IMAGE ANALYSIS

Image analysis methods extract information from an image by using automatic or semiautomatic techniques termed: ***scene analysis, image description, image understanding, pattern recognition, computer/machine vision*** *etc.*).

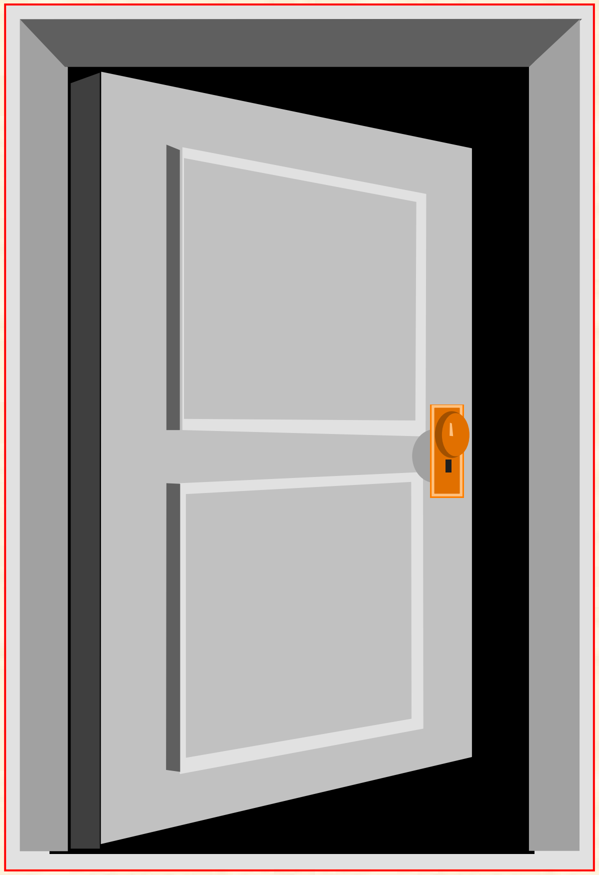
Image analysis differs from other types of image processing methods, such as ***enhancement*** or ***restoration*** in that the final result of image analysis procedures is a ***numerical output*** rather than a picture.

IMAGE ANALYSIS

Classification

Search the database for this fingerprint

Image understanding



Door open?

Image analysis steps

Preprocessing

Segmentation

Feature extraction

Classification and interpretation

database query

The best enemy spy!

Examples of Computer Vision Applications

⇒ **optical character recognition (OCR)**

- mail sorting, label reading, product billing, ban-check processing, text reading ,

⇒ **Medical image analysis**

- Tumor detection, size/shape measurement of organs, chromosome analysis, blood cell count ,…

⇒ **Industrial robotics**

- Recognition and interpretation of objects in a scene, motion control through visual feedback

⇒ **Cartography, Geology**

- Map making from photographs, plotting weather maps, oil exploration

⇒ **forensics (biometry) and military applications**

- fingerprint and face recognition, target detection and identification, guidance of helicopters and aircraft in landing, remote guiding of missiles

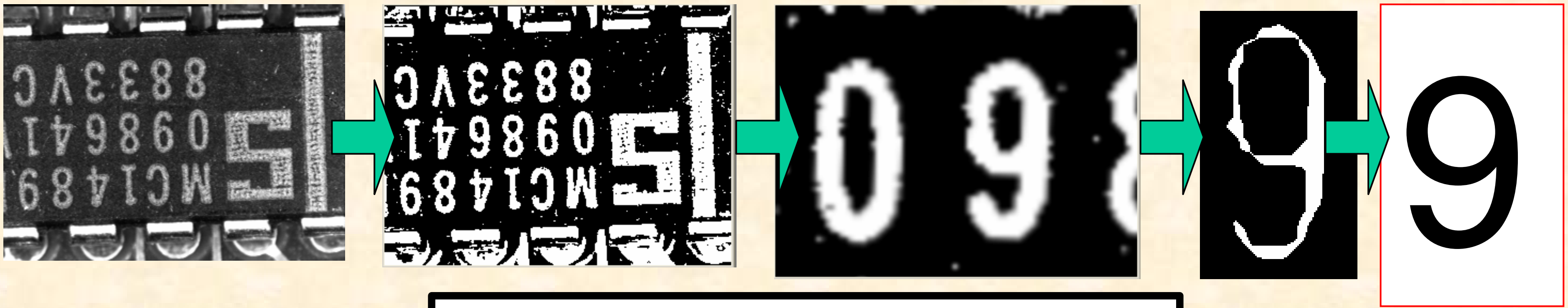


Image analysis techniques

1. **Segmentation**
   - Thresholding
   - Boundary based segm.
   - Region based segm.
   - Template matching
   - Texture segmentation

2. **Feature description**
   - Spatial features
   - Transform features
   - Edges and boundaries
   - Shape features
   - Moments
   - Texture

3. **Classification**
   - Clustering
   - Statistical classif.
   - Decision trees
   - Neural networks
   - Similarity measures

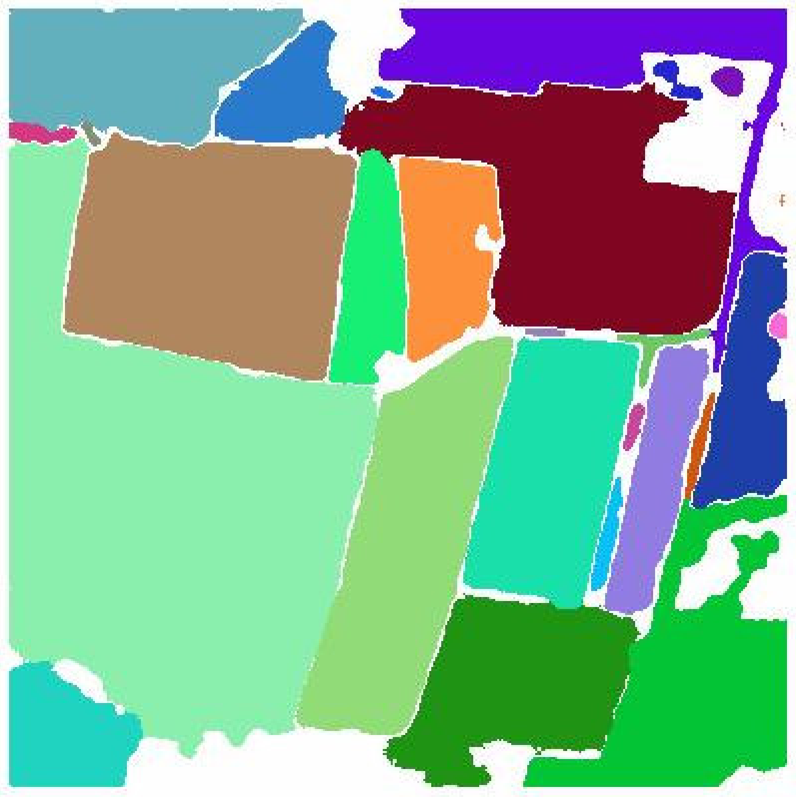
Segmentation

Image segmentation is a key step in image analysis.

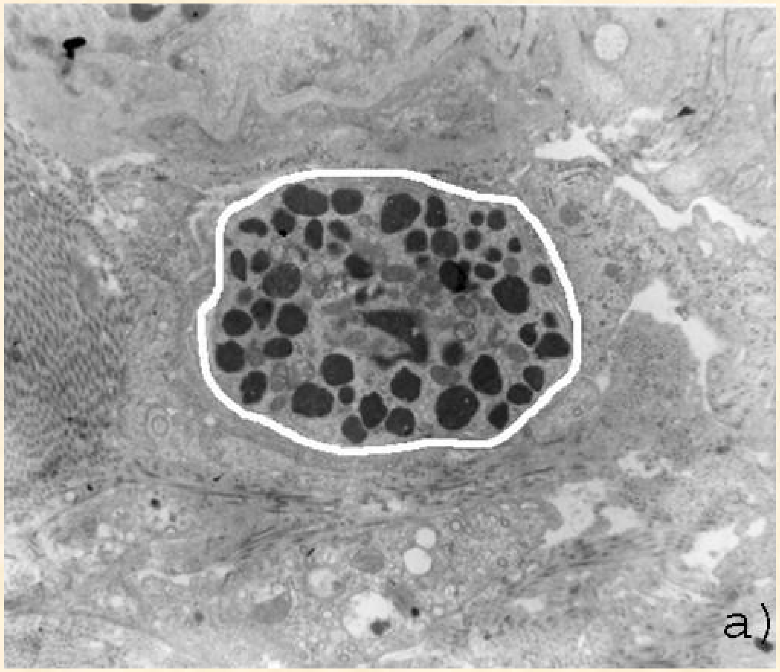
Segmentation subdivides an image into its components. It distinguishes objects of interest from background, e.g. Optical Character Recognition (OCR) systems first segment character shapes from an image before they start to recognise them.

