

University of Bologna



Image Segmentation

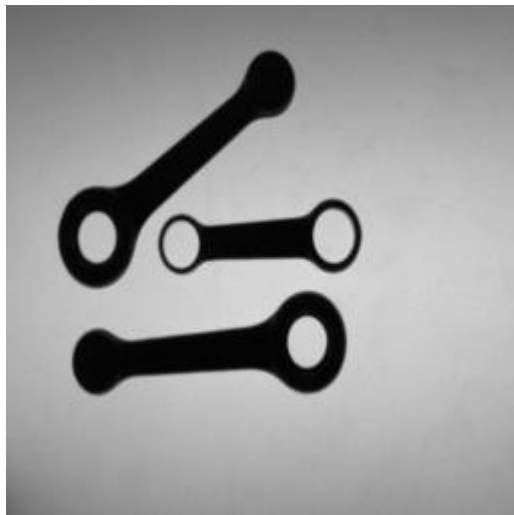
Luigi Di Stefano (luigi.distefano@unibo.it)

Definition



- Denoted as $P(x,y)$ a vector-valued function encoding a set of image properties (such as e.g. intensity, colour, texture, contrast ...), the goal of segmentation is to partition the image into disjoint homogeneous regions according to P .
- Typically, a good segmentation should preserve spatial proximity (i.e. two nearby pixels must belong to the same region unless they exhibit significantly different P values) and provide relatively large regions featuring a few holes and well-localized, smooth boundaries.
- In many computer vision tasks segmentation brings in key *semantic knowledge* on the scene, as it splits the image into semantically meaningful parts (e.g. foreground/background, individual objects, moving/static pixels...) on which further analysis can then be focused.
- In many practical applications related to machine vision, segmentation relies on just a single image property, such as e.g. intensity ($P(x,y)=I(x,y)$) or colour ($P(x,y)=[I_r(x,y) \ I_g(x,y) \ I_b(x,y)]^T$).

Inherently-binary Images (1)



Inherently-binary Images (2)

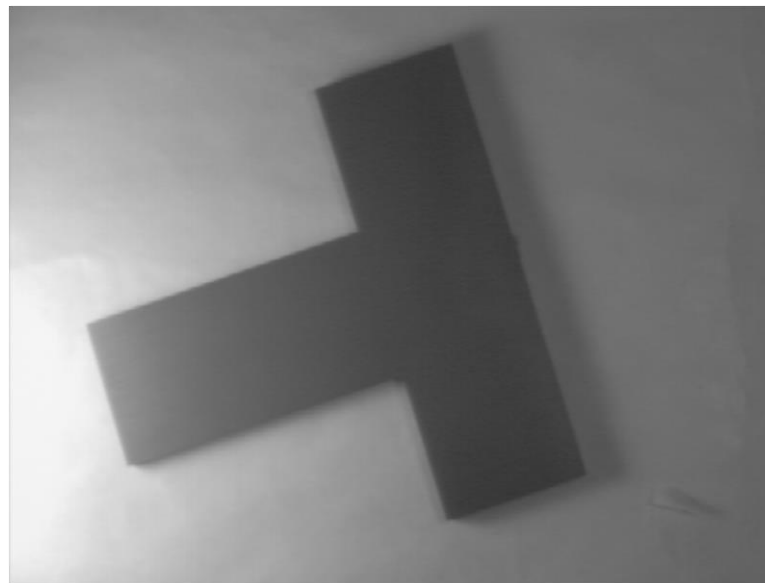
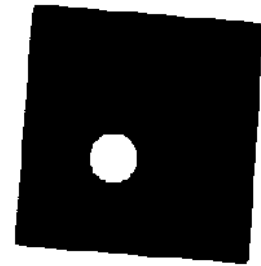
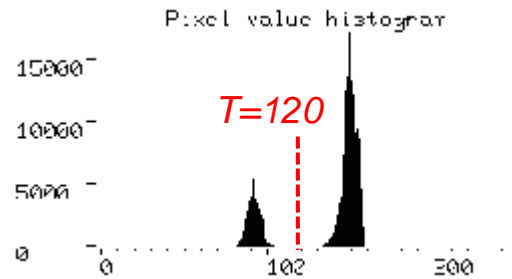
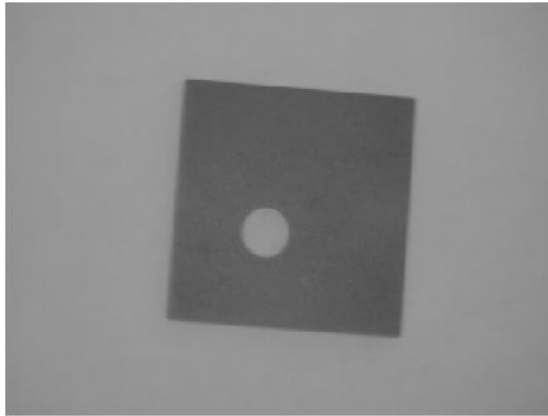


Image Binarization



- In a variety of applications the objects of interest (*foreground*) are neatly darker/brighter than the irrelevant areas of the scene (*background*). In many industrial applications this is achieved by *backlighting*, whereby the object is placed between the light source and the camera so as to cast onto the image a very dark shadow representing object's shape.
- In such circumstances, the first image analysis step consists typically in image *binarization*, i.e. segmentation of image pixels into two disjoint regions corresponding to foreground (dark/bright intensities) and background (bright/dark intensities).
- Should the application call for analysis of a single object per image, binarization would deliver all the required segmentation information.
- Conversely, the foreground region must be further split into sub-regions corresponding to individual objects, which is usually achieved by a successive image analysis step referred to as *image labeling* (also *connected components labeling*) due to pixels belonging to different objects being given different labels.

Binarization by Intensity Thresholding



- Inherently-binary images exhibit a clearly *bimodal* gray-level histogram, with two well-separated peaks corresponding to foreground and background pixels. Therefore, binarization can be achieved straightforwardly by a means of a *thresholding operator* deploying a suitably chosen threshold, T :

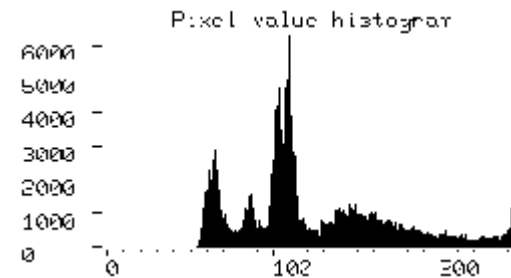
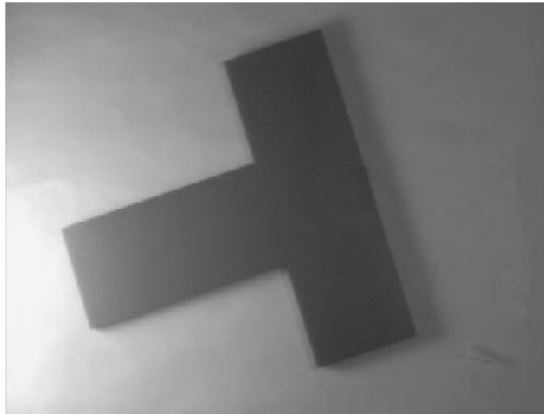
```
/* Binarization by a global threshold, objects darker than background
# define  FOREGROUND  0
# define  BACKGROUND  255
# define  THRESHOLD   120

for(i=0;i<N; i++)
for(j=0;j<M; j++)
if (I[i][j]<=THRESHOLD)
    O[i][j]=FOREGROUND;
    else
    O[i][j]=BACKGROUND;
```

Failure Cases

- Whenever the histogram is not clearly bimodal, e.g. due to illumination varying significantly across the scene (example below), binarization by intensity thresholding fails to provide the correct segmentation.

