Region-based processing

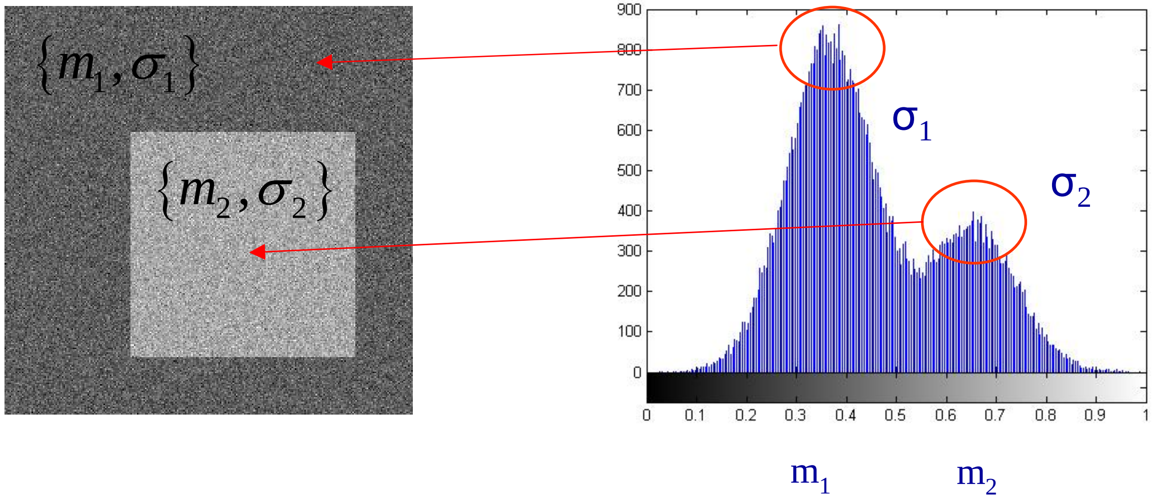
Feature extraction

Topics

- Feature extraction
- Image segmentation
- Clustering and classification

Region based processing

- Complementary to edge detection
- Based on neighborhood characteristics
- Local descriptors represent properties of sets of pixels. Typically these are representative of the pdf (histogram) of the gray values in each region



Applications

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

Segmentation strategy

Edge-based

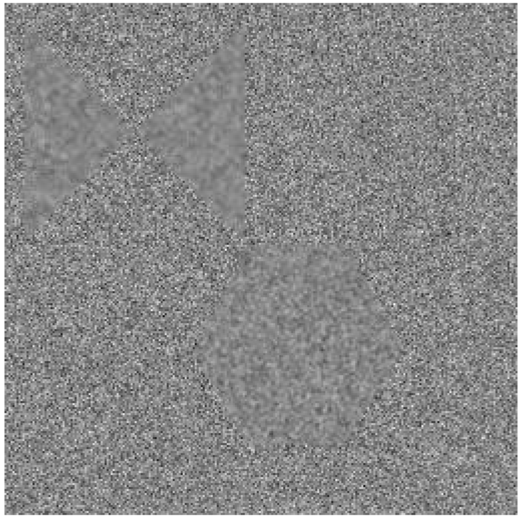
- Assumption: different objects are separated by edges (grey level discontinuities)
- The segmentation is performed by identifying the grey level gradients
- The same approach can be extended to color channels

Region-based

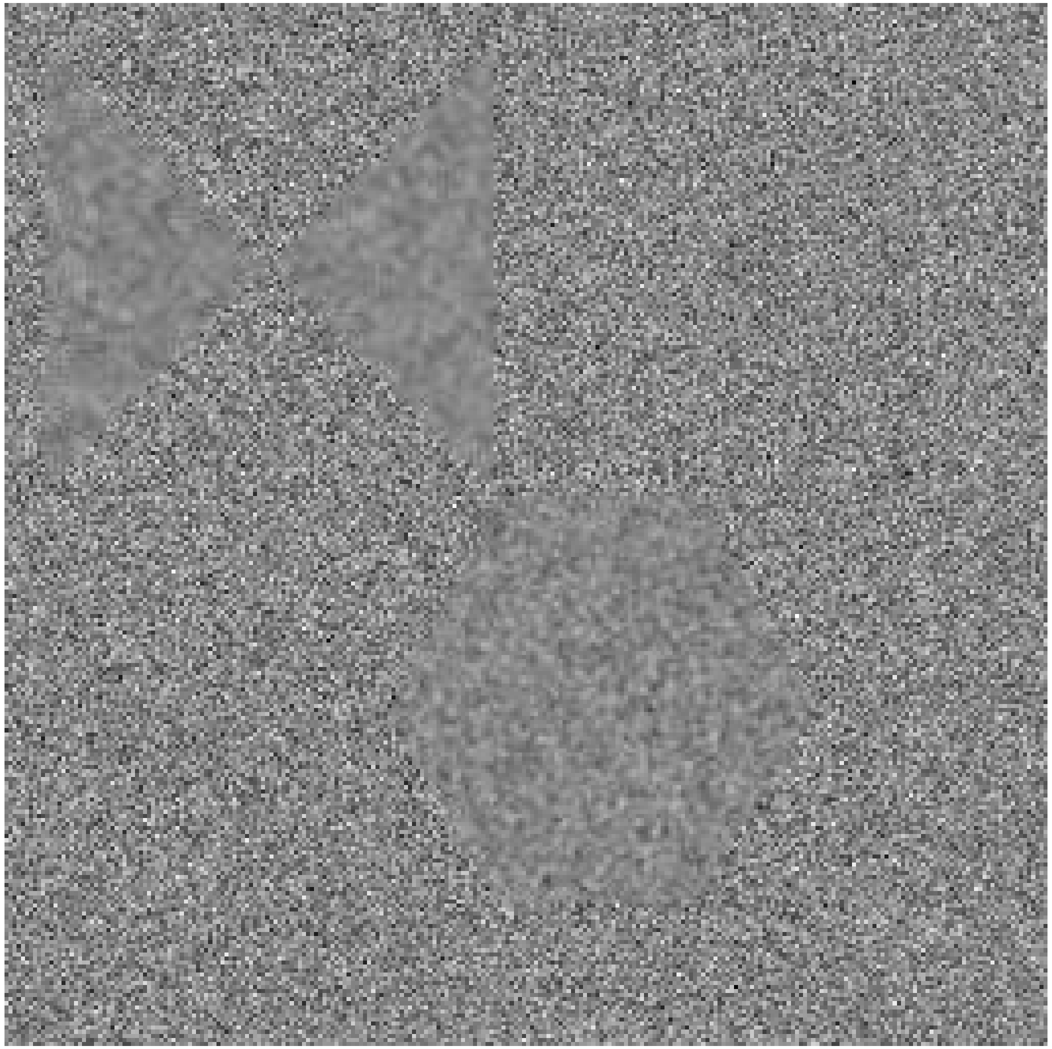
- Assumption: different objects are separated by other kind of perceptual boundaries
  - neighborhood features
- Most often texture-based
  - Textures are considered as instantiations of underlying stochastic processes and analyzed under the assumptions that stationarity and ergodicity hold
- Method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

Examples

original

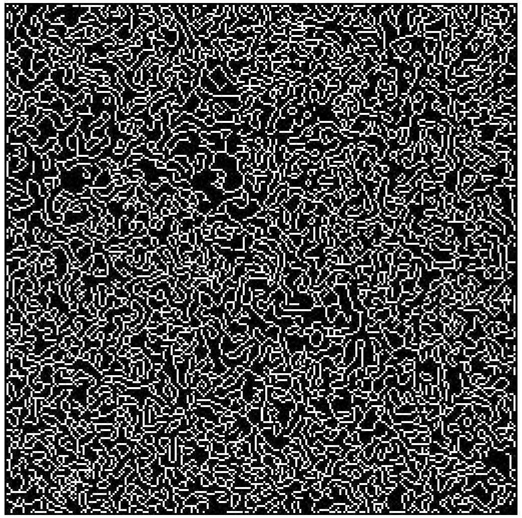


zoomed

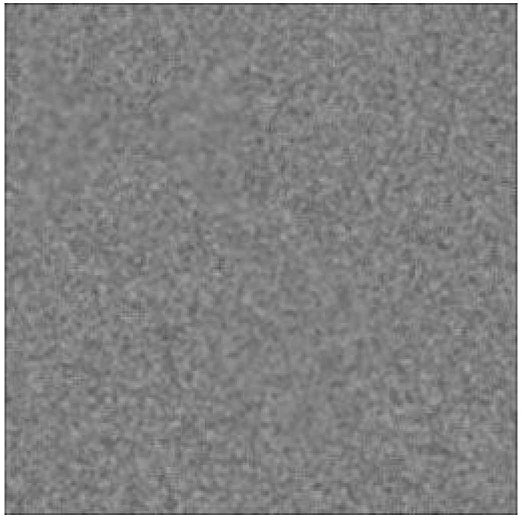


Examples

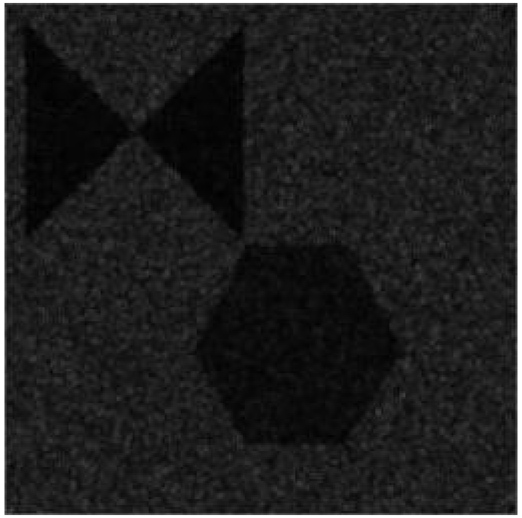
Canny



block mean



block std



Feature extraction

Types of features

- An *image feature* is a distinguishing primitive characteristic or attribute of an image.
- Amplitude features: image domain
- Transformed coefficients features: transformed domain
  - Fourier domain
  - Wavelet domain
  - Principal components (PCA)

Amplitude features

- Image variables such as luminance or tristimulus values may be utilized directly, or alternatively, some linear, nonlinear, or perhaps noninvertible transformation can be performed to generate variables in a new amplitude space.
- Amplitude measurements may be made at specific image points f[i,j], [e.g., the amplitude at pixel coordinate] , or over a neighborhood centered at [i,j].
  - An advantage of a neighborhood, as opposed to a point measurement, is a *diminishing of noise effects* because of the averaging process.
  - A disadvantage is that object edges falling within the neighborhood can lead to erroneous measurements.

Amplitude features

- Mean over a window W=2w+1

$$M(j,k) = \frac{1}{W^2} \sum_{m=-w}^{w} \sum_{n=-w}^{w} F(j+m, k+n)$$

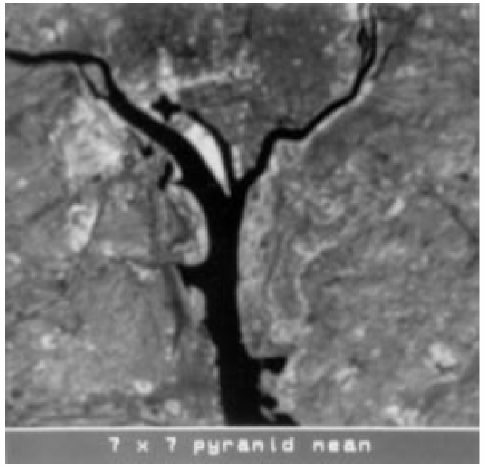
- Median over a window W=2w+1
  - The *median* is defined to be that pixel amplitude in the window for which one-half of the pixels are equal or smaller in amplitude, and one-half are equal or greater in amplitude.

- Variance over a window W=2w+1

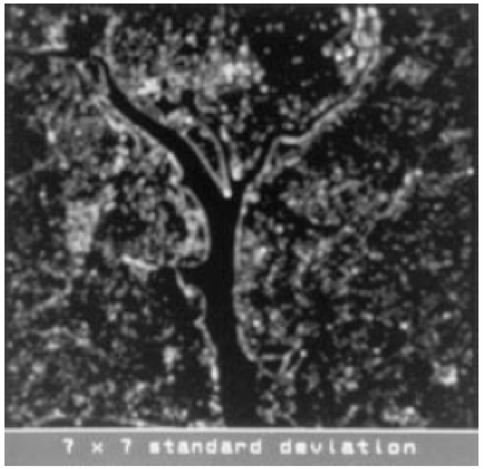
$$S(j,k) = \frac{1}{W} \left[\sum_{m=-w}^{w} \sum_{n=-w}^{w} [F(j+m, k+n) - M(j+m, k+n)]^2 \right]^{1/2}$$

Example

(*a*) Original



(*b*) 7 × 7 pyramid mean



(*c*) 7 × 7 standard deviation

(*d*) 7 × 7 plus median

FIGURE 16.2-1. Image amplitude features of the `washington_ir` image.

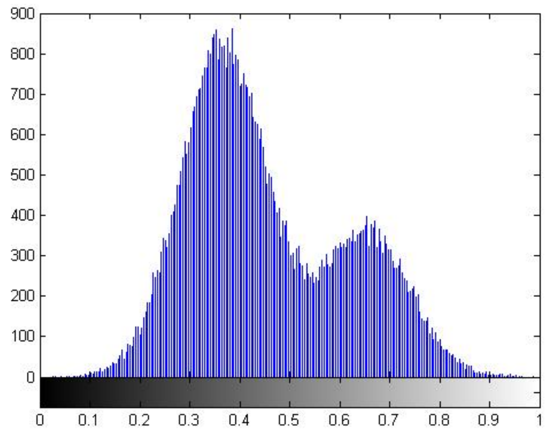
Histogram features

- Mean and standard deviation can be calculated based on the histogram, as well as other features representing the distribution of gray level (or tristimulus) values
- First order histogram

$$h[g] = \frac{N(g)}{N}$$

$N(g)$ → number of pixels with graylevel=g

N → total number of pixels



Quantitative histogram shape descriptors

- First order descriptors

Mean:

$$S_M \equiv \bar{b} = \sum_{b=0}^{L-1} b P(b)$$

Standard deviation:

$$S_D \equiv \sigma_b = \left[\sum_{b=0}^{L-1} (b - \bar{b})^2 P(b) \right]^{1/2}$$

Skewness:

$$S_S = \frac{1}{\sigma_b^3} \sum_{b=0}^{L-1} (b - \bar{b})^3 P(b)$$

Quantitative histogram shape descriptors

- First order descriptors

Kurtosis:

$$S_K = \frac{1}{\sigma_b^4} \sum_{b=0}^{L-1} (b - \bar{b})^4 P(b) - 3$$

Energy:

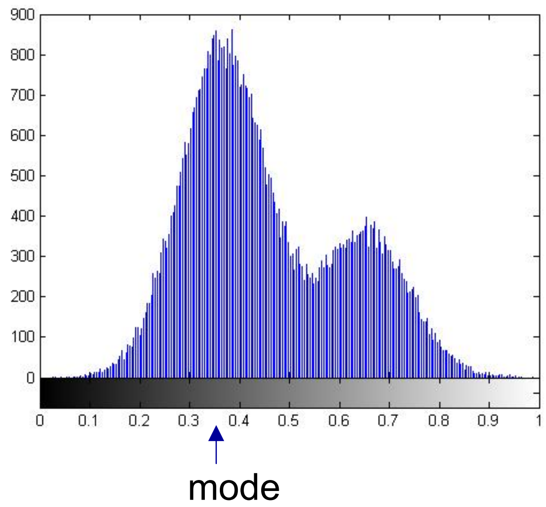
$$S_N = \sum_{b=0}^{L-1} [P(b)]^2$$

Entropy:

$$S_E = -\sum_{b=0}^{L-1} P(b) \log_2\{P(b)\}$$

Quantitative histogram shape descriptors

- *histogram mode*: the pixel amplitude corresponding to the histogram peak (i.e., the most commonly occurring pixel amplitude in the window).
- If the histogram peak is not unique, the pixel at the peak closest to the mean is usually chosen as the histogram shape descriptor.



bi-modal histogram

Quantitative histogram shape descriptors

- Second-order histogram features are based on the definition of the joint probability distribution of pairs of pixels.

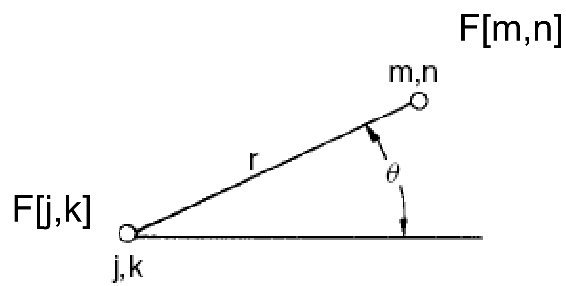


FIGURE 16.2-2. Relationship of pixel pairs.

The joint probability depends on the pixel relative position!

$$p(a,b) = p\{F(j,k) = a, F(n,m) = b\}$$

- Histogram estimate of the second-order distribution

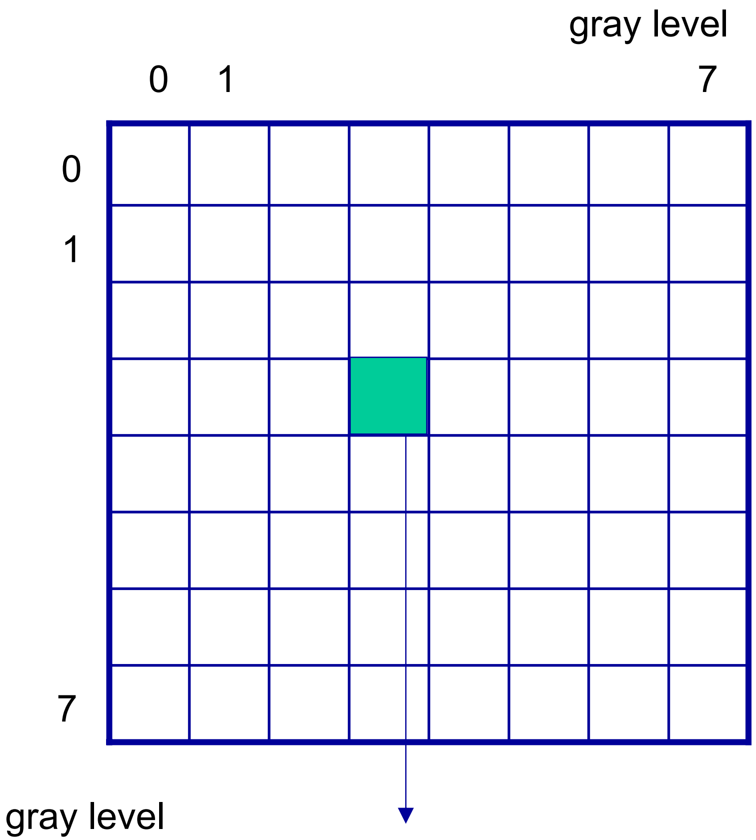
$$p(a,b) \simeq \frac{N(a,b)}{N}$$

$N(a,b)$ → number of occurrences for which F[j,k]=a AND F[n,m]=b

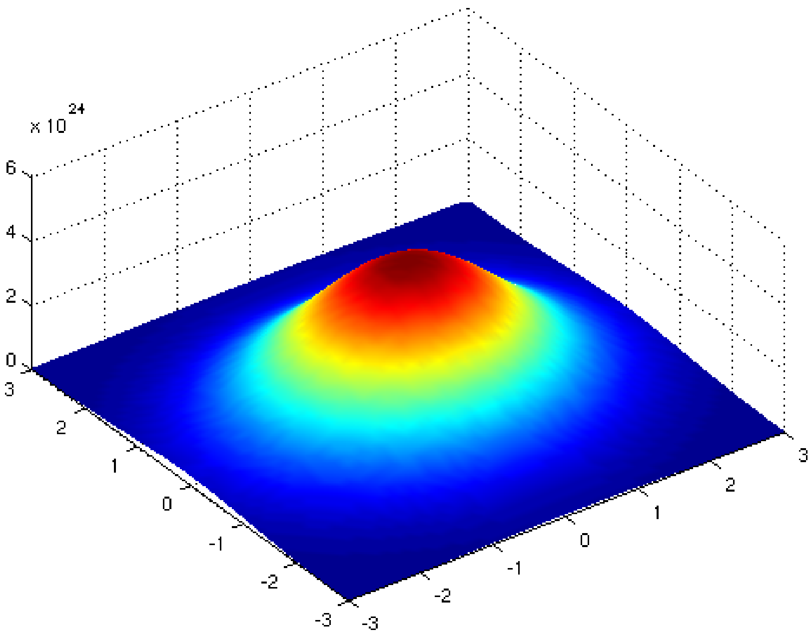
N → total number of pixels in the observation window

Second order histogram estimates

Given r, theta



number of occurrences of the two gray levels



co-occurrence matrix

Descriptors

Autocorrelation:

$$S_A = \sum^{L-1} \sum^{L-1} abP(a, b)$$

Covariance:

$$S_C = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} (a - \bar{a})(b - \bar{b})P(a, b)$$

where

$$\bar{a} = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} aP(a, b)$$

$$\bar{b} = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} bP(a, b)$$

Descriptors

Inertia:

$$S_I = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} (a-b)^2 P(a,b)$$

Absolute value:

$$S_V = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} |a-b| P(a,b)$$

Inverse difference:

$$S_F = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} \frac{P(a,b)}{1+(a-b)^2}$$

Energy:

$$S_G = \sum_{a=0}^{L-1} \sum_{b=0}^{L-1} [P(a,b)]^2$$

Entropy:

$$S_T = -\sum_{a=0}^{L-1} \sum_{b=0}^{L-1} P(a,b) \log_2\{P(a,b)\}$$

Transform features: Fourier

- The distribution (histogram) of the transformed coefficients is characterized either considering the whole frequency domain or partitioning it according to different criteria
- In each frequency region, the most common choice consists in using first order descriptors
  - mean
  - variance (as indicator of the energy)

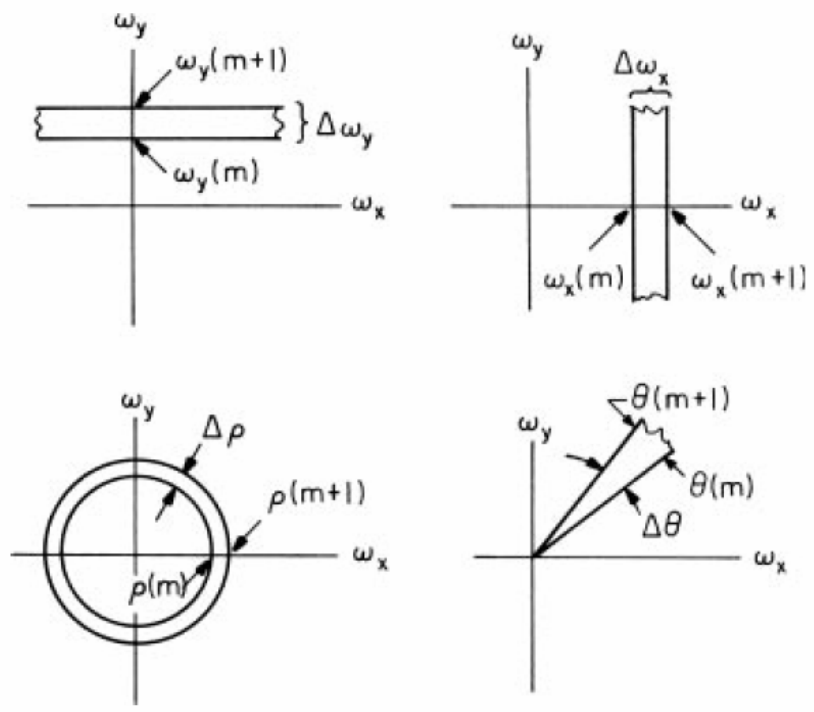


FIGURE 16.3-1. Fourier transform feature masks.