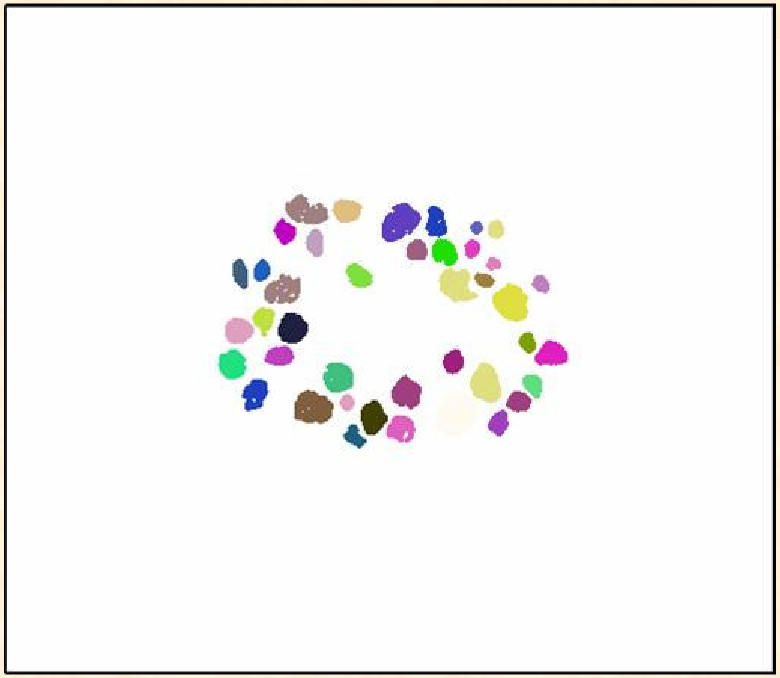
The **segmentation** operation only **subdivides an image**; it **does not attempt to recognise** the segmented image parts.

Aerial photos



Microscope image of cells





Thresholding

Amplitude **thresholding** (i.e. in the brightness domain) is the basis approach to image segmentation.

A threshold T is selected a that would separate the two modes, i.e. any image point for which *f(x,y)>T* is considered as an **object**; otherwise, the point is called a **background** point.

The thresholded image (binary image) is defined by:

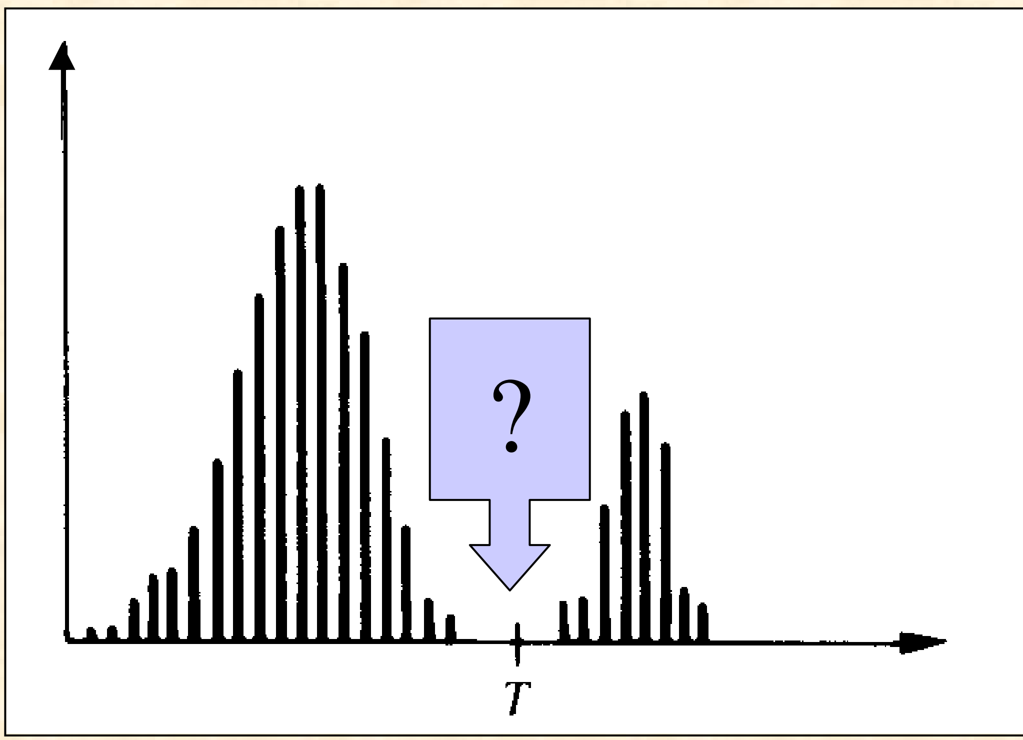
$$g(x,y)=\begin{cases}1 & for\ \ f(x,y)\geq T\\0 & for\ \ f(x,y)<T\end{cases}$$

object points

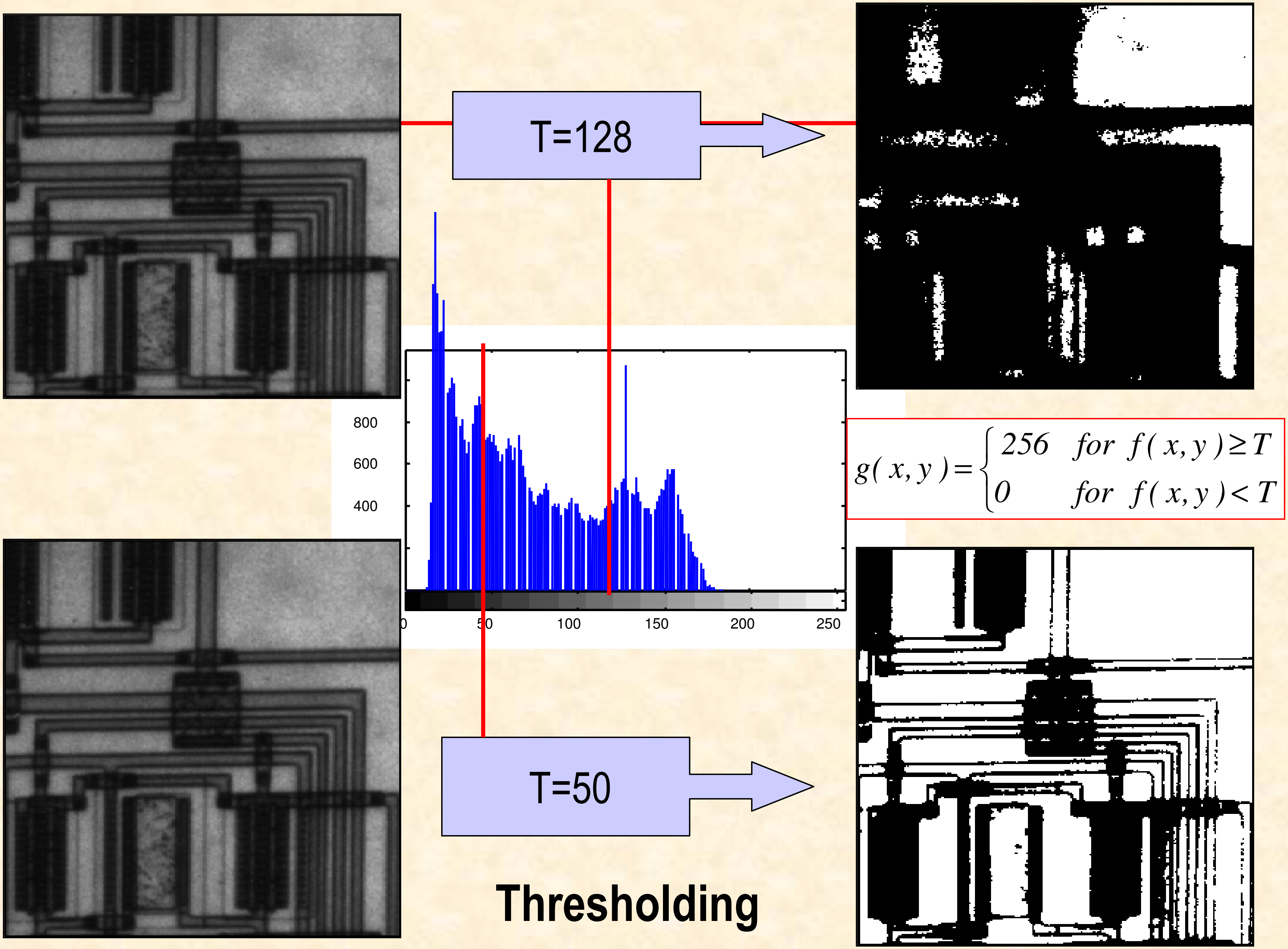
background points

Thresholding

Suppose an image *f(x,y)* contains bright objects surrounded by dark background and its gray-level histogram is shown in the figure.



the threshold is identified on the basis of image histogram



$$g(x,y)=\begin{cases}256 & \text{for } f(x,y)\geq T\\ 0 & \text{for } f(x,y)<T\end{cases}$$

Thresholding

DEMO MATLAB

DEMO MATLAB

```
%MATLAB
x=imread('cameraman.tif'); figure(1), imshow(x);
bw=im2bw(x,0.5); figure(2), imshow(bw)
```

Thresholding

When *T* is set on the basis of the entire image *f(x,y)* the threshold is called ***global***.

If *T* depends on spatial coordinates *(x,y)* the threshold is termed ***dynamic***.