Original Image



Gray-level Histogram

T=80

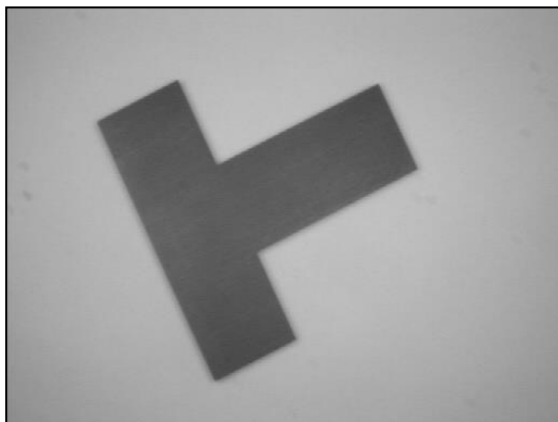


T=120

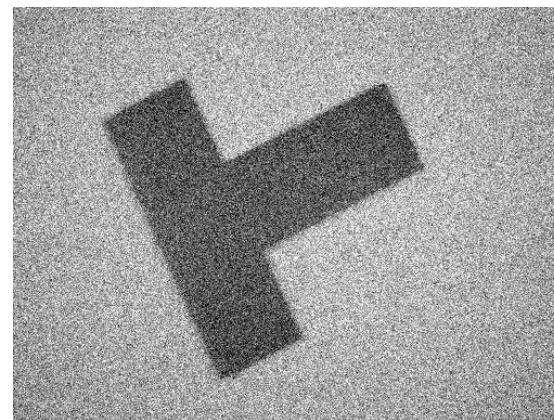
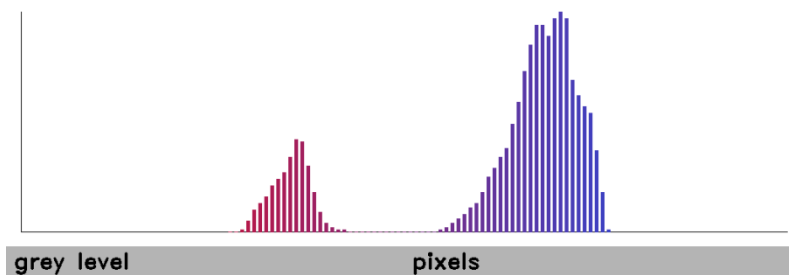


Smoothing and Thresholding (1)

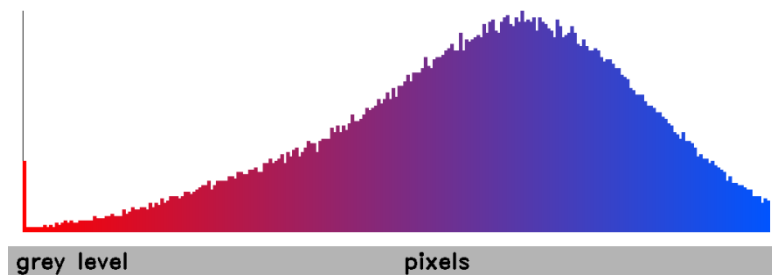
- When the overlap between the two modes is due to noise, image smoothing may help improving binarization.



*Noiseless Image
and its Histogram*



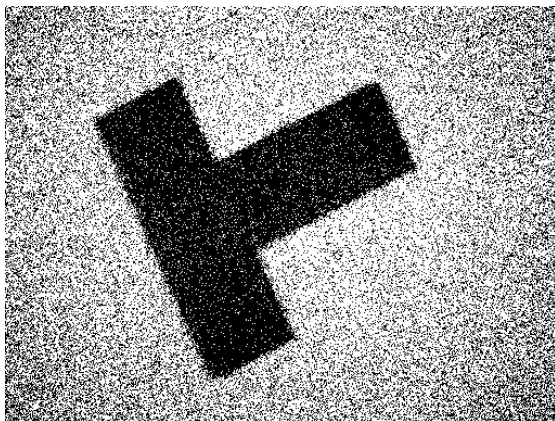
*Noisy Image and
its Histogram*



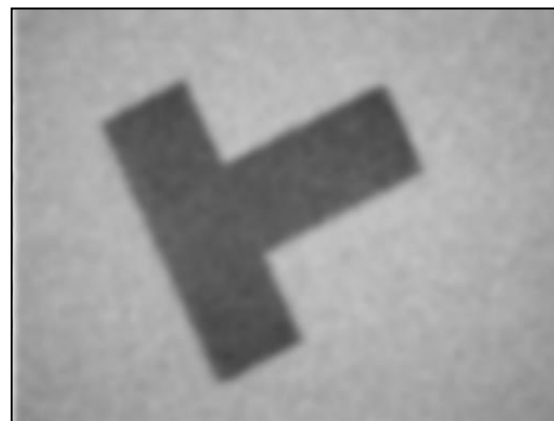
Smoothing and Thresholding (2)

- Obviously, binarization of the noisy image fails. Yet smoothing by a Gaussian filter improves the histogram and allows for a more correct binarization.

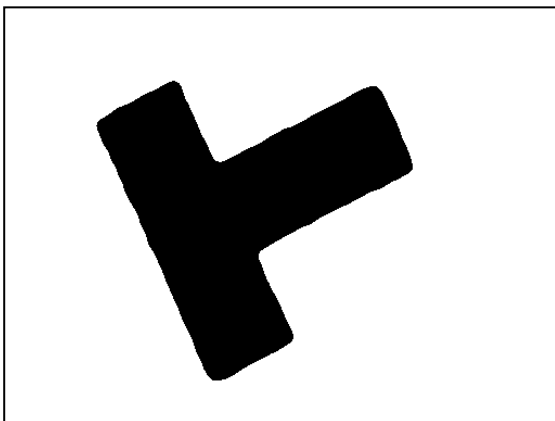
*Binarization of
the Noisy Image*



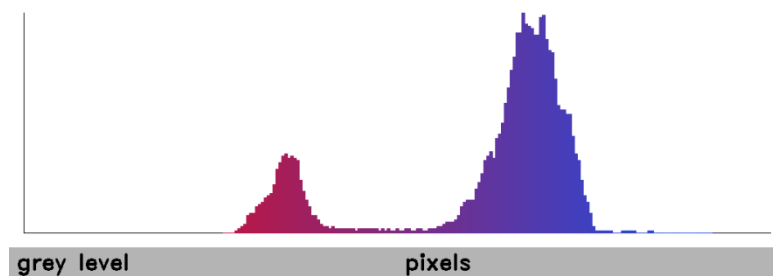
*Filtered
Image*



*Binarization of
the Filtered
Image*

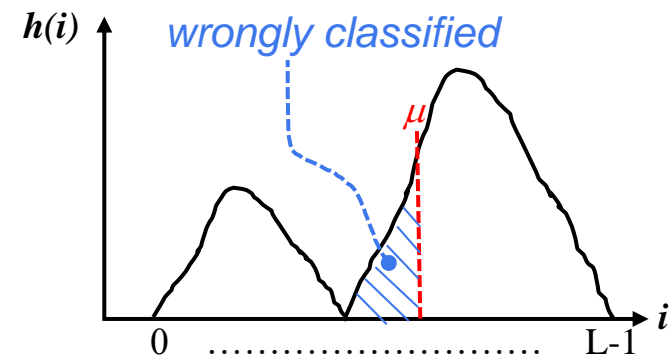
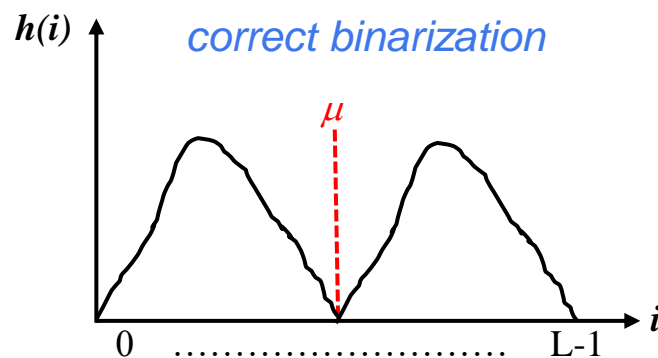


