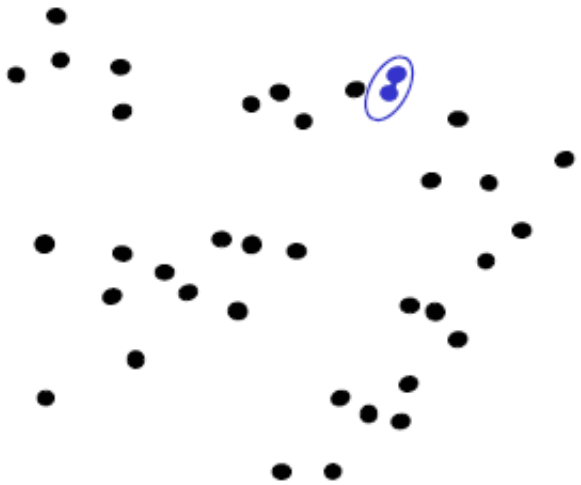
Single Link Clustering



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters

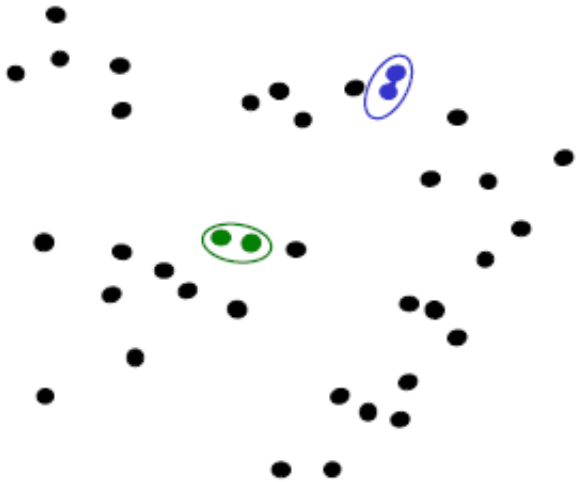


Single Link Clustering



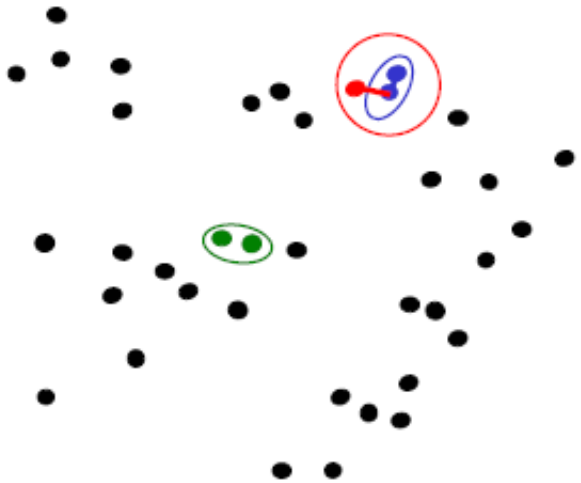
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster

Single Link Clustering



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

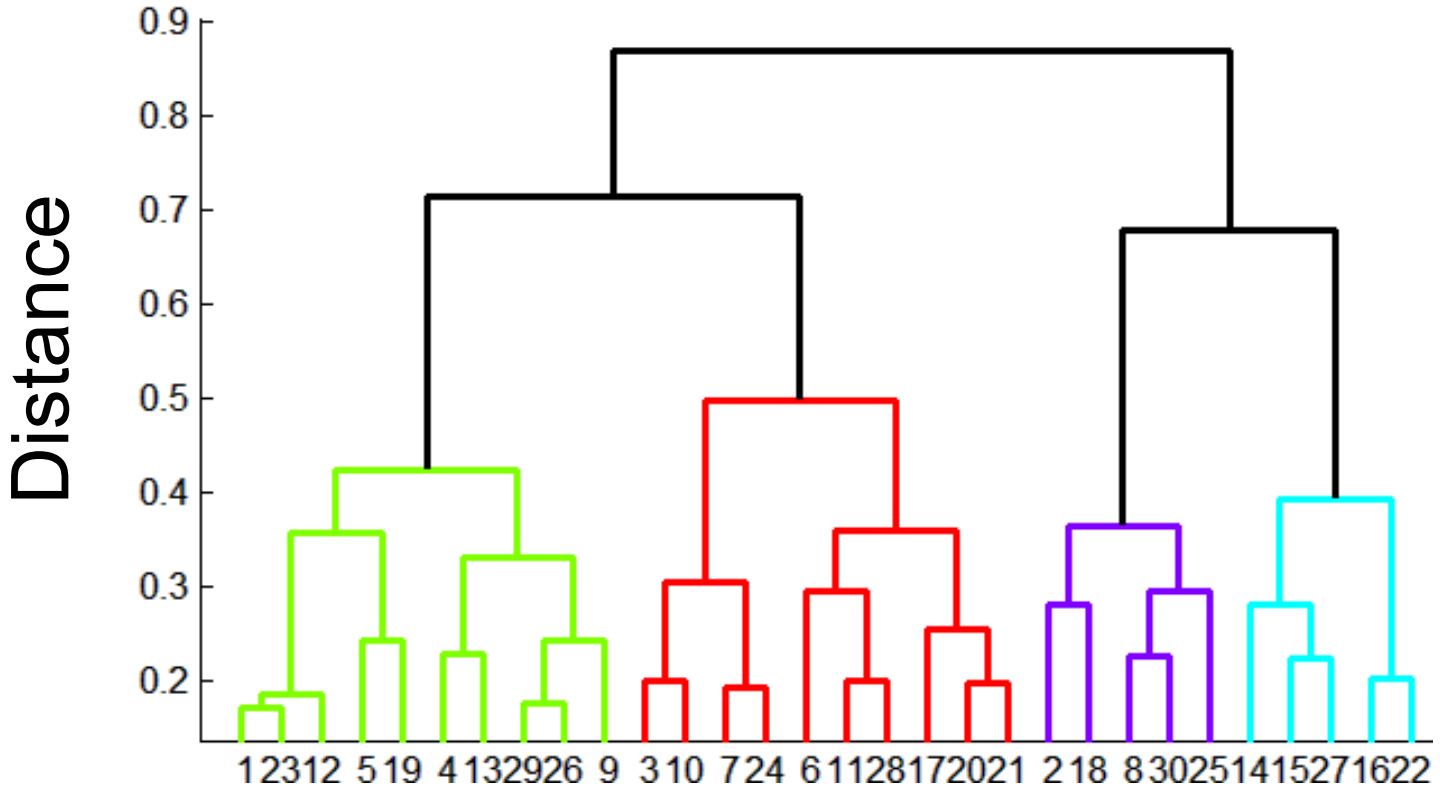
Single Link Clustering



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

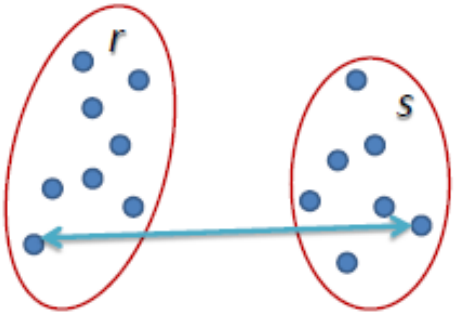
Single Link Clustering

- How many clusters?
  - Threshold based on max number of clusters or based on distance between merges