If *T* depends on both *f(x,y)* and some local image property *p(x,y)* - e.g., the average gray level in a neighbourhood centred on *(x,y),* the threshold is called ***local*** and *T* is set according to a test function:

$$T = T\left[p(x,y), f(x,y)\right]$$

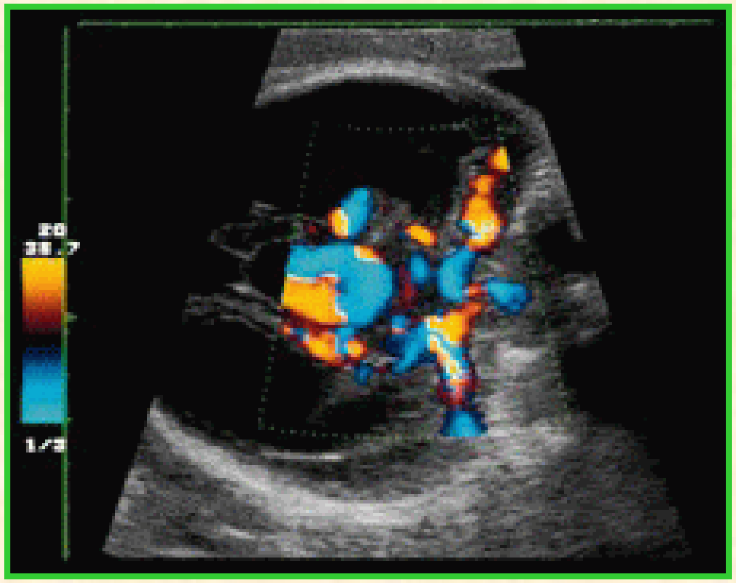
Thresholding

In the case of **multilevel thresholding** the threshold becomes a vector $T = [T_1, T_2, \ldots T_N]$ and an image is partitioned into $N+1$ sub-regions, e.g. for two-level thresholding:

R1: if $f(x,y) \leq T_1$,   R2: if $T_1 < f(x,y) \leq T_2$,   R3: if $f(x,y) > T_2$.

Image thresholding can be extended into more than one dimensions.

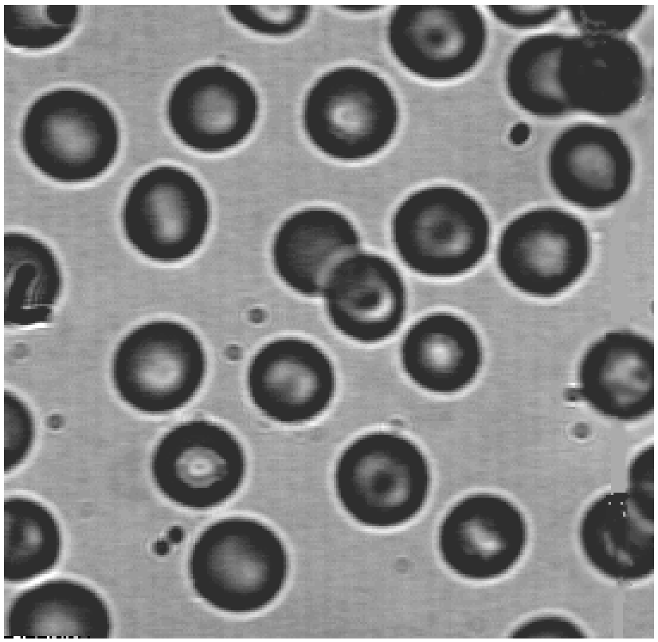
This may apply to **multi-band thresholding of colour images** in any colour coordinates, e.g. RGB or HIS.

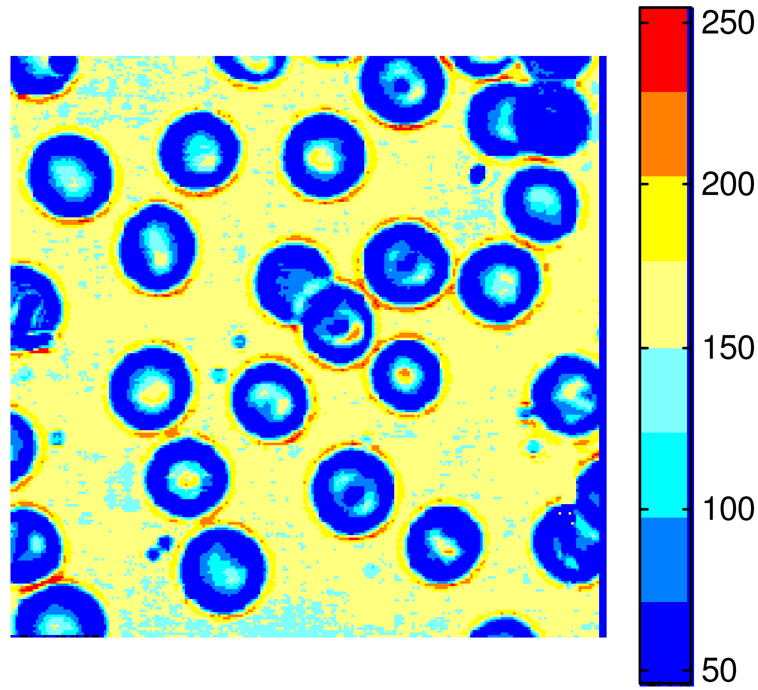


DEMO MATLAB

```
%MATLAB
x=imread('cameraman.tif'); figure(1), imshow(x);
bw=im2bw(x,0.5); figure(2), imshow(bw)
```

Multilevel thresholding -example



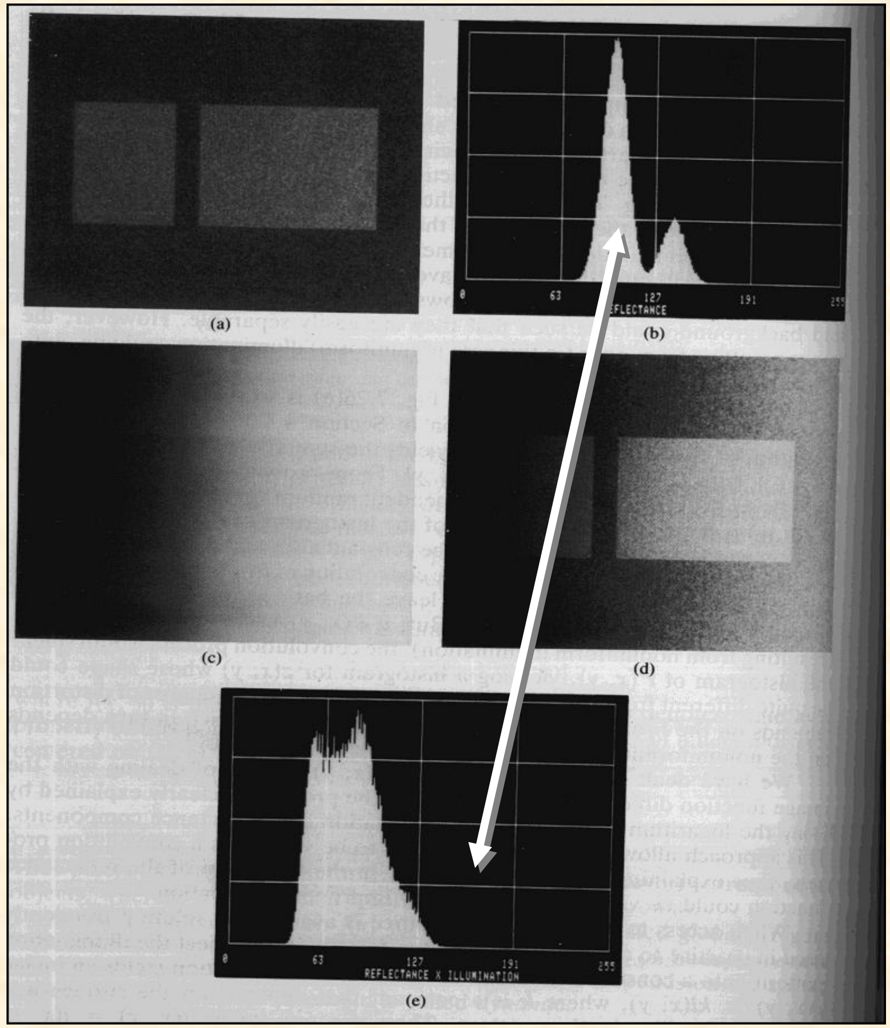



```
x=imread('blood1.tif');
figure(1), imshow(x);
figure(2),imshow(x), colormap(jet(16))
```

