Region-based Segmentation

Image Segmentation

- Group similar components (such as, pixels in an image, image frames in a video) to obtain a compact representation.
- Applications: Finding tumors, veins, etc. in medical images, finding targets in satellite/aerial images, finding people in surveillance images, summarizing video, etc.
- Methods: Thresholding, K-means clustering, etc.

Segmentation strategy

Edge-based

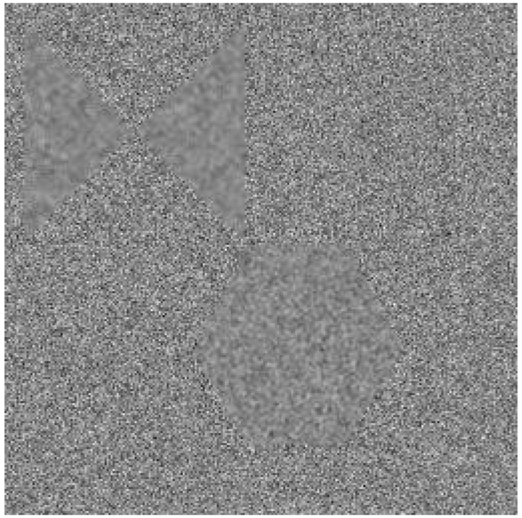
- Assumption: different objects are separated by edges (grey level discontinuities)
- The segmentation is performed by identifying the grey level gradients
- The same approach can be extended to color channels

Region-based

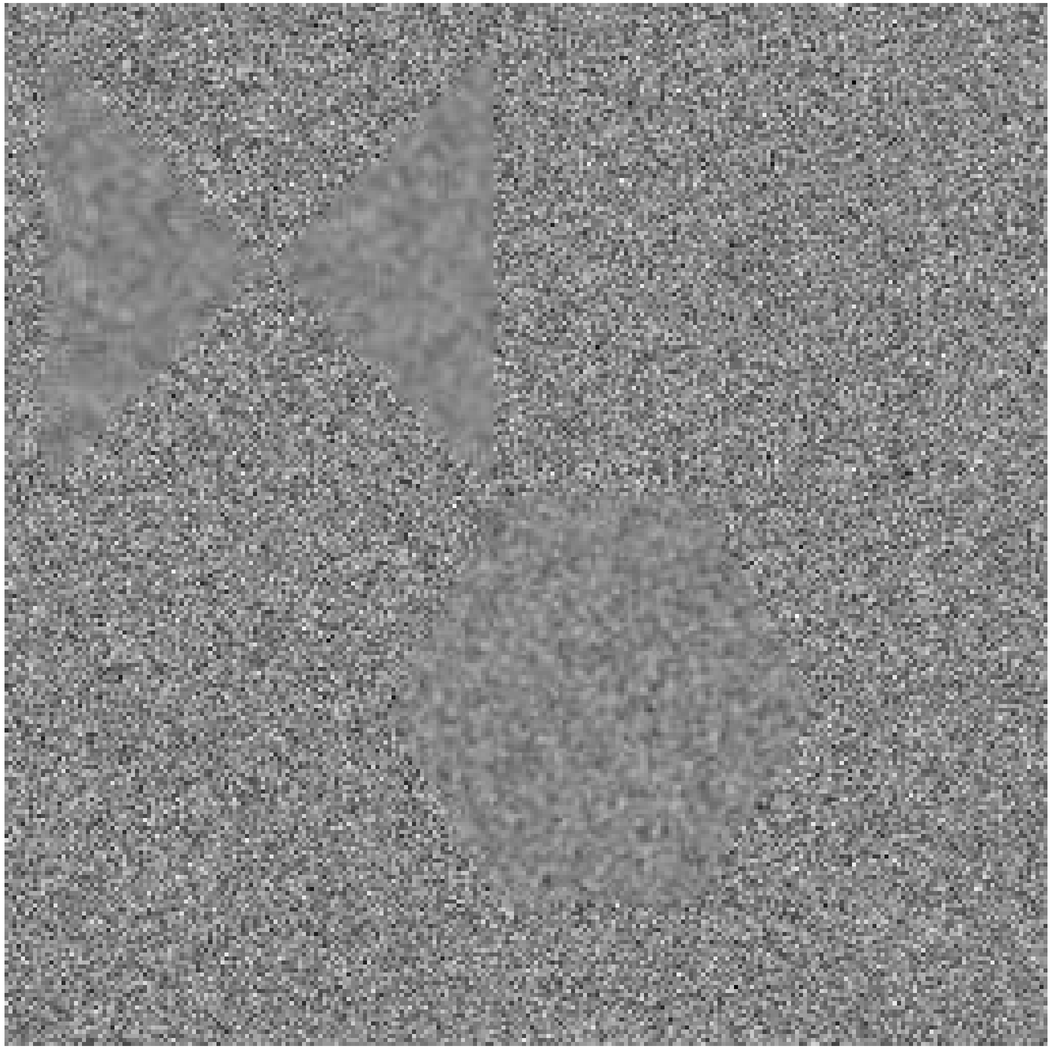
- Assumption: different objects are separated by other kind of perceptual boundaries
  - neighborhood features
- Most often texture-based
  - Textures are considered as instantiations of underlying stochastic processes and analyzed under the assumptions that stationarity and ergodicity hold
- Method
  - Region-based features are extracted and used to define “classes”

Examples

original

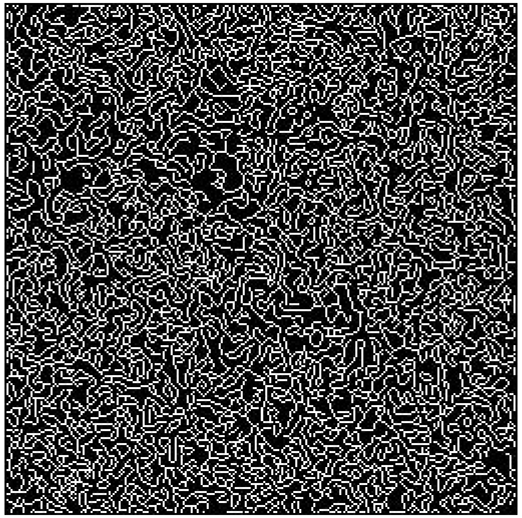


zoomed

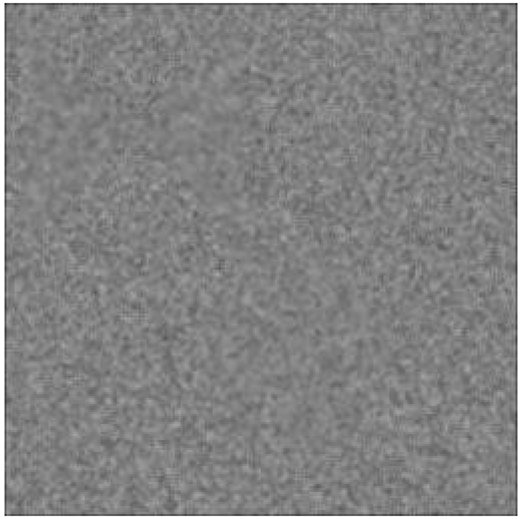


Examples

Canny



block mean



block std

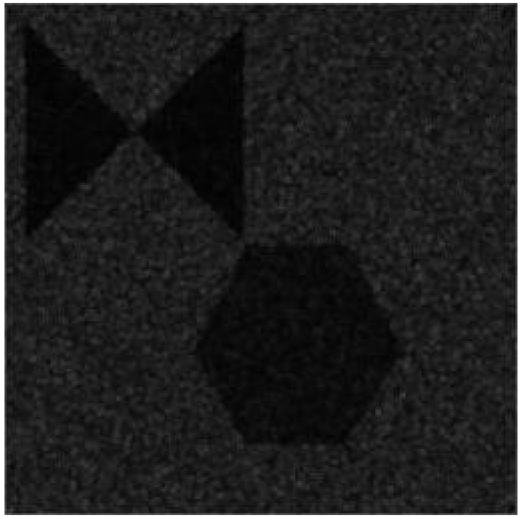


Image Segmentation

Contour-based

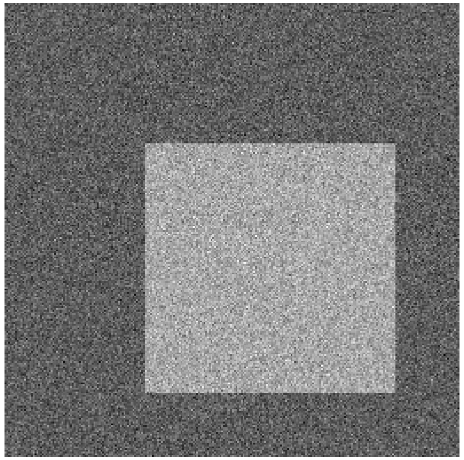
- Discontinuity
  - The approach is to partition an image based on *abrupt changes* in gray-scale levels.
  - The principal areas of interest within this category are detection of isolated points, lines, and edges in an image.

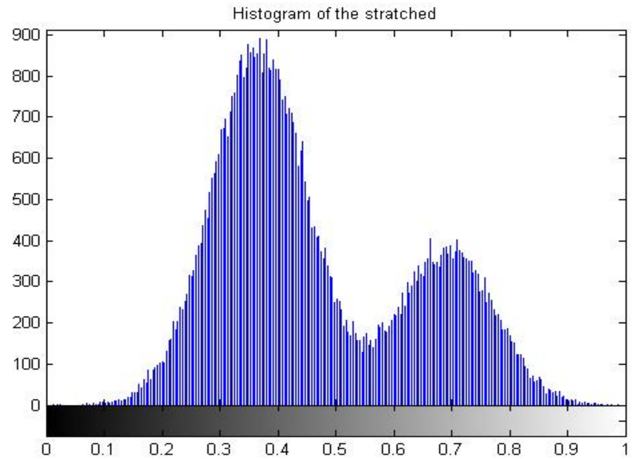
Region-based

- Similarity, homogeneity
- The principal approaches in this category are based on
  - thresholding,
  - region growing
  - region splitting/merging
  - clustering in feature space

Thresholding

- Image model
  - The objects in the image differ in the graylevel distribution
    - Simplest: object(s)+background
  - The spatial (image domain) stochastic parameters (i.e. mean, variance) are sufficient to characterize each object category
    - rests on the ergodicity assumption
  - Easily generalized to multi-spectral images (i.e. color images)





Thresholding

- Individual pixels in an image are marked as “object” pixels if their value is greater than some threshold value and as “background” pixels otherwise → *threshold above*
  - assuming an object to be brighter than the background
  - Variants
    - *threshold below*, which is opposite of threshold above;
    - *threshold inside*, where a pixel is labeled "object" if its value is between two thresholds
    - *threshold outside*, which is the opposite of threshold inside
  - Typically, an object pixel is given a value of “1” while a background pixel is given a value of “0.” Finally, a binary image is created by coloring each pixel white or black, depending on a pixel's label.

Thresholding types

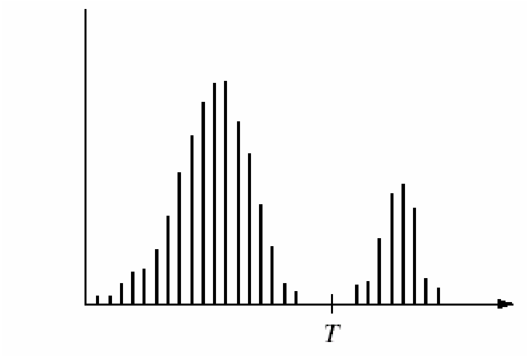
- Histogram shape-based methods
  - Peaks, valleys and curvatures of the smoothed histogram are analyzed
- Clustering-based methods
  - gray-level samples are clustered in two parts as background and foreground (object), or alternately are modeled as a mixture of two Gaussians
- Entropy-based methods
  - Entropy of the foreground and background regions, cross-entropy between the original and segmented image, etc.
- Object attribute-based methods
  - Based on a measure of similarity between the gray-level and the binarized images, such as fuzzy shape similarity, edge coincidence, etc.

Thresholding types

- Stchastic methods using higher-order probability distributions and/or correlation between pixels
- Local methods adapt the threshold value on each pixel to the local image characteristics

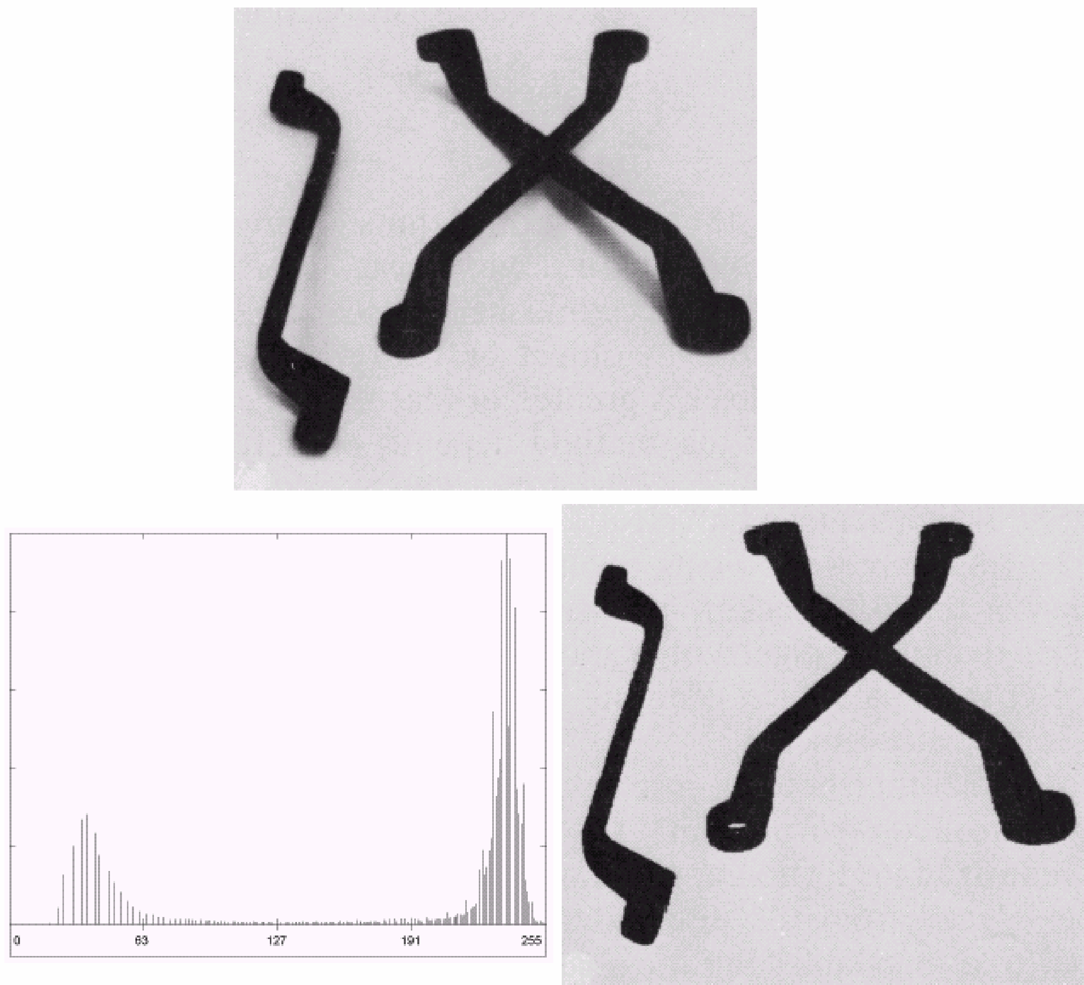
Histogram thresholding

- Suppose that an image, f(x,y), is composed of light objects on a dark background, and the following figure is the histogram of the image.