Segmentation

Image Segmentation

Contour-based

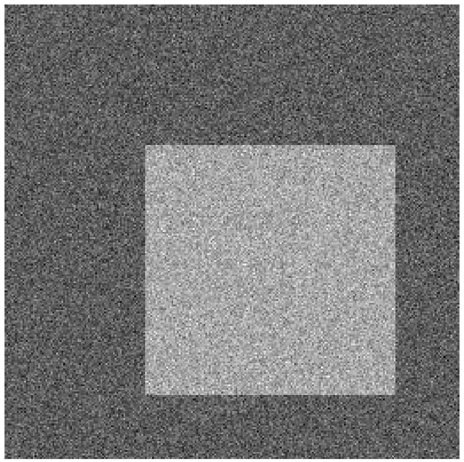
- Discontinuity
  - The approach is to partition an image based on *abrupt changes* in gray-scale levels.
  - The principal areas of interest within this category are detection of isolated points, lines, and edges in an image.

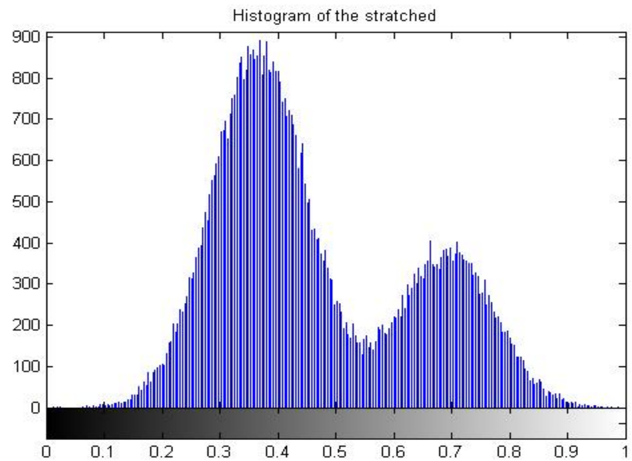
Region-based

- 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) 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

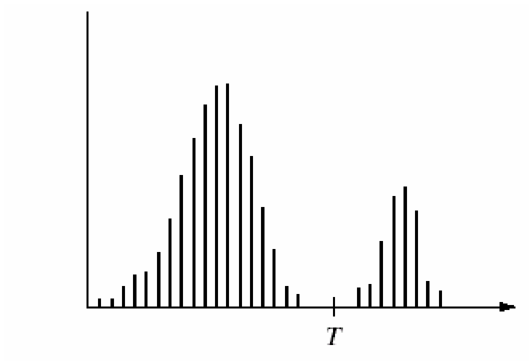
- Histogram shape-based methods
  - Peaks, valleys and curvatures of the smoothed histogram are analyzed
- Clustering-based methods
  - gray-level samples are clustered in two parts as background and foreground (object), or alternately are modeled as a mixture of two Gaussians
- Entropy-based methods
  - 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

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

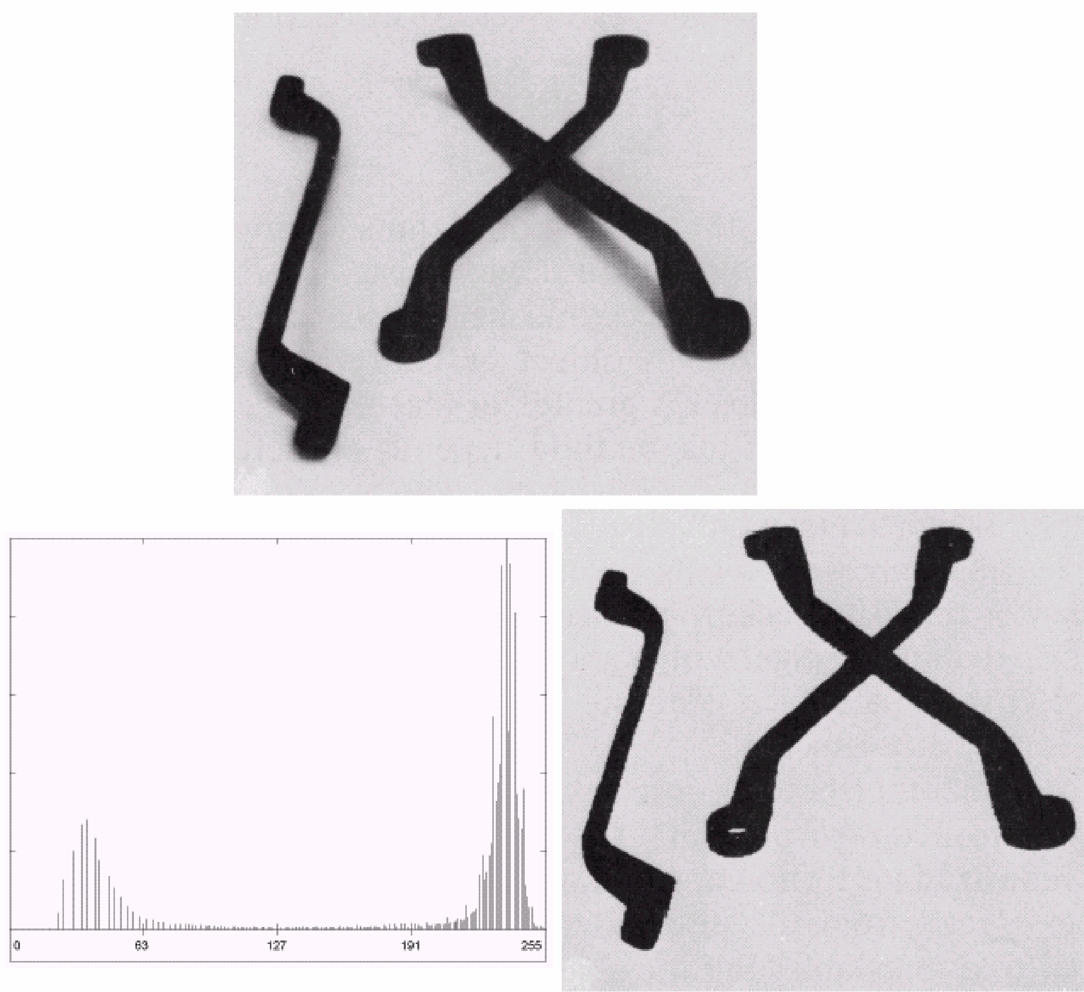
Histogram thresholding

- 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.

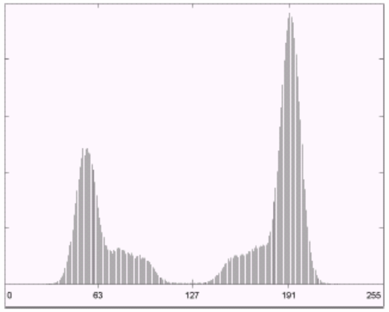
Thresholding

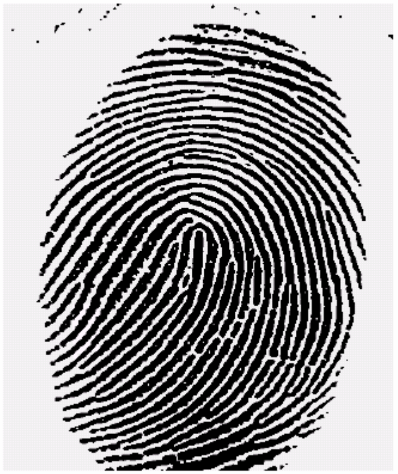


a
b c

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

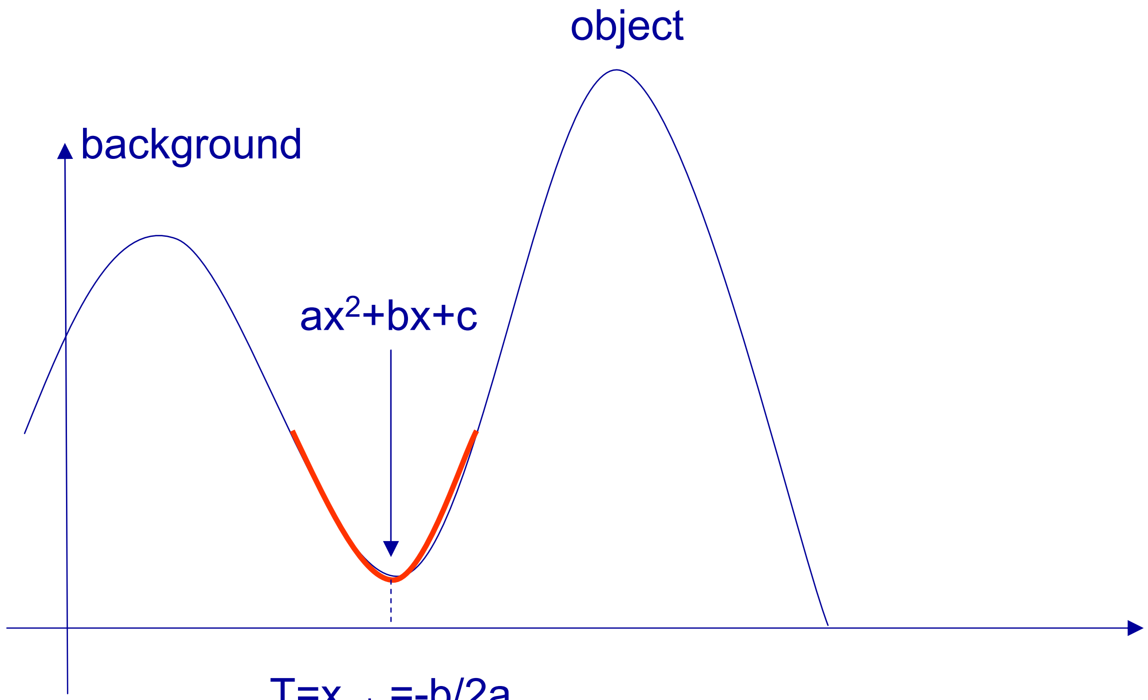
Thresholding





Histogram thresholding

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



$T = x_{min} = -b/2a$

Choice of the T value

- Empirical, by inspection
- Automatic
  1. Choose an initial value T
  2. Segment the image accordingly
     - This will produce twos sets of pixels, G1 and G2

$$G_1 : g > T$$

$$G_2 : g \leq T$$

  3. Update the threshold

$$T = \frac{1}{2}(\mu_1 + \mu_2), \qquad \mu_1 = \frac{1}{|G_1|} \sum_{i,j \in G_1} g[i,j], \qquad \mu_2 = \frac{1}{|G_2|} \sum_{i,j \in G_2} g[i,j]$$

  4. Go back to 2 until the change due to the update of T reaches a lower bound ΔT_0

Multilevel luminance thresholding

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

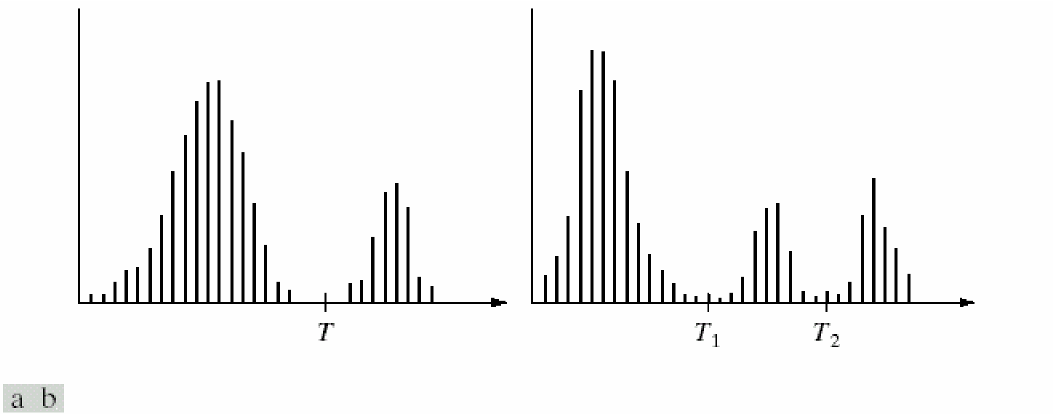


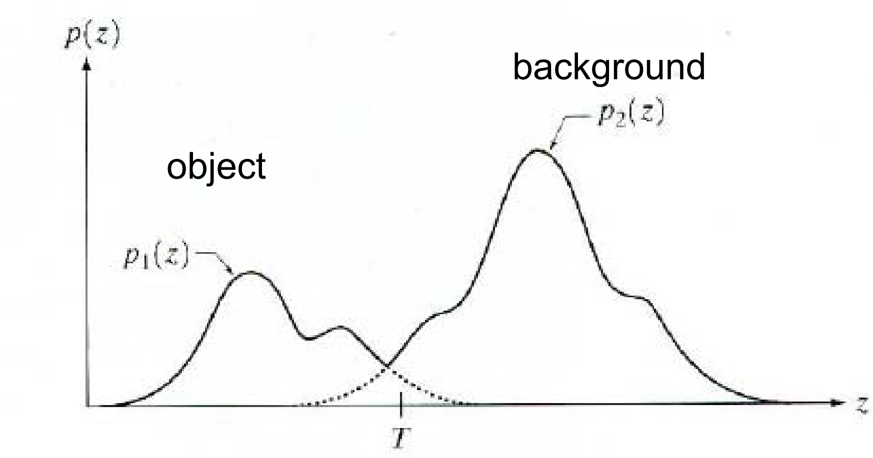
FIGURE 10.26 (a) Gray-level histograms that can be partitioned by (a) a single threshold, and (b) multiple thresholds.

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…

Optimal global and adaptive thresholding

- Assumptions:
  - The image contains only two principal gray level regions
  - The histogram is bimodal
  - It can be considered as a good estimate of the pdf
- Model:
  - The global histogram is given by the mixture (sum) of the two pdfs
  - The weights are proportional to the relative areas of the dark and light regions
    - And thus are given by the areas under the two, respectively

Optimal global and adaptive thresholding



Mixture pdf describing the global gray level variations in the image

$$p(z) = P_1 p_1(z) + P_2 p_2(z)$$

$$P_1 + P_2 = 1$$

Probability of errors

- Misclassification of a background point as object

$$E_1(T) = \int_{-\infty}^{T} p_2(z)\,dz$$

- Misclassification of an object point as background

$$E_2(T) = \int_{T}^{+\infty} p_1(z)\,dz$$

- Total error probability

$$E(T) = P_2 E_1(T) + P_1 E_2(T)$$

  - E_1 is weighted by P_2 because if the probability of background points is zero than the contribution to such points to the error is zero too

Finding the threshold

- Take the derivative of E with respect to T and set it to zero

$$E(T) = P_2 E_1(T) + P_1 E_2(T)$$

$$\frac{dE}{dt} = P_2 \frac{dE_1}{dT} + P_1 \frac{dE_2}{dT} = p_2 P_2 - p_1 P_1 = 0$$

$$p_1 P_1 = p_2 P_2$$

- Notes
  - If P1=P2 then the optimal value for T corresponds to the intersect of the curves
  - The explicit calculation of T requires the knowledge of the pdf, which is not always the case
  - In general, it is assumed that the two pfs are Gaussian

Finding the threshold

- For Gaussian mixtures

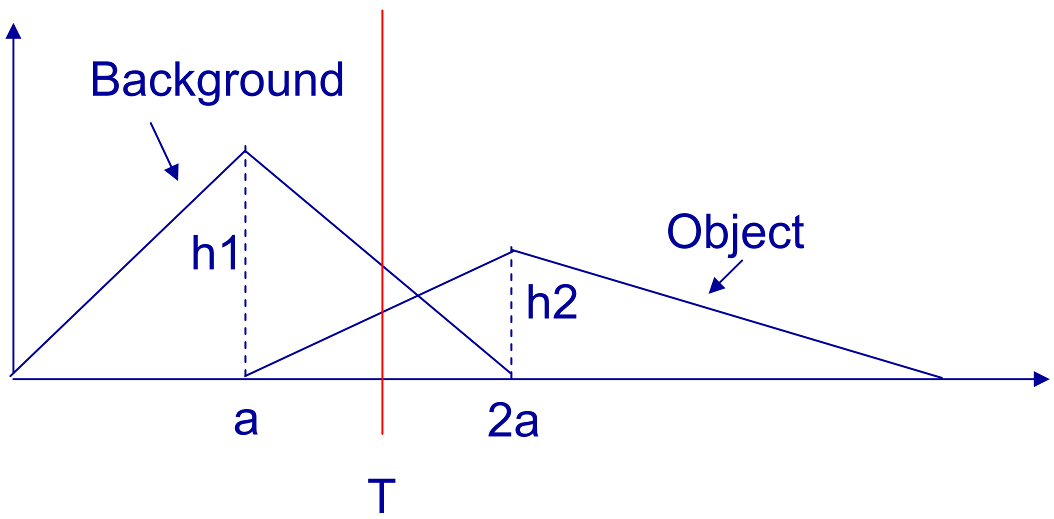
$$p(z) = \frac{P_1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(z-\mu_1)^2}{2\sigma_1^2}} + \frac{P_2}{\sqrt{2\pi}\sigma_2} e^{-\frac{(z-\mu_2)^2}{2\sigma_2^2}}$$

$$\text{if} \quad \sigma = \sigma_1 = \sigma_2$$

$$\rightarrow T = \frac{\mu_1 + \mu_2}{2} \frac{\sigma^2}{\mu_1 - \mu_2} \ln\left(\frac{P_2}{P_1}\right)$$

Clustering based thresholding

- Definition and optimization of a *cost (or objective) function*
  - 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 grey levels for the object are closest to the average of the object and the grey levels 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:

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

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

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

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

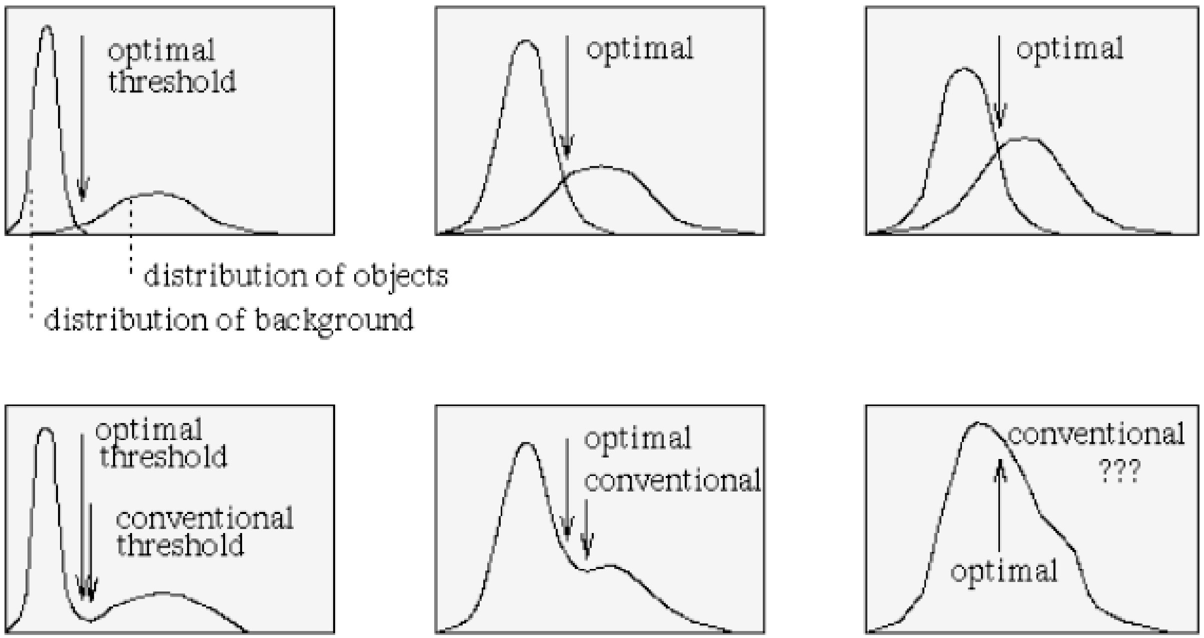
$\sigma^2_0(T)$: variance of the pixels in the object (g $\geq$ T)