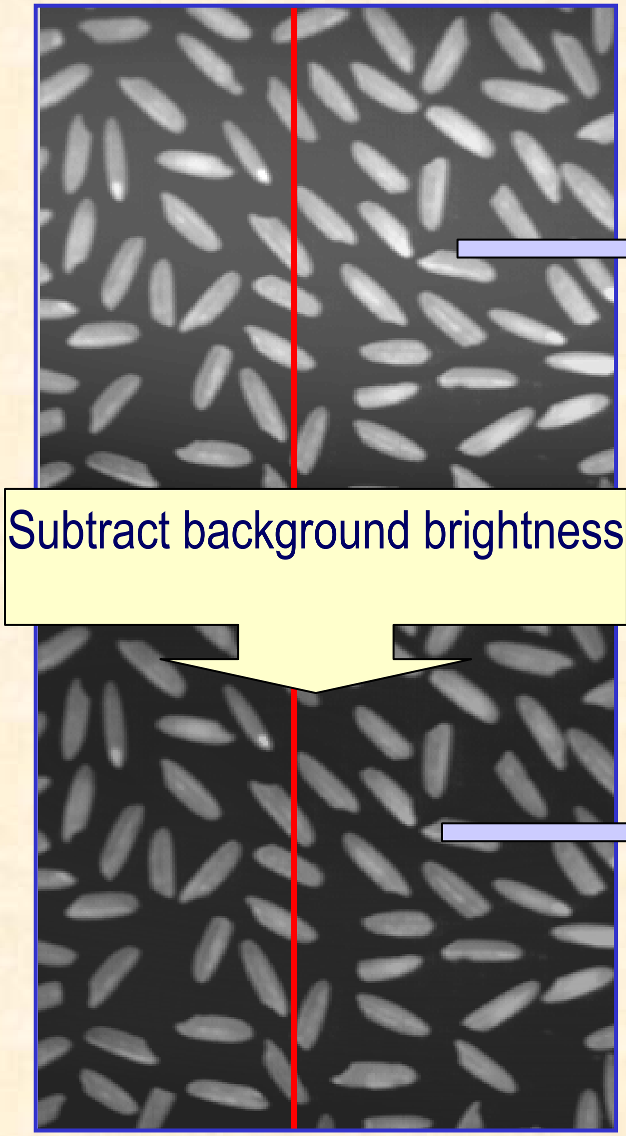
Nonuniform scene illumination

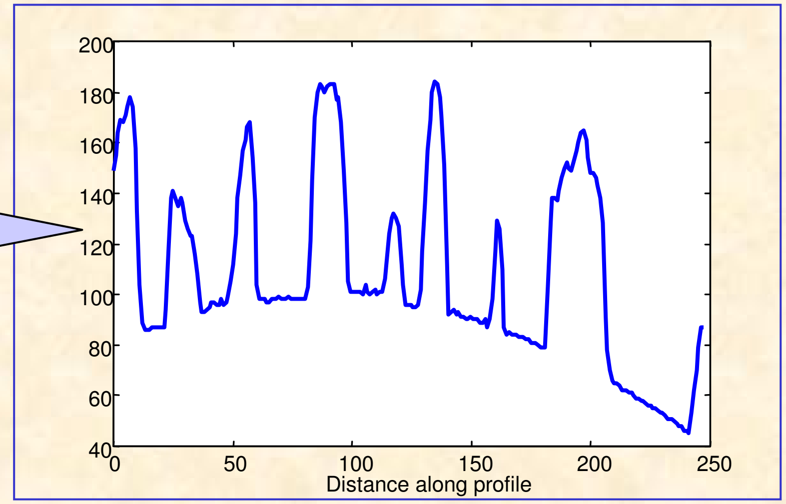
Uniform scene illumination

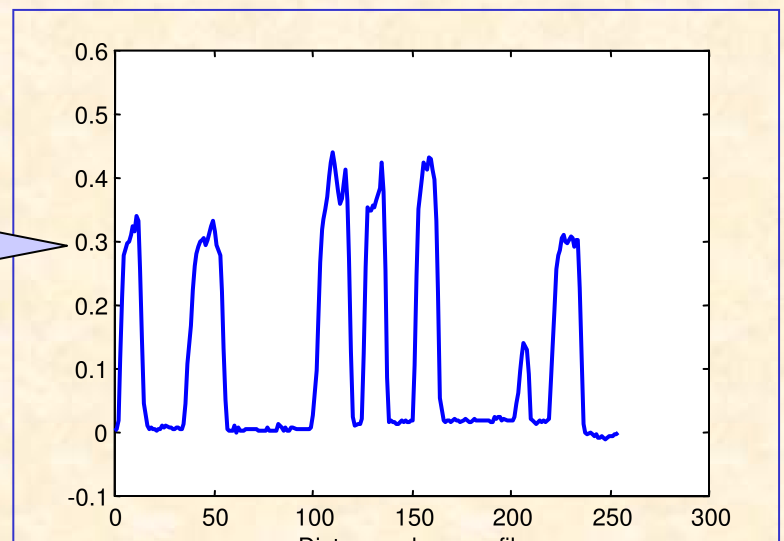
Nonuniform scene illumination



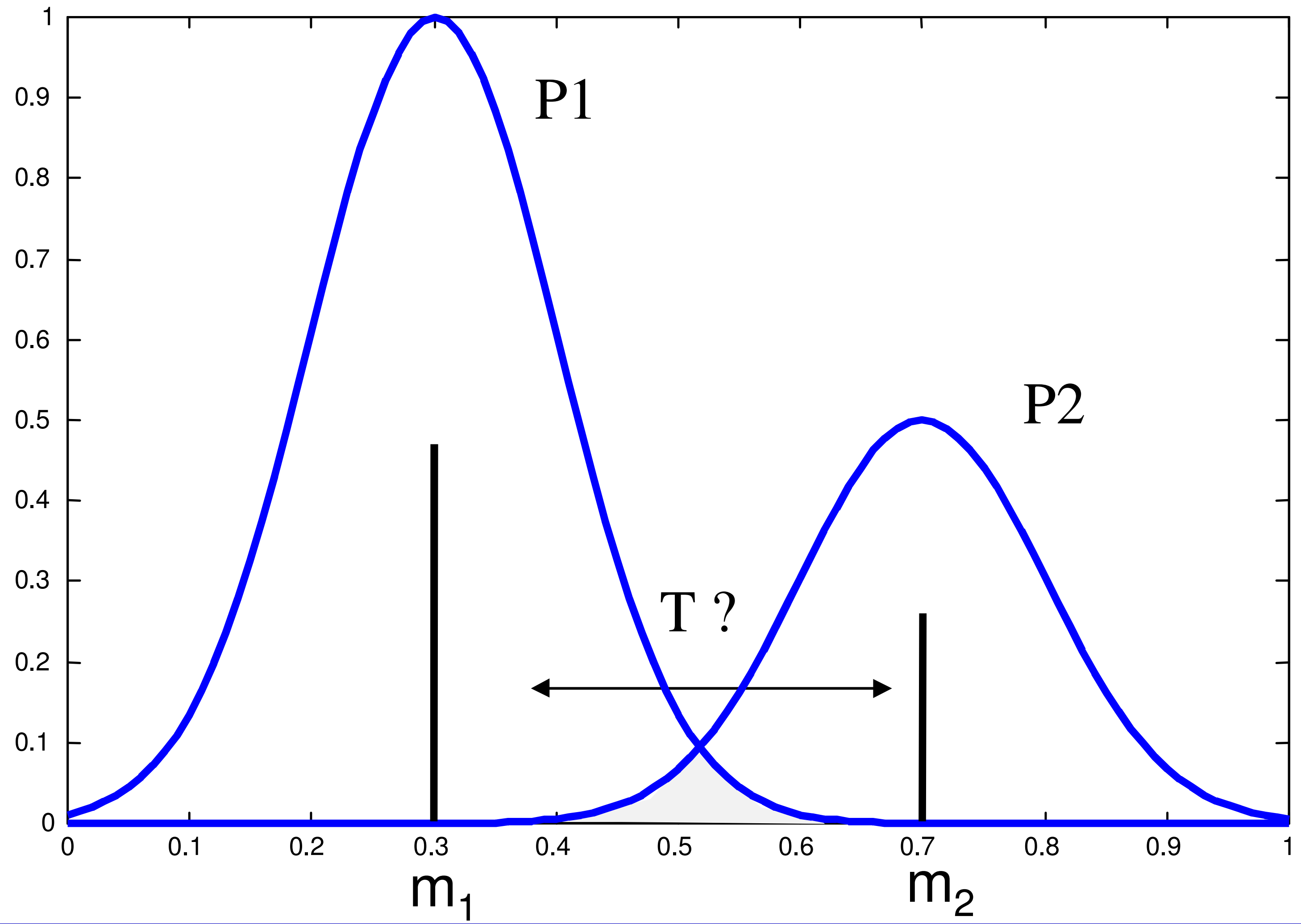
Correction of monuniform illumination







Thresholding - revisited



Optimum Threshold

Suppose an image contains two intensity values m_1, m_2 combined with additive Gaussian noise $N(0,\sigma)$ that appear in an image with apriori probabilities P_1 and P_2 correspondingly $P_1 + P_2=1$.

The task is to define a threshold level T that would minimise the overall segmentation error

Optimum Threshold

For $\sigma_1, \ \sigma_2$ solution to this optimisation problem is the following:

$$T = \frac{m_1 + m_2}{2} + \frac{\sigma^2}{m_1 - m_2} \ln\left(\frac{P_2}{P_1}\right)$$

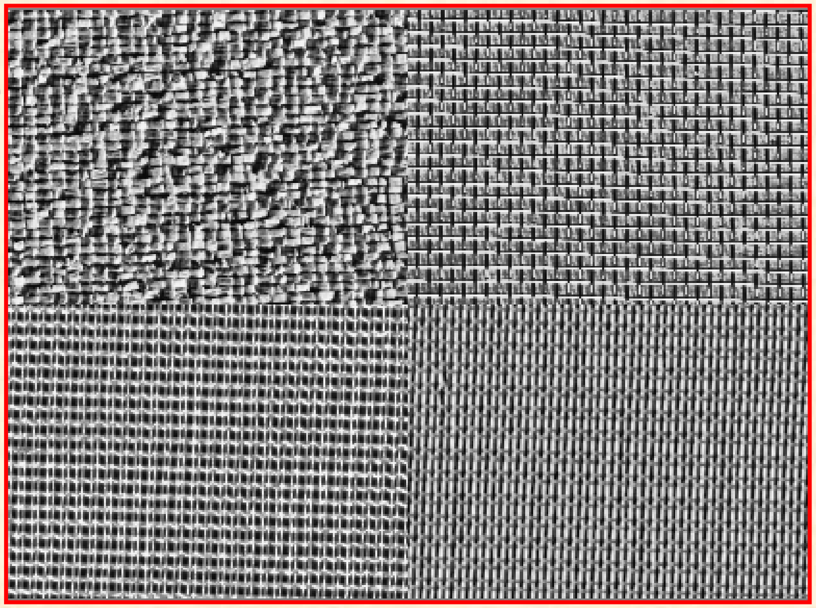
If $P_1 = P_2$, the optimal threshold is the average of the image intensities.