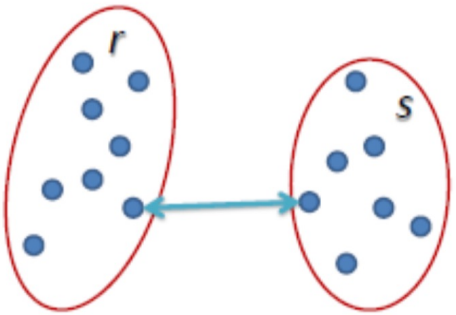


Single Link Clustering

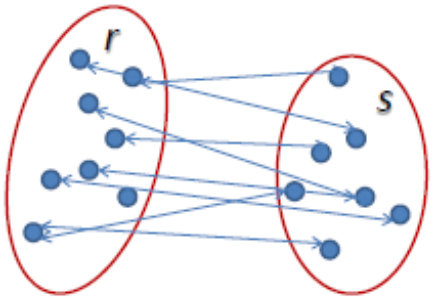
- How to define cluster similarity?
  - Average/maximum/minimum distance of points
  - Distance between means/medoids.



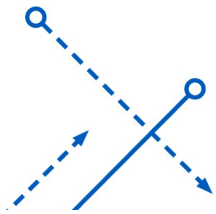
$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$



$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$



$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$



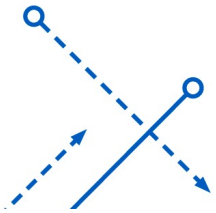
Summary: Single Link Clustering

Good

- Simple to implement, widespread application
- Clusters have adaptive shapes
- Provides a hierarchy of clusters

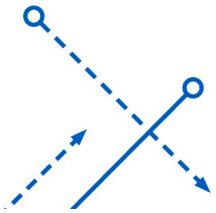
Bad

- May have imbalanced clusters
- Still have to choose number of clusters or threshold
- Need to use an “ultrametric” to get a meaningful hierarchy



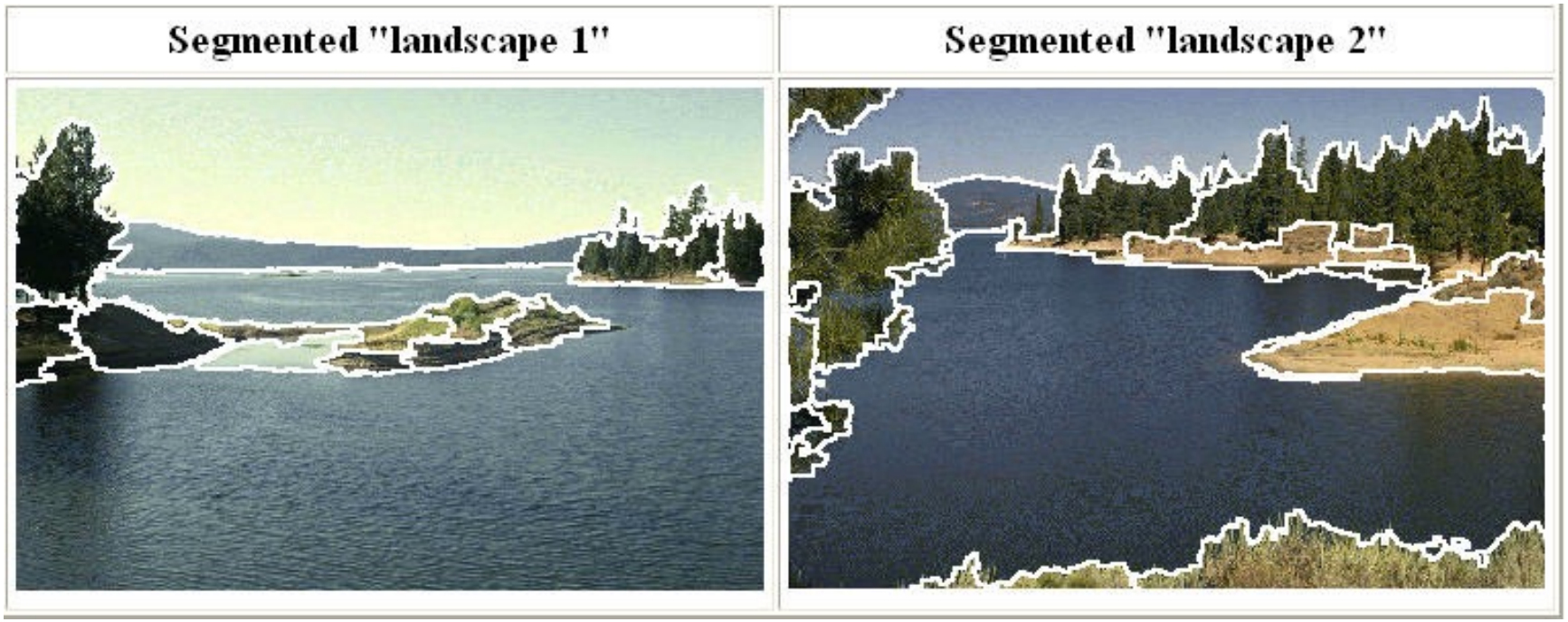
How do we cluster?

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Single-link clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of PDF
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights



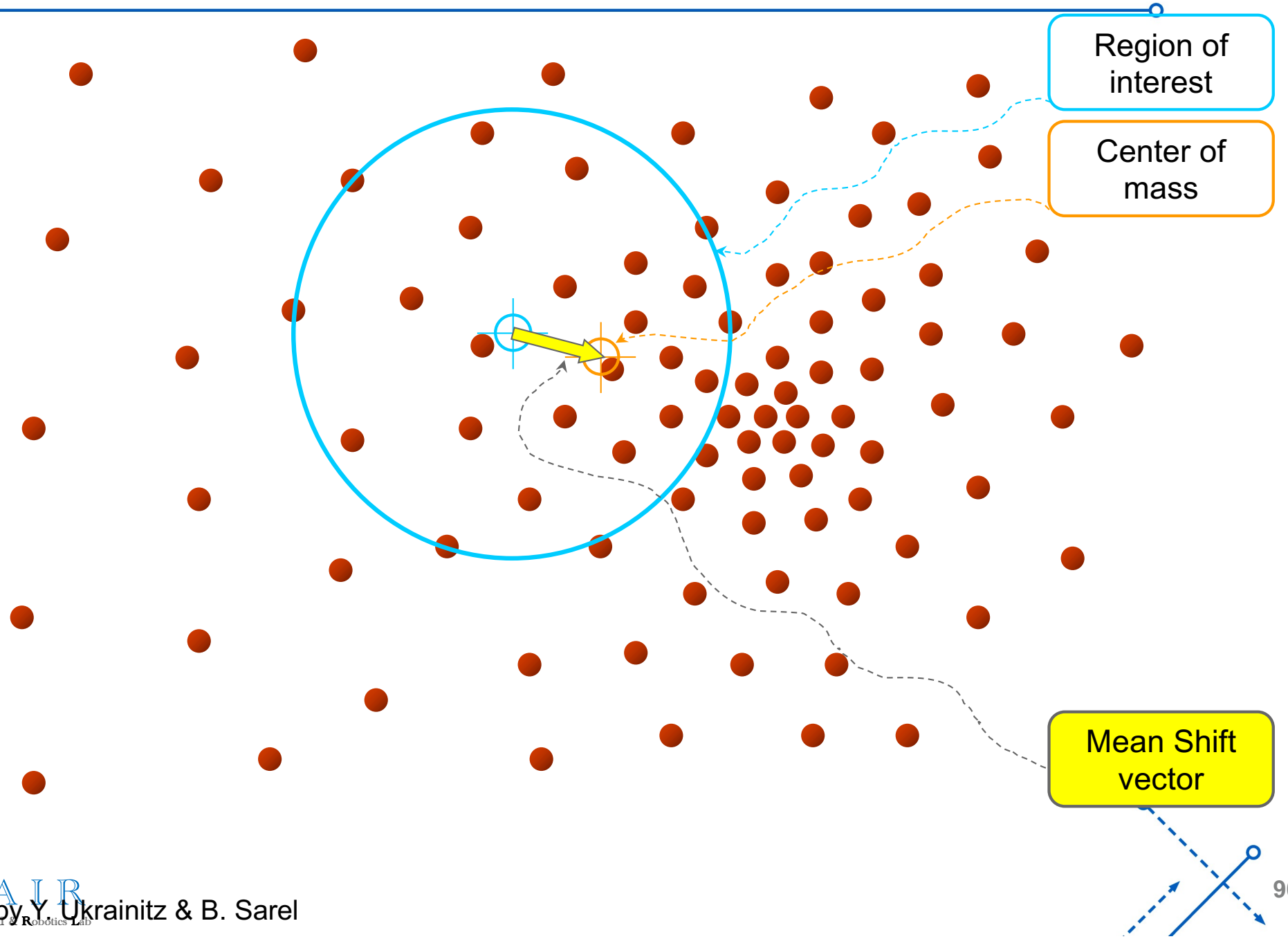
Mean shift algorithm

- Non-parametric feature-space analysis for locating the maxima of a density function.
- Versatile technique for clustering-based segmentation.



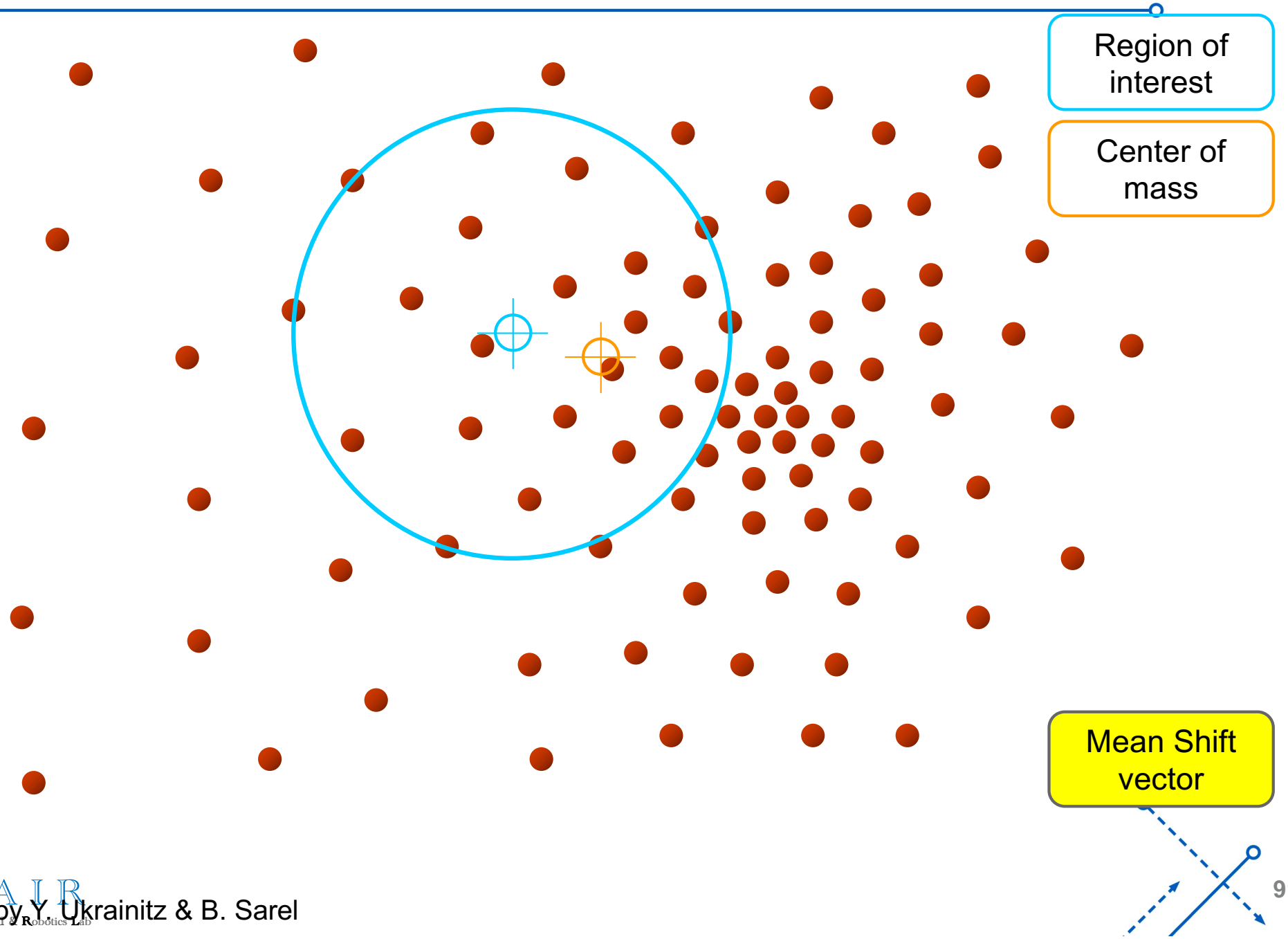
D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002.