*Histogram of the
Filtered Image*

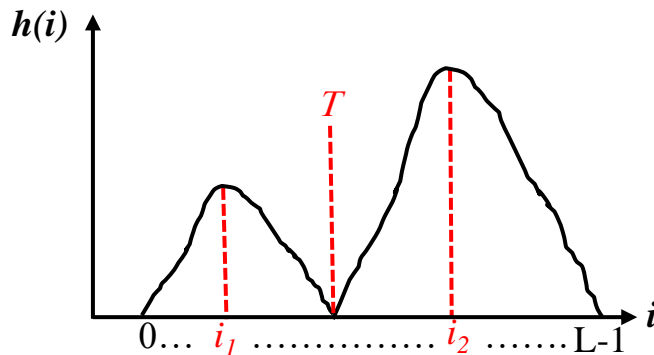


Automatic Threshold Selection

- In many practical applications stability over time of the lighting conditions cannot be guaranteed. Such applications mandate a robust, though computationally more demanding, approach whereby an algorithm computes automatically a suitable binarization threshold in each image under analysis.
- Simple heuristic approaches are as follows:
 - ✓ $T = \mu$: works as long as pixels are equally distributed between the two classes. Likewise, a certain properly estimated percentile may be chosen (e.g. either the 20th or 80th percentile if we know *dark/bright* objects to cover approximately 20% of the image).



Peaks Method



✓ $T = \arg \min \{ h(i) \mid i \in [i_1, i_2] \}$
require finding the two main peaks,
which often implies smoothing the
histogram before looking for peaks to
avoid the search be trapped into
spurious local maxima.

To find the two main peaks, i_1, i_2 :

```
/* 3-levels neighbourhood  
((h[i]>h[i-1])&&(h[i]>h[i+1]))
```

```
/* 5-levels neighbourhood  
((h[i]>h[i-1])&&(h[i]>h[i+1])&&(h[i-1]>h[i-2])&&(h[i+1]>h[i+2]))
```

Otsu's Algorithm (1)



- A principled and effective automatic threshold selection algorithm is due to Otsu. The key intuition is to segment the image into two maximally homogeneous regions. Accordingly, the *optimal* threshold is chosen so as to minimize across the gray-level range the so-called *Within-group Variance* of the resulting regions, such an indicator measuring how spread turn out region intensities upon binarization by a given gray-level.
 - Let us define:
 - $i = 1 \dots L$: gray-levels of the image
 - N : number of pixels of the image
 - $h(i)$: i^{th} entry of the image histogram
 - $p(i) = h(i)/N$: probability of gray-level i $\left(\sum_{i=1}^L p(i) = 1 \right)$

Otsu's Algorithm (2)

- Accordingly, the mean, μ , and variance, σ^2 , of the *pmf* associated with image gray-levels can be expressed as:

$$\mu = \sum_{i=1}^L i p(i) \quad \sigma^2 = \sum_{i=1}^L (i - \mu)^2 p(i)$$

- Any threshold value, t , would split pixels into two disjoint regions whose associated *pmfs* have mean and variance given by:

$$\begin{aligned} \mu_1(t) &= \sum_{i=1}^t i p(i)/q_1(t) & \sigma_1^2(t) &= \sum_{i=1}^t (i - \mu_1(t))^2 p(i)/q_1(t) \\ \mu_2(t) &= \sum_{i=t+1}^L i p(i)/q_2(t) & \sigma_2^2(t) &= \sum_{i=t+1}^L (i - \mu_2(t))^2 p(i)/q_2(t) \end{aligned}$$

with

$$q_1(t) = \sum_{i=1}^t p(i) \quad q_2(t) = \sum_{i=t+1}^L p(i)$$

Otsu's Algorithm (3)

- The Within-group Variance of the two regions is defined as the weighted sum of their variances:

$$\sigma_W^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t)$$

- Minimization of the above quantity would require computing μ_1 , μ_2 , σ_1^2 , σ_2^2 and q_1 ($q_2 = 1 - q_1$) for each gray-level values. Yet, a more efficient approach can be pursued.

$$\begin{aligned}\sigma^2 &= \sum_{i=1}^L (i - \mu)^2 p(i) \\ &= \sum_{i=1}^t (i - \mu_1(t) + \mu_1(t) - \mu)^2 p(i) + \sum_{i=t+1}^L (i - \mu_2(t) + \mu_2(t) - \mu)^2 p(i) \\ &= \sum_{i=1}^t \left[(i - \mu_1(t))^2 + 2 (i - \mu_1(t)) (\mu_1(t) - \mu) + (\mu_1(t) - \mu)^2 \right] p(i) \\ &\quad + \sum_{i=t+1}^L \left[(i - \mu_2(t))^2 + 2 (i - \mu_2(t)) (\mu_2(t) - \mu) + (\mu_2(t) - \mu)^2 \right] p(i)\end{aligned}$$

Otsu's Algorithm (4)

$$\begin{aligned} \text{---} \rightarrow \sum_{i=1}^t (i - \mu_1(t)) (\mu_1(t) - \mu) p(i) &= 0 \\ \text{---} \rightarrow \sum_{i=t+1}^L (i - \mu_2(t)) (\mu_2(t) - \mu) p(i) &= 0 \end{aligned}$$

Therefore, σ^2 can be expressed as:

$$\begin{aligned} \sigma^2 &= \sum_{i=1}^t (i - \mu_1(t))^2 p(i) + \sum_{i=t+1}^L (i - \mu_2(t))^2 p(i) + (\mu_1(t) - \mu)^2 q_1(t) + (\mu_2(t) - \mu)^2 q_2(t) \\ \sigma^2 &= [q_1(t) \sigma_1^2(t) + q_2(t) \sigma_2^2(t)] + [(\mu_1(t) - \mu)^2 q_1(t) + (\mu_2(t) - \mu)^2 q_2(t)] \\ \sigma^2 &= \sigma_W^2(t) + \sigma_B^2(t) \end{aligned}$$


$\sigma_B^2(t)$ being the *Between-group Variance*, i.e. an indicator of how well classes are separated one to the other.

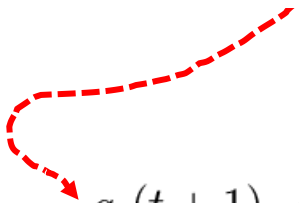
- As σ^2 is independent of t , the above relation suggests we might wish to maximize σ_B^2 rather than minimizing σ_W^2 , the former being a more efficient procedure as variances need not be calculated.

