



Universidad de Valladolid



**ESCUELA DE INGENIERÍAS
INDUSTRIALES**

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería de electrónica industrial y automática

**Detection of presence, position and correct
positioning of components on a PCB by image
processing.**

Autor:

Querol Gil, Antonio

Responsable de intercambio en la Uva

Ángel Manuel Gento Municio

Universidad de destino

Albstadt-Sigmaringen University

Valladolid, septiembre 2022.

TFG REALIZADO EN PROGRAMA DE INTERCAMBIO

TÍTULO: Detection of presence, position and correct positioning of components on a PCB by image processing.

ALUMNO: Antonio Querol Gil

FECHA: 25 de agosto de 2022

CENTRO: Fakultät für Informatik (facultad de informática)

UNIVERSIDAD: Albstadt-Sigmaringen University

TUTOR: Bernd Stauß

RESUMEN Y PALABRAS CLAVE

Un recorrido por el funcionamiento del procesamiento de imagen y sus diferentes procesos, entrando en diferentes funcionamiento matemáticos de los mismos. Además, voy un poco más allá entrando en procesos adyacentes no imprescindible de el proceso de procesar una imagen, como puede ser el tratamiento que se hace del color o como se realiza un realce de una imagen durante el procesamiento de esta. Por último, se cuenta el funcionamiento de las redes neuronales, tanto como funciona matemática y lógicamente una neurona por si sola y un esquema de esta. Añadiendo como se realizaría una red neuronal para el procesamiento de imagen en Python.

Image, Neuronal, Processing, Python, Segmentation



Hochschule
Albstadt-Sigmaringen
Albstadt-Sigmaringen University

FINAL THESIS

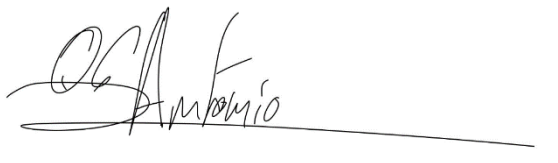
**Detection of presence, position and
correct positioning of components
on a PCB by image processing.**

AUTHOR: ANTONIO QUEROL

PROFFESOR: BERND STAUB

I declare that this thesis has been carried out without external assistance and using only the sources and bibliography indicated in this thesis.

Signed by Antonio Querol Gil



A handwritten signature in black ink, appearing to read 'Antonio Querol Gil', is written above a solid horizontal line. The signature is stylized and cursive.

INDEX

1.INTRODUCTION.....	8
1.1. Concepts you must know	8
1.2. Introduction to the Thesis.....	9
1.2.1. Overview	9
1.2.2. Materials required.....	9
1.2.3. Market research.....	10
1.2.4. Objectives.....	11
1.2.5 Possible improvements.....	12
1.3 Visit to EPIS.....	13
2.PROJECT BEGINNING.....	14
2.1. Introduction.....	14
2.2. Precedents.....	15
2.3. Processes Classification.....	16
3.PROCESSES.....	18
1.Adquisition.....	18
2.Preprocessing.....	20
3.Segmentation.....	20
3.1.Discontinuity segmentation.....	20
3.2.Similarity segmentation.....	28
4.Representation and description.....	40
5.Recognition and interpretation.....	53
4.Image enhancement.....	55
5.Color in images.....	64
1.RGB Model.....	65
2.CMYK Model.....	66
3.YIQ Model.....	67
4.HSI Model.....	67

6.LABELLING.....	69
7.NEURAL NETWORK.....	72
1.Introduction.....	72
1.1. Face and facial expression recognition.....	72
1.2. Signature recognition.....	73
1.3. Character recognition.....	74
2.Automatic learning.....	75
2.1. Learning methods.....	75
2.2. Training and evaluation sets.....	76
3.What is an artificial neural network?.....	77
3.1. Artificial Neuron.....	77
3.2. Connection architecture.....	79
3.3. Feed-Forward Networks.....	80
4.Network learning.....	82
4.1. Activation function.....	82
5.Example.....	83
8.REFERENCES AND BIBLIOGRAPHY.....	88

Images index

Image 1- Camera photo.....	9
Image2- Components photo.....	9
Image3- Spotlight photo.....	10
Image4- PCBs photo.....	10
Image5- Possible improvements scheme.....	12
Image6- Possible improvements scheme.....	12
Image7- Influence's areas image processing.....	16
Image8- Type of lights.....	19
Image9- Type of sources.....	19
Image10- Radiography of a human body (RHB).....	21
Image11- Point discontinuity of a RHB.....	21
Image12- Same image as Image10.....	22
Image13- Horizontal discontinuity of a RHB.....	22
Image14- Vertical discontinuity of a RHB.....	22
Image15- 135° discontinuity of a RHB.....	23
Image16- 45° discontinuity of a RHB.....	23
Image17- Same image as Image10.....	25
Image18- RHB segmentate with Roberts' operator.....	25
Image19- Same image as Image10.....	26
Image20- Vertical segmentation by Sobel's operator.....	26
Image21- Horizontal segmentation by Sobel's operator.....	26
Image22- Building photo.....	27
Image23- Building after Laplacian.....	27
Image24 -Building after Laplacian and subtraction.....	27
Image25- Building after Laplacian of a Gaussian function.....	28
Image26- Histogram example threshold.....	29
Image27- An image of random book.....	35
Image28- Book image after global thresholding.....	35
Image29- 2 seed points of different regions.....	38

Image30- Result in growing regions with $T=3$	38
Image31- Result in growing regions with $T=8$	38
Image32- Example of a brain with different thresholds.....	39
Image33- Chain codes structure.....	42
Image34 and 35- Chain code examples.....	42
Image36- Polygonal example.....	43
Image37- Another polygonal example.....	44
Image38- Signature example.....	45
Image39- Skeleton of a region example.....	47
Image40- Shape numbers example, order 4,6 and 8.....	50
Image41- Topological examples.....	52
Image42- Another topological example.....	52
Image43- Transformation function for get the negative of an image.....	56
Image44 and 45- Example of a transformation function(TF).....	57
Image46 and 47- Another TF for grey fractionation.....	57
Image48- Zebras photo.....	58
Image49- Histogram of Zebras photo.....	58
Image50- Zebras photo after 64-level equalization.....	59
Image51- Histogram of the equalized photo.....	59
Image52- Scheme of a spatial filter.....	62
Image53- Scheme of a frequency filter.....	63
Image54- Electromagnetic spectrum.....	64
Image55- RGB model.....	66
Image56 and 57- Labelling code.....	69
Image58- Labelling code.....	70
Image59- Labelling code.....	71
Image60- How does a neuron work?.....	77
Image61- 3-layer architecture perceptron.....	81
Image62- Representation of different activation function.....	82
Image63 and 64- Neuronal network code.....	83
Image65- Neuronal network code.....	84
Image66- Neuronal network code.....	85
Image67 and 68- Neuronal network code.....	86

Image69 and 70- Neuronal network code.....87

1. INTRODUCTION

1.1. Concepts you must know:

Mounting: Action in which several specific pieces are placed on a prefixed assembly space to obtain a final product which performs the required mission.

Vision System: This is a set of hardware and software that has the capacity to detect different objects (including the relative position of one object respect to another, the correct position of that object or whether it is present or not) through a camera.

Production: Process in which an object is manufactured through different processes, which make up a larger process, in which different raw materials are manipulated to obtain the final product, which must have special characteristics. These characteristics must be predefined before the manufacturing process, as this process must be designed beforehand.

Programming: The process of writing code in the appropriate language to carry out the task as efficiently as possible.

Image: Is a representation or an imitation of an object. Contain descriptive information of the object is represented on it.

Digital: It relates to calculation by numerical methods or through discrete units.

Digital Image: After define Image and Digital, we conclude that a Digital Image is a numeric representation of an object.

Image processing: The set of techniques that are applied to digital images with the aim of improving the quality or facilitating the search for information.

1.2. Introduction to the Thesis

- 1.2.1. Overview

We will carry out a thesis in which we will demonstrate how to create an artificial vision system for the detection of different objects. This system will be able to detect if a particular object is there or not, and if it is there, it will inform if the position is the right one. Finally, the system will have to provide the information we need (For example: number of pieces, pieces which are in the system etc.).

- 1.2.2. Materials.



Image 1

USB camera 2.0 Megapixels with variable focal length (2.8-12mm)

Different components. Those are which we will study.



Image 2

Spotlight



Image 3



Image 4

2 PCBs. One of those without components and the other one with components.

- 1.2.3. Market research

After an exhaustive analysis of a large part of the market, I have been able to reach the conclusion that there is little or no real competition for this system. To all this we must add that there are complications in finding real competition for this system, as many of the implementations of this type of system are carried out in machinery that, in addition to the functions that our system will perform, also perform many other functions that ours does not, as our system is a simple system that is easy to adapt to different products.

We will not end this section without first commenting that we are looking for a system that is flexible, as we have mentioned before, as well as simple, which implies that it should be low cost, easy to maintain and easy to improve if an element is found that, due to its characteristics, greatly improves the efficiency of the system.

- **1.2.4. Objectives**

1. Create a system that will be able to determine whether a component is on the PCB and, if so, whether it is in the corresponding position.
2. Learn about software development related to the world of image processing and artificial intelligence.
3. Learn how to develop a project from scratch since we only have the idea and the mental structure of the project.
4. Improve own skills in the field of software development and problem solving.

- 1.2.5. Possible improvements

With this system, a control system can be realized, which would imply a highly efficient automation, which can feedback three types of parts, as shown in the following image:

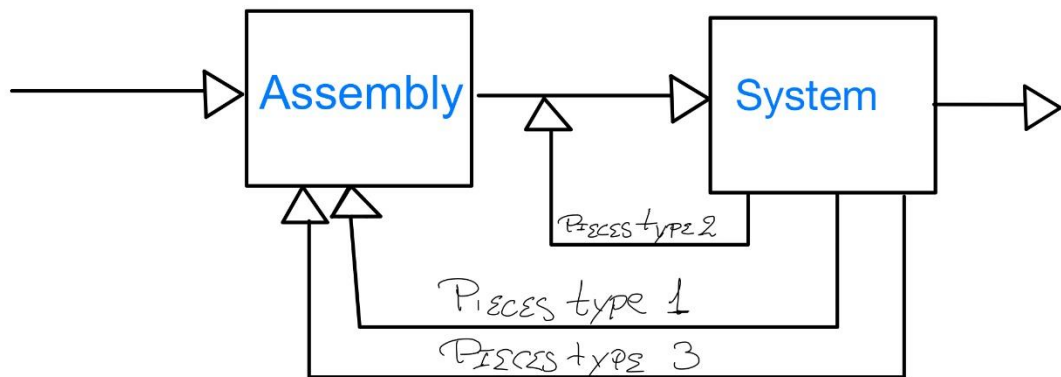


Image 5

-Pieces type 1: Pieces detected as incomplete, which could be automatically returned to the assembly part of the part.

-Pieces type 2: Parts which are complete but have an error in orientation shall be returned to the beginning of the system for further recognition.

-Pieces type 3: In case of a new positioning error, they shall be returned to the assembly part.

Another possible improvement that we can add, once we have already detected that the part is correct, is an automatic checking system, which makes the corresponding electrical checks and in case of failure it is taken to the repair department. So, the general system, if the two improvements mentioned above are applied, would be as follows:

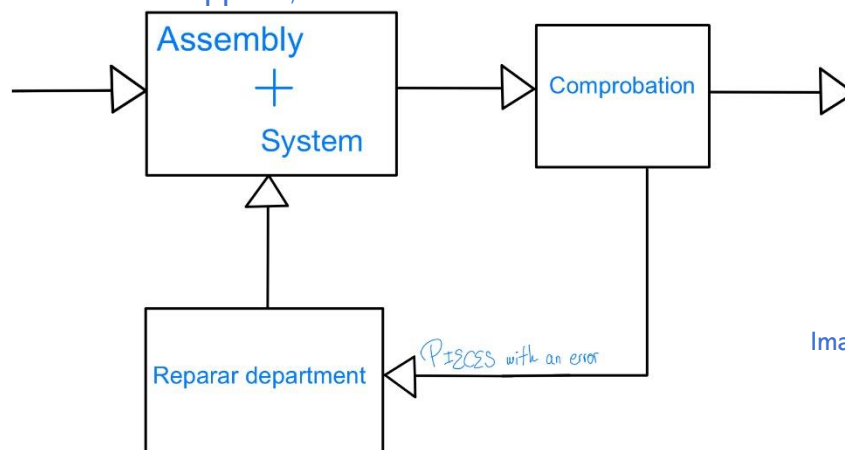


Image 6

1.3. Visit to EPIS

We went because we wanted to see a clear example of what we want to do. Initially, we must know that they have different mounting sites where they did several types of PCBs and they putted in the same transport line. Once I knew that, of all the processes they had there, two were the ones that surprised me the most:

-How PCB's were soldered. That thing made me think how they could do it. Due to my inexperience in solder processes, at the beginning, my idea was that the welding machine changed with every PCB but later, talking with the responsible person, I discovered that the welding machine did not change anything, it is only a wave that soaks the area under the PCB and, when it comes out of the machine, it is already welded.

-How PCB's were checked to see if a component was missing. Once the PCB was mounted, it was placed in a special tray where an especial element is assembled above the tray and PCBS and it just closes if all the pieces are on the PCBS on the correct position. After this check, they putted the tray on the line to go to the welding process.

In the company certain changes could be applied for the improvement of the welding line system. We then proceed to discuss some ideas that I have:

- The first one is: The implementation of the system we are going to design, together with the improvements mentioned in the previous section. This will require, apart from experts in the field, a remodeling of the plant as there will be processes that will now be carried out automatically.
- The second one is: The remodeling of the tray collection system. In case the first idea is too big and cannot be carried out in the short or medium term, the tray pick-up from the conveyor belt should be automated, as this repetitive movement is very damaging to the workers and therefore, in the long term, will lead to injuries to them.

2.PROJECT'S BEGINNING

2.1. Introduction

To explain the theoretical functioning of image processing, it is important to discuss why we want to do this processing to obtain information. This is because our vision allows us to do this processing easily, quickly and moreover, in a super-detailed way. The only problem with human vision is that it does not allow us to store this information permanently and it is possible that the information obtained over a period, whether short or long, is lost and cannot be used. As mentioned in the beginning, image processing deals with two main functions:

- Altering visual information for enhancement.
- Isolate some particular characteristics of the images.

There are three types of image processing:

- Optical processing.
- Analogue processing.
- Digital processing.

There are several factors driving the advance of digital image processing, of which two of the most important are:

- Reduction of the cost of computational equipment.
- Increased availability of digital imaging equipment in the market. mercado.

In addition, innovative technologies are entering the market, such as matrix processors, high-resolution CCDs* or artificial intelligence neural networks.

High Resolution CCD*: As far as our field of study is concerned, the CCD is the sensor that has photoelectric cells that record the image, which is transmitted and then stored in memory.

2.2. Precedents

It can be said that the first attempts at image processing came about in 1920, at which time a system for transmitting photographs via a transatlantic cableway was invented. It allowed the encoding of an image in five grey levels. Improved to fifteen greys in 1929. This invention improved the delivery of quality images from 15 days (using newspapers) to 15 minutes.

In 1964, these techniques were reconsidered due to the processing of images received from one of the satellites transmitting from the moon. Only this image was treated to remove distortions produced by the TV cameras, such as:

- The geometric distortion produced by the difference in scanning speeds between the vidicon* of the probe and the reproducing tube on Earth.
- Photometric non-linearity due to irregular tube response.
- Oscillatory noise due to contamination of the television signal by the probe electronics.

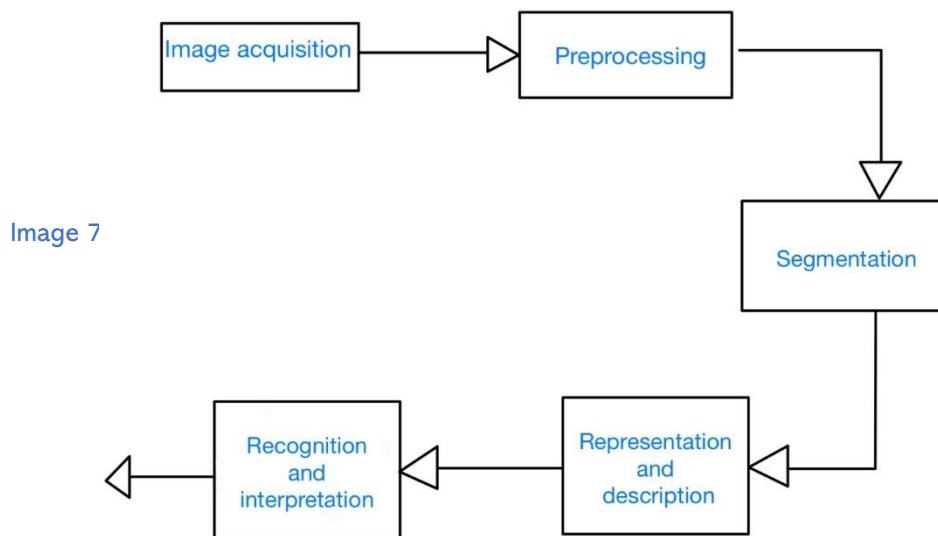
Vidicon*: In television, a tube that takes images very sensitively, used for hand-held cameras. Also used for closed-circuit connections, traffic control cameras.