$0,\ldots,N-1$: 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

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



Bottlenecks

- 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
- Guideline: partition the image in blocks of almost uniform luminance and perform the segmentation locally
  - In alternative, one can apply chromatic adaptation transforms which compensate for differences in the scene illumination, such as retinex

Examples

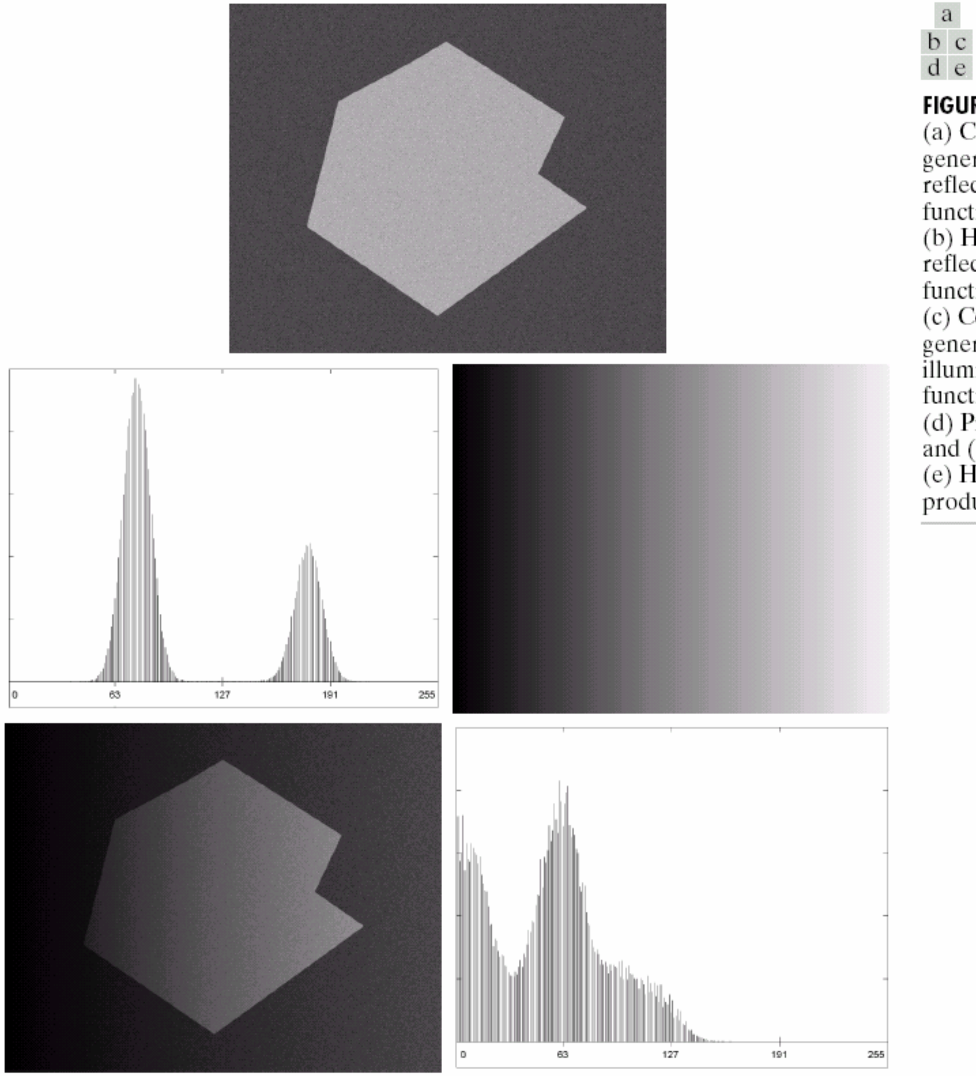
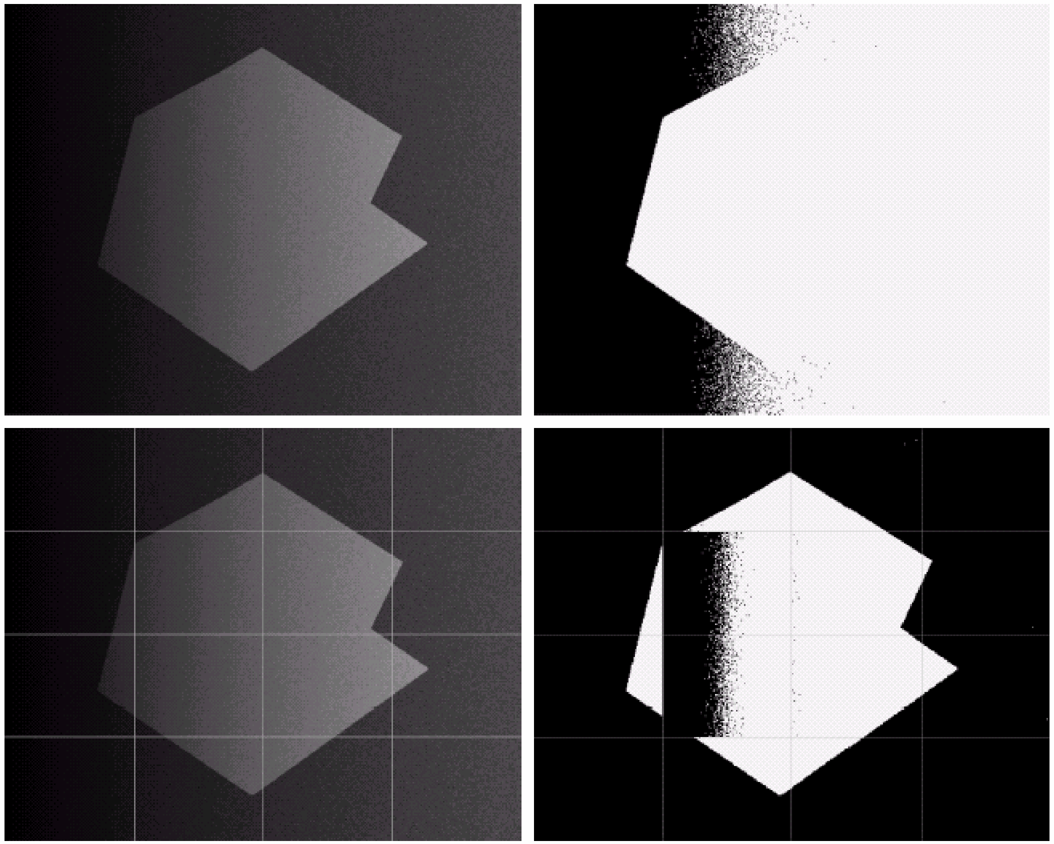


FIGURE 10.27
(a) Computer generated reflectance function.
(b) Histogram of reflectance function.
(c) Computer generated illumination function.
(d) Product of (a) and (c).
(e) Histogram of product image.

Examples

- 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 based segmentation

- Formulation

$\bigcup_{i=1}^{n} R_i = R$	the segmentation must be complete
R_i is a connected region $i = 1, \ldots, n$	the points in a region must be connected according to a predefined criterion
$R_i \cap R_j = \varnothing \quad \forall i \neq j$	the regions must be disjoint
$PR(R_i) = TRUE \quad i = 1, \ldots, n$	condition that is satisfied by all points in R_i
$PR(R_i \cup R_j) = FALSE \quad \forall i \neq j$	regions Ri and Rj are different with respect to predicate PR

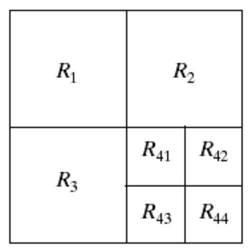
PR: logical predicate defined over the region. Ex: all points in the region have the same gray level

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

- 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.



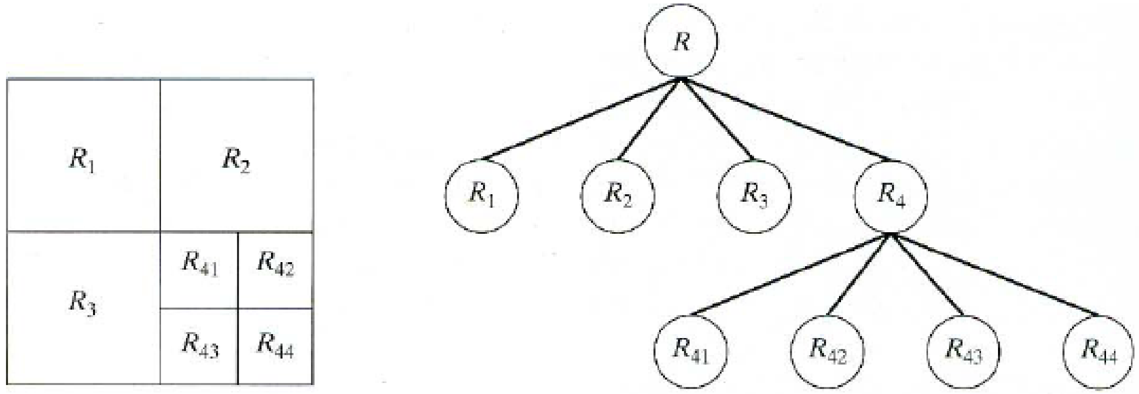
Region-Oriented Segmentation

- Region splitting and merging
  - The image is initially splitted into regions arbitrarily. These are subsequently merged and/or further splitted to satisfy the predefined homogeneity criterion
  - Let R be a region and PR a predicate. The approach consists in taking initially R=entire image and splitting it in subregions such that at the end of the process $PR(R_i)$=TRUE in every region.
  - Recipe:

1. Evaluate PR over R: if it is FALSE then split R in, let's say, 4 subregions
2. Repeat the procedure for each resulting region
3. For each couple i,j evaluate $PR(R_i \cup R_j)$. If this is TRUE then merge R_i and R_j
4. Stop when no further splitting or merging is possible

Region splitting and merging

- Image *quadtree* resulting for the considered type of splitting



Region-Oriented Segmentation

a b c

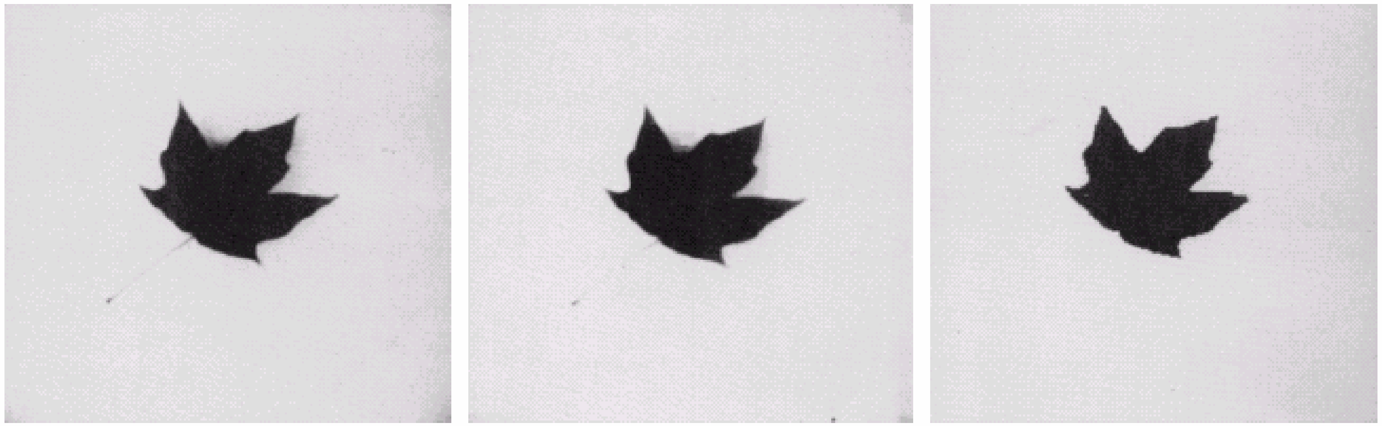


FIGURE 10.43 (a) Original image. (b) Result of split and merge procedure. (c) Result of thresholding (a).

Towards texture segmentation

- All the methods using means and variances to characterize regions basically characterize the *texture* of the region
- The concept of texture segmentation consists in using *texture features* as predicates

Example

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

Example

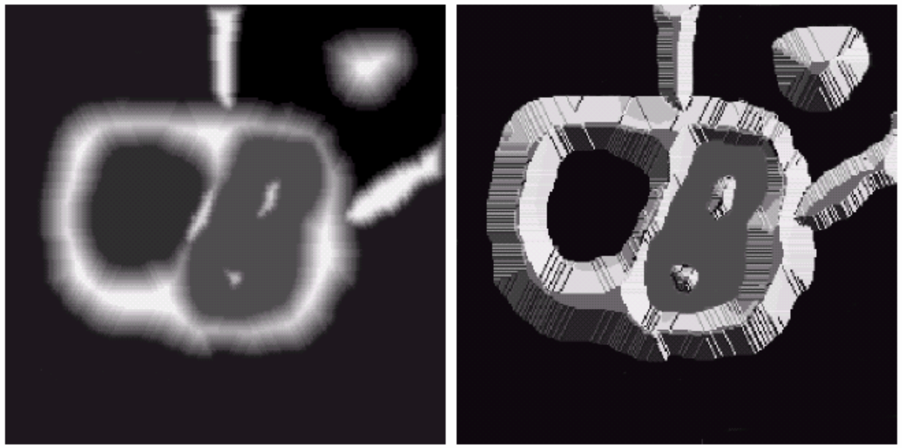
(b) What will be the segmentation if region is grown in horizontal, vertical, and diagonal directions?

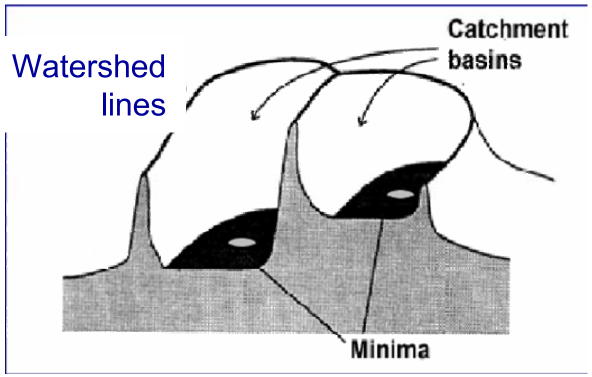
Table 2: Show the result of Part (b) on this figure.

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

Watershed Segmentation Algorithm

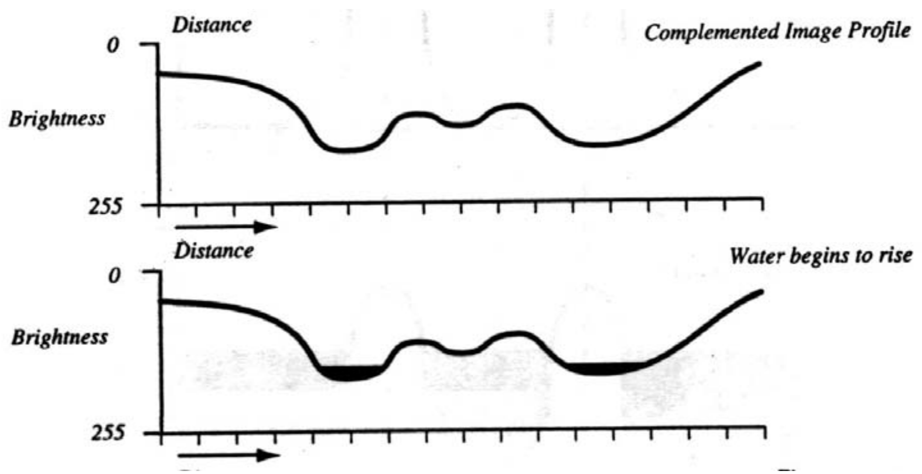
- Visualize an image in 3D: spatial coordinates and gray levels.
- In such a topographic interpretation, there are 3 types of points:
  - Points belonging to a regional minimum
  - Points at which a drop of water would fall to a single minimum. (→The *catchment basin* or *watershed* of that minimum.)
  - Points at which a drop of water would be equally likely to fall to more than one minimum. (→The *divide lines* or *watershed lines*.)



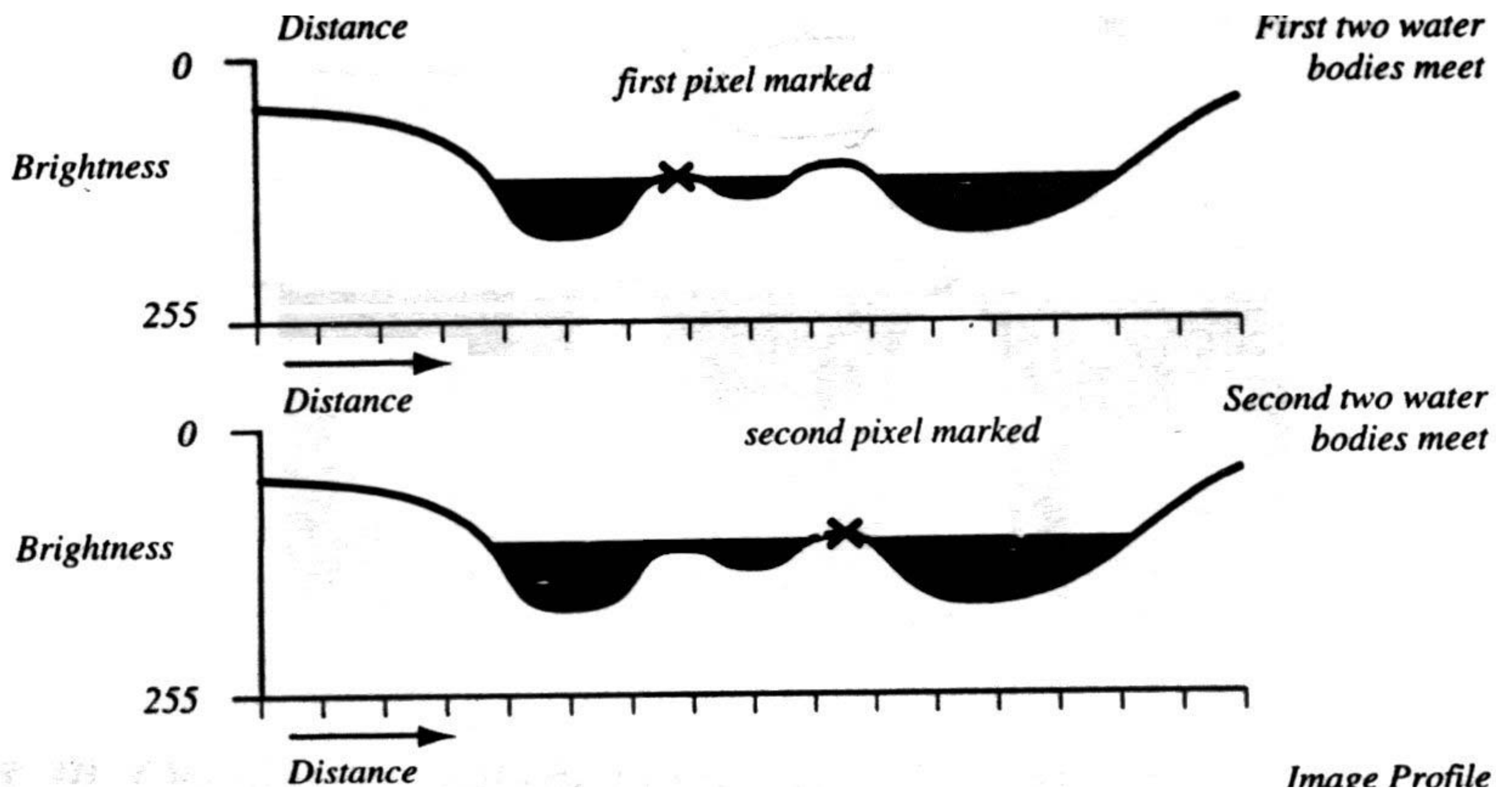


Watershed Segmentation Algorithm

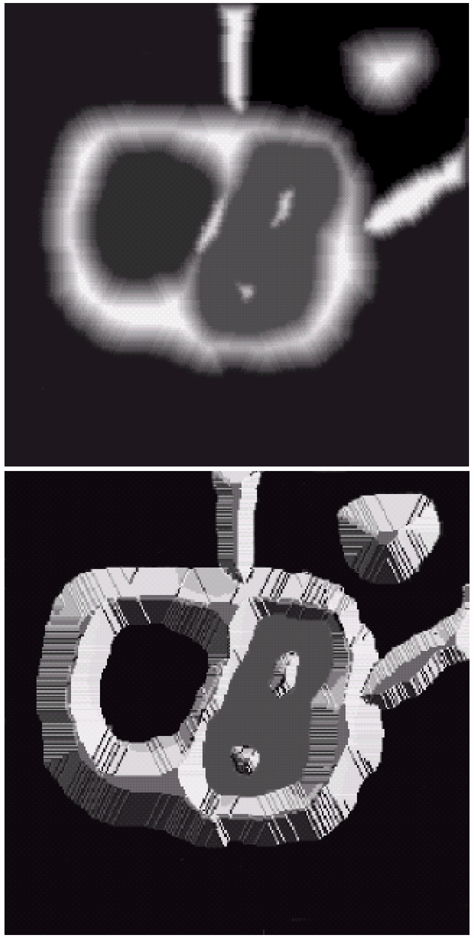
- *The objective is to find watershed lines.*
- The idea is simple:
  - Suppose that a hole is punched in each regional minimum and that the entire topography is flooded from below by letting water rise through the holes at a uniform rate.
  - When rising water in distinct catchment basins is about the merge, a dam (*diga*) is built to prevent merging.
  - Dam boundaries correspond to the watershed lines.