- Then, the objects can be extracted by comparing pixel values with a threshold T.

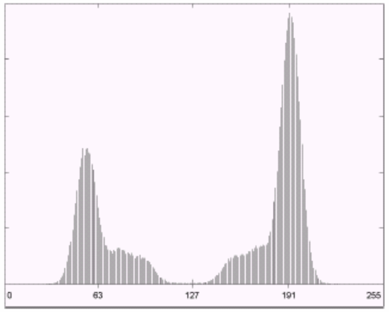
Thresholding



a
b c

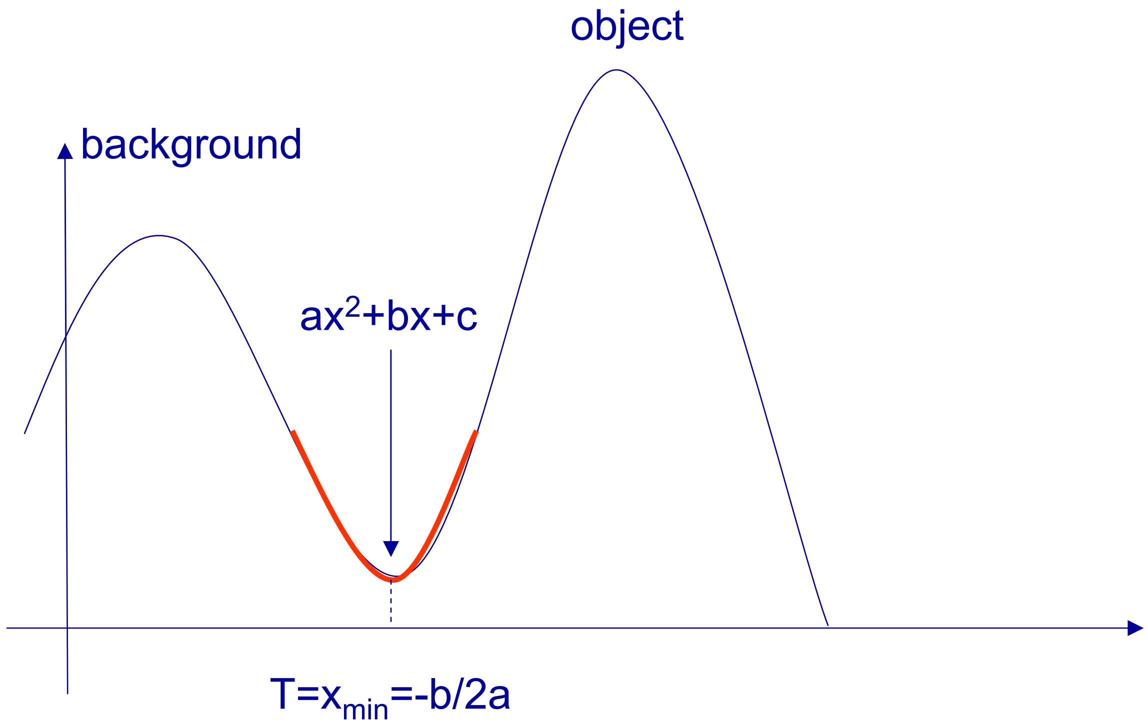
FIGURE 10.28 (a) Original image. (b) Image histogram. (c) Result of global thresholding with T midway between the maximum and minimum gray levels.

Thresholding



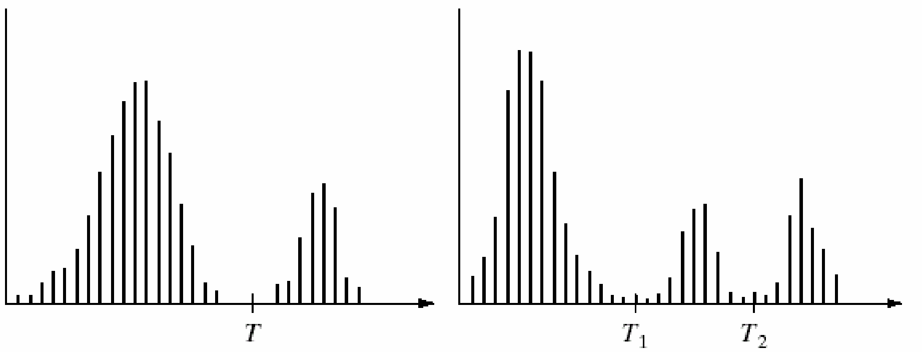
Histogram thresholding

- Analytical models can be fit to the valleys of the histogram and then used to find local minima



Multiple histogram thresholding

- It is also possible to extract objects that have a specific intensity range using multiple thresholds.



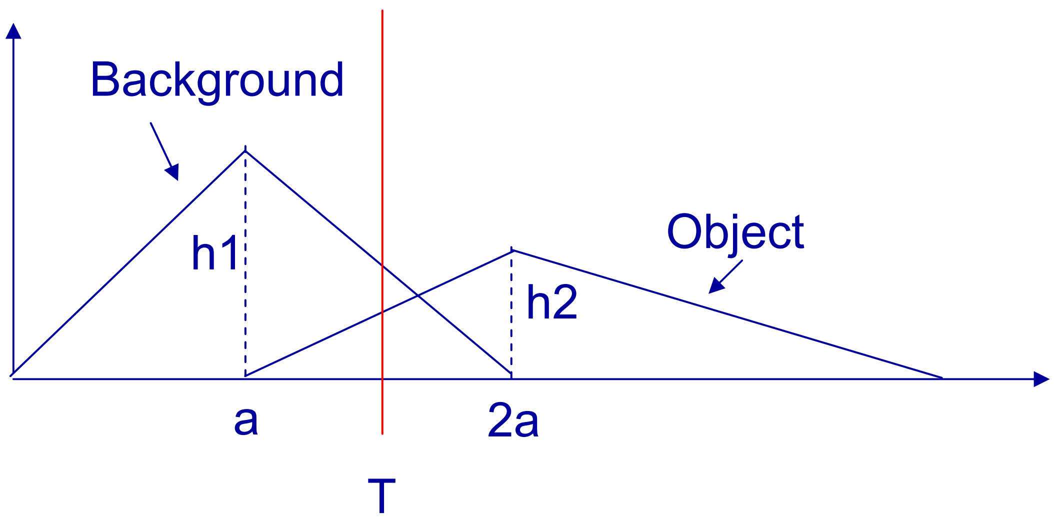
a b

FIGURE 10.26 (a) Gray-level histograms that can be partitioned by (a) a single threshold, and (b) multiple thresholds.

Extension to color images is straightforward: There are three color channels, in each one specifies the intensity range of the object… Even if objects are not separated in a single channel, they might be with all the channels… Application example: Detecting/Tracking faces based on skin color…

Clustering based thresholding

- Exercise: Cost of classifying a background pixel as an object pixel is Cb.
- Cost of classifying an object pixel as a background pixel is Co.
- Find the threshold, T, that minimizes the total cost.



Clustering based thresholding

- Idea 1: pick a threshold such that each pixel on each side of the threshold is closer in intensity to the mean of all pixels on that side of the threshold than the mean of all pixels on the other side of the threshold. Let

  $\mu_B(T)$ = the mean of all pixels less than the threshold (background)

  $\mu_O(T)$ = the mean of all pixels greater than the threshold (object)

- We want to find a threshold such that the greylevels for the object are closest to the average of the object and the greylevels for the background are closest to the average of the background:

$$\forall g \geq T \rightarrow |g - \mu_o(T)| < |g - \mu_B(T)|$$
$$\forall g < T \rightarrow |g - \mu_o(T)| \geq |g - \mu_B(T)|$$

Clustering based thresholding

- Idea 2: select T to minimize the within-class variance—the weighted sum of the variances of each cluster:

$$\sigma^2_{within}(T) = n_B(T)\sigma^2_B(T) + n_o(T)\sigma^2_o(T)$$

$$n_B(T) = \sum_{g=0}^{T-1} p(g)$$

$$n_o(T) = \sum_{g=T}^{N-1} p(g)$$

$\sigma^2_B(T)$: variance of the pixels in the background (g<T)

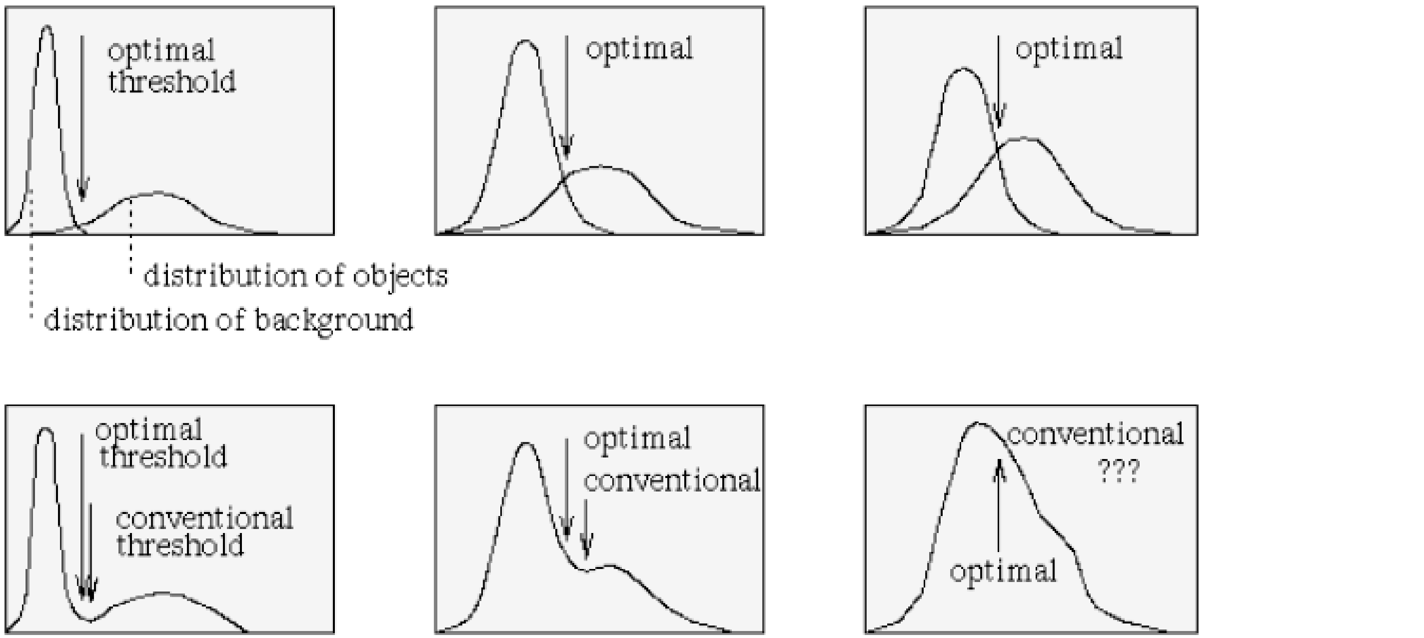
$\sigma^2_0(T)$: variance of the pixels in the object (g $\geq$ T)

$0,\ldots,N-1$: range of intensity levels

Clustering based thresholding

- Idea 3: Modeling the pdf as the superposition of two Gaussians and take the overlapping point as the threshold

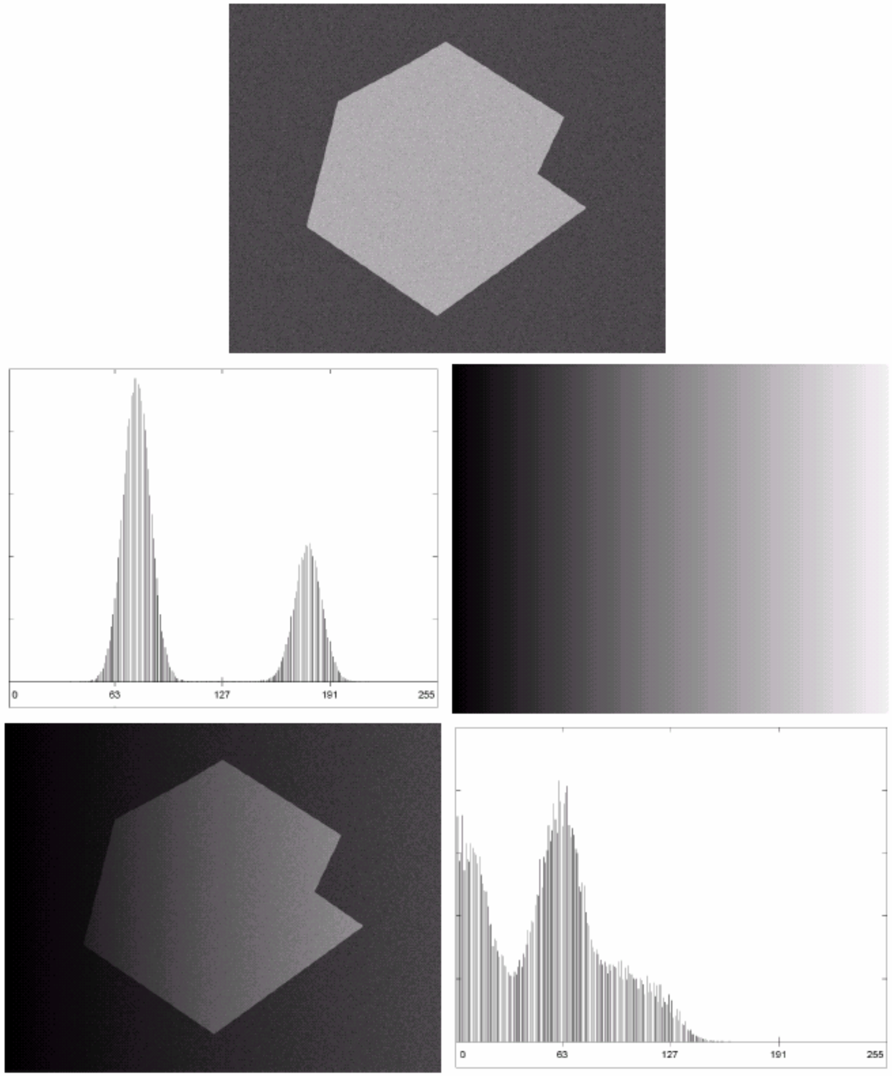
$$h(x) = P_1 p_1(x) + P_2 p_2(x) = \frac{P_1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{1}{2}\left(\frac{x-\mu_1}{\sigma_1}\right)^2} + \frac{P_2}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{1}{2}\left(\frac{x-\mu_2}{\sigma_2}\right)^2}$$



Thresholding

- Non-uniform illumination may change the histogram in a way that it becomes impossible to segment the image using a single global threshold.
- Choosing local threshold values may help.