Domingo Mery Youtube channel. 02 Procesamiento de imagenes:Introducción-
<https://www.youtube.com/watch?v=ou1R9UbwP00>

2.3. Different Classifications

We define artificial vision as the processes of obtaining, characterizing and interpreting information from images taken from a three-dimensional world. These processes are divided into five areas of influence:

- Acquisition
- Preprocessing
- Segmentation
- Representation and Description
- Recognition and Interpretation



What does each area do?

- Acquisition: Process from which the image is obtained.
- Preprocessing: Includes different image enhancement techniques (e.g., noise reduction or detail enhancement).
- Segmentation: Divides the image into objects of interest.
- Description: The necessary characteristics are obtained to differentiate one object from another.
- Recognition: Process that identifies the objects.
- Interpretation: Assigns meaning to a set of recognized objects.

These processes can be divided into three distinct stages:

-Low-level vision: These are the processes that we can consider as automatic reactions. These processes usually do not have intelligence requirements, such as acquisition and pre-processing.

Medium-level vision: This is associated with the processes that extract, characterize and label the different components of the image obtained in the previous stage. This includes the processes of segmentation, description and recognition.

High-level vision: Related to processes that attempt to reproduce cognition. While the processes included in the previous two stages are known and well-defined processes, the processes in this stage are more diffuse and subjective.

(Eugene Woods, R., Gonzalez, R., Image processing, Pearson, New York, 2018, p.41, 42, 43)

3. Processes

1 Acquisition: Because computers can only process digital images, WE must convert the images.

An image digitizer must be able to divide the image into small regions called pixels, measure the grey level of the image for each pixel (intensity), quantize that continuous measurement to produce an integer value, and then write this data to a storage device.

The most common subdivision scheme is the rectangular sampling grid in which the image is subdivided into horizontal lines consisting of a certain number of adjacent pixels. Because of this, the digitizer must have five fundamental elements:

- Sampler
- Scanning mechanism
- Brightness sensor
- Quantizer
- Output way.

The most important characteristics of a digitizer are:

- Sampling width
- Sampling frequency
- Linearity of digitization
- Number of grey levels

1.1. Lighting techniques and sources.

Most used techniques:

- Diffused lighting: Used for objects with smooth surfaces.
- Backlighting: When the important part of the object is its shadow.
- Directional illumination: Used for the inspection of object surfaces.
- Front illumination: Widely used for presence/absence detection.

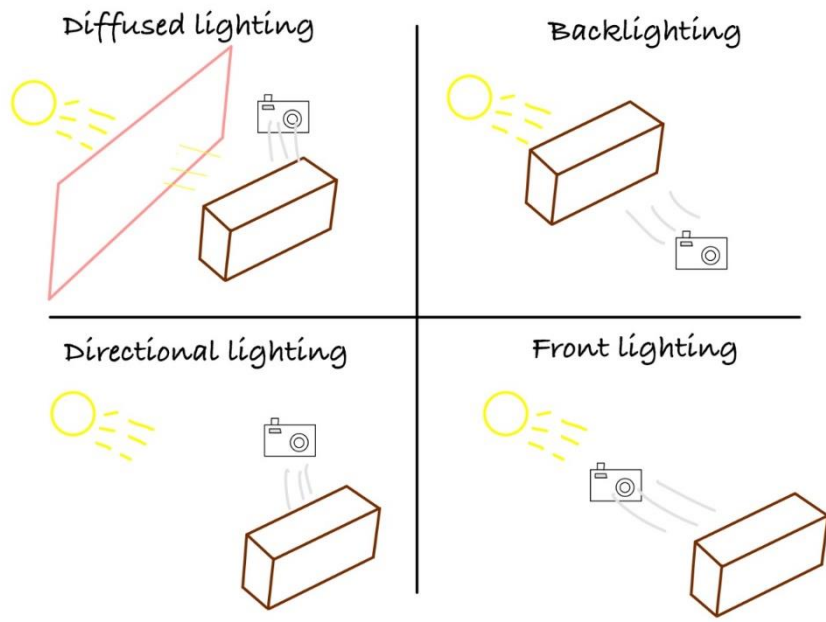


Image 8

1.2. Most used sources:

- Incandescent lamps
- Fluorescent lamps
- Fiber optics
- Lasers (the most powerful)
- Flashes
- Ultraviolet light



Image 9

2 Preprocessing

It aims to improve the appearance of the images and to make certain details more evident in the images that you want to be noticed.

To do the preprocessing, the image must be divided according to a predefined homogenization feature (e.g., image binarization or edge detection technology).

3 Segmentation [2]

The goal of segmentation is to partition the image into meaningful regions. These segmentation methods assume that these regions possess some distinctive homogeneous characteristics. Therefore, there may be physical boundaries between similar regions that do not appear in the image.

Translating this, we can consider segmentation as a problem in which we must discern what the regions are or whether a pixel belongs to one region or another.

The main segmentation techniques are:

3.1 Discontinuity segmentation.

This technique divides an image using as a basis the abrupt changes in the grey level. We can distinguish three types of discontinuities: points, lines and edges.

The most common way to see discontinuities is to pass the data through a mask*.

3.1.1. Point discontinuity: this discontinuity is direct. Using a mask, you can tell if a point has been detected at the position where it is centered if,

$$|R| > T$$

Where T is a non-negative threshold and R is the measurement with the mask of the evaluated pixel.

(Point discontinuity) Marcos Martín. Tecnicas de segmentacion clasica. Archivo electronico. <http://poseidon.tel.uva.es/~carlos/ltif10001/segmenclasica.pdf> (last time:27/8/2022)

(Eugene Woods, R., Gonzalez, R., Image processing, Pearson, New York,2018,p.706-707)

The most common mask is:

$$\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Image: $\begin{pmatrix} 86 & 85 & 25 \\ 125 & 214 & 126 \\ 25 & 14 & 25 \end{pmatrix}$

Grey levels \rightarrow

Mask: $\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$

$$\begin{aligned} \text{Result} &= (-86 - 85 - 25 - 125 - 126 - 25 - 14 - 25) + 1172 \\ &= 1286 \end{aligned}$$

We do a convolution operation and if the result obtained meets the above condition then it will be a point.

One example could be the following images:



Image 10

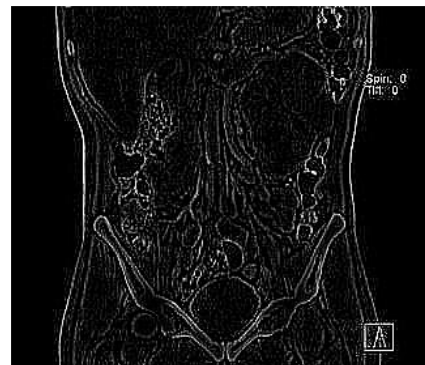


Image 11

Where the first image is the original and the second one is after the point detection.

3.1.2. Lines discontinuity: We will use different masks depending on the angle at which we want to detect the lines.

(Lines and border discontinuity) Marcos Martín. Tecnicas de segmentacion clasica. Archivo electronico. <http://poseidon.tel.uva.es/~carlos/ltif10001/segmenclasica.pdf> (last time:27/8/2022)

(Eugene Woods, R., Gonzalez, R., Image processing, Pearson, New York,2018,p.707-715)

If we wanted to detect horizontal lines, we would use the following mask:

$$\begin{pmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{pmatrix}$$

If we wanted vertical lines:

$$\begin{pmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{pmatrix}$$

An also we can do it for different angles like 45° or 135°:

$$\begin{pmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{pmatrix} \text{ or } \begin{pmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{pmatrix}$$

Let R1, R2, R3 and R4 be the responses of the masks in the central pixel of an image. If at a point in the image it is true:

$$|R_i| > |R_j| \quad i \neq j$$

Where point i is the most likely to be associated with a line in the direction of the i mask.

Example:

Original



Image 12

Horizontal and vertical

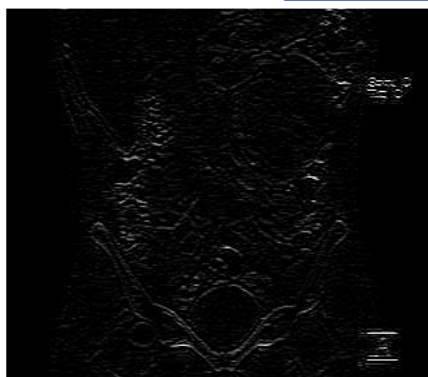


Image 13 and 14

135° and 45°

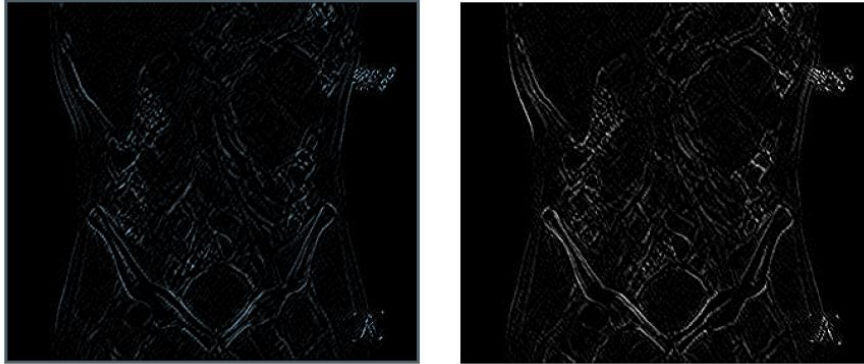


Image 15
and 16

3.1.3. Border discontinuity: In this case we can comment on several things:

- An edge is a boundary between two grey levels.
- Normally, operators like the derivative are used.
- The magnitude of the first derivative is always zero except at points where there is a transition between grey levels.
- The sign of the second derivative will determine which side of the edge the pixel is on.
- Edge enhancement can be calculated from spatial filtering performed by convolution mask methods, the most used of which are:

- Displacement and subtraction
- Gradient
- Laplacian

3.1.3.1. *Displacement and subtraction* is the simplest of the three. It is used for the detection of vertical and horizontal edges. The method is simple. To detect vertical edges, you shift the image to the left and subtract the original ones, which is equivalent to

performing a drift operation. The same should be done to detect horizontal edges by changing the left shift to a downward shift.

3.1.3.2. The *gradient* of an image $f(x,y)$ at position (x,y) is the vector:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

The mathematical definition of gradient is taken from vector analysis, in which the gradient is defined as the direction of the maximum variation of the function f at the point (x,y) .

One value that is important in the field of edge detection is that of magnitude:

$$\text{mag}(\nabla f) = \sqrt{(G_x^2 + G_y^2)}$$

The direction of the gradient vector is also important:

$$\alpha(x, y) = \tan^{-1} \left(\frac{G_y}{G_x} \right)$$

The calculation of derivatives in digital form can be done in different ways, always conveniently specifying the different masks. The simplest are:

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix} \text{ and } [1 \quad -1]$$

These result in the following equations of the derivatives:

$$G_x = f(x, y) - f(x + 1, y)$$

$$G_y = f(x, y) - f(x, y + 1)$$

Calculating the derivatives in this way makes the gradient calculation very sensitive to local orientations.

This is solved by *Roberts' Operator*, which analyses two different directions at right angles to calculate the gradient. Looking at it from a matrix point of view, the Roberts' Operator is as follows:

$$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \text{ and } \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

The biggest problem with this operator is that it is sensitive to directions and noise. In the following images we can see the original photo of a body and after segmenting it with Roberts' operator.

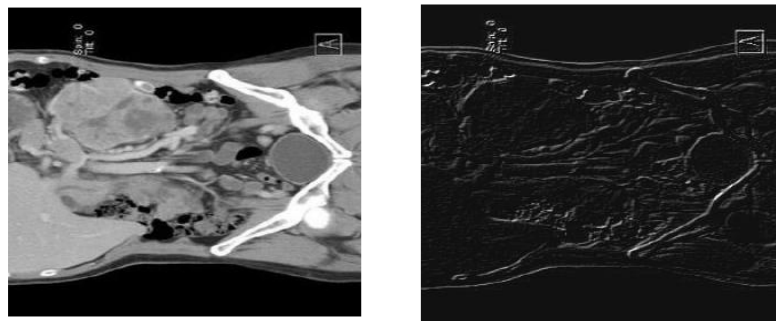


Image 17
and 18

As we can see in the second image, there are many details missing that cannot be observed due to the weakness of Roberts' operator in direction changes and noise.

By calculating the derivatives in two directions and combining them as the square root of the sum of the squares of the derivatives we obtain a result that is independent of orientation, solving the problem that Roberts' operator had with changes of direction. So, we could say that this is what Sobel's operators take advantage of.

The general shape of the masks can be formulated as follows:

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \text{ and } \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

In practice, we will only calculate the derivatives in two orthogonal directions and then calculate the modulus. The advantage of using 3x3 pixel masks is that you have an additional smoothing that 2x2 Roberts-type operators do not have.

(Gradient and Laplacian)Técnicas de segmentacion clasica.Archivo electronico.

<http://poseidon.tel.uva.es/~carlos/ltif10001/segmenclasica.pdf> (last time:27/8/2022)

We can see in the following images the change with respect to Roberts' operator:

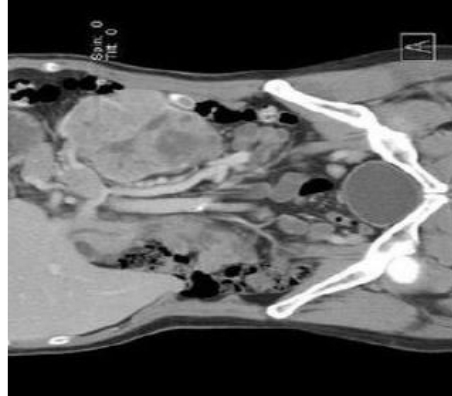


Image 19

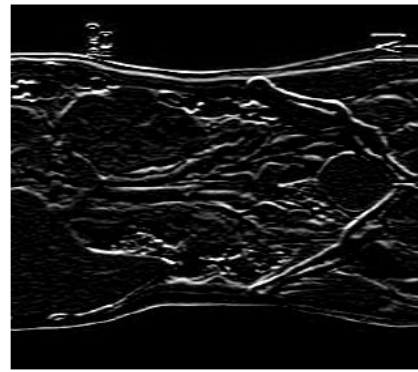
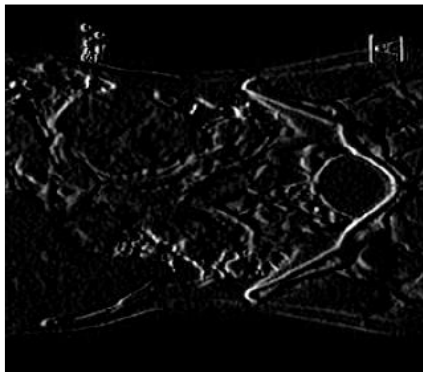


Image 20
and 21

The images 20 and 21 correspond to the segmentation by Sobel operator, both vertically and horizontally.

3.1.3.3. The *Laplacian* is an elliptic differential operator of the second degree. To study a two-dimensional function as in our case we must use the following formula:

$$\nabla^2 f = \frac{\partial^2 f}{\partial^2 x} + \frac{\partial^2 f}{\partial^2 y}$$

And the most frequently used masks are:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \text{ and } \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Normally, positive and negative values can be obtained, so an average grey value is added to the image so that the values at zero remain at that value and, however, the darker values correspond to the negative values produced by this operation. Subtracting this image will produce an increase in contrast in the discontinuities.



Image 22

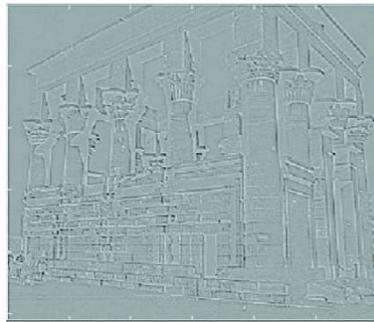


Image 23



Image 24

In the following images we can see:

- 1º-The original image.
- 2º- The image once we apply the Laplacian.
- 3º- The 2nd image once we apply the subtraction previously mentioned.