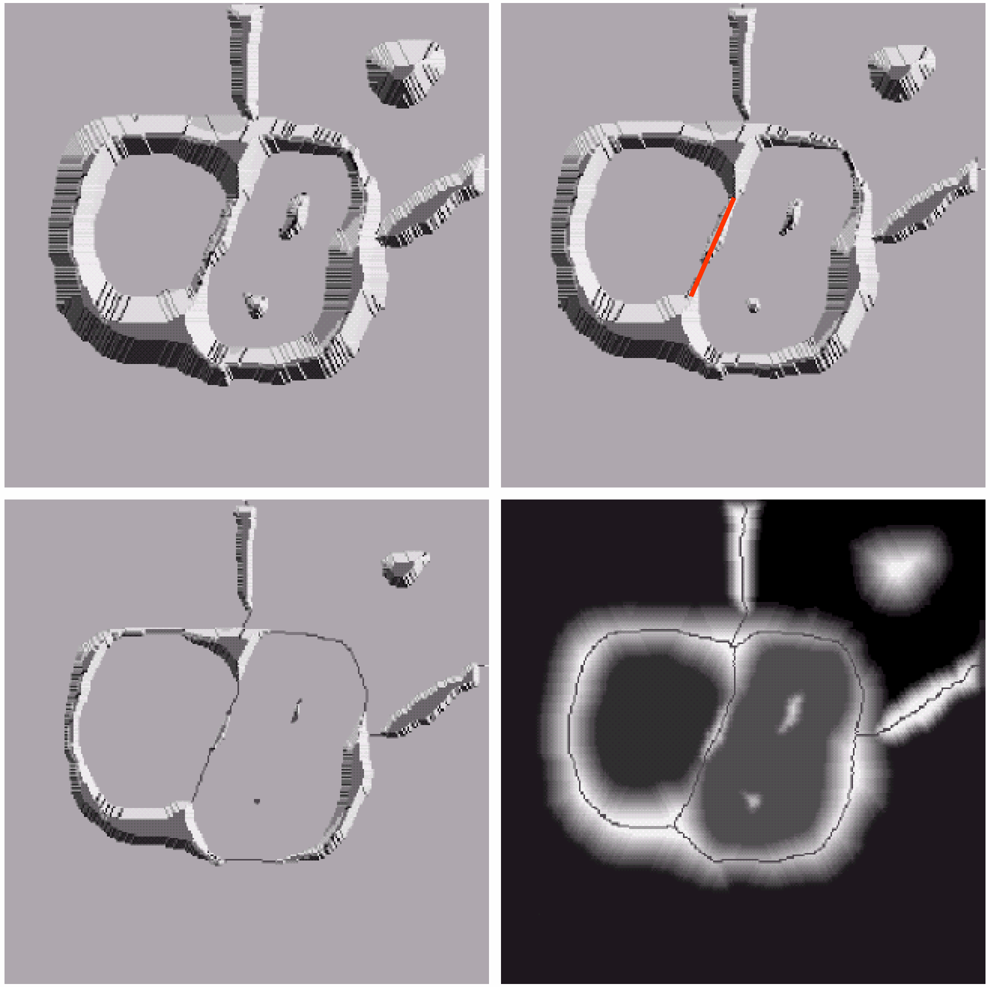


Watershed Segmentation Algorithm



Watershed Segmentation Algorithm





e f
g h

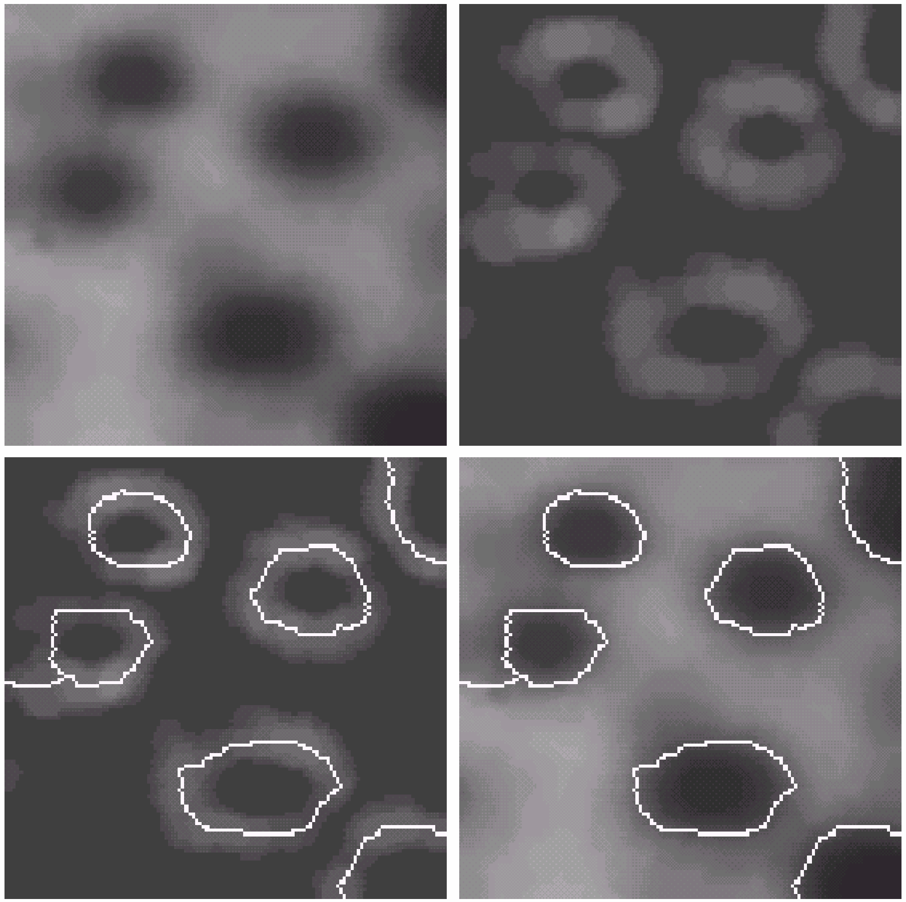
FIGURE 10.44 *(Continued)* (e) Result of further flooding. (f) Beginning of merging of water from two catchment basins (a short dam was built between them). (g) Longer dams. (h) Final watershed (segmentation) lines. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

Watershed Segmentation Algorithm

- Start with all pixels with the lowest possible value.
  - These form the basis for initial watersheds
- *For each intensity level k*:
  - For each group of pixels of intensity k
    - If adjacent to exactly one existing region, add these pixels to that region
    - Else if adjacent to more than one existing regions, mark as boundary
    - Else start a new region

Watershed Segmentation Algorithm

Watershed algorithm might be used on the gradient image instead of the original image.

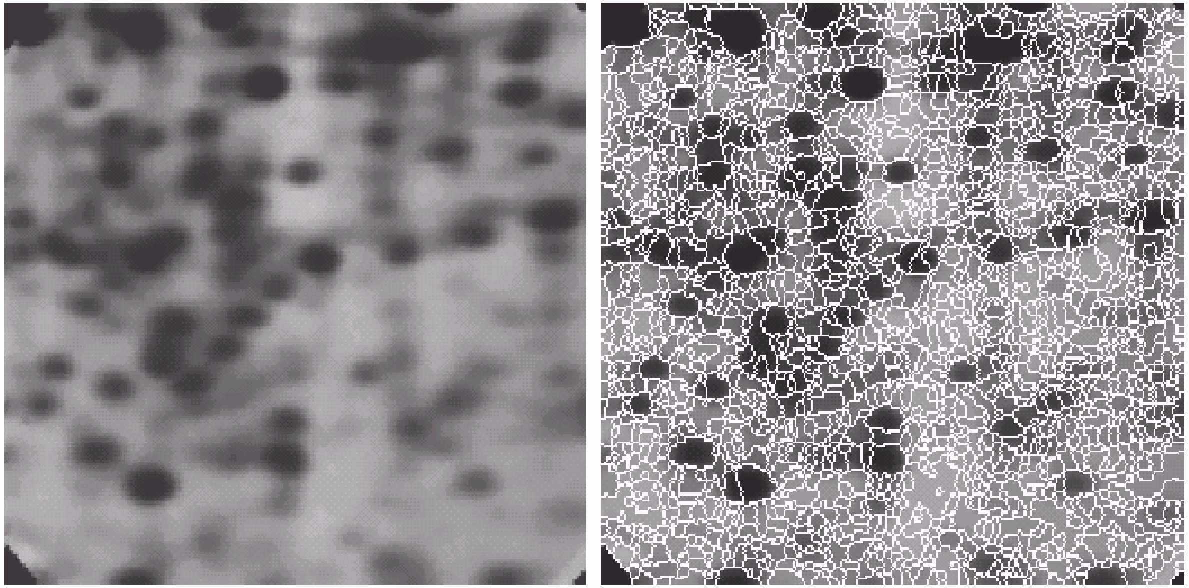


a b
c d

FIGURE 10.46 (a) Image of blobs. (b) Image gradient. (c) Watershed lines. (d) Watershed lines superimposed on original image. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

Watershed Segmentation Algorithm

Due to noise and other local irregularities of the gradient, over-segmentation might occur.

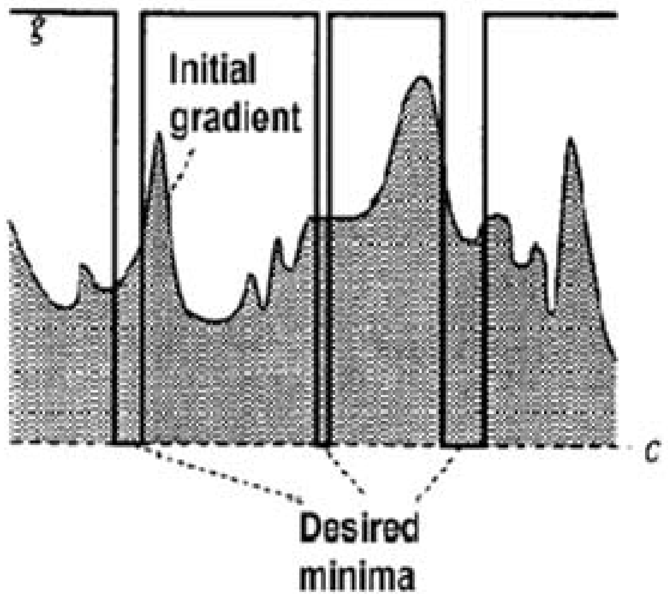


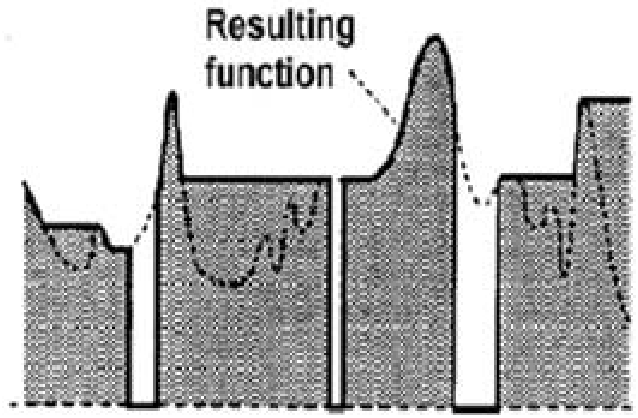
a b

FIGURE 10.47 (a) Electrophoresis image. (b) Result of applying the watershed segmentation algorithm to the gradient image. Oversegmentation is evident. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

Supervised Watershed Segmentation

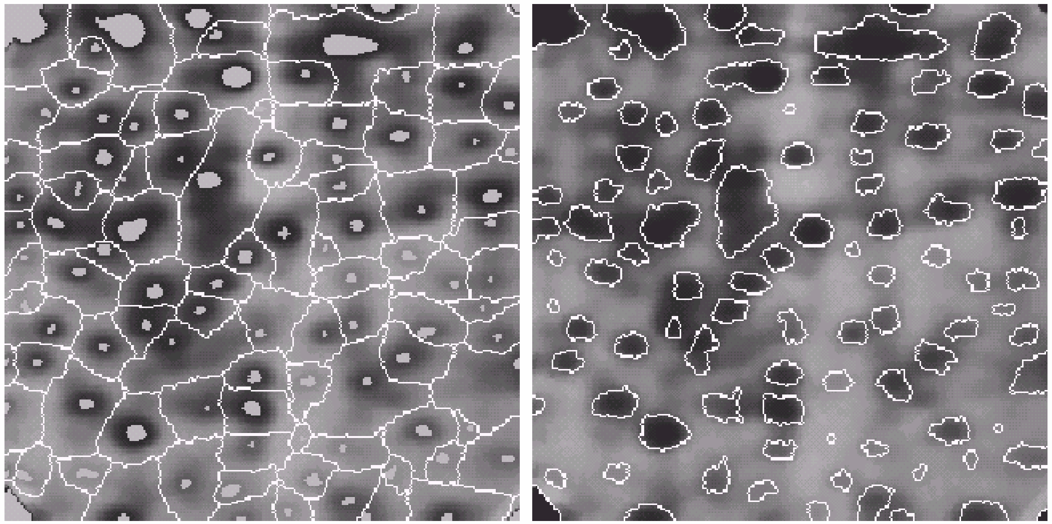
A solution is to limit the number of regional minima. Use markers to specify the only allowed regional minima.





Watershed Segmentation Algorithm

A solution is to limit the number of regional minima. Use markers to specify the only allowed regional minima. (For example, gray-level values might be used as a marker.)



a b

FIGURE 10.48 (a) Image showing internal markers (light gray regions) and external markers (watershed lines). (b) Result of segmentation. Note the improvement over Fig. 10.47(b). (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

A detailed description of the algorithm can be found in Gonzalez, Chapt. 10.

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.)

Motion based segmentation

- Video sequences
- Concept: detect the changes from one image to the next
- Possible approaches
  - Taking image differences
  - Block matching
  - Optical flow

Difference images

- Difference image between two images taken at time points ti and tj

$$d_{ij}(x,y) = \begin{cases} 1 & \text{if } |f(x,y,t_i) - f(x,y,t_j)| > T \\ 0 & \text{otherwise} \end{cases}$$

  - d_{ij}=1 only if the difference between the pixel values in the two images are above a given threshold T
  - d_{ij} has the same size as the two images
- Drawbacks
  - Sensitivity to noise
    - *Accumulation* strategies can be devised
  - Only allows to detect motion but not to characterize it
    - This would require establishing correspondences among pixels to calculate *motion vectors*

Block matching

Block-matching

16x16 pixels/block

Search window: ±16 pixels from the original position

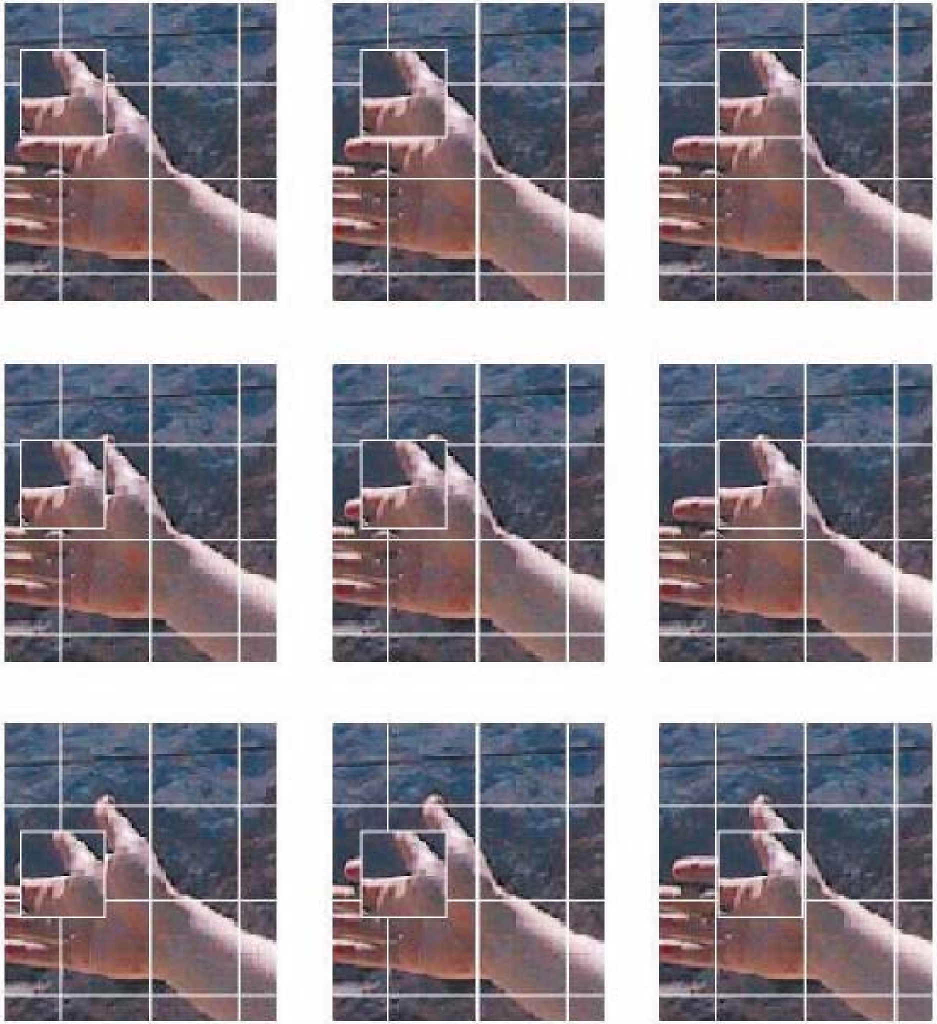
Computationally heavy!