Thresholding



a
b c
d e

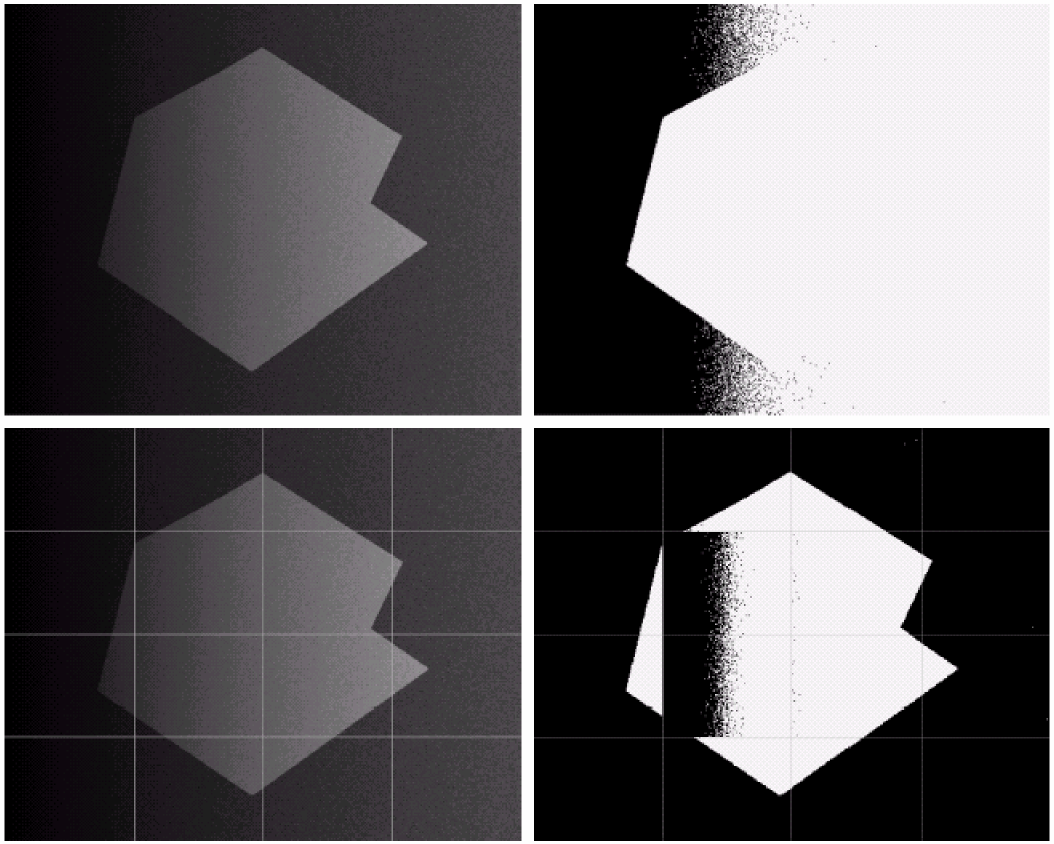
FIGURE 10.27
(a) Computer generated reflectance function.
(b) Histogram of reflectance function.
(c) Computer generated illumination function.
(d) Product of (a) and (c).
(e) Histogram of product image.

Thresholding

- Adaptive thresholding

a	b
c	d

FIGURE 10.30 (a) Original image. (b) Result of global thresholding. (c) Image subdivided into individual subimages. (d) Result of adaptive thresholding.



Region-Oriented Segmentation

- Region Growing
  - Region growing is a procedure that groups pixels or subregions into larger regions.
  - The simplest of these approaches is *pixel aggregation*, which starts with a set of "seed" points and from these grows regions by appending to each seed points those neighboring pixels that have similar properties (such as gray level, texture, color, shape).
  - Region growing based techniques are better than the edge-based techniques in noisy images where edges are difficult to detect.

Region-Oriented Segmentation

Suppose that we have the image given below.

(a) Use the region growing idea to segment the object. The seed for the object is the center of the image. Region is grown in horizontal and vertical directions, and when the difference between two pixel values is less than or equal to 5.

Table 1: Show the result of Part (a) on this figure.

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	<u>60</u>	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

Region-Oriented Segmentation

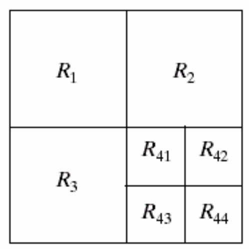
(b) What will be the segmentation if region is grown in horizontal, vertical, and diagonal directions?

Table 2: Show the result of Part (b) on this figure.

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	<u>60</u>	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

Region-Oriented Segmentation

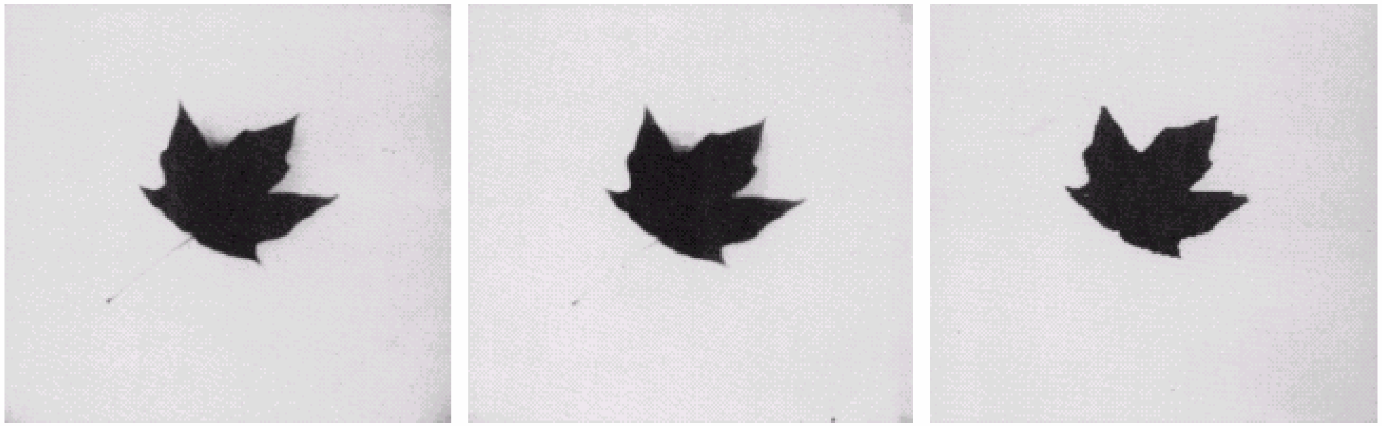
- Region Splitting
  - Region growing starts from a set of seed points.
  - An alternative is to start with the whole image as a single region and subdivide the regions that do not satisfy a condition of homogeneity.
- Region Merging
  - Region merging is the opposite of region splitting.
  - Start with small regions (e.g. 2x2 or 4x4 regions) and merge the regions that have similar characteristics (such as gray level, variance).
  - Typically, splitting and merging approaches are used iteratively.



Region-Oriented Segmentation

a b c

FIGURE 10.43
(a) Original image. (b) Result of split and merge procedure. (c) Result of thresholding (a).



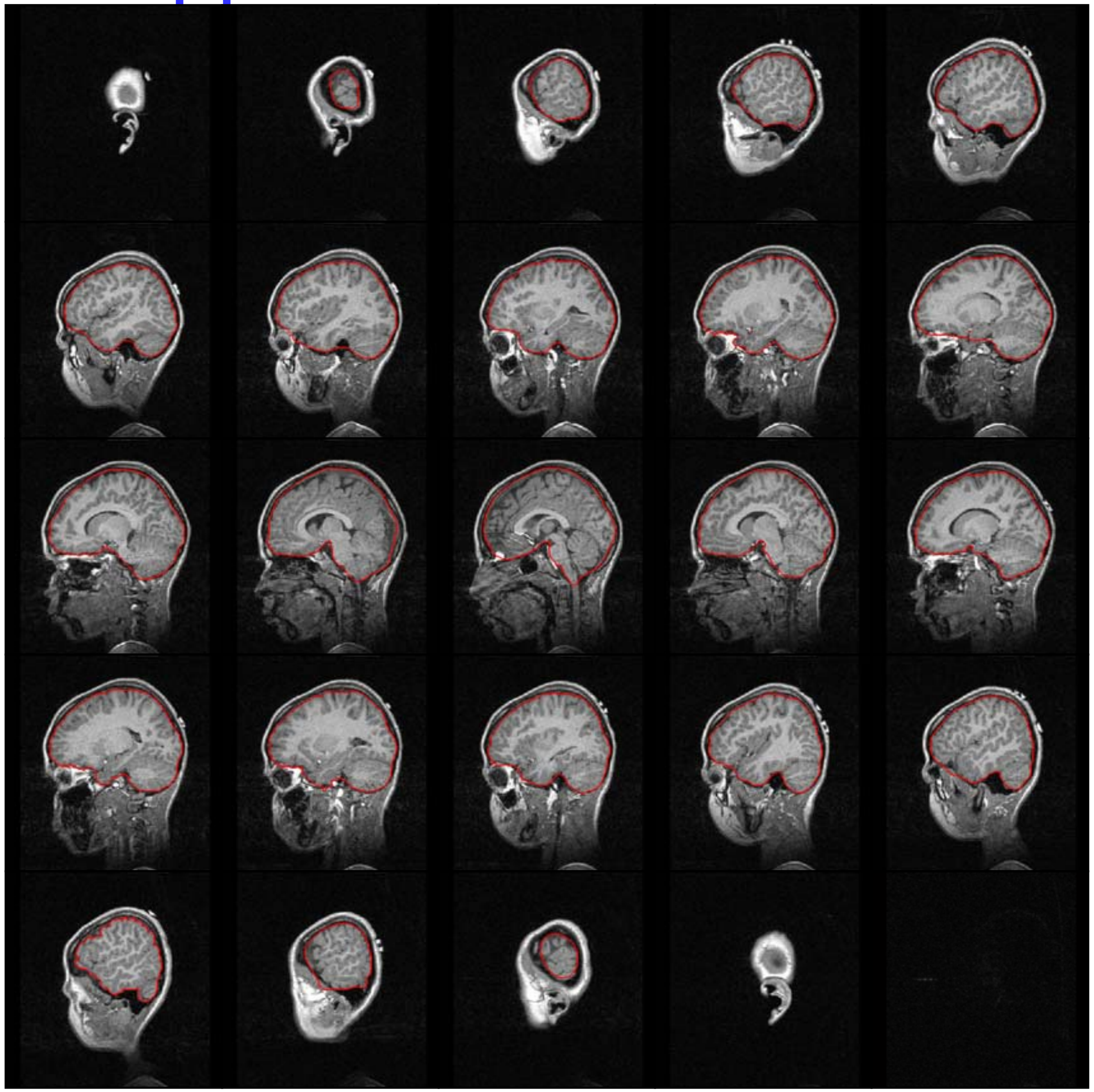
Use of Motion In Segmentation

Take the difference between a reference image and a subsequent image to determine the still elements image components.

a b c

FIGURE 10.50 Building a static reference image. (a) and (b) Two frames in a sequence. (c) Eastbound automobile subtracted from (a) and the background restored from the corresponding area in (b). (Jain and Jain.)

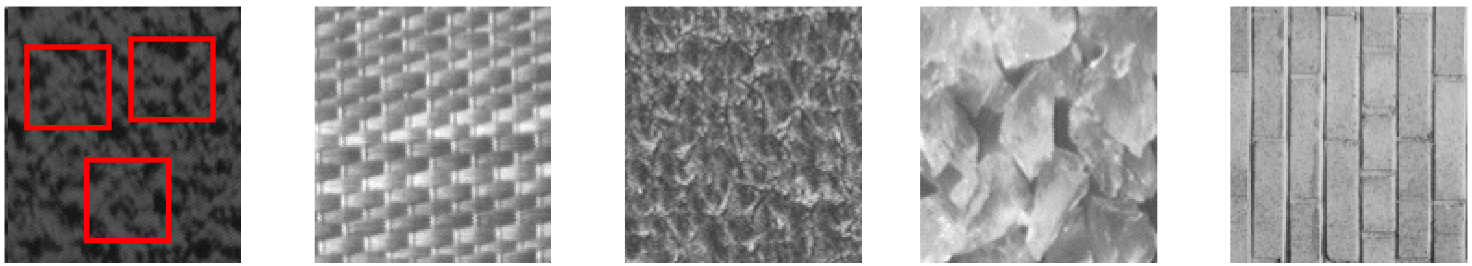
Application to 3D data



Clustering and Classification

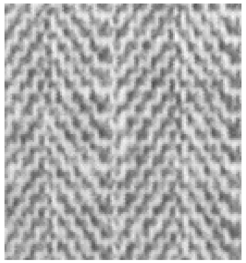
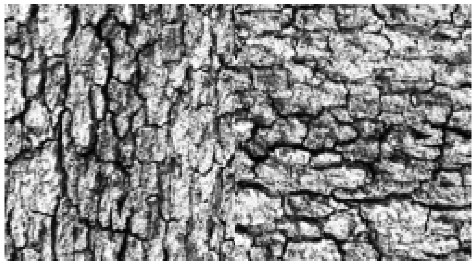
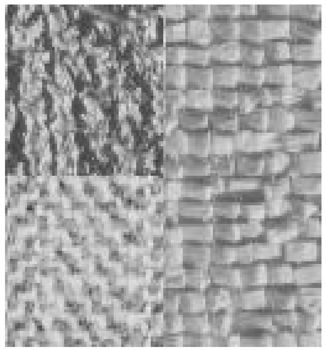
What is texture?