As we can see in the second image, the Laplacian is very sensitive to noise. We already knew this because it is an operator made up of partial derivatives of the second degree, which means that small distortions can produce large changes.

It also produces double edges. A more correct use of this operator is to find the location of the edge by exploiting its zero-crossing property. This concept is based on the convolution of an image with the Laplacian of a Gaussian function with the following form:

$$h(x, y) = \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

The resulting Laplacian operator is also known as the *Marr-Hildreth Operator*.

$$\nabla^2 h = \left(\frac{r^2 - \sigma^2}{\sigma^4}\right) \exp\left(-\frac{r^2}{2\sigma^2}\right) \quad r^2 = x^2 + y^2$$

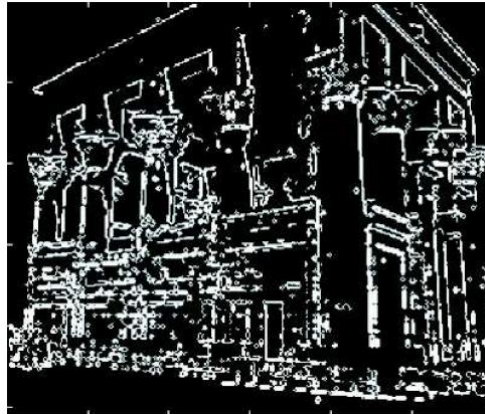
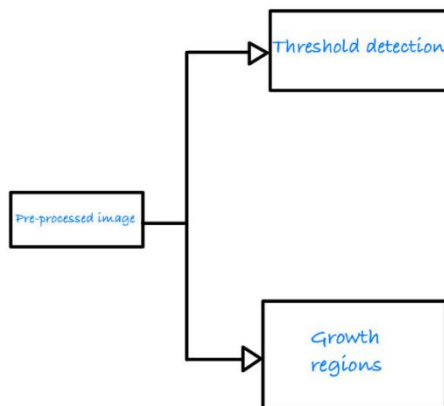


Image 25

3.2. Similarity segmentation.



We will perform threshold detection when the grey levels of the objects in the background are easily distinguishable. However, when it is not possible to distinguish the objects from the background, we will perform a luminance gradient study.

3.2.1. Thresholding:

It provides an easy and convenient way to perform a separation of image regions that correspond to objects of interest based on different intensities or colors between the background and the objects.

How it works: Normally the input to a thresholding operation is a grey level image or a color image and the output is a binary image representing the segmentation.

The black pixels will correspond to the background and the lighter pixels will correspond to the foreground of the image. When we are working in simple applications, the criterion to differentiate whether a pixel corresponds to the background or to the object is only one and it is called intensity threshold.

In more complex applications, different thresholds or multiple thresholds can be specified, where one band can be set to white and the rest to black.

To know how many pixels there are as a function of their intensity, we use histograms.

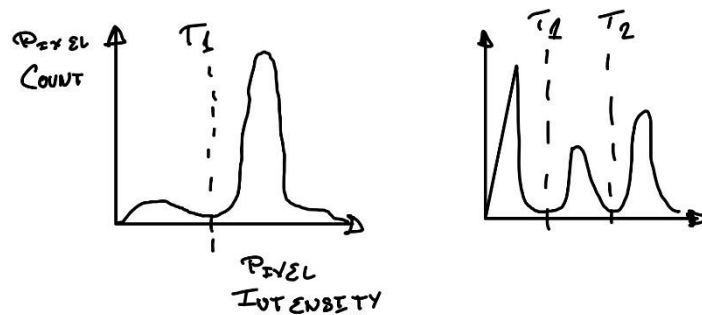


Image 26

As we can see in the first histogram, we have only one intensity threshold and therefore all pixels above this threshold will be painted white. However, in the second image we can see that we have two different intensity thresholds in which pixels whose intensity falls within this threshold band will be painted white.

For color images it is possible to place different thresholds so that each space within the pixel intensity domain corresponds to a color within the RGB space. Another common variant is to paint only the background black and keep the information of the pixels in the foreground of the image.

Not all images can be segmented with the thresholding technique. This will depend on how the intensity histogram is arranged. Therefore, we will have to analyze the histogram to decide. For a successful thresholding segmentation, we must be able to separate the intensity of the background from the intensity of the foreground in the histogram. Looking at image twenty-eight, we see that in both the first and the second case, the foreground intensity peak can be separated from the background intensity peak, so this segmentation with these histograms would be satisfactory.

In case such a separation is not possible, segmentation by thresholding can be done by other methods that will be discussed later, but this has its drawbacks: Increased computational time and requirements.

(Introduction thresholding) http://asignatura.us.es/imagendigital/Tema5-2_SegmentacionRegionesUmbralizacion.pdf (last time:27/8/2022)

3.2.1.1. Techniques

3.2.1.1.1. Simple Global Thresholding

This is the simplest technique in which the histogram of an image is simply divided by an intensity threshold T . The segmentation is completed by checking pixel by pixel and labelling each pixel whether it belongs to the background or to the object of interest. Depending on whether the grey level of the analyzed pixel is higher or lower than the threshold T , the pixel will belong to the object or to the background. The success of this technique depends, as mentioned above, on the possibility to divide the histogram appropriately.

The aim of this segmentation is to produce a binary image. Therefore, it will only be possible to do this in very controlled spaces where the lighting can be properly controlled, as poor lighting will result in a histogram impossible to divide properly.

3.2.1.1.2. Optimal Thresholding

Suppose an image contains only 2 main brightness regions. Then the histogram of such an image can be considered as an estimate of the brightness probability density function $p(z)$. This total density function is the mixture of two unimodal probabilities. The parameters of this mixture depend directly on the surface of the image brightnesses.

Suppose an image contains two values combined with additive Gaussian noise. The probability function will look like this:

$$p(z) = P_1p_1(z) + P_2p_2(z)$$

which for the Gaussian case would be:

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

Where μ_1 y μ_2 are the mean values of the two brightness levels, σ_1 y σ_2 are the standard deviations from the mean and P_1 y P_2 are the probabilities of the two levels, which must satisfy the following constraint:

$$P_1 + P_2 = 1$$

In relation to the errors that can be made when classifying the different points, we can say:

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

This equation indicates the probability of assigning a point on the object the category of point in the background.

For the opposite case, a very similar equation remains:

$$E_2(T) = \int_T^{\infty} p_1(z) dz$$

Therefore, the total error is:

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

To obtain the most optimal threshold value at which the error is minimized, we must derive the above equation and set it to 0. This results in the following equation:

$$P_1 p_2(T) = P_2 p_1(T)$$

(Optimal thresholding) Djemel Zlou. Optimal Thresholding for image segmentation
<https://www.researchgate.net/publication/32973889> Optimal thresholding for image segmentati
on#:~:text=Image%20thresholding%20is%20a%20technique.analyze%20and%20is%20more%
20meaningful.

Applying this to the Gaussian density, applying logarithms and simplifying, we obtain the quadratic equation.

$$AT^2 + BT + C = 0$$

With the following coefficients:

$$A = \sigma_1^2 - \sigma_2^2$$

$$B = 2(\mu_1\sigma_2^2 - \mu_2\sigma_1^2)$$

$$C = \mu_2^2\sigma_1^2 - \mu_1^2\sigma_2^2 + 2\sigma_1^2\sigma_2^2 \ln \frac{\sigma_2 P_1}{\sigma_1 P_2}$$

As we know, the quadratic equation has two different solutions, which indicates that there is a possibility that two thresholds are needed to obtain the most optimal solution.

If the standard deviations are equal, then a single threshold will be enough because the coefficient A of the quadratic equation would be zero. So, solving the equation, we are left with the following solution:

$$T = \frac{C'}{B'} = \frac{\mu_2^2\sigma_1^2 - \mu_1^2\sigma_2^2 + 2\sigma_1^2\sigma_2^2 \ln \frac{\sigma_2 P_1}{\sigma_1 P_2}}{2(\mu_1\sigma_2^2 - \mu_2\sigma_1^2)}$$

Simplifying:

$$T = \frac{\mu_1 + \mu_2}{2} + \frac{\sigma^2}{\mu_2 - \mu_1} \ln \frac{P_1}{P_2}$$

As we can see, there are 2 options in which the solution to the optimal threshold is the average of the mean values of the brightness levels:

- When $P_1=P_2$
- When $\sigma^2 = 0$.

The latter is very difficult to control since chance plays a large role in the variance of the brightness, but decreasing the variance is a good way to make the solution less dependent on the probabilities P_1 and P_2 .

3.2.1.1.3. Limit-based thresholding

One of the most important aspects in finding the most suitable threshold is to be able to reliably differentiate the different peaks of the given histogram. Based on the above, if we want to obtain narrow, high peaks with deep valleys, it is convenient to look at the pixels that are close to the boundary between object and background to eliminate the pixels that do not give us information but distort the histogram and eliminate the clear dependence of the histogram on the relative size of the object with respect to the background.

If we take pixels that are close to or on the boundary, the histogram is very likely to have the same peaks as those described in the previous paragraph. Furthermore, it is very likely that the probability of a pixel being from the object, or the background is approximately the same, so the symmetry of the peaks is considerably improved. Finally, the use of the gradient or Laplacian causes the valleys to be deeper. This is because a property of the gradient/Laplacian methods is that they have a mean value of 0 when they are on the edge so one would expect the valleys to be very close to zero due to this property. In addition, the use of the Laplacian provides us with the information whether a pixel is on the object or on the background.

The main problem with this method described above is the assumption that the boundary between object and background is known.

The gradient at any point (x,y) and the Laplacian are described in equations mentioned above. These two quantities can be used to form a three-level image as follows:

$$s(x, y) = \begin{cases} 0 & \text{if } \nabla f < T \\ + & \text{if } \nabla f \geq T \text{ and } \nabla^2 f \geq 0 \\ - & \text{if } \nabla f \geq T \text{ and } \nabla^2 f < 0 \end{cases}$$

where 0, + and - represent three different grey levels, T is the threshold, and the gradient and Laplacian are calculated for all (x,y) points. For a dark object on a light background an image s(x,y) is produced in which all pixels that are not on the boundary are marked with zero, those on the dark side of the border are marked with a + and those on the light side with a -. The transition between an object point and a background point is always characterized in the transition line as (+,-) or (-,+).

3.2.1.1.4. Adaptive Thresholding

While all other conventional methods have an equal global intensity threshold for all pixels, this adaptive method dynamically modifies the threshold for each pixel. This more advanced version of thresholding can accommodate the threshold so that better results are obtained in the face of abrupt changes in the image. This technique usually takes as input a monochrome or color image and in its simplest implementation generates as output a binary image representing the segmentation.

The threshold must be calculated at each pixel and there are two different methods for this purpose:

- Chow and Kaneko's method.

This strategy divides the image into overlapping regions. For each region, a threshold is estimated based on the histogram of that region. This estimation involves the minimization of an objective error function based on certain assumptions made about the image and in some cases fails to converge to a satisfactory value.

As the regions overlap each other, values from neighboring regions are used to interpolate the threshold for the regions where a threshold has not been resolved. Once a threshold is

obtained for each region, a threshold is calculated for each pixel of the image by interpolating the threshold values of the surrounding regions.

The main disadvantage of Chow and Kanenko's method is that it is computationally very expensive and not practical for real-time applications.

·Local Thresholding

It is based on calculating the threshold for each pixel statistically using the intensity values of neighboring pixels. The type of statistical function to use depends largely on the input image. The most common and simplest functions to use are the mean, median and average of minimum and maximum values. Using local thresholds based on this strategy it is possible to treat images whose histograms do not show clearly differentiated gray levels.

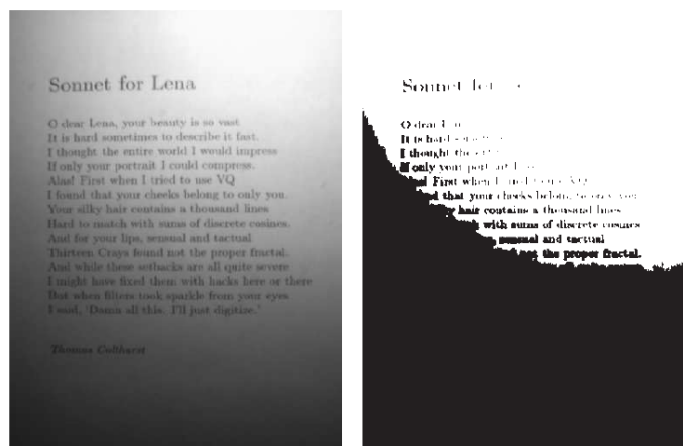


Image 27 and 28

We can see in this image how a global thresholding does not work properly when the illumination gradient of the original image is very large.

Much better results can be obtained using the local thresholding approach. Recall that this strategy assumes that illumination is more likely to be uniform in small areas of the image. However, these areas must be large enough to contain

pixels corresponding to the object and the background. In this way, the mean (or other statistical operator) of the intensities of the pixels neighboring the point we are thresholding at that time may be an appropriate threshold to separate the object from the background.

(Local Thresholding and introduction) (Eugene Woods, R., Gonzalez, R., Image processing, Pearson, New York, 2018, p. 761-763)

(Chow and Kaneko) Santiago Gonzalez Benitez. End of degree project. (p. 36-37)
<https://repositorio.upct.es/bitstream/handle/10317/228/pfc1820.pdf?sequence=1>

3.2.2. Growth regions [4]

As the name implies, region growing is a procedure by which pixels or groups of pixels are grouped into larger regions. The simplest form of this technique is known as pixel summation. This strategy starts with a given number n of seed pixels that form the initial regions R_1, R_2, \dots, R_n . From the seed pixels the regions start to grow by annexing neighboring pixels that meet some predefined similarity criteria (gray level, texture, color...).

The choice of a similarity criterion depends not only on the problem we are dealing with but also on the type of data available. In geographic analysis applications based on satellite images, the use of color is very important. Solving problems of this type would be much more difficult with monochromatic images. Unfortunately, in most cases we do not have multispectral image data and, typically, analysis using region-based techniques has to be carried out using descriptors based on spatial and intensity properties from a single image.

On the other hand, the exclusive use of descriptors for region-based segmentation can lead to erroneous results if connectivity information is not used during the growth process. Grouping the pixels present in an image according to their descriptors (e.g., intensity) regardless of their position relative to each other within the image will not produce any satisfactory results.

Another problem is the formulation of a stopping criterion for the algorithm. In principle, a region should stop growing when there are no more pixels that satisfy its similarity criterion.

However, it is often useful to add an additional criterion that considers the size of the region or its shape. Descriptors such as intensity or texture do not consider the "history" of the region, so it is often advisable to compare the descriptor of the candidate pixel to join the region with the descriptor of the whole region (e.g., the intensity of the candidate pixel and the average intensity of all pixels forming the region). The use of these additional criteria is always conditioned by the a priori information available.

3.2.2.1. Formulation

Let R be the representation of the entire region of an image, segmentation can be thought of as a process that divides R into n subregions:

$$\cdot \bigcup_{i=1}^n R_i = R$$

$\cdot R_i$ is a connected region for all i .

$$\cdot R_i \cap R_j = \emptyset, \forall i, j, i \neq j$$

$$\cdot P(R_i) = TRUE \text{ for } i = 1, 2, 3, \dots$$

$$\cdot P(R_i \cap R_j) = FALSE \text{ for } i \text{ different from } j$$

The first condition tells us that the union of all regions from 1 to n must form the complete region

The second condition tells us that any of the subregions is continuous throughout it. There are no intermediate points that belong to a different region.

The third condition tells us that the intersection between two different subregions must be the empty set.

The fourth condition deals with the conditions that the pixels of a segmented region must satisfy.

The fifth condition tells us that the indicated regions are different in the logical sense that P tells us.

3.2.2.2 Examples

We have the following image in which we have two seed points from 2 different regions.