Region-oriented segmentation

The main idea in ***region-based*** segmentation techniques is to identify different regions in an image that have similar features (gray level, colour, texture, etc.).

There are two main region-based image segmentation techniques:

- ***region growing (merging)***
- ***region splitting*** **.**



General formulation

Let R represent the entire image. Segmentation may be viewed as a process that partitions R into N disjoint regions, $R_1, R_1, \ldots, R_N$, such that:

a) $\sum_{i=1}^{N} R_i = R$

b) R_i is a connected region, $i = 1, 2, \ldots N$,

c) $R_i \cap R_j = \varnothing$ for all i and j, $i \neq j$,

d) $P(R_i) = TRUE$ for $i = 1, 2, \ldots N$,

e) $P(R_i \cup R_j) = FALSE$ *for* $i \neq j$

where $P(R_i)$ is a logical predicate over the points in set R_i and $\varnothing$ is the empty set.

General formulation

Condition **a)** indicates that the segmentation must be complete (all points assigned a region).

Condition **c)** indicates that the regions must be disjoint. Condition **d)** states that all pixels in a segmented region must satisfy the assumed predicate.

Condition **e)** indicates that distinct regions must be different according to predicate *P*. An example predicate:

$$\left| z(x, y) - m_i \right| \leq 2\sigma_i$$

where: $z(x,y)$ is a gray level in (x,y) coordinate, m_i, σ_i are the mean and standard variation of the analysed region.

Region growing

In region growing the image is divided into ***atomic regions*** (e.g., pixels, templates). These “**seed**” points grow by appending to each point other points that have similar properties. The key problem lies in selecting proper criterion (predicate) for merging. A frequently used merging rule is:

> *Merge two regions if they are „similar”*
> *(in terms of a predefined features)*

Similarity measures

Popular similarity measures are:

- **dot product** (projection of one vector onto direction of the other):

$$\left\langle x_i^T x_j \right\rangle = \|x_i\| \|x_j\| cos(x_i, x_j)$$

- and **Euclidean distance**:

$$d(x_i, x_j) = \sqrt{\sum_k (x_i(k) - x_j(k))^2}$$

Region growing

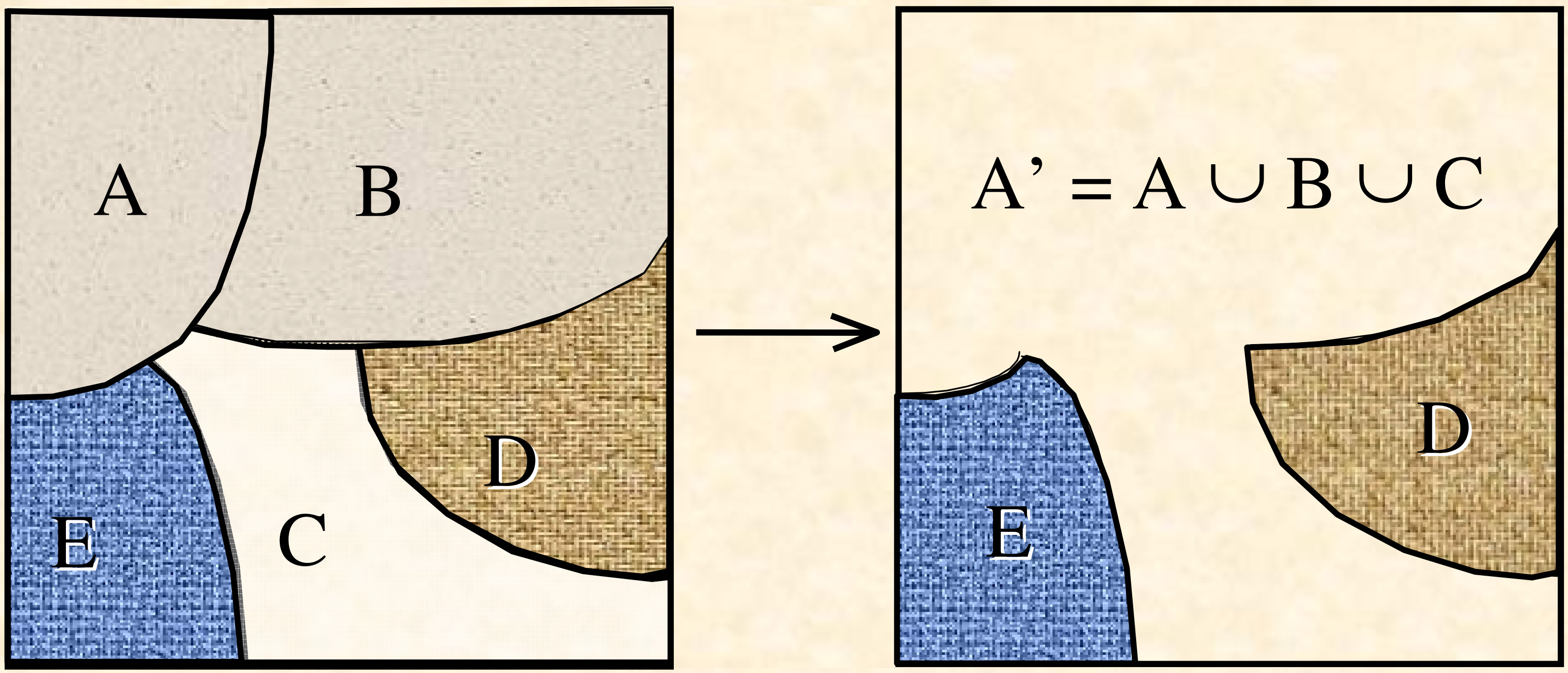
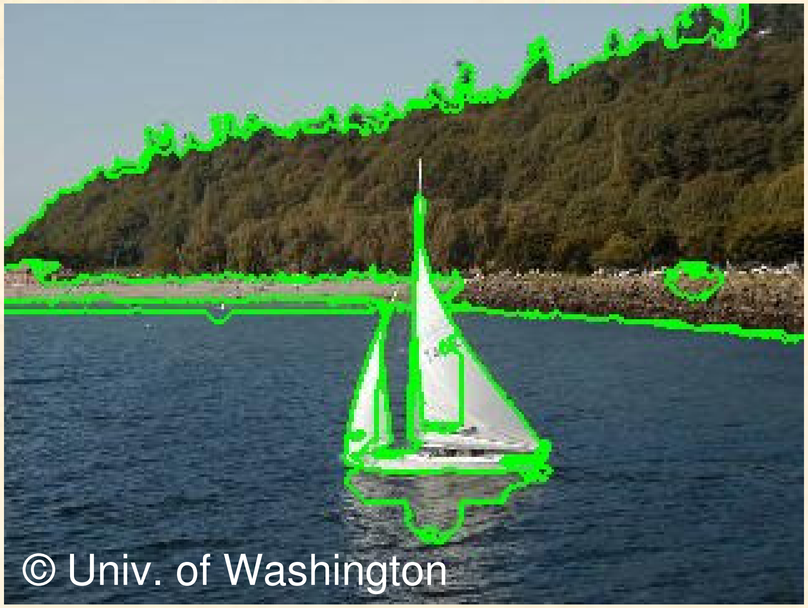
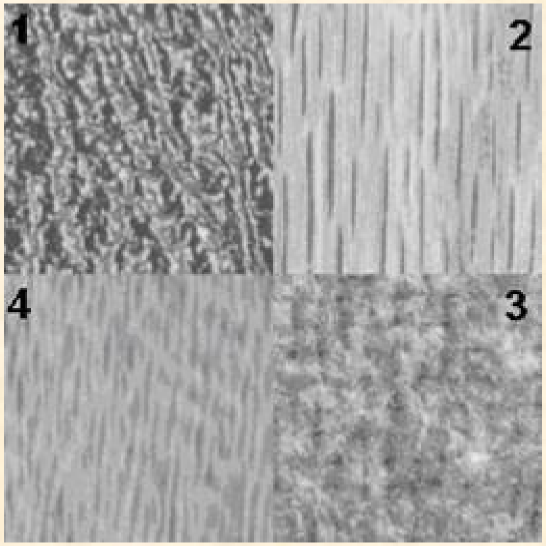
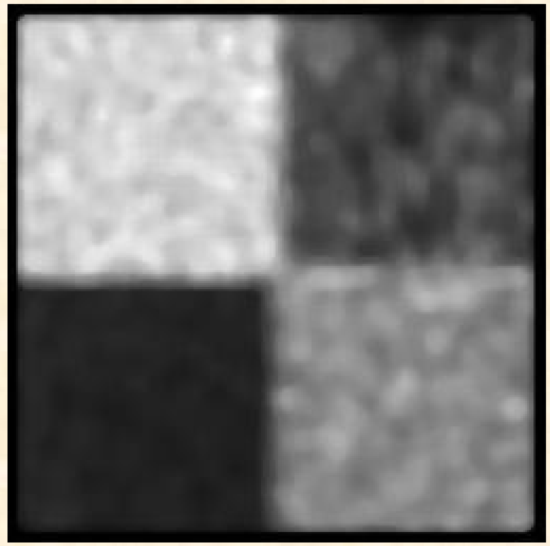