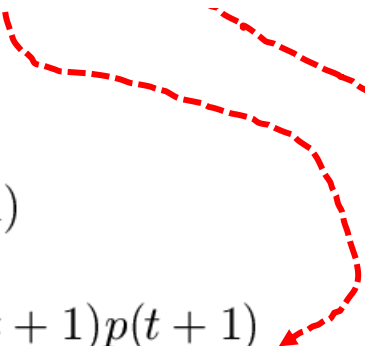
Otsu's Algorithm (5)

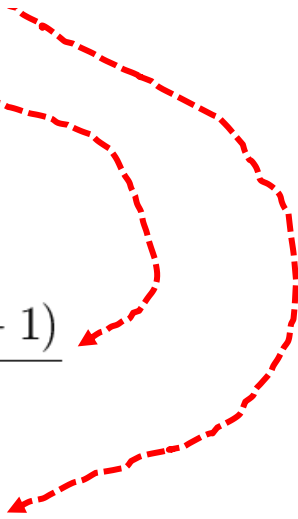
- Further computational savings can be achieved as follows:

$$\mu = q_1(t)\mu_1(t) + q_2(t)\mu_2(t) \quad q_2(t) = 1 - q_1(t)$$


$$\sigma_B^2(t) = q_1(t) (1 - q_1(t)) (\mu_1(t) - \mu_2(t))^2$$


$$q_1(t+1) = q_1(t) + p(t+1)$$


$$\mu_1(t+1) = \frac{q_1(t)\mu_1(t) + (t+1)p(t+1)}{q_1(t+1)}$$

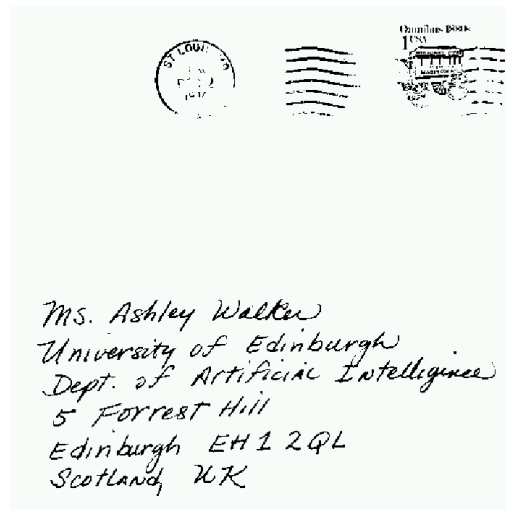

$$\mu_2(t+1) = \frac{\mu - q_1(t+1)\mu_1(t+1)}{1 - q_1(t+1)}$$

Examples (1)

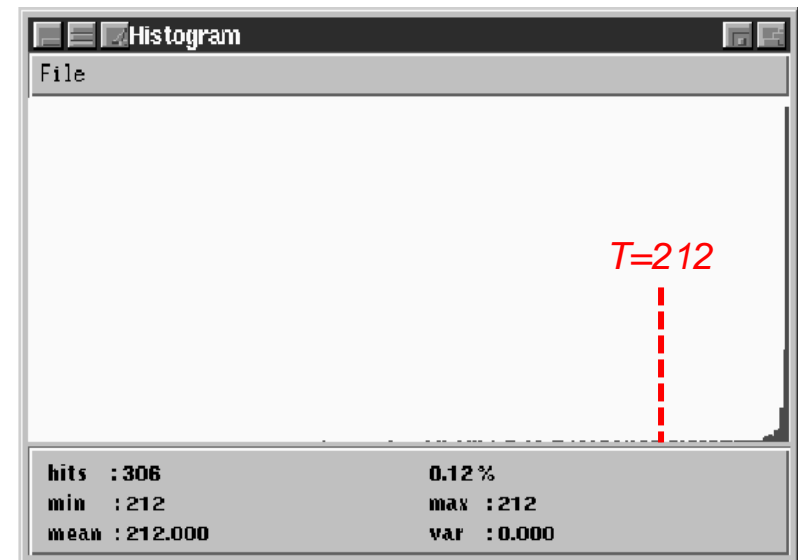
Input
Image



Binarized
Image



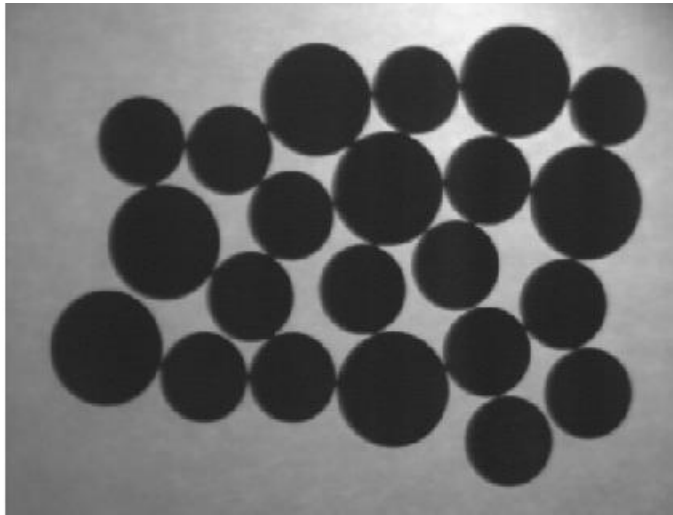
Gray-level
Histogram



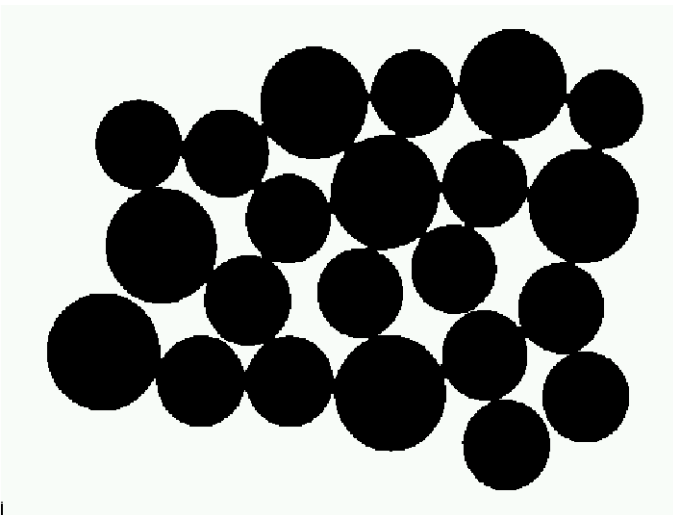
with threshold computed
by Otsu's

Examples (2)