Mean shift



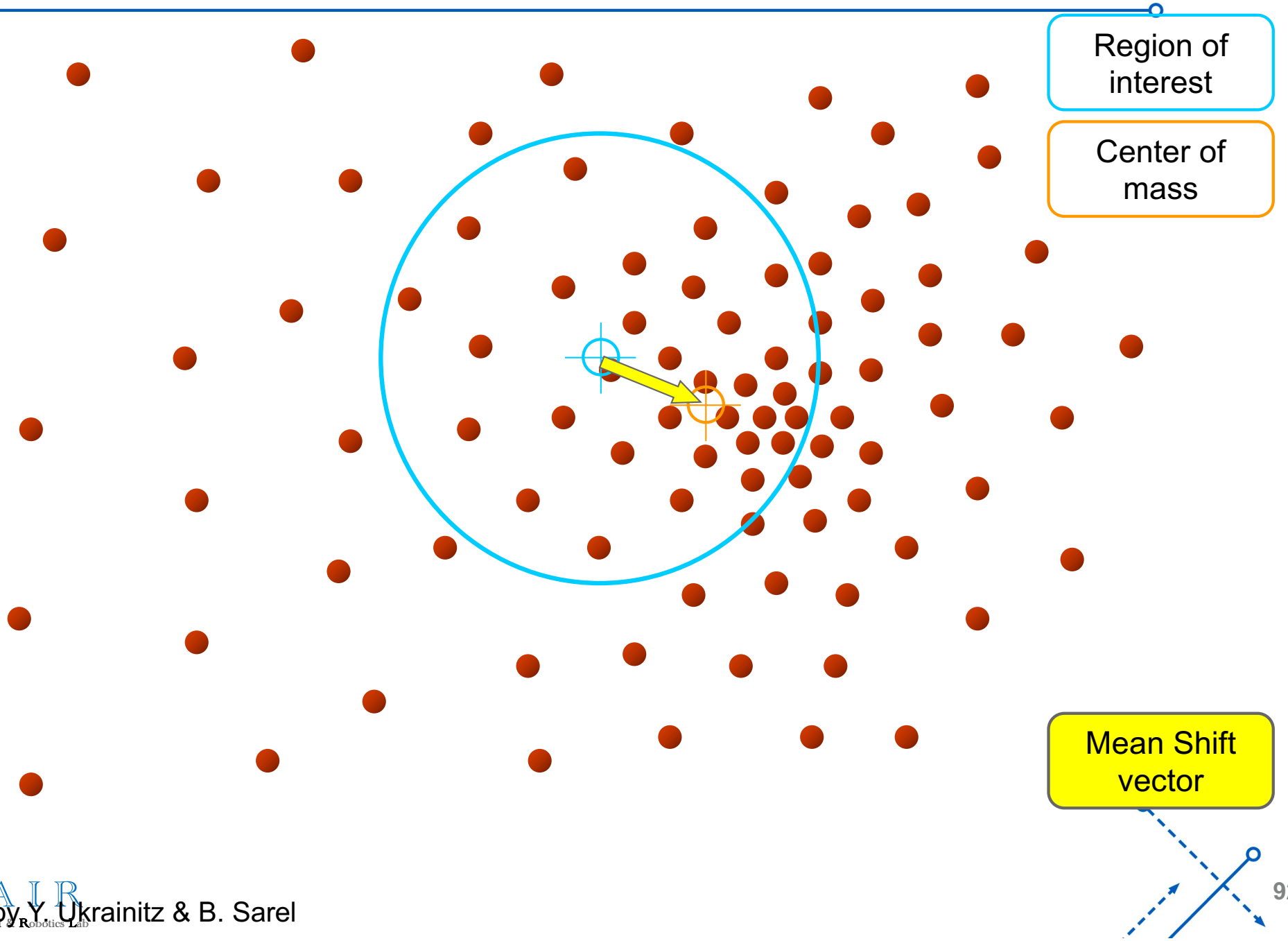
Slide by Y. Ukrainitz & B. Sarel

Mean shift



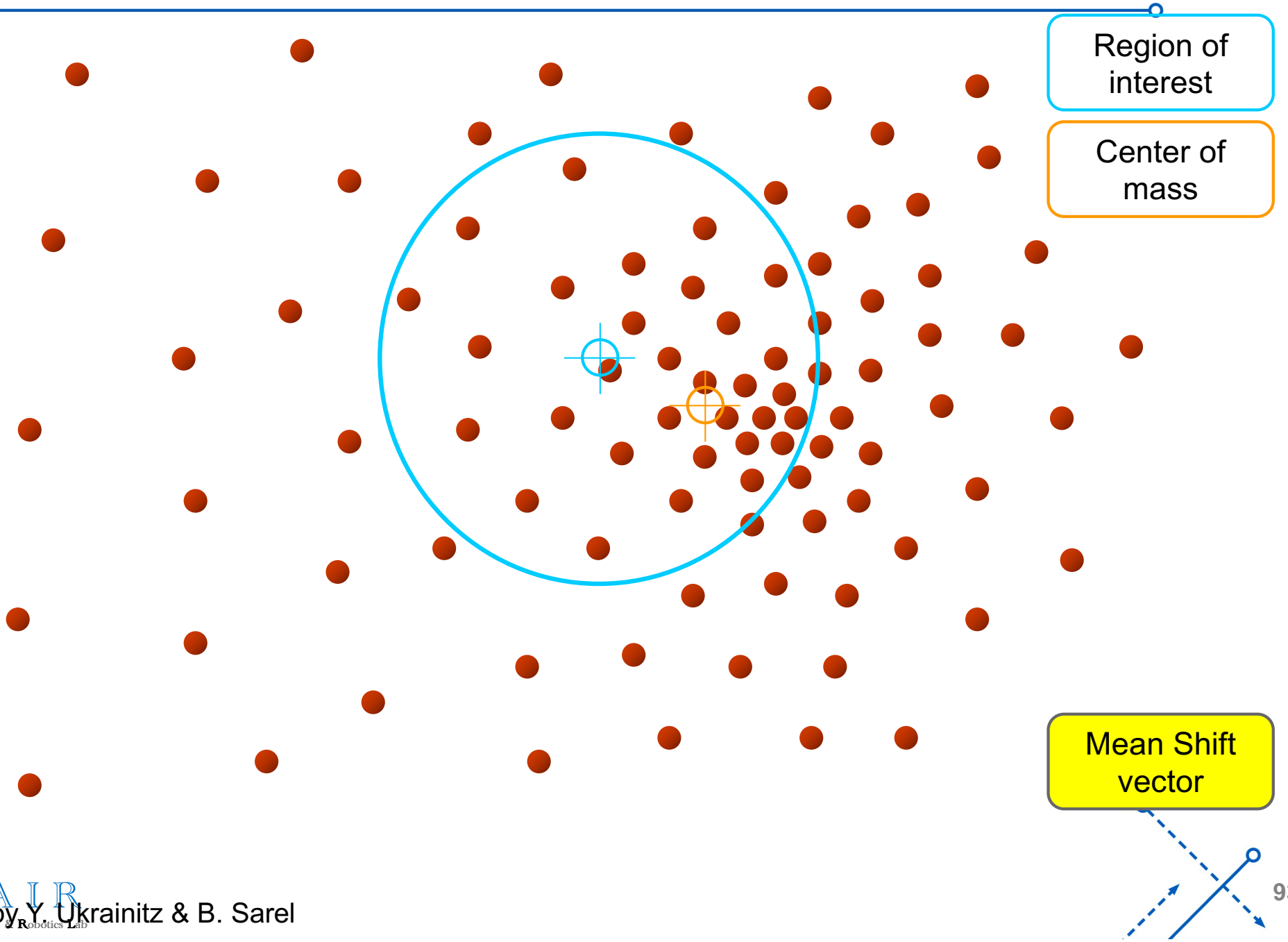
Slide by Y. Ukrainitz & B. Sarel

Mean shift



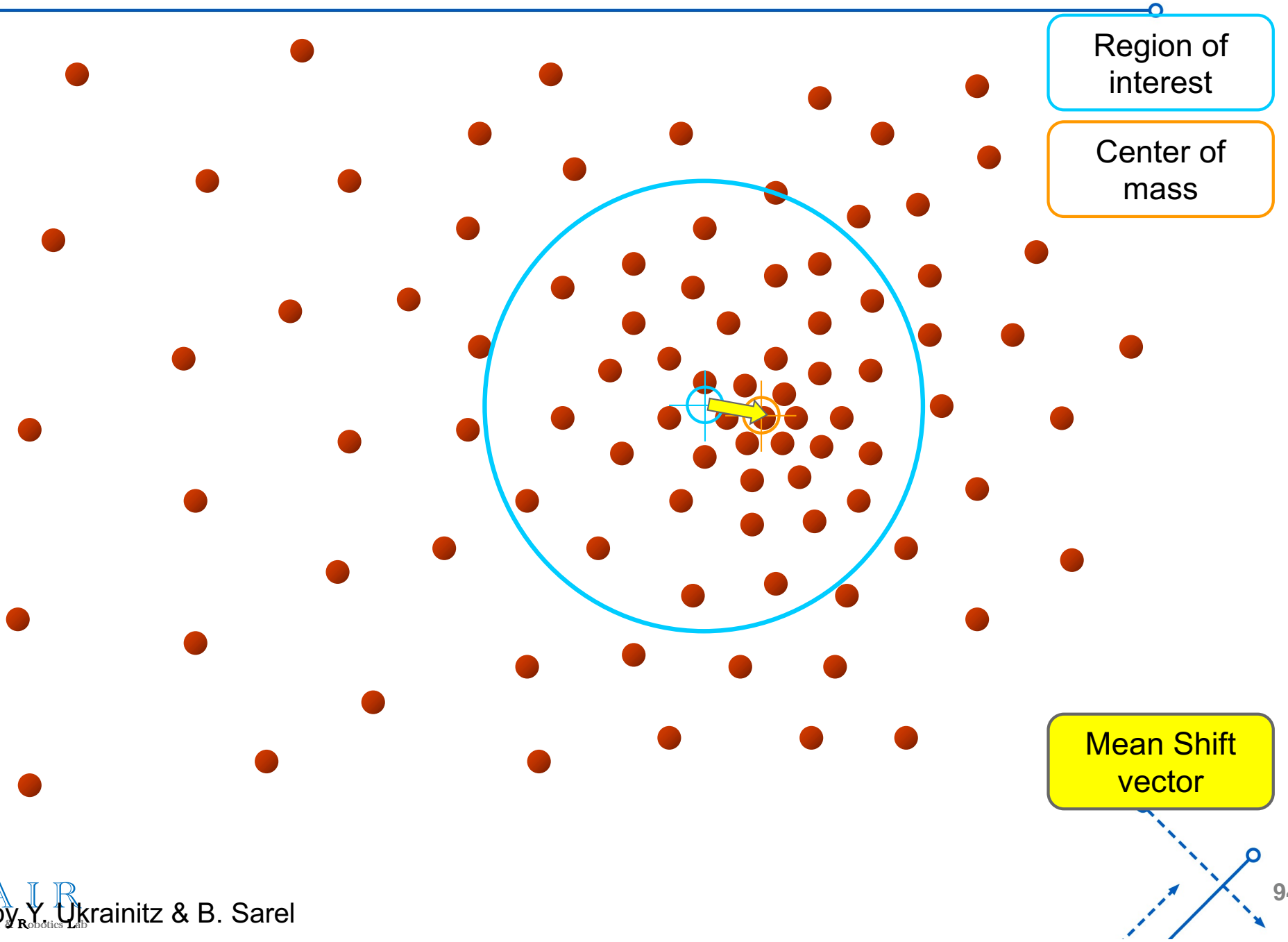
Slide by Y. Ukrainitz & B. Sarel

Mean shift



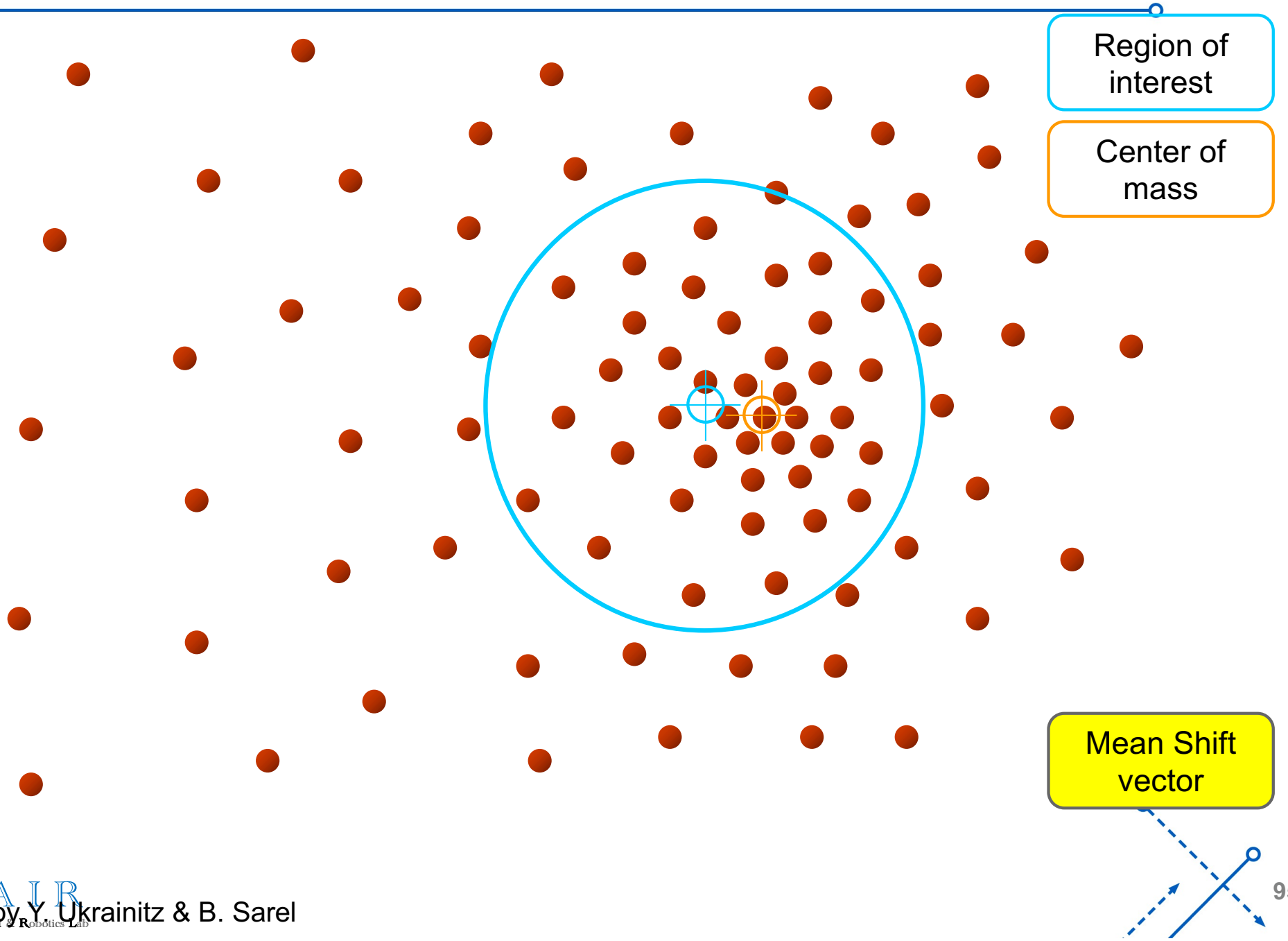
Slide by Y. Ukrainitz & B. Sarel

Mean shift



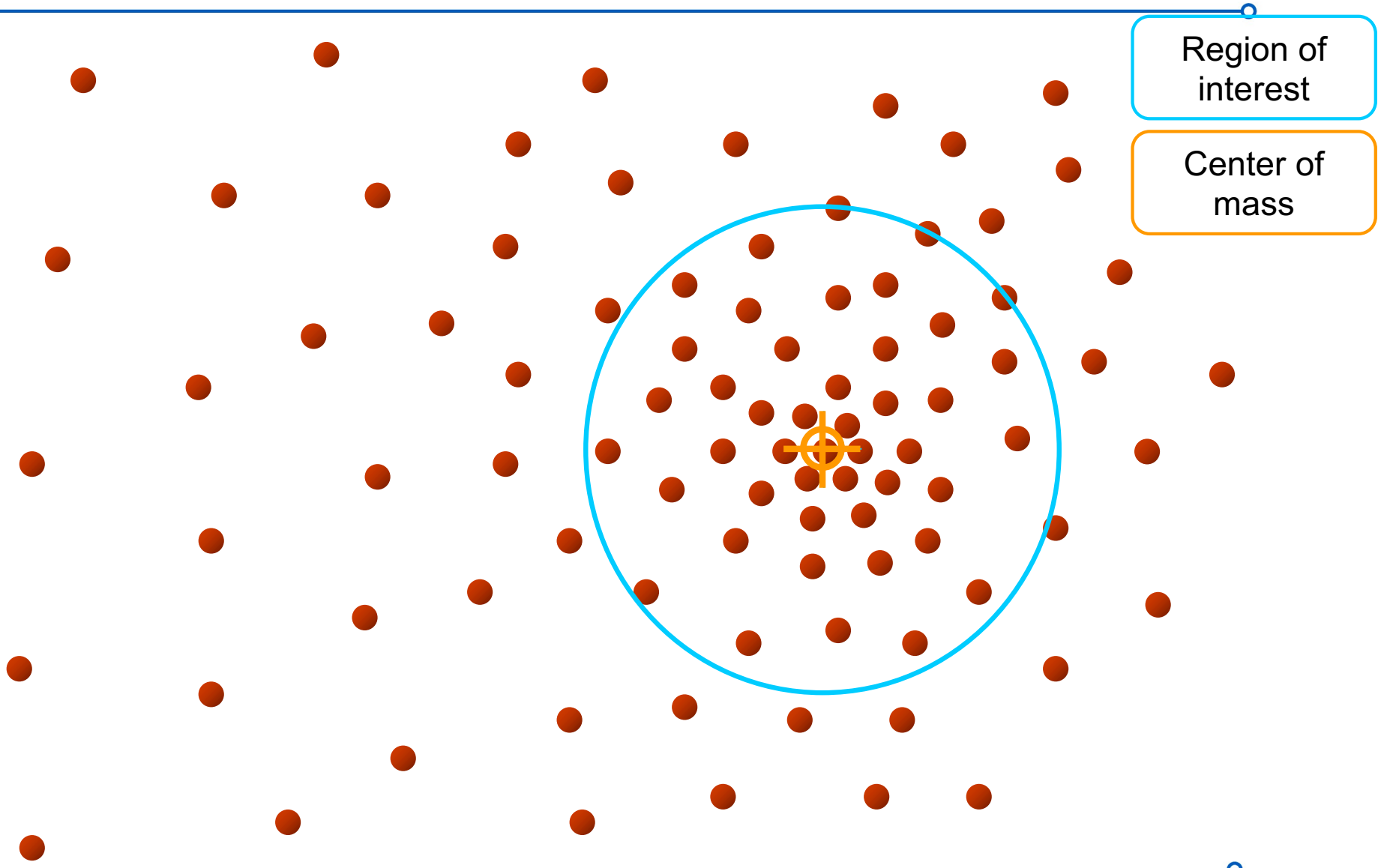
Slide by Y. Ukrainitz & B. Sarel

Mean shift



Slide by Y. Ukrainitz & B. Sarel