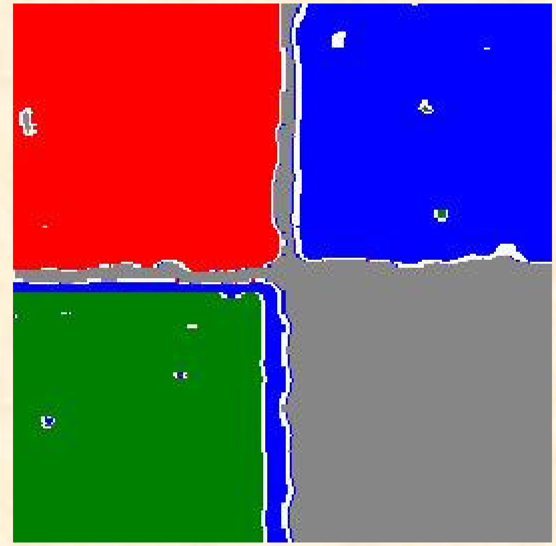
Image segmentation example

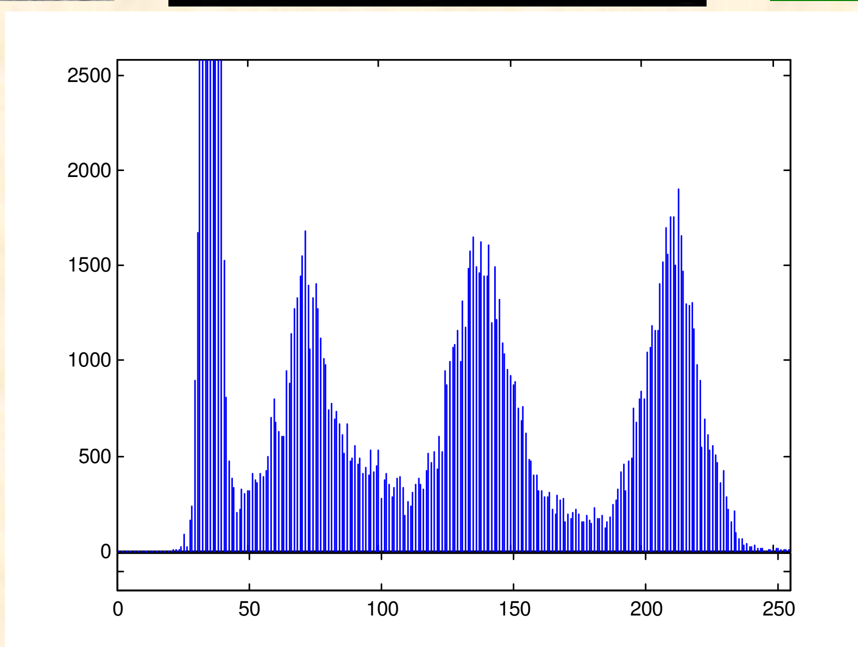


Textures from the Brodatz album







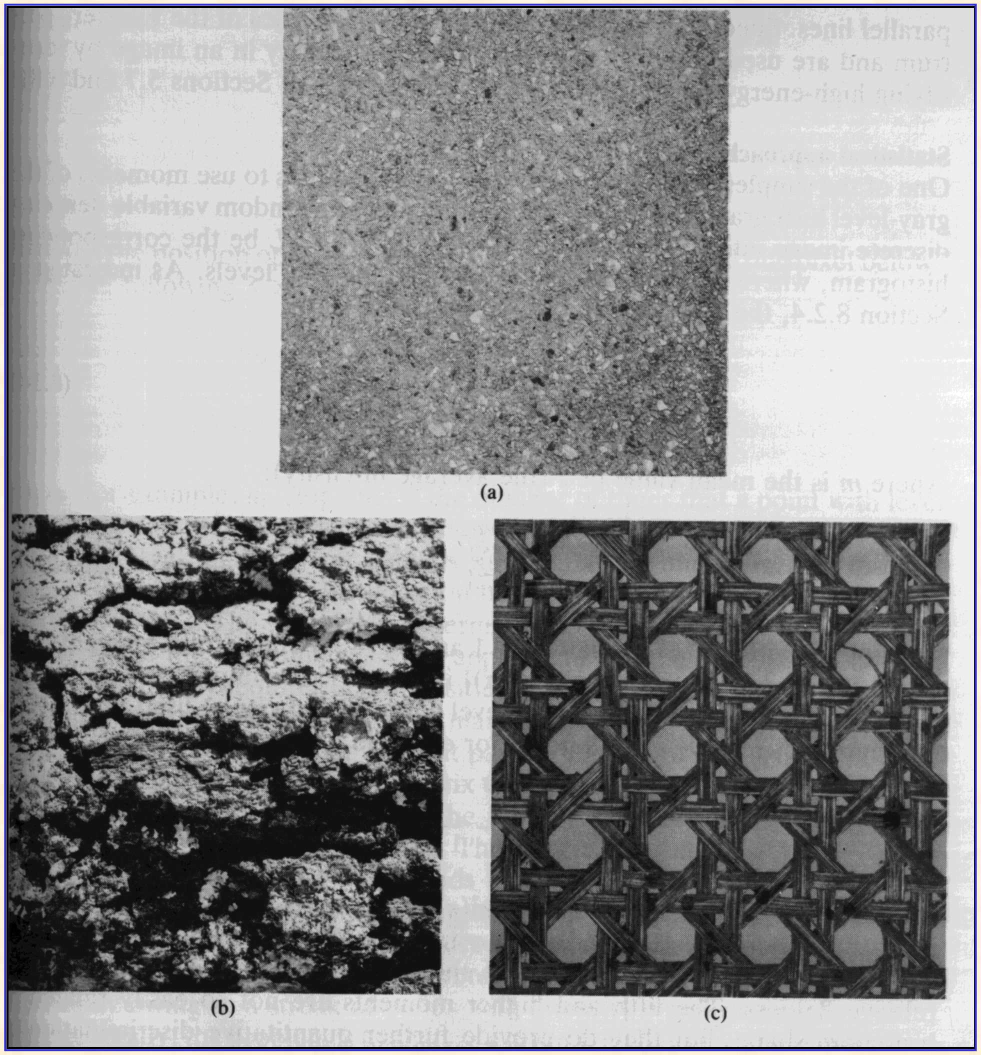


Texture

Texture is observed in the structural patterns of surfaces of objects such as wood, grain, sand, grass, and cloth The term texture generally refers to repetition of basic image patterns called *texels*. The three main approaches used in image processing to describe the texture are:

- *statistical*
- *structural*
- *spectral*

Texture



Examples of smooth, coarse, and regular textures.

Region splitting

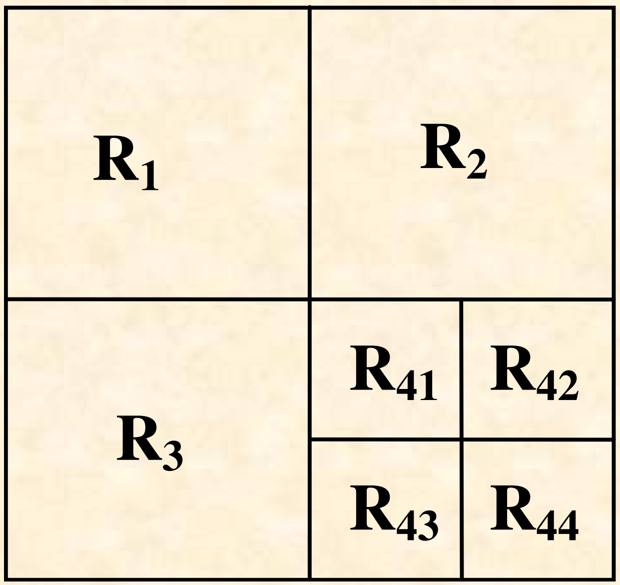
Initially, the image is subdivided into a set of arbitrary disjoint regions, then splits of the regions take place in attempt to satisfy region uniformity criteria (a-e).

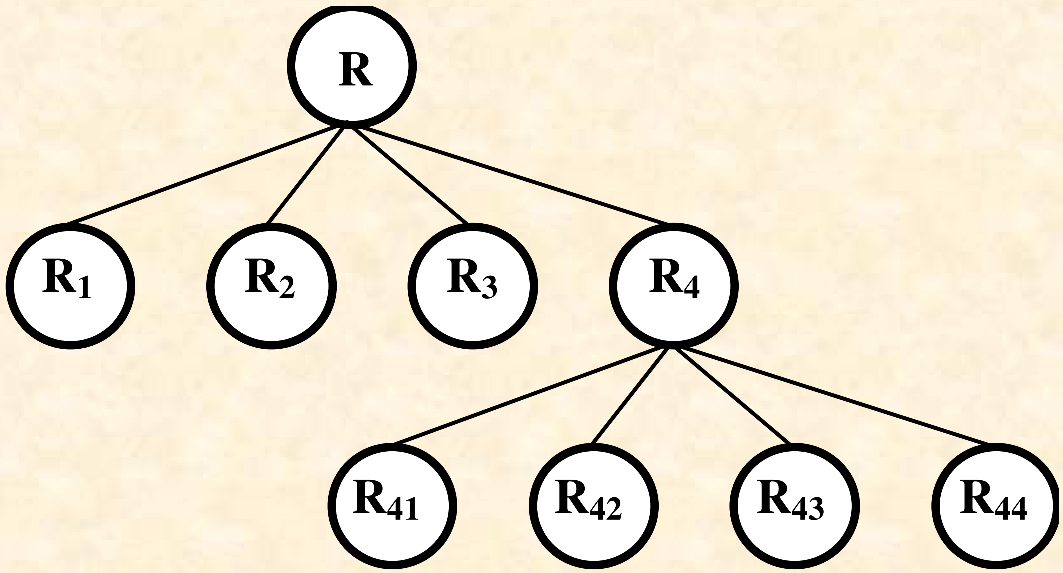
An example split algorithm works as follows:

Subdivide the entire image successively into smaller and smaller quadrant regions until P(Ri) = TRUE for any region; that is if P is FALSE for any quadrant, subdivide the quadrant into sub-quadrants, and so on.