- No agreed reference definition
  - Texture is property of areas
  - Involves spatial distributions of grey levels
  - A region is perceived as a texture if the number of primitives in the field of view is sufficiently high
  - Invariance to translations
  - Macroscopic visual attributes
    - uniformity, roughness, coarseness, regularity, directionality, frequency [Rao-96]
  - Sliding window paradigm



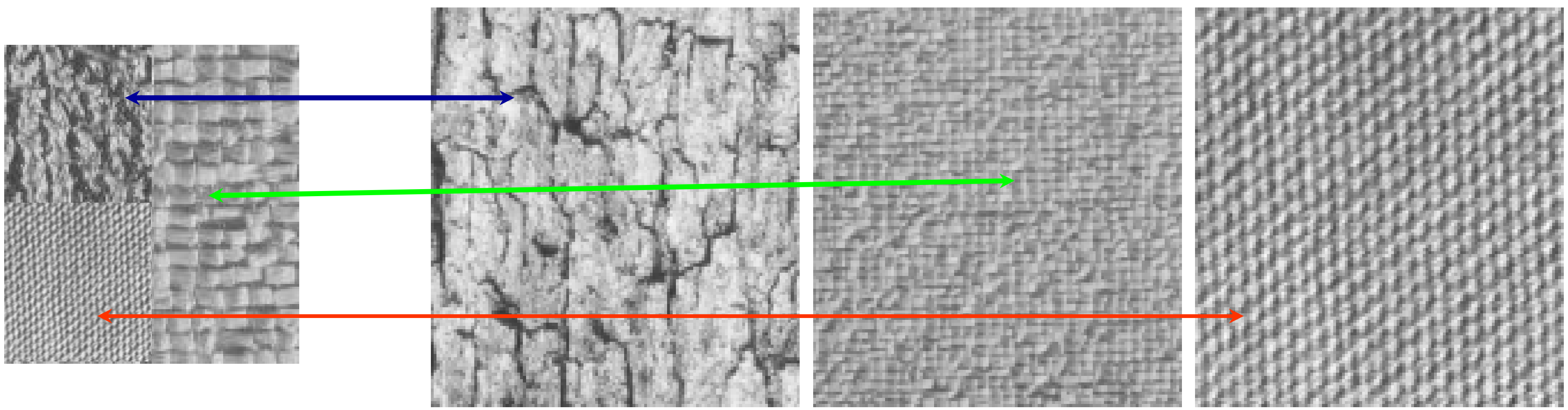
Texture analysis

- Texture segmentation
  - Spatial localization of the different textures that are present in an image
  - Does not imply texture recognition (classification)
  - The textures do not need to be *structurally* different
  - *Apparent* edges
    - Do not correspond to a discontinuity in the luminance function
    - Texture segmentation ↔ Texture segregation
  - *Complex* or *higher-order* texture channels

Texture analysis

- Texture classification (recognition)
  - Hypothesis: textures pertaining to the same class have the same visual appearance → the same *perceptual features*
  - Identification of the class the considered texture belongs to within a given set of classes
  - Implies texture recognition
  - The classification of different textures within a composite image results in a segmentation map



Co-occurrence matrix

- A co-occurrence matrix, also referred to as a co-occurrence distribution, is defined over an image to be the distribution of co-occurring values at a given offset.
- Mathematically, a co-occurrence matrix $C_{k,l}[i,j]$ is defined over an NxM image I, parameterized by an offset (k,l), as:

$$C_{k,l}[i,j] = \sum_{p=1}^{N}\sum_{q=1}^{M} \begin{cases} 1, \text{ if } I(p,q) = i \text{ and } I(p+k, q+l) = j \\ 0, \text{ otherwise} \end{cases}$$

- The co-occurrence matrix depends on (k,l), so we can define as many as we want

Texture Classification

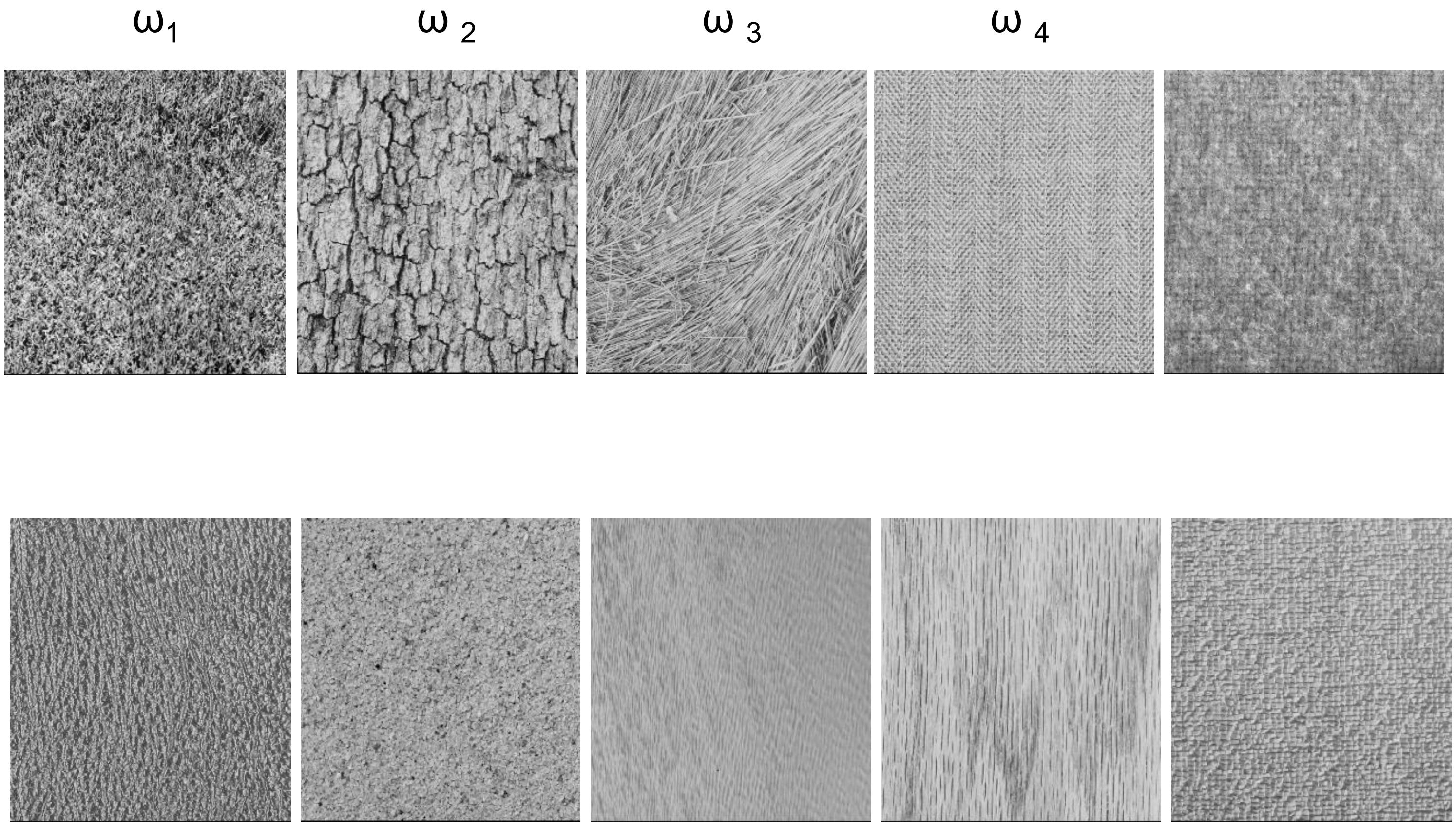
- Problem statement
  - Given a set of classes $\{\omega_i,\ i=1,\ldots N\}$ and a set of observations $\{\mathbf{x}_{i,k},k=1,\ldots M\}$ determine the most probable class, given the observations. This is the class that maximizes the conditional probability:

$$\omega_{winner} = \max_k P(\omega_i | x_k)$$

Texture classification

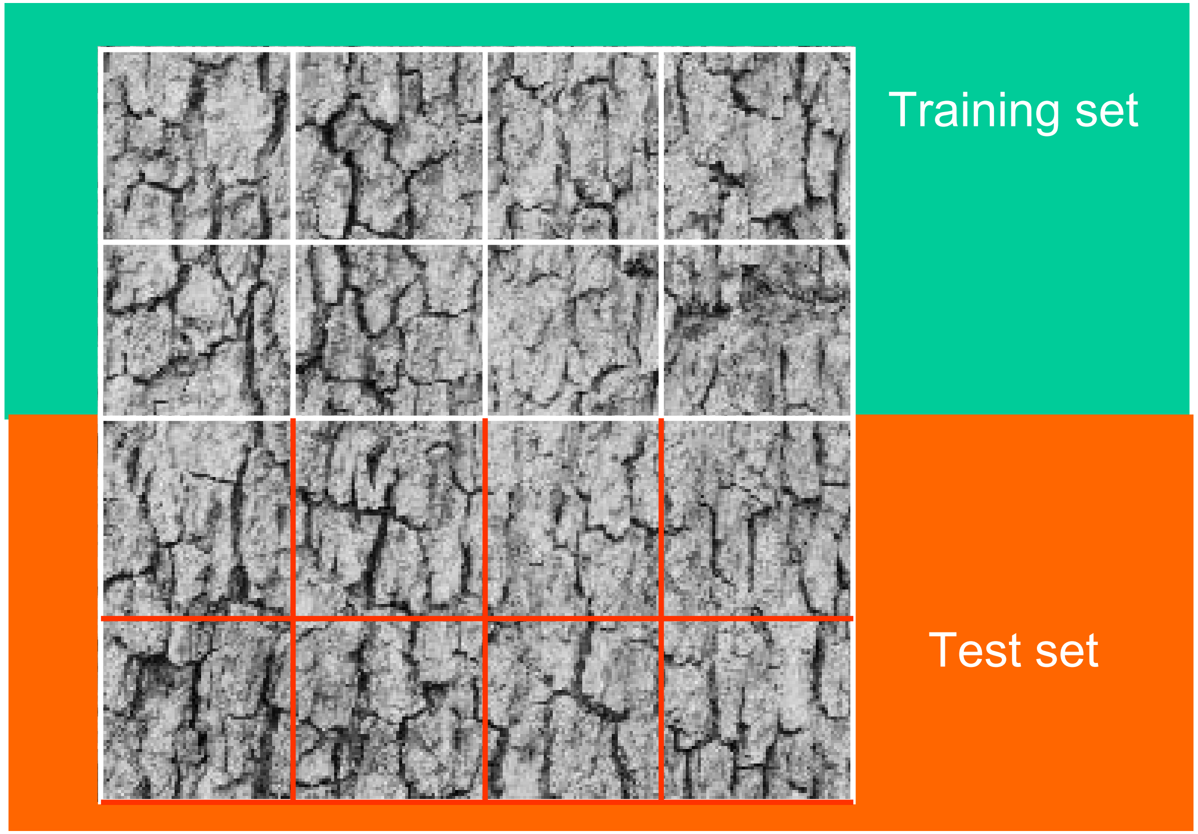
- Method
  - Describe the texture by some *features* which are related to its *appearance*
    - Texture → class → ω_k
    - Subband statistics → Feature Vectors (FV) → $\boldsymbol{x}_{i,k}$
  - Define a distance measure for FV
    - Should reflect the *perceived similarity/dissimilarity* among textures (**unsolved**)
  - Choose a *classification rule*
    - Recipe for comparing FV and choose 'the winner class'
  - Assign the considered texture sample to the class which is the *closest* in the feature space

Exemple: texture classes



FV extraction

- Step 1: create independent texture instances



Feature extraction

- Step 2: extract features to form *feature vectors*

Intensity image

↓

DWT/DWF

↓

For each sub-image

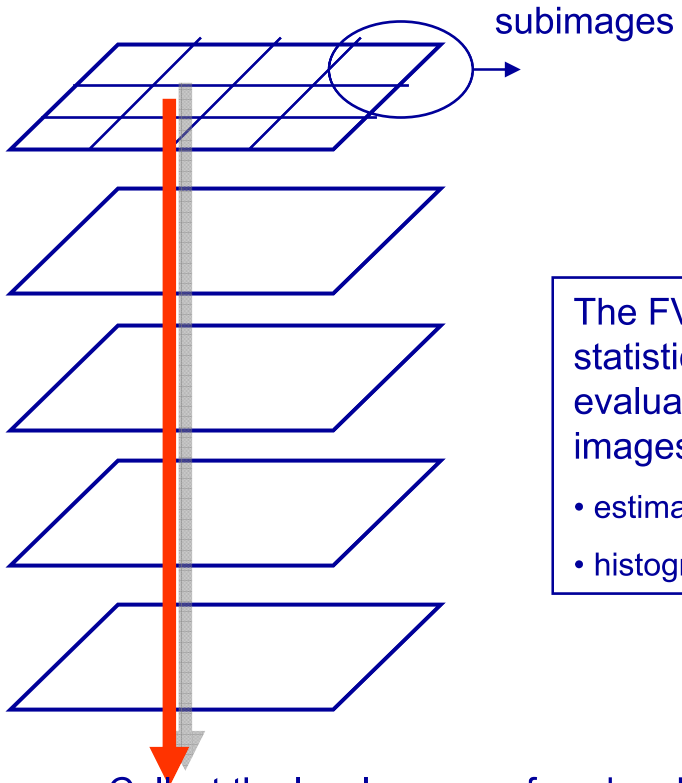
For each subband

Calculate the local energy (variance)

Fill the corresponding position in the FV

↓

One FV for each sub-image ⇒ Classification algorithm

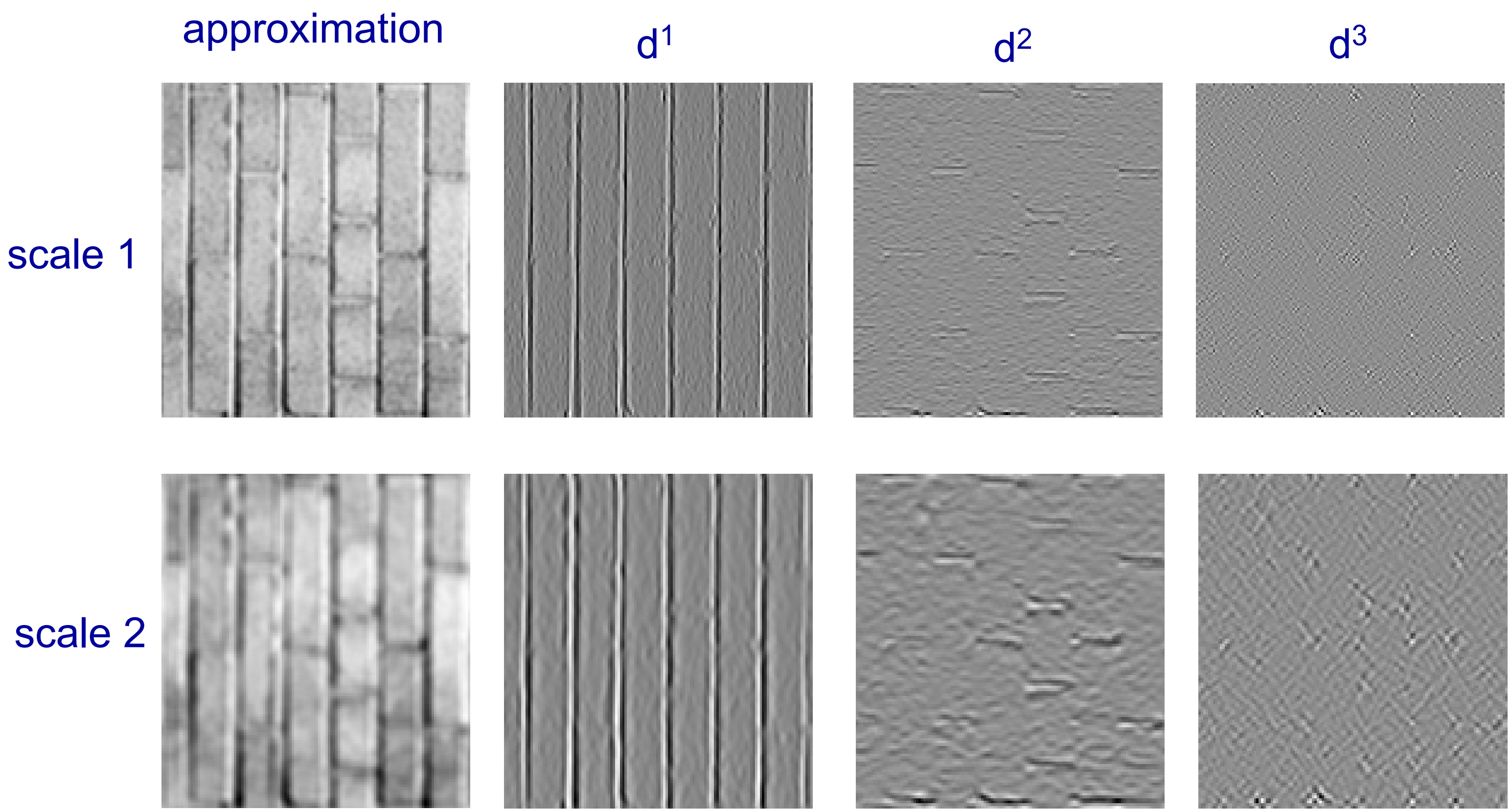


⇒ Collect the local energy of each sub-image in the different subbands in a vector