Mean shift

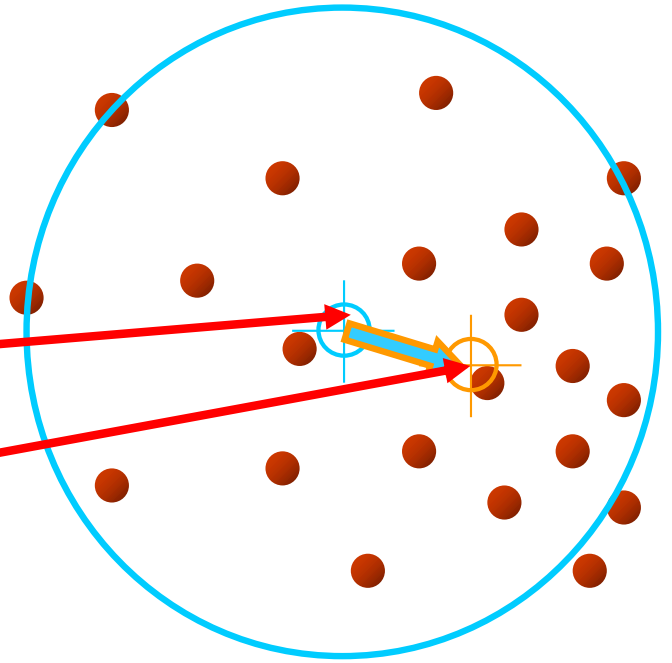


Slide by Y. Ukrainitz & B. Sarel

Computing the Mean Shift

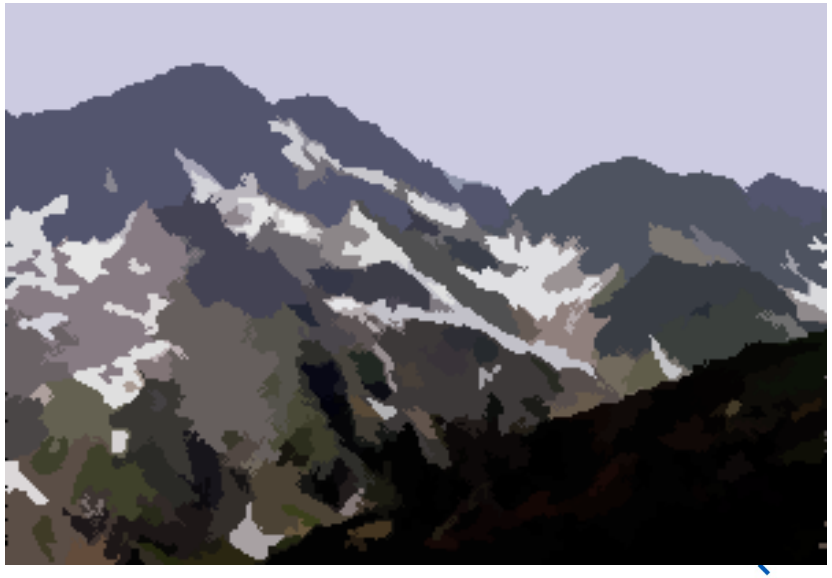
- Compute mean shift vector.
- Translate mean by m(x), weighted by kernel function.

$$\mathbf{m}(\mathbf{x}) = \left[\frac{\sum_{i=1}^{n} \mathbf{x}_i g\left(\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{h} \right)}{\sum_{i=1}^{n} g\left(\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{h} \right)} \right] - \mathbf{x}$$