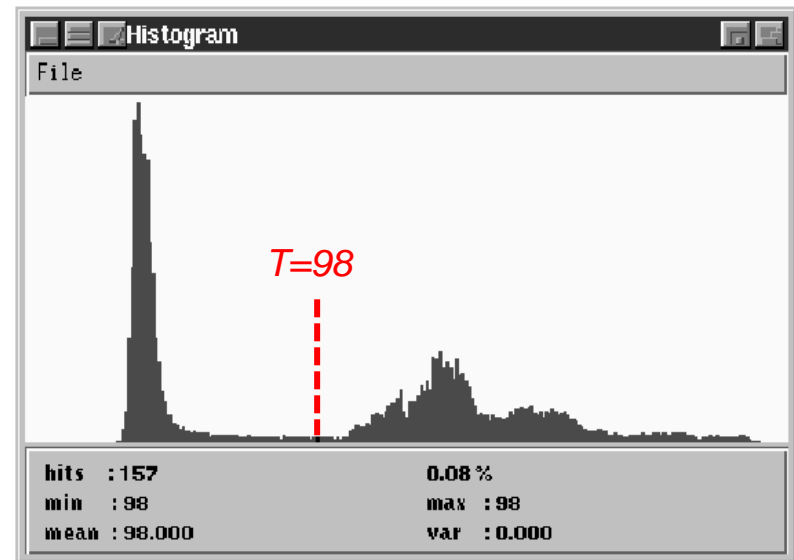
*Input
Image*



*Binarized
Image*



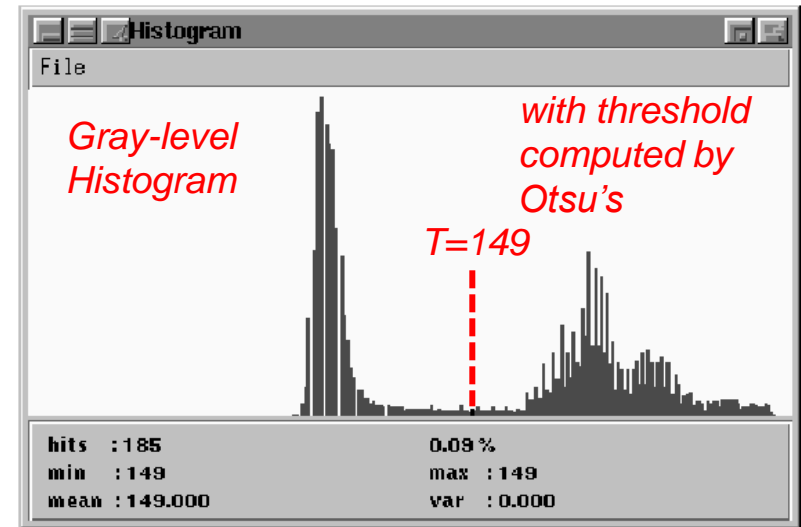
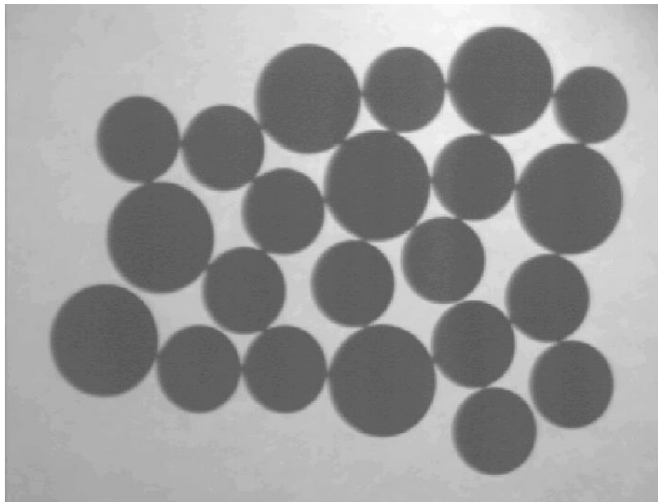
*Gray-level
Histogram*



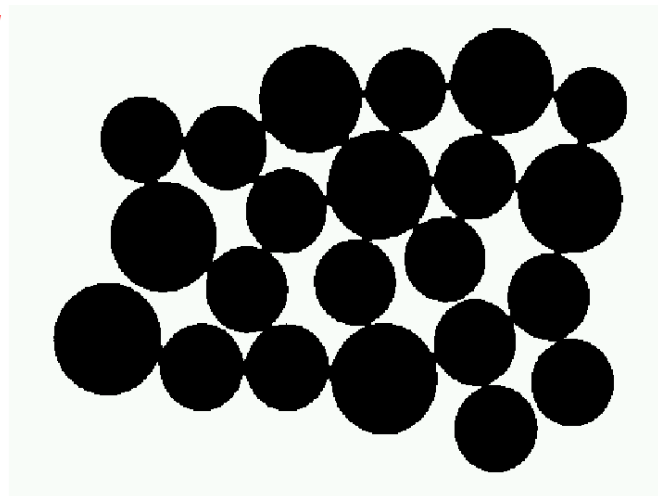
*with threshold computed
by Otsu's*

Examples (3)

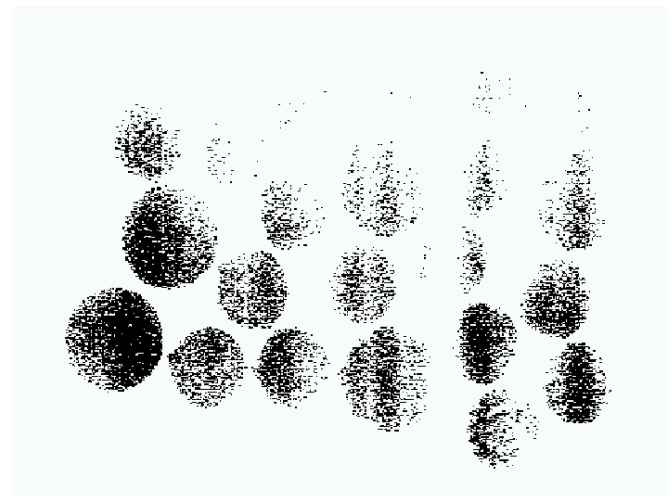
*Brighter
Input
Image*



*Binarized
Image
($T=149$)*



*Binarized
Image
($T=98$)*



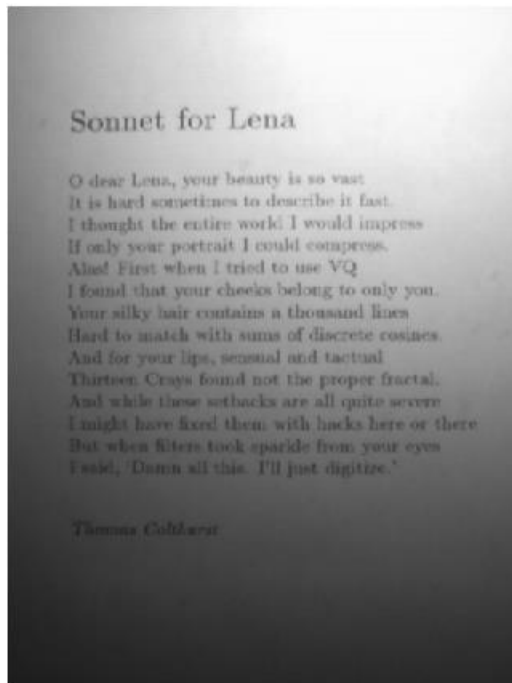
Adaptive Thresholding



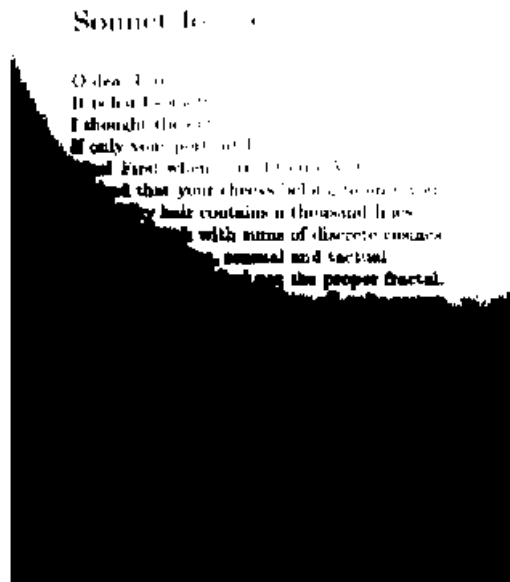
- Any global thresholding method relies on the assumption of uniform lighting across the scene. Should this assumption be violated, e.g. due to shadows, a spatially varying (i.e. adaptive) threshold ought be employed in case one still wishes to binarize the image.
- Usually, adaptive methods compute a specific threshold at each image pixel ($T \rightarrow T(x,y)$) based on the intensities within a small neighbourhood (such as e.g. 5×5 , 7×7 , 9×9 ..). However, too small a neighbourhood might lack either background or foreground pixels, which would imply segmentation errors unless the issue is dealt with explicitly.
- For the sake of efficiency, rather simple operators, such as the *Mean* or the *Median*, are typically adopted to compute the threshold value at each pixel.