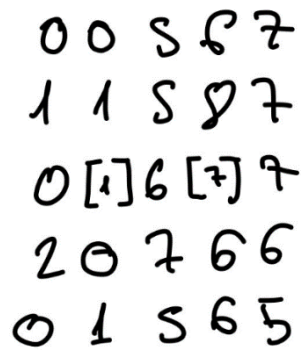


Image 29

The first T we are going to use is 3, which will give us the following result:

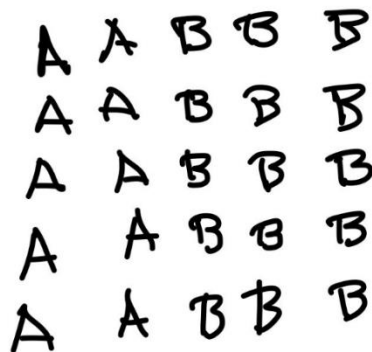


Image 30

And we are also going to use a T of value 8:

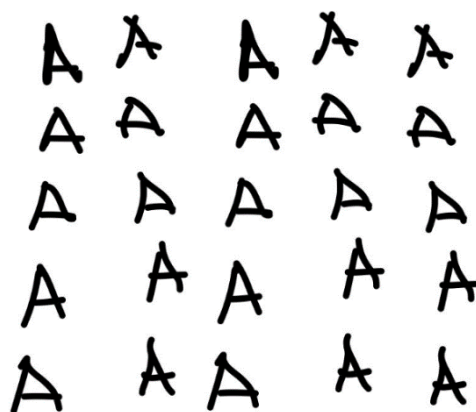


Image 31

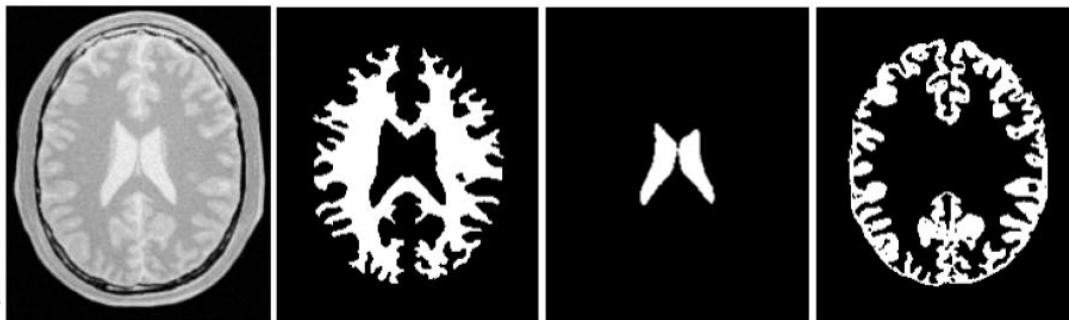
As we can see in images 30 and 31, it is very important the correct choice of a threshold since in image 30 we can see how we can perfectly differentiate the 2 regions of the proposed image and in image 31, with the threshold $T=8$, the result tells us that it is the same region and that there are not two different ones as we know there were.

Another example is with the following image:

$$I(\mathbf{X}) \in [\text{lower}, \text{upper}]$$

Structure	Seed Index	Lower	Upper
White matter	(60, 116)	150	180
Ventricle	(81, 112)	210	250
Gray matter	(107, 69)	180	210

Image 32



The most important features for this method of growing regions by pixel clustering are the determination of the seed points and the type of clustering property.

(Introduction growing regions) (Eugene Woods, R., Gonzalez, R., Image processing, Pearson, New York, 2018, p. 764-768)

4 Representation and description

When you finish the segmentation process what you have is only the raw pixels, which constitute the contour of a region or the whole region, depending on whether you have done a threshold detection or a grown region.

Within this process we must decide how we are going to represent this data, as an outline or as a region. It all depends on what the purpose of our work is.

If we are only interested in the outer parts, we will represent the data as contours. However, if we are interested in the internal data of the object such as texture or structure, we will represent it as a region.

There will also be times when we need to combine both representations. An example might be in character recognition applications.

4.1. Diagrams of representation

The segmentation techniques studied above produce raw data in the form of pixels of a contour or region. Although sometimes these data are used in this manner to obtain descriptors, the common practice is to use schemes that compact the data into representations that are more useful in calculating descriptors.

The techniques can be classified as follows:

- Chain codes
- Polygonal approximations
- Signatures
- Contour sides
- Skeleton of a region

4.1.1. Chain codes

Chain codes are used to represent a contour by means of a connected succession of segments of specified length and direction. Typically, this representation is based on segments of connectivity 4 or 8. The address of each segment is encoded using a numbering scheme as shown in the image below.

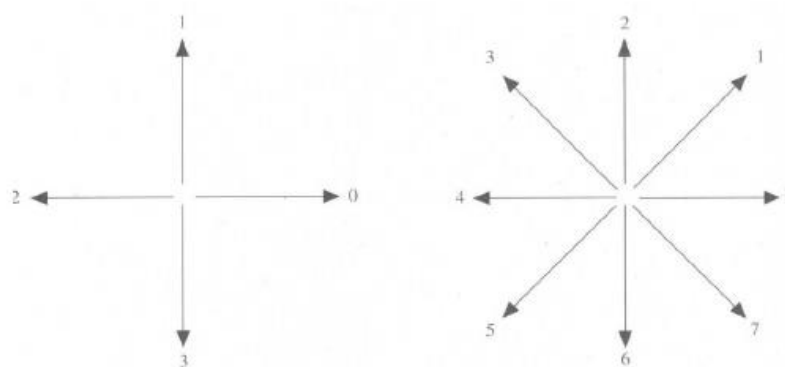


Image 33

Because digital images are acquired on a grid with equal horizontal and vertical spacing, a string code could be generated by following the contour, for example, in a clockwise direction and assigning a direction to the segments connecting each pair of pixels.

This method is unacceptable for two main reasons:

1. The resulting chain is generally very long.
2. A small perturbation due to noise or imperfect segmentation causes changes in the code, usually not related to the contour.

One of the solutions that can be used is to resample the contour by selecting a larger grid spacing. Then as the contour is traversed, one point of the contour is assigned to each node of the new grid. The new contour obtained can be represented by a connectivity code 4 or 8. The starting point is defined at will. Obviously, the accuracy of the code obtained depends on the spacing selected for the new sampling.

This technique is exemplified in Figure 34 and Figure 35 shows the resulting string codes.

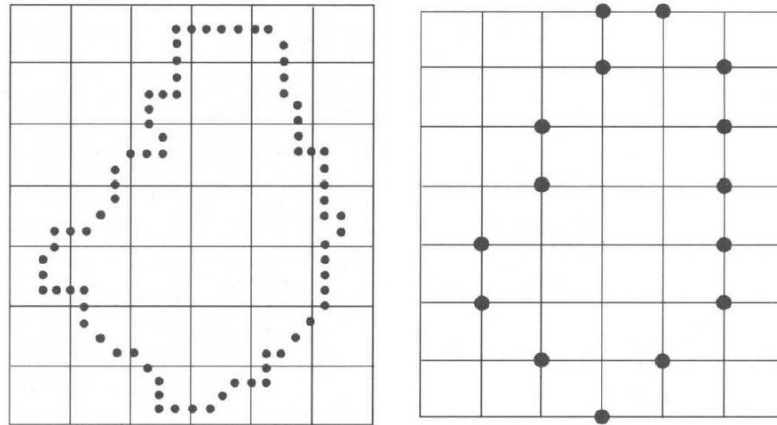


Image 34

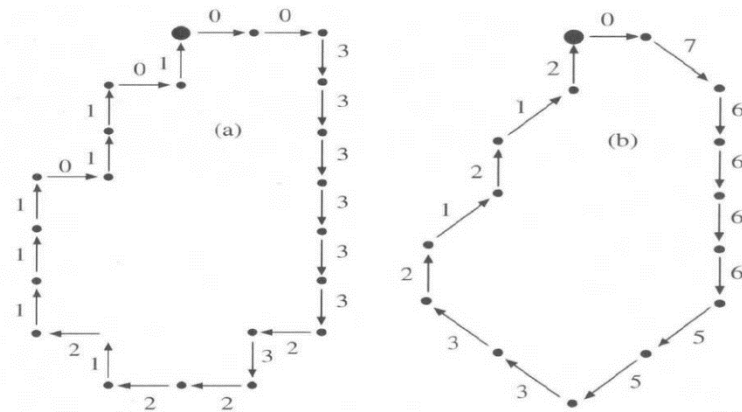


Image 35

(Ernesto Bribiesca, A new chain code, Pattern Recognition, 1999, p. 235-251)

Anonimo.Codigo de cadena https://hmong.es/wiki/Chain_code

https://www.fceia.unr.edu.ar/dip/Representacion_Descriptores.pdf (last time:27/08/2022)

4.1.2. Polygonal approximations [7]

A digital contour can be treated with arbitrary accuracy by a polygon. For a closed curve, the approximation is exact when the number of sides of the polygon is equal to the number of points of the contour, such that each pair of adjacent points defines one side of the polygon. In practice, the objective of a polygonal approximation is to capture the essence of the shape of the contour, with a polygon with as few sides as possible.

There are several polygonal approximation techniques of moderate complexity that are suitable for image processing applications.

One such technique is to find a polygon of minimum perimeter. Suppose that the contour is enclosed in a set of concatenated cells. These cells can be thought of as defining two walls corresponding to the outer and inner edges of the array of cells, and the contour can be thought of as a rubber band contained between the walls. If the rubber band is allowed to shrink, it would take the shape of the image 36 producing a polygon of minimum perimeter that conforms to the geometry established by the succession of cells. If each cell spans only one point on the contour, the error in each cell between the original contour and the rubber band approximation would be, at most, $\sqrt{2}d$, where d is the distance between pixels.

This error can be halved by making each cell centered on its corresponding pixel.

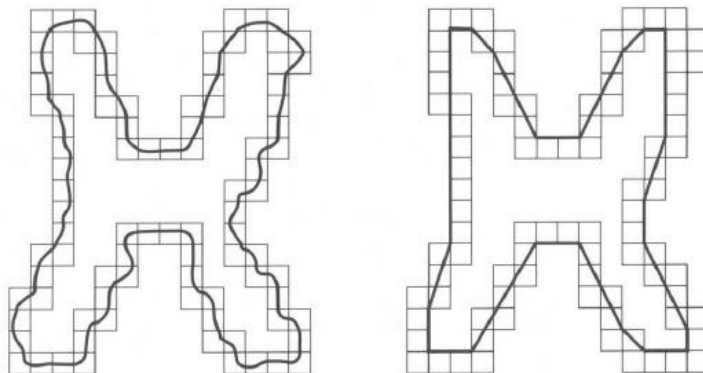


Image 36

One method for dividing contour sides consists of successively subdividing the side into two parts until a given criterion is satisfied. For example, a requirement could be that the maximum perpendicular distance from one side of the contour to the line joining its two ends does not exceed a previously established threshold.

If it does, the farthest point becomes a vertex, thus subdividing the side into two sub-sides. This approach has the advantage of looking for prominent inflection points. For a closed contour, the best points to start with are usually the two points furthest apart on the contour. Image 37 exemplifies this procedure, it shows the contour of an object (a), the subdivision of this contour (b), point (c) has the largest normal distance from the top to line ab. The normal distance from the lower side to the line ab corresponds to the point d. The result of using this procedure with a threshold equal to 0.25 times the length of the line ab is shown in (c). The final result is shown in (d).

The normal distance from the lower side to the line ab corresponds to the point d. The result of using this procedure with a threshold equal to 0.25 times the length of the line ab is shown in (c). The final result is shown in (d).

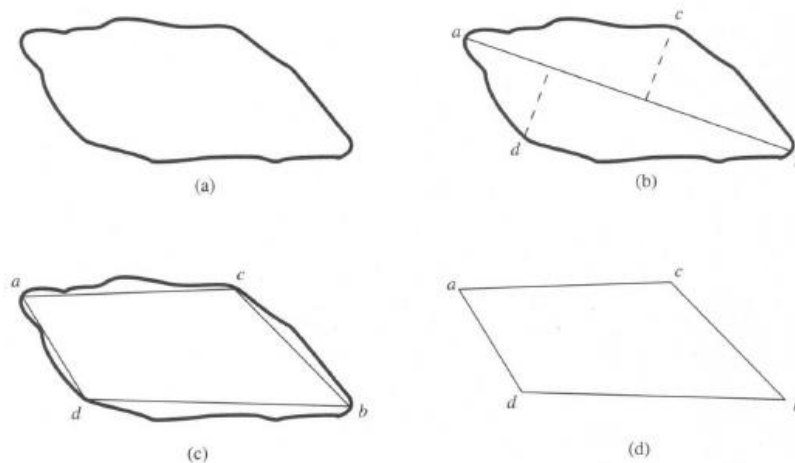


Image 37

(Eugene Woods, R., Gonzalez, R., Image processing, Pearson, New York,2018,p.821-826)

<http://imgbiblio.vaneduc.edu.ar/fulltext/files/TC111475.pdf> (last time:27/08/2022)

https://www.fceia.unr.edu.ar/dip/Representacion_Descriptores.pdf (last time:27/08/2022)

4.1.3. Signatures

A signature is a one-dimensional functional representation of a contour and can be generated in several ways. One of the simplest is to represent the distance from the center to the contour as a function of angle. Signatures generated in this way do not vary with translation but depend on rotation and change of scale. It can be normalized with respect to rotation by finding a way to select the same starting point to generate the signature to be independent of object orientation. One method could be to choose the point farthest from the center of the object as the starting point. Another way would be to choose a point on the object's major axis. Regarding the scaling variation, it can be normalized by scaling all functions in such a way that it always covers the range of values $[0,1]$.

Image 38 shows an example of an object signature.

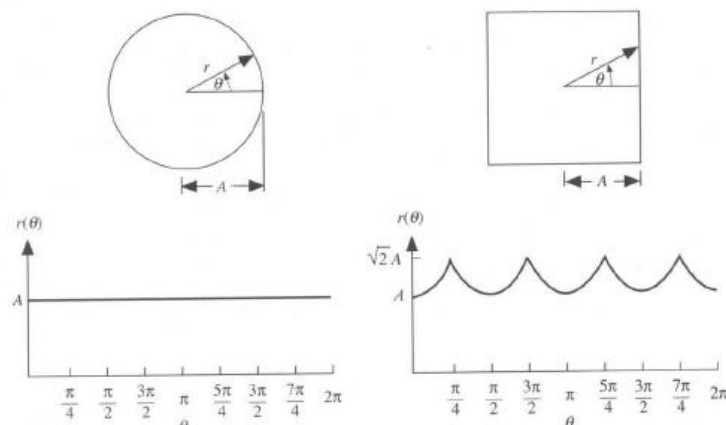


Image 38

<https://www.studocu.com/es/document/universidad-publica-de-navarra/procesado-digital-de-imagen/tema-4-parte-2-apuntes-4/5898509> (last time:27/8/2022)

(Eugene Woods, R., Gonzalez, R., Image processing, Pearson, New York,2018,p.826-828)

https://www.fceia.unr.edu.ar/dip/Representacion_Descriptores.pdf (last time:27/08/2022)

4.1.4. Skeleton of a region [9]

An important approach to represent the structural shape of a planar region is to reduce it to a graph. In this reduction, the skeleton of the region can be obtained by means of a reduction algorithm, called skeletonization.

The skeleton of a region can be defined based on the Mean Axis Transform (MAT) proposed by [Blum, 1967].

The MAT of a region R with edge B is defined as, "For each point p of R , its nearest neighbor in B is found. If p has more than one of these neighbors, it is said to belong to the middle (skeleton) axis of R ."

The concept of nearest neighbor depends on the definition of distance used, and therefore the results of an MAT operation are influenced by this choice. Figure 40 shows the process of obtaining the MAT.

Although the MAT of a region provides an instinctually appealing skeleton, direct implementation of that definition is usually computationally prohibitive. The implementation potentially involves calculating the distance from each interior point to each contour point of a region. Many algorithms have been proposed to improve computational efficiency while attempting to produce a representation of the median axis of a region.

Typically, these are reduction algorithms that iteratively remove points from the margin of a region subject to the constraints that the removal of these points:

- 1) does not remove extreme points
- 2) does not break continuity
- 3) does not cause excessive erosion in the region.

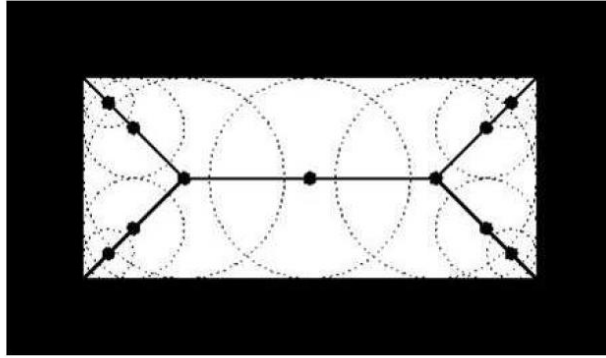


Image 39

A binary region reduction algorithm is presented. It is assumed that the region points have value 1 and the background points have value 0. The method consists of successive passes of two basic steps applied to the contour points of a given region, a contour point being any pixel with value 1 that has at least one 8-neighbor with value 0.