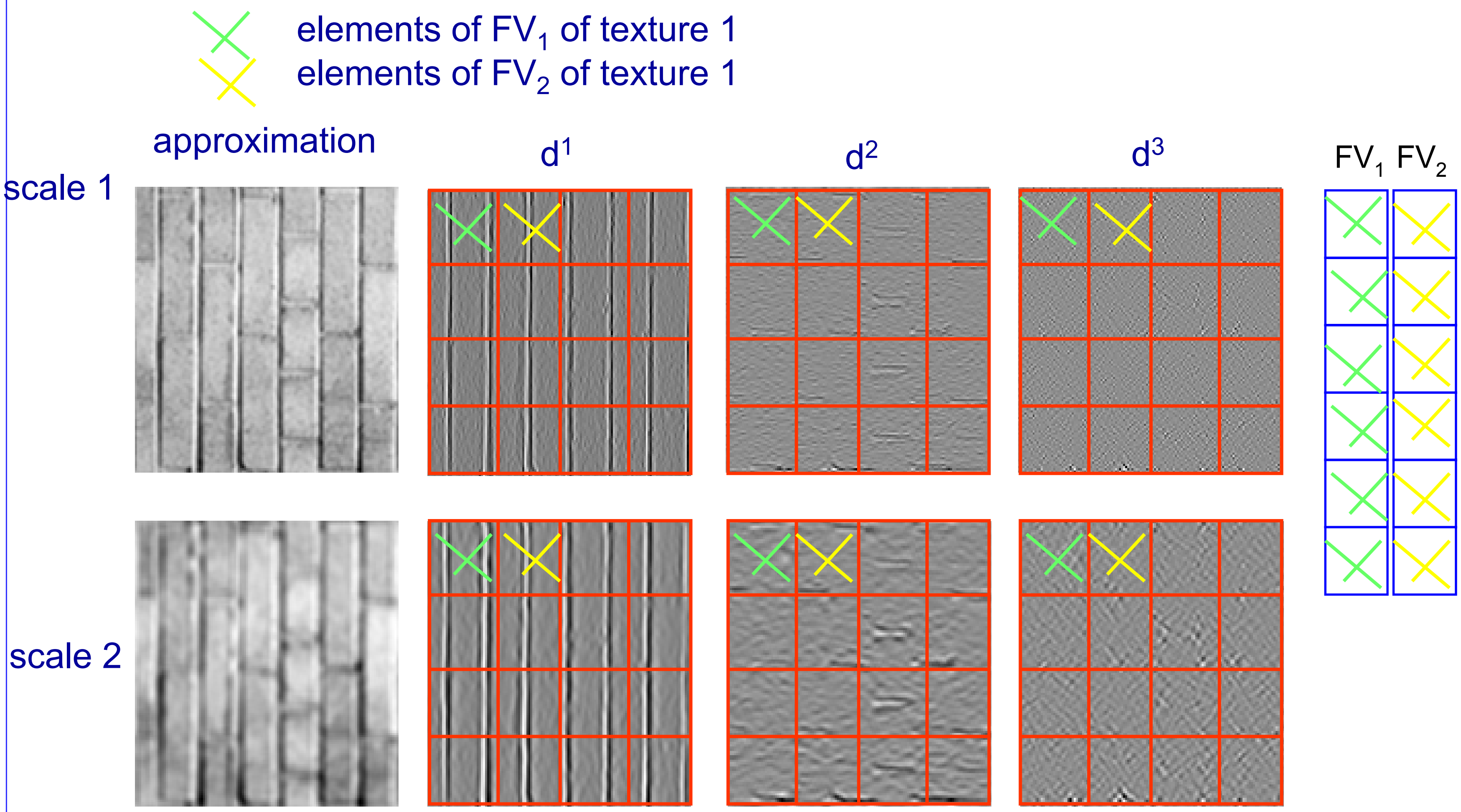
The FVs contain some statistical parameters evaluated on the subband images

- estimates of local variances
- histograms

Building the FV

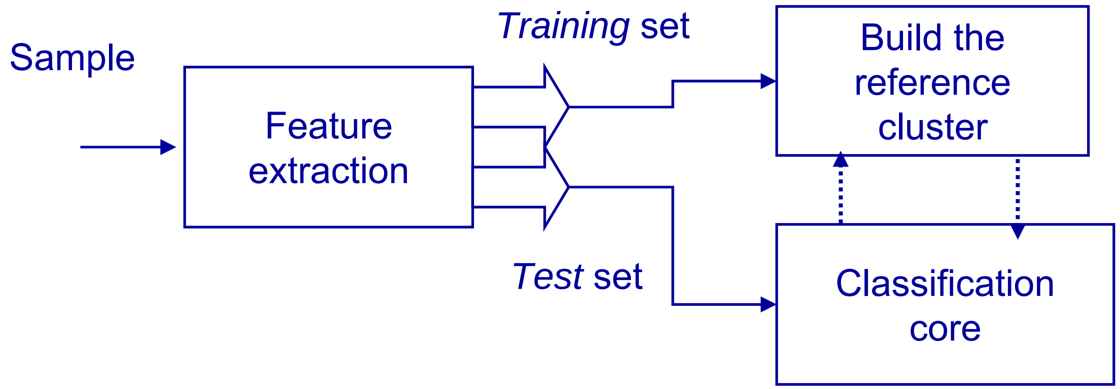


Building the FV



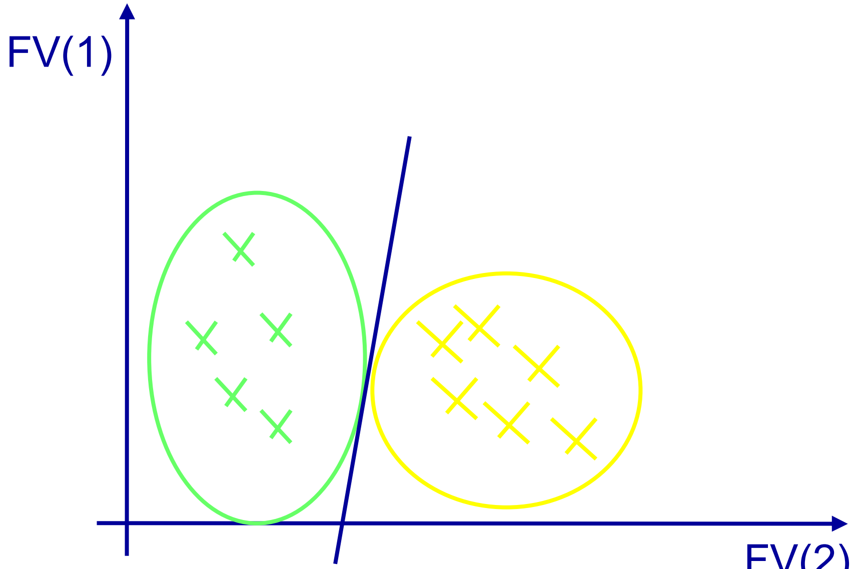
Implementation

- Step 1: Training
  - The classification algorithm is provided with many examples of each texture class in order to build clusters in the feature space which are representative of each class
    - Examples are sets of FV for each texture class
    - Clusters are formed by aggregating vectors according to their “distance”
- Step 2: Test
  - The algorithm is fed with an example of texture ω_i (vector $\boldsymbol{x}_{i,k}$) and determines which class it belongs as the one which is “closest”

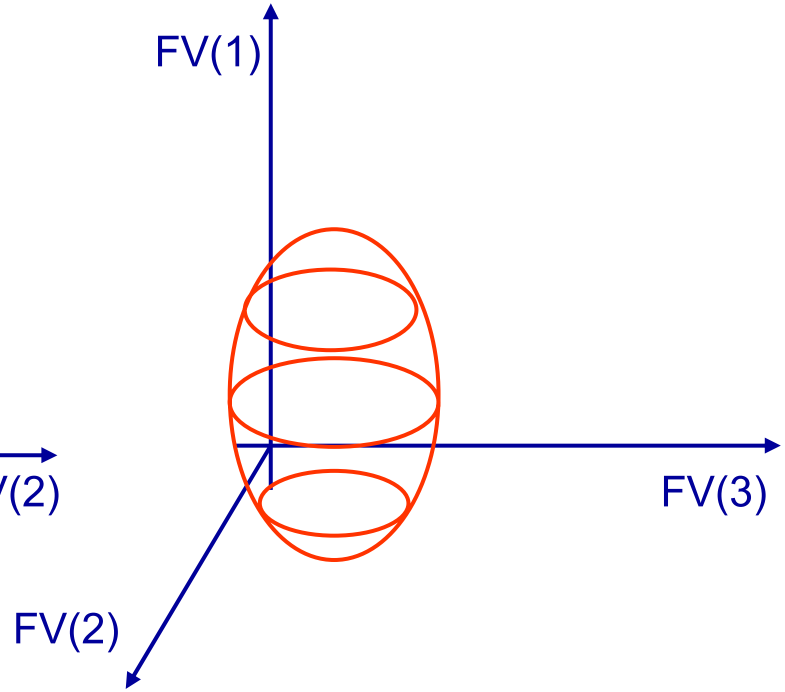


Clustering in the Feature Space

Bi-dimensional feature space (FV of size 2)



Multi-dimensional feature space



FV classification: identification of the cluster which best represents the vector according to the chosen distance measure

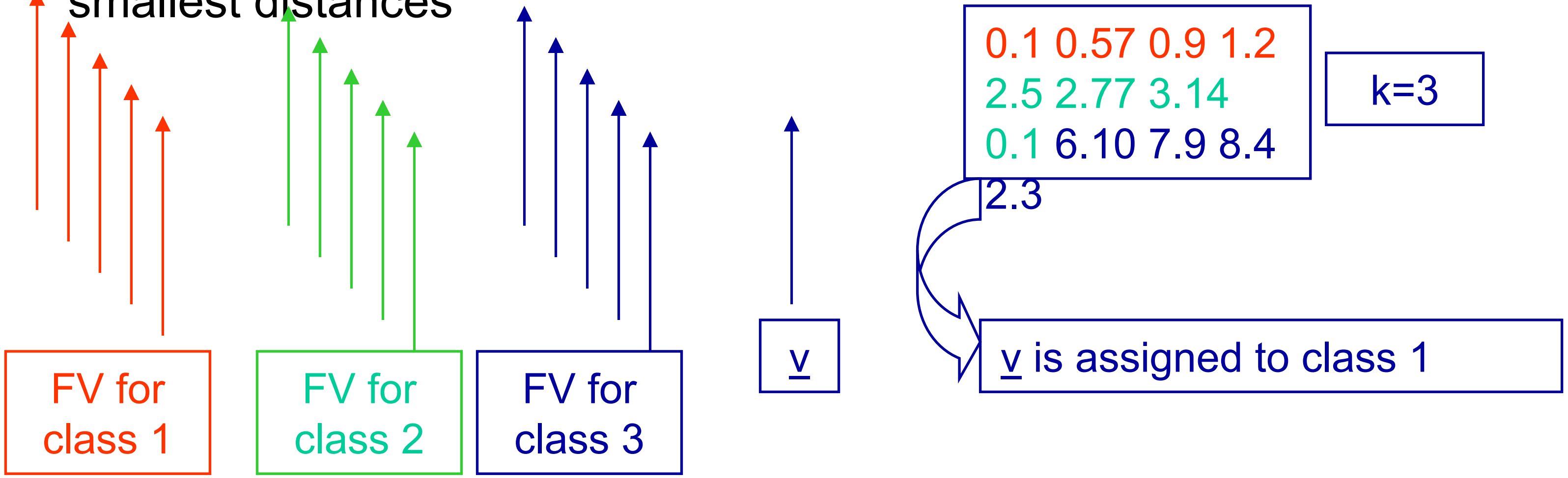
Classification algorithms

- Measuring the distance among a *class* and a *vector*
  - Each class (set of vectors) is represented by the mean (m) vector and the vector of the variances (s) of its components ⇒ the training set is used to build m and s
  - The distance is taken between the test vector and the m vector of each class
  - The test vector is assigned to the class to which it is closest
    - Euclidean classifier
    - Weighted Euclidean classifier

- Measuring the distance among *every couple* of vectors
  - kNN classifier

kNN classifier

- Given a vector v of the test set
  - Take the distance between the vector v and ALL the vectors of the training set
  - (while calculating) keep the k smallest distances and keep track of the class they correspond to
  - Assign v to the class which is most represented in the set of the k smallest distances



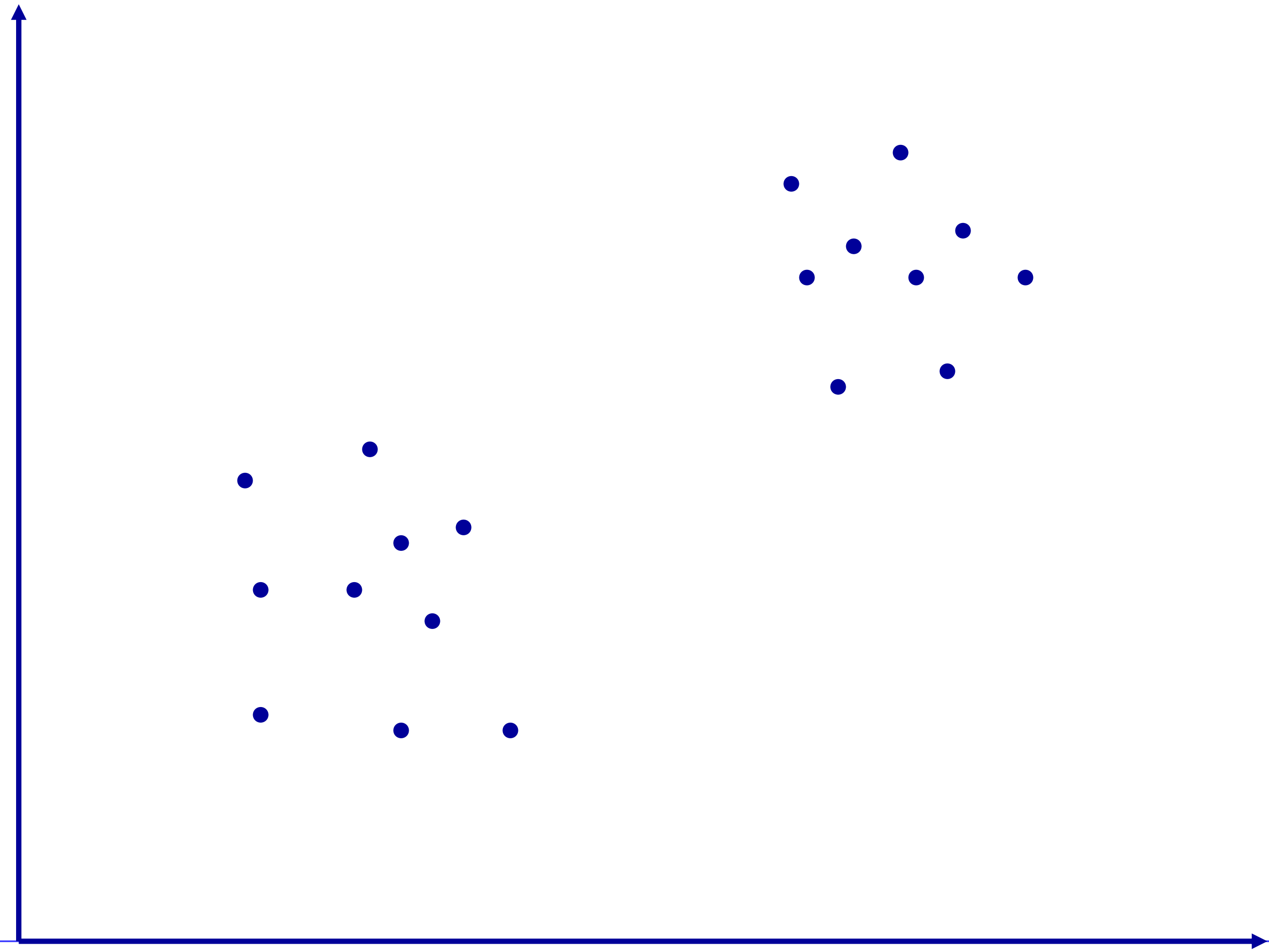
Confusion matrix

textures	1	2	3	4	5	6	7	8	9	10	% correct
1	841	0	0	0	0	0	0	0	0	0	100.00%
2	0	840	1	0	0	0	0	0	0	0	99.88%
3	2	0	839	0	0	0	0	0	0	0	99.76%
4	0	0	0	841	0	0	0	0	0	0	100.00%
5	0	0	88	0	753	0	0	0	0	0	89.54%
6	0	0	134	0	0	707	0	0	0	0	84.07%
7	0	66	284	0	0	0	491	0	0	0	58.38%
8	0	0	58	0	0	0	0	783	0	0	93.10%
9	0	0	71	0	0	0	0	0	770	0	91.56%
10	0	4	4	0	0	0	0	0	0	833	99.05%
				Average recognition rate							91.53%