Example (1)

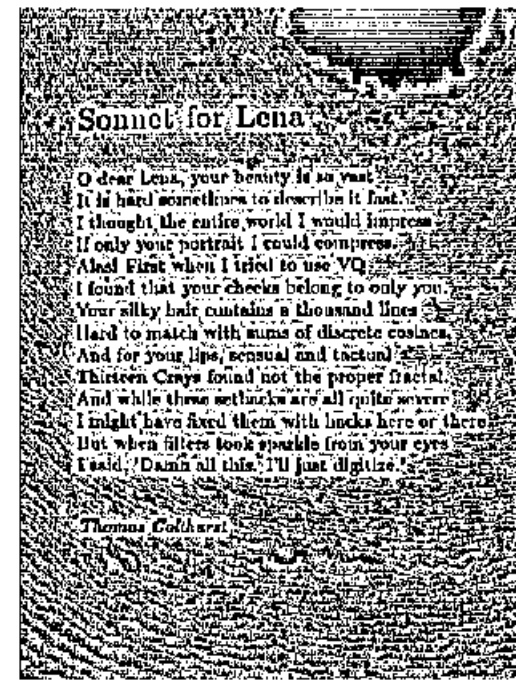
*Image of an unevenly
lit scene*



*Binarization by
global
thresholding*



*Binarization by
adaptive
thresholding*



$$T(x,y) = \mu(x,y),$$

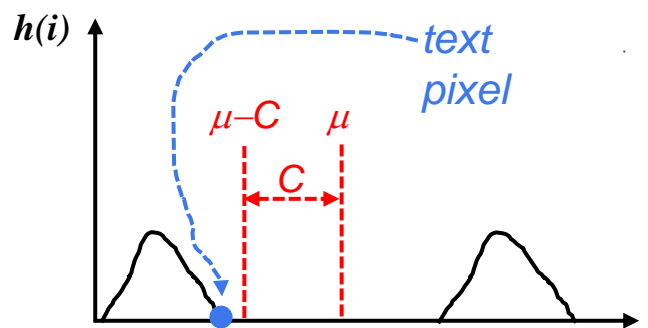
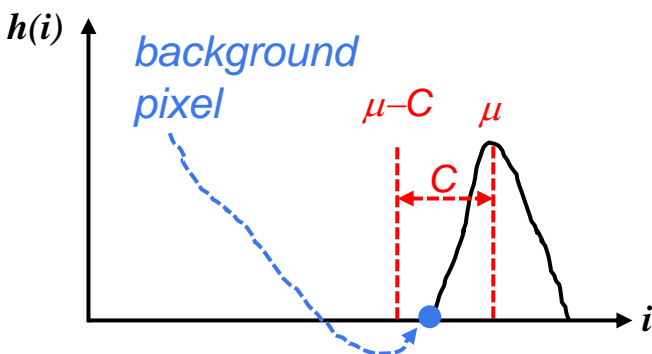
7×7 neighbourhood

Example (2)

In text images, the issue of neighbourhoods lacking both types of pixels can be dealt with by deploying the peculiarity of most such neighbourhoods being associated with background pixels. Hence, a simple trick to improve segmentation dramatically consist in subtracting a suitable constant

$$T(x, y) = \mu(x, y) - C$$

so as to push above threshold most wrongly classified background pixels.



*Binarization by
adaptive
thresholding*

Sonnet for Lena

O dear Lena, your beauty is so vast
It is hard sometimes to describe it fast.
I thought the entire world I would impress
If only your portrait I could compress.
Alas! First when I tried to use VQ
I found that your cheeks belong to only you.
Your silky hair contains a thousand lines
Hard to match with sums of discrete cosines.
And for your lips, sensual and factual
Thirteen Crays found not the proper fractal.
And while these setbacks are all quite severe
I might have fixed them with hacks here or there
But when filters took sparkle from your eyes
I said, 'Damn all this. I'll just digitize.'

Thomas Culhane

$T(x, y) = \mu(x, y), C=10$
 7×7 neighbourhood

Colour-based segmentation



- In several applications, the sought-for objects exhibit a known colour quite different from that of background structures (e.g. “*blue screen*”). Hence, it is reasonable to try segmenting-out foreground from background based on colour information.

- Let us denote a pixel's colour as: $\mathbf{I}(p) = \begin{bmatrix} I_r(p) \\ I_g(p) \\ I_b(p) \end{bmatrix}$

- Then, segmentation can be achieved by computing and thresholding the distance (e.g. Euclidean) between each pixel's colour and the reference background (foreground) colour μ :

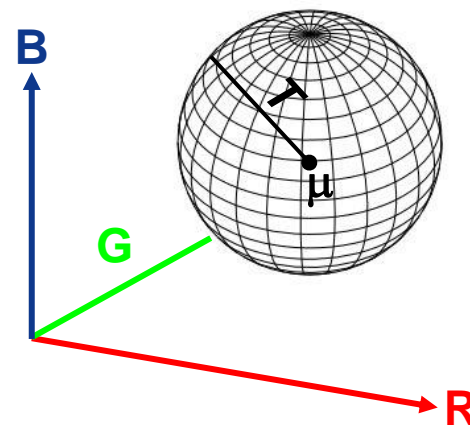
$$\rightarrow \forall p \in \mathbf{I}: \begin{cases} d(\mathbf{I}(p), \mu) \leq T \rightarrow O(p) = B \quad (F) \\ d(\mathbf{I}(p), \mu) > T \rightarrow O(p) = F \quad (B) \end{cases}$$

$$d(\mathbf{I}(p), \mu) = \left((I_r(p) - \mu_r)^2 + (I_g(p) - \mu_g)^2 + (I_b(p) - \mu_b)^2 \right)^{\frac{1}{2}}$$

Estimating the reference colour

- It is thus necessary to know the reference colour, μ , which can be conveniently estimated (“learned”) from one or more *training images*.
- For example, modelling the colour of a background (foreground) pixel as a multivariate random variable (i.e. a 3D random vector), the reference colour can be taken to be the mean (expected value) over the available training samples ($I(p_k)$, $k=1..N$):

$$\mu = \begin{bmatrix} \mu_r \\ \mu_g \\ \mu_b \end{bmatrix} = \frac{1}{N} \sum_{k=1}^N I(p_k)$$



Accordingly, segmentation consists in classifying as background (foreground) all pixels lying within a 3D sphere of the RGB colour space centred at μ and having radius as large as T .

Mahalanobis Distance (1)



- A far richer probabilistic characterization of colour distribution among foreground pixels can be obtained estimating from training samples not only the mean but also the covariance matrix:

$$\Sigma = \begin{pmatrix} \sigma_{rr}^2 & \sigma_{rg}^2 & \sigma_{rb}^2 \\ \sigma_{gr}^2 & \sigma_{gg}^2 & \sigma_{gb}^2 \\ \sigma_{br}^2 & \sigma_{bg}^2 & \sigma_{bb}^2 \end{pmatrix} \rightarrow \begin{cases} \sigma_{ij}^2 = \frac{1}{N} \sum_{k=1}^N (I_i(p_k) - \mu_i)(I_j(p_k) - \mu_j) \\ i, j \in \{r, g, b\} \end{cases}$$

- Recalling that the Euclidean can also be written as :

$$d(\mathbf{I}(p), \boldsymbol{\mu}) = \left((\mathbf{I}(p) - \boldsymbol{\mu})^T (\mathbf{I}(p) - \boldsymbol{\mu}) \right)^{\frac{1}{2}}$$

the *Mahalanobis Distance* is given by :

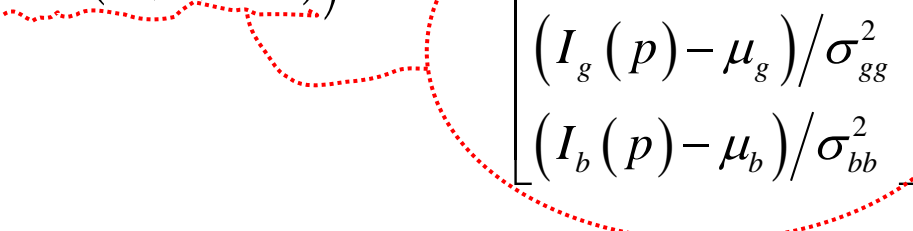
$$d_M(\mathbf{I}(p), \boldsymbol{\mu}) = \left((\mathbf{I}(p) - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{I}(p) - \boldsymbol{\mu}) \right)^{\frac{1}{2}}$$