With reference to the definition of 8-neighbor, step 1 marks a contour point p for deletion if the following conditions are satisfied:

1. $2 \leq N(p_1) \leq 6$
2. $S(p_1) = 1$
3. $p_2 \cdot p_4 \cdot p_6 = 0$
4. $p_8 \cdot p_4 \cdot p_6 = 0$

where $N(p_1)$ is the number of non-zero neighbors of p_1 , that is:

$$N(p_1) = p_2 + p_3 + \dots + p_9$$

and $S(p_1)$ is the number of 0-1 transitions in the ordered sequence $p_2, p_3, \dots, p_8, p_9$. For example, $N(p_1) = 4$ and $S(p_1) = 3$ in the following subpicture.

$$\begin{array}{cccccc}
 p_9 & p_2 & p_3 & 0 & 0 & 1 \\
 p_8 & p_1 & p_4 & 1 & p_1 & 0 \\
 p_7 & p_6 & p_5 & 1 & 0 & 1
 \end{array}$$

In step 2, conditions a) and b) remain the same but conditions c) and d) change to:

$$3. p_2 \cdot p_4 \cdot p_8 = 0$$

$$4. p_8 \cdot p_2 \cdot p_6 = 0$$

Step 1 applies to every pixel on the edge of the binary region under consideration. If one or more of conditions c) and d) are violated, the value of the point in question is not changed.

If all conditions are satisfied, the point is marked for deletion.

However, the point is not deleted until all points on the edge have been processed.

This delay prevents the structure of the dates from being changed during the execution of the algorithm.

(Eugene Woods, R., Gonzalez, R., Image processing, Pearson, New York,2018,p.828-831)

4.2. Contour descriptor

4.2.1. Simple descriptor

Contour length: is the number of pixels along the contour. It is the number of horizontal components + the number of vertical components + $\sqrt{2}$ times the number of diagonal components.

Diameter of a contour: Let B be the contour of an object. Its diameter is defined as:

$$\text{Diam}(B) = \max[D(p_i, p_j)]$$

where $D(p_i, p_j)$ is a measure of distance.

Corners: These are places on the contour where the curvature $k(B)$ is not bounded:

$$|k(B)|^2 = \left(\frac{d^2y}{d^2B}\right)^2 + \left(\frac{d^2x}{d^2B}\right)^2$$

4.2.2. Shape numbers

As explained, the first difference of a string-coded contour depends on the starting point. The shape number of a contour based on the 4-way code is defined as the minimum modulus integer of the first difference.

The order n of a shape number is defined as the number of digits of its representation. Furthermore, n is even for a closed contour and its value limits the number of possible shapes.

Image 4.1 shows all possible shapes of order 4, 6 and 8.

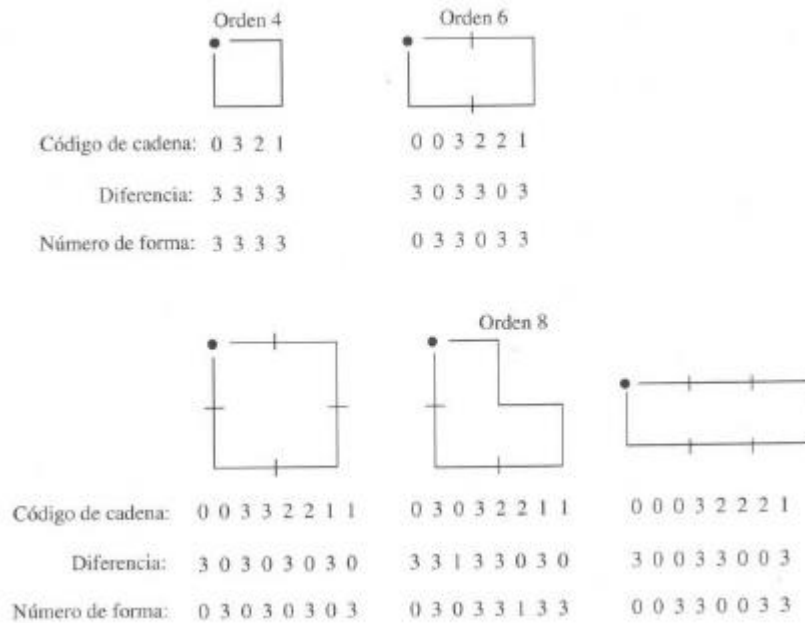


Image 40

Although the first difference of a string code is independent of rotation, in general the encoded contour depends on the orientation of the grid used. One way to avoid this is:

The major axis of a contour is the line segment joining the two points farthest apart.

The minor axis is perpendicular to the major axis and of such a length that a rectangle can be formed that exactly contains the contour (minimum rectangle).

The ratio of the major axis to the minor axis is called the contour eccentricity, and the rectangle thus formed is called the basic rectangle

Ekeeda Youtue channel. Shape Numbers -Representation and Description – Image processing
<https://www.youtube.com/watch?v=KRDRK6gaJuw>

Bribiesca E. and Guzmán A., Shape Description and Shape Similarity Measurement for Two Dimensional Regions, Proceedings of The 4th International Conference on Pattern Recognition, pp. 608-612, Kyoto, Japan (1978).

4.2.3. Region descriptor

Area: is defined as the number of pixels enclosed by its contour. (A_o).

Perimeter: The length of the contour of the region. (P).

Density: This parameter, also called Circularity, is defined as,

$$C = \frac{P^2}{A_o}$$

Rectangularity: defined as the ratio of the total area of the region over the area of the basic rectangle of the region,

$$R = \frac{A_o}{A_R}$$

(Eugene Woods, R., Gonzalez, R., Image processing, Pearson, New York,2018,p.840-843)

4.2.4. Topological descriptors

Topological properties are useful for global descriptions of regions. If a topological descriptor is defined as the number of holes in the region, this number may vary if the region is doubled or split. But if we stretch the region this descriptor will not be affected.

Another useful property as a topological descriptor is the number of connected components.

The Number of Hollows H and the Number of Connected Components C of a figure are used to define the Euler Number:

$$E = H - C$$

This number is also a topological property. In image 42 this number is shown.

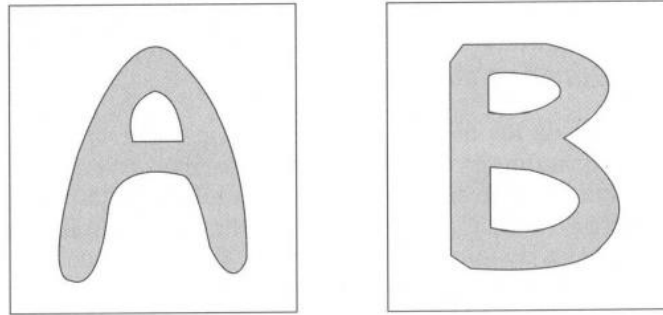


Image 41

As we can see in the case of the first image $H=1$ and $C=1$ so $E=0$.
 However, in the second case, $H=2$ and $C=1$ so $E=1$.

The regions represented by line segments (polygonal networks) have a simple interpretation in terms of Euler's number.

Sometimes it is important to classify the interior regions of the lattice into faces and holes. Representing the number of vertices by W , the number of edges by Q and the number of faces by F gives the relation, called Euler's formula:

$$E = H - C = W - Q + F$$

The example is shown in the following image:

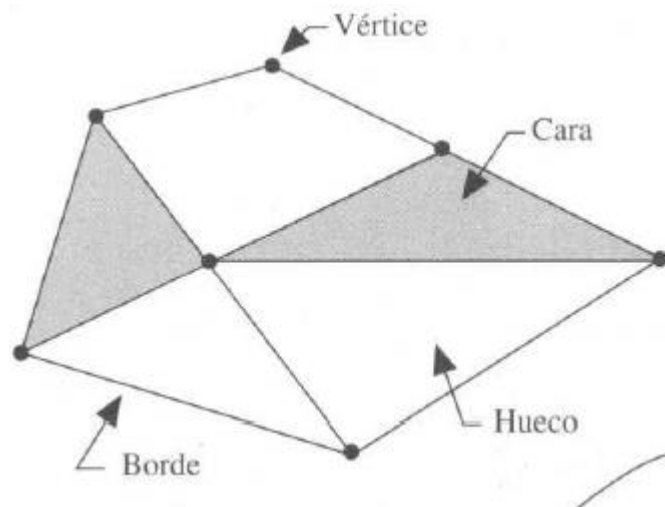


Image 42

As we see in the image: $W=7$, $Q=11$ and $F=2$. Then, applying Euler's formula we obtain $E=-2$.

(Eugene Woods, R., Gonzalez, R., Image processing, Pearson, New York, 2018, p.843-846)

https://www.fceia.unr.edu.ar/dip/Representacion_Descriptores.pdf (last time: 27/8/2022)

Recognition and interpretation

We will conclude the study of digital image processing by developing several techniques that make possible their recognition and interpretation. This part is mainly related to applications that require automated image analysis. Image analysis is a process of discovering, identifying and understanding patterns that are relevant to the performance of an image-based job. One of the main goals of computer image analysis is to give a machine, in some sense, the ability to approximate, similar to that of humans. For example, in a system for automatic image reading of typed documents, the patterns of interest are alphanumeric characters, and the goal is to achieve accurate character recognition that is as close as possible to the superb ability exhibited by humans to perform such tasks. Thus, an automatic image analysis system should be able to offer varying degrees of intelligence.

The concept of intelligence is somewhat vague, particularly with reference to a machine. However, it is not difficult to express the concept of the different types of behavior generally associated with intelligence. Several characteristics quickly come to mind:

1. The ability to extract the information of interest, separating it from a set of irrelevant details.
2. The ability to learn from examples and to generalize this knowledge so that it can be applied in new and different circumstances.
3. The ability to make inferences from incomplete information.

5.1. Elements of image analysis

5.1.1. Techniques used

The three most commonly used methods are:

1. Decision-theoretic methods for recognition.
2. Structural methods for recognition.
3. Methods for image interpretation.

Theoretical decision recognition is based on the representation of patterns in vector form and the subsequent search for approximations that allow grouping and assigning these vector patterns to different classes of patterns. The main techniques of decision-theoretic recognition are minimum distance classifiers, correlators, Bayes classifiers and neural networks. In structural recognition, patterns are represented in symbolic form (such as strings and trees), and recognition methods are based on symbol matching or models that treat symbol patterns as sentences of an artificial language.

The interpretation of an image consists of assigning meaning to a set of recognized elements in that image. The main concept underlying the various image interpretation methodologies is the effective organization and use of domain-specific knowledge of a problem.

4. Image enhancement [12]

The main objective of the techniques that will be explained is to process an image in such a way that it is more suitable than the original image for a specific application.

1 Classification

1.1 Methods in the space domain.

Space domain methods are based on the direct manipulation of image pixels. While frequency domain methods are based on the modification of the Fourier transform of an image.

Spatial domain methods are procedures that operate directly on the image pixels.

The image processing functions in the spatial domain can be expressed as:

$$g(x, y) = T[f(x, y)]$$

where $f(x,y)$ is the input image and $g(x,y)$ is the output image. T is an operator acting on f defined over some environment of (x,y) .

The environments are generally defined as square or rectangular subimages centered on the pixel (x,y) . The simplest form of T corresponds to a 1×1 environment. In this case g depends only on the value of f at the point (x,y) , and T becomes a gray level transformation function of the shape:

$$s = T(r)$$

where r and s are variables indicating the gray level of $f(x,y)$ and $g(x,y)$.

-Negative images

Digitized image negatives are useful in medical imaging. The negative of an image is obtained by using the transformation function shown in the following figure

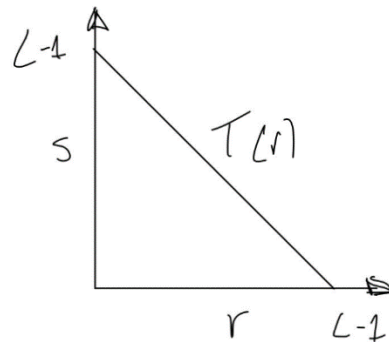


Image 43

- Contrast enhancement

Low contrast images are due to many causes, such as poor illumination, lack of dynamic range of the sensor or incorrect selection of the aperture at the time of capture. The main idea for contrast enhancement is to increase the dynamic range of the gray levels in the image. A typical transformation used for contrast enhancement is shown in the following image.

The location of the points (r_1, s_1) and (r_2, s_2) controls the shape of the transformation function. For example, if $r_1 = s_1$ and $r_2 = s_2$ a linear function is generated that produces no change in gray levels (Image 46). If on the other hand $r_1 = r_2$, $s_1 = 0$ and $s_2 = L-1$ the transformation becomes a threshold function (Threshold) that generates a binary image (Image 47). It is assumed that $r_1 \leq r_2$ and $s_1 \leq s_2$ so that the function is single-valued and monotonically increasing.

Anonimo. <http://www.eie.polyu.edu.hk/~enyhchan/imagee.pdf> (last time:27/8/2022)

Santiago Aja Fernández <https://www.lpi.tel.uva.es/muitic/pim/docus/Realce.pdf> (last time:27/8/2022)

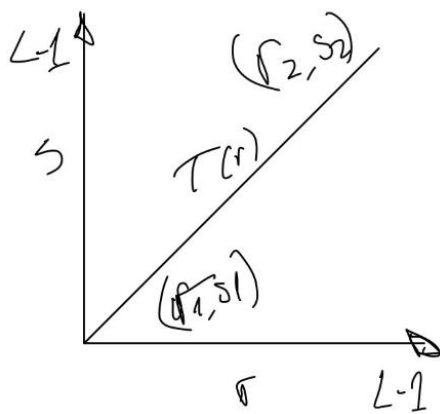


Image 44

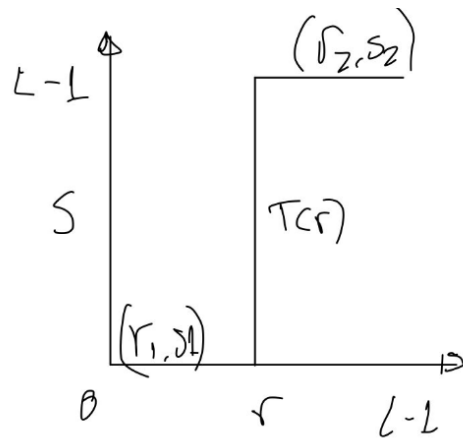


Image 45

- Gray level fractionation

Often it is desired to highlight a specified range of gray levels in an image. Such applications may include the enhancement of ranges such as water bodies in satellite images or flawed X-ray images. Two basic ideas are used for this: The first is to assign a high value to the range of interest and low to the remainder (Image 48). The second is similar except that the image background is preserved (Image 49).

Image 46

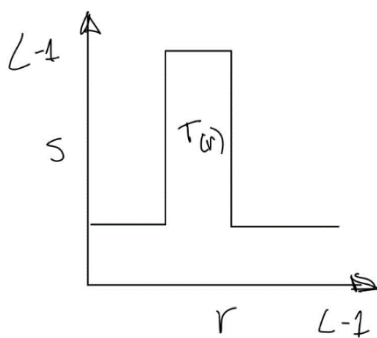
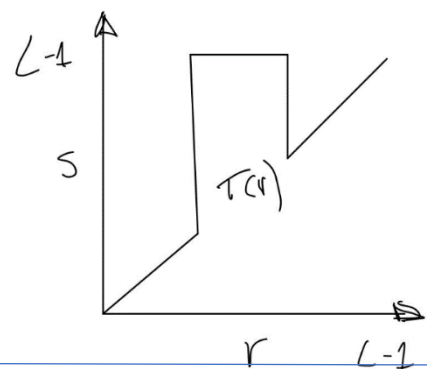


Image 47



Anonimo. <http://www.eie.polyu.edu.hk/~enyhchan/imagee.pdf> (last time:27/8/2022)

Santiago Aja Fernández <https://www.lpi.tel.uva.es/muitic/pim/docus/Realce.pdf> (last time:27/8/2022)

-Histogram

In general, it can be said that the histogram presents the probability of distribution $p(r_k)$ of the different amplitudes or gray levels of the image.

The shape of the histogram allows to highlight certain particularities of the image, such as the type of background, the contrast and in general whether the values of the gray levels are homogeneously distributed or not.

Different images may have gray level distributions that are far from ideal. Typical examples are images acquired under bad conditions, where there are too many shadows or, on the contrary, too many white levels.

·Example

The following image is a photograph that is presented as it was taken (in this case the photographer was more concerned with taking the photograph before the animals left the field of view than with adjusting the equipment).