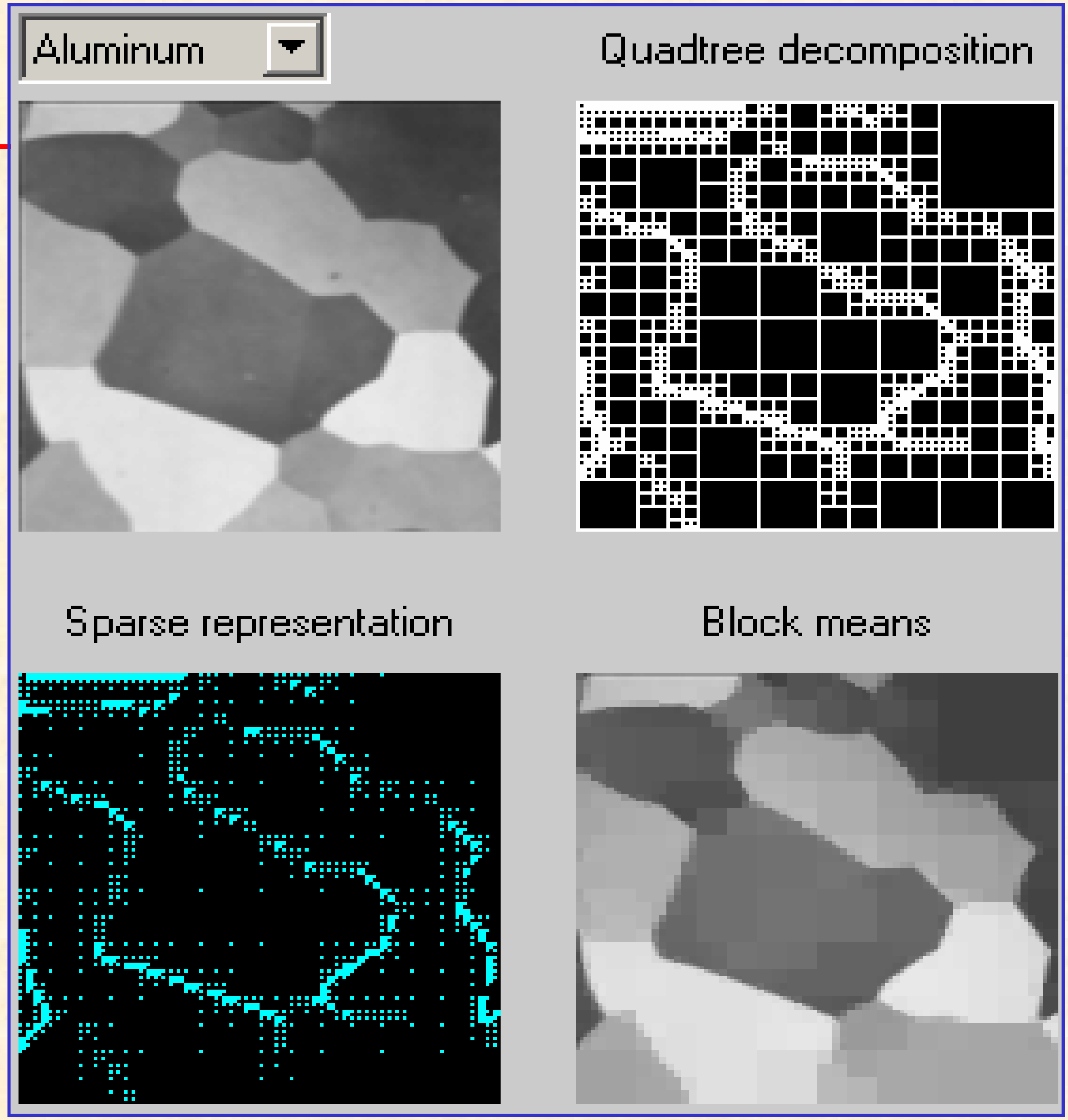
This splitting technique is called the **quad-tree decomposition.**

The quad tree





DEMO
MATLAB
(quad tree)



Region splitting and merging

However, if only splitting is used the final result likely would contain adjacent regions with identical properties.

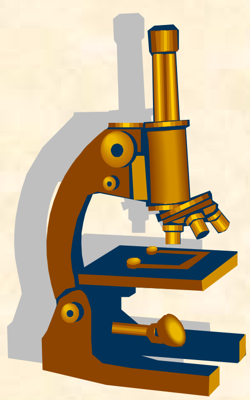
This can be improved by allowing merging as well as splitting. The following procedure can be implemented:

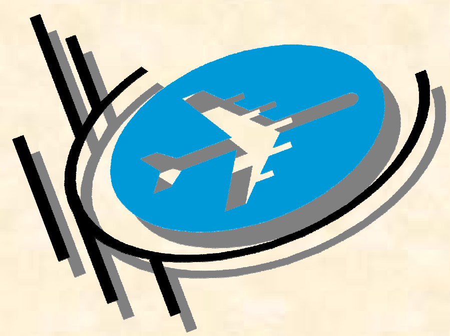
- Split any region R_i if $P(R_i) = FALSE$,
- Merge any adjacent regions for which $P(R_i \cup R_j) = TRUE$,
- Stop when no further merging or splitting is possible.

Template matching

An important problem in image analysis is the **detection** of a presence of an object in a scene.

This problem can be solved by using a special type of an image segmentation technique in which an a priori knowledge about the detected object (**a template**) is used to identify its location in a given scene.



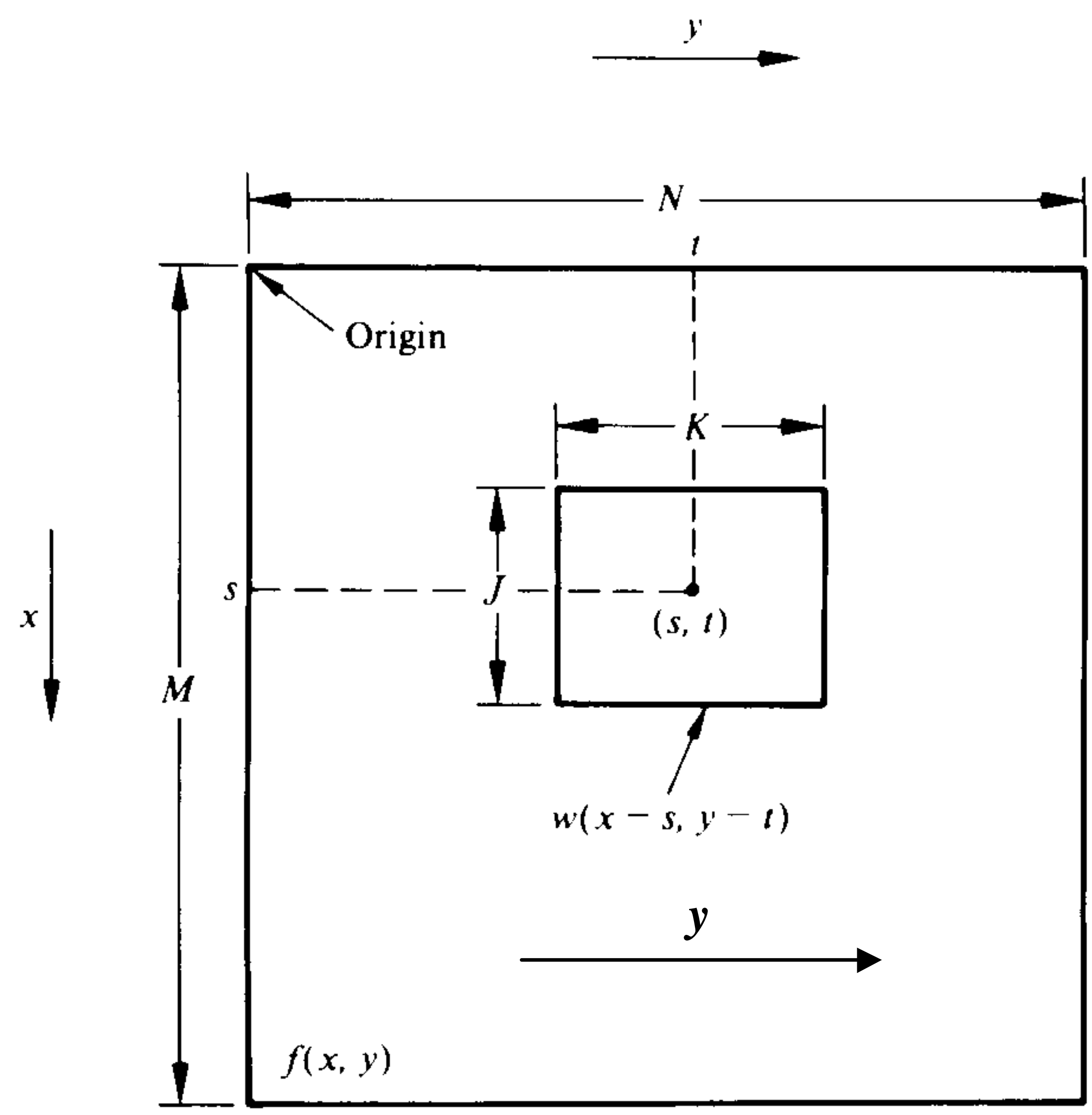


Template matching

Image correlation technique can be used as the basis for finding matches of a searched pattern *w(x,y)* of size $J \times K$ within an image *f(x,y)* of a larger size $M \times N$.

$$c(s,t) = \sum_x \sum_y f(x,y) w(x-s, y-t) \qquad \forall (s,t)$$

the summation is taken over the image region where *w* and *f* overlap.