Mahalanobis Distance (2)



- To understand the difference between the two distances, it is worth considering the case of diagonal covariance matrix (i.e. independent components in $\mathbf{I}(p)$)

$$\Sigma = \begin{pmatrix} \sigma_{rr}^2 & 0 & 0 \\ 0 & \sigma_{gg}^2 & 0 \\ 0 & 0 & \sigma_{bb}^2 \end{pmatrix} \quad \rightarrow \quad \Sigma^{-1} = \begin{pmatrix} 1/\sigma_{rr}^2 & 0 & 0 \\ 0 & 1/\sigma_{gg}^2 & 0 \\ 0 & 0 & 1/\sigma_{bb}^2 \end{pmatrix}$$

$$d_M(\mathbf{I}(p), \boldsymbol{\mu}) = \left((\mathbf{I}(p) - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{I}(p) - \boldsymbol{\mu}) \right)^{\frac{1}{2}}$$

$$\begin{bmatrix} (I_r(p) - \mu_r) / \sigma_{rr} \\ (I_g(p) - \mu_g) / \sigma_{gg} \\ (I_b(p) - \mu_b) / \sigma_{bb} \end{bmatrix}$$


$$d_M(\mathbf{I}(p), \boldsymbol{\mu}) = \left(\frac{(I_r(p) - \mu_r)^2}{\sigma_{rr}^2} + \frac{(I_g(p) - \mu_g)^2}{\sigma_{gg}^2} + \frac{(I_b(p) - \mu_b)^2}{\sigma_{bb}^2} \right)^{\frac{1}{2}}$$

Mahalanobis Distance (3)



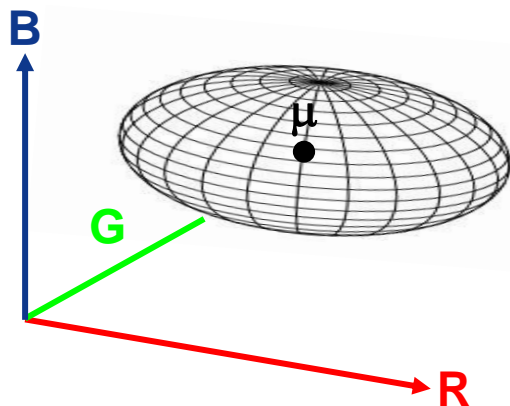
- Contrary to the Euclidean distance, the Mahalanobis distance weighs unequally the differences along the components of the random vector, in particular according to inverse proportionality to the learned variances. As such, the more spread has been learned to be a component, the less the difference along it will contribute to the overall distance.

$$d_M(\mathbf{I}(p), \boldsymbol{\mu}) = \left(\frac{(I_r(p) - \mu_r)^2}{\sigma_{rr}^2} + \frac{(I_g(p) - \mu_g)^2}{\sigma_{gg}^2} + \frac{(I_b(p) - \mu_b)^2}{\sigma_{bb}^2} \right)^{\frac{1}{2}}$$

Mahalanobis Distance (4)

- Which region of the RGB space determines now the segmentation ?

$$d_M(\mathbf{I}(p), \boldsymbol{\mu})^2 = \left(\frac{(I_r(p) - \mu_r)^2}{\sigma_{rr}^2} + \frac{(I_g(p) - \mu_g)^2}{\sigma_{gg}^2} + \frac{(I_b(p) - \mu_b)^2}{\sigma_{bb}^2} \right) \leq T^2$$



Ellipsoid
centred at :

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_r \\ \mu_g \\ \mu_b \end{bmatrix}$$

With axes aligned to
coordinate axes and
semi-axes lengths
given by:

$$\mathbf{L} = \begin{bmatrix} L_r \\ L_b \\ L_g \end{bmatrix} = T \begin{bmatrix} \sigma_{rr} \\ \sigma_{bb} \\ \sigma_{gg} \end{bmatrix}$$

Mahalanobis Distance (4)



- The interpretation of the Mahalanobis distance which has been given in case of diagonal covariance matrix is of general validity.
- Indeed, the covariance matrix can always be diagonalized by a rotation of the coordinate axes. Thus, in the new coordinate system the Mahalanobis distance is a weighted sum of the contributions along the new axes, with weights being inversely proportional to the variances along the new axes.
- This is due to the *EigenValue Decomposition* (EVD) of any real and symmetric matrix (Σ):

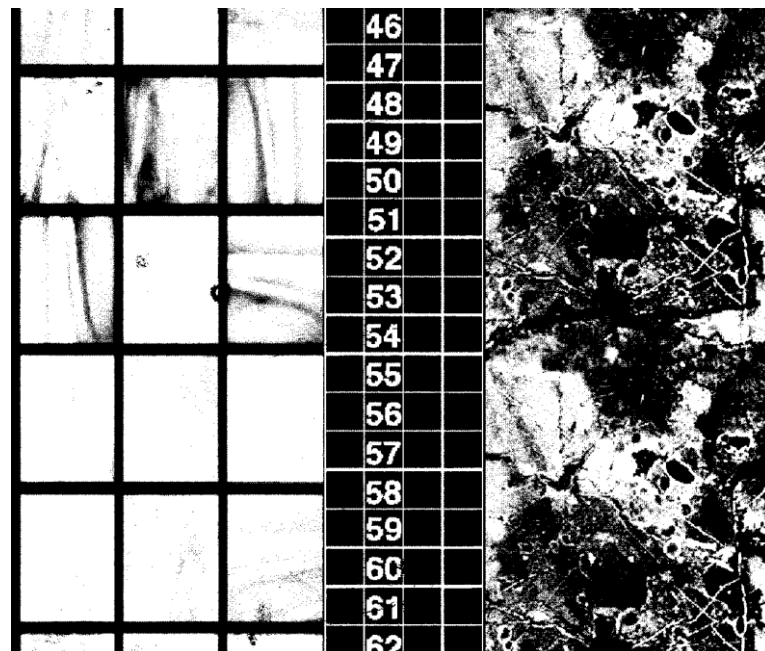
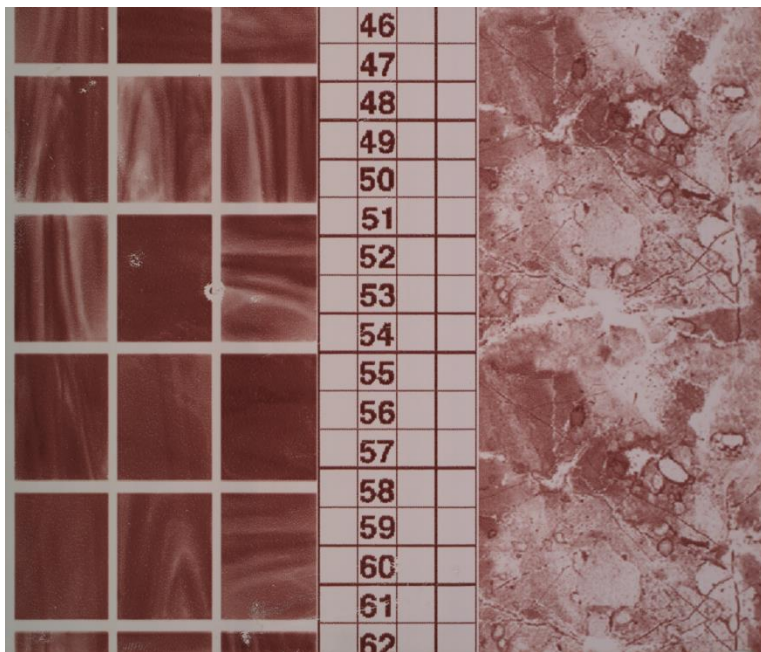
$$\Sigma = \mathbf{R} \mathbf{D} \mathbf{R}^T : \quad \mathbf{R} = (\mathbf{e}_1 \ \mathbf{e}_2 \ \mathbf{e}_3), \quad \mathbf{D} = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix}$$

Where \mathbf{e}_i are the (orthonormal) *eigenvectors* of Σ , λ_i the corresponding *eigenvalues* and \mathbf{R}^T the rotation matrix to transform the data into a new coordinate system having axes aligned to the eigenvectors.