$$g(x_i - x) = e^{-c||x_i - x||^2}$$

Mean shift segmentation results



Mean shift segmentation results

Comaniciu and Meer 2002

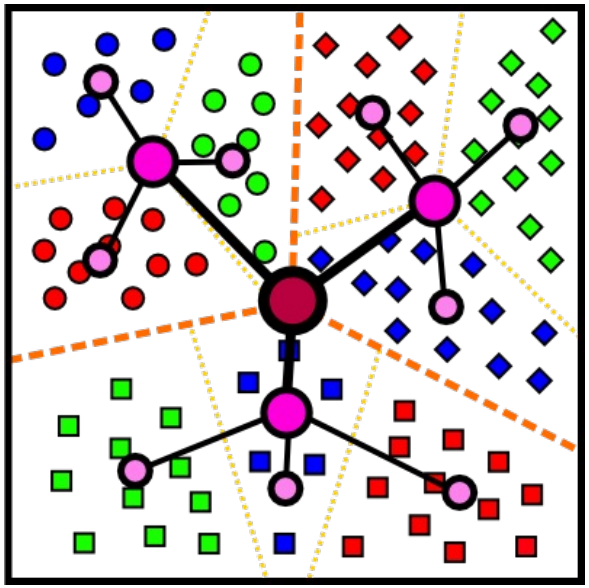
Mean shift pros and cons

- Pros
  - Good general-practice segmentation
  - Flexible in number and shape of regions
  - Robust to outliers
- Cons
  - Have to choose kernel size in advance
    - Use kNN to determine window sizes adaptively
  - Not suitable for high-dimensional features