To reduce the complexity

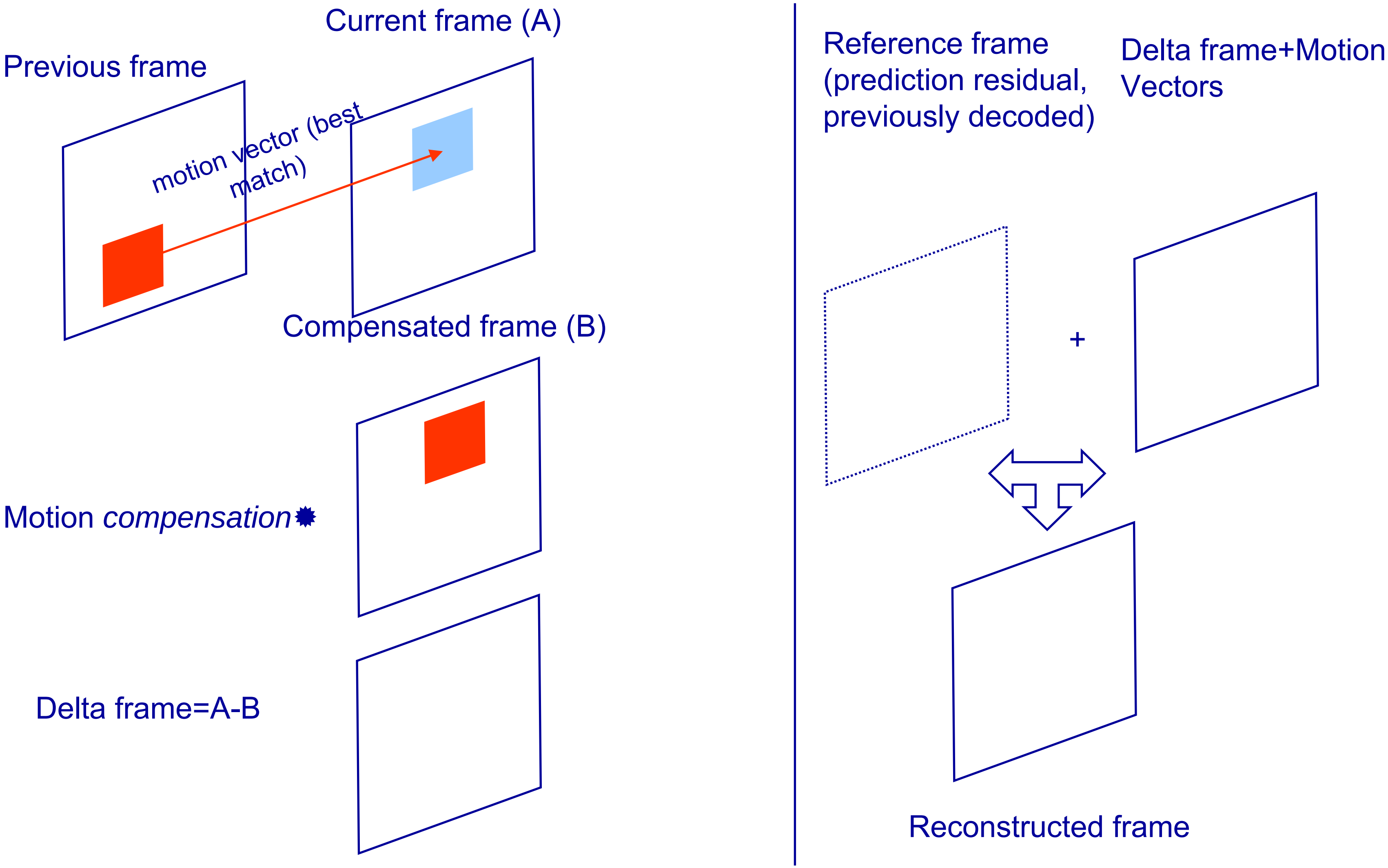
Sub-optimal algorithms

Hardware assisted

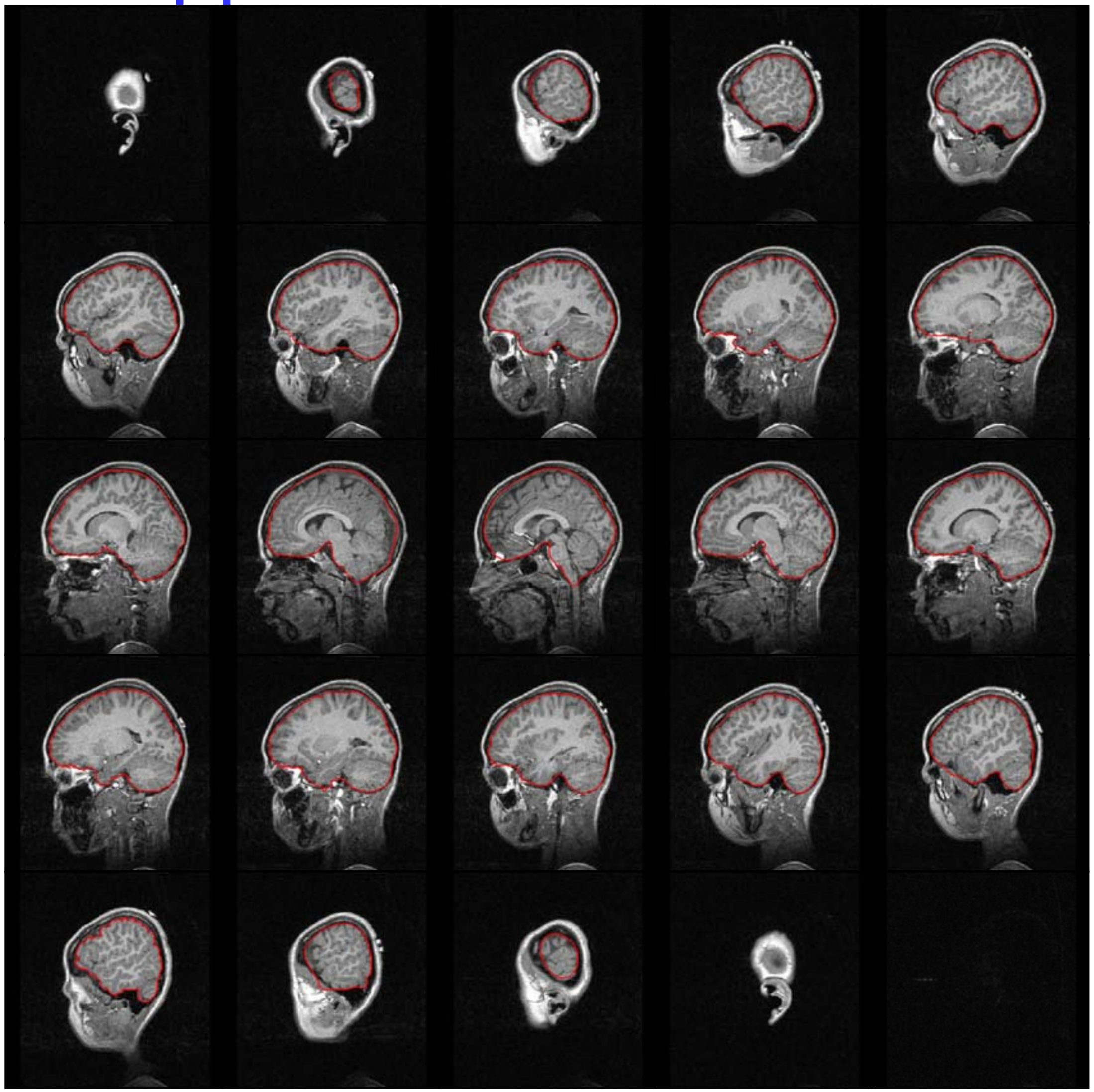
Block matching



Motion estimation & Motion compensation



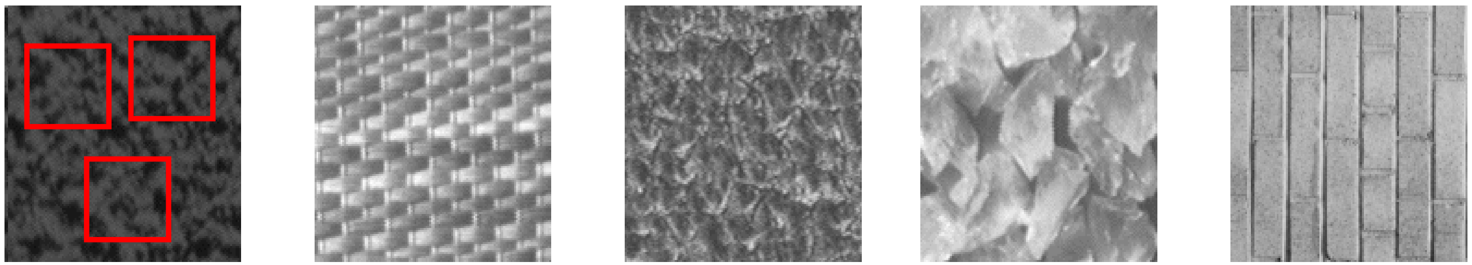
Application to 3D data



Texture recognition

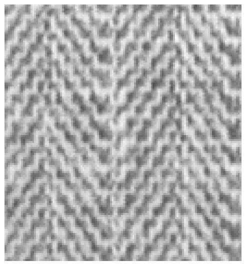
What is texture?

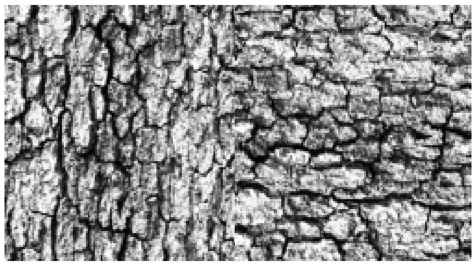
- No agreed reference definition
  - Texture is property of areas
  - Involves spatial distributions of grey levels
  - A region is perceived as a texture if the number of primitives in the field of view is sufficiently high
  - Invariance to translations
  - Macroscopic visual attributes
    - uniformity,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

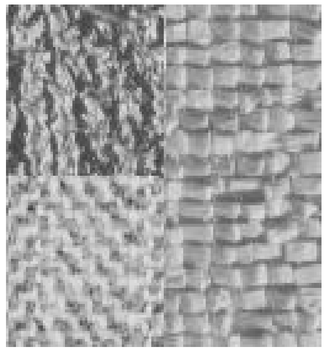


Texture analysis

- 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

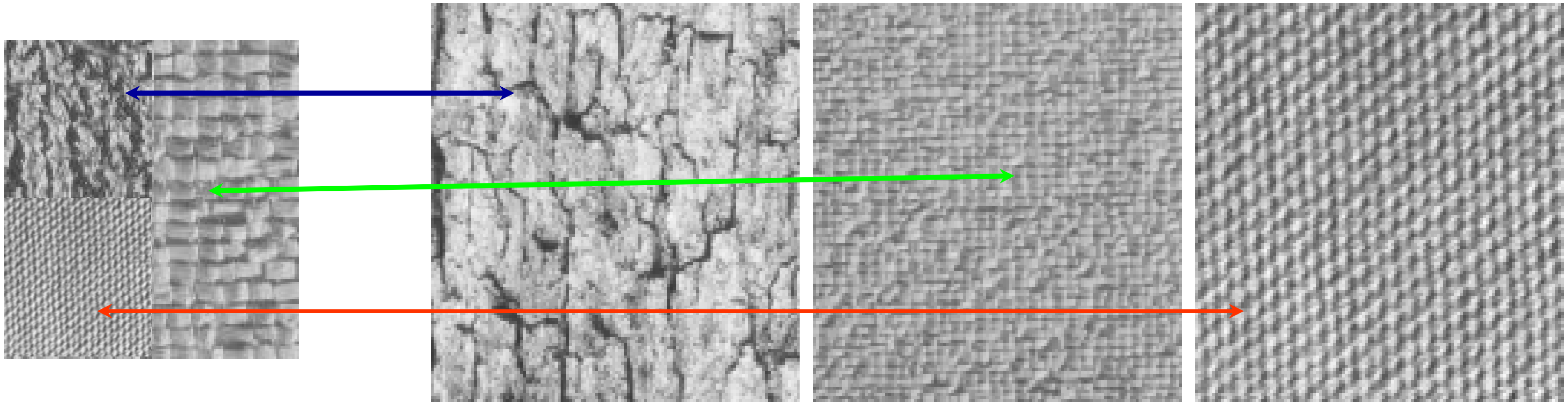






Texture analysis

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

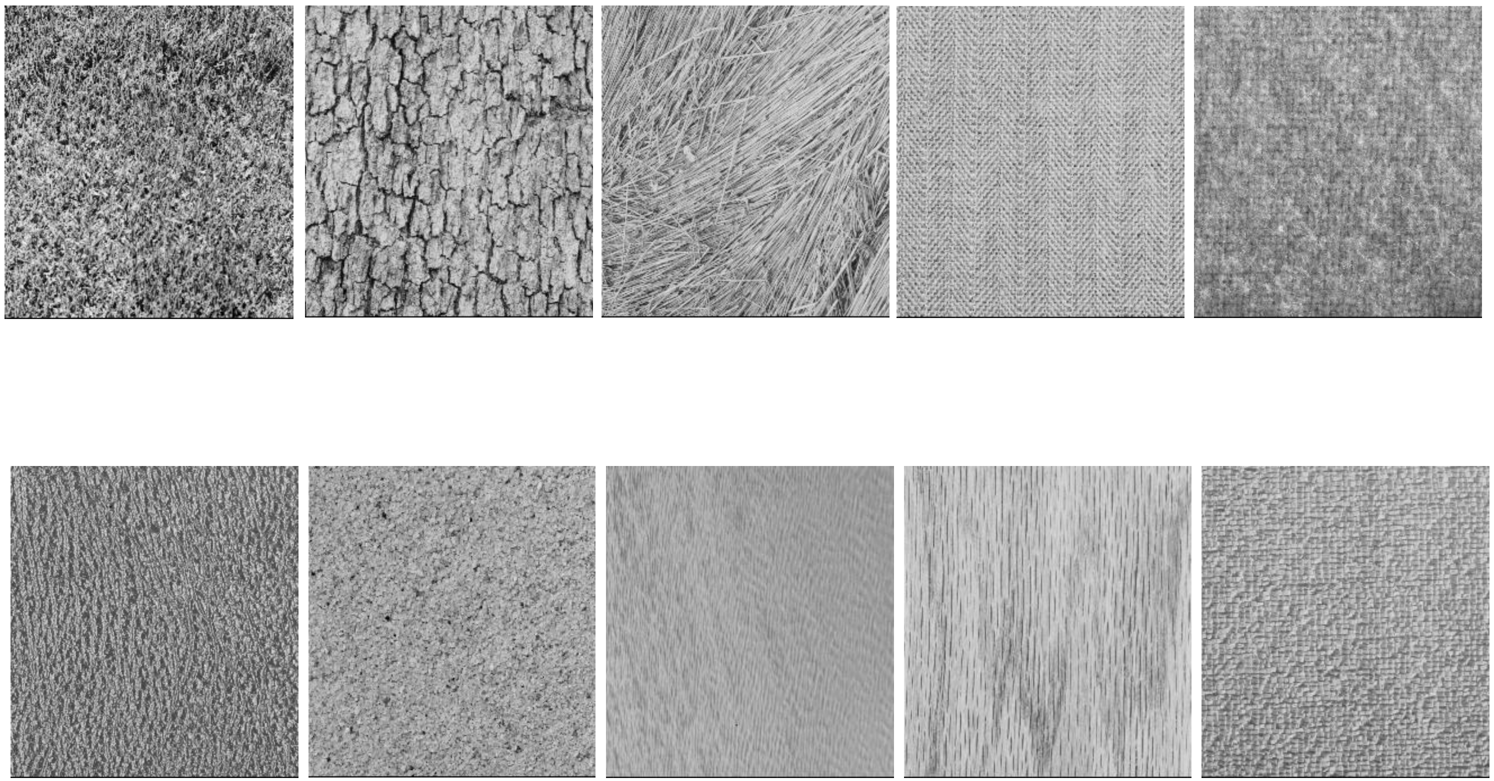
$$\omega_{winner} = \max_k P(\omega_i | x_k)$$

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

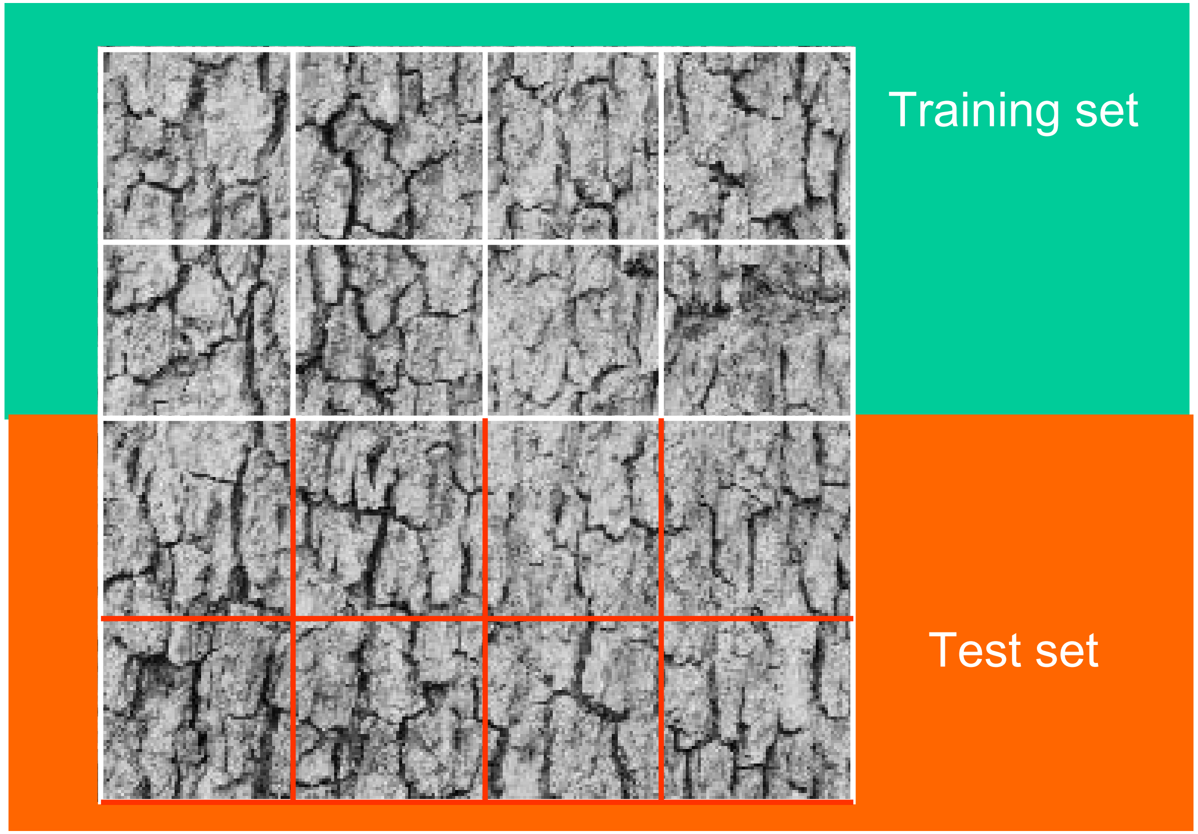
Exemple: texture classes

ω_1 ω_2 ω_3 ω_4



FV extraction

- Step 1: create independent texture instances



Feature extraction

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

Intensity image

↓

Transformation

↓

For each sub-image

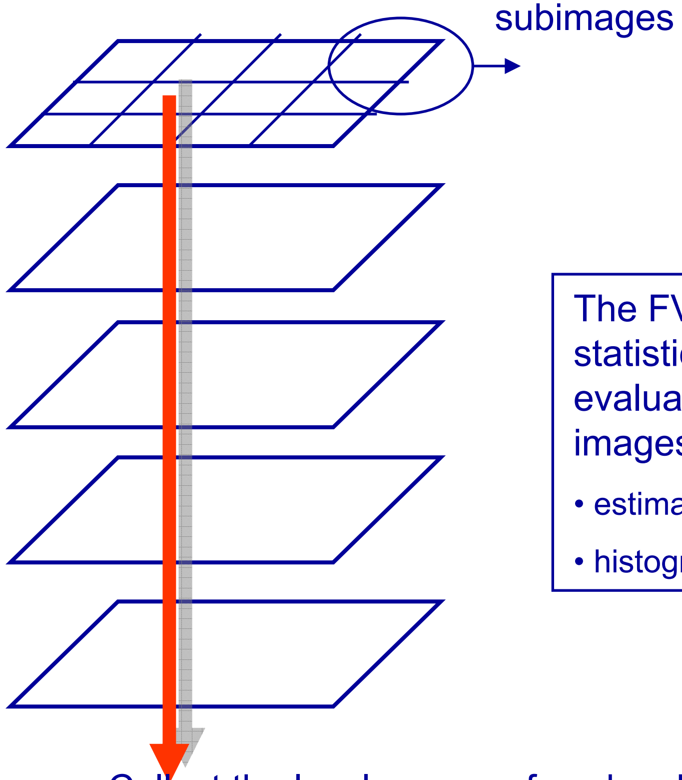
For each subband

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

Image pyramids: concept

Multiresolution signal analysis
(computer vision)

Image pyramids

Idea: Represent NxN image as a "pyramid" of 1×1, 2×2, $4 \times 4, \ldots, 2^k \times 2^k$ images (assuming $N = 2^k$)