Upon rotation by \mathbf{R}^T the new covariance matrix becomes \mathbf{D} , so that the eigenvalues of Σ represent the variances along the direction of the eigenvectors.

(More details concerning the case of general covariance matrix can be found in the appendix)

Example



$I(p)$



$$O(p) = \begin{cases} 255, & d_M(I(p), \mu) \leq T \\ 0, & d_M(I(p), \mu) > T \end{cases}$$

Appendix - Mahalanobis distance with general covariance matrix (1)



$$d_M(\mathbf{I}(p), \boldsymbol{\mu})^2 = (\mathbf{I}(p) - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{I}(p) - \boldsymbol{\mu})$$

$$\boldsymbol{\Sigma} = \mathbf{R} \mathbf{D} \mathbf{R}^T \rightarrow \boldsymbol{\Sigma}^{-1} = (\mathbf{R} \mathbf{D} \mathbf{R}^T)^{-1} = (\mathbf{R}^T)^{-1} (\mathbf{D})^{-1} = \mathbf{R} \mathbf{D}^{-1} \mathbf{R}^T$$

Rotation bringing $\mathbf{I}(p)$ and $\boldsymbol{\mu}$ into the new coordinate system with axes aligned to the eigenvectors of $\boldsymbol{\Sigma}$

$$d_M(\mathbf{I}(p), \boldsymbol{\mu})^2 = (\mathbf{I}(p) - \boldsymbol{\mu})^T \mathbf{R} \mathbf{D}^{-1} \mathbf{R}^T (\mathbf{I}(p) - \boldsymbol{\mu})$$

$$[\mathbf{I}^T \mathbf{e}_1 - \boldsymbol{\mu}^T \mathbf{e}_1 \quad \mathbf{I}^T \mathbf{e}_2 - \boldsymbol{\mu}^T \mathbf{e}_2 \quad \mathbf{I}^T \mathbf{e}_3 - \boldsymbol{\mu}^T \mathbf{e}_3]$$

$$= [\mathbf{e}_1^T \mathbf{I} - \mathbf{e}_1^T \boldsymbol{\mu} \quad \mathbf{e}_2^T \mathbf{I} - \mathbf{e}_2^T \boldsymbol{\mu} \quad \mathbf{e}_3^T \mathbf{I} - \mathbf{e}_3^T \boldsymbol{\mu}]$$

$$(\mathbf{I}'(p) - \boldsymbol{\mu}')^T$$

$$[I_{r'}(p) - \mu_{r'} \quad I_{g'}(p) - \mu_{g'} \quad I_{b'}(p) - \mu_{b'}]$$

$$(\mathbf{I}'(p) - \boldsymbol{\mu}') = \begin{bmatrix} \mathbf{e}_1^T \mathbf{I} \\ \mathbf{e}_2^T \mathbf{I} \\ \mathbf{e}_3^T \mathbf{I} \end{bmatrix} - \begin{bmatrix} \mathbf{e}_1^T \boldsymbol{\mu} \\ \mathbf{e}_2^T \boldsymbol{\mu} \\ \mathbf{e}_3^T \boldsymbol{\mu} \end{bmatrix}$$

$$\mathbf{D}^{-1}(\mathbf{I}'(p) - \boldsymbol{\mu}') = \begin{bmatrix} \frac{I_{r'}(p) - \mu_{r'}}{\lambda_1} \\ \frac{I_{g'}(p) - \mu_{g'}}{\lambda_2} \\ \frac{I_{b'}(p) - \mu_{b'}}{\lambda_3} \end{bmatrix}$$

Appendix - Mahalanobis distance with general covariance matrix (2)



$$d_M(\mathbf{I}(p), \boldsymbol{\mu})^2 = \begin{bmatrix} I_{r'}(p) - \mu_{r'} & I_{g'}(p) - \mu_{g'} & I_{b'}(p) - \mu_{b'} \end{bmatrix} \begin{bmatrix} \frac{I_{r'}(p) - \mu_{r'}}{\lambda_1} \\ \frac{I_{g'}(p) - \mu_{g'}}{\lambda_2} \\ \frac{I_{b'}(p) - \mu_{b'}}{\lambda_3} \end{bmatrix}$$



$$d_M(\mathbf{I}(p), \boldsymbol{\mu})^2 = \frac{(I_{r'}(p) - \mu_{r'})^2}{\lambda_1} + \frac{(I_{g'}(p) - \mu_{g'})^2}{\lambda_2} + \frac{(I_{b'}(p) - \mu_{b'})^2}{\lambda_3}$$

$$d_M(\mathbf{I}(p), \boldsymbol{\mu})^2 = T^2 \quad \longrightarrow \quad \text{Ellipsoid with semi-axes of length } T\sqrt{\lambda_i} \text{ and axes aligned to the eigenvectors of } \Sigma$$

Hence, in the general case, the contributions to the overall distance are inversely proportional to the eigenvalues of Σ , which in turn represent the variances of the data along the directions of the eigenvectors.

Appendix - Mahalanobis distance with general covariance matrix (3)



Eventually, we verify here that in the new coordinate system obtained by a rotation through \mathbf{R}^T , the new covariance matrix, Σ' , is diagonal and its non-null elements are the eigenvalues of Σ .

Given an n -dimensional random vector \mathbf{x} , with mean and covariance denoted as \mathbf{m}_x and \mathbf{C}_x respectively, let us consider a generic linear transformation defined through the $n \times n$ matrix \mathbf{A} :

$$\mathbf{y} = \mathbf{A}\mathbf{x}$$

It can be shown that the mean and covariance of the transformed data are given by:

$$\mathbf{m}_y = \mathbf{A}\mathbf{m}_x, \quad \mathbf{C}_y = \mathbf{A}\mathbf{C}_x\mathbf{A}^T$$

Therefore, in the considered case :

$$\Sigma' = \mathbf{R}^T \Sigma \mathbf{R} = \mathbf{R}^T (\mathbf{R} \mathbf{D} \mathbf{R}^T) \mathbf{R} = (\mathbf{R}^T \mathbf{R}) \mathbf{D} (\mathbf{R}^T \mathbf{R}) = \mathbf{I} \mathbf{D} \mathbf{I} = \mathbf{D} = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix}$$

with \mathbf{I} denoting here the 3×3 identity matrix: $\mathbf{I} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

Main References



- 1) R. Gonzales, R. Woods, “Digital Image Processing – Third Edition”, Pearson Prentice-Hall, 2008.
- 2) R. Fisher, S. Perkins, A. Walker, E. Wolfart, “Hypermedia Image Processing Reference”, Wiley, 1996 (<http://homepages.inf.ed.ac.uk/rbf/HIPR2/>)
- 3) R. Haralick, L. Shapiro “Computer and Robot Vision – Vol. I, Addison-Wesley Publishing Company, 1993.
- 4) A. Rosenfeld, A. Kak, “Digital Picture Processing – Vol. 2, Academic Press, 1982.