- When to use it
  - Oversegmentatoin
  - Multiple segmentations
  - Tracking, clustering, filtering applications

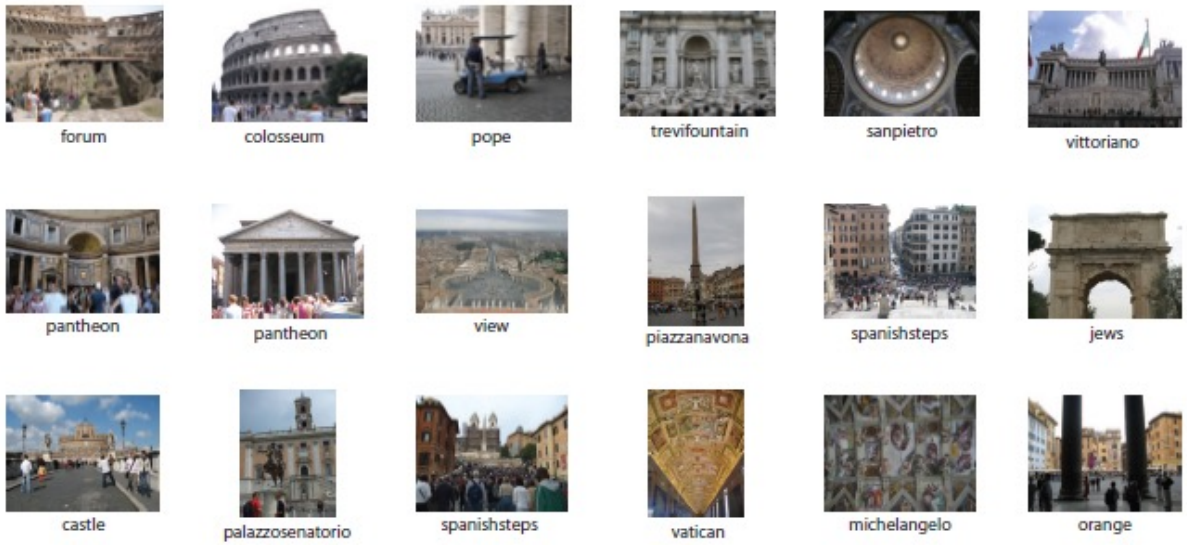
100

Which algorithm to use?

- Quantization/Summarization: K-means
  - Aims to preserve variance of original data
  - Can easily assign new point to a cluster



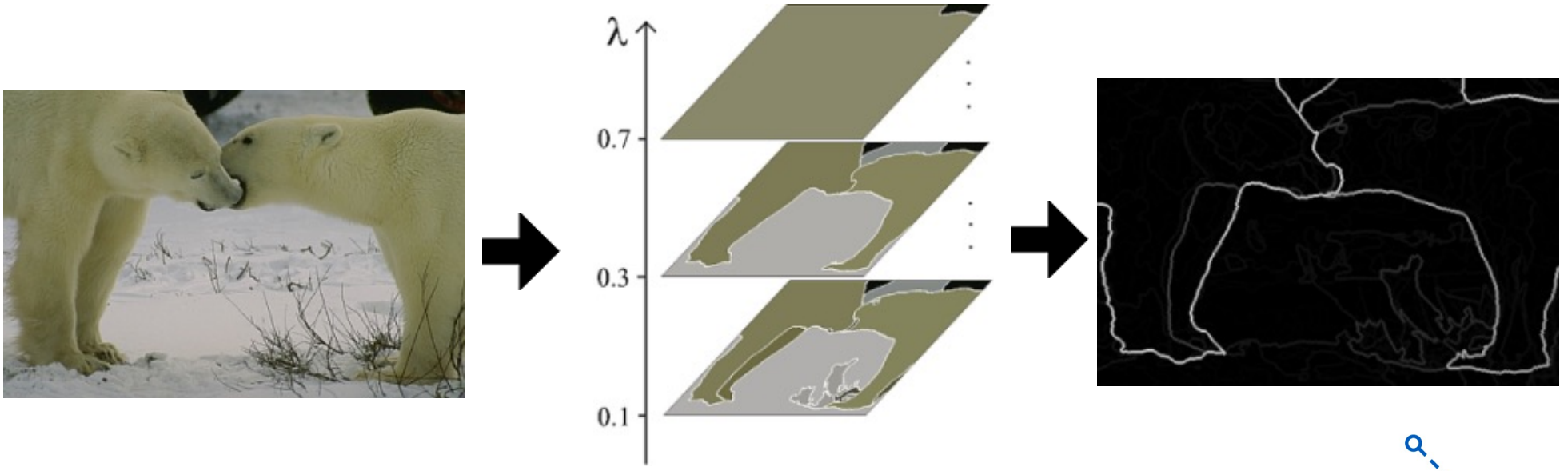
Quantization for computing histograms



Summary of 20,000 photos of Rome using “greedy k-means”

Which algorithm to use?