Image 48

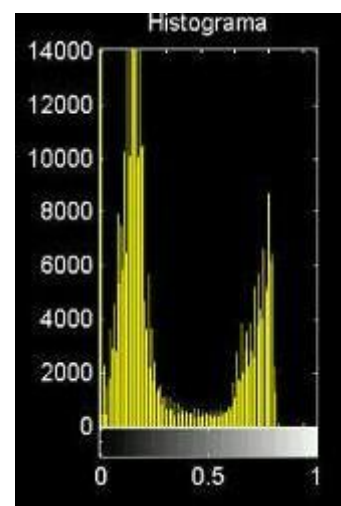


Image 49

Anonimo. <http://www.eie.polyu.edu.hk/~enyhchan/imagee.pdf> (last time:27/8/2022)

Santiago Aja Fernández <https://www.lpi.tel.uva.es/muitic/pim/docus/Realce.pdf> (last time:27/8/2022)

It can be clearly seen that the distribution of gray levels is far from acceptable. In this case, it is clear that the colors of the zebras should be much more clearly contrasted. If the histogram is plotted, it can be seen that it does not show a homogeneous distribution, but shows a bimodal distribution, i.e., there are many dark gray levels and many light levels, with a deficiency of levels in the middle of the range and at the highest levels.

Sometimes it is desirable to have an algorithm that automatically provides reproducible and good quality results. The most commonly used algorithm for these cases is histogram equalization or histogram equalization. If you try to make the distribution as uniform as possible, you will get an "equalized" or equalized image, which is not always close to the image as it should have looked originally.

The following image shows the original image after having followed a 64-level equalization procedure.



Image 50

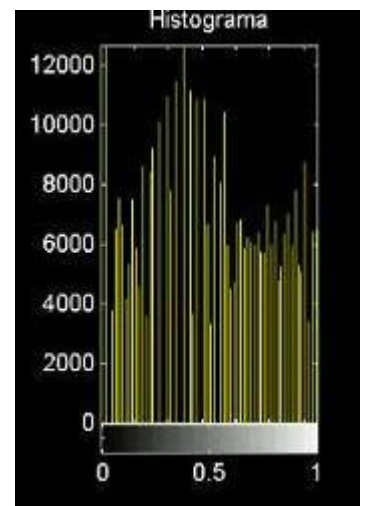


Image 51

- Histogram equalization

The basic principle of this type of operation is the redistribution of the gray levels of the original over all the levels available in the system, following a particular law that determines the type of modification.

Let r be a variable representing the gray levels of the image to be enhanced. Assume that the pixel intensities are continuous quantities that have been normalized to the range $[0,1]$, where $r=0$ is black and $r=1$ represents white.

Let the transformation, $s=T(r)$. There are 2 conditions:

a) $T(r)$ is single-valued and monotonically increasing over the interval $0 \leq r \leq 1$.

b) $0 \leq T(r) \leq 1$ for $0 \leq r \leq 1$

Condition a) preserves the order between black and white of the grayscale. Condition b) ensures an application that is consistent with the range of allowed values for pixel intensity.

The inverse transformation function of s to r is denoted by:

$$r = T^{-1}(s)$$

Assume r and s as continuous random variables in the range $[0,1]$. The original gray levels and their transform can be characterized by their probability density functions $p_r(r)$ and $p_s(s)$.

From elementary probability theory, if $p_r(r)$ and $T(r)$ are known and $T^{-1}(s)$ verifies condition a), then the probability function of the transformed gray levels is:

$$p_s(s) = \left[p_r(r) * \frac{dr}{ds} \right]_{r = T^{-1}(s)}$$

Consider the transformation function:

$$s = T(r) = \int_0^r pr(w)dw$$

This is called the cumulative distribution function of r. From this equation, the derivative image of s with respect to r is:

$$\frac{ds}{dr} = pr(r)$$

Replacing:

$$ps(s) = \left[pr(r) * \frac{dr}{ds} \right] r = \left[\frac{ds}{dr} * \frac{dr}{ds} \right] r = 1$$

which gives a uniform density in the interval of definition of the variable s. This means that when a transformation function equal to the cumulative distribution function is used, an image with uniformly distributed gray levels is produced.

In order to be applied in digital image processing the above concepts must be expressed in discrete form. For the gray levels that constitute the discrete values we have the probabilities:

$$pr(rk) = \frac{nk}{n} \quad 0 \leq rk \leq 1 \quad k = 0, 1, \dots, L - 1$$

where rk is the k-th gray level. nk is the number of pixels with gray level rk and n is the total number of pixels in the image.

For the histogram equalization of an image the continuous relationship seen above is expressed as:

$$sk = T(rk) = \sum_{j=0}^k \frac{nj}{n} = \sum_{j=0}^k pr(rj)$$

Anonimo. <http://www.eie.polyu.edu.hk/~enyhchan/imagee.pdf> (last time:27/8/2022)

Shreenidhi Sudhakar.Histogram equalization.2017 <https://towardsdatascience.com/histogram-equalization-5d1013626e64> (last time:27/5/2022)

-Spatial filtering

The use of spatial masks for image processing is often referred to as spatial filtering, and the masks used are called spatial filters.

Many image enhancement operations are performed on pixel neighborhoods or regions of interest. This is because regions near the pixel in question can provide valuable information about illumination levels and scene details. The use of this information from adjacent pixels is linked to the concept of spatial filtering. The figure below shows a model of the filtering process.

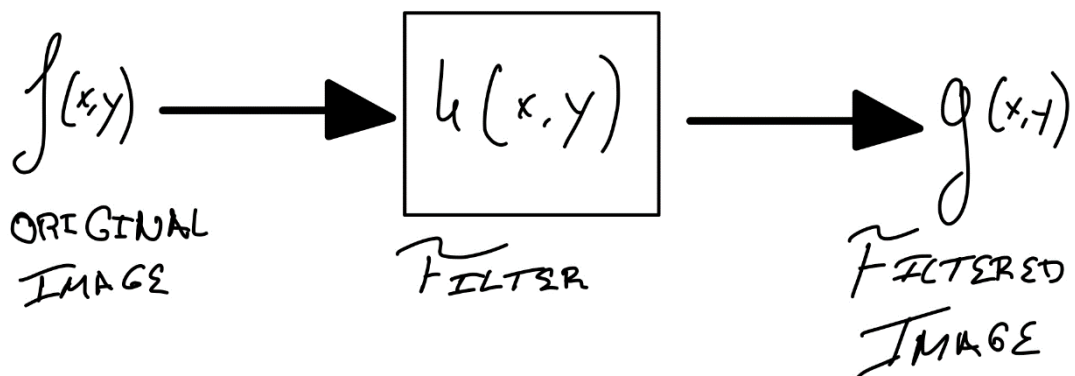


Image 52

There are 3 types of filters:

- Low-pass filter: Attenuate high frequencies.
- High-pass filter: Attenuate low frequencies.
- Band-pass filter: Attenuate low and high frequencies.

1.2. Methods in the frequency domain.

Frequency domain filters are mainly used to remove high or low frequencies from the image, which means to smooth the image, or to enhance or detect edges.

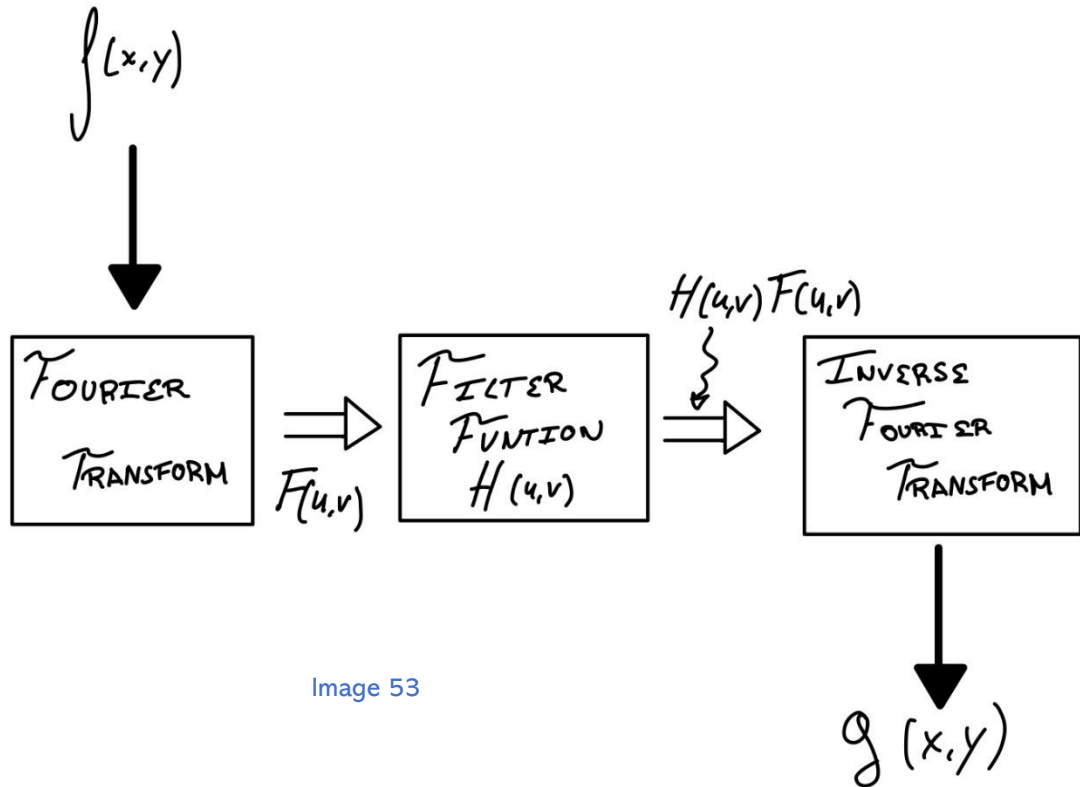


Image 53

The steps to follow, according to the above diagram:

1. Each entry is multiplied by $(-1)^{x+y}$.
2. The image is transformed into its frequency domain using the discrete Fourier transform. $F(u,v)$
3. It is multiplied by a frequency filter. $H(u,v)*F(u,v)$
4. The image is transformed in its spatial domain using the discrete inverse Fourier transform.
5. It is multiplied by $(-1)^{x+y}$.

The types of frequency filters are the same as in the case of spatial filters.

5. Color in images. [13]

The use of Color in image processing is important for two fundamental reasons. First, in automatic image analysis, Color represents a powerful descriptor that often simplifies the identification of an object and its extraction from a scene. Second, in human image analysis, the interest in Color lies in the fact that our eye is able to discern thousands of shades and intensities of Color compared to only a few levels of grey. Color image processing can be divided into two fundamental areas, true or full Color and false or pseudo-Color processing. In the first category the images are acquired through a Color sensor, such as a television camera or a Color scanner. In the second category the problem is to assign a Color level to a certain intensity or group of monochromatic intensities.

The Colors that humans perceive in an object are characterized by the nature of the light reflected by the object. Visible light consists of a relatively narrow band of frequencies of the electromagnetic spectrum as we can see in the following image.

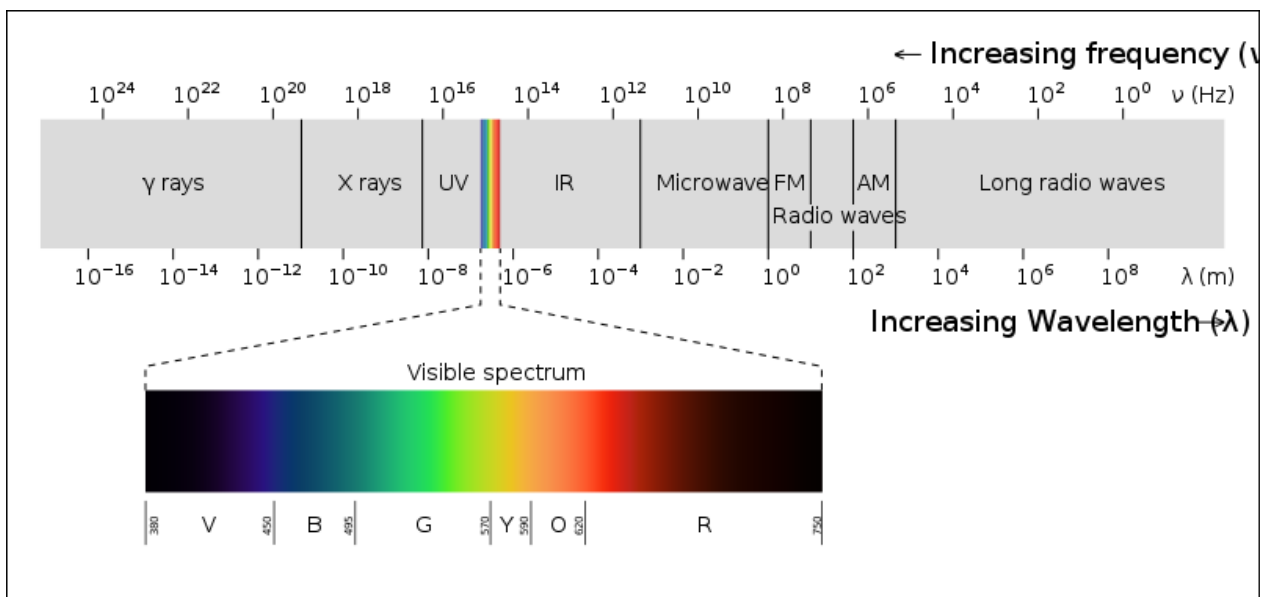


Image 54

Three basic quantities are used to describe the characteristics of a chromatic light source:

-Radiance: The total amount of energy emitted by the light source.

-Luminance: The total amount of energy perceived by the receiver from a light source.

-Brightness: Same meaning as intensity.

The general characteristics used to distinguish one Color from another are:

-Brightness, as we saw, is related to the chromatic notion of intensity.

-Hue is an attribute associated with the dominant wavelength in a mixture of light waves.

-Saturation refers to the relative purity or amount of white light mixed with a hue. Pure Colors are fully saturated. Hue and saturation taken together constitute chromaticity and therefore a Color can be specified by its brightness and chromaticity.

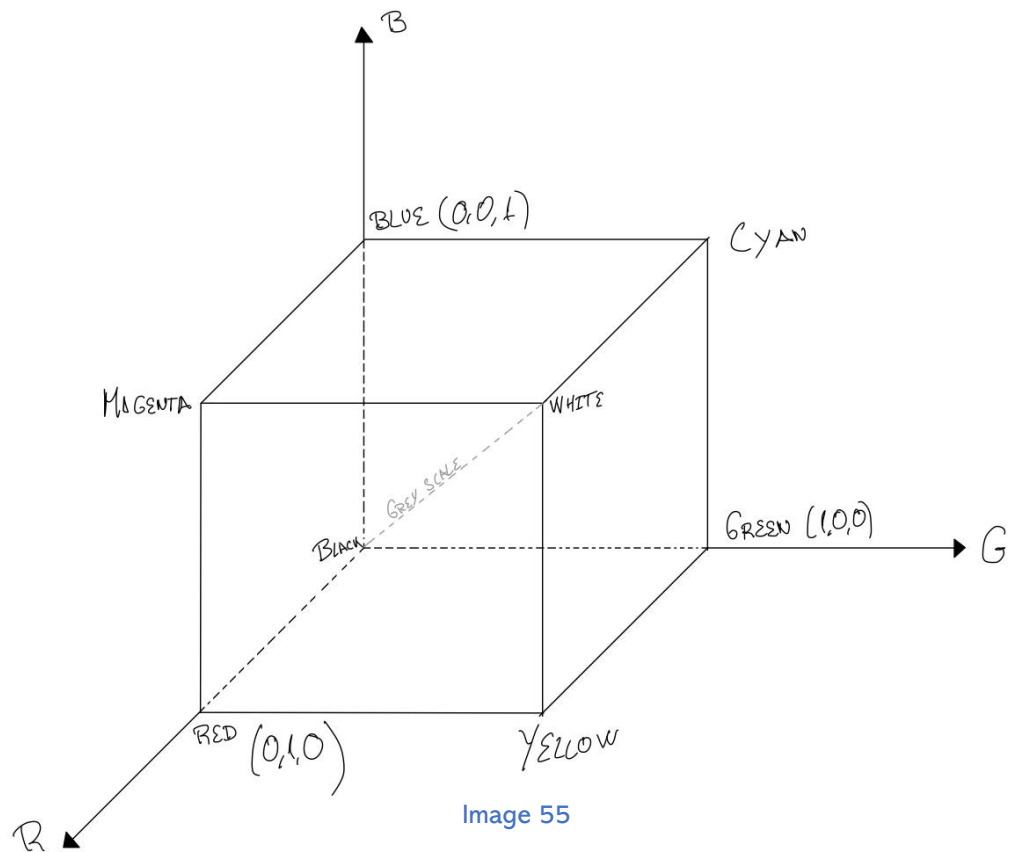
The most commonly used models for Color image processing are RGB, CMYK, YIQ and HSI.