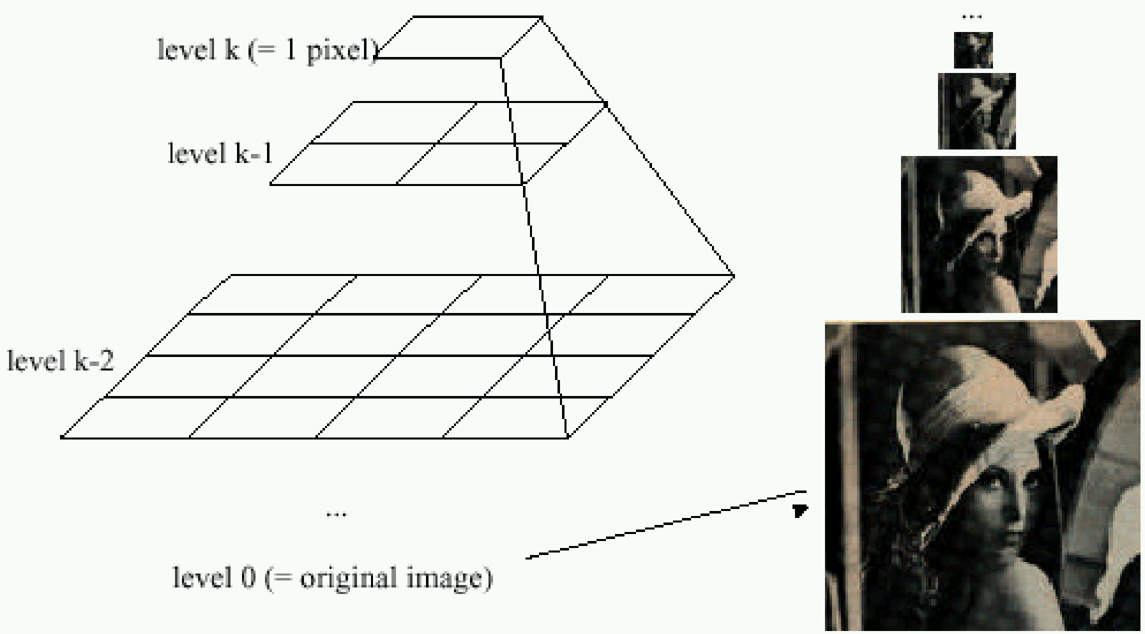


Image pyramid: feature extraction

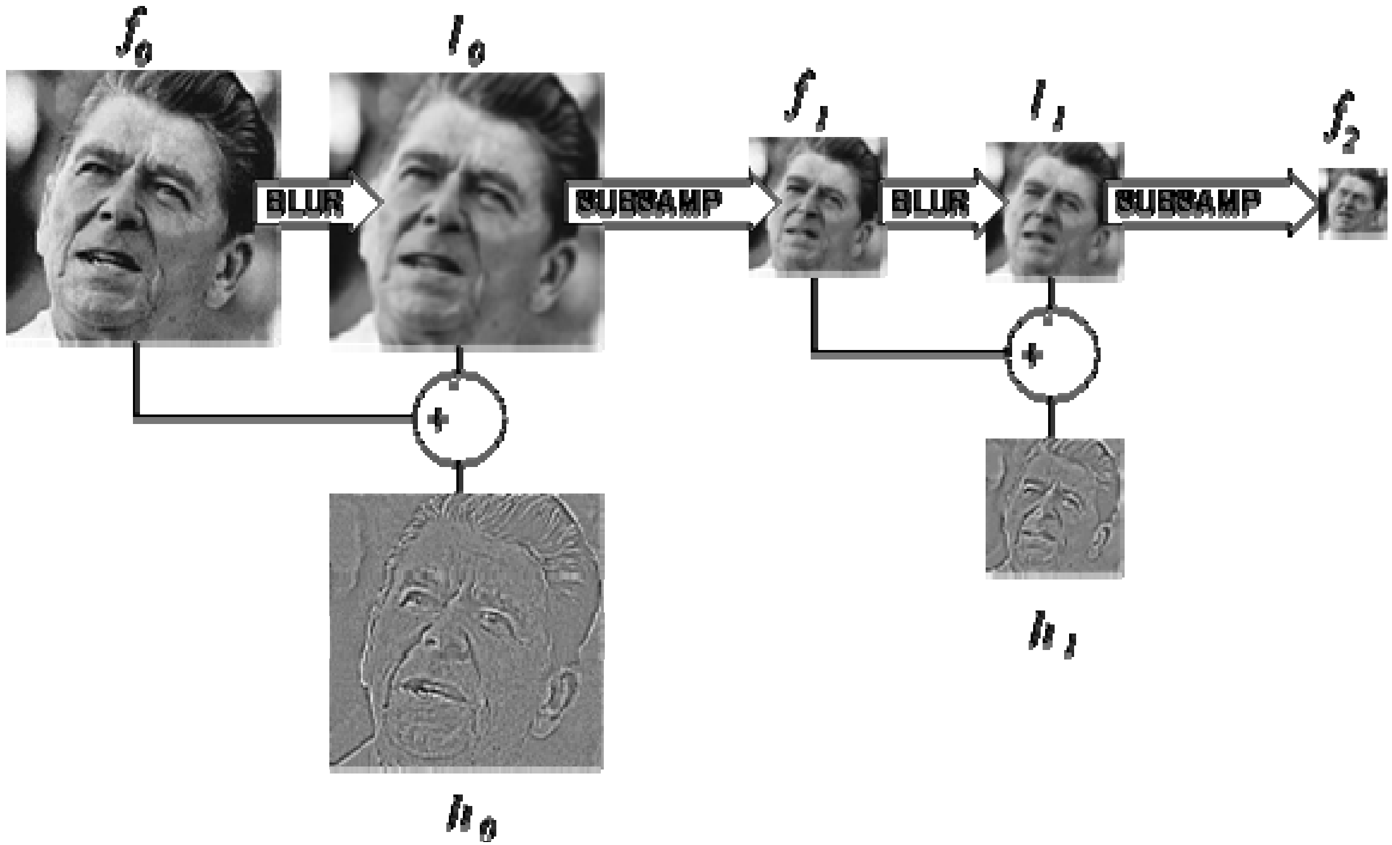
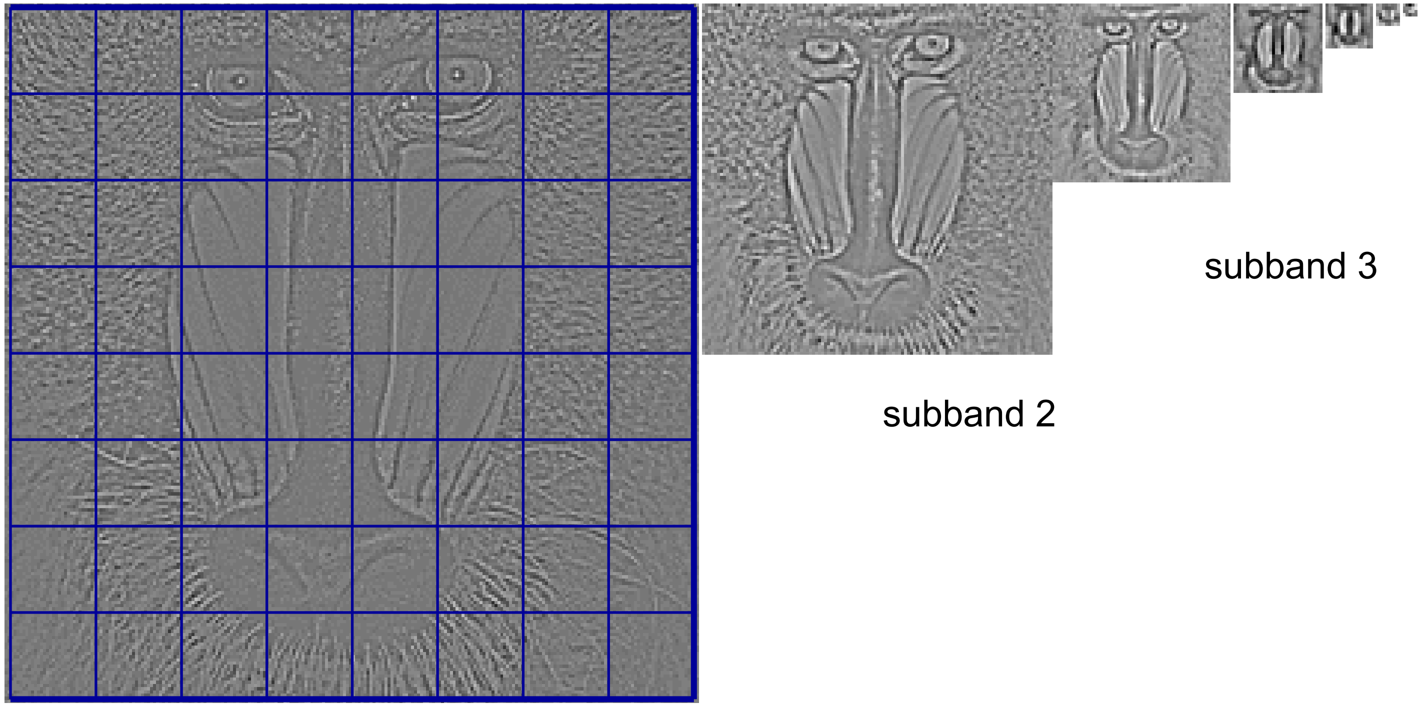
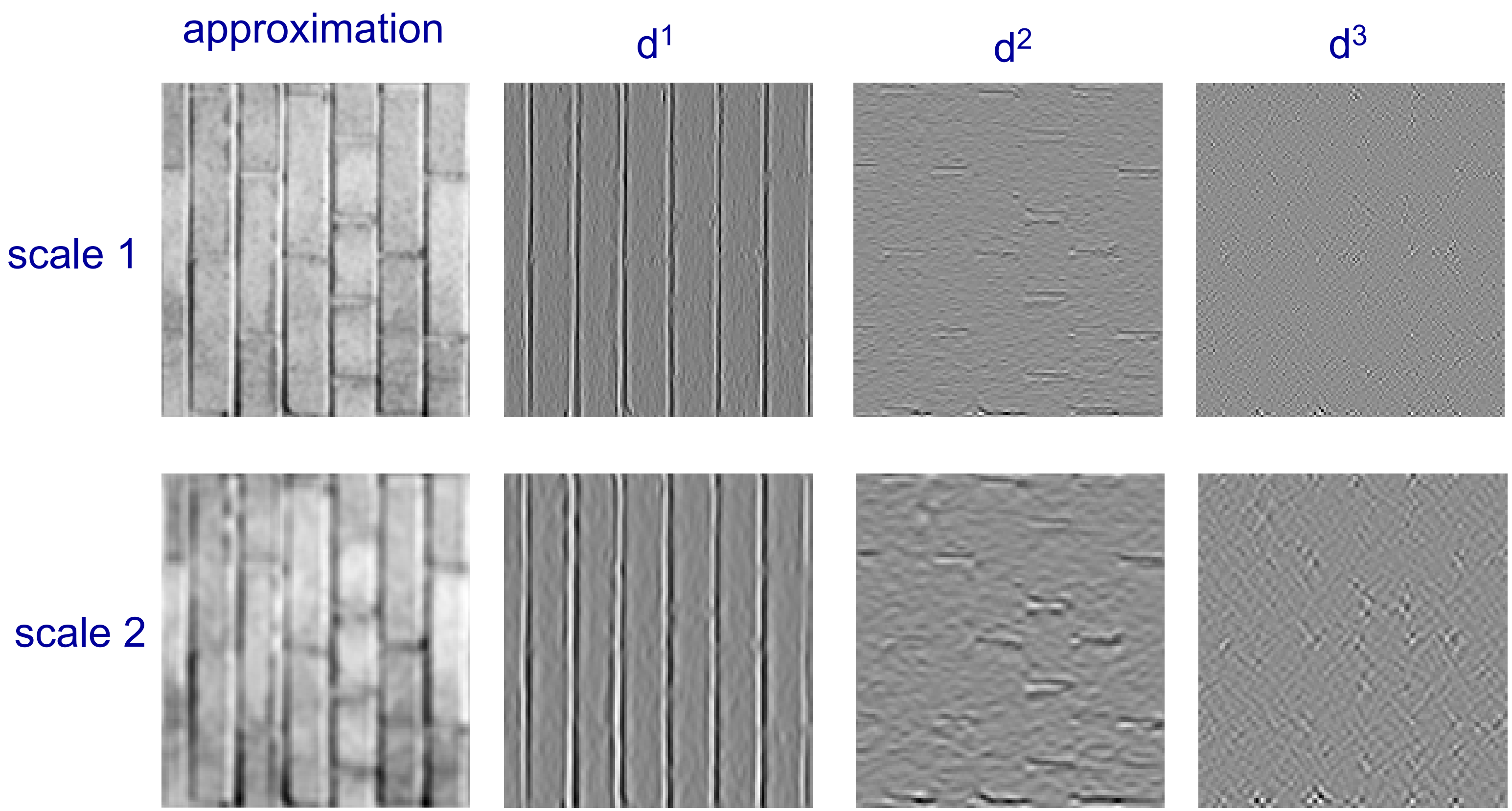


Image pyramid: feature extraction

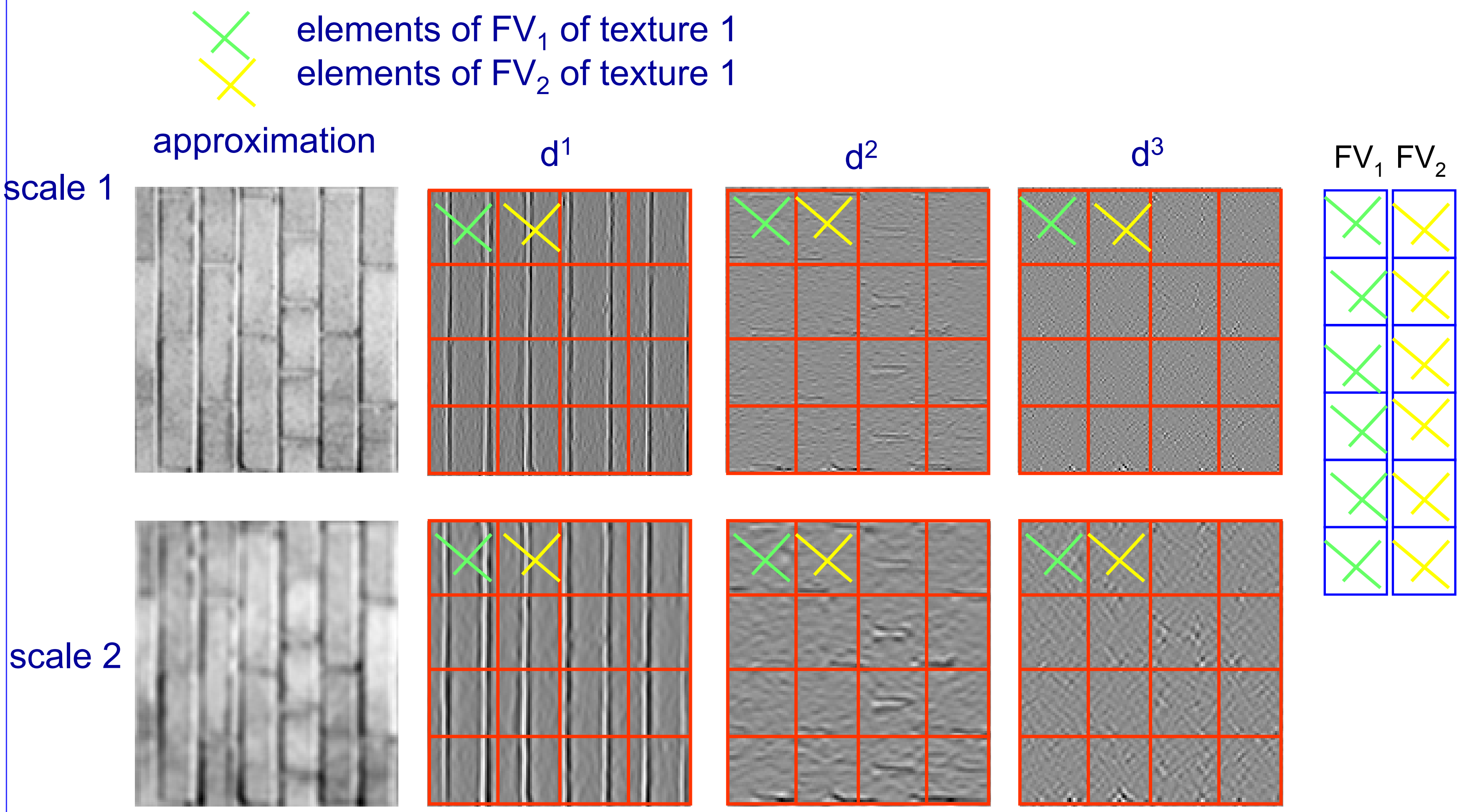


Texture features are claculated over the blocks and gathered into feature vectors.

Building the FV

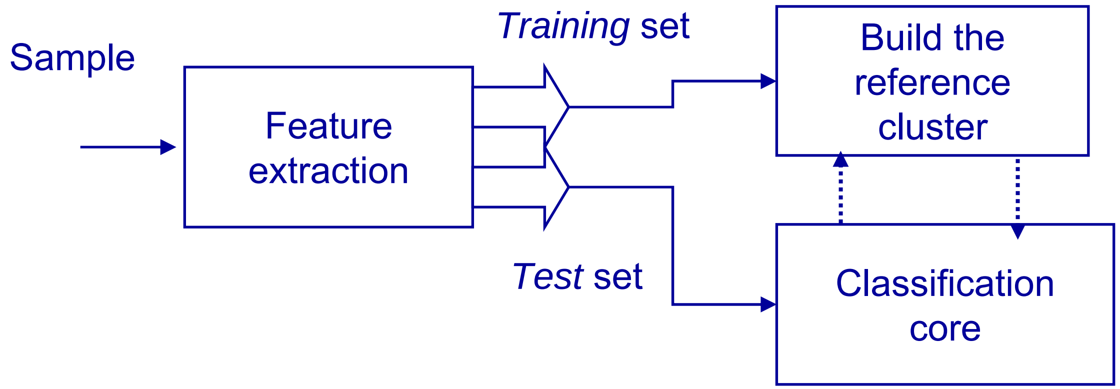


Building the FV



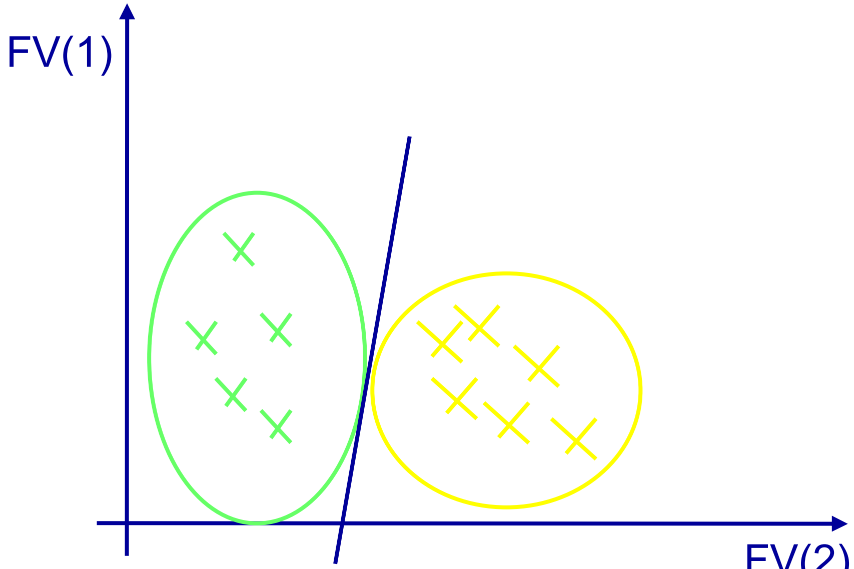
Implementation

- 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”

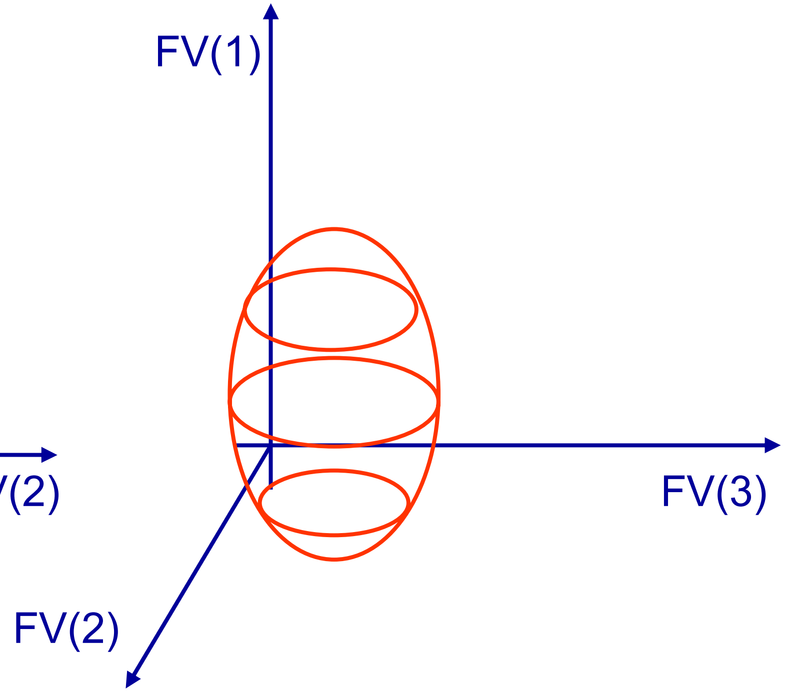


Clustering in the Feature Space

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

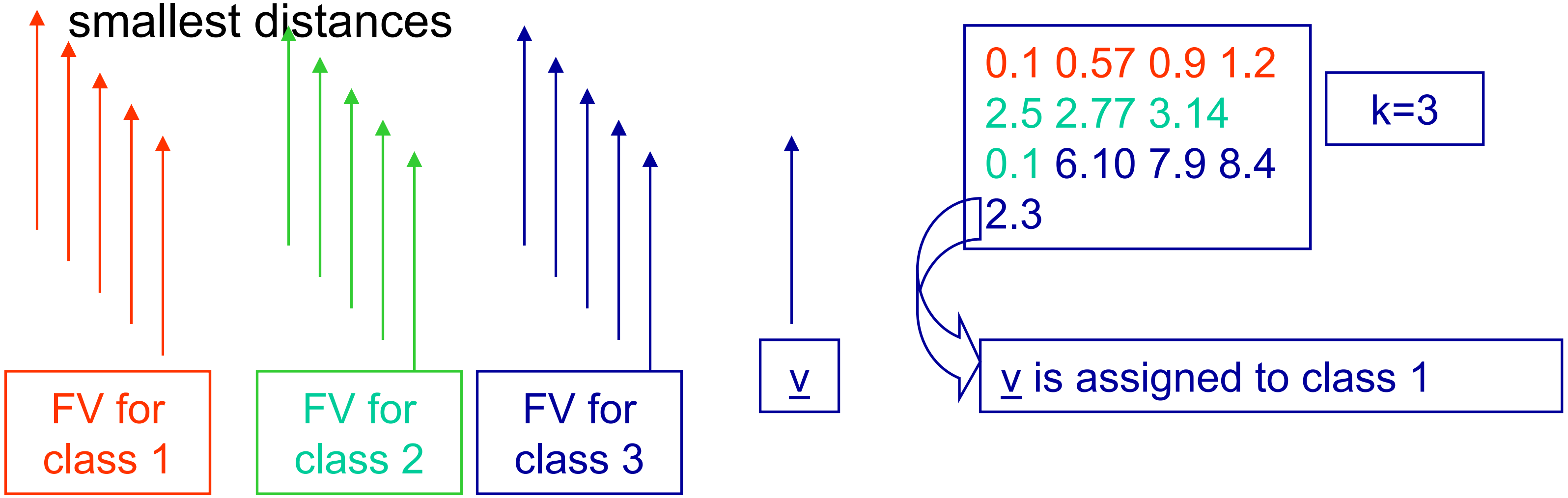
Classification algorithms

- 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 $\Rightarrow$ 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