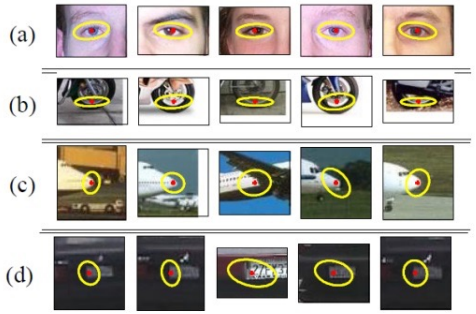
- Image segmentation: Single Link
  - More flexible with distance measures
    - e.g., can be based on boundary prediction
  - Adapts better to specific data
  - Hierarchy can be useful



http://www.cs.berkeley.edu/~arbelaez/UCM.html

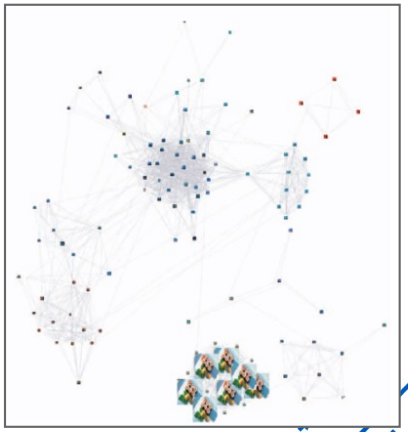
Things to remember

- K-means useful for summarization, building dictionaries of patches, general clustering



- Single link clustering useful for segmentation, general clustering

- Spectral clustering useful for determining relevance, summarization, segmentation



Motion Segmentation

Given a set of image points obtain:

- Number of independently moving objects
- Segmentation: object to which each point belongs
- Motion: rotation and translation of each object
- Structure: depth of each point

Segmentation by Flow

- Three frames from the “flower garden” sequence

- Given optical flow at each point
- Segment the flow field into regions belonging to individual planes “layers”.

Figure from “Representing Images with layers,”, by J. Wang and E.H. Adelson
IEEE Transactions on Image Processing, 1994, c 1994, IEEE
Some example slides from Forsythe and Ponce. Computer Vision, A modern approach.

Segmentation by Flow

- Segments and motion fields associated with them.
- Grey level shows region no. with the highest probability

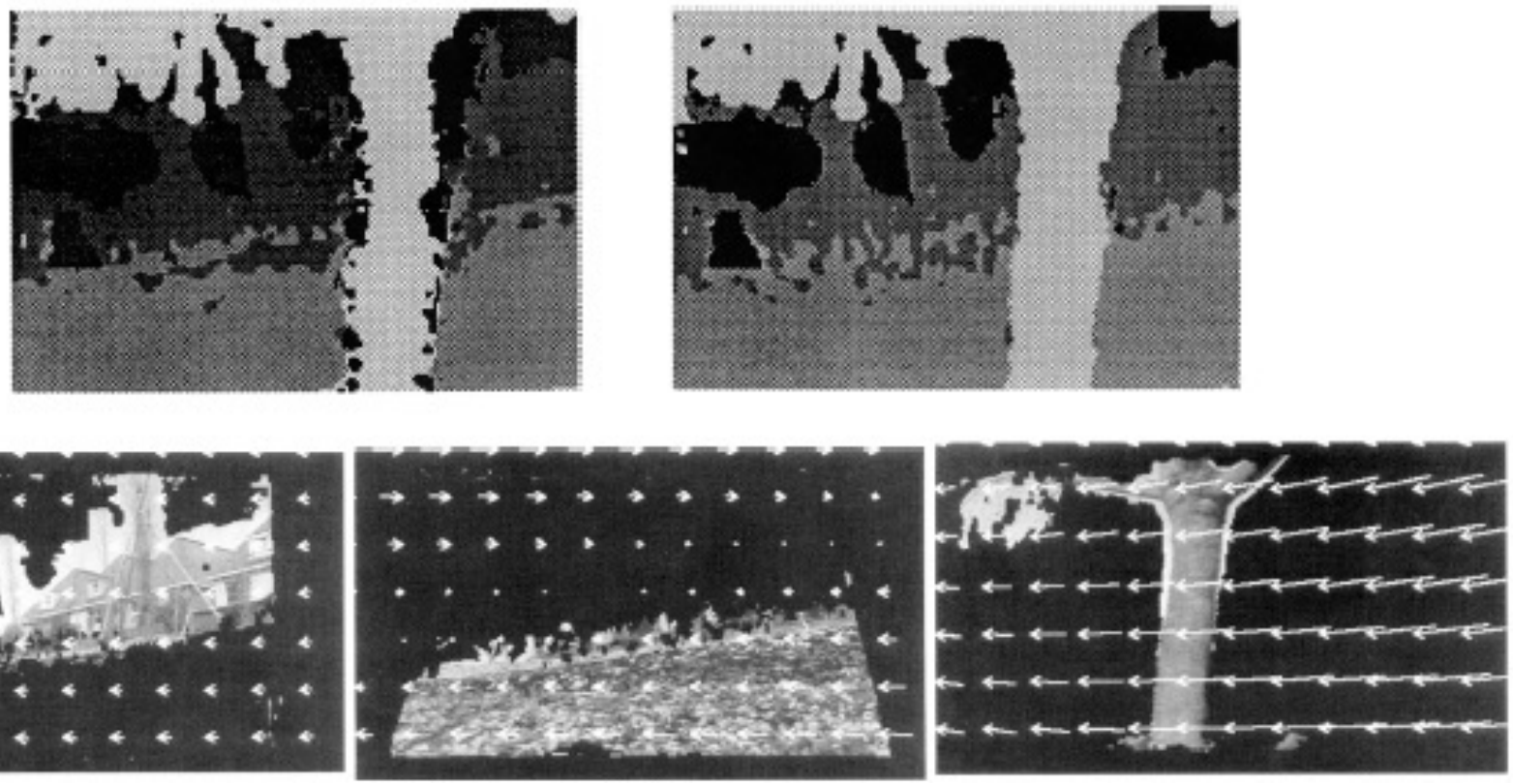


Figure from "Representing Images with layers,", by J. Wang and E.H. Adelson
IEEE Transactions on Image Processing, 1994, c 1994, IEEE
Some example slides from Forsythe and Ponce. Computer Vision, A modern approach.