RGB Model

This model is based on a Cartesian coordinate system. The Color subspace of interest is the cube shown in image 57. For convenience it has been assumed that the Color values are normalized so that the cube is unitary. The images in this model consist of three independent image planes, one for each primary Color. At the hardware level, most video cameras used for Color image acquisition use this Color model. One of the best examples of the usefulness of the RGB model is in satellite images, frames. A frame consists of four digital images. Each corresponds to the same scene but captured through a different spectral window.

Two of these correspond to red and green, the other two are in the infrared region of the spectrum. Suppose you want to improve the Color image of a face with a poor Color distribution.

Theoretically, one could apply the histogram equalization technique to each RGB image. Although there might be parts of the image where improvement is obtained, we must consider that as each plane is equalized independently, each one will be modified differently, disturbing the overall appearance of the image.



CMYK model

These three are the secondary light Colors or primary pigment Colors. Most of the devices that deposit Colored pigments on paper, printers, photocopiers, need an input in the CMYK model or a conversion from RGB to CMYK. This conversion is obtained through the following formula:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$K = \min\{C, M, Y\}$$

Where:

C=Cyan

M=Magenta

Y=Yellow

K=Key (Black)

YIQ model

This model is the one used in commercial television broadcasts, mainly because of its transmission efficiency and to maintain compatibility with black and white television standards.

The conversion from RGB to YIQ is done through the formula:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Where:

Y: Has the luminance information.

I: In phase

Q: Quadrature

HSI model

This model owes its usefulness mainly to two basic facts. First, the intensity component is decoupled from the chromatic information contained in the image. Second, the hue and saturation components are intimately related to the way humans perceive Color. Examples of the usefulness of this Color representation model range from automatic design to determine the degree of ripeness of fruits to systems for comparing Color samples or inspecting the quality of Colored products. The conversion formulas between HSI and RGB and vice versa are more complex than in previous models.

$$H = \begin{cases} 0, & \text{if } MAX = MIN \\ (60 \times \frac{G-B}{MAX-MIN} + 360) \bmod 360, & \text{if } MAX = R \\ 60 \times \frac{B-R}{MAX-MIN} + 120, & \text{if } MAX = G \\ 60 \times \frac{R-G}{MAX-MIN} + 240, & \text{if } MAX = B \end{cases}$$

$$L = \frac{1}{2} (MAX + MIN)$$

$$S = \begin{cases} 0, & \text{if } MAX = MIN \\ \frac{MAX-MIN}{MAX+MIN} = \frac{MAX-MIN}{2L}, & \text{if } L \leq \frac{1}{2} \\ \frac{MAX-MIN}{2-(MAX+MIN)} = \frac{MAX-MIN}{2-2L}, & \text{if } L > \frac{1}{2} \end{cases}$$

Where:

H: Hue

MAX: The biggest between R,G y B

S: Saturation

MIN: The smallest between R,G y B

I: Intensity

(Introduction, RGB model and CMYK) Jose Ramon Mejía. Procesamiento digital de imágenes
<http://laurence.com.ar/artes/comun/Apuntes%20procesamiento%20digital%20de%20imagenes.pdf>
(Last time: 20/06/2022)

(YIQ model) Anónimo. <http://dea.unsj.edu.ar/imagenes/recursos/capitulo4.pdf> (Last
time: 20/5/2022)

(HSL model) https://es.wikipedia.org/wiki/Modelo_de_color_HSL#Conversi3n_desde_RGB_a_HSL
(Last time: 20/6/2022)

6. Labelling

In this section we are going to explain all the code that has been used to perform the necessary labelling for further processes.

```
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
import json
import base64
import os.path
from os.path import splitext
from datetime import datetime
import cv2
import numpy as np
import matplotlib.pyplot as plt
import re
import random
import pandas as pd
```

Image 56

Initially the program will consist of a part dedicated exclusively to the import of libraries. These are necessary to be able to use the functions or methods that compose it without having to program each of the functions that we are going to use.

As we can see there are diverse ways to import:

1°- The import mode. In which we import that library and to be able to use the methods that are inside it we will have to use the name of the library to the complete one.

2°. The mode from Import. In this mode the only thing that we import is one of the methods that contains the library.

3°. The import mode.... As. This method imports the complete library in the same way as the first mode with the difference that in this case we will use the text after the "as" to call the method we need.

```
basepath = r"C:\Users\Antonio\Desktop\MIS DOCS\Ingenieria\Erasmus 2021-2022\temario Erasmus\TFG"
videos = "Videos"
videopath = os.path.join(basepath,videos)
assert(os.path.isdir(basepath))
assert(os.path.isdir(videopath))

videonamepre = "Todas_las_fotos"
videonameext = "mp4"
datacsv = videonamepre + "_data.csv"
videoname = videonamepre + "." + videonameext
video = os.path.join(videopath,videoname)
assert(os.path.isfile(video))
```

Image 57

First, we create a base route that will be used to create or check routes later on. In that path is where folders with videos, photos and Excel will be stored.

The `os.path.join` function is a function that adds another part (2nd argument) to an already established path (1st argument).

The `os.path.isdir` function checks if the path we are putting as argument exists or not. If it exists, it generates a `TRUE` response and otherwise it generates a `FALSE` output.

The `assert` function generates an interrupt in case its argument is `FALSE`. That is why we combine the `os.path.isdir` function with `assert`, because in case the path specified as argument in the `os.path.isdir` function does not exist, it will generate a `FALSE` and a program interrupt will be generated.

Finally, we note that we also have the `os.path.isfile` function, which works the same as the `os.path.isdir` function but instead of checking if the path/directory exists, it checks the existence of that file in the specified path. The argument of this function is the path where the video is located together with the name of this, as we can observe that before performing this function an `os.path.join` is performed to join the name of the video together with the path/directory where the video is located (`videopath`).

```
picnames = ["cap", "adap", "RJ45", "Boc", "Ganc", "Grey", "Black", "Red", "Big", "Small", "Medium"]
picstats = [False, False, False, False, False, False, False, False, False, False, False]
picspath = os.path.join(basepath, "Fotos")
assert(os.path.isdir(picspath))
```

Image 58

Initially, we are going to introduce into a matrix all the characteristics that we are going to modify later, and we are going to initialize all of them to `FALSE`.

Subsequently we are going to create a directory of photos where these will be saved as we will see later.

And finally, we are going to verify that this directory exists with a function explained previously.

```

def save_entry(name):
    global picnames
    global picstats

    arr = os.listdir(picspath)

    df = pd.DataFrame()
    datalist = []

    if os.path.isfile(os.path.join(os.path.join(picspath, datacsv))):
        df=pd.read_csv(os.path.join(os.path.join(picspath, datacsv)))
        for index, row in df.iterrows():
            datalist.append({'name': row['name'], picnames[0]: row[picnames[0]], picnames[1]: row[picnames[1]], picnames[2]: row[
dataitem = {'name': name, picnames[0]: picstats[0], picnames[1]: picstats[1], picnames[2]: picstats[2], picnames[3]: picstats[
datalist.append(dataitem)
df = pd.DataFrame(datalist)
df.to_csv(os.path.join(picspath, datacsv))

return

```

Image 59

The `os.listdir()` function generates a list of all files in the specified directory.

The function `pd.DataFrame()` creates a table.

The function `pd.read_csv()` loads the csv data in table previously created with the function `dataframe` from a specified file.

For `index, row in df.iterrows()`, this part of the code iterates through all the rows of an array. So, inside we put a `datalist.append`, which adds to the `datalist` the elements that we tell it to.

7. Neural networks

1 Introduction

The field of image processing is continually evolving. Over the past few years there has been a significant increase in interest in fields such as image morphology, artificial neural networks, color and/or grayscale image processing, image data understanding, image recognition, and knowledge-based analysis systems.

A large number of image recognition applications can be found in the market, to highlight a few, we could mention:

- Face and facial expression recognition.
- Signature recognition.
- Character recognition.

1.1 Face and facial expression recognition

For humans, the easiest way to recognize people is through their face, as it has unique features such as distance between eyes, nose width, chin shape, cheekbones, mouth shape, etc.

Numerous works related to face recognition can be found [14] [15] [16].

In this section, we briefly describe how a computational security system based on face recognition has been developed [14]. The system studied consists of a computer connected to a webcam placed on a support, which in turn will have a base where the face is placed so that the camera can capture the image, once the image is captured, it is sent to the computer to proceed to the study of the image.

The program that studies the image, allows segmenting the captured images to obtain only the images of the face and thus discard non-relevant information from the image, then proceeds to the decomposition of the images to then apply the technique of Principal Component Analysis technique, to finally recognize the image.

The segmentation of the images is based on the projections of the derivatives of the rows and columns of the image values.

Once the image has been segmented, the image is decomposed into four sub-images containing main details of the face.

Finally, the Principal Component Analysis is performed, which consists of the collection of images of faces of several people, which are then combined and converted into a matrix. The vectors that make up the matrix can be properly combined to adequately reconstruct any facial image in the set.

1.2. Signature recognition

Currently, signature recognition and security has become a priority issue to ensure the protection of systems against voluntary misuse, so it is essential to perform a process of identification and verification of the identity of each person in a secure, but at the same time simple and natural way.

Biometrics facilitates the identification of each individual univocally by measuring different personal and non-transferable characteristics. These individual characteristics are classified into physical characteristics (fingerprint, iris, retina, etc.); and behavioral characteristics (speaking, writing, signing, typing, etc.).

In [17] biometric identification systems are used to identify the signatures of each individual. For this purpose, an initial multimodal biometric database was used, in which there were 3750 signatures corresponding to 75 individuals. Half of them were true signatures half of them were true and the other half false. In this paper we discuss two diverse ways of generating models depending about the perimeter points obtained from the analysis of the blob structures that make up the signature image. A blob represents a continuous and identifiable structure in an image because it is of a different color from the background.

In the first form the blobs are sorted by size and in the second by their natural reading order, i.e., from top to bottom and from left to right.

The conclusions of this research showed that better results are obtained with top-down and left-to-right blob sorting models.

1.3. Character recognition

An original and simple algorithm for region-based recognition of handwritten characters, in particular the five vowels, is presented in [18]. The algorithm is based on a region-based strategy in the image domain. Preliminary evaluation results of the proposed algorithm, on a small database, showed a high recognition rate for the different vowels. In order to place the proposed algorithm in the context of the general

within the context of the general architecture of a handwritten character recognition system, a description of the functions and tasks performed by each of the elements of the architecture is given.

the functions and tasks performed by each of the elements that make up the architecture, followed by a brief explanation of the design and its implementation. On this work it was concluded that handwriting recognition requires the study of a large number of variables. The results of applying this algorithm yielded a vowel recognition rate of over 97%, which makes the proposed algorithm a good candidate for the

the proposed algorithm a good candidate for a real application. It is important to mention that the algorithm was tested with a small database, and it would then remain to validate the results with a larger database.

From the systems mentioned above, it can be observed that in all of them it is necessary to extract the characteristics of the images to be studied and that these characteristics depend on the type of image being processed.

2 Automatic learning

Process automation is an idea that has accompanied humans throughout history. The idea that a process can be carried out with as little human assistance as possible optimizes costs, increases performance, and even makes it possible to carry out tasks that were previously impossible in order to improve the quality of human life.

The advent of computers made it possible to maximize the automation of tasks in specific areas, in which the basis is the performance of large quantities of arithmetic operations. In these tasks, the proficiency in computers is enormous, having far surpassed human capacity.

However, there are other tasks for which humans have always shown better performance, for example in image recognition tasks, text interpretation, artistic skills, among others.

Furthermore, it is clear that the human brain has desirable properties when it comes to artificially address tasks that are not simple to imitate with classic computer programs, such as the ability to be tolerant to system failures, incomplete, inconsistent or noisy data, and above all, the ability to adapt and learn.

It is based on these limitations that a new paradigm in computational systems, machine learning, was developed. The underlying idea is to design computational algorithms that "learn" to perform a task, i.e., that without programming a successive series of rules, and given a task T , a metric P that evaluates the performance in that task, and an experience E , the program can best perform the task T , as measured by P , using its experience E .

2.1. Learning methods

The methods by which a system "learns" to perform a task can be roughly classified into three categories:

- Supervised learning: In supervised learning, the system learns to perform a task based on data labeled by human experts. After the system returns a result, the result is compared with the label, considered the correct answer, and from that comparison the system learns.

-Reinforcement learning: In reinforcement learning, unlike supervised learning, the information provided to the system is not the correct answer, but information about whether it has made a mistake or not.

-Unsupervised learning: In unsupervised learning, no extra information is provided to the system in addition to the input data and the task it has to perform is to extract relevant features about the data.

The method we will apply in this work is supervised learning, since we have labeled data that we want the model to learn.

Damián Jorge Match. Redes neuronales: Conceptos basicos y aplicaciones.

https://www.frro.utn.edu.ar/repositorio/catedras/quimica/5_anio/orientadora1/monograias/match-redesneuronales.pdf (last time: 1/7/2022)

Ruben Rodríguez Abril <https://lamaquinaoraculo.com/computacion/aprendizaje-de-redes-neuronales/> (last time:1/7/2022)

Carlos Delgado Youtube Channel <https://www.youtube.com/watch?v=qu8vBb8wIFY> (last time:

2.2. Training and evaluation sets

In supervised learning, one has a set of data associated with their respective labels. This is usually divided into two subsets, called training and evaluation, respectively.

This is done in order to train the model based on the first set and report the performance of the model based on a metric measured on the second set.

· Training set

This one is used to train the neural network.

·Evaluation set

This one is used to evaluate that the neural network is working correctly.

3 What is an artificial neural network?

An artificial neural network is a conglomerate of artificial neurons, linked through an architecture of synaptic connections, which has the ability to learn to perform a task, i.e., it is a machine learning tool. Two relevant components appear then, which deserve more detail.

3.1. Artificial Neuron

As mentioned above, the artificial neuron embedded in machine learning networks is a simple dynamic system. Unlike an artificial modeling of a biological neuron, in this artificial neuron it is neither necessary nor desirable to model all the essential elements for the life of the neuron, but only to represent those that play a crucial part in the storage and transmission of information between these units. From the simplest point of view, a neuron is a processing unit that receives input signals and, depending on these signals, emits an output signal (image 63).

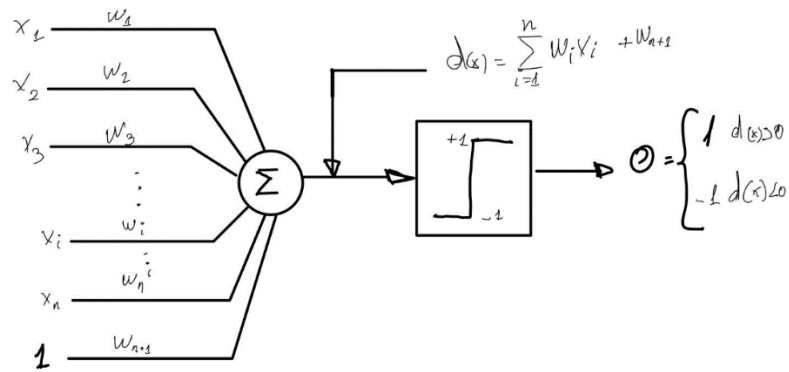


Image 60

To simplify the analysis let us assume that these input and output signals are binary. Then, the behavior of the neuron can be modeled as follows:

$$\zeta^n = g\left(f\left(\vec{\zeta}\right)\right) = \begin{cases} 0 & \text{if } f\left(\vec{\zeta}\right) < 0 \\ 1 & \text{if } f\left(\vec{\zeta}\right) \geq 0 \end{cases}$$

The function $f(x)$ represents the behavior of the neuron against inhibitory or excitatory signals sent by other neurons. We call the function $g(x)$ an activation function since it reflects the behavior of the output for a value of $f(x)$.

Two things can be observed from here: first, that the behavior of the artificial neuron is quite simple. Second, that the behavior of the neuron depends essentially on being connected to other neurons.

An efficient way to aggregate the input signals is by means of a function $f\left(\vec{\zeta}\right)$ linear:

$$f\left(\vec{\zeta}\right) = \sum_{i=0}^{n-1} w_i \zeta^i$$

Where:

$\zeta^0 = 0$ and w_0 is an independent term called bias.

This provides simplicity in the calculation and allows us to modify the w_i weights to give more importance to some inputs than others.