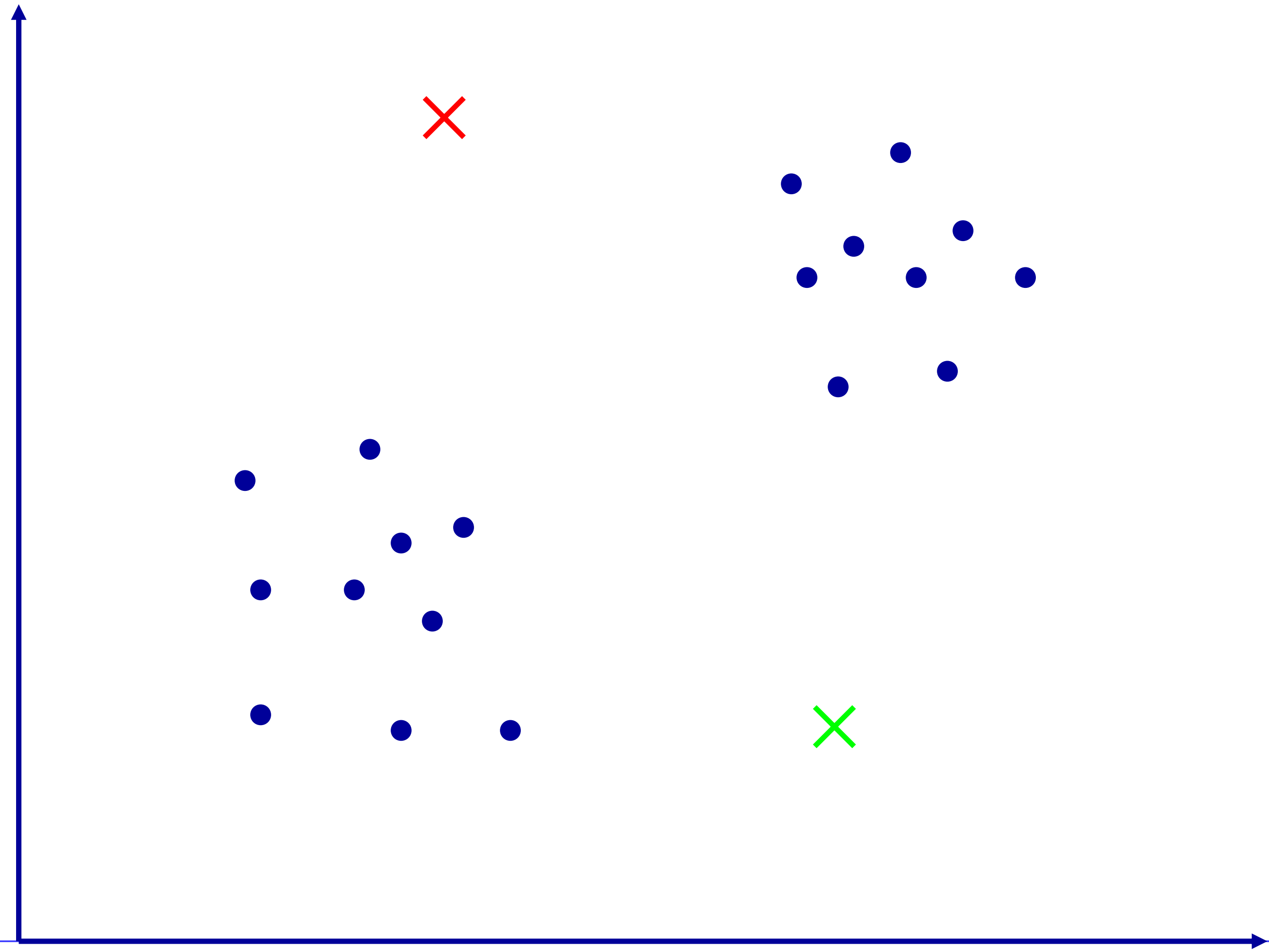
K-Means Clustering

1. Partition the data points into K clusters randomly. Find the centroids of each cluster.
2. For each data point:
   - Calculate the distance from the data point to each cluster.
   - Assign the data point to the closest cluster.
3. Recompute the centroid of each cluster.
4. Repeat steps 2 and 3 until there is no further change in the assignment of data points (or in the centroids).

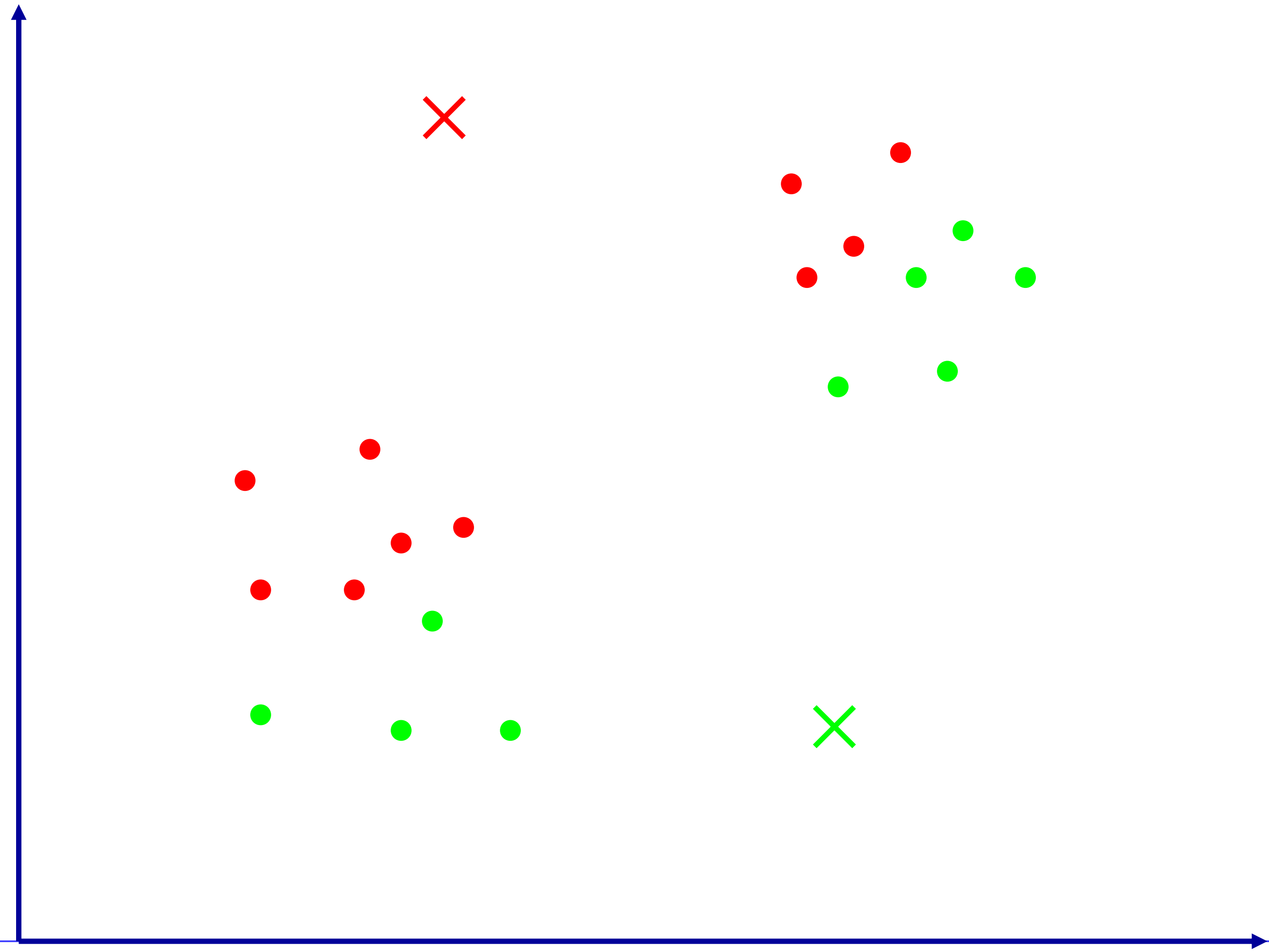
K-Means Clustering



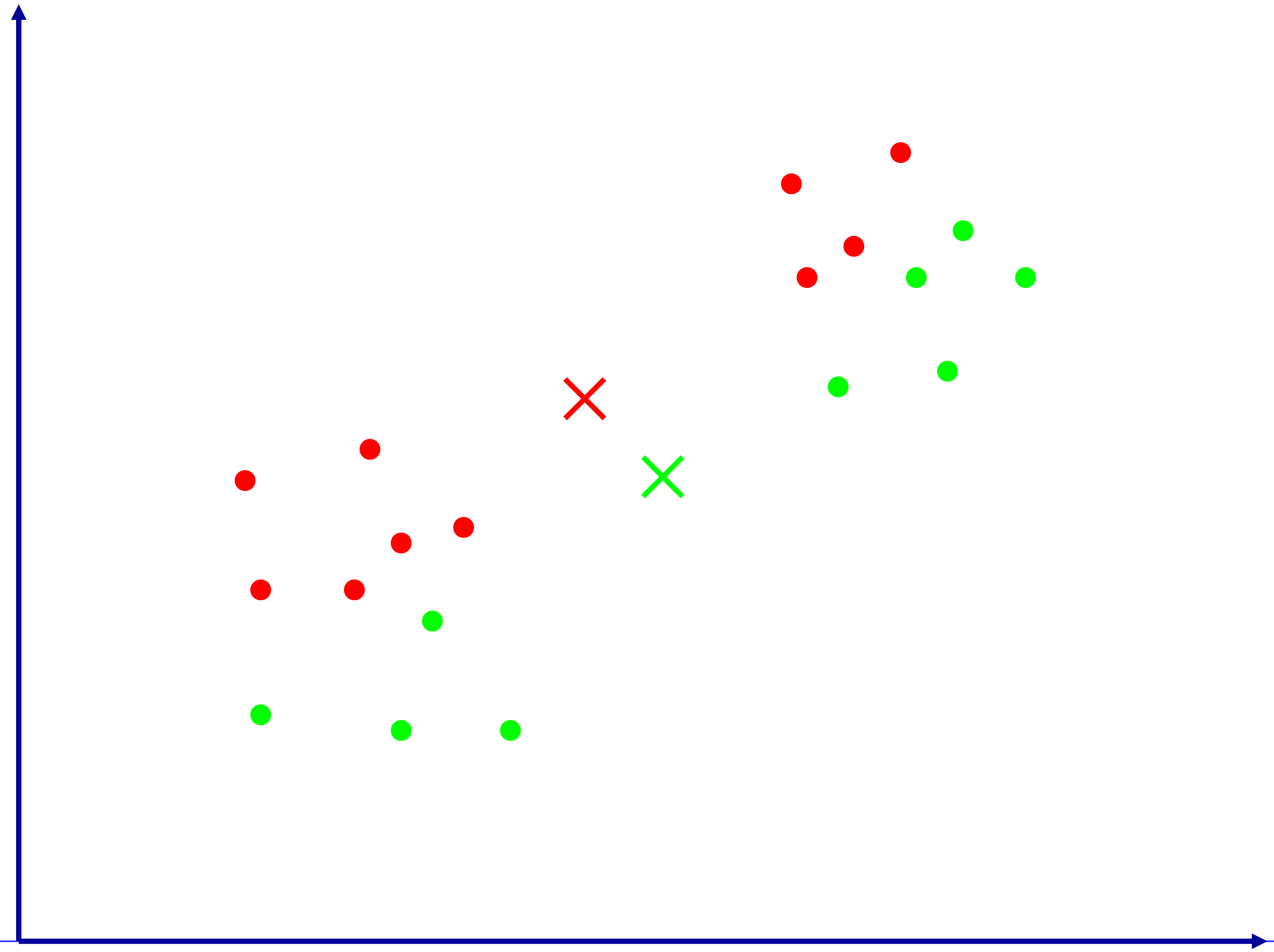
K-Means Clustering



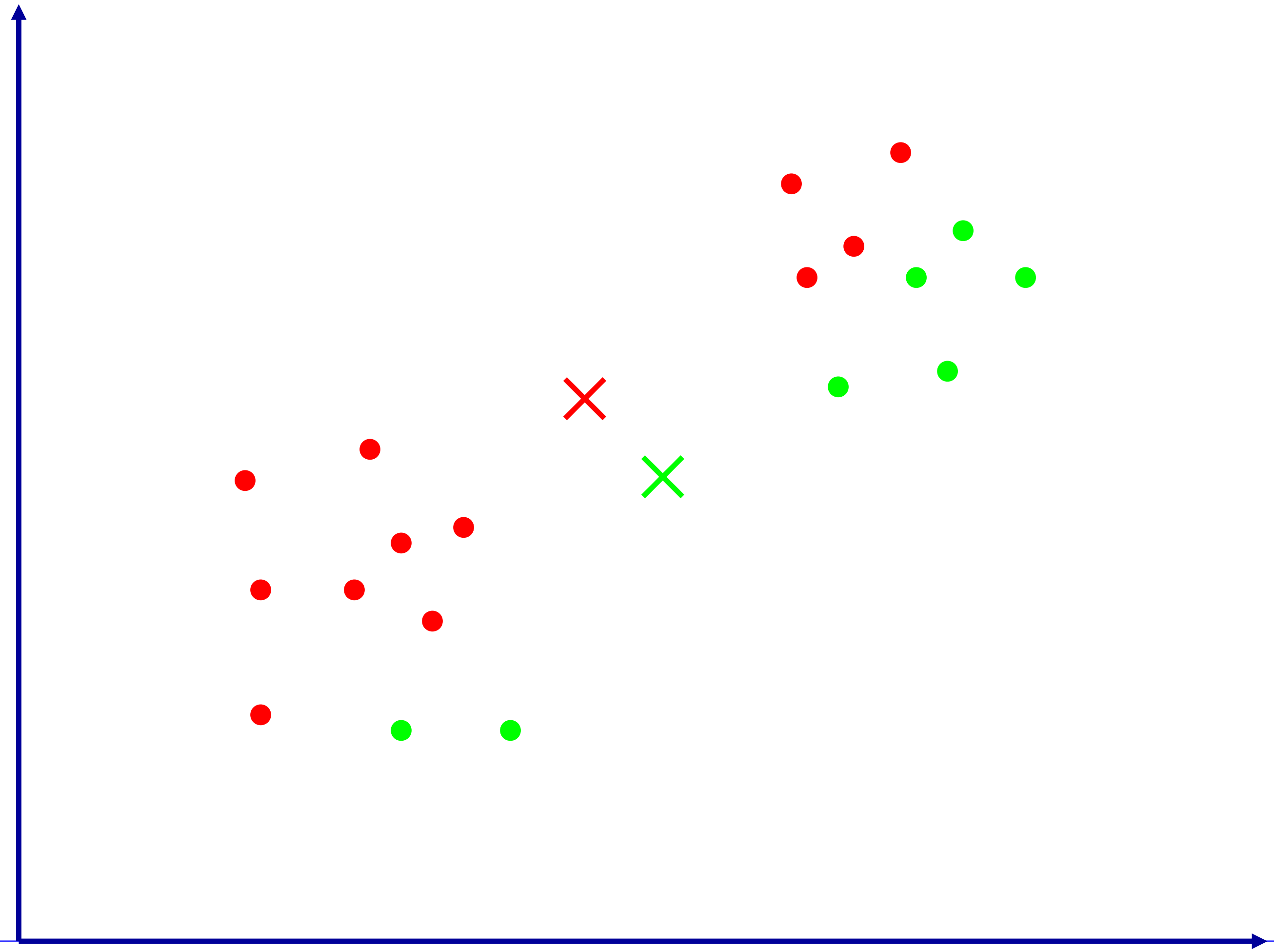
K-Means Clustering



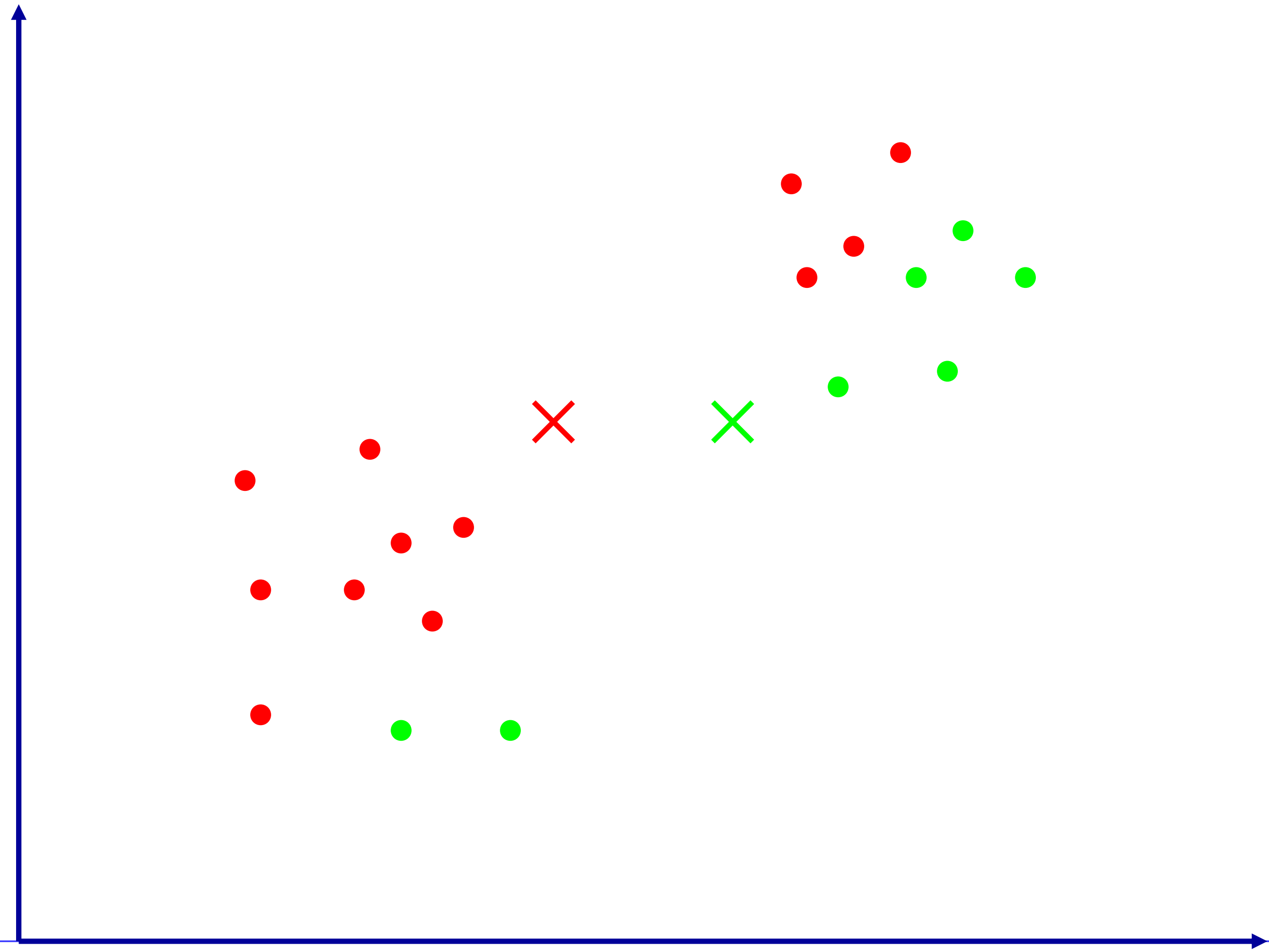
K-Means Clustering



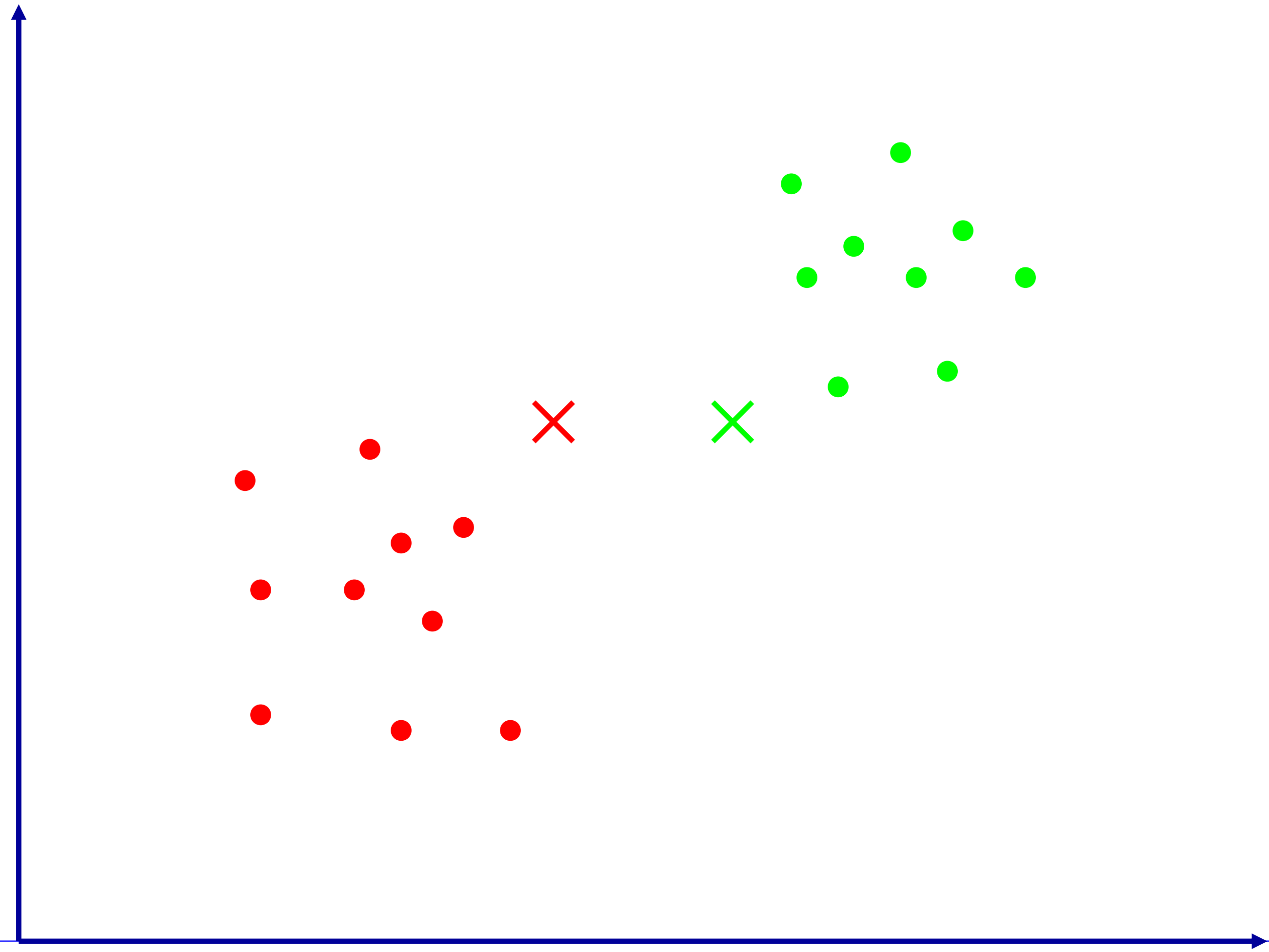
K-Means Clustering



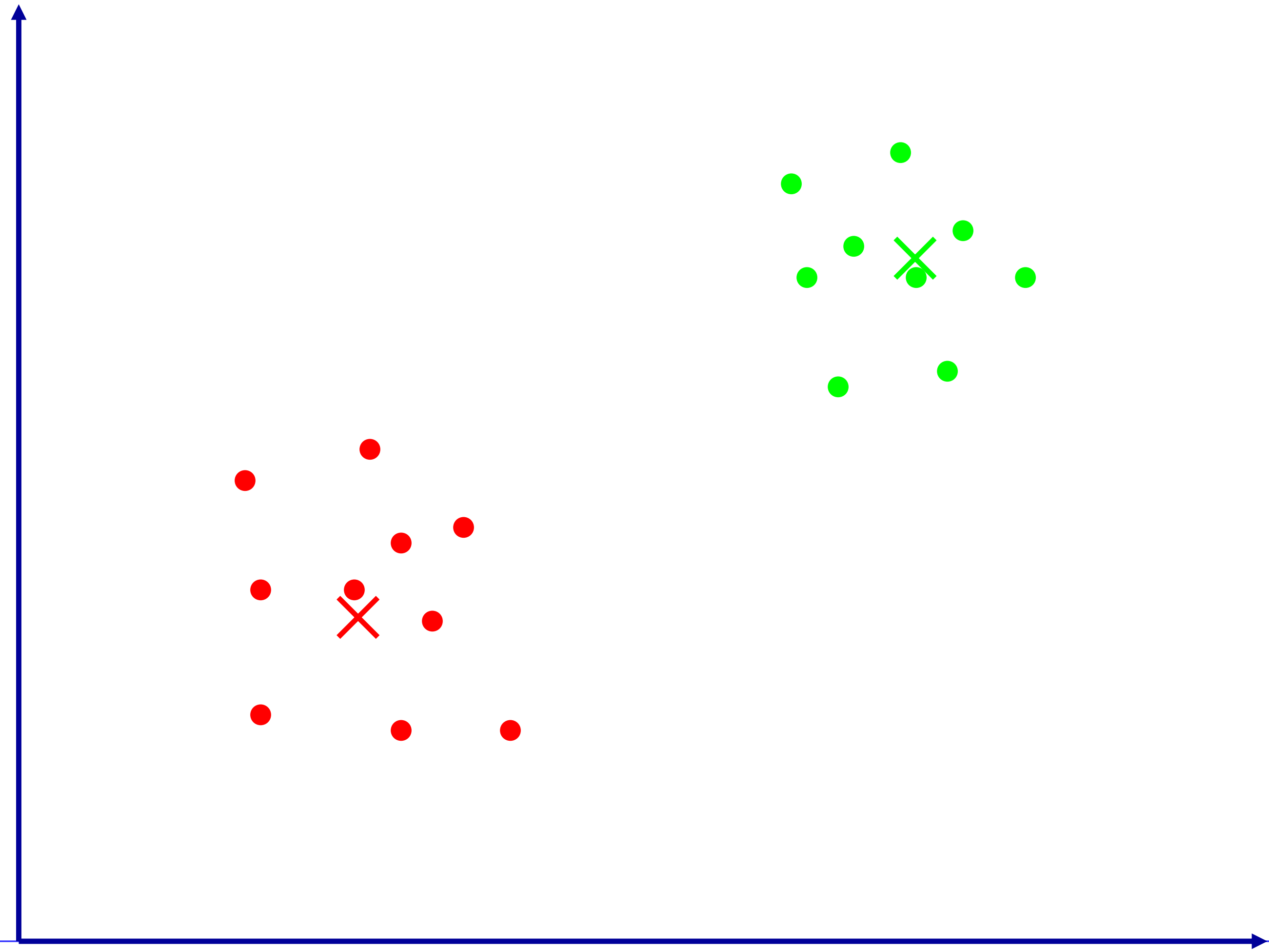
K-Means Clustering



K-Means Clustering

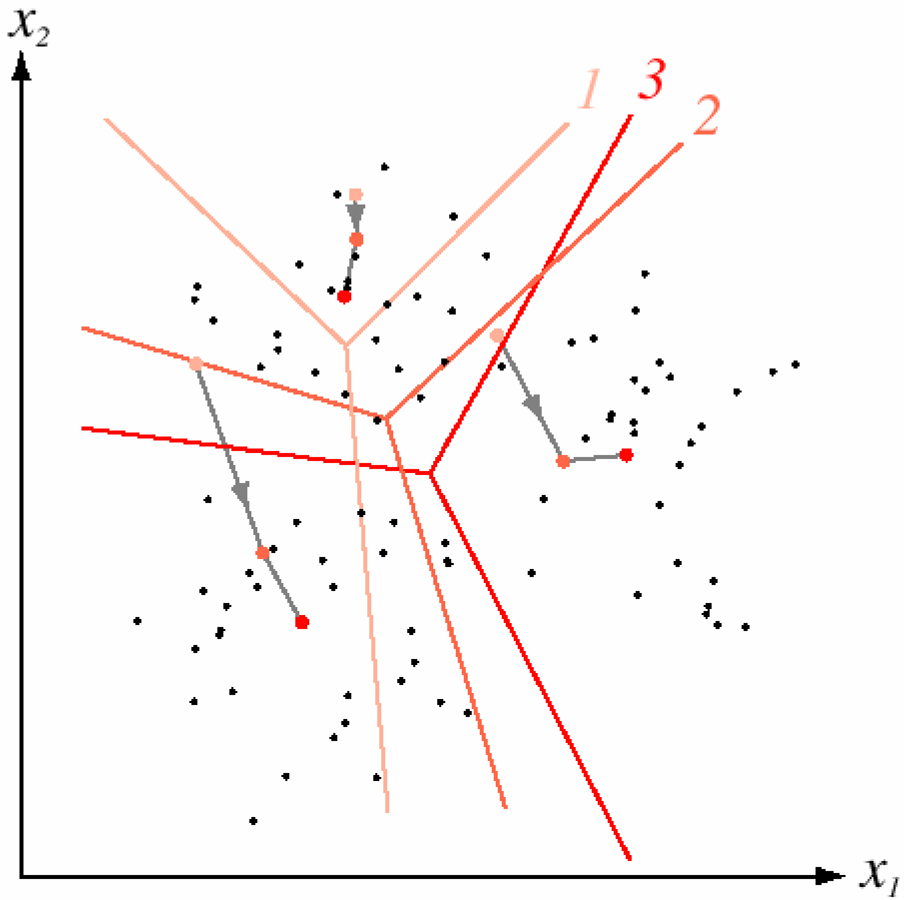


K-Means Clustering



K-Means Clustering

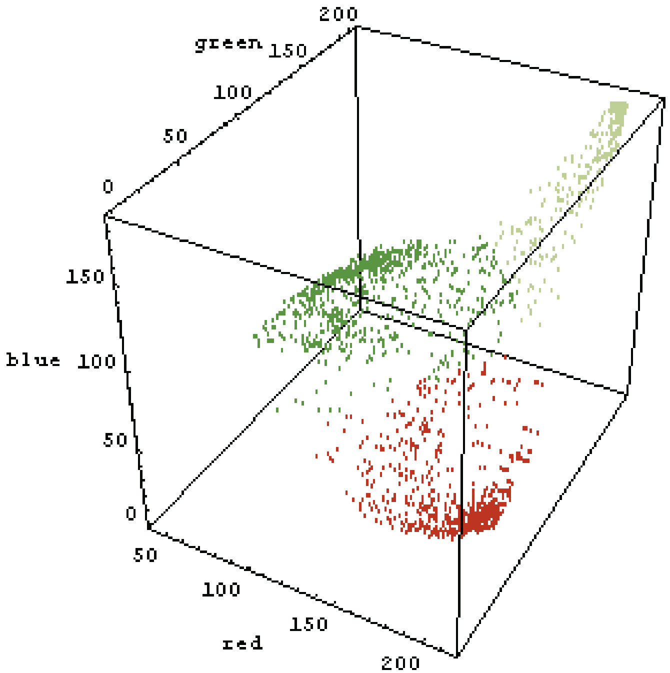
- Example



Duda et al.

K-Means Clustering

- RGB vector

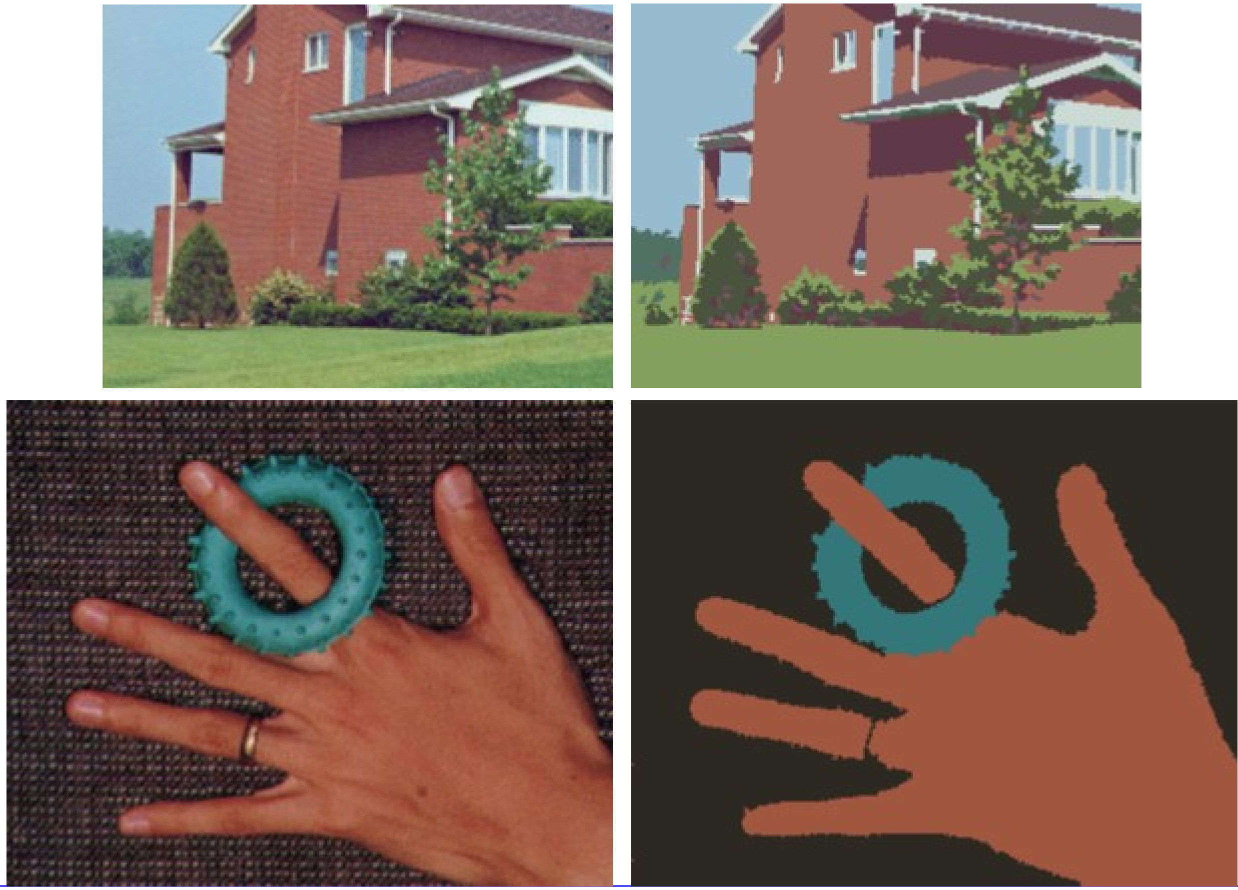


K-means clustering minimizes

$$\sum_{i \in \text{clusters}} \left\{ \sum_{j \in \text{elements of i'th cluster}} \left\| x_j - \mu_i \right\|^2 \right\}$$

Clustering

- Example



D. Comaniciu and P. Meer, *Robust Analysis of Feature Spaces: Color Image Segmentation*, 1997.

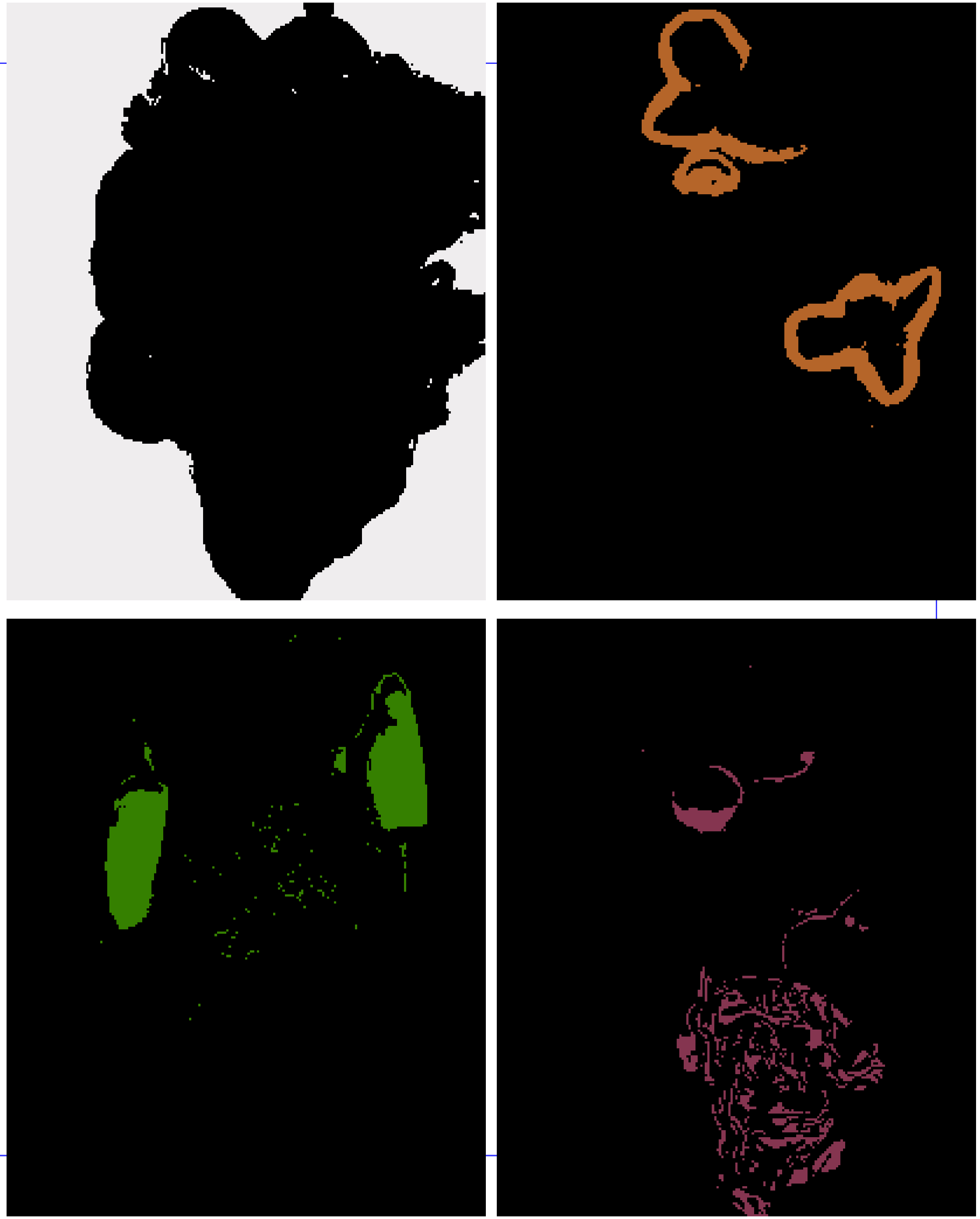
K-Means Clustering

- Example

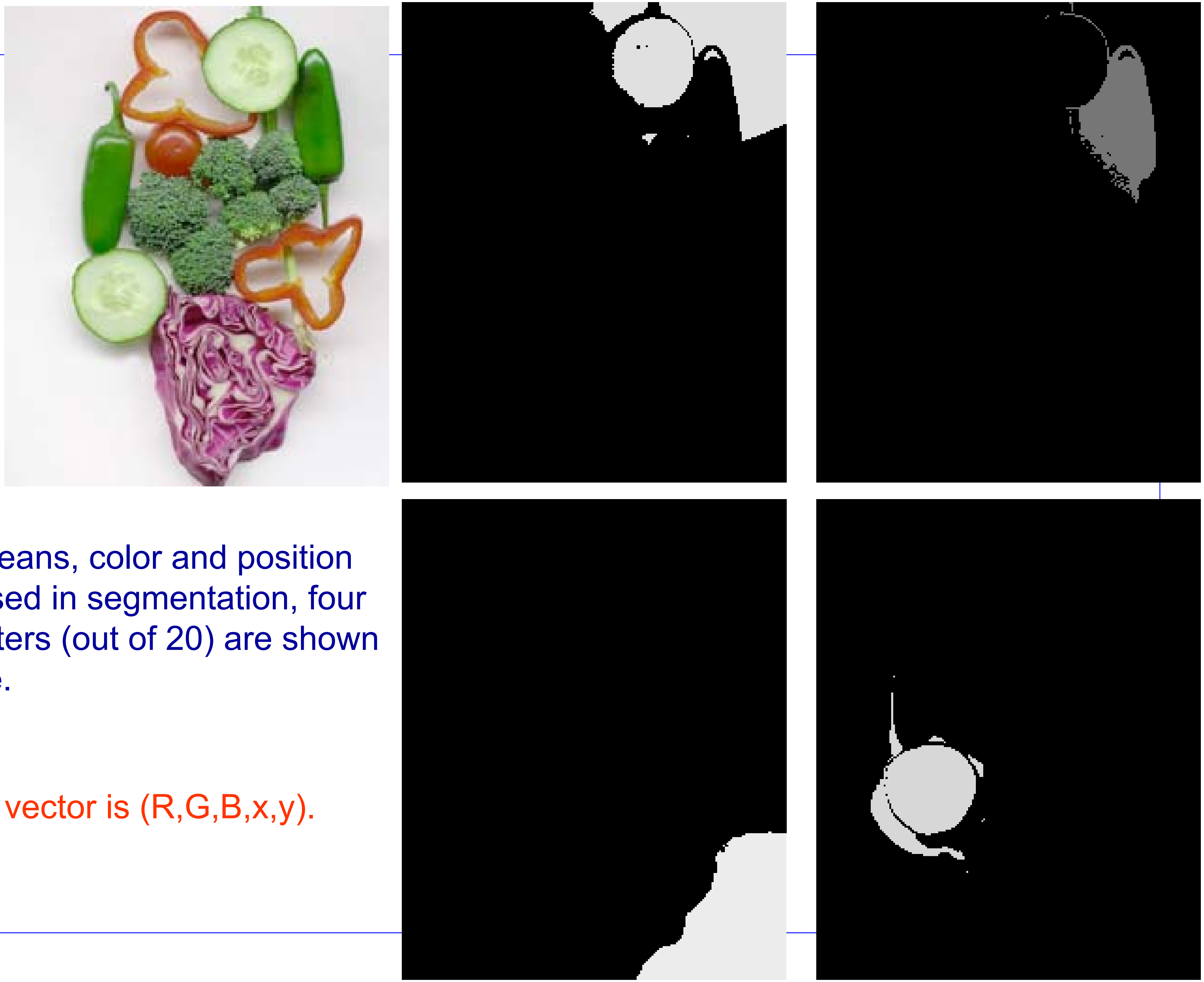
Original

K=5

K=11



K-means, only color is used in segmentation, four clusters (out of 20) are shown here.



K-means, color and position is used in segmentation, four clusters (out of 20) are shown here.

Each vector is (R,G,B,x,y).

K-Means Clustering: Axis Scaling

- Features of different types may have different scales.
  - For example, pixel coordinates on a 100x100 image vs. RGB color values in the range [0,1].
- Problem: Features with larger scales dominate clustering.
- Solution: Scale the features.