Template matching

Template matching

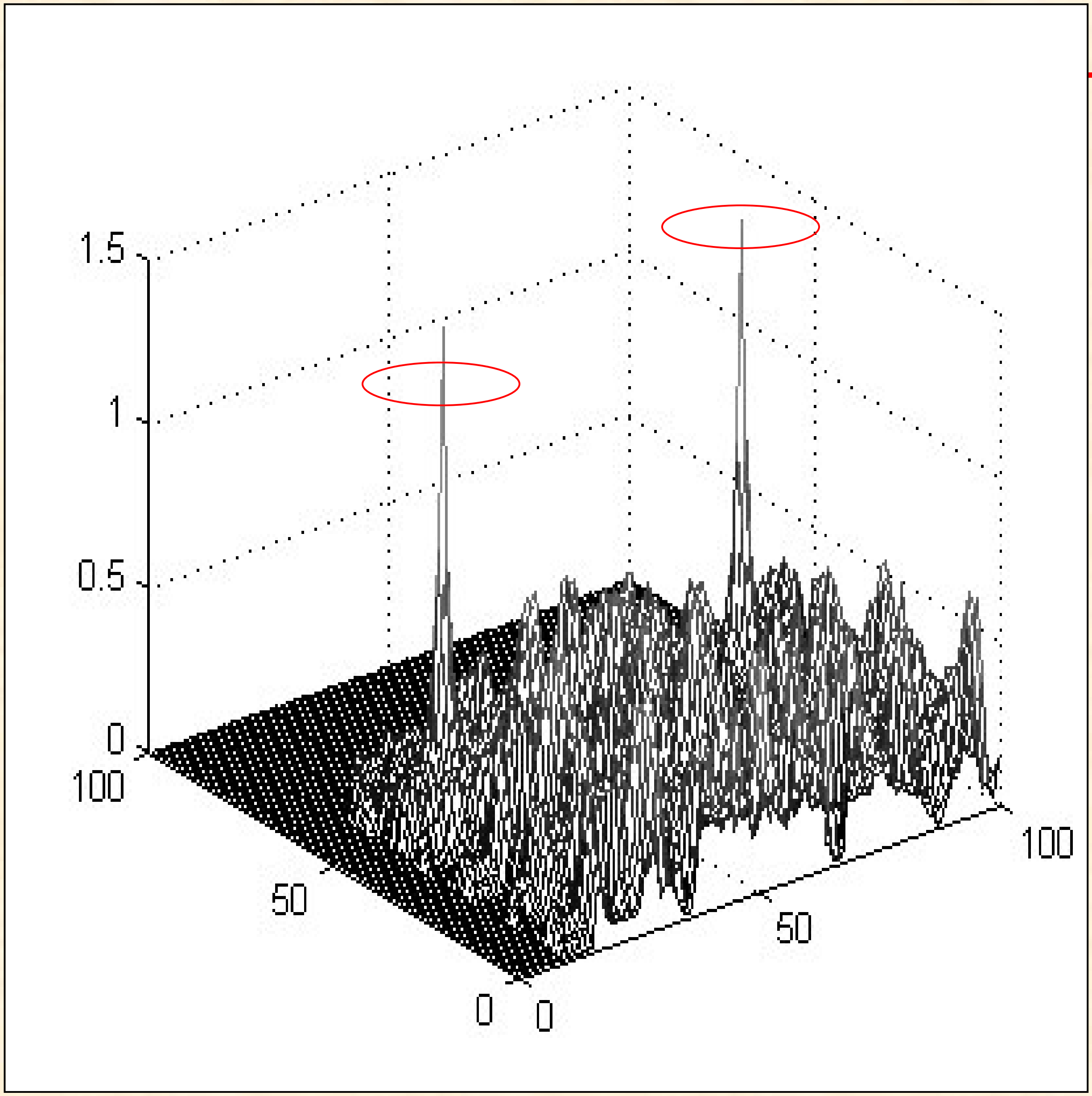
The correlation function has the disadvantage of being sensitive to local intensities of *w(x,y)* and *f(x,y).* In order to get rid of this difficulty pattern matching by using the *correlation coefficient* is used:

$$\gamma(s,t)=\frac{\sum_x\sum_y\left[f(x,y)-\bar{f}(x,y)\right]\left[w(x-s,y-t)-\bar{w}\right]}{\left\{\sum_x\sum_y\left[f(x,y)-\bar{f}(x,y)\right]^2\sum_x\sum_y\left[w(x-s,y-t)-\bar{w}\right]^2\right\}^{1/2}}$$

Template matching

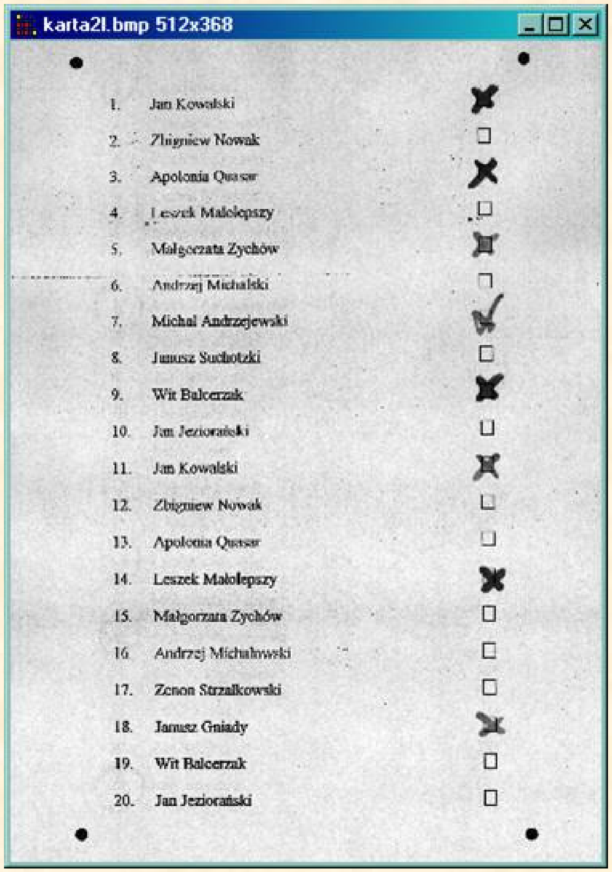

```
%MATLAB
z=xcorr2(c,x);
imshow(z,[ ]);
```

Illustration of the template matching technique (above and below). The detected patter is letter “C”.



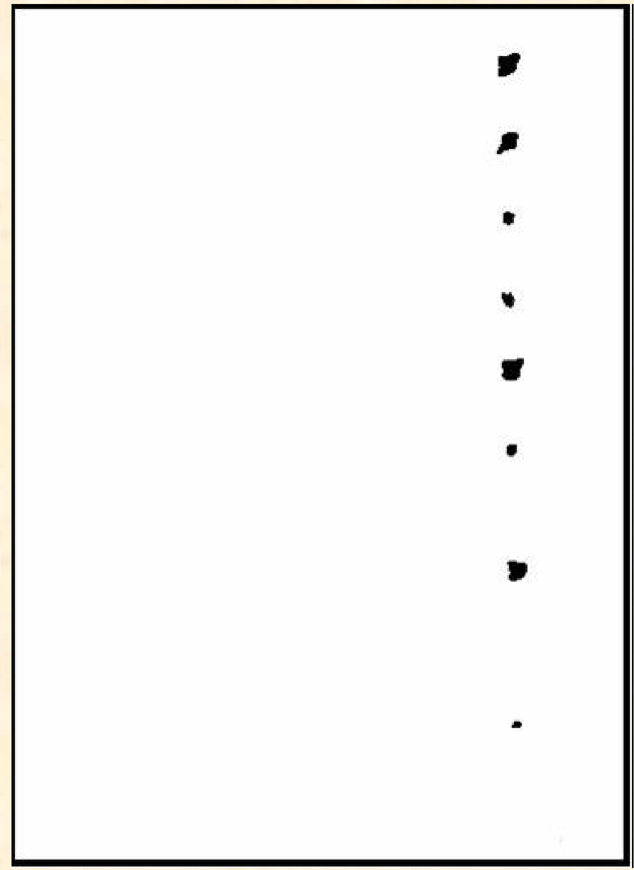
Template matching

Template matching - example



Voting form

Correlation result



Theresholding