<https://www.xeridia.com/blog/redes-neuronales-artificiales-que-son-y-como-se-entrenan-parte-i>
(last time:2/7/2022)

<https://www.unir.net/ingenieria/revista/redes-neuronales-artificiales/> (last time:2/7/2022)

3.2. Connection architecture

If we analyze the above two equations, we can notice that some things have been tacitly assumed:

- Neurons are not self-connected (their output does not function as input to the same neuron).
- In principle, all other neurons connect to this neuron (assuming that there are n neurons in total).

This is providing us with information on how these neurons are connected. The complete information of the connections of each neuron in the network is what is known as the Connections Architecture. It should be noted that when n is large, the complexity of the architecture can increase dramatically, especially if we assume that not all neurons are connected to all other neurons. Broadly speaking, we could classify neural network architectures into two types, depending on how the information flow in the network is produced:

- Feed-Forward network, which are characterized by the fact that the graph that represents them does not present cycles, that is, the information flows in a defined direction, from a beginning to an end.
- Recurrent Networks, which allow cyclic behavior in the total set or in a subset of neurons.

In our problem we will be particularly interested in architectures of the first type.

<http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/t8neuronales.pdf> (last time:2/7/2022)

<https://interactivechaos.com/es/manual/tutorial-de-machine-learning/arquitectura-de-redes-neuronales#:~:text=El%20concepto%20de%20arquitectura%20referida,en%20la%20que%20son%20entrenadas.> (Last time: 2/7/2022)

3.3. Feed-Forward Networks

We are going to analyze the general structure of Feed-Forward networks, in which the learning process is particularly simple to understand. A Feed-Forward network has the particularity that its component neurons can be characterized as input neurons, hidden neurons and output neurons. Most simply, in these models the neurons are organized in layers, where each neuron in a layer has input connections with neurons in the layer immediately preceding it and output connections with the layer immediately following it. There are exceptions within these models, such as routed connections within the same layer or connections that "skip" one or more layers and connect layers that are not consecutive. The simplest type of layered feed-forward networks is the multilayer perceptron, which was originally studied in detail by Rosenblatt (1961), and whose mathematical modeling is as follows:

$$\zeta_i^j = g \left(\sum_{k=0}^{n_{i-1}} w_{i,k}^j \zeta_{i-1}^k \right)$$

Where:

- ζ_i^j is the output of neuron j in layer i .
- n_{i-1} is the number of neurons in the $i-1$ layer.
- $w_{i,k}^j$ is the weight of the connection between neuron ζ_i^j in layer i and neuron ζ_{i-1}^k in layer $i-1$.

Image 65 shows a schematic of the architecture of a 3-layer perceptron (note that the input layer is not considered). The units of the input layer have the sole function of feeding the network with input features. Then come one or more hidden layers or intermediate layers, followed by the output layer where the computed result is read.

It is clear from the figure that the network is directional, has no loops and no connections between neurons in the same layer.

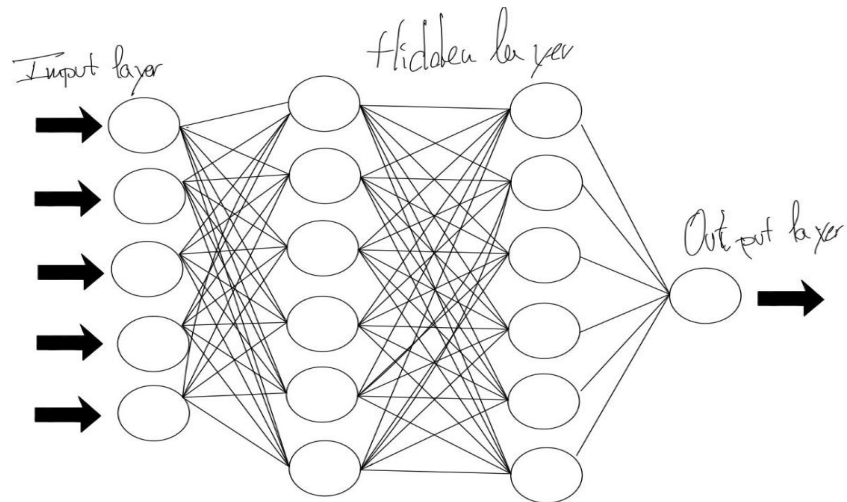


Image 61

<https://www.inf.utfsm.cl/~hallende/bajadas/Redes/Cap3-ANN-2007.pdf> (last time:3/7/2022)

<https://www.linkedin.com/pulse/red-neuronal-feedforward-diego-lópez-g-/?originalSubdomain=es>
(last time:3/7/2022)

Network learning

Neural networks have the ability to learn to perform a specific task. This is done by learning the weights w_i^j of the connections. In order to perform this

4.1. Activation function.

The dependence of the cost function on the parameters must be differentiable or at least Lipschitz continuous.

An activation function must be sought that responds correctly to the inputs of that layer, and that satisfies the conditions of continuity and differentiability. Some of the most common functions that satisfy these conditions are:

·ReLU:

$$g(x) = \max(0, x)$$

·Sigmoid:

$$g(x) = \frac{1}{1+e^{-x}}$$

·Tangent hyperbolic:

$$g(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

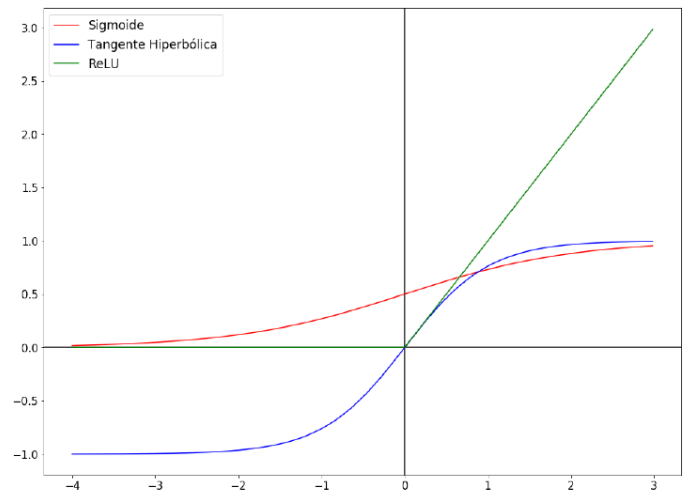


Image 62

Of these functions, the most numerically stable, which generates fewer gradient extinction problems, and which has empirically shown better results in imaging problems is the ReLU function.

Example

In this example, we are going to see how we should do an artificial neural network. This example has been taken from the following website:

<https://www.aprendemachinlearning.com/clasificacion-de-imagenes-en-python/>

1°. Import libraries

The first step that we can see is the import of libraries that are necessary for the correct execution i.e. numpy, keras or matplotlib...

```
import numpy as np
import os
import re
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
import keras
from keras.utils import to_categorical
from keras.models import Sequential,Input,Model
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras.layers.normalization import BatchNormalization
from keras.layers.advanced_activations import LeakyReLU
```

Image 63

2°. Load the images

The second step, as we can see in the following image, is, using this process `plt.imread(filepath)`, load the 70,000 images into memory in an array, so it may take several minutes and will consume some of your computer's RAM.

```
dirname = os.path.join(os.getcwd(), 'sportimages')
imgpath = dirname + os.sep

images = []
directories = []
dircount = []
prevRoot=""
cant=0

print("leyendo imagenes de ",imgpath)

for root, dirnames, filenames in os.walk(imgpath):
    for filename in filenames:
        if re.search("\.(jpg|jpeg|png|bmp|tiff)$", filename):
            cant=cant+1
            filepath = os.path.join(root, filename)
            image = plt.imread(filepath)
            images.append(image)
            b = "Leyendo.." + str(cant)
            print (b, end="\n")
            if prevRoot !=root:
                print(root, cant)
                prevRoot=root
                directories.append(root)
                dircount.append(cant)
                cant=0
    dircount.append(cant)

dircount = dircount[1:]
dircount[0]=dircount[0]+1
print('Directorios leidos:',len(directories))
print("Imagenes en cada directorio", dircount)
print('suma Total de imagenes en subdirs:',sum(dircount))
```

Image 64

3°. Create tags and classes.

In the third step, the program will create the labels in the array called labels and it will give values from 0 to 9 to each sport. The program do this because it is mandatory to use the supervised algorithm and indicate that when a sport image is loaded in the network, it corresponds to "label 6". And with that information, the network when is training, will adjust the weights of the neurons.

Then it has to convert the labels and images into numpy array with `np.array()`.

```
labels=[]
indice=0
for cantidad in dircount:
    for i in range(cantidad):
        labels.append(indice)
        indice=indice+1
print("Cantidad etiquetas creadas: ",len(labels))

deportes=[]
indice=0
for directorio in directories:
    name = directorio.split(os.sep)
    print(indice , name[len(name)-1])
    deportes.append(name[len(name)-1])
    indice=indice+1

y = np.array(labels)
X = np.array(images, dtype=np.uint8) #convierto de lista a numpy

# Find the unique numbers from the train labels
classes = np.unique(y)
nClasses = len(classes)
print('Total number of outputs : ', nClasses)
print('Output classes : ', classes)
```

Image 65

4°. Create Training and Test, Validation and Preprocessing sets.

Note the "shape" of the arrays: the arrays will be 21×28 because is the size of the images the program is using and by 3 because the 3 refers to the 3 color channels that each image has: RGB (red, green, blue) which has values from 0 to 255.

It preprocess the pixel values and normalize them to have a value between 0 and 1, so it has to divide the RGB values by 255.

In addition, it do the "One-Hot encoding" with `to_categorical()` which refers to convert the labels. The way the program is converting the labels is the following: If it has a soccer image, convert the 6 is has in the label to an output like this (0 0 0 0 0 0 0 0 0 1 0 0 0 0 0).

This is because this is how neural networks work best to classify and corresponds to an output layer of the 10-neuron neural network.

In the last part of this step, it split the data in 80-20. 80% to test and the rest for training. This process is done it with “train_test_split()”. And the training is also split it with the same proportion to get a validation subset.

```
train_X, test_X, train_Y, test_Y = train_test_split(X, y, test_size=0.2)
print('Training data shape : ', train_X.shape, train_Y.shape)
print('Testing data shape : ', test_X.shape, test_Y.shape)

train_X = train_X.astype('float32')
test_X = test_X.astype('float32')
train_X = train_X / 255.
test_X = test_X / 255.

# Change the labels from categorical to one-hot encoding
train_Y_one_hot = to_categorical(train_Y)
test_Y_one_hot = to_categorical(test_Y)

# Display the change for category label using one-hot encoding
print('Original label:', train_Y[0])
print('After conversion to one-hot:', train_Y_one_hot[0])

train_X, valid_X, train_label, valid_label = train_test_split(train_X, train_Y_one_hot, test_size=0.2, random_state=42)
print(train_X.shape, valid_X.shape, train_label.shape, valid_label.shape)
```

Image 66

5°. Create the network

Now it use Keras to create the Convolutional Neural Network. These would be the steps:

- Declare 3 "constants":
 - the initial value of the learning rate INIT_LR
 - number of “epochs” and
 - batch size of images to process “batch size” (loaded in memory).
- Create a first layer of neurons "Convolutional 2-Dimensional Conv2D() , where the 21x28x3 images will enter.
- Apply 32 filters (kernel) of size 3×3 that detect certain features of the image (e.g., vertical lines).
- Use the LeakyReLU function as activation of the neurons.

Do a MaxPooling (of 2×2) that reduces the incoming image from 21×28 to half, (11×14) keeping the "unique" features detected by each kernel.

- To avoid overfitting, add a technique called Dropout.
- "Flatten" Flatten() the 32 filters and create a layer of 32 "traditional" Dense() neurons.
- And finalize the output layer with 10 neurons with Softmax activation, to match the "hot encoding" did it before.
- Then compile the network with "sport_model.compile()" and assign an optimizer (in this case called Adagrad).

```
INIT_LR = 1e-3
epochs = 6
batch_size = 64

sport_model = Sequential()
sport_model.add(Conv2D(32, kernel_size=(3, 3), activation='linear', padding='same', input_shape=(21, 28, 3)))
sport_model.add(LeakyReLU(alpha=0.1))
sport_model.add(MaxPooling2D((2, 2), padding='same'))
sport_model.add(Dropout(0.5))

sport_model.add(Flatten())
sport_model.add(Dense(32, activation='linear'))
sport_model.add(LeakyReLU(alpha=0.1))
sport_model.add(Dropout(0.5))
sport_model.add(Dense(nClasses, activation='softmax'))

sport_model.summary()

sport_model.compile(loss=keras.losses.categorical_crossentropy, optimizer=keras.optimizers.Adagrad(lr=INIT_LR))
```

Image 67

6°. Train the CNN

With this line `sport_model.fit()` will start the training and validation of the machine.

Finally, save the trained network `sport_model.save()` in a h5py file format, as it will be able to use it in the future without doing the same process again.

```
sport_train_dropout = sport_model.fit(train_X, train_label, batch_size=batch_size, epochs=epochs, verbose=1, validation_data=(valid_X, valid_label))
sport_model.save("sports_mnist.h5py")
```

Image 68

7°. Results obtained

Now that the network has been trained, it is time to test it with the set of Test images that was separated at the beginning and that are samples that have never been "seen" by the machine.

```
test_eval = sport_model.evaluate(test_X, test_Y_one_hot, verbose=1)
print('Test loss:', test_eval[0])
print('Test accuracy:', test_eval[1])
```

Image 69

```
Test loss: 0.6687967825782881
```

```
Test accuracy: 0.8409179307662388
```

Image 70

8. REFERENCES AND BIBLIOGRAFY

[14] Sandra María Palacios. Sistema de reconocimiento de rostros. Universidad peruana de ciencias aplicadas. Archivo electrónico
[http://www.cibertec.edu.pe/RepositorioAPS/0/13/JER/PARTICIPACIONENCONGRESOS/Reconocimiento-](http://www.cibertec.edu.pe/RepositorioAPS/0/13/JER/PARTICIPACIONENCONGRESOS/Reconocimiento-Rostros.PDF)

Rostros.PDF

[15] Enrique Cabello Pardos. Técnicas de reconocimiento facial mediante redes neuronales. Tesis Doctoral. Archivo electrónico
<http://oa.upm.es/215/1/10200404.pdf>

[16] Luis Blázquez Pérez. Reconocimiento facial basado en puntos característicos de la cara en entornos no controlados. Archivo electrónico
http://atvs.ii.uam.es/seminars/PFC_Luis_Blazquez.pdf

[17] Juan J. Igarza, Amalur Colau, Iñaki Goirizelaia, Inmaculada Hernández, Raúl Méndez. Universidad del País Vasco. Sistema de verificación de firmas off-line basado en modelos ocultos de Markov. Archivo electrónico
http://ursi.usc.es/articulos_modernos/articulos_coruna_2003/actas_pdf/SESION%208/S8.%20Aula%202.4/1355-SISTEMA.PDF

[18] Nancy Rangel Galán, Israel Delgado Guerra, Bravo Zúñiga Haydee. Reconocimiento de caracteres manuscritos. Archivo electrónico,
<http://www.encuentra.gob.mx/APF?q=RECONOCIMIENTO%20POR%20REGIONES%20DE%20CARACTERES%20MANUSCRITOS&client=cio>

[19] (Anonymous, Clasificación de imágenes en Python, 2018
<https://www.aprendemachinelearning.com/clasificacion-de-imagenes-en-python/>)

Fundamentals of Digital Image Processing . Anil K. Jain. Prentice Hall.

Digital Image Processing. Gonzalez, Woods

Digital Image Processing and Computer Vision. Robert J. Schalkoff

Advanced Signal Processing Handbook. Edited by Stergios Stergiopoulos.

Hertz, J ., Introduction To The Theory Of Neural Computation.

Mitchell, Tom ,Machine Learning.

Guarnieri, Ricardo A, Enio B Pereira y Sin Chan Chou «Solar radiation forecast using artificial neural networks in South Brazil».

Rosenblatt, Frank, Principles of neurodynamics. perceptrons and the theory of brain mechanisms

Torre, Ana ,Prediction of compression strength of high performance concrete using artificial neural networks

Kingma, Diederik P. y Jimmy Ba Adam: A Method for Stochastic Optimization

Image references:

-From image 10 to 21 and 32.

dea.unsj.edu.ar/imagenes/recursos/capitulopres5.pdf (last time visited: 29/07/22)

-From image 33 to 43.

dea.unsj.edu.ar/imagenes/recursos/capitulopres6.pdf (last time visited: 29/07/22)

-Image 56.

en.wikipedia.org/wiki/Visible_light_communication#/media/File:EM_spectrum.svg (last time visited: 29/07/22)