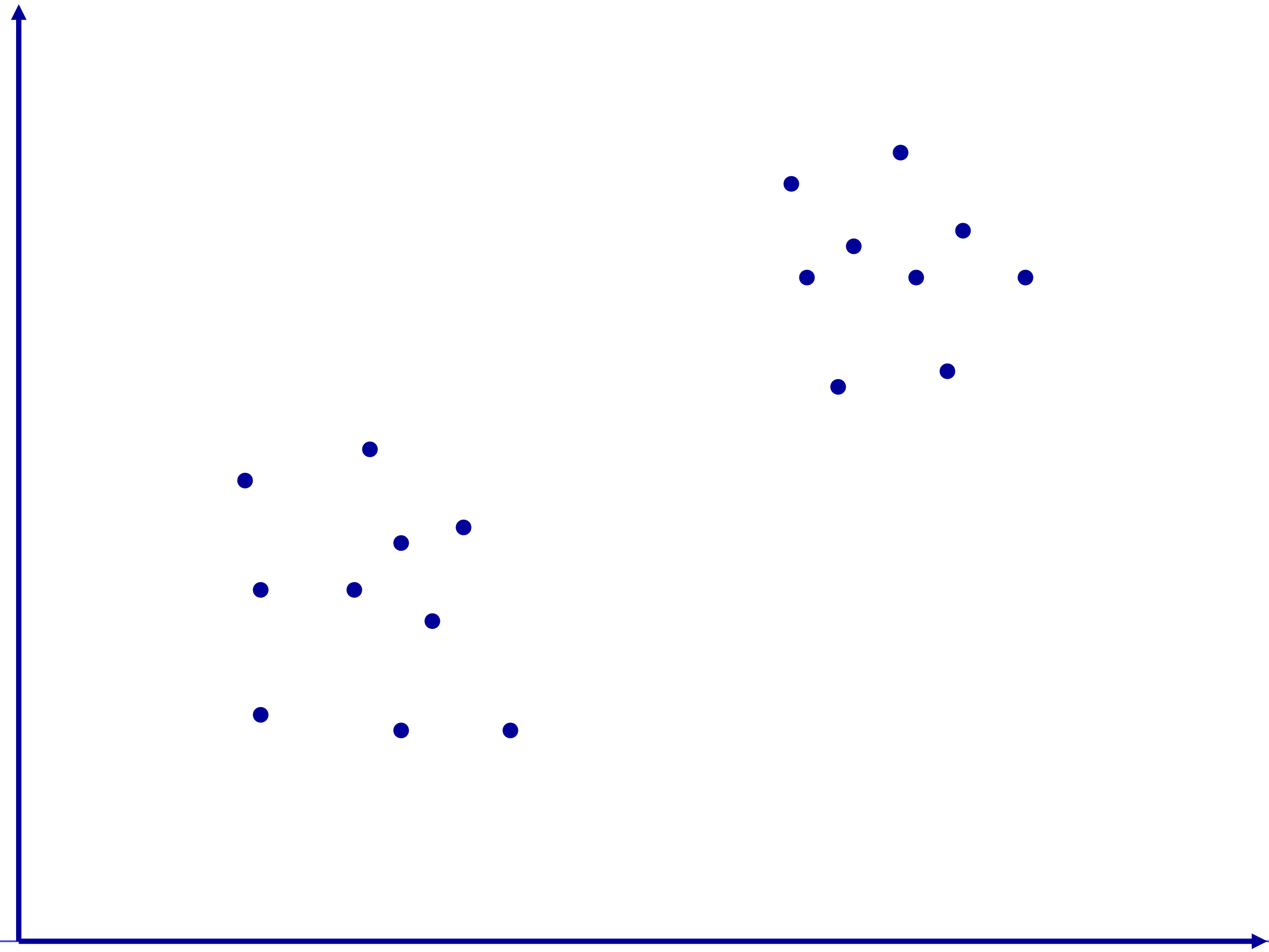
Confusion matrix

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%

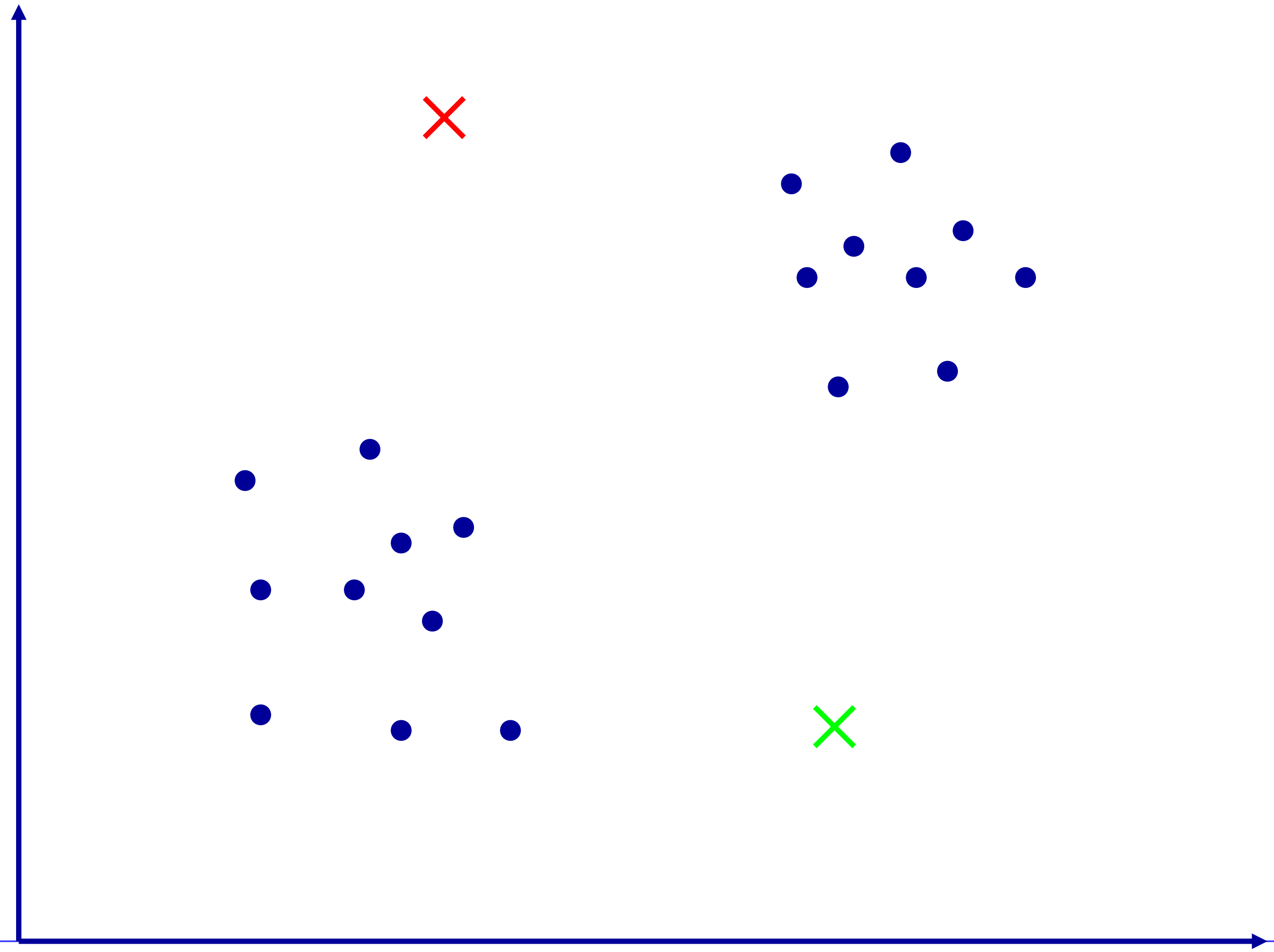
K-Means Clustering

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).

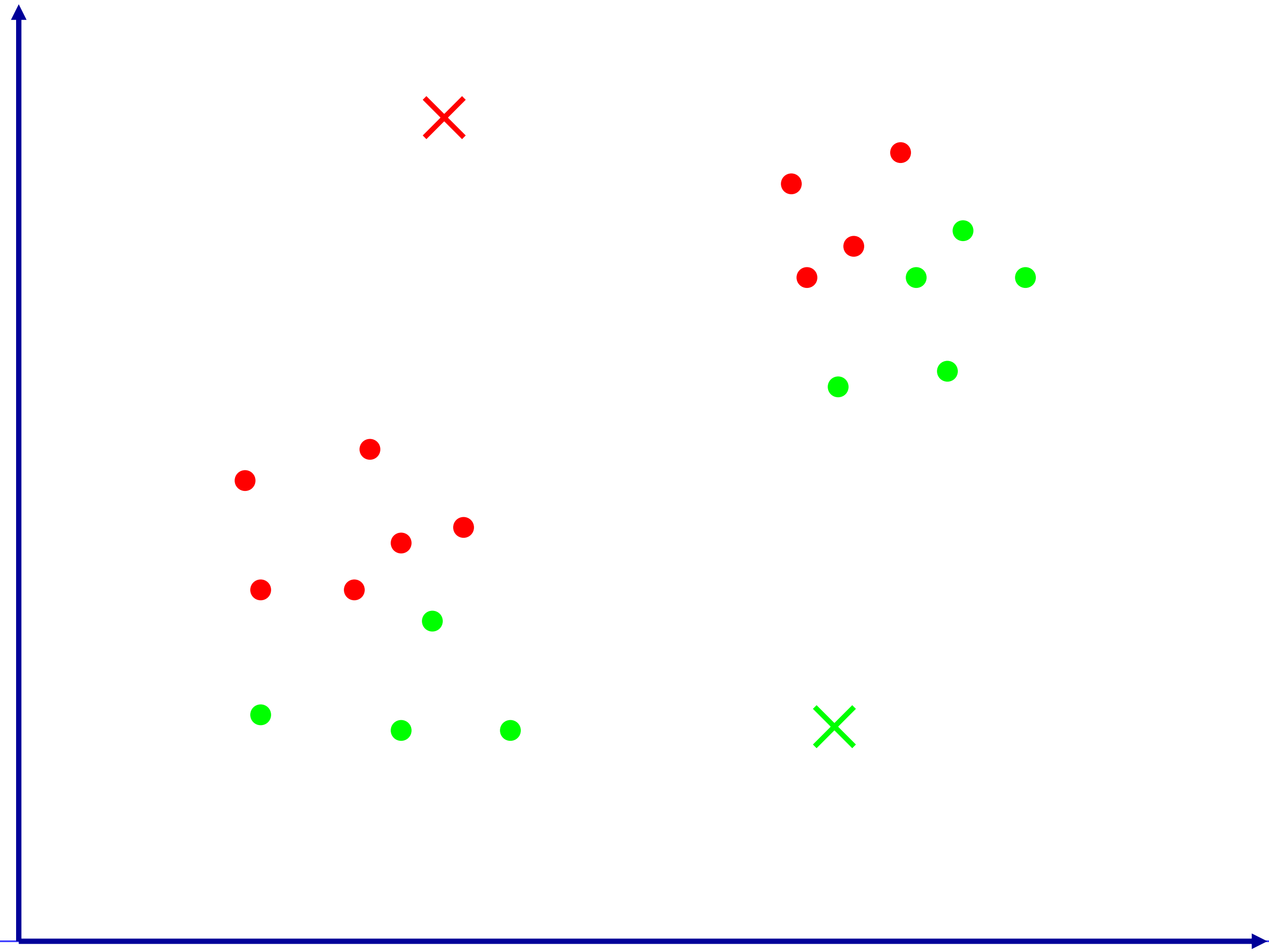
K-Means Clustering



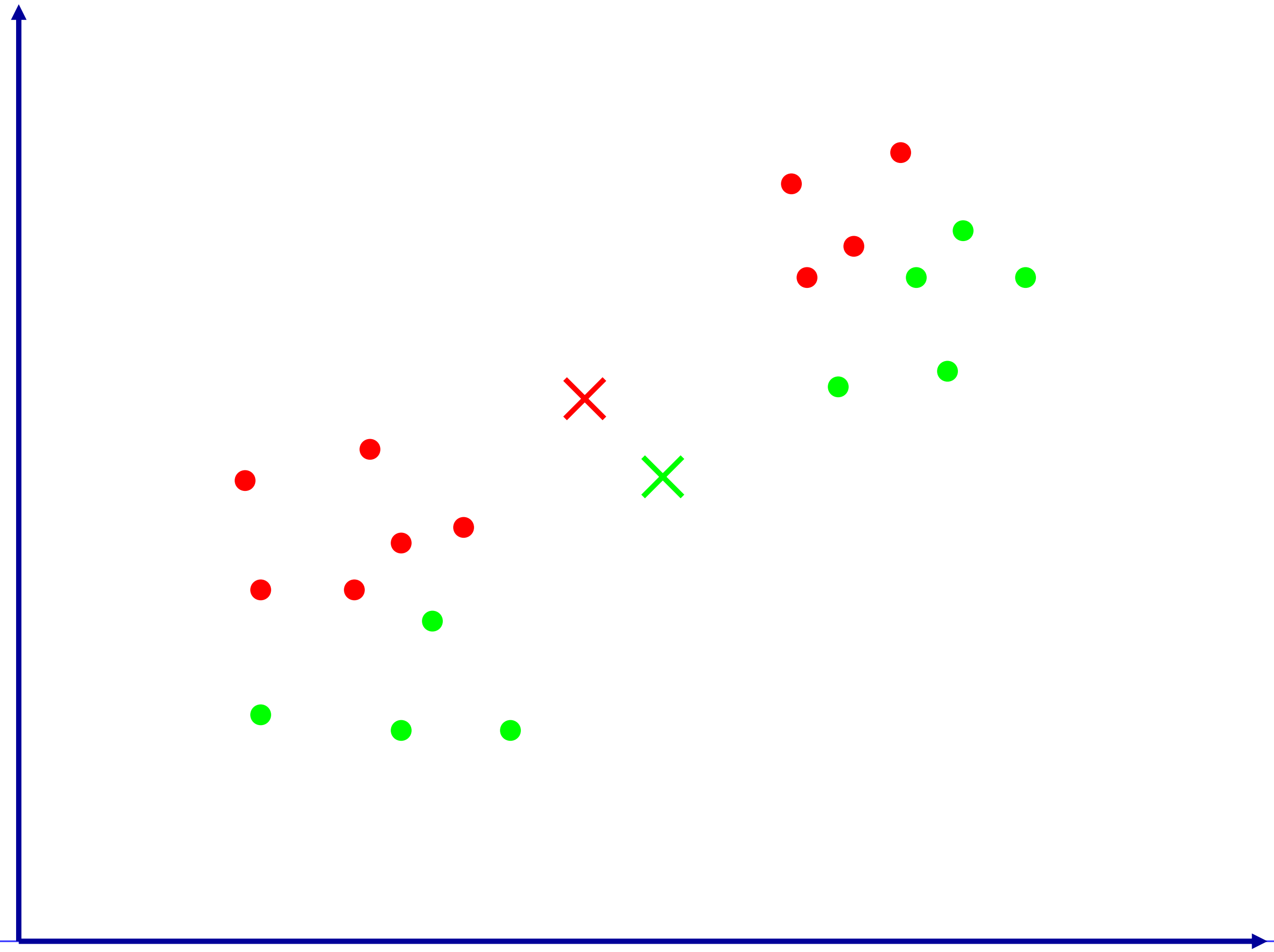
K-Means Clustering



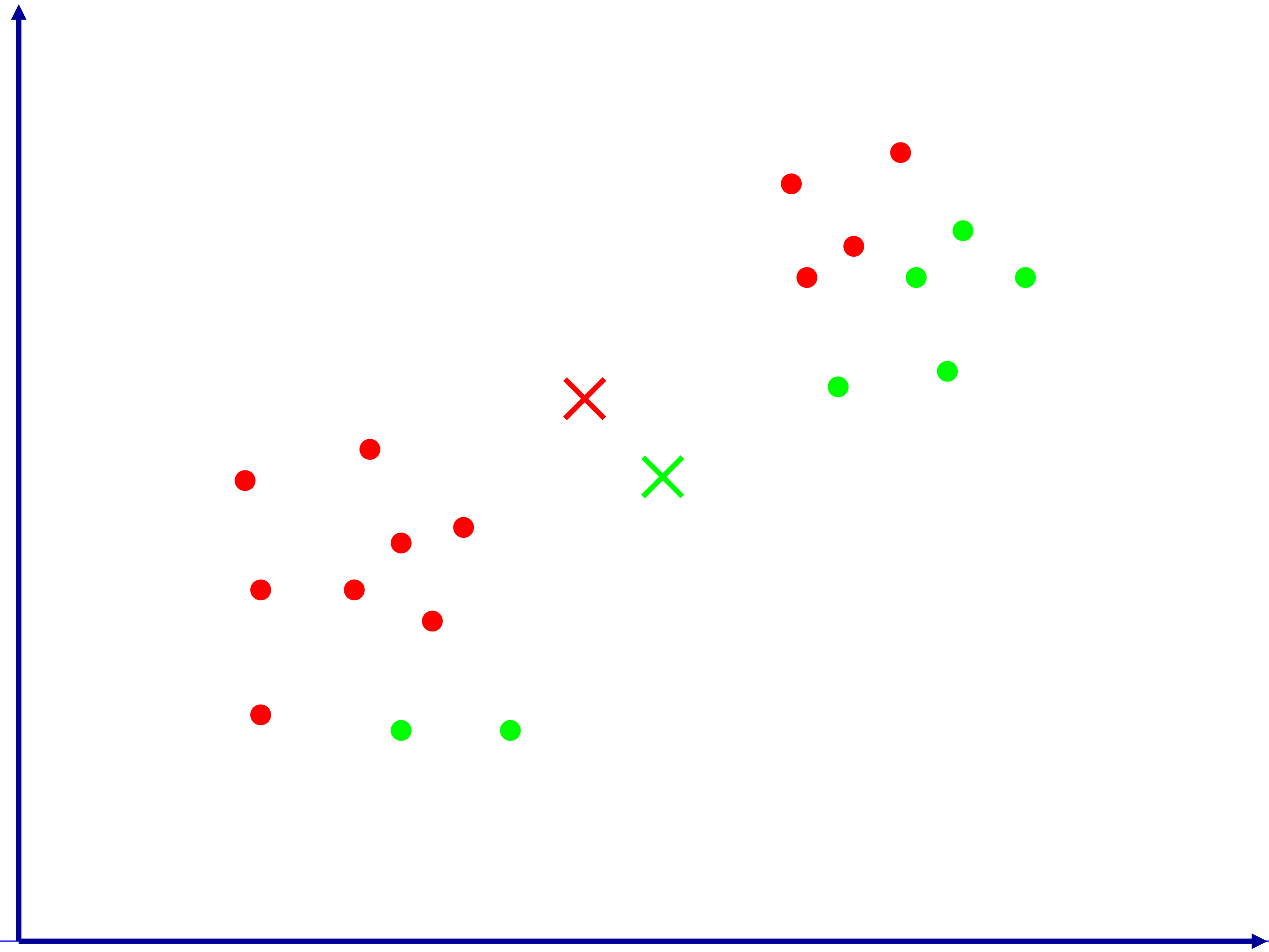
K-Means Clustering



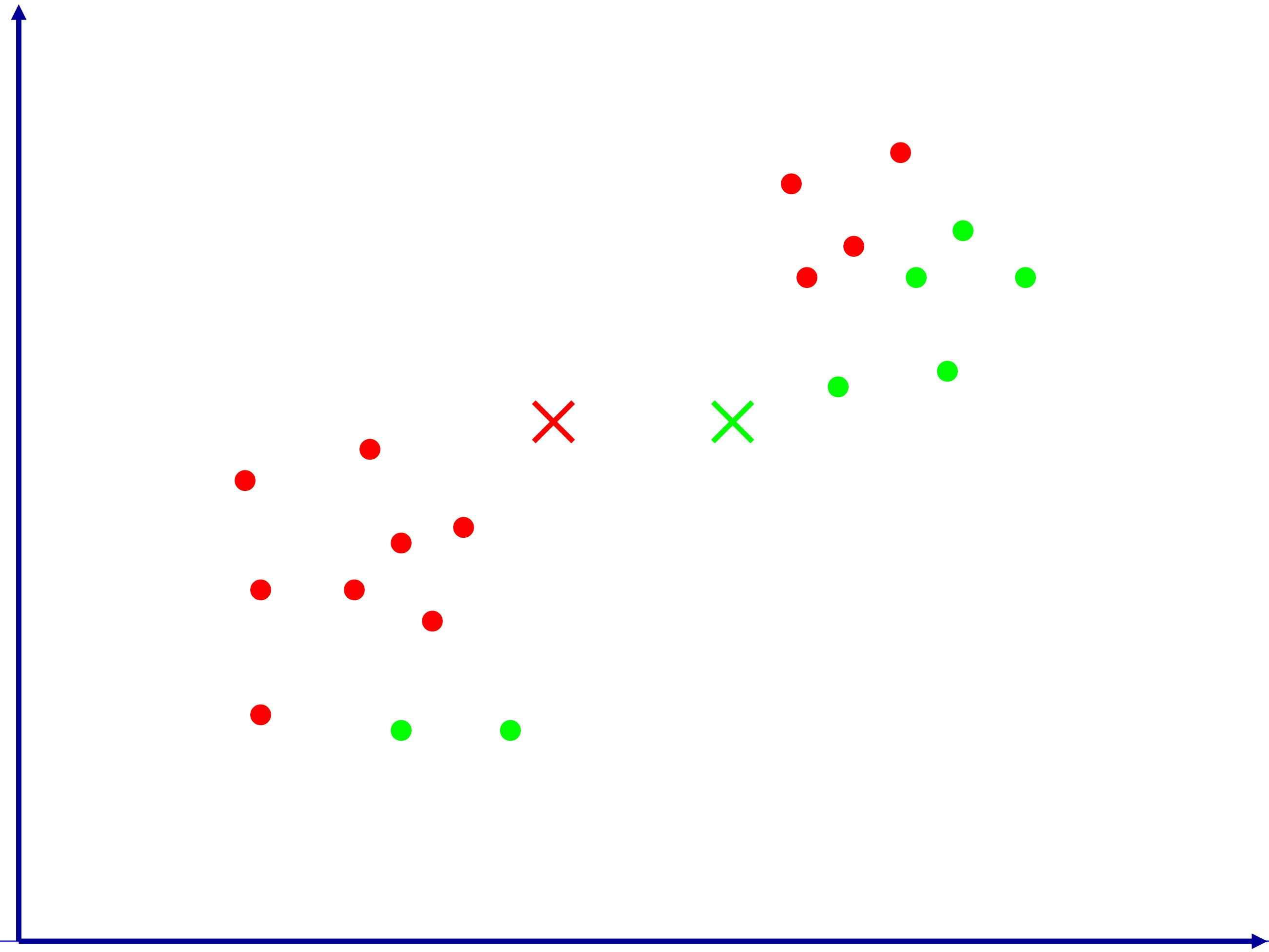
K-Means Clustering



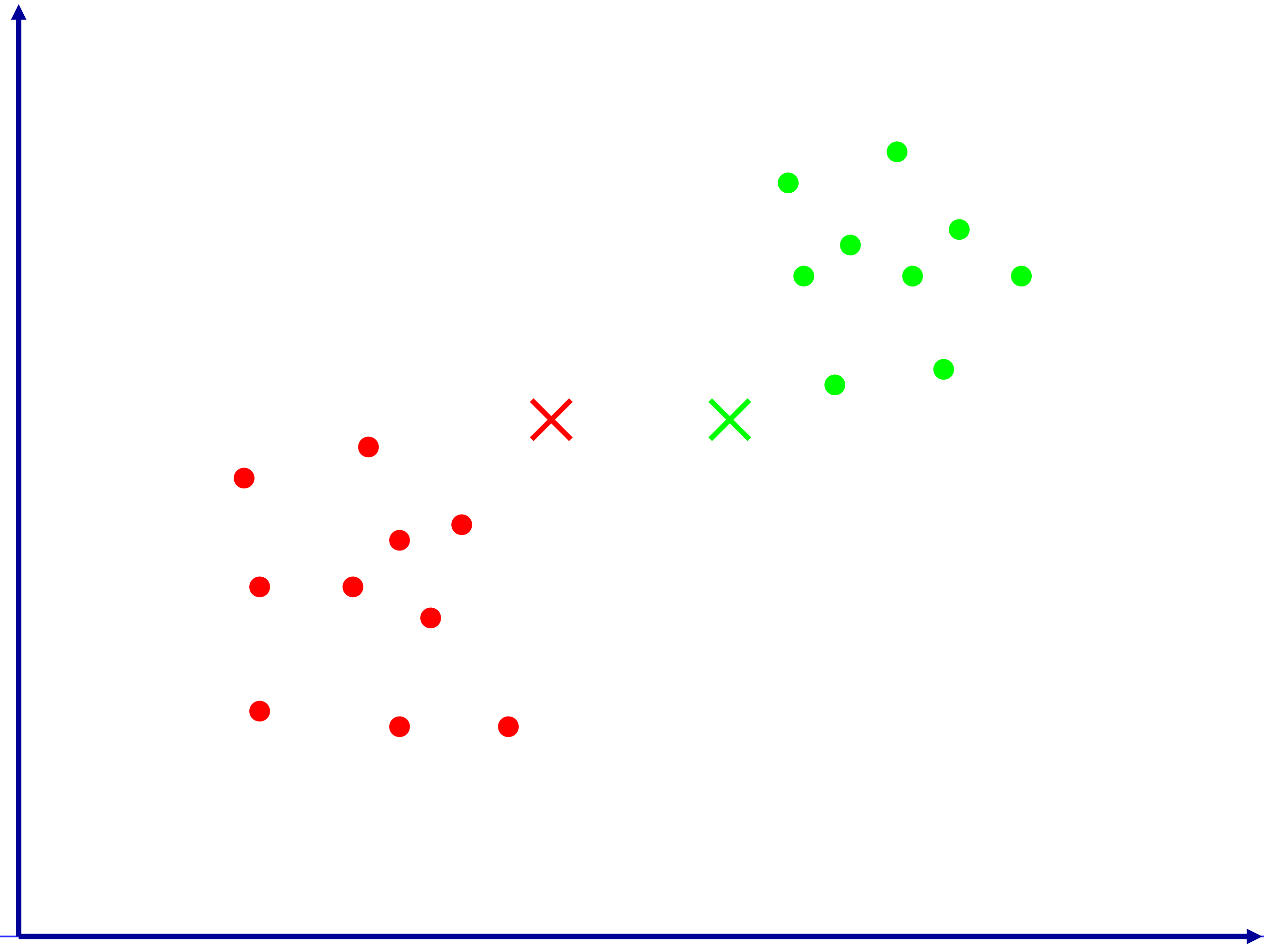
K-Means Clustering



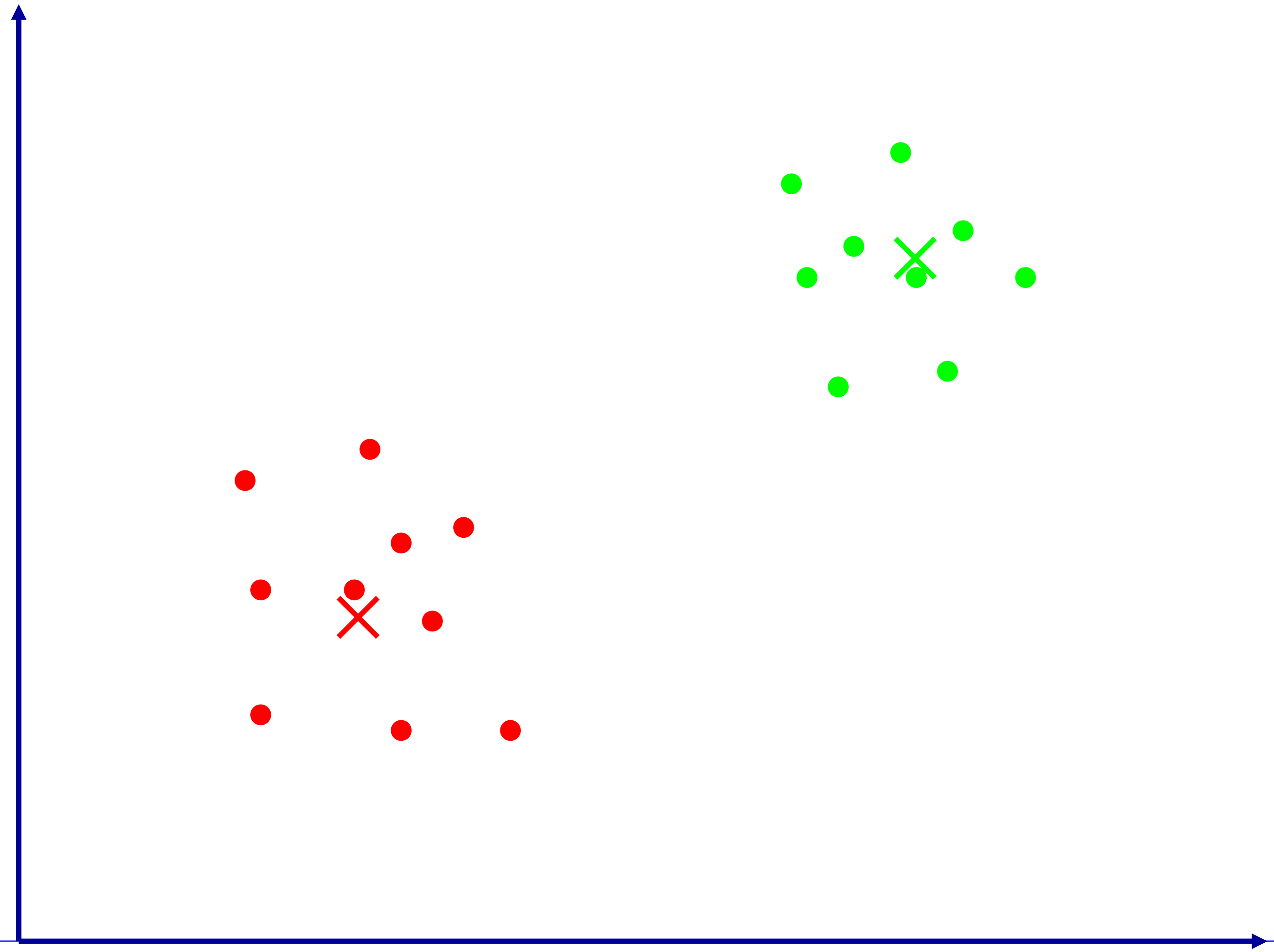
K-Means Clustering



K-Means Clustering

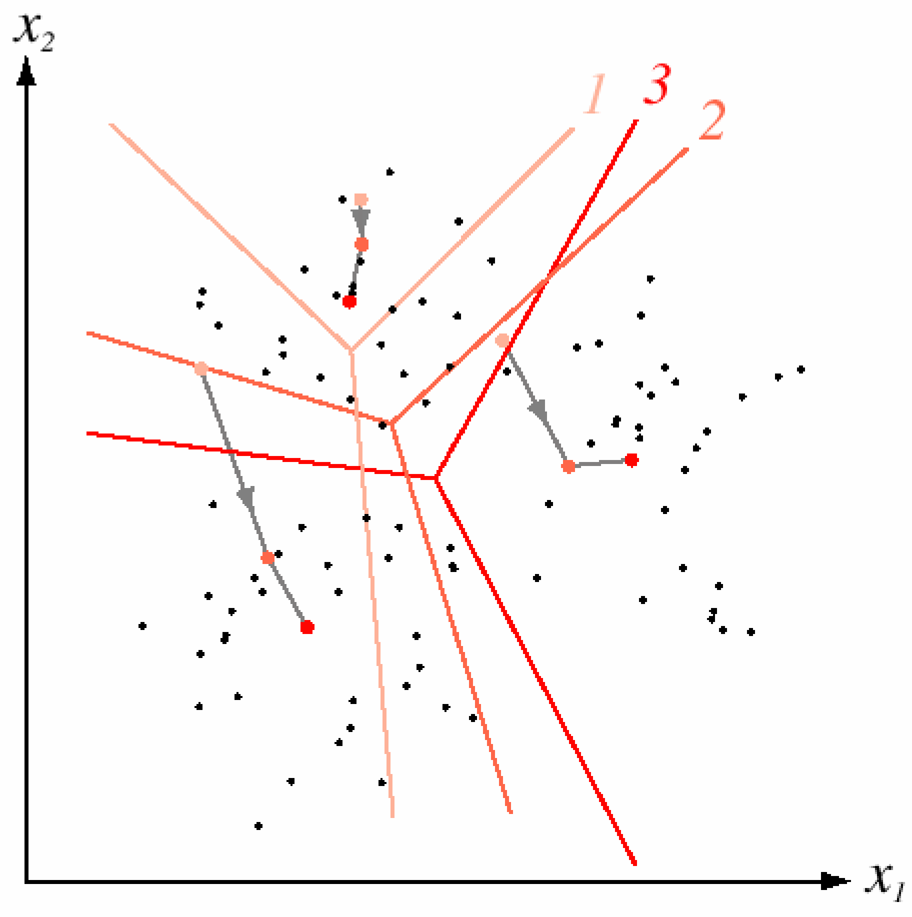


K-Means Clustering



K-Means Clustering

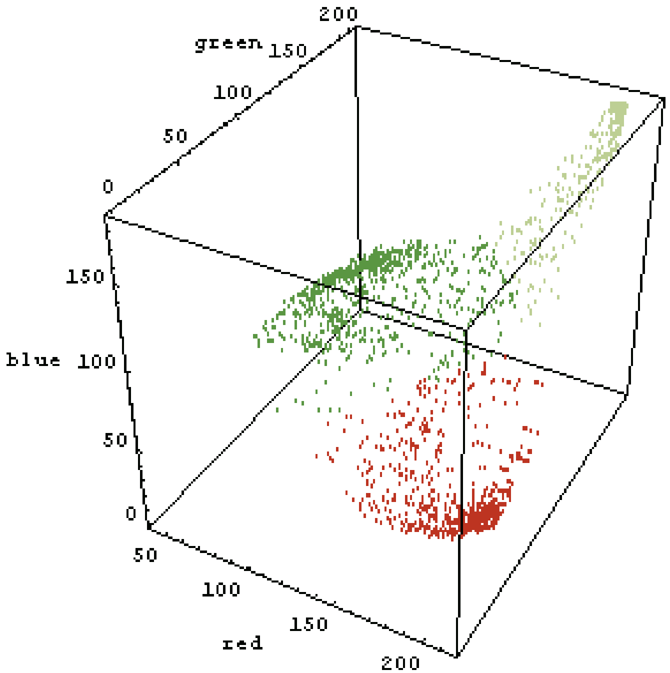
- Example



Duda et al.

K-Means Clustering

- RGB vector

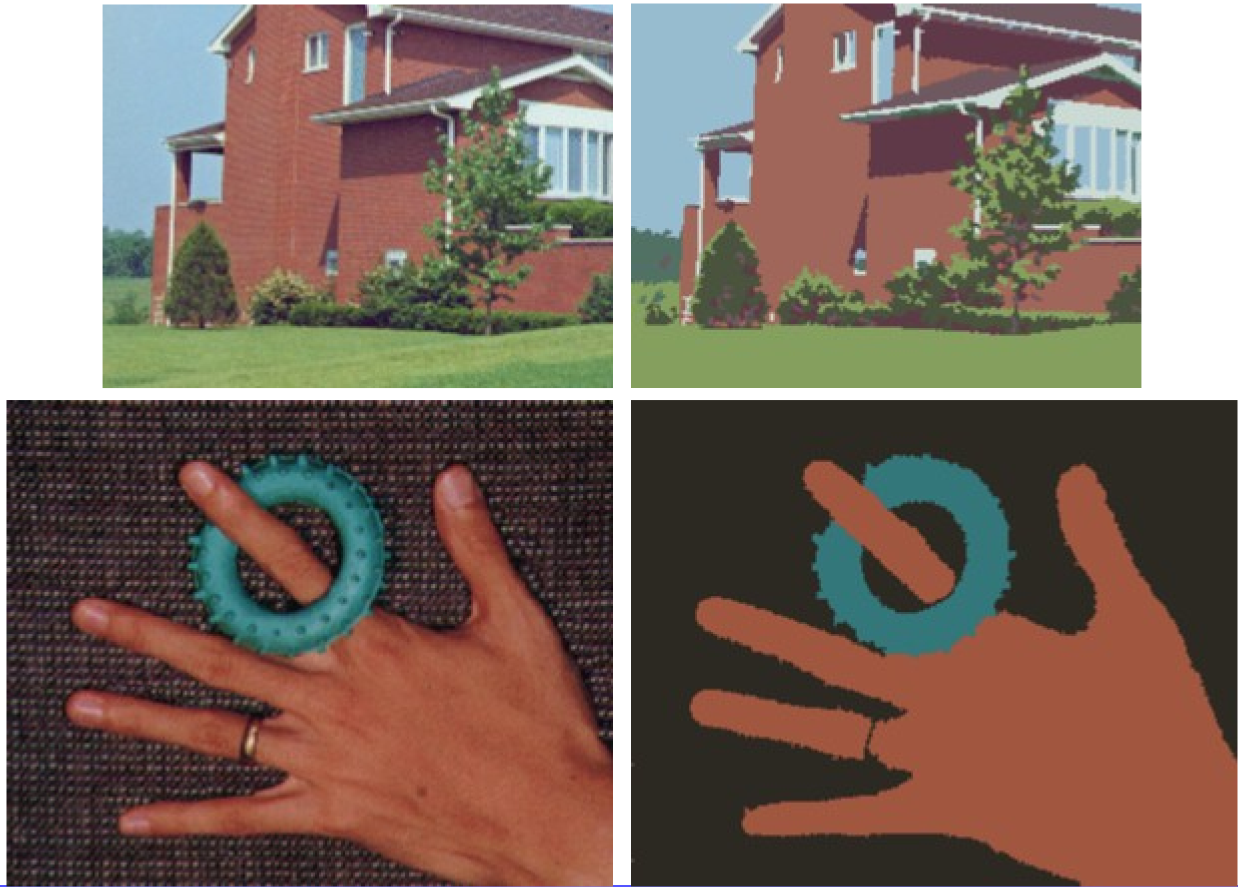


K-means clustering minimizes

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

Clustering

- Example



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

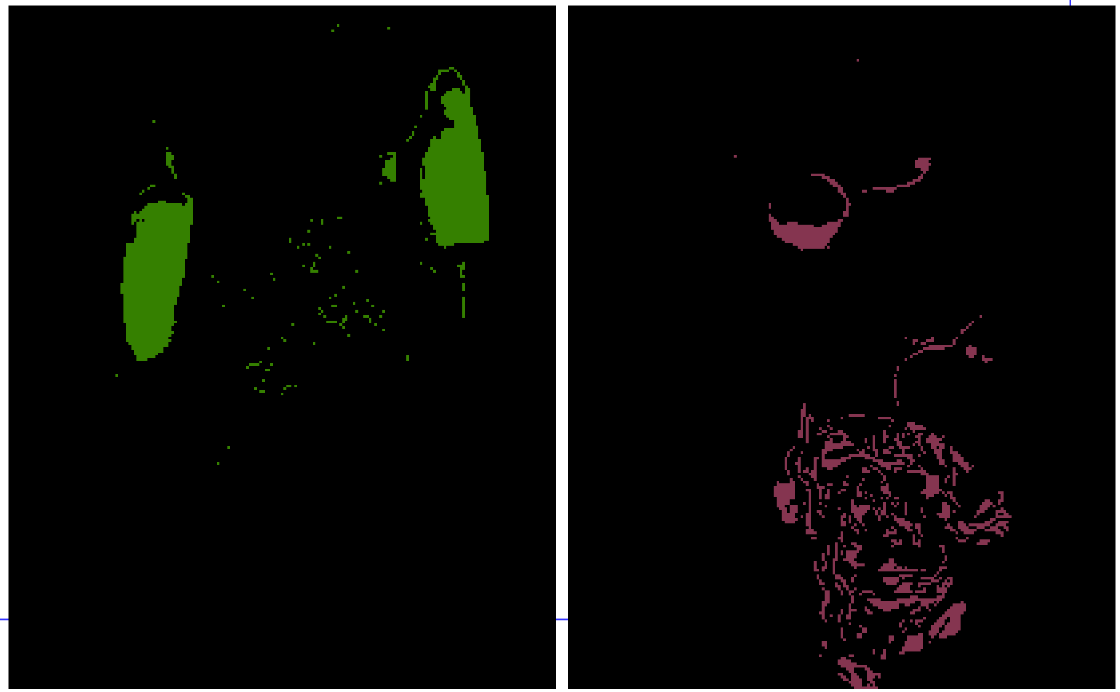
K-Means Clustering

- Example

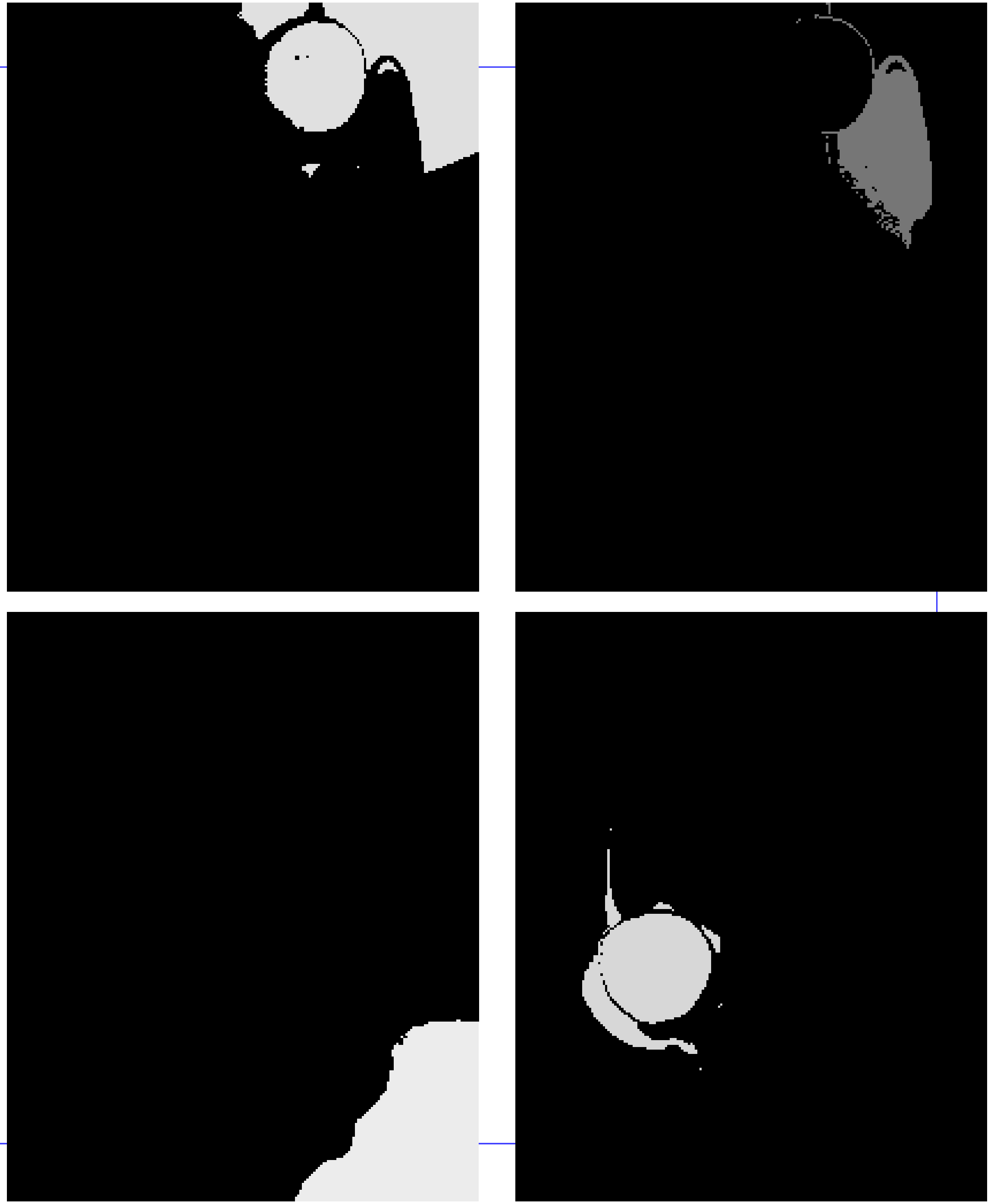
